ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

Κατανεμημένα Συστήματα
9ο εξάμηνο, Ακαδημαϊκή περίοδος 2021 – 2022

Αναφορά Εξαμηνιαίας Εργασίας

Μάριος Μητρόπουλος 03117078 (marios.mitropoulos@yahoo.gr)

Κυριάκος Παναγιωτίδης 03117206 (kyriakos.8p@gmail.com)

Εισαγωγή – Σκοπός

Το blockchain είναι η τεχνολογία πίσω από τα περισσότερα κρυπτονομίσματα και αποτελεί στην πραγματικότητα μια κατανεμημένη βάση που επιτρέπει στους χρήστες της να κάνουν δοσοληψίες (transactions) μεταξύ τους με ασφάλεια, χωρίς να χρειάζονται κάποια κεντρική αρχή (π.χ. τράπεζα). Πάνω σε αυτή την τεχνολογία βασίστηκε το Bitcoin, που δημιουργήθηκε το 2008 και έμελλε να αλλάξει τον τρόπο διεξαγωγής των συναλλαγών. Στις μέρες μας παρατηρείται αυτή η τάση της αγοράς για εναλλακτική μορφή οικονομιών συναλλαγών, καθώς, πέρα από το Bitcoin (η αξία του οποίου τη στιγμή συγγραφής αυτής της αναφοράς είναι >$40,000), έχει δημιουργηθεί πλέον μια πληθώρα διαφορετικών κρυπτονομισμάτων.

Έτσι, κρίνεται σημαντική η ενασχόλησή μας με τη συγκεκριμένη τεχνολογία. Σκοπός της εργασίας είναι η δημιουργία του noobcash, ένα απλό σύστημα blockchain, όπου θα καταγράφονται οι δοσοληψίες μεταξύ των συμμετεχόντων και θα εξασφαλίζεται το consensus με χρήση Proof-of-Work.

Σχεδιασμός συστήματος

Το σύστημα υλοποιήθηκε με χρήση Python. Για την επικοινωνία των κόμβων μεταξύ τους κατασκευάστηκε ένα REST API. Με αυτό τον τρόπο, οι clients ανταλλάσσουν πληροφορίες μέσω GET και POST requests. Το framework που χρησιμοποιήθηκε για την υλοποίηση του API είναι το Flask της Python, ενώ για το CLI απλή Python.

Κλάσεις

Block

Η κλάση Block αποτελεί ένα instance του Blockchain. Περιέχει ένα πλήθος συναλλαγών όπως αυτό ορίζεται από τη χωρητικότητα (block capacity). Επιπλέον, περιέχει τον αύξων αριθμό του (index), το timestamp της δημιουργίας του και τη λύση του Proof-of-Work (nonce). Με τις παραπάνω πληροφορίες υπολογίζεται μέσω της μεθόδου calc_hash το hash του block, που αποτελεί και το μοναδικό αναγνωριστικό του.

Blockchain

Η κλάση Blockchain αναλαμβάνει την υλοποίηση των μεθόδων που χρειάζονται για την εύρυθμη λειτουργία και συντήρηση της αλυσίδας του κάθε κόμβου. Αποτελείται από μια λίστα με αντικείμενα της κλάσης Block και μεθόδους προσθήκης ενός block στην αλυσίδα (μέθοδος add_block), validation ενός block (μέθοδος validate_block) και validation του blockchain (μέθοδος validate_chain), που καλεί τη validate_block για όλα τα blocks εκτός του genesis.

Wallet

Η κλάση αυτή ορίζει το πορτοφόλι ενός κόμβου. Περιλαμβάνει ένα ζεύγος RSA κλειδιών από το οποίο το public key αποτελεί τη διεύθυνση του πορτοφολιού, ενώ το private key είναι γνωστό μόνο στον κάτοχο του πορτοφολιού και χρησιμοποιείται για την προσθήκη ψηφιακής υπογραφής στις συναλλαγές. Επίσης, περιέχει μια λίστα από unspent transaction outputs (UTXOs), το άθροισμα των οποίων μας δίνει το συνολικό balance του πορτοφολιού (μέθοδος wallet_balance).

Transaction

Η κλάση Transaction περιέχει τις απαραίτητες πληροφορίες για τη μεταφορά χρημάτων από ένα wallet σε ένα άλλο, δηλαδή τα public keys του αποστολέα και του παραλήπτη, το ποσό που μεταφέρεται, τα UTXOs που χρησιμοποιήθηκαν για τη συγκεκριμένη συναλλαγή και τα νέα UTXOs που παράγονται, και ένα μοναδικό ID που παράγεται με hash των πεδίων (μέθοδος calc_hash), όπως αυτό της κλάσης Block. Η δημιουργία της ψηφιακής υπογραφής (μέθοδος sign_transaction) και επαλήθευσής της (μέθοδος verify_signature) γίνεται με χρήση του encryption scheme PKCS1_PSS, ενώ η παραγωγή των transactions outputs γίνεται με τη μέθοδο compute_transaction_outputs η οποία υπολογίζει τα ρέστα για τον αποστολέα και το ποσό για τον παραλήπτη σε μορφή UTXO.

Node

Η κλάση αυτή αποτελεί την πιο καίρια για τη λειτουργία του συστήματος, καθώς χρησιμοποιεί όλες τις προηγούμενες για την υλοποίηση των σεναρίων χρήσης του συστήματος. Ο κάθε κόμβος κατέχει ένα wallet, ένα αντίγραφο του

blockchain, το ring που περιέχει πληροφορίες για τους υπόλοιπους κόμβους συνδεδεμένους στο σύστημα και μια λίστα με pending transactions, δηλαδή transactions που έχουν γίνει validated (είτε εισερχόμενες είτε δημιουργήθηκαν από τον ίδιο τον κόμβο) αλλά δεν αποτελούν μέρος του blockchain καθώς δεν έχει γίνει ακόμα mine ένα block που τις περιέχουν. Η κλάση περιλαμβάνει μεθόδους που χρησιμοποιούνται για την αρχικοποίηση του συστήματος, δηλαδή μεθόδους που καλούνται μόνο από το bootstrap κόμβο (create_genesis_block και register_node_to_ring). Ταυτόχρονα, έχει τις απαραίτητες μεθόδους και endpoints για τη δημιουργία και λήψη συναλλαγών (create_transaction και register_transaction, αντίστοιχα), το mining (mining_handler και mine_block) και τη λήψη νέου block από το δίκτυο (register_block), και την επίλυση συγκρούσεων (resolve_conflicts). Οι παραπάνω λειτουργίες θα μελετηθούν αναλυτικά σε παρακάτω ενότητα.

Εφαρμογή & Command Line Interface (CLI)

Για το τρέξιμο της εφαρμογής πρέπει να εκτελεστεί το αρχείο “rest.py” για κάθε κόμβο, το οποίο λαμβάνει options για το σενάριο προσομοίωσης: το port στο οποίο συνδέεται ο κόμβος (-p), ο συνολικός αριθμός κόμβων (-n), ο βαθμός δυσκολίας του mining (-d), η χωρητικότητα του κάθε block (-c) και αν ο κόμβος είναι ο bootstrap node (-b).

Για την παρακολούθηση εξέλιξης του συστήματος και την επιλογή δημιουργίας συναλλαγής «χειροκίνητα», δημιουργήθηκε ένα CLI με τις εξής εντολές:

• t <recipient_id> <amount>

Δημιουργεί μια νέα συναλλαγή του συγκεκριμένου ποσού από τον κόμβο στον οποίο τρέχει το CLI προς τον κόμβο με το ID που προσδιορίζεται.

• view

Τυπώνει τις συναλλαγές που περιέχονται στο τελευταίο επικυρωμένο block του blockchain.

• balance

Τυπώνει το υπόλοιπο του wallet.

• exit

Τερματίζει το CLI.

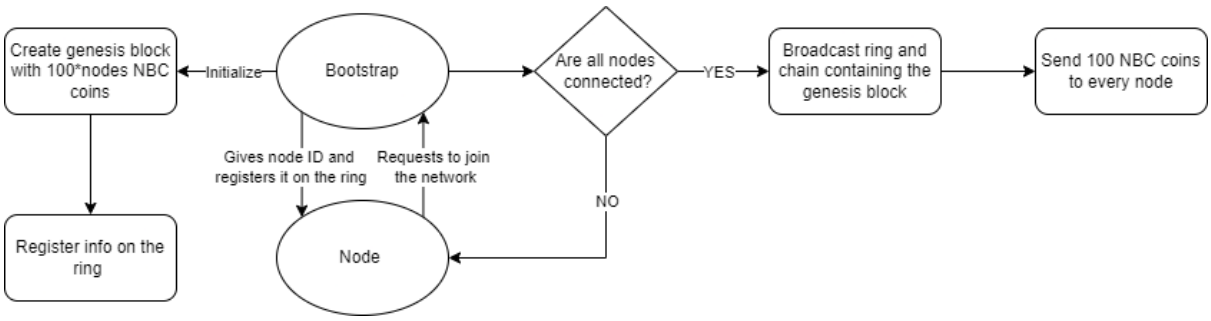
• help <command>

Επεξηγεί την προσδιοριζόμενη εντολή.

Αλληλεπίδραση κλάσεων – Λειτουργίες συστήματος

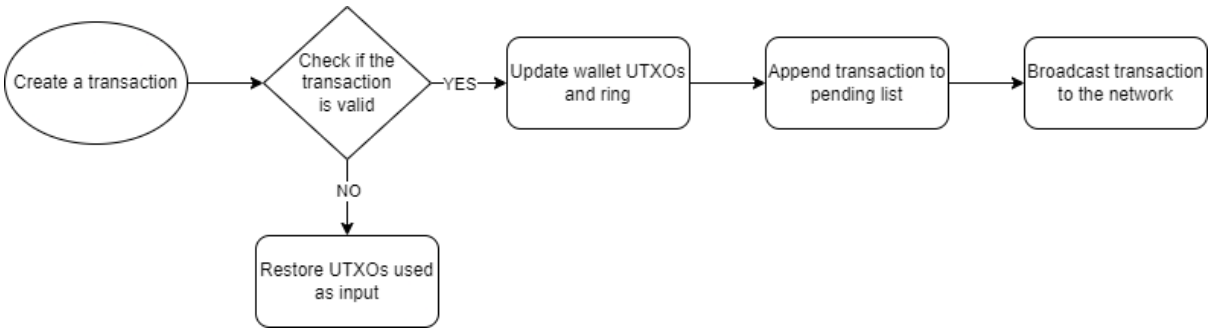
Αρχικοποίηση

Η αρχικοποίηση του συστήματος ξεκινά με την είσοδο του πρώτου κόμβου στο σύστημα, που ορίζεται ως bootstrap node. Ο κόμβος αυτός είναι υπεύθυνος για το διαμοιρασμό των NBC coins σε όλους τους υπόλοιπους κόμβους που θα συνδεθούν, καθώς και για τον ορισμό του ID κάθε άλλου κόμβου. Αρχικά, ο bootstrap node λαμβάνει 100 NBC coins επί το συνολικό πλήθος των κόμβων που θα συνδεθούν σε μια συναλλαγή που καταγράφεται στο genesis block, δηλαδή στο πρώτο block του blockchain που περιέχει αυτή τη συναλλαγή και μόνο (δεν πραγματοποιείται mine). Κάθε κόμβος που συνδέεται επικοινωνεί με το bootstrap node για να λάβει ID (register_node endpoint). Αφού συνδεθούν όλοι οι κόμβοι στο δίκτυο, ο bootstrap στέλνει σε όλους τους κόμβους το αρχικό blockchain και το ring με τις πληροφορίες όλων των κόμβων. Έπειτα, δημιουργεί συναλλαγές ποσού 100 NBC coins προς κάθε άλλον κόμβο. Έτσι, κάθε κόμβος έχει από 100 NBC coins και το σύστημα είναι, πλέον, αρχικοποιημένο.



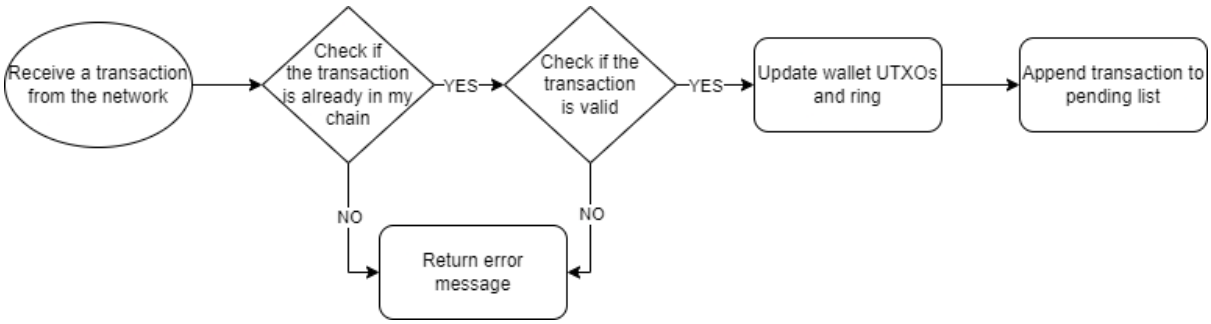
Δημιουργία συναλλαγής

Για τη δημιουργία συναλλαγής (μέθοδος create_transaction), ο κόμβος χρησιμοποιεί UTXOs του πορτοφολιού του ως inputs της συναλλαγής για να συμπληρώσει το ποσό το οποίο επιθυμεί να μεταφέρει. Ύστερα, ελέγχεται η εγκυρότητα της συναλλαγής, δηλαδή αν ο κόμβος έχει αρκετά NBC coins για να δημιουργήσει μια συναλλαγή με το προσδιοριζόμενο ποσό και αν η υπογραφή του αποστολέα (για τη δημιουργία της οποίας χρησιμοποιήθηκε το private key του) είναι έγκυρη. Αν η συναλλαγή δεν είναι έγκυρη, η συναλλαγή δεν πραγματοποιείται και τα UTXOs που χρησιμοποιήθηκαν ως inputs επιστρέφουν στο πορτοφόλι του αποστολέα. Αλλιώς, ο αποστολέας ενημερώνει το πορτοφόλι του και το ring του με τα transaction outputs της συναλλαγής, ένα με τα ρέστα του αποστολέα και ένα με το ποσό του παραλήπτη. Ύστερα, προσθέτει τη συναλλαγή στο pending list του, για να εισαχθεί αργότερα σε block που θα γίνει mined. Τέλος, κάνει broadcast τη συναλλαγή στο δίκτυο για να ενημερωθούν όλοι οι κόμβοι (και ο παραλήπτης).



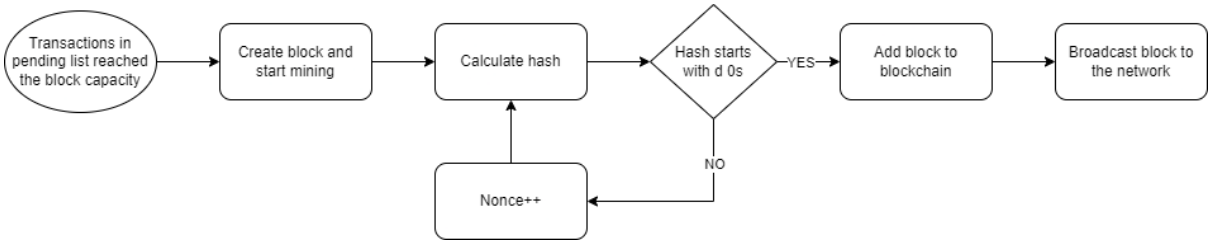
Λήψη συναλλαγής

Όταν ένας κόμβος λαμβάνει μια συναλλαγή (register_transaction endpoint), αρχικά ελέγχει αν η συναλλαγή υπάρχει ήδη σε κάποιο block του blockchain. Στην περίπτωση αυτή ή στην περίπτωση που η συναλλαγή δεν είναι μέρος του blockchain αλλά είναι μη έγκυρη, ο κόμβος δεν χρειάζεται να πραγματοποιήσει κάποια ενημέρωση. Αλλιώς, ο κόμβος ενημερώνει το ring του για τους κόμβους αποστολέα και παραλήπτη, και αν ο κόμβος είναι ο παραλήπτης της συναλλαγής ενημερώνει τα UTXOs του πορτοφολιού του. Έπειτα, προσθέτει τη συναλλαγή στο pending list του, για να εισαχθεί αργότερα σε block που θα γίνει mined.



Διαδικασία mining

Κάθε κόμβος έχει ένα thread που αναλαμβάνει τη διαδικασία του mining και τρέχει παράλληλα με την υπόλοιπη εφαρμογή (mining_handler). Όταν συμπληρωθεί ο αριθμός συναλλαγών που ορίζεται από τη χωρητικότητα του block, δημιουργείται ένα block με συναλλαγές που αφαιρούνται από το pending list μέχρι να συμπληρωθεί το capacity. Έτσι, ξεκινά το mining (μέθοδος mine_block), δηλαδή η αναζήτηση ενός hash που ξεκινάει από d μηδενικά, όπου d είναι η δυσκολία του mining που έχει οριστεί. Για να υπολογίζεται νέο hash σε κάθε επανάληψη αυξάνεται η μεταβλητή του block nonce κατά ένα, η οποία στο τέλος του mining μας δείχνει πόσες προσπάθειες χρειάστηκαν για την εύρεση κατάλληλου hash. Αν ένας κόμβος ολοκληρώσει το mining ενός block το προσθέτει στο blockchain του και το στέλνει σε όλους τους υπόλοιπους κόμβους του δικτύου για να ενημερωθούν.

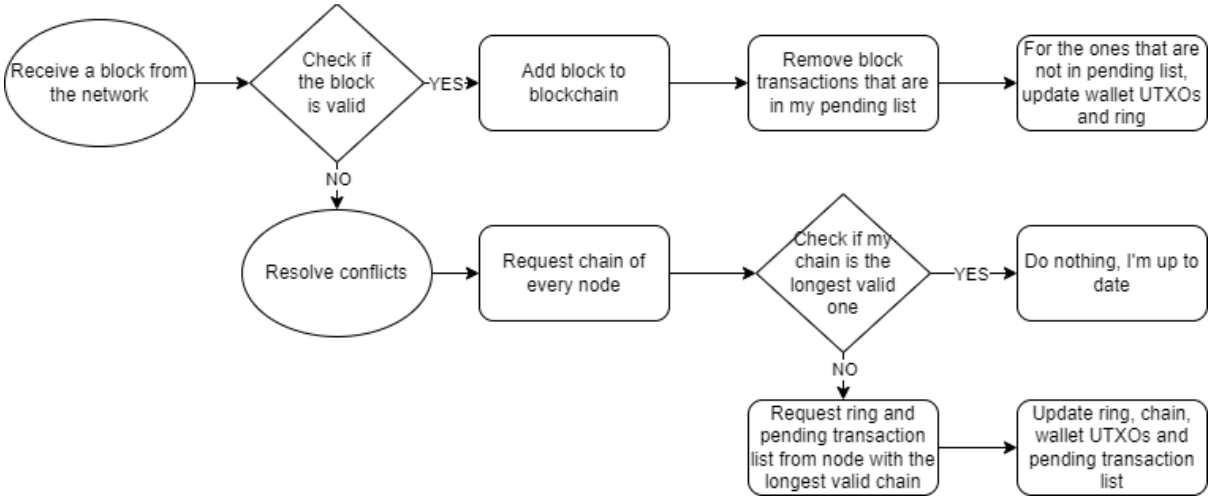


Λήψη block – αλγόριθμος consensus

Όταν ένας κόμβος λαμβάνει ένα block (register_block endpoint), σταματά τη διαδικασία mining και εξετάζει αν το block που έλαβε είναι έγκυρο, δηλαδή αν μπορεί να εισαχθεί ως το επόμενο block στο blockchain του κόμβου. Ο έλεγχος αυτός αποτελείται από τον έλεγχο του index του block που λήφθηκε που θα πρέπει να είναι το επόμενο αυτού του τελευταίου block στο τρέχον blockchain, καθώς και από τους ελέγχους ότι το πεδίο current_hash είναι πράγματι σωστό

