那麼對於一排有連續三個 X 所有狀態獲勝的機率是 1，因為這樣的情況下我們已經贏了。同樣地，對於一排有連續三個 O 的所有狀態或棋盤下滿了卻沒有任何一名玩家放置出一排同樣三枚棋子的狀態，其獲勝的機率為 0，因為我們已經不可能從中獲勝。我們將所有其他狀態的初始值設置為 0.5，表示我們有 50% 的獲勝機會。

接著我們與對手下許多盤棋。為了選擇下一步，我們檢查下棋（當前棋盤上的每個可下棋的區域）之後可能產生的狀態，並在表中查找各個狀態當前的估計值。大多數時候，我們**貪婪地**（*greedily*）下每一步棋，選擇能導向擁有最高價值狀態的下一步棋，也就是具有獲勝率估計值最高的下一步。然而，我們偶爾會隨機下棋。這些被稱為**探索性**（*exploratory*）下法，因為這樣的下法可以讓我們探索到原先無法經歷的狀態。在遊戲中一系列所考慮的和所做出的下法如圖 1.1 所示。

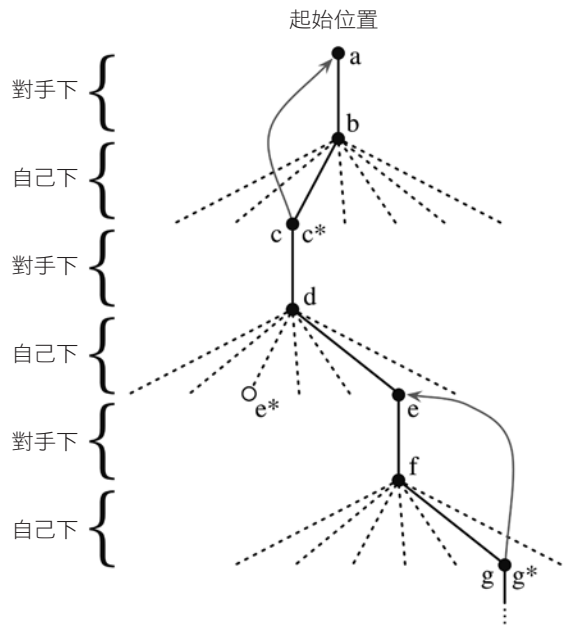


圖 1.1 一系列井字遊戲下法。黑色實線表示遊戲中實際下法，虛線表示我們（強化學習玩家）考慮但未實際採用的下法。我們下的第二步棋是一個探索性下法，這表示另一個兄弟節點 e^* 對應的節點估計值更高。探索下法並不能導致學習，但圖中我們其他的下法則會進行學習，如圖中黑色箭頭所示的更新，其中估計值由樹的子節點流向父節點，細節將在文中進行描述。

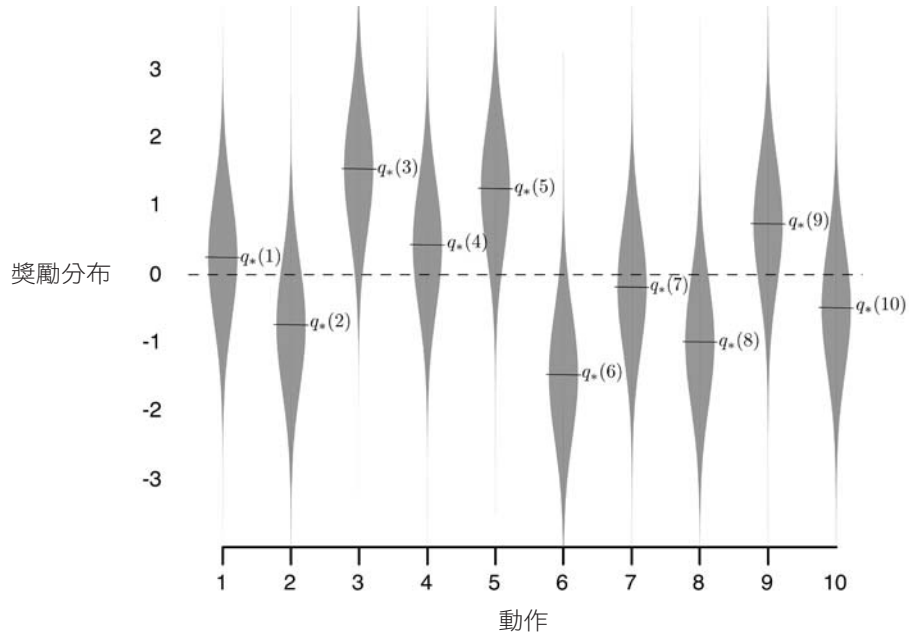


圖 2.1 來自 10- 搖臂測試平台的拉霸機問題範例。十個動作中每一個的真實值 $q_*(a)$ 是根據均值為 0 和變異數為 1 的常態分布採樣所獲得的，而實際獎勵則根據均值為 $q_*(a)$ 和變異數為 1 的常態分布產生，如圖中灰色分布所示。

