Vérification des systèmes parallèles par la méthode des invariants

Liège, 1999-2000, 2002-2003

Pascal GRIBOMONT
Université de Liège
Belgium

Automatisation de la méthode des invariants : la théorie et la pratique

Pascal Gribomont
Université de Liège
Belgium

17-2-2000, 8-12-2000, 15-2-2002,

AFFECTATION

$$(x_1, \ldots, x_n) := (e_1, \ldots, e_n)$$

On exclut les cas d'échec.

Sémantique relationnelle :

$$(\sigma, \rho) \in R(X := E)$$
$$\equiv$$
$$[\forall v\,(v \notin X \Rightarrow \rho_v = \sigma_v) \wedge \forall i\,(\rho_{x_i} = \sigma_{e_i})]$$

Une affectation (et plus tard, une transition) est la spécification d'une transformation dans l'espace des états. On admet souvent le non-déterminisme, d'où on considérera plutôt des *ensembles* d'états.

Les ensembles d'états sont souvent représentés par des formules de logiques appelées *assertions*, voire identifiés à des assertions. Une transition transforme donc une assertion en une autre assertion. Un programme est une suite de transitions; s'il se termine, il transforme une assertion initiale, telle que

$$at\ \ell_0 \wedge x = n \wedge y = 1$$

en une assertion finale, telle que

$$at\ \ell_f \wedge x = 0 \wedge y = n!$$

TRIPLET DE HOARE

$$\{P\}\ X := E\ \{Q\}$$

Sémantique :

$$\forall\sigma\,\forall\rho\,[(\sigma \models P \wedge (\sigma,\rho) \in R(X := E)) \Rightarrow \rho \models Q]$$

$\{\mathit{false}\}\ S\ \{Q\}$: vrai

$\{P\}\ S\ \{\mathit{true}\}$: vrai

Si $P' \models P$, $\{P\}\ S\ \{Q\}$, $Q \models Q'$,

alors $\{P'\}\ S\ \{Q'\}$

PRECONDITION

$$\mathit{pre}[X := E;\ Q] = \{\, P :\ \{P\}\ X := E\ \{Q\}\,\}.$$

Les formules suivantes sont toutes équivalentes :

$\mathcal{A} \subset \mathit{pre}[X := E;\ Q]$,
$\bigwedge_{P\in\mathcal{A}}(P \in \mathit{pre}[X := E;\ Q])$,
$\forall P\,(P \in \mathcal{A} \Rightarrow [\,\{P\}\ X := E\ \{Q\}\,])$,
$\forall P\,(P \in \mathcal{A} \Rightarrow \forall\sigma\forall\rho\,[(\sigma \models P \wedge (\sigma,\rho) \in R(X := E)) \Rightarrow \rho \models Q])$,
$\forall\sigma\,\forall\rho\,\forall P\,[(P \in \mathcal{A} \wedge \sigma \models P \wedge (\sigma,\rho) \in R(X := E)) \Rightarrow \rho \models Q]$,
$\forall\sigma\,\forall\rho\,[\exists P\,(P \in \mathcal{A} \wedge \sigma \models P \wedge (\sigma,\rho) \in R(X := E)) \Rightarrow \rho \models Q]$,
$\forall\sigma\,\forall\rho\,[((\sigma \models \bigvee\mathcal{A}) \wedge (\sigma,\rho) \in R(X := E)) \Rightarrow \rho \models Q]$,
$\{\bigvee\mathcal{A}\}\ X := E\ \{Q\}$,
$\bigvee\mathcal{A} \in \mathit{pre}[X := E;\ Q]$.

L'ensemble $\mathit{pre}[X := E;\ Q]$ est un treillis booléen;
les bornes sont false et $\bigvee \mathit{pre}[X := E;\ Q]$.

PLUS FAIBLE PRECONDITION

Définition

$$\mathit{wlp}[X := E;\ Q] =_{\mathit{def}} \bigvee \mathit{pre}[X := E;\ Q]$$

Critère

$(\,\{P\}\ X := E\ \{Q\}\,) \equiv (P \Rightarrow \mathit{wlp}[X := E;\ Q])$

On a
$\{\mathit{wlp}[X := E;\ Q]\}\ X := E\ \{Q\}$.
on a aussi
$(P \Rightarrow \mathit{wlp}[X := E;\ Q]) \Rightarrow (\,\{P\}\ X := E\ \{Q\}\,)$.
Si $\mathcal{A}$ est un ensemble d'assertions, on a
$\forall P\,[P \in \mathcal{A} \Rightarrow (P \Rightarrow \bigvee\mathcal{A})]$,
et en particulier
$(\,\{P\}\ X := E\ \{Q\}\,) \Rightarrow (P \Rightarrow \mathit{wlp}[X := E;\ Q])$.

Soit B : pour tout P, $(\,\{P\}\ X := E\ \{Q\}\,) \equiv (P \Rightarrow B)$.
Pour $P = \mathit{wlp}[X := E;\ Q]$, on a $(\mathit{wlp}[X := E;\ Q] \Rightarrow B)$;
en choisissant $P = B$, on obtient $\{B\}\ X := E\ \{Q\}$,
d'où $(B \Rightarrow \mathit{wlp}[X := E;\ Q])$.

$\sigma \models \mathit{wlp}[X := E;\ Q] \equiv \forall\rho\,[(\sigma,\rho) \in R(X := E) \Rightarrow \rho \models Q]$

(PLUS FORTE) POSTCONDITION

$$\mathit{post}[P;\ X := E] = \{Q :\ \{P\}\ X := E\ \{Q\}\,\}.$$

$\mathit{post}[P;\ X := E]$ est un treillis booléen;
les bornes sont true et $\bigwedge \mathit{post}[P;\ X := E]$.

$$\mathit{slp}[P;\ X := E] =_{\mathit{def}} \bigwedge \mathit{post}[P;\ X := E]$$

$$(\,\{P\}\ X := E\ \{Q\}\,) \equiv (\mathit{slp}[P;\ X := E] \Rightarrow Q)$$

$\rho \models \mathit{slp}[P;\ X := E] \equiv \exists\sigma\,[\sigma \models P \wedge (\sigma,\rho) \in R(X := E)]$

FORMULAIRE

Trois formules équivalentes

$$P \;\Rightarrow\; wlp[X := E;\, Q]\,,$$
$$\{P\}\; X := E\; \{Q\}\,,$$
$$slp[P;\, X := E] \;\Rightarrow\; Q$$

Règle de conséquence

$$\frac{P' \;\Rightarrow\; P\,,\;\; \{P\}\; X := E\; \{Q\}\,,\;\; Q \;\Rightarrow\; Q'}{\{P'\}\; X := E\; \{Q'\}}$$

Règle de conjonction et disjonction :

les triplets de Hoare

$$\{P_i\}\; X := E\; \{Q_j\}\,,\;\; i \in I\,, j \in J$$

sont tous vrais si et seulement si le triplet

$$\{\bigvee_{i\in I} P_i\}\; X := E\; \{\bigwedge_{j\in J} Q_j\}$$

est vrai.

EXEMPLES

Triplets vrais

$$\{x = 5 \wedge y = 2\}\; x := x/y\; \{x = 2 \wedge y = 2\}\,,$$
$$\{y \neq 0\}\; x := x/y\; \{true\}\,,$$
$$\{false\}\; x := x/y\; \{x = 5 \wedge y = -2\}\,.$$

Triplets faux

$$\{x = 5 \wedge y = 2\}\; x := x/y\; \{x = 3\}\,,$$
$$\{x = x_0 \wedge y \neq 0\}\; x := x/y\; \{x * y = x_0\}\,,$$
$$\{y \neq 0\}\; x := x/y\; \{x = 5 \wedge y = -2\}\,.$$

Triplets "problématiques" (échec)

$$\{x = 5 \wedge y = 0\}\; x := x/y\; \{x = 2 \wedge y = 2\}\,,$$
$$\{y = y_0\}\; x := x/y\; \{y = y_0\}\,,$$
$$\{true\}\; x := x/y\; \{true\}\,.$$

REGLES DE CALCUL

$$wlp[x := f(x,y)\,;Q(x,y,z)] \;\equiv\; Q(f(x,y),y,z)$$

$$slp[P(x,y,z)\,;\; x := f(x,y)]$$
$$\equiv$$
$$\exists x_0\,[P(x_0,y,z) \wedge x = f(x_0,y)]$$

Preuve

$$\rho \;\models\; slp[P(x,y,z)\,;\; x := f(x,y)]\,,$$
$$\exists\sigma\,[\sigma \models P(x,y,z) \wedge (\sigma,\rho) \in R(x := f(x,y))]\,,$$
$$\exists\sigma\,[P(\sigma_x,\sigma_y,\sigma_z) \wedge (\rho_x,\rho_y,\rho_z) = (f(\sigma_x,\sigma_y),\sigma_y,\sigma_z)]\,,$$
$$\exists\sigma_x\,[P(\sigma_x,\rho_y,\rho_z) \wedge \rho_x = f(\sigma_x,\rho_y)]\,,$$
$$\rho \;\models\; \exists x_0\,[P(x_0,y,z) \wedge x = f(x_0,y)]\,.$$

TROIS CAS PARTICULIERS

$$slp[x = x_0 \wedge P(x_0,y,z)\,;\; x := f(x,y)]\,,$$
$$\equiv\;\; \exists x'_0\,[x'_0 = x_0 \wedge P(x'_0,y,z) \wedge x = f(x'_0,y)]\,,$$
$$\equiv\;\; [P(x_0,y,z) \wedge x = f(x_0,y)]\,.$$

$$slp[P(x,y,z)\,;\; x := e_0]\,,$$
$$\equiv\;\; \exists x_0\,[P(x_0,y,z) \wedge x = e_0]\,,$$
$$\equiv\;\; [\exists x_0\, P(x_0,y,z) \wedge x = e_0]\,.$$

$$slp[P(x,y,z)\,;\; x := f(x,y)]\,,$$
$$\equiv\;\; \exists x_0\,[P(x_0,y,z) \wedge x = f(x_0,y)]\,,$$
$$\equiv\;\; \exists x_0\,[P(x_0,y,z) \wedge x = f_y(x_0)]\,,$$
$$\equiv\;\; P(f_y^{-1}(x),y,z)\,,$$

AFFECTATION CONDITIONNELLE

$$R(C \longrightarrow A) \;\; =_{def} \;\; R(A) \cap (C \times \Sigma)\,.$$

$$(\sigma, \rho) \in R(C \longrightarrow A) \;\; \equiv \;\; [(\sigma, \rho) \in R(A) \wedge \sigma \models C]\,.$$

$\{P\}\; C \longrightarrow A\; \{Q\}\,,$
$\forall\sigma\,\forall\rho\,[(\sigma \models P \wedge (\sigma, \rho) \in R(C \longrightarrow A)) \;\Rightarrow\; \rho \models Q]\,,$
$\forall\sigma\,\forall\rho\,[(\sigma \models P \wedge (\sigma, \rho) \in R(A) \wedge \sigma \models C) \Rightarrow \rho \models Q]\,,$
$\forall\sigma\,\forall\rho\,[(\sigma \models (P \wedge C) \wedge (\sigma, \rho) \in R(A)) \;\Rightarrow\; \rho \models Q]\,,$
$\{P \wedge C\}\; A\; \{Q\}\,.$

$$\begin{array}{rcl} wlp[(C \longrightarrow A)\,;\; Q] & \equiv & (C \;\Rightarrow\; wlp[A;\; Q]) \\ slp[P\,;\; (C \longrightarrow A)] & \equiv & slp[(P \wedge C);\; A] \end{array}$$

EXEMPLE

$$\begin{array}{l} wlp[A[x] := 0;\; A(1) = 0] \;\; \equiv \\ (A; x : 0)\,(1) = 0 \;\; \equiv \\ (\lambda i : \text{if } i = x \text{ then } 0 \text{ else } A(i))\,(1) = 0 \;\; \equiv \\ (\text{if } 1 = x \text{ then } 0 \text{ else } A(1)) = 0 \;\; \equiv \\ (1 = x \wedge 0 = 0) \;\vee\; (1 \neq x \wedge A(1) = 0) \;\; \equiv \\ x = 1 \;\vee\; A(1) = 0\,. \end{array}$$

VARIABLES STRUCTUREES

$$A[e] := e'$$

$$A := \lambda i : \text{if } i = e \text{ then } e' \text{ else } A(i)$$

$$A := (A; e : e')\,.$$

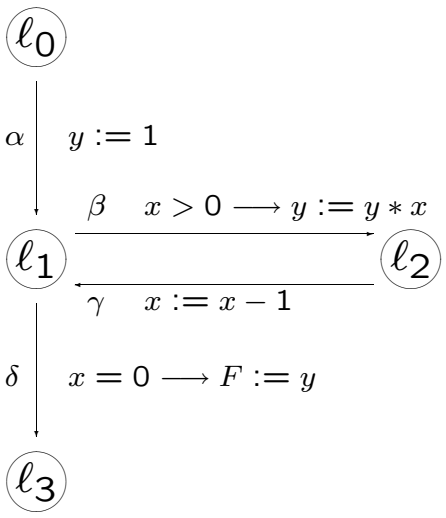
Si f est une fonction, on introduit la notation

$$f' = (f;\; e_1 : e'_1,\; e_2 : e'_2, \ldots, e_n : e'_n)$$

Règle

$$wlp\,[\,A[e] := e';\; P(A)] \;\; \equiv \;\; P((A; e : e'))$$

SYSTEMES FORMELS SEQUENTIELS



$P = \{\ell_0, \ell_1, \ell_2, \ell_3\}\,;$
$\mathcal{M} = \{x, y, F : entiers\}\,;$
$\mathcal{T} = \;\{\,\alpha :\;\; (\ell_0,\, y := 1,\, \ell_1)\,,$
$\beta :\;\; (\ell_1,\, x > 0 \longrightarrow y := y * x,\, \ell_2)\,,$
$\gamma :\;\; (\ell_2,\, x := x - 1,\, \ell_1)\,,$
$\delta :\;\; (\ell_1,\, x = 0 \longrightarrow F := y,\, \ell_3)\,\}\,.$

TRANSITION

$$\alpha \;=_{def}\; (\ell,\, C \longrightarrow A\,,\; \ell')\,.$$

$$((m,\sigma),(m',\rho)) \in R(\alpha)$$
$$\equiv$$
$$[m = \ell \wedge m' = \ell' \wedge (\sigma,\rho) \in R(C \longrightarrow A)]$$

POINT DE CONTROLE, NOTATION

$$(m,\sigma) \models at\ \ell \;\; =_{def} \;\; m = \ell$$

$$\sum_{\ell \in P} at\ \ell \;=\; 1\,.$$

$A[at\ \ell]$: remplacer dans A toutes les occurrences de $at\ \ell$ par *true* et toutes les occurrences d'autres prédicats de contrôle par *false*.

TRIPLET DE HOARE, WLP, SLP

$$\{P\}\ \alpha\ \{Q\}$$

est une abréviation de la formule

$$\forall s\, \forall r\, [(s \models P \wedge (s,r) \in R(\alpha)) \;\Rightarrow\; r \models Q]\,.$$

qui se réduit à

$$\{P[at\ \ell]\}\ C \longrightarrow X := E\ \{Q[at\ \ell']\}\,.$$

Si P et Q sont des assertions et
si $\alpha = (\ell,\, C \longrightarrow A,\, \ell')$ est une transition,
on a :

$$\begin{array}{rcl} wlp[\alpha\,;\, Q] & \equiv & (at\ \ell \;\Rightarrow\; wlp[(C \longrightarrow A);\; Q[at\ \ell']])\,, \\ slp[P\,;\, \alpha] & \equiv & (at\ \ell' \wedge slp[P[at\ \ell];\; (C \longrightarrow A)])\,. \end{array}$$

PROPRIETES, INVARIANT

Une *propriété d'invariance* est vraie de tous les états d'une exécution, tandis qu'une *propriété de progrès* est vraie d'au moins un état de l'exécution (dans le cas d'une exécution finie, ce sera en général le dernier état). Bien souvent, seules les exécutions dont l'état initial satisfait une certaine condition, dite *condition initiale*, sont considérées.

Un *INVARIANT* est une relation entre les valeurs des variables du programme qui est respectée par chaque transition (prise isolément).

EXEMPLE

$$\begin{array}{l} P = \{\ell_0, \ell_1, \ell_2, \ell_3\}\,; \\ \mathcal{M} = \{x, y, F : entiers\}\,; \\ \mathcal{T} = \{\, \alpha :\; (\ell_0,\, y := 1,\, \ell_1)\,, \\ \qquad\quad \beta :\; (\ell_1,\, x > 0 \longrightarrow y := y * x,\, \ell_2)\,, \\ \qquad\quad \gamma :\; (\ell_2,\, x := x - 1,\, \ell_1)\,, \\ \qquad\quad \delta :\; (\ell_1,\, x = 0 \longrightarrow F := y,\, \ell_3)\,\}\,. \end{array}$$

$$\begin{array}{ll} & [\,at\ \ell_0 \;\Rightarrow\; x = x_0\,] \\ \wedge & [\,at\ \ell_1 \;\Rightarrow\; (0 \le x \le x_0 \;\wedge\; y * x! = x_0!)\,] \\ \wedge & [\,at\ \ell_2 \;\Rightarrow\; (0 < x \le x_0 \;\wedge\; y * (x-1)! = x_0!)\,] \\ \wedge & [\,at\ \ell_3 \;\Rightarrow\; (x = 0 \;\wedge\; F = y = x_0!)\,]\,. \end{array}$$

SEQUENTIEL STRUCTURE

$$\frac{\{A\}\ S_1\ \{B\}\ ,\ \ \{B\}\ S_2\ \{C\}}{\{A\}\ S_1; S_2 \{C\}}$$

$$\frac{\{A \wedge B\}\ S_1\ \{C\}\ ,\ \ \{A \wedge \neg B\}\ S_2\ \{C\}}{\{A\}\ \text{if } B \text{ then } S_1 \text{ else } S_2\ \{C\}}$$

$$\frac{\{I \wedge B\}\ S\ \{I\}}{\{I\}\ \text{while } B \text{ do } S\ \{I \wedge \neg B\}}$$

$$\frac{A \Rightarrow B\ ,\ \ \{B\}\ S\ \{C\}\ ,\ \ C \Rightarrow D}{\{A\}\ S\ \{D\}}$$

PREUVE COMPLETE

$\{x = x_0 \wedge x_0 \geq 0 \wedge 1 = 1\}\ y := 1$
$\{x = x_0 \wedge x_0 \geq 0 \wedge y = 1\}$

$\{x = x_0 \wedge x_0 \geq 0\}\ y := 1\ \{x \geq 0 \wedge y * x! = x_0!\}$

$\{x > 0 \wedge y * x * (x-1)! = x_0!\}\ y := y * x$
$\{x > 0 \wedge y * (x-1)! = x_0!\}$

$\{x > 0 \wedge y * x! = x_0!\}\ y := y * x$
$\{x > 0 \wedge y * (x-1)! = x_0!\}$

$\{x - 1 \geq 0 \wedge y * (x-1)! = x_0!\}\ x := x - 1$
$\{x \geq 0 \wedge y * x! = x_0!\}$

$\{x > 0 \wedge y * (x-1)! = x_0!\}\ x := x - 1\ \{x \geq 0 \wedge y * x! = x_0!\}$

$\{x > 0 \wedge y * x! = x_0!\}\ [y := y * x;\ x := x - 1]$
$\{x \geq 0 \wedge y * x! = x_0!\}$

$\{x \geq 0 \wedge y * x! = x_0! \wedge x \neq 0\}\ [y := y * x; x := x - 1]$
$\{x \geq 0 \wedge y * x! = x_0!\}$

$\{x \geq 0 \wedge y * x! = x_0!\}$ [while $x \neq 0$ do $[y := y * x;\ x := x - 1]$]
$\{x \geq 0 \wedge y * x! = x_0! \wedge x = 0\}$

$\{x \geq 0 \wedge y * x! = x_0!\}$ [while $x \neq 0$ do $[y := y * x;\ x := x - 1]$]
$\{y = x_0!\}$

$\{x = x_0 \wedge x_0 \geq 0\}\ F\ \{y = x_0!\}$

PREUVE ABREGEE

$\{x = x_0 \wedge x_0 \geq 0\}$
$y := 1;$
$\{x \geq 0 \wedge y * x! = x_0!\}$
while $x \neq 0$ do
$\{x > 0 \wedge y * x! = x_0!\}$
$y := y * x;$
$\{x > 0 \wedge y * (x-1)! = x_0!\}$
$x := x - 1$
$\{x \geq 0 \wedge y * x! = x_0!\}$
$\{y = x_0!\}$

TRI (1)

Array $A[1:n]$ of integers; initial : A_0 , final : A_f

A_f version triée de A_0.

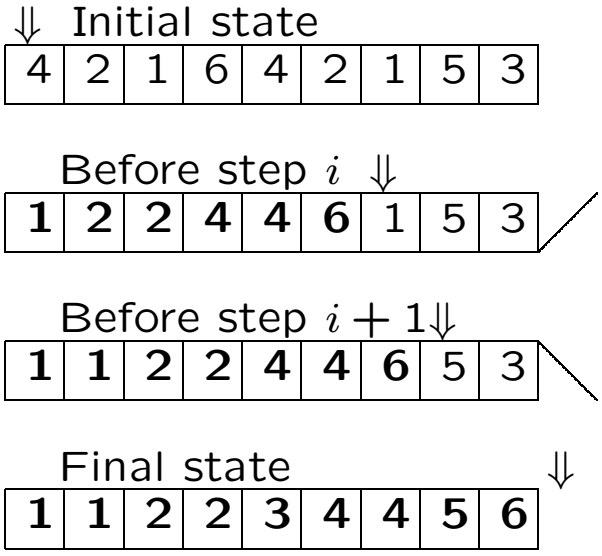
$\mathit{equal}(A, i, j, B, k, l) =_{def}$
$(j - i = l - k \wedge \forall r\, [0 \leq r \leq j - i \Rightarrow A(i + r) = B(k + r)])$.

$\mathit{card}(A, i, j, z) =_{def}$
$|\{r : i \leq r \leq j \wedge A(r) = z\}|$.
$\mathit{perm}(A, i, j, B, k, l) =_{def}$
$\forall u\, [u \in \mathbf{Z} \Rightarrow \mathit{card}(A, i, j, u) = \mathit{card}(B, k, l, u)]$.

