Chapitre 6

Raisonnement en logique modale

Nous présentons dans ce chapitre trois approches de déduction automatique pour la logique modale, la méthode des tableaux sémantiques, la méthode des séquents de Gentzen ainsi que la programmation logique.

6.1. Tableaux sémantiques

La méthode des tableaux sémantiques est désormais une méthode de déduction bien établie dans le domaine de la déduction automatique [OPP 88, BAU 95, BEC 95]. Elle constitue une alternative à des méthodes plus traditionnelles basées sur la résolution [CHA 73]. Nous présentons la méthode des tableaux sémantiques pour la déduction automatique en logique propositionnelle, logique des prédicats et logique modale. Ces logiques ainsi que les notations qui s'y réfèrent sont présentées, respectivement dans les annexes A, B et C.

La méthode des tableaux sémantiques basée sur une idée originale de E. Beth [BET 55], a été proposée par Smullyan [SMU 68], puis popularisée par M. Fitting [FIT 83]. Cette méthode est une méthode de réfutation [1], c'est à dire que pour prouver qu'une formule F est valide (ou est une tautologie), on montre que $\neg F$ est incohérente (ou insatisfaisable). Pour cela, on décompose la formule $\neg F$ en sous-formules, de telle sorte que si $\neg F$ est satisfaite alors les sous-formules qui la composent sont aussi satisfaites, jusqu'à l'obtention de formules atomiques et on montre que des contradictions sont produites. La décomposition s'effectue par la construction d'un arbre dont les noeuds sont étiquetés par les sous-formules obtenues dans les différentes étapes de la décomposition.

Chapitre rédigé par Odile PAPINI.

1. Pour plus de détails sur les méthodes de réfutation voir section 5.1.

6.1.1. *Tableaux sémantiques en logique propositionnelle*

Soit F une formule propositionnelle, la méthode des tableaux sémantiques repose sur la construction d'un arbre comme suit :

DÉFINITION 6.1.– *Le tableau sémantique pour la formule $\neg F$, noté T, est un arbre dont :*

– la racine est étiquetée par $\neg F$;

– les noeuds sont étiquetés par des formules propositionnelles ;

– les successeurs des noeuds sont produits par des règles d'expansion.

Il existe plusieurs règles d'expansion pour construire les tableaux :

6.1.1.1. *Règle de prolongation*

La règle de prolongation (appelée également règle α) représente la satisfaction d'une formule, notée $\alpha = \alpha_1 \wedge \alpha_2$, composée de la conjonction de deux sous-formules α_1 et α_2 et signifie que pour satisfaire α il faut satisfaire α_1 et α_2. La règle est représentée par la table 6.1.

α
α_1
α_2

Tableau 6.1. *Règle de prolongation.*

Les règles de prolongation pour la logique propositionnelle sont données dans la table 6.2.

α	$x \wedge y$	$\neg(x \vee y)$	$\neg(x \rightarrow y)$	$x \leftrightarrow y$
α_1	x	$\neg x$	x	$x \rightarrow y$
α_2	y	$\neg y$	$\neg y$	$x \rightarrow y$

Tableau 6.2. *Règles de prolongation pour la logique propositionnelle.*

6.1.1.2. *Règle de ramification*

La règle de ramification (appelée également règle β) représente la satisfaction d'une formule, notée $\beta = \beta_1 \vee \beta_2$, composée de la disjonction de deux sous-formules β_1 et β_2 et signifie que pour satisfaire β, il faut satisfaire β_1 ou β_2. La règle est représentée par la table 6.3.

Les règles de ramification pour le logique propositionnelle sont sont données dans la table 6.4.

$$\frac{\beta}{\beta_1 \mid \beta_2}$$

Tableau 6.3. *Règle de ramification.*

$$\frac{\beta}{\beta_1 \mid \beta_2} \qquad \frac{x \vee y}{x \mid y} \qquad \frac{\neg(x \wedge y)}{\neg x \mid \neg y}$$

$$\frac{\neg(x \rightarrow y)}{\neg x \mid y} \qquad \frac{\neg(x \leftrightarrow y)}{\neg(x \rightarrow y) \mid \neg(y \rightarrow x)}$$

Tableau 6.4. *Règles de ramification pour la logique propositionnelle.*

6.1.1.3. *Règle de négation*

Les règles de négation représentent la satisfaction des formules composées de la négation des constantes propositionnelles $\top$ et $\bot$ et sont représentées par la table 6.5.

$$\frac{\neg\top}{\bot} \qquad \frac{\neg\bot}{\top}$$

Tableau 6.5. *Règles de négation.*

6.1.1.4. *Règle de double négation*

La règle de double négation représente la satisfaction d'une formule, notée $\neg\neg\alpha = \alpha$, composée de la double négation d'une sous-formule propositionnelle α et est donnée par la figure 6.6. L'exemple suivant illustre la construction d'un tableau sémantique.

EXEMPLE 6.1.– Soit la formule propositionnelle $F = (\neg p \wedge q) \vee (p \vee \neg q)$, le tableau sémantique pour la formule $\neg F = \neg((\neg p \wedge q) \vee (p \vee \neg q))$ est donné par la figure 6.1. La racine de l'arbre, c'est à dire, le noeud (1), est étiquetée par la formule $\neg F$. La règle de prolongation produit les noeuds (2) et (3). La règle de ramification sur

$$\frac{\neg\neg\alpha}{\alpha}$$

Tableau 6.6. *Règle de double négation.*

la sous-formule étiquetant le noeud (2) produit les noeuds (4) et (5), et la règle de prolongation sur la sous-formule étiquetant le noeud (3) produit les noeuds (6) et (7) et les noeuds (8) et (9). La règle de double négation sur la sous-formule étiquetant le noeud (4) produit le noeud (10), la règle de double négation sur la sous-formule étiquetant le noeud (7) produit le noeud (11), la règle de double négation sur la sous-formule étiquetant le noeud (9) produit le noeud (12). Les feuilles de l'arbre, les noeuds (11) et (12) sont étiquetés par des formules propositionnelles atomiques et plus aucune règle d'expansion ne peut s'appliquer.

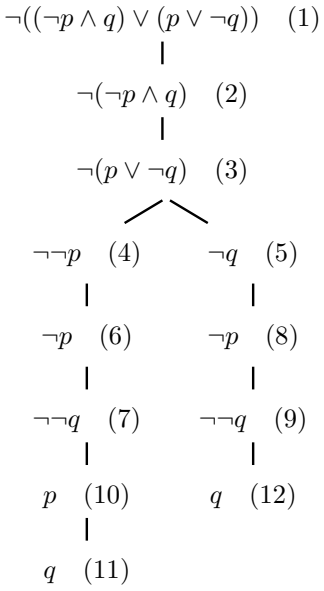

Figure 6.1. *Tableau sémantique pour* $\neg F = \neg((\neg p \wedge q) \vee (p \vee \neg q))$.

Une branche d'un tