ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΛΟΠΟΝΝΗΣΟΥ ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

"Χρήση Ενισχυτικής Μάθησης για Επίλυση Προβλημάτων Εύρεσης Βραχύτερου Μονοπατιού σε Περιβάλλοντα Βιντεοπαιχνιδιών"

Κριλής Χρήστος

Αριθμός Μητρώου 2941

Επιβλέπων καθηγητής: Αλεφραγκής Παναγιώτης

Επιτροπή αξιολόγησης: Αλεφραγκής Παναγιώτης, Χριστοδούλου Σωτήρης, Τζίμας Ιωάννης

Πάτρα, 2023

ΠΡΟΛΟΓΟΣ

Η παρούσα πτυχιακή εργασία εκπονήθηκε στο Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών του Πανεπιστημίου Πελοποννήσου και ο στόχος της είναι η χρήση της Ενισχυτικής Μάθησης (Reinforcement Learning) σε τρισδιάστατο περιβάλλον για την αντιμετώπιση συγκεκριμένων προβλημάτων. Συγκεκριμένα, ο σχεδιασμός και η ανάπτυξη ενός νευρωνικού δικτύου (Neural Network) που είναι ικανό να επιλύει προβλήματα συντομότερης διαδρομής μίας πηγής (Single-Source Shortest Path ή SSSP) και να συγκρίνει τη διανυσματική απόσταση με τον αλγόριθμο του Dijkstra. Το πώς θα δομηθεί το περιβάλλον μάθησης και πώς θα αποζημιωθεί ο πράκτορας στα πλαίσια μοντέλων ανταμοιβής και τιμωρίας είναι τα πιο ουσιώδη αντικείμενα προβληματισμού όταν απαιτείται RL για την επίλυση ενός προβλήματος. Στην παρούσα εργασία θα εξετάσουμε μια σειρά από τεχνικές και εφαρμογές για τη βελτιστοποίηση του χρόνου εκπαίδευσης του πράκτορα και την ανάπτυξη μιας σωστής και προβλέψιμης συμπεριφοράς. Για το έργο αυτό θα χρησιμοποιηθούν οι γλώσσες προγραμματισμού C# και Python, καθώς και εργαλεία εξαγωγής δεδομένων όπως το PyTource και το TensorBoard, ενώ για την εκπαίδευση θα χρησιμοποιηθεί το περιβάλλον Unity Engine σε συνδυασμό με το MLAgents Kit.

Θα ήθελα να επαινέσω τον κ. Παναγιώτη Αλεφραγκή, που μου έδωσε την ευκαιρία να εργαστώ πάνω σε ένα τόσο ενδιαφέρον θέμα, καθώς και για την αμέριστη υποστήριξή του καθ' όλη τη διάρκεια της ακαδημαϊκής μου σταδιοδρομίας.

ΠΕΡΙΛΗΨΗ

Σκοπός της παρούσας διατριβής είναι να διερευνήσει και να αναπτύξει έναν ευφυή πράκτορα εκπαιδευμένο με ενισχυτική μάθηση (Reinforcement Learning) για την επίλυση ενός προβλήματος συντομότερης διαδρομής μίας πηγής (Single Source Shortest Path, SSSP). Συγκεκριμένα, ο πράκτορας πρέπει να προσδιορίσει τη βέλτιστη διαδρομή από ένα σημείο εκκίνησης προς έναν προορισμό που αποτελείται από ενδιάμεσα τμήματα και στη συνέχεια τον τελικό προορισμό. Η κύρια εστίαση αυτής της εργασίας θα είναι ο θεωρητικός σχεδιασμός μιας προσαρμοσμένης συνάρτησης ανταμοιβής (custom reward function) και οι πιθανές δευτερεύουσες επιδράσεις της. Στην παρούσα εργασία, ο πράκτορας θα υλοποιηθεί και θα εκτελεστεί χρησιμοποιώντας τις γλώσσες προγραμματισμού C# και Python, καθώς και τα εργαλεία Tensorboard, PyTourch και Unity Engine. Η παρούσα εργασία περιλαμβάνει έξι κεφάλαια. Στο πρώτο κεφάλαιο θα γίνει μια εισαγωγή και στο δεύτερο κεφάλαιο θα παρουσιαστούν μια βιβλιογραφική ανασκόπηση και βασικές θεωρίες στη βαθιά μάθηση (Deep Learning, DL), την ενισχυτική μάθηση και τον αλγόριθμο εύρεσης βέλτιστων μονοπατιών του Dijkstra. Στο τρίτο κεφάλαιο, θα εξεταστούν ο σχεδιασμός εξατομικευμένων συναρτήσεων ανταμοιβής και μέθοδοι για την αποτροπή της εμφάνισης ανεπιθύμητων συμπεριφορών. Στο τέταρτο κεφάλαιο, θα εξετάσουμε λεπτομερέστερα τα απαιτούμενα εργαλεία, όπως το Unity ML Kit, τους RL αλγορίθμους και τη στρατηγική επίλυσης. Στο πέμπτο κεφάλαιο, θα ορίσουμε το πρόβλημα και θα περιγράψουμε τη μεθοδολογία που θα χρησιμοποιηθεί για την ανάπτυξη ενός RL μοντέλου ικανού να το επιλύσει. Στο έκτο και τελευταίο κεφάλαιο, θα συνοψιστούν τα αποτελέσματα, θα αναλυθούν και θα εξαχθούν τα τελικά συμπεράσματα.

ABSTRACT

The purpose of this thesis is to investigate and develop a Reinforcement learning Intelligent Agent for solving a Single-Source shortest path problem. Specifically, the Agent must determine the optimal route from a starting point to a destination consisting of intermediate segments and then the ultimate destination. The primary focus of this paper will be on the theoretical design of a custom reward function and its potential secondary effects. In this paper, the agent will be implemented and executed using the programming languages C# and Python, as well as the tools Tensorboard, PyTourch, and the Unity Engine. This work contains six chapters. In the first chapter, an introduction will be provided, and in the second, a literature review and fundamental theories in Deep Learning, Reinforcement Learning, and Dijkstra's algorithm will be presented. In the third chapter, we will examine the design of individualized reward functions and methods for preventing the emergence of undesirable behaviors. In Chapter Four, we will examine the required tools, such as the Unity ML Kit, the RL algorithms, and the solution strategy in greater detail. In Chapter Five, we will define the problem and describe the methodology that will be used to develop a RL model capable of solving it. In the sixth and final chapter, the results will be summarized, analyzed, and the final conclusions derived.

Περιεχόμενα

Λίστα Σχημάτων:

Λίστα Πινάκων:

Λίστα Εικόνων:

Λίστα Λέξεις Κλειδιά:

RL – Reinforcement Learning

SSSP – Single Source Shortest Path

NN – Neural Network

DL – Deep Learning

RF – Reward Function

ANN - artificial neural networks

CNN – Convolutional Neural Network

DNN – Deep Neural Network

PPO – Proximal Policy Optimization

RND – Random Network Distillation

BC – Behavior Cloning

NPC – Non-Player Character

Ο πηγαίος κώδικας, το περιβάλλον, οι οδηγίες και τα μοντέλα μπορούν να βρεθούν στο παρακάτω σύνδεσμο :

https://github.com/ChristosKrilisDev/ml-agents-thesis-project

Κεφάλαιο 1 Εισαγωγή

1.1 Βιβλιογραφικό Υπόβαθρο

Τα τελευταία χρόνια, οι εφαρμογές τεχνητής νοημοσύνης γίνονται όλο και πιο δημοφιλείς. Στα βιντεοπαιχνίδια, η Ενισχυτική Μάθηση (Reinforcement Learning) έχει δώσει εκπληκτικά αποτελέσματα. Για να ολοκληρωθεί αυτής της αποστολής, θα αναπτυχθεί ένας ευφυής πράκτορας και θα εκπαιδευτεί χρησιμοποιώντας αλγορίθμους ενισχυτικής μάθησης. Ο Alpha Go, ένας παίκτης τεχνητής νοημοσύνης, που εκπαιδεύτηκε με αλγορίθμους βαθιάς ενισχυτικής μάθησης (Deep RL), νίκησε τρεις φορές τον καλύτερο, άνθρωπο, παίκτη Go στον κόσμο στο Future of Go Summit το 2017. Η επιτυχία της RL σε αυτόν τον τομέα εντυπωσιάζει τον κόσμο και έχουν ήδη ξεκινήσει πολυάριθμα ερευνητικά έργα, όπως προσομοιώσεις αυτόνομων αυτοκινήτων και μελέτες της βιολογικής εξέλιξης και της φυσικής επιλογής. Οι τεχνικές βαθιάς μάθησης, όπως τα συνελικτικά νευρωνικά δίκτυα (Convolutional Neural Network), συμβάλλουν σημαντικά σε αυτό, επειδή αντιμετωπίζουν τα ζητήματα της αντιμετώπισης δεδομένων εισόδου υψηλής διάστασης και της εξαγωγής χαρακτηριστικών.

1.2 Στόχος Της Εργασίας

Ο στόχος αυτού του έργου είναι να δημιουργηθεί ένας RL πράκτορας που εκπαιδευτεί σε διαφορετικούς RL αλγορίθμους και τεχνικές για την επίλυση ενός προβλήματος Single-Source Shortest Path (SSSP) και να συγκριθεί η απόδοσή τους. Η RL είναι πολύ εφαρμόσιμη σε έργα

όπου μπορούμε εύκολα να ορίσουμε ποια είναι η επιτυχία, αλλά δεν ξέρουμε τι πρέπει να κάνουμε για να φτάσουμε εκεί. Ένα πολύ σημαντικό μέρος στα προβλήματα ενισχυτική μάθησης είναι ο χρόνος που χρειάζεται ο πράκτορας για να καταλήξει σε ένα σύνολο αποφάσεων που θα ολοκληρώνουν το πρόβλημα με επιτυχία, και προφανώς, υπό το πρίσμα ορισμένων λογικών κανόνων (π.χ., σε μια προσομοίωση αυτοκινήτου, είναι πιθανόν να θέλουμε οι επιβάτες να παραμείνουν ζωντανοί κατά τη μεταφορά). Πολύ σημαντικό ρόλο παίζει η ανταμοιβή και η τιμωρία (reward και punishment) που λαμβάνει ο πράκτορας για τις αποφάσεις του. Καθώς θα μάθει να κάνει ακριβώς αυτό για το οποίο ανταμείφθηκε και τίποτα περισσότερο ή λιγότερο. Ο σχεδιασμός και η ανάπτυξη μιας συνάρτησης ανταμοιβής είναι επομένως μια αρκετά μεγάλη και σημαντική πρόκληση, έχοντας καθοριστικό ρόλο, αφού είναι υπεύθυνη για την αποδοτικότητα και την αποτελεσματικότητα που θα έχει ο πράκτορας. Οι γενικοί στόχοι αυτού του έργου παρατίθενται παρακάτω.

- Δημιουργία ενός πράκτορα RL για την επίλυση προβλημάτων SSSP
- Σχεδιασμός της συνάρτησης ανταμοιβής (Reward Function) και εφαρμογή τεχνικών εκπαίδευσης βελτιστοποίησης χρόνου
- Σύγκριση των διαφορών και των αποτελεσμάτων μεταξύ διαφορετικών RL μοντέλων

1.3 Σύνοψη

Η μελέτη αυτή ξεκινά με μια ανασκόπηση της βιβλιογραφίας σχετικά με τη βαθιά μάθηση (DL) και την ενισχυτική μάθηση (RL). Κάθε κεφάλαιο περιέχει μια σύντομη επισκόπηση της ιστορίας

του πεδίου και των σχετικών μεθοδολογιών. Στα κεφάλαια 3,4 και 5 θα δούμε τα reward signals, τον σχεδιασμό τις συνάρτησης ανταμοιβής, την μεθοδολογία και τέλος, στο κεφάλαιο 6 τα αποτελέσματα και συμπεράσματα.

Κεφάλαιο 2 Βιβλιογραφικό Υπόβαθρο

Εισαγωγή

Αυτό το κεφάλαιο παρουσιάζει τις τεχνικές που χρησιμοποιούνται για τη δημιουργία ενός πράκτορα ικανού να επιλύει προβλήματα SSSP. Η βαθιά μάθηση και η ενισχυτική μάθηση (RL) είναι οι δύο κύριες ενότητες. Θα περιγραφή το συνοπτικό ιστορικό και τα σημαντικά ορόσημα, καθώς και η εισαγωγή βασικών μεθόδων.

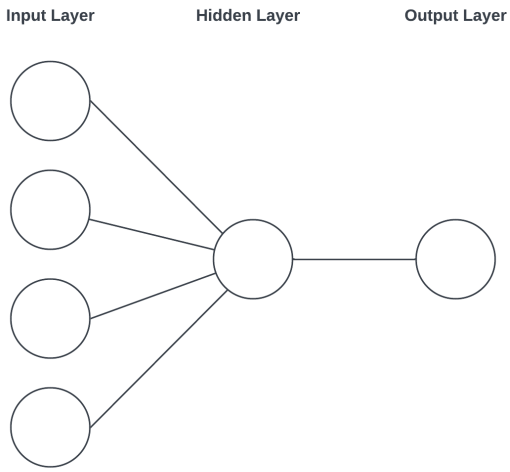
2.1 Βαθιά Μάθηση (Deep Learning)

Η βαθιά μάθηση είναι ένας τύπος μοντέλου μηχανικής μάθησης που βασίζεται στα Τεχνητά Νευρωνικά Δίκτυα (ANN)**.** Υπάρχουν δύο τύποι μοντέλων βαθιάς μάθησης που χρησιμοποιούνται σήμερα ευρέως. Το Recurrent Neural Network είναι μία από τις τεχνικές που αποδεικνύει την αποτελεσματικότητά της στην Επεξεργασία Φυσικής Γλώσσας. Το Συνελικτικό Νευρωνικό Δίκτυο (CNN) είναι το δεύτερο στοιχείο που έπαιξε σημαντικό ρόλο στη βαθιά ενισχυτική μάθηση. Είναι ένα από τα πιο αποτελεσματικά μοντέλα για θέματα όρασης υπολογιστών, όπως η αφαίρεση αντικειμένων και η ταξινόμηση εικόνων. Αυτή η ενότητα παρέχει μια επισκόπηση της βαθιάς μάθησης και εκτενείς πληροφορίες σχετικά με το CNN.

2.1.1 Βιβλιογραφικό Υπόβαθρο στη Βαθιά Μάθηση

Ένα ANN είναι ένα υπολογιστικό σύστημα εμπνευσμένο από τα βιολογικά νευρωνικά δίκτυα, τα οποία προτάθηκαν για πρώτη φορά από τον νευροφυσιολόγο McCulloch [14]. Το 1957, ο Frank [16] ανακάλυψε το Perceptron. Τρία χρόνια αργότερα, οι έρευνές του αποδεικνύουν ότι ο αλγόριθμος αυτός μπορεί να αναγνωρίσει ορισμένα αλφαβητικά στοιχεία [17]. Ο Marvin απέδειξε, ωστόσο, ότι ένα μόνο στρώμα perceptron δεν μπορεί να λύσει το πρόβλημα XOR [15]. Αυτό σταμάτησε την ανάπτυξη των ANN μέχρι το 1988, όταν οι Rumelhart et al. απέδειξαν με το perceptron πολλαπλών στρωμάτων, γνωστό και ως Neural Network (NN), και τον αλγόριθμο backpropagation [18] ότι μπορούν να διδαχθούν χρήσιμες αναπαραστάσεις. Οι LeCun et al. [13] χρησιμοποίησαν ένα NN πέντε επιπέδων και backpropagation για να λύσουν το πρόβλημα ταξινόμησης ψηφίων ένα χρόνο αργότερα, με εξαιρετικά αποτελέσματα. Το καινοτόμο μοντέλο του είναι γνωστό ως LeNet και χρησιμεύει ως βάση για το CNN. Ο Fukushima πρότεινε το Neocognitron, ένα αυτοοργανωμένο μοντέλο NN πολλαπλών επιπέδων, ως θεμέλιο του CNN [10]. Αυτό το μοντέλο είχε καλές επιδόσεις σε εργασίες ανίχνευσης αντικειμένων, επειδή είναι αναίσθητο στη θέση. Όπως αναφέρθηκε προηγουμένως, οι LeCun et al. επινόησαν το LeNet το 1998 και πέτυχαν ποσοστό σφάλματος μικρότερο από 1% στο σύνολο δεδομένων χειρόγραφων MNIST ψηφίων [13]. Η συσχέτιση και η υποδειγματοληψία, γνωστές σήμερα ως συγχώνευση στρωμάτων και συνάθροιση στρωμάτων, χρησιμοποιήθηκαν από το μοντέλο για να μετατρέψει τις αρχικές εικόνες σε διανύσματα χαρακτηριστικών και να εκτελέσει ταξινόμηση με πλήρως συνδεδεμένα στρώματα. Ταυτόχρονα, ορισμένα μοντέλα δικτύων NN επιδεικνύουν αποδεκτές επιδόσεις στην αναγνώριση προσώπων [12], στην αναγνώριση ομιλίας [20] και στην ανίχνευση αντικειμένων. Ωστόσο, η απουσία αξιόπιστης θεωρίας σταμάτησε την έρευνα για τα CNN για

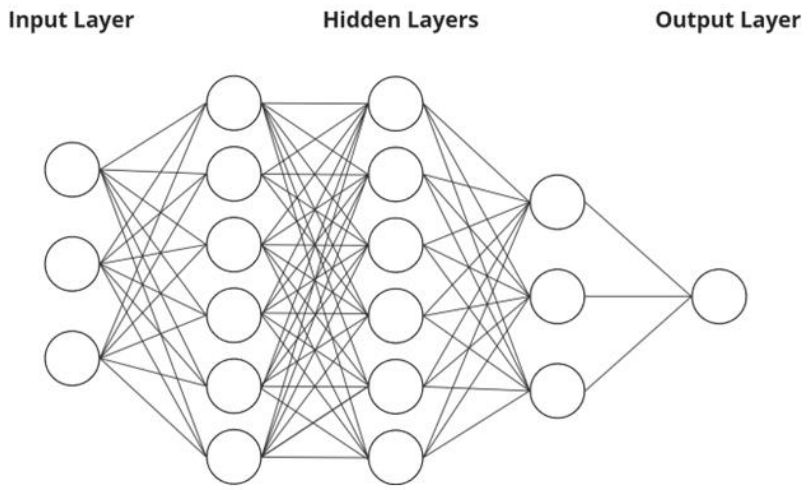
πολλά χρόνια. Ο Alex και η ομάδα του πέτυχαν ποσοστό σφάλματος 16,4% στην πρόκληση του 2012 ImageNet Large Scale Visual Recognition (ILSVRC) [8] χρησιμοποιώντας ένα βαθύ νευρωνικό δίκτυο (DNN) οκτώ επιπέδων (AlexNet) [11]. Αυτό ήταν ένα σημαντικό αποτέλεσμα σε σύγκριση με το 26,2% του συμμετέχοντος που κατέλαβε τη δεύτερη θέση. Εκτός από το LeNet, το AlexNet χρησιμοποίησε οκτώ επίπεδα για την εκπαίδευση του ταξινομητή με αύξηση δεδομένων, εγκατάλειψη και ReLU, μειώνοντας έτσι το πρόβλημα της υπερπροσφοράς.



Εικόνα 1 Ένα απλό NN δίκτυο

Μια άλλη σημαντική ανακάλυψη ήταν ότι ο παράλληλος υπολογισμός με πολλαπλές GPU μπορεί να μειώσει δραστικά τον χρόνο εκπαίδευσης. Δύο χρόνια αργότερα, οι Simonyan και Zisserman εισήγαγαν ένα NN δεκαέξι επιπέδων (VG-GNet) και κέρδισαν το πρώτο βραβείο για εργασίες ταξινόμησης και εντοπισμού στο ILSVRC 2014. Εκείνη την εποχή, το ποσοστό σφάλματος 7,3% αυτού του μοντέλου το καθιστούσε το πιο προηγμένο διαθέσιμο μοντέλο. Η αποτελεσματικότητα του CNN μπορεί να ενισχυθεί με τη χρήση μικρότερου μεγέθους φίλτρου και βαθύτερου δικτύου, όπως παραδειγματικά δείχνει το VGGNet. Όλα τα φίλτρα του μοντέλου είχαν μέγιστο μέγεθος

3x3, ενώ τα δύο πρώτα στρώματα του AlexNet ήταν 11x11 και 5x5. Την ίδια χρονιά, το GoogLeNet, το κορυφαίο μοντέλο των εργασιών ταξινόμησης και ανίχνευσης του ILSVRC 2014, χρησιμοποιήθηκε για πρώτη φορά στην πρόταση επίλυσης προβλημάτων ταξινόμησης του Lin [23]. Η αρχή αντικατέστησε έναν κόμβο με ένα δίκτυο που αποτελείται από πολλαπλά σύνθετα στρώματα και στρώματα συνάθροισης, τα συνέθεσε και στη συνέχεια τα πέρασε στο επόμενο στρώμα.



Εικόνα 2 Απλό DNN με τρία κρυφά στρωματά

Αυτή η τροποποίηση έκανε τις επιλογές μεταξύ δύο στρωμάτων πιο ευέλικτες. Ένα άλλο ζήτημα με το DNN ήταν η υποβάθμιση με αποτέλεσμα την εντατική εκπαίδευση λόγω της πολυπλοκότητας της βελτιστοποίησης. Για να αντιμετωπιστεί αυτό το ζήτημα, οι He et al. πρότειναν ένα βαθύ πλαίσιο υπολειμματικού δικτύου (ResNet) που χρησιμοποιεί μια χαρτογράφηση υπολειμματικού δικτύου αντί για ρητή διαστρωμάτωση στρωμάτων [9]. Αυτό το μοντέλο κέρδισε το πρωτάθλημα ILSVRC του 2015 με ποσοστό σφάλματος 3,57 %. Τα δεδομένα του αποδεικνύουν ότι αυτό το νέο πλαίσιο δεν είναι μόνο ικανό να επιλύσει ζητήματα

υποβάθμισης, αλλά και να ενισχύσει την υπολογιστική αποδοτικότητα. Υλοποιήθηκαν διάφορες παραλλαγές βασισμένες στο ResNet, συμπεριλαμβανομένων των Inception-ResNet [19] και DenseNet [36]. Η πρώτη ενσωμάτωσε τις βελτιωμένες διαδικασίες σύλληψης του ResNet. Στο επόμενο μοντέλο, κάθε ζεύγος σύνθετων στρωμάτων συνδέθηκε. Αυτή η τροποποίηση μείωσε τα προβλήματα κλίσης εξαφάνισης και ενίσχυσε τη διάδοση των χαρακτηριστικών.

2.2 Ενισχυτική Μάθηση (Reinforcement Learning)

Η ενισχυτική μάθηση (RL) είναι μια διαβάθμιση της μηχανικής μάθησης που επιδιώκει τη μεγιστοποίηση της ανταμοιβής κατά τη λήψη αποφάσεων. Ο πράκτορας και το περιβάλλον είναι τα κύρια συστατικά της ενισχυτικής μάθησης. Μετά από κάθε ενέργεια, ο πράκτορας θα λάβει διατήρηση και ανταμοιβή από το περιβάλλον. Για να αναπτύξει μια ανώτερη πολιτική, θα συνεχίσει να αλληλοεπιδρά με το περιβάλλον και να βελτιώνει σταδιακά τις ικανότητές του στη λήψη αποφάσεων μέχρι να συγκλίνουν οι πολιτικές.

2.2.1 Βιβλιογραφικό Υπόβαθρο στη Ενισχυτική Μάθηση

Τα τελευταία χρόνια, η RL έχει αποκτήσει μεγάλη δημοτικότητα λόγω του Alpha Go, ενός προγράμματος υπολογιστή που μπορεί να νικήσει έναν άνθρωπο εμπειρογνώμονα στο παιχνίδι Go [30]. Στο Future of Go Summit το 2017, ο Alpha Go Master εξέπληξε τον κόσμο νικώντας τον

Ke Jie, τον μεγαλύτερο παίκτη του κόσμου στο Go, και στα τρία παιχνίδια. Ωστόσο, η έρευνα της RL ξεκίνησε πρόωρα. Σύμφωνα με τον Sutton [32], η πρώιμη ιστορία της RL μπορεί να χωριστεί σε δύο μεγάλα σκέλη. Μεταξύ αυτών, ο βέλτιστος έλεγχος ήταν το ένα μέρος. Ο Bellman [24] εισήγαγε τη θεωρία του Δυναμικού Προγραμματισμού για να αντιμετωπίσει τα προβλήματα βέλτιστου ελέγχου που προκύπταν, τα οποία ονόμασε "επεξεργασία αποφάσεων σε πολλά στάδια", το 1954.

Σε θεωρητικό επίπεδο πρότεινε τη "λειτουργική εξίσωση", γνωστή και ως εξίσωση Bellman. Αν και η DP(Decision Process) ήταν μια από τις πιο αποτελεσματικές μεθόδους για την επίλυση προβλημάτων βέλτιστου ελέγχου εκείνη την εποχή, οι υψηλές υπολογιστικές απαιτήσεις, που ονομάστηκαν από τον Bellman "κατάρα της διαστατικότητας", καθιστούσαν δύσκολη την εφαρμογή της [23]. Τρία χρόνια αργότερα, ανέπτυξε το μοντέλο των Διαδικασιών Απόφασης Markov (Markov Decision Process ή MDPs) για να χαρακτηρίσει τις διακριτές ντετερμινιστικές διαδικασίες [22]. Αυτό το ντετερμινιστικό σύστημα και η έννοια της συνάρτησης που περιγράφεται από την εξίσωση Bellman αποτελούν τη θεμελιώδη θεωρία της σύγχρονης RL.
Στο σκέλος του βέλτιστου ελέγχου, η επίλυση προβλημάτων απαιτούσε μια ολοκληρωμένη κατανόηση του περιβάλλοντος και ήταν ανεφάρμοστη στην πλειονότητα των ζητημάτων του πραγματικού κόσμου. Η εστίαση του νήματος δοκιμής και σφάλματος ήταν στην ανατροφοδότηση και όχι στο ίδιο το περιβάλλον. Στο βιβλίο "Animal Intelligence" του Edward Thorndike, ο όρος "Law of Effect" χρησιμοποιήθηκε για πρώτη φορά για να περιγράψει τη θεμελιώδη έννοια της δοκιμής και του σφάλματος, συμπεριλαμβανομένων των όρων "τμηματική" και "συνειρμική" [33]. Παρόλο που η επιβλεπομένη μάθηση δεν ήταν "τμηματική", ορισμένοι ερευνητές

εξακολουθούσαν να την μπερδεύουν με την ενισχυτική μάθηση και να επικεντρώνονται στην αναγνώριση προτύπων [1, 42]. Αυτό είχε ως αποτέλεσμα σπάνιες μελέτες γνήσιας μάθησης δοκιμής και λάθους μέχρι που ο Klopf εντόπισε τη διάκριση μεταξύ της επιβλεπομένης μάθησης και της ενισχυτικής μάθησης: το κίνητρο για να κερδίσει κανείς περισσότερες ανταμοιβές από το περιβάλλον [1]. Ωστόσο, υπήρχαν ακόμη αξιοσημείωτες εξελίξεις, όπως ο κανόνας RL "επιλεκτικής προσαρμογής bootstrap" του Widrwo το 1973 [35]. Και τα δύο σκέλη μπορούν να βρεθούν στη σύγχρονη RL. Η Μάθηση Χρονικής Διαφοράς (temporal difference ή TD) είναι μια τεχνική που προέρχεται από την ψυχολογία της μάθησης των ζώων και προβλέπει ότι οι μελλοντικές τιμές εξαρτώνται από το τρέχον σήμα. Ο Samuel επινόησε και εφάρμοσε πρώτος αυτή την έννοια. Το 1972, ο Klopf επινόησε την έννοια της "γενικευμένης ενίσχυσης" και συνέδεσε τη μάθηση μέσω δοκιμής και σφάλματος με την ψυχολογία μάθησης των ζώων [25]. Με βάση μια ιδέα του Klopf [21], ο Sutton σχεδίασε και υλοποίησε το 1983 την αρχιτεκτονική actor-critic για τη μάθηση με δοκιμή και σφάλμα. Πέντε χρόνια αργότερα, πρότεινε τους αλγορίθμους TD, οι οποίοι χρησιμοποιούσαν πρόσθετα δεδομένα για την ενημέρωση της πολιτικής και έκαναν την εκμάθηση TD μια γενική μέθοδο για την πρόβλεψη απροσδιόριστων προβλημάτων [31]. Ο Chris χρησιμοποίησε μεθόδους βέλτιστου ελέγχου για την αντιμετώπιση προβλημάτων χρονικής διαφοράς ένα χρόνο αργότερα και επινόησε τον αλγόριθμο Q-learning, ο οποίος υπολόγιζε την καθυστερημένη ανταμοιβή από τη συνάρτηση αξίας-δράσης [34]. Οι Rummery και Niranjan πρότειναν ένα online σύστημα Q-learning γνωστό ως SARSA [29] το 1994. Στο SARSA, ο πράκτορας χρησιμοποιούσε την ίδια πολιτική καθ' όλη τη διάρκεια της διαδικασίας εκμάθησης, ενώ στο Q-learning, ο πράκτορας επέλεγε πάντα τη βέλτιστη δράση με βάση τη συνάρτηση αξίας. Η DeepMind πρότεινε τον αλγόριθμο Deep Q-learning Network (DQN), ο οποίος χρησιμοποίησε ένα σύνθετο NN για την επίλυση καταστάσεων υψηλής διάστασης σε προβλήματα RL [27] με την

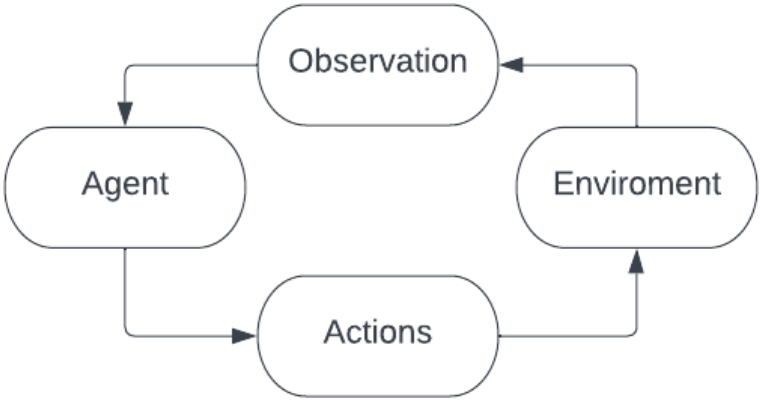
ανάπτυξη του DNN. Δύο χρόνια αργότερα, πρόσθεσαν μια στοχευμένη πολιτική στο DQN για να αυξήσουν τη σταθερότητά του [28]. Εκτός από τον συνδυασμό βαθιάς μάθησης και ενισχυτικής μάθησης, το χαρακτηριστικό γνώρισμα του DQN ήταν ο μηχανισμός αναπαραγωγής εμπειριών. Ο Mnih δοκίμασε μια τυχαία δειγματοληψία μιας μίνι παρτίδας για τη βελτιστοποίηση του NN με βάση μια έννοια που πρότεινε ο Lin [26] προκειμένου να επιλύσουν προβλήματα εξάρτησης που αντιμετωπίζονται κατά τη βελτιστοποίηση του CNN. Αυτός ο μηχανισμός ενισχύθηκε το 2015 με την ποσοτικοποίηση της σημασίας της εμπειρίας με το σφάλμα της χρονικής διαφοράς [36]. Εν τω μεταξύ, ο Wang πρότεινε το Dueling DQN, το οποίο χρησιμοποίησε μια συνάρτηση πλεονεκτήματος ανακαλύπτοντας την τιμή μιας κατάστασης χωρίς να εκτιμά την τιμή ενέργειας κάθε κατάστασης. Αυτή η νέα αρχιτεκτονική NN ήταν επωφελής όταν η σχέση μεταξύ των ενεργειών και του περιβάλλοντος ήταν αδύναμη. Το Double DQN προτάθηκε από την DeepMind το 2016, επιδεικνύοντας μεγαλύτερη σταθερότητα πολιτικής με τη μείωση των υπερεκτιμημένων τιμών δράσης. Παρά το γεγονός ότι ορισμένοι αλγόριθμοι που βασίζονται στο DQN επιτυγχάνουν επιδόσεις ανθρώπινου επιπέδου σε παιχνίδια Atari, δεν είναι το ίδιο επιτυχημένοι σε ορισμένα παιχνίδια. Το DQN ήταν μια μέθοδος βασισμένη σε τιμές, οπότε η επιλογή δράσης εξαρτιόταν από τις τιμές των δράσεων. Σε ορισμένα παιχνίδια, όπως το πέτρα-ψαλίδι-χαρτί, η τυχαία επιλογή δράσης μπορεί να είναι η βέλτιστη στρατηγική. Ο Sutton πρότεινε μια διαβαθμισμένη πολιτική για την επίλυση αυτού του ζητήματος, η οποία επέτρεπε στον πράκτορα να βελτιστοποιεί έμμεσα την πολιτική. Το OpenAI πρότεινε μια νέα οικογένεια αλγορίθμων, όπως η βελτιστοποίηση εγγύς πολιτικής (Proximal Policy Optimization ή PPO) [38] με βάση αυτή την έρευνα. Αυτή η απλή τροποποίηση βελτιώνει την απόδοση του RoboschoolHumanoidFlagrun. Λόγω της υψηλής διαστατικότητας του χώρου-δράσης και του γεγονότος ότι η βασική μέθοδος αξιολόγησης πολιτικής χρησιμοποιεί δεδομένα από πλήρη επεισόδια, η διακύμανση της εκτίμησης ήταν υψηλή.

Παρόμοια με την προσέγγιση με βάση την αξία, η μέθοδος Actor-critic προτάθηκε [18] ως λύση σε αυτό το ζήτημα. Σε αντίθεση με την πολιτική αξιολόγησης, αυτή η προσέγγιση αξιολογούσε την επιλεγμένη δράση χρησιμοποιώντας έναν κριτή. Αυτό επέτρεψε την ενημέρωση της πολιτικής μετά από κάθε απόφαση, γεγονός που όχι μόνο μείωσε τη διακύμανση αλλά και επιτάχυνε τη σύγκλιση. Ο γνωστός βελτιωμένος κριτής-πράκτορας**,** είναι ο κριτής-πράκτορας με ασύγχρονο πλεονέκτημα (A3C) [26]. Παρόμοια με το Dueling DQN, αυτή η μέθοδος χρησιμοποιεί τη συνάρτηση πλεονεκτήματος για την εκτίμηση της συνάρτησης αξίας και χρησιμοποιεί παράλληλο υπολογισμό, ο οποίος μπορεί να επιταχύνει σημαντικά τη διαδικασία εκμάθησης.

2.2.2 RL και Αλληλεπίδραση του Πράκτορα με το Περιβάλλον

Στην ενισχυτική μάθηση (RL), ο πράκτορας αλληλεπιδρά με το περιβάλλον μέσω ενός συνεχούς κύκλου ενεργειών, παρατηρήσεων, μεταβάσεων κατάστασης και ανταμοιβών. Σε κάθε χρονικό βήμα, ο πράκτορας παρατηρεί την κατάσταση του περιβάλλοντος, αντιπροσωπεύοντας τις σχετικές πληροφορίες που απαιτούνται για τη λήψη αποφάσεων. Με βάση την παρατηρούμενη κατάσταση, ο πράκτορας επιλέγει μια ενέργεια για να εκτελέσει, η οποία οδηγεί σε **μετάβαση** σε μια νέα κατάσταση του περιβάλλοντος. Ως συνέπεια της δράσης και της μετάβασης στην κατάσταση, ο πράκτορας λαμβάνει ένα σήμα ανταμοιβής, το οποίο υποδεικνύει την επιθυμητότητα των ενεργειών του. Ο πράκτορας χρησιμοποιεί την παρατηρούμενη κατάσταση, την ενέργεια που εκτελεί, την κατάσταση που προκύπτει και τη ληφθείσα ανταμοιβή για να ενημερώσει τη γνώση του και να βελτιώσει την πολιτική λήψης αποφάσεων. Αυτή η επαναληπτική διαδικασία μάθησης περιλαμβάνει μια ισορροπία μεταξύ εξερεύνησης και εκμετάλλευσης, καθώς

ο πράκτορας εξερευνά διαφορετικές ενέργειες και τις συνέπειές τους για να ανακαλύψει τις βέλτιστες στρατηγικές, ενώ παράλληλα ευνοεί ενέργειες με υψηλότερες αναμενόμενες ανταμοιβές.

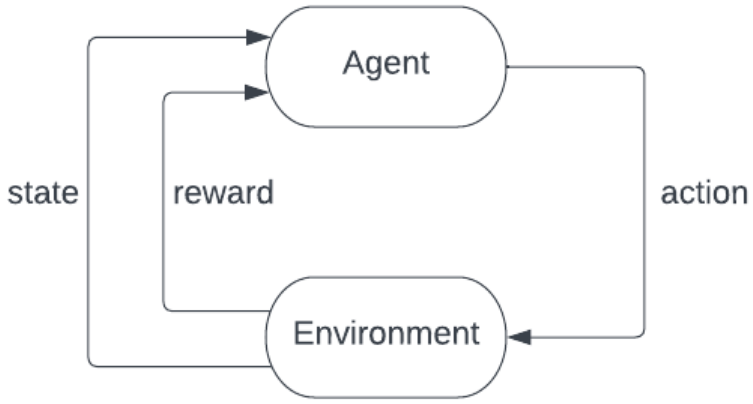


Εικόνα 3 Αλληλεπίδραση του RL πράκτορα με το περιβάλλον

Μέσω επαναλαμβανόμενων αλληλεπιδράσεων με το περιβάλλον, ο πράκτορας βελτιώνει την πολιτική του και προσαρμόζει τη συμπεριφορά του ώστε να επιτυγχάνει υψηλότερες αθροιστικές ανταμοιβές με την πάροδο του χρόνου. Η αλληλεπίδραση πράκτορα-περιβάλλοντος βρίσκεται στον πυρήνα της RL, επιτρέποντας την εκπαίδευση ευφυών πρακτόρων που μπορούν να βελτιστοποιούν τη λήψη αποφάσεων και να αποδίδουν καλά σε δυναμικά και πολύπλοκα περιβάλλοντα.

Αυτή η αλληλεπίδραση μεταξύ του πράκτορα RL και του περιβάλλοντος είναι κρίσιμη για τη διαδικασία μάθησης του πράκτορα. Ο πράκτορας μπορεί να αποκτήσει σημαντική εμπειρία και να βελτιώσει τις ικανότητές του στη λήψη αποφάσεων μελετώντας συνεχώς το περιβάλλον,

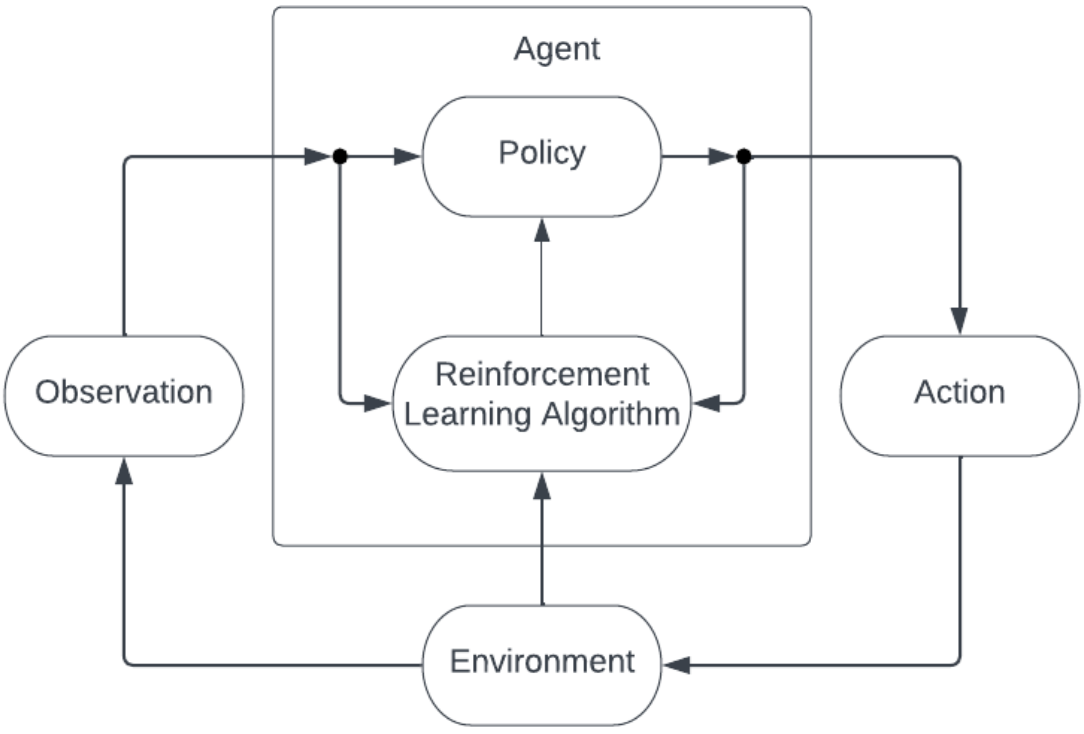
δρώντας και κερδίζοντας ανταμοιβές. Η παρατηρούμενη κατάσταση ενημερώνει τον πράκτορα για την παρούσα κατάσταση και επηρεάζει την επιλογή των ενεργειών του. Η επιλεγμένη δράση προκαλεί μια μετάβαση κατάστασης, επιτρέποντας στον πράκτορα να εξερευνήσει άλλα τμήματα του κόσμου και να κατανοήσει τις συνέπειες των αποφάσεών του.



Εικόνα 4 Αλληλεπίδραση πράκτορα-περιβάλλοντος σε πλαίσιο ενισχυτικής μάθησης

Το σήμα ανταμοιβής ενημερώνει τον πράκτορα σχετικά με την επιθυμητότητα ή την ποιότητα των πράξεων του. Οι θετικές ανταμοιβές προωθούν δραστηριότητες που οδηγούν σε θετικά αποτελέσματα, ενώ οι αρνητικές ανταμοιβές αποθαρρύνουν τις πράξεις που οδηγούν σε αρνητικά αποτελέσματα. Ο πράκτορας μαθαίνει να αναγνωρίζει με την πάροδο του χρόνου ορισμένες δραστηριότητες με μεγαλύτερες ανταμοιβές και μεταβάλλει ανάλογα την πολιτική του. Η πολιτική του πράκτορα ενημερώνεται κατά τη διάρκεια της διαδικασίας μάθησης με βάση τις παρατηρούμενες ανταμοιβές και τις μεταβάσεις καταστάσεων. Για τη μεγιστοποίηση της πολιτικής λήψης αποφάσεων του πράκτορα, οι αλγόριθμοι RL χρησιμοποιούν τεχνικές όπως η εκτίμηση των συναρτήσεων αξίας, ο υπολογισμός των κλίσεων και η ενημέρωση των εκτιμήσεων της αξίας των ενεργειών. Ο πράκτορας βελτιώνει την πολιτική του μέσω επαναληπτικής μάθησης,

βελτιώνοντας σταδιακά τις επιδόσεις του και τις ικανότητες λήψης αποφάσεων. Ένα άλλο βασικό χαρακτηριστικό της αλληλεπίδρασης πράκτορα-περιβάλλοντος στην RL είναι η ισορροπία μεταξύ εξερεύνησης και εκμετάλλευσης. Αρχικά, ο πράκτορας εξερευνά το περιβάλλον του προκειμένου να αποκτήσει πληροφορίες και να μάθει για τα αποτελέσματα διαφόρων συμπεριφορών. Αυτή η διερεύνηση επιτρέπει στον πράκτορα να εντοπίσει πιθανώς καλύτερες επιλογές, αποφεύγοντας παράλληλα την πρόωρη σύγκλιση σε κατώτερες λύσεις. Καθώς ο πράκτορας αναπτύσσει εμπειρία και εμπιστοσύνη στην πολιτική του, αρχίζει να εκμεταλλεύεται τις πληροφορίες που έχει μάθει επιλέγοντας πράξεις που έχουν προηγουμένως προσφέρει υψηλότερες ανταμοιβές.



Εικόνα 5 Πράκτορας Ενισχυμένης Μάθησης

Η αλληλεπίδραση πράκτορα-περιβάλλοντος στην ενισχυτή μάθηση αποτελεί ένα ισχυρό πλαίσιο για την εκπαίδευση ευφυών πρακτόρων σε διάφορους τομείς, όπως η ρομποτική και τα αυτόνομα

συστήματα, καθώς και στα βιντεοπαιχνίδι. Οι πράκτορες RL μπορούν να αποκτήσουν γνώση, να αναπτύξουν τις ικανότητές τους στη λήψη αποφάσεων και τελικά να επιτύχουν αποτελέσματα υψηλής απόδοσης μέσω της προσομοίωσης και της αλληλεπίδρασης με το περιβάλλον. Αυτή η αλληλεπίδραση χρησιμεύει ως βάση για τους αλγορίθμους ενισχυτής μάθησης, ανοίγοντας το δρόμο για τη δημιουργία ευφυών συστημάτων ικανών να χειρίζονται δύσκολες προκλήσεις.

2.2.3 Ο Ρόλος της RL στην Ανάπτυξη Βιντεοπαιχνίδια

Ως ισχυρή τεχνολογία μηχανικής μάθησης, η RL έχει τη δυνατότητα να αλλάξει σημαντικά την παραγωγή βιντεοπαιχνιδιών. Οι προγραμματιστές μπορούν να κατασκευάσουν ευφυείς πράκτορες, προσαρμόσιμες εμπειρίες παιχνιδιού και καθηλωτικούς εικονικούς κόσμους ενσωματώνοντας αλγορίθμους RL σε συστήματα παιχνιδιών.

Η ενισχυτική μάθηση μπορεί να χρησιμοποιηθεί για την ανάπτυξη ευφυών NPC χαρακτήρων με ρεαλιστική και δυναμική συμπεριφορά. Οι NPC μπορούν να προσαρμόζονται και να μαθαίνουν από τις αλληλεπιδράσεις τους με τον κόσμο του παιχνιδιού και τους άλλους παίκτες εκπαιδεύοντάς τους με αλγορίθμους ενισχυτικής μάθησης, με αποτέλεσμα πιο απαιτητικές και διασκεδαστικές εμπειρίες παιχνιδιού. Επιπλέον, η ενισχυτική μάθηση επιτρέπει στους δημιουργούς παιχνιδιών να σχεδιάζουν προσαρμοστικά συστήματα παιχνιδιών που τροποποιούν δυναμικά τα επίπεδα δυσκολίας, την παραγωγή περιεχομένου και τις δυναμικές του παιχνιδιού με βάση τις επιδόσεις και τις προτιμήσεις των χρηστών. Αυτή η προσαρμογή ενισχύει τη δέσμευση και την ευχαρίστηση

των παικτών. Οι αλγόριθμοι ενισχυτικής μάθησης μπορούν επίσης να χρησιμοποιηθούν για τη δημιουργία διαδικαστικού περιεχομένου, το οποίο παράγει μια ποικιλία δυναμικού υλικού παιχνιδιού, όπως επίπεδα, περιβάλλοντα και αντικείμενα. Αυτό όχι μόνο μειώνει το κόστος παραγωγής, αλλά παρέχει επίσης στους παίκτες νέες και διαρκώς μεταβαλλόμενες εμπειρίες, αυξάνοντας την επαναληψιμότητα.

Επιπλέον, η ενισχυτική μάθηση υποστηρίζει την επαναληπτική εξισορρόπηση και δοκιμή των συστημάτων του παιχνιδιού, επιτρέποντας στους δημιουργούς να τελειοποιήσουν τις ρυθμίσεις, να αποκαλύψουν πιθανά σφάλματα και να εξασφαλίσουν μια πιο ισορροπημένη και ελκυστική εμπειρία παιχνιδιού. Λόγω των περίπλοκων σεναρίων λήψης αποφάσεων, αρκετά είδη παιχνιδιών, όπως τα παιχνίδια στρατηγικής, οι προσομοιώσεις, τα αθλητικά παιχνίδια και τα παιχνίδια ανοικτού κόσμου (sandbox), είναι κατάλληλα για εφαρμογές ενισχυτικής μάθησης. Η χρήση της ενισχυτικής μάθησης στα βιντεοπαιχνίδια προσφέρει πολλά οφέλη, όπως η παραγωγή ευφυών πρακτόρων, εξατομικευμένες εμπειρίες, παραγωγή ποικίλου περιεχομένου και μειωμένη χειρωνακτική εργασία στην ανάπτυξη παιχνιδιών. Με την προσαρμογή του επιπέδου δυσκολίας, την παροχή εξατομικευμένων συμβουλών ή βοήθειας και την ανάπτυξη δυναμικών και ευέλικτων ρυθμίσεων παιχνιδιού, οι προσεγγίσεις ενισχυτικής μάθησης μπορούν να χρησιμοποιηθούν για να προσελκύσουν το ενδιαφέρον των παικτών. Οι δημιουργοί παιχνιδιών μπορούν να διευρύνουν τα όρια των διαδραστικών εμπειριών με τη χρήση αλγορίθμων ενισχυτικής μάθησης, παρέχοντας μοναδικό, ελκυστικό και δυναμικό παιχνίδι που συναρπάζει τους παίκτες.

Εντέλει, η ενισχυτική μάθηση έχει αναδειχθεί ως ένα ισχυρό εργαλείο για τη μεταμόρφωση της παραγωγής βιντεοπαιχνιδιών. Η ενσωμάτωση αλγορίθμων ενισχυτικής μάθησης επιτρέπει στους παραγωγούς παιχνιδιών να δημιουργούν έξυπνους NPCs, προσαρμοστικές εμπειρίες παιχνιδιού και διαδικαστική παραγωγή περιεχομένου, τα οποία βελτιώνουν τη δέσμευση και την ευτυχία των χρηστών. Με την ενισχυτική μάθηση, τα συστήματα παιχνιδιών μπορούν να τροποποιούν δυναμικά τα επίπεδα δυσκολίας και να προσαρμόζουν τις εμπειρίες, με αποτέλεσμα μοναδικό και συναρπαστικό gameplay. Η συνεχής εξισορρόπηση και δοκιμή που καθίσταται δυνατή με τη ενισχυτική μάθηση συμβάλλει σε μια καλύτερη εμπειρία παιχνιδιού. Συνολικά, η ενισχυτική μάθηση υπόσχεται πολλά για να διευρύνει τα όρια των διαδραστικών εμπειριών, προσφέροντας στους παίκτες ελκυστικό και δυναμικό παιχνίδι που τους κρατάει το ενδιαφέρον και τη διασκέδαση.

2.3 Εισαγωγή στον αλγόριθμο του Dijkstra

Ο αλγόριθμος αυτός επινοήθηκε και δημοσιεύθηκε από τον λαμπρό Ολλανδό επιστήμονα υπολογιστών και μηχανικό λογισμικού Dr. Edsger W. Dijkstra. Το 1959 δημοσίευσε ένα τρισέλιδο άρθρο με τίτλο "A note on two problems in connection with graphs" [6] στο οποίο περιέγραφε τον νέο του αλγόριθμο. Ο αλγόριθμος του Dijkstra, εφεξής αλγόριθμος του Dijkstra, είναι ένας πολύ απλός και άπληστος αλγόριθμος για τον εντοπισμό της διαδρομής με το χαμηλότερο κόστος από έναν αρχικό κόμβο σε έναν ή περισσότερους κόμβους ενός γράφου (Single-Source shortest-paths)[7]. Εν συντομία, ο αλγόριθμος του Dijkstra λειτουργεί εντοπίζοντας πρώτα τον κόμβο που βρίσκεται πλησιέστερα στην πηγή και στη συνέχεια εντοπίζει επαναληπτικά κόμβους που

βρίσκονται όλο και πιο μακριά. Στην πράξη, αυτό δημιουργεί ένα μονοπάτι με αφετηρία τον αρχικό κόμβο (κόμβος πηγής). Εάν αναζητούμε το μονοπάτι με το χαμηλότερο κόστος προς έναν μόνο κόμβο (κόμβος προορισμού), μπορούμε να σταματήσουμε όταν ο κόμβος αυτός εντοπιστεί. Εάν η αναζήτηση δεν τερματιστεί, θα υπολογιστεί ολόκληρος ο γράφος και οι αποστάσεις από τον κόμβο προέλευσης σε όλους τους άλλους κόμβους. Θα ξεκινήσουμε περιγράφοντας έναν αλγόριθμο που απλώς προσδιορίζει το μήκος του συντομότερου μονοπατιού από τον αρχικό κόμβο (ή τους αρχικούς κόμβους) σε οποιονδήποτε άλλο κόμβο του γραφήματος.

2.3.1 Περιγραφή

Το "εξερευνημένο" τμήμα του γραφήματος διατηρείται από τον αλγόριθμο ως ένα σύνολο (S) κορυφών (u) για τις οποίες έχει προσδιοριστεί η απόσταση της συντομότερης διαδρομής (d(u)) από τον κόμβο (s). Οι αρχικές συνθήκες είναι S = s και ds = 0. Τώρα, για κάθε κόμβο v ϵ V, V - S, προσδιορίζουμε το συντομότερο μονοπάτι που μπορεί να κατασκευαστεί διατρέχοντας μια διαδρομή μέσω του εξερευνημένου τμήματος S σε έναν κόμβο u ϵ S και στη συνέχεια ακολουθώντας την ακμή (u, v), η οποία είναι η συντομότερη διαδρομή. Με άλλους όρους, θεωρείται η ποσότητα d΄(v) = min $_{e=(u,v):\, u \in S}$ d(u) + l_e. Επιλέγουμε τον κόμβο v ϵ V - S όπου αυτή η ποσότητα είναι η μικρότερη, προσθέτουμε τον *v* στο *S* και ορίζουμε ως d(v) την τιμή του d΄(v). Είναι απλό να προκύψουν οι τροχιές s-u που αντιστοιχούν στις αποστάσεις που ανακαλύπτονται από τον αλγόριθμο του Dijkstra. Καθώς κάθε κόμβος *v* προστίθεται στο σύνολο S, η ακμή (u, v) λαμβάνεται με την τιμή *min* $_{e=(u,v):\, u \in S}$ d(u) + l_e. Το μονοπάτι P_v αναπαρίσταται έμμεσα από αυτές τις ακμές: αν (u, v) είναι η αποθηκευμένη ακμή για τον κόμβο v, τότε το P_v είναι απλώς το

μονοπάτι P_u που ακολουθείται από την ακμή (u, v). Για να κατασκευάσουμε το P_v, απλά ξεκινάμε από τον κόμβο v, ακολουθούμε την ακμή που έχουμε αποθηκεύσει για τον v προς την αντίθετη κατεύθυνση για να φτάσουμε στον κόμβο u, στη συνέχεια ακολουθούμε την ακμή που έχουμε αποθηκεύσει για τον u προς την αντίθετη κατεύθυνση για να φτάσουμε στον προκάτοχό του, και ούτω καθεξής μέχρι να φτάσουμε στον κόμβο s. Σημειώστε ότι πρέπει τελικά να φτάσουμε στον κόμβο s επειδή η αντίστροφη διαδρομή μας από τον κόμβο v έρχεται συνεχώς σε επαφή με κόμβους που έχουν προστεθεί στο S σε προηγούμενα στάδια. Παρακάτω μπορούμε να δούμε τον ψευδοκώδικα του Αλγόριθμου Dijkstra

Έστω S το σύνολο των εξερευνημένων κόμβων

For each $u \in S$, αποθηκεύουμε μια απόσταση $d(u)$

Αρχικά $S = \{s\}$ και $d(s) = 0$

While $S \not\equiv V$

Διάλεξε έναν κόμβο $v \notin S$ με τουλάχιστον μια ακμή από το S για τον οποίο ο

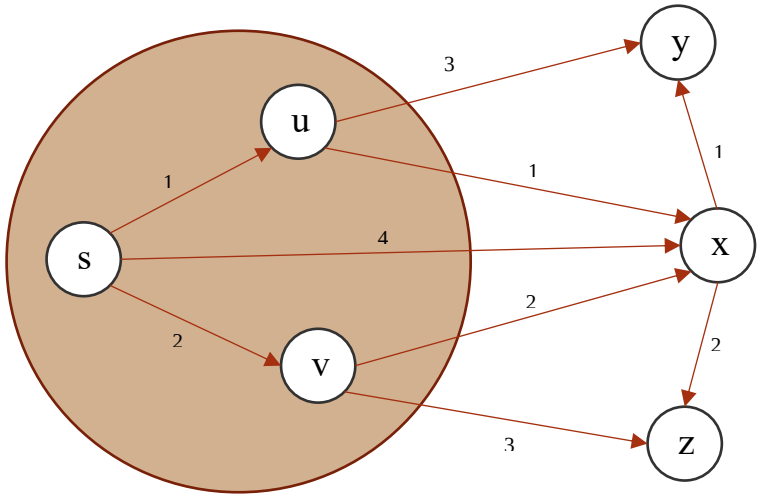
$d'(v) = \min_{e=(u,v): u \in S} d(u) + l_e$ είναι όσο το δυνατόν μικρότερο.

Πρόσθεσε τον v στο S και όρισε $d(v) = d'(v)$

End While

Δείτε το στιγμιότυπο (Εικόνα 6) εκτέλεσης που παρουσιάζεται παρακάτω για να κατανοήσετε καλύτερα τι κάνει ο αλγόριθμος. Στη θέση που απεικονίζεται στην εικόνα, έχουν ολοκληρωθεί δύο επαναλήψεις: η πρώτη πρόσθεσε τον κόμβο u και η δεύτερη τον κόμβο v. Στη θέση που

απεικονίζεται στην εικόνα, έχουν ολοκληρωθεί δύο επαναλήψεις: η πρώτη πρόσθεσε τον κόμβο u και η δεύτερη τον κόμβο v.



Εικόνα 6 Στιγμιοτυπο Εκτελεσης του Dijksta

Στην επόμενη επανάληψη, ο κόμβος x θα προστεθεί επειδή, λόγω της ακμής (u, x), αποκτά τη μικρότερη απόσταση d′(x) και d′(x) - d′(u) + l_{ux} - 2. Σημειώστε ότι η προσπάθεια προσθήκης των κόμβων y ή z στο σύνολο S αυτή τη στιγμή θα οδηγήσει σε εσφαλμένη τιμή για τις αποστάσεις των συντομότερων διαδρομών τους, ωστόσο, οι κόμβοι αυτοί θα προστεθούν τελικά λόγω των ακμών τους από τον κόμβο x. Είναι σημαντικό να επισημανθεί ότι ο αλγόριθμος του Dijkstra δεν ανακαλύπτει πάντα το συντομότερο μονοπάτι εάν υπάρχουν ακμές με αρνητικά βάρη. Ο αλγόριθμος εφαρμόζεται μόνο σε γραφήματα με θετικά βάρη. Αυτό οφείλεται στο γεγονός ότι τα βάρη των ακμών πρέπει να προστεθούν για τον προσδιορισμό του συντομότερου μονοπατιού. Εάν ο γράφος περιέχει αρνητικό βάρος, ο αλγόριθμος δεν θα λειτουργήσει σωστά. Μόλις ένας κόμβος χαρακτηριστεί ως "επισκέψιμος", το τρέχον μονοπάτι προς τον κόμβο αυτό σημειώνεται ως το

συντομότερο μονοπάτι για να φτάσει σε αυτόν. Και τα αρνητικά βάρη μπορούν να το αλλάξουν αυτό εάν το συνολικό βάρος μπορεί να μειωθεί μετά από αυτό το βήμα.

Κεφάλαιο 3 Σχεδιασμός της Συνάρτησης ανταμοιβής

3 Εισαγωγή

Οι πράκτορες ενισχυτικής μάθησης διδάσκονται να μεγιστοποιούν μια συνάρτηση ανταμοιβής (Reward Function ή RF). Είναι γνωστό ότι η επινόηση συναρτήσεων ανταμοιβής μπορεί να είναι δύσκολη[39, 38] και ο πράκτορας μπορεί να ανακαλύψει απρόβλεπτες και ανεπιθύμητες συντομεύσεις για την απόκτηση ανταμοιβών (reward), γεγονός που καθιστά αναγκαία την προσαρμογή της συνάρτησης ανταμοιβής στην εκάστοτε εργασία. Αυτό το ζήτημα μπορεί να είναι αρκετά σοβαρό ώστε να ακυρώσει όλες τις συναρτήσεις ανταμοιβής [40]. Ο Murphy [38] παρέχει πληροφορίες ενός πράκτορα που μαθαίνει να παίζει Tetris επ' αόριστον για να αποφύγει την ήττα. Εκτός από την οριοθέτηση του τι είναι αποδεκτή συμπεριφορά του πράκτορα, μπορεί να υπάρχουν απαιτήσεις φυσικής ασφάλειας ή περιορισμών κατά τη διάρκεια της εκπαίδευσης [41] ο πράκτορας δεν πρέπει να βλάψει το περιβάλλον του ή να "σπάσει" το παιχνίδι. Η συμπεριφορά του πράκτορα θα είναι ανάλογη με τις επιλογές που ανταμείβονται. Αυτό σημαίνει ότι αν παραλείψουμε βασικούς κανόνες, είναι πιθανό να προκύψει μια λύση που θα εκμεταλλεύεται ακατάλληλα σχεδιασμένους περιορισμούς ή θα παραβιάζει τους περιορισμούς προκειμένου να μεγιστοποιήσει τη συνολική ανταμοιβή. Οι έννοιες AI safety και Reward Hacking, σε συνδυασμό με την πολυπλοκότητα του συγκεκριμένου προβλήματος, είναι οι κύριοι λόγοι για τους οποίους ο σχεδιασμός ενός RF είναι τόσο δύσκολος. Όπως αναφέρθηκε στο σημείο Κ.2.2, στην RL επιθυμούμε να εκπαιδεύσουμε έναν πράκτορα για να επιλύσει ορισμένα προβλήματα. Δηλαδή, να ανακαλύψει μια λύση που μεγιστοποιεί τη συνολική ανταμοιβή του. Η επιλογή των Συναρτήσεων Ανταμοιβής είναι πιθανώς η πιο κρίσιμη απόφαση κατά την προσπάθεια επίλυσης προβλημάτων

RL. Παρακάτω, θα συζητήσουμε την έννοια της διαμόρφωσης ανταμοιβής, πώς μπορεί να βοηθήσει στη μείωση του χρόνου εκπαίδευσης, στον προσδιορισμό της βέλτιστης πολιτικής και σε τυχόν μειονεκτήματα που σχετίζονται με αυτή την τεχνική. Η RL είναι αποτελεσματική όταν είναι απλό να προσδιοριστεί η επιτυχία, αλλά δύσκολο να προσδιοριστεί ο τρόπος επίτευξής της. Για παράδειγμα, αν θέλουμε ένα αυτοκίνητο να ταξιδέψει από μια πόλη σε μια άλλη, είναι απλό να προσδιορίσουμε πότε το αυτοκίνητο έχει φτάσει στον προορισμό του, αλλά πολύ πιο δύσκολο να προσδιοριστεί πόσο συχνά πρέπει να γυρίσει το τιμόνι και να πατήσει το γκάζι. Ο πράκτορας θα αναπτύξει μια συμπεριφορά ανάλογη με τις επιλογές για τις οποίες ανταμείβεται, τίποτα περισσότερο τίποτα λιγότερο. Αυτό σημαίνει ότι αν αγνοήσουμε σημαντικούς κανόνες, όπως στο παράδειγμα με το αυτοκίνητο, όπου ένας σημαντικός κανόνας θα ήταν να κρατήσουμε τους επιβάτες ζωντανούς, μπορεί να ακολουθήσει τη συντομότερη διαδρομή προς τον προορισμό οδηγώντας κατευθείαν σε έναν γκρεμό. Επομένως, είναι σημαντικό να ενσωματώσουμε όλες τις σχετικές πληροφορίες στη Συνάρτηση Ανταμοιβής και να διασφαλίσουμε ότι αντιπροσωπεύει με ακρίβεια την επιθυμητή συμπεριφορά.

3.1 Συνάρτηση Ανταμοιβής (Reward Function)

Η συνάρτηση ανταμοιβής (RF) είναι μια χαρτογράφηση από μια κατάσταση σε μια απόφαση με μια κλιμακωτή τιμή ανταμοιβής (reward). Ο πράκτορας θα αναπτύξει μια πολιτική με βάση την RF με την οποία του δόθηκαν οδηγίες. Μια ακολουθία κατάστασης, δράσης, ανταμοιβής και νέας κατάστασης το ονομάζουμε επεισόδιο. Κάθε φορά που ο πράκτορας εκτελεί μια ενέργεια, θα φτάνει σε μια νέα κατάσταση και θα λαμβάνει μια ανταμοιβή, την οποία αναφέρουμε ως δείγμα.

Το επεισόδιο ολοκληρώνεται όταν ενεργοποιηθεί μια τερματική κατάσταση και τότε το περιβάλλον επανέρχεται στην αρχική του κατάσταση.

3.1.1 Διαμόρφωση των ανταμοιβών

Η διαμόρφωση των ανταμοιβών (reward shaping) είναι η διαδικασία τροποποίησης τις RF ώστε να παρέχει στους πράκτορες καθοδήγηση ή "ενδείξεις" για να τους βοηθήσει να μάθουν πιο γρήγορα. Είναι μια αρκετά αποτελεσματική τεχνική κλιμάκωσης και διαχείρισης για πολύπλοκα RL προβλήματα [29,66,86]. Ωστόσο, ορισμένες φυσικές επιλογές για τη διαμόρφωση των ανταμοιβών μπορεί να οδηγήσουν σε ανεπαρκείς λύσεις. Η προκλητική πτυχή της διαμόρφωσης ανταμοιβών είναι ότι με την τροποποίηση της αρχικής RF, το αρχικό πρόβλημα M μετατρέπεται σε ένα νέο πρόβλημα M'[37] και ο αλγόριθμος RL καλείται να λύσει το M' με την ελπίδα ότι η λύση μπορεί να είναι απλούστερη ή να εντοπίζεται ταχύτερα από το αρχικό πρόβλημα. Δεν είναι πάντα προφανές αν οι λύσεις που ανακαλύπτονται για το τροποποιημένο πρόβλημα M' είναι εφαρμόσιμες στο αρχικό πρόβλημα M. Ιδεαλιστικά, θέλουμε οι βέλτιστες λύσεις να είναι αμετάβλητες σε αλλαγές στην RF, έτσι ώστε οι καλές πολιτικές που χρησιμοποιούνται από τις ανταμοιβές διαμόρφωσης να είναι αποτελεσματικές και για το αρχικό πρόβλημα. Οι Randlov και Alstrom [86] περιγράφουν ένα σύστημα που μαθαίνει να οδηγεί ένα ποδήλατο σε μια συγκεκριμένη τοποθεσία. Για να επιταχύνουν την εκπαίδευση, έδωσαν στον πράκτορα θετικές ενισχύσεις καθώς πλησίαζε το στόχο. Επειδή δεν υπήρχε ποινή όταν ο πράκτορας απομακρυνόταν από το στόχο, έμαθε να κάνει ποδήλατο σε κύκλους γύρων από την αρχική του κατάσταση. Αυτό υποδηλώνει ότι οι διαμορφωμένες ανταμοιβές πρέπει να τηρούν ορισμένες προϋποθέσεις για να

αποφευχθεί η χειραγώγηση του πράκτορα ώστε να αναπτύξει λανθασμένες πολιτικές, όπως η κυκλική συμπεριφορά που παρατηρήθηκε στην προηγούμενη ενότητα (στην οποία ο πράκτορας επαναλάμβανε κυκλικά μια σειρά καταστάσεων για να αυξήσει τη συνολική ανταμοιβή).

3.2 Είδη Διαμορφωμένων Συναρτήσεων Ανταμοιβής

Θα προσπαθήσουμε να μάθουμε μια πολιτική και αναμένουμε να βοηθήσουμε τον αλγόριθμο ενισχυτικής μάθησης παρέχοντάς του πρόσθετες "ανταμοιβές διαμόρφωσης" που πιστεύουμε ότι θα τον βοηθήσουν να αναπτύξει μια καλή βέλτιστη πολιτική πιο γρήγορα. Υποθέτουμε ότι αντί να εκτελούμε τον αλγόριθμο RL στο πρόβλημα M = (State, Action, Reward), τον εκτελούμε σε κάποιο τροποποιημένο πρόβλημα M' = (State, Action, Reward'), όπου R' – R + F είναι το RF του τροποποιημένου M και F: S x A x S > R είναι μια συνάρτηση οριοθέτησης γνωστή ως συνάρτηση ανταμοιβής διαμόρφωσης. Έτσι, στο αρχικό πρόβλημα M, θα μπορούσαμε να λάβουμε R(s, a, s') από τη μετάβαση s σε s' με την ενέργεια a, και στο νέο πρόβλημα M', θα μπορούσαμε να λάβουμε R(s, a, s') + F(s, a, s') στην ίδια κατάσταση.

Έτσι, για να παρακινηθεί ο πράκτορας να επιδιώξει τον στόχο, μια συνάρτηση διαμόρφωσης ανταμοιβής μπορεί να είναι (a για action, s για state, r για rewards):

F (s, a, s') = r, για όποτε το s' είναι πιο κοντά στον στόχο από τον s, (F1)

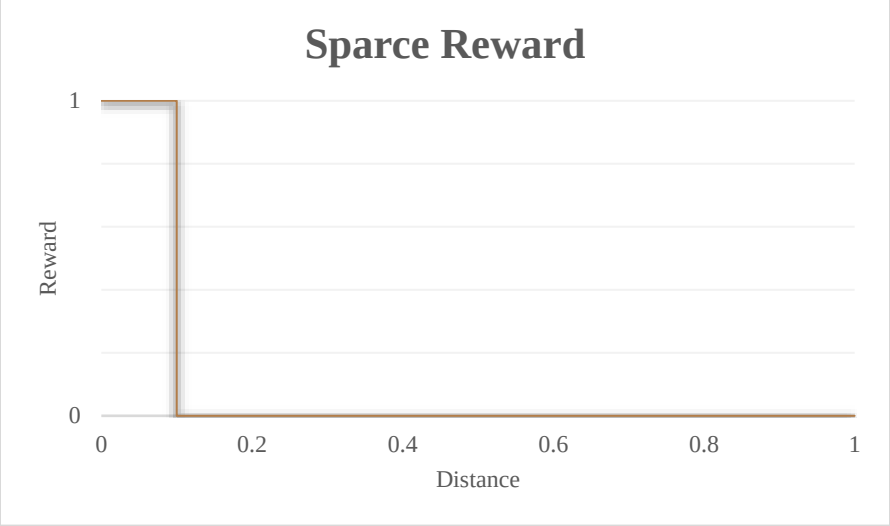
F (s, a, s') = -r || 0, για όποτε το s' είναι πιο μακριά από τον στόχο από τον s ή για s' = S_0,

Και για να τον ενθαρρύνουμε να πάρει actions a_1 σε κάποιο σετ από states S_0:

$$F(s, a, s') = r, \text{ για όποτε } a = a_1, s \in S_0, \qquad (F2)$$

$$F(s, a, s') = 0 \text{ διαφορετικά.}$$

Η κλιμάκωση είναι μια σχετικά απλή μέθοδος για τη μείωση της διάρκειας της εκπαίδευσης. Μπορούμε να χωρίσουμε τις ανταμοιβές σε δύο κατηγορίες: τις προαναφερθείσες σπάνιες ανταμοιβές(sparce rewards) και τις διαμορφούμενες ανταμοιβές (shaped rewards). Η σπάνια ανταμοιβή είναι η πιο βασική ανταμοιβή που μπορούμε να παρέχουμε.

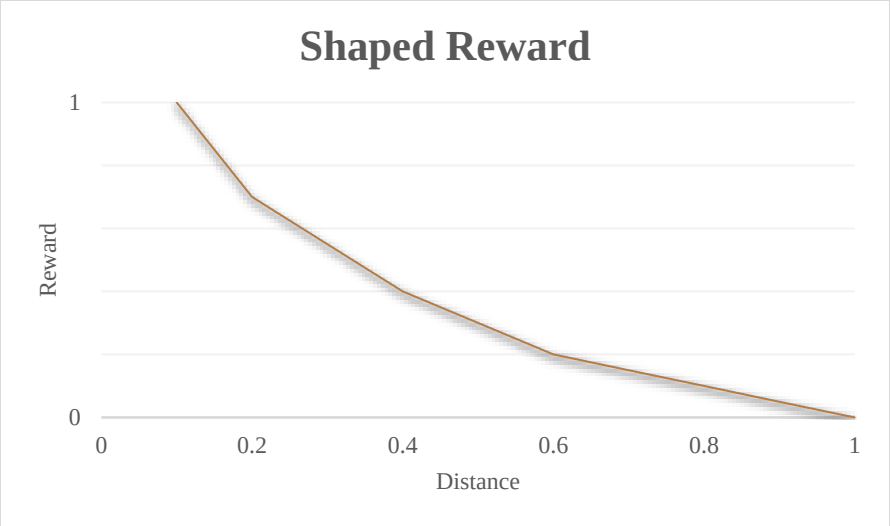


Σχήμα 1 Sparce Reward

Ο πράκτορας λαμβάνει μια ανταμοιβή αν ολοκληρώσει αποτελεσματικά τη δοκιμή ή αν ολοκληρώσει συγκεκριμένες ενέργειες, διαφορετικά δεν λαμβάνει τίποτα. Αυτές οι ανταμοιβές είναι απλές στη σύνταξη, αλλά έχουν μακρά περίοδο εκπαίδευσης λόγο της απώλειας πληροφοριών κατά τη διάρκεια της εκπαίδευσης. Στην αραιή/ σπάνια συνάρτηση (Σχήμα 1)

υπάρχει μια συνάρτηση απόστασης-ανταμοιβής. Όταν η απόσταση είναι μικρότερη ή ίση με 0.1, ο πράκτορας λαμβάνει ανταμοιβή ίση με 1, διαφορετικά, λαμβάνει ανταμοιβή με τιμή 0.

Η εξερεύνησή του δεν έχει προκαθορισμένο προσανατολισμό, οπότε πρέπει να εξερευνήσει αυθαίρετα το περιβάλλον μέχρι να συναντήσει μια συνθήκη επιτυχίας, μετά την οποία είναι ικανός να μάθει την ακολουθία ενεργειών που οδήγησε σε αυτή τη συνθήκη. Η παράλειψη ουσιωδών πληροφοριών, όπως η κατεύθυνση προς τον στόχο, μπορεί να δυσχεράνει την έγκαιρη εύρεση κατάλληλου συμβολαίου από τον πράκτορα. Κωδικοποιώντας πληροφορίες σχετικά με το έργο/εργασία, μπορούμε να επιταχύνουμε τη διαδικασία. Για παράδειγμα, αν γνωρίζουμε ότι το να είμαστε κοντά στον στόχο είναι προτιμότερο από το να είμαστε μακριά, μπορούμε να χρησιμοποιήσουμε τη συνάρτηση διαμόρφωσης (RF) για να μεταφέρουμε αυτή την πληροφορία στον πράκτορα. Όταν χρησιμοποιούμε τη διαμόρφωση ανταμοιβής, περνάμε μια ομαλή, συνεχή κλίση που καθορίζει, για κάθε δεδομένη κατάσταση, ότι αν ο πράκτορας βρίσκεται πιο κοντά στο στόχο, θα πρέπει να διακρίνει μεγαλύτερη ανταμοιβή από ό,τι αν ήταν πιο μακριά. Παρατηρούμε μια διαμορφωμένη συνάρτηση ανταμοιβής στην Σχήμα 2. Παρατηρούμε μια ομοιόμορφη κλίση στον άξονα "απόσταση", που υποδηλώνει ότι ο πράκτορας λαμβάνει μεγαλύτερες ανταμοιβές όσο πιο κοντά βρίσκεται στον στόχο. Αυτό το αποτέλεσμα είναι στραμμένο προς την κατάσταση που είναι πιο κοντά στον στόχο και θα βοηθήσει τον πράκτορα να καταλήξει σε μια σωστή πολιτική πιο γρήγορα. Αυτή η συνάρτηση ανταμοιβής που διαμορφώνει την απόσταση αυξάνει την τιμή της ανταμοιβής καθώς πλησιάζει ο στόχος. Έτσι, ενθαρρύνει τον πράκτορα να πλησιάσει όσο το δυνατόν γρηγορότερα και παρέχει επαρκείς πληροφορίες σχετικά με τη θέση του στόχου.



Σχήμα 2 Shaped Reward

3.3 Τερματικές Συνθήκες

Οι Τερματικές Συνθήκες (Terminal Conditions ή TC) είναι οι αρχικοποιημένες προϋποθέσεις στις οποίες το περιβάλλον επιστρέφει στην αρχική του κατάσταση και αρχίζει ένα νέο επεισόδιο, επιτρέποντας στον πράκτορα να επιχειρήσει να ξανά λύση το πρόβλημα από την αρχή. Οι τερματικές συνθήκες βοηθούν στον περιορισμό της εξερεύνησης για την αποφυγή απώλειας χρόνου. Αν ο πράκτορας κολλήσει ή αν φτάσει σε μια κατάσταση όπου η εργασία δεν μπορεί να ολοκληρωθεί (για παράδειγμα, αν το αυτοκίνητο πέσει από έναν γκρεμό, μπορεί να μην θέλουμε να εξερευνήσουμε τις συνέπειες της πτώσης), θέλουμε να τερματίσουμε το επεισόδιο και να επαναφέρουμε το περιβάλλον στην αρχική του κατάσταση, ώστε ο πράκτορας να εξερευνήσει τις πιο σημαντικές περιοχές του χώρου.

Μπορούμε να χωρίσουμε τις TC σε τρεις κατηγορίες:

1. χρονικά όρια
2. θετικές τερματικές συνθήκες
3. αρνητικές τερματικές συνθήκες

Συχνά, τα χρονικά όρια χρησιμοποιούνται για την παράκαμψη της συνθήκης αδράνειας. Εάν ο πράκτορας φτάσει σε ένα σημείο όπου δεν μπορεί να λύσει το πρόβλημα, είναι καλύτερο να τερματίσει το επεισόδιο και να το ξαναρχίσει από την αρχή, ώστε να μπορέσει να προβάρει εκ νέου τις αρχικές συνθήκες. Είναι σημαντικό ότι τα χρονικά όρια περιορίζουν τις δαπάνες εκπαίδευσης που κατανέμονται σε κάθε επεισόδιο και παρέχουν στον πράκτορα περισσότερη εμπειρία στα μονοπάτια που απέδωσαν την υψηλότερη τιμή ανταμοιβής μεταξύ των αρχικών συνθηκών και οποιασδήποτε τελικής συνθήκης.

Στην συνέχεια, έχουμε τις θετικές τερματικές συνθήκες. Για παράδειγμα, αν ο πράκτορας φτάσει επιτυχώς στον προορισμό και το έχουμε ως επιτυχία υπό ορισμένους περιορισμούς/κανόνες, αυτό είναι μια θετική τελική συνθήκη. Όταν λαμβάνει μια τέτοια συνθήκη, συνήθως υποθέτουμε ότι ο πράκτορας ολοκλήρωσε επιτυχώς την εργασία (ή ένα μέρος της) και κάνουμε επαναφορά του επεισοδίου ώστε να συνέχιση να βελτιώνεται.

Οι αρνητικές τερματικές συνθήκες υποδηλώνουν την αποτυχία του πράκτορα να ολοκληρώσει το επεισόδιο. Για παράδειγμα, εάν το αυτοκίνητο έχει πέσει από έναν γκρεμό ή ο πράκτορας έχει

συγκρουστεί με κάτι που δεν θέλουμε ή έχει φτάσει σε μια κατάσταση στην οποία γνωρίζουμε ότι δεν μπορεί να συνέχιση επιτυχώς τη συγκεκριμένη εργασία, θέλουμε να επαναφέρουμε το επεισόδιο στις αρχικές συνθήκες και να του δώσουμε την ευκαιρία να προσπαθήσει ξανά. Ωστόσο, δίνουμε μια μεγάλη τιμωρία (punishment), ώστε ο πράκτορας να μάθει να αποφεύγει αυτές τις ενέργειες σε μελλοντικά επεισόδια.

Η αλληλεπίδραση του πράκτορα με τα θετικά rewards και τα αρνητικά reward, καθώς και ο τρόπος τερματισμού του περιβάλλοντος, είναι ένας κρίσιμος παράγοντας που πρέπει να ληφθεί υπόψη κατά το σχεδιασμό ενός RF, λαμβάνοντας υπόψη τις συνθήκες που αναφέρθηκαν παραπάνω. Με βασικούς όρους, οι TC (τερματικές καταστάσεις) καθοδηγούν τον πράκτορα περιορίζοντας τις επιλογές του και ενθαρρύνοντάς τον να συνεχίσει να επιδιώκει τη μεγιστοποίηση της συνολικής ανταμοιβής του. Αυτό μπορεί, ωστόσο, να έχει απρόβλεπτες ή παράλογες επιπτώσεις στις εκπαιδεύσεις. Για να διευκρινίσουμε, αν η ανταμοιβή του πράκτορα είναι ανάλογη με το πόσο κοντά βρίσκεται στο στόχο (Πίνακας 3.2), ο πράκτορας θα αναπτύξει μια ιδιότυπη στρατηγική κατά την οποία θα πλησιάσει αρκετά το στόχο και στη συνέχεια θα μείνει ακίνητος ή θα κάνει κύκλους γύρω του. Δηλαδή, ο πράκτορας βρίσκει τρόπους να εκμεταλλευτεί το RF προς όφελός του, κάτι που είναι γνωστό ως reward hacking. Εναλλακτικά, η απλή πρόσβαση στο RF σε ένα πολύπλοκο πρόβλημα μπορεί να οδηγήσει τον πράκτορα στο να τερματίσει το επεισόδιο το συντομότερο δυνατό, αντί να προσπαθήσει να λύσει το πρόβλημα προκειμένου να λάβει τη μέγιστη ανταμοιβή. Αυτό μπορεί να οφείλεται στο κίνητρο του πράκτορα να συγκεντρώσει τη μέγιστη ανταμοιβή μέχρι το τέλος του χρόνου του επεισοδίου που έχει οριστεί στην αρχή. Η χρήση ενός συνόλου τερματικών συνθηκών θα αποτελούσε λύση σε αυτό το ζήτημα. Με την επαλήθευση

ότι κάθε θετική τερματική συνθήκη αποδίδει μια αρκετά μεγαλύτερη ανταμοιβή για κάθε single-step που περνάει. Ο πράκτορας, με αυτόν τον τρόπο, θα τα αναζητήσει τη λύση του προβλήματος πιο γρήγορα αντί να προσπαθήσει να τα αποφύγει, εξαλείφοντας έτσι τη χειραγώγηση της ανταμοιβής (reward hacking).

3.4 Σχεδιασμός Ειδικής Διαμορφωμένης Συνάρτηση Ανταμοιβής

Σε αυτή την ενότητα, θα υλοποιήσουμε μια προσαρμοσμένη συνάρτηση ανταμοιβής και θα εφαρμόσουμε τις έννοιες από τις ενότητες 3.2 και 3.3. Η συγκεκριμένη εργασία που πρέπει να εκτελέσει ο πράκτορας είναι ένα πρόβλημα εύρεσης βέλτιστου μονοπατιού. Το επιθυμητό αποτέλεσμα είναι ο πράκτορας να εντοπίσει τον στόχο και να ολοκληρώσει την αποστολή διανύοντας όσο το δυνατόν λιγότερη απόσταση.

Με βάση αυτό το πρόβλημα, μπορούμε να αρχίσουμε να σχεδιάζουμε μια συνάρτηση ανταμοιβής (RF) με βάση την απόσταση μεταξύ του πράκτορα και του στόχου. Αυτή θα μπορούσε να είναι μια απλή διαμορφωμένη συνάρτηση(K3.2)(F1)(Πίνακας 3.2) της απόστασης, έτσι ώστε όσο πιο κοντά βρίσκεται ο πράκτορας στο στόχο, τόσο μεγαλύτερη είναι η ανταμοιβή του και τόσο πιο απότομη είναι η κλίση επιτυχίας του στόχου. Σε συνέχεια από το Πίνακας 3.2, λαμβάνουμε την τρέχουσα απόσταση (Dx), την οποία τη διαιρούμαι με την ελάχιστη απόσταση πράκτορα-προορισμού (P), που μας δίνετε από τον αλγόριθμο του Dijkstra. Το αποτέλεσμα το βάζουμε σε δύναμη (e) με τιμή μικρότερη του 1 με σκοπό να παρέχει μια πιο απότομη κλίση. Στη συνέχεια, αφαιρούμε την τιμή με 1 για να τη διατηρήσουμε θετική και εντός του επιτρεπόμενου εύρους,

επιτρέποντάς μας να τοποθετήσουμε αρνητικές τερματικές συνθήκες χωρίς πρόβλημα. Η κλίση θα είναι πιο απότομη καθώς ο πράκτορας πλησιάζει το στόχο, οπότε μπορούμε να υποθέσουμε ότι ο πράκτορας θα έχει επαρκή κίνητρα/ενδιαφέρον για να πλησιάσει το στόχο και ότι η τιμή της ανταμοιβής δεν θα υπερβαίνει ποτέ το 1 (για αριθμητικούς λόγους, είναι συχνά προτιμότερο να θέτουμε την τιμή της ανταμοιβής μεταξύ -1 και 1).

Η συνάρτηση που προκύπτει έχει την εξής μορφή:

- Η Dx είναι η απόσταση του πράκτορα από τον στόχο,
- Η P είναι η απόσταση του συντομότερου μονοπατιού από τον αλγόριθμο του Dijkstra,
- Το e είναι το πόσο απότομη θα είναι η κλίση της καμπύλης ανταμοιβής,
- Το r αντιπροσωπεύει την αμοιβή για κάθε φάση, $r = 1 – (Dx / P)^{e}$

Αν και φαίνεται αρκετά αποτελεσματικό με μια πρώτη ματιά, υπάρχουν ορισμένα ζητήματα που μπορεί να οδηγήσουν στη δημιουργία παράλογων πολιτικών. Τις αναφέρθηκε στην ενότητα 3.3, ο πράκτορας προσπαθεί να μεγιστοποιήσει τη συνολική ανταμοιβή. Επομένως, ένα ελαττωματικό RF μπορεί να οδηγήσει σε reward hacking. Μέχρι στιγμής, έχουμε περιγράψει στον πράκτορα ότι όσο μεγαλύτερη είναι η ανταμοιβή τόσο πιο κοντά στον στόχο βρίσκεται, αλλά δεν του έχουμε δώσει επαρκείς πληροφορίες σχετικά με τον στόχο ή τις τερματικές καταστάσεις. Ως αποτέλεσμα, ο πράκτορας θα κατάληξη να κάνει κύκλο γύρω από τον στόχο σε μια προσπάθεια να

μεγιστοποιήσει την ανταμοιβή του ανά επεισόδιο. Αυτό οφείλετε επειδή δεν έχουμε καθορίσει τι πρέπει να κάνει για να τερματίσει αποτελεσματικά το επεισόδιο.

3.6 Συνθέτη Συνάρτηση Ανταμοιβής

Τις είδαμε στην προηγούμενη παράγραφο σχετικά με το πρόβλημα τις single-step ανταμοιβής, ένα πιο σύνθετο και ακριβές RF θα μπορούσε να έχει μια συγκεκριμένη Τερματική Κατάσταση για κάθε κατάσταση στην οποία ο πράκτορας δεν χρειάζεται να συνεχίσει την εξερεύνηση ώστε να αποτρέψει την ανάπτυξη τις λανθασμένης πολιτικής. Άρα, πρέπει να επινοήσουμε μια συνάρτηση που να λαμβάνει υπόψη τις σχετικές πληροφορίες που πρέπει να γνωρίζει ο πράκτορας προκειμένου να διαμορφώσει μια βέλτιστη πολιτική και να μειώσει το χρόνο εκπαίδευσης. Πρώτον, μπορούμε να δώσουμε οδηγίες στον πράκτορα σχετικά με τις "καταναλώσεις". Δηλαδή, μια τιμή που θα πολλαπλασιάζεται με κάθε ανταμοιβή για κάθε βήμα που περνάει, υποδεικνύοντας ότι ο πράκτορας θα λαμβάνει μεγαλύτερη ανταμοιβή όσο λιγότερα βήματα κάνει. Τις αναφέρθηκε στην ενότητα Κ 3.3, είναι απαραίτητο να αποτρέψουμε τον πράκτορα από το να βρεθεί σε καταστάσεις όπου δεν μπορεί να ολοκληρώσει το επεισόδιο. Στη συγκεκριμένη ανάθεση, οι καταστάσεις αυτές είναι οι εξής:

- Ο πράκτορας καθηλώνεται σε έναν τοίχο, ενώ ο στόχος βρίσκεται ακριβώς πίσω του,
- Δεν ολοκληρώνει το επεισόδιο και δεν έχει στόχο PR,
- Εντοπίζοντας τον πρώτο στόχο αλλά όχι τον δεύτερο,
- Βρίσκει όλους τις στόχους, αλλά Dx είναι μεγαλύτερο από το P,

Η συγκεκριμένη αποστολή που θα προσπαθήσει να επιλύσει ο πράκτορας μπορεί να καθοριστεί από τον αρχικό κόμβο, τον τελικό κόμβο, τη διανυσματική απόσταση του πράκτορα (Dx), την ελάχιστη απόσταση του αλγορίθμου του Dijkstra(P) και τις συνθήκες τερματισμού (πχ. Όταν έρχεται σε επαφή με τοίχους). Είναι πιο ακριβές να «πειράξουμε» το RF, δηλαδή να το σπάσουμε σε μικροσκοπικά μέρη και να στοχεύσουμε το RF σε κάθε θετική τερματική συνθήκη. Για παράδειγμα, είναι προτιμότερο να εντοπίσουμε τον αρχικό κόμβο, να σταματήσουμε το επεισόδιο όταν ο πράκτορας εξακολουθεί να μην κατανοεί ολόκληρο το πρόβλημα και στη συνέχεια να αυξήσουμε τα όρια εκπαίδευσης. Παρακάτω δίνεται ο ψευδοκώδικας τις RF.

START

Αρχικά :
e ϵ [0.1,1], η κλίση τον rewards
s ϵ [1,0], τα steps που έχει εκτελέσει ο agent
L ϵ [0,1], η διαφορά απόστασης agent-στόχου

While δεν έχει τελειώσει το επεισόδιο **OR δεν** έχει ενεργοποιηθεί κάποια Τερματική Κατάσταση

s = ενημερώνουμε την τιμή τον steps που έχει εκτελέσει ο agent

If το επιδίδει **ΔΕΝ** έχει τελειώσει
Then υπολόγιζε το reward απόστασης reward = (1 – τωρινή απόσταση / L)^e * s
Return το reward.

If Βρήκε τον Πρώτο Στόχο **Then** reward = 0.25
If Βρήκε τον Τελικό Στόχο **Then** reward = 0.5
If Η διανυσματική απόσταση είναι ίση με την εκτιμώμενη
Then reward = 1 **Return** reward
Else Return reward = -0.25
Else Return reward = -0.5
Else Return reward = -1

End While
END

Κεφάλαιο 4 Εργαλεία και προσέγγιση

4.1 Εισαγωγή

Τα κύρια εργαλεία που θα χρησιμοποιηθούν σε αυτήν την πτυχιακή εργασία είναι η μηχανή δημιουργίας βιντεοπαιχνιδιών της Unity και το ML-Agents Toolki. Το ML-Agents Toolkit, είναι ένα πακέτο ανοικτού κώδικα που περιλαμβάνει έτοιμα προς χρήση εργαλεία, συμπεριλαμβανομένων αλγορίθμων ενισχυτικής μάθησης, Python APIs, Communicators και παραδειγμάτων RL στο τρισδιάστατο περιβάλλον τις Unity. Αυτό το πακέτο μπορεί να ενσωματωθεί εύκολα και γρήγορα σε οποιοδήποτε Unity Project που πληρεί τις προδιαγραφές. Έχει σχεδιαστεί για την κατασκευή σύνθετων περιβαλλόντων, για την έρευνα αλγορίθμων τελευταίας τεχνολογίας, για τη δοκιμή αυτών των αλγορίθμων σε νέα περιβάλλοντα και για την απλούστευση της υλοποίησης πρακτόρων με πολλαπλές λειτουργίες από τους προγραμματιστές παιχνιδιών. Επιπλέων, θα χρησιμοποιηθούν οι γλώσσες προγραμματισμού C# και Python, καθώς και βιβλιοθήκες όπως το TensorBoard για την οπτικοποίηση και την αποσφαλμάτωση των μοντέλων.

Όλοι οι εκπαίδευση του πράκτορα, πραγματοποιήθηκε σε υπολογιστή με τα εξής χαρακτηρίστηκα:

CPU	Ryzen 7 5800H
GPU	NVIDIA GeForce RTX 3060
RAM	12 GB

4.2 Απαιτήσεις

Για τη συγκεκριμένη εργασία, θα πρέπει να εγκαταστήσουμε συγκεκριμένα εργαλεία. Συγκεκριμένα, πρέπει να εγκαταστήσουμε την Python 3.3.3 μαζί με τις βιβλιοθήκες PyTorce και PyNum. Στη συνέχεια, τη μηχανή ανάπτυξης παιχνιδιών Unity με έκδοση 2020.x ή μεταγενέστερη. Τέλος, το πακέτο ML-Agent, το οποίο είναι το API για τη σύνδεση της μηχανής Unity, με την Python όπου εκτελούνται οι αλγόριθμοι RL.

4.3 RL Αλγόριθμοι και ML Agents Toolkit

Σε αυτή την ενότητα, θα συζητήσουμε τους αλγορίθμους μηχανικής μάθησης που ενσωματώνονται στο ML-Agent Tool Kit. Θα συζητήσουμε τα σήματα ανταμοιβής (reward signals) και τις εσωτερικές και εξωτερικές ανταμοιβές(intrinsic και extrinsic rewards), οι οποίες θα μας βοηθήσουν να διαφωτίσουμε ορισμένες από τις τεχνικές εκπαιδεύσεις. Στην RL, ο απώτερος στόχος του πράκτορα είναι να εφαρμόσει μια συμπεριφορά (πολιτική) που μεγιστοποιεί την ανταμοιβή που λαμβάνει. Κατά τη διάρκεια της εκτέλεσής της, θα πρέπει να παρέχονται στον πράκτορα ένα ή περισσότερα σήματα ανταμοιβής. Με την ολοκλήρωση μιας αποστολής ή κάποιου στόχου, ο πράκτορας λαμβάνει κάποια ανάλογη ανταμοιβή και καθορίζονται από το περιβάλλον. Αυτές οι ανταμοιβές ονομάζονται εξωγενείς ανταμοιβές (extrinsic rewards) επειδή ο αλγόριθμος μάθησης τις καθορίζει εξωτερικά. Οι εσωτερικές ανταμοιβές (intrinsic rewards) μπορούν να καθοριστούν για να ενθαρρύνουν έναν πράκτορα να εκτελέσει με έναν συγκεκριμένο τρόπο ή για

να του διδάξουν την πραγματική εξωγενή ανταμοιβή(extrinsic rewards). Η συνολική ανταμοιβή μπορεί να είναι ένας συνδυασμός εξωτερικών και εσωτερικών σημάτων ανταμοιβής.

Το ML-Agent Tool Kit επιτρέπει τον ορισμό των σημάτων ανταμοιβής (reward signals) και μας παρέχει τέσσερα reward signals που μπορούν να αναμειχθούν και να συνδυαστούν για να επηρεάσουν τη συμπεριφορά του πράκτορά μας. Εν κατακλείδι, αυτά τα τέσσερα reward signals είναι:

- Curiosity Signal,
- RNG Signal,
- Gail Signal,
- BC Signal,

4.3.1 Curiosity Signal

Σε περιβάλλοντα όπου ο πράκτορας λαμβάνει λίγες ή σπάνιες ανταμοιβές (sparse rewards), ένας πράκτορας μπορεί να μην λάβει ποτέ ένα σήμα ανταμοιβής για να ξεκινήσει την εκπαίδευση. Αυτή είναι μια κατάσταση στην οποία η χρήση εγγενών σημάτων ανταμοιβής (intrinsic signal rewards) μπορεί να είναι επωφελής. Το curiosity signal είναι ένα από τα σήματα που μπορεί να βοηθήσει τον πράκτορα στην εξερεύνηση του περιβάλλοντος όταν οι εξωτερικές ανταμοιβές είναι σπάνιες. Η μονάδα ενδογενής περιέργεια ενεργοποιείται από το σήμα ανταμοιβής περιέργειας. Πρόκειται για μια εφαρμογή της μεθόδου που περιγράφεται στο Pathak et al.'s Curiosity-driven Exploration by Self-supervised Prediction[2]. Εκπαιδεύει δύο δίκτυα: ένα αντίστροφο μοντέλο που

κωδικοποιεί την τρέχουσα και την επόμενη παρατήρηση του πράκτορα και χρησιμοποιεί την κωδικοποίηση για να προβλέψει τη δράση που θα πραγματοποιηθεί μεταξύ των παρατηρήσεων, και ένα μπροστινό μοντέλο που κωδικοποιεί την τρέχουσα παρατήρηση και δράση και προβλέπει την επόμενη κωδικοποιημένη παρατήρηση. Η απώλεια του εμπρόσθιου μοντέλου (η διαφορά μεταξύ των προβλεπόμενων και των πραγματικών κωδικοποιημένων παρατηρήσεων) χρησιμοποιείται ως εγγενές κίνητρο (intrinsic reward), οπότε όσο μεγαλύτερη είναι η 'έκπληξη' του μοντέλου, τόσο μεγαλύτερη είναι η ανταμοιβή.

4.3.2 RND Signal (Random Network Distillation)

Η τυχαία απόσταξη δικτύου (RND), όπως και το curiosity signal, είναι επωφελής σε περιβάλλοντα με σπάνιες ή σπάνιες ανταμοιβές, επειδή ενθαρρύνει την εξερεύνηση από τον πράκτορα. Η υλοποίηση της μονάδας RND ακολουθεί την εργασία για την εξερεύνηση μέσω τυχαίας απόσταξης δικτύου[3]. Το RND χρησιμοποιεί δύο δίκτυα: το πρώτο είναι ένα δίκτυο με προκαθορισμένα τυχαία βάρη που λαμβάνει παρατηρήσεις ως εισόδους και παράγει μια κωδικοποίηση. Το δεύτερο δίκτυο έχει παρόμοια αρχιτεκτονική και εκπαιδεύεται για να προβλέπει τις εξόδους του πρώτου δικτύου χρησιμοποιώντας τις παρατηρήσεις του πράκτορα ως δεδομένα εκπαίδευσης. Ως εγγενής ανταμοιβή χρησιμοποιείται η απώλεια (η τετραγωνική διαφορά μεταξύ των προβλεπόμενων και των πραγματικών κωδικοποιημένων παρατηρήσεων) του εκπαιδευμένου μοντέλου. Όσο πιο συχνά επισκέπτεται ένας πράκτορας μια κατάσταση, τόσο πιο ακριβείς γίνονται οι προβλέψεις και όσο μικρότερες είναι οι ανταμοιβές, τόσο περισσότερο ενθαρρύνεται ο πράκτορας να εξερευνήσει νέες καταστάσεις με μεγαλύτερα σφάλματα πρόβλεψης.

4.3.3 BC signal (Behavioral Cloning)

Το BC signal δίνει εντολή στην πολιτική του πράκτορα να αναπαράγει με ακρίβεια τις ενέργειες σε ένα σύνολο επιδείξεων. Η λειτουργία BC μπορεί να ενεργοποιηθεί στους εκπαιδευτές PPO και SAC. Δεδομένου ότι το BC signal δεν μπορεί να γενικεύσει τα παραδείγματα που εμφανίζονται στις επιδείξεις, τείνει να λειτουργεί καλύτερα όταν υπάρχουν επιδείξεις για σχεδόν όλες τις πιθανές καταστάσεις που μπορεί να συναντήσει ο πράκτορας, σε συνδυασμό με GAIL ή/και μια εξωτερική ανταμοιβή.

4.3.4 GAIL (Generative Adversarial Imitation Learning)

Το GAIL signal, ή Genetic Adversary Imitation Learning[4], επιβραβεύει τον πράκτορά για την ίδια συμπεριφορά με ένα σύνολο επιδείξεων, χρησιμοποιώντας μια αντίθετη στρατηγική. Το GAIL signal μπορεί να χρησιμοποιηθεί με ή χωρίς ανταμοιβές περιβάλλοντος και είναι αποτελεσματικό όταν υπάρχουν λίγες επιδείξεις. Σε αυτό το πλαίσιο, ένα δεύτερο νευρωνικό δίκτυο, ο διαχωριστής, εκπαιδεύεται για να καθορίσει αν μια παρατήρηση ή ενέργεια είναι αποτέλεσμα μιας επίδειξης ή της παραγωγής του ίδιου του πράκτορα. Αυτός ο διαχωριστής μπορεί στη συνέχεια να διερευνήσει μια νέα παρατήρηση ή ενέργεια και να την επιβραβεύσει με βάση το πόσο παρόμοια πιστεύει ότι είναι με τις παρεχόμενες επιδείξεις.

Ο πράκτορας προσπαθεί να μάθει πώς να μεγιστοποιεί αυτή την ανταμοιβή σε κάθε βήμα εκπαίδευσης. Ο διαχωριστής εκπαιδεύεται στη συνέχεια να διακρίνει μεταξύ της κατάστασης και

των ενεργειών του πράκτορα και των επιδείξεων. Με αυτόν τον τρόπο, καθώς ο πράκτορας γίνεται πιο επιδέξιος στην αναπαραγωγή των επιδείξεων, ο διαχωριστής γίνεται όλο και πιο αυστηρός, απαιτώντας από τον πράκτορα να καταβάλει μεγαλύτερη προσπάθεια για να τον "ξεγελάσει". Αυτή η μέθοδος ανακαλύπτει μια πολιτική που παράγει καταστάσεις και ενέργειες παρόμοιες με τις επιδείξεις, απαιτώντας λιγότερες επιδείξεις από την άμεση αναπαραγωγή ενεργειών. Εκτός από τη μάθηση αποκλειστικά από τις επιδείξεις, το σήμα ανταμοιβής GAIL μπορεί να συνδυαστεί με ένα εξωτερικό σήμα ανταμοιβής για να κατευθύνει τη διαδικασία της μάθησης.

4.3.5 PPO trainer (Proximal Policy Optimization)

Ο PPO είναι ένας αλγόριθμος πολιτικής που στοχεύει στη βελτίωση της αποτελεσματικότητας και της σταθερότητας των μεθόδων διαβάθμισης πολιτικής με την ενίσχυση της αποτελεσματικότητας του δείγματος [5]. Αυτό επιτυγχάνεται μέσω μιας προσέγγισης περιορισμένης ενημέρωσης του δικτύου πολιτικής, η οποία διασφαλίζει ότι η ενημερωμένη πολιτική είναι παρόμοια με την προηγούμενη πολιτική, αποφεύγοντας έτσι σημαντικές αλλαγές που μπορεί να οδηγήσουν σε μη βέλτιστες πολιτικές. Ο PPO ενσωματώνει μια παράμετρο αποκοπής για τον περιορισμό των ενημερώσεων πολιτικής και τη βελτίωση της σταθερότητας. Έχει επιδείξει αποτελεσματικότητα σε περιβάλλοντα συνεχούς χώρου δράσης και έχει χρησιμοποιηθεί για ποικίλες εφαρμογές, όπως το παίξιμο βιντεοπαιχνιδιών και ο έλεγχος ρομποτικών συστημάτων.

Κεφάλαιο 5 Μεθοδολογία

5.1 Εισαγωγή

Σε αυτό το κεφάλαιο, θα εξετάσουμε σε μεγαλύτερο βάθος το επιδιωκόμενο έργο του πράκτορα. Επιπλέον, θα εξετάσουμε τις απαιτήσεις που είναι απαραίτητες για τη λειτουργία του έργου καθώς και τον τρόπο ανάπτυξης μίας αποτελεσματικής εκπαιδευτικής και θα περιγράψουμε το εξελιγμένο σύστημα RF για κάθε επίπεδο εκπαίδευσης.

5.2 Περιγραφή της εργασίας

Τα Single-Source Shortest Path (SSSP) προβλήματα είναι θεμελιώδεις υπολογιστικές εργασίες που συναντώνται σε διάφορες εφαρμογές, συμπεριλαμβανομένων των βιντεοπαιχνιδιών. Χρησιμοποιώντας αλγορίθμους ενισχυτικής μάθησης, τα προβλήματα SSSP στα βιντεοπαιχνίδια μπορούν να αντιμετωπιστούν με την εκπαίδευση πρακτόρων για την πλοήγηση σε πολύπλοκα περιβάλλοντα και την ανακάλυψη βέλτιστων διαδρομών για την επίτευξη συγκεκριμένων στόχων.

Το Pac-Man και το Super Mario Bros είναι δύο παραδείγματα βιντεοπαιχνιδιών που απεικονίζουν τη χρήση RL για προβλήματα SSSP. Στο Pac-Man, ένας πραγματικός πράκτορας μαθαίνει να πλοηγείτε στο λαβύρινθο και να βρίσκει το συντομότερο μονοπάτι για να συλλέξει όλα τα σφαιρίδια αποφεύγοντας τα φαντάσματα, λαμβάνοντας μπόνους για τη συσσώρευση σφαιριδίων

και ποινές για τη σύγκρουση με φαντάσματα. Παρομοίως, στο Super Mario Bros, ο πράκτορας μαθαίνει να διασχίζει το επίπεδο και να εντοπίζει τη γρηγορότερη διαδρομή προς τη σημαία λαμβάνοντας ανταμοιβές για τη συσσώρευση νομισμάτων, power-ups και την ολοκλήρωση του επιπέδου. Ο πράκτορας χρησιμοποιεί αλγορίθμους RL για να μάθει πώς να αποφεύγει εμπόδια, εχθρούς και κινδύνους, καθορίζοντας παράλληλα τη βέλτιστη διαδρομή προς τον στόχο.

Επιτρέποντας στους πράκτορες να μάθουν πώς να πλοηγούνται σε πολύπλοκα περιβάλλοντα, να βρίσκουν τη συντομότερη διαδρομή και να βελτιώνουν την πολιτική τους για να αποφεύγουν εμπόδια και κινδύνους, τεχνικές RL μπορούν να χρησιμοποιηθούν για την επίλυση προβλημάτων SSSP σε βιντεοπαιχνίδια. Αυτή η στρατηγική έχει ευρύτερες επιπτώσεις για τους σχεδιαστές και τους ερευνητές παιχνιδιών, καθώς η RL μπορεί να εφαρμοστεί σε άλλα παιχνίδια και σε εφαρμογές του πραγματικού κόσμου όπου πρέπει να αντιμετωπιστούν παρόμοια προβλήματα.

Σκοπός αυτής της εργασίας είναι να κατασκευάσουμε έναν πράκτορα, ο οποίος θα εκπαιδευτεί με αλγόριθμους RL, που θα έχει σαν στόχο να διασχίσει από σημεία ελέγχου και να φτάσει στον τελικό στόχο για να ολοκληρώσει το επεισόδιο. Ο πράκτορας θα πρέπει να μάθει να αποφεύγει τα εμπόδια, να εντοπίζει και να πηγαίνει στα σημεία ελέγχου και τον τελικό στόχο και να βρίσκει τη βέλτιστη διαδρομή προς τον τελικό στόχο.

5.3 Μεθοδολογία

Η εκπαίδευση του πράκτορα έχει χωριστεί σε δύο κατηγορίες εκπαίδευσης, εκ τον οποίων, η κάθε μια αποτελείτε από τέσσερις φάσης. Στο τέλος, θα γίνει η σύγκρισή μεταξύ αυτόν τον κατηγοριών εκπαίδευσης και θα συζητήσουμε τα σχετικά αποτελέσματα που θα μας επιστρέψουν εφόσον τα εφαρμόσουμε σε σταθερά περιβάλλοντα εκπαιδεύσεις, ώστε ο πράκτορας να εκπαιδεύεται στο ίδιο το περιβάλλον και να έχουμε ακριβής δεδομένα.

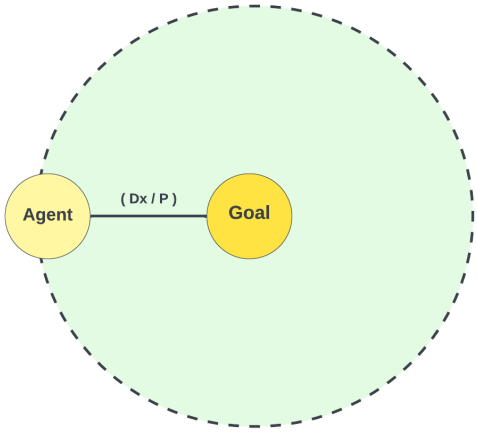
Στην Simple εκπαίδευση (Simple μοντέλα), ο πράκτορας θα εκπαιδευτεί σε ένα περιβάλλον με sparce rewards, δηλαδή τα reward signals που θα λαμβάνει από το περιβάλλον θα είναι πολύ αραιά, από άποψη χρόνου, και θα σηματοδοτούνται μόνο όταν ο πράκτορας φτάσει σε κάποια τερματική κατάσταση και πότε άλλοτε. Στη συγκεκριμένη εργασία, κάποιες τερματικές καταστάσεις θα μπορούσαν να είναι οι εξής :

- Να ολοκληρώσει το πρόβλημα,
- Να έρθει σε επαφή με κάποιο απαγορευμένο αντικείμενο,
- Η απόσταση που διανύει να είναι αρκετά μεγαλύτερη από την εκτίμηση του Dijkstra,
- Να μείνει στο ίδιο σημείο για αρκετή ώρα.

Να σημειωθεί ότι οι τερματικές καταστάσεις είναι ιδίες ανεξάρτητος της κατηγορίας εκπαίδευσης. Καθώς, θέλουμε τον πράκτορα να αναπτύξει την ίδια πολιτική για οποιαδήποτε μορφή

εκπαίδευσης, η αλλαγή τον τερματικών καταστάσεων θα μπορούσε να οδηγήσει σε ανεπιθύμητα αποτελέσματα και συμπεριφορές.

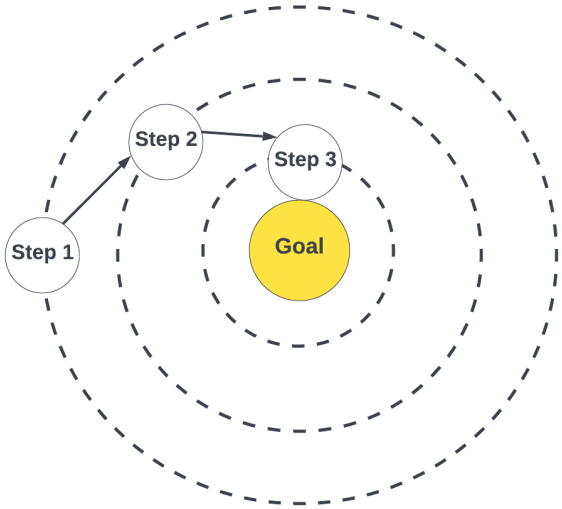
Η δεύτερη κατηγορία εκπαίδευσης, η Advanced, εφαρμόζει την λογική τις περίπλοκης συνάρτησης ανταμοιβής που αναλύσαμε στο Κεφάλαιο 4 (υλοποίηση τις complex RF). Αυτός ο τρόπος εκπαίδευσης χρησιμοποιεί έναν πιο αναπτυγμένο τρόπο ανταμοιβής. Ο πράκτορας λαμβάνει reward signals σε όλοι την διάρκεια τις εκπαίδευσης δίνοντας τους πληροφορίες για το περιβάλλον. Όπως έχει συζητηθεί και στο Κεφάλαιο 4, το κύριο χαρακτηριστικά αυτής τις συνάρτησης είναι η κλιμάκωση τις επιβράβευσης του πράκτορα όσο πλησιάζει τον στόχο από τον τύπο : $r = 1 - (Dx / P)^{e}$.



Εικόνα 7 Διαφορά απόστασης agent-goal

Όπως μας αναδεικνύει η Εικόνα 7, ο πράκτορας θα λαμβάνει μια μικρή επιβράβευση όσο κινητέ εντός του κύκλου (πράσινη περιοχή). Στο επόμενο βήμα, ο κύκλος θα έχει μέγιστη ακτίνα την καινούργια απόσταση (Dx/P) του πράκτορα από τον στόχο που αυτό θα οδηγήσει τον πράκτορα σε μια σπειροειδής κίνηση προς τον στόχο (Εικόνα 8). Κάθε ενέργεια που κάνει ο πράκτορας, αν

έχει οριστεί ως σημαντική ενέργεια, αποθηκεύεται και γίνεται συντελεστής στη τελική βαθμολογία κάθε επεισοδίου, εφόσον ο πράκτορας φτάσει σε κάποια τερματική κατάσταση. Αυτό δημιουργεί μια αλυσιδωτή εκπαίδευση οπού ο πράκτορας μαθαίνει να εκπληρώνει όλες τις δηλωμένες ενέργειες για να μεγιστοποίηση την τελική επιβράβευση.



Εικόνα 8 Σπειροειδής κίνηση του agent προς τον στόχο

Η εκπαίδευση, όπως προαναφέραμε, έχει χωριστή σε δυο κατηγορίες εκπαίδευσης, την Simple και την Advanced, και κάθε κατηγορία αποτελείτε από τέσσερις φάσης (Πίνακας 1).

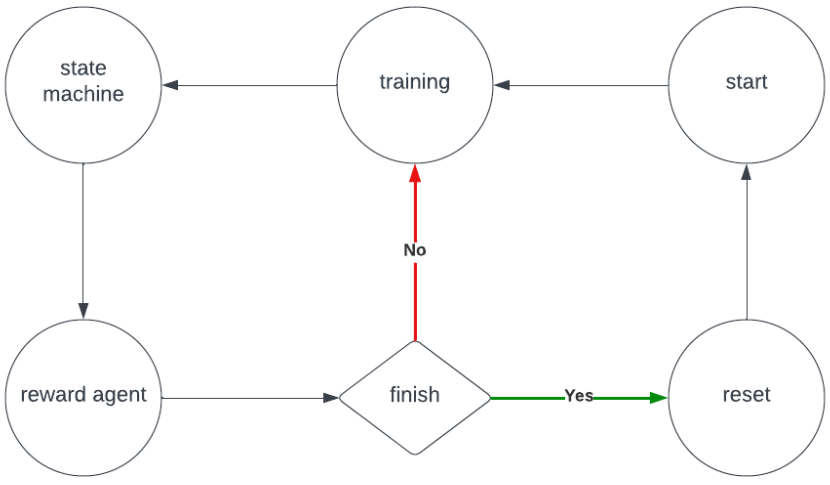
Models	Phase A	Phase B	Phase C	Phase D
Simple	SA	SB	SC	SD
Advanced	AA	AB	AC	AD

Πίνακας 1 Simple και Advanced μοντέλα

Συγκεκριμένα, κάθε φάση καθορίζει τη συμπεριφορά και τον στόχο τις εκπαίδευσης. Μπορούμε να το σκεφτούμε σαν να έχουμε χωρίσει το πρόβλημα σε μικρότερα υπό προβλήματα, που είναι

πιο εύκολο για τον πράκτορα να τα λύση ένα προς ένα, σε αντίθεση με το αρχικό πρόβλημα που μπορεί να τον δυσκολέψει αρκετά λόγο περιπλοκότητας.

Στην έναρξη κάθε προπόνησης, ορίζετε εξ αρχής σε ποιο μοντέλο εκπαίδευσης (Simple ή Advanced) και σε ποια φάση (A,B,C ή D) θα προπονηθεί ο πράκτορας. Η επιλογή μοντέλου και φάσης περνάνε σαν παράμετροι στο state machine, όπου αυτό αναλαμβάνει να αρχικοποίηση τις τερματικές κατάστασης, τον σκοπό και τις επιβράβευσης. Από εκεί και έπειτα, ακολουθεί μια επαναληπτική διαδικασία οπού ο πράκτορας εκτελεί ενέργειες, αυτές τις ενέργειες τις επεξεργάζεται το state machine και επιστρέφει κάποια επιβράβευση εάν είναι απαραίτητο. Στο τέλος, ελέγχει για τυχόν τερματικές κατάστασης που μπορεί να ενεργοποιήθηκαν και είτε συνεχίζει το επεισόδιο (αν δεν έχει βρεθεί κάποια τερματική κατάσταση), είτε κάνει επαναφορά του επεισοδίου στις αρχικές συνθήκες και ξανά αρχίζει από την αρχή (Εικόνα 9).



Εικόνα 9 state machine flow

Και στις δυο περίπτωσης, είτε Simple είτε Advanced, οι φάσης εκπαίδευσης ακολουθούν την ιδιά λογική, με την διαφορά ότι το τελικό αποτελέσματα για την Advanced εκπαίδευση δίνετε αφού υπολογιστεί από την Complex RF. Σε αντίθεση με την Simple εκπαίδευση που χρησιμοποιεί μόνο sparce rewards.

Η εκπαίδευση αποτελείτε από μια πληθώρα λίστα με τερματικές συνθήκες (Πίνακας 2) που χρησιμοποιούνται, αναφορικά :

- TC 1: όταν, άφιξη στον στόχο,
- TC 2: όταν, άφιξη στον τελικό στόχο,
- TC 3: εάν, σύγκρουση με ανεπιθύμητα αντικείμενα.
- TC 4: εάν, η διανυσματική απόσταση που έκανε ο πράκτορας είναι N φορές μεγαλύτερη από την ελάχιστη απόσταση (υπολογιζόμενή από τον αλγόριθμο του Dijkstra)
- TC 5: εάν, τα steps που έχει κάνει ισούνται με το Max Steps που έχει οριστεί να κάνει για κάθε επεισόδιο,

	Φάση A	**Φάση B**	**Φάση C**	**Φάση D**
TC 1	✓	✓		
TC 2			✓	✓
TC 3	✓			
TC 4		✓		✓
TC 5	✓	✓	✓	✓

Πίνακας 2 Τερματικές Συνθήκες

Πάρα τις ομοιότητες στις τερματικές συνθήκες, η σημαντική διαφορά τις Advanced εκπαίδευσης από τις Simple είναι τα Reward Signals που δημιουργεί η κάθε μια και η συχνότητα τους. Αναφέραμε συνοπτικά προηγουμένως, ότι η Advanced εκπαίδευση δημιουργεί rewards σε όλοι την διάρκεια του επεισοδίου ενώ η Simple εκπαίδευση επιβραβεύει μόνο όταν τερματική συνθήκη ενεργοποιηθεί. Αυτό είναι και το κύριο μέρος που θα συγκρίνουμε στο τέλος. Το πόσο και το εάν είναι βέλτιστο να χρησιμοποιείτε μια Custom RF με shaped rewards για να λυθούν SSSP με την χρήση RL. Τα reward signals που έχουν ενσωματωθεί στην Advanced εκπαίδευση αναγράφονται παρακάτω (για διευκόλυνση θα ονομάσουμε τα reward signals ως RS):

- RS 1: Επιβραβεύσει του πράκτορα όσο πλησιάζει τον στόχο χρησιμοποιώντας το τύπο απόστασης, $r = 1 – (Dx / P)^{e}$,
- RS 2: Παροχή μιας ποινή με πολύ μικρή τιμή, λειτουργώντας σαν παράγοντας μεγιστοποίησης του τελικού συνολικού reward. Όσο πιο γρήγορα τελειώσει τόσο μεγαλύτερο το τελικό reward και επομένως λιγότερες ενέργειες και χρόνος,
- RS 3: Λαμβάνει μια μικρή ποινή μετά από t δευτερόλεπτα στην ίδια περιοχή, ωθώντας τον πράκτορα να εξερεύνηση και άλλες περιοχές,
- RS 4: Κάθε φορά που αλλάζει περιοχή, ελέγχεται εάν έχει ξαναπεράσει από εκεί. Στην περίπτωση αυτή, λαμβάνει μια ποινή διαφορετικά θεωρείτε ότι δεν έχει επισκεφτεί την περιοχή και επιβραβεύεται.
- RS 5: Αυτό το RS δημιουργεί ένα μονοπάτι από N αντικείμενα πού δίνουν μια μικρή ανταμοιβή εάν τα συλλέξει ο πράκτορας. Για κάθε κόμβο κατά μήκος της βέλτιστης, υπολογισμένης από τον Dijkstra διαδρομής, τοποθετούνται σε ευθεία γραμμή από κόμβο σε κόμβο οδηγώντας προς την βέλτιστη διαδρομή.

Κατά συνέπεια, ο πράκτορας "βλέπει" ορισμένους όρους και δέχεται πληροφορίες μέσο τον reward signals οπού συνδυάζοντας αυτές τις πληροφορίες, μας δίνουν την τελική τιμή αξιολόγησης για κάθε επεισόδιο.

	Φάση A	Φάση B	Φάση C	Φάση D
RS 1	✓	✓	✓	✓
RS 2		✓		✓
RS 3	✓	✓	✓	✓
RS 4*		✓		✓
RS 5*	✓	✓	✓	✓

Πίνακας 3 Reward signals ανά φάση

*αυτά τα RS μπορούν να ενεργοποιηθούν / απενεργοποιηθούν ανάλογα την κρίση του ερευνητή.

5.4 Reward Hacking και Side Effects

Σε αυτή την ενότητα θα εξετάσουμε μερικά ακόμη τεχνικά ζητήματα, τόσο στην εφαρμογή όσο και σε ζητήματα που προέκυψαν κατά την εφαρμογή και την εκπαίδευση. Όπως αναφέρθηκε στα Κεφάλαια 2 και 3, η προκλητική πτυχή της RL είναι το πώς παρέχονται σήματα ανταμοιβής στον πράκτορα για να μάθει μια απολύτως σωστή πολιτική και στη συνέχεια να τη βελτιστοποιήσει, ενώ βρίσκεται πάντα κάτω από το πρίσμα κάποιου κανόνα/ες που ορίζει ο ερευνητής. Είναι προφανές ότι ο πράκτορας μπορεί να παράγει λύσεις με απρόβλεπτη συμπεριφορά, ακόμη και αν αυτό έχει σαν αποτέλεσμα να ολοκληρώσει με επιτυχία το επεισόδιο. Οι τυπικές ανησυχίες της RL περιλαμβάνουν τη χειραγώγηση της ανταμοιβής(reward hacking), τις αρνητικές παρενέργειες(side effects), την κλιμακούμενη εποπτεία (scalable oversight) και την ασφαλή

εξερεύνηση (safe exploration). Διάφορα ζητήματα, συμπεριλαμβανομένης της χειραγώγηση της ανταμοιβής, αρνητικές παρενέργειες και την ασφαλή εξερεύνηση εμφανίστηκαν κατά τη διάρκεια της εκπαίδευσης στην παρούσα μελέτη. Παρακάτω, θα εξετάσουμε σε μεγαλύτερο βάθος τα προβλήματα που αντιμετωπίσαμε και τις λύσεις για το καθένα.

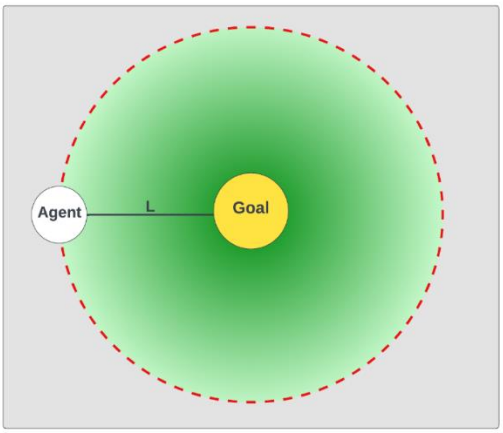
5.4.1 Reward Hacking

Κατά τη διάρκεια της εκπαίδευσης εμφανίστηκαν διάφορες περίεργες συμπεριφορές καθώς ο πράκτορας προσπαθούσε να λύση το πρόβλημα. Κάποια από αυτά μπορούν να χαρακτηριστούν ως reward hacking προβλήματα. Το reward hacking, αναφέρεται σε θέματα συμπεριφοράς στα οποία ο πράκτορας ανακαλύπτει τρόπους για να αυξήσει τη συνολική ανταμοιβή χωρίς να εφαρμόσει την κατάλληλη πολιτική ή να βρει την λύση του προβλήματος. Είναι προφανής ότι τέτοια προβλήματα χρειάζονται άμεση αντιμετώπιση και ανασχεδιασμό τον reward signals που δέχεται ο πράκτορας από το περιβάλλον. Παρακάτω θα δούμε κάποια από τις reward hacking περιπτώσεις που συνέβησαν και τον τρόπο αντιμετώπισης που λήφθηκε.

1. Ο πράκτορας έβρισκε όσο πιο γρήγορα μπορούσε τον στόχο αλλά κρατάει μια απόσταση ασφαλείας από τον στόχο ώστε να μην τελειώσει το επεισόδιο και να αύξηση την συσσωρευόμενη ανταμοιβή,
2. Ο πράκτορας κάνει κύκλους γύρω από τον τελικό στόχο ή κάνει επαναληπτικά κινήσεις προς τα εμπρός και πίσω χωρίς να τελειώνει το επεισόδιο και να αύξηση την συσσωρευόμενη ανταμοιβή,

Το πρώτο πρόβλημα δημιουργήθηκε επειδή οι ρυθμίσεις του step reward signal που λάμβανε δεν του απαγόρευαν να λαμβάνει ανταμοιβή για κάθε βήμα που ολοκλήρωνε. Ως εκ τούτου, ο πράκτορας το εκμεταλλευόταν αυτό και παρέμενε στο πλησιέστερο δυνατό σημείο για να αυξήσει την τελική συνολική ανταμοιβή. Η αρχική λύση αυτής της συμπεριφοράς ήταν σχετικά απλή. Έγινε μία μετατροπή στο step reward signal, οπού πλέον ο πράκτορας λάμβανε επιβράβευση μόνο άμα το καινούργιο reward είναι μεγαλύτερο του προηγούμενου. Με αυτόν τον τρόπο δημιουργείτε ένα κύκλος με κέντρο τον στόχο, οπού για να λάβει κάποια ανταμοιβή ο πράκτορας πρέπει να κυκλοφορεί εντός αυτού. Επιπλέον για κάθε αντίθετα, προς τον στόχο, κατευθυνόμενο βήμα λαμβάνει μια μικρή αρνητική ανταμοιβή κατευθύνοντας τον πράκτορα προς τον στόχο χωρίς να τον εμποδίζουμε να εξερευνήσει το γύρω περιβάλλον, αν το επιλέξει. Όπως απεικονίζεται στην Εικόνα 10, ο πράκτορας βρίσκεται σε μια απόσταση από το στόχο, παράγοντας έναν κύκλο ακτίνας L και με κέντρο τον στόχο. Αυτό μας αφήνει με τέσσερις πιθανές περιπτώσεις:

- Ο πράκτορας παραμένει ανενεργός ή κουνιέται πάνω στην κόκκινη γραμμή, όπου δεν λαμβάνει κάποιο step reward signal,
- Ο πράκτορας κουνιέται εκτός του κύκλο(γκρι περιοχή), λαμβάνοντας μια αρνητική ανταμοιβή για κάθε βήμα,
- Ο πράκτορας πάει σκόπιμα πάνω σε κάποια τερματική συνθήκη(πχ. πέφτει σε τοίχο),
- Ο πράκτορας θα κινηθεί εντός του κύκλου(πράσινη περιοχή), όπου θα λάβει θετική ανταμοιβή,



Εικόνα 10 Διαχωρισμός Περιοχών

Μετά την εφαρμογή της προηγούμενης λύσης, προέκυψε ένα δεύτερο ζήτημα. Ο πράκτορας δημιουργούσε πλέον μια συμπεριφορά όπου κάνει κύκλους γύρω από τον στόχο ή κινείται προς τα εμπρός και πίσω σε μια προσπάθεια να πλησιάσει όσο το δυνατόν πιο κοντά στον στόχο αποφεύγοντας να έρθει σε επαφή μαζί του. Αυτό οφείλεται στο γεγονός ότι το σήμα ανταμοιβής που θα λάβει όταν ακουμπήσει τον στόχο είναι μικρότερο από τα step reward signals που θα λάβει αν πλησιάσει αρκετά κοντά στον στόχο. Εφόσον ο πράκτορας επιδιώκει να μεγιστοποιήσει την ανταμοιβή, θα προσπαθήσει να πλησιάσει αρκετά κοντά στο στόχο ώστε να λάβει τα μέγιστο συνολική επιβράβευση από τα step reward signals και σε αυτήν την προσπάθεια θα κατάληξη να ακουμπήσει τον στόχο. Δηλαδή με άλλα λόγια, ο πράκτορας δεν ψάχνει να βρει τον στόχο, απλά τυχαίνει ο στόχος να είναι στον δρόμο του. Αυτό έχει ως αποτέλεσμα την πιθανότητα ο πράκτορας να μην έρθει ποτέ σε επαφή με τον στόχο, εάν ο πράκτορας κατανάλωση όλον τον διαθέσιμο χρόνο προσπαθώντας να μεγιστοποιήσει το step reward signal.

Για την επίλυση αυτού το προβλήματος τροποποιήθηκαν διαφορά δεδομένα στα reward signals. Συγκεκριμένα, αυξήθηκε η ανταμοιβή που λαμβάνει αν ακουμπήσει στο στόχο, με τιμή ίση ή μεγαλύτερη από την συνολική τιμή που μπορεί να δώσει το step reward signal. Με αυτόν τον τρόπο, ανεξάρτητα από το πόσο γρήγορα ο πράκτορας εντοπίζει το στόχο, θα λαμβάνει πάντα μεγαλύτερη ανταμοιβή από το στόχο. Επιπλέον, εφαρμόστηκε μια αρνητική ανταμοιβή για κάθε απόφαση που λαμβάνει ο πράκτορας. Με αυτόν τον τρόπο, ο πράκτορας προσπαθεί να ολοκληρώσει τον επεισόδιο όσο το δυνατόν γρηγορότερα, απαλείφοντας τον ενδεχόμενο να μένει ακίνητος ή να κάνει κύκλους γύρω από τον στόχο.

5.4.1 Side Effects

Οι παρενέργειες (side effects) είναι προβλήματα που προκύπτουν κυρίως ως αποτέλεσμα της λήψης λανθασμένων σημάτων ανταμοιβής από το περιβάλλον. Όπως αναφέρθηκε στο Κεφάλαιο 3, ο πράκτορας μπορεί να φτάσει σε ένα σημείο όπου η βέλτιστη επιλογή για τη μεγιστοποίηση της συνολικής ανταμοιβής μπορεί να είναι να επιλέξει κάποια τερματική συνθήκη το συντομότερο δυνατό. Στην προκειμένη περίπτωση, ο πράκτορας εφάρμοσε μια στρατηγική κατά την οποία μόλις ξεκινούσε το επεισόδιο, αναζητούσε τον πλησιέστερο τοίχο και πήγαινε απευθείας πάνω σε αυτόν για να τελειώσει το επεισόδιο. Έτσι στα ματιά του πράκτορα, εξασφάλιζε ότι η συνολική ανταμοιβή θα υπερέβαινε την ανταμοιβή επιτυχίας κατά την ολοκλήρωση του έργου. Αυτό οφειλόταν σε εσφαλμένη κατανομή των ανταμοιβών. Ο πράκτορας, προκειμένου να ολοκληρώσει σωστά το έργο, έφτασε σε μια κατάσταση στην οποία η συνολική ανταμοιβή ήταν μικρότερη από την ανταμοιβή όταν έκανε άμεση ολοκλήρωση του έργου. Ως λύση, εφαρμόστηκε η

αναπροσαρμογή στις τιμές των ανταμοιβών. Σε κάθε τελική συνθήκη ανατέθηκε ένας περιορισμός με τη μορφή τιμωρίας, δηλαδή με κάθε αρνητή τερματική συνθήκη ο πράκτορας λαμβάνει μια αρκετά μεγάλη αρνητική ποινή. Μαζί με τον συνδυασμό και την ισορρόπηση της επιβράβευσης του step reward signal και των θετικών τερματικών συνθηκών. Με αυτόν τον τρόπο, ο πράκτορας κατανοεί ότι για να αυξήσει την ανταμοιβή, πρέπει να επιτύχει τον στόχο του και ότι κάθε άλλη μη επιεικής τερματική συνθήκη έχει αρκετά μεγάλο αρνητικό αριθμητικό κόστος που αντικαθιστά την προηγούμενη πρόοδο.

Κεφαλαίο 6 Αποτελέσματα και Συμπεράσματα

6.1 Εισαγωγή

Στην παρούσα ενότητα αναλύουμε και συγκρίνουμε τα αποτελέσματα της εκπαίδευσης των πρακτόρων RL στο πρόβλημα της βέλτιστης διαδρομής χρησιμοποιώντας τον αλγόριθμο PPO σε συνδυασμό με τους αλγορίθμους και reward signals GAIL, RND, BC και Curiosity Signal (K 4) για κάθε φάση εκπαίδευσης. Θα παρουσιάσουμε τη σωρευτική ανταμοιβή κάθε φάσης, την περίοδο εκπαίδευσης και το ποσοστό επιτυχίας εύρεσης βέλτιστου μονοπατιού για κάθε κατηγορία εκπαίδευσης, Simple και Advanced (K 5.3). Για την απόκτηση ακριβέστερων ευρημάτων, τα αποτελέσματα θα δημιουργηθούν από εκπαιδεύσεις στο ίδιο περιβάλλον με παρόμοια δυσκολία και τυχαιοποίηση σε όλα τα επίπεδα. Για την εξαγωγή όλων των δεδομένων θα χρησιμοποιηθεί το TensorBoard, ένα διαδικτυακό εργαλείο οπτικοποίησης δεδομένων που παρέχεται από την TensorFlow.

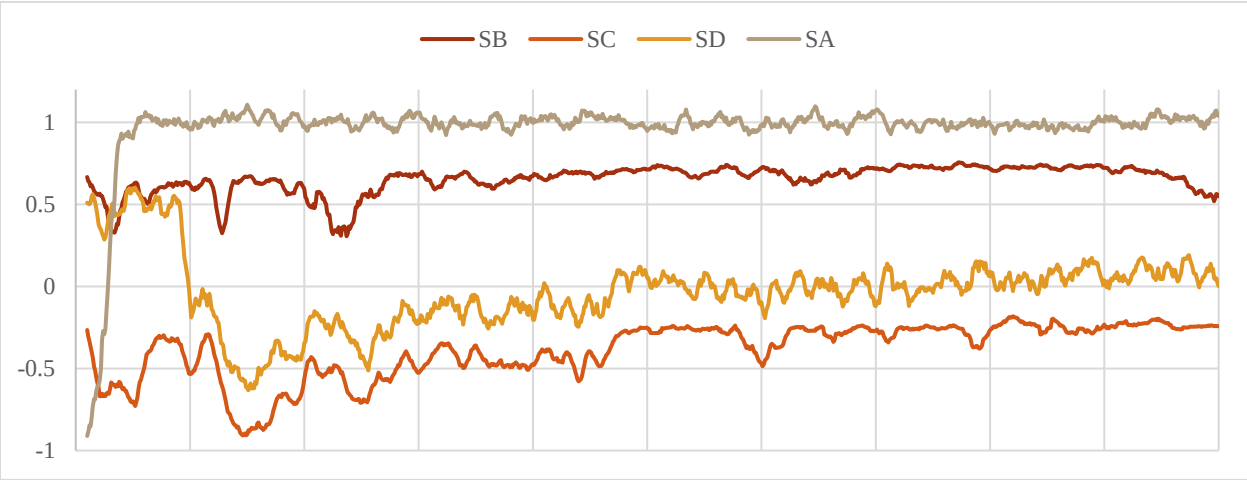
6.2 Αποτελέσματα

Στη συνέχεια, θα εξετάσουμε τα αποτελέσματα για κάθε κατηγορία εκπαίδευσης και για κάθε βήμα που τη χειρίζεται. Αρχικά θα εξετάσουμε τα δεδομένα για τα Simple μοντέλα, στη συνέχεια για τα Advanced μοντέλα και στο τέλος θα συγκρίνουμε τις εύρεσης του βέλτιστου μονοπατιού.

Ξεκινώντας από τα Απλά μοντέλα, δημιουργούν ένα σενάριο στο οποίο ο RL πράκτορας λαμβάνει περιορισμένα ερεθίσματα (K3.2, sparce rewards) στην προσπάθειά του να λύσει το πρόβλημα. Δηλαδή, όταν επιτυγχάνει μια τελική κατάσταση, ανταμείβεται ή τιμωρείται. Ο RL πράκτορας γίνεται "τυφλός", αφού λαμβάνει ελάχιστες έως καθόλου οδηγίες σχετικά με την κατεύθυνση του στόχου ή σήματα ανταμοιβής από το περιβάλλον.

Η εξέλιξη της συσσωρευμένης ανταμοιβής απεικονίζεται στο Σχήμα 3. Βλέπουμε ότι το μοντέλο SA έχει ήδη αναπτύξει μια πολιτική που επιστρέφει ανταμοιβές με τιμές μεγαλύτερες του 1 από τα πρώτα 1 εκατομμύρια steps. Συγκριτικά, το μοντέλο SB φαίνεται να βρίσκεται σε σταθερή πορεία χωρίς διακριτή βελτίωση. Ενώ η συσσωρευμένη ανταμοιβή στα μοντέλα SC και SD μειώνεται σημαντικά μετά από 2 εκατομμύρια steps και μετά.

Παρά την αρχική του επιτυχία, ο πράκτορας φαίνεται να μειώνει σημαντικά την απόδοσή του καθώς το πρόβλημα γίνεται πιο δύσκολο.

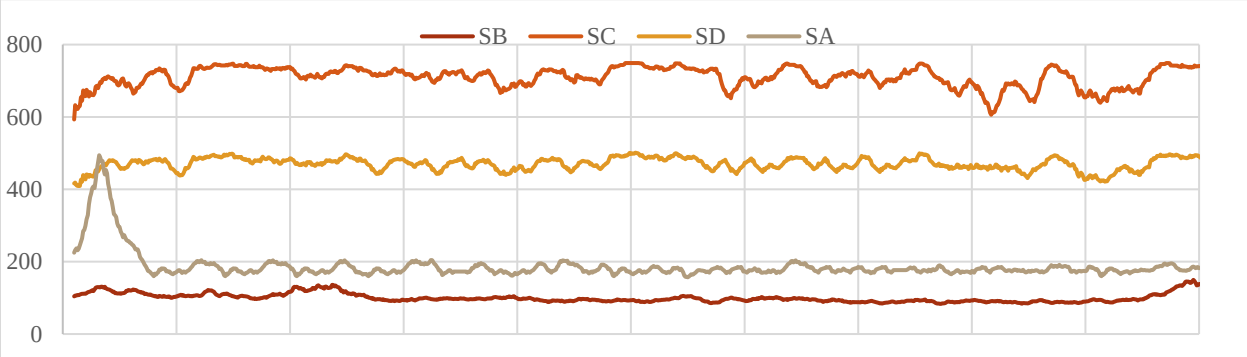


Σχήμα 3 Simple Training, Cumulative Reward

Cumulative Reward	Average	Min	Max	Training (Hours)	Steps (Millions)
SA	0.95	-1	1.1	5.3	10
SB	0.65	-0.01	0.8	10.3	10
SC	-0.38	-1	0.1	5.1	10
SD	-0.3	-1	0.4	4.2	10

Πίνακας 4 Simple Training, Cumulative Reward Stats

Το Σχήμα 4 και ο Πίνακας 5 απεικονίζουν τον χρόνο που απαιτείται για την ορθή ολοκλήρωση κάθε επεισοδίου. Η ανάλυση διάκρισης έδειξε σημαντική μείωση του χρόνου ολοκλήρωσης για τα μοντέλα SB σε σχέση με τα μοντέλα SA, με διαφορά 59,52%. Ομοίως, τα μοντέλα SC και SD επιδεικνύουν αξιοσέβαστη μείωση, με αξιοσημείωτη διαφορά 33,86%.

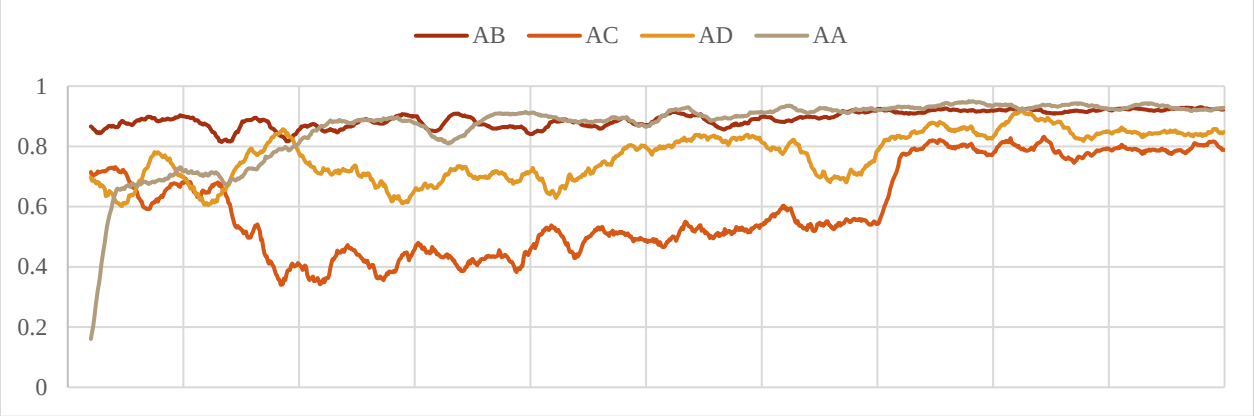


Σχήμα 4 Simple Training, Episode Length

Episode Length	Average	Min	Max
SA	247	92	562
SB	100	70	198
SC	709	165	749
SD	469	258	525

Πίνακας 5 Simple Training, Episode Length Stats

Ακολουθούν τα Advanced μοντέλα. Αυτά τα μοντέλα χρησιμοποιούν τη συνάρτηση ανταμοιβής που ορίζεται στο κεφάλαιο 3 για τη συλλογή σημάτων ανταμοιβής. Σε αντίθεση με το μοντέλο SA, το μοντέλο AA χρειάστηκε 2 εκατομμύρια βήματα (όπως φαίνεται στο Σχήμα 5) προτού φτάσει την τιμή ανταμοιβής 0.8. Το μοντέλο AB, από την άλλη πλευρά, διατήρησε ένα σταθερό εύρος ανταμοιβής από 0.85 έως 0.99, ενισχύοντας σημαντικά την πολιτική του μοντέλου. Το μοντέλο AC, για παράδειγμα, παρουσίασε μια μικρή πτώση μετά από 3 εκατομμύρια βήματα, αλλά κατάφερε και ανέκαμψε, ξεκινώντας μια εξαιρετική αυξητική τάση που έφτασε την τιμή ανταμοιβής στα 0.8 μετά των 8 εκατομμυρίων βημάτων. Ακολουθώντας τα χνάρια του AC, το μοντέλο AD επέδειξε πιο σταθερή απόδοση, ενώ βελτίωνε συνεχώς την πολιτική του και τις αποδόσεις του μοντέλου

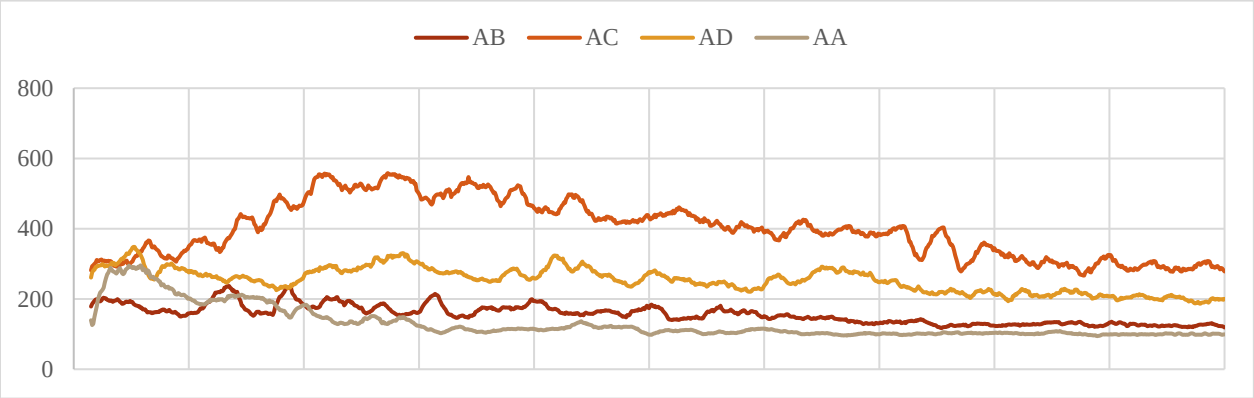


Σχήμα 5 Advanced Training, Cumulative Reward

Cumulative Reward	Average	Min	Max	Training (Hours)	Steps (Millions)
AA	0.85	0.01	0.99	11.5	10
AB	0.89	0.65	0.97	9	10
AC	0.59	0.056	0.86	7.3	10
AD	0.76	0.30	0.99	5.2	10

Πίνακας 6 Advanced Training, Cumulative Reward Stats

Εξετάζοντας τα ευρήματα στο Σχήμα 6 και στον Πίνακα 7, μπορούμε να δούμε σαφώς ότι τα μοντέλα AB σε σχέση με τα μοντέλα AA εμφανίζουν μια ορατή αύξηση στις χρονικές ολοκληρώσεις, με σημαντική αύξηση του χρόνου ολοκλήρωσης κατά 20,18%. Τα μοντέλα AC και AD, από την άλλη πλευρά, παρουσιάζουν σημαντική μείωση των χρονικών ολοκληρώσεων, με διαφορά 70,90% μείωση στον χρόνο εκτέλεσης στο AD μοντέλο.



Σχήμα 6 Advanced Training, Episode Length

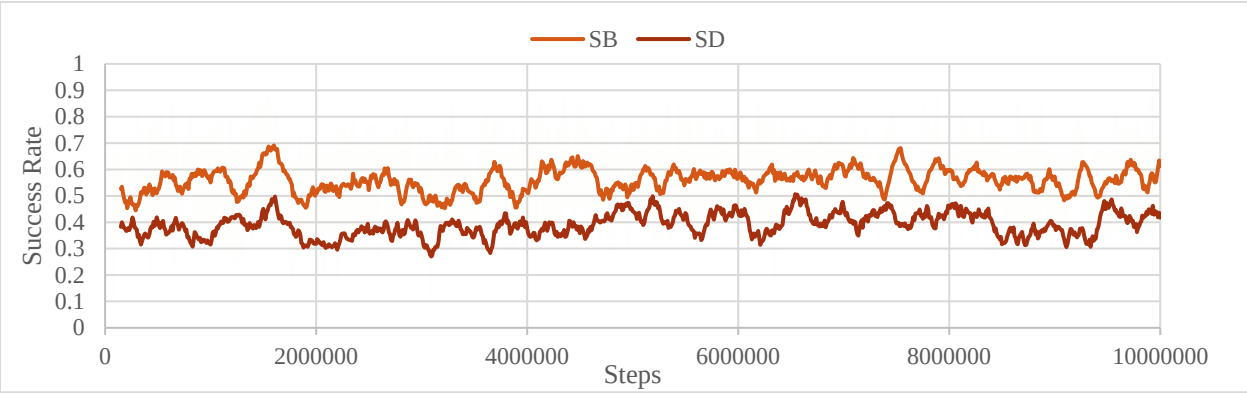
Episode Length	Average	Min	Max
AA	130	47	422
AB	156	92	368
AC	393	125	670
AD	253	114	429

Πίνακας 7 Advanced Training, Episode Length Stats

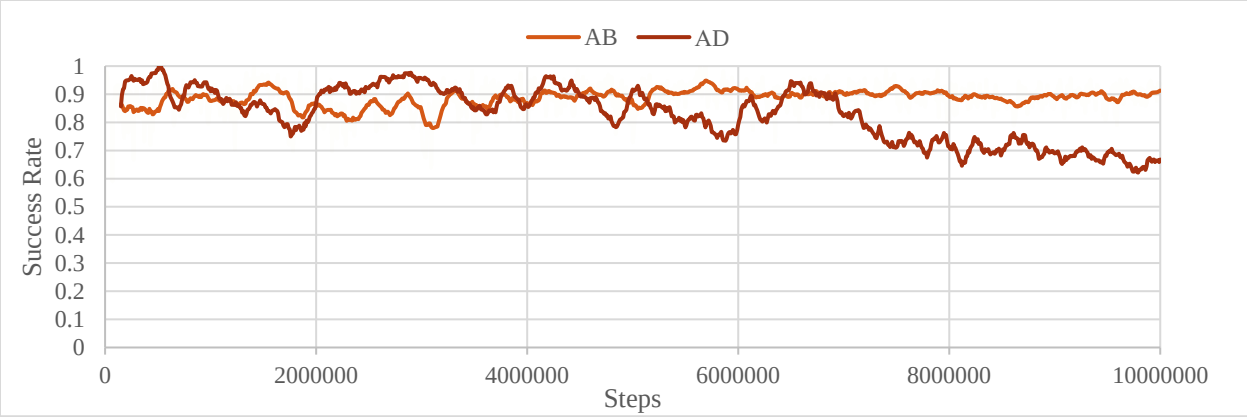
Με μια πιο προσεκτική εξέταση, γίνεται φανερό ότι τα Simple μοντέλα παρουσιάζουν ταχύτερους ρυθμούς ολοκλήρωσης των επεισοδίων, αλλά οι χρόνοι ολοκλήρωσης υπολείπονται σε σύγκριση με τα Advanced μοντέλα κατά σχεδόν διπλάσια διάρκεια. Συγκεκριμένα, κατά την εξέταση των φάσεων D (όπως περιγράφεται στους Πίνακες 5 και 7), η διαφορά μεταξύ των δύο μοντέλων

γίνεται έντονα εμφανής, με σημαντική διαφορά 37.05% και το AD μοντέλο να αναδεικνύεται ως ‘νικητής’, ξεπερνώντας το SD με σημαντική διαφορά όσον αφορά το χρόνο εκτέλεσης.

Στρέφοντας την προσοχή μας στην ουσία του προβλήματος και στο θεμελιώδες θέμα της διατριβής, εμβαθύνουμε στο κρίσιμο ερώτημα κατά πόσον τα μοντέλα μπορούν να παράγουν μια πολιτική που εντοπίζει την βέλτιστη διαδρομή. Επιπλέον, διερευνούμε πώς τα μοντέλα με διαφορετικές προσεγγίσεις επηρεάζονται από τον τρόπο με τον οποίο αποκτούν πληροφορίες από το περιβάλλον τους. Για να αποκτήσουμε χρήσιμες γνώσεις σχετικά με την ολοκλήρωση επεισοδίων και την εύρεση βέλτιστης διαδρομής από τον πράκτορα RL, προσθέσαμε νέες μεταβλητές στον πίνακα TensorBoard και τις υλοποιήσαμε στον κώδικα του πράκτορα RL. Μπορούμε να συγκρίνουμε αποτελεσματικά τα ποσοστά επιτυχίας του απλού και του προηγμένου μοντέλου εξετάζοντας προσεκτικά τα Σχήματα 7 και 8.



Σχήμα 7 Simple Training, Shortest Path Success Rate



Σχήμα 8 Advanced Training, Shortest Path Success Rate

Κατά τη διάρκεια της ανάλυσης της Φάσης B, γίνεται ξεκάθαρα εμφανές ότι τα Advanced μοντέλα έχουν εξαιρετική ικανότητα στον προσδιορισμό της ιδανικής διαδρομής, έχοντας ένα εκπληκτικό ποσοστό επιτυχίας 88,6%. Αυτό το εκπληκτικό επίτευγμα ξεπερνά την απόδοση των Simple μοντέλων με μεγάλη διαφορά, μια βελτίωση κατά 32,7%. Προχωρώντας στη Φάση D, τα Advanced μοντέλα συνεχίζουν να εκπλήσσουν, με τις επιδόσεις τους να φτάνουν σε πρωτοφανή ύψη. Με ένα εξαιρετικό ποσοστό επιτυχίας 82,8%, όχι μόνο επιδεικνύουν την απαράμιλλη ικανότητά τους να ταξιδεύουν και να βρίσκουν τη βέλτιστη διαδρομή, αλλά και ξεπερνούν τα Simple μοντέλα κατά 43,8%.

Success Rate	Simple	Advanced	Diff
Φάση B	55.9%	88.6%	32.7%
Φάση D	39.0%	82.8%	43.8%

Πίνακας 8 Shortest Path Success Rates

Αυτές οι εντυπωσιακές διαφορές στα ποσοστά επιτυχίας μεταξύ του Advanced και του Simple μοντέλου αποδεικνύουν απερίφραστα τα σημαντικά πλεονεκτήματα και τις μεγαλύτερες ικανότητες των Advanced μοντέλων στην ακριβή επιλογή της ιδανικής διαδρομής.

6.3 Συμπεράσματα

Σε αυτή τη διατριβή, το ενδιαφέρον επικεντρώθηκε στην εκπαίδευση ενός RL πράκτορα για την επίλυση προβλημάτων εύρεσης συντομότερης διαδρομής σε περιβάλλον βιντεοπαιχνιδιών. Η κύρια μετρική αξιολόγησης της απόδοσης ήταν το ποσοστό επιτυχίας στην εύρεση του συντομότερου μονοπατιού, το οποίο μετρήθηκε με τη χρήση του αλγορίθμου του Dijkstra. Εξετάστηκαν δύο μοντέλα:

- Simple μοντέλα με αραιές ανταμοιβές (sparce rewards) και
- Advanced μοντέλα με προσαρμοσμένη συνάρτηση ανταμοιβής που ενσωματώνει διαμορφωμένες ανταμοιβές (shaped rewards).

Τα αποτελέσματα τις εκπαίδευσης και αξιολόγησης παρέχουν χρήσιμες πληροφορίες σχετικά με την αποτελεσματικότητα αυτών των μοντέλων. Κατά τη διάρκεια της Φάσης Β, τα Simple μοντέλα βρήκαν το συντομότερο μονοπάτι 55.9% του χρόνου, ενώ κατά μόλις 39% του χρόνου κατά τη διάρκεια της Φάσης D. Ενώ τα αποτελέσματα αυτά δείχνουν επαρκή απόδοση, υπογραμμίζουν επίσης τους περιορισμούς των αραιών ανταμοιβών στην καθοδήγηση του RL πράκτορα σε βέλτιστες απαντήσεις σε πιο περίπλοκες καταστάσεις.

Τα Advanced μοντέλα, από την άλλη πλευρά, ξεπέρασαν κατά πολύ τα Simple μοντέλα χρησιμοποιώντας μια τροποποιημένη συνάρτηση ανταμοιβής με shaped rewards. Στη Φάση Β, τα προηγμένα μοντέλα βρήκαν το συντομότερο μονοπάτι με ένα εκπληκτικό ποσοστό επιτυχίας 88,6% και ένα ποσοστό επιτυχίας 82,8% στη Φάση D. Μια αύξηση 32,7% και 43,8% αντίστοιχα για τις Φάσεις Β και D. Αυτά τα αποτελέσματα καταδεικνύουν την αποτελεσματικότητα των

διαμορφωμένων ανταμοιβών στην αποστολή πιο ενημερωτικών και ουσιαστικών σημάτων στον RL πράκτορα, με αποτέλεσμα την καλύτερη λήψη αποφάσεων και μεγαλύτερα ποσοστά επιτυχίας.

Η αξιοποίηση των διαμορφωμένων ανταμοιβών στα Advanced μοντέλα προσφέρει πολλά υποσχόμενες συνέπειες για την ανάπτυξη βιντεοπαιχνιδιών. Αξιοποιώντας RL τεχνικές για τη βελτίωση των συστημάτων εύρεσης διαδρομής και πλοήγησης, οι σχεδιαστές παιχνιδιών μπορούν να δημιουργήσουν πιο καθηλωτικές και προκλητικές εμπειρίες παιχνιδιού. Οι RL πράκτορες που εκπαιδεύονται για την επίλυση προβλημάτων SSSP μπορούν να συμβάλουν σε πιο ρεαλιστική συμπεριφορά NPC, έξυπνο σχεδιασμό επιπέδων και δυναμικές αλληλεπιδράσεις των παικτών στο περιβάλλον του παιχνιδιού.

Ωστόσο, είναι σημαντικό να αναγνωριστούν οι περιορισμοί αυτής της έρευνας. Το συγκεκριμένο περιβάλλον βιντεοπαιχνιδιών που χρησιμοποιήθηκε και η πολυπλοκότητα των SSSP προβλημάτων ενδέχεται να περιορίζουν τη γενίκευση του εκπαιδευμένου RL πράκτορα σε άλλα σενάρια. Μελλοντική έρευνα μπορεί να διερευνήσει πιο σύνθετες τεχνικές διαμόρφωσης ανταμοιβής, την χρήση περίπλοκων ANN, όπως το DNN, και του MDP μοντέλου για την μεγιστοποίηση της συσσωρευμένης ανταμοιβής , να αξιολογήσει την απόδοση των πρακτόρων RL σε διαφορετικά είδη παιχνιδιών και να διερευνήσει την επεκτασιμότητα και τη δυνατότητα μεταφοράς των εκπαιδευμένων μοντέλων σε διαφορετικά περιβάλλοντα βιντεοπαιχνιδιών διευρύνοντας την χρησιμότητα τους.

Εν κατακλείδι, η παρούσα πτυχιακή εργασία κατέδειξε τις δυνατότητες των RL τεχνικών στην επίλυση SSSP προβλημάτων σε περιβάλλοντα βιντεοπαιχνιδιών. Τα Advanced μοντέλα, εξοπλισμένα με μια προσαρμοσμένη συνάρτηση ανταμοιβής που ενσωματώνει διαμορφωμένες ανταμοιβές, παρουσίασαν αξιοσημείωτα ποσοστά επιτυχίας στην εύρεση της συντομότερης διαδρομής. Τα ευρήματα αυτά συμβάλλουν στην πρόοδο της ανάπτυξης βιντεοπαιχνιδιών, παρέχοντας πληροφορίες για την εφαρμογή της ενισχυτικής μάθησης για τη βελτίωση της εύρεσης μονοπατιών και της πλοήγησης. Η συνέχιση της έρευνας σε αυτόν τον τομέα θα ανοίξει το δρόμο για πιο εξελιγμένες και καθηλωτικές εμπειρίες βιντεοπαιχνιδιών, διευρύνοντας τα όρια της ενισχυτικής μάθησης στο σχεδιασμό παιχνιδιών και όχι μόνο.

Βιβλιογραφία

[1] A. H. Klopf. The hedonistic neuron: a theory of memory, learning, and intelligence. Toxicology-Sci, 1982.

[2] Pathak D., Agrawal P., Efros A., Darrell T., Curiosity-driven Exploration by Self-supervised Prediction, 2017

[3] Burda Y., Edwards H., Storkey A., Klimov O., Exploration by Random Network Distillation, 2018

[4] Ho J., Ermon S., Generative Adversarial Imitation Learning, 2016

[5] Schulman J., Wolski F., Dhariwal P., Radford A., Klimov O., Proximal Policy Optimization Algorithms, 2017

[6] E. W. Dijkstra. A note on two problems in connexion with graphs, 1959

[7] Algorithm Design , Kleinberg ,Jon Klenberg – Eva Tardos, 2006

[8] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In 2009 IEEE conference on computer vision and pattern recognition, pages 248–255. Ieee, 2009.

[9] K. He, X. Zhang, S. Ren, and J. Sun. Resnet-deep residual learning for image recognition. ResNet: Deep Residual Learning for Image Recognition, 2015.

[10] K. Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. Biological cybernetics, 36(4):193–202, 1980.

[11] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems, pages 1097–1105, 2012.

[12] S. Lawrence, C. L. Giles, A. C. Tsoi, and A. D. Back. Face recognition: A convolutional neural-network approach. IEEE transactions on neural networks, 8(1):98–113, 1997.

[13] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, et al. Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11):2278–2324, 1998.

[14] W. S. McCulloch and W. Pitts. A logical calculus of the ideas immanent in nervous activity. The bulletin of mathematical biophysics, 5(4):115–133, 1943.

[15] M. Minsky and S. Papert. Perceptron: an introduction to computational geometry. The MIT Press, Cambridge, expanded edition, 19(88):2, 1969.

[16] F. Rosenblatt. The perceptron, a perceiving and recognizing automaton Project Para. Cornell Aeronautical Laboratory, 1957.

[17] F. Rosenblatt. Perceptron simulation experiments. Proceedings of the IRE, 48(3):301–309, 1960

[18] D. E. Rumelhart, G. E. Hinton, R. J. Williams, et al. Learning representations by back-propagating errors. Cognitive modeling, 5(3):1, 1988.

[19] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In Thirty-First AAAI Conference on Artificial Intelligence, 2017.

[20] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K. J. Lang. Phoneme recognition using time-delay neural networks. Backpropagation: Theory, Architectures and Applications, pages 35–61, 1995.

[21] A. G. Barto, R. S. Sutton, and C. W. Anderson. Neuronlike adaptive elements that can solve difficult learning control problems. IEEE transactions on systems, man, and cybernetics, (5):834–846, 1983.

[22] R. Bellman. A markov decision process. journal of mathematical mechanics. 1957.

[23] R. Bellman. Combinatorial processes and dynamic programming. Technical report, RAND CORP SANTA MONICA CA, 1958.

[24] R. Bellman et al. The theory of dynamic programming. Bulletin of the American Mathematical Society, 60(6):503–515, 1954.

[25] A. H. Klopf. Brain function and adaptive systems: a heterostatic theory. Technical report, AIR FORCE CAMBRIDGE RESEARCH LABS HANSCOM AFB MA, 1972.

[26] L.-J. Lin. Self-improving reactive agents based on reinforcement learning, planning and teaching. Machine learning, 8(3-4):293–321, 1992.

[27] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller. Playing atari with deep reinforcement learning. arXiv preprint arXiv:1312.5602, 2013.

[28] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al. Human-level control through deep reinforcement learning. Nature, 518(7540):529, 2015.

[29] G. A. Rummery and M. Niranjan. On-line Q-learning using connectionist systems, volume 37. University of Cambridge, Department of Engineering Cambridge, England,

1994.

[30] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, et al. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. arXiv preprint arXiv:1712.01815, 2017.

[31] R. S. Sutton. Learning to predict by the methods of temporal differences. Machine learning, 3(1):9–44, 1988.

[32] R. S. Sutton and A. G. Barto. Reinforcement learning: An introduction. MIT press, 2018.

[33] E. Thorndike. Animal intelligence; experimental studies, by Edward l. Thorndike, 1911.

[34] C. J. C. H. Watkins. Learning from delayed rewards. 1989.

[35] B. Widrow, N. K. Gupta, and S. Maitra. Punish/reward: Learning with a critic in adaptive threshold systems. IEEE Transactions on Systems, Man, and Cybernetics, (5):455–465, 1973.

[36] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger. Densely connected convolutional networks. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 4700–4708, 2017.

[37] Andrew Y., Shaping and Policy Search in Reinforcement Learning, 2003

[38] Thomas VII Murphy. The first level of super mario bros. is easy with lexicographic orderings and time travel. The Association for Computational Heresy (SIGBOVIK) 2013, pages 112–133, 2013.

[39] Mark Humphrys. Action selection in a hypothetical house robot: Using those rl numbers. In Proceedings of the First International ICSC Symposia on Intelligent Industrial Automation (IIA-96) and Soft Computing (SOCO96), pages 216–22, 1996.

[40] Mark Ring and Laurent Orseau. Artificial General Intelligence: 4th International Conference

[41] Martin Pecka and Tomas Svoboda. Modelling and Simulation for Autonomous Systems: First International Workshop (MESAS 2014), chapter Safe Exploration Techniques for Reinforcement Learning – An Overview, pages 357–375. Springer International Publishing, 2014

[42] B. Widrow and M. E. Hoff. Adaptive switching circuits. Technical report, Stanford Univ Ca Stanford Electronics Labs, 1960.