然後，當一個應用於本問題的學習方法在時步 t 時選擇動作 A_t 時，從具有均值 $q_*(A_t)$ 和變異數為 1 的常態分布中獲得實際獎勵 R_t。這些分布在圖 2.1 中以灰色表示。我們將這組測試任務稱為 **10- 搖臂測試平台**（*10-armed testbed*）。對於任何學習方法，當將其運用在測試平台其中一個拉霸問題時，我們可以測量此學習方法在 1000 個時步中逐步提升的表現和行為。這構成了一個**行程**（*run*）。將學習方法重複 2000 個獨立行程，且每個行程使用不同的拉霸機問題，我們將可以獲得學習演算法平均行為的評估。

如上所述，圖 2.2 為在 10- 搖臂測試平台上對一種貪婪方法與兩種 ε - 貪婪方法（ε = 0.01 和 ε = 0.1）進行比較的情形。所有方法都使用樣本平均法構成它們的動作值估計。上方的圖顯示了期望獎勵會隨經驗而提升。貪婪方法在開始時比其他方法的提升速度略快，但隨後就穩定在一個較低的水平。它獲得了每步僅大約 1 的獎勵，而在此測試平台上可能的最佳值大約為 1.55。貪婪方法在長期的表現明顯較差，因為它經常陷入執行次佳動作的選擇。下方的圖顯示貪婪方法大概在三分之一的任務中發現了最佳動作。

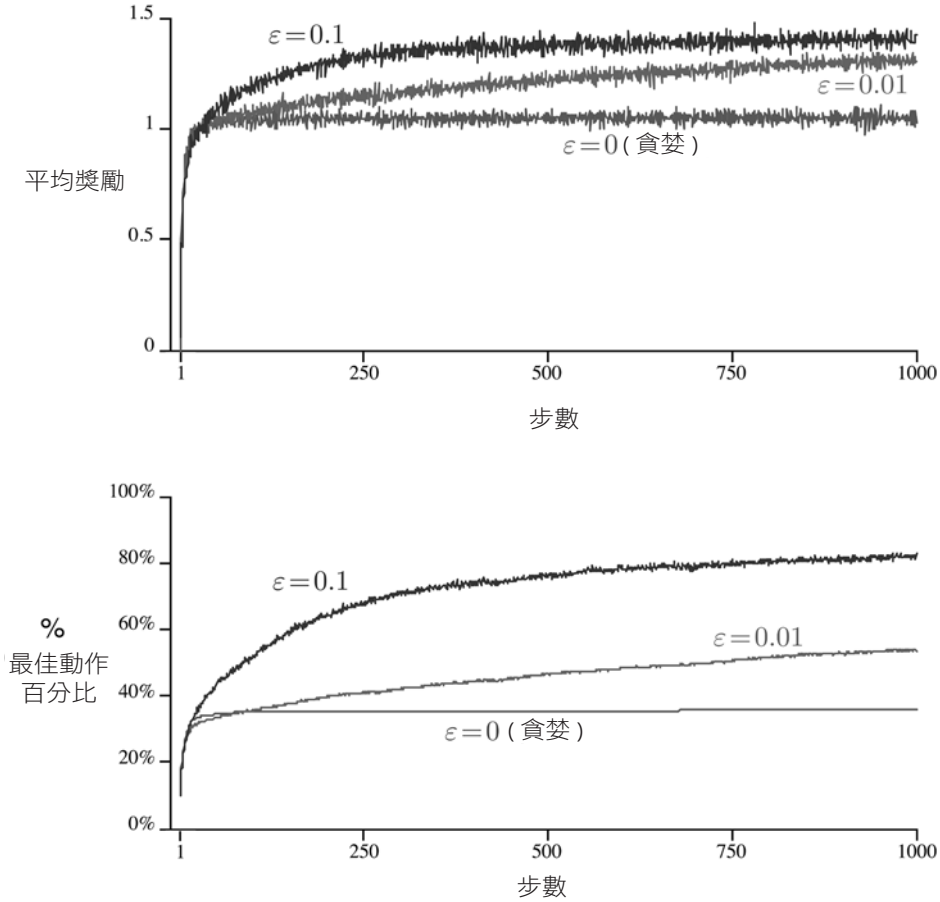


圖 2.2 ε 貪婪動作值方法在 10- 搖臂測試平台的平均表現。這些數據是 2000 次使用不同的拉霸機問題執行行程的平均值。所有方法都使用樣本平均值作為其動作值估計。

在之後三分之二的任務中，其最佳動作的初始值非常差，因此最佳動作並未被再次選擇。ε - 貪婪方法最終表現得比貪婪方法更好，因為它們持續探索並提升他們識別最佳動作的機會。ε = 0.1 的方法探索次數更多，所以經常更早地發現最佳動作，但它選擇最佳動作的機率不會超過 91%。ε = 0.01 方法的提升速度更為緩慢，但最終會在圖中所示的兩種性能指標上優於 ε = 0.1 的方法。我們也可以隨時間減少 ε 以充分利用高 ε 值和低 ε 值各自的優點。

ε - 貪婪方法優於貪婪方法的優勢取決於任務。例如，假設獎勵變異數更大，比如說是 10 而不是 1。對於分布範圍更廣的獎勵，需要更多的探索才能找到最佳

顯而易見的實現方式是維持所有獎勵的記錄，然後在需要估計值時執行上方的計算。然而以這種方式實現的話，隨著時間的推移，對於資訊儲存和計算要求會隨著更多的獎勵而增長。每個新增的獎勵都需要額外的資源進行儲存，並需要額外的計算來計算分子中的總和。

你可能會懷疑，這事實上並不是必需的。透過處理每個新獎勵所需極少量的、常數的計算，很容易設計出用於更新平均值的增量公式。給定 Q_n 和第 n 個獎勵 R_n，所有 n 個獎勵的新平均值可以透過以下的計算得出

$$\begin{aligned} Q_{n+1} &= \frac{1}{n}\sum_{i=1}^{n} R_i \\ &= \frac{1}{n}\left(R_n + \sum_{i=1}^{n-1} R_i\right) \\ &= \frac{1}{n}\left(R_n + (n-1)\frac{1}{n-1}\sum_{i=1}^{n-1} R_i\right) \\ &= \frac{1}{n}\Big(R_n + (n-1)Q_n\Big) \\ &= \frac{1}{n}\Big(R_n + nQ_n - Q_n\Big) \\ &= Q_n + \frac{1}{n}\Big[R_n - Q_n\Big], \end{aligned} \tag{2.3}$$

以上的公式即使在 $n = 1$ 時也成立，在任意 Q_1 的情況下我們將得出 $Q_2 = R_1$。這種實現方式僅需要儲存 Q_n 和 n，對於每個新的獎勵僅需要使用 (2.3) 進行極少量的計算。

此更新公式 (2.3) 是本書中經常出現的一種更新形式。一般表達式為

$$\text{新估計值} \leftarrow \text{舊估計值} + \text{步長}\Big[\text{目標} - \text{舊估計值}\Big] \tag{2.4}$$

表達式 $\Big[$ 目標 – 舊估計值 $\Big]$ 是估計中的誤差（*error*）。透過向「目標」靠近來減少誤差。目標預示著移動的理想方向，儘管它可能是充滿雜訊的。比如在上面的情形中，目標是第 n 個獎勵。

注意在增量式方法 (2.3) 中使用的步長參數會隨時間變化。在處理動作 a 的第 n 個獎勵時，該方法使用的步長參數為 $\frac{1}{n}$。在本書中，我們使用 α 或者更普遍地使用 $\alpha_t(a)$ 來表示步長參數。

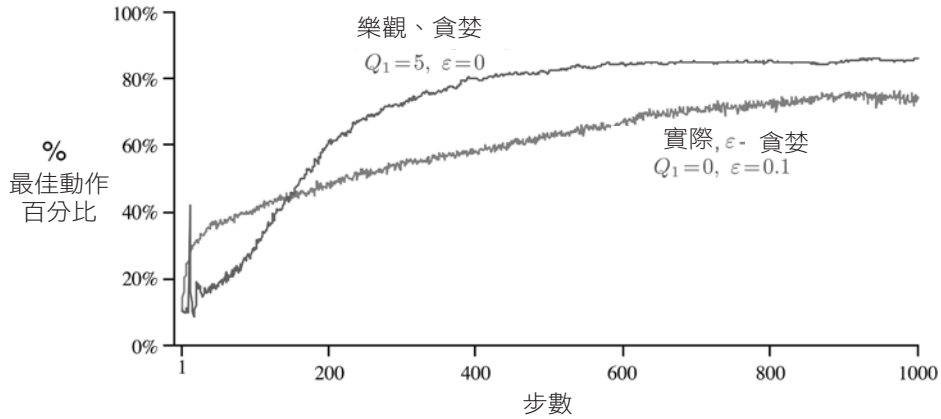


圖 2.3 樂觀的初始動作值估計在 10- 搖臂測試平台上的表現。兩種方法都使用常數步長參數 $\alpha = 0.1$。

如果任務發生變化並對探索產生了新的需求，這種方法就無能為力了。實際上以任何特定方式關注初始條件的方法，都不太可能有助於一般的非固定性情形。開始時刻只會出現一次，因此我們不應過分關注它。這同樣也適用於樣本平均方法，它也將時間的開始視為一種特殊事件，以均等的權重平均所有後續獎勵。話雖如此，這些方法都非常簡單，在實際應用中使用它們其中一個或一些簡單組合通常是足夠的。在本書的其餘部分我們將頻繁使用這些簡單的探索技術。

練習 2.6：神秘的峰值。圖 2.3 所示的結果應該非常可靠，因為它們是由 2000 個獨立的、隨機選擇的 10- 搖臂測試問題上的結果進行平均。那麼為什麼樂觀方法的曲線早期會出現振盪和峰值？換句話說，是什麼原因可能使這種方法在特定的早期步驟中，就平均的意義上而言表現得特別好或特別差？ □

練習 2.7：無偏差的恆定步長技巧。在本章的大部分內容中，我們使用樣本平均值來估計動作值，因為樣本平均值不會產生如恆定步長方式的初始偏差（詳見 (2.6) 中的分析）。然而，樣本平均值並不是一個完全令人滿意的解決方案，因為它們可能在非固定性問題上表現不佳。是否有可能避免恆定步長的偏差，同時保留其對非固定性問題的優勢？一種方法是使用以下步長：

$$\beta_n \doteq \alpha/\bar{o}_n, \tag{2.8}$$

來處理特定動作的第 n 個獎勵，其中 $\alpha > 0$ 是常規恆定步長，$\bar{o}_n$ 是從 0 開始逐步靠近 1 的軌跡：

$$\bar{o}_n \doteq \bar{o}_{n-1} + \alpha(1 - \bar{o}_{n-1}), \text{對於 } n \geq 0, \text{其中 } \bar{o}_0 \doteq 0. \tag{2.9}$$

進行類似於 (2.6) 的分析，以證明 Q_n 是一個*沒有初始偏差*的指數近期加權平均值。

2.7 信賴上界動作選擇

因為動作價值估計的準確性始終存在著不確定性，所以需要進行探索。貪婪的動作是目前看起來最好的行為，但其他一些動作可能事實上更好。ε - 貪婪動作會使非貪婪的動作被選擇，這個過程對於各個動作是不加區分的，並沒有偏好於近乎貪婪或特別不確定的動作。從非貪婪的動作中進行選擇時，最好將其估計值與最大值的接近程度及估計值中的不確定性考慮在內，以評估這些非貪婪的動作實際上成為最佳動作的潛力。這樣做的一個有效方法是根據

$$A_t \doteq \underset{a}{\operatorname{arg\,max}} \left[Q_t(a) + c\sqrt{\frac{\ln t}{N_t(a)}} \right], \tag{2.10}$$

其中 ln t 表示為 t 的自然對數（即 $e \approx 2.71828$ 的多少次方等於 t），$N_t(a)$ 表示在時間 t 之前選擇動作 a 的次數（(2.1) 中的分母），常數 $c > 0$ 以控制其探索的程度。如果 $N_t(a) = 0$，則 a 被認為是最大化動作。

這種*信賴上界*（*upper confidence bound, UCB*）動作選擇的概念是平方根項對 a 的估計值其不確定性或變異數的衡量。因此最大值的大小是動作 a 的可能真實值的上限，其中 c 決定了信賴區間。每當 a 被選擇時，不確定性可能會降低：$N_t(a)$ 增加，因為它出現在分母中，所以不確定性的項減少。另一方面，每當選擇除了 a 之外的動作時，t 增加而 $N_t(a)$ 不會增加；因為 t 出現在分子中，所以對不確定性的估計值會增加。使用自然對數意味著隨著時間的推移增加的速率會變慢，但是其值依然會趨近於無限大；最終所有動作都會被選擇，但是將隨著時間的推移，選擇具有較低值估計值或已經頻繁選擇的動作的機率將降低。

在 10- 搖臂測試平台上的信賴上界結果如圖 2.4 所示。信賴上界通常表現得很好，但是相比於 ε - 貪婪，它更難以從拉霸機問題擴展到本書其餘部分所考慮更為一般的強化學習情形。其中一個困難點在於處理非問定性問題；它需要比第 2.5 節中所提到的方法更複雜。另一個困難點是對於巨大狀態空間的處理，特別是當使用本書第二部分中所描述的函數近似方法時。在這些更為複雜的情形下，信賴上界動作選擇的概念通常是不理想的。

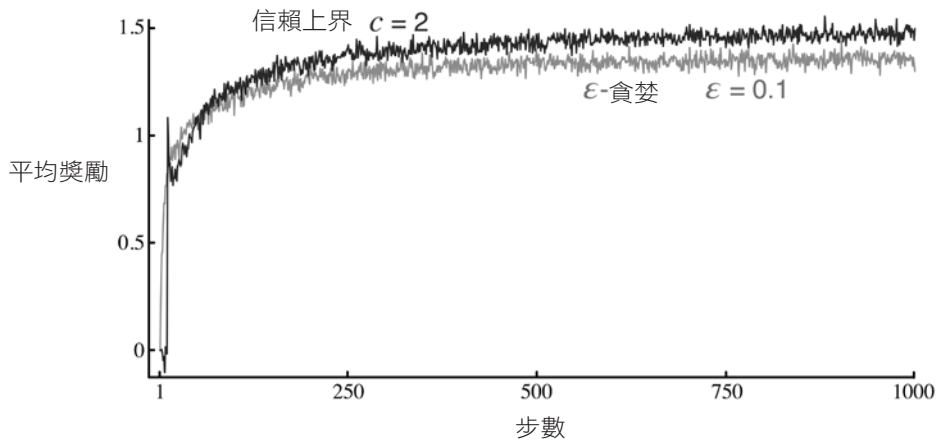


圖 2.4 10- 搖臂測試平台上 UCB 動作選擇的平均表現。如圖所示，UCB 通常比 ε - 貪婪動作選擇更好，除了在前 k 個步驟中，當 UCB 從尚未嘗試的動作中隨機選擇時。

練習 2.8：UCB 峰值。在圖 2.4 中，UCB 演算法在第 11 步時出現一個明顯的峰值。為什麼會這樣？請注意，為了使您的答案完全令人滿意，必須解釋為什麼獎勵在第 11 步增加及為什麼在隨後的步驟中減少。提示：如果 $c = 1$，則峰值將變得不明顯。 □

2.8 梯度拉霸機演算法

到目前為止，我們在本章已經學習到對動作值進行估計，並使用這些估計值來選擇動作的多種方法。這通常是一種很好的途徑，但並不是唯一可行的。在本節中，我們學習將每個動作 a 的**偏好**（*preference*）以數字量化，我們將其表示為 $H_t(a)$。偏好值越大，所對應的動作被採用的次數就越多，但偏好不能使用獎勵的觀念來解釋。只有一種動作相對於另一種動作的相對偏好才是有意義的；如果我們將 1000 加到所有動作的偏好值，那麼各動作被選擇的機率仍然保持不變，這是根據 ***soft-max* 分布**（即 Gibbs 或 Boltzmann 分布）所決定的，如下所示：

$$\Pr\{A_t = a\} \doteq \frac{e^{H_t(a)}}{\sum_{b=1}^{k} e^{H_t(b)}} \doteq \pi_t(a), \tag{2.11}$$

在這裡我們引入了一個實用的新符號 $\pi_t(a)$，用於表示在時步 t 採取行動 a 的機率。在初始時，所有動作偏好值都是相同的（例如對於所有 a，$H_1(a) = 0$），因此所有動作具有相同的被選擇的機率。

隨機梯度上升的梯度拉霸機演算法

我們可以將梯度拉霸機演算法理解為梯度上升的隨機近似來獲得更深入的理解。在典型的**梯度上升**（*gradient ascent*）中，各個動作的偏好值 $H_t(a)$ 將會正比於增量對性能表現的影響：

$$H_{t+1}(a) \doteq H_t(a) + \alpha \frac{\partial \mathbb{E}[R_t]}{\partial H_t(a)}, \tag{2.13}$$

其中表現的衡量標準為獎勵的期望值：

$$\mathbb{E}[R_t] = \sum_x \pi_t(x) q_*(x),$$

且對增量效果的衡量標準為其表現的衡量標準相對於動作偏好值的**偏導數**（*partial derivative*）。因為根據假設我們不知道 $q_*(x)$，所以不可能在現有情況下實現典型的梯度上升，但事實上從期望值的角度來看，演算法 (2.12) 中的更新等同於 (2.13) 中的更新，這使得前者成為了**隨機梯度上升**（*stochastic gradient ascent*）方法的實例。要證明這一點的推導只需要基礎的微積分知識，但是需要花費數個步驟。

首先讓我們更近一步地觀察表現的梯度：

$$\begin{aligned}
\frac{\partial \mathbb{E}[R_t]}{\partial H_t(a)} &= \frac{\partial}{\partial H_t(a)} \left[\sum_x \pi_t(x) q_*(x) \right] \\
&= \sum_x q_*(x) \frac{\partial \pi_t(x)}{\partial H_t(a)} \\
&= \sum_x \big(q_*(x) - B_t\big) \frac{\partial \pi_t(x)}{\partial H_t(a)},
\end{aligned}$$

其中 B_t 被稱為**基準線**，可以為任何不依賴於 x 的純量。我們可以在不改變等號的情況下加入基準線項，這是因為所有動作上梯度值的總和為 0，即 $\sum_x \frac{\partial \pi_t(x)}{\partial H_t(a)} = 0$ — 如果 $H_t(a)$ 改變的話，其中一些動作被選擇的機率將會增加而有一些將會減小，但機率的變化量的總和為 0，因為機率的總和永遠為 1。

接下來我們令上一式總和中的每一項都乘以 $\pi_t(x)/\pi_t(x)$：

$$\frac{\partial \mathbb{E}[R_t]}{\partial H_t(a)} = \sum_x \pi_t(x) \big(q_*(x) - B_t\big) \frac{\partial \pi_t(x)}{\partial H_t(a)} / \pi_t(x).$$

現在這一等式符合期望值的形式，對隨機變數 A_t 的所有可能值 x 加總，然後乘以這些值的機率。因此：

$$
\begin{aligned}
&= \mathbb{E}\left[(q_*(A_t) - B_t) \frac{\partial \pi_t(A_t)}{\partial H_t(a)} / \pi_t(A_t) \right] \\
&= \mathbb{E}\left[(R_t - \bar{R}_t) \frac{\partial \pi_t(A_t)}{\partial H_t(a)} / \pi_t(A_t) \right],
\end{aligned}
$$

其中我們選擇了基準線 $B_t = \bar{R}_t$，並將 $q_*(A_t)$ 替換為 R_t，可以這麼做的原因是因為 $\mathbb{E}[R_t|A_t] = q_*(A_t)$。等等我們會證明 $\frac{\partial \pi_t(x)}{\partial H_t(a)} = \pi_t(x)\big(\mathbb{1}_{a=x} - \pi_t(a)\big)$，該式中如果 $a = x$ 那麼 $\mathbb{1}_{a=x}$ 為 1，反之為 0。先假設這一點是成立的，那麼我們可以得知：

$$
\begin{aligned}
&= \mathbb{E}\big[(R_t - \bar{R}_t) \pi_t(A_t) \big(\mathbb{1}_{a=A_t} - \pi_t(a)\big) / \pi_t(A_t) \big] \\
&= \mathbb{E}\big[(R_t - \bar{R}_t) \big(\mathbb{1}_{a=A_t} - \pi_t(a)\big) \big].
\end{aligned}
$$

回想一下，我們想要將性能表現的梯度表示成我們可以在每一步採樣的期望值，如同我們剛剛完成的那樣，然後我們可以正比於樣本值對偏好值進行更新。將上式中的期望值替換為樣本值，並帶入 (2.13) 中，我們可以獲得：

$$
H_{t+1}(a) = H_t(a) + \alpha \big(R_t - \bar{R}_t\big) \big(\mathbb{1}_{a=A_t} - \pi_t(a)\big), \text{ 對於所有的 } a
$$

你會發現它等同於我們的原先的演算法 (2.12)。

接下來我們只需證明 $\frac{\partial \pi_t(x)}{\partial H_t(a)} = \pi_t(x)\big(\mathbb{1}_{a=x} - \pi_t(a)\big)$。回想一下導數的標準除法定則：

$$
\frac{\partial}{\partial x} \left[\frac{f(x)}{g(x)} \right] = \frac{\frac{\partial f(x)}{\partial x} g(x) - f(x) \frac{\partial g(x)}{\partial x}}{g(x)^2}.
$$

使用此定則，我們可以得到：

$$
\begin{aligned}
\frac{\partial \pi_t(x)}{\partial H_t(a)} &= \frac{\partial}{\partial H_t(a)} \pi_t(x) \\
&= \frac{\partial}{\partial H_t(a)} \left[\frac{e^{H_t(x)}}{\sum_{y=1}^{k} e^{H_t(y)}} \right] \\
&= \frac{\frac{\partial e^{H_t(x)}}{\partial H_t(a)} \sum_{y=1}^{k} e^{H_t(y)} - e^{H_t(x)} \frac{\partial \sum_{y=1}^{k} e^{H_t(y)}}{\partial H_t(a)}}{\left(\sum_{y=1}^{k} e^{H_t(y)} \right)^2} \qquad \text{（透過除法定則）}
\end{aligned}
$$

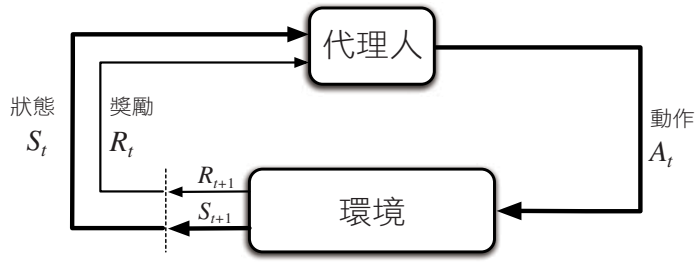


圖 3.1 在馬可夫決策過程中的代理人與環境之間的交互作用。

更具體地說，代理人和環境在一系列離散時步 $t = 0, 1, 2, 3, \cdots$進行交互作用[2]。在每個時步 t，代理人會接收到環境的狀態（*state*）$S_t \in \mathcal{S}$ 的一些表徵，並在此基礎上選擇一個動作（*action*）$A_t \in \mathcal{A}(s)$[3]。在下一個時步，代理人將會收到一個數字獎勵（*reward*）$R_{t+1} \in \mathcal{R} \subset \mathbb{R}$，此獎勵中有部分來自於於動作的結果，同時將發現自己處於一個新的狀態 S_{t+1}[4] 中。MDP 和代理人共同產生了一個如下所示的序列或軌跡：

$$S_0, A_0, R_1, S_1, A_1, R_2, S_2, A_2, R_3, \ldots \tag{3.1}$$

在有限 MDP 中，狀態、動作和獎勵（$\mathcal{S}$、$\mathcal{A}$ 和 $\mathcal{R}$）的集合都只有有限數量的元素。在這種情形下，隨機變數 R_t 和 S_t 具有明確定義的離散機率分布，該分布僅取決於先前的狀態和動作。也就是說，在給定前一狀態和動作的情形下，對於這兩個隨機變數的特定值，$s' \in \mathcal{S}$ 和 $r \in \mathcal{R}$，在時步 t 發生的機率為：

$$p(s', r \mid s, a) \doteq \Pr\{S_t = s', R_t = r \mid S_{t-1} = s, A_{t-1} = a\}, \tag{3.2}$$

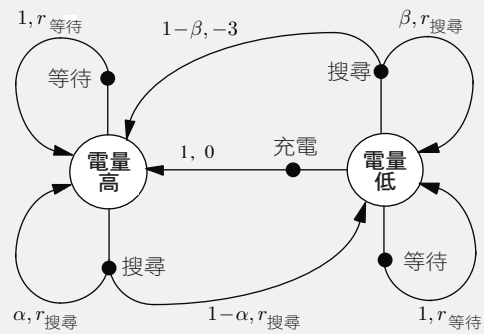
2 我們專注於離散的情形以簡化問題，即使許多想法可以擴展到連續時間的情形（例如，詳見 Bertsekas and Tsitsiklis, 1996; Doya, 1996）。

3 為了簡化符號，我們有時會假設一種特殊情形，即所有狀態下的動作集合都相同，並將其簡單地表示為 $\mathcal{A}$。

4 我們使用 R_t+1 而非 R_t 來表示由 A_t 產生的獎勵，因為它強調下一個獎勵 R_t+1 和下一個狀態 S_t+1 是共同決定的。不幸的是，在文獻中這兩種表示方式都是常用的表示方式。

一方面，如果起始電量等級為「低」，那麼在經過一段時間的搜尋後，電量等級以 β 的機率維持在「低」，而以 $1-\beta$ 的機率耗盡電池。在耗盡電池電量的情況下，機器人必須等待救援，且電量等級在充電後恢復為「高」。機器人收集的每一個汽水罐都可以作為一個單位獎勵，而每當機器人必須被救援時，會產生一個 -3 的獎勵。讓我們使用 $r_{搜尋}$ 和 $r_{等待}$（$r_{搜尋} > r_{等待}$）分別表示機器人在搜尋和等待時收集到的汽水罐預期數量（因此就是預期的獎勵）。最後，假設機器人在回充電站期間及電池耗盡時不能收集汽水罐。那麼這個系統就是一個有限的 MDP，我們可以獲得轉移機率和預期的獎勵，其動態如左表所示：

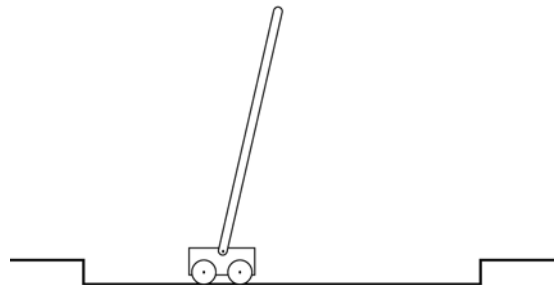
s	a	s'	$p(s'\|s,a)$	$r(s,a,s')$
高	搜尋	高	α	$r_{搜尋}$
高	搜尋	低	1−α	$r_{搜尋}$
低	搜尋	高	1−β	−3
低	搜尋	低	β	$r_{搜尋}$
高	等待	高	1	$r_{搜尋}$
高	等待	低	0	$r_{等待}$
低	等待	高	0	$r_{等待}$
低	等待	低	1	$r_{等待}$
低	充電	高	1	0
低	充電	低	0	0



表中標示了當前狀態 s、動作 $a \in \mathcal{A}(s)$ 和下一個狀態 s' 的每種可能組合。總結有限 MDP 動態的另一種有用方法是如右上圖所示的**轉移圖**（*transition graph*）。有兩種節點：**狀態節點**（*state nodes*）和**動作節點**（*action nodes*）。

每個可能的狀態都對應於一個狀態節點（一個以狀態名稱標記的大圓），以及每個狀態 - 動作對應的一個動作節點（一個小的實心圓圈，以動作的名稱標記並以一條線連接到狀態節點）。從狀態 s 開始並採取行動 a，你將從狀態節點 s 沿著線移動到動作節點（s, a），然後環境透過其中一個箭頭離開動作節點（s, a），轉移到下一個狀態的節點。每個箭頭對應一個 3 個參數的 (s, s', a)，其中 s' 是下一個狀態，我們用轉移機率 $p(s'|s,a)$ 以及該轉移的預期獎勵 $r(s,a,s')$ 標記箭頭。請注意，離開同一個動作節點的轉移機率總和為 1。

練習 3.4：請參考範例 3.3 中的表，繪製出對應 $p(s',r \mid s,a)$ 的表格。表格中應該包含 s、a、s'、r 以及 $p(s',r \mid s,a)$ 的行，以及每一個 $p(s',r \mid s,a) > 0$ 對應的列。 □



範例 3.4：桿平衡。此任務的目標是將力施加到沿著軌道移動的推車上，以便保持鉸接在推車上的桿不會翻倒：如果桿與垂直方向的夾角超過了給定值或推車出軌，則視為失敗。每次失敗後，桿子都會重置到垂直位置。這個任務可以被視為分節性事件，每個自然劃分的分節就是重複嘗試平衡桿子。在這種情形下，對於沒有發生失敗時每個時步的獎勵為 +1，因此每個時步中的回報是發生失敗前所經歷的時步數。在這種情形下，永遠成功地平衡桿子將使回報趨近於無窮大。或者我們可以使用折扣將桿平衡視為一種連續性任務。在這種情形下，每次失敗時獎勵為 –1，其他時間獎勵為 0。每次回報將與 $-\gamma^K$ 相關，其中 K 是失敗前的時步數。無論在哪種情形下，透過盡可能長時間保持桿平衡使回報最大化。 ■

練習 3.6：假設你將桿平衡視為一種分節性任務，但同時也使用折扣，除了失敗時的 –1 以外，所有獎勵均為 0。那麼每一時步的回報會是如何？這種回報與連續性並包含折扣的回報有何不同？ □

練習 3.7：想像你正在設計一個走迷宮的機器人。你決定在其逃出迷宮時獎勵為 +1，而在其他時間獎勵為 0。任務似乎自然地被劃分為各個分節 — 即連續走迷宮的嘗試 — 所以你決定將它視為一個分節性任務，其目標是最大化預期的總獎勵 (3.7)。在執行學習代理人一段時間之後，你發現它在逃離迷宮時沒有任何改進。這是發生了什麼問題？你是否有效地向代理人傳達了希望達成的目標？ □

練習 3.8：假設 $\gamma = 0.5$ 並且接收以下獎勵序列 $R_1 = -1$，$R_2 = 2$，$R_3 = 6$，$R_4 = 3$ 及 $R_5 = 2$，其中 $T = 5$。那麼 G_0、$G_1 \cdots G_5$ 分別為多少？提示：由後往前計算。□

練習 3.9：假設 $\gamma = 0.9$ 且獎勵序列是 $R_1 = 2$，緊接著是數值為 7 的無限序列。那麼 G_1 和 G_0 分別為多少？ □

練習 3.10：證明 (3.10) 中的第二個等式。 □