Algoritmi paraleli
Curs 8

Vlad Olaru

vlad.olaru@fmi.unibuc.ro

Calculatoare si Tehnologia Informatiei

Universitatea din Bucuresti

Algoritmi asincroni
-Metode iterative-

Algoritmul Jacobi sincron

$$Ax = b$$

Este echivalent cu

$$x_1 = -\frac{1}{A_{11}}(A_{12}x_2 + \cdots + A_{1n}x_n - b_1)$$

$$x_2 = -\frac{1}{A_{22}}(A_{21}x_1 + \cdots + A_{1n}x_n \ - b_2)$$

$$\dots$$

$$x_n = -\frac{1}{A_{nn}}(A_{n1}x_1 + \cdots + A_{nn-1}x_{n-1} \ - b_n)$$

Algoritmul Jacobi sincron

$$Ax = b$$

La pasul $t + 1$:

$$x_1(t+1) = -\frac{1}{A_{11}}(A_{12}x_2(t) + \cdots + A_{1n}x_n(t) - b_1)$$

$$x_2(t+1) = -\frac{1}{A_{22}}(A_{21}x_1(t) + \cdots + A_{1n}x_n(t) \ - b_2)$$

$$\dots$$

$$x_n(t+1) = -\frac{1}{A_{nn}}(A_{n1}x_1(t) + \cdots + A_{nn-1}x_{n-1}(t) \ - b_n)$$

Algoritmul Jacobi sincron

Ideea algoritmului Jacobi sincron:

$$x_i(t+1) = -\frac{1}{A_{ii}}\left(\sum_{j\neq i} A_{ij}x_j(t) - b_i\right)$$

$$Echivalent: [x(t+1) = D^{-1}\big(b - Rx(t)\big)]$$

Sincronizare:

Pentru calculul $x_i(t+1), P_i$ asteapta $x_j(t)$ de la P_j, unde $j = 1, \ldots, p, j \neq i$

Este necesar ca fiecare procesor sa aiba $x(t)$!

Algoritmul Jacobi sincron

- O matrice A este **diagonal dominanta** (pe linii) daca:

$$\sum_{j \neq i} |A_{ij}| < |A_{ii}|$$

- Exemplu: $\begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix}$ este diagonal dominanta, $\begin{bmatrix} -1 & 2 \\ 2 & -1 \end{bmatrix}$ nu este!

Teorema: Daca A este diagonal dominanta, atunci sirul generat de algoritmul Jacobi:

$$x_i(t+1) = -\frac{1}{A_{ii}}\left(\sum_{j \neq i} A_{ij} x_j(t) - b_i\right)$$

converge.

Algoritmi asincroni

- dorim paralelizarea unui algoritm pe o arhitectura in care **fiecare procesor ruleaza un (sub)algoritm local.**
- algoritmii locali nu sunt constransi sa astepte la momente predeterminate mesaje predeterminate
- procesoarele ruleaza la viteze diferite => unele executa un nr mai mare de iteratii
- unele procesoare comunica mai des decat altele
- intarzierile mesajelor pot fi substatiale si sunt impredictibile
- mesajele pot fi primite in alta ordine decat cea in care au fost trimise

Avantaje algoritmi asincroni

- reducerea penalizarilor de sincronizare

⇒ castig de viteza comparativ cu algoritmii sincroni

  - acest castig poate fi diminuat de o complexitate sporita a comunicarii

- flexibilitate sporita a implementarii

- toleranta mai mare pentru schimbari ale datelor problemei

Dezavantaje algoritmi asincroni

- conditiile de validitate sunt in general mai stringente decat cele ale algoritmilor sincroni
- detectia terminarii algoritmului e indeobste mai dificila
- comportament particular fata de timpi de comunicare si calcul cu durata arbitrara
  - *algoritmi total asincroni (s.n. si haotici)* – tolereaza intarzieri de comunicatie/calcul oricat de mari
  - *algoritmi partial asincroni* – ruleaza corect (converg) doar in conditiile in care intarzierile sunt limitate

  => mecanismele de convergenta (si analiza lor) sunt fundamental diferite

Modelarea algoritmilor total asincroni

Consideram: $X = X_1 \times X_2 \times \ldots \times X_n$

si

$$x \in X \text{ a.i. } x = (x_1, x_2, \ldots, x_n) \text{ si } x_i \in X_i \ \forall i = 1, \ldots, n$$

Problema: fie functia $f: X \to X, f(x) = \big(f_1(x), f_2(x), \ldots, f_n(x)\big)$ a.i.

$$f_i: X \to X_i \, si \; x_i = f_i(x), \qquad \forall i = 1, \ldots, n$$

Sa se determine punctul fix al lui f , adica $x^* \in X$ a.i. $x^* = f(x^*)$ sau

$$x_i^* = f_i(x_i^*)$$

Notatii:

$x_i(t)$ = valoarea componentei x_i la momentul t.

$T = \{0,1,2, \ldots\}$ setul timpilor la care una sau mai multe componente x_i ale lui x sunt actualizate de un procesor sau altul din sistemul distribuit

T^i = setul momentelor de timp la care este actualizat x_i; $T^1 = \{0,2,5,6, \ldots .\}$

Modelarea algoritmilor total asincroni

Pp procesorul P_i nu are neaparat access la cele mai noi valori ale componentelor lui x pentru actualizarea lui x_i:

$$x_i(t+1) = f_i\left(x_1\left(\tau_1^i(t)\right), x_2\left(\tau_2^i(t)\right), \ldots, x_n\left(\tau_n^i(t)\right)\right), \forall\, t\ \in T^i$$

unde:

$\tau_j^i(t)$ = momentele de timp ale actualizarilor a.i.

$$0 \leq \tau_j^i(t) \leq t, \qquad \forall t \in T.$$

De asemenea, $x_i(t+1) = x_i(t), \quad \forall t \notin T^i$

Modelarea algoritmilor total asincroni - observatii

- T poate fi vazut si ca un set de indecsi al momentelor de timp la care se petrec actualizarile
- multimile T^i (si secventele de timpi fizici asociate) nu sunt necesare tuturor procesoarelor pt ca nu sunt folosite in iteratie => nu e nevoie de ceas global sau de sincronizarea ceasurilor locale ale procesoarelor
- diferenta $t - \tau_j^i(t)$ reprezinta intarzierea de comunicatie intre timpul curent si timpul la care componenta j este disponibila pentru actualizarea $x_i(t)$

Model conceptual util

- P_i se trezeste spontan la momentul $t \in T^i$
- P_i primeste printr-un mechanism oarecare valorile

$$x_1\left(\tau_1^i(t)\right), x_2\left(\tau_2^i(t)\right), \dots, x_n\left(\tau_n^i(t)\right)$$

- P_i actualizeaza x_i
- actualizarea se face fara cunostinta valorilor t, $\tau_1^i(t)$, … $\tau_n^i(t)$ sau a oricarui element din T^j, $j = 1, \dots, n$
- obs: Jacobi, Gauss-Seidel (si variantele bloc iterative) sunt cazuri speciale ale acestui tip de iteratie

Model explicitat

- P_i stocheaza local $x^i(t) = (x_1^i(t), x_2^i(t), \ldots, x_n^i(t))$, pe baza caruia actualizeaza $x_i^i(t)$ la $t \in T^i$ prin relatia:

$$x_i^i(t+1) = f_i(x^i(t))$$

- ocazional (la momente nespecificate), P_i transmite x_i^i celorlalte procesoare
- cand P_j primeste (dupa o intarziere oarecare, nespecificata) noul x_i^i, il memoreaza in componenta i local stocata a lui x_i^j; la acest moment avem

$$x_i^j(t) = x_i^i\left(\tau_i^j(t)\right) \quad \text{iar } (t - \tau_i^j(t)) \text{ este intarzierea de comunicare}$$

De asemenea, in mod natural $\tau_i^i(t) = t$

- Obs: - in general, vectorii locali $x^i(t)$ au valori diferite pe procesoare diferite la acelasi moment de timp t!

- nu exista nici o presupunere asupra ordinii transmiterii mesajelor
- procesoarele nu pot determina daca valorile actualizarilor primite sunt mai vechi decat valorile stocate in memoria locala

- **Scop final**: $x(t) = (x_1^1(t), x_2^2(t), \ldots, x_n^n(t))$ converge catre solutia $x = f(x)$.

Exemplu algoritm total asincroni

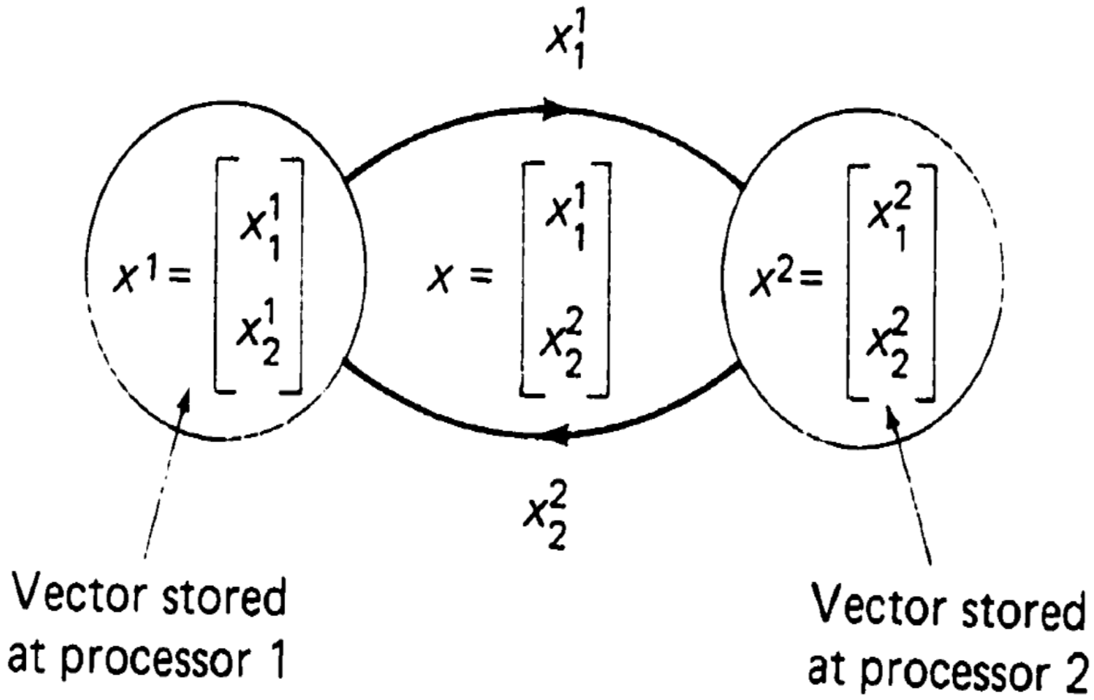
Determinati x *astfel incat*

$$x_1 = 0.5x_1 + 0.5x_2$$
$$x_2 = 0.2x_1 + 0.7x_2$$

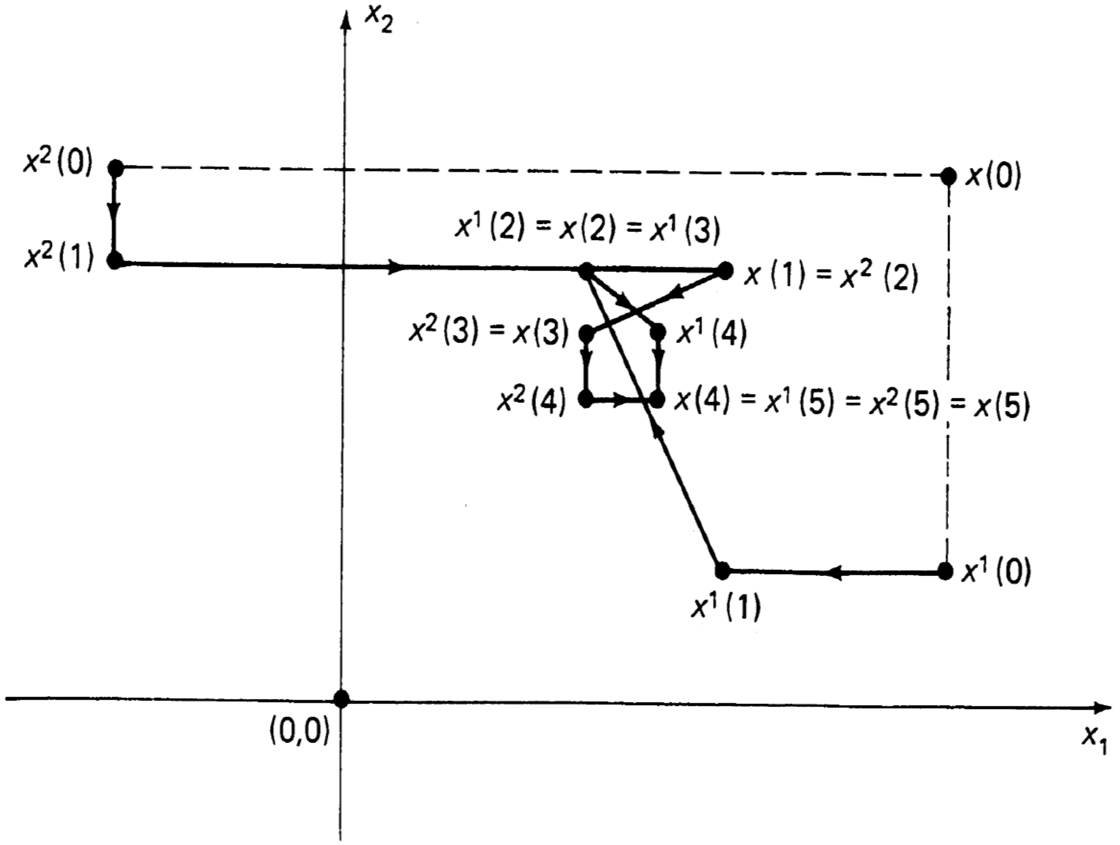
2 procesoare P_1 si P_2:

P_1 stocheaza $x^1 = (x_1^1, x_2^1)$ si actualizeaza x_1^1 la momentele $t \in T^1$

P_2 stocheaza $x^2 = (x_1^2, x_2^2)$ si actualizeaza x_2^2 la momentele $t \in T^2$



Exemplu algoritmi total asincroni



t=0: Processor 1 updates x_1^1 and transmits it to processor 2, where it is received at a time between $t = 1$ and $t = 2$. Processor 2 updates x_2^2 and transmits it to processor 1, where it is received at a time between $t = 1$ and $t = 2$.

t=1: Processor 1 updates x_1^1 and transmits it to processor 2, where it is received at a time between $t = 2$ and $t = 3$. [Processor 2 does not update X_2^2 but X_1^2 will change at some time $t \in (1, 2)$ because of the reception of $X_1^1(1)$; for this reason, $x^2(1) \neq x^2(2)$ as shown in the figure.]

t=2: Processor 2, having received $x_1^1(1)$, updates x_2^2 and transmits it to processor 1, where it is received at a time between $t = 3$ and $t = 4$. [Processor 1 does not update X_1^1 and does not receive a value of X_2^2 from Processor 2 at any time $t \in (2, 3)$; for this reason, $X^1(2) = X^1(3)$ as shown in the figure.]

t=3: Processor 1, having received $x_2^2(1)$ [at some time $t \in (1, 2)$], updates x_1^1 and transmits it to processor 2, where it is received at a time between $t = 4$ and $t = 5$. Processor 2, having received $x_1^1(2)$, updates x_2^2 and transmits it to processor 1, where it is received at a time between $t = 4$ and $t = 5$.

t=4: Processor 1 has already received $x_2^2(3)$; no updating takes place.

t=5: Processor 1 has already received $x_2^2(4)$ and processor 2 has already received $x_1^1(4)$.

Teorema de convergenta asincrona

- Fie secventa de multimi nevide $\{X(k)\}\ a.i.$

$$\ldots \subset X(k+1) \subset X(k) \subset \cdots \subset X$$

si

$$(a)\quad conditia\ de\ convergenta\ sincrona$$

$$f(x) \in X(k+1)\ \forall k\ si\ x \in X(k).$$

In plus, daca $\{y^k\}$ e o secventa a.i. $y^k \in X(k)\ \forall k \Rightarrow$ daca y punct limita al $\{y^k\}$, y punct fix al lui f

si

$$(b)\ box\ condition$$

$$\forall k \text{ exista } X_i(k) \subset X_i \text{ a.i. } X(k) = X_1(k) \times X_2(k) \times \ldots \times X_n(k)$$

Atunci, daca $x(0) = (x_1(0), x_2(0), \ldots, x_n(0)) \in X(0)$

si $\quad \forall\ x^*$ a.i. $\{x(t)\} \to x^*$

$$=> f(x^*) = x^*$$

Iteratii asincrone pt. sisteme de ecuatii liniare

- fie functia

$$f(x) = Ax + b \text{ unde } A \in R^{n \times n} \text{ si } b \in R^n$$

- vrem sa gasim solutia x^* a.i.

$$x^* = Ax^* + b$$

folosind o iteratie asincrona definita astfel

$$x_i(t+1) = \sum_{j=1}^{n} A_{ij} x_j\left(\tau_j^i(t)\right) + b_i \text{ , } t \in T^i$$

$$x_i(t+1) = x_i(t), \quad t \notin T^i$$

Conditie de convergenta iteratie

- **Propozitie**: Fie $A \in R^{n \times n}$ a.i. $I - A$ inversabila. Atunci

(i) $\rho(|A|) < 1$ unde $\rho(A) = \max_{1 \le i \le n} |\lambda_i(A)|$ e *raza spectrala* a matricii A

$\Leftrightarrow$

$(ii) \forall\ x(0)\ conditie\ initiala$, $\forall b \in R^n$ si

$\forall\ T^i$ a.i. fiecare T^i e infinita, $\forall\ \tau_j^i(t)$ a.i. $t - 2 \le \tau_j^i(t) \le \mathrm{t}$, $\forall t$

secventa $\{x(t)\}$ converge la $(I - A)^{-1} b$

Concluzie: "intarzieri" $t - \tau_j^i(t)$ de doar doua unitati de timp pot induce divergenta cand $\rho(|A|) \ge 1$

Algoritmi iterativi partial asincroni

- dorim paralelizarea unui algoritm pe o arhitectura in care **fiecare procesor ruleaza un (sub)algoritm local.**
- am retinut ca nu toti algoritmii total asincroni converg
- pt. anumiti algoritmi asincroni exista posibilitatea de a “redeveni” convergenti daca se impun anumite constrangeri asupra gradului de asincronism
- pp. ca exista o constanta B (numita masura asincronismului) a.i.

(a) fiecare procesor executa o actualizare cel putin o data pe durata unui interval de timp de durata B

(b) informatia folosita de orice procesor nu este mai veche de B unitati de timp

Un algoritm asincron care satisface (a) si (b) s.n. *partial asincron*

Algoritmi iterativi partial asincroni

- analiza convergentei algoritmilor partial asincroni nu se poate reduce la un singur rezultat general de tipul *teoremei de convergenta asincrona* pt. algoritmi total asincroni
- exista doua tipuri de algoritmi partial asincroni

(a) algoritmi care nu converg total asincron, dar executia lor partial asincrona converge pt. orice valoare finita a lui B

(b) algoritmi care converg partial asincron pt valori suficient de mici ale lui B, iar pt valori mari ale lui B diverg

Modelarea algoritmilor partial asincroni

Consideram: $X = X_1 \times X_2 \times \ldots \times X_n$ unde X_i sunt submultimi de spatii Euclidiene

si

$$x \in X \text{ a.i. } x = (x_1, x_2, \ldots, x_n) \text{ si } x_i \in X_i \; \forall i = 1, \ldots, n$$

Problema: fie functia $f: X \to X, f(x) = \big(f_1(x), f_2(x), \ldots, f_n(x)\big)$ a.i.

$$f_i: X \to X_i \;\; si \; x_i = f_i(x), \qquad \forall i = 1, \ldots, n$$

Executia iteratiei de mai sus e complet determinata de:

(a) Conditiile intiale $x(t) \in X \; \forall t \leq 0$

(b) T^i = multimea momentelor de timp la care este actualizata componenta x_i

(c) variabilele $\tau_j^i(t) \;\; \forall i, j$ si $t \in T^i$ care:

- reprezinta durata de timp cu care informatia folosita la actualizarea x_i este invechita

- satisfac conditia $0 \leq \tau_j^i(t) \leq t$

Ecuatii algoritmi partial asincroni

$$\forall t \geq 0$$

$$x_i(t+1) = f_i\left(x_1\left(\tau_1^i(t)\right), x_2\left(\tau_2^i(t)\right), \dots, x_n\left(\tau_n^i(t)\right)\right), \text{ daca } t \in T^i$$

$$x_i(t+1) = x_i(t), \quad \forall t \notin T^i$$

- o alegere particulara a multimilor T^i si a valorilor variabilelor $\tau_j^i(t)$ s. n. *scenariu*
- Obs: - pt. orice scenariu fixat, valoarea lui $x(t)$ $pt. t > 0$ e determinata unic de conditiile intiale

- pt scenarii diferite, valorile pot diferi

Presupuneri fundamentale asincronism partial

$\exists\, B \in N$ a.i.

(a) $\forall\, i\ si\ t \geq 0$ cel putin un element al multimii $\{t, t+1, \ldots, t+B-1\}$ apartine T^i

(b) $t - B < \tau_j^i(t) \leq t,\ \forall\, i, j$ si $\forall\, t \geq 0, t \in T^i$

(c) $\tau_i^i(t) = t\ \forall\, i$ si $t \in T^i$

Obs: variabila t nu e accesibila procesoarelor

- t poate fi vazut ca o variabila de timp masurata de un observator extern
- t nu corespunde in mod necesar timpului real, poate fi doar o variabila care indexeaza timpul evenimentelor de interes (eg, actualizarea variabilelor)

Exemplu algoritm partial asincron care converge pt orice valoare a lui B

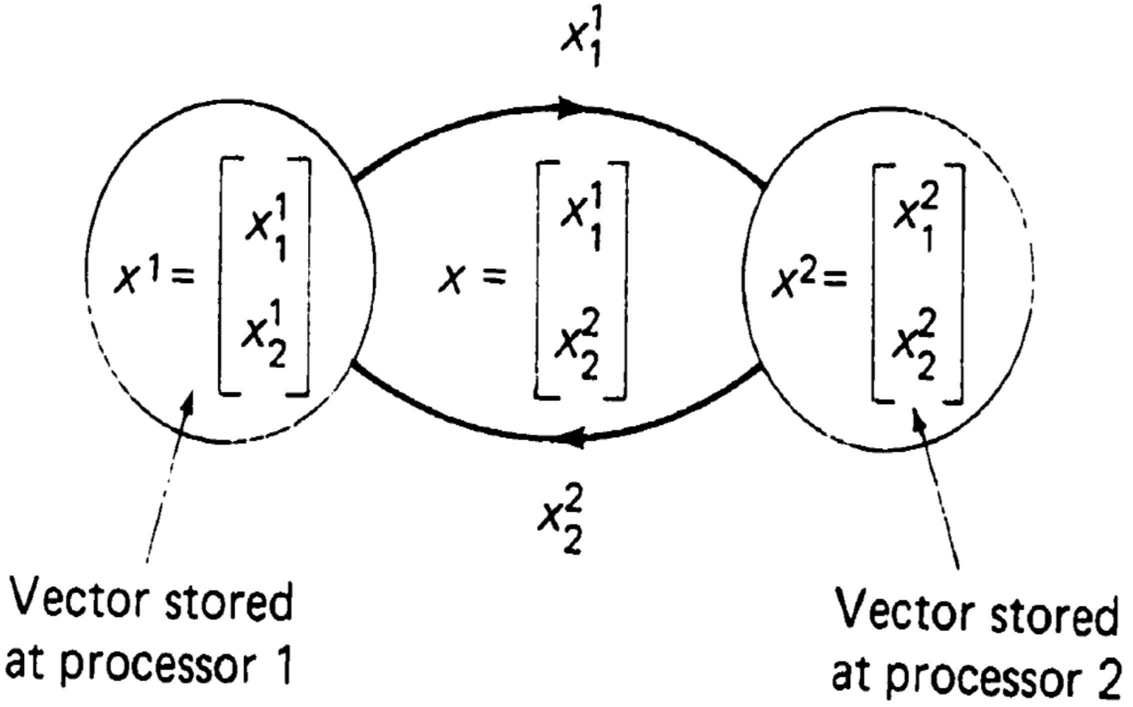
Fie iteratia $x = Ax$

$$x_1 = \frac{1}{2}x_1 + \frac{1}{2}x_2$$
$$x_2 = \frac{1}{2}x_1 + \frac{1}{2}x_2$$

2 procesoare P_1 si P_2:

P_1 stocheaza $x^1 = \left(x_1^1, x_2^1\right)$ si actualizeaza x_1^1 la momentele $t \in T^1$

P_2 stocheaza $x^2 = \left(x_1^2, x_2^2\right)$ si actualizeaza x_2^2 la momentele $t \in T^2$



Exemplu algoritm partial asincron care converge pt orice valoare a lui B

- Obs: multimea X^* a punctelor fixe ale lui $f(x) = Ax$ este multimea vectorilor $(x_1, x_2) \in R^n$ a. i. $x_1 = x_2$
- sincron, iteratia $x(t+1) = Ax(t)$, ajunge dupa 1 pas la solutie !

$$x(1) = \begin{bmatrix} \frac{1}{2}(x_1(0) + x_2(0)) \\ \frac{1}{2}(x_1(0) + x_2(0)) \end{bmatrix}$$

Exemplu algoritm partial asincron care converge pt orice valoare a lui B

Scenariu:

- 2 procesoare P_1, P_2 actualizeaza variabila corespunzatoare la fiecare pas
- procesoarele comunica variabila lor la anumite momente $\{t_1, t_2, \dots\}$
- transmiterea/folosirea informatiei comunicate se face instant

=> iteratia este

$$x_1(t+1) = \frac{x_1(t)}{2} + \frac{x_2(t_k)}{2}, \qquad t_k \leq t < t_{k+1}$$

$$x_2(t+1) = \frac{x_1(t_k)}{2} + \frac{x_2(t)}{2}, \qquad t_k \leq t < t_{k+1}$$

Exemplu algoritm partial asincron care converge pt orice valoare a lui B

$$x_1(t+1) = \frac{x_1(t)}{2} + \frac{x_2(t_k)}{2}, \qquad t_k \le t < t_{k+1}$$

- intre t_k si t_{k+1}, P_1 mentine $x_2(t_k)$ constant si executa iteratie de mai sus de $t_{k+1} - t_k$ ori:

$$x_1(t_k+1) = \tfrac{1}{2} x_1(t_k) + \tfrac{1}{2} x_2(t_k)$$

$$x_1(t_k+2) = \frac{1}{2} x_1(t_k+1) + \frac{1}{2} x_2(t_k)$$

$$= \frac{1}{2}\left[\frac{1}{2} x_1(t_k) + \frac{1}{2} x_2(t_k)\right] + \frac{1}{2} x_2(t_k) = \boldsymbol{\frac{1}{4} x_1(t_k)} + \left[\boldsymbol{\frac{1}{2}} + \boldsymbol{\frac{1}{4}}\right] \boldsymbol{x_2(t_k)}$$

Exemplu algoritm partial asincron care converge pt orice valoare a lui B

$$x_1(t_k+2)=\frac{1}{2}x_1(t_k+1)+\frac{1}{2}x_2(t_k)$$

$$=\frac{1}{2}\left[\frac{1}{2}x_1(t_k)+\frac{1}{2}x_2(t_k)\right]+\frac{1}{2}x_2(t_k)=\mathbf{\frac{1}{4}}\boldsymbol{x_1(t_k)}+\left[\mathbf{\frac{1}{2}}+\mathbf{\frac{1}{4}}\right]\boldsymbol{x_2(t_k)}$$

$$x_1(t_k+i)=\left(\frac{1}{2}\right)^i x_1(t_k)+\left[\frac{1}{2}+\frac{1}{4}+\cdots+\left(\frac{1}{2}\right)^i\right]x_2(t_k)$$

Algoritmi asincroni - exemplu

$$x_1(t+1) = \frac{x_1(t)}{2} + \frac{x_2(t_k)}{2}, \qquad t_k \le t < t_{k+1}$$

- intre t_k si t_{k+1}, P_1 mentine $x_2(t_k)$ constant si executa iteratia de mai sus de $t_{k+1} - t_k$ ori:

$$x_1(t_{k+1}) = \left(\frac{1}{2}\right)^{t_{k+1}-t_k} x_1(t_k) + \left(1 - \left(\frac{1}{2}\right)^{t_{k+1}-t_k}\right) x_2(t_k)$$

Exemplu algoritm partial asincron care converge pt orice valoare a lui B

$$x_1(t_{k+1}) = \left(\frac{1}{2}\right)^{t_{k+1}-t_k} x_1(t_k) + \left(1 - \left(\frac{1}{2}\right)^{t_{k+1}-t_k}\right) x_2(t_k)$$

$$x_2(t_{k+1}) = \left(\frac{1}{2}\right)^{t_{k+1}-t_k} x_2(t_k) + \left(1 - \left(\frac{1}{2}\right)^{t_{k+1}-t_k}\right) x_1(t_k)$$

Exemplu algoritm partial asincron care converge pt orice valoare a lui B

$$x_1(t_{k+1}) = \left(\frac{1}{2}\right)^{t_{k+1}-t_k} x_1(t_k) + \left(1 - \left(\frac{1}{2}\right)^{t_{k+1}-t_k}\right) x_2(t_k)$$

$$x_2(t_{k+1}) = \left(\frac{1}{2}\right)^{t_{k+1}-t_k} x_2(t_k) + \left(1 - \left(\frac{1}{2}\right)^{t_{k+1}-t_k}\right) x_1(t_k)$$

Deci:

$$|x_2(t_{k+1}) - x_1(t_{k+1})| \leq (1 - \epsilon_k)|x_2(t_k) - x_1(t_k)|$$

$$\leq \prod_k (1 - \epsilon_k)\, |x_2(0) - x_1(0)|,$$

unde $\epsilon_k = 2\left(\frac{1}{2}\right)^{t_{k+1}-t_k}$

Exemplu algoritm partial asincron care converge pt orice valoare a lui B

Deci:

$$|x_2(t_{k+1}) - x_1(t_{k+1})| \leq (1-\epsilon_k)|x_2(t_k) - x_1(t_k)|$$

$$\leq \prod_k (1-\epsilon_k)\,|x_2(0) - x_1(0)|$$

Convergenta catre un vector din X^* e garantata doar daca

$$\prod_k (1-\epsilon_k) = 0$$

Dar, daca alegem diferentele $t_{k+1} - t_k$ suficient de mari a.i. $\epsilon_k < k^{-2}$

$\Rightarrow \prod_k (1-k^{-2}) > 0$

$\Rightarrow$ **$x = Ax$ nu converge total asincron !**

Exemplu algoritm partial asincron care converge pt orice valoare a lui B

Daca impunem presupunerea de asincronism partial din scenariu, avem

$$t_{k+1} - t_k \leq B$$

$\Rightarrow$ $$\epsilon_k > \left(\frac{1}{2}\right)^B \Rightarrow \prod_k (1 - \epsilon_k) = 0$$

$\Rightarrow$ $|x_2(t_{k+1}) - x_1(t_{k+1})|$ converge la zero

$\Rightarrow$ secventa $\{x(t_k)\}$ converge la X^*

$\Rightarrow$ conditia de convergenta e indeplinita indiferent de valoarea lui B (presupunerea din scenariu e ca B e numar intreg pozitiv)

$\Rightarrow$ **algoritm partial asincron convergent, fara a fi total convergent !**

Consecinte privitoare la structura iteratiilor partial asincrone

Daca avem pt $\forall t \geq 0$

$$x_i(t+1) = f_i\left(x_1\left(\tau_1^i(t)\right), x_2\left(\tau_2^i(t)\right), \dots, x_n\left(\tau_n^i(t)\right)\right), \text{ pt } t \in T^i$$

$$x_i(t+1) = x_i(t), \quad \forall t \notin T^i$$

si $\exists\, B \in N$ a.i.

(a) $\forall\, i\ si\ t \geq 0$ cel putin un element al multimii $\{t, t+1, \dots, t+B-1\}$ apartine T^i

(b) $t - B < \tau_j^i(t) \leq t$, $\forall\, i, j$ si $\forall\, t \geq 0, t \in T^i$

(c) $\tau_j^i(t) = t\ \forall\, i$ si $t \in T^i$

=> $\boldsymbol{x(t+1)}$ depinde doar de $\boldsymbol{x(t)}$, $\boldsymbol{x(t-1)}$, …, $\boldsymbol{x(t-B+1)}$ si nu de informatie anterioara $\boldsymbol{x(\tau)}$, cu $\boldsymbol{\tau \leq t - B}$!

Reformulare

- daca informatia veche este eliminata din sistem dupa maximum B unitati de timp, putem introduce vectorul

$$z(t) = (x(t), x(t-1), \ldots, x(t-B+1))$$

care sumarizeaza informatia utila (ne-eliminata inca)

- pt. orice scenariu, $x(t+1)$ se poate determina din $z(t)$
- pe cale inductiva, pt orice scenariu fixat si orice intreg pozitiv s, valoarea $z(t+s)$ este unic determinata de $z(t)$
- de asemenea, daca f e continua, pt orice scenariu fixat si orice intreg pozitiv s, $z(t+s)$ este o functie continua de $z(t)$

Convergenta generalizata algoritmi partial asincroni

- fie $X* = \{x \in X | x = f(x)\}$ multimea punctelor fixe ale lui f
- Z produsul cartezian al B copii ale lui X
- $Z* = \{(x^*, x^*, \ldots, x^*) \mid x^* \in X^*\}$ => daca $z(t) = z* \in Z*$ atunci

$$z(t+s) = z* \ \ \forall s > 0 \text{ si orice scenariu}$$

- daca definim si o distanta d: $Z -> [0, \infty)$ care masoara progresul $z(t)$ catre $Z*$ putem defini urmatorul rezultat valabil pt algoritmi partial asincroni definiti de ecuatiile

$$x_i(t+1) = f_i\left(x_1\left(\tau_1^i(t)\right), x_2\left(\tau_2^i(t)\right), \ldots, x_n\left(\tau_n^i(t)\right)\right), \text{ pt } t \in T^i$$

$$x_i(t+1) = x_i(t), \quad \forall t \notin T^i$$

Teorema Lyapunov

- ipoteze: f continua si presupunerile asincronismului partial sunt indeplinite
- Pp. $\exists\ t^* \in N$ si d: $Z -> [0, \infty)$ functie continua cu urmatoarele proprietati

(a) $\forall\ z(0) \notin Z^*$ si orice scenariu, avem $d(z(t^*)) < \mathrm{d}\big(\mathrm{z}(0)\big)$

(b) $\forall\ z(0) \notin Z$ si $\forall\ t \geq 0$ si orice scenariu, avem

$$d\big(z(t+1)\big) \leq d\big(z(t)\big)$$

$\Rightarrow z^* \in Z^*$ pt. $\forall\ \ z^* \in Z$ punct limita al secventei $\{z(t)\}$

Obs: - rezultat general, nu vizeaza viteza de convergenta

- prea general pt a avea implicatii practice utile

- se poate demonstra insa ca pentru algoritmi interativi liniari convergenta este geometrica

Algoritmi paraleli – concluzii

- arhitecturi paralele (taxonomia Flynn) si modele de programare paralela (SPMD)
- topologii interconectare procesoare (inel, grid, hipercub), complexitatea comunicarii, mapari inter-topologii
- masuri de performanta (speed-up, eficienta) si limitari (Amdahl, Gustafson)
- analiza algoritmi paraleli: raportul optimal comunicare/calcul, incarcarea echilibrata a procesoarelor, penalizarea sincronizarii, detectia terminarii
- clase generale de algoritmi: sincroni vs asincroni