Κατανεμημένα Συστήματα

ΑΝΑΦΟΡΑ

ΕΛΕΥΘΕΡΙΟΣ ΧΑΪΔΕΜΕΝΟΣ 03117035
ΣΙΑΝΑ ΑΝΑΣΤΑΣΙΑ ΆΝΝΑ 03117113

Σχεδιασμός Συστήματος

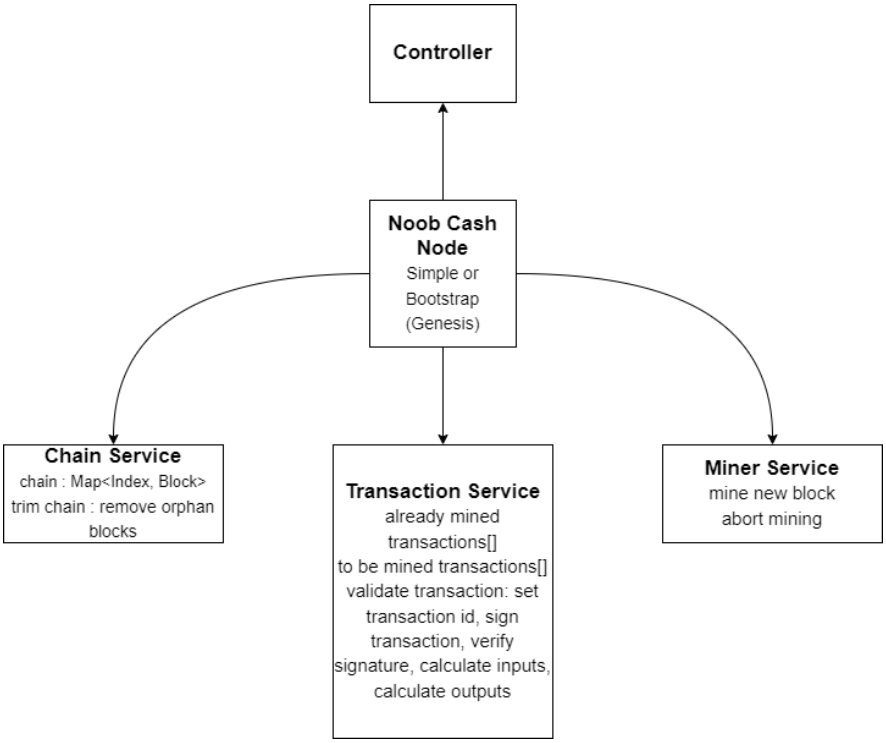
Στα πλαίσια της φετινής εργασίας Κατανεμημένων Συστημάτων, υλοποιήσαμε το απλό σύστημα blockchain του noobcash. Χρησιμοποιήσαμε nodejs και κατασκευάσαμε μία single thread εφαρμογή με την οποία μπορεί κανείς να επικοινωνήσει μέσω ενός REST API.

Για την εκκίνηση της εφαρμογής, στέλνουμε σε κάθε κόμβο ένα αίτημα POST στο endpoint /ignite το οποίο αρχικοποιεί το σύστημα. Έτσι, δημιουργείται το genesis block, το πρώτο του νέου blockchain που δίνει ίση ποσότητα αρχικών χρημάτων στο wallet του κάθε κόμβου.

Πλέον, κάθε χρήστης μπορεί να δημιουργήσει μία νέα συναλλαγή κάνοντας ένα αίτημα POST στο endpoint /transactions. Το αίτημα αυτό πυροδοτεί ένα broadcast αίτημα PUT στο endpoint /transactions, ώστε να γίνει γνωστή η συναλλαγή σε όλους τους κόμβους του συστήματος. Ο κάθε κόμβος τοποθετεί τη νέα συναλλαγή σε ένα heap που διαθέτει για να κρατάει προσωρινά τις συναλλαγές που δεν βρίσκονται σε κάποιο block (οι συναλλαγές ταξινομούνται με βάση το timestamp τους), ελέγχει αν υπάρχουν αρκετές για να δημιουργηθεί ένα block, τις ελέγχει για εγκυρότητα, υπολογίζει τα νέα outputs και ξεκινάει το mining. Όταν τελειώσει το mining, το κάνει broadcast στους υπόλοιπους κόμβους και το τοποθετεί στο τέλος του blockchain του. Αν, στη διάρκεια του mining, μεγαλώσει το blockchain, γίνεται abort το mining, τα transactions που περιέχει το block επανατοποθετούνται στο heap και ξεκινάει την διαδικασία δημιουργίας νέου block.

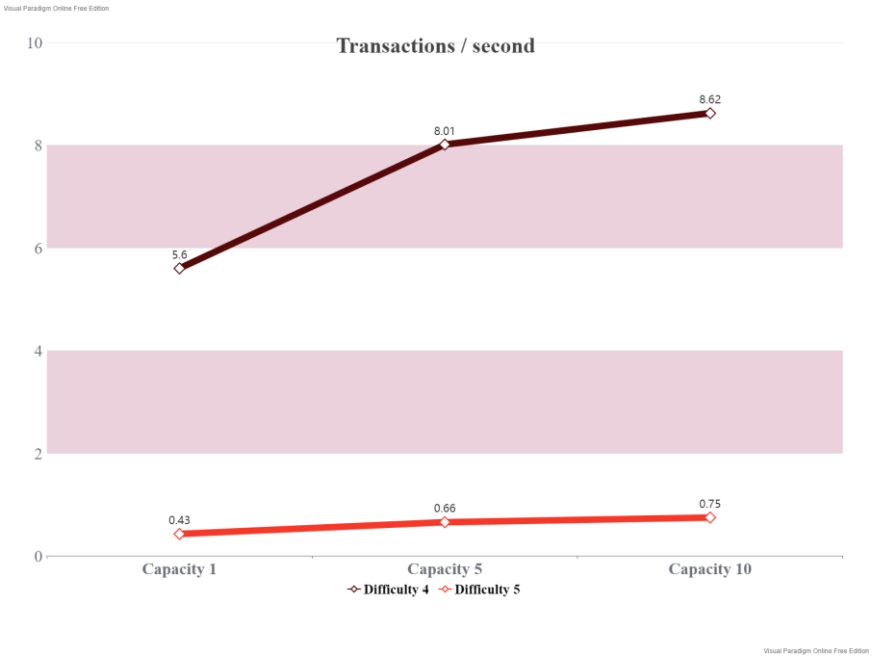
Όταν ένας κόμβος δέχεται ένα νέο block, ελέγχει αν είναι το επόμενο(καινούριο) block του blockchain του. Αν είναι, το προσθέτει στο τέλος της αλυσίδας του. Αλλιώς, αν δεν έχει κάποιο reference γι' αυτό, δηλαδή αν δεν γνωρίζει τον πρόγονο του, το κρατάει για να το ελέγξει αργότερα στην περίπτωση που ήρθε πρώτο από κάποιο προηγούμενό του. Έτσι, όποτε αυξάνεται το μήκος του blockchain του, ο κάθε κόμβος ελέγχει τα blocks που έχει κρατήσει.

Παρακάτω, φαίνεται σχηματικά η υλοποίηση της εφαρμογής στα επιμέρους κομμάτια της:

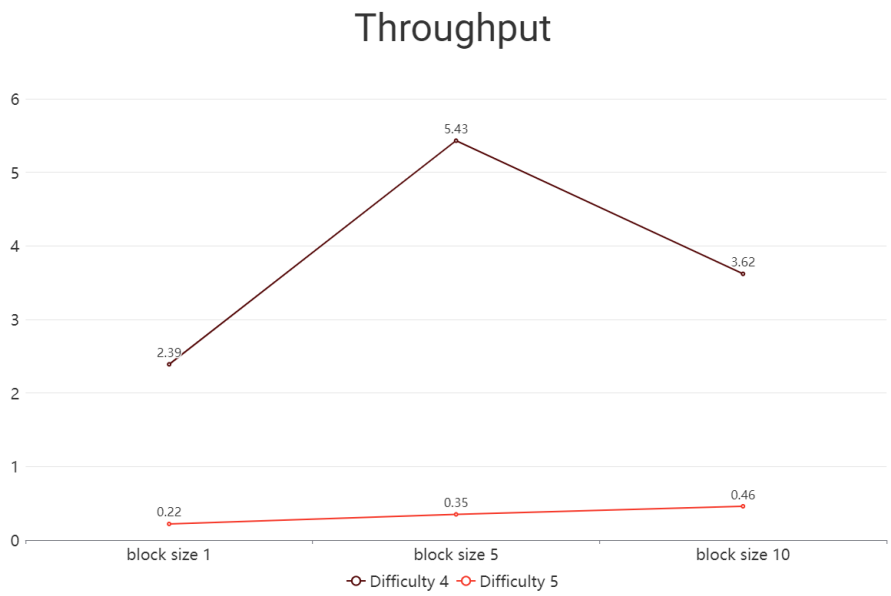


Αποτελέσματα

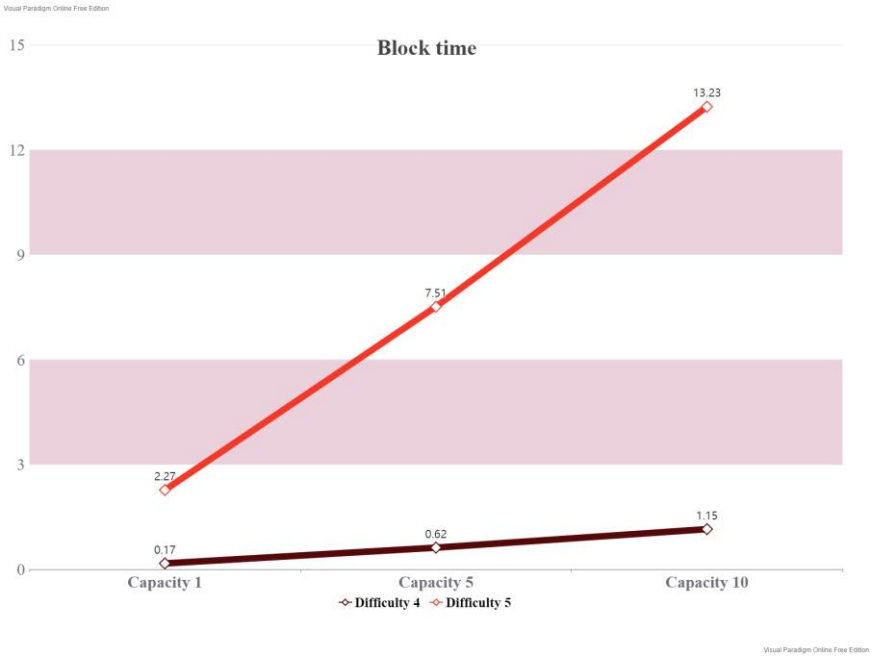
Παρακάτω, φαίνεται το throughput της εφαρμογής για 5 nodes, με block size 1, 5, 10 και difficulty 4 και 5. Φαίνεται πως, όσο αυξάνεται το block size, αυξάνεται και το throughput, ενώ η αύξηση στο difficulty το ρίχνει σημαντικά (περίπου κατά 16 φορές). Τρέχοντας την εφαρμογή σε δικούς μας πόρους έχουμε:



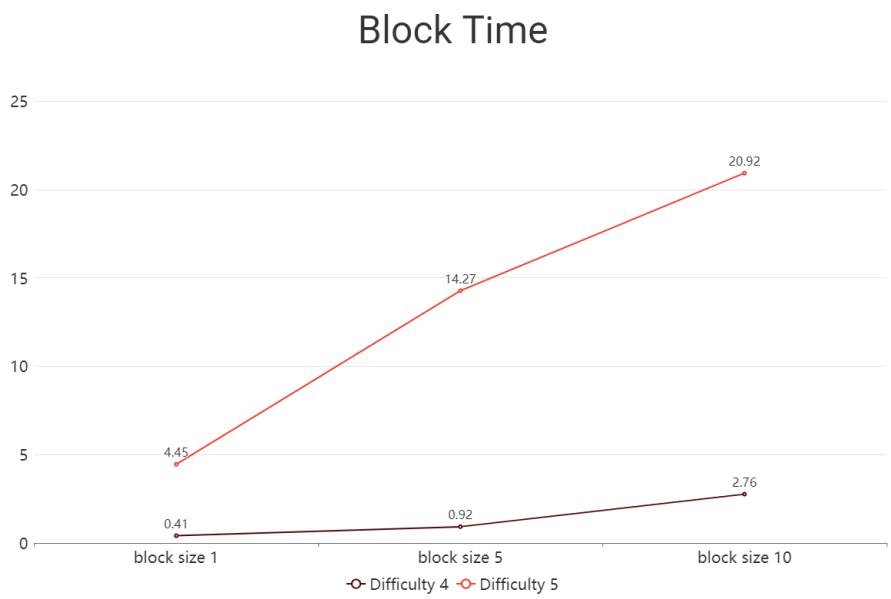
Ενώ τρέχοντάς τη στους πόρους του Okeanos, έχουμε:



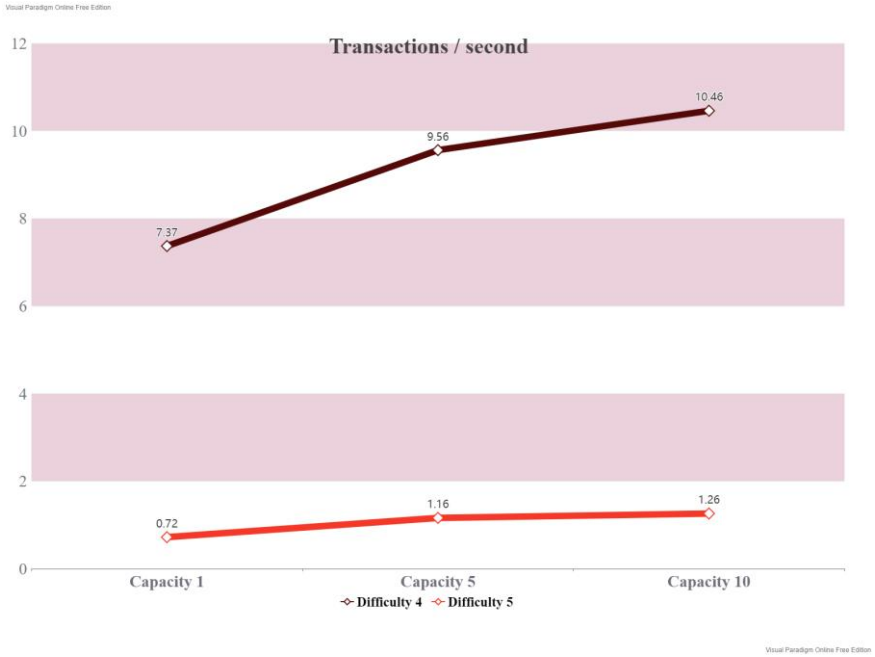
Για το block time, παρατηρούμε ότι όσο αυξάνεται το block size και το difficulty, αυτό αυξάνεται. Παρακάτω, φαίνεται το διάγραμμα του block time για 5 nodes σε δικούς μας πόρους:



