

METHODS AND SYSTEMS FOR GENERATING PARETO-OPTIMAL SMART CONTRACTS IN A PARETO-OPTIMAL DECENTRALIZED APPLICATION

CROSS REFERENCE TO RELATED APPLICATIONS

5 [0001] This application claims priority to U.S. Provisional Application No. 63/616,763, filed on 31 December 2023, the entire disclosure of which is hereby incorporated by reference.

BACKGROUND

10 [0002] Exchange of value is at the heart of every human economic activity and collaboration. Providers and recipients agree on contracts that define the terms and conditions for achieving a fair and balanced value transfer in each direction. The provider produces a product and provides it to the recipient in exchange for settlement. If the product is an asset (e.g., delivery of goods or services), the recipient sends a cash payment to the provider. If it is a liability (e.g., risk or debt transfer), the provider sends the recipient a cash payment. However, almost all
15 exchanges of value are affected by the principal-agent problem: the provider (the agent) is better informed about the product than the recipient (the principal). The provider can use his informational advantage to ensure that the settlement is in his favour, resulting in a net loss of value for the recipient, known as agency cost (including fraud cases). Information asymmetry on the characteristics of the product (adverse selection) or on the behaviour of the provider (moral hazard) threatens the
20 equilibrium of contracts. The lack of transparent information and incentives reduces the market efficiency associated with these contracts and sometimes leads to market failure.

25 [0003] Conventionally, most of these contracts take the form of traditional contracts and are not economically effective because their interpretation renders them incomplete, and their formation, negotiation, performance, enforceability and opposability entail large friction costs (including intermediary costs that are often prohibitive, particularly within the legal industry).

30 [0004] Smart contracts, on the other hand, reduce the need for intermediaries and are determined entirely by code. The decentralized nature of blockchain technology enables these smart contracts to be efficient, self-executing, and censorship-free. The use of smart contracts reduces the cost of friction in the exchange of value and makes it possible to design directly enforceable incentive mechanisms, but the Pareto optimal parametrization of these mechanisms, integration of real-world data,
35 and a solution to the principal-agent problem remain unsolved.

[0005] Therefore, improved methods of generating and facilitating smart contracts are needed.

SUMMARY

5 [0006] This Summary is intended to introduce, in an abbreviated form, various topics to be elaborated upon below in the Detailed Description. This Summary is not intended to identify key or essential aspects of the claimed invention. This Summary is similarly not intended for use as an aid in determining the scope of the claims.

10 [0007] In some aspects, the techniques described herein relate to a method, including: receiving, at a processor of a node of a plurality of nodes, a set of mechanisms, wherein each mechanism defined by at least a provider action and a feasible contract; defining, using the processor, a recipient loss function based on the set of mechanisms; determining, using the processor, a mechanism of the set of mechanisms that minimizes of a recipient loss function; generating, using the processor, a smart contract based on the mechanism; deploying, using the processor, the smart contract to a blockchain maintained across a plurality of nodes connected via a computer network by broadcasting the smart contract to the plurality of nodes via a network interface in electronic communication with the processor and the computer network; receiving, at the processor, a transaction address from the blockchain corresponding to the smart contract; and storing, using the processor, the transaction address in a database on an electronic storage device of the node and in electronic communication with the processor.

15 [0008] In some aspects, the techniques described herein relate to a system, including: a processor of a node of a plurality of nodes; a network interface in electronic communication with the processor and a computer network; and an electronic storage device in electronic communication with the processor, the electronic storage device having a database stored thereon; wherein the processor is configured to: receive a set of mechanisms, wherein each mechanism defined by at least a provider action and a feasible contract; define a recipient loss function based on the set of mechanisms; determine a mechanism of the set of mechanisms that minimizes of a recipient loss function; generate a smart contract based on the mechanism; deploy the smart contract to a blockchain maintained across a plurality of nodes connected via a computer network by broadcasting the smart contract to the plurality of nodes via the network interface; receive a transaction address from the blockchain corresponding to the smart contract; and store the transaction address in the database of the electronic storage device.

[0009] In some aspects, the techniques described herein relate to a tangible, non-transitory, computer-readable medium having instructions thereupon which when implemented by a processor of a node of a plurality of nodes cause the processor to perform a method including: Receiving a set of mechanisms, wherein each
5 mechanism defined by at least a provider action and a feasible contract; defining a recipient loss function based on the set of mechanisms; determining a mechanism of the set of mechanisms that minimizes of a recipient loss function; generating a smart contract based on the mechanism; deploying the smart contract to a
10 blockchain maintained across a plurality of nodes connected via a computer network by broadcasting the smart contract to the plurality of nodes via a network interface in electronic communication with the processor and the computer network; receiving a transaction address from the blockchain corresponding to the smart contract; and storing the transaction address in a database on an electronic storage device of the node and in electronic communication with the processor.

15 BRIEF DESCRIPTION OF THE FIGURES

- [0010] For a fuller understanding of the nature and objects of the disclosure, reference should be made to the following detailed description taken in conjunction with the accompanying drawings, in which:
- [0011] FIG. 1 illustrates an example method illustrating a model based on the Optimal
20 Smart Contract Resolution Algorithm, according to one or more implementations herein.
- [0012] FIG. 2 illustrates a system for generating and implementing Optimal Smart Contracts (OSCs), according to one or more implementations herein.
- [0013] FIG. 3 illustrates a plot of the minting function, according to one or more
25 implementations herein.
- [0014] FIG. 4 Illustrates an operational environment, according to one or more of the implementations herein.
- [0015] FIG. 5 is a diagram of example components of a device, according to one or more implementations herein.
- 30 [0016] FIG. 6 is a diagram of example components of a device, according to one or more implementations herein.
- [0017] FIG. 7 is a flowchart illustrating an example method, according to one or more implementations herein.

[0018] FIG. 8 illustrates an artificial neural network (ANN), according to an implementation.

[0019] FIG. 9 illustrates a node of a machine learning model, according to an implementation.

5 [0020] FIG. 10 illustrates a method of training a machine learning model of a machine learning module, according to an implementation.

[0021] FIG. 11 illustrates a method of analyzing input data using a machine learning module, according to an implementation.

DETAILED DESCRIPTION

10 [0022] Implementations disclosed herein solve the principal-agent problem by presenting a novel solution for generating and establishing smart contracts.

[0023] Crypto-assets can be transferred on a sufficiently decentralized blockchain without relying on trust. For on-chain value exchange of real-world assets (RWAs) and real-world liabilities (RWLs), blockchain technology is not suitable on its own to avoid relying on trust due to the oracle problem. The oracle problem may refer to the
15 underlying inability of blockchains and smart contracts to access real-world, or “off-chain,” data. An oracle may provide data and source of truth for smart contracts.

[0024] However, smart contracts do not have access to oracles that are specific to exchanges of value, in particular to obtain data relating to the characteristics of the product and the provider’s actions. In addition, they must provide off-chain
20 computational intelligence that reduces agency problems. If this is not the case, the savings in friction costs will no longer compensate for the increase in agency costs. Without solving the oracle problem for this specific data, the irreversible nature of these value exchanges makes the principal-agent problem even more acute.

25 [0025] Contract theory may refer to the study and modelling the behavior of economic agents within a contractual relationship in the presence of asymmetric information. Contract theory may provide tools for determining the Pareto optimal contract, including signaling and screening to limit adverse selection and the design of Pareto optimal incentive mechanisms to mitigate moral hazard. It can contribute to the development of new products that are profitable, competitive and sustainable,
30 with a good incentive structure. Above all, it allows recipients to avoid financing high agency costs. However, in view of the technical limitations of traditional contracts and conventional smart contracts, contract theory has neither been adequately practically applied nor is capable of solving the principal-agent problem in real value exchanges.

- [0026] Implementations herein address these shortcomings and enable industrial applications of contract theory. Implementations enable recipients to minimize agency and frictional cost in the exchange of value with providers across potentially all industries. Implementations solve the oracle problem, addressing frictional costs associated with the need for trusted third parties and the principal-agent problem.
- [0027] Implementations herein proffer applied solutions to the principal-agent problem and its underlying oracle problem in the exchange of value between the provider and the recipient by developing a general framework for optimal smart contracts (OSCs), using blockchain and AI/ML technologies.
- [0028] Maximum information on the characteristics of the product and the behavior of the provider is provided to the recipient, while preserving the privacy of recipients and providers. Next, the oracle is used to solve the OSC resolution algorithm (OSCRA), generating Pareto optimal incentive mechanisms between the recipient and the provider, thereby restoring the balance of contractual relationships in the initial presence of information asymmetry.
- [0029] Implementations may achieve Pareto optimality in all exchanges of value, in all sectors of activity, through the tailor-made development of decentralized applications (DApps) in each corresponding industry. Governance of such DApps may be fully decentralized, for example, using a decentralized autonomous organization (DAO), based solely on usage and totally under the control of recipients as DAO members whose humanity and uniqueness has been verified by a zero knowledge proof protocol. The supply of the DAO's economic token may be minted slowly as a function of time and usage, for example, until it is used globally to ensure the sustainable development of such DApps. Demand may be driven by usage.
- [0030] Implementations may account for provider types, provider actions or effort, signals, and rewards as inputs to the OSCRA. The sets of types θ , actions $\mathcal{A}$, signals $\mathcal{S}$, and rewards $\mathcal{R}$ can be multi-dimensional and may even be a wide range of function spaces. These sets θ , $\mathcal{A}$, $\mathcal{S}$, and $\mathcal{R}$ may be respectively associated with their sigma-algebras of measurable subsets $\mathcal{F}_\theta$, $\mathcal{F}_\mathcal{A}$, $\mathcal{F}_\mathcal{S}$, and $\mathcal{F}_\mathcal{R}$. Therefore, $(\theta, \mathcal{F}_\theta)$, $(\mathcal{A}, \mathcal{F}_\mathcal{A})$, $(\mathcal{S}, \mathcal{F}_\mathcal{S})$, and $(\mathcal{R}, \mathcal{F}_\mathcal{R})$ are measurable spaces.
- [0031] The set of feasible rewards can depend on the signal and is captured by Φ , a measurable subset of $\mathcal{R} \times \mathcal{S}$. For each signal s , the set of feasible rewards given the signal s is $\Phi_s = \{r : (r, s) \in \Phi\}$.

- [0032] We denote Δ as the function which, given a set X endowed with a measurable space $(X, \mathcal{F}_X)$, gives the set of probability measures $\mathbb{P}$ on the measurable subsets of $\mathcal{F}_X$. Let $\mathbb{Q} \in \Delta(\Theta)$, $\mathbb{A} \in \Delta(\mathcal{A})$, $\mathbb{S} \in \Delta(\mathcal{S})$, $\mathbb{C} \in \Delta(\mathcal{R})$. As a result, $(\Theta, \mathcal{F}_\Theta, \mathbb{Q})$, $(\mathcal{A}, \mathcal{F}_\mathcal{A}, \mathbb{A})$, $(\mathcal{S}, \mathcal{F}_\mathcal{S}, \mathbb{S})$, and $(\mathcal{R}, \mathcal{F}_\mathcal{R}, \mathbb{C})$ are probability spaces.
- 5 [0033] We may denote the provider's von Neumann-Morgenstern utility function u and the recipient's von Neumann-Morgenstern loss (disutility) function l as follows: $u : \mathcal{R} \times \mathcal{S} \times \mathcal{A} \times \Theta \rightarrow \mathbb{R}$ and $l : \mathcal{R} \times \mathcal{S} \times \mathcal{A} \times \Theta \rightarrow \mathbb{R}$. u and l are measurable.
- [0034] We may denote $\mathbb{A}_\theta$ as the conditional probability measure $\mathbb{A}$ conditional on the event $\theta \in \Theta$ and $\mathbb{C}_{s,a,\theta}$ as the conditional probability measure on the event $(s, a, \theta) \in \mathcal{S} \times \mathcal{A} \times \Theta$.
- 10 [0035] Applying the principle of revelation:
- Definition 1.** A mechanism is defined as any tuple $(\mathbb{A}, \mathbb{C})$ in which $\mathbb{A} : \Theta \rightarrow \Delta(\mathcal{A})$ and $\mathbb{C} : \mathcal{S} \times \mathcal{A} \times \Theta \rightarrow \Delta(\mathcal{R})$ are transition probabilities satisfying $\mathbb{C}(\Phi_s \mid s, a, t) = 1$ for every $(s, a, t) \in \mathcal{S} \times \mathcal{A} \times \Theta$.
- 15 [0036] A mechanism, denoted by $(\mathbb{A}, \mathbb{C})$, may operate in the following way: the provider's type may be drawn from Θ by nature, based on $\mathbb{Q}$. Once the provider learns their true type, θ , the provider may report a type, θ' to the mechanism. The mechanism may then recommend to the provider an action a' that may be generated by the probability measure $\mathbb{A}_{\theta'} \in \Delta(\mathcal{A})$. After learning the recommended action a' , the
- 20 provider may choose an action a from $\mathcal{A}$. It must be noted that the report on the type and choice of action by the provider need not alter the fact that contracts can remain "self-executing."
- [0037] Finally, given the signal s generated by $\mathcal{S}_{a,\theta}$, the mechanism may generate the provider's reward r according to the probability measure $\mathbb{C}_{s,a',\theta'}$. Signals may be
- 25 generated according to the provider's true type and action, while rewards may depend on the reported type and recommended action.
- [0038] The function $\mathbb{C}_{a,\theta} : \mathcal{S} \rightarrow \Delta(\mathcal{R})$, which gives a conditional probability measure in $\Delta(\mathcal{R})$ for a given set of type and action $(a, \theta) \in \mathcal{A} \times \Theta$ is then interpreted as a contract. The formulation of the principal-agent problem herein may be general enough to
- 30 allow the recipient to randomize the rewards offered to the provider according to the observed signal. The provider may know the design of the contract before choosing his action.
- [0039] **Definition 2.** We denote the expected utility $U(\mathbb{A}, \mathbb{C})$ of a single provider:

$$U(\mathbb{A}, \mathbb{C}) \equiv \int_{\mathcal{R} \times \mathcal{S} \times \mathcal{A} \times \Theta} u(r, s, a, \theta) d\mathbb{C}_{s,a,\theta}(r) d\mathbb{S}_{a,\theta}(s) d\mathbb{A}_{\theta}(a) d\mathbb{Q}(\theta)$$

[0040] **Definition 3.** We denote the expected loss $L(\mathbb{A}, \mathbb{C})$ of the recipient:

$$L(\mathbb{A}, \mathbb{C}) \equiv \int_{\mathcal{R} \times \mathcal{S} \times \mathcal{A} \times \Theta} l(r, s, a, \theta) d\mathbb{C}_{s,a,\theta}(r) d\mathbb{S}_{a,\theta}(s) d\mathbb{A}_{\theta}(a) d\mathbb{Q}(\theta)$$

[0041] **Definition 4.** We define a mechanism $(\mathbb{A}, \mathbb{C})$ satisfying Incentive Compatibility (IC) in a single-provider model if for $\mathbb{Q}$ -almost every type θ and for every type θ' of the single provider,

$$\begin{aligned} & \int_{\mathcal{R} \times \mathcal{S} \times \mathcal{A}} u(r, s, a, \theta) d\mathbb{C}_{s,a,\theta}(r) d\mathbb{S}_{a,\theta}(s) d\mathbb{A}_{\theta}(a) \\ & \geq \int_{\mathcal{A}} \left(\sup_{a \in \mathcal{A}} \int_{\mathcal{R} \times \mathcal{S}} u(r, s, a, \theta) d\mathbb{C}_{s,a,\theta'}(r) d\mathbb{S}_{a,\theta}(s) \right) d\mathbb{A}_{\theta'}(a') \end{aligned}$$

[0042] $\mathcal{M}_{IC}$ is denoted as the set of mechanisms satisfying Incentive Compatibility (IC) (also called incentive compatible mechanisms) in a single-provider model on $\mathcal{R} \times \mathcal{S} \times \mathcal{A} \times \Theta$.

[0043] **Definition 5.** We define a mechanism $(\mathbb{A}, \mathbb{C})$ satisfying Individual Rationality (IR) if there is a real $\underline{U}$ for which:

[0044] $U(\mathbb{A}, \mathbb{C}) \geq \underline{U}$

[0045] $\mathcal{M}_{IR}$ is denoted as the set of mechanisms satisfying Individual Rationality (IR) on $\mathcal{R} \times \mathcal{S} \times \mathcal{A} \times \Theta$.

[0046] **Definition 6.** We define a mechanism $(\mathbb{A}, \mathbb{C})$ satisfying Other Constraints (OC) if it verifies any other constraints other than Incentive Compatibility (IC) or individual rationality (IR) on the provider or the recipient, in relation for example to solvency, regulatory, technical, market dynamics, etc.

[0047] $\mathcal{M}_{OC}$ is denoted as the set of mechanisms satisfying Other Constraints (OC) on $\mathcal{R} \times \mathcal{S} \times \mathcal{A} \times \Theta$.

[0048] **Definition 7.** We define a mechanism $(\mathbb{A}, \mathbb{C})$ as feasible if it satisfies Incentive Compatibility (IC), Individual Rationality (IR), and Other Constraints (OC).

[0049] $\mathcal{M} = \mathcal{M}_{IC} \cap \mathcal{M}_{IR} \cap \mathcal{M}_{OC}$ is denoted as the set of feasible mechanisms on $\mathcal{R} \times \mathcal{S} \times \mathcal{A} \times \Theta$.

[0050] For any feasible mechanism $(\tilde{\mathbb{A}}, \tilde{\mathbb{C}}) \in \mathcal{M}$, let

$$L(\tilde{\mathbb{A}}, \tilde{\mathbb{C}}) \equiv \int_{\mathcal{R} \times \mathcal{S} \times \mathcal{A} \times \Theta} l(r, s, a, \theta) d\tilde{\mathbb{C}}_{s,a,\theta}(r) d\mathbb{S}_{a,\theta}(s) d\tilde{\mathbb{A}}_{\theta}(a) d\mathbb{Q}(\theta),$$

be the recipient's expected loss when the provider reports honestly and takes the recommended action.

[0051] **Definition 8.** *The Principal-Agent problem between a provider (Agent) and the recipient (Principal) for a set of feasible mechanisms $\mathcal{M}$ is as follows:*

[0052] $\min_{(\tilde{\mathbb{A}}, \tilde{\mathbb{C}}) \in \mathcal{M}} L(\tilde{\mathbb{A}}, \tilde{\mathbb{C}}).$

[0053] *The set of solutions to this Principal-Agent problem are called Pareto optimal mechanisms and therefore define the Pareto optimal contracts associated solutions between the provider and the recipient.*

10 [0054] According to Kadan, Reny, and Swinkels (2017), there is a set of known conditions that can be used to prove the existence of an Pareto optimal mechanism and therefore a Pareto optimal contract solution.

Assumptions

15 [0055] We refer here extensively to the definition and main result on the existence of a Pareto optimal mechanism of Kadan, Reny, and Swinkels (henceforth KRS) from their paper *Existence of optimal mechanisms in principal-agent problems* (2017), the entirety of which is hereby incorporated by reference.

[0056] Recall that a Polish space is a separable topological space that can be metrized by means of a complete metric. The measurable subsets of any Polish space $\mathcal{X}$ will always be the Borel sets $\mathcal{B}(\mathcal{X})$, and $\Delta(\mathcal{X})$ will denote the space of probability measures on $\mathcal{B}(\mathcal{X})$ endowed with the topology of weak convergence (the weak* topology).

Ambient Space Assumptions

[0057] **Assumption 1.** $\mathcal{R}, \mathcal{S}, \mathcal{A}$, and θ are nonempty Polish spaces, and $\mathcal{A}$ is compact.

25 [0058] **Assumption 2.** Φ is a closed subset of $\mathcal{R} \times \mathcal{S}$, where, for each s , $\Phi_s = \{r: (r, s) \in \Phi\}$ is nonempty.

Payoff Assumptions

[0059] **Assumption 3.** $u: \mathcal{R} \times \mathcal{S} \times \mathcal{A} \times \Theta \rightarrow \mathbb{R}$ and $l: \mathcal{R} \times \mathcal{S} \times \mathcal{A} \times \Theta \rightarrow \mathbb{R}$ are measurable and bounded below, without loss of generality by 0, and, for every $(a, \theta) \in \mathcal{A} \times \Theta$ and for $\mathbb{P}_{a,\theta}$ almost everywhere $s \in \mathcal{S}$, $u(\cdot, \theta)$ is continuous at (r, s, a) and $l(\cdot, \theta)$ is lower semicontinuous at (r, s, a) for all $r \in \mathcal{R}$.

[0060] **Assumption 4.** For any type θ , for any $c \in \mathbb{R}$, and for any compact subset Y of $\mathcal{S}$, the closure of $\{(r, s, a) \in \Phi \times \mathcal{A} : s \in Y \text{ and } l(r, s, a, \theta) \leq c\}$ is compact.

[0061] **Assumption 5.** For any type θ , and for any sequence (r_n, s_n, a_n) in $\Phi \times \mathcal{A}$, if $u(r_n, s_n, a_n, \theta) \rightarrow \infty$, then $l(r_n, s_n, a_n, \theta)/u(r_n, s_n, a_n, \theta) \rightarrow \infty$.

5 *Information Assumptions*

[0062] **Assumption 6.** The transition probability $\mathbb{P} : \mathcal{A} \times \theta \rightarrow \Delta(\mathcal{S})$ is such that $\mathbb{P}_{a,\theta}$ is continuous in a for each θ .

[0063] **Assumption 7.** There is a collection $\{\mathcal{S}_{a,\theta}\}_{(a,\theta) \in \mathcal{A} \times \theta}$ of subsets of $\mathcal{S}$ such that $\{(s, a, \theta) : s \in \mathcal{S}_{a,\theta}\}$ is a measurable subset of $\mathcal{S} \times \mathcal{A} \times \theta$; $\mathbb{P}_{a,\theta}(\mathcal{S}_{a,\theta}) = 1$ for all a, θ ; and $\mathbb{P}_{a',\theta'}(E) = 0 \Rightarrow \mathbb{P}_{a,\theta}(E) = 0$ for all a, θ, a', θ' and for all measurable $E \subset \mathcal{S}_{a',\theta'}$.

10

[0064] Under Assumption 7, $\mathbb{P}_{a,\theta}$ is absolutely continuous with respect to $\mathbb{P}_{a',\theta'}$ when both measures are restricted to the measurable subsets of $\mathcal{S}_{a',\theta'}$. Equivalently, the unrestricted measure $\mathbb{P}_{a,\theta}(\cdot \cap \mathcal{S}_{a',\theta'})$ on the measurable subsets of $\mathcal{S}$ is absolutely continuous with respect to the unrestricted measure $\mathbb{P}_{a',\theta'}$. Applying the Radon-Nikodym theorem to the unrestricted measures, a Radon-Nikodym derivative defined on all of $\mathcal{S}$ exists and so we denote by $g_{a,\theta/a',\theta'} : \mathcal{S} \rightarrow [0, +\infty]$ any version of the Radon-Nikodym derivative of $\mathbb{P}_{a,\theta}(\cdot \cap \mathcal{S}_{a',\theta'})$ with respect to $\mathbb{P}_{a',\theta'}$.

15

[0065] **Assumption 8.** For all a, θ, a', θ' , and for every sequence $a'_n \rightarrow a'$, there is $a_n \rightarrow a$ and there are versions of the Radon-Nikodym derivatives $g_{a_n,\theta/a'_n,\theta'}$ and $g_{a,\theta/a',\theta'}$ such that

20

$$\lim_n g_{a_n,\theta/a'_n,\theta'}(s_n) \geq g_{a,\theta/a',\theta'}(s) \quad \forall s \in \mathcal{S} \text{ and } \forall s_n \rightarrow s$$

[0066] **Assumption 9.** If the collection $\{\mathcal{S}_{a,\theta}\}$ in Assumption 7 is such that $\mathbb{P}_{a,\theta}(\mathcal{S}_{a',\theta'}) < 1$ for some a, θ, a', θ' , then there is $r_* \in \mathcal{R}$ such that $r_* \in \Phi_s$ for all $s \in \mathcal{S}$, and $u(r_*, s, a, \theta) \leq u(r, s, a, \theta)$ for all $(r, s, a, \theta) \in \Phi \times \mathcal{A} \times \theta$.

25 [0067] For numerous applications, it is possible to verify all nine of our assumptions by recasting one's model so that only Assumptions 1-5 need be checked. One such leading case is when all spaces are Euclidean and the signal distribution admits a density that is continuous in s, a , and θ .

Theorems

30 [0068] **Theorem 1.** Endow $\mathcal{M}$ with the $d_{\mathcal{M}}$ -metric. Under Assumptions 1-9, $\mathcal{M}_c$ is a compact subset of $\mathcal{M}$ for all $c \geq 0$, and the principal's loss function $L : \mathcal{M} \rightarrow [0, +\infty]$

defined in (1) is lower semicontinuous. Thus, if $\mathcal{M}$ is nonempty, then the principal's problem (2) possesses a solution.

[0069] We can now refer to the main result of Kadan, Reny, and Swinkels.

[0070] **Theorem 2.** *If Assumptions 1-9 are satisfied, then, provided that there is at least one incentive compatible mechanism, the Principal-Agent problem has a solution.*

[0071] The set of incentive compatible mechanisms can be metrized so that (i) for any constant, the subset of incentive compatible mechanisms that bounds the principal's expected losses weakly below that constant is compact, and (ii) the principal's loss function is lower semicontinuous as a function of the mechanism and that the set of incentive compatible mechanisms is nonempty. While in applications it is often easy to establish the existence of at least one incentive compatible mechanism, the following proposition (Proposition 1) provides general conditions under which an incentive compatible mechanism exists.

[0072] **Proposition 1.** *Suppose that Assumptions 1, 3, and 6 hold and that there is a measurable $\phi: \mathcal{S} \rightarrow \mathcal{R}$ such that $\phi(s) \in \Phi_s$ for every s , and such that,*

$$\int_{\mathcal{S}} u(\phi(s), s, a, \theta) d\mathbb{P}_{a, \theta}(s)$$

is continuous for each θ .

[0073] *Then an incentive compatible mechanism exists, that is, $\mathcal{M}$ is nonempty.*

[0074] Given the other assumptions, the displayed continuity requirement in the last is satisfied if either (a) some fixed reward is always feasible and the agent's utility given that reward does not depend on the signal, or (b) the function ϕ is continuous and, for each θ , $\int_{[x: n(\phi(s), s, a, \theta) > n]} u(\phi(s), s, a, \theta) d\mathbb{P}_{a, \theta}(s) \rightarrow_n 0$ uniformly in $a \in \mathcal{A}$.

[0075] Theorem 2 follows directly from Theorem 1, according to which the set of incentive compatible mechanisms can be metrized so that (i) for any constant, the subset of incentive compatible mechanisms that bounds the principal's expected losses weakly below that constant is compact, and (ii) the principal's loss function is lower semicontinuous as a function of the mechanism and that the set of incentive compatible mechanisms is nonempty. While in applications it is often easy to establish the existence of at least one incentive compatible mechanism, Proposition 1 provides general conditions under which an incentive compatible mechanism exists.

Principal-Agent Model with Multiple Providers

[0076] Let's now introduce providers indexed by $i \in \{1, \dots, I\}$, and recipients indexed by $j \in \{1, \dots, J\}$. In the context of this section, we omit the j because we are talking about recipient j without loss of generality.

[0077] In this section, KRS show how Theorem 1 leads to an existence result for a model in which a single recipient interacts with multiple providers. The notation will be as in the single provider case except for the presence of a subscript $i \in \{1, \dots, I\}$ for each of the I providers. So, θ_i and $\mathcal{A}_i$ are provider i 's type and action spaces, and we define $\theta = \times_i \theta_i$ and $\mathcal{A} = \times_i \mathcal{A}_i$. We shall assume that the providers' types are drawn from θ according to $\mathbb{Q} \in \Delta(\theta)$. Provider i 's utility function is $u_i: \mathcal{R} \times S \times \mathcal{A} \times \theta \rightarrow \mathcal{R}$, and the recipient's loss function is $l: \mathcal{R} \times S \times \mathcal{A} \times \theta \rightarrow \mathcal{R}$. A mechanism here is $(\mathbb{A}, \mathbb{C})$ specifying transition probabilities $\mathbb{C}: S \times \mathcal{A} \times \theta \rightarrow \Delta(\mathcal{R})$ and $\alpha: \theta \rightarrow \Delta(\mathcal{A})$ and where $\mathbb{C}(\Phi_\theta \mid s, a, \theta) = 1$ for all $(s, a, \theta) \in S \times \mathcal{A} \times \theta$.

[0078] The mechanism $(\mathbb{A}, \mathbb{C})$ works as follows. Nature draws a vector of types θ according to $\mathbb{Q}$. Each provider i simultaneously learns his type θ_i , and then privately reports his type to the mechanism. For any vector of reported types $\theta' \in \theta$, the mechanism chooses a vector of actions $a' \in \mathcal{A}$ according to $\mathbb{A}(\cdot \mid \theta')$. Each provider i is then privately informed of his recommended action a'_i . Then all providers simultaneously choose an action, with provider i choosing an action from $\mathcal{A}_i$. If the vector of chosen actions is $a \in \mathcal{A}$, then the signal $s \in S$ is drawn according to $\mathbb{S}_{u, \theta} \in \Delta(S)$. Finally, a reward $r \in \mathcal{R}$ is drawn according to $\mathbb{C}(\cdot \mid s, a', \theta') \in \Delta(\mathcal{R})$. Provider i receives utility $u_i(r, s, a, \theta)$ and the recipient receives loss $l(r, s, a, \theta)$.

Absolute Continuity: Types and Signals

[0079] For every provider i , let $\mathbb{Q}_i$ denote the marginal of $\mathbb{Q}$ on θ_i and let $\tilde{\mathbb{Q}} = \times_i \mathbb{Q}_i$ denote the product of the marginals. We shall assume from now on that $\mathbb{Q}$ is absolutely continuous with respect to the product of its marginals $\tilde{\mathbb{Q}}$. Consequently, by the Radon-Nikodym theorem, there is a measurable $h: \theta \rightarrow [0, \infty)$ such that, for every measurable $\mathcal{F} \subseteq \theta$,

$$\mathbb{Q}(\mathcal{F}) = \int_{\mathcal{F}} h(\theta) d\mathbb{Q}_1(\theta_1) \dots d\mathbb{Q}_I(\theta_I) = \int_{\mathcal{F}} h(\theta) d\tilde{\mathbb{Q}}(\theta)$$

[0080] **Assumption 10.** *There is $\tilde{\mathbb{S}} \in \Delta(S)$ such that, for every (a, θ) , $\mathbb{S}_{a, \theta}$ is absolutely continuous with respect to $\tilde{\mathbb{S}}$. Specifically, we assume that there is a measurable $f: S \times \mathcal{A} \times \theta \rightarrow [0, \infty)$ such that, for every $(a, \theta) \in \mathcal{A} \times \theta$ and for every measurable subset $\mathcal{E}$ of S ,*

$$\mathbb{S}(\mathcal{E} \mid a, \theta) = \int_{\mathcal{E}} f(s \mid a, \theta) d\tilde{\mathbb{S}}(s)$$

[0081] We shall also assume from now on Assumption 10. This is the same assumption that was made in the single-provider model to arrive at the "standard case".

Incentive Compatibility

5 [0082] For any mechanism $(\mathbb{A}, \mathbb{C})$, and for any provider i , by Bertsekas and Shreve (1978), Proposition 7.27, we may decompose $\bar{\mathbb{Q}} \otimes \mathbb{A} \in \Delta(\mathcal{A} \times \theta)$ as $\bar{\mathbb{Q}} \otimes \mathbb{A} = \mathbb{Q}_i \otimes \mathbb{A}_i \otimes \mathbb{B}_i$ for some transition probabilities $\mathbb{A}_i: \theta_i \rightarrow \Delta(\mathcal{A}_i)$ and $\mathbb{B}_i: \mathcal{A}_i \times \theta_i \rightarrow \mathcal{A}_{-i} \times \theta_{-i}$. Moreover, the transition probability $\mathbb{A}_i$ is unique up to an $\mathbb{Q}_i$ measure zero set of types, and the transition probability $\mathbb{B}_i$ is unique up to subset of $\mathcal{A}_i \times \theta_i$ that has measure zero according to the marginal of $\bar{\mathbb{Q}} \otimes \mathbb{A}$ on $\mathcal{A}_i \times \theta_i$, which, it should be noted, is equal to $\mathbb{Q}_i \otimes \mathbb{A}_i$. Because $\mathbb{A}_i$ is unique only $\mathbb{Q}_i$ almost everywhere, it is most natural to define incentive compatibility so that only almost all (rather than all) untruthful reports are non-improving.

15 [0083] **Definition 9.** We define a mechanism $(\mathbb{A}, \mathbb{C})$ satisfying Incentive Compatibility (IC) in a multi-provider model if, for every provider i , there is a measurable $\mathcal{F}_i \subseteq \theta_i$ such that $\mathbb{Q}_i(\mathcal{F}_i) = 1$, and, for all $\theta_i, \theta'_i \in \mathcal{F}_i$:

$$\begin{aligned} & \int_{\mathcal{R} \times \mathcal{S} \times \mathcal{A} \times \theta_{-i}} u_i(r, s, a, \theta_i, \theta_{-i}) h(\theta_i, \theta_{-i}) \\ & \quad \times f(s \mid a, \theta_i, \theta_{-i}) d\mathbb{C}(r \mid s, a, \theta_i, \theta_{-i}) d\tilde{\mathbb{S}}(s) d\mathbb{B}_i(a_{-i}, \theta_{-i} \mid a_i, \theta_i) d\mathbb{A}_i(a_i \mid \theta_i) \\ & \geq \int_{\mathcal{A}_i} \sup_{a_i \in \mathcal{A}_i} \left\{ \int_{\mathcal{R} \times \mathcal{S} \times \mathcal{A}_{-i} \times \theta_{-i}} u_i(r, s, a_i, a_{-i}, \theta_i, \theta_{-i}) h(\theta_i, \theta_{-i}) \times f(s \mid a_i, a_{-i}, \theta_i, \theta_{-i}) d\mathbb{C}(r \right. \\ & \quad \left. \mid s, a'_i, a_{-i}, \theta'_i, \theta_{-i}) d\tilde{\mathbb{S}}(s) d\mathbb{B}_i(a_{-i}, \theta_{-i} \mid a'_i, \theta'_i) \right\} d\mathbb{A}_i(a'_i \mid \theta'_i). \end{aligned}$$

20

[0084] $\mathcal{M}_{IC}$ is denoted as the set of mechanisms satisfying Incentive Compatibility (IC) (also called incentive compatible mechanisms) in a multi-provider model on $\mathcal{R} \times \mathcal{S} \times \mathcal{A} \times \theta$.

25 [0085] **Definition 10.** We denote the expected utility $U_i(\mathbb{A}, \mathbb{C})$ of provider i in a multi-provider model, with $i \in \{1, \dots, I\}$ is:

$$\begin{aligned} U_i(\mathbb{A}, \mathbb{C}) & \equiv \int_{\mathcal{R} \times \mathcal{S} \times \mathcal{A} \times \theta_{-i} \times \theta_i} u_i(r, s, a, \theta_i, \theta_{-i}) h(\theta_i, \theta_{-i}) \\ & \quad \times f(s \mid a, \theta_i, \theta_{-i}) d\mathbb{C}(r \mid s, a, \theta_i, \theta_{-i}) d\tilde{\mathbb{S}}(s) d\mathbb{B}_i(a_{-i}, \theta_{-i} \mid a_i, \theta_i) d\mathbb{A}_i(a_i \\ & \quad \mid \theta_i) d\mathbb{Q}_i(\theta_i) \end{aligned}$$

[0086] To obtain type θ_i 's conditional expected utility, divide both sides of Definition 9 by $\int h(\theta_i, \theta_{-i}) d(\sum_{k \neq i} \mathbb{Q}_k(\theta_k))$ when this quantity is positive.

[0087] The condition on $\mathbb{Q}(\mathcal{F})$ has the important implication that certain conditional distributions depend on i 's true type θ_i only through the Radon-Nikodym derivative $h(\theta_i, \theta_{-i})$. These conditional distributions are, first, the conditional distribution over the recommended action a'_i given that i 's type is θ_i and that he reported θ'_i , and second, the conditional distribution over the others' actions and types (a_{-i}, θ_{-i}) given that i 's type is θ_i and that he was asked to take action a'_i after reporting type θ'_i . This is why $\mathbb{A}_i(\cdot | \theta'_i)$ and $\mathbb{B}_i(\cdot | \theta'_i)$ in Definition 9 depend only on the reported type θ'_i and not also on the true type θ_i .

[0088] Changing the transition probabilities $\mathbb{A}_i$ or $\mathbb{B}_i$ on any measure zero sets where they are not uniquely defined does not affect whether $(\mathbb{A}, \mathbb{C})$ satisfies Definition 9. Therefore, the incentive compatibility of any $(\mathbb{A}, \mathbb{C})$ does not depend on which particular versions of the transition probabilities $\mathbb{A}_i$ and $\mathbb{B}_i$ are chosen.

15 *The Induced Single-Provider -i Model*

[0089] In this section, we show that, from the perspective of each provider i , the multi-provider model can be reduced to a Principal-Agent model involving only that single provider i . The idea is to interpret all aspects of the problem that refer to the other providers as part of provider i 's reward space, and to eliminate any correlation across types and any dependence of the signal on actions or on types by rescaling provider i 's payoff by the densities on types and signals.

[0090] The key property of the resulting "single-provider- i " model is that a mechanism is incentive compatible in the multi-provider model if and only if the mechanism that it induces for each provider i is incentive compatible in the single-provider- i model (Proposition 2). KRS established an existence result (Theorem 3) for the multi-provider model by relying on results established for the single-provider model.

[0091] Given the multi-provider model above, let us define, for any provider $i \in I$, the single-provider- i model to be a Principal-Agent model with a single provider, but where the reward space is $\mathcal{R}_i = \mathcal{R} \times \mathcal{A}_{-i} \times \Theta_{-i}$, the signal space is S , the signal is always drawn according to $\tilde{\mathbb{S}}$, the action space is $\mathcal{A}_i$, the type space is Θ_i with prior $\mathbb{Q}_i$ and, for any $(r_i, s, a_i, \theta_i) = ((r, a_{-i}, \theta_{-i}), s, a_i, \theta_i) \in \mathcal{R}_i \times S \times \mathcal{A}_i \times \Theta_i$, provider i 's utility is $\tilde{u}_i(r_i, s, a_i, \theta_i) = u_i(r, s, a, \theta) h(\theta) f(s | a, \theta)$ and the recipient's loss is $\tilde{l}(r_i, s, a_i, \theta_i) = l(r, s, a, \theta) h(\theta) f(s | a, \theta)$. Finally, for any $s \in S$, the set of feasible rewards in $\mathcal{R}_i$ is $\Phi_{i,s} = \Phi_s \times \mathcal{A}_{-i} \times \Theta_{-i}$.

- [0092] Then, a mechanism for the single-provider- i model is a pair of transition probabilities $(\mathbb{A}_i, \mathbb{C}_i)$ such that $\mathbb{A}_i: \theta_i \rightarrow \Delta(\mathcal{A}_i)$, $\mathbb{C}_i: S \times \mathcal{A}_i \times \theta_i \rightarrow \Delta(\mathcal{R}_i)$, and $\mathbb{C}_i(\phi_{i,s} \mid s, a_i, \theta_i) = 1$ for every $(s, a_i, \theta_i) \in S \times \mathcal{A}_i \times \theta_i$.
- [0093] The next proposition (Proposition 2), whose proof is in the Appendix of KRS,
5 connects the single-provider- i model and the multi-provider model.
- [0094] **Proposition 2.** *A mechanism $(\mathbb{A}, \mathbb{C})$ is incentive compatible for the multi-provider model iff for every provider i , there are transition probabilities $\mathbb{A}_i: \theta_i \rightarrow \Delta(\mathcal{A}_i)$ and $\mathbb{C}_i: S \times \mathcal{A}_i \times \theta_i \rightarrow \Delta(\mathcal{R} \times \mathcal{A}_{-i} \times \theta_{-i})$ such that $\mathbb{Q} \otimes \mathbb{A} \otimes \tilde{\mathbb{S}} \otimes \mathbb{C} = \mathbb{Q}_i \otimes \mathbb{A}_i \otimes \tilde{\mathbb{S}} \otimes \mathbb{C}_i$ and $(\mathbb{A}_i, \mathbb{C}_i)$ is incentive compatible for the single-provider- i model.*
- 10 [0095] The proof of the next result is in the Appendix of KRS. It relies on Proposition 2 and Theorem 1.
- [0096] **Theorem 3.** *Suppose that for each provider $i \in \{1, \dots, I\}$, Assumptions 1-5 hold for the single-provider- i model. Then, provided that at least one multi-provider incentive compatible mechanism exists, a mechanism that minimizes the recipient's expected losses exists for the multi-provider model.*
15
- [0097] Conditions for the existence of at least one multi-provider incentive compatible mechanism are provided by the following.
- [0098] **Proposition 3.** *Suppose that Assumptions 1, 3, and 6 hold and that there are measurable functions $\phi: S \rightarrow \mathcal{R}$ and $\xi: \theta \rightarrow [0, \infty)$, such that $\phi(s) \in \Phi_s$ for every s , $\int \xi(\theta) d\mathbb{Q}(\theta) < \infty$, and, letting $v_i(a, \theta) = \int_X u_i(\phi(s), s, a, \theta) d\mathbb{S}(s \mid a, \theta)$, suppose that
20 for each provider $i \in \{1, \dots, I\}$, $|v_i(a, \theta)| \leq \xi(\theta)$ for every a and θ , and $v_i(a, \theta)$ is continuous in a for each θ . Then at least one multi-provider incentive compatible mechanism exists.*
- [0099] OSCRA Algorithm
- 25 [0100] This may be used to determine the OSC. OSCs may include hybrid smart contracts that solve both the principal-agent problem and its underlying oracle problem in the exchange of value between the provider and the recipient. OSCs may be founded on blockchain, AI/ML technologies, and contract theory. The oracle may include a combination of off-chain AI and on-chain reputation scoring/certificates, based on
30 privacy-enhancing technologies that may guarantee strict privacy for both recipients and providers. OSCs may achieve Pareto optimality by solving the OSCRA. OSCs may therefore generate Pareto optimal incentive mechanisms between the provider and the recipient, thereby restoring the balance of contractual relationships in the initial presence of information asymmetry. The OSCRA may include modeling the

effort and the cost of effort of the provider, modeling the utility functions of the recipient and provider, and estimating the value of the reservation utility (e.g., a next best option of the provider in a competitive market). The OSC may then be the smart contract that minimizes a recipient's expected loss (e.g., maximizes the recipient's expected utility) as a function of the provider's effort and the parameters of the feasible contracts for the recipient under constraints, which may include, depending on the use cases:

- The Incentive Compatibility (IC): the provider may choose, from among all the feasible contracts that the recipient can agree to, the one that maximizes its own expected utility function;
- The Individual Rationality (IR): the provider may accept contracts only if this effort generates a utility greater than its reservation utility; or
- The Solvency Constraint (SC): the various components of the provider's capital must remain positive (e.g., financial and health capital).

[0101] These constraints may generally apply in the case of the free market. Other contracts may also be subject to other constraints (e.g., regulatory, technical, market dynamics) which must be considered in a given way for each industry. These constraints may be considered, for example, in the design and modelling of the probability spaces for types, actions, signals and rewards, and the provider's utility and recipient's loss functions.

[0102] Optimal Smart Contracts may include hybrid smart contract which solve both the Principal-Agent problem and its underlying Oracle problem in the exchange of value between a provider and a recipient in a given industry. Industry-specific Oracles based on AI and blockchain technologies may incorporate information on the exchange of value to enforce the Pareto optimality at all times between providers and recipients.

[0103] In this context, Oracle requirements may provide trusted real-world information about a product of a complex nature such as a good, a service, or a liability and the utility functions of providers and recipients in order to be able to solve a generalized Principal-Agent model as seen in the previous section.

[0104] The Optimal Smart Contract may be a hybrid smart contract that minimizes the recipient's expected loss (e.g., maximizes the recipient's expected utility) as a function of the provider's effort and the parameters of the feasible contracts for the

recipient under constraints, which are generally as follows, depending on the industry and specific uses cases:

- The Incentive Compatibility (IC): the provider chooses, from among all the feasible contracts that the recipient can agree to, the one that maximizes its own expected utility function;
- The Individual Rationality (IR): the provider accepts contracts only if this effort generates a utility greater than its reservation utility; and,
- The Solvency Constraint (SC): the various components of the provider's capital must remain positive (e.g., financial and health capital).

[0105] These constraints may generally apply in the case of the free market. Other contracts may also be subject to other constraints (regulatory, technical, market dynamics) which must be considered in a specific way for each industry. The very broad formulation of the framework proposed by Kadan, Reny, and Swinkels (2017) allows all these constraints to be considered, particularly in the design and modelling of the probability spaces for types, actions, signals and rewards, and the provider's utility and recipient's loss functions.

[0106] The Optimal Smart Contract Resolution Algorithm (OSCRA) may include an algorithm that, based on the product information and the utility functions of both provider and recipient in a given industry, solves a generalized Principal-Agent model between the provider and the recipient and generate a set of Optimal Smart Contracts to be potentially signed by both the provider and the recipient.

[0107] Optimal Smart Contracts achieve Pareto optimality by solving the Optimal Smart Contract Resolution Algorithm (OSCRA). Optimal Smart Contracts therefore generate Pareto optimal incentive mechanisms between the provider and the recipient, thereby restoring the balance of contractual relationships in the initial presence of information asymmetry. The main steps in this OSCRA Algorithm are modelling the effort and the cost of effort of the provider, modelling the utility functions of the recipient and provider, and estimating the value of the reservation utility (next best option of the provider in a competitive market). These algorithms must be designed, calibrated, tested, and validated according to the specificities and requirements of each use case. They are specific to each industry and will be the subject of specific research each time.

[0108] The parameters of the Pareto optimal contract solution of the Principal-Agent model as studied in the theory section are then encoded in smart contracts which then

bind the provider and the recipient, providing them with transparency and trust in the agreement which binds them. Optimal Smart Contracts may thus implement blockchain and AI technologies applied to contract theory.

Underlying Oracles of OSCRA and Optimal Smart Contracts

5 [0109] To solve the OSCRA algorithm and generate an Optimal Smart Contract, access to off-chain data is necessary to model all inputs and solve the Pareto optimal mechanism problem. This allows for finding the parameters of the Pareto optimal contract between the provider and the recipient. A similar Oracle problem occurs to enforce the Optimal Smart Contract and verify the alignment between the provider and recipient at all times. Specific oracles for each industry and use case are developed to achieve this. Access to neutral, impartial and quality information about the product on the different markets today is often done with the help of an expert, but the cost of accessing this information is really high and often prohibitive, which exacerbates the problem of information asymmetry linked to the Principal-Agent problem. Artificial Intelligence (AI) technology can drastically reduce the cost of accessing this information and seems to be a good candidate for helping to solve the underlying Oracle problem of the Principal-Agent problem. The potential use of AI technology and other mechanisms enable the acquisition of knowledge about the transferred product, whether it be goods, services, risks, or debts, as well as the components of the provider and recipient's utility function. During each product transfer iteration, the provider's reputation and certificate rating is calculated, updated, and stored on-chain to prevent corruption. The Oracle would then be a combination of off-chain calculation using AI and on-chain reputation/certificate scoring. It should be based on privacy-enhancing technologies that ensure strict confidentiality for both recipients and providers. These desired features of the Oracle will be specific to each industry and will be the subject of specific research each time.

[0110] Pareto-Optimal DApps or Optimal DApps may include a collection of industry specific Decentralized Applications (DApps) which generate Optimal Smart Contracts between providers and recipients using the OSCRA algorithm and enable providers and recipients to sign and commit to a given Optimal Smart Contract.

[0111] Optimal DApps may be implemented in various sectors of activity. Each Optimal DApp may provide a marketplace for recipients and providers to enter into Optimal Smart Contracts. At first, use case-specific systems are designed that induce the greatest possible transparency on the characteristics and behavior of the provider.

Next, the Oracle solves the OSCRA Algorithm. By observing a fairly accurate approximation of the provider's real effort, strong incentives are created on-chain that make it costly to lose reputation because of adverse behavior. In addition to the reputation, on-chain financial retention mechanisms at the start of the contract, reward afterwards providers who have made the most effort and therefore punish (or reward less) providers who have made the least effort. The Oracle also helps the provider to deliver better services/goods or reduce the frequency and intensity of the risk.

[0112] FIG. 1 illustrates an example method 100 illustrating a model based on the Optimal Smart Contract Resolution Algorithm, according to one or more implementations herein.

[0113] The model presented above includes a parametric function optimization model. The study was carried out using version 3.4.0 of R open-source statistical analysis software (R Foundation for Statistical Computing, Vienna, Austria; <http://www.R-project.org/>). The software provided:

- Management of a given database (data processing and analysis);
- Statistical analysis to estimate probability distributions from data;
- Formal computation (or symbolic computation) to allow the reduction of complex expressions, including integral computation and optimization computation;
- Numerical analysis when expressions cannot be evaluated symbolically.

[0114] Blockchain technology is also used to encode the optimal smart contracts after resolution of the model.

[0115] The Optimal Smart Contract Resolution Algorithm includes determining the characteristics of the contract that minimize the expected loss (equivalent to maximize the expected utility) of the recipient by maximizing the expected utility of the provider. It is the characteristics of the contract that are most advantageous to the recipient and for which the provider reveals its type and selects the action that best suits him under the conditions of this contract, while checking that this contract remains sufficiently attractive for the provider. We will then present the corresponding optimal smart contract, the solution of the algorithm. We recall that the expression of the expected utility $U(\mathbb{A}, \mathbb{C})$ of the provider in the single-provider model is (Definition 2):

$$U(\mathbb{A}, \mathbb{C}) \equiv \int_{\mathcal{R} \times \mathcal{S} \times \mathcal{A} \times \Theta} u(r, s, a, \theta) d\mathbb{C}_{s,a,\theta}(r) d\mathbb{S}_{a,\theta}(s) d\mathbb{A}_\theta(a) d\mathbb{Q}(\theta),$$

[0116] In the multi-provider model, with $i \in \{1, \dots, I\}$, the expected utility $U_i(\mathbb{A}, \mathbb{C})$ of provider i is (Definition 10):

$$U_i(\mathbb{A}, \mathbb{C}) \equiv \int_{\mathcal{R} \times \mathcal{S} \times \mathcal{A} \times \Theta_{-i} \times \Theta_i} u_i(r, s, a, \theta_i, \theta_{-i}) h(\theta_i, \theta_{-i}) \\ \times f(s \mid a, \theta_i, \theta_{-i}) d\mathbb{C}(r \mid s, a, \theta_i, \theta_{-i}) d\tilde{\mathbb{S}}(s) d\mathbb{B}_i(a_{-i}, \theta_{-i} \mid a_i, \theta_i) d\mathbb{A}_i(a_i \\ \mid \theta_i) d\mathbb{Q}_i(\theta_i)$$

[0117] In both case, the expression of the expected loss $L(\mathbb{A}, \mathbb{C})$ of the recipient ($j \in \{1, \dots, J\}$) is (Definition 3):

$$L(\mathbb{A}, \mathbb{C}) \equiv \int_{\mathcal{R} \times \mathcal{S} \times \mathcal{A} \times \Theta} l(r, s, a, \theta) d\mathbb{C}_{s,a,\theta}(r) d\mathbb{S}_{a,\theta}(s) d\mathbb{A}_\theta(a) d\mathbb{Q}(\theta),$$

10 *Preparation of the OSCRA Algorithm*

[0118] At 101, the process may start and at 105, The algorithm may receive an input dataset.

[0119] At 110, the Optimal Smart Contract Resolution Algorithm may be prepared, comprising the steps 111 to 114:

15 [0120] First, at 111, the components of the Principal-Agent problem (Optimal Smart Contract Model) may be designed:

- Design of the reward function, whether the contract is deterministic (e.g., function from the signal space into a deterministic reward) or randomized (e.g., function from the signal space to a probability distribution over rewards);
- 20 • Design of the structure of the types, actions, signals (and rewards if the contract is randomized over the rewards) probability spaces (complexity depends on the use case and the available data, in other words, the sample spaces associated with these probability spaces can be finite sets, real numbers, vector spaces, function spaces, etc.);
- 25 • Design of the types Q , actions A , signals S (and rewards R if the contract is randomised over the rewards) random variables (can be discrete or real-valued random variables, random vectors, stochastic processes, random functions or random fields, etc.) and parametrisation with types, actions, signals, and rewards indicators;

- Design and parametrization of the utility function of the provider (including cost of effort function);
- Design of the loss function of the recipient;
- Design of the constraints of the Principal-Agent problem (Incentive Compatibility, Individual Rationality including reservation utility design, Solvency Constraint, etc.).

[0121] Second, at 112, data mining with the available data may be implemented, while respecting the privacy of our members and partners. Data mining included selecting and making the raw data reliable, then synthesizing them and creating new indicators. The other part of the model inputs is produced using statistical methods known as “data analysis”, such as data reduction. Predictive models often assume linear independence: the explanatory variables should not be collinear. The aim is to reduce the dimensions of the problem:

- Synthesis of the required inputs / data for the calibration of the Principal-Agent model;
- Build homogeneous classes of providers in regard to potential types and actions; and
- Perform the segmentation based on potential types and actions.

[0122] Third, at 113, indicators may be constructed:

- Type indicators which measure the characteristics of the product (hidden information for the recipient). Types indicators with low values reveal a high probability of adverse selection within the recipient-provider relationship. The higher the value of types of indicators, the more it increases the expected utility of the recipient and therefore reduces his expected loss.
- Action indicators which measure the behavior of the provider (hidden action for the recipient). Action indicators with low values reveal a high probability of moral hazard within the recipient-provider relationship. The higher the value of action indicators, the more it increases the expected utility of the recipient and therefore reduces his expected loss. Action indicators may depend stochastically on the type indicators, depending on the use case.
- Signal indicators which measure the provider’s success in fulfilling the contract (the output / outcome). Signal indicators are generally stochastically dependent on type and action indicators.

- Rewards indicators which measure the reward given by the recipient to the service provider in exchange for the performance of the contract, and which therefore depend directly on the signal. Generally positive for the delivery of goods and services in the case of an exchange of the transfer of responsibilities from the provider to the recipient (such as risk or debt), it is negative (one or more positive cash settlements are made from the provider to the recipient to settle the contract).

[0123] Fourth, at 114, model calibration is provided by the construction of indicators, which enables the estimation of probability distribution of random variables designed and parametrized in the design phase:

- Estimation of the marginal probability distribution of the types Q , actions A , signals S (and rewards R if the contract is randomized over the rewards) random variables on each segment. There is information asymmetry on the distributions of types and actions, which are known and certain for the provider, type and action indicators are random variables for the recipient. However, there is no information asymmetry on the signal, which derive from types and actions: we may assume in some cases that the recipient's information on the signal random variable coincides with the provider's information;
- Estimation of the (potentially parametric) copulas associated with the joint probability distribution (C, S, A, Q) of the types Q , actions A , signals S (and rewards R if the contract is randomised over the rewards) random variables on each segment;
- Estimation of the joint probability distribution (C, S, A, Q) and conditional probability distributions $C|(S, A, Q)$, $S|(A, Q)$, and $A|Q$.
- Calibration of other component of the Principal-Agent model (especially utility function of the providers and loss function of the recipients).

Initialization of the OSCRA Algorithm

[0124] Once the components of the model have been designed, parametrized and implemented, the OSCRA algorithm may be initialized at 115 for a new recipient (j) becoming a member of the DAO.

- [0125] Before using the ecosystem, the recipient (j) concluded an initial contract $\mathbb{C}_{a_{i_0}, \theta_{i_0}}^0$ with provider i_0 its characteristics may be learned when the new member (j) is onboarded.
- [0126] The initial mechanism $\mathbb{A}^0, \mathbb{C}^0$) was feasible and satisfied *Incentive Compatibility* (IC), *Individual Rationality* (IR), and *Other Constraints* (OC) by construction (Definition 7). Indeed, by signing (e.g., executing the smart contract, e-signing the contract, etc.) the initial contract,
- The provider i_0 chose the action a_{i_0} that best suits him in terms of initial contract, so verified *Incentive Compatibility* (IC) in Definition 9;
 - The provider i_0 verified *Individual Rationality* (IR) in Definition 5 because he has not chosen another contract or option (reservation utility) but the condition of this initial contract; and
 - The provider i_0 and the recipient verified *Other Constraints* (OC) in Definition 6 related to solvency, regulatory, technical, market dynamics, etc.
- [0127] This can allow the following data to be estimated, in particular:
- The type θ_{i_0} and action a_{i_0} indicators and the utility function u_{i_0} and reservation utility $\underline{U}_{i_0}$ of the provider i_0 ; and
 - The loss function l of the recipient.
- [0128] The initialization of the algorithm may include the estimation (of the initial type $\widehat{\theta}_{i_0}$ and action $\widehat{a}_{i_0}$, utility function $\widehat{u}_{i_0}$ and reservation utility $\widehat{\underline{U}}_{i_0}$ of the provider i_0 , and loss function $\widehat{l}$ of the recipient) in the initial mechanism:
- Estimation of the initial values of type $\widehat{\theta}_{i_0}$ and action $\widehat{a}_{i_0}$;
 - Calibration of the parameters of the initial contract $\mathbb{C}_{\widehat{a}_{i_0}, \widehat{\theta}_{i_0}}^0$.
 - Calibration of the parameters of the utility function $\widehat{u}_{i_0}$ according to the risk-aversion of provider i_0 (including cost of effort function $\widehat{c}$). The degree of risk-aversion and effort aversion of a provider are intrinsic characteristics that do not depend on the contract offered to the provider: the utility function and the disutility function of effort are also those found in the initial contract;
 - Calibration of the loss function $\widehat{l}$ of the recipient;

- Calibration of the constraints of the Principal-Agent problem (Incentive Compatibility, Individual Rationality including reservation utility design, Solvency Constraint, etc.).

5 [0129] These estimates give the algorithm information / input dataset 105 for the next recurring data preparation / recalibration phase.

[0130] Once the algorithm has been initialized, the parameters of the expected utility functions and the cost of effort are calibrated regardless of the contract between the provider and the recipient.

Resolution of the Principal-Agent Problem

10 [0131] At 120, the Principal-Agent problem of Definition 8 (Optimal Smart Contract Model) may be solved.

[0132] This is a constrained non-linear optimization problem, where the resolution of the constraints (the incentive constraint IC (Definition 9), participation constraint IR (Definition 5) and other constraint OC (Definition 6)) is itself an constrained non-linear optimization problem.

15 [0133] The provider's program is to determine the $(\tilde{A}, \tilde{C})$ such that:

$$\max_{(\tilde{A}, \tilde{C}) \in \mathcal{M}_{IR} \cap \mathcal{M}_{OC}} U(\tilde{A}, \tilde{C}).$$

[0134] The solutions of the provider's program are the set of feasible mechanisms $\mathcal{M}$ with $\mathcal{M} = \mathcal{M}_{IC} \cap \mathcal{M}_{IR} \cap \mathcal{M}_{OC}$ on $\mathcal{R} \times \mathcal{S} \times \mathcal{A} \times \Theta$

20 [0135] The recipient's program is to determine the $(\tilde{A}, \tilde{C})$ such that:

$$\min_{(\tilde{A}, \tilde{C}) \in \mathcal{M}} L(\tilde{A}, \tilde{C}).$$

[0136] We may solve for example the recipient's program by using the Lagrangian augmented method for provider i . They correspond to the rewards structure that minimises the recipient's loss while encouraging the provider i to reveal his true type and make the recommended action. The convergence of the optimisation calculation completes the model resolution algorithm, solves the optimal smart contract, and thus validate the algorithm.

25 [0137] At 125, an output dataset of the Optimal Smart Contract Model may be produced.

[0138] The parameters may then be encoded into smart contracts.

30 [0139] At 130, the status of whether the provider is a new provider may be evaluated.

- If so, at 140, the provider can be offered all the Optimal Smart Contracts to choose from.
 - Otherwise, if the provider already has a contract, at 135, whether the current contract belongs to the set of optimal smart contracts may be assessed. If so, at 150 the renewal date of the contract and new data to restart the algorithm may be awaited.
 - Otherwise, at 140, the provider can be offered all the Optimal Smart Contracts to choose from.
- At 140, the recipient may be offered to subscribe to one of the optimal smart contracts,
 - If at 145 he accepts and signs, at 150, the renewal date of the contract and new data to restart the algorithm may be awaited.
 - Otherwise, at 160, the process ends.

[0140] DAOs, as member-owned communities without centralized management, may present a structure for solving the major economic and organizational governance problems. Thus, OSCRAs may be delivered via a DAO configured to facilitate Pareto optimality in exchanges of value across industries. DApps configured to generate OSCs as disclosed herein may be implemented using one or more DAOs. One or more tokens may facilitate related to the DAO:

- A usage token, which may include a non-transferable and non-burnable non-fungible token (NFT), may be configured as a usage indicator and confer specific rights to the recipients as DAO members who own it. For example, each year of usage of at least one DApp in the ecosystem, a member may be allocated one such token. The supply of this token may therefore be uncapped and strictly non-decreasing.
- A board token, which may include a transferable and non-burnable NFT token allocated to board members of the DAO board to vote on matters related to an agile execution of DAO operational decisions and protection in the event of events. Its supply may be constant. It may initially be allocated to the founder and early advisors. It may be transferable by a vote of the governance, motivated by the reputation of the board members of the DAO and designed to meet the best compromise between agility and decentralization for each stage of the project.

- A governance token, which may include a non-transferable and burnable voting NFT, which may be used to vote, for example, on DAO strategic decisions. Each usage token associated with an active DAO member and each board token may give access to a governance token. The supply therefore may be equal at the initiation of the DAO to the supply of board tokens to allow agile development of the DAO. Over time, as governance tokens are distributed through usage, the weight of the board in the DAO's strategic decisions may become insignificant compared to that of the DAO members. This transition enables a reduction in the operational risk at the start of the project by enabling the project to be scaled up in an agile but centralized way, to become fully decentralized in the long run.
- An economic token, which may be configured, for example, based on an existing cryptocurrency such as Bitcoin.

[0141] FIG. 2 illustrates a system 200 for generating and implementing OSCs, according to one or more implementations herein.

[0142] Each DApp 210 associated with a DAO 202 may provide a marketplace for DAO members, such as a recipient 222 and DAO partners such as a provider 224 to agree on OSCs such as an OSC 226. Use case-specific systems may provide for transparency on the characteristics and behavior of the provider 224 and a provider signal 224a. Next, an oracle 230 may solve an OSCRA 228 to provide OSC terms 226a using off-chain data at 232. Signaling from the recipient 222 may be included also at 222a. By observing a fairly accurate approximation of the provider 224's real effort, strong incentives may be created on-chain that make it costly for the provider 224 to lose reputation, for example, because of adverse behavior. In addition to the reputation, on-chain financial retention mechanisms at the start of the contract, may at 224b reward the provider 224 afterwards if the provider 224 has made the most effort or punish (or reward less) the provider 224 if the provider 224 has made the least effort. The oracle 230 may also help the provider 224 at 226b deliver better services/goods or reduce the frequency and intensity of the risk. The oracle 230 may also judge quality and can help settle any disputes (DAO governance can settle in the last instance in the event of an appeal by the recipient or provider). The recipient 222 may transact with the provider 224 at 226c using economic tokens scoped to the DAO 202 or scoped to the DApp 210. The provider 224 or the recipient 222 may at 224d and 222d, respectively, purchase or sell economic tokens or other tokens from the DAO 202. The recipient 222 may pay for its

membership in the DAO 202, for example, using a token such as a stablecoin or an economic token, or a combinations thereof, at 222b.

- [0143] For each DApp, the initial conditions of the mechanism in place (e.g., corresponding to the existing probability measures tuple $(\mathbb{A}^0, \mathbb{C}^0)$, when the ecosystem is not used), which induce an initial loss L^0 for the recipient $\tilde{\mathbb{A}}, \tilde{\mathbb{C}}$ may be studied:

$$L^0 = L(\mathbb{A}^0, \mathbb{C}^0) \equiv \int_{\mathcal{R} \times \mathcal{S} \times \mathcal{A} \times \Theta} l(r, s, a, \theta) d\mathbb{C}_{s,a,\theta}^0(r) d\mathbb{S}_{a,\theta}(s) d\mathbb{A}_\theta^0(a) d\mathbb{Q}(\theta),$$

- [0144] Then, the OSC may be solved for that minimizes the recipient's expected loss:

$$L_{min} = \min_{(\tilde{\mathbb{A}}, \tilde{\mathbb{C}}) \in \mathcal{M}} L(\tilde{\mathbb{A}}, \tilde{\mathbb{C}}).$$

- [0145] In this way, we have created value G for the recipient, which may represent a difference between the initial loss L_0 and the minimum loss generated by the OSC L_{min} : $G = L_0 - L_{min}$. G may have a positive or zero value: zero in the case where the contract was already Pareto optimal and positive in the case where the contract was Pareto sub-optimal in the initial situation.

- [0146] The protocol may be rewarded with a ratio α of the created value G . Also, the most efficient providers may be rewarded by redistributing part of this created value with a rate β . The value capture for the protocol may be $\alpha \cdot G = \alpha \cdot (L_0 - L_{min})$. The total reward for providers may be $\beta \cdot G = \beta \cdot (L_0 - L_{min})$.

- [0147] A recipient, member of a DApp according to implementations herein can subsequently be a provider of the product he has received and then transformed or developed as part of another OSC with other recipients, in the same DApp or another one. For example:

- A buyer of intermediate goods may be a recipient and can sell the final product, and he may become a goods provider in another optimal smart contract.
- The client of an outsourcing service may be a recipient and may become a service provider in another optimal smart contract.
- The buyer of a property may be a recipient and may become a debt provider in another DApp.
- The client of a risk prevention and repair service provider may be a recipient and may become a risk provider in another DApp.

- [0148] Zero knowledge proof identification certifiers may be used to verify the humanity and uniqueness of DAO members to counter, for example, Sybil attacks. There may be no way of recovering the identity of DAO members except by a massive vote by the DAO governance in the event of abuse of the system. The governance of the DAO may be configured as fully decentralized using a governance protocol, for example, by controlling the treasury of economic tokens, and older DAO members may receive more voting power. Trust may thus be a function of time.
- [0149] In some implementations, DAO members may receive one usage token, a non-transferable NFT for every year they use at least one DApp. When a DAO member actively uses at least one DApp, the DAO member may receive one governance token, a non-transferable voting NFT for each usage token they have collected. Voting power may thus grow with time, rather than by buying more tokens. This may thus prevent a hostile takeover of power by buying economic tokens or the vote-buying phenomena exhibited in some other crypto-asset protocols.
- [0150] Governance can also be distributed at the level of each DApps by introducing DApp-specific governance tokens to handle feature requests, for example. They also resolve voting fatigue if DAO members were to have to vote, for example in a DAO member vote, for DApps that did not concern them.
- [0151] The full decentralized governance of the DAO may be thus under the total control of real humans, who can decide on the strategic direction of DApp development, the scope, the values (e.g., decentralization, privacy, justice), and on the handling of externalities (total privacy and ownership of data for DAO members, impact on the environment, and more). The development of DApps may rely heavily on artificial intelligence technologies (machine learning, deep learning) and have the potential to challenge the market of conventional, or “web2,” AI applications. These centralized applications (CApps) may meet the need for efficiency in the production of value for their end-users, but may also be naturally against their privacy, which may be an essential externality. Implementations of DAO herein may leverage fully decentralized governance under the total control of real humans, while keeping values intact, thereby guaranteeing safe AI applications to bring Pareto optimality to all value exchanges as a service to humans.
- [0152] In addition to the usage token, the board token and the governance token, which are all non-economic tokens, an economic token may be issued. Aspects of the economic token and its minting process will now be discussed.

- [0153] The supply side may comprise a minting supply of up to 21 million economic tokens. The majority of the minting supply may usage-based with 11 million economic tokens, minted progressively in a linear relationship with adoption, and the last economic token will be minted—as far as usage is concerned—when the total usage duration of the ecosystem has exceeded 11 billion years. This may be equivalent to an emission of 11 billion usage tokens, that is, if 1 billion DAO members have used at least one DApp of the ecosystem for an average of 11 years). The second portion of the minting process may be time-based: 10 million economic tokens may be minted following a geometric series with a common ratio of 4/5 for one-year cycles, starting on the date of the DAO's introduction. The minting function may therefore be two-dimensional.
- [0154] FIG. 3 illustrates a plot 300 of the minting function, according to one or more implementations herein. In FIG. 3, axis 310 corresponds to the total minted supply of economic tokens, axis 320 corresponds to usage, and axis 330 corresponds to time (measured in increments of 5 years).
- [0155] The supply side may further comprise minting allocation aspect. Half of the allocation, whether issued on a usage or time basis (10.5 million economic tokens), may go to the DAO's community treasury, which may own a large proportion of these minted tokens. The other half of the tokens issued by usage (5.5 million economic tokens) may be distributed to reward the community, for example to incentivize DAO members or providers to develop marketplaces where optimal smart contracts can be concluded, and also to DAO members to exercise governance of the DAO. The other half of the tokens issued by time (5 million economic tokens) may be used to reward stakeholders such as contributors, intellectual property, long-term investors and audits.
- [0156] With respect to circulation, part of the minting supply may be sold dynamically directly (without vesting) to DAO members, providers and investors against stablecoins to enable the financing of the DAO and to scale the development of DApps for all the different (industry) use cases.
- [0157] Half of the vesting of the rewards to contributors, intellectual property, and long-term investors may be based linearly over time, for example, up to 10 years, and the other half may be released linearly between current usage duration and the objective of multiplying the usage duration by 10. This may create an incentive for contributors, researchers, and long-term investors to take actions that are aligned with the DAO's objectives of scaling up worldwide.

[0158] Usage may be the main driver of demand, as this token may enable interaction with all DApps. Such a regime may provide a unique value proposition, the marginal capture of the total value created combined with the potentially very high growth rate, the rising demand driven by usage of the DAO, together with a slowly increasing supply and the absence of pre-minted tokens, could result in significant upside for early token adopters, especially for DAO members, providers and investors, and prove very attractive and secure for token holders.

[0159] Equilibrium between supply and demand, after decades of potentially strong increases in the value of economic tokens (in other words, demand for economic tokens will exceed supply), should be achieved in the long term thanks to the following elements:

- The DAO's community treasury may make part of its economic token reserves available to the DAO's liquidity pool. In some implementations, the minting process may further provide economic tokens directly to the liquidity pool.
- The DAO's liquidity pool may allow the buying and selling of economic tokens against a stablecoin, which may enable collateralization.
- Bonding curves may be used, which may be a type of automated market maker (AMM). Such bonding curves may include a mathematical function of the price as a function of the number of economic tokens in circulation and the number of economic tokens and stablecoins available in the liquidity pool.
- There may thus always a positive spread between the sell and buy bonding curve, which avoids arbitrage against the DAO.
- The bonding curve may thus provide stability between supply and demand.

[0160] FIG. 4 illustrates an operational environment 400, according to one or more of the implementations herein. As illustrated in FIG. 4, the operational environment 400 may include actors, including a user (recipient or provider) device 410, a network 420, a node 430 having at least a computing resource, such as a processor 432, and an electronic storage device 434, and a remote device 440.

[0161] The user (recipient or provider) device 410 may include any variety of devices a user (recipient or provider) may use to interface with the node 430 via the network 420, including, for example, a server, a desktop computer, a laptop computer, a handheld computer, a tablet computing platform, a Netbook, a Smartphone, a gaming console, and/or other computing platforms.

- [0162] The network 420 may include any variety of devices configured to enable a device to communicate with other devices, such as via a wired connection and/or a wireless connection, for example, via the internet and/or other networks using, for example, TCP/IP or cellular hardware enabling wired or wireless (e.g., cellular, 2G, 3G, 4G, 4G LTE, 5G, or wireless local area network) communication. For example, the network 420 may include, for example, a receiver, a transmitter, a transceiver, a modem, a network interface card, and/or an antenna.
- [0163] The node 430 may include any variety of devices configurable to perform the implementations and methods disclosed herein and interface with the user (recipient or provider) device 410 via the network 420, including, for example, a server, a desktop computer, a laptop computer, a handheld computer, a tablet computing platform, a Netbook, a Smartphone, a gaming console, and/or other computing platforms.
- [0164] The node 430 may include the processor 432. The processor 432 may include, for example, one or more processor(s) configured to execute machine-readable instructions for implementing all or some of the implementations herein. The processor 432 may be configured to access the electronic storage device 434 to retrieve and/or write electronic data from and to the electronic storage device 434.
- [0165] The node 430 may include the electronic storage device 434. The electronic storage device 434 may be configured to electronically store data (e.g., host) corresponding to one or more databases or other forms of data storage for use in implementations herein. The electronic storage device 434 may be accessible by the processor 432.
- [0166] The remote device 440 may include any variety of devices a user (recipient or provider) may use to interface with the node 430 via the network 420, including, for example, a server, a desktop computer, a laptop computer, a handheld computer, a tablet computing platform, a Netbook, a Smartphone, a gaming console, and/or other computing platforms.
- [0167] One or more further nodes, for example, a node 450, may be in electronic communication with one or more of the user (recipient or provider) device 410, the remote device 440, and the node 430 via the network 420. The node 450 may have similar aspects and capabilities to, for example, the node 430, that is, the node 450 may also store some or all of the blockchain on a database thereon to form a decentralized network comprising at least the node 430 and the node 450.

- [0168] In an example, if the user (recipient or provider) of the user (recipient or provider) device 410 and the user (recipient or provider) of the remote device 440 seek to establish an OSC between them, they may engage the network, initially through the node 430. It will be understood that either or both of the user (recipient or provider) device 410 or the remote device 440 may also be a node of the decentralized network, or they may interface with a node (as is presented in this example).
- [0169] The processor 432 may thus receive a set of mechanisms. Each mechanism may be defined by at least a provider action and a feasible contract. The processor 432 may define a recipient loss function based on the set of mechanisms. The processor 432 may determine a mechanism of the set of mechanisms that minimizes of a recipient loss function. The processor 432 may generate a smart contract based on the mechanism. The processor 432 may deploy the smart contract to a blockchain maintained across a plurality of nodes (e.g., including the node 430 and the node 450) connected via a computer network 420 by broadcasting the smart contract to the plurality of nodes via the network interface over the network 420. The processor 432 may receive a transaction address from the blockchain corresponding to the smart contract. The processor 432 may store the transaction address in the database of the electronic storage device 434 and/or the recipient device 410.
- [0170] In some implementations, the processor 432 or a processor of the recipient device may be further configured to verify the smart contract by transmitting to the blockchain a request for data in response to the transaction address.
- [0171] The processor 432 may be further configured to receive oracle data. Each mechanism may be further defined by the oracle data. The oracle data may be generated by, for example, transmitting the prompt to a trained machine learning model and receiving outputs of the trained machine learning model as the oracle data. Receiving outputs of the trained machine learning model may include performing a data operation such as a transformation, a filtering, a sorting, another data operation, or combinations thereof, on raw output data of the trained machine learning model to yield the oracle data. The oracle data may be configured by a prompt received from a provider node or a recipient node.
- [0172] In some implementations, transactions corresponding to an OSC may require the transaction of a token, for example, by the processor, to the blockchain. Such a token may be a non-fungible or non-burnable token.

- [0173] Each mechanism may be further defined by a provider type, a provider signal, and a provider reward.
- [0174] Each feasible contract may be further defined by a provider signal and the provider action of the mechanism. Each feasible contract may be further defined by a provider reward. Such a provider reward may be selected by a user (e.g., a recipient or a provider) from a set of possible provider rewards. The set of possible provider rewards may be selected based on the provider signal.
- [0175] Each feasible contract may further be constrained by one or more of an incentive compatibility constraint, an individual rationality constraint, or a solvency constraint.
- [0176] The processor 432 may be further configured to determine provider validation data and deploy the provider validation data to the blockchain. The provider validation data may include, for example, a provider reputation or a certificate rating. The processor 432 may retrieve from the blockchain available provider validation data, and each mechanism may be further defined by the provider validation data.
- [0177] While functions herein are described as being performed by hardware of the node 430 (e.g., the processor 432 and the electronic storage device 434), it will be understood that any of or any combination of the hardware of the various devices (e.g., the node 430, the node 450, the user (recipient or provider) device 410, or the remote device 440) may be used to implement the above.
- [0178] FIG. 5 is a diagram of example components of a device 500, according to one or more implementations herein. The device 500 may correspond to one or more device, network, resource, or service of FIG. 4. In some implementations, one or more device, network, resource, or service of FIG. 4 may include one or more of the devices 500 and/or one or more components of the device 500, for example, according to a client/server architecture, a peer-to-peer architecture, and/or other architectures, which may include a plurality of hardware, software, and/or firmware components operating together to provide the functionality attributed herein to the device 500. In some implementations, the device 500 may include a distributed computing architecture (e.g., one or more individual computing platforms operating in concert to accomplish a computing task). For example, the device 500 may be implemented by a cloud of computing platforms operating together as the device 500. By way of non-limiting example, a given device 500 may include one or more of a server, a desktop computer, a laptop computer, a handheld computer, a tablet computing platform, a Netbook, a Smartphone, a gaming console, and/or other computing platforms.

- [0179] As used herein, the term “component” is intended to be broadly construed as hardware, firmware, or a combination of hardware and software. It will be apparent that systems and/or methods described herein may be implemented in different forms of hardware, firmware, and/or a combination of hardware and software. The actual specialized control hardware or software code used to implement these systems and/or methods is not limiting of the implementations. Thus, the operation and behavior of the systems and/or methods are described herein without reference to specific software code—it being understood that software and hardware can be used to implement the systems and/or methods based on the description herein.
- [0180] The device 500 may include a bus 510, a processor 520, a memory 530, a storage component 540, an input component 550, an output component 560, and a communication component 570.
- [0181] The bus 510 includes a component that enables wired and/or wireless communication among the components of device 500. The bus 510 may enable various components of a computer system to communicate with each other, allowing for the transfer of data from one part to another.
- [0182] The processor 520 may include a central processing unit, a graphics processing unit, a microprocessor, a controller, a microcontroller, a digital signal processor, a field-programmable gate array (FPGA), an application-specific integrated circuit, and/or another type of processing component. The processor 520 may be implemented in hardware, firmware, or a combination of hardware and software. In some implementations, the processor 520 may include one or more processors capable of being programmed to perform a function. Such processors may or may not be all integral to the same physical device and may in some embodiments be distributed among several devices.
- [0183] The processor 520 may be configured to execute one or more of the modules disclosed herein, and/or other modules by software; hardware; firmware; some combination of software, hardware, and/or firmware; and/or other mechanisms for configuring processing capabilities on the processor 520. As used herein, the term “module” may refer to any component or set of components that perform the functionality attributed to the module. This may include one or more physical processors during execution of processor readable instructions, the processor readable instructions, circuitry, hardware, storage media, or any other components. Various modules or portions thereof may be implemented in any of various ways, including procedure-based techniques, component-based techniques, and/or

object-oriented techniques, among others. For example, the program instructions may be implemented using system libraries, language libraries, model-view-controller (MVC) principles, application programming interfaces (APIs), large language models (LLMs), system-specific programming languages and principles, cross-platform programming languages and principles, pre-compiled programming languages, markup programming languages, stylesheet languages, “bytecode” programming languages, object-oriented programming principles or languages, other programming principles or languages, C, C++, C#, Java, JavaScript, Python, PHP, HTML, CSS, TypeScript, R, Elm, Unity, VB.Net, Visual Basic, Swift, Objective-C, Perl, Ruby, Go, SQL, Haskell, Scala, Arduino, assembly language, Microsoft Foundation Classes (MFC), Streaming SIMD Extension (SSE), or other technologies or methodologies, as desired.

[0184] It should be appreciated that although some modules disclosed herein may be illustrated for example as being implemented within a single processing unit, in embodiments in which the processor 520 includes multiple processing units, one or more of modules disclosed herein may be implemented remotely from the other modules. The description of the functionality provided by the different modules disclosed herein is for illustrative purposes, and is not intended to be limiting, as any of modules described herein may provide more or less functionality than is described. For example, one or more of modules disclosed herein may be eliminated, and some or all of its functionality may be provided by other ones of modules disclosed herein. As another example, the processor 520 may be configured to execute one or more additional modules that may perform some or all of the functionality attributed herein to one of modules disclosed herein.

25 [0185] The memory 530 may include a random-access memory, a read only memory, and/or another type of memory (e.g., a flash memory, a magnetic memory, and/or an optical memory).

[0186] The electronic storage component 540 may store information and/or software related to the operation of the device 500. For example, the electronic storage component 540 may include a solid-state disk drive, a hard disk drive, a magnetic disk drive, an optical disk drive, a compact disc, a digital versatile disc, and/or another type of non-transitory computer-readable medium. Implementations of the electronic storage component 540 may include one or more of optically readable storage media (e.g., optical disks, etc.), magnetically readable storage media (e.g., magnetic tape, magnetic hard drive, floppy drive, etc.), electrical charge-based storage media (e.g., EEPROM, RAM, etc.), solid-state storage media (e.g., flash

drive, etc.), and/or other electronically readable storage media. Implementations of the electronic storage component 540 may include one or both of system storage provided integrally (*i.e.*, substantially non-removable) to the device 500 and/or removable storage that is removably connectable to the device 500 via, for example, a port (*e.g.*, a serial port, a USB port, an IEEE 1394 port, a THUNDERBOLT™ port, etc.) or a drive (*e.g.*, disk drive, flash drive, or solid-state drive etc.). The electronic storage component 540 may also or alternatively include one or more virtual storage resources (*e.g.*, cloud storage, a virtual private network, and/or other virtual storage resources). An electronic storage may store software algorithms, information determined by one or more processors, information received from one or more computing platforms, information received from one or more remote platforms, databases (*e.g.*, structured query language (SQL) databases (*e.g.*, MYSQL®, MARIADB®, MONGODB®), NO-SQL databases, among others), data files, compiled data, analyzed data, charts, tables, videos, images, presentations, and 3D content in the respective format and/or other information enabling a computing platform to function as described herein.

[0187] The electronic storage component 540 may host a distributed ledger, for example, a blockchain. A blockchain can be used to maintain a continuously growing list of records called blocks. A blockchain can be used to maintain a record of transactions or events between parties in a way that is difficult to falsify. Each block in a blockchain may include several records as well as a hash of previous blocks in the blockchain. If a record in a previous block is changed, the hash may be disrupted in any following blocks. The result is that a system can easily detect whether the fidelity of data has been preserved. Similarly, to falsify a given record, a malicious actor would be required to falsify that record and all subsequent records so that the hashes end up the same. This is extremely difficult in practice. Additionally, a blockchain may be distributed among many entities. Any changes to the blockchain may be verified by comparing it to the numerous individual records.

[0188] The input component 550 may enable the device 500 to receive input, such as user (recipient or provider) input and/or sensed inputs. For example, the input component 550 may include a touch screen, a keyboard, a keypad, a mouse, a button, a microphone, a switch, a sensor (internal and/or external), a global positioning system component, an accelerometer, a gyroscope, and/or an actuator.

[0189] The output component 560 may enable the device 500 to provide output, such as via a display, a speaker, and/or one or more light-emitting diodes.

- [0190] The communication component 570 may enable the device 500 to communicate with other devices, such as via a wired connection and/or a wireless connection, for example, via the internet and/or other networks using, for example, TCP/IP or cellular hardware enabling wired or wireless (*e.g.*, cellular, 2G, 3G, 4G, 4G LTE, 5G, wireless local area network, near field communication (NFC), BLUETOOTH®) communication. For example, the communication component 570 may include a receiver, a transmitter, a transceiver, a modem, a network interface card, and/or an antenna.
- [0191] As used herein, “internet” may include an interconnected network of systems and a suite of protocols for the end-to-end transfer of data therebetween. A model describing may be the Transport Control Protocol and Internet Protocol (TCP/IP), which may also be referred to as the internet protocol suite. TCP/IP provides a model of four layers of abstraction: an application layer, a transport layer, an internet layer, and a link layer. The link layer may include hosts accessible without traversing a router, and thus may be determined by the configuration of the network (*e.g.*, a hardware network implementation, a local area network, a virtual private network, or a networking tunnel). The link layer may be used to move packets of data between the internet layer interfaces of different hosts on the same link. The link layer may interface with hardware for end-to-end transmission of data. The internet layer may include the exchange of datagrams across network boundaries (*e.g.*, from a source network to a destination network), which may be referred to as routing, and is performed using host addressing and identification over an internet protocol (IP) addressing system (*e.g.*, IPv4, IPv6). A datagram may include a self-contained, independent, basic unit of data, including a header (*e.g.*, including a source address, a destination address, and a type) and a payload (*e.g.*, the data to be transported), to be transferred across a packet-switched network. The transport layer may utilize the user (recipient or provider) datagram protocol (UDP) to provide for basic data channels (*e.g.*, via network ports) usable by applications for data exchange by establishing end-to-end, host-to-host connectivity independent of any underlying network or structure of user (recipient or provider) data. The application layer may include various user (recipient or provider), and support protocols used by applications users (recipient or provider) may use to create and exchange data, utilize services, or provide services over network connections established by the lower layers, including, for example, routing protocols, the hypertext transfer protocol (HTTP), the file transfer protocol (FTP), the simple mail transfer protocol (SMTP), and the dynamic host configuration protocol (DHCP). Such data creation

and exchange in the application layer may utilize, for example, a client-server model or a peer-to-peer networking model. Data from the application layer may be encapsulated into UDP datagrams or TCP streams for interfacing with the transport layer, which may then effectuate data transfer via the lower layers.

5 [0192] The communication component 570 may further implement an internet-of-things (“IoT”) configuration, which may include a network of physical objects—devices, vehicles, buildings, and other items—embedded with electronics, software, sensors, and network connectivity that enables these objects to collect and exchange data via the Internet. Each IoT product/device may be an endpoint device having its own
10 Internet address (e.g., IPv4, IPv6 address). The IoT allows objects to be sensed and controlled remotely across an existing network infrastructure (e.g., the Internet), creating opportunities for more direct integration of the physical world into computer-based systems.

[0193] The device 500 may perform one or more processes described herein. For
15 example, a non-transitory computer-readable medium (e.g., the memory 530 and/or the storage component 540) may store a set of instructions (e.g., one or more instructions, code, software code, and/or program code) for execution by the processor 520. The processor 520 may execute the set of instructions to perform one or more processes described herein. In some implementations, execution of
20 the set of instructions, by one or more processors 520, causes the one or more processors 520 and/or the device 500 to perform one or more processes described herein. In some implementations, hardwired circuitry may be used instead of or in combination with the instructions to perform one or more processes described herein. Thus, implementations described herein are not limited to any specific
25 combination of hardware circuitry and software.

[0194] The number and arrangement of components shown in FIG. 5 are provided as an example. The device 500 may include additional components, fewer components, different components, or differently arranged components than those shown in FIG. 5. Additionally, or alternatively, a set of components (e.g., one or more
30 components) of the device 500 may perform one or more functions described as being performed by another set of components of the device 500.

[0195] In addition to the example configuration described herein in FIG. 5, various steps, functions, and/or operations of the device 500 and the methods disclosed herein may be carried out by one or more of, for example, electronic circuits, logic gates,
35 multiplexers, programmable logic devices, ASICs, analog or digital

controls/switches, microcontrollers, or computing systems. Program instructions implementing methods such as those described herein may be transmitted over or stored on carrier medium. The carrier medium may include a storage medium such as a read-only memory, a random-access memory, a magnetic or optical disk, a non-volatile memory, a solid-state memory, a magnetic tape, and the like. A carrier medium may include a transmission medium such as a wire, cable, or wireless transmission link.

[0196] FIG. 6 is a diagram of example components of a device 670, according to one or more implementations herein. The device 670 may correspond to the communication component 570 or another device or component illustrated in FIG. 4 or FIG. 5. In some implementations, the communication component 570 or another device or component illustrated in FIG. 4 or FIG. 5 may include one or more of the devices 670 and/or one or more components of the device 670. As shown in FIG. 6, the device 670 may include one or more input components 672 (herein referred to collectively as the input components 672 or individually as the input component 672), a switching component 674, one or more output components 676 (herein referred to collectively as the output components 676 or individually as the output component 676), and a controller 678.

[0197] The input component 672 may be one or more points of attachment for one or more input physical links 671 (herein referred to collectively as the input physical links 671 or individually as the input physical link 671) and include one or more points of entry for incoming traffic, such as packets. The input component 672 may process incoming traffic, such as by performing data link layer encapsulation or decapsulation. In some implementations, the input component 672 may transmit and/or receive packets. In some implementations, the input component 672 may include an input line card that includes one or more packet processing components (e.g., in the form of integrated circuits), such as one or more interface cards (IFCs), packet forwarding components, line card controller components, input ports, processors, memories, and/or input queues. In some implementations, the device 670 may include one or more the input components 672.

[0198] The switching component 674 may interconnect the input components 672 with the output components 676. In some implementations, the switching component 674 may be implemented via one or more crossbars, via busses, and/or with shared memories. The shared memories may act as temporary buffers to store packets from the input components 672 before the packets are eventually scheduled for delivery to the output components 676. In some implementations, the switching

component 674 may enable the input components 672, the output components 676, and/or the controller 678 to communicate with one another.

[0199] The output component 676 may store packets and may schedule packets for transmission on the output physical link(s) 679 (herein referred to collectively as the output physical links 679 or individually as the output physical link 679). The output component 676 may support data link layer encapsulation or decapsulation, and/or a variety of higher-level protocols. In some implementations, the output component 676 may transmit packets and/or receive packets. In some implementations, the output component 676 may include an output line card that includes one or more packet processing components (e.g., in the form of integrated circuits), such as one or more IFCs, packet forwarding components, line card controller components, output ports, processors, memories, and/or output queues. In some implementations, the device 670 may include one or more output components 676. In some implementations, the input component 672 and the output component 676 may be implemented by the same set of components (e.g., an input/output component may be a combination of the input component 672 and the output component 676).

[0200] The controller 678 includes a processor in the form of, for example, a CPU, a GPU, an APU, a microprocessor, a microcontroller, a DSP, an FPGA, an ASIC, and/or another type of processor. The processor is implemented in hardware, firmware, or a combination of hardware and software. In some implementations, the controller 678 may include one or more processors that can be programmed to perform a function.

[0201] In some implementations, the controller 678 may include a RAM, a ROM, and/or another type of dynamic or static storage device (e.g., a flash memory, a magnetic memory, an optical memory, etc.) that stores information and/or instructions for use by the controller 678.

[0202] In some implementations, the controller 678 may communicate with other devices, networks, and/or systems connected to the device 670 to exchange information regarding network topology. The controller 678 may create routing tables based on the network topology information, may create forwarding tables based on the routing tables, and may forward the forwarding tables to the input components 672 and/or the output components 676. The input components 672 and/or the output components 676 may use the forwarding tables to perform route lookups for incoming and/or outgoing packets.

- [0203] The controller 678 may perform one or more processes described herein. The controller 678 may perform these processes in response to executing software instructions stored by a non-transitory computer-readable medium. A computer-readable medium is defined herein as a non-transitory (*e.g.*, the medium itself (*i.e.*, tangible, not a signal) as opposed to a limitation on data storage persistency (*e.g.*, RAM vs. ROM)) memory device. A memory device includes memory space within a single physical storage device or memory space spread across multiple physical storage devices.
- [0204] Software instructions may be read into a memory and/or storage component associated with the controller 678 from another computer-readable medium or from another device via a communication interface. When executed, software instructions stored in a memory and/or storage component associated with the controller 678 may cause the controller 678 to perform one or more processes described herein. Additionally, or alternatively, hardwired circuitry may be used in place of or in combination with software instructions to perform one or more processes described herein. Thus, implementations described herein are not limited to any specific combination of hardware circuitry and software.
- [0205] The number and arrangement of components shown in FIG. 6 are provided as an example. In practice, the device 670 may include additional components, fewer components, different components, or differently arranged components than those shown in FIG. 6. Additionally, or alternatively, a set of components (*e.g.*, one or more components) of the device 670 may perform one or more functions described as being performed by another set of components of the device 670.
- [0206] The following figures illustrate example methods and operations thereof. In some implementations, a method illustrated herein may include additional operations, fewer operations, differently arranged operations, or different operations than the operations depicted in the following figures. Moreover, or in the alternative, two or more of the operations depicted in the following figures may be performed at least partially in parallel.
- [0207] In implementations of the methods illustrated in the following figures, various operations may be performed by one or more hardware processors configured by machine-readable instructions (*e.g.*, instructions stored electronically on an electronic storage medium), which may include a module in accordance with one or more embodiments. Such a hardware processor may include one or more processing devices (*e.g.*, one or more digital processors, analog processors, digital

circuits designed to process information, analog circuits designed to process information, state machines, and/or other mechanisms for electronically processing information). The one or more processing devices may include one or more devices configured through hardware, firmware, and/or software, which may be specifically designed for execution of one or more of the operations of methods illustrated herein.

- 5
- [0208] FIG. 7 is a flowchart illustrating an example method 700, according to one or more implementations herein. In some implementations, one or more operations illustrated in FIG. 7 may be performed by one or more of the devices or
- 10 components depicted in FIG. 2 through FIG. 6, in concert, in the alternative, or in combinations thereof. In some implementations, one or more operations may be performed by another device, system, or group of devices or systems separate from or including these. Additionally, or alternatively other devices, components, or systems, may be employed to perform the operations.
- 15 [0209] An operation 702 may include receiving, at a processor of a node of a plurality of nodes, a set of mechanisms, wherein each mechanism defined by at least a provider action and a feasible contract and may be performed alone or in combination with one or more other operations depicted in FIG. 7.
- 20 [0210] An operation 704 may include defining, using the processor, a recipient loss function based on the set of mechanisms, and may be performed alone or in combination with one or more other operations depicted in FIG. 7.
- [0211] An operation 706 may include determining, using the processor, a mechanism of the set of mechanisms that minimizes of a recipient loss function, and may be performed alone or in combination with one or more other operations depicted in
- 25 FIG. 7.
- [0212] An operation 708 may include generating, using the processor, a smart contract based on the mechanism, and may be performed alone or in combination with one or more other operations depicted in FIG. 7.
- [0213] An operation 710 may include deploying, using the processor, the smart contract to
- 30 a blockchain maintained across a plurality of nodes connected via a computer network by broadcasting the smart contract to the plurality of nodes via a network interface in electronic communication with the processor and the computer network, and may be performed alone or in combination with one or more other operations depicted in FIG. 7.

- [0214] An operation 712 may include receiving, at the processor, a transaction address from the blockchain corresponding to the smart contract and may be performed alone or in combination with one or more other operations depicted in FIG. 7.
- 5 [0215] An operation 714 may include storing, using the processor, the transaction address in a database on an electronic storage device of the node and in electronic communication with the processor, and may be performed alone or in combination with one or more other operations depicted in FIG. 7.
- [0216] An implementation of the present disclosure may include a collaboration use case, such as for collaborative work ("Optimal Work"). In this example:
- 10
- Providers: a coworker, employee, freelancer
 - Product: work
 - Recipients: a project owner
- [0217] Problem: Information asymmetry and misalignment between employers and employees is a major challenge in the work industry. It can lead to
- 15 misunderstandings, dissatisfaction, and inefficiencies. This problem is seen through expectation misalignment, communication gaps, lack of transparency, skills and training mismatches, and ineffective feedback mechanisms. From an economic perspective, it illustrates the principal-agent problem in the work industry and can lead to adverse selection and moral hazard. This causes inefficiencies and results
- 20 in high opportunity costs and under investment in human capital.
- [0218] Description of applicable OSCRA:
- Types indicators/variables: role, education level, skills, experience
 - Action indicators/variables: performance, trust/reliability
 - Signal indicators/variables: milestones achieved, wellbeing of the team
 - 25 • Rewards indicators/variables: hourly rate, bonus/slashing
- [0219] In the initial state, there is no OSC (optimal smart contract) for the application Optimal Work and no information on the performance and trust of the providers on the market. The initial performance scoring and trust values of any new providers joining the ecosystem may be initialized at 0.5 (standard continuous uniform
- 30 distribution between 0 and 1). The contributors' initial hourly rates correspond to the 50% percentiles (the median) of the salary distribution taken from statistical market data (e.g., retrieved from salary and market data via the internet). corresponding to the factors explaining the salary level (roles, locations, etc.) for this role.

[0220] The salary is paid in a given asset (for example cash) to the work provider (contributor) according to the hourly rates agreed and the number of hours worked. The effective hours worked are entered into the reporting tool by the contributor and approved by the project manager. Over time, the contributor's scoring changes based on list of indicators, for example performance and trust indicators.

[0221] An implementation of the present disclosure may include asset tracking and transactions ("Optimal Asset"). In this example:

- Providers: the issuers of these asset (shares / token / commodities) have privileged information on the risk (operational risk, credit risk, etc.) and potential of the underlying asset
- Product: a fair and risk adjusted valuation of the asset
- Recipients: the investor of this asset

[0222] Description of applicable OSCRA:

- Types indicators/variables: risk level of the asset, asset class (innovation/potential, sustainability, ESG, etc.), team potential, etc.
- Action indicators/variables: performance, ESG scores
- Signal indicators/variables: milestones from project achieved
- Rewards indicators/variables: Cash payment for the purchase of the asset

[0223] An implementation of the present disclosure may include investment management ("Optimal Invest"). Optimal Invest is a solution for the retail to invest a portfolio of assets in self-custody with a Pareto optimal asset allocation (smart contract based asset management tool according to their personalized risk profile - the assets can be Bitcoin, Altcoins, Stablecoins and other RWA of commodities like Gold or tokenized stocks). It protect retail users from crypto scams with an asset management solution in self-custody aligned to their risk appetite. It's an Optimal DApp with these key roles and characteristics:

- Providers: asset manager, having more information on the market forces (holders and shorters of these assets) and could collude with the issuers of these assets (e.g., on the order books of the market)
- Product: the assets of the portfolio
- Recipients: the investor of these assets

[0224] Today, unfairness may be triggered through credit risk from the issuer and pump and dump schemes from market forces. From these both player types, there are lack of transparency and incentives to behave good. With this solution, we provide to the recipient transparency, verifiability and eventually fairness and real ownership of his asset strategy.

[0225] Problem: Today's providers or issuers of investment portfolios have multiple choices of assets (such as stocks, forex, CDFs, ETFs commodities, bonds, mutual fund, cryptocurrencies, tokenized stocked etc.), have privileged information on market order books, for example, or specific market forces (e.g., holders and shorter of these assets), so the contractual relationship is imbalanced. Unfairness may be triggered through credit risk from the issuer or pump and dump schemes from market forces.

[0226] In this example, the following shortcomings described above are resolved:

- Pareto optimal asset allocation is calculated according to risk appetite of the member and the actual risk on the market for every asset invested
- Pareto optimal Invest also integrates a weighted DCA approach (Dollar Cost Averaging approach) based on current asset risk and on the member's risk aversion to smooth out the risk associated with market volatility: each period (e.g. week), the allocation is readjusted automatically and on chain and in self-custody to get closer to the Pareto optimal asset allocation
- Even assuming that the member's risk aversion is unchanged, the Pareto optimal asset allocation changes according to the market risk for each asset (and in particular its phase in its cycle). In this scenario, the onset of a recession lowers the price of risk-on assets such as Bitcoin and Ethereum: this is the time to increase allocations to these assets if the long-term fundamental value has not changed.
- Over time, the current allocation converges towards the Pareto optimal allocation
- Optimal Invest is both an Optimal DApp and an asset management tool.
- Our smart contract-based asset management tool with Pareto optimal resolution algorithm is designed to deliver on an investor's personalized risk profile in a transparent, decentralized manner, free from the hidden motives of issuers.

- This on-chain asset management tool also features a DCA approach based on the current asset risk and investor risk aversion, meaning allocation is automatically adjusted to smooth out risks associated with market volatility.
- The Optimal Invest tool will be available to investors via a DAO membership fee.

5

[0227] This example provides for:

- Customization: tailor-made risk profile with Pareto optimal asset allocation
- True ownership: self-custody investment portfolio in the investor's control that lives securely on-chain (immutable).
- Risk transparency: investors have access to the same risk information as issuers.
- Native On-chain weighted DCA approach: automated interval investing to save cost and minimize risk.
- Competitive membership fee: Our business model is competitive with typical asset management fees to lower the user entry barrier and promote mass adoption.
- Decentralization & Governance: Investors can participate in the governance of an overarching DAO (Optimal DAO) and have a say on how to shape the services they use.

10

15

20 [0228] How the process works for the recipient:

- Onboarding process: connect on blockchain (ICP) and identify on Decide AI for unique ID
- Register and indicate amounts of assets to be taken under management
- Answer questions:
 - Choose assets X, Y, Z, ... among Bitcoin and one Fiat stablecoin like USDT for the POC (then we add progressively Crypto Gold, other Fiat Stablecoins and Ethereum for the first iteration of the MVP) to invest in.
 - Choose the initial quantity Q0_X, Q0_y, Q0_Z you would deposit into the smart contract (in self-custody).

25

30

- 5
- Is there some of the X, Y, or Z assets you would need to redeem from the smart contract at any point of time in the future with a 95%+ probability (or other probability)? -> we would add a disclaimer taking no liability. If yes, the investment strategy should be highly constrained and really stable on this asset to guarantee with more than 95%+ at any time that we can ensure this.
 - And then some scenarios like "would you feel ok to loose 50% of X, Y, Z in average one year every 4 years but target +30% yearly in average?) to capture the risk aversion of the member
 - 10 ○ Agree on main model assumptions made by our algorithm (bitcoin dominating USD on the long run) about Pareto optimal asset allocation
 - Calculation of the risk level:
 - at first on a centralized server,
 - 15 ○ in the long run on the smart contract (e.g., on ICP)
 - Deposit of one or more assets under management
 - Pay fee based on the assets under management (basically a bit less than 1% of total assets under management)
 - Deposit funds
 - 20 [0229] Description of applicable OSCRA:
 - Types indicators/variables: alignment of the asset manager underlying assumptions with those of the investor, alignment of risk level of the asset management strategy to the risk appetite of the investor, asset classes in portfolio (innovation/potential, sustainability, ESG, etc.), etc.
 - 25 • Action indicators/variables: performance indicator based on previous asset allocation strategy performance or back testing, ESG scores of the assets
 - Signal indicators/variables: P&L of the portfolio
 - Rewards indicators/variables: asset management fee, bonus & risk sharing for good or bad performance
 - 30 • At first, an associated company of Optimal DAO is the only provider (asset manager). The competition to have different providers active on the platform will be opened after the MVP stage.

- In the initial state, there is no OSC (optimal smart contract) for the application Optimal Invest and no information on the performance indicators of the provider (asset manager) on the market. The initial performance scoring and trust values of any new provider joining the ecosystem are initialized at 0.5 (standard continuous uniform distribution between 0 and 1).

[0230] An implementation of the present disclosure may include medical operations, such as dental operations ("Optimal Teeth"). In this example:

- Providers: a medical practitioner
- Product: diagnostic and treatment
- Recipients: the patient

[0231] A second opinion may be provided based on AI models: X-Ray of the member is scanned, and predictive model deliver a diagnostic, so we can protect patients from overtreatment in healthcare and enforce the Optimal Smart Contract in the medical space. In the preparation / training phase of the algorithm (periodically), all X-Rays of the members are scanned, and a predictive model deliver a scoring of the success rate of the practitioner. Inference phase (every time a member wish to generate an OSC), we already have the following data:

- on the product & utility function of the provider: quality scoring, prices, location, availability, UX, etc.
- on the utility function of the recipient (done at the onboarding)

[0232] Description of applicable OSCRA:

- Types indicators/variables: dentist location, UX, availability, waiting time, etc.
- Action indicators/variables: diagnostic accuracy score, treatment success rate, trust/reliability score
- Signal indicators/variables: Best Estimate of AI model (second opinion of the diagnostic), UX rating
- Rewards indicators/variables: price of the diagnostic

[0233] An implementation of the present disclosure may include the used car market ("Optimal Cars"). In this example:

- Providers: used car dealer
- Product: used car market

- Recipients: used car buyer

[0234] An implementation of the present disclosure may include risk pooling (“Optimal Pool”). In this example:

- Providers: risk provider / insured
- Product: transfer of the risk
- Recipients: risk taker / insurer (mutual)

[0235] Description of applicable OSCRA:

- Types indicators/variables: risk level
- Action indicators/variables: effort to reduce the risk exposure (frequency, intensity, etc.)
- Signal indicators/variables: realization of the risk
- Rewards indicators/variables: payment of the claim (minus the premium paid before the cover starts)

[0236] An implementation of the present disclosure may include dynamic token splitting for collaborative projects. Such an implementation may combine aspects of a collaboration implementation and an asset tracking and transaction implementations (“Optimal Dynamic Token Split” or “ODTS”).

[0237] OTDS may be similar to a combination of Optimal Work and Optimal Asset to insure a fair dynamic equity / token split among cofounders of a project. Protect cofounders from free riders in startups with a fair shares/token redistribution according to the risk adjusted valuation of the added value they provided to the project.

[0238] An implementation of the present disclosure may include insurance. Such an implementation may combine aspects of a risk pooling implementation and a medical operations implementations (“Optimal Insurance”).

[0239] Optimal Insurance may be similar to a combination of Optimal Teeth (or Optimal Cars, etc.) and Optimal Pool to reduce dramatically the premium in these markets because the information asymmetry and misalignment between risk carriers and third parties that underpin risk prevention and claims management and repair have been resolved by one of the corresponding Optimal DApp.

[0240] Implementations may implement machine learning, a type of artificial intelligence (AI) that provides computers with an ability to learn how to process data without being explicitly programmed. Machine learning focuses on the development of

computer programs that can teach themselves to grow and change when exposed to new data. Machine learning explores the study and construction of algorithms that can learn from and make predictions based on data. Such algorithms may overcome following strictly static program instructions by making data-driven predictions or decisions, through building a model from sample inputs.

[0241] Machine learning may refer to a variety of AI software algorithms, which may be used to perform supervised learning, unsupervised learning, reinforcement learning, deep learning, or any combination thereof. A variety of different machine learning algorithms may be employed in implementations. Examples of machine learning algorithms may include, inter alia, artificial neural network algorithms, Gaussian process regression algorithms, fuzzy logic-based algorithms, or decision tree algorithms.

[0242] In some implementations, more than one machine learning algorithm may be employed. For example, automated classification may be implemented using one type of machine learning algorithm, and adaptive real-time process control may be implemented using a different type of machine learning algorithm. In some implementations, hybrid machine learning algorithms including features and properties drawn from two, three, four, five, or more different types of machine learning algorithms may be employed in implementations.

[0243] Supervised learning algorithms may use labeled training data to infer a relationship between one or more identifiable aspects of a given entity and a classification of the entity according to a specified set of criteria or to infer a relationship between input process control parameters and desired outcomes. The training data may include paired training examples. For example, each training data example may include aspects identified for a given entity and the resultant classification of the given entity. As a further example, each training data example may include process control parameters used in a process and a known outcome of the process.

[0244] Unsupervised learning algorithms may be used to draw inferences from training data including entity data not paired with labeled entity classification data, or input process control parameter data not paired with labeled process outcomes. An example unsupervised learning algorithm is cluster analysis, which may be used for exploratory data analysis to find hidden patterns or groupings in process data.

[0245] Semi-supervised learning algorithms may use both labeled and unlabeled object classification or process data for training. Semi-supervised learning algorithms may typically use a small amount of labeled data with a large amount of unlabeled data.

[0246] Reinforcement learning algorithms may be used, for example, to optimize a process (e.g., steps or actions of the process) to maximize a process reward function or minimize a process loss function. In machine learning environments, reinforcement learning algorithms may be formulated as Markov decision processes. Reward functions or loss functions, which may also be referred to as cost functions or error functions, may map values of one or more process variables and/or outcomes to a real number that represents a reward or cost, respectively, associated with a given process outcome or event. Examples of process parameters and process outcomes include, inter alia, process throughput, process yield, production quality, or production cost. In some cases, the definition of the reward or loss function to be maximized or minimized, respectively, may depend on the choice of machine learning algorithm used to run the process control method, or vice versa. For example, if an objective is to maximize a total reward/value function, a reinforcement learning algorithm may be chosen. If the objective is to minimize a mean squared error loss function, a decision tree regression algorithm or linear regression algorithm may be chosen. In general, the machine learning algorithm used to run the process control method will seek to optimize the reward function or minimize the loss function by identifying the current state of the process; comparing the current state to the reference state, which may be a target intermediate or final state; and adjusting one or more process control parameters to minimize a difference between the two states. This adjustment may include reference to past learning provided by a training data set. Reinforcement learning algorithms differ from supervised learning algorithms in that correct training data input/output pairs are not presented, nor are Pareto sub-optimal actions explicitly corrected. Implementations of these algorithms tend to focus on real-time performance by finding a balance between exploration of possible outcomes based on updated input data and exploitation of past training.

[0247] Deep learning, which may also be known as deep structured learning, hierarchical learning, or deep machine learning, may be based on a set of algorithms that attempt to model high level abstractions in data. Deep learning algorithms may be inspired by the structure and function of the human brain and are part of a broader family of machine learning methods based on learning representations of data. Rooted in neural network technology, deep learning may involve a probabilistic graph model having many neuron layers, commonly known as a deep architecture. Deep learning technology may process information such as, inter alia, image, text, or sound information in a hierarchical manner. An observation (e.g., a feature to be

extracted for reference) can be represented in many ways including, for example, a vector of intensity values, a set of edges, regions of shape, or in another abstract manner. Some representations may simplify the learning task (e.g., face recognition or facial expression recognition). Deep learning can provide efficient algorithms for unsupervised or semi-supervised feature learning and hierarchical feature extraction. Implementations employing deep learning can further benefit from the advantage of deep learning concepts in solving a normally intractable representation inversion problem.

[0248] A deep learning module may be configured as a neural network. The deep learning module may further be a deep neural network with a set of weights that model the world based on training using training data. Neural networks can be understood to implement a computational approach—based on a relatively large collection of neural units—to loosely model the way a human brain solves problems with large clusters of biological neurons connected by axons. Each neural unit may be connected to one or more others, and links can be enforcing or inhibitory in their effect on the activation state of connected neural units. These systems may be self-learning and trained rather than explicitly programmed. Neural network systems excel in areas where a solution or feature detection is difficult to express in a traditional computer program.

[0249] An example of a deep learning algorithm may be an artificial neural network (ANN). Large ANNs including many layers may be used, for example, to map entity data to entity classification decisions or to map input process control parameters to desired process outcomes. ANNs will be discussed in further detail below.

[0250] Neural networks typically include multiple layers, and the signal path may traverse from front to back. The goal of neural networks may be to solve problems in a similar manner to the human brain, although several neural networks may be much more abstract. In a simple example of a neural network, there may be two layers (i.e., sets) of neurons: an input layer that receives an input signal and an output layer that sends an output signal. When the input layer receives an input, it may pass a modified version of the input to the next layer. In a deep network, there may be many layers between the input layer and output layer, allowing the algorithm to use multiple processing layers, which may include multiple linear and non-linear transformations. Modern neural networks typically work with a few thousand to a few million neural units and millions of connections. Neural networks may have various suitable architectures and/or configurations known in the art.

[0251] There are many variants of neural networks with deep architecture depending on the probability specification and network architecture, including, inter alia, deep belief networks (DBN), restricted Boltzmann machines (RBM), random forests, and autoencoders. Implementations of neural networks may vary depending on the size of input data, the number of features to be analyzed, and the nature of the problem. Other layers may be included in the deep learning module besides the neural networks disclosed herein.

[0252] Another type of deep neural network may be a convolutional neural network (CNN), which can be used for analysis of an entity or process. CNNs are commonly composed of layers of different types: convolution, pooling, upscaling, and fully connected layers. In some cases, an activation function such as a rectified linear unit (ReLU) function may be used in some of the layers. In a CNN architecture, there can be one or more layers for each type of operation performed. A CNN architecture may include any number of layers in total, and any number of layers for the different types of operations performed. The simplest CNN architecture starts with an input layer followed by a sequence of convolutional layers and pooling layers (e.g., layers otherwise configured for reducing the dimensionality of the feature map generated by the one or more convolutional layers while retaining the most important features, for example, max pooling layers) and ends with fully connected layers (e.g., a layer in which each of the nodes is connected to each of the nodes in the previous layer). Each convolution layer may include a plurality of parameters used for performing the convolution operations. Each convolution layer may also include one or more filters, which in turn may include one or more weighting factors or other adjustable parameters. In some instances, the parameters may include biases (e.g., parameters that permit an activation function to be shifted). In some cases, the convolutional layers may be followed by an ReLU activation function layer. Other activation functions can also be used, for example, inter alia, saturating hyperbolic tangent, identity, binary step, logistic, arctan, softsign, parametric rectified linear unit, exponential linear unit, softPlus, bent identity, softExponential, Sinusoid, Sinc, Gaussian, or sigmoid functions. The convolutional, pooling and ReLU layers may function as learnable feature extractors, while the fully connected layers may function as machine learning classifiers. As with other artificial neural networks, the convolutional layers and fully connected layers of CNN architectures may include various computational parameters, for example, weights, bias values, and threshold values, which may be trained in a training phase.

- [0253] Another type of deep neural network may be a visual geometry group (VGG) network. For example, VGG networks may be created by increasing the number of convolutional layers while fixing other parameters of the architecture. Adding convolutional layers to increase depth may be made possible by using substantially small convolutional filters in all of the layers. VGG networks may also include convolutional layers followed by fully connected layers.
- [0254] Another type of deep neural network may be a deep residual network. Like some other networks described herein, a deep residual network may include convolutional layers followed by fully connected layers, which may be, in combination, configured and trained for feature property extraction. A deep residual network's layers may be configured to learn residual functions with reference to layer inputs, instead of learning unreferenced functions. Instead of relying on a direct fit of few stacked layers to a desired underlying mapping, a deep residual network's layers may be explicitly allowed to fit a residual mapping, which may be realized by feedforward neural networks having shortcut connections (*i.e.*, connections that skip one or more layers). A deep residual network may be created by inserting shortcut connections into a plain neural network structure including convolutional layers, thereby modifying the plain neural network into a residual learning network.
- [0255] In some implementations, the machine learning module may include a support vector machine (SVM), an artificial neural network (ANN), a decision tree-based expert learning system, an autoencoder, a clustering machine learning algorithm, or a nearest neighbor (*e.g.*, kNN) machine learning algorithm, or combinations thereof, some of which will be described in further detail below.
- [0256] Support vector machines (SVMs) may be supervised learning algorithms used for classification and regression analysis of entity classification data or process control. Given a set of training data examples (*e.g.*, entity or process data), each marked as belonging to a category, an SVM training algorithm may build a model that assigns new examples (*e.g.*, data from a new entity or process) to a given category.
- [0257] FIG. 8 illustrates an artificial neural network (ANN) 800, according to an implementation. ANN 800 may be used for, inter alia, classification or process control optimization according to various implementations.
- [0258] ANN 800 may include any type of neural network module, such as, inter alia, a feedforward neural network, radial basis function network, recurrent neural network, or convolutional neural network.

- [0259] In implementations implementing ANN 800 for entity classification, ANN 800 may be employed to map entity data to entity classification data. In implementations implementing ANN 800 for process optimization, ANN 800 may be employed to determine a Pareto optimal set or sequence of process control parameter settings for adaptive control of a process in real-time based on a stream of process monitoring data and/or entity classification data provided by, for example, observation or from one or more sensors. ANN 800 may include an untrained ANN, a trained ANN, pre-trained ANN, a continuously updated ANN (e.g., an ANN utilizing training data that is continuously updated with real time classification data or process control and monitoring data from a single local system, from a plurality of local systems, or from a plurality of geographically distributed systems).
- [0260] ANN 800 may include interconnected nodes (e.g., x_1-x_i , $x_1'-x_j'$, and y_1-y_k) organized into n layers of nodes, where x_1-x_i represents a group of i nodes in an input layer 802 (e.g., layer 1), $x_1'-x_j'$ represents a group of j nodes in one or more hidden layers 803 (e.g., layer(s) 2 through $n-1$), and y_1-y_k represents a group of k nodes in a final layer 804 (e.g., layer n). Input layer 802 may be configured to receive input data 801 (e.g., sensor data, image data, sound data, observed data, automatically retrieved data, manually input data, etc.). Final layer 804 may be configured to provide result data 805.
- [0261] There may be one or more hidden layers 803, and the number j of nodes in the one or more hidden layers 803 may vary from implementation to implementation. Thus, ANN 800 may include any total number of layers (e.g., the one or more hidden layers 803). One or more of the hidden layers 803 may function as trainable feature extractors, which may allow mapping of input data 801 to preferred result data 805.
- [0262] FIG. 9 illustrates a node 900, according to an implementation. Each layer of a neural network may include one or more nodes similar to node 900, for example, nodes x_1-x_i , $x_1'-x_j'$, and y_1-y_k depicted in FIG. 8. Each node may be analogous to a biological neuron.
- [0263] Node 900 may receive node inputs 901 (e.g., a_1-a_n) either directly from the ANN's input data (e.g., input data 801) or from the output of one or more nodes in a different layer or the same layer. With node inputs 901, the node 900 may perform an operation 903, which while depicted in FIG. 9 as a summation operation, would be readily understood to include various other operations known in the art.
- [0264] In some cases, node inputs 901 may be associated with one or more weights 902 (e.g., w_1-w_n), which may represent weighting factors. For example, operation 903

may sum the products of each of node inputs 901 and associated weights 902 (e.g., $a_i w_i$).

[0265] The result of operation 903 may be offset with one or more biases 904 (e.g., bias b), which may be a value or a function.

5 [0266] Output 906 of node 900 may be gated using an activation (or threshold) function 905 (e.g., function f), which may be a linear or a nonlinear function. Activation function 905 may be, for example, a ReLU activation function or other function such as a saturating hyperbolic tangent, identity, binary step, logistic, arctan, softsign, parametric rectified linear unit, exponential linear unit, softPlus, bent identity, softExponential, Sinusoid, Sinc, Gaussian, or sigmoid function, or any combination thereof.

[0267] Weights 902, biases 904, or threshold values of activation function 905, or other computational parameters of the neural network, can be “taught” or “learned” in a training phase using one or more sets of training data. For example, the parameters may be trained using input data from a training data set and a gradient descent or backward propagation method so that the output value(s) (e.g., a set of predicted adjustments to classification or process control parameter settings) computed by the ANN may be consistent with the examples included in the training data set. The parameters may be obtained, for example, from a back propagation neural network training process, which may or may not be performed using the same hardware as that used for automated classification or adaptive, real-time deposition process control.

[0268] Decision tree-based expert systems may be supervised learning algorithms designed to solve entity classification problems or process control problems by applying a series of conditional (e.g., if-then) rules. Expert systems may include two subsystems: an inference engine and a knowledge base. The knowledge base may include a set of facts (e.g., a training data set including entity data for a series of entities, and the associated entity classification data provided by, for example, a skilled operator, technician, or inspector) and derived rules (e.g., derived entity classification rules). The inference engine may then apply the rules to input data for a current entity classification problem or process control problem to determine a classification of the entity or a next set of process control adjustments.

[0269] Autoencoders (also sometimes referred to as an auto-associator or Diabolo network), may be an ANN used for unsupervised and efficient mapping of input data (e.g., entity data or process data), to an output value (e.g., an entity

classification or optimized process control parameters). Autoencoders may be used for the purpose of dimensionality reduction, that is, a process of reducing the number of random variables under consideration by deducing a set of principal component variables. Dimensionality reduction may be performed, for example, for the purpose of feature selection (e.g., selecting a subset of the original variables) or feature extraction (e.g., transforming of data in a high-dimensional space to a space of fewer dimensions).

[0270] FIG. 10 illustrates a method 1000 of training a machine learning model of a machine learning module, according to an implementation. Use of method 1000 may provide for use of training data to train a machine learning model for concurrent or later use.

[0271] At 1001, a machine learning model including one or more machine learning algorithms may be provided.

[0272] At 1002, training data may be provided. Training data may include one or more of process simulation data, process characterization data, in-process or post-process inspection data (including inspection data provided by a skilled operator and/or inspection data provided by any of a variety of automated inspection tools), or any combination thereof, for past processes that are the same as or different from that of the current process. One or more sets of training data may be used to train the machine learning algorithm used for object defect detection and classification. In some cases, the type of data included in the training data set may vary depending on the specific type of machine learning algorithm employed.

[0273] At 1003, the machine learning model may be trained using the training data. For example, training the model may include inputting the training data to the machine learning model and modifying one or more parameters of the model until the output of the model is the same as (or substantially the same as) external validation data. Model training may generate one or more trained models. One or more trained models may be selected for further validation or deployment, which may be performed using validation data. The results produced by each trained model for the validation data input to the training model may be compared to the validation data to determine which of the models is the best model. For example, the trained model that produces results most closely matching the validation data may be selected as the best model. Test data may then be used to evaluate the selected model. The selected model may also be sent to model deployment in which the best model may be sent to the processor for use in a post-training mode.

- [0274] FIG. 11 illustrates a method 1100 of analyzing input data using a machine learning module, according to an implementation. Use of the machine learning module described by method 1100 may enable, for example, automatic classification of an entity or optimized process control.
- 5 [0275] At 1101, a trained machine learning model may be provided to the machine learning module. The trained machine learning model may have been trained, or under continuous or periodic training by one or more other systems or methods. The machine learning model may be pre-generated and trained, enabling functionality of the module as described herein, which can then be used to perform
10 one or more post-training functions of the machine learning module.
- [0276] For example, the provided trained machine learning model may be similar to ANN 800, include nodes similar to node 900, and may have been trained (or be under continuous or periodic training) using a method similar to method 1000.
- 15 [0277] At 1102, input data may be provided to the machine learning module for input into the machine learning model. The input data may result from or be derived from a variety of different sources, similar to input data 801.
- [0278] The provision of input data at 1102 may further include removing noise from the data prior to providing it to the machine learning algorithm. Examples of data processing algorithms suitable for use in removing noise from the input data may
20 include, inter alia, signal averaging algorithms, smoothing filter algorithms, Kalman filter algorithms, nonlinear filter algorithms, total variation minimization algorithms, or any combination thereof.
- [0279] The provision of input data at 1102 may further include subtraction of a reference data set from the input data to increase contrast between aspects of interest of an entity or process and those not of interest, thereby facilitating classification or
25 process control optimization. For example, a reference data set may include input data for a real or contrived ideal example of the entity or process. If an image sensor or machine vision system is used for entity observation, the reference data set may include an image or set of images (e.g., representing different views) of an
30 ideal entity.
- [0280] At 1103, the machine learning module may process the input data using the trained machine learning model to yield results from the machine learning module. Such results may include, for example, an entity classification or one or more optimized process control parameters.

- [0281] The following clauses may provide additional context for the present disclosure but should be taken in no way as limiting.
- [0282] Clause 1. A method, comprising: receiving, at a processor of a node of a plurality of nodes, a set of mechanisms, wherein each mechanism defined by at least a
5 provider action and a feasible contract; defining, using the processor, a recipient loss function based on the set of mechanisms; determining, using the processor, a mechanism of the set of mechanisms that minimizes the recipient loss function;
generating, using the processor, a smart contract based on the mechanism;
10 deploying, using the processor, the smart contract to a blockchain maintained across a plurality of nodes connected via a computer network by broadcasting the smart contract to the plurality of nodes via a network interface in electronic communication with the processor and the computer network; receiving, at the
processor, a transaction address from the blockchain corresponding to the smart contract; and storing, using the processor, the transaction address in a database on
15 an electronic storage device of the node and in electronic communication with the processor.
- [0283] Clause 2. The method of clause 1, further comprising verifying the smart contract by transmitting, by the processor, to the blockchain, a request for data in response to the transaction address.
- 20 [0284] Clause 3. The method of one or more of clauses 1–2, further comprising receiving, at the processor, oracle data, wherein each mechanism is further defined by the oracle data.
- [0285] Clause 4. The method of clause 3, wherein the oracle data is generated by:
transmitting, by the processor, a prompt to a trained machine learning model; and
25 receiving, at the processor, outputs of the trained machine learning model as the oracle data.
- [0286] Clause 5. The method of clause 4, wherein receiving the outputs of the trained machine learning model as the oracle data includes performing, using the
processor, a data operation on raw output data of the trained machine learning
30 model to yield the oracle data.
- [0287] Clause 6. The method of one or more of clauses 3–5, wherein the oracle data is configured by a prompt received from a provider node.
- [0288] Clause 7. The method of one or more of clauses 3–6, wherein the oracle data is configured by a prompt received from a recipient node.

- [0289] Clause 8. The method of one or more of clauses 1–7, further comprising deploying, using the processor, a transaction of a token to the blockchain.
- [0290] Clause 9. The method of clause 8, wherein the token is a non-fungible token.
- [0291] Clause 10. The method of clause 9, wherein the non-fungible token is a non-
5 burnable non-fungible token.
- [0292] Clause 11. The method of one or more of clauses 1–10, wherein each mechanism is further defined by a provider type, a provider signal, and a provider reward.
- [0293] Clause 12. The method of one or more of clauses 1–11, wherein each feasible contract is defined by a provider signal and the provider action of the mechanism.
- 10 [0294] Clause 13. The method of clause 12, wherein each feasible contract is further defined by a provider reward.
- [0295] Clause 14. The method of clause 13, wherein the provider reward is selected by a user from a set of possible provider rewards.
- [0296] Clause 15. The method of clause 14, wherein the set of possible provider rewards
15 is selected based on the provider signal.
- [0297] Clause 16. The method of one or more of clauses 1–15, wherein each feasible contract is constrained by one or more of an incentive compatibility constraint, an individual rationality constraint, or a solvency constraint.
- [0298] Clause 17. The method of one or more of clauses 1–16, further comprising:
20 determining, using the processor, provider validation data; and deploying, using the processor, the provider validation data to the blockchain.
- [0299] Clause 18. The method of clause 17, wherein the provider validation data includes a provider reputation.
- [0300] Clause 19. The method of one or more of clauses 17–18, wherein the provider
25 validation data includes a certificate rating.
- [0301] Clause 20. The method of one or more of clauses 1–19, further comprising retrieving, using the processor, from the blockchain, provider validation data, wherein each mechanism is further defined by the provider validation data.
- [0302] Clause 21. A system, comprising: a processor of a node of a plurality of nodes; a
30 network interface in electronic communication with the processor and a computer network; and an electronic storage device in electronic communication with the processor, the electronic storage device having a database stored thereon; wherein the processor is configured to: receive a set of mechanisms, wherein each

- mechanism defined by at least a provider action and a feasible contract; define a recipient loss function based on the set of mechanisms; determine a mechanism of the set of mechanisms that minimizes the recipient loss function; generate a smart contract based on the mechanism; deploy the smart contract to a blockchain maintained across a plurality of nodes connected via a computer network by broadcasting the smart contract to the plurality of nodes via the network interface; receive a transaction address from the blockchain corresponding to the smart contract; and store the transaction address in the database of the electronic storage device.
- 5
- 10 [0303] Clause 22. The system of clause 21, wherein the processor is further configured to verify the smart contract by transmitting to the blockchain a request for data in response to the transaction address.
- [0304] Clause 23. The system of one or more of clauses 21–, wherein the processor is further configured to receive oracle data, wherein each mechanism is further
- 15 defined by the oracle data.
- [0305] Clause 24. The system of clause 23, wherein the oracle data is generated by: transmitting, by the processor, a prompt to a trained machine learning model; and receiving, at the processor, outputs of the trained machine learning model as the oracle data.
- 20 [0306] Clause 25. The system of clause 24, wherein receiving the outputs of the trained machine learning model as the oracle data includes performing a data operation on raw output data of the trained machine learning model to yield the oracle data.
- [0307] Clause 26. The system of clause 23, wherein the oracle data is configured by a prompt received from a provider node.
- 25 [0308] Clause 27. The system of one or more of clauses 23–26, wherein the oracle data is configured by a prompt received from a recipient node.
- [0309] Clause 28. The system of one or more of clauses 21–27, wherein the processor is further configured to deploy a transaction of a token to the blockchain.
- [0310] Clause 29. The system of clause 28, wherein the token is a non-fungible token.
- 30 [0311] Clause 30. The system of clause 29, wherein the non-fungible token is a non-burnable non-fungible token.
- [0312] Clause 31. The system of one or more of clauses 21–30, wherein each mechanism is further defined by a provider type, a provider signal, and a provider reward.

- [0313] Clause 32. The system of one or more of clauses 21–32, wherein each feasible contract is defined by a provider signal and the provider action of the mechanism.
- [0314] Clause 33. The system of clause 32, wherein each feasible contract is further defined by a provider reward.
- 5 [0315] Clause 34. The system of clause 33, wherein the provider reward is selected by a user from a set of possible provider rewards.
- [0316] Clause 35. The system of clause 34, wherein the set of possible provider rewards is selected based on the provider signal.
- 10 [0317] Clause 36. The system of one or more of clauses 21–35, wherein each feasible contract is constrained by one or more of an incentive compatibility constraint, an individual rationality constraint, or a solvency constraint.
- [0318] Clause 37. The system of one or more of clauses 21–36, wherein the processor is further configured to: determine provider validation data; and deploy the provider validation data to the blockchain.
- 15 [0319] Clause 38. The system of clause 37, wherein the provider validation data includes a provider reputation.
- [0320] Clause 39. The system of one or more of clauses 37–38, wherein the provider validation data includes a certificate rating.
- 20 [0321] Clause 40. The system of one or more of clauses 21–39, wherein the processor is further configured to retrieve, from the blockchain, provider validation data, wherein each mechanism is further defined by the provider validation data.
- 25 [0322] Clause 41. A tangible, non-transitory, computer-readable medium having instructions thereupon which when implemented by a processor of a node of a plurality of nodes cause the processor to perform a method comprising: Receiving a set of mechanisms, wherein each mechanism defined by at least a provider action and a feasible contract; defining a recipient loss function based on the set of mechanisms; determining a mechanism of the set of mechanisms that minimizes the recipient loss function; generating a smart contract based on the mechanism; deploying the smart contract to a blockchain maintained across a plurality of nodes
- 30 connected via a computer network by broadcasting the smart contract to the plurality of nodes via a network interface in electronic communication with the processor and the computer network; receiving a transaction address from the blockchain corresponding to the smart contract; and storing the transaction address

in a database on an electronic storage device of the node and in electronic communication with the processor.

- 5 [0323] Clause 42. The tangible, non-transitory, computer-readable medium of clause 41, wherein the method further comprises verifying the smart contract by transmitting to the blockchain a request for data in response to the transaction address.
- [0324] Clause 43. The tangible, non-transitory, computer-readable medium of one or more of clauses 41–42, wherein the method further comprises receiving oracle data, wherein each mechanism is further defined by the oracle data.
- 10 [0325] Clause 44. The tangible, non-transitory, computer-readable medium of clause 43, wherein the oracle data is generated by: transmitting, by the processor, a prompt to a trained machine learning model; and receiving, at the processor, outputs of the trained machine learning model as the oracle data.
- [0326] Clause 45. The tangible, non-transitory, computer-readable medium of clause 44, wherein receiving the outputs of the trained machine learning model as the oracle data includes performing a data operation on raw output data of the trained machine learning model to yield the oracle data.
- 15 [0327] Clause 46. The tangible, non-transitory, computer-readable medium of clause 43, wherein the oracle data is configured by a prompt received from a provider node.
- [0328] Clause 47. The tangible, non-transitory, computer-readable medium of one or more of clauses 43–46, wherein the oracle data is configured by a prompt received from a recipient node.
- 20 [0329] Clause 48. The tangible, non-transitory, computer-readable medium of one or more of clauses 41–47, wherein the method further comprises deploying a transaction of a token to the blockchain.
- 25 [0330] Clause 49. The tangible, non-transitory, computer-readable medium of clause 48, wherein the token is a non-fungible token.
- [0331] Clause 50. The tangible, non-transitory, computer-readable medium of clause 49, wherein the non-fungible token is a non-burnable non-fungible token.
- 30 [0332] Clause 51. The tangible, non-transitory, computer-readable medium of one or more of clauses 41–50, wherein each mechanism is further defined by a provider type, a provider signal, and a provider reward.

- [0333] Clause 52. The tangible, non-transitory, computer-readable medium of one or more of clauses 41–51, wherein each feasible contract is defined by a provider signal and the provider action of the mechanism.
- 5 [0334] Clause 53. The tangible, non-transitory, computer-readable medium of clause 52, wherein each feasible contract is further defined by a provider reward.
- [0335] Clause 54. The tangible, non-transitory, computer-readable medium of clause 53, wherein the provider reward is selected by a user from a set of possible provider rewards.
- 10 [0336] Clause 55. The tangible, non-transitory, computer-readable medium of clause 54, wherein the set of possible provider rewards is selected based on the provider signal.
- [0337] Clause 56. The tangible, non-transitory, computer-readable medium of one or more of clauses 41–55, wherein each feasible contract is constrained by one or more of an incentive compatibility constraint, an individual rationality constraint, or a solvency constraint.
- 15 [0338] Clause 57. The tangible, non-transitory, computer-readable medium of one or more of clauses 41–56, wherein the method further comprises: determining provider validation data; and deploying the provider validation data to the blockchain.
- [0339] Clause 58. The tangible, non-transitory, computer-readable medium of clause 57, wherein the provider validation data includes a provider reputation.
- 20 [0340] Clause 59. The tangible, non-transitory, computer-readable medium of one or more of clauses 57–58, wherein the provider validation data includes a certificate rating.
- [0341] Clause 60. The tangible, non-transitory, computer-readable medium of one or more of clauses 41–59, wherein the method further comprises retrieving, from the blockchain, provider validation data, wherein each mechanism is further defined by the provider validation data.
- 25 [0342] The invention is limited only by the appended claims. Variations, characteristics, advantages, implementations, constructions, arrangements, terminology, materials, dimensions, embodiments, illustrations, depictions, and examples composing the above description and accompanying drawings show some possible implementations of the invention without limiting the invention. It is not necessary that every implementation of the invention achieve or possess every advantage, purpose, or characteristic identified herein, and as such, one skilled in the art may
- 30

effect various additions, changes, modifications, or omissions without departing from the scope or spirit of the invention or its legal equivalents.

5 [0343] All ranges are inclusive of the stated limits, the orders of magnitude thereof, and all values and ranges substantially therebetween unless otherwise defined. Unless otherwise stated, every use of “and” forms an inclusive list comprising at least the conjoined elements, and every use of “or” forms an inclusive list comprising at least one element of conjoined elements. Unless otherwise stated, singular usage (*e.g.*, 'a', 'an', or 'the') includes plurals of the same.

10 [0344] The order of recitations in a claim do not imply a temporal or ordered relationship unless unavoidable by the plain language of that claim. No claim may be interpreted to invoke 35 U.S.C. § 212(f) unless that claim recites “means for” or “step for.”

CLAIMS

I claim:

1. A method, comprising:
receiving, at a processor of a node of a plurality of nodes, a set of mechanisms,
5 wherein each mechanism defined by at least a provider action and a feasible contract;
defining, using the processor, a recipient loss function based on the set of mechanisms;
determining, using the processor, a mechanism of the set of mechanisms that
10 minimizes the recipient loss function;
generating, using the processor, a smart contract based on the mechanism;
deploying, using the processor, the smart contract to a blockchain maintained across a plurality of nodes connected via a computer network by broadcasting the smart contract to the plurality of nodes via a network interface in electronic
15 communication with the processor and the computer network;
receiving, at the processor, a transaction address from the blockchain corresponding to the smart contract; and
storing, using the processor, the transaction address in a database on an electronic storage device of the node and in electronic communication with the processor.
- 20 2. The method of claim 1, further comprising verifying the smart contract by transmitting, by the processor, to the blockchain, a request for data in response to the transaction address.
3. The method of claim 1, further comprising receiving, at the processor, oracle data, wherein each mechanism is further defined by the oracle data.
- 25 4. The method of claim 3, wherein the oracle data is generated by:
transmitting, by the processor, a prompt to a trained machine learning model; and
receiving, at the processor, outputs of the trained machine learning model as the oracle data.
5. The method of claim 4, wherein receiving the outputs of the trained machine learning
30 model as the oracle data includes performing, using the processor, a data operation on raw output data of the trained machine learning model to yield the oracle data.
6. The method of claim 3, wherein the oracle data is configured by a prompt received from a provider node.

7. The method of claim 3, wherein the oracle data is configured by a prompt received from a recipient node.
8. The method of claim 1, further comprising deploying, using the processor, a transaction of a token to the blockchain.
- 5 9. The method of claim 8, wherein the token is a non-fungible token.
10. The method of claim 9, wherein the non-fungible token is a non-burnable non-fungible token.
11. The method of claim 1, wherein each mechanism is further defined by a provider type, a provider signal, and a provider reward.
- 10 12. The method of claim 1, wherein each feasible contract is defined by a provider signal and the provider action of the mechanism.
13. The method of claim 12, wherein each feasible contract is further defined by a provider reward.
14. The method of claim 13, wherein the provider reward is selected by a user from a set of possible provider rewards.
- 15 15. The method of claim 14, wherein the set of possible provider rewards is selected based on the provider signal.
16. The method of claim 1, wherein each feasible contract is constrained by one or more of an incentive compatibility constraint, an individual rationality constraint, or a solvency constraint.
- 20 17. The method of claim 1, further comprising:
determining, using the processor, provider validation data; and
deploying, using the processor, the provider validation data to the blockchain.
18. The method of claim 17, wherein the provider validation data includes a provider reputation.
- 25 19. The method of claim 17, wherein the provider validation data includes a certificate rating.

20. The method of claim 1, further comprising retrieving, using the processor, from the blockchain, provider validation data, wherein each mechanism is further defined by the provider validation data.
21. A system, comprising:
- 5 a processor of a node of a plurality of nodes;
a network interface in electronic communication with the processor and a computer network; and
an electronic storage device in electronic communication with the processor, the electronic storage device having a database stored thereon;
- 10 wherein the processor is configured to:
- receive a set of mechanisms, wherein each mechanism defined by at least a provider action and a feasible contract;
define a recipient loss function based on the set of mechanisms;
determine a mechanism of the set of mechanisms that minimizes the recipient
15 loss function;
generate a smart contract based on the mechanism;
deploy the smart contract to a blockchain maintained across a plurality of nodes connected via a computer network by broadcasting the smart contract to the plurality of nodes via the network interface;
- 20 receive a transaction address from the blockchain corresponding to the smart contract; and
store the transaction address in the database of the electronic storage device.
22. The system of claim 21, wherein the processor is further configured to verify the smart contract by transmitting to the blockchain a request for data in response to the
25 transaction address.
23. The system of claim 21, wherein the processor is further configured to receive oracle data, wherein each mechanism is further defined by the oracle data.
24. The system of claim 23, wherein the oracle data is generated by:
transmitting, by the processor, a prompt to a trained machine learning model; and
30 receiving, at the processor, outputs of the trained machine learning model as the oracle data.
25. The system of claim 24, wherein receiving the outputs of the trained machine learning model as the oracle data includes performing a data operation on raw output data of the trained machine learning model to yield the oracle data.

26. The system of claim 23, wherein the oracle data is configured by a prompt received from a provider node.
27. The system of claim 23, wherein the oracle data is configured by a prompt received from a recipient node.
- 5 28. The system of claim 21, wherein the processor is further configured to deploy a transaction of a token to the blockchain.
29. The system of claim 28, wherein the token is a non-fungible token.
30. The system of claim 29, wherein the non-fungible token is a non-burnable non-fungible token.
- 10 31. The system of claim 21, wherein each mechanism is further defined by a provider type, a provider signal, and a provider reward.
32. The system of claim 21, wherein each feasible contract is defined by a provider signal and the provider action of the mechanism.
33. The system of claim 32, wherein each feasible contract is further defined by a
15 provider reward.
34. The system of claim 33, wherein the provider reward is selected by a user from a set of possible provider rewards.
35. The system of claim 34, wherein the set of possible provider rewards is selected based on the provider signal.
- 20 36. The system of claim 21, wherein each feasible contract is constrained by one or more of an incentive compatibility constraint, an individual rationality constraint, or a solvency constraint.
37. The system of claim 21, wherein the processor is further configured to:
25 determine provider validation data; and
deploy the provider validation data to the blockchain.
38. The system of claim 37, wherein the provider validation data includes a provider reputation.
39. The system of claim 37, wherein the provider validation data includes a certificate rating.

40. The system of claim 21, wherein the processor is further configured to retrieve, from the blockchain, provider validation data, wherein each mechanism is further defined by the provider validation data.
41. A tangible, non-transitory, computer-readable medium having instructions thereupon which when implemented by a processor of a node of a plurality of nodes cause the processor to perform a method comprising:
Receiving a set of mechanisms, wherein each mechanism defined by at least a provider action and a feasible contract;
defining a recipient loss function based on the set of mechanisms;
determining a mechanism of the set of mechanisms that minimizes the recipient loss function;
generating a smart contract based on the mechanism;
deploying the smart contract to a blockchain maintained across a plurality of nodes connected via a computer network by broadcasting the smart contract to the plurality of nodes via a network interface in electronic communication with the processor and the computer network;
receiving a transaction address from the blockchain corresponding to the smart contract; and
storing the transaction address in a database on an electronic storage device of the node and in electronic communication with the processor.
42. The tangible, non-transitory, computer-readable medium of claim 41, wherein the method further comprises verifying the smart contract by transmitting to the blockchain a request for data in response to the transaction address.
43. The tangible, non-transitory, computer-readable medium of claim 41, wherein the method further comprises receiving oracle data, wherein each mechanism is further defined by the oracle data.
44. The tangible, non-transitory, computer-readable medium of claim 43, wherein the oracle data is generated by:
transmitting, by the processor, a prompt to a trained machine learning model; and
receiving, at the processor, outputs of the trained machine learning model as the oracle data.
45. The tangible, non-transitory, computer-readable medium of claim 44, wherein receiving the outputs of the trained machine learning model as the oracle data

includes performing a data operation on raw output data of the trained machine learning model to yield the oracle data.

46. The tangible, non-transitory, computer-readable medium of claim 43, wherein the oracle data is configured by a prompt received from a provider node.
- 5 47. The tangible, non-transitory, computer-readable medium of claim 43, wherein the oracle data is configured by a prompt received from a recipient node.
48. The tangible, non-transitory, computer-readable medium of claim 41, wherein the method further comprises deploying a transaction of a token to the blockchain.
49. The tangible, non-transitory, computer-readable medium of claim 48, wherein the
10 token is a non-fungible token.
50. The tangible, non-transitory, computer-readable medium of claim 49, wherein the non-fungible token is a non-burnable non-fungible token.
51. The tangible, non-transitory, computer-readable medium of claim 41, wherein each
15 mechanism is further defined by a provider type, a provider signal, and a provider reward.
52. The tangible, non-transitory, computer-readable medium of claim 41, wherein each feasible contract is defined by a provider signal and the provider action of the mechanism.
53. The tangible, non-transitory, computer-readable medium of claim 52, wherein each
20 feasible contract is further defined by a provider reward.
54. The tangible, non-transitory, computer-readable medium of claim 53, wherein the provider reward is selected by a user from a set of possible provider rewards.
55. The tangible, non-transitory, computer-readable medium of claim 54, wherein the set of possible provider rewards is selected based on the provider signal.
- 25 56. The tangible, non-transitory, computer-readable medium of claim 41, wherein each feasible contract is constrained by one or more of an incentive compatibility constraint, an individual rationality constraint, or a solvency constraint.
57. The tangible, non-transitory, computer-readable medium of claim 41, wherein the
30 method further comprises:
determining provider validation data; and

deploying the provider validation data to the blockchain.

58. The tangible, non-transitory, computer-readable medium of claim 57, wherein the provider validation data includes a provider reputation.
59. The tangible, non-transitory, computer-readable medium of claim 57, wherein the provider validation data includes a certificate rating.
60. The tangible, non-transitory, computer-readable medium of claim 41, wherein the method further comprises retrieving, from the blockchain, provider validation data, wherein each mechanism is further defined by the provider validation data.