Systèmes de preuve

Sylvain Schmitz

SYSTÈMES DE PREUVE
NOTES DE RÉVISION POUR L'AGRÉGATION

SYLVAIN SCHMITZ

ENS Paris-Saclay, France

Résumé. Ces notes de révision sont consacrées à la leçon **918** « Systèmes formels de preuve en logique du premier ordre. Exemples. » de l'agrégation de mathématiques, option D, session 2019.

Contenu des notes

Le programme officiel 2019 correspondant à cette leçon est [1] :

> *Logique du premier ordre : langages, termes, formules. Variables libres et variables liées, substitutions, capture de variables. Interprétation d'une formule dans un modèle. Validité, satisfiabilité. Systèmes formels de preuve. Calcul des séquents, déduction naturelle. Algorithme d'unification des termes. Preuves par résolution.*

Le rapport du jury 2018 pour cette leçon en précise le contenu attendu [2] :

> *Le jury attend du candidat qu'il présente au moins la déduction naturelle ou un calcul de séquents et qu'il soit capable de développer des preuves dans ce système sur des exemples classiques simples. La présentation des liens entre syntaxe et sémantique, en développant en particulier les questions de correction et complétude, et de l'apport des systèmes de preuves pour l'automatisation des preuves est également attendue.*
>
> *Le jury appréciera naturellement si des candidats présentent des notions plus élaborées comme la stratégie d'élimination des coupures mais est bien conscient que la maîtrise de leurs subtilités va au-delà du programme.*

Les systèmes de preuve permettent d'établir la validité de formules logiques par des moyens purement syntaxiques, en construisant des preuves finies sans faire appel à des raisonnements sur les modèles. Ils se prêtent donc à des procédures de *recherche de preuve*, permettant d'établir cette validité de manière automatique pour de nombreuses instances pratiques – bien que le problème soit indécidable, comme vu dans la leçon **924**.

Ces notes de révision se concentrent sur deux types de systèmes de preuve : le calcul des séquents et la résolution ; les systèmes de preuve à la Hilbert, de déduction naturelle, et les tableaux ne sont pas traités.

💬 *L'existence de tels systèmes de preuve signifie que l'ensemble des formules valides est récursivement énumérable, et donc que l'ensemble des formules satisfiables n'est pas récursivement énumérable – un exemple intéressant pour l'ancienne leçon **922** « Ensembles récursifs, récursivement énumérables. Exemples ».*

1. http://media.devenirenseignant.gouv.fr/file/agregation_externe/13/1/p2019_agreg_ext_maths_929131.pdf#section.16
2. http://agreg.org/Rapports/rapport2018.pdf#subsection.5.4.3

L'objectif de ces notes de révision est de fournir une référence avec des notations unifiées et des pointeurs vers différents ouvrages susceptibles d'être utilisés lors de la préparation au concours de l'agrégation. Ces notes ne sont pas un substitut à l'étude approfondie d'ouvrages et d'articles sur le sujet, référencés par le symbole « 🔖 » dans les notes en marge ; les deux principaux ouvrages sur lesquels je m'appuie sont :

René David, Karim Nour et Christophe Raffalli, 2003. *Introduction à la logique*. Dunod, Paris, 2e édition.

Jean Goubault-Larrecq et Ian Mackie, 1997. *Proof Theory and Automated Deduction*, volume 6 de *Applied Logic Series*. Kluwer Academic Publishers, Amsterdam.

Une partie de ces notes est inspirée des notes de cours du « MOOC » *Introduction à la logique informatique* par David Baelde, Hubert Comon et Étienne Lozes [3]. Je remercie David Baelde et Hubert Comon pour leur aide précieuse.

3. Voir les pages `https://www.fun-mooc.fr/courses/ENSCachan/20004S02/session02/about` et `https://www.fun-mooc.fr/courses/ENSCachan/20009/session01/about`

Partie 1. Logique classique du premier ordre

Commençons, afin de fixer les notations employées dans ces notes, par rappeler la syntaxe et la sémantique de la logique du premier ordre (aussi appelée « calcul des prédicats »). Outre les connecteurs booléens habituels en logique ($\wedge$, $\vee$, $\neg$, $\Rightarrow$, ...) et les quantificateurs ($\forall$, $\exists$), la logique du premier ordre permet d'exprimer des propriétés de symboles auxiliaires associés à ses modèles.

On définit pour cela une *signature* du premier ordre (aussi appelée un « langage du premier ordre ») comme une paire $(\mathcal{F}, \mathcal{P})$ formée de deux ensembles dénombrables $\mathcal{F}$ de symboles de fonction et $\mathcal{P} \neq \emptyset$ de symboles de relation (aussi appelés « symboles de prédicat »), avec $\mathcal{F} \cap \mathcal{P} = \emptyset$, et tous deux munis d'une fonction d'*arité* $r \colon \mathcal{F} \cup \mathcal{P} \to \mathbb{N}$. On note $\mathcal{F}_n \stackrel{\text{def}}{=} \{f \in \mathcal{F} \mid r(f) = n\}$ et $\mathcal{P}_n \stackrel{\text{def}}{=} \{R \in \mathcal{P} \mid r(R) = n\}$ leurs restrictions aux symboles d'arité n. Une fonction d'arité zéro est appelée une *constante* ; une relation d'arité zéro est appelée une *proposition*.

⚠ *Le contenu de cette partie peut se retrouver au moins partiellement dans un plan de leçon, mais ne saurait servir en développement, puisqu'il n'y est pas question de systèmes de preuve.*

🔖 *[David et al., 2003, sec. 1.2], [Goubault-Larrecq et Mackie, 1997, sec. 6.1].*

💬 *On peut aussi travailler avec des signatures non dénombrables, mais les preuves nécessitent alors d'utiliser des notions de théorie des ensembles comme le lemme de Zorn.*

1. Syntaxe

Soit X un ensemble infini dénombrable de symboles de variables. Les formules de la logique du premier ordre sur une signature $(\mathcal{F}, \mathcal{P})$ sont définies par la syntaxe abstraite

$$\begin{aligned} t &::= x \mid f(t_1, \ldots, t_m) && \text{(termes)} \\ \alpha &::= R(t_1, \ldots, t_m) && \text{(formules atomiques)} \\ \varphi &::= \alpha \mid \neg\varphi \mid \varphi \vee \varphi \mid \exists x.\varphi && \text{(formules)} \end{aligned}$$

où $x \in X$, $m \in \mathbb{N}$, $f \in \mathcal{F}_m$ et $R \in \mathcal{P}_m$. L'ensemble des termes sur X et $\mathcal{F}$ est aussi dénoté $T(\mathcal{F}, X)$. Un *littéral* est une formule atomique α ou sa négation $\neg\alpha$.

💬 *La* logique propositionnelle *(aussi appelée « calcul des propositions ») de la leçon **916** est obtenue en posant $\mathcal{F} = \emptyset$ et $\mathcal{P} = \mathcal{P}_0$, et en interdisant la quantification dans la syntaxe des formules ; la syntaxe abstraite résultante est $\varphi ::= P \mid \neg\varphi \mid \varphi \vee \varphi$ où $P \in \mathcal{P}_0$.*

On peut ajouter d'autres symboles booléens à cette syntaxe minimale (ou choisir une autre syntaxe minimale complète). Alternativement, ces symboles peuvent être définis dans la logique :

$$\varphi \wedge \psi \stackrel{\text{def}}{=} \neg(\neg\varphi \vee \neg\psi) \qquad \forall x.\varphi \stackrel{\text{def}}{=} \neg\exists x.\neg\varphi \qquad \varphi \Rightarrow \psi \stackrel{\text{def}}{=} \neg\varphi \vee \psi$$
$$\top \stackrel{\text{def}}{=} \forall x.R(x, \ldots, x) \vee \neg R(x, \ldots, x) \qquad \bot \stackrel{\text{def}}{=} \neg\top$$

où R est un symbole arbitraire de $\mathcal{P}$ (qui est supposé non vide).

Remarquons qu'en informatique, les formules sont couramment représentées par des graphes acycliques dirigés plutôt que par des arbres, ce qui évite la duplication de sous-arbres identiques ; par exemple, la figure 1 décrit la représentation mémoire de la formule du buveur $\exists x.(B(x) \Rightarrow \forall y.B(y))$.

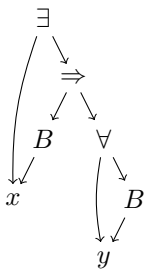

Figure 1. Le graphe dirigé acyclique de la formule du buveur $\exists x.(B(x) \Rightarrow \forall y.B(y))$.

1.1. **Variables libres et variables liées.** L'ensemble $\mathrm{fv}(t)$ des *variables libres* (« *free variables* » en anglais) d'un terme t est défini inductivement par

$$\mathrm{fv}(x) \stackrel{\text{def}}{=} \{x\} , \qquad \mathrm{fv}(f(t_1,\dots,t_m)) \stackrel{\text{def}}{=} \bigcup_{1\le i\le m} \mathrm{fv}(t_i) .$$

Un terme t sans variable libre (*i.e.* $\mathrm{fv}(t)=\emptyset$) est dit *clos* ; l'ensemble des termes clos est noté $T(\mathcal{F})$.

Les ensembles $\mathrm{fv}(\varphi)$ des *variables libres* et $\mathrm{bv}(\varphi)$ des *variables liées* (« *bound variables* » en anglais) d'une formule φ sont définis inductivement par

$$\begin{aligned}
\mathrm{fv}(R(t_1,\dots,t_m)) &\stackrel{\text{def}}{=} \bigcup_{1\le i\le m} \mathrm{fv}(t_i) , & \mathrm{bv}(R(t_1,\dots,t_m)) &\stackrel{\text{def}}{=} \emptyset ,\\
\mathrm{fv}(\neg\varphi) &\stackrel{\text{def}}{=} \mathrm{fv}(\varphi) , & \mathrm{bv}(\neg\varphi) &\stackrel{\text{def}}{=} \mathrm{bv}(\varphi)\\
\mathrm{fv}(\varphi\vee\psi) &\stackrel{\text{def}}{=} \mathrm{fv}(\varphi)\cup\mathrm{fv}(\psi) , & \mathrm{bv}(\varphi\vee\psi) &\stackrel{\text{def}}{=} \mathrm{bv}(\varphi)\cup\mathrm{bv}(\psi) ,\\
\mathrm{fv}(\exists x.\varphi) &\stackrel{\text{def}}{=} \mathrm{fv}(\varphi)\setminus\{x\} & \mathrm{bv}(\exists x.\varphi) &\stackrel{\text{def}}{=} \{x\}\cup\mathrm{bv}(\varphi) .
\end{aligned}$$

Une formule φ sans variable libre (*i.e.* $\mathrm{fv}(\varphi)=\emptyset$) est dite *close*.

2. Sémantique

[David et al., 2003, sec. 2.2], [Goubault-Larrecq et Mackie, 1997, sec. 6.2].

Fixons $(\mathcal{F},\mathcal{P})$ une signature du premier ordre. On note $\mathbb{B} \stackrel{\text{def}}{=} \{\bot,\top\}$ pour l'ensemble des *valeurs de vérité* (aussi appelé « algèbre de Boole »), muni des opérations $\neg\colon\mathbb{B}\to\mathbb{B}$ et $\wedge,\vee\colon\mathbb{B}^2\to\mathbb{B}$ définies par $\neg\top=\bot\vee\bot=\bot\wedge\bot=\top\wedge\bot=\bot\wedge\top=\bot$ et $\neg\bot=\top\vee\top=\top\vee\bot=\bot\vee\top=\top\wedge\top=\top$.

L'ensemble des formules valides change si l'on permet un domaine vide.

Interprétations. Une *interprétation* I (aussi appelée une « $(\mathcal{F},\mathcal{P})$-structure ») est constituée d'un ensemble $D_I\neq\emptyset$, appelé son domaine ou son support, d'une fonction $f^I\colon D_I^{r(f)}\to D_I$ pour chaque $f\in\mathcal{F}$, et d'une relation $R^I\colon D_I^{r(R)}\to\mathbb{B}$ pour chaque $R\in\mathcal{P}$. Une *valuation* dans I est une fonction $\rho\colon X\to D_I$. On notera $\rho[e/x]$ pour la valuation qui associe e à x et $\rho(y)$ à $y\neq x$.

*C'est une sémantique dénotationnelle au sens de la leçon **930**.*

Satisfiabilité. Pour un terme t, sa sémantique $[\![t]\!]^I_\rho$ dans une interprétation I pour une valuation ρ est un élément de D_I défini inductivement par

$$[\![x]\!]^I_\rho \stackrel{\text{def}}{=} \rho(x) , \qquad [\![f(t_1,\dots,t_m)]\!] \stackrel{\text{def}}{=} f^I([\![t_1]\!]^I_\rho,\dots,[\![t_m]\!]^I_\rho)$$

pour tout m et tout $f\in\mathcal{F}_m$.

La sémantique $[\![\varphi]\!]^I_\rho$ d'une formule dans une interprétation I pour une valuation ρ est une valeur de vérité dans $\mathbb{B}$ définie inductivement par

$$\begin{aligned}
[\![R(t_1,\dots,t_m)]\!]^I_\rho &\stackrel{\text{def}}{=} R^I([\![t_1]\!]^I_\rho,\dots,[\![t_m]\!]^I_\rho) , & [\![\neg\varphi]\!]^I_\rho &\stackrel{\text{def}}{=} \neg[\![\varphi]\!]^I_\rho ,\\
[\![\varphi\vee\psi]\!]^I_\rho &\stackrel{\text{def}}{=} [\![\varphi]\!]^I_\rho\vee[\![\psi]\!]^I_\rho , & [\![\exists x.\varphi]\!]_\rho &\stackrel{\text{def}}{=} \bigvee_{e\in D_I} [\![\varphi]\!]^I_{\rho[e/x]} .
\end{aligned}$$

On dit que (I,ρ) *satisfait* φ, noté $I,\rho\vDash\varphi$, si $[\![\varphi]\!]^I_\rho=\top$; cette écriture peut être définie de manière équivalente par

$$\begin{aligned}
&I,\rho\vDash R(t_1,\dots,t_m) && \text{si } ([\![t_1]\!]^I_\rho,\dots,[\![t_m]\!]^I_\rho)\in R^I ,\\
&I,\rho\vDash\neg\varphi && \text{si } I,\rho\nvDash\varphi ,\\
&I,\rho\vDash\varphi\vee\psi && \text{si } I,\rho\vDash\varphi \text{ ou } I,\rho\vDash\psi ,\\
&I,\rho\vDash\exists x.\varphi && \text{si } \exists e\in D_I.\, I,\rho[x\mapsto e]\vDash\varphi .
\end{aligned}$$

On dit que I est un *modèle* de φ, noté $I\vDash\varphi$, s'il existe ρ telle que $I,\rho\vDash\varphi$. Une formule φ est *satisfiable* s'il existe un modèle de φ. Elle est *valide*, noté $\vDash\varphi$, si pour toute interprétation I et toute valuation ρ, $I,\rho\vDash\varphi$. Pour un ensemble de formules S, une interprétation I et une valuation ρ, on écrit $I,\rho\vDash S$ si $I,\rho\vDash\psi$ pour toutes les formules $\psi\in S$. Si de plus φ est une

formule, on écrit $S \vDash \varphi$ si pour toute paire (I, ρ) telle que $I, \rho \vDash S$ on a $I, \rho \vDash \varphi$. Un ensemble S est *insatisfiable* s'il n'existe pas I et ρ tels que $I, \rho \vDash S$, ou de manière équivalente si $S \vDash \bot$.

Exemple 2.1 (formule du buveur)**.** Considérons une signature avec une relation unaire B. La formule $\exists x.(B(x) \Rightarrow \forall y.B(y))$, qui se lit habituellement « il existe un individu tel que s'il boit alors tout le monde boit », est valide. En effet, dans toute interprétation I,

- soit $D_I \subseteq B^I$, et alors $I, [e/x] \vDash \forall y.B(y)$ pour tout choix de $e \in D_I$ et donc la formule du buveur est satisfaite car D_I est non vide et un tel élément e existe,
- soit il existe un individu « sobre » $s \in D_I$ tel que $s \notin B^I$, et alors $I, [s/x] \vDash \neg B(x)$ et la formule du buveur est satisfaite.

On notera que cette formule n'est pas *valide si l'on permet un domaine d'interprétation vide. Elle n'est pas non plus valide en logique intuitionniste du premier ordre.*

Notons enfin que la satisfaction d'une formule ne dépend que de la valuation de ses variables libres :

Propriété 2.2. *Pour toute formule φ (resp. terme t), toute interprétation I, et toutes valuations ρ et ρ' telles que pour tout $x \in \mathrm{fv}(\varphi)$ (resp. $x \in \mathrm{fv}(t)$) $\rho(x) = \rho'(x)$, $[\![\varphi]\!]^I_\rho = [\![\varphi]\!]^I_{\rho'}$ (resp. $[\![t]\!]^I_\rho = [\![t]\!]^I_{\rho'}$).*

En particulier, par la propriété 2.2, φ avec $\mathrm{fv}(\varphi) = \{x_1, \dots, x_n\}$ est satisfiable si et seulement la formule close $\exists x_1 \dots \exists x_n.\varphi$ est satisfiable, et φ est valide si et seulement si la formule close $\forall x_1 \dots \forall x_n.\varphi$.

3. Substitutions

Une *substitution* est une fonction $\sigma \colon X \to T(\mathcal{F}, X)$ de *domaine* $\mathrm{dom}(\sigma) \stackrel{\text{def}}{=} \{x \in X \mid \sigma(x) \neq x\}$ fini. On note aussi σ comme $[\sigma(x_1)/x_1, \dots, \sigma(x_n)/x_n]$ où $x_1, \dots, x_n$ sont les variables distinctes de $\mathrm{dom}(\sigma)$.

Une substitution définit un unique *homomorphisme* $t \mapsto t\sigma$ de $T(\mathcal{F}, X)$ dans $T(\mathcal{F}, X)$ par

$$x\sigma \stackrel{\text{def}}{=} \sigma(x)\ , \qquad\qquad f(t_1, \dots, t_m)\sigma \stackrel{\text{def}}{=} f(t_1\sigma, \dots, t_m\sigma)$$

pour tout $m \in \mathbb{N}$ et tout $f \in \mathcal{F}_m$. Une substitution à image dans $T(\mathcal{F})$ est dite *close*.

Ces définitions s'étendent aussi aux formules. Cependant, du fait de la présence de variables liées dans les formules, une substitution n'est pas applicable à n'importe quelle formule. Pour une substitution σ et une formule φ, on note $\mathrm{rv}_\varphi(\sigma) \stackrel{\text{def}}{=} \bigcup_{x \in \mathrm{dom}(\sigma) \cap \mathrm{fv}(\varphi)} \mathrm{fv}(\sigma(x))$ son ensemble de variables images (« *range variables* » en anglais). On dit que σ est *applicable* à φ si $(\mathrm{dom}(\sigma) \cup \mathrm{rv}_\varphi(\sigma)) \cap \mathrm{bv}(\varphi) = \emptyset$, et dans ce cas on définit l'application $\varphi\sigma$ de σ à φ par

$$R(t_1, \dots, t_m)\sigma \stackrel{\text{def}}{=} R(t_1\sigma, \dots, t_m\sigma)\ , \qquad (\neg\varphi)\sigma \stackrel{\text{def}}{=} \neg(\varphi\sigma)\ ,$$
$$(\varphi \vee \psi)\sigma \stackrel{\text{def}}{=} (\varphi\sigma) \vee (\psi\sigma)\ , \qquad (\exists x.\varphi)\sigma \stackrel{\text{def}}{=} \exists x.(\varphi\sigma)\ ,$$

où $m \in \mathbb{N}$ et $R \in \mathcal{P}_m$. La composition $\sigma\sigma'$ de deux substitutions σ et σ' est définie par $\varphi(\sigma\sigma') \stackrel{\text{def}}{=} (\varphi\sigma)\sigma'$. Par exemple $(B(x) \vee B(y))[y/x, z/y] = B(y) \vee B(z)$, $(B(x) \vee B(y))[y/x][z/y] = B(z) \vee B(z)$, et $(B(x) \vee B(y))[z/x] = B(z) \vee B(y)$.

Lemme de substitution. Les valuations fournissent le pendant côté modèles des substitutions côté formules. Pour une substitution σ et une valuation ρ, notons $\sigma\rho$ la valuation $x \mapsto [\![\sigma(x)]\!]^I_\rho$ pour tout $x \in X$. Nous préparons ici le terrain pour le lemme 3.5 de substitution, qui sera énoncé sous une forme plus générale un peu plus loin.

Lemme 3.1. *Pour tout terme t, toute substitution σ, toute interprétation I, et toute valuation ρ, $[\![t\sigma]\!]^I_\rho = [\![t]\!]^I_{\sigma\rho}$.*

Démonstration. Par induction structurelle sur t.

Pour le cas de base d'une variable x, $[\![x\sigma]\!]^I_\rho = [\![\sigma(x)]\!]^I_\rho = (\sigma\rho)(x) = [\![x]\!]^I_{\sigma\rho}$.

Pour un terme $f(t_1, \dots, t_m)$, $[\![f(t_1, \dots, t_m)\sigma]\!]^I_\rho = [\![f(t_1\sigma, \dots, t_m\sigma)]\!]^I_\rho = f^I([\![t_1\sigma]\!]^I_\rho, \dots, [\![t_m\sigma]\!]^I_\rho) \stackrel{\text{h.i.}}{=} f^I([\![t_1]\!]^I_{\sigma\rho}, \dots, [\![t_m]\!]^I_{\sigma\rho}) = [\![f(t_1, \dots, t_m)]\!]^I_{\sigma\rho}$. □

Lemme 3.2. *Pour toute formule φ, toute substitution σ applicable à φ, toute interprétation I, et toute valuation ρ, $[\![\varphi\sigma]\!]^I_\rho = [\![\varphi]\!]^I_{\sigma\rho}$.*

Démonstration. Par induction structurelle sur φ.

Pour une formule atomique $R(t_1, \dots, t_m)$, une négation $\neg\varphi$, ou une disjonction $\varphi \vee \psi$, cela découle du lemme 3.1 et de l'hypothèse d'induction. Pour une quantification $\exists x.\varphi$,

$$[\![(\exists x.\varphi)\sigma]\!]^I_\rho = [\![\exists x.(\varphi\sigma)]\!]^I_\rho = \bigvee_{e\in D_I} [\![\varphi\sigma]\!]^I_{\rho[e/x]} \overset{\text{h.i.}}{=} \bigvee_{e\in D_I} [\![\varphi]\!]^I_{\sigma(\rho[e/x])} = [\![\exists x.\varphi]\!]^I_{\sigma\rho} ,$$

où il faut montrer pour justifier cette dernière étape que, pour toute variable libre $z \in \mathrm{fv}(\varphi)$, $(\sigma(\rho[e/x]))(z) = ((\sigma\rho)[e/x])(z)$, ce qui permettra de conclure par la propriété 2.2. Si $z = x$, alors $(\sigma(\rho[e/x]))(x) = [\![\sigma(x)]\!]^I_{\rho[e/x]} = [\![x]\!]^I_{\rho[e/x]} = e = [\![x]\!]^I_{(\sigma\rho)[e/x]} = ((\sigma\rho)[e/x])(x)$ où $\sigma(x) = x$ puisque, comme σ est applicable à $\exists x.\varphi$ et $x \in \mathrm{bv}(\exists x.\varphi)$, $x \notin \mathrm{dom}(\sigma)$. Si $z \neq x$, alors $(\sigma(\rho[e/x]))(z) = [\![\sigma(z)]\!]^I_{\rho[e/x]} = [\![\sigma(z)]\!]^I_\rho = ((\sigma\rho)[e/x])(z)$, où l'égalité centrale repose sur le fait que $x \notin \mathrm{fv}(\sigma(z))$, qui à son tour découle de $\mathrm{bv}(\exists x.\varphi) \cap \mathrm{rv}_{\exists x.\varphi}(\sigma) = \emptyset$ puisque σ est applicable à $\exists x.\varphi$. □

💬 *L'α-renommage est aussi une notion de base du λ-calcul dans la leçon **929**.*

α-renommages. Quand une substitution σ n'est pas applicable à une formule φ, il est cependant possible d'effectuer un α-renommage à φ pour obtenir une formule φ' sur laquelle appliquer σ. On raisonnera donc implicitement non sur des formules mais sur des classes de formules équivalentes à α-renommage près. Il est important dans ce cas que cette opération définisse une congruence (l'α-congruence) sur les formules pour pouvoir continuer à raisonner inductivement, et que l'opération préserve la sémantique des formules.

On définit l'opération d'α-renommage par

$$\exists x.\varphi \to_\alpha \exists y.\varphi[y/x] \qquad (\alpha\text{-renommage})$$

où $y \notin \mathrm{fv}(\exists x.\varphi)$ et $[y/x]$ est applicable à φ. À noter que l'α-renommage n'impacte que les variables liées de φ, ce qui explique pourquoi il en préserve la sémantique (voir le lemme 3.4 ci-dessous).

On définit l'*α-congruence* $=_\alpha$ comme la plus petite congruence entre formules engendrée par $\to_\alpha$, c'est-à-dire la plus petite relation réflexive, symétrique et transitive telle que $\varphi \to_\alpha \psi$ implique $\varphi =_\alpha \psi$ et telle que si $\varphi =_\alpha \psi$ et $\varphi' =_\alpha \psi'$, alors $\neg\varphi =_\alpha \neg\psi$, $\varphi \vee \varphi' =_\alpha \psi \vee \psi'$ et $\exists x.\varphi =_\alpha \exists x.\psi$.

En appliquant des α-renommages inductivement sur les sous-formules croissantes de φ, on peut au besoin renommer toutes les variables liées de φ, et on déduit :

Propriété 3.3 (applicabilité)**.** *Pour toute substitution σ et toute formule φ, il existe une formule φ' telle que $\varphi =_\alpha \varphi'$ et que σ soit applicable à φ'. De plus, si $\varphi =_\alpha \varphi''$ et σ est aussi applicable à φ'', alors $\varphi' =_\alpha \varphi''$ et $\varphi'\sigma =_\alpha \varphi''\sigma$.*

Il reste à vérifier que les α-renommages n'impactent pas la sémantique des formules.

Lemme 3.4 (α-congruence)**.** *Soient φ et ψ deux formules telles que $\varphi =_\alpha \psi$. Alors pour toute interprétation I et toute valuation ρ, $[\![\varphi]\!]^I_\rho = [\![\psi]\!]^I_\rho$.*

Démonstration. Par induction sur la congruence $=_\alpha$.

Le cas de base est celui d'un α-renommage $\exists x.\varphi \to_\alpha \exists y.\varphi[y/x]$ avec $y \notin \mathrm{fv}(\exists x.\varphi)$ et $[y/x]$ applicable. Puisque $[y/x]$ est applicable, par le lemme 3.2,

$$[\![\exists y.\varphi[y/x]]\!]^I_\rho = \bigvee_{e\in D_I} [\![\varphi[y/x]]\!]^I_{\rho[e/y]} = \bigvee_{e\in D_i} [\![\varphi]\!]^I_{[y/x](\rho[e/y])} = \bigvee_{e\in D_i} [\![\varphi]\!]^I_{\rho[e/x]} = [\![\exists x.\varphi]\!]^I_\rho$$

où nous devons justifier pour l'avant-dernière étape que, pour toute variable libre $z \in \mathrm{fv}(\varphi)$, $([y/x](\rho[e/y]))(z) = (\rho[e/x])(z)$, ce qui permettra de conclure ce cas de base par la propriété 2.2.

Si $x = z$, alors $([y/x](\rho[e/y]))(x) = [\![y]\!]^I_{\rho[e/y]} = e = [\![x]\!]^I_{\rho[e/x]} = (\rho[e/x])(x)$. Si $x \neq z$, alors d'une part $([y/x](\rho[e/y]))(z) = [\![z]\!]^I_{\rho[e/y]} = [\![z]\!]^I_\rho$ puisque $y \notin \mathrm{fv}(\exists x.\varphi)$ et donc $y \neq z$, et d'autre part $(\rho[e/x])(z) = [\![z]\!]^I_{\rho[e/x]} = [\![z]\!]^I_\rho$.

Le cas de base de la réflexivité ainsi que les étapes d'induction pour la symétrie, la transitivité, et la congruence pour $\neg$, $\vee$ et $\exists$ sont évidents puisque $[\![\varphi]\!]^I_\rho = [\![\psi]\!]^I_\rho$ vue comme une relation entre formules est aussi une congruence. □

La propriété 3.3 et le lemme 3.4 justifient un abus de notation : nous écrirons désormais « $\varphi\sigma$ » pour une formule $\varphi'\sigma$ où $\varphi =_\alpha \varphi'$ et σ est applicable à φ'. Le lemme 3.1 ainsi que le lemme 3.4 d'α-congruence combiné au lemme 3.2 nous permettent alors d'énoncer le lemme de substitution.

Lemme 3.5 (substitution)**.** *Pour tout terme t, toute formule φ, toute substitution σ, toute interprétation I, et toute valuation ρ, $[\![t\sigma]\!]^I_\rho = [\![t]\!]^I_{\sigma\rho}$ et $[\![\varphi\sigma]\!]^I_\rho = [\![\varphi]\!]^I_{\sigma\rho}$.*

[Goubault-Larrecq et Mackie, 1997, thm. 6.8].

On utilisera par la suite deux identités aisément démontrables par induction sur φ : en appliquant au besoin une α-congruence à φ pour rendre les substitutions applicables, si $x \notin \mathrm{fv}(\varphi)$, alors

$$\varphi[t/x] =_\alpha \varphi \ , \tag{1}$$

et si $x \notin \mathrm{fv}(\varphi) \setminus \{y\}$, alors

$$\varphi[x/y][t/x] =_\alpha \varphi[t/y] \ . \tag{2}$$

4. Formes normales

[David et al., 2003, sec. 2.6], [Goubault-Larrecq et Mackie, 1997, sec. 6.5.1]

Comme en logique propositionnelle, les formules du premier ordre peuvent être mises sous différentes formes normales préservant la sémantique (formules *équivalentes*) ou préservant la satisfiabilité (formules *équi-satisfiables*).

Forme normale négative. Une formule du premier ordre est en *forme normale négative* si elle respecte la syntaxe abstraite

$$\ell ::= \alpha \mid \neg\alpha \qquad \text{(littéraux)}$$
$$\varphi ::= \ell \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \exists x.\varphi \mid \forall x.\varphi \qquad \text{(formules)}$$

où α est une formule atomique et $x \in X$. En d'autres termes, les négations ne peuvent apparaître que devant des formules atomiques. La mise sous forme normale négative procède en « poussant » les négations vers les feuilles ; pour une formule φ, on notera $\mathrm{nnf}(\varphi)$ sa forme normale négative obtenue inductivement par

$$\mathrm{nnf}(\alpha) \stackrel{\text{def}}{=} \alpha \ , \qquad \mathrm{nnf}(\neg\alpha) \stackrel{\text{def}}{=} \neg\alpha \ ,$$
$$\mathrm{nnf}(\varphi \vee \psi) \stackrel{\text{def}}{=} \mathrm{nnf}(\varphi) \vee \mathrm{nnf}(\psi) \ , \qquad \mathrm{nnf}(\varphi \wedge \psi) \stackrel{\text{def}}{=} \mathrm{nnf}(\varphi) \wedge \mathrm{nnf}(\psi) \ ,$$
$$\mathrm{nnf}(\neg(\varphi \vee \psi)) \stackrel{\text{def}}{=} \mathrm{nnf}(\neg\varphi) \wedge \mathrm{nnf}(\neg\psi) \ , \qquad \mathrm{nnf}(\neg(\varphi \wedge \psi)) \stackrel{\text{def}}{=} \mathrm{nnf}(\neg\varphi) \vee \mathrm{nnf}(\neg\psi) \ ,$$
$$\mathrm{nnf}(\exists x.\varphi) \stackrel{\text{def}}{=} \exists x.\mathrm{nnf}(\varphi) \ , \qquad \mathrm{nnf}(\forall x.\varphi) \stackrel{\text{def}}{=} \forall x.\mathrm{nnf}(\varphi) \ ,$$
$$\mathrm{nnf}(\neg\exists x.\varphi) \stackrel{\text{def}}{=} \forall x.\mathrm{nnf}(\neg\varphi) \ , \qquad \mathrm{nnf}(\neg\forall x.\varphi) \stackrel{\text{def}}{=} \exists x.\mathrm{nnf}(\neg\varphi) \ ,$$
$$\mathrm{nnf}(\neg\neg\varphi) \stackrel{\text{def}}{=} \mathrm{nnf}(\varphi) \ .$$

En termes algorithmiques, cette mise sous forme normale négative se fait en temps linéaire. Elle préserve visiblement la sémantique des formules : φ et $\mathrm{nnf}(\varphi)$ sont équivalentes. On notera en général

$$\overline{\varphi} \stackrel{\text{def}}{=} \mathrm{nnf}(\neg\varphi)$$

pour la forme normale négative de la négation de φ.

Exemple 4.1. La formule du buveur de l'exemple 2.1 s'écrit sous forme normale négative comme $\exists x.(\neg B(x) \vee \forall y.B(y))$.

On peut observer par induction structurelle sur φ que

Propriété 4.2. *Pour toute formule φ en forme normale négative et toute substitution σ, $(\overline{\varphi})\sigma = \overline{(\varphi\sigma)}$.*

La *profondeur* $p(\varphi)$ d'une formule φ en forme normale négative est définie inductivement par $p(\ell) \stackrel{\text{def}}{=} 0$ pour un littéral, $p(\varphi \vee \psi) = p(\varphi \wedge \psi) \stackrel{\text{def}}{=} 1 + \max\{p(\varphi), p(\psi)\}$, et $p(\exists x.\varphi) = p(\forall x.\varphi) \stackrel{\text{def}}{=} 1 + p(\varphi)$. À noter que la profondeur d'une formule est la même, qu'elle soit représentée sous forme d'arbre ou sous forme de graphe acyclique orienté comme dans la figure 1.

[David et al., 2003, sec. 2.6.2], [Goubault-Larrecq et Mackie, 1997, thm. 6.10]

Forme prénexe. Une formule est sous *forme prénexe* si elle est de la forme $Q_1 x_1 \dots Q_n x_n.\psi$ où $Q_i \in \{\forall, \exists\}$ pour tout $1 \leq i \leq n$ et ψ est sans quantificateur, c'est-à-dire ψ respecte la syntaxe abstraite

$$\psi ::= \ell \mid \psi \vee \psi \mid \psi \wedge \psi \,. \qquad \text{(formules sans quantificateur)}$$

Propriété 4.3 (forme prénexe)**.** *Pour toute formule φ, on peut construire une formule prénexe équivalente.*

Démonstration. Sans perte de généralité, on peut supposer que φ est déjà sous forme normale négative. On procède alors par induction sur φ :

Cas de ℓ : Alors ℓ est déjà sous forme prénexe.

Cas de $\varphi \vee \varphi'$: Par hypothèse d'induction, φ et φ' sont équivalentes à $Q_1 x_1 \dots Q_n x_n.\psi$ et $Q'_1 y_1 \dots Q'_m y_m.\psi'$ où ψ et ψ' sont sans quantificateur. Par le lemme 3.4 d'α-congruence, on peut supposer que ces formules utilisent des ensembles disjoints de variables liées : $\{x_1, \dots, x_n\} \cap \{y_1, \dots, y_m\} = \emptyset$. Montrons que $(Q_1 x_1 \dots Q_n x_n.\psi) \vee (Q'_1 y_1 \dots Q'_m y_m.\psi')$ est équivalente à $Q_1 x_1 \dots Q_n x_n . Q'_1 y_1 \dots Q'_m y_m.(\psi \vee \psi')$, qui sera donc équivalente à $\varphi \vee \varphi'$. On procède pour cela par récurrence sur $n + m$. Le cas de base pour $n + m = 0$ est trivial. Pour l'étape de récurrence, il suffit de montrer que pour toutes formules θ et θ' et $x \notin \mathrm{fv}(\theta')$,

- $(\exists x.\theta) \vee \theta'$ est équivalente à $\exists x.\theta \vee \theta'$: en effet, pour toute interprétation I et toute valuation ρ,

$$\begin{aligned}
[\![(\exists x.\theta) \vee \theta']\!]^I_\rho &= (\bigvee_{e \in D_I} [\![\theta]\!]^I_{\rho[e/x]}) \vee [\![\theta']\!]^I_\rho \\
&= (\bigvee_{e \in D_I} [\![\theta]\!]^I_{\rho[e/x]}) \vee (\bigvee_{e \in D_I} [\![\theta']\!]^I_{\rho[e/x]}) \qquad \text{(car } D_I \neq \emptyset \text{ et par la propriété 2.2)} \\
&= \bigvee_{e \in D_I} [\![\theta \vee \theta']\!]^I_{\rho[e/x]} \\
&= [\![\exists x.(\theta \vee \theta')]\!]^I_\rho \,.
\end{aligned}$$

- $(\forall x.\theta) \vee \theta'$ est équivalente à $\forall x.\theta \vee \theta'$: c'est bien le cas par un raisonnement similaire.

Cas de $\varphi \wedge \varphi'$: Similaire au cas précédent.

Cas de $\exists x.\varphi$: Par hypothèse d'induction, φ est équivalente à $Q_1 x_1 \dots Q_n x_n.\psi$ où ψ est sans quantificateur. Alors $\exists x.\varphi$ est équivalente à $\exists x.Q_1 x_1 \dots Q_n x_n.\psi$.

Cas de $\forall x.\varphi$: Similaire au cas précédent. □

Exemple 4.4. La négation de la formule du buveur de l'exemple 2.1 s'écrit sous forme normale négative comme $\forall x.(B(x) \wedge \exists y.\neg B(y))$. Une forme prénexe de cette négation est alors $\forall x.\exists y.(B(x) \wedge \neg B(y))$.

Skolémisation. Une formule $\varphi = Q_1 x_1 \dots Q_n x_n . \psi$ sous forme prénexe est *existentielle* si $Q_i = \exists$ pour tout $1 \leq i \leq n$, et *universelle* si $Q_i = \forall$ pour tout $1 \leq i \leq n$. La skolémisation d'une formule φ est une formule universelle équi-satisfiable.

[David et al., 2003, sec. 2.7], [Goubault-Larrecq et Mackie, 1997, thm. 6.12]

On étend pour cela l'ensemble $\mathcal{F}$ des symboles de fonction. Pour toute formule φ et toute variable x, on fixe une énumération $\vec{x}_\varphi$ de $\mathrm{fv}(\exists x.\varphi)$ et on ajoute un nouveau symbole de fonction $f_{\varphi,x} \notin \mathcal{F}$ d'arité $|\mathrm{fv}(\exists x.\varphi)|$. Soit $\mathcal{F}' \stackrel{\text{def}}{=} \mathcal{F} \uplus \{f_{\varphi,x} \mid f \text{ formule sur } (\mathcal{F},\mathcal{P}) \text{ et } x \in X\}$.

La *skolémisation* $s(\varphi)$ d'une formule φ en forme normale négative est alors définie par

$$\begin{aligned} s(\ell) &\stackrel{\text{def}}{=} \ell \ , \\ s(\varphi \vee \psi) &\stackrel{\text{def}}{=} s(\varphi) \vee s(\psi) \ , \\ s(\varphi \wedge \psi) &\stackrel{\text{def}}{=} s(\varphi) \wedge s(\psi) \ , \\ s(\exists x.\varphi) &\stackrel{\text{def}}{=} (s(\varphi))[f_{\varphi,x}(\vec{x}_\varphi)/x] \\ s(\forall x.\varphi) &\stackrel{\text{def}}{=} \forall x.s(\varphi) \ . \end{aligned}$$

On peut noter que, si φ était sous forme prénexe, alors $s(\varphi)$ est universelle.

Théorème 4.5 (Skolem)**.** *Soit φ une formule du premier ordre. Alors on peut construire une formule universelle équi-satisfiable.*

Démonstration. On peut supposer φ sous forme prénexe. Il suffit alors de montrer que φ et $s(\varphi)$ sont équi-satisfiables.

Commençons par montrer que, si φ est satisfiable, alors $s(\varphi)$ l'est aussi. Pour toute interprétation I sur la signature $(\mathcal{F},\mathcal{P})$, on construit une interprétation I' sur la signature $(\mathcal{F}',\mathcal{P})$. L'interprétation I' a le même domaine $D_{I'} \stackrel{\text{def}}{=} D_I$ que I et interprète les symboles de $\mathcal{F}$ et $\mathcal{P}$ de la même manière : $f^{I'} \stackrel{\text{def}}{=} f^I$ et $R^{I'} \stackrel{\text{def}}{=} R^I$ pour tout $f \in \mathcal{F}$ et $R \in \mathcal{P}$. Il faut maintenant fournir une interprétation des symboles $f_{\varphi,x}$. Considérons pour cela une formule φ et une variable x. Soit $x_1,\dots,x_n$ l'énumeration de $\mathrm{fv}(\exists x.\varphi)$. Pour tout tuple $(e_1,\dots,e_n) \in D_I^n$, on définit $D_{I,\varphi,x}(e_1,\dots,e_n) \stackrel{\text{def}}{=} \{e \in D_I \mid I, \rho[e_1/x_1,\dots,e_n/x_n,e/x] \vDash \varphi\}$. Comme $D_I \neq \emptyset$, on peut choisir un élément $e_\emptyset \in D_I$. On étend alors l'interprétation I : si $D_{I,\varphi,x}(e_1,\dots,e_n)$ est non vide, on choisit un représentant $f^{I'}_{\varphi,x}(e_1,\dots,e_n) \in D_{I,\varphi,x}(e_1,\dots,e_n)$, et sinon on pose $f^{I'}_{\varphi,x}(e_1,\dots,e_n) = e_\emptyset$.

Montrons maintenant par induction sur φ que, pour toute interprétation I et toute valuation ρ, $I,\rho \vDash \varphi$ implique $I',\rho \vDash s(\varphi)$. On ne traite ici que le cas d'une formule $\exists x.\varphi$. Soit $x_1,\dots,x_n$ l'énumération de $\mathrm{fv}(\exists x.\varphi)$. On a les implications

$$\begin{aligned} & I,\rho \vDash \exists x.\varphi \\ \Rightarrow \quad & \exists e \in D_I,\ I,\rho[e/x] \vDash \varphi \\ \Rightarrow \quad & I,\rho[f^{I'}_{\varphi,x}(\rho(x_1),\dots,\rho(x_n))/x] \vDash \varphi \\ \Rightarrow \quad & I',\rho[f^{I'}_{\varphi,x}(\rho(x_1),\dots,\rho(x_n))/x] \vDash s(\varphi) && \text{par hyp. ind.} \\ \Rightarrow \quad & I',\rho \vDash (s(\varphi))[f_{\varphi,x}(x_1,\dots,x_n)] && \text{par le lemme 3.5} \\ \Rightarrow \quad & I',\rho \vDash s(\exists x.\varphi) \ . \end{aligned}$$

Inversement, montrons que si $s(\varphi)$ est satisfiable, alors φ l'est aussi. Pour une interprétation I sur la signature $(\mathcal{F}',\mathcal{P})$, on définit l'interprétation $I_{|\mathcal{F}}$ comme sa restriction à la signature $(\mathcal{F},\mathcal{P})$. Montrons par induction sur φ que $I,\rho \vDash s(\varphi)$ implique $I_{|\mathcal{F}},\rho \vDash \varphi$. Encore une fois, nous ne traitons que le cas d'une formule $\exists x.\varphi$ où $x_1,\dots,x_n$ est l'énumération de $\mathrm{fv}(\exists x.\varphi)$. On a les

implications

$$\begin{aligned} & I, \rho \vDash s(\exists x.\varphi) \\ \Rightarrow \quad & I, \rho \vDash (s(\varphi))[f^I_{\varphi,x}(x_1, \dots, x_n)/x] \\ \Rightarrow \quad & I, \rho[f^I_{\varphi,x}(\rho(x_1), \dots, \rho(x_n))/x] \vDash s(\varphi) && \text{par le lemme 3.5} \\ \Rightarrow \quad & I_{|\mathcal{F}}, \rho[f^I_{\varphi,x}(\rho(x_1), \dots, \rho(x_n))/x] \vDash \varphi && \text{par hyp. ind.} \\ \Rightarrow \quad & I_{|\mathcal{F}}, \rho \vDash \exists x.\varphi \ . \end{aligned}$$ □

[David et al., 2003, sec. 7.4.2]

Forme clausale. Une *clause* est une formule close universelle $\forall x_1 \dots \forall x_n.\psi$ où ψ est une disjonction de littéraux. Une formule est sous *forme clausale* si c'est une conjonction de clauses, que l'on présente aussi comme un ensemble de clauses.

Théorème 4.6 (forme clausale)**.** *Soit φ une formule. Alors on peut construire un ensemble équi-satisfiable $Cl(\varphi)$ de clauses.*

Démonstration. Soit φ une formule. En quantifiant existentiellement ses variables libres $\mathrm{fv}(\varphi) = \{x_1, \dots, x_n\}$, on obtient une formule close équi-satisfiable $\varphi' \stackrel{\text{def}}{=} \exists x_1 \dots \exists x_n.\varphi$. Par le théorème 4.5 de Skolem, on construit une formule universelle close équi-satisfiable $s(\varphi')$. Il reste simplement à mettre $s(\varphi')$ sous forme normale conjonctive, en utilisant le fait que $\forall x.\varphi \wedge \psi$ est équivalente à $(\forall x.\varphi) \wedge (\forall x.\psi)$. On obtient ainsi une écriture de $s(\varphi)$ comme $\bigwedge_i \forall \vec{x}. \bigvee_{j_i} \ell_{j_i}$, et on définit alors $\mathrm{Cl}(\varphi) \stackrel{\text{def}}{=} \{\forall \vec{x}. \bigvee_{j_i} \ell_{j_i}\}_i$. □

*Voir la leçon **916** pour la mise sous forme normale conjonctive.*

Exemple 4.7. La forme clausale de la *négation* $\neg(\exists x.B(x) \Rightarrow \forall y.B(y))$ de la formule du buveur est $\neg B(a) \wedge \forall x.B(x)$.

5. Modèles de Herbrand

[Goubault-Larrecq et Mackie, 1997, sec. 5.2]

Supposons $\mathcal{F}_0 \neq \emptyset$, de telle sorte que $T(\mathcal{F})$ soit non vide. La *base de Herbrand*

$$\mathcal{B} \stackrel{\text{def}}{=} \{R(t_1, \dots, t_m) \mid m \in \mathbb{N}, R \in \mathcal{P}_m, t_1, \dots, t_m \in T(\mathcal{F})\}$$

est l'ensemble des instanciations des symboles de relation de $\mathcal{P}$. Une *interprétation de Herbrand* est une valuation $H : \mathcal{B} \to \mathbb{B}$ qui associe une valeur de vérité à chaque élément de la base de Herbrand. On identifie H avec l'interprétation de domaine $D_H \stackrel{\text{def}}{=} T(\mathcal{F})$ qui associe

$$f^H(t_1, \dots, t_m) \stackrel{\text{def}}{=} f(t_1, \dots, t_m) \ , \qquad R^H(t_1, \dots, t_m) \stackrel{\text{def}}{=} H(R(t_1, \dots, t_m))$$

à tout $f \in \mathcal{F}_m$ et $R \in \mathcal{P}_m$ pour tout m.

Théorème 5.1 (Herbrand)**.** *Soit S un ensemble de formules universelles closes. Alors S est satisfiable si et seulement si S est satisfiable dans une interprétation de Herbrand.*

Démonstration. Seul le sens direct nécessite une preuve. Soit I une interprétation telle que $I \vDash S$. On définit l'interprétation de Herbrand H par $H(R(t_1, \dots, t_m)) \stackrel{\text{def}}{=} [\![R(t_1, \dots, t_m)]\!]^I$ pour tous $m \in \mathbb{N}$, $R \in \mathcal{P}_m$, et $t_1, \dots, t_m \in T(\mathcal{F})$; comme les formules et termes en question sont clos, il n'est pas utile de préciser sous quelle valuation.

⚠ *Les formules de S doivent être* closes *: $S = \{P(x), \neg P(a)\}$ avec $\mathcal{F} = \{a\}$ est satisfiable mais ne l'est pas dans une interprétation de Herbrand.*

Montrons que $H \vDash S$. Pour toute formule close $\forall x_1 \dots \forall x_n.\varphi \in S$ où φ est sans quantificateur et $\mathrm{fv}(\varphi) = \{x_1, \dots, x_n\}$, et pour tous $t_1, \dots, t_n \in D_H = T(\mathcal{F})$, on définit $\sigma \stackrel{\text{def}}{=} [t_1/x_1, \dots, t_n/x_n]$ – qui peut être vue à la fois comme une substitution close et comme une valuation dans D_H – et $\rho \stackrel{\text{def}}{=} [[\![t_1]\!]^I/x_1, \dots, [\![t_n]\!]^I/x_n]$ une valuation dans I. On montre $[\![\varphi]\!]^H_\sigma = [\![\varphi]\!]^I_\rho$ par induction sur φ ; par suite, comme $I \vDash \forall x_1 \dots \forall x_n.\varphi$, on en déduit $H \vDash \forall x_1 \dots \forall x_n.\varphi$ comme voulu.

- Pour une formule atomique α, par définition de H, $[\![\alpha]\!]^H_\sigma = [\![\alpha\sigma]\!]^H$ par le lemme 3.5, et cette sémantique est égale à $[\![\alpha\sigma]\!]^I$ par définition de R^H, qui n'est autre que $[\![\alpha]\!]^I_\rho$ par le lemme 3.5.

— Pour une négation $\neg\varphi$, $[\![\neg\varphi]\!]^H_\sigma = \neg[\![\varphi]\!]^H_\sigma \overset{\text{h.i.}}{=} \neg[\![\varphi]\!]^I_\rho = [\![\neg\varphi]\!]^I_\rho$.
— Pour une disjonction $\varphi \vee \psi$, $[\![\varphi \vee \psi]\!]^H_\sigma = [\![\varphi]\!]^H_\sigma \vee [\![\psi]\!]^H_\sigma \overset{\text{h.i.}}{=} [\![\varphi]\!]^I_\rho \vee [\![\psi]\!]^I_\rho = [\![\varphi \vee \psi]\!]^I_\rho$. □

Le théorème de HERBRAND permet de « réduire » la logique du premier ordre à la logique propositionnelle. Pour une formule universelle close $\varphi = \forall\vec{x}.\varphi'$ où φ' est sans quantificateur, on note $\varphi\Sigma \overset{\text{def}}{=} \{\varphi'\sigma \mid \sigma \text{ substitution close}\}$ l'ensemble de ses *instances de HERBRAND*. Pour un ensemble de formules universelles closes S, on écrit $S\Sigma \overset{\text{def}}{=} \bigcup_{\varphi\in S} \varphi\Sigma$. Une formule de $S\Sigma$ peut ainsi être vue comme une formule propositionnelle avec la base de HERBRAND $\mathcal{B}$ comme ensemble de propositions.

Propriété 5.2. *Soit S un ensemble de formules universelles closes et H une interprétation de HERBRAND. Alors $H \vDash S$ si et seulement si $H \vDash S\Sigma$.*

Démonstration. Pour toute formule close $\varphi = \forall x_1 \ldots \forall x_n.\varphi' \in S$ où φ' est sans quantificateur et $\text{fv}(\varphi') = \{x_1, \ldots, x_n\}$, $H \vDash \varphi$ si et seulement si pour tous termes clos $t_1, \ldots, t_n \in D_H$, $H, [t_1/x_1, \ldots, t_n/x_n] \vDash \varphi'$, si et seulement si pour tous termes clos $t_1, \ldots, t_n \in T(\mathcal{F})$, $H \vDash \varphi'[t_1/x_1, \ldots, t_n/x_n]$, si et seulement si pour toute formule $\psi \in \varphi\Sigma$, $H \vDash \psi$. □

Corollaire 5.3. *Soit S un ensemble de formules universelles closes. Alors S est satisfiable si et seulement si $S\Sigma$ est propositionnellement satisfiable sur l'ensemble de propositions $\mathcal{B}$.*

Démonstration. Par le théorème 5.1, S est satisfiable si et seulement s'il existe une interprétation de HERBRAND H telle que $H \vDash S$. Il suffit alors d'appliquer la propriété 5.2 à S et H. □

Partie 2. Calcul des séquents

[David et al., 2003, ch. 5], [Goubault-Larrecq et Mackie, 1997, fig. 6.2] pour le calcul des séquents. Voir [Goubault-Larrecq et Mackie, 1997, sec. 6.3] pour un système de preuve à la Hilbert, et [David et al., 2003, sec 1.3] et [Goubault-Larrecq et Mackie, 1997, fig. 2.2 et 6.1] pour des systèmes de déduction naturelle.

Le premier système de preuve de ces notes de révision est un système de calcul des séquents. Comparé aux systèmes de preuve à la Hilbert ou de déduction naturelle, son avantage est que la *recherche de preuve* y est fortement guidée, ce qui le rend relativement facile à employer sur des exemples – que le jury d'agrégation ne manquera pas de demander – et à implémenter. Il y a quantité de variantes du calcul des séquents pour la logique classique du premier ordre. Le système que nous étudions ici est un calcul *monolatère*, qui a l'avantage de ne comporter que peu de règles.

*Le calcul de la figure 2 ne contient ni la règle structurelle d'*échange*, qui est implicite parce que nous travaillons avec des multi-ensembles, ni la règle structurelle d'*affaiblissement*, qui est implicite dans la règle d'axiome* (ax) *(c.f. lemme 6.3), ni la règle structurelle de* contraction*, qui est implicite dans la règle de l'existentielle* (∃) *(c.f. lemme 6.6). La règle* (ax) *est souvent présentée sous une forme plus générale (c.f. lemme 6.4).*

$$\frac{}{\vdash \Gamma, \ell, \overline{\ell}}\ (\mathsf{ax}) \qquad \frac{\vdash \Gamma, \varphi \quad \vdash \Delta, \overline{\varphi}}{\vdash \Gamma, \Delta}\ (\mathsf{cut})$$

$$\frac{\vdash \Gamma, \varphi \quad \vdash \Gamma, \psi}{\vdash \Gamma, \varphi \wedge \psi}\ (\wedge) \qquad \frac{\vdash \Gamma, \varphi, \psi}{\vdash \Gamma, \varphi \vee \psi}\ (\vee)$$

$$\frac{\vdash \Gamma, \varphi[y/x]}{\vdash \Gamma, \forall x.\varphi}\ (\forall) \quad \text{où } y \notin \mathrm{fv}(\Gamma, \forall x.\varphi) \qquad \frac{\vdash \Gamma, \varphi[t/x], \exists x.\varphi}{\vdash \Gamma, \exists x.\varphi}\ (\exists) \quad \text{où } t \in T(\mathcal{F}, X)$$

Figure 2. Calcul des séquents monolatère.

Un *multi-ensemble* fini sur un ensemble E est formellement une fonction $m : E \to \mathbb{N}$ de domaine $\mathrm{dom}(m) \stackrel{\text{def}}{=} \{e \in E \mid m(e) > 0\}$ fini ; on peut aussi voir m comme une séquence finie de E^* modulo permutation.

Dans notre calcul, un *séquent* est un multi-ensemble fini de formules en forme normale négative noté $\vdash \Gamma$. La virgule dénote l'union de multi-ensembles : par exemple, Γ, Δ, φ dénote le multi-ensemble avec $\Gamma(\varphi) + \Delta(\varphi) + 1$ occurrences de la formule φ, et $\Gamma(\psi) + \Delta(\psi)$ occurrences de $\psi \neq \varphi$. La notion de variable libre est étendue aux multi-ensembles de formules : $\mathrm{fv}(\Gamma) \stackrel{\text{def}}{=} \bigcup_{\varphi \in \mathrm{dom}(\Gamma)} \mathrm{fv}(\varphi)$. On note le séquent vide $\vdash \bot$, où $\bot(\varphi) \stackrel{\text{def}}{=} 0$ pour toute formule φ.

Une règle du calcul des séquents permet de déduire un séquent *conclusion* d'un nombre fini de séquents *prémisses*. Chaque règle excepté la règle de coupure comprend une *formule principale* dans sa conclusion, indiquée en orange dans les règles de la figure 2 ; la règle (cut) élimine une *formule de coupure* indiquée en violet.

Un séquent $\vdash \Gamma$ est *prouvable*, noté $\vdash_{\mathbf{LK}} \Gamma$, s'il en existe une dérivation dans le système de la figure 2.

Exemple 5.4. La formule du buveur de l'exemple 4.1 est prouvable. Une dérivation possible du séquent correspondant (avec la formule principale indiquée en orange à chaque étape) est :

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash \neg B(x), B(y), \neg B(y), \forall y.B(y), \exists x.(\neg B(x) \vee \forall y.B(y))}\,(\mathsf{ax})}{\vdash \neg B(x), B(y), \neg B(y) \vee \forall y.B(y), \exists x.(\neg B(x) \vee \forall y.B(y))}\,(\vee)}{\vdash \neg B(x), B(y), \exists x.(\neg B(x) \vee \forall y.B(y))}\,(\exists)}{\vdash \neg B(x), \forall y.B(y), \exists x.(\neg B(x) \vee \forall y.B(y))}\,(\forall)}{\vdash \neg B(x) \vee \forall y.B(y), \exists x.(\neg B(x) \vee \forall y.B(y))}\,(\vee)}{\vdash \exists x.(\neg B(x) \vee \forall y.B(y))}\,(\exists)$$

Exemple 5.5. La condition $y \notin \mathrm{fv}(\forall x.\varphi)$ dans la définition de la règle (∀) est nécessaire. Sinon, comme $(B(x) \vee \neg B(y))[y/x] = B(y) \vee \neg B(y)$, on pourrait dériver

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash B(y), \neg B(y)}\,(\text{ax})}{\vdash B(y) \vee \neg B(y)}\,(\vee)}{\vdash \forall x.B(x) \vee \neg B(y)}\,(\forall)}{\vdash \forall y \forall x.B(x) \vee \neg B(y)}\,(\forall)$$

où l'avant-dernière étape est incorrecte. Cette formule n'est cependant pas valide : par exemple $D_I \stackrel{\text{def}}{=} \{a, b\}$ avec $B^I \stackrel{\text{def}}{=} \{a\}$ en fournit un contre-modèle puisque $I, [a/y, b/x] \not\models B(x) \vee \neg B(y)$.

6. Règles admissibles

Le calcul des séquents de la figure 2 peut être enrichi par plusieurs règles *admissibles*, qui n'affectent pas la prouvabilité, et récapitulées dans la figure 3 ci-dessous. Les preuves en sont établies *syntaxiquement*, par manipulation des dérivations, et préfigurent les techniques employées pour l'élimination des coupures que nous verrons en section 9.

$$\dfrac{\vdash \Gamma}{\vdash \Delta}\,(=_\alpha) \quad \text{où } \Gamma =_\alpha \Delta \qquad \dfrac{\vdash \Gamma}{\vdash \Gamma\sigma}\,(\mathsf{S}) \quad \text{où } \sigma \text{ est une substitution} \qquad \dfrac{\vdash \Gamma}{\vdash \Gamma, \Delta}\,(\mathsf{W}) \qquad \dfrac{}{\vdash \Gamma, \varphi, \overline{\varphi}}\,(\mathsf{ax}')$$

$$\dfrac{\vdash \Gamma, \varphi, \varphi}{\vdash \Gamma, \varphi}\,(\mathsf{C}) \qquad \dfrac{\vdash \Gamma, \varphi \vee \psi}{\vdash \Gamma, \varphi, \psi}\,(\mathsf{E}_\vee) \qquad \dfrac{\vdash \Gamma, \varphi \wedge \psi}{\vdash \Gamma, \varphi}\,(\mathsf{E}^1_\wedge) \qquad \dfrac{\vdash \Gamma, \varphi \wedge \psi}{\vdash \Gamma, \psi}\,(\mathsf{E}^2_\wedge)$$

$$\dfrac{\vdash \Gamma, \forall x.\varphi}{\vdash \Gamma, \varphi[y/x]}\,(\mathsf{E}_\forall) \quad \text{où } y \notin \mathrm{fv}(\Gamma, \forall x.\varphi) \qquad \dfrac{\vdash \Gamma, \exists x.\varphi}{\vdash \Gamma, \varphi[t/x], \exists x.\varphi}\,(\mathsf{E}_\exists) \quad \text{où } t \in T(\mathcal{F}, X)$$

Figure 3. Quelques règles admissibles du calcul des séquents.

On définit la *profondeur* $p(\pi)$ d'une dérivation π dans le calcul des séquents comme celle de l'arbre sous-jacent.

6.1. Substitution syntaxique.

Commençons par étendre l'α-congruence aux multi-ensembles : si $\varphi_i =_\alpha \psi_i$ pour tout $1 \leq i \leq n$, alors $\varphi_1, \ldots, \varphi_n =_\alpha \psi_1, \ldots, \psi_n$. Le lemme suivant montre l'admissibilité de la règle $(=_\alpha)$.

Lemme 6.1 (α-congruence syntaxique)**.** *Si $\vdash_{\mathbf{LK}} \Gamma$ par une dérivation π, alors pour tout multi-ensemble $\Delta =_\alpha \Gamma$, $\vdash_{\mathbf{LK}} \Delta$ par une dérivation de profondeur $p(\pi)$ et sans coupure si π était sans coupure.*

Démonstration. On montre que si $\vdash_{\mathbf{LK}} \Gamma, \varphi$ par une dérivation π, alors pour toute formule $\psi =_\alpha \varphi$, $\vdash_{\mathbf{LK}} \Gamma, \psi$ par une dérivation de profondeur $p(\pi)$ et sans coupure si π était sans coupure, ce qui démontrera le lemme par induction sur la taille du séquent.

On procède pour cela par récurrence sur la profondeur de la dérivation π de $\vdash_{\mathbf{LK}} \Gamma, \varphi$. On opère à une distinction de cas selon la dernière règle appliquée dans la dérivation π.

Si π se termine par une règle (R) autre que (cut) où φ n'est pas principale, alors par inspection des règles, on est nécessairement dans une situation

$$\dfrac{\begin{array}{ccc}\pi_1 & & \pi_k \\ \vdots & & \vdots \\ \vdash \Gamma_1, \varphi & \cdots & \vdash \Gamma_k, \varphi\end{array}}{\vdash \Gamma, \varphi}\,(\mathsf{R})$$

où $0 \leq k \leq 2$ ($k = 0$ correspondant au cas de la règle (ax)). Pour tout $1 \leq i \leq k$, par hypothèse de récurrence sur π_i, il existe une dérivation π_i' de $\vdash \Gamma_i, \psi$ de profondeur $p(\pi_i)$ et sans (cut) si π_i était sans (cut). On a donc la dérivation

$$\dfrac{\begin{array}{ccc}\pi_1' & & \pi_k' \\ \vdots & & \vdots \\ \vdash \Gamma_1, \psi & \cdots & \vdash \Gamma_k, \psi\end{array}}{\vdash \Gamma, \psi}\ (\mathrm{R})$$

Si π se termine par une règle (cut), alors sans perte de généralité, on est dans une situation

$$\dfrac{\begin{array}{cc}\pi_1 & \pi_2 \\ \vdots & \vdots \\ \vdash \Gamma, \varphi, \theta & \vdash \Delta, \overline{\theta}\end{array}}{\vdash \Gamma, \Delta, \varphi}\ (\mathrm{cut})$$

(l'autre situation étant celle où φ apparaissait dans la seconde prémisse). Par hypothèse de récurrence sur π_1, il existe une dérivation π_1' de $\vdash \Gamma, \psi, \theta$ de profondeur $p(\pi_1)$, et on a donc la dérivation

$$\dfrac{\begin{array}{cc}\pi_1' & \pi_2 \\ \vdots & \vdots \\ \vdash \Gamma, \psi, \theta & \vdash \Delta, \overline{\theta}\end{array}}{\vdash \Gamma, \Delta, \psi}\ (\mathrm{cut})$$

Il reste à traiter les cas où φ était principale dans la dernière règle appliquée par π. On procède par induction sur la congruence $\varphi =_\alpha \psi$. Les cas de la réflexivité, de la symétrie, et de la transitivité de $=_\alpha$ sont triviaux.

Cette preuve serait beaucoup plus difficile si (ax) *était remplacé par* (ax′) *: avec* (ax), $\ell =_\alpha \ell'$ *si et seulement si* $\ell = \ell'$ *et est donc traité par le cas de la réflexivité ; avec* (ax′), *chacun des cas ci-dessous peut provenir d'une application de l'axiome étendu.*

Cas de la congruence pour $\vee$: alors $\varphi = \varphi_1 \vee \varphi_2$ et $\psi = \psi_1 \vee \psi_2$ avec $\varphi_1 =_\alpha \psi_1$ et $\varphi_2 =_\alpha \psi_2$. Comme φ est principale la dernière règle appliquée dans π était ($\vee$), et donc $\vdash_{\mathbf{LK}} \Gamma, \varphi_1, \varphi_2$ par une dérivation de profondeur $p(\pi) - 1$. Par hypothèse de récurrence, $\vdash_{\mathbf{LK}} \Gamma, \psi_1, \varphi_2$ par une dérivation de profondeur $p(\pi) - 1$, et par une seconde application de l'hypothèse de récurrence, $\vdash_{\mathbf{LK}} \Gamma, \psi_1, \psi_2$ par une dérivation de profondeur $p(\pi) - 1$. Par une application de ($\vee$), $\vdash_{\mathbf{LK}} \Gamma, \psi_1 \vee \psi_2$.

Autres cas de congruence : similaires au cas précédent.

Cas de l'α-renommage pour $\forall$: alors $\varphi = \forall x.\varphi'$ et $\psi = \forall z.\varphi'[z/x]$ où $z \notin \mathrm{fv}(\forall x.\varphi')$ et $[z/x]$ est applicable à φ'. Comme φ est principale, la dernière règle appliquée dans π était ($\forall$), et donc $\vdash_{\mathbf{LK}} \Gamma, \varphi'[y/x]$ pour une variable $y \notin \mathrm{fv}(\Gamma, \forall x.\varphi')$ par une dérivation π' de profondeur $p(\pi) - 1$. Par l'équation (2), $\varphi'[y/x] =_\alpha \varphi'[z/x][y/z]$ puisque $z \notin \mathrm{fv}(\varphi')$. Par hypothèse de récurrence sur π', $\vdash_{\mathbf{LK}} \Gamma, \varphi'[z/x][y/z]$ par une dérivation de profondeur $p(\pi')$. Comme $y \notin \mathrm{fv}(\Gamma, \varphi'[z/x][y/z])$, on peut appliquer ($\forall$) et dériver $\vdash_{\mathbf{LK}} \Gamma, \forall z.\varphi'[z/x]$.

Cas de l'α-renommage pour $\exists$: alors $\varphi = \exists x.\varphi'$ et $\psi = \exists z.\varphi'[z/x]$ où $z \notin \mathrm{fv}(\exists x.\varphi')$ et $[z/x]$ est applicable à φ'. Comme φ est principale,la dernière règle appliquée dans π était ($\exists$), et donc $\vdash_{\mathbf{LK}} \Gamma, \varphi'[t/x]$ pour un terme $t \in T(\mathcal{F}, X)$ par une dérivation π' de profondeur $p(\pi) - 1$. Par l'équation (2), $\varphi'[t/x] =_\alpha \varphi'[z/x][t/z]$ puisque $z \notin \mathrm{fv}(\varphi')$. Par hypothèse de récurrence sur π', $\vdash_{\mathbf{LK}} \Gamma, \varphi'[z/x][t/z]$ par une dérivation de profondeur $p(\pi')$. Par une application de ($\exists$), $\vdash_{\mathbf{LK}} \Gamma, \exists z.\varphi'[z/x]$. □

Une substitution σ est *applicable* à un multi-ensemble Γ si elle est applicable à chacune des formules de $\mathrm{dom}(\Gamma)$. Si c'est le cas, on définit l'application d'une substitution σ à $\Gamma = \varphi_1, \ldots, \varphi_n$ comme une application simultanée de σ à toutes les occurrences de formules dans Γ : $\Gamma\sigma \stackrel{\text{def}}{=}$

$\varphi_1\sigma, \ldots, \varphi_n\sigma$. Comme d'habitude, nous faisons un abus de notation et nous écrivons « $\Gamma\sigma$ » pour un multi-ensemble $\Delta\sigma$ tel que $\Delta =_\alpha \Gamma$ et que σ soit applicable à Δ. Le lemme suivant montre l'applicabilité de la règle (S).

Lemme 6.2 (substitution syntaxique)**.** *Si $\vdash_{\mathbf{LK}} \Gamma$ par une dérivation π, alors pour toute substitution σ, $\vdash_{\mathbf{LK}} \Gamma\sigma$ par une dérivation de profondeur $p(\pi)$ et sans coupure si π était sans coupure.*

Démonstration. Par le lemme 6.1 d'α-congruence syntaxique, on peut supposer σ applicable à Γ. On procède par récurrence sur la profondeur de la dérivation π de $\vdash \Gamma$. Considérons pour cela la dernière règle employée dans cette dérivation.

Cas de (ax) : alors $\Gamma = \Gamma', \ell, \overline{\ell}$. Par la propriété 4.2, $\overline{\ell\sigma} = (\overline{\ell})\sigma$ et (ax) permet aussi de dériver le séquent $\vdash \Gamma'\sigma, \ell\sigma, (\overline{\ell})\sigma$ en une étape.

Cas de (cut) : alors $\Gamma = \Gamma', \Delta$ et $\vdash_{\mathbf{LK}} \Gamma', \varphi$ et $\vdash_{\mathbf{LK}} \overline{\varphi}, \Delta$ pour une formule φ. Par hypothèse de récurrence, on obtient des dérivations de même profondeur de $\vdash \Gamma'\sigma, \varphi\sigma$ et de $\vdash (\overline{\varphi})\sigma, \Delta\sigma$, et la propriété 4.2 montre que l'on peut appliquer (cut) pour obtenir une dérivation de profondeur $p(\pi)$ de $\vdash \Gamma'\sigma, \Delta\sigma$.

Cas de ($\vee$) et ($\wedge$) : similairement par hypothèse de récurrence.

Cas de ($\forall$) : alors $\Gamma = \Gamma', \forall x.\varphi$ et $\vdash_{\mathbf{LK}} \Gamma', \varphi[y/x]$ où $y \notin \mathrm{fv}(\Gamma', \forall x.\varphi)$ par une dérivation π' de profondeur $p(\pi) - 1$.

Par hypothèse de récurrence sur π', on a une dérivation de même profondeur $p(\pi)-1$ du séquent $\vdash \Gamma'[z/y]\sigma[u/z], \varphi[y/x][z/y]\sigma[u/z]$ où on a choisi $z \notin \mathrm{fv}(\Gamma'\sigma, \forall x.\varphi) \cup \mathrm{dom}(\sigma) \cup \mathrm{rv}_\varphi(\sigma)$ et $u \notin \mathrm{fv}(\Gamma'\sigma, \forall z.\varphi[z/x]\sigma)$ tels que les substitutions soient applicables. Comme $y \notin \mathrm{fv}(\Gamma'), \Gamma'[z/y] =_\alpha \Gamma'$ par l'équation (1), et comme $z \notin \mathrm{fv}(\Gamma'\sigma)$, de même $\Gamma'\sigma[u/z] =_\alpha \Gamma'\sigma$. Comme $y \notin \mathrm{fv}(\forall x.\varphi)$, par l'équation (2), $\varphi[y/x][z/y]\sigma[u/z] =_\alpha \varphi[z/x]\sigma[u/z]$. Par le lemme d'$\alpha$-congruence syntaxique, on a donc une dérivation de profondeur $p(\pi) - 1$ de $\vdash \Gamma'\sigma, \varphi[z/x]\sigma[u/z]$. Comme $u \notin \mathrm{fv}(\Gamma'\sigma, \forall z.\varphi[y/x][z/y]\sigma)$, on peut appliquer ($\forall$) pour obtenir une dérivation de profondeur $p(\pi)$ de $\vdash \Gamma'\sigma, \forall z.(\varphi[z/x]\sigma)$.

Finalement, comme $z \notin \mathrm{fv}(\forall x.\varphi)$, par α-renommage $(\forall x.\varphi)\sigma =_\alpha (\forall z.\varphi[z/x])\sigma = \forall z.(\varphi[z/x]\sigma)$ où σ est applicable puisque $z \notin \mathrm{dom}(\sigma) \cup \mathrm{rv}_\varphi(\sigma)$. Par le lemme 6.1 d'$\alpha$-congruence syntaxique, on a donc une dérivation de profondeur $p(\pi)$ de $\vdash \Gamma'\sigma, (\forall x.\varphi)\sigma$.

Cas de ($\exists$) : alors $\Gamma = \Gamma', \exists x.\varphi$ et $\vdash_{\mathbf{LK}} \Gamma', \varphi[t/x]$ par π' de profondeur $p(\pi) - 1$.

Par hypothèse de récurrence sur π', on a une dérivation de profondeur $p(\pi) - 1$ du séquent $\vdash \Gamma'\sigma, \varphi[t/x]\sigma$. Observons que si on choisit $z \notin \mathrm{fv}(\exists x.\varphi) \cup \mathrm{dom}(\sigma) \cup \mathrm{rv}_\varphi(\sigma)$, alors $\varphi[z/x]\sigma[t\sigma/z] =_\alpha \varphi[t/x]\sigma$. Par le lemme d'$\alpha$-congruence syntaxique, on a une dérivation de profondeur $p(\pi) - 1$ de $\vdash \Gamma'\sigma, \varphi[z/x]\sigma[t\sigma/z]$. On peut alors appliquer ($\exists$) pour dériver $\vdash \Gamma'\sigma, \exists z.(\varphi[z/x]\sigma)$. Comme $z \notin \mathrm{fv}(\exists x.\varphi)$, $\exists z.(\varphi[z/x]\sigma) =_\alpha (\exists x.\varphi)\sigma$ et donc $\vdash_{\mathbf{LK}} \Gamma'\sigma, (\exists x.\varphi)\sigma$ par une dérivation de profondeur $p(\pi)$ par une autre application du lemme d'α-congruence syntaxique. □

6.2. **Affaiblissement.** Bien que le système de la figure 2 ne comporte pas de manière explicite la règle d'affaiblissement (notée (W) pour « *weakening* »), celle-ci est admissible.

Lemme 6.3 (affaiblissement)**.** *Si $\vdash_{\mathbf{LK}} \Gamma$ par une dérivation π, alors pour tout multi-ensemble Δ, $\vdash_{\mathbf{LK}} \Gamma, \Delta$ par une dérivation de profondeur $p(\pi)$ et sans coupure si π était sans coupure.*

[David et al., 2003, lem. 7.3.1], [Goubault-Larrecq et Mackie, 1997, lem. 2.31].

Démonstration. Par récurrence sur la profondeur de la dérivation de $\vdash \Gamma$, on ajoute systématiquement Δ à tous les séquents. On fait pour cela une distinction de cas selon la dernière règle employée dans la dérivation π.

Cas de (ax) : alors $\Gamma = \Gamma', \ell, \overline{\ell}$ et on a aussi une dérivation de $\vdash \Gamma', \ell, \overline{\ell}, \Delta$ par (ax).

Cas de ($\forall$) : alors $\Gamma = \Gamma', \forall x.\varphi$ et $\vdash_{\mathbf{LK}} \Gamma', \varphi[y/x]$ par une dérivation de profondeur $p(\pi) - 1$ où $y \notin \mathrm{fv}(\Gamma', \forall x.\varphi)$.

Soit $z \notin \mathrm{fv}(\Gamma', \forall x.\varphi, \Delta)$. Par le lemme 6.2, $\vdash_{\mathbf{LK}} \Gamma'[z/y], \varphi[y/x][z/y]$ par une dérivation de profondeur $p(\pi)-1$. Comme $y \notin \mathrm{fv}(\Gamma')$, par l'équation (1), $\Gamma'[z/y] =_\alpha \Gamma'$, et comme $y \notin \mathrm{fv}(\forall x.\varphi)$, par l'équation (2), $\varphi[y/x][z/y] =_\alpha \varphi[z/x]$. Par le lemme 6.1 d'$\alpha$-congruence syntaxique on peut donc dériver $\vdash \Gamma', \varphi[z/x]$. Par hypothèse de récurrence, $\vdash_{\mathbf{LK}} \Gamma', \varphi[z/x], \Delta$ par une dérivation de profondeur $p(\pi) - 1$, et comme $z \notin \mathrm{fv}(\Gamma, \forall x.\varphi, \Delta)$, une application de (∀) permet de dériver $\vdash \Gamma', \forall x.\varphi, \Delta$.

Les autres cas découlent aisément de l'hypothèse de récurrence. □

Nous sommes maintenant en mesure de démontrer l'admissibilité de la règle d'axiome (ax′). Cet axiome est usuellement employé à la place de (ax) – qu'il subsume – dans les calculs des séquents de la littérature [David et al., 2003 ; Goubault-Larrecq et Mackie, 1997].

Lemme 6.4 (axiome étendu)**.** *Pour tout multi-ensemble* Γ *et toute formule* φ, $\vdash_{\mathbf{LK}} \Gamma, \varphi, \overline{\varphi}$.

Démonstration. On procède par récurrence sur $p(\varphi)$.

Cas de base ℓ : C'est immédiat par une application de l'axiome (ax).

Cas de $\varphi \wedge \psi$: Alors $\overline{\varphi \wedge \psi} = \overline{\varphi} \vee \overline{\psi}$. Par hypothèse de récurrence, $\vdash_{\mathbf{LK}} \Gamma, \varphi, \overline{\varphi}$ par une dérivation π_1 et $\vdash_{\mathbf{LK}} \Gamma, \psi, \overline{\psi}$ par une dérivation π_2. On a donc la dérivation

$$\dfrac{\dfrac{\dfrac{\begin{array}{c}\pi_1\\ \vdots \\ \vdash \Gamma, \varphi, \overline{\varphi}\end{array}}{\vdash \Gamma, \varphi, \overline{\varphi}, \overline{\psi}}\,(\mathsf{W}) \qquad \dfrac{\begin{array}{c}\pi_2\\ \vdots \\ \vdash \Gamma, \psi, \overline{\psi}\end{array}}{\vdash \Gamma, \psi, \overline{\psi}, \overline{\varphi}}\,(\mathsf{W})}{\vdash \Gamma, \varphi \wedge \psi, \overline{\varphi}, \overline{\psi}}\,(\wedge)}{\vdash \Gamma, \varphi \wedge \psi, \overline{\varphi} \vee \overline{\psi}}\,(\vee)$$

Cas de $\varphi \vee \psi$: Similaire au cas précédent.

Cas de $\forall x.\varphi$: Alors $\overline{\forall x.\varphi} = \exists x.\overline{\varphi}$. Par hypothèse de récurrence et la propriété 4.2, on a $\vdash_{\mathbf{LK}} \Gamma, \varphi[y/x], \overline{\varphi}[y/x]$ pour $y \notin \mathrm{fv}(\Gamma, \forall x.\varphi)$ par une dérivation π. On a donc la dérivation

$$\dfrac{\dfrac{\dfrac{\begin{array}{c}\pi\\ \vdots \\ \vdash \Gamma, \varphi[y/x], \overline{\varphi}[y/x]\end{array}}{\vdash \Gamma, \varphi[y/x], \overline{\varphi}[y/x], \exists x.\overline{\varphi}}\,(\mathsf{W})}{\vdash \Gamma, \varphi[y/x], \exists x.\overline{\varphi}}\,(\exists)}{\vdash \Gamma, \forall x.\varphi, \exists x.\overline{\varphi}}\,(\forall)$$

Cas de $\exists x.\varphi$: Similaire au cas précédent. □

6.3. **Inversibilité.** Une règle de déduction est *inversible* si, quand il existe une dérivation (que l'on supposera sans coupure grâce au théorème 9.2) de son séquent conclusion, alors il existe une dérivation sans coupure de chacun de ses séquents prémisses. La règle (ax) est bien sûr inversible, mais ce qui est plus intéressant, c'est que toutes les autres règles sauf (cut) le sont aussi. Cela signifie que dans une recherche de preuve sans coupure, on peut appliquer ces règles de manière gloutonne sans faire de *backtrack* – donc sans avoir à mémoriser de points de choix – et que si l'on arrive à un séquent pour lequel aucune règle ne s'applique, alors c'est qu'il n'y a *pas* de preuve. Cela démontre aussi l'admissibilité des règles d'élimination ($\mathsf{E}_\vee$), ($\mathsf{E}^1_\wedge$), ($\mathsf{E}^2_\wedge$), ($\mathsf{E}_\forall$) et ($\mathsf{E}_\exists$) de la figure 3.

Lemme 6.5 (inversibilité)**.** *Les règles du calcul des séquents sauf* (cut) *sont inversibles.*

Démonstration. On ne traite ici que les règles (∃) et (∀), les autres étant traitées dans les notes de la leçon **916**.

Pour la règle ($\exists$) **:** s'il y a une preuve sans coupure de $\vdash \Gamma, \exists x.\varphi$, alors par affaiblissement il en existe aussi une de $\vdash \Gamma, \exists x.\varphi, \varphi[t/x]$.

Pour la règle ($\forall$) **:** soit π une dérivation sans coupure du séquent $\vdash \Gamma, \forall x.\varphi$. On montre par récurrence sur la profondeur de π que pour toute variable $y \notin \mathrm{fv}(\Gamma, \forall x.\varphi)$, $\vdash_{\mathbf{LK}} \Gamma, \varphi[y/x]$.

— Si π se termine par ($\forall$) où $\forall x.\varphi$ est principale, alors évidemment il existe une sous-dérivation de π de la prémisse $\vdash \Gamma, \varphi[z/x]$ pour une certaine variable $z \notin \mathrm{fv}(\Gamma, \forall x.\varphi)$. Par le lemme 6.2 de substitution syntaxique, il existe aussi une dérivation du séquent $\vdash \Gamma[y/z], \varphi[z/x][y/z]$ pour tout $y \notin \mathrm{fv}(\Gamma, \forall x.\varphi)$, et comme $z \notin \mathrm{fv}(\Gamma, \forall x.\varphi)$, par les équations (1) et (2) ce dernier séquent est α-congruent à $\vdash \Gamma, \varphi[y/x]$ et est dérivable par le lemme d'α-congruence syntaxique.

— Si π se termine par ($\forall$) où $\forall x.\varphi$ n'est pas principale, alors $\Gamma = \Gamma', \forall z.\psi$ et on a la dérivation

$$\dfrac{\begin{array}{c}\pi' \\ \vdots \\ \vdash \Gamma', \psi[v/z], \forall x.\varphi\end{array}}{\vdash \Gamma', \forall z.\psi, \forall x.\varphi}\ (\forall)$$

où $v \notin \mathrm{fv}(\Gamma', \forall z.\psi, \forall x.\varphi)$.
On prend une variable fraîche $w \notin \mathrm{fv}(\Gamma', \psi[v/z], \forall x.\varphi)$; par hypothèse de récurrence appliquée à π', il existe une dérivation de $\vdash \Gamma', \psi[v/z], \varphi[w/x]$.
Comme $v \notin \mathrm{fv}(\Gamma', \forall z.\psi, \varphi[w/x])$, on peut appliquer ($\forall$) avec $\forall z.\psi$ comme formule principale pour obtenir $\vdash \Gamma', \forall z.\psi, \varphi[w/x]$.
Par le lemme de substitution syntaxique, pour toute variable $y \notin \mathrm{fv}(\Gamma', \forall z.\psi, \forall x.\varphi)$ on a donc aussi une dérivation du séquent $\vdash \Gamma'[y/w], (\forall z.\psi)[y/w], \varphi[w/x][y/w]$. Comme $w \notin \mathrm{fv}(\Gamma', \psi[v/z], \forall x.\varphi)$, $w \notin \mathrm{fv}(\Gamma', \forall z.\psi, \forall x.\varphi)$ et donc par les équations (1) et (2) ce dernier séquent est α-congruent à $\vdash \Gamma', \forall z.\psi, \varphi[y/x]$ et est dérivable par le lemme d'α-congruence syntaxique.

— Sinon π se termine par une règle (R) autre que ($\forall$) où $\forall x.\varphi$ n'est pas principale. Par inspection des règles, on est nécessairement dans une situation

$$\dfrac{\begin{array}{c}\pi_1 \\ \vdots \\ \vdash \Gamma_1, \forall x.\varphi\end{array} \quad \cdots \quad \begin{array}{c}\pi_k \\ \vdots \\ \vdash \Gamma_k, \forall x.\varphi\end{array}}{\vdash \Gamma, \forall x.\varphi}\ (\mathrm{R})$$

où $0 \leq k \leq 2$ ($k = 0$ correspondant au cas de la règle (ax)). Pour tout $y \notin \mathrm{fv}(\Gamma, \forall x.\varphi)$ et tout $1 \leq i \leq k$, par hypothèse de récurrence sur π_i, il existe une dérivation sans coupure π_i' de $\vdash \Gamma_i, \varphi[y/x]$. On a donc la dérivation

$$\dfrac{\begin{array}{c}\pi_1' \\ \vdots \\ \vdash \Gamma_1, \varphi[y/x]\end{array} \quad \cdots \quad \begin{array}{c}\pi_k' \\ \vdots \\ \vdash \Gamma_k, \varphi[y/x]\end{array}}{\vdash \Gamma, \varphi[y/x]}\ (\mathrm{R})$$

□

6.4. **Contraction.** L'admissibilité de la règle (C) est un peu plus délicate à démontrer, et nous allons utiliser une stratégie de démonstration similaire à celle de l'élimination des coupures. Le lemme suivant, combiné au théorème 9.2, montre son admissibilité.

Lemme 6.6 (contraction)**.** *Si* $\vdash_{\mathbf{LK}} \Gamma, \varphi, \varphi$ *par une dérivation sans coupure, alors* $\vdash_{\mathbf{LK}} \Gamma, \varphi$ *par une dérivation sans coupure.*

Démonstration. Le *rang de contraction* d'une dérivation π d'un séquent $\vdash \Gamma, \varphi, \varphi$ est le couple $(p(\varphi), p(\pi))$ dans $\mathbb{N}^2$. On ordonne les rangs de contraction par l'ordre lexicographique $<_{\mathrm{lex}}$ sur $\mathbb{N}^2$; il s'agit d'un ordre bien fondé.

💬 *L'intérêt de définir le calcul des séquents sans la règle de contraction est bien sûr de diminuer le nombre de règles. Mais cela simplifie aussi la preuve de l'élimination des coupures, qui nécessite sinon l'ajout d'une règle (mix). Enfin, dans le cas propositionnel, cela permet de démontrer une propriété de* profondeur polynomiale *des dérivations sans coupure, et de déduire que la recherche de preuve se fait en* coNP*, voir la leçon* ***916****.*

On procède par induction bien fondée sur le rang de la dérivation sans coupure π de $\vdash \Gamma, \varphi, \varphi$. Si φ n'est pas principale dans la dernière règle (R) appliquée dans π, alors par inspection des règles, cette dérivation π est nécessairement de la forme

$$\dfrac{\begin{array}{ccc}\pi_1 & & \pi_k \\ \vdots & & \vdots \\ \vdash \Gamma_1, \varphi, \varphi & \cdots & \vdash \Gamma_k, \varphi, \varphi\end{array}}{\vdash \Gamma, \varphi, \varphi}\ (\mathsf{R})$$

où $0 \leq k \leq 2$ ($k = 0$ correspondant à une règle (ax)). Comme $(p(\varphi), p(\pi_i)) <_{\text{lex}} (p(\varphi), p(\pi))$ pour tout $1 \leq i \leq k$, on peut appliquer l'hypothèse d'induction à chaque π_i pour obtenir des dérivations π_i' de $\vdash \Gamma_i, \varphi$. Encore par inspection des règles, on obtient une nouvelle dérivation

$$\dfrac{\begin{array}{ccc}\pi_1' & & \pi_k' \\ \vdots & & \vdots \\ \vdash \Gamma_1, \varphi & \cdots & \vdash \Gamma_k, \varphi\end{array}}{\vdash \Gamma, \varphi}\ (\mathsf{R})$$

Sinon, si φ est principale dans la dernière règle appliquée dans π, on fait une distinction de cas selon cette règle.

Cas de (ax) : alors $\varphi = \ell$ et $\Gamma = \Gamma', \overline{\ell}$, et on a aussi une dérivation de $\vdash \Gamma', \overline{\ell}, \ell$ par (ax).

Cas de ($\vee$) : alors $\varphi = \varphi' \vee \psi$ et π est de la forme

$$\dfrac{\begin{array}{c}\pi_1 \\ \vdots \\ \vdash \Gamma, \varphi', \psi, \varphi' \vee \psi\end{array}}{\vdash \Gamma, \varphi' \vee \psi, \varphi' \vee \psi}\ (\vee)$$

Par le lemme 6.5 appliqué à π_1, on a aussi une dérivation π_1' de $\vdash \Gamma, \varphi', \psi, \varphi', \psi$. Comme $(p(\varphi'), p(\pi_1')) <_{\text{lex}} (p(\varphi' \vee \psi), p(\pi))$, par hypothèse d'induction, on a une dérivation π_1'' de $\vdash \Gamma, \varphi', \psi, \psi$. Comme $(p(\psi), p(\pi_1'')) <_{\text{lex}} (p(\varphi' \vee \psi), p(\pi))$, par hypothèse d'induction, on a une dérivation de $\vdash \Gamma, \varphi', \psi$. Par une application de ($\vee$), on obtient alors une dérivation de $\vdash \Gamma, \varphi' \vee \psi$.

Cas de ($\wedge$) : alors $\varphi = \varphi' \wedge \psi$ et π est de la forme

$$\dfrac{\begin{array}{cc}\pi_1 & \pi_2 \\ \vdots & \vdots \\ \vdash \Gamma, \varphi', \varphi' \wedge \psi & \vdash \Gamma, \psi, \varphi' \wedge \psi\end{array}}{\vdash \Gamma, \varphi' \wedge \psi, \varphi' \wedge \psi}\ (\wedge)$$

Par le lemme 6.5 appliqué à π_1 et π_2, on a aussi une dérivation π_1' de $\vdash \Gamma, \varphi', \varphi'$ et une dérivation π_2' de $\vdash \Gamma, \psi, \psi$. Comme $(p(\varphi'), p(\pi_1')) <_{\text{lex}} (p(\varphi' \wedge \psi), p(\pi))$ et $(p(\psi), p(\pi_2')) <_{\text{lex}} (p(\varphi' \wedge \psi), p(\pi))$, par hypothèse d'induction, on a des dérivations de $\vdash \Gamma, \varphi'$ et de $\vdash \Gamma, \psi$. Par une application de ($\wedge$), on obtient alors une dérivation de $\vdash \Gamma, \varphi' \wedge \psi$.

Cas de ($\forall$) : alors $\varphi = \forall x.\psi$ et π est de la forme

$$\dfrac{\begin{array}{c}\pi_1 \\ \vdots \\ \vdash \Gamma, \psi[y/x], \forall x.\psi\end{array}}{\vdash \Gamma, \forall x.\psi, \forall x.\psi}\ (\forall)$$

où $y \notin \text{fv}(\Gamma, \forall x.\psi)$. Par le lemme 6.5 appliqué à π_1, on a aussi une dérivation π_1' de $\vdash \Gamma, \psi[y/x], \psi[z/x]$ où $z \notin \{y\} \cup \text{fv}(\Gamma, \forall x.\psi)$. Par le lemme 6.2, on a aussi une dérivation π_1'' de $\vdash \Gamma[y/z], \psi[y/x][y/z], \psi[z/x][y/z]$. Comme $z \notin \{y\} \cup \text{fv}(\Gamma, \forall x.\psi)$, par les équations (1) et (2), ce dernier séquent est α-congruent à $\vdash \Gamma, \psi[y/x], \psi[y/x]$ et donc dérivable par le lemme d'α-congruence syntaxique. Comme $(p(\psi[y/x]), p(\pi_1'')) <_{\text{lex}} (p(\forall x.\psi), p(\pi))$, par hypothèse d'induction on obtient une dérivation de $\vdash \Gamma, \psi[y/x]$. Comme $y \notin \text{fv}(\Gamma, \forall x.\psi)$, on peut appliquer ($\forall$) et on obtient une dérivation de $\vdash \Gamma, \forall x.\psi$.

Cas de ($\exists$) **:** alors $\varphi = \exists x.\psi$ et π est de la forme

$$\dfrac{\begin{array}{c}\pi_1 \\ \vdots \\ \vdash \Gamma, \psi[t/x], \exists x.\psi, \exists x.\psi\end{array}}{\vdash \Gamma, \exists x.\psi, \exists x.\psi}\ (\forall)$$

où $t \in T(\mathcal{F}, X)$. Comme $(p(\exists x.\psi), p(\pi_1)) <_{\text{lex}} (p(\exists x.\psi), p(\pi))$, par hypothèse d'induction sur π_1 on obtient une dérivation de $\vdash \Gamma, \psi[t/x], \exists x.\psi$ à laquelle on applique ($\exists$) pour obtenir $\vdash \Gamma, \exists x.\psi$. □

7. Correction

Un séquent $\vdash \Gamma$ *satisfait* une interprétation I sous une valuation ρ, noté $I, \rho \vDash \Gamma$, s'il existe une formule $\vartheta \in \text{dom}(\Gamma)$ telle que $I, \rho \vDash \vartheta$; la formule ϑ en question est appelée *formule témoin*. Un séquent $\vdash \Gamma$ est *valide*, noté $\vDash \Gamma$, si $I, \rho \vDash \Gamma$ pour tous I et ρ. La *correction* d'un système de preuve consiste à montrer que toute formule prouvable est valide.

Théorème 7.1 (correction). *Si* $\vdash_{\mathbf{LK}} \Gamma$*, alors* $\vDash \Gamma$.

⚠ *Cette preuve manque quelque peu d'intérêt pour servir de développement à l'agrégation.*

Démonstration. On procède par induction structurelle sur une dérivation de $\vdash \Gamma$, en montrant pour chaque règle du calcul des séquents que si les prémisses sont valides, alors la conclusion l'est aussi. On ne traitera ici que les règles de quantification, les autres cas ayant été traités dans les notes de la leçon **916**.

Pour ($\forall$) **:** supposons la prémisse $\vdash \Gamma, \varphi[y/x]$ valide pour $y \notin \text{fv}(\Gamma, \forall x.\varphi)$. Prenons (I, ρ) arbitraire et montrons que $I, \rho \vDash \Gamma, \forall x.\varphi$.

Comme la prémisse est valide, pour tout $e \in D_I$, $I, \rho[e/y] \vDash \Gamma, \varphi[y/x]$, donc il existe une formule témoin $\vartheta_e \in \text{dom}(\Gamma) \cup \{\varphi[y/x]\}$ telle que $I, \rho[e/y] \vDash \vartheta_e$

S'il existe $e \in D_I$ tel que $\vartheta_e \in \text{dom}(\Gamma)$, alors comme $y \notin \text{fv}(\Gamma)$ et donc $y \notin \text{fv}(\vartheta_e)$, par la propriété 2.2, on a aussi $I, \rho \vDash \vartheta_e$. Donc $I, \rho \vDash \Gamma, \forall x.\varphi$.

Sinon, $\vartheta_e = \varphi[y/x]$ pour tout $e \in D_I$, c'est-à-dire $I, \rho[e/y] \vDash \varphi[y/x]$ pour tout $e \in D_I$, et donc $I, \rho \vDash \forall x.\varphi$. Donc $I, \rho \vDash \Gamma, \forall x.\varphi$.

Pour ($\exists$) **:** supposons la prémisse valide. Prenons (I, ρ) arbitraire et montrons que $I, \rho \vDash \Gamma, \exists x.\varphi$.

Si la formule témoin de $I, \rho \vDash \Gamma, \varphi[t/x], \exists x.\varphi$ est dans $\{\exists x.\varphi\} \cup \text{dom}(\Gamma)$, alors elle peut aussi servir de témoin pour $I, \rho \vDash \Gamma, \exists x.\varphi$. Si la formule témoin est $\varphi[t/x]$, c'est-à-dire si $I, \rho \vDash \varphi[t/x]$, alors $I, \rho[[\![t]\!]_\rho^I/x] \vDash \varphi$ par le lemme 3.5, et donc $I, \rho \vDash \exists x.\varphi$, qui peut servir de témoin pour $I, \rho \vDash \Gamma, \exists x.\varphi$. □

8. Complétude

La complétude des systèmes de preuves en logique du premier ordre, c'est-à-dire que toute formule valide est dérivable dans le système de preuve en question, se montre par contraposée : un séquent $\vdash \Gamma$ non prouvable n'est pas valide, ce qui par définition signifie qu'il a un *contre-modèle*, c'est-à-dire une interprétation I et une valuation ρ telles que $I, \rho \nvDash \varphi$ pour toute formule $\varphi \in \text{dom}(\Gamma)$.

⚠ *La preuve de complétude du calcul des séquents de [Goubault-Larrecq et Mackie, 1997, pp. 215–219], basée sur une version syntaxique du théorème de Herbrand, est incorrecte.*

[Ebbinghaus et al., 1994, ch. V] pour la construction de Henkin pour le calcul des séquents, et [David et al., 2003, ch. 2] et [Lassaigne et de Rougemont, 2004, sec. 4.3] dans le cadre de la déduction naturelle.
[Fitting, 1996, sec. 3.5 et 5.6]

Ce contre-modèle est habituellement obtenu par la « construction de Henkin ». La construction présentée ici en est une variante, qui repose plutôt sur les travaux de Hintikka sur les tableaux en logique du premier ordre. Nous la voyons ici dans le cas du calcul des séquents monolatère de la figure 2, mais l'approche se généralise aisément à d'autres systèmes de preuve.

8.1. **Lemme de Hintikka.** Le principe de la construction est de considérer des ensembles de formules suffisamment « saturés ». Comme l'objectif est de construire un contre-modèle plutôt qu'un modèle, la définition qui suit est duale de la définition usuelle pour les tableaux.

Définition 8.1 (ensemble de Hintikka)**.** Un ensemble E de formules en forme normale négative est *dualement saturé* si

(1) si E contient $\varphi \vee \psi$, alors E contient φ et ψ,

(2) si E contient $\varphi \wedge \psi$, alors E contient φ ou ψ,

(3) si E contient $\exists x.\varphi$, alors E contient $\varphi[t/x]$ pour tout terme $t \in T(\mathcal{F}, X)$, et

(4) si E contient $\forall x.\varphi$, alors E contient $\varphi[y/x]$ pour une variable y.

Un ensemble est *cohérent* s'il ne contient pas une formule atomique α et sa négation $\neg\alpha$. Un *ensemble de Hintikka* est un ensemble dualement saturé cohérent.

Lemme 8.2 (Hintikka)**.** *Si H est un ensemble de Hintikka, alors il existe une interprétation I et une valuation ρ telles que, pour toute formule $\varphi \in H$, $I, \rho \not\models \varphi$.*

On peut remarquer que ce modèle est presque une structure de Herbrand, si ce n'est que son domaine est $T(\mathcal{F}, X)$ et non $T(\mathcal{F})$.

Démonstration. L'interprétation I a pour domaine $T(\mathcal{F} \cup X)$ l'ensemble des termes clos sur l'alphabet $\mathcal{F} \cup X$, où les variables de X sont considérées comme des constantes d'arité zéro. L'interprétation d'un symbole de fonction $f \in \mathcal{F}_m$ est $f^I(t_1, \ldots, t_m) \stackrel{\text{def}}{=} f(t_1, \ldots, t_m)$. L'interprétation d'un symbole de relation $R \in \mathcal{P}_m$ est $R^I \stackrel{\text{def}}{=} \{(t_1, \ldots, t_m) \mid \neg R(t_1, \ldots, t_m) \in H\}$. La valuation ρ est l'identité sur X.

On vérifie aisément que pour tout terme $t \in T(\mathcal{F}, X)$, $[\![t]\!]^I_\rho = t$. Cela implique que pour une substitution $[t/x]$ où t est vu comme un terme de $T(\mathcal{F}, X)$, la valuation $[t/x]\rho$ n'est autre que $\rho[t/x]$ où t est vu comme un élément du domaine D_I.

Montrons par induction structurelle sur $\varphi \in H$ que $I, \rho \not\models \varphi$.

— Pour un littéral positif $R(t_1, \ldots, t_m) \in H$, comme H est cohérent, $\neg R(t_1, \ldots, t_m) \notin H$ et donc $([\![t_1]\!]^I_\rho, \ldots, [\![t_m]\!]^I_\rho) = (t_1, \ldots, t_m) \notin R^I$: on a bien $I, \rho \not\models R(t_1, \ldots, t_m)$.

— Pour un littéral négatif $\neg R(t_1, \ldots, t_m) \in H$, $([\![t_1]\!]^I_\rho, \ldots, [\![t_m]\!]^I_\rho) = (t_1, \ldots, t_m) \in R^I$: on a bien $I, \rho \not\models \neg R(t_1, \ldots, t_m)$.

(1) Pour une formule $\varphi \vee \psi \in H$, comme H est dualement saturé il contient φ et ψ, donc par hypothèse d'induction $I, \rho \not\models \varphi$ et $I, \rho \not\models \psi$: on a bien $I, \rho \not\models \varphi \vee \psi$.

(2) Pour une formule $\varphi \wedge \psi \in H$, comme H est dualement saturé il contient φ ou ψ, donc par hypothèse d'induction $I, \rho \not\models \varphi$ ou $I, \rho \not\models \psi$: on a bien $I, \rho \not\models \varphi \wedge \psi$.

(3) Pour une formule $\exists x.\varphi \in H$, comme H est dualement saturé il contient $\varphi[t/x]$ pour tout terme t, donc par hypothèse d'induction $I, \rho \not\models \varphi[t/x]$ pour tout terme t et par le lemme 3.5, $I, \rho[t/x] \not\models \varphi$ pour tout $t \in D_I$: on a bien $I, \rho \not\models \exists x.\varphi$.

(4) Pour une formule $\forall x.\varphi \in H$, comme H est dualement saturé il contient $\varphi[y/x]$ pour une variable y, donc par hypothèse d'induction $I, \rho \not\models \varphi[y/x]$ et par le lemme 3.5, $I, \rho[y/x] \not\models \varphi$ où $y \in D_I$: on a bien $I, \rho \not\models \forall x.\varphi$. □

8.2. **Théorème de complétude.** Un ensemble E de formules en forme normale négative est *prouvable* s'il existe un multi-ensemble fini Γ avec $\mathrm{dom}(\Gamma) \subseteq E$ tel que $\vdash_{\mathbf{LK}} \Gamma$ par une dérivation sans coupure, et *non prouvable* sinon.

Lemme 8.3 (propagation)**.** *Soit E un ensemble non prouvable :*

(1) si E contient $\varphi \vee \psi$, alors $E \cup \{\varphi, \psi\}$ est non prouvable,

(2) si E contient $\varphi \wedge \psi$, alors $E \cup \{\varphi\}$ est non prouvable ou $E \cup \{\psi\}$ est non prouvable,

(3) si E contient $\exists x.\varphi$, alors $E \cup \{\varphi[t/x]\}$ est non prouvable pour tout terme $t \in T(\mathcal{F}, X)$, et

(4) si E contient $\forall x.\varphi$, alors $E \cup \{\varphi[y/x]\}$ est non prouvable pour toute variable $y \notin \mathrm{fv}(E)$.

Démonstration. On procède dans chaque cas par l'absurde.

(1) Si $E \cup \{\varphi, \psi\}$ est prouvable mais E n'est pas prouvable, alors nécessairement $\{\varphi, \psi\} \not\subseteq E$ et cette dérivation doit faire intervenir φ ou ψ. Sans perte de généralité, supposons $\vdash_{\mathbf{LK}} \Gamma, \varphi^m, \psi^n$ pour un multi-ensemble fini Γ avec $\mathrm{dom}(\Gamma) \subseteq E$ et $m + n > 0$. Par affaiblissement, on a aussi $\vdash_{\mathbf{LK}} \Gamma, \varphi^{\max(m,n)}, \psi^{\max(m,n)}$. Par $\max(m, n)$ applications de $(\vee)$, on a donc $\vdash_{\mathbf{LK}} \Gamma, (\varphi \vee \psi)^{\max(m,n)}$ et E prouvable, contradiction.

(2) Si $E \cup \{\varphi\}$ et $E \cup \{\psi\}$ sont tous les deux prouvables mais E n'est pas prouvable, alors nécessairement $\varphi \notin E$ et $\psi \notin E$ et ces dérivations doivent faire intervenir φ et ψ : $\vdash_{\mathbf{LK}} \Gamma, \varphi^m$ et $\vdash_{\mathbf{LK}} \Delta, \psi^n$ pour des multi-ensemble finis Γ et Δ avec $\mathrm{dom}(\Gamma) \cup \mathrm{dom}(\Delta) \subseteq E$ et $m, n > 0$. Par affaiblissement et contraction, on a aussi $\vdash_{\mathbf{LK}} \Gamma, \Delta, \varphi$ et $\vdash_{\mathbf{LK}} \Gamma, \Delta, \psi$. Par une application de $(\wedge)$, on a donc $\vdash_{\mathbf{LK}} \Gamma, \Delta, \varphi \wedge \psi$ et E prouvable, contradiction.

(3) Si $E \cup \{\varphi[t/x]\}$ est prouvable pour un terme t mais E n'est pas prouvable, alors $\varphi[t/x] \notin E$ et on a $\vdash_{\mathbf{LK}} \Gamma, (\varphi[t/x])^m$ pour un multi-ensemble fini Γ avec $\mathrm{dom}(\Gamma) \subseteq E$ et $m > 0$. Par affaiblissement, $\vdash_{\mathbf{LK}} \Gamma, (\varphi[t/x])^m, (\exists x.\varphi)^m$. Par m applications de $(\exists)$, on a donc $\vdash_{\mathbf{LK}} \Gamma, (\exists x.\varphi)^m$ et E prouvable, contradiction.

(4) Si $E \cup \{\varphi[y/x]\}$ est prouvable pour une variable $y \notin \mathrm{fv}(E)$, alors $\varphi[y/x] \notin E$ et $\vdash_{\mathbf{LK}} \Gamma, (\varphi[y/x])^m$ pour un multi-ensemble fini Γ avec $\mathrm{dom}(\Gamma) \subseteq E$ et $m > 0$. Par contraction, on a aussi $\vdash_{\mathbf{LK}} \Gamma, \varphi[y/x]$, et comme $\mathrm{dom}(\Gamma) \subseteq E$ et $\forall x.\varphi \in E$, $y \notin \mathrm{fv}(\Gamma, \forall x.\varphi)$, donc on peut appliquer $(\forall)$ pour dériver $\vdash_{\mathbf{LK}} \Gamma, \forall x.\varphi : E$ est prouvable, contradiction. □

Le cas (4) est le seul où l'admissibilité de la contraction soit vraiment utile.

Le cœur de la preuve du théorème de complétude est le lemme suivant :

Lemme 8.4 (saturation)**.** *Tout ensemble fini non prouvable est inclus dans un ensemble de* Hintikka.

Démonstration. Appelons une *tâche* une formule qui n'est pas de la forme $\exists x.\varphi$ ou une paire $(\exists x.\varphi, t)$ où $t \in T(\mathcal{F}, X)$. Comme il n'y a qu'un nombre dénombrable de formules φ et de termes $t \in T(\mathcal{F}, X)$, il y a un nombre dénombrable de tâches. Fixons une énumération $\theta_0, \theta_1, \ldots$ de toutes les tâches, telle que chaque tâche apparaisse infiniment souvent.

La preuve du lemme de saturation correspond intuitivement à la construction d'une branche infinie d'une recherche de preuve pour un séquent non prouvable.

Soit E_0 un ensemble fini non prouvable. On construit une séquence croissante d'ensembles finis non prouvables $E_0 \subseteq E_1 \subseteq \cdots$. Étant donné E_n, on construit E_{n+1} en utilisant θ_n la $n$ième tâche.

(1) Si $\theta_n = \varphi \vee \psi$: si $\varphi \vee \psi \in E_n$, alors par le lemme 8.3.(1), $E_{n+1} \stackrel{\text{def}}{=} E_n \cup \{\varphi, \psi\}$ est non prouvable ; sinon on pose $E_{n+1} \stackrel{\text{def}}{=} E_n$.

(2) Si $\theta_n = \varphi \wedge \psi$: si $\varphi \wedge \psi \in E_n$, alors par le lemme 8.3.(2), $E_n \cup \{\varphi\}$ ou $E_n \cup \{\psi\}$ est non prouvable et on pose $E_{n+1} \stackrel{\text{def}}{=} E_n \cup \{\varphi\}$ dans le premier cas et $E_{n+1} \stackrel{\text{def}}{=} E_n \cup \{\psi\}$ dans le second ; sinon on pose $E_{n+1} \stackrel{\text{def}}{=} E_n$.

(3) Si $\theta_n = (\exists x.\varphi, t)$: si $\exists x.\varphi \in E_n$, alors par le lemme 8.3.(3), $E_{n+1} \stackrel{\text{def}}{=} E_n \cup \{\varphi[t/x]\}$ est non prouvable ; sinon on pose $E_{n+1} \stackrel{\text{def}}{=} E_n$.

(4) Si $\theta_n = \forall x.\varphi$: si $\forall x.\varphi \in E_n$, alors il existe $y \notin \mathrm{fv}(E_n)$ puisque ce dernier est fini, et par le lemme 8.3.(4) $E_{n+1} \stackrel{\text{def}}{=} E_n \cup \{\varphi[y/x]\}$ est non prouvable ; sinon on pose $E_{n+1} \stackrel{\text{def}}{=} E_n$.

La restriction à un ensemble initial fini est importante pour ce cas (4).

Définissons $H \stackrel{\text{def}}{=} \bigcup_{n \in \mathbb{N}} E_n$. Il reste à montrer que H est dualement saturé et cohérent. Commençons par vérifier qu'il est dualement saturé :

(1) Si H contient $\varphi \vee \psi$, alors il existe $m \in \mathbb{N}$ tel que $\varphi \vee \psi \in E_m$ et $n > m$ tel que $\theta_n = \varphi \vee \psi$, donc E_{n+1} et donc H contiennent φ et ψ.

(2) Si H contient $\varphi \wedge \psi$, alors il existe $m \in \mathbb{N}$ tel que $\varphi \wedge \psi \in E_m$ et $n > m$ tel que $\theta_n = \varphi \wedge \psi$, donc E_{n+1} et donc H contiennent φ ou ψ.

(3) Si H contient $\exists x.\varphi$ et $t \in T(\mathcal{F}, X)$, alors il existe $m \in \mathbb{N}$ tel que $\exists x.\varphi \in E_m$ et $n > m$ tel que $\theta_n = (\exists x.\varphi, t)$, donc E_{n+1} et donc H contiennent $\varphi[t/x]$.

(4) Si H contient $\forall x.\varphi$, alors il existe $m \in \mathbb{N}$ tel que $\forall x.\varphi \in E_m$ et $n > m$ tel que $\theta_n = \forall x.\varphi$, donc il existe $y \in X$ telle que E_{n+1} et donc H contiennent $\varphi[y/x]$.

L'ensemble H est de plus cohérent : si α et $\neg\alpha$ appartenaient à H, alors il existerait $n \in \mathbb{N}$ tel que $\{\alpha, \neg\alpha\} \subseteq E_n$, mais alors E_n serait prouvable par une application de la règle d'axiome. □

Théorème 8.5 (complétude). *Si $\vDash \Gamma$, alors $\vdash_{\mathbf{LK}} \Gamma$ par une dérivation sans coupure.*

Démonstration. Par contraposée, supposons que le séquent $\vdash \Gamma$ ne soit pas dérivable sans coupure. Alors l'ensemble $\mathrm{dom}(\Gamma)$ n'est pas prouvable : si Δ est un multi-ensemble fini tel que $\mathrm{dom}(\Delta) \subseteq \mathrm{dom}(\Gamma)$, alors $\vdash \Delta$ n'est pas dérivable sans coupure, sans quoi on pourrait aussi dériver $\vdash \Gamma$ par affaiblissement et contraction.

Par le lemme 8.4 de saturation, $\mathrm{dom}(\Gamma)$ est inclus dans H un ensemble de Hintikka. Par le lemme 8.2 de Hintikka, il existe une interprétation I et une valuation ρ telles que, pour toute formule φ de H (et donc en particulier pour toute formule φ de $\mathrm{dom}(\Gamma)$), $I, \rho \nvDash \varphi$: on a bien $\nvDash \Gamma$. □

9. Élimination des coupures

[David et al., 2003, sec. 5.4]

Les théorèmes 7.1 et 8.5 montrent que la règle de coupure est inutile dans le calcul des séquents : si $\vdash_{\mathbf{LK}} \Gamma$ en utilisant (cut), alors $\vDash \Gamma$ par le théorème de correction, et donc $\vdash_{\mathbf{LK}} \Gamma$ par une dérivation sans coupure par le théorème de complétude.

Proposer l'élimination des coupures en développement d'agrégation serait pour le moins hasardeux.

On peut cependant faire une preuve directe de ce résultat, sans faire appel à des notions sémantiques. La stratégie générale est de réécrire les dérivations du calcul des séquents avec coupure en des dérivations sans coupure, en faisant « remonter » les instances de (cut) vers les axiomes. L'intérêt de cette preuve syntaxique est qu'étant donné une preuve avec coupure, elle *construit* une preuve sans coupure. La difficulté principale est de montrer que ce processus termine.

Une *coupure maximale* est une dérivation π finissant par une règle (cut) de la forme

$$\dfrac{\begin{array}{cc}\pi_1 & \pi_2 \\ \vdots & \vdots \\ \vdash \Gamma, \varphi & \vdash \overline{\varphi}, \Delta\end{array}}{\vdash \Gamma, \Delta}\ (\text{cut})$$

où π_1 et π_2 sont sans coupure. Son *rang de coupure* est $r(\pi) \stackrel{\text{def}}{=} (p(\varphi), p(\pi_1)+p(\pi_2))$. On ordonne les rangs de coupure dans $\mathbb{N}^2$ lexicographiquement ; il s'agit d'un ordre bien fondé.

Lemme 9.1. *Soit π une coupure maximale. Alors il existe une dérivation π' sans coupure du même séquent.*

Démonstration. On procède par induction bien fondée sur $r(\pi)$. Rappelons qu'une telle dérivation π est de la forme

$$\dfrac{\begin{array}{cc}\pi_1 & \pi_2 \\ \vdots & \vdots \\ \vdash \Gamma, \varphi & \vdash \overline{\varphi}, \Delta\end{array}}{\vdash \Gamma, \Delta}\ (\text{cut})$$

où π_1 et π_2 sont sans coupure.

On procède à une première analyse par cas, selon que φ soit principale dans la dernière règle de π_1 ou $\overline{\varphi}$ dans la dernière règle de π_2.

cas « np1 » : si φ n'est pas principale dans la dernière règle (R) de π_1. Alors π_1 était de la forme

$$\dfrac{\begin{array}{ccc}\pi'_1 & & \pi'_k\\ \vdots & & \vdots\\ \vdash \Gamma_1, \varphi & \cdots & \vdash \Gamma_k, \varphi\end{array}}{\vdash \Gamma, \varphi}\ (\mathsf{R})$$

pour un certain $0 \leq k \leq 2$, où $k = 0$ correspond au cas de la règle (ax). Par inspection des règles (R) du calcul des séquents, comme φ n'est pas principale, elle apparaît en effet dans toutes les prémisses de (R). Encore par inspection des règles, on observe que l'on peut transformer π en

$$\dfrac{\dfrac{\begin{array}{cc}\pi'_1 & \pi_2\\ \vdots & \vdots\\ \vdash \Gamma_1, \varphi & \vdash \overline{\varphi}, \Delta\end{array}}{\vdash \Gamma_1, \Delta}\ (\mathsf{cut}) \quad \cdots \quad \dfrac{\begin{array}{cc}\pi'_k & \pi_2\\ \vdots & \vdots\\ \vdash \Gamma_k, \varphi & \vdash \overline{\varphi}, \Delta\end{array}}{\vdash \Gamma_k, \Delta}\ (\mathsf{cut})}{\vdash \Gamma, \Delta}\ (\mathsf{R})$$

Les k nouvelles instances de (cut) entre π'_i et π_2 sont maximales et de rangs respectifs $(p(\varphi), p(\pi'_i) + p(\pi_2))$ pour $1 \leq i \leq k$, où $p(\pi'_i) < p(\pi_1)$. On peut donc appliquer l'hypothèse d'induction pour obtenir des dérivations sans coupure de chaque $\vdash \Gamma_i, \Delta$. La dérivation résultante de $\vdash \Gamma, \Delta$ est sans coupure.

cas « np2 » : si $\overline{\varphi}$ n'est pas principale dans la dernière règle de π_2. Similaire au cas précédent.

On peut donc supposer maintenant que φ est principale dans la dernière règle de π_1 et $\overline{\varphi}$ dans la dernière de π_2. Nous faisons une nouvelle distinction de cas selon que ces dernières règles soient (ax) ou non.

cas « ax1 » : si φ est principale dans π_1 finissant par (ax). Alors $\varphi = \ell, \Gamma = \Gamma, \overline{\ell}$ et π était de la forme

$$\dfrac{\dfrac{}{\vdash \Gamma', \ell, \overline{\ell}}\ (\mathsf{ax}) \qquad \begin{array}{c}\pi_2\\ \vdots\\ \vdash \overline{\ell}, \Delta\end{array}}{\vdash \Gamma', \overline{\ell}, \Delta}\ (\mathsf{cut})$$

Par le lemme 6.3, on aurait pu obtenir le même résultat par affaiblissement de π_2. On obtient alors directement une preuve sans coupure de $\vdash \Gamma, \Delta$.

cas « ax2 » : si $\overline{\varphi}$ est principale dans π_2 finissant par (ax). Similaire au cas précédent.

Il reste maintenant le cas où π_1 et π_2 ont pour dernières règles ($\vee$) et ($\wedge$), ou ($\exists$) et ($\forall$) : ce sont les seules paires possibles puisque φ et $\overline{\varphi}$ sont principales et que (ax) a déjà été traité.

cas « $\vee\wedge$ » : $\varphi = \varphi' \vee \psi$ est principale dans la dernière règle ($\vee$) de π_1 et $\overline{\varphi} = \overline{\varphi'} \wedge \overline{\psi}$ dans la dernière règle ($\wedge$) de π_2. La dérivation π est donc de la forme

$$\dfrac{\dfrac{\begin{array}{c}\pi'_1\\ \vdots\\ \vdash \Gamma, \varphi', \psi\end{array}}{\vdash \Gamma, \varphi' \vee \psi}\ (\vee) \qquad \dfrac{\begin{array}{cc}\pi'_2 & \pi'_3\\ \vdots & \vdots\\ \vdash \overline{\varphi'}, \Delta & \vdash \overline{\psi}, \Delta\end{array}}{\vdash \overline{\varphi'} \wedge \overline{\psi}, \Delta}\ (\wedge)}{\vdash \Gamma, \Delta}\ (\mathsf{cut})$$

On transforme cette dérivation en

$$\dfrac{\dfrac{\begin{array}{c}\pi'_1\\ \vdots\\ \vdash \Gamma, \varphi', \psi\end{array} \qquad \begin{array}{c}\pi'_2\\ \vdots\\ \vdash \overline{\varphi'}, \Delta\end{array}}{\vdash \Gamma, \psi, \Delta}\,(\text{cut}) \qquad \begin{array}{c}\pi'_3\\ \vdots\\ \vdash \overline{\psi}, \Delta\end{array}}{\vdash \Gamma, \Delta, \Delta}\,(\text{cut})$$

La coupure entre π'_1 et π'_2 est maximale et de rang $(p(\varphi'), p(\pi'_1)+p(\pi'_2))$ où $p(\varphi') < p(\varphi)$; on peut donc appliquer l'hypothèse d'induction pour obtenir une dérivation π' sans coupure de $\vdash \Gamma, \psi, \Delta$. La dérivation résultante est maintenant

$$\dfrac{\begin{array}{c}\pi'\\ \vdots\\ \vdash \Gamma, \psi, \Delta\end{array} \qquad \begin{array}{c}\pi'_3\\ \vdots\\ \vdash \overline{\psi}, \Delta\end{array}}{\vdash \Gamma, \Delta, \Delta}\,(\text{cut})$$

La coupure y est maximale et de rang $(p(\psi), p(\pi')+p(\pi'_3))$ où $p(\psi) < p(\varphi)$; on peut encore appliquer l'hypothèse d'induction pour obtenir une dérivation sans coupure de $\vdash \Gamma, \Delta, \Delta$. Enfin, par le lemme 6.6 de contraction, puisque $\vdash \Gamma, \Delta, \Delta$ est dérivable sans coupure, $\vdash \Gamma, \Delta$ l'est aussi.

cas « $\wedge\vee$ » : $\varphi = \psi \wedge \psi'$ est principale dans la dernière règle ($\wedge$) de π_1 et $\overline{\varphi} = \overline{\psi} \vee \overline{\psi'}$ dans la dernière règle ($\vee$) de π_2. Similaire au cas précédent.

cas « $\exists\forall$ » : $\varphi = \exists x.\psi$ est principale dans la dernière règle ($\exists$) de π_1 et $\overline{\varphi} = \forall x.\overline{\psi}$ dans la dernière règle ($\forall$) de π_2. La dérivation π est donc de la forme

$$\dfrac{\dfrac{\begin{array}{c}\pi'_1\\ \vdots\\ \vdash \Gamma, \psi[t/x], \exists x.\psi\end{array}}{\vdash \Gamma, \exists x.\psi}\,(\exists) \qquad \dfrac{\begin{array}{c}\pi'_2\\ \vdots\\ \vdash (\overline{\psi})[y/x], \Delta\end{array}}{\vdash \forall x.\overline{\psi}, \Delta}\,(\forall)}{\vdash \Gamma, \Delta}\,(\text{cut})$$

où $y \notin \text{fv}(\forall x.\overline{\psi}, \Delta)$. Par le lemme 6.2, il existe une dérivation sans coupure π''_2 du séquent $\vdash (\overline{\psi})[y/x][t/y], \Delta[t/y]$. Comme $y \notin \text{fv}(\forall x.\overline{\psi})$, par l'équation (2), $(\overline{\psi})[y/x][t/y] =_\alpha (\overline{\psi})[t/x]$ et comme $y \notin \text{fv}(\Delta)$, par l'équation (1), $\Delta[t/y] =_\alpha \Delta$. De plus, par la propriété 4.2, $(\overline{\psi})[t/x] = \overline{(\psi[t/x])}$. On obtient donc une nouvelle dérivation par le lemme 6.1 d'α-congruence syntaxique

$$\dfrac{\dfrac{\begin{array}{c}\pi'_1\\ \vdots\\ \vdash \Gamma, \psi[t/x], \exists x.\psi\end{array} \qquad \begin{array}{c}\pi_2\\ \vdots\\ \vdash \forall x.\overline{\psi}, \Delta\end{array}}{\vdash \Gamma, \psi[t/x], \Delta}\,(\text{cut}) \qquad \begin{array}{c}\pi''_2\\ \vdots\\ \vdash \overline{(\psi[t/x])}, \Delta\end{array}}{\vdash \Gamma, \Delta, \Delta}\,(\text{cut})$$

La coupure entre π'_1 et π_2 est maximale et de rang $(p(\varphi), p(\pi'_1)+p(\pi_2))$ où $p(\pi'_1) < p(\pi_1)$; on peut donc appliquer l'hypothèse d'induction pour obtenir une dérivation π' sans coupure de $\vdash \Gamma, \psi[t/x], \Delta$. La dérivation résultante est

$$\dfrac{\dfrac{\begin{array}{c}\pi'\\ \vdots\\ \vdash \Gamma, \psi[t/x], \Delta\end{array} \qquad \begin{array}{c}\pi''_2\\ \vdots\\ \vdash \overline{(\psi[t/x])}, \Delta\end{array}}{\vdash \Gamma, \Delta, \Delta}\,(\text{cut})$$

La coupure y est maximale et de rang $(p(\psi[t/x]), p(\pi') + p(\pi''_2))$ où $p(\psi[t/x]) = p(\psi) < p(\varphi)$; on peut encore appliquer l'hypothèse d'induction pour obtenir une dérivation sans coupure de $\vdash \Gamma, \Delta, \Delta$. Enfin, par le lemme 6.6 de contraction, $\vdash \Gamma, \Delta$ est dérivable sans coupure.

cas « ∀∃ » : $\varphi = \forall x.\psi$ est principale dans la dernière règle (∀) de π_1 et $\overline{\varphi} = \exists x.\overline{\psi}$ dans la dernière règle (∃) de π_2. Similaire au cas précédent. □

Théorème 9.2 (élimination des coupures)**.** *Soit π une dérivation du calcul des séquent. Alors il existe une dérivation sans coupure du même séquent.*

Démonstration. On procède par récurrence sur le nombre n d'instances de la règle (cut) dans π. Pour le cas de base $n = 0$, π est déjà une dérivation sans coupure. Supposons donc $n > 0$. Il existe donc une coupure maximale π' qui est une sous-dérivation de π. Par le lemme 9.1, il existe une dérivation sans coupure équivalente à π' ; on remplace π' par celle-ci dans π pour obtenir une dérivation équivalente à π avec $n - 1$ instances de (cut). Par hypothèse de récurrence, cette nouvelle dérivation a une dérivation sans coupure équivalente. □

L'élimination des coupures est souvent combinée avec la propriété de sous-formule : pour une formule φ en forme normale négative, son ensemble de *sous-formules* $\mathrm{sub}(\varphi)$ est défini inductivement par

$$\mathrm{sub}(\ell) \stackrel{\text{def}}{=} \{\ell\}\,,$$

$$\mathrm{sub}(\varphi \vee \psi) \stackrel{\text{def}}{=} \{\varphi \vee \psi\} \cup \mathrm{sub}(\varphi) \cup \mathrm{sub}(\psi)\,, \qquad \mathrm{sub}(\varphi \wedge \psi) \stackrel{\text{def}}{=} \{\varphi \wedge \psi\} \cup \mathrm{sub}(\varphi) \cup \mathrm{sub}(\psi)\,,$$

$$\mathrm{sub}(\exists x.\varphi) \stackrel{\text{def}}{=} \{\exists x.\varphi\} \cup \{\varphi[t/x] \mid t \in T(\mathcal{F}, X)\},\ \mathrm{sub}(\forall x.\varphi) \stackrel{\text{def}}{=} \{\forall x.\varphi\} \cup \{\varphi[y/x] \mid y \in X\}\,.$$

Propriété 9.3 (sous-formule)**.** *Dans toutes les règles du calcul des séquents sauf* (cut)*, toutes les formules apparaissant dans les prémisses sont des sous-formules de formules apparaissant dans la conclusion.*

9.1. **Cohérence.** Le calcul des séquents et l'élimination des coupures ont été introduits par Gentzen dans le cadre de l'arithmétique de Peano pour en démontrer la *cohérence*. Ce résultat vaut aussi pour la logique du premier ordre.

[Goubault-Larrecq et Mackie, 1997, sec. 6.6.2]

Théorème 9.4 (cohérence)**.** *Le calcul des séquents est cohérent : le séquent vide $\vdash \bot$ n'est pas prouvable.*

[David et al., 2003, thm. 5.5.1], [Goubault-Larrecq et Mackie, 1997, cor. 6.31].

Démonstration. Supposons par contradiction que $\vdash \bot$ soit dérivable. Alors par le théorème 9.2 d'élimination des coupures, il en existerait une preuve sans coupure. Par la propriété de sous-formule, tous les séquents apparaissant dans cette dérivation seraient formés de sous-formules du séquent $\vdash \bot$, et seraient donc eux aussi vides. Mais il n'y a pas moyen de clore une telle dérivation par des instances de la règle d'axiome (ax), puisqu'elle nécessite des séquents non vides. □

Le théorème d'interpolation est un exemple de résultat méta-mathématique que l'on aurait aussi pu prouver par des méthodes sémantiques. On voit ici une application des systèmes de preuve : démontrer des résultats méta-mathématiques par des méthodes purement syntaxiques.

9.2. **Interpolation de Craig.** La notion d'inversibilité ne s'applique pas à la règle de coupure. En revanche, si $\vdash_{\mathbf{LK}} \Gamma, \Delta$, alors il existe un *interpolant* de Γ et Δ, c'est-à-dire une formule η telle que $\vdash_{\mathbf{LK}} \Gamma, \eta$ et $\vdash_{\mathbf{LK}} \overline{\eta}, \Delta$. Intuitivement, un interpolant de Γ et Δ est une formule impliquée par $\overline{\Gamma}$ et qui implique Δ. L'intérêt de cette notion est que l'on peut toujours *calculer* un tel interpolant à partir d'une dérivation sans coupure de $\vdash_{\mathbf{LK}} \Gamma, \Delta$.

Théorème 9.5 (interpolation)**.** *Si $\vdash_{\mathbf{LK}} \Gamma, \Delta$ par une dérivation sans coupure, alors il existe un interpolant η de Γ et Δ, dont les sous-formules atomiques sont issues de* $(\mathrm{sub}(\Gamma) \cap \mathrm{sub}(\Delta)) \cup \{\top, \bot\}$.

[David et al., 2003, thm. 5.5.21]

Démonstration. On procède par récurrence sur la profondeur de la dérivation sans coupure π de $\vdash \Gamma, \Delta$. On fait une distinction de cas selon la dernière règle (R) employée dans π.

Pour (ax) : quatre cas sont possibles, selon que les formules principales ℓ et $\overline{\ell}$ apparaissent

- toutes deux dans $\Gamma = \Gamma', \ell, \overline{\ell}$, et alors $\eta \stackrel{\text{def}}{=} \bot$ convient ;
- l'une dans $\Gamma = \Gamma', \ell$ et l'autre dans $\Delta = \overline{\ell}, \Delta'$, et alors $\eta \stackrel{\text{def}}{=} \overline{\ell}$ convient ;
- l'une dans $\Delta = \Delta', \ell$ et l'autre dans $\Gamma = \Gamma', \overline{\ell}$, et alors $\eta \stackrel{\text{def}}{=} \ell$ convient ;
- ou toutes deux dans $\Delta = \ell, \overline{\ell}, \Delta'$, et alors $\eta \stackrel{\text{def}}{=} \top$ convient.

Pour ($\vee$) : on suppose que la formule principale apparaît dans $\Delta = \Delta', \varphi \vee \psi$. Par hypothèse de récurrence, il existe un interpolant η de Γ et Δ', φ, ψ qui est aussi un interpolant de Γ et Δ ; on a en effet les dérivations de

$$\vdash \Gamma, \eta \qquad \text{et} \qquad \frac{\vdash \overline{\eta}, \Delta', \varphi, \psi}{\vdash \overline{\eta}, \Delta', \varphi \vee \psi}\ (\vee)$$

Si la formule principale apparaît dans Γ, on applique le même type de raisonnement.

Pour ($\wedge$) : on suppose que la formule principale apparaît dans $\Delta = \Delta', \varphi \wedge \psi$. Par hypothèse de récurrence, il existe un interpolant η_1 de Γ et Δ', φ et un interpolant η_2 de Γ et Δ', ψ. Alors $\eta \stackrel{\text{def}}{=} \eta_1 \wedge \eta_2$ est un interpolant de Γ et Δ ; on a en effet les dérivations de

$$\frac{\vdash \Gamma, \eta_1 \qquad \vdash \Gamma, \eta_2}{\vdash \Gamma, \eta_1 \wedge \eta_2}\ (\wedge) \quad \text{et} \quad \frac{\dfrac{\dfrac{\vdash \overline{\eta_1}, \Delta', \varphi}{\vdash \overline{\eta_1}, \overline{\eta_2}, \Delta', \varphi}\ (\mathsf{W}) \qquad \dfrac{\vdash \overline{\eta_2}, \Delta', \psi}{\vdash \overline{\eta_1}, \overline{\eta_2}, \Delta', \psi}\ (\mathsf{W})}{\vdash \overline{\eta_1}, \overline{\eta_2}, \Delta', \varphi \wedge \psi}\ (\wedge)}{\vdash \overline{\eta_1} \vee \overline{\eta_2}, \Delta', \varphi \wedge \psi}\ (\vee)$$

Si la formule principale apparaît dans Γ, on applique le même type de raisonnement.

Pour ($\exists$) : on suppose que la formule principale apparaît dans $\Delta = \Delta', \exists x.\varphi$. Par hypothèse de récurrence, il existe un interpolant η de Γ et $\Delta', \varphi[t/x], \exists x.\varphi$ qui est aussi un interpolant de Γ et $\Delta', \exists x.\varphi$; on a en effet les dérivations de

$$\vdash \Gamma, \eta \qquad \text{et} \qquad \frac{\vdash \overline{\eta}, \Delta', \varphi[t/x], \exists x.\varphi}{\vdash \overline{\eta}, \Delta', \exists x.\varphi}\ (\exists)$$

Si la formule principale apparaît dans Γ, on applique le même type de raisonnement.

Pour ($\forall$) : on suppose que la formule principale apparaît dans $\Delta = \Delta', \forall x.\varphi$. Par hypothèse de récurrence, il existe un interpolant η' de Γ et $\Delta', \varphi[y/x]$ où $y \notin \mathrm{fv}(\Gamma, \Delta', \forall x.\varphi)$. On définit $\eta \stackrel{\text{def}}{=} \forall y.\eta'$ et soit $z \notin \mathrm{fv}(\forall y.\eta', \Gamma, \Delta', \forall x.\varphi)$; on a les dérivations de

$$\frac{\dfrac{\dfrac{\vdash \Gamma, \eta'}{\vdash \Gamma[z/y], \eta'[z/y]}\ (\mathsf{S})}{\vdash \Gamma, \eta'[z/y]}\ (=_\alpha)}{\vdash \Gamma, \forall y.\eta'}\ (\forall) \quad \text{et} \quad \frac{\dfrac{\dfrac{\dfrac{\dfrac{\vdash \overline{\eta'}, \Delta', \varphi[y/x]}{\vdash \overline{\eta'}[z/y], \Delta'[z/y], \varphi[y/x][z/y]}\ (\mathsf{S})}{\vdash \overline{\eta'}[z/y], \Delta', \varphi[z/x]}\ (=_\alpha)}{\vdash \overline{\eta'}[z/y], \exists y.\overline{\eta'}, \Delta', \varphi[z/x]}\ (\mathsf{W})}{\vdash \exists y.\overline{\eta'}, \Delta', \varphi[z/x]}\ (\exists)}{\vdash \exists y.\overline{\eta'}, \Delta', \forall x.\varphi}\ (\forall)$$

Si la formule principale apparaît dans Γ, on applique le même type de raisonnement. □

*La logique de HOARE est officiellement au programme de l'option D, et peut être abordée aux leçons **927** dans le cadre d'exemples et **930** dans le cadre de la sémantique axiomatique des langages de programmation.*

Exemple 9.6 (Application à la logique de HOARE)**.** Une application des interpolants est la recherche de preuve en *logique de HOARE*. Cette dernière manipule des triplets $\{\varphi\}c\{\varphi'\}$ où φ est une pré-condition, φ' une post-relation, et c une instruction du langage de programmation.

Une cas typique d'application est celui où on sait démontrer $\{\varphi_1\}c_1\{\varphi_1'\}$ et $\{\varphi_2\}c_2\{\varphi_2'\}$ pour deux instructions c_1 et c_2 – possiblement φ_1' est la plus forte post-relation $sp(\varphi_1, c_1)$ pour φ_1 et c_1, tandis que φ_2 est la plus faible pré-condition $wp(\varphi_2', c_2)$ pour φ_2' et c_2 – et que l'on souhaite dériver $\{\varphi_1\}c_1; c_2\{\varphi_2'\}$ pour la composition séquentielle des deux instructions, mais où $\varphi_1' \neq \varphi_2$. Une telle dérivation aura alors typiquement la forme

$$\frac{\dfrac{\{\varphi_1\}c_1\{\varphi_1'\} \qquad \vDash \varphi_1' \Rightarrow \eta}{\{\varphi_1\}c_1\{\eta\}}\ (\text{conseq}) \qquad \dfrac{\vDash \eta \Rightarrow \varphi_2 \qquad \{\varphi_2\}c_2\{\varphi_2'\}}{\{\eta\}c_2\{\varphi_2'\}}\ (\text{conseq})}{\{\varphi_1\}c_1; c_2\{\varphi_2'\}}\ (\text{seq})$$

où η est un interpolant de $\overline{\varphi_1'}$ et φ_2, qui peut être calculé par le théorème d'interpolation si l'on dispose d'une preuve de $\vdash_{\mathbf{LK}} \overline{\varphi_1'}, \varphi_2$. L'avantage d'utiliser un tel interpolant est que η est typiquement une formule plus « simple » que $\overline{\varphi_1'}$ ou φ_2.

Partie 3. Résolution

[David et al., 2003, sec 7.4], [Goubault-Larrecq et Mackie, 1997, sec. 7.3]

[David et al., 2003, sec. 7.4.3], [Goubault-Larrecq et Mackie, 1997, sec. 7.4] pour un aperçu de quelques-unes de ces optimisations.

Le second système de preuve de ces notes est le système de résolution. Son principal intérêt est qu'il est extrêmement simple (il ne contient que deux règles), mais qu'il se prête à de nombreuses optimisations et sert de point de départ pour la plupart des implémentations disponibles pour les problèmes de validité et de satisfiabilité en logique du premier ordre. L'idée du système de résolution est de montrer l'insatisfiabilité de $\neg\varphi$ plutôt que la validité de φ, en dérivant « $\bot$ » depuis $\neg\varphi$. Ce système de preuve repose crucialement sur l'*unification* ; voir les annexes.

On supposera sans perte de généralité $\neg\varphi$ close et mise sous forme clausale, c'est-à-dire sous la forme $\bigwedge_i(\forall\vec{x}_i.\bigvee_{j_i}\ell_{j_i})$. Le système de déduction de la figure 4 travaille sur de telles formules, mais représente chaque clause $\forall\vec{x}.\ell_1\vee\cdots\vee\ell_n$ comme un ensemble fini de littéraux $\{\ell_1,\dots,\ell_n\}$ avec $\vec{x}=\mathrm{fv}(\{\ell_1,\dots,\ell_n\})$. Les symboles C et D dénotent de tels ensembles finis, et la virgule dénote l'union ; on notera l'ensemble vide « $\bot$ ». À noter que les variables libres d'un ensemble $C=\{\ell_1,\dots,\ell_n\}$ sont en fait des variables *liées* de la formule close $\forall\vec{x}.\ell_1\vee\cdots\vee\ell_n$. L'application d'une substitution σ à un ensemble de littéraux $C=\{\ell_1,\dots,\ell_n\}$ résulte en un ensemble $C\sigma\stackrel{\text{def}}{=}\{\ell_1\sigma,\dots,\ell_n\sigma\}$.

La règle de résolution (res) *combine* (S) *avec* (cut)*, tandis que la règle de factorisation* (fac) *combine* (S) *avec* (C) *; voir [Goubault-Larrecq et Mackie, 1997, sec. 7.5] sur les liens avec le calcul des séquents.*

$$\frac{C}{C[y/x]}\ (\to_\alpha)\quad \text{où } y\notin\mathrm{fv}(C)\setminus\{x\} \qquad \frac{C,\alpha\quad\neg\alpha',D}{C\sigma,D\sigma}\ (\text{res})\quad \text{où } \sigma=\mathrm{mgu}(\alpha\stackrel{?}{=}\alpha') \text{ et } \mathrm{fv}(C,\alpha)\cap\mathrm{fv}(\neg\alpha',D)=\emptyset \qquad \frac{C,\ell,\ell'}{C\sigma,\ell\sigma}\ (\text{fac})\quad \text{où } \sigma=\mathrm{mgu}(\ell\stackrel{?}{=}\ell')$$

FIGURE 4. Règles de la résolution en logique du premier ordre.

La règle $(\to_\alpha)$ *effectue bien un* α*-renommage* $\forall x\forall\vec{z}.\ell_1\vee\cdots\vee\ell_n\to_\alpha\forall y\forall\vec{z}.\ell_1[y/x]\vee\cdots\vee\ell_n[y/x]$ *où* $y\notin\mathrm{fv}(\forall x\forall\vec{z}.\ell_1\vee\cdots\vee\ell_n)=\emptyset$ *et* $x,y\notin\mathrm{bv}(\forall\vec{z}.\ell_1\vee\cdots\vee\ell_n)=\vec{z}$.

Exemple 9.7. Rappelons comme vu dans l'exemple 4.7 que la forme clausale de la *négation* de la formule du buveur est $\neg B(a)\wedge\forall x.B(x)$. Cette négation n'est pas satisfiable, comme le montre la dérivation

$$\frac{B(x)\qquad\neg B(a)}{\bot}\ (\text{res})$$

où $\mathrm{mgu}(B(x)\stackrel{?}{=}B(a))=[a/x]$.

La condition de variables disjointes dans (res) *permet en réalité d'appliquer la règle dans* plus *de cas, car il y a plus de termes unifiables après application de la règle* $(\to_\alpha)$.

Exemple 9.8. La règle de factorisation est nécessaire à la complétude de la résolution. Par exemple, la dérivation suivante montre que la formule $(\forall x.R(x,a)\vee R(a,x))\wedge(\forall y.\neg R(y,a)\vee\neg R(a,y))$ est insatisfiable :

$$\frac{\dfrac{R(x,a),R(a,x)}{R(a,a)}\ (\text{fac})\qquad\dfrac{\neg R(y,a),\neg R(a,y)}{\neg R(a,a)}\ (\text{fac})}{\bot}\ (\text{res})$$

Cependant, la règle de résolution seule ne peut que produire les clauses suivantes : $R(x,a)\vee\neg R(x,a)$, $R(a,x)\vee\neg R(a,x)$ et $R(a,a)\vee\neg R(a,a)$, à renommage des variables près.

10. Correction et complétude réfutationnelle

Un ensemble de littéraux $C=\{\ell_1,\dots,\ell_n\}$ est satisfait dans une interprétation I – noté $I\vDash C$ – si, pour toute valuation ρ, $I,\rho\vDash\ell_1\vee\cdots\vee\ell_n$. Un ensemble S d'ensembles de littéraux est satisfait dans une interprétation I – noté $I\vDash S$ – si $I\vDash C$ pour tout $C\in S$. Si pour toute interprétation I, $I\vDash S$ implique $I\vDash C$, alors C est une conséquence logique de S et on écrit $S\vDash C$.

10.1. Correction.

Si S est un ensemble de clauses, on note $S\vdash_{\mathbf{R}}C$ quand la clause C peut être dérivée à partir de clauses de S par le système de la figure 4.

Théorème 10.1 (correction)**.** *Soit S un ensemble de clauses et C une clause. Si $S\vdash_{\mathbf{R}}C$ alors $S\vDash C$.*

Démonstration. Par induction structurelle sur la dérivation de C. Le cas de base $C \in S$ est évident, et il reste à vérifier que les règles de la figure 4 préservent la satisfaction.

- **Pour $(\rightarrow_\alpha)$:** supposons $S \vdash_{\mathbf{R}} C$ pour $C = \{\ell_1, \dots, \ell_n\}$ et $y \notin \mathrm{fv}(C) \setminus \{x\}$. Soit I une interprétation telle que $I \vDash S$; on souhaite montrer que $I \vDash C[y/x]$. Par hypothèse d'induction, $I \vDash C$. Comme $(\rightarrow_\alpha)$ opère à un α-renommage, par le lemme 3.4 d'α-congruence, $I \vDash C[y/x]$.
- **Pour (res) :** supposons $S \vdash_{\mathbf{R}} C, \alpha$ et $S \vdash_{\mathbf{R}} \neg\alpha', D$ avec $\sigma = \mathrm{mgu}(\alpha \stackrel{?}{=} \alpha')$. Soit I une interprétation telle que $I \vDash S$; on souhaite montrer que $I \vDash C\sigma, D\sigma$. Par hypothèse d'induction, $I \vDash C, \alpha$ et $I \vDash \neg\alpha', D$. Soit ρ une valuation choisie arbitrairement. Alors en particulier $I, \sigma\rho \vDash \ell$ pour un littéral $\ell \in C \cup \{\alpha\}$ et $I, \sigma\rho \vDash \ell'$ pour un littéral $\ell' \in \{\neg\alpha'\} \cup D$. Par le lemme 3.5 de substitution, $I, \rho \vDash \ell\sigma$ et $I, \rho \vDash \ell'\sigma$. Alors $(\ell, \ell') \neq (\alpha, \neg\alpha')$ puisque $\alpha\sigma = \alpha'\sigma$, et donc $I, \rho \vDash C\sigma, D\sigma$.
- **Pour (fac) :** supposons $S \vdash_{\mathbf{R}} C, \ell, \ell'$ avec $\sigma = \mathrm{mgu}(\ell \stackrel{?}{=} \ell')$. Soit I une interprétation telle que $I \vDash S$; on souhaite montrer que $I \vDash C\sigma, \ell\sigma$. Par hypothèse d'induction, $I \vDash C, \ell, \ell'$. Soit ρ une valuation choisie arbitrairement. Alors en particulier $I, \sigma\rho \vDash \ell''$ pour un littéral $\ell'' \in C \cup \{\ell, \ell'\}$. Par le lemme 3.5 de substitution, $I, \rho \vDash \ell''\sigma$. Comme $\ell\sigma = \ell'\sigma$, $I, \rho \vDash C\sigma, \ell\sigma$. □

10.2. **Complétude réfutationnelle par le théorème de Herbrand.** Cette première preuve de la complétude réfutationnelle du système de résolution en logique du premier ordre s'appuie sur la complétude réfutationnelle de la résolution propositionnelle vue à la leçon **916** et sur le théorème de Herbrand. On utilise ici la résolution propositionnelle sur l'ensemble de propositions $\mathcal{B}$, la base de Herbrand sur $\mathcal{F}$ et $\mathcal{P}$. Si S est un ensemble de clauses propositionnelles et C une clause propositionnelle, on notera $S \vdash_{\mathbf{R}_0} C$ quand la clause C peut être dérivée depuis des clauses de S en résolution propositionnelle.

Lemme 10.2 (relèvement)**.** *Soient C_1 et C_2 deux clauses. Si $C_1' \in C_1\Sigma$ et $C_2' \in C_2\Sigma$ en sont des instances de Herbrand et se résolvent propositionnellement en C', alors il existe une clause C telle que $\{C_1, C_2\} \vdash_{\mathbf{R}} C$ et $C' \in C\Sigma$.*

🔖 *[Chang et Lee, 1973, lem. 5.1]*

Démonstration. Modulo applications de la règle $(\rightarrow_\alpha)$, on peut supposer que C_1 et C_2 soient telles que $\mathrm{fv}(C_1) \cap \mathrm{fv}(C_2) = \emptyset$. Soit α' la formule atomique (nécessairement close) utilisée dans la résolution $\{C_1', C_2'\} \vdash_{\mathbf{R}_0} C'$. Écrivons $C_1' \stackrel{\text{def}}{=} D_1', \alpha'$ et $C_2' \stackrel{\text{def}}{=} \neg\alpha', D_2'$; alors $C' = D_1', D_2'$.

Comme les variables libres de C_1 et C_2 sont disjointes, il existe une substitution close σ telle que $C_i' = C_i\sigma$ pour $i \in \{1, 2\}$. Soient $L_1 \stackrel{\text{def}}{=} \{\alpha \in C_1 \mid \alpha\sigma = \alpha'\}$ et $L_2 \stackrel{\text{def}}{=} \{\neg\alpha \in C_2 \mid \alpha\sigma = \alpha'\}$; ces deux ensembles sont non vides. Soient $D_1 \stackrel{\text{def}}{=} C_1 \setminus L_1$ et $D_2 \stackrel{\text{def}}{=} C_2 \setminus L_2$.

Si L_1 (resp. L_2) est un singleton $\{\alpha_1\}$ (resp. $\{\neg\alpha_2\}$), alors on a $\{C_1\} \vdash_{\mathbf{R}} D_1\mu_1, \alpha_1$ (resp. $\{C_2\} \vdash_{\mathbf{R}} D_2\mu_1, \neg\alpha_2$) pour μ_1 (resp. μ_2) la substitution identité. Sinon, on pose $\mu_1 \stackrel{\text{def}}{=} \mathrm{mgu}(L_1)$ (resp. $\mu_2 \stackrel{\text{def}}{=} \mathrm{mgu}(L_2)$), ce qui fournit une formule atomique $\alpha_1 = \alpha\mu_1$ pour tout $\alpha \in L_1$ (resp. $\alpha_2 = \alpha\mu_2$ pour tout $\neg\alpha \in L_2$). Par applications de (fac), on déduit $\{C_1\} \vdash_{\mathbf{R}} D_1\mu_1, \alpha_1$ (resp. $\{C_2\} \vdash_{\mathbf{R}} D_2\mu_2, \neg\alpha_2$).

Soit $\mu \stackrel{\text{def}}{=} \mu_1\mu_2$. Comme les variables libres de C_1 et C_2 sont disjointes, dans tous les cas $\{C_1\} \vdash_{\mathbf{R}} D_1\mu, \alpha_1$ et $\{C_2\} \vdash_{\mathbf{R}} D_2\mu, \neg\alpha_2$. De plus, α' est une instance close de α_1 et α_2, qui ont donc un unificateur $\nu \stackrel{\text{def}}{=} \mathrm{mgu}(\alpha_1 \stackrel{?}{=} \alpha_2)$.

Soit $C \stackrel{\text{def}}{=} D_1\mu\nu, D_2\mu\nu$. Par une application de (res), on obtient donc $\{C_1, C_2\} \vdash_{\mathbf{R}} C$. Comme α' est une instance close de $\alpha\mu\nu$ pour tout $\alpha \in L_1$ et tout $\neg\alpha \in L_2$, $\mu\nu$ est plus général que σ, et donc C' est une instance close de C. □

Par induction sur la longueur des dérivations en résolution propositionnelle, on en déduit :

Corollaire 10.3. *Soit S un ensemble de clauses. Si $S\Sigma \vdash_{\mathbf{R}_0} C'$, alors il existe C telle que $S \vdash_{\mathbf{R}} C$ et $C' \in C\Sigma$.*

💬 *La réciproque est vraie.*

[Goubault-Larrecq et Mackie, 1997, thm. 7.11]. Voir aussi [David et al., 2003, thm. 7.4.9] pour une preuve utilisant la construction de Henkin.

Théorème 10.4 (complétude réfutationnelle)**.** *Soit S un ensemble de clauses. Si S est insatisfiable alors $S \vdash_{\mathbf{R}} \bot$.*

Démonstration. Supposons S insatisfiable. Par le corollaire 5.3 du théorème de Herbrand, $S\Sigma$ est aussi insatisfiable propositionnellement. Par la complétude de la résolution propositionnelle, $S\Sigma \vdash_{\mathbf{R}_0} \bot$. Par le corollaire 10.3 du lemme de relèvement, $S \vdash_{\mathbf{R}} \bot$ puisque $\bot$ est la seule clause qui s'instancie en $\bot$. □

10.3. **Complétude réfutationnelle par arbres sémantiques.** La preuve précédente suppose la complétude réfutationnelle de la résolution propositionnelle et le théorème de Herbrand connus. Nous allons voir ici une preuve plus directe, qui adapte la preuve de la complétude réfutationnelle de la résolution propositionnelle par arbres sémantiques ; à noter que nous utiliserons ici aussi le lemme 10.2 de relèvement et la propriété 5.2.

Soit S un ensemble de clauses insatisfiable et A l'ensemble des formules atomiques qui apparaissent dans S. Alors $A\Sigma$ l'ensemble de leur instanciations est dénombrable, et on fixe $\alpha_1, \alpha_2, \ldots$ une énumération sans répétition de $A\Sigma$.

*Dans les notes de la leçon **916**, on utilise une construction plus générale où toutes les branches de l'arbre binaire n'utilisent pas forcément les instanciations dans $A\Sigma$ dans le même ordre pour toutes les branches.*

Considérons l'arbre binaire, généralement infini, de hauteur $|A\Sigma|$. Un nœud de profondeur n est associé à une séquence dans $\mathbb{B}^n$ dénotant le chemin de la racine au nœud, et peut être vu comme une interprétation *partielle* de Herbrand $H : \{\alpha_1, \ldots, \alpha_n\} \to \mathbb{B}$; une branche de longueur $|A\Sigma|$ dénote alors un interprétation de Herbrand. Pour une interprétation partielle H, on écrit $H \vDash C$ s'il existe une interprétation H' compatible avec H telle que $H' \vDash C$. On appelle un *nœud d'échec* pour $C \in S$ un nœud minimal (le plus proche de la racine) d'interprétation partielle H telle que $H \nvDash C$.

Comme S est insatisfiable, pour toute interprétation H, il existe une clause $C \in S$ telle que $H \nvDash C$. Donc toutes les branches de l'arbre binaire contiennent un nœud d'échec. L'*arbre sémantique fermé* de S est l'arbre binaire élagué dès que l'on rencontre un nœud d'échec. Par le lemme de Kőnig, c'est un arbre fini.

On appelle *nœud d'inférence* un nœud dont les deux fils sont des nœuds d'échec.

Propriété 10.5. *Si l'arbre sémantique fermé pour S n'est pas réduit à sa seule racine, alors il contient un nœud d'inférence.*

Démonstration. Cela vaut en réalité pour tout arbre binaire fini non réduit à sa seule racine : il contient au moins un nœud dont les deux fils sont des feuilles. Cela se vérifie par induction sur la taille N de l'arbre. Pour le cas de base $N = 3$, les deux nœuds fils de la racine sont des feuilles. Pour l'étape d'induction $N > 3$, l'un des deux fils de la racine n'est pas une feuille, et enracine un sous-arbre de taille au plus $N - 2$, donc par hypothèse d'induction ce sous-arbre contient au moins un nœud dont les deux fils sont des feuilles. □

[Goubault-Larrecq et Mackie, 1997, p. 50 et p .242], [Chang et Lee, 1973, thm. 5.3]

Théorème 10.6 (complétude réfutationnelle)**.** *Soit S un ensemble de clauses. Si S est insatisfiable alors $S \vdash_{\mathbf{R}} \bot$.*

Démonstration. Soit S insatisfiable. On montre qu'en ajoutant à S un nombre fini de clauses C telles que $S \vdash_{\mathbf{R}} C$, on obtient un ensemble S' dont l'arbre sémantique fermé est réduit à sa seule racine, qui est alors un nœud d'échec. Mais alors, il existe une clause de S' qui n'est satisfiable dans aucune interprétation : ce ne peut être que la clause vide. Donc $S \vdash_{\mathbf{R}} \bot$.

Par définition, si l'arbre sémantique fermé de S n'est pas réduit à sa seule racine, par la propriété 10.5 il contient un nœud d'inférence à une certaine profondeur n, qui correspond à une interprétation partielle $H : \{\alpha_1, \ldots, \alpha_n\} \to \mathbb{B}$. Donc il existe une clause $C_1 \in S$ pour laquelle $H[\bot/\alpha_{n+1}] \nvDash C_1$ et une clause $C_2 \in S$ pour laquelle $H[\top/\alpha_{n+1}] \nvDash C_2$. Cependant, le nœud d'inférence n'est pas un nœud d'échec, donc $H \vDash C_1$ et $H \vDash C_2$.

Cela signifie qu'il y a un littéral $\ell \in C_1$ qui est satisfait par H mais pas par $H[\bot/\alpha_{n+1}]$; ce littéral est donc de la forme $\forall x_1 \ldots \forall x_m . R(x_1, \ldots, x_m)$ avec $\alpha_{n+1} = R(t_1, \ldots, t_m)$ pour des

termes clos $t_1, \dots, t_m \in D_H$ et $R \in \mathcal{P}_m$. Alors $\alpha_{n+1} = \ell\sigma_1$ pour $\sigma_1 \stackrel{\text{def}}{=} [t_1/x_1, \dots, t_m/x_m]$. De même, il y a un littéral $\ell' \in C_2$ tel que $\neg\alpha_{n+1} = \ell'\sigma_2$ pour σ_2 une substitution close.

Soient $C'_1 \stackrel{\text{def}}{=} C_1\sigma_1$ et $C'_2 \stackrel{\text{def}}{=} C_2\sigma_2$. Alors $C'_1, C'_2 \vdash_{\mathbf{R}_0} C'$ par résolution sur la formule atomique close α_{n+1}. Par le lemme 10.2 de relèvement, il existe donc C tel que $C_1, C_2 \vdash_{\mathbf{R}} C$ et $C' \in C\Sigma$.

On observe que $H \not\models C'$: en effet, $H \not\models C'_1 \setminus \{\alpha_{n+1}\}$ et $H \not\models C'_2 \setminus \{\neg\alpha_{n+1}\}$, sans quoi on aurait $H[\bot/\alpha_{n+1}] \models C'_1$ alors que $\alpha_{n+1} \in C'_1$, ou $H[\top/\alpha_{n+1}] \models C'_2$ alors que $\neg\alpha_{n+1} \in C'_2$. Par la propriété 5.2 appliquée à l'ensemble $\{C\}$, on a donc $H \not\models C$. Donc le nœud d'inférence devient un nœud d'échec de $S \cup \{C\}$, et l'arbre sémantique fermé associé à $S \cup \{C\}$ est strictement plus petit que celui associé à S. □

10.4. **Compacité.** Comme dans le cas propositionnel vu dans la leçon **916**, la correction et la complétude réfutationnelle de la résolution fournissent une preuve du théorème de compacité.

[Goubault-Larrecq et Mackie, 1997, thm. 6.21], [David et al., 2003, thm. 2.5.21]

Théorème 10.7 (compacité)**.** *Soit S un ensemble insatisfiable de formules du premier ordre. Alors il existe un sous-ensemble fini F de S qui est déjà insatisfiable.*

Démonstration. En mettant les formules de S sous forme clausale par le théorème 4.6, on obtient un ensemble insatisfiable $S' \stackrel{\text{def}}{=} \mathrm{Cl}(S)$ de clauses. Par le théorème 10.4 de complétude réfutationnelle, $S' \vdash_{\mathbf{R}} \bot$. Il existe donc un sous-ensemble fini $F' \subseteq_{\text{fin}} S'$ de clauses décorant les feuilles de cette dérivation tel que $F' \vdash_{\mathbf{R}} \bot$. Par le théorème 10.1 de correction, F' est insatisfiable. Comme F' est fini, il existe un sous-ensemble fini $F \subseteq_{\text{fin}} S$ tel que $F' \subseteq \mathrm{Cl}(F)$; un tel ensemble F est bien insatisfiable. □

Annexes. Unification

⚠ *Ces annexes sont tirées de la leçon **919** « Unification : algorithmes et applications », qui n'est plus au programme de l'agrégation. Il serait du coup maladroit de proposer un développement pour la leçon **918** sur l'unification. Il est cependant nécessaire de connaître les définitions et de savoir unifier des termes pour comprendre le système de preuve par résolution, et votre plan peut certainement mentionner des éléments de ces annexes.*
🔖 *La référence principale pour ces annexes est [Baader et Nipkow, 1998].*

Deux termes t et t' de $T(\mathcal{F}, X)$ s'*unifient* s'il existe une substitution σ telle que $t\sigma = t'\sigma$, appelée leur *unificateur*. On définit le pré-ordre « $\precsim$ » entre substitutions par $\sigma \precsim \sigma'$ s'il existe une substitution τ telle que $\sigma' = \sigma\tau$; on dira que σ est plus *générale* que σ'. On peut montrer que s'il existe un unificateur de t et t', alors il en existe un *plus général* (à renommage $\sim \stackrel{\text{def}}{=} \precsim \cap \succsim$ près), noté $\text{mgu}(t \stackrel{?}{=} t')$. Plus généralement, on pourra considérer l'unification d'un ensemble fini E d'équations $\{t_1 \stackrel{?}{=} t'_1, \dots, t_n \stackrel{?}{=} t'_n\}$ (existe-t'il σ telle que $t_i\sigma = t'_i\sigma$ pour tout $1 \leq i \leq n$?) ou d'un ensemble T de termes (existe-t'il σ telle que $t\sigma = t'\sigma$ pour tous $t, t' \in T$?), dont on notera les unificateurs les plus généraux $\text{mgu}(E)$ et $\text{mgu}(T)$ respectivement.

L'unification entre formules sans quantificateurs du premier ordre, qui est utilisée dans le système de preuve par résolution, est définie comme celle sur les termes sur un alphabet $\mathcal{F} \cup \{\vee, \wedge, \neg\} \cup \mathcal{P}$ où $\vee$ et $\wedge$ sont binaires et $\neg$ unaire.

Le problème d'unification consiste à déterminer si deux termes donnés sont unifiables, et le cas échéant à retourner leur mgu. L'algorithme « naïf » pour ce problème est celui de Robinson et peut nécessiter un temps exponentiel. Le problème peut en fait être résolu en temps déterministe linéaire par un algorithme dû à Paterson et Wegman [1978], mais celui-ci est assez complexe. Nous allons voir un algorithme simple dû à Huet [1976, sec. 5.7], basé sur `union-find`, qui travaille en temps déterministe presque linéaire.

Annexe A. Algorithme de Robinson

🔖 *[Baader et Nipkow, 1998, sec. 4.6], [David et al., 2003, sec 7.2], [Goubault-Larrecq et Mackie, 1997, sec. 7.2].*

L'algorithme de Robinson est basé sur le système de réécriture de la figure 5 qui agit sur des ensembles finis E d'équations. Dans ce système de réécriture, les équations $t \stackrel{?}{=}{}' t$ sont vues comme commutatives. Pour un ensemble d'équations $E = \{t_1 \stackrel{?}{=} t'_1, \dots, t_n \stackrel{?}{=} t'_n\}$, on note $\text{fv}(E) \stackrel{\text{def}}{=} \bigcup_{1 \leq i \leq n} \text{fv}(t_i) \cup \text{fv}(t'_i)$, et si σ est une substitution, on note $E\sigma \stackrel{\text{def}}{=} \{t_1\sigma \stackrel{?}{=} t'_1\sigma, \dots, t_n\sigma \stackrel{?}{=} t'_n\sigma\}$.

$$E \cup \{t \stackrel{?}{=} t\} \to_u E \tag{eff}$$

$$E \cup \{f(t_1, \dots, t_n) \stackrel{?}{=} f(t'_1, \dots, t'_n)\} \to_u E \cup \{t_1 \stackrel{?}{=} t'_1, \dots, t_n \stackrel{?}{=} t'_n\} \tag{dec}$$

$$E \cup \{x \stackrel{?}{=} t\} \to_u E[t/x] \cup \{x \stackrel{?}{=} t\} \quad \text{si } x \in \text{fv}(E), x \notin \text{fv}(t) \text{ et } (t \notin X \text{ ou } t \in \text{fv}(E)) \tag{elim}$$

Figure 5. Les règles de l'algorithme de Robinson.

On dit qu'une variable x est *résolue* dans E si $E = \{x \stackrel{?}{=} t\} \cup E'$ et $x \notin (\text{fv}(t) \cup \text{fv}(E'))$. Un ensemble d'équations est en *forme résolue* s'il est de la forme $\{x_1 \stackrel{?}{=} t_1, \dots, x_n \stackrel{?}{=} t_n\}$ où les variables $x_1, \dots, x_n$ sont résolues. On dénote par $\mathcal{U}(E)$ l'ensemble des unificateurs de E : $\mathcal{U}(\{t_1 \stackrel{?}{=} t'_1, \dots, t_n \stackrel{?}{=} t'_n\}) \stackrel{\text{def}}{=} \{\sigma \mid \forall 1 \leq i \leq n \,.\, t_i\sigma = t'_i\sigma\}$.

🔖 *[Baader et Nipkow, 1998, lem. 4.6.2].*

Lemme A.1. *Si $E = \{x_1 \stackrel{?}{=} t_1, \dots, x_n \stackrel{?}{=} t_n\}$ où $x_1, \dots, x_n$ sont résolues, alors pour tout $\sigma \in \mathcal{U}(E)$, $\sigma = [t_1/x_1, \dots, t_n/x_n]\sigma$.*

Démonstration. Soit $\sigma \in \mathcal{U}(E)$. Montrons que pour toute variable $x \in X$, $x[t_1/x_1, \dots, t_n/x_n]\sigma = x\sigma$. Si $x = x_i$ pour un certain $1 \leq i \leq n$, alors d'une part $x_i[t_1/x_1, \dots, t_n/x_n]\sigma = t_i\sigma$, et d'autre part $x_i\sigma = t_i\sigma$ puisque $\sigma \in \mathcal{U}(E)$. Si $x \notin \{x_1, \dots, x_n\}$, alors $x[t_1/x_1, \dots, t_n/x_n]\sigma = x\sigma$. □

🔖 *[Baader et Nipkow, 1998, lem. 4.6.3].*

Corollaire A.2 (formes résolues)**.** *Si $E = \{x_1 \stackrel{?}{=} t_1, \dots, x_n \stackrel{?}{=} t_n\}$ où $x_1, \dots, x_n$ sont résolues, alors* $\text{mgu}(E) = [t_1/x_1, \dots, t_n/x_n]$.

Démonstration. Posons $\sigma_E \stackrel{\text{def}}{=} [t_1/x_1, \dots, t_n/x_n]$. Alors σ_E est un unificateur : pour tout $1 \leq i \leq n$, $x_i\sigma_E = t_i = t_i\sigma_E$, où la dernière égalité est justifiée par $\text{dom}(\sigma)_E \cap \text{fv}(\{t_1, \dots, t_n\}) = \emptyset$.

Soit maintenant un unificateur $\sigma \in \mathcal{U}(E)$. Par le lemme A.1, $\sigma_E\sigma = \sigma$ donc $\sigma_E \precsim \sigma$. On a bien $\sigma_E = \text{mgu}(E)$. □

Proposition A.3 (terminaison). *Le système de la figure 5 termine : il n'y a pas de séquence infinie* $E_0 \to_u E_1 \to_u E_2 \to_u \cdots$.

■ *[Baader et Nipkow, 1998, lem. 4.6.5].*

Démonstration. On définit une fonction de rang r des ensembles finis d'équations dans $\mathbb{N}^2$ telle que, si $E \to_u E'$ alors $r(E) >_{\text{lex}} r(E')$ où $<_{\text{lex}}$ est l'ordre lexicographique sur $\mathbb{N}^2$. Alors l'existence d'une séquence infinie $E_0 \to_u E_1 \to_u E_2 \to_u \cdots$ impliquerait celle d'une descente infinie $r(E_0) >_{\text{lex}} r(E_1) >_{\text{lex}} r(E_2) >_{\text{lex}} \cdots$ dans $\mathbb{N}^2$, en contradiction avec le fait que $<_{\text{lex}}$ est bien fondé.

On considère pour cela l'ensemble $\text{nrv}(E) \stackrel{\text{def}}{=} \{x \in \text{fv}(E) \mid x \text{ non résolue}\}$ des variables non résolues de E et la taille $|E| \stackrel{\text{def}}{=} \sum_{t \stackrel{?}{=} t' \in E} |t| + |t'|$ de E. Posons $r(E) \stackrel{\text{def}}{=} (|\text{nrv}(E)|, |E|)$. Il reste à montrer que r est une fonction de rang, c'est-à-dire que $E \to_u E'$ implique $r(E) >_{\text{lex}} r(E')$. On procède pour cela à une distinction de cas.

Cas de (eff) : alors $\text{nrv}(E') \subseteq \text{nrv}(E)$ et $|E'| = |E| - 2|t|$ et donc $r(E') <_{\text{lex}} r(E)$.

Cas de (dec) : alors $\text{nrv}(E') \subseteq \text{nrv}(E)$ et $|E'| = |E| - 2$ et donc $r(E') <_{\text{lex}} r(E)$.

Cas de (elim) : alors $E = F \cup \{x \stackrel{?}{=} t\}$ et $E' = F[t/x] \cup \{x \stackrel{?}{=} t\}$ où $x \in \text{fv}(F) \setminus \text{fv}(t)$ et $t \notin X$ ou $t \in \text{fv}(F)$. Montrons que $\text{nrv}(E') \subsetneq \text{nrv}(E)$ et donc que $r(E') <_{\text{lex}} r(E)$.

D'une part $x \in \text{fv}(F)$ donc $x \in \text{nrv}(E)$, et d'autre part $x \notin \text{fv}(F[t/x])$ et $x \notin \text{fv}(t)$ donc $x \notin \text{nrv}(E')$.

Considérons maintenant une variable $y \in \text{nrv}(E')$, donc telle que $x \neq y$, et montrons que $y \in \text{nrv}(E)$. Par l'absurde, supposons que $y \in \text{nrv}(E')$ mais que $y \notin \text{nrv}(E)$, c'est-à-dire que $F = F' \cup \{y \stackrel{?}{=} t'\}$ où $y \notin \text{fv}(t') \cup \text{fv}(F') \cup \text{fv}(t)$. On a alors $F[t/x] = F'[t/x] \cup \{y \stackrel{?}{=} t'[t/x]\}$ puisque $x \neq y$. Comme $x \neq y$ et $y \notin \text{fv}(t') \cup \text{fv}(F') \cup \text{fv}(t)$, $y \notin \text{fv}(t'[t/x]) \cup \text{fv}(F'[t/x]) \cup \text{fv}(t) \cup \{x\} = \text{fv}(t'[t/x]) \cup \text{fv}(F'[t/x] \cup \{x \stackrel{?}{=} t\})$ donc y est résolue dans E', contradiction. □

Proposition A.4 (correction). *Les règles de la figure 5 sont correctes : si* $E \to_u E'$, *alors* $\mathcal{U}(E) = \mathcal{U}(E')$.

■ *[Baader et Nipkow, 1998, lem. 4.6.6].*

Démonstration. On procède par analyse de cas, selon la règle de la figure 5 employée.

Cas de (eff) : alors $E = E' \cup \{t \stackrel{?}{=} t\}$. On a $\sigma \in \mathcal{U}(E)$ si et seulement si $t\sigma = t\sigma$ et $\sigma \in \mathcal{U}(E')$, ce qui est si et seulement si $\sigma \in \mathcal{U}(E')$.

Cas de (dec) : alors $E = F \cup \{f(t_1, \ldots, t_n) \stackrel{?}{=} f(t'_1, \ldots, t'_n)\}$ et $E' = F \cup \{t_1 \stackrel{?}{=} t'_1, \ldots, t_n \stackrel{?}{=} t'_n\}$. On a les équivalences

$$\begin{aligned}\sigma \in \mathcal{U}(E) &\text{ ssi } f(t_1, \ldots, t_n)\sigma = f(t'_1, \ldots, t'_n)\sigma \text{ et } \sigma \in \mathcal{U}(F)\\ &\text{ssi } t_1\sigma = t'_1\sigma, \ldots, t_n\sigma = t'_n\sigma \text{ et } \sigma[t/x] \in \mathcal{U}(F)\\ &\text{ssi } \sigma \in \mathcal{U}(E') \ .\end{aligned}$$

Cas de (elim) : alors $E = F \cup \{x \stackrel{?}{=} t\}$ et $E' = F[t/x] \cup \{x \stackrel{?}{=} t\}$ avec $x \in \text{fv}(F) \setminus \text{fv}(t)$ et $t \notin X$ ou $t \in \text{fv}(F)$. Comme $x \notin \text{fv}(t)$, $\{x \stackrel{?}{=} t\}$ est un ensemble d'équations sous forme résolue. Donc par le lemme A.1, si $x\sigma = t\sigma$ alors $\sigma = [t/x]\sigma$. On a alors les équivalences

$$\begin{aligned}\sigma \in \mathcal{U}(E) &\text{ ssi } x\sigma = t\sigma \text{ et } \sigma \in \mathcal{U}(F)\\ &\text{ssi } x\sigma = t\sigma \text{ et } [t/x]\sigma \in \mathcal{U}(F)\\ &\text{ssi } x\sigma = t\sigma \text{ et } \sigma \in \mathcal{U}(F[t/x])\\ &\text{ssi } \sigma \in \mathcal{U}(E') \ .\end{aligned}$$

□

[Baader et Nipkow, 1998, lem. 4.6.10].

Théorème A.5 (Robinson)**.** *Soit E un ensemble fini d'équations. On peut décider si E est unifiable, et le cas échéant, calculer* $\mathrm{mgu}(E)$.

Démonstration. Étant donné E, on applique les règles de la figure 5 dans un ordre quelconque. Par la proposition A.3, on obtient après un nombre fini d'étapes un système d'équations E' tel que $E \to_u^* E'$ et aucune règle ne s'applique à E'. Par correction, $\mathcal{U}(E) = \mathcal{U}(E')$. On distingue tout d'abord deux cas particuliers.

(1) Si $(t \stackrel{?}{=} t') \in E'$ avec $t, t' \notin X$, alors comme (dec) ne s'applique pas, $t = f(t_1, \dots, t_n)$ et $t' = g(t'_1, \dots, t'_m)$ avec $f \neq g$, mais alors $\mathcal{U}(E') = \mathcal{U}(E) = \emptyset$.

(2) Si $(x \stackrel{?}{=} t) \in E'$ avec $x \in \mathrm{fv}(t)$, alors $\mathcal{U}(E') = \mathcal{U}(E) = \emptyset$.

Supposons maintenant qu'aucun des deux cas précédents ne s'applique et montrons que E' est nécessairement en forme résolue. Observons que E' ne contient donc que des équations de la forme $x \stackrel{?}{=} t$ pour $x \in X$ et $x \notin \mathrm{fv}(t)$. Soit $x \stackrel{?}{=} t$ une telle équation ; écrivons E' comme $E' = \{x \stackrel{?}{=} t\} \uplus E''$. Comme (elim) ne s'applique pas et $x \notin \mathrm{fv}(t)$, seuls deux cas sont possibles :
- si $x \notin \mathrm{fv}(E'')$, alors x est résolue ;
- sinon t est une variable $y \notin \mathrm{fv}(E'')$ et comme $x \notin \mathrm{fv}(y)$, $y \notin \mathrm{fv}(x)$: alors y est résolue.

Ce raisonnement s'applique à toutes les équations de E', qui est donc en forme résolue. Or, par le corollaire A.2, E' est alors unifiable et on peut calculer $\mathrm{mgu}(E')$; comme $\mathcal{U}(E) = \mathcal{U}(E')$ par la proposition A.4, on a finalement $\mathrm{mgu}(E') = \mathrm{mgu}(E)$. □

Exemple A.6. Soit le problème d'unification $f(x, y) \stackrel{?}{=} f(g(y), g(z))$. On a la réécriture suivante dans le système de la figure 5 :

$$\{f(x,y) \stackrel{?}{=} f(g(y),g(z))\} \xrightarrow{\text{(dec)}}_u \{x \stackrel{?}{=} g(y), y \stackrel{\text{g}}{=} (z)\} \xrightarrow{\text{(elim)}}_u \{x \stackrel{?}{=} g(g(z)), y \stackrel{?}{=} g(z)\}$$

qui est une forme résolue d'où l'on extrait $\mathrm{mgu}(f(x, y) \stackrel{?}{=} f(g(y), g(z))) = [g(g(y))/x, g(z)/y]$.

Exemple A.7. Soit le problème d'unification $f(x, y) \stackrel{?}{=} f(g(y), g(x))$. On a la réécriture suivante dans le système de la figure 5 :

$$\{f(x,y) \stackrel{?}{=} f(g(y),g(z))\} \xrightarrow{\text{(dec)}}_u \{x \stackrel{?}{=} g(y), y \stackrel{?}{=} g(x)\} \xrightarrow{\text{(elim)}}_u \{x \stackrel{?}{=} g(y), y \stackrel{?}{=} g(g(y))\}\ .$$

Aucune règle ne s'applique à ce dernier système d'équations qui n'est pas en forme résolue, et en effet les termes n'étaient pas unifiables.

[Baader et Nipkow, 1998, ex. 4.6.11].

Exemple A.8. Soit le problème d'unification $\{x_1 \stackrel{?}{=} f(x_2, x_2), x_2 \stackrel{?}{=} f(x_3, x_3), x_3 \stackrel{?}{=} f(x_4, x_4)\}$. On a la réécriture suivante dans le système de la figure 5 :

$$\begin{aligned}
&\{x_1 \stackrel{?}{=} f(x_2,x_2), x_2 \stackrel{?}{=} f(x_3,x_3), x_3 \stackrel{?}{=} f(x_4,x_4)\}\\
&\xrightarrow{\text{(elim)}}_u \{x_1 \stackrel{?}{=} f(f(x_3,x_3),f(x_3,x_3)), x_2 \stackrel{?}{=} f(x_3,x_3), x_3 \stackrel{?}{=} f(x_4,x_4)\}\\
&\xrightarrow{\text{(elim)}}_u \{x_1 \stackrel{?}{=} f(f(f(x_4,x_4),f(x_4,x_4)),f(f(x_4,x_4),f(x_4,x_4))),\\
&\qquad\qquad x_2 \stackrel{?}{=} f(f(x_4,x_4),f(x_4,x_4)), x_3 \stackrel{?}{=} f(x_4,x_4)\}
\end{aligned}$$

qui est une forme résolue d'où l'on pourrait extraire le mgu. En généralisant cet exemple sur n variables, on peut observer que l'algorithme de Robinson peut avoir un coût exponentiel, qui est inévitable si l'on souhaite explicitement écrire le mgu.

Annexe B. Algorithme par `union-find`

B.1. **Treillis des substitutions.** Nous allons voir que l'ensemble des substitutions sur $T(\mathcal{F}, X)$ ordonné par $\precsim$ (une fois quotienté par $\sim$) forme un treillis complet. Nous allons construire pour cela une correspondance avec un sous-treillis du treillis des relations d'équivalence sur $T(\mathcal{F}, X)$.

On dénote par $\sqsubset$ l'ordre sous-terme strict sur $T(\mathcal{F}, X)$, c'est-à-dire la clôture transitive de la relation $\{(f(t_1, \ldots, t_m), t_i) \mid m \in \mathbb{N}, 1 \leq i \leq m, f \in \mathcal{F}_m\}$. Appelons une relation d'équivalence sur $T(\mathcal{F}, X)$ *valide* si elle satisfait les trois propriétés qui suivent :

finitaire : il existe un nombre fini de classes d'équivalence $[x]_\equiv$ pour $x \in X$ qui ne soient pas des singletons $\{x\}$,

homogène : $f(t_1, \ldots, t_m) \equiv g(t'_1, \ldots, t'_n)$ implique $f = g$ (et donc $m = n$) et $t_i \equiv t'_i$ pour tous $1 \leq i \leq m$, et

acyclique : la composition $\sqsubset \fatsemi \equiv$ est bien fondée,

où l'on définit la composition entre relations binaires R et R' par $R \fatsemi R' \stackrel{\text{def}}{=} \{(x, z) \mid \exists y\,.\,(x, y) \in R \text{ et } (y, z) \in R'\}$. À noter qu'une équivalence valide est nécessairement une congruence.

Propriété B.1. *L'ensemble des relations d'équivalence valides sur $T(\mathcal{F}, X)$ ordonné par inclusion forme un treillis complet.*

Démonstration. Il suffit d'observer que si $(\equiv_j)_{j \in J}$ est un ensemble de relations d'équivalence valides sur $T(\mathcal{F}, \mathcal{X})$, alors son intersection $\bigcap_{j \in J} \equiv_j$ est valide. □

B.1.1. *Des substitutions aux équivalences valides.* Soit σ une substitution. On définit la relation d'équivalence $\equiv_\sigma$ sur $T(\mathcal{F}, X)$ par $t \equiv_\sigma t'$ si $t\sigma = t'\sigma$.

Propriété B.2. *L'équivalence $\equiv_\sigma$ est valide.*

Démonstration. La relation $\equiv_\sigma$ est manifestement homogène ; elle est finitaire puisque les seules classes $[x]_{\equiv_\sigma} \neq \{x\}$ apparaissent pour $x \neq x\sigma$, c'est-à-dire pour $x \in \mathrm{dom}(\sigma)$, qui est fini. Enfin, $\sqsubset \fatsemi \equiv$ est bien fondée, sans quoi on aurait $t_0\sigma = s_0\sigma \sqsupset t_1\sigma = s_1\sigma \sqsupset \cdots$ une descente infinie dans $t_0\sigma$ par l'ordre de sous-terme strict. □

Propriété B.3. *Si $\sigma \precsim \sigma'$ alors $\equiv_\sigma \subseteq \equiv'_\sigma$.*

Démonstration. Soit τ une substitution telle que $\sigma' = \sigma\tau$. Alors pour tous termes t et t', $t\sigma = t'\sigma$ implique $t\sigma\tau = t'\sigma\tau$. □

B.1.2. *Des équivalences valides aux substitutions.* Soit $\equiv$ une relation d'équivalence valide. Pour toute classe d'équivalence e de $\equiv$, on choisit un représentant canonique $c(e)$ tel que, s'il existe un terme $t \in e \setminus X$ qui ne soit pas une variable, alors $c(e) \notin X$ ne soit pas une variable non plus. Il faut noter ici que le seul véritable choix est celui d'un représentant dans X quand $e \subseteq X$ n'est pas réduit à un singleton : l'homogénéité de $\equiv$ fait que le choix entre différents représentants canoniques quand $e \setminus X \neq \emptyset$ est immatériel.

On ordonne X par $x \prec y$ si $y \sqsubset c([x]_\equiv)$; comme $\equiv$ est acyclique, $\prec$ est bien fondé. On peut alors définir une fonction $\sigma_{c,\equiv} \colon X \to T(\mathcal{F}, X)$ par induction bien fondée sur $\prec$:

$$\sigma_{c,\equiv}(x) \stackrel{\text{def}}{=} \big(c([x]_\equiv)\big)\sigma_{c,\equiv}\ .$$

Propriété B.4. *La fonction $\sigma_{c,\equiv}$ est une substitution telle que, pour tous termes $t, t' \in T(\mathcal{F}, X)$, $t \equiv t'$ implique $t\sigma_{c,\equiv} = t'\sigma_{c,\equiv}$.*

Démonstration. La fonction $\sigma_{c,\equiv}$ est bien une substitution : $\mathrm{dom}(\sigma)_{c,\equiv}$ est fini puisque $\equiv$ est finitaire et donc $c[x]_\equiv = x$ pour un nombre cofini de variables x.

On montre que par induction bien fondée sur t, t' pour l'ordre produit induit par $\sqsubset \fatsemi \equiv$ que $t \equiv t'$ implique $t\sigma_{c,\equiv} = t'\sigma_{c,\equiv}$.

Cas $c([t]_\equiv), c([t']_\equiv) \notin X$: alors $c([t]_\equiv) = f(t_1, \ldots, t_m)$ et $c([t']_\equiv) = g(t_1, \ldots, t_n)$.

Si $t \equiv t'$ alors $c([t]_\equiv) \equiv c([t']_\equiv)$, et comme $\equiv$ est homogène, $f = g$, $m = n$, et $t_i \equiv t'_i$ pour tout $1 \leq i \leq m$. Par hypothèse d'induction, applicable car $t \equiv c([t]_\equiv) \sqsupset t_i$ et $t' \equiv c([t']_\equiv) \sqsupset t'_i$ pour tout $1 \leq i \leq m$, $t_i\sigma_{c,\equiv} = t'_i\sigma_{c,\equiv}$ pour tout $1 \leq i \leq m$ et donc $t\sigma_{c,\equiv} = t'\sigma_{c,\equiv}$.

Cas $c([t]_\equiv) \in X$ et $c([t']_\equiv) \notin X$: alors $t \not\equiv t'$ par définition de c.

Cas $c([t]_\equiv) \notin X$ et $c([t']_\equiv) \in X$: symétrique du cas précédent.

Cas $c([t]_\equiv), c([t']_\equiv) \in X$: alors $c([t]_\equiv) = x = c([x]_\equiv)$ et $c([t']_\equiv) = x' = c([x']_\equiv)$. Alors $t \equiv t'$ implique $[t]_\equiv = [t']_\equiv$ et donc $t\sigma_{c,\equiv} = x = x' = t'\sigma_{c,\equiv}$. □

Propriété B.5. *Si $\equiv \subseteq \cong$ sont valides et c est un choix de représentants canoniques pour $\equiv$, alors il existe c' un choix pour $\cong$ tel que $\sigma_{c,\equiv} \precsim \sigma_{c',\cong}$.*

Démonstration. Pour tout $x \in X$ tel que $c([x]_\equiv) \in X$ et $[x]_\cong \setminus X \neq \emptyset$, on choisit $c'([x]_\cong) \in [x]_\cong \setminus X$. Dans tous les autres cas on pose $c'([t]_\cong) \stackrel{\text{def}}{=} c([t]_\equiv)$.

On définit ensuite pour tout $x \in X$

$$\tau(x) \stackrel{\text{def}}{=} \begin{cases} c'([x]_\cong) & \text{si } c([x]_\equiv) = x \neq c'([x]_\cong) \\ x & \text{sinon.} \end{cases}$$

C'est bien une substitution puisque $\cong$ est valide. Il reste à vérifier que $\sigma_{c',\cong} = \sigma_{c,\equiv}\tau$. Il suffit de vérifier que $\sigma_{c',\cong}(x) = \tau(\sigma_{c,\equiv}(x))$ pour tout $x \in X$ tel que $c([x]_\equiv) \in X$ et $[x]_\cong \setminus X \neq \emptyset$. Mais alors $c([x]_\equiv) = y = c([y]_\equiv)$ et $\tau(y) = c'([y]_\cong)$ par définition, et $x \equiv y$ implique $x \cong y$ et donc $c'([y]_\cong) = c'([x]_\cong)$ comme désiré. □

B.1.3. *Lemme de* HUET. Les propriétés précédentes permettent de déduire le résultat suivant.

Lemme B.6 (HUET)**.** *Deux termes t et t' de $T(\mathcal{F}, X)$ sont unifiables si et seulement s'il existe une relation d'équivalence valide telle que $t \equiv t'$, auquel cas il en existe une minimale.*

Démonstration. La première partie de l'énoncé découle des propriétés B.2 et B.4. La seconde découle du fait que les relations d'équivalence valides forment un treillis complet par la propriété B.1. □

De plus, par les propriétés B.3 et B.5, le mgu$(t \stackrel{?}{=} t')$ n'est autre que la substitution associée à cette équivalence valide minimale, pour un certain choix de représentants canoniques – mais les mgu ne sont uniques qu'à renommage par $\sim$ près.

Remarque B.7 (congruence). Soient $t_1 \stackrel{\text{def}}{=} f(f(x_1, x_2), f(x_3, x_4))$ et $t_2 \stackrel{\text{def}}{=} f(f(x_2, x_3), f(x_4, x_1))$ et l'instance du problème d'unification $t_1 \stackrel{?}{=} t_2$. Un mgu est $\sigma = [x_1/x_2, x_1/x_3, x_1/x_4]$ qui rend $f(x_1, x_2)$, $f(x_2, x_3)$, $f(x_3, x_4)$ et $f(x_4, x_1)$ tous équivalents par $\equiv_\sigma$. Cependant, la plus petite relation d'équivalence valide $\equiv$ telle que $t_1 \equiv t_2$ a bien $f(x_1, x_2) \equiv f(x_2, x_3)$ et $f(x_3, x_4) \equiv f(x_4, x_1)$ par homogénéité, mais $f(x_1, x_2) \not\equiv f(x_3, x_4)$.

Plus généralement, pour toute substitution σ, $\equiv_\sigma$ est une *congruence* : $t_i \equiv_\sigma t'_i$ pour $1 \leq i \leq m$ implique $f(t_1, \ldots, t_m) \equiv_\sigma f(t'_1, \ldots, t'_m)$ pour tout $f \in \mathcal{F}_m$. On a alors une réciproque de la propriété B.4 : si $\equiv$ est une congruence valide, alors $t\sigma_{c,\equiv} = t'\sigma_{c,\equiv}$ implique $t \equiv t'$.

Enfin, si $\equiv$ est une relation d'équivalence valide et $\cong$ la plus petite congruence telle que $\equiv \subseteq \cong$, alors $\cong$ est valide et de plus on peut utiliser le même choix c de représentants canoniques pour les deux relations et $\sigma_{c,\equiv} = \sigma_{c,\cong}$.

[BAADER et NIPKOW, 1998, sec. 4.8] pour ce premier algorithme.

B.2. **Un premier algorithme.** Le principe de ce premier algorithme d'unification de t et t' est

(1) de mettre t et t' sous la forme d'un unique graphe dirigé acyclique où chaque variable n'apparaît qu'une seule fois ;

(2) de construire, s'il en existe une, la relation d'équivalence finitaire homogène minimale $\equiv$ telle que $t \equiv t'$;

(3) de tester si $\equiv$ est acyclique : si elle ne l'est pas, alors aucune relation plus grande ne peut l'être non plus donc il n'existe pas de relation valide pour laquelle t et t' sont équivalents.

On représente les classes d'équivalence avec les structures d'`union-find`. En particulier, le choix d'un représentant canonique pour chaque classe d'équivalence découle naturellement des structures de données mises en place par `union-find`. L'algorithme fonctionne en temps déterministe $O(n \lg n)$ où $n \stackrel{\text{def}}{=} |t| + |t'|$ est la taille de la représentation des termes d'entrée, qui peuvent être fournis sous forme arborescente ou de graphes dirigés acycliques.

Étape (1). Cette étape est typiquement implémentée à l'aide d'une table de hachage. Si l'on fait l'hypothèse d'un hachage parfait, cette étape est en $O(n)$.

Par exemple, le graphe de gauche de la figure 6 montre le résultat de l'étape (1) pour l'entrée $f(x, y) \stackrel{?}{=} f(g(y), g(z))$, si l'on ignore les arcs en violet.

Étape (2). Cette étape commence par initialiser son entrée en ajoutant un pointeur « parent » de chaque nœud du graphe vers lui-même : ces pointeurs sont indiqués en violet dans la figure 6 et dénotés par $x.p$ dans le pseudo-code. Le représentant canonique d'une classe d'équivalence est obtenu en suivant ces pointeurs jusqu'à trouver un élément qui pointe sur lui-même.

Cette étape fait appel aux procédures `union` et `find` définies ci-après : il s'agit ici d'une union naïve de classes d'équivalence et d'une recherche avec compression de chemins du représentant canonique d'une classe d'équivalence.

*[Beauquier et al., 1992, sec. 3.5], [Cormen et al., 2009, ch. 21]. Les procédures d'`union-find` en tant que telles sont normalement traitées dans la leçon **926**, voire la leçon **901**.*

Procédure `naive-union`(x, y)
1 $y.p := x$

Procédure `find-and-compress`(x)
1 **si** $x.p \neq x$ **alors**
2 $\quad x.p :=$ `find-and-compress`$(x.p)$
3 **retourner** $x.p$

La complexité de cette implémentation d'`union-find` est connue.

Théorème B.8 (Tarjan et van Leeuwen, 1984, thm. 4)**.** *L'algorithme `union-find` avec compression de chemin et union naïve sur n éléments effectue m appels à `find` en temps $\Theta(n + m \log_{2+m/n} n)$.*

L'étape (2) calcule ensuite récursivement la plus petite relation homogène qui rend t et t' équivalents. Par le théorème B.8, ce calcul termine en $O(n \lg n)$.

⚠ À la ligne 8, il est nécessaire de rendre t' représentant canonique pour la classe contenant $t \in X$ si $t' \notin X$; de même à la ligne 10 avec les rôles de t et t' échangés. Cela empêche d'utiliser les techniques d'union par rang ou par taille ; voir l'annexe B.3 pour une solution.

Procédure `homogenise`(t, t)
1 $t :=$ `find`(t)
2 $t' :=$ `find`(t')
3 **si** $t = f(t_1, \ldots, t_m), t' = g(t'_1, \ldots, t'_n)$ *et* $f \neq g$ **alors** *(non homogène !)*
4 $\quad$ **échec :** t *et* t' *non unifiables*
5 **sinon si** $t = f(t_1, \ldots, t_m)$ *et* $t' = f(t'_1, \ldots, t'_m)$ **alors**
6 $\quad$ `union`(t, t')
7 $\quad$ **pour** $i = 1$ **à** m **faire** `homogenise`(t_i, t'_i)
8 **sinon si** $t \in X$ **alors**
9 $\quad$ `union`(t', t)
10 **sinon** *(nécessairement $t' \in X$)*
11 $\quad$ `union`(t, t')

Exemple B.9. La figure 6 présente la seconde étape de l'algorithme d'unification sur l'entrée $f(x, y) \stackrel{?}{=} f(g(y), g(z))$ déjà rencontrée dans l'exemple A.6. Le mgu est $[g(g(z))/x, g(z)/y, z/z]$, lu en suivant les représentants canoniques de chaque variable.

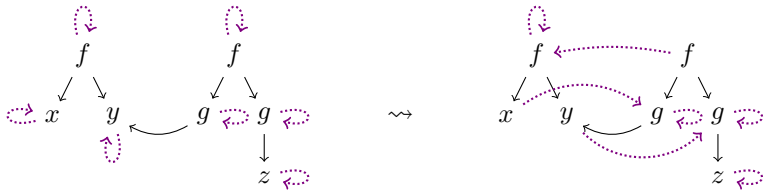

FIGURE 6. Étape (2) de l'algorithme d'unification sur les termes d'entrée $f(x, y)$ et $f(g(y), g(z))$.

Exemple B.10. La figure 7 présente la seconde étape de l'algorithme d'unification sur l'entrée $\bullet(x_1, x_2, x_3) \stackrel{?}{=} \bullet(f(x_2, x_2), f(x_3, x_3), f(x_4, x_4))$, qui encode l'exemple A.8. On retrouve bien le même résultat, mais la représentation du mgu est linéaire en la taille de l'entrée.

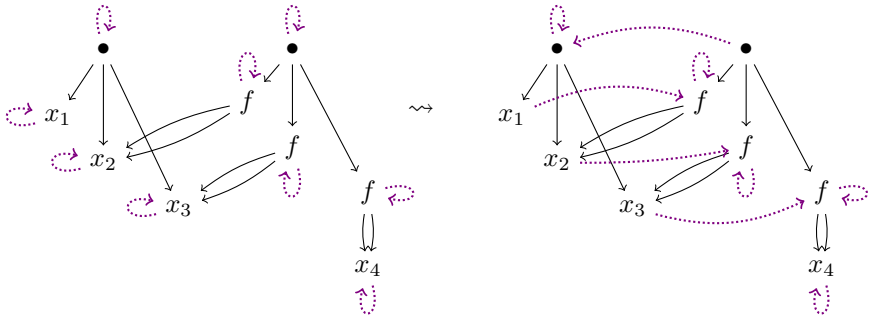

FIGURE 7. Étape (2) de l'algorithme d'unification sur les termes d'entrée $\bullet(x_1, x_2, x_3)$ et $\bullet(f(x_2, x_2), f(x_3, x_3), f(x_4, x_4))$.

Exemple B.11. La figure 8 présente la seconde étape de l'algorithme d'unification sur l'entrée $f(x, y) \stackrel{?}{=} f(g(y), g(z))$. Le résultat est la substitution $[g(z)/x, g(z)/y, z/z]$, lue en suivant les représentants canoniques de chaque variable.

FIGURE 8. Étape (2) de l'algorithme d'unification sur les termes d'entrée $f(x, y)$ et $f(y, g(z))$.

On peut remarquer sur cette exemple que, lors de l'appel $\texttt{union}(g, y)$ par $\texttt{homogenise}(y, g)$, la classe d'équivalence de y était $\{x, y\}$, plus grande pour les algorithmes d'union par rang ou par taille que la classe de g qui était $\{g\}$. Nous avons cependant été contraints de choisir g comme représentant canonique.

Étape (3). La dernière étape est de vérifier si la relation d'équivalence obtenue est bien acyclique. Il s'agit d'un simple test d'acyclicité dans le graphe obtenu, où l'on cherche un cycle non trivial utilisant les pointeurs « parents » *et* la relation de sous-terme. Ce test s'implémente en $O(n)$ par un parcours de graphe.

Exemple B.12. La figure 9 présente la seconde étape de l'algorithme d'unification sur l'entrée $f(x, y) \stackrel{?}{=} f(g(y), g(x))$ déjà rencontrée dans l'exemple A.7. La troisième étape échoue car il y a une cycle de x à lui-même utilisant au moins un arc de relation sous-terme ; les termes n'étaient en effet pas unifiables.

FIGURE 9. Étape (2) de l'algorithme d'unification sur les termes d'entrée $f(x, y)$ et $f(g(y), g(x))$.

B.3. **Algorithme d'unification au premier ordre de HUET.** Des améliorations de ce premier algorithme permettant une union par rang ou par taille sont possibles, avec une complexité finale en $O(n \cdot \alpha(n, n))$ où α est la fonction d'ACKERMANN inverse. Nous allons voir la version donnée par HUET. Les trois étapes de l'algorithme restent les mêmes, mais la structure sous-jacente pour `union-find` à l'étape (2) est différente, et les procédures `union` et `find` doivent être mises à jour.

[HUET, 1976, sec. 5.7] pour cet algorithme d'unification rapide.

Structure de données. En plus du pointeur « parent » $x.p$, chaque nœud a aussi un booléen $x.r$ indiquant s'il est à la *racine* de la structure d'`union-find` ; au début de l'étape (2), ce booléen est initialement 1. Le représentant canonique d'une classe est maintenant le nœud pointé par $x.p$ tel que $x.r = 1$. Chaque nœud possède également un attribut de « taille » $x.t$, initialement 1.

Procédure `find`. La procédure `find-and-compress` fait appel à une procédure auxiliaire `root` pour retourner le nœud racine ; `find-and-compress` retourne alors le parent du nœud racine, qui est le représentant canonique de la classe.

Procédure `root`(x)
1 **si** $\neg x.r$ **alors**
2 $\quad x.p :=$ `root`$(x.p)$
3 **retourner** $x.p$

Procédure `find-and-compress`(x)
1 **retourner** `root`$(x).p$

Procédure `union`. La procédure `union-by-size` reçoit en argument des nœuds x et y qui sont les *représentants canoniques* de leurs classes respectives, mais pas nécessairement les *racines* de leurs classes. On commence donc par chercher ces racines aux lignes 1 et 2. La nouvelle racine sera la racine de la classe de plus grande taille (voir les lignes 4 et 8). Enfin, le représentant canonique

de la nouvelle classe sera pointée par cette racine de l'union (rien à faire aux lignes 4–6, mais voir la ligne 9).

Procédure `union-by-size`(x,y)

```
1  x := root(x)
2  y := root(y)
3  si x.t ≥ y.t alors
4  |  y.r := 0
5  |  y.p := x
6  |  x.t := x.t + y.t
7  sinon
8  |  x.r := 0
9  |  y.p := x.p
10 |  x.p := y
11 |  y.t := x.t + y.t
```

Exemple B.13. Reprenons l'exemple B.11. La figure 10 présente la seconde étape de l'algorithme d'unification de Huet sur l'entrée $f(x,y) \stackrel{?}{=} f(g(y),g(z))$.

Initialement, tous les nœuds ont $x.r = 1$, et sont indiqués en orange. Les tailles $x.t$ sont indiquées en vert à coté de chaque nœud.

Le résultat est la substitution $[g(z)/x, g(z)/y, z/z]$, lue en suivant dans la figure de droite les représentants canoniques de chaque variable : on remonte jusqu'à la racine de la classe indiquée en orange, et on suit son lien parent en violet pour trouver le représentant canonique.

Figure 10. Étape (2) de l'algorithme d'unification de Huet sur les termes d'entrée $f(x,y)$ et $f(y,g(z))$.

Exemple B.14. Reprenons l'exemple B.12. La figure 11 présente la seconde étape de l'algorithme d'unification de Huet sur l'entrée $f(x,y) \stackrel{?}{=} f(g(y),g(x))$. La troisième étape échoue comme il se doit, car il y a une cycle de x à lui-même utilisant au moins un arc de relation sous-terme ; les termes n'étaient en effet pas unifiables.

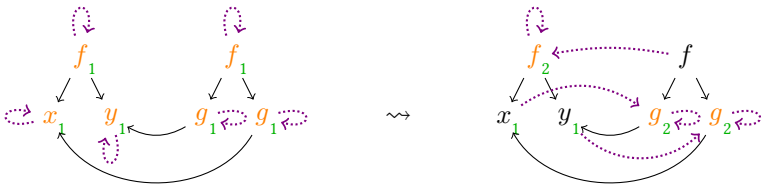

Figure 11. Étape (2) de l'algorithme d'unification de Huet sur les termes d'entrée $f(x,y)$ et $f(g(y),g(x))$.

La complexité de cette implémentation d'`union-find` est elle aussi connue, et donne au final une complexité en $O(n \cdot \alpha(n, n))$ pour l'algorithme d'unification de Huet.

Théorème B.15. *L'algorithme `union-find` avec compression de chemin et union par taille sur n éléments effectue m appels à `find` en temps $\Theta(m \cdot \alpha(m, n))$.*

Voir [Beauquier et al., 1992, thm. 5.3], ou [Cormen et al., 2009, thm. 21.14] pour une version avec union par rang.

Références

Franz Baader et Tobias Nipkow, 1998. *Term Rewriting and All That*. Cambridge University Press. doi:10.1017/CBO9781139172752.

Danièle Beauquier, Jean Berstel et Chrétienne Philippe, 1992. *Éléments d'algorithmique*. Masson, Paris. URL http://www-igm.univ-mlv.fr/~berstel/Elements/Elements.html.

Chin-Liang Chang et Richard Char-Tung Lee, 1973. *Symbolic Logic and Mechanical Theorem Proving*. Computer Science and Applied Mathematics. Academic Press, New York. doi:10.1016/B978-0-08-091728-3.50001-1.

Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest et Clifford Stein, 2009. *Introduction to Algorithms*. MIT Press, Cambridge, Massachusetts, USA, 3e édition.

René David, Karim Nour et Christophe Raffalli, 2003. *Introduction à la logique*. Dunod, Paris, 2e édition.

Heinz-Dieter Ebbinghaus, Jörg Flum et Wolfgang Thomas, 1994. *Mathematical Logic*. Undergraduate Texts in Mathematics. Springer, New York, 2e édition. doi:10.1007/978-1-4757-2355-7.

Melvin Fitting, 1996. *First-Order Logic and Automated Theorem Proving*. Graduate Texts in Computer Science. Springer, New York, 2e édition. doi:10.1007/978-1-4612-2360-3.

Jean Goubault-Larrecq et Ian Mackie, 1997. *Proof Theory and Automated Deduction*, volume 6 de *Applied Logic Series*. Kluwer Academic Publishers, Amsterdam.

Gérard Huet, 1976. *Résolution d'équations dans les langages d'ordre* $1, 2, \ldots, \omega$. Thèse de doctorat d'état, Université Paris VII.

Richard Lassaigne et Michel de Rougemont, 2004. *Logic and Complexity*. Discrete Mathematics and Theoretical Computer Science. Springer, London. doi:10.1007/978-0-85729-392-3.

Michael S. Paterson et Mark N. Wegman, 1978. Linear unification. *Journal of Computer and System Sciences*, 16(2):158–167. doi:10.1016/0022-0000(78)90043-0.

Robert E. Tarjan et Jan van Leeuwen, 1984. Worst-case analysis of set union algorithms. *Journal of the ACM*, 31(2):245–281. doi:10.1145/62.2160.