Contract networking for crowdsourced connectivity

Emmanouil Dimogerontakis[*], Leandro Navarro[*], Mennan Selimi[*†], Sergio Mosquera[*] and Felix Freitag[*]
[*] Universitat Politècnica de Catalunya, BarcelonaTech, Spain
[†] Max van der Stoel Institute, South East European University, North Macedonia

***Abstract*—Universal connectivity is still unavailable or expensive for half of the global population, despite being critical for social participation. The deployment of crowdsourced networking infrastructures creates an opportunity for local development, where anyone can deploy a new device. In such infrastructures connectivity offer can expand incrementally and be sustainable through investment and fees resulting from the demand and consumption of content and services, including Internet access, that compensate the cost of the underlying network. While routing coordinates network data flows, economic flows can be coordinated by smart contracts built over a local distributed ledger. We define crowdsourced networks, the concept, architecture, and implementation using a local Ethereum PoA blockchain with Solidity smart contracts that compensate the data traffic contribution and consumption recorded by a traffic monitoring system, on a wireless mesh network. The prototype software has been validated in a controlled mesh network environment. Functional tests show its ability to account and route economic flows with small resource consumption, and therefore confirms these networks can develop organically by the addition of consumer and provider participants to reach the typical scale of most wireless mesh access networks and deliver networking services that aim to be socially and economically sustainable.**

***Index Terms*—smart contracts; service pricing; mesh networks; reliable monitoring; crowdsourcing; blockchain.**

I. Introduction

Nearly half of the global population is without Internet access according to ITU [1]. Considering the wider role the Internet plays today in providing access to crucial services like education, health, e-governance, social networks, etc, enabling universal access to the Internet is a major challenge. Several community networks have appeared to enable wider community connectivity across rural and underserved areas.

Unlicensed wireless is a key solution to the last mile access problem as evidenced by many community network initiatives across the world [2]. In a mesh network, each node (router) is capable of relaying data for others, all nodes cooperate in the distribution of data throughout the network to the mutual benefit of its participants. Each participating node expands the reach, throughput and resilience of a network. An ad-hoc mesh network can grow to provide shared connectivity in a region and at cheaper cost than a centralized topology [3].

Mesh networks face the challenge to scale in absence of an economic sustainability model. The main challenges are around trust among peers and ensuring the capacity and economic sustainability of this collective effort, balancing resource contribution and consumption. For instance, we consider the *economic compensation system* used in Guifi.net [4], a bottom-up, citizen-driven community mesh network with the aim of a free, open and neutral telecommunications network based on a commons model [5]. The idea of an economic compensation system is to balance the cost of resource contribution and its consumption. Currently this is done manually: each participant declares its costs and consumption (traffic) and an arbiter validates the claims by cross checking it with their own network traffic measurements. There is room for error from data loss or tampering, or intentional false claims by a participant. Truthful and complete data and the application of compensations is key for the economic sustainability of a network, the return of investments and maintenance of its infrastructure.

Fintech [6] technology, such as distributed ledgers, blockchain, smart contracts for trust, digital currencies, token payments allow to implement economic compensation mechanisms for the exchange of value [7] [8]. This technology enables disintermediation, with direct peer transfers or crowdfunding investment campaigns automated through smart contracts.

Therefore, we explore the feasibility of an automated mechanism where diverse participants, resource providers and consumers, can pool resources with the confidence that the contribution and consumption of resources is accounted fairly, and that these calculations and value transfers are automated, irreversible, inexorable, transparent, shared across different participants, to avoid the cost, delays, errors and potential mistrust from manual accounting and external payments.

We present a model (Section II), design (Section III) and evaluation (Section IV) of blockchain-enabled crowdsourced mesh networks. From an economic perspective, consumers provide funds in exchange for connectivity-based services, such as local services or Internet access, and providers get a return on their investment in the infrastructure required to supply such services [9]. Both sides should achieve a positive utility balance, i.e., consumers get social benefits of the connectivity and providers get a return on investment. To this end, consistent data collected about traffic feeds a set of smart contracts that deal with the economic compensation, comparable to the pricing and operation of dynamic electricity markets. We fully implement the system on real hardware and showcase in Section IV the efficiency, feasibility of deployment, and effectiveness in an controlled mesh network. Section V discusses related and future work, concluding in Section VII.

