The blockEDU Protocol

A Peer-to-Peer Education Network

July Xth, 2017

Nathan Ginnever
Dylan Lott

Abstract

BlockEDU is a protocol built on-top of the Ethereum blockchain and a network of distributed reputation where data is stored in a private and decentralized cloud. BlockEDU implements a consensus mechanism similar to that proposed by backfeed [cite backfeed] called Proof of Affordable Education. It is an alternative education protocol that can host free--and open source--course management systems (CMS) where Distributed Open Collaborative Courses (DOCC), Small Private Online Courses (SPOC), Massive Open Online Courses (MOOC), and institution free traditional universities can be explored. The goal of BlockEDU is to solve some of the apparent issues with current accredited universities as well as online courses in a way that makes quality and reputable education affordable and accessible to all. BlockEDU offers the ability for anyone to create custom courses or structure degree programs that conform to current academic paths without the centralized trust provided by the current system. BlockEDU will remain an uncontrolled non-profit that has no ability to impose any unnecessary fees on students or educators. There are no C-titles that will take unreasonable profits from this organization, and it will remain a community driven project throughout its lifespan.

1 Introduction

The question has been raised as to whether institutional universities and online course systems are continuing to provide an affordable and quality education for students. The increase of student loan debt to income ratio has caused many students to begin looking for alternatives. Alternative forms of education enabled by the web have become a popular industry in the past decade as students face increasing tuition rates and the inability to afford traditional accredited universities. With MOOCs you do not need to be geographically close to your institution of learning. As long as the academic community has an internet connection and a cursory knowledge of technology they may connect with each other to advance knowledge. Online institutions such as Udacity, edX, and Coursera have seen success as a perhaps viable ways of spreading mass education at an affordable price. They promise to offer affordable and accessible information to those who otherwise would not have had a chance to learn. BlockEDU is an attempt to aid in Goal 4 of the 2030 Agenda for Sustainable Development where the UN has envisioned a way to ensure inclusive and quality education for all and promote lifelong learning. It is encouraged that all existing education platforms, online and offline work together to achieve this goal. One can imagine traditional universities such as Harvard and MIT and online systems such as the MOOCs listed previously committing to this consensus and building their governance model on-top of blockEDU.

2 Problem Statement

[research education outside of the united states, it is clear that the industry only cares for profit here, but other countries should receive credit for doing it better]

Education has become a booming industry [pull stats on education industry]. We have come to call this *the industry of education*. What was once thought to be a service to society has become a system where companies can take profit from a core need. Students and teachers rely on centralized institutions to provide an understanding of trust. Who should a student trust to provide a quality or highest tier education or who should a teacher trust to be a worthy student in their course? As institutions both online and offline take on higher numbers of students, there is a lack of personal interaction between educator and student and students to other students that is important to building relationships that facilitate knowledge. Not everyone learns at the same pace or in the same way, therefore it is a challenge to provide a template to teach mass numbers of students at the same time.

Germany has made their public universities essentially free, which is a great step in the right direction, but this says nothing of the quality of education they're offering. As we intend to show, the blockEDU model can not only incentivize the creation and dissemination of affordable education, but can also offer a way to gauge the quality of that education, while allowing a system to discourage bad actors.

2.1 The Profit Model - Industry of Education

[spend more time expanding this idea of the industry of education, there are a ton of resources here. This needs the most fact checking and is perhaps the most convincing problem statement]

The industry of education is run just like every other industry, with profit as its goal. Organizations that once started with a non-profit business model have inevitably moved towards institutionalization, advertising, and profit. These evolutions toward a profit are methods to sustain the organization, whether that be attracting better educators and affording better tools, or sustaining the non-profit business. There is an argument that the free market and private education incentivises the best educators and provides the best tools to the select few that can afford it.

2.1 (a) The Online Education Profit

Every system has a cost to building and maintaining its infrastructure and systems such as the platform provided by edX try to encourage the open source community to do this service to reduce costs, however their open source system is hard to run. The fact that these systems are too complicated to set up for an average educator it enables the creation of an entire industry of cloud service providers to take profit from students and teachers. In order to maintain the software, these online course providers populate their systems with advertisement. Developers and maintainers should be paid fair wages for creating these systems, but online universities and course management systems fall into the same pattern as traditional universities where CEOs and CTOs are taking large salaries at the cost of both the student and educator. There must be a reasonable cost to developing these organizations that is not being explored.

2.1 (a) The University Profit Model

Offline private institutions are also failing to keep courses affordable for students and are facilitating the rise of online education. Current institutions have too many auxiliary fees that don't benefit the student such as administrative fees. Higher up positions such as deans and CEOs are taking multimillion dollar salaries [source open information on actual salaries]. The

increase in student loan debt discourages students from enrolling in traditional universities. [source student debt information].

2.3 Accreditation

Universities maintain their place in society because there is nowhere else for teachers to have the reputation they need to be trusted [back this statement up, seems reasonable but may not have evidence]. Until recently it has not been understood how trust can be accomplished without a centralized authority to issue it. [more research into the need for accreditation, question why it has to be granted from a university. Trust models will be discussed further down in the rep system. We will be describing how we can offer trust without a central authority. Perhaps create an analogy to CA problems vs web of trust]

2.4 Lack of Interactions, Feedback and Community in MOOC

MOOC lack a flexibility that allows the community to decide how courses should be structured. Some universities such as Amherst and Duke have taken hard stances against using MOOC and have expressed concern that the massive nature of MOOCs or the need for those involved to understand technology or a lack of a strong sequenced structure makes them unacceptable. They often forget about foreign language speakers and only cater to English as they are not able to coordinate on a local level. These MOOC systems fail to explore the difficulties of relevance and knowledge transfer across borders.

The ability to scale class sizes up is potentially one of the largest weaknesses presented by traditional MOOCs and institutions. There is often a lack of interpersonal connections and diversity in scope that cater to all types of learning. Because of this there are large incompletion rates and only a select few that will succeed from MOOCs. Others will drop off and lose money to a for-profit institution. Large class sizes also limit the ability of the educator to learn better ways of structuring their courses in the future. Teachers often report that personal feedback is a valuable resource for both educator and student.

2.5 Learning Models

[let's discuss standardized tests, and the lack of value that letter grades give in understanding someone's knowledge]

[cite algebraic reasoning] A case example of the value of personal feedback can be found in teachers and researched beliefs about the development of algebraic reasoning in high school students. In this example the teachers must organize their mathematics testing and problem sets by their predicted difficulty for students. Textbooks sometimes are not a reliable source for

this reasoning. "The Symbol Precedence Model of development of algebraic reasoning, in which symbolic problem solving precedes verbal problem solving and arithmetic skills strictly precede algebraic skills, was contrasted with the Verbal Precedence Model of development, which provided a better quantitative fit of students' performance data."

3 Proposed Solution

Decentralization of knowledge provided by centralized institutions is, in our opinion, a good thing, for reasons we'll look into shortly. The burst of new technologies built around distribution and decentralization in the last few years have presented new and exciting ways for information to exist, and we think that education is a prime use case for these new technologies.

The blockEDU protocol does not define a single method that should be considered best for educators or students. We build communities that get to decide what is best. We understand that education benefits when there is greater communication and enthusiasm to be a part of a community that progresses the area of knowledge. Our protocol can not dictate that a course should remain small so that personal attention can be achieved, but we can offer a system that makes this possible at the choice of every educator and student. We strive to make a global system that knows no boundaries and can express knowledge at a local level relevant to those involved. blockEDU is a protocol, thus it is possible to be used in anyway necessary.

