Reinforcement Learning and Dynamic Optimization

Assignment 3 (2024) (Project 2) - Stock Portfolio Optimization

Τσικριτζάκης Γεώργιος – Μάριος, ΑΜ: 2020030055
Κουτσοβασίλης Βασίλειος, ΑΜ: 2020030137

Task 1

Assume your agent knows all environment parameters (the current state of every stock, the transition probabilities and expected rewards at each state, etc.). In other words, assume this is an MDP problem. Write down the MDP components: State Space S, Action Space A, Transition Probabilities P(s,a,s'), Rewards R(s,a,s'), terminal states (if any).

- **State Space S**:

  Επειδή υπάρχουν N διαθέσιμα stocks να επενδύσουμε , ο χώρος καταστάσεων πρέπει να περιέχει $N * 2^N$ καταστάσεις. Κάθε κατάσταση πρέπει να περιέχει τα state των stocks , και ποιο stock επενδύουμε εμείς : $\{\boldsymbol{current\ stock}, \boldsymbol{State\ of\ all\ stocks}\}$.Το πρώτο μέρος της κατάστασης το χρειαζόμαστε για να ξέρουμε πότε θα υπάρχει transaction fee επειδή αλλάξαμε μετοχή , και το δεύτερο το χρειαζόμαστε για να ξέρουμε ποια θα είναι η σωστή πιθανότητα μετάβασης ,πολλαπλασιάζοντας τις επιμέρους πιθανότητες μετάβασης κατάστασης των stocks.
  Για παράδειγμα μερικές καταστάσεις είναι :

  $$\{2, (1H, 2L, 3L \dots., NH)\}$$
  $$\{1, (1H, 2H, 3H \dots. NH\ )\}$$
  $$\{4\ , (1H, 2L, 3H \dots. NL)\}$$

- **Action Space S**:

  Σε κάθε state , έχουμε N δυνατά actions . Μπορούμε να το αναπαραστήσουμε ώς το σύνολο $A = \{1\ , 2\ , 3\ , \dots , N\}$ που δείχνει ποιο stock θα αγοράσουμε στη συνέχεια.

- **Transition probabilities P(s,a,s') :**

  Οι πιθανότητες μεταβάσεις εξαρτώνται άμεσα από το Markov chain του κάθε stock.Το action που θα επιλέξει ο agent θα επηρεάσει μόνο το πρώτο μέρος της κατάστασης , δηλαδή το stock που θα επενδύσουμε , αλλά όχι τα state των stock. Μερικά παραδείγματα πιθανοτήτων μεταβάσεων:

  - Έστω ότι βρισκόμαστε στην κατάσταση $\{2, (1H, 2L, 3L \dots., NH)\}$ και το action που δίνεται είναι το a = 3 .Τότε έχουμε 2^N πιθανές μεταβάσεις.

    Η επόμενη κατάσταση είναι η $\{3, (1L, 2L, 3L \dots., NL)\}$ με
    $$P = {P_{HL}}^1 \cdot {P_{LL}}^2 \cdot {P_{LL}}^3 \dots \cdot {P_{HL}}^N$$
    $$\vdots$$
    Η επόμενη κατάσταση είναι η $\{3, (1H, 2L, 3L \dots., NH)\}$ με
    $$P = {P_{HH}}^1 \cdot {P_{LL}}^2 \cdot {P_{LL}}^3 \dots \cdot {P_{HH}}^N$$
    Η επόμενη κατάσταση είναι η $\{3, (1L, 2H, 3L \dots., NL)\}$ με
    $$P = {P_{HL}}^1 \cdot {P_{LH}}^2 \cdot {P_{LL}}^3 \dots \cdot {P_{HL}}^N$$
    $$\vdots$$
    Η επόμενη κατάσταση είναι η $\{3, (1H, 2H, 3H \dots., NH)\}$ με
    $$P = {P_{HH}}^1 \cdot {P_{LH}}^2 \cdot {P_{LH}}^3 \dots \cdot {P_{HH}}^N$$

- **Rewards R(s,a,s')**

  Το κέρδος εξαρτάται μόνο από το stock που θα έχουμε στην s' και το state του , δηλαδή αν είναι H ή L . Για παράδειγμα:

  - $\{2, (\cdot,\cdot ,3L,\cdot \dots.,\cdot)\} \xrightarrow{a=3} R_t = {r_3}^H - c$ με $P = {P_{LH}}^3$ ή $R_t = {r_3}^L - c$ με $P = {P_{LL}}^3$
  - $\{N, (\cdot,\cdot,\cdot \dots., NH)\} \xrightarrow{a=N} R_t = {r_N}^H$ με $P = {P_{HH}}^N$ ή $R_t = {r_N}^L$ με $P = {P_{HL}}^N$

Ορίζοντας το κέρδος παραμετρικά έχουμε:

$$\{i, (\cdot,\cdot,\cdot, jH,\cdot,\dots,\cdot)\} \xrightarrow{yields} \begin{cases} R_t = r_j^H \ με\ P = {P_{HH}}^j \text{ ή } R_t = r_j^L \text{ με } P = {P_{HL}}^j,\ \alpha = j\ και\ i = j \\ R_t = r_j^H - c\ με\ P = {P_{HH}}^j \text{ ή } R_t = r_j^L - c \text{ με } P = {P_{HL}}^j,\ \alpha = j\ και\ i \neq j \end{cases}$$

$$\{i, (\cdot,\cdot,\cdot, jL,\cdot,\dots,\cdot)\} \xrightarrow{yields} \begin{cases} R_t = r_j^L \ με\ P = {P_{LL}}^j \text{ ή } R_t = r_j^H \text{ με } P = {P_{LH}}^j,\ \alpha = j\ και\ i = j \\ R_t = r_j^L - c\ με\ P = {P_{LL}}^j \text{ ή } R_t = r_j^H - c \text{ με } P = {P_{LH}}^j,\ \alpha = j\ και\ i \neq j \end{cases}$$

- **Terminal states**
  Τελικές καταστάσεις δεν υπάρχουν σε αυτό το πρόβλημα γιατί υπάρχει άπειρο horizon και επενδύουμε συνεχώς.

Questions To Answer

Question 1

Θεωρώντας ότι το περιβάλλον έχει N=2 μετοχές, γ = 0 και ότι $r_1^H = 2r_2^H$, θέσαμε τα κέρδη ως εξής: $r_1^H = 0.08, r_1^L = 0.01, r_2^H = 0.04, r_2^L = -0.02$. Επίσης θέσαμε τα transition probabilities ως εξής:

1. Για την πρώτη μετοχή: $P_1^{HH} = 0.9, P_1^{HL} = 0.1, P_1^{LH} = 0.1, P_1^{LL} = 0.9$
2. Για την δεύτερη μετοχή: $P_1^{HH} = 0.85, P_1^{HL} = 0.15, P_1^{LH} = 0.2, P_1^{LL} = 0.8$

Τέλος ορίσαμε το transaction fee να είναι υψηλό και ίσο με $c = 0.8$.
Επομένως για γ = 0 ο αλγόριθμος θα δώσει μεγαλύτερο βάρος στα άμεσα κέρδη και επειδή το κόστος της μεταφοράς σε άλλη μετοχή από αυτήν που είχε πριν είναι μεγάλο αναμένεται ο αλγόριθμος να επιλέγει σε κάθε κατάσταση να επενδύσει ξανά στην ίδια μετοχή καθώς π.χ.: Αν βρισκόμαστε στην κατάσταση (1, 1L, 2H). Αν επιλέξουμε να επενδύσουμε στην μετοχή 1 τότε το συνολικό κέρδος θα είναι R = 0.01. Αν επιλέξουμε να αλλάξουμε μετοχή και να επενδύσουμε στην δεύτερη τότε το κέρδος θα είναι R'=0.04-0.8=-0.76. Άρα έχουμε ότι R > R' και άρα η καλύτερη επιλογή είναι να επενδύσουμε ξανά στην πρώτη μετοχή. Όμοια συμπεράσματα προκύπτουν και στις υπόλοιπες περιπτώσεις.
Αυτό επαληθεύεται και από τον αλγόριθμο καθώς:

```
Converged in 2 iterations
Optimal value function:
[ 0.017  0.017  0.073  0.073 -0.008  0.031 -0.008  0.031]

Optimal policy:

State:['1', '1L', '2L']  , action:1
State:['1', '1L', '2H']  , action:1
State:['1', '1H', '2L']  , action:1
State:['1', '1H', '2H']  , action:1
State:['2', '1L', '2L']  , action:2
State:['2', '1L', '2H']  , action:2
State:['2', '1H', '2L']  , action:2
State:['2', '1H', '2H']  , action:2
```

Όπως φαίνεται ο αλγόριθμος επιλέγει να ξανά-επενδύσει στην ίδια μετοχή που είχε και πριν σε κάθε κατάσταση.

Question 2

Θεωρώντας τώρα το ίδιο περιβάλλον με πριν αλλά με γ = 0.9 οι παράμετροι που χρειάστηκαν τροποποίηση ήταν το κόστος μεταφοράς, όπου τέθηκε σε $c = 0.05$ καθώς και τα κέρδη στην περίπτωση που οι μετοχές ήταν σε κατάσταση Low. Τα κέρδη τέθηκαν σε: $r_1^L = -0.01, r_2^L = 0.01$. Για γ = 0.9 ο αλγόριθμος θα δώσει βάρος στα μακροχρόνια κέρδη και επειδή το κόστος μεταφοράς δεν είναι υψηλό αναμένεται σε κάποιες περιπτώσεις να επιλέξει να αλλάξει μετοχή και σε κάποιες να κρατήσει την ίδια.
Για παράδειγμα:
Αν βρισκόμαστε στην κατάσταση (1, 1L, 2L) η καλύτερη επιλογή θα είναι να επιλέξει να επενδύσει στην μετοχή 2, καθώς ναι μεν το άμεσο κέρδος θα είναι μικρότερο από το να μείνει στην ίδια μετοχή (αλλαγή: 0.01-0.05=-0.04, να μείνει: -0.01) αλλά η μετοχή 2 έχει μεγαλύτερη πιθανότητα να μεταβεί στην κατάσταση High(P=0.2) από την μετοχή 1 (P = 0.1) και άρα να έχει μακροχρόνια περισσότερα κέρδη αν μεταβεί εκεί. Επίσης το κέρδος της μετοχής 2 είναι μεγαλύτερο της 1 στην κατάσταση Low οπότε εκτός από την στιγμή μεταφοράς θα συσσωρεύει περισσότερα κέρδη από το να έμενε στην ίδια μετοχή.
Αυτό επαληθεύεται και από τον αλγόριθμο καθώς:

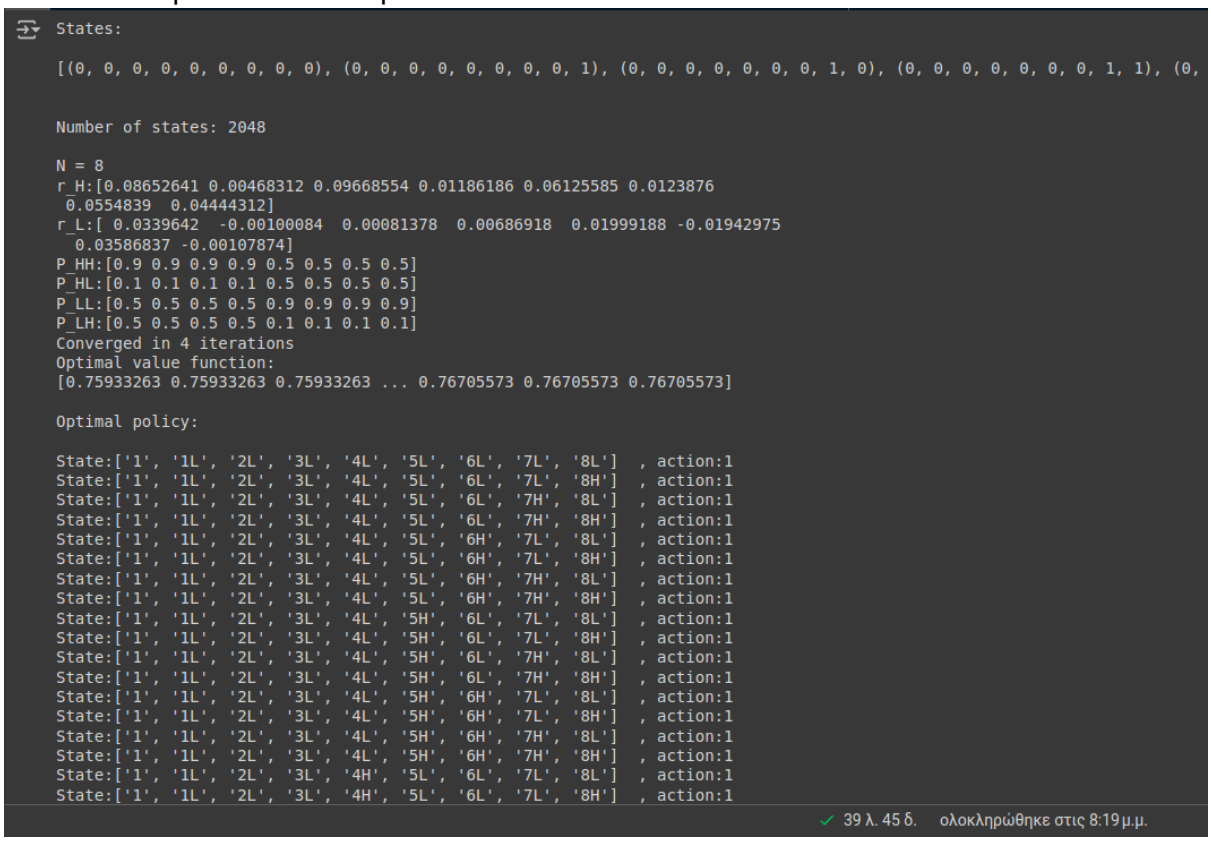
```
Converged in 3 iterations
Optimal value function:
[0.29350022 0.33532126 0.52107809 0.526245   0.34350022 0.38532126
 0.47107809 0.476245  ]

Optimal policy:

State:['1', '1L', '2L']  , action:2
State:['1', '1L', '2H']  , action:2
State:['1', '1H', '2L']  , action:1
State:['1', '1H', '2H']  , action:1
State:['2', '1L', '2L']  , action:2
State:['2', '1L', '2H']  , action:2
State:['2', '1H', '2L']  , action:1
State:['2', '1H', '2H']  , action:1
```

Όπως φαίνεται ο αλγόριθμος επιλέγει σε κάποιες καταστάσεις να παραμείνει στην ίδια μετοχή και σε κάποιες να επενδύσει σε διαφορετική. Συγκεκριμένα:

i. Στις περιπτώσεις όπου έχουμε (1L, 2L) ανεξάρτητα σε ποια μετοχή βρισκόταν θα επιλέξει την μετοχή 2.
ii. Στις περιπτώσεις όπου έχουμε (1L, 2H) ανεξάρτητα σε ποια μετοχή βρισκόταν θα επιλέξει την μετοχή 2.
iii. Στις περιπτώσεις όπου έχουμε (1H, 2L) ανεξάρτητα σε ποια μετοχή βρισκόταν θα επιλέξει την μετοχή 1.
iv. Στις περιπτώσεις όπου έχουμε (1H, 2H) ανεξάρτητα σε ποια μετοχή βρισκόταν θα επιλέξει την μετοχή 1.

Question 3

Δοκιμάζοντας τον αλγόριθμο για διάφορες τιμές του N παρατηρήσαμε ότι για N =1,2,3,4,5,6,7 ο χρόνος εκτέλεσης κυμαίνονταν μεταξύ δευτερολέπτων και μερικών λεπτών. Όμως δοκιμάζοντας N = 8 είδαμε ότι ο χρόνος εκτέλεσης ήταν περίπου σαράντα λεπτά και μεγαλώνοντας το N σε τιμές μεγαλύτερες του 8 ξεπερνούσαμε την μία ώρα. Αυτό φανερώνει ότι για N μεγαλύτερο του 8 το μέγεθος του προβλήματος αρχίζει να καθυστερεί κατά πολύ τον υπολογισμό της βέλτιστης πολιτικής καθώς το σύνολο των καταστάσεων που πρέπει να εξεταστούν είναι πάρα πολύ μεγάλο και αυξάνεται εκθετικά. Επομένως σε τέτοιες περιπτώσεις χρειαζόμαστε κάτι καλύτερο για να μπορέσουμε να χειριστούμε τόσο μεγάλα State Spaces.
Το αποτέλεσμα του κώδικα για N = 8 είναι:



Όπως φαίνεται παρόλο τις καθυστερήσεις ο αλγόριθμος καταφέρνει να βρει την βέλτιστη πολιτική για κάθε κατάσταση.