NOTES SUR LA RAISONNEMENT LOGIQUE ET LES PREUVES

PREPRINT

Dominique Méry *
LORIA
Université de Lorraine
dominique.mery@loria.fr
https://members.loria.fr/Mery
https://mery54.github.io/mery/

February 27, 2025

ABSTRACT

Cette note décrit le raisonnement logique tel que nous le faisons quand nous devons vérifier des conditions de vérifications dans le cas des contrats. Nous expliquons comment il faut faire pour dériver une preuve de manière systématique selon les séquents.

Keywords Formal method · sequents · proving · safety · verification conditions · invariant · context · Formal IDE

*Supported by the ANR Project EBRP-EventB-Rodin-Plus (ANR-19-CE25-0010) and by the ANR Project DISCONT (ANR-17-CE25-0005)

Contents

1 Raisonnement logique

1.1 Raisonnement systématique

Nous allons utiliser des expression logiques de la forme suivante $A \Rightarrow B$ où A et B sont des expressions construites à l'aide des connecteurs logiques $\wedge$ et des formules élémentaires comme des équations $E = F$, des inéquations $E \leq F$ ou $E \geq F$, des expressions d'appartenance $x \in U, \ldots$ Par exemple, $x = 1 \wedge y = 2 \Rightarrow x + y = 3$. On peut essayer de s'assurer qu'une telle expression est vallide en utilisant des opérations de simplification et en essayant de réduire c ette expression soit en $FALSE \Rightarrow B$ soit $A \Rightarrow TRUE$. On peut alors construire u!n raisonnement qui consiste à trouver un cheminement de simplification comme suit selon un raisonnement déductif consistant à remplacer un terme par un autre de manière correcte:

$$x = 1 \wedge y = 2 \Rightarrow x + y = 3 \tag{1}$$

On remplace x par 1 à droite.

$$x = 1 \wedge y = 2 \Rightarrow 1 + y = 3 \tag{2}$$

On remplace y par 2 à droite.

$$x = 1 \wedge y = 2 \Rightarrow 1 + 2 = 3 \tag{3}$$

On simplifie par calcul.

$$x = 1 \wedge y = 2 \Rightarrow 3 = 3 \tag{4}$$

On simplifie $3 = 3$ en TRUE.

$$x = 1 \wedge y = 2 \Rightarrow TRUE \tag{5}$$

On atteint une expression de la forme $A \Rightarrow TRUE$. On en déduit que l'expression $x = 1 \wedge y = 2 \Rightarrow x + y = 3$ est valide.

L'objet de cette section est de donner quelques éléments de la logique élémentaire qu'on peut utiliser pour dériver la validité ou non des expressions de la forme $A \Rightarrow B$. La vérification de la validité est une opération qui peut être mécanisée jusqu'à un certain niveau de complexité des formules $A \Rightarrow B$. enons un traitement automatique de cette vérification par Frama-c comme l'indique le message émis à l'exécution de la vérification du contrat de cet example.

```
macmery$ frama-c -wp  ex.c
[kernel] Parsing ex.c (with preprocessing)
[wp] Warning: Missing RTE guards
[wp] 2 goals scheduled
[wp] Proved goals:    2 / 2
  Qed:             2  (0.40ms-0.96ms)
```

Npus allons approfondir ces éléments en introduisant deux familles de formules logiques avec des règles simples de manipulation qui garantissenent la préserbation de la validité des formules. Dans le petit exemple ci-dessus Figure **??**, les formules des équuations $1 \ldots 4$ sont équivalentes et valides puisque la dernière formule est valide.

Le calcul des propositions est un premier cadre dans lequel nous allons définir les notions relatives à la syntaxe et à la sémantique comme les modèles, la consistance, la complétude, la déduction. .... Il parait, à première vue, rudimentaire mais néanmoins permet de bien poser les problèmes des systèmes formels.

1.2 Syntaxe de formules propositionnelles

Nous donnons dans cette partie quelques notations très utiles par la suite. Nous définissons la notion de formules qui sont les termes manipulés dans ce système. On note $\mathcal{V}$ un ensemble infini dénombrable d'éléments appelés symboles de variables propositionnelles:

$\mathcal{V} = \{A, B, C, D, \ldots, A_i, B_i, \ldots, P, Q, R, S, \ldots\}$.

On note $\mathcal{C}$ un ensemble d'éléments appelés symboles de connection et de séparation:

$\mathcal{C} = \{\vee, \wedge, \sim, \Rightarrow, \equiv, (,), \ldots\}$

On suppose que $\mathcal{C} \cap \mathcal{P} = \varnothing$.

Definition 1 Syntaxe des formules

Une formule est une suite finie de symboles de $\mathcal{C} \cup \mathcal{P}$ obtenue par application finie des règles suivantes:

1. tout symbole de variable propositionnelle est une formule dite atomique.
2. si φ est une formule, alors $\sim (\varphi)$ est une formule.
3. si φ et ψ sont deux formules, alors $(\varphi) \wedge (\psi)$, $(\varphi) \Rightarrow (\psi)$, $(\varphi) \vee (\psi)$, $(\varphi) \equiv (\psi)$ sont des formules.

