

Université de Nancy I  
DEA d'informatique  
Tronc Commun: Cours de Logique

Denis Lugiez & Michael Rusinowitch



# Contents

<b>1</b>	<b>Le calcul propositionnel</b>	<b>7</b>
1.1	Le langage du Calcul propositionnel . . . . .	7
1.1.1	Définitions et Propriétés . . . . .	7
1.1.2	Formalisation de problème . . . . .	8
1.2	Sémantique . . . . .	9
1.3	Manipulations algébriques de formules . . . . .	13
1.3.1	Lemme de Koenig . . . . .	13
1.3.2	Equivalence de formules . . . . .	13
1.3.3	Formes particulières de formules . . . . .	14
1.4	Le théorème de compacité . . . . .	16
1.5	Système déductif: méthode des tableaux sémantiques . . . . .	17
1.6	Correction et complétude . . . . .	20
1.6.1	Preuve de correction . . . . .	21
1.6.2	Le théorème de complétude . . . . .	21
1.6.3	Traitement de la conséquence logique . . . . .	22
1.7	Autres systèmes déductifs . . . . .	23
1.7.1	La Résolution . . . . .	23
1.7.2	Le système de Hilbert . . . . .	24
<b>2</b>	<b>Calcul des Prédicats du premier ordre</b>	<b>27</b>
2.1	Le langage du calcul des prédicats . . . . .	27
2.1.1	Syntaxe . . . . .	27
2.1.2	Termes et substitutions . . . . .	28
2.1.3	Formules . . . . .	29
2.2	Sémantique . . . . .	30
2.3	Théorèmes caractéristiques . . . . .	33
2.4	Forme normale prenexe. Skolémisation . . . . .	34
2.4.1	Forme prénexe . . . . .	34

2.4.2	Skolémisation . . . . .	36
2.5	Méthodes des tableaux pour le calcul des prédicats . . . . .	38
2.5.1	Présentation . . . . .	38
2.5.2	Correction . . . . .	40
2.5.3	Complétude . . . . .	41
2.6	Amélioration de la méthode des tableaux . . . . .	42
2.6.1	Traitement de la conséquence logique . . . . .	43
2.6.2	Utilisation de l'unification . . . . .	43
<b>3</b>	<b>Logique modale</b> . . . . .	<b>45</b>
3.1	Introduction . . . . .	45
3.2	Logique du possible . . . . .	46
3.3	Logique de la connaissance ou de la croyance . . . . .	47
3.3.1	Logiques $\mathcal{S}5$ et $\mathcal{S}4$ . . . . .	49
3.3.2	Un exemple . . . . .	52
3.4	Sémantique de la logique modale: modèles de Kripke . . . . .	53
3.5	Systèmes de preuve pour les logiques modales . . . . .	58
3.5.1	Systèmes de Hilbert . . . . .	58
3.5.2	Tableaux sémantiques . . . . .	59
3.6	Logique temporelle linéaire . . . . .	61
3.6.1	Langage . . . . .	61
3.6.2	Sémantique . . . . .	62
3.6.3	Système formel . . . . .	62
3.6.4	Applications . . . . .	63
3.6.5	Expressivité . . . . .	65
3.7	Logique dynamique . . . . .	66
3.7.1	Langage . . . . .	66
3.7.2	Sémantique . . . . .	67
3.7.3	Système formel . . . . .	67
<b>4</b>	<b>Décidabilité</b> . . . . .	<b>69</b>
4.1	Les grandes techniques de décidabilité . . . . .	69
4.2	Une méthode de décision à base de modèles pour la classe monadique . . . . .	70
4.3	Méthode par élimination de quantificateurs . . . . .	71
4.3.1	Exemple 1: théorie de l'égalité sur un ensemble infini . . . . .	72
4.3.2	Exemple 2: théorie élémentaire du successeur . . . . .	73

4.4	Une méthode à base d'automates pour l'arithmétique de Presburger . . . . .	74
4.4.1	Codage de l'addition $x = y + z$ . . . . .	75
4.4.2	Résolution de formules . . . . .	76
4.4.3	Calcul direct de l'automate reconnaissant $\forall x \varphi$ . . . . .	77
4.4.4	Application aux formules closes . . . . .	79
<b>5</b>	<b>Indécidabilité</b> . . . . .	<b>81</b>
5.1	Calculabilité . . . . .	81
5.1.1	Machines de Turing . . . . .	81
5.1.2	Code d'une machine de Turing . . . . .	83
5.1.3	Fonctions calculables . . . . .	84
5.1.4	La thèse de Church . . . . .	85
5.2	Problèmes indécidables . . . . .	85
5.2.1	Existence de fonction non-calculables . . . . .	85
5.2.2	Le problème de l'arrêt . . . . .	85
5.2.3	Problèmes indécidables . . . . .	87



# Chapter 1

## Le calcul propositionnel

### 1.1 Le langage du Calcul propositionnel

#### 1.1.1 Définitions et Propriétés

Le calcul propositionnel (abrégeé en CP0) permet d'exprimer des propriétés élémentaires (ou atomiques) et de les combiner entre elles à l'aide des connecteurs logiques. On peut ainsi modéliser une partie importante du raisonnement. Plus formellement, le langage se définit comme suit.

**Definition 1** *Le langage  $\mathcal{L}$  du CP0 est composé:*

- *des symboles (ou variables) propositionnels (notés usuellement en lettres minuscules  $p, q, r$ )*
- *connecteurs logiques  $\wedge, \vee, \Rightarrow, \Leftrightarrow, \neg$*
- *de symboles auxiliaires: “)” “(“ “,” “ “*

Dans la suite on supposera que l'ensemble des symboles propositionnels est dénombrable, et  $\circ$  désignera l'un quelconque des connecteurs binaires ( $\wedge, \vee, \Rightarrow, \Leftrightarrow$ ). Un symbole propositionnel s'appelle aussi un *atome*. On définit maintenant les formules du CP0:

**Definition 2** *FORM l'ensemble des formules du CP0 est le plus petit ensemble tel que*

- *tout atome est dans FORM*

- si  $\varphi \in FORM$  alors  $\neg\varphi \in FORM$
- si  $\varphi, \psi \in FORM$  alors  $\varphi \circ \psi \in FORM$

Exemple:  $p \vee q \in FORM, q \Rightarrow q \in FORM$  mais  $\neg \vee q \notin FORM$

Principe d'induction structurelle sur  $FORM$ :

**Théorème 1** Une propriété  $P$  est vraie pour toute formule de  $FORM$  ssi:

- $P$  est vraie sur les symboles propositionnels.
- si  $\varphi$  vérifie  $P$  alors  $\neg\varphi$  vérifie  $P$ .
- si  $\varphi$  et  $\psi$  vérifient  $P$  alors  $\varphi \circ \psi$  vérifie  $P$ .

Par conséquent, pour définir une fonction  $f$  sur  $FORM$ , il suffit de la définir sur les symboles propositionnels, et de définir  $f(\neg\varphi)$  à partir de  $f(\varphi)$  et  $f(\varphi \circ \psi)$  à partir de  $f(\varphi)$  et  $f(\psi)$ .

Une notion utile est celle de sous-formule:

**Definition 3** On définit la propriété  $\varphi$  est une sous-formule de  $\psi$  par:

- une sous-formule d'un symbole propositionnel est le symbole propositionnel.
- une sous-formule de  $\neg\varphi$  est soit une sous-formule de  $\varphi$  soit  $\neg\varphi$
- une sous-formule de  $\varphi \circ \psi$  est soit une sous-formule de  $\varphi$ , soit une sous-formule de  $\psi$  soit  $\varphi \circ \psi$

Une sous-formule stricte de  $\varphi$  est une sous-formule de  $\varphi$  distincte de  $\varphi$ . Par exemple les sous-formules strictes de  $p \vee (q \Rightarrow r)$  sont  $p, q, r, q \Rightarrow r$ .

### 1.1.2 Formalisation de problème

Nous montrons comment ce langage peut s'utiliser pour passer d'un exposé informel à un exposé formel qui est une formule du CP0.

On donne le règlement d'un club:

- Les membres de la direction financière sont choisis parmi ceux de la direction générale

- Nul ne peut être à la fois membre de la direction générale et de la direction de la bibliothèque, s'il n'est membre de la direction financière
- Aucun membre de la direction de la bibliothèque ne peut être membre de la direction financière

peut se formaliser en, si  $F$  signifie “être membre de la direction financière”,  $G$  “être membre de la direction générale”,  $B$  “être membre de la direction de la bibliothèque”,  $((F \Rightarrow G) \wedge (((\neg(G \wedge B)) \vee F) \wedge (\neg(B \wedge F))))$  ce qui peut se simplifier en la formule équivalente  $(G \vee \neg F) \wedge (\neg G \vee \neg B)$  ce qui permet d’élaborer le règlement simplifié qui fusionne les deux derniers points en: “aucun membre de la direction générale ne peut être membre de la direction de la bibliothèque”

## 1.2 Sémantique

La sémantique classique utilisée comporte 2 valeurs de vérité 0 et 1 (0 pour faux et 1 pour vrai)<sup>1</sup>, et une valuation servira à associer une valeur de vérité à une formule une fois fixées les valeurs de vérité des atomes et en respectant les définitions des tables de vérité des connecteurs.

$p$	$q$	$(p \wedge q)$
0	0	0
0	1	0
1	0	0
1	1	1

$p$	$q$	$(p \vee q)$
0	0	0
0	1	1
1	0	1
1	1	1

$p$	$(\neg p)$
0	1
1	0

$p$	$q$	$(p \Rightarrow q)$
0	0	1
0	1	1
1	0	0
1	1	1

$p$	$q$	$(p \Leftrightarrow q)$
0	0	1
0	1	0
1	0	0
1	1	1

---

<sup>1</sup>certaines logiques ont 3 valeurs de vérité 0 pour faux, 1 pour vrai et 1/2 pour “on ne sait pas”, d’autres ont un nombre fini de valeurs de vérité, d’autres encore un ensemble infini de valeurs de vérité.

Il n'est pas inutile de remarquer que  $(p \Rightarrow q)$  est vrai, quand  $p$  est faux. Ainsi l'implication "New-York est la capitale de l'URSS"  $\Rightarrow$  "Moscou est la capitale des USA" est vraie, ce qui peut paraître choquant. Néanmoins, notre définition est la seule réaliste si on veut garder le sens de si  $p$  alors  $q$ . Ne la considérer vraie que si  $p$  et  $q$  sont vrais revient à identifier  $\Rightarrow$  et  $\wedge$ . La considérer fausse quand  $p$  et  $q$  le sont fait perdre l'équivalence (qui est la base du raisonnement par contraposition)  $(p \Rightarrow q)$  équivaut à  $(\neg q \Rightarrow \neg p)$ . Affecter une valeur de vérité à une formule se fait à l'aide de valuations.

**Definition 4** Une assignation  $A$  est une application de l'ensembles des variables propositionnelles dans  $\{0, 1\}$ .

Une valuation est une application  $v$  de l'ensemble des formules dans  $\{0, 1\}$  telle que:

- $v(\neg\varphi) = 1 - v(\varphi)$
- $v(\varphi \wedge \psi) = \text{Min}(v(\varphi), v(\psi))$
- $v(\varphi \vee \psi) = \text{Max}(v(\varphi), v(\psi))$
- $v(\varphi \Rightarrow \psi) = 0$  ssi  $v(\varphi) = 1$  et  $v(\psi) = 0$
- $v(\varphi \Leftrightarrow \psi) = 1$  ssi  $v(\varphi) = v(\psi)$

**Proposition 1** Etant donné une assignation  $A$ , il existe une unique valuation  $v$  qui la prolonge (c.a.d.  $v(p) = v(A)$  pour tout variable propositionnelle  $p$ ).

**Preuve** Par induction structurelle sur les formules. □

La notion centrale est celle de tautologie ( formule toujours vraie)

**Definition 5** Une formule  $\varphi$  est une tautologie (noté  $\models \varphi$ ) ssi pour toute valuation  $v$  on a  $v(\varphi) = 1$ .

Exemple:  $p \vee \neg p$  est une tautologie mais pas  $p \vee q$ . D'autres notions sont utiles:

**Definition 6** (satisfiabilité, insatisfiabilité, conséquence logique)

- Une formule  $\varphi$  est satisfiable ssi il existe une valuation  $v$  avec  $v(\varphi) = 1$  (on dit que  $v$  est un modèle de  $\varphi$ ). Un ensemble  $E$  de formules est satisfiable ssi il existe une valuation  $v$  telle que  $v(\varphi) = 1$  pour toutes les formules  $\varphi$  de  $E$ . Une formule (ou un ensemble de) est insatisfiable s'il n'existe pas de valuation  $v$  telle que  $v(\varphi) = 1$ .
- une valuation  $v$  telle que  $v(\varphi) = 0$  falsifie la formule  $\varphi$ . Si  $v(\varphi) = 1$  elle valide  $\varphi$ .
- Une formule  $\varphi$  est une conséquence logique d'un ensemble de formules  $\Gamma$  (noté  $\Gamma \models \varphi$ ) ssi pour toute valuation  $v$  qui valide toutes les formules de  $\Gamma$  alors  $v(\varphi) = 1$ .

**Théorème 2** Il existe un algorithme qui permet de savoir si la propriété  $\varphi$  est une tautologie est vraie (méthode des tables de vérité).

Méthode des tables de vérité: si  $p_1, \dots, p_n$  sont les  $n$  atomes intervenant dans  $\varphi$ , on dresse la table à  $2^n$  entrées qui calcule la valeur de vérité de  $\varphi$  par calcul des valeurs de vérité des sous-formules de  $\varphi$  (une fois données les  $v(p_i)$ ). On peut ainsi vérifier si la formule est satisfiable, valide ou insatisfiable.

Exemple: soit  $\varphi$  la formule  $((\neg q) \Rightarrow (\neg p)) \Leftrightarrow (p \Rightarrow q)$ , sa table de vérité est calculée comme suit:

$p$	$q$	$(\neg p)$	$(\neg q)$	$((\neg q) \Rightarrow (\neg p))$	$(p \Rightarrow q)$	$\varphi$
0	0	1	1	1	1	1
0	1	1	0	1	1	1
1	0	0	1	0	0	1
1	1	0	0	1	1	1

On vérifie donc que pour toute valuation,  $\varphi$  a pour valeur 1, ce qui prouve que  $\varphi$  est une tautologie. S'il n'y avait eu que des 0, alors  $\varphi$  aurait été insatisfiable, et s'il y avait eu des 0 et des 1 alors  $\varphi$  aurait été satisfiable. Si cette méthode est effective, elle est impraticable, car si la formule considérée comporte  $n$  symboles propositionnels, il y a  $2^n$  cas à examiner, donc le coût est exponentiel. En particulier, cette méthode qui serait la plus simple dans le cadre des bases de données n'est absolument pas utilisable, pour autre chose que des exemples d'école. La complexité exacte du problème "est-ce qu'une formule est une tautologie" est un problème ouvert. Toutes les méthodes connues sont exponentielles, et on conjecture que le problème n'est pas polynômial (cf la conjecture  $P \neq NP$ ).

A titre d'exercice, on peut montrer ainsi que  $q$  est une conséquence logique de  $\{p, (p \Rightarrow q)\}$ ,  $\{p, (q \Rightarrow p)\}$  est satisfiable, et que  $\{p, (\neg p)\}$  est insatisfiable.

**Exercice :** Pierre, Paul et Jacques sont prévenus de fraude fiscale et prêtent serment de la façon suivante:

- Pierre: Paul est coupable et Jacques est innocent
- Paul: Si Pierre est coupable alors Jacques aussi
- Jacques: Je suis innocent mais au moins un des deux autres est coupable.

Exprimer le témoignage de chacun des suspects sous forme propositionnelle. Dresser la table de vérité de ces formules. Les témoignages des trois suspects sont-ils compatibles? Le témoignage de l'un des suspects découle de celui d'un autre suspect. Lequel? En supposant que tous sont innocents, qui a fait un faux témoignage?

**Exercice :** On dit qu'un ensemble de connecteurs est adéquat s'il permet d'exprimer toutes les fonctions de vérité (applications de PROP dans  $0,1$ ). Les ensembles suivants sont-ils adéquats:  $\{\neg, \wedge\}, \{\vee, \wedge\}, \{\vee, \neg\}, \{\mid\}$  ou  $\mid$  (opérateur de contradiction de Sheffer) est défini par sa table de vérité:

$p$	1	1	0	0
$q$	1	0	1	0
$p \mid q$	0	0	0	1

**Exercice :** On donne deux formules A et B telles que  $A \models B$  et telle que  $\mathcal{S}$  l'ensemble des atomes communs à A et B est non vide. Montrer qu'il existe C (l'interpolant de A et B) tel que  $A \models C$  et  $C \models B$  et tel que l'ensemble des symboles propositionnels de C soit  $\mathcal{S}$

**Exercice :** Une clause de Horn est une disjonction de littéraux dont un seul est positif. Une valuation  $v$  est générique pour un ensemble de formules  $\Sigma$  si  $v(p) = 1$  ssi  $v'(p) = 1$  pour tout autre valuation  $v'$  qui valide  $\Sigma$ . On dit qu'un ensemble de formules  $\Sigma$  admet une valuation générique ssi pour tout symbole propositionnel  $p$ , il existe une valuation générique pour  $\Sigma \cup \{p\}$ .

Montrer que  $\Sigma$  admet une valuation générique ssi  $\Sigma$  est un ensemble de clauses de Horn.

## 1.3 Manipulations algébriques de formules

### 1.3.1 Lemme de Koenig

Nous détaillons le lemme de König qui sera utilisé plusieurs fois dans la suite.

**Definition 7** *Un arbre est branchement fini si chaque noeud a un nombre fini de fils.*

**Lemme 1** *Un arbre branchement fini dont les branches sont finies est fini.*

**Preuve** Supposons que  $T$  soit un arbre branchement fini dont les branches sont finies. Si  $T$  est infini on peut construire par récurrence une suite infinie  $(n_i)_i$  de noeuds de la manière suivante:  $n_0$  est la racine. Supposons par hypothèse de récurrence que le sous-arbre de racine  $n_i$  est infini. Alors parmi les fils  $f_1, f_2, \dots, f_{k_i}$  de  $n_i$  l'un d'eux (au moins) est racine d'un sous-arbre infini. Prenons pour  $n_{i+1}$  l'un de ces fils. La suite  $(n_i)_i$  définit une branche infinie, ce qui est contradictoire avec l'hypothèse.  $\square$

Fond sur ce lemme, voici un argument simple pour montrer la terminaison de certains processus ou programmes. Nous présentons cet argument sous forme de jeu.

On a un ensemble fini de balles numrotées par des entiers positifs. Plusieurs balles peuvent porter le même numéro. Chaque joueur remplace tour de rôle une balle par un nombre quelconque de balles de numéros strictement inférieurs puisés dans une réserve infinie. Il s'agit de montrer que ce jeu termine.

**Preuve** Chaque balle initiale est considérée comme la racine d'un arbre. Ses fils sont les balles qui la remplacent à la suite d'un coup. Les arbres obtenus ainsi sont branchement fini et branches finies car les numéros décroissent strictement le long de chaque branche. Par le lemme de König ces arbres sont finis, donc le jeu s'arrête.  $\square$

### 1.3.2 Equivalence de formules

On introduit une relation entre formules qui joue un rôle similaire à l'égalité, ce qui justifiera certaines transformations algébriques sur les formules.

**Definition 8** *On définit  $\varphi \equiv \psi$  ssi  $\varphi \Leftrightarrow \psi$  est une tautologie.*

**Proposition 2**  $\equiv$  est une relation d'équivalence sur *FORM*.

La preuve est laissée au lecteur.

Cette relation d'équivalence est similaire à une égalité, en particulier si on note  $F(p \leftarrow \varphi)$  la formule obtenue en remplaçant dans la formule  $F$  toutes les occurrences de  $p$  par  $\varphi$ , on a le théorème de remplacement:

**Proposition 3**  $\varphi \equiv \psi$  alors  $F(p \leftarrow \varphi) \equiv F(p \leftarrow \psi)$ .

**Preuve** par induction structurelle sur  $F$  □

D'autres propriétés algébriques sont utiles, comme l'associativité et commutativité de  $\wedge$  et  $\vee$ , ainsi que la dualité de ces 2 connecteurs par négation (lois de Morgan).

### 1.3.3 Formes particulières de formules

Les méthodes de preuve automatique demandent à travailler sur des formules particulières afin d'être efficaces. Pour commencer, on définit la *forme normale négative*:  $\varphi$  est en forme normale négative (f.n.n en abrégé) ssi toutes les négations sont devant des atomes.

**Proposition 4** Pour toute formule  $\varphi$  il existe une  $\psi$  en f.n.n. telle que  $\varphi \equiv \psi$ .

**Preuve** Par induction structurelle sur  $\varphi$ . □

**Exemple 1**  $\neg(p \wedge \neg q) \vee \neg r \equiv \neg p \vee q \vee \neg r$

La forme la plus utilisée en démonstration automatique est la *forme normale conjonctive*, en abrégé f.n.c., (ou clause) avec sa duale la *forme normale disjonctive*, en abrégé f.n.d.

**Definition 9** (*forme clause*)

- un *litéral* est un atome ou bien sa négation.
- une *clause* est une disjonction de littéraux.
- une formule est en f.n.c si elle est une conjonction de clauses.

- une formule est en f.n.d si elle est une disjonction de conjonctions de littéraux.

(une conjonction ou disjonction peut-être réduite à un élément).

**Théorème 3** Pour toute  $\varphi \in FORM$  il existe  $\psi$  en f.n.c (resp  $\psi'$  en f.n.d.) tq  $\varphi \equiv \psi$  (resp  $\varphi \equiv \psi'$ ).

Règles de transformation pour la mise en f.n.c.

$$\begin{array}{l} \text{Etape 1: } \varphi \Leftrightarrow \psi \rightarrow (\varphi \Rightarrow \psi) \wedge (\psi \Rightarrow \varphi) \\ \varphi \Rightarrow \psi \rightarrow \neg\varphi \vee \psi \end{array}$$

$$\begin{array}{l} \text{Etape 2: } \neg\neg\varphi \rightarrow \varphi \\ \neg(\varphi \vee \psi) \rightarrow \neg\varphi \wedge \neg\psi \\ \neg(\varphi \wedge \psi) \rightarrow \neg\varphi \vee \neg\psi \end{array}$$

$$\begin{array}{l} \text{Etape 2: } \varphi \vee (\psi_1 \wedge \psi_2) \rightarrow (\varphi \vee \psi_1) \wedge (\varphi \vee \psi_2) \\ (\psi_1 \wedge \psi_2) \vee \varphi \rightarrow (\psi_1 \vee \varphi) \wedge (\psi_2 \vee \varphi) \end{array}$$

**Proposition 5** L'algorithme calcule une forme clausale et termine.

**Preuve** Correction: La formule résultat ne contient ni  $\Leftrightarrow$  ni  $\Rightarrow$  et une sous-formule  $\psi_1 \vee \psi_2$  du résultat est telle que  $\psi_i$  ne contient pas de conjonction sinon on applique la distributivité, et une négation s'applique forcément à une variable propositionnelle.

Terminaison: la terminaison est facile à prouver si on fixe une stratégie, sinon on utilise le lemme de König (ou un ordre de terminaison comme le rpo).

□

Exemple: calcul d'une f.n.c. de  $(p \wedge \neg q) \vee (q \wedge \neg r) \vee r$ .

$$\frac{\frac{(p \wedge \neg q) \vee (q \wedge \neg r) \vee r}{(p \vee (q \wedge \neg r) \vee r) \wedge (\neg q \vee (q \wedge \neg r) \vee r)}}{(p \vee q \vee r) \wedge (p \vee \neg r \vee r) \wedge (\neg q \vee q \vee \neg r) \wedge (\neg q \vee q \vee r)}$$

Remarque: Une f.n.c. (resp. f.n.d.) n'est pas forcément unique même à l'associativité commutativité de  $\wedge$  et  $\vee$  près (voir exercice).

**Exercice :** On donne la table de vérité d'une formule: retrouver la (une) formule en fnd (resp fnc) correspondante. Traiter le cas:

$p$	1	1	1	1	0	0	0	0
$q$	1	1	0	0	1	1	0	0
$r$	1	0	1	0	1	0	1	0
$f(p, q, r)$	1	1	0	0	0	0	0	1

**Exercice :** Montrer que la fnd (resp. fnc) n'est pas unique même à l'associativité-commutativité de  $\wedge$  et  $\vee$  près. (on peut considérer la formule  $(p \wedge q \wedge r) \vee (\neg p \wedge \neg q \wedge \neg r)$ ). Calculer la forme clausale de  $(p_1 \wedge q_1) \vee \dots \vee (p_n \wedge q_n)$ . Si on appelle taille d'une formule le nombre de symboles propositionnels de son écriture, que peut-on dire de la mise en forme clausale? Ecrire un algorithme de transformation de la fnc en fnd.

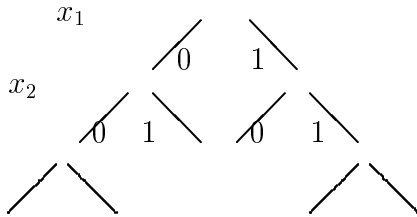
## 1.4 Le théorème de compacité

Le théorème de compacité permet de relier satisfiabilité pour un ensemble infini et satisfiabilité pour des ensembles finis. Il peut s'interpréter de manière topologique d'où son nom.

**Théorème 4** Soit  $\Sigma$  un ensemble infini de formules, alors  $\Sigma$  est satisfiable ssi tout sous-ensemble fini de  $\Sigma$  est satisfiable.

**Preuve** La condition nécessaire est évidente.

Pour la condition suffisante, on considère l'arbre des valuations sur l'ensemble des variables propositionnelles.



Chaque noeud  $n$  correspond à la restriction d'une assignation  $\tilde{n}$  à un sous-ensemble initial de variable propositionnelles  $\{p_1, \dots, p_n\}$ . On dit que  $n$  est un noeud d'échec s'il existe une formule  $\sigma$  de  $\Sigma$  dont les variables propositionnelles sont dans  $\{p_1, \dots, p_m\}$  telle que  $m \leq n$  et  $\tilde{n}(\sigma) = 0$ . Supposons que tout sous-ensemble fini de  $\Sigma$  est satisfiable. Alors toute branche de l'arbre falsifie  $\Sigma$  et donc contient un noeud d'échec. Considérons le sous-arbre obtenu

en coupant tout ce qui est sous un noeud d'échec. Ce sous-arbre n'a que des branches finies, donc d'après le lemme de König, est fini. Chacune de ses branches est terminée par un noeud d'échec pour une formule  $\sigma_i$ . Comme il n'y a qu'un nombre fini de branches l'ensemble  $\Sigma'$  des formules  $\sigma_i$  est un sous-ensemble fini de  $\Sigma$  qui est insatisfiable, d'où la contradiction.  $\square$

## 1.5 Système déductif: méthode des tableaux sémantiques

En logique, on cherche à démontrer des théorèmes (les tautologies). On peut le faire soit directement à partir de la sémantique ce qui est coûteux et même impossible dès que la logique est suffisamment riche, soit on utilise un moyen *mécanique* qui ne travaille que sur la syntaxe (l'écriture) des formules. Ce moyen est un *système déductif* (ou formel) qui permet donc de mécaniser le raisonnement. Un système déductif est formé d'un langage, d'un ensemble (peut-etre vide) de schéma d'axiomes et de règles d'inférence. Le système déductif dit "méthode des tableaux sémantiques" est formé:

- du langage des formules du CP0
- ne contient pas de schéma d'axiomes
- des règles d'inférences:

$$\begin{array}{ccccc}
 \frac{1, \neg\varphi}{0, \varphi} & \frac{1, \varphi \wedge \psi}{1, \varphi} & \frac{1, \varphi \vee \psi}{1, \varphi} & \frac{1, \varphi \Rightarrow \psi}{0, \varphi} & \frac{1, \varphi \Leftrightarrow \psi}{1, \varphi} \\
 & | & \backslash & \backslash & \backslash \\
 & 1, \psi & 1, \psi & 1, \psi & 0, \varphi \\
 & & & & | \\
 & & & & 1, \psi \\
 & & & & 0, \psi \\
 \\
 \frac{0, \neg\varphi}{1, \varphi} & \frac{0, \varphi \wedge \psi}{0, \varphi} & \frac{0, \varphi \vee \psi}{0, \varphi} & \frac{0, \varphi \Rightarrow \psi}{1, \varphi} & \frac{0, \varphi \Leftrightarrow \psi}{1, \varphi} \\
 & / & | & | & / \\
 & 0, \varphi & 0, \psi & 0, \psi & 0, \varphi \\
 & & & & | \\
 & & & & 1, \psi
 \end{array}$$

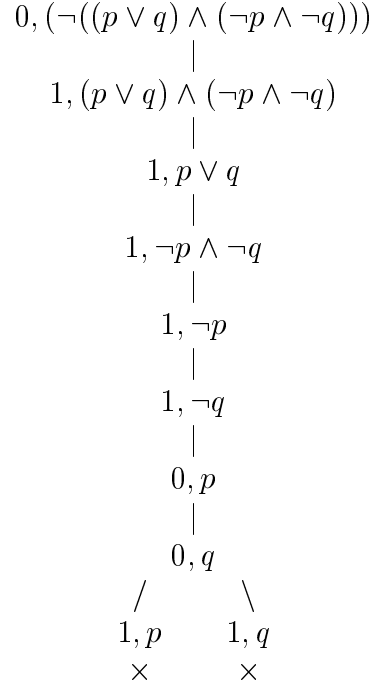
Remarque: une autre règle possible pour l'implication est

$$\frac{1, \varphi \Rightarrow \psi}{\begin{array}{l} / \quad \backslash \\ 0, \varphi \quad 1, \varphi \\ \quad \quad | \\ \quad \quad 1, \psi \end{array}}$$

Les principes de base de la méthode des tableaux sémantiques sont:

1. La méthode procède par l'absurde (méthode réfutationnelle): on montre que donner la valuation 0 au théorème conduit à une impossibilité dans tous les cas.
  
2. on construit un arbre (fini) étiqueté par des formules, appelé tableau sémantique, de la façon suivante:
  - on initialise la construction en mettant  $0, \varphi$  à la racine si  $\varphi$  est le théorème à prouver.
  
  - on choisit une branche  $B$  de l'arbre et un noeud  $x, \psi_0$  dans  $B$  et on fait pousser la branche  $B$  en utilisant les règles d'inférence.
  
  - une branche qui contient un noeud  $0, \gamma$  et un noeud  $1, \gamma$  ne sera plus développée car elle contient une contradiction (ce qu'on symbolise avec  $\times$ ).

Exemple de construction d'un tableau:



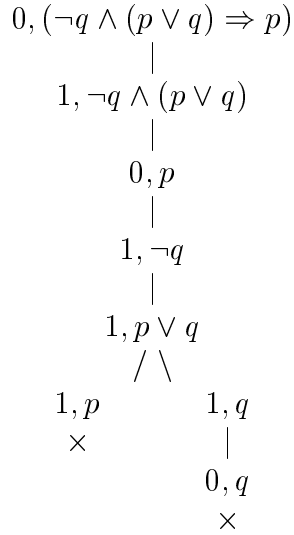
**Definition 10** Les concepts essentiels sur les tableaux sont les suivants.

- un noeud  $N$  d'une branche  $B$  est réduit si  $B$  contient une branche du tableau atomique de racine  $N$ .
- Une branche est contradictoire ou fermée si elle contient les noeuds  $0, \alpha$  et  $1, \alpha$  pour un  $\alpha$  quelconque.
- Une branche est terminée si elle est contradictoire ou si tous ses noeuds sont réduits.
- Un tableau est terminé si toutes ses branches le sont. Un tableau est contradictoire si toutes ses branches le sont

On peut alors définir la notion de preuve et la relation de déduction  $\vdash$ .

**Definition 11** Une preuve (par la méthode des tableaux) de  $\varphi$  est un tableau contradictoire de racine  $0, \varphi$ . On note  $\vdash \varphi$  s'il existe une preuve de  $\varphi$ .

Exemple: prouver  $(\neg q \wedge (p \vee q) \Rightarrow p)$



**Exercice :** Un ensemble d'axiomes d'un système formel est dit indépendant ssi aucun des axiomes ne peut être déduit à partir des autres en utilisant les règles d'inférence de S. Donner un ensemble d'axiomes non indépendants pour le calcul propositionnel.

**Exercice :** Utiliser la méthode des tableaux sémantiques pour vérifier ou infirmer les affirmations:

- $\vdash ((s \Rightarrow r) \wedge p \wedge q) \Rightarrow (\neg r \wedge \neg s \wedge p)$
- $\vdash (((p \Rightarrow r \wedge t) \wedge (t \vee s)) \Rightarrow (\neg q)) \Rightarrow (\neg(p \wedge q))$
- $\vdash ((p \Rightarrow (\neg q \vee r)) \wedge (q \Rightarrow (p \wedge \neg r))) \Rightarrow (r \Rightarrow q)$

## 1.6 Correction et complétude

Dès qu'on introduit un système déductif, on se pose deux questions:

Est-il *correct*, c.a.d est-ce que le mécanisme de déduction ne contredit pas la sémantique choisie?

Est-il *complet*, c.a.d est-il suffisamment riche pour prendre en compte tout ce qui est décrit par la sémantique?

Tout système doit être correct, par contre certains systèmes sont incomplets (mais restent adéquats pour l'usage qui en est fait, par exemple la logique intuitionniste)

### 1.6.1 Preuve de correction

Dans un premier temps nous montrons la correction de la méthode des tableaux sémantiques, c.a.d  $\vdash \varphi$  entraîne  $\models \varphi$ . Le raisonnement est par contraposée, c.a.d on montre que si  $\not\models \varphi$  alors  $\not\vdash \varphi$ .

**Definition 12** *Pour tout  $\varphi$  il existe des tableaux terminés de racine  $0, \varphi$  et  $1, \varphi$ .*

**Preuve** Par induction structurelle sur  $\alpha$ . □

Avant de prouver le résultat de correction, nous montrons le lemme suivant.

**Lemme 2** *Soit  $T$  un tableau de racine  $i, \varphi$  avec  $i = 0$  ou  $i = 1$ . S'il existe une valuation  $v$  telle que  $v(\varphi) = i$  alors il existe une branche de  $T$  telle que pour tout  $j, \psi$  sur cette branche,  $v(\psi) = j$ .*

**Théorème 5**  $\vdash \varphi$  alors  $\models \varphi$ .

**Preuve** Par contraposition. Supposons que  $\not\models \varphi$ , alors il existe une valuation  $v$  telle que  $v(\varphi) = 0$ . S'il existait un tableau contradictoire de racine  $0, \varphi$  la valuation  $v$  serait modèle d'une branche de ce tableau et donc on aurait  $v(\alpha) = 0$  et  $v(\alpha) = 1$  pour une formule  $\alpha$  de cette branche, ce qui est impossible. □

### 1.6.2 Le théorème de complétude

Le théorème de complétude montre que la méthode des tableaux permet de prouver toutes les tautologies. Nous donnons d'abord un lemme.

**Proposition 6** *Soit  $B$  une branche non contradictoire d'un tableau terminé  $T$ , alors toute valuation  $v$  obtenue en posant  $v(p) = i$  si  $i, p$  est sur  $B$  avec  $i = 0, 1$ ,  $p$  une variable propositionnelle, vérifie  $v(i, \alpha) = i$  pour tout noeud  $i, \alpha$  de  $B$ .*

**Preuve** Par induction structurelle sur  $\alpha$ . □

Nous en déduisons le théorème de complétude:

**Théorème 6**  $Si \models \varphi$  alors  $\vdash \varphi$ .

**Preuve** Par contraposée. Supposons  $not \vdash \varphi$ . Alors soit  $B$  une branche non-contradictoire d'un tableau terminé  $T$  de racine  $0, \varphi$ . Par application du lemme nous savons une valuation telle que  $v(\varphi) = 0$  et donc  $\varphi$  n'est pas une tautologie c.a.d.  $\not\models \varphi$ . □

On peut poser la question du coût de la méthode des tableaux: si  $n$  est le nombre de symboles propositionnels du théorème à prouver, on définit la complexité d'une preuve par le nombre de symboles apparaissant dans la preuve. Il est alors facile de construire une formule dont toute preuve par la méthode des tableaux a un coût exponentiel en  $n$  (exercice).

### 1.6.3 Traitement de la conséquence logique

Les tautologies ne sont pas des formules très intéressantes en soi ( $p \vee \neg p$  par exemple). Par contre, il est plus utile de savoir ce qu'on peut déduire d'un ensemble d'hypothèses, c.a.d si une formule  $\varphi$  est conséquence logique d'un ensemble de formules  $\Gamma$  éventuellement infini. D'après le théorème de compacité on a:  $\Gamma \models \varphi$  ssi il existe  $\Gamma_0 \subseteq \Gamma$  tel que  $\Gamma_0$  est fini et  $\Gamma_0 \models \varphi$ .

Il suffit de modifier la méthode des tableaux en rajoutant une règle de  $\Gamma$ -introduction qui autorise à rajouter  $1, \gamma$  à une branche si  $\gamma \in \Gamma$ . On écrit  $\Gamma \vdash \varphi$  ssi il existe un arbre fermé pour  $\neg \varphi$  (avec utilisation de la règle de  $\Gamma$ -introduction). On a alors:

**Théorème 7**  $\Gamma \vdash \varphi$  ssi  $\Gamma \models \varphi$

**Exercice :** Montrer que  $\{\varphi_1, \dots, \varphi_n\} \vdash \psi$  ssi  $\models (\varphi_1 \wedge \dots \wedge \varphi_n) \Rightarrow \psi$ .

**Exercice :** Peut-on montrer à l'aide de la méthode des tableaux (si non justifier, si oui donner une preuve effective c.a.d un tableau fermé):

- $\{\neg p \Rightarrow q, r \Rightarrow \neg p\} \models \neg p \Rightarrow (\neg r \Rightarrow q)$
- $\{(s \Rightarrow r) \wedge p, q\} \models \neg r \wedge \neg s \wedge p$
- $\{p \Rightarrow (\neg q \vee r), q \Rightarrow (p \wedge \neg r)\} \models r \Rightarrow q$

## 1.7 Autres systèmes déductifs

### 1.7.1 La Résolution

A la différence de la méthode des tableaux sémantiques, la résolution impose que les formules traitées soient en forme clausale <sup>2</sup> (donc un prétraitement est nécessaire avant de commencer les déductions). La méthode est aussi réfutationnelle (on débute avec la négation du théorème), et la règle de résolution <sup>3</sup> est:

$$\frac{L \vee C, \neg L \vee C'}{C \vee C'}$$

Une preuve par résolution consiste à construire un ensemble de clauses à partir des clauses initiales (une formule en f.n.c. donne l'ensemble de ses conjoints) et à y rajouter les clauses qu'on peut engendrer par la règle d'inférence jusqu'à obtenir un ensemble qui contient la clause vide (notée  $\square$ ).

Exemple: prouver  $(\neg q \wedge (p \vee q) \Rightarrow p)$ . On commence par calculer la f.n.c. de la négation du théorème  $\neg q \wedge (p \vee q) \wedge \neg p$  qui correspond à l'ensemble de clauses  $E = \{\neg q, p \vee q, \neg p\}$  et on effectue les résolutions:

$$\frac{\frac{p \vee q \quad \neg q}{p} \quad \neg p}{\square}$$

Dans la preuve ci-dessus, les prémisses d'une règle d'inférence sont des clauses de l'ensemble déjà construit, et la conclusion est une nouvelle clause qu'on peut donc utiliser comme prémisse pour les inférences suivantes. La résolution sera étudiée en détail par la suite.

**Exercice :** Formaliser le principe des tiroirs et chaussettes (si on a  $n > 0$  tiroirs et  $n + 1$  paires de chaussettes rangées dans les tiroirs, alors il y a au moins un tiroir avec 2 paires de chaussettes).

- Introduire  $n * (n + 1)$  prédicats  $h_{i,j}$  dont la signification est "la  $i$ ème paire est dans le  $j$ ème tiroir" et exprimer le théorème à l'aide d'une formule  $T_n$  qui utilise ces prédicats. Prouver informellement que la formule est une tautologie.

---

<sup>2</sup>il est possible de donner des règles pour des formules générales, qui correspondent en fait à une mise en f.n.c

<sup>3</sup>pour avoir un système déductif complet, il faut aussi avoir la règle de factorisation, voir cours de logique II

- Prouver par résolution la formule dans le cas  $n = 2$ . (on peut montrer que la taille d'une preuve de  $T_n$  par résolution croît exponentiellement avec  $n$ )

## 1.7.2 Le système de Hilbert

Le système de Hilbert est un système adapté à la modélisation du raisonnement mathématique, qu'on présente ici dans un cadre restreint à 1 connecteur logique  $\Rightarrow$  et un seul symbole propositionnel spécial  $\perp$  signifiant "faux". Le système de Hilbert défini sur un langage construit avec  $\Rightarrow$  et  $\perp$  et des atomes, comporte des (schémas d') axiomes et une règle d'inférence qui sont:

$\mathcal{L}$  = formules construites avec  $\Rightarrow$  et un ensemble de symboles propositionnel contenant  $\perp$   
 $\mathcal{R}$  est composé d'une seule règle le modus ponens:  
 (MP)  $\frac{A \quad A \Rightarrow B}{B}$   
 $\mathcal{A}$  l'ensemble des axiomes est :  
 $A \Rightarrow (B \Rightarrow A)$   
 $(A \Rightarrow (B \Rightarrow C)) \Rightarrow ((A \Rightarrow B) \Rightarrow (A \Rightarrow C))$   
 $((A \Rightarrow \perp) \Rightarrow \perp) \Rightarrow A$  (tiers exclu car équivalent à  $A \vee \neg A$ )

Remarque: si on élimine l'axiome du tiers exclu on obtient la logique appelée logique minimale.

Une déduction de  $\varphi$  à partir de  $\Gamma$  est une suite  $\varphi_0, \varphi_1, \dots, \varphi_n$  telle que:

- $\varphi_n = B$
- $\varphi_i$  est soit un axiome, soit un élément de  $\Gamma$ , soit  $\varphi_i$  est la conclusion d'une règle d'inférence dont les prémisses sont des  $\varphi_j$  avec  $j < i$

et on note  $\Gamma \vdash_H \varphi$ .

Montrons que  $\vdash_H A \Rightarrow A$ .