II. Conceptual Architecture

A mesh network island in an area, provides connectivity services such as Internet access, content and local services. We have multiple network devices, such as Access Points (AP), Routers (R), servers (S), and Internet Gateways (GW). Consumers (users) can connect to network services and the Internet through AP devices in various locations, interconnected by several intermediate mesh routers. Servers deliver local services, and gateway nodes deliver Internet connectivity.

Each device in a mesh has a service cost, that corresponds to the expected return of an initial investment on network capacity (CAPEX) such as devices, links, servers and its maintenance and operation (OPEX). Service cost can fluctuate according to usage (related to traffic peaks and congestion). Similarly to

DOI 10.1109/DAPPS49028.2020.00016

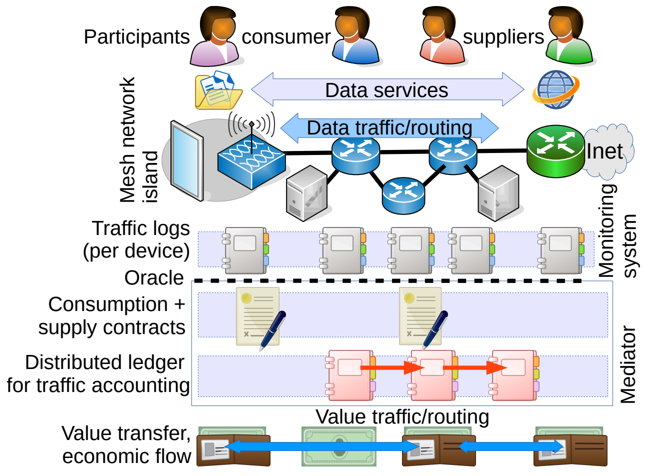

Figure 1. Conceptual architecture of the proposed system

the electricity market, a market maker can find optimal retail service prices (e.g. the MBh equivalent to the kWh) and the optimal allocation of connectivity demand and supply that balance the long-term and short-term costs of the infrastructure. Each participating device is rewarded by payments from consumers. Such a system can help accommodate participation, capacity, growth, variability and sustainability. While the routing tries to optimize the allocation of capacity to traffic, the price optimization can compute retail tariffs for traffic and services offered to consumers (price plans).

The main stakeholders are the *consumer* with a *user device* that obtains connectivity by connecting to a retailer AP under a service contract (price plan), and a *provider*, that owns and provides devices $\{AP, R, GW, S\}$ that supply connectivity or services in exchange of an (economic) compensation.
The main components of the model are the following:

– *Mesh Network Island:* consisting on the nodes, the wireless physical connection between them, and the stack of network protocols that enable their connectivity. A Node-DB contains information relevant to the mesh about all the network nodes.
– *Monitoring System:* measures the resources consumed, forwarding traffic, and supply economic data. The traffic corresponds to bytes forwarded per device over a time period, collected from the operating system of each device. The quality of the data, its completeness and consistency across all devices in a network island, determines the accuracy and confidence in the economics of a network.
– *Oracle:* a replicated service that supplies reliable data to the smart contracts in the mediator that compute the economic compensations.
– *Mediator:* the core agent that, in every mesh island, is responsible for matching the demand and the supply of the offered service as well as maintaining the balances among the different participants, based on the monitoring information supplied by an oracle service. The Mediator applies the logic to compute the distribution of economic value, the *settlement* to providers in exchange of services, and determines a fair price for retail and aggregate traffic for each usage cycle.

We argue that smart contracts can automatically facilitate, verify and enforce the negotiation or performance of these mechanisms in a traceable and irreversible way. That can result in value transfers in the form of payments of token units. In the next section, we explain how a blockchain (i.e., smart contracts) can help the implementation of the economic interaction between the components mentioned in the architecture.

