

Modèle des jeux et des mécanismes

Michel de Rougemont
 Université Paris II
<http://www.lri.fr/~mdr>



Plan

Partie 1

1. Exemples de Jeux et de Mécanismes
2. Jeux à somme nulle
3. Jeux matriciels: équilibres de Nash
4. Calcul des équilibres: Lemke-Howson

Partie 2

1. Approximation d'équilibres
2. Calcul polynomial d'équilibres de marché
3. Valeur relative pour XML

I.1 Jeux et Mécanismes

1. Modèle de calculs, adapté à un nombre important d'agents, suivant une fonction d'utilité.
2. Jeux: N joueurs suivant chacun un but. Quels sont les Equilibres?
3. Mécanismes: observons un équilibre, de quel jeu sommes nous l'équilibre?

Jeux Matriciels

Dilemme des Prisonniers: deux décisions C (collaborer), D (Trahir)

		II	
		C	D
I	C	(3,3)	(0,4)
	D	(4,0)	(1,1)

N joueurs: $A(s_1, s_2, \dots, s_N) = (a_1, a_2, \dots, a_N)$

Stratégie du joueur j, $x^j = \begin{pmatrix} x_1 \\ \dots \\ x_k \end{pmatrix}$ tel que $\sum_i x_i = 1$. Gain de I: $(x^j)^T \cdot A \cdot x^2$

Définition: (x^1, x^2) est un équilibre de Nash, si:

$$\forall x^1 (x^1)^T \cdot A \cdot x^2 \geq (x^1)^T \cdot A \cdot x^2 \text{ et } \forall y^1 (x^1)^T \cdot B \cdot x^2 \geq (y^1)^T \cdot B \cdot x^2$$

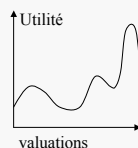
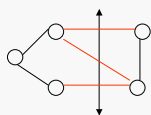
Exemples de Jeux

MaxSAT
SAT

0 1 1 0 1

I II ... V

MaxCUT
Gn jeux à N joueurs



Jeux sous forme extensive
 Jeux de vérification. Graphes d'accessibilité.
 Jeux logiques: Nord-Est, dames, Echecs.

Exemple de mécanisme

1. Comment faire la différence entre un vrai mail et un SPAM?
2. Modifications au protocole de mail (pop, smtp)
3. Valeur d'un Email?
4. Mécanismes classiques: enchères

Valeur proportionnelle aux calculs demandés à Alice par Bob

Modifications au protocole de mail (pop, smtp)

1. A prend un ticket sur la page Web de B. (Entrée x d'un problème)
2. A calcule $f(x)=y$
3. A envoie y et l'Email
4. B vérifie y



A calcule une fonction polynomiale

A prend un ticket sur la page Web de B.

B : génère un polynôme aléatoire de degré n

$$P(x)=a_1 \cdot x + a_2 x^2 + \dots + a_n x^n$$

B: choisit n+1 valeurs aléatoires

$$y_1=P(x_1), y_2=P(x_2), \dots, y_{n+1}=P(x_{n+1})$$

Ticket = $(x_1, y_1), (x_2, y_2), \dots, (x_{n+1}, y_{n+1})$

A doit trouver P(x) à partir du ticket.

Interpolation ou Inversion matricielle

B vérifie le calcul

B garde P(x) lorsqu'il génère le ticket.

Vérifier consiste à comparer les coefficients de P(x) avec ceux envoyés par A.

$$P(x)=a_1 \cdot x + a_2 x^2 + \dots + a_n x^n$$

On peut paramétrer:

le degré, la précision des valeurs aléatoires pour forcer A à calculer 10 minutes 30 minutes....

Interpolation est polynomiale

La vérification est triviale

I.2 Jeux à somme nulle

Deux joueurs I et II:

$$\begin{pmatrix} 0 & 2 & -3 & 0 \\ -2 & 0 & 0 & 3 \\ 3 & 0 & 0 & -4 \\ 0 & -3 & 4 & 0 \end{pmatrix}$$



Gain de II = - Gain de I

Jeu Morra: chaque joueur cache 1 ou 2 Euros et cherche à deviner le choix de l'autre joueur. Il gagne s'il devine correctement. Si 1 seul joueur gagne, son gain est le montant caché total, payé par l'autre joueur, sinon le gain est de 0

Gain du Jeu

Gain du jeu :

$$\sum_{i=1}^n \sum_{j=1}^n a_{i,j} x_i y_j$$

Joueur I : $Max_x Min_y x^T \cdot A \cdot y$

$$Min_y x^T \cdot A \cdot y = Min_j \sum_{i=1}^n a_{i,j} x_i = t$$

Réponse de II peut être pure

$$x^T \cdot A \cdot y = \sum_{j=1}^n y_j \left(\sum_{i=1}^n a_{i,j} x_i \right) \geq \sum_{j=1}^n y_j t = t$$

Donc : $Min_y x^T \cdot A \cdot y \geq t$

Toute solution pure doit satisfaire

$$Min_y x^T \cdot A \cdot y \leq Min_j \sum_{i=1}^n a_{i,j} x_i = t$$

Stratégie optimale

Conclusion $t \leq Min_y x^T \cdot A \cdot y \leq t$

$$Min_y x^T \cdot A \cdot y = t$$

Joueur II peut jouer une stratégie pure

$$Max Min_j \sum_{i=1}^n a_{i,j} x_i$$

$$\sum_{i=1}^n x_i = 1$$

$$Max z$$

$$z - \sum_{i=1}^n a_{i,j} x_i \leq 0$$

$$\sum_{i=1}^n x_i = 1$$

Stratégie optimale

Jeu de Morra:

$$\begin{array}{rcl} \text{Max} & z & \\ z+ & -2x_2+3x_3 & \leq 0 \\ z+ & 2x_1 & -3x_4 \leq 0 \\ z & -3x_1 & +4x_4 \leq 0 \\ z & +3x_2-4x_3 & \leq 0 \\ & x_1 +x_2 +x_3 +x_4 & =1 \end{array}$$

Solution $x^* = [0, 3/5, 2/5, 0]$

Résolution par simplex.

Théorème Minmax

Situation pour le joueur II

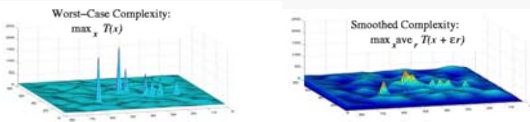
$$\begin{array}{l} \text{Min} \text{ Max}_i \sum_{j=1}^n a_{ij} y_j \\ \sum_{j=1}^n y_j = 1 \\ \text{Min} \sum_{j=1}^n w \\ w - \sum_{j=1}^n a_{ij} y_j \leq 0 \\ \sum_{j=1}^n y_j = 1 \end{array}$$

Problème dual du précédent. Par dualité:

Théorème (Von Neuman) : Max Min = Min Max

Analyse du Simplex

D. Spielman, M.I.T., 2001



- Simplex peut être exponentiel.
- Simplex EST polynomial pour la complexité de lissage.
- Application pratique: modifier aléatoirement la matrice A, et l'algorithme converge plus vite.

1.3 Jeux matriciels généraux

Deux joueurs: les gains de I et II sont définis par deux matrices A,B de même dimension. Pour n joueurs, n hypercubes.

$$A = \begin{pmatrix} 2 & 1 & 3 \\ 3 & 3 & 1 \end{pmatrix} \quad B = \begin{pmatrix} 1 & 3 & 2 \\ 4 & 1 & 3 \end{pmatrix}$$

$$\text{Max } x^T A y$$

$$\text{Min } e^T u$$

$$E x = e$$

$$E^T u \geq A y$$

Solution possible:

$$x^* = [1, 0], y^* = [0, 0, 1]$$

Solution (x^*, y^*) est un équilibre de Nash.

Jeux matriciels

Par dualité: $x^T A y = e^T u$ Primal = Dual

$$E x = e \Rightarrow e^T = x^T E^T$$

$$x^T A y = x^T E^T u$$

$$x^T (E^T u - A y) = 0$$

$$E^T u - A y \geq 0$$

Pour le joueur II:

$$\text{Max } x^T B y$$

$$F x = f$$

$$y^T (F^T v - B^T x) = 0$$

$$(F^T v - B^T x) \geq 0 \text{ par dualité}$$

C.N.S. pour être un équilibre de Nash

Un couple (x, y) est un équilibre de Nash ssi il existe u, v tel que:

$$E x = e$$

$$F y = f$$

$$E^T u - A y \geq 0$$

$$F^T v - B^T x \geq 0$$

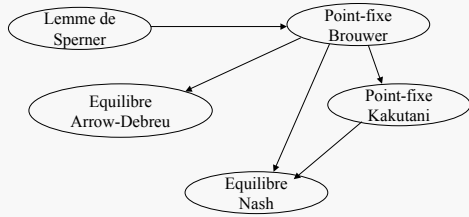
$$x^T (E^T u - A y) = 0$$

$$y^T (F^T v - B^T x) = 0$$

Programme linéaire + contraintes quadratiques de complémentarité.

Simplex + complémentarité = Lemke-Howson

Existence d'Equilibres

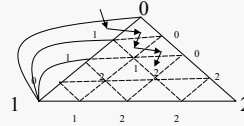


Preuves non-constructives.

Lemme de Sperner

Etiqueter un simplex:

- Chaque point frontière ne peut pas avoir l'étiquette du sommet opposé.
- Chaque point intérieur a une étiquette arbitraire.

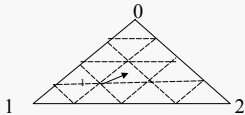


Sperner : il existe un triangle 0-1-2

Commencer sur le côté gauche avec une arête 0-1 qui détermine un triangle qui admet une autre arête 0-1. On parcourt ainsi des triangles 1 seule fois. Il existe un nombre fini de triangles et on doit terminer sur 0-1-2.

Point fixe de Brouwer

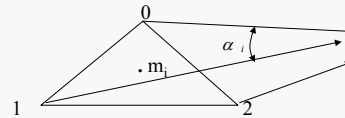
Brouwer: Soit Φ une fonction continue sur un n -simplex. Il existe x tel que $\Phi(x) = x$.



Soit $T_1, T_2, \dots, T_i, \dots$ un découpage en triangles de plus en plus fins.

Déterminer un coloriage en détectant le côté traversé par $\Phi(x)$. C'est un étiquetage de Sperner. Il existe un triangle t_i 0-1-2 de centre m_i . Pour une séquence de m_i il existe une sous-séquence x_i qui converge vers x , point fixe.

Point fixe de Brouwer



Pour une séquence de m_i il existe une sous-séquence x_i qui converge vers x^* .

Si le coloriage est 0-1-2, la situation ci-dessus est impossible, au moins un des angles $\alpha_i \geq \frac{\pi}{3}$.

Continuité de f implique que $\Phi(x^*) = x^*$

Existence de Nash

Soit s une stratégie pure du joueur j :

$$d_{j,s} = A_j(p_1, p_2, \dots, p_{i-1}, s, p_{i+1}, \dots, p_n) - A_j(p_1, p_2, \dots, p_{i-1}, p_i, p_{i+1}, \dots, p_n)$$

$$f(p_1, p_2, \dots, p_n) = (p_1', p_2', \dots, p_n')$$

Idée:
$$p_{j,s}' = p_{j,s} + d_{j,s}$$

Pour éviter valeur négative et maintenir une distribution:

$$p_{j,s}' = (p_{j,s} + \text{Max}(d_{j,s}, 0)) / (\sum_s p_{j,s} + \text{Max}(d_{j,s}, 0))$$

Nash est le point-fixe de f .

Equilibre Arrow-Debreu

Entrée:

- Ensemble B d'acheteurs
- Ensemble A de biens divisibles
- Vecteur M de valeurs m_i entières pour chaque acheteur
- Matrice Utilité: u_{ij} donnant l'utilité du produit i pour l'acheteur j .

Sortie: vecteur de prix p_i pour chaque produit i

- Chaque acheteur maximise son utilité
- Tout est dépensé
- Tout est acheté

x_i est le vecteur de biens achetés par i

$$p \cdot x_i = m_i \quad \sum_j x_{i,j} = a_j$$

$$\text{Max} \quad x_{i,j} \cdot u_{i,j}$$

Modèle Arrow-Debreu

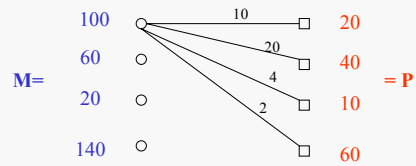
Entrée:

- A, ensemble de n produits
- B, ensemble de m consommateurs (Buyers)
- m, vecteur entier de ressources m_i
- Utilités, matrice d'entiers $u_{i,j}$ du cons. i pour le produit j

Sortie:

- Vecteur de prix p_j
- Allocation $x_{i,j}$
- Marché s'équilibre: (tout est dépensé et tout est consommé)
- Chaque consommateur maximise son utilité.

Modèle Arrow-Debreu



B: Buyers

A: Products

$$\text{Buyer } i: \begin{cases} \text{Max} & \sum_j x_{i,j} u_{i,j} \\ m_i & = \sum_j x_{i,j} p_j \\ x_{i,j} & \leq 1 \end{cases}$$

Il existe P, tel que l'allocation de chaque consommateur est optimum.

Equilibre Arrow-Debreu

Arrow-Debreu: il existe un vecteur p qui résout le marché.

Preuve: définir un potentiel pour p.

- Si la demande trop forte, augmenter p

$$\Phi(p_j) = \frac{p_j + (m_j - \sum_i p_j x_{i,j})}{C}$$

D'après Brouwer, il existe un point fixe qui résout le marché.

Observations:

- L'équilibre peut-être non calculable au sens des réels (Richter et Wong)
- Algorithme polynomial au sens BSS (Devanur, Papadimitriou, Saberi, Vazirani)

I.4 Algorithme de Lemke-Howson

Procédure algorithmique pour trouver des équilibres: Simplex+complémentarité.

$$A = \begin{pmatrix} 2 & 1 & 3 \\ 3 & 3 & 1 \end{pmatrix} \quad B = \begin{pmatrix} 1 & 3 & 2 \\ 4 & 1 & 3 \end{pmatrix}$$

Algorithme LH

1. Déterminer les points frontières et le graphe LH dans chaque Simplex,
2. Colorier les simplex de I et II avec des couleurs représentant les stratégies pures de I et II,
3. Naviguer à partir de l'origine jusqu'à un couple (x,y) avec toutes les couleurs (Nash).

Coloriage Shapley dans Lemke-Howson

M={1,2} pour 2 stratégies pures de I

N={3,4,5} pour 3 stratégies pures de II

$$A = \begin{pmatrix} 2 & 1 & 3 \\ 3 & 3 & 1 \end{pmatrix} \quad B = \begin{pmatrix} 1 & 3 & 2 \\ 4 & 1 & 3 \end{pmatrix}$$

Une stratégie x est coloriée par (1,3) si x=(0,1), i.e. I n'utilise pas la 1ère décision, et 3 est la meilleure réponse de II.

Théorème (Nash 1951): (x,y) est un équilibre pour (A,B) ssi

$$x_i > 0 \Rightarrow a_i y = \text{Max}_{k \in M} a_k y$$

$$y_i > 0 \Rightarrow b_i x = \text{Max}_{l \in N} b_l x$$

Théorème: (x,y) est un équilibre pour (A,B) ssi les couleurs de x et de y couvrent M+N.

Algorithme de Lemke-Howson

LH Graphes dans les simplex de I et II

- Extrémités du simplex

• Points frontières

$$\sum_i x_{i,1}, b_{i,1} = \sum_i x_{i,2}, b_{i,2} \quad \text{ou} \quad \sum_i x_{i,2}, b_{i,2} = \sum_i x_{i,3}, b_{i,3} \quad \text{ou} \quad \sum_i x_{i,1}, b_{i,1} = \sum_i x_{i,3}, b_{i,3}$$

Exemple :

$$\text{Soit} \quad \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} \lambda \\ \lambda \end{pmatrix}$$

$$\text{Soit} \quad \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix} \mu \\ 1 - \lambda - \mu \\ \mu \end{pmatrix}$$

Algorithme de Lemke-Howson

$$A = \begin{pmatrix} 2 & 1 & 3 \\ 3 & 3 & 1 \end{pmatrix} \quad B = \begin{pmatrix} 1 & 3 & 2 \\ 4 & 1 & 3 \end{pmatrix}$$

Colonnes de B:

Colonnes 1, 2: $\lambda + 4(1-\lambda) = 3\lambda + (1-\lambda)$ d'où $\lambda = 3/5$
 Pour $\lambda = 3/5$ valeur = 11/5 pour colonnes 1,2
 Pour $\lambda = 3/5$ valeur = 12/5 pour colonne 3. Non point limite.

Colonnes 2, 3: $3\lambda + (1-\lambda) = 2\lambda + 3(1-\lambda)$ d'où $\lambda = 2/3$
 Pour $\lambda = 2/3$ valeur = 7/3 pour colonnes 1,2
 Pour $\lambda = 2/3$ valeur = 2 pour colonne 3. Point limite.

Colonnes 1, 3: $\lambda + 4(1-\lambda) = 2\lambda + 3(1-\lambda)$ d'où $\lambda = 1/2$
 Pour $\lambda = 1/2$ valeur = 5/2 pour colonnes 1,3
 Pour $\lambda = 1/2$ valeur = 2 pour colonne 2. Point limite.

Lignes de A:

Lignes 1,2 dans A 2. $\lambda + \mu + 3(1-\lambda-\mu) = 3\lambda + 3\mu + (1-\lambda-\mu)$
 $2 = 3\lambda + 4\mu$

Algorithme de Lemke-Howson

$$A = \begin{pmatrix} 2 & 1 & 3 \\ 3 & 3 & 1 \end{pmatrix} \quad B = \begin{pmatrix} 1 & 3 & 2 \\ 4 & 1 & 3 \end{pmatrix}$$

Coloriage LH Graphes dans les simplexes de I et II

5 couleurs: (1,2) pour I et (3,4,5) pour II.

Coloriage dans le simplexe de I:

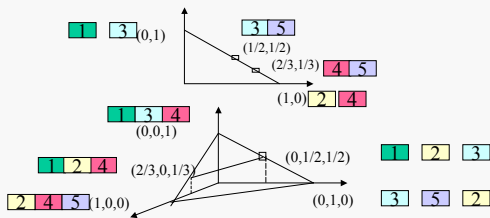
Si $x_i = 0$, point (x_1, x_2) colorié i pour $i = 1$ ou 2
 Pour un point x, si colonne j optimale (B),
 point x colorié j ($j = 3,4,5$)

Coloriage symétrique pour II

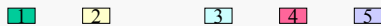
Si $y_j = 0$, point (y_1, y_2, y_3) colorié j pour $j = 3,4,5$
 Pour un point y, si ligne i optimale (A),
 point y colorié i ($i = 1,2$)

Algorithme de Lemke-Howson

$$A = \begin{pmatrix} 2 & 1 & 3 \\ 3 & 3 & 1 \end{pmatrix} \quad B = \begin{pmatrix} 1 & 3 & 2 \\ 4 & 1 & 3 \end{pmatrix}$$



Couleurs Lemke-Howson



Algorithme de Lemke-Howson

$$A = \begin{pmatrix} 2 & 1 & 3 \\ 3 & 3 & 1 \end{pmatrix} \quad B = \begin{pmatrix} 1 & 3 & 2 \\ 4 & 1 & 3 \end{pmatrix}$$

Lemke-Howson:

Paire (x_1, x_2) (y_1, y_2, y_3) est un équilibre de Nash
 ssi l'union de leurs couleurs = l'ensemble C des couleurs.

Exemple:

Paire (0,1) (1,0,0) est un équilibre de Nash
 Paires (2/3, 1/3) (0, 1/2, 1/2) et (1/2, 1/2) (2/3, 0, 1/3)
 sont les 2 autres équilibres de Nash

Procédure algorithmique:

Commencer en (0,0),(0,0,0) et choisir une couleur à exclure pour x puis pour y.

On termine sur un équilibre de Nash.

Développement récents

1. Approximation de Nash : (Lipton et al. EC 03)
2. Calcul Polynomial d'Equilibres de marchés (Devanur et al. STOC 01)
3. Valeur de l'Information relative à un schéma XML
 Property testing of regular trees, ICALP 04,
 Property and Equivalence testing on strings, ECC04
 - Rationalité limitée (stratégies d'automates)
 - Jeux de congestion et de potentiel
 - Mécanismes véreux, enchères combinatoires
 - Equilibres corrélés

II.1 Equilibres approchés

Une paire (x', y') est un équilibre approché si:

$$\forall x, x.A.y' \leq x'.A.y + \epsilon$$

$$\forall y, x'.B.y \leq x'.B.y' + \epsilon$$

Lipton, Markakis, Mehta 2003: Pour tout équilibre de Nash (x^*, y^*) il existe un équilibre approché qui l'approxime, de support:

On a:

$$|x'.A.y' - x^*.A.y^*| \leq \epsilon$$

$$|x'.B.y' - x^*.B.y^*| \leq \epsilon$$

Référence: Playing large games using simple strategies, R. Lipton, E. Markakis, A. Mehta, ECOM 03

Existence d'équilibres approchés

Existence d'équilibres approchés démontrée par la méthode probabiliste.

Prob [il existe un tel (x',y')] > 0

Exemple : (3/5,2/5) (0,1/2,1/2) est Nash
(1,0) (0,1/3,2/3) est-il ϵ -Nash ?

Preuve: Soit k tirages de stratégies pures selon x^* .

Soit x' la stratégie mixte uniforme obtenue.

$$\Phi_1 : \left\{ \left| \sum_{i=1}^n x'^i . A_{i,y} - \sum_{i=1}^n x^* . A_{i,y} \right| \leq \epsilon / 2 \right\}$$

$$\Phi_2 : \left\{ \left| \sum_{j=1}^n x'^i . B_{i,y} - \sum_{j=1}^n x^* . B_{i,y} \right| \leq \epsilon / 2 \right\}$$

$$\pi_{1,i} : \left\{ x'^i . A_{i,y} \leq x^i . A_{i,y} + \epsilon \right\} \text{ pour } i=1, \dots, n$$

$$\pi_{2,j} : \left\{ x'^j . B_{i,y} \leq x^j . B_{i,y} + \epsilon \right\} \text{ pour } j=1, \dots, n$$

$$\text{Good} = \Phi_1 \cap \Phi_2 \cap \pi_{1,i} \cap \pi_{2,j}$$

Estimation de la probabilité d'existence

$$\text{On veut montrer : } \Pr [\text{Good}] > 0$$

$$\Pr [\neg \text{Good}] \leq \Pr [\neg \Phi_1] + \Pr [\neg \Phi_2] + \sum_{i=1}^n \Pr [\neg \pi_{1,i}] + \sum_{j=1}^n \Pr [\neg \pi_{2,j}]$$

Borner chaque probabilité:

$$\text{Pour } \Phi_1 : \left\{ \left| \sum_{i=1}^n x'^i . A_{i,y} - \sum_{i=1}^n x^* . A_{i,y} \right| \leq \epsilon / 2 \right\}$$

$$\Phi_{1,a} : \left\{ \left| \sum_{i=1}^n x'^i . A_{i,y} - \sum_{i=1}^n x^* . A_{i,y} \right| \leq \epsilon / 4 \right\}$$

$$\Phi_{1,b} : \left\{ \left| \sum_{i=1}^n x'^i . A_{i,y} - \sum_{i=1}^n x^* . A_{i,y} \right| \leq \epsilon / 4 \right\}$$

$$\Phi_{1,b} \cap \Phi_{1,b} \subseteq \Phi_1$$

$\sum_{i=1}^n x^i . A_{i,y}$ est la somme de k variables indépendantes de moyenne $x^* . A_{i,y}$

Prob [Good] > 0

$$\Phi_{1,a} : \left\{ \left| \sum_{i=1}^n x'^i . A_{i,y} - \sum_{i=1}^n x^* . A_{i,y} \right| \leq \epsilon / 4 \right\}$$

$$\Pr [\neg \Phi_{1,a}] \leq 2 \cdot e^{-k \epsilon^2 / 8}$$

Par Chernoff-Hoeffding:

Similairement pour

$$\neg \Phi_{1,b} \text{ et donc } \Pr [\neg \Phi_1] \leq 4 \cdot e^{-k \epsilon^2 / 8}$$

$$\Pr [\neg \text{Good}] \leq \Pr [\neg \Phi_1] + \Pr [\neg \Phi_2] + \sum_{i=1}^n \Pr [\neg \pi_{1,i}] + \sum_{j=1}^n \Pr [\neg \pi_{2,j}]$$

$$\Pr [\neg \text{Good}] \leq 8 \cdot e^{-k \epsilon^2 / 8} + 8 \cdot n \cdot (e^{-k \epsilon^2 / 8}) < 1$$

$$\text{pour } k = (\log n) / \epsilon^2$$

II.2 Equilibre de marché (Arrow-Debreu)

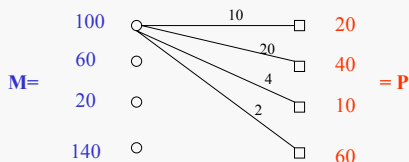
Entrée:

- A, ensemble de n produits
- B, ensemble de m consommateurs (Buyers)
- m, vecteur entier de ressources m_i
- Utilités, matrice d'entiers $u_{i,j}$ du cons. i pour le produit j

Sortie:

- Vecteur de prix p_j
- Allocation $x_{i,j}$
- Marché s'équilibre: (tout est dépensé et tout est consommé)
- Chaque consommateur maximise son utilité.

Modèle Arrow-Debreu



$$\text{Buyer } i : \begin{pmatrix} \text{Max} & \sum_{j=1}^n x_{i,j} u_{i,j} \\ m_i = & \sum_{j=1}^n x_{i,j} p_j \\ x_{i,j} \leq & 1 \end{pmatrix} \quad \text{Il existe P, tel que l'allocation de chaque consommateur est optimum.}$$

Arrow-Debreu

Historique:

• Irving Fisher 1891 (concave functions)

–Hydraulic apparatus for calculating equilibrium

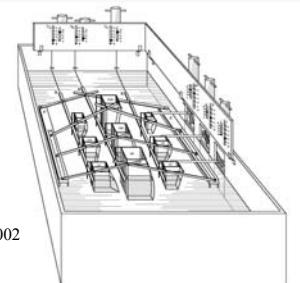
• Eisenberg & Gale 1959

–(unique) equilibrium exists

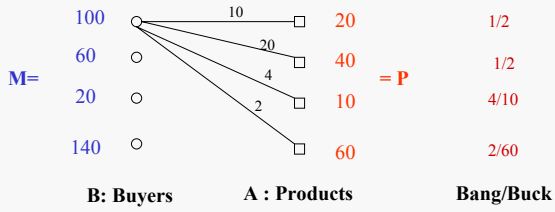
• Devanur, Papadimitriou, Saberi & V. 2002

–poly time alg for linear case

• V. 2002: Generalization of linear case



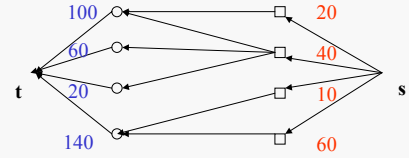
Modèle Arrow-Debreu



$$\text{Buyer } i : \begin{cases} \text{Max } \sum_j x_{i,j} u_{i,j} \\ m_i = \sum_j x_{i,j} p_j \\ x_{i,j} \leq 1 \end{cases}$$

Approche DPSV : Devanur, Papadimitriou, Daberi, Vazirani

BB : Bang per buck of Buyer $i : \alpha_i = \max_j \frac{u_{i,j}}{p_j}$



Sous-graphe BB : (i,j) existe si BB maximum.

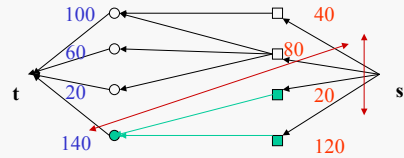
Algorithme DPSV pour Arrow-Debreu

- Initialisation des prix:** $p_j = \frac{1}{n}$
- Tester si tous les produits sont connectés à un acheteur.
 - Si non, baisser les prix correspondants.

- Augmentation des prix jusqu'à atteindre l'équilibre:**
- Maintenir s comme coupe minimum (tout est consommé).
 - Tester si t est une coupe minimum (tout est dépensé).
 - Trouver l'augmentation des prix
 - Déterminer une partie du sous-graphe qui réalise un sous-équilibre

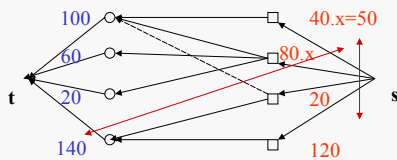
Primal Dual : augmenter les prix, Max-Flot

Flot Maximum dans le graphe DPSV



- Augmenter les prix pour le graphe non-figé:**
- Maintenir (t+A+B, s) comme coupe minimum
 - Une autre coupe fige un sous-graphe (Event 1)
 - Si une nouvelle arête apparaît défiger la composante (Event 2).

Flot Maximum dans le graphe DPSV



$$S \subseteq A, \quad p(S) = \sum_{i \in S} p_i$$

$$\Gamma(S) = \{i \in B : \exists j \in A, (i, j) \in E\} \quad m(\Gamma(S)) = \sum_{i \in \Gamma(S)} m_i$$

Lemme1 : s est une coupe minimum ssi

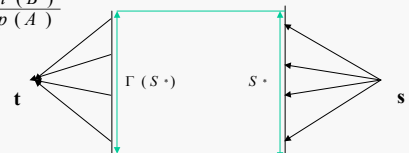
$$\forall S \subseteq A, \quad p(S) \leq m(\Gamma(S))$$

Itérations dans le graphe DPSV

Lemme2 : on peut déterminer x^* avec n Max-Flot itérations.

$$x^* = \min_{S \subseteq A} \frac{m(\Gamma(S))}{p(S)} \quad \text{pour } S^*$$

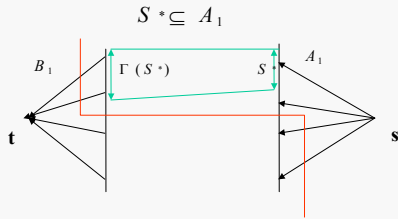
$$x = \frac{m(B)}{p(A)}$$



Si {s} est une coupe minimum, alors $x=x^*$.

Récursion DPSV avec Max-Flot

Lemme 3: si $\{s\}$ n'est pas une coupe minimum, mais $\{s+A_1+B_1\}$ alors :



Réappliquer le raisonnement sur (A_1, B_1) .
Au plus n itérations.

Nombre d'itérations de DPSV

Lemme 3: A chaque itération les prix sont de la forme a/b où $b \leq \Delta = n \cdot U^n$ où $U = \max_{i,j} u_{i,j}$

$$\frac{u_{1,1}}{p_1} = \frac{u_{1,2}}{p_2} \quad \circ \begin{array}{l} \text{---} \square \\ \text{---} \square \\ \text{---} \square \end{array}$$

$$\frac{u_{2,1}}{p_2} = \frac{u_{2,3}}{p_3} \quad \circ \begin{array}{l} \text{---} \square \\ \text{---} \square \end{array}$$

$$p_3 = \frac{u_{2,3}}{u_{2,2}} \cdot \frac{u_{1,2}}{u_{1,1}} \cdot p_1$$

$$m(\Gamma(S)) = p(S) = \sum_{j \in S} p_j = p_j \cdot \sum_{k \in S} \frac{a_k}{b_k}$$

$$p_j = \frac{a}{b} = \frac{m(\Gamma(S))}{\sum_{k \in S} \frac{a_k}{b_k}} \text{ et } b \leq \Delta$$

Analyse de DPSV

Lemme 4: Au plus $m(B) \cdot n^2 \cdot \Delta^2 \cdot T_{MaxFlot}$
 $p_j = \frac{a}{b} > \frac{c}{d}$ et $b, d \leq \Delta$ alors $\frac{a}{b} - \frac{c}{d} \geq \frac{1}{\Delta^2}$

Algorithme Pseudo-polynomial.

Version Polynomiale:

• Figurer S tel que:

$$S^* \subseteq S \subseteq A, \quad p(S) + \varepsilon \leq m(\Gamma(S)) \leq p(S) + |S| \cdot \varepsilon$$

• Augmenter les prix de ε

• Surplus $\leq n \cdot \varepsilon$

Versión polynomiale de DPSV

Versión Polynomiale: $\varepsilon = \frac{m(B)}{2n}$

• DPSV implique le surplus est $\leq n \cdot \varepsilon = \frac{m(B)}{2}$

• Diviser ε par 2 et après i itérations le surplus est $\leq \frac{m(B)}{2^i}$

• Après $O(\log(m(B) \cdot \Delta^2))$ surplus $\frac{m(B)}{2^i} \leq \frac{1}{\Delta^2}$

• Appliquer DPSV avec $\varepsilon = 0$

II. 3 Valeur de l'Information XML

Testers for Regular tree languages, Mdr and Magniez, ICALP 2004

Valeur de l'information: développements en théorie des jeux et en Informatique.

Informatique: Pagerank, valeur relative (Google News),...

Problème: Etant donné une DTD, et un fichier F dans le langage XML, quelle est la distance $\text{dist}(\text{DTD}, F)$?

Versión simplifiée: étant donné une expression régulière R et un mot x, quelle est la distance $\text{dist}(R, x) = \min_{x' \in R} \text{dist}(x', x)$?

X=000111001100

R=0*1* $\text{dist}(R, X)=4$ (2 si déplacement)

Distance d'édition sur les mots

1. Distance d'édition:

Insertions, Effacements, Modifications

2. Distance d'édition avec déplacements (moves)

0111000011110011001

0111011110000011001

3. Distance d'édition avec déplacements sur les Arbres



Testeurs sur une classe K

Soit F une propriété sur une classe K de structures U

Un ϵ -testeur pour F est un algorithme probabiliste tel que:

- Si $U \models F$, A accepte
- Si U est ϵ loin de F, A rejette avec grande probabilité
- Temps(A) indépendant of n.

(Goldreich, Golwasser, Ron 1996, Rubinfeld, Sudan 1994)

Testeur fournit aussi un correcteur en temps linéaire.

Testeurs pour les mots et les arbres

Résultats:

1. Langages réguliers d'arbres sont testables.
2. Equivalence approximative d'automates est polynomiale (version exacte est exponentielle)

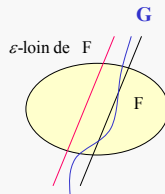
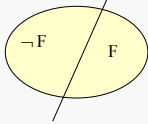
Conséquences:

Validité approximative de fichiers XML peut être testée en $O(1)$
Fichiers XML peuvent être corrigés en temps $O(n)$.

Satisfaisabilité et Equivalence approchés

1. Satisfaisabilité: $T \models F$
2. Satisfaisabilité approchée $Tree \models_{\epsilon} F$
3. Equivalence approchée $F \equiv_{\epsilon} G$

Classe K d'arbres



Approximations

1. Fonction $f(x) \in [0, 1]$
 $f(x)(1 - \epsilon) \leq A(x) \leq f(x)(1 + \epsilon)$

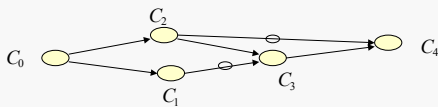
($\epsilon, \log n$) Approximation

- si $f(x) < \delta \Rightarrow f(x)(1 - \epsilon) \leq A(x) \leq f(x)(1 + \epsilon)$
si $f(x) \geq \delta \Rightarrow f(x)(1 - \epsilon \cdot \log n) \leq A(x) \leq f(x)(1 + \epsilon \cdot \log n)$

2. Décision $f(x) \in \{0, 1\}$
 - Décider si $f(x) = 1$
 - Décider si x est ϵ -loin de $f(x) = 1$

Testers on words

L is a regular language and A an automaton for L.



Admissible $Z = C_0 \cdot C_1 \cdot C_2 \cdot C_3 \cdot C_4$
init accept

A word W is Z-feasible if there are two states

$q \in C_i, q' \in C_j$ such that $q \xrightarrow{W} q'$ and $Z = \dots C_i \dots C_j \dots$

The Tester

Tester. Input : W, A, ϵ

For $i=1, \dots, \log(m/\epsilon)$

Choose $N_i = \Theta(2^i \cdot m^2 / \epsilon)$ random subwords w_j^i of size 2^{i+1}

For every admissible path Z:

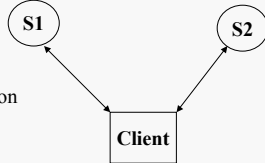
If all w_j^i of W are Z feasible, ACCEPT.
else REJECT.

Theorem: Tester(W, A, ϵ) is an ϵ -tester for L(A).

Jeux et valeur de l'information

Mesures classiques: Pagerank, ...

Généralisation de la distance pour une DTD arbitraire.
Applications: P2P, data-exchange, ...



Equilibres: définition d'une solution.

Calcul: approximer l'équilibre (Maxsat)

Conclusion

Partie 1

1. Exemples de Jeux et de Mécanismes
2. Jeux à somme nulle
3. Jeux matriciels: équilibres de Nash
4. Calcul des équilibres: Lemke-Howson

Partie 2: Importance des Modèles de calcul

1. Approximation d'équilibres

Equilibres approchés (classiques) à N joueurs de support faible

1. Calcul polynomial d'équilibres de marché

Non récursif (modèle numérique) ou P (modèle BSS)

2. Valeur relative pour XML

NP-dur, non-approximable (classique), approximable (testeurs)