(1)  $A \Rightarrow (B \Rightarrow A)$  premier axiome

(2)  $A \Rightarrow ((B \Rightarrow A) \Rightarrow A)$  premier axiome instancié par  $B \leftarrow B \Rightarrow A$

(3)  $(A \Rightarrow ((B \Rightarrow A) \Rightarrow A)) \Rightarrow ((A \Rightarrow (B \Rightarrow A)) \Rightarrow (A \Rightarrow A))$  deuxième axiome avec  $B \leftarrow (B \Rightarrow A)$  et  $C \leftarrow A$

(4)  $(A \Rightarrow (B \Rightarrow A)) \Rightarrow (A \Rightarrow A)$  modus ponens entre (2) et (3)

(5)  $A \Rightarrow A$  par modus ponens entre (4) et (1)

**Exercice:** : démontrer  $\vdash ((A \Rightarrow \perp) \Rightarrow \perp) \Rightarrow A$  puis  $\vdash_H (A \Rightarrow B) \Rightarrow ((B \Rightarrow C) \Rightarrow (A \Rightarrow C))$

Un théorème très utile dans ce système formel est:

**Théorème 8** (*théorème de la déduction*)  $\Gamma \vdash_H (A \Rightarrow B)$  ssi  $\Gamma \cup \{A\} \vdash_H B$

**Preuve** Par induction sur la longueur de la dérivation (exercice) □

Ce théorème facilite énormément les preuves dans le système de Hilbert, par exemple la preuve de  $\vdash_H A \Rightarrow A$  est immédiate.

**Exercice:** : démontrer les deux formules de l'exercice précédent en utilisant le théorème de la déduction. Démontrer  $\{p \Rightarrow q, (\neg p) \Rightarrow q\} \vdash_H q$ .

Il existe d'autres systèmes de preuve classiques comme la déduction naturelle (plus proche du système à la Hilbert car contenant le modus ponens mais qui contient aussi des règles structurelles sur la forme des formules comme la méthode des tableaux) ou le calcul des séquents.



# Chapter 2

## Calcul des Prédicats du premier ordre

Le calcul propositionnel a une expressivité limitée. En particulier on ne peut pas parler d'individus, ni de fonctions applicables à des individus. Le calcul des prédicats du premier ordre étend le calcul propositionnel en ajoutant cette notion de variable d'individus, de fonction et de relations (ou prédicats) sur ces individus ainsi que les connecteurs  $\forall$  et  $\exists$  qui permettent de quantifier sur l'ensemble des individus.

### 2.1 Le langage du calcul des prédicats

#### 2.1.1 Syntaxe

Pour commencer nous donnons les éléments syntaxiques de base du langage.

**Definition 13** *Un langage  $L$  du CP1 est formé de:*

- *connecteurs logiques  $\wedge, \vee, \Rightarrow, \Leftrightarrow, \neg,$*
- *quantificateurs  $\exists, \forall,$*
- *un ensemble dénombrable de variables  $V,$*
- *un ensemble de symboles de relation  $R,$*
- *un ensemble de symboles de fonction  $F,$*

- des symboles de ponctuation “(, ”, ”, “

Chaque symbole de relation ou de fonction a une arité fixe.

## 2.1.2 Termes et substitutions

L'ensemble des termes sur  $F$  et  $X$ , noté  $T(V, F)$  est le plus petit ensemble tel que:

- si  $x \in V$  alors  $x \in T(V, F)$ ,
- si  $f \in F$  et  $f$  d'arité  $n$ ,  $t_1, \dots, t_n$  sont dans  $T(V, F)$  alors  $f(t_1, \dots, t_n) \in T(V, F)$ .

Les symboles de fonction d'arité 0 sont les *constantes*.

L'ensemble noté  $Var(t)$  est l'ensembles des variables du terme  $t$ . Un terme  $t$  est clos ou fermé s'il est sans variable, c.a.d  $Var(t) = \emptyset$ .

**Definition 14** Une substitution  $\sigma$  est un morphisme de  $T_F(X)$ , c.a.d. une application de  $T_F(X)$  dans lui-même telle que  $f(t_1, \dots, t_n)\sigma = f(t_1\sigma, \dots, t_n\sigma)$  (l'application de  $\sigma$  au terme  $t$  se note  $t\sigma$ ).

Pour déterminer complètement une substitution il suffit de connaître les valeurs  $x\sigma$  pour les variables  $x$  telles que  $x\sigma \neq x$ .

**Definition 15** Le domaine d'une substitution  $Dom(\sigma)$  est l'ensemble des variables telles que  $x\sigma \neq x$ .

Le domaine d'une substitution peut être infini, mais dans la suite seules interviennent des substitutions à domaine fini, et on décrira une substitution  $\sigma$  par  $\{x_1 \leftarrow t_1, \dots, x_n \leftarrow t_n \dots\}$  avec  $Dom(\sigma) = \{x_1, \dots, x_n, \dots\}$  et  $x_i\sigma = t_i$  pour  $x_i \in Dom(\sigma)$ .

Exemple d'application de substitution: soit  $t = f(x, y)$  et  $\sigma = \{x \leftarrow s(z)\}$ ,  $\theta = \{x \leftarrow y, y \leftarrow s(z)\}$  alors  $t\sigma = f(s(z), y)$ ,  $t\theta = f(y, s(z))$ .

Une opération utile dans les procédures de déduction automatiques est l'unification: une substitution  $\sigma$  unifie  $s$  et  $t$  ssi  $s\sigma = t\sigma$ . Un unificateur de  $s$  et  $t$   $\sigma$  est plus général qu'un unificateur  $\theta$  de  $s$  et  $t$  ssi  $\theta$  est la composition de  $\sigma$  et d'une autre substitution. On peut montrer que deux termes du premier ordre soit n'ont aucun unificateur, soit ont un unificateur le plus général (p.g.u.) qui est unique à renommage près. Le calcul d'un p.g.u. peut se faire en temps linéaire en la taille des termes.

### 2.1.3 Formules

L'objet essentiel du CP1 est la formule.

**Definition 16** *Un atome est une formule de la forme  $p(t_1, \dots, t_m)$  avec  $t_1, \dots, t_m$  des termes,  $p$  un symbole de prédicat d'arité  $m$ . L'ensemble des formules bien formées est le plus petit ensemble  $FORM$  tel que:*

- *tout atome est dans  $FORM$ ,*
- *si  $\varphi$  est dans  $FORM$  alors:  $(\neg\varphi)$ ,  $(\forall x : \varphi)$ <sup>1</sup>,  $(\exists x : \varphi)$ <sup>2</sup> sont dans  $FORM$ .*
- *si  $\varphi$  et  $\psi$  sont dans  $FORM$  alors  $(\varphi \wedge \psi)$ ,  $(\varphi \vee \psi)$ ,  $(\varphi \Rightarrow \psi)$ ,  $(\varphi \Leftrightarrow \psi)$  sont dans  $FORM$ .*

Par exemple,  $plus-grand(x, s(y))$  est une formule bien formée mais pas  $plus-grand(plus-grand(x, y), z)$ .

Un littéral est un atome ou la négation d'un atome. Une variable peut être liée par un quantificateur ou non, et son rôle diffère selon le cas. Définissons les variables libres:

**Definition 17** *Si  $\varphi$  est une formule, alors on définit  $FV(\varphi)$  l'ensemble des variables libres de  $\varphi$  par:*

- *si  $\varphi$  est un atome,  $FV(\varphi)$  est l'ensemble des variables de  $\varphi$*
- *$FV(\forall x : \varphi) = FV(\exists x : \varphi) = FV(\varphi) - \{x\}$*
- *$FV(\neg\varphi) = FV(\varphi)$*
- *$FV(\varphi \circ \psi) = FV(\varphi) \cup FV(\psi)$  où  $\circ$  est  $\wedge$  ou  $\vee$  ou  $\Rightarrow$  ou  $\Leftrightarrow$ .*

Les occurrences de  $x$  dans  $(\exists x : \varphi)$  (respectivement  $(\forall x : \varphi)$ ) sont *liées* et  $x$  est appelée une variable liée. Dans une formule, une variable peut être à la fois liée et libre, comme dans  $p(x) \wedge \forall x : p(x)$  On dit qu'une formule est *close* (ou fermée) si elle ne contient pas de variables libres. Par exemple  $(\forall x : (p(x) \wedge (\exists y, q(s(y)))))$  est close mais  $((\forall x : \exists y : p(x, y)) \wedge q(y))$  ne l'est pas.

---

<sup>1</sup>on dira que  $x$  est liée par  $\forall$

<sup>2</sup>idem pour  $x$  et  $\exists$

On choisit de n'appliquer une substitution  $\sigma$  à une formule  $\varphi$  que si elle est libre pour  $\varphi$ <sup>3</sup>, notion définie comme suit:  $\sigma$  est libre pour un littéral, pour  $\neg\varphi$  si elle est libre pour  $\varphi$ , pour  $\varphi \circ \psi$  si elle est libre pour  $\varphi$  et  $\psi$  et pour  $\forall x : \varphi$  (resp.  $\exists x : \varphi$ ) si  $y\sigma$  ne contient pas  $x$  pour toute variable libre  $y$  de  $\varphi$  et  $\sigma_x$  la substitution égale à  $\sigma$  pour toute variable  $y \neq x$  et telle que  $x\sigma_x = x$  est libre pour  $\varphi$ . On définit alors  $\varphi\sigma$  par:

- $\varphi = p(t_1, \dots, t_n)$  alors  $\varphi\sigma = p(t_1\sigma, \dots, t_n\sigma)$
- $\varphi = \varphi_1 \circ \varphi_2$  alors  $\varphi\sigma = \varphi_1\sigma \circ \varphi_2\sigma$
- $\varphi = (\forall x : \psi)$  alors  $\varphi\sigma = \forall x : \psi\sigma_x$
- $\varphi = (\exists x : \psi)$  alors  $\varphi\sigma = \exists x : \psi\sigma_x$

Exemple: si  $\sigma = \{x \leftarrow s(y)\}$  et  $\varphi = (\forall z : (\forall x : p(x)) \wedge q(x, z))$  alors  $\varphi\sigma = (\forall z : (\forall x : p(x)) \wedge q(s(y), z))$ . Les restrictions sur les variables n'en sont pas réellement car il est toujours possible d'obtenir une expression équivalente à  $\varphi$  par renommage des variables quantifiées. Pour une substitution  $\{x \leftarrow t\}$  on pourra noter  $\varphi\sigma$  par  $\varphi_{x \leftarrow t}$ .

## 2.2 Sémantique

Donner un sens 1 (pour vrai) ou 0 (pour faux) aux formules demande plus d'effort que pour le calcul propositionnel car il faut donner un sens aux variables d'individus, aux fonctions et aux relations.

**Definition 18** Soit  $L$  un langage sur  $V, R, F$ , alors une  $L$ -structure est une paire  $\langle D, I \rangle$  avec  $D$  un ensemble non vide, le domaine,  $I$  une interprétation qui associe

- à une constante  $c$  de  $F$ , un élément  $c^I$  de  $D$ ,
- à un symbole de fonction  $f$  d'arité  $n$ , une fonction  $f^I$  de  $D^n \rightarrow D$ ,
- à un symbole de relation  $r$  d'arité  $n$ , une relation  $r^I$  d'arité  $n$  sur  $D^n$ .

Une fois fixée l'interprétation des symboles de fonction et de relation, il ne reste plus qu'à donner des valeurs aux variables.

---

<sup>3</sup>on peut définir  $\varphi\sigma$  pour  $\sigma$  quelconque

**Definition 19** Une assignation  $A$  est une application  $A : V \rightarrow D$  qu'on étend aux termes par morphisme, c.a.d en posant  $A(c) = c^I$

- $A(c) = c^I$  pour une constante  $c$ ,
- $A(f(t_1, \dots, t_n)) = f^I(A(t_1), \dots, A(t_n))$  pour un terme fonctionnel.

**Exemple 2**  $F = \{0, s, +\}$   $R = \{=\}$ , une structure possible est  $\langle \mathbb{N}, I \rangle$  avec  $0^I = 0$  (le zéro usuel),  $s^I$  la fonction successeur qui à un entier associe l'entier plus un,  $+^I$  l'addition usuelle. de même  $=^I$  l'interprétation de  $=$  sera l'égalité entre entiers.

Une assignation  $B$  est une  $x$ -variante de  $A$  ssi elle est identique à  $A$  sur toute variable  $y$  différente de  $x$ .

**Definition 20** Etant données  $\langle D, I \rangle$  une structure et une assignation  $A$ , on peut donner à une formule  $\varphi$  une valeur de vérité  $\varphi^{I,A}$  qui vaut 0 ou 1 comme suit.

- $p(t_1, \dots, t_n)^{I,A} = 1$  ssi  $(t_1^{I,A}, \dots, t_n^{I,A}) \in p^I$
- $(\neg\varphi)^{I,A} = 1$  ssi  $(\varphi)^{I,A} = 0$
- $(\varphi\{\wedge, \vee, \Rightarrow, \Leftrightarrow\}\psi)^{I,A} = \varphi^{I,A}\{\wedge, \vee, \Rightarrow, \Leftrightarrow\}\psi^{I,A}$ ,
- $(\exists x \varphi)^{I,A} = 1$  ssi il existe une  $x$ -variante  $B$  de  $A$  s.t.  $\varphi^{I,B} = 1$
- $(\forall x \varphi)^{I,A} = 1$  ssi pour toutes  $x$ -variantes  $B$  de  $A$  s.t.  $\varphi^{I,B} = 1$

Une remarque importante est que si  $\varphi$  est une formule close alors pour toutes assignations  $A, B$ , on a  $\varphi^{I,A} = \varphi^{I,B}$  qu'on notera alors  $\varphi^I$ . Les notions utilisées sur les formules sont les suivantes:

- la formule  $\varphi$  est satisfiable dans  $\langle D, I \rangle$  ssi il existe  $A$  telle que  $\varphi^{I,A} = 1$ . On dit alors que  $\langle D, I \rangle$  est un modèle de  $\varphi$ .
- la formule  $\varphi$  est vraie dans  $\langle D, I \rangle$  ssi  $\varphi^{I,A} = 1$  pour toute assignation  $A$ .
- la formule  $\varphi$  est valide si elle est vraie dans toute structure (sur son langage) ce qu'on note par  $\models \varphi$ .

- une formule close  $\varphi$  est conséquence logique d'un ensemble de formules closes  $\Gamma$  ssi  $\varphi$  est vraie dans toute structure ou toutes les formules de  $\Gamma$  sont vraies.

La notion de modèle est très générale car le choix des interprétations est illimité. On peut prendre n'importe quel ensemble comme domaine ce qui rend toute approche sémantique très difficile. Par exemple le programme logique écrit à la Prolog:

*aime-bien(enfant(x), x)  $\Leftarrow$  humain(x)  $\wedge$  sage(enfant(x))*  
*sage(enfant(enfant(Paul))).*  
*humain(enfant(enfant(Paul))).*

(tout est implicitement universellement quantifié) a un modèle dont le domaine est  $\mathbf{R}$ , où *Paul* s'interprète par la constante 2, *enfant* par la fonction  $x \rightarrow x^3$  et les prédicats *aime-bien(x, y)* par  $x \geq y$ , *sage(x)* par  $x \geq 1$  et *humain(x)* par  $x \in \mathbf{R}$ .

Heureusement certains modèles ont des propriétés très remarquables et simplifient l'approche sémantique: ce sont les modèles de termes ou de Herbrand.

**Definition 21** *Un modèle est un modèle de Herbrand ssi*

- le domaine est l'ensemble des termes clos,
- chaque fonction  $f$  d'arité  $n$  s'interprète par "elle-même", c.a.d.  $f^I$  est la fonction  $f^I : t_1, \dots, t_n \rightarrow f(t_1, \dots, t_n)$ .

Par exemple, un modèle de Herbrand de  $\forall x : (\text{pair}(x) \Rightarrow \text{pair}(s(s(x)))) \wedge p(0)$  (le lecteur a reconnu un programme PROLOG) est (si  $F = \{0, s\}$ ) déterminé par la relation  $I(\text{pair})(y)$  est vrai ssi  $y$  est soit 0 soit de la forme  $s^{2n}(0)$ , qu'on abrège souvent en *le modèle est*  $\{\text{pair}(0), \text{pair}(s(s(0))), \dots, \text{pair}(s^{2n}(0)), \dots\}$  car  $D$  et l'interprétation des symboles de fonction est connue. Cela dit une formule peut ne pas avoir de modèle de Herbrand (tout au moins sur le langage initial), par exemple rajouter l'axiome  $\exists y : 0 < y < s(0)$  aux axiomes de Peano donnés dans le paragraphe suivant.

**Exercice:** : Un modèle fini est un modèle dont le domaine est fini. Donner une formule satisfiable qui n'a pas de modèle fini. Donner une formule qui est satisfiable par tout modèle fini. Peut-on trouver une formule qui

admette des modèles de n'importe quelle cardinalité finie et qui n'est pas satisfiable par un modèle infini. Donner une formule satisfiable par toute interprétation de domaine de cardinalité 1 qui peut être falsifiée dans tout domaine de cardinalité supérieure ou égale à 2.

**Exercice :** Donner un modèle de la formule  $\forall x : \forall y (P(f(x, y), a) \Rightarrow P(x, y))$  puis en donner une interprétation qui la falsifie. Même question avec  $\forall x : \exists y : P(x, y) \Rightarrow \forall x P(x, f(x))$

**Exercice :** Donner un modèle de la formule:  $\forall x : \exists y : P(x, f(f(x, y), y))$ . Donner un modèle de Herbrand de la formule:  $P(s(0)) \wedge \forall x : (P(x) \Rightarrow P(s(x)))$  Donner un modèle de la formule précédente dans lequel  $\forall x : P(s(x))$  est faux.

## 2.3 Théorèmes caractéristiques

Les deux théorèmes suivants, que nous donnons sans démonstration sont caractéristiques de la logique du premier ordre.

**Théorème 9** (*Lowenheim-Skolem*)

*Si un ensemble dénombrable de formules closes est satisfiable, alors cet ensemble admet un modèle dénombrable.*

Une application est la suivante: le corps des nombres réels  $\mathbf{R}$  n'est pas axiomatisable au premier ordre c.a.d. qu'il n'existe pas d'ensemble d'axiomes du premier ordre dont le seul modèle est le corps des réels. En effet s'il existe un tel ensemble d'axiome, cet ensemble a un modèle dénombrable qui n'est donc pas  $\mathbf{R}$ , celui-ci n'étant pas dénombrable.

**Théorème 10** (*compacité*)

*Un ensemble de formules closes est satisfiable ssi tous ses sous-ensembles finis sont satisfiables.*

Une application est de montrer que tout ensemble d'axiome qui admet des modèles finis de cardinalité quelconque admet un modèle infini (exercice).

Une autre application est l'existence de modèles non-standards pour l'arithmétique de Péano. L'arithmétique de Péano est définie à partir du langage:

- les symboles de fonction 0 d'arité 0,  $s$  d'arité 1 (la fonction successeur),  $+$ ,  $*$  d'arité 2 (l'addition et la multiplication)
- le symbole de prédicat  $=$  (l'égalité),
- les axiomes  $\forall x(s(x) \neq 0)$ ,  $\forall xy(s(x) = s(y) \Rightarrow x = y)$ ,  $\forall x(x + 0 = x)$ ,  $\forall xy(s(x) + y = s(x + y))$ ,  $\forall x(x * 0 = 0)$ ,  $\forall xy(x * s(y) = x * y + x)$ ,  $\varphi(0) \wedge \forall x(\varphi(x) \Rightarrow \varphi(s(x))) \Rightarrow \forall x\varphi(x)$ . Ce dernier axiome appelé *schéma d'induction* est un schéma d'axiome qui représente tous les axiomes obtenus en remplaçant  $\varphi$  par n'importe quelle formule ayant une variable libre. *Peano* désigne l'ensemble de ces axiomes.

L'arithmétique usuelle est modèle de ces axiomes et interprète  $\underbrace{s(\dots s(0))}_n$

par l'entier  $n$ , c'est le *modèle standard*. On peut se demander si c'est le seul, le théorème de compacité permet de répondre non. Soit  $a$  un nouveau symbole de constante, posons  $A_n$  comme étant la formule  $0 < a \wedge s(0) < a \wedge \dots \wedge s^n(0) < a$  où  $x < y$  un synonyme de  $\exists z (y = x + z) \wedge \neg(z = y)$ . Alors pour tout  $n$  il existe un modèle de  $Peano \cup \{A_n\}$  obtenu en prenant l'interprétation standard et en interprétant  $a$  par  $n + 1$ . Il est facile de voir que tout sous-ensemble fini de  $Peano \cup \{A_1, A_2, \dots\}$  a un modèle construit sur le même principe. Par conséquent  $Peano \cup \{A_1, A_2, \dots\}$  a un modèle qui ne peut pas être le modèle standard ( $a$  doit s'interpréter par un objet supérieur à tout entier, c'est un entier non-standard). De plus ce modèle est nécessairement un modèle de *Peano* qui a donc des modèles non-standards.

## 2.4 Forme normale préfixe. Skolémisation

### 2.4.1 Forme préfixe

Les formules du premier ordre sont compliquées et on cherche à en simplifier la manipulation en utilisant des formes plus agréables. La mise en forme préfixe permet de regrouper tous les quantificateurs en tête de formule. Une première opération est de lever les ambiguïtés qui apparaissent quand une variable a des occurrences libres et liées ou bien des occurrences liées par des quantificateurs distinctes dans une même formule, comme  $x$  dans  $\forall y : (p(x, y) \wedge \exists x : (q(x) \vee \forall x : p(x, y)))$ . Les deuxième et troisième occurrences de  $x$  qui sont liées, ne jouent pas le même rôle que la première qui est libre.

Dans une occurrence liée le nom de la variable n'a que peu d'importance, la variable est muette. On peut sans changer le sens de la formule, changer la variable pour une autre, à condition de ne pas faire un mauvais choix. Ici, la première occurrence de  $x$  est libre, on n'a donc pas le droit de la toucher. Dans la deuxième partie on a  $\exists x : (q(x) \vee \forall x : p(x, y))$  En enlevant  $\exists x$  on trouve  $q(x) \vee \forall x : p(x, y)$ . On renomme l'occurrence liée de  $x$  en  $z$  pour obtenir  $q(x) \vee \forall z : p(z, y)$  (un mauvais choix est de renommer  $x$  en  $y$ ), et on renomme maintenant  $x$  en  $u$  dans  $\exists x : (q(x) \vee \forall z : p(z, y))$  pour obtenir  $\exists u : (q(u) \vee \forall z : p(z, y))$  et on obtient finalement la formule  $\forall y : (p(x, y) \wedge \exists u : (q(u) \vee \forall z : p(z, y)))$ . On dit qu'on a effectué un *renommage* de variables. Cette opération préserve le sens des formules:

**Proposition 7** *Pour toute formule  $\varphi$  il existe  $\psi$  telle que  $\varphi \equiv \psi$  et toutes les variables quantifiées ou libres sont distinctes et deux occurrences liées de la même variable sont quantifiées par le même quantificateur.*

La deuxième étape est de remonter les quantificateurs (les formules sont supposées renommées) par les règles:

$$\begin{array}{cccc}
\frac{\neg(\exists x \varphi)}{\forall x (\neg\varphi)} & \frac{\neg(\forall x \varphi)}{\exists x \neg\varphi} & \frac{(\forall x \varphi) \wedge \psi}{\forall x (\varphi \wedge \psi)} & \frac{\varphi \wedge (\forall x \psi)}{\forall x (\varphi \wedge \psi)} \\
\frac{(\forall x \varphi) \vee \psi}{\forall x (\varphi \vee \psi)} & \frac{\varphi \vee (\forall x \psi)}{\forall x (\varphi \vee \psi)} & \frac{\varphi \Rightarrow (\forall x \psi)}{\forall x (\varphi \Rightarrow \psi)} & \frac{\varphi \Rightarrow (\exists x \psi)}{\exists x (\varphi \Rightarrow \psi)} \\
\frac{(\forall x \varphi) \Rightarrow \psi}{\exists x (\varphi \Rightarrow \psi)} & \frac{(\exists x \varphi) \Rightarrow \psi}{\forall x (\varphi \Rightarrow \psi)} & \frac{\varphi \Leftrightarrow \psi}{(\varphi \Rightarrow \psi) \wedge (\psi \Rightarrow \varphi)} & 
\end{array}$$

et le résultat:

**Proposition 8** *Toute formule  $\varphi$  est équivalente à une formule en forme préfixe (c.a.d. de la forme  $Q_1 x_1 \dots Q_n x_n \psi$  où  $\psi$  est sans quantificateur et les  $x_i$  distincts, et qu'on peut calculer après renommage et utilisation des règles ci-dessus.*

Exemple: mise en forme préfixe de  $\exists y, z (\forall x p(x, y) \Rightarrow \exists y q(y, x', z))$ .

Étape 1: Renommage ce qui donne  $\exists y, z (\forall x p(x, y) \Rightarrow \exists u q(u, x', z))$ .

Étape 2: utilisation des règles.

$$\frac{\frac{\exists y, z (\forall x p(x, y) \Rightarrow \exists u q(u, x', z))}{\exists y, z, u (\forall x p(x, y) \Rightarrow q(u, x', z))}}{\exists y, z, u, x (p(x, y) \Rightarrow q(u, x', z))}$$

(on abrège  $\forall x \forall y \varphi$  en  $\forall x, y \varphi$  -idem pour  $\exists$ - et on a d'ailleurs  $\forall x, y \varphi \equiv \forall y, x \varphi$ )

## 2.4.2 Skolémisation

On peut réduire le calcul des prédicats au calcul propositionnel en un certain sens. L'idée est de remplacer une formule

$$\forall x \forall y \exists u \exists v P(x, y, u, v)$$

par une formule

$$\forall x \forall y P(x, y, f(x, y), g(x, y))$$

où  $f$  et  $g$  sont de nouveaux symboles: les fonctions de skolem. On remarquera que si la nouvelle formule est satisfiable ssi la formule originale l'est.

### Règle de skolémisation:

$\forall x_1 \dots x_n \exists y \varphi \rightarrow \forall x_1 \dots x_n y \varphi \{y \leftarrow f(x_1, \dots, x_n, X_1, \dots, X_p)$   
 where  $X_1, \dots, X_p$  sont les variables libres de  $\forall x_1 \dots x_n \exists y \varphi$  et où  $f$  est un nouveau symbole de fonction d'arité  $n$  (donc une constante si  $n = 0$ ).

En itérant ce processus sur une formule en forme prénex on obtient une formule universelle après un nombre fini d'étapes. Cette formule est une *forme de skolem* de la formule initiale.

**Proposition 9** *Soit  $\varphi$  une formule et  $\varphi_S$  une de ses forme de skolem, alors  $\varphi$  est satisfiable ssi  $\varphi_S$  est satisfiable.*

**Preuve** Il suffit de montrer que cela est vrai pour la règle de transformation. Soit une formule  $\psi$  de la forme  $\forall x_1 \dots x_n \exists y \varphi$  et sa skolemisée  $\psi$  qui est donc  $\forall x_1 \dots x_n y \varphi \{y \leftarrow f(x_1, \dots, x_n, X_1, \dots, X_p)$ .

- l'implication  $\psi_S$  satisfiable entraîne  $\psi$  est évidente.

- Réciproquement, soit  $\langle D, I \rangle$  une structure et  $A$  une assignation telle que  $(\forall x_1 \dots x_n \exists y \varphi)^{I, A}$  soit vraie. Construisons une fonction  $f_D$  comme suit. A chaque  $d_{x_1}, \dots, d_{x_n}, A(X_1), \dots, A(x_n)$  tuple d'éléments de  $D$ , on associe un élément  $d_y$  tel que  $\varphi\{x_1 \leftarrow d_{x_1}, \dots, x_n \leftarrow d_{x_n}, A(X_1), \dots, A(X_p)\}$  soit vrai. Alors l'interprétation obtenue en prolongeant  $\langle D, I \rangle$  en posant  $f^I = f_D$  est un modèle de  $\forall x_1 \dots x_n y \varphi\{y \leftarrow f(x_1, \dots, x_n, X_1, \dots, X_p)\}$ .

□

**Remarque 1** *Il est essentiel de noter les points suivants:*

- *une formule et sa skolémisée ne sont pas équivalentes en général: considérer la formule  $\forall x \exists y p(x, y)$  et sa skolémisée  $\forall x p(x, f(x))$  et l'interprétation  $D = \mathbb{N}$ ,  $p^I$  est le prédicat  $\geq$  et  $f^I$  la fonction  $x \rightarrow x + 1$ . Cette interprétation est un modèle de la formule mais falsifie sa skolémisée.*
- *une formule peut avoir plusieurs formes prénexes différentes et plusieurs formes skolémisées différentes.*
- *La méthode de skolémisation présentée est particulièrement inefficace. Un bon algorithme utilisera des méthodes de miniscoping (cf l'algorithme de Pierre Marchand)*

**Exercice :** Mettre sous forme prénexe les formules:

- $(\exists x : \varphi(x)) \Rightarrow \forall y \varphi(y)$
- $\neg((\neg(\forall x : \varphi(x)) \vee \forall x : \psi(x)) \wedge ((\exists x \rho(x)) \Rightarrow \forall x : \tau(x)))$
- $(\forall x : \varphi(x)) \Leftrightarrow (\exists x : \psi(x))$

et les mettre en forme de Skolem.

**Exercice :** On donne un langage dont le seul symbole de fonction est  $e$ , et le seul symbole de prédicat est  $=$  (utilisé de façon infixée). Skolémiser la formule  $(\exists x : \forall y : x * y = y * x \wedge x * y = x) \wedge (\forall x : \exists y : x * y = e)$ .

## 2.5 Méthodes des tableaux pour le calcul des prédicats

### 2.5.1 Présentation

Les connecteurs logiques s'interprètent comme en cp0, mais il faut donner des *témoins* de vérité pour les formules existentielles. Pour cela on étend le langage avec un nombre dénombrables de nouvelles constantes  $c_1, c_2, \dots$ , et on appelle  $L_c$  le nouveau langage. La procédure de preuve ne s'applique qu'à des formules closes et on supposera maintenant que **toutes les formules considérées sont closes**. Le principe de la méthode est le même que pour le cp0. On affecte la valuation 0 à la formule et on montre que cela conduit à une impossibilité dans tous les cas. Les règles supplémentaires sont:

$$\frac{1, \forall \varphi}{1, \varphi\{x \leftarrow t\}} \quad t \text{ un terme clos de } L_c \qquad \frac{0, \forall \varphi}{0, \varphi\{x \leftarrow c\}} \quad \begin{array}{l} c \text{ une nouvelle constante de } c_1, c_2, \dots \\ \text{qui n'apparaît pas au-dessus dans la branche} \end{array}$$

$$\frac{0, \exists \varphi}{0, \varphi\{x \leftarrow t\}} \quad t \text{ un terme clos de } L_c \qquad \frac{1, \exists \varphi}{1, \varphi\{x \leftarrow c\}} \quad \begin{array}{l} c \text{ une nouvelle constante de } c_1, c_2, \dots \\ \text{qui n'apparaît pas au-dessus dans la branche} \end{array}$$

**Exemple 3** l'arbre  $0, \forall x p(x) \Rightarrow \forall x p(x)$

$$\begin{array}{c} | \\ 1, \forall x p(x) \\ | \\ 0, \forall x p(x) \\ | \\ 0, p(c) \qquad c \text{ nouveau symbole} \\ | \\ 1, p(c) \end{array}$$

est un tableau construit selon les règles indiquées.

Les notions de branche contradictoire et de tableau fermé restent les mêmes que pour le cp0.

**Definition 22** On écrit  $\vdash \varphi$  ssi il existe un tableau fermé pour  $0, \varphi$ .

**Exemple 4** Prouver  $\vdash \exists x : (P(x) \vee Q(x)) \Rightarrow (\exists x : P(x) \vee \exists x : Q(x))$ .



a les yeux noirs et  $Q(x)$  comme  $x$  a les yeux bleus)

## 2.5.2 Correction

A la différence du cp0, la construction d'un tableau peut ne pas terminer, et on peut donc avoir des tableaux infinis. Pour cette raison nous introduisons la notion de noeud réduit et de tableau terminé. La suite  $t_1, t_2, \dots$  désigne une énumération des termes clos de  $L_c$  (ce qui est possible puisque  $L_c$  est dénombrable).

**Definition 23** *Un noeud  $N$  de la branche  $B$  d'un tableau est réduit sur  $B$  ssi*

- soit il est du type  $1, \forall\varphi$  (resp.  $0, \exists\varphi$ ) et  $B$  contient  $1, \varphi\{x \leftarrow t_1\}, 1, \varphi\{x \leftarrow t_2\}, \dots$  (resp.  $0, \varphi\{x \leftarrow t_1\}, 0, \varphi\{x \leftarrow t_2\}, \dots$ )
- soit il n'est pas de la forme précédente et on a appliqué une règle à ce noeud.

*Un tableau est terminé si tous les noeuds d'une branche non-fermée sont réduits.*

Pour montrer que la méthode est correcte, nous prouvons un premier lemme.

**Lemme 3** *Si  $\mathcal{M}$  est un modèle de  $\neg\varphi$  et  $T$  un tableau  $T$  de racine  $0, \alpha$  alors il existe un chemin  $P$  de  $T$  et une extension de  $\mathcal{M}$  en  $\mathcal{M}'$  une  $L_c$ -structure telle que pour tout  $1, \alpha$  (resp.  $0, \alpha$ ) sur  $P$ ,  $\mathcal{M}'$  est un modèle de  $\alpha$  (resp.  $\neg\alpha$ ).*

**Preuve** La preuve est par induction sur le nombre d'étapes de construction de  $T$  par la suite  $T_0, T_1, T_2 \dots$ . On suppose que le noeud ajouté prolonge le chemin  $P$  sur lequel la propriété est vraie.

- Cas de base.  $\mathcal{M}$  convient car  $T$  se réduit à  $0, \alpha$ .
- Induction. On considère  $N$  le noeud de  $P$  développé pour passer de  $T_{n-1}$  à  $T_n$ .

- $N$  de la forme  $\exists x \psi(x)$  et le noeud ajouté est  $psi(c)$  avec  $c$  une nouvelle constante. Par hypothèse il existe des termes  $t$  tels que  $\psi(t)$  est vrai et donc il suffit d'interpréter  $c$  (dont l'interprétation n'est pas encore fixée puisque  $c$  est nouveau) par un de ces termes.
- $N$  de la forme  $\forall x \psi(x)$ , et on rajoute un nouveau noeud  $\psi(t)$ . Si  $t$  ne contient que des symboles déjà utilisés aucun problème:  $\psi(t)$  est vrai car pour tout élément du domaine, l'interprétation de  $\psi$  est vraie. Sinon  $t$  contient de nouvelles constantes, qu'on interprètera comme des éléments arbitraires du modèle. L'interprétation de  $\psi$  étant vraie pour tout élément du modèle, restera vraie.
- les autres cas sont similaires à ceux-ci ou au cas propositionnel.

□

**Théorème 11**  $Si \vdash \varphi$  alors  $\models \varphi$ .

**Preuve** On raisonne par contraposition. On suppose que  $\not\models \varphi$ , donc il existe un modèle  $M$  de  $\neg\varphi$ . S'il existait un tableau contradictoire pour  $0, \varphi$ , alors une branche  $B$  et une extension de  $M$  en  $M_c$  telle que si  $i, \alpha$  est dans  $B$  alors  $\alpha$  vaut  $i$  dans  $B$ . Comme  $B$  est contradictoire, il existe une formule  $\beta$  qui vaut 0 et 1 en même temps dans ce modèle, contradiction. □

**Exercice:** : Prouver à l'aide de la méthode des tableaux sémantiques que le raisonnement suivant est correct:

$$((\forall x : (P(x) \Rightarrow \forall y (Q(y) \Rightarrow \neg R(x, y)))) \wedge (\forall x : (P(x) \Rightarrow \exists y (S(y) \wedge R(x, y)))) \wedge \exists x : P(x)) \Rightarrow \exists x : (S(x) \wedge \neg Q(x))$$

### 2.5.3 Complétude

**Proposition 10** *Pour toute formule close  $\varphi$  il existe un tableau terminé de racine  $0, \varphi$  et un tableau terminé de racine  $1, \varphi$*

**Preuve** On définit  $rang(\varphi)$  comme l'indice maximal d'un terme  $t_i$  de  $L_c$  apparaissant dans  $\varphi$ . Un tableau est  $n$ -terminé si chacun de ses noeuds d'une forme autre que  $1, \forall x\psi$  ou  $0, \exists x\psi$  qui est sur une branche non-contradictoire est réduit, et si toute branche non-contradictoire qui contient  $1, \forall\psi$  (resp.  $0, \exists x\psi$ ) contient aussi  $\{1, \psi\{x \leftarrow t_i\} \mid rang(\psi\{x \leftarrow t_i\}) \leq n\}$  (resp.  $\{0, \psi\{x \leftarrow t_i\} \mid rang(\psi\{x \leftarrow t_i\}) \leq n\}$ ). Un tableau  $n$ -terminé se construit aisément à

partir d'un tableau  $n-1$ -terminé par extension des branches. Comme seul un nombre fini de noeud peuvent être ajoutés (rang borné par  $n$ ) la construction termine. L'arbre limite  $T_\infty$  est un tableau terminé par construction.  $\square$

**Théorème 12**  $Si \models \varphi$  alors  $\vdash \varphi$ .

**Preuve** Par contraposée: on suppose que  $\not\vdash \varphi$  et on va montrer qu'alors  $\not\models \varphi$ . Si  $\not\vdash \varphi$ , alors considérons une branche non contradictoire d'un tableau terminé pour  $0, \varphi$ , qui existe d'après la proposition précédente. Prenons  $M$  une interprétation de Herbrand sur  $L_c$  définie par pour tout  $i, p(t_1, \dots, t_n)$  apparaissant sur la branche, l'interprétation de  $p(t_1, \dots, t_n)$  est  $i$ . Une telle interprétation existe ( $B$  n'est pas contradictoire) et pour tout  $i, \alpha$  sur la branche, l'interprétation de  $\alpha$  dans  $M$  est  $i$ . La preuve est par induction structurelle sur  $\alpha$ :

- $\alpha$  un atome: c'est vrai par hypothèse.
- $\alpha$  est de la forme  $\neg\beta$  ou  $\alpha_1\{\wedge, \vee, \Rightarrow, \Leftrightarrow\}\alpha_2$ : la preuve est évidente.
- $\alpha = \forall x \beta(x)$  et  $1, \alpha$  est dans  $B$ . Alors tous les  $1, \beta(t_i)$  sont sur  $B$  valent 1 dans l'interprétation par hypothèse de récurrence et donc  $\alpha$  vaut 1 puisque le domaine de l'interprétation est l'ensemble des termes clos de  $L_c$ . Le cas  $\alpha = \exists x \beta(x)$  et  $0, \alpha$  sur  $B$  est analogue.
- $\alpha = \exists x \beta(x)$  et  $1, \alpha$  est dans  $B$ . Alors  $1, \beta(c)$  est sur  $B$ , donc  $\beta(c)$  vaut 1 et donc  $\alpha$  vaut 1 également.

Une fois ceci prouvé, le résultat est immédiat: il existe un modèle de  $\neg\varphi$  puisque  $0, \varphi$  est sur  $B$ .  $\square$

**Exercice:** : Peut-on prouver avec la méthode des tableaux sémantiques que:  $\vdash \forall x, y, (P(x, y) \vee Q(x, y)) \Rightarrow \forall x, y, P(x, y) \vee \forall x, y, Q(x, y)$  (Si la réponse est oui, donner une démonstration, si elle est non, donner un argument qui la justifie).

## 2.6 Amélioration de la méthode des tableaux

Le problème en cp1 est est qu'il peut être nécessaire d'appliquer une règle d'inférence plusieurs fois à la même formule. C'est la grande différence avec

le cp0: le cp1 n'est pas décidable, mais seulement semi-décidable donc si une formule est un théorème, il est possible d'en trouver une preuve sinon la procédure de recherche de preuve peut continuer indéfiniment. Dans ce qui suit on montre comment ajouter des hypothèses et ensuite comment utiliser l'unification de termes pour maîtriser les instantiations de variables existentielles ou universelles.

### 2.6.1 Traitement de la conséquence logique

Le traitement de la conséquence logique est similaire à celui réalisé pour le cas propositionnel: on ajoute une règle de  $\Gamma$ -introduction qui autorise à rajouter  $1, \gamma$  à une branche de l'arbre si  $\gamma \in \Gamma$  où  $\Gamma$  est un ensemble de formules closes. Le résultat fondamental est le suivant.

**Théorème 13**  $\Gamma \vdash \varphi$  ssi  $\Gamma \models \varphi$

### 2.6.2 Utilisation de l'unification

La règle sur les formules universelles est trop permissive, puisqu'elle autorise n'importe quelle instance et que seul un choix judicieux de l'instance permet de fermer le tableau en général.

La solution va exiger de modifier la méthode des tableaux de plusieurs façons:

- on autorise des variables libres dans les tableaux.
- on utilisera la skolémisation pendant la preuve.
- les règles d'instanciation sont:

– Formule universelle:  $\frac{1, \forall x \gamma(x)}{1, \gamma(X)}$  avec  $X$  variable libre qui n'est pas liée ailleurs dans le tableau. On a une règle similaire pour  $0, \exists x \gamma(x)$ .

– Formule existentielle:  $\frac{1, \exists x \gamma(x)}{1, \gamma(f(X_1, \dots, X_n))}$  avec  $f$  nouveau symbole de fonction et  $X_1, \dots, X_n$  toutes les variables libres qui apparaissent sur la branche.

- (règle de substitution) on pourra appliquer une substitution  $\sigma$  à un tableau  $T$  pour obtenir un tableau valide  $T\sigma$  si  $\sigma$  est libre pour toute formule dans  $T$ .

Comme précédemment un tableau est fermé ssi toutes ses branches le sont. Choisir la bonne substitution  $\sigma$  est équivalent à choisir la bonne instance, l'unification est l'outil adéquat.

**Proposition 11** (règle de fermeture atomique par p.g.u) Soit  $T$  un tableau pour un ensemble de formules closes  $S$ , soit  $B$  une branche qui contienne  $A_1$  et  $\neg A_2$  où  $A_1$  et  $A_2$  sont deux atomes unifiables avec  $\sigma$  un p.g.u., alors  $T\sigma$  est un tableau valide pour  $S$ .

Exemple: Une preuve de  $\forall x : P(x, f(x)) \Rightarrow \forall x : \exists y : P(x, y)$ . Dans un souci de clarté on notera les variables libres avec des majuscules:

$$\begin{array}{c}
0, (\forall x : p(x, f(x)) \Rightarrow \forall x : \exists y : p(x, y)) \\
| \\
1, \forall x : p(x, f(x)) \\
| \\
0, (\forall x : \exists y : p(x, y)) \\
| \\
0, (\exists y : p(a, y)) \\
| \\
0, p(a, Y) \\
| \\
1, p(X, f(X))
\end{array}$$

et on applique le p.g.u.  $\sigma = \{X \leftarrow a, Y \leftarrow f(a)\}$  qui permet de fermer le tableau.

On peut montrer la correction et la complétude de la méthode (même en se restreignant la substitution à la règle de fermeture atomique). La preuve est similaire à celle de la résolution et utilise un lemme de relèvement.

# Chapter 3

## Logique modale

### 3.1 Introduction

L'évolution de l'univers est difficile à représenter en logique classique. La proposition *il pleut* de la logique classique signifie:

- il a toujours plu et il pleuvra toujours
- il pleut pour tout le monde

Une solution directe pour relativiser les énoncés consisterait à introduire des prédicats d'ordre supérieur: par exemple *il pleut à l'instant  $T$*  pourrait s'écrire  $Valide(T, il\ pleut)$  et *Jean sait que John sait qu'il pleut* s'écrirait  $Sait(Jean, (Sait(John, il\ pleut)))$ . Cette méthode à l'inconvénient d'introduire trop radicalement la logique d'ordre supérieure, où l'unification est indécidable (même restreinte au second ordre, Goldfarb 82). La solution présentée dans ce chapitre est une extension plus prudente du langage classique.

La logique peut être rendue plus expressive par l'introduction de *modalités* traduisant des notions a priori non quantifiables: possibilité, obligation, croyance . . . . Les opérateurs modaux sont des opérateurs qui portent sur des formules logiques pour en modifier, en infléchir le sens. On peut considérer les opérateurs modaux comme de nouveaux quantificateurs. Leur sens dépendra du contexte.

Logique	Modes	$\Box A$	$\Diamond A$
aléthique	possibilité, nécessité	il est nécessaire que A	il est possible que A
déontique	permission, nécessité	A est obligatoire	A est permis
temporelle	parfois, toujours	A sera toujours vrai	A sera parfois vrai
épistémique	connaissance croyance	A est su A est connu	l'inverse de A n'est pas su l'inverse de A n'est pas connu
dynamique		toute execution du programme produit A	il existe une execution du programme qui produit A

## 3.2 Logique du possible

Un système logique utilisant les opérateurs *il est possible que* et *il est nécessaire que* s'appelle logique du possible ou logique *aléthique*. Le symbole  $\Box$  est associé à la modalité de nécessité;  $\Box P$  signifie: *il est nécessaire que P*. Dualement, le symbole  $\Diamond$  est associé à la modalité de possibilité;  $\Diamond P$  signifie: *il est possible que P*. Chacun de ces opérateurs peut s'exprimer à l'aide de l'autre:  $\Box P$  équivaut à  $\neg \Diamond \neg P$  (cf. les quantificateurs classiques). Par exemple, le schéma d'axiome suivant, appelé *K*:

$$K \quad \Box(P \Rightarrow Q) \Rightarrow (\Box P \Rightarrow \Box Q)$$

exprime que *s'il est nécessaire que P implique Q alors il est nécessaire que P implique qu'il est nécessaire que Q*. La logique modale étend la logique classique. Les règles de déduction classiques comme le modus ponens, sont encore disponibles (en général):

$$\text{modus ponens : } \frac{A \quad A \Rightarrow B}{B}$$

Ces règles seront complétées par la règle de nécessité:

$$\text{nécessité : } \frac{B}{\Box B}$$

**Exercice:** Traduire sous forme logique les énoncés suivants:

- Il est possible que Denis envoie le livre à Martine.
- Il n'était pas possible pour Pierre d'écrire à chacun.

Introduisons de nouvelles règles d'inférence dérivées permettant de raccourcir les preuves.

$$\text{R1 : } \left\} \frac{A \Rightarrow B}{\Box A \Rightarrow \Box B}$$

Preuve:

- 1  $A \Rightarrow B$  hypothèse
- 2  $\Box(A \Rightarrow B)$   $N(1)$
- 3  $\Box A \Rightarrow \Box B$   $MP(2, K)$

De la même manière, on peut vérifier:

$$\text{R2 : } \left\} \frac{A \Leftrightarrow B}{\Box A \Leftrightarrow \Box B}$$

$$\text{R3 : } \left\} \frac{A \Rightarrow B}{\Diamond A \Rightarrow \Diamond B}$$

Preuve:

- 1  $A \Rightarrow B$  hypothèse
- 2  $\neg B \Rightarrow \neg A$  *classique*
- 3  $\Box \neg B \Rightarrow \Box \neg A$   $R1$
- 4  $\neg \Box \neg A \Rightarrow \neg \Box \neg B$  *classique*
- 5  $\Diamond A \Rightarrow \Diamond B$  *def.*

### 3.3 Logique de la connaissance ou de la croyance

En logique épistémique, l'opérateur  $\Box$  (parfois noté  $L$ ) signifie *est cru* ou bien *est connu*. Son dual  $\Diamond$  (parfois noté  $M$ ) signifie *le contraire n'est pas cru* ou *le contraire n'est pas connu*. Voici quelques schémas d'axiomes qu'il peut être intéressant d'introduire pour préciser l'intention que l'on attribue aux modalités:

- **Axiome de la connaissance**

$$T \quad \Box p \Rightarrow p$$

Intention: une connaissance est par définition une information vraie. L'axiome  $T$  formalise la connaissance certaine par opposition à la croyance.

La logique modale fondée sur  $K$  et  $T$  s'appelle logique  $T$ .

- **Axiome de l'introspection positive**

$$4 \quad \Box p \Rightarrow \Box \Box p$$

Intention: si je sais (crois)  $p$  alors je sais (crois) que je sais (crois)  $p$ . La logique modale fondée sur  $K$ ,  $T$  et l'axiome 4 s'appelle logique  $\mathcal{S}4$ .

- **Axiome de l'introspection négative**

$$5 \quad \Diamond p \Rightarrow \Box \Diamond p$$

Ce schéma d'axiome est équivalent à:

$$\neg \Box p \Rightarrow \Box \neg \Box p$$

Intention: si je ne sais (crois) pas que  $p$  est vrai alors je sais (crois) que je ne sais (crois) pas que  $p$  est vrai.

La logique modale fondée sur  $K$ ,  $T$  et les axiomes 4 et 5 s'appelle logique  $\mathcal{S}5$ . Le système formé des axiomes  $K$ , 4 et 5 permet de décrire une situation où l'agent possède des croyances erronées mais admet une parfaite capacité d'introspection logique (logique  $\mathcal{S}5$  faible). Lorsque le système contient plusieurs agents, une modalité sera associée à chaque agent: *Paul croit ce que Pierre croit* sera noté  $[Pierre]p \Rightarrow [Paul]p$ , par exemple.

**Exercice:** Traduire sous forme logique les énoncés suivants:

- Pierre ne croit pas que Jean croit ce qu'il (Pierre) dit.
- Denis croit qu'il a donné un livre à Jean (2 interprétations).

### 3.3.1 Logiques $S5$ et $S4$

Dans ce paragraphe nous allons nous familiariser avec des systèmes déductifs simples associés aux logiques  $S5$  et  $S4$ . Ces systèmes sont présentés à la Hilbert. Dans une autre section, on verra des systèmes fondés sur la méthode des tableaux sémantiques.

#### Formules

L'ensemble des formules de  $S5$  est le plus petit ensemble  $E$  qui contient les formules du calcul propositionnel classiques et tel que si  $A$  est dans  $E$ , alors  $\Box A$  et  $\Diamond A$  sont dans  $E$ .

#### Axiomes de $S5$

$$\text{axiomes classiques} + \left\{ \begin{array}{l} K \quad \Box(A \Rightarrow B) \Rightarrow (\Box A \Rightarrow \Box B) \\ T \quad \Box A \Rightarrow A \\ 4 \quad \Box A \Rightarrow \Box \Box A \\ 5 \quad \Diamond A \Rightarrow \Box \Diamond A \end{array} \right.$$

#### Règles d'Inférence

$$\text{MP : } \frac{A \quad A \Rightarrow B}{B}$$

$$\text{N : } \frac{B}{\Box B}$$

**Théorème 14** Dans  $S5$ , on a  $\vdash \Box(A \wedge B) \Leftrightarrow (\Box A \wedge \Box B)$

Preuve:

- |   |  |                  |
|---|--|------------------|
| 1 | $A \wedge B \Rightarrow A$   | <i>classique</i> |
| 2 | $\Box(A \wedge B) \Rightarrow \Box A$  | $R1(1)$          |
| 3 | $\Box(A \wedge B) \Rightarrow \Box B$  | $R1$             |
| 4 | $\Box(A \wedge B) \Rightarrow \Box A \wedge \Box B$                                | <i>classique</i> |
|   | <i>Reciproque</i>  |                  |
| 5 | $A \Rightarrow (B \Rightarrow (A \wedge B))$                                       | <i>classique</i> |
| 6 | $\Box A \Rightarrow \Box(B \Rightarrow (A \wedge B))$                              | $R1(5)$          |
| 7 | $\Box(B \Rightarrow A \wedge B) \Rightarrow (\Box B \Rightarrow \Box(A \wedge B))$ | $K$              |
| 8 | $\Box A \Rightarrow (\Box B \Rightarrow \Box(A \wedge B))$                         | $MP(6, 7)$       |
| 9 | $(\Box A \wedge \Box B) \Rightarrow \Box(A \wedge B)$                              | <i>classique</i> |

Nous obtenons en corollaire:

**Théorème 15** Dans  $S5$ , on a  $\vdash \diamond(A \vee B) \Leftrightarrow (\diamond A \vee \diamond B)$

Nous allons montrer maintenant que les axiomes de  $S5$  ne sont pas indépendants les uns des autres. En fait, l'axiome 4 peut se déduire des autres.

**Théorème 16** Dans  $S5$ , on peut déduire des autres axiomes:  $\Box A \Rightarrow \Box \Box A$

Preuve:

- 1  $A \Rightarrow \diamond A$  *contraposition de T*
- 2  $\diamond A \Rightarrow \Box \diamond A$  *5*
- 3  $A \Rightarrow \Box \diamond A$  *MP(1, 2)*
- 4  $\diamond \Box A \Rightarrow \Box A$  *contraposition de 5*
- 5  $\Box \diamond \Box A \Rightarrow \Box \Box A$  *R1(4)*
- 6  $\Box A \Rightarrow \Box \diamond \Box A$  *substituer  $\Box A$  à  $A$  dans 3*
- 7  $\Box A \Rightarrow \Box \Box A$  *MP(5, 6)*

Considérons maintenant le système  $S4$  obtenu à partir de  $S5$  en ôtant l'axiome 5, les règles d'inférence étant les mêmes.

Axiomes de  $S4$

$$\text{axiomes classiques} + \begin{cases} K & \Box(A \Rightarrow B) \Rightarrow (\Box A \Rightarrow \Box B) \\ T & \Box A \Rightarrow A \\ 4 & \Box A \Rightarrow \Box \Box A \end{cases}$$

Nous appellerons **modalité**, une suite d'opérateurs modaux. Nous allons maintenant montrer un certain nombre de propositions qui permettront de réduire la taille des modalités. Toute modalité de  $S4$  peut être réduite à une modalité de longueur inférieure ou égale à 3. Il suffit de montrer que

$$\begin{aligned} \vdash \Box \Box A &\Leftrightarrow \Box A \\ \vdash \diamond A &\Leftrightarrow \diamond \diamond A \\ \vdash \Box \diamond A &\Leftrightarrow \Box \diamond \Box \diamond A \\ \vdash \diamond \Box \diamond \Box A &\Leftrightarrow \diamond \Box A \end{aligned}$$

En effet les deux premières règles permettent de ramener toute modalité à:  $(\Box \diamond)^n \Box$  ou  $(\Box \diamond)^n \diamond$  ou  $(\Box \diamond)^n$  ou  $(\Box \diamond)^n$ . Les deux autres permettent de

ramener  $(\Box\Diamond)^n$  et  $(\Diamond\Box)^n$  à  $\Box\Diamond$  et  $\Diamond\Box$  respectivement.

**Réduction 1**

- 1  $\Box\Box A \Rightarrow \Box A$   $T$
- 2  $\Box A \Rightarrow \Box\Box A$  4
- 3  $\Box\Box A \Leftrightarrow A$

**Réduction 2**

- 1  $\Box\Box\neg A \Leftrightarrow \Box\neg A$
- 2  $\neg\Diamond\Diamond A \Leftrightarrow \neg\Diamond A$
- 3  $\Diamond\Diamond A \Leftrightarrow \Diamond A$

**Réduction 3**

- 1  $\neg A \Rightarrow \neg\Box A$  *contrapose(T)*
- 2  $\neg A \Rightarrow \Diamond\neg A$
- 3  $A \Rightarrow \Diamond A$  *subst A par  $\neg A$*
- 4  $\Box A \Rightarrow \Box\Diamond A$   $R1(3)$
- 5  $\Box\Box\Diamond A \Rightarrow \Box\Diamond\Box\Diamond A$  *subst A par  $\Box\Diamond A$*
- 6  $\Box\Diamond\mathbf{A} \Rightarrow \Box\Diamond\Box\Diamond\mathbf{A}$
- 7  $\Box\Diamond A \Rightarrow \Diamond A$   $T$
- 8  $\Diamond\Box\Diamond A \Rightarrow \Diamond\Diamond A$   $R3(4)$
- 9  $\Diamond\Box\Diamond A \Rightarrow \Diamond A$
- 10  $\Box\Diamond\Box\Diamond\mathbf{A} \Rightarrow \Box\Diamond\mathbf{A}$   $R1(9)$
- 11  $\Box\Diamond\Box\Diamond\mathbf{A} \Leftrightarrow \Box\Diamond\mathbf{A}$  6 – 10

**Réduction 4**

- 1  $\Box\Diamond\Box\Diamond\neg A \Leftrightarrow \Box\Diamond\neg A$  *red.3*
- 2  $\neg\Diamond\Box\Diamond\Box A \Leftrightarrow \neg\Diamond\Box A$
- 3  $\Diamond\Box\Diamond\Box A \Leftrightarrow \Diamond\Box A$  *classique*

En conclusion, dans  $S4$  il existe sept modalités (positives) différentes: *vide*,  $\Box$ ,  $\Diamond$ ,  $\Box\Diamond$ ,  $\Diamond\Box$ ,  $\Box\Diamond\Box$ ,  $\Diamond\Box\Diamond$ .  
Cela reste bien sûr valable pour  $S5$ . Mais on peut même aller plus loin:

- 1  $\Diamond A \Rightarrow \Box\Diamond A$  5
- 2  $\Box\Diamond A \Rightarrow \Diamond A$   $T$
- 3  $\Box\Diamond\mathbf{A} \Leftrightarrow \Diamond\mathbf{A}$
- 4  $\Box\Diamond\neg A \Leftrightarrow \Diamond\neg A$
- 5  $\neg\Diamond\Box\mathbf{A} \Leftrightarrow \neg\Box\mathbf{A}$
- 6  $\Diamond\Box A \Leftrightarrow \Box A$

Enfin toute suite de modalités dans  $S5$  se ramène à une seule modalité.

**Exercice:** Montrer:  $S5 \vdash (\Box(A \Rightarrow \Diamond B) \wedge \Box(C \Rightarrow \Diamond D)) \Rightarrow ((A \vee C) \Rightarrow \Diamond(B \vee D))$

### 3.3.2 Un exemple

Nous allons étudier un exemple où plusieurs modalités interviennent, ces modalités étant chacune associée à un agent. Cet exemple montre que la logique modale se prête bien à la formalisation du raisonnement sur le raisonnement lui-même.

Le problème est classique:

*Un roi désire savoir lequel de ses trois conseillers est le plus sage. Il peint un point blanc sur le front de chacun et leur dit qu'il a peint un point noir ou blanc sur chacun et qu'au moins l'un des points est blanc. Il demande ensuite à chaque conseiller la couleur de son point. Le premier répond qu'il ne sait pas. Le deuxième, ensuite, répond qu'il ne sait pas non plus. Ayant entendu cela, le troisième répond que son point est blanc. Trouver son raisonnement.*

Pour simplifier, nous allons supposer qu'il n'y a que deux conseillers, A et B, B dit qu'il ne sait pas puis A dit qu'il a trouvé. Le problème sera formalisé par une logique multi-modale de connaissance. Nous introduisons un opérateur modal pour chaque agent  $i \in \{A, B\}$  et nous le notons  $\Box_i$ .

Axiomes

$$\text{axiomes classiques} + \begin{cases} K_i & \Box_i(p \Rightarrow q) \Rightarrow (\Box_i p \Rightarrow \Box_i q) \\ T_i & \Box_i p \Rightarrow p \end{cases}$$

Règles d'Inférence

$$\text{MP : } \frac{p \quad p \Rightarrow q}{q}$$

$$N_i : \frac{p}{\Box_i p}$$

Les axiomes  $K_i$  et la règle de *modus ponens* nous assurent que chaque agent connaît toutes les conséquences logiques de ses connaissances. Cela peut s'exprimer par la règle dérivée d'omniscience logique:

$$\text{OL : } \frac{\Box_i A \quad A \vdash_{\text{classiquement}} B}{\Box_i B}$$

Détaillons le raisonnement de A, le vainqueur de l'épreuve. Nous noterons  $b(x)$  la phrase *x a un point blanc*.

A et B savent que chacun peut voir le point de l'autre. Si A n'a pas de point blanc, B le sait.

$$\neg b(A) \Rightarrow \Box_B \neg b(A)$$

B sait qu'au moins l'un d'eux a un point blanc. Donc:

$$\Box_B (\neg b(A) \Rightarrow b(B))$$

B déclare qu'il ne sait pas s'il a un point blanc donc A le sait:

$$\Box_A (\neg \Box_B (b(B)))$$

- |   |  |  |
|---|--|--|
| 1 | $\neg \mathbf{b}(\mathbf{A}) \Rightarrow \Box_{\mathbf{B}}(\neg \mathbf{b}(\mathbf{A}))$ | <i>hyp</i>                               |
| 2 | $\Box_{\mathbf{B}}(\neg \mathbf{b}(\mathbf{A}) \Rightarrow \mathbf{b}(\mathbf{B}))$      | <i>hyp</i>                               |
| 3 | $\Box_B \neg b(A) \Rightarrow \Box_B b(B)$   | <i>R1(2) ou MP(<math>K_B, 2</math>)</i>  |
| 4 | $\neg b(A) \Rightarrow \Box_B b(B)$  | <i>MP(1, 3)</i>                          |
| 5 | $\Box_A (\neg \Box_B b(B) \Rightarrow b(A))$   | <i><math>N_A(\text{contr}(4))</math></i> |
| 6 | $\Box_A (\neg \Box_B b(B)) \Rightarrow \Box_A b(A)$                                      | <i>R1(5) ou MP(<math>K_A, 5</math>)</i>  |
| 7 | $\Box_{\mathbf{A}}(\neg \Box_{\mathbf{B}}(\mathbf{b}(\mathbf{B})))$                      | <i>hyp.</i>                              |
| 8 | $\Box_A b(A)$  | <i>MP(6, 7)</i>                          |

### 3.4 Sémantique de la logique modale: modèles de Kripke

L'interprétation du calcul propositionnel classique comme composition de fonctions booléennes élémentaires ne s'applique pas à la logique modale. En effet la composition d'un opérateur de nécessité avec une proposition vraie

peut conduire, suivant les cas, soit à un énoncé vrai, soit à un énoncé faux. Par exemple, *la neige est blanche* est un énoncé vrai, mais *nécessairement, la neige est blanche* n'est pas vrai. Pourtant, *nécessairement, la neige est blanche ou la neige n'est pas blanche* est vrai. La différence entre les deux énoncés auxquels s'applique l'opérateur de nécessité est que l'un est vrai à cause des circonstances, i.e des lois de la physique, alors que l'autre est vrai en vertu des lois de la logique. Il faut donc distinguer entre les vérités de fait et les vérités logiques. Le point de vue généralement adopté, est de considérer un énoncé comme nécessairement vrai si sa vérité est indépendante des circonstances, s'il est vérifié dans tous les *mondes possibles*: c'est la sémantique de Kripke. Chaque monde possible est un ensemble de connaissances. Il existe par ailleurs une *relation d'accessibilité* sur l'ensemble des mondes possibles; cette relation définit les transitions d'un monde à un autre. La sémantique des logiques modales dépend donc de deux éléments: l'ensemble des mondes possibles et la relation d'accessibilité. Des propriétés de cette relation vont dépendre les notions de validité et de satisfaisabilité.

En fait, la relation d'accessibilité sera liée à la forme des axiomes propres à chaque logique modale. Considérons par exemple le schéma d'axiome  $\Box A \Rightarrow A$ . Il sera vrai dans toute structure où la vérité d'un énoncé  $A$  dans tous les mondes accessibles à partir du monde actuel implique la vérité de  $A$  dans le monde actuel. Il est donc nécessaire que le monde actuel soit accessible à lui-même. Autrement dit la relation d'accessibilité est *réflexive*.

La sémantique de la logique modale sera définie en terme de modèles de Kripke. Nous nous restreindrons au cas propositionnel. On notera  $P$  l'ensemble des symboles propositionnels.

**Definition 24** *Un modèle de Kripke  $M$  est un triplet  $(W, R, I)$  où:*

- $W$  est l'ensemble des mondes possibles.
- $R$  est une relation binaire sur  $W$ , appelée relation d'accessibilité.
- $I : P \rightarrow 2^W$  est une fonction qui à chaque variable propositionnelle lui associe l'ensemble des mondes où elle est vraie.

Remarquons qu'il est équivalent de se donner à la place de  $I$ , une valuation  $v : W \times P \rightarrow \{\text{vrai}, \text{faux}\}$ .

Introduisons maintenant la relation de satisfaction dont la définition fait intervenir non seulement une formule et un modèle, mais aussi un monde:

Si  $A$  désigne une formule,  $M$  un modèle de Kripke et  $w$  un monde de  $M$ , la relation:

Le monde  $w$  de  $M$  satisfait  $A$

sera notée

$$M, w \models A$$

Cette relation est définie par induction sur la structure des formules:

- $M, w \models A$  si  $A$  est une variable propositionnelle et  $w \in I(A)$
- $M, w \models A \wedge B$  si  $M, w \models A$  et  $M, w \models B$
- $M, w \models A \vee B$  si  $M, w \models A$  ou  $M, w \models B$
- $M, w \models \neg A$  si  $M, w \models A$  est faux
- $M, w \models \Box A$  si pour tout  $t$  tel que  $wRt$  on a  $M, t \models A$

Nous pouvons en déduire la règle sémantique suivante:

- $M, w \models \Diamond A$  si il existe un  $t$  tel que  $wRt$  et  $M, t \models A$

**Exemple 5** Nous avons:

- $M, a \models \Diamond \Box A$  ssi  $\exists b(aRb \text{ et } \forall c(bRc \Rightarrow M, c \models A))$
- $M, a \models \Box \Diamond A$  ssi  $\forall b(aRb \Rightarrow \exists c(bRc \text{ et } M, c \models A))$ .

Une formule  $A$  est *satisfaisable* s'il existe un modèle  $M$  et un monde  $w$  de  $M$  tel que  $M, w \models A$ . Une formule  $A$  est valide dans un modèle  $M$  si pour tout monde  $w$  de  $M$  on a  $M, w \models A$ . Une formule  $A$  est valide dans une structure  $(W, R)$  si, pour toute interprétation (ou valuation)  $I$ ,  $A$  est valide dans le modèle  $(W, R, I)$ . Si  $A$  est valide dans toute structure  $(W, R)$ , on dira simplement que  $A$  est valide.

**Exemple 6** Vérifions si  $\Box p$  est satisfaisable dans la structure définie par la Figure 3.4. L'énoncé  $\Box p$  est vrai en  $w_1$ , car tout monde accessible de  $w_1$  contient  $p$ . De même  $\Box p$  est vrai en  $w_5$ , mais  $\Diamond \neg p$  est insatisfaisable alors que  $\neg p$  est satisfaisable (en  $w_5$ ).

**Exercice:** Montrer que les formules suivantes sont valides:

$$\Box(A \Rightarrow B) \Rightarrow (\Box A \Rightarrow \Box B)$$

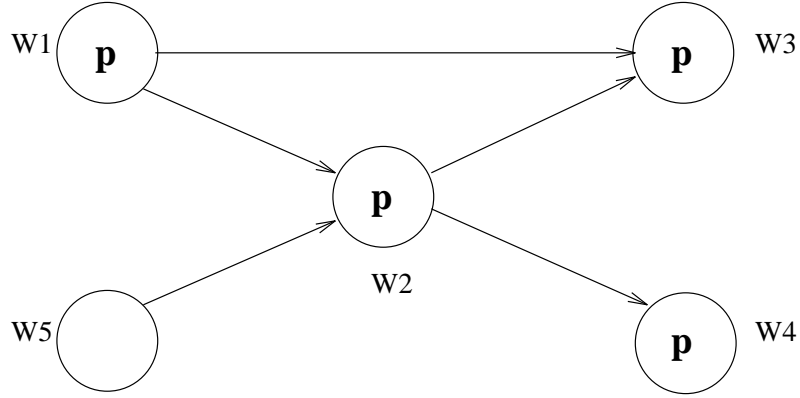


Figure 3.1: Modèle de Kripke

$$\begin{aligned} \Box(A \Rightarrow B) &\Rightarrow (\Diamond A \Rightarrow \Diamond B) \\ \Box(A \wedge B) &\Leftrightarrow (\Box A \wedge \Box B) \\ \Box(A \vee B) &\Leftrightarrow (\Box A \vee \Box B) \end{aligned}$$

**Exercice:** Dénombrer les modèles de Kripke que l'on peut construire sur un ensemble à deux éléments.

**Exercice:** Construire une valuation  $I$  sur la structure de Kripke donnée de manière à satisfaire les propriétés indiquées:

$$\begin{aligned} \text{i) } (N, <) \\ 0 &\models P, \neg\Box P, \Diamond P, \Box\Diamond P, \Box\Diamond\neg P \\ 1 &\models \neg P, \neg\Box P, \Diamond P, \Box\Diamond P, \Box\Diamond\neg P \end{aligned}$$

$$\begin{aligned} \text{ii) } (R, <) \\ 1 &\models P, \Diamond P, \Diamond\neg P, \Box\Diamond P, \Box\Diamond\neg P \\ \sqrt{2} &\models \neg P, \Diamond P, \Diamond\neg P, \Box\Diamond P, \Box\Diamond\neg P \end{aligned}$$

Solution: i) prendre  $I(p) = \{x; x \text{ pair}\}$ ; ii) prendre  $I(p) = Q$

Il existe de nombreux schémas d'axiomes intéressants qui **correspondent** à des propriétés particulières de la relation d'accessibilité  $R$ . Cette correspondance est précisée dans le résultat suivant:

**Théorème 17** Dans le tableau ci-dessous, un axiome est valide dans  $(W, R)$ , si et seulement si  $R$  a la propriété correspondante:

$T$	$\Box A \Rightarrow A$	<i>réflexive</i>	$\forall x xRx$
$B$	$A \Rightarrow \Box \Diamond A$	<i>symétrique</i>	$\forall x, y xRy \Rightarrow yRx$
$D$	$\Box A \Rightarrow \Diamond A$	<i>totale</i>	$\forall x \exists y xRy$
4	$\Box A \Rightarrow \Box \Box A$	<i>transitive</i>	$\forall x, y, z xRy \wedge yRz \Rightarrow xRz$
5	$\Diamond A \Rightarrow \Box \Diamond A$	<i>euclidienne</i>	$\forall x, y, z xRy \wedge xRz \Rightarrow yRz$

Preuve: Montrons la ligne  $T$ :

$\Rightarrow$ : Si  $\Box A \Rightarrow A$  est vrai dans  $(W, R)$  alors choisissons un  $w \in W$  quelconque.

En particulier la propriété est vraie pour  $I(A) = \{v \in W | wRv\}$ . Donc  $(W, R, I), w \models \Box A$  et par hypothèse cela entraîne  $(W, R, I), w \models A$ . D'après le choix de  $I$  cela prouve que  $wRw$ .

$\Leftarrow$ : si  $R$  est réflexive alors la validité dans tous les mondes accessibles entraîne la validité dans le monde actuel.

Introduisons les schémas d'axiomes suivant:

- $L$  :  $\Box(\Box\phi \Rightarrow \phi) \Rightarrow \Box\phi$
- $G$  :  $\Diamond\Box\phi \Rightarrow \Box\Diamond\phi$
- $P$  :  $\phi \Rightarrow \Box\phi$
- $Q$  :  $\Diamond\phi \Rightarrow \Box\phi$
- $R$  :  $\Box\Box\phi \Rightarrow \Box\phi$

Rappelons quelques propriétés d'une relation binaire. La relation  $R$  est

- **bien fondée** s'il n'existe pas de suite infinie  $a_0Ra_1Ra_2R\dots$
- **confluente** si  $aRb$  et  $aRc$  implique qu'il existe  $d$  avec  $bRd$  et  $cRd$
- **pathétique** si  $xRy$  implique  $x = y$
- **partiellement fonctionnelle** si  $aRx$  et  $aRy$  implique  $x = y$
- **faiblement dense** si  $aRb$  implique qu'il existe  $c$  avec  $aRcRb$

Nous pouvons maintenant étendre le théorème précédent:

**Théorème 18** *Dans le tableau ci-dessous, un axiome est valide dans  $(W, R)$ , si et seulement si  $R$  a les propriétés correspondantes:*

- L* transitive et bien fondée
- G* confluente
- P* pathétique
- Q* partiellement fonctionnelle
- R* faiblement dense

## 3.5 Systèmes de preuve pour les logiques modales

### 3.5.1 Systèmes de Hilbert

Les systèmes formels modaux sont obtenus en étendant le calcul propositionnel classique:

- Les formules classiques sont des formules modales. Si  $A$  est une formule modale alors il en est de même pour  $\Box A$  et  $\Diamond A$ .
- Les règles de déduction sont celles de la logique classique plus la règle de nécessité
- Les axiomes sont les axiomes classiques plus d'autres suivant la logique modale concernée:

logique	axiomes
normale	K
T	KT
S4	KT4
S5	KT45
S5 faible	K45

Rappelons que pour modéliser la connaissance d'un agent ayant une parfaite capacité d'introspection, on choisira le système S5. S'il s'agit de croyance, on utilisera S5 faible. On peut dès lors montrer des théorèmes d'adéquation:

**Théorème 19** *Une formule est un théorème de la logique normale ssi elle est valide. Une formule est un théorème de la logique T (resp. S4, resp. S5) ssi A est valide dans toute structure de Kripke où R est réflexive (resp. réflexive et transitive, resp. une relation d'équivalence).*

Remarquons qu'il faut être très prudent avant de formuler un théorème de la déduction, du genre *si de A on peut déduire B alors  $A \Rightarrow B$  est prouvable*. En effet  $A \vdash B$  signifie:

$$\forall M, (\text{si } (\forall w M, w \models A) \text{ alors } (\forall w M, w \models B))$$

Par contre  $A \Rightarrow B$  est un théorème signifie:

$$\forall M, \forall w, (\text{si } M, w \models A \text{ alors } M, w \models B)$$

Par exemple  $A \vdash \Box A$  mais on n'a pas en général  $\vdash A \Rightarrow \Box A$ .

### 3.5.2 Tableaux sémantiques

Un tableau est un arbre fini étiqueté. Chaque nœud est étiqueté par une formule  $1, v \models A$  ou  $0, v \models A$  dont le sens est respectivement *dans le monde  $v$ ,  $A$  est vrai* ou *dans le monde  $v$ ,  $A$  est faux*. L'idée de la méthode des tableaux pour vérifier la validité de  $A$  est de supposer le contraire et de rechercher un contre-exemple à cette hypothèse en développant un tableau à partir de la racine  $0, w \models A$ , où  $w$  est un monde constant qui n'apparaît pas dans  $A$ . Le tableau est développé en considérant toutes les manières de falsifier  $w \models A$ . Si, à un certain stade, une contradiction immédiate apparaît sur chaque branche du tableau, la preuve est obtenue. Une branche fermée est une branche avec une contradiction à l'extrémité.

$\wedge$	$1, w \models \phi \wedge \psi$ $\quad  $ $1, w \models \phi$ $\quad  $ $1, w \models \psi$	$0, w \models \phi \wedge \psi$ $\quad  $ $0, w \models \phi$ $0, w \models \psi$ $\quad \wedge$
$\vee$	$1, w \models \phi \vee \psi$ $\quad  $ $1, w \models \phi$ $1, w \models \psi$	$0, w \models \phi \vee \psi$ $\quad  $ $0, w \models \phi$ $\quad  $ $0, w \models \psi$
$\Rightarrow$	$1, w \models \phi \Rightarrow \psi$ $\quad  $ $0, w \models \phi$ $1, w \models \psi$	$0, w \models \phi \Rightarrow \psi$ $\quad  $ $1, w \models \phi$ $\quad  $ $0, w \models \psi$
$\neg$	$1, w \models \neg \phi$ $\quad  $ $0, w \models \phi$	$0, w \models \neg \phi$ $\quad  $ $1, w \models \phi$
$\Box$	$1, w \models \Box \phi$ $\quad  $ $1, v \models \phi$	$0, w \models \Box \phi$ $\quad  $ $0, v \models \phi$ $\quad  $ $wRv$
$\Diamond$	$1, w \models \Diamond \phi$ $\quad  $ $1, v \models \phi$ $\quad  $ $wRv$	$0, w \models \Diamond \phi$ $\quad  $ $0, v \models \phi$
	$C_1$	$C_2$
	$C_2$	$C_1$

Condition  $C_1$ :  $wRv$  est un nœud ancêtre  
Condition  $C_2$ :  $v$  est un nouveau symbole de monde

Figure 3.26 Règles 1-12

### Exemple 7

$0, w$	$\models$	$\Box(A \Rightarrow B) \Rightarrow (\Box A \Rightarrow \Box B)$
$1, w$	$\models$	$\Box(A \Rightarrow B)$
$0, w$	$\models$	$\Box A \Rightarrow \Box B$
$1, w$	$\models$	$\Box A$
$0, w$	$\models$	$\Box B$
	$wRv$	$v$ est nouveau
$0, v$	$\models$	$B$
$1, v$	$\models$	$A$
$1, v$	$\models$	$A \Rightarrow B$
	$\wedge$	
$0, v \models A$		$1, v \models B$

**Théorème 20 (Correction)** *Toute proposition ayant une preuve par les règles 1-12 est valide dans la logique normale.*

**Théorème 21 (Adéquation)** *Toute proposition valide dans la logique normale admet une preuve par la méthode des tableaux.*

## 3.6 Logique temporelle linéaire

### 3.6.1 Langage

Les opérateurs sont les suivants:

1.  $\bigcirc$  se lit "à l'instant suivant"
2.  $\Box$  se lit "toujours"

3.  $\diamond$  se lit "inévitablement" ou "au moins une fois".

4.  $\mathcal{U}$  se lit "jusqu'à"

Les formules sont :i) Les variables propositionnelles,  $V, F$

ii)  $A \vee B, A \wedge B, \neg A, \diamond A, \square A, \circ A, A\mathcal{U}B$  où  $A, B$  sont des formules.

### 3.6.2 Sémantique

Nous interpréterons les formules construites avec ces opérateurs en considérant la structure de Kripke formée par les entiers naturels  $\mathbb{N}$  munis de la relation d'ordre habituelle et représentant la suite des instants futurs. Un modèle de Kripke pour cette logique est donc donné par une application:

$$\pi : P \mapsto 2^{\mathbb{N}}$$

où  $P$  est l'ensemble des variables propositionnelles. Définissons la sémantique des formules temporelles dans le modèle  $S$  pour le monde (= instant )  $k$ :

- $S, k \models p$  ssi  $k \in \pi(p)$  pour  $p$  variable propositionnelle.
- $S, k \models \circ A$  ssi  $S, k + 1 \models A$ .
- $S, k \models \square A$  ssi  $\forall i \geq k, S, i \models A$
- $S, k \models \diamond A$  ssi  $\exists i \geq k, S, i \models A$
- $S, k \models A\mathcal{U}B$  ssi  $\exists j \geq k, S, j \models B$  and  $\forall i$  tel que  $k \leq i < j, S, i \models A$ .

### 3.6.3 Système formel

La logique PTL admet l'axiomatisation suivante:

#### Axiomes

A0. les tautologies du calcul prop. classique

A1.  $\square(A \Rightarrow B) \Rightarrow (\square A \Rightarrow \square B)$

A2.  $\circ(\neg A) \Leftrightarrow \neg(\circ A)$

$$A3. \circ(A \Rightarrow B) \Rightarrow (\circ A \Rightarrow \circ B)$$

$$A4. \Box A \Rightarrow A \wedge \circ \Box A$$

$$A5. \Box(A \Rightarrow \circ A) \Rightarrow (A \Rightarrow \Box A)$$

$$A6. \Diamond A \Leftrightarrow \neg \Box \neg A$$

$$U1. AUB \Rightarrow \Diamond B$$

$$U2. AUB \Leftrightarrow (B \vee (A \wedge \circ(AUB)))$$

## Règles d'Inférence

$$MP : \frac{A \quad A \Rightarrow B}{B} \quad N1 : \frac{B}{\Box B} \quad N2 : \frac{B}{\circ B}$$

**Commentaires.** La logique PTL est une extension de  $K$  par  $A_0, A_1, A_3$  (et les règles d'inférence). D'autres opérateurs exprimables avec  $U$ :

Until faible:  $AU_f B \Leftrightarrow \Box A \vee (AUB)$

Précède:  $APr B \Leftrightarrow \neg BU_f A$

$A_2$ : tout état n'a qu'un seul suivant.

$A_4$ : définition de  $\Box$  par induction.

$A_5$ : principe d'induction

$U_2$ : définition de  $U$  par induction.

**Exemple de théorème.**  $\Box(B \Rightarrow C) \Rightarrow (AUB \Rightarrow AUC)$

**Théorème 22** *Le système formel de PTL est correct, complet et décidable. Autrement dit,  $\models A$  ssi  $A$  est démontrable et il existe une procédure de décision pour déterminer si une formule est valide.*

### 3.6.4 Applications

La logique temporelle permet d'étudier les programmes concurrents. Considérer par exemple un ensemble de  $n$  processus  $D_i$  pouvant accéder à une ressource commune  $R$  et un processus  $G$  chargé de gérer cette ressource. La variable  $d_i$  est affectée de la valeur  $V$  (resp.  $F$ ) par  $D_i$  lorsque  $D_i$  demande (resp. relache)  $R$ . La variable  $a_i$  est affectée de  $V$  par  $G$  si  $R$  a été accordée à  $D_i$  et ce dernier ne l'a pas rendue.

### Propriétés d'invariance

Exprimées à l'aide de  $\square$  uniquement.

la ressource est accordée à au plus un processus demandeur à la fois.

$$\square \bigwedge_{i \neq j} \neg(a_i \wedge a_j)$$

### Propriétés de vivacité

Exprimées à l'aide de  $\square$  et  $\diamond$ .

si le processus  $D_i$  demande la ressource il l'obtiendra

$$\square(d_i \Rightarrow \diamond a_i)$$

si le processus  $D_i$  obtient la ressource il la relachera

$$\square(a_i \Rightarrow \diamond \neg d_i)$$

### Propriétés de précedence

Exprimées à l'aide de  $U$  ou d'opérateurs dérivés.

si  $D_i$  n'a pas la ressource maintenant, alors avant qu'il l'obtienne, il doit en faire la demande.

$$\square(\neg a_i \Rightarrow (d_i Pr \ a_i))$$

### 3.6.5 Expressivité

Soit  $p$  une variable propositionnelle. On considère un langage construit uniquement sur  $p$ . Un modèle de Kripke temporel pour le langage associé est décrite par une suite infinie d'éléments de  $\{p, \neg p\}$ .

**Proposition 12** *Soit  $n$  un entier naturel. Toute formule  $F$  ayant moins de  $n$  occurrences de  $\circ$  s'évalue de la même manière sur  $p^i(\neg p)p^\omega$  quel que soit  $i > n$ .*

Preuve: Induction sur la structure de  $F$ . On note  $F_i$  la valeur de  $F$  sur  $p^i(\neg p)p^\omega$  à l'instant 0. Hypothèse de récurrence: pour tout  $i, n$  tels que  $i > n$ ,  $F_i$  est indépendant de  $i$ .

$p$  pour tout  $i > n$ ,  $F_i$  est vrai car  $0 \models p$ .

$G \wedge H$  hyp. d'induction.

$\neg H$  hyp. d'induction.

$\Box H$   $(\Box H)_i = H_i \wedge H_{i-1} \cdots \wedge H_{n+1} \wedge (\Box H)_n$  par définition de  $\Box$ .  $H_i = H_{i-1} = \cdots = H_{n+1}$  par hypothèse d'induction.  $H_{n+1} \wedge (\Box H)_n = (\Box H)_{n+1}$  par définition. Donc  $(\Box H)_i = (\Box H)_{n+1}$ .

$\circ H$   $(\circ H)_i = H_{i-1}$ . Par hypothèse d'induction, cette expression est indépendante de  $i$  (car on a  $i > n$  donc  $i - 1 > n - 1$ ).

$GUH$   $(GUH)_i = H_i \vee (G_i \wedge (H_{i-1} \vee G_{i-1} \wedge \cdots \wedge (H_{n+1} \vee (G_{n+1} \wedge (GUH)_n))))$ .  
 $(GUH)_i = (H_{n+1} \vee (G_{n+1} \wedge (GUH)_n))$ . par hypothèse d'induction.

Donc le résultat est indépendant de  $i$ .

On en déduit le résultat suivant du à Pierre Wolper:

**Proposition 13** *Il n'existe pas de formule  $F$  dont les modèles soient exactement tous ceux dans lesquels  $p$  est vraie aux instants pairs.*

Preuve: Par l'absurde soit  $F$  une telle formule et  $l$  le nombre de ses opérateurs  $\circ$ . D'après la proposition,  $F$  a la même valeur sur  $p^{2k}(\neg p)p^\omega$  et sur  $p^{2k-1}(\neg p)p^\omega$  dès que  $2k - 1 > l$ . La première suite vérifie la propriété mais pas la seconde.

**Remarque 2** *On aurait envie d'écrire que les modèles satisfont la formule:  $pair(p) \Rightarrow \circ \circ pair(p)$ .*

Une manière d'étendre l'expressivité de PTL pour capturer les propriétés du type ci-dessus est de définir de nouveaux opérateurs à partir d'une grammaire linéaire droite. Soit  $V_N$  des non terminaux,  $V_T = \{t_1, t_2, \dots, t_n\}$  des terminaux, et un ensemble  $P$  de règles de productions  $V \rightarrow mV'$  où  $m$  est

un mot de  $V_T^*$ . A chaque non-terminal  $V$  on associe un opérateur  $n$ -aire  $G_V$ , où  $n$  est le cardinal de  $V_T$ . La sémantique de l'opérateur  $G_V$  est la suivante: Pour tout modèle  $S$  et tout instant  $k$

$S, k \models G_V(F^1, \dots, F^n)$  ssi il existe un mot  $a_1 a_2 a_3 \dots$  fini ou infini issu de  $V$  engendré par la grammaire tel que pour tout  $i$ , si  $a_i$  est la lettre  $t_j$  alors  $S, k + i \models F_j$ .

**Exemple 8** Soit la grammaire  $V \rightarrow t_1 t_2 V$

$S, 0 \models G_0(F_1, F_2)$  ssi  $S, 0$  satisfait  $F_1, F_2$  alternativement.

$S, 0 \models G_0(p, \text{vrai})$  ssi  $S, 0$  satisfait  $p$  une fois sur deux. Nous avons donc résolu le problème.

**Exemple 9** Soit la grammaire  $V \rightarrow t_2$   
 $V \rightarrow t_1 V$

Alors  $G_0$  définit l'opérateur *jusqu'à* faible.

L'opérateur  $\square$  est défini par la grammaire  $V \rightarrow v_1 V$ . L'opérateur  $\circ$  est défini par la grammaire  $V \rightarrow v_1 v_2$  et par  $\circ A \Leftrightarrow G(\text{vrai}, A)$ .

## 3.7 Logique dynamique

La logique dynamique traite les programmes comme des modalités qui affecte la valeur des formules. Ainsi  $\langle p \rangle \phi$  signifie qu'il est possible d'exécuter  $p$  et de s'arrêter dans un état satisfaisant  $\phi$ .

### 3.7.1 Langage

Il y a deux sortes d'expressions: les programmes:  $p, q, r, \dots$  et les formules:  $\phi, \psi, \dots$ . Si  $\phi, \psi$  sont des formules et  $p, q$  des programmes alors:  $\phi \vee \psi, \neg \phi, \langle p \rangle \phi$  sont des formules et  $p; q, p \cup q, p^*, \phi?$  sont des programmes. La signification intuitive de ces constructions est:

$p; q$  exécuter  $p$  puis  $q$

$p \cup q$  choisir de manière non déterministe  $p$  ou  $q$  et l'exécuter

$p^*$  exécuter  $p$  un nombre de fois choisi de manière non déterministe

$\phi?$  tester  $\phi$ ; continuer si vrai, échouer sinon.

On peut utiliser ces primitives pour construire des instructions plus communes. Par exemple:

**if**  $\phi$  **then**  $p$  **else**  $q$  **fi**     $\phi?; p \cup \neg \phi?; q$

**while**  $\phi$  **do**  $p$  **od**     $(\phi?; p)^*; \neg \phi?$

Autres constructions:

$[p]\phi$  quand  $p$  termine, c'est dans un état vérifiant  $\psi$   
 $\{\phi\}p\{\psi\} \quad \phi \Rightarrow [p]\psi$

### 3.7.2 Sémantique

Un modèle de Kripke est formé d'un ensemble d'états (ou mondes) et d'une fonction d'interprétation  $I$ . Comme précédemment,  $I$  interprète une formule comme l'ensemble des mondes où elle est vérifiée:  $I(\phi) \subseteq S$ . Par contre  $I$  interprète les programmes comme des relations binaire sur  $S$ :  $I(p)$  est la relation input/output de  $p$ .

$$I(\langle p \rangle \phi) = \{u | \exists v(u, v) \in I(p) \text{ and } v \in I(\phi)\} \quad (3.1)$$

$$I(p; q) = \{(u, v) | \exists w(u, w) \in I(p) \text{ and } (w, v) \in I(q)\} \quad (3.2)$$

$$I(p \cup q) = I(p) \cup I(q) \quad (3.3)$$

$$I(p^*) = \text{la cloture réflexive transitive de } I(p) \quad (3.4)$$

$$I(\phi?) = \{(u, u) | u \in I(\phi)\} \quad (3.5)$$

**Exemple 10** On définit un modèle  $M$  par:  $S = \{u, v, w\}$ ,  $I(\phi) = \{u, v\}$ ,  $I(p) = \{(u, v), (u, w), (v, w), (w, v)\}$ . On peut vérifier que  $M, u \models \langle p \rangle \neg\phi \wedge \langle p \rangle \phi$  puis  $M, v \models [p]\neg\phi$ ,  $M, w \models [p]\phi$  et finalement:

$$M \models \langle p^* \rangle [(p; p)^*]\phi \wedge \langle p^* \rangle [(p; p)^*]\neg\phi$$

**Exemple 11** On peut montrer des équivalences de programmes du type:

$$[\text{ while } (p \vee q) \text{ do } i \text{ od } ]r \Leftrightarrow [(\text{ while } p \text{ do } i \text{ od } ); (\text{ while } q \text{ do } i; (\text{ while } p \text{ do } i \text{ od } ) \text{ od } )]r$$

### 3.7.3 Système formel

#### Axiomes de PDL

- 1 axiomes de la logique propositionnelle
- 2  $\langle p \rangle \phi \wedge [p]\psi \Rightarrow \langle p \rangle (\phi \wedge \psi)$
- 3  $\langle p \rangle (\phi \vee \psi) \Leftrightarrow \langle p \rangle \phi \vee \langle p \rangle \psi$
- 4  $\langle p \cup q \rangle \phi \Leftrightarrow \langle p \rangle \phi \vee \langle q \rangle \phi$
- 5  $\langle p; q \rangle \phi \Leftrightarrow \langle p \rangle \langle q \rangle \phi$
- 6  $\langle \psi? \rangle \phi \Leftrightarrow \psi \wedge \phi$
- 7  $(\phi \vee \langle p \rangle \langle p^* \rangle \phi) \Rightarrow \langle p^* \rangle \phi$
- 8  $\langle p^* \rangle \phi \Rightarrow (\phi \vee \langle p^* \rangle (\neg\phi \wedge \langle p \rangle \phi))$

## Règles d'Inférence

$$\text{MP} : \frac{\phi \quad \phi \Rightarrow \psi}{\psi} \quad \text{N1} : \frac{\phi}{[p]\phi}$$

## Règles d'Inférence Dérivées

On peut montrer que les règles d'inférence suivantes sont correctes. On remarquera que la logique dynamique contient la logique de Hoare.

$$\text{monotonique } \langle p \rangle : \frac{\phi \Rightarrow \psi}{\langle p \rangle \phi \Rightarrow \langle p \rangle \psi}$$

$$\text{monotonique } [p] : \frac{\phi \Rightarrow \psi}{[p]\phi \Rightarrow [p]\psi}$$

$$\text{fermeture } \langle p \rangle : \frac{(\phi \vee \langle p \rangle \psi) \Rightarrow \psi}{\langle p \rangle^* \phi \Rightarrow \psi}$$

$$\text{invariant de boucle} : \frac{\psi \Rightarrow [p]\psi}{\psi \Rightarrow [p^*]\psi}$$

$$\text{Hoare composition} : \frac{\{\phi\}p\{\sigma\} \quad \{\sigma\}q\{\psi\}}{\{\phi\}p; q\{\psi\}}$$

$$\text{Hoare condition} : \frac{\{\phi \wedge \sigma\}p\{\psi\} \quad \{\neg\phi \wedge \sigma\}q\{\psi\}}{\{\sigma\}\text{if } \phi \text{ then } p \text{ else } q \text{ fi}\{\psi\}}$$

$$\text{Hoare while} : \frac{\{\phi \wedge \psi\}p\{\psi\}}{\{\psi\}\text{while } \phi \text{ do } p \text{ od}\{\neg\phi \wedge \psi\}}$$

**Théorème 23** *Si la formule  $\phi$  de PDL est satisfaisable alors elle admet un modèle qui n'a pas plus de  $2^{|\phi|}$  états, où  $|\phi|$  est le nombre de symboles de  $\phi$ .*

# Chapter 4

## Décidabilité

### 4.1 Les grandes techniques de décidabilité

Une *théorie* est un ensemble de formule closes. En général une théorie est présentée par un nombre fini de formules closes (les axiomes de la théorie) et la théorie est l'ensemble de toutes les conséquences logiques de ces axiomes. la théorie est *décidable* s'il existe un algorithme qui prend une formule close en argument et répond oui si elle est dans la théorie (c.a.d. elle est conséquence des axiomes) et non sinon. Il y a trois grandes techniques<sup>1</sup> permettant de montrer qu'une théorie est décidable:

- Les méthodes à base de théorie des modèles.
- Les méthodes à base d'élimination de quantificateurs.
- les méthodes à base d'automates.

En général le premier type fait appel à des résultats mathématiques profonds et difficile. L'idée est de caractériser la théorie à l'aide d'un modèle dans lequel il suffit alors de travailler. On obtient la décidabilité par énumération des preuves jusqu'à trouver celle de la formule ou de sa négation. Nous donnons ici un exemple qui repose sur l'existence de modèles finis.

Les méthodes à base d'élimination de quantificateur ont un principe simple: Partant d'une formule  $Q_1x_1..Q_nx_n \varphi$  avec  $Q_i = \forall$  ou  $\exists$  et  $\varphi$  sans quantificateur on montre qu'elle est équivalente à une formule  $Q_2x_1 \dots Q_nx_n\psi$  avec

---

<sup>1</sup>pour le moment!

$\psi$  sans quantificateur et  $VL(\psi) \subseteq VL(\varphi)$ . Si la théorie permet de décider les formules closes sans quantification, on a alors une technique de décision. Le deuxième exemple traité illustre cette approche.

Les méthodes à base d'automates utilisent différentes sortes d'automates pour reconnaître les valeurs qui valident ou falsifient les formules. Les classes d'automates considérées ont de bonnes propriétés de stabilité qui permettent cette reconnaissance à n'importe quelle formule. Nous allons voir un exemple: celui de l'arithmétique de Presburger (qui se traite aussi avec une méthode classique d'élimination de quantificateur)

## 4.2 Une méthode de décision à base de modèles pour la classe monadique

**Definition 25** Une classe de formules est dite finiment contrôlable si toute formule de cette classe qui admet un modèle admet aussi un modèle fini .

**Proposition 14** Si une classe  $C$  de formules est finiment contrôlable alors la sous-classe des formules de  $C$  qui sont satisfaisables est décidable.

**Preuve** l'algorithme de décision consiste à énumérer en parallèle les preuves  $\pi(n)$  et les interprétations finies  $I(m)$ . Considérons une formule  $\phi$ ,

soit il existe  $n$  tel que  $\pi(n)$  est une preuve de  $\neg\phi$

soit il existe  $m$  tel que  $I(m) \models \phi$

et il n'y a pas d'autre possibilité. □

L'exemple le plus simple et le plus connu est la **classe monadique** formée des formules (sans symboles de fonctions) dont tous les prédicats sont d'arité 1.

**Proposition 15** Soit  $\psi$  une formule monadique dont les prédicats sont  $P_1, \dots, P_q$ . Alors si  $\psi$  est satisfaisable elle admet un modèle de taille  $2^q$ .

**Preuve** Soit  $M$  un modèle de  $\psi$ . On définit une relation d'équivalence sur le domaine de  $M$  par

$$a \sim b \text{ ssi } M \models \bigwedge_{i=1}^q P_i(a) \Leftrightarrow P_i(b)$$

On définit l'interprétation  $\widehat{M}$  dont le domaine est formé en prenant un représentant  $\widehat{a}$  dans chaque classe d'équivalence de  $\sim$  et par

$$\widehat{P}_i(\widehat{a}) = P_i(a)$$

où  $\widehat{a}$  est le représentant de la classe de  $a$ .

Montrons que  $\widehat{M}$  est un modèle de  $\psi$ . Il suffit de montrer par induction sur les sous-formules  $\phi$  de  $\psi$ :  $\forall \widehat{a}_1, \dots, \widehat{a}_k \in \widehat{M}$ , and  $\forall a_1, \dots, a_k \in M$  tel que  $\widehat{a}_i \sim a_i$ ,

$$M \models \phi(a_1, \dots, a_k) \text{ ssi } \widehat{M} \models \phi(\widehat{a}_1, \dots, \widehat{a}_k)$$

- 1  $\phi$  est atomique: l'équivalence est évidente d'après la définition de  $\widehat{M}$
- 2  $\phi$  est une combinaison booléenne de formules pour lesquelles la propriété est vraie par hypothèse d'induction. Ce cas est également facile.
- 3  $\phi(a_1, \dots, a_k) \equiv \exists x \phi'(x, a_1, \dots, a_k)$ . Si  $M \models \phi(a_1, \dots, a_k)$  alors il existe  $b \in M$  tel que  $M \models \phi'(b, a_1, \dots, a_k)$ . Par hypothèse d'induction,  $\widehat{M} \models \phi'(\widehat{b}, \widehat{a}_1, \dots, \widehat{a}_k)$ , et donc  $\widehat{M} \models \phi(\widehat{a}_1, \dots, \widehat{a}_k)$ . La réciproque est identique.

□

**Remarque 3** *La proposition reste valide mais avec une borne légèrement différente pour des formules contenant le prédicat = (simple).*

**Remarque 4** *La proposition reste valide pour des formules contenant des fonctions unaires (assez difficile).*

## 4.3 Méthode par élimination de quantificateurs

Rappelons que toute formule  $F$  peut se mettre en forme prénexe équivalente  $Q_1 x_1 \dots Q_n x_n F'$ , où  $Q_i \in \{\forall, \exists\}$ .

**Definition 26** *Une formule primitive est une formule de la forme  $\exists x(F_1 \wedge \dots \wedge F_n)$  où les  $F_i$  sont des littéraux.*

**Lemme 4** *Toute formule est équivalente à une formule sans quantificateurs dans la théorie des axiomes Ax ssi c'est le cas pour toute formule primitive.*

**Preuve** La nécessité est triviale. Pour la suffisance, on peut se restreindre à considérer des formules prénexes  $Q_1x_1 \dots Q_nx_nF'$ :

**n=0** : c'est fini.

**n=1** : Supposons  $Q_1 = \exists$ . Mettons  $F'$  en forme normale disjonctive  $G_1 \vee \dots \vee G_p$  (chaque  $G_i$  est une conjonction de littéraux).

$$Ax \models \exists F' \Leftrightarrow \exists xG_1 \vee \dots \vee \exists xG_p$$

Comme chaque  $\exists xG_i$  est primitive on peut lui appliquer l'hypothèse et obtenir le résultat.

Supposons  $Q_1 = \forall$ . Il suffit d'appliquer ce qui précède à  $\exists \neg F'$  puis prendre la négation du résultat.

$n \rightarrow n + 1$  : soit la formule  $Q_1x_1 \dots Q_nx_nQ_{n+1}x_{n+1}F'$ . Par hypothèse de récurrence ,

$$Ax \models Q_2x_2 \dots Q_nx_nQ_{n+1}x_{n+1}F' \Leftrightarrow G$$

où  $G$  est sans quantificateur. On applique ensuite le résultat pour  $n = 1$  à  $Q_1x_1G$ .

□

### 4.3.1 Exemple 1: théorie de l'égalité sur un ensemble infini

La théorie Ax comprend les axiomes d'égalité ainsi que la famille infinie de formules:

$$\exists z_0, z_1, \dots, z_n \bigwedge_{0 \leq i < j \leq n} z_i \neq z_j$$

Considérons une formule primitive  $F: \exists x(F_1 \wedge \dots \wedge F_n)$  où chaque  $F_i$  est du type  $x = x, x \neq x, x = y, x \neq y$ .

Si un  $F_i$  est du type  $x \neq x$  alors  $Ax \models F \Leftrightarrow x \neq x$ .

Sinon on peut supprimer dans  $F$  les  $F_i$  du type  $x = x$ .

Donc on se ramène aux formules du type:

$$\exists x(x = x_1 \wedge \dots \wedge x = x_m \wedge x \neq y_1 \wedge \dots \wedge x \neq y_n)$$

Si  $m = 0$  la formule est une conséquence d'un axiome. Sinon elle est équivalente (modulo les axiomes) à:

$$x_1 = x_2 \wedge \dots \wedge x_1 = x_m \wedge x_1 \neq y_1 \wedge \dots \wedge x_1 \neq y_n$$

qui n'a pas de quantificateurs.

### 4.3.2 Exemple 2: théorie élémentaire du successeur

La signature contient la constante 0 et la fonction unaire  $s$ . Les axiomes de Ax sont les suivants:

les axiomes d'égalité

$$\forall x \ 0 \neq s(x)$$

$$\forall x, y \ s(x) = s(y) \Rightarrow x = y$$

$$\forall x \ x = 0 \vee \exists y \ x = s(y)$$

$$\neg \exists x_1, \dots, x_n (x_2 = s(x_1) \wedge x_3 = s(x_2) \wedge \dots \wedge x_1 = s(x_n)).$$

Un atome contenant  $x$  est soit  $s^m(x) = s^n(x)$  soit  $s^m(x) = t$  avec  $x \notin t$ . Remarquons que

$$n = m \ Ax \models s^m(x) = s^n(x) \Leftrightarrow 0 = 0$$

$$n \neq m \ Ax \models s^m(x) = s^n(x) \Leftrightarrow 0 \neq 0$$

On est donc ramené aux formules suivantes:

$$\exists x(s^{n_1}(x) = t_1 \wedge \dots \wedge s^{n_k}(x) = t_k \wedge s^{m_1}(x) \neq u_1 \wedge \dots \wedge s^{m_l}(x) \neq u_l)$$

avec  $x \notin t_i, u_j$ . D'autre part on peut supposer  $n_1 = \dots = n_k = m_1 = \dots = m_l = n$  quitte à modifier les membres droits:

$$\exists x(s^n(x) = t_1 \wedge \dots \wedge s^n(x) = t_k \wedge s^n(x) \neq u_1 \wedge \dots \wedge s^n(x) \neq u_l)$$

qui est équivalente (modulo Ax) à:

$$\exists y(y \neq 0 \wedge y \neq s(0) \wedge \dots \wedge y \neq s^{n-1}(0) \wedge y = t_1 \wedge \dots \wedge y = t_k \wedge y \neq u_1 \wedge \dots \wedge y \neq u_l)$$

Pour  $k \neq 0$  cette formule est équivalente (modulo Ax) à:

$$(t_1 \neq 0 \wedge t_1 \neq s(0) \wedge \dots \wedge t_1 \neq s^{n-1}(0) \wedge t_1 = t_2 \wedge \dots \wedge t_1 = t_k \wedge t_1 \neq u_1 \wedge \dots \wedge t_1 \neq u_l)$$

Pour  $k = 0$  la formule est équivalente (modulo Ax) à  $0 = 0$  car il est toujours possible de choisir un terme différent des éléments d'un ensemble fini.

On en déduit que pour toute formule close  $F$  il existe une formule sans quantificateurs  $G$  tel que:

$$Ax \models F \Leftrightarrow G(x_1, \dots, x_n)$$

En particulier:

$$Ax \models F \Leftrightarrow G(0, \dots, 0)$$

Mais il est facile de décider la validité de  $G(0, \dots, 0)$  composé d'atomes  $s^m(0) = s^n(0)$ . Donc la théorie élémentaire du successeur est décidable.

## 4.4 Une méthode à base d'automates pour l'arithmétique de Presburger

**Definition 27** *L'arithmétique de Presburger se compose de*

- l'ensemble  $\mathbb{N} = \{0, 1, 2, \dots\}$  muni de l'addition  $+$  sur les entiers,
- du prédicat d'égalité  $=$ .

Malgré sa pauvreté apparente, ce langage permet d'exprimer un certain nombre de prédicats intéressants. par exemple  $x < y$  peut se définir par la formule  $\exists z y = x + z \wedge z \neq 0$ .

Nous allons montrer que la théorie de l'arithmétique de Presburger, c.a.d l'ensemble des formules closes de ce langage, est décidable en utilisant une méthode à base d'automates finis.

Les entiers sont écrits en base 2, donc comme une suite de 0 et 1. Nous les écrirons en mettant les bits de poids faibles à gauche, et nous lirons ces nombres de la gauche vers la droite. De plus nous nous compléterons les nombres par des 0 à droite afin de travailler sur des chaînes de même longueur. Pour commencer nous allons montrer comment résoudre l'équation  $x = y + z$  avec des automates d'états finis.

### 4.4.1 Codage de l'addition $x = y + z$

Il suffit en fait de coder l'addition en binaire et les souvenirs d'école primaire permettent de voir qu'il y a deux états liés au fait qu'il peut y avoir une retenue ou pas.

- s'il n'y a pas de retenue alors les triplets suivant correspondent à l'addition  $x$  0 1 1 et aussi  $x$  0 mais dans ce cas on engendre

$$\begin{array}{r} y \ 0 \ 0 \ 1 \\ z \ 0 \ 1 \ 0 \end{array} \qquad \begin{array}{r} y \ 1 \\ z \ 1 \end{array}$$

une retenue.

- s'il y a une retenue alors les triplets sont admissibles  $x$  1 0 0 et on

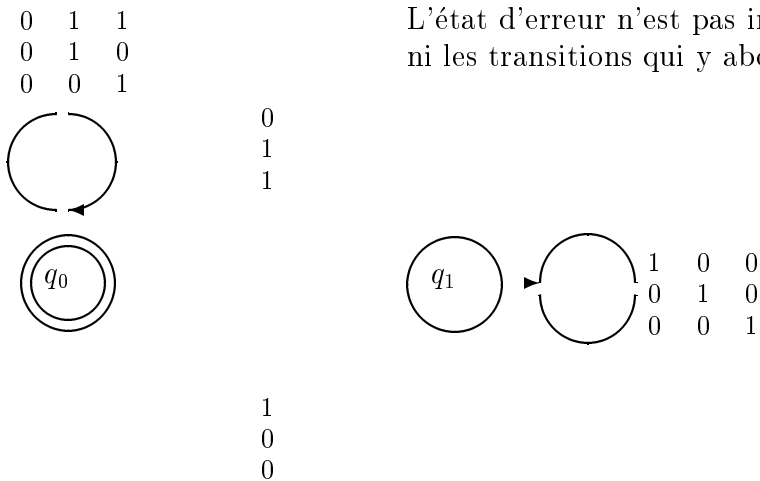
$$\begin{array}{r} y \ 1 \ 0 \ 1 \\ z \ 1 \ 1 \ 0 \end{array}$$

reste avec une retenue pour l'étape suivante, mais on a aussi  $x$  1 et

$$\begin{array}{r} y \ 0 \\ z \ 0 \end{array}$$

il n'y a plus de retenue.