3.1.1 Censorship

Media and speech are constantly under attack. A distributed platform for education allows for censorship to be almost nullified, especially as more nodes participate in the network. Countries with less than ideal freedom of speech laws can now access this network, and

3.1.2 Quality of Information

3.1.3 Dissemination and Spread

3.1.4 Data Persistence

3.1 Profit Solution

BlockEDU puts forth an idea that using decentralized technologies can enforce a non-profit model that is self sustaining. The organization is maintained by reserving a portion of the total supply of EDU tokens. This gives our developers a chance to improve the software in an open manner. We build a community that is encouraged to participate in building this with the help of

properly funded individuals that bring the level of expertise that closed systems normally have. Doing so commits the organization to being only worth what the community believes it is worth. As the free market decides to support or not support blockEDU.

3.1.1 Online Model

BlockEDU offers an easy to use platform for educators and students such that there is not a need for unnecessary technical installations or over priced cloud data solutions. This reduces the cost burdened to both educators and students that were once forced to pay for services to run a CMS. This eliminates the need for an industry of service providers that have been working with institutions in the industry of education. While there are still monetary costs to using blockEDU, it is expected that these fees will be less than cost of scaling a course on traditional online education platforms.

3.1.2 Offline Model

There are no C-titles with large salaries here. A public and transparent record is available from the non-profit for each of the salaried employees in the organization. We do not require the maintenance of sophisticated buildings to achieve the same caliber of education provided by private institutions. Anyone can set the meeting points for courses where it may not be necessary to have all of the equipment needed for a chemistry course paid for by students taking a single math course. Removing the physical institution offers a flexibility never seen in a quality education. We encourage world renowned educators in all disciplines that do not require a laboratory or gymnasium to teach a course in a local library or park.

3.2 Consensus

To encourage that education remain free, we present a consensus model that *rewards educators for maintaining a free course* as well as nodes for *reporting correctly on this consensus*. Just as bitcoin [cite bitcoin] maintains consensus on the correctness of financial transactions, blockEDU maintains consensus on the idea that education should be affordable. The consensus protocol incentivises educators to not charge students for their service. While giving education and course content away for free is not mandatory, the protocol rewards teachers that maintain this standard, and does not reward those that present a cost for their time and content. [backfeed] “These mechanisms ensure a fair distribution of the generated value to each individual, according to the perceived value of their respective contribution to the organization as a whole.”

3.3 Accreditation

We understand that blockchains don’t do everything well and certainly are costly. It does not always make sense to store information on every machine in a network that does not need this data, however there is a compelling argument that blockchains provide publicly verifiable information that cannot be altered when statements need to be made. If used appropriately, this can be a powerful feature of a network of nodes agreeing on such a statement.

One of the benefits of institutions is the ability to stamp approval on a record that others can trust to be accurate. We propose that we can replace this institutional trust in a student's records, or an educator's reputation with a blockchain. Digital signatures within a blockchain offer a way for identities to broadcast their reputation and value without the need to consult centralised institutions since we can trust the statements made by owners of these private keys linked to identities.

3.3 (a) Educator Reputation

The blockEDU protocol creates the ability to use distributed reputation in a way that provides the accreditation needed for students to trust that they are getting a quality education. Educators can use this proof as a way to demonstrate their value to other educators and for side organizations that encourage the development of knowledge in their specialty. We discuss further in section **5** how educators gain their reputation in our system. These records are also stored in a blockchain so there will always be a pointer that anyone can view to determine if they are interacting with a trusted educator.

3.3 (b) Student Reputation

Educators now have the ability to create a permanent log of signed records that students have completed courses that anyone can verify. Students are given a reputation and are given a database to maintain their course records to pass knowledge forward to future employers or education groups to prove they have achieved a quality education. Each community member has their course records permanently printed in the Ethereum blockchain. Students maintain these records on a distributed cloud storage platform with encryption keys so that they always retain who gets to see their achievements. To prove the accumulation of knowledge to the protocol, smart contracts read course completion registers that allow them to unlock more advanced courses. In this way you could imagine building entire degree paths on-top of the blockEDU protocol.

3.4 Personal Interactions

The possibility of peer-to-peer education will ensure that the community always forms the organization. There will be no centralized authority to dictate how a course should be sized, scheduled, or monetized.

Choice is a core tenant of blockEDU and if other organizations are not comfortable creating a decentralized community, they may choose to establish a more traditional centralized and permissioned university role. You may set the permissions on your course contracts to allow anyone to join or only those that you deem fit or that have completed enough prerequisites. More information on this is presented in section **7.3**.

Just as massive course structures are possible so are small localized communities with personal interaction. Educators choose the max number of identities that may register in the course contract. If they wish to keep a course small it is as simple as limiting this number and structuring the course to use the frontend CMS forum application to facilitate personal communications. You may also use the system to advertise local meetup facilities that will happen at predetermined times. Ultimately the members of this community get to determine where they belong and what works best for them.

4 blockEDU Protocol

Here we present how the blockEDU protocol is structured. The protocol is inspired by the blockchain system first laid out by bitcoin [cite bitcoin]. BlockEDU can be seen as a related to the work done by backfeed [cite backfeed paper] protocol differing in that it is not general distributed governance but instead specific to education.

4.1 Consensus - Proof of Affordable Education

Analogous to bitcoin's proof of work scheme (PoW), proof of affordable education (PoAE) will be the protocol for educators to receive financial incentives. Students have the ability to give a reputation to each individual course taught so that there is a mechanism for deciding if the course was *of quality* and *affordable*. Educators will be rewarded from the total reserves of EDU tokens created much in the same way that mining occurs in other cryptocurrencies. Proof must be supplied in reputation that the course was of value to the students given as a real number in the range [0,1]. A monetary reputation is assigned to the course as a separate unit to decide that the course was taught affordably by the students also in the range [0,1]. A student may get a good education from a teacher, but still assess a low monetary reputation for the course. If either of those conditions is below their respective thresholds, the teacher will not be paid out from the reserve of tokens.

4.2 EDU Tokens

EDU is a standard Ethereum token and the total supply is held in reserve on the blockchain state. The tokens provide a method for rewarding the educators for maintaining the consensus that education should be affordable. EDU works as a method for students to provide a fee for entering the courses which becomes the only method for educators being rewarded once the total supply has been depleted. Section **4.2.5** provides more information on how the total supply is awarded over time. EDU also enables educators to create courses by incentivising full nodes in the reputation system to report their course reputation to the course contracts.

4.2.1 zEDU Tokens

To maintain an objective that any users in the protocol may remain private, we offer a mechanism for converting traceable Ethereum tokens to zkSNARK Zcash tokens. While still in the early stages of development during the time of writing this, blockEDU will provide a method for atomically swapping the ERC20 token for a future standard Zcash token. Zcash and tokens are deployed as the payment mechanism for the Orc network. This is done by [cite sapling, read method]... This enables private payment methods to both students and educators in the event that certain education becomes oppressed or deemed illegal by authorities.

4.2.2 Mining EDU

While there is no computational work or proof of stake for mining to ensure the network reaches consensus, a loose analogy to bitcoins [cite bitcoin] mining can be made in the sense that tokens are still distributed from a total supply when a proof has been established. As stated in **4.1** the proof here is the reputation scores of the course from student feedback passed in by oracles that are further described in **4.4.1**.

4.2.3 EDU Blocks

A course block is similar to bitcoin where miners create blocks of transactions to be rewarded from the total supply. However, instead of blocks filled with financial transactions or state updates, educators are creating courses filled with students receiving knowledge. The proof attached to each block, or course, in this sense is the reputation ratings derived by the students and the ability for the system to check that the course was provided in an affordable and quality way. During the mining process in bitcoin, mining hardware computes hash values until it hits a certain target. In blockEDU, teachers will produce informative education until their reputation score for doing so is recognized.


