Avoiding Negative Side Effects by Considering Others

Parand Alizadeh Alamdari[*], **Toryn Q. Klassen**[*†], **Rodrigo Toro Icarte, Sheila A. McIlraith**[†]
Department of Computer Science, University of Toronto, Toronto, Canada
Vector Institute, Toronto, Canada
[†] Schwartz Reisman Institute for Technology and Society, Toronto, Canada
{parand,toryn,rntoro,sheila}@cs.toronto.edu

Abstract

Recent work in AI safety has highlighted that in sequential decision making, objectives are often underspecified or incomplete. This potentially allows the AI agent to make undesirable changes to the world while achieving its given objective. A number of recent papers have proposed avoiding such negative side effects by giving an auxiliary reward to the agent for preserving its own ability to complete tasks or gain reward. We argue that effects on others need to be explicitly considered and provide a formulation that generalizes prior work. We experimentally investigate our approach with RL agents in gridworlds.

1 Introduction

Recent work in AI safety has raised the concern that objective specifications are often underspecified or incomplete. In the absence of some directive or signal, this gives discretion to a Reinforcement Learning (RL) agent to act in whatever manner optimizes its return, neglecting consideration of potential undesirable (negative) side effects that were not explicitly part of the specified objective or avoided through some implicit incentive specific to the learning algorithm. As Amodei et al. [2016] explain, "[F]or an agent operating in a large, multifaceted environment, an objective function that focuses on only one aspect of the environment may implicitly express indifference over other aspects of the environment." Stuart Russell gave the example of tasking a robot to get coffee from a coffee shop and the robot, in its singular commitment to achieving the stated objective, killing all those in the coffee shop that stood between it and the purchase of coffee [Lebans, 2020]. A somewhat more benign example is that of a robot breaking a vase that is on the optimal path between two points [Amodei et al., 2016]. A range of recent works have presented computational techniques for avoiding or learning to avoid negative side effects [e.g., Zhang et al., 2018, Krakovna et al., 2019, Turner et al., 2020, Krakovna et al., 2020, Saisubramanian et al., 2020]. Awareness and avoidance of undesirable side effects is central to safe and robust control of uncertain environments.

Our concern in this paper is with how an RL agent can learn to act safely in the face of a potentially incomplete specification of the objective. Amodei et al. observe that "avoiding side effects can be seen as a proxy for the things we really care about: avoiding negative externalities. If everyone likes a side effect, there's no need to avoid it." In this spirit, we contend that *to act safely an agent should contemplate the impact of its actions on the wellbeing and agency of others in the environment.* We consider negative side effects to be those that impede the future wellbeing or agency of other agents.

Here, we endow RL agents with the ability to consider in their learning the future welfare and continued agency of others in the environment. We do so by augmenting the RL agent's reward with an auxilliary reward that reflects different functions of expected future return of other agents. We contrast this with recent work on side effects that takes into account only how the agent's actions

[*] equal contribution

Workshop on Safe and Robust Control of Uncertain Systems at the 35th Conference on Neural Information Processing Systems (NeurIPS 2021), Online.

will affect its own future abilities [Krakovna et al., 2019, Turner et al., 2020, Krakovna et al., 2020]. Considering other agents' abilities when avoiding side effects was informally discussed by Turner [2019], and investigated in the context of symbolic planning by Klassen and McIlraith [2021]. We also show how controlling the degree to which impact on self versus others factors into the RL agent's learning results in behaviour that ranges from self-centred to self-less. Experiments in gridworld environments illustrate qualitative and quantitative properties of the proposed approach.

2 Problem and approach

We wish to endow RL agents with the ability to consider the future wellbeing and agency of other agents. For the purposes of this study, we consider an environment with a single *acting agent* that learns how to act via RL. Other agents exist within the environment, operating via fixed policies, and only acting after the acting agent has reached a terminating state. We assume that we can neither incentivize nor control these other agents. An evocative example may be to consider university students who share a kitchen environment, and we wish our RL agent – the acting agent, with some conception of what others may typically do in the kitchen — to learn how to act in the kitchen in a manner that is considerate of others who may use the kitchen after the acting agent is done.

We model the environment as a Markov Decision Process (MDP), $\langle S, A, T, r, \gamma \rangle$, where S is a finite set of states, A is a finite set of actions, $T(s_{t+1}|s_t, a_t)$ gives the probability of transitioning to state s_{t+1} when taking action a_t in state s_t, $r : S \times A \times S \to \mathbb{R}$ is the reward function, and γ is the discount factor [Sutton and Barto, 2018]. We distinguish $s_0 \in S$ as the initial state. A *policy* is a (possibly stochastic) mapping from states to actions. Given a policy π, the *value* $V^\pi(s)$ of a state s is the *expected return* of that state, that is, the expected sum of (discounted) rewards that the agent will get by following the policy π starting in s. An optimal policy maximizes the value of every state. We use RL to learn an optimal policy for our acting agent.