και το πεδίο previous_hash ισούται πράγματι με το hash του προηγούμενου block. Αν το block είναι έγκυρο, ο κόμβος προσθέτει το block στο blockchain του και εξετάζει τις συναλλαγές που περιέχονται σε αυτό. Από αυτές, κάθε συναλλαγή που υπάρχει ήδη εντός της λίστας με τις pending συναλλαγές αφαιρείται από αυτή, ενώ αυτές που δεν υπάρχουν προστίθενται στη λίστα και ενημερώνονται το πορτοφόλι και το ring.

Αν το block δεν είναι έγκυρο, εκτελείται ο αλγόριθμος consensus (μέθοδος resolve_conflicts). Ο αλγόριθμος ξεκινά με broadcast αιτήματος του κόμβου προς τους υπόλοιπους για το blockchain τους. Αφού λάβει όλες τις απαντήσεις, ελέγχει αν το δικό του blockchain είναι το μεγαλύτερο. Αν ναι, δεν χρειάζεται να λάβει κάποιο action καθώς έχει ένα έγκυρο blockchain. Διαφορετικά, στέλνει αίτημα στον κόμβο από τον οποίο έλαβε τη μακρύτερη έγκυρη αλυσίδα για να πάρει το ring και τη λίστα με τα pending transactions, αφού ο κόμβος έχει μη έγκυρη αλυσίδα και συνεπώς η εγκυρότητα των υπόλοιπων στοιχείων δεν είναι εγγυημένη. Για το λόγο αυτό, λαμβάνεται ως παραδοχή η ενημέρωση του κόμβου αυτού από έναν κόμβο με έγκυρα στοιχεία, παρά την πιθανή απώλεια μερικών συναλλαγών. Άλλη υλοποίηση θα μπορούσε να είναι η επαναφορά των ενημερώσεων των συναλλαγών των μη κοινών block μεταξύ του blockchain του κόμβου και του μακρύτερου έγκυρου, και ύστερα ενημέρωση για κάθε block της μακρύτερης έγκυρης αλυσίδας μετά από το τελευταίο κοινό block.



Πειράματα – Αξιολόγηση συστήματος

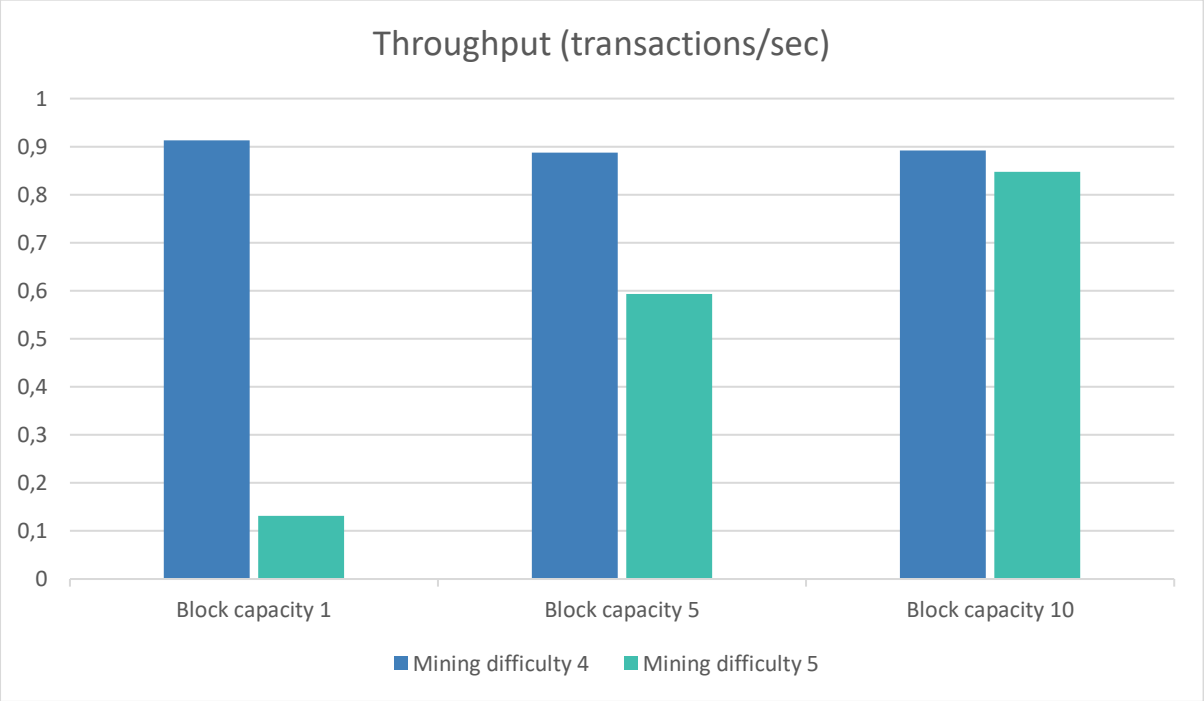
Για την αξιολόγηση του συστήματος, η εφαρμογή στήθηκε σε υποδομή του εργαστηρίου ~okeanos-knossos. Χρησιμοποιήθηκαν 5 Virtual Machines (VMs), στο καθένα από τα οποία έτρεχε η εφαρμογή ενός client. Στην περίπτωση των 10 clients, σε κάθε VM έτρεχαν 2 clients, αλλά η ανεξαρτησία τους ήταν εγγυημένη αφού κάθε VM που χρησιμοποιήθηκε είχε 2 CPUs.