4.2.4 Block Transaction Fees

Transaction fees are an optional parameter set by the student before entering a course. Instructors may choose to pick up students that provide a higher fee for entering the course. This is analogous to a bitcoin transaction fee, where a miner is rewarded an extra amount for picking up the transaction into their block. When the total reserve of EDU runs out in [calculate time it runs out] this will be the only remaining reward for educators.

4.2.5 Token Distribution

[formulate exact details on the reserves and the deflation of issuance, create a graph plotting the issuance over time]

We control the distribution of tokens to maintain a timeline of token depletion. Transaction fees will be the only mechanism for paying educators when the total supply runs out, and we speculate that this may not be a feasible mechanism to maintain the consensus just as Bitcoin will face uncertainty in its mining protocol in the approximate year 2140 when the block reward stops. The answer is not know but blockEDU may update the protocol in the future or rely on the transaction fee for entering a course as a method for balancing affordable education.

Just as in bitcoin, we devise a method similar to that of difficulty adjustment to keep the total distribution of tokens at a relatively constant rate. The bitcoin protocol attempts to control the time it takes to mine a block by increasing or decreasing the hash target, thus changing the computational power needed to hit the target. In blockEDU, there will be a variable number of courses active at any given time, which will cause the distribution to become variable.

Another factor to inconsistent supply depletion is the average rating an educator has and how much they will be paid for each class as is described further in **4.2.6**. The issuer contract keeps track of the distribution of tokens over time. This is done by the checkIssuance() function that can be called by anyone. As in bitcoin, this function call will be available to be called approximately every two weeks. This is done every 2016 blocks in bitcoin, however we will be using timestamps provided by the miners to allow this check as block times are quite variable in Ethereum. This call checks the total issued during the last interval and adjusts the max reward per course accordingly.

[supply some distribution methods]

4.2.6 Block Rewards

An issuer contract holds the reserve of all tokens that will be issued to educators. During four stages of the course contract, the issuer will rely on nodes in the reputation network reporting on the course like an oracle to determine if the educator should be rewarded for maintaining

consensus. The course contracts has a time-to-live (TTL) for the course length as provided by the educator creating the course via a CMS. Each course contract has four states that will open their contract up for rewarding the educator from the issuer contract. This is built into the constructor of valid course contracts and calculated automatically upon a course being instantiated. This mechanism is to ensure that a teacher may be paid in installments during the length of the course. The educator may send reputation requests through designated nodes handling their course context area to initiate these reports.

Every quarter the course contract will allow for the nodes to begin reporting the reputation to the contract. The fields that are able to be reported on are an *affordability value* and an *education quality value*. Recommendations are issued by the enrolled students In order for a node to know how to report correctly to a contract during each quarter of the courses TTL. The thresholds for the contracts to pay from the reserve are an affordability score above 0.8 and a quality score above 0.7.

The amount of tokens issued to each educator is already not a static value as mentioned in **4.2.5**. Block rewards should also have different values depending on how reputable the educator giving the course is. In this way a teacher with a high value of recommendations is paid more for their time. Teacher context area recommendations come with a value based score between [0,1] as well as textual information about the teacher. The contract can't read the textual information so the value score of the reputation of the teacher will be used to determine how much of the max possible block reward they should get. If Alice has an educator trust rating of 0.8, then she will be rewarded 80% of the highest block reward during each of the quarterly payout rounds. 80% of the fees placed in the course contract by students will also be paid to Alice, and the remaining 20% of the reserve tokens ramains in the token reserves and 20% off the fee tokens will be rewarded back to the students. This scheme however will have a bootstrapping problem where the educators global reputation starts with a 0 meaning that all new educators must give their courses away for free. This is encouraged in the consensus protocol so we will maintain this and it also works to prevent sybil attacks.

In order for the contract to get an accurate report on the reputation and affordability scores for the course, it must not take the first report it sees. There is a window of time that each contract can take scores for each payout period. Since not all nodes in the reputation system have the same network graph and pull the same amount of recommendations, there can be an issue with nodes that report from a limited view of what students are recommending. In order to solve this, each report from the nodes comes with not just the average scores from the students, but also the number of recommendations they have seen to compute that average. The contract takes the score of nodes that report with the highest number of seen recommendations.

4.3 Identity

Identity in our system is in the same format as Ethereum as the network is bootstrapped from the RLPx cryptographic network and transport protocol . This gives our reputation system the ability to sign transactions on the blockchain that correspond one-to-one with the signatures in the reputation system.

Other forms of identity are required to work with the Orc Network and Zcash tokens (should privacy in token be required). These identities work the same way with slightly different formats, namely the hash functions and base encodings differ.

As to not burden each user of the system and fulfill the requirement that the protocol is as easy to use as current systems, blockEDU provides a service on the network that is entrusted to manage the private keys. In this way, students and teachers can simply remember a username and password, however, it is encouraged that those who are willing to store their own private keys do so. Private keys that are used to issue tokens will not be stored on any nodes. It is the responsibility of the user or the blockEDU organization to input public keys and maintain their token private keys. The blockEDU non-profit org has a centralized service to help facilitate the exchange of tokens to fiat currency to enable ease of use.

4.4 Nodes

Unlike users that solely interact with the CMS on-top of the blockEDU protocol, nodes on our reputation network provide a service to the user community. They *report the reputation records*, they *maintain the reputation database*, they *report as oracles* if they have enough positive recommendations, and *find trust paths* by facilitating user requests so that the ease of use requirement can be fulfilled.

4.4.1 Nodes As Oracles

A specific set of nodes chosen in the blockEDU protocol report to the course contracts on whether or not consensus is maintained. In doing so the protocol will reward the educator for mining the course. This decision is recommended by the context area nodes for that course and determined locally by assessing the recommendations given by the students in the reputation system by the oracle. Each course contract will look at the knowledge contract before allowing a report to update the state of their contract. This is to prevent arbitrary entities from reporting these scores. It is important to incentivise these nodes to maintain the consensus protocol, thus we reward reporting nodes based on their reputation in the system. These rewards are further discussed in section **4.4.3**.

As nodes are responsible for storing and maintaining the reputation system in the blockEDU protocol, they make for a distributed oracle that behaves in their best interest of keeping their reputation. If a node does not report the reputation data they are getting correctly to the course

contracts, they will receive bad recommendations when receiving feedback for this and be unable to facilitate further reports to the contracts for a time as designed by the UniTEC[cite unitec] system. In order to decide what nodes make good oracles, when the CMS looks for a node to facilitate educator requests initially, it looks at the knowledge registry for the *oracles context area* and determines the nodes with the best expertise as an oracle. A node is randomly selected encouraging nodes to gain the best possible trust to be in this set and therefore receive more requests so that they may pick up more fees.

Oracles are placed in the registry when any request facilitating node reaches the threshold for high enough reputation. Again we have a bootstrapping problem that we solve in this case by providing a few trusted oracles from a set of secured blockEDU organization nodes. These nodes will only decentralize this process in the beginning of the system until further nodes come online with a high reputation. A transparent exit of these nodes is made once there are enough oracles.

4.4.2 Nodes As Request Facilitators

Each node/s is responsible for managing a context area/s in the network. It is encouraged that these nodes are selected by the community for having some interest in promoting the health of their academic area if they manage the academic reputation context areas. They are responsible for connecting each user to the reputation system so it is essential that these nodes be trusted. Just as every identity in the system has a reputation, these nodes have a reputation which ensures that they behave in the best interest of the community. Any user may decide to run a full node and advertise themselves as a facilitator. Doing so gives the nodes the ability to begin building trust in the reputation system. Ideally the entire network would be made up of these full nodes and not end users.

The request facilitator reputation is the recommendations issued to these nodes for providing the service of handling all of the requests passed to them from user identities not running nodes from the CMS bridges.

4.4.3 Oracle Fees

In order to incentives nodes into becoming well behaved oracles in maintaining consensus, the protocol requires a fee from the educator that is initiating the reward function call on the course contract as overseen by the issuer contract. As mentioned in **4.4.1**, nodes with a high enough threshold of reputation in the oracle reporting context area are chosen at random as being eligible for facilitating these requests, thus being eligible to receive oracle fees. The mechanism for achieving this is to contact the knowledge contract to find a node that can act as the oracle, when the block reward is issued, these nodes will receive 10% as an oracle fee. [supply the algorithm for this]

4.4.4 Storage Fees

A token transfer method is created to enable the payment of the distributed cloud storage when requests are made that carry a data upload. This is important due to the cost that these nodes incur for satisfying this role. The nodes must pay for the farmers in the Orc network to hold the recommendation data and the course material data as well as facilitate the bandwidth traffic necessary for the communication between other nodes. It expected that educators and students pay for their storage costs for the duration of time in which the plan to have their data stored. In the case of student course records, where data is meant to be stored permanently, recurring payments will be necessary to compensate the Orc farmers storing these records. Section **6** goes into further detail of how a distributed cloud intends to be less expensive than traditional cloud storage providers.

4.4.5 Optional Request Fees

Nodes must also pay miners in the Ethereum and Zcash network for interacting with the smart contracts. It is only required that individuals pay for their own storage. If the node is processing a request for a user, the nodes is incentivised to pay the auxiliary transaction fees as a method for building trust, where being a trusted authority on the network is considered valuable since trusted nodes may receive oracle transaction fees. This is also a sybil attack prevention measure as discussed in section **5.7**.

To incentivise nodes to maintain their reputation, an optional *request fee* is given to each request a user makes to a node. These fees are calculated by the CMS on-top of the protocol for each request when a facilitating node is selected automatically so that teachers and students do not have to understand the protocol costs. The teachers and students are presented with these fees before making their requests to the network and can optionally choose to reduce them or not pay them at all. Each node may choose to reject the request and require a fee from the user to cover their costs. Nodes that do not have a high reputation are incentivised to process transactions with no fees to build their reputation. As the network builds trust, the fees remain set to 0 until the majority of the network is comprised of nodes with good reputation.

4.4.5 Offline Nodes

Since a node is the bridge for users to communicate with the network, it is important that there are backups when a node goes offline. Pointers to the location of reputation and course data collected are stored in the blockchain as a way of backing up data loss. Users are prompted to record their private key phrases on paper and keep them secured as a last line of defense in the event that the nodes facilitating their identities disappear. Recovering data will require consulting the data registry contract with the public key of the user's identity. Their private data was encrypted with this key so all data stored in the Orc network will still be recoverable.

4.6 Apathetic Students

The oracles that enable educators to be rewarded for their contributions to an affordable and quality education are only able to make their decisions based on the feedback from students. If students do not report quarterly on their enrolled course, or do not offer feedback on the quality of their professor, there will be no way for the protocol to determine the parameters of consensus are upheld. [There is no reason for a student to not rate their professors, as they are being rewarded in education, but I have not currently thought of the case that students either forget to report, or are too encumbered to do so. I need to enforce this in the protocol, perhaps the contract withholds some information or the student grades until it receives input that a threshold of students or each individual student in the case of grades withheld has provided feedback.]

5 Distributed Reputation System

The distributed reputation system deployed in the blockEDU protocol is based on the research done in the UniTEC system [cite UniTEC]. The reputation system presented here provides a consistent mechanism for providing recommendations and requesting recommendations about the oracles, educators, students, and individual courses.

5.1 Overview

The UniTEC reputation system paper is summarized here, pulling out what we believe to be the core elements to understanding how it is deployed in the blockEDU protocol. It is encourage that the reader review the UniTEC design for a better understanding.

Nodes form trust in other nodes and thus should be required to behave properly, or they would not become a part of the trusted network. This network is responsible for determining whether a teacher can provide reliable education to their students, determining if students can trust other students to be good peers in a course, determining whether the course offered was of value to the students and education was passed to them, whether or not the course was provided affordably, whether a node has a high enough reputation to be considered an oracle, and in general whether a node can trust the recommendations passed to them from another node.

5.2 Modeling

Below we describe the models of the distributed reputation system as formed by UniTEC, the *trust*, *knowledge* and *system* models.

5.2.1 Trust Model

Context areas are the identifiers of nodes responsible for storing recommendations. Based on a coral dsht. the hash of the context area relates to which nodes stores pointers to the nodes that are responsible for holding the recommendations for that context area.

Trust Context Areas

The context areas in the blockEDU protocol are partitioned by student and educator identities, academic knowledge areas i.e. Mathematics, Physics, or Literature, oracle nodes, and each individual course. Trust may be related in the individual, oracle, and knowledge areas as they can contain some degree of answering who to trust overall in our education protocol.

Here we must make a partition between the reputation that is derived for the individual, oracle, and academic knowledge context areas and the individual course context areas. We do not want recommendations from students that have not been enrolled in a certain course carrying any trust confidence weights over for recommendations that do not relate to that specific course. Doing this prevent students from continuing to provide feedback on courses that they are no longer enrolled in as each course should be an independent event from all others. The overall reputation of the educator in the superset of educators does carry over between courses so the issuer contract knows how much to reward each teacher.

As UniTEC describes, each context area has a trust value in the range from [0,1] and a given confidence vector in that user for giving the recommendation of that value. In this representation, 0 indicates that there is either no previous experiences with the user in the given context area while a 1 represents maximum trust in the user for either giving recommendations. For example Bob may trust Alice with a value of 1 meaning that Bob believes she is capable of recommending a good English teacher.

The confidence vectors store metadata that defines the quality of trust in the entity and contains the following entries:

- Number of direct experiences in that context area
- Number of indirect experiences (from context areas influencing the one in question)
- The last N direct experiences
- A blacklist (set by the user to prevent recommendations from certain identities)

Keeping track of the number of experiences with another identity gives the system a way to calculate trust on the fly. It is a parameter to the trust update algorithm as discussed further in section **5.6**.

5.2.2 Knowledge Model

To develop a local profile of “who knows what”, the number of personal recommendations is stored to build the *own expertise* indicator for a certain context area. Figure **5.2** shows an example of how these recommendations are stored to build this expertise.

To evaluate the *expertise of others*, we store the number of recommendation requests a user gets for each context area. This is also how we express each user’s understanding of a context area.

figure 5.2 Sample Knowledge model entry for the Mathematics context area

```
* @value {string} Context Area- “Mathematics”
* @value {Int} Stored RECs - 4

* @value {Hex} Public Key - 0x1234...
* @value {Int} Advertised RECs - 7
* @value {Int} Received RECs (authority) - 3
* @value {Int} Received RECs (hub) - 12

* @value {Hex} Public Key - 0x5678...
* @value {Int} Advertised RECs - 14
* @value {Int} Received RECs (authority) - 15
* @value {Int} Received RECs (hub) - 2
```

UniTEC definitions:

Definition (Authority): *Someone that is expected to have a direct knowledge of the given context area and can provide helpful recommendation in that area.*

Definition (Hub): *Someone that does not have a high level expertise in any one particular context area, but knows many Authorities that do.*

Recording both of these pieces of information allows us to to discern information about each user’s expertise as an authority (likely a teacher or advanced student themselves) or a hub (perhaps a dean of a university or a teacher's assistant).

5.2.3 System Model

Our system will be comprised of both light clients (end users) and full nodes. Nodes will be responsible for storing both the trust and knowledge models described in section **5.2.1** and **5.2.2** for each of the light clients.