Une formule sur $\mathcal{C} \cup \mathcal{P}$ est appelée une formule propositionnelle. Dans ce chapitre, une formule sera implicitement propositionnelle. Le parenthésage permet de rendre l'analyse non ambigue. Nous pourrions aussi utiliser une grammaire pour définir l'ensemble des formules. On peut construire une grammaire des ces formules et ainsi les représenter et les analyser.

Une suite finie de formules $(\varphi_0, \varphi_1, \ldots, \varphi_n)$ est une construction si elle vérifie pour tout i de $\{0, \ldots, n\}$, l'une des conditions suivantes:

1. φ_i est un symbole de variable propositionnelle.
2. il existe $j < i$ tel que φ_i est $\sim (\varphi_j)$.
3. il existe j et k tels que $j < i$ et $k < i$ et φ_i est $(\varphi_j)\ op\ (\varphi_k)$ où $op\ \in \{\wedge, \vee, \Rightarrow, \equiv\}$

La construction $(\varphi_0, \varphi_1 \ldots \varphi_n)$ est une construction de φ_n. Toute sous-suite $(\varphi_0, \varphi_1 \ldots \varphi_i)$ est une construction de φ_i, pour tout i de $\{0, \ldots, n\}$. Une formule φ admet une infinité dénombrable de constructions, il suffit d'ajouter des variables propositionnelles de $\mathcal{P}$ dans une construction de φ. Le nombre d'application des règles (2) ou (3) est l'ordre de complexité de la construction.

Exemple 1

1. $(P, (P) \wedge (P), P)$ est une construction de P d'ordre 1.
2. $(P, Q, R, (P) \wedge (Q), ((P) \wedge (Q)) \Rightarrow (R))$ est une construction de $((P) \wedge (Q)) \Rightarrow (R)$ d'ordre 2.
3. $(P, ((P) \wedge (P)) \Rightarrow (P), P)$ n'est pas une construction.

Une formule φ se décompose suivant les règles de construction. On définit parallèlement la notion d'arbre de décomposition d'une formule φ. Cette décomposition correspond à la construction d'un arbre syntaxique associé à φ en utilisant les règles de grammaire adéquates.

Definition 2 Décomposition d'une formule φ

La décomposition d'une formule φ et la construction d'un arbre de décomposition associé obéissent aux règles suivantes:

1. si φ est un symbole de variable propositionnelle, alors φ ne se décompose pas et son arbre de décomposition est réduit à φ.
2. si φ est obtenue par la règle 2, alors φ se décompose en ψ où φ est $\sim (\psi)$ et son arbre de décomposition est:
   $\sim$
   |
   ψ
3. si φ est obtenue par la règle 3, alors φ se décompose en ψ_1 et ψ_2 où φ est $(\psi_1)\ op\ (\psi_2)$ et son arbre de décomposition est:

op

$\psi_1 \quad \psi_2$

On peut ajouter aux formules une formule de base qui est la constante false. Son statut est celui d'un connecteur d'arité 0. Nous le définirons ainsi $FAUX \stackrel{def}{=} \sim (P) \wedge (P)$. De même, nous pouvons définir une constante true comme suit: $VRAI \stackrel{def}{=} \sim (P) \vee (P)$. le propositioon P est quelconque. Cette définition sera justifiée par la suite. Les expressions de la forme $A \Rightarrow B$ sont des formules logiques propositionnelles étendues par des variables dans des expressions de la forme $x = y$ ou $x > y$ etc On note $\mathcal{P}rop(\mathcal{P})$, l'ensemble des formules sur $\mathcal{P} \cup \mathcal{C}$. On peut considérer les expressions éléments iare de la forme $x = 3$ ou $x > 5$ ou $x < y$ comme des propositions particulières.D'une certaine mesure, il s'agit d'une technique d'abstraction associant à toute formule élémentaire une variable propositionnelle.

1.3 Sémantique des propositions

Les formules de $\mathcal{P}rop(\mathcal{P})$ ont une existence purement syntaxique. Nous leur associons un sens permettant de signifier les connecteurs. On note $\mathcal{B} = \{0, 1\}$ et $\mathcal{B}_n$, les fonctions n-aires à valeur dans $\mathcal{B}$. Soit $f \in \mathcal{B}_n$ une table de vérité de f est une structure ayant n+1 colonnes et 2^n lignes.
Exemple 2

$$f \in B_1 : \begin{array}{cc} 0 & \epsilon_0 \\ 1 & \epsilon_1 \end{array}$$

$$f \in B_2 : \begin{array}{ccc} 0 & 0 & \epsilon_0 \\ 0 & 1 & \epsilon_1 \\ 1 & 0 & \epsilon_2 \\ 1 & 1 & \epsilon_3 \end{array}$$

A tout symbole de $\mathcal{C}$, on associe une fonction de $\bigcup_{i \geq 0} \mathcal{B}_i$ ou une table de vérité. Soit $\mathcal{B} = (\mathcal{B}, \bigcup_{i \geq 0} \mathcal{B}_i$. $\mathcal{B}$ est une algèbre. $\mathcal{B}$ est l'algèbre booléenne. Soit φ une formule de $\mathcal{P}rop(\mathcal{P})$. Une fonction booléenne $[\![\varphi]\!]$, dont l'arité est le nombre de variables figurant dans φ, peut être associée à φ. On dit que $[\![\varphi]\!]$ est la fonction de vérité de φ.
Donnons une justification à cette définition. Pour justifier cette définition, nous en donnons une version constructive.
$[\![\ ]\!] : \mathcal{P}rop(\mathcal{P}) \longrightarrow \bigcup_{i \geq 0} \mathcal{B}_i$

