1 Avant-propos

La notion de treillis est très générale comme nous allons le voir. De part cette généralité, les treillis sont omniprésents. Ils sont utilisés entre autres en théorie des choix, en analyse de concepts, en représentation de l'information, en logique et en sémantique. C'est à ce dernier domaine que nous allons nous intéresser. Plus précisément, je me suis appuyé sur la thèse de Damien Pous "Techniques modulo pour les bisimulations" pour expliquer l'utilité des treillis dans l'étude des bisimulations.

Ce rapport prouve la correction d'une méthode de preuve pour la bisimulation forte. Je n'ai gardé que les notions et lemmes utilisés dans la preuve de cette correction. Il serait possible de prouver ce résultat dans un cadre moins général. Cependant, j'ai tenu à rester dans le cadre très général des treillis complets monoïdaux et à garder les lemmes et définitions intérmédiaire. Ceci afin de montrer que l'on pourrait se servir de ces résulatats intérmédiaires pour montrer des résultats similaires sur d'autres bisimulations.

2 Bisimilarité

2.1 Intérêt

La sémantique s'intéresse au comportement des programmes. Une des questions centrale du domaine est : soit deux programmes P et Q donnés, ont ils le même comportement? La notion d'"avoir le même comportement" n'est pas aussi triviale qu'on pourrait le croire de prime abord, comme le démontre cet exemple sur les automates finis. (fig. 1)

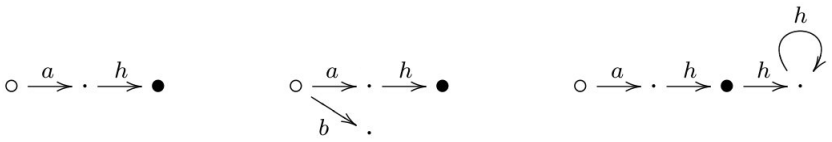

Figure 1: ∘ représente l'état initial. Les • représentent les états acceptant

Si l'on considère les automates comme des fonctions prenant en entrée des mots finis et renvoyant "oui, le mot est accepté par l'automate" ou "non, le mot n'est pas accepté", alors ces trois automates sont équivalents. L'équivalence entre deux automates A_1 et A_2 est ici définie par : quelque soit le choix du mot en entrée par l'utilisateur, il existe un choix de chemin acceptant dans A_1 si et seulement si il en existe un dans A_2.

Cependant, il est possible de considérer autrement les automates. Un automate peut-être une machine écrivant des lettres en suivant les arêtes de son graphe. Dans ce cas là, le deuxième automate peut écrire b avant de se bloquer, ce que les deux autres ne peuvent pas faire. L'équivalence entre deux automates A_1 et A_2 est ici définie par : quelque chemin que A_1 choisisse, A_2 peut prendre un chemin équivalent.

Nous le voyons, une transition peut être considérée ou bien comme une action de l'environnement ou bien comme une action du programme lui-même. La notion d'équivalence est alors changée car le choix n'est pas du même "côté". Lorsque nous considérons les systèmes intéractifs (le π-calcul par exemple), nous devons généraliser ce point de vue en disant que toute transition peut-être considérée des deux côtés. Par exemple les automates de la figure 2 représentent l'interaction entre une machine à café et un informaticien fatigué. Le deuxième automate est problématique, car il ne laisse pas l'informaticien choisir entre thé et café. Nous ne voulons donc plus que ces deux comportements soit considérés comme équivalents. La notion d'équivalence désirée exigerait que le comportement soit le même en donnant alternativement le choix à chaque côté. Cette notion intuitive se formalise avec la notion de bisimilarité.

Figure 2: Signification possible des actions : s= "fournit un sou", t="demander un thé", c="demander un café", T="servir un thé", C="servir un café"

Pour définir l'équivalence entre deux automates, nous allons définir l'équivalence entre deux états p et q. Intuitivement nous voudrions que p et q soit équivalent si et seulement si :

Si p fait une action a en aboutissant dans un état p',
Alors q peut faire la même action a et aboutir dans un état q' équivalent à p'. (1)

Nous pouvons remarquer que cette définition fait appel à elle même, il faut donc être sûr que c'est une définition correcte. Le premier réflexe serait de

définir la relation d'équivalence par induction. Cependant, ce n'est pas possible car il se peut que le processus étudié contienne des cycles. Nous utiliserons donc la co-induction, qui est la notion duale de l'induction.

