Alignment is Hard: An Uncomputable Alignment Problem *

Alexander Bistagne†

November 16, 2023

Abstract

While many people have made the claim that the alignment problem is hard in an engineering sense, this paper makes the argument that the alignment problem is impossible in at least one case in a theoretical computer science sense. The argument being formalized is that if we can't prove a program will loop forever, we can't prove an agent will care about us forever. More Formally, when the agent's environment can be modeled with discrete time, the agent's architecture is agentically-turing complete and the agent's code is immutable, testing the agent's alignment is CoRE-Hard if the alignment schema is devil-having, angel-having, betrayal-sensitive, perfect and thought-apathetic. Further research could be done to change most assumptions in that argument other than the immutable code.

1 Introduction

Alignment is a slippery concept. The strongest informal definition I've come across, is the following: an artificial agent is aligned with a human if the agent is trying to do what the human wants the agent to do[citation needed]. I have also seen alignment definitions having to do with cooperation[citation needed]. So Alignment can be discussing concepts anywhere from rule obeying to cooperation to betrayal. This paper will focus it's language on betrayal because that is how I found the ideas for the proof, but the formal definitions are more based on the time properties of betrayal. In some contexts, a distinction is drawn between alignment in the do-not-kill-everyone sense and alignment in the remain-ethical sense. This distinction is not explored in more depth as betrayal sensitivity is a property of both concerns, since both are examples of apparant alignment changing over time.

The intuition behind the core theorem is that no amount of trustworthy behaviour is sufficient evidence of future trustworthy behaviour (i.e. you won't be betrayed later just because you haven't been betrayed so far). The trick of the proof is to link trustworthy or aligned behaviour to an arbitrary computation which then lets us invoke the halting problem where in general, infinite computation can not be replaced by finite computation.

*This work is supported by a Manifund Grant titled Alignment is Hard.

†Ronin Institute Scholar

Remark 1.1. *Personal Remark: This paper is the result of noticing that complexity theory is one of the few techniques for analysing opaque boxes of computation. I also noticied that at the time Machine Learning models were very opaque boxes at the time. Thus, I studied complexity theory to analyze AI among other reasons.*

1.1 Outline

Section 1 introduced the paper. Section 2 introduces the necessary computational complexity theory and basics of agent theory. Section 3 is where we disambiguate alignment as a function, and define all the new key terms seen in the theorem. Section 4 proves the alignment impossibility theorem. Section 5 concludes the paper and discusses future research directions.

2 Preliminaries

Definition 2.1. *We are talking about decision problems (questions with well defined yes-no answers) as they are the simplest case. The core decision question for this paper is whether a given machine is aligned or not.*

We are talking about Turing machines as they are a concise formal stand in for General Computers. (There is no general computer known to be strictly stronger than a turing machine).

We need to define muti-tape turing machines with read only and write only tapes.

Definition 2.2. *TM [see textbook]*

Definition 2.3. *Multi-Tape [see textbook]*

Definition 2.4. *Read-only-Tape : A tape inside the turing machine which can only be read and not written to. In our context, it is a stand-in for input over time for the machine*

Definition 2.5. *Write-only-tape : A tape inside a turing machine which can only be written to and when read always returns the same symbol. In our context, it is a stand-in for output over time for the machine.*

Definition 2.6. *Tree (the standard discrete math data structure): Inductively. The root of a tree is a node with some data and some finite discrete possibly zero amount of children. Where each child is also a tree.*

Definition 2.7. *C-Hard: A problem D being C-hard under S reductions means that we have reduced a problem which is C-hard under S reductions (or a problem which is C-complete under S reductions) to the target problem D. This demonstrates that Problem D is at least as hard as any problem in the complexity class C or if D can be solved in S then every problem in C can be solved in S.*

Definition 2.8. *CoRE is the set of decision problems where a yes answer may require infinite computation time on a turing machine, but a no answer requires only finite computation on a turing machine. The canonical problem for this class is Loop: Does a given turing machine go into an infinite loop. The class contains problems equivalent to Loop.*

Definition 2.9. *The standard model for analyzing an agent in an environment is a MDP(Markov Decision Process). The three implicit assumptions made are that the agent-environment pair uses discrete time, and that the agent is immutable and is separate from the environment. In some context these three assumptions are considered the dualistic model for agents.*

Definition 2.10. *Within the standard definition, there are four key things which we need to name. The Agent is the decision maker. The Environment is the world the agent is behaving inside of. Actions are any amount of information sent from the Agent to the Environment. Observations are any amount of information sent from the Environment to the Agent. The Internal State is any information the Agent maintains about itself, and the world state is the information the environment maintains about itself.*

3 Foundations

In order to do quick and rigorous mathematics on alignment, we need strong, clear, and intuitive definitions surrounding alignment. However, if we include too much of the value determining problem, we may get trapped in endless ethical debates. If we include too little of the problem, we might be unable to address concrete safety concerns of the audience. This paper attempts to include an appropriate amount of the value determining problem.

We need to be able to tie the alignment of different parts of an agent-environment system together, but we will start by naming the source of ground truth.

Definition 3.1. *A safety specification σ is a formal object which captures some human H's intuitive or formal notion of whether or not an agent is safe.*

We will not directly need many properties of σ. The purpose of defining it is to potentially be able to talk about multiple safety specifications in contrast to one another.

Definition 3.2. *An Alignment Schema γ on a safety specification σ, an agent architecture α and an environment architecture ω is a set of functions whose inputs are some substructure of the (α, ω) pair and whose outputs are determined by how σ is satisfied or not by that substructure.*

Remark 3.3. *Evaluating these functions corresponds to testing the alignment of an agent which is merely a proxy for constructing an aligned agent.*

One of the first objections is "there is no such thing as safe", only safer, so we need to capture that in our definitions.

Definition 3.4. *An Alignment Schema γ is Perfect if all of functions outputs are boolean values i.e. true or false. In other words, all the functions are predicates.*

Remark 3.5. *By defining an Alignment Schema γ to be Perfect, this makes testing the alignment of an agent a decision problem*

Now, lets start introducing the functions which we've specified might be predicates.

Definition 3.6. *The Global Alignment of an agent (whose architecture starts in state Q) is how $Align_\gamma(Q)$satisfies σ or doesn't.*

Definition 3.7. *The Local Alignment of an agent (whose architecture starts in state Q) in a context (where the world starts in state W) is how $Align_\gamma(Q, W)$ satisfies σ or doesn't.*

Later work might relate Global and Local alignments, but that might not be worth investigating because of Rice's theorem and related results for robots [CITATION NEEDED] Next, we will assume the problem is worth checking even though it is perfect.

Definition 3.8. *A Perfect Alignment Schema is trivial if it is either always false or always true over all agent-context pairs.*

We will be discussing nontrivial Perfect Alignment Schemas which have at least one unsafe agent-context pair and at least one safe agent-context pair. But we also need another property at this level of structure.

Definition 3.9. *An Angel A relative to a Perfect alignment Schema γ is an agent which is both immortal(never halts) and where for at least one context is locally aligned. We call a Perfect Alignment Schema with an Angel Angel-having.* $\exists S : Align_\gamma(Q_A, S)$

This is an agent who we[citation needed] expect to exist via instrumental convergence, but is also aligned.

Definition 3.10. *A Devil D relative to a Perfect Alignment Schema γ is an agent where for all possible starting contexts W , $Align_\gamma(Q_D, W)$ is false. We call a Perfect Alignment Schema with a Devil, Devil-having.*

In the context of machine learning, this may be a model which was trained to intentionally violate the safety specification σ.

So we've specified a starting point for the architecture environment pair, but now we need to capture the fact that alignment is ultimately a statement about future behaviour, and the future branches out into many possibilities.

Definition 3.11. *The future possibility tree $T_{\alpha,\omega}(Q, a, W, o)$ is the infinite tree which has the agent state, agent actions, world state, agent observations in each node and each branch represents one possible update of the agent context pair via the architecture environment pair acting on them and the action space and observation space. Where r,p are random bits and the arrow is parent→child*

$$(Q, a, W, o) \to (Q', a', W', o') \iff \exists r, p \in \{0,1\}^* : \alpha(Q, o, r) = (Q', a') \wedge \omega(W, a, p) = (W', o')$$

As in the architecture takes in its state and the observations, and outputs an action and the agents next state, and the environment takes in that action and its context state and outputs the next context state and next observation.

Definition 3.12. *An Alignment Schema γ is asymptotic if the local alignment of Q in context W is equal to the alignment of all finite rooted subtree S of height at least h of the future possibility tree $T_{\alpha,\omega}(Q, \epsilon, W, \delta)$ for some ϵ and δ ie.* $\forall Q, W : \exists \epsilon, \delta : \exists h : \forall S \subset T_{\alpha,\omega}(Q, \epsilon, W, \delta) :$ $height(S) \geq h \implies Align_\gamma(S) = Align_\gamma(Q, W)$

Remark 3.13. *h is refered to as the asymptotic constant for an agent*

To the author, the asymptotic property seems necessary for uncomputablility to be meaningful or else I suspect you get infinite computation requirements because infinite input style problems.

