Salvaguarda Distribuída

Distributed snapshots

Cortes coerentes:
capturar o **estado global** de um SD

- Para quê?
- Pode ser útil para vários fins:
  - Guardar o estado de forma a poder recomeçar a aplicação em caso de falha (recuperação “para trás”)
  - Verificar propriedades do sistema:
    - O sistema está em interbloqueio?
    - Foi violada uma propriedade?
  - Depurar o sistema

Desafios

- Desafio: não há **tempo global**
  - Os relógios não estão todos sincronizados

- O estado da aplicação inclui não só o estado de cada processo, mas também **possíveis mensagens em trânsito**

Abordagem ingénua

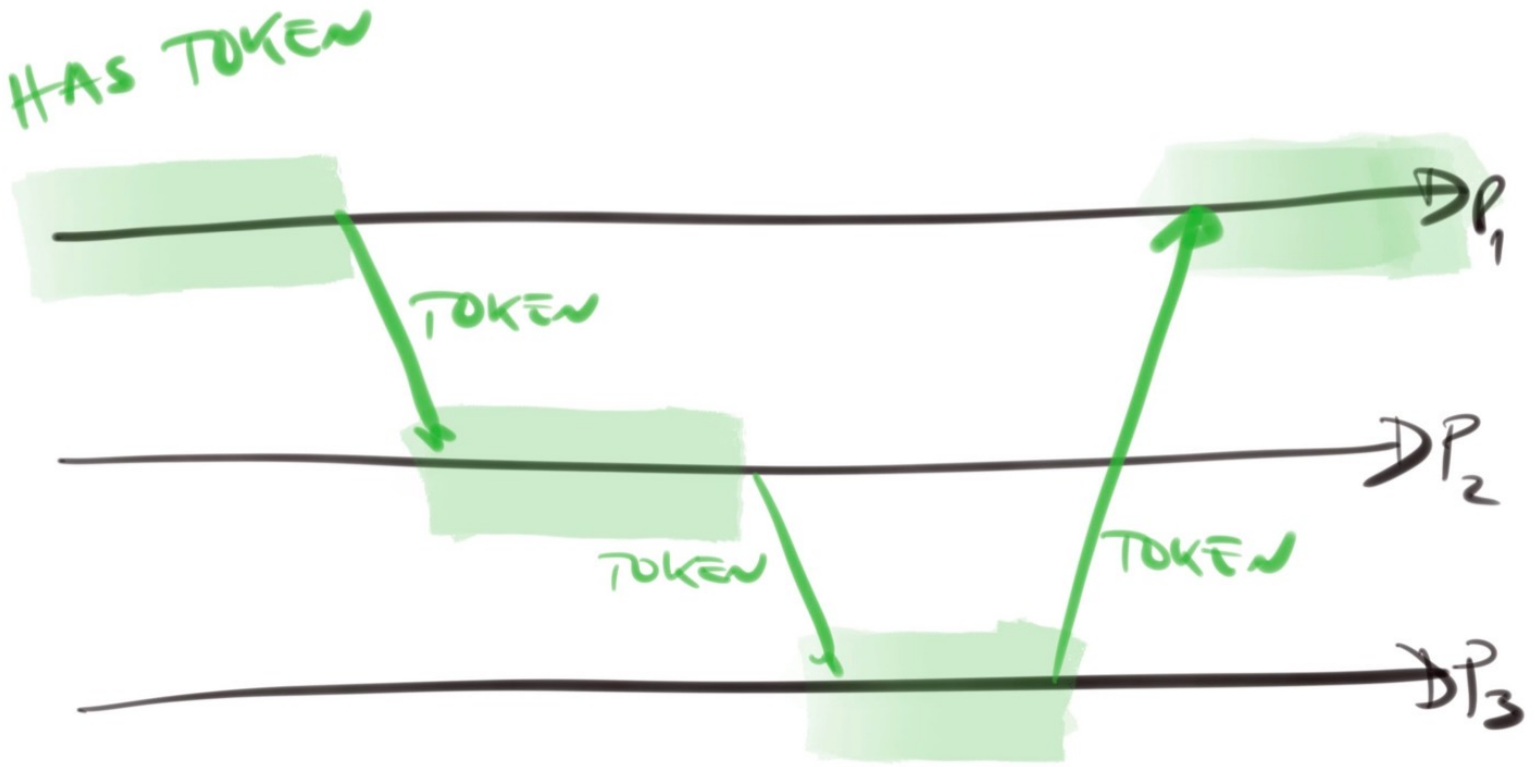
- Coordenador central que ordena a todos os processos que:
  - Parem o que estão a fazer
  - Guardem o seu estado local (por exemplo, num serviço remoto de armazenamento)
  - Esperem até o coordenador os notificar (após o estado global estar completo)

- Poderíamos construir um algoritmo à volta desta ideia...

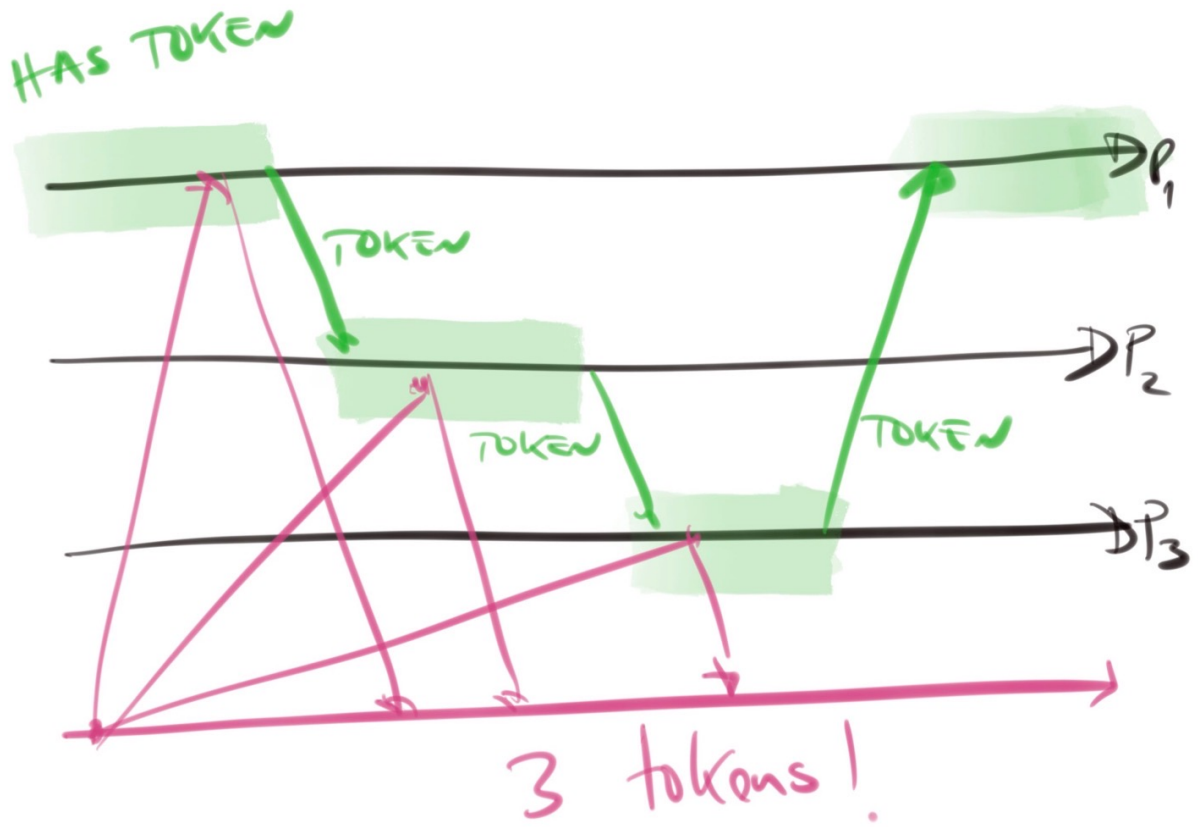
... Mas “parar o mundo” não é nada eficiente

E se tentarmos guardar o estado global sem parar o sistema?

Vamos considerar este exemplo

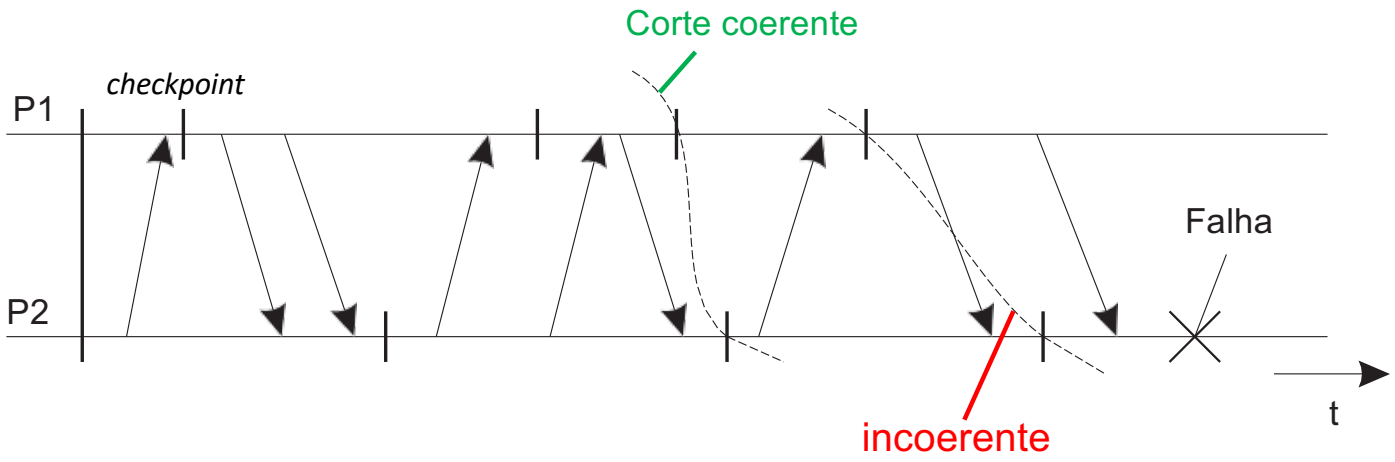


Cuidado com cortes incoerentes!

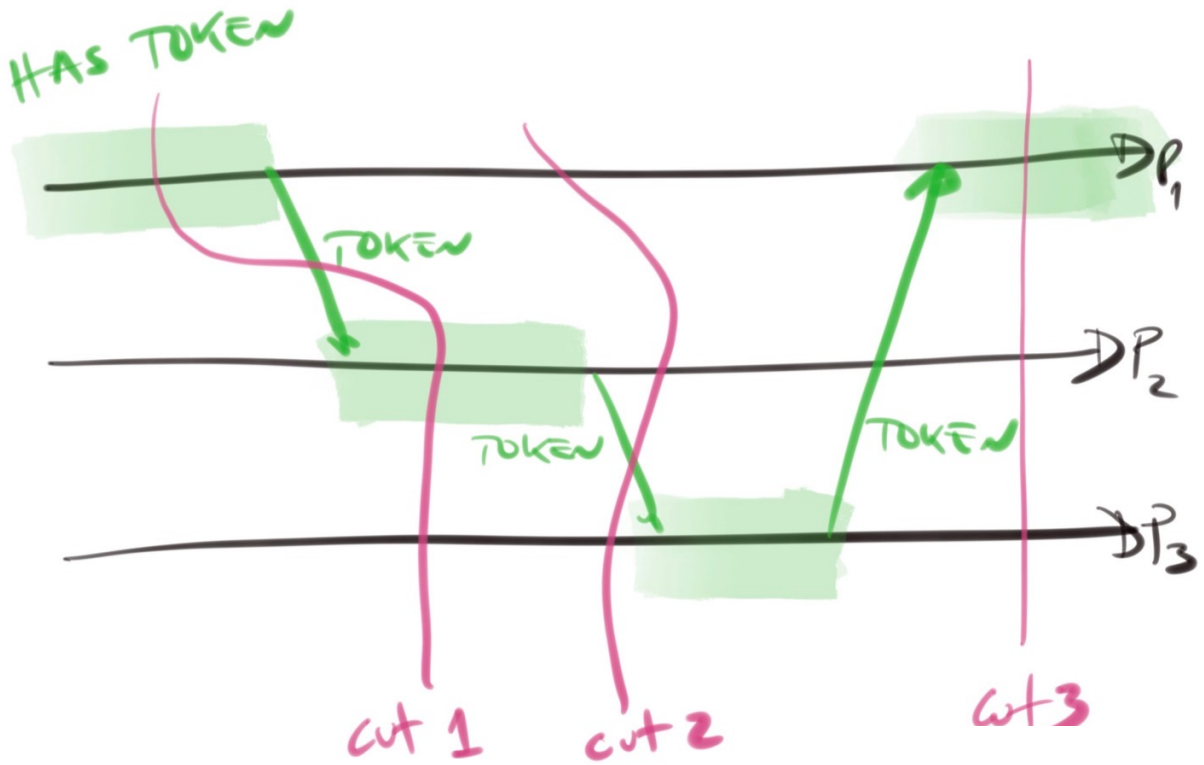


Corte coerente: definição

- Um corte é coerente se, para cada um dos seus eventos, o corte inclui todos os eventos que ***acontecem-antes*** desse evento



Cortes incoerentes vs coerentes



Tentemos construir algo melhor que abordagem “pára o mundo”

Um primeiro algoritmo de “snapshot” que não pára o mundo

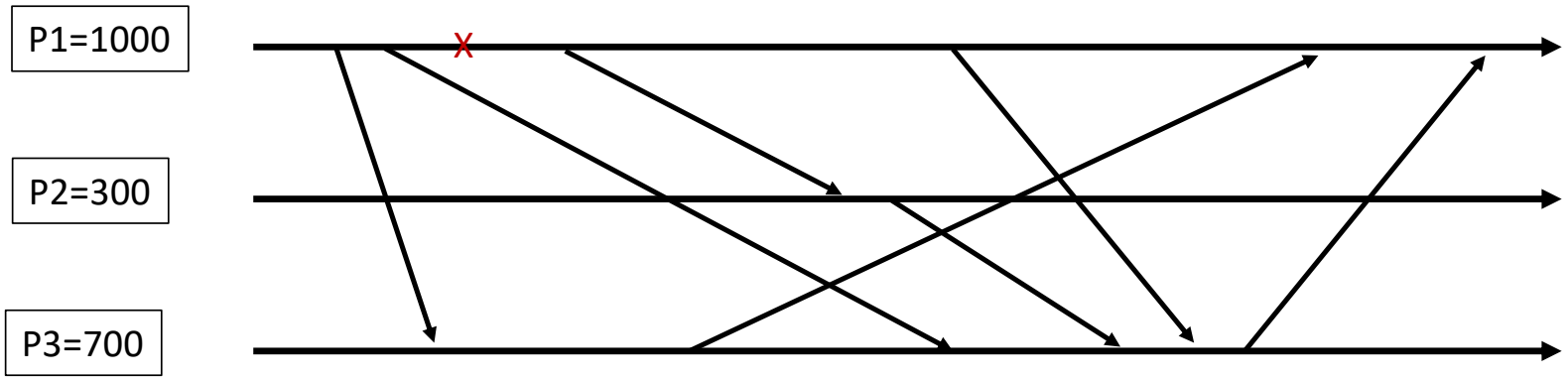
- Pressupostos: processos não falham e rede garante entrega ordenada (FIFO)
- Qualquer processo pode iniciar um snapshop global a qualquer momento, guardando o seu estado local

