



קריית החינוך
פארק המדע
בית לערכים
למצוינות ולחדשנות

Model Free Monte Carlo Method



Model free

- Value Iteration מחייב אותנו לדעת מהו המודל של הסביבה. כלומר, מה יהיה המצב הבא בעקבות כל פעולה שהסוכן ינקוט בה.
- השימוש במשוואת בלמן אפשרי רק אם אנו יודעים מהו המצב הבא בעקבות פעולת הסוכן.

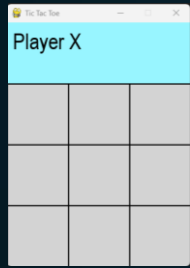
$$V_p(s) = r + \gamma V_p(s')$$

- במקרים רבים הסוכן אינו מכיר את המודל, ואינו יודע מהו המצב העוקב לפעולה בה נקט. אנחנו לא יכולים לבנות את שרשרת הפעולות:

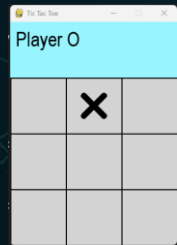
$$S_0, A_0, R_0, S_1, A_1, R_1, S_2, A_2, R_2, \dots$$

הדגמת הבעייתיות במשחק איקס עיגול

- נדגים את הבעייתיות בהעדר מודל ידוע באמצעות משחק איקס עיגול:
- השחק שלנו הוא איקס והמצב הנוכחי s - הוא מצב התחלתי.
- נניח שהשחקן בוחר פעולה $(0,1)$.
- הלוח שנוצר אינו מהווה את המצב הבא. אלא הוא ממתין לסביבה (השחקן השני) כדי שיבצע מהלך.
- רק לאחר שהשחקן היריב מבצע את המהלך אנחנו יודעים מהו s' , ויכולים לחשב את התגמול reward, ולבחור את הפעולה הבאה.



s - מצב התחלתי



s, a - מצב פעולה



s' - המצב הבא

$$S_0, A_0, R_0, S_1, A_1, R_1, S_2, A_2, R_2, \dots, S_T, A_T, R_T$$

Value Iteration – Unknown Model

- מכאן, בהעדר מודל ידוע לסוכן איננו יכולים להשתמש באלגוריתם של Value Iteration, כיוון שאין לנו כל דרך לחשב את נוסחת בלמן:

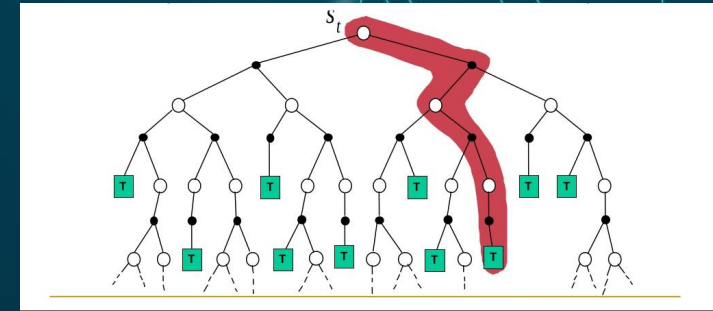
$$V_p(s) = r + \gamma V_p(s')$$

יתר על כן, גם אם נתונה לנו פונקציית הערך $V(s)$, אנחנו לא יכולים למצוא את המדיניות (policy) כפי שעשינו ב Value Iteration. אנחנו לא יודעים לבחור את הפעולה שתביא אותנו ל $V(s')$ המירבי.

0	0	0.729	0.81
0	0.729	0	0.9
0.729	0.81	0.9	1
0.81	0.9	1	0

הפתרון – Monte Carlo Method

- הפתרון לבעיית המודל החסר הינה לבצע דגימות של הסביבה (לשחק מול השחקן היריב) ולרשום את שרשרת הפעולות עד לסיום משחק אחד:



$$S_0, A_0, R_0, S_1, A_1, R_1, S_2, A_2, R_2, S_3, A_3, R_3 \dots$$

- כעת יש לנו את כל הנתונים כדי לחשב את הערך של המצב:
- $V(s_0) = G_0 = R_0 + \gamma R_1 + \gamma^2 R_2 + \gamma^3 R_3 + \dots$
- ניתן לבצע את החישוב לגבי כל אחד מהמצבים בשרשרת המצבים אותה דגמנו.
- פעולה זו מבוצעת מספר רב של פעמים כדי שנעבור על כל המצבים האפשריים.
- אנחנו מבצעים מספר רב של ניסויים (דגימות) על מנת לקבל שרשראות רבות.

