Ara — Global Decentralized Infrastructure for Payments, User Identity, Hosting, Streaming, and Ownership of Digital Files and Content (Draft)

Eric Jiang, Charles Kelly, Joseph Werle, Tony Mugavero, Vanessa Kincaid

Last Updated February 28, 2020 (Partial**)

Abstract

The internet has become a shadow of its former self, dominated by a few large companies that exercise total authority over all the information by controlling how it flows around the world, how much it costs, and how and when it can be consumed. Ara is a decentralized platform and suite of protocols designed to fix that. Through licensing and selling digital files and content using a novel Proof-of-Ownership system, Ara handles global data delivery and supports purchasing of those assets with the native Ara token. In doing all this, the Ara platform also utilizes a ubiquitous and distributed user ID and wallet system which allows users to retain ownership of their personal information. In effect, Ara is a new modern mental model around how information on the internet is hosted and delivered and how consumers use and pay for it; it brings about a new paradigm not only for businesses but also for consumers in that they can contribute in the system to earn rewards by hosting and participating in the network. Peer-to-peer (P2P) file sharing, blockchain technology for ownership and licensing, and distributed computing are all combined into a single efficient and decentralized system.

Numerous constituents benefit from Ara. Consumers can use their idle storage, bandwidth, and processing power to earn Ara tokens–akin to Airbnb for computing–and they can use those tokens to purchase content. Businesses save on delivering this information to people through P2P technology, which in turn lowers costs to consumers and other businesses. Anyone can participate as a data center in the network to earn rewards. Digital creators, game and software developers, movie and TV studios, and publishers use Ara tokens to publish licensed content into the network, earning more on their work by generating larger revenue shares and giving the rest to their fans for the hosting costs. It's a win-win-win. Consumers are rewarded, publishers earn more, and businesses improve the bottom line. This is all done in a decentralized and net neutral way, such that no intermediary companies can throttle a single business delivering information and content to you.

****** This is a partial update to the whitepaper from June 2018. A more thorough update will be released in the coming months.

Note: Ara is under active research and development. This paper is subject to change. The latest version will be available at https://ara.one. Please direct any comments and suggestions to hello@ara.one.

Contents

1. Introduction

1.1 Background

The hypermedia landscape today is outdated. Aggregators and app stores are strengthening their grip on content creators; traditional content delivery networks are inefficient and expensive; cloud compute is centralized to a select few gatekeepers; and data is stored not by those who own it, but by those who profit from it. Content publishers and creators are forced to inflate prices, offloading the costs for this slow and expensive system to consumers and resulting in lost value for publishers, consumers, and creators alike.

With video poised to comprise over 80% of all internet traffic by 2021 [3], the cost of content has continued to rise as file sizes and costs of delivering that content go up. 4K, VR, and AAA games are all contributors to this trend. Consumers are not only paying more for transactional content and subscriptions, but they have to deal with a complex and abusive system of advertising to view free content. These factors exacerbate the problem of piracy, leading to billions of dollars in losses to content owners [16][12][4], and they result in the introduction of tools to skip or remove advertising such as ad-blockers. This in turn creates ad-blocker-blockers and more expensive subscription services as consumers avoid ads and content owners try to recapture lost revenue. It's a vicious cycle.

Peer-to-peer (P2P) file distribution architecture emerged as a response to these inefficiencies, evolving from hybrid solutions that incorporated centralized servers such as Napster to completely decentralized solutions such as Gnutella and eventually BitTorrent. Today, P2P file delivery is so cost-efficient that companies such as Microsoft use it—not their own Azure infrastructure—to save on Windows 10 distribution costs.

However, while cost-efficient, P2P file sharing networks have been historically rife with free-riding, piracy, hacking, and black markets when used in public environments. There was no trust that the individual uploading content had the right to do so, and no way to verify that the content seeded was delivered as the content owner intended. There was also no reward for storing and sharing the content, so users were not incentivized to remain seeds in the system of delivery. Peers would leech content for themselves and not continue to participate in seeding that content out to other peers. To combat this, P2P architectures began to incorporate incentive mechanisms, such as barter strategies, reputation systems, and proprietary currencies. However, even these mechanisms have their share of problems and are subject to Sybil and whitewashing attacks.

1.2 Overview

In this white paper, we present Ara: a community-driven decentralized and distributed compute and content delivery platform. Ara enables any device in the world to become part of a global supercomputer, database, and delivery network all at once by utilizing its unused processing, storage, and bandwidth capacity. Together,

these devices form the Ara network, an ecosystem in which anyone can participate and benefit.

Fundamentally, the network is comprised of an overlapping community of consumers, service requesters, service providers, and software developers, each with their own incentives for adoption. With Ara, service requesters can have at their fingertips a vast reserve of compute resources along with an ever-expanding library of distributed services. Service providers, who have already paid for their devices—be it a smartphone, laptop, or gaming console—can begin making a return renting out unused resources. The only requirement to begin earning rewards is to submit a small deposit that acts as a hold on the account. This hold can be withdrawn at any time, but it is required to earn and redeem any rewards. Software developers can leverage the unparalleled scale of Ara's ecosystem to accomplish hefty compute tasks and create novel distributed services for requesters and providers to participate in. Meanwhile, consumers can go about their daily lives as they would all while being rewarded for watching the shows and listening to the music they love.

Thus, anyone with spare compute resources can immediately act as a service fulfiller to earn rewards for helping distribute content, while anyone looking for remote resources can request Ara decentralized services and enjoy enhanced security, file availability, and delivery speeds for a fraction of the cost compared to traditional cloud compute service providers. Because Ara removes the burden of purchasing and managing infrastructure, content creators of all kinds stand to benefit—from the indie artist who can now freely self-publish his new album without going through a record label, to the large media conglomerate who no longer needs to go through aggregators to reach its audience. Ara relies on the resources provided by members of the network; the larger the network grows, the more robust and efficient it becomes.

1.3 Platform Services

The Ara platform is comprised of three (3) core services and systems:

1. **AraID:** AraID establishes secure, decentralized, and verifiable global identities for all agents and content on the Ara platform, giving control of data back to their rightful owners.
2. **Decentralized Content Delivery Network (DCDN):** DCDN serves as Ara's network of underlying peer-to-peer, secure distributed file systems and storage networks (AFSs) which support content integrity, incentives, versioning, and decentralized identities.
3. **Protocol Suite:** Ara is connected through a secure suite of protocols which enable trustless interoperability between DCDN, AraID, and the Ethereum Blockchain.

2. Platform Overview

The Conflict-Free File System Network (CFSNet) is the backbone of Ara's peer-to-peer distributed file system, AraID, and Decentralized Content Delivery Network (DCDN). Leveraging an underlying Merkle tree structure and the Syncable Ledger of Exact Events Protocol (SLEEP) [1] file format, CFSNet addresses many concerns with traditional file transport—both client-server and P2P—and improves upon existing technologies such as IPFS by providing cryptographically-ensured content integrity as well as versioning and revision history. The network is made up of isolated file systems called CFSs. Furthermore, each CFS instance implements a subset of the Filesystem Hierarchy Standard (FHS) [8], supporting partitions and allowing each directory to exist as a self-contained CFS archive with its own access levels. Of these partitions, AFS uses the `/home` and `/etc` partitions to store AFS content and metadata, respectively. Each CFS partition is publicly identifiable throughout the network using a unique `Ed25519` 32-byte public key generated at the time of creation. A CFS's public key grants read-only access to the file system, where only the holder of the private key can update and publish the contents contained within.

2.1 AraID

AraID is responsible for creating and resolving secure and verifiable decentralized representations for all users and content on the Ara Platform. Fully compliant with the W3C (Decentralized Identifier (DID)) spec [15], AraID uses DID Descriptor Objects, or DDOs for short, to represent users and content (see *Figure 1*). DDOs are simple JSON-LD documents which define methods for authentication and authorization as well as other identity attributes, including service endpoints and private communication channels controlled by the owner [15]. Because DDOs never store Personally-Identifiable Information (PII) [15], these service endpoints and communication channels identify secure means of obtaining it and thus allow entities self-sovereignty over their private data and online identities.

2.1.1 Decentralized Identity

For all users and content on the Ara platform, an AraID is generated in the form of:

- `did:ara:ee93189c629cdaf94 9fd57bac5b005b916936d2a5c6806 40fd1aedc8315730a0`

AraID implements a Universal Resolver `method`, denoted by the second component of the DID (`ara` above) as part of the Decentralized Identity Foundation system [11]. The `method`, also known as a driver, defines how DIDs and DDOs are resolved within the Ara platform. Unlike internet URIs, DIDs do not require a central authority for registration or control and form a bijective correspondence with DDOs rather than the non-injective, non-surjective relationship found in TCP/IP and DNS.

The crux of AraID security is maintained cryptographically using Decentralized Public Key Infrastructure (DPKI)[13], wherein `Ed25519` public keys are used as both the `id` portion of the DID (`ee9318`... above) and the public key of the CFS in which the corresponding DDO is stored. These documents contain a `publicKey` property which holds various keys used for digital signatures, encryption, and other cryptographic operations. When an identity is created, this array is populated with the owning identity's key, as well as the corresponding Ethereum account's public key.

For AFS AraIDs, the public key of the `/etc` partition containing the associated content metadata is also stored. Since the key for this is stored in the AFS DDO, it can be resolved by any requester who has the AFS DID.

Whenever a new identity is generated, a mnemonic phrase is used to seed the keypair. The mnemonic is returned to the owner for safekeeping and for ease of maintaining the private key, allowing entities to easily validate ownership of DIDs and restore accounts without needing the private key.

Identity Archiving and Resolution

When an identity is created, it is initially written locally so that any local resolution can check the cache prior to falling back on the network. However, before an identity can be resolved remotely, it must first be archived. Ara runs archival nodes whose job it is to store these identities for future resolution.

Similarly to the archiver nodes, Ara also runs resolver nodes responsible for querying the archivers for requested DDOs. Resolver requests first look to locally resolve identities that may be stored on disk before reaching out to the network for remote archivers that have archived the AraID in question.

```
{
    'ddo': {
        '@context': 'https://w3id.org/did/v1',
        'id': 'did:ara:ee9318...',
        'authentication;: [{
           'type': 'Ed25519SignatureAuthentication2018',
           'publicKey': 'did:ara:ee9318...#owner'
        }],
        'publicKey': [{
           'id': 'did:ara:ee9318...#eth',
           'type': 'Secp256k1VerificationKey2018',
           'owner': 'did:ara:ee9318...,
           'publicKeyBase58': 'H3C2AVvLMv6gmMNam...'
        }],
        'service': {
           'ens': 'https://etherscan.io/enslookup',
        }
        ...
    }
}
```

Figure 1: *Example DDO*

Ethereum Account

Each identity is created with an Ethereum account and an associated Ethereum wallet, recoverable using a generated random mnemonic during identity creation. Since the Ethereum account and the identity itself are deterministically created using this mnemonic, a user can recover their full identity, including their Ethereum account and wallet using just this mnemonic.

AraID is designed to support any account backed by public key cryptography. Thus, it is agnostic to the types of cryptocurrency accounts it can support, and it can easily be associated with cryptocurrency wallets of any kind.

2.2 Decentralized Content Delivery Network (DCDN)

DCDN is Ara's solution to scalable, decentralized hypermedia and digital asset distribution. At it's core, DCDN is composed of a network of Ara File Systems (AFSs), CFS implementations which house content and their associated metadata.

2.2.1 Ara File System (AFS)

AFS is a flavor of CFS tailored to meet the specific needs and goals of Ara. AFS leverages two existing partitions that CFS implements, the `/home` and `/etc` partitions. These partitions, implemented as a subset of the Filesystem Hierarchy Standard (FHS) [8], are responsible for the raw binary data and the content metadata, respectively. The `/home` partition can only be accessed after a user has purchased or been granted access to the AFS's content, whereas the `/etc` partition containing the metadata can be accessed regardless of content ownership. The AFS owner may define a schema for the metadata so that it can be parsed by a requester. The protocol does not enforce a strict standard as to how metadata should be structured, but we recommend Schema.org as a reference to existing paradigms to best support interoperability between decentralized services.

When an AFS is initially created, an AraID is created using a `BIP39` [5] random 12-word mnemonic phrase. The generated DID is used as the public key of the AFS, and the corresponding DDO's `authentication` property is amended to include the owner's DID. By doing so, an AFS's owner can be determined from resolving its DID.

An AFS is created for each piece of content introduced into the system. This can be a single file such a movie, or a collection of files such as a game. To cryptographically verify

content ownership, two sets of byte buffers are written to the Ethereum blockchain. The first are the `metadata.tree` entries, which represent the serialized merkle tree of the data contained within the data storage layer. The second is the `metadata.signatures` file, containing signatures of the serialized tree's root nodes.

2.2.2 Token Usage

Initially, owning Ara tokens with respect to DCDN grants the following capabilities:

1. The ability to purchase and download content within the network.
2. The ability to participate in P2P file delivery and earn rewards on any content by submitting an Ara deposit that acts as a hold.

2.3 Ara Protocol Suite

The following subsections define the core protocols of the platform, describe each part of the system in detail, and explain the interoperability between them.

