

経営経済のための 最適化理論特講

複数財オークションのアルゴリズムと 離散最適化

第12回 均衡を近似的に求めるアルゴリズム

塩浦昭義

東京工業大学 経営工学系

shioura.a.aa@m.titech.ac.jp

均衡配分の計算：評価値が既知の場合

- 以前示した定理より，総評価値最大の財の配分は均衡配分
- 総評価値最大配分の計算： 増加路を使ったアルゴリズムを
（大幅に）一般化した方法により可能

均衡配分の計算: 評価値が既知の場合

ステップ0: $X_1 = X_2 = \dots = X_m = \emptyset$ とおく.

ステップ1: $N \setminus \bigcup_{i=1}^m X_i = \emptyset$ ならば終了.

現在の配分 $(X_1, X_2, \dots, X_m)$ は総評価値最大の均衡配分.

ステップ2: 配分に対する以下の更新方法で,

総評価値が最大になるものを求める.

- X_{i_1} に, ある $j_1 \in N \setminus \bigcup_{i=1}^m X_i$ を追加, ある $j_2 \in X_{i_1}$ を削除.
- X_{i_2} に j_2 を追加, ある $j_3 \in X_{i_2}$ を削除.
- . . .
- $X_{i_{k-1}}$ に j_{k-1} を追加, ある $j_k \in X_{i_{k-1}}$ を削除.
- X_{i_k} に j_k を追加.

更新方法には,
厳密には
細かい条件が
必要

ステップ3: 上記の更新方法で配分を更新. ステップ1へ.

反復オークション

反復オークションのアルゴリズム

- その1: 均衡を近似的に計算 [Kelso-Crawford 1982]
 - Bertsekas (1979), Crawford, Knoer (1981) の一般化
 - 単調に価格を増加, **均衡配分** (および均衡価格の近似値) を求める
 - 各反復で, 入札者の利得最大の財集合**ひとつ**の情報が必要
 - 価格増加のルールは簡単: 希望が重複 → 価格を増やす
- その2: 均衡を厳密に計算 [Gul-Stacchetti 2000]
 - Demange, Gale, Sotomayor (1986) の一般化
 - 単調に価格を増加, **均衡価格** (および均衡配分) を求める
 - 各反復で, 入札者の利得最大の財集合**すべての**情報が必要
 - 価格増加のルールは複雑:
 - 得た情報を使い, 価格を増やす財をうまく選ぶ

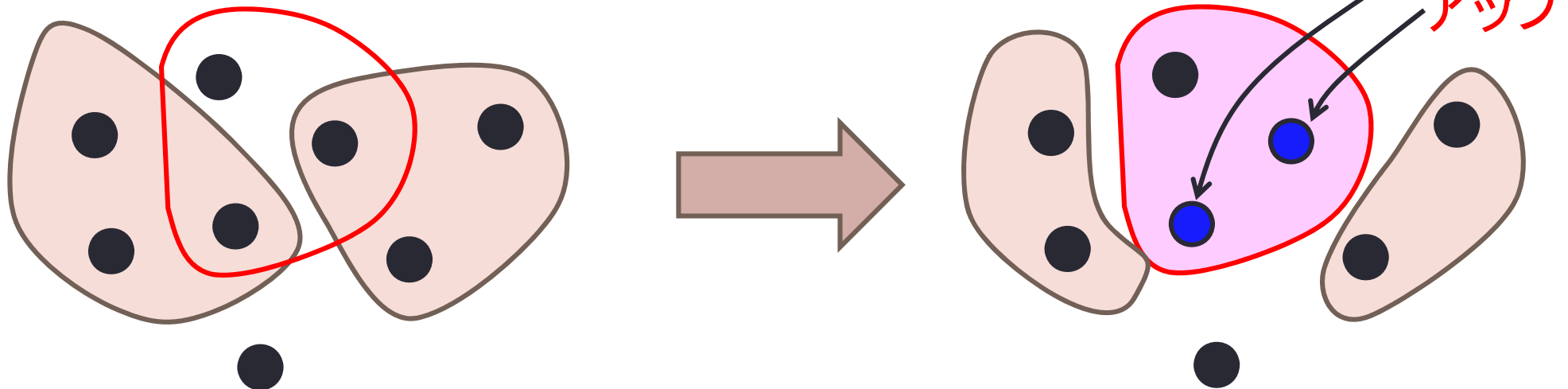
均衡を近似的に計算するアルゴリズム

アルゴリズムの概要

Kelso-Crawford (1982) が提案 (Blumrosen-Nisan(2007)も参照)

アルゴリズムの流れ

- 各入札者は順番に,
現在の価格の下で(ほぼ)利得最大の財集合を一つ選ぶ
- 選んだ財集合に, 他の入札者が既に選んだ財が含まれる
→ 重複した財を奪い, 価格を上げる
- 財を取られた入札者: 利得最大の財集合を選び直す
- 財の重複がなくなったら終了



均衡を近似的に計算：変数と出力

δ : アルゴリズムのパラメータ, > 0

変数: $p(j)$ --- 財 j の暫定価格, δ の整数倍

S_i --- 入札者 i に割り当ての財集合, 常に重複無し

出力: δ 均衡 --- 以下の条件を満たす

財の配分 $S_1, \dots, S_m$ と価格 $p(1), \dots, p(n)$

(条件1) 各入札者 i に対し, S_i は価格 p' において利得最大

$$p'(j) = \begin{cases} p(j) & (j \in S_i) \\ p(j) + \delta & (j \in N \setminus S_i) \end{cases}$$

(価格 p において, ほぼ利得最大の財集合が割り当て)

(条件2) $p(j) = 0$ ($\forall j \in N \setminus \bigcup_{i \in B} S_i$)

(未割り当ての財の価格 = 0)

均衡を近似的に計算: アルゴリズム

ステップ0: 各財 j に対し $p(j) = 0$. 各入札者 i に対し $S_i = \emptyset$.

ステップ1: 各入札者 i に対し, 価格 p' での利得最大の財集合で, S_i を含むものを D_i とおく.

$$p'(j) = \begin{cases} p(j) & (j \in S_i) \\ p(j) + \delta & (j \in N \setminus S_i) \end{cases}$$

ステップ2: 各入札者 i に対し $S_i = D_i \rightarrow \delta$ 均衡 (終了)

ステップ3: $S_i \neq D_i$ なる入札者 i を選ぶ.

ステップ4: 各 $j \in D_i \setminus S_i$ に対し, $p(j) := p(j) + \delta$.

S_i を D_i に置き換える.

他の入札者 h に対し, S_h と D_i に重複があれば,
それを削除.

ステップ1へ戻る.

均衡を近似的に計算：詳しいアルゴリズム

ステップ0: 各財 j に対し $p(j) = 0$, 各入札者 i に対し $S_i = \emptyset$,
 $B' = B$ とおく.

B' = 不満をもっている可能性
 のある入札者の集合

ステップ1: $B' = \emptyset$ ならば終了.

ステップ2: B' から入札者 i を選ぶ.

ステップ3: 入札者 i に対し, 価格 q での利得最大の
 財集合で, S_i を含むものを D とおく(←必ず存在)

$$q(j) = \begin{cases} p(j) & (j \in S_i) \\ p(j) + \delta & (j \in N \setminus S_i) \end{cases}$$

ステップ4: 各 $j \in D \setminus S_i$ に対し, $p(j) := p(j) + \delta$.

S_i を D に置き換える. B' から i を削除.

他の入札者 h に対し, S_h と D に重複があれば,

それを S_h から削除し, h を B' に追加.

ステップ1へ戻る.

単一需要モデルに特殊化

ステップ0: 各財 j に対し $p(j) = 0$, 各入札者 i に対し $j_i = 0$,
 $B' = B$ とおく.

B' = 財の割当のない
 入札者の集合

ステップ1: $B' = \emptyset$ ならば終了.

ステップ2: B' から入札者 i を選ぶ.

ステップ3: 入札者 i に対し, 価格 p での利得最大の財を j_i とおく

ステップ4: $p(j_i) := p(j_i) + \delta$.

B' から i を削除.

