Προγραμματιστική Άσκηση 2

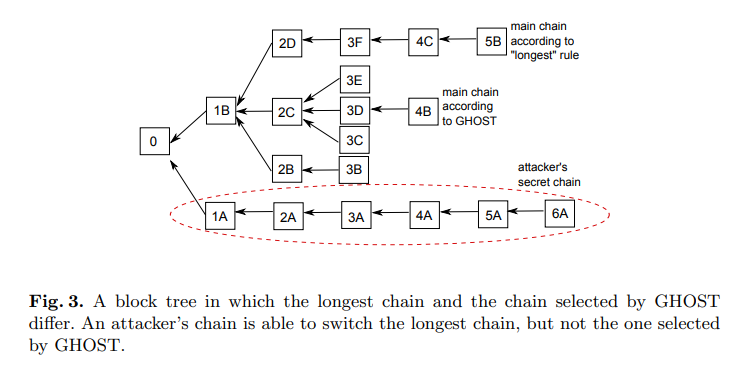
# μέρος 1ο:

Όταν έχεις ενα σύστημα blockchain πρέπει να αποφασίσεις στον μηχανισμό επιλογής επόμενου block και επικρατέστερης αλυσίδας. Μερικές φορές το ένα προκείπτει από το άλλο, πχ σε περίπτωση Proof of Work συστημάτων η επιλογή της μακρύτερης αλυσίδας είναι η πιο συχνή λόγω του ότι για τα μπλοκ της αλυσίδας ξοδεύτηκαν αρκετοί πόροι και δεν θέλουμε πρώτον σπατάλη και δεύτερον είναι πάρα πολύ δύσκολο κάποιος να την αλλοιώσει με κακόβουλες βλέψεις.

Αυτό όμως δεν λειτουργεί ακριβώς όπως θα θέλαμε με άλλους μηχανισμούς όπως Proof of Stake, όπου κάποιος με το μεγαλύτερο stake θα μπορεί να παράξει μόνος του την μακρύτερη αλυσίδα και να την αλλάξει με σκοπό το δικό του συμφέρον. Άλλοι μηχανισμοί επιλογής της κύριας αλυσίδας είναι η επιλογή της **βαρύτερης αλυσίδας** και το προτόκολο **GHOST** ( Greedy Heaviest-Observed Sub Tree ) [1] [<https://eprint.iacr.org/2013/881.pdf>].

Στην επιλογή της **βαρύτερης αλυσίδας**, διαλέγεται να συνεχιστεί η αλυσίδα που έχει αθροιστικά το μεγαλύτερο βάρος από τις άλλες ως προς μία μετρική (πχ effort σε PoW και stake σε PoS). Το καλό αυτής της επιλογής είναι ότι είναι πιο ευέλικτη από την πιο απλή επιλογή της μακρύτερης αλυσίδας καθώς ορίζεται ανάλογα με τις ανάγκες η μετρική του βάρους αλλά επίσης δεν είναι τόσο εύκολο να λάβει κάποιο αυτόβουλα μέρος στην αλυσίδα καθώς θα χρειαζόταν σημαντικό μέρος από την έκαστη μετρική (πχ υπολογιστική ισχύ ή οικονομία). Σε αντιστοιχία όμως ανάλογα με την μετρική υπάρχει ακόμα κίνδυνος για μερική κεντρικοποίηση λόγω της συσσόρευσης κάποιων πόρων από λίγους χρήστες. Επίσης δεν γίνεται να διαβαθμιστεί τόσο καλά, χτυπάει ένα ταβάνι απόδοσης όπως και το longest chain.

Από την άλλη μεριά, το **GHOST** επιλέγει το επόμενο block από το υπο-δέντρο των αλυσιδών με το πιο βαρύ υπόδεντρο, αξιοποιώντας έτσι την ύπαρξη και γειτονικών αλυσίδων στο υπο-δέντρο για την επιλογή.



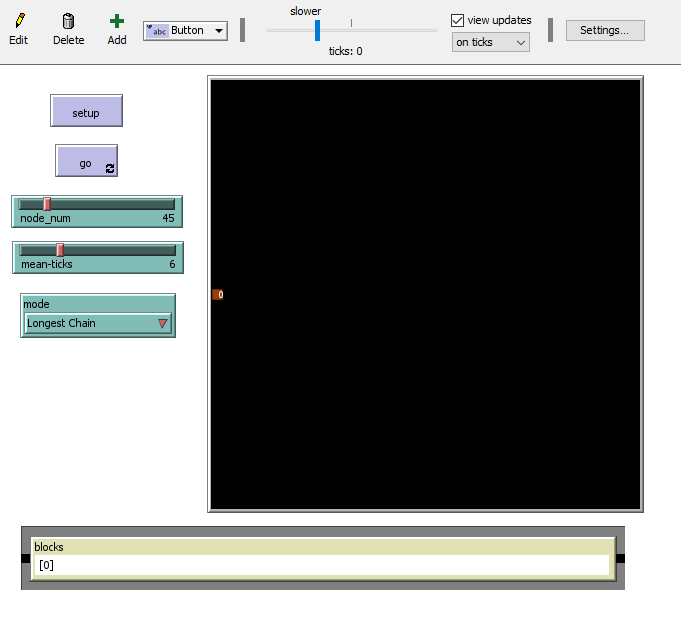
Πχ στο παράδειγμα πιο πάνω σύμφωνα με το longest chain protocol η επιλογή θα γινόταν ή στον attacker ή στο longest chain αλλά με το GHOST θα γίνει στην μεσαία αλυσίδα αφού φαίνεται ότι οι περισσότεροι προτίμησαν εκείνο το μονοπάτι για να προσθέσουν το block τους όταν ήρθε η ώρα της επιλογής. Με αυτόν τον τρόπο επιτυγχάνεται πιο γενική αξιολόγηση της αλυσίδας και προσμετριέται το βάρος κάθε block παιδιού του κάθε genesis block. Επίσης είναι αρκετά πιο scalable από το longest chain καθώς σύμφωνα με το [1] επιτυγχάνεται αύξηση του throughput και των συνολικών transactions μέσα σε ένα block χωρίς να υπονομευτεί η ασφάλεια. Αρνητικό αυτής της υλοποίησης είναι ότι είναι αρκετά πιο περίπλοκο στην υλοποίηση και τον υπολογισμό του βάρους κάθε υπο-δέντρου.

1. - <https://eprint.iacr.org/2013/881.pdf>

## Μέρος 2ο:

Για την υλοποίηση του δεύτερου μέρους κάναμε μερικές καίριες υποθέσεις για την διευκόλυνση μας όπως προτείνεται και από την εκφώνηση.

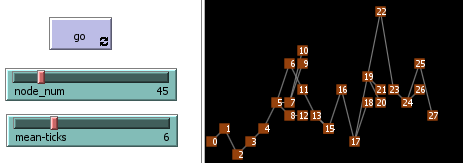
Αρχικά να τονιστεί ότι ο κώδικας στην netlogo είναι γεμάτος επεξηγητικά σχόλια για τυχόν έξτρα πληροφορίες που μπορεί να χρειαστούν. Ακολουθεί screenshot από το περιβάλλον υλοποίησης μετά από ένα setup:



Οι υποθέσεις που κάναμε για την υλοποίηση είναι

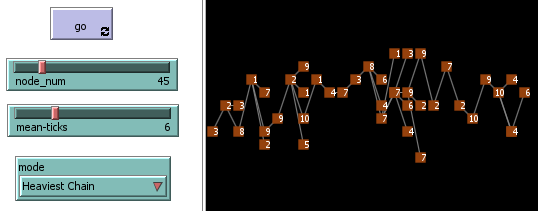
* Στο μοντέλο μας κάθε **κόμβος** και κάθε **block** είναι ένας ξεχωριστός agent που δημιουργείται μέσω των breeds για καλύτερη διαχείριση μέσα στον κώδικα.
* Το **blockchain** στην ολότητα του είναι μία **global λίστα** που περιέχει όλα τα id (headers) των blocks που δημιουργήθηκαν, προσομοιώνοντας την άμεση γνώση του κάθε κόμβου του blockchain και τον υποτιθέμενων συναλλαγων που έγιναν. Επίσης λόγω του ότι είναι global, σε κάθε βήμα κάθε κόμβος ξέρει όλη την αλυσίδα, μία ακόμα παραδοχή που κάναμε.
* Κάθε block αποθηκεύει το id του και το id του block στο οποίο δείχνει. Έτσι προσομοιώνεται πιστά η φύση της αλυσίδας και των διακλαδώσεων όπου κάθε σε block μπορεί να δείχνει παραπάνω από ένα blocks.
* Για χάρην υλοποίησης και ορθής λειτουργίας, στο setup δημιουργείται ένα αρχικό block από τον observer και έπειτα τρέχει το μοντέλο όπου κάθε κόμβος προσπαθεί να φτιάξει το δικό του block.
* Όπως δίνεται να εννοειθεί από την εκφώνηση, το σύστημα συναίνεσης στην προκειμένη είναι ένα **Proof of Work** σύστημα καθώς όπως καταλάβαμε ο χρόνος παραγωγής ενός block εξαρτάται από μια τιμή της τυχαίας μεταβλητής (**nonce**) που ακολουθεί την κατανομή Poisson. Άρα με μία αναμενόμενη μέση τιμή ως χρόνο παραγωγής (**mean**-**ticks**) κάθε μπλοκ προσπαθεί να φτιάξει αυτόνομα από τους υπόλοιπους το block του ανάλογα με το δικό του *nonce.*
* Υλοποίησαμε την διαδικασία με το σκεπτικό ότι σε κάθε τικ περνάει και μία χρονική μονάδα. Όταν δημιουργείται ένα block, όλοι οι κόμβοι μαθαίνουν για αυτό και ξεκινούν από την αρχή και με νέο nonce να φτιαχνουν το νέο block. To nonce *συγκρίνεται* από κάθε κόμβο με τα ticks που πέρασαν από την δημιουργία του τελευταίου block και *αν είναι ίσα*, θεωρούμε ότι ο κόμβος βρήκε κατάλληλο nonce και προτεινει το επόμενο block στην αλυσίδα.
* Για λόγους παρατήρησης, οι αλυσίδες αναπαριστόνται ως σύνδεσμοι μεταξύ των blocks. Κάθε block έχει ως συντεταγμένη x το βάθος του στην αλυσίδα (μόνο αυξάνεται) και σαν y ένα τυχαίο y (αυτό για λόγους ευκολίας της υλοποίησης, ήταν πιο δύσκολο να θέσουμε σαν y το y του block που συνδέεται το νέο block και να μην φτάσει στα όρια του κόσμου όταν αυτα αυξάνονται). Έτσι για να δείξουμε σε πιο block συνδέεται το κάθε νέο block του είπαμε να δημιουργήσει ένα link με το block που έχει σαν id το parent\_id του νέου block.
* Όσον αφορά την μέθοδο επιλογής του block για την σύνδεση με το επόμενο, υλοποιήθαν 2 μέθοδοι: Της **μακρύτερης αλυσίδας** και της **Βαρύτερης αλυσίδας**.
  + Στην περίπτωση της **μακρύτερης αλυσίδας**, κάθε κόμβος αναζητά στην συνολική αλυσίδα τους κόμβους με το μεγαλύτερο βάθος και επιλέγει τυχαία έναν από αυτούς ως το ιδανικό. Η επιλογή γίνεται τυχαία για να προσομοιώσει κάπως την μερική άγνοια της καλύτερης επιλογής της αλυσίδας από όλους τους κόμβους την ίδια χρονική στιγμή.

Σαν αποτέλεσμα έχει μερικές φορές να δημιουργόυνται παράλληλες αλυσίδες που πιθανόν να τρέξουν παράλληλα για μερικές γεννήσεις blocks αλλά όταν σε μία από αυτές προστεθεί ένα μόνο block σε ένα βήμα, τότε η κύρια αλυσίδα θα θεωρηθεί αυτή και θα συνεχιστεί από εκεί.



Πχ στην συγκεκριμένη περίπτωση στο block 5 συνδέθηκαν 3 blocks και σε αυτά συνδέθηκαν συνολικά 4 νέα. Όμως μόνο ένα συνδέθηκε στο 13 και έπειτα η αλυσίδα συνεχίστηκε από αυτό. Άρα όντως λειτουργεί με τον κανόνα της μακρύτερης αλυσίδας.

* Για την Βαρύτερη αλυσίδα έπρεπε κάπως να προσομοιώσουμε το βάρος κάθε block. Αυτό γίνεται μέσω μιας μεταβλητής κάθε block που παίρνει τυχάια τιμές μεταξύ 0 και 1. Για λόγους ευανάγνωσης, σαν label όταν τίθεται σαν mode λειτουργίας το Heaviest Chain, ανατίθεται το πρώτο δεκαδικό ψηφίο του βάρους.



Αυτό που παρατηρήθηκε κατά την υλοποίηση είναι ότι δεν υπήρχαν παράλληλες αλυσίδες, όπου υπήρχαν διακλαδώσεις ήταν ένα μόνο το block με το μεγαλύτερο βάρος οπότε ήταν η πρώτη επιλογή κάθε κόμβου. Αφου δεν συνεχίστηκαν παράλληλες αλυσίδες το θέμα της βαρύτερης αλυσίδας μετατρέπεται σε βαρύτερο block, που στην πραγματικότητα επειδή κάθε κόμβος δεν μπορεί να ξέρει όλη την αλυσίδα δεν γίνεται αλλά στα πλαίσια της υλοποίησης μας βγάζει νόημα.

Για να καταφέρει η μακρύτερη αλυσίδα να μην έχει πολλές διακλαδώσεις πρέπει να βρεθεί μία ισορροπία στο πλήθος τον κόμβων που προσπαθούν να φτιάξουν blocks και της μέσης τιμής του nonce που θέτουμε. Αν ένα από τα δύο ξεφεύγει σαν τιμή τότε ή πάρα πολλά παράλληλα blocks θα δημιουργούνται (χαμηλό nonce και μεγάλο πλήθος nodes) ή πολλή σπάνια δημιουργία blocks (στην αντίστροφη περίπτωση) άρα και μη παρατήρηση διακλαδώσεων.

Αυτό σαν παρατήρηση **επιβεβαιώνεται** από την πραγματικότητα που το **Bitcoin** προσπαθεί συνεχώς να προσαρμόζει την δυσκολία των *hash puzzles* για τα *nonce* (αντίστοιχα το mean-ticks στο μοντέλο μας) ώστε να μένει σε ένα *σταθερό χρόνο* η παραγωγή κάθε νέου block όσο αυξάνονται οι κόμβοι ή η υπολογιστική ισχύς αυτών (αυξάνουμε το πλήθως των nodes αντίστοιχα)