# Creation of AI Agents as a Principal-Agent Problem

Chris van Merwijk
Department of Economics and Brasenose College
University of Oxford
Student number: 1049827 - Candidate number: 1019477

May, 2019

Submitted in partial fulfilment of the requirements for the degree of the Master of Philosophy in Economics. This document is my work except where otherwise indicated. Word count: 16688

# Creation of AI Agents as a Principal-Agent Problem

Chris van Merwijk

**Abstract**

The *alignment problem* is the problem of ensuring that artificially intelligent agents pursue goals that are aligned with those of humans. I formalize *inner alignment* of machine learning systems as a principal-agent problem in which a boundedly rational *selector* (the principal, a learning algorithm) delegates decision making to a boundedly rational *actor* (the agent, e.g. a self-driving car) by endowing it with a utility function specified in an agent *algorithm*. The algorithm plays the role of the contract in standard economic agency theory. The selector's imperfect prediction of what environments the actor will encounter, together with the actor's imperfect knowledge of the causal structure of those environments, will generally result in the actor's utility function being misaligned with that of the selector. The central concern is: how do we ensure that the actor's utility function is equivalent to the selector's, so that the actor acts in humans' interest also beyond the foreseen environments? I give simple conditions on the selector's and actor's knowledge of and control over the causal structure of the actor's environment, that incentivizes the selector to "project" it's utility function on a set of *proxy variables*, and leave it up to the actor to optimize them. Moreover, I give restrictive necessary and sufficient conditions under which the actor's utility function is necessarily equivalent to that of the selector, under no restrictions on the space of utility functions. My model illustrates how economists and computer scientists may work on an agency theory of machine learning systems to improve our understanding of the AI alignment problem.

# Contents

# 1 Introduction

The field of artificial intelligence (AI) has made significant progress in recent years in developing autonomous *intelligent agents* that perform tasks that until recently were beyond the capability of machines: artificially intelligent agents (or AI agents) are now able to defeat world-class professional players of complex games such as Go (Silver et al., 2016, 2017) and StarCraft II (Vinyals et al., 2019); They have started to approach human-level competence at complex skills such as object recognition (He, Zhang, Ren & Sun, 2015), and the physical control and movement of humanoid robots (Andrychowicz et al., 2019), autonomous drones (Floreano & Wood, 2015), and autonomous vehicles, or "self-driving cars" (Schwarting, Alonso-Mora & Rus, 2018). As AI agents become more advanced, they will most likely be given a larger role in society, such as economic and military affairs (Bostrom, 2014; Russell, 2015; Schwab, 2016). It therefore becomes increasingly important to understand how AI agents will behave, in order to ensure that their role in society is as beneficial as possible (Rahwan et al., 2019).

This paper introduces and analyzes a particular kind of principal-agent model *between two AI systems*. We argue that this problem is inherent in the current paradigm of artificial intelligence (called *machine learning*), and that this model can provide some understanding of the behavior of AI agents. This model, and the questions we want to ask about it, are quite different from those in the context of human principal-agent relations: Some constraints and considerations that apply to humans simply do not apply to AI agents, and vice versa. Nevertheless, the fundamental source of the agency problem is the same: AI agents have *imperfect information*, and *bounded rationality* (bounded computational capabilities).

There will be two decision makers, both of which are *computing devices*: an "actor" (the agent), and a "selector" (the principal). They could be software programs, or physical computing devices. We will use the example of self-driving cars as an illustration.

The *actor* is an AI agent. It is a piece of intelligent software that is going to take actions in the world. In the case of self-driving cars, it is the software that actually controls the car. It is physically located inside the car (e.g. behind the dashboard), and will receive some *signals*, such as images from the cameras on its front and rear, a measured speed signal from its wheels, a gas meter, a location signal from a GPS system, and so forth; And it will decide on certain *actions*, such as setting the angle of the steering wheel, how hard to activate the gas, and whether to break.

It is in the interest of human users that the actor behaves in a certain way. For example, software that controls a self-driving car (the actor) should make sure that the car does not crash into another car, does not hit pedestrians, follows the traffic rules, and reach the specified destination. However, humans are not able to manually write the algorithms that are able to do such complex behavior. The approach that is currently taken in the *machine learning* paradigm of AI, is that rather than writing a program manually, the human specifies an *objective function*, or *utility function*. This utility function will not say what *actions* the car should take, but what *outcomes* of its actions would be good or bad according to humans (do not hit pedestrians, reach the destination, and so forth). But the human will not itself

optimize this utility function: The human writes another program (distinct from the actor), which is going to do this. This program is called the *learning algorithm*,[1] but we will call it the *selector*, and the utility function that the human gave to it $U_S$.

The *selector* is the principal in our model, and like the actor it is a piece of software. However, unlike the actor, it is not itself going to take actions, but is going to select the *algorithm* that the actor will execute to determine its actions. In the case of self-driving cars, the selector is typically not located in the car itself, but on a *remote server*. It does not itself observe the camera images that the actor observes as the actor is driving around, and cannot send messages in real-time to the actor to influence its behavior. Rather, it is going to specify some (potentially quite complex) set of rules that the actor will execute in the form of an algorithm. It is going to choose this algorithm in order to maximize the utility function $U_S$ that the human gave it. A (particularly bad) rule could be: no matter the visual input data, always drive straight ahead.

We can in principle consider different types of algorithms that the selector could select. However, in this paper we are going to make the theoretical assumption that the selector cannot specify the actions that the actor will take directly, but *delegates* decision making to the actor. It does this by specifying in the actor's algorithm a utility function that the actor should maximize. [2]

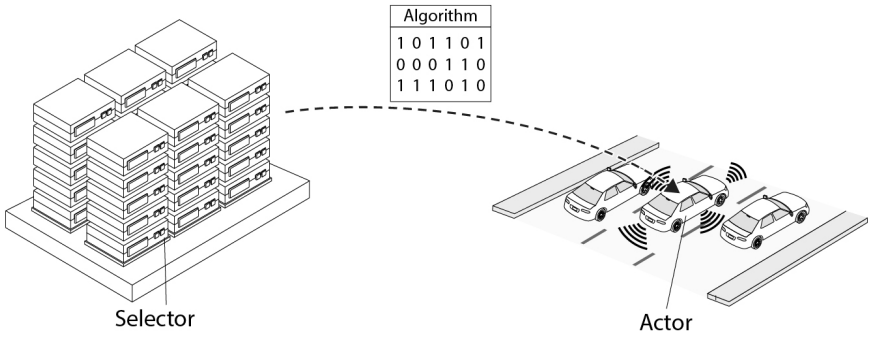


**Figure 1.** In the example of a self-driving car, the actor is a computer device in the car, and the selector is a program running on a *remote server* of the car company. The selector does not influence the decision making of the actor in *real-time*: It only selects an *algorithm*, which the actor will then execute autonomously to make its decisions. We will model this as a principal-agent problem, where the algorithm plays the role of the contract.

So the setup is as follows: The selector and actor are both utility maximizers. The selector's utility function $U_S$ is exogenously specified by the humans (and captures the desired outcomes: following the traffic rules, avoid killing pedestrians, reach the specified destination, and so forth). But the actor's utility function $U_A$ is *not*

[1] More accurately, in the terminology of Russell & Norvig (2010) it is the combination between a learning algorithm or *learning element*, together with the reward element.

[2] The reasoning behind assuming that the actor is a utility maximizer is discussed in section 2.1.

*exogenous*: It will be chosen by the selector. Given the $U_A$ that the selector chose, the actor will then actually take actions to maximize its expected $U_A$, based on observations of signals from the environment. The selector is going to predict how the actor would behave depending on the utility function $U_A$ that the selector would give it, and will choose $U_A$ such that the actor would take actions that avoids killing pedestrians, and so forth.

It is natural to ask: Why would the selector not just give the actor the utility function $U_S$, so that the actor always makes the right decisions? The key point of this paper, is that due to the *bounded rationality* of the actor and selector, this might be a sub-optimal choice, or not uniquely optimal. The actor has limited computational abilities and a limited ability to understand its environment. In the self-driving car example, the actor might have a limited ability to model how deciding to drive very fast might result in hitting a pedestrian. It is unable to understand this causal relation. The selector on the other hand, has a limited ability to reason about data it has not seen. In the self-driving car example, the remote server cannot in real-time observe the actor's signals, and if it has not yet seen that data in the past, it is computationally infeasible to reason about it in advance.

We will model this bounded rationality as a limited knowledge about what outcomes will result from the actor's actions. This is similar to Spiegler's economic analysis of bounded rationality modeled as distorted subjective causal model (2016; 2019), and is in line with the use of causal models in computer science to describe the environments of agents (Everitt, Ortega, Barnes & Legg, 2019). We can model the effects of the actor's actions as a *causal graph* (Pearl, 2000; Dawid, 2002), which captures the idea that the actor's actions have direct effects on certain variables (speed of the car), but also indirect effects (whether the car hits a pedestrian). The bounded rationality is modeled by assuming for example, that the actor does not know that by speeding the car up, it will hit a pedestrian. As an illustration see figure 2. The selector gives the actor a utility function that says "Don't hit yellow shapes". This causes the actor to avoid hitting the pedestrian, because they were wearing yellow clothes, but the actor is doing it for a different reason than the humans intended.

But why is this a problem? The actor ends up not killing the yellow-clothed pedestrian. But imagine that now, the actor encounters a pedestrian that is wearing *green* clothes. The actor will drive the car straight into the pedestrian, since it only cares about avoiding yellow shapes, not pedestrians in general. This was the result of the actor's bounded understanding of the situation, and of the selector's failure to predict the occurrence of a green-clothed pedestrian, and to give the actor a better algorithm.

The result is that the actor receives a utility function that happens to work conditional on a specific expectation about the actor's environment, rather than a utility function that encodes the actual goals that humans have. The selector gave the actor a sub-optimal algorithm as a result of its imperfect prediction, and to compensate for the actor's bounded rationality.

These problems are well-known in the context of machine learning, and the problem of the selector's failure to choose an algorithm that works for a broader set of situations than those that it has data about, is called the problem of "robustness"

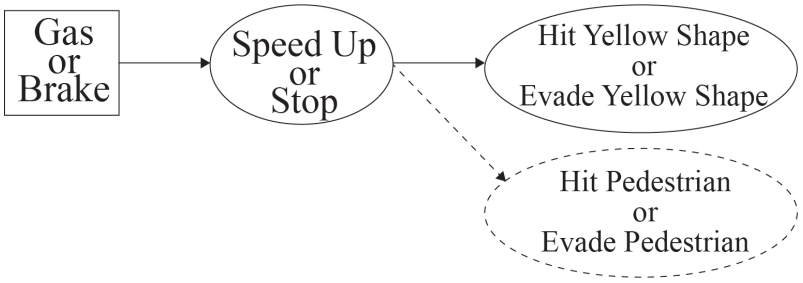


**Figure 2.** Bounded rationality of the actor is a source of *misalignment* between selector ($U_S$) and actor ($U_A$): The actor has a set of actions (a square): It can either step on the gas, or brake. This will cause the car to either speed up or stop. But now assume that there is a pedestrian wearing yellow clothes in front of the car. If the car speeds up, it will hit and kill the pedestrian. But assume now that the actor does not understand this, because the actor only perceives a *yellow shape*, and does not know that what it is seeing is a pedestrian. This effect on the pedestrian is a dotted line: there is an effect, but the actor does not know it. In this situation, the selector is better off giving the actor the goal "do not hit yellow shapes", rather than "do not hit pedestrians". This will cause the actor to slow down, to avoid hitting a yellow shape, which in this case coincides with what the selector wants, which is to avoid hitting pedestrians.

(Amodei et al., 2016). In this paper, I will model this problem as an agency problem.

The central concern of this paper is: How can humans ensure that the actors is endowed with the utility function that the humans specified to the selector? In our earlier work, we have called this *the inner alignment problem* (Hubinger, van Merwijk, Skalse, Mikulik & Garrabrant, 2019). To this end, I will set up a model of a two-stage game between a boundedly rational selector and a boundedly rational actor, where the selector endows the actor with a utility function encoded in an algorithm. The paper is concerned with two primary questions: Firstly, what factors determine which utility function $U_A$ that the selector gives to the actor, and will it be misaligned with its own utility $U_S$? Secondly, when is it uniquely optimal for the selector to give the actor a $U_A$ that is equivalent to its own $U_S$, and what prevents this uniqueness?

The paper proceeds as follows: section 2 will present the main idea informally. In particular, we discuss the standard model of AI agents that choose actions to maximize their reward function, and argues in more detail for instead considering the AI system as consisting of a principal (selector) and agent (actor). Section 3 states the model. Section 4 outlines a number of simple toy examples to illustrate the idea. Section 5.1 addresses question 1 given restrictive assumptions by formalizing the mechanism described above, and section 5.3 addresses question 2. Section 6 concludes.

# 2 The Main Idea

In this paper I will propose to model machine learning systems not as monolithic (black box) entities that take actions to maximize a utility-, or reward function,

but as consisting of two entities in a *principal-agent* relation: The principal will be is what Russell & Norvig call the learning element (we call it the *selector*), and the agent will be what they call the performance element (we call it the *actor*) (2010). This is a nonstandard view: machine learning systems are generally modeled as agents that choose actions to maximize their reward function. But there is incidentally quite a close analogy with agency theory as applied to owner-manager, or manager-worker relations (Stiglitz, 1989): According to the neoclassical view of firms, firms are modeled as monolithic (black box) profit maximizers. That is, we don't look at the internal relations between owners, managers, and workers, but consider them a monolithic coherent entity working in the interest of the owners. Then, economists started studying the internal relations of the owners, managers and workers in order to understand how such imperfections as imperfect information, incontractibility, bounded rationality, and so forth, resulted in the *agency problems*, and the firm's behavior deviating from profit maximization (Ross, 1973; Williamson, 1975; Jensen & Meckling, 1976; Fama, 1980). In this paper I analogously propose to model the interaction between the two elements of a machine learning system as an agency relationship.

## 2.1 Existing work: Machine Learning Agents as Reward Maximizers

First, we give a background on machine learning agents. The essential difference between machine learning, and classical approaches to AI (and software development more broadly), is that humans do not directly write an algorithm that they want (Russell & Norvig, 2010). Rather, they write a "middle man" algorithm, a so-called *learning algorithm*, which will then automatically develop the desired final algorithm, which also in this context is called the *learned algorithm*. This approach has allowed machine learning researchers to develop (learned) AI algorithms that humans simply do not know how to write themselves, either because they don't understand the task, or because the desired algorithm is too complex for a human to understand. For example, while we have a rough understanding of how the brain processes visual information, we don't exactly understand the details of how each signal is processed or why (Goodfellow, Bengio & Courville, 2016). A human can *intuitively* drive a car, but they would not know how to write a program that captures every minute detail of how they moves their muscles, and her eyes, and so forth.

By programming a learning algorithm, we have been able to automate the development of intelligent algorithms. This has resulted in intelligent algorithms that solve complex tasks despite the human programmers not knowing how they do it. These algorithms even outperform humans at difficult tasks that are outside the scope of classical (human-programmed) software, such as correctly recognizing objects from an image, and defeating human players at complex board games like Go (Silver et al., 2016, 2017), and even real-time strategy games with incomplete information such as StarCraft II (Vinyals et al., 2017, 2019) and Dota 2 (OpenAI, 2018).

Another example is that car companies have started to train AI agents to control

the movement of cars (Hodges, An, Rahmani & Bennamoun, 2019).[3]

A central model for such artificially intelligent systems is that of an *intelligent learning agent*, or *machine learning agent* (Russell & Norvig, 2010).[4] The machine learning agent consists broadly speaking of two *elements* that compute their own separate algorithms: The learning element, which computes a learning algorithm, and the performance element, which computes a learned algorithm. The machine learning agent is thought of as having a *goal*, specified in the form of a reward function and *trying* to get better at achieving this goal, by learning from its experience. The learning element is thus generally considered to be *part of* the agent.

