3.6 Systèmes de preuves

3.6.1 La notion de système formel

Un *système de preuves* est une définition inductive d'un ensemble permettant, en général de déduire des couples de la forme $\Gamma \vdash \Delta$ où Γ et Δ sont des listes (non ordonnées) de formules. Les règles de la définition inductive tentent de simuler le raisonnement.

On note $\Gamma \vdash_L \Delta$ si $\Gamma \vdash \Delta$ peut se déduire grâce au système de preuves L.

Notation 3.56. Dans ce qui suit, on notera $\varphi_1, \dots, \varphi_n \models \psi_1, \dots, \psi_m$ si il existe un $i \in [1, m]$ tel que $\{\varphi_1, \dots, \varphi_n\} \models \psi_i$.

Deux questions fondamentales sont systématiquement posées pour relier le calcul et la sémantique dans une logique : existe-il un système de preuve L qui soit

1. **correct** : toute élément déduit est vrai : $\Gamma \vdash_L \Delta$ implique $\Gamma \models \Delta$;
2. **complet** : tout ce qui est vrai peut être déduit : $\Gamma \models \Delta$ implique $\Gamma \vdash_L \Delta$.

Nous présentons ici deux systèmes de preuve : le premier appelé **méthode de la résolution** est assez éloigné du raisonnement classique mais est remarquable pour sa simplicité car il ne contient que deux règles. Il permet de déduire un algorithme très simple pour déterminer si une formule est une tautologie. Le second est le **calcul des séquents** qui utilise un nombre plus important de règles mais simule de façon assez fidèle le raisonnement mathématique.

Pour information : hormis la méthode de la résolution, il ne vous est pas demandé de connaître les règles des différents systèmes par coeur, mais simplement d'être capable de réaliser des preuves et d'utiliser correctement les règles.

3.6.2 La méthode de la résolution

Introduite en 1965 par Robinson, la résolution est un système formel pour le calcul des prédicats qui utilise des formules sous formes de clauses ; nous présenterons ce système plus tard. Ici, nous allons présenter la méthode de la résolution, qui n'est rien d'autre que la restriction de la résolution à la logique propositionnelle.

Il ne permet de démontrer que des expressions de la forme $\Gamma \vdash \bot$ où Γ est une liste de clauses. Il permet donc de prouver que l'ensemble Γ est insatisfaisable.

Définition 3.57. Etant donné un ensemble de clauses Γ, l'ensemble $Resolution(\Gamma)$ est défini inductivement de la façon suivante :

pour toute variable propositionnelle p, tout littéral ℓ, toutes clauses C, C_1, C_2 :

$$\frac{}{C}\ C \in \Gamma \text{ (Axiome)} \qquad \frac{C \vee \ell \vee \ell}{C \vee \ell} \text{ (Factorisation)} \qquad \frac{C_1 \vee p \qquad C_2 \vee \neg p}{C_1 \vee C_2} \text{ Résolution}$$

Notez le cas particulier de la règle de résolution : $\dfrac{p \qquad \neg p}{\bot}$ qui est la seule façon de générer la clause $\bot$.

Elle peut se peut se réécrire comme $\dfrac{\bot \vee p \qquad \bot \vee \neg p}{\bot}$ Résolution

ce qui correspond donc bien (à factorisation près) à un cas particulier de la règle de résolution

Exemple 3.58. Soit $\mathcal{C} = \{p \vee r \vee s,\ r \vee \neg s,\ \neg r,\ \neg p\}$ un ensemble de clauses. Pour montrer que $\mathcal{C} \vdash_r \bot$, il suffit de construire une preuve de $\bot \in Resolution(\mathcal{C})$, en utilisant les clauses de $\mathcal{C}$ comme axiomes.

Voici une preuve possible.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{p \vee r \vee s} \qquad \dfrac{}{r \vee \neg s}}{p \vee r \vee r}\ \text{Résolution}}{p \vee r}\ \text{Factorisation} \qquad \dfrac{}{\neg r}}{p}\ \text{Résolution} \qquad \dfrac{}{\neg p}}{\bot}\ \text{Résolution}$$

Notation 3.59. On utilise les notations suivantes :

1. Pour tout ensemble de clauses $\mathcal{C}$, on note $\mathcal{C} \vdash_r \bot$ ssi $\bot \in$ *Resolution*$(\mathcal{C})$.
2. Par extension, si Γ est un ensemble de formules, on note $\Gamma \vdash_r \bot$ ssi $\bot \in$ *Resolution*$(\mathcal{C})$ pour un ensemble de clauses $\mathcal{C}$ équivalent à Γ.

Exemple 3.60. Soit $\mathcal{C}$ l'ensemble de clauses donné à l'exemple 3.58, on a $\mathcal{C} \vdash_r \bot$.

Théorème 3.61 (Complétude et Correction)**.** *Pour tout ensemble* Γ *:*

$$\Gamma \vdash_r \bot \quad \textit{ssi } \Gamma \textit{ est insatisfaisable.}$$

Avant de démontrer ce théorème, détaillons quelques exemples.

Exemple 3.62. Soit $\varphi = ((p \Rightarrow q) \wedge (q \Rightarrow p) \wedge (p \vee q)) \Rightarrow (p \wedge q)$.
Montrons que φ est est une tautologie ($\models \varphi$).
Nous allons procéder en 2 étapes :

1. mettre $\neg\varphi$ sous forme d'un ensemble de clauses $\mathcal{C}$
2. montrer que $\mathcal{C} \vdash_C \bot$ en dérivant la clause $\bot$ à partir des clauses de $\mathcal{C}$.

L'ensemble de clause est $\mathcal{C} = \{p \vee \neg q, \neg p \vee q, p \vee q, \neg p \vee \neg q\}$. Une preuve par la méthode de la résolution est la suivante :

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{p \vee \neg q \qquad \neg q \vee \neg p}{\neg q \vee \neg q}\ \text{résolution}}{\neg q}\ \text{factorisation} \qquad \neg p \vee q}{\neg p}\ \text{résolution} \qquad p \vee q}{q}\ \text{résolution} \qquad \neg q}{\bot}\ \text{résolution}$$

Donc nous avons montré qu'on peut dériver $\bot$ à partir de la clause vide, ce qui implique que $\neg\varphi$ est insatisfaisable, et donc que φ est une tautologie.

Remarque 3.63. L'exemple précédent montre

1. qu'une même clause peut-être utilisée plusieurs fois : la clause $\neg q$ est utilisée deux fois ;
2. la règle de factorisation est nécessaire, sans elle toutes les résolvantes ont deux littéraux et on ne peut donc jamais dériver la clause vide.
3. l'ordre des littéraux dans une clause n'est pas important quand on applique la règle de résolution. Nous avons en fait inféré comme suit :

$$\dfrac{p \vee \neg q \qquad \neg q \vee \neg p}{\neg q \vee \neg q}\ \text{résolution}$$

De façon semblable, l'ordre de littéraux n'est pas important pour appliquer la règle de factorisation.

3.6.2.1 Preuve du Théorème 3.61

Notons d'abord qu'il n'est nécessaire de le démontrer que pour les ensembles de clauses puisque si Γ est équivalent à un ensemble de clauses $\mathcal{C}$ alors Γ est insatisfaisable ssi $\mathcal{C}$ est insatisfaisable.

Notation 3.64. Etant donné un ensemble de clause $\mathcal{C}$ et une clause C, nous écrirons $\mathcal{C} \vdash_r C$ si et seulement si $C \in$ *Resolution*$(\mathcal{C})$, c'est-à-dire si il existe une preuve de C utilisant les clauses de $\mathcal{C}$ comme axiomes.

Commençons par prouver la correction du système. Nous utiliserons le résultat intermédiaire suivant :

Lemme 3.65. *Soient* C_1 *et* C_2 *deux clauses, et* p *une variable propositionnelle. Alors*

$$\{C_1 \vee p, C_2 \vee \neg p\} \models C_1 \vee C_2.$$

Démonstration. Il faut démontrer que pour tout valuation v, si $v(C_1 \vee p) = 1$ et $v(C_2 \vee \neg p) = 1$, alors $v(C_1 \vee C_2) = 1$. Soit v une valuation telle que $v(C_1 \vee p) = 1$ et $v(C_2 \vee \neg p) = 1$:
— si $v(p) = 1$, alors $v(C_2) = 1$ et donc $v(C_1 \vee C_2) = 1$,
— si $v(p) = 0$, alors $v(C_1) = 1$ et donc $v(C_1 \vee C_2) = 1$.

