

Seminar über Algorithmen

Kombinatorische Algorithmen zur Berechnung von Marktequilibria

12.11.2013, Sebastian Stugk

Übersicht

1. Marktmodelle und Gleichgewichtsdefinition
2. Das Eisenberg-Gale-Programm (EG)
3. Kombinatorische Bestimmung des Optimums von EG
4. Schlussbetrachtung



Marktmodelle und Gleichgewichtsdefinition

Fisher-Märkte

Definition:

Ein Fisher-Markt besteht aus einer Menge von unterscheidbaren und teilbaren Produkten A , einer Menge von Käufern B . Jeder der Käufer hat ein Startguthaben in Höhe von e_i .

Jedes Produkt ist eine Stückzahl in Höhe von b_i verfügbar.

Jeder Käufer versucht eine konkave Nutzfunktion u_i zu maximieren.

Gleichgewicht in Fisher-Märkten

Definition:

Ein Marktgleichgewicht ist ein Preisvektor $p = (p_1, \dots, p_n)^\top \in \mathbb{R}_+^n$, sodass für jeden Käufer eine Produktmenge $x_i^* = (x_{i1}^*, \dots, x_{in}^*)^\top \in \mathbb{R}_+^n$ existiert und die folgenden beiden Bedingungen erfüllt sind:

1. x_i^* maximiert $u_i(x)$ mit Rücksicht auf Budgetbeschränkung $p \cdot x \leq e_i$
2. Für jedes Produkt j gilt: $\sum_{i=1}^m x_{ij}^* = b_j$

Unsere Einschränkung:

Die Nutzenfunktion ist linear: $u_i(x) = \sum_{j=1}^n u_{ij} x_{ij}$

Das Eisenberg-Gale-Programm (EG)

Das primale Programm (EG)

$$\begin{aligned} \max \quad & \sum_{i=1}^m e_i \log u_i \\ \text{mit} \quad & u_i = \sum_{j=1}^n u_{ij} x_{ij} \quad \forall i \in B \\ \text{unter Nb.} \quad & \sum_{i=1}^m x_{ij} \leq 1 \quad \forall j \in A \\ \text{und Vzb.} \quad & x_{ij} \geq 0 \quad \forall i \in B, \forall j \in B \end{aligned}$$

Wir setzen voraus, dass zu jedem Produkt j ein potenzieller Käufer existiert, also $u_{ij} > 0$.

Das duale Lagrange-Programm (EG_D)

$$\max_p \quad L(x, p) = - \sum_{i=1}^m e_i \log u_i + \sum_{j=1}^n p_j f_j(x)$$

$$\text{mit} \quad u_i = \sum_{j=1}^n u_{ij} x_{ij} \quad \forall i \in B$$

$$f_j(x) = \left(\sum_{i=1}^m x_{ij} \right) - 1 \quad \forall j \in A$$

$$\text{unter Vzb.} \quad p_i \geq 0$$

Karush-Kuhn-Tucker-Bedingungen

Hinreichende Bedingungen für primal und dual optimale Lösungen von konvexen Programmen (bei starker Dualität)

Für EG und EG_D lässt sich aus ihnen ableiten:

$$(i) \quad \forall j \in A : p_j \geq 0$$

$$(ii) \quad \forall j \in A : p_j > 0 \Rightarrow \sum_{i=1}^m x_{ij} = 1$$

$$(iii) \quad \forall i \in B \quad \forall j \in A : \frac{u_{ij}}{p_j} \leq \frac{\sum_{j \in A} u_{ij} x_{ij}}{e_i}$$

$$(iv) \quad \forall i \in B \quad \forall j \in A : x_{ij} > 0 \Rightarrow \frac{u_{ij}}{p_j} = \frac{\sum_{j \in A} u_{ij} x_{ij}}{e_i}$$

Marktgleichgewicht als Lösung von EG und EG_D

Aus diesen 4 Bedingungen folgt:

- Alle Preise sind positiv und alle Güter können verkauft werden.
- Jeder Käufer erhält optimale Menge von Gütern.
- Das Guthaben aller Käufer wird vollständig aufgebraucht.