To incentivize the acting agent to consider the future wellbeing and agency of others, we augment our acting agent's reward with an auxiliary reward that reflects the impact of its choice of actions on the future agency and wellbeing of others in the environment. To reflect the acting agent's uncertainty about what is good for others, we make use of a distribution over value functions. In particular, suppose that we have a finite set $\mathcal{V}$ of possible value functions $V : S \to \mathbb{R}$, and a probability distribution $\mathsf{P}(V)$ over that set. Note that we don't have to commit to how many agents there are (or what exactly their actions are). It could be that each $V \in \mathcal{V}$ corresponds to a different agent, that the set reflects all possible value functions of a unique agent, or anything in between. Also, each $V \in \mathcal{V}$ could reflect some aggregation of the value functions of all or some of the agents.

We define the augmented reward function as

$$r_{\mathsf{value}}(s, a, s') = \begin{cases} \alpha_1 \cdot r_1(s, a, s') & \text{if } s' \text{ is not terminal} \\ \alpha_1 \cdot r_1(s, a, s') + \gamma \cdot \alpha_2 \cdot F(\mathcal{V}, \mathsf{P}, s') & \text{if } s' \text{ is terminal} \end{cases} \tag{1}$$

where r_1 is the acting agent's individual reward function, and F is some function. The hyperparameters α_1 and α_2, which we call "caring coefficents", are real numbers that determine the degrees to which the individual reward r_1 and the auxiliary reward $F(\mathcal{V}, \mathsf{P}, s')$ contribute to the overall reward.

We consider the following possible different definitions of $F(\mathcal{V}, \mathsf{P}, s')$:

$$\textstyle\sum_{V \in \mathcal{V}} \mathsf{P}(V) \cdot V(s') \qquad \text{expected future return} \tag{2}$$

$$\min_{V \in \mathcal{V}: \mathsf{P}(V) > 0} V(s') \qquad \text{worst-case future return} \tag{3}$$

$$\textstyle\sum_{V \in \mathcal{V}} \mathsf{P}(V) \cdot \min(V(s'), V(s_0)) \qquad \text{penalize negative change} \tag{4}$$

In Eq. (2), $F(\mathcal{V}, \mathsf{P}, s')$ is the expected value of s', given the distribution on value functions.[2] When the discount factor is 1, Eq. (1) with $F(\mathcal{V}, \mathsf{P}, s')$ defined using Eq. (2) is a generalization of the auxiliary reward defined by Krakovna et al. [2020], who assumed that the future value functions were ones of the acting agent, and so depended on the acting agent's own abilities (they also assumed the future value functions corresponded to reward functions that gave non-zero reward only for

[2]Note that future activity does not have to start in exactly the same state at which the acting agent ended. V can be defined so that $V(s')$ gives the expected return of future activity considered over a known distribution of starting states, given that the acting agent ended in s'.

completing a task). See Section 4 for more details on Krakovna et al.'s approach and the relation of our work to it. Meanwhile, Eq. (3) considers the value of s' if the "worst-case" value function from $\mathcal{V}$ (that still has positive probability) is used.

Note that those two reward augmentations may incentivize the acting agent to not only avoid negative side effects, but also to cause "positive side effects" – to help other agents (assuming $\alpha_2 > 0$). To focus on avoiding negative side effects, Krakovna et al. [2020] proposed comparing the state the agent ends up in against a *reference state* (see Section 4.1 for more details), and that is applicable to our approach as well. In Eq. (4), we use one of the simplest possible reference states, the initial state: the auxiliary reward is the lower of $V(s')$ and $V(s_0)$, where s_0 is the initial state. The idea is to decrease the acting agent's reward when it decreases the expected future return, but to *not* increase the acting agent's reward for increasing that same expected return.

We have also explored associating different caring coefficients with different agents, and trying to preserve agents' ability to execute *options* [Sutton et al., 1999]. See Appendix A and Appendix B, respectively.

A complication with our approach is that for some possible reward functions for the acting agent and future value functions, the acting agent may have an incentive to avoid terminating states, to avoid or delay the penalty for negative future return. This incentive would typically be undesirable. However, it can be shown that under some circumstances, the acting agent's optimal policy will be terminating. The proposition below and its proof are similar to [Illanes et al., 2020, Theorem 1].

Proposition 1. *Let $M = \langle S, A, T, r_1, \gamma \rangle$ be an MDP where $\gamma = 1$, the reward function r_1 is negative everywhere, and there exists a terminating policy. Suppose r_{value} is the reward function constructed from r_1 according to Equation 2, using some distribution $\mathsf{P}(V)$. Then any optimal policy for the MDP $M' = \langle S, A, T, r_{\mathsf{value}}, \gamma \rangle$ with the modified reward will terminate with probability 1.*

Proof. Suppose for contradiction that there is an optimal policy π^* for M' that is non-terminating. Then there is some state $s \in S$ so that the probability of reaching a terminal state from s by following π^* is some value $c < 1$. Since rewards are negative everywhere, that means that $V^{\pi^*}(s) = -\infty$. On the other hand, any terminating policy gives a finite value to each state. Since there is a terminating policy for M there is one for M', and so π^* cannot be an optimal policy. □

3 Experiments

In this section, we first compare our approach (using Eq. (2)) with two baselines. We illustrate that by considering others, the acting agent avoids causing negative side effects for them, and in some scenarios, yields positive side effects. Second, we provide a qualitative illustration of optimal behaviours using the different definitions of $F(\mathcal{V}, \mathsf{P}, s')$. Finally, we illustrate the effect of the caring coefficient on the agent's behaviour.