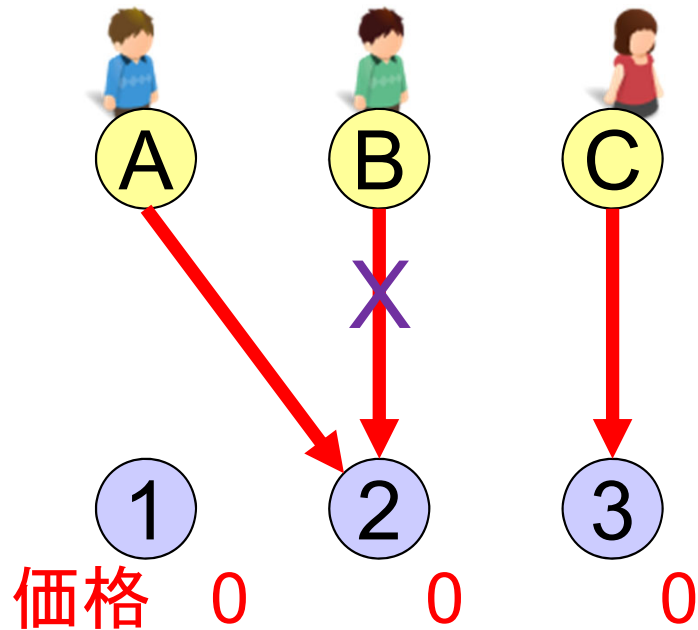
j^* を持っていた入札者 h を B' に追加.

ステップ1へ戻る.

