Zusammenfassung Nichtkooperative Netze

1. Einleitung

Voraussetzungen: Systeme eigenständiger Agenten, die nur ihren persönlichen Nutzen maximieren möchten. Dabei hängt dieser Nutzen aber auch vm Verhalten der anderen Agenten ab. Keine zentrale Regulierungsinstanz oder Kooperation.
$\Rightarrow$ Nicht-kooperative Spieltheorie

In unseren Systemen gibt es auch einen sozialen Nutzen, der im Allgemeinen durch eigennütziges Verhalten nicht maximiert wird.

Fragen:

i) Stabile Zustände? $\to$ Nash-Equilibrium
ii) Verlust durch eigennütziges Verhalten? $\to$ Preis der Anarchie
iii) Wovon ist dieser Verlust abhängig?
iv) Konstruktion, so dass stabile Zustände möglichst optimalen sozialen Nutzen aufweisen?

Definition: Ein Spiel $\Gamma = (n, (S_i)_{i=1}^n, (u_i)_{i=1}^n)$ besteht aus n Spielern, pro Spieler $i \in [n]$ einer Strategiemenge S_i und einer Nutzenfunktion $u_i : S_1 \times \cdots \times S_n \to \mathbb{R}$. (Wir betrachten, sofern nicht anders angemerkt, nur endliche Spiele.)

Definition: Gemischte Strategien $\sigma_i \in \Delta(S_i)$. Gemischtes Strategieprofil ergibt (eindeutige!) Wahrscheinlichkeitsverteilung über S, $\mathsf{S} := (\sigma_i)_{i=1}^n \in \Delta_{\perp\!\!\!\perp}(S) \cong \times_{i=1}^n \Delta(S_i)$. Dabei bezeichnet $\Delta_{\perp\!\!\!\perp}(S)$ die Menge der komponentenweise stochastisch unabhängigen Wahrscheinlichkeitsverteilungen über S.

Jedes reine Strategie kann auch als (degenerierte) gemischte Strategie aufgefasst werden!

Für $\mathsf{s} = (s_i)_{i=1}^n \in S$: $\mathsf{S}(\mathsf{s}) = \prod_{i=1}^n \sigma(s_i)$. Es wird u_i auf $\times_{i=1}^n \Delta(S_i)$ eindeutig multilinear fortgesetzt, $u_i(\mathsf{S}) := \sum_{\mathsf{s} \in S} \mathsf{S}(\mathsf{s}) \cdot u_i(\mathsf{s})$. Demnach gilt $u_i(\mathsf{S}) = \mathrm{E}_S[u_i(T)]$, wobei T Zufallsvariable für das tatsächlich gespielte reine Strategieprofil ist. Mit anderen Worten, $u_i(\mathsf{S})$ ist der Erwartungswert für den privaten Nutzen für Spieler i bei gemeinsamer Wahrscheinlichkeitsverteilung S.

Notation:

- $S := S_1 \times \cdots \times S_n$
- $(\mathsf{s}_{-i}, s_i') := (s_1, \ldots, s_{i-1}, s_i', s_{i+1}, \ldots, s_n)$

Da wir meistens von Kosten sprechen (und nicht vom Nutzen), verwenden wir statt u_i meist eines private Kostenfunktion $\mathsf{PC}_i : S \to \mathbb{R}_{\geq 0}$. Die sozialen Kosten sind $\mathsf{SC} : S \to \mathbb{R}_{\geq 0}$. Beide Funktionen werden ebenfalls eindeutig fortgesetzt auf $\times_{i=1}^n \Delta(S_i)$.

Definition: Ein eigennütziger Schritt (selfish step) ist $(\mathsf{S}, (\mathsf{S}_{-i}, \sigma_i')) \in \Delta_{\perp\!\!\!\perp}(S)^2$, wobei $u_i(\mathsf{S}_{-i}, \sigma_i') > u_i(\mathsf{S})$.

Definition: $G_\Gamma = (S, E)$ mit $E := \{(\mathsf{s}, \mathsf{t}) \mid (\mathsf{s}, \mathsf{t})$ (reiner) eigennütziger Schritt$\}$ heißt Nash-Graph von Γ.

Klar: Ein NE hat in G_Γ Ausgangsgrad 0. Hat G_Γ keine Kreise, existiert ein NE. G_Γ kann Pfad exponentielle Länge enthalten.

Satz: (Nash, 1950) Ein gemischtes NE existiert immer.

Komplexität der Berechnung eines (gemischten) NE ist unbekannt. Selbst bei Beschränkung der Eingabe auf beliebige Spiels mit nur 2 Spielern ist nicht klar, ob es einen polynomiellen Algorithmus zur Berechnung eines (gemischten) NE gibt.

2. Auslastungspiel I

Definition: Auslastungsspiel $\Gamma = (n, (w_i)_{i=1}^n, E, (f_e)_{e \in E}, (S_i)_{i=1}^n)$, wobei n Anzahl Spieler, w_i Gewicht von Spieler i, E Menge von Betriebsmitteln, $f_e : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ Kosten-/Latenzfunktionen und $S_i \subseteq \mathcal{P}(E)$. Γ heißt ungewichtet, falls $w_i = 1$ für alle $i \in [n]$.

Definition: Die Last auf einer Ressource e bei reinem Strategieprofil s ist $l_e(\mathsf{s}) = \sum_{i \in [n] | s_i \ni e} w_i$. Analog zum Nutzen wird auch die Last kanonisch zur erwarteten Last bei gemischten Strategieprofilen fortgesetzt.

Die Latenz auf einer Ressource e bei reinem Strategieprofil $\mathsf{s} \in S$ ist demnach $f_e(l_e(\mathsf{s}))$. Die (erwarteten) privaten Kosten für Spieler i sind $\mathsf{PC}_i(\mathsf{S}) = \sum_{\mathsf{s} \in S} \mathsf{S}(\mathsf{s}) \sum_{e \in s_i} f_e(l_e(\mathsf{s}))$. (Den Nutzen u_i könnte man bspw. als additives Inverses auffassen.)

Definition: Netzwerk-Auslastungsspiel $\Gamma = (n, (w_i)_{i=1}^n, V, E, (f_e)_{e \in E}, (o_i, d_i)_{i=1}^n)$, wobei n Anzahl Spieler, w_i Gewicht von Spieler i, (V, E) ein Graph, f_e Latenzfunktionen auf Kante e und (o_i, d_i) Start-/Ziel-Paar für Spieler i. Die (impliziten) Strategiemengen S_i sind jeweils alle Pfade von o_i nach d_i.

2.1. Existenz reiner Nash-Equilibrien

Satz: Für ungewichtete Auslastungsspiele ist $\mathsf{s} \mapsto \sum_{e \in E} \sum_{k=1}^{l_e(\mathsf{s})} f_e(k)$ eine exakte Potenzialfunktion.

Folglich existiert immer ein reines NE.

Folgerung: Ein reines Nash-Equilibrium kann in Zeit $\mathcal{O}(\sum_{e \in E} \sum_{k=1}^{n} f_e(k))$ berechnet werden.

Lemma: Auslastungsspiele mit Gewichten besitzten im Allgemeinen kein reines NE.

Satz: Ein gewichtetes Auslastungsspiel mit nichtfallenden Latenzfunktionen, bei dem jede Strategie nur aus genau einer Ressource besteht, hat immer ein reines Nash-Equilibrium.

Satz: Für gewichtete Auslastungsspiele mit Latenzfunktionen ausschließlich der Form $x \mapsto ax + b$, $a, b \geq 0$, ist $\mathsf{s} \mapsto \sum_{i=1}^{n} \sum_{e \in s_i} w_i \cdot (f_e(l_e(\mathsf{s})) + f_e(w_i))$ eine exakte Potenzialfunktion. Folglich existiert immer ein reines NE.

Folgerung: Ein reines Nash-Equilibrium für gewichtete Auslastungsspiele mit affin-linearen Latenzfunktionen kann in pseudo-polynomieller Zeit berechnet werden.

3. KP-Spiel

Definition: Das KP-Modell entspricht einem einfachen Auslastungsspiel mit m disjunkten Pfaden (Links) von einem Start- zu einem Zielknoten. Formal: $\Gamma = (n, (w_i)_{i=1}^{n}, m, (c_k)_{k=1}^{m})$, wobei n Anzahl Spieler, w_i Gewicht von Spieler i, m Anzahl Links, c_k Kapazität von Link k. Die (impliziten) Strategiemengen sind $[m]$ bzw. (im Falle von restricted links) Teilmengen davon.

Klar: $f_k(x) = \frac{1}{c_k} \cdot x$, $l_k(\mathsf{s}) = \sum_{i \in [n] | s_i = k} w_i$. Ferner $\mathsf{PC}_i(\mathsf{s}) = f_{s_i}(l_{s_i}(\mathsf{s})) = \frac{l_{s_i}(\mathsf{s})}{c_{s_i}}$.

Definition: Die sozialen Kosten werden definiert als erwartete maximale private Kosten, $\mathsf{SC}(\mathsf{S}) = \sum_{\mathsf{s} \in S} \mathsf{S}(\mathsf{s}) \cdot \max_{k \in [m]} f_k(l_k(\mathsf{s}))$.

Bemerkungen: Das KP-Modell entspricht dem Scheduling-Problem (Last-Balanzierung). Bekannte Approximationsalgorithmen sind die Graham-Heuristik mit relativer Güte $2 - \frac{1}{m}$ oder „Least Processing Time First“ (= Graham-Heuristik mit vorheriger absteigender Sortierung der Job-Größen) mit relativer Güte $\frac{4}{3} - \frac{1}{m}$. (Güte jeweils bei identischen Maschinen.)

3.1. Berechnung reiner Nash-Equilibrien

Satz: Der LPT-Algorithmus berechnet für das KP-Modell ein reines NE s mit $\frac{\mathsf{SC}(s)}{\mathsf{OPT}} \leq 2 - \frac{1}{m}$. (Dies ist ensprechend auch die relative Güte als Scheduling-Approximationsalgorithmus für verwandte Maschinen!)

Folgerung: Im KP-Modell existiert immer ein reines NE. Ein solches ist polynomiell berechenbar.

Satz: Der selfish step Algorithmus, gestartet mit einem beliebigen Strategieprofil, terminiert immer.

Definition: Ein selfish step heißt greedy, wenn es der beste für den seine Strategie wechselnden Benutzer ist.

Satz: Es gibt Instanzen mit exponentiell (in der Anzahl Links) großen Folgen von greedy selfish Steps.

Algorithmus NASHIFY:
Eingabe: Anzahl Spieler $n \in \mathbb{N}$, Anzahl Links $m \in \mathbb{N}$, Strategieprofil $\mathsf{s} = (s_i)_{i=1}^{n} \in S$, Spielergewichte $(w_i)_{i=1}^{n}$

for $k := 1, \dots, m$ **do** $W[k] := 0$
for $i := 1, \dots, n$ **do** $W[s_i] := W[s_i] + w_i$
Ohne Einschränkung sei (w_i) absteigend sortiert (andernfalls entsprechend umbenennen)
for $i := 1, \dots, n$ **do**
    $W[s_i] := W[s_i] - w_i$
    $s_i := \operatorname{argmin} k \in [m]\{W[k] + w_i\}$
    $W[s_i] := W[s_i] + w_i$
Gebe s aus

Satz: NASHIFY berechnet in $\mathcal{O}(n \cdot \log n)$ ein NE s' mit $\mathsf{SC}(\mathsf{s}') \leq \mathsf{SC}(\mathsf{s})$.

Bemerkungen: Für verwandte (related) Links gibt es NASHIFYRELATED mit Laufzeit $\mathcal{O}(m^2 \cdot n)$.

3.2. Beste und schlechteste Nash-Equilibrien

Satz: Das Entscheidungsproblem, ob es ein NE s mit $\mathsf{SC}(\mathsf{s}) \leq k$ gibt, ist NP-vollständig. Ebenso für „$\geq$“.

3.3. Koordinationsfaktor

Satz: Der reine Preis der Anarchie für KP-Spiele mit identischen Links ist $2 - \frac{2}{m+1}$ [6, Theorem 9].

Satz: Es existiert ein PTAS zur Approximation eines besten reinen NE [4, Corollary 1]. Gäbe es ein PTAS zur Approxmation des schlechtesten reinen NE, so ist $\mathsf{P} = \mathsf{NP}$ [6, Theorem 10].

Satz: Der reine Preis der Anarchie für nicht-identische (verwandte) Links ist $\Theta(\frac{\log m}{\log \log m})$.

Bemerkungen: Der (gemischte) Preis der Anarchie für nicht-identische Links ist $\Theta(\frac{\log m}{\log\log\log m})$ [3, Theorems 1.1, 1.3].

Satz: Für KP-Spiele gilt: $\mathsf{PC}_i(\mathsf{S}) \stackrel{\text{Def.}}{=} \sum_{\mathsf{s}\in S} \mathsf{S}(\mathsf{s}) \cdot f_{s_i}(l_{s_i}(\mathsf{s})) = \sum_{s_i\in S_i} \sigma_i(s_i) \cdot f_{s_i}(l_{s_i}(\mathsf{S}_{-i}, s_i))$.

Satz: Im KP-Spiel mit identischen links ist $\mathsf{F} = (\zeta_i)_{i=1}^n \in \Delta_{\perp\!\!\perp}(S)$ mit $\zeta_i(s_i) = \frac{1}{m}$ für alle $i \in [m]$, $s_i \in [m]$ das einzige vollständig gemischte (fully mixed) NE.

Bemerkungen: Das vollständig gemischte NE ist nicht immer das schlechteste NE im KP-Modell mit identischen Links [5].

Satz: Sei F das vollständig gemischte NE, S ein beliebiges (gemischtes) NE in einem KP-Spiel mit identischen Links. Dann gilt $\min_{s_i\in[m]} l_k(\mathsf{S}_{-i}, s_i) \le \min_{s_i\in[m]} l_k(\mathsf{F}_{-i}, s_i)$ [6, Lemma 4].

Folgerung: Voraussetzungen wie zuvor, s sei ferner beliebiges reines NE. Dann gilt $\mathsf{SC}(\mathsf{s}) \le \mathsf{SC}(\mathsf{F})$.

Satz: In einem KP-Spiel mit identischen Spielern und identischen Links gilt für das vollständig gemischte NE F: $\mathsf{SC}(F) = \Theta(\frac{\log m}{\log\log m})$.

Satz: Der (gemischte) Preis der Anarchie für identische Links ist $\Theta(\frac{\log m}{\log\log m})$ [3, Theorems 1.1, 1.3].

4. Auslastungsspiel II

Definition: Min Cost Flow: Gegeben gerichteter Graph $G = (V, E)$, Kosten c_e und Kapazität u_e für Kanten $e \in E$, Anforderung a_v für $v \in V$.

Lineares Programm MinCostFlow:

$$\begin{aligned}
\text{min.} \quad & \sum_{e\in E} c_e \cdot x_e \\
\text{gem.} \quad & \sum_{e\in E|e=(v,\cdot)} x_e + \sum_{e\in E|e=(\cdot,v)} x_e = a_v && \forall v \in V \\
& 0 \le x_e \le u_e && \forall e \in E
\end{aligned}$$

4.1. Komplexität der Berechnung reiner Nash-Equilibrien

Satz: Das Min Cost Flow-Problem kann in Polynomialzeit gelöst werden.

Satz: Für ein symmetrisches Netzwerk-Auslastungsspiel (mit beliebigen Latenzfunktionen) kann in Polynomialzeit ein reines NE mittels Reduktion auf das Min Cost Flow Problem berechnet werden.

4.2. PLS-Vollständigkeit

Definition: PLS-Problem $L = (D, F, c, N, A, B, C)$, wobei

- $D \subset \Sigma^*$ Menge der Probleminstanzen ($I \in D$ kann in Polynomialzeit entschieden werden),
- $F(I)$ Menge zulässiger Lösungen zu jedem $I \in D$,
- $c(s, I)$ für $I \in D$ und $s \in F(I)$ Kostenwert ist und $N(s, I) \subseteq F(I)$ die Nachbarschaft von s angibt.

Ferner sind A, B, C Algorithmen, wobei A zu $I \in D$ eine zulässige Lösung $s \in F(I)$ zurückgibt, B zu $s \in F(I)$ den Kostenwert $c(s, I)$ berechnet und C zu $I \in D$ und $s \in F(I)$ entweder ein besseres $s' \in N(s, I)$ zurückgibt oder anzeigt, dass es keine bessere Lösung in der Nachbarschaft gibt.

Definition: Ein Such-Problem ist eine Relation $R \subseteq \Sigma^* \times \Sigma^*$. Ein Algorithmus löst ein Such-Problem R, falls er zu Eingabe $x \in \Sigma^*$ ein $y \in \Sigma^*$ zurückgibt, so dass $(x, y) \in R$ oder er korrekt meldet, dass es ein solches y nicht gibt.

Definition: Ein Such-Problem R welches durch einen Polynomialzeitalgorithmus gelöst wird, liegt in P_S. Es liegt in NP_S, wenn es einen nicht-deteterministischen Algorithmus polynomieller Laufzeit gibt, der die Sprache $D_R = \{x \in \Sigma^* \mid \exists y \in \Sigma^* : (x, y) \in R\}$ akzeptiert und für $x \in D_R$ auch ein entsprechendes y ausgeben kann.

Definition: Sei L ein PLS-Problem wie zuvor. Das zu L gehörige Suchproblem R_L ist $R_L = \{(x, y) \in L \mid y \in F(x) \text{ ist lokal optimal}\}$. PLS ist die Klasse aller zu einem PLS-Problem gehörigen Suchprobleme.

PLS-Maximierungsproblem PosNae3Flip $= (D, F, c, N, A, B, C)$. Hier:

- $D = \{\langle\phi, w\rangle \mid \phi \text{ ist Boole'sche } (n, m)\text{-Formel in KNF mit Klauseln } C_i\text{, max. 3 positiven Literalen pro Klausel und positiven Klauselgewichten } w(C_i) > 0\}$
- $F(\phi) = \{0, 1\}^n$
- $c(s, \phi)$ ist Anzahl der Klauseln, in denen durch s nicht alle Literale gleich belegt sind
- $N(s, \phi) := \{s' \in F(\phi) \mid s \text{ und } s' \text{ unterscheiden sich in genau einem Bit}\}$

Bemerkungen: Es gibt passende Polynomialzeitalgorithmen A,B und C

Definition: Seien L, K PLS-Probleme. Wir sagen, L ist PLS-reduzierbar auf K, wenn es Polynomialzeitfunktionen $f : D_L \to K_L$ und $g_\phi : F(f(\phi)) \to F(\phi)$ gibt, so dass gilt: s lokales Optimum für $f(\phi) \in D_K \Rightarrow g_\phi(s)$ lokales Optimum für alle $\phi \in D_L$.

Definition: Ein PLS-Problem L ist PLS-vollständig, wenn jedes PLS-Problem auf L PLS-reduzierbar ist.

PLS-Problem NE in ungewichtetem CG finden, CGNash $= (D, F, c, N, A, B, C)$.

- $D = \{\langle \Gamma \rangle \mid \Gamma = (n, (1)_{i=1}^n, E, (f_e)_{e\in E}, (S_i)_{i=1}^n)\}$.
- $F(\phi) = \{\langle \mathsf{s} \rangle \mid \mathsf{s} \in S\}$
- $c(\mathsf{s}, \Gamma) = h_\Gamma(\mathsf{s})$, wobei $h_\Gamma : S \to \mathbb{R}$ eine feste zu Γ gehörige Potenzialfunktion sei.
- $N(\mathsf{s}, \Gamma) = \{(\mathsf{s}_{-i}, s'_i \mid i \in [n] \text{ und } s'_i \neq s_i\}$

Satz: PosNae3Flip ist PLS-vollständig.

Bemerkungen: MaxCut, GraphPartition, Max2Sat sind ebenfalls PLS-vollständig.

Satz: Die Berechnung eines reinen NE ein einem ungewichteten Auslastungsspiel ist PLS-vollständig (selbst dann, wenn die Anzahl Spiele auf 2 beschränkt ist und jede Ressource von maximal 3 Spielern genutzt werden kann).

Satz: Wenn $\mathsf{P}_S \neq \mathsf{PLS}$, kann ein NE zu einem Auslastungsspiel mit polynomiellen Latenzfunktionen von konstantem maximalem Grad nur dann in pseudopolynomieller Zeit bestimmt werden, wenn man zulässige Instanzen auf symmetrische ungewichtete Netzwerk-Auslastungsspiele beschränkt.

Definition: Der Standard-Algorithmus zu einem PLS-Problem (D, F, c, N, A, B, C) ist:
Eingabe: $I \in D$
Ausgabe: Lokales Optimum $s \in F(I)$

$s := A(x)$
while s nicht optimal **do**
  $s' := C(I, s)$
  if C war erfolgreich **then** $s := s'$

Definition: Das Standard-Algorithmen-Problem zu einem PLS-Problem wie zuvor ist: Bestimme zu Instanz $I \in D$ das lokale Optimum $s \in F(I)$, das der Standard-Algorithmus bei Eingabe I ausgeben würde.

Satz: Das Standard-Algorithmen-Problem zu den vier oben aufgeführten PLS-Problemen ist PSPACE-vollständig und benötigt im schlimmsten Fall exponentielle Zeit.

Lemma: Es gibt PLS-Probleme, deren Standard-Algorithmen-Problem NP-schwer ist (bspw. ein 3Sat-ähnliches PLS-Problem).

Lemma: $\mathsf{P}_S = \mathsf{NP}_S \Leftrightarrow \mathsf{P} = \mathsf{NP}$

Lemma: $\mathsf{P}_S \subseteq \mathsf{PLS} \subseteq \mathsf{NP}_S$

Lemma: Wenn ein PLS-Problem NP-schwer ist, so folgt $\mathsf{NP} = \mathsf{coNP}$.

4.3. Koordinationsfaktor

Definition: Bei Auslastungsspielen betrachten wir die Gesamtlatenz als soziale Kosten. Für $\mathsf{S} \in \Delta_{\perp\!\!\perp}(S)$ ist $\mathsf{SC}(\mathsf{S}) := \sum_{\mathsf{s}\in S} \mathsf{S}(\mathsf{s}) \sum_{e\in E} l_e(\mathsf{s}) \cdot f_e(l_e(\mathsf{s}))$.

Bemerkungen: Es gilt: $\mathsf{SC}(\mathsf{S}) = \sum_{i=1}^n w_i \cdot \mathsf{PC}_i(\mathsf{S})$.

Satz: Für Auslastungsspiele mit linearen Latenzfunktionen ist der Preis der Anarchie genau $\frac{5}{2}$. Im Fall symmetrischer Spiele (mit n Spielern) ist er $\frac{5n-2}{2n+1}$ [2, Theorems 1, 2, 3]. Durch Beschränkung auf reine NE verbessert sich der Preis der Anarchie jeweils nicht! Für Polynome allgemeinen Grads d siehe [unser Paper]. Asymptotisch: $d^{\Theta(d)}$.

Definition: Auslastungsspiel auf parallelen Kanten mit spieler-spezifischen Verzögerungsfunktionen (ASSV): Ähnlich wie KP-Spiel: n Spieler, m parallele Links, Stratiemengen $S_i = [m]$ für alle $i \in [n]$. Jedoch spieler-spezifischer Kostenfunktion $f_{i,k} : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$. $f_{i,k}(x)$ gibt Kosten von Spieler i auf Link k bei Gesamtlast x an [7].

Die Last auf einem Link k ist demnach (wie beim KP-Modell) $l_k(\mathsf{s}) = \sum_{i\in[n]|s_i=k} w_i$. Die privaten Kosten sind $\mathsf{PC}_i(\mathsf{s}) = f_{i,s_i}(l_{s_i}(\mathsf{s}))$.

Lemma: Bei einem ASSV mit $m = 2$ Links führt jede Folge von selfish steps in ein NE. Dies ist nicht der Fall bei $m = 3$.

Satz: Jedes ASSV mit identischen Spielergewichten besitzt ein reines NE.

Satz: Jedes ASSV mit $n = 2$ Spieler besitzt ein reines NE. Dies gilt nicht im Fall $n = 3$.

Satz: In einem ASSV mit identischen Spielergewichten existiert von jedem $\mathsf{s} \in S$ eine Folge von (in n und m polynomiell vielen) greedy selfish steps, die in ein reines NE führen.

5. Wardrop Modell

Anmerkung: Um die Notation nicht zu weit von der aus der Vorlesung zu entfernen, bezeichnen wir im Folgenden Latenzfunktionen mit ℓ, und Routings bzw. Flüsse (die eigtl. eher vergleichbar mit einem Strategieprofil sind) mit f.

Definition: Wardrop-Spiel $\Gamma = (n, V, E, (\ell_e)_{e\in E}, (a_i, o_i, d_i)_{i=1}^n)$, wobei n Anzahl Routing-Anforderungen, (V, E) ein gerichteter Graph und $f_e : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ stetige nicht-fallende Latenzfunktionen. Ferner: a_i ist ein Fluss (Bedarf), der von $o_i \in V$ nach $d_i \in V$ geroutet werden muss.

$\mathcal{P}_i := \{P \mid P \text{ ist Pfad von } o_i \text{ nach } d_i\}$. $\mathcal{P} := \bigcup_{i\in[n]} \mathcal{P}_i$.

Definition: Ein Routing (oder Pfadfluss) ist ein nicht-negativer Vektor $f := (f_P)_{P\in\mathcal{P}}$. Dabei heißt f zulässig, wenn es den Bedarf erfüllt: $\forall i \in [n] : \sum_{P\in\mathcal{P}_i} f_P = a_i$. Es bezeichne S die Menge aller zulässigen Routings.

Die Last auf einer Kante ist $l_e(f) := \sum_{P\in\mathcal{P}|P\ni e} f_P$. Abkürzend: $f_e := l_e(f)$. Reisezeit entlang eines Pfades ist $\ell_P(f) := \sum_{e\in P} \ell_e(l_e(f)) = \sum_{e\in P} \ell_e(f_e)$.

Definition: Die (sozialen) Kosten eines Routings sind definiert als Gesamtreisezeit: $\mathsf{SC}(f) := \sum_{P\in\mathcal{P}} \ell_P(f) f_P = \sum_{e\in E} \ell_e(f_e) f_e$.

Nicht-lineares Optimierungsprogramm für das Systemoptimum:

$$\begin{aligned} \text{max.} \quad & \sum_{e\in E} \ell_e(f_e) f_e && (1) \\ \text{gem.} \quad & \sum_{P\in\mathcal{P}|P\ni e} f_P = f_e \quad \forall e \in E \\ & \sum_{P\in\mathcal{P}_i} f_P = a_i \quad \forall i \in [n] \\ & f_P \geq 0 \quad \forall P \in \mathcal{P} \end{aligned}$$

5.1. Existenz von Nash-Equilibrien

Definition: $f \in S$ ist NE, wenn für alle $i \in [n]$, alle $P, Q \in \mathcal{P}_i$, und alle $\epsilon \in [0, f_P]$ gilt, dass $\ell_P(f) \leq \ell_Q(f^\epsilon)$, wobei $f^\epsilon = (f_R)_{R\in\mathcal{P}}$,

$$f_R^\epsilon := \begin{cases} f_P - \epsilon & \text{für } R = P \\ f_Q + \epsilon & \text{für } R = Q \\ f_R & \text{else.} \end{cases}$$

Satz: $f \in S$ ist NE genau dann, wenn für alle $i \in [n]$ und alle $P \in \mathcal{P}_i$ gilt, dass $f_P > 0 \Rightarrow \ell_P(f) \leq \min_{Q\in\mathcal{P}_i}\{\ell_Q(f)\}$ [9].

Folgerung: Bezeichnet $L_i(f)$ die (eindeutige) Latenz aller benutzten Pfade von o_i nach d_i, so gilt: $\mathsf{SC}(f) = \sum_{i=1}^n L_i(f) a_i$.

