1 Exclusion Mutuelle

1.1 Algorithmes basés sur les permissions

Algorithme 1 Alorithme de Ricart Agrawal, local au site i

$etat \longleftarrow S\ E, SC, S$ éttat du site i
$h \longleftarrow 0$ entier date des demandes
$last \longleftarrow 0$ entier date de la derniere demande
$attendu \longleftarrow \phi$ ensemble de sites qui attendent la permission
$differe \longleftarrow \phi$ ensemble de sites qui retardent l'envoi d'une perm
$priorit \longleftarrow faux$ booleen si i prioritaire ou non
demande d'entree en section critique
$etat \longleftarrow E$
$last \longleftarrow last + 1$
$attendu \longleftarrow R$
for all $j \in attendu$ **do**
  Envoi $msg(dem, (lasti, i))$ à j
end for
while $attendu \neq \phi$ **do**
  reception $msg(perm(j))$ de j
  $attendu \longleftarrow attendu\ j$
end while
$etati \longleftarrow SC$
Sortie de section critique
$etat \longleftarrow S$
for all $j \in differe$ **do**
  Envoi $msg(perm(i))$ à j
end for
$differei \longleftarrow \phi$
Réception de $msg(dem, (h', j))$ **de** j
$h \longleftarrow \max(h, h')$
$priorit \longleftarrow (etat \neq sortie) \wedge (last, i) < (h', j)$
if $priorit = vrai$ **then**
  $differe \longleftarrow differe \cup j$
else
  Envoi $msg(perm(i))$ à j
end if

Algorithme 2 Algorithme de Carvalho Roucairol, local au site i

$R \longleftarrow \{j \in V$ tel que i ne possedes pas $perm(i,j)\}$
$etat \longleftarrow S\ E, SC, S$ etat du site i
$h \longleftarrow 0$ entier date des demandes
$last \longleftarrow 0$ entier date de la derniere demande
$differe \longleftarrow \phi$ ensemble de sites qui retardent l'envoi d'une permission
$priorit \longleftarrow faux$ booleen si i prioritaire ou non
Demande d'entree en section critique
$etat \longleftarrow E$
$last \longleftarrow h + 1$
for all $j \in R$ **do**
  Envoi $msg(dem(last, i)$ à j
end for
while $R \neq \phi$ **do**
  Reception $msg(perm(i,j))$ de j
  $R \longleftarrow R \backslash \{j\}$
end while
$etat \longleftarrow SC$
Sortie
$etat \longleftarrow S$
for all $j \in differe$ **do**
  Envoi $msg(perm(i,j))$ à j
end for
$R \longleftarrow differe$
$differe \longleftarrow \emptyset$
Reception de $msg(dem, (h', j))$ **de** j
$h \longleftarrow \max(hi, h')$
$priorit \longleftarrow (etat = SC) \vee [(etat = E) \wedge (last, i) < (h', j)]$
if $priorit = VRAI$ **then**
  $differe \longleftarrow differe \cup \{j\}$
else
  Envoi $msg(perm(i,j))$ à j
  $R \longleftarrow R \cup \{j\}$
  if $etat = E$ **then**
    Envoi $msg(dem, (last, i))$ à j
  end if
end if

1.2 Algorithmes basés sur un jeton

Algorithme 3 Algorithme de Lelann, local au site i

$etat \longleftarrow S\ E, SC, S$
$suivant \longleftarrow$ le site voisin dans l'anneau
Reception du jeton
if $etat = E$ **then**
  $etat \longleftarrow SC$
else
  Envoi ($jeton$) à $suivant$
end if
sortie
$etat \longleftarrow S$
Envoi ($jeton$) à $suivant$

Algorithme 4 Algorithme de Ricart Agrawal, local au site i

$Etat \longleftarrow S\ E, SC, S$
$Nbdem \longleftarrow 0$ tableau de N entiers
$jetonpresent \longleftarrow faux$ booleen (si le site a le jeton) sauf pour un site
Demande d'entree en section critique
$Etat \longleftarrow E$
if $\neg jetonpresent$ **then**
  $NBdem[i] \longleftarrow Nbdem[i] + 1$
  for all $j \neq i$ **do**
    Envoi dem à j
  end for
  Réception de jeton
  $jetonpresent \longleftarrow VRAI$
end if
$Etat \longleftarrow SC$
Sortie de SC
$Etat \longleftarrow S$
$jeton[i] \longleftarrow Nbdem[i]$
if COND_ATT **then**
  Envoi $jeton$ à j
  $jetonpresent \longleftarrow FAUX$
end if
Reception de dem **de** j
$Nbdem[j] \longleftarrow Nbdem[j] + 1$
if $jetonpresent \wedge (Etat = S) \wedge COND\_ATT$ **then**
  Envoi $jeton$ à j
  $jetonpresent \longleftarrow FAUX$
end if
COND_ATT ($site\ i$)
$jeton[j] < Nbdem[j]$

2 Election

Algorithme 5 Algorithme de Lelann, local au site i

Variables
List ensemble de d'identifiants, init. à $\emptyset$
$etat \in \{elu, battu, init, noninit\}$ init. à *init* pour les initiateurs et *noninit* pour les autres
if $etat = noninit$ **then**
  while $false$ **do**
    Reception msg
    Envoi msg
    if $etat = noninit$ **then**
      $etat \longleftarrow battu$
    end if
  end while
else
  $list \longleftarrow list \cup \{i\}$
  Envoi $msg(i)$
  Recoit $msg(j)$
  while $i \neq j$ **do**
    $list \longleftarrow list \cup \{j\}$
    Envoi $msg(j)$
    Reception $msg(j)$
  end while
  if $i = \min list$ **then**
    $etat \longleftarrow elu$
  else
    $etat \longleftarrow battu$
  end if
end if

Algorithme 6 Algorithme de Chang Roberts, local au site i

```
Variables
etat ∈ {elu, battu, init, noninit} init. à init pour les initiateurs et noninit pour les autres
if etat = noninit then
  while false do
    Reception msg
    Envoi msg
    if etat = noninit then
      etat ⟵ battu
    end if
  end while
else
  Envoi msg(i)
  repeat
    Recoit msg(j)
    if i = j then
      etat ⟵ elu
    else
      if j < i then
        if etat = init then
          etat ⟵ battu
        end if
        Envoi msg(j)
      end if
    end if
  until etat = elu
end if
```

Algorithme 7 Election Peterson Dolev Rodeh Klawe, Algo local au site i

```
/* Variables*/
c : identifiant initialisé à i
acn : identifiant initialisé à ndef
win : identifiant initialisé à ndef
state : {actif, passif, leader, battu} initialisé à actif
/* Instructions*/
if i est un initiateur then
  state ⟵ actif
else
  state ⟵ passif
end if
while win = ndef do
  if state = actif then
    Envoi (un, c), Recoit (un, q)
    acn ⟵ q
    if acn = c then
      Envoi (small, acn), Recoit (small, q)
      win ⟵ acn
    else
      Envoi (deux, acn), Recoit (deux, q)
      if (acn < c) ∧ (acn < q) then
        c ⟵ acn
      else
        state ⟵ passif
      end if
    end if
  else
    Recoit (un, q), Envoi (un, q)
    Recoit m, Envoi m
    if m est un (small, q) message then
      win ⟵ q
    end if
  end if
end while
if i = win then
  state = leader
else
  state = battu
end if
```

3 Algorithmes a vagues

Algorithme 8 Vague sur un anneau

Site Initiateur
Envoi $jeton$
Recoit $jeton$
Prise de decision
Site Non Initiateur
Recoit $jeton$
Envoi $jeton$

Algorithme 9 Algorithme de PIF (Propogation of Information with Feedback) sur le site i