בעיות שיש לפתור באלגוריתם MC

- מודל אקראי – כיצד נתמודד עם האפשרות שלאחר שביצענו דגימות של הסביבה קיבלנו נתונים סותרים. למשל, בדגימה של אותו מצב-פעולה שהסוכן ביצע הסביבה החזירה מצבים שונים?
 - לדוגמה במשחק מול יריב לאחר צעד שלנו היריב בחר צעדים שונים בכל משחק.
- מודל לא ידוע (Model Free) – כיצד נתמודד עם הבעיה שהמודל שלנו לא ידוע, ולכן אנחנו לא יכולים לדעת בעת ביצוע פעולה מסויימת מה המצב הבא שהסביבה תחזיר לנו, ולא יכולים לבחור את הפעולה שתביא לנו את הערך הטוב ביותר.
- דרך הדגימה – כיצד הסוכן יבחר את הפעולות שהוא מבצע במהלך הדגימה של הסביבה? האם בצורה אקראית? האם לפי מדיניות מסוימת?

בעיה 1 - סביבה סטוכסטית (אקראית)

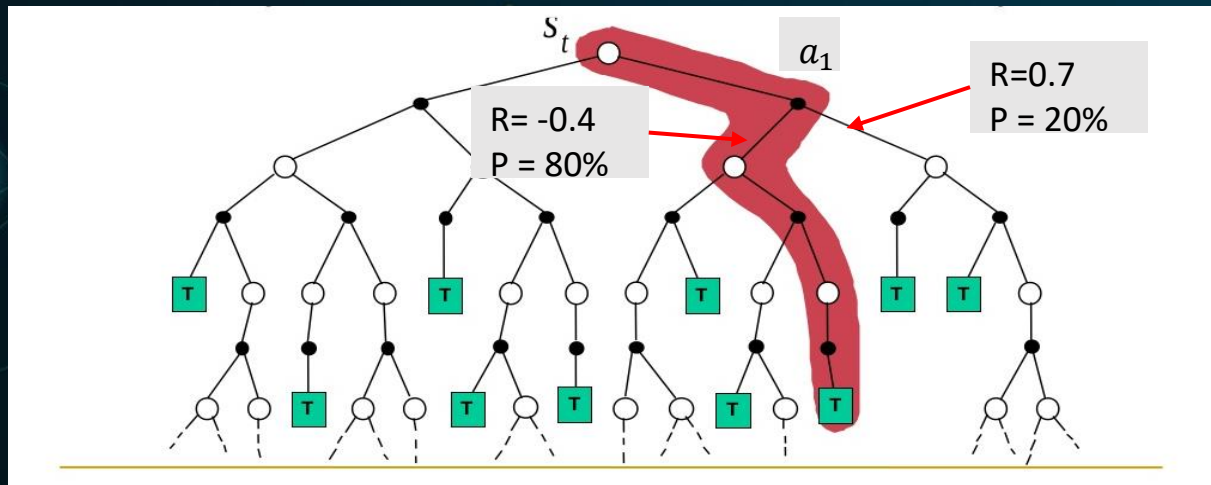
- עד כה התעלמנו מהמקרים בהם המודל של הסביבה הוא אקראי (סטוכסטי) בעת חישוב ערכי המצב.
- במודל אקראי לאחר שהסוכן מבצע פעולה מסויימת המצב הבא לא קבוע מראש, ויתכן שהסביבה תחזיר מספר מצבים, לדוגמה:
 - בעת כניסה לעיקול עם מכונית כאשר מצב הכביש לא ידוע: אם הכביש יבש - המכונית תצליח לסיים את הסיבוב. אם הכביש רטוב או עם שמן - המכונית תחליק. הכביש חלק ב- 10% מהזמן.
 - בעת משחק איקס עיגול לאחר שאנחנו מבצעים מהלך מסויים לא ידוע לנו מה היריב יבצע. יתכן שיבחר באותו מצב בפעולות אחרות - למשל סוכן אקראי - רנדומלי.
 - במשחק קלפים - נניח שבחרנו לקבל קלף נוסף - איננו יכולים לדעת מהו הקלף שנקבל - כלומר לאיזה מצב נגיע.
 - במשחקי קוביות - משחקי מזל המצב הוא דומה.