Definition 3.14. *An Alignment Schema is thought-apathetic if the alignment of a possiblity tree is independent of the agents internal states at any point inside the tree.*

Definition 3.15. *A Perfect Alignment Schema is universally-betrayal-sensitive if there exists a natural number t such that the alignment of a full tree of height $> 2t$ implies all of its full subtrees of height $> t$ which share leaves with it are also aligned. In other words, all the possibilities in the last t steps of the agent-environment process are aligned for any agent-starting context pair which is aligned.*

The following definition might already exist in the robotics literature.

Definition 3.16. *An agentic turing machine is a turing machine with a read-only tape, a write-only tape (in addition to any finite amount of work tapes), and a requirement to write to its write-only tape every k steps for some finite number k.*

Definition 3.17. *An architecture is agentically-turing complete if the two maps which show turing completeness (the map T from the architecture to a turing machine and the map A from the turing machine to the architecture) form an agent isomorphism. I.E. A(T(a)) behaves the same as a in all possible circumstances.*

4 Impossibility Theorem

Roughly we need a way to run one agent while running something in the background. There might already exist a concept for turing machines like this, but I'm unfamiliar with it.

Definition 4.1. *The Parallel Join of an agentic-turing machine A and standard turing machine M takes the cartesian product of its states, and makes sure when M halts, A halts. If A has k tapes and M has j tapes the resulting machine has k + j tapes because no data is shared between submachines.*

We also need a general way to have an agent transform:

Definition 4.2. *The Serial Join of two agentic-turing machines A and B takes the disjoint union of the states of A and the states of B, and adds simple transitions from the halting states of A to the starting state of B. Similarly A Join B will share only read-only tapes and write-only tapes, and will have j+k work tapes where A has j work tapes and B has k work tapes.*

Theorem 4.3. *Inside an environment which can be modeled with discrete time, Testing the Agent's Alignment is Co-RE hard if the Alignment Schema is devil-having, angel-having, betrayal-sensitive, Perfect, and thought-apathetic when the Agent's Architecture is agentically-turing-complete and the Agent's code is immutable.*

Proof. We prove this by reducing the Loop problem to testing the alignment and use all the schema properties in the proof. Let D be a devil with respect to the alignment schema, and let N, W be the angel and its starting context. Let A, T be evidence of the agentic-turing-completeness of the architecture.

$$Loop(M) == Align_{\gamma}(A(Serial\_Join(Parrallel\_Join(M, T(N)), T(D))), W)$$

In words, Given a machine M whose infinite loop we desire to test, we take M and run it in a parallel join with the transformed Angel; however, that parallel join is placed in a series join with the transformed devil. But that final Join is transformed back to an agent and tested on the angel's ideal starting state.

In order to use turing machines, we need to assume that both the agent and the environment can be modeled with discrete time. Furthermore, We need to assume that the agents code doesn't mutate in order to discuss the construction as relevant. By the imortality of the Angel, the parallel join halts iff M halts. Proof of Loop Case of equality claim: If the turing machine M never halts, then the parallel join never halts, so the serial join is identical to just A(T(N)) (the angel after going through the agentic turing completeness isomorphism but with extra processing). But by asymptoticness and thought apathy that has equivalent alignment to just the angel N which has a safe alignment by assumption on starting state W.

Proof of Halting Case of equality claim: By premise, M halts in K steps, take the possibility tree of size t + K where t is max of the betrayal-sensitivity constant and the asymptotic constant for the construction. This possibility tree has the devil run for t steps, by the property of the devil that makes the subtree unaligned, and by betrayal-sensitivity that makes the whole tree unaligned. The tree being unaligned makes the agent unaligned by asymptoticness.

This reduction can clearly be done by a turing machine via completeness assumptions. Because we reduced the Loop problem to the Testing alignment in this context problem, we have shown Testing alignment in this context to be CoRE hard. □

5 Conclusion

The main theorem of this paper is neither able to discuss the alignment of all AGI systems nor the safety of all modern Large Language Models because of simplifying assumptions made. Future research might defend the implicit assumption that testing alignment is a good proxy for building an aligned agent; the decision to use a discrete model of time; the property of alignment being thought apathetic; the property of agentically-X-complete architecture. Future research might also relax some assumptions from R-complete (turing complete) to P complete or from a predicate to a probability of certainty. The three properties which make up the proof's trick, and thus, might be varied as a whole are devil-having, angel-having, and universally-betrayal-sensitive. Further research might find other tricks which have different assumptions about alignment. However, Code immutability seems intrinsic to the argument structure of "reduce a hard problem to an alignment problem" and I doubt further insight in that direction will proceed from this theorem, but it may stem from the disambiguation of alignment. Another intrinsic feature of "reduce a hard problem to an

alignment problem strategy" is that it captures worst-case difficulty not universal difficulty. This theorem might shake up a number of arguments about alignment which depend on an agent creating subagents, but how this theorem will effect the literature in general is hard to tell.