ORIE 4580/5580
Course Project

1 Outline

A team at Llenroc University are studying the feasibility of creating a **BigRedCoins** (or BRC) system: a virtual currency that can be used among students to create an internal marketplace for goods and services. At the heart of the BRC system is an online ledger (or *blockchain*), that is maintained by all the students in a decentralized way (i.e., without any central authority that manages student accounts and verifies transactions). Each student on the system has access to the blockchain, and can use their computers to verify BRC transactions that are posted by other students – this verification process is necessary to ensure that students don't promise BRCs to others that they do not have themselves. Designing this blockchain correctly is critical to the success of the BRC system.

The team has already got experts in software engineering, networking and cryptography, and are confident they can program the BRC system correctly. However, they are worried that once the system starts getting used by all the students at Llenroc, then it may get very slow, leading to long delays in processing the transactions. They want some help from a simulation expert to figure out how to configure the system.

The BRC team are aware that you may not have a background in cryptography and blockchains – they however feel that the basic technology can be understood quite easily, and that they have expertise to handle any technical aspects of the system. What they need help with is understanding how different system parameters will affect the speed and throughput of the system. For this they are turning to your team for help.

The aim of the project is to build a (simplified) simulation model of the BRC blockchain, and study its steady-state behavior. Before you jump in to making your model, the BRC team recommend that you get some basic background into the blockchains by going through the following:

- Video tutorial on basics of blockchains (h/t 3Blue1Brown)
- Khan Academy tutorial on Bitcoin (this is more detailed then we need, but the lectures on *proof of work* and the *blockchain* may be of interest if you want to understand the inner working of the system.)
- Article on the economics (and operations) of the blockchain, which the BRC team thinks provides a useful model for the system. You do not need to read this to start (we summarize the main features below), but it may be of interest to you if you want to understand more details.

Once you start getting the basic idea of the system, you are ready to try and build your simulation model!

2 Detailed Description

At a high level, BRCs are a virtual currency which students can exchange. An important component of enabling such a currency is to be able to verify transactions, i.e., ensure that when any person A transfers some money to another person B, then A really has the money available in his account for transferring. The interesting feature of a blockchain-based currency is that this verification is done not by a central authority, but by the participants of the system.

To join/leave the system, students buy/sell BRCs from an automated exchange – once they are on the system, however, the manner in which BRC transactions are verified are completely decentralized. In particular, the students provide free computing cycles on their personal computers towards verifying the transactions – this also allows them to earn BRCs, based on transaction fees associated with each transaction.

2.1 The Blockchain

The centerpiece of the BRC system is the blockchain, which is a ledger keeping track of all past transactions between students. Its main features are as follows.

- Each BRC user has an account with a unique id, and the only transaction that the blockchain tracks are transfers between accounts. The system starts off with a single special BRC_Core account, which has a initial (large) number of BRCs – subsequent account creations and account closings are implemented as transfers between students and the BRC_Core account.
- Each transaction comprises of a sender id, a receiver id, a transfer amount and a transaction fees. For a new transaction to be added to the blockchain, the participants first submit a transaction request to the system. Subsequently, this request is verified by other students (who are referred to as *miners* in cryptocurrency jargon), and noted together with other transactions as part of a *block*. The transfer however occurs only when the block with the transaction is added to the end of the blockchain.
- Each block on the blockchain can contain up to K transactions. To add a block on the blockchain, after verifying the constituent transactions, miners need to solve a certain cryptographic 'puzzle'. This deters miners from adding false blocks on the blockchain, as being able to solve the puzzle is essentially a matter of luck, and hence no miner can be sure that they can add a block on the blockchain when they want to. Assume that as soon as a block is created, all other miners immediately become aware of it.
- The BRC system is designed so that any miner has exactly the same computing power (so you can not try and solve the puzzle faster then anyone else). As a result, each new block is created by a uniform random active miner.
- New blocks are added to the blockchain according to a Poisson process with rate μ – this rate is controlled (by adjusting the difficulty of the cryptographic puzzles) to be constant independent of the number of miners.

2.2 Transactions

An important part of your model are transactions – in particular, their rate, amount and fees.

- Transactions arrive according to a Poisson process with rate λ. Though in practice this varies with time, to keep things simple in your model, you can assume that the rate is constant.
- Each transaction specifies a transfer amount a and a transaction fee b, which the user chooses. You can assume that each arriving transaction corresponds to a random integer number of BRCs, and that each transaction also chooses a random percentage of this amount to be its fee.
- Pending transactions are either part of the current block, or are waiting to be processed in a common transaction queue.
- The order in which pending transactions get processed from the transaction queue is determined by the miners (see below). A crucial performance metric is how long it takes for transactions to get processed. The BRC designers want to understand this delay, and also the trade-off between how much fees is offered and the corresponding delay.

