Az UCT és a MoGo algoritmusokról

Szörényi Balázs

1 UCT: UCB-1 for trees

Algorithm 1 UCT-KIÉRTÉKELŐ($s_0 \in \mathcal{S}$)

INICIALIZÁLÁS
minden (s,a) párra $n(s,a) := 0$, $Q_{\text{UCT}}(s,a) := 0$
TANULÁS (ON-LINE)
for `szim` := 1 to `SzimulációSzám` **do**
  EPIZÓD GENERÁLÁSA
  Generáljunk egy $s_0, a_0, r_0, s_1, a_1, r_1, \ldots, s_T, a_T, r_T$ epizódot (avagy játszmát) a
  $\pi_{\text{UCT}} : s \mapsto \begin{cases} \text{egy véletlen választott lépés az } s\text{-ből még nem próbáltak közül,} & \text{ha van ilyen} \\ \text{argmax}_a \left(Q_{\text{UCT}}(s,a) + c \cdot \sqrt{\frac{n(s)}{n(s,a)}} \right) & \text{egyébként} \end{cases}$
  stratégiával self-play módban.
  UPDATE (MONTE-CARLO)
  for $t := 1$ to T **do**
    $n(s_t, a_t) := n(s_t, a_t) + 1$
    $n(s_t) := \sum_a n(s_t, a_t)$
    $Q_{\text{UCT}}(s_t, a_t) := Q_{\text{UCT}}(s_t, a_t) + (1/n(s_t, a_t))\big(R_t - Q_{\text{UCT}}(s_t, a_t)\big)$
    $\{R_t = r_t + r_{t+1} + \cdots + r_T =$ a t-edik lépés után szerzett összjutalom$\}$
  end for
end for
TÉNYLEGES LÉPÉS
return $\text{argmax}_a\, Q_{\text{UCT}}(s_0, a)$

Pl. $T = 3000$ vagy $T = 70000$.

2 MoGo: Monte-Carlo Go

π_{MoGo}: offline (azaz még éles alkalmazás előtt) kézzel összeállított véletlen stratégia, beépített (emberi) Go háttértudással. (Persze más probléma esetén használható egy ilyen helyett is valamilyen offline módon tanított stratégia.)

Q_{RLGO}: offline (azaz még éles alkalmazás előtt) megtanult cselekvés-érték függvény. Konkrétabban: ℓ darab, kézzel összeválogatott bináris $\phi_1, \ldots, \phi_\ell : \mathcal{S} \times \mathcal{A} \to \{0,1\}$ jellemző megfelelő $(\theta_1, \ldots, \theta_\ell$ súlyozására tanultak rá TD(0) algoritmussal (gradiens módszerrel) self-play-t alkalmazva. A súlyokból a cselekvés-érték függvény $\sigma(\sum_{i=1}^{\ell} \theta_i \cdot \phi_l(s,a))$ képlettel került kiszámításra, ahol $\sigma(y) = 1/(1-e^y)$ a szigmoid függvény.

Algorithm 2 $\mathrm{UCT}_{\mathrm{RAVE}}$-KIÉRTÉKELŐ$(s_0 \in \mathcal{S})$

INICIALIZÁLÁS
minden (s,a) párra $n(s,a) := m(s,a) := 50$, $Q_{\mathrm{UCT}}(s,a) := Q_{\mathrm{RLGO}}$
TANULÁS (ON-LINE)
for `szim` := 1 to `SzimulációSzám` **do**
  EPIZÓD GENERÁLÁSA
  Generáljunk egy $s_0, a_0, r_0, s_1, a_1, r_1, \ldots, s_T, a_T, r_T$ epizódot (avagy játszmát) az

$$s \mapsto \begin{cases} \pi_{\mathrm{MoGo}} \text{ szerint, ha van olyan lépés, amit az } s\text{-ből még nem próbáltunk} \\ \mathrm{argmax}_a (1-\beta(s,a)) \left(Q_{\mathrm{UCT}}(s,a) + c \cdot \sqrt{\frac{n(s)}{n(s,a)}} \right) + \beta(s,a) \left(Q_{\mathrm{RAVE}}(s,a) + c \cdot \sqrt{\frac{m(s)}{m(s,a)}} \right) \\ \text{egyébként} \end{cases}$$

  stratégiával self-play módban, ahol $\beta(s,a) = \sqrt{k/(k+3n(s))}$.
  UPDATE (MONTE-CARLO)
  for $t := 1$ to T **do**
    $n(s_t, a_t) := n(s_t, a_t) + 1$
    $n(s_t) := \sum_a n(s_t, a_t)$
    $Q_{\mathrm{UCT}}(s_t, a_t) := Q_{\mathrm{UCT}}(s_t, a_t) + (1/n(s_t, a_t)\big(R_t - Q_{\mathrm{UCT}}(s_t, a_t)\big)$
  end for
  UPDATE (RAVE: Rapid Action Value Estimation)
  {Heurisztika: "a későbbi lépések akár hamarabb is ugyanolyan jók lennének".}
  for $t := 1$ to T **do**
    for $\tau := t$ to T **do**
      $m(s_t, a_\tau) := n(s_t, a_\tau) + 1$
      $m(s_t) := \sum_a n(s_t, a_\tau)$
      $Q_{\mathrm{RAVE}}(s_t, a_\tau) := Q_{\mathrm{RAVE}}(s_t, a_\tau) + (1/n(s_t, a_\tau)\big(R_t - Q_{\mathrm{RAVE}}(s_t, a_\tau)\big)$
    end for
  end for
end for
TÉNYLEGES LÉPÉS
return $\mathrm{argmax}_a\, Q_{\mathrm{UCT}}(s_0, a)$ vagy $Q_{\mathrm{RAVE}}(s_0, a)$

Pl. $k = 1000$.

References

[1] P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Mach. Learn.*, 47(2-3):235–256, 2002.

[2] S. Gelly and D. Silver. Combining online and offline knowledge in UCT. In *ICML '07: Proceedings of the 24th international conference on Machine learning*, pages 273–280, New York, NY, USA, 2007. ACM.

[3] L. Kocsis and C. Szepesvári. Bandit based monte-carlo planning. In *ECML*, pages 282–293, 2006.