Σημειώνουμε ότι στη διεξαγωγή των πειραμάτων χρησιμοποιήθηκε η συνάρτηση sleep της βιβλιοθήκης time με σκοπό να μην πλημμυρίσει το δίκτυο με συναλλαγές, και να είναι ελαφρώς πιο ρεαλιστικό το σενάριο υπό εξέταση. Συνεπώς, οι τιμές των μετρικών επηρεάζονται σε κάποιο βαθμό από την παραδοχή αυτή. Για τη μέτρηση των απαραίτητων τιμών χρησιμοποιήθηκαν συναρτήσεις για την καταγραφή των χρονικών στιγμών σε αρχεία (write_block_time, write_mine_time, και write_validated_transactions). Χρησιμοποιώντας αυτά τα αρχεία υπολογίστηκαν οι μετρικές μέσω του script metrics.py που επισυνάπτεται στο παραδοτέο (στο φάκελο logs).

Απόδοση

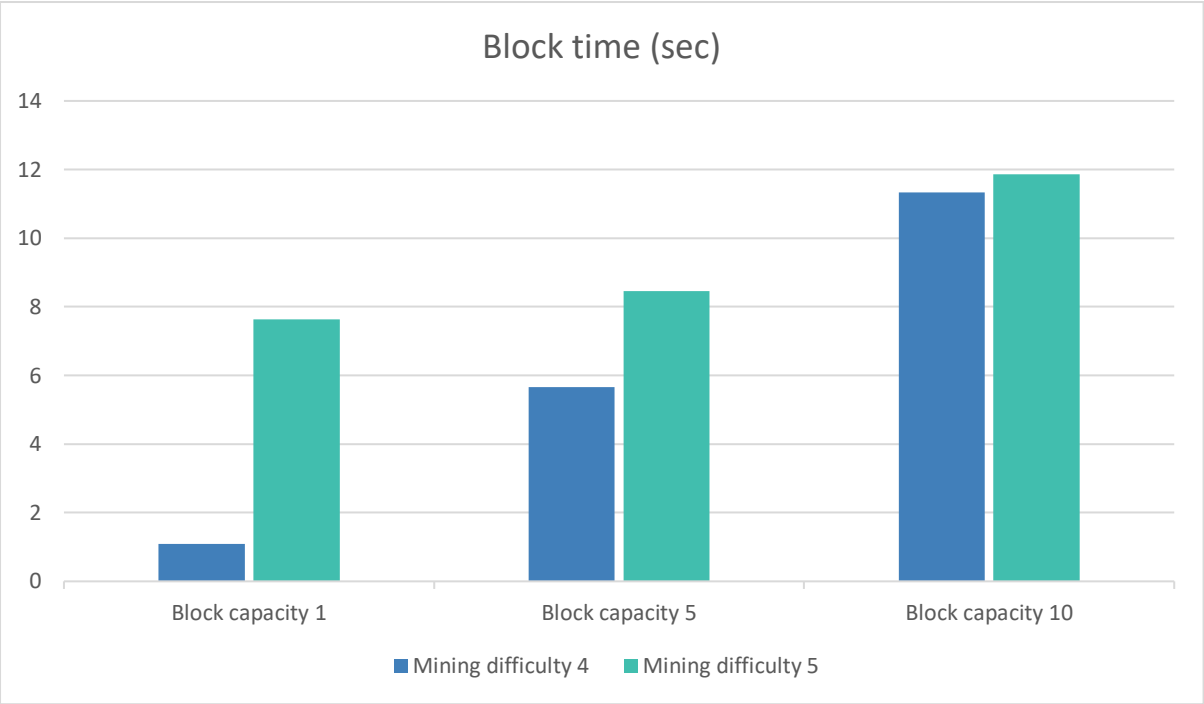
Για τη μέτρηση της απόδοσης του συστήματος καταγράφτηκαν η ρυθμαπόδοση (throughput), δηλαδή πόσες συναλλαγές εξυπηρετούνται στη μονάδα του χρόνου, και το block time, δηλαδή το μέσο χρόνο που απαιτείται για να προστεθεί ένα νέο block στο blockchain. Το πείραμα αυτό διεξήχθη με 5 κόμβους συνδεδεμένους στο δίκτυο, με τις συναλλαγές από το αντίστοιχο δοσμένο αρχείο. Οι συναλλαγές εκτελέστηκαν ταυτόχρονα για όλους τους clients. Εξετάσαμε τις περιπτώσεις που προκύπτουν με mining difficulty 4 και 5 και block capacity 1, 5 και 10.

Το throughput υπολογίστηκε ως το πλήθος των validated transactions προς το συνολικό χρόνο εκτέλεσης, δηλαδή από τη στιγμή που εστάλη η πρώτη συναλλαγή μέχρι να γίνει mine και το τελευταίο block. Συνεπώς, μετριέται σε transactions/sec και αποτελεί μέτρο για την ταχύτητα εξυπηρέτησης συναλλαγών του συστήματος. Παρακάτω παρουσιάζονται τα αποτελέσματα σε ραβδόγραμμα.



Παρατηρούμε ότι η ρυθμαπόδοση του συστήματος για mining difficulty 5 είναι μικρότερη σε όλες τις περιπτώσεις από αυτή για mining difficulty 4. Αυτό είναι αναμενόμενο, καθώς ο απαιτούμενος χρόνος για mining αυξάνεται εκθετικά σε σχέση με τη δυσκολία. Έτσι, ενώ για mining difficulty 4 η ρυθμαπόδοση παραμένει σχεδόν σταθερή σε σχέση με το block capacity, για mining difficulty 5 η αύξηση της ρυθμαπόδοσης με την αύξηση του block capacity είναι ξεκάθαρη, αφού το πλήθος των συναλλαγών είναι σταθερό και άρα τα blocks που δημιουργούνται είναι λιγότερα. Συνεπώς, απαιτείται λιγότερος χρόνος για mining συνολικά και η δυσκολία του δεν επηρεάζει τόσο το αποτέλεσμα, όπως φαίνεται για block capacity 10.

Το block time υπολογίστηκε ως ο μέσος όρος, όλων των clients, του χρόνου που απαιτείται για να προστεθεί ένα νέο block στο blockchain. Συνεπώς, μετριέται σε sec. Παρακάτω παρουσιάζονται τα αποτελέσματα σε ραβδόγραμμα.



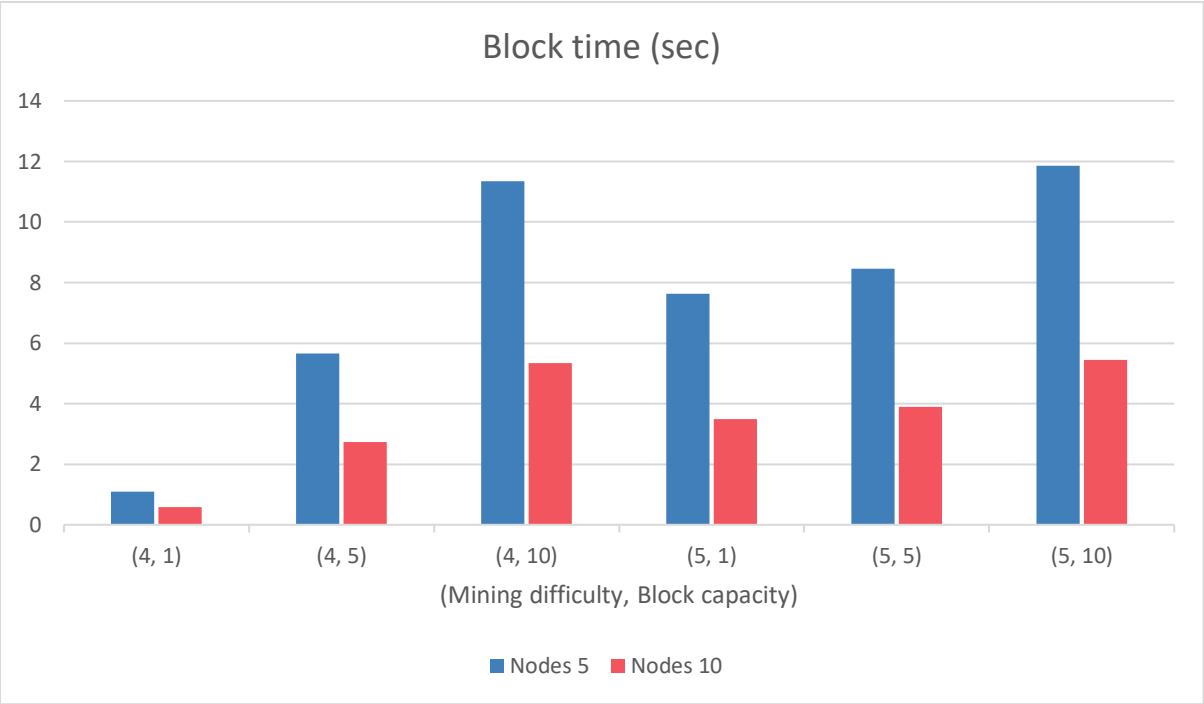
Παρατηρούμε ότι το block time του συστήματος για mining difficulty 4 είναι μικρότερο σε όλες τις περιπτώσεις από αυτή για mining difficulty 5. Αυτό είναι αναμενόμενο για τους λόγους που εξηγήσαμε και παραπάνω, δηλαδή λόγω της εκθετικής αύξησης του χρόνου απαιτούμενου για mining ανάλογα με τη δυσκολία. Βλέπουμε, επίσης, ότι με την αύξηση του block capacity αυξάνεται και το block time, γεγονός που είναι λογικό αφού λόγω της μικρής αδράνειας που έχουμε προσθέσει στο σύστημα (sleep) υπάρχουν στιγμές που δεν έχει γεμίσει ακόμα το επόμενο block πριν τελειώσει η διαδικασία mining του τρέχοντος. Αυτό το effect δεν παρατηρείται τόσο στην περίπτωση για mining difficulty 5 και block capacity 1 και 5, καθώς το mining απαιτεί αρκετό χρόνο και, μέχρι να ολοκληρωθεί, το επόμενο block έχει σχεδόν συμπληρωθεί.

Κλιμακωσιμότητα

Στη συνέχεια, επαναλάβαμε τα παραπάνω πειράματα για 10 κόμβους συνδεδεμένους στο δίκτυο. Συγκρίνοντας τα δύο αποτελέσματα θα μπορέσουμε να καταλήξουμε σε κάποιο συμπέρασμα για την κλιμακωσιμότητα του συστήματος. Παρακάτω παρουσιάζονται τα αποτελέσματα για τη ρυθμαπόδοση σε ραβδόγραμμα.

Παρατηρούμε ότι και για τις δύο περιπτώσεις η μορφή του διαγράμματος είναι παρόμοια. Ωστόσο, βλέπουμε ότι για 10 κόμβους η ρυθμαπόδοση του συστήματος είναι περίπου διπλάσια σε σχέση με αυτή του αντίστοιχου σεναρίου για 5 κόμβους. Αυτό ήταν αναμενόμενο, αφού έχουμε διπλασιάσει τους κόμβους και, συνεπώς, τους miners. Έτσι, οι απαιτούμενοι χρόνοι για mining έχουν μειωθεί, γεγονός που αυξάνει τη ρυθμαπόδοση, καθώς οι συναλλαγές εξυπηρετούνται γρηγορότερα. Επίσης, η ρυθμαπόδοση αυξάνεται λόγω του διπλασιασμού του πλήθους συναλλαγών που διεξάγονται στο πείραμα με τους 10 κόμβους, καθώς σε κάθε πείραμα κάθε κόμβος εκτελεί 100 συναλλαγές.

Παρακάτω παρουσιάζονται τα αποτελέσματα για το block time σε ραβδόγραμμα.



Παρατηρούμε ότι και για τις δύο περιπτώσεις η μορφή του διαγράμματος είναι παρόμοια. Ωστόσο, βλέπουμε ότι το block time για 10 κόμβους είναι περίπου το μισό σε σχέση με αυτό του αντίστοιχου σεναρίου για 5 κόμβους. Όπως αναλύσαμε και προηγουμένως, αυτό ήταν αναμενόμενο, καθώς έχουμε πλέον 10 miners. Έτσι, η διαδικασία του mining γίνεται στατιστικά στο μισό χρόνο. Ταυτόχρονα, το διπλάσιο πλήθος συναλλαγών μειώνει το χρόνο αναμονής συμπλήρωσης του επόμενου block. Όλα αυτά συμβάλλουν στη μείωση του χρόνου που απαιτείται για να προστεθεί ένα νέο block στο blockchain στο μισό.

Συμπερασματικά, βλέπουμε ότι το σύστημά μας λειτουργεί γρηγορότερα με 10 clients, με σχεδόν διπλάσια ταχύτητα, σε σχέση με 5. Έτσι, μπορούμε να αποφανθούμε ότι είναι κλιμακώσιμο (scalable), αφού με την είσοδο περισσότερων miners το σύστημα γίνεται ταχύτερο. Σημειώνουμε ότι η αύξηση των κόμβων δεν επιφέρει και μεγαλύτερους απαιτούμενους χρόνους για τη μετάδοση των πακέτων επικοινωνίας μεταξύ των κόμβων, καθώς για τη διαδικασία αυτή χρησιμοποιούνται threads. Ωστόσο, μπορεί να προκληθεί bottleneck στο δίκτυο από την ταυτόχρονη μετάδοση πακέτων σε σημαντικό αριθμό κόμβων. Άρα, έχουμε ένα trade-off μεταξύ υπολογιστικής ισχύς και χρόνο εκτέλεσης.