$\mathit{ord}(A, i, j) =_{def}$
$\forall r, s\, [i \leq r \leq s \leq j \Rightarrow A(r) \preceq A(s)]$.

$\{\ \mathit{equal}(A, 1, n, A_0, 1, n)\ \}$
SORT
$\{\ \mathit{perm}(A, 1, n, A_0, 1, n) \wedge \mathit{ord}(A, 1, n)\ \}$

TRI (2)



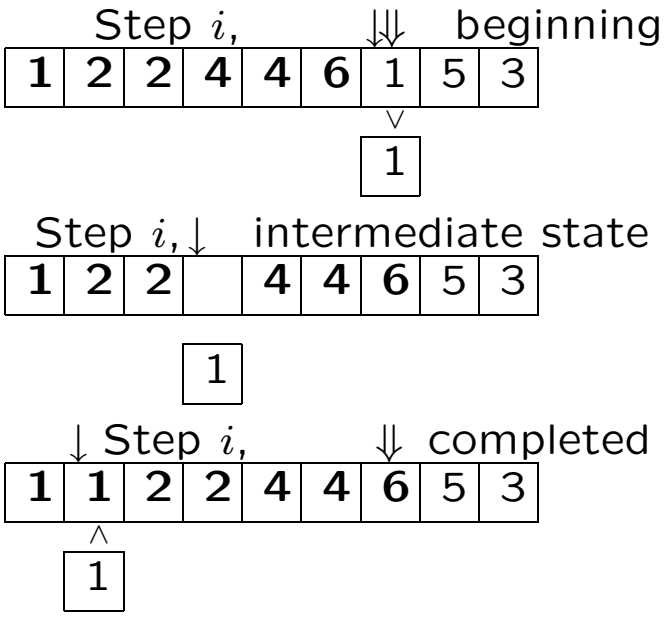
TRI (3) Invariant externe

$A[1:i-1]$ trié, $A[i:n]$ inchangé

$1 \le i \le n+1 \ \wedge$
perm$(A, 1, i-1, A_0, 1, i-1) \ \wedge$
ord$(A, 1, i-1) \ \wedge$
equal(A, i, n, A_0, i, n).

$i := 1$
{ Inv-Ext $\wedge$ $i = 1$ }
while $i \le n$ do
{ Inv-Ext $\wedge$ $i \le n$ }
"perform step i"
{ Inv-Ext $[i+1 \,/\, i]$ $\wedge$ $i \le n$ }
$i := i + 1$
{ Inv-Ext $\wedge$ $i \le n+1$ }
{ Inv-Ext $\wedge$ $i = n+1$ }

TRI (4)



TRI (5)

Invariant interne

$1 \le j \le i \le n \ \wedge \ x = A_0(i) \ \wedge$ *inf*$(x, A, j+1, i) \ \wedge$
perm2$(A, 1, j-1, A, j+1, i, A_0, 1, i-1) \ \wedge$
ord2$(A, 1, j-1, A, j+1, i) \ \wedge$
equal$(A, i+1, n, A_0, i+1, n)$.

inf(x, A, k, ℓ) : x less than $A(k), \ldots, A(\ell)$.

perm2$(A, i, j, B, k, \ell, C, m, n)$:

$A(i:j) \mid B(k:\ell)$ permutation de $C(m:n)$.

ord2(A, i, j, B, k, ℓ) : $A(i:j) \mid B(k:\ell)$ is sorted.

j divise $A(1:i)$ en deux :

$A(j+1:i)$ a été scanné, $A(1:j-1)$ non.

TRI (6)

$i := 1$;
{ Inv-Ext $\wedge$ $i = 1$ }
while $i \leq n$ do
{ Inv-Ext $\wedge$ $i \leq n$ }
$x := A[i]$; $j := i$;
{ Inv-Int $\wedge$ $j = i$ }
while $j > 1 \wedge A[j-1] > x$ do
$A[j] := A[j-1]$; $j := j - 1$;
{ Inv-Int }
$A[j] := x$; $i := i + 1$
{ Inv-Ext }
{ Inv-Ext $\wedge$ $i = n + 1$ }

Terminaison ("variants")

$n + 1 - i$ (ext); j (int)

PARALLELISME IMPLICITE

Idée :

Appliquer la méthode séquentielle

aux systèmes parallèles

Domaine : parallélisme synchrone

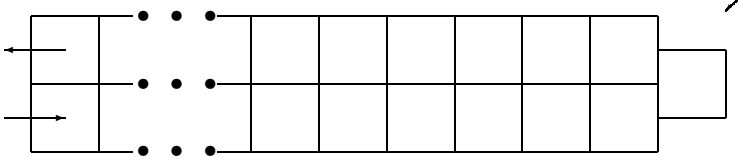
Exemple : Tri parallèle

Trier n données en un temps linéaire

au moyen de $O(n)$ machines parallèles

TRI SYSTOLIQUE

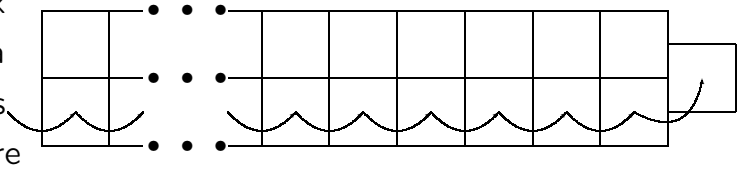
Machine cellulaire à $2n + 1$ éléments. Le flux des données est introduit dans la machine par la cellule inférieure gauche, et le flux des résultats (données triées) est extrait de la cellule supérieure gauche.



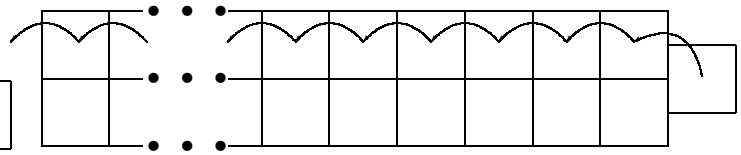
La machine peut exécuter trois actions :

1. entrée d'une donnée;
2. sortie d'un résultat;
3. tri binaire entre cellules appariées hautes et basses.

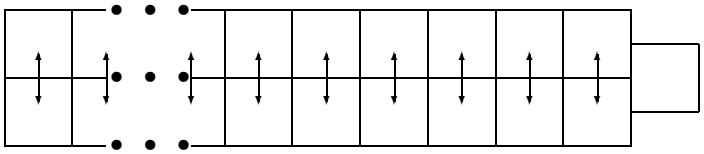
TROIS ACTIONS



Introduction d'un élément du flux d'entrée



Extraction d'un élément du flux de sortie



Composition parallèle de n tris binaires

PRINCIPE DE FONCTIONNEMENT

IN destruit une donnée,
OUT duplique une donnée,
BIN fait "remonter" les valeurs les plus "légères" dans les cellules du haut.

Seul fonctionnement possible :
répéter IN ; BIN ; OUT ; BIN

Tableau $V[-n:n]$
cellule supérieure gauche : $V[-n]$
cellule inférieure gauche : $V[n]$
cellule de l'extrémité droite : $V[0]$

SPECIFICATIONS (1)

Notations :
$*(r)S$: "répéter r fois S"
$sort(1..n)$: "$sort(1) \parallel \cdots \parallel sort(n)$"
$sort(i)$: if $V[i] < V[-i]$
then $(V[i], V[-i]) := (V[-i], V[i])$

Réduction séquentielle générique du système :

$*[\ V[0:n-1] := V[1:n]\ ;\ V[n] := A[j]\ ;$
$sort(1..n)\ ;$
$B[j] := V[-n]\ ;\ V[-n:-1] := V[1-n:0]\ ;$
$sort(1..n)\ ;$
$j := j+1\ ]\ .$

Propriété d'invariance requise :

$$V[-n] \ = \ min(V[-n:n])\ .$$

SPECIFICATIONS (2)

Chaque segment droit $V[-i:i]$ du système de tri entier $V[-n:n]$ est aussi un système de tri :

$I \ =_{def} \ \forall i(0 \leq i \leq n \ \Rightarrow \ V[-i] = min(V[-i:i]))$.

Si $in ::= [IN\,;BIN]$ et $out ::= [OUT\,;BIN]$
sont respectivement le programme
$V[0:n-1] := V[1:n];\ V[n] := A[j];\ sort(1..n)$
et le programme
$B[j] := V[-n]; V[-n:-1] := V[1-n:0]; sort(1..n)$
et si I est l'invariant,
alors les triplets $\{I\}\,in\,\{I\}$, $\{I\}\,out\,\{I\}$ sont vrais.

SCHEMA DE PREUVE

Premier triplet :

$$\{\forall i\,(V[-i] = min(V[-i:i])\ )\}$$
$$V[0:n-1] := V[1:n]\ ; V[n] := A[j];$$
$$\{\forall i > 0\,(V[-i] = min(V[-i:i-1]) \leq min(V[1-i:i-1]))\}$$
$$sort(1..n)$$
$$\{\forall i > 0\,(V[-i] = min(V[-i], V[i]) \leq min(V[1-i:i-1]))\}$$
$$\{\forall i\,(V[-i] = min(V[-i:i]))\}$$

Second triplet : analogue

Corollaire de la spécification :

$V[-n] \leq V[1-n] \leq \cdots \leq V[-1] \leq V[0]$

MODE D'EMPLOI

Valeurs initiales : $-\infty$

Première phase :

T est introduit, $Y(-\infty)$ est extrait.

Seconde phase :

$Z(+\infty)$ est introduit, $S(sorted\ T)$ est extrait.

Taille de T, Y, Z, S : $2n+1$

cycle(s)	$V[-n:n]$
0	$V[-n:n] = \langle -\infty \rangle$
$n-j$	$V[-n:j] = \langle -\infty \rangle \wedge V[j+1:n] = T[-n:-j-1]$
$n-j+\frac{1}{2}$	$V[-n:j-1] = \langle -\infty \rangle \wedge V[j:n] = T[-n:-j]$
$n+k$	$V[-n:-k-1] = \langle -\infty \rangle \wedge \pi(V_0[-k:n], T[-n:k-1])$
$n+k+\frac{1}{2}$	$V[-n:-k-1] = \langle -\infty \rangle \wedge \pi(V[-k:n], T[-n:k])$
$2n+\frac{1}{2}$	$\pi(V[-n:n], S[-n:n]) \wedge \pi(V[-n:n], T[-n:n])$
$3n+\frac{1}{2}-j$	$\pi(V[-n:j], S[-j:n]) \wedge V[j+1:n] = \langle +\infty \rangle$
$3n+1-j$	$\pi(V_0[-n:j], S[1-j:n]) \wedge V[j+1:n] = \langle +\infty \rangle$
$3n+\frac{1}{2}+k$	$V[-n:-k] = S[k:n] \wedge V[-k+1:n] = \langle +\infty \rangle$
$3n+1+k$	$V[-n:-k-1] = S[k+1:n] \wedge V[-k:n] = \langle +\infty \rangle$
$4n+1$	$V[-n:n] = \langle +\infty \rangle$

REALISATION SEQUENTIELLE

trier $2n+1$ valeurs en $4n+1$ étapes

affectations multiples,

concernant n ou $2n+1$ variables

$j := 1$;
$*(4n+1)$ [
  $i := 1$; $*(n)$ [$V[i-1] := V[i]$; $i := i+1$] ;
  if $j \leq 2n+1$ then $V[n] := T[j-n-1]$
    else $V[n] := +\infty$;
  $i := 1$; $*(n)$ [sort(i) ; $i := i+1$] ;
  if $j \geq 2n+1$ then $S[j-3n-1] := V[-n]$;
  $i := -n$; $*(n)$ [$V[i] := V[i+1]$; $i := i+1$] ;
  $i := 1$; $*(n)$ [sort(i) ; $i := i+1$] ;
  $j := j+1$].

COMMUNICATION SYNCHRONE (1)

$C!y || C?x$ remplace $x := y$

Cycle de base :

$V[0:n-1] := V[1:n]$; $V[n] := A[j]$;
$sort(1..n)$;
$B[j] := V[-n]$; $V[-n:-1] := V[1-n:0]$;
$sort(1..n)$;
$j := j+1$.

Décomposition : le processus E

$D_n!A[j]$;
$U_n?B[j]$;
$j := j+1$,

modélise l'*environnement* et en le processus Q_n

$V[0:n-1] := V[1:n]$;
$D_n?V[n]$;
$sort(1..n)$;
$U_n!V[-n]$;
$V[-n:-1] := V[1-n:0]$;
$sort(1..n)$,

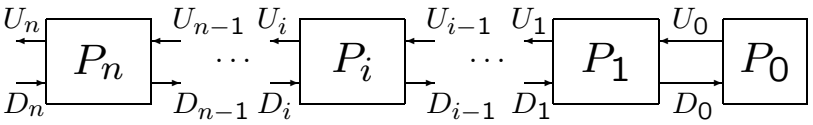
COMMUNICATION SYNCHRONE (2)

Le processus Q_n résulte lui-même
de la mise en parallèle du processus P_0,

$$D_0?V[0]\ ;\ U_0!V[0]\,,$$

et des processus P_i $(i = 1, \ldots, n)$

$$\begin{array}{l} D_{i-1}!V[i]\ ; \\ D_i?V[i]\ ; \\ sort(i)\ ; \\ U_i!V[-i]\ ; \\ U_{i-1}?V[-i]\ ; \\ sort(i)\,. \end{array}$$



VARIANTE

"Systolic" $\Rightarrow$ "wavefront"

Avec un tampon, le processus P_i devient

$$\begin{array}{l} w[i] := V[i]\ ; \\ D_{i-1}!w[i] \parallel D_i?V[i]\ ; \\ sort(i)\ ; \\ w[i] := V[-i]\ ; \\ U_i!w[i] \parallel U_{i-1}?V[-i]\ ; \\ sort(i)\,. \end{array}$$

BILAN ... TROMPEUR !

L'invariant est une expression formelle
de l'idée du programmeur.

Oui, mais ...

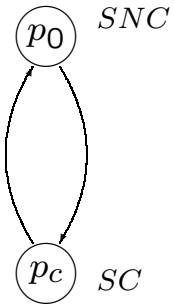
si le programme est structuré, déterministe et terminant, les tests suffisent !

donc on s'intéresse aux programmes peu structurés, non déterministes et non terminants.

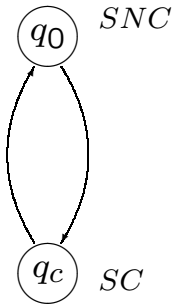
Un certain type de parallélisme suscite tout cela.

On verra aussi que l'automatisation pure est théoriquement possible, mais utopique dans la plupart des cas.

EXCLUSION MUTUELLE

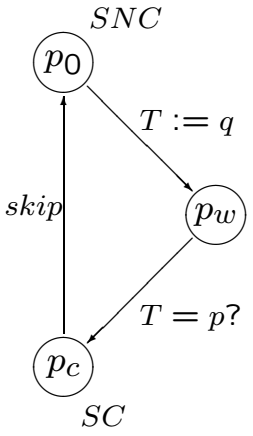


Processus P

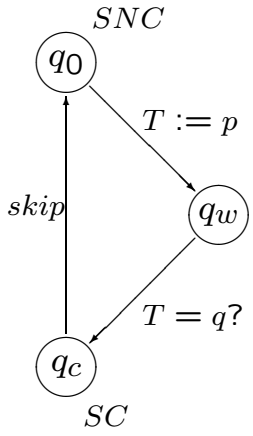


Processus Q

UN ALGORITHME NAIF



Processus P



Processus Q

INVARIANT

$$
\begin{array}{l}
at\ p_w\ \Rightarrow\ (T = q\ \vee\ at\ q_w)\,, \\
at\ q_w\ \Rightarrow\ (T = p\ \vee\ at\ p_w)\,, \\
at\ p_c\ \Rightarrow\ (T = p\ \wedge\ \neg at\ q_0)\,, \\
at\ q_c\ \Rightarrow\ (T = q\ \wedge\ \neg at\ p_0)\,.
\end{array}
$$

Corollaire

$$(at\ p_c\ \Rightarrow\ T = p)\ \wedge\ (at\ q_c\ \Rightarrow\ T = q)\,,$$

UNE SOLUTION MOINS RIGIDE

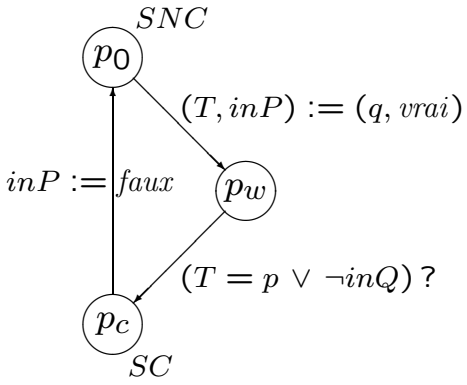
on remplace les tests $T = p\,?$ et $T = q\,?$

par $(T = p \vee at\ q_0)\,?$ et $(T = q \vee at\ p_0)\,?$

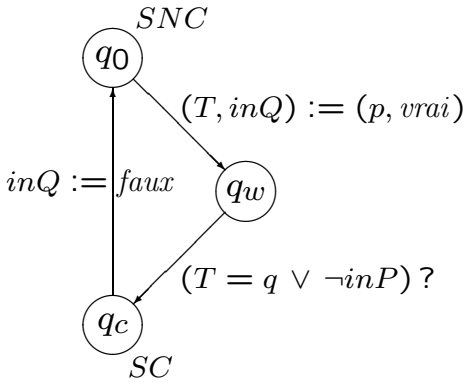
Invariant :

$$
\begin{array}{l}
at\ p_w\ \Rightarrow\ (T = q\ \vee\ at\ q_w)\,, \\
at\ q_w\ \Rightarrow\ (T = p\ \vee\ at\ p_w)\,, \\
at\ p_c\ \Rightarrow\ (T = p\ \vee\ at\ q_0) \\
at\ q_c\ \Rightarrow\ (T = q\ \vee\ at\ p_0)\,.
\end{array}
$$

INTRODUCTION DE VARIABLES



Processus P



Processus Q

$$\mathcal{P} ::= \{P, Q\}\,;\ \ P = \{p_0, p_w, p_c\}\,,\ \ Q = \{q_0, q_w, q_c\}\,;$$

$$\mathcal{M} ::= \{inP, inQ, T\}\,;$$

$$
\begin{array}{rl}
\mathcal{T} ::= \{ & (p_0,\ (T, inP) := (q, vrai)\,,\ p_w)\,, \\
& (p_w,\ T = p\ \vee\ \neg inQ \longrightarrow skip\,,\ p_c)\,, \\
& (p_c,\ inP := faux\,,\ p_0)\,, \\
& (q_0,\ (T, inQ) := (p, vrai)\,,\ q_w)\,, \\
& (q_w,\ T = q\ \vee\ \neg inP \longrightarrow skip\,,\ q_c)\,, \\
& (q_c,\ inQ := faux\,,\ q_0)\ \}\,.
\end{array}
$$

Invariant :

$(at\ p_0\ \equiv\ \neg inP)\ \wedge\ (at\ q_0\ \equiv\ \neg inQ)$
$(at\ p_w \Rightarrow (T = q \vee at\ q_w))\ \wedge\ (at\ q_w \Rightarrow (T = p \vee at\ p_w))$
$(at\ p_c\ \Rightarrow\ (T = p\ \vee\ \neg inQ))$
$(at\ q_c\ \Rightarrow\ (T = q\ \vee\ \neg inP))$

Conséquence logique :

$$at(p_c, q_c)\ \Rightarrow\ (T = p\ \wedge\ T = q)$$

PROBLEME DU GRAIN

$$(T, inP) := (q, vrai)$$

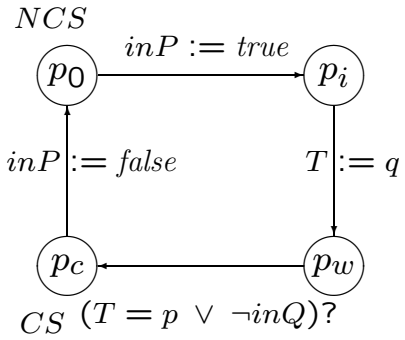
à décomposer

$$T := q\,;\ inP := vrai$$

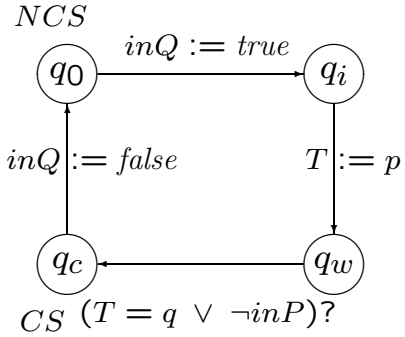
ou en

$$inP := vrai\,;\ T := q\,.$$

LE PREMIER CHOIX



Processus p



Processus q

Le premier choix est correct.

Invariant adapté :

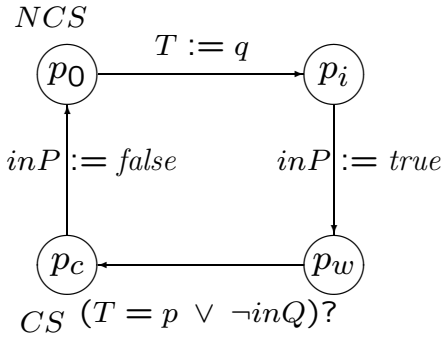
$$\begin{aligned} I = \ & (at\ p_c \ \Rightarrow\ (T = p \ \vee\ \neg inQ \ \vee\ at\ q_i)) \\ & \wedge\ (at\ q_c \ \Rightarrow\ (T = q \ \vee\ \neg inP \ \vee\ at\ p_i)) \\ & \wedge\ (at\ p_w \ \Rightarrow\ (T = q \ \vee\ at\ q_w)) \\ & \wedge\ (at\ q_w \ \Rightarrow\ (T = p \ \vee\ at\ p_w)) \\ & \wedge\ (at\ p_0 \ \equiv\ \neg inP) \\ & \wedge\ (at\ q_0 \ \equiv\ \neg inQ)\,. \end{aligned}$$

$$\begin{aligned} I[at\ p_c q_c] \ &= \ (T = p \ \vee\ \neg inQ) \\ & \quad \wedge\ (T = q \ \vee\ \neg inP)) \\ & \quad \wedge\ inP \\ & \quad \wedge\ inQ \end{aligned}$$

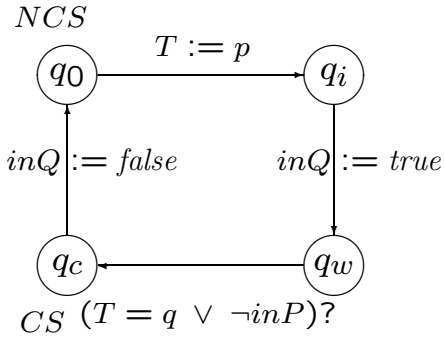
$$I[at\ p_c q_c] \ = \ T = p \wedge T = q$$

$$I[at\ p_c q_c] \ = \ false$$

LE SECOND CHOIX



Processus p



Processus q

Le second choix est incorrect.

Contre-exemple :

0. $at\ p_0 \wedge at\ q_0 \wedge \neg inP \wedge \neg inQ \wedge T = p$;
1. $at\ p_i \wedge at\ q_0 \wedge \neg inP \wedge \neg inQ \wedge T = q$;
2. $at\ p_i \wedge at\ q_i \wedge \neg inP \wedge \neg inQ \wedge T = p$;
3. $at\ p_i \wedge at\ q_w \wedge \neg inP \wedge inQ \wedge T = p$;
4. $at\ p_i \wedge at\ q_c \wedge \neg inP \wedge inQ \wedge T = p$;
5. $at\ p_w \wedge at\ q_c \wedge inP \wedge inQ \wedge T = p$;
6. $at\ p_c \wedge at\ q_c \wedge inP \wedge inQ \wedge T = p$.

AUTOMATISATION ?

L'invariant vérifie $I \Rightarrow wlp[\mathcal{S}; I]$

ou encore $slp[I; \mathcal{S}] \Rightarrow I$

De plus, on a $A \Rightarrow I$ et $I \Rightarrow B$

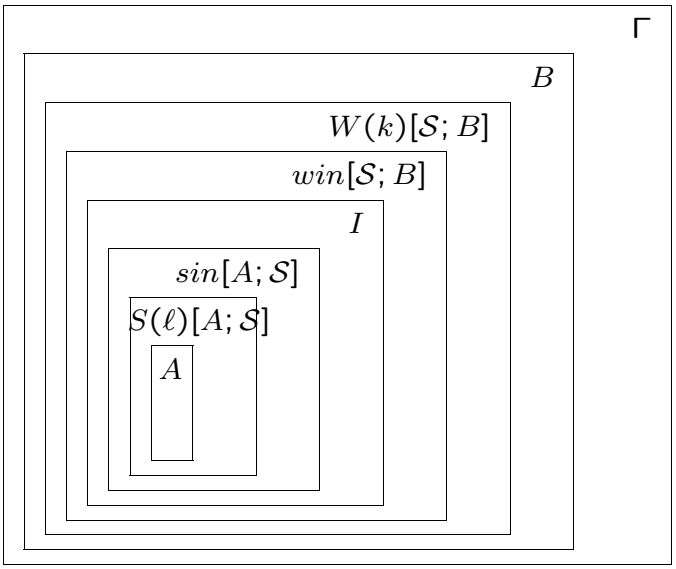
A : condition initiale

B : propriété de sûreté

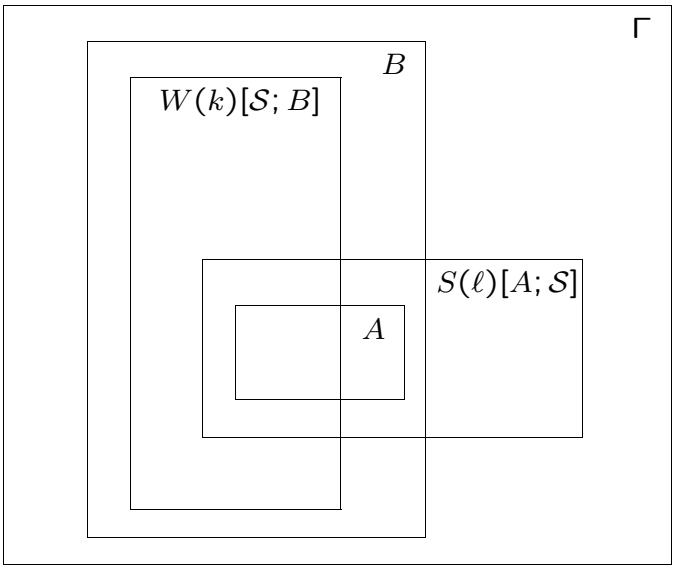
Résolution de point fixe

par approximations successives.

UN PROGRAMME CORRECT



UN PROGRAMME INCORRECT



EN PRATIQUE . . .

Si Γ est fini,
les suites sont stationnaires . . .
mais encombrantes !

Si Γ est infini,
les suites sont non stationnaires . . .
il faut "deviner" la limite !

Remède : Raffinements successifs.

PROTOCOLE DE STENNING I

$$\mathcal{P}_0 \;=\; \emptyset\,;$$
$$\mathcal{M}_0 \;=\; \{HS : nat;\; X, Y : array[nat]\ of\ string\}\,;$$
$$\mathcal{T}_0 \;=\; \{((HS, Y[HS{+}1]) := (HS{+}1, X[HS{+}1]))\}\,.$$

Invariant I_0

$$\forall s\,(1 \le s \le HS \Rightarrow Y[s] = X[s]) \;\wedge$$
$$\forall s\,(HS < s \Rightarrow Y[s] = NIL)\,.$$

Superposition : introduction de LR

La transition devient

$$(HS, LR, Y[HS{+}1]) := (HS{+}1, LR{+}1, X[HS{+}1])\,.$$

L'invariant devient $I_1 \;=\; (I_0 \wedge HS = LR)$

PROTOCOLE DE STENNING II

Raffinement : perte de message

Nouvelles transitions :

$$(LR = HS \;\longrightarrow\; (HS, LR, Y[HS+1]) := (HS+1, LR+1, X[HS+1]))$$
$$(LR = HS \;\longrightarrow\; HS := HS+1)$$
$$(LR < HS \;\longrightarrow\; (LR, Y[HS]) := (LR+1, X[HS]))$$
$$(LR < HS \;\longrightarrow\; skip)$$

Nouvel invariant I_2

$$(LR \le HS \le LR+1) \;\wedge$$
$$\forall s\,(1 \le s \le LR \Rightarrow Y[s] = X[s]) \;\wedge$$
$$(Y[HS] = X[HS] \vee Y[HS] = NIL) \;\wedge$$
$$\forall s\,(HS < s \Rightarrow Y[s] = NIL]) \;\wedge$$
$$(LR+1 = HS \;\Rightarrow\; Y[HS] = NIL)\,.$$

PROTOCOLE DE STENNING III

Raffinement : plus de parallélisme

Les transitions :

$$(LR{=}HS \longrightarrow (HS, Y[HS{+}1]) := (HS{+}1, X[HS{+}1]))$$
$$(LR{=}HS \longrightarrow HS := HS{+}1)$$
$$(LR{<}HS \longrightarrow Y[HS] := X[HS])$$
$$(LR{<}HS \longrightarrow skip)$$
$$(Y[LR{+}1] \neq NIL \;\longrightarrow\; LR := LR{+}1)\,.$$

L'invariant I_3 :

$$(LR \le HS \le LR+1) \wedge$$
$$\forall s\,(1 \le s \le LR \Rightarrow Y[s] = X[s]) \wedge$$
$$(Y[HS] = X[HS] \;\vee\; Y[HS] = NIL) \wedge$$
$$\forall s\,(HS < s \Rightarrow Y[s] = NIL])\,.$$

PROTOCOLE DE STENNING IV

Superposition : introduction de LA

Les transitions :

$$(LA{=}HS \longrightarrow (HS, Y[HS{+}1]) := (HS{+}1, X[HS{+}1]))$$
$$(LA{=}HS \longrightarrow HS := HS{+}1)$$
$$(LA{<}HS \longrightarrow Y[HS] := X[HS])$$
$$(LA{<}HS \longrightarrow skip)$$
$$(Y[LR{+}1] \neq NIL \longrightarrow (LA, LR) := (LR{+}1, LR{+}1))$$

L'invariant $I_4 =_{def} (I_3 \wedge LA = LR)$

PROTOCOLE DE STENNING V

Les transitions :

$$(LA{=}HS \longrightarrow (HS, Y[HS{+}1]) := (HS{+}1, X[HS{+}1]))$$
$$(LA{=}HS \longrightarrow HS := HS{+}1)$$
$$(LA{<}HS \longrightarrow Y[HS] := X[HS])$$
$$(LA{<}HS \longrightarrow skip)$$
$$(Y[LR{+}1] \neq NIL \longrightarrow (LA, LR) := (LR{+}1, LR{+}1))$$
$$(Y[LR{+}1] \neq NIL \longrightarrow LR := LR{+}1)$$
$$(Y[LR{+}1] = NIL \longrightarrow LA := LR)$$
$$(Y[LR{+}1] = NIL \longrightarrow skip)$$

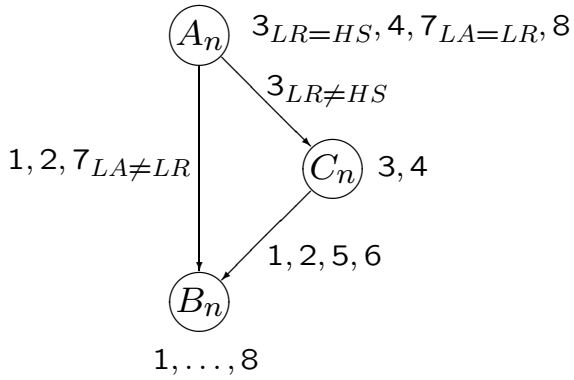
L'invariant I_5 :

$$(LA \leq LR \leq HS \leq LA + 1) \ \wedge$$
$$\forall s\,(1 \leq s \leq LR \Rightarrow Y[s] = X[s]) \ \wedge$$
$$(Y[HS] = X[HS] \ \vee \ Y[HS] = NIL) \ \wedge$$
$$\forall s\,(HS < s \Rightarrow Y[s] = NIL])$$

PROTOCOLE DE STENNING V (bis)

Vivacité : preuve graphique

$A_n : LA{+}LR{+}HS{=}n \ \wedge \ Y[LR{+}1]{=}NIL$,
$B_n : LA{+}LR{+}HS{>}n$,
$C_n : LA{+}LR{+}HS{=}n \ \wedge \ Y[LR{+}1]{=}X[LR{+}1]$.



PROTOCOLE DE STENNING VI

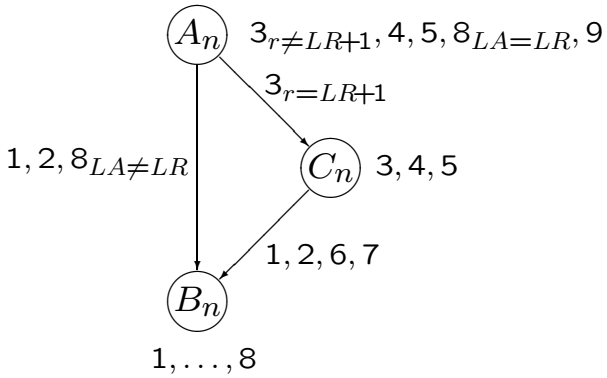
1. $(HS{-}LA < W \longrightarrow (HS, Y[HS{+}1]) := (HS{+}1, X[HS{+}1]))$,
2. $(HS{-}LA < W \longrightarrow HS := HS{+}1)$,
3. $(LA < r \leq HS \ \longrightarrow \ (Y[r], r) := (X[r], r{+}1))$,
4. $(LA < r \leq HS \ \longrightarrow \ r := r{+}1)$,
5. $(\neg(LA < r \leq HS) \ \longrightarrow \ r := LA{+}1)$,
6. $(Y[LR{+}1] \neq NIL \ \longrightarrow \ (LA, LR) := (LR{+}1, LR{+}1))$,
7. $(Y[LR{+}1] \neq NIL \ \longrightarrow \ LR := LR{+}1)$,
8. $(Y[LR{+}1] = NIL \ \longrightarrow \ LA := LR)$.

$$I_6 \equiv \ (LA \leq LR \leq HS \leq LA{+}W) \ \wedge$$
$$\forall s\,(Y[s] \in \{X[s], NIL\}) \ \wedge$$
$$\forall s\,(1 \leq s \leq LR \Rightarrow Y[s] = X[s]) \ \wedge$$
$$\forall s\,(HS < s \Rightarrow Y[s] = NIL)\,.$$

PROTOCOLE DE STENNING VI (bis)

$HS - W < LA$:
transitions 1, 2 exécutables $[\{A_n, C_n\} \longrightarrow B_n]$.
$Y[LR{+}1] \neq NIL$:
transitions 6, 7 exécutables $[C_n \longrightarrow B_n]$.
$LA < LR \wedge Y[LR{+}1] = NIL$:
transition 8 exécutable $[A_n \longrightarrow B_n]$.
$LR < HS \wedge Y[LR{+}1] = NIL$: transition 3
(avec $r = LR{+}1$) exécutable infiniment souvent
$[A_n \longrightarrow C_n]$.

PROTOCOLE DE STENNING VI (ter)



PROTOCOLE DE STENNING VII

$A{=}\{A_0\}$, $B{=}\{B_0\}$, $C{=}\{C_0\}$, $D{=}\{D_0\}$, $E{=}\{E_0\}$.

1 : $(A_0D_0,\ HS{-}LA < W \longrightarrow (HS, Y[HS{+}1]) := (HS{+}1, X[HS{+}1]),\ A_0D_0)$,
2 : $(A_0,\ HS - LA < W \longrightarrow HS := HS{+}1\ ,\ A_0)$,
3 : $(B_0D_0,\ LA < r \leq HS \longrightarrow (Y[r], r) := (X[r], r{+}1)\ ,\ B_0D_0)$,
4 : $(B_0,\ LA < r \leq HS \longrightarrow r := r{+}1\ ,\ B_0)$,
5 : $(B_0,\ \neg(LA < r \leq HS) \longrightarrow r := LA{+}1\ ,\ B_0)$,
6 : $(C_0E_0,\ Y[LR{+}1] \neq NIL \longrightarrow (LA, LR) := (LR{+}1, LR{+}1)\ ,\ C_0E_0)$,
7 : $(E_0,\ Y[LR{+}1] \neq NIL \longrightarrow LR := LR{+}1\ ,\ E_0)$,
8 : $(C_0E_0,\ Y[LR{+}1]{=}NIL \longrightarrow LA := LR,\ C_0E_0)$,
9 : $(E_0,\ Y[LR{+}1] = NIL \longrightarrow skip,\ E_0)$.

PROTOCOLE DE STENNING VII (bis)

Envoyer nouveaux messages (rôle A),

Renvoyer messages de fenêtre (rôle B),

Recevoir acquits (rôle C),

Recevoir messages (rôle D),

Envoyer acquits (rôle E).

PROTOCOLE DE STENNING VIII

1 : $(A_0,\ HS{-}LA < W \longrightarrow (HS, MB) := (HS{+}1, (HS{+}1, X[HS{+}1]))\ ,\ A_0)$,
2 : $(B_0,\ LA < r \leq HS \longrightarrow (MB, r) := ((r, X[r]), r{+}1)\ ,\ B_0)$,
3 : $(B_0,\ \neg(LA < r \leq HS) \longrightarrow r := LA{+}1\ ,\ B_0)$,
4 : $(D_0,\ MB \neq NIL \longrightarrow (Y[MB.1], MB) := (MB.2, NIL)\ ,\ D_0)$,
5 : $(C_0E_0,\ Y[LR{+}1] \neq NIL \longrightarrow (LA, LR) := (LR{+}1, LR{+}1)\ ,\ C_0E_0)$,
6 : $(E_0,\ Y[LR{+}1] \neq NIL \longrightarrow LR := LR{+}1\ ,\ E_0)$,
7 : $(C_0E_0,\ Y[LR{+}1] = NIL \longrightarrow LA := LR\ ,\ C_0E_0)$,
8 : $(E_0,\ Y[LR{+}1] = NIL \longrightarrow \mathit{skip}\ ,\ E_0)$,
9 : $(\emptyset,\ MB := NIL\ ,\ \emptyset)$.

PROTOCOLE DE STENNING VIII (bis)

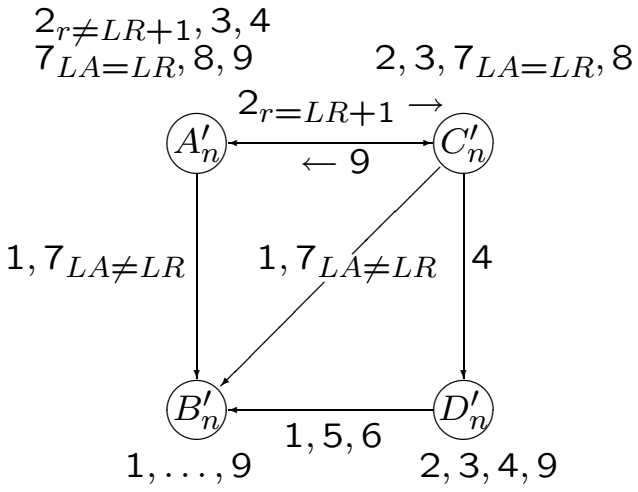
Invariant : ajouter l'assertion

$$MB{=}NIL \ \vee\ \exists s\,[(1{\leq}s{\leq}HS) \wedge MB{=}(s, X[s])]\,.$$

Propriétés de vivacité

A'_n : $LA + LR + HS = n \ \wedge\ MB.1 \neq LR + 1 \ \wedge\ Y[LR + 1] = NIL$,
B'_n : $LA + LR + HS > n$,
C'_n : $LA + LR + HS = n \ \wedge\ MB.1 = LR + 1 \ \wedge\ Y[LR + 1] = NIL$,
D'_n : $LA + LR + HS = n \wedge Y[LR + 1] = X[LR + 1]$.

PROTOCOLE DE STENNING VIII (ter)



PROTOCOLE DE STENNING IX

1 : $(A_0,\ HS{-}LA < W \longrightarrow (HS, MB) := (HS{+}1, (HS{+}1, X[HS{+}1]))\ ,\ A_0)$,
2 : $(B_0,\ LA < r \leq HS \longrightarrow (MB, r) := ((r, X[r]), r + 1)\ ,\ B_0)$,
3 : $(B_0,\ \neg(LA < r \leq HS) \longrightarrow r := LA + 1\ ,\ B_0)$,
4 : $(C_0,\ AB \neq NIL \longrightarrow (LA, AB) := (AB, NIL)\ ,\ C_0)$,
5 : $(D_0,\ MB \neq NIL \longrightarrow (Y[MB.1], MB) := (MB.2, NIL)\ ,\ D_0)$,
6 : $(E_0,\ Y[LR + 1] \neq NIL \longrightarrow (AB, LR) := (LR + 1, LR + 1)\ ,\ E_0)$,
7 : $(E_0,\ Y[LR + 1] = NIL \longrightarrow AB := LR\ ,\ E_0)$,
8 : $(\emptyset,\ MB := NIL\ ,\ \emptyset)$,
9 : $(\emptyset,\ AB := NIL\ ,\ \emptyset)$.

PROTOCOLE DE STENNING IX (bis)

$$
\begin{array}{l}
I_9 \;\; =_{def} \\
(LA \leq LR \leq HS \leq LA + W) \wedge \\
\forall s\,(Y[s] \in \{X[s], NIL\}) \wedge \\
\forall s\,(1 \leq s \leq LR \Rightarrow Y[s] = X[s]) \wedge \\
\forall s\,(HS < s \Rightarrow Y[s] = NIL]) \wedge \\
(MB = NIL \vee \\
\;\exists s\,[(1 \leq s \leq HS) \wedge MB = (s, X[s])]) \wedge \\
(AB = NIL \;\vee\; AB = LR)\,.
\end{array}
$$

PROTOCOLE DE STENNING X

Version finale (processus)

$$
\begin{array}{rl}
\mathcal{P}_f \;=\; \{A, B, C, D, E\} & \\
\text{où} \quad A = \{A_0, A_1, A_2\}\,, & \\
\quad B = \{B_0, B_1, B_2\}\,, & \\
\quad C = \{C_0, C_1, C_2\}\,, & \\
\quad D = \{D_0, D_1, D_2\}\,, & \\
\quad E = \{E_0, E_1, E_2\}\,. &
\end{array}
$$

PROTOCOLE DE STENNING X (bis)

Version finale (émetteur)

$$
\begin{array}{ll}
1: & (A_0,\; HS - LA < W \;\longrightarrow\; skip\,,\; A_1)\,, \\
2: & (A_1,\; HS := HS + 1\,,\; A_2)\,, \\
3: & (A_2,\; MB := (HS, X[HS])\,,\; A_0)\,, \\
4: & (B_0,\; LA < r \leq HS \;\longrightarrow\; MB := (r, X[r])\,,\; B_2)\,, \\
5: & (B_2,\; r := r + 1\,,\; B_0)\,, \\
6: & (B_0,\; \neg(LA < r \leq HS) \;\longrightarrow\; skip\,,\; B_1)\,, \\
7: & (B_1,\; r := LA + 1\,,\; B_0)\,, \\
8: & (C_0,\; AB \neq NIL \;\longrightarrow \\
 & \qquad (ackno, AB) := (AB, NIL)\,,\; C_1)\,, \\
9: & (C_1,\; ackno \leq LA \;\longrightarrow\; skip\,,\; C_0)\,, \\
10: & (C_1,\; ackno > LA \;\longrightarrow\; skip\,,\; C_2)\,, \\
11: & (C_2,\; LA := ackno\,,\; C_0)\,.
\end{array}
$$

PROTOCOLE DE STENNING X (ter)

Version finale (récepteur)

$$
\begin{array}{ll}
12: & (D_0,\; MB \neq NIL \;\longrightarrow \\
 & \qquad (m, MB) := (MB, NIL)\,,\; D_1)\,, \\
13: & (D_1,\; Y[m.1] := m.2\,,\; D_2)\,, \\
14: & (D_2,\; AR[m.1] := 1\,,\; D_0)\,, \\
15: & (E_0,\; AR[LR + 1] = 1 \;\longrightarrow\; skip\,,\; E_1)\,, \\
16: & (E_0,\; AR[LR + 1] = 0 \;\longrightarrow\; skip\,,\; E_2)\,, \\
17: & (E_1,\; LR := LR + 1\,,\; E_2)\,, \\
18: & (E_2,\; AB := LR\,,\; E_0)\,.
\end{array}
$$

Version finale (milieu de transmission)

$$
\begin{array}{ll}
19: & (\emptyset,\; MB := NIL\,,\; \emptyset)\,, \\
20: & (\emptyset,\; AB := NIL\,,\; \emptyset)\,\}\,.
\end{array}
$$

PROTOCOLE DE STENNING X (quater)

Version finale (invariant)

$(LA \leq LR \leq HS \leq LA + W) \wedge$
$(at\ A_1\ \Rightarrow\ HS < LA + W) \wedge$
$\forall s\,(Y[s] \in \{X[s], NIL\}) \wedge$
$\forall s\,(1 \leq s \leq LR \Rightarrow Y[s] = X[s]) \wedge$
$\forall s\,(HS < s \Rightarrow (Y[s] = NIL] \wedge AR[s] = 0)) \wedge$
$(MB{=}NIL \vee \exists s\,[(1{\leq}s{\leq}HS) \wedge MB{=}(s, X[s])]) \wedge$
$(\neg at\ E_2 \Rightarrow AB \in \{NIL, LR\}) \wedge$
$(at\ E_2 \Rightarrow AB \in \{NIL, LR-1, LR\}) \wedge$
$\forall s\,([AR[s]{=}1 \vee (at\ D_2 \wedge s{=}m.1)] \equiv Y[s]{=}X[s]) \wedge$
$(at\ C_1 \Rightarrow ackno \leq LR) \wedge$
$(at\ C_2 \Rightarrow LA < ackno \leq LR) \wedge$
$(at\ D_1 \Rightarrow (m.2 = X[m.1] \wedge m.1 \leq HS)) \wedge$
$(at\ D_2 \Rightarrow (Y[m.1] = X[m.1] \wedge m.1 \leq HS))\,.$

Beaucoup de versions . . . et d'invariants!
Un invariant contient beaucoup d'assertions.
Une assertion contient beaucoup d'atomes.
Il faut automatiser !

Invariant $I = \bigwedge_{i=1}^{n} a_i$ et transition τ.
$a'_k\ =_{def}\ wlp[\tau; a_k]$,
wlp est $\wedge$-additive:
$wlp[\tau; (a_k \wedge a_\ell)]\quad\equiv\quad wlp[\tau; a_k] \wedge wlp[\tau; a_\ell]\,.$

$$V\ =_{def}\ \bigwedge_{k\in K}\ \bigwedge_{\tau\in\mathcal{S}} (I \Rightarrow wlp[\tau; a_k])\,.$$

Condition de vérification : $(\bigwedge_{i=1}^{n} a_i) \Rightarrow (c \Rightarrow a'_k)\,.$

schéma VC :

$$(\bigwedge_{i=1}^{n} h_i) \Rightarrow c\,.$$

Suppositions (souvent raisonnables) :

1. $H = \{h_1, \ldots, h_n\}$ *plutôt grand;*
2. h_i *et* c *: petits, pas de quantificateur;*
3. *Beaucoup de booléens, peu de prédicats;*
4. $H \models c$ *se réduit à* $H_0 \models c$ *(*H_0 *petit).*

Comment obtenir H_0 ?
Eliminer ce qui est *prouvé* inutile est NP-hard.
Eliminer ce qui est "probablement" inutile?

Idée générale

Eliminer ce qui est *prouvé* inutile (partiel).

Classifier le reste.

Considérer en séquence

$$\begin{array}{l}(c \Rightarrow a'_k)\,,\\ a_{j_1} \Rightarrow (c \Rightarrow a'_k)\,,\\ a_{j_2} \Rightarrow (a_{j_1} \Rightarrow (c \Rightarrow a'_k))\,,\\ \ldots\end{array}$$

jusqu'à réussite . . . ou épuisement.

Notion de type

Type : mot de $(c \cup d \cup n)^*$.

- Le type de B dans B est λ (mot vide).
- Si le type de B dans A est w, alors le type
  - dans $\neg A$ est nw,
  - dans $(A \wedge X)$ ou $(X \wedge A)$ est cw,
  - dans $(A \vee X)$, $(X \vee A)$ ou $(X \Rightarrow A)$ est dw,
  - dans $(A \Rightarrow X)$ est dnw.

Type réduit:
c devient ndn, nn devient λ.

Première solution : analyse de la formule

Exemple: $(p \Rightarrow q) \vee \neg(r \wedge (\neg s \Rightarrow \neg(t \Rightarrow u)))$

	p	q	r	s	t	u
type	ddn	dd	dnc	$dncdnn$	$dncdndn$	$dncdnd$
t. réd.	ddn	dd	ddn	$ddnd$	$ddndndn$	$ddndnd$
pure dis	oui	oui	oui	non	non	non
rang	2	2	2	2	3	3
polarité	$-$	$+$	$-$	$-$	$-$	$+$

La *polarité* d'une occurrence B_A de B dans A est *positive* si le type (ou le type réduit) de B_A contient un nombre pair de "n";
elle est *négative* sinon.

Règle de polarité. (PR)

Si la proposition p n'a pas d'occurrence négative dans $\mathcal{H}$ ni d'occurrence positive dans C, alors C est conséquence logique de $\mathcal{H}$ si et seulement si $C[p/true]$ est conséquence logique de $\mathcal{H}[p/true]$.

Si la proposition p n'a pas d'occurrence positive dans $\mathcal{H}$ ni d'occurrence négative dans C, alors C est conséquence logique de $\mathcal{H}$ si et seulement si $C[p/false]$ est conséquence logique de $\mathcal{H}[p/false]$.

Le *rang* d'une occurrence B_A de B dans A est le nombre de "d" précédés (immédiatement ou non) d'un nombre pair de "n" dans le type réduit de B_A. L'occurrence B_A de B dans A est *purement disjonctive* si son type réduit w ne comporte que des lettres d, suivie ou non de la lettre n; elle est *purement conjonctive* if si son type réduit est vide, ou n suivi d'un type réduit purement disjonctif.

Idée : A faux ssi B est bien choisi.

Lemme du rang.
Si B_A est purement disjonctive dans A et si $v(A) = false$, alors $v(B) = false(true)$ si B_A est positive (négative).

Preuve par induction sur le rang de B_A dans A.

Règle du rang. (RR)

Si p a une occurrence positive, purement conjonctive dans une hypothèse $h \in \mathcal{H}$ ou une occurrence négative, purement disjonctive dans C, alors C est conséquence logique de $\mathcal{H}$ si et seulement si $C[p/true]$ est conséquence logique de $\mathcal{H}[p/true]$.

Si p a une occurrence négative, purement conjonctive dans une hypothèse $h \in \mathcal{H}$ ou une occurrence positive, purement disjonctive dans C, alors C est conséquence logique de $\mathcal{H}$ si et seulement si $C[p/false]$ est conséquence logique de $\mathcal{H}[p/false]$.

Soit Π un ensemble de variables propositionnelles et H et C des formules sur le lexique Π. La formule H est *pertinente pour C par rapport à Π* si une interprétation v sur Π existe telle que $v(H) = v(C) = \mathit{false}$.

Théorème de la pertinence. Si h_1 n'est pas pertinent pour C par rapport à Π, alors les formules

$$\begin{array}{lll} \phi_1 & =_{def} & (h_1 \wedge h_2 \wedge \cdots \wedge h_n) \Rightarrow C\,, \\ \phi_2 & =_{def} & (h_2 \wedge \cdots \wedge h_n) \Rightarrow C\,, \end{array}$$

sont logiquement équivalentes, quelles que soient les formules $h_2, \ldots, h_n$.

Preuve. Par l'absurde: ϕ_1 vrai et ϕ_2 faux n'est possible que si h_1 est faux, $h_2, \ldots, h_n$ sont vrais et C est faux. De plus ϕ_1 faux et ϕ_2 vrai est impossible.

Réciproquement, si h_1 est pertinent pour C par rapport à Π, alors des formules $h_2, \ldots, h_n$ existent telles que ϕ_1 soit valide et ϕ_2 soit non valide.

Preuve. Soit $n = 2$ avec $h_2 : (h_1 \Rightarrow C)$.

Règle d'élimination. (ER)
Soit A une sous-formule commune à H et C, et A_H et A_C les occurrences correspondantes. Si A_H et A_C sont purement disjonctives et de polarités opposées, alors H est non pertinent pour C.

Preuve. Soient A_H positive et A_C négative.
Si v existe telle que $v(H) = v(C) = \mathit{false}$,
alors (lemme du rang), A_H) $v(A) = \mathit{false}$,
et (lemme du rang, A_C) $v(A) = \mathit{true}$.

Exemple. Soient $H =_{def} \alpha \vee (\beta \Rightarrow \gamma)$ et $C =_{def} \alpha \vee (\gamma \Rightarrow \delta)$. La sous-formule γ est purement disjonctive dans H et dans C, positive dans H et négative dans C. La formule H n'est donc pas pertinente pour C.

Remarque. La règle d'élimination est sûre et efficace, mais visiblement incomplète.

Pertinence directe

La formule H est *directement pertinente* pour C si elle est pertinente et si H et C admettent une sous-formule commune, de même polarité.

Examples et contre-exemples.
$p \Rightarrow q$ est directement pertinente pour $p \Rightarrow r$ et $r \Rightarrow q$
mais pas pour p, ni pour $q \Rightarrow r$.

Interpolation polarisée.

Si $\models (H \Rightarrow C)$, alors il existe un interpolant polarisé J:
$\models (H \Rightarrow J)$, $\models (J \Rightarrow C)$ et toute proposition p intervenant dans J intervient aussi dans H et dans C, avec la même polarité dans les trois formules.

Construction récursive.
On obtient $J(p)$ pour $(H(p), C(p))$ à partir de $J(\mathit{true})$ et $J(\mathit{false})$, resp. pour $(H(\mathit{true}), C(\mathit{true}))$ et $(H(\mathit{false}), C(\mathit{false}))$.

Exemple.
$H =_{def} r \wedge p \wedge a$ and $C =_{def} r \vee \neg p \vee b$.
$J_1 =_{def} r \wedge p$ and $J_2 =_{def} r \vee \neg p$ interpolants de H et C,
r interpolant polarisé de H et C

Théorème de la pertinence directe.
$\mathcal{H}$: ensemble consistant de formules (hypothèses)
C : formule non valide (conclusion).
Si $\mathcal{H} \models \mathcal{C}$, alors $\mathcal{H}$ contient une formule h
telle que h est directement pertinente C.

Preuve. On considère un interpolant polarisé J pour $\bigwedge \mathcal{H}$ and C, et n'importe quelle proposition p ayant une occurrence dans J.

Soient $\mathcal{H}$ un ensemble consistant d'hypothèses et C une conclusion. Une *chaîne polarisée* est une suite non vide $(\phi_0, \ldots, \phi_n)$ telle que

- ϕ_0 est $\neg C$;
- $\phi_i \in \mathcal{H}$, pour tous $i = 1, \ldots, n$;
- ϕ_i et ϕ_{i-1} comportent au moins une proposition en commun, pour tous $i = 1, \ldots, n$.
- ϕ_i est pertinente pour $\neg\phi_{i-1}$, pour tous $i = 1, \ldots, n$.

L'ensemble $\mathcal{H}$ et la formule C étant donnés, on note $\mathcal{H}_C$ l'ensemble des hypothèses qui apparaissent dans au moins une chaîne polarisée.

Règle de la connectivité polarisée. (PCR)

La conclusion C est une conséquence logique de l'ensemble $\mathcal{H}$ des hypothèses si et seulement si cette conclusion est conséquence logique du sous-ensemble $\mathcal{H}_C \subset \mathcal{H}$.

H_1 : c_1
H_2 : $a_2 \vee (c_1 \Rightarrow e_1)$
H_3 : $b_0 \Rightarrow \neg(a_2 \vee e_0)$
H_4 : $e_1 \Rightarrow \neg(a_2 \wedge c_2)$
H_5 : $e_1 \Rightarrow (a_0 \Rightarrow d_0)$
H_6 : $(a_0 \wedge d_0) \Rightarrow ((c_0 \wedge d_2) \Rightarrow f_0)$
H_7 : $(f_1 \wedge f_0) \Rightarrow ((a_0 \wedge a_2) \vee (c_0 \wedge e_0))$
H_8 : $f_2 \Rightarrow (c_0 \vee e_2)$
H_9 : $((a_0 \vee c_0) \wedge f_0) \Rightarrow (f_1 \vee (e_0 \wedge f_2))$
H_{10} : $(\neg c_2 \vee f_0) \Rightarrow (e_2 \vee (b_2 \wedge \neg f_1))$
H_{11} : $((d_0 \vee e_0) \Rightarrow ((b_0 \Rightarrow f_0) \vee (f_0 \Rightarrow c_2)))$
H_{12} : $\neg c_2 \Rightarrow (e_2 \vee (b_2 \wedge \neg f_1))$
H_{13} : $(a_2 \wedge f_0) \Rightarrow (b_0 \vee (e_0 \wedge e_1))$
H_{14} : $((a_1 \wedge c_1) \vee (b_1 \wedge a_2)) \Rightarrow a_0$
H_{15} : $((d_0 \vee a_2) \wedge \neg b_0) \Rightarrow (c_2 \vee (c_0 \Rightarrow f_0))$
H_{16} : $((a_1 \vee c_1) \wedge d_0) \Rightarrow (b_0 \vee (e_0 \wedge a_1))$
H_{17} : $b_1 \Rightarrow a_1$
H_{18} : $(d_2 \vee (a_1 \Rightarrow f_1)) \Rightarrow ((b_1 \wedge c_1) \vee f_2)$
H_{19} : $(a_2 \vee c_1) \Rightarrow (b_1 \Rightarrow b_2)$
H_{20} : $(d_1 \vee f_1) \Rightarrow (a_1 \wedge c_2)$
H_{21} : $(c_2 \wedge b_2) \Rightarrow (d_1 \vee d_2)$
C : $(a_0 \wedge b_0) \Rightarrow$
$((c_0 \vee c_1) \Rightarrow ((d_0 \vee e_0) \vee (c_2 \Rightarrow e_0)))$

Elimination de H_{11}
(ER, sous-formule $(d_0 \vee e_0)$).

Elimination de H_{15}
(ER, proposition b_0).

Elimination de H_6 et H_{16}
(ER, proposition d_0).

Proposition f_0 : toutes occurrences négatives dans les hypothèses restantes,
pas d'occurrence dans la conclusion;
f_0 remplacé par $false$ (PR).

Elimination de H_7, H_9, H_{13} (BR).

H_{10} est simplifié et s'identifie à H_{12}.

Elimination de
H_8, H_{10}, H_{12}, H_{17}, H_{18}, H_{19}, H_{20} et H_{21},
(PCR)

H_1 : c_1
H_2 : $a_2 \vee (c_1 \Rightarrow e_1)$
H_3 : $b_0 \Rightarrow \neg(a_2 \vee e_0)$
H_4 : $e_1 \Rightarrow \neg(a_2 \wedge c_2)$
H_5 : $e_1 \Rightarrow (a_0 \Rightarrow d_0)$
H_{14} : $((a_1 \wedge c_1) \vee (b_1 \wedge a_2)) \Rightarrow a_0$
C : $(a_0 \wedge b_0) \Rightarrow$
$((c_0 \vee c_1) \Rightarrow ((d_0 \vee e_0) \vee (c_2 \Rightarrow e_0)))$

Simplifications supplémentaires

H'_2 : $a_2 \vee e_1$
H'_3 : $\neg(a_2 \vee e_0)$
H'_4 : $e_1 \Rightarrow \neg(a_2 \wedge c_2)$
H'_5 : $e_1 \Rightarrow d_0$
C' : $d_0 \vee e_0 \vee (c_2 \Rightarrow e_0)$

Simplifications supplémentaires

true

Autre solution : OBDD

c

$(h_1 \Rightarrow c), \ldots, (h_n \Rightarrow c)$

choisir le plus petit : $c_1 =_{def} (h_{j_1} \Rightarrow c)$

$(h_i \Rightarrow (h_{j_1} \Rightarrow c))$, for all $i \neq j_1$,

choisir le plus petit : $c_2 =_{def} (h_{j_2} \Rightarrow (h_{j_1} \Rightarrow c))$

...

Cas favorable :

$c_k =_{def} (h_{j_k} \Rightarrow (\cdots \Rightarrow (h_{j_1} \Rightarrow c) \cdots))$,

se réduit à *true*

- $\models c_k$
- $\{h_1, \ldots, h_n\} \models c$
- $\{h_1, \ldots, h_n\} \setminus \{h_{j_1}, \ldots, h_{j_k}\}$ inutiles;
- Hypothèse h_{j_k} utile.

La plupart des $h_{j_1}, \ldots, h_{j_{k-1}}$ sont utiles aussi.

Cas général :

On a des prédicats

par exemple $x = y$, $x \leq y$, $x < y$, $x \geq y$, $x > y$.

On utilise une base de données arithmétique

contenant notamment

$\neg(x < y \equiv x \geq y)$, $(x \leq y \equiv (x < y \vee x = y))$

Ces données sont des hypothèses additionnelles.

Exemple (algorithme de Szymanski)

```
H44 (at_p8[j] ==> (at_p10[k] or at_p10[l]))
H43 (at_p8[k] ==> (at_p10[j] or at_p10[l]))
H42 (at_p8[l] ==> (at_p10[j] or at_p10[k]))
H41 ((at_p9[j] or at_p10[j] or at_p11[j] or at_p12[j]) ==> (not at_p4[k]))
H40 ((at_p9[j] or at_p10[j] or at_p11[j] or at_p12[j]) ==> (not at_p4[l]))
H39 ((at_p9[k] or at_p10[k] or at_p11[k] or at_p12[k]) ==> (not at_p4[j]))
H38 ((at_p9[k] or at_p10[k] or at_p11[k] or at_p12[k]) ==> (not at_p4[l]))
H37 ((at_p9[l] or at_p10[l] or at_p11[l] or at_p12[l]) ==> (not at_p4[j]))
H36 ((at_p9[l] or at_p10[l] or at_p11[l] or at_p12[l]) ==> (not at_p4[k]))
H35 ((at_p11[j] or at_p12[j]) ==> #nw=n)
H34 ((at_p11[k] or at_p12[k]) ==> #nw=n)
H33 ((at_p11[l] or at_p12[l]) ==> #nw=n)
H32 (((at_p12[j] or (at_p11[j] and (#ns mod (2**(j-1))=(2**(j-1))-1))) and
     ==> (at_p0[k] or at_p3[k]))
H31 (((at_p12[k] or (at_p11[k] and (#ns mod (2**(k-1))=(2**(k-1))-1))) and
     ==> (at_p0[j] or at_p3[j]))
H30 (((at_p12[k] or (at_p11[k] and (#ns mod (2**(k-1))=(2**(k-1))-1))) and
     ==> (at_p0[l] or at_p3[l]))
H29 (not w[i])
H28 ((at_p5[j] or at_p6[j] or at_p7[j] or at_p8[j] or at_p9[j]) ==> w[j])
H27 (w[j] ==> (at_p5[j] or at_p6[j] or at_p7[j] or at_p8[j] or at_p9[j]))
H26 ((at_p5[k] or at_p6[k] or at_p7[k] or at_p8[k] or at_p9[k]) ==> w[k])
H25 (w[k] ==> (at_p5[k] or at_p6[k] or at_p7[k] or at_p8[k] or at_p9[k]))
H24 ((at_p5[l] or at_p6[l] or at_p7[l] or at_p8[l] or at_p9[l]) ==> w[l])
H23 (w[l] ==> (at_p5[l] or at_p6[l] or at_p7[l] or at_p8[l] or at_p9[l]))
H22 (not s[i])
H21 ((at_p5[j] or at_p6[j] or at_p9[j] or at_p10[j] or at_p11[j] or at_p12
H20 (s[j]==> (at_p5[j] or at_p6[j] or at_p9[j] or at_p10[j] or at_p11[j] o
H19 ((at_p5[k] or at_p6[k] or at_p9[k] or at_p10[k] or at_p11[k] or at_p12
H18 (s[k]==> (at_p5[k] or at_p6[k] or at_p9[k] or at_p10[k] or at_p11[k] o
H17 ((at_p5[l] or at_p6[l] or at_p9[l] or at_p10[l] or at_p11[l] or at_p12
H16 (s[l]==> (at_p5[l] or at_p6[l] or at_p9[l] or at_p10[l] or at_p11[l] o
H15 (a[i])
H14 ((not at_p0[j]) ==> a[j])
H13 (a[j] ==> (not at_p0[j]))
H12 ((not at_p0[k]) ==> a[k])
H11 (a[k] ==> (not at_p0[k]))
H10 ((not at_p0[l]) ==> a[l])
```

```
H09 (a[l] ==> (not at_p0[l]))
H08 (#anw>0)
H07 ((at_p3[j] or at_p4[j] or at_p10[j] or at_p11[j] or at_p12[j]) ==> (#a
H06 ((at_p3[k] or at_p4[k] or at_p10[k] or at_p11[k] or at_p12[k]) ==> (#a
H05 ((at_p3[l] or at_p4[l] or at_p10[l] or at_p11[l] or at_p12[l]) ==> (#a
H04 ((at_p10[j] or at_p11[j] or at_p12[j]) ==> (#snw>0))
H03 ((at_p10[k] or at_p11[k] or at_p12[k]) ==> (#snw>0))
H02 ((at_p10[l] or at_p11[l] or at_p12[l]) ==> (#snw>0))
H01 ((#snw>0) ==> (at_p10[j] or at_p11[j] or at_p12[j] or at_p10[k] or
                   at_p11[k] or at_p12[k] or at_p10[l] or at_p11[l] or at_
C   ((#ns=n) ==> (not (at_p9[j] or at_p10[j] or at_p11[j] or at_p12[j])))
```

Seule H21 est nécessaire

si `(#ns=n) ==> s[j]` est connu, OK

Sinon, ordonnancement des hypothèses.

H21 classé en quatrième position.

70 conditions de vérification non triviales.

En moyenne 42 hypothèses.

Cas favorable :

Toutes les hypothèses additionnelles connues.

Pour 59 des 70 conditions,
le premier choix est le bon.

Dans les autres cas,
peu de tentatives sont nécessaires.

Cas le moins favorable :

Hypothèses additionnelles non connues.

Pour 53 conditions,
le premier choix est le bon.

Pour 13 conditions,
maximum 10 sélections nécessaires.

Pour 4 conditions,
une hypothèse nécessaire au moins
est sélectionnée tardivement.

CAVEAT

Computer

Aided

VErification

And

Transformation of concurrent systems