נוסחת בלמן בסביבה אקראית

• נוסחת בלמן המלאה מתחשבת בהסתברות של המודל:

$$V(s) = \mathbb{E}[R_{s'} + \gamma V(s')] \text{ for all } s'$$

• ערך המצב הוא התוחלת (הממוצע המשוקלל) של כל המצבים הבאים האפשריים.



• נדגים זאת, באמצעות גרף מצבים הבא:

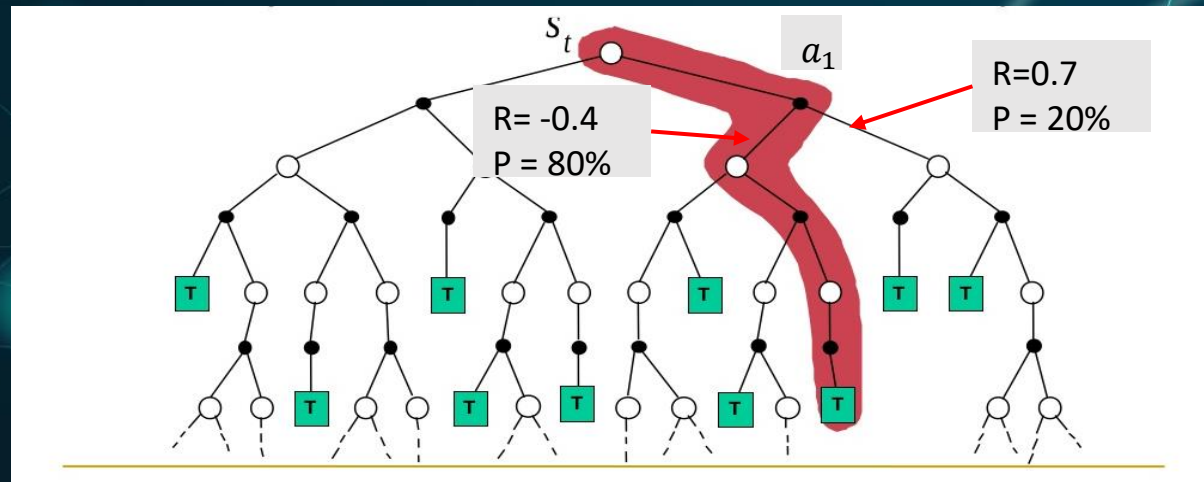
• עיגול לבן – מצב.

• עיגול שחור – הפעולה שנבחרה.

$$Q(s_t, a_1) = -0.4 * 0.8 + 0.7 * 0.2 = -0.18$$

דגימה של סביבה אקראית

- ביצוע מספר רב של דגימות חוזרות ונשנות יגלה לנו את ההתפלגות של המצבים (למשל כיוון שהיריב בחר כל פעם בפעולה אחרת).
- הפתרון לפי נוסחת בלמן המלאה הוא לבצע חישוב ממוצע של התוצאות שקיבלנו וכך נקבל את הערך האמיתי (המשוקלל) של המצב.



חישוב ממוצע של הערכים

- חישוב הממוצע יכול להיעשות באמצעות החזקת מונה של מספר הפעמים שביקר הסוכן במצב.
- במקרה זה החישוב פשוט:

$$V = \frac{V * (n - 1) + V_{new}}{n} = \frac{V * n + (V_{new} - V)}{n} = V + \frac{1}{n} (V_{new} - V)$$

- למשל, אם אנחנו נמצאים במצב s ומבצעים פעולה a 100 פעמים וקיבלנו את מצב s' ב 80 מקרים ואילו מצב s'' ב 20 מקרים – גילינו את ההתפלגות של המודל ונוכל לחשב את ערך $V(s)$.

חישוב ממוצע ללא החזקת מונה

- דרך נוספת לחשב את הממוצע ללא החזקת מונה היא באמצעות שיטת תיקון הטעות.

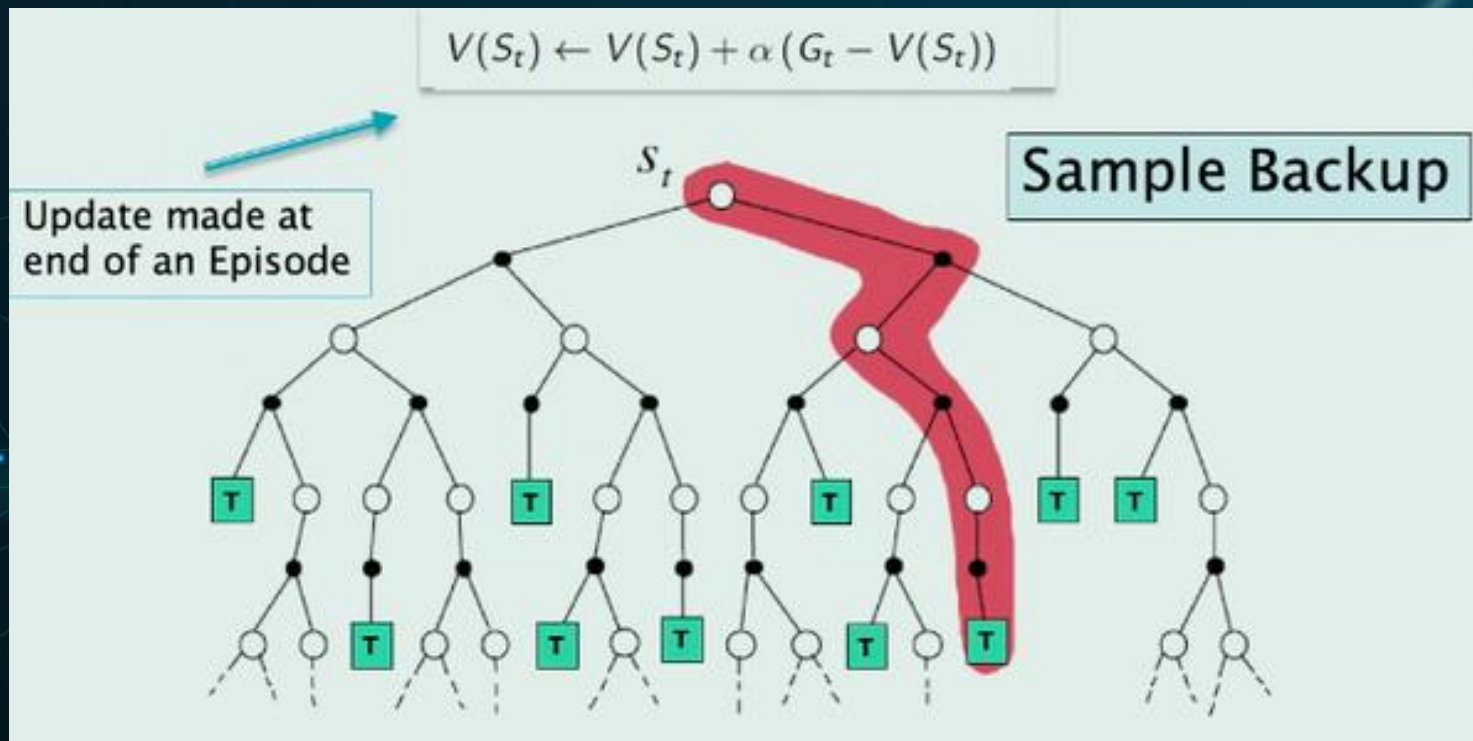
$$V = V + \frac{1}{n}(V_{new} - V)$$

$$V = V + \alpha(V_{new} - V)$$

- α – הוא מספר קטן (מקדם הלמידה) במקום $\frac{1}{n}$.
- בכל פעם אנחנו מגדילים או מקטינים את הערך הממוצע בהתאם לכיוון הטעות שלנו. אם הערך החדש גדול מהממוצע נוסיף ל Value עוד ערך קטן. אם הערך קטן מהממוצע נפחית.
- אנחנו יוצאים מתוך הנחה כי נבצע את החישוב מספר רב מאוד של פעמים ולכן נגיע לממוצע האמיתי.
- השיטה מזכירה לנו את שיטת ה- Gradients Descent רק בלי הנגזרת.

Monte Carlo Method

- לסיכום - ב MC אנחנו נגיע למצב S_T מספר רב של פעמים. בכל פעם נחשב את התגמול העתידי G_t לפי המסלולים השונים אליהם נגיע, ונבצע ממוצע.
- החישוב הסופי יהיה קרוב מאוד לממוצע ויתן לנו את הערך של $V(S_t)$.



בעיה 2 – מציאת המדיניות בהעדר מודל

- מצאנו דרך לחשב את פונקציית הערך Value גם בהעדר מודל ידוע על ידי סדרה של ניסויים ושמירת תוצאות הניסוי.
- נותרה לנו הבעיה שאיננו יכולים לקבל את המדיניות מפונקציית הערך. בעבר בחרנו את הפעולה שתביא לנו את המצב עם הערך הגבוה ביותר. אבל אם איננו יודעים מהו המצב הבא – כיצד נבחר את הפעולה הכדאית ביותר ?

```
State
[' ', ' ', ' ']
[' ', ' ', ' ']
[' ', ' ', ' ']
```

State	Value
[' ', 'X', ' ']	
[' ', ' ', '0']	
[' ', ' ', ' ']	0.125
[' ', 'X', ' ']	
['0', ' ', ' ']	
[' ', ' ', ' ']	0.752
['X', ' ', ' ']	
[' ', ' ', ' ']	
[' ', '0', ' ']	0.225
[' ', ' ', 'X']	
[' ', ' ', ' ']	
['0', ' ', ' ']	0.691

בעיה 3 – דרך ביצוע הבדיקה

- בחירת הפעולות שנבצע במהלך ביצוע הדגימה של הסביבה יכולה להיעשות במספר אפשרויות:
- **בחירה אקראית** – הסוכן יבחר בכל פעם פעולה חוקית אקראית:
 - שיטה זו תביא לתוצאה נכונה אך תחייב מספר רב מאוד של דגימות והלמידה תהיה מאוד איטית.
- **בחירה בהתאם לטבלת Q** – נבחר בכל פעם את הפעולה הטובה ביותר לפי הידע שיש לנו בעת ביצוע הפעולה (טבלת הערכים).
 - בהתחלה הטבלה אקראית אך תוך כדי ניסוי הערכים מתקרבים לערכים האופטימליים.
 - הבעיה – ברגע שמצאנו פעולה אחת טובה אנחנו עלולים להפסיד ולא לנסות פעולות אחרות שאולי הן טובות יותר, וכיוון שלא בחרנו אותן לא נדע שהן טובות יותר.
- זוהי הדילמה המכונה בספרות **Exploitation vs. Exploration**. מתי אנחנו צריכים לנסות לחקור מצבים חדשים (Explore) ומתי לנצל את הידע הקיים וללכת לפיו (Exploitation).

Epsilon-Greedy Function

- אחד הפתרונות לדילמת Exploration vs Exploitation הוא לבחור בדרך כלל את הפעולה שלפי הידע העכשווי שלנו היא הטובה ביותר (לפי טבלת Q_Values), אך מידי פעם לבחור פעולה אקראית.
- האלגוריתם נקרא epsilon-greedy בו הסיכוי לבחור פעולה אקראית נקבע על ידי המספר epsilon:

```
epsilon = 0.001

def Epsilon_Greedy (state, Q_Value):
    r = Random()
    if r < epsilon:
        return random legal action
    else:
        return maxArg_a (Q(s, a))
```

Epsilon-greedy decay

- ניתן לשפר את האלגוריתם בכך שבתחילת האימון הסוכן יבצע הרבה Exploration, ואילו ככל שהמודל שלנו מאומן יותר נסתמך יותר על טבלה Q ונבצע Exploitation.
- השיפור יעשה על ידי כך שהערך של Epsilon ירד במהלך זמן האימון. בהתחלה יהיה גבוה וככל שהמודל יהיה יותר מאומן הערך ירד:

```
epsilon_start = 1.0
epsilon_final = 0.01
epsiln_decay = 100000

def epsilon_greedy(epoch, start = epsilon_start, final=epsilon_final, decay=epsiln_decay):
    res = final + (start - final) * math.exp(-1 * epoch/decay)
    return res
```

סיכום - עקרונות אלגוריתם מונטה קרלו

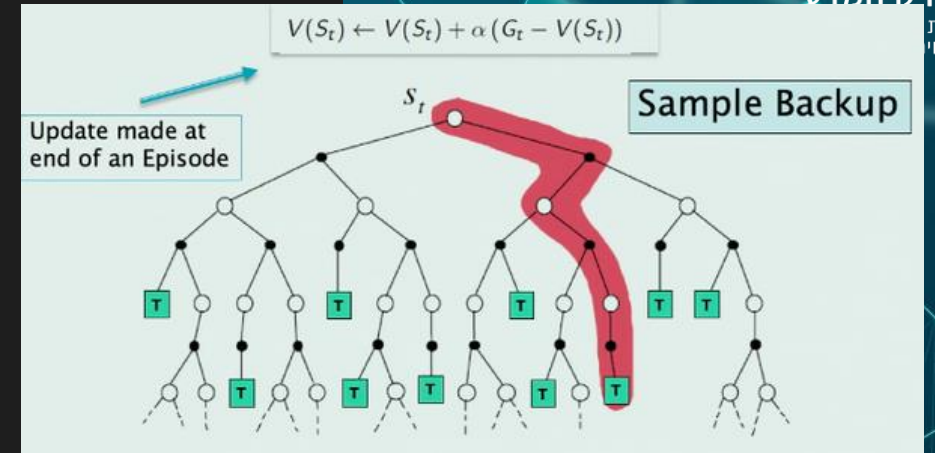
- **ביצוע דגימות של הסביבה** – כיוון שאיננו יודעים את המודל אנחנו מבצעים דגימה של הסביבה עד למצב סופי (אפיזודה), וכך מקבלים את התגמולים לחישוב הערכים הנדרשים.
- **חישוב ממוצע הערכים** – כיוון שהסביבה היא אקראית (כל פעולה יכולה להוביל למצבים שונים בזמנים שונים) אנחנו נחשב את ממוצע הערכים של המצב והפעולה שהתקבלו בדגימות השונות.
- **שימוש בטבלת מצב פעולה Q** – כיוון שהמודל לא ידוע לנו אנחנו נשמור את הערכים בטבלת Q של מצב-פעולה, וכך בכל מצב נוכל לבחור את הפעולה עם הערך הגבוה ביותר.
- **Exploitation vs. Exploration** – בבחירת הפעולות שנבצע במהלך הדגימה נשתמש בפונקציית ϵ -greedy, כך שברוב המקרים נבחר את הפעולה המיטבית לפי טבלת Q עימה אנחנו עובדים, אף לעיתים נבחר פעולה אקראית.

פסאודו-קוד מונטה קרלו



קריית החינוך
פארק המדע
בית
למצוי

```
1 def Monte_Carlo ():
2
3     #initialize
4     alpha = small number
5     epsilon = small number or epsilon_decay()
6     for all s in S, a in A
7         Q(s,a) = random()
8     epochs = big number
9     for epoch in epochs (for each episode):
10        epsilon = epsilon_decay()
11        Generate an episode using epsilon-greedy(): S0,A0,R0,S1,A1,R1 ...S_n
12        G = 0
13        for t=n-1 to 0:
14            G = R_t + gamma * G
15            Q(S_t, A_t) = Q(S_t, A_t) + alpha * (G - Q(S_t, A_t))    #Average
16
17 def epsilon_greedy ():
18     r = Random()
19     if r < epsilon:
20         return random legal action
21     else:
22         return maxArg_a (Q(s, a))
```



דוגמה – מונטה קרלו במשחק איקס עיגול

- נדגים כיצד ליישם את אלגוריתם מונטה קרלו במשחק איקס עיגול.
- כיוון במשחק איקס עיגול יש לכל היותר $3^9 = 19,683$ מצבים (ויש אפילו פחות), ניתן ליישם את המשחק באמצעות תכנון דינמי.
- נאמן את המודל שלנו מול סוכן רנדומלי. הסוכן שלנו ישחק את השחקן הראשון עם איקס.

• תרגיל

- עליכם לשנות את הקוד ולבנות סוכן שישחק את שחקן 2 – המשחק עם העיגול.

מימוש Q-Table

- אחת הדרכים לממש Q-Table היא באמצעות מילון (Dictionary):
 - המפתח (key) יהיה (state, action)
 - הערך (Value) יהיה הערך המספרי של המצב-פעולה.
- במשחקי לוח המפתח של (state, action) יכול להיות state_q וכך נפשט את המימוש. נסביר באמצעות דוגמה של איקס עיגול:

• נניח מצב התחלתי s הוא לוח ריק

(s,a) = (

, (1,1) , מצב-פעולה שלנו הוא אם כן: (1,1), נבחר פעולה (1,1) = action. מצב-פעולה שלנו הוא אם כן: (1,1),

state_q =

	x	

: ניתן להציג את המצב-פעולה כלוח שביצע את הפעולה:

	x	
0		

• המצב הבא s' כבר תלוי במהלך של השחקן השני והוא יהיה לדוגמה: