Teoria dei giochi: / 23-05

Morra cinese

$$\begin{array}{cc} & \text{Carlo} \\ \text{Roberta} & \begin{pmatrix} 0 & 1 & -2 \\ -1 & 0 & 3 \\ 2 & -3 & 0 \end{pmatrix} \end{array} \quad \text{matrice} \rightarrow M$$

$M_{i,j}$ è payoff
→ se > 0
è quanto Carlo paga Roberta se C gioca la colonna j quando Roberta le riga i
→ se < 0
Roberta paga Carlo

1 carta
2 sasso
3 forbice
} indici

se Carlo gioca carta e Roberta forbice, Carlo paga 2
se Carlo gioca forbice e Roberta carta, Carlo incassa 2 ("paga -2")

E' a somma zero: somma di quanto paga Carlo e quanto riceve Roberta è uguale a zero
(gioco la stessa cosa)

Panni di Roberta: gioco sempre sasso per giocare al meglio
scegliere il migliore degli scenari peggiori } → perché perde 1 alle peggio (non conta quanto guadagna)

$V_R = -1$

Panni di Carlo: gioca sempre sasso, alle peggio perde $V_C = 1$

Quando $V_R \neq V_C$ il gioco non ha valore max per riga

$$V_R = \max_i \min_j M_{ij} = \max_i \{-2, -1, -3\} = -1$$

$$V_C = \min_j \max_i M_{ij} = \min_j \{2, 1, 3\} = 1$$

min per colonna

Disuguaglianza fondamentale: $\max_i \min_j M_{ij} \leq \min_j \max_i M_{ij}$

Dimostrazione

$$\min_j M_{kj} \leq M_{kl} \qquad \forall k, l$$

per ogni riga k, ogni elemento l ($\forall l$) non è più piccolo del minimo

$$\max_i \min_j M_{ij} \leq \max_i M_{il}$$

allora vale anche per il massimo (vero per la riga dei massimi)

$$\max_i \min_j M_{ij} \leq \min_j \max_i M_{ij}$$

ma allora vale anche per il minimo del massimo

Teorema di Von Neumann

In un gioco a somma zero, quello che non si può ottenere con strategie pure si può ottenere con strategie miste

$$p^* = \left[\tfrac{1}{2}, \tfrac{1}{3}, \tfrac{1}{6}\right]^T \text{ carta} \qquad p^{*T} M q^* = 0$$

$$q^* = \left[\tfrac{1}{2} \; \tfrac{1}{3} \; \tfrac{1}{6}\right]^T \text{ roberta} \qquad p^{*T} \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} = 0$$

$$= \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} q^* = 0$$

Il gioco ha così valore, ovvero alla lunga Carlo e Roberta sono pari

Nuovo caso, Roberta sa la strategia di Carlo, la miglior strategia e' sasso:

$$V_R: \frac{2}{3} \quad \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \longrightarrow \cdot \; M \begin{pmatrix} 1/3 \\ 1/3 \\ 1/3 \end{pmatrix} = \begin{pmatrix} -1/3 \\ 2/3 \\ -1/3 \end{pmatrix} \qquad = \frac{2}{3}$$

metodi equivalenti valore atteso $V_R = \frac{2}{3}$

Roberta: Perde $\frac{1}{3}$ vs carta; Patta $\frac{1}{3}$ vs sasso; Vince $\frac{1}{3}$ vs forbice; delle volte

$$-\frac{1}{3} + 0 + 1 = \frac{2}{3}$$

$$M_{ij} = T(I_i, A_j)$$

tempo di esecuzione di un dato algoritmo

Principio di Yao

Tempo migliore su input peggiore

$$V_C = \min_{A \in \mathcal{A}} \max_{I \in \mathcal{I}} T(I, A)$$

$\forall$ q su A e p su $\mathcal{I}$

$$\underbrace{\min_{A \in \mathcal{A}}}_{\text{alg det}} E[T(I_p, A)] \leq \max_{I \in \mathcal{I}} E[T(I, A_q)]$$

alg randomizzato

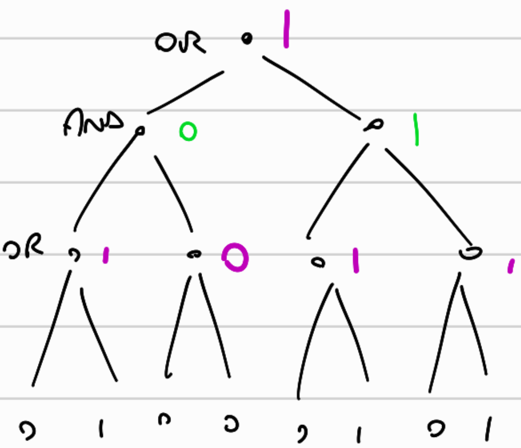
T. Von Neumann

$$\forall p', q' \quad \min_{q'} \max_{p'} E[T(I_{p'}, A_{q'})] = \max_{p'} \min_{q'} E[T(I_{p'}, A_{q'})]$$

La disuguaglianza sopra mi afferma che il val atteso (caso medio) del miglior alg deterministico è un lim inferiore per il tempo di esecuzione di un alg randomizzato

$$\min_{q'} \max_{p'} E[T(I_{p'}, A_{q'})] \le \max_{p'} \min_{p'} E[T(I_{p'}, A_{q'})]$$

Albero di un gioco



un giocatore vuole massimizzare (avere 1)
↳ OR

l'altro " minimizzare (avere zero)
↳ AND

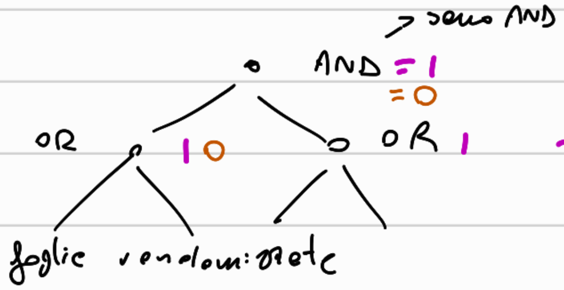
Posso leggere le foglie fino alla radice, mi fa capire che se OR gioca per primo AND non vince, se si muove verso un nodo 1

$T_{d,k}$ d sono figli di un nodo
d^{2k} sono i nodi a distanza $2k$ dalla radice

in un albero binario $T_{2,k}$ dove k è la profondità

si vuole dimostrare che $T_{2,k} \sim 3^k < 2^{2k} = 4^k$

Con 16 foglie avrei un $3^2 = 9$ nodi



$k = 1$

→ valori richiesti per vincere

$$\frac{1}{2}(1+2) + \frac{1}{2}(1+2) = 3$$

↗ p unif.

↑ a volte ne leggo una (basta il 1° uno) altrimenti 2

Ho AND = 0 invece per un OR = 0: devo leggere entrambe le foglie

$\frac{1}{2}(2)$ se zero ho finito

se uno: $\frac{1}{2}(2) + \frac{1}{2}(2+2) = 3$

devo aver letto le 2 precedenti + le 2 attuali

se randomizzo non devo leggere 4 foglie ma 3 (meglio su grandi n)

$n = 4^k$ $\quad \log_4 3 = 0{,}793$

↓

$\log_4 n = k \longrightarrow 3^k = 3^{\log_4 n} = 4^{(\log_4 3)\log_4 n} = n^{0{,}793}$

sublineare in n

si vuole ottimizzare $\alpha = 0{,}793$

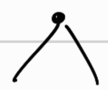
$$p = \frac{3-\sqrt{5}}{2} \qquad 1-p = \frac{\sqrt{5}-1}{2}$$

proprietà

$p + (1-p) = 1$

$(1-p)^2 = p$

Sia T un albero NOR $(1-p)$



0 con $p = (1-p)$ $\qquad$ 0 con $p = (1-p)$

Data la profondità in $\log_2 n$

$$W_T(h) = W_T(h-1) + \underbrace{(1-p)\, W_T(h-1)}_{\text{se zero devo controllare entrambi}}$$

$$W_T(h) = (2-p)\, W_T(h-1)$$

$$\rightarrow (2-p)^{\log_2 n} = n^{0,694}$$ migliore di $\alpha = 0,793$?

Non proprio, ho considerato un'indipendenza fra i nodi per questo viene più piccolo, ma questa indipendenza non esiste.

GUAR DIPENDENTI

Quindi posso trovare una strategia vincente tramite un alg rand che risolve l'albero con un tempo atteso sublineare