ETAT GLOBAL ET PROPRIÉTÉS STABLES

PR. SAÏDOUNI DJAMEL EDDINE

EQUIPE CFSC – LABORATOIRE MISC

FACULTÉ DES NTIC – DÉPARTEMENT IFA

UNIVERSITÉ CONSTANTINE 2 – ABDELHAMID MEHRI

DJAMEL.SAIDOUNI@UNIV-CONSTANTINE2.DZ

INTRODUCTION

La situation à l'instant t est difficile à connaître: Etat des processus + Communications en cours

Exemple: Somme des comptes dans une agence bancaire ?

- On va chercher si la situation globale satisfait ou non une certaine propriété:
  - Exemple : L'application est-elle terminée ?
    - Processus passifs + canaux vides
  - Exemple : y-a-t-il interblocage ?
  - Exemple : Le jeton est-il perdu?
- Propriétés stables : Elles peuvent passer de False à True mais jamais de True à False.

ALGORITHME DE MISRA ET LAMPORT

Détection d'une propriété stable quelconque.

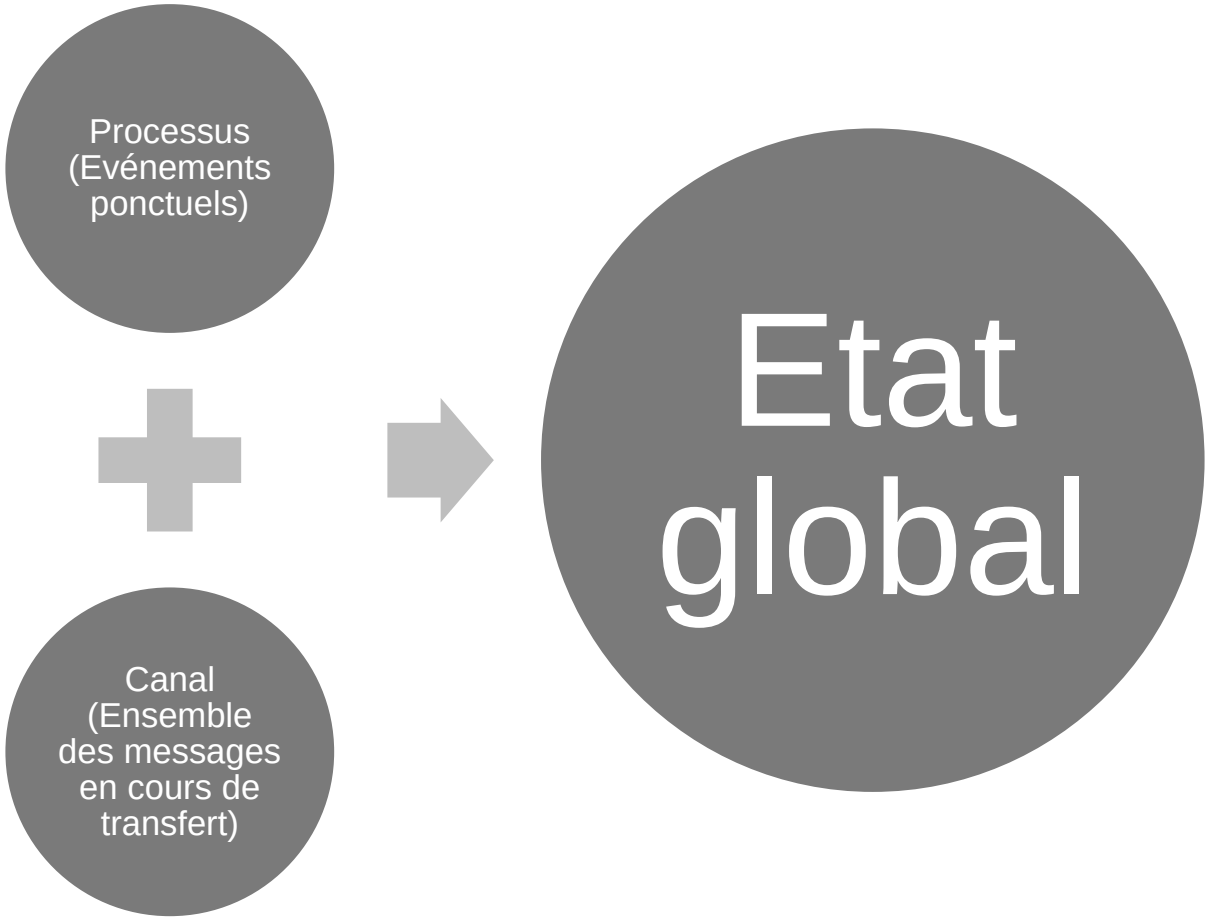
Hypothèses :

- Réseau à topologie quelconque
- Canaux unidirectionnels
- Réseau régulier

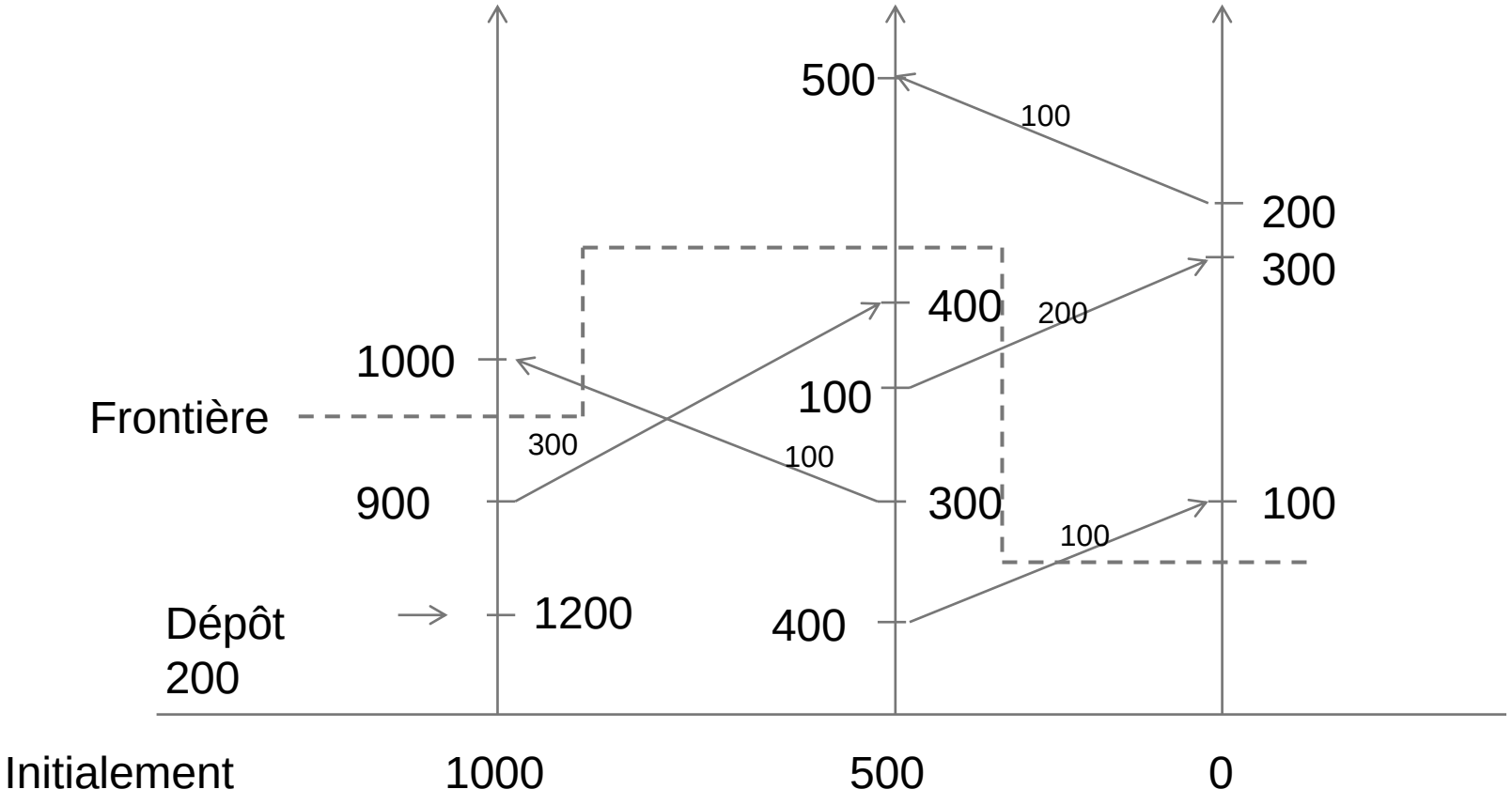
L'état global correspondra à l'état des processus à des instants différents. Pour que l'information soit significative,

- Les instants d'observation des processus, bien que différents, ne peuvent pas être totalement quelconques
- Il faut déterminer les contraintes minimales sur les instants d'observation pour que la collecte des informations présente une cohérence suffisante

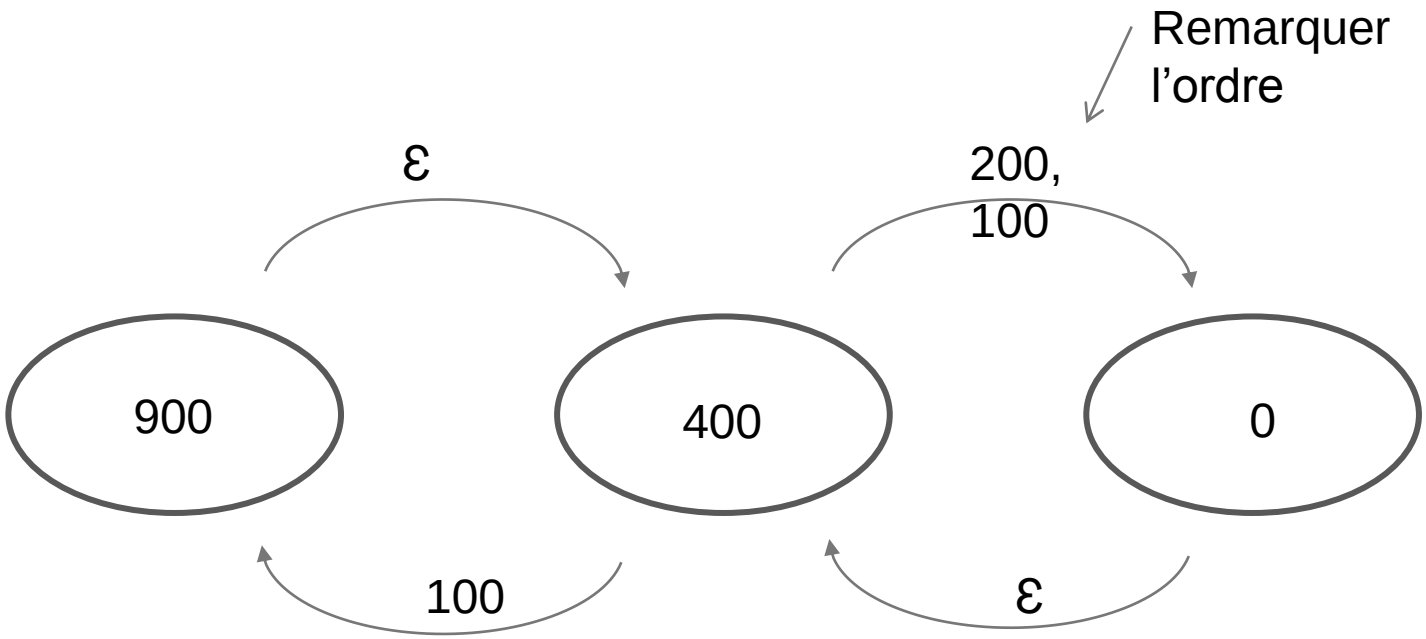
ETAT GLOBAL


Processus (Evénements ponctuels)
Canal (Ensemble des messages en cours de transfert)
Etat global

EXEMPLE DES AGENCES BANCAIRES



SITUATION DES PROCESSUS PAR RAPPORT À LA FRONTIÈRE



- Sous la frontière c'est le passé
- Au dessus de la frontière c'est le futur
- Message en transfert : L'émission appartient au passé et la réception appartient au futur
- La frontière ne peut pas être totalement arbitraire, elle doit vérifier certaines propriétés

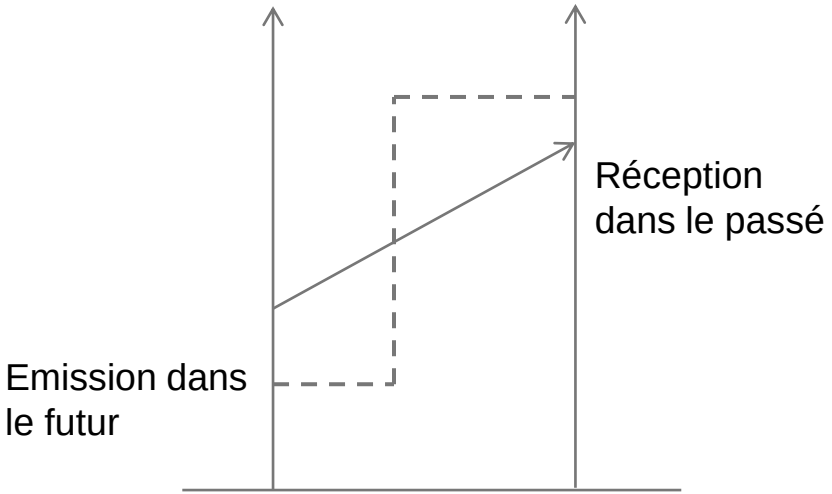
QUELQUES CONCEPTS

Une tranche T définit un ensemble d'événements tel que :

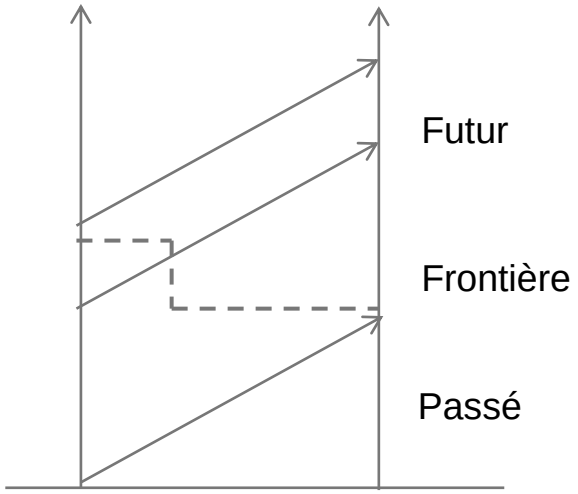
Si $f \in T$ et si e précède f alors e $\in T$

Relation de précédence:

- Tous les événements d'un processus sont totalement ordonnés
- L'émission d'un message précède toujours sa réception



NON



OUI

ETAT GLOBAL

Définition: L'état global g(T) associé à une tranche T est défini par:

1. Pour chaque processus P:
   a) Si aucun événement de P dans T alors g(T) contient l'état initial de P.
   b) Sinon g(T) contient l'état après le dernier événement de P dans T.
2. Pour chaque canal (unidirectionnel) c: L'état de c est la séquence des messages émis par les événements appartenant à T, et reçus par les événements n'appartenant pas à T.

LANCEMENT DE L'ALGORITHME

Un processus quelconque (voir plusieurs processus) peut déclencher l'opération.

Chaque processus va mémoriser:

- Son état
- L'état des canaux en réception

Le processus désirant connaitre l'état global pourra ensuite interroger tous les autres processus.

ORGANISATION DE L'ALGORITHME

A tout processus on associe une couleur (blanc ou noir).

Initialement chaque processus est blanc.

La couleur ne fait pas partie de l'état à observer.

Les événements sont colorés (blanc ou noir) de la couleur du processus où ils ont lieu.

Les messages sont colorés de la couleur du processus qui les émet.

La tranche de temps sera définie par l'ensemble des événements blancs.

Le passage de blanc à noir pour un processus correspond à la frontière de la tranche.

Le processus qui déclenche les opérations devient immédiatement noir:

- Il diffuse un signal de noircissement sur tous ses canaux en émission (donc à tous ses voisins)

Un processus qui reçoit un signal de noircissement:

- Devient noir et mémorise son état (s'il n'est pas déjà noir)
- Diffuse le signal de noircissement à tous ses voisins

Lorsque tous les processus sont noirs, l'état global de la tranche est défini.

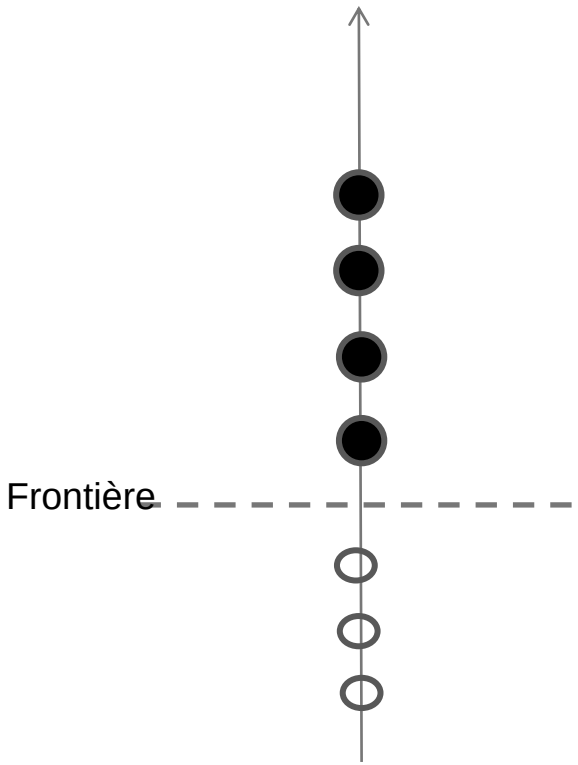
L'algorithme fonctionne aussi si plusieurs processus déclenchent les événements.

ALGORITHME (SUITE)

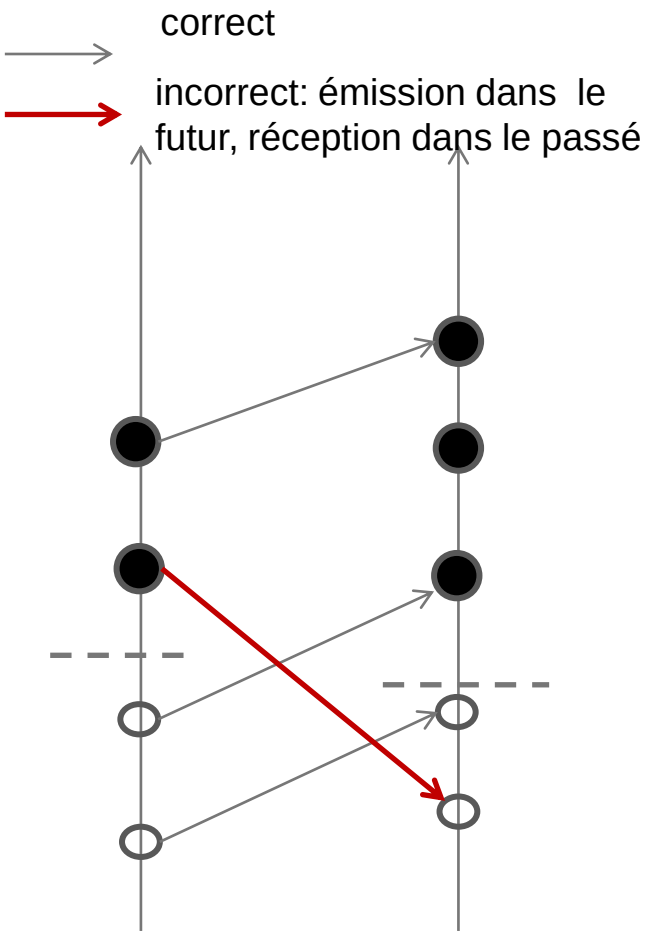
Reste à vérifier que l'ensemble des événements blancs vérifie la définition d'une tranche.

- Pour un processus donné: la définition est vérifiée.
- Pour plusieurs processus:
  - Il faut interdire la réception d'un message noir par un processus blanc.
  - Il faut garantir donc qu'un message noir est reçu par un processus noir. Pour cela, il suffit que tout processus devenant noir émette un signal de noircissement, avant tout autre émission de message. Ainsi, tous ses voisins seront devenus noirs lorsqu'ils recevront un message noir émis par lui (le canal est FIFO).

OPÉRATION DE NOIRCISSEMENT



Noircissement d'un processus



L'ÉTAT GLOBAL G(T)

Définition :

g(T) = états des processus lorsqu'ils deviennent noirs

U

pour chaque canal c la séquence des messages blancs reçus par un récepteur noir

PROCÉDURES DE L'ALGORITHME

Procédure noircir:

- Couleur := noir;
- Enregistrer l'état local;
- Pour chaque canal de réception commencer à enregistrer les messages;
- Pour chaque canal en émission, envoyer un signal de noircissement;

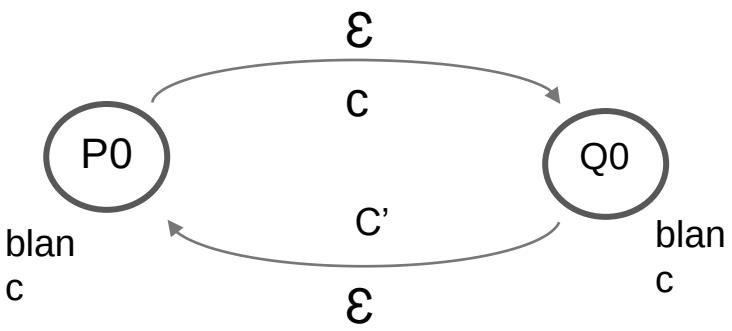
Réception d'un signal sur le canal c:

- Si couleur = blanc alors noircir;
- Arrêter l'enregistrement des messages en réception sur le canal c;

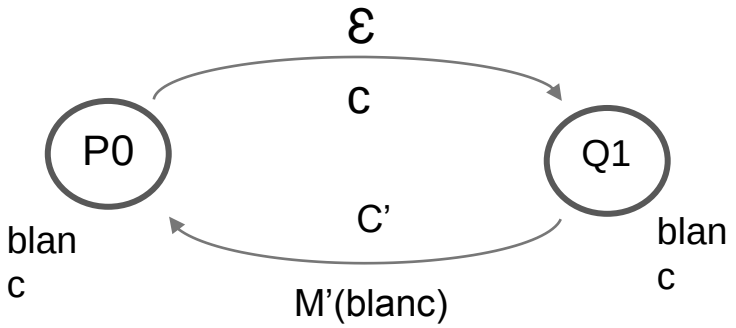
Lorsque tous les processus sont devenus noirs:

- Ils ont enregistré leur état local;
- Ils ont terminé l'enregistrement des messages sur leurs canaux de réception;
- ***Il suffit donc de les visiter par une procédure quelconque pour connaître g(T)***.

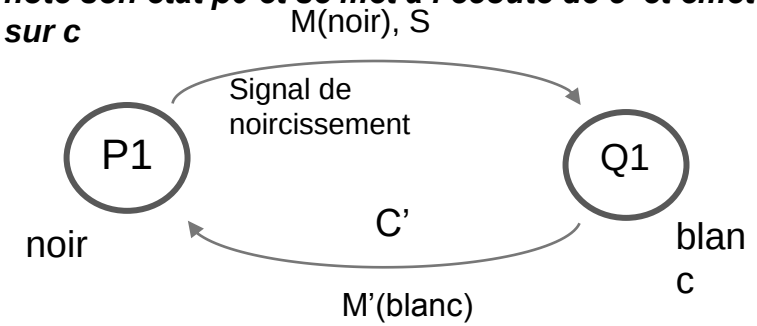
S0



S1

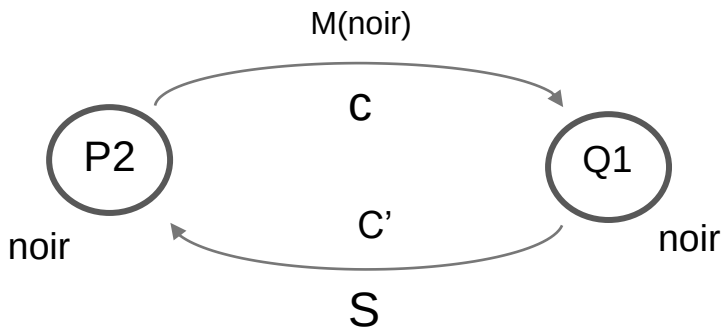


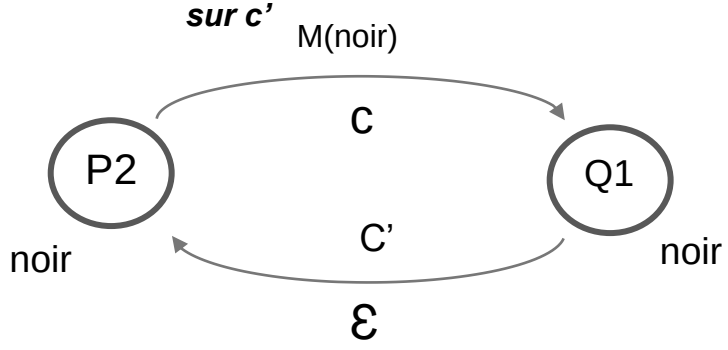
P note son état p0 et se met à l'écoute de c' et émet S sur c



S2

P détecte M'(blanc) sur c'





Q noircit et note son état Q1 et l'état de c:ε et émet S sur c'

S3

S4

EXEMPLE (SUITE)

g(T) = {(P0,Q1), c:Ɛ, c':M'}

Dans ce cas l'état global correspond à un état effectivement observé dans l'évolution du système.

Si on suppose que P noircit avant S1 et que M est émis avant M', on obtiendra le même état global, mais celui-ci ne correspond plus à un état physiquement observable.

UTILISATION DE G(T) POUR LA DÉTERMINATION DE PROPRIÉTÉS STABLES

Bien que g(T) ne corresponde pas nécessairement à un état physiquement observable, il permet de déterminer des propriétés stables.

Supposons un observateur global capable de définir un ordre total sur les événements.

Etat global S, qui évolue à chaque événement.

S0 : état initial

S' := next(S,e) : à chaque événement e pouvant survenir dans l'état S

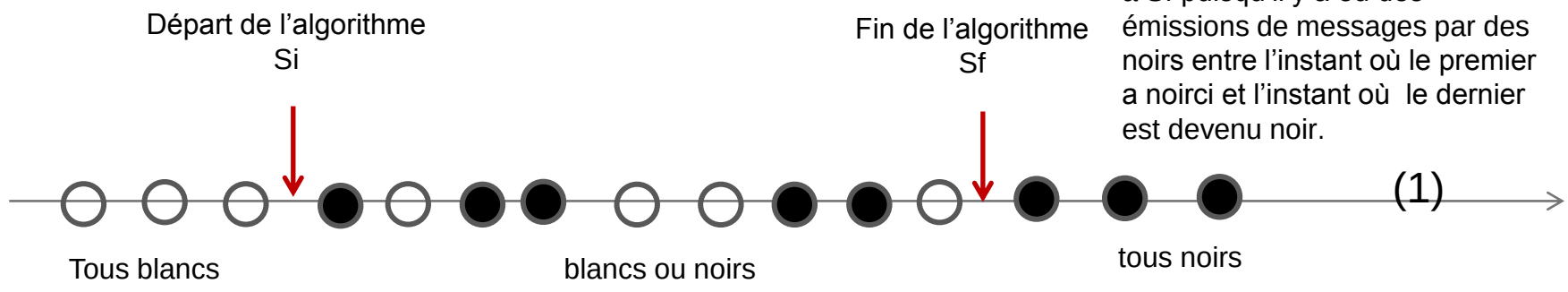
- **Exemple**: si e est la réception d'un message, il faut que l'émission de ce message ait été enregistrée dans S
- La fonction next dépend:
  - De l'application
  - D'une exécution particulière de l'application (non déterminisme)
- La séquence S0, S1, … correspond à un calcul.
- L'état **Sf** peut être atteint à partir de l'état **Si** s'il existe (au moins) un calcul faisant passer de **Si** à **Sf**.

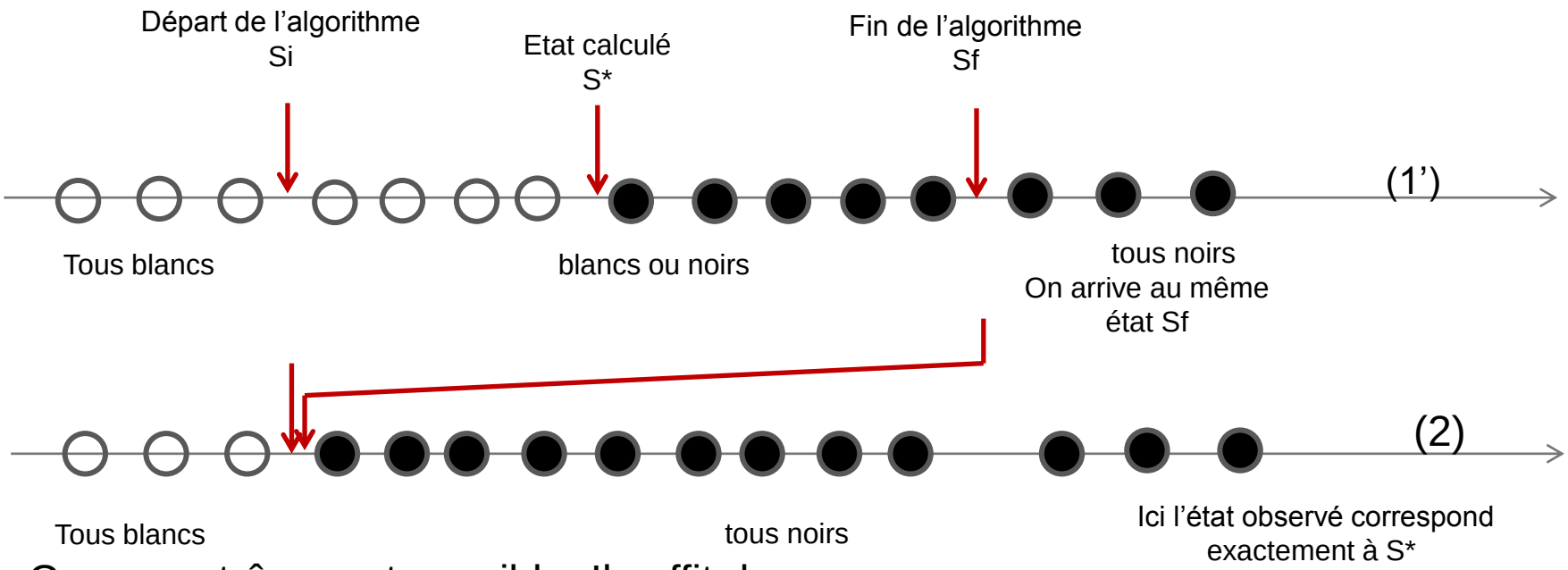
L'algorithme construit l'état global S* (S* correspond-il à un état réalisable ?)

Il existe toujours au moins un calcul de l'application qui fait passer le système par l'état S*

DÉMONSTRATION

On général S* ne correspond pas à Sf puisqu'il y a eu des émissions de messages par des noirs entre l'instant où le premier a noirci et l'instant où le dernier est devenu noir.





Ce cas extrême est possible. Il suffit de:

- arrêter totalement l'application
- Relancer l'application

Donc il existe au moins un calcul permettant d'obtenir cette configuration.

DÉMONSTRATION (SUITE)

Au lieu d'agir aussi brutalement, il est possible d'intervenir sur le calcul pour obtenir la situation (2).

Considérons deux événement consécutifs: un noir suivi d'un blanc (N suivi de B).

Cas 1: 2 événements du même processus. Impossible, puisqu'un processus noir ne peut pas redevenir blanc.

Cas 2: 2 événements liés à 2 processus distincts (événements internes). La précédence n'est pas significative, puisqu'il suffit de ralentir le processus où s'est produit N pour obtenir un nouveau calcul où B précède N.

Cas 3: émission de m, émission de m'. L'ordre n'est pas significatif: On peut, comme dans le cas 2, inverser l'ordre des émissions.

Cas 4: réception de m, réception de m'. Idem ci-dessus.

Cas 5: émission de m, réception m' (ou réception de m, émission m) avec m ≠ m'. Idem ci-dessus.

Cas 6: émission de m, réception de m. C'est le seul cas où les événements ne peuvent pas être permutés. Ce cas n'existe pas, puisque la construction de l'algorithme interdit qu'un message noir soit reçu par un processus blanc.

Il suffit donc de permuter tous les couples (N,B) pour arriver à la situation (2). Donc la précédence noir, blanc n'est pas intrinsèque à l'algorithme; elle est le fait du hasard.

Donc il existe au moins un calcul où tous les blancs précédent tous les noirs. De ce fait, il existe au moins un calcul qui fait passer de Si à S.* *(3)*

DÉMONSTRATION (SUITE)

Soit σ une propriété stable de l'application; on veut savoir si elle est atteinte ou non.

σ est stable *implique* (σ(S) *implique* σ(next(S,e)) pour tout e possible dans l'état S).

On peut construire S* et tester σ(S*).

S* peut être atteinte à partir de Si d'après (3)

Donc σ(Si) *implique* σ(S*)

Sf peut être atteinte à partir de S* (cf schéma(2))

Donc σ(S*) *implique* σ(Sf)

Si non σ(S*) on ne peut rien dire sur σ(Sf)

Si la propriété stable est vraie au moment où l'algorithme démarre, on est sûr qu'elle sera détectée.

Si la propriété stable devient vraie pendant l'exécution de l'algorithme, elle risque de ne pas être détectée.

L'algorithme construit S* et se termine dans la situation Sf.

En général S* ≠ Sf.

RÉSUMÉ

Si la propriété stable σ est vraie en Si (au moment où l'algorithme démarre)

- σ(Si) ***implique*** σ(S*). Donc σ sera certainement détectée en analysant S*. D'autre part σ sera évidement vraie en Sf (σ(S*) ***implique*** σ(Sf))
- Si σ n'est pas vraie en Si, mais devient vraie en cours d'exécution de l'algorithme, alors on ne sait rien sur σ(S*) : elle pourra être vraie ou fausse. Or σ(S*) ***implique*** σ(Sf), donc, dans le cas où σ(S*) est vraie, on est sûr que σ est vraie en Sf