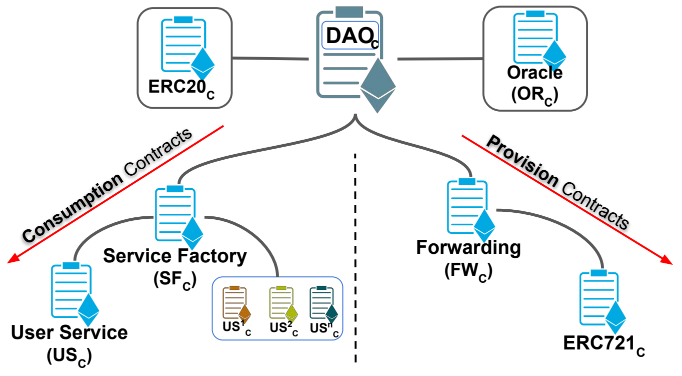

Figure 2. Smart Contract Ecosystem

III. The Implementation

Our proof-of-concept implementation of the aforementioned architecture focused on the use case of the Internet access service. Nevertheless, it is designed to incorporate other types of network services. A set of smart contracts, together with the necessary tools, implement the mediator between the service supply, such as Internet access, in a mesh network and consumer demand, using client devices such as a mobile or a computer. Acting as a mediator, the platform proposes to the clients smart contract templates (retail price plans) at price points that reflect the network costs from every participant, and avoid negative utilities for each one and all, similar to the case of electricity retail spot prices [10]. This means adjusting prices and payments according to the history and trends of demand and supply, and therefore optimize revenue sharing across all. Optionally, it may need to keep reserve funds that can be used to even out variability across different payment cycles. In this section, we present the core set of the implemented smart contracts, and we describe the most important processes that are performed based on these smart contracts.

A. Smart Contracts

The set of Consumption and Provision based smart contracts, as depicted in Figure 2, are implemented in each instance of the platform, deployed in every local and autonomous mesh network. The following smart contracts are implemented:
– *Decentralized Autonomous Organization (DAO_c):* The local DAO contract allows the storage and management of several main global variables and constants, e.g. consumer prices. It is important to point out here that in the DAO_c and where necessary we have implemented a trivial access policy according to which only the account who deployed the contract is allowed to perform transactions that execute critical code. This contract is, also, used as a registry pointing to the correct instances of the remaining deployed contracts.
– *Fungible connectivity tokens ($ERC20_c$):* An ERC20-based contract [11] implements the tokens used for exchanging value between the participating entities.

– *Non-fungible device ownership tokens ($ERC721_c$):* Network resources, such as links, network devices, or servers can be represented as tokens: registering a new device in a network results in a new title/attestation of ownership, that can be represented as a new ERC721 token, a deed associated to each unique device through its serial ID and current owner. The inventory of devices, each tokenized as an ERC721 token in the local blockchain, includes ownership information. Therefore, payments to devices can be directed to each device owner. Each provider device token maintains attributes related to the payment system (*ownerAddress*, *forwardingPrice*, *forwardedBytes*), together with a *URI*, a record of the device on a node database, that points to detailed information in the NodeDB of the mesh network island. The $ERC721_c$ is used for client devices as well, providing them an identity in the local network.
– *Oracle (OR_c):* This contract is responsible for the trusted communication between the blockchain environment and the outer world. It intermediates to retrieve information about the nodes either from the monitoring server or the NodeDB.
– *Service Factory (SF_c):* This is the root contract that allows the creation of User Service Agreement contracts for service provision to connectivity consumers, under various predefined pricing policies. In our current implementation we assume the existence of single (non differentiated) consumer price level. Nevertheless, this contract factory allows the integration of other policies that could consider the history and trends of demand and supply, as well as different payment schemes.
– *User Service contract (US_c):* This contract, representing an instance of a User Service Agreement, ensures the consumer and provider fulfill the terms of contract, monitoring the value transfer and the usage of network resources occur at the agreed rate for the agreed amount of data. The pricing policy offered in the current implementation is based on an agreed *Token/MB* rate. This contract is a starting point for developing more sophisticated agreements, such as considering duration (e.g. a day), or quality (e.g. minimum rate, or maximum latency).
– *Forwarding Contract (FW_c):* This contract implements the economic compensation of providers. Its address acts as a reserve fund, since it receives the tokens collected from the consumers. Periodically, based on device traffic reports, the contract generates a device payment vector and uses the reserved token to compensate the providers or device owners. As explained later, in case of deficit, it calculates fair forwarding prices dividing deficit proportional to the data forwarded.