Variables
$Nrec \longleftarrow 0$
$pere \longleftarrow ndef$

Site Initiateur
for all $j \in Vois$ **do**
  Envoi $jeton$ a j
end for
while $Nrec < \#Vois$ **do**
  Reception($jeton$)
  $Nrec \longleftarrow Nrec + 1$
end while
Prise de Decision

Site Non Initiateur
Reception $jeton$ de j
$pere \longleftarrow j$
$Nrec \longleftarrow Nrec + 1$
for all $j \in Vois\ pere$ **do**
  Envoi $jeton$ a j
end for
while $Nrec < \#Vois$ **do**
  Reception($jeton$)
  $Nrec \longleftarrow Nrec + 1$
end while
Envoi $jeton$ a $pere$

Algorithme 10 Algorithme de parcours, Algo local au site i

Variables
$Pere$: identifiant de site, initialisé à $ndef$
$utilise[]$: tableau de booléen (indicé par l'identifiant des voisins), initialisé à $Faux$

Initiateur seulement
$Pere \longleftarrow i$
$Choisir j_0 \in Vois$
$utilise[j_0] \longleftarrow Vrai$
Envoi jeton à j_0

Tous les sites
Reception jeton de j_0
if $Pere = ndef$ **then**
  $Pere \longleftarrow j_0$
end if
if $\forall j \in Vois, utilise[j] = Vrai$ **then**
  Décision
else
  if $\exists j \in Vois, j \neq Pere \wedge \neg utilise[j]$ **then**
    Choisir $j \in Vois \backslash \{Pere\} \wedge \neg utilise[j]$
    $utilise[j] \longleftarrow Vrai$
    Envoi jeton à j
  else
    $utilise[Pere] \longleftarrow Vrai$
    Envoi jeton à j
  end if
end if

Algorithme 11 Parcours DFS, Algo local au site i

Variables
$Pere$: identifiant de site, initialisé à $ndef$
$utilise[]$: tableau de booléen (indicé par l'identifiant des voisins), initialisé à $Faux$

Initiateur seulement
$Pere \longleftarrow i$
$Choisir j_0 \in Vois$
$utilise[j_0] \longleftarrow Vrai$
Envoi jeton à j_0

Tous les sites
Reception jeton de j_0
if $Pere = ndef$ **then**
  $Pere \longleftarrow j_0$
end if
if $\forall j \in Vois, utilise[j] = Vrai$ **then**
  Décision
else
  if $\exists j \in Vois, j \neq Pere \wedge \neg utilise[j]$ **then**
    if $Pere \neq j_0 \wedge \neg utilise[j_0]$ **then**
      $j \longleftarrow j_0$
    else
      Choisir $j \in Vois \backslash \{Pere\} \wedge \neg utilise[j]$
    end if
    $utilise[j] \longleftarrow Vrai$
    Envoi jeton à j
  else
    $utilise[Pere] \longleftarrow Vrai$
    Envoi jeton à j
  end if
end if

Algorithme 12 Election par Vagues, Algo local au site i

Variables
$utilise[j] \longleftarrow$ init. à $faux$
$varp$ init. à $undef$
elu init. à 0
Code initiateur
if $varp = ndef$ **then**
  $varp \longleftarrow i$
  $elu \longleftarrow i$
  Choix j dans $vois$
  $utilise[j] \longleftarrow vrai$
  Envoi $msg(elect, elu)$
end if

Code tout site
Reception de $msg(elect, val)$ de j_0
if $elu < val$ **then**
  $elu \longleftarrow val$
  $varp \longleftarrow j_0$
  for all $j \in vois$ **do**
    $utilise[j] \longleftarrow faux$
  end for
  $utilise[j0] \longleftarrow vrai$
  if $\exists k \in vois \backslash varp | \neg utilise[k]$ **then**
    choisir $k \in vois \backslash varp \wedge \neg utilise[k]$
    $utilise[k] \longleftarrow vrai$
    Envoi $msg(elect, elu)$ à k
  end if
else
  if $elu = val$ **then**
    $utilise[j_0] \longleftarrow vrai$
    Envoi de $msg(dejavu)$ à j_0
  end if
end if
Reception de $msg(pere)$ ou $msg(dejavu)$ de j
if $\exists k \in vois | \neg utilise[k]$ **then**
  $utilise[k] \longleftarrow vrai$
  Envoi de $msg(elect, elu)$ à k
else
  if $varp \neq i$ **then**
    Envoi $msg(pere)$ à $varp$
  end if
end if

4 Algorithme Gallager Humblet Spira (83)

Algorithme 13 Algorithme GHS83

Bloc 1 : initialisation

Choix de (i, j) canal de i avec le plus petit poids
$canal[j] \longleftarrow branch$
$niv \longleftarrow 0$
$etat \longleftarrow found$
$recu \longleftarrow 0$
Envoi $(connect, 0)$ à j

Bloc 2 : Reception de $(connect, L)$ **de** j

if $L < niv$ **then**
  $canal[j] \longleftarrow branch$
  Envoi $(initiate, niv, nom, etat)$ à j
else
  if $canal[j] = basic$ **then**
    Traiter le message plus tard
  else
    Envoi $(initiate, niv + 1, poids(i, j), find)$ à j
  end if
end if

Bloc 3 : Reception de $(initiate, L, F, S)$ **de** j

$niv \longleftarrow L$
$nom \longleftarrow F$
$etat \longleftarrow S$
$pere \longleftarrow j$
$mcan \longleftarrow ndef$
$mpoids \longleftarrow \infty$
for all $k \in vois | canal[k] = branch \cap k \neq j$ **do**
  envoi $(initiate, L, F, S)$ à k
end for
if $etat = find$ **then**
  $recu \longleftarrow 0$
  Appel Procedure TEST
end if

Algorithme 14 Algorithme GHS83

Bloc 4 : Procedure TEST

if $\exists j \in vois | canal = basic$ **then**
  Choix de $j | canal[j] = basic \wedge poids(i,j)$ minimal pour tout $j \in Vois$
  $testcan \longleftarrow j$
  Envoi $(test, niv, nom)$ à $testcan$
else
  $testcan \longleftarrow udef$
  Appel Procedure REPORT
end if

Bloc 5 : Reception de $(test, L, F)$ **de** j

if $L > niv$ **then**
  Traiter le message plus tard
else
  if $F = nom$ **then**
    if $canal[j] = basic$ **then**
      $canal[j] \longleftarrow reject$
    end if
    if $j \neq testcan$ **then**
      Envoi $(reject)$ à j
    else
      Appel Procedure TEST
    end if
  else
    Envoi $(accept)$ à j
  end if
end if

Bloc 6 : Reception de $accept$ **de** j

$testcan \longleftarrow udef$
if $poids(i,j) < mpoids$ **then**
  $mpoids \longleftarrow poids(i,j)$
  $mcanal \longleftarrow j$
end if
Appel Procedure REPORT

Bloc 7 : Reception de $(reject)$ **de** j

if $canal[j] = basic$ **then**
  $canal[j] \longleftarrow reject$
end if
Appel Procedure TEST

Algorithme 15 Algorithme GHS83

Bloc 8 : Procedure REPORT

if $recu = \#\{j/canal[j] = branch \cap j \neq pere\} \cap testcan = undef$ **then**
  $etat \longleftarrow found$
  Envoi $(report, mpoids)$ à $pere$
end if
Bloc 9 : Reception de $(report, poids)$ **de** j

if $j \neq pere$ **then**
  if $poids < mpoids$ **then**
    $mpoids \longleftarrow poids$
    $mcan \longleftarrow j$
  end if
  $recu \longleftarrow recu + 1$
  Appel Procedure REPORT
else
  if $etat = find$ **then**
    Traiter le message plus tard
  else
    if $poids > mpoids$ **then**
      Appel Procedure CHANGEROOT
    else
      if $poids = mpoids = \infty$ **then**
        TERMINE
      end if
    end if
  end if
end if

Bloc 10 : Procedure CHANGEROOT

if $canal[mcan] = branch$ **then**
  Envoi $(changeroot)$ à $mcan$
else
  Envoi $(connect, niv)$ à $mcan$
  $canal[mcan] \longleftarrow branch$
end if

Bloc 11 : Reception de $(changeroot)$

Appel Procedure CHANGEROOT

5 Algorithmes de Routage

Algorithme 16 Algorithme de Bellman Ford (version centralisée pour le site i)

$\forall k, D_i^{(k)} = 0$
$D_j^{(0)} = \infty, \forall j \neq i$
for k de 0 à $n-2$ **do**
  $D_j^{k+1} = \min_{l \in V}(D_l^{(h)} + d_l j)$ avec $j \neq i$
end for

Algorithme 17 Algorithme de Floyd Warshall

Variables
D tableau d'entiers de 1 à n
Routage tableau d'entiers de 1 à n
Algo
$D[i] \longleftarrow 0$
$Routage[i] \longleftarrow ndef$
$\forall j \in VoisD[j] \longleftarrow w(ij)$
$\forall j \in VoisRoutage[j] \longleftarrow j$
$\forall j \in V \backslash \{Vois \cup \{i\}\} D[j] \longleftarrow \infty$
$\forall j \in V \backslash \{Vois \cup \{i\}\} Routage[j] \longleftarrow ndef$
for $k = 1$ à n **do**
  if $k = i$ **then**
    Diffuse D
  else
    Reception de D_k
    for all $j \in V$ **do**
      if $D[k] + D_k[j] < D[j]$ **then**
        $D[j] \longleftarrow D_k[j] + D[k]$
        $Routage[j] \longleftarrow Routage[k]$
      end if
    end for
  end if
end for

Algorithme 18 Algorithme de Tajibnapis 77 (Netchange Partie 1)

Variables

$Vois$: ens. de sites (initialisé aux voisins effectifs du site)
D : Tableau d'entiers de taille n
Nb : Tableau d'id de sites de taille n
$ndis$: Tableau d'entiers de taille n^2
Initialisation
for all $w \in Vois, v \in V$ **do**
  $ndis[w, v] \longleftarrow n$
end for
for all $v \in V$ **do**
  $D[v] \longleftarrow n$
  $Nb[v] \longleftarrow ndef$
end for
$D[i] \longleftarrow 0$
$Nb[i] \longleftarrow local$
for all $w \in Vois$ **do**
  $Envoi(Mydist, i, 0)$ à w
end for

Procedure Recompute (v)
if $v = i$ **then**
  $D[v] \longleftarrow 0$
  $Nb[v] \longleftarrow local$
else
  $d = 1 + \min_{w \in Vois}\{ndis[w, v]\}$
  if $d < n$ **then**
    $D[v] \longleftarrow d$
    $Nb[v] \longleftarrow w$ (avec $d = 1 + \min_{w \in Vois}\{ndis[w, v]\}$)
  else
    $D[v] \longleftarrow n$
    $Nb[v] \longleftarrow ndef$
  end if
end if
if $D[v]$ a changé **then**
  for all $x \in Vois$ **do**
    Envoi $(Mydist, v, D[v])$ à x
  end for
end if

Algorithme 19 Algorithme de Tajibnapis 77 (Netchange Partie 2)

Reception de $(Mydist, v, d)$ **d'un voisin** w
$ndis[w, v] \longleftarrow d$
$Recompute(v)$

Sur une defaillance du canal (i, w)
Recoit $(fail, w)$
$Vois \longleftarrow Vois \backslash \{w\}$
for all $v \in V$ **do**
  $Recompute(v)$
end for

Sur une reparation du canal (i, w)
Recoit $(repair, w)$
$Vois \longleftarrow Vois \cup \{w\}$
for all $v \in V$ **do**
  $ndis[w, v] \longleftarrow n$
  Envoi $(Mydist, v, D[v])$ à w
end for