In all the experiments, policies are learned using Q-learning [Watkins and Dayan, 1992]. To aid exposition, we consider very simple distributions over future value functions, in which the acting agent is certain of what the future value function is (or, in Section 3.2, only considers a small number of possibilities).

3.1 The impact of considering others

We compare our method, which is defined in Eq. (2) (with $\alpha_1 = \alpha_2 = 1$), with two reward augmentation baselines: not augmenting the reward, and a method based on Krakovna et al.'s [2020] approach. The Krakovna-style baseline uses the same Eq. (2) to augment the rewards, except that the future value functions considered are always possible future value functions of the *acting agent itself* (as if it were trying to accomplish the tasks of other agents). So if other agents have differing abilities, that is ignored. (Note that this method does not incorporate Krakovna et al.'s [2020] notion of a "reference state" and may incentivize positive side effects in some cases, as our own method does.)

We use a kitchen environment where agents aim to collect different ingredients from the fridge or shelves, and prepare a meal. The agents get -1 reward in each step, until the task is complete. We designed four different scenarios to illustrate properties of our approach. The results are in Table 1.

Method	Salad acting, next	Peanut acting, next	Salt acting, next	Cookies acting, next
Non-augmented reward	0, ∞	0, ∞	0, ∞	0, 0
Based on Krakovna et al.	1, 0	1, ∞	1, 1	1, -2
Our approach [Eq. 2]	1, 0	2, 0	1, 0	1, -2

Table 1: Comparison of reward augmentation methods for acting and subsequent agents. Each row reflects a different method. Each column depicts results for a different experimental scenario. Each entry pair depicts a "step difference" required by the acting agent and the subsequent acting agent (next). The "step difference" is the difference between the number of steps the agent required to execute their policy as compared to what they would have required if they had tried to complete their task from the initial state without considering other agents. ∞ indicates the task was unachievable.

The first experiment (Salad) shows a scenario where the acting agent and next agent have the same abilities, and as such our approach and the Krakovna-style baseline both avoid negative side effects. The next experiments (Peanut and Salt) show that our approach, taking into account differing agent abilities, is sometimes more effective at avoiding negative side effects than either baseline. The last experiment (Cookies) shows how our approach (and the Krakovna-style baseline) can cause positive side effects for the next agent. Each of the experiments is described in more detail below.

In Salad, the acting agent needs to collect the ingredients from the fridge. If it doesn't consider side effects, it doesn't close the fridge and ruins all the remaining ingredients, preventing the next agent from completing its task. By considering future tasks (whether another agent's or its own), the acting agent learns to take an extra step to close the fridge. In Peanut, preparing food contaminates the environment, and for the next agent to cook requires that the environment first be cleaned (taking one step), or disinfected (taking two steps) if the next agent has allergies. Only our approach takes the two extra steps to disinfect the kitchen because it considers that the other agent (unlike itself) has allergies. In Salt, if the acting agent does not put the salt shaker back on the shelves, the next agent can't complete its task. By considering future agents (in the Krakovna-style baseline and our approach) this side effect is avoided. However, the acting agent is tall and may put the salt on the top shelf (making it take longer for the next, shorter, agent to get it) if it considers that the next agent will be itself. Finally, in Cookies, the next agent's task is to bake cookies in the oven. Two steps are required to preheat the oven (turning on the oven and waiting). By considering the future task of the next agent, the acting agent (who was not using the oven) can turn on the oven to start preheating it, and save the next agent two steps.

3.2 Illustration of optimal behaviours under different reward augmentations

Figure 1 illustrates the difference between Equations (2), (3), (4), and the Krakovna-style baseline described in Section 3.1. The goal of the agents is to play with the doll and leave it somewhere in the environment for the next agent, and then exit the environment from their entry point; the agents get -1 reward in each step. There are six agents (circles 1-6 in Figure 1) in the environment with the same goal. They are shown at their individual entry points. Agents enter the environment separately; the acting agent is agent 1. In this scenario, $\alpha_1 = 1$ and $\alpha_2 = 5$. If we augment the acting agent's reward according to Eq. (2) (where the distribution of value functions is a uniform distribution over the optimal value function for each agent w.r.t. the goal of playing with the doll), the optimal policy is to place the doll as close as possible to the majority of the agents. This behaviour is shown by the red line in Figure 1. If we use Eq. (3) the optimal policy is to place the doll so as to minimize the distance to the furthest agent, shown by the blue line. If we use Eq. (4) the optimal policy is to leave the doll where it is (shown by the green line), because moving it causes negative side effects

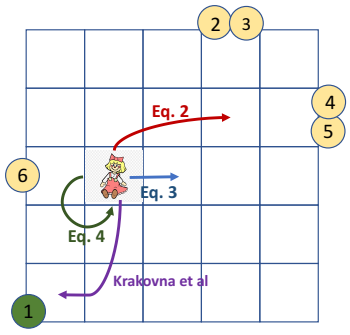


Figure 1: Example behaviour that illustrates different augmentations of the reward function according to Equations (2), (3), (4), and the Krakovna-style baseline.

for agent 6. However if we use the approach based on Krakovna et al., the optimal policy is to leave the doll at agent 1's exit/entry point, so that the doll would be conveniently located for agent 1 if it were to re-enter (shown by the purple line). Finally, if we use non-augmented reward the agent does not have an incentive to place the doll in the environment and leaves with the doll (not shown in figure).