[create a graphic here showing the top level context areas (student teachers and oracles) as a well as the subsets for each (the different academic areas of interest). This will give an idea of how neighborhoods are formed and how trust will be built by what is relevant to each individual user]

The systems develops *neighborhoods* for each identity as an overlay to the context area network topology derived from the dht. When a user wishes to be informed of a certain trust in another user, they reach out to the network and look for a connection in their neighborhood by issuing recommendation requests. Hops in the network are described by whether or not a facilitating node is receiving direct recommendations from another node that it is connected to and trusts. For example, if Alice wished to take a Mathematics course from Bob, she may begin a request for a recommendation from Bob by first contacting a node or nodes in the educators context area superset in the neighborhood of Mathematics. UniTEC defines neighborhoods as follows:

Definition (Neighborhood): *For a node N and a trust context area C, the neighborhood ONet(context area C, node N, level L) is the set of identities that can be reached from N in L hops. The set ONet(C, N, 1) is the set of identities directly connected to node N. The membership of an identity I in this local view or level 1 view of the neighborhood of a node N is determined by an algorithm that relies on two inputs: N's trust in I (stored in its trust model) and I's advertised expertise (from N's knowledge model entry about I) Generally:*

$$ONet(C,E,L) = \bigcup_{\forall E' \in ONet(C,E,1)} ONet(C,E',L-1)$$

[consider giving the example in UniTEC paper]

5.2.4 Recommendations (RECs)

UniTEC provides a data structure for storing trusted data items that serve as recommendations that we have altered. As such we are calling these data items RECs. RECs in blockEDU are comprised of *recommendation data*, *recommender data*, and *metadata*.

Just as in UniTEC we attach RECs to a specific context area and allow for these units to be flexible and contain arbitrary fields that specify the recommendation target and content. blockEDU recommendation data is as follow:

```
* @value {Boolean} Recommended - Positive or negative REC
* @value {float} quality percent - The course/teacher/student rating
* @value {float} affordability percent - In the case the REC is for a course
* @value {String} arbitrary length - Text specification of the REC
```

The recommender data contains the Ethereum public key of the recommender and their confidence in the given recommendation. This gives a statement about the recommenders own confidence in their recommendation and influences the trust update when processing the requester's feedback

```
* @value {String} Identity - Ethereum Hex encoded public key identifier
* @vaule {Integer} Recommender Confidence - between [0,1]
```

The metadata contains a timestamp and the recommendation ID. This is to give each recommendation a TTL that affects the ratings in the trust algorithm. Optionally there is a blacklist that can block identities from receiving the recommendation.

```
* @value {REC ID} Integer - An identifier for this REC
* @value {Unix Epoch} Timestamp - A TTL for RECs
* @value {Array} Blacklist - List of identities to not forward to
```

Recommendations are signed by the Ethereum private key of the recommender in order to prove the authenticity of the REC. We now have a system that supports a REC data structure that can be verified and quality checked.

5.3 Interactions

Here we describe how the models and components discussed until this point interact in relation to how the reputation system publishes and locates recommendations. The process as described by UniTEC is divided into five subtasks, "*disseminating* the request, *collecting* the responses, *organizing* the results, and *providing* and *processing* feedback for the system".

5.3.1 Publishing Recommendations / Advertising Knowledge

Any user may publish RECs as a way to provide experiences. For example in the student teacher relation, a student may request these RECs to provide a way for getting a good idea of whether or not an educator has positive past experience with their students and is knowledgeable about the given field of study. Students may want to read experiences with other students to see if they would be likely to have a good experience with others in a course they are about to take. Teachers may publish their experiences with other teachers in the field which

can also establish a greater trust that they are qualified to give a course within a field of knowledge context area.

The other side of publishing RECs comes into the system when a course completes and the consensus protocol needs to know whether or not to **a)** decide to take this REC into account, if the student is not enrolled in the course then it should be discarded by the node processing it, and **b)** decide if the course received a good rating from its students and is within the threshold for the monetary rating suggesting it was provided affordably, then the teacher should be rewarded the mining fees and block reward for the course.

When users fill out the REC fields and wants to publish, they may send the REC to their designated node/s. The user or blockEDU bridge will sign with the associated Ethereum private key and the node will timestamp the REC. A publish request is made to the dht to advertise that this node stores the new REC, as identified by the hash of the context area it is related to. The knowledge model can now be updated to reflect that this node stores a new REC.

In order for a user to request RECs for certain context area, their node must gather knowledge from other identities their stored RECs. This is accomplished in the blockEDU protocol by relying on the dht to locate pointers to nodes responsible for storing this data. Alternatively publishing advertisement messages to interested nodes for a given context area or knowledge providing contact can ensure these RECs are able to be located.

[spend some time reviewing this system]
The knowledge provider contract has a registry that stores the expertise areas of identities partitioned by context area. As Alice begins rating her professor Bob in mathematics, the knowledge contract will be updated periodically with a record that her identity is experienced in Bob's courses and that her node's identity is experienced in issuing these RECs. This helps prevent the startup problem of their not being enough direct experts to find RECs about Bob's course when there has not been many students that have taken the course. At anytime a node may consult the knowledge contract by passing Bob's public key to see who knows about him. In the case of course recommendations, this is the public key address of the course. These updates are not as frequent as publishing messages to the dht when RECs are created to avoid the costs of registering data in a blockchain.

5.3.2 Handling Requests

As UniTEC defines, the request message contains the *identity* of the requester, a *request identifie*r, the *recommendation target*, the *trust chain* and a *hop counter*. The hop counter and the number of identities that are reachable define the number of identities the requests reaches.

The *trust chain* is built during the request and gives a confidence to the requestor that they can trust the recommendation they are being served. If Alice wishes to get a recommendation for

Bob who is three intermediaries away, the full list of intermediaries and their corresponding trusts will be sent back to Alice so that she may decide whether or not Bob is for example, a good teacher, or whether Bob's course was provided affordably as recommended by the students in the course in the event that a node is requesting to report the block reward. Alice's request is first sent to her node, which forwards the request to all reachable members in the level 1 neighborhood of the request context area. Each recipient of the request for *L* hops computes the algorithm described in figure **5.3** as first put forth by the UniTEC system. Each new request created by the identities in the neighborhood of the context area of the request is formed by creating an addition to the signed trust chain link they are adding with their level of trust in the next hop to the recommender. The identity of the person they are passing the request to is also attached. Section **5.7** goes into further detail about how these chains are used to derive trust in the requested identity from the requestor, as there may be more than one path to a requested identity.

[create graphic to illustrate a request]

5.3.3 Collecting REC Responses

Those nodes that are able to fulfil the recommendation for an identity that was requested (meaning they have a stored REC for the corresponding requested identity) will create a recommendation response for the requestor. The response contains a request identifier, the signed REC, and the trust chain with the last link being the trust of the identity in the recommender.

```
calculate trust of Req. in Ii via the current trust chain
if not((request already processed) AND (with higher trust)) {
   if (suitable recommendation available) {
      create recommendation response
      send recommendation response back to Req.
   }
   decrease hopcounter
   if (hopcounter > 0) {
      foreach (member Ij of ONet-1 of Ii) {
         (* create set of new requests *)
         create copy of received request
         insert digitally signed trust statement:
         {
            trust of Ii in Ij
            (pseudonymous) identity of Ii
         }
         send request to Ij
       }
   }
}
```

Figure 5.3 UniTEC simplified directed dissemination algorithm

Only the most recent recommendations decided from the timestamp are kept. The requestor may receive multiple recommendations for the identity in the request and the ones with the highest trust ratings are displayed first.

5.3.4 Handling Feedback

The requestor needs to make a statement about which recommendations received were perceived as useful. The UniTEC system describes three steps for handling feedback by "*collecting the user feedback*, *updating trust* and *updating the neighborhood"*. This will require interaction from the user or node but it creates the necessary *experience* with the recommending identity which is either positive or negative. As described by UniTEC, "for each of the identities the experience and recommender confidence are added to the trust model and the number of direct experiences is increased in the appropriate context area".

This feedback is used in what is called the trust update algorithm. Section **5.6** goes into further detail about how blockEDU uses the original trust update algorithm presented by UniTEC. It is noted that algorithms are components that can be swapped in our reputation system if it is decided that a better one suites the blockEDU protocol at a later date.

The level 1 neighborhood is updated to reflect the new trust values. UniTEC cites Claus Offe [cite] who rationalized that it would be inappropriate to only get trust values from already trusted identities thereby abandoning the possibility for gaining experience from other sources. Choosing the neighborhood comes from the highly trusted reputable identifies the node knows about, consulting the knowledge contract for knowledgeable identities, and selecting random identities in the network from the DHT.

5.4 System Architecture

The UniTEC, blockEDU uses a Coral [cite coral] dsht in combination with smart contract registries as a method for accessing the stored reputation information that has been laid out previously.

5.4.1 Data Management Component (DMC)

A node's data storage happens both locally and remotely (DHT). The local storage stores information from the trust and knowledge models described in section **5.2** as well as the RECs created by the user identities. The RECs are formatted in JSON and transmitted over JSON-RPC. Figure **5.4 (a)** shows an example of a REC.

The remote storage uses a Coral DHT to organize participating nodes in a logical ring. Each blockEDU node stores part of the whole database of identities. This gives nodes the ability to

find other nodes that are responsible for maintaining the identity in questions information. This discovery mechanism is the primary way for a requestor to get information about an identity. The identity may also be a course contract that will have RECs from the students stored on the responsible node.

Figure 5.4 [create an example JSON REC]

5.4.2 Peer-to-Peer Overlay Component (POC)

The POC is an overlay network on top of the dht overlay that is responsible for managing the neighborhood view each node has. This is notified by the trust management component (TMC) upon receiving knowledge advertisements, RECs, and feedback on the quality of RECs. As designed by UniTEC, when a node receives a request for a certain recommendation, the POC checks the following:

Check 1: Check the trust chain in the request and check if the last link points to an identity that the node is responsible for. The request is discarded if not.

Check 2: Validate the signatures on all links in the trust chain and discard the request if any of them are broken.

Check 3: Contact the TMC to calculate the transitive trust of the requester in the current local identity. If the request has not been processed before (check by the request identifier) or if it has been processed with a lower trust value, the new trust is stored and processing continues else it is discarded.

If a REC is found that satisfies the request then it is sent back directly to the originator of the request with a finalized trust chain. If it is found that the RECs found originate from non local identities to the node then multiple messages are sent back for each non-local identity with an expanded trust chain containing the trust of the local identity in the respective non-local identity.

Finally if the hop counter in the request is not 0, the request is disseminated to the neighborhood. Each request sent to a neighborhood member contains a trust chain that is expanded with a link containing the trust of the local user in that particular member.

5.4.3 Trust Management Component (TMC)

The TMC handles evaluating the user's transitive trust in the REC issuers after receipt of the REC responses, updating the user's trust in the REC issuers after the feedback step, and handling and updating the expertise information. Multiple trust chains are evaluated here which is discussed further in section **5.6.**

The TMC keeps track of the trust in each identity it has been in contact with, specifically storing the trust in the DMC database in accordance to the trust model. The TMC updates the trust in the identities when it receives feedback on the quality of received RECs according to the trust update algorithm discussed in **5.6** which influences the neighborhood selection the next time a query is received.

Updating the knowledge model does not require any user feedback as was described in the case of REC feedback for the trust updates. Here the knowledge model is simply incremented to correspond with new RECs.

5.5 Trust Updating

Trust update algorithms in general specify how to compute trust from a certain given set of inputs. Due to the complexity of deriving meaning in trust from events, there are different algorithms that may be well suited for different use cases. This section highlights the chosen general trust model as described by UniTEC and used in blockEDU. This contribution by UniTEC shows that, while algorithms to compute trust values are different, they rely on the same data and often come to the same conclusions. With this generic model, we may have the flexibility to understand multiple update algorithms, being able to fine tune the algorithms in blockEDU protocol to best suit the needs of its users.

Reputation of an identity is the average trust the network has for the entity, where trust is derived locally from experiences. There is a global quality to reputation while trust is derived subjectively in the given context area. Here we begin setting up the understanding of a generic trust model.

5.5.1 UniTEC Trust Relationships

Here we define further what trust means in the context of our reputation system as first described by the work done by UniTEC.

Trust measure: This is the measure of quality of the trust relationship ranging from distrust to full trust.

Trust certainty: Based on personal experience, this the measure of confidence in the trustee by the trustor

Trust context: This is defined in section **5.2.1** as the context areas.

Trust directness: This specifies between direct and referred trust. If Alice takes a course from Bob, this would be a direct trust. While if Alice was told to trust Bob by Clark who took a course from Bob, this is referred trust.

Trust dynamics: This defines the way in which trust is updated over time.

5.5.2 Generic Trust Model

The *trust measure* is defined in the interval [0,1]. Complete distrust is represented by 0 while complete trust is represented with 1. *Trust certainty* is likewise measured in the same interval.

Context areas are not necessarily independent of each other. For example the fields Mathematics, Computer Science and Physics share many of the same concepts. It can be seen that physics and computer science have a *part-of* relationship with mathematics. This measure may not work for all areas so UniTEC formalizes a distance measure between context areas in the interval [0,1]. A distance close to 1 represents a high dependency, for example the distance between math and computer science may be a 0.6 while the distance between math and physics may closer with a 0.7. Do to the subjective nature of trust, these distances may be modified by each user to accommodate their own personal views. Doing this allows us to spread the impact of trust updates in one area throughout the whole reputation system.

This update semantic can however not carry over to our trust in the consensus algorithm, where each course is localized to their context area island. We want a strict definition of trust in regards to whether or not a given course was offered affordably. Figure 5.5 illustrates the graph formed from distance measures in the context areas, where each node's edge is weighted by the perceived distance of the user.

[create context area trust graph graphic, highlight the courses as islands with no edges]

5.5.3 Dynamics

Here we describe the change in trust as feedback of an experience with an oracle, course, educator, or student happens. The metric used to measure the quality of the information received is again a real number between [0,1] where a 1 represents the best quality.

Trustors can specify a *trustor confidence* in their own offered information in the same interval where 1 represents complete confidence that the information they are recommending is correct. Each request for information is a transaction where a *transaction utility* can specify max utility with a 1.

The age of the recommendations and quality of trustees changes over time, thus *experience aging* is reported in two ways, a feedback window that has a specified number of experiences

where older ones fall off and an aging factor that weights newer experiences with a higher value in the range [0,1], 1 being the newest. Finally the *related context areas* play a role in the trust update for experiences where the semantic distance as described in section **5.5.2** are accounted for.

5.5.4 Update Algorithm

UniTEC presents a basic trust update algorithm based on geometric learning:

$$T_{new} = (1 - (a \cdot c \cdot d)) \cdot T_{old} + (a \cdot c \cdot d) \cdot E$$

The new trust is calculated from the old in each context area and the new experience *E* is influenced by an aging factor *a*, with the recommender's own confidence *c* in the recommendation and a distance factor *d* as reported by the distance measure between the context area in question and the original context area of the recommendation in the context dependency graph illustrated in Figure **5.5.2**.

The trust confidence vector that contains the last n experiences expresses a certainty in the trust measure. This is calculated by consolidating the number of experiences and the variability of values into a single value in the range of [0,1].

A simple fading factor *f* algorithm is used where time is partitioned into discrete units. If within a time interval one or more experiences are made then there is no fading, otherwise trust will drop linearly to the minimal trust value in 1/*f* time units.

5.6 Trust Transitivity

This section describes how trust is derived in the trust chain. Trust transitivity means that if Alice trusts Bob, who in turn trusts Clark, then Alice will also trust Clark to some degree. Since a REC can be found through multiple trust chains, UniTEC in collaboration with Prof. Audun Josang from Queensland University of Technology in Brisbane and Elizabeth Gray from Trinity College Dublin, Ireland provide a method for finding the best path.

It is important to separate *referral trust* and *function trust*. Where Alice trusts Bob's recommendation that Clark is a good math teacher would be a referral trust and Alice trusting Clark by having taking one of his previous math courses would be a functional trust. The context area in this example is the same, trusting Clark to be a good math teacher. In relation to the context area it is defined that the last link in the trust path must be at least a subset of all of the previous trust links. An example being that Alice trusts Clark to be a good math teacher because she also trusts Bob to be a good dean of mathematics.

5.6.1 Normalisation of Trust Measures

The objective here is to retain the whole trust graph and normalize the computed trust measures in order to maintain consistency.

The original UniTEC algorithm for transitivity is used by the blockEDU protocol. Previously in section **5.3.3** we discussed how RECs and their trust chain are collected by requests. To find the transitive trust, UniTEC uses an approach similar to Eigen trust where the product of the individual trusts are calculated. The strongest trust chain takes precedence and direct function trust is the highest order of trust possible.

$$T = \prod_{i=1}^{n} T_i$$

The protocol simplifies normalisation by removing loops and dependencies from the graph between source and target parties that results in a direct series-parallel graph that eliminates the need for normalisation. We compute all possible paths from a source to a target and select a subset of those paths and select the most appropriate one to discern the best possible trust.

5.7 Defending Against Sybil Attacks

Sybil attacks are a common threat to distributed systems where it is cheap and easy to create multiple identities. The most common way to discourage such attacks is to put some cost on the creation of nodes.

The blockEDU protocol has a reputation system to identify malicious participants. However attacks are still possible. An attacker may register a new identity, place themselves in the knowledge contract as a highly knowledgeable identity in many context areas, and once requests are received start issuing false RECs. The disadvantage here to the attacker is the TMC where their recommendations are not yet trusted by the network. An attacker may choose to gain trust by behaving reputably or copying valuable RECs from others and later turn malicious, but this is time consuming. The amount of damage a node could incur by turning malicious is still costly to the network. The feedback mechanism will detect this switch to malicious behavior, but only after the damage is done.

5.7.1 Approach to solve the attack

The reputation system is linked to Ethereum so that there is a payment mechanism to increase the costs of performing bad transactions in the blockEDU network. In order to build trust, the entities must perform actual financial transactions in Ether. There is already a cost incurred by

nodes when they have to upload data to the knowledge contract, store pointers, report to the course contracts as oracles or facilitate course contract creation. Therefore building reputation on the blockEDU protocol does cost a real world value that could prevent attackers from gaming the system.

6 Distributed Cloud Storage - Orc Network

[cite Orc]

A persistent data storage network is needed for both educators and students to store course content and reputation records. The Orc Network offers [Have at it Gordon :)]

7 Ethereum Contracts

Ethereum is a decentralized network of nodes that process transactions and maintain consensus on general state. Ethereum is used in the blockEDU protocol to maintain the state of reputation, consensus, and data. It will provide a mechanism for unlocking courses and course content, it maintains the identities of the users in the system, and it provides the registries that provide pointers to the current state of the applications running on the protocol.

7.1 Ethereum Shortcomings

Ethereum is used as the network to provide a decentralized point of authority on the state of the blockEDU protocol. Storing data in the ethereum network is expensive. [gather details on op storage costs per byte of hex data] and every transaction that updates the state of a contract costs a miner fee for computing the new state. There will be a large amount of transactions to this state machine and as a method for offsetting the costs we propose two solutions.

7.1 a State Channels

State channels are a method of offseting the costs of state updates to contracts. This is done in the same manner that micropayment channels work in the Lighting network of bitcoin [cite lighting]. In the case of state channels, the state is not limited to that of only financial transaction but opened to the broader updates of general state.

The largest amount of state updates to the blockEDU protocol are found when we are updating the knowledge contract for finding RECs, updating the various registries that store the oracles

and new courses, and when the data pointers for forums and course related information needs to be updated.

Instead of broadcasting all of these updates directly to the network, we hold on to signed updates locally that are hash locked with their counterparty. This way these signed broadcasts can be sent to the contract when intermediate state can be thrown out. An example of this is when student and teacher are sending homework assignments back and forth that are stored in the orc network. Instead of broadcasting the pointer to the state head of the data, they will receive the updates on the side network and only broadcast the final state of their homework interaction once the assignment is completed. This also works for the communication forums where students involved in a thread are collectively updating the pointer to the latest state of the thread. Only when a thread is closed will this pointer be sent to the contract as method of permanent storage.

7.1 b Linked Data pointers

The Orc Network provides meta information as to the location of uploaded data in the distributed network. This meta information is updated in the state of the contracts as a way to offset the data storage costs of large data such as written recommendations on the blockchain. These pointers may point to either encrypted personal file locations, or public shared data.

The meta information needed to locate a file on the Orc Network is as follows.
[This is the information needed to find the location of the farmers in the bridge database, I need to research the meta info needed to contract the farmer directly storing the user's contract. Alternatively we can keep these semantics as we are designing bridges. A choice should be made here on how we want to design these pointers]

```
 * @param {String} id - Unique bucket ID
 * @param {String} file - Unique file ID
```

[revise this for SSTORE costs]
Each string is a X bytes of X hash algorithm. The maximum registry size for the Ethereum VM is 32 bytes so concatenation of these two fields fits into one register with an MLOAD instruction from the contract state. The total cost for a X byte MLOAD instruction is X units of gas. [fill in the Xs]

```
pragma solidity ^0.4.11;

contract OrcDataPointer {

    address owner;
    bytes32 pointer;

    modifier onlyOwner() {
        if (msg.sender != owner) throw;
        _;
    }

    /// Create ownership of this pointer to data.
    function OrcDataPointer() {
        owner = msg.sender;
    }

    /// resets the pointer.
    function removePointer() onlyOwner {
        pointer = 0x0;
    }

    /// Update the given pointer to a new state of the data.
    function updatePointer(bytes32 _pointer) onlyOwner{
        pointer = _pointer;
    }

    /// returns the pointer data
    function getPointer() onlyOwner returns(bytes){
        bytes memory _pointer;
        assembly {
            _pointer := sload(add(_pointer, 24))
        }

        return _pointer;
    }
}
```

Figure 7.1 Simple contract to store pointers to the orc network.

As shown in Figure **7.1** a simple method in a contract can be used to load pointers to Orc data onto and read from the Ethereum blockchain to reduce the cost of storing actual data. The above example uses a permissioned modifier to ensure the community is not able to destroy data pointers, and thus the data itself, if they do not own it.

7.2 Issuer Contract

The issuer contract is the mechanism in which educators are paid for mining and maintaining consensus. It is registered as the owner of the total supply of EDU tokens and has the ability to issue tokens.

The issuer contract contains a mapping of all active course contracts. When an educator initiates an issuance request this contract instantiates the given course contract keyed with the educators identity. It then determines if the reputation system has enabled the educator to receive EDU by checking the course state and reported affordability and quality metric.

Determining the amount of tokens issued to the educator is two-fold. It first reads the current max issuance amount as outlined in section **4.2.5** of the deflationary token model. It then also

contacts the knowledge contract to get the overall reputation of the teacher in the range [0,1] as provided by the oracles. The amount awarded is computed as described in section **4.2.6**.

[create visual for this simple equation?]

7.3 Course Contracts

The course contracts make up the structuring of a decentralized education. This can be a single siloed course, or these courses may depend on each other to form a more complex sequence.