- Quando o processo p_i guardou o seu estado:
  - **Envia um marcador** a todos os outros processos
    Depois de ter enviado os marcadores, p_i pode continuar a executar normalmente (enviando e recebendo mensagens “normais”)
- Quando o processo p_i **recebe** uma mensagem **marcador:**
  - Se p_i **ainda não guardou** o seu estado, então guarda o seu estado local

O que aconteceria se rede não assegurasse FIFO?

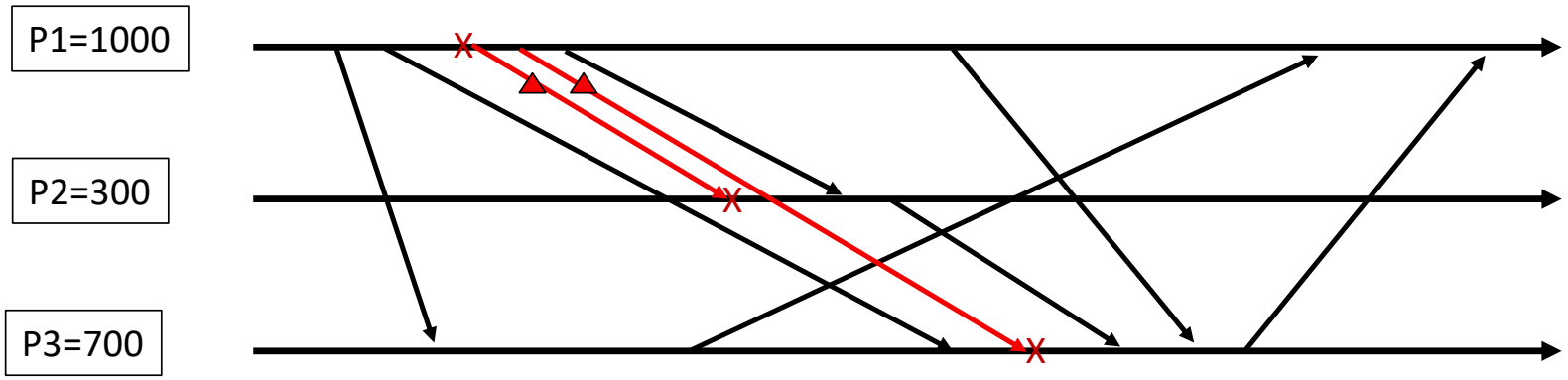
Exercício

- Cada processo possui alguns tokens no início, e cada mensagem transfere 100 tokens entre 2 processos. Considere que o processo p1 inicia um snapshot/salvaguarda no instante X. Qual vai ser o estado capturado por este algoritmo?



Exercício

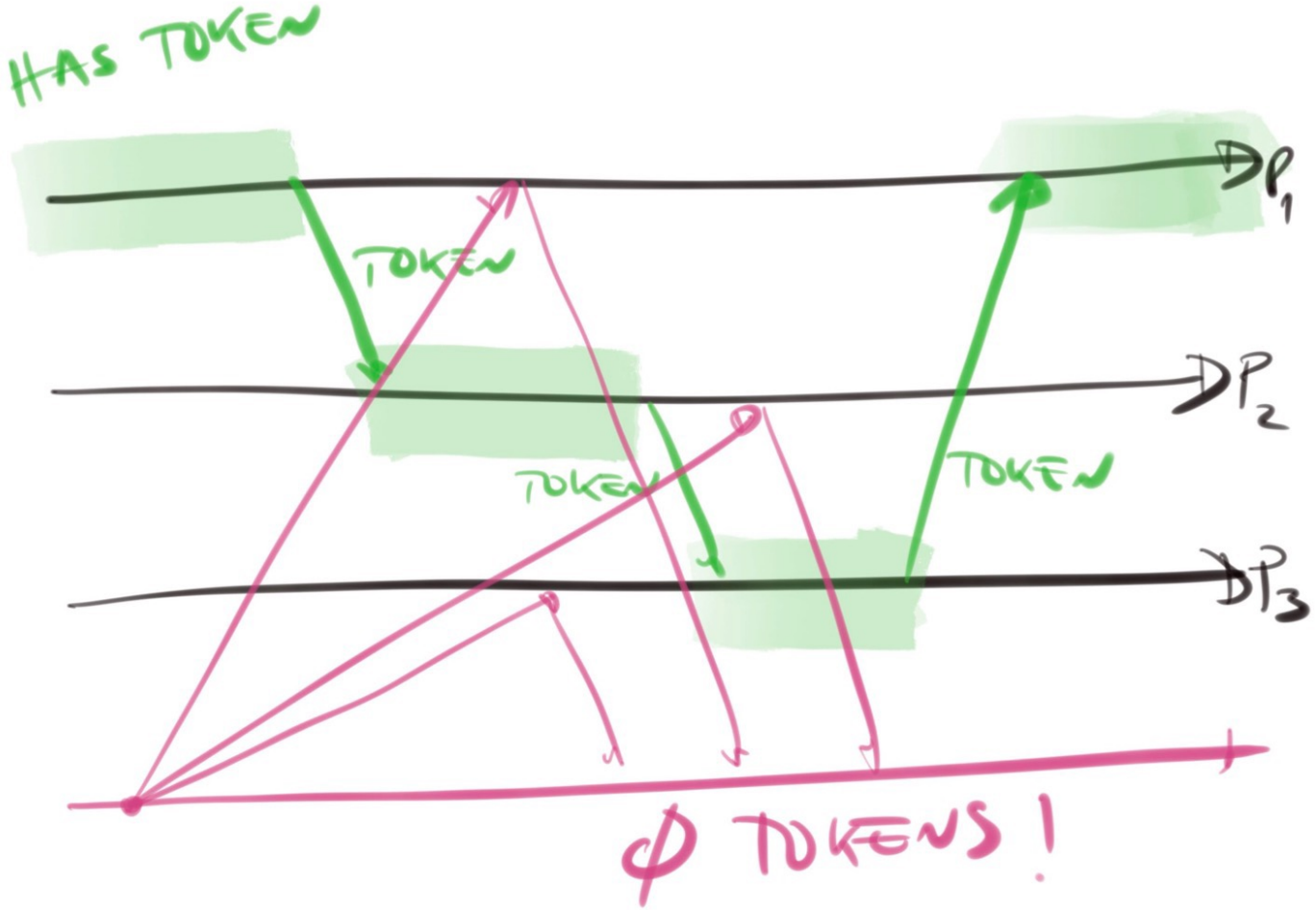
- Cada processo possui alguns tokens no início, e cada mensagem transfere 100 tokens entre 2 processos. Considere que o processo p1 inicia um snapshot/salvaguarda no instante X. Qual vai ser o estado capturado por este algoritmo?



Resposta: P1=800, P2=300, P3=800

No início, a soma de tokens era 2000. Mas o checkpoint soma 1900. Porquê?

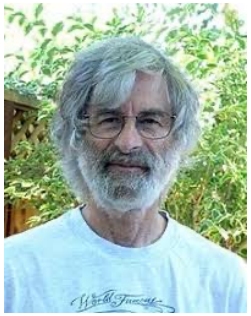
Importante capturar também o estado dos canais!



Algoritmo de *snapshot* distribuído de Chandy-Lamport

Estender a solução anterior para capturar também o estado dos canais

Chandy-Lamport



Distributed Snapshots: Determining Global States of Distributed Systems

K. MANI CHANDY
University of Texas at Austin
and
LESLIE LAMPORT
Stanford Research Institute

This paper presents an algorithm by which a process in a distributed system determines a global state of the system during a computation. Many problems in distributed systems can be cast in terms of the problem of detecting global states. For instance, the global state detection algorithm helps to solve an important class of problems: stable property detection. A stable property is one that persists: once a stable property becomes true it remains true thereafter. Examples of stable properties are "computation has terminated," "the system is deadlocked" and "all tokens in a token ring have disappeared." The stable property detection problem is that of devising algorithms to detect a given stable property. Global state detection can also be used for checkpointing.

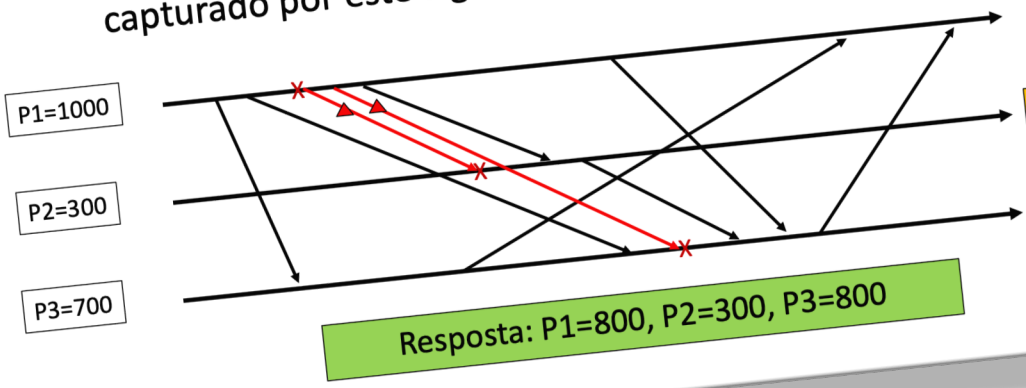
Algoritmo de Chandy-Lamport

- Pressupostos
  - Não há falhas dos processos nem dos canais de comunicação (que são fiáveis)
  - Canais unidirecionais, ordem FIFO
- Tal como antes:
  - Qualquer processo pode iniciar um **snapshop global** a qualquer momento
  - Os processos **podem continuar a sua execução** enviando e recebendo mensagens “normais” durante o snapshot
- O que há de novo: estado do canal também é guardado

Quais mensagens precisamos salvaguardar?

Exercício

- Cada processo possui alguns tokens no início, e cada mensagem transfere 100 tokens entre 2 processos. Considere que o processo p1 inicia um snapshot/salvaguarda no instante X. Qual vai ser o estado capturado por este algoritmo?



Resposta: P1=800, P2=300, P3=800

No início, a soma de tokens era 2000. Mas o checkpoint soma 1900. Porquê?

Para um dado canal (unidirecional),
se o recetor salvaguardou o seu estado local **antes** do emissor ter feito o mesmo,
é necessário salvaguardar as mensagens recebidas nesse canal **entre os dois instantes**

O algoritmo Chandy-Lamport em 2 regras

Marker receiving rule for process p_i

On receipt of a *marker* message at p_i over channel c:

if (p_i has not yet recorded its state) it

records its process state now;

records the state of c as the empty set;

turns on recording of messages arriving over other incoming channels;

else

p_i records the state of c as the set of messages it has received over c since it saved its state.

end if

Marker sending rule for process p_i

After p_i has recorded its state, for each outgoing channel c:

p_i sends one marker message over c (before it sends any other message over c).

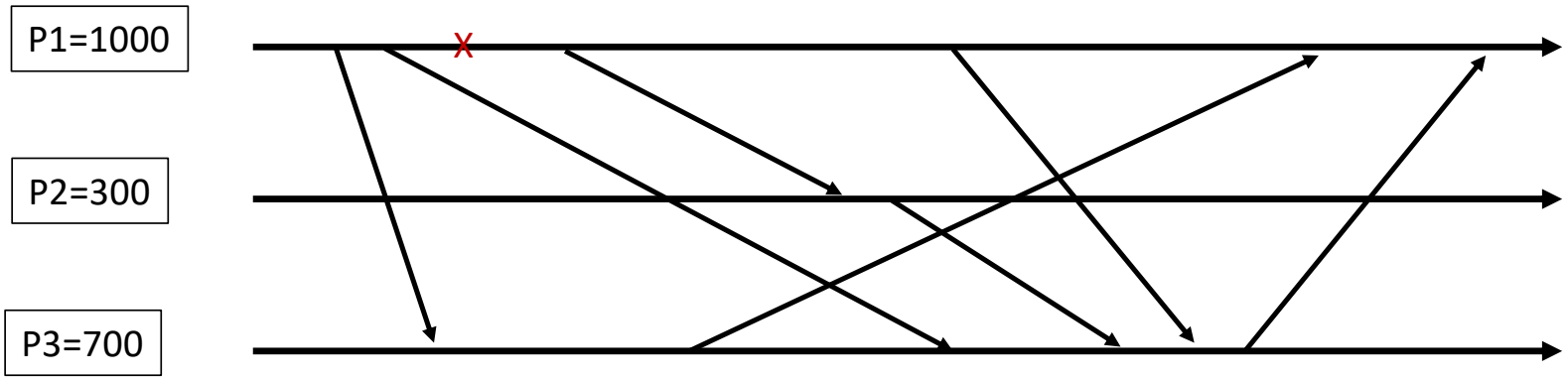
Retirado da página 616 do livro da cadeira!

Algoritmo “snapshot” (Chandy-Lamport): terminação

- Nenhum processo decide quando termina
- Todos os processos vão receber um marcador (e gravar o seu estado)
- Todos os processos receberam um marcador nos outros N–1 canais de receção (e gravaram os seus estados)
- Mais tarde, um servidor central pode obter os estados locais coerentes para construir o snapshot global

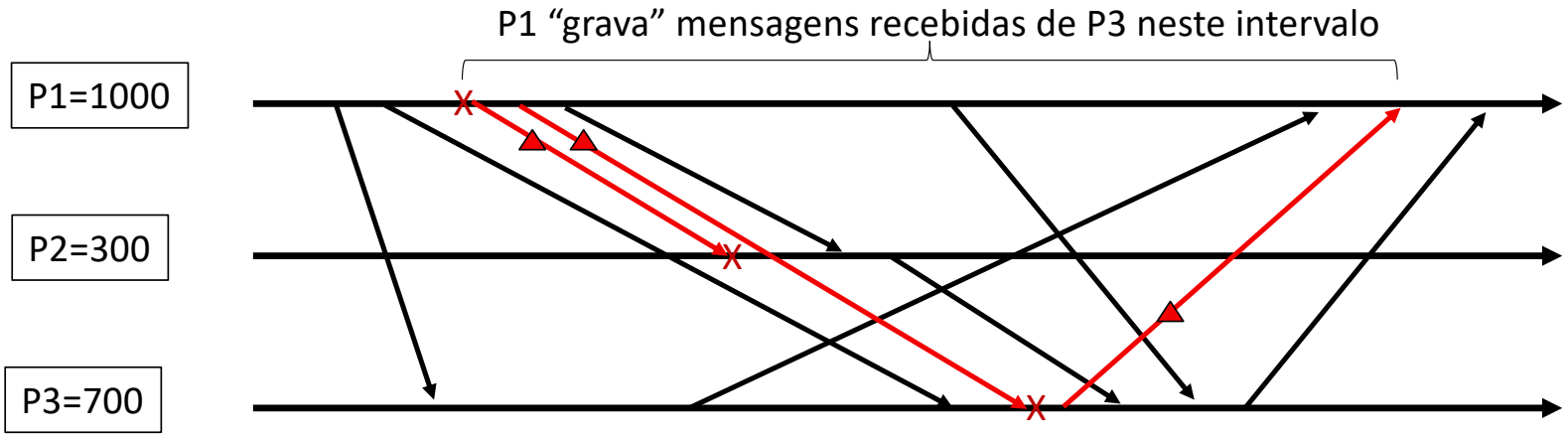
Exercício

- Cada processo possui alguns tokens no início, e cada mensagem transfere 100 tokens entre 2 processos. Considere que o processo p1 inicia um snapshot/salvaguarda no instante X. Qual vai ser o estado capturado pelo algoritmo Chandy-Lamport?



Exercício

- Cada processo possui alguns tokens no início, e cada mensagem transfere 100 tokens entre 2 processos. Considere que o processo p1 inicia um snapshot/salvaguarda no instante X. Qual vai ser o estado capturado por este algoritmo?



Resposta: P1=800, P2=300, P3=800,
Canal P3>P1=100, outros canais ”vazios”

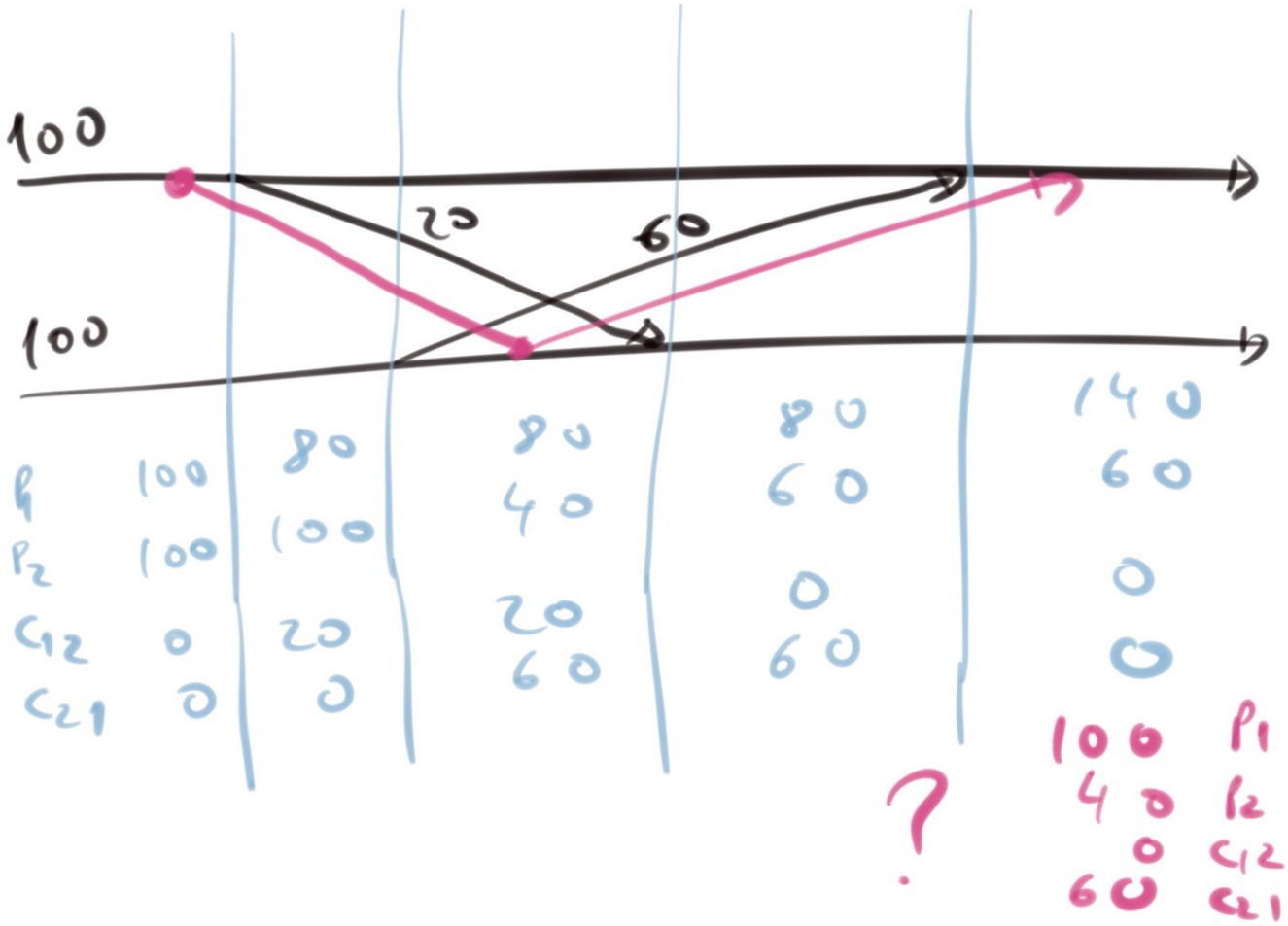
(*) Alguns marcadores omitidos na figura.

Corte coerente

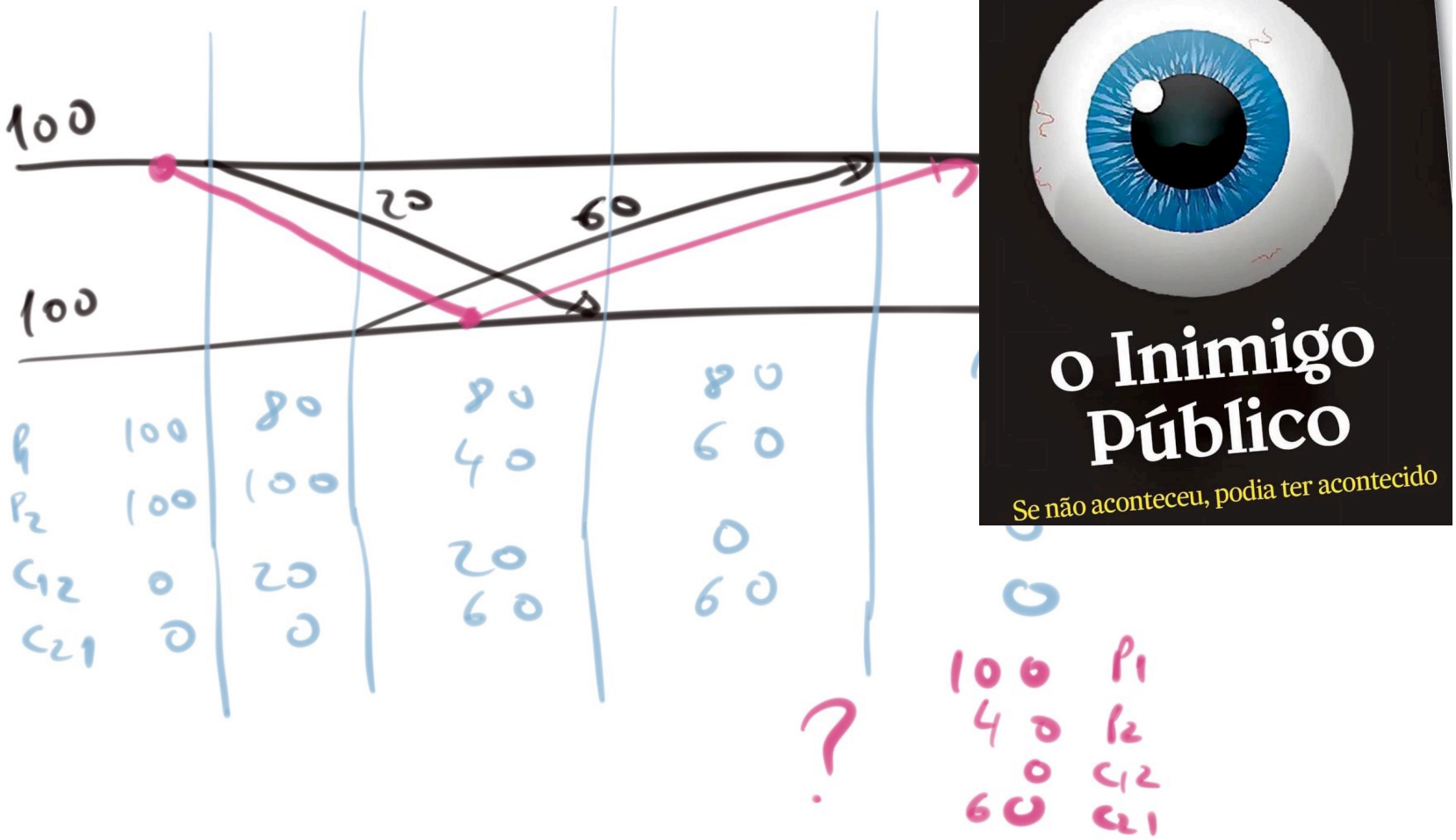
=

estado global do SD num dado instante?

Corte Coerente vs Execução Real



Corte Coerente vs Execução Real



Bibliografia recomendada

- [Coulouris et al]
  - Secções 14.5 e 17.6