Ragtag™

Decentralized asset protocol

Empowering ownership and usage rights for the music industry

Working Group Document

Open Technology Draft

V0.41

Single paragraph summary (TL;DR:)

The Ragtag protocol is a more secure and private way to manage assets using a special type of decentralized ledger technology. It is designed to make sure that the rights and rules for assets are clearly communicated and to protect privacy using advanced security techniques. Additionally, it has strong safeguards in place to prevent fraud. The protocol will first be used for digital music assets.

Executive summary

The Ragtag protocol is a decentralized ledger technology solution for the management of music assets. It aims to simplify the process of obtaining licenses and the payment of royalties, while also making the process more secure and transparent. The protocol provides a secure and private way to manage music assets, aligns incentives across the music industry, and ensures that the rights and rules for music assets are clearly communicated. The Ragtag protocol is the first of its kind to be used specifically for music assets, and the protocol proposal is designed to be flexible and adaptable to a wide range of asset classes and use cases.

The protocol includes the following specifications:

- Definition of the basic classes of assets, including the asset class, asset definition, ownership rights, transfer rules, access rules, royalty rules, and dispute resolution.
- Implementation of measures to prevent gaming, specifically a Bayesian-based inverted consensus mechanism.
- Handling of privacy considerations, such as anonymous ownership, data encryption, access controls, data minimization, and data retention.
- API with appropriate endpoints, such as asset creation, transfer, access, valuation, dispute resolution, and enforcement of anti-fraud measures, sanctions and penalties, audit and compliance, and liability and indemnification.

Intended Audience:
The protocol proposal document is intended for a wide range of stakeholders, including blockchain developers, asset owners, legal professionals, regulators and organizations. The document provides a comprehensive framework for the representation and management of assets on the ledger, and is designed to meet the needs of a diverse set of industries and sectors. The document is intended to be a resource for those seeking to implement a secure and trustworthy Ragtag protocol, and to support the development of industry protocols for the management of assets on the ledger.

Introduction

The problem

In 2014, DA Wallach wrote a blog post, “Bitcoin for Rockstars: How Cryptocurrency Can Revolutionize The Music Industry”. The post raised three major problems for the music industry:

- Lack of a single dataset for music credits and rights information causing difficulties in obtaining licenses.
- Fragmentation of information among territorial organizations leading to proprietary and valuable data.
- Failure of previous efforts to create a single authoritative global database due to coordination problems among stakeholders.

Wallach broadly discussed a solution, including its potential incentives and benefits:

- A decentralized database, similar to the Bitcoin ledger, could align incentives across the music industry.
- Incentives must exist for existing database-keepers to contribute to a decentralized solution.
- A music credits and rights database could reward contributors with payment for access to the data.
- Writing to the database would require specific permission and would be managed by an independent authority.
- A decentralized, open, global ledger could simplify and efficiently process royalty and licensing payments in the music industry.
- Each song, recording, rights-holder, and payor would have a unique address on the ledger connected by smart contracts.
- Services such as Spotify could issue all-in royalty micro-payments directly to the address of a song.
- The network could replace work done by outdated accounting systems, putting creators at the center of the action.
- The cryptocurrency movement may benefit from exploring the possibilities in this area.

Now almost a decade later, the industry is in a similar position; but a solution has become much easier to build.

Furthermore, such a system implementation would be applicable to many other asset classes, such as Non-Fungible Tokens (NFTs):

> *“Someone should produce a decent intellectual property rights contract for the art underlying collectibles and art NFTs. The Dapper Labs license and the Bored Apes Yacht Club license do not pass muster…The creative commons license, the MIT software license, the Apache license, and the GNU General Public license are all renowned open source legal documents. And there is a glaring gap in the NFT space for a suitable protocol license like those.” Keir Finlow-Bayes*

There are implicit expectations among NFT owners regarding their rights to the underlying art of the NFTs. However, these expectations are not reflected in the existing contracts or the absence thereof, which are intended to establish these rights. The existing intellectual property contracts for NFTs are characterized by contradictions, confusion, and restrictive terms that grant NFT owners limited or no value. NFT owners are looking for a clear understanding of their rights and privileges.

Background and context

The cryptocurrency movement

Since 2009, there has been a rise and development of digital currencies that use cryptography to secure transactions and control the creation of new units. Bitcoin, the first and most well-known cryptocurrency, was created in 2009 and operates on a decentralized ledger technology known as blockchain.

Blockchain technology is a decentralized, digital ledger that records transactions across a network of computers in a secure and transparent manner. In a blockchain, data is stored in blocks that are linked and secured using cryptography. Each block contains a cryptographic hash of the previous block, a timestamp, and transaction data. This creates a secure and tamper-proof chain of blocks that cannot be altered retroactively.

The decentralized nature of blockchain technology means that there is no central authority controlling the ledger and the information it contains. Instead, the ledger is maintained by a network of nodes, which are computers that have a copy of the ledger and validate transactions. This decentralized structure provides a high degree of security and transparency, as any changes to the ledger must be agreed upon by a majority of the nodes in the network.

The cryptocurrency movement has been driven by the belief that decentralized ledger technology and digital currencies can offer a more secure and efficient alternative to traditional financial systems. The technology has the potential to disrupt many industries and provide solutions to various problems, including those faced by the music industry as discussed in the blog post. The use of decentralized ledger technology in the music industry could simplify the process of obtaining licenses and simplify the payment of royalties, while also making the process more secure and transparent.

Smart Contracts

Smart contracts are self-executing computer programs stored on the blockchain and used to manage the representation and transfer of assets on the blockchain. Smart contracts are designed to enforce the rules and conditions associated with the ownership and transfer of assets, and to provide a secure and transparent mechanism for the representation and management of assets on the blockchain. Smart contracts provide a secure and transparent mechanism for the creation, management, and transfer of assets on the blockchain. By using smart contracts to enforce the rules and conditions associated with the ownership and transfer of assets, a protocol could provide a secure and transparent mechanism for the representation and management of assets.

Zero-knowledge proofs and ZK-SNARKS

Zero-knowledge (ZK) proofs are a type of cryptographic technology that allows one party (the prover) to prove to another party (the verifier) that they possess certain information or knowledge, without revealing the information or knowledge itself. This means that the verifier can be convinced that the prover has the required information or knowledge, without actually learning what that information or knowledge is. ZK proofs are often used in applications where it is important to preserve the privacy and confidentiality of sensitive information, such as in asset transactions or identity verification.

In simple terms: Zero-knowledge (ZK) proofs are a way for one person to show another person that they know something, without actually telling them what they know. This way, the person can be sure that the other person knows the information, without having to learn the information themselves. This is useful when people want to keep their information private and safe.

Zero-knowledge Succinct Non-Interactive Argument of Knowledge, or ZK-SNARKs, are a specific type of zero-knowledge proof that allows for efficient and verifiable computation on encrypted data. Unlike traditional zero-knowledge proofs, which can be complex and computationally intensive, ZK-SNARKs are designed to be short, simple, and easy to verify. This makes them particularly useful for applications where it is important to preserve the privacy and confidentiality of sensitive data, while also ensuring the integrity and correctness of the computation. ZK-SNARKs are often used in blockchain and other decentralized systems, where they can help to enable private, secure, and efficient transactions and other operations.

