ε.δε.μ²

ΕΠΙΣΤΗΜΗ ΔΕΔΟΜΕΝΩΝ & ΜΗΧΑΝΙΚΗ ΜΑΘΗΣΗ

ΣΤΟΧΑΣΤΙΚΕΣ ΔΙΕΡΓΑΣΙΕΣ & ΒΕΛΤΙΣΤΟΠΟΙΗΣΗ

Αίθουσα 005 - Νέα Κτίρια ΣΗΜΜΥ Ε.Μ.Π.

Δυναμικός Προγραμματισμός με Μεθόδους Monte Carlo:
1. Μάθηση Χρονικών Διαφορών (Temporal-Difference Learning)
2. Στοχαστικός Αλγόριθμος Q-Learning

καθ. Βασίλης Μάγκλαρης

maglaris@netmode.ntua.gr

www.netmode.ntua.gr

Πέμπτη 16/5/2019

Αλγόριθμος Policy Iteration (1/2) (*Επανάληψη*)

Ορισμός Q-factor

Έστω χρονοσταθερή πολιτική $\pi = \{\mu, \mu, \dots\}$ που οδηγεί σε γνωστά ***costs-to-go*** $J^{\mu}(i), \forall i \in \mathcal{X}$ (καταστάσεις του ***περιβάλλοντος***) με αποφάσεις του ***agent*** $a = \mu(i) \in \mathcal{A}_i$

Για κάθε ζεύγος (i, a) στο υπό εξέταση βήμα και πολιτική για τα υπολειπόμενα βήματα $\pi = \{\mu, \mu, \dots\}$ ορίζω τους ***Q-factors*** σαν μέτρο κατάταξης εναλλακτικών άμεσων αποφάσεων $a \in \mathcal{A}_i$ του ***agent***

$$Q^{\mu}(i, a) \triangleq c(i, a) + \gamma \sum_{j=1}^{N} p_{ij}(a) J^{\mu}(j)$$

Μια πολιτική $\pi = \{\mu, \mu, \dots\}$ ικανοποιεί τις συνθήκες απληστίας (***greedy conditions***) σε σχέση με τα ***costs-to-go*** $J^{\mu}(i)$ όταν

$$Q^{\mu}(i, \mu(i)) = \min_{a \in \mathcal{A}_i} Q^{\mu}(i, a)$$

Μια πολιτική $\pi^* = \{\mu^*, \mu^*, \dots\}$ είναι βέλτιστη αν ικανοποιεί τις συνθήκες απληστίας (***greedy conditions***) του δυναμικού προγραμματισμού:

$$Q^{\mu^*}(i, \mu^*(i)) = \min_{a \in \mathcal{A}_i} Q^{\mu^*}(i, a)$$

Σημείωση: Όταν τα άμεσα αναμενόμενα κόστη $c(i, a)$ αντικαθίστανται από ***rewards*** $r(i, a)$, τα ***costs-to-go*** $J^{\mu}(i)$ αποκαλούνται ***Value Functions*** $V^{\mu}(i)$ και έχουμε κατ' αντιστοιχία:

$$Q^{\mu}(i, a) \triangleq r(i, a) + \gamma \sum_{j=1}^{N} p_{ij}(a) V^{\mu}(j) \text{ και } Q^{\mu^*}(i, \mu^*(i)) = \max_{a \in \mathcal{A}_i} Q^{\mu^*}(i, a)$$

ΣΤΟΧΑΣΤΙΚΕΣ ΔΙΑΔΙΚΑΣΙΕΣ & ΒΕΛΤΙΣΤΟΠΟΙΗΣΗ

Αλγόριθμος Policy Iteration (2/2) (*Επανάληψη*)

Αλγόριθμος Reinforcement Learning

(Αρχιτεκτονική ***Actor – Critic***)

Επαναλήψεις $n = 1,2,\ldots$ από δύο βήματα μέχρι σύγκλισης πολιτικής $\pi_n = \pi_{n+1}$