□

Proposition 3.66. *La résolution est correcte : si $\mathcal{C} \vdash_r \bot$, alors $\mathcal{C}$ est insatisfaisable.*

Démonstration. On montre par induction sur la structure des éléments de $\mathit{Resolution}(\mathcal{C})$ que pour toute clause $C \in \mathit{Resolution}(\mathcal{C})$, on a $\mathcal{C} \models C$. Le cas particulier $C = \bot$ démontrera alors directement la proposition.
— (Base) si $C \in \mathcal{C}$, on a évidemment $\mathcal{C} \models C$.
— (Induction) sinon on a deux cas possibles :
— (Résolution) supposons qu'il existe deux clauses $C_1 \vee p$ et $C_2 \vee \neg p$ dans $\mathit{Resolution}(\mathcal{C})$ telles que $C = C_1 \vee C_2$ et (hypothèse d'induction) $\mathcal{C} \models C_1 \vee p$ et $\mathcal{C} \models C_2 \vee \neg p$.
D'après le Lemme 3.65, $\{C_1 \vee p, C_2 \vee \neg p\} \models C$, et on a alors :
$\mathsf{mod}(\mathcal{C}) \subseteq \mathsf{mod}(\{C_1 \vee p, C_2 \vee \neg p\}) \subseteq \mathsf{mod}(C)$, et donc $\mathcal{C} \models C$.
— (Factorisation) Supposons qu'il existe une clause $C_1 \vee \ell \vee \ell \in \mathit{Resolution}(\mathcal{C})$ telle que $C = C_1 \vee \ell$ et (hypothèse d'induction) $\mathcal{C} \models C_1 \vee \ell \vee \ell$. Puisque clairement $C_1 \vee \ell \vee \ell \equiv C$, on a directement $\mathcal{C} \models C$.

□

Proposition 3.67. *La résolution est complète : si $\mathcal{C}$ est insatisfaisable, alors $\mathcal{C} \vdash_r \bot$.*

Démonstration. Par le théorème de compacité, nous pouvons supposer que $\mathcal{C}$ est un ensemble fini et que donc les symboles propositionnels apparaissant dans ses clauses sont en nombre fini.

Nous procédons par une preuve par contraposition en prouvant que si $\mathit{Resolution}(\mathcal{C})$ ne contient pas $\bot$, alors $\mathcal{C}$ est consistant (le principe de preuve par contraposition est issu de la règle de contraposition : $\varphi \Rightarrow \psi \equiv \neg\psi \Rightarrow \neg\varphi$).

Pour cela, on montre par récurrence sur le nombre de symboles propositionnels que si $\mathit{Resolution}(\mathcal{C})$ ne contient pas $\bot$, alors $\mathit{Resolution}(\mathcal{C})$ est satisfaisable.

La proposition découlera alors immédiatement de ce résultat car $\mathcal{C} \subseteq \mathit{Resolution}(\mathcal{C})$ et donc $\mathsf{mod}(\mathit{Resolution}(\mathcal{C})) \subseteq \mathsf{mod}(\mathcal{C})$. En particulier, si $\mathsf{mod}(\mathit{Resolution}(\mathcal{C})) \neq \varnothing$ alors $\mathsf{mod}(\mathcal{C}) \neq \varnothing$.

Dans ce qui suit, on note pour simplifier $\mathcal{S} = \mathit{Resolution}(\mathcal{C})$.

Cas de base : $n = 1$. Soit donc p le seul symbole propositionnel ayant une occurrence dans $\mathcal{S}$. Clairement, $\mathcal{S}$ ne peut contenir à la fois les deux clauses unitaires p et $\neg p$, sinon il contiendrait aussi donc la clause vide. Donc $\mathcal{S}$ admet un modèle (est satisfaisable).

Pas d'induction. Supposons que la propriété est vraie pour n et que $\mathcal{S}$ contient $n+1$ symboles propositionnels. Soit p un symbole propositionnel apparaissant dans une clause de $\mathcal{S}$. On définit :

$$\mathcal{S}_0 := \{\, C \in \mathcal{S} \mid \neg p \notin C \,\}, \qquad \mathcal{S}'_0 := \mathcal{S}_0[p \leftarrow \bot],$$
$$\mathcal{S}_1 := \{\, C \in \mathcal{S} \mid p \notin C \,\}, \qquad \mathcal{S}'_1 := \mathcal{S}_1[p \leftarrow \top].$$

Remarquez que :
— si $\bot \in \mathcal{S}'_0$, alors $p \in \mathcal{S}_0$;
— si $\bot \in \mathcal{S}'_1$ alors $\neg p \in \mathcal{S}_1$.
On peut alors en conclure que $\bot$ ne peut appartenir à $\mathcal{S}'_0$ et $\mathcal{S}'_1$, puisque dans ce cas on aurait $p, \neg p \in \mathcal{S}$ et donc $\bot \in \mathcal{S}$. Notez de plus que les ensembles $\mathcal{S}'_i$ sont saturées, c'est à dire que $\mathcal{S}'_i = \mathit{Resolution}(\mathcal{S}'_i)$, puisque la suppression d'une variable ne crée pas de nouvelles possibilités d'appliquer les règles.
Par conséquent, un parmi $\mathcal{S}'_i$, $i = 0, 1$, est saturé et ne contient pas la clause $\bot$; sans perte de généralité, nous pouvons supposer qu'il s'agit de $\mathcal{S}'_1$. Par hypothèse d'induction, $\mathcal{S}'_1$ est satisfaisable. Soit v un modèle de $\mathcal{S}'_1$ et u la valuation telle que $u(p) = 1$ et $u(q) = v(q)$ pour $p \neq q$; alors u est un modèle de $\mathcal{S}$. En effet on a alors :
— pour toute clause $C \in \mathcal{S}_1$: $v(C) = u(C[p \mapsto \top]) = 1$
— pour toute clause $C \notin \mathcal{S}_0$, par définition C contient p et donc $v(C) = 1$.

□

3.6.2.2 Extension de la méthode aux conséquences logiques

On rappelle qu'une formule close φ est conséquence logique d'un ensemble de formules Γ (noté $\Gamma \models \varphi$) si tout modèle de Γ est modèle de φ. On étend la méthode le la résolution aux conséquences logiques en notant $\Gamma \vdash_r \varphi$ si $\Gamma \cup \{\neg\varphi\} \vdash_r \bot$. En particulier, on note donc $\vdash_r \varphi$ si $\neg\varphi \vdash_r \bot$

Théorème 3.68. *Pour tout ensemble* Γ*, pour toute formule* φ*,*

$$\Gamma \vdash_r \varphi \textit{ ssi } \Gamma \models \varphi.$$

Démonstration.

$\Gamma \vdash_r \varphi$ ssi $\Gamma \cup \{\neg\varphi\} \vdash_r \bot$
ssi $\Gamma \cup \{\neg\varphi\}$ est insatisfaisable (Théorème **??**)
ssi $\Gamma \models \varphi$ (Proposition 3.52). □

En corollaire immédiat, on a

Corollaire 3.69. *Pour toute formule* φ*,*

$$\models \varphi \textit{ ssi } \vdash_r \varphi.$$

3.6.2.3 L'algorithme de résolution

La méthode de la résolution ne peut pas être considérée comme un algorithme, car rien n'assure sa terminaison, comme le montre l'exemple suivant :

Exemple 3.70. Si on considère l'ensemble de clauses $\mathcal{C} = \{p \vee \neg q, q \vee \neg p\}$. La méthode engendre $\{p \vee \neg p, q \vee \neg q, p \vee \neg q, q \vee \neg p\}$ (et rien de plus, mais il faut un raisonnement pour le prouver !) qui a un modèle $v(p) = 1$, $v(q) = 1$. Donc la formule $\neg((p \vee \neg q) \wedge (q \vee \neg p))$ n'est pas une tautologie.

On voit sur l'exemple suivant que la méthode de la résolution peut engendrer un ensemble infini de clauses et donc ne pas terminer.

Exemple 3.71. $\mathcal{C} = \{p \vee \neg p \vee \neg q\}$. On commence par appliquer la règle de résolution, et on obtient la clause $p \vee \neg p \vee \neg q \vee \neg q$, si lui applique la règle de factorisation, on retombe sur la clause initiale $p \vee \neg p \vee \neg q$, ce qui est inutile. On ne peut donc que réappliquer la résolution aux clauses $p \vee \neg p \vee \neg q$ et $p \vee \neg p \vee \neg q \vee \neg q$.

$$\dfrac{\dfrac{\dfrac{p \vee \neg p \vee \neg q \qquad p \vee \neg p \vee \neg q}{p \vee \neg p \vee \neg q \vee \neg q}\ \text{résolution} \qquad p \vee \neg p \vee \neg q}{p \vee \neg p \vee \neg q \vee \neg q \vee \neg q}\ \text{résolution} \qquad p \vee \neg p \vee \neg q}{\vdots}\ \text{résolution}$$

La méthode va générer les clauses de la forme $p \vee \neg p \vee \neg q \vee \neg q \vee \ldots \neg q$ et aucune autre (la encore un raisonnement est à faire). Donc la méthode ne termine pas et engendre un ensemble de clauses infini qui ne contient pas la clause vide.

On peut modifier la méthode pour obtenir un algorithme qui se termine toujours, si on se restreint à considérer les clauses factorisées.

Définition 3.72. Une clause C est *factorisée* si elle ne contient pas deux occurrences du même littéral.

Si C est une clause, nous allons dénoter par $\mathfrak{f}(C)$ la clause factorisée obtenue de C par une suite de règles de factorisation. L'algorithme est donné ci-dessous. Sa terminaison est assurée par le fait que le nombre de clauses factorisées ayant au plus n symboles propositionnels est borné supérieurement par une fonction de n.

```
Algorithme DP60
  input: un ensemble C de clauses factorisees
  output: insatisfaisable si C ⊨ ⊥ ou satisfaisable sinon
  repeter
       choisir une variable x apparaissant dans la formule
       Ajouter a C toutes les resolvantes possibles entre
             toutes les clauses contenant x et
             toutes les clauses contenant sa negation
       Retirer de  C toutes les clauses contenant x ou sa negation
       si ⊥ ∈ C, alors retourner insatisfaisable
       si C est vide retourner satisfaisable
  fin
```

3.6.3 Le calcul des séquents

Dans ce système, on démontre des énoncés de la forme $\Gamma \vdash \Delta$ appelés séquents, où Γ et Δ sont des **multi-ensembles** [1] finis de formules.

Définition 3.73. Un séquent est une paire d'ensembles de formules $\Gamma \vdash \Delta$. On représente un séquent par une liste de symboles de la forme $\varphi_1, \ldots, \varphi_n \vdash \psi_1, \ldots, \psi_m$, où φ_i et ψ_i sont des formules propositionnelles.

Définition 3.74. L'ensemble des **séquents valides** est défini inductivement par les règles suivantes :

— Axiomes :

$$\frac{}{\Gamma, \varphi \vdash \varphi, \Delta}$$

— Règles logiques : il y a deux règles par connecteur, une à gauche, une à droite.

Connecteur	Gauche	Droite
$\bot$	$\dfrac{}{\Gamma, \bot \vdash \Delta}\, G_\bot$	$\dfrac{\Gamma \vdash \Delta}{\Gamma \vdash \bot, \Delta}\, D_\bot$
$\top$	$\dfrac{\Gamma \vdash \Delta}{\Gamma, \top \vdash \Delta}\, G_\top$	$\dfrac{}{\Gamma \vdash \top, \Delta}\, D_\top$
$\neg$	$\dfrac{\Gamma \vdash \varphi, \Delta}{\Gamma, \neg\varphi \vdash \Delta}\ (G_\neg)$	$\dfrac{\Gamma, \varphi \vdash \Delta}{\Gamma \vdash \neg\varphi, \Delta}\ (D_\neg)$
$\wedge$	$\dfrac{\Gamma, \varphi, \psi \vdash \Delta}{\Gamma, \varphi \wedge \psi \vdash \Delta}\ (G_\wedge)$	$\dfrac{\Gamma \vdash \varphi, \Delta \quad \Gamma \vdash \psi, \Delta}{\Gamma \vdash \varphi \wedge \psi, \Delta}\ (D_\wedge)$
$\vee$	$\dfrac{\Gamma, \varphi \vdash \Delta \quad \Gamma, \psi \vdash \Delta}{\Gamma, \varphi \vee \psi \vdash \Delta}\ (G_\vee)$	$\dfrac{\Gamma \vdash \varphi, \psi, \Delta}{\Gamma \vdash \varphi \vee \psi, \Delta}\ (D_\vee)$
$\Rightarrow$	$\dfrac{\Gamma \vdash \varphi, \Delta \quad \Gamma, \psi \vdash \Delta}{\Gamma, \varphi \Rightarrow \psi \vdash \Delta}\ (G_\Rightarrow)$	$\dfrac{\Gamma, \varphi \vdash \psi, \Delta}{\Gamma \vdash \varphi \Rightarrow \psi, \Delta}\ (D_\Rightarrow)$

3.6.3.1 Exemples de séquents valides

Exemple 3.75. Une preuve du séquent $\vdash p \vee \neg p$ est :

$$\dfrac{\dfrac{\dfrac{}{p \vdash p}\,(A_x)}{\vdash p, \neg p}\,(D_\neg)}{\vdash p \vee \neg p}\,(D_\vee)$$

1. un multi-ensemble est un ensemble qui peut contenir plusieurs fois le même élément

Exemple 3.76. Une preuve du séquent $\vdash p \Rightarrow (q \Rightarrow (p \wedge q))$ est :

$$\dfrac{\dfrac{\dfrac{\dfrac{}{p,q \vdash p}(A_x) \quad \dfrac{}{p,q \vdash q}(A_x)}{p,q \vdash p \wedge q}(D_\wedge)}{p \vdash q \Rightarrow (p \wedge q)}(D_\Rightarrow)}{\vdash p \Rightarrow (q \Rightarrow (p \wedge q))}(D_\Rightarrow)$$

Exemple 3.77. Preuve du séquent $(p \wedge q) \vee r \vdash (p \vee r) \wedge (q \vee r)$:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{r \vdash p,r}(A_x) \quad \dfrac{\dfrac{}{p,q \vdash p,r}(A_x)}{p \wedge q \vdash p,r}(G_\wedge)}{(p \wedge q) \vee r \vdash p,r}(G_\vee)}{(p \wedge q) \vee r \vdash p \vee r}(D_\vee) \quad \dfrac{\dfrac{\dfrac{}{r \vdash q,r}(A_x) \quad \dfrac{\dfrac{}{p,q \vdash q,r}(A_x)}{p \wedge q \vdash q,r}(G_\wedge)}{(p \wedge q) \vee r \vdash q,r}(G_\vee)}{(p \wedge q) \vee r \vdash q \vee r}(D_\vee)}{(p \wedge q) \vee r \vdash (p \vee r) \wedge (q \vee r)}(D_\wedge)$$

On peut représenter les arbres de preuves de façon linéaire et de haut en bas. C'est l'indentation qui donne les branchements de l'arbre de preuve.

Notre exemple se représente de la façon suivante :

$(p \wedge q) \vee r \vdash (p \vee r) \wedge (q \vee r)$ $(D_\wedge)$
  $(p \wedge q) \vee r \vdash p \vee r$ $(D_\vee)$
    $(p \wedge q) \vee r \vdash p,r$ $(G_\vee)$
      $p \wedge q \vdash p,r$ $(G_\wedge)$
        $p,q \vdash p,r$ (A_x)
      $r \vdash p,r$ (A_x)

$(p \wedge q) \vee r \vdash q \vee r$ $(D_\vee)$
  $(p \wedge q) \vee r \vdash q,r$ $(G_\vee)$
    $p \wedge q \vdash q,r$ $(G_\wedge)$
      $p,q \vdash q,r$ (A_x)
    $r \vdash q,r$ (A_x)

La preuve du lemme suivant est laissée en exercice.

Lemme 3.78. *Les règles du calcul des séquents sont inversibles :*

— *pour toute règle* $\dfrac{\Gamma_1 \vdash \Delta_1}{\Gamma \vdash \Delta}$ *:* $\Gamma_1 \models \Delta_1$ *ssi* $\Gamma \models \Delta$.

— *pour toute règle* $\dfrac{\Gamma_1 \vdash \Delta_1 \qquad \Gamma_2 \vdash \Delta_i}{\Gamma \vdash \Delta}$ *:* $\Gamma_1 \models \Delta_1$ *et* $\Gamma_2 \models \Delta_2$ *ssi* $\Gamma \models \Delta$.

Théorème 3.79 (Théorème de correction et completude)**.** *Il existe une preuve de* $\Gamma \vdash \Delta$ *dans le calcul des séquents ssi* $\Gamma \models \Delta$.

Démonstration. Les axiomes étant corrects, le Lemma 3.78 implique directement que le système des correct. Pour vérifier que le système est complet, on procède comme suit.

Etant donné un séquent s, on peut construire un arbre en appliquant sur chacune des formules, la règle logique correspondant à son connecteur principal. On itére ce processus jusqu'à avoir des sous buts ne contenant que des formules atomiques. Si chacun d'eux se termine par la règle (Ax), alors on a une preuve.

Si ce n'est pas le cas alors chaque séquent de feuille qui n'est pas un axiome donne une valuation qui falsifie l'énoncé de départ : en effet, si il n'est pas un axiome, alors il est de la forme $p_1, \ldots, p_n \vdash q_1, \ldots q_m$ où $\{p_1, \ldots, p_n\} \cap \{q_1, \ldots q_m\} = \varnothing$. On choisit la valuation v telle que $v(p_i) = 1$ et $v(q_j) = 0$ pour tous i, j. Le séquent à donc la valeur 0. Les règles étant inversibles, tous les séquents apparaissant dessous auront également la valeur 0. Ainsi v donne la valeur 0 au séquent initial. □

Exemple 3.80. Réfutation d'un séquent :

$$\dfrac{\dfrac{\dfrac{q \vdash (p \Rightarrow q) \qquad \overline{q \vdash p}\ \text{echec !!}}{q \vdash (p \Rightarrow q) \wedge p}\ (D_\wedge)}{\vdash q \Rightarrow ((p \Rightarrow q) \wedge p)}\ (D_\Rightarrow)}$$

la valuation $v(p) = 0$ et $v(q) = 1$ satisfait $v(q \Rightarrow ((p \Rightarrow q) \wedge p)) = 0$.

Proposition 3.81 (Propriété de la sous formule)**.** *Etant donné une preuve d'un séquent s. Toutes les formules figurant dans la preuve sont des sous-formules de s.*

Démonstration. Se vérifie facilement par induction sur la longueur de la preuve. Il suffit de remarquer que les règles simplifient les formules, sans en ajouter de nouvelles. □

Cette propriété est importante : elle signifie qu'**un théorème contient tout ce qu'il faut pour sa démonstration**.

3.6.3.2 Extension du système

On dispose donc d'un système complet et correct. Pour simplifier les preuve, il est possible d'ajouter de nouvelles règles. Nous en donnons ici quelques unes. Notez que l'ajout de nouvelles règles n'affecte pas la complétude du système. Par contre, il peut affecter sa correction. Il faudra donc s'assurer que chaque nouvelle règle est correcte.

Règles structurelles : il s'agit de deux règles d'affaiblissement, et de deux règles de contraction : une à gauche, et une à droite.

$$\dfrac{\Gamma, \varphi, \varphi \vdash \Delta}{\Gamma, \varphi, \vdash \Delta}\ (G_{Contr}) \qquad \dfrac{\Gamma \vdash \Delta, \varphi, \varphi}{\Gamma \vdash \Delta, \varphi}\ (D_{Contr})$$

$$\dfrac{\Gamma \vdash \Delta}{\Gamma, \varphi \vdash \Delta}\ (G_{Aff}) \qquad \dfrac{\Gamma \vdash \Delta}{\Gamma \vdash \Delta, \varphi}\ (D_{Aff})$$

Règle de coupure : c'est l'équivalent à l'utilisation d'un théorème annexe.

$$\dfrac{\Gamma_1 \vdash \varphi, \Delta_1 \quad \Gamma_2, \varphi \vdash \Delta_2}{\Gamma_1, \Gamma_2 \vdash \Delta_1, \Delta_2}\ (C)$$

La règle de coupure dit que l'on peut toujours ajouter en hypothèse une formule φ du moment que l'on sait en produire une preuve.

Proposition 3.82. *Le calcul des séquents augmenté des règles structurelles et de la règle de coupure est complet et correct.*

Démonstration. Le système initial était complet, donc le système enrichi l'est aussi. Il s'agit donc de montrer que ce nouveau système est correct. Pour cela, il suffit de vérifier pour chaque règle que si les prémisses sont vraies, la conclusion l'est aussi. Ces vérifications sont laissées en exercice. □

3.6.4 Correction, complétude et décidabilité

On dit d'une logique qu'elle est correcte et complète lorsqu'il existe un système de preuve correct et complet pour cette logique.

On dit qu'une logique est décidable si il existe un algorithme capable de décider, pour toute formule de cette logique, si la formule est valide.

Ces deux notions sont très proches, il est important de saisir la nuance entre les deux. Une logique correcte et complète n'est pas forcément décidable. En effet, il existe alors une preuve de tout théorème, mais il n'y a pas forcément de procédure effective permettant de construire une preuve.