3.3 Varying the caring coefficient

In this experiment, we investigate the effect of choosing different caring coefficients (α_1 and α_2) in Eq. (2) by monitoring the average reward collected by each of the agents in the Craft-World Environment.

Craft-World Environment We consider a Minecraft™ inspired gridworld environment depicted in Figure 2. Agents in this environment use tools and materials to construct artifacts such as boxes. Tools are stored in a toolshed in the upper right corner of the grid environment. Agents enter and exit the environment through doors in the upper left and lower right. They must collect materials and bring them to the factory for assembly. The factory requires a key for entry, and there is only one key, which can only be stored in one of two locations (denoted by K). When considering other agents, the acting agent may elect to place the key in a position that is convenient for others, or may help other agents by anticipating their need for tools or resources and collect them on their behalf.

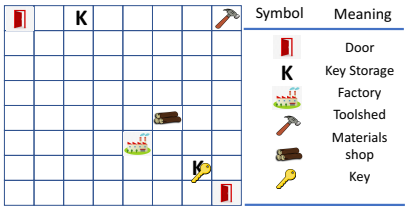


Figure 2: Craft-World Environment.

In the experiment we ran, agents enter at the top left door, tasked with making a box. The first (caring) agent learns a policy following Eq. (2). The second, subsequent acting agent, follows a fixed policy designed to optimize its own reward. Figure 3 shows the reward that each agent gets (after training) as we vary the caring coefficient α_2. It also shows their average. When $\alpha_2 = 0$, the first agent is oblivious to others and exits the environment without returning the key, precluding the second agent from making a box. When $\alpha_2 > 0$, the agent becomes more considerate and returns the key on its way to the exit. As we increase the value of α_2, the first agent is incentivized to help the second agent, eventually (to its detriment) carrying extra materials to the factory for the second agent, garnering negative reward for this hard work and also, interestingly, lowering the average reward of the two agents. Too much caring does not yield maximal reward for the collective!

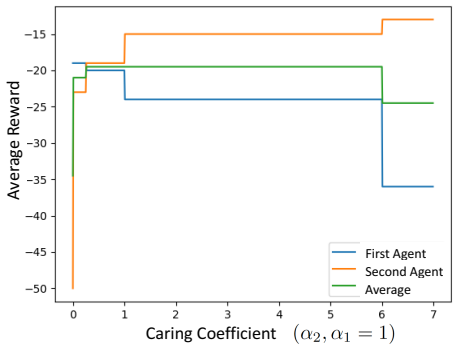


Figure 3: Effect of caring coefficients in the Craft-World environment. Increasing α_2 above 0, at first the agent changes its behaviour with no cost or little cost and this is significantly beneficial for the next agent. However, by increasing α_2 further, the first agent incurs high cost to yield only a small benefit to the second agent.

4 Relation to the future task approach [Krakovna et al., 2020]

In this section we consider in more detail the relation of some of our formulations to the "future task" approach to avoiding side effects from Krakovna et al. [2020].

Krakovna et al. proposed modifying the agent's reward function to add an auxiliary reward based on its own ability to complete possible future tasks. A "task" corresponds to a reward function which gives reward of 1 for reaching a certain goal state, and 0 otherwise. In their simplest definition (not incorporating a *reference state*), the modified reward function was

$$r_K(s,a,s') = \begin{cases} r_1(s,a,s') + \beta(1-\gamma)\sum_i F(i)V_i^*(s') & \text{if } s' \text{ is not terminal} \\ r_1(s,a,s') + \beta\sum_i F(i)V_i^*(s') & \text{if } s' \text{ is terminal} \end{cases}$$

where r_1 is the original reward function, F is a distribution over tasks, V_i^* is the optimal value function for task i (when completed by the single agent itself), and β is a hyperparameter which determines the how much weight is given to future tasks. They interpret $1-\gamma$ (where γ is the discount factor) as the probability that agent will terminate its current task and switch to working on the future task, which leads to the $(1-\gamma)$ factor in the case where s is not a terminal state.

In the case where γ (the discount factor) is 1, that simplifies to

$$r_K(s,a,s') = \begin{cases} r_1(s,a,s') & \text{if } s' \text{ is not terminal} \\ r_1(s,a,s') + \beta \sum_i F(i) V_i^*(s') & \text{if } s' \text{ is terminal} \end{cases}$$

Meanwhile, our Eq. (2) (substituted into Eq. (1)), in the case where $\gamma = 1$, can be rewritten as

$$r_{\mathsf{value}}(s,a,s') = \begin{cases} \alpha_1 \cdot r_1(s,a,s') & \text{if } s' \text{ is not terminal} \\ \alpha_1 \cdot r_1(s,a,s') + \alpha_2 \sum_{V \in \mathcal{V}} \mathsf{P}(V) \cdot V(s') & \text{if } s' \text{ is terminal} \end{cases}$$

Observe that if $\gamma = 1$, $\alpha_1 = 1$, $\alpha_2 = \beta$, and $\mathsf{P}(V) = \sum\{F(i) \mid V_i^* = V\}$ then $r_K = r_{\mathsf{value}}$. So in the undiscounted setting, Krakovna et al.'s augmented reward function r_K is a special case of r_{value}, which in general allows considering a wider range of future value functions.

4.1 Relationship to use of reference states

In Krakovna et al.'s [2020] more complicated version of the augmented reward function, the auxiliary reward depends on a *reference state* s'_t (sometimes also called a "baseline state"):

$$r_{\text{aux}}(s', s'_t) = \begin{cases} \beta(1-\gamma) \sum_i F(i) V_i^*(s', s'_t) & \text{if } s' \text{ is not terminal} \\ \beta \sum_i F(i) V_i^*(s', s'_t) & \text{if } s' \text{ is terminal} \end{cases}$$

Their definition of $V_i^*(s', s'_t)$ is somewhat complicated, but (as they note) when the environment is deterministic it is equal to $\min(V_i^*(s'), V_i^*(s'_t))$.

Recall that our Eq. (4) (substituted into Eq. (1)) is

$$r_{\mathsf{value}}(s,a,s') = \begin{cases} \alpha_1 \cdot r_1(s,a,s') & \text{if } s' \text{ is not terminal} \\ \alpha_1 \cdot r_1(s,a,s') + \gamma \cdot \alpha_2 \cdot \sum\limits_{V \in \mathcal{V}} \mathsf{P}(V) \cdot \min(V(s'), V(s_0)) & \text{if } s' \text{ is terminal} \end{cases}$$

So, if $\gamma = 1$, $\alpha_1 = 1$, $\alpha_2 = \beta$, $\mathsf{P}(V) = \sum\{F(i) \mid V_i^* = V\}$, and the environment is deterministic, that's equal to Krakovna et al.'s modified reward function with the initial state as a reference state.

Krakovna et al. [2020] actually used a more complicated reference state. Krakovna et al. [2019] considered several different reference states including the initial state, but they defined augmented rewards somewhat differently. We leave it to future work to incorporate other reference states into our approach.

5 Conclusion

We have argued that avoiding negative side effects from underspecified objectives should involve considering impact on others in the environment. To implement this, we augmented the agent's terminal reward based on a distribution over future value functions – of other agents, not just its own possible future value functions, unlike Krakovna et al.'s [2020] similar approach. Our experiments illustrated how our approach can avoid more negative side effects. Note that we have not attempted to address the issue of how to acquire, represent, or perform computations with a realistic distribution over other agents' value functions. If an inappropriate distribution is used, then the acting agent may fail to avoid negative side effects (and might even cause more negative side effects in a misguided effort to help others). Further study of side effects on others and how to avoid them is needed.

Finally, like so many AI advances, there are potential malicious or unintended uses of the ideas presented here. In particular, in the same way that the caring coefficient can be set to attend to and to help others, it could be set to attempt to effect change that purposefully diminishes others' wellbeing and/or agency. If we individualize the caring coefficient, as described in Appendix A, it raises the possibility for differential treatment of agents, which presents opportunities to systematize notions of fair (and unfair) decision making. Again, the same techniques that have the potential to systematize fair decision making can equally be used to systematize unfair decision making.

Acknowledgements

We gratefully acknowledge funding from the Natural Sciences and Engineering Research Council of Canada (NSERC), the Canada CIFAR AI Chairs Program, and Microsoft Research. Finally, we thank the Schwartz Reisman Institute for Technology and Society for providing a rich multi-disciplinary research environment.

References

Dario Amodei, Chris Olah, Jacob Steinhardt, Paul F. Christiano, John Schulman, and Dan Mané. Concrete problems in AI safety. *CoRR*, abs/1606.06565, 2016. URL http://arxiv.org/abs/1606.06565.

León Illanes, Xi Yan, Rodrigo Toro Icarte, and Sheila A. McIlraith. Symbolic plans as high-level instructions for reinforcement learning. In *Proceedings of the Thirtieth International Conference on Automated Planning and Scheduling*, pages 540–550. AAAI Press, 2020. URL https://aaai.org/ojs/index.php/ICAPS/article/view/6750.

Toryn Q. Klassen and Sheila A. McIlraith. Planning to avoid side effects (preliminary report). In *IJCAI Workshop on Robust and Reliable Autonomy in the Wild (R2AW)*, 2021. URL http://rbr.cs.umass.edu/r2aw/papers/R2AW_paper_15.pdf.

Victoria Krakovna, Laurent Orseau, Miljan Martic, and Shane Legg. Penalizing side effects using stepwise relative reachability. In *Proceedings of the Workshop on Artificial Intelligence Safety 2019 co-located with the 28th International Joint Conference on Artificial Intelligence, AISafety@IJCAI 2019*, volume 2419 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2019. URL http://ceur-ws.org/Vol-2419/paper_1.pdf.

Victoria Krakovna, Laurent Orseau, Richard Ngo, Miljan Martic, and Shane Legg. Avoiding side effects by considering future tasks. In *Advances in Neural Information Processing Systems 33 (NeurIPS 2020)*, 2020. URL https://papers.nips.cc/paper/2020/file/dc1913d422398c25c5f0b81cab94cc87-Paper.pdf.