Grundidee des Algorithmus

Abschwächung von Bedingung (iii) und (iv) zu:

$$(iii') \quad \forall i \in B, \forall j \in A : \frac{u_{ij}}{p_j} \leq \frac{\sum_{j \in A} u_{ij} x_{ij}}{m_i}$$

$$(iv') \quad \forall i \in B, \forall j \in A : x_{ij} > 0 \Rightarrow \frac{u_{ij}}{p_j} = \frac{\sum_{j \in A} u_{ij} x_{ij}}{m_i}$$

Kontinuierliche Verringerung der Verletzung der Nebenbedingungen bis alle KKT-Bedingungen erfüllt sind.

Konkrete Berechnung des Optimums von EG und EG_D durch Berechnung von maximalen Netzwerkflüssen.

Equality subgraph

Definition: Maximalnutzen pro Einheit Geld

Sei der Preisvektor p gegeben. Der Maximalnutzen pro Einheit Geld ist eines Käufers i ist:

$$\alpha_i = \max_j \left\{ \frac{u_{ij}}{p_j} \right\}$$

Definition: Equality subgraph

Seien A die Menge der Produkte, B die Menge der Käufer sowie α_i für $1 \leq i \leq m$. Der Equality subgraph $G = (V, E)$ ist definiert durch:

$$V = A \cup B$$

$$E = \left\{ (a, b) \mid \frac{u_{ij}}{p_j} = \alpha_i \right\}$$

Flussnetzwerk $N(p)$

Definition:

Basierend auf dem Equality subgraph definieren wir das Flussnetzwerk $N(p)$ wie folgt:

$$V = \{s\} \cup A \cup B \cup \{t\} \quad \text{mit Quelle } s \text{ und Senke } t$$

$$E = \{(s, b) \mid a \in A\} \cup \{(a_j, b_i) \mid \frac{u_{ij}}{p_j} = \alpha_i\} \cup \{(b, t) \mid b \in B\}$$

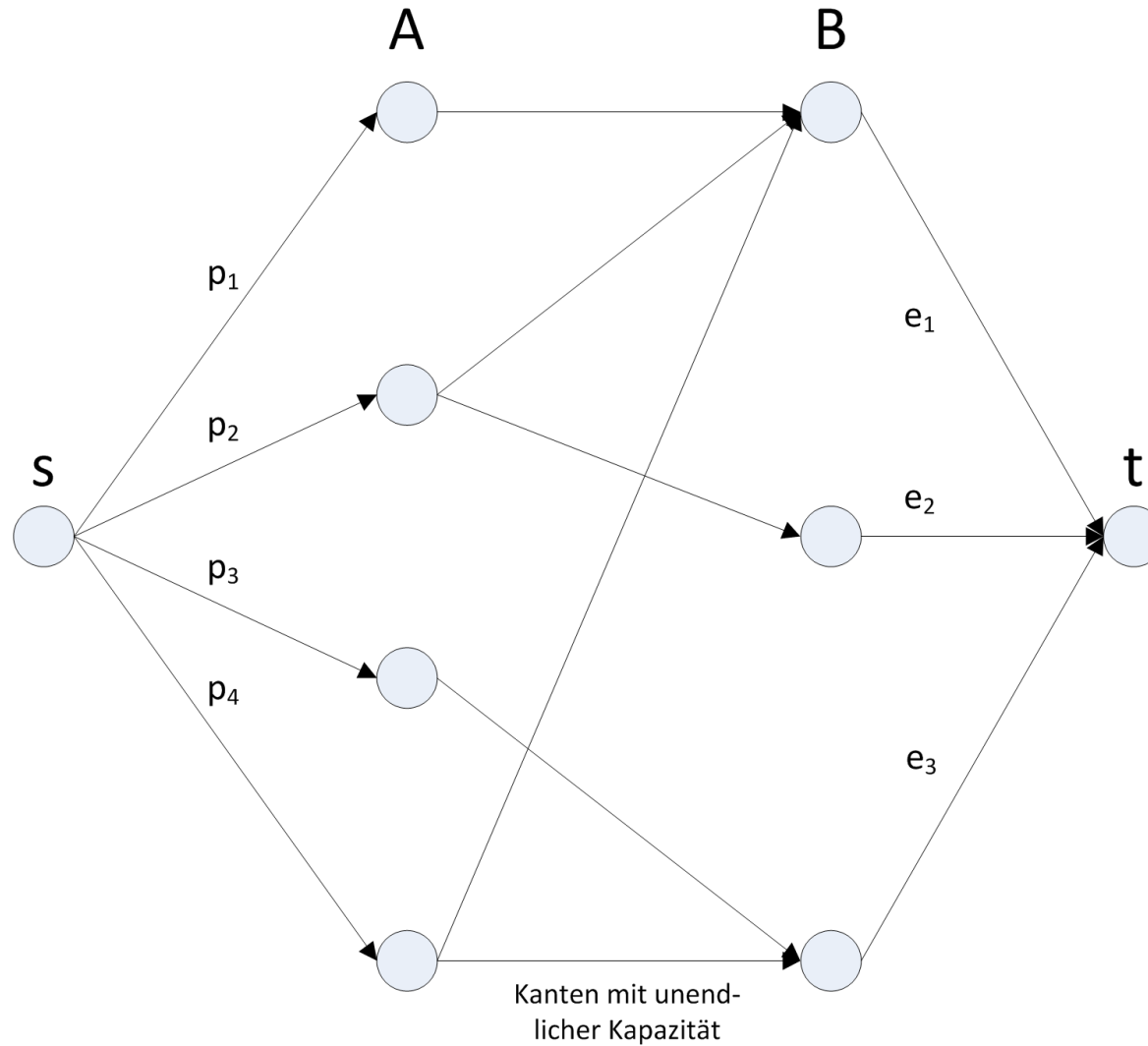
Kapazitäten:

$$c(s, a_j) = p_j \quad \text{für } a_j \in A$$

$$c(b_i, t) = e_i \quad \text{für } b_i \in B$$

$$c(a_j, b_i) = \infty \quad \text{für } (a_j, b_i) \in \{(a_j, b_i) \mid \frac{u_{ij}}{p_j} = \alpha_i\}$$

Flussnetzwerk $N(p)$



Invariante

"Jeder Preisvektor p gewährleistet, dass der Schnitt $(\{s\}, A \cup B \cup \{t\})$ minimal in $N(p)$ ist."

Sie stellt sicher, dass Gleichgewichtspreise zu keinem Zeitpunkt überschritten werden.

Käufer können jedoch einen Überschuss haben.

Balancierte Flüsse: Definitionen

Definition: Überschuss

Seien ein Flussnetzwerk N gegeben, dass die Invariante erfüllt. Sei weiterhin f ein Fluss in diesem Netzwerk und $R(f)$ der Restgraph bezüglich dieses Flusses.

- Der Überschuss von Käufer i , bezeichnet als $\gamma_i(N, f)$ ist die Restkapazität der Kante (i, t) mit Rücksicht auf f .
- Der Überschussvektor bezüglich N ist $\gamma(N, f) := (\gamma_1(N, f), \dots, \gamma_n(N, f))$.

Definition: Balancierter Fluss

Ein balancierter Fluss in N ist ein Fluss, der die L2-Norm $\|\gamma(N, f)\|$ des Überschussvektors minimiert.

Balancierte Flüsse: Algorithmus

Gegeben: Flussnetzwerk N , sodass Invariante erfüllt ist

1. Reduziere Kapazitäten aller Kanten von B nach t bis die Kapazität des Schnittes $(\{s\} \cup A \cup B, \{t\})$ gleich der Kapazität des Schnittes $(\{s\}, A \cup B \cup \{t\})$ ist. Sei N' das resultierende Netzwerk und f' ein maximaler Fluss in N' .
2. Finde einen maximalen min-s-t-cut (S, T) mit $s \in S$ und $t \in T$ in N' .
 - Fall 1: $T = \{t\}$
Finde maximalen Fluss in N' und gib ihn aus.
 - Fall 2: Sonst.
Finde rekursiv balancierte Flüsse f_1 und f_2 in den Teilnetzwerken
 - $N_1 \dots$ mit Knoten nur aus $S \cup \{t\}$
 - $N_2 \dots$ mit Knoten nur aus $T \cup \{s\}$Gib $f = f_1 \cup f_2$ zurück.

Balancierte Flüsse: Algorithmus

Satz:

Der vorgestellte Algorithmus berechnet einen balancierten Fluss in N .

Beweisskizze:

Sei der berechnete Fluss f gegeben.

f ist maximaler Fluss

- Bei Terminierung in Fall 1:
Schnitt 2 ist minimaler Schnitt mit gleicher Kapazität wie Schnitt 1.
- Bei Terminierung in Fall 2:
 f sättigt alle Kanten von s nach A in N und muss daher maximal sein.

f ist balanciert

- Kann durch Induktion über Tiefe der Rekursion gezeigt werden.

Nachbarschaften & feste Mengen

Definition: Nachbarschaft

Sei das Flussnetzwerk $N(p)$ gegeben und sei $S \subseteq A$. Die Nachbarschaft von S in $N(p)$ ist:

$$\Gamma(S) = \{j \in B \mid \exists i \in S \text{ mit } (i, j) \in N(p)\}$$

Definition: Feste Menge

Sei $N(p)$ gegeben. Bezeichne außerdem $p(S)$ den Gesamtwert der Produkte in S (die Summe ihrer Preise) und bezeichne analog $m(T)$ das Gesamtvermögen einer Teilmenge $T \subseteq B$ von Käufern. S wird als feste Menge bezeichnet, gdw. $p(S) = m(\Gamma(\bar{S}))$.



Hauptalgorithmus

Initialisierung des Preisvektors:

- Setze $p_i = \frac{1}{n}$ für $1 \leq i \leq m$.
- Berechne α_i für $1 \leq i \leq n$ und das daraus resultierende Netzwerk $N(p)$.
- Reduziere den Preis von Produkt j auf $p_j = \max_i \left\{ \frac{u_{ij}}{\alpha_i} \right\}$, falls es keine inzidente Kante in $N(p)$ hat.

Lauf des Algorithmus ist in Phasen unterteilt.

- Am Ende jeder Phase wird eine neue Menge fest.
- Jede Phase beinhaltet eine Reihe von Iterationen zur Preiserhöhung.

Hauptalgorithmus: Ablauf einer Phase

1. Berechnung eines balancierten Flusses f im aktuellen Netzwerk $N(p)$.
2. Falls der Flussalgorithmus in Fall 1 endet:
Gib den Preisvektor p und die Produktverteilung x zurück.
Sonst: Fahre fort.
3. Iterative Preiserhöhung

Hauptalgorithmus: Ablauf einer Iteration

Setze:

- δ = maximaler Überschuss der Käufer mit Rücksicht auf f
- I = Teilmenge der Käufer B mit Überschuss δ
- $J \subseteq A$ Produkte mit Kanten zu I in $N(p)$

Beginne mit $\Delta_0 = 1$ und erhöhe in jeder Iteration Δ_i um einen kleinen Wert bis für $p_i = \Delta_i \cdot p$ und $N(p_i)$ eines der beiden folgenden Ereignisse eintritt:

(a) Eine Teilmenge $S \subseteq J$ wird fest:

- Fahre mit der nächsten Phase fort.

(b) Neue Kante (i,j) mit $i \in I$ und $j \in A \setminus J$ tritt in Eq. subgraph ein:

- Füge Kante (i,j) zu $N(p)$ hinzu berechne einen balancierten Fluss f .
- Falls Flussalgorithmus in Fall 1 terminiert:

Gib Preisvektor p und Produktverteilung x zurück.

- Sonst:

Bestimme Menge B' aller Käufer mit gerichtetem Pfad zu Käufern aus I .

Setze $I := B'$ und $J = \{j \mid j \in A; j \text{ hat Kante zu } i \in I \text{ in } N(p)\}$ und fahre mit Iteration $i+1$ fort.

Laufzeit

Dieser Algorithmus berechnet ein Marktgleichgewicht in

$$\mathcal{O}(n^4(\log n + n \log U + \log M))$$

Berechnungen von maximalen Netzwerkflüssen mit

$$U = \max_{i \in B, j \in A} \{u_{ij}\}$$

($M = \text{Summe der Gelder aller Käufer}$)

Weitere Marktmodelle

Allgemeiner: Arrow-Debreu-Tauschökonomien

- Marktteilnehmer sind allgemeine "Händler"
- Ziel: Preise finden, sodass jeder Händler
 - seine Waren vollständig verkauft
 - eine bzgl. der Nutzenfunktion optimale Warenmenge erhält

Spezieller: Ressourcenverteilungsmärkte

- Gegeben
 - Menge von Ressourcen R mit verfügbaren Kapazitäten $c : R \rightarrow \mathbb{Z}^+$
 - Menge von Marktteilnehmern $A = \{a_1, \dots, a_n\}$
 - Jeder Marktteilnehmern i verfügt über Geldmenge $m_i \in \mathbb{Z}^+$.
- Jeder von ihnen
 - kann bestimmte Objekte $V \subseteq \mathcal{P}(R)$ produzieren.
 - versucht so viele Objekte aus V wie möglich herzustellen.



Fragen und Diskussion



Danke.