ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΌΓΩΝ ΜΗΧΑΝΙΚΏΝ ΚΑΙ ΜΗΧΑΝΙΚΏΝ ΥΠΟΛΟΓΙΣΤΏΝ

[PLH 423]-Reinforcement Learning & Dynamic Optimization

Αναφορά 1ης εργασίας

Ιωάννης Περίδης

Α.Μ. 2018030069

1 Εισαγωγή:

Στην προγραμματιστική άσκηση αυτή, μας ζητήθηκε η υλοποίηση και σύγκριση των αλγορίθμων ε-greedy και upper confidence bound(UCB). Οι αλγόριθμοι, χρησιμοποιούνται σε ένα στοχαστικό περιβάλλον με k arm bandits, στο οποίο κάθε φορά επιλέγεται μόνο ένα χέρι από τα πολλά πιθανά και έχουν σαν στόχο την βελτιστοποίηση του αποτελέσματος(maximize reward και minimize regret), μέσα σε έναν πεπερασμένο ορίζοντα με T timesteps. Το πρόβλημα ανήκει στην ενότητα των προβλημάτων ενισχυτικής μάθησης.

2 Βασική Λειτουργία - Ομοιότητες & Διαφορές:

Και οι δύο αλγόριθμοι, προσπαθούν με διαφορετικούς τρόπους, να εξισορροπήσουν τα ποσοστά exploration και exploitation για να πετύχουν το ιδανικό trade-off μεταξύ τους.

Ο ***ε-greedy***, είναι ένας πολύ απλός αλγόριθμος, ο οποίος επιλέγει με μια πιθανότητα ε ένα τυχαίο χέρι και με πιθανότητα 1-ε το χέρι με το μέγιστο μέσο sampling $\hat{\boldsymbol{\mu}}_{\boldsymbol{i}}(\boldsymbol{t})$ (δηλαδή το πειραματικό , όχι το θεωρητικό) reward. Η παράμετρος ε, δεν είναι σταθερή (διότι αν ήταν, ο αλγόριθμος δεν θα είχε sublinear regret), αλλά εξαρτάται από τα timesteps t, σε κάθε μια επανάληψη και συγκεκριμένα, είναι επιθυμητό, να αρχίζει από κάποια μεγάλη τιμή και σταδιακά να μειώνεται. Με αποτέλεσμα να γίνεται περισσότερο exploration στην αρχή που δεν υπάρχει αρκετή γνώση για τα rewards των χεριών και περισσότερο exploitation κατά την διάρκεια, που είναι γνωστό, πιο είναι το βέλτιστο χέρι. Τίθεται , σύμφωνα με το αντίστοιχο θεώρημα, στην τιμή $\boldsymbol{t}^{-\frac{1}{3}} * \left(\mathbf{k} * \mathbf{log}(\mathbf{t})\right)^{\frac{1}{3}}$, έτσι επιτυγχάνεται sublinear regret καθ' όλη την διάρκεια του αλγορίθμου (και όχι μόνο για συνολικό χρόνο T) και ακόμη, παρατηρείται πως για την εύρεση της τιμής της παραμέτρου δεν είναι αναγκαία η γνώση του ορίζοντα. Χαρακτηριστικό του αλγορίθμου , είναι πως καταφέρνει να διαχωρίσει (σε κατάλληλες ποσότητες) το exploration από την αρχή , μέχρι το τέλος του αλγορίθμου και έτσι να αποφεύγονται τα πολύ υψηλά regrets στην αρχή και το «κόλλημα» σε κάποια suboptimal κατάσταση στο τέλος. Παρόλα αυτά, πρέπει να γίνει αντιληπτό πως το πότε θα γίνει ακριβώς exploration και ποιο χέρι είναι καλύτερο να δοκιμαστεί , γίνεται στην τύχη και δεν βασίζεται σε κάποια πληροφορία ή γνώση που χρησιμοποιεί ο αλγόριθμος.

