Invatarea prin recompensa

Value Iteration

Algoritmul Value Iteration

Calculeaza utilitatea starilor (recompensa pe termen lung):

$$U_{t+1}(S_i) = R(S_i) + \delta \max_a \sum_j T(S_i, a, S_j) * U_t(S_j)$$

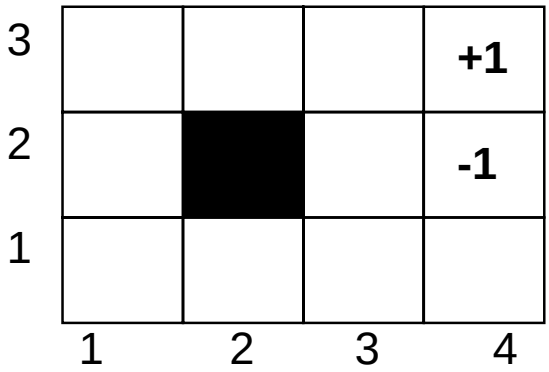
Intrare: T(model al tranzitiilor), R (recompensa starilor)
Iesire: U (utilitatea fiecarei stari)

{
U = utilitatea fiecarei stari (initial identica cu R)
U' = utilitatea fiecarei stari (la pasul urmator - initial identica cu R)
repeta
{
U ← U'
pentru fiecare stare S_i
$U'(S_i) = R(S_i) + \delta \max_a \sum_j T(S_i, a, S_j) * U(S_j)$
}pana cand distanta(U, U') < ε
return U
}

$\text{distanta}(U, U') = \frac{1}{S}\sqrt{\sum_{i=1}^{S}(U(S_i) - U'(S_i))^2}$, S = numarul de stari

Politica optima $\pi^*(S_i) = \arg\max_a \sum_j (R(S_i) + \delta * T(S_i, a, S_j) * U(S_j))$

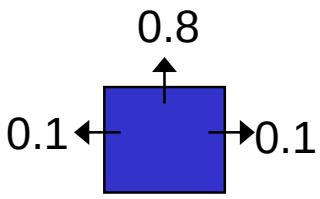
Exemplu: grid 3 x 4



- Doua stari terminale cu recompensele +1 si -1
- Toate celelalte stari au recompensele -0.04

- Deplasarea se poate face in directiile Nord/Sud/Est/Vest

Exemplu: Modelul mediului T - deplasare in directia N



Exemplu de calcul a utilitatii pentru starea S (δ = 1)

a)

	U=10	
U=5	**S (R=1)**	**U= -8**
	U=1	

U_{t+1}(S) = 1 + max(0.8*5+0.1*10+0.1*1, (←)
0.8*10+0.1*(-8)+0.1*5, (↑)
0.8*(-8)+0.1*10+0.1*1, (→)
0.8*1+0.1*5+0.1*(-8) (↓))=
= 1 + max (5,1 (←), 7.7 (↑), -5.3 (→), 0.5 (↓))
= 1 + 7.7 = 8.7 (↑)

b)

U=5	**U=10**
S (R=1, U_t=1)	**U=1**

U_{t+1}(S) = 1 + max(0.8*1+0.1*5+0.1*1, (←)
0.8*5+0.1*1+0.1*1, (↑)
0.8*1+0.1*1+0.1*5, (→)
0.8*1+0.1*1+0.1*1 (↓))=
= 1 + max (1.4 (←), 4.2 (↑), 1.4 (→), 1 (↓))
= 1 + 4.2 = 5.2 (↑)