5.2. Berechnung von Nash-Equilibrien

Satz: Für alle Latenzfunktionen sei $\ell_e \cdot \mathrm{id}$ konvex. Ein Fluss ist optimal genau dann, wenn für alle $i \in [n]$ und alle $P \in \mathcal{P}_i$ gilt, dass $f_P > 0 \Rightarrow \ell_P^*(f) \leq \min_{Q\in\mathcal{P}_i}\{\ell_Q^*(f)\}$. Hierbei ist $\ell_P^* := (\ell_P \cdot \mathrm{id})' = \ell_P + \ell_P' \cdot \mathrm{id}$ [1].

Folgerung: Ein Systemoptimum eines Wardrop-Spiels mit Latenzfunktionen, für die $\ell_e \cdot \mathrm{id}$ jeweils konvex ist, kann als NE der gleichen Instanz, jedoch mit Latenzfunktionen ℓ_e^*, aufgefasst werden.

Folgerung: Im Falle von Latenzfunktionen, wie beschrieben, existiert immer ein NE. Die sozialen Kosten müssen dabei für alle NE gleich sein.

5.3. Koordinationsfaktor

Satz: Es gelte für ein $\alpha \geq 1$, dass $x \cdot \ell_e(x) \leq \alpha \int_0^x \ell_e(t)\,dt$ für alle $e \in E$ und alle $x \geq 0$. Dann ist α eine obere Schranke für $\frac{\mathsf{SC}(f)}{\mathsf{SC}(f^*)}$, wenn f ein beliebiges NE und f^* ein Systemoptimum ist.

Folgerung: Der Preis der Anarchie für Wardrop-Spiele mit Latenzfunktionen vom Grad maximal d ist von oben beschränkt durch $d + 1$.

Bemerkungen: Der exakte Wert [8, Proposition 5.1] ist:

$$\frac{(d+1)\sqrt[d]{d+1}}{(d+1)\sqrt[d]{d+1} - d}$$

Literatur

[1] M. Beckmann, C. B. McGuire, and C. B. Winsten. *Studies in the Economics of Transportation.* Yale University Press, 1956. 5

[2] G. Christodoulou and E. Koutsoupias. The price of anarchy of finite congestion games. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing (STOC'05)*, pages 67–73. ACM Press, 2005. URL http://doi.acm.org/10.1145/1060590.1060600. 4

[3] A. Czumaj and B. Vöcking. Tight bounds for worst-case equilibria. In *Proceedings of the thirteenth annual ACM-SIAM symposium on Discrete algorithms (SODA'02)*, pages 413–420, Philadelphia, PA, USA, 2002. Society for Industrial and Applied Mathematics. URL http://www-i1.informatik.rwth-aachen.de/~voecking/publications/SODA02.pdf. 3

[4] R. Feldmann, M. Gairing, T. Lücking, B. Monien, and M. Rode. Nashification and the coordination ratio for a selfish routing game. In *Proceedings of the 30th International Colloquium on Automata, Languages, and Programming (ICALP'03)*, volume 2719, pages 514–526, 2003. URL http://www.springerlink.com/link.asp?id=9nee54hyetwkr4r2. 2

[5] S. Fischer and B. Vöcking. A counterexample to the fully mixed nash equilibrium conjecture. Technical report, RWTH Aachen, 2005. URL http://www-i1.informatik.rwth-aachen.de/~fischer/publications/Fischer_Voecking_FMNE_05.pdf. 3

[6] M. Gairing, T. Lücking, M. Mavronicolas, B. Monien, and P. Spirakis. Extreme nash equilibria. In *Proceedings of the 8th Italian Conference on Theoretical Computer Science (ICTCS'03)*, volume 2841, pages 1–20, 2003. URL http://springerlink.metapress.com/link.asp?id=n54at3ubuhbcb9qv. 2, 3

[7] I. Milchtaich. Congestion games with player-specific payoff functions. *Games and Economic Behavior*, 13(1):111–124, 1996. URL http://dx.doi.org/10.1006/game.1996.0027. 4

[8] T. Roughgarden. The price of anarchy is independent of the network topology. *Journal of Computer and System Sciences*, 67(2):341–364, 2003. URL http://dx.doi.org/10.1016/S0022-0000(03)00044-8. 5

[9] J. G. Wardrop. Some theoretical aspects of road traffic research. In *Proceedings of the Institution of Civil Engineers*, volume 1, pages 325–378, 1952. 5