Ο ***UCB***, είναι ένας πιο σύνθετος αλγόριθμος, ο οποίος επίσης, εξερευνά κατάλληλα σε όλων τον ορίζοντα αλλά, σε αντίθεση με τον ε-greedy, δεν κάνει το explore στην τύχη, αλλά αποθηκεύει και ενημερώνει χρήσιμες παραμέτρους, τις οποίες χρησιμοποιεί για να ξέρει ακριβώς πότε θα είναι καλύτερο να επιλέξει κάποιο διαφορετικό χέρι και συγκεκριμένα ποιο, θα του δώσει πιθανότατα καλύτερο reward, καθ' όλη την διάρκεια των επαναλήψεων. Η λειτουργία έχει ως εξής, επιλέγεται σε κάθε γύρο το χέρι με την μεγαλύτερη UCB τιμή. Η τιμή αυτή εξαρτάται από το άθροισμα δύο όρων, το μέγιστο μέσο sampling $\hat{\boldsymbol{\mu}}_{\boldsymbol{i}}(\boldsymbol{t})$ (ίδιο όπως πριν), το οποίο αντιπροσωπεύει την αξία ενός χεριού και τον όρο $\left(\frac{\boldsymbol{lnt}}{\boldsymbol{Q_i(t)}}\right)^{\frac{1}{2}}$,(όπου $\boldsymbol{Q_i(t)}$ το πλήθος φορών που τραβήχτηκε ένα χέρι), ο οποίος αντιπροσωπεύει το διάστημα εμπιστοσύνης, γύρω από την εκτιμώμενη τιμή. Συγκεκριμένα, ένα χέρι επιλέγεται είτε γιατί έχει πολύ μεγάλη αξία (μεγάλος πρώτος όρος), είτε επειδή δεν έχει επιλεχθεί (δεν έχει εξερευνηθεί) αρκετά (μικρό $\boldsymbol{Q_i(t)}$, δηλαδή μεγάλος δεύτερος όρος). Επίσης ισχύει και πάλι πως δεν χρειάζεται η γνώση του ορίζοντα T και η κάθε τιμή του εξαρτάται από το timestep t.

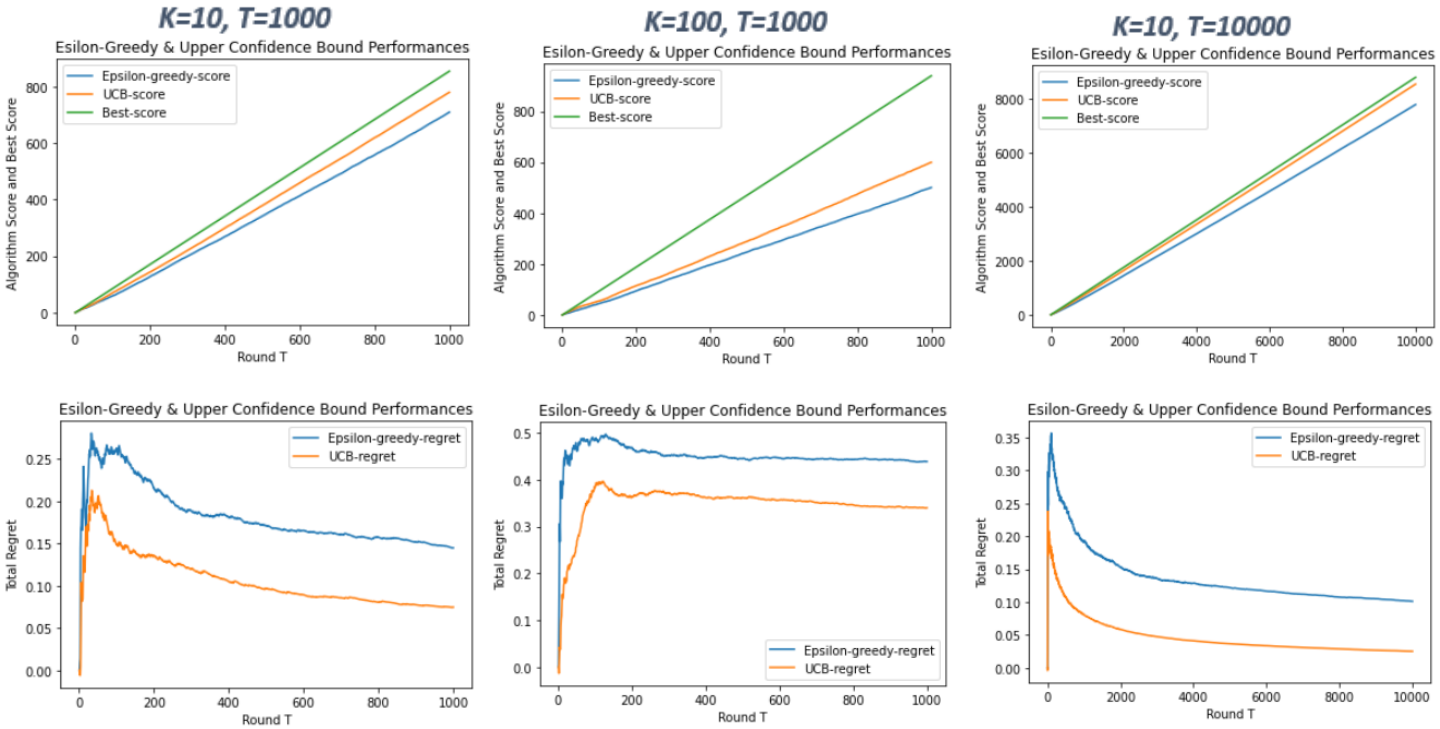
Είναι γνωστό από την θεωρία πως και οι δύο αλγόριθμοι πετυχαίνουν sublinear regret, συγκεκριμένα, μεγαλώνει το πολύ λογαριθμικά με τον χρόνο, παρόλα αυτά , ο UCB είναι καλύτερος αφού έχει το order optimal regret. Ακόμη, ο UCB θα έχει γρηγορότερο convergence speed και μεγαλύτερη προσαρμοστικότητα σε περιβάλλοντα που μεταβάλλονται. Παρόλα αυτά, έχει μεγαλύτερη πολυπλοκότητα στην υλοποίηση του και απαιτεί περισσότερη υπολογιστική δύναμη.

3 Υλοποίηση Κώδικα:

Ο κώδικας ήταν αρκετά απλός. Αρχικά υλοποιείται ο πίνακας k*2 με τα τυχαία διαστήματα [α,β] από 0 έως 1 που αντιστοιχούν σε κάθε χέρι. Ύστερα εκτυπώνονται η πραγματική μέση τιμή κάθε χεριού ,δηλαδή το σκορ και επισημαίνεται το καλύτερο σκορ και το χέρι στο οποίο παρατηρείται. Δημιουργήθηκε μια συνάρτηση για κάθε αλγόριθμο. Οι συναρτήσεις epsilon_greedy και UCB, παίρνουν σαν όρισμα τα timesteps, καθώς και τα χρησιμοποιούν για τον υπολογισμό του ε και του UCB, και επιστρέφουν το instance score, δηλαδή το reward για τον γύρο t. Μέσα στις συναρτήσεις, υπολογίζονται τα διανύσματα μεγέθους k(πλήθος χεριών) που υπολογίζουν τα sample means, δηλαδή τις μέσες τιμές , οι οποίες αντλούνται από Uniform κατανομές στα διαστήματα [α,β] που έχει το κάθε

χέρι. Ο UCB, επιπλέον κρατάει και διάνυσμα με τις τιμές UCB των χεριών. Ύστερα, βρίσκεται το καλύτερο χέρι για κάθε αλγόριθμο αντίστοιχα , το οποίο χρησιμοποιείται για να παραχθεί το instance score από αυτό και δίνεται σαν όρισμα στην συνάρτηση update arm. Η συνάρτηση αυτή, είναι υπεύθυνη για να ανανεώσει τον αριθμό φορών που τραβήχτηκε το χέρι(αύξηση κατά 1) και να αθροίσει το reward που έδωσε στο συνολικό. Ξεκινώντας τον αλγόριθμο, καλείται η συνάρτηση play_each_arm_once, η οποία τραβάει 1 φορά κάθε χέρι και πρακτικά είναι υπεύθυνη για την αρχικοποίηση των μέσων τιμών και των pulls. Έπειτα ,σε ένα for loop ,με πλήθος επαναλήψεων T(ορίζοντας), καλούνται οι συναρτήσεις και αποθηκεύονται ξεχωριστά τα algorithm_scores(αθροιστικά rewards), το best score(αθροιστικά βέλτιστα rewards) και τα regrets. Τέλος εκτυπώνονται ο συνολικός αριθμός φορών που τραβήχτηκε κάθε χέρι και όλα τα παραπάνω και συγκρίνονται όπως φαίνεται στις παρακάτω γράφηκες. Πιο αναλυτικές λεπτομέρειες βρίσκονται στα σχόλια του κώδικα.