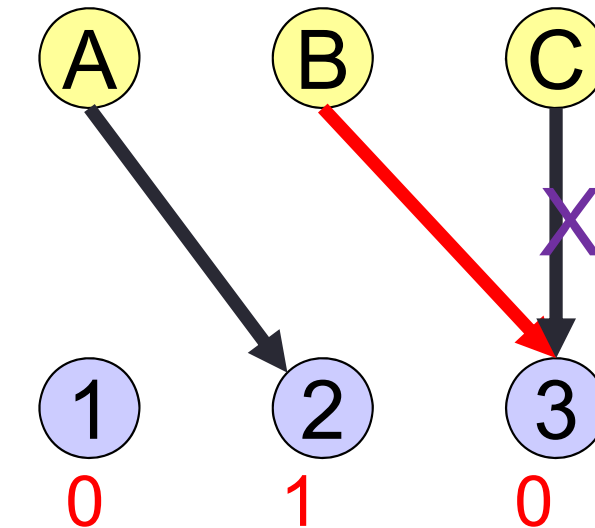
単一需要モデルのときのアルゴリズムに一致する

アルゴリズムの実行例

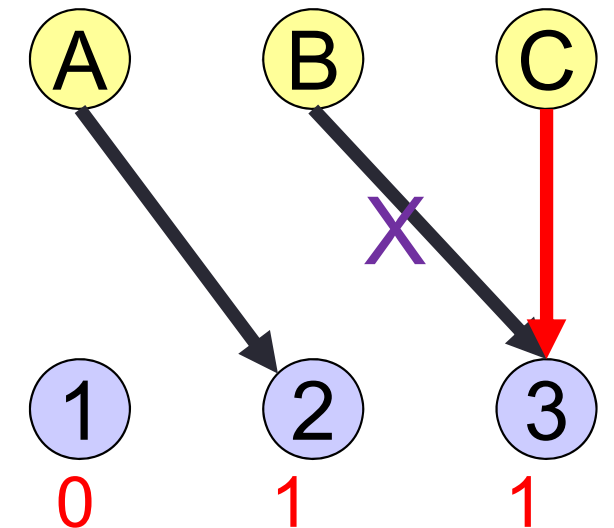
$\delta=1$ のとき



$v(i,j)$	A	B	C
①	3	1	1
②	6	5	3
③	2	4	4



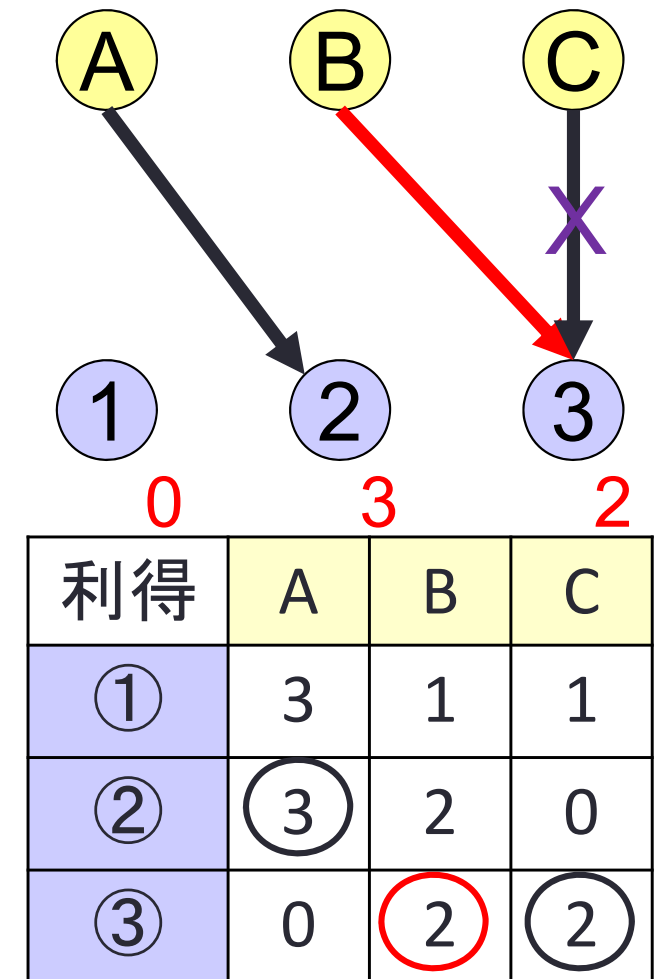
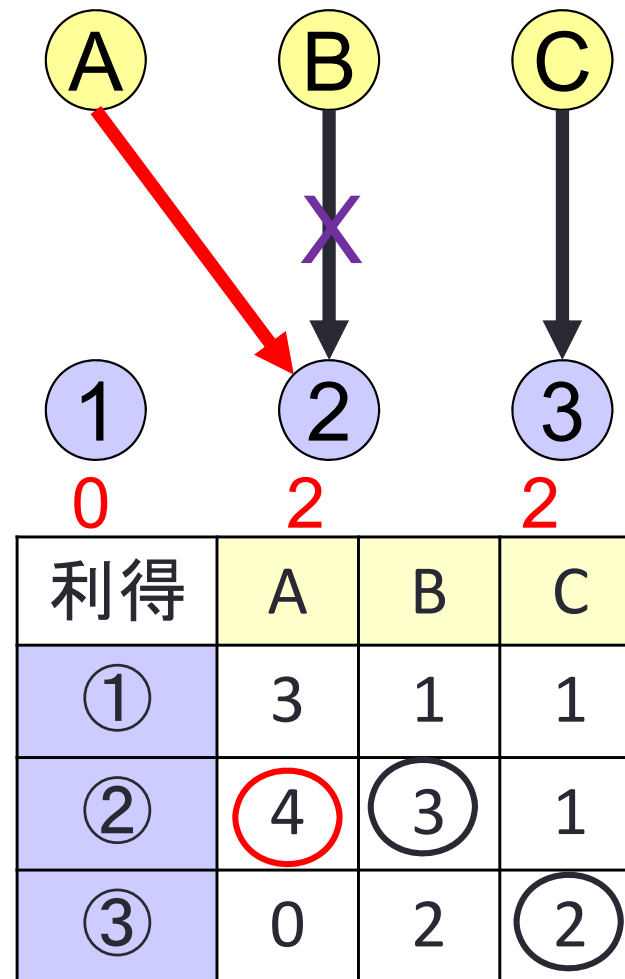
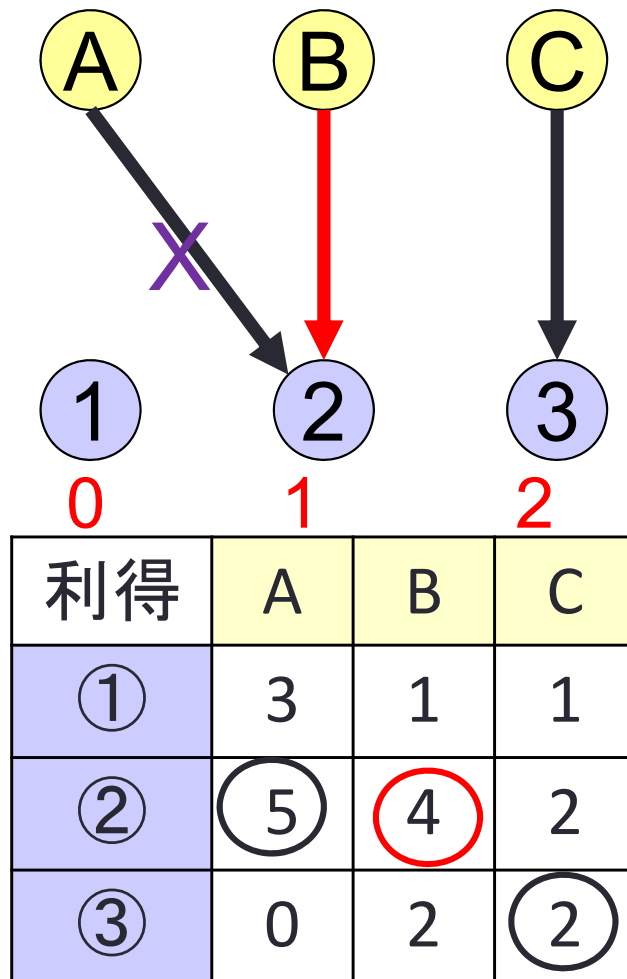
利得	A	B	C
①	3	1	1
②	5	4	2
③	2	4	4