1. Pour toute variable propositionnelle P de $\mathcal{P}$, on associe la fonction notée $[\![P]\!]$ et définie par: $[\![P]\!](\varepsilon) = \varepsilon$, pour tout $\varepsilon \in \mathcal{B}$.
2. Si la formule φ est de la forme $\sim (\psi)$, alors $[\![\varphi]\!] = \overline{[\![\psi]\!]}$.
3. Si la formule φ est de la forme $(\varphi_1)\ op\ (\varphi_2)$, alors $[\![\varphi]\!] = [\![\varphi_1]\!]\ fb(op)\ [\![\varphi_2]\!]$ où $fb(op)$ est la fonction booléenne associée à op.

Une formule φ a donc une sémantique complètement définie.
Une formule φ est une tautologie, si $[\![\varphi]\!]$ est la fonction booléenne uniformément égale à 1. Une formule φ est une antilogie, si $[\![\varphi]\!]$ est la fonction booléenne uniformément égale à 0. Deux formules ayant la même table de vérité sont dites synonymes.

► **Théorème 1**

Pour toute fonction booléenne f de $\mathcal{B}^n$, il existe une formule φ écrite avec $\sim, \wedge, \vee$, à partir de n symboles de proposition et telle que $[\![\varphi]\!] = f$.

PREUVE:

1. $f = 0$:
   $f = [\![(A_1 \wedge \sim A_1)) \vee \ldots (A_n \wedge \sim A_n)]\!]$
2. $f \neq 0$:
   Soit $(\varepsilon_1, \ldots, \varepsilon_n)$ telle que $f(\varepsilon_1, \ldots, \varepsilon_n) = 1$. On associe à $(\varepsilon_1, \ldots, \varepsilon_n)$ la formule suivante:
   $\alpha_1 \wedge \ldots \wedge \alpha_n$ où $\alpha_i = \sim A_i$ si $\varepsilon_i = 0$ et $\alpha_i = A_i$, si $\varepsilon_i = 1$
   Alors $f = [\![\bigvee_{j=\{1,\ldots,2^n\}} \bigwedge_{i \in \{1,n\}} \alpha_i^j]\!]$. On suppose que les termes non nuls figurent dans la sommation.

□

Corollaire 1 *Toute formule φ est synonyme d'une forme dite en forme normale disjonctive ou conjonctive.*

On appelle valuation ou réalisation, une application de $\mathcal{P}$ dans $\mathcal{B}$, qui associe à tout symbole de $\mathcal{P}$ une valeur 0 ou 1. On appelle valeur d'une formule φ pour une valuation δ donnée et on note $[\![\varphi]\!](\delta)$, la valeur de $[\![\varphi]\!]$ pour δ.

Definition 3 modèle d'une formule propositionnelle

Une valuation δ est un modèle pour φ, si $[\![\varphi]\!](\delta) = 1$, que l'on notera aussi sous la forme "$\delta \models \varphi$".

Soit φ une formule. $\models \varphi$ dénotera le fait que φ est une tautologie c'est-à-dire que toutes les valuations possibles sont modèles de cette formule. Soient deux ensembles de formules Σ_1 et Σ_2 telles que $\Sigma_1 \subseteq \Sigma_2$. Toute valuation modèle des formules de Σ_2 est un modèle des formules de Σ_1.

1.4 Raisonnement en logique propositionnelle

On applique des transformations sur les expressions logiques de la forme $A \Rightarrow B$ mais on applique des transformations qui conserve la sémantique des formules. En fait, on peut considérer que cette formule à un contexte, c'est-à-dire des formules propositionnelles qui sont des hypothèses. On notera cela sous la forme $\Gamma \rightarrow A \Rightarrow B$ et plus généralement on manipule des expressions de la forme: $\Gamma \rightarrow \Delta$ avec

- $\Gamma = \{\phi_1, \ldots, \phi_n\}$
- $\Delta = \{\psi_1, \ldots, \psi_p\}$
- $\phi_1 \wedge \ldots \wedge \phi_n \Rightarrow \psi_1 \vee \ldots \vee \psi_p$
- La partie gauche constitue les hypothèses supposées vraies et la partie droite les conclusions possibles mais pas nécessairement toutes les conclusions.

Si l'ensemble des formules de Γ est $\{A_1, \ldots, A_n\}$ et si l'ensemble des formules de Δ est $\{B_1, \ldots, B_p\}$, alors on écrira $\Gamma \rightarrow \Delta$ sous la forme $A_1, \ldots, A_n \rightarrow B_1, \ldots, B_p$.

L'interprétation d'une telle forme est la suivante:*Si les formules $A_1, \ldots, A_n$ sont toutes vraies, alors l'une au moins des formules $B_1, \ldots, B_p$ est vraie.*. Pour simplifier certaines écritures, on écrira $\Gamma, A \rightarrow \Delta$ à la place de $\Gamma \cup \{A\} \rightarrow \Delta$.

Definition 4

Soit une valuation δ de $\mathcal{P}$ dans $\mathcal{B}$ associant à tout symbole de $\mathcal{P}$ une valeur booléenne de $\{0, 1\}$. On étend l'interprétation de cette valuation comme suit:

si $]\!]A[\![(\delta) = 0$ pour une formule de Γ, ou si $]\!]B[\![(\delta) = 1$ pour une formule B de Δ, $[\![\Gamma \rightarrow \Delta]\!](\delta) = 1$ et vaut 0 sinon.

On remarque que les deux formes suivantes sont équivalentes:

- $\Gamma \rightarrow A \Rightarrow B$
- $\Gamma, A \rightarrow B$

On remarque aussi que les formes suivantes sont valides quelle que soit la valuation des variables propositionnelles:

- $P \to P$
- $\to P \Rightarrow P$
- $FALSE \to$
- $\to TRUE$
- $\to P \wedge Q \Rightarrow P$ et $\to P \wedge Q \Rightarrow Q$

On peut donc déduire une liste de formes qui sont valides pour toute valuation.

Propriété 1 *()*

- Pour toute formule A, $A \to A$ est une forme valide. ou axiome..
- $false \to$ est une forme valide ou axiome.
- $\to true$ est une forme valide ou axiome.

Les formes $\Gamma \to \Delta$ sont transformées avec des transformations qui doivent être correctes.

Règles structurelles

- Si Γ_1, Γ_2, Δ_1, Δ_2 sont des ensembles finis de formules tels que $\Gamma_1 \subseteq \Gamma_2$, $\Delta_1 \subseteq \Delta_2$, alors

$$\frac{\Gamma_1 \to \Delta_1}{\Gamma_2 \to \Delta_2}$$

- atténuations:
  - $\dfrac{\Gamma \to \Delta}{\phi, \Gamma \to \Delta}$
  - $\dfrac{\Gamma \to \Delta}{\Gamma \to \Delta, \psi}$
- contraction:
  - $\dfrac{\phi, \phi, \Gamma \to \Delta}{\phi, \Gamma \to \Delta}$
  - $\dfrac{\Gamma \to \Delta, \psi, \psi}{\Gamma \to \Delta, \psi}$
- permutation:
  - $\dfrac{\Gamma_1, \phi, \psi, \Gamma_2 \to \Delta}{\Gamma_1, \psi, \phi, \Gamma_2 \to \Delta}$
  - $\dfrac{\Gamma \to \Delta_1, \phi, \psi, \Delta_2}{\Gamma \to \Delta_1, \psi, \phi, \Delta_2}$
- **Coupure**

$$\frac{\Gamma \to \Delta, \phi, \quad \phi, \Lambda \to \Pi}{\Gamma, \Lambda \to \Delta, \Pi}$$

Règles opératoires

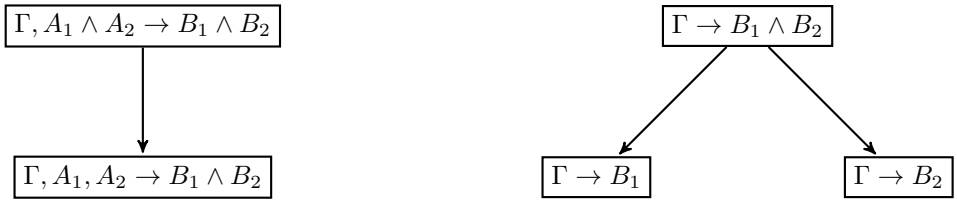
- **Négation**

$$\frac{\Gamma \to \Delta, B}{\Gamma, \neg B \to \Delta} \qquad \frac{\Gamma, A \to \Delta}{\Gamma \to \Delta, \neg A}$$

- **Conjonction**

$$\frac{\Gamma, A, B \to \Delta}{\Gamma, A \wedge B \to \Delta} \qquad \frac{\Gamma \to \Delta, A \;\; \Gamma \to \Delta, B}{\Gamma \to \Delta, A \wedge B}$$

Les deux règles peuvent être appliquées et leur application conduit à un arbre dont les feuilles doivent être valides.



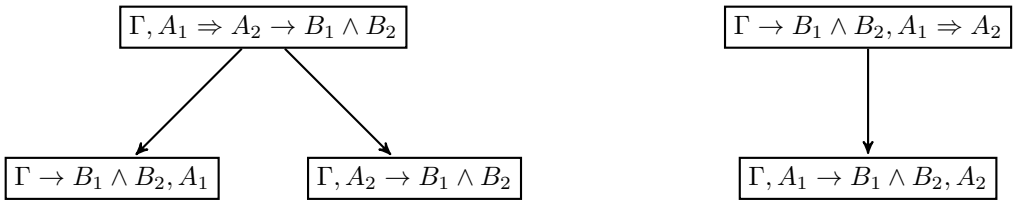
- **Disjonction**

$$\frac{\Gamma, A \to \Delta \quad \Gamma, B \to \Delta}{\Gamma, A \vee B \to \Delta} \qquad \frac{\Gamma \to \Delta, A, B}{\Gamma \to \Delta, A \vee B}$$

- **Implication**

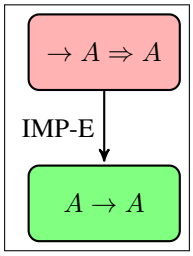
$$\frac{\Gamma \to \Delta, A \quad \Gamma, B \to \Delta}{\Gamma, A \Rightarrow B \to \Delta} \qquad \frac{\Gamma, A \to \Delta, B}{\Gamma \to \Delta, A \Rightarrow B}$$

L'opérateur d'implication dispose de deux règles qui suppriment à gauche ou à droite les formes avec l'implication.