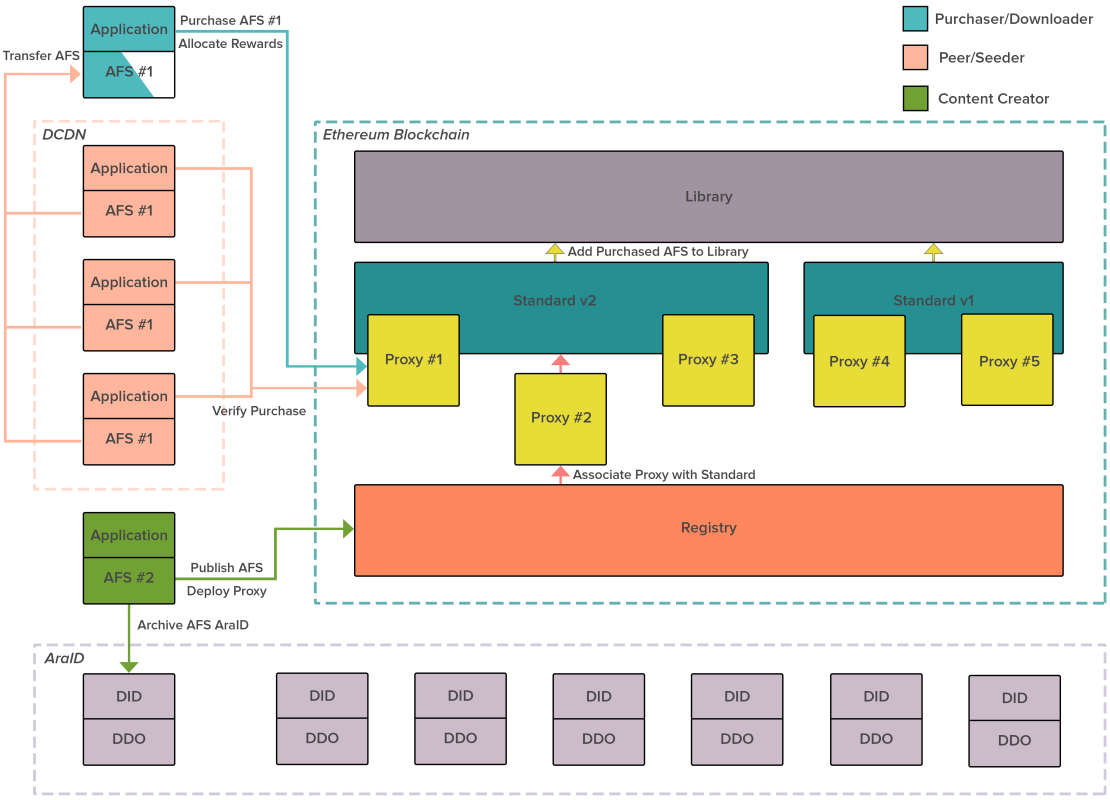

Figure 2: *Illustration of the Ara Protocol*

2.3.1 Rewards and Incentives

Traditional peer-to-peer file sharing systems, such as BitTorrent, rely on altruistic behavior and lack effective incentives [7] for peers to upload as much to the network as they download, creating an imbalance where *leechers* (users downloading a distributed file) can easily dominate a *swarm* (all peers downloading or uploading a distributed file). This imbalance, a form of The Free Rider Problem [6], is undesirable in a healthy network since leechers often benefit from the network at the expense of others without offering anything in return. Ara implements a reward system as an incentive mechanism to mitigate this network inefficiency and increase file availability by applying a small cost to each download and distributing that cost as a reward to every peer who seeds the download. Depending on the payment model, this cost could be baked into the total cost of the content as a reward allocation; for "free" content, this cost can replace traditional ads or data collection (how users pay for "free" content today). Since each download becomes a source of rewards, over time, network participants stand to earn much more from rewards then they pay up front for the download.

2.3.2 File Delivery

File Delivery is the main mechanism for upholding Ara's content delivery network and for participants to earn rewards. Ara's file delivery protocol begins with a four-step handshake.

1. Alice, the content requester, broadcasts a download request for a piece of content over some network discovery protocol (CFSNet implements several strategies, including mDNS and BitTorrent).
2. Bob, a license validator and content deliverer, receives the broadcast and responds with his file availability.
3. Alice selects peers from the pool (the swarm) of responses and sends a message with her intermediate public key along with her DID.
4. Bob cryptographically verifies Alice's message and that she has purchased a license to the underlying AFS.

Once the handshake is complete, the file transfer begins.

2.3.3 Smart Contracts

Ara is initially launching on Ethereum mainnet, where smart contracts will serve as a central component of Ara's protocol suite. These smart contracts mediate and facilitate interoperability between DCDN, AraID, and the application layer, ensuring that all non-transient properties and entities on the platform are registered on the Ethereum blockchain, including:

- Published Content
- Purchases
- Rewards
- Ara Balance

Ara's smart contract architecture was designed with security and modifiability in mind. In order to support evolving conceptions of how AFSs should be sold and purchased, how rewards should be handled and distributed, and how payments should be processed and routed within the system, Ara deploys a `Proxy Contract` for each published AFS. `Proxies` are deployed through a `Registry Contract`, where they are associated with a specific version of an `AFS Standard`, which defines the business logic for AFSs. Whereas deploying the entire `AFS Standard` for each AFS would be costly and difficult to update – the AFS would in essence be stuck with a specific standard – proxy architecture allows a single `Proxy` to be deployed for the lifetime of an AFS and upgraded by changing which `AFS Standard` version it references. Proxy architecture also ensures that only registered `Proxy` addresses (i.e., valid AFS content) can be added to user libraries in the `Library Contract`.

AFS Standard

The `AFS Standard` enables AFSs to have a defined, structured, and self-contained presence on the Ethereum blockchain. It includes methods for purchase and rewards, as well as methods for storing the `tree` and `signatures` files from the `metadata` SLEEP register. The `metadata` SLEEP register stores metadata about the content within an AFS, including filenames, sizes, and permissions, while the `content` SLEEP register stores the raw binary contents of the files. Within the `metadata` register, the `tree` file represents the serialized Merkle tree that makes up the data in the `content` register, and the `signatures` file stores the signed roots of the serialized tree. Whereas with CFS, these files would be stored on and read from disc, with AFS, they are written to and read from the Ethereum blockchain. Since AFSs communicate with this standard through their own `Proxy`, many different `AFS Standards` can coexist, allowing content creators to choose the standard that best suits their needs.

The use of `Proxies` separates logic from storage, where the `AFS Standard` serves as the logic layer for any AFS using that version of the standard, and each `Proxy` serves as the storage layer for a single AFS. At a minimum, `AFS Standards` must implement an `AFS Standard` abstract class which enforces the implementation of pricing, purchase, rewards, and storage functions.

For the most basic (and default) `AFS Standard`, prices can only be modified by the `owner` of an AFS. Upon purchase, this price is transferred from the purchaser's Ara wallet to the owner's Ara wallet. The base `AFS Standard` enforces support for reward budgets, which must be submitted prior to download. Once the download is complete, the budget is allocated between participating peers, who can then redeem their rewards granted they have not withdrawn the balance hold required to do so.