利得	A	B	C
①	3	1	1
②	5	4	2
③	1	3	3

アルゴリズムの実行例(2)

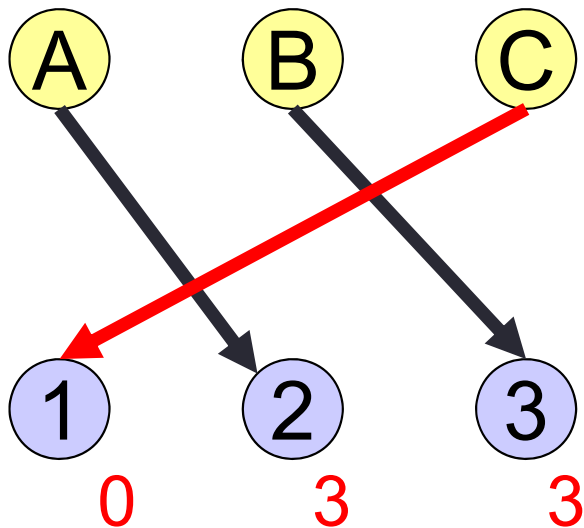
$\delta=1$ のとき



アルゴリズムの実行例(2)

$\delta=1$ のとき

終了
均衡配分 ×
均衡価格 ○



利得	A	B	C
①	3	1	1
②	3	2	0
③	-1	1	1

反復オークションのための問題例

- 入札者a: 重み和(①:2, ②:4, ③:3, ④:1, ⑤:5)
- 入札者b: 財集合(①:2, ②:5, ③:1, ④:4, ⑤:3) の中の
上位2つの財に依存
 $\{\text{①}, \text{②}, \text{③}\} \rightarrow \text{評価値} 2+5, \{\text{③}, \text{④}, \text{⑤}\} \rightarrow \text{評価値} 4+3$
- 入札者c: 財の数に依存(1つ:4, 2つ:7, 3つ:10, 4つ:12, 5つ:13)

均衡価格=(2,4,3,3,3) ←極小均衡価格

均衡配分: a {5}, b {2,4}, c {1,3}

または a {3,5}, b {2,4}, c {1}

アルゴリズムの実行例(1)

- 入札者a: 重み和(①:2, ②:4, ③:3, ④:1, ⑤:5)
- 入札者b: 財集合(①:2, ②:5, ③:1, ④:4, ⑤:3) の中の上位2つの財に依存
- 入札者c: 財の数に依存(1つ:4, 2つ:7, 3つ:10, 4つ:12, 5つ:13)

- ステップ0: 価格 $p=(0,0,0,0,0)$, 配分 $\emptyset, \emptyset, \emptyset$, $B'=\{a,b,c\}$
- ステップ2: $i=a$ を選択
- ステップ3: 入札者aの価格 $(1,1,1,1,1)$ での利得最大の集合 $D=N$
- ステップ4:
 - $D - S_a = N$ に含まれる財の価格を上げる $\rightarrow p=(1,1,1,1,1)$
 - $S_a = D = N$, $S_b = \emptyset$, $S_c = \emptyset$, $B' = \{b,c\}$

アルゴリズムの実行例(2)

- 入札者a: 重み和(①:2, ②:4, ③:3, ④:1, ⑤:5)
- 入札者b: 財集合(①:2, ②:5, ③:1, ④:4, ⑤:3) の中の上位2つの財に依存
- 入札者c: 財の数に依存(1つ:4, 2つ:7, 3つ:10, 4つ:12, 5つ:13)

- $p=(1,1,1,1,1)$
- $S_a = N, S_b = \emptyset, S_c = \emptyset,$
- $B' = \{b,c\}$
- ステップ2: $i=b$ を選択
- ステップ3: 入札者bの価格 $(2,2,2,2,2)$ での利得最大の集合
 $D=\{2,4\}$
- ステップ4:
 - $D - S_b = \{2,4\}$ に含まれる財の価格を上げる $\rightarrow p=(1,2,1,2,1)$
 - $S_b = D = \{2,4\}, S_a = N-D=\{1,3,5\}, S_c = \emptyset,$
 - $B' = \{a,c\}$

アルゴリズムの実行例(3)

- 入札者a: 重み和(①:2, ②:4, ③:3, ④:1, ⑤:5)
- 入札者b: 財集合(①:2, ②:5, ③:1, ④:4, ⑤:3) の中の上位2つの財に依存
- 入札者c: 財の数に依存(1つ:4, 2つ:7, 3つ:10, 4つ:12, 5つ:13)

- $p=(1,2,1,2,1)$
- $S_b = \{2,4\}$, $S_a = \{1,3,5\}$, $S_c = \emptyset$
- $B' = \{a,c\}$

- ステップ2: $i=c$ を選択
- ステップ3: 入札者cの価格 $(2,3,2,3,2)$ での利得最大の集合

$$D=\{1,3,5\}$$

- ステップ4:
 - $D - S_c = \{1,3,5\}$ に含まれる財の価格を上げる $\rightarrow (2,2,2,2,2)$
 - $S_c = D = \{1,3,5\}$, $S_a = \{1,3,5\} - D = \emptyset$, $S_b = \{2,4\} - D = \{2,4\}$,
 - $B' = \{a\}$

アルゴリズムの実行例(4)

- 入札者a: 重み和(①:2, ②:4, ③:3, ④:1, ⑤:5)
- 入札者b: 財集合(①:2, ②:5, ③:1, ④:4, ⑤:3) の中の上位2つの財に依存
- 入札者c: 財の数に依存(1つ:4, 2つ:7, 3つ:10, 4つ:12, 5つ:13)

- $p=(2,2,2,2,2)$
- $S_c = \{1,3,5\}$, $S_a = \emptyset$, $S_b = \{2,4\}$, $B' = \{a\}$
- ステップ2: $i=a$ を選択
- ステップ3: 入札者cの価格 $(3,3,3,3,3)$ での利得最大の集合
 $D=\{2,5\}$
- ステップ4:
 - $D - S_a = \{2,5\}$ に含まれる財の価格を上げる $\rightarrow(2,3,2,2,3)$
 - $S_a = D = \{2,5\}$, $S_b = \{2,4\}-D=\{4\}$, $S_c = \{1,3,5\}-D=\{1,3\}$,
 - $B' = \{b,c\}$

アルゴリズムの実行例(5)

- 入札者a: 重み和(①:2, ②:4, ③:3, ④:1, ⑤:5)
- 入札者b: 財集合(①:2, ②:5, ③:1, ④:4, ⑤:3) の中の上位2つの財に依存
- 入札者c: 財の数に依存(1つ:4, 2つ:7, 3つ:10, 4つ:12, 5つ:13)

- $p=(2,3,2,2,3)$
- $S_a = \{2,5\}$, $S_b = \{4\}$, $S_c = \{1,3\}$, $B' = \{b,c\}$
- ステップ2: $i=b$ を選択
- ステップ3: 入札者cの価格 $(3,4,3,2,4)$ での利得最大の集合
 $D=\{2,4\}$
- ステップ4:
 - $D - S_b = \{2\}$ に含まれる財の価格を上げる $\rightarrow (2,4,2,2,3)$
 - $S_b = D = \{2,4\}$, $S_c = \{1,3\}-D=\{1,3\}$, $S_a = \{2,5\}-D=\{5\}$,
 - $B' = \{a,c\}$

アルゴリズムの実行例(6)

- 入札者a: 重み和 (①:2, ②:4, ③:3, ④:1, ⑤:5)
- 入札者b: 財集合 (①:2, ②:5, ③:1, ④:4, ⑤:3) の中の上位2つの財に依存
- 入札者c: 財の数に依存 (1つ:4, 2つ:7, 3つ:10, 4つ:12, 5つ:13)

- $p=(2,4,2,2,3)$
- $S_a = \{5\}$, $S_b = \{2,4\}$, $S_c = \{1,3\}$, $B' = \{a,c\}$
- ステップ2: $i=c$ を選択
- ステップ3: 入札者cの価格 (2,5,2,3,4)での利得最大の集合
 $D=\{1,3\}$
- ステップ4:
 - $D - S_c = \emptyset$ に含まれる財の価格を上げる $\rightarrow$ p 変更せず
 - $S_a = \{5\}$, $S_b = \{2,4\}$, $S_c = \{1,3\}$, 変更せず
 - $B' = \{a\}$

アルゴリズムの実行例(7)

- 入札者a: 重み和 (①:2, ②:4, ③:3, ④:1, ⑤:5)
- 入札者b: 財集合 (①:2, ②:5, ③:1, ④:4, ⑤:3) の中の上位2つの財に依存
- 入札者c: 財の数に依存 (1つ:4, 2つ:7, 3つ:10, 4つ:12, 5つ:13)

- $p=(2,4,2,2,3)$
- $S_a = \{5\}$, $S_b = \{2,4\}$, $S_c = \{1,3\}$, $B' = \{a\}$
- ステップ2: $i=a$ を選択
- ステップ3: 入札者cの価格 (3,5,3,3,3)での利得最大の集合
 $D=\{5\}$
- ステップ4:
 - $D - S_a = \emptyset$ に含まれる財の価格を上げる $\rightarrow$ p 変更せず
 - $S_a = \{5\}$, $S_b = \{2,4\}$, $S_c = \{1,3\}$, 変更せず
 - $B' = \emptyset \rightarrow$ 次の反復でアルゴリズム終了

得られた財配分の近似最適性

定理 アルゴリズムで得られた財配分の総評価値
 $\geq$ 財配分の総評価値の最大値 $- n\delta$

系 評価値が整数値のとき, $\delta=1/(n+1)$ とおくと,
 アルゴリズムで得られた財配分 = 総評価値最大の財配分

[証明] $(S_1, \dots, S_m)$: アルゴリズムの財配分,

$(X_1, \dots, X_m)$: 総評価値最大の財配分

(条件1) 各入札者 i に対し, S_i は価格 p' において利得最大

$$p'(j) = \begin{cases} p(j) & (j \in S_i) \\ p(j) + \delta & (j \in N \setminus S_i) \end{cases}$$

(条件2) $p(j) = 0 \quad (\forall j \in N \setminus \bigcup_{i \in B} S_i)$

得られた財配分の近似最適性

$$p(S_i) \equiv \sum_{j \in S_i} p(j)$$

[証明のつづき]

$$\begin{aligned} \text{条件1より, } v_i(S_i) - p(S_i) &= v_i(S_i) - p'(S_i) \geq v_i(X_i) - p'(X_i) \\ &= v_i(X_i) - p(X_i) - |X_i \cap S_i|\delta \end{aligned}$$

$$\begin{aligned} \therefore \sum_{i \in B} v_i(S_i) - \sum_{i \in B} p(S_i) \\ \geq \sum_{i \in B} v_i(X_i) - \sum_{i \in B} p(X_i) - \sum_{i \in B} |X_i \cap S_i|\delta \end{aligned}$$

$$\text{条件2より, } \sum_{i \in B} p(S_i) = p(N)$$

$$\text{価格の非負性より, } \sum_{i \in B} p(X_i) \leq p(N)$$

$$X_i \cap S_i \text{ は互いに素なので, } \sum_{i \in B} |X_i \cap S_i|\delta \leq n\delta$$

$$\therefore \sum_{i \in B} v_i(S_i) - p(N) \geq \sum_{i \in B} v_i(X_i) - p(N) - n\delta$$

$$\therefore \sum_{i \in B} v_i(S_i) \geq \sum_{i \in B} v_i(X_i) - n\delta \quad \blacksquare$$

アルゴリズムの正当性

命題 各反復のステップ1において, 各入札者 h に対し,

$$\exists X \in D_h(p') \text{ s.t. } X \supseteq S_h$$

[証明] 命題が $k-1$ 回目のステップ1で成立と仮定,
 k 回目の反復でも成り立つことを証明.

仮定: 第 $k-1$ 回目のステップ1: $\exists X \in D_h(p') \text{ s.t. } X \supseteq S_h^{(k-1)}$

$S_h^{(k-1)}$: 第 $k-1$ 回目の反復における S_h

k 回目のステップ1の価格 p' を q' とおく

$\rightarrow \exists Y \in D_h(q') \text{ s.t. } Y \supseteq S_h^{(k)}$ を示せば良い

2つのケース

① $h=i$ (入札者 $h=i$ が直前のステップ4 で他人の財を奪っている場合)

$S_h^{(k)} = X \in D_h(p')$ なので, $q' = p'$ を示せばよい ($Y = S_h^{(k)}$ とすればよい)

$p' = p$ において, $S_h^{(k-1)}$ 以外の財のみ価格を δ だけアップ

$q = p$ において, $S_h^{(k)} - S_h^{(k-1)}$ の財のみ価格を δ だけアップ

$q' = q$ において, $S_h^{(k)}$ 以外の財のみ価格を δ だけアップ

$= p$ において, $S_h^{(k-1)}$ 以外の財のみ価格を δ だけアップ $= p'$

アルゴリズムの正当性

② $h \neq i$ (h は財を奪われる側の入札者の場合)

第 $k-1$ 回目のステップ4で $S_h^{(k-1)}$ から財が奪われる(かもしれない)

→その残りの財集合が $S_h^{(k)}$

入札者 i が他者から奪った財の価格のみアップ

$$\text{つまり, } q(j) = p(j) + \delta \left(j \in D_i \setminus S_i^{(k-1)} \right),$$

$$q(j) = p(j) \text{ (それ以外の財 } j) \leftarrow S_h^{(k)} \text{ の財はこちら}$$

$p' = p$ において, $S_h^{(k-1)}$ 以外の財のみ価格を δ だけアップ

$q' = q$ において, $S_h^{(k)} (\subseteq S_h^{(k-1)})$ 以外の財のみ価格を δ だけアップ

よって, $p' \leq q'$ かつ $p'(j) = q'(j) (j \in S_h^{(k)})$ 成立

粗代替性より, $\exists Y \in D_i(q')$

s.t. $Y \supseteq D_h$ の中で価格不変の財すべて $\supseteq S_h^{(k)}$

演習問題

問題1：財集合が $\{1,2,3,4\}$, 入札者3人の評価関数が以下のように与えられているとする. $\delta=10$ として, 均衡を近似的に計算するアルゴリズムを適用せよ.

- Aさん: ①を含む財集合は70,
①を含まず, ④を含む財集合は50,
それ以外は0
- Bさん: 重み和 (①:60, ②:70, ③:40, ④:30, ⑤:80)
- Cさん: 財の数依存 (1つ:90, 2つ:150, 3つ:200, 4つ:240, 5つ:250)