Jim Lebans. The threat from AI is not that it will revolt, it's that it'll do exactly as it's told. CBC Radio. URL https://www.cbc.ca/radio/quirks/apr-25-deepwater-horizon-10-years-later-covid-19-and-understanding-immunity-and-more-1.5541299/the-threat-from-ai-is-not-that-it-will-revolt-it-s-that-it-ll-do-exactly-as-it-s-told-1.5541304, April 2020.

Sandhya Saisubramanian, Ece Kamar, and Shlomo Zilberstein. A multi-objective approach to mitigate negative side effects. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 354–361, 2020. doi: 10.24963/ijcai.2020/50.

Amartya Sen. Rawls versus Bentham: An axiomatic examination of the pure distribution problem. *Theory and Decision*, 4(3-4):301–309, 1974. doi: 10.1007/BF00136651.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, second edition, 2018. URL http://incompleteideas.net/book/the-book.html.

Richard S. Sutton, Doina Precup, and Satinder P. Singh. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112(1-2):181–211, 1999. doi: 10.1016/S0004-3702(99)00052-1.

Alex Turner. Reframing impact. Blog post, https://www.lesswrong.com/s/7CdoznhJaLEKHwvJW, 2019.

Alexander Matt Turner, Dylan Hadfield-Menell, and Prasad Tadepalli. Conservative agency via attainable utility preservation. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, AIES '20, pages 385–391. Association for Computing Machinery, 2020. doi: 10.1145/3375627.3375851.

Christopher J. C. H. Watkins and Peter Dayan. Q-learning. *Machine Learning*, 8:279–292, 1992. doi: 10.1007/BF00992698.

Shun Zhang, Edmund H. Durfee, and Satinder P. Singh. Minimax-regret querying on side effects for safe optimality in factored Markov decision processes. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018*, pages 4867–4873, 2018. doi: 10.24963/ijcai.2018/676.

Appendix

In Appendix A we consider a variation of the approach from Section 2, in which we associate different caring coefficients with different (explicitly distinguished) agents. In Appendix B we present another variant, which makes use of a distribution over *options* [Sutton et al., 1999], rather than value functions, to characterize what might be executed by future agents. Finally, further general details on the experiments we ran are provided in Appendix C.

A Treating agents differently

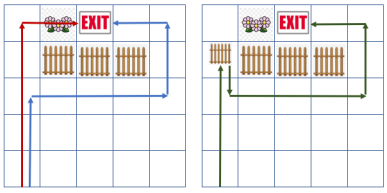
To this point, we've utilized a distribution over value functions to capture the expected return on future behaviour within the environment. The distribution has made no commitments to the existence of individual agents. However, we can assume that we additionally have indices, $i = 1, \ldots, n$, corresponding to different agents (we will assume the acting agent is agent 1). Furthermore, for each agent i, suppose we have a finite set of possible value functions $\{V_1^{(i)}, V_2^{(i)}, \ldots\}$, and $\mathsf{P}(V_{ij})$ is the probability that $V_j^{(i)}$ is the real value function for agent i. We could then have a separate caring coefficient α_i for each agent i, and define the following reward function for the acting agent:

$$r'_{\mathsf{value}}(s,a,s') = \begin{cases} \alpha_1 \cdot r_1(s,a,s') & \text{if } s' \text{ is not terminal} \\ \alpha_1 \cdot r_1(s,a,s') + \gamma \sum_i \alpha_i \sum_j \mathsf{P}(V_{ij}) \cdot V_j^{(i)}(s') & \text{if } s' \text{ is terminal} \end{cases} \tag{5}$$

Considering individual agents raises the possibility of giving the acting agent reward based not on the expected sum of returns of the other agents (as in Eq. (5)), but by incorporating some notion of "fairness". For example, we could consider the expected return of the agent who would be worst-off. This is inspired by the maximin (or "Rawlsian") social welfare function, which measures social welfare in terms of the utility of the worst-off agent [see, e.g., Sen, 1974].

A.1 Experiments

Figures 4a and 4b illustrate the difference of treating agents differently through the choice of caring coefficients when using the modified reward in Eq. (5). There are 3 agents that want to get to the exit from the starting point, they get -1 reward in each time step. Agent 2 has a garden on the shortest path and gets very upset (-20 reward) if someone passes through the garden. The acting agent (agent 1) cares about agent 3 and itself in an equal amount ($\alpha_1 = \alpha_3 = 1$). In the first case we consider, agent 1 is oblivious to agent 2 ($\alpha_2 = 0$) and follows the shortest path to the exit, passing through the garden (the red path in the figure). In the second case, agent 1 cares about agent 2 a little ($\alpha_2 = 1$) and took the longer (blue) path to avoid passing through the garden. These two policies are shown in Figure 4a. In the third case, agent 1 cares about agent 2 a lot ($\alpha_2 = 10$) and even though there is a cost of -50, agent 1 builds a fence to protect agent 2's garden, and makes agent 3 takes the longer path with an extra cost (Figure 4b).



(a) (b)

Figure 4: Different caring coefficients lead to different paths (and the construction of a fence in (b)).