Nous définirons en fait la bisimilarité $\sim$ comme la plus grande relation satisfaisant (1). Alors, pour montrer que deux automates sont équivalents, nous définiront une relation $\approx$ entre les états et supposeront que deux états en relation sont équivalents. Puis nous vérifierons que cette équivalence vérifie (1). C'est à dire, en supposant que $p \approx q$:

$$\text{Si } p \rightarrow^a p' \text{ Alors il existe } q' \text{ tel que } q \rightarrow^a q' \text{ et } p' \approx q' \tag{2}$$
$$\text{Si } q \rightarrow^a q' \text{ Alors il existe } p' \text{ tel que } p \rightarrow^a p' \text{ et } p' \approx q' \tag{3}$$

Une formulation équivalente mais moins intuitive de (2) est $\forall a, \leftarrow^a \ . \approx \subseteq \approx . \leftarrow^a$. C'est sous cette forme que nous retrouverons la bisimilarité dans les définitions formelles.

Le résulat de ce rapport a pour but de faciliter les preuves par coinduction. Il permet de remplacer (2) par

$$\text{Si } p \rightarrow^a p' \text{ Alors il existe } q' \text{ tel que } q \rightarrow^a q' \text{ et } p' \simeq q' \tag{4}$$

Avec $\simeq$ une relation plus grande que $\approx$.

3 Treillis

3.1 définitions de base

Définition 3.0.1. *Un treillis est un ensemble ordonné $(X, \leq, \wedge, \vee)$ tel que toute paire d'éléments a, b de X admette une borne inférieure (notée $a \wedge b$) et une borne supérieure (notée $a \vee b$) dans X.*

Définition 3.0.2. *Un treillis $(X, \leq, \vee)$ est dit complet si tout sous-ensemble de X possède une borne supérieure.*

Il n'est pas utile de noter la borne inférieure car elle se déduit de la borne supérieure : $\bigwedge Y = \bigvee \{x \in X / \forall y \in Y, x \leq y\}$.

Exemple 3.1.

- $(\mathcal{P}(E), \subseteq, \cup)$ *est est un treillis complet.*
- $(\mathbb{R}, \leq, sup)$ *est un treillis non complet car* $\mathbb{R}$ *lui même n'a pas de majorant. Cependant,* $\mathbb{R} \cup \{-\infty, +\infty\}$ *est un treillis complet.*
- *Par contre* $(\mathbb{Q}, \leq, sup)$ *n'est pas complet, même en lui rajoutant les infinis. En effet,* $\{q / q^2 \leq 2\}$ *n'a pas de borne supérieure dans* $\mathbb{Q}$.

Théorème 3.1 (Principe de dualité)**.** *Si un énoncé est vrai, son énoncé dual obtenu en remplaçant respectivement* $(\leq, \wedge, \vee)$ *par* $(\geq, \vee, \wedge)$*est vrai.*

Proof. On vérifie aisément que si $< X, \leq, \wedge, \vee >$ est un treillis, alors $< X, \geq, \vee, \wedge >$ est également un treillis. Ce principe nous permettra de prouver deux théorèmes duaux pour le prix d'un. □

Le lemme suivant, intuitivement évident, montre le genre de raisonnement que l'on fait sur un treillis :

Lemme 3.1.1. *Soit* $Y, Z \subseteq X$*,*
Si $\forall y \in Y, \exists z \in Z, y \leq z$ *Alors* $\bigvee Y \leq \bigvee Z$

Proof.

$$\left.\begin{array}{c}\text{Par hypothèse, il existe } z \text{ tel que, } y \leq z \\ \text{La borne sup. est un majorant, } z \leq \bigvee Z\end{array}\right\} \rightarrow y \leq \bigvee Z$$

Donc, $\bigvee Z$ est un majorant de Y.
Or, par définition, $\bigvee Y$ est le plus petit des majorants de Y.
Donc, $\bigvee Y \leq \bigvee Z$ □

3.2 Treillis complet monoïdal continu

Afin de représenter la composition de deux relations, nous allons enrichir les treillis d'une structure de monoïde.