Ownership Certificates

Ownership certificates represent ownership of a physical or digital asset. These certificates are used in a variety of industries, including real estate, art and collectibles, and more. With the advent of blockchain technology, ownership certificates have become a popular way to manage assets in a decentralized and secure manner. However, there is a need for a standardized and consistent protocol for managing these certificates to ensure the security and transparency of the assets they represent.

Ownership certificates have a long and fascinating history, with roots stretching back to the earliest forms of commerce and trade. In many ways, ownership certificates can be seen as precursors to modern-day stock certificates and other forms of financial instruments.

- One of the earliest examples of ownership certificates can be found in the field of agriculture, where certificates were used to represent ownership of land and livestock. In medieval Europe, for instance, lords would issue certificates to tenants, granting them the right to use a portion of land for a specified period of time. These certificates were often adorned with intricate designs and seals, serving as both a symbol of ownership and a form of currency that could be traded or sold.

- In the 16th and 17th centuries, the Dutch East India Company issued certificates to shareholders, granting them partial ownership of the company's vast trade empire. These certificates were highly valued by collectors and often passed down from generation to generation as family heirlooms.

- In the 19th century, ownership certificates took on new forms as the rise of corporations and the stock market led to the creation of modern-day stock certificates. These certificates represented ownership in companies and allowed shareholders to profit from the growth and success of the enterprise.

Despite their diverse history, ownership certificates have remained a vital part of the world of finance and commerce, serving as a means of tracking ownership and facilitating transactions in everything from real estate to fine art. With the advent of decentralized technologies like blockchain, ownership certificates are undergoing a resurgence, offering new opportunities for secure, transparent ownership and transfer of assets.

Purpose of the Protocol

The purpose of the Ragtag protocol is designed to address the issues faced by the music industry with regards to obtaining licenses and the fragmentation of information. The protocol aims to provide a secure and private way to manage music assets using decentralized ledger technology. The main purpose of the Ragtag protocol is to provide a solution that will simplify the process of obtaining licenses and simplify the payment of royalties, while also making the process more secure and transparent.

The Ragtag protocol is a secure and private way to manage music assets using decentralized ledger technology. It aligns incentives across the music industry, simplifies the process of obtaining licenses, and ensures that the rights and rules for music assets are clearly communicated. Additionally, it has strong safeguards in place to prevent fraud and protect privacy.

The Ragtag protocol is the first of its kind to be used specifically for music assets, providing a secure and transparent solution to the issues faced by the music industry. The protocol will provide a decentralized and open global ledger, connecting each song, recording, rights-holder, and payor with a unique address. This will allow for efficient processing of royalty and licensing payments, putting creators at the center of the action.

The protocol is designed to be flexible and adaptable to a wide range of asset classes and use cases, and includes elements to ensure compliance with industry protocols.

Roadmap for Implementation and Deployment:

The implementation and deployment of the Ragtag protocol will be a multi-step process that involves the following steps:

1. The first step will be the creation of a detailed specification for the Ragtag protocol, including early consensus on the basic ownership rights, transfer rules, access rules, and dispute resolution.

2. Technical Implementation: The next step will be the technical implementation of the protocol, which will involve the development of software and systems that support the creation, management, and access of assets on the ledger.

3. Testing and Validation: The third step will be the testing and validation of the implementation, which will involve the use of test cases, simulations, and other methods to ensure that the implementation meets the requirements of the protocol.

4. Deployment and Adoption: The final step will be the deployment and adoption of the protocol, which will involve the implementation of the protocol in real-world applications and the development of industry protocols for the management of assets on the ledger.

The implementation and deployment of the Ragtag protocol will be an ongoing process that will involve the continuous improvement and refinement of the protocol to meet the evolving needs of the music industry. The protocol will provide a secure and trustworthy framework for the representation and management of assets on the ledger, and will support the development of secure and transparent systems for the management of assets.

Technical Requirements

Functional Requirements

The functional requirements for the Ragtag protocol must ensure that the system is able to perform the necessary functions that are required to manage ownership certificates securely and transparently. The following functional requirements must be met:

Asset Definition: The system must allow for the definition and management of music assets on the ledger. This includes the ability to define the type of asset, its ownership, and the associated claims and rights.

Claimant Management: The system must be able to manage claimants and their claims to the assets. This includes the ability to verify the authenticity of claims and to manage the relationships between claimants and assets.

Rights Management for Music Assets: The system must provide a mechanism for managing the rights and privileges associated with the ownership of music assets. This includes the definition and management of usage rights that are specific to music, such as:

- Access rights: the right to access and listen to a specific piece of music
- Display rights: the right to display and perform a piece of music publicly
- License rights: the right to grant a license to others to use a piece of music
- Sell rights: the right to sell a piece of music
- Transfer rights: the right to transfer ownership of a piece of music
- Trade rights: the right to trade a piece of music for other assets
- Auction rights: the right to auction a piece of music

These usage rights will be managed and defined using smart contracts on the decentralized ledger, providing a secure and transparent way to manage the rights and privileges associated with the ownership of music assets. The Ragtag Protocol will simplify the process of obtaining licenses, and ensure that the rights and rules for music assets are clearly communicated, putting creators at the center of the action.

Data Management: The system must be able to securely store and manage the data associated with assets, including metadata and ownership certificates, without counterparty risks from centralized authorities. This includes the ability to securely store and retrieve data, and to ensure the integrity and immutability of the data.

Security: The system must be designed to ensure the security of the assets, ownership certificates, and metadata. This includes the use of secure encryption and authentication techniques, as well as the implementation of access controls to restrict access to the data.

Non-Functional Requirements

In addition to the functional requirements, the Ragtag protocol must also meet certain non-functional requirements that are critical to its performance and user experience. The following non-functional requirements must be met:

Scalability: The system must be scalable and capable of supporting a large number of assets and claimants. This includes the ability to handle high volumes of transactions and data, and to accommodate the growth of the system over time.

Performance: The system must be designed to provide high performance and responsiveness, even under high load conditions. This includes the ability to process transactions quickly and efficiently, and to provide real-time updates to the data.

Usability: The system must be user-friendly and easy to use, with a simple and intuitive interface. This includes the provision of clear and concise documentation, and the ability to perform common tasks without the need for extensive technical knowledge.

Interoperability: The system must be designed to be interoperable with other systems and technologies, including existing ownership certificate protocols and blockchain platforms. This includes the ability to seamlessly integrate with other systems and to exchange data with other platforms.

Reliability: The system must be designed to be highly reliable and to provide continuous availability, even in the face of hardware failures and other unexpected events. This includes the use of redundant components, and the implementation of disaster recovery and business continuity strategies.

Performance Requirements

The Ragtag protocol must be designed to handle high volumes of transactions efficiently and with low latency. This is crucial to ensure the seamless transfer of ownership certificates and the management of assets on the ledger. The protocol must be able to process thousands of transactions per second, with a response time of less than a second. Furthermore, the protocol must be scalable to accommodate growth in the number of assets and users on the ledger.

Security Requirements

The Ragtag protocol must be secure against threats such as hacking, tampering, and theft. This requires the implementation of robust security measures such as encryption, secure key management, and secure storage of assets and certificates on the ledger. The protocol must also be resistant to denial-of-service attacks and other malicious activities that could compromise the integrity of the data stored on the ledger. Additionally, the protocol must be auditable and transparent, with a clear audit trail of all transactions and changes made to the ledger.

Interoperability Requirements

The Ragtag protocol must be designed to be interoperable with other ledgers and protocols. This means that it should be able to support the transfer of assets and ownership certificates between different ledgers and protocols, as well as the integration of new assets and certificates. The protocol must also support the integration of legacy systems, such as existing databases and systems used to manage assets, to ensure a smooth transition to the Ragtag protocol. Additionally, the protocol must be designed to be flexible and adaptable, with the ability to support new use cases and applications as they arise.

Architecture

Overview of the System Architecture

This proposed system architecture provides a scalable, secure, and transparent mechanism for representing and managing ownership rights on the ledger. It can be easily adapted to a wide range of asset classes and use cases, and can ensure compliance with industry protocols and regulations.

To decentralize the ownership certificate protocol, the system architecture is split into eight smart contract classes:

1. Asset Class Smart Contract: This contract defines the characteristics and properties of a specific asset class, such as music assets. It specifies the attributes of the assets, such as the type of asset, the rights and privileges associated with ownership, and the rules for managing the assets.

2. Asset Creation Smart Contract: This contract allows for the creation of new assets, such as songs, and assigns a unique identifier to each asset. It also specifies the initial ownership rights and privileges for the asset.

3. Transfer Smart Contract: This contract manages the transfer of ownership rights and privileges for an asset. It ensures that the transfer of ownership is performed in compliance with the rules and regulations defined in the Asset Class Smart Contract.

4. Usage Rights Smart Contract: This contract defines the usage rights associated with an asset, such as the right to access, display, or license a piece of music. It specifies the rules for using an asset and ensures that usage rights are managed in compliance with the rules and regulations defined in the Asset Class Smart Contract.

5. Licensing Smart Contract: This contract manages the licensing of assets. It ensures that licensing agreements are compliant with the rules and regulations defined in the Asset Class Smart Contract and that license fees are properly collected and distributed.

6. Finance Smart Contract: This contract type manages financial transactions such as for specific usage rights and payments of royalties for assets. It calculates and distributes royalty payments to the appropriate parties based on usage rights and licensing agreements.

7. Compliance Smart Contract: This contract ensures that the system operates in compliance with industry protocols and regulations. It monitors the system for any violations of rules and regulations and triggers appropriate action, such as the suspension of assets or termination of licenses.

8. Dispute Resolution Smart Contract: This contract provides a mechanism for resolving disputes that may arise in the system. It allows parties to submit disputes for resolution and provides a process for resolving disputes in a fair and impartial manner.

Overview of the data model

The Ragtag protocol data model is designed to provide a comprehensive and protocol-driven framework for the representation and management of music assets on the ledger. The model incorporates elements to ensure the security and transparency of the data, and is capable of accommodating a diverse range of assets and use cases.

Music assets in the Ragtag protocol data model are composed of the following attributes:

- Asset ID: a unique identifier for the asset
- Asset Class: the class of asset, such as a song or recording
- Asset Properties: the attributes of the asset, such as the title, artist, and date of creation
- Asset Rights: the rights and privileges associated with the ownership of the asset, such as the right to access, display, license, sell, or trade the asset
- Asset Location: the physical location of the asset, if applicable
- Asset State: the current state of the asset, such as active or inactive

Claims: Claims in the Ragtag protocol data model are statements made by a claimant about the ownership or other attribute of an asset. Claims consist of the following attributes:

- Claimant: the entity making the claim
- Claim Type: the type of claim being made, such as ownership or usage rights
- Claimed Asset: the asset being claimed
- Claim Properties: the attributes of the claim, such as the date the claim was made and the basis for the claim
- Claim State: the current state of the claim, such as approved or denied

Counterclaims: Counterclaims are disputes made about the assets. They consist of the following attributes:

- Counterclaimant: the entity making the counterclaim
- Counterclaimed Asset: the asset being disputed
- Counterclaim Properties: the attributes of the counterclaim, such as the date the counterclaim was made and the basis for the counterclaim
- Counterclaim State: the current state of the counterclaim, such as pending or resolved

Distinct classes of music assets and their attributes

Master Recordings

This class encompasses the original recordings of a song or composition, including the audio, video, and multimedia elements.

- Recording artist
- Recording studio
- Recording date
- Sound engineer
- Producer
- Publisher
- Copyright owner
- ISRC (International Standard Recording Code)

Sheet Music

This class encompasses the written and printed representation of a song or composition, including notation, chords, lyrics, and other elements.

- Composer
- Arranger
- Publisher
- Copyright owner
- Year of composition
- Key signature
- Time signature
- Lyrics
- Chords
- Instrumentation
- Copyright registration

Performance Rights

This class encompasses the rights associated with the public performance of a song or composition, including the right to play, perform, or broadcast a song or composition.

- Song title
- Composer
- Performing artist
- Venue
- Date of performance
- Performing rights organization (PRO)
- License type
- License fee

Sync Licenses

This class encompasses the rights associated with the use of a song or composition in a film, television show, video game, or other media.

- Song title
- Composer
- Licensee (i.e. film, TV, or video game company)
- License type
- License fee
- Territory
- Duration of license
- Type of use (i.e. background, feature, or theme)

Mechanical Licenses

This class encompasses the rights associated with the reproduction and distribution of a song or composition, including the right to make and distribute physical copies of a song or composition.

- Song title
- Composer
- Record label
- Distributor
- Copyright owner
- Mechanical rights organization
- License type
- License fee
- Number of copies to be produced

Digital Downloads

This class encompasses the digital distribution of a song or composition, including the right to download, stream, or play a song or composition.

- Song title
- Composer
- Record label
- Distributor
- Copyright owner
- Digital rights organization
- License type
- License fee
- Territory
- File format
- Bit rate

Lyrics

This class encompasses the written words and lyrics of a song or composition, including the right to publish, distribute, and use the lyrics.

- Song title
- Composer
- Lyricist
- Publisher
- Copyright owner
- Date of creation
- Language
- Copyright registration number

Music Videos

This class encompasses the visual representation of a song or composition, including the right to produce, distribute, and display music videos. It includes Spotify Canvas.

- Song title
- Director
- Production company
- Record label
- Distributor
- Copyright owner
- Release date
- Territory
- Type of use (i.e. broadcast, streaming, or download)
- License type
- License fee

Music Stems/Samples

This asset class encompasses musical elements that can be used to create new songs and compositions, including stems and samples.

