Korelirana ravnovesja
Projekt pri predmetu Matematika z računalnikom

Melisa Brulić

19. maj 2024

1 Uvod

Oglejmo si primer igre Strahopetec (angl. Game of Chicken). Dva voznika iz različnih ulic vozita proti križišču. Vsak ima dve možnosti - lahko se ustavi ali vozi naprej. Če bo prvi vozil naprej, je za drugega bolje, da se ustavi. Če pa se bo prvi prestrašil in ustavil, bo drugemu ljubše, če vozi naprej. Nobeden ne želi biti strahopetec, vendar če se nobeden ne ustavi, je izid trčenje, kar je najslabši možen izid za oba. To pripelje do zanimive situacije, kjer si oba želita voziti, a samo v primeru, ko se drugi ustavi. Igro lahko predstavimo v matrični obliki:

		Voznik 2: Ustavi	Voznik 2: Vozi
Voznik 1	Ustavi	4,4	1,5
	Vozi	5,1	0,0

Ta igra ima tri Nasheva ravnovesja. Čisti Nashevi ravnovesji sta (Ustavi, Vozi) in (Vozi, Ustavi). Igra ima še eno mešano Nashevo ravnovesje, kjer se oba voznika ustavita z verjetnostjo $\frac{1}{2}$.

Opazimo, da sta čisti Nashevi ravnovesji "nepravični", saj je za enega od igralcev izid veliko boljši kot za drugega. Tisti, ki se ustavi, dobi koristnost 1, tisti, ki vozi naprej pa koristnost 5. Skupno zadovoljstvo pri čistih Nashevih ravnovesjih je tako 6. Mešano Nashevo ravnovesje je bolj "pravično", saj imata v tem primeru oba igralca enako pričakovano koristnost $\frac{1}{4}(4+1+5) = \frac{5}{2}$. Skupno zadovoljstvo pa je tako samo $\frac{5}{2}$, kar je manj kot pri čistem Nashevem ravnovesju.

Denimo, da je na križišču postavljen semafor, ki vozniku pove, ali naj se ustavi ali naj vozi naprej. Denimo, da semafor naključno pokaže (Ustavi, Vozi) ali (Vozi, Ustavi) z verjetnostjo $\frac{1}{2}$. V tem primeru je pričakovana koristnost posameznega voznika $\frac{1}{2}(5 + 1) = 3$, skupno zadovoljstvo pa 6. To je torej "poštenašitaucija, ki pa hkrati nudi večje skupno zadovoljstvo kot mešano Nashevo ravnovesje.

Lahko premislimo, da se nobenemu od voznikov ne splača odstopiti od predlagane poteze:

- če semafor vozniku pove, da vozi naprej: voznik pričakuje, da se bo drugi ustavil in tako sam dobi 5. Če bi spremenil potezo in se tudi on ustavil, bi dobil samo 1.
- če semafor vozniku pove, da se ustavi: voznik pričakuje, da bo drugi vozil in tako bi sam dobil 1. Če bi spremenil potezo in tudi on vozil naprej, bi dobil 0.

S spremembo poteze bi tako oba voznika dobila manj. Primer služi za motivacijo novega ravnovesja, ki ga imenujemo *korelirano ravnovesje*.

2 Cilj projekta

V projektu obravnavamo strateške igre z dvema igralcema, kjer ima vsak igralec na voljo dve potezi in je koristnost vsakega igralca naključno celo število med 0 in 10. Za tako igro izračunamo Nasheva in korelirana ravnovesja ter za vsako ravnovesje izračunamo, koliko je pričakovana koristnost posameznega igralca in koliko je skupno zadovoljstvo.

3 Definicije

Definicija 1. Končna strateška igra *je* $\langle N, S, u\rangle$*, kjer:*

- $N = \{1, \ldots, n\}$ *je* množica igralcev
- *za vsak* $p \in N$ *je* S_p, $|S_p| \geq 2$, *končna množica* strategij *oz.* potez *igralca* p
- izid igre *oz.* profil strategij *je vektor potez vseh igralcev* $s = (s_1, \ldots, s_n)$*;* $\forall p \in N : s_p \in S_p$
- množica vseh možnih izidov *oz.* profilov strategij *je* $S = S_1 \times \cdots \times S_n$
- funkcija koristnosti *je funkcija* $u_p : S \to \mathbb{R}$*, kjer za izid* $s \in S$ *vrednost* $u_p(s)$ *predstavlja preference igralca* p*:*

$$p \text{ ima raje izid } s \text{ kot } \tilde{s} \iff u_p(s) \geq u_p(\tilde{s}).$$

Za igralca $p \in N$ in izid $s \in S$ bomo namesto $u_p(s)$ pisali u_s^p. S S_{-p} označimo poteze nasprotnikov igralca p, tj. $\prod_{q \neq p} S_q$.

Definicija 2. *Porazdelitev* x *na množici vseh možnih izidov* S *je* produktna porazdelitev*, če za vsakega igralca* p *obstaja porazdelitev* x^p *na množici* S_p*, tako da je verjetnost izida* $s = (s_1, \ldots, s_n)$ *enaka* $x_s = \prod_{p=1}^{n} x^p_{s_p}$.

Definicija 3. Korelirano ravnovesje *je slučajna porazdelitev x na množici vseh možnih izidov S, da za vsakega igralca $p \in N$ in vse poteze $i, j \in S_p$ velja naslednje: Pod pogojem, da je pri izbranemu izidu iz x p-ta komponenta enaka i, je pričakovana koristnost za igralca p pri igranju strategije i vsaj toliko kot pri igranju j:*

$$\sum_{s \in S_{-p}} [u^p_{is} - u^p_{js}] x_{is} \geq 0,$$

kjer is označuje izid, ki ga dobimo tako, da izidu $s \in S_{-p}$ dodamo še komponento $i \in S_p$.

Intuitivno si lahko predstavljamo, da nek posrednik izbere izid s iz porazdelitve x in vsakemu igralcu posebej pove, katero potezo naj odigra. Če posrednik igralcu p svetuje, da odigra potezo i in ta predvideva, da bodo vsi ostali igralci upoštevali posrednikov nasvet, igralec p ne bo želel igrati nobene druge poteze, saj bi s tem dobil manj oz. kvečjemu toliko kot z igranjem poteze i. Vsako (mešano) Nashevo ravnovesje je tudi korelirano ravnovesje,

4 Izračun Nashevih ravnovesij

Za izračun Nashevih ravnovesij 2x2 igre uporabimo Pythonovo knjižnico Nashpy. Pri tem uporabimo že implementirani algoritem (Vertex enumeration algorithm).

5 Izračun koreliranih ravnovesij

Za izračun koreliranih ravnovesij uporabimo algoritem iz članka. Omejili se bomo na 2×2 strateške igre. Imamo torej 2 igralca z dvema možnima potezama 0 in 1. Naj bodo koristnosti igralcev dane z matriko

	0	1
0	a_{00}, b_{00}	a_{01}, b_{01}
1	a_{10}, b_{10}	a_{11}, b_{11}

5.1 Linearen program in obstoj koreliranega ravnovesja

Naj bo $x = (x_{00}, x_{01}, x_{10}, x_{11})$. Pogoj za korelirano ravnovesje je potem

$$\begin{aligned}
(a_{00} - a_{10})x_{00} - (a_{01} - a_{11})x_{01} &\geq 0 \\
(a_{10} - a_{00})x_{10} - (a_{11} - a_{01})x_{11} &\geq 0 \\
(b_{00} - b_{01})x_{00} - (b_{10} - b_{11})x_{10} &\geq 0 \\
(b_{01} - b_{00})x_{01} - (b_{11} - b_{10})x_{11} &\geq 0,
\end{aligned}$$

kar lahko zapišemo z matriko kot $Ux \geq 0$, kjer je

$$U = \begin{bmatrix} a_{00} - a_{10} & a_{01} - a_{11} & 0 & 0 \\ 0 & 0 & a_{10} - a_{00} & a_{11} - a_{01} \\ b_{00} - b_{01} & 0 & b_{10} - b_{11} & 0 \\ 0 & b_{01} - b_{00} & 0 & b_{11} - b_{10} \end{bmatrix}.$$