B. Processes

The main processes that define the activity of networks is the bootstrapping or initialization of a local network, a new island of coverage and service, in a new location. Once a local network is instantiated (that requires at least one server S), new devices will be added to the infrastructure and registered as economic agents, with at least an access point (AP) to allow access, and a gateway (GW) to provide Internet connectivity. Additional routers R, AP, GW, S can expand coverage, capacity and services. Once a network island is operational, consumers can access the network through a US_c on a retailer AP that provides Internet connectivity in exchange of service fees. The fees paid by consumers must then be distributed proportionally to all network devices that contribute to deliver services, and subsequently to the device owners to compensate and recover their investment, maintenance and operation.

Bootstrapping a new local network: Bootstrapping a new local network island refers to the self-starting processes (smart contracts) that need to be deployed in sequential order. Each network island should have one or several server nodes. An Ethereum Proof of Authority (PoA) is initialized, with an initial local Ether supply. Contracts are deployed in the following order; initially the DAO_c contract is deployed setting the basic variables, followed by the Oracle contract, since it is the basis for other contracts that need to communicate (interact) with the non-blockchain environment. Finally, the $ERC20_c$ and $ERC721_c$ tokens are deployed, followed by the rest of the contracts.

Registering (Minting) a new device: Each device introduced in the network has to be *minted* first as an ERC721 token. The procedure for minting a device is as follows: the device owner sends a request to mint the corresponding ERC721-type token. The device ownership contract ($ERC721_c$) performs a check if the node already exists, and if not it proceeds to further to the Oracle contract. The Oracle verifies the owners' wallet address by means of a web request to the device's node agent, that is designed to prove that the owner has access to the device he tries to mint. After the ownership is verified, the Oracle retrieves the node details from the NodeDB and calls the device ownership contract callback to finalize node minting. Finally, the $ERC721_c$ performs the internal minting of the new token. As an outcome of the above process, that device is registered and obtains an identity in the local network.

Establishing an Internet connection: Establishing a network connection, requires clients to connect first to a Wi-Fi access point portal (AP) and get an local network IP. As mentioned previously, clients need also to register their identity/wallet or their device if not done in previous usage. Once connected to an AP, the client selects one of the offered retail Internet services offered in that network. The available options can vary in terms of QoS, depending on the gateway, and retail pricing policy, according to options provided by the US_c. Then, the client deploys a US_c with the provider of the Internet service of his choice and deposits in the contract, in an *escrow* fashion, an amount of tokens corresponding to the amount he would have to pay after consuming the service plan unit or their membership fee. After the provider accepts the retail contract and the necessary information is exchanged, the Internet connection (or any other selected service) is enabled. The provider is responsible for invoking a monitoring check, through the US_c, that upon completion of the data plan would result in transferring the tokens to the reserve fund.

Economic compensation: We implemented a periodic (every hour in our prototype) economic compensation. In every payment iteration device owners are compensated according to the current balance or goals of the reserve fund. The address of the FW_c acts as a reserve fund (in token units), balancing the router payments whenever the user payments in a given period are not sufficient, or saving excess payments otherwise for adjustments in future iterations. Moreover, if the reserve fund is unable to pay the providers, the debt is logged in the

FW_c for future resolution.

Forwarding prices are set by the device owners. The prices are set per unit (e.g. MB) of data they forward. Nevertheless, the maximum forwarding price is dynamically defined by the FW_c, to ensure the sustainability of the economic model.

In our prototype, the initial forwarding price of a node f is set for a depreciation period (RoI) in N iterations, considering its installation cost (CAPEX) and operation cost per iteration (OPEX$_1$), as $FPr_{f,0} = capex_f/N + opex_1$. We have developed a simple *price readjustment* factor $\alpha_i = Collected\_funds_i/Owed\_funds_i$, that can either affect forwarding prices as $FPr_{i+1} = FPr_i \cdot \alpha_i$ or retail prices as $RPr_{f,i+1} = RPr_{f,i}/\alpha_i$, aiming at avoiding long-term increasing debt and overall utility maximization around low retail prices while ensuring the target return of investment rate (cost oriented). Based on this initial idea, we plan to test more complex and long-term economic sustainability policies that consider the fluctuation of demand, including also the ability to set funding goals for network investment (crowdfunding).

In each iteration, the devices, using their $ERC721_c$ identity, send an invoice (traffic report) to the FW_c, which retrieves the corresponding data from the monitoring service, through the Oracle. The $ERC20_c$ tokens, that correspond to the contribution of each device, are calculated according to the amount of forwarded data and the defined node forwarding price $FPr_{f,i}$. Considering that these calculations can become expensive and time-demanding, we decided to outsource them from FW_c to an external trusted server, using the Oracle as communication endpoint. Finally, the calculated compensation is transferred to the owner of the device, as defined in the $ERC721_c$.

C. Oracle, monitoring, nodes

Our implementation is based on a private permissioned Ethereum blockchain using Geth, for the stability and extensive support compared to alternatives [8]. Several instances, operated by separate owners, are expected to run in a local network to ensure the safety and liveness of the distributed ledger. The smart contracts are implemented in Solidity language, together with the Truffle suite for managing and deploying contracts in a private Ethereum network.

In the *Oracle service* a smart contract communicates with a server that is responsible for making external queries to the monitoring service and sanitizing their responses to make them available to the smart contracts.

The *monitoring service* gathers traffic forwarding counts (bytes, packets) in each device over time, in aggregate form, without details about attribution to users (retailers). It combines the Prometheus time-series database with a distributed traffic monitoring system [12] to provide a highly available, transactional storage that can tolerate and adapt to network partitions and disconnections that can happen in network islands. This information will be supplied that will be further used to calculate the economic compensation.

Additionally, a network island maintains a *node database* with information about the nodes, links, servers, users, etc. that participate in the network. Storing all the information inside an ERC721 token is neither safe nor efficient. Also, it makes no sense to save information which is not useful within the blockchain environment inside a smart contract. For this reason, we use a MongoDB database. Communication with it is secured and restricted only to Oracle requests.

Table I
FUNCTIONAL TESTS

Process	#Trans	Latency	Gas Used	Trans Fee
Bootstrap Network	14	22768ms	22141303	0.44282606
Device Mint	3	2444ms	35212	0.00070424
Internet Access	14	8657ms	3586777	0.07173554
Compensation	27	11781ms	1348380	0.0269676

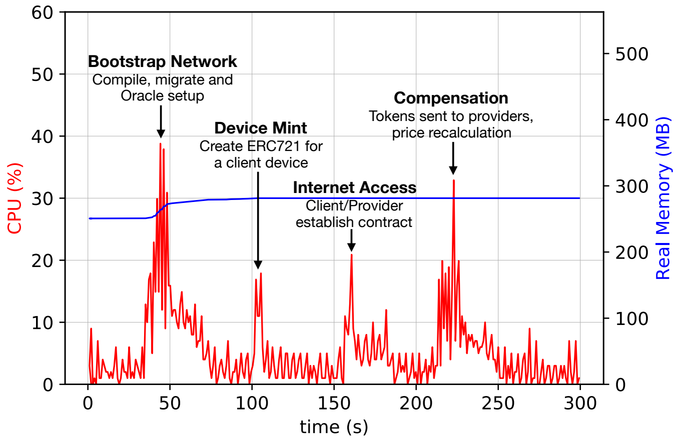

Figure 3. CPU and Memory of Processes

IV. EVALUATION

We aim to provide an evaluation of the prototype implementation and evaluate its suitability for a scenario with resource-constrained hardware. We first deploy it in a controlled subnet of a real community mesh network to perform a functional evaluation and resource consumption analysis per process, as they are described in III. Then, we analyze the scalability of the system from the perspective of transaction rate and finally we propose a study case to verify the validity of the proposed value transfer approach.

A. Testbed Setup

For our experiments we deploy commodity, off-the-shelf devices (Quad Core 1.4 GHz, 4GB RAM) in the Guifi-Sants mesh network [13]. These characteristics conform with the device choices in usual mesh deployments. On these devices we deploy an Ethereum Proof-of-Authority (PoA) network with 0 blocktime (instant upon transactions). We choose the PoA consensus mechanism since it consumes less resources, which appears to be more suitable for a mesh network environment. PoA networks consist of two kinds of nodes –*validators*, who sign and create new blocks– and *non-validators or clients*, who do not have the authority to create new blocks and are deployed as interface for users to connect to the blockchain network and submit transactions.

B. Functional Evaluation and Resource Consumption

The analysis of this section is done per process as described in III-B, with focus on the *economic compensation*, since this is the only process that is repeated periodically and the number of transactions involved (token transfers) depends on the number of provider devices in a network. *Device Minting* is executed only once per new device. *Internet Access* is associated to a pair of devices, and while it is a recurring process, possibly with multiple coexistent instances, its transactions are triggered by external actors, with less probability of generating a significant concurrent transaction load.

Table I presents an overview of the execution of the smart contracts on a local Ethereum, listing the number of Ethereum transactions needed to perform each process, the overall transaction latency and the gas used. Even though we observe that the processes are quite costly in gas, the private deployment permits us to ignore gas costs. For the case of the *Compensation* process, that was measured considering one provider, increasing the number of nodes in the local network would increase the necessary calculations for the distribution of the tokens, the number of transactions and, hence, the response time of the FW_c. The increased complexity of the calculations, as described in section III, has been addressed by outsourcing the calculations to an external server. On the other hand, the increased number of token transfer transactions could affect the performance of the *Compensation* process. This issue is studied further in the following scalability analysis.

In order to understand better the resource usage of the core processes of the system platform, we measured the CPU and memory usage of the validator node. Figure 3 shows the CPU usage along with the RAM occupied by the different processes. The tests were performed with an additional delay of 50 seconds between them, to clearly identify the different processes. Overall, we can observe that, using the described commodity devices, the CPU consumption never reaches more than two cores and that the RAM memory stays constant in a range between 250 and 300 MB. As a result, we claim that it is possible to deploy it in commodity devices, similar to those used in mesh networks.

C. Transaction Scalability

To study in more depth the scalability of transactions, we built a synthetic application where Ether token is transferred from one account to another, using one PoA validator node. Considering that this is one of the less complex transactions in Ethereum, this experiment gives rough bounds for the number of concurrent transactions, or in other words, bounds about the number of concurrent nodes doing one transaction. Transactions are generated from within the host, and sent to the miner through Inter-process communication (IPC). Experiments are performed measuring transaction latency (completion time) from 2^0 to 2^8 transactions fired concurrently.

Figure 4 shows the relation between the number of concurrent transactions and the average transaction completion latency. As depicted, the transaction latency varies from a few milliseconds to seconds. Values higher than 2^8 do not appear in the figure since before reaching concurrent 500 transactions an important percentage of them are not completed or delayed significantly. This is attributed to queues in the Ethereum validators and the fact that the Ethereum implementation rejects transactions that are waiting to be mined more than 50 blocks.

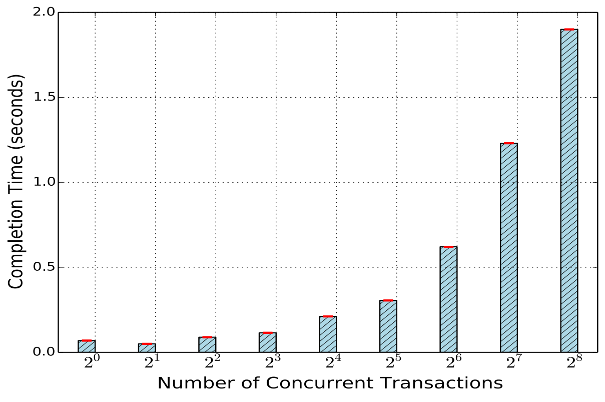

Figure 4. Simple Transaction Scalability

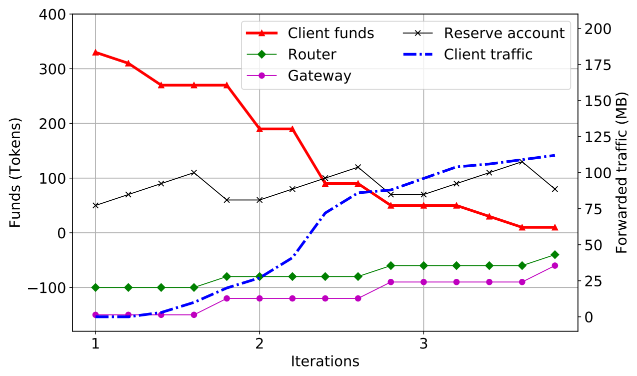

Figure 5. Expected value flow among consumers, providers and reserve.

As mentioned earlier, transaction concurrency is a possible bottleneck mostly for the *Compensation* process. Such concurrency could only be theoretically achieved either when the set of devices in a mesh island issue their traffic invoices, in follow-up to a request of the FW_c to the Oracle service, or when the resulting distribution of tokens is executed. The invoicing of the different nodes is spread throughout the compensation period. Therefore, we observe that there exists a tradeoff between the number of nodes that can participate in a local network and the length of the compensation period, considering values of seconds or minutes. Moreover, the distribution of tokens can face a bottleneck in mesh networks with about 500 nodes. Although such scenarios are not common and difficult to manage due to pressure of aggregate traffic in the most busy paths and Internet gateways, they can be dealt by splitting the distribution of tokens to the nodes into multiple iterations.

D. Value Flows

To verify the correctness of our compensation system, we demonstrate here how the tokens owned by the different actors (clients, providers and reserve fund account) flow in a way where the funds distribution performed by the system is reliable. To this end, we study a case with three consumers (clients) and two providers. We run three iterations of the functional tests, focusing only in the two phases where the token flow occurs: accepting a service contract, and compensation.

Figure 5 shows a visual representation of the study case, calculated from a simple analytical simulation. For clarity, we associate the start of each iteration with the compensation period, during which the providers receive tokens proportional to the amount invested on the devices (i.e. CAPEX and OPEX). Additionally, we are applying an aggregate function to plot both the clients' funds and traffic.

As illustrated, providers' funds are negative in the beginning, representing the initial investment of the device owners, and increase after every iteration. Moreover, the figure demonstrates how the reserve fund can act as a value buffer for the compensation system, as it is increased when clients add a lot of value (iteration 2) that can be used to compensate the providers in the future (iteration 3) when there is less client demand. As the clients are not obtaining new tokens, their funding expenses follow an asymptotic behavior towards zero, which consequently leads to a significant decrease in their forwarded traffic.

Although not represented in this case study, reductions in available reserve funds are corrected by the maximum price readjustment functionality, as described in III-B.

V. Related Work

There are related works that explore aspects of how mesh networks can combine with blockchain to provide connectivity under a decentralized economic model involving independent providers and consumers of a crowdsourced network.

Our design relies on a local blockchain infrastructure that operate an Ethereum-based lightweight consensus protocol (permissioned and low overhead, PoA) but Hyperledger Fabric is an alternative [7]. Other projects in development combine the payment in mesh networks with blockchain, such as Althea Mesh [14] where routers pay each other for bandwidth using cryptocurrency payment channels.

Several studies provide economic analysis and designs for resource trading. Our system is inspired by the governance, operation and economic analysis of the Guifi.net community network [4]. Route Bazaar [15] is a backward-compatible system for flexible Internet connectivity, inspired by the decentralized construction of trust in cryptocurrencies, looks at agreements among Autonomous Systems with automatic means to form, establish, and verify end-to-end connectivity agreements. Request Network [16] is a decentralized network that allows anyone to request a payment (a Request Invoice) for which the recipient can pay in a secure way. Request can be seen as a layer on top of Ethereum which allows requests for payments that satisfy a legal framework.

VI. Future Work

Several aspects of the economic model are worth to explore, such as fairness across resource providers, proportionality to investment or effort, predictability of investment returns. Pilot experiments will test the operation and friendliness of the system in real-world community mesh networks. The local token economy can expand beyond network services, as a exchange means for other transactions in a community. Therefore, we plan to explore how communities can benefit from local currencies. The needs for identity, anonymity, authorization, traceability are aspects to develop further in pilots.

VII. Conclusion

We investigate how blockchain and networking technologies can combine to enable universal connectivity in a decentralized and sustainable manner through the deployment of local access networks that share network services including Internet gateways, crowdsourced by several independent participants. Networking infrastructures and services need a balance between the contribution (network capacity) and consumption (network traffic). Economic flows across all participants can ensure the right capacity and maintenance of the network (compensating human efforts and economic investment), and therefore provide satisfactory quality to each participant, and scalability as more participants join.

The presented results demonstrate that our platform can enable value and data flows in mesh networks, and can be deployed in resource-constrained environments.

As far as we know, our work is first to analyze the intersection between blockchain technologies and networking technologies from the experience of production mesh networks at city scale. Decentralized solutions, that combine connectivity, data traffic, and value flow, lower the barrier to entry for crowdsourced connectivity solutions that promotes participation, creating social and economic benefits supporting sustainable local socio-economic development.

References

[1] I. T. Union, "ICT Facts and Figures 2017." [Online]. Available: https://www.itu.int/en/ITU-D/Statistics/Pages/facts/default.aspx

[2] L. Maccari and R. L. Cigno, "A week in the life of three large wireless community networks," *Ad Hoc Networks*, vol. 24, pp. 175 – 190, 2015.

[3] P. Micholia et al., "Community networks and sustainability: a survey of perceptions, practices, and proposed solutions," *CoRR*, vol. abs/1707.06898, 2017.

[4] R. Baig et al., "Making community networks economically sustainable, the guifi.net experience," in *Proceedings of the 2016 workshop on Global Access to the Internet for All, Florianopolis, Brazil, August 22-26, 2016*, 2016, p. 31–36.

[5] D. V. et al., "A technological overview of the guifi.net community network," *Computer Networks*, vol. 93, pp. 260 – 278, 2015.

[6] K. Gai, M. Qiu, and X. Sun, "A survey on fintech," *Journal of Network and Computer Applications*, vol. 103, pp. 262 – 273, 2018.

[7] M. Selimi, A. R. Kabbinale, A. Ali, L. Navarro, and A. Sathiaseelan, "Towards blockchain-enabled wireless mesh networks," in *1st Workshop on Cryptocurrencies and Blockchains for Distributed Systems*, ser. CryBlock'18. New York, NY, USA: ACM, 2018, pp. 13–18.

[8] A. R. Kabbinale, E. Dimogerontakis, M. Selimi, A. Ali, L. Navarro, A. Sathiaseelan, and J. Crowcroft, "Blockchain for economically sustainable wireless mesh networks," *Concurrency and Computation: Practice and Experience*, e5349 cpe.5349.

[9] E. e. a. Dimogerontakis, "MeshDapp – Blockchain-Enabled Sustainable Business Models for Networks," in *Economics of Grids, Clouds, Systems, and Services*. Springer, 2019, pp. 286–290.

[10] M. G. Lijesen, "The real-time price elasticity of electricity," *Energy Economics*, vol. 29, no. 2, pp. 249 – 258, 2007.

[11] F. Vogelsteller and V. Buterin. (2015) EIP 20: ERC-20 Token Standard. [Online]. Available: https://eips.ethereum.org/EIPS/eip-20

[12] R. P. Centelles, M. Selimi, F. Freitag, and L. Navarro, "Dimon: Distributed monitoring system for decentralized edge clouds in guifi. net," *12th IEEE International Conference on Service Oriented Computing and Applications (SOCA 2019)*, 2019.

[13] M. Selimi, L. Cerda-Alabern, F. Freitag, L. Veiga, A. Sathiaseelan, and J. Crowcroft, "A lightweight service placement approach for community network micro-clouds," 2018.

[14] Althea Mesh, "Althea: An incentivized mesh network protocol." [Online]. Available: https://altheamesh.com/documents/whitepaper.pdf

[15] I. Castro et al., "Route bazaar: Automatic interdomain contract negotiation," in *15th Workshop on Hot Topics in Operating Systems (HotOS XV)*. Switzerland: USENIX Association, 2015.

[16] Request Network, "A decentralized network for payment requests." [Online]. Available: https://request.network/assets/pdf/request_whitepaper.pdf