7. Βαθιά Ενισχυτική Μάθηση και Γενετικοί Αλγόριθμοι

***Σύνοψη**: Στο έβδομο κεφάλαιο θα παρουσιάσουμε αλγορίθμους **ενισχυτικής μάθησης** (**reinforcement learning**) -διότι ενισχύει επιθυμητές συμπεριφορές επιβραβεύοντάς τις-, όπου το σύστημα συστάσεων προσπαθεί να μάθει μέσα από τη διάδρασή του με τον χρήστη. Ακόμη, θα αναλύσουμε αλγορίθμους συστάσεων (**Markov Chains, Markov Decision Processes,** κ.λπ.) που βασίζονται σε δεδομένα χρονικώς εξελισσόμενα, τα οποία δηλαδή έχουν μια αλληλουχία και βρίσκονται σε σειριακή διάταξη. Επίσης, θα μελετήσουμε τους αλγορίθμους **Advantage Actor-Critic learning** (**A2C**) και **Deep Q-learning Network** (**DQN**), οι οποίοι είναι αλγόριθμοι ενισχυτικής μάθησης που συνδυάζουν μεθόδους της με νευρωνικά δίκτυα πολλών επιπέδων. Τέλος, θα περιγράψουμε τα συστήματα συστάσεων που βασίζονται σε **γενετικούς αλγορίθμους** (**genetic algorithms**), οι οποίοι προσομοιώνουν το φυσικό φαινόμενο της **εξέλιξης** και της **φυσικής επιλογής**: η αναζήτηση λοιπόν της κατάλληλης γειτονιάς του υπό εξέταση χρήστη ξεκινάει με έναν αριθμό τυχαίων γειτονικών χρηστών βάσει ενός συνόλου αρχικών παραδοχών. Επί αυτού του αρχικού πληθυσμού, και εφόσον τα μέλη του αξιολογηθούν με τη βοήθεια μιας **συνάρτησης καταλληλότητας** (**fitness function**), παράγεται η νέα γενιά γειτόνων μέσω διαδικασιών αναπαραγωγής (π.χ. **διασταύρωσης, μετάλλαξης,** κ.λπ.).*

***Προαπαιτούμενη γνώση**: Ο αναγνώστης προτείνεται να έχει μελετήσει προηγουμένως τη σχετική θεωρία των **Μαρκοβιανών αλυσίδων** καθώς και αυτή των **διαδικασιών αποφάσεων**.*

7.1 Μοντέλο Μαρκοβιανής Αλυσίδας

Στην ενότητα αυτή θα αναφερθούμε στο *μοντέλο της Μαρκοβιανής αλυσίδας* (*Markov Chain Model*). Η MCM περιγράφει ένα μαθηματικό σύστημα που βρίσκεται σε συγκεκριμένη κάθε φορά *κατάσταση* (*state*) ανά βήμα, όπου η κατάσταση όμως αυτή μεταβάλλεται τυχαία μεταξύ των βημάτων, τα οποία συχνά θεωρούνται ως στιγμές μέσα στον χρόνο.

Στην περίπτωση των συστημάτων συστάσεων, ως *κατάσταση της μαρκοβιανής αλυσίδας* θεωρούμε το στοιχείο με το οποίο αλληλεπιδρά ο χρήστης μία συγκεκριμένη χρονική στιγμή, ενώ το

επόμενο βήμα της *μαρκοβιανής αλυσίδας* είναι το στοιχείο με το οποίο είναι πιθανό να αλληλεπιδράσει ο χρήστης αμέσως μετά. Σε πολλές ερευνητικές εργασίες προτάθηκαν μέθοδοι βασισμένες σε *Markov Chain Models* (*MCMs*) (βλ. [Chen et al., 2012], [Esiyok et al., 2014], [Garcin et al., 2013], [He et al., 2009]). *Η Μαρκοβιανή αλυσίδα* είναι μια στοχαστική διαδικασία πιθανών γεγονότων που ικανοποιεί την *ιδιότητα Markov*, η οποία δηλώνει ότι η πιθανότητα να συμβεί κάποιο γεγονός εξαρτάται μόνο από την εκάστοτε παρούσα κατάσταση και όχι από τις προηγούμενες. Ένας από τους τρόπους με τους οποίους μπορεί να αναπαρασταθεί το *μοντέλο αλυσίδας Markov* $\{X\}$ είναι ο πίνακας μετάβασης $P_{i,j}$, όπου κάθε του γραμμή εμπεριέχει την πιθανότητα μετάβασης από μία κατάσταση σε οποιαδήποτε άλλη, σύμφωνα με την Εξίσωση 7.1:

$$P_{i,j} = \mathbb{P}(X_{t+1} = x_j | X_t = x_i) \tag{7.1}$$

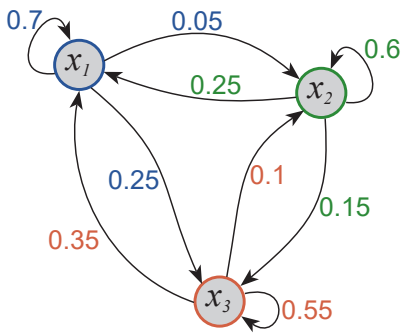
Κάθε σειρά του πίνακα $P_{i,j}$ είναι ένα διάνυσμα πιθανότητας και το άθροισμα των συνιστωσών του είναι ίσο με 1. Για παράδειγμα, παρακάτω δίνεται ένας *πίνακας μετάβασης* (*transition matrix*) που αντιπροσωπεύει ένα μοντέλο με τρεις πιθανές καταστάσεις:

$$P = \begin{bmatrix} 0,7 & 0,05 & 0,25 \\ 0,25 & 0,6 & 0,15 \\ 0,35 & 0,1 & 0,55 \end{bmatrix}$$

Εάν η τρέχουσα κατάσταση της διαδικασίας είναι x_2, δηλαδή λαμβάνεται υπόψη η δεύτερη σειρά του πίνακα, τότε η πιθανότητα μετάβασης στην κατάσταση x_3 ισούται με $P(X_{t+1} = x_3 \mid X_t = x_2) = 0,15$.

Ένας άλλος τρόπος αναπαράστασης ενός *μοντέλου Μαρκοβιανής αλυσίδας* είναι μ' ένα *διάγραμμα μετάβασης*, δηλαδή ένα *σταθμισμένο κατευθυνόμενο γράφημα* όπου κάθε κορυφή αντιπροσωπεύει μια κατάσταση της προαναφερθείσας αλυσίδας, και υπάρχει μια κατευθυνόμενη ακμή από την κορυφή x_i στην κορυφή x_j εάν η πιθανότητα μετάβασης είναι μεγαλύτερη του μηδενός $P_{i,j} > 0$. Ένα παράδειγμα τέτοιου διαγράμματος απεικονίζεται στο Σχήμα 7.1.

Ο Rendle και οι λοιποί συν-συγγραφείς [Rendle et al., 2010] χρησιμοποίησαν *εξατομικευμένες Μαρκοβιανές αλυσίδες* ώστε, δοθέντος του ιστορικού των ηλεκτρονικών αγορών του καταναλωτή-χρήστη, να λύσουν το πρόβλημα της πρόβλεψης των προϊόντων που αυτός θα βάλει στο καλάθι του κατά την επόμενη διαδικτυακή περιήγησή του. Περιορίζοντας μάλιστα το μέγεθος του καλαθιού σ' ένα μόνο προϊόν και λαμβάνοντας υπόψη την τρέχουσα συνεδρία ενός χρήστη (user session), η μέθοδός τους μπορεί να εφαρμοστεί άμεσα για την *πρόβλεψη του επόμενου κλικ* (*next click prediction*) στις ιστοσελίδες που επισκέπτονται οι χρήστες. Με το ίδιο σκεπτικό, ο Esiyok και οι συν-συγγραφείς του [Esiyok et al., 2014] πρότειναν μια μελέτη για τη μέτρηση της συμπεριφοράς των χρηστών σ' ένα σύστημα συστάσεων ειδησεογραφίας. Δημιούργησαν λοιπόν ένα *μοντέλο Μαρκοβιανής αλυσίδας* βασισμένο στο σύνολο δεδομένων *PLISTA* [Tavakolifard et al., 2013] και ανα-



Σχήμα 7.1: Παράδειγμα διαγράμματος μετάβασης σε μια *Μαρκοβιανή αλυσίδα*. Υπάρχουν τρεις *καταστάσεις* $\{x_1, x_2, x_3\}$ του μοντέλου. Οι αριθμοί που βρίσκονται δίπλα στις ακμές δείχνουν την πιθανότητα μετάβασης στη νέα *κατάσταση*.

ζήτησαν μοτίβα στις προτιμήσεις των αναγνωστών ως προς τις κατηγορίες των ειδησεογραφικών άρθρων με τα οποία αυτοί αλληλεπιδρούν.

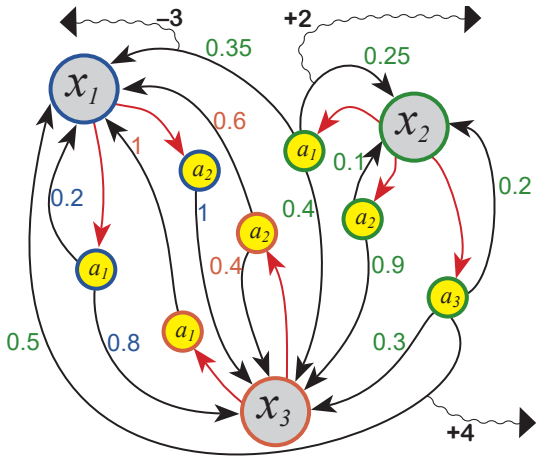
7.1.1 Μαρκοβιανές Διαδικασίες Αποφάσεων

Οι *Μαρκοβιανές Διαδικασίες Αποφάσεων* (*Markov Decision Processes - MDPs*) αποτελούν μια επέκταση του *μοντέλου της Μαρκοβιανής αλυσίδας* με την έννοια ότι σε κάθε χρονική στιγμή t, όταν η διαδικασία βρίσκεται σε κάποια κατάσταση x_t, ο λήπτης απόφασης μπορεί να επιλέξει οποιαδήποτε ενέργεια $a \in A_{x_t}$ που είναι διαθέσιμη στην κατάσταση x_t. Η διαδικασία τότε μεταβαίνει τυχαία σε μια νέα κατάσταση x_{t+1} και δίνει στον αποφασίζοντα μια αντίστοιχη *ανταμοιβή* (*reward*) $R(x_t, a, x_{t+1})$ [Shani et al., 2005]. Η επιλεγμένη ενέργεια επηρεάζει την πιθανότητα η διαδικασία να μεταβεί στη νέα της κατάσταση x_{t+1}. Ωστόσο, η πιθανότητα μετάβασης στη νέα κατάσταση x_{t+1} εξαρτάται μόνο από την τρέχουσα κατάσταση x_t και την επιλεγμένη ενέργεια a, και ως εκ τούτου είναι ανεξάρτητη από όλες τις προηγούμενες καταστάσεις και ενέργειες, δηλαδή $P(x_{t+1} \mid x_t, a)$. Συνεπώς, ικανοποιείται και στην περίπτωση αυτή η *ιδιότητα Markov*.

Ο στόχος λοιπόν της *Μαρκοβιανής διαδικασίας απόφασης* είναι να βρεθεί η σωστή *πολιτική* (*policy*) π -το θεωρητικό δηλαδή πλαίσιο που θα καθορίσει και τις μετέπειτα δράσεις- η οποία θα αντιστοιχίζει κάθε κατάσταση x_n με κάθε δυνατή ενέργεια a που είναι διαθέσιμη γι' αυτήν την κατάσταση ώστε να μεγιστοποιείται σωρευτικά η *ανταμοιβή*, ως ακολούθως:

$$\max_a \sum_{t=0}^{\infty} R(x_t, a, x_{t+1}) \tag{7.2}$$

Εφόσον μια *Μαρκοβιανή διαδικασία απόφασης* συνδυαστεί με μια *πολιτική* π, τότε καθορίζεται η ενέργεια που αντιστοιχεί στην υπό εξέταση κατάσταση, δηλαδή $\pi(x_n) \rightarrow a$. Η *Μαρκοβιανή διαδικασία απόφασης* μπορεί επίσης να αναπαρασταθεί μ' ένα διάγραμμα μετάβασης όπως του Σχήματος 7.2.



Σχήμα 7.2: Ένα παράδειγμα διαγράμματος μετάβασης της *Μαρκοβιανής διαδικασίας λήψης απόφασης*.

Λόγω όμως του ότι το παράδειγμα του Σχήματος 7.2 έχει μια σχετική πολυπλοκότητα, θα κάνουμε μια σύντομη περιγραφή των στοιχείων του. Υπάρχουν λοιπόν τρεις καταστάσεις του μοντέλου: x_1, x_2, x_3. Οι διαφορετικές καταστάσεις έχουν και διαφορετικές πιθανές ενέργειες a_n. Οι αριθμοί με χρώμα, που βρίσκονται δίπλα στις ακμές, δείχνουν την πιθανότητα αλλαγής στη νέα κατάσταση. Οι κόκκινες γραμμές με τους κίτρινους κύκλους δείχνουν την ενέργεια του αποφασίζοντος, ενώ οι κυματιστές γραμμές και οι αριθμοί με μαύρο χρώμα δείχνουν την αντίστοιχη "ανταμοιβή/τιμωρία" που δίνεται στον αποφασίζοντα βάσει της ενέργειας που επέλεξε.

Εν συνεχεία, αφού ληφθεί μια απόφαση ή γίνει κάποια ενέργεια, το σύστημα μεταβαίνει με μια ορισμένη πιθανότητα σε μια νέα κατάσταση. Οι μεταβάσεις έχουν είτε τιμές *ανταμοιβής* (*reward*) είτε *τιμωρίας* (*penalty*), οι οποίες στο σχήμα μας συνδέονται με κυματιστές γραμμές και θετικούς ή αρνητικούς αριθμούς αντίστοιχα. Με άλλα λόγια, η ανταμοιβή μπορεί να είναι μια θετική τιμή (επιβραβεύοντας μια ενέργεια του αποφασίζοντος) ή μια αρνητική τιμή ("τιμωρώντας" μια ενέργειά του). Μόλις επιτευχθεί ο στόχος της *Μαρκοβιανής διαδικασίας απόφασης*, κάθε κατάσταση θα οδηγεί σε μία μόνο βέλτιστη ενέργεια και αυτό θα γίνεται με τρόπο που θα μεγιστοποιεί την ανταμοιβή του αποφασίζοντος.

Ακόμη, στα συστήματα συστάσεων δημοσιεύθηκε [Moling et al., 2012] μια μέθοδος που χρησιμοποιεί *Μαρκοβιανές διαδικασίες λήψης αποφάσεων* αλλά εκτελείται στη συσκευή του χρήστη αντί για έναν απομακρυσμένο διακομιστή. Η προτεινόμενη μέθοδος παρέχει συστάσεις ειδήσεων και νέων σε αναγνώστες διαφορετικών παρόχων περιεχομένου. Κατά τους συγγραφείς, η μέθοδός τους είχε σημαντικά καλύτερες επιδόσεις από το αντίστοιχο απλό *μοντέλο Μαρκοβιανής αλυσίδας*.

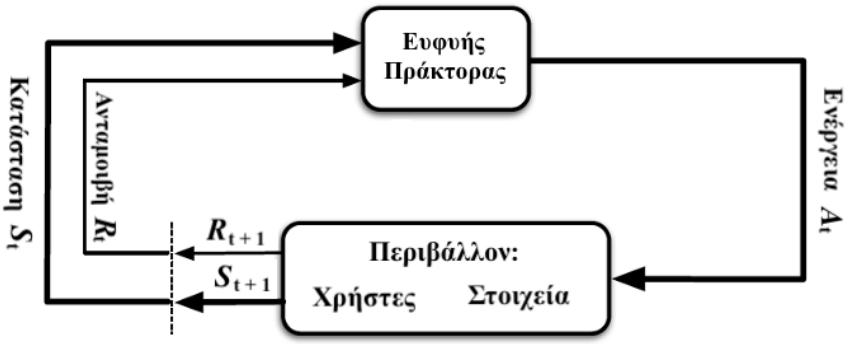
7.2 Αλγόριθμοι Ενισχυτικής Μάθησης

Η *ενισχυτική μάθηση* είναι ένα σύνολο αλγορίθμων *μηχανικής μάθησης* και κατατάσσεται μεταξύ της *μάθησης με επίβλεψη* (*supervised learning*) και της *μάθησης χωρίς επίβλεψη* (*unsupervised learning*). Στις δύο προηγούμενες υποενότητες του κεφαλαίου μελετήθηκαν αλγόριθμοι λήψης αποφάσεων που λειτουργούν σε κλειστά "ντετερμινιστικά" συστήματα, όπου δηλαδή ο αριθμός των *καταστάσεων* (*states*) και ο αριθμός των *ενεργειών* (*actions*) είναι αυστηρά *προκαθορισμένος* (*Finite Markov Decision Processes*). Παρόλα αυτά, όπως και στην πραγματική ζωή, έτσι και στα συστήματα συστάσεων υπάρχουν προβλήματα όπου τόσο ο αριθμός των καταστάσεων (π.χ. οι διαφορετικές αλληλεπιδράσεις των χρηστών με τα στοιχεία) όσο και ο αριθμός των ενεργειών (π.χ. ο αριθμός των στοιχείων που είναι υποψήφια προς σύσταση) είναι πάρα πολύ μεγάλος. Σε τέτοια πολύπλοκα στοχαστικά περιβάλλοντα εφαρμόζονται οι *αλγόριθμοι ενισχυτικής μάθησης*, οι οποίοι δεν έχουν ρητά προσδιορισμένες πιθανότητες μετάβασης από μία κατάσταση σε μια άλλη και ενσωματώνουν τα δύο παρακάτω βασικά χαρακτηριστικά:

- Ένα περιβάλλον: Αυτό θα μπορούσε να είναι ένα ολοκληρωμένο σύστημα όπως ένα βιντεοπαιχνίδι, το χρηματιστήριο κ.λπ. Στα συστήματα συστάσεων το περιβάλλον είναι τα προφίλ των χρηστών και των στοιχείων, όπως φαίνεται και στο Σχήμα 7.3.

- Ένας *ευφυής πράκτορας*: Πρόκειται για μια οντότητα λογισμικού που μαθαίνει πώς να λειτουργεί και να είναι αποτελεσματική σ' ένα δεδομένο περιβάλλον.

Αναλυτικότερα, ο *ευφυής πράκτορας* βασίζεται σ' έναν *επαναληπτικό βρόχο ανατροφοδότησης* παρόμοιο μ' αυτόν που εμφανίζεται στο Σχήμα 7.3:



Σχήμα 7.3: Η αλληλεπίδραση μεταξύ *ευφυούς πράκτορα* και *περιβάλλοντος* στα συστήματα συστάσεων.

Όπως φαίνεται στο Σχήμα 7.3, σε κάθε χρονική στιγμή t που ένας χρήστης αλληλεπιδρά μ' ένα στοιχείο, μια κατάσταση S_t, η οποία αναπαριστά το προφίλ του χρήστη, διαβιβάζεται στον *ευφυή πράκτορα*, ο οποίος μπορεί να επιλέξει οποιαδήποτε ενέργεια A_t (δηλαδή να επιλέξει να προτείνει ένα οποιοδήποτε υποψήφιο στοιχείο) που είναι διαθέσιμη για την κατάσταση S_t. Έπειτα, το περιβάλλον μεταβαίνει σε μια νέα κατάσταση S_{t+1} και δίνει στον *ευφυή πράκτορα* μια άμεση *ανταμοιβή*

(*reward*) R_t για την επιλογή της ενέργειας A_t, αλλά και μια μελλοντική R_{t+1} *ανταμοιβή* για την ίδια ενέργεια. Κατ' αυτόν τον τρόπο, ο *ευφυής πράκτορας* δοκιμάζει διαφορετικές ενέργειες σε διαφορετικές καταστάσεις μέσω της *διαδικασίας δοκιμής και σφάλματος* (*trial and error*), και έτσι μαθαίνει σταδιακά (μέσα από τα σφάλματά του) την αναμενόμενη ανταμοιβή για κάθε ζεύγος κατάστασης-ενέργειας. Η χρήση της *διαδικασίας δοκιμής και μέτρησης του σφάλματος*, ώστε να μάθει ο *ευφυής πράκτορας* το περιβάλλον με το οποίο αλληλεπιδρά, ονομάζεται **Εξερεύνηση (Exploration)**. Εφόσον λοιπόν ο *ευφυής πράκτορας* εντοπίσει τη βέλτιστη ανταμοιβή ενός ζεύγους κατάστασης-ενέργειας, τότε ο ίδιος, όντας σε μια κατάσταση S, μεγιστοποιεί την αναμενόμενη ανταμοιβή του επιλέγοντας την ενέργεια A με την προαναφερθείσα υψηλότερη αναμενόμενη ανταμοιβή R. Η ρητή επιλογή της καλύτερης δυνατής ενέργειας σε μια κατάσταση ονομάζεται **Εκμετάλλευση (Exploitation)**.

Σ' αυτό το σημείο, ορίζουμε ως **πολιτική (policy)** την αντιστοίχιση μεταξύ των *καταστάσεων* (*states*) του περιβάλλοντος και των *ενεργειών* (*actions*) που πρέπει να εκτελέσει ο *ευφυής πράκτορας* όταν του διαβιβάζονται αυτές οι καταστάσεις από το περιβάλλον. Στα συστήματα συστάσεων οι μέθοδοι για την *ενισχυτική μάθηση* αποτελούνται συνήθως από τα ακόλουθα στοιχεία:

S: Το σύνολο των *καταστάσεων* (*states*) που είναι δυνατόν να συμβούν. Κάθε φορά δηλαδή που ένας χρήστης αλληλεπιδρά μ' ένα στοιχείο, η κατάστασή του αλλάζει δεδομένου ότι προστίθεται μια νέα διάδραση. Αυτή η επαυξημένη κατάσταση του χρήστη ορίζεται και ως το προφίλ του.

A: Το σύνολο των *ενεργειών* (*actions*) που είναι δυνατόν να πραγματοποιηθούν. Πρόκειται λοιπόν για το πλήθος των υποψήφιων προς σύσταση στοιχείων.

T: $S \times A \rightarrow S'$: Είναι η *συνάρτηση υπολογισμού της πιθανότητας μετάβασης* (*transition function*) από μια κατάσταση S σε μια νέα κατάσταση S' επιλέγοντας μια συγκεκριμένη *ενέργεια* (*action*).

R: $S \times A \rightarrow \mathbb{R}$: Είναι η *συνάρτηση επιβράβευσης* (*reward function*). Έτσι, υπολογίζει την *επιβράβευση* (*reward*) για κάθε δυνατό ζευγάρι {*state, action*}.

π: $S \rightarrow A$: Είναι η λεγόμενη *πολιτική* π, η οποία είναι μια *συνάρτηση αντιστοίχισης* (*mapping function*) που υποδεικνύει ποια ενέργεια πρέπει να γίνει σε κάθε κατάσταση.

Στη συνέχεια παραθέτουμε τον Πίνακα 7.1 που συγκεντρώνει τα σημαντικότερα σύμβολα που χρησιμοποιούνται στην *ενισχυτική μάθηση*, ενώ ταυτόχρονα εμπεριέχει και μια σύντομη περιγραφή τους. Ο κύριος στόχος της *επαναληπτικής διαδικασίας δοκιμής και ελέγχου του σφάλματος* είναι να υπολογιστεί η *βέλτιστη πολιτική* π^*, η οποία μεγιστοποιεί τις σωρευτικές ανταποδώσεις κατά τη λειτουργία του *ευφυούς πράκτορα*. Οι *αλγόριθμοι ενισχυτικής μάθησης* χωρίζονται σε δύο κατηγορίες, *on-policy* και *off-policy*:

- Οι *αλγόριθμοι (εκ)μάθησης on-policy* αξιολογούν και βελτιώνουν την ίδια πάντα *πολιτική* (*policy*), η οποία χρησιμοποιείται επίσης και για την επιλογή των *τρεχουσών ενεργειών* (*current actions*).

- Οι *αλγόριθμοι (εκ)μάθησης off-policy* μπορούν να αξιολογούν και να βελτιώνουν ακόμη περισσότερο τη λεγόμενη *βέλτιστη πολιτική* (*optimal policy*), και η οποία είναι διαφορετική από την *πολιτική* που χρησιμοποιείται για την επιλογή των *τρεχουσών ενεργειών* (*current actions*).

Σύμβολα	Περιγραφή
$\mathcal{I}$	το σύνολο των στοιχείων
$\mathcal{A}, a$	το σύνολο των ενεργειών $\mathcal{A} \in \{0,1\}^{\mathcal{I}}, a \in \mathcal{A}, a_i \in \{0,1\}:$ υποδεικνύει εάν το i-οστό στοιχείο προτάθηκε σ' έναν χρήστη
$\mathcal{S}, s$	το σύνολο των καταστάσεων $\mathcal{S}, s \in \mathcal{S}$: αποτελείται από τα στοιχεία με τα οποία αλληλεπίδρασε ο κάθε χρήστης (δηλ. το προφίλ του χρήστη)
$\mathcal{R}, r$	η συνάρτηση ανταμοιβής: εάν το προτεινόμενο στοιχείο αρέσει στον χρήστη, r = +1, διαφορετικά r = −1
γ	$\gamma \in [0,1]$ είναι ένας συντελεστής εξασθένισης για την εξισορρόπηση της σημασίας των άμεσων και των μελλοντικών ανταμοιβών
$\pi^*_\theta(a \mid s)$	η πολιτική που μαθαίνεται με βάση μια ακολουθία γεγονότων θ, επιλογή της ενέργειας a όταν βρισκόμαστε στην κατάσταση s
$Q(s,a)$	η αξία – Q μιας ενέργειας (action's Q-value), η προσδοκώμενη ανταμοιβή στην κατάσταση s όταν επιλέγεται η ενέργεια a
$V(s)$	η αξία της κατάστασης (state value), η προβλεπόμενη μέγιστη ανταμοιβή από την κατάσταση s μετά την πραγματοποίηση της ενέργειας a

Πίνακας 7.1: Τα σημαντικότερα σύμβολα στην *ενισχυτική μάθηση*.

Ένα πλεονέκτημα των *αλγορίθμων off-policy* -όπως οι *Q-learning* και *Deep Q-Network* (*DQN*) που θα περιγράψουμε παρακάτω- είναι ότι κατά τη διάρκεια της εκμάθησης μπορούν να *εξερευνούν* (*exploration*) συνεχώς αναζητώντας τη *βέλτιστη πολιτική* π^*. Τα μειονεκτήματα τους είναι ότι είναι ασταθείς και δεν έχουν γρήγορη σύγκλιση, επειδή αρχικά μαθαίνουν μια προσέγγιση της *συνάρτησης επιβράβευσης* (*reward function*), η οποία στη συνέχεια χρησιμοποιείται για την εύρεση της αντίστοιχης *βέλτιστης πολιτικής* π^* (*state-action*).

Οι *αλγόριθμοι on-policy* -όπως ο *SARSA (State-Action-Reward-State'-Action')*- από την άλλη μεριά, δεδομένου ότι βελτιστοποιούν άμεσα την *τρέχουσα πολιτική*, τείνουν να είναι πολύ σταθεροί στις προβλέψεις τους και να συγκλίνουν γρήγορα στην εύρεση μιας συνήθως *βελτιωμένης* αλλά όχι *βέλτιστης πολιτικής*.

7.3 Ο Αλγόριθμος Q-learning

Ο *αλγόριθμος Q-learning* χρησιμοποιεί έναν *ευφυή πράκτορα* που προσπαθεί να μάθει μέσα από την άμεση διάδραση μ' ένα περιβάλλον. Πιο συγκεκριμένα, ο *ευφυής πράκτορας* συλλέγει πληροφορίες από το περιβάλλον (δηλαδή τους χρήστες και τα στοιχεία στα συστήματα συστάσεων) με σκοπό να το κατανοήσει και να λάβει το μέγιστο κέρδος ανταμοιβής σε σχέση με τις ενέργειες (τις συστάσεις στοιχείων) που πραγματοποιεί μέσα σ' αυτό. Η λήψη της ανταμοιβής είναι το μοναδικό ερέθισμα που λαμβάνει ο *ευφυής πράκτορας* από τη σύνδεσή του με το περιβάλλον. Κατόπιν, ο ίδιος, ανάλογα με την τρέχουσα κατάσταση (π.χ. την αλληλεπίδραση ενός χρήστη μ' ένα στοιχείο) που του διαβιβάζει το περιβάλλον, ενεργεί προσπαθώντας να πετύχει τους στόχους του (π.χ. προτείνει στον χρήστη παρόμοια στοιχεία μ' αυτό με το οποίο έχει μόλις αλληλεπιδράσει). Το κατά πόσο

λοιπόν ο *ευφυής πράκτορας* πετυχαίνει τους στόχους του ή όχι εκφράζεται από ένα *αριθμητικό σήμα ενίσχυσης ή αλλιώς επιβράβευσης* (*reinforce or reward signal*). Εν τέλει, ο *ευφυής πράκτορας* δρα με τρόπο τέτοιο ώστε να μεγιστοποιεί το άθροισμα των *σημάτων επιβράβευσης* κατά τη διάρκεια της λειτουργίας του.

Επιπρόσθετα, ο προαναφερθείς αλγόριθμος (*Q-learning*) χρησιμοποιεί μια συνάρτηση, γνωστή ως *συνάρτηση Q*, για να υπολογίσει την ανταμοιβή ή αλλιώς *αξία* (*Q-value*) κάθε ενέργειας a δεδομένης μιας κατάστασης s. Ο στόχος είναι να βρεθεί μια *πολιτική* που να μεγιστοποιεί την ανταμοιβή:

$$\pi^*(a \mid s) = \underset{a}{\operatorname{argmax}}\, Q(s, a).$$

Επομένως, καθώς πρόκειται για μια διαδικασία *ενισχυτικής μάθησης* χρησιμοποιούμε τη *συνάρτηση Q*, η οποία υπολογίζει την ανταμοιβή της ενέργειας a στην κατάσταση s για μια χρονική στιγμή t. Με τη χρήση ενός *ευφυούς πράκτορα* προσπαθούμε επί της ουσίας να προσομοιώσουμε τη συμπεριφορά και τις ενέργειες ενός πραγματικού ανθρώπου που προτείνει στοιχεία σ' έναν φίλο του. Εάν το προβλεπόμενο στοιχείο είναι σωστό (εάν δηλαδή αρέσει στον χρήστη), η ανταμοιβή του *ευφυούς πράκτορα* είναι ίση με 1, διαφορετικά "τιμωρούμε" τον *ευφυή πράκτορα* δίνοντάς του αρνητική βαθμολογία ίση με -1. Συνεπώς, όπως έχει ήδη περιγραφεί, στόχος μας είναι να βρούμε μια *πολιτική* που να μεγιστοποιεί σωρευτικά την *ανταμοιβή* R σύμφωνα με την παρακάτω εξίσωση:

$$R = \max \sum_{t=0}^{\infty} \gamma_t \cdot r_t \tag{7.3}$$

όπου γ ο *συντελεστής εξασθένισης των ανταμοιβών* -για να χάνουν την αξία τους με την πάροδο του χρόνου, ώστε οι πιο άμεσες χρονικά ανταμοιβές να αποτιμώνται περισσότερο-, r η ανταμοιβή και t διαφορετικά χρονικά σημεία. Σημειώστε ότι η *συνάρτηση Q* υπακούει στην *αρχή βελτιστοποίησης της εξίσωσης Bellman*, η οποία αναφέρεται στην άμεση και τη μελλοντική ανταμοιβή, και χαρακτηρίζει τη βέλτιστη συμπεριφορά του *ευφυούς πράκτορα* σύμφωνα με την Εξίσωση 7.4:

$$Q(s_t, a_t) = r(s_t, a_t) + \gamma \cdot \max_a Q(s_{t+1}, a) \tag{7.4}$$

όπου s_t η τρέχουσα κατάσταση του περιβάλλοντος, a_t η ενέργεια που επιλέγει ο *ευφυής πράκτορας* στην τρέχουσα κατάσταση (π.χ. το στοιχείο που ο ίδιος προτείνει σ' έναν χρήστη), γ *ο συντελεστής εξασθένισης για τις μελλοντικές ανταμοιβές*, και $\max_a Q(s_{t+1}, a)$ η *μέγιστη προσδοκώμενη ανταμοιβή* όταν ο *ευφυής πράκτορας* βρίσκεται στην κατάσταση s_{t+1}.

Η διαφορά μεταξύ των δύο μελών της ισότητας της Εξίσωσης 7.4 είναι γνωστή ως *μάθηση χρονικής διαφοράς* ή αλλιώς *σφάλμα χρονικής διαφοράς*:

$$\delta_t = r(s_t, a_t) + \gamma \cdot \max_a Q(s_{t+1}, a) - Q(s_t, a_t).$$

Το *σφάλμα χρονικής διαφοράς* δ_t είναι μια σημαντική έννοια και βρίσκεται στην καρδιά του αλγορίθμου *Q-learning*. Η βασική ιδέα πίσω από αυτό είναι ότι οι τρέχουσες εκτιμήσεις των *Q-αξιών* βασίζονται εν μέρει σε μεταγενέστερες χρονικά εκτιμήσεις των *Q-αξιών* χωρίς να αναμένουν προηγουμένως την ύπαρξη ενός τελικού αποτελέσματος. Με την *ελαχιστοποίηση του σφάλματος της χρονικής διαφοράς* επιδιώκουμε να ενημερώνεται κατάλληλα η πρόβλεψη σε μία χρονική στιγμή, ώστε να έρθει πιο κοντά στην πρόβλεψη της ίδιας ποσότητας στην επόμενη χρονική στιγμή. Αναλυτικότερα, οι δύο πρώτοι όροι της παραπάνω εξίσωσης εκφράζουν τη χρονικώς νέα *αξία-Q* μιας ενέργειας (δηλαδή την ανταμοιβή που τελικά εισπράξαμε), ενώ ο τρίτος όρος της παραπάνω εξίσωσης εκφράζει τη χρονικώς παλιά *αξία-Q* μιας ενέργειας (δηλαδή την ανταμοιβή που προβλέπαμε ότι θα εισπράξουμε). Συνοψίζοντας, η *μάθηση (ή σφάλμα) χρονικής διαφοράς* βασίζεται στην αφαίρεση της *Q-αξίας* που αφορούσε στη χρονική στιγμή t από την *Q-αξία* της μελλοντικής χρονικής στιγμής $t+1$.

Στο σημείο αυτό επανερχόμαστε πίσω στην πρώτη εκδοχή της *εξίσωσης Bellman* (Εξίσωση 7.4), όπου η τρέχουσα *Q-αξία* μιας ενέργειας υπολογίζεται εάν κανείς προσθέσει και τις μελλοντικές *Q-αξίες* επί της τρέχουσας όπως φαίνεται στο Σχήμα 7.4:

$$Q(s_t, a_t) \leftarrow (1-\lambda) \cdot \underbrace{Q(s_t, a_t)}_{\text{παλιά αξία}} + \underbrace{\lambda}_{\text{ποσοστό μάθησης}} \cdot \overbrace{\left(\underbrace{R_t}_{\text{ανταμοιβή}} + \underbrace{\gamma}_{\substack{\text{συντελεστής} \\ \text{Εξασθένισης}}} \cdot \underbrace{\max_a Q(s_{t+1}, a)}_{\text{εκτίμηση βέλτιστης μελλοντικής αξίας}} \right)}^{\text{εκπαιδευόμενη τιμή}}$$

Σχήμα 7.4: Η *Εξίσωση Bellman*, η οποία χρησιμοποιείται στον *αλγόριθμο Q-learning*.

Τονίζουμε ότι στον παραπάνω *κανόνα ενημέρωσης* (*update rule*) υπάρχει ένας *συντελεστής βαρύτητας* λ (*learning rate*) ο οποίος σταθμίζει τόσο την άμεση όσο και τη μελλοντική αξία (ή αλλιώς *αξία-Q*) αντίστοιχα. Η παραπάνω εξίσωση μας επιτρέπει να ενημερώνουμε σταδιακά τον *πίνακα Q* (*Quality table* ή *Q-table*) μετά από κάθε βήμα εκτέλεσης του *αλγορίθμου Q-learning*. Η διαδικασία θα περιγραφεί αναλυτικά στην επόμενη υποενότητα.

7.3.1 Ο Αλγόριθμος Q-learning σε Μορφή Ψευδοκώδικα

Στην υποενότητα αυτήν παρουσιάζουμε (σε μορφή ψευδοκώδικα) τον *αλγόριθμο Q-learning*, όπως φαίνεται στον Αλγόριθμο 3. Μεταξύ των δεδομένων εισόδου του αλγορίθμου είναι και ο *πίνακας-Q* (*Quality table*), ο οποίος είναι ένας δισδιάστατος πίνακας που κρατά την *αξία* (*Q-value*) κάθε δυνατής ενέργειας a δοθείσης μιας κατάστασης s. Η εξίσωση του Σχήματος 7.4 εμφανίζεται στο τέλος της γραμμής 4 του αλγορίθμου 3 όπου ο *ευφυής πράκτορας* υπολογίζει την ανταμοιβή της τρέχουσας χρονικής στιγμής t με βάση την *προσδωκόμενη βέλτιστη μελλοντική ανταμοιβή* της χρονικής στιγμής $t+1$. Η βασική παραδοχή που κάνουμε είναι ότι ο *ευφυής πράκτορας* επιλέγει σε κάθε κατάσταση s με πιθανότητα 1-β την καλύτερη τρέχουσα γνωστή ενέργεια a, ενώ με πιθανότητα β επιλέγει μια τυχαία ενέργεια. Σ' ένα σύστημα συστάσεων ο *ευφυής πράκτορας* θα αναζητήσει όλα

τα δυνάμει υποψήφια προς σύσταση στοιχεία για το προφίλ ενός συγκεκριμένου χρήστη, και θα επιλέξει με πιθανότητα 1-β το ζεύγος χρήστη-στοιχείου (κατάσταση-ενέργεια) με την υψηλότερη μελλοντική ανταμοιβή. Τονίζουμε ότι κάθε φορά ο αλγόριθμος θα επαναλαμβάνεται όσο οι *αξίες-Q* δεν συγκλίνουν ή για όσο δεν έχει φτάσει έναν μέγιστο αριθμό επαναλήψεων (βλέπε γραμμή 3). Κάθε φορά λοιπόν επιλέγεται τυχαία μια κατάσταση s και η ενημέρωση των μεταβλητών συνεχίζεται μέχρι να βρεθούμε σε μια *τερματική* κατάσταση s (βλέπε γραμμή 4), κατά την οποία δηλαδή ο *ευφυής πράκτορας* θα πάρει τη μέγιστη δυνατή ανταμοιβή. Ακολουθεί ο *αλγόριθμος Q-learning* σε μορφή ψευδοκώδικα:

Αλγόριθμος 3 Q-learning

Είσοδος:

Καταστάσεις $S = \{s_1, s_2, ..., s_n\}$

Ενέργειες $A = \{a_1, a_2, ..., a_n\}$

Συνάρτηση Ανταμοιβής $R : S \times A \rightarrow \mathbb{R}$

Συνάρτηση Πιθανοτήτων Μετάβασης $T : S \times A \rightarrow S$

Ποσοστό μάθησης $\lambda \in [0, 1]$, συνήθως $\lambda = 0.1$

Συντελεστής Εξασθένισης Ανταμοιβών $\gamma \in [0, 1]$

Πιθανότητα β για την επιλογή μιας τυχαίας ενέργειας όπου $\beta \in [0, 1]$

Έξοδος:

Ένας πίνακας Q που κρατά την αξία για καθεμία από τις ενέργειες a στην κατάσταση s

1: **procedure** Q-LEARNING($S, A, R, T, \lambda, \gamma, \beta$)

2: Αρχικοποίηση των τιμών του πίνακα Q

3: **Όσο** οι αξίες-Q δεν συγκλίνουν **ή** δεν έχει φτάσει έναν αριθμό επαναλήψεων **επανάλαβε**

Επίλεξε τυχαία μία κατάσταση $s \in S$

4: **Όσο** η κατάσταση s δεν είναι τερματική **επανάλαβε**

Επίλεξε βάσει μιας *πολιτικής* π με πιθανότητα β μια τυχαία ενέργεια a_t, αλλιώς επίλεξε με πιθανότητα $1 - \beta$ τη βέλτιστη ενέργεια $\pi(s_t) \leftarrow \max_a Q(s_{t+1}, a)$

$a_t \leftarrow \pi(s_t)$ ▷$\pi : S \rightarrow A$

$r \leftarrow R(s_t, a_t)$ ▷ Λήψη της ανταμοιβής

$s_{t+1} \leftarrow T(s_t, a_t)$ ▷ Μετάβαση στη νέα κατάσταση

$Q(s_{t+1}, a) \leftarrow (1 - \lambda) \cdot Q(s_t, a_t) + \lambda \times (r + \gamma \cdot max_a Q(s_{t+1}, a))$

$s \leftarrow s_{t+1}$

τέλος_όσο

5: **τέλος_όσο**

6: **Επίστρεψε** πίνακα Q

7: **end procedure**

7.3.2 Βήμα-Βήμα Εκτέλεση του Αλγορίθμου Q-learning

Παράδειγμα 7.1 Για την ευχερέστερη περιγραφή του *αλγορίθμου Q-learning* θα χρησιμοποιήσουμε τα δεδομένα του Σχήματος 7.5 που αφορούν σ' έναν πίνακα βαθμολογίας χρηστών και στοιχείων όπου U_{0-5} είναι οι χρήστες και I_{0-5} τα στοιχεία που αυτοί έχουν βαθμολογήσει. Η κλίμακα βαθμολογίας είναι 0 έως 5. Για τις ανάγκες του παραδείγματός μας θεωρούμε ότι οι χρήστες-καταναλωτές τελικά αγοράζουν μόνο τα στοιχεία που έχουν βαθμολογήσει με 5 βαθμούς. Η αγορά ή μη ενός στοιχείου από τον υπό εξέταση χρήστη είναι αυτό που θέλουμε να προβλέπει ο *αλγόριθμος Q-learning*, τον οποίο και εφαρμόζουμε σταδιακά (για 3 επαναλήψεις) σ' αυτό το παράδειγμα (βλέπε Αλγόριθμος 3).

	I_0	I_1	I_2	I_3	I_4	I_5
U_0	5	1	1	4	0	0
U_1	1	5	4	0	0	4
U_2	2	1	5	0	0	4
U_3	1	2	1	5	0	0
U_4	5	1	2	0	1	0
U_5	1	5	4	0	4	0

Σχήμα 7.5: Δισδιάστατος Πίνακας βαθμολογίας Χρηστών-Στοιχείων A (6×6).

Προτού λοιπόν ξεκινήσει η πρώτη επανάληψη του αλγορίθμου, ο *Q-table* αρχικοποιείται με μηδενικά, όπως φαίνεται και στο αριστερό μέρος του Σχήματος 7.6.

Στην πρώτη επανάληψη:
Ο αλγόριθμος επιλέγει με πιθανότητα β μια τυχαία ενέργεια a (δηλαδή ένα στοιχείο i) για μια τυχαία κατάσταση s (δηλαδή έναν χρήστη u). Ο *ευφυής πράκτορας* δηλαδή επιλέγει τυχαία για τον χρήστη με id = 3 (U_3) το στοιχείο με id = 0 (I_0) και παίρνει ανταμοιβή r = 1. Η ανταμοιβή r είναι 1, επειδή ο U_3 βαθμολόγησε το I_0 με μονάδα (1) όπως φαίνεται στο Σχήμα 7.5. Η νέα *αξία-Q* για την ενέργεια αυτή είναι Q(U_3,I_0) = 0.25. Στη συνέχεια ακολουθεί μία ακόμη ενημέρωση των μεταβλητών, δεδομένου ότι η κατάσταση s_3 δεν είναι τερματική και ο *ευφυής πράκτορας* δεν θα πάρει την μέγιστη ανταμοιβή. Τονίζουμε ότι για το παράδειγμά μας *τερματική είναι μια κατάσταση όταν ο χρήστης έχει προβεί σε αγορά ενός στοιχείου*, πράγμα που θα κάνει μόνο για στοιχείο που έχει βαθμολογήσει με 5 βαθμούς. Βάσει του Αλγορίθμου 3 επιλέγεται η επόμενη κατάσταση $s = 4$ (δηλαδή ο χρήστης U_4), η οποία είναι τερματική. Οι ενημερώσεις των μεταβλητών για την πρώτη επανάληψη είναι οι ακόλουθες: $a \leftarrow 0$, $r \leftarrow 1$, $s' \leftarrow 3$,Q(s',a)$\leftarrow (1-\lambda) \times Q(s,a) + \lambda \times (r + \gamma \times max_{a'}\, Q(s',a')) \Rightarrow$
$\Rightarrow Q(U_3, I_0) \leftarrow 0.75 * 0 + 0.25 * (1 + 0.75 * 0) \Rightarrow Q(U_3, I_0) \leftarrow 0.25$,

$s \leftarrow 4$, (η νέα κατάσταση s_4 είναι τερματική)
$a \leftarrow 0$, $r \leftarrow 5$, $s' \leftarrow 4$, $Q(U_4, I_0) \leftarrow 1.25$, $s \leftarrow 5$
Συνεπώς, μετά το τέλος της 1ης επανάληψης ο *Q-table* ενημερώνεται στο κελί Q(U_3,I_0) με την τιμή 0.25, και στο κελί Q(U_4,I_0) με την τιμή 1.25 όπως φαίνεται με κόκκινο χρώμα στο δεξιό μέρος του Σχήματος 7.6.

	I_0	I_1	I_2	I_3	I_4	I_5
U_0	0	0	0	0	0	0
U_1	0	0	0	0	0	0
U_2	0	0	0	0	0	0
U_3	0	0	0	0	0	0
U_4	0	0	0	0	0	0
U_5	0	0	0	0	0	0

Αρχικοποίηση του πίνακα Q με μηδενικά

	I_0	I_1	I_2	I_3	I_4	I_5
U_0	0	0	0	0	0	0
U_1	0	0	0	0	0	0
U_2	0	0	0	0	0	0
U_3	0.25	0	0	0	0	0
U_4	1.25	0	0	0	0	0
U_5	0	0	0	0	0	0

Πίνακας Q μετά την 1η Επανάληψη

Σχήμα 7.6: Ο πίνακας που κρατά τις *αξίες-Q* του παραδείγματός μας στην αρχικοποίησή του (βλ. αριστερά), και μετά την 1η επανάληψη του *αλγορίθμου Q-learning* (βλ. δεξιά).

Στη δεύτερη επανάληψη:
Ο *ευφυής πράκτορας* επιλέγει με πιθανότητα β τυχαία την κατάσταση $s = 0$ (δηλαδή τον χρήστη U_0), και προτείνει το στοιχείο με id = 0 (I_0) και ανταμοιβή r = 5. Άρα, η προαναφερθείσα κατάσταση s_0 του δίνει τη μέγιστη ανταμοιβή και επομένως καθίσταται τερματική. Η νέα *αξία-Q* για την ενέργεια αυτή είναι $Q(U_0, I_0) = 1.25$. Οι ενημερώσεις των μεταβλητών του Αλγορίθμου 3 για τη δεύτερη επανάληψη είναι οι ακόλουθες:
$a \leftarrow 0$, $r \leftarrow 5$, $s' \leftarrow 0$ (η s_0 είναι τερματική), $Q(U_0, I_0) \leftarrow 1.25$, $s \leftarrow 1$
Συνεπώς, μετά το τέλος της 2ης επανάληψης ο *Q-table* ενημερώνεται στο κελί $Q(U_0, I_0)$ με την τιμή 1.25 όπως φαίνεται με κόκκινο χρώμα στο αριστερό μέρος του Σχήματος 7.7.

Στην τρίτη επανάληψη:
Επιλέγεται, τυχαία, πάλι η κατάσταση s_0. Ως εκ τούτου, ο *ευφυής πράκτορας* προτείνει ξανά στον χρήστη με id = 0 (U_0) το στοιχείο με id = 0 (I_0) και με ανταμοιβή r = 5. Η νέα *αξία-Q* για την ενέργεια αυτή είναι Q(U_0,I_0) = 2.42. Οι ενημερώσεις των μεταβλητών του Αλγορίθμου 3 για την τρίτη επανάληψη είναι οι ακόλουθες:
$a \leftarrow 0$, $r \leftarrow 5$, $s' \leftarrow 0$ (η s_0 είναι τερματική), $Q(U_0, I_0) \leftarrow 2.42$, $s \leftarrow 1$
Μετά το τέλος της 3ης επανάληψης ο *Q-table* ενημερώνεται στο κελί $Q(U_0, I_0)$ με την τιμή 2.42 όπως φαίνεται με κόκκινο χρώμα στο δεξιό μέρος του Σχήματος 7.7. Να σημειωθεί εδώ

ότι στο τέλος του κεφαλαίου υπάρχει λυμένη προγραμματιστική άσκηση στη γλώσσα *Python* με τις τελικές τιμές των *αξιών-Q* του παραδείγματος 7.1.

	I_0	I_1	I_2	I_3	I_4	I_5
U_0	1.25	0	0	0	0	0
U_1	0	0	0	0	0	0
U_2	0	0	0	0	0	0
U_3	0.25	0	0	0	0	0
U_4	1.25	0	0	0	0	0
U_5	0	0	0	0	0	0

Πίνακας Q μετά τη 2η Επανάληψη

	I_0	I_1	I_2	I_3	I_4	I_5
U_0	2.42	0	0	0	0	0
U_1	0	0	0	0	0	0
U_2	0	0	0	0	0	0
U_3	0.25	0	0	0	0	0
U_4	1.25	0	0	0	0	0
U_5	0	0	0	0	0	0

Πίνακας Q μετά την 3η Επανάληψη

Σχήμα 7.7: Ο πίνακας που κρατά τις *αξίες-Q* του παραδείγματός μας μετά τη 2η επανάληψη (βλ. αριστερά) και μετά την 3η επανάληψη (βλ. δεξιά) του *αλγορίθμου Q-learning*.

7.4 Βαθιά Ενισχυτική Μάθηση

Το κύριο χαρακτηριστικό ενός *αλγορίθμου βαθιάς ενισχυτικής μάθησης* είναι ότι υλοποιείται με τη βοήθεια ενός *νευρωνικού δικτύου πολλαπλών επιπέδων νευρώνων*. Τα τελευταία χρόνια αναπτύχθηκαν πολλοί *αλγόριθμοι βαθιάς ενισχυτικής μάθησης* (*Deep Reinforcement Learning*), όπως ο *Deep Q-Learning (DQN)* [Hester et al., 2018], ο *Double DQN* [Hasselt, 2010] και *το μοντέλο Advantage Actor-Critic (A2C)* [Mnih et al., 2016], για την αντιμετώπιση διαφόρων προβλημάτων στη γνωστική περιοχή των *Παιγνίων* και της *Τεχνητής Νοημοσύνης*. Η *μέθοδος Double DQN (DDQN)*, παραδείγματος χάρη, χρησιμοποιείται για την αντιμετώπιση της υπερεκτίμησης των *αξιών-Q* (*Q-values*). Πιο συγκεκριμένα, χρησιμοποιεί ένα δευτερεύον μοντέλο πρόβλεψης (έναν δεύτερο *Q-table*) που είναι το αντίγραφο του κύριου μοντέλου πρόβλεψης από το τελευταίο μόνο επεισόδιό του. Όταν η διαφορά μεταξύ των *αξιών-Q* του δεύτερου μοντέλου είναι μικρότερη από αυτές στο κύριο μοντέλο, τότε χρησιμοποιούμε το δεύτερο μοντέλο για να προβλέψουμε μια ενέργεια, έτσι ώστε η πρόβλεψή μας να μη μεροληπτεί έναντι των ενεργειών που κυριαρχούν στο κύριο μοντέλο πρόβλεψης.

Αυτές οι καινοτομίες και η αποτελεσματικότητα των *μεθόδων βαθιάς μηχανικής μάθησης* στα *Παίγνια* οδήγησαν στην προσαρμογή τους προκειμένου να εφαρμοστούν και στα συστήματα συστάσεων. *Το μοντέλο Advantage Actor-Critic (A2C)* [Mnih et al., 2016], λόγου χάρη, αποτελείται από δύο νευρωνικά δίκτυα: το πρώτο, *ο actor (ηθοποιός)*, παίρνει στην είσοδό του το *τρέχον προφίλ* του χρήστη (*state*) και παράγει την *καλύτερη σύσταση* (*action*), ενώ το δεύτερο, ο *κριτικός (critic)*, αξιολογεί τη σύσταση ως προς το πόσο καλή είναι σε σχέση με το προφίλ του υπό εξέταση χρήστη. Το *μοντέλο A2C*, λοιπόν, συνδυάζει τα πλεονεκτήματα τόσο των *off-policy μεθόδων ενισχυτικής μάθησης*,

οι οποίες βασίζονται στην *αξία της κατάστασης* (*value-based*) όσο και των *on-policy μεθόδων* RL, οι οποίες βελτιστοποιούν απευθείας την *τρέχουσα πολιτική*.

Στη συνέχεια, θα περιγράψουμε καταρχάς τον *αλγόριθμο Deep Q-learning (DQN)*, ο οποίος είναι η υλοποίηση του *αλγορίθμου Q-learning* με τη χρήση ενός νευρωνικού δικτύου. Τέλος, ομοίως θα πράξουμε και για τον *αλγορίθμο A2C*, δεδομένου ότι συγκεντρώνει τα πλεονεκτήματα τόσο των *on-policy RL αλγορίθμων* όσο και των *off-policy RL αλγορίθμων*.

7.4.1 Ο Αλγόριθμος Deep Q-Network (DQN)

Ο *αλγόριθμος Q-learning* μετατρέπεται σ' ένα *Deep Q-Network (DQN)* όταν η συνάρτηση *αξίας-Q* μιας ενέργειας a δοθείσης μιας κατάστασης s, αναπαρίσταται με τη χρήση ενός νευρωνικού δικτύου (π.χ. με τη χρήση ενός *multi-layer perceptron*). Αναλυτικότερα, έστω $Q(s,a;\theta)$ η συνάρτηση *αξίας-Q* με παραμέτρους $\theta \in \Theta$ - όπου θ είναι διάφορα γεγονότα (π.χ. αλληλεπίδραση των χρηστών με στοιχεία, βαθμολογίες χρηστών σε στοιχεία κ.λπ.). Οι ενημερώσεις των παραμέτρων θ μπορούν να προκύψουν μέσω του *αλγορίθμου Q-learning*, ο οποίος στοχεύει στην εύρεση προσεγγιστικά της βέλτιστης συνάρτηση *αξίας-Q* ως ακολούθως:

$$Q^{\star}(s,a) \approx Q(s,a;\theta).$$

Ο στόχος του *DQN* είναι να ελαχιστοποιήσει το λεγόμενο *σφάλμα χρονικής διαφοράς* (ή αλλιώς, *μάθηση χρονικής διαφοράς*), που αφορά στις προβλεπόμενες *αξίες-Q* μεταξύ δύο διαδοχικών χρονικών στιγμών (t, και t+1) με τη χρήση της ακόλουθης *συνάρτησης απώλειας* (*loss function*):

$$L(\theta) = \mathbb{E}_{(s_t,a_t,r(s_t,a_t),s_{t+1})}\left[\left(r\left(s_t,a_t\right) + \gamma \cdot \max_a Q\left(s_{t+1},a,\theta\right) - Q\left(s_t,a_t;\theta\right)\right)^2\right] \tag{7.5}$$

όπου θ οι παράμετροι του *μοντέλου σφάλματος χρονικής διαφοράς*, και $\mathbb{E}_{(\theta)}$ η προσδοκώμενη διαφορά μεταξύ των δύο *αξιών-Q* (τρέχουσας και μελλοντικής). Ο δε *κανόνας ενημέρωσης* (*update rule*) για τις παραμέτρους θ του *μοντέλου σφάλματος χρονικής διαφοράς* μπορεί να περιγραφεί ως εξής:

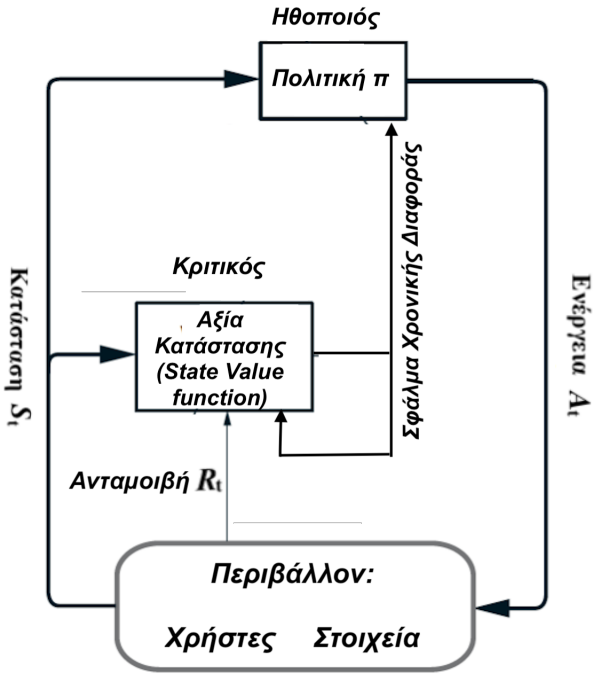
$$\theta \leftarrow \theta + \lambda \cdot \left(r\left(s_t,a_t\right) + \gamma \cdot \max_a Q\left(s_{t+1},a,\theta\right) - Q\left(s_t,a_t;\theta\right)\right) \cdot \nabla_\theta Q\left(s_t,a_t;\theta\right)$$

όπου λ ο *ρυθμός μάθησης* (*learning rate*) και $\nabla_\theta Q\left(s_t,a_t;\theta\right)$ το διάνυσμα με τις κλίσεις (μερικές παραγώγους) της *συνάρτησης απώλειας* ή αλλιώς *ανάδελτα* (*gradient*).

7.4.2 Ο Αλγόριθμος Advantage Actor Critic (A2C)

Ο απλός *αλγόριθμος Actor Critic* βασίζεται στη *μάθηση χρονικής διαφοράς* (*temporal difference learning*) - την οποία και παρουσιάσαμε σε προηγούμενη υποενότητα. Οι *μέθοδοι χρονικής διαφοράς* αναπαριστούν με τη χρήση διαφορετικών νευρωνικών δικτύων την *πολιτική* π που ακολουθεί ο *ευφυής πράκτορας* και τη *συνάρτηση αξίας της κατάστασης* (*state value function*) $V(s)$. Η δομή του νευρωνικού δικτύου που αναπαριστά την *πολιτική* π είναι γνωστή και ως *ο ηθοποιός (actor)*, επειδή

χρησιμοποιείται για την επιλογή των ενεργειών, ενώ η εκτιμώμενη συνάρτηση *αξίας της κατάστασης* (*state value*) είναι γνωστή ως *ο κριτικός (critic)*, επειδή ασκεί κριτική στις ενέργειες που γίνονται από τον *ηθοποιό*. Η "κριτική" παίρνει τη μορφή ενός *σφάλματος χρονικής διαφοράς* (*time difference error*), το οποίο είναι η έξοδος του *κριτή*, και καθοδηγεί όλη την *ενισχυτική μάθηση* τόσο στον *ηθοποιό* όσο και στον *κριτή* όπως φαίνεται και στο Σχήμα 7.8.



Σχήμα 7.8: Η διάδραση μεταξύ *ευφυούς πράκτορα* (*Ηθοποιός* και *Κριτικός*) και του περιβάλλοντος στα συστήματα συστάσεων.

Όπως φαίνεται στο Σχήμα 7.8, μετά την επιλογή μιας ενέργειας A_t ο *κριτικός* αξιολογεί τη νέα κατάσταση S_t για να καθορίσει εάν τα πράγματα πήγαν καλύτερα ή χειρότερα από την προσδοκώμενη εκτίμηση. Αυτή η αξιολόγηση συνιστά το *σφάλμα χρονικής διαφοράς* και ορίζεται ως ακολούθως:

$$\delta_t = R_{t+1} + \gamma \cdot V_t(S_{t+1}) - V(S_t)$$

όπου V_t η συνάρτηση *αξίας της κατάστασης* S_t που εφαρμόζεται από τον *κριτικό* τη χρονική στιγμή t. Αυτό το *σφάλμα χρονικής διαφοράς* μπορεί να χρησιμοποιηθεί για την αξιολόγηση της ενέργειας που μόλις επιλέχθηκε, δηλαδή της ενέργειας A_t που λήφθηκε στην κατάσταση S_t. Εάν το *σφάλμα χρονικής διαφοράς* είναι θετικό, τότε υποδηλώνει ότι η τάση επιλογής της A_t πρέπει να ενισχυθεί για το μέλλον, ενώ, εάν το *σφάλμα χρονικής διαφοράς* είναι αρνητικό, τότε υποδηλώνει ότι η τάση επιλογής της A_t πρέπει να αποδυναμωθεί.

Όσον αφορά στον *αλγόριθμο Advantage Actor Critic (A2C)*, αυτός είναι μια *επέκταση του αλγορίθμου Actor - Critic*, και βασίζεται σ' έναν διαφορετικό τρόπο εκπαίδευσης του *ευφυούς πράκτορα*, και συγκεκριμένα, στη *μέθοδο διαβαθμισμένης πολιτικής* (*gradient policy*). Αυτή η μέθοδος προσπαθεί να εκπαιδεύσει απευθείας την *πολιτική* π του *ευφυούς πράκτορα* (δηλαδή την ενέργεια που αυτός επιλέγει κάθε φορά) αντί να προσπαθεί προηγουμένως να προβλέψει την αξία της ενέργειας (*Q-value*) - όπως αναλύσαμε σχετικώς με την προσέγγιση *Q-learning* σε προηγούμενη υποενότητα. Σύμφωνα λοιπόν με τον *αλγόριθμο A2C*, κάθε φορά που ένας χρήστης αλληλεπιδρά με το σύστημα συστάσεων, του προσφέρεται μια λίστα με προτεινόμενα στοιχεία. Σημειώνουμε ότι το σύστημα συστάσεων είναι δυνατόν να καταγράφει διαφορετικά γεγονότα $\theta \in \Theta$, όπως τις αλληλεπιδράσεις (*clicks*) των χρηστών με τα στοιχεία ή τις βαθμολογίες (*ratings*) των χρηστών για τα επιμέρους στοιχεία. Δεδομένου ενός γεγονότος θ, ο *ευφυής πράκτορας* μπορεί να χρησιμοποιήσει μια ποικιλία *πολιτικών* $\pi \in \Pi$, οι οποίες επιβραβεύονται με μια ανταμοιβή $r \in \mathcal{R}$, εφόσον τα προτεινόμενα στοιχεία αρέσουν στον υπό εξέταση χρήστη. Η *αντικειμενική συνάρτηση* γι' αυτήν τη μέθοδο μπορεί να διατυπωθεί ως εξής:

$$L(\theta) = \mathbb{E}_t \left[\pi_\theta(a \mid s) \cdot A(s, a) \right] \tag{7.6}$$

όπου $\pi_\theta(a \mid s)$ η κατανομή πιθανότητας της *πολιτικής* π βάσει ενός γεγονότος θ με την οποία επιλέγεται η ενέργεια a όταν βρισκόμαστε στην κατάσταση s, και $A(s, a)$ η **συνάρτηση πλεονεκτήματος (Advantage function)**, η οποία μετρά τον βαθμό βελτίωσης των σωρευτικών ανταμοιβών που λαμβάνονται (δηλαδή το όφελος ή αλλιώς πλεονέκτημα που έχουμε) κατά την επιλογή της ενέργειας a στην κατάσταση s σε σύγκριση με κάποια άλλη βασική ανταμοιβή. Η *συνάρτηση πλεονεκτήματος* μπορεί επίσης να γραφεί ως ακολούθως:

$$A(s, a) = Q(s, a) - V(s)$$

όπου $V(s)$ η *συνάρτηση αξίας* της κατάστασης s. Εναλλακτικά, μπορούμε να χρησιμοποιήσουμε αντί του $Q(s, a)$ την πρόβλεψη των σωρευτικών μελλοντικών ανταμοιβών $R = \max \sum_{t=0}^{\infty} \gamma_t \times r_t$, όπου γ *ο συντελεστής εξασθένισης* και r_t η *άμεση ανταμοιβή*.

Στην περίπτωση του *αλγορίθμου A2C*, *ο κριτικός* εκτιμά τη *συνάρτηση αξίας της κατάστασης* (δηλαδή την αξία $V(s)$ της νέας κατάστασης που προέκυψε μετά από την επιλογής μιας ενέργειας), ενώ *ο ηθοποιός* ενημερώνει την κατανομή πιθανότητας της *πολιτικής* π προς την κατεύθυνση που προτείνει *ο κριτικός*. Άρα, ένα *βαθύ νευρωνικό δίκτυο* (*ο κριτικός*), το οποίο αποτελείται από διαφορετικά πλήρως συνδεδεμένα επίπεδα νευρώνων, εκτιμά πόσο καλή είναι μια ενέργεια για μια συγκεκριμένη κατάσταση. Επίσης, ένα δεύτερο νευρωνικό δίκτυο (*ο ηθοποιός*) εκτιμά πόσο καλή είναι μια *πολιτική* $\pi(s)$ υπολογίζοντας την αξία των διαφορετικών ενεργειών που μπορούν να επιλεχθούν στην κατάσταση s. Ο *ευφυής πράκτορας* υπολογίζει για κάθε κατάσταση s τη *συνάρτηση απώλειας* (*loss function*) της αξίας της κατάστασης καθώς και τη *συνάρτηση απώλειας* της *διαβαθμισμένης πολιτικής*, και τις αναμεταδίδει στα δύο νευρωνικά δίκτυα ως εξής :

$$\begin{aligned} &\text{Value Loss: } L_v = \sum (R_t - V(s_t; \theta_v))^2 \\ &\text{Policy Loss: } L_p = -\log(\pi(a_t \mid s_t; \theta)) \cdot A(s_t, a_t; \theta, \theta_v) - \beta \cdot H(\pi(s_t; \theta)) \end{aligned} \tag{7.7}$$

όπου το $H(\pi(s_t; \theta)$ δηλώνει τον βαθμό *εντροπίας/αβεβαιότητας (entropy)* της υπό εξέταση *πολιτικής*. Η προσθήκη αυτού του *όρου κανονικοποίησης* (*regularization term*) στην παραπάνω εξίσωση βελτιώνει την *εξερεύνηση* (*exploration*) του *αλγορίθμου A2C*, ενώ η παράμετρος β ελέγχει την ισχύ του *όρου κανονικοποίησης* (*regularization term*) της *εντροπίας*. Έτσι, η παράμετρος β μπορεί να επηρεάσει την κατανομή των πιθανοτήτων ενέργειας. Εάν η *πολιτική* επιλέγει περισσότερες ενέργειες με σχετικά παρόμοιες πιθανότητες, τότε η *εντροπία* θα είναι υψηλή, αλλά εάν η *πολιτική* προτείνει μια συγκεκριμένη ενέργεια με μεγάλη πιθανότητα, τότε η *εντροπία* θα είναι χαμηλή. Με άλλα λόγια, χρησιμοποιούμε την *εντροπία* ως μέσο βελτίωσης της *εξερεύνησης*, ενθαρρύνοντας το μοντέλο να είναι λιγότερο συντηρητικό όσον αφορά στη βεβαιότητά του για την επιλογή της σωστής ενέργειας.

Τέλος, ο *ευφυής πράκτορας* υπολογίζει τις κλίσεις που ελαχιστοποιούν τις προαναφερθείσες *συναρτήσεις απώλειας* (*Συνάρτηση Απώλειας της πολιτικής* L_p και *Συνάρτηση Απώλειας της αξίας της κατάστασης* L_v) για κάθε βήμα που εκτελείται χρησιμοποιώντας τους ακόλουθους *κανόνες ενημέρωσης*:

$$\theta \leftarrow \theta + \nabla_\theta \cdot L_p$$

και

$$\theta_v \leftarrow \theta_v + \nabla_{\theta_v} \cdot L_v$$

όπου θ και θ_v οι παράμετροι της *πολιτικής* π και της *αξίας της κατάστασης* $V(s)$ αντίστοιχα, ενώ ∇_θ και ∇_{θ_v} τα *διανύσματα των κλίσεων* (*ανάδελτα*) των δύο *συναρτήσεων απώλειας* αντίστοιχα.

Συνοψίζοντας, ο *ευφυής πράκτορας* χρησιμοποιεί την *εκτίμηση της αξίας της κατάστασης* (δηλαδή τον *κριτικό*) για να ενημερώσει την *πολιτική* (δηλαδή τον *ηθοποιό*) με παρόμοιο τρόπο που στο θέατρο ένας κριτικός καθοδηγεί τους συντελεστές μιας θεατρικής παράστασης μέσω της κριτικής αξιολόγησης που τους κάνει, ώστε αυτοί να βελτιωθούν και να επιτύχουν υψηλότερη απόδοση στον ρόλο τους.

7.5 Γενετικοί Αλγόριθμοι

Στην ακόλουθη υποενότητα περιγράφονται οι *γενετικοί αλγορίθμοι* και το πώς αυτοί εφαρμόζονται στα συστήματα συστάσεων. Ο Holland [Holland, 1992] ήταν ο πρώτος που παρουσίασε στο

βιβλίο του "Adaptation in Natural and Artificial Systems" τους *γενετικούς αλγορίθμους*, οι οποίοι βασίζονται στους μηχανισμούς της (γενετικής) *εξέλιξης* των οργανισμών και ειδικότερα στη *φυσική επιλογή*. Πιο συγκεκριμένα, ο Holland και οι συνεργάτες του κατάφεραν να σχεδιάσουν τεχνητά συστήματα λογισμικού που αξιοποιούσαν σημαντικούς μηχανισμούς των φυσικών λειτουργιών των οργανισμών, όπως **η φυσική επιλογή (selection)**, **η διασταύρωση (crossover)**, **η μετάλλαξη (mutation)** κ.λπ. *Οι γενετικοί* λοιπόν *αλγόριθμοι* ακολουθούν μια διαδικασία αναζήτησης της βέλτιστης λύσης, η οποία (αναζήτηση) καθοδηγείται από μια **συνάρτηση καταλληλότητας (fitness function)**, που αξιολογεί έναν μεγάλο αριθμό διαφορετικών δυνατών λύσεων. Τονίζουμε ότι η διαδικασία αναζήτησης της βέλτιστης λύσης χρησιμοποιεί ως εργαλείο την *τυχαία επιλογή* για να οδηγηθεί σε μια υψηλής ποιότητας λύση. Γενικά, μπορούμε να περιγράψουμε έναν *γενετικό αλγόριθμο* ως μια διαδικασία συλλογής ποιοτικά καλών δομικών συστατικών με τη χρήση γενετικού τύπου πράξεων. Ως εκ τούτου, η βασική ιδέα πίσω από τους *γενετικούς αλγορίθμους* εδράζεται στο γεγονός ότι οι καλές λύσεις συγκροτούνται από καλά δομικά συστατικά.

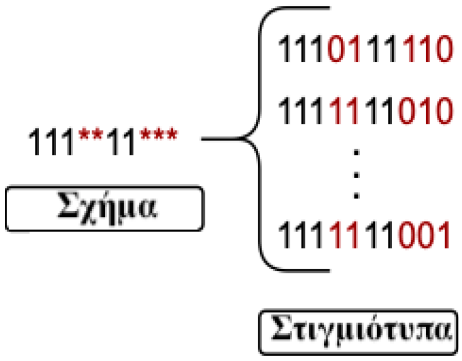
7.5.1 Τα Δομικά Συστατικά των Γενετικών Αλγορίθμων

Οι *γενετικοί αλγόριθμοι* συνίστανται από τα ακόλουθα δομικά συστατικά:

- **Πληθυσμός Χρωμοσωμάτων**.

  Στα *φυσικά συστήματα* δύο ή περισσότερα χρωμοσώματα συνδυάζονται για να σχηματίσουν τη συνολική συνταγή, προκειμένου να ικανοποιηθούν οι κατασκευαστικές οδηγίες και οι βασικές λειτουργίες των οργανισμών. Τα χρωμοσώματα με τη σειρά τους συντίθενται από γονίδια, τα οποία ελέγχουν συγκεκριμένα δομικά στοιχεία του οργανισμού, όπως, για παράδειγμα, το γονίδιο που ελέγχει το χρώμα ματιών ενός θηλαστικού και έχει τιμή "μπλε μάτια". Από την άλλη μεριά, στα *τεχνητά συστήματα* λογισμικού τα δομικά στοιχεία, όπως τα χρωμοσώματα, μπορούν να οριστούν από ένα σύνολο δυαδικών ψηφίων (δηλαδή 0 και 1). Επίσης, οι *γενετικοί αλγόριθμοι* απαιτούν το σύνολο των παραμέτρων ενός προβλήματος βελτιστοποίησης να κωδικοποιηθεί με μια πεπερασμένου μήκους δυαδική αναπαράσταση, η οποία μπορεί να θεωρηθεί στιγμιότυπο ενός γενικού σχήματος. Το σχήμα $1**1$, λόγου χάρη, αναπαριστά όλες τις πιθανές δυαδικές κωδικοποιήσεις που αρχίζουν και τελειώνουν με 1 και έχει μήκος ίσο με 4. Οι αστερίσκοι σ' αυτό αντιπροσωπεύουν μια τιμή που μπορεί να είναι 0 ή 1. Στο ακόλουθο Σχήμα 7.9 απεικονίζεται ένα γενικό σχήμα ή *μάσκα* ενός χρωμοσώματος και μερικά από τα έγκυρα *στιγμιότυπά* του:

  Συνεπώς, η επίλυση προβλημάτων με τη χρήση *γενετικών αλγορίθμων* περιλαμβάνει τη δημιουργία ενός αρχικού πληθυσμού χρωμοσωμάτων. Καθώς ο *γενετικός αλγόριθμος* εξερευνά τον χώρο των δυνατών λύσεων μέσω της διαδικασίας της *εξέλιξης*, τα αρχικά χρωμοσώματα, τα οποία συνήθως βρίσκονται σε διαφορετική θέση στον χώρο αναζήτησης, συνδυάζονται προκειμένου να δημιουργήσουν τη νέα γενιά χρωμοσωμάτων. Έτσι, ο *γενετικός αλγόριθμος* μπορεί να εξερευνήσει



Σχήμα 7.9: Ένα απλό παράδειγμα στιγμιοτύπων χρωμοσωμάτων από ένα γενικό σχήμα.

διαφορετικές περιοχές των πιθανών λύσεων συνδυάζοντας τα χρωμοσώματα μέσα από τις γενετικές πράξεις της *επιλογής* και της *διασταύρωσης*.

- **Συνάρτηση Καταλληλόλητας (fitness function)**

  Στους *φυσικούς οργανισμούς* η υγεία είναι αποφασιστικής σημασίας και υπάρχει ως εκ γενετής (έμφυτη) ικανότητα προκειμένου ο οργανισμός να μπορεί να επιβιώνει από τις επιδημίες και όλα όσα απειλούν την ύπαρξή του, αλλά και για να επιτυγχάνεται η αναπαραγωγή του. Κατ' αναλογία, στα *τεχνητά συστήματα λογισμικού* η *αντικειμενική συνάρτηση καταλληλότητας* (*fitness function*) είναι εκείνη η οποία έχει τον τελικό λόγο για τη "ζωή" των δημιουργημάτων από *strings* (*αλφαριθμητικά*) ή για τον "θάνατό" τους (*εξαφάνιση*). Μ' άλλα λόγια, οι *γενετικοί αλγόριθμοι* απαιτούν την ύπαρξη μιας συνάρτησης που να αποδίδει έναν βαθμό καταλληλότητας σε κάθε χρωμόσωμα του τρέχοντος πληθυσμού, ώστε αυτό να επιλεγεί ως γονέας για τη δημιουργία των χρωμοσωμάτων του επόμενου πληθυσμού (της νέας γενιάς). Άρα, η καταλληλότητα ενός χρωμοσώματος εξαρτάται από το πόσο καλά αυτό το χρωμόσωμα επιλύει το υπό εξέταση πρόβλημα.

7.5.2 Γενετικές Πράξεις

Οι *γενετικοί αλγόριθμοι* υποστηρίζουν συνήθως τις παρακάτω βασικές γενετικές πράξεις: (α) *τελεστής αναπαραγωγής* ή αλλιώς, *επιλογής γονέων* (β) *τελεστής διασταύρωσης*, και (γ) *τελεστής μετάλλαξης*. Παρακάτω θα τις περιγράψουμε αναλυτικά:

- **Τελεστής Αναπαραγωγής** ή **Επιλογής Γονέων** (**Reproduction** or **Selection operator**)
  Η διαδικασία *αναπαραγωγής* επιτρέπει την επιλογή των καλύτερων χρωμοσωμάτων (βάσει μιας *συνάρτησης καταλληλότητας*) από τον τρέχοντα πληθυσμό, τα οποία και θα παραγάγουν τους απογόνους του επόμενου πληθυσμού. Για τον λόγο αυτόν, ο *γενετικός αλγόριθμος* θα πρέπει να ισορροπεί μεταξύ της **εκμετάλλευσης (exploitation)** του τρέχοντος πληθυσμού με τη μεγιστοποίηση της σωρευτικής καταλληλότητας των χρωμοσωμάτων από τη μία, και της **εξερεύνησης**

(exploration) του χώρου αναζήτησης με τη διατήρηση της ποικιλομορφίας των χρωμοσωμάτων από την άλλη. Στην πραγματικότητα, υπάρχει η παρακάτω *αντιστάθμιση* (*trade-off*) μεταξύ της *εκμετάλλευσης* και της *εξερεύνησης* του τρέχοντος πληθυσμού:

- Μια αυστηρή επιλογή των καλύτερων χρωμοσωμάτων θα μειώσει την ποικιλομορφία του επόμενου πληθυσμού, γεγονός που, ναι μεν θα οδηγήσει σε γρήγορη σύγκλιση των λύσεων του αλγορίθμου (από τις πρώτες κιόλας γενιές), αλλά συνήθως θα καταλήξει και σε μια καλή αλλά όχι άριστη λύση δεδομένης της αναποτελεσματικής εξερεύνησης.
- Μια πιο ελαστική επιλογή των καλύτερων χρωμοσωμάτων οδηγεί σε πολύ αργή χρονικά βελτίωση του πληθυσμού, εξαιτίας της μη αποδοτικής εκμετάλλευσής του.

Ως αποτέλεσμα της παραπάνω *αντιστάθμισης*, έχουν υιοθετηθεί διαφορετικές στρατηγικές επιλογής γονέων, τις πιο σημαντικές εκ των οποίων θα παραθέσουμε στη συνέχεια:

- **Επιλογή ρουλέτας (roulette selection)** ή **αναλογική επιλογή καταλληλότητας**: *Η αναλογική επιλογή καταλληλότητας* (*Fitness proportionate selection*), επίσης γνωστή ως *επιλογή ρουλέτας* (*Roulette wheel Selection*), είναι ένας γενετικός τελεστής που χρησιμοποιείται στους *γενετικούς αλγορίθμους* για την επιλογή δυνητικά χρήσιμων ατόμων (χρωμοσωμάτων) από τον πληθυσμό, προκειμένου να γίνουν γονείς του επόμενου πληθυσμού. Εάν f_i είναι η καταλληλότητα του ατόμου i στον πληθυσμό βάσει μιας *συνάρτησης καταλληλότητας* f, τότε η πιθανότητα επιλογής του είναι:

  $$p_i = \frac{f_i}{\sum_{j=1}^{N} f_j}$$

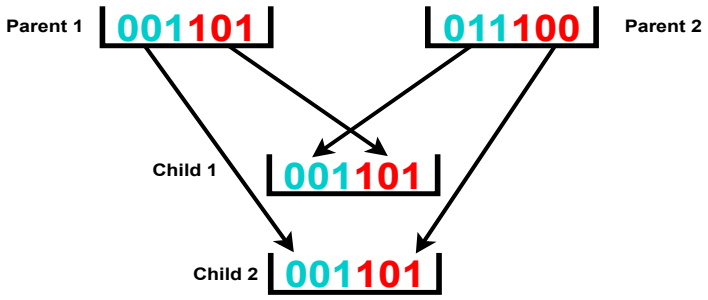
  όπου N ο αριθμός των ατόμων στον πληθυσμό. Το κύριο πρόβλημα της *αναλογικής επιλογής καταλληλότητας*, γνωστό ως *πρόωρη σύγκλιση*, είναι η περιορισμένη εξερεύνηση των ατόμων του πληθυσμού. Πιο συγκεκριμένα, στα πρώτα βήματα του *γενετικού αλγορίθμου* ένας μικρός μόνο αριθμός ατόμων εκτιμάται να είναι πολύ πιο ικανός και κατάλληλος από τα υπόλοιπα άτομα. Συνεπώς, αυτό το τμήμα του πληθυσμού αναπαράγεται πολύ συχνότερα από τα υπόλοιπα άτομα, με αποτέλεσμα να μην εξερευνώνται άλλες πιθανές λύσεις του προβλήματος που μπορεί να οδηγούσαν σε καλύτερη λύση.

– **Επιλογή κλίμακας σ (Sigma scaling selection)** [Goldberg, 1989]: Ο βαθμός καταλληλότητας ενός χρωμοσώματος βάσει της *επιλογής κλίμακας σ* υπολογίζεται από τη *συνάρτηση καταλληλότητας*, η οποία λαμβάνει υπόψη τη μέση καταλληλότητα του πληθυσμού και την τυπική απόκλιση του πληθυσμού. Έτσι, η *επιλογή κλίμακας σ* υποθέτει ότι η καταλληλότητα ενός ατόμου αυξάνεται ανάλογα με την τυπική απόκλισή του από τη μέση καταλληλότητα του πληθυσμού. Η μέθοδος αυτή επιτρέπει τη διατήρηση μιας σταθερής ισορροπίας μεταξύ εξευρεύνησης και εκμετάλλευσης του πληθυσμού.

– **Επιλογή βάσει Τουρνουά Boltzmann (Boltzmann Tournament Selection)** [Goldberg, 1990]: Σε αντίθεση με την *επιλογή κλίμακας σ*, η επιλογή χρωμοσωμάτων βάσει της *κατανομής Boltzmann* εξελίσσεται με την πάροδο των γενεών, οπότε και *ο γενετικός αλγόριθος* ξεκινά τη βελτιστοποίηση με υψηλά επίπεδα εξερεύνησης των δυνατών λύσεων και τελειώνει με ανάλογα υψηλά επίπεδα εκμετάλλευσης του πληθυσμού.

- **Τελεστής Διασταύρωσης (Crossover operator)**

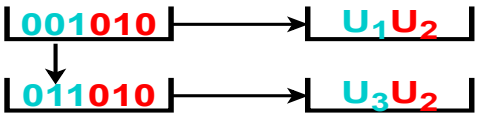
Η *διασταύρωση* επιτρέπει την ανταλλαγή ενός ή περισσότερων "γονιδίων" (δηλ. blocks αποτελούμενων από bits) μεταξύ των γονικών χρωμοσωμάτων. Η απλούστερη μορφή αυτού του τελεστή είναι η **διασταύρωση ενός σημείου (single-point crossover)**, η οποία επιλέγει τυχαία μια θέση στα χρωμοσώματα βάσει της οποίας ο τελεστής προσδιορίζει τα γονίδια που θα ανταλλαγούν. Στη συνέχεια θα παρουσιάσουμε μια σχηματική αναπαράσταση του *τελεστή διασταύρωσης ενός σημείου* που οδηγεί σε 2 "απογόνους-παιδιά".



Σχήμα 7.10: Παράδειγμα *τελεστή διασταύρωσης ενός σημείου* για τη δημιουργία 2 "απογόνων-παιδιών".

Ωστόσο, ο τελεστής αυτός περιορίζεται από δύο γνωστά προβλήματα: α) το πρόβλημα της *μεροληψίας θέσης* (*positional bias*) που προκαλείται από την υψηλή συσχέτιση των θέσεων των bit και της πιθανότητας διασταύρωσής τους, και β) την αδυναμία εφαρμογής της διασταύρωσης σε χρωμοσώματα μη ίσου μήκους. Για να αμβλυνθεί λοιπόν *το πρόβλημα της μεροληψίας θέσης*, προτάθηκε ο τελεστής **διασταύρωσης δύο σημείων (two-points crossover)**, σύμφωνα με τον οποίο δύο bit επιλέγονται τυχαία, και με βάση αυτά μπορούν να προσδιοριστούν τα γονίδια που θα ανταλλαγούν. Ακόμη, αυτός ο τελεστής επιτρέπει τη διασταύρωση περισσότερων γενετικών σχημάτων συγκριτικά μ' αυτόν του ενός σημείου. Και πάλι όμως, η διασταύρωση δύο σημείων δεν μπορεί να εφαρμοστεί σ' όλα τα γενετικά σχήματα. Ως εκ τούτου χρησιμοποιήθηκαν περισσότερες στατιστικές μέθοδοι (όπως η *κατανομή Poisson*) για την επιλογή του αριθμού των σημείων διασταύρωσης. Οι ερευνητικές εργασίες όμως που έχουν δημοσιευτεί μέχρι πρότινος δεν έχουν καθορίσει σαφείς κατευθυντήριες γραμμές σχετικά με το ποια συγκεκριμένη διασταύρωση μπορεί να προτιμάται έναντι κάποιας άλλης.

- **Τελεστής Μετάλλαξης (Mutation operator)**
  Ο τελεστής μετάλλαξης επιτρέπει την αντιστροφή της τιμής ενός bit του χρωμοσώματος σύμφωνα με κάποια πιθανότητα, όπως φαίνεται στο παρακάτω Σχήμα:



Σχήμα 7.11: Παράδειγμα πράξης *μετάλλαξης* ενός bit.

Επίσης, μπορεί να εφαρμοστεί στα bit οποιουδήποτε γονιδίου ή στο *σχήμα (μάσκα)* της *διασταύρωσης* έτσι ώστε τα σημεία *διασταύρωσης* να μπορούν να εξελίσσονται κατά τη διάρκεια της διαδικασίας βελτιστοποίησης. Η λειτουργία της *μετάλλαξης* συχνά προστατεύει από ανεπανόρθωτες απώλειες καλών λύσεων. Μ' άλλα λόγια, όταν χρησιμοποιείται με τρόπο συνετό μαζί με την *αναπαραγωγή* και τη *διασταύρωση* είναι μια ασφαλής *πολιτική* ενάντια σε μια πρόωρη απώλεια μιας καλής λύσης. Η δε συχνότητα *μετάλλαξης* για να αποκτήσουμε καλά αποτελέσματα, εμπειρικά μιλώντας, είναι περίπου μια αλλαγή ανά μερικές χιλιάδες θέσεις bit.

7.5.3 Σχετική Βιβλιογραφία

Οι γενετικοί αλγόριθμοι μπορούν να αναζητήσουν εξαντλητικώς έναν μεγάλο αριθμό διαφορετικών συνόλων γειτονιών (δηλαδή πλησιέστερων χρηστών) του υπό εξέταση χρήστη ενός συστήματος συστάσεων. Για τον λόγο αυτόν, έχουν χρησιμοποιηθεί στο παρελθόν πολλές φορές για την επίλυση του *προβλήματος της ομαδοποίησης των χρηστών*. Παραδείγματος χάρη, οι Rahman και Islam [Rahman and Islam, 2014] τροποποίησαν τον *αλγόριθμο k-means* έτσι ώστε να μπορεί να χρησιμοποιηθεί μαζί μ' έναν *γενετικό αλγόριθμο* για την αντιμετώπιση των γνωστών προβλημάτων -ένα εκ των οποίων είναι και ο τρόπος *επιλογής* των αρχικών κέντρων και του αριθμού των συστάδων που θα δημιουργηθούν- του πρώτου. Οι συγγραφείς του άρθρου λοιπόν πρότειναν τη χρήση μιας *συνάρτησης καταλληλότητας*, η οποία από τη μία μεριά ελαχιστοποιεί τις αποστάσεις μεταξύ των μελών εντός των συστάδων, και από την άλλη μεγιστοποιεί τις αποστάσεις μεταξύ των χρηστών που ανήκουν σε διαφορετικές συστάδες.

Επιπρόσθετα, ένας *γενετικός αλγόριθμος* χρησιμοποιήθηκε από τον Bobadilla [Bobadilla et al., 2011] για την αντιμετώπιση του *προβλήματος της αραιότητας του πίνακα βαθμολογιών χρηστών-στοιχείων*. Έτσι, για κάθε ζεύγος χρηστών υπολογίστηκε ένα διάνυσμα ομοιότητας σε σχέση με τη συνάφειά τους προς κάθε κοινή προτίμησή τους για ένα στοιχείο. Ορίστηκε, εν συνεχεία, με τον ίδιο τρόπο ένα σταθμισμένο διάνυσμα για την ομοιότητα μεταξύ των στοιχείων. Στο τέλος, μια *συνάρτηση καταλληλότητας* συνδύαζε τα δύο προαναφερθέντα διανύσματα για να επιλεχθούν οι "καλύτεροι" χρήστες ως γονείς, ώστε να χρησιμοποιηθούν για την αναπαραγωγή της επόμενης, βελτιωμένης, γενιάς.

Οι Katarya και Verna [Katarya and Verma, 2016], επιπλέον, πρότειναν μια *υβριδική* τεχνική ομαδοποίησης χρηστών που αποτελείται από τρεις αλγορίθμους: τον *αλγόριθμο k-means*, τη *βελτιστοποίηση σμήνους σωματιδίων* (*Particle Swarm Optimization - PSO*) και τον *αλγόριθμο ασαφούς ομαδοποίησης c-means* (*Fuzzy c-means clustering*). Αρχικά, οι χρήστες ομαδοποιούνται σύμφωνα με τις προτιμήσεις τους ως προς τις κατηγορίες των στοιχείων που αλληλεπιδρούν. Στη συνέχεια, εκτελείται ένας συνδυασμός των *αλγορίθμων PSO* και *k-means* για την επιλογή των αρχικών κέντρων των υπό δημιουργία ομάδων χρηστών. Τέλος, αυτά τα κέντρα χρησιμοποιούνται ως είσοδος του *αλγορίθμου c-means* για την οριστικοποίηση των δημιουργηθέντων ομάδων χρηστών.

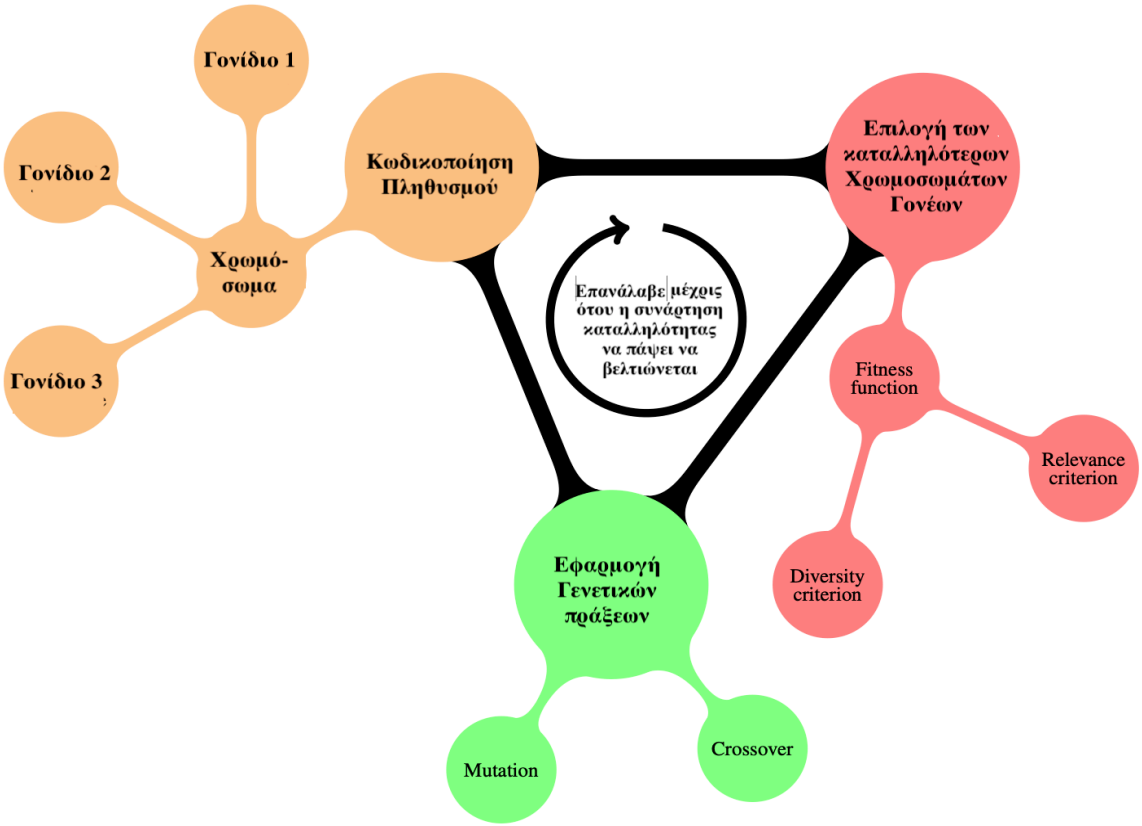
Επίσης, άλλοι *γενετικοί αλγόριθμοι* έχουν χρησιμοποιηθεί για τη βελτίωση της ποιότητας των συστάσεων τόσο ως προς την ακρίβεια όσο και ως προς την ποικιλομορφία των προτεινόμενων στοιχείων. Για παράδειγμα, οι Zhang και Hurley [Zhang and Hurley, 2009] πρότειναν έναν *αλγόριθμο ομαδοποίησης* για να εισάγουν μεγαλύτερη ποικιλομορφία μεταξύ των προτεινόμενων στοιχείων της λίστας συστάσεων. Για τον λόγο αυτόν, υιοθέτησαν μια μέθοδο επανακατάταξης της λίστας των προτεινόμενων στοιχείων για την αύξηση της ποικιλομορφίας τους. Τέλος, ο Bag και οι συνεργάτες του [Bag et al., 2019], όπως και ο Berbague και οι λοιποί συν-συγγραφείς [Berbague et al., 2021], πρότειναν μια *συνάρτηση καταλληλότητας* για την επιλογή των *k-πλησιέστερων γειτόνων* (*k-Nearest Neighbors*) του υπό εξέταση χρήστη, με στόχο τη μεγιστοποίηση τόσο της συνάφειας όσο και της ποικιλομορφίας των προτεινόμενων στοιχείων.

7.5.4 Η Αρχιτεκτονική ενός Γενετικού Αλγορίθμου για τα Συστήματα Συστάσεων

Ένας από τους στόχους των συστημάτων συστάσεων είναι να προσφέρουν συστάσεις που θα ικανοποιούν διαφορετικές μετρικές υπολογισμού της ποιότητας των προτεινόμενων στοιχείων (π.χ. να δίνουν ακριβείς και ταυτοχρόνως πρωτότυπες συστάσεις). Ένας *γενετικός αλγόριθμος* λοιπόν, προκειμένου να καλύψει τις παραπάνω απαιτήσεις, θα πρέπει να χρησιμοποιεί μια *συνάρτηση καταλληλότητας*, η οποία θα συνδυάζει δύο διαφορετικά κριτήρια (της *συνάφειας* και της *ποικιλομορφίας*), όπως αυτά εμφανίζονται στο δεξιό μέρος του Σχήματος 7.12:

Όσον αφορά τώρα στον πληθυσμό των χρωμοσωμάτων, η κωδικοποίηση ενός χρωμοσώματος συχνά έχει τη μορφή μιας ακολουθίας γονιδίων, όπως φαίνεται στο αριστερό μέρος του Σχήματος 7.12. Τα αρχικά χρωμοσώματα εξελίσσονται κατά τη διάρκεια των γενεών με γενετικές πράξεις όπως η *επιλογή*, η *μετάλλαξη* και η *διασταύρωση* για τη βελτιστοποίηση της ποιότητάς τους (βλ. το κέντρο του Σχήματος 7.12). Αυτή η διαδικασία επαναλαμβάνεται μέχρις ότου πάψει να βελτιώνεται η *συνάρτηση καταλληλότητας* ή επιτευχθεί ένας μέγιστος αριθμός καθορισμένων βημάτων του αλγορίθμου. Όσον αφορά δε στις άλλες δύο γενετικές πράξεις, η μεν *διασταύρωση* παράγει πάντα δύο νέα χρωμοσώματα με βάση τα δύο γονικά χρωμοσώματα ανταλλάσσοντας μερικά από τα γονίδιά τους, ενώ η *μετάλλαξη* αλλάζει τυχαία μια δυαδική τιμή ενός χρωμοσώματος.

Με βάση το προαναφερθέν Σχήμα 7.12, κάθε φορά που επιλέγουμε έναν υπό εξέταση χρήστη δημιουργούμε τυχαία ένα αρχικό σύνολο χρωμοσωμάτων. Αυτά τα χρωμοσώματα εξελίσσονται *n*



Σχήμα 7.12: Η βασική αρχιτεκτονική ενός *γενετικού αλγορίθμου* για τα συστήματα συστάσεων.

φορές, ενώ σε κάθε γενιά εφαρμόζονται στον πληθυσμό οι τελεστές *διασταύρωσης* και *μετάλλαξης* για την παραγωγή νέων χρωμοσωμάτων (δηλαδή την παραγωγή κάθε φορά των *πλησιέστερων γειτόνων του υπό εξέταση χρήστη*). Κατόπιν, επιλέγεται ο νέος πληθυσμός που αποτελεί τη νέα γενιά ανεξάρτητα από τη *συνάρτηση καταλληλότητας*. Τέλος, η *συνάρτηση καταλληλότητας* καθοδηγεί τη βελτιστοποίηση της *γειτονιάς των πλησιέστερων χρηστών* ως προς τη συνάφειά τους με τον υπό εξέταση χρήστη, καθώς και ως προς την ποικιλομορφία των στοιχείων αυτής της γειτονιάς.

7.5.5 Ο Γενετικός Αλγόριθμος σε μορφή ψευδοκώδικα

Στην παρούσα υποενότητα θα παρουσιάσουμε με τη μορφή ψευδοκώδικα έναν γενικό *γενετικό αλγόριθμο* η βασική αρχιτεκτονική του οποίου περιγράφηκε με το Σχήμα 7.12. Όπως ήδη αναφέραμε, με τον αλγόριθμο βελτιστοποίησης της *γειτονιάς του υπό εξέταση χρήστη* δημιουργείται σε κάθε επανάληψη μια νέα γενιά (δηλαδή μια νέα ομάδα διαφορετικών χρηστών που συγκροτούν τη γειτονιά του) μέχρι να καταλήξουμε στη βέλτιστη γειτονιά. Σε κάθε γενιά εφαρμόζουμε γενετικές πράξεις όπως η *επιλογή*, η *μετάλλαξη* και η *διασταύρωση* για την παραγωγή της επόμενης γενιάς. Ο αλγόριθμος λαμβάνει ως είσοδό του τον υπό εξέταση χρήστη u_0 και ένα σύνολο U χρηστών που τον πλαισιώνουν, και προσπαθεί να εντοπίσει μια βέλτιστη γειτονιά N_0 με τους κοντινότερους χρήστες του u_0.

Αλγόριθμος 4 Βελτιστοποίηση της γειτονιάς των κοντινότερων χρηστών.

Είσοδος:

$U = \{u_1, u_2, ..., u_n\}$: το σύνολο των χρηστών.

u_0: ο υπό εξέταση χρήστης.

m: ένας προκαθορισμένος μέγιστος αριθμός γενεών που θα παραχθούν.

f: το μέγεθος του αρχικού πληθυσμού.

Έξοδος:

N_0: η βέλτιστη γειτονιά του χρήστη u_0 ή αλλιώς το βέλτιστο χρωμόσωμα.

1: πληθυσμός ← Δημιουργία_τυχαίου_πληθυσμού (μέγεθος = f)

2: **Όσο** ο αριθμός των γενεών $g < m$ **επανάλαβε**

απόγονοι ← applySelectionOperator(πληθυσμός) Επιλογή Γονέων

απόγονοι ← applyCrossoverOperator(απόγονοι) Διασταύρωση

απόγονοι ← applyMutationOperator(απόγονοι) Μετάλλαξη

πληθυσμός ← απόγονοι ∪ πληθυσμός

$g \leftarrow g$+1

3: **τέλος_όσο**

4: N_0 ← Επίλεξε_βέλτιστη_γειτονιά (πληθυσμός) {βάσει του Fitness score τους}

Όπως λοιπόν φαίνεται στη γραμμή 2 του Αλγορίθμου 4, η διαδικασία βελτιστοποίησης *της γειτονιάς των κοντινότερων χρηστών του υπό εξέταση χρήστη* u_0 περιλαμβάνει m γενιές. Σε κάθε γενιά g εφαρμόζονται οι τρεις παρακάτω γενετικές λειτουργίες: (1) ο τελεστής της *επιλογής* των γονέων (χρωμοσωμάτων) από τον συνολικό πληθυσμό, (2) ο τελεστής *διασταύρωσης* για τα επιλεγμένα χρωμοσώματα και (3) ο τελεστής *μετάλλαξης*, ο οποίος εφαρμόζεται στα χρωμοσώματα που δημιουργήθηκαν μετά τη *διασταύρωση*. Τέλος, μετά τη βελτιστοποίηση του αλγορίθμου για m γενιές, το καλύτερο χρωμόσωμα (*γειτονιά χρηστών*) του πληθυσμού επιλέγεται με βάση το *σκορ καταλληλότητάς* (*fitness score*) του.

7.5.6 Βήμα-Βήμα Εκτέλεση του Γενετικού Αλγορίθμου

Σ' αυτήν την παράγραφο εφαρμόζουμε σταδιακά τον *γενετικό αλγόριθμο* που περιγράφηκε στην προηγούμενη ενότητα επί των δεδομένων του παραδείγματος 7.2, τα οποία και εμφανίζονται στον Πίνακα 7.13.

Παράδειγμα 7.2 Δίνεται ο Πίνακας 7.13, ο οποίος κρατά τις βαθμολογίες των χρηστών επί των στοιχείων και αποτελείται από 7 χρήστες και 9 στοιχεία. Έστω ότι αναζητούμε τη βέλτιστη *γειτονιά με τους κοντινότερους χρήστες του υπό εξέταση χρήστη* u_0. Θεωρούμε ότι η γειτονιά του υπό εξέταση χρήστη u_0 αποτελείται από ένα σύνολο 2 γειτονικών χρηστών (γειτονιά με-

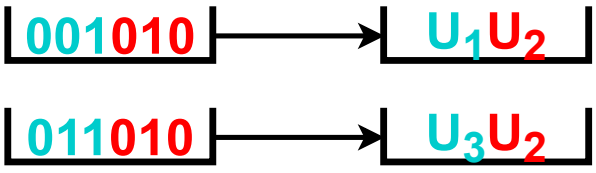
γέθους k=2). Έτσι, θέτουμε ένα σύνολο υποψήφιων γειτόνων $U = \{u_1, u_2, u_3, u_4, u_5, u_6\}$ και ένα σύνολο στοιχείων $I = \{i_1, i_2, i_3, i_4, i_5, i_6, i_7\}$ που θα αποτελέσουν τα δεδομένα εκπαίδευσης του μοντέλου πρόβλεψής μας. Κάθε χρήστης προσδιορίζεται από τις βαθμολογίες του για τα στοιχεία, και κάθε βαθμολογία βρίσκεται στο εύρος τιμών $[1 - 5]$. Σ' αυτό το παράδειγμα θέλουμε να προβλέψουμε τις βαθμολογίες του u_0 στα στοιχεία i_8 και i_9 (*δεδομένα ελέγχου*), οι οποίες εμφανίζονται με κόκκινα αγγλικά ερωτηματικά (?) στον Πίνακα 7.13.

	i_1	i_2	i_3	i_4	i_5	i_6	i_7	i_8	i_9
u_0	5	4	2	1	5	4	5	?	?
u_1	4	5	2	1	5	2	2	3	5
u_2	4	3	4	5	4	2	1	4	2
u_3	4	1	2	5	2	2	1	2	1
u_4	2	1	5	3	4	5	4	3	2
u_5	5	4	1	1	5	1	4	5	1
u_6	1	2	4	5	2	2	5	1	2

Σχήμα 7.13: Πίνακας βαθμολογίας Χρηστών-Στοιχείων, όπου επιχειρούμε να προβλέψουμε τη βαθμολογία του χρήστη u_0 στα στοιχεία i_8 και i_9.

Κωδικοποίηση Χρωμοσώματος

Για την αναπαράσταση της υποψήφιας γειτονιάς μεγέθους $k = 2$ του υπό εξέταση χρήστη u_0 θα υιοθετήσουμε μια απλή δυαδική κωδικοποίηση ενός χρωμοσώματος που αποτελείται από δύο γονίδια, ώστε να έχουμε ένα γονίδιο για κάθε υποψήφιο μέλος της γειτονιάς. Εφόσον έχουμε 6 πιθανούς γείτονες που μπορούν να συμπεριληφθούν στη γειτονιά των $k = 2$ κοντινότερων γειτόνων του υπό εξέταση χρήστη, αρκεί να αναπαραστήσουμε κάθε γονίδιο με 3 μόνο bit. Συνεπώς, το 001 μπορεί να αντιπροσωπεύει τον χρήστη u_1, το 010 τον χρήστη u_2, και ούτω καθεξής. Στο Σχήμα 7.14 εμφανίζουμε 2 παραδείγματα γειτονιών βάσει της κωδικοποίησης χρωμοσώματος που ακολουθούμε:



Σχήμα 7.14: Παραδείγματα της κωδικοποίησης ενός χρωμοσώματος που αποτελείται από 2 γονίδια που αναπαριστούν γειτονιές μεγέθους $k = 2$ γειτονικών χρηστών.

Χρησιμοποιώντας την παραπάνω κωδικοποίηση έχουμε 2^6= 64 διαφορετικές δυαδικές αναπαραστάσεις. Τονίζουμε ότι κατά τη διάρκεια της αξιολόγησης ενός χρωμοσώματος θα πρέπει να

εξαλείψουμε όλες τις μη αποδεκτές κωδικοποιήσεις. Λόγου χάρη, θα πρέπει να απορρίψουμε τις κωδικοποιήσεις που περιλαμβάνουν τους πλεονάζοντες γείτονες (π.χ. $001001 \rightarrow u_1u_1$). Από την άλλη μεριά, θα πρέπει να απορρίψουμε και τα χρωμοσώματα που δεν περιλαμβάνουν χρήστες που ανήκουν στο σύνολο χρηστών $U = \{u_1, u_2, .., u_6\}$ (π.χ. $111001 \rightarrow u_7u_1$, $u_7 \notin U$).

Συνάρτηση Καταλληλότητας (fitness function)

Προκειμένου να υπολογίσουμε την ομοιότητα του υπό εξέταση χρήστη u_0 με τους άλλους χρήστες, μπορούμε εμπειρικά να υποθέσουμε ότι, εφόσον δύο χρήστες βαθμολογούν παρόμοια κάποια στοιχεία, μπορεί να συνεχίσουν να έχουν ανάλογη (βαθμολογική) συμπεριφορά και για άλλα στοιχεία στο μέλλον. Μπορούμε να υιοθετήσουμε αυτήν την υπόθεση για να ορίσουμε ένα *μέτρο συνάφειας* ή αλλιώς *ομοιότητας* μεταξύ δύο χρηστών u_0 και u_c, όπως στην Εξίσωση 7.8:

$$Rel(u_0, u_c) = \frac{\sum_{\substack{|R_{0,i} - R_{c,i}| <= 1 \\ i \in I_{u_c} \cap I_{u_0}}} 1}{|I_{u_c} \cap I_{u_0}|} \tag{7.8}$$

όπου $R_{0,i}$ και $R_{c,i}$ οι βαθμολογίες των χρηστών u_0 και u_c στο στοιχείο i. Επίσης, τα I_{u_c} και I_{u_0} είναι τα σύνολα των στοιχείων που έχουν βαθμολογηθεί αντίστοιχα από τους χρήστες u_c και u_0. Ακόμη, στον αριθμητή της παραπάνω εξίσωσης μετράμε τον αριθμό των στοιχείων που είχαν βαθμολογήσει και οι δύο χρήστες, όπου διαπιστώνουμε ότι η βαθμολογία τους διέφερε το πολύ κατά μία μονάδα ($|R_{0,i} - R_{c,i}| <= 1$). Στον δε παρονομαστή της μετράμε τον αριθμό των στοιχείων που έχουν βαθμολογηθεί και από τους δύο χρήστες. Έτσι, οι τιμές *συνάφειας* (*relevance*) μεταξύ των δύο χρηστών κανονικοποιούνται στο διάστημα τιμών [0,1].

Εφαρμόζοντας, τώρα, την Εξίσωση 7.8 στα δεδομένα του Παραδείγματος 7.2 υπολογίζουμε το διάνυσμα ομοιότητας των κοντινότερων χρηστών του υπό εξέταση χρήστη u_0 όπως παρουσιάζεται στον Πίνακα 7.2. Παρατηρήστε ότι ο χρήστης u_5 έχει την υψηλότερη ομοιότητα, ίση με 0.85, με τον υπό εξέταση χρήστη u_0. Υπογραμμίζουμε ότι για τον υπολογισμό του διανύσματος ομοιοτήτων λαμβάνουμε υπόψη μόνο τις βαθμολογίες για τα στοιχεία $\{i_1, i_2, ...i_7\}$, ενώ τα στοιχεία i_8 και i_9 προορίζονται μόνο για να προβλέψουμε τη βαθμολογία τους.

	u_1	u_2	u_3	u_4	u_5	u_6
u_0	0.71	0.57	0.28	0.42	0.85	0.14

Πίνακας 7.2: Υπολογισμός της ομοιότητας των χρηστών ως προς τον υπό εξέταση χρήστη u_0. Η ομοιότητα ορίζεται στο διάστημα τιμών [0,1].

Επομένως, βασιζόμενοι στην Εξίσωση 7.8 μπορούμε να αξιολογήσουμε την ποιότητα ενός συγκεκριμένου χρωμοσώματος N υπολογίζοντας τη μέση ομοιότητα μεταξύ των δύο χρηστών του χρωμοσώματος και του υπό εξέταση χρήστη u_0, όπως φαίνεται στην παρακάτω *συνάρτηση καταλληλότητας* (*fitness function*):

$$Fitness\_function(u_0, N) = \begin{cases} 0, & N \text{ περιέχει απορριπτέες κωδικοποιήσεις χρηστών} \\ \frac{1}{|N|} \sum_{v \in N} Rel(v, u_0), & \text{διαφορετικά} \end{cases} \tag{7.9}$$

όπου N το μέγεθος της υπό εξέταση γειτονιάς του χρήστη u_0 ή αλλιώς το μέγεθος του υπό εξέταση χρωμοσώματος, το οποίο αποτελείται από δύο γονίδια (από δύο χρήστες δηλαδή).

Υπογραμμίζουμε ότι γι' έναν αριθμό n χρηστών ο συνολικός αριθμός των πιθανών ζευγαριών των χρηστών ανά δύο που μπορεί να παραχθεί είναι $\frac{n!}{2!(n-2)!}$. Δηλαδή, ισχύει $\begin{pmatrix} n \\ 2 \end{pmatrix} = \frac{n!}{2!(n-2)!}$. Στην περίπτωσή μας (Παράδειγμα 7.2), τα ζευγάρια χρηστών που μπορούν να συγκριθούν είναι 15 δεδομένου ότι έχουμε έξι υποψήφιους χρήστες $U = \{u_1, u_2, u_3, u_4, u_5, u_6\}$. Μ' άλλα λόγια, ισχύει $\begin{pmatrix} 6 \\ 2 \end{pmatrix} = \frac{6!}{2!(6-2)!} = 15$.

Μετά τη δημιουργία όλων των πιθανών συνδυασμών των χρηστών ανά δύο (ή αλλιώς τη δημιουργία όλων των πιθανών χρωμοσωμάτων), χρησιμοποιούμε από την Εξίσωση 7.9 τη *συνάρτηση καταλληλότητας* (*fitness function*) -το ανάλογο *σκορ* της οποίας (*fitness score*) φαίνεται υπολογισμένο στον Πίνακα 7.3-, για να αξιολογήσουμε την ποιότητα κάθε πιθανής γειτονιάς χρηστών ως προς τον υπό εξέταση χρήστη u_0.

Στη συνέχεια, εφαρμόζοντας τον ψευδοκώδικα του Αλγορίθμου 4 στα δεδομένα του Παραδείγματος 7.2, προκύπτει ο Πίνακας 7.4 που καταγράφει τις τιμές των κύριων γενετικών πράξεων. Σύμφωνα λοιπόν με τον Αλγόριθμο 4, αρχικά επιλέγουμε τυχαία τα ζευγάρια των χρηστών (χρωμοσώματα) που θα συμπεριληφθούν στον αρχικό πληθυσμό, ο οποίος εμφανίζεται στην 1η γραμμή του Πίνακα 7.4. Καθώς έχουμε 4 γειτονιές χρηστών (4 χρωμοσώματα) στον αρχικό πληθυσμό, επιλέγουμε τα 2 πιο κατάλληλα χρωμοσώματα (αυτά που έχουν το υψηλότερο fitness score) για να τα αναπαράξουμε, όπως φαίνεται στη 2η γραμμή του Πίνακα 7.4. Στην 1η γενιά, η οποία εμφανίζεται στη γραμμή 2 του Πίνακα 7.4.

Οι απόγονοι δημιουργούνται με την εφαρμογή των τελεστών *διασταύρωσης* και *μετάλλαξης*, οι οποίοι περιγράφονται αναλυτικά παρακάτω:

- **Τελεστής διασταύρωσης**: Η γειτονιά N_{10} είναι ο συνδυασμός των u_3 και u_4 (δηλαδή $N_{10} \leftarrow \{u_3, u_4\}$.), ενώ N_5 είναι ο συνδυασμός των u_1 και u_6 (δηλαδή $N_5 \leftarrow \{u_1, u_6\}$.). Χρησιμοποιώντας τον τελεστή *διασταύρωσης* συνδυάζουμε u_1 και u_4 για να δημιουργήσουμε τη γειτονιά N_3, ενώ ομοίως συνδυάζουμε u_3 και u_6 για να δημιουργήσουμε τη γειτονιά N_{12}.

- **Τελεστής Μετάλλαξης**: Με *μετάλλαξη* μπορούμε να δημιουργήσουμε από τον συνδυασμό $N_3 \leftarrow \{u_1, u_4\}$ τον συνδυασμό $N_2 \leftarrow \{u_1, u_3\}$ όπου ο u_4 μεταλλάσσεται σε u_3. Ανάλογα δημιουργούμε τη γειτονιά $N_9 \leftarrow \{u_2, u_6\}$ από τον συνδυασμό $N_{12} \leftarrow \{u_3, u_6\}$ όπου το u_3 μεταλλάσσεται σε u_2.

Με παρόμοιο τρόπο λοιπόν εφαρμόζονται οι γενετικές πράξεις για την παραγωγή της 2ης και της 3ης γενιάς αντίστοιχα, ώσπου να καταλήξουμε στη βέλτιστη γενιά (βλ. την τελευταία γραμμή

n	Χρωμόσωμα	Γονίδια	Fitness score
1	N_1	$\{u_1, u_2\}$	0.64
2	N_2	$\{u_1, u_3\}$	0.49
3	N_3	$\{u_1, u_4\}$	0.56
4	N_4	$\{u_1, u_5\}$	0.78
5	N_5	$\{u_1, u_6\}$	0.42
6	N_6	$\{u_2, u_3\}$	0.42
7	N_7	$\{u_2, u_4\}$	0.49
8	N_8	$\{u_2, u_5\}$	0.71
9	N_9	$\{u_2, u_6\}$	0.35
10	N_{10}	$\{u_3, u_4\}$	0.35
11	N_{11}	$\{u_3, u_5\}$	0.56
12	N_{12}	$\{u_3, u_6\}$	0.21
13	N_{13}	$\{u_4, u_5\}$	0.63
14	N_{14}	$\{u_4, u_6\}$	0.28
15	N_{15}	$\{u_5, u_6\}$	0.49

Πίνακας 7.3: Για τα δεδομένα του Παραδείγματος 7.2 έχουμε σχηματίσει συνδυασμούς χρηστών μεγέθους $k = 2$ (οι λεγόμενες γειτονιές) για τους 6 υποψήφιους χρήστες και το αντίστοιχο Fitness score τους ως προς τον υπό εξέταση χρήστη u_0.

Γενιά	Πληθυσμός	Επιλογή	Διασταύρωση	Μετάλλαξη	Fitness score
τυχαία	$\{N_{10}, N_5, N_{12}, N_{14}\}$	-	-	-	0.31
1	$\{N_{10}, N_5, N_{12}, N_{14}\}$	$\{N_{10}, N_5\}$	$\{N_3, N_{12}\}$	$\{N_2, N_9\}$	0.31
2	$\{N_{10}, N_5, N_2, N_9\}$	$\{N_2, N_5\}$	$\{N_2, N_5\}$	$\{N_6, N_3\}$	0.40
3	$\{N_3, N_6, N_2, N_5\}$	$\{N_3, N_6\}$	$\{N_7, N_2\}$	$\{N_8, N_2\}$	0.47
βέλτιστη	$\{N_3, N_6, N_2, N_8\}$	-	-	-	0.54

Πίνακας 7.4: Βελτιστοποίηση της *γειτονιάς των κοντινότερων χρηστών του υπό εξέταση χρήστη* του Παραδείγματος 7.2 (m=3).

του Πίνακα 7.4). Σημειώστε ότι στην τελευταία στήλη του Πίνακα 7.4 η μέση βαθμολογία καταλληλότητας του πληθυσμού βελτιώνεται σιγά σιγά όσο παράγονται νέες γενιές. Τέλος, σύμφωνα με τη γραμμή εντολών 4 του Αλγορίθμου 4 το χρωμόσωμα N_8 θα επιλεχθεί ως η βέλτιστη γειτονιά του υπό εξέταση χρήστη u_0, δεδομένου ότι έχει το μεγαλύτερο fitness score έναντι των υπολοίπων χρωμοσωμάτων του βελτιστοποιημένου πληθυσμού, ο οποίος εμφανίζεται στην τελευταία γραμμή του Πίνακα 7.4.

7.6 Ερωτήσεις Κεφαλαίου

1. Ποια είναι η κύρια διαφορά μεταξύ της *Μαρκοβιανής Αλυσίδας* και της *Μαρκοβιανής Διαδικασίας Απόφασης*; Να αναπαραστήσετε σχηματικά την καθεμία με τη χρήση ενός σχετικού παραδείγματος.

2. Να αναπαραστήσετε σχηματικά την αλληλεπίδραση μεταξύ *ευφυούς πράκτορα* και *περιβάλλοντος* στα συστήματα συστάσεων. Περιγράψτε τα σχετικά βήματα.

3. Ποια είναι τα βασικά δομικά στοιχεία της *Ενισχυτικής Μάθησης* για τα συστήματα συστάσεων; Περιγράψτε σύντομα το καθένα από αυτά.

4. Να ορίσετε τους *on* και *off-policy αλγορίθμους ενισχυτικής μάθησης*. Ποια τα πλεονεκτήματα και τα μειονεκτήματα της κάθε κατηγορίας;

5. Να αναλύσετε τα βασικά βήματα του *αλγορίθμου Q-learning*. Τί είναι η *αξία-Q*;

6. Ποιος είναι ο μαθηματικός τύπος της εξίσωσης του *σφάλματος χρονικής διαφοράς* στην *ενισχυτική μάθηση*; Περιγράψτε τους σημαντικότερους όρους και μεταβλητές της εξίσωσης.

7. Ποιος είναι ο μαθηματικός τύπος της *εξίσωσης Bellman*; Περιγράψτε τους σημαντικότερους όρους και μεταβλητές της εξίσωσης.

8. Να παρουσιάσετε τα βασικά βήματα του *αλγορίθμου Advantage Actor Critic*. Τί είναι η *αξία της κατάστασης V*;

9. Τί εννοούμε με τον όρο *βαθιά ενισχυτική μάθηση*; Δώστε σχετικά παραδείγματα.

10. Να αναφερθείτε στα βασικά βήματα του *αλγορίθμου Deep Q-Network*.

11. Ποια είναι τα βασικά δομικά συστατικά των *γενετικών αλγορίθμων*; Περιγράψτε συνοπτικά το κάθε στοιχείο ξεχωριστά.

12. Ποιοι είναι οι βασικοί τελεστές των *γενετικών πράξεων*; Αναφερθείτε συνοπτικά στον κάθε τελεστή ξεχωριστά.

13. Να περιγράψετε τα βασικά βήματα του *γενικού γενετικού αλγορίθμου* για τα συστήματα συστάσεων είτε με τη βοήθεια ενός σχήματος είτε με τη χρήση ψευδοκώδικα.

7.7 Ασκήσεις Προγραμματισμού

7.7.1 Άσκηση Υλοποίησης του Αλγορίθμου Q-learning για Εφαρμογή στα Συστήματα Συστάσεων

Δίνεται ο πίνακας βαθμολογιών χρηστών-στοιχείων A του Πίνακα 7.5. Όπως απεικονίζεται στον Πίνακα 7.5, I_{0-5} είναι τα στοιχεία και U_{0-5} είναι οι χρήστες. Να εκτελέσετε τον *αλγόριθμο Q-learning* και να υπολογίσετε τον *πίνακα αξιών-Q*.

	I_0	I_1	I_2	I_3	I_4	I_5
U_0	5	1	1	4	0	0
U_1	1	5	4	0	0	4
U_2	2	1	5	0	0	4
U_3	1	2	1	5	0	0
U_4	5	1	2	0	1	0
U_5	1	5	4	0	4	0

Πίνακας 7.5: Δισδιάστατος Πίνακας βαθμολογιών Χρηστών-Στοιχείων A (6×6).

Λύση:
Στον Πίνακα 7.6 παρουσιάζεται ο τελικός πίνακας Q με τις προβλέψεις του ($\lambda = 0.25$, $\gamma = 0.75$, β=0.1 και *αριθμός επαναλήψεων* = 100). Ακολουθεί ο κώδικας που δίνει αυτά τα αποτελέσματα καθώς και αναλυτική περιγραφή του.

	I_0	I_1	I_2	I_3	I_4	I_5
U_0	17.11	3.77	1.61	0	0	4.23
U_1	2.89	1.654	0	0	0	0
U_2	7.26	0	2.01	1.36	0.48	0
U_3	3.87	1.79	2.12	1.95	0	0
U_4	19.71	3.61	3.98	0	7.15	3.27
U_5	0.70	0	0	0	10.70	0

Πίνακας 7.6: Πίνακας με τις προβλεφθείσες *αξίες-Q* μετά από 100 επαναλήψεις.

Μας δίνεται, λοιπόν, μια δομή δεδομένων ενός *dictionary* που αποτελεί το περιβάλλον με το οποίο αλληλεπιδρά ο *ευφυής πράκτορας*. Το *dictionary env* αποτελείται από 6 γραμμές (χρήστες 0 έως 5) και 6 στήλες (στοιχεία 0 έως 5). Για κάθε συνδυασμό χρήστη-στοιχείου καταγράφουμε τη βαθμολογία του πρώτου για το δεύτερο σε κλίμακα 0 έως 5, καθώς επίσης

σημειώνουμε με τη χρήση μιας *boolean μεταβλητής* {*True ή False*} και το εάν ο χρήστης τελικά αγόρασε το στοιχείο αυτό. Επίσης, ορίζουμε τη δομή ενός πίνακα που αρχικοποιούμε με μηδενικές τιμές, και με τον οποίο θα υπολογίσουμε τις *αξίες-Q*. Τονίζουμε ότι οι τελευταίες θα προβλέπουν για το παράδειγμά μας το κατά πόσο η αγορά ή μη ενός στοιχείου θα ενδιέφερε έναν χρήστη.

```
import numpy as np
import random
from IPython.display import clear_output

env = [{0: (5, True), 1: (1, False), 2:(1, False), 3:(4, False)
    , 4:(0, False), 5:(0, False)},
       {0: (1, False), 1:(5, True),  2:(4, False), 3:(0, False)
    , 4:(0, False), 5:(4, False)},
       {0: (2, False), 1:(1, False), 2:(5, True),  3:(0, False)
    , 4:(0, False), 5:(4, False)},
       {0: (1, False), 1:(2, False), 2:(1, False), 3:(5, True),
      4:(0, False), 5:(0, False)},
       {0: (5, True),  1:(1, False), 2:(2, False), 3:(0, False)
    , 4:(1, False), 5:(0, False)},
       {0: (1, False), 1:(5, True),  2:(4, False), 3:(0, False)
    , 4:(4, False), 5:(0, False)}
      ]
print(env)
print("\n")

q_table = np.zeros([6, 6])
print(q_table)
```

Παρακάτω "εκτυπώνεται" ο *Q-table*, ο οποίος αρχικοποιείται με μηδενικές τιμές:

```
[[0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0.]]
```

Στη συνέχεια ορίζουμε τις τιμές των βασικών παραμέτρων του *αλγορίθμου Q-learning* (βλ. Αλγόριθμο 3 - υποενότητα *Βήμα-Βήμα εκτέλεση του Αλγορίθμου Q-learning*), ο οποίος θα εκτελεστεί για 100 επαναλήψεις:

```

lamda = 0.25 # ποσοστό μάθησης
gamma = 0.75 # συντελεστής εξασθένισης μελλοντικών ανταμοιβών
beta = 0.10  # προσδιορίζει πόσο συχνά θα επιλέγουμε μια ενέργεια με τυχαίο τρόπο

for i in range(1, 100):

    print("=====Iteration ",i," =========")

    state = random.randrange(0,6,1) # επίλεξε τυχαία μια κατάσταση ή αλλιώς χρήστη, δηλαδή μια γραμμή από τον πίνακα Q
    print("randomly selected state=",state)
    terminal_state = False

    while not terminal_state:

        if random.uniform(0, 1) < beta:
            action = random.randrange(0,6,1)
            # επίλεξε τυχαία μια ενέργεια, δηλαδή μια στήλη από τον πίνακα Q
            print("randomly selected action =",action)
        else:
            action = np.argmax(q_table[state]) # Επίλεξε τη βέλτιστη ενέργεια
            print("selected max action that gives the highest reward=",action)

        reward, terminal_state = env[state][action]
        print("reward=",reward)

        q_value = q_table[state, action]
        print("qvalue_old=",q_value)
```

```
29
30          next_state = (state + 1 if state + 1 < 6 else 0)
31          max_future_reward = np.max(q_table[state])
32
33          #Q(s_{t}, a_{t})=(1-\lambda) Q(s_{t}, a_{t})+\lambda *(
    R_{t}+\gamma* \max _{a} Q(s_{t+1}, a))
34          next_q_value  = ((1 - lamda) * q_value) + (lamda *(
    reward + gamma * max_future_reward))
35          print("qvalue_new=",next_q_value)
36
37          q_table[state, action] = next_q_value
38
39          state = next_state
40          print("new state =",state)
41
42 print("Τελικός πίνακας Q \n")
43 print(q_table)
```

```
1
2 =====Iteration  1  =========
3 randomly selected state= 3
4 selected max action that gives the highest reward= 0
5 reward= 1
6 qvalue_old= 0.0
7 qvalue_new= 0.25
8 new state = 4
9 selected max action that gives the highest reward= 0
10 reward= 5
11 qvalue_old= 0.0
12 qvalue_new= 1.25
13 new state = 5
14 =====Iteration  2  =========
15 randomly selected state= 0
16 selected max action that gives the highest reward= 0
17 reward= 5
18 qvalue_old= 0.0
```

```
qvalue_new= 1.25
new state = 1
=====Iteration  3  =========
randomly selected state= 0
selected max action that gives the highest reward= 0
reward= 5
qvalue_old= 1.25
qvalue_new= 2.421875
new state = 1
.....
.....
=====Iteration  99  =========
randomly selected state= 3
selected max action that gives the highest reward= 0
reward= 1
qvalue_old= 3.869221111928994
qvalue_new= 3.877394792433432
new state = 4
selected max action that gives the highest reward= 0
reward= 5
qvalue_old= 19.698586318419636
qvalue_new= 19.71742467351841
new state = 5

 #Τελικός πίνακας Q

[[17.11485073  3.77600558  1.6131221   0.          0.
   4.23137609]
 [ 2.89976484  1.65428631  0.          0.          0.
   0.        ]
 [ 7.2654401   0.          2.01247746  1.36227002  0.48159876
   0.        ]
 [ 3.87739479  1.78963146  2.12786962  1.95616738  0.
   0.        ]
 [19.71742467  3.60823472  3.98401717  0.          7.15247425
   3.27454205]
```

```
[ 0.70410156  0.          0.          0.         10.70589661
  0.        ]]
```

Έπειτα, ο παρακάτω κώδικας αξιολογεί τις προβλέψεις του *αλγορίθμου Q-learning* ως προς τη σύσταση του στοιχείου που πρέπει να προταθεί προς πώληση στον υπό εξέταση χρήστη. Γίνονται συνολικά 100 προβλέψεις:

```
"""Αξιολόγηση του μοντέλου πρόβλεψης του Q-learning"""
n_predictions = 100
n_wrong_predictions = 0

for _ in range(n_predictions):
    state = random.randrange(0,6,1)
    act = np.argmax(q_table[state])
    reward, terminal_state = env[state][act]
    if reward <5:
        n_wrong_predictions += 1
print(f"{n_predictions - n_wrong_predictions} σωστές προβλέψεις
  σε σύνολο {n_predictions} προβλέψεων.")
```

Τέλος, παρακάτω "εκτυπώνεται" το αποτέλεσμα που δείχνει την *ακρίβεια* (*precision*) των προβλέψεων για 100 επαναλήψεις:

- 29 σωστές προβλέψεις σε σύνολο 100 προβλέψεων.

Τονίζουμε ότι, εάν εκπαιδεύσουμε το μοντέλο μας με περισσότερες επαναλήψεις (π.χ. 1000 ή, ακόμη περισσότερο, 10000 επαναλήψεις), τότε θα αυξηθεί δραστικά η ακρίβεια και η αποτελεσματικότητα των προβλέψεών μας.

Βιβλιογραφία

S. Bag, A. Ghadge, and M. K. Tiwari. An integrated recommender system for improved accuracy and aggregate diversity. *Computers & Industrial Engineering*, 130:187--197, 2019.

C. E. Berbague, N. E. islem Karabadji, H. Seridi, P. Symeonidis, Y. Manolopoulos, and W. Dhifli. An overlapping clustering approach for precision, diversity and novelty-aware recommendations. *Expert Systems with Applications*, 177:114917, 2021. ISSN 0957-4174. https://doi.org/10.1016/j.eswa.2021.114917. URL https://www.sciencedirect.com/science/article/pii/S0957417421003584.

J. Bobadilla, F. Ortega, A. Hernando, and J. Alcalá. Improving collaborative filtering recommender system results and performance using genetic algorithms. *Knowledge-Based Systems*, 24(8):1310--1316, 2011. 10.1016/j.knosys.2011.06.005. URL https://doi.org/10.1016/j.knosys.2011.06.005.

S. Chen, J. L. Moore, D. Turnbull, and T. Joachims. Playlist prediction via metric embedding. In *The 18th ACM International Conference on Knowledge Discovery and Data Mining, KDD'12, Beijing, China*, pages 714--722. ACM, 2012. 10.1145/2339530.2339643. URL http://doi.acm.org/10.1145/2339530.2339643.

C. Esiyok, B. Kille, B. J. Jain, F. Hopfgartner, and S. Albayrak. Users' reading habits in online news portals. In *Fifth Information Interaction in Context Symposium, IIiX '14, Regensburg, Germany*, pages 263--266. ACM, 2014. 10.1145/2637002.2637038. URL http://doi.acm.org/10.1145/2637002.2637038.

F. Garcin, C. Dimitrakakis, and B. Faltings. Personalized news recommendation with context trees. In *Proceedings of the 7th ACM conference on Recommender systems, RecSys'13, Hong Kong, China*, pages 105--112. ACM, 2013. 10.1145/2507157.2507166. URL http://doi.acm.org/10.1145/2507157.2507166.

D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc., USA, 1st edition, 1989. ISBN 0201157675.

D. E. Goldberg. A note on boltzmann tournament selection for genetic algorithms and population-oriented simulated annealing. *Complex Syst.*, 4, 1990.

H. Hasselt. Double q-learning. *Advances in neural information processing systems*, 23, 2010.

Q. He, D. Jiang, Z. Liao, S. C. H. Hoi, K. Chang, E. Lim, and H. Li. Web query recommendation via sequential query prediction. In *Proceedings of the 25th International Conference on Data Engineering, ICDE'09, Shanghai, China*, pages 1443--1454. IEEE, 2009. 10.1109/ICDE.2009.71. URL https://doi.org/10.1109/ICDE.2009.71.

T. Hester, M. Vecerik, O. Pietquin, M. Lanctot, T. Schaul, B. Piot, D. Horgan, J. Quan, A. Sendonaris, I. Osband, et al. Deep q-learning from demonstrations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

J. H. Holland. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press, 1992.

R. Katarya and O. P. Verma. A collaborative recommender system enhanced with particle swarm optimization technique. *Multimedia Tools and Applications*, 75(15):9225--9239, 2016. 10.1007/s11042-016-3481-4. URL https://doi.org/10.1007/s11042-016-3481-4.

V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928--1937. PMLR, 2016.

O. Moling, L. Baltrunas, and F. Ricci. Optimal radio channel recommendations with explicit and implicit feedback. In *Sixth ACM Conference on Recommender Systems, RecSys'12, Dublin, Ireland*, pages 75--82. ACM, 2012. 10.1145/2365952.2365971. URL http://doi.acm.org/10.1145/2365952.2365971.

M. Rahman and M. Islam. A hybrid clustering technique combining a novel genetic algorithm with k-means. *Knowledge-Based Systems*, 71:345--365, 2014. 10.1016/j.knosys.2014.08.011. URL https://doi.org/10.1016/j.knosys.2014.08.011.

S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of the 19th International Conference on World Wide Web, WWW'10, Raleigh, North Carolina, USA*, pages 811--820. ACM, 2010. 10.1145/1772690.1772773. URL http://doi.acm.org/10.1145/1772690.1772773.

G. Shani, D. Heckerman, and R. I. Brafman. An mdp-based recommender system. In *Journal of Machine Learning Research (2005)*, volume 6, pages 1265--1295. JMLR, 2005. URL http://www.jmlr.org/papers/v6/shani05a.html.

M. Tavakolifard, J. A. Gulla, K. C. Almeroth, F. Hopfgartner, B. Kille, T. Plumbaum, A. Lommatzsch, T. Brodt, A. Bucko, and T. Heintz. Workshop and challenge on news recommender systems. In *Seventh ACM Conference on Recommender Systems, RecSys '13, Hong Kong, China*, pages 481--482. ACM, 2013. 10.1145/2507157.2508004. URL http://doi.acm.org/10.1145/2507157.2508004.

M. Zhang and N. Hurley. Novel item recommendation by user profile partitioning. In *2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology*. IEEE, 2009. 10.1109/wi-iat.2009.85. URL https://doi.org/10.1109/wi-iat.2009.85.