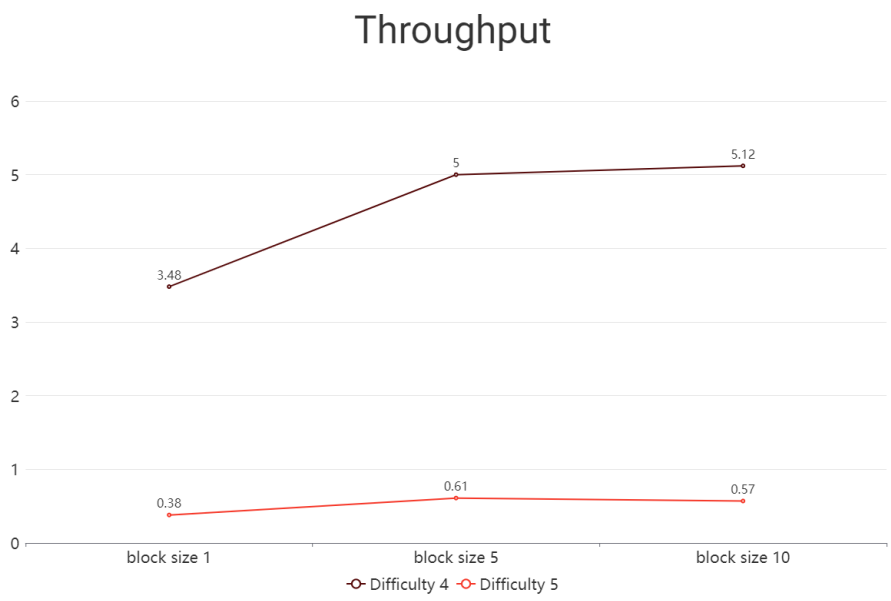
Και στους πόρους του Okeanos:



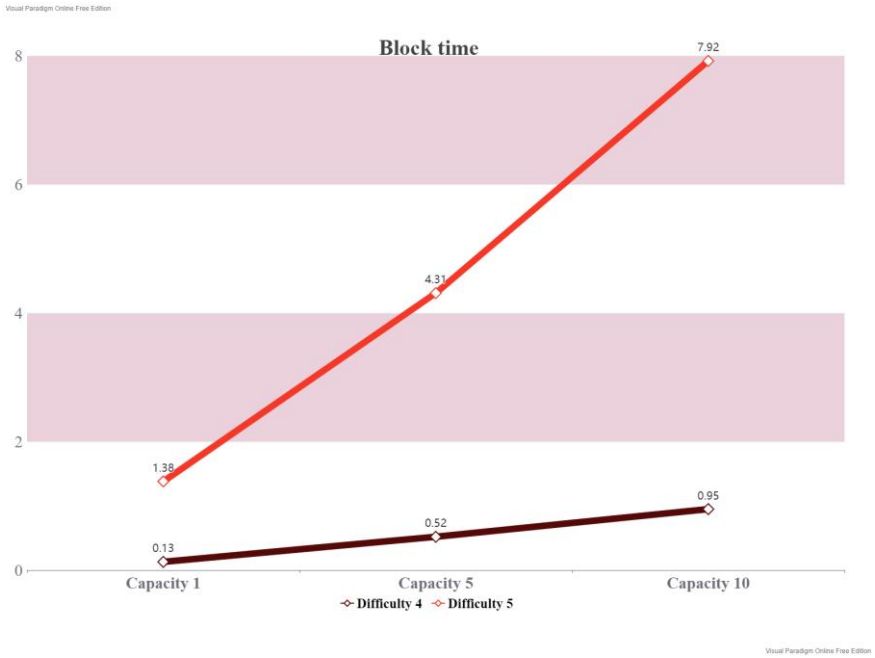
Για να δούμε και το scalability, παρουσιάζουμε διάγραμμα του throughput και για 10 nodes, με block size 1, 5, 10 και difficulty 4 και 5. Για τους 10 κόμβους, παρατηρούμε αντίστοιχα αποτελέσματα με αυτά που πήραμε για τους 5. Τρέχοντας την εφαρμογή σε δικούς μας πόρους έχουμε:



Ενώ τρέχοντάς τη στους πόρους του Okeanos, έχουμε:



Έχουμε το διάγραμμα του block time για 10 nodes σε δικούς μας πόρους:



Και στους πόρους του Okeanos: