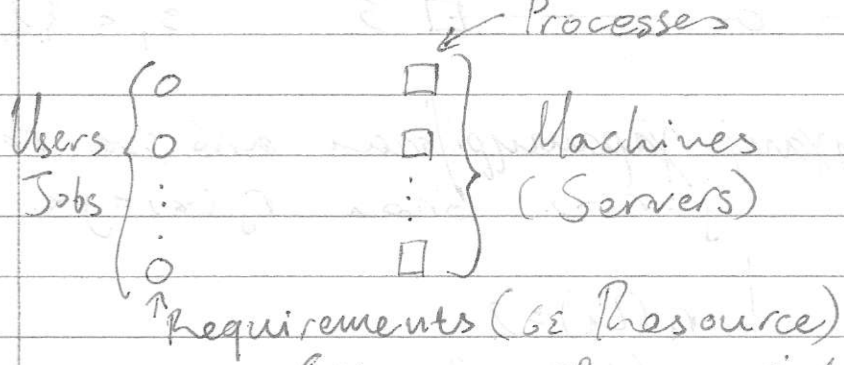
9/11/2010 7ο Μάθημα Μπούγλος Σπύρος (I)

Παίγνια Εξισορρόπησης Φορτίου — Load Balancing Games.



p_i: size (χαρακτηρίζει το job)
ή w_i: weight.

Load Balancing Games
- Splittable (non-atomic)
- Non-Splittable (atomic)

<u>Objective</u>: Προσπαθεί να μην είναι κανένας server υπερφορτωμένος (με υπερβολικό φόρτο) μοιράζοντάς τα σε πολλούς servers $\min\max_i L_i$

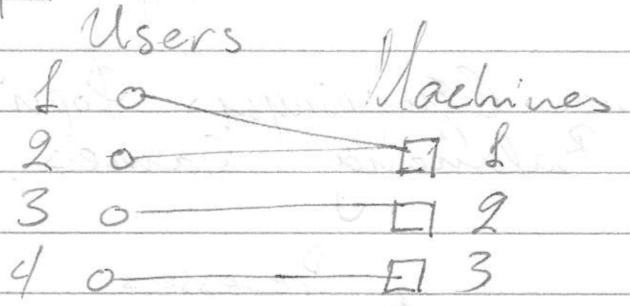
(Virtualization $\rightsquigarrow$ προσπαθούμε να χρησιμοποιούνται όσο το δυνατόν λιγότεροι servers για εξοικονόμηση ενέργειας).

User $i \rightsquigarrow S_i \subseteq \{1, \dots, m\}$ (Ένας χρήστης μπορεί να εξυπηρετηθεί από ένα υποσύνολο των μηχανών)

Feasible assignment A:
$\forall$ job i $\exists j \in S_i : (i,j) \in A$.

$L_j = \sum_{i:(i,j)\in A} p_i \rightsquigarrow$ Συνολικός φόρτος μηχανής.

Παράδειγμα:



$S_1 = \{1\}$
$S_2 = \{1,2\}$
$S_4 = \{1,2,3\}$

Κάθε μηχανή χαρακτηρίζεται από ένα response time $j : r(x)$, όπου $r_j(x) = j \cdot x$.

$L_1 = 2$ | $r_1(L_1) = 2$
$L_2 = 1$ | $r_2(L_2) = 2$
$L_3 = 1$ | $r_3(L_3) = 3$

Μια ανάθεση A λέγεται N.E. όταν $\forall\, i : (i,j) \in A$, $\forall\, k \in S_i$

$$r_j(L_j) \le r_k(b_k + P_i)$$

(Δηλ. αν πάει από τη μηχανή j στη μηχανή k θα αυξηθεί ο φόρτος της k μηχανής κατά P_i και ο response time θα είναι μεγαλύτερος από ότι στο j).

Παράδειγμα: unsplittable (atomic)

1 o □ 1 $r_1(x) = 2$ (για κάθε φορτίο)

2 o □ 2 $r_2(x) = x$

Τα πιθανά N.E. είναι

(II)

$\{(1,1), (2,2)\}$ ✓
$\{(1,2), (2,1)\}$ ✓
$\{(1,1), (2,1)\}$ X Δεν είναι N.E.
$\{(1,2), (2,2)\}$ ✓ ← Είναι σαν το Pigou.

Για την πρώτη περίπτωση: $r_1(x_1) \leq r_2(x_2+1)$
$r_2(x_2) \leq r_1(x_1+1)$.

$$\left.\begin{array}{l} r_1(x_1) \leq r_2(x_2+1) \Rightarrow 2=2 \\ r_2(x_2) \leq r_1(x_1+1) \Rightarrow 1 \leq 2 \end{array}\right\} \Rightarrow \text{Είναι N.E.}$$

Τώρα το social objective είναι
minimize $\max\limits_{j: L_j \leq 0} r_j(L_j)$ ⇝ makespan

minimize $\sum\limits_{i:(i,j)\in A} r_j(L_j)$

minimize $\sum\limits_j L_j r_j(L_j)$.

$$L_j = \sum_{i:(i,j)\in A} \frac{P_i}{S_j} \rightsquigarrow \text{Speed.}$$

(Όταν έχω $P_i = 1$ η συνάρτηση δυναμικού (Potential) $\Phi = \sum\limits_{i=1} \sum\limits_{j=1} r_i(\xi)$ ελαχιστοποιείται από ένα assignment όταν είναι N.E.)

---

fluid (splittable) ⇝ Στο χαρτί το Load Balancing (Eva Tardos)

$r_i(\ )$
j: job $\quad P_j \quad i \rightsquigarrow$ μηχανή

$x_{ij} : j \to i$

j : job, P_j, $S_j \subset \{1, \dots, m\}$

Αν $i \notin S_j$, $x_{ij} = 0$ (Δλδ αν το i δεν μπορεί να ανατεθεί στο S_j)

Φόρτος $L_j = \sum_j x_{ij}$

<u>Παραδείγματα</u>

N M

$P_1 = 1$ ● $\longrightarrow$ □ $r_1(x) = \varepsilon$

$P_2 = 1$ ● $\longrightarrow$ □ $r_2(x) = x$

$X_{N\times M}$ είναι N.E. $\Leftrightarrow \forall i,j : x_{ij} > 0 \ \forall k \in S_j$:

$r_i(L_i) \le r_k(L_k)$ $\left(r_i(L_i) \le r_k(L_k + \cancel{P_i}^{\,0})\right)$

$\left(r_i(L_i) \le r_k(L_k) \sim r_i(L_i) = a\right)$.

N.E. και τα δύο πάνε στο 2 (κάτω)

Makespan: $r = \max\limits_{i: L_i > 0} r_i(L_i)$

<u>Θεώρημα</u>: Ένα N.E. X στο non-atomic (splittable) Load Balancing Game minimizes the max response time over all solutions.

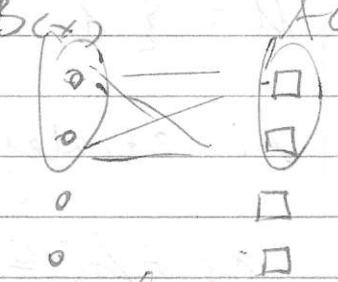
Έστω $X_{i,j}$ (πίνακας) N.E. $i : r_i(L_i) = \max\limits_{j \in M} r_j(L_j)$

$A(x) = \{$i: machines με το μεγαλύτερο response time$\}$

$B(x) = \{$jobs $j : X_{ij} > 0$ για κάποια $i \in A(x)\}$.

Για το N.E. X, $j \in B(x)$ έχουν $\boxed{S_j \subseteq A(x)}$

(Στέλνουνε στο X γιατί δεν έχουν κάτι καλύτερο να κάνουνε).

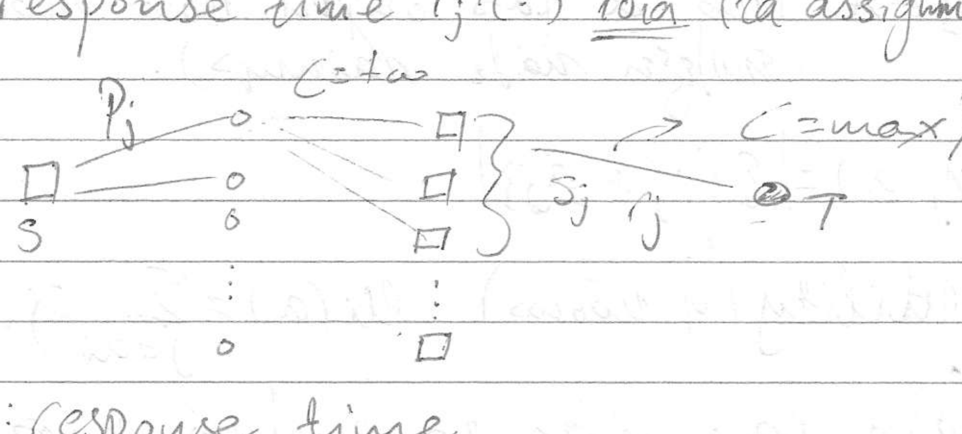


Έστω $\exists\, j \in B(x)$: αναθέτει load στην $k \notin A(x)$
$r_k(L_k) < r_j$

Το X είναι N.E. σημαίνει ότι $\exists$ μηχανή k: $r_e(L_k) \leq r_i(L_i)$, άρα δε μπορεί να είναι (N.E. το X $\rightarrow$ άτοπο).

$\forall$ N.E. το r είναι το μικρότερο δυνατό.

Από αυτό προκύπτει ότι $\forall$ N.E. X το r έχει την ίδια τιμή $\Longrightarrow$
response time $r_j(\cdot)$ <u>ίδια</u> (τα assignments αλλάζουν)



r: response time

$\exists$ λύση με response time $\leq r$ (LB)? $\Longleftrightarrow$
$\exists$ flow $\sum P_j$ (1)

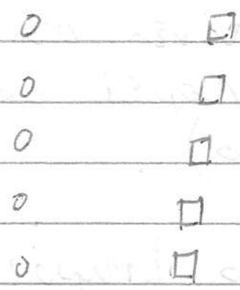
Congestion Games

$N = \{1, \dots, n\}$ n παίκτες
$M = \{1, \dots, m\}$ resources

A_i strategy set του παίκτη i

$a_i \in A_i \subseteq$ (Δυναμοσύνολο του M) ⇝ δηλ.

όλα τα δυνατά υποσύνολα



$\forall j \in M$
$c_j(k)$

$A = \times A_i$ (Strategy space όλου του συστήματος)

$\underline{a} \in A$ (το a μας δείχνει ποιο υποσύνολο έχει επιλέξει κάθε παίκτης).

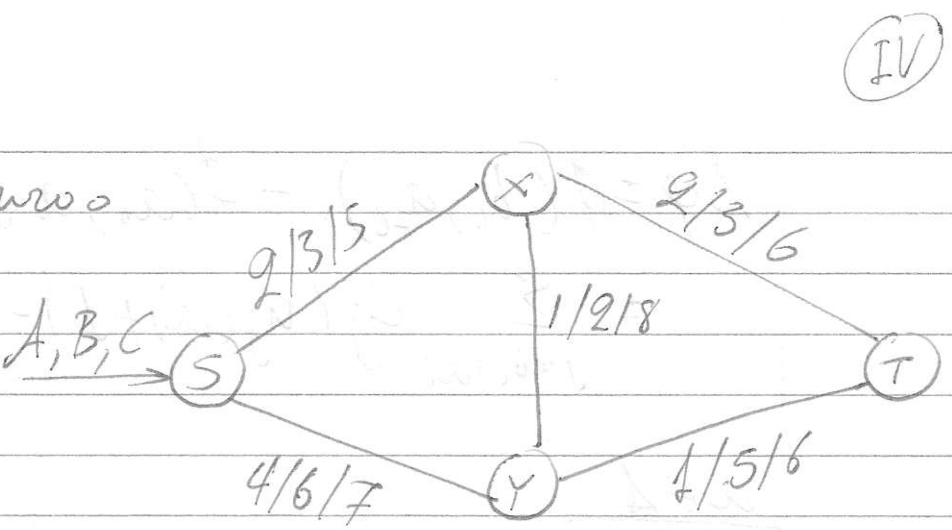
$n_j(a) = |\{i : j \in \underline{a}_i\}|$

utility (ή κόστος) $u_i(a) = \sum_{j \in a_i} c_j(n_j(a))$

(δηλ. ποιο είναι το συνολικό κόστος σε όλα τα resources που χρησιμοποιεί, όπου παίζουν ρόλο και οι άλλοι).

(IV)

Έστω δίκτυο



a/b/γ ⇝ για έναν / για δύο / ή τρεις παίκτες
Για το κοινό resourse το κόστος είναι ίδιο.

<u>Θεώρημα</u>: Κάθε congestion game έχει τουλάχιστον ένα pure Nash Equilibrium

Potential function
$a \in A$ $\qquad \Phi(a) = \sum_{j=1}^{m} \sum_{k=1}^{n_j(a)} c_j(k)$.

$\Phi : A \to \mathbb{R}$

Όταν ένας παίκτης i αλλάζει τη στρατηγική του από a_i σε b_i, θα αποδείξουμε ότι η διαφορά στο utility είναι $\Delta u = \Delta \Phi$

$i : a_i \to b_i \qquad \Delta u = \Delta \phi$

$\Delta u_i = u_i(b_i, a_{-i}) - u_i(a_i, a_{-i}) =$

= (κόστος για τη χρήση νέων resources) − (κόστος από τη μη χρήση resources που χρησιμοποιούσε πριν)

$= \sum_{j \in b_i \setminus a_i} c_j(n_j(a)+1) - \sum_{a_i \setminus b_i} c_j(n_j(a))$

$$\Delta\Phi = \Phi(b_i, a_{-i}) - \Phi(a_i, a_{-i}) =$$

$$= \sum_{j \in b_i / a_i} c_j(n_j(a)+1) - \sum_{a_i / b_i} c_j(n_j(a))$$

<u>$a \in A$</u>

Κάθε παίκτης (σε αυθαίρετη σειρά και αυθαίρετο αριθμό φορών) $b_i \leftarrow a_i$

Κάνει best response $a_i \rightarrow b_i$

Κάθε φορά το Φ μειώνεται ($\Phi\downarrow$) και επειδή το Φ παίρνει πεπερασμένο αριθμό τιμών τελικά φτάνει σε <u>τοπικό ελάχιστο</u> και έχουμε N.E.

(κανένας παίκτης δεν μπορεί να βελτιωθεί περαιτέρω)