Problematiche di scheduling

- gestione fair dell'accesso
- accesso esclusivo
- presenza di una sola risorsa condivisa

La soluzione più semplice è uno scheduler round-robin

Gestione centralizzata

- clock globale ed un complesso arbitraggio centralizzato
- possibile spreco dei quanti di tempo o negoziazione esplicita per rispondere a esigenze di accesso variabili nel tempo
- negoziazione esplicita per gestire il churn
- strategia esplicita di fault detection

Gestione decentralizzata

- nessun clock e nessun arbitraggio
- stato locale e un algoritmo semplice eseguito uniformemente da tutti i nodi
- gestione totalmente dinamica dell'accesso: i nodi ricreano costantemente un perfetto round-robin, gestendo implicitamente fault detection e churn

Proposte di soluzione decentralizzata

- due varianti dell'algoritmo DESYNC, che si ispirano alla diffusione di un gradiente di concentrazione non uniforme in una soluzione
- algoritmo INVERSE-MS, basato sulla mutua sincronizzazione (inversa) ed ispirato al lampeggiare delle lucciole

Applicazione: Reti wireless Time Division Multiple Access (TDMA)

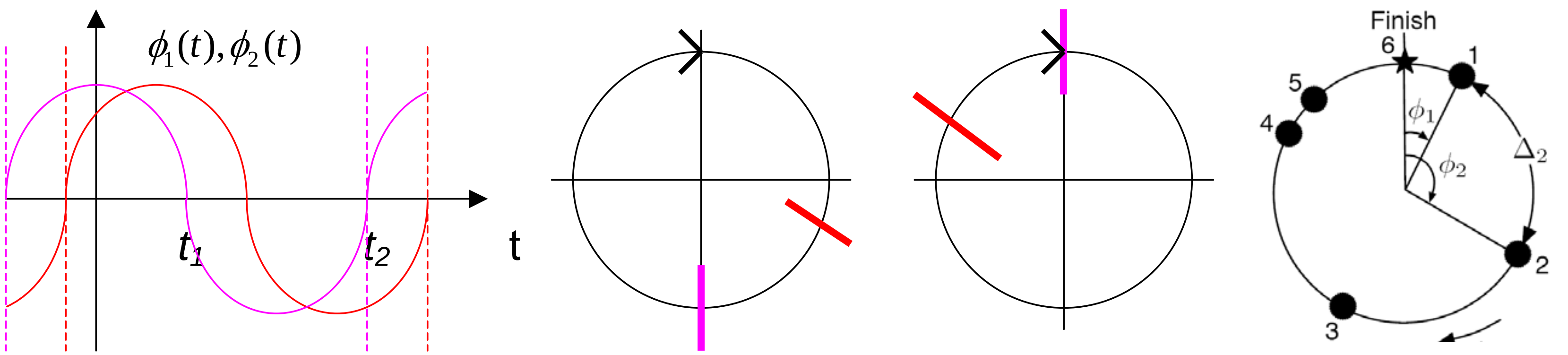
- l'802.11 è un semplice protocollo di contesa per l'accesso
- il protocollo TDMA applica un round-robin ma può sprecare bandwidth
- serve un protocollo che massimizzi la bandwidth in condizioni di high load

Applicazione: Sensori e Analog Digital Converter (ADC)

- un ADC deve mantenere dinamicamente un interleave uniforme fra gli istanti di campionamento
- una rete di sensori (specialmente in ambito critico) deve desincronizzare la frequenza di campionamento, aumentando così anche la copertura

Applicazione: Intersezioni di traffico

- dotare le auto di un meccanismo di desincronizzazione che massimizzi il throughput e minimizzi il tempo di attesa nell'utilizzo delle intersezioni



(a) (b) (c) (d)

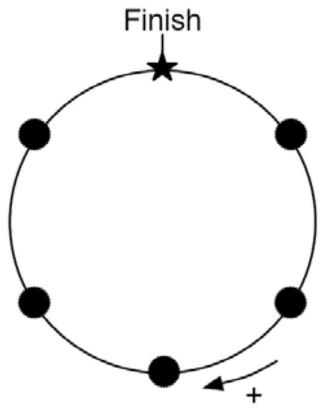
Ogni nodo è un oscillatore accoppiato in pulsazione

- descritto dalla funzione $\phi_i(t) = \sin(\omega * t + \Phi_i)$ (fig. a)
- con velocità angolare ω e periodo $T = 1/\omega$
- con fase Φ_i e fase istantanea ϕ_i da interpretare come percentuale di completamento del periodo: all'istante t_1 (fig. b) ed all'istante t_2 (fig. c)

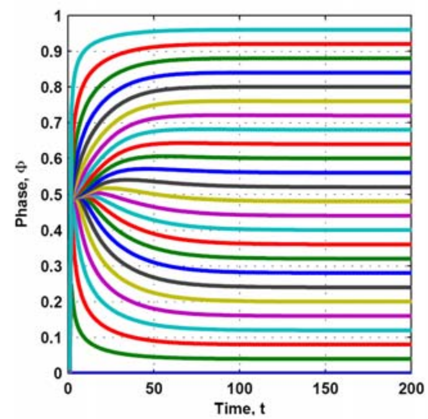
Ogni oscillatore è etichettato per fase crescente ϕ_i (fig. d)

- con due vicini $\phi_{i\pm1}$
- con delta fase $\Delta_i = d(\phi_i) = \phi_i - \phi_{i-1}$

DESINCRONIA: MODELLO TEORICO 2/4



(e)



(f)

Lo stato del sistema è descritto (dato $S=[0,1]$ di valori normalizzati e n oscillatori)

- da $\overrightarrow{\phi} \in S^n$
- da $\overrightarrow{\Phi} \in S^n$ che astrae dalla percentuale di fase completata
- da $\overrightarrow{\Delta} = d(\overrightarrow{\phi}) \in S^n$ che astrae dalla fase

La desincronia è descritta formalmente

- da $\overrightarrow{\phi^*}$, che continua a variare una volta raggiunto lo steady state (fig. e)
- da $\overrightarrow{\Phi^*}$, costante una volta raggiunto lo steady state (fig. f)
- da $\overrightarrow{\Delta^*} = d(\overrightarrow{\phi^*}) = \{1/n, \ldots, 1/n\}^T$, costante una volta raggiunto lo steady state

L'entropia del sistema

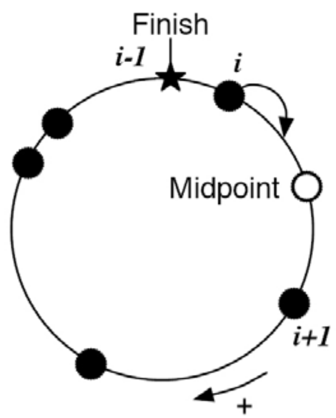
- è definita in funzione del delta fase per astrarre dalla fase istantanea $\overrightarrow{\phi}$
- considera il delta fase come una distribuzione di probabilità $\sum_{i=1}^{n} \Delta_i = 1$
- è sempre positiva, $H(\overrightarrow{\Delta}) = -\sum_{i=1}^{n} \Delta_i * \ln(\Delta_i) \geq 0$
- ha il suo minimo in 0 (sincronia) e massimo in $H_{MAX} = H(\overrightarrow{\Delta^*}) = -\sum_{i=1}^{n} \frac{1}{n} * \ln(\frac{1}{n}) = \ln(n)$

La distanza del sistema dalla desincronia

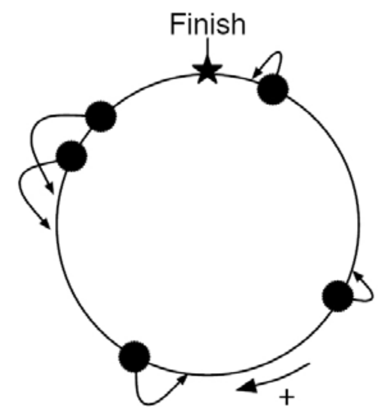
- è definita in funzione dell'entropia
- è sempre positiva, $\left\|\overrightarrow{\phi}\right\| = H_{MAX} - H(d(\overrightarrow{\phi}))$
- ha il suo minimo in 0 (desincronia) e massimo in H_{MAX}

Un desincronizzatore è una funzione $f: S^n \rightarrow S^n$ (sufficiente, non necessario)

- mantiene la desincronia $\left\|f(\overrightarrow{\phi^*})\right\| = \left\|\overrightarrow{\phi^*}\right\| = 0$
- aumenta l'entropia $\exists m > 0. \left\|\overrightarrow{\phi}\right\| > 0 \Rightarrow \left\|f^m(\overrightarrow{\phi})\right\| < \left\|\overrightarrow{\phi}\right\|$



(g)



(h)

Funzione di salto

- ogni nodo, una completato il periodo, esce con $\phi_i = 1$ e rientra con $\phi_i = 0$
- l'uscita di un nodo qualsiasi è un evento che viene comunicato a tutti gli altri nodi e utilizzato dal vicino (fig. g, DESYNC) o da tutti gli altri nodi (fig. h, INVERSE-MS)
- i nodi che utilizzano l'evento applicano una funzione di salto $\phi_i' = f(\vec{\phi}, \alpha)$ che utilizza come parametri: lo stato dei vicini (DESYNC) o di tutti gli altri nodi (INVERSE-MS); un parametro α che controlla la dimensione del salto

Algoritmo desincronizzatore

- definisce una funzione di salto che è anche un desincronizzatore
- definisce le ipotesi (i vincoli) che garantiscono la convergenza alla desincronia

DESYNC-IDEAL 1/2

Caratteristiche (comuni a DESYNC-STALE)

- ogni nodo utilizza per saltare solo informazioni relative ai suoi vicini $\phi_{i\pm1}$
- ogni nodo salta una sola volta per periodo
- il sistema raggiunge un equilibrio statico $\left\| f(\overrightarrow{\phi^*}) \right\| = \left\| \overrightarrow{\phi^*} \right\| = 0$

Ipotesi (vincoli)

- ogni nodo può accedere sempre istantaneamente la fase dei propri vicini $\phi_{i\pm1}$
- $\alpha =]0,2]$

Algoritmo

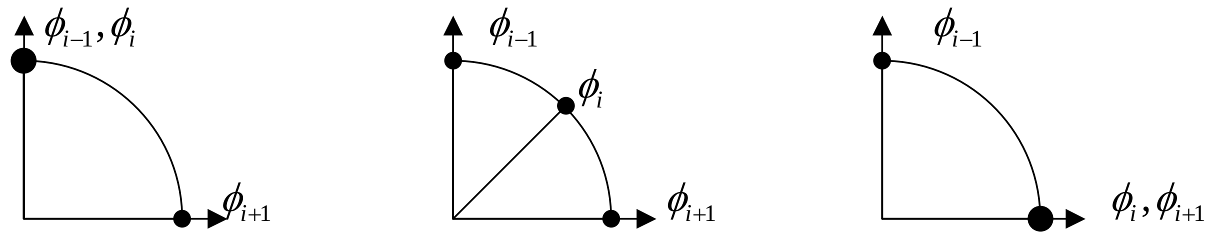
- il nodo i esce e rientra, restando in attesa dell'uscita di $i-1$
- sa quindi che: $\phi_{i-1} = 0$ e può per ipotesi accedere a ϕ_{i+1}
- effettua il salto $\phi_i' = f_{IDEAL,i}(\overrightarrow{\phi}, \alpha) = (1-\alpha)*\phi_i + mean(\phi_{i-1}, \phi_{i+1}) = (1-\alpha)*\phi_i + 0.5*\alpha*\phi_{i+1}$
- $f_{IDEAL,i}$ è un 1-desincronizzatore, necessita cioè dell'uscita di un solo nodo

Influenza del parametro α

- al crescere di α il valore di ϕ_{i+1} influenza sempre più ϕ_i'
- al crescere di α raggruppa sempre più in caso di sparpagliamento e viceversa

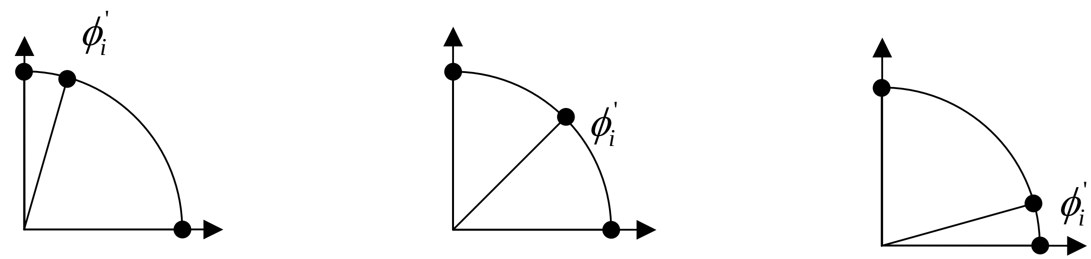
DESYNC-IDEAL 2/2

Prima del salto

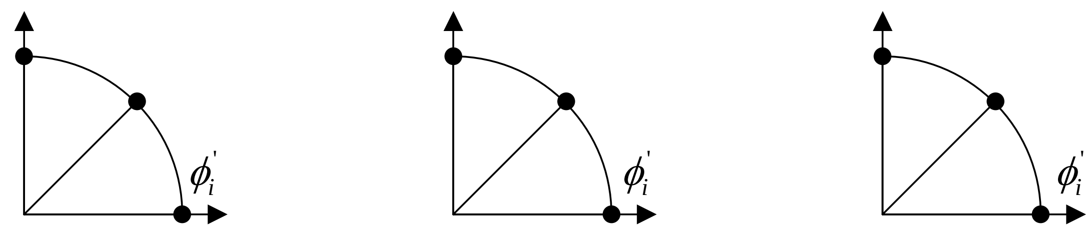


Dopo il salto

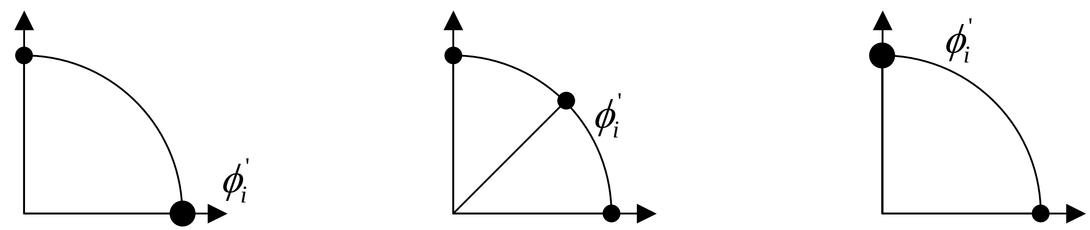
$$\alpha = 0.5 \Rightarrow \phi_i^{'} = \frac{\phi_{i+1} + 2 * \phi_i}{4}$$



$$\alpha = 1.0 \Rightarrow \phi_i^{'} = 0.5 * \phi_{i+1}$$



$$\alpha = 2.0 \Rightarrow \phi_i^{'} = \phi_{i+1} - \phi_i$$



DESYNC-STALE

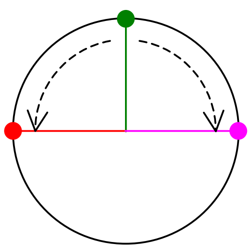
Ipotesi (vincoli)

- non è possibile accedere la fase dei propri vicini $\phi_{i\pm1}$, ma si possono avere informazioni precise sulla fase di ϕ_{i-1} in seguito alla sua uscita
- $\alpha =]0,1]$

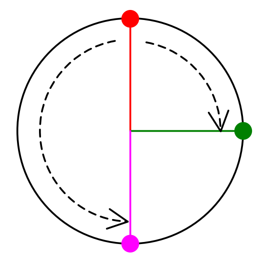
Algoritmo

- il nodo i esce e rientra, restando in attesa dell'uscita di $i-1$
- sa quindi che: $\phi_{i-1} = 0$ e ha memoria (*stale*) di $\tilde{\phi}_{i+1}$
- effettua il salto $\phi_i^{'} = f_{STALE,i}(\overrightarrow{\tilde{\phi}}, \alpha) = (1-\alpha) * \phi_i + mean(\phi_{i-1}, \tilde{\phi}_{i+1}) = (1-\alpha) * \phi_i + 0.5 * \alpha * \tilde{\phi}_{i+1}$
- $f_{STALE,i}$ è un n-desincronizzatore, necessita cioè dell'uscita di n nodi (un ciclo)

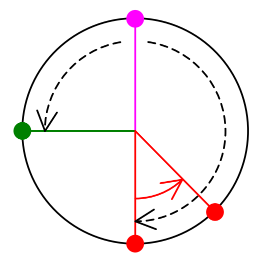
Esempio di informazioni *stale* con $n = 3$ e $\alpha = 1$



1:
2: 1=+0.25
3: 1=+0.75

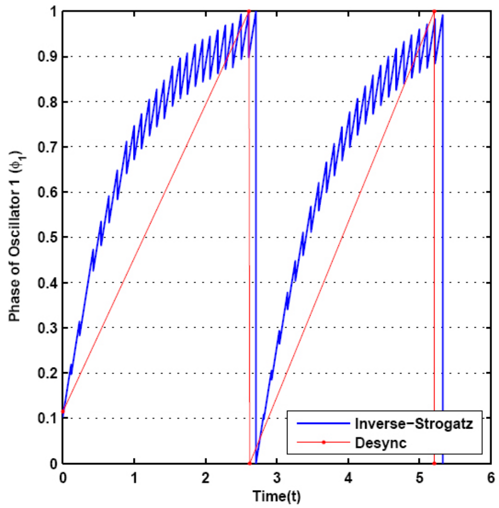


1: 2=0.75
2: 1=0.25
3: 1=0.75, 2=0.50



1: *2=0.75*, 3=0.25
2: 1=0.25, 3=0.50
3: 1=0.75, *2=0.50*

INVERSE-MS 1/2



(i)

Desincronia debole

- i tempi intermedi fra le uscite di due nodi consecutivi devono essere uguali $S \approx \alpha * T$
- durante il ciclo un nodo può comunque accelerare

Caratteristiche

- ogni nodo utilizza per saltare informazioni relative a tutti gli altri nodi
- ogni nodo salta n volte per periodo
- il sistema raggiunge un equilibrio dinamico (debolmente desincronizzato) (fig. i)

Ipotesi (vincoli)

- non è possibile accedere la fase degli altri nodi, ma si possono avere informazioni precise sulla loro fase in seguito agli eventi di uscita
- $\alpha =]0,1]$

Algoritmo

- il nodo n esce e provoca il salto di tutti gli altri nodi $\phi_i^{'} = f_{INVERSE-MS,i}(\overrightarrow{\phi}, \alpha) = (1-\alpha) * \phi_i$
- tutti i nodi sono rietichettati secondo $i \rightarrow i+1$

Periodo di equilibrio dinamico

- per determinare il periodo di equilibrio dinamico T^* bisogna trovare il tempo intermedio fra le uscite di due nodi consecutivi:

$$(i=1) \Rightarrow \Delta_i = \alpha + (1-\alpha) * \Delta_n \qquad (i \neq 1) \Rightarrow \Delta_i = (1-\alpha) * \Delta_{i-1}$$

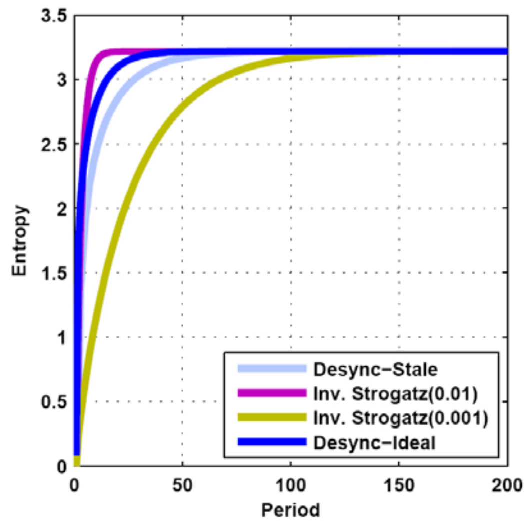
$$da\ cui\ \Delta_n^* = (1-\alpha)^{n-1} * \Delta_1^* = (1-\alpha)^{n-1} * ((1-\alpha) * \Delta_n^* + \alpha)$$

$$risolto\ da\ \Delta_n^* = \frac{(1-\alpha)^{n-1} * \alpha}{1-(1-\alpha)^n}\ e\ da\ cui\ \Delta_i^* = \frac{(1-\alpha)^{i-1} * \alpha}{1-(1-\alpha)^n}$$

$$applicando\ l'Hopital\ \lim_{\alpha \to 0} \Delta_i^* = \frac{\dfrac{d((1-\alpha)^{i-1} * \alpha)}{d\alpha}}{\dfrac{1-(1-\alpha)^n}{d\alpha}} = \frac{1}{n} \quad \text{(desincronia forte)}$$

$$T^*(n,\alpha) = n * \Delta_1^* * T = \frac{n * \alpha * T}{1-(1-\alpha)^n}\ per\ il\ quale\ \lim_{\mathrm{n} \to \infty} T^*(n,\alpha) = n * \alpha * T$$

$$da\ cui\ \frac{T^*}{T} \approx n * \alpha$$



(j)

Influenza del parametro α (per INVERSE-MS)

- al crescere di α la velocità aumenta, al costo di un aumento del periodo T^*
- in un contesto dinamico non è possibile individuare $\alpha = 1/n$ che per n grandi porterebbe a $T^* \approx T$

DESYNC-IDEAL vs. DESYNC-STALE

- la versione *stale* converge più lentamente a causa delle oscillazioni indotte da salti basati su valori *stale* di ϕ_{i+1}

Accuratezza di desincronizzazione

- dipende dalla deviazione assoluta del delta fase rispetto allo stato di desincronia: $\left|\vec{\Delta}\right| = \sum_{i=1}^{n}\left|\Delta_i - \frac{1}{n}\right|$; uno stato è ε-desincronizzato se $\left|\vec{\Delta}\right| < \varepsilon$

Velocità di convergenza

- DESYNC: $O(n^2 * \alpha^{-1} * \ln(\varepsilon^{-1}))$
- INVERSE-MS: $O(n^{-1} * \alpha^{-1} * \ln(n * \varepsilon^{-1}))$

DESINCRONIZZAZIONE: DESYNC VS. INVERSE-MS 1/2

Vincolo di velocità di convergenza

- bisogna reagire molto rapidamente ai mutamenti
- è preferibile INVERSE-MS, tenendo conto della distorsione $O(n * \alpha)$ del periodo
- DESYNC converge più lentamente (garantendo però maggior robustezza)

Vincolo di robustezza

- bisogna garantire la convergenza anche in caso di perdite di messaggi
- è preferibile DESYNC, in cui un miss influenza solo i nodi vicini
- INVERSE-MS risente anche di un singolo miss, che squilibra tutti gli altri nodi

Vincolo di churn

- bisogna garantire la convergenza anche in presenza di un churn costante
- è preferibile DESYNC con α piccolo, che presenta una maggior inerzia nei mutamenti del periodo
- INVERSE-MS può produrre un periodo scorretto

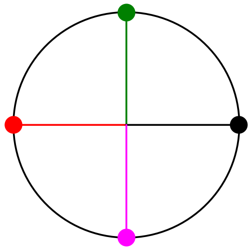
Confronto dei vincoli

	INVERSE-MS	DESYNC-IDEAL	DESYNC-STALE
Velocità di convergenza	$O(\ln(n)/n)$	$O(n^2)$	$O(n^2)$
Distorsione del periodo	$O(n\alpha)$	No	No
Robustezza	No	Si	Si
Churn	No	Si	Si

Confronto degli ambiti di applicazione

	INVERSE-MS	DESYNC-IDEAL	DESYNC-STALE
ADC	No – Distorsione	Si	Si
Processore multicore	Si	Si	Si
Wireless TDMA	No – Robustezza	No – Accesso vicini	Si
Intersezione del traffico	No – Churn	Si	Si

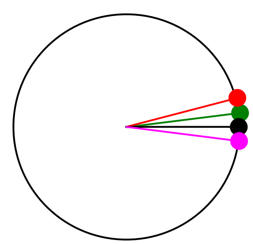
[1] - Desincronia



1: I/O bound, 25% bandwidth
2: I/O bound, 25% bandwidth
3: I/O bound, 25% bandwidth
4: I/O bound, 25% bandwidth

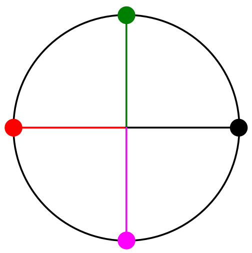
[2] – Transizione di sincronia

- i nodi CPU bound rinunciano al loro quanto e al momento del salto si spostano esattamente alla fase di ϕ_{i-1}



1: CPU bound, 0% bandwidth
2: CPU bound, 0% bandwidth
3: CPU bound, 0% bandwidth
4: I/O bound, 100% bandwidth

[3] - Ritorno alla desincronia



1: I/O bound, 25% bandwidth
2: I/O bound, 25% bandwidth
3: I/O bound, 25% bandwidth
4: I/O bound, 25% bandwidth