Seminar über Algorithmen	3. Dezember 2013
Combinatorial Auctions	
Olaf Parczyk	*Wolfgang Mulzer, Yannik Stein*

1 Einleitung und Begriffe

Gegenüber normalen Spielen oder Versteigerungen haben bei combinatorischen Auktionen n Spieler die Möglichkeit auf m nicht zerteilbare Objekte zu bieten. Jeder ist dabei an bestimmten kombinationen von Objekten interessiert und bietet auf Teilmengen der m Objekte. Formal hat jeder Spieler i eine **Bewertungsfunktion** v_i, die jeder Menge S eine Wert $v(S)$ zuweist, die dieser Spieler für die Menge bereit ist zu zahlen. Ein soche Funktion soll immer Monoton sein, also für $S \subseteq T$ gilt $v(S) \leq v(T)$ und $v(\emptyset) = 0$.

Unser Ziel als Auktionator ist eine **Aufteilung** zu finden, also den n Spielern disjunkte Teilmengen von Objekte $S_1, \ldots, S_n$ für Preise $p_1, \ldots, p_n$ zu verkaufen und den **sozialen Nutzen** $\sum_i v_i(S_i)$ zu maximieren. Dabei gibt es verschiedene Szenarien, in denen entweder die Bewertungsfunktion dem Auktionator bekannt ist, oder in mehreren Runden gebote eingeholt werden und die v_i von jedem Spieler geheim ist.

Ein einzelner Spieler will seinen **Gewinn** $v_i(S_i) - p_i$ maximieren. Optimalerweise finden wir das Maximum des sozialen Nutzen in einem Gleichgewichtspunkt, dass heißt jeder Spieler der seine Strategie ändert schneidet höchstens genauso gut ab.

Es treten verschiedene Schwierigkeiten auf, um die wir uns kümmern müssen:

- *Komplexität*: Das Problem eine Aufteilung zu finden is NP-vollständig, sogar für Spezialfälle.
- *Dasrstellung und Kommunikation*: Die Bewertungsfunktionen sind exponentiell groß in m und können also weder vollständig gespeichert noch übertragen werden.
- *Strategien*: Wie sehen Strategien für die Spieler aus?

Beispiele für solche Auktionen entstehen, wenn die Bieter nur einen Nutzen aus den Objekten ziehen können, wenn diese in bestimmten kombinationen vorliegen. Telefonkonzerne in den vereinigten Staaten erwerben in solchen Auktionen die Lizenzen für bestimmte Wellenlängen in bestimmten Gebieten.

Im folgenden wollen wir uns den drei genannten Probleme witmen. Dabei lassen wir zun"chst das Problem der Darstellung außer acht indem wir us auf simple Bewertungsfunktionen beschränken. Danach schauen wir uns genauer an wie ein Gleichgewicht aussehen kann und im letzten Teil gehen wir auf verschiedene Möglichkeiten ein die Bewertung darzustellen und zu übertragen.

2 Simple Bewertungen

2.1 Komplexität

Eine **simple** Bewertungsfunktion ist ein Paar (S^*, v^*) mit $v(S) = v^*$ für alle $S \supseteq S^*$ und $v(S) = 0$ sonst. In einer Aufteilung bekommt jeder Spieler i dann entweder seine gewünschte Menge S_i^* oder garnichts. Die Lösung des Problems ist also eine Menge von Gewinnern W mit disjunkten S_i^* und maximalem $\sum_{i \in W} v_i^*$.

Proposition 1. *Das* Aufteilugnsproblem *ist NP-vollständig.*

Beweisskizze. Reduktion vom *Problem eine Unabh"ngige Menge in einem Graphen zu finden*, wobei die Spieler die Knoten sind und die Objekte die Kanten. □

Sogar eine Approximation innerhalb eines Faktors von $m^{\frac{1}{2}-\varepsilon}$ ist NP-hart, wir sehen später eine $m^{\frac{1}{2}}$ Approximation. Neben verschiedenen Spezialfällen, die sich effizient lösen lassen, kann man das Problem auch als lineares Programm formulieren und somit für die meissten natürlichen Eingaben optimale Ergebnisse ermitteln.

2.2 Ein approximativer Mechanismus

Nun nehmen wir wieder an, dass die Bewertungsfunktion eine private Information ist, wollen aber einen Mechanismus konstruieren, bei dem jeder Spieler bestmöglich abschneidet, wenn er seine private Information wahrheitsgemäß weitergibt.