Définition 3.1.1. *Un* ***treillis complet monoïdal*** *est un quintuplet* $< X, \leq, \bigvee, ., 1 >$ *tel que:*

- $< X, \leq, \bigvee >$ *est un treillis complet*
- $< X, ., 1 >$ *est un monoïde*
- $\forall x, x', y, y' \in X, \left.\begin{array}{c} x \leq x' \\ y \leq y' \end{array}\right\} \Rightarrow x.y \leq x'.y'$

Un treillis complet monoïdal est dit ***continu*** *si pour toutes parties* Y*,* Z *de* X*,*

$$\bigvee(Y.Z) = (\bigvee Y).(\bigvee Z) \tag{5}$$

Exemple 3.1.1. *Le treillis complet monoïdal que nous utiliserons est* $< \mathcal{R}, \subseteq, \bigcap, ., I >$ *avec*

- $\mathcal{R} = \mathcal{P}(E \times E)$ *l'ensemble de relations binaires sur E. Avec E l'ensemble des états de notre système.*
- $x\ (R.R')\ y \Leftrightarrow \exists z, xRz$ *et* zRy
- $I = \{(x, x)/x \in E\}$

C'est dans l'optique de cette application qu'il faut comprendre les définitions suivantes.

Définition 3.1.2. *Soit $x \in X$ et P une propriété sur X,*
*x est **réflexif** ssi $1 \leq x$*
*x est **transitif** ssi $x.x \leq x$.*
*La **P-fermeture** de x est, si il existe, le plus petit y tel que $x \leq y$ et $P(y)$.*

Définition 3.1.3. *Soit $x \in X$, x^n est défini inductivement par :*

$$x^n = \begin{cases} 1, \text{ si } n = 0 \\ x.x^{n-1}, \text{ si } n \geq 1 \end{cases}$$

Proposition 3.2. *Tout élement x de X possède:*

Une fermeture transitive	$x^+ = \bigvee_{n \geq 1} x^n$
Une fermeture réflexive	$x^= = x \vee 1$
Une fermeture reflexo-transitive	$x^* = (x^+)^= = \bigvee_{n \geq 0} x^n$

3.3 Extension aux fonctions

Définition 3.2.1. *$f : X \to X$ est une **fonction croissante** si et seulement si*

$$\forall x, y \in X, \quad x \leq y \Rightarrow f(x) \leq f(y)$$

On note $X^{<X>}$ l'ensemble des fonctions croissantes de X.

Définition 3.2.2. *Une fonction est dite **extensive** si et seulement si*

$$\forall x, x \leq f(x)$$

Définition 3.2.3. *Nous pouvons étendre les opérations précédentes aux fonctions :*

$f \leq g$	$\Leftrightarrow$	*Pour tout x, $f(x) \leq g(x)$*
$(\bigvee F)(x)$	$=$	$\bigvee_{f \in F}(Fx)$
$(f \hat{} g)(x)$	$=$	$(fx).(gx)$

Définition 3.2.4. *On définit $f^{(k)}$ comme l'itérée k-ième de f (par la composition fonctionnelle)*
$f^\omega = \bigvee_{k \in \mathbb{N}} f^{(k)}$

Proposition 3.3. *$< X^{<X>}, \leq, \bigvee, ., \hat{1} >$ est un treillis complet monoïdal. $\hat{}$ est continu si et seulement si . est continu*
$< X^{<X>}, \leq, \bigvee, \circ, id_X >$ est un treillis complet monoïdal.

Le treillis défini avec la composition n'est pas continu. Cependant, nous avons quand même des propriétés intéressantes entre la composition et la borne supérieure, comme le lemme suivant.

Lemme 3.3.1. *Soit f une fonction croissante et G un ensemble de fonctions croissantes*

$$\bigvee_{g \in G} f \circ g \leq f \circ (\bigvee G)$$

Proof. Soit $x \in X$,

$$\forall g \in G, g(x) \leq (\bigvee G)(x)$$
$$\forall g \in G, f(g(x)) \leq f((\bigvee G)(x)) \text{ par croissance de f}$$
$$\forall g \in G, (f \circ g)(x) \leq (f \circ \bigvee G)(x) \text{ par croissance de f}$$
$$\bigvee_{g \in G} (f \circ g(x)) \leq (f \circ \bigvee G)(x)$$
$$(\bigvee_{g \in G} f \circ g)(x) \leq (f \circ \bigvee G)(x)$$

□

4 Preuves par coinduction

En informatique, nous utilisons régulièrement des preuves par induction. La plus simple étant la récurrence sur les entiers. $\mathbb{N}$ est défini comme le plus petit point fixe de la fonction $\phi : T \to \{0\} \cup s(T)$ définie sur les sous-ensembles de termes sur $\{0, s\}$. Pour démontrer que $\mathbb{N}$ satisfait une propriété P, il suffit de montrer que l'ensemble E_P des termes vérifiant P est un point fixe de la fonction ϕ. Comme $\mathbb{N}$ est le plus petit des points-fixes, nous avons $\mathbb{N} \subseteq E_P$ donc tous les entiers vérifient P.

La co-induction est la notion duale de l'induction. Un ensemble X est défini par coinduction si il est le plus grand point fixe d'une certaine fonction ϕ. Pour montrer qu'un certain ensemble Y est inclus dans X, il suffit alors de montrer que Y est un point fixe de ϕ. Par exemple, les listes infinies triées peuvent être définies comme le plus grand point fixe de $\phi : X \to \{[]\} \cup \{x :: t :: q \ / \ t :: q \in X \text{ et } x \leq t\}$. On prouve alors que $0 :: 1 :: 2 :: 3 :: ...$ est triée en montrant que $\{n :: n+1 :: n+2 :: .../n \in \mathbb{N}\}$ est un point fixe de ϕ. Comme nous le verrons dans les sections suivantes, la bisimilitude introduite informellement dans la section 1 est en fait une définition par coinduction. Dans cette partie, nous allons étudier la notion de preuves par coinduction en se plaçant dans le cadre des treillis.

Théorème 4.1 (Knaster-Tarski)**.** *Soit L un treillis complet et $F : L \to L$ une fonction croissante, alors F admet un plus petit point fixe et un plus grand point fixe qui sont respectivement :*

$$\mu(F) := \bigwedge \{x \in L / F(x) \leq x\}$$

$$\nu(F) := \bigvee \{x \in L / x \leq F(x)\}$$

Proof. Il est suffisant de montrer l'existence d'un plus grand point fixe. L'existence du plus grand point fixe en découlant par dualité (3.1).

Posons $PostPF(F) = \{x \in L / x \leq F(x)\}$, l'ensemble des post point fixes de F.

Soit $x \in PostPF(F)$,

$$\begin{aligned} x &\leq \nu(F) && \text{par définition de } \nu(F) \\ F(x) &\leq F(\nu(F)) && \text{par croissance de F} \\ x \leq F(x) &\leq F(\nu(F)) && \text{par définition de PostPF} \end{aligned}$$

$$\left.\begin{array}{r} F(\nu(F)) \text{ majore } PostPF(F) \\ \nu(F) \text{ plus petit majorant de } PostPF(F) \end{array}\right\} \rightarrow \nu(F) \leq F(\nu(F)) \qquad (6)$$

Par croissance de F, $F(\nu(F)) \leq F(F(\nu(F)))$
Donc $F(\nu(F)) \in PostPF(F)$.
Or, $\nu(F)$ majore PostPF(F), donc

$$F(\nu(F)) \leq \nu(F) \qquad (7)$$

En combinant (6) et (7), on obtient $\nu(F) = F(\nu(F))$.
$\nu(F)$ est un point fixe de F.

Tout point fixe étant un post point fixe, et $\nu(F)$ majorant PostPF(F), $\nu(F)$ est le plus grand point fixe de F. □

Maintenant que nous avons défini formellement la définition par coinduction, nous allons formaliser notre objectif (4). Ici, nous n'avons pas de relations mais de simples treillis complets. Remplacer la relation $\approx$ par une relation $\simeq$ plus grande revient ici à remplacer la génératrice s par $s \circ f$ avec f une fonction extensive.

Définition 4.1.1. *Soit s une fonction croissante*
f est ***correcte*** *pour s, si* $\nu(s \circ f) \leq \nu(s)$
f est ***compatible*** *avec s, si* $f \circ s \leq s \circ f$

Théorème 4.2. *Si f est compatible avec s, alors f est correcte pour s*

Proof. Soit f une fonction croissante compatible avec s,
Nous allons montrer

$$\nu(s \circ f) \leq f^{\omega}(\nu(s \circ f)) \leq s(f^{\omega}(\nu(s \circ f))) \leq \nu s$$

- La première inégalité découle de la définition de f^ω, en effet :

$$f^\omega = \bigvee_{n\in\mathbb{N}} f^n$$
$$f^\omega = id_X \vee \bigvee_{n\geq 1} f^n$$
$$f^\omega \geq id_X$$

- Prouvons la seconde inégalité : $f^\omega(\nu(s \circ f)) \leq s(f^\omega(\nu(s \circ f)))$
  Dans un souci de clarté, on notera $\nu := \nu(s \circ f))$:

$$\left.\begin{array}{l} \nu \leq (s \circ f)\nu \\ f^k(\nu) \leq s \circ f^{k+1}(\nu) \Rightarrow f^{k+1}(\nu) \leq s \circ f^{k+2}(\nu) \end{array}\right\} \rightarrow \text{par récurrence } f^k(\nu) \leq s \circ f^{k+1}(\nu)$$

$$\forall k \in \mathbb{N}, f^k(\nu) \leq s \circ f^{k+1}(\nu)$$
$$\forall k \in \mathbb{N}, f^k(\nu) \leq s \circ f^\omega(\nu)$$
$$\bigvee_{k\in\mathbb{N}} f^k(\nu) \leq s \circ f^\omega(\nu)$$
$$(\bigvee_{k\in\mathbb{N}} f^k)\nu \leq s \circ f^\omega(\nu)$$
$$f^\omega(\nu) \leq s \circ f^\omega(\nu)$$

- La seconde inégalité dit que $\nu(s \circ f)$ est un pré-point fixe de s. Or, νs est le plus grand pré-point fixe de s, donc $\nu(s \circ f) \leq \nu s$.

□

La propriété de compatibilité ne nous intéresse pas en soi. Nous nous intéressons à la correction pour s. Cependant, les fonctions compatibles jouissent de nombreuses propriétés de préservation intéressantes (du côté de f aussi bien que du côté de s). Ce qui permet de prouver la compatibilité de manière modulaire. Pour prouver qu'une fonction f complexe est compatible avec s il suffit souvent de prouver que les fonctions élémentaires à partir desquelles f est construites sont compatibles avec s.

Proposition 4.3. *L'identité est compatible avec s.*
Pour toute s-simulation x, $\hat{x} : y \rightarrow x$ est compatible avec s.
La borne supérieure, la composition et l'itération infinie ($f \rightarrow f^\omega$) préservent la compatibilité avec s.

Proof. Montrons la préservation par borne supérieure.
Supposons que pour tout $f \in F$, f est compatible avec s.
Soit $x \in X$

$$\begin{aligned}((\bigvee F) \circ s)x &= (\bigvee F)(sx) \\ &= \bigvee_{f \in F} (f(s\ x)) \\ &\leq \bigvee_{f \in F} (s(f\ x)) \quad (3.1.1) \\ ((\bigvee F) \circ s)x &\leq (s \circ \bigvee F)(x) \quad (3.3.1)\end{aligned}$$

Donc $\bigvee F$ est bien compatible avec s. □

Définition 4.3.1. *s* ***respecte*** *le monoïde si :*
1 est une simulation
$\forall x, y \in X,\ s(x).s(y) \leq s(x.y)$

Proposition 4.4. *Si s respecte le monoïde,*
Le produit de deux fonctions compatibles avec s est compatible avec s.
La fonction de fermeture réflexive transitive id_X^* est compatible avec s.

Proof. Soient f et g deux fonctions compatibles avec s.
Soit $x \in X$.

$$\begin{aligned}((f\hat{.}g) \circ s)(x) &= (f\hat{.}g)(sx) \\ &= (f(s\ x)).(g(s\ x)) \\ &\leq (s(f\ x)).(s(g\ x)) && \text{compatibilité de f et g} \\ &\leq s(fx.gx) && \text{s respecte le monoïde} \\ &\leq (s \circ (f\hat{.}g))x\end{aligned}$$

Donc on a bien $f\hat{.}g$ compatible avec s.

Montrons maintenant par réccurrence la compatibilité de id_X^n avec s pour tout $n \in \mathbb{N}$:

- $id_X^0 = \hat{1}$.

$$\begin{aligned}(\hat{1} \circ s)(x) &= 1 \\ &\leq s(1) && \text{respect du monoïde} \\ (\hat{1} \circ s)(x) &\leq (s \circ \hat{1})(x) \\ (id_X^0 \circ s)(x) &\leq (s \circ id_X^0)(x)\end{aligned}$$

- $id^1_X = id_X$

$$id^1_X \circ s = s$$
$$id^1_X \circ s = s \circ id^1_X$$
$$id^1_X \circ s \leq s \circ id^1_X$$

- Supposons id^k_X compatible avec s. Donc, d'après la première partie de la proposition, $id^{k+1}_X = id^k_X \hat{} id^1_X$ est compatible avec s.

Donc, pour tout $n \in \mathbb{N}, id^k_X$ est compatible avec s.
Donc, $id^*_X = \bigvee_{k\in\mathbb{N}} id^k_X$ est compatible avec s (4.3). □

Proposition 4.5. *Si f est compatible avec tous les éléments de S, f est compatible avec* $\bigwedge S$

Proof. Soit $x \in X$,

$$(f \circ \bigwedge S)(x) = f(\bigwedge_{s\in S} s(x))$$
$$\forall s \in S, (f \circ \bigwedge S)(x) \leq f(s(x))$$
$$\forall s \in S, (f \circ \bigwedge S)(x) \leq s(f(x))$$
$$(f \circ \bigwedge S)(x) \leq \bigwedge_{s\in S} (s(fx))$$
$$(f \circ \bigwedge S)(x) \leq (\bigwedge S)(fx)$$
$$(f \circ \bigwedge S)(x) \leq (\bigwedge S \circ f)(x)$$

□

Définition 4.5.1. *$i : X \to X$ est une **fonction de conversion** si et seulement si i est croissante et involutive.*
Si l'on fixe une fonction de conversion, on définit alors :
$\bar{x} = i(x) \quad \bar{f} = i \circ f \circ i \quad \tilde{f} = f \wedge \bar{f}$

Proposition 4.6. *f est compatible avec s si et seulement si $\bar{f}$ est compatible avec $\bar{s}$*

Proof.

$$\begin{aligned}\bar{f} \circ \bar{s} &= (i \circ f \circ i) \circ (i \circ s \circ i) \\ &= i \circ (f \circ s) \circ i \\ &\leq i \circ (s \circ f) \circ i \\ &\leq i \circ s \circ i \circ i \circ f \circ i \\ \bar{f} \circ \bar{s} &\leq \bar{s} \circ \bar{f}\end{aligned}$$

□

5 Bisimulations étiquetées

Définition 5.0.1. *Une **algèbre relationnelle** est un treillis complet monoïdal continu muni d'une fonction de conversion tel que*
$\forall x, y \in X \quad x^{-}.y = \bar{y}.\bar{x}$
$\forall x, y, z \in X \quad (x.y) \vee z \leq x.(y \vee (\bar{x}.z))$

Définition 5.0.2. *On définit $\mathcal{R} = \mathcal{P}(E \times E)$ l'ensemble des relations binaires sur E. On notera dans la suite xPy pour $(x, y) \in P$. On définit les opérations suivantes :*

$$\begin{aligned} S \subseteq R \text{ si } \forall p, q \in E, pSq &\Rightarrow pRq \\ R.S =&\{(p, q)/\exists r \in E, pRr \text{ et } rRq\} \\ I =&\{(p, p)/p \in E\} \\ \bar{R} =&\{(q, p)/pRq\} \end{aligned}$$

Proposition 5.1. *$< \mathcal{R}, \subseteq, \cup, ., I, ^{-} >$ est une algèbre relationnelle appellée algèbre relationnelle propre.*

C'est désormais dans le cadre de cette algèbre relationnelle particulière que nous travaillerons. Nous pourrions donner des résultats un peu plus généraux. Cependant, la forme des résultats sont de toute manières fortement guidés par l'application sous-jacente aux relations binaires.

Définition 5.1.1. *Un **système abstrait de transitions étiquetées** (que nous noterons **LTS** pour "labelled transition system") est une collection de relations indexée par L.*

Définition 5.1.2. *Pour tout $P \in \mathcal{R}$, nous notons*

$$S_P = \{Q \in \mathcal{R}/Q \leq P \text{ et } \forall \alpha \in L, \leftarrow^{\alpha} .Q \subseteq P. \leftarrow^{\alpha}\}$$

On appelle alors génératrice de la simulation forte

$$\vec{s}(P) = \bigvee S_P$$

On vérifie aisément que $\vec{s}$ est croissante. Il est moins aisé de vérifier que cette génératrice définit effectivement la simulation forte. Dans sa thèse, Damien Pous introduit deux formalismes. Le formalisme que nous avons présenté qui est plus facilement manipulable pour les preuves, ainsi qu'un formalisme basé sur la notion de progression. Il est possible de prouver que les deux formalismes sont équivalents. Afin de simplifier, nous n'étudierons pas dans ce rapport les progressions. La définition de la simulation forte est donc alourdie.

Lemme 5.1.1. *$\vec{s}$ respecte le monoïde*

Proof. Soit $P, Q \in R$, montrons que $\vec{s}(P).\vec{s}(Q) \leq \vec{s}(P.y)$.
Soit $P' \in S_P$ and $y' \in S_y$,

$$\left.\begin{array}{l} P' \leq P \text{ par définition de } S_P \\ Q' \leq Q \text{ par définition de } S_Q \end{array}\right\} \Rightarrow P'.Q' \leq P.Q(\text{car . préserve l'ordre})$$

Soit $\alpha \in L$

$$\begin{array}{rl} \alpha.P' \leq P.\alpha & \text{définition de } S_P \\ (\alpha.P').Q' \leq (P.\alpha).Q' & \text{. préserve l'ordre} \\ \alpha.(P'.Q') \leq P.(\alpha.Q') & \\ \alpha.(P'.Q') \leq P.Q.\alpha & \text{définition de } S_Q \end{array}$$

Donc $P'.Q' \in S_{P.Q}$

$$\begin{array}{rl} S_P.S_Q \subseteq S_{P.Q} & \\ \forall z \in S_P.S_Q, z \leq \bigvee S_{P,Q} = \vec{s}(P.Q) & \\ \bigvee(S_P.S_Q) \leq \vec{s}(P.Q) & \\ (\bigvee S_P).(\bigvee S_Q) \leq \vec{s}(P.Q) & \text{Continuité du monoïde} \\ \vec{s}(P).\vec{s}(Q) \leq \vec{s}(P.Q) & \end{array}$$

□

Corollaire 5.1.1. *$id^*_{\mathcal{R}} : P \to P^*$ est compatible avec $\vec{s}$*

Proof. Immédiat d'après le lemme précédent et 4.4 □

5.1 Extension aux jeux à deux côtés

Définition 5.1.3. *Une **clôture** C est une fermeture symétrique telle que*

$$C(P.Q) \leq C(P).C(Q)$$

Afin d'obtenir la bisimilitude forte, il suffit de symmétriser :

Définition 5.1.4. *On définit $\tilde{s} = \vec{s} \wedge \overleftarrow{\vec{s}}$, clôture symétrique de $\vec{s}$ La bisimilarité forte est définie par $\sim = \nu\tilde{s}$*

Corollaire 5.1.2. *Si C est une clôture compatible avec $\vec{s}$, Alors $P \to (C(P) \vee \sim)^*$ est compatible avec $\tilde{s}$.*

Proof.

$$\left.\begin{array}{l} C \text{ compatible avec } \vec{s} \\ \hat{\sim} \text{ compatible avec } \vec{s} \end{array}\right\} \to C \vee \hat{\sim} \text{ compatible avec } \vec{s} \text{ grâce à 4.3}$$

Or $\vec{s}$ respecte le monoïde, donc $(C \vee \hat{\sim})^*$ est compatible avec $\overleftarrow{\vec{s}}$ grâce à (4.4).
Par symétrie puis passage à la borne inférieure, $(C \vee \hat{\sim})^*$ respecte $\tilde{s}$. □

Nous avons donc une technique très puissante. La clôture peut, entre autres, représenter une clôture par contexte.