Cours ASPD
Problèmes de l'élection du leader
Telecom Nancy 2A Apprentissage

Dominique Méry
Telecom Nancy
Université de Lorraine

Année universitaire 2024-2025
3 mars 2025

Summary

Section Courante

Election du leader

- Un algorithme d'élection dun leader satisfait les propriétés suivantes :
  - Chaque processus ou site du réseau exécute le même algorithme.
  - L'algorithme est décentralisé : le calcul peut être initialisé par un sous-ensemble arbitraire de processus
  - L'algorithme atteint une configuration terminale pour chaque calcul et dans toute configuration terminale accessible, il n'y a qu'un seul processus dans l'état de savoir qu'il est leader et tous les autres savent qu'ils ne sont pas leader c'est-à-dire perdus.
- Algorithmes considérés
  - Algorithme de LeLann et Chang et Roberts (anneau)
  - Algorithme IEEE 1394 (arborescence)

Problème de l'élection d'un leader

- Les sites d'un réseau n'ont pas conscience des autres sites non connectés à eux.
- Chaque site a une connaissance locale
- Organiser des calculs dans un réseau nécessite de coordiner les calculs : besoin d'un leader.
- Calculs répartis par ondelettes

Algorithme de LeLann

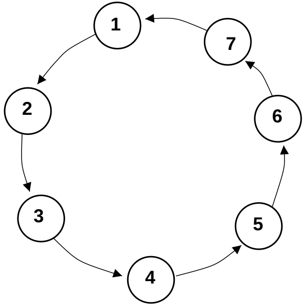
- Un ensemble de nœuds connectés en anneau
- Un sous-ensemble des nœuds est appelé l'ensemble des **candidats** à l'élection
- Chaque candidat maintient un ensemble de valeurs reçues des autres nœuds.
- Si un nœud reçoit le jeton d'un autre nœud et qu'il n'est pas candidat, il ne peut plus être candidat et passe le jeton à son voisin.
- Quand un nœud a reçu son jeton envoyé, il regarde s'il est le jeton de numéro le plus petit et il est élu.

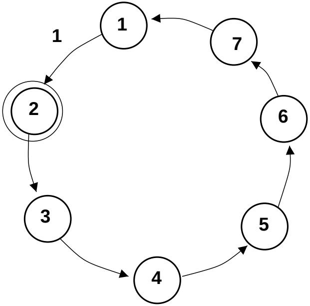
Algorithme de Chang et Roberts

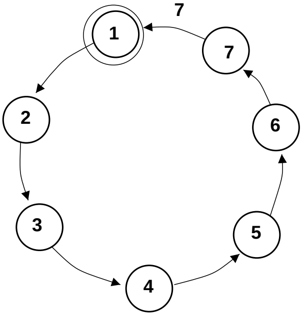
- Un ensemble de nœuds connectés en anneau
- Un sous-ensemble des nœuds est appelé l'ensemble des candidats à l'élection
- Si un nœud candidat reçoit un jeton plus petit que son jeton, il passe le jeton et se déclare pedu, sinon il garde le jeton.
- Si un nœud reçoit le jeton d'un autre nœud et qu'il n'est pas candidat, il ne peut plus être candidat et passe le jeton à son voisin.
- Quand un nœud a reçu son jeton envoyé, il est élu.

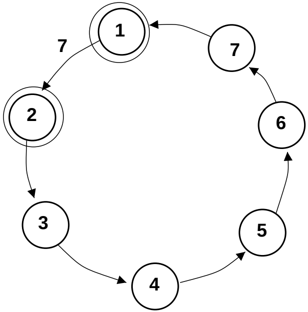
Algorithme LRC Chang et Roberts

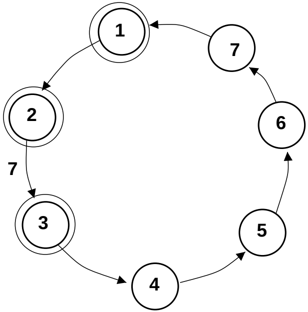
- Un réseau en anneau comporte des nœuds ayant un numéro différent.
- Un des nœuds ets le maximum du réseau
- Le processus d'élection transmet un jeton avec une valeur mise à jour à chaque nœud.

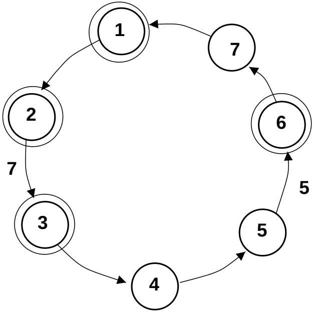

1
2
3
4
5
6
7

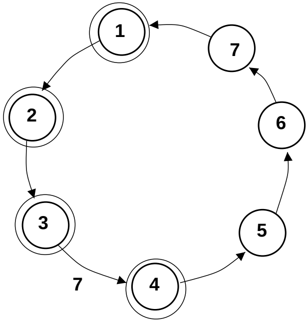

1
1
2
3
4
5
6
7

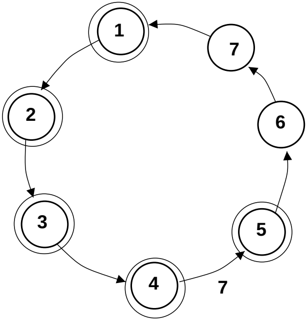

7
1
7
2
6
3
5
4


1
7
7
2
6
3
5
4


1
7
2
6
7
3
5
4


1
7
2
6
7
5
3
5
4


1
7
2
6
3
5
7
4


1
7
2
6
3
5
4
7

Algorithme LCR

- Seul le nœud de numéro maximal sera élu leader
- Les nœuds de numéro minimal restent dans l'état inconnu

Section Courante

]

(FireWire)

- standard international
- Intérêts commerciaux pour sa correction
- Sun, Apple, Philips, Microsoft, Sony, etc impliqués dans son développment
- Trois couches (physique, liaison, transaction)
- Le protocole d'étude est le Tree Identify Protocol
- Localisé dans la phase Bus Reset de la couche physique

Le Problème (1)

- Le bus est utilisé pour acheminer des siganux audio et vidéo digitalisés.
- Il est “hot-pluggable”
- Unités et périphériques peuvent être ajoutés ou retirés à tout instant.
- De tels changements sont suivis d'un bus reset
- L'élection du leader a lieu après un reset dans le réseau.
- Un leader doit être choisi pour agir en tant que manager du bus

Le Problème (2)

- Après un reset du bus tous les nœuds ont même statut.
- Tout nœud sait à quels nœuds il est directement connecté.
- Le réseau est connexe
- Le réseau est acyclique

Références (1)

BASE

- IEEE. *IEEE Standard for a High Performance Serial Bus. Std 1394-1995*. 1995
- IEEE. *IEEE Standard for a High Performance Serial Bus (supplement). Std 1394a-2000*. 2000

Références (2)

GENERAL

- N. Lynch. *Distributed Algorithms*. Morgan Kaufmann. 1996
- R. G. Gallager et al. *A Distributed Algorithm for Minimum Weight Spanning Trees*. IEEE Trans. on Prog. Lang. and Systems. 1983.

Références (3)

MODEL CHECKING

- D.P.L. Simons et al. *Mechanical Verification of the IEE 1394a Root Contention Protocol using Uppaal2* Springer International Journal of Software Tools for Technology Transfer. 2001
- H. Toetenel et al. *Parametric verification of the IEEE 1394a Root Contention Protocol using LPMC* Proceedings of the 7th International Conference on Real-time Computing Systems and Applications. IEEE Computer Society Press. 2000

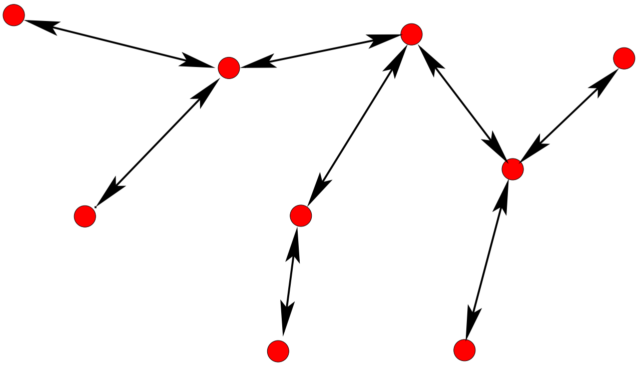
Références (4)

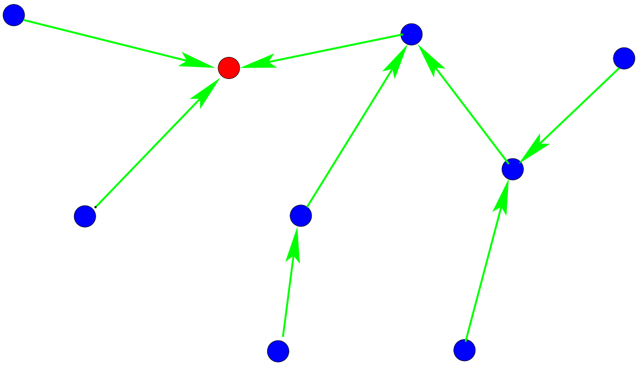
THEOREM PROVING

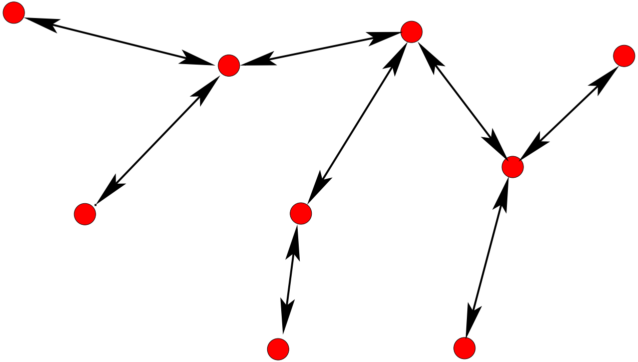
- M. Devillers et al. *Verification of the Leader Election : Formal Method Applied to IEEE 1394*. Formal Methods in System Design. 2000
- J.R. Abrial et al. *A Mechanically Proved and Incremental Development of IEEE 1394*. Formal Aspects of Computing, 2003.