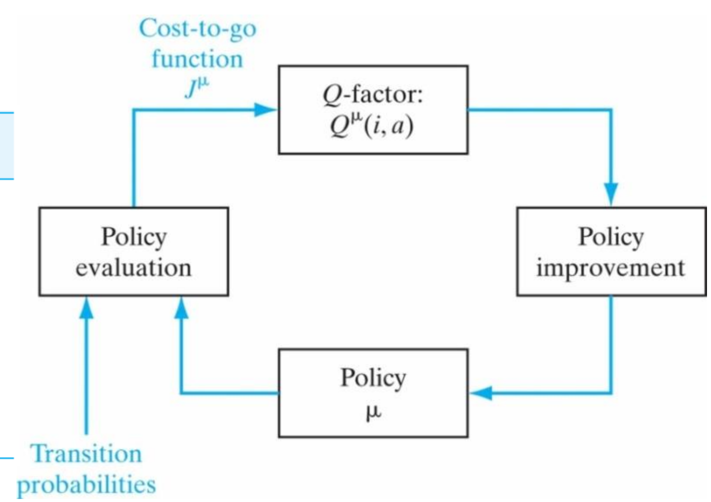
Βήμα 1. **Policy Evaluation** (ο ***critic*** αναλύει τις αποφάσεις του ***agent***):
Με βάση την παρούσα πολιτική $\pi_n = \{\mu_n, \mu_n, \ldots\}$ υπολογίζονται τα ***costs-to-go***
$J^{\mu_n}(i) = c(i,a) + \gamma \sum_{j=1}^{N} p_{ij}(a) J^{\mu_n}(j)$ για $i = 1,2,\ldots,N$
και οι **Q-factors** $Q^{\mu_n}(i,a) = c(i,a) + \gamma \sum_{j=1}^{N} p_{ij}(a) J^{\mu_n}(j)$ για $i = 1,2,\ldots,N$ και $a \in \mathcal{A}_i$

Βήμα 2. **Policy Improvement** (ο ***actor*** καθοδηγεί τις αποφάσεις του ***agent***):
Η πολιτική π_n βελτιώνεται σε π_{n+1} μέσω της $\mu_{n+1}(i) = \arg\min_{a \in \mathcal{A}_i} Q^{\mu_n}(i,a)$ για $i = 1,2,\ldots,N$

$\arg\min_x f(x)$: Η τιμή της x που οδηγεί την $f(x)$ σε ελάχιστο

TABLE 12.1 Summary of the Policy Iteration Algorithm

1. Start with an arbitrary initial policy μ_0.
2. For $n = 0, 1, 2, \ldots$, compute $J^{\mu_n}(i)$ and $Q^{\mu_n}(i, a)$ for all states $i \in \mathscr{X}$ and actions $a \in \mathscr{A}_i$.
3. For each state i, compute
$$\mu_{n+1}(i) = \arg\min_{a \in \mathscr{A}_i} Q^{\mu_n}(i, a)$$
4. Repeat steps 2 and 3 until μ_{n+1} is not an improvement on μ_n, at which point the algorithm terminates with μ_n as the desired policy.



Ο αλγόριθμος συγκλίνει σε βέλτιστη πολιτική σε πεπερασμένα βήματα n λόγω πεπερασμένου πλήθους καταστάσεων N και επιλογών αποφάσεων

Value Iteration Algorithm (*Επανάληψη*)

Εκτίμηση των Συναρτήσεων Cost-to-Go μέσω Διαδοχικών Προσεγγίσεων $J_n(i) \to J_{n+1}(i)$

- Εκκίνηση με αυθαίρετες τιμές $J_0(i)\ \ \forall i$
- Επαναλήψεις $n \to n+1$ μέχρι ***ανεκτή σύγκλιση*** (θεωρητικά $n \to \infty$) μέσω σχέσεων ***backup***:

$$J_{n+1}(i) = \min_{a\in\mathcal{A}_i}\left\{c(i,a) + \gamma \sum_{j=1}^{N} p_{ij}(a)\, J_n(j)\right\} \text{ για } i = 1,2,\dots,N \quad \text{(από εξισώσεις } \textbf{\textit{Bellman}}\text{)}$$

- Τελικός υπολογισμός των βέλτιστων ***Costs-to-Go***

$$J^*(i) = \lim_{n\to\infty} J_n(i),\ \ Q^*(i,a) = c(i,a) + \gamma \sum_{j=1}^{N} p_{ij}(a)\, J^*(j)$$

και προσδιορισμός της ***βέλτιστης πολιτικής*** $\mu^*(i) = \arg\min_{a\in\mathcal{A}_i} Q^*(i,a)$ για $i = 1,2,\dots,N$

TABLE 12.2 Summary of the Value Iteration Algorithm

1. Start with arbitrary initial value $J_0(i)$ for state $i = 1, 2, \dots, N$.
2. For $n = 0, 1, 2, \dots,$ compute

$$J_{n+1}(i) = \min_{a\in\mathcal{A}_i}\left\{c(i,a) + \gamma \sum_{j=1}^{N} p_{ij}(a) J_n(j),\right\}, \qquad \begin{array}{l} a \in \mathcal{A}_i \\ i = 1, 2, \dots, N \end{array}$$

   Continue this computation until

$$|J_{n+1}(i) - J_n(i)| < \epsilon \qquad \text{for each state } i$$

   where ϵ is a prescribed tolerance parameter. It is presumed that ϵ is sufficiently small for $J_n(i)$ to be close enough to the optimal cost-to-go function $J^*(i)$. We may then set

$$J_n(i) = J^*(i) \qquad \text{for all states } i$$

3. Compute the Q-factor

$$Q^*(i,a) = c(i,a) + \gamma \sum_{j=1}^{N} p_{ij}(a) J^*(j) \qquad \begin{array}{l} \text{for } a \in \mathcal{A}_i \text{ and} \\ i = 1, 2, \dots, N \end{array}$$

   Hence, determine the optimal policy as a greedy policy for $J^*(i)$:

$$\mu^*(i) = \arg\min_{a\in\mathcal{A}_i} Q^*(i,a)$$

Ο αλγόριθμος ***Value Iteration*** συνήθως συγκλίνει ικανοποιητικά και θεωρείται αποτελεσματικότερος του ***Policy Iteration*** καθώς αποφεύγει υπολογισμούς όλων των ***Costs-to-Go*** $J^{\mu_n}(i)$ σε κάθε βήμα

Παράδειγμα Δυναμικού Προγραμματισμού: Βελτιστοποίηση Δρομολόγησης (*Επανάληψη*)

Εύρεση Δρόμων Ελάχιστου Κόστους από Κόμβο A σε Κόμβο J μέσω του μονο-κατευθυντικού γράφου όπως στο σχήμα με κατεύθυνση γραμμών ***Δ → A***

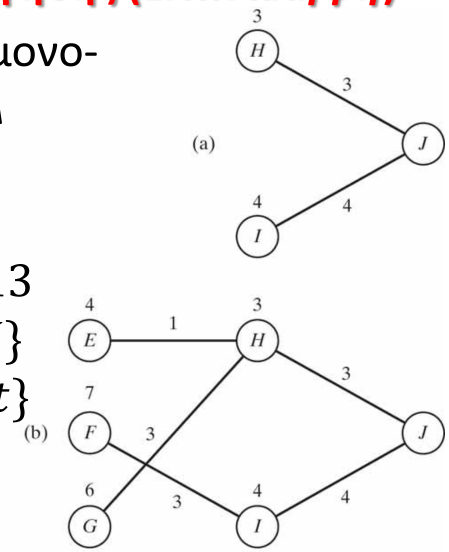
Ενδεικτικό κόστος γραμμών: $A \to B{:}\, 2,\ B \to A{:}\, \infty$

$B \to F{:}\, 4,\ F \to B{:}\, \infty$

Ενδεικτικό κόστος δρόμου: Δρόμος $\{A, B, F, I, J, Q\}$: $2 + 4 + 3 + 4 = 13$

Κατάσταση Περιβάλλοντος: Κόμβος σε παρούσα διερεύνηση $\{A, B, \ldots, J\}$

Αποφάσεις Agent: Επόμενος κόμβος για διερεύνηση $\{up, down, staight\}$



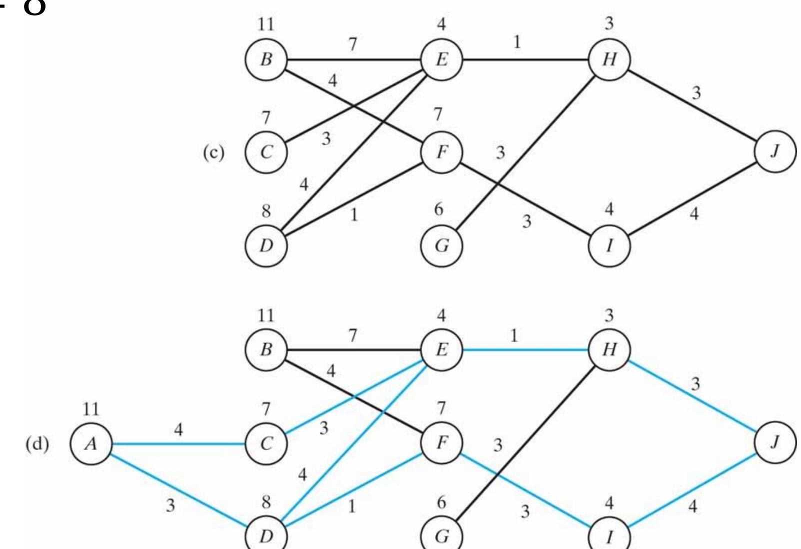
Αναδρομικός Υπολογισμός *Q-Factors*:

$Q(H, down) = 3 \quad Q(I, up) = 4$

$Q(E, staight) = 1 + 3 = 4 \quad Q(E, down) = 4 + 4 = 8$

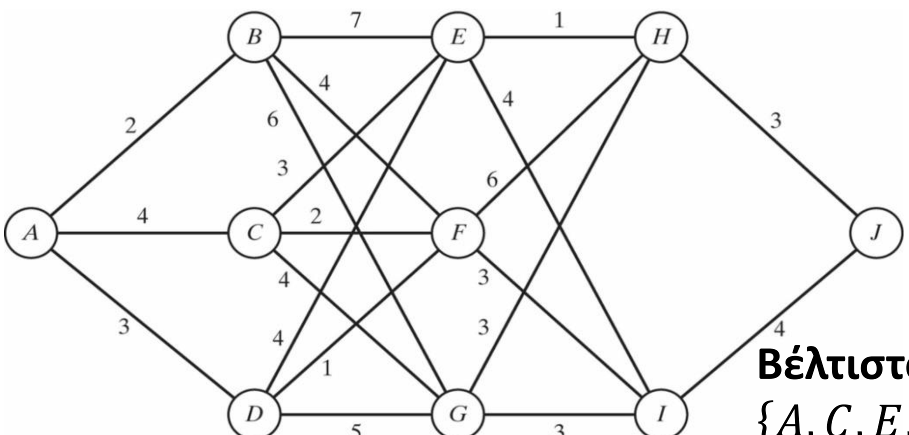
$Q(F, up) = 6 + 3 = 9 \quad Q(F, down) = 3 + 4 = 7$

.....



Κατεύθυνση Γραμμών

Δ → ***A***



Βέλτιστοι Δρόμοι Κόστους 11:

$\{A, C, E, H, J\}, \{A, D, E, H, J\}, \{A, D, F, I, J\}$

Αλγόριθμοι Δυναμικού Προγραμματισμού ***Bellman-Ford*** στηρίζουν την δρομολόγηση ***Border Gateway Protocols*** (***BGP***) ανάμεσα στα ~62,000 Αυτόνομα Συστήματα (***Autonomous Systems, AS***) στο ***Internet*** (~750,000 γνωστά δίκτυα)

ΣΤΟΧΑΣΤΙΚΕΣ ΔΙΑΔΙΚΑΣΙΕΣ & ΒΕΛΤΙΣΤΟΠΟΙΗΣΗ

Δυναμικός Προγραμματισμός με Προσέγγιση Monte Carlo

ΦΑΣΗ ΕΝΙΣΧΥΤΙΚΗΣ ΜΑΘΗΣΗΣ

- Με βάση το μοντέλο Δυναμικού Προγραμματισμού του συστήματος (***Markov Decision Process***) αξιολογούνται αποφάσεις του ***agent*** $a = \mu(i) \in \mathcal{A}_i$ για όλες τις καταστάσεις του ***περιβάλλοντος*** $i \in \mathcal{X}, i = 1,2, \dots, N$ οι οποίες επηρεάζουν τη εξέλιξη του συστήματος $(i, a) \to j$ με πιθανότητες $p_{ij}(a)$ και τα αναμενόμενα κόστη $c(i, a) = \sum_{j=1}^{N} p_{ij}(a) g(i, a, j)$
- Οι εναλλακτικές πολιτικές $\pi = \{\mu, \mu, \dots\}$ συγκρίνονται ως προς τα αναμενόμενα μακροπρόθεσμα ***costs-to-go*** $J^{\mu}(i)$ μέσω επιλογής απόφασης a στην παρούσα κατάσταση i που βελτιώνει (μειώνει) τους ***συντελεστές κόστους – απόφασης*** $Q^{\mu}(i, a) \to Q^{*}(i, a)$

$$Q^{*}(i, a) = \min_{a \in \mathcal{A}_i} Q^{\mu}(i, a) = \min_{a \in \mathcal{A}_i} \left\{ c(i, a) + \gamma \sum_{j=1}^{N} p_{ij}(a) J^{\mu}(j) \right\}$$

Οι συντελεστές συνυπολογίζουν αναδρομικά: (1) Τα $g(i, a, j)$, ***άμεσα κόστη μετάβασης*** $i \to j$ με επιλογή απόφασης a από τον ***agent*** , (2) το αναμενόμενο κόστος εναλλακτικής άμεσης απόφασης $i \to a$ αν ο ***agent*** συνεχίσει προς τα υπόλοιπα βήματα με μ

- Ο ***agent*** ανανεώνει ***Lookup Table*** για όλες τις καταστάσεις του ***περιβάλλοντος*** $i \in \mathcal{X}$ και της απόφασης του $a = \mu(i) \in \mathcal{A}_i$ μέχρι τη σύγκλιση σε βέλτιστη πολιτική $\pi^{*} = \{\mu, \mu, \dots\}$
- Οι καταχωρήσεις σε ***lookup tables*** έχουν απαιτήσεις σε μνήμη ανάλογες με τον αριθμό καταστάσεων του περιβάλλοντος

ΦΑΣΗ ΕΦΑΡΜΟΓΗΣ ΤΗΣ ΒΕΛΤΙΣΤΗΣ ΠΟΛΙΤΙΚΗΣ

- Ο ***agent*** καθοδηγεί το περιβάλλον επιβάλλοντας ενέργειες για τις καταστάσεις του βάση του ***Lookup Table*** στον οποίο συνέκλινε η **Φάση Ενισχυτικής Μάθησης**

Δυναμικός Προγραμματισμός με Προσέγγιση Monte Carlo

Οι δύο αλγόριθμοι Δυναμικού Προγραμματισμού (***Value Iteration*** & ***Policy Iteration***) προαπαιτούν γνώση των **πιθανοτήτων μεταβάσεων** $p_{ij}(a)$ και του ***άμεσα αναμενόμενου κόστους κατάστασης*** $c(i,a) = \sum_{j=1}^{N} p_{ij}(a) g(i,a,j)$ **εκτιμώμενου** με βάση τα **γνωστά** $g(i,\mu(i),j) = g(i,a,j)$ (***άμεσα κόστη μετάβασης*** $i \to j$ με απόφαση a). Η απόφαση $\mu(i) = a$ του ***agent*** ορίζεται **μονοσήμαντα** για την κατάσταση $i \Rightarrow g(i,a,j) \equiv g(i,j)$

Οι απευθείας προσεγγιστικές μέθοδοι (***Direct Approximate Dynamic Programming Methods***) εκτιμούν τις πιθανότητες μετάβασης και τα αναμενόμενα κόστη μεταβάσεων - αποφάσεων μακροπρόθεσμων πολιτικών με προσομοιώσεις ***Monte Carlo***

Ενσωματώνονται στη **Φάση Ενισχυτικής Μάθησης** των δύο αλγορίθμων Δυναμικού Προγραμματισμού με τις εξής παραλλαγές:

- **Value Iteration →** **Temporal-Difference *TD*(0) Learning**
- **Policy Iteration →** ***Q*-Learning**

Γενική Μεθοδολογία - Απαιτήσεις

- Οι προσομοιώσεις ***Monte Carlo*** δημιουργούν σενάρια πολλαπλών πιθανών τροχιών (***system trajectories***) της εξέλιξης του ***Markov Decision Process***
- Οι τιμές συναρτήσεων ***cost-to-go*** $J(i)$ ανανεώνονται σε κάθε προσομοίωση με προσθήκη του (**γνωστού**) ***άμεσου*** (***observed***) ***κόστους μετάβασης*** $g(i,j)$ σε επισκέψεις προσομοιωμένης τροχιάς μεταβάσεων από κατάσταση i προς κατάσταση j
- Οι μέθοδοι ***Monte Carlo*** απαιτούν γνώση της δομής του περιβάλλοντος, διαχειρήσιμο αριθμό καταστάσεων και σημαντικό αριθμό από ***trajectories*** για καλές εκτιμήσεις

ΣΤΟΧΑΣΤΙΚΕΣ ΔΙΑΔΙΚΑΣΙΕΣ & ΒΕΛΤΙΣΤΟΠΟΙΗΣΗ

Προσεγγιστικός Αλγόριθμος *TD*(0) Learning

Value Iteration → Temporal-Difference *TD*(0) Learning

Εξισώσεις ***Bellman*** υπολογισμού ***costs-to-go*** από i_nστο βήμα $n < N$, τελική κατάσταση $i_N = 0$:

$$J^{\mu}(i_n) = \mathrm{E}[g(i_n, i_{n+1}) + \gamma J^{\mu}(i_{n+1})] = \mathrm{E}\left[\sum_{k=0}^{N-n-1} \gamma^k g(i_{n+k}, i_{n+k+1})\right], n = 0,1, \dots, N-1$$

Με επανειλημμένες προσομοιώσεις ***Monte Carlo*** δημιουργούμε ***trajectories*** του συστήματος σύμφωνα με μια πολιτική (***on-policy***) και μαθαίνουμε τα $J^{\mu}(i_n)$ μέσω ***Robbins-Monroe Successive Approximations*** που διορθώνουν εκτιμήσεις τιμών τους (***updates***) κατά την επίσκεψη της κατάστασης i_n με συντελεστή μάθησης (***learning rate***) η_n:

$$J^{\mu}(i_n) \coloneqq J^{\mu}(i_n) + \eta_n[g(i_n, i_{n+1}) + \gamma J^{\mu}(i_{n+1}) - J^{\mu}(i_n)] = J^{\mu}(i_n) + \eta_n d_n$$

Το σφάλμα $d_n \triangleq g(i_n, i_{n+1}) + \gamma J^{\mu}(i_{n+1}) - J^{\mu}(i_n), n = 0,1, \dots, N-1$ ονομάζεται χρονική διαφορά (***Temporal Difference***, ***TD***) στο βήμα n και οδηγεί τα $J^{\mu}(i_n)$ προς τη ***σύγκλιση***

Εναλλακτικός αλγόριθμος ***update*** προκύπτει από την μακρόχρονη επαναληπτική σχέση:

$$J^{\mu}(i_n) \coloneqq J^{\mu}(i_n) + \eta_n\left(\sum_{k=0}^{N-n-1} \gamma^k g(i_{n+k}, i_{n+k+1}) - J^{\mu}(i_n)\right) = J^{\mu}(i_n) + \eta_n \sum_{k=0}^{N-n-1} \gamma^k d_{n+k}$$

με αρχικές συνθήκες $J^{\mu}(i_n) = 0$, τελικά κόστη $J^{\mu}(i_N) = 0$ και ***learning rate*** $\eta_n = {}^{1}\!/_{n}$

Τα ***costs-to-go*** εκτιμώνται σαν μέσοι όροι σε μεγάλο αριθμό επαναλήψεων προσομοιώσεων με πάρα πολλές επισκέψεις $T \to \infty$ καταστάσεων i_n στο βήμα n κάποιου ***trajectory*** :
$J^{\mu}(i_n) = \mathrm{E}\left[\sum_{k=0}^{N-n-1} \gamma^k g(i_{n+k}, i_{n+k+1})\right] \cong \frac{1}{T}\sum_T c(i_n)$ όπου $c(i_n) \triangleq \sum_{k=0}^{N-n-1} \gamma^k g(i_{n+k}, i_{n+k+1})$

Προσεγγιστικός Αλγόριθμος *Q*-Learning (1/2)

Policy Iteration → *Q*-Learning

- Προσδιορισμός πολιτικής βέλτιστης συμπεριφοράς (***off-policy behavior generation***) μέσω διερεύνησης (***exploration***) όλων των εναλλακτικών αποφάσεων στο παρόν βήμα για εκμετάλλευση (***exploitation***) σεναρίων ***greedy*** αποφάσεων Δυναμικού Προγραμματισμού
- Ορίζουμε $s_n \triangleq (i_n, a_n, j_n, g_n)$ για μεταβάσεις $i_n \rightarrow j_n = i_{n+1}$ στο βήμα n με απόφαση a_n και άμεσο κόστος μετάβασης $g_n = g(i_n, a_n, j_n)$
- Με βάση παρατηρήσεις δειγμάτων s_n και αποφάσεις ***greedy*** ο αλγόριθμος ***Q*-Learning** οδηγεί το σύστημα στη μάθηση βέλτιστης πολιτικής κατά προσέγγιση του ***policy iteration***
- **Προϋπόθεση**: Η i_n πρέπει να είναι ***fully observable***

Σύνοψη Εννοιών Δυναμικού Προγραμματισμού

Βέλτιστα ***Cost-to-Go*** (***Bellman***): $J^*(i) = \min_{a \in \mathcal{A}_i} \left(c(i,a) + \gamma \sum_{j=1}^{N} p_{ij} J^*(j) \right), i = 1,2, \ldots, N$

Ορισμός ***Q-Factors***: $Q(i,a) \triangleq c(i,a) + \gamma \sum_{j=1}^{N} p_{ij}(a) J(j)$

Ορισμός ***Άμεσου Αναμενόμενου Κόστους***: $c(i,a) \triangleq \sum_{j=1}^{N} p_{ij} g(i,a,j)$

Ορισμός ***Βέλτιστων Q-Factors***: $Q^*(i,a) = \sum_{j=1}^{N} p_{ij}(a) \left(g(i,a,j) + \gamma \min_{b \in \mathcal{A}_j} Q^*(j,b) \right)$

Σημείωση: Ορισμοί ***on-policy***, ***off-policy***

- Η ***on-policy*** εκτιμά το συνολικό κόστος σε κάθε βήμα συνυπολογίζοντας την απόφαση του παρόντος βήματος της υπό αξιολόγηση πολιτικής (π.χ. ***TD*(0)-Learning**)
- Η ***off-policy*** συγκρίνει εναλλακτικές αποφάσεις στο παρόν βήμα με δεδομένες τις μελλοντικές αποφάσεις της υπό αξιολόγηση πολιτικής και επιλέγει με απληστία την απόφαση που μειώνει το αναμενόμενο κόστος στην παρούσα κατάσταση (π.χ. ***Q*-Learning**)

ΣΤΟΧΑΣΤΙΚΕΣ ΔΙΑΔΙΚΑΣΙΕΣ & ΒΕΛΤΙΣΤΟΠΟΙΗΣΗ

Προσεγγιστικός Αλγόριθμος *Q*-Learning (2/2)

Policy Iteration → *Q*-Learning

Αλγόριθμος Υπολογισμού $Q^*(i,a)$ με Successive Approximations (*Robins-Monro*)

$$Q(i,a) \coloneqq (1-\eta)Q(i,a) + \eta \sum_{j=1}^{N} p_{ij}(a)\left[g(i,a,j) + \gamma \min_{b\in\mathcal{A}_j} Q(j,b)\right] \text{ για } \forall (i,a)$$

Από τα $Q^*(i,a)$ προσδιορίζεται ο πίνακας βέλτιστης πολιτικής π με αντιστοίχηση

$$\mu^*(i) = \arg\min_{a\in\mathcal{A}_i} Q^*(i,a) \text{ για } i = 1,2,\dots,N$$

Στοχαστική Παραλλαγή

Αν οι $p_{ij}(a)$ δεν είναι διαθέσιμες ο αλγόριθμος βασίζεται σε ***Monte Carlo trajectories***:

Στην επανάληψη n με $J_n(j) = \min_{b\in\mathcal{A}_j} Q_n(j,b)$:

- $Q_{n+1}(i,a) \coloneqq (1-\eta_n)Q_n(i,a) + \eta_n[g(i,a,j) + \gamma J_n(j)]$ για $(i,a) = (i_n, a_n)$
- $Q_{n+1}(i,a) \coloneqq Q_n(i,a)$ για $\forall (i,a) \neq (i_n, a_n)$

Στο όριο $Q^*(i,a) = \lim_{n\to\infty} Q_n(i,a)$

Η ***learning parameter*** η_n είναι φθίνουσα ως προς n, π.χ. $\eta_n = \alpha/(\beta+n)$ με α,β θετικά