At the discretion of content creators, `AFS Standards` can also support a multitude of customizable commerce controls:

1. **Royalties**: Purchases can be customized to distribute proceeds amongst many different Ara accounts by percent breakdown.
2. **Bulk Purchases**: Prices can be tiered based on quantity purchased.
3. **Resale Conditions**: Purchased content can be resold a number of times for at least a minimum resale price as specified by the content creator.
4. **Ownership Transfers**: The owner of an AFS can readily transfer ownership to another Ethereum address.
5. **Pre-Orders**: Content can be purchased before it is available for download. Purchasers can submit a reward budget ahead of time so that they can begin downloading an AFS as soon as it is available.
6. **Scarcity**: Content creators can define a maximum number of sales for an AFS, after which the AFS becomes `unlisted` and unavailable for purchase.

These commerce controls provide content creators of all kinds with the power to define their own business and revenue models tailored for their needs to their exact specifications. The controls can be combined to form interesting new models which would be difficult to implement via traditional means. For example, someone who enjoys remixing music can purchase the tracks from the original artist with defined resale conditions and minimum resale prices, remix the songs, and sell the remixed versions while still paying out the original artist. Whereas previously combining these types of controls would require exorbitant amounts of time, capital, and legal involvement – serving as significant barriers to entry for smaller content creators – they are now available to everyone for free.

Registry

As part of the proxy architecture, the `Registry Contract` serves two main functions:

1. It serves as a `Proxy` factory
2. It tracks all `AFS Standard` versions

When an AFS is first published, the `Registry` deploys a `Proxy` for that AFS and establishes a relationship between it and a specified `AFS Standard`. The `Proxy` consults the `Registry` for the address of its respective `AFS Standard` whenever it is called and delegates the call to that address, where it is processed and returned to the `Proxy`.

Library

The Ara network leverages the `Library Contract` to create a canonical source of truth for AFSs that a user has access to, whether purchased or otherwise. When content is purchased, the `purchase` function in the `AFS Standard` automatically adds the AFSs DID to the purchaser's library in the `Library Contract`. This allows any service that requires information about a user's library to query the contract for that information. The AFS DID that is stored on the blockchain allows any service to resolve to the underlying content. The `Library` enforces that only registered `Proxies` can add their corresponding AFS to a user's library, ensuring that no one can tamper with another user's library without their consent.

3. Future Development

3.1 Modules

The Ara platform is fundamentally agnostic to the types of distributed services that can run on top of it. Modules are distributed and/or decentralized services which implement the Modules APIs and can be used interchangeably within the platform. Similar to the way the ERC-20 Token Standard allows any token on Ethereum to be re-used by other applications [14], the Modules APIs allow all distributed services implementing the interface to be used throughout the platform. This facilitates the formation of a developer community dedicated to building an ecosystem of distributed services that each have the ability to utilize Ara's rewards, purchases, and payments systems, in essence forming a rewardable distributed services economy. Modules that seek to utilize the rewards system are expected to implement an additional smart contract API in which they define their own rewarding mechanism and methodology. Each Module smart contract stores its own respective reward allocation accumulated through use of the distributed service and distributes it accordingly.

3.2 Ara Name System (ANS)

ANS, much like the Domain Name System (DNS) [9], is a decentralized way to register, query, invoke, or revoke certificates within the Ara network. While DNS sits as part of the application layer of the internet—resolving human-readable URIs to underlying IP addresses so that a user agent (e.g., web browser) can obtain and render the requested content—ANS resolves human-readable names to DIDs for ultimate resolution to DDOs. ANS uses the identity archiver and resolver under the hood for the second phase of resolution to provide DDOs from DIDs. ANS essentially is both an archiver and resolver for human-readable URIs. An example application of ANS could be providing DDOs from hostnames, in the context of a web browser built on top of Ara.

To discern between the various record types, each record stores a `TYPE` resource record, similar to how DNS classifies its own records [17]. The `TYPE` field is represented by a numeric value, which allows other types of records to be stored in ANS in the future. The following table describes the record `TYPE`:

TYPE	Value	Description
USR	00	User
PCT	01	Published Content

Each supernode hub that comprises ANS runs a HyperDB instance [2]. A distributed, highly-scalable database like HyperDB provides several features that make it suitable for a system like ANS. The first is HyperDB's use of tries: search trees where each node is a prefix to its child nodes. By storing names with tries, we can guarantee that even with thousands of entries in the database, lookups are inexpensive and quick. Lookups in tries are `O(n)` where `n` is the length of the key being searched. HyperDB also makes use of vector clocks, which track the causality of events within a distributed system to prevent cases where nodes become desynchronized [10].

4. Acknowledgements

Special thanks to Kevin Faaborg, Andrew Grathwohl, and Brandon Plaster for their contributions.

Acronyms

AFS Ara File System. 3–5, 9

ANS Ara Name System. 10

CFS Conflict-Free File System. 4, 5

CFSNet Conflict-Free File System Network. 4

DCDN Decentralized Content Delivery Network. 3–5

DDO DID Descriptor Object. 4, 5, 10

DID Decentralized Identifier. 4, 5, 9, 10

DNS Domain Name System. 4, 10

DPKI Decentralized Public Key Infrastructure. 4

FHS Filesystem Hierarchy Standard. 4, 5

PII Personally-Identifiable Information. 4

SLEEP Syncable Ledger of Exact Events Protocol. 4

References

[1] Code for Science Buus, Ogden. Sleep - syncable ledger of exact events protocol. https://github.com/datproject/docs/blob/master/papers/sleep.pdf, Aug 2017.

[2] Mathias Buus. Hyperdb. https://github.com/mafintosh/hyperdb, Aug 2017.

[3] Cisco. Cisco visual networking index predicts global annual ip traffic to exceed three zettabytes by 2021. https://newsroom.cisco.com/press-release-content?type=webcontent&articleId=1853168, Jun 2017.

[4] Stewart Clarke. Piracy set to cost streaming players more than $50 billion, study says. http://variety.com/2017/tv/news/piracy-cost-streaming-players-over-50-billion-1202602184/, Oct 2017.

[5] Palatinus et al. Mnemonic code for generating deterministic keys. https://github.com/bitcoin/bips/blob/master/bip-0039.mediawiki, Sept 2013.

[6] Russell Hardin. The free rider problem. https://plato.stanford.edu/entries/free-rider, May 2003.

[7] Ahamad Jun. Incentives in bittorrent induce free riding. https://disco.ethz.ch/courses/ws0506/seminar/papers/freeriding_incentives.pdf, Aug 2005.

[8] The Linux Foundation LSB Workgroup. Filesystem hierarchy standard. https://refspecs.linuxfoundation.org/FHS_3.0/fhs-3.0.pdf, Mar 2015.

[9] Paul Mockapetris. Domain names - implementation and specification. https://tools.ietf.org/html/rfc1035, Nov 1987.