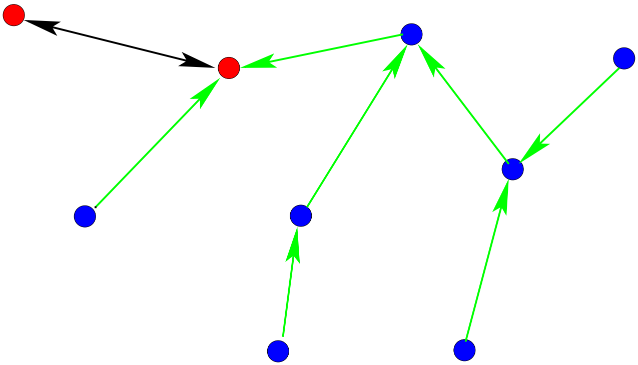
Propriétés informelles abstraites du protocole

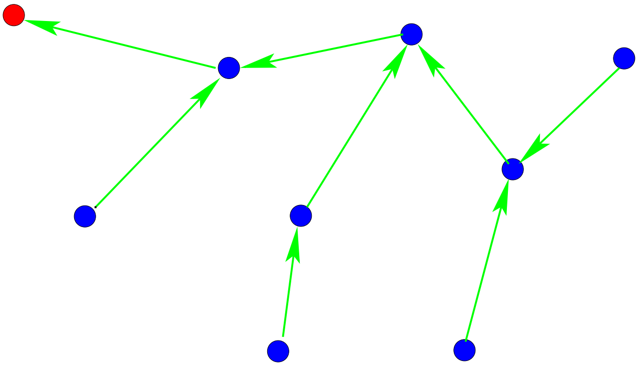
- Un réseau connexe et acyclique de nœuds.
- Nœuds reliés par des canaux bidirectionnels
- Election en un temps fini d'un leader.
- De manière répartie et non-déterministe.
- Un exemple d'animation du protocole.





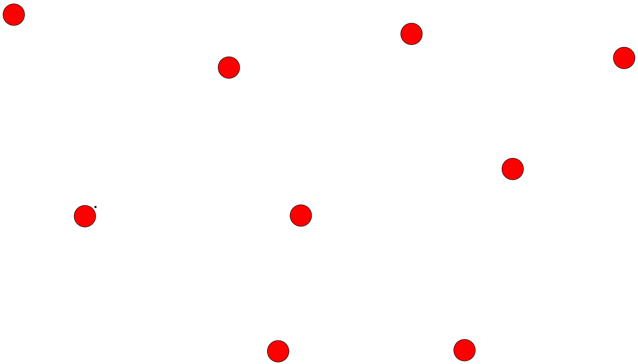




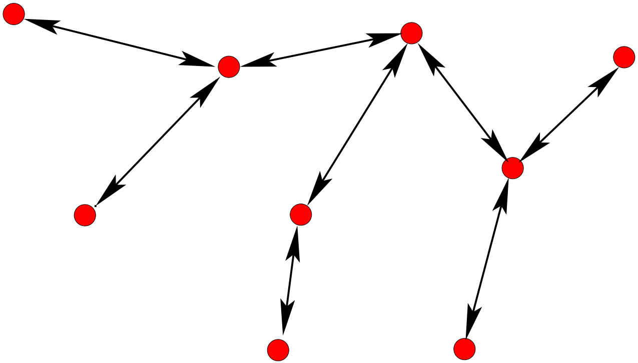


Le processus de développement

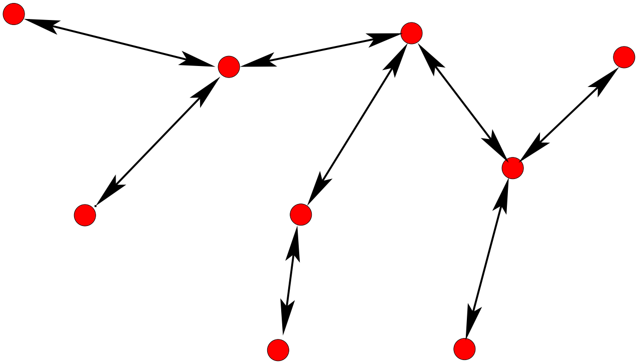
- Définition et propriétés du réseau dans un style formel
- Un modèle abstrait ¡¡one-shot¿¿ du protocole.
- Présentation d'une solution centralisée mais encore abstraite !
- Introduction du mécanisme de message passing entre les nœuds et des délais
- Modification de la structure de donnée pour répartir le protocole



ND un ensemble de noeuds (au moins 2 noeuds)



gr un graphe construit et défini sur ND



gr est un graphe symétrique et non réflexif

gr : un graphe construit sur ND $\qquad gr \subseteq ND \times ND$

gr : un graphe construit sur ND $\qquad gr \subseteq ND \times ND$

gr est défini sur ND $\qquad \mathrm{dom}\,(gr) = ND$

gr : un graphe construit sur ND $\qquad gr \subseteq ND \times ND$

gr est défini sur ND $\qquad \mathrm{dom}\,(gr) = ND$

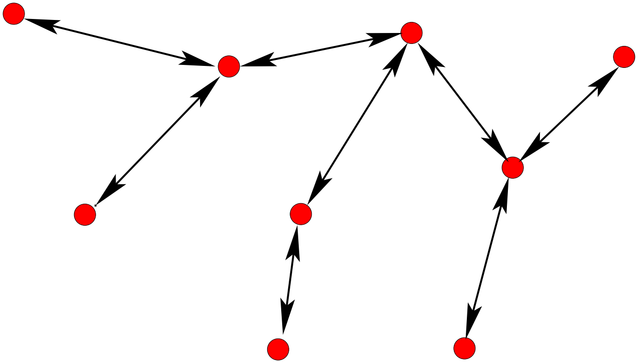
gr est symétrique $\qquad gr = gr^{-1}$

gr : un graphe construit sur ND $\quad gr \subseteq ND \times ND$

gr est défini sur ND $\quad \text{dom}\,(gr) = ND$

gr est symétrique $\quad gr = gr^{-1}$

gr est non réflexif $\quad \text{id}\,(ND) \cap gr =$



gr est connexe et acyclique

Un détour par les Arbres

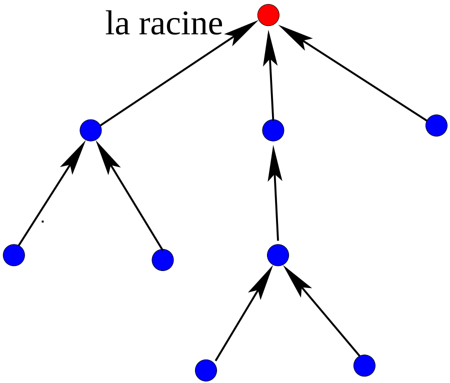
- Un arbre est un graphe particulier

- Un arbre a une racine

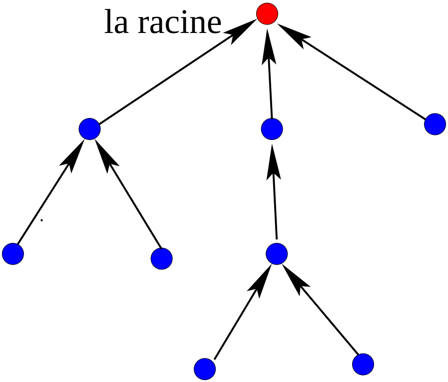
- Un arbre a une fonction parentale

- Un arbre est acyclique

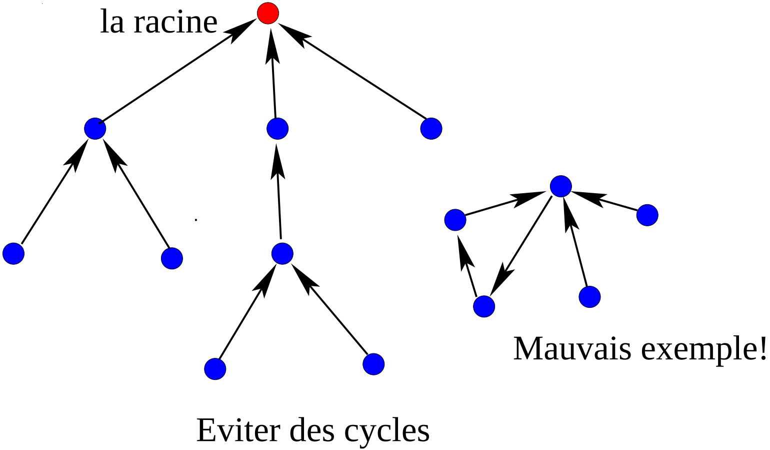
- Un arbre est connexe à partir de la racine

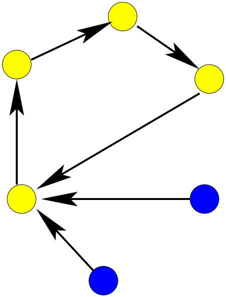


Un arbre t construit sur les noeuds

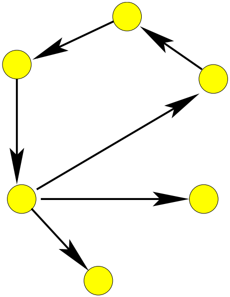


t est une fonction définie sur ND sauf pour la racine!


la racine
Mauvais exemple!
Eviter des cycles



Un cycle



Son image inverse

Les noeuds d'un cycle sont inclus
dans leur image inverse

Le prédicat tree (r, t)

Le prédicat tree (r, t)

r est un élément de ND $\quad r \in ND$

Le prédicat tree (r, t)

r est un élément de ND $\quad r \in ND$

t est une fonction $\quad t \in ND - \{r\} \rightarrow ND$

Le prédicat tree (r, t)

r est un élément de ND $\quad r \in ND$

t est une fonction $\quad t \in ND - \{r\} \rightarrow ND$

t est acyclique $\quad \forall p \cdot \left(\begin{array}{l} p \subseteq ND \;\wedge \\ p \subseteq t^{-1}[p] \\ \Longrightarrow \\ p = \varnothing \end{array} \right)$

t est acyclique : formulations équivalentes

$$\forall p \cdot \begin{pmatrix} p \subseteq ND \;\wedge \\ p \subseteq t^{-1}[p] \\ \Longrightarrow \\ p = \varnothing \end{pmatrix} \quad \Leftrightarrow \quad \forall q \cdot \begin{pmatrix} q \subseteq ND \;\wedge \\ r \in q \;\wedge \\ t^{-1}[q] \subseteq q \\ \Longrightarrow \\ ND \subseteq q \end{pmatrix}$$

On obtient une Règle d'Induction

$$\forall q \cdot \left(\begin{array}{l} q \subseteq ND \;\wedge \\ r \in q \;\wedge \\ \forall x \cdot (\, x \in ND - \{r\} \;\wedge\; t(x) \in q \implies x \in q\,) \\ \implies \\ ND \subseteq q \end{array} \right)$$

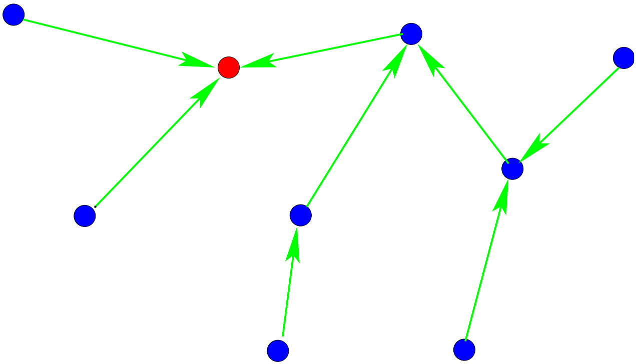
Le prédicat tree (r, t)

r est un élément de ND $\quad r \in ND$

t est une fonction $\quad t \in ND - \{r\} \rightarrow ND$

t est acyclique

$$\forall q \cdot \left(\begin{array}{l} q \subseteq ND \;\wedge \\ r \in q \;\wedge \\ t^{-1}[q] \subseteq q \\ \Longrightarrow \\ ND \subseteq q \end{array} \right)$$



Un arbre de recouvrement t de gr

Le prédicat spanning (r, t, gr)

r, t est un arbre	tree (r, t)
t est inclus dans gr	$t \subseteq gr$

Le graphe gr est connexe et acyclique (1)

- Définir une relation fn liant un noeud aux possibles spanning trees de gr ayant ce noeud comme racine :

$$fn \subseteq ND \times (ND \mathrel{+\!\!\!\!\rightarrow} ND)$$

$$\forall(r,t) \cdot \left(\begin{array}{l} \quad r \in ND \ \wedge \\ \quad t \in ND \mathrel{+\!\!\!\!\rightarrow} ND \\ \Longrightarrow \\ \quad (r,t) \in fn \ \Leftrightarrow \ \text{spanning}\,(r,t,gr) \end{array} \right)$$

Le graphe gr est connexe et acyclique (2)

Totalité de la relation fn $\Longrightarrow$ Connectivité of gr

Fonctionnalité d'une relation fn $\Longrightarrow$ Acyclicité de gr

Point sur les constantes gr and fn

$$gr \subseteq ND \times ND$$
$$\mathrm{dom}\,(gr) = ND$$
$$gr = gr^{-1}$$
$$\mathrm{id}\,(ND) \;\cap\; gr \;=\; \varnothing$$

$$fn \in ND \rightarrow (ND \nrightarrow ND)$$

$$\forall (r,t) \cdot \left(\begin{array}{l} r \in ND \;\wedge \\ t \in ND \nrightarrow ND \\ \Longrightarrow \\ t = fn(r) \;\Leftrightarrow\; \mathrm{spanning}\,(r,t,gr) \end{array} \right)$$

Tree

- Variables rt et ts

$$rt \in ND$$
$$ts \in ND \leftrightarrow ND$$

$$\begin{array}{l} \text{EVENT elect} \quad \widehat{=} \\ \quad \textbf{BEGIN} \\ \quad\quad rt, ts : \text{spanning}\,(rt, ts, gr) \\ \quad \textbf{END} \end{array}$$

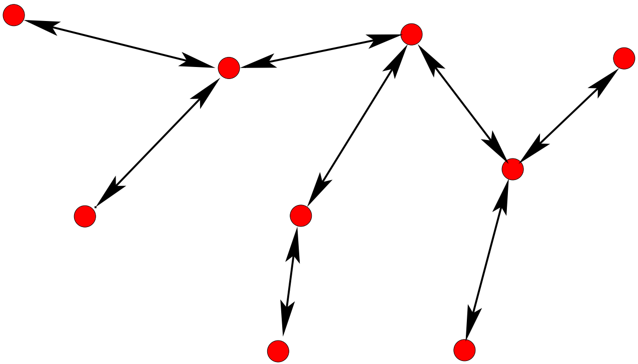
Premier raffinement (1)

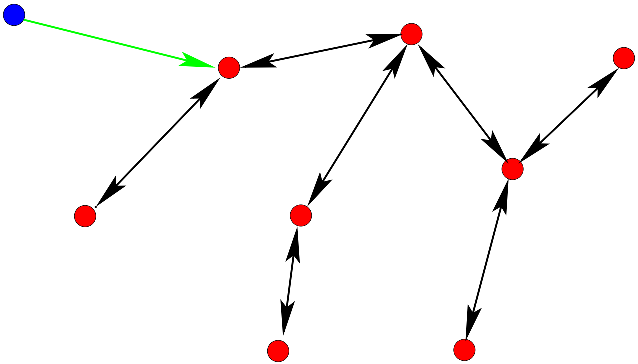
- Introduire une nouvelle variable, tr, correspondant à l'
  "arbre" en construction

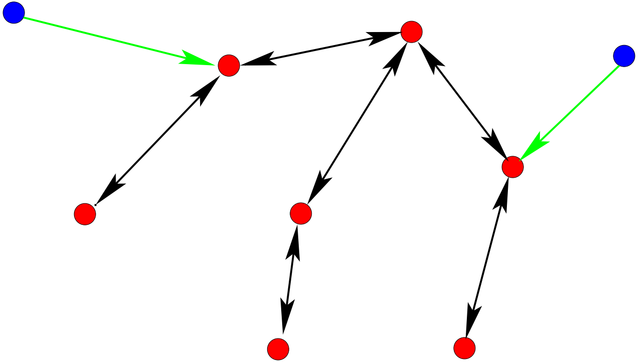
- Introduire un nouvel événement : progression

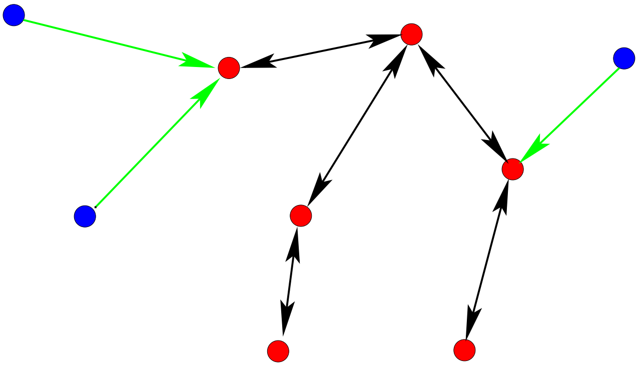
- Définir l'invariant

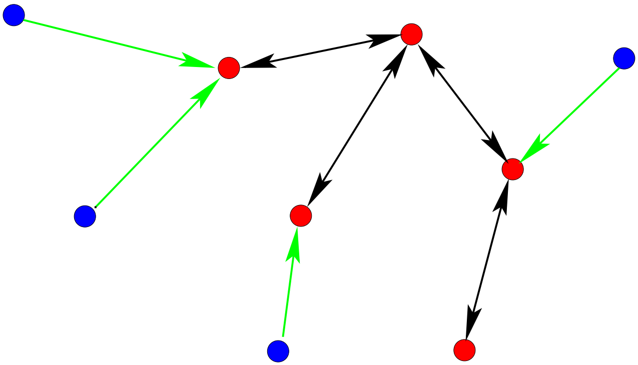
- Retour à l'animation : Observez la construction
  de l'arbre

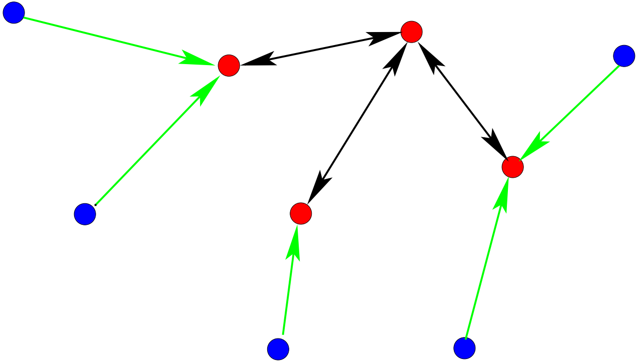


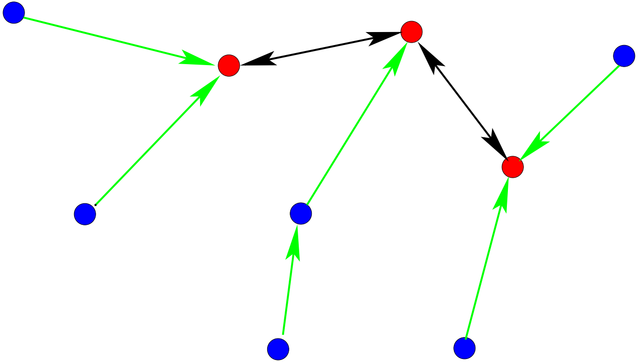


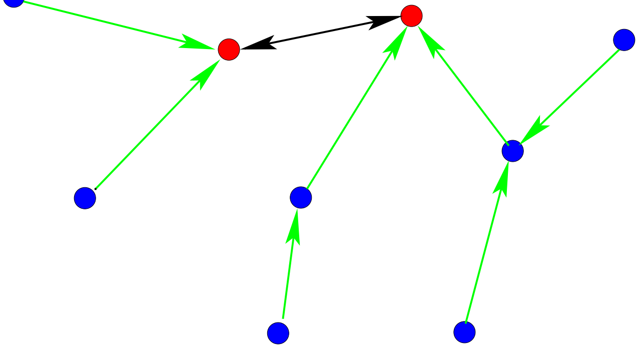


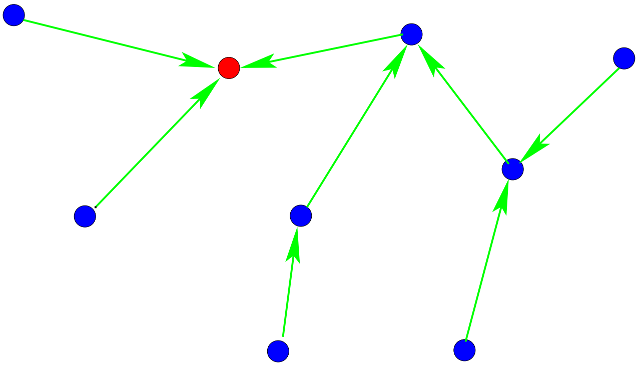












- Les flèches vertes correspondent à la fonction tr

- Les nœuds bleus sont le domaine de tr

- La fonction tr est une forêt sur les nœuds

- Les nœuds rouges sont les racines de ce arbres

Le prédicat invariant (tr)

$$tr \in ND \nrightarrow ND$$

Le prédicat invariant (tr)

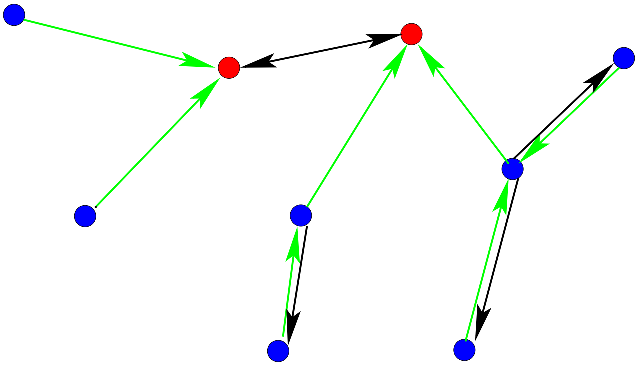
$$tr \in ND \prarr ND$$

$$\forall p \cdot \left(\begin{array}{l} \quad p \subseteq ND \qquad \wedge \\ \quad ND - \mathrm{dom}\,(tr) \subseteq p \qquad \wedge \\ \quad tr^{-1}\,[p] \subseteq p \\ \Longrightarrow \\ \quad ND \subseteq p \end{array} \right)$$

Le prédicat invariant (tr)

$$tr \in ND \nrightarrow ND$$

$$\mathsf{dom}\,(tr) \lhd (tr \cup tr^{-1}) \;=\; \mathsf{dom}\,(tr) \lhd gr$$

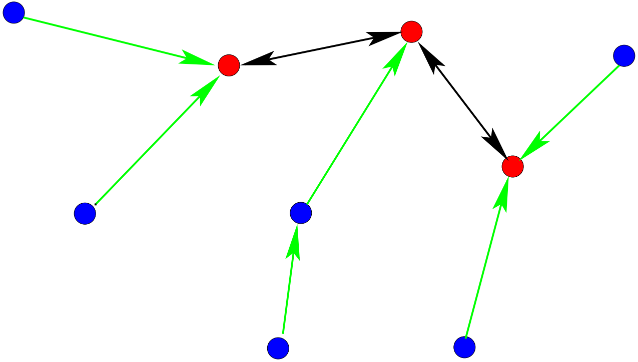


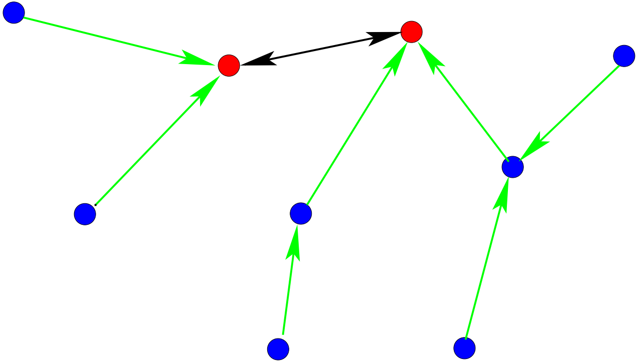
Premier raffinement (2)

- Introduire le nouvel événement "progress"

- Raffiner l'événement abstrait "elect"

- Retour à l'animation : Observez la "garde" de progress



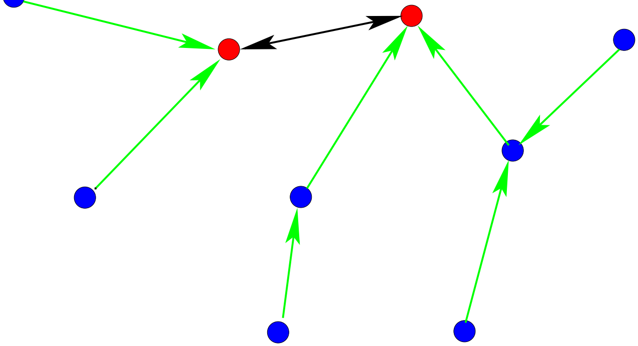


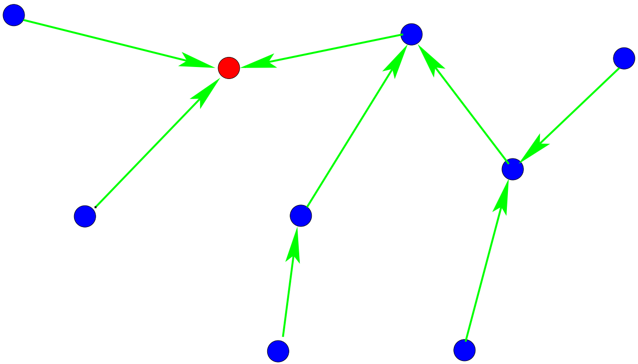
Quand un nœud rouge x est connecté à un autre nœud rouge y alors l'événement "progress" peut apparaître

$$
\begin{array}{l}
\text{EVENT progress} \;\; \widehat{=} \\
\quad \textbf{ANY } x, y \textbf{ WHERE} \\
\qquad x, y \in gr \;\; \wedge \\
\qquad x \notin \mathrm{dom}\,(tr) \;\; \wedge \\
\qquad y \notin \mathrm{dom}\,(tr) \;\; \wedge \\
\qquad gr[\{x\}] = tr^{-1}[\{x\}] \cup \{y\} \\
\quad \textbf{THEN} \\
\qquad tr := tr \cup \{x \mapsto y\} \\
\quad \textbf{END}
\end{array}
$$

Il faut prouver :

$$
\begin{array}{l}
\text{invariant}(tr) \quad \wedge \\
x, y \in gr \quad \wedge \\
x \notin tr \quad \wedge \\
y \notin tr \quad \wedge \\
gr[\{x\}] = tr^{-1}[\{x\}] \cup \{y\} \\
\Longrightarrow \\
\text{invariant}(tr \cup \{x \mapsto y\})
\end{array}
$$





Quand un nœud rouge x est SEULEMENT connecté aux nœuds bleus alors l'événement "elect" peut apparaître :

EVENT elect $\widehat{=}$
ANY x **WHERE**
$x \in ND \ \wedge$
$gr[\{x\}] = tr^{-1}[\{x\}]$
THEN
$rt, ts := x, tr$
END

$$\text{EVENT elect} \ \widehat{=}$$

BEGIN

$$rt, ts : \text{spanning}\,(rt, ts, gr)$$

END

$$\text{EVENT elect} \ \widehat{=}$$

ANY x **WHERE**

$$x \in ND \ \land$$

$$gr[\{x\}] = tr^{-1}[\{x\}]$$

THEN

$$rt, ts := x, tr$$

END

Il faut prouver :

$$
\begin{array}{l}
\quad \mathsf{invariant}(tr) \quad \wedge \\
\quad x \in ND \quad \wedge \\
\quad gr[\{x\}] = tr^{-1}[\{x\}] \\
\quad ts = tr \\
\Longrightarrow \\
\quad \mathsf{spanning}(x, ts, gr)
\end{array}
$$

Un lemme

$$\begin{array}{l} \mathsf{invariant}(tr) \quad \wedge \\ x \in ND \quad \wedge \\ gr[\{x\}] = tr^{-1}[\{x\}] \\ \Longrightarrow \\ tr = fn(x) \end{array}$$

Récapitulatif du premier raffinement

- 12 preuves

- Parmi lesquelles 5 interactives (une assez difficile !)

Second Raffinement

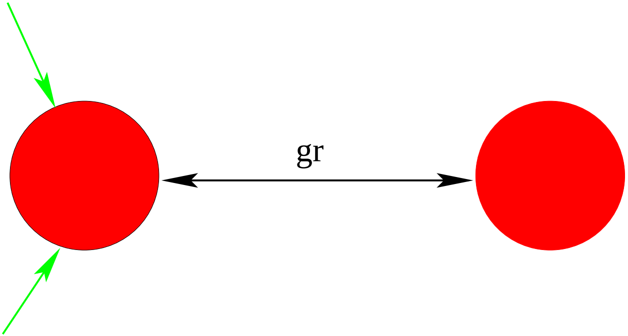
- Les nœuds communiquent avec leurs voisins

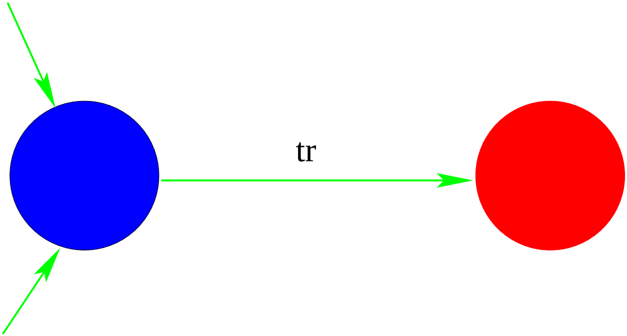
- On utilise des messages

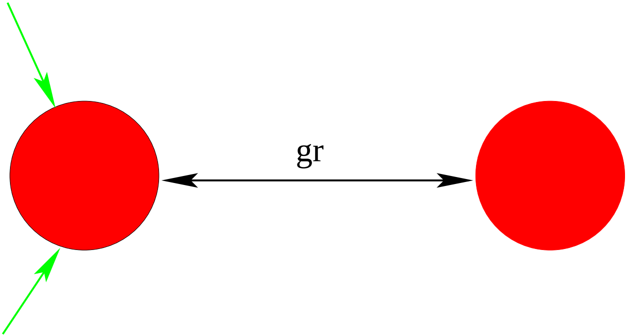
- Les messages ont des accusés de réception

- Les accusés de réception sont confirmés

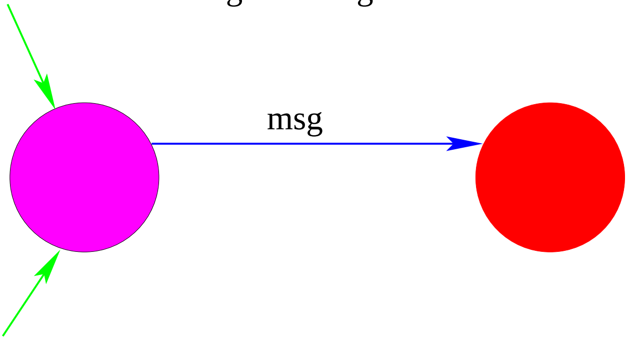
- Ensuite une animation locale

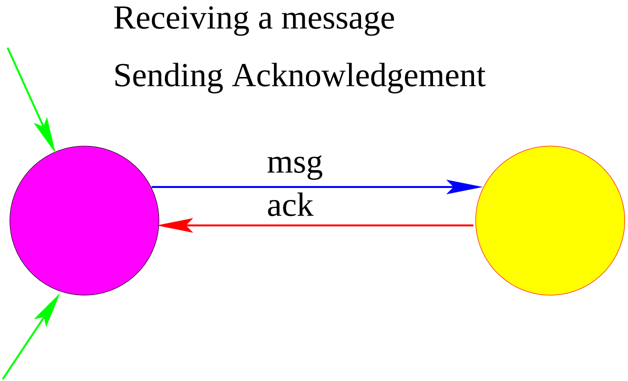

gr


tr


gr

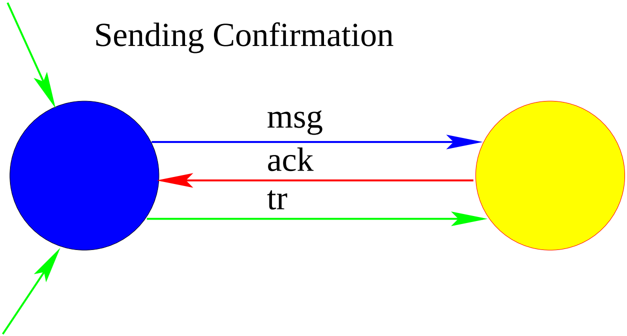
Sending a message

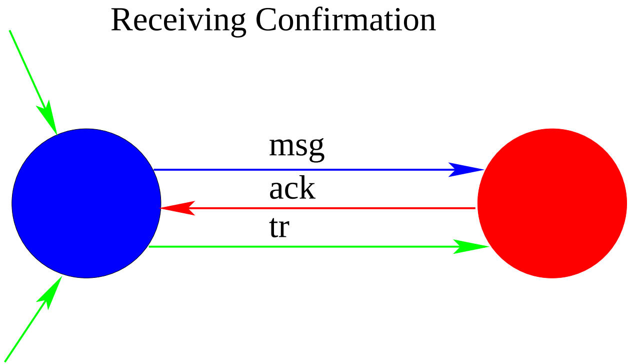



Receiving a message
Sending Acknowledgement
msg
ack

Receiving Acknowledgement

Sending Confirmation




Receiving Confirmation
msg
ack
tr

Invariant (1)

$$msg \in ND \nrightarrow ND$$

$$ack \in ND \nrightarrow ND$$

$$tr \subseteq ack \subseteq msg \subseteq gr$$

Nœud x envoie un message au nœud y

$$
\begin{array}{l}
\text{EVENT send\_msg} \;\widehat{=} \\
\quad \textbf{ANY } x, y \textbf{ WHERE} \\
\quad\quad x, y \in gr \;\wedge \\
\quad\quad gr[\{x\}] = tr^{-1}[\{x\}] \cup \{y\} \;\wedge \\
\quad\quad y, x \notin ack \;\wedge \\
\quad\quad x \notin \mathrm{dom}\,(msg) \\
\quad \textbf{THEN} \\
\quad\quad msg := msg \cup \{x \mapsto y\} \\
\quad \textbf{END}
\end{array}
$$

Nœud y envoie un acknowledgement au nœud x

$$
\begin{array}{l}
\mathrm{EVENT}\ \mathsf{send\_ack}\ \ \widehat{=}\\
\quad \textbf{ANY}\ x, y\ \textbf{WHERE}\\
\quad\quad x, y \in msg - ack\ \ \wedge\\
\quad\quad y \notin \mathsf{dom}\,(msg)\\
\quad \textbf{THEN}\\
\quad\quad ack := ack \cup \{x \mapsto y\}\\
\quad \textbf{END}
\end{array}
$$

Nœud x envoie une confirmation au nœud y

$$
\begin{array}{l}
\mathrm{EVENT}\ \textsf{progress}\ \ \widehat{=}\\
\quad \textbf{ANY}\ x, y\ \textbf{WHERE}\\
\qquad x, y \in ack\ \ \wedge\\
\qquad x \notin \mathsf{dom}\,(tr)\\
\quad \textbf{THEN}\\
\qquad tr := tr \cup \{x \mapsto y\}\\
\quad \textbf{END}
\end{array}
$$

Invariant (2)

$$\forall (x,y)\cdot \left(\begin{array}{l} x,y \in msg{-}ack \\ \Longrightarrow \\ x,y \in gr \quad \wedge \\ x \notin \mathrm{dom}\,(tr) \;\wedge\; y \notin \mathrm{dom}\,(tr) \;\wedge \\ gr[\{x\}] \;=\; tr^{-1}[\{x\}] \cup \{y\} \end{array}\right)$$

$$\forall (x,y)\cdot \left(\begin{array}{l} x,y \in ack \quad \wedge \\ x \notin \mathrm{dom}\,(tr) \\ \Longrightarrow \\ x,y \in gr \quad \wedge \\ y \notin \mathrm{dom}\,(tr) \;\wedge \\ gr[\{x\}] \;=\; tr^{-1}[\{x\}] \cup \{y\} \end{array}\right)$$

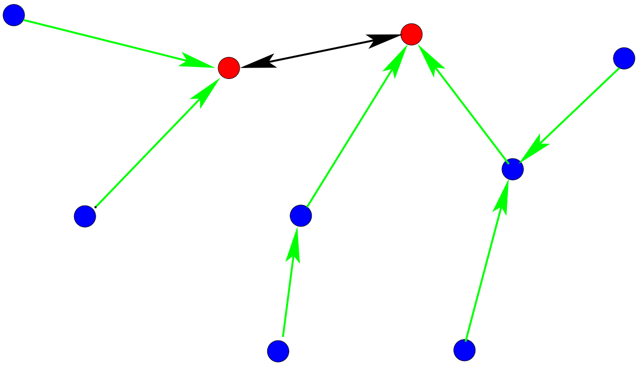
contention

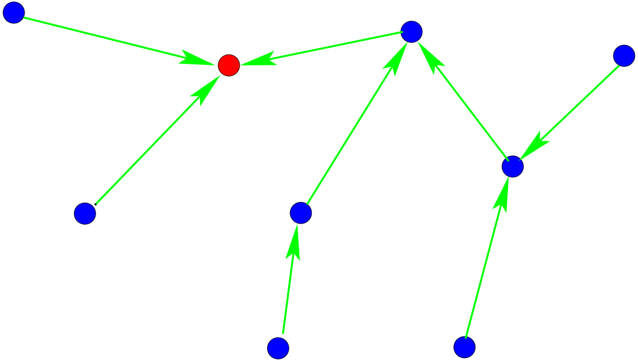
- Expliquer le problème

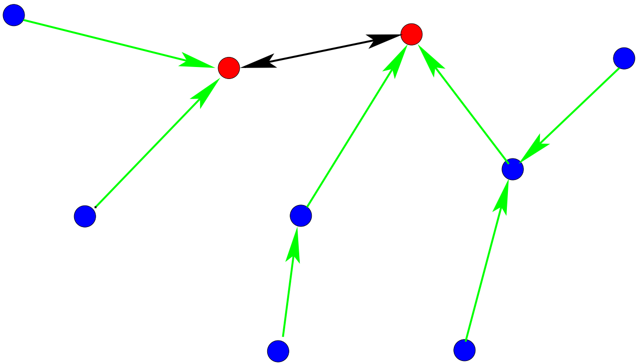
- Proposer une solution partielle.

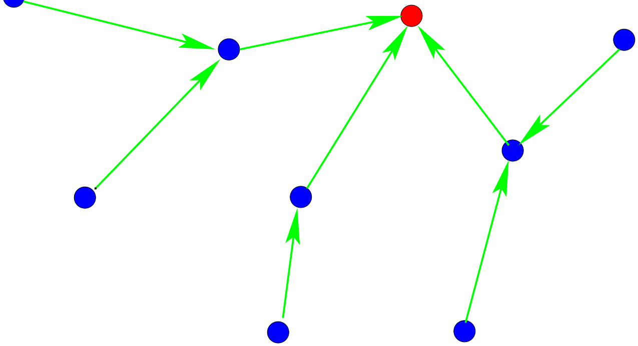
- Vers un meilleur traitement

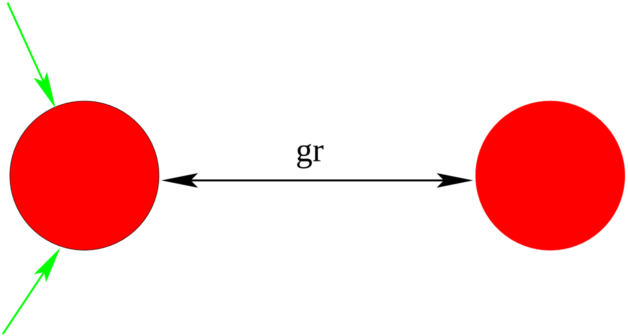
- Retour à l'animation locale.



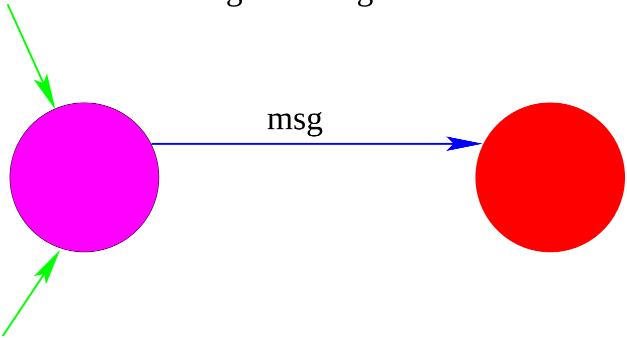


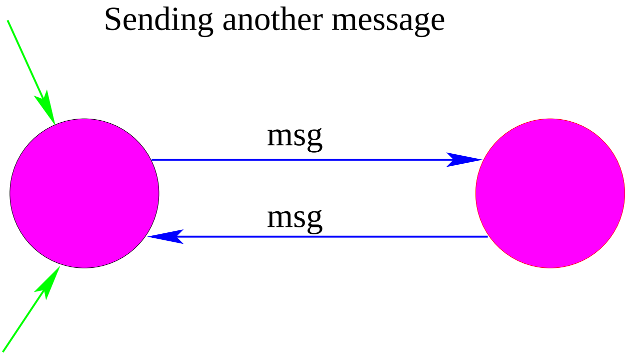


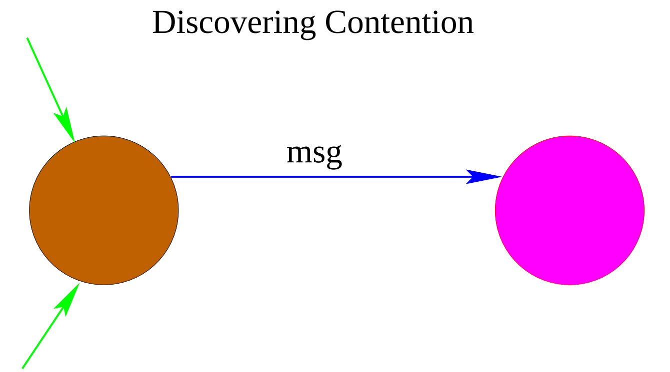



gr

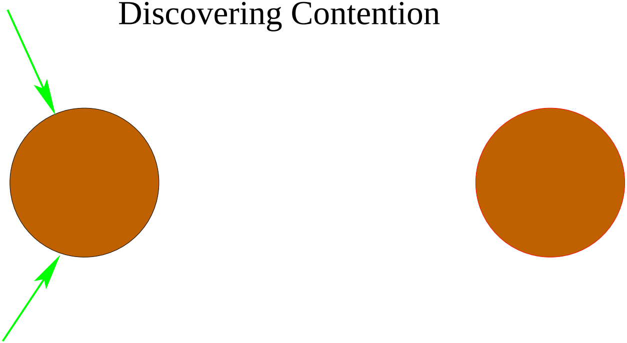
Sending a message



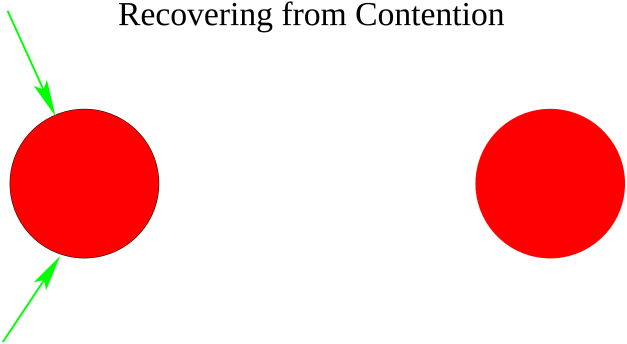

Sending another message
msg
msg


Discovering Contention
msg

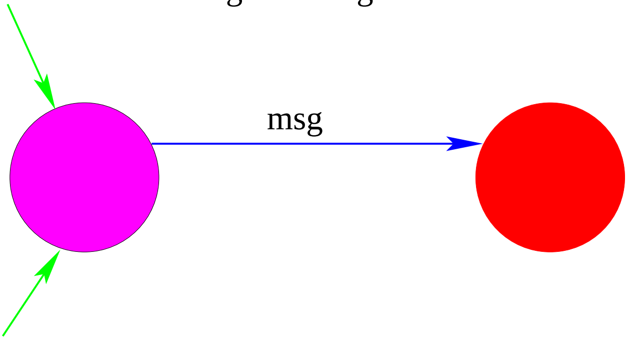
Discovering Contention

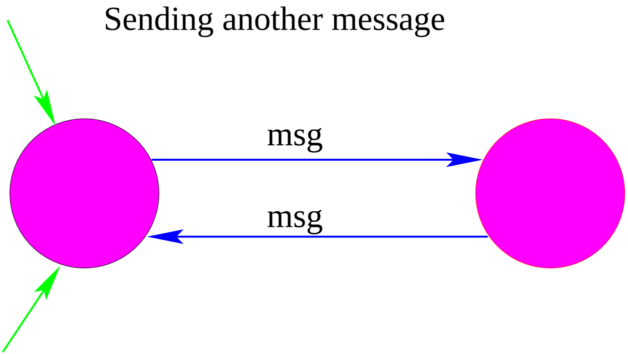


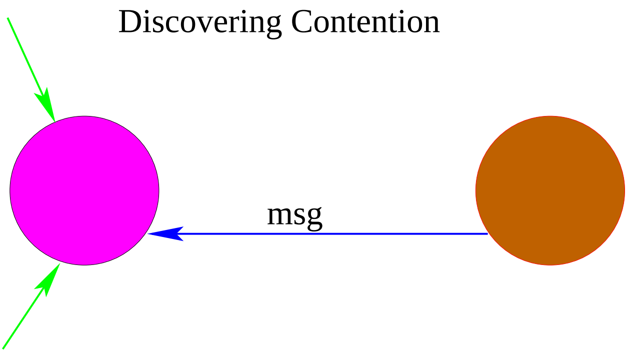
Recovering from Contention

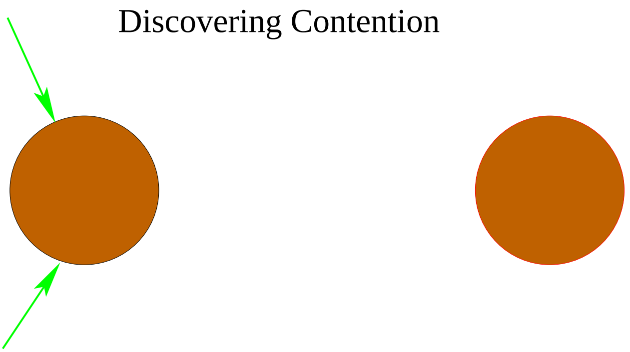


Sending a message

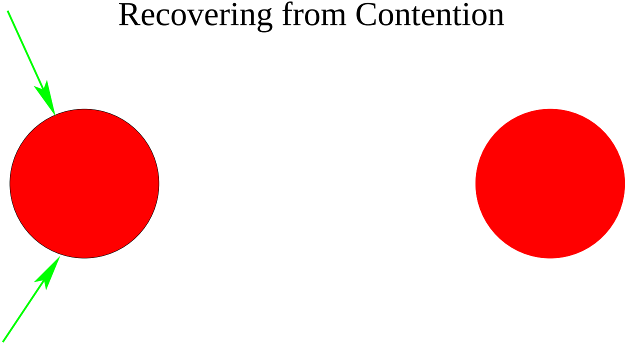


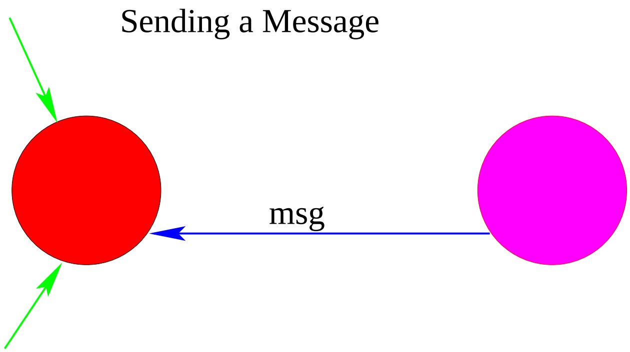

Sending another message
msg
msg


Discovering Contention
msg

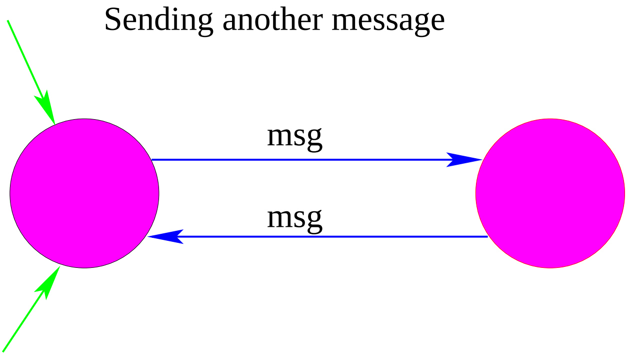

Discovering Contention

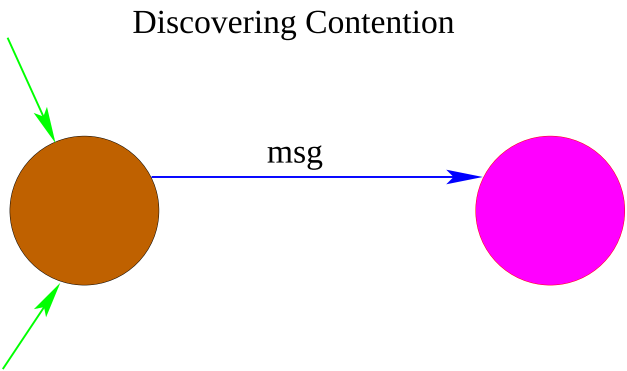
Recovering from Contention



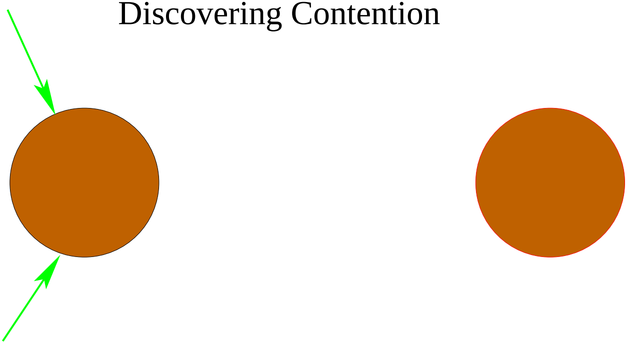


Sending a Message
msg

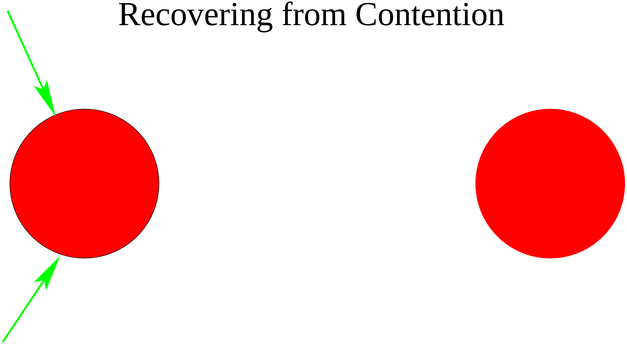

Sending another message
msg
msg

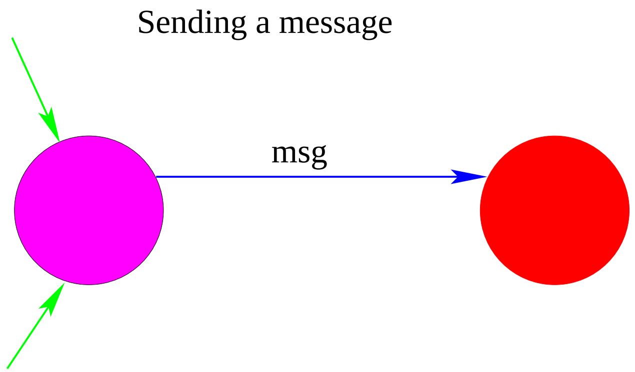

Discovering Contention
msg

Discovering Contention



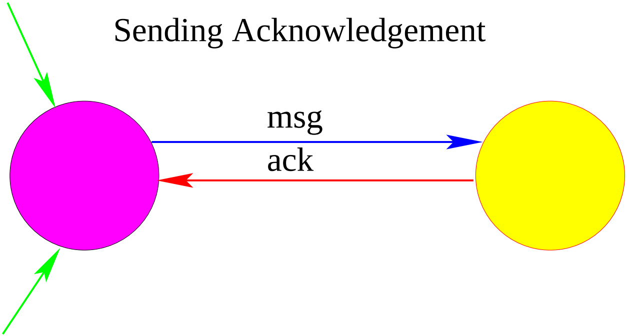
Recovering from Contention

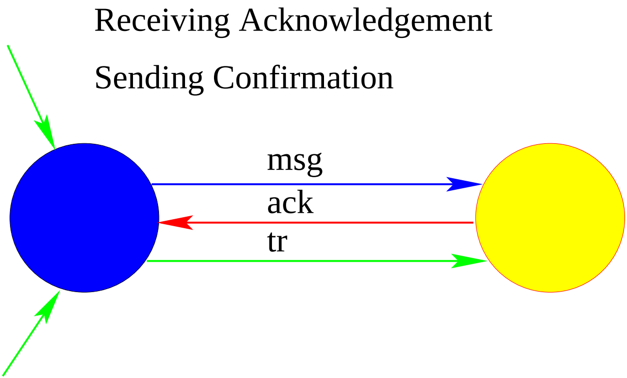


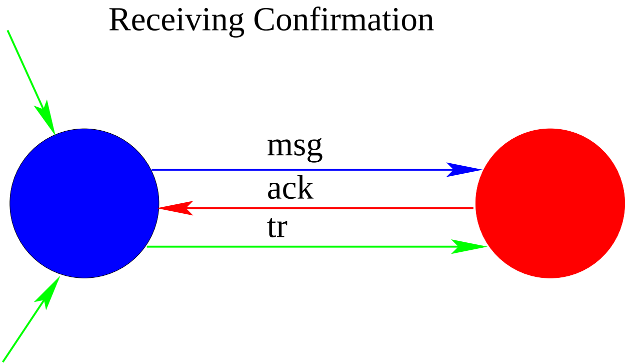

Sending a message
msg

Receiving a message

Sending Acknowledgement




Receiving Acknowledgement
Sending Confirmation
msg
ack
tr


Receiving Confirmation
msg
ack
tr

Découverte de la contention (1)

- Nœud y découvre la contention avec x car :
  - Il a envoyé un message à x
  - Il n'a pas encore reçu une réponse du nœud x
  - Il reçoit à la place un message de x

Découverte de la contention (2)

- x découvre aussi la contention avec y

- Hypothèse : Le temps entre deux découvertes
  EST SUPPOSÉ BORNÉ
  PAR τ ms

- Le temps τ est le temps maximum de transmission entre 2 noeuds connectés

Une solution partielle

- Chaque noeud attend τ ms après sa découverte

- Après cela, chaque noeud sait que l'autre
  a aussi découvert la contention

- Chaque noeud essaie à nouveau immédiatement

- PROBLEME : Cela peut continuer indéfiniment

Une meilleure solution (1)

- Tout nœud attend τ ms après sa propre découverte

- Chaque nœud choisit avec équiprobabilité :
  - soit d'attendre un délai court
  - soit d'attendre un délai long

- Chaque nœud essaie à nouveau

nœud x

$$
\begin{array}{l}
\text{EVENT send\_ack} \;\widehat{=}\\
\quad \textbf{ANY}\; x, y \;\textbf{WHERE}\\
\qquad x, y \in msg{-}ack \;\wedge\\
\qquad y \notin \mathrm{dom}\,(msg)\\
\quad \textbf{THEN}\\
\qquad ack := ack \cup \{x \mapsto y\}\\
\quad \textbf{END}
\end{array}
$$

$$
\begin{array}{l}
\text{EVENT contention} \;\widehat{=}\\
\quad \textbf{ANY}\; x, y \;\textbf{WHERE}\\
\qquad x, y \in msg{-}ack \;\wedge\\
\qquad y \in \mathrm{dom}\,(msg)\\
\quad \textbf{THEN}\\
\qquad cnt := cnt \cup \{x \mapsto y\}\\
\quad \textbf{END}
\end{array}
$$

Introduire un canal de contention factice : cnt

Invariant (3)

$$\forall (x, y) \cdot \left(\begin{array}{l} x, y \in msg{-}ack \ \ \wedge \\ y \in \mathsf{dom}\,(msg) \\ \Longrightarrow \\ y, x \in msg{-}ack \end{array} \right)$$

$$\forall (x, y, z) \cdot \left(\begin{array}{ll} x, y \in msg & \wedge \\ z \in gr[\{x\}] & \wedge \\ z \neq y & \\ \Longrightarrow & \\ z, x \in tr & \end{array} \right)$$

Invariant (4)

$$\forall (x,y) \cdot \left(\begin{array}{ll} x,y \in msg{-}ack & \wedge \\ y \notin \mathsf{dom}\,(msg) & \\ \Longrightarrow & \\ x,y \notin cnt & \end{array} \right)$$

$$ack \cap ack^{-1} = \varnothing$$

$$ack \cap cnt = \varnothing$$

Résolution de la contention (simuler le délai τ)

$$
\begin{array}{l}
\mathrm{EVENT}\ \textsf{solve\_contention} \quad \widehat{=} \\
\quad \textbf{ANY}\ x, y\ \textbf{WHERE} \\
\qquad x, y \in cnt \cup cnt^{-1} \\
\quad \textbf{THEN} \\
\qquad msg := msg - cnt \quad \| \\
\qquad cnt := \varnothing \\
\quad \textbf{END}
\end{array}
$$

Récapitulatif du Second Raffinement

- 63 preuves

- Parmi lesquelles 31 interactives

développement

- Etablir le cadre mathématique
- Résoudre le problème mathématique en un coup.
- Résoudre le même problème progressivement.
- Inclure les communications par des messages.
- Localiser les structures de donnée
- Méthodologie fondée sur le raffinement et la preuve
- Autres développements : algorithmique répartie, algorithmique séquentielle, systèmes, SoCs, ...

The predicate invariant (tr)

$tr \in ND \nrightarrow ND$

The predicate invariant (tr)

$$tr \in ND \nrightarrow ND$$

$$\mathsf{dom}\,(tr) \lhd (tr \cup tr^{-1}) \;=\; \mathsf{dom}\,(tr) \lhd gr$$

Invariant (3)

$$\forall (x,y) \cdot \left(\begin{array}{ll} x,y \in msg & \wedge \\ y,x \in msg & \\ \Longrightarrow & \\ y,x \notin ack & \end{array} \right)$$

$$\forall (x,y,z) \cdot \left(\begin{array}{ll} x,y \in msg & \wedge \\ z \in gr[\{x\}] & \wedge \\ z \neq y & \\ \Longrightarrow & \\ z,x \in tr & \end{array} \right)$$

Invariant (4)

$$\forall (x,y) \cdot \left(\begin{array}{ll} x,y \in cnt & \\ \Longrightarrow & \\ x,y \in msg - ack & \wedge \\ y \in \mathsf{dom}\,(msg) & \end{array} \right)$$

$$ack \cap ack^{-1} = \varnothing$$

$$ack \cap cnt = \varnothing$$

Third Refinement : Localization

- The representation of the graph gr is modified

- The representation of the tree tr is modified

- Other data structures are localized

Localization (1)

The graph gr and the tree tr are now localized

$$nb \in ND \to \mathbb{P}(ND)$$

$$\forall x \cdot (\, x \in ND \implies nb(x) = gr[\{x\}]\,)$$

$$sn \in ND \to \mathbb{P}(ND)$$

$$\forall x \cdot (\, x \in ND \implies sn(x) \subseteq tr^{-1}[\{x\}]\,)$$

Localization (2)

$$bm \subseteq ND$$

$$bm = \mathsf{dom}\,(msg)$$

$$bt \subseteq ND$$

$$bt = \mathsf{dom}\,(tr)$$

$$ba \in ND \rightarrow \mathbb{P}(ND)$$

$$\forall x \cdot (\, x \in ND \implies ba(x) = ack^{-1}[\{x\}]\,)$$

- Node x is elected the leader

EVENT elect $\widehat{=}$
  ANY x **WHERE**
    $x \in ND \ \wedge$
    $nb(x) = sn(x)$
  THEN
    $rt := x$
  END

- Node x sends a message to node y (y is unique)

$$
\begin{array}{l}
\text{EVENT } \mathsf{send\_msg} \quad \widehat{=} \\
\quad \textbf{ANY } x, y \textbf{ WHERE} \\
\quad\quad x \in ND - bm \ \wedge \\
\quad\quad y \in ND - ba(x) \ \wedge \\
\quad\quad nb(x) = sn(x) \cup \{y\} \\
\quad \textbf{THEN} \\
\quad\quad msg := msg \cup \{x \mapsto y\} \quad \| \\
\quad\quad bm := bm \cup \{x\} \\
\quad \textbf{END}
\end{array}
$$

- Node y sends an acknowledgement to node x

$$
\begin{array}{l}
\text{EVENT send\_ack} \quad \widehat{=} \\
\quad \textbf{ANY } x, y \textbf{ WHERE} \\
\quad\quad x, y \in msg \ \wedge \\
\quad\quad x \notin ba(y) \ \wedge \\
\quad\quad y \notin bm \\
\quad \textbf{THEN} \\
\quad\quad ack := ack \cup \{x \mapsto y\} \quad \| \\
\quad\quad ba(y) := ba(y) \cup \{x\} \\
\quad \textbf{END}
\end{array}
$$

- Node x sends a confirmation to node y

$$
\begin{array}{l}
\text{EVENT progress} \quad \widehat{=} \\
\quad \textbf{ANY } x, y \textbf{ WHERE} \\
\quad\quad x, y \in ack \ \wedge \\
\quad\quad x \notin bt \\
\quad \textbf{THEN} \\
\quad\quad tr := tr \cup \{x \mapsto y\} \quad \| \\
\quad\quad bt := bt \cup \{x\} \\
\quad \textbf{END}
\end{array}
$$

- Node y receives confirmation from node x

$$
\begin{array}{l}
\mathrm{EVENT}\ \textsf{rcv\_cnf}\ \ \widehat{=} \\
\quad \textbf{ANY}\ x,y\ \textbf{WHERE} \\
\quad\quad x,y \in tr\ \wedge \\
\quad\quad x \notin sn(y) \\
\quad \textbf{THEN} \\
\quad\quad sn(y) := sn(y) \cup \{x\} \\
\quad \textbf{END}
\end{array}
$$

$$
\begin{array}{l}
\text{EVENT } \textsf{contention} \quad \widehat{=} \\
\quad \textbf{ANY } x, y \textbf{ WHERE} \\
\quad\quad x, y \in cnt \cup cnt^{-1} \;\; \wedge \\
\quad\quad x \notin ba(y) \;\; \wedge \\
\quad\quad y \in bm \\
\quad \textbf{THEN} \\
\quad\quad cnt := cnt \cup \{x \mapsto y\} \\
\quad \textbf{END}
\end{array}
$$

$$
\begin{array}{l}
\text{EVENT } \mathsf{solve\_contention} \quad \widehat{=} \\
\quad \textbf{ANY } x, y \textbf{ WHERE} \\
\qquad x, y \in cnt \cup cnt^{-1} \\
\quad \textbf{THEN} \\
\qquad msg := msg - cnt \quad \| \\
\qquad bm := bm - \mathsf{dom}\,(cnt) \quad \| \\
\qquad cnt := \varnothing \\
\quad \textbf{END}
\end{array}
$$