Sei V_s die Menge aller simplen Bewertungsfunktionen auf m Objekte und A alle möglichen Aufteilungen auf die n Spieler. Ein Mechanismus besteht jetzt aus einer Funktion $f : (V_s)^n \to A$ und einer Auszahlungsfunktion $p_i : (V_s)^n \to \mathbb{R}$ für $i = 1, \ldots, n$. Der Mechanismus ist effizient, wenn f und p_i in polynomieller Zeit berechnet werden können. Außerdem fordern wir für jedes i und $v_1, \ldots, v_n, v_i' \in V_s$, dass

$$v_i(a) - p(v_1, \ldots, v_i, \ldots, v_n) \geq (v_i(a') - p(v_1, \ldots, v_i', \ldots, v_n)),$$

wobei $a = f(v_1, \ldots, v_i, \ldots, v_n)$, $a' = f(v_1, \ldots, v_i', \ldots, v_n)$ und $v_i(a) = v_i$ wenn i in a gewinnt und Null andernfalls. Dies Forderung nennt man **incentive compatible**.

Wir schauen uns eine greedy Mechanismus an: Zuerst werden alle Gebote sortiert, so dass

$$\frac{v_1^*}{\sqrt{|S_1^*|}} \geq \frac{v_2^*}{\sqrt{|S_2^*|}} \geq \cdots \geq \frac{v_n^*}{\sqrt{|S_n^*|}}$$

und wir starten mit einer leeren Gewinnermenge $W = \emptyset$. Von $i = 1$ bis n wird jetzt geschaut, ob Gebot i noch erfüllt werden kann, also ob $S_i^* \cap (\cup_{j \in W} S_j^*) = \emptyset$ and then $W := W \cup \{i\}$. Am Ende wird die Menge der Gewinner W und die Preise

$$p_i = \frac{v_j^*}{\sqrt{|S_j^*|/|S_i^*|}}$$

ausgegeben, wobei j der kleinste Index is, so dass $S_i^* \cap S_j^* \neq \emptyset$ und für alle $k < j$, $k \neq i$, $S_k^* \cap S_j^* = \emptyset$ gilt, andernfalls $p_i = 0$.

Offensichtlich ist der Mechanismus effizient und erzeugt eine zulässige Menge von Gewinnern. Folgendes Theorem folgt direkt aus den beiden nachstehenden Lemmata.

Satz 2. *Der greedy Mechanismus berechnet effizient eine incentive compatible $\sqrt{m}$-Approximation des optimalen Nutzens.*

Lemma 3. *Ein Mechanisms ist incentive compatible, falls:*

- Monotonität*: Jedes Gebot* (S_i^*, v_i^*) *welches gewinnt, gewinnt auch für jedes* $v_i' > v_i^*$ *und für jedes* $S_i' \subset S_i^*$.
- Kritische Bezahlung*: Wenn ein Gebot gewinnt bezahlt es den minimal möglichen Betrag mit dem es gewinnt.*

Beweis. □

Lemma 4. *Sei* OPT *eine Verteilung mit maximalem Nutzen* $\sum_{i \in OPT} v_i^*$ *und* W *die Ausgabe des Mechanismus, dann* $\sum_{i \in OPT} v_i^* \leq \sqrt{m} \sum_{i \in W} v_i^*$.

Beweis. □

3 Lineares Programm und Gleichgewicht

3.1 Relaxation des linearen Programms

Wir kehren nun zu Allgemeinen Bewertungsfunktionen zurück und betrachten die Relaxation des ganzzahligen linearen Programms und den Zusammenhang mit dem Walrasischen Gleichgewicht.

Relaxation des linearen Programms (mit variablen $x_{i,S}$ für alle $i \in N$ und $S \subseteq M$):

$$\begin{aligned} \textbf{Maximiere} \quad & \sum_{i \in N, S \subseteq M} x_{i,S} v_i(S) & \\ \textbf{mit Nebenbedingungen} \quad & \sum_{i \in N, S \text{ mit } j \in S} x_{i,S} \leq 1 & \forall j \in M \\ & \sum_{S \subseteq m} x_{i,S} \leq 1 & \forall i \in N \\ & x_{i,S} \geq 0 & \forall i \in N, S \subseteq M \end{aligned}$$

Vor der Relaxation war $x_{i,S} \in \{0,1\}$ je nachdem ob Spieler i die Menge S erhält oder nicht. Bei simplen Bewertungen wird nur eine Variable pro Spieler benötigt und man bekommt so direkt eine effiziente Lösung. Sogar im allgemeinen Fall lässt sich das Programm in polynomieller Zeit (in n,m und der Anzahl von Bits der Genauigkeit t) bestimmen. Dabei hilft eine geschickte Anwendunge der ellipsoid Method, die Lineare Programme in polynomieller Zeit lösen kann. Das Problem ist die Anzahl der Nebenbedingungen, die exponentiell in m sein kann. Mit Hilfe einer Vereinfachung des dualen linearen Programms lässt sich dieses umgehen.

3.2 Walrasisches Gleichgewicht

Für ein Spielergebot v_i und einzelne Objektpreise $p_1, \ldots, p_m$ ist eine Menge T ein **Bedarf** von Spieler i, wenn für jede andere Menge $S \subseteq M$ gilt $v_i(S) - \sum_{j \in S} p_j \leq v_i(T) - \sum_{j \in T} p_j$. Ein **Walrasisches Gleichgewicht** ist dann eine Aufteilung $S_1^*, \ldots, S_n^*$ mit Preisen $p_1^*, \ldots, p_m^*$, falls für jeden Spieler i die Menge S_i^* ein Bedarf ist zu den gegeben Preisen und für alle nicht zugewiesenen Objekte $p_j = 0$ gilt.

Satz 5. *Sei* $p_1^*, \ldots, p_m^*$ *und* $S_1^*, \ldots, S_n^*$ *ein Walrasisches Gleichgewicht, dann maximiert diese Zuweisung den Nutzen. Desweiteren maximiert es den Nutzen auch über alle nicht ganzzahligen Zuweisungen.*

Inbesondere ist also ein Walrasicehs Gleichgewicht eine ganzzahlige Lösung zu der Relaxation des linearen Programms.

Beweis. □

[Beispiel einfügen für Walrasisches Glecihgewicht?]

Andersherum gilt aber auch:

Satz 6. *Wenn eine ganzzahlige Lösung für die Relaxation des linearen Programmes existiert, dann gibt es auch ein Walraisches Gleichgewicht, dessen Zuweisung die gefundene Lösung ist.*

Beweis. □

Zusammenfassend lässt sich also sagen:

Folgerung 7. *Ein Walrasisches Gleichgewicht existiert genau dann, wenn die Relaxation des zugehörigen linearen Programms eine ganzahlige Lösung besitzt.*

4 Gebote abgeben

Im letzten Abschnitt soll es darum gehen, wie die Gebote der Spieler dargestellt und übertragen werden können ohne den Wert für alle 2^m Teilmengen festzulegen. Dafür gibt es zwei Grundverschiedene Ansätze. Entweder werdem in einem Schritt die Gebote abgegeben und der Auktionator berechnet daraus eine Aufteilung, oder es wird iterativ immer wieder vom Auktionator nachgefragt und dann die Lösung ermittelt.

Bei der ersten Variante wird aus den simplen Bewertungen mithilfe von Verknüpfungen komplexere Bewertungefunktionen erstellt. Aus Bewertungen v und u entstehen neue $(vXORu)$ und $(vORu)$ definiert durch:

- $(vXORu)(S) = max(v(S), u(S))$
- $(vORu)(S) = max_{R,T \subseteq S, R \cap T = \emptyset}(v(R) + v(T))$.

[downward slopping + Lemma?]

Bei der zweiten Variante sollen immer nur relevante Informationen für den aktuellen Berechnungsschritt übertragen werden. Wir betrachten zwei mögliche Anfragen vom Auktionator an einen Spieler:

- *Wertanfragen*: Der Auktionator fragt nach dem Wert $v(S)$ für eine bestimmte Menge S.
- *Bedarfsanfrage*: Der Auktionator nennt Preise $p_1, \ldots, p_m$ für die Objekte und der Spieler antwortet mit einem Bedarf T, also eine Menge, die $v(T) - \sum_{i \in T} p_i$ maximiert.

Lemma 8. *Eine Wertanfrage läst sich mit mt Bedarfsanfragen simulieren, wobei t die Bitlänge der Genauigkeit in der Darstellung des Wertes der Menge ist.*

Beweis. □

Lemma 9. *Umgekehrt kann es aber sein, dass exponentiell viele Wertanfragen nötig sind um eine Bedarfsanfrage zu simulieren.*