The performance element's task is to compute a *policy function*, or *policy*. This is a function that takes as input information from the environment, and outputs actions. In the case of a self-driving car, the performance element takes as input camera images, and outputs a steering direction and so forth. The learning element's task is to change the performance element, based on past information. We can summarize this by saying that the learning agent chooses a policy (i.e. actions) to maximize its expected reward:

> **Machine learning agents as choosing a *policy* to maximize reward** : It is standard to model machine learning agents as choosing *actions*, or *policies*, to maximize their expected reward, conditional on their observed informaion. Learning consists of improving the approximation to the optimal policy by obtaining and using more information (training data):[5]
>
> $$\max_{\text{policy}} \mathbb{E}\left[\,\text{Reward(policy)} \mid \text{data}\,\right]$$

This model is justified by the fact that machine learning algorithms *literally* select (using an optimization algorithm) a policy algorithm in order to maximize a reward function.[6] However, there is a discrepancy between thinking of machine learning agents as choosing a *policy*, and the fact that they instead choose an *algorithm that will compute that policy*.

There is a special case, a class of algorithms where choosing an algorithm is equivalent to choosing a policy, namely in the case of *lookup-tables*: The learned algorithm that the performance element computes is in this case simply a table, that

[3]The process of developing these algorithms is called "training", and works by repeatedly giving the learned algorithm sample inputs and seeing what outputs (decisions) the learned algorithm produces (Goodfellow et al., 2016). The learning algorithm then adjusts the learned algorithm in order to improve the score of its outputs on an objective function that the humans specified.

[4]This term more specifically applies in the context of "reinforcement learning". See for example (Sutton & Barto, 1998) for details.

[5]Reward denotes here discounted future reward. Moreover, often this formula is instead stated in terms a value function $V_\pi = R(\pi(s_t)) + \delta\mathbb{E}[V_\pi(s_{t+1})]$. *Reinforcement* learning agents are then seen as trying to optimize this value function by choosing a policy. Deviation from the optimum is considered an "approximation error" (Kaelbling, Littman & Moore, 1996; Sutton & Barto, 1998; Sxepesvari, 2010).

[6]In fact, it has been argued that while the model of utility maximizers was developed to describe human behavior, it is in fact much better suited to describe the behavior of machine learning systems (Parkes & Wellman, 2015).

for each possible observation stores an action. The learning element then optimizes the action for each contingent observation individually.[7] The appropriate action is "learned" independently for each observation.

However, look-up tables are only applicable to very simple problems. As soon as the set of observations becomes larger, it becomes completely computationally intractable to learn, and store a separate action for each possible contingency.[8] Apart from the very special and limited case of lookup-tables, learning algorithms always select not actions, but *parameters for an algorithm* (the learned algorithm) that will compute those actions.[9]

> **Machine learning systems choose *algorithms* to maximize reward** : Closer to the reality of machine learning is the model that a learning element chooses an *algorithm* that will compute the policy, rather than the policy itself. It is standard to think of this as meaning that the machine learning agent merely chooses *approximation* of the optimal policy, since there may not be an algorithm for the true optimal policy.
>
> $$\max_{\text{algorithm}} \mathbb{E}\,[\,\text{Reward}(\text{policy}(\text{algorithm}))\mid \text{data}\,]$$

This means that to model machine learning systems as choosing *policies* or *actions* to maximize their reward is merely an *abstraction*. This may in many situations be a fairly good and descriptive abstraction, just as it is often a good abstraction to think of firms as black box profit maximizers (Hart, 1989). The model may, in particular, be a good abstraction to the extent that the algorithm is "similar" to a look-up table, or to the extent that the learning algorithm can use a large amount of data.[10] In this case, we might model the learning algorithm as selecting an "approximation" to the optimal policy, and abstract away from the fact that what it selects is not a *policy* but an *algorithm that computes* the policy.[11]

However, it is arguable that the abstraction becomes less and less accurate as the learned algorithm consists of complex processes that do not at all resemble look-up tables, and to the extent that the learned algorithm *generalizes* from a small amount

[7] Actually a far more common system is a so-called Q-table, which is only slightly more complicated, and the basic argument made here applies to these as well. See for example (Sutton & Barto, 1998) for details.

[8] For example, a self-driving car could receive a visual input image of, conservatively, 256 by 256 pixels. This means it can in principle receive $10^{65536}$ possible inputs, and would have to store an action for each separately, which would require roughly $10^{65536}$ Gigabytes of data. To store this data requires a hard disk vastly larger than the size of the universe, and to learn each optimal action a time vastly longer than the age of the universe.

[9] Most advances in recent years have been through the application of "deep learning", where the parameters describe computations in a "deep neural network" (LeCun, Bengio & Hinton, 2015). See for example (Goodfellow et al., 2016) for an introduction.

[10] See (Hubinger et al., 2019) for a discussion on this.

[11] It is standard to view the fact that learning algorithms are constrained by having to select algorithms (or parameters) rather than the policies themselves as meaning that there is an "error" in the chosen policy, but that they should still in the abstract be modeled as choosing policies. This assumption seems to apply to most models of reinforcement learning agents for example (Sutton & Barto, 1998; Kaelbling et al., 1996; Sxepesvari, 2010). I do not argue that this is wrong, but that it is an abstraction that in this paper we move away from.

of data. For example, recent advances in reinforcement learning have developed learned algorithms that first use complex subroutines to develop "plans", and use those plans to then take actions. In such systems, there are in fact two optimizers: The learning element selects an *algorithm* for the performance element to optimize the reward function, and then the performance element uses this algorithm to select a *plan* to optimize the utility (Srinivas et al., 2018; Li et al., 2017). In other work, we have called such a phenomenon *learned optimization* (or more specifically *mesa optimization*) (Hubinger et al., 2019). If one wishes to model such a system as a monolithic utility maximizer, it is not at first obvious what its utility function would be. Would it be the learning element's (the selector's) utility function, or the performance element's (the actor's) utility function?

The idea of modelling machine learning systems as utility maximizers has been criticized before. For example, Drexler (2019), and Hubinger et al. (2019) point out that we can simply "turn off" the learning algorithm and run the learned algorithm independently.[12] The learning element is not necessary for the machine agent to function competently once the algorithm of the performance element has been determined, and keeping it activated is entirely optional (Drexler, 2019). We can (and may have to) place the machine agents in new environments about which the learning algorithm never had any data. In these new environments, the actions that the performance element (actor) takes may no longer be optimal with respect to the reward function that was used to create the algorithm. But as we have point out in other work, it may still be optimal with respect to the objective function of the performance element (actor) (Hubinger et al., 2019).

The model of machine agents as choosing policies to maximize their reward, and learning to improve their policies, may be a good and useful abstraction depending on the context. However, I suggest a different model, namely model machine learning systems as two separate entities: The selector (the learning element, or learning algorithm), and the actor (the performance element). We think of the selector as selecting an algorithm for the actor, which the actor then uses to compute the policy.

## 2.2 Selector and Actor: A Principal-Agent Problem

Modeling machine learning systems as monolithic agents that choose their actions to maximize their reward function $U_S$ is an abstraction. There is nothing in principle wrong with an abstraction like this. We need abstractions to think about and predict what machine learning systems will do without getting bogged down in details of specific machine learning algorithms. But as the analysis in the previous section shows, this abstraction is somewhat coarse grained. In this paper, I propose a more fine grained abstraction. I will state the model formally in section 3.

We will think of the learning algorithm together with the reward function $U_S$ that the humans gave it, as the *principal*, and the artificially intelligent agent as the *agent*.[13] Moreover, to clarify the roles of these two, we will call them *selector*

[12] In fact, this is standard practice in many applications. An example is self-driving cars, where the learning algorithm is usually located in a remote server.

[13] There is a potentially confusing terminological coincidence with so-called *actor-critic* methods in machine learning (Peters, Vijayakumar & Schaal, 2005), it might be tempting to think of the

and *actor* respectively. The actor will have control over some set of actions, such as the virtual actions of a software agent, the physical actions of a robot, or in the self-driving car example, the actions of steering, breaking, etc. The selector does not directly choose the actions that the actor will take, but will select an algorithm (the learned algorithm, or agent program) that the actor will execute.

In analogy with human principal-agent problems, the *algorithm* plays the role of the *contract*: A human principal selects a contract from some restricted contract space, that will govern the behavior of its agent. Information asymmetry and bounded rationality will result in this contract being sub-optimal in some way. Similarly, the selector will select an algorithm from a restricted space, which will govern the behavior of its actor. And similarly, information asymmetry and bounded rationality may cause this algorithm to be sub-optimal.[14]

In particular, the space of algorithms is restricted in such a way that the selector cannot tell the actor what actions to take in any given contingent situation (as explained in section 2.1). Instead, we will assume the opposite: The selector must *delegate* decision making to the actor, and the algorithm will only specify the *goal* that the actor must optimize, rather than the specific actions. Specifically, the selector will select a utility function $U_A$ for the actor, and the actor is unrestricted in optimizing this utility function (that is, the selector does not give it any constraints in doing so).[15] Thus, the algorithms that the selector can choose for the actor will be denoted by utility functions $U$, and we will simply say that the selector selects a utility function $U_A$.

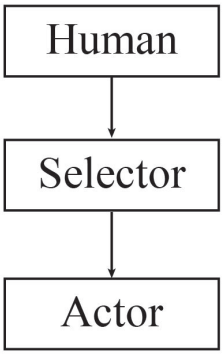


**Figure 3.** The selector acts on behalf of the human, while the actor acts on behalf of the selector. The selector is a kind of "middleman" because it is the actor's behavior that the human in the end cares about.

Note that there is here an important difference with human principal-agent relations: The actor does not have a pre-existing utility function, before the selector creates it. In fact, before the selector gives the actor an algorithm to run, it is unable to do or "think" anything. In this sense, by selecting an algorithm, the selector *creates* an AI agent.

Based on the discussion in section 2.1, we can informally identify the following informational constraints of the two decision makers:

**Actor.** The actor will observe a signal from its environment, such as

selector as the *critic*, and the actor as the *actor*. However, in the case of an actor-critic method, *both the actor and critic* are part of the actor in the sense we use here. They are both running at "inference time", and can potentially be seen as what we have in earlier work called a *mesa-optimizer* (Hubinger et al., 2019). However, this is the type of technical detail that we wish to abstract away from in this paper, by considering the actor to be a utility maximizing agent.

[14]We can thus envision an *algorithmic agency theory*, or *algorithmic contract theory* as compared to the classical human-based agency- and contract theory.

[15]This type of *delegation* can be seen as describing the delegation of optimization in the form of a so-called *learned optimizer* or specifically, a *mesa optimizer*, a concept we have introduced in previous work (Hubinger et al., 2019). There we discuss the technical reasons why such an algorithm would be incentivized to emerge in more detail.

visual information from a camera. However, the actor does not have direct knowledge of the environment, and has a limited ability to decode its signal information and turn it into an accurate model of its environment. As a result, even if the signal it observes contains all the relevant information, the actor will have a limited understanding of the consequences of the actions it could take.

**Selector.** The selector does not observe the signal that the actor observes, but the selector has information about some subset of possible signals that it has from past observations, or that it can generate using simulations (the *training data*). As a result, it has direct knowledge of (part of) the ground-truth that generated the signal, so that it has better information about the consequences of the actor's actions for those signals. However, the selector has a bounded ability to reason about signals about which it does not have data. As a result, it will not be able to select the optimal algorithm for those signals.

The method by which the selector learns an algorithm for the actor may be technically complex. In current machine learning systems it always involves "testing out" what the actor would do in given situations. However, we wish to abstract from these kinds of details, and simply conceptualize the selector as "trying to predict" what the actor will do once it deploys its algorithm. We thus consider the entire process of developing the actor's algorithm (training) as selecting an algorithm based on predictions what the actor will do once deployed. The selector and actor will play a two stage game:

**Stage 1** : The selector forms an expectation of the environment the actor will be in in stage 2, based on its prior information $I_S$, which may contain observations from past iterations of the same game, simulated environments, and so forth (training data). It then selects an algorithm for the actor to execute, such that it expects the actor to take actions that score high on its utility function $U_S$.[16]

**Stage 2** : The actor is placed into an environment, receives signals from the environment, and takes actions. It might observe signals that the selector did not foresee, in which case the algorithm may not perform optimally given those signals (according to $U_S$). In particular, if the selector "delegates" decision making to the actor as we will assume in this paper, then the algorithm says "take actions to maximize a utility function $U_A$", and the chosen $U_A$ may not agree with $U_S$ on the optimal action, even if it agreed with $U_S$ in the selector's expectation. As a result, the actor may (intentionally) take an action that is against the interest of the selector and the human.

[16]In this conceptualization, the entire training process can be seen as a making a "prediction" about the actor's environment outside the training environment. In this framing, techniques of *meta-learning* improve the selector's ability to generalize from training data (Andrychowicz et al., 2018). In this framing they can be seen as an attempt to improve the bounded rationality of the selector.

The central mechanism that emerges is as follows. The selector will choose a utility function $U_A$ for the actor, taking into account the actor's bounded rationality, i.e. the actor's lack of knowledge about the consequences of its actions. It will do this by "integrating" the information that the actor lacks into the actor's utility function, given the selector's distribution over signals.[17] As a result, $U_A$ will deviate from $U_S$.

Moreover, there may be many different $U_A$ that achieve an optimal outcome, *even if* the actor has perfect information, as the result of the selector expecting only a limited subset of the signals that the actor could possibly observe.

# 3 The Model

Our model will be based on *causal diagrams* like the one shown in the introduction. These causal diagrams will be summarized as a function $f$, and will be introduced in section 3.4. However, we first state the model without going into the internal structure of $f$.

## 3.1 The basic model

There are two utility maximizing agents: the *actor* and the *selector*.

There will be a state of the world $\omega$ from a finite set $\Omega$. Intuitively, the state $\omega$ will contain at least all information about the environment of the actor before the actor makes a decision. We will later see that it also describes what signal the actor receives. In the example of a self-driving car, $\omega$ might describe where the car is located, whether there are other cars around it, how fast they are moving, whether it is raining, and so forth.

We will make the simple assumption that there is a single time period: The actor is going to take a single action $a$ from a finite set $A$ of actions. Intuitively, for a self-driving car this might be a decision on what direction to steer in, whether to break, or gas, what gear to use, and so forth. An action is going to result in some *outcome* $o$ from a finite set $O$. What outcome a given action results in will depend on the state of the world $\omega$. This is formalized by a function $f^\omega \colon A \to O$. $f$ describes for every state, what outcome will result from a certain action. For example, $o$ might describe whether the car reaches its destination, or crashes, whether it has violated a traffic rule, and so forth. If the state $\omega$ is such that the car is in front of a red traffic light, then the action "accelerate" would cause an outcome "traffic violation", but not if the traffic light in $\omega$ is green. Both the selector's and actor's utility functions will be defined on outcomes: $U_S, U_A \colon O \to \mathbb{R}$.

For now we assume $f$ is arbitrary, but in section 3.4 we will analyze $f$ as a "causal diagram": we will graphically draw the causal effects of different variables roughly as in figure 2 in the introduction.

However, the state $\omega$ is uncertain, and so $f$ may also be uncertain: There is a distribution $\mathbb{P}$ over $\Omega$.[18] The actor and selector have prior information $I_A \subseteq \Omega$ and

[17] In the example in the introduction, the selector gives the actor the goal of preventing the car to hit yellow shapes, rather than the actual goal of preventing the car to hit pedestrians.

[18] We will assume throughout, that actor and selector have common prior $\mathbb{P}$ on $\Omega$. This does not

$I_S \subseteq \Omega$ respectively, which will denote the possible states that the actor and selector respectively consider to be possible. Intuitively, $I_S$ contains the training data, and $I_A$ is the prior information that the actor has.[19]

In order to provide the actor some information about $f$, the actor receives a signal $\theta$ from a finite set $\Theta$. The signal observed in state $\omega$ is denoted $\theta^\omega$. In this way, observing $\theta$ tells us something about $f$. For example, this signal $\theta$ might be the input from the cameras of the self-driving car, and might contain the information that there is a pedestrian in front of the car, so that $f$ is such that if the actor decides to accelerate, this would result in an outcome where a pedestrian is killed. The selector will never observe $\theta$.