Par conséquent l'automate suivant qui travaille sur un alphabet dont les symboles sont des triplets sur le vocabulaire 0 ou 1 reconnaît toutes les solutions de  $x = y + z$ . Un automate possible est:



## 4.4.2 Résolution de formules

Nous rappelons que les langages réguliers sont stables par union intersection, complémentaire. Nous noterons  $\mathcal{A}_1 + \mathcal{A}_2$  un automate qui reconnaît l'union des langages reconnus par  $\mathcal{A}_1$  et  $\mathcal{A}_2$ , par  $\mathcal{A}_1 \times \mathcal{A}_2$  un automate qui reconnaît l'intersection et  $\bar{\mathcal{A}}$  un automate qui reconnaît le complémentaire. Ces automates sont effectivement calculables.

### Formules sans quantificateur

Dans cette partie nous montrons que la recherche de solution d'une équation se ramène à la recherche des solutions d'un système d'équation, après ajout de variables supplémentaires, et que le calcul des solutions se fait à l'aide d'automates.

- Cas initial: équation  $x = y + z$  ou  $x = n$  ( $n \in \mathbb{N}$ ). Alors le résultat est évident d'après ce qui précède ou immédiat.
- Equation quelconque.

– Remplacer  $n_1x_1 + \dots + n_px_p + a = m_1y_1 + \dots + m_qy_q + b$  par le système

$$\begin{cases} X_1 + \dots + X_p + Z = Y_1 + \dots + Y_q + T \\ X_i = n_ix_i \\ Y_i = m_iy_i \\ Z = a \\ T = b \end{cases}$$

– une équation  $n.X = Y$  se remplacera par:

$$\begin{cases} X_1 = Y \\ X_1 = X + X_2 \\ \dots \\ X_{n_2} = X + X_{n-1} \\ X_{n-1} = X + X \end{cases}$$

Donc en itérant le processus, on peut transformer toute équation en conjonctions d'équations élémentaires  $X = a$  ou  $X = Y + Z$  au prix de l'augmentation du nombre de variables. Comme on sait reconnaître

les solutions des équations élémentaires (en les considérant comme des formules sur toutes les variables utilisées), on saura reconnaître les solutions de toute combinaison utilisant la conjonction, la disjonction et la négation d'après les propriétés de stabilité des langages réguliers par union intersection et complément.

### Cas général

Nous montrons de même que les solutions des formules quelconques sont reconnues par un automate d'état fini.

- Formule de la forme  $\varphi\{\wedge, \vee\}\psi$  ou  $\neg\varphi$ .  
Si  $X_1, \dots, X_n$  sont les variables libres de  $\varphi$  et  $\psi$ , on construit les automates  $\mathcal{A}$  et  $\mathcal{B}$  qui reconnaissent les solutions de  $\varphi$  et  $\psi$ , en considérant que les variables libres de  $\psi$  et  $\varphi$  sont  $X_1, \dots, X_n$ . Alors
  - les solutions de  $\varphi \wedge \psi$  sont reconnues par  $\mathcal{A} \times \mathcal{B}$ ,
  - les solutions de  $\varphi \vee \psi$  sont reconnues par  $\mathcal{A} + \mathcal{B}$ ,
  - les solutions de  $\neg\varphi$  sont reconnues par  $\bar{\mathcal{A}}$
- Formule de la forme  $\exists x \varphi$ .  
Si  $\mathcal{A}$  reconnaît les solutions de  $\varphi$ , et si  $x$  est une variable libre de  $\varphi$  alors l'automate  $\mathcal{B}$  obtenu en supprimant la ligne correspondant à  $x$  reconnaît les solutions de  $\exists x \varphi$ .
- Formule de la forme  $\forall x \varphi$ .  
Il suffit de remarquer que  $\forall x \varphi$  est équivalent à  $\neg\exists x \neg\varphi$  dont on sait reconnaître les solutions d'après ce qui précède.

### 4.4.3 Application aux formules closes

Lorsqu'on applique la technique précédente à des formules closes, on obtient un automate réduit à sa plus simple expression qui est l'état initial. La formule sera vraie ssi cet état initial est aussi un état final. Nous en déduisons le résultat annoncé au début.

**Théorème 24** *L'arithmétique de Presburger est décidable.*

La technique présentée ne dépend pas de la base choisie. Il est possible de travailler avec les entiers en base  $2, 3, \dots$ . De plus il est facile de reconnaître par automate travaillant en base  $p$  le prédicat *est une puissance de  $p$*  et d'ajouter ce prédicat tout en gardant la décidabilité.

# Chapter 5

## Indécidabilité

### 5.1 Calculabilité

La notion de calcul a été au centre de la recherche du début du siècle 1930 à 1940), alors que les ordinateurs n'existaient pas. Il est intéressant de voir que la notion de calcul a été élaboré avant l'outil qui permet de les mécaniser. Plusieurs formalismes ont été proposés par Church (lambda-calcul) et Turing (machines de Turing) notamment. En même temps Gödel prouvait ses théorèmes d'incomplétude qui posent une limite précise à ce qu'on peut espérer: tout système logique assez puissant i.e. contenant l'arithmétique, est incomplet c'est à dire contient des propositions qu'on ne peut pas démontrer en se limitant aux règles du systèmes. Le deuxième théorème de Gödel montre que la proposition qui établit la consistance du système n'est pas démontrable dans le système lui-même. Nous présentons ici la notion de calcul à l'aide des machines de Turing.

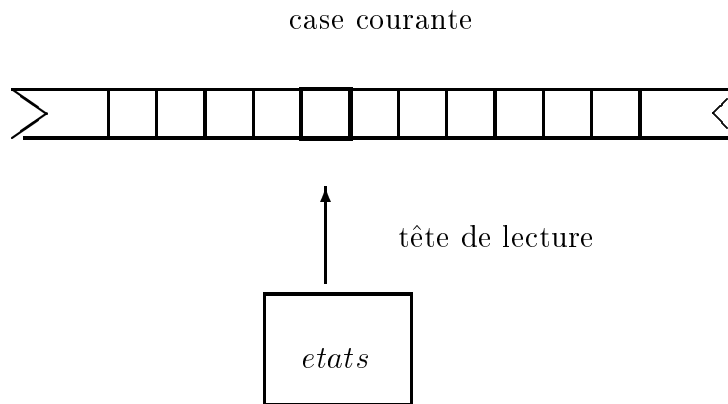
#### 5.1.1 Machines de Turing

Une machine de Turing a été conçue pour modéliser le comportement d'un mathématicien qui effectue des calculs sur une feuille de papier (aussi grande qu'on veut) en écrivant ou effaçant des symboles sur cette feuille. En fait, une machine de Turing est une extension simple de la notion usuelle d'automate d'état fini où d'automates à pile.

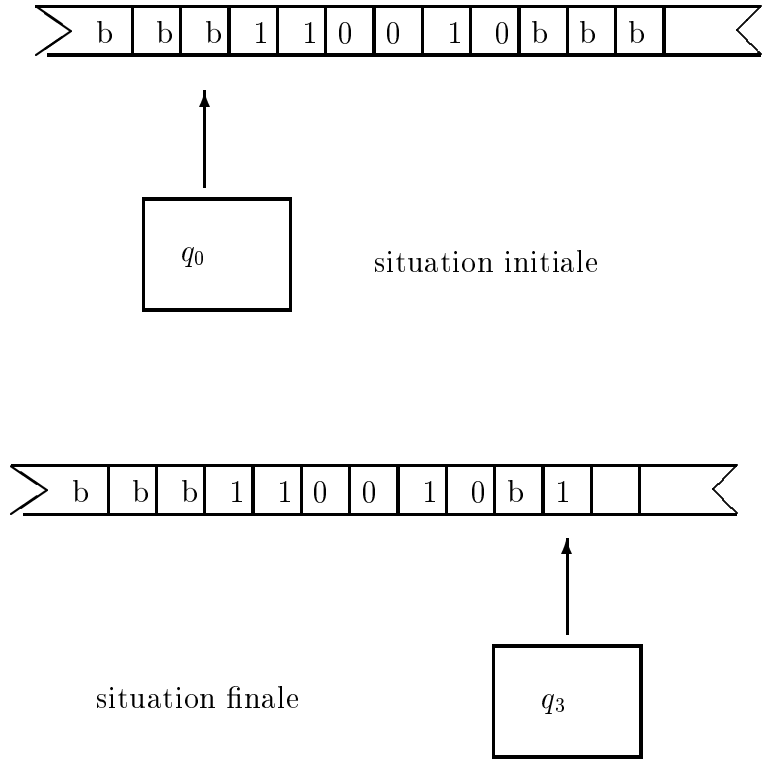
**Definition 28** *Une machine de Turing est composée*

- d'un alphabet fini  $X$  qui comporte un symbole particulier le blanc  $b$ ,

- d'une bande infinie à gauche et droite, formée de cases marquées par un symbole de  $X$ , de plus seul un nombre fini de cases est marqué par autre chose que le symbole  $b$ ,
- d'une tête de lecture positionné devant une case de la bande,
- d'un ensemble d'états.



**Exemple 12** Machine effectuant un comptage de parité. L'alphabet  $X$  est formé de  $b$  et de 0 et 1. La donnée est un mot de  $\{0, 1\}^*$  et le résultat sera 1 si le nombre de 1 est pair et 0 sinon. Partant de la configuration où la tête de lecture est positionnée sur le premier caractère du mot, on arrive à une configuration finale où le résultat est affiché après le mot en laissant un blanc entre le mot et le résultat comme dans l'exemple suivant.



La machine de Turing peut se décrire par la table de transition:

$(q_0, 0) \rightarrow (q_0, D)$	$(q_0, 1) \rightarrow (q_1, D)$
$(q_1, 0) \rightarrow (q_1, D)$	$(q_1, 1) \rightarrow (q_0, D)$
$(q_0, b) \rightarrow (q_2, D)$	$(q_1, b) \rightarrow (q_3, D)$
$(q_2, b) \rightarrow (q_2, 0)$	$(q_3, b) \rightarrow (q_3, 1)$

La machine s'arrête si elle est dans un état ou plus aucune transition n'est possible.

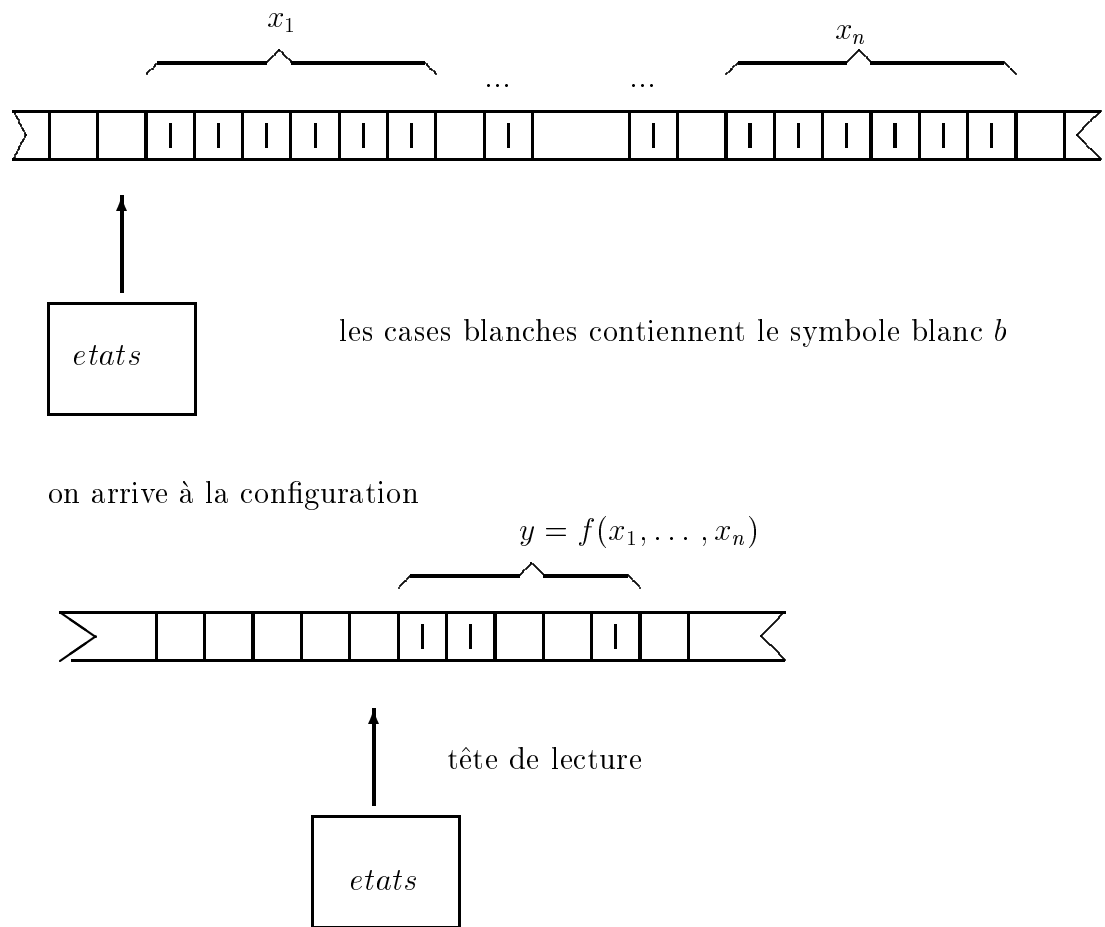
### 5.1.2 Code d'une machine de Turing

On peut se limiter aux machines de Turing sur l'alphabet  $X = \{ |, b \}$  avec | un *baton*. Le codage d'un entier  $n$  se fait par la juxtaposition de  $n$  batons cote à cote. Pareillement un état  $q_i$  peut se coder par l'entier  $i$ . D'autre part une machine de Turing est complètement déterminée par sa table de transition. Si on code le retour à la ligne par un caractère particulier  $\alpha$  et la séparation

entre les colonnes de la table par un autre caractère  $\beta$ , on peut écrire cette table de transition comme un mot sur l'alphabet  $0, \dots, 9, |, ", \alpha, \beta, D, G$  qu'on considère comme l'écriture en base 15 d'un entier. Ce nombre est appelé le *code* de la machine de Turing. Comme le code est un entier, il est possible de faire calculer la machine sur cette donnée.

### 5.1.3 Fonctions calculables

Etant donné une fonction à  $n$  arguments, on dit que la machine  $T$  calcule  $f$  ssi pour chaque configuration initiale



### 5.1.4 La thèse de Church

La thèse de Church est le postulat que toute fonction calculable (sens intuitif) est calculable par les machines de Turing.

En effet on a prouvé que toutes les propositions de fonctions calculables données par ailleurs (lambda calcul, programme *do while*, ...) sont équivalentes à la définition au sens des machines de Turing. Ceci est un argument fort en faveur de la thèse de Church qui est admise communément.

## 5.2 Problèmes indécidables

La notion de calculabilité que nous avons utilisée permet de trouver aisément qu'il y a des fonctions non-calculables et des problèmes indécidables.

### 5.2.1 Existence de fonction non-calculables

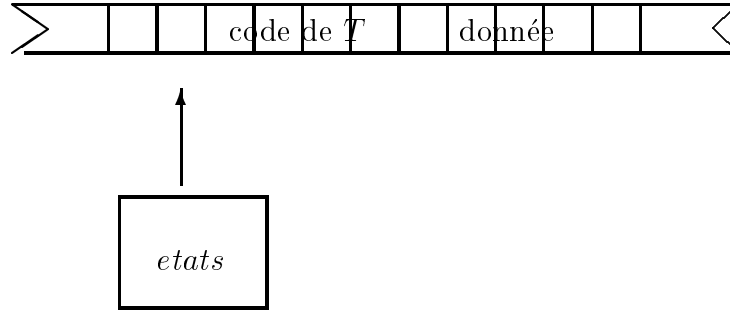
Un argument de cardinalité simple montre qu'il existe des fonctions non-calculables. L'ensemble des machines de Turing est dénombrable, par contre l'ensemble des fonctions de  $\mathbb{N}$  dans  $\mathbb{N}$  n'est pas dénombrable, donc il existe au moins une (en fait beaucoup!) fonction non-calculable. Le paragraphe suivant décrit une fonction non-calculable: celle qui donne vrai (codé par 1) si une machine de Turing  $T$  s'arrête sur une donnée  $d$ , et faux (codé par 0) sinon.

### 5.2.2 Le problème de l'arrêt

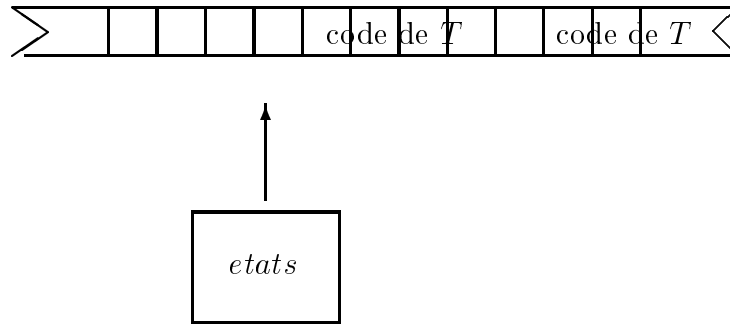
Le problème de l'arrêt est le suivant: *existe-t-il une machine de Turing  $T_H$  ( $H$  pour Halting problem) qui prend comme arguments le code d'une machine de Turing  $T$ , un entier  $n$  et répond oui (codé par 1) si le calcul de la machine  $T$  sur  $n$  s'arrête et répond non (codé par 0) sinon.*

Cette formulation est pratiquement identique à celle donnée dans le paragraphe précédent, la machine de Turing étant identifiée par son code qui est un entier qu'on peut donc donner comme argument à la machine  $T_H$ . Nous allons montrer que cette machine n'existe pas en raisonnant par l'absurde.

Si  $T_H$  existe, elle calcule sur des configurations du type suivant.



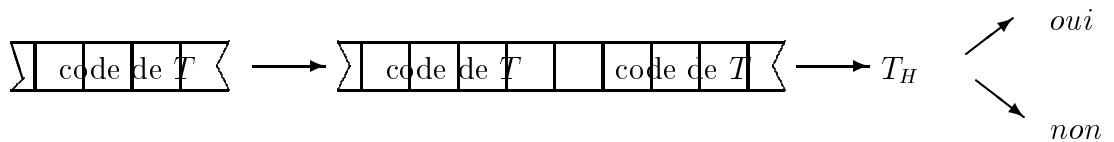
En particulier on peut l'appliquer à:



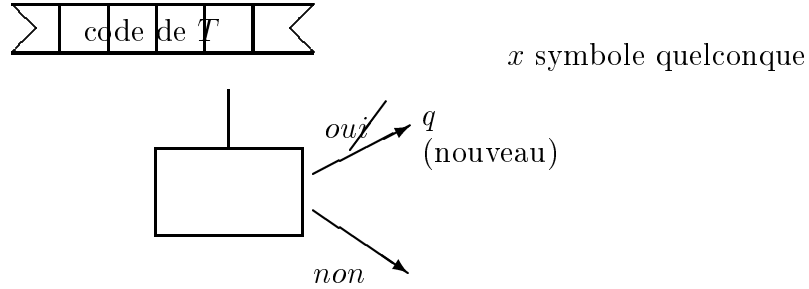
On construit de plus une nouvelle machine  $S$  telle que on ait:

- donnée:  $code_T$  le code de d'une machine de Turing
- resultat: oui si  $T$  s'arrête sur  $code_T$ , non sinon.

Il suffit de connecter une machine qui duplique une bande à la machine  $T_H$ .



On modifie  $S$  de façon à ce qu'elle boucle au lieu de donner oui:



Appliquons  $S$  à son code, on a la situation suivante:

- $S$  s'arrête si  $S$  appliquée à son code ne s'arrête pas.
- $S$  ne s'arrête pas si  $S$  appliquée à son code ne s'arrête pas.

D'où la contradiction obtenue à l'aide d'une technique de réflexivité (appliquer l'objet à lui-même) classique et source de nombreux paradoxes.

### 5.2.3 Problèmes indécidables

Parmi les problèmes indécidables les plus classiques on peut citer:

- La décision de la logique du premier ordre.
- La théorie de l'arithmétique  $(\mathbb{N}, +, *, 0, 1)$ .
- Le problème de l'existence de solution entières positives aux équations de la forme  $P(x_1, \dots, x_n) = 0$  avec  $P$  un polynome à coefficients entiers relatifs (dixième problème de Hilbert). Remarquez que ce problème est un cas particulier du précédent.
- Le problème de l'ambiguité, de l'équivalence des grammaires algébriques.
- Le problème de Post: on donne un ensemble fini de couples de mots  $(\alpha_1, \beta_1), \dots, (\alpha_n, \beta_n)$  sur un vocabulaire  $V = \{a_1, \dots, a_p\}$ , et on cherche à savoir s'il existe une suite  $i_1, i_2, \dots, i_m$  telle que:  $\alpha_{i_1} \dots \alpha_{i_m} = \beta_{i_1} \dots \beta_{i_m}$ .