2.3 Mining

Finally, we also need a simple model how miners behave in the BRC system.

- The blockchain is maintained by the BRC members acting as miners. Active miners use free computer cycles to solve the cryptographic puzzles required for adding a block of transactions to the blockchain. Active miners also observe the pending transactions pool, and always keep an updated copy of the blockchain.
- The BRC team has designed their app to ensure that each active miner contributes a fixed amount of computing power to the system for verifying transactions, and hence each active miner has an equal chance of adding the next block. A miner who mines a new block is rewarded with the transaction fees of transactions included in that block.
- Miners are strategic in choosing which pending transaction to combine into a block, preferring to choose ones with the highest offered fees. For simplicity, assume that as soon as a block is added to the blockchain, all miners pick the K pending transactions with the highest fees, and start trying to verify and add that block. If there are less than K pending transactions, then all those are combined to make a block. Transactions that arrive while a block is being mined must wait till the block is added to the blockchain.
- Denote the number of active miners by N. Although having large N is crucial for the system, to simplify things, you can assume that N is large enough for the system to be reliable and secure, and for each miner to be small. Instead, you should use your simulator to predict how big N can be while the system remains cost-effective (see objectives below).

2.4 Simulation Model

Your task is to build a DES simulator to capture the main details of the BRC system. Some guidelines regarding what you want the model to capture:

- The main system parameters are the block mining rate μ, the number of transactions per block K, and the transaction arrival rate λ. Your basic model should keep these fixed; in a more advanced model you should be able to allow these to vary.
- The BRC team wants you to focus on the parameter range where all transaction are processed. In particular, it recommends you start with $\lambda = 120/hr$ and choose values of μ and K such that all transactions are processed. The only technical constraint imposed by the cryptographic protocols is that μ can be at most 30 blocks/hr.
- For the basic model, you can assume that the arriving transaction amounts are uniformly distributed as integers between 5 and 25, and are i.i.d.; moreover, you can assume each transaction chooses its fees to be either 1% or 2% of its amount with equal probability.

Performance Measures and Objectives There basic metrics of interest to BRC:

- Throughput of system in terms of number of transactions and BRCs
- Delay experienced by transactions
- The rate of fees collected

For all of the above, the team is interested in the mean as well as the distributions, and moreover, the behavior of these metrics with changes in design parameters μ and K.

Steady-State Performance Measurement For a system such as this which is non-terminating, a natural objective is to study it in steady-state, with λ and transaction amounts/fees as described above, and understand how the choices of μ and K affect the above metrics.

Understanding the economics of the BRC System Once you have a working simulator, you can use it to tackle harder questions about how the BRC system may look like in real life, when users decide to post transactions based on the system delays, and miners choose to mine only if the fees cover their costs. Understanding this in detail is difficult, but you can take some steps towards this by thinking about the following questions:

- How does the delay experience by a transaction vary based on the offered fees? In particular, if a BRC user told you that they needed to get their transaction processed within 1 hour – can you recommend them a fee they should offer?
- On the other hand, suppose a miner pays a fixed cost of c BRCs/hr for mining. How many active miners can be in the system for mining to be profitable?

In reality in any system like this, participants are always trying to figure out how to use the system to get the best outcome at the lowest cost. Economists have a way of thinking of such systems in *equilibrium* – where each user acts selfishly to maximize their own rewards, while being aware of what others may do. Understanding questions like the ones above can help to understand the equilibrium that emerges from such behavior. However, to really understand the economics of the system, you want to think of the following question (*Warning: This is completely optional, and is more difficult then the rest of the project...*):

- For the basic model, you assume that each transaction has a fee percentage that is uniformly chosen, no matter what the transaction amount is. However, does this make sense in practice? For example, if you are posting a transaction of 100 BRCs, would you ever offer 2% as fees?

  A more natural assumption is that each user has a *value* for the transaction being processed, which is a non-increasing function of the delay it takes in processing. A common way to model this is to say that a sender with transaction amount a who experiences a delay D obtains a transaction value of $v(a, D) = 1 - \kappa(a)D$, where $\kappa(a)$ is some cost per unit delay for a user with transaction amount a ('time is money'...). For simplicity, you can assume $\kappa(a) = \kappa$ is a constant, although in reality, you may think that transactions with higher amounts may have higher delay costs. Now what each user wants to do is to choose a fees so as to maximize its expected transaction value, knowing that everyone else is trying to do the same. How can you handle this in your model?

If you are interested in learning more, the article mentioned before uses the above model to explore these questions in depth.

3 Deliverables

As with any project, this is a fairly deep and open-ended set of questions. The above suggestions provide a way for you to get started – however, the ultimate aim is to make sure you are able to build a simulator for a complex system, use it to explore design choices and their impact on participants, and finally, able to communicate your findings to a client. We now provide some guidelines on how you should approach the project and report.

3.1 Coding Recommendations

We strongly recommend building the simulation model in Python. The basic BRC system is complex enough that although it could possibly be implemented in Simio, it would require a lot of hacks.

Please try to plan your project and design your code to be modular. Use classes/functions wherever possible to ensure that the code is easy to build on. Also make sure the code is well commented and documented, as this will help a lot when working with teammates.

Once you have a basic back-end simulation platform, you should consider using separate Jupyter notebooks for testing and validation, and for analysis and design of the system.

3.2 Deadlines

Your deliverable is in two parts: the simulation model, and the project report.

Your simulation model (code + demo) is due on **Monday, December 3**. All teams must demonstrate their projects **during the recitation times**. You should come as a team and demonstrate your code to us – in particular, you should demonstrate the steady-state analysis. We will suggest corrections if necessary.

Your final reports are due on Monday **December 10 at 4:30pm** on **CMS**. Late reports will be penalized by 20% per day for up to 2 days (so the latest you can hand in a project is 4.30pm on Wednesday December 12. Be sure to give the name and Net IDs of all team members on the title page or equivalent.

3.3 Report Guidelines

Your project report should be made up of the main report for the BRC executive team, and appendices to allow other simulation specialists to repeat your analysis. The main part of the report should be designed for non-specialists. You can put the technical details in the appendices.

Your project report should not exceed **15 pages** with **1.5-spacing** and at least **12pt font**. Appendices are not included in the page count. Make sure that your report flows well. Below are possible sections you may want to consider in your project report – if you feel uncomfortable with any portion of the outline suggested below except the Executive Summary, then feel free to modify the suggested outline.

- Executive summary: This is the part of your report that the project managers will read. Give a synopsis of what you were asked to do, what you did and what you discovered. Do not just outline what is in the report. Focus on the results. Think what you would want to know if you were a high-level manager and this was all that you read. This section should be at most two pages.
- Problem description: Describe the current situation and the goal of the study.
- Modeling approach and assumptions: Describe the simulation model in general terms. This section is for intelligent non-specialists, who want to ensure that your model is reasonable and captures all of the important aspects.
- Model verification: Explain in broad strokes how you checked that your model was correct as implemented. The goal is to convince decision makers that your model can be trusted to correctly answer the questions it was designed to answer.

- Model analysis: Describe how you used your model to answer the questions in which the BRC team is interested. Analyze the results and make recommendations. Think about sensitivity analysis for quantities that are uncertain or approximate, and how robust your recommendation is to such changes. Your readers may not be simulation specialists and cannot possibly know all that simulation has to offer. Therefore be creative and try to go above and beyond what has been asked for.
- Conclusions: Briefly recap your findings and recommendations.
- Appendices: These are intended for a simulation specialist who has training at the level of having taken ORIE 4580/5580 in the past. Readers of this section understand simulation concepts and have some familiarity with Python. Think separately about the model and the implementation, and give any critical details about the model, and its implementation in Python. In particular, include enough details so that someone can replicate your work without actually looking at your code.
- Your reports will be graded based on the following criteria:
  1. Does the report answer all of the questions of interest?
  2. Is the model built cleanly and correctly?
  3. Is the model appropriately verified?
  4. Does the report explain the results and recommendations in a clear fashion?
  5. How well written is the report? Does it have a logical progression that describes the problem, modeling approach, analysis, results and recommendations?

4 Notes

- A large-scale project rarely has completely specified deliverables, and often it is up to you to decide how much to do, and how to manage your time and effort. If you have doubts over any particular details of the problem, you should try to make assumptions about it, and use your report to outline and justify these assumptions. We will of course be available to help with this, but may choose not to specify all details.
- A common mistake is to assume that the project is only an exercise in model building. The project is an exercise in model building, model analyzing and report writing. Make sure that you spend at least 30% of your time on each one of these activities.
- The quality of the report (grammar, spelling, structure, quality of the writing, confining technical specifics to the appendix) is very important.
- We are interested in average performance measures, as well as histograms. Also, think about (and report) other performance measures that could be of interest.
- Please remember that this is a bigger task than a typical homework assignment, so needs much more time. Moreover, this is a team project – make sure you work well as a team to get things done. Using tools like version control is particularly useful, specially if you are working separately.