Building an agent for the board game Pandemic

Κοσμάς Πίνήτας – 2016030010
Κωνσταντίνος Πάντος – 2016030015
Μαρία Τσολάκη – 2017030164

Institution: Πολυτεχνείο Κρήτης

Departement : Η.Μ.Μ.Υ

Course: Πολυπρακτορικά Συστήματα

Professor: Γεώργιος Χαλκιαδάκης

1 Εισαγωγή

1.1 Σκοπός Εργασίας

Σκοπός της εργασίας αυτής ήταν η δημιουργία συνεργατικών και λογικών πρακτόρων οι οποίοι θα έχουν ως κοινό στόχο να νικήσουν στο παιχνίδι pandemic. Έχοντας ήδη έτοιμο το functional framework του παιχνιδιού κατασκευάσαμε πράκτορες οι οποίοι προσπαθούν αρχικά να εξερευνήσουν (exploration) το περιβάλλον τους και στη συνέχεια να το εκμεταλλευτούν (exploitation) με σκοπό να πετύχουν το στόχο τους δηλαδή να νικήσουν.

1.2 Pandemic

Το Pandemic είναι ένα επιτραπέζιο παιχνίδι στο οποίο οι παίκτες πρέπει να συνεργαστούν με σκοπό να σώσουν τον κόσμο από 4 ασθενείς οι οποίες εξαπλώνονται με γρήγορο ρυθμό. Στην δική μας περίπτωση είχαμε μια παραλλαγή του αρχικού παιχνιδιού κατά την οποία οι παίκτες μας μπορούσαν να έχουν μόνο τους ρόλους των Medic, Scientist, Quarantine Specialist και Operations Expert ενώ δεν είχαμε καθόλου Event Cards. Οι παραπάνω αλλαγές απλούστευσαν λίγο το παιχνίδι καθώς τόσο η προσθήκη και άλλων ρόλων όσο και η προσθήκη των event cards θα αύξαναν σημαντικά τη πολυπλοκότητα του παιχνιδιού. Αξίζει να σημειωθεί ότι το Pandemic είναι ένα συνεργατικό παιχνίδι το οποίο δεν μας δίνει πλήρη πληροφορία.

2 Agent

2.1 Markov Decision Process

Η μοντελοποίηση ενός περιβάλλοντος ως μια διαδικασία Markov είναι ένα αρκετά χρήσιμο μαθηματικό εργαλείο το οποίο μπορεί να μας βοηθήσει να μοντελοποιήσουμε με σχετική ακρίβεια καταστάσεις στις οποίες υπάρχει μερική τυχαιότητα έτσι ώστε να πάρουμε τις κατάλληλες αποφάσεις. Σε κάθε βήμα η διαδικασία βρίσκεται σε μια κατάσταση s από την οποία μεταβαίνει τυχαία σε μια άλλη κατάσταση s' η οποία όμως εξαρτάται από τη δράση a αυτού που λαμβάνει τις αποφάσεις. Η μετάβαση από μια κατάσταση s σε μια κατάσταση s' επιστρέφει ένα βραβείο $R(s, s')$ Αξίζει μα σημειωθεί ότι οι $s, s' \in S$ με S να είναι ο χώρος καταστάσεων. Ακόμα $a \in A$ με A να είναι ο χώρος δράσεων ενώ μπορούμε να αναπαραστήσουμε κάθε ομάδα κατάστασης, δράσης, μετάβασης και βραβείου ως εξής: (s, a, r, s')

Στο Pandemic κάθε agent εκτελεί 4 κινήσεις συνεχόμενα. Ξεκινάει από μια πόλη (κατάσταση s_i) εκτελεί κάποια δράση a_t και καταλήγει σε μια πόλη (κατάσταση s_j) ενώ λαμβάνει κάποιο βραβείο r_x. Μπορούμε λοιπόν να αντιστοιχίσουμε τις πόλεις σε 4 διαφορετικές καταστάσεις και να διαχωρίσουμε τις δράσεις σε 4 ήδη ως εξής:

- **Πόλεις**

  $\mathbf{s_0}$: Δεν έχουν ούτε κύβους ούτε ερευνητικά κέντρα

  $\mathbf{s_1}$: Έχουν κύβους αλλά δεν έχουν ερευνητικά κέντρα

  $\mathbf{s_2}$: Δεν έχουν κύβους αλλά έχουν ερευνητικά κέντρα

  $\mathbf{s_3}$: Έχουν και κύβους και ερευνητικά κέντρα

- **Δράσεις**

  $\mathbf{a_0}$: Κάθε είδους μεταφορά (DT. DF. CF, SF, OET)

  $\mathbf{a_1}$: Χτίσιμο ερευνητικού κέντρου (BRS)

$\mathbf{a_2}$: Θεραπεία κάποιας ασθένειας (TD)

$\mathbf{a_3}$: Ανακάλυψη αντιδότου (CD1)

- **Βραβεία**
  * **Medic**

    $WtrS0 = -300$: $r = WtrS0 - handPenalty$
    $WtrS1 = 100$: $r = WtrS1 - handPenalty$
    $WtrS2 = 20$: $r = WtrS2 - handPenalty$
    $WtrS3 = 150$: $r = WtrS3 - handPenalty$
    $Wbrs = 180$: $r = Wbrs - handPenalty$
    $Wtd = 200$: $r = Wtd$
    $Wcd = 300$: $r = Wcd$

  * **Scientist**

    $WtrS0 = -300$: $r = WtrS0 - handPenalty$
    $WtrS1 = 20$: $r = WtrS1 - handPenalty$
    $WtrS2 = 100$: $r = WtrS2 - handPenalty$
    $WtrS3 = 150$: $r = WtrS3 - handPenalty$
    $Wbrs = 180$: $r = Wbrs - handPenalty$
    $Wtd = 200$: $r = Wtd$
    $Wcd = 300$: $r = Wcd$

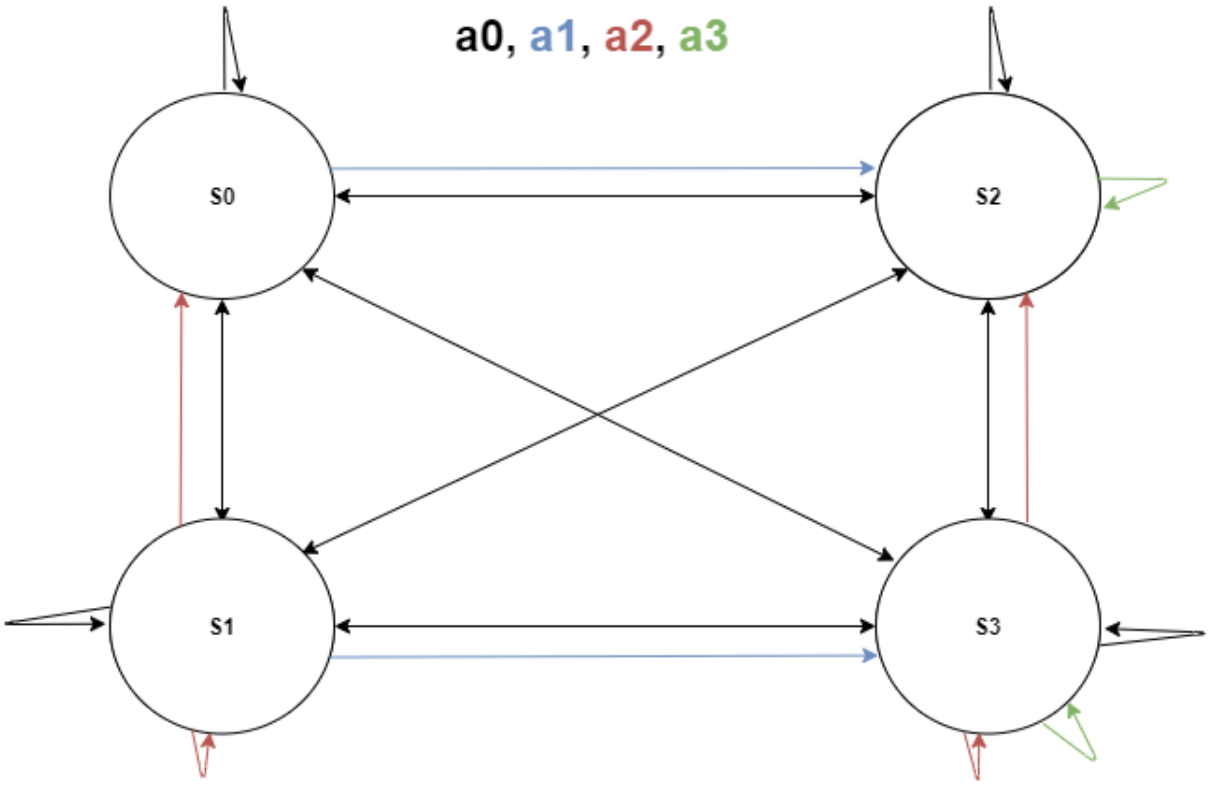
  * **Operations Expert**

    $WtrS0 = 50$: $r = WtrS0 - handPenalty$
    $WtrS1 = 100$: $r = WtrS1 - handPenalty$
    $WtrS2 = 50$: $r = WtrS2 - handPenalty$
    $WtrS3 = 100$: $r = WtrS3 - handPenalty$
    $Wbrs = 180$: $r = Wbrs - handPenalty$
    $Wtd = 200$: $r = Wtd$
    $Wcd = 300$: $r = Wcd$

  * **Quarantine Specialist**

    $WtrS0 = -100$: $r = WtrS0 - handPenalty$
    $WtrS1 = 100$: $r = WtrS1 - handPenalty$
    $WtrS2 = 20$: $r = WtrS2 - handPenalty$
    $WtrS3 = 150$: $r = WtrS3 - handPenalty$
    $Wbrs = 180$: $r = Wbrs - handPenalty$
    $Wtd = 200$: $r = Wtd$
    $Wcd = 300$: $r = Wcd$

Αξίζει να σημειωθεί ότι $\mathbf{handPenalty} = \frac{\mathbf{7-handSize}}{\mathbf{7}}$ καθώς θέλουμε ο agent να διαλέγει actions τα οποία έχουν μεγάλο reward χωρίς να ξοδεύει πολλές κάρτες (Ο αριθμός 7 είναι ο αριθμός των καρτών που επιτρέπεται να έχει ο παίκτης στο χέρι του). Τέλος **τα βαρη επιλέχθηκαν μετά από 10 δοκιμές για κάθε ρόλο** και έχουν ως στόχο να αναγκάσουν τους agents να χρησιμοποιούν τις ειδικές τους ικανότητες. Για παράδειγμα ο Medic έχει μεγαλύτερο συμφέρον να πηγαίνει σε πόλεις που έχουν cubes. **Παρακάτω φαίνεται η μοντελοποίηση του παιχνιδιού ως MDP με όλες τις ομάδες** (s, a, r, s') **να έχουν ίση πιθανότητα εμφάνισης.**



2.2 Q Learning

Για να μπορέσουμε να αποφανθούμε σχετικά με το αν η παραπάνω μοντελοποίηση θα βοηθήσει τους πράκτορες να εξάγουν μια ορθολογική πολιτική στο συγκεκριμένο περιβάλλον εφαρμόσαμε τον κλασικό αλγόριθμο Q Learning. Η μέθοδος αυτή είναι αρκετά διαδεδομένη στο πεδίο της ενισχυτικής μάθησης (Reinforcement Learning). **Ο αλγόριθμος αυτός μας βοηθά να βρούμε την ποιότητα κάθε δράσης, ενώ προσπαθεί να μεγιστοποιήσει την αναμενόμενη επιβράβευση για κάθε δράση σε κάθε κατάσταση.** Η ποιότητα κάθε δράσης μπορεί να υπολογιστεί ως εξής:

$$Q(s,a) = Q(s,a) + \mu\big(r + \gamma max_a\{Q(s',a)\} - Q(s,a)\big)$$

- μ : Ο βαθμός εκμάθησης (learning rate) ελέγχει σε τι ποσοστό συμβάλει η νέο πληροφορία στην ενημέρωση της τιμής. Τιμή 0 σημαίνει ότι δεν έχουμε μάθηση νέας πληροφορίας ενώ τιμή 1 σημαίνει ότι κρατάμε μόνο πρόσφατη πληροφορία. (Στη περίπτωση μας $\mu = 0.9$)
- $Q(s,a)$: Ο πίνακας Q αρχικοποιήθηκε ως μηδενοπίνακας
- γ : Ο παράγοντας έκπτωσης (discount factor) ελέγχει σε το ποσοστό ο αλγόριθμος λαμβάνει υπόψιν του τη μελλοντική επιβράβευση. Τιμή 0 σημαίνει ότι η πολιτική μας δεν είναι διορατική καθώς λαμβάνουμε υπόψιν μας μόνο τωρινά rewards. Γενικά προσπαθούμε να έχουμε τιμές κοντά στο 1 καθώς για $\gamma \geq 1$ ενδέχεται να υπάρξουν προβλήματα σύγκλισης. (Στη περίπτωση μας $\gamma = 0.9$)

Η πολιτική που θα ακολουθήσει ο πράκτορας όταν βρεθεί σε κατάσταση s μπορεί να βρεθεί ως εξής:

$$\pi(s) = argmax_a\{Q(s,a)\}$$

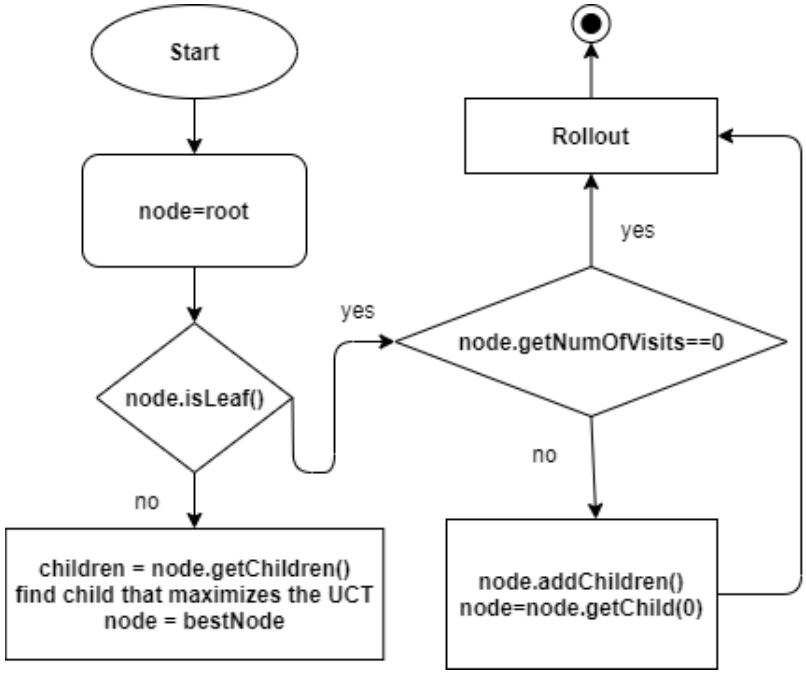
Όπως αναφέραμε και στην ενότητα για τα MDPs οι παράμετροι W προέκυψαν μετά apo 10 trials για κάθε agent (ρύθμιση για 1 παίκτη στο Server). Σε κάθε trial δημιουργούσαμε όλες τις πιθανές αρχικές καταστάσεις και ξεκινώντας από αυτές (100 παιχνίδια για κάθε κατάσταση) κάναμε sample τυχαίες επόμενες καταστάσεις (100 τυχαίες επιτρεπτές κινήσεις) με σκοπό να εξάγουμε τη ποιότητα των δράσεων του πράκτορα. Για παράδειγμα αν η τυχαία κίνηση ήταν a_0 και μας οδηγούσε σε κάποια κατάσταση s_1 τότε επιλέγαμε μια υπαρκτή τυχαία κίνησης σχετική με transportation σε μια τυχαία πόλη η οποία δεν είχε ερευνητικό κέντρο αλλά είχε κύβους. Συνολικά έχπυμε 4 καταστάσεις με 4 δράσεις ανα κατάσταση οπότε $\mathbf{4^4}$ **πολιτικές**. Για εμάς ένας πράκτορας με ορθολογική πολιτική έπρεπε να κανει TD σε πόλεις με κύβους και να φεύγει απο πόλεις χωρις κύβους (δεδομένου ότι ο αροθμός των RS και των CD είναι αρκετά μικρότερος του 100). **Το πρόβλημα με αυτή τη μέθοδο είναι ότι λαμβάνουμε υπόψιν μας μόνο τη κίνηση χωρίς να μας ενδιαφέρει η κατάσταση του board ενώ στη περίπτωση του $\mathbf{a_0}$ δεν μπορούμε να ξέρουμε ακριβώς ποιά κίνηση θέλουμε να εκτελεστεί, επίσης η μετακίνηση σε πολεις ίδιας κατάστασης δεν είναι απαραίτητο ότι έχει ίδια αξία.**

2.3 Monte Carlo Tree Search

Ο αλγόριθμος αυτός είναι ένας heuristic-based αλγόριθμος αναζήτησης ο οποίος μας βοηθά να λάβουμε αποφάσεις και χρησιμοποιείται ευρέως στην επίλυση επιτραπέζιων παιχνιδιών αλλά και παιχνιδιών που δεν μας δίνουν πλήρη πληροφόρηση.

2.3.1 Αλγόριθμος

Ο αλγοριθμος όπως των υλοποιήσαμε φαίνεται στο παρακάτω σχήμα ενώ αξίζει να σημειωθεί ότι κατά τη διαδικασία του Rollout απλά κάνουμε sample τυχαίες επιτρεπτές κινήσεις μέχρι να φτάσουμε είτε σε κάποια τερματική κατάσταση είτε να ξεπεράσουμε ένα ανώτατο όριο κινήσεων. (Στη περίπτωση μας 100) και μετά ενημερώνουμε τους κόμβους του δέντρου μας σύμφωνα με τη τιμή που μας επέστρεψε η τελευταία τυχαία κίνηση.



2.3.2 UCT

Το UCT είναι μια ντετερμινιστική μετρική που μας βοηθά να καθορίσουμε την ποιότητα ενός κόμβου και μπορεί να υπολογιστεί ως εξής:

$$UCT(node) = \frac{node.value}{node.numOfVisits} + c \cdot \sqrt{\frac{\log parent.numOfVisits}{node.numOfVisits}}$$

Ο πρώτος λόγος δίχνει την αναμενόμενη αξία που θα κερδίσουμε αν επιλέξουμε αυτον τον κόμβο και αφορά το κομμάτι του exploitation ενώ ο δεύτερος λόγος μας δίχνει σε τί ποσοστό έχουμε περάσει απο το κόμβο node συγκριτικά με τον πατέρα τπυ και αφορά το κομμάτι του exploration. Η σταθερά c ειναι ένας όρος που μας βοηθά να ορίσουμε τη σημασία που έχει για εμάς το exploration έναντι του exploitation. Τιμή 0 σημαίνει ότι δεν κάνουμε καθόλου expkoration. Στη περίπτωση μας έχουμε $c = \sqrt{2}$.

Κάνοντας χρήση του UCT (μετά τη διαδικασία της "εκπαίδευσης") μπορούμε να ορίσουμε τη πολιτική που ακολουθεί ο agent μας ως εξής:

$$\pi(node) = argmax_{child \in children}\{UCT(child)\}$$

Δηλαδή αν κάθε κόμβος node περιέχει ένα action και το state στο οποίο μας οδηγεί αυτό το action τότε επιλέγω το επόμενο action (δηλαδή τον κόμβο παιδί) ο οποίος έχει το μέγιστο UCT

Υπάρχουν αρκετοί τρόποι επιλογής κατάλληλου κόμβου με τους πιο διαδεδομένους να είναι:

- επιλογή του κόμβου με το μεγαλύτερο αριθμό επισκέψεων
- επιλογή με βάση το UCT
- επιλογή με βάση το exploitation (UCT για $c = 0$)

Παρόλο που η γενική πρακτική είναι η επιλογή με βάση το μεγαλύτερο αριθμό επισκέψεων, εμείς παρατηρήσαμε ότι η επιλογή αυτή μείωσε τον μέσο αριθμό γύρων (για 20 παιχνίδια) που επιβίωναν οι πράκτορες μας απο 15 σε 12.

2.3.3 Evaluation Function

Κανονικά κατά το rollout θα έπρεπε να κάνουμε sample τυχαίες κινήσεις μέχρι το τέλος του παιχνιδιού. Όμως κάτι τέτοιο απαιτεί αρκετο χρόνο και συνεπώς μια τέτοια λύση δεν είναι εφικτή. Για το λόγο αυτό υλοποιήσαμε μια συνάρτηση η οποία δέχεται σαν όρισμα το board και μας επιστρέφει μια τιμή η οποία μας δίχνει το πόσο καλή η κακή είναι η πορεία των κινήσεων μας. Για την ακρίβεια υλοποιήσαμε μια τροποποιημένη έκδοση του evaluation function απο το paper με τίτλο PAIndemic: A Planning Agent for Pandemic. Συγκεκριμένα:

$$h_{state} = h_{state_{max}} - (0.5(regionRisk + h_{dcure} + h_{disc}) + 0.6(h_{inf} + h_{dist}) + h_{cards} + 24h_{cures})$$

Τα βάρη είναι τα ίδια με αυτά που προτείνουν οι συγγραφείς του paper ενώ για να εξηγήσουμε τον τρόπο με τον οποίο αξιλογούμε τις καταστάσεις αξίζει να αναφερθούμε σε κάθε παράμετρο της h_{state} ξεχωριστά.

- $\mathbf{regionRisk} = \mathbf{remainingCubes(CityColour)}$
  Αναγκάζει τον agent να κινηθεί σε πόλεις που ανοίκουν σε περιοχές με πολλούς κύβους.

- $\mathbf{h_{dcure}} \sum_{\mathbf{p=0}}^{\mathbf{nOfPlayers}} \mathbf{min_{c \in RSLocations}\{distance(p, c)\}}$
  Αναγκάζει τον agent να μένει κοντά σε περιοχές που έχουν RS.

- $\mathbf{h_{cards}} = \sum_{\mathbf{k}}^{\mathbf{nColours}} \mathbf{active_i min_{p \in Players}\{R - cards(p, k)\}}$
  Αναγκάζει τον agent να ομαδοποιεί κάρτες ίδιου χρώματος για τις ενεργές ασθένειες. Όπου R είναι ο αριθμός των καρτών που πρέπει να έχει ένας παίκτης για να κάνει cure (γενικά 4 αλλα για τον Scientist 3).

- $\mathbf{h_{disc}} = \sum_{\mathbf{k}}^{\mathbf{nColours}} \mathbf{active_k discardedPile(k)}$
  Αναγκάζει τον agent να μη χαλάει κάρτες ενός χρώματος που κοντεύει να εξαφανιστεί απο το deck.

- **cubesOnBoard**
  Αναγκάζει τον agent να ελέγξει τη μετάδωση των ασθενειών.

- $\mathbf{h_{disc}} = \sum_{\mathbf{k}}^{\mathbf{nColours}} \mathbf{active_k}$
  Αναγκάζει τον agent να βρεί cures.

- $\mathbf{h_{dist}} = \sum_{\mathbf{c_1}}^{\mathbf{Cities}} \sum_{\mathbf{c_2}}^{\mathbf{Cities}} \mathbf{distance(c_1, c_2)} \frac{\mathbf{turns_{remaining}}}{\mathbf{turns_{max}}}$
  Αυτή η παράμετρος λειτουργεί συνδιαστικά με τη παράμετρο h_{dcure} και έχει να κάνει με τη κατασκευή RS σε καλές θέσεις έτσι ώστε να μειώθούν οι απαιτούμενες κινήσεις για cure. Οι παράμετροι $turns_{max}$ και $turns_{remaining}$ υπολογίζονται με βάση τον συνολικό αροθμό των καρτών αλλα και τον αριθμό των καρτών που απομένουν.

- $\mathbf{h_{state_{max}}} = \mathbf{200}$
  Ο αριθμός ατοός είναι η μεγαλύτερη τιμή που μπορεί να πάρουν συνολικά τα παραπάνω αθροίσματα και ο ρόλος του είναι να μετατρέπει τη συνάρτηση h_{state} απο φθίνουσα σε αύξουσα για να μπορούμε να χρησημοποιήσουμε το UCT

Ενώ αρχικά παρατηρούσαμε ότι οι agents έπαιζαν καλά και έκαναν maximize το evaluation function. Μετά απο περίπου 5 γύρους οι agents άρχιζαν να χάνουν έδαφος έναντι του παιχνιδιού. Είδαμε λοιπόν ότι υπήρχαν κομβικά σημεία στα οποία είχαμε ίσο value ανάμεσα σε διαφορετικές κινήσεις. Για να λύσουμε αυτό το πρόβλημα κάναμε χρήση των παραμέτρων W (MDP) με σκοπό να εισάγουμε ένα bias στην επιλογή της κίνησης έτσι ώστε να έχουμε $CD1 > TD > BRS > TRANSPORTATION$ ενώ πολλαπλασιάσαμε το τελικό αποτέλεσμα με τον γύρο στον οποίο βρισκόμαστε για να δώσουμε μεγαλύτερο reward στις ορθολογικές κινήσεις και μεγαλύτερο penalty στις μη ορθολογικές με το πέρασμα του χρόνου.
Η τελική evaluation function φαίνεται παρακάτω:

$$f(state) = round \cdot (0.95h_{state} + 0.05reward)$$

Η επιλογή των βαρών έγινε έγινε πειραματικά μετά απο 3 trials για κάθε agent.

2.3.4 Βελτιστοποιήσεις

Κάθε φορά που τρέχουμε τον αλγόριθμο στην ουσία αρχικοποιούμε ένα δέντρο με ρίζα τη τωρινή κατάσταση του παιχνιδιού και ψάχνουμε τον κόμβο παιδί ο οποίος περιέχει την καλύτερη κίνηση. Αυτό σημαίνει ότι μετά την επιστροφή της πρώτης κίνησης χάνουμε σημαντική πληροφορία η οποία θα μπορούσε να μας οδηγήσει σε πιο αξιόπιστες κινήσεις. Για το λόγο αυτό σε κάθε γύρο αρχικοποιούμε μια φορά το δέντρο και στην ουσία όταν επιστρέψουμε κάποια κίνηση τότε παίρνουμε το υποδέντρο που έχει ρίζα την κίνηση αυτή με σκοπό να βρούμε την επόμενη κίνηση.

3 Συνεργασία

Αφού πάρουμε τα suggestions κάθε παίκτη, τα κάνουμε evaluate με (βάση το evaluation function που αναφέραμε παραπάνω) και τα συγκρίνουμε με το evaluation της δικής μας κίνησης. Κάθε καλύτερη πρόταση από τη δική μας παίρνει +1 πόντο εμπιστοσύνης ενώ κάθε χειρότερη παίρνει +0 πόντους. Αν κάποιος παίκτης φτάσει τους 3 πόντους εμπιστοσύνης τότε παίζουμε τη κίνηση που μας πρότεινε και του μειώνουμε 1 πόντο έτσι ώστε να δώσουμε ευκαιρίες για συνεργασία και σε άλλους παίκτες. Αν έχουμε 2-3 παίκτες στους 3 πόντους τότε παίζουμε τη κίνηση με το καλύτερο evaluation και αφαιρούμε 1 πόντο εμπιστοσύνης από όλους τους παίκτες με σκορ 3 έτσι ώστε να μην αδικήσουμε κανένα παίκτη στον επόμενο γύρο. Η παραπάνω στρατηγική θα μπορούσαμε να πούμε ότι αποτελεί ένα απλοϊκό αλλά αποδοτικό profiling strategy καθώς αν έχω ένα παίκτη ο οποίος έχει ανακαλύψει τη καλύτερη στρατηγική τότε οι υπόλοιποι παίκτες θα συνεργαστούν μαζί του ενώ η πρόταση ενός κακού παίκτη δεν θα επηρεάσει τους άλλους παίκτες.

4 Αξιολόγιση

Κατα μέσο όρο το παιχνίδι διαρκεί 18 γύρους και σε 50 παιχνίδια έχουμε σημειώσει 1 νίκη ενώ σε 2 παιχνίδια οι πράκτορες μας ανακάλυψαν 2 cures και έθεσαν υποέλεγχο περιοχές οι οποίες είχαν κάποιο outbreak. Παρόλα αυτα οι agents μας εμφανίζουν ορθολογική συμπεριφορά σε αρκετές περιπτώσεις και περίπου κάθε 4 γύρους κάποιος agent διαλέγει να συνεργαστεί. Αυτο σημαίνει ότι δεν υπάρχει κάποιος agent που να ακολουθεί κάποια στρατιγική η οποία είναι φανερά καλύτερη απο τη στρατιγική των υπολοίπων παικτών. Αξίζει να σημειωθεί ότι στο φάκελο logs θα βρείτε ένα ολόκληρο παιχνίδι.

5 Μελλοντική Δουλειά

Υπάρχουν αρκετές βελτιστοποιήσεις που θα μπορούσαν να γίνουν τόσο στο κομμάτι του agent όσο και στο κομμάτι του cooperation. Σε ότι αφορά τον agent είναι αναγκαίο να γίνει καλύτερο exploration και tuning των hyperparameters με σκοπό να βελτιώσουμε την απόδοση του σε ότι αφορά το single-agent κομμάτι καθώς ένας κακός παίκτης είναι δυσκολότερο να κάνει καλά suggestions. Στη περίπτωση των suggestions θα μπορούσαμε να πειραματιστούμε με το scoring system, δηλαδή να παίξουμε με τα όρια της εμπιστοσύνης αλλά και την αξιολόγηση των κινήσεων. Για παράδειγμα θα μπορούσαμε να παίρνουμε μία κίνηση αντί για 4 και όχι μόνο από έναν αλλά από πολλούς agents. Αξίζει να σημειωθεί ότι τα αποτελέσματα μπορεί φαινομενικά να μην είναι ενθαρρυντικά αλλά ήταν αναμενόμενα γιατί το Pandemic είναι ένα ανοικτό ερευνητικό πρόβλημα και συνεπώς μια τόσο απλοϊκή προσέγγιση δεν είναι αρκετή για την επίλυση του.

6 Αναφορές

- PAIndemic: A Planning Agent for Pandemic
- Bandit based Monte-Carlo Planning
- A Survey of Monte Carlo Tree Search Methods