4 Αποτελέσματα & Συγκρίσεις:



Στις γραφικές παραστάσεις ,πάνω απεικονίζονται τα αθροιστικά rewards των 2 αλγορίθμων σε σχέση με το βέλτιστο αθροιστικό reward και από κάτω φαίνονται τα regrets τους στον χρόνο.

Αρχικά, όπως ήταν και αναμενόμενο από την θεωρία ο UCB σε κάθε περίπτωση, είναι καλύτερος από τον ε-greedy, δηλαδή συγκλίνει πιο γρήγορα και μακροπρόθεσμα σε καλύτερη λύση με μικρότερο regret και μεγαλύτερο αθροιστικό σκορ. Επομένως, ο UCB έχει ταχύτερο χρόνο εκμάθησης , παρόλα αυτά όντως και οι 2 αλγόριθμοι έχουν sublinear regret, αφού πέφτει εκθετικά.

Συγκρίνοντας στην συνέχεια , κάθε περίπτωση ξεχωριστά παρατηρήθηκε πως :

Στην **1η περίπτωση** με **κ=10 και Τ=1000**, δηλαδή με αναλογία χεριών-timesteps **1:100**, οι αλγόριθμοι πετυχαίνουν αρκετά καλά αποτελέσματα, με ξεκάθαρη εκθετική μείωση του regret που κυμαίνονται: ε-greedy: από **0,35 έως 0,15 στο τέλος** και UCB: από **0,25 έως 0,1** στο τέλος.

Στην **2η περίπτωση** με κ=100 και Τ=1000, δηλαδή με αναλογία χεριών-timesteps **1:10**, και οι 2 αλγόριθμοι έδωσαν αρκετά χειρότερα αποτελέσματα, με μειωμένα σκορ και αυξημένα regrets, που κυμαίνονται: ε-greedy: από **0,5 έως 0,4 στο τέλος** και UCB: από **0,4 έως 0,3** στο τέλος. Το γεγονός αυτό οφείλεται στο πολύ μεγάλο πλήθος χεριών σε σχέση με τα βήματα, το οποίο εμποδίζει τους αλγορίθμους να μάθουν γρήγορα τα καλύτερα χέρια , αφού υπάρχουν πάρα πολλά πιθανά explorations να γίνουν σε πολύ μικρό χρονικό διάστημα. Έτσι, φαίνεται και στις γραφικές πως πλέον η εκθετική μείωση του regret έχει αλλοιωθεί και τείνει να γίνει ευθεία.

Στην **3η περίπτωση** με **κ=10 και Τ=10000**, δηλαδή με αναλογία χεριών-timesteps **1:1000**, και οι 2αλγόριθμοι, πετυχαίνουν πολύ καλύτερα αποτελέσματα από όλες τις περιπτώσεις , με αυξημένα σκορ πολύ κοντά στο βέλτιστο και μειωμένο regret, πολύ κοντά στο 0, , που κυμαίνονται: ε-greedy: από **0,35 έως 0,1 στο τέλος** και UCB: από **0,25 έως 0,05** στο τέλος. Το γεγονός αυτό οφείλεται στο πολύ μικρό πλήθος χεριών σε σχέση με τα timesteps, το οποίο επιτρέπει στους αλγορίθμους, να έχουν πολύ μεγάλο χρονικό περιθώριο για να κάνουν τα exploration που χρειάζεται και να καταλήξουν στα βέλτιστα αποτελέσματα ακριβώς και ύστερα να τα εκμεταλλευτούν όσο το δυνατό περισσότερο. Έτσι φαίνεται και στις γραφικές μια πολύ ακριβής και ξεκάθαρη εκθετική μείωση του regret , η οποία συγκλίνει στην βέλτιστη λύση κοντά στο 0.