Oglejmo si linearen program (P)

$$\begin{aligned} \max \quad & x_{00} + x_{01} + x_{10} + x_{11} \\ \text{pri pogojih:} & \\ & Ux \geq 0 \\ & x \geq 0. \end{aligned}$$

Ta linearen program je ali trivialen (z maximumom 0) ali pa neomejen, kar je natanko tedaj, ko ima igra korelirano ravnovesje (to dobimo potem tako, da x normiramo). Spomnimo se, da je linearen program neomejen natanko tedaj, ko je njegov dual nedopusten. Dualni problem (D) je

$$\begin{aligned} U^T y &\leq -1 \\ y &\geq 0, \end{aligned}$$

kjer je $y = (y_0 0, y_0 1, y_1 0, y_1 1)^T$ in y_{ij} predstavlja težnjo, da igralec spremeni potezo iz i v j.

Lema 1. *Za vsak $y \geq 0$ obstaja produktna porazdelitev x, da velja $xU^T y = 0$.*

Iz leme sledi, da je (D) nedopusten, saj je $xU^T y$ konveksna kombinacija elementov $U^T y$. Če bi bil y dopusten, bi tako vrednosti $U^T y$ bile kvečjemu -1 in bi posledično $xU^T y$ moralo biti negativno število.

Dokaz leme najdemo v članku. Za nas bo pomembna samo ideja dokaza, ki jo potrebujemo za algoritem, zato izpustimo tehnične podrobnosti.

Ideja dokaza. Lema zagotavlja produktno porazdelitev x, zato naj bo $x_{ij} = x_i^1 x_j^2$, kjer je x_i^1 verjetnost, da igralec 1 igra i in x_j^2 verjetnost, da igralec 2 igra j za $i, j = 0, 1$. DOKONČAJ □ □

S tem smo dokazali, da ima vsaka igra korelirano ravnovesje.

5.2 Algoritem Ellipsoid against hope

Korelirana ravnovesja poiščemo s pomočjo algoritma, opisanega v članku. Naj bo $A = \{y \geq 0 | U^T y \leq -1\}$ množica dopustnih rešitev duala D. Ideja je, da uporabimo t.i. elipsoidni algoritem (angl. Ellipsoid algorithm) na dualu (D) in v vsakem koraku algoritma, uporabimo lemo.

5.2.1 Ideja elipsoidnega algoritma

Elipsoidni algoritem v splošnem deluje tako, da najprej poišče elipsoid, ki vsebuje množico A. Če je središče elipsoida element množice A, se algoritem konča in vrne to središče. Če pa je vrednost v središču izven množice, elipsoid razpolovimo tako, da je množica A vsebovana v eni od polovic elipsoida in ustvarimo nov elipsoid, ki pokrije želeno polovico prvotnega elipsoida. Postopek ponavljamo, dokler ne najdemo središče elipsoida, ki leži v množici A ali pa je elipsoid premajhen. V tem primeru zaključimo, da je množica A prazna. Ker je dual (D) nedopusten, bo elipsoidni algoritem sporočil, da je množica prazna.

5.2.2 Algoritem

Začetek algoritma V našem primeru za prvotni elipsoid vzamemo kroglo s središčem v 0 in polmerom $r = u^N$, kjer je $u = \max |U_{ij}|$ in $N = 4$ število spremenljivk dualnega problema.

Korak algoritma V i-tem koraku algoritma preverimo, ali središče elipsoida y_i zadošča linearnemu programu (D). Ker je (D) nedopusten, mu y_i ne more zadoščati, zato neenakost $U^T y \leq -1$ ni izpolnjena. Lema nam zagotovi porazdelitev x_i, tako da je $x_i U^T y_i = 0$. Naprej itak ne razumem...

Konec algoritma

6 Primeri in rezultati

6.1 Strahopetec

6.2 Bitka spolov