- Sample pack or stem library title
- Creator or producer of the sample/stem pack
- File format (e.g. WAV, MP3, MIDI)
- Bit rate and/or sample rate
- License type (e.g. royalty-free, limited use, commercial use)
- License fee (if applicable)
- Number of samples/stems included in the pack
- Musical key or tempo (if applicable)

Cover Art

- Artist or designer of the cover art
- Image file format (e.g. JPEG, PNG, TIFF)
- Image resolution and dimensions
- Color mode (e.g. RGB, CMYK)
- Copyright owner of the image
- Release date of the cover art
- Title of the album or single
- Artist or band name
- Record label

Film Footage

- Film title
- Director
- Producer
- Film studio or production company
- Release date
- Running time
- Genre
- Cast and crew credits
- Distribution company

- Territory
- Aspect ratio
- Film format (e.g. 35mm, digital)
- Sound format (e.g. Dolby Digital, DTS)
- Copyright owner

Remixes and Covers

- Original song title
- Artist
- Composer
- Remix or cover artist
- Record label
- Release date
- License type and fee
- Number of copies produced or downloaded

Music Libraries

- Library name
- Composer
- Genre
- Style
- Instrument
- Track title
- Track length
- Copyright owner
- License type and fee
- Usage type (e.g. background, foreground, theme)

Music Equipment

- Equipment type
- Model
- Manufacturer
- Serial number
- Owner
- Purchase date
- Warranty expiration
- Maintenance history
- Used in which recordings

Music Awards

- Award name
- Category
- Recipient
- Year
- Awarding organization

Music Events

- Event name
- Venue
- Date
- Organizer
- Performers
- Ticket sales
- Aattendance
- Merchandise sales

Overview of the API

The API (Application Programming Interface) is a crucial component of the Ragtag protocol, as it provides the interface for interaction between the various elements of the system. The API allows for the creation, transfer, and management of ownership certificates, as well as the exchange of information between the various components of the system, such as the distributed ledger, the smart contract, and the front-end user interface.

The API is designed to be flexible and scalable, able to accommodate the needs of a wide range of use cases and industries. It is built using modern, secure, and well-established web standards and protocols, such as HTTP, REST, and GraphQL, and uses encryption to ensure the privacy and security of the data transmitted. The API is also designed to be easy to use, with clear and well-documented interfaces, and a comprehensive suite of tools and libraries to support developers in integrating it into their applications and systems.

In addition to its core functions, the API also provides various additional features and capabilities, such as the ability to query and retrieve information about ownership certificates and assets, to manage the lifecycle of certificates, and to enforce the conditions and rules associated with their use. The API also provides support for the management of user accounts and permissions, and for the integration with other systems and services, such as identity management and payment processing.

The key endpoints in version 1 of the API include:

- Asset creation endpoint: This endpoint allows users to create new assets, such as songs, and register them on the distributed ledger.

- Asset transfer endpoint: This endpoint enables the transfer of ownership rights and privileges for an asset.

- Asset information endpoint: This endpoint retrieves information about an asset, such as its attributes, usage rights, and licensing agreements.

- Licensing endpoint: This endpoint manages the licensing of assets and ensures that licensing agreements are compliant with the rules and regulations defined in the Asset Class Smart Contract.

- Royalty payment endpoint: This endpoint calculates and distributes royalty payments based on usage rights and licensing agreements.

- Compliance endpoint: This endpoint monitors the system for any violations of rules and regulations and triggers appropriate action, such as the suspension of assets or termination of licenses.

- Dispute resolution endpoint: This endpoint provides a mechanism for resolving disputes that may arise in the system.

- User account management endpoint: This endpoint manages user accounts and permissions, including authentication and authorization.

- Payment processing endpoint: This endpoint integrates with payment processing systems and services to facilitate the collection of fees and the distribution of royalties.

Claimant Management

The management of claims made by individuals or organizations about the ownership or other attributes of an asset is a critical aspect of the ownership certificate protocol. This includes the management of claims about ownership, location, and other attributes of assets, and the process for resolving disputes related to the ownership, transfer, or value of the asset.

Claim verification

The claimant management process begins with the verification of claims. This includes the use of digital signatures and other security measures to ensure the authenticity of the claims. The verification process helps to prevent fraudulent or false claims and ensures that all claims are made by authorized claimants.

Dispute resolution

In the event of a dispute, the ownership certificate protocol provides a clear and transparent process for resolving the dispute. This may include the use of arbitration or court litigation, if necessary. The dispute resolution process should be fair, impartial, and accessible to all stakeholders, ensuring that all parties have a clear understanding of their rights and responsibilities.

Claimant registry

The registry provides a comprehensive record of all claims made about the ownership or other attributes of assets on the distributed ledger. This registry is maintained and managed by the protocol, and it is designed to be transparent and secure. The registry ensures that all stakeholders have a clear understanding of the rights and responsibilities associated with making and resolving claims about the ownership or other attributes of assets on the distributed ledger.

Rights management framework

The distributed ledger-based music asset ownership certificate protocol is designed to provide a comprehensive and flexible approach to the management of ownership rights and privileges. This framework is specifically designed to support the management of music assets and includes the following key components:

- Asset Registry: The asset registry is a database that maintains a record of all music assets on the distributed ledger, including the specific characteristics and attributes of each asset. This registry ensures that each asset is unique and identifiable and provides a secure and tamper-proof record of the assets' ownership and usage rights.

- Rights Registry: The rights registry is a database that maintains a record of all rights and privileges associated with each music asset, including usage rights, transfer rights, and other rights as defined by the asset owner or other stakeholders. This registry provides a comprehensive view of the rights and privileges associated with each asset, ensuring that the rules for managing the assets are clear and enforceable.

- Contract Management System: The contract management system is responsible for managing and executing smart contracts that define the rights and responsibilities of all stakeholders in the management of music assets on the distributed ledger. These contracts are automatically executed and enforceable, providing a secure and transparent way to manage the ownership and usage of music assets.

- Access Control System: The access control system is responsible for managing access to music assets and enforcing access rules, including the use of digital signatures, encryption, authentication techniques, and other security measures. This system ensures that only authorized parties have access to the assets and that the assets are protected against unauthorized access or modification.

- Valuation System: The valuation system is responsible for determining the value of music assets, including the use of market data, valuation algorithms, and other methods for determining the fair market value of assets. This system provides a transparent and objective way to determine the value of music assets, ensuring that the assets are fairly valued and that royalty payments are correctly calculated and distributed.

- Dispute Resolution System: The dispute resolution system is responsible for resolving disputes related to the ownership, transfer, or value of music assets, including the use of arbitration, court litigation, or other dispute resolution mechanisms. This system provides a fair and impartial way to resolve disputes, ensuring that all parties are protected and that the integrity of the system is maintained.

Usage Rights

This comprehensive and flexible framework for the management of usage rights will ensure that all stakeholders have a clear understanding of the rights and responsibilities associated with the usage and management of assets on the ledger

To ensure a comprehensive and flexible approach to the management of usage rights, the following definitions and guidelines have been established, but can be extended to support any action verb that can be applied to any asset:

- Access: The right to access and use an asset, including the right to view, play, or interact with the asset in a specified manner.

- Display: The right to display an asset, including the right to publicly display, share, or showcase the asset.

- Destroy: The right to destroy an asset, such as the destruction of a CD or the deletion of a digital file.

- Move: The right to move an asset, such as the transfer of ownership or the relocation of property.

- License: The right to license an asset, including the right to sublicense, assign, or transfer the license to another party.

- Exhibit: The right to exhibit an asset, including the right to display the asset in a museum, art gallery, or other public setting.

- Transport: The right to transport an asset, including the right to ship, truck, or otherwise move the asset from one location to another.

- Sell: The right to sell an asset, including the right to negotiate, offer, and accept an offer for the sale of an asset.

- Purchase: The right to purchase an asset based on local laws such as KYC/AML or other

- Store: The right to store an asset, including the right to store an asset in a warehouse, storage facility, or other secure location.

- Transfer: The right to transfer an asset, including the right to transfer ownership, control, or other rights and privileges associated with an asset.

- Trade: The right to trade an asset, including the right to exchange, barter, or trade an asset for another asset or for monetary compensation.

- Auction: The right to auction an asset, including the right to sell an asset through a public auction process.

- Derive: The right to derive new assets from an existing asset, such as the creation of derivative works or the extraction of raw materials from a natural resource.

- Perform: The right to perform a musical composition in public, such as through live concerts, radio broadcasts, or streaming services.

- Record: The right to record a musical composition, including the right to create and distribute physical or digital copies of a recording.

- Reproduce: The right to reproduce a musical composition or recording, such as through the creation of physical or digital copies of a work.

- Distribute: The right to distribute a musical composition or recording, such as through the sale, rental, or streaming of copies of a work.

- Broadcast: The right to broadcast a musical composition or recording, such as through radio, television, or online streaming services.

- Public Display: The right to publicly display a musical composition or recording, such as through music videos, album artwork, or other visual media.

- Performing Rights: The right to collect and distribute royalties for public performances of a musical composition, such as through a performing rights organization (PRO).

- Mechanical Rights: The right to collect and distribute royalties for the reproduction and distribution of a musical composition, such as through a mechanical rights organization.

- Sync Rights: The right to grant licenses for the use of a musical composition or recording in film, television, video games, or other media.

- Master Use: The right to grant licenses for the use of a master recording, such as for sample or remix purposes.

- Public Performance: The right to publicly perform a musical composition, such as through live performances, background music, or other public use.

- Sound Recording Performance: The right to perform a sound recording, such as through radio or other public broadcast.

- Digital Performance: The right to perform a musical composition or sound recording in a digital format, such as through streaming or downloads.

- Print: The right to print sheet music, songbooks, or other musical works.

- Reprint: The right to reprint sheet music, songbooks, or other musical works.

- Arrangement: The right to create an arrangement of a musical composition, such as through adaptation, remix, or sampling.

- Public Domain: The right to access and use a musical composition or recording that is in the public domain, such as through creative commons licensing or other public domain designations.

Incentivizing participation and accuracy

The Ragtag protocol creates a more secure way to manage digital music assets with a new type of decentralized ledger technology. The technology incentivizes accuracy with a rating system that improves on platforms like AirBnB and Uber, that encourages high-quality interactions and benefits all parties.

Game theory

Game theory is a branch of mathematics that studies decision-making in situations where multiple individuals or organizations have conflicting interests. Game theory can help us understand how people make decisions about their behavior and how these decisions can affect their future interactions. In this context, the rating system in the Ragtag protocol can be viewed as a repeated game where the players (claimants and music industry organizations) have the choice to either submit truthful claims or false ones; or not choose to involve themselves at all.

The Nash Equilibrium is a concept in game theory that refers to a situation where each player in a game has chosen the best strategy for themselves given the strategies of the other players. In the context of Uber ratings, The driver and passenger rating system on Uber can also be modeled as a repeated game, where each player (i.e., the driver and passenger) makes decisions about whether to give a high or low rating based on their experience with the other player. The outcome of this repeated game can have an impact on both players' future interactions. In this game, the driver's rating is a measure of the quality of their service, while the passenger's rating is a measure of their behavior during the ride. Both the driver and passenger have an incentive to provide a high rating to each other, as it can increase their chances of getting matched with high-quality partners in the future.

Accuracy and participation

In the case of the music industry, the objective is to incentivize everyone to a) participate and b) provide accurate claims about the underlying assets they submit. In Wallach's original blog post, a structure was proposed for the system to create a Nash Equilibrium across the music industry designed to improve both accuracy and participation:

- use of a decentralized database (accuracy)
- writing to the database would require specific permission and would be managed by an independent authority (accuracy)
- providing incentives for existing database-keepers to contribute (participation)
- rewards for contributors with payment for access to the data (participation)

Using an open database would automatically provide financial incentive through efficiency and cost savings: the primary requirement, then, is for the system to incentivize accuracy.

Crowd intelligence, individual skin in the game

Dragonfly Data Science is a consulting firm that specializes in data science, statistical analysis, and machine learning. They were employed to research and fine-tune a decentralized rating system, and build a simulation to demonstrate the underlying accuracy of a system to be successfully calculated *even when the quality of the submissions is not 100% accurate*. This means that over time, regardless of an individual's accuracy, the system becomes increasingly accurate.

How it works: Bayesian inference

Imagine that you have a bag of marbles that contains red, blue, and green marbles. You know that the bag contains mostly red marbles, but you don't know exactly how many of each color there are. Now, imagine that you draw a marble from the bag, and you see that it is red. How does this new information change your belief about the number of red marbles in the bag?

In the case of the music industry, the marbles represent digital music assets, and the different colors represent the level of data accuracy.

Bayesian inference can help you to answer this question of the marbles. Using Bayesian inference, you can update your belief about the number of red marbles in the bag, by taking into account the prior probability of there being a certain number of red marbles (i.e., before you saw the red marble), and the likelihood of seeing a red marble given the number of red marbles in the bag. This updated belief is called the posterior probability, and it reflects the new information that you have obtained by seeing the red marble.

Bayesian inference is a type of mathematical approach used in data analysis to update probabilities based on new information. They are named after 18th-century mathematician Thomas Bayes and allow for the calculation of probabilities for a particular event or outcome, taking into account prior knowledge and new data. Bayesian models have been successfully applied in a wide range of fields, including finance, healthcare, and marketing, to detect fraud, evaluate risk, and inform decision making. For example, Bayesian models have been used to detect credit card fraud by modeling the probability of a transaction being fraudulent based on the transaction's characteristics, such as the amount, location, and time of day.

Karl Friston, a neuroscientist, has proposed that this is also how our brains work. The brain's goal is to be as accurate as possible about the world: or in other words, minimize the surprise of what is going to happen next. This is achieved through a process of prediction and correction, where the brain predicts the sensory input it will receive and updates its model if the input is different than expected. In the same way, the Ragtag protocol aims to minimize the surprise in its representation of ownership rights by incorporating new information and making corrections to its model. This approach ensures that the protocol is constantly improving its accuracy and reliability, much like the way the human brain is constantly improving its understanding of the world.

Proposed system

Every asset submission to the Ragtag database is called a “claim”, made by an “agent”.