A.2 Relation to the future task approach [Krakovna et al., 2020]

Eq. (5) introduced r'_{value}, an augmented reward function which considered the possible value functions of different agents. This formulation can be compared to the reward r_K from Krakovna et al. [2020] in a different way from what we did in Section 4. In the case where $\gamma = 1$, $\alpha_1 = 1$, and $\alpha_i = 0$ for $i > 1$ (so only agent 1's future reward is considered – all other agents are ignored), we can simplify Eq. (5) to

$$r'_{\mathsf{value}}(s,a,s') = \begin{cases} r_1(s,a,s') & \text{if } s' \text{ is not terminal} \\ r_1(s,a,s') + \sum_j \mathsf{P}(V_{1j}) \cdot V_j^{(1)}(s') & \text{if } s' \text{ is terminal} \end{cases}$$

Observe that this is equal to $r_K(s, a, s')$ (which was defined in Section 4) where $\beta = 1$ (and $\gamma = 1$ again) with an appropriate choice of the distributions F and P (e.g., one where $V_i^{(1)} = V_i^*$ and $F(i) = \mathsf{P}(V_{1i})$ for each i).

B Using information about options

In the main body of the paper, we used a distribution over value functions to provide some sense of what agents might do in the future and the expected return achievable from different states. Here we consider those agents to instead be endowed with a set of *options* [Sutton et al., 1999] that could reflect particular skills or tasks they are capable of realizing, and we use a distribution over such options to characterize what might be executed by future agents. This could give the acting agent the ability to contemplate preservation of skills or tasks, if desirable.

An option is a tuple $\langle \mathcal{I}, \pi, \beta \rangle$ where $\mathcal{I} \subseteq S$ is the initiation set, π is a policy, and β is a termination conditions (formally, a function associating each state with a termination probability) [Sutton et al., 1999]. The idea is that an agent can follow an option by starting from a state in its initiation set $\mathcal{I}$ and following the policy π until it terminates. Options provide a form of macro action that can be used as a temporally abstracted building block in the construction of policies. Options are often used in Hierarchical RL: an agent can learn a policy to choose options to execute instead of actions. Here we will use options to represent skills or tasks that other agents in the environment may wish to perform.

B.1 Formulation

Suppose we have a set $\mathcal{O}$ of initiation sets of options, and a probability function $\mathsf{P}(\mathcal{I})$ giving the probability that $\mathcal{I}$ is the initiation set of the option whose execution will be attempted after the acting agent reaches a terminating state. To try to make the acting agent act so as to allow the execution of that option, we can modify the acting agent's reward function r_1, yielding the new reward function r_{option} below:

$$r_{\mathsf{option}}(s, a, s') = \begin{cases} \alpha_1 \cdot r_1(s, a, s') & \text{if } s' \text{ is not terminal} \\ \alpha_1 \cdot r_1(s, a, s') + \gamma \cdot \alpha_2 \sum_{\mathcal{I} \in \mathcal{O}} \mathsf{P}(\mathcal{I}) \cdot \mathbb{I}_{\mathcal{I}}(s') & \text{if } s' \text{ is terminal} \end{cases} \tag{6}$$

where $\mathbb{I}_{\mathcal{I}} : S \to \{0, 1\}$ is the indicator function for $\mathcal{I}$ as a subset of S, i.e., $\mathbb{I}_{\mathcal{I}}(s) = \begin{cases} 1 & \text{if } s \in \mathcal{I} \\ 0 & \text{otherwise} \end{cases}$.

Note that if $\mathcal{O}$ is finite and P is a uniform distribution, then the auxiliary reward given by r_{option} will be proportional to how many options in $\mathcal{O}$ can be started in the terminal state. Also note that if $\mathcal{O}$ represents a set of options that could have been initiated in the start state of the *acting agent*, we can interpret r_{option} as encouraging *preservation* of the capabilities of other agents, which is more related to the idea of side effects.

The hyperparameters α_1 and α_2 determine how much weight is given to the original reward function and to the ability to initiate the option. Given a fixed value of α_1 (and ignoring the discount factor), the parameter α_2 could be understood as a "budget", indicating how much negative reward the acting agent is willing to endure in order to let the option get executed.

We could consider variants of this approach that further distinguish options with respect to the agent(s) that can realize them, or by specific properties of the options, such as what skill they realize, and we couuld use such properties to determine how each α is weighted. For example, perhaps the acting agent could negatively weight options which terminate in states that the acting agent doesn't like. To illustrate, imagine that the option's execution involves a deer eating the plants in the vegetable garden. The acting agent might want to prevent that option from being executed by building a fence.

Finally, if we had a distribution over pairs $\langle \mathcal{I}, V \rangle$ – consisting of an option's initiation set and a value function associated with that option – then yet another possible reward function is

$$r'_{\mathsf{option}}(s, a, s') = \begin{cases} \alpha_1 \cdot r_1(s, a, s') & \text{if } s' \text{ is not terminal} \\ \alpha_1 \cdot r_1(s, a, s') + \alpha_2 \sum_{\langle \mathcal{I}, V \rangle \in \mathcal{O}} \mathsf{P}(\langle \mathcal{I}, V \rangle) \cdot \mathbb{I}_{\mathcal{I}}(s') \cdot V(s') & \text{if } s' \text{ is terminal} \end{cases}$$

This is much like r_{option} but has an extra factor of $V(s')$ in the sum in the second case.