7.3.1 Course Instantiation

An educator is prompted by a CMS to fill in the necessary fields for creating a course before the contract is deployed to the network by a facilitating node. Here the educator must determine the max enrollment size, the prerequisites required, the Unix epoch of when the enrollment period will end, the Unix epoch of when the course will end, the minimum number of students required, and link to the syllabus stored in Orc. The course has a data upload that stores the course information beforehand to get the Orc pointer and autofills this into the request.

When the course is created, a call is made to the global course registry contract that places it in the mapping for advertising it to students. At this point students may begin registering until the designated enrollment period ends. If the course does not meet the minimum number of students as set by the educator, the course contract will self destruct.

[layout course fields]

7.3.1 Enrollment and Fees

When a course is created by an educator, the student may find it via the CMS that consults the course registry contract and attempt to enroll. The student must decide how much of an EDU fee they are willing to pay during this process. There is an enrollment time where the educator is given the ability to chose students who have supplied a higher fee as defined by the provided timestamp during instantiation. There are two mappings in the course contract for enrollment, one for potential students and one for confirmed students.

7.3.3 Content Pointers

Each course contract will have a storage for pointers to the Orc network for course content. Educators get a pointer to the course content state for uploading assignments. The students mapping contains addresses to Orc pointer to the latest state of the student's data. This pointer is updated when students need to upload data to the course such as homework assignments or

engage in forum activity. As new assignments are rolled out during the course, the state of this content is updated by the educator updating the pointer in the contract. There are permissions set on these pointers such that only educators and students can update state changes to their respective pointers.

7.3.4 Dependent Course Contracts

Organizations have the ability to create sequential courses for full degree paths. An example being the calculus 1 through 3 series in traditional universities. When creating a course contract, educators have the ability to designate what courses are required by an identity before enrolling. This is done by supplying a list of other course contract addresses to the course in question.

When a student uses their identity to be placed into the enrollment mapping for a course, a preloaded list of course contracts will be consulted. If these students are not stored in the state data for the list of courses as having completed the course prerequisite, their enrollment transaction will fail.

7.4 Registry Contracts

Registry contracts are used as way for the other contracts in the system to understand prerequisites, the bootstrapping process of the knowledge provider to find recommenders, the total list of active courses, and the pointers that hold each user's distributed file bucket that holds their data.

7.4.1 Knowledge Provider Registry

The knowledge contract will contain registries partitioned by knowledge areas to return experts in the desired area. This is to defeat the bootstrapping problem of a node in the system not being able to find RECs for given context areas. If a node is not well connected to the network, and they have no RECs stored locally, they may consult this contract as a method of trying to find a node that may have a REC stored locally.

The knowledge contract also stores information on the oracles in the network. It provides the overall registry that each course contract will consult to determine if an identity providing reputation information to the course is valid.

7.4.2 Course Registry

This contract provides the CMS with a way of aggregating all of the deployed contracts on the blockEDU network. Each time the CMS is loaded for the user, this contract will be consulted to provide a view to students of what courses are active to be enrolled in, and what courses have

completed and are still open to review course content. If it was decided during instantiation by the educator that the course material should live for free, the course will fall into a completed mapping where students may consult the course contract and find the content data pointer still intact. Assuming the Orc network is still being paid to maintain these records, the student will be able to review the content.

This registry is also used by the issuer contract when calculating block rewards.

7.4.3 Public Profile Registry

The public profile contract holds the identity information for the users of the CMS. This data is public names, addresses, profile pictures and textual data. This can be replaced with other standard identity registries like uPort.

7.5 University Contracts (Future Implementation)

While not currently supported by the first CMS application built on-top of the blockEDU protocol, we are pre-planning for a future CMS that is geared toward traditional universities. These contracts are an attempt to build a fully decentralized university that will rival private for profits in their quality of education. Universities are complex organisms so we are challenging the online education model first. It is not impossible for organisations to utilize the blockEDU protocol in it's current state as they may handle their governance off-chain and still bootstrap the reputation system into theirs.

8 Aristotle - The first CMS

The first CMS is built by blockEDU on top of the protocol token, reputation system, and Orc Network. It is not necessarily the only system that can be built. Other organization that are currently in existence are welcome to replace their backend with the blockEDU protocol and auxiliary systems like the Orc Network distributed storage method presented here. [enter name] serves as an example of this may be done.

A market of courses similar to Udemy is being built here. All courses will be listed with their uploaded information about what the course is. The reputation scores for each educator in the market can be determined by issuing a request in the local view of the course.

The CMS will allow users to log in, check their courses, sign up for new ones, create a course, and more. Forums will foster discussion in specific topics, and links to outside materials will be recognized for faster integration into the student's workflow. For example, GitHub links will auto-pull stats for the repo in question, relevant info about the repo, and other pertinent

information. Wikipedia links will pull up a quick preview of the page and quick links to the article's major citations. These features are aimed at making the students study sessions more productive, helping them find information faster and more reliably.

8.1 Client Application Features and Scope

Structure
The client application will be built in JavaScript with the Vue framework, possibly with React in some parts if necessary.

The client application will be a static app, packaged up and bundled using Webpack, minified and kept with performance in mind. Vue is often noted as the fastest framework right now, and offers a good tradeoff between speed of development and app performance.

Deployment
The backend application will be a Node server, serving up a local app of Vue.

The backend application will be deployed in a Docker container, using Rancher to orchestrate container deployments and uptime. This backend will connect to the BlockEDU Bridge [cite next section] to interact with the blockEDU platform.

Data persistence will be achieved with MongoDB at the Bridge layer.

Mobile
We will package up a native implementation for mobile and desktop in either a Vue native framework such as Weex or a React framework React-native.

Authentication

8.2 Requirements

This is a broad, technology-agnostic overview of the requirements of the client-side and server-side applications.

Client-side

- Video upload
- File upload
- Sign up, login, and authentication
- Course management
    - Syllabus creation
    - Milestones in each course

    - Course creation
    - Course deletion
    - Maintenance and editing
- Reputation and feedback management per course
- Course discussion
- Forums and discussion areas
- User reputation
- Administration and moderation

Server-side

- Authentication
    - Key management
    - Passwords
- blockEDU network interface and interaction layer

8.3 Future CMS Implementations

The first CMS will be a marketplace for courses and discussions, which we see as valuable, but short-term. The long-term goal of our platform is to compete with universities and colleges in research, offer degrees provably tied to people, and focus on education as a human right that should be accessible from all corners of the world and to any who desire it.

9 The blockEDU Bridge

Platform Bootstrapping

To fulfill ease of use requirements, the blockEDU organization hosts a bridge server that connects the CMS applications to the blockEDU network. This bridge facilitates HTTP requests from the users not running nodes to the full nodes in the network. As a point of centralization, this is a weakness in the protocol and a temporary solution. Full documentation is provided and anyone can run a bridge in the event that the blockEDU organization is attacked or no longer exists.

Long Term Decentralization

Eventually, we would like to stray away from this centralized bridge model as new methods for handling private keys, browser side encryption, and light client APIs to blockEDU nodes are built, but for bootstrapping the platform, some amount of centralization will have to occur.

We have been exploring the possibility of full nodes running in the browser, this way every teacher and student can participate directly in the network to fully decentralize the reputation network. At the current time of writing this, browsers do not make for good nodes. Mobile

applications are not long living nodes in the DHT, there is no stream cipher encryption in browser to allow fast private data uploads, and communication through browsers relies on centralized webrtc servers.

10 Possible Attack Vectors

10.2 Course DDoS

This comes from the ability of educators with low reputation being able to register a large amount of courses and reduce the current block reward for the system. It may be a good idea to require a bond or some course creation cost for educators.

11 Conclusions

[TODO: Get citation sources]