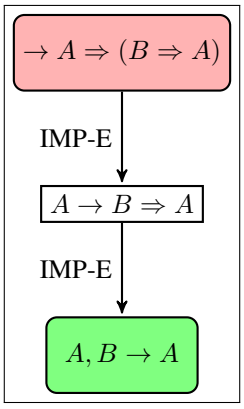
Exemple 3

Dans l'arbre construit ci-dessous, le nœud Ã©tiquettÃ© $A \to A$ vert est un axiome et une feuille. On en dÃ©duit que $\to A \Rightarrow A$ est dÃ©rivable puisque toutes les feuilles sont vertes.



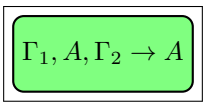
Exemple 4

Un autre exemple est la dÃ©rivation de la tautologie $A \Rightarrow (B \Rightarrow A)$.



Exemple 5

Un autre exemple est l'identification d'un axiome de cette forme $\Gamma_1, A, \Gamma_2 \rightarrow A$.



Definition 5 (Transformation correcte)

Soit la classe des formules propositionnelles sur $\mathcal{C} \cup \mathcal{P}$. On dit qu'une transformation T associant à toute formule de la classe des formules propositionnelles sur $\mathcal{C} \cup \mathcal{P}$, une formule de la classe des formules propositionnelles sur $\mathcal{C} \cup \mathcal{P}$ est correcte, si pour toute formule propositionnelle φ de la classe des formules propositionnelles sur $\mathcal{C} \cup \mathcal{P}$, $[\![T(\varphi)]\!] = [\![\varphi]\!]$.

Voici quelques règles de transformation:

- TSimp1 $TSimp1([A \wedge (A \Rightarrow B)]) = [A \wedge B]$
  $[\![[A \wedge (A \Rightarrow B)]]\!] = a.(\overline{a} + b) = 0 + a.b = [\![[A \wedge B]]\!]$
- TSimp2 $TSimp2([A \wedge (B = C) \wedge D \Rightarrow E \wedge (B = F) \wedge G]) = [A \wedge (B = C) \wedge D \Rightarrow E \wedge (C = F) \wedge G]$
- TSimp3 $TSimp3([A \wedge (B = C) \wedge D \Rightarrow E \wedge (F = F) \wedge G] == [A \wedge (B = C) \wedge D \Rightarrow E \wedge TRUE \wedge G]$
- TSimp4 $TSimp4([A \Rightarrow B \wedge TRUE \wedge C]) = [A \Rightarrow B \wedge C]$
- TSimp5 $TSimp5([A \wedge (B = C \Rightarrow U) \wedge (B = D \wedge B = C \Rightarrow V) wedge C \neq D \wedge E]) = [A \wedge B = C \wedge U \wedge C \neq D \wedge E]$
- Permutation $TPermut([E \wedge A \wedge B \wedge D]) = [E \wedge B \wedge A \wedge D]$
  $[\![[E \wedge A \wedge B \wedge D]]\!] = e.a.b.d == e.b.a.d = [\![[E \wedge B \wedge A \wedge D]]\!]$

On montre que ces tranformations sont correctes en montrant que la transformation préserve les modèles propositionnels.

Si on considère une propriété de la forme $A \Rightarrow B$ dans le contexte Γ, on est amené à transformer la forme suivante: $\Gamma \rightarrow A \Rightarrow B$. Les transformations induites par les règles ci-dessus peuvent permettre de produire des formes comme:

$$\Gamma \rightarrow A \Rightarrow B \tag{6}$$

Cette forme est transformée en:

$$\Gamma, A \rightarrow B \tag{7}$$

Nous considérons plusieurs cas d'analyse:

- B est TRUE: dans ce cas, $A \Rightarrow B$ est valide ou correct.
- A est FALSE: dans ce cas, $A \Rightarrow B$ est valide ou correct.
- A est TRUE et B est FALSE: dans ce cas, $A \Rightarrow B$ n'est pas valide ou incorrect.

Les trois cas considérés devraient suffire dans un premier temps pour analyser les conditions de vérificaton de type propositionnel. Nous avons évoqué la possibilité d'utiliser des abstractions propositionnelles. Par exemple, l'équation **??** peut être vue comme l'équation **??** où A et l'abstraction de $x = 1$ et B est celle de $y = 2$:

$$\rightarrow x = 1 \wedge y = 2 \Rightarrow x = 1 \tag{8}$$

$$x = 1 \wedge y = 2 \rightarrow x = 1 \tag{9}$$

$$x = 1, y = 2 \rightarrow x = 1 \tag{10}$$

1.5 Syntaxe de formules du premier ordre

Nous allons introduire la notion de variable d'individu et la notion de quantificateurs. Les classes suivantes sont définies :

1. Symboles logiques : $\sim, \Rightarrow, \forall, \wedge, \vee, \equiv, \exists, \ldots$
2. Symboles de variables : $\mathcal{V} = \{x, y, z, \ldots\}$
3. Symboles de constantes : $\mathcal{C} = \{a, b, c, \ldots\}$
4. Symboles de fonctions : $\mathcal{F} = \{f, g, h, \ldots\}$
5. Symboles de relations: $\mathcal{R} = \{P, Q, R, \ldots\}$

Exemple 6

1. $\forall x(\exists y\ (P(x, y) \wedge R(x, y))\ \Rightarrow\ Q(z, x))$
2. $\forall x((>(x, 0))\ \Rightarrow \exists y\ (>(y, 0)\ \Rightarrow\ (-(x, y) < 0)))$

Remarque

la quantification porte sur les variables de constantes. Il n'y a pas de quantification sur les fonctions.

Tout triplet $\mathcal{L} = (\mathcal{C}; \mathcal{R}; \mathcal{F})$ est appelé un langage du premier ordre. Deux catégories d'objets sont à définir:

$\longrightarrow$ les termes de $\mathcal{L}$

$\longrightarrow$ les formules de $\mathcal{L}$

Les termes de $\mathcal{L}$, notés $\mathcal{T}[\mathcal{L}]$, sont les éléments de la $\mathcal{F} \cup \mathcal{C}$-algèbre $M(\mathcal{F}, \mathcal{V})$.
Les formules de $\mathcal{L}$ sont obtenues à partir des formules atomiques de $\mathcal{L}$, noté $\mathcal{A}[\mathcal{L}]$:
$\mathcal{A}[\mathcal{L}] = \{R(t_1, \ldots, t_n)/t_1, \ldots, t_n \in M(\mathcal{F}, \mathcal{V})\}$ et $R \in \mathcal{R}\}$
Les formules atomiques permettent de construire les formules de $\mathcal{L}$. Il faut noter qu'il y a une formule particulière atomique construite à l'aide de la relation d'égalité; dans ce cas, on parle de calcul des prédicats avec égalité.

Une formule de f de $\mathcal{L}$, ou un élément de $\mathcal{F}(\mathcal{L})$, est :

1. une formule atomique de $\mathcal{A}[\mathcal{L}]$
2. la négation d'une formule : $\sim f$ où $f \in \mathcal{F}(\mathcal{L})$
3. la forme implicative $f_1 \Rightarrow f_2$ où $f_1, f_2 \in \mathcal{F}(\mathcal{L})$
4. la forme quantifiée $\forall x(f)$ où $x \in \mathcal{V}$ et $f \in \mathcal{F}(\mathcal{L})$
5. le résultat de l'application finie des règles 1,2,3,4 et uniquement celles-ci.

L'occurence d'une variable x d'une formule f est liée, si elle se trouve dans la partie d'un quantificateur $\forall x$, sinon on dit qu'elle est libre.
Exemple 7
$\forall x(\exists y(P(f(x, y)) \wedge Q(x, z))) \wedge R(x, z)$

- les occurences de z sont libres
- les 2 premières occurences de x sont liées
- la dernière occurence de x est libre
- les occurences de y sont liées

Une variable peut avoir des occurences libres et des occurences liées. Une variable est libre, s'il y a au moins une occurence libre de celle-ci dans la formule considérée. On note $\mathcal{V}(\varphi)$, l'ensemble des variables libres de φ.

Soit x une variable, t un terme et φ une formule. On dit que t est libre pour x dans φ, quand aucune occurence libre de x dans φ n'est dans le champ d'un quantificateur liant une variable de t.

Exemple 8

$\varphi \ = \ \forall y \ (\exists z \ (R(x, y, z)))$

(1) $t = f(x, u)$: t est libre pour x dans φ
(2) $t = f(x, y)$: t n'est pas libre pour x dans φ car y est lié par $\forall$.

Une formule φ est ouverte, s'il y a au moins une occurence libre d'une variable. Sinon elle est close. Nous noterons $\varphi(x_1, x_2, \ldots, x_n)$ une formule ouverte avec $x_1, x_2, \ldots, x_n$ les variables ayant des occurences libres. Une formule libre peut être close par quantification $\forall$:

$\forall x_1(\forall x_2(\ldots(\forall x_n(\varphi(x_1, \ldots, x_n))\ldots))$, c'est la cloture universelle de φ. La formule φ peut être close par $\exists$, c'est la cloture existencielle.

Une formule φ est dite sous forme prénexe, quand elle s'écrit
$Q_1x_1(Q_2x_2(\ldots(Q_nx_n(\varphi(x_1, \ldots, x_n, y_1, \ldots, y_p))$ où $Q_i \in \{\forall, \exists\}$ et $\varphi(x_1, \ldots, y_p)$ sans quantificateur.

Soit t un terme, φ une formule et x une variable de φ. $\varphi[t/x]$ ou $\varphi_x[t]$ est la formule obtenue en remplaçant les occurences libres de x par t où t est libre pour x dans φ. Avant de substituer une variable par un terme, on prendra soin de vérifier cette condition d'application.

Exemple 9

Soit $\varphi = \ \exists y.(x = 2y)$

- $\varphi_x[y + 1] \ = \ \exists y. \ (y + 1 = 2y)$ n'a pas de sens car $y + 1$ n'est pas libre pour y dans ϕ.
- $\varphi_x[z + 5] \ = \ \exists y. \ (z + 5 = 2y)$.

La substitution est donc à manipuler avec soin !

1.6 Interprétation des formules du calcul des prédicats du premier ordre

Une interprétation $\mathcal{I}$ de $\mathcal{L}$ ou une réalisation de $\mathcal{L}$ est la donnée :

1. d'un ensemble non-vide D, appelé domaine de $\mathcal{I}$.
2. pour chaque symbole de fonction f de $\mathcal{F}$, d'une fonction ou application $f_{\mathcal{I}} : |\mathcal{I}|^n \longrightarrow |\mathcal{I}|$.
3. pour chaque symbole de relation R de $\mathcal{R}$, d'une relation $R_{\mathcal{I}}$ sur D.

Une interprétation $\mathcal{I}$ de $\mathcal{L}$ est une $\mathcal{F}$-algèbre où les relations sont des fonctions à valeurs dans les booléens BOOL. Le symbole d'égalité est interprété par l'égalité. Les constantes seront interprétées par des éléments de D. Le langage étendu pour une interprétation sera obtenu en ajoutant des symboles de constantes pour les valeurs des éléments de D. Soit une formule close φ et $\mathcal{I}$ une interprétation.

1. φ est atomique:
   φ s'écrit $R(t_1, \ldots, t_n)$ où $R \in \mathcal{R}$ et $t_1 \ldots t_n$ des termes clos (sans variables).
   $\mathcal{I} \models \varphi$ ssi $(t_{1\mathcal{I}}, \ldots, t_{n\mathcal{I}}) \in R_{\mathcal{I}}$
2. φ est $\sim \Psi$:
   $\mathcal{I} \models \varphi$ ssi $non(\mathcal{I} \models \Psi)$
3. φ est $\alpha \Rightarrow \beta$:
   $\mathcal{I} \models \varphi$ ssi $(\mathcal{I} \models \beta \ ou \ \mathcal{I} \models\sim \alpha)$
4. φ est $\forall x(\psi(x))$:
   $\mathcal{I} \models \varphi$ ssi , pour tout i de D, $\mathcal{I} \models \psi(\bar{i}/x)$.

Le point (4) suppose que les quantificateurs lient des variables libres ayant des occurences sinon l'interprétation est lue même celle sans le quantificateur. "$\mathcal{I} \models \varphi$" se lit "I satisfait φ"

Soit φ une formule ouverte. Soit δ une application de $\mathcal{V}$ dans D est une valuation.
On dit qu'une formule ouverte $\varphi(x_1, \ldots, x_n)$ est satisfaite dans une interprétation $\mathcal{I}$, si elle est satisfaite pour toutes les valuations possibles :

$\mathcal{I} \models \varphi(x_1 \ldots x_n)$ ssi $\mathcal{I} \models \forall x_1(\forall x_2 \ldots (\forall x_n \varphi(x_1 \ldots x_n)) \ldots)$.

On note $\mathcal{I}, \delta \models \varphi(x_1 \ldots x_n)$ la satisfaction de φ dans $\mathcal{I}$ pour δ. Cela veut aussi signifier que les occurences de x_i sont substitués par $\delta(x_i)$ dans la, formule $\varphi(x_1 \ldots x_n)$.

Propriété 2 Soit φ une formule close.

1. φ est satisfaite dans $\mathcal{I}$ ssi $\sim \varphi$ n'est pas satisfaite dans $\mathcal{I}$.
2. Soit $\mathcal{I}$ une interprétation. φ est soit satisfaite, soit non satisfaite.
3. Si $\mathcal{I}$ satisfait φ et $\varphi \Rightarrow \psi$, alors $\mathcal{I}$ satisfait ψ.

Une formule φ est satisfaisable, s'il existe une interprétation $\mathcal{I}$ telle que $\mathcal{I} \models \varphi$.
Une formule φ est valide, si elle est satisfaite dans toutes les interprétations : on note $\models \varphi$.
Deux formules φ et ψ sont équivalentes, si, pour toute interprétation $\mathcal{I}$, $\mathcal{I} \models \varphi$ ssi $\mathcal{I} \models \psi$.

Exemple 10

1. $\forall x \varphi(x, x_1, \ldots, x_n)$ équivaut à $\forall y\ \varphi(y, x_1, \ldots, x_n)$.
2. $\forall x\ \varphi(x)$ équivaut à $\sim \exists x \sim \varphi(x)$

Une interprétation $\mathcal{I}$ satisfaisant une formule φ est un modèle de φ. Une formule φ qui admet un modèle est non contradictoire. Une formule valide est une thèse.
Soit Σ un ensemble de formules. Un modèle pour Σ est une interprétation $\mathcal{I}$ satisfaisant chaque formule de Σ :

1. Soit φ et ψ deux formules closes. ψ se déduit sémantiquement de φ, si tout modèle de φ est un modèle de ψ : $\varphi \models \psi$.
2. Soit Σ un ensemble de formules closes et ψ une formule close. $\Sigma \models \psi$, si tout modèle de Σ est un modèle de ψ.

► **Théorème 2** Théorème de la déduction sémantique

Soient φ, ψ des formules closes. Soit Σ un ensemble de formules closes.

1. $\varphi \models \psi$ ssi $\models \varphi \Rightarrow \psi$
2. $\Sigma \cup \{\varphi\} \models \psi$ ssi $\Sigma \models \varphi \Rightarrow \psi$

Un ensemble Σ de formules closes est consistant, si Σ a un modèle.
Proposition 1

Soit Σ ensemble de formules closes et φ closes. $non(\Sigma \models \varphi)$ ssi $\Sigma \cup \{\sim \varphi\}$ est consistant.

Preuve $\Sigma \cup \sim \varphi$ consistant ssi il existe $\mathcal{I}$ modèle de Σ et de $\sim \varphi$ ssi il existe $\mathcal{I}$ modèle de Σ et non modèle de Σ ssi $non(\Sigma \models \varphi)$. *fin de la preuve*

Cette présentation est empruntée aux logiciens et on peut en donner une version qui se rapproche d'une vue plus proche de la sémantique des langages de programmation en proposant une reformulation assez simple de la sémantique des formules.

1.7 Sémantique des formules

Dans la sous-section précédente, nous avons défini les points suivants:

- $\mathcal{L} = (\mathcal{C}; \mathcal{R}; \mathcal{F})$ est un langage du premier ordre où
  - $\mathcal{C}$ désigne l'ensemble des symboles de constantes.
  - $\mathcal{R}$ désigne l'ensemble des des symboles de relations
  - $\mathcal{F}$ désigne l'ensemble des des symboles de fonctions
- $\mathcal{E}(\mathcal{L})$ désigne les énoncés ou les formules construites pour le langage $\mathcal{L}$.
- $\mathcal{V}$ désigne l'ensemble des symboles de variables.
- $\mathcal{O}$ désigne l'ensemble des symboles d'opérateurs logiques ($\wedge$, $\vee$, $\neg$, $\Rightarrow$, ...).
- $\mathcal{I}$ désigne une interprétation pour les formules $\mathcal{E}(\mathcal{L})$ définie par une paire $(D, [\![\bullet]\!])$ où $[\![\bullet]\!]$ est un opérateur associant à toute formule un élément défini sur le domaine D.
- $\mathcal{C}$ désigne l'ensemble des symboles de constantes.
- $\mathcal{S}$ désigne l'ensemble des valuations c'est-à-dire $\mathcal{V} \longrightarrow D$.

L'opérateur $[\![\bullet]\!]$ est défini par induction sur la structure des termes sur $\mathcal{L}$ et sur la structure des fomules de $\mathcal{E}(\mathcal{L})$. La défintion est donnée selon les différents cas syntaxiques des termes et des formules; on note $s \in \mathcal{S}$ une valuation quelconque et $BOOL = \{ff, tt\}$:

- $[\![c]\!](s) \in D$ où c est un symbole de constantes ($c \in \mathcal{C}$).
- $[\![x]\!](s) = s(x)$ où x est un symbole de variables ($x \in \mathcal{V}$).
- $[\![f(t_1, \ldots, t_n)]\!](s) = f_{[\![I]\!]}([\![t_1]\!], \ldots, [\![t_n]\!])(s)$ où $f_{[\![I]\!]}$ est l'interprétation de f pour l'interprétation $[\![I]\!]$ et $f_{[\![I]\!]} \in D^n \longrightarrow D$.
- $[\![R(t_1, \ldots, t_n)]\!](s) = R_{[\![I]\!]}([\![t_1]\!], \ldots, [\![t_n]\!])(s) \in BOOL$ où $R_{[\![I]\!]}$ est l'interprétation de R pour l'interprétation $[\![I]\!]$ et $R_{[\![I]\!]} \in D^n \longrightarrow BOOL$.
- $[\![TRUE]\!](s) = ttet$[FALSE](s)=ff
- $[\![\varphi_1 \; op \; \varphi_2]\!](s) = op_{[\![I]\!]}([\![\varphi_1]\!](s), [\![\varphi_2]\!](s))$ où op est un symbole d' opérateur de $\mathcal{O}$ et leur interprétation est canonique pour les connecteurs logiques usuels.
- $[\![\forall x.\varphi]\!](s) = tt$, si pour tout valeur d de D, $[\![\varphi]\!](s[x \mapsto d]) = tt$.
- $[\![\exists x.\varphi]\!](s) = tt$, si pour une valeur d de D, $[\![\varphi]\!](s[x \mapsto d]) = tt$.

Nous avons utilisé la notation $s[x \mapsto d]$ pour désigner la fonction identique à s sauf en x où elle vaut d. Enfin,

Dans notre section précédente, nous avons défini la notation suivante $\mathcal{I}, \delta \models \varphi(x_1 \ldots x_n)$ et la relation avec la définition de $[\![\bullet]\!]$ est la suivante: $\mathcal{I}, \delta \models \varphi$ si, et seulement, $[\![\varphi]\!](\delta) = tt$.

Avant de poursuivre, il est important de noter que la notion de variable logique et celle de variable informatique sont différentes. En effet, une variable logique désigne un emplacement dans une formule pour désigner une valeur constante. Pour une variable informatique, une variable informatique a un nom qui identifie une mémoire dont le contenu peut changer. Le lien entre les deux notions existent, puisque nous pouvons utiliser une variable logique pour représenter la valeur de la variable informatique à un instant donné. Nous pouvons donc utiliser les formules de la logique du premier ordre pour caractériser ce qui est vrai à un instant donné et aussi utiliser les variables logiques pour représenter les valeurs des variables informatiques. On définira des conventions de nommage des variables que nous utiliserons.