Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών & Μηχανικών Υπολογιστών
Ροή Υ: Κατανεμημένα Συστήματα

(9ο Εξάμηνο)

Αναφορά Εξαμηνιαίας Εργασίας

Ομάδα 33:

Δημήτρης Καλαθάς 03118016
Δημήτρης Μπακάλης 03118163
Γρηγόρης Παπανικολάου 03118649

Εισαγωγή

Στην εργασία αυτή θα υλοποιήσουμε το σύστημα noobcash, το οποίο βασίζεται στην τεχνολογία του blockchain. Το συγκεκριμένο σύστημα θα έχει 5 ή 10 κόμβους (ανάλογα με το πείραμα που θέλουμε να εκτελέσουμε) που θα επικοινωνούν μεταξύ τους και θα επιτελούν όλες τι βασικές εργασίες ενός κόμβου-miner σε ένα πραγματικό σύστημα blockchain. Για να πετύχει αυτό υλοποιήσαμε τις 5 βασικές κλάσεις σε κάθε κόμβο: block, node, blockchain, wallet, transaction, ένα rest api (μέσω της βιβλιοθήκης flask της python) για την επικοινωνία μεταξύ τους, καθώς και ένα cli για να μπορούμε κάθε στιγμή να ζητήσουμε πληροφορίες ή να δημιουργήσουμε ένα νέο transaction σε κάθε κόμβο.

Το Σύστημα

Block: Αποτελεί τη βασική μονάδα ενώς blockchain και περιέχει όλες εκείνες τις πληροφορίες ενός block αλλά και που θα πρέπει να αποθηκευτούν τελικά στην αλυσίδα.

Περιέχει τις εξής πληροφορίες:

- *Index:* Το id κάθε block.
- *Previous_hash:* Το hash του προηγούμενου block στην αλυσίδα.
- *Timestamp:* Ο χρόνος δημιουργίας του block.
- *Nonce:* Ο αύξων αριθμός που προστίθεται στο hash μέχρι να γίνει valid με βάση το difficulty.
- *Transactions:* Τα transactions που έχει το block.
- *Hash:* Το Hash.
- *Confirmed:* Ένα flag του block, που σηματοδοτεί το αν έχει valid hash.

Περιέχει τις εξής συναρτήσεις:

- *myHash:* Η οποία δημιουργεί το hash του κάθε block.
- *Add_transaction:* Προσθέτει ένα transaction στο block.

Blockchain: Ανεβαίνοντας ένα επίπεδο αφαιρετικότητας πιο πάνω συναντάμε το Blockchain που, όπως υποδηλώνει το όνομα του, αποτελείται από την αλυσίδα των blocks και μερικές συναρτήσεις.

Περιέχει τις εξής πληροφορίες:

- *Chain:* Τα blocks που έχουν γίνει valid και μπαίνουν στην αλυσίδα.

Περιέχει τις εξής συναρτήσεις:

- *Greate_genesis_block:* Δημιουργεί το genesis block (μόνο ο bootstrap).
- *Add_block:* Προσθέτει ένα block στην αλυσίδα.
- *Print_contents:* Εμφανίζει το περιεχόμενο για όλα τα blocks τις αλυσίδας.

Wallet: Εδώ υπάρχουν πληροφορίες για το wallet του κάθε node.

Περιέχει τις εξής πληροφορίες:

- *Public_key:* Το public key κάθε node, που περιέχεται στο ring.
- *Private_key:* Χρησιμοποιείται για να υπογράφονται τα transactions.
- *Address:* Η διεύθυνση κάθε node που συμπίπτει με το public key του.
- *Transactions:* Λίστα με τα transactions που έχει κάνει ο node.
- *UTXOs:* Λίστα με τα UTXOs όλων των nodes.

Περιέχει τις εξής συναρτήσεις:

- *Add_transaction:* Προσθέτει ένα transaction στην λίστα που έχει το κάθε wallet.

- *Balance:* Υπολογίζει το υπόλοιπο κάθε node σύμφωνα με τα UTXOs.

Transaction: Κάθε transaction περιέχει πληροφορίες για μεταφορά χρημάτων από ένα wallet σε ένα άλλο.

Περιέχει τις εξής πληροφορίες:

- *sender_address:* Το public key του wallet από το οποίο προέρχονται τα χρήματα.
- *receiver_address:* Το public key του wallet στο οποίο θα καταλήξουν τα χρήματα.
- *amount:* Το ποσό που θα μεταφερθει.
- *transaction_id:* Το hash του transaction
- *transaction_inputs:* Λίστα από Transaction Inputs (ids των transactions, των οποίων τα utxos χρησιμοποιούνται).
- *transaction_outputs:* Λίστα από Transaction Outputs (χρήματα που μεταφέρονται + ρέστα).
- *signature:* Υπογραφή που αποδεικνύει ότι ο κάτοχος του wallet δημιούργησε αυτό το transaction.

Περιέχει τις εξής συναρτήσεις:

- *to_dict:* Μετασχηματισμός των πληροφοριών του transaction σε Dictionary.
- *sign_transaction:* Υπογράφεται το transaction με το private key του wallet.

Node: Η κλάση αυτή αποτελεί το πυρήνα του συστήματος μας και αντιπροσωπεύει τον κάθε χρήστη καθώς και τις ενέργειες που θα ακολουθηθούν στην αρχή ώστε το σύστημα να λειτουργήσει ομαλά.

Περιέχει τις εξής συναρτήσεις:

- *create_transaction:* Δημιουργείται ένα νέο transaction που περιέχει όλα τα στοιχεία που απαιτούνται και καθορίζεται το αν είναι εφικτό να γίνει (βάσει του balance). Στη συνέχεια, αφαιρούνται ενημερώνονται αναλόγως τα transaction outputs και τα UTXOs των κόμβου.
- *broadcast_transaction:* Το transaction γίνεται broadcast σε όλους τους κόμβους και ενημερώνονται τα UTXOs τους.
- *verify_signature:* Επαληθεύεται η υπογραφή του transaction αμέσως μετά τη λήψη του.
- *mine_block:* Η συνάρτηση αυτή καλείται μόλις ένα transaction εληφθεί και επαληθευτεί από κάποιον κόμβο και το τελευταίο block του blockchain είναι γεμάτο. Σε περίπτωση που συμβεί αυτό, όλοι οι κόμβοι αρχίζουν να κάνουν mining ταυτόχρονα, δοκιμάζοντας διαφορετικές τιμές της μεταβλητής nonce και hashάροντας το block μέχρι το hash που θα προκύψει να αρχίζει από έναν συγκεκριμένο αριθμό από μηδενικά (proof of work). Ο αριθμός αυτός καθορίζεται από τη σταθερά difficulty.

- *broadcast_block:* Μόλις βρεθεί ο κατάλληλος nonce, ο κόμβος κάνει broadcast το επαληθευμένο block σε όλους τους υπόλοιπους κόμβους, ώστε να σταματήσουν κι αυτοί το mining (consensus).
- *validate_block:* Καλείται από τους nodes κατά τη λήψη ενός νέου block (εκτός του genesis block). Επαληθεύεται ότι (a) το πεδίο current_hash είναι πράγματι σωστό και ότι (b) το πεδίο previous_hash ισούται πράγματι με το hash του προηγούμενου block.
- *validate_chain:* Καλείται από τους νεοεισερχόμενους κόμβους, οι οποίοι επαληθεύουν την ορθότητα του blockchain που λαμβάνουν από τον bootstrap κόμβο.
- *resolve_conflict:* Καλείται όταν ένας κόμβος λάβει ένα block το οποίο δεν μπορεί να κάνει validate, επειδή το πεδίο previous_hash δεν ισούται με το hash του προηγούμενου block στο blockchain. Σε αυτή την περίπτωση το consensus επιτυγχάνεται, μέσω μιας διαδικασίας, όπου επιλέγεται το blockchain με τη μεγαλύτερη αλυσίδα σε μήκος και γίνεται broadcast σε όλο το δίκτυο.

Rest & Client: Το rest αποτελείται από ένα από API με τις αναγκαίες λειτουργηκότητες για το σύστημα μας και τα κατάλληλα μηνύματα και responses αναλόγως τον τρόπο που αυτό καλείται. Κάθε χρήστης ξεκινά ένα rest πρόγραμμα στη διεύθυνση που ακούει δίνοντας τα κατάλληλα ορίσματα σχετικά με τα παιδιά και τον πατέρα. Από εκεί και πέρα το σύστημα τρέχει αυτόματα μέσω του node. Ο client αποτελείται καλεί συγκεκριμένα endpoints για τις ζητούμενες λειτουργικότητες.

Πειράματα

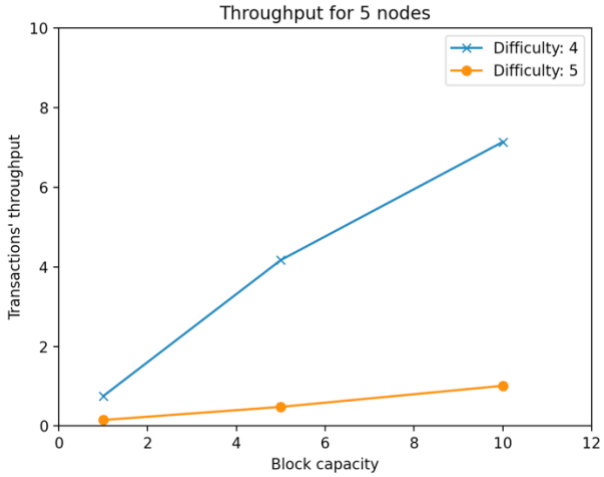
Το blockchain που έχουμε δημιουργήσει θα εξεταστεί τόσο με 5, όσο και 10 nodes, προκειμένου να βγάλουμε συμπεράσματα σχετικά με το scalability του. Επιπλέον, για το evaluation της επίδοσης του μοντέλου θα εξάγουμε ορισμένα στατιστικά από την εκτέλεση του κώδικα. Πιο συγκεκριμένα, θα μετρήσουμε το μέσο throughput (αριθμός transactions που εκτελούνται ανά δευτερόλεπτο) και το μέσο block time (χρόνος που απαιτείται για το mining ενός block).

Αρχικά, θα παρουσιάσουμε τα αποτελέσματα για τους 5 nodes:

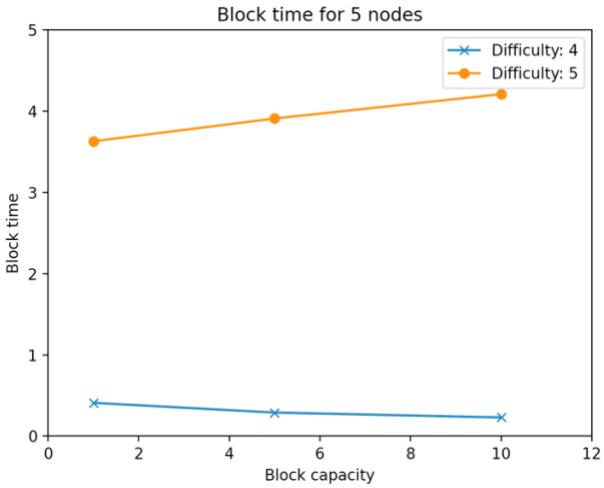
	Capacity						
Difficulty		1		5		10	
		Throughput	Block time	Throughput	Block time	Throughput	Block time
	4	0.75 tr/s	0.41s	4.17 tr/s	0.29s	7.14 tr/s	0.23s
	5	0.15 tr/s	3.63s	0.48 tr/s	3.91s	1.01 tr/s	4.21s

Σχηματικά, οι παραπάνω μετρήσεις αποτυπώνονται ως εξής:

Για το throughput:



Για το block time:

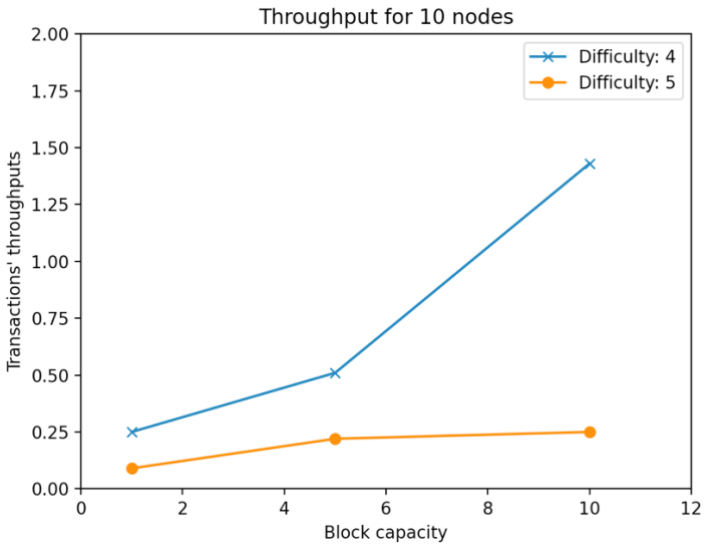


Ακολουθούν τα αντίστοιχα αποτελέσματα για τους 10 nodes:

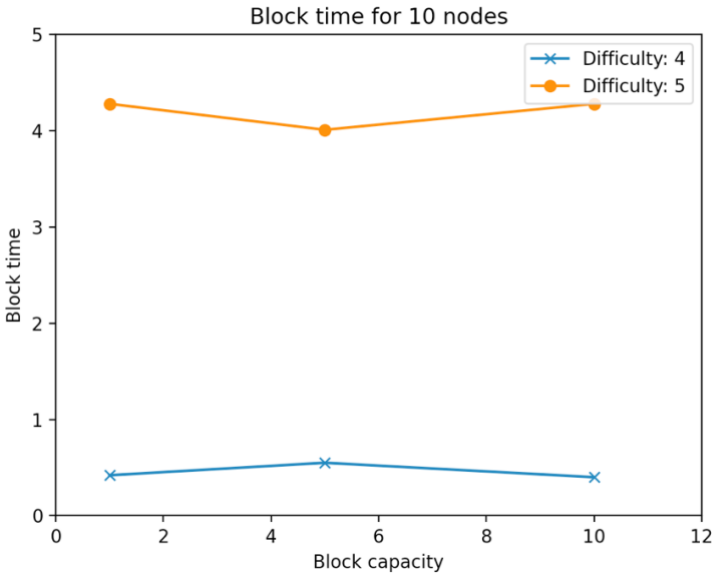
	Capacity					
Difficulty		1		5		10	
		Throughput	Block time	Throughput	Block time	Throughput	Block time
	4	0.25 tr/s	0.42s	0.51 tr/s	0.55s	1.43 tr/s	0.4s
	5	0.09 tr/s	4.28s	0.22 tr/s	4.01s	0.25 tr/s	4.28s

Σχηματικά, οι παραπάνω μετρήσεις αποτυπώνονται ως εξής:

Για το throughput:

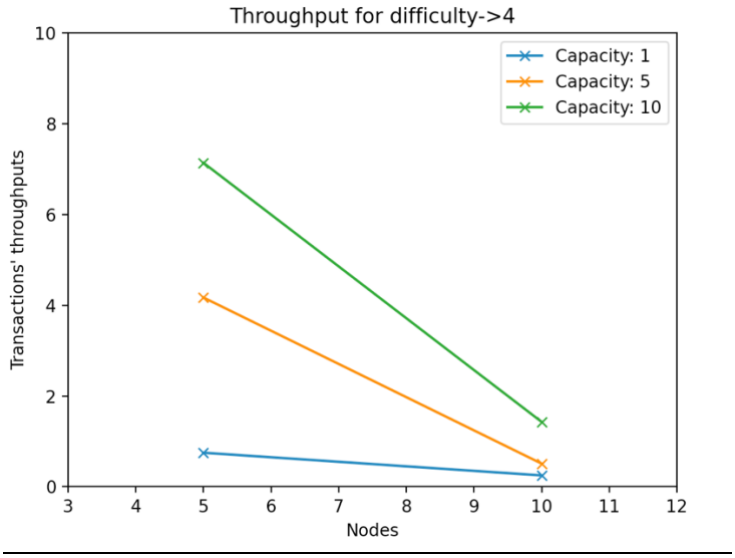


Για το block time:

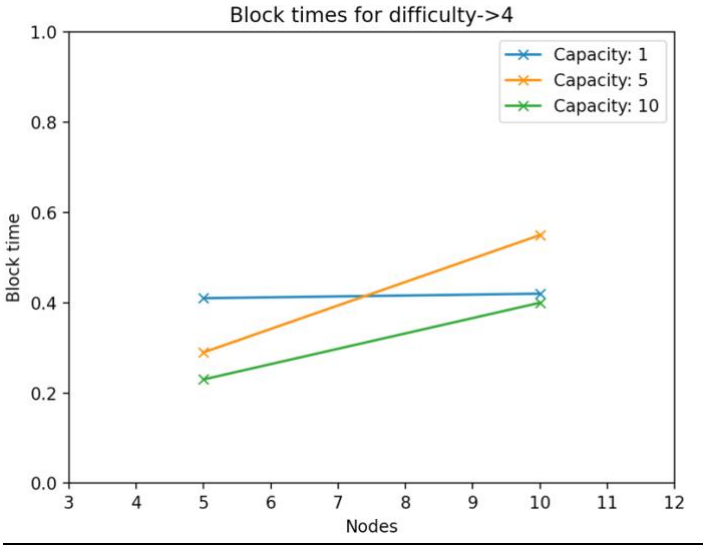


Συγκριτικά, για τους δύο αριθμούς από nodes προκύπτουν τα ακόλουθα αποτελέσματα:

Για το throughput (με difficulty 4):



Για το block time (με difficulty 4):



Παρατηρήσεις:

- Αρχικά, όπως ήταν αναμενόμενο, τόσο στους 5, όσο στους 10 nodes παρατηρούμε πως το throughput αυξάνεται όσο αυξάνεται το capacity των blocks, καθώς όλο και λιγότερα transactions θα καταλήξουν σε mining κατά τη διάρκεια της προσθήκης τους σε κάποιο block.
- Επιπλέον, σχετικά με το block time, παρατηρούμε σημαντική διαφορά (και στους 5 και στους 10 nodes) ανάμεσα στα πειράματα με difficulty 4 και τα αντίστοιχα με difficulty 5. Το γεγονός αυτό, είναι επίσης, αναμενόμενο, καθώς όσο αυξάνεται το difficulty, η απαίτηση για ένα valid proof γίνεται αυστηρότερη, με συνέπεια να απαιτείται περισσότερος χρόνος για την ικανοποίησή της. Επιπροσθέτως, λογικό είναι το ότι η τιμή του block time παραμένει σχεδόν σταθερή ανά το capacity, καθώς η συχνότητα που λαμβάνει χώρα ένα mining είναι ανεξάρτητη από τη διάρκειά του.
- Τέλος, σχετικά με το scalability του συστήματος, παρατηρούμε πως η επίδραση του πλήθους των κόμβων επηρεάζει κυρίως το throughput, καθώς τα τετραπλάσια transactions πρέπει να ικανοποιηθούν, ενώ το block time αυξάνεται με σχετικά αργό ρυθμό. Ωστόσο, η μείωση του throughput δεν είναι τόσο απότομη, ώστε να είναι απαγορευτική η συμμετοχή παραπάνω κόμβων στο δίκτυο.