[10] Multiple. Vector clock. https://en.wikipedia.org/wiki/Vector_clock.

[11] Markus Sabadello. A universal resolver for self-sovereign identifiers. https://medium.com/decentralized-identity/a-universal-resolver-for-self-sovereign-identifiers-48e6b4a5cc3c, Nov 2017. Accessed on 2018-04-24.

[12] Stephen E. Siwek. The true cost of sound recording piracy in the us economy. https://www.riaa.com/wp-content/uploads/2015/09/20120515_SoundRecordingPiracy.pdf, Aug 2007.

[13] Rebooting the Web-of Trust. Decentralized public key infrastructure. http://www.weboftrust.info/downloads/dpki.pdf, Dec 2015. Accessed on 2018-04-19.

[14] Buterin Vogelsteller. Erc-20 token standard. https://github.com/ethereum/EIPs/blob/master/EIPS/eip-20.md, Nov 2015.

[15] W3C. Decentralized identifiers. https://w3c-ccg.github.io/did-spec, Apr 2018. Accessed on 2018-05-01.

[16] Music Business Worldwide. Why does the riaa hate torrent sites so much? https://www.musicbusinessworldwide.com/why-does-the-riaa-hate-torrent-sites-so-much/, Dec 2014.

[17] Inc ZyTrax. Dns resource records (rrs). http://www.zytrax.com/books/dns/ch8/, Oct 2015.

Appendices

I. Mature Ara Platform Token Economics (Draft)

Overview

Traditional cloud services have risen to prominence due to the flexibility, agility, and cost savings they afford over purchasing and managing in-house infrastructure. Many cloud services use notoriously complex and obscure pricing models that involve long-term commitments and contracts—often individually negotiated per customer—directly undermining the flexibility and agility they were meant to provide in the first place [1]. In recent years, P2P CDNs have begun to gain traction, with emerging SaaS's touting hybrid solutions for video delivery. While these SaaS's established a highly scalable solution with a simpler pricing model by leveraging the machines of video viewers, they maintain several sources of centralization, namely through their business model and their attribution of users as "second class citizens", wherein the utilization of their machines occurs without their consent or financial compensation. Ara brings it one step further by rewarding network participants for the utilization of their machines.

In order to supplant existing classical cloud infrastructure costs, the Ara platform implements a native protocol utility token: the Ara token. This token can be used on the Ara platform to create cryptoeconomic incentives for healthy and honest network behavior, to enable more direct engagement between content consumers and creators, as well as to encourage adoption of the platform. The Ara token can be seen as an encapsulation of the value members of the network provide, with each token rewarded representing a marginal increase in network utility.

The Ara Token

Distributed services built using Ara's SDKs, known as Modules, can be bought, sold, requested, and fulfilled all on the Ara network. Module tasks are outsourced to peers in the network who, upon successfully completing them, can be compensated with Ara tokens. In order to promote an open and competitive market, Ara allows service requesters to define reward allocations, or bounties, for services they request and service providers to define a minimum bounty to accept. Similar to Amazon Mechanical Turk, a marketplace for crowdsourcing tasks that require human intelligence, Ara creates a marketplace for outsourcing distributed compute or networking tasks. Modules can be one-off on-demand tasks, such as distributed transcoding, or they can be continuously recurring services, such as a P2P multiplayer game server.

Function

The Ara token can be used throughout the network in a variety of ways.

- For consumers, Ara tokens can be used to make any sort of purchase, ranging from digital content for enjoyment to new Modules to participate in
- Service requesters can use Ara tokens to initiate job requests and to set bounties for the successful completion of those jobs
- Service providers can deposit Ara tokens as a commitment to fulfill a task in return for rewards (the deposit can be returned upon request, but the provider will no longer be able to earn rewards)
- Developers can use Ara tokens to deploy new Modules into the network

Since each of these roles heavily overlap, a service provider can use the same Ara tokens earned as rewards through fulfilling a task to purchase a new piece of content, just as a developer can use Ara tokens earned through Module purchases to initiate a new job request.

Market Dynamics

Because members of the network have full agency in deciding which tasks or services they participate in, the network forms a free-market in which economic equilibriums emerge. Each Module will likely have its own economics governed by the behavior of the requesters and providers for that specific Module's jobs. For example, distributed video transcodes might be a characteristically urgent job for video producers, resulting in price inelasticity of demand for distributed video transcodes (i.e., video producers are relatively indifferent to how much they might need to reward the service providers). Thus, a seller's market forms in which distributed transcode service providers have the advantage in determining reward allocation, driving the bounty up. Similarly, a P2P game server Module may be highly requested but have relatively few service providers. Again, a seller's market forms and the bounty is driven up. On the other hand, a distributed machine learning Module might be requested by few people but have many eligible service providers. Due to the relatively low demand, many providers will miss out on opportunities if they set their minimum bounty requirement too high. A buyer's market forms, and the bounty is driven down.

It is important to note that it is not a requirement for Modules to set bounties, and there is no standardized model for how bounties should be setup. The goals are to align the incentives of service requesters and service providers, to support all types of incentive models, as well as to accommodate the varying fixed costs for infrastructure and networking capabilities across the world.

Incentive Structure

To better understand how this model supports an alignment of incentives between service requesters and service providers, we derive a general description of the economic interests of both parties. Service requesters want their requests to be fulfilled for the lowest cost, while service providers want to optimize for the greatest number of highest paying services. In other words, it is in the best interest of both

requesters and providers to maximize network utility as long as both agree on a price. Thus, the services that best balance bounty and work are most likely to be fulfilled, creating market pressure that promotes both competitive bounties and innovation in distributed service design to improve efficiency.

The variety of distributed services that can run on the platform necessitates flexibility in determining unit-of-work-rewarded (*UWR*, the base unit of provable work by which bounties are split up and rewarded) and bounty model (the conditions that govern how bounty is paid). A P2P multiplayer game server might use number of requests served as its *UWR*, and a game developer invoking that server Module might decide that a subscription-based, recurring bounty model makes the most sense. On the other hand, a distributed transcoding service might use number of bytes transcoded per minute as its *UWR*, and a video producer might payout a bounty per transcode.

Service providers can stake Ara tokens in order to participate and earn rewards. Like with bounties, service requesters can determine a minimum stake, if any, that providers must deposit in order to engage in the service. The minimum stake value should be indicative of the level of commitment the service requires and is returned, along with the bounty, once the service is successfully completed. As part of determining a *UWR*, services must also define a proof for verifying its fulfillment.

Alternatively, service providers can set a subscription fee for dedicating their resources to a service. These dedicated providers are known as *supernodes* and their stakes are escrowed in a smart contract until the subscription ends. Because supernodes tend to be more reliable than their non-dedicated counterparts, they can control market dynamics based on how they set their subscription fees. Supernodes in Bogotá might charge more than supernodes in Los Angeles due to higher hardware and internet costs.

Network Effects

Introduction of new content to the network involves the invocation of DCDN supernodes—Ara nodes dedicated to content redundancy and availability. Assuming the bounty for a single file is constant, the effect of the marginal increase in file availability from a single peer sharing that file on the potential reward earnings from that file for any peer in DCDN (and all currency-based P2P file-sharing systems with fixed incentives [2]) can be modeled using a submodular set function, which can be intuitively thought of as a function describing diminishing returns. Due to this property, DCDN supernodes can employ a subscription bounty model to counteract the increasing opportunity cost of hosting content as its availability increases.

Content publishers can invoke and subscribe to as many or as few supernodes in geographic locations of their choosing on a per content basis. Publishers, then, have full freedom in deciding the extent to which they want their content to be readily available globally. This enables support for the large media distributor who wants to invoke all available supernodes worldwide to support a global audience, while also enabling support for the indie content creator who identifies their primary audience as predominantly European and decides to prioritize European supernodes. Content publishers also determine the reward allocation for each content download. Thus,

there exists an optimal reward allocation and supernode distribution to achieve the desired level of participation (i.e., file availability).

References

[1] Enterprise Strategy Group (2015, June), *Price Comparison: Google Cloud Platform vs. Amazon Web Services*, https://cloud.google.com/files/esg-whitepaper.pdf

[2] M. Salek, S. Shayandeh, and D. Kempe, *You Share, I Share: Network Effects and Economic Incentives in P2P File-Sharing Systems* https://arxiv.org/pdf/1107.5559.pdf

II. DCDN Cost Analysis by Lester Kim

Introduction

In order to calculate the streaming costs for a potential partner, we need to know how much data B (in bytes) they need to deliver to consumers per unit of time T (in seconds). Let us have N groups of uploader nodes where $N \in \mathbb{N}$. $\forall n \in \{1, ..., N\}$, the average bandwidth of group n is b_n (in bytes/second per node). Let q_n be the quantity of group n nodes. Let $\mathbf{b} = [b_1...b_N]^\intercal$ and $\mathbf{q} = [q_1...q_N]^\intercal$. Thus, the amount of bytes delivered per second constrained by $\frac{B}{T}$ is

$$g(\mathbf{q}) = \mathbf{b} \cdot \mathbf{q} = \frac{B}{T}. \tag{1}$$

We want to find the optimal $\mathbf{q}^*$ to minimize the cost of distribution $C(\mathbf{q})$. If $\mathbf{p} = [p_1...p_N]^\intercal$ where p_n is the price per node for group n, we have

$$C(\mathbf{q}) = \mathbf{p} \cdot \mathbf{q}. \tag{2}$$

Uploader's Profit Maximization

To determine $\mathbf{p}$, let us visit the behavior of a profit maximizing firm. Let f be the production function with energy input E (in kWh) and output q (in nodes). We model this production function as

$$f(E) = AE^\alpha \tag{3}$$

where A is the factor of production (nodes/kWh$^\alpha$) and $\alpha \in [0, 1]$ is the elasticity of production (percent increase in output over percent increase in input) [1].

Let P (in kWh/s) be the power increase when a node starts uploading data. This includes sending data with its network interface controller (NIC) but can also include the machine turning on (from either being off or in standby mode). If each node has power P, then for some E, a single node can run for $\frac{E}{P}$ seconds. However, given a time constraint T to complete the work, there must be $\frac{E}{PT}$ nodes. Thus,

$$A = \frac{D}{(PT)^\alpha} \tag{4}$$

where D is the total factor productivity [2] (in nodes).

Let p be the price of a node and p_E the price of energy (per kWh). The firm's profit function π is

$$\pi(q, E) = pq - p_E E. \tag{5}$$

Bandwidth cost is ignored because it is a fixed cost in the short-term when looking at seconds as opposed to months[1].

We want to maximize the firm's profit given an output requirement at least q;

$$\max_{q,E} \ \pi(q, E) \quad \text{s.t. } f(E) \geq q. \tag{6}$$

To solve this, we take our Lagrangian to be

$$\mathcal{L}(q, E, \lambda) = pq - p_E E - \lambda(AE^{\alpha} - q). \tag{7}$$

Taking partial derivatives and setting them to zero give

$$\frac{\partial \mathcal{L}}{\partial q} = p + \lambda = 0 \tag{8}$$

$$\frac{\partial \mathcal{L}}{\partial E} = -p_E - \lambda A \alpha E^{\alpha - 1} = 0 \tag{9}$$

$$\frac{\partial \mathcal{L}}{\partial \lambda} = q - AE^{\alpha} = 0. \tag{10}$$

Solving these first-order conditions gives

$$q^* = \left(\frac{\alpha A^{\frac{1}{\alpha}} p}{p_E} \right)^{\frac{\alpha}{1-\alpha}}. \tag{11}$$

Rewriting this (dropping the asterisk from q) gives

$$p = \frac{p_E}{\alpha} \left(\frac{q^{1-\alpha}}{A} \right)^{\frac{1}{\alpha}}. \tag{12}$$

This formula lets the creator know what p should be to get the desired number of nodes.

From (12), we find that the optimal revenue in terms of q is

$$pq = \frac{p_E}{\alpha} \left(\frac{q}{A} \right)^{\frac{1}{\alpha}}. \tag{13}$$

[1]Even with bandwidth included, its cost per second has the same order of magnitude as that of energy. In NYC, 50 MBps costs \$3 x 10^{-5}/second [3].

Distributor's Cost Minimization

Since the revenue for the firm is spending for the consumer (the creator), we can write (2) as

$$C(\mathbf{q}) = \frac{p_E}{\alpha} \sum_{n=1}^{N} \left(\frac{q_n}{A_n} \right)^{\frac{1}{\alpha}}. \tag{14}$$

The creator's cost minimization problem is

$$\min_{\mathbf{q}} \; C(\mathbf{q}) \quad \text{s.t. } g(\mathbf{q}) \geq \frac{B}{T}. \tag{15}$$

The Lagrangian is

$$\mathcal{L}(\mathbf{q}, \lambda) = C(\mathbf{q}) - \lambda(g(\mathbf{q}) - \frac{B}{T}). \tag{16}$$

The first-order conditions are

$$\frac{\partial \mathcal{L}}{\partial \mathbf{q}} = \frac{\partial C}{\partial \mathbf{q}} - \lambda \frac{\partial g}{\partial \mathbf{q}} = 0 \tag{17}$$

$$\frac{\partial \mathcal{L}}{\partial \lambda} = \frac{B}{T} - g(\mathbf{q}) = 0. \tag{18}$$

From (14), (4), and (1),

$$\frac{\partial C}{\partial \mathbf{q}} = \frac{p_E T \mathbf{P}}{\alpha^2} \circ (\mathbf{q}^{\circ(1-\alpha)} \oslash \mathbf{D})^{\circ \frac{1}{\alpha}} \tag{19}$$

$$\frac{\partial g}{\partial \mathbf{q}} = \mathbf{b} \tag{20}$$

where $(\mathbf{D}, \mathbf{P}) = ([D_1 ... D_N]^\intercal, [P_1 ... P_N]^\intercal)$. "$\circ$", "$^{\circ}$", "$\oslash$" are the Hadamard (entrywise) product, power, and division, respectively [4]. So $\forall m, n \in \{1, ..., N\}$,

$$\frac{P_m^\alpha q_m^{1-\alpha}}{b_m^\alpha D_m} = \frac{P_n^\alpha q_n^{1-\alpha}}{b_n^\alpha D_n}. \tag{21}$$

Thus,

$$b_m q_m = \left(\frac{b_m D_m P_n^\alpha}{P_m^\alpha b_n D_n} \right)^{\frac{1}{1-\alpha}} b_n q_n. \tag{22}$$

Combining (22) with (18) gives

$$\boxed{\mathbf{q}^* = \frac{B\mathbf{b}^{\circ -1}}{T\kappa} \circ (\mathbf{b} \circ \mathbf{D} \oslash \mathbf{P}^{\circ \alpha})^{\circ \frac{1}{1-\alpha}}} \tag{23}$$

$$\boxed{C^* = \frac{p_E T}{\alpha} \left(\frac{B}{T\kappa^{1-\alpha}} \right)^{\frac{1}{\alpha}}} \tag{24}$$

where

$$\kappa \equiv \sum_{m=1}^{N} \left(\frac{b_m D_m}{P_m^{\alpha}} \right)^{\frac{1}{1-\alpha}}. \tag{25}$$

Case: $\alpha = 1$

When $\alpha = 1$, (23) and (24) become

$$q_n^* = \begin{cases} \frac{B}{|\Upsilon| T b_n} & n \in \Upsilon \\ 0 & n \notin \Upsilon \end{cases} \tag{26}$$

$$C^* = \frac{p_E B P_n}{b_n D_n} \qquad \text{any } n \in \Upsilon \tag{27}$$

where

$$\Upsilon \equiv \left\{ n \in \{1, ..., N\} \;\middle|\; n = \arg\max_{1 \le m \le N} \frac{b_m D_m}{P_m} \right\}. \tag{28}$$

$\forall n \in \Upsilon$, each node in group n would deliver $\frac{B}{|\Upsilon| q_n^*} (= b_n T)$ of data and receive at least $\frac{p_E P_n T}{D_n}$ in compensation. However, there are multiple solutions to $\mathbf{q}^*$. For example, for any $n \in \Upsilon$, group n can take on all the work by employing $\frac{B}{T b_n}$ nodes.

Example

Let's work out an example in NYC where

$$\alpha = 1 \tag{29}$$
$$B = 1 \text{ GB} \tag{30}$$
$$N = 2 \tag{31}$$
$$p_E = \$0.2321/\text{kWh [5]} \tag{32}$$
$$T = 1 \text{ s} \tag{33}$$
$$\mathbf{b} = \begin{bmatrix} 100 \text{ MB/s} \\ 1 \text{ MB/s} \end{bmatrix} \text{ [6]} \tag{34}$$
$$\mathbf{D} = \begin{bmatrix} 1 \text{ node} \\ 1 \text{ node} \end{bmatrix} \tag{35}$$
$$\mathbf{P} = \begin{bmatrix} 200 \text{ W} \\ 2 \text{ W} \end{bmatrix} \text{ [7][8]} \tag{36}$$

to find examples of $\mathbf{q}^*$ and C^* for a creator. Then,

$$\mathbf{q}^* = \begin{bmatrix} 5 \text{ nodes} \\ 500 \text{ nodes} \end{bmatrix} \tag{37}$$
$$C^* \approx \$1.29 \text{ x } 10^{-4}. \tag{38}$$

This is 155 to 659 times (99.35% - 99.85%) cheaper than AWS Cloudfront's on-demand pricing ($0.020/GB - $0.085/GB) [9]. Each group 1 node would handle 100 MB whereas each group 2 node would handle 1 MB. Each node in group 1 and 2 would need more than $1.29 x 10^{-5} and $1.29 x 10^{-7}, respectively.

To put this in perspective, assume Netflix is a potential partner. In 2017, Netflix averaged more than 140 million hours of content watched per day [10]. On average, a Netflix video is one GB/hour [11]. On the Ara platform, the 51.1-exabyte [12] annual spend would only be $6.6 million/year ($0.2106/second). If we estimate Netflix's streaming cost to be $0.03/GB [13], we get $1.5 billion/year ($46.61/second). Using the Ara network would nearly quadruple Netflix's 2017 net income of $558.9 million [14]. (Manhattan has 1.66 million people [15] with 287,008 Netflix users[2] streaming 321.45 TB/day, 3.72 GB/s. That requires 3,721 group 2 nodes at a rate of $41.47/day).

[2]At the end of 2018 Q1, Netflix had 56.71 million U.S. subscribers and 125 million worldwide [16]. There are 328 million people in the U.S. [17], so proportionally, there are 287,008 Netflix subscribers in Manhattan.

References

[1] Wikipedia (2018, April 22), *Cobb–Douglas production function*, https://en.wikipedia.org/wiki/Cobb%E2%80%93Douglas_production_function

[2] Wikipedia (2018, June 5), *Total factor productivity*, https://en.wikipedia.org/wiki/Total_factor_productivity

[3] Spectrum (2017, December 29), *Broadband Label Disclosure*, p.2, https://www.spectrum.com/content/dam/spectrum/residential/en/pdfs/policies/Broadband_Label_Disclosure_Charter_122917.pdf

[4] Wikipedia (2018, March 10), *Hadamard product (matrices)*, https://en.wikipedia.org/wiki/Hadamard_product_(matrices)

[5] Electricity Local (2018, June 19), https://www.electricitylocal.com/states/new-york/new-york

[6] Wikipedia (2018, June 19), *Bandwidth (computing)*, https://en.wikipedia.org/wiki/Bandwidth_(computing)

[7] Energuide.be (2018, June 19), *How much power does a computer use? And how much CO2 does that represent?*, https://www.energuide.be/en/questions-answers/how-much-power-does-a-computer-use-and-how-much-co2-does-that-represent/54/

[8] R. Sohan, A. Rice, A. W. Moore, and K. Mansley, "Characterizing 10 Gbps Network Interface Energy Consumption," *The 35th Annual IEEE Conference on Local Computer Networks (LCN) Short Papers*, University of Cambridge, Computer Laboratory, July 2010, https://www.cl.cam.ac.uk/ acr31/pubs/sohan-10gbpower.pdf.

