מערכות מרובות סוכנים

סיכום שיעור מספר 6

אלעד מרגלית

בשיעורים הקודמים צפינו במודל המורה-תלמיד, בדוגמא הנשלטת חלקית על ידי מערכות מרובות סוכנים. המורה הוא הסוכן הנשלט היודע את המודל, השחקנים והמצבים כאשר התלמיד סוכן לומד שאינו נשלט. בעבר ראינו כי שאפנו להגיע לOptimal Teaching Policy כאשר יש לנו שליטה על הסוכנים (הן המורה והן על התלמיד).

הגדרה:
אנו משתמשים בתהליך החלטה מרקובי להלן MDP (Marcov Decision Process).
לצורך כך P היא ההסתברות שתוגדר כפונקציה בהתאם למצב בו אנו נמצאים, למצב שאנו עוברים ולפעולה שננקטה לתוך הקטע [0,1].
γ מוגדרת כפונקצית תועלת ו Π היא האסטרטגיה שבה אנו משתמשים.

לכן:

$$P : S \times S \times A \mapsto [0,1] \qquad \gamma : S \mapsto R \qquad \Pi : S \mapsto A$$

במקרה שאנו יודעים בדיוק את אלגוריתם הסטודנט

$$Val(\Pi) = \sum_{k=0}^{\infty} \gamma^k \mathrm{E}_k(u)$$

כאשר ידוע כי

$$\mathrm{E}_k(u) = \sum_{u \in A_s} P_{\Pi,k}(a) \cdot u(a)$$

מכאן

$$Val(\Pi) = \sum_{k=0}^{\infty} \gamma_0{}^k \sum_{\rho \in S} P_{S,\Pi,k}(S') \cdot \gamma(a')$$

$P_{S,\Pi,k}(S')$ מוגדרת כהסתברות במצב S תחת מדיניות π ב K שלבים למצב 'S.

הנחות המודל:

המורה יודע הכל – יש מצב שבו סטודנט (פונקציה שנלקח מאוסף מצבים $\sum$)
פעולת התלמיד במצב S נתונה על ידי פ' כלשהי ρ
ופונקצית מעברים $\tau : \sum \times A_s \times A_t \mapsto \sum$
ההסתברות לבחור a במצב s היא ρ_{sa}
המורה רואה את המצב, הפעולות שנבחרו ולכן גם את המצב החדש אליו נגיע.
כל שנותר למורה לקבוע היא פעולה בהינתן מצב ותלמיד

נגדיר MDP מהצורה TDMP

$$\rho(S,S',a_t) \equiv \sum_{a_s \in A_s} \rho(s,a_s) \cdot \delta_{s',\tau(s,a_s,a_t)}$$

$$U(S) \equiv \sum_{a_s \in A_s} \rho(s,a_s) \cdot U(a_s)$$

ניתן לבדוק שהרדוקציה הנ"ל בין השאר מגדירה לנו שהאסטרטגיה האופטימאלית ב-TMDP היא Optimal Teaching Policy.

על מנת שהאסטרטגיה אופטימאלית יש למקסם את הביטוי:

$$\sum_{k=0}^{\infty} \gamma_0^{\,k} \left(\sum_{s' \in \sum} P_{S,\Pi,k}(S') \cdot u(S') \right)$$

לאחר הצבה מקבלים

$$\sum_{k=0}^{\infty} \gamma_0^{\,k} \left[\sum_{s' \in \sum} P_{S,\Pi,k}(S') \cdot \left(\sum_{a_j \in A_j} \rho(s',a_s) \cdot U(a_j) \right) \right]$$

נשנה את סדר הסכימה ונקבל

$$\sum_{k=0}^{\infty} \gamma_0^{\,k} \sum_{a_j \in A_j} \sum_{s' \in \sum} \rho(s',a_s) \cdot P_{S,\Pi,k}(S') \cdot U(a_s)$$

ניתן לראות שגם כאן מתקיים

$$Val(\Pi) = \sum_{s' \in \sum} \rho(s', a_s) \cdot P_{S,\Pi,k}(S')$$

Q-Learning

המקרה הכללי נקרא Q-Learning. המקרה הספציפי בו מתקיים $\gamma = 0$ נקרא Classical Reinforcement.

בחלק זה אנו נבין אלגוריתם בסיסי עבור הסטודנט הנקרא
כאן הלומד (סטודנט) שומר לכל מצב זוגות של S ו A
נמצאים במצב S, מבצעים פעולה a מקבלים תשלום R ומגיעים למצב 'S.
נניח ערכים התחלתיים כלשהם שומרים לכל מצב s ופעולה a ערך q(s,a).
הלומד מעדכן את Q בכל שלב.
הלומד בוחר פעולה מסט הפעולות מהתפלגות ומקבל תשלום בהתאם למצב שעבר וזז למצב החדש 'S

כלל העדכון הוא:

$$q(s,a) = (1-\alpha)q(s,a) + \alpha(R + \gamma v(s'))$$

כאשר

$$v(s') = \max_{a_s \in A_s} q(s,a)$$

ו R הוא התשלום שהתקבל במצב הנוכחי.

כאן α נקרא קצב הלימוד, γ נקרא ערך ההנחה אנו מניחים ש $0 < \alpha < 1$ ו $0 \le \delta < 1$
V(s') הוא השערה הנוכחית של המדיניות הטובה ביותר ב-'S
לכן, כלל העדכון של הסטודנט הוא שילוב בין מה שהוא חווה עד כה והפעולה הטובה ביותר שהתלמיד יכול לבחור במצב החדש שהגיע אליו

ערכי ה Q צריכים לעזור להגדיר את ההתנהגות של הסטודנט. אנו מעריכים שהסטודנט בוחר בפעולותיו לפי התפלגות בולצמן. זהו הסיכוי שפעולה a שייכת למצב s.

$$P_s(a) = \frac{e^{\left(\frac{q(s,a)}{T}\right)}}{\sum_{a' \in A} e^{\left(\frac{q(s,a')}{T}\right)}}$$

T נקרא טמפרטורה, בדרך כלל T מוגדר כמספר גדול על מנת שבחירת הפעולה תהיה רנדומאלית ככל שניתן ועם הזמן הוא קטן לאט.

Q-Learning תחת הנחות חלשות מתאפיין בתכונה הבאה : (Watkins & Dayan, 1992)

<u>משפט</u>
נחשב MDP שבו תשלומי המצבים לא ידועים, בנוסף פונקציות המעבר גם אינן ידועות אזי Q-Learning Algorithm יהיה אלגוריתם לימוד אופטימאלי.

Reinforcement Learning

כאן האינטרס שלנו הוא ללמוד בסביבה בלתי ידוע לפי משוב מהצופים.
ראינו בחלק הקודם אלגוריתם קלאסי – Q-Learning. כעת נביא דוגמא לבעיית Reinforcement Learning.

דוגמא
סוכן שלומד לנווט בסביבה הוא MDP (יש מחליט החלטות יחיד) כאשר המצבים הם הכיוונים שאליהם הוא רוצה לנווט. הפעולות הן תזוזות (יחד עם הכיוונים) והסוכן מקבל תשלום לפי מצבים טובים ורעים (למשל מצבים יותר קרובים לנקודת היעד יזכו בניקוד גבוה יותר)
כאשר מתרכזים ב Reinforcement Learning ישנם מספר נושאים כגון הסביבה, קריטריון בחירה, למידה, Exploration vs. Exploitation יעילות וכו'.

בחלק זה אנו נניח שהסביבה היא משחק סטוכסטי (Stochastic Game) עם שני שחקנים. הסוכן יהיה היועץ ב-SG השחקנים משחקים ברצף (יכול להיות גם אינסופי) של משחקים מכמה משחקים אפשריים.
בכל שלב במשחק השחקנים בוחרים פעולות, התשלום ניתן לכל שחקן ועוברים לשחק משחק חדש.
המשחק החדש נקבע על ידי ההתפלגויות המשותפות בהתאם למשחקים.
SG הם מקרה כללי יותר של MDP ומשחקים חוזרים.

אנו מקיימים את ההנחות הבאות על המודל:
הסוכן יודע את הזהות של המשחק בכל שלב אך לא את התשלומים, אחרי כל שלב במשחק הסוכן מסתכל על מה שהרוויח והפעולות ששיחק וגם על פעולות היריב. בנוסף אנו מניחים שהסוכן יודע את מספר המקסימאלי של התשלום בתחילת המשחק.
מדיניות של סוכן ב-SG קובעת את ההתפלגות של סוג הפעולות לכל היסטוריה אפשרית, כלומר בכל שלב המדיניות אומרת לסוכן איך לבחור את הפעולה כפונקציה של ההיסטוריה שנצפתה.
המשימה שלנו היא לבנות אלגוריתם שיוכל להשיג תשלום ממוצע צפוי מהר ככל שניתן ושיהיה כמעט מובטח אופטימאלית.

אלגוריתם R-MAX

האלגוריתם המקיים את הדרישות הנ"ל ובזמן פולינומיאלי נקרא R-MAX.
כפי שהוזכר קודם לכן, המטרה היא למצוא תשלום ממוצע קרוב לאופטימאלי ומהר ככל שניתן.
נניח שM הוא משחק סטוכסטי נגדיר את הערך של מדיניות הסוכן (Π) תוך כדי שימוש במוצע הצפוי.
נניח ש ρ תהיה מדיניות היריב.
נסמן את $U_M(s,\Pi,\rho,T)$ בתור הממוצע בשלב ה T.
נסמן
$U_M(s,\Pi,T)=\min_\rho U_M(s,\Pi,\rho,T)$ כך שהמדיניות תבטיח ערך ממוצע בT שלבים תחת מצב נתון S.

נסמן $U_M(s,\Pi)=LIM\inf T\to\infty U_M(s,\Pi,T)$. המדיניות Π שתתקבל תוגדר על ידי
$U_M(\Pi)=Min_{s\in S}U_M(s)$

אלגוריתם R-MAX מקבל כפרמטרים את M,SG והתשלום המקסימאלי האפשרי R-MAX, את השגיאה ε, ההסתברות לכישלון δ, ואת T - ה $\varepsilon - return$ הזמן המשולב בין מדיניות אופטימאלית.

<u>לב האלגוריתם:</u>
אתחול:
מצבים: המצבים המקוריים + מצב פיקטיבי
כל מטריצות המשחקים מסומנים כלא ידועים
כל הפעולות המשותפות מובילים למצב הפיקטיבי עם הסתברות 1. התשלום עבור RMAX הוא בכל מקום.

חזור
1. חשב את המדיניות האופטימאלית ב- T שלבים עבור המצב הנוכחי
2. הרץ את המדיניות הנוכחית
3. עדכון את המודל, אחרי הפעולות ששוחקו – שמור את התשלומים במטריצת המשחק, שמור את השינויים, כאשר ישנם מספיק שינויים מהמצב האחרון (היו מספיק ביקורים במצב הקיים) עדכן את השינויים במודל בהתבסס על התוצאות הקודמות וסמן את המצב כידוע.

ניתן לראות ש-RMAX מתנהג כאופטימי במשחקי SG.

מכאן מקבלים את המשפט הבא:

נניח ש M הוא SG עם N מצבים ו K פעולות. בנוסף נניח קבועים $0 \le \delta < 1$ ו. $\varepsilon > 0$
בנוסף נניח שהמדינות עבור M עם ε–return mixing time הוא T כך ש $OPT\left(\prod_M(\varepsilon,T)\right)$.
אזי מתקיים שעבור הסתברות לא קטנה מ $1-\delta$ האלגוריתם RMAX ייתן תוצאה אופטימאלית $OPT\left(\prod_M(\varepsilon,T)\right)-2\varepsilon$ תוך מספר צעדים פולינומיאלי התלוי ב $\frac{1}{\delta}\frac{1}{\varepsilon}$N k T .

ההוכחה מסתמכת על מספר למות, תחילה ניתן מספר הגדרות

<u>הגדרה</u>
נניח ש M ו $\overline{M}$ יהיו SG עבור אותם מצבים ופעולות. אנו אומרים ש $\overline{M}$ הוא $\alpha - approximation$ עבור M אם לכל מצב S מתקיים:
1. אם $P_M(s,t,a,a')$ ו $P_{\overline{M}}(s,t,a,a')$ ההסתברויות מעבר ממצב s ל t, פעולות הסוכן והיריב הן a,a' בהתאמה ב M ו $\overline{M}$' אזי
$$P_{\overline{M}}(s,t,a,a')-\alpha \le P_{\overline{M}}(s,t,a,a') \le P_{\overline{M}}(s,t,a,a')+\alpha$$
2. עבור כל מצב s אותו מצב המשויך ב M משויך ב M' ולכן התשלומים יהיו זהים

<u>למת הסימולציה</u>

נניח M ו $\overline{M}$ SG עבור N מצבים כאשר $\overline{M}$ הוא $\frac{\varepsilon}{NTR_{\max}} - approximation$ עבור M, אזי לכל מצב S מדיניות הסוכן Π ומדיניות היריב ρ מתקיים

$$\left|U_M(s,\Pi,\rho,T)-U_{\overline{M}}(s,\Pi,\rho,T)\right| \le \varepsilon$$

הלמה הבאה נקראת implicit explore or exploit lemma. נניח ש M הוא SG ו M_L הוא SG המשתמש ב R-MAX כאשר L הוא סט מצבים לא ידוע. $R_{ML} - MAX$ יהיה המדיניות האופטימאלית עבור המשחק M_L SG .

נניח ש M הוא SG בנוסף נניח L ו M_L כמו שצוין לעיל. נניח ש ρ הוא מדיניות היריב ו S הם מצבים כלשהם. בנוסף קבוע $0 < \alpha < 1$. אזי מתקיים:

$$V_{R-MAX} > OPT_M\left(\prod \varepsilon, T\right) - \alpha$$

כאשר V_{R-MAX} הוא התשלום הממוצע הצפוי אחרי T צעדים עבור מדיניות $R_{ML} - MAX$ או שהאלגוריתם יבחר במצב לא ידוע על M ב-T צעדים בהסתברות של לפחות $\frac{\alpha}{R\max}$.

אלגוריתם RMAX יודע לבצע exploration או exploitation ביעילות. היריב יכול למנוע מהסוכן לעשות exploration או exploitation אך לא את שניהם. לכן הסוכן מבצע exploration או exploitation. מכאן מובטח לסוכן לבצע את אחד משניהם ביעילות.