Synapse: Decentralized Intelligence

A Decentralized Data and Machine Learning Network

Dan P. Gailey

Version 1.0

Abstract

Discusses the architecture and protocol specifications for building the infrastructure that supports a decentralized data and machine learning network, exchange, marketplace, and its participants.

Contents

List of Figures

List of Tables

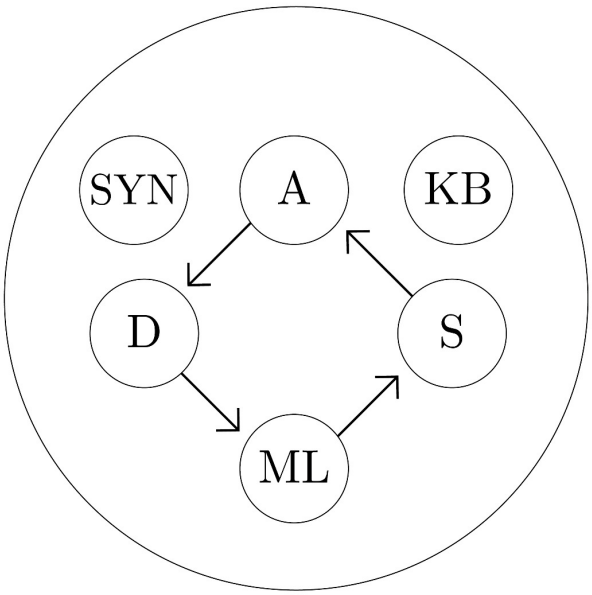

Figure 1: Synapse Network System Components: Agents (**A**), Data Pools (**D**), Models (**M**), Services (**S**), Knowledge Base (**KB**), Tokens (**SYN**), Blockchain (**BC**)

1 Introduction

Machine learning models, a subset of Artificial Intelligence, are only as good as the training data they receive. Until now, access to data has been monopolized and locked away in walled gardens[1].

For the first time, we have an opportunity to give an option for users to re-sell their data on open and decentralized marketplaces on a blockchain[2][3]. This allows more companies to compete for the true value of data in open exchanges, in addition to facilitating provenance and access control of both the data and services provided therein.

This data can be used in turn to train machine learning models, and these models used in turn to build intelligent services and applications both internally and externally to help control all manner of resources.

2 Overview

We discuss the various components and sub-components that make a decentralized network, exchange, and marketplace for data and machine learning feasible.

A central theme for this work acknowledges an emergent stack (web3) whereby distributed and decentralized services with token networks operate in an ecosystem to support higher level applications.

3 System Description

The topology (Figure 1) of the network is defined by multiple sub-systems, or components, working to facilitate the network. In no particular order, these are the components: Agents (**A**), Data Pools (**D**), Models (**M**), Services (**S**), Knowledge Base (**KB**), Tokens (**T, SYN**), Blockchain (**BC**).

3.1 Agents

Agents are defined by any participant in this network. The type of agent (human, firm, autonomous agent) will determine the interface surfaced to the participant.

3.2 Data Storage

Data Pools (**P**) pools are defined as a logical repository for the storage of data and are owned and created by contracts on the blockchain. Data pools exist off-chain while provenance[4] and access control exist on-chain. Data pools should contain the qualities of storing both static and dynamic data. Data pools are managed and metered by off-chain oracles.

3.3 Models

Models (**M**) are trained using the ingress data from data pools. Models can be an endpoint for data fulfillment by participating as a subscriber to a fulfillment contract. Models can be registered and classified through the knowledge base. Models can be graded on accuracy, availability, and throughput.

3.4 Services

Services (**S**) are oracle interfaces that act as metered API gateways for both ingress and egress data.

3.5 Knowledge Base

The Knowledge Base[5] (**KB**) acts as an ontological database that describes a schema of types and relationships for agents, models, services, interfaces with additional considerations for meta data about when these relationships are applied.

3.6 Tokens

Tokens (**T**) are the main asset used to transact license to participate on the network. Our token designation is the Synapse (SYN) token. Currently the token is an ERC20 token that will also participate in an ecosystem of services using atomic swaps.

3.7 Blockchain

A blockchain (**BC**) will be used to capture contracts, subscription state, act as access control, and serve to preserve provenance. The blockchain is a perfect candidate for distributed addressing and cooperative fulfillment.

4 Agents

Agents (**A**) exist in multiple forms, such as people, devices, and firms. In the most general sense, an agent is any participant on the network that also contributes in the fulfillment of data or allocation of services.

4.1 Identity

Identity is afforded through asymmetric encryption where the public key is the addressable identity and wallet address for the participant.

Partnering with identification systems such as Civic to correlate network activity with external identification systems would add extra value to the data as well as allow for more parameters by which we can guarantee intent and enforce reputation among many points of participation.

4.2 Participation

Agents can participate in various ways across the network. The two main ways to participate will be as a light or full node.

A light node contains identity, a local schema, can create and participate in contract creation, has a wallet, and optionally full block access.

A full node represents all the features of a light node, in addition to participating in distributed storage, compute, and has full block access.

4.3 Local Schema

A local ontological schema reflects the specs of the remote schema (Table 1), and describes the local agent services and states.

Changes to this local schema create events that can also be subscribed to. These events can seek to fulfill contracts made in the past as well.

The local agent node contains the ability to query this schema when new blocks are attained to see if fulfillment can be made.

4.4 Contract Interface

[TBD]

4.5 Wallets

Each node will be able to participate in local currency storage and transaction.

4.6 Blockchain Access

Each node will be able to observe and store the blockchain to varying levels of required interest. (e.g. For some nodes particular blocks and contracts are more relevant and interesting than others, therefore some blocks will be pruned while the current hash will be store to validate new blocks.)

5 Contracts

Contracts (**C**) are stored publicly on the blockchain, and are responsible for creating lineage (provenance), and access control through its defined schema and structure (Table 2).

5.1 Contribution Mechanics

Figure 2 demonstrates how Agents can create contracts and contribute to Data Pools.

5.2 Access Control

Access control is granted through the creation and modification of participants, owners, and contributors inside of a contract. An example of state change due to access control can be seen in Table 4. Each of these groups have different levels of access to some resource.

Owners have both read/write access and control the **D** service (defined here as any data or machine learning access node). *Participants* have *write* access and are credited for their contributions on the network. *Subscribers* have *read* access and are charged for their use of the service.

6 Knowledge Base

The Knowledge Base (**KB**) is an ontological database with a shared global addressable schema that describes all agent (devices), interfaces, and services, and data types and formats authorized on the network.

6.1 Storage

Incentivized and metered Mutable DHT[6] torrent files[7] to store the entries and indexes accessible and inherent in full node services.

6.2 Access Control

Reads would be cheap (1E-14 SYN) yet the query order would be highest SYN first. Reads go to incentivize the storage and maintenance of the Knowledge Base.

Writes will be bonded, and forfeited (burned) upon consensus these writes are nefarious/malformed.

6.3 Schema

Schema describes Agents (devices), interfaces, services, and data types and formats on the network5.

6.4 More Consideration

Additional consideration should be given to scoped queries in the form of meta information about the entities contained within the schema. This can further define the scope and state by which the requisite information can be filtered or predicted.

Using the DCIA (Domain, Context, Intent, Action) model we can capture the threads required to carry out some directive. These exist as the vertices (nodes) in a Probablistic Graphical Model (PGM)[8].

6.5 Future Concerns

Table 7 illustrates future concerns to facilitate another database type containing pointers to applied situational-

Type	Resource	Parameters
iPhone	Pandora	Playing
Wallet	Transactions	Amazon
Agent Client	Location	[Boundaries]

Table 1: Local Schema

Contract Type	$\langle Fullfilment, AdHoc, \ldots \rangle$	Immutable
Version	$\{ \ldots \}$	Immutable
Owner(s)	$\{A_0, A_1, \ldots, A_n\}$	Mutable
Participant(s)	$\{A_0, A_1, \ldots, A_n\}$	Mutable
Subscribers	$\{A_0, A_1, \ldots, A_n\}$	Mutable
Balance	$\langle Integer \rangle$	Mutable
Data Type(s)	$\{ \ldots \}$	Mutable
Query	$\{ \ldots \}$	Immutable
Storage Endpoint	$\langle Pointer \rangle$	Immutable
Storage Type	$\langle Type \rangle$	Immutable

Table 2: Contract (**C**) Schema

models (**M**) via probabilistic relative state transitions.

There is also planned concern for meta models of models.

7 Marketplace

The marketplace is where contracts fulfillment happens. The marketplace provides access to services for all agents listening. Using Figure 3 we illustrate a decentralized example transaction in Table 8 where two agents **A** and **B** trade tokens for services.

7.1 Agent Fulfillment protocol

Here we describe a fulfillment protocol for large scale data acquisition from individual agents with local ontologies and queues.

8 Storage

Storage, like all services, are maintained by an off-chain oracle that sits and acts as access control to a logical, distributed storage layer.

9 Models

Models are defined as a service that provides association, classification, and/or prediction. An oracle interface can host a model and serve to subscribe to ingress data events for training. Oracles can also be used to turn the trained model into a service.

Type	Expression	Explaination
Self owned	$C_0(A_0) \rightarrow D_0\{A_0\}$	Contract (C_0) owned by Agent (A_0) such that data storage (D_0) Consists of Participants (A_0)
Multiple contributors	$C_1(A_2) \rightarrow D_1\{A_0, A_1, A_2\}$	Contract (C_1) created and owned by Agent (A_2), with contributors $\{A_0, A1, A_2\}$

Table 3: Contract (**C**) Mechanics

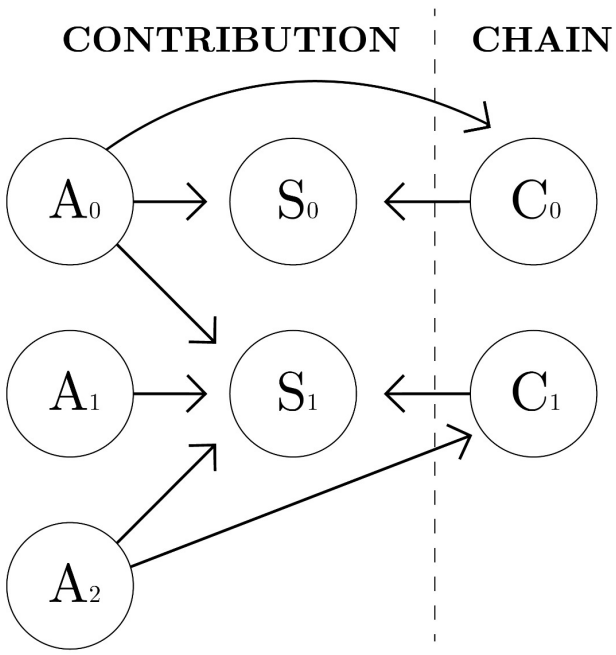

Figure 2: Agents (**A**), Data Pools (**D**), Blockchain (**BC**), Contracts (**C**)

Once a model is trained and available as a service, it can be thought of as some function F that takes an input and delivers some output.

$$[M_{current} \mapsto F(M_{prev})]_{out} \rightarrow M_{next}$$

The marketplace itself is not confined to any specific type of model.

9.1 Model Network

A model network is possible when a third party (agent, oracle, contract) holds contracts and states with a series of models. Each model's output can act as the input of the next model. The network can act to distribute processing by delegating tasks to context specific models.

Cost function:

$$\delta(M_{final}) = \sum_{0}^{n} \delta(M_n)$$

10 Economics

There are multiple points where the economy exists inside the network. The KB, processing, marketplace participation, bonding, contract creation,

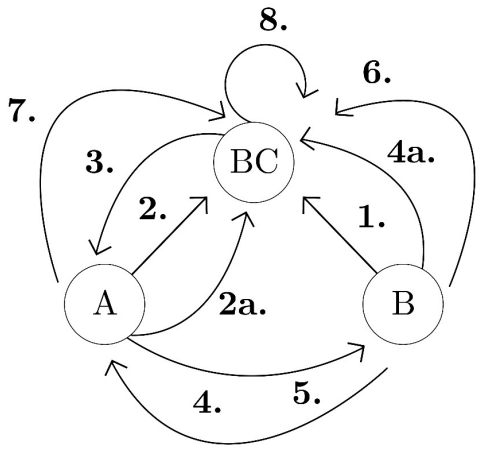

Figure 3: Two agents **A** and **B** interacting with the smart contract on the blockchain **BC**

circular economics [9], but one of the most interesting emergent properties of the network is the ability to create provenance and from the usage of models, reward those that contributed to them. It creates a beautifully similar model to what's seen in propagation functions in neural networks. δ is the revenue for an agent A multiplied by some weight. A_0 denotes the Owner Agent, and w_0 denotes the weight of the model owner's share.

$$Royalties = (\sum_{0}^{n} \delta(A_n) * w_n) + \delta(A_o) * w_o$$

where

$$Royalties = \sum Revenue - \sum Costs$$

$Cm_0 \xrightarrow{\triangle t} Cm_1 \rightarrow Cm_2$			
Action	Expression	State	Explanation
Cm_0	$C(A_0) \rightarrow D_0(A_0)$	$D_0\{A_0\}$	Contract Created
Cm_1	$C(A_0) \rightarrow D_0(A_1)$	$D_0\{A_0, A_1\}$	Access Granted
Cm_2	$C(A_0) \rightarrow D_0(\neg A_1)$	$D_0\{A_0\}$	Access Rescinded

Table 4: State change: Access Control

Type	Resource	Parameters
Human	Genome	Variant Call Format
Android	Spotify	Playlists

Table 5: Global Schema

Domain	Context	Intent	Action
Home	Cooking	Cook Egg	Find Egg
Outside	Driving	Parking	Park

Table 6: DCIA Model

$P(M\|S_A, S_B)$	Probability of M given states A, B
$P(M\|S_A \rightarrow S_B)$	Probability of M given state transition $A \rightarrow B$
$P(M\|M_A \rightarrow M_B)$	Probability of meta M given model state transition $M_A \rightarrow M_B$

Table 7: Probabilistic Model database

Action	Description
$B \rightarrow BC$	**B** post service and cost
$A \rightarrow BC$	**A** search for service
	A deposits bond for service
$BC \rightarrow A$	**A** gets agent **B** service address
$A \rightarrow B$	**A** contacts **B** service
$B \rightarrow BC$	**B** checks for request and bond from **A**
$B \rightarrow A$	**B** sends / fulfills request to **A**
$B \rightarrow BC$	**B** sends confirm to **BC**
$A \rightarrow BC$	**A** sends confirm to **BC**
$BC_{A \rightarrow B}$	**BC** transaction made from **A** to **B**

Table 8: Marketplace Transaction

Figure	Description
1	A_0 listens to BC for some C_0 such that $C_{DataType,Ontology}$ matches O_{local} and $C_{Budget} > 0 + C_{RequestCost} > C_{Budget}$
2	A_0 event oracle adds subscription to event queue Q_{Local}, ordered by price DESC
3	A_0 event oracle sends data requested over transmission protocol to destination, as long as $C_{RequestCost} > C_{Budget}$
4	Once the S repo confirms *ingress* data received E_{A_d}, $C_{Budget} = C_{Budget} - C_{RequestCost}$, and $C_{RequestCost}$ is transacted to wallet WA_o

Table 9: Description of the fulfillment stages

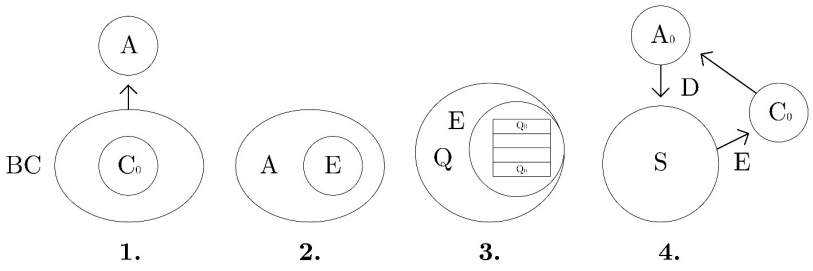

Figure 4: Stage representation in fulfillment

Direction	Action	Description
Ingress	$A_n :: P_{K_o}(D_n) \rightarrow A_o :: P_{K_o}(D_0, D_1, \ldots, D_n)$	A_n sends data D_n encrypted with A_o's public key P_{K_o}
Egress	$A_n :: Req(D_\alpha) \rightarrow A_o$	A_n sends a request for data D_α from A_o
-	$A_o :: S_{K_o}(P_{K_o}(D_\alpha)) = D_\alpha$	Oracle decrypts data requested
-	$A_n :: P_{K_n} \leftarrow A_o :: P_{K_n}(D_\alpha)$	Oracle re-encrypts data and sends it to requestor.

Table 10: Ingress data storage between agents A_n and A_{oracle}

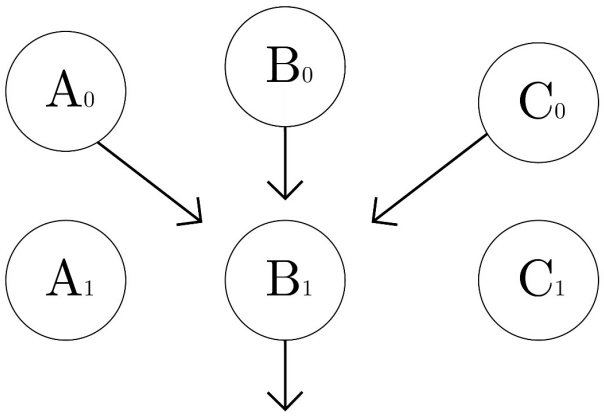

Figure 5: Model Network

References

[1] Various, "Closed platform", [Online]. Available: https://en.wikipedia.org/wiki/Closed_platform.

[2] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system", 2009. [Online]. Available: https://bitcoin.org/bitcoin.pdf.

[3] V. Buterin, "Ethereum white paper", 2015. [Online]. Available: https://github.com/ethereum/wiki/wiki/White-Paper.

[4] Wikipedia, "Data lineage", [Online]. Available: https://en.wikipedia.org/wiki/Data_lineage.

[5] ——, "Knowledge base", [Online]. Available: https://en.wikipedia.org/wiki/Knowledge_base.

[6] ——, "Distributed hash tables", [Online]. Available: https://en.wikipedia.org/wiki/Distributed_hash_table.

[7] L. Matteis, "Updating torrents via dht mutable items", [Online]. Available: http://www.bittorrent.org/beps/bep_0046.html.

[8] D. Koller, "Probabilistic graphical models: Principles and techniques", [Online]. Available: https://www.amazon.com/Probabilistic-Graphical-Models-Principles-Computation/dp/0262013193/.

[9] Wikipedia, "Circular economics", [Online]. Available: https://en.wikipedia.org/wiki/Circular_economy.