The system organizes claims by using an index, signature, and value. It also has models that help to limit the number of possibilities and make it easier to understand. The “world model” shows how claims at different index numbers relate to each other, the agent model shows how claims made by different people relate to each other, and the measurement model shows how the results for a specific index and person are distributed.

The system looks at all the claims and uses world, agent, and measurement models to give out information, like how reliable the person making the claim is and how accurate the claim is. The more surprising and reliable a claim is, the more important it is. The system works like a group of people, where each person looks at what others are saying and shares what they think is important. The system has low activity but gets more active when things change. It's made to be able to deal with people who say things that aren't true and to make the overall group more resistant to those kinds of problems.

In the system, if someone makes a claim, anyone can challenge it, which makes everyone less sure about what's going on. Claims with the most challenges are the most important to figure out because they add the most confusion. To make things more reliable, the system can ask people who are usually reliable to make a similar claim to the one that's confusing. This lets people look at the claim from different points of view and makes it more likely that they'll find out what's really going on.

The system rewards people who are helpful in figuring out confusing claims with better ratings, so it encourages people to work together and be helpful. This makes the system more reliable and trustworthy.

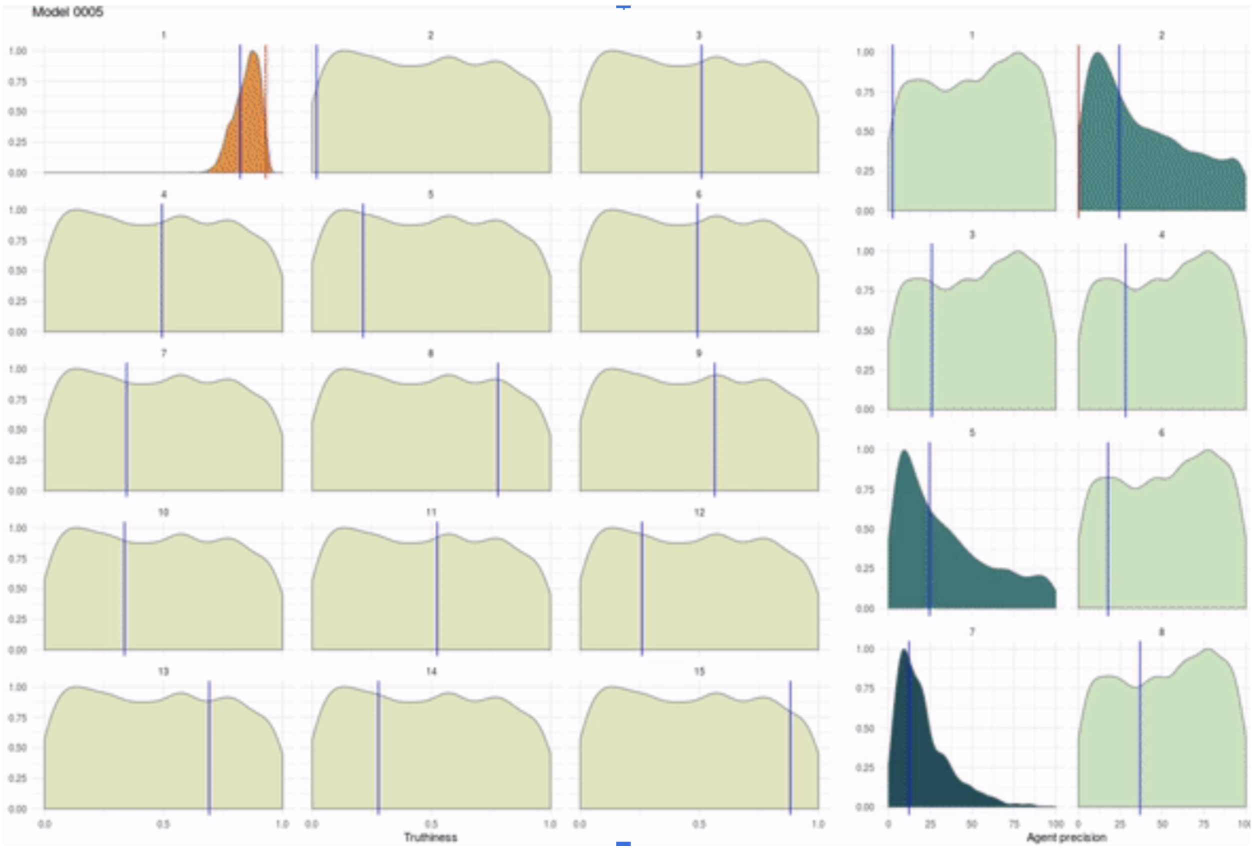
Let's say Alice claims she owns a digital asset called "Song #105". Bob disagrees with Alice and says that he owns Song #105. The system uses something called a probability distribution to show how likely it is that Alice and Bob are the owner. The system measures how different Alice and Bob's probability distributions are, which shows how much confusion there is.

To solve the disagreement, the system asks other people to say who they think owns Song #105. Each time someone says who they think owns Song #105 the system updates the probability distributions for Alice and Bob to be more accurate. If the new probability distributions are similar enough, the dispute is considered solved. If not, more people are asked for their opinion until the dispute is solved.

The system rewards people who helped solve the disagreement by updating their probability distributions to be more reliable.

Simulation

The R programming language was used to show the results of a simulation study that tries to find out how reliable the agents are and how accurate their claims are (see appendix). The code reads in the data from the simulation, analyzes it, and then makes pictures that show how accurate the claims are and how reliable the agents are. The pictures show how the accuracy and reliability change over time, and the code measures how much they change.



This research found that it is possible to estimate the reliability of individuals and the accuracy of their claims at the same time. This means that by using messages from a few agents of different reliability, it's possible to estimate the true accuracy of the music data, even if the agents don't agree with each other. This is more reliable than just averaging out the answers. By taking into account the precision of the agents, we get a more accurate estimate of the underlying value, including how certain we are about the estimate.

Preservation of privacy

The digital ownership certificate protocol will use the Privacy Smart Contract to make sure that the privacy of the people who own the assets is protected. One way it does this is by using something called ZK-SNARKs, which can check if a claim is true without revealing who made the claim. This is useful if someone wants to prove they own something without giving away their identity.

ZK-SNARKs work by using a common reference string that verifies the claim without revealing the data behind it. To use ZK-SNARKs in the Ragtag protocol, the system will be set up to create and check ZK-SNARKs, and a library for ZK-SNARKs will be added to the system. This way, the protocol will be able to use ZK-SNARKs instead of other proofs of ownership.

The Privacy Contract will protect the privacy of asset owners in the digital ownership certificate protocol in several ways. First, anonymous ownership will be allowed, so asset owners can hold and transfer assets without revealing their identity. Second, data encryption will protect the data associated with assets from unauthorized access. Third, access controls will allow asset owners to give or remove access to their data based on specific conditions. Fourth, data minimization will ensure only the necessary data is collected and stored to reduce privacy risks. Finally, data retention policies will make sure that data is only stored for as long as it's necessary and then deleted securely. All of these measures will help protect the privacy of asset owners and their data.

Conclusion

The Ragtag protocol is a significant step forward in the management of music assets, addressing the challenges faced by the music industry with regards to obtaining licenses and the fragmentation of information. The decentralized ledger technology solution provides a secure and private way to manage music assets, aligns incentives across the music industry, and ensures that the rights and rules for music assets are clearly communicated.

The Ragtag protocol is a comprehensive and flexible solution, designed to be adaptable to a wide range of asset classes and use cases. The protocol will play a critical role in promoting the widespread adoption of the Ragtag protocol and will help to ensure its long-term success.

Appendix

Bayesian system detail

(Dragonfly Data Science)

Agents and claims

We consider problems involving a collection of agents making claims where we want to jointly infer the accuracy of those claims as well as the reliability of the agents.

Getting reliable information from a network of (potentially) unreliable agents is a problem with applications in coordination, specifically music industry assets.

In this context, it is also useful to be able to estimate, in a principled way, the number of bits of useful information contained in a message so that the most important claims can be processed first.

We consider a collection of agents $A = \{a_i\}$ and a set of claims $C = \{(a, i, x)\}$. Each claim has three parts:

- an index, i (what the claim is about)
- a signature, a (the agent which makes this claim)
- a value, x (a numeric, or categorical, measurement)

Examples of claims :

- That the barometric pressure at index i, (latitude, longitude, time), has value, x, as measured by the given weather gauge, a
- That an agent will not default on a specified loan, i, as claimed (promised) by that agent, a

- That an agent, b, has a probability of default, x, on a specified loan i, as estimated by another agent, a
- That the bridge at location i is 'at risk of over topping', as observed by agent a
- That in a claim made by an agent b, being generally consistent with other valid information held by agent a, has a validity, x as estimated by that agent

We also aim to ascribe a property validity to all claims, where $v \in [0, 1]$. In the case of claims about measurable reality we define this as the probability that an 'objective measurement' made at index i will return (or would have returned) the same value, x as in the claim.

$$v(a, i, x) = p(x = \hat{x})$$

As we shall see, it is not always necessary to directly measure v in order to be able to usefully reason with it, as we can sometimes infer its value from data.

World model, agent model, measurement model

In the general case, claims are independent and the inference as described above is impossible. However, for domains where we can build (or assume) models that constrain the degrees of freedom, some useful inference is possible. Such constraints generally fall into three areas:

- world model
- agent model
- measurement model

A world model is domain specific and describes how claims made at one index relate to claims made at another index. For example, if weather gauge measurements are indexed by <latitude, longitude, time> a simple world model might note that nearby measurements tend to be similar.

A more complex world model might take into account weather patterns and their tendency to move across space and time in various ways. Whether complex or simple, the world model can be used to spot measurements at one gauge that are inconsistent with the bulk of data from other 'nearby' gauges. These inconsistent measurements are said to be 'surprising'.

An agent model describes how the claims made by one agent relate to claims made by other agents.

For example in Bachrach, Yoram, et al. "How to grade a test without knowing the answers — a Bayesian graphical model for adaptive crowdsourcing and aptitude testing." (2012) they describe a Bayesian model with a latent variable called 'ability' attached to each student. To adapt our model to this scenario, we might consider agents to be the students taking the test, the index to be the question asked, the value, x, to be the answer given by the student and validity to be whether or not the answer was correct. In this case, ability creates a fixed variance or accuracy over student answers. Some agent models ascribe a latent precision as well as an accuracy to agents. More complex agent models may group agents into classes such as in Venanzi, Matteo, et al. "Community-based bayesian aggregation models for crowdsourcing." Proceedings of the 23rd international conference on World wide web. ACM, 2014

A measurement model describes how the measurement, x, for a given agent given a particular index is distributed.

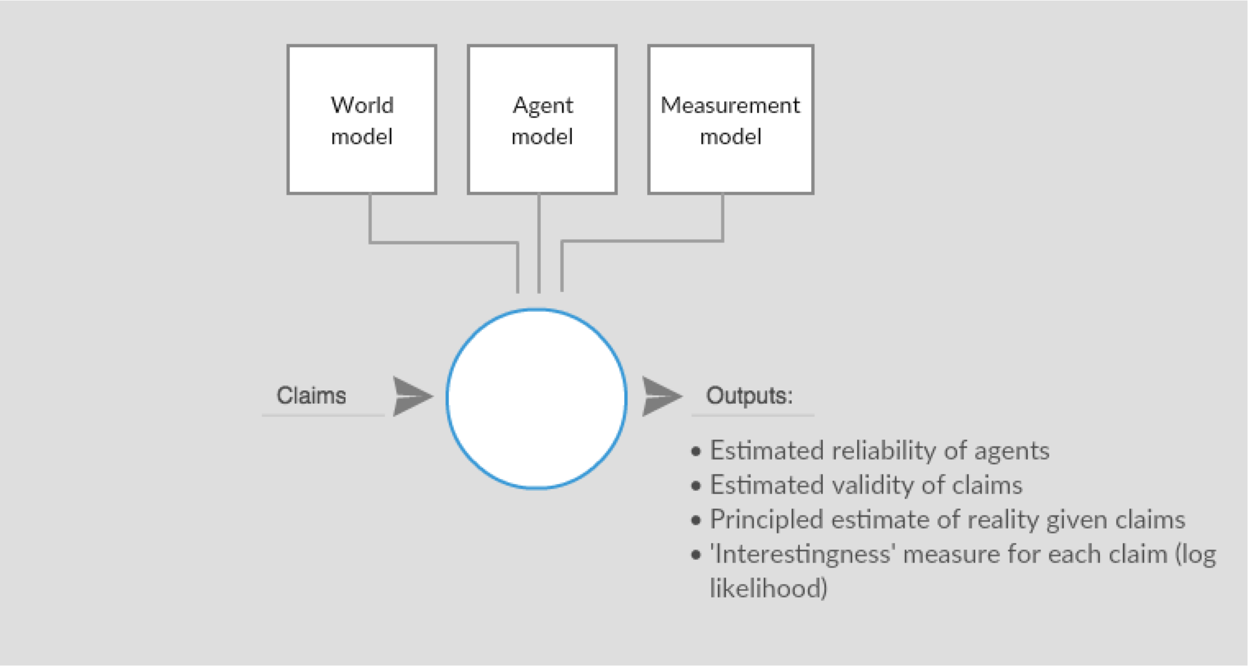
The measurement model describes how the world is represented inside the model. In the simplest case, such as a rain gauge, the measurement model is the accuracy and precision with which rainfall is measured. In the waggle dance carried out by bees, if the world is the distribution of flowers and the time of day, the measurement model described how that relates to the waggle dance.

--

When the domain is constrained, it is possible to infer information about the world, and about the agents. In particular, external validation (such as calibration measurements, or example answers) may not be required. In a simple case, Bayesian modeling can be used to infer both the agent reliability and the claim accuracy in a principled way from observed data.

Furthermore, we can measure the log likelihood of a claim as a proxy for the 'importance' of that message. For instance, given an agent, world, and measurement model, an otherwise surprising claim from a reliable source has a low likelihood and therefore should be considered 'important'. By contrast, surprising claims (per the world model) from unreliable sources have a higher likelihood and can therefore be considered to contain fewer bits of information and be less important.