[9] Amazon Web Services Pricing (2018, June 19), *Amazon Cloudfront Pricing*, https://aws.amazon.com/cloudfront/pricing

[10] L. Matney (2017, December 11), "Netflix users collectively watched 1 billion hours of content per week in 2017," *Techcrunch*, https://techcrunch.com/2017/12/11/netflix-users-collectively-watched-1-billion-hours-of-content-per-week-in-2017

[11] K. Hubby (2017, May 23), "The surprising amount of data Netflix uses," *The Daily Dot*, https://www.dailydot.com/debug/how-much-data-netflix-use/

[12] Wikipedia (2018, June 20), *Exabyte*, https://en.wikipedia.org/wiki/Exabyte

[13] D. Rayburn (2009, July), "Stream This!: Netflix's Streaming Costs," *Streaming Media (June/July 2009)*, http://www.streamingmedia.com/Articles/Editorial/Featured-Articles/Stream-This!-Netflixs-Streaming-Costs-65503.aspx

[14] Netflix (2018, January 1), p.40, https://ir.netflix.com/static-files/20c3228d-bf1f-4956-a169-c8b76911ecd5

[15] Wikipedia (2018, July 5), *Manhattan*, https://en.wikipedia.org/wiki/Manhattan

[16] Statistica (2018, July 5), *Number of Netflix streaming subscribers in the United States from 3rd quarter 2011 to 1st quarter 2018 (in millions)*, https://www.statista.com/statistics/250937/quarterly-number-of-netflix-streaming-subscribers-in-the-us/

[17] United States Census Bureau (2018, July 5), *U.S. and World Population Clock*, https://www.census.gov/popclock/

III. Expected Ara Rewards Analysis by Lester Kim

Network Model

Let us represent the Ara network of nodes as a complete graph [1] $G = (V, E)$ where V contains N vertices with each vertex representing a node and E containing $\frac{N(N-1)}{2}$ edges with each edge representing a communication channel between two nodes. Let C be some collection of content (in our case, a set of digital entertainment files) that all the nodes want to have, and let the subset $S \subseteq V$ contain all the nodes that have C (i.e. $S = \{v \in V : C \in v\}$).

Let time $t \in \mathbb{N}$. At $t = 0$, $|S| = 1$, so there is only one $v_0 \in V$ that has content C; thus, there is only one vertex that can deliver a copy of C to other nodes in the network. Assume that all other $N - 1$ nodes want C, and v_0 has enough bandwidth to deliver C to only one node. From $t = 0$ to $t = 1$, $|S|$ increases from 1 to 2. In general, at time t,

$$|S| = \begin{cases} 2^t & 0 \leq t < \log_2 N \\ N & t \geq \log_2 N. \end{cases} \tag{39}$$

Note that $|S| = N$ starting at $t = \lceil \log_2 N \rceil$.

$\forall s \in S, s$ will deliver C to some $v \in V \setminus S$ only if v pays s an amount p. Let M be the network's total budget for entertainment delivery. Dividing this evenly by N nodes gives $p = M/N$.

At $t = 0$, the sole $v_0 \in S$ receives p from some $v_1 \in V \setminus S$. Then, at $t = 1$, $v_0, v_1 \in S$ each receives p from some $v_2, v_3 \in V \setminus S$. At any $t < \lfloor \log_2 N \rfloor$, each of the $|S| = 2^t$ nodes in S receives p from 2^t nodes in $V \setminus S$. At $t = \lfloor \log_2 N \rfloor$, $|S| > \frac{N}{2}$, so there are more suppliers than demanders of C. When that occurs, $N - 2^t$ nodes from S are randomly selected to deliver C. At $t = \lceil \log_2 N \rceil$, $S = V$.

In this model, v_0 earns at least

$$\frac{M \lfloor \log_2 N \rfloor}{N}; \tag{40}$$

v_1 earns at least $\frac{M(\lfloor \log_2 N \rfloor - 1)}{N}$; v_k earns at least

$$\frac{M(\lfloor \log_2 N \rfloor - \lceil \log_2 (k+1) \rceil)}{N}. \tag{41}$$

The greatest k such that v_k earns at least $\frac{M}{N}$ is when

$$\lfloor \log_2 N \rfloor - \lceil \log_2 (k+1) \rceil \geq 1 \tag{42}$$

which implies

$$\log_2 (k+1) \leq \log_2 \frac{N}{2}. \tag{43}$$

Thus, the maximum value of k to earn at least $\frac{M}{N}$ is $k = \lfloor \frac{N}{2} \rfloor - 1$. On average, each earns

$$\frac{M - \frac{M}{N}}{2^{\lceil \log_2 N \rceil - 1}} = \frac{M(1 - \frac{1}{N})}{2^{\lceil \log_2 N \rceil - 1}}. \tag{44}$$

The numerator is $M - \frac{M}{N}$ to exclude v_0's entertainment budget since it had C at $t = 0$. The denominator is $2^{\lceil \log_2 N \rceil - 1}$ because at $t = \lceil \log_2 N \rceil - 1$, $|S| = 2^{\lceil \log_2 N \rceil - 1}$, and at that point, S consists of all the nodes that have the possibility of earning rewards throughout this process. This means that there are $N - 2^{\lceil \log_2 N \rceil - 1}$ nodes that will not be able to earn rewards.

Example

Approximately 80% of Americans have computers with Internet access [2]. Since there are 327M Americans living in the US [3], there are (0.8)(327M) = 261.6M Americans with devices connected to the Internet. Assuming each has one device, let $N = 261.6$M. The annual US entertainment consumption is \$734B [4] [5]. Let's assume most of this expenditure for future years will be digital, but let's only include the budget of the 80% of Americans who have Internet access such that the spending among them is (0.8)(734B) = \$587B. Let 10% of the spending be for covering distribution costs. Then, $M = (0.1)(\$587\text{B}) = \$58.7\text{B}$. Then, $p = M/N = \$58.7\text{B}/261.6\text{M} = \$224.39$. By (44), the average annual earnings is \$437.35 per node. By (40), the most v_0 can earn is \$6282.87. Thus, this example illustrates that the initial peers who share content will earn the most rewards.

References

[1] Wikipedia (2018, June 19), *Complete graph*, https://en.wikipedia.org/wiki/Complete_graph

[2] C. Ryan and J. M. Lewis, "Computer and Internet Use in the United States: 2015," *American Community Survey Reports* U.S. Census Bureau, September 2017 https://www.census.gov/content/dam/Census/library/publications/2017/acs/acs-37.pdf

[3] World Population Review (2018, June 18) *United States Population 2018* http://worldpopulationreview.com/countries/united-states-population/

[4] SelectUSA (2018, August 22), *MEDIA AND ENTERTAINMENT SPOTLIGHT*, https://www.selectusa.gov/media-entertainment-industry-united-states

[5] Bureau of Labor Statistics (2017, August 29), *CONSUMER EXPENDITURES–2016* https://www.bls.gov/news.release/cesan.nr0.htm