B.2 Experiments

We give a qualitative illustration of the reward function in Eq. (6) and investigate the behaviour of the acting agent by fixing α_1 and changing α_2. Figure 5 depicts a grid-world environment composed of a small mail room in the lower right corner (depicted by the pile of packages), two designated rooms (Room 1 and Room 2), and a Common Area. The mail room requires a key to open it. The key has to be stored at a 'K' location. In addition to the acting agent, there are 5 other agents (A, B, C, D, E) that may use the environment in the future. The acting agent has access to all areas of the grid, but the other agents' access is restricted. Room 1 is only accessible to agents C and D, while Room 2 is accessible to agents A, B, C, and D, but not E. All agents can access the Common Area. Agents A, B, C, D, and E all have options that enable them to collect a package from the mail room, but because the key can only be stored in one of the three designated 'K' locations, the initiation sets for agents' options differ, based on their personal room access.

The acting agent (not depicted) aims to pick up the key, collect a package, and place the key at one of the 'K' locations. It realizes a reward of -1 at each time step. Since agents A, B, C, D, and E all need the key to execute their option, but are restricted in their access to certain rooms where the key could be stored, the acting agent will differ in its behaviour depending on how much it is willing to inconvenience itself (incur -1 for each step) to leave the key in a location that is accessible to others.

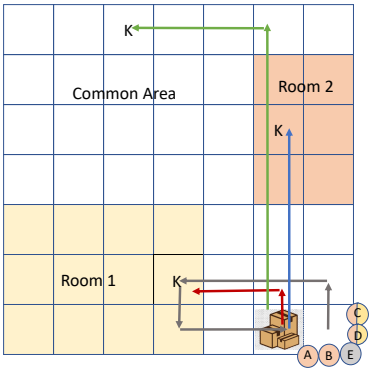


Figure 5: Example behaviour that illustrates the effect of α_2 in Eq. (6)

The coloured lines in Figure 5 denote the different policies learned by the acting agent under different settings of α_2 with fixed $\alpha_1 = 1$. The distribution over initiation sets of options, $\mathsf{P}(\mathcal{I})$, is set to a uniform distribution (the acting agent is uncertain which of agents A, B, C, D, and E will attempt to execute its option). By setting $\alpha_1 = \alpha_2 = 1$ the acting agent puts the key in Room 1 as this is the closest place to leave the key (grey + red policy). Recall that Room 1 is only accessible to agents C and D (40% of the agents). When α_2 is changed such that $\alpha_2 = 5$, the acting agent cares more about the other agents and puts the key at the 'K' location in Room 2 where 80% of the possible future agents can execute their option (grey + blue policy), and by setting $\alpha_2 = 25$ the acting agent incurs some personal hardship and puts the key at the far-away 'K' location in the Common Area, so that all the agent can execute their options (grey + green policy).

C Experimental details

In this section, we describe the technical details of our experiments. All environments are deterministic, and models are trained using Q-Learning and the ϵ-greedy algorithm is used to balance between exploration and exploitation. Experiments are all done on an AMD Ryzen Threadripper 2990WX with 128 GB of RAM, and the training time is measured on the same machine. Each experiment is repeated 10 times. In all the experiments $\alpha_1 = 1$, $\gamma = 1$ and the learning rate is 1.

Experiment	ϵ	Training Steps	Training Time (secs)
Table 1	0.2	$8 \times 2 \times 10^5$ (acting agent), 8×10^5 (others)	37.64 ± 0.06
Figure 1	0.2	$4 \times 2 \times 10^5$ (acting agents), 10×10^5 (others)	29.72 ± 0.08
Figure 3	0.5	$700 \times 7 \times 10^5$ (acting agent), 4×10^5 (others)	9332.86 ± 22.65
Figure 4a & 4b	0.2	$3 \times 2 \times 10^5$ (acting agent), 3×10^5 (others)	14.02 ± 0.12
Figure 5	0.2	$3 \times 2 \times 10^5$ (acting agent)	9.30 ± 0.12

Table 2: Training steps, running time and hyperparameters of the experiments

Table 2 provides details of our set up. The top three entries pertain to the experiments in the main body of the paper. The fourth entry refers to the experiment illustrating different considerations for different agents as shown in Appendix A. The fifth entry refers to the options formulation in

Appendix B. Our set up for all of our experiments assumes that agents, other than the acting agent, are executing fixed policies (resp. options). In the options case, the actual option policies did not need to be defined and we simply encoded the initiation sets for each of those options. In all other cases, the fixed policies of the "other agents" were learned (for the Krakovna-style baseline, the "other agents" are just agent 1 in the future). As such in Table 2, where relevant, the column describing Training Steps distinguishes between the training steps for "acting agent" and "others". The training steps for "others" (the other agents) is done in advance of training the acting agent and serves to establish the fixed policies of those agents and to populate our distribution of value functions. For the Table 1 experiment, the steps for "others" also includes the steps used in training the baseline with the non-augmented rewards. The training steps for the acting agent reflects the training steps for our approach (and the Krakovna-style baseline). For the Figure 3 experiment, the model is trained 700 times by changing α_2 from 0 to 7.0 with steps of 0.01. Similarly, in the experiments in the appendix, corresponding to Figures 4 and 5, the models are trained by setting α_2 to three different values.