We hope to build a general model that, given a stream of messages as well as an appropriate world, agent and measurement models can provide the following outputs in a general way.



- Estimated reliability of agents
- Estimated validity of claims
- Principled estimate of reality given claims
- 'Interestingness' measure for each claim (log likelihood)

Networks of agents (channeled communication)
So far, we have assumed a 'god view' where we have access to all claims from all agents. In practice, however, we may only be connected to a limited number of agents.

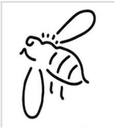
For various reasons we believe it may be useful to consider a network of agents where every agent is performing these kind of evaluations on the messages received from other agents and then preferentially emitting what they consider to be important, valid messages. Each agent may also be initiating measurements or claims. In such a case we can consider the entropy of the system as a whole.

Early investigations suggest that when the 'reality' is held fixed, the messages sent by the network of agents eventually settles to a low-entropy equilibrium with low activity, but jumps to a high activity as soon as the underlying reality changes and measurements become low likelihood.

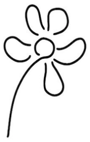
We also wish to explore how resistant such a network would be to 'rogue' agents who, whether through malice or incompetence, tend to emit claims of low validity, and how we can refine the agent and world models to make the overall network more resistant to such attacks.

Some example domains

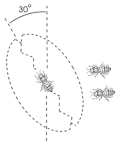
The waggle dance



Agent model = bees are trustworthy



World model = flowers have pollen at specific locations



Measurement model = the waggle dance is used to communicate location of the pollen.

Self grading test / crowdsourcing generally

- Agent model = student aptitude
- World model = no correlation between questions
- Measurement model = some questions are harder than others
- Validity = answer is correct

Claims about claims

- Agents can say they attended a meeting or that they didn't
- Other agents can confirm those claims
- World model = meeting attendance claims should be consistent in that agents can accurately confirm another's attendance at a meeting if they were in attendance themselves
- Measurement model = exchange of keys provides verifiable truth, lowers expected variance in such case
- Agent model = agents should be 100% correct on their own attendance or be considered untrustworthy
- Validity = objective truth

Simulation of Bayesian system in R

```
library(data.table)
library(ggplot2)
library(gganimate)
library(transport)
library(colorspace)
library(gridExtra)

allresults <- readRDS('../generated/truthiness-precision-all-models.rds')
setkey(allresults, iter, st_or_ag)

dir.create('frames', showWarnings=F)

max_st <- 15

load('../generated/data.rdata', v=T)

simdata[, iter := sprintf('%0.4i', 1:.N)]

statements <- seq_len(max_st) # sort(as.numeric(na.omit(allresults[vartype ==
'truthiness', unique(st_or_ag)])))
agents <- seq_len(data$N_AGENTS) #sort(as.numeric(na.omit(allresults[vartype ==
'agent_precision', unique(st_or_ag)])))
n_statements <- length(statements)
n_agents <- length(agents)
```

```
priors <- allresults[iter == '0000']
prior_truth <- priors[vartype == 'truthiness']
prior_prec  <- priors[vartype == 'agent_precision']

## * Restrict number of statements answered
r <- allresults[, max(st_or_ag), iter]
w <- r[max(which(V1 %in% max_st)), iter]
allresults <- allresults[iter <= w]

iters <- setdiff(sort(unique(allresults$iter)), '0000')

t_real <- data.table(st_or_ag = seq_len(data$N_STATEMENTS), value =
data$statements_truthiness)
t_real <- t_real[st_or_ag <= max_st]

p_real <- data.table(st_or_ag = seq_len(data$N_AGENTS),     value =
data$agents_precision)


pids <- progressbar(length(iters), 100)
i=361
for (i in seq_along(iters)) {
  frame <- iters[i]

  outfile <- sprintf('frames/frame_%s.png', frame)
  if (!file.exists(outfile)) {
    png(outfile, width = 1200, height = 800)

    df <- allresults[iter == frame]

    truth <- df[vartype == 'truthiness']
    prec  <- df[vartype == 'agent_precision']

    ## truth[, real_value := data$statements_truthiness[st_or_ag]]
    ## prec[ , real_value := data$agents_precision[st_or_ag]]

    answer <- simdata[iter == frame]
    answer_t <- answer[, .(st_or_ag = statement, value = answer)]
    answer_p <- answer[, .(st_or_ag = agent, value = 0)]

    ## * Truthiness
    missing_st <- setdiff(statements, unique(truth$st_or_ag))
    if (length(missing_st)) {
      truth <- rbind(truth,
             rbindlist(lapply(missing_st, function(st) {
               p <- copy(prior_truth)
               p[, st_or_ag := st]
               return(p)
             })))
    }

    ## * Agent precision
    missing_pr <- setdiff(agents, unique(prec$st_or_ag))
    if (length(missing_pr)) {
      prec <- rbind(prec,
             rbindlist(lapply(missing_pr, function(ag) {
               p <- copy(prior_prec)
               p[, st_or_ag := ag]
               return(p)
             })))
    }

    truth_wdist <- rbindlist(lapply(statements, function(st) {
      data.table(st = st, wdist = wasserstein1d(truth[st_or_ag == st, value], prior_truth[,
value]))
    }))
    prec_wdist <- rbindlist(lapply(agents, function(ag) {
      data.table(ag = ag, wdist = wasserstein1d(prec[st_or_ag == ag, value],
prior_prec[, value]))
    }))

    truth[truth_wdist, wdist := i.wdist, on = c('st_or_ag' = 'st')]
    prec[prec_wdist, wdist := i.wdist, on = c('st_or_ag' = 'ag')]

    gt <- ggplot(truth, aes(x = value)) +
      geom_density(aes(y = ..scaled.., fill = wdist), color = 'grey50', show.legend=F) +
      geom_vline(data = t_real, aes(xintercept = value), colour = 'blue') +
      facet_wrap(~ st_or_ag, nrow = 6) +
      scale_fill_continuous_sequential('Heat2', trans = 'sqrt', limits = c(0, 1)) +
      theme_minimal() +
      scale_x_continuous(limits = c(0, 1), breaks = c(0, .5, 1)) +
      geom_vline(data = answer_t, aes(xintercept = value), color = 'red') +
      labs(x = 'Truthiness', y = '', title = paste0('Model ', frame))

    gp <- ggplot(prec, aes(x = value)) +
      geom_density(aes(y = ..scaled.., fill = wdist), color = 'grey50', show.legend=F) +
      geom_vline(data = p_real, aes(xintercept = value), colour = 'blue') +
      facet_wrap(~ st_or_ag, ncol = 2) +
      scale_fill_continuous_sequential('BluGrn', trans = 'sqrt') +
      theme_minimal() +
      geom_vline(data = answer_p, aes(xintercept = value), color = 'red') +
      scale_x_continuous(limits = c(0, NA)) +
      labs(x = 'Agent precision', y = '', title = '')

    print(cowplot::plot_grid(gt, gp, rel_widths = c(2, 1)))

    dev.off()
  }
}
```