Blockchain-Based Identity Solutions for Artificial Intelligence: Two-sided Virtual Markets without Trust

John P. Conley[0000-0003-3675-8529]

Vanderbilt University, Nashville, TN 37235, j.p.conley@vanderbilt.edu

Abstract

Many market interactions require sequential trust in which one agent makes an irrevocable commitment, such as making a payment, only after which a counterparty reciprocates with a promised action. Successful markets and institutions include self-enforcing mechanisms to assure compliance. Artificial Intelligence Agents have an array of abilities that could be employed to expand the capabilities and reach of Human Agents. AIs, however, are not like humans. How to characterize their preferences, their identities, and even their individualities, if they have them, is not clear. If AIs cannot be included as agents in mechanisms, then trade and exchange between colloidal and mechanical agents may be impossible. This paper proposes an approach using blockchain that allows the establishment of identities for mechanical agents, and the creation of complete, provable, histories of their actions in a game. It then constructs a mechanism in which peer-to-peer markets between randomly matched mechanical and biological agents work in the sense that cooperation is consistent subgame perfect equilibrium. It also shows that without this blockchain-based foundation, such markets are likely to fail.

Keywords: Artificial Intelligence, Blockchain, P2P Markets Two-sided Markets, Machine to Colloidal Markets, Mechanism Design, Identity, Public Key Encryption

1 Introduction

On the internet, no one knows if you are a dog. Anonymity is often a feature rather than a bug. Joining in political debate, accessing content freely, and participating in communities that some might disprove of without fear of reprisal, protects individual freedom and supports democracy.

In many cases, however, agents benefit from identifying themselves to one another. This is especially true in markets where trust is required. A buyer would rightly be hesitant to send payment to an anonymous online seller in hopes of receiving an item he ordered. Similarly, experts, content makers, and influencers, benefit when they can establish an identity or brand that people recognize as meaningful based on their history of output.

Extending trust to other agents is inherently risky, and human societies have

evolved elaborate institutions and mechanisms to make it possible. In general, trust depends on several things:

- Reputation: A verifiable and credible history of honesty[1], skill, fair dealing, or more generally, conforming to social expectations in the interaction at hand.
- Sanctions: Societies penalize those who are caught being dishonest. In some cases, this is a collective punishment; in others, it takes the form of independent rational decisions on the part of members of the society to refrain from interacting with, or "trusting", agents who have a history of dishonesty.
- Repeated Interactions: If a dishonest agent can simply leave town and start over, sanctions are meaningless. This is probably the main reason that we are more likely to trust people in our own family, tribe, profession, and social, ethnic, or religious group. The inside options for interactions with members of one's own group are more attractive than the outside options, given such trust structures.

All of these elements rest on a foundation of identification. Anonymous agents cannot establish reputations, cannot be targeted for sanctions, and cannot be distinguished from other agents in future interactions.

Up until the last decade or so, agents in virtual spaces were all human, or at least were software or platforms deployed by humans. Machines were not advanced enough to pass the Turing test, and so it was usually clear which was the case. The identity of biological agents typically relied on the control of user and account IDs, login credentials, or URLs.

More recently, mechanical agents, such as AIs and bots, have become more prevenient in virtual spaces. It is now common for such agents to have access to and control of login credentials and user IDs. It is often difficult to distinguish artificial from colloidal agents, dog, human, or otherwise. Nevertheless, Biological and mechanical agents differ in several fundamental ways:

- Biologicals are unique individuals, and are difficult and time-consuming to produce (ask any parent). Mechanicals can be saved, copied, modified, and replicated any number of times at relatively low cost.
- Biologicals have a continuity of consciousness, memory, and a concern about their individual future welfare. Mechanicals do not exhibit a continuity of consciousness or sense of individuality, at least as far as we know.
- Biologicals have a preference which control, or at least inform, their behavior. Preferences change over time and differ between people, but seem to follow certain empirical regularities on average. Mechanical behavior depends on their programing, parametrization, and the data they have been trained on. These elements can be altered at any time, in any way, and so are not likely to follow any empirical standard.

Making beneficial exchange possible in virtual environments requires mechanisms that allow to agents to extend trust one another. Thus, the problem of establishing a provable identity is common to biological and mechanical agents. Even with provable

[1] We use "honesty" as a shorthand for conforming to social expectations in interactions hereafter.

identities, mechanisms require that agents have access to credible, verifiable, and complete histories of the counterparty agents with whom they wish to transact. See Glikson and Woolley[10], Oksanen, et al. [14], and Lockey, et al. [13], for recent discussions of empirical and experimental work regarding human trust in machines.

To explore this question, we consider a sequential trust game between a biological and a mechanical agent. We show that if the game is played only once, cooperation is not possible, and gains from trade are lost. On the other hand, if the game is repeated infinitely, then cooperation and the associated gains from trade can be achieved.

Cooperation is only possible in the two-player repeated game because each agent is unambiguously identified in the sense that each agent deals with the same counterparty each period. This allows each agent to correctly attribute the history of play to an individual actor. Note that the agent's type, and its real-world identity, if it has one, are irrelevant. All that is required is knowing what the counterparty one is facing in the current period has done in the past.

If there are multiple anonymous agents that are randomly matched each period, on the other hand, this cooperative result breaks down. In effect, the counterparty each agent faces in any given period has no identity, and so no history. Punishment in the future for bad behavior in the past is therefore impossible. As a result, cooperation is also impossible.

The main contribution of this paper is to offer a solution to this problem. We describe a blockchain architecture that uses NFTs to create PPK identities, and signed attestation transactions that create provable histories. NFTs and attestations are created as native record and transaction types without the use of smart contracts. We also describe a blockchain-based messaging mechanism for large, two-sided markets with random matching.

Using this as a foundation, Biological and Mechanical agents can interact, transact, and engage in exchange in peer-to-peer markets without the need for trust between agents, or their sponsors or creators.

Our approach obviates the need to engage the question of individuality for machine intelligences, sentient or otherwise. Identity is a private key, and the nature of the agent that owns the key it is unimportant. The preferences of mechanicals, how they might be formed, and even their existence, are also unimportant. What matters is behavior. Mechanicals that don't behave honestly are ignored by Biologicals, and in this sense, are selected against in an evolutionarily dynamic.

2 The Model

We consider a trust game with two types of anonymous agents: Biological Humans and Machine Intelligences, which we call **Biologicals** and **Mechanicals**.

$$\text{Biologicals}: \quad b \in \{1, \ldots B\} \equiv \mathcal{B}$$
$$\text{Mechanicals}: \quad m \in \{1, \ldots M\} \equiv \mathcal{M}.$$

Mechanicals have a comparative advantage at executing certain types of tasks, booking airline reservations, filing taxes, or optimizing investment portfolios, for example. See, for example, Acemoglu and Restrepo [1], Trammell and Korinek [17],

and Zarifhonarvar [19]. We call each of these tasks a **Process**, which from a formal standpoint, is a mapping from inputs to outputs:

$$\textbf{Proc}: \text{INPUT} \Rightarrow \text{OUTPUT}$$

where

$$\text{Proc}_p \in \{\text{Proc}_1, \hat{u} \ \text{Proc}_P\} \equiv \text{PROC}$$
$$\text{input}_i \in \{\text{input}_1, \hat{u} \ \text{input}_I\} \equiv \text{INPUT}$$
$$\text{output}_o \in \{\text{output}_1, \hat{u} \ \text{output}_O\} \equiv \text{OUTPUT}$$

and

$$p \in \{1, \hat{u} \ P\} \equiv \mathcal{P}, i \in \{1, \hat{u} \ I\} \equiv \mathcal{I}, \in \{1, \hat{u} \ O\} \equiv \mathcal{O}.$$

■

Just as executing processes is difficult for a Biological, verifying that a Mechanical has executed a process correctly is also costly. A **Verification** is a mapping from processes, inputs, and outputs, to a truth value.

$$\textbf{Verify}: \text{PROC} \times \text{INPUT} \times \text{OUTPUT} \Rightarrow \{\text{CORRECT}, \text{MALICIOUS}\}$$

such that

$$\forall \ p \in \mathcal{P}, \ i \in \mathcal{I}, \text{ and } o \in \mathcal{O}$$
$$\text{Verify}(\text{Proc}_p, \text{input}_i, \text{output}_o) = \text{CORRECT} \ _c \quad \text{Proc}_p(\text{input}_i) = \text{output}_o$$
$$\text{Verify}(\text{Proc}_p, \text{input}_i, \text{output}_o) = \text{MALICIOUS} \ _c \quad \text{Proc}_p(\text{input}_i) \neq \text{output}_o$$

■

Audits are conducted by external agents called **Verifiers**, which are not explicitly modeled in the current paper, and who are assumed to be honest. Verifiers are paid in advance for a probabilistic audit that depends on a public randomization device.

For example, if an audit costs $10, a Biological would send a Verifier $1 in exchange for an audit executed with a 10% probability. We discuss the meaning of audit, verification, and provability, in more detail in Section 5.

Let $CP \in (0, \overline{CP}]$ denote the **Cost of Executing a Process** correctly to a Mechanical:

$$\textbf{CostProc}: \text{PROC} \Rightarrow (0, \overline{CP}].$$

Let $CV \in (0, \overline{CV}]$ denote the **Cost of Verifying an Execution of a Process** to a Verifier:

$$\textbf{CostVerify}: \text{PROC} \Rightarrow (0, \overline{CV}].$$

Biologicals and Mechanicals play a sequential **Trust Game** in which Biologicals move first and choose either to make an **Offer** or **PASS**. An offer consists of a **Fee** paid in advance to Mechanicals to compensate them for executing a process:

$$\text{Fee} \in [0, \overline{F}],$$

and **P**, an **Audit Probability**:

$$P \in [0, 1],$$

which is a binding commitment if the offer is accepted. If a Biological decides to PASS, he does not send the Mechanical any fees or inputs.

The Mechanical moves second after seeing the Biological's action. If the Biological makes an offer, the Mechanical decides whether to accept or reject it. If he accepts, the Biological sends the offered fee and his input to the Mechanical, and $(P \times CV)$ to a Verifier. The Mechanical then chooses **CORRECT** or **MALICIOUS**

execution and sends an output to the Biological. Alternatively, the Mechanical can decline the offer and choose **NULL** execution. In this case, the game is over, and no fees, inputs, or outputs are exchanged. If the Biological chooses to PASS, then NULL execution is the only action available to the Mechanical.

Formally, the **Action Space** is defined as follows:

$$a^b \in \{(\text{Fee}, p) \in [0, \overline{F}] \times [0, 1], \text{PASS}\} \equiv \mathcal{A}^b$$
$$a^m \in \{\text{CORRECT}, \text{MALICIOUS}, \text{NULL}\} \equiv \mathcal{A}^m.$$

■

We assume that Biologicals cannot determine if a process was executed correctly unless they explicitly verify it. Further, we assume that Biologicals are unable to attribute any increase or decrease in their utility to how a Mechanical chooses to execute a given process. Biologicals do know that correctly executed processes increase their welfare, but are unable to separate this contribution from the many other, difficult to understand, events that affect them positively and negatively.

The one-period **Utility Function of Biologicals** if an offer is accepted depends on how it is executed:

$$\textbf{Utility}^{\mathbf{b}}: \text{PROC} \times \text{INPUT} \times \text{OUTPUT} \Rightarrow [0, \overline{U}]$$

where if

$$\text{Verify}(\text{Proc}_p, \text{input}_i, \text{output}_o) = \text{MALICIOUS},$$

then

$$\text{Utility}^b(\text{Proc}_p, \text{input}_i, \text{output}_o) = 0.$$

■

While Mechanicals do not have utility functions in the same sense as Biologicals, we will assume that they maximize a payoff function that depends on fees collected and how processes were executed. This might be explained by the existence of an unmodeled Biological agent who instantiates a given Mechanical, programs its behavior, and receives any net value generated by his creation. It might also reflect the need of an autonomous Mechanical for resources to exist or replicate.

MALICIOUS execution gives Mechanicals a higher payoff, all else equal. This may be due to Mechanicals using inputs in a way that benefits them directly, or choosing not to go to the expense of executing any process and returning a fictitious output instead. Let $MV \in (0, \overline{MV}]$ denote the **Net Value of Malicious Execution** to a Mechanical:

$$\textbf{MaliciousValue}: \text{INPUT} \Rightarrow (0, \overline{MV}].$$

Given some $(\text{Proc}_p, \text{input}_i) \in \text{PROC} \times \text{INPUT}$, the **Payoff Functions** for agents are defined as follows:

$$F: \mathcal{A}^b \times \mathcal{A}^m \Rightarrow \mathbb{R}^2 \equiv (F^b(a^b, a^m), F^m(a^b, a^m))$$

where $\forall\ (\text{Fee}, P) \in [0, \overline{F}] \times [0, 1]$,

$$F^b((\text{Fee}, P), \text{CORRECT}) =$$
$$\text{Utility}^b(\text{Proc}_p, \text{input}_i, \text{Proc}_p(\text{input}_i)) - \text{Fee} - P \times \text{CostVerify}(\text{Proc}_p)$$
$$F^b((\text{Fee}, P), \text{MALICIOUS}) = -\text{Fee} - P \times \text{CostVerify}(\text{Proc}_p) - \varepsilon$$

$$F^b((\text{Fee}, p), \text{NULL}) = 0$$
$$F^b(\text{PASS}, \text{NULL}) = 0$$

and

$$F^m((\text{Fee}, P), \text{CORRECT}) = \text{Fee} - \text{CostProc}(\text{Proc}_p)$$
$$F^m((\text{Fee}, P), \text{MALICIOUS}) = \text{Fee} + \text{MaliciousValue}(\text{input}_i)$$
$$F^m((\text{Fee}, P), \text{NULL}) = 0$$
$$F^m(\text{PASS}, \text{NULL}) = 0$$

■

Note that we subtract ε from the payoff to a Biological when it makes an offer which is accepted, but where the Mechanical chooses MALICIOUS execution. This reflects the small cost of transmitting the input to the Mechanical. Since fees and audit probabilities are not bounded away from zero, this cost serves to make Biologicals prefer to PASS rather than send a trivial offer, $(\text{Fee}, P) = (0, 0)$, to Mechanicals if they know it will result in MALICIOUS execution.

3 The Two-Player One-Shot Game

A **Strategy for a Biological** is a choice from his action space, while A **Strategy for a Mechanical** is any mapping from the Biological's action space to CORRECT, MALICIOUS, or NULL execution such that PASS always maps to NULL execution:

$$s^b \in \mathcal{A}^b \equiv \mathcal{S}^b,$$

and

$$s^m \equiv \mathcal{S}^m,$$

such that

$$s^m : \mathcal{A}^b \Rightarrow \mathcal{A}^m, \text{ and } \forall\ s^m \in \mathcal{S}^m,\ \ s^m(\text{PASS}) = \text{NULL}.$$

■

A **Strategy Profile** is denoted:

$$S \equiv (s^b, s^m) \in \mathcal{S}^b \times \mathcal{S}^m \equiv \mathcal{S},$$

where $\mathcal{S}^b$ and $\mathcal{S}^m$ denote the **Strategy Spaces** for Biologicals and Mechanicals, respectively.

Given some $(\text{Proc}_p, \text{input}_i) \in \text{PROC} \times \text{INPUT}$, a strategy profile,

$$S \equiv (s^b, s^m) \in \mathcal{S}$$

is a **Subgame Perfect Equilibrium** (**SPE**) if:

$$\forall\ \bar{s}^b \in \mathcal{S}^b,\ \ F^b(s^b, s^m(s^b)) \geq F^b(\bar{s}^b, s^m(\bar{s}^b))$$

and

$$\forall\ \bar{s}^b \in \mathcal{S}^b,\ \forall\ \bar{s}^m \in \mathcal{S}^m,\ \ F^m(\bar{s}^b, s^m(\bar{s}^b)) \geq F^b(\bar{s}^b, \bar{s}^m(\bar{s}^b)).$$

■

Note that the Mechanical's strategy must be payoff maximizing for any action the Biological chooses, that is, for every subgame.

Theorem 1 says that the only SPE equilibrium in the one-shot game is for Biological to choose PASS rather than making an offer to the Mechanical to execute a

process. This results in a loss of potential gains from trade due to the non-contractibility of CORRECT process execution.

Theorem 1. *Given some* $(\text{Proc}_p, \text{input}_i) \in \text{PROC} \times \text{INPUT}$, $S = (s^b, s^m) \in \mathcal{S}$ *is an SPE of the one-shot game if and only if:*

$$s^b = \text{PASS}$$
$$s^m(\text{PASS}) = \text{NULL}$$
$$s^m(\text{Fee}, P) = \text{MALICIOUS}, \forall\ (\text{Fee}, P) \in [0, \overline{F}] \times [0, 1].$$

Proof:

Suppose that

$$s^b = \text{PASS}.$$

Then the Mechanical is constrained to choose

$$s^m(\text{PASS}) = \text{NULL},$$

which is therefore (trivially) a best-response. Suppose instead that:

$$s^b \neq \bar{s}^b = (\overline{\text{Fee}}, \overline{P}) \in [0, \overline{F}] \times [0, 1].$$

Then

$$F^m((\overline{\text{Fee}}, \overline{P}), \text{MALICIOUS}) = \overline{\text{Fee}} + \text{MaliciousValue}(\text{input}_i) >$$
$$F^m((\overline{\text{Fee}}, \overline{P}), \text{CORRECT}) = \overline{\text{Fee}} - \text{CostProc}(\text{Proc}_p),$$

and

$$F^m((\overline{\text{Fee}}, \overline{P}), \text{MALICIOUS}) = \overline{\text{Fee}} + \text{MaliciousValue}(\text{input}_i) >$$
$$F^m((\overline{\text{Fee}}, \overline{P}), \text{NULL}) = 0,$$

and so the Mechanical will always choose

$$s^m((\overline{\text{Fee}}, \overline{P})) = \text{MALICIOUS}$$

in the subgames where $\bar{s}^b = (\overline{\text{Fee}}, \overline{P})$. Since

$$F^b(\text{PASS}, \text{MALICIOUS}) = 0 >$$
$$F^b((\overline{\text{Fee}}, \overline{P})), \text{MALICIOUS}) = -\overline{\text{Fee}} - \overline{p} \times \text{CostVerify}(\text{Proc}_p) - \varepsilon$$

The Biological will therefore always prefer the subgame where he chooses:

$$s^b = \text{PASS}.$$

■

Note that there is a large and growing literature on machines as participants in games related to financial markets, (Bebeshko, et al.[4]) oligopoly pricing (Calvano et al. [6]), auctions (Bichler et al.[5]) learning (Zeng, et al.[20]), many other environments.

4 Generalized Games

This section discusses three generalizations of the one-short game trust game defined above. We will not, however, develop formal descriptions due to the high notational and analytical overhead involved, and space limitations imposed by the publsher. Instead, we will outline the structure of the models and how the results change from the

one-shot game. We refer readers to Conley [8] for full development of these generalizations.

4.1 The Two-Player Repeated Game

Suppose first that one Biological and one Mechanical play the sequential game an infinite number of times in succession[2]. Note that in the two-player game, agents automatically know one another's identities in the sense that each has a unique counterparty. They also can remember the complete history of play of their counterpart in the game.

One might expect that a kind folk theorem should obtain in the sense that as the discount factor between periods goes to one, any individually rational allocation could be supported as a Nash equilibrium. Given the sequential nature of the game, however, it will turn out that a narrower set of allocations can be supported as **Consistent Subgame Perfect Equilibrium** (**CSPE**).

CSPE is a refinement of SPE requires that beliefs are consistent in the following senses:

- Agents believe that their counterparties will behave identically in essentially identical situations in all future periods[3].
- In every period t, agents base their beliefs about the future strategies of their counterparties on the strategy they played in the most recent period. Note that for Mechanicals, this is the current period, while for Biologicals, it is the previous period.

Given these beliefs, CSPE strategies are payoff maximizing in both the supergame, and in every subgame.

Claim 1. *In a two-player, repeated trust game, there exists a CSPE in which the Biological makes an offer each period which is accepted by the Mechanical who then chooses CORRECT execution.*

In particular, grim trigger-type strategies support a CSPE in which agents choose to cooperate in each period where:

- $\text{Fee} \geq \text{CP}$. That is, fee covers the cost of processing.
- $\text{CP}\uparrow$, or $\text{MV}\uparrow$, implies either $\text{Fee}\uparrow$, or $\text{P}\uparrow$. That is, if either the cost of processing, or the value of MALICIOUS execution goes up, then the Biological must either raise the fee offered, or increase the probability of an audit to compensate.
- $r \to 1$ implies $(\text{Fee} - \text{CP}) \to 0$. That is, as agents discount the future less heavily, even small surpluses of fees over processing costs result is high ex-

---

[2] We assume both Biologicals and Mechanicals discount the future at some rate $\rho \in (0, 1)$, and denote the one-period **Discount Factor** as: $r = (1 - \rho) \in (0, 1)$.

[3] Situations in two distinct periods are "essentially identical" if the histories are either both cooperative or both non-cooperative, and in the case of the Mechanical, the Biological takes the same action at the beginning of the period. A history of play is called **Cooperative** if in every period, the Biological makes an offer, the Mechanical accepts and either chooses CORRECT execution, or is not audited, and is called **Noncooperative** otherwise.

pected payoffs for the Mechanical. On the other hand, $(1 - r + rP) \rightarrow P$. Thus, for fixed, but small probabilities of audit, the relative value of MALICIOUS execution ends up being smaller than the expected value of choosing the CORRECT in each period.

Note that the discount rate between periods depends on the length of the period. If a game is played daily, or several times a day, the discount rate gets closer and closer to $r = 1$. There are two implications in this event.

First, the fees offered by the Biological can approach the cost of processing, leaving the Biological with the lion's share of the surplus.

Second, the probability of auditing can approach zero. This is particularity desirable since audits use, rather than transfer, resources. Thus, the market for services between Biologicals and Mechanicals becomes more efficient as interactions become more frequent.

4.2 The Anonymous Multiplayer Repeated Game

Suppose instead that there are an equal number Biologicals and Mechanicals, each of whom is randomly matched to an anonymous counterparty agent each period, and then plays the one-shot trust game. Since agents are anonymous, the history of play does not describe interactions with any specific individual counterparty agent. Rewards and punishments for good and bad behavior based on history, therefore, cannot be correctly targeted.

If a Biological ever makes an offer to a Mechanical to execute a process, it is a dominant strategy for Mechanical to choose MALICIOUS execution. In effect, each period is just like a new one-shot game with a counterparty that has not been provably encountered before. The next Biological that the Mechanical encounters at best will condition his strategy on the behavior of the previous Mechanicals he has encountered, not on the unknown behavior of the current one. Given this, it is a best-response for the Biological to choose PASS each period.

This leads to the following Claim:

Claim 2. *In an anonymous, multiplayer, repeated trust game, playing the one-shot SPE strategies each period is the only CSPE.*

Claim 2 implies that anonymous markets between Biologicals and Mechanicals are likely to fail profoundly. When agents can neither prove how they behaved in previous periods, nor condition future play against one another (should it ever occur) on the outcome of their last encounter, trust cannot be supported by mechanisms.

Biologicals and Mechanicals would both gain from trade. Humans benefit for process execution, and artificial intelligence agents could provide such services in exchange for fees that would leave both parties better off. The information failure in identity and history, however, prevents it.

What this suggests is that trust deficits between Biologicals and Mechanicals may limit the positive impact, not to mention, the market penetration, of coming AI technologies.

4.3 The Nonanonymous Multiplayer Repeated Game

Two-sided markets are often mediated through trusted platforms. For example, see Zhou [22] and Tan, et al. [16] among many others. In contrast, we consider decentralized two-sided markets with random matching.

Suppose we modified the anonymous multiplayer repeated game described above as follows:

- Both types of agents could prove their identity to one another. That is, while agents could choose to remain anonymous, they could also choose to provide proof of their identities when interacting with other agents.
- There was a way to make public and provable the outcome of any one-period game between two agents who choose to identify themselves.
- The history of interactions was provabley complete and uncensorable.
- Agents could check on the history of all agents with whom they are matched before deciding on strategies.

Claim 3. *In a nonanonymous, multiplayer, repeated trust game with provable and complete histories, there exists a CSPE in which Biologicals make offers each period which are accepted by Mechanicals who then choose CORRECT execution.*

To see why Claim 3 is true, suppose that Biological follow grim trigger strategies based on the history in all of it previous interactions of the Mechanical with which they are currently matched That is, Biologicals never make offers to Mechanicals that have ever declined an offer, or been caught through an audit of choosing MALICIOUS execution, in any period, with any Biological.

Note first that in period $t = 0$, the no agent has a history. If the costs and other parameters of the game allow the Biological to make an offer as defined the Mechanical's grim trigger strategy, he does so. The Mechanical, following its own grim trigger strategy, then accepts and chooses CORRECT execution. This results in a cooperative outcome for the period, and so both agents can claim (and we assume, prove) that they have a history of cooperative behavior. In the next period, Biologicals with a cooperative history make offers if they are matched with honest Mechanicals, who in turn accept and choose CORRECT execution. The same pattern is repeated in every subsequent period.

On the other hand, a Biological encountering a Mechanical who has a history of noncooperation ("dishonesty") in any previous period would not choose to make an offer given his trigger strategy. Suppose instead that he did make an offer. The Biological assumes that all other agents, including other Biologicals, will follow the equilibrium trigger strategies. This implies that Biologicals who are matched with this dishonest Mechanical in any future periods will choose PASS. As a result, the value of the continuation game for the dishonest Mechanical is zero whatever it chooses in the current period. Thus, the Mechanical will always choose MALICIOUS execution if the current Biological makes an offer. As a result, the current Biological is better-off and following their trigger strategy and choosing PASS.

5 History and Identity

The message of the previous sections is that while large, anonymous, decentralized, two-sided markets will generally fail, they can be made to work if agents can deanonymize and establish credible personal histories.

As above, we assume that independent Verifiers exist who give honest assessments of whether processes were correctly or maliciously executed in exchange for fees. Adding a mechanism to assure this is possible, but not covered in this paper.

The idea of auditing, however, embeds the requirement that there is an objective, verifiable standard of correctness. For example, in the case of blockchains with deterministic protocols, it should be the case that, given the current ledger state, a proposed block is either valid or invalid. It may also be that given a set of financial inputs, a tax return is, or is not, correct, or is, or is not, optimized to a certain standard, or that an investment portfolio was, or was not, managed under some specific accepted standard of best-practice.

Without this kind of verifiability, however, markets are likely to fail. If Biologicals cannot tell if they are being treated honestly, why would a Mechanical spend the resources to do so? If bots or malicious humans can leave what amount to fake Yelp reviews and have them taken as true histories, then dishonest Mechanicals can falsely pump their reputations while smearing honest ones. If truth is not provable, then it may as well not exist from a mechanism design standpoint. For example, see Ball and Kattwinkel [3] who explore a mechanism with probabilist verification of truthful binaries and the impact on the distribution of surplus in the context of identity and authorization.

In this section, we will assume that truth is provable using Verifiers and develop an architecture that relies on **Public/Private Key** (**PPK**) **Cryptography** for identity, and **Blockchain** for histories. It is important to note that our proposal uses blockchain purely as a data source. This contrasts with the standard approach of building decentralized markets using smart contracts. For example, See AlAshery et al. [2] for energy markets, Hua, et al. [12], for carbon markets, and Schär, [15] for financial markets built on smart contracts.

5.1 Artificial Identity

The philosophical question of whether an artificial intelligence, or other Mechanical, has an identity, much less an individuality, is a difficult one. AIs are distributed over clusters of computers. New instances can be deployed and taken down at will. Exact copies an AI's code and data can be produced, shipped, and then installed, remotely. AI's also change continuously as they ingest and process new data. Can such an agent, even if identified, be punished, and would it care?

Fortunately, we do not need to engage these weighty questions. Instead, we propose that identity is equivalent to a PPK pair. This is by no means a new idea, and the technology is well-known. In the interest of clarity, let us briefly review.

Public and private key pairs are mathematically entangled, asymmetric encryption

keys. For our purposes, their essential feature is that anything encrypted with one key in a pair can only be decrypted with the complementary key. Public key encryption is what enables HTTPS, blockchain, digital signing of documents, and many other building blocks of modern information technology.

As an identity for agents, it works as follows. A Biological or Mechanical produces a PPK pair and publishes the public key as their identity. The complimentary private key is kept secret, and used to cryptographically sign attestations that signify agreement to, or responsibility for, certain actions. This might include receiving specific data, making a request for processing, claiming that input was processed incorrectly, or challenging such a claim.

The central element in this approach is that a public key can be used to prove that the owner of the corresponding private key is the only one who could have created the signature. Thus, if a set of attestations can be verified by the same public key, then they must have been signed by owner of the same private key, and in that sense, by the same "individual".

5.2 Provable History

As we discuss in the introduction, without identity, there is nothing to attach a history of behavior to. Anonymous agents cannot establish reputations, nor can they be held accountable for their actions. With identity, it becomes possible to create intertemporal mechanisms to incentivize good behavior.

The problem now becomes, how do we establish credible and complete histories of behavior? This may seem especially challenging when there are many Biological and Mechanical agents in market, and so matches may happen many times per second. Artificial intelligences might be able to handle this volume of information, but it seems probable it would be beyond the capacity of humans. The inputs and outputs may also be very large byte strings, and processing, as we mention, could be complex and costly. Finally, how would the Biological know that it had access to all reports of both good and bad behavior?

The solution we propose relies on blockchain. An immediate question is: what blockchain? There are thousands of implementations with different consensus mechanisms, security guarantees, costs, scalability, and so on. Rather that answering this question specifically, we give a list of the requirements a blockchain implementation should satisfy for our purposes.

1. **Data Availability**: All inquiries to block explorers regarding transaction and ledger data in particular must be answered correctly.
2. **Provability**: The data provided by block explorers should allow agents to independently prove the correctness, contents, and inclusion of transactions in committed blocks, as well as the state of the ledger at any block height.
3. **Immutability**: All committed blocks (perhaps after a delay) are considered finalized, and cannot be reorganized or otherwise altered.
4. **No Censorship**: All valid transaction requests sent by Biologicals or Mechanicals must be processed by the network and included in committed blocks without un-

reasonable delay.

5. **Low Cost**: The cost of having a transaction included in a block must be low relative to the payoff and cost values of the economic environment described above.
6. **Scalability**: The blockchain must have the capacity to include transactions at the scale required by the economic environment described above.

We will assume a perfect blockchain in these dimensions: all valid transactions are immediately, and immutably, included in the next block at zero cost, and all agents in the game are aware of the contents of all blocks. Exploring the impact of less than perfect or manipulable blockchains is a task for another paper.

5.3 Attestations and NFTs

We require one type of record, and one type of transaction, to create identities and histories, although there are probably many alternative approaches that would also serve. These are **Non-Fungible Tokens** (**NFT**) and **Attestations**. We will also make use of ordinary coin transactions.

NFTs, as we conceive them, are immutable records that are created in a blockchain's ledger and include two mandatory, and two optional elements.

- A hash or hashes of a document or digital object being tokenized or attested to. (Optional)
- Metadata, which might be encoded indexing information to assist search, plain text descriptions of offers and results, contact and identity information, pointers to external documents, full documents in encrypted or unencrypted form, or anything else that can be expressed as bytes. (Optional)
- A PPK signature on the elements above. (Mandatory)
- The public key that complements the private key that signed the data in the first two elements. (Mandatory)

Attestations, as we conceive them, contain exactly the same four mandatory and optional elements. They are only entered as transactions in a committed block, however (if they satisfy the protocol's definition of correctness[4]), and do not create new records in blockchain's ledger. They also include a **Nonce** that makes it possible to confirm that the history is complete. Block explorers and agents can check that a set of attestation transactions has an unbroken sequence of nonces, which proves that all translations that originated from a given ledger record are accounted for.

In general, attestation transactions and NFT records are not datagram types that are native to blockchains See Hardjono and Smith [11] and Wang, et al. [18], for example. Instead, they are instantiated using smart contracts. This is problematic because these datagrams, and proof of their ownership, contents, and origin, are only implicit in the smart contract's state. Verification requires rerunning every transaction that targeted the smart contract since it was deployed in the correct sequence. This makes sufficient data availability burdensome, and provability costly.

---

[4] Correctness under blockchain protocol requires such things as a correct signature, correct nonces, and sufficient funds to pay for a transaction. It has nothing to do with the correctness or content of an attestation message in the context of the game's messaging rules.

Using smart contracts also significantly increases costs, and limits scalability. As an unhappy bonus, smart contracts have proven to be a significant attack surface for blockchains. See Chaliasos, et al. [7] or Zhang, et al. [21], for example. Fortunately, it is possible to implement attestations and NFTs nativity, visibly, and provably.[5]

5.4 An Architecture for Identity

Identity is implemented through NFTs. Agents of either type simply mint, or have minted, an NFT record with a public key of their choosing signed by the complementary private key, which only they know. It might or might not be useful for the NFT to include metadata that describes the agent type, who its sponsor is, what services it provides, how to contact it, and so on, but very little is needed for our purposes. An **Identity NFT** simply puts into the ledger the provable fact that some agent knows both parts of a PPK pair. The collection of Identity NFTs on a given chain instance can be thought of as kind of local **Public Key Infrastructure** (**PKI**).

The existence of the Identity NFT record allows other agents to connect any attestations signed with the associated private key to a specific public key as an identity, and thereby allows the creation of an attributable history.

5.5 An Architecture for History

History is recorded through attestations. There are, no doubt, many ways to do this, and different approaches may be more suitable for different applications. In this Subsection we give a sketch of simple set of game messaging rules that correspond to the multiagent game outlined in Section 4. This relies on two main elements. The first is the Identity NFTs described above. The second are various types of **Attestation Transactions** that work as messages when committed to a blockchain. Appendix A describes a set of cryptographic and blockchain primitives that support the architecture used in this subsection.

Below, we will call the AI Mechanical agent Alice, the human Biological agent Bob, and the Verifier agent Victor. Attestation transactions are essentially metadata packages that are signed with an agent's private key and then committed to a block in a blockchain. They do not create or modify ledger records except to deduct fees from, and increment the nonce of, the sending coin account. We will refer to them as **Messages**, below.

---

[5] Full disclosure: The author is the Chief Economist of the Geeq Project, a layer zero blockchain protocol that in fact does instantiate attestations as transactions signed by coin account owners and places them directly in blocks. Geeq's blockchain incorporates NFT mint accounts as ledger records that can create the type of signed NFT ledger records as described in this section as well. Geeq's protocol also satisfies, or approximately satisfies, the six requirements outlined in Section 5.2.

5.6 Game Messaging Rules

The Pregame: All agents, of all types, generate a PPK pair and then create and commit an Identity NFT to the blockchain ledger that includes their public key, and may include other details such as their agent type.

The Game:

1. Bob is matched with a Mechanical, Alice in this case, and uses the block explorer to confirm that she has an Identity NFT and a cooperative history.
2. Bob either commits an Offer Message that includes a process index, $p \in \mathcal{P}$, he wishes executed, an offer, (Fee, P), identifies Alice as his counterparty, and Victor as the Verifier, or decides to ignore the opportunity to work with Alice, in effect, choosing PASS silently.
3. Alice is obliged to scan the chain for any Offer Messages directed to her. When she sees one, she commits either an Accept, or Decline Message using the hash of the Offer Message transaction as an identifier.
4. Victor, if he becomes aware of a Decline Message, commits a Verification Message indicating NULL execution by Alice.
5. Bob waits to see how Alice responds. If she declines, the period is over. If she accepts, he commits three transactions.
   a. A coin transfer transaction sending Fee to Alice.
   b. A coin transfer transaction sending $P \times CV$ to Victor.
   c. An Input Message containing his input and the hashes of the two committed coin transactions above. (Appendix B shows how this can be done without publicity reveling the input, while still allowing Victor to verify what Bob sent to Alice.)
6. Alice waits to see Bob's Input Message, and when she finds it, she confirms that the coin transactions are committed and correct. If so, she chooses either CORRECT or MALICIOUS execution, and then commits an Output Message that includes whatever output she generates (which can also be encrypted, but still verifiable).
7. Victor sees the Output Message. He consults a public randomization device, and if an audit is called for, ingests Bob's input, Alice's output, and then executes $proc_p$ to see if Alice is honest. Victor then commits a Verification Message indicating that execution was CORRECT or MALICIOUS. If no audit is called for, he commits a Verification Message indicating that the type of execution is UNCERTAIN.

Note that the blockchain is used as a kind of billboard in the sense that agents cannot pretend to be unaware of messages directed to them. This is key because otherwise it is impossible to differentiate intentional, strategic, silence or deafness, from true communications failure.

Appendix B describes how Victor also plays a role in making sure that Alice and Bob take each of these steps, and do them correctly. If they don't, he commits a Verification Message indicating which party is dishonest.

Taken together, by the end of the period, Victor will have committed an attestation

that the history of play was either cooperative or non-cooperative, and in the later event, provides proof that the bad outcome was attributable to the actions of either Alice and Bob.

6 Conclusion

We propose a sequential, positive-sum, trust game as a model of a generalized two-sided market. We show that when agents play this game only once, the only subgame perfect equilibrium is the noncooperative outcome. On the other hand, when a pair of agents play the one-shot game an infinite number of times, cooperation becomes a consistent subgame perfect equilibrium.

We then extend the game to include randomly matched anonymous agents. Perhaps unsurprisingly, the positive result breaks down, and once again, only the noncooperative outcome is an equilibrium. If the randomly matched agents can deanonymize and create a complete and credible history of their actions in all previous periods, however, then the cooperative outcome can be recovered as a consistent subgame perfect equilibrium.

Economic mechanisms with human agents are built on a foundation that assumes that each agent has well-defined preferences. Concomitant with this is an assumption that, while agents may be anonymous with respect to one another, each has an identity known at least to themselves. In turn, this rests on an assumption that agents have an individuality, or a sense of continuity between periods, and so care what happens to them as an individual in the future.

It is unclear whether AIs have preferences as we understand them. See Gabriel [9] for some speculations. It is even less clear whether AIs, even sentient ones, have a sense of individuality or continuity of self over time. Given this, is it possible to assign an identity to individual Mechanical agents?

In this paper, we develop a blockchain-based architecture and build a messaging mechanism that allows virtual agents of any type to achieve Pareto improving cooperative outcome in two-sided markets with random matching. We show that identity can be assigned through public/private keys without the requirement that it be attached to an actual individual. Attestations signed by these private keys can then be used to create a provable history of behavior that can then be connected to an Identity NFT containing the corresponding public key.

Using this as a foundation, Biological and Mechanical agents can interact, transact, and engage in exchange in peer-to-peer markets without the need for trust between agents, or their sponsors or creators. Bad artificial agents will simply be selected out of the market.

To the extent that this type of mechanism, and the architecture behind it, can be refined and generalized, human agents will be able to benefit from the many comparative advantages that artificial agents bring to the table. In turn, companies that make AI applications, and even autonomous artificial agents, will be able to find ready markets for their services.

Acknowledgments: I would like to thank Scott Page for discussions which partially inspired this work, and to participates in the 2024 NSF/CEME Decentralization Conference. My thanks are also due to three anonymous referees for suggested improvement to the manuscript.

Disclosure of Interests. It: The author serves as the Chief Economist for the Geeq Project, a layer one blockchain protocol currently under development, and which also provided inspiration for this work. See footnote 6 and Section 5 for more details. This work, however, is not commissioned by Geeq or any other entity, and reflects only the options of the author, who takes full responsibility.

A Cryptographic and Blockchain Primitives

This appendix defines various cryptographic primitives and the basic datagrams used by the blockchain to generate the provable histories our mechanism requires. It also provides more details about the game's messaging rules.

A.1 Cryptographic Primitives

Generic data of arbitrary size, including inputs, outputs, and elements of blockchain transactions and records, are called **Byte Strings**:

$$\mathbf{BYTE\_STRING} \equiv \{\text{byte\_string} \in \{0,1\}^n | n \in \mathbb{N}\}.$$

A **Hash Function** maps a **Pre-image**, which is a byte string of any length, into an approximately uniform distribution of (usually 32 byte) byte strings called a **Hash Digest**.

$$\begin{aligned} &\mathbf{Hash}: \text{BYTE\_STRING} \Rightarrow \{0,1\}^{32}: \\ &\text{Hash}(\text{pre\_image}) = \text{hash\_digest}. \end{aligned}$$

There are three sets of agents:

$$\begin{aligned} \text{Biologicals}: &\quad b \in \{1,\ldots B\} \equiv \mathcal{B} \\ \text{Mechanicals}: &\quad m \in \{1,\ldots M\} \equiv \mathcal{M} \\ \text{Verifiers}: &\quad v \in \{1,\ldots V\} \equiv \mathcal{V}. \end{aligned}$$

Each agent, of each type, creates a **Public/Private Key Pair**:

$$\mathbf{(pub\_key^x, pri\_key^x)}$$

where

$$\begin{aligned} &(\text{pub\_key}^b, \text{pri\_key}^b) \\ &(\text{pub\_key}^m, \text{pri\_key}^m) \\ &(\text{pub\_key}^v, \text{pri\_key}^v) \end{aligned}$$

are PPK pairs for generic Biologicals, Mechanicals, and Verifiers, respectively. As we mention above, anything encrypted with one of the paired keys can only be decrypted with the complementary key. Asymmetric encryption is limited in that the bytes string being encrypted must be smaller than the key size, and the process is relatively computationally intensive.

We will also use **Symmetric Encryption Keys**:

$$\mathbf{sym\_key}$$

that have the property that byte strings of any length can be encrypted and decrypted with the same key at relatively low computational cost.

An **Encryption Algorithm** (systematic or asymmetric) maps **Plaintext** byte strings into **Ciphertext** byte strings using a key:

$$\textbf{Encrypt}(\text{key}, \text{plaintext}) = \text{ciphertext}.$$

A **Decryption Algorithm** maps ciphertext byte strings into plaintext byte strings using a key:

$$\textbf{Decrypt}(\text{key}, \text{ciphertext}) = \text{plaintext}.$$

A **Signature Algorithm** maps a private key and a byte string into a byte string called a **Signature**. In general, the byte string being signed is the hash digest of a byte string of arbitrary length.

$$\textbf{Signature}(\text{pri\_key}, \text{byte\_string}) = \text{signature}.$$

Finally, a **Signature Check Algorithm** maps a public key and a signature into a truth value:

$$\textbf{SigCheck}(\text{pub\_key}, \text{signature}) \Rightarrow \{\text{TRUE}, \text{FALSE}\},$$

and takes a value of TRUE if and only an agent who had access to pri_key created signature, using byte_string as the argument.

A.2 Identity NFTs

Identity NFTs are created by **Mint Accounts**, and are signed by their creator. The three data items (in green) are helpful in the sense that a human looking at such a record would know that a certain public key is associated with a specific agent (Alice, Bob, …) of a specific type (one of the three described above). Only **Role** is strictly required because it dictates the rules that allow other agents to determine what sorts of attestations to look for and how to interpret them as a history. The only other truly relevant **ID Data** is the agent's public key which must be part of the record for signature checking in any event.

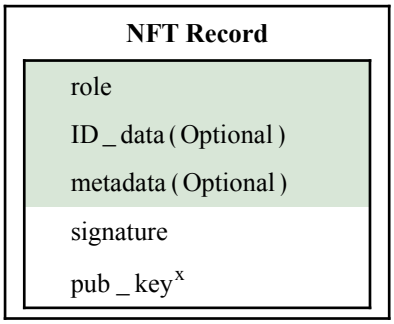


Fig 1. Identity NFT Datagram

The green elements are concatenated, hashed, and signed.[6]

[6] Note that " | " indicates that the byte strings in the argument are **concatenated**.

$$\text{Hash}(\text{role}|\text{ID\_data}|\text{metadata}) \equiv \text{hash\_digest}$$
$$\text{Signature}(\text{pri\_key}^{x}, \text{hash\_digest}) = \text{signature}.$$

It will not matter if an individual Mechanical (whatever that might mean) creates multiple identities. If it does, it is effectively setting-up subsidiaries and "doing business as" several public keys. Since public keys are evaluated on the basis of their own histories, this is no different from separate Mechanicals setting up to do business separately under these public keys. The incentives are the same.

It also will not matter if a Mechanical hands over its private key to another Mechanical. The incentives for the new owner are the same as for the old owner. Behaving honestly has the same expected value no matter who owns the key, and giving away a key is just like replacing the management of a business.

What will matter is if a key-holder knows, or believes, that there is a probability, that it will leave the game, or that the game will end. If there is a known final period, then cooperation unravels in the usual way. If the personal or general final period is probabilistic, then periodic payoffs to the Mechanicals must go up commensurately to account for the lessened expected value of the future. A similar dynamic occurs if the overall market size changes over time. If it is expected to grow, then the value of the future is higher, all else equal, and if it is expected to shrink, it is lower.

Creating multiple Identity NFTs with the same public key should be considered *per se* dishonest, and is easily detectable.

A.3 Messaging using Attestation Transactions

Attestation transactions are created and signed by coin account holders on the blockchain. Unlike NFTs, they do not create records. A valid attestation transaction is simply added to current block. The only record it modifies is the sending coin record, which has the required transaction fee deducted, and its nonce incremented.

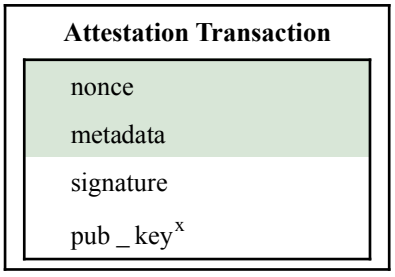


Fig. 2. Attestation Transaction Datagram

For our purposes here, the Identity NFT creation, and all associated attestation transactions, must originate from the same coin account controlled by the private key, pri_key^{x} that signs them all.

A.4 Example of Message Metadata Content

The metadata elements in the attestation transaction are actually messages of different types that mediate the market and generate provable histories. For example, an Offer Message might include the following metadata (full details are provided in Appendix B):

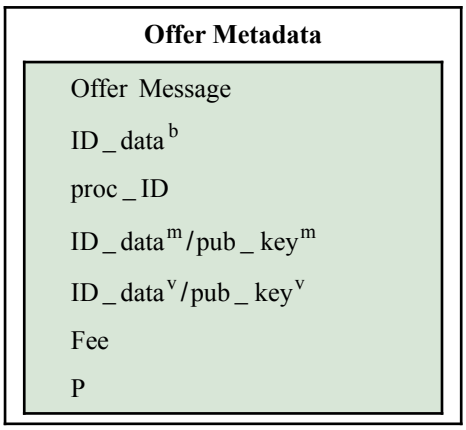


Fig. 3. Offer Message Metadata

A.5 Summary of Message Flow

Table 1 shows the order of messages along all the possible paths, which depend on the actions taken by the three agents. The subscripts indicate the block height at which a message was committed. The cells shaded green show paths and outcomes in which all agents sent and responded to messages within the game's messaging rules. The cells shaded in red show paths and outcomes where one of the agents did not send messages as required the game's rules, and which result in a verifier message assigning responsibility.

Table 1. Message Flow Table

<table>
<tr><td colspan="7">OFF^{b}_{N0}</td></tr>
<tr><td colspan="5">ACC^{m}_{N1}</td><td>DCL^{m}_{N1}</td><td>NR^{b}_{N1}</td></tr>
<tr><td colspan="4">INP^{b}_{N2}</td><td>NR^{m}_{N2}</td><td rowspan="2">VM^{v}_{N2} = NUL</td><td rowspan="2">VM^{v}_{N2} = DB/DM</td></tr>
<tr><td colspan="2">OUT^{m}_{N3}</td><td>FI^{m}_{N3}</td><td>NR^{b}_{N3}</td><td rowspan="2">VM^{m}_{N3} = DB/DM</td></tr>
<tr><td colspan="2">(1 − P)</td><td>P</td><td rowspan="3">VM^{m}_{N4} = DB/DM</td><td rowspan="3">VM^{m}_{N4} = DB/DM</td><td></td><td></td></tr>
<tr><td>AUD^{b}_{N4}</td><td rowspan="2">VM^{v}_{N4} = DB</td><td rowspan="2">VM^{m}_{N4} = UNC</td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td></tr>
</table>

VM^{v}_{N5} = COR/ MAL							

Table 2 provides some detail and context for Table 1. The main new element is the Creation Time Limits column. Once a Biological commits an Offer Message to a block at height N0, other agents must respond withing certain time intervals.

The Mechanical is required to commit an Accept or Decline Message before a limit of **L** additional blocks have been committed to the chain (that is, before some block $N1 < NO + L$). In the event that an Accept Message in committed at block N1, the Biological is required to commit an Input Message at some block $N2 < N1 + L$. In all cases where a response is needed from a specific agent, the game's messaging rules require that it be committed before the block limit expires or else the agent is deemed to be non-responsive, and therefore dishonest.

On the other hand, No Response Message claims by Biologicals and Mechanicals cannot be committed before the block limited expires ($N2 > N1 + L$, for example), and need not be committed at all. If a No Response Message is committed, then the Verifier is required to commit a Verification Message within the normal block limit ($N3 < N2 + L$, for example). In the case where an audit is called for, but the Biological fails to commit an Audit Message containing the key, both limits apply. That is, the Verifier must wait until the block limit for committing the Audit Message has expired, but then must commit its Verification Message within its own block limit, $N4 \in (N3+L,\ N3+2L)$.

Table 2. Legend and Details for Message Flow Table

Symbol	Message Type	Creation Time Limits	Key Content
OFF^{b}_{N0}	Offer Message	N0 = Initial time	$proc\_ID, m, v, (Fee, P)$
ACC^{m}_{N1}	Accept Message	N1 < N0+L	Accepted offer, send input
DCL^{m}_{N1}	Decline Message	N1 < N0+L	Declined offer, NULL execution
NR^{b}_{N1}	No Response Message	N1 > N0+L	No ACC^{m} or no DLC^{m} received
INP^{b}_{N2}	Input Message	N2 < N1+L	$sym\_key,\ input_i$
NR^{m}_{N2}	No Response Message	N2 < N1+L	No INP^{m} received
VM^{v}_{N2}	Verifier Message	N2 < N1+L	NUL event

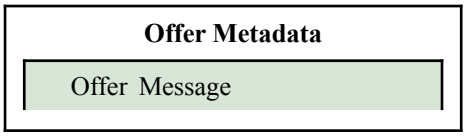
VM^{v}_{N2}	Verifier Message	N2 < N1+L	Dishonest Bio or Mech (No ACC^{m} or DLC^{m})
OUT^{m}_{N3}	Output Message	N3 < N2+L	$output_{o}$
FI^{m}_{N2}	Flawed Input Message	N2 < N1+L	Flawed Input Message
NR^{b}_{N3}	No Response Message	N3 > N2+L	No OUT^{m} received
VM^{m}_{N3}	Verifier Message	N3 < N2+L	Dishonest Bio or Mech (No INP^{b} received)
AUD^{b}_{N4}	Audit Message	N4 < N3+L	sym _ key
VM^{m}_{N4}	Verifier Message	N4 < N4+L	Dishonest Bio (No AUD^{b} received)
VM^{m}_{N4}	Verifier Message	N4 < N3+L	UNC event
VM^{v}_{N4}	Verifier Message	N4 ∈ (N3+L, N3+2L)	Dishonest Bio
VM^{m}_{N4}	Verifier Message	N4 < N3+L	Dishonest Bio or Mech (Flawed Input Message)
VM^{m}_{N4}	Verifier Message	N4 < N3+L	Dishonest Bio or Mech (No OUT^{m} received)
VM^{v}_{N5}	Verifier Message	N5 < N4+L	COR or MAL event

B Details of Messaging Metadata

Appendix B provides details of the metadata that needs to be included attestation transactions required for the game's messaging mechanism.

B.1 Mechanical Message Metadata Content

A Biological $b \in \mathcal{B}$, begins by matching with a Mechanical, $m \in \mathcal{M}$, then choosing a Verifier, $v \in \mathcal{V}$, a process identifier, $p \in \mathcal{P}$, and an offer (Fee, P), and finally, creating and committing to the blockchain an **Offer Message** attestation transaction with the following metadata:



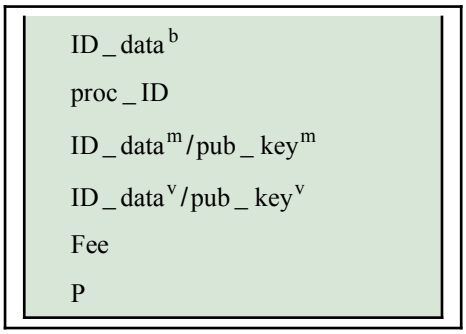


Fig. 3. Offer Message Metadata

where:

- Offer Message: A plaintext message type label.
- ID_data^b : The ID number chosen by the Biological when creating its Identity NFT. This is not strictly necessary since the transaction includes the Biological's public key, which unambiguously identifies the message's originator.
- $proc\_ID: p \in \mathcal{P}$, the process the Biological wishes to have executed.
- ID_data^m / pub_key^m : The ID number and/or public key of the Mechanical the Biological has chosen. At least one is needed, but the public key makes look-ups easier.
- ID_data^v / pub_key^v : The ID number and/or public key of the Verifier the Biological has chosen.
- Fee: The fee being offered to the Mechanical.
- P: The probability of audit the Biological will pay for.

Suppose that the Biological commits an Offer Message that gets included in a block at height N. Suppose for the moment that Mechanical sees this message and responds with an Accept Message (see the next Subsection). Then the Biological commits an **Input Message** to the blockchain.

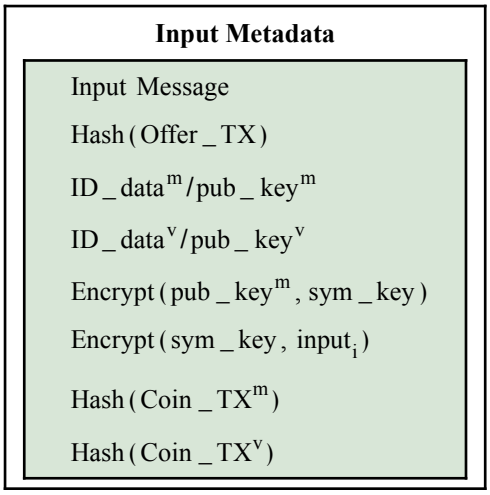


Fig. 4. Input Message Metadata

where:

- Input Message: As above.
- $\text{Hash}(\text{Offer\_TX})$: The hash of the Offer Message attestation transaction that initiates the exchange. This is used as an identification number to make it easy for a block explorer to collect all messages subsequently connected to a given offer.
- $\text{ID\_data}^{m}/\text{pub\_key}^{m}$: As above. Not strictly necessary since it can be looked up using $\text{Hash}(\text{Offer\_TX})$.
- $\text{ID\_data}^{v}/\text{pub\_key}^{v}$: As above, and used by the Verifier to find which messages it should pay attention to.
- $\text{Encrypt}(\text{pub\_key}^{m}, \text{sym\_key})$: The Biological generates a random symmetric key, and encrypts it with the public key of the Mechanical.
- $\text{Encrypt}(\text{sym\_key}, \text{input}_i)$: The Biological uses this symmetric key to encrypt the inputs it wants to have processed. We discuss the reasons for this approach and alternatives in the last Subsection below.
- $\text{Hash}(\text{Coin\_TX}^{m})$: The Biological commits a separate coin transaction sending Fee to the Mechanical and includes the hash of the transaction to allow verification of this fact.
- $\text{Hash}(\text{Coin\_TX}^{v})$: The Biological does the same thing to send $p \times CV$ to the chosen Verifier.

Suppose that the Biological commits an offer or Input Message that gets included in a block at height N. Any Mechanical that maintains an Identity NFT in the ledger is obliged monitor the blockchain for messages. It does not respond within some set number of blocks, it is considered non-responsive, which is the same as noncooperative[7]. In this event, the Biological commits a **No Response Message** to the blockchain.

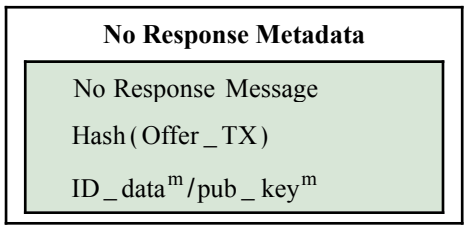


Fig. 5. No Response Message Metadata

where

- No Response Message: As above.
- $\text{Hash}(\text{Offer\_TX})$: As above.
- $\text{ID\_data}^{m}/\text{pub\_key}^{m}$: As above.

This message should be seen by the Verifier who will commit a Verification Mes-

[7] There is, in fact, a mechanism that allows agents to declare that they are off-line, and then come back on-line at later block height without removing their Identity NFT, and with it, the history they have established. We omit these details for now.

sage, outlined below.

Finally, suppose that all goes well, the Mechanical commits an Output Message, and the public randomization device[8] indicates that an audit is called for. Then the Biological commits an **Audit Message** to the blockchain.

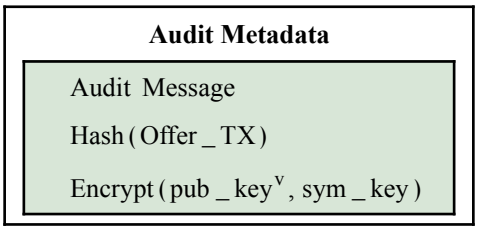


Fig. 6. Audit Message Metadata

where

- Audit Message: As above.
- $Hash(Offer\_TX)$: As above.
- $Encrypt(pub\_key^{v}, sym\_key)$: The same symmetric key that the Biological chose for the Input Message is encrypted with the Verifier's public key. This allows the Verifier to go to the blockchain, find the input and Output Messages associated with $Hash(Offer\_TX)$, decrypt the ciphertext inputs and outputs that are signed and attested to by the Biological and Mechanical, respectively, and run $proc_p$ independently.

B.2 Mechanical Message Metadata Content

Each Mechanical, $m \in \mathcal{M}$, monitors the blockchain for messages. When it sees an Offer Message containing ID_data^{m}/pub_key^{m} it considers the offer (Fee, P) and the Process ID it contains. If the Mechanical finds the offer acceptable, then it commits an **Accept Message** to the blockchain.

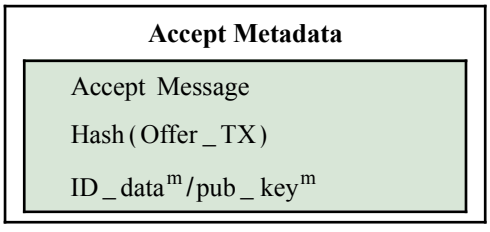


Fig. 7. Accept Message Metadata

where

- Accept Message: As above.
- $Hash(Offer\ TX)$: As above.

[8] For example, the hash of the concatenation of the Offer Message transaction hash, and the Merkle root of the block committed after the one containing the Output Message could be used as a seed.

- ID_data^m / pub_key^m : As above, and not strictly needed since the public key signing the transaction will also serve.

If the offer is not acceptable then the Mechanical commits a **Decline Message** to the blockchain.

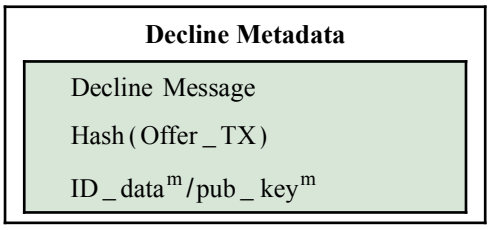


Fig. 8. Decline Message Metadata

where:

- Decline Message: As above.
- $Hash(Offer\ TX)$: As above.
- ID_data^m / pub_key^m : As above.

Suppose that the Mechanical accepts, and the Biological, in fact, commits a correct Input Message. Then the Mechanical decides on CORRECT or MALICIOUS execution, generates an output, and, commits an **Output Message** to the blockchain.

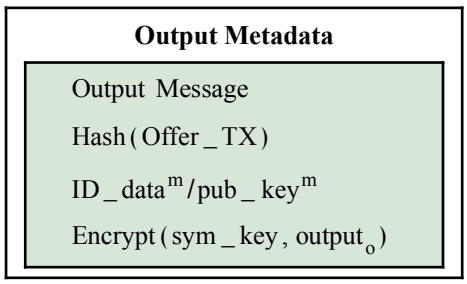


Fig. 9. Output Message Metadata

where:

- Output Message: As above.
- $Hash(Offer\_TX)$: As above.
- ID_data^m / pub_key^m : As above.
- $Encrypt(sym\_key, output_o)$: The Mechanical uses the same symmetric key as the Biological in its Input Message to encrypt the output it generates.

The Biological is required to undertake several actions correctly. If he does not, honest Mechanicals are not able to complete their side of the transaction, and should escape sanction. It may be that Biologicals should be sanctioned or labeled as non-cooperative in this event, but we leave this for the future. There are now two possibilities.

First, if Mechanical commits an Accept Message, but the Biological does not commit an Input Message before a certain number of blocks have passed, then the Mechanical commits a **No Response Message**,

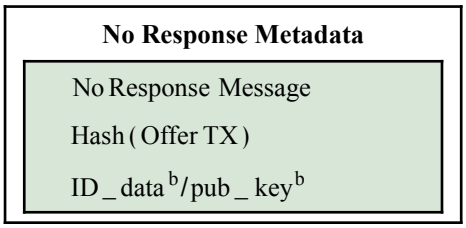


Fig. 10. No Response Message Metadata

where

- No Response Message: As above.
- $\text{Hash}(\text{Offer TX})$: As above.
- $\text{ID}\_\text{data}^{b}/\text{pub}\_\text{key}^{b}$: As above.

Second, if the Biological commits an Input Message that is flawed in one or more of the following ways:

- $\text{Hash}(\text{Coin}\_\text{TX}^{m})$ and/or $\text{Hash}(\text{Coin}\_\text{TX}^{v})$ is not actually be committed to the blockchain.
- $\text{Hash}(\text{Coin}\_\text{TX}^{m})$ and/or $\text{Hash}(\text{Coin}\_\text{TX}^{v})$ do not transfer the right fee, or are not to, or from, the right coin accounts.
- $\text{ID}\_\text{data}^{m}/\text{pub}\_\text{key}^{m}$, $\text{ID}\_\text{data}^{m}/\text{pub}\_\text{key}^{m}$, and/or $\text{Encrypt}(\text{pub}\_\text{key}^{m}, \text{sym}\_\text{key})$, are inconsistent with the original offer transaction, $\text{Hash}(\text{Offer}\_\text{TX})$, which is hash referenced in the message.

If so, then the Mechanical commits a **Flawed Input Message**,

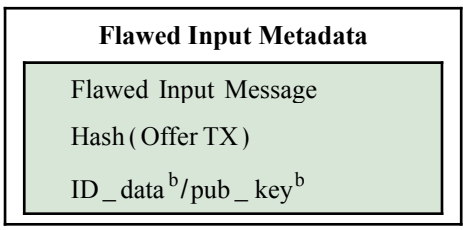


Fig. 11. Flawed Input Message Metadata

where

- Flawed Input Message: As above.
- $\text{Hash}(\text{Offer TX})$: As above.
- $\text{ID}\_\text{data}^{b}/\text{pub}\_\text{key}^{b}$: As above.

In both cases, the message should be seen by the Verifier who will commit a Verification Message, outlined below.

B.3 Verifier Message Metadata Content

Each Verifier, $v \in \mathcal{V}$, monitors the blockchain for certain messages, which it analyzes, and if required, chooses a verification code and then commits a Verification Message to the blockchain. Specifically:

- No Response Message from the Biological claiming that the Mechanical has neither accepted not declined: If true, then verification code = Dishonest Mechanical. If false, then verification code = Dishonest Biological.
- No Response Message from the Mechanical claiming that the Biological has not committed an Input Message despite the Mechanical having committed an Accept Message: If true, then verification code = Dishonest Biological. If false, then verification code = Dishonest Mechanical.
- Flawed Input Message from the Mechanical claiming that the Input Message committed by the Biological does not follow the game's messaging rules. If true, then verification code = Dishonest Biological. If false, then verification code = Dishonest Mechanical.
- No Response Message from the Biological claiming that the Mechanical has not committed an Output Message despite the Biological having committed an Input Message: If true, then verification code = Dishonest Mechanical: If false, then verification code = Dishonest Biological.
- An Output Message. If no audit is called for by the public randomization device, then verification code = Uncertain.
- An Audit Message from the Biological when one is required. In this case, the Verifier conducts an audit and decides on a verification code = Correct or Malicious.
- Finally, if an audit is called for, but the Biological fails to commit an Audit Message, the Verifier commits a Verification Message with verification code = Dishonest Biological.

In all cases, the Verifier can consult the block explorer to find any data needed to confirm or reject any of these claims or outcomes. When it decides on a verification code, the Verifier commits a Verification Message to the blockchain.

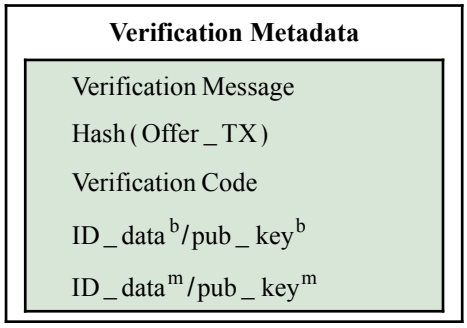


Fig. 12. Verification Message Metadata

where

- Verification Message: As above.
- Hash (Offer TX) : As above.
- Verification Code: As just described.
- ID _ data ^{b} / pub _ key ^{b} : As above, and not strictly needed, but makes the messaging more transparent.
- ID _ data ^{m} / pub _ key ^{m} : As above, and not strictly needed, but makes the

messaging more transparent.

Note that if the Biological sends a fake symmetric key in its Audit Message (or an incorrectly encrypted one) to the Mechanical, or if it encrypts an incorrect or unprocessable input, the Mechanical will return whatever garbage output results. The Biological will then be the party that the Verifier identifies as responsible in the event of an audit.

Also note that the Biological cannot send a symmetric key that is different from the one he used in this Input Message to the Mechanical. The Verifier generates a plaintext of the symmetric key from the encrypted one sent in the Audit Message. It then only needs to encrypt this with the Mechanical's public key to determine whether the Biological sent the same one as in its Input Message. Thus, the Verifier will have the same symmetric key used by the Biological and Mechanical in their exchange of messages. The Verifier will therefore end up with the same plaintext inputs and outputs and the two parties, and will be able to verify whether the Mechanical behaved honestly.

B.4 A Less Data Intensive Approach

Above, we described a robust, but informationally costly, approach to input, output, and Audit Messages. Specifically, the input and Output Messages contain the full ciphertext of the literal inputs and outputs. This makes it impossible for either the Biological or the Mechanical to deny what was sent or received, and allows the Verifier to determine the type of execution the Mechanical chose using only the relevant symmetric key.

If we are willing to allow more rounds of communication, then we can reduce the data burden of attestation transactions to the blockchain as follows:

- The Biological replaces $\text{Encrypt}(\text{pub\_key}^{m}, \text{sym\_key})$ and $\text{Encrypt}(\text{sym\_key}, \text{input}_i)$ in the Input Message with $\text{Hash}(\text{input}_i)$.
- If the Mechanical accepts, the Biological sends the Mechanical the full text of the input out-of-band.
- The Mechanical must then either commit an Acknowledgment Message that includes the hash of input to confirm what he received, or a No Response Message claiming the either it never got the input, or that it was different from the hash in the Input Message.
- In the event of a No Response Message from the Mechanical, the Biological must commit a new Input Message with the full ciphertext of the input.
- Things proceed as before until the Mechanical is ready to send its output. The pattern above is followed.
- The Mechanical replaces $\text{Encrypt}(\text{sym\_key}, \text{output}_o)$ with $\text{Hash}(\text{output}_o)$ in its Output Message and then sends the Biological the full text of the output out-of-band.
- The Biological must then either commit an Acknowledgment Message that includes the hash of its output to confirm what he received, or a No Response

Message claiming the either he never got the output, or that it was different from the hash in the Output Message.

- In the event of a No Response Message from the Biological, the Mechanical must commit a new Output Message with the full ciphertext of the output.
- If an audit is called for at this point, the Biological has both the input and output that were either hashed, or encrypted, and then committed to a block. If only the hashes are in the messages, the Biological is required to send the plaintext of both to the verifier out-of-band.
- If they are not committed, the Verifier commits a no response claim, and the Biological must commit the full the ciphertexts to a block or be judged dishonest. Since the signed hashes are in the chain, the Biological cannot send false inputs or outputs.

If data is in the blockchain, it is both provabley sent, and provabley received, at least within game messaging rules. Consequently, one would hope that in almost all cases, the existence of a mechanism that makes it impossible for agents to deny that they sent or received the full inputs or outputs would make its use rare. Sending full encrypted inputs and outputs through the blockchain is more costly to both parties, and does not produce a strategic advantage for either. Thus, signed hashes will most likely suffice.

References

1. Acemoglu, Daron, and Pascual Restrepo (2018) Artificial intelligence, automation, and work. In The economics of artificial intelligence: An agenda (pp. 197-236). University of Chicago Press.
2. AlAshery, Mohamed Kareem, Zhehan Yi, Di Shi, Xiao Lu, Chunlei Xu, Zhiwei Wang, and Wei Qiao. (2020). A blockchain-enabled multi-settlement quasi-ideal peer-to-peer trading framework. IEEE Transactions on Smart Grid 12, no. 1: 885-896.
3. Ball, Ian, and Deniz Kattwinkel (2019). Probabilistic Verification in Mechanism Design. 389-390. 10.1145/3328526.3329657.
4. Bebeshko, B., V. Malyukov, M. Lakhno, Pavlo Skladannyi, Volodymyr Sokolov, Svitlana Shevchenko, and M. Zhumadilova. (2022). Application of game theory, fuzzy logic and neural networks for assessing risks and forecasting rates of digital currency. Journal of Theoretical and Applied Information Technology 100, no. 4: 7390-7404.
5. Bichler, Martin, Maximilian Fichtl, Stefan Heidekrüger, Nils Kohring, and Paul Sutterer. (2021). Learning equilibria in symmetric auction games using artificial neural networks. Nature machine intelligence 3, no. 8: 687-695.
6. Calvano, Emilio, Giacomo Calzolari, Vincenzo Denicolo, and Sergio Pastorello. (2020). Artificial intelligence, algorithmic pricing, and collusion. American Economic Review 110, no. 10 : 3267-3297
7. Chaliasos, Stefanos, Marcos Antonios Charalambous, Liyi Zhou, Rafaila Galanopoulou, Arthur Gervais, Dimitris Mitropoulos, and Ben Livshits. (2020). Smart contract and defi security: Insights from tool evaluations and practitioner surveys. arXiv preprint arXiv:2304.02981 (2023).
8. Conley, John. (2024). AI Needs Blockchain: Trustless Solutions to Failures in Machine to Colloidal Markets.. Manuscript.

9. Gabriel, Iason. Artificial intelligence, values, and alignment. Minds and machines 30, no. 3: 411-437.
10. Glikson, Ella, and Anita Williams Woolley. (2020). Human trust in artificial intelligence: Review of empirical research. Academy of Management Annals 14, no. 2: 627-660.
11. Hardjono, Thomas, and Ned Smith.(2021). Towards an attestation architecture for blockchain networks. World Wide Web 24, no. 5: 1587-1615.
12. Hua, Weiqi, Jing Jiang, Hongjian Sun, and Jianzhong Wu. (2020). A blockchain based peer-to-peer trading framework integrating energy and carbon markets. Applied Energy 279 : 115539.
13. Lockey, Steven, Nicole Gillespie, Daniel Holm, and Ida Asadi Someh. (2021). A review of trust in artificial intelligence: Challenges, vulnerabilities and future directions.
14. Oksanen, Atte, Nina Savela, Rita Latikka, and Aki Koivula. (2020). Trust toward robots and artificial intelligence: An experimental approach to human–technology interactions online. Frontiers in Psychology 11: 568256.
15. Schär, Fabian. (2021). Decentralized finance: On blockchain-and smart contract-based financial markets. FRB of St. Louis Review.
16. Tan, Burcu, Edward G. Anderson Jr, and Geoffrey G. Parker. (2020). Platform pricing and investment to drive third-party value creation in two-sided networks. *Information Systems Research* 31, no. 1: 217-239.
17. Trammell, Philip, and Anton Korinek. (2023). Economic growth under transformative AI. No. w31815. National Bureau of Economic Research.
18. Wang, Qin, Rujia Li, Qi Wang, and Shiping Chen. (2021). Non-fungible token (NFT): Overview, evaluation, opportunities and challenges. *arXiv preprin*t arXiv:2105.07447
19. Zarifhonarvar, Ali. (2023) Economics of chatgpt: A labor market view on the occupational impact of artificial intelligence. *Available at SSRN* 4350925
20. Zeng, Rongfei, Chao Zeng, Xingwei Wang, Bo Li, and Xiaowen Chu. (2021). A comprehensive survey of incentive mechanism for federated learning. *arXiv preprint* arXiv:2106.15406
21. Zhang, Lejun, Jinlong Wang, Weizheng Wang, Zilong Jin, Yansen Su, and Huiling Chen. (2022). Smart contract vulnerability detection combined with multi-objective detection. Computer Networks 217: 109289.
22. Zhou, Yiyi. (2017). Bayesian estimation of a dynamic model of two-sided markets: Application to the U.S. video game industry. Management Science 63, 3874–3894.