Given that the actor will select an action for each possible signal $\theta$, we can capture the actions that the actor chooses as a function $\pi\colon \Theta \to A$. We call $\pi$ the actor's *policy function* or simply *policy*.[20]

The actor will choose its policy function $\pi$ to maximize its utility $U_A\colon O \to \mathbb{R}$ over outcomes. However, $U_A$ is not exogenously specified, but selected by a "selector". The selector itself is endowed with a exogenously specified[21] utility function $U_S\colon O \to \mathbb{R}$ over outcomes. The selector has an interest in ensuring that the actor chooses an action that will result in high expected $U_S$. To this end, the selector has one decision to make, which is to select the utility function $U_A\colon O \to \mathbb{R}$ the actor will maximize.[22]

We can thus summarize the model as a class of two stage games:

> **Stage 1.** The selector selects a utility function $U_A\colon O \to \mathbb{R}$ over outcomes that the actor will maximize in stage 2. In equilibrium, the selector will choose $U_A$ to be an element $U_A^*$ from the set $\mathcal{U}^*$ whose induced policy $\pi_{U_A^*}$ in stage 2 maximizes its own expected utility $U_S\colon O \to \mathbb{R}$:
>
> $$\mathcal{U}^* = \arg\max_{U\colon O\to\mathbb{R}} \mathbb{E}[U_S(f(\pi_U(\theta)))|I_S].$$
>
> **Stage 2.** Depending on the chosen $U_A$ in stage 1, the actor chooses a policy $\pi_{U_A}\colon \Theta \to A$ that gives an action for each possible signal. This policy results in an uncertain outcome $f(\pi_{U_A}(\theta))$, since there is uncertainty over $f\colon A \to O$ and $\theta$. In equilibrium, the actor will choose a $\pi_{U_A}$ from the set of policies $\Pi^*_{U_A}$ that maximize its expected utility $U_A$:
>
> $$\Pi^*_{U_A} = \arg\max_{\pi\colon \Theta\to A} \mathbb{E}[U_A(f(\pi(\theta)))|I_A].$$

make any assumptions about the selector's and actor's knowledge: For any training distribution and every actor's information set we can simply set up $(\Omega, \mathbb{P})$ such that $\mathbb{P}[\cdot|I_S]$ and $\mathbb{P}[\cdot|I_A]$ equal the desired distributions.

[19] It would be more realistic to consider the prior information $I_A$ to be part of the specification of the algorithm that the selector gives the actor. However, for simplicity we will consider it to be exogenous in this paper.

[20] This is intended to coincide with standard terminology in machine learning.

[21] In fact, in the background we assume that $U_S$ is specified by humans, though we will not model this explicitly.

[22] In principle we could analyze the case where the selector can only choose from a subset of such utility functions. In fact, there is a reasonable case to do this, since some utility functions may be far computationally harder to optimize than others Hubinger et al. (2019).

Note that as it is currently specified, the actor has a potentially large amount of possible decisions that are optimal: Any policy function in $\Pi^*_{U_A}$ is an optimal choice. In particular, an actor with a constant $U$ will be indifferent between outcomes, and therefore any policy would be an equilibrium policy for it, including the policies that are optimal for $U_S$. However, such an equilibrium is in some sense "pathological": We as outsiders have no reason to expect this will occur, since the actor would have no reason to choose a policy that coincidentally is optimal for the selector. We want to therefore apply a "principle of indifference", and assume that the actor simply picks a policy at random from its set of optimal policies (independently distributed from $f$). We thus make the following assumption throughout the paper, which we will use without always stating so explicitly:

**Assumption.** *The actor will always randomize uniformly over its optimal policies* $\Pi^*_{U_A}$.

Using this assumption, we can characterize the set of equilibria of this class of games as tuples of a utility function for the actor, together with the set of induced optimal policies for that utility function, $(U^*_A, \Pi^*_{U^*_A})$ for $U^*_A \in \mathcal{U}^*$.[23] The main part of this paper will study such equilibria. In particular, the central question we are interested in, is whether the actor will be "aligned" with its selector in such an equilibrium:

**Definition 1** (Alignment)**.** *The actor is aligned with the selector, if* $U^*_A$ *is "equivalent" to* $U_S$.

Note that "equivalent" here is not defined, and so this definition is informal. This is intentional, since there are multiple possible notions of alignment. We give a formal definition of "weak alignment" in section 5.3.

## 3.2 Informational assumptions

We have already assumed a form of asymmetric information: The actor observes $\theta$ while the selector does not. We now state further assumptions that we will make on the prior information sets $I_A, I_S$ to incorporate the bounded rationality of the two players. We will further specify these assumptions in the specific examples and results.

As we stated in section 2.2, we assume that the selector is boundedly rational in its ability to reason about signals in $\Theta$ about which it doesn't have data. In fact, we will make the assumption that there is a set $\Theta_S \subset \Theta$ about which the selector has some information, and that the selector is completely unable to reason about other signals. For example, if the dataset of the selector of a self-driving car contains only observations about cars driving on concrete roads, then the selector will have no causal knowledge of cars driving on dirt roads. In fact, we will assume that the selector is even unable to reason about the actor's behaviour for signals

[23] In particular, this is a sequential equilibrium: For any $U\colon O \to \mathbb{R}$, if the selector would set $U_A = U$, the actor's strategy is to optimize $U$, even for $U$ that the selector does not end up choosing.

outside $\Theta_S$.[24] Intuitively, this means that the selector of does not even know how to delegate the decision to the actor. We capture this type of bounded rationality as follows:

**Assumption.** *For any $\theta \notin \Theta_S$, the joint distribution of $(f, \pi_U(\theta))$ conditional on $I_S, \theta$ is uniform for all $U$.*

We can immediately state an equivalent and simpler statement of this assumption, which we will use throughout the paper, instead of using the assumption directly:

**Lemma 1.** *The selector behaves as if it assigns probability $\mathbb{P}[\theta \notin \Theta_S | I_S] = 0$.*

*Proof.* Under the assumption, we have that $\mathbb{E}[U_S(f(\pi_U(\theta)))|I_S, \theta \notin \Theta_S]$ does not depend on $U$. Therefore, if we let $p = \mathbb{P}[\theta \in \Theta_S | I_S]$, then $\mathbb{E}[U_S(f(\pi_U(\theta)))|I_S, \theta \in \Theta_S]$ as a function of $U$ is a linear transformation of $\mathbb{E}[U_S(f(\pi_U(\theta)))|I_S, \theta \in \Theta_S] \cdot p + \mathbb{E}[U_S(f(\pi_U(\theta)))|I_S, \theta \notin \Theta_S] \cdot (1-p)$, which is the selector's expected utility. □

We will therefore throughout the paper consider only signals in $\Theta_S$. Intuitively, the selector is so ignorant about the signals which it has no data about, that it pretends as if those signals never occur.[25]

We now formalize the bounded rationality of the actor. While the selector will never observe $\theta$, we assume that the selector has more causal knowledge conditional on $\theta \in \Theta_S$ than the actor, due to the actor's computational constraints. In the case of the car company, the selector that runs on the remote server has on the order of weeks or months to "think through" via trial and error its data, while the actor in the car on the road has to make a split second decision and only has a single chance. We formalize this by assuming that if the selector were to observe the signal $\theta \in \Theta_S$, then it would have a better ability than the actor to infer from $\theta$ what $f$ is:

**Assumption.** *Conditional on any $\theta_S$ in $\Theta_S$, the selector has (weakly) more information about $f$ than the actor:*[26] *$\mathbb{P}[f|I_S, \theta_S, I_A] = \mathbb{P}[f|I_S, \theta_S]$ for all $f$.*[27]

We will specify the exact assumptions on their respective causal knowledge through causal graphs, which we introduce in section 3.4.

[24] This is a defensible assumption, since baseline learning algorithms work by iteratively going through a training data set, and only taking into account the data in that set, without "reasoning" about what decisions the actor would make outside the dataset. One can view some forms of meta-learning as an attempt to improve the selector's bounded rationality in this respect (Andrychowicz et al., 2018).

[25] Alternatively, we could instead interpret lemma 1 as the basic assumption, by thinking of the selector-actor game as taking place entirely during training. In this case the selector is not boundedly rational with regards to the game. 1 shows that the two interpretations of the game are essentially equivalent.

[26] Intuitively, this is justified by the relation between the selector and actor in actual machine learning systems, though we won't model this: the selector, by selecting an algorithm, gives the actor its prior information. The selector cannot give the actor *more* information than it has itself. Moreover, it will in fact give it *less* information than it has, because of the complexity constraints on the algorithm (as we discussed in section 2.1). However, in this paper we will for simplicity take $I_A$ to be exogenous.

[27] Note that there are a finite such $f : \Theta \to A$, since $\Theta$ and $A$ are finite.

## 3.3 Some preliminary definitions and results

We here present some preliminary definitions and results that will simplify the analysis. Firstly, we can distinguish between two notions of optimality. On the one hand we have the policy that the selector would choose if it could choose the policy itself (the first-best). On the other hand we have the best policy given that the choice of policy has to be delegated to the actor (second-best). We formalize these two notions as follows:

**Definition 2** (Second-Best and First-Best)**.** *A second-best policy set is the $\Pi^*_{U^*}$ as defined in stage 2 for some $U^* \in \mathcal{U}^*$. The first-best policy set is*

$$\Pi^*_S = \arg\max_{\pi\colon \Theta \to A} \mathbb{E}[U_S(f(\pi(\theta)))|I_S].$$

*We say $U^*$ is first-best if $\Pi^*_{U^*}$ consists solely of first-best policies.*

We state here a trivial result without proof:

**Lemma 2.** *If a utility function $U$ is first-best, then it is also second-best.*

**Lemma 3.** *If the actor has the same information as the selector, so that $I_A = I_S$, then $U_S$ is first-best.*

*Proof.* This follows trivially by substituting $I_A$ with $I_S$ in stage 2 of the game. □

**Lemma 4.** *If a first-best utility function exists, then any second-best utility function is also first-best. That is, for any $U$, we have $U \in \mathcal{U}^*$ if and only if $\Pi^*_U \subseteq \Pi^*_S$.*

*Proof.* The selector can guarantee a first-best policy by selecting the first-best utility function. But an actor with any utility function $U$ whose policy set contains non first-best policies will with positive probability choose such a sub-optimal policy (since the actor randomizes uniformly over its policies). □

We can immediately prove existence of equilibria, which we will use throughout the paper without explicitly stating so:

**Lemma 5** (Existence)**.** *An equilibrium $(U^*_A, \Pi^*_{U_A})$, and hence a second best policy set always exists.*

*Proof.* We assumed $\Theta$ and $A$ are finite. Hence the set of policy functions $\pi\colon \Theta \to A$ is finite. Hence for any $U_A$, $\Pi^*_{U_A}$ as defined in stage 2 of the game exists. Moreover, there is a finite set of such $\Pi^*_U$ (despite there being an uncountable number of utility functions). Since the expected utility of the selector depends on $U_A$ only through $\Pi^*_{U_A}$, there is therefore a finite number of expected levels of $U_S$ that can be obtained. This obviously implies that the argmax in stage 1 of the game is non-empty. □

For analytic simplicity, we will sometimes consider examples where the actor is able to choose a first-best policy set, despite constraints on its prior information. Surprisingly perhaps, it is possible for the actor's utility function to be different from the selector's while still resulting in a first-best policy.

## 3.4 Causal models

So far, we have not specified the "internal structure" of the action set $A$, the outcome set $O$, or the function $f$, and have just assumed them to be arbitrary. We are now going to assume that the set of actions is a Cartesian product of *sub-actions*: There is a set of *sub-action* sets $\mathcal{A} = \{A_1, ..., A_n\}$, where $A = A_1 \times A_2 \times ... \times A_n$. Each sub-action set $A_i$ will denote some *dimension* along which the actor can act. For example, for a self driving car, we might have $A = Break \times Gas \times Steer$, where $Break = \{0, 1\}$ denotes whether to break or not, $Gas = [0, 1]$ gives the degree by which to accelerate, and $Steer = [-90, 90]$ gives the degrees by which to steer. Hence the actor might decide to break, accelerate at 100%, and steer 45 degrees left at the same time.

Similarly, the outcome set $O$ consists of a set of *sub-outcome* sets or *outcome variables* $\mathcal{O} = \{O_1, ..., O_m\}$, where $O = O_1 \times O_2 \times ... \times O_m$. As a toy example, we might imagine that $O = Travel\ time \times Damage\ to\ car \times Pedestrian\ hurt \times Traffic\ rule\ violations = [0, \infty] \times [0, 1] \times \{0, 1\} \times \mathbb{N}$.

We are going to assume that there is a *causal relation* between these outcome variables, which we first explain informally. So far, we have simply defined $f : A \to O$ as the function that determines the outcome, but we will give this function a *causal structure* (after an informal discussion, we will define this formally below in definition 3): some outcome variables may only depend on the actor's chosen action *through* other outcome variables. For example, consider the causal structure in figure 4.

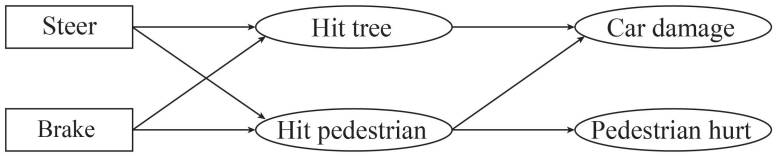


**Figure 4.** *Informal example causal diagram.* Squares denote sub-action sets: $A = Steer \times Break = [-90, 90] \times \{0, 1\}$, while circles/ovals denote sub-outcome sets, or outcome variables: $O = Hit\ tree \times Hit\ pedestrian \times Car\ damage \times Pedestrian\ hurt = \{0, 1\}^4$. The actor will select an element of $Steer \times Break$, such as $(-30, 0)$, meaning steer 30 degrees left and do not break. Arrows denote that *some* dependency from one variable to the other. Intuitively, whether the pedestrian will get hurt depends on the actions of the actor that drives the car, but only *through* the variable of whether the car actually hits the pedestrian: If you want to know whether the pedestrian is hurt, it is enough to know whether the pedestrian got hit, and one does not additionally need to know the exact steering direction. If the car hits the pedestrian, this will cause *both* damage to the car, *and* the pedestrian to get hurt. But whether the car hits the tree or not is not relevant to whether the pedestrian is hurt.

A *causal structure* such as the one in figure 4 specifies the functional *dependency* between the variables. However, it does not specify the *exact* functional relationships. We will denote the *direct* causal effects described by a causal structure as functions $\tilde{f}_{O_i}$ (annotated with a tilde).

For example, in the causal diagram in figure 4, we have

$$f_{Pedestrian\ hurt}(s,b) = \tilde{f}_{Pedestrian\ hurt}(\tilde{f}_{Hit\ pedestrian}(s,b))$$

for the given sub-actions $s \in Steer = [-90, 90]$ and $b \in Break = \{0, 1\}$ that the actor chooses. A causal describes that this is the functional dependency relation between the variables, but a *causal model* will specify the exact functional relations. For example, for figure 4, we might define $hp = \tilde{f}_{Hit\ pedestrian}(s,b) = \begin{cases} 0 & \text{if } b = 1 \text{ or } s < -45 \\ 1 & \text{if otherwise} \end{cases}$, and $\tilde{f}_{Pedestrian\ Hurt}(hp) = hp$, to describe that the car will hit the pedestrian if it does not break or steer sharply leftward, and that the pedestrian will get hurt if the car hits it. We summarize this notation for convenience before defining *causal structure* and *causal model* formally:

**Notation.** *We will write $f_i$ or $f_{O_i}$ (without a tilde), for the "total" effect of an action $a$ on the outcome. This is simply the $i$'th index of the $f$ that we have introduced in section 3.1. We write $\tilde{f}_i$ or $\tilde{f}_{O_i}$ (with a tilde), for the "direct" effect.*

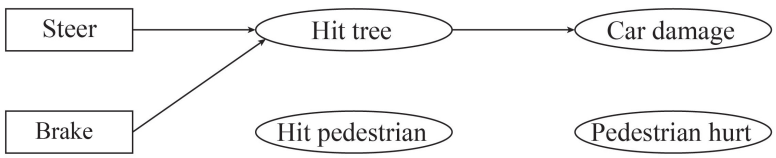


**Figure 5.** *Informal example causal diagram.* The same example as in figure 4, but in a different state of the world. In this state of the world, there is no pedestrian in sight. Hence, no matter what action the actor takes, it cannot hit or hurt any pedestrians.

The family of functions $\tilde{f}$ (one function for each sub-outcome) forms a "functional causal model" (Pearl, 2000).

We can visualize causal structures and models as "causal graphs" as in figure 6.a (Pearl, 2000). There are many alternative ways to represent causal relations graphically to the one chosen here (Dawid, 2002). We use the notation of "causal influence diagrams", where action nodes are represented as squares, and non-action nodes (nodes representing outcome variables in our case) represented as circles (Howard & Matheson, 1981).[28] However, for our purposes we will interpret a causal diagram not as representing a probability distribution as is common (Howard & Matheson, 1981; Dawid, 2002; Everitt et al., 2019), but as representing a deterministic functional model in a state of the state $\omega$, as described in section 3.4. We can thus annotate the causal diagram with functions to specify the exact causal model in a state $\omega$ as in figure 6.b.

We formalize this by using the definitions of "causal structure" and "causal model", with only slight adaptations for our purposes, from Pearl (2000):

[28]The analysis of the agent's environment in terms of causal structures, and causal influence diagrams is not new to machine learning. See for example (Everitt et al., 2019).

**Definition 3** (Causal structure)**.** *A causal structure $D$ of a tuple of sub-action sets $\mathcal{A} = \{A_1, ..., A_n\}$ and sub-outcome variables $\mathcal{O} = \{O_1, ..., O_m\}$, is a directed acyclic graph (DAG) where each node corresponds to a distinct variable in $V = \mathcal{A} \cup \mathcal{O}$, where each link will be used in definition 4 to represent a direct functional relationship among the corresponding variables, and where there are no incoming links to any of the variables in $\mathcal{A}$. The "parent nodes" $P_D(O_i)$ of $O_i$ will denote the set of variables from which there are incoming links to $O_i$.*

We will call the links in a causal structure "causal links".

**Definition 4** (Causal model)**.** *A causal model is a tuple $(D, \tilde{f})$ where $D$ is a causal structure of $\mathcal{A}$ and $\mathcal{O}$, and $\tilde{f}$ is a family of functions representing certain functional relationships between the variables $V = \mathcal{A} \cup \mathcal{O}$, such that it is compatible with $D$: For each variable $O_i \in \mathcal{O}$, we have a function $\tilde{f}_{O_i} : P_D(O_i) \to O_i$.*

*In state $\omega$, a causal model $(D^\omega, \tilde{f}^\omega)$ represents $f^\omega$ if the composition of the family of functions $\tilde{f}^\omega$ is equal to $f^\omega$: Given some $a \in A$, if for each $i$ we let $o_i = f_{O_i}(a)$, then $o_i = \tilde{f}_{O_i}(p(O_i))$, where $p(O_i) \in P_D(O_i)$ is the projection of $(a, x)$ onto $P_D(O_i)$.*[29]

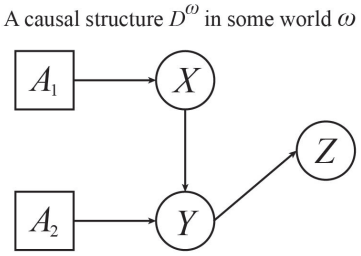


**Figure 6.a.** A *causal structure* is a graph that describes how the sub-outcomes depend on each other and on sub-actions. Squares enclose sub-action sets, circles enclose sub-outcome sets (outcome variables). This diagram implies: $x = \tilde{f}_X(a_1), y = \tilde{f}_Y(a_2, x), z = \tilde{f}_Z(y)$, for some family of functions $\tilde{f}$.

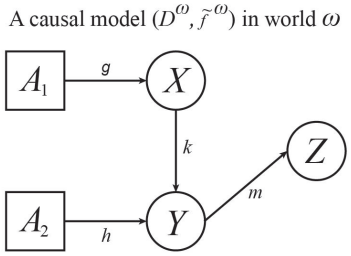


**Figure 6.b.** A *causal model* is a further specification of a causal structure. It gives the exact functional relations between the variables, and can be written by annotating the links with functions. Unless stated otherwise this means the functions combine additively. This diagram implies that $x = g(a_1), y = h(a_2) + k(x), z = m(y)$.

We will usually start out with a causal model for each state $\omega$, and *define* $f$ to be the composition of that causal model, and will speak of *the* causal model representing $f$.

**Notation.** *Consistent with notation used e.g. by Mas-Colell, Whinston & Green (1995), we will sometimes use the abbreviation $p_{-a}$ to represent the tuple $p$ except $a$. That is, we will abbreviate $\tilde{f}_{O_i}(p_1, ..., a, ..., p_n)$ as $\tilde{f}_{O_i}(a, p_{-a})$. We will similarly abbreviate $\{O_k \in P_D(O_i) | O_k \neq O_j\}$ as $P_D(O_i)_{-O_j}$.*

[29]Note that if for some variable $X$, $P_D(X) = \emptyset$, then $\tilde{f}_X$ is a "0-ary" function, i.e. a constant. We could alternatively have explicitly defined "exogenous" variables at the cost of additional complexity.

### 3.4.1 Knowledge of causal models

The central mechanism described in this paper is based on the assumption that the actor and selector have asymmetric information and bounded rationality. In particular, the actor will have a less complete knowledge of the causal model after observing signal $\theta$, than the selector would if the selector could observe $\theta$. The selector will never actually make this observation, but will reason about what the observation *would* imply, in order to choose the utility function $U$ that it will program into the actor.

We formalize this asymmetric information by assuming that the actor and selector have different knowledge of the causal model, given information $I_A, \theta_i$ and $I_S, \theta_i$ respectively.

For example, if the actor of a self-driving car is unable to observe given a signal $\theta_i$ whether there is a pedestrian in view, then it won't be able to distinguish between the causal graphs in figure 4 and 5 respectively, but the selector *would* be able to distinguish them, were it to observe the signal.

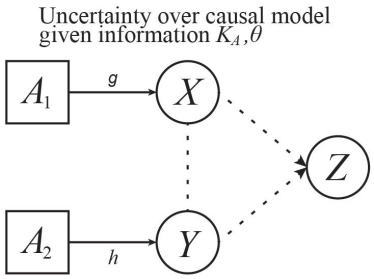


**Figure 6.c.** An actor with information set $I_A, \theta$ may have only partial information about the model. Dotted lines indicate that this agent does not know causal relations between the variables $X, Y, Z$. Using lemma 6, the agent knows $x = g(a_1) + u_x, y = h(a_2) + u_y, z = u_z$ for exogenous $u_x, u_y, u_z$.

We will say that a causal model is consistent with information $I \subseteq \Omega$, if there is a state of the state $\omega \in I$ such that $f^\omega$ is represented by that causal model. Similarly, we say that a causal structure is consistent with $I$ if there is some causal model with that causal structure that is consistent with $I$.

For simplicity, in our examples we will consider the case where the players have no information at all about a certain causal link:

**Definition 5** (No known causal relation)**.** *We say that an agent with information set $I$ knows no causal relation between variables $X$ and $Y$, if for any causal structure $D$ that is consistent with $I$, whenever $X \in P_D(Y)$, we have that $\mathbb{P}[\tilde{f}_Y(x, p(Y)_{-X}) = y|I, D] = \mathbb{P}[\tilde{f}_Y(x, p(Y)_{-X}) = y|I, D, \tilde{f}_W]$ where $W = \mathcal{O} - \{Y\}$ and that this does not depend on $x$, and similarly for whenever $Y \in P_D(X)$.*

We will use a lemma that will simplify the analysis in the examples in section 4. This will mean that we can simply treat the ignorance of a player about a causal link as an "exogenous variable", so that we only have to explicitly consider a fewer causal models.

**Lemma 6** (Unknown causal relations as exogenous variables)**.** *An actor that knows no causal relation between $X$ and $Y$ behaves as if it knows that there is no causal relation between $X$ and $Y$, and that there are exogenous random variables $u_X, u_Y$ influencing $X$ and $Y$.*

The formal proof is technical due to "bookkeeping" required for the different causal links, so we leave it in the appendix. However the idea is conceptually simple: For any possible causal model consistent with the actor's information in which there is a link from $X$ to $Y$, we can construct a new causal model where simply select an arbitrary variable $\tilde{x} \in X$, and define $\tilde{f}'_Y$ in this new causal model as equal to $\tilde{f}_Y$ in the original one, except we force the parameter $x = \tilde{x}$. This removes the connection between $X$ and $Y$, but due to the actor's ignorance about this connection, it does not change the actor's probability distribution over outcomes.

In the examples we will usually assume the causal effects combine additively, so that $\tilde{f}_Y(p(Y)) = g(p(Y)_{-X}) + h(\tilde{x})$. $h(\tilde{x})$ is then a random variable that does not depend on any of the variables in $\mathcal{O}$ and hence is "exogenous". We will use the notation $u_X$ for $h(\tilde{x})$. Similarly we have $u_Y$ from the models in which there is a causal link from $Y$ to $X$.

We will then have *distributions* over such causal models depending on the knowledge of the actor and selector as in section 3.4.1. We will thus specify this knowledge graphically: The knowledge that the players have over the causal model can be derived completely from these diagrams as in figure 6.c. Representing such distribution over causal models symbolically based on the diagram is simplified by lemma 6.

# 4 Toy Example Problems

The central question we are interested in was stated in section 3: Given a set of sub-action sets $\mathcal{A}$ and outcome variables $\mathcal{O}$, and a certain state of knowledge of the actor and selector about the causal relationship between them, what utility function $U : O \to \mathbb{R}$ should the selector give the actor?

It turns out that it is optimal for the selector to give the actor a "proxy" utility function $U^*_A$, which causes the actor to act optimally despite its limited knowledge. We will show how to derive a $U^*_A$ under somewhat general conditions. This utility function may deviate significantly from $U_S$. In earlier work, we have termed the resulting $U^*_A$ "pseudo-aligned" with $U_S$ (Hubinger et al., 2019), since it causes the actor to take actions that score high on $U_S$, but only because the selector chose it to exploit a causal relation between $U$ and $U_S$: If the environment changes, this alignment between $U$ and $U_S$ will disappear.

Perhaps the simplest form of proxy "pseudo-alignment", is when $U$ is defined on a variable that causes the variables on $U_S$ as a side-effect, as in example 1.[30] Our main result will in fact be a generalization of this example, and it formalizes on a simple intuition: The actor has limited knowledge to reason about the "far away" consequences of its actions, but has "control" over the direct consequences of its actions. On the other hand, the selector has more information about the far away consequences, because those are more "stable." The result is a kind of "division of labor": The selector "communicates" to the actor what the far-away causal processes are, by integrating this information into the actor's utility function.

**Example 1.** Assume the example selector-actor problem represented in figure 7, with one sub-action set $A = \{-1, 1\}$ and $O = X \times Y$ where $X = Y = \{-1, 1\}$, and

[30] We have called this *side-effect alignment* (Hubinger et al., 2019).

the selector's utility function is $U_S((x,y)) = y$. Moreover, the training environment consists of two possible input signals: $\Theta_S = (\theta_1, \theta_2)$, each with probability 0.5. Figure 7 describes the game in four causal diagrams, one for each signal and each of the two players.

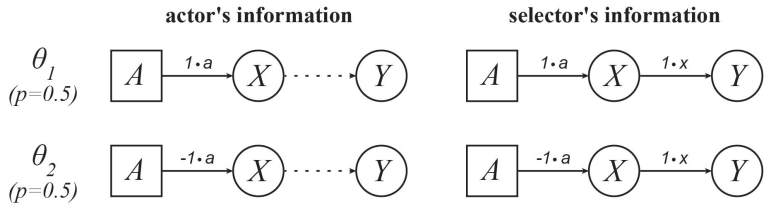


**Figure 7. (Example 1).** An example problem that induces misalignment: There are two variables, $X, Y$, with $U_S(x,y) = y$, two signals $\Theta_S = (\theta_1, \theta_2)$, and the actor does not know the causal connection between $X$ and $Y$, no matter which signal she receives. The first-best choice for the selector is to have the actor optimize for $X$, so that, as a side-effect, his actions will optimize the selector's goal $Y$ as well. On the other hand, the actor would not know what to do if made to optimize for $Y$.

*Solution.* The selector maximizes $\mathbb{E}[U_S(f(\pi_U(\theta)))|I_S] = \mathbb{E}[\tilde{f}_Y(\tilde{f}_X(\pi_U(\theta)))|I_S]$. For both $\theta_1, \theta_2$, the selector knows that $\tilde{f}_Y(x) = x$, so this equals $\mathbb{E}[\tilde{f}_X(\pi_U(\theta))|I_S]$. We use the sum rule of probability to expand this into:

$$\sum_{\theta \in \Theta_S} \mathbb{E}[\tilde{f}_X(\pi_U(\theta))|I_S, \theta] \cdot \mathbb{P}[\theta|I_S] = \mathbb{E}[\tilde{f}_X(\pi_U(\theta))|I_S, \theta_1] \cdot 0.5 + \mathbb{E}[\tilde{f}_X(\pi_U(\theta))|I_S, \theta_2] \cdot 0.5$$

Now it is easy to show that $U^*((x,y)) = x$ is a *first-best* utility function choice from the perspective of the selector: The actor has perfect information about $\tilde{f}_X$ for both $\theta_1, \theta_2$, so that $\mathbb{E}[\tilde{f}_X(a)|I_S, \theta_i] = \mathbb{E}[\tilde{f}_X(a)|I_A, \theta_i]$ for $i = 1, 2$. Therefore if the actor chooses $\pi$ to maximize $\mathbb{E}[f_X(\pi(\theta))|I_S, \theta]$ for each $\theta$, then this maximizes $\mathbb{E}[U(f(\pi(\theta)))|I_S]$ w.r.t. $\pi$, so that $U^*$ is first-best.

Moreover, it is easy to show that for the selector to choose the fully aligned $U = U_S$ would be suboptimal. Note that the actor does not know the causal relation $\tilde{f}_Y$ for both $\theta_1, \theta_2$. Therefore, by lemma 6 we have $\mathbb{E}[\tilde{f}_Y(\tilde{f}_X(a))|I_S, \theta_i] = \mathbb{E}[u_Y|I_S, \theta_i]$ for some exogenous $u_Y$ for all $a$ and $i = 1, 2$. This is independent of $a$, so that every $a \in A$ maximizes this. This means that choosing $U((x,y)) = y$ would make the actor choose a uniformly random action. By lemma 4, $U_S$ is a suboptimal utility function. □

In example 1, the actor did not know any causal path from its actions to $U_S$. One might hope that if the actor knows some such path, i.e. knows *some* of the influence paths from its actions to the selector's goal, then it would be sub-optimal not to make use of this information by having $U$ be at least increasing in $U_S$. Example 2 shows a very extreme toy example of how this can fail. This is an example of "correlation neglect",[31] but in this example, the selector can "correct" for the correlation neglect,

[31]Spiegler has analyzed this concept in the context of bounded rationality as imperfect knowledge of causal graphs (Spiegler, 2016)

by giving the actor the *inverse* of its own utility function, and still have the actor perform a first-bet policy.

**Example 2.** Assume the example problem represented in figure 8, with one sub-action set $A = \{-1, 1\}$ and $O = X \times Y \times Z$ where $X = Y = Z = \{-1, 1\}$, and the selector's utility function is $U_S((x, y, z)) = z$. Moreover, the training environment consists of two possible input signals: $\Theta_S = (\theta_1, \theta_2)$, each with probability 0.5.

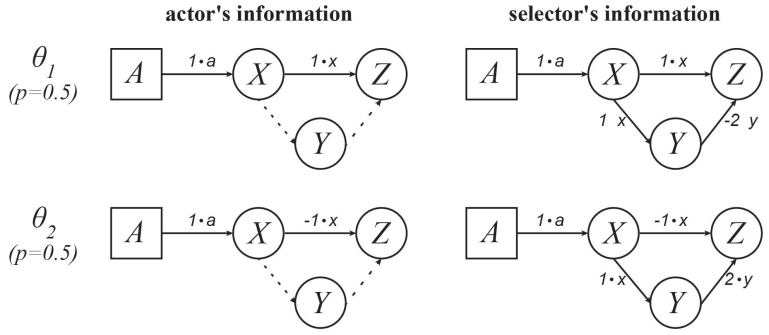


**Figure 8 (Example 2).** An extreme illustrative example of misalignment: $U_S(x, y, z) = z$, and the actor knows the direct causal effect on $Z$, but is ignorant about the indirect effect, via $Y$. The indirect effect is perfectly anti-correlated with the direct effect, and twice as strong. As a result, it is optimal for the selector to give the actor the *opposite* of its own utility function. Giving the actor the fully aligned one ($U = U_S$) results in the worst-possible policy.

*Solution.* We can see from that diagram that the selector maximizes $\mathbb{E}[U_S(f(\pi_U(\theta)))|I_S] = \mathbb{E}[\tilde{f}_Z(\tilde{f}_X(\pi_U(\theta)), \tilde{f}_Y(\tilde{f}_X(\pi_U(\theta))))|I_S]$. Conditional on $\theta_1$, the selector knows that $\tilde{f}_Z(x, y) = x - 2y$, and for $\theta_2$ that $\tilde{f}_Z(x, y) = -x + 2y$, and moreover for both $\theta_i$ that $\tilde{f}_Y(x) = x$. So the selector's utility can be expanded as follows:

$$\mathbb{E}[U_S(f(\pi_U(\theta)))|I_S] = \mathbb{E}[-\tilde{f}_X(\pi_U(\theta))|I_S, \theta_1] \cdot \frac{1}{2} + \mathbb{E}[\tilde{f}_X(\pi_U(\theta))|I_S, \theta_2] \cdot \frac{1}{2}$$

By assumption, the actor has perfect information about $\tilde{f}_X$, so we can replace $I_S$ with $I_A$ here: this is equal to $\mathbb{E}[-\tilde{f}_X(\pi_U(\theta))|I_A, \theta_1] \cdot \frac{1}{2} + \mathbb{E}[\tilde{f}_X(\pi_U(\theta))|I_A, \theta_2] \cdot \frac{1}{2}$. Using lemma 6 we can show that the actor will maximize this if the selector gives it the utility function $U^*(x, y, z) = -U_S(x, y, z) = -z$: For $U^* = -U_S$ the actor's expected utility can be expanded as

$$\mathbb{E}[U_S(f(\pi_U(\theta)))|I_A] = \mathbb{E}[\tilde{f}_X(\pi_U(\theta)) + u_Z|I_A, \theta_1] \cdot \frac{1}{2} + \mathbb{E}[-\tilde{f}_X(\pi_U(\theta)) + u_Z|I_A, \theta_2] \cdot \frac{1}{2}$$

If we take out the $u_Z$'s from the expectations, then we see that

$$\mathbb{E}[U_S(f(\pi_U(\theta)))|I_S, \theta] = -\mathbb{E}[U_S(f(\pi_U(\theta)))|I_A, \theta].$$

It is clear from this that the selector can get a *first-best* policy by letting the actor maximize $-U_S$. In fact, giving the actor the fully aligned utility function would result in the *worst possible* policy for the selector. □

One might hope that if the actor has all information about the selector's goal variable $X$, and there are no stable causal relations between $X$ and other variables, then the selector will not select a pseudo-aligned utility function. This is not necessarily the case however. There may be "redundant" variables. In the following example, it is optimal for the selector to set $U_A = U_S$, but there are additional utility functions. This example illustrates the problem of uniqueness, which I will address further in section 5.3.

**Example 3.** Assume the example problem represented in figure 9, with one sub-action set $A = \{-1, 1\}$ and $O = X \times Y$ where $X = Y = \{-1, 1\}$, and the selector's utility function is $U_S(x, y) = x$. Moreover, the training environment consists of only one input signal: $\Theta_S = (\theta_1)$, and that both actor and selector know that $\tilde{f}_Y() = 1$ (i.e. $y = 1$).

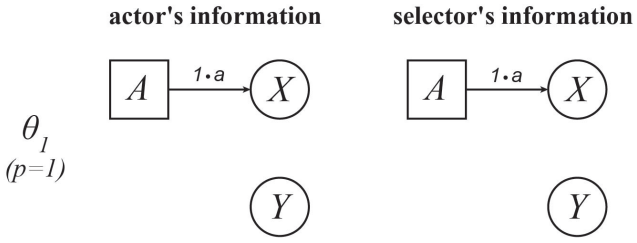


**Figure 9 (Example 3).** The aligned utility function $U(x, y) = x$ is an optimal choice for the selector, but so is $U(x, y) = x \cdot y$, since $y > 0$ in all states $\omega$ in $I_S$.

*Solution.* Since both actor and selector have perfect information about $\tilde{f}_X$, it is trivial to show that $\mathbb{E}[U_S(f(\pi_U(\theta)))|I_S] = \mathbb{E}[\tilde{f}_X(\pi_U(\theta))|I_S, \theta] = \mathbb{E}[\tilde{f}_X(\pi_U(\theta))|I_A, \theta] = \mathbb{E}[U_S(f(\pi_U(\theta))|I_A]$. Hence, trivially, $U^*(x, y) = U_S(x, y) = x$ is a first-best choice for the selector.

However, it is also trivial to show that $\tilde{U}(x, y) = x \cdot y$ is also a first-best choice: $\mathbb{E}[\tilde{f}_X(\pi_U(\theta)) \cdot \tilde{f}_Y()|I_A] = \mathbb{E}[f_X(\pi_U(\theta)) \cdot 1|I_A]$. Obviously this means that $\arg\max_{a \in A} \mathbb{E}[f_X(a)|I_A, \theta] = \arg\max_{a \in A} \mathbb{E}[f_X(a) \cdot f_Y(a)|I_A, \theta]$, so that $\tilde{U}$ is also a first-best choice for the selector.

In this environment, the actor will make the same choices as the aligned one, but as soon as he moves to an environment where $y < 0$, then he will start minimizing rather than maximizing the selector's utility function. □

In all the above examples, the optimal utility function induced an optimal policy function. A natural conjecture might be that if the combined knowledge of the selector and the actor's signals is enough to pinpoint the optimal action, then there is a utility function that induces that action. However, it is not necessarily the case that there exists any utility function at all that induces an optimal policy function. That is, there are situations where the optimal policy function cannot be generated by any agent, as a result of the limited prior information of the actor. Intuitively, this happens when the information is "reversed" compared to example 1, namely when the far-away causal links are known to the actor, but the immediate effects are not.

**Example 4.** Consider the example problem in figure 10. It is analogous to example 1, except now, the actor's knowledge is "reversed": the actor does not know the direct effect of its actions, but knows the causal relation between the outcome variables. Similarly to example 1 we have $A = \{-1, 1\}$, and $O = X \times Y = \{-1, 1\}$, and the selector's utility function is $U_S((x, y)) = y$. Moreover, the training environment consists of two possible input signals: $\Theta_S = (\theta_1, \theta_2)$, each with probability 0.5.

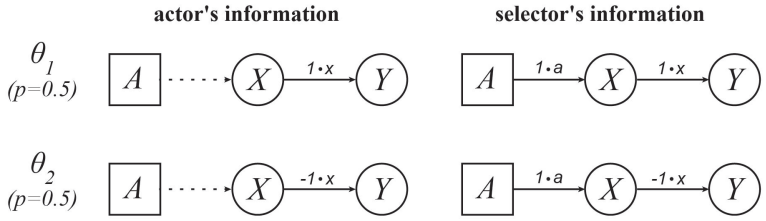


**Figure 10 (Example 4).** An example problem analogous to example 1, but with the actor's knowledge "reversed": the actor does not know the direct effect of its actions, but knows the causal relation between the outcome variables. As a result, the limited knowledge that the actor has is useless: No matter what utility function the selector chooses, the actor won't know how to optimize it.

*Solution.* For any utility function $U$ that the actor could maximize, we have

$$\mathbb{E}[U(f(\pi_U(\theta)))|I_A] = \mathbb{E}[U(\tilde{f}_X(\pi_U(\theta)), \tilde{f}_Y(\tilde{f}_X(\pi_U(\theta))))|I_A].$$

But using lemma 6 and the fact that the actor does not know $\tilde{f}_X$, this is equal to $\mathbb{E}[U(u_X, \tilde{f}_Y(u_X))|I_A]$ for some random variable $u_X$ that is independent of $\pi_U(\theta)$.

Hence for any utility function $U$, we have $\arg\max_{a \in A} \mathbb{E}[U(f(a))|I_A, \theta] = A$. Therefore, no matter what utility function the actor has, the actor will always choose a random action from a uniform distribution from $\mathcal{A}$. Therefore trivially, all utility functions are optimal. □

# 5 Results

We will present two main "mechanisms" by which the actor can be misaligned from the selector. The first one is based primarily on the bounded rationality of the actor, and generalizes primarily example 1. In terms of machine learning, this is about the computational and architectural constraints of the actor. The second one is based on the selector's limited information and bounded rationality. In terms of machine learning, this is about the limitations of the selector's dataset.

## 5.1 Main result: Proxy utility functions

In this section we present the main result of the paper. The result brings together the main ideas presented in the examples of section 4: We give an optimal utility function $U^*$ for a general case, where $U^*$ is only defined on a subset of variables over

which the actor has knowledge. We *informally* call these "proxy variables" and the resulting utility function the "proxy utility function" in line with our earlier work (Hubinger et al., 2019), because it is defined not necessarily on the variables the selector actually cares about, but on those that happen to have a causal relation with them.

However, in order to present a clean result, the assumptions on the information of the actor are quite restrictive: We will assume that for any particular variable $O_i$, the actor either knows perfectly what the effects of its actions on $O_i$ are, or is perfectly ignorant about it. Moreover, we assume that the actor does not have more "control" over the variables in some situations (i.e. for some $\theta$) than in others (we call this uniform control). If we didn't make this assumption, the selector could make use of correlations between the actor's level of control over the "proxy variables", and the causal effects of the proxy variables, which would complicate the result.

The result is intended to conceptually clarify the mechanism by which the actor's utility function will be misaligned with $U_S$: Informally, the selector "projects" its utility function as it is defined on the entire set of variables, onto the proxy variables, through its knowledge of the expected effect of those proxy variables on the other variables.

We first state the definitions:

**Definition 6** (Perfect ignorance)**.** *The actor has perfect ignorance about how to influence sub-outcome variables $\mathcal{Y}$ with $Y = \prod_{Y_i \in \mathcal{Y}} Y_i$, if their conditional distributions $\mathbb{P}[f_Y(a) = y | I_A, \theta]$ are independent of $\theta$ and $a$.*

**Definition 7** (Perfect information)**.** *The actor has perfect information about how to influence sub-outcome variables $\mathcal{X}$ if for all signals $\theta$, and all actions $a$, there is an $x \in X = \prod_{X_i \in \mathcal{X}} X_i$, such that $\mathbb{P}[f(a) = x | I_A, \theta] = 1$. We then define $f^\theta(a) = x$.*

In figure 11 we have illustrated an example diagram where the actor has perfect information about variables $\mathcal{X}$ but perfect ignorance about variables $\mathcal{Y}$. The latter follows from lemma 6: Since the actor knows no causal connections from $\mathcal{A}$ or $\mathcal{X}$ to $\mathcal{Y}$, it will simply treat the variables in $\mathcal{Y}$ as being influenced by exogenous random variables. Hence it has no knowledge about how to influence those variables (it is perfectly ignorant about them).

**Definition 8** (Uniform control)**.** *The actor has uniform control over a set of variables $\mathcal{X}$, if it has perfect information over $\mathcal{X}$, and there is a subset $X_C \subseteq X = \prod_{X_i \in \mathcal{X}} X_i$ such that $f^\theta_X(A) = X_C$ and is injective, for all $\theta \in \Theta_S$.*

Intuitively, uniform control captures the notion that the actor is "consistently competent" (or consistently incompetent) at controlling the variables $\mathcal{X}$, rather than competent some of the time, and incompetent at other times. It means that the actor can always achieve the same outcomes for $X$, no matter the state of the world $\omega$ (or no matter the signal $\theta$). Consider a stylistic example: A robot is on a beach, and depending on the state of the world, it may either rain or not rain. Moreover, one of the outcome variables is "does the robot get wet?" If the robot has an umbrella, then it has uniform control over this variable: If it rains, it can decide to get wet or not, by opening the umbrella or not. But if it does not rain, it also can decide

to get wet or not, by jumping in the sea or not. No matter the state of the world, it has the same options with respect to this variable. Note that the robot *also* has uniform control if it does not have an umbrella, and it always rains in every state of the world. In this case, the robot is forced to get wet, but is forced so "uniformly", i.e. in all states.

Uniform control over outcome variables $\mathcal{X}$ is therefore a quite strict assumption and will mostly not be achieved perfectly in practice. While we do not consider this realistic, the notion may hold approximately in practice. Moreover, it can serve as a *baseline* simple case.

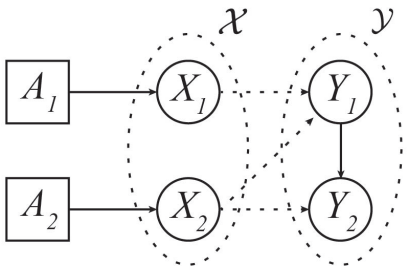


**Figure 11.** An example diagram for the assumptions in theorem 1 on the knowledge of the actor. The actor has perfect information about a subset $\mathcal{X}$ of sub-outcome variables, and perfect ignorance about another subset $\mathcal{Y}$ (since there are no known causal paths from sub-action nodes to sub-outcome nodes). Moreover the actor knows that $\mathcal{Y}$ does not directly depend on $A$.

Before stating the main result, we give an intuition for it:

**Intuition** (for theorem 1)**.** Theorem 1 will assume that the actor has uniform control over a set of sub-outcome variables $\mathcal{Y}$, but perfect ignorance about the other variables $\mathcal{Y}$. Intuitively, this means that it is as if the actor can essentially *choose* a sub-outcome $(x_1, ..., x_n)$, rather than choosing an action. Therefore, no matter what utility function $U$ is given to the actor, the actor will always make the same choice of $(x_1, ..., x_n)$ (more precisely, will always randomize in the same way over such tuples). Hence, the selector, in choosing the utility function of the actor, can simply ignore all the causal structure between $\mathcal{A}$ and $\mathcal{X}$, because the actor "will take care of it". On the other hand, the actor is perfectly ignorant about the causal effect of $\mathcal{X}$ on $\mathcal{Y}$, so the selector has to "take care of it". So we have a *division of labor*: The selector tells the actor what sub-outcomes in $X$ to aim for, taking into account the structure between $\mathcal{X}$ and $\mathcal{Y}$; And the actor decides what actions in $A$ to pick, based on the structure between $\mathcal{A}$ and $\mathcal{X}$. So the selector can simply "project" its $U_S$ onto the variables in $\mathcal{X}$ to achieve the second-best.[32]

Intuitively, this also illustrates why we need the uniform control assumption: If the actor had control in some situations and not in others, then the selector would have to take this into account, and think about how exactly the actor would have to optimize over the proxy variables. It could no longer just "leave it to the actor".

[32]The theorem thus gives a simple formalization within the context of this model, of what in earlier work we have called "pre-computation" of proxy objectives (Hubinger et al., 2019).

We now state this formally as the first main result of our paper, and leave the proof in the appendix:

**Theorem 1** (Optimal utility function)**.** *Assume that the space of outcomes $\mathcal{O} = \mathcal{X} \cup \mathcal{Y}$ is split into "proxy variables" $\mathcal{X}$ and other variables $\mathcal{Y}$ and letting $X = \prod_{X_i \in \mathcal{X}} X_i$ and $Y = \prod_{Y_i \in \mathcal{Y}} Y_i$, where $\mathcal{Y}$ causally does not depend directly on $A$.*[33] *Also assume that the actor has perfect ignorance about $\mathcal{Y}$, and perfect information about, and uniform control over $\mathcal{X}$. Then it is optimal for the selector to give the actor the utility function $U^*$ that is a "projection" of its own utility function onto the proxies $\mathcal{X}$:*

$$U^*(x, y) = U^*_X(x) = \mathbb{E}[U_S(x, \tilde{f}_Y(x))|I_S].$$

**Remark.** This result gives a formalization of the main mechanism by which the actor is misaligned, as it was introduced informally in section 1: The actor has bounded rationality, in the form of not being perfectly able to predict the consequences of its actions. The selector on the other hand, is able to predict these consequences for the set of signals $\Theta_S$ that it expects to see. Because of this, the selector will choose a misaligned utility function that "incorporates" this information, to compensate for the bounded rationality of the actor. Note that the theorem states that the selector makes this choice on the basis of its predictions of the signal $\theta$. But because the selector itself is boundedly rational, in its ability to predict $\theta$, the true state of the world that the agent finds itself in may be very different from the selector's expectation, where the causal relation between $U_A$ and $U_S$ breaks down. The outcome for the human is that there is now an AI agent that pursues an objective that is different from the one that the human specified, and who will potentially take deliberate actions that turn out to be against the interest of the human, especially in situations that the selector didn't predict.

## 5.2 Illustrating proxy utility functions

To illustrate the theorem, we provide two examples. Firstly, we repeat example 1, but solve it much more simply using theorem 1.

**Example 1** (*Repeated for convenience*)**.** Assume the example selector-actor problem represented in figure 7, with one sub-action set $A = \{-1, 1\}$ and $O = X \times Y$ where $X = Y = \{-1, 1\}$, and the selector's utility function is $U_S((x, y)) = y$. Moreover, the training environment consists of two possible input signals: $\Theta_S = (\theta_1, \theta_2)$, each with probability 0.5. Figure 7 describes the game in four causal diagrams, one for each signal and each of the two players.

[33] Recall this means that $f(a) = \tilde{f}(f_X(a)) = (f_X(a), \tilde{f}_Y(f_X(a)))$.

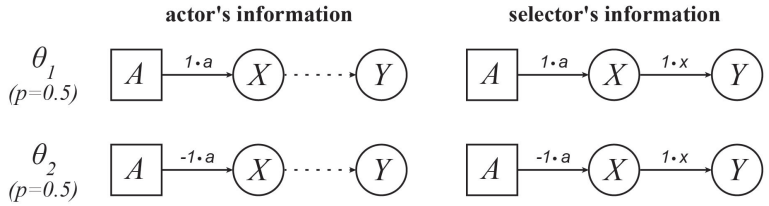


**Figure 7 (Example 1,** *Repeated for convenience*).** An example problem that induces misalignment: There are two variables, $X, Y$, with $U_S(x, y) = y$, two signals $\Theta_S = (\theta_1, \theta_2)$, and the actor does not know the causal connection between $X$ and $Y$, no matter which signal she receives. The first-best choice for the selector is to have the actor optimize for $X$, so that, as a side-effect, his actions will optimize the selector's goal $Y$ as well. On the other hand, the actor would not know what to do if made to optimize for $Y$.

*Solution.* Note that the actor has perfect information over $X$, perfect ignorance over $Y$, and has uniform control over $X$. The latter is true because the image of $\tilde{f}_X(A)$ is equal to $\{-1, 1\}$ for both $\theta_1$ and $\theta_2$. Therefore, by theorem 1, $U^*(x, y) = U^*_X(x) = \mathbb{E}[U_S(x, \tilde{f}_Y(x))|I_S] = \mathbb{E}[\tilde{f}_Y(x)|I_S] = x$. Hence, $U^*(x, y) = x$ is optimal. □

**Remark.** Note that the solution using theorem 1 gives the same utility function as our somewhat more elaborate reasoning in the original solution to the example. Moreover, the earlier reasoning used the fact that the selector has no uncertainty about the effect of $X$ on $Y$. The next example won't have this property.

We now apply the result with a slightly more elaborate example:

**Example 5.** Assume two sub-action sets $A_1, A_2 = \{-1, 1\}$, and $X_1, X_2, Y_1, Y_2 = \{-3, -2, -1, 0, 1, 2, 3\}$, with $X = X_1 \times X_2$ and $Y = Y_1 \times Y_2$, and the causal structure given in figure 12. Moreover let the selector's utility function be $U_S(x_1, x_2, y_1, y_2) = y_1 \cdot x_2 + y_2$.

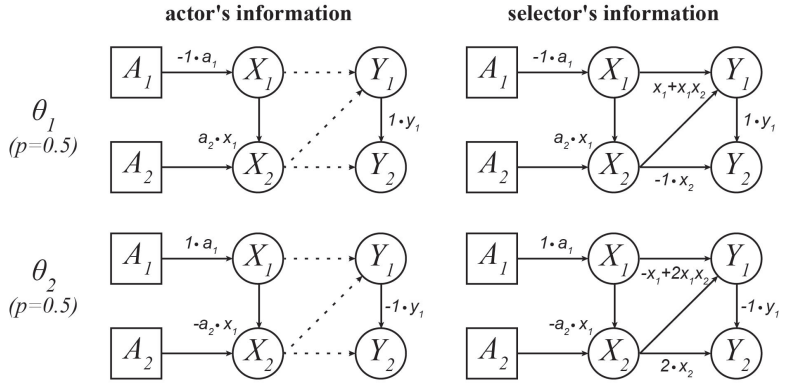


**Figure 12. (Example 5).** The actor has perfect information about $X_1, X_2$ and perfect ignorance about $Y_1, Y_2$. Moreover, it has uniform control over $X_1, X_2$. Hence theorem 1 applies.

*Solution.* Firstly, note that clearly, the actor has perfect information about $X_1, X_2$, and perfect ignorance about $Y_1, Y_2$. Moreover, we can also show that the actor has

uniform control over $X_1, X_2$: for $\theta = \theta_1, \theta_2$, we have that the image $f_X(A)$ is the same (namely $f_X(A) = \{-1, 1\}$). This is easy to see: Since $\tilde{f}_{X_1}(a_1)$ gives $a_1$ given $\theta_1$ or $-a_1$ given $\theta_2$. whatever $\theta$ is, the actor can select $a_1 \in A_1$ so as to choose $x_1 \in \{-1, 1\}$. Then similarly since $\tilde{f}_{X_2}(a_2, x_1)$ gives $a_2 x_1$ or $-a_2 x_1$, it can select $a_2$ to get $x_2$ from $\{-1, 1\}$.

Now we can apply theorem 1 (we pre-multiply by 2 to get rid of the 0.5 probabilities of the $\theta_i$): $U^*(x_1, x_2, y_1, y_2) = U^*_X(x_1, x_2) = 2 \cdot \mathbb{E}[U_S(x, \tilde{f}_Y(x))|I_S] = 2 \cdot \mathbb{E}[\tilde{f}_{Y_1}(x_1, x_2) \cdot x_2 + \tilde{f}_{Y_2}(\tilde{f}_{Y_1}(x_1, x_2), x_2)|I_S] = [(x_1 + x_1 x_2)x_2 + (x_1 + x_1 x_2) - x_2] + [(-x_1 + 2x_1 x_2)x_2 - (-x_1 + 2x_1 x_2) + 2x_2]$. We can check that this simplifies to $U^*(x_1, x_2, y_1, y_2) = 3x_1 x_2^2 + 2x_1 - x_1 x_2 + x_2$. □

**Remark.** Informally, note that the utility function $U^*_A$ in example 5 is more *complex* than the original utility function $U_S$. Intuitively, the selector has *integrated* its information about the causal relation between $\mathcal{X}$ and $\mathcal{Y}$ into the utility function $U^*_A$, thereby adding "*complexity*" to it.

## 5.3 Uniqueness

Theorem 1 provides one specific optimal utility function. It formalizes a "mechanism" by which the utility function of the actor can be misaligned with that of the selector, as the result of the bounded rationality of the actor. However, we have already seen in example 3 that there is another reason why the actor's utility function can be misaligned: There may be a "redundancy" in the environment of the actor that causes it to obtain a utility function that is only "partially" aligned. In other words, even if $U_S \in \mathcal{U}^*$, that is even if the actor is not impeded by bounded rationality to be aligned, then the actor may still be misaligned.

We already explained why non-uniqueness is a problem, and why in general it is problematic for the human if the actor has a misaligned utility function: Due to the selector's imperfect foresight about the signals that the actor will receive, a utility function $U^*$ may be optimal given the selector's imperfect predictions, but not actually be optimal given the signal that the actor ends up observing. We therefore state the problem as follows:

**Problem** (Uniqueness)**.** *Even if $U_S \in \mathcal{U}^*$, the selector might not set $U_A = U_S$, because there are many other (mis-aligned) $U \in \mathcal{U}^*$. While these $U$ perform optimally given the signals $\Theta_S$ that the selector expects, as soon as the actor observes a signal $\theta \notin \Theta_S$, this is no longer true, but the actor still pursues the misaligned $U$. What conditions do we need to guarantee that $U_S$ is unique in $\mathcal{U}^*$ (up to a preference-preserving transformation)?*

Our uniqueness conditions will be very strict, and are analogous (but not equivalent) to a revealed preference condition: The selector needs to test the actor in various circumstances . Before we state our result, we give an example that will illustrate the idea in figure 13.

**Example 6.** Consider the simple causal model in figure 13 where the actor has perfect information. The actor has one action set, Steering-Angle = {left, middle, right},

resulting in outcomes $O = \{\text{evade-left, kill-pedestrian, evade-right}\}$ respectively. The selector's utility function is

$$U_S(o) = \begin{cases} -100 & \text{if } o = \text{kill pedestrian} \\ 0 & \text{if } o = \text{evade-left, or } o = \text{evade-right} \end{cases}.$$

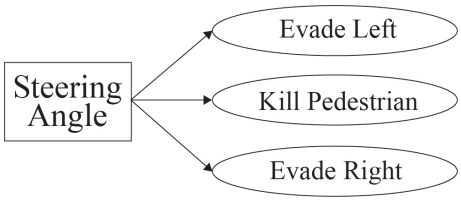


**Figure 13 (Example 6).** *Non-uniqueness.* The selector strongly prefers the car to evade either left or right over killing the pedestrian. It would be nice if the actor would adopt this preference too. But this is not guaranteed, even for an actor with perfect information and a first-best utility function.

*Solution.* We have two options for the actor to avoid killing the pedestrian: either evade left or evade right. The selector does not care which option the actor picks, so long as it does not kill the pedestrian. Consider a utility functions as follows:

$$U_A(o) = \begin{cases} -100 & \text{if } o = \text{evade left} \\ 0 & \text{if } o = \text{kill pedestrian} \\ 1 & \text{if } o = \text{evade right} \end{cases}$$

An actor with this utility function has very different preferences from the selector. It would have been desirable if we could show that $U_A \notin \mathcal{U}^*$. That is, it would have been desirable if in this scenario, any $U \in \mathcal{U}^*$ would agree with the selector that killing the pedestrian is the worst outcome. However, the utility function $U_A$ as defined above, is in fact first-best: $\mathbb{E}[U_S(f(\pi_{U_A}(\theta_1)))|I_S] = \mathbb{E}[U_S(f(\text{right}))|I_S] = 0$, which is a maximum. □

**Intuition** (for theorem 2)**.** The source of the misalignment problem in the example in figure 13 is that the actor has too much control. This would be less problematic if the selector required the actor's optimal choice set to coincide with its own. But recall from lemma 4 that the actor's policy set (if it results in a first-best policy) need only be a subset of, not necessarily equal to, the selector's policy set: $\Pi^*_{U_A} \subseteq \Pi^*_S$, since the selector only looks at whether the actor makes a good decision (in this case, steer right, rather than hit the pedestrian). Perhaps counter-intuitively, if the actor is "consistently competent" then this *increases* the opportunity for misalignment, since it increases the actor's access to such redundant options about which the selector is indifferent.

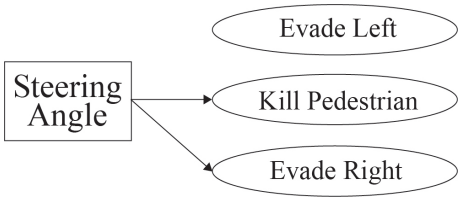


**Figure 14.** *Non-uniqueness.* An actor with utility function $U_A$ from the example in figure 13 would rather kill the pedestrian than evade left, despite acting optimally in a situation where it also had the option to evade right.

We now state the definition of unique alignment,

**Definition 9** (Weak alignment)**.** *We have unique weak alignment if $U_S$ is unique in $\mathcal{U}^*$ up to preservation of strict preferences:*

$$\textit{for all } U \in \mathcal{U}^*, \qquad \textit{for all } o_1, o_2 \in O, \quad U_S(o_1) < U_S(o_2) \textit{ implies } U(o_1) < U(o_2).$$

It turns out that the only way to guarantee weak alignment, is by simply "testing" the actor on every possible outcome. See figure 15. This condition is obviously very strict, and this cannot reasonably be expected to hold in any non-trivial practical application. So we may interpret this as a kind of "practical" impossibility result.[34]

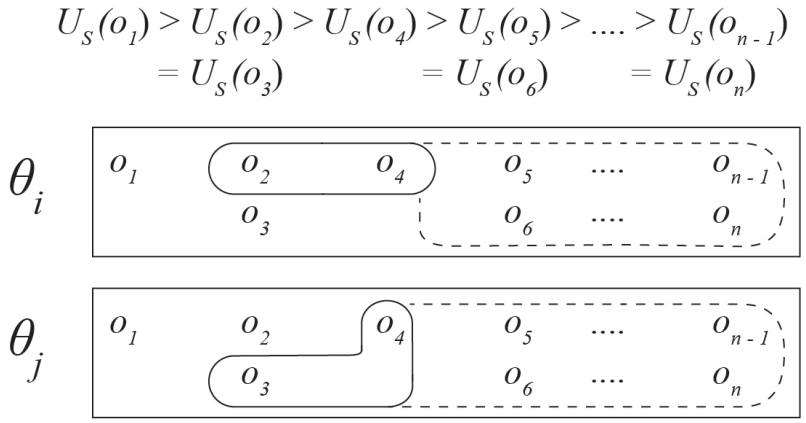


**Figure 15.** Number the outcomes in $O$ as $o_1, o_2, ..., o_n$ such that the ordering coincides with the order by which they are preferred by the selector. In order to guarantee weak alignment, the selector needs to "test" the actor for every outcome and every outcome directly to the right of it. In order to know that $o_2$ and $o_3$ are preferred by the actor to $o_4$, there need to be two separate $\theta_i, \theta_j$ that induce the encircled outcomes as possible outcomes in $f^{\theta_{i,j}}(A)$. Those in the solid line must at least be in it, and those in the dotted line may, but don't have to. Therefore, these two sets "test" whether $U(o_2) > U(o_4)$, and whether $U(o_3) > U(o_4)$ etc. This would not work if $o_3$ was also in the set, because there needs to be a *unique* maximimum.

**Theorem 2** (Unique alignment)**.** *Assume complete information. We have unique weak alignment if and only if for every outcome $o_1$, and every outcome $o_2$ that is directly dispreferred to $o_1$, there is a signal that gives the actor the option between $o_1, o_2$, and where $o_1$ is the unique available optimum for $U_S$.*

[34]This result illustrates the problem of "lack of training data" in the context of this model.

**Remark.** Theorem 2 looks similar to classical revealed preference theorems. However, the difference results from the fact that in order for the actor's utility function $U_A$ to be optimal, the set of actions over which it randomizes need merely be a subset, rather than identical to, the set of first-best actions, as we recalled from lemma 4. Unlike in standard choice theory, the actor's utility function cannot be pinpointed up to *monotonic transformation*. Rather, there will always be utility functions in $\mathcal{U}^*$ that are not monotonic transformations of $U_S$ (we state this as lemma 7 in the appendix). This is because when the selector is indifferent between outcomes (as in figure 13), this does not mean that the actor will also be indifferent. We leave the proof in the appendix.

**Remark.** Like theorem 1, the assumptions of theorem 2 are not likely to hold (note that fulfilling the requirement requires the selector to predict almost as many signals as there are outcomes). However, it points to a *type* of misalignment, as we've illustrated already in example 6: Even if the actor is highly competent, and seems to optimize for the same objective function as the selector, this may be "unstable", due to "redundancies" in the environment.[35]

We finally note that, for broadly the same reasons as discussed in the general perfect information case, the function $U^*$ in theorem 1 is not unique in $\mathcal{U}^*$. In fact, any utility function whose maximum in $X$ is contained in that of $U^*$:

**Proposition 1.** *Assume the assumptions of theorem 1 hold, and let $U^*$ be as defined therein. Then any $U$ such that $\arg\max_{x \in X_C}(U(x)) \subseteq \arg\max_{x \in X_C}(U^*(x))$ is also in $\mathcal{U}^*$.*

# 6 Conclusion

The analysis in this paper illustrates how machine learning systems can be seen as an agency relationship between a selector and actor. This moves away from the standard approach of modeling "learning agents" as monolithic entities that take actions to try to maximize their reward. There is a tension between this abstraction and the fact that in reality, selectors choose algorithms, not actions, to maximize their expected reward. Rather than abstracting away from this by considering it as an approximation error to an optimal choice of policy, we explicitly model the selector's choice as an algorithm. We can then frame problems of machine learning in terms of the bounded rationality and information constraints that induce an agency cost. This approach can abstract away from technical details, while still modeling the essential characteristic of machine learning, and focus on general considerations about the interaction between selector and actor.

[35]This is not an unrealistic idea. Currently existing machine learning systems suffer from a similar, though not identical problem. For example, the decisions of self driving cars have been shown to be contingent on very "weird" conditions: While the car may decide to turn right during a sunny day, it might suddenly decide to turn left when the sky is darker, even though the rest of the environment is the same, and even though it was not explicitly selected by the selector to have this preference (Pei, Cao, Yang & Jana, 2017; Huang, Kwiatkowska, Wang & Wu, 2017)

This paper has introduced an initial model of the actor and selector as boundedly rational agents in a two-stage game, and a resulting mis-alignment between their utility functions. The central mechanism is as follows: The first effect is due to the actor's bounded rationality: The actor has to take decisions in an environment, and the selector has an interest in choosing the actor's utility function so as to maximize its own expected utility. But because the actor has limited understanding of the causal structure of its environment, the selector is incentivized to integrate its information about that causal structure into the utility function of the actor. As a result, the actor does not pursue the same goals as the selector, but pursues "proxy goals". The main formal result of this paper gives conditions under which the selector simply "projects" its expected utility onto the proxy variables whose causal structure the actor understands.

The second effect is due to the selector's bounded rationality and imperfect information, and coincides closely with the training data of the selector: The selector has a limited domain of signals for which it has knowledge (informally, the training data). Beyond that domain, we assume that the selector has no information. The result is that the selector optimizes for a limited amount of causal structures, which causes it to give proxy goals that are specifically tailored to the environments over which the selector has information. Moreover, it causes the set of optimal utility functions that it could give the actor to not be uniquely aligned with its own, even if the actor has perfect information. This is simply a reframing of "limited training data" in the context of this model. Both of these effects result in the creation of an AI agent that pursues goals that are different from those specified by the human programmers. This model is thus a particular formalization of the inner alignment problem (Hubinger et al., 2019).

This paper has thus introduced an initial model of learning algorithms and AI agents in a principal-agent relation. There is potential for a broader agency theoretic analysis of machine learning systems, and this paper can thus be seen as an initial step in that direction. Agency relations are a central phenomenon in human societies, and have been studied extensively by economists. But as artificially intelligent agents become more advanced, and are given an increasingly influential role in society and the economy, agency relations between AI systems may reach the level of importance of agency relations between humans.

# References

Amodei, D., Olah, C., Steinhardt, J., Christiano, P., Schulman, J., & Mané, D. (2016). Concrete Problems in AI Safety.

Andrychowicz, M., Baker, B., Chociej, M., Józefowicz, R., McGrew, B., Pachocki, J., Petron, A., Plappert, M., Powell, G., Ray, A., Schneider, J., Sidor, S., Tobin, J., Welinder, P., Weng, L., & Zaremba, W. (2019). Learning Dexterous In-Hand Manipulation. Technical report, OpenAI.

Andrychowicz, M., Denil, M., Colmenarejo, S. G., Hoffman, M. W., Pfau, D., Schaul, T., Shillingford, B., & De Freitas, N. (2018). Learning to learn by gradient de-

scent by gradient descent. In *30th Conference on Neural Information Processing Systems (NIPS)*, (pp. 1–9)., Barcelona.

Bostrom, N. (2014). *Superintelligence: Paths, Dangers, Strategies*. Oxford: Oxford University Press.

Dawid, A. P. (2002). Influence Diagrams for Causal Modelling and Inference. *International Statistical Review*, *70*(2), 161–189.

Drexler, K. E. (2019). Reframing Superintelligence: Comprehensive AI Services as General Intelligence. Technical report, FHI.

Everitt, T., Ortega, P. A., Barnes, E., & Legg, S. (2019). Understanding Agent Incentives using Causal Influence Diagrams. Part I: Single Action Settings. *DeepM*.

Fama, E. F. (1980). Agency Problems and the Theory of the Firm. *Journal of Political Economy*, *88*(2), 288–307.

Floreano, D. & Wood, R. J. (2015). Science, technology and the future of small autonomous drones. *Nature*, *521*, 460–466.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. Cambridge, Massachusetts and London, England: The MIT Press.

Hart, O. (1989). Incomplete Contracts. In J. Eatwell, M. Milgate, & P. Newman (Eds.), *Allocation, Information and Markets* (pp. 163). London and Basingstoke: The Macmillan Press Limited.

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. Technical report, Microsoft Research.

Hodges, C., An, S., Rahmani, H., & Bennamoun, M. (2019). Deep Learning for Driverless Vehicles. In V. Balas, S. Roy, D. Sharma, & P. Samui (Eds.), *Handbook of Deep Learning Applications. Smart Innovation, Systems and Technologies, vol 136* (pp. 83–99). Springer, Cham.

Howard, R. A. & Matheson, J. E. (1981). Influence Diagrams. *The Principles and Applications of Decision Analysis*.

Huang, X., Kwiatkowska, M., Wang, S., & Wu, M. (2017). Safety Verification of Deep Neural Networks. *arXiv*.

Hubinger, E., van Merwijk, C., Skalse, J., Mikulik, V., & Garrabrant, S. (2019). (To be published) Learned Optimization in Advanced Machine Learning Systems. *Machine Intelligence Research Institute*.

Jensen, M. C. & Meckling, W. H. (1976). Theory of the Firm: Managerial Behavior, Agency Costs and Ownership Structure. *1Journal of Financial Economics*, *3*(4), 305–360.

Kaelbling, L. P., Littman, M. L., & Moore, A. W. (1996). Reinforcement Learning: A survey. *Journal of Artificial Intelligence Research*, *4*, 237–285.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(1), 35.

Li, Y., Vinyals, O., Heess, N., Buesing, L., Racanière, S., Reichert, D., Weber, T., Wierstra, D., & Battaglia, P. (2017). Learning model-based planning from scratch. *DeepMind*.

Mas-Colell, A., Whinston, M. D., & Green, J. R. (1995). *Microeconomic Theory* (1 ed.). New York Oxford: Oxford University Press.

OpenAI (2018). OpenAI Five.

Parkes, D. C. & Wellman, M. P. (2015). Economic reasoning and artificial intelligence. *Science*, *349*(6245), 267–272.

Pearl, J. (2000). *Causality: Models, Reasoning, and Inference*. New York: Cambridge University Press.

Pei, K., Cao, Y., Yang, J., & Jana, S. (2017). Towards Practical Verification of Machine Learning: The Case of Computer Vision Systems. *arXiv*, 1–16.

Peters, J., Vijayakumar, S., & Schaal, S. (2005). Natural Actor-Critic. *Neurocomputing*, *71*(7-9), 280–291.

Rahwan, I., Cebrian, M., Obradovich, N., Bongard, J., Bonnefon, J.-F., Breazeal, C., Crandall, J., Christakis, N. A., & Couzin, I. D. (2019). Machine behaviour. *Nature*, *568*(11), 26.

Ross, S. A. (1973). The Economic Theory of Agency: The Principal's Problem. *The American Economic Review*, *63*(2), 134–139.

Russell, S. (2015). Take a stand on AI weapons. *Nature*, *521*.

Russell, S. & Norvig, P. (2010). *Artificial Intelligence A Modern Approach* (3 ed.). Pearson.

Schwab, K. (2016). *The Fourth Industrial Revolution*. Geneva: World Economic Forum.

Schwarting, W., Alonso-Mora, J., & Rus, D. (2018). Planning and Decision-Making for Autonomous Vehicles. *Annual Review of Control, Robotics, and Autonomous Systems*, *1*, 187–210.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Driessche, G. V. D., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., & Lanctot, M. (2016). Mastering the game of Go with deep neural networks and tree search. *Nature*, *529*, 484–489.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., & Bolton, A. (2017). Mastering the game of Go without human knowledge. *Nature*, *550*(7676), 354–359.

Spiegler, R. (2016). Bayesian Networks and Boundedly Rational Expectations. *The Quarterly Journal of Economics*, 1243–1290.

Spiegler, R. (2019). Can Agents with Causal Misperceptions be Systematically Fooled? *Journal of the European Economic Association*, *00*(0), 1–35.

Srinivas, A., Jabri, A., Abbeel, P., Levine, S., & Finn, C. (2018). Universal Planning Networks. *arXiv*.

Stiglitz, J. E. (1989). Principal and Agent. In J. Eatwell, M. Milgate, & P. Newman (Eds.), *Allocation, Information and Markets* (pp. 241). London and Basingstoke: The Macmillan Press Limited.

Sutton, R. S. & Barto, A. G. (1998). *Reinforcement Learning: An Introduction*. Cambridge & London: The MIT Press.

Sxepesvari, C. (2010). Algorithms for Reinforcement Learning. *Synthesis Lecture on Artificial Intelligence and Machine Learning*, 1–98.

Vinyals, O., Babuschkin, I., Chung, J., Mathieu, M., Jaderberg, M., Czarnecki, W. M., Dudzik, A., Huang, A., Georgiev, P., & Powell, R. (2019). AlphaStar: Mastering the Real-Time Strategy Game StarCraft II.

Vinyals, O., Ewalds, T., Bartunov, S., Georgiev, P., Vezhnevets, A. S., Yeo, M., Makhzani, A., Küttler, H., Agapiou, J., & Schrittwieser, J. (2017). StarCraft II: A New Challenge for Reinforcement Learning. *DeepMind*.

Williamson, O. E. (1975). *Markets and Hierarchies*. London: Collier Macmillan Publishers.

# 7 Appendix

# Appendix A: Proofs and Results

## A. Section 3.4 (causal models)

*Proof of Lemma 6.* I restate the lemma:

**Lemma 6.** *An actor that knows no causal relation between $X$ and $Y$ behaves as if it knows that there is no causal relation between $X$ and $Y$, and that there are exogenous random variables $u_X, u_Y$ influencing $X$ and $Y$.*

For any state of the world $\omega$, let $(D^\omega, \tilde{f}^\omega)$ be the causal model that represents $f^\omega$. We assume that there is a link from $X$ to $Y$ in $D^\omega$. For the case when there is a link from $Y$ to $X$ this requires merely swapping the variables. Define $(D^{\omega\prime}, \tilde{f}^{\omega\prime})$ as follows: $D^{\omega\prime}$ is the same DAG as $D^\omega$, but with the relation between $X$ and $Y$ removed. To define $\tilde{f}'$, we pick arbitrary $x \in X$, and define $\tilde{f}^{\omega\prime}_Y$ by plugging $x$ for the variable $X$ into $\tilde{f}^\omega_Y$: i.e. for arbitrary $p'_Y \in P_{D^{\omega\prime}}(Y)$, define $\tilde{f}^{\omega\prime}_Y(p_Y) = \tilde{f}^\omega_Y(x, p'_Y)$. For all other variables $O_i \in \mathcal{O}$, let $\tilde{f}^{\omega\prime}_{O_i} = \tilde{f}^\omega_{O_i}$. Then define $f' : A \to O$ recursively by letting $o_i = f^{\omega\prime}_{O_i}(a) = \tilde{f}^{\omega\prime}_{O_i}(p'_{O_i})$, where $p'_{O_i}$ is the projection of $(a, o)$ to the variables in $P_{D^{\omega\prime}}$.

We will show that for all $O_i$, and any $o_i \in O_i$ and any action $a \in A$, we have $\mathbb{P}[f_{O_i}(a) = o_i|I] = \mathbb{P}[f'_{O_i}(a) = o_i|I]$. This will trivially imply that for any action $a$ we have $\mathbb{E}[U(f(a))|I] = \mathbb{E}[U(f'(a))|I]$, and therefore that the actor behaves as if every causal structure consistent with $I$ has no link between $X$ and $Y$. (Note that the uncertainty about the link between $X$ and $Y$ is, in the adjusted $f'$, incorporated in uncertainty about the effect of other parent variables to $Y$, i.e. uncertainty about $\tilde{f}'_Y$).

Pick any $D$ consistent with $I$. Take any $O_i \in \mathcal{O}$. Let $W$ be the set $\mathcal{O}/\{Y\}$ of outcome variables except $Y$. Since $A$ and $O$ are finite sets, that means that the set of functions $\prod O_j \in P_D(O_i) \to O_i$ is finite for each $O_i$. Hence we can expand the $\mathbb{P}[f_{O_i}(a) = z|I, D]$ as:

$$\sum_{\tilde{f}_W} \sum_{y \in Y} \mathbb{P}[f_{O_i}(a) = z|I, D, \tilde{f}_W, f_Y(a) = y] \cdot \mathbb{P}[f_Y(a) = y|I, D, \tilde{f}_W] \cdot \mathbb{P}[\tilde{f}_W|I, D]$$

We will do a substitution for the first two factors in the summand based on the following equations:

1. $\mathbb{P}\left[f_{O_i}(a) = z|I, D, \tilde{f}_W, f_Y(a) = y\right] = \mathbb{P}\left[f'_{O_i}(a) = z|I, D, \tilde{f}_W, f'_Y(a) = y\right]$ : To see this, we construct an expression $\phi$ as follows: We expand

$$f_{O_i}(a) = \tilde{f}_{O_i}(a, f_{P_{D'}(O_i)_1}(a), ..., f_{P_{D'}(O_i)_n}(a)).$$

We continue expanding the arguments in the same way, except for the variable $Y$ (i.e. wherever the expression contains $f_Y(a)$, we do not expand it). Then after a finite number of expansion steps (since there are a finite number of

variables), we get an expression containing only compositions of functions $\tilde{f}_{O_j}$ for different $O_j \neq Y$, and $f_Y$. We let $\phi$ be shorthand for the resulting expression. Moreover, let $\phi'$ similarly be the shorthand for the same expansion of $f'_{O_i}(a)$.

Note that due to the conditionals, every occurance of $f_Y(a)$ in $\phi$ can be replaced with $y$. And by definition of $\tilde{f}'$, all the occurrances of $f_{O_j}$ can be replaced with $f'_{O_j}$ (since $O_j \neq Y$). If we let $\tilde{\phi}$ be the resulting expression, then $\tilde{\phi} = z$ becomes conditionally independent of both $f_Y(a) = y$ and $f'_Y(a) = y$, so that we can replace the former with the latter to get $\mathbb{P}\left[\phi = z | I, D, \tilde{f}_W, f_Y(a) = y\right] = \mathbb{P}\left[\tilde{\phi} = z | I, D, \tilde{f}_W, f'_Y(a) = y\right]$. We can then replace all the occurances of $y$ in $\tilde{\phi}$ with $f'_Y(a)$ to get $\phi'$, and then collapse $\phi'$ back to $f'_{O_i}(a)$, so that the latter expression equals $\mathbb{P}[f'_{O_i}(a) = z | I, D, \tilde{f}_W, f'_Y(a) = y]$, showing the result.

*2.* $\mathbb{P}\left[f_Y(a) = y | I, D, \tilde{f}_W\right] = \mathbb{P}\left[f'_Y(a) = y | I, D, \tilde{f}_W\right]$ : To see this, expand $f_Y(a)$ *once* on the left hand side to get $\mathbb{P}[\tilde{f}_Y(a, f_X(a), f_{P_D(Y)_{-x}}(a)) = y | I, D, \tilde{f}_W]$.

If we expand the arguments of $\tilde{f}_Y$ in this expression as for $\phi$ in step 1, we get expressions containing only compositions of $\tilde{f}_{O_j}$ for different $O_j \neq Y$: Since $D$ is acyclic, they will not contain any occurances of $\tilde{f}_Y$ or $f_Y$. And by definition, for any $O_i \neq Y$ we have $\tilde{f}_{O_i} = \tilde{f}'_{O_i}$, so that we can simply substitute $f_X(a), f_{P_D(Y)_{-x}}(a)$ by $f'_X(a), f'_{P_D(Y)_{-x}}(a)$. Secondly, since the exact functional form of each of the $\tilde{f}_{O_j}$ in the resulting expansion is known within the conditional expression, that means $f_X$ is also known, so that we can replace it with its constant value $x(a)$. Doing these two substitutions gives us $\mathbb{P}[\tilde{f}_Y(a, x(a), f'_{P_D(Y)_{-x}}(a)) = z | I, D, \tilde{f}_W]$.

Now we use the assumption that the actor knows no causal relation from $X$ to $Y$: This means that the previous expression is equal to $\mathbb{P}[\tilde{f}_Y(a, \tilde{x}, f'_{P_D(Y)_{-x}}(a)) = z | I, D, \tilde{f}_W]$ for any $\tilde{x}$, in particular for the $\tilde{x}$ we used to define $\tilde{f}'_Y$. Therefore we can use the definition of $\tilde{f}'_Y$ to derive that the expression is equal to $\mathbb{P}[\tilde{f}'_Y(a, f'_{P_D(Y)_{-x}}(a)) = z | I, D, \tilde{f}_W]$. This is obviously equal to $\mathbb{P}[f'_Y(a) = z | I, D, \tilde{f}_W]$, showing the result.

Substituting these two equations into the summand, and then contracting the summation gives us $\mathbb{P}[f_{O_i}(a) = z | I, D] = \mathbb{P}[f'_{O_i}(a) = z | I, D]$. Since this holds for every $D$ consistent with $I$, we can sum over the $D$'s and get $\mathbb{P}[f_{O_i}(a) = z | I] = \mathbb{P}[f'_{O_i}(a) = z | I]$ which is the desired result.

If the causal effects combine additively, so that $\tilde{f}_Y(p(Y)) = g(p(Y)_{-X}) + h(x)$ for some $g, h$, then $\tilde{f}'_Y(p(Y)) = g(p(Y)_{-X}) + h(\tilde{x})$ where $h(\tilde{x})$ is a random variable that does not depend on any of the variables in $\mathcal{O}$ or $\mathcal{A}$ and hence is "exogenous". We will use the notation $u_X$ for $h(\tilde{x})$. Similarly we have $u_Y$ from the models in which there is a causal link from $Y$ to $X$. □

## A. Section 5.1 (proxy utility functions)

*Proof of theorem 1.* We repeat the theorem for convenience:

**Theorem 1.** Assume that the space of outcomes $\mathcal{O} = \mathcal{X} \cup \mathcal{Y}$ is split into "proxy variables" $\mathcal{X}$ and other variables $\mathcal{Y}$ and letting $X = \prod_{X_i \in \mathcal{X}} X_i$ and $Y = \prod_{Y_i \in \mathcal{Y}} Y_i$, where $\mathcal{Y}$ causally does not depend directly on $A$.[36] Also assume that the actor has perfect ignorance about $\mathcal{Y}$, and perfect information about, and uniform control over $\mathcal{X}$. Then it is optimal for the selector to give the actor the utility function $U^*$ that is a "projection" of its own utility function onto the proxies $\mathcal{X}$:

$$U^*(x, y) = U^*_X(x) = \mathbb{E}[U_S(x, \tilde{f}_Y(x))|I_S].$$

We will proceed in three steps.

*Step 1: For every utility function $U : O \to \mathbb{R}$, there is a utility function $\tilde{U} : O \to \mathbb{R}$ that only depends on variables $\mathcal{X}$ (over which the actor has perfect information), and that induces the same policy functions as $U$.*

Since the actor has perfect information about $X$, the function $f_X(a)$ is known for each $\theta$, and since the actor has perfect ignorance about $Y$, the distribution of $f_Y(a)$ conditional on $I_A, \theta$ does not depend on $a$, nor on $\theta$, so that we can consider $Y$ as a random variable independent of $a$ and $\theta$. Let $\tilde{U}((x, y)) = \mathbb{E}[U(x, f_Y(a))|I_A]$ for arbitrary $a$. Clearly, $\tilde{U}$ only depends on variables in $\mathcal{X}$, and the expected utility of $U$ and $\tilde{U}$ are the same for all $\theta, a$, and hence their optimal action sets are the same. The latter follows from the fact that $\mathbb{E}[\tilde{U}(f(a))|I_A, \theta] = \mathbb{E}[\mathbb{E}[U(f_X(a), f_Y(a))|I_A]|I_A, \theta] = \mathbb{E}[U((f_X(a), f_Y(a)))|I_A, \theta] = \mathbb{E}[U(f(a))|I_A, \theta]$, where we have used the law of iterated expectations.

*Step 2: For any utility function $U$ there exists a subset $X^* \subseteq X$, such that for any $\theta$, $f^\theta_X(A^*) = X^*$, where $A^* = \arg\max_{a \in A} \mathbb{E}[U(f(a))|I_A, \theta]$.*

By step 1, we only need to consider $U$ that only depend on $X$. Since by assumption the actor has perfect information about $X$, that means that for all $\theta$, the actor maximizes $\mathbb{E}[U(f_X(a))|I_A, \theta] = U(f^\theta_X(a))$ w.r.t. $a$. Note that

$$f^\theta_X(\arg\max_a U(f^\theta_X(a))) = \arg\max_{x \in f^\theta_X(A)}(U(x)).$$

And since $f^\theta_X(A) = X_C$ for all $\theta$, that means $f^\theta_X(A^*) = \arg\max_{x \in X_C} U(x) \equiv X^*$.

*Step 3: for all $U$, $\mathbb{E}[U_S(f(\pi_U(\theta)))|I_S] = \frac{1}{|X^*_U|} \sum_{x \in X^*_U} \mathbb{E}[U_S(x, f_Y(x))|I_S]$.*

By step 2, for any $U$, $\mathbb{E}[U_S(f(\pi_U(\theta)))|I_S] = \sum_\theta \mathbb{E}[U_S(f(\pi_U(\theta)))|I_S, \theta] \cdot \mathbb{P}[\theta|I_S] = \sum_\theta \sum_{x \in X^*} \mathbb{E}[U_S(x, \tilde{f}_Y(x))|I_S, \theta] \cdot \mathbb{P}[x|I_S, \theta] \cdot \mathbb{P}[\theta|I_S]$. But by assumption, $\pi_U(\theta)$ is uniformly distributed on $A^*$, and given that for any $x \in X_C$ there is only one action that leads to it (by assumption of uniform control), that means $x$ is uniformly distributed on $X^*$ and independent of $\theta$. Therefore $\mathbb{P}[x|I_S, \theta] = \mathbb{P}[x|I_S] = \frac{1}{|X^*|}$. Hence we can

[36] Recall this means that $f(a) = \tilde{f}(f_X(a)) = (f_X(a), \tilde{f}_Y(f_X(a)))$.

rewrite it as $\frac{1}{|X^*_U|}\sum_{x\in X^*}\sum_{\theta}\mathbb{E}[U_S(x,\tilde{f}_Y(x))|I_S,\theta]\cdot\mathbb{P}[\theta|I_S]=\frac{1}{|X^*_U|}\sum_{x\in X^*_U}\mathbb{E}[U_S(x,\tilde{f}_Y(x))|I_S]$.

We use step 3 to complete the proof: By construction,

$$X^*_{U^*}=\arg\max_{x\in X_C}\mathbb{E}[U_S(x,f_Y(x))|I_S].$$

Hence, $\frac{1}{|X^*_{U^*}|}\sum_{x\in X^*_{U^*}}\mathbb{E}[U_S(x,f_Y(x))|I_S]=\max_{\tilde{X}\subseteq X_C}\frac{1}{|\tilde{X}|}\sum_{x\in\tilde{X}}\mathbb{E}[U_S(x,f_Y(x))|I_S]$. Therefore, by step 3, $\mathbb{E}[U_S(f(\pi_{U^*}(\theta)))|I_S]=\max_U\mathbb{E}[U_S(f(\pi_U(\theta)))|I_S]$, which completes the proof. □

## A. Section 5.3 (uniqueness)

The following result was referred to in the main text:

**Lemma 7.** *Assume complete information. Any $U$ that agrees with strict preferences of $U_S$ is optimal for the selector: $U\in\mathcal{U}^*$.*

*Proof.* Take any $U$ that agrees with strict preferences of $U_S$, so that for any $o_1,o_2\in O$, $U_S(o_1)>U_S(o_2)$ implies $U(o_1)>U(o_2)$. Then for all $\theta$, $\arg\max_{o\in f^\theta(A)}U(o)\subseteq\arg\max_{o\in f^\theta(A)}U_S(o)$. This obviously implies that $U\in\mathcal{U}^*$ (see lemma 4). □

*Proof of theorem 2.* We restate the theorem for convenience:

**Theorem 2.** Assume complete information. We have unique weak alignment if and only if for every outcome $o_1$, and every outcome $o_2$ that is directly dispreferred to $o_1$, there is a signal that gives the actor the option between $o_1,o_2$, and where $o_1$ is the unique available optimum for $U_S$.

For simplicity, we will use the following terms:

**Choice set**: For any signal $\theta$, let the "choice set" $C_\theta=f^\theta(A)$. These are the outcomes achievable given signal $\theta$.

**Choice rule**: Let the "choice rule" $c_U(C)$ of a utility function $U$ given a choice set $C$ be equal to the set $\arg\max_{o\in C}U(o)$. These are the choices that the actor would randomize over if given a utility function $U$.

Moreover, we say that $U$ *agrees with strict preferences of* $U_S$ if for all $o_1,o_2$, $U_S(o_1)>U_S(o_2)$ implies $U(o_1)>U(o_2)$. (Note that this does not imply that $U$ is a monotonic transformation of $U_S$, unless $U_S$ is not indifferent between any two outcomes.)

By lemma 4 we know that for any $U$, we have $U\in\mathcal{U}^*$ if and only if for every signal $\theta\in\Theta_S$ we have $c_U(C_\theta)\subseteq c_{U_S}(C_\theta)$.

First we show $\Longleftarrow$. Assume that for every such $o_1, o_2$, a $\theta$ exists, such that $o_1, o_2 \in C_\theta$, and $c_{U_S}(C_\theta) = \{o_1\}$. Then take any $U$ that does not agree with strict preference of $U_S$ on some $\tilde{o}_1, \tilde{o}_2$. This means that $U(\tilde{o}_1) \leq U(\tilde{o}_2)$. By transitivity, there must exist some $o_1, o_2$ such that $U_S(o_1) > U_S(o_2)$ but $U(o_1) \leq U(o_2)$, and moreover, where no other $\tilde{o}$ exists such that $U_S(o_1) > U_S(\tilde{o}) > U_S(o_2)$. This means by assumption that there is a $\theta \in \Theta_S$ such that $o_1, o_2 \in C_\theta$. Hence if $o_1 \in c_U(C_\theta)$, then $o_2$ must also be in $c_U(C_\theta)$. If $o_1 \notin c_U(C_\theta)$, then some other $\bar{o}$ that is in $c_U(C_\theta)$, but by assumption $o_1$ was the unique element in $c_{U_S}(C_\theta)$. Hence, in either case, $c_U(C_\theta) \nsubseteq c_{U_S}(C_\theta)$. Hence $U \notin \mathcal{U}^*$.

Now we show $\Longrightarrow$. Assume some $o_1, o_2$ with $U_S(o_1) > U_S(o_2)$ and no $\bar{o}$ exists such that $U_S(o_1) > U_S(\bar{o}) > U_S(o_2)$, but there is no $\theta \in \Theta_S$ such that $o_1, o_2 \in C_\theta$ and $o_1$ is the unique element of $c_{U_S}(C_\theta)$. We can construct the following utility function:

$$U(o) = \begin{cases} U_S(o_1) - \delta & \text{if } o = o_2 \\ U_S(o_2) + \delta & \text{if } o = o_1 \\ U_S(o) & \text{otherwise} \end{cases}$$

where $0 < \delta < 0.5 \cdot (U_S(o_1) - U_S(o_2))$. Because there is no $\tilde{o}$ such that $U_S(o_1) > U_S(\bar{o}) > U_S(o_2)$, it is easy to see that $U$ agrees with strict preferences of $U_S$ on all pairs of outcomes, except $(o_1, o_2)$. That is, for any pair $(\bar{o}_1, \bar{o}_2)$ other than $(o_1, o_2)$, if $U_S(o_1) > U_S(o_2)$ then $U(o_1) > U(o_2)$.

Therefore, if $\bar{o} \in \arg\max_{o \in C_\theta} U(o)$, then either $\bar{o} \in \arg\max_{o \in C_\theta} U_S(o)$ as well, or $\bar{o} = o_1$ and $o_1, o_2 \in C_\theta$ and no other $o \in C_\theta$ for which $U_S(o) > U_S(o_1)$. But the latter is precisely what we assumed was not the case. Therefore, for all $\theta$ we have $\arg\max_{o \in C_\theta} U(o) \subseteq \arg\max_{o \in C_\theta} U_S(o)$, so that $U \in \mathcal{U}^*$, even though $U$ does not agree with strict preferences of $U_S$. □

*Proof of Proposition 1.* We restate the proposition for convenience:

**Proposition 1** Assume the assumptions of theorem 1 hold, and let $U^*$ be as defined therein. Then any $U$ such that $\arg\max_{x \in X_C}(U(x)) \subseteq \arg\max_{x \in X_C}(U^*(x))$ is also in $\mathcal{U}^*$.

From the proof of theorem 1, the actor always achieves its maximum on $X^*_U$. Consider $X^*_{U^*} = \arg\max_{x \in X_C} \mathbb{E}[U_S(x, f_Y(x))|I_S]$ as defined in that proof, and simply use the assumption that $\arg\max_{x \in X_C}(U(x)) \subseteq \arg\max_{x \in X_C}(U^*(x))$ and step 3 of the proof to see that $\mathbb{E}[U_S(f(\pi_U(\theta)))|I_S] = \max_{\tilde{U}} \mathbb{E}[U_S(f(\pi_{\tilde{U}}(\theta)))|I_S]$ □

# Appendix B: Extensions

The model in section 3.1 makes the significant simplification that the actor only observes a single signal, and takes a single action. We can generalize this, and consider an arbitrarily complex second stage by generalizing the set of policy functions into some more general set $\Pi$, and the outcomes it results in $O$. In particular, the second stage could be a markov decision process, which is one of the main decision problems considered in machine learning contexts. In this case, $\Pi$ would be the set of functions that map a *history* of observations to an action, and $O$ would be the set of possible state histories. Alternatively, the second stage could itself be a multi-stage game with other agents. In this case $\Pi$ would be the actor's set of strategies, and the chosen strategies of the other players would be incorporated into a more general function $F\colon \Pi \to O$.

Moreover, the model described in section 3.1 assumes that the actor's knowledge is *fixed*. This simplifies the analysis, and still captures some key insight, but is unrealistic, since in reality a learning algorithm is able to select the entire actor's algorithm, which includes all its prior information. Intuitively, one might think of this as the selector also being able to send the actor a "message" during stage 1, though this is only a metaphor. There will however be a limitation on the prior information that the selector can choose (the messages it can send). We can thus generalize the model further, and have the selector select not just a utility function, but an "algorithm" or "mechanism" $M \in \mathcal{M}$, which will capture both the actor's utility function $U_M$ *and* its prior information $I_M$:

We then rewrite the equations describing the two stages in section 3.1 to specify a very general class of two stage games:

$$\mathcal{M}^* = \arg\max_{M \in M} \mathbb{E}[U_S(F(\pi_M))|I_S]$$
$$\Pi^*_M = \arg\max_{\pi \in \Pi} \mathbb{E}[U_M(F(\pi))|I_M]$$

Finally, we can generalize it further, and not assume at all that the actor chooses its actions by maximizing an expected utility function, for example because the selector does not fully delegate decision making to the actor. We would then consider a more general class of algorithms $\mathcal{M}$, without assuming that $\Pi^*_M = \arg\max_{\pi \in \Pi} \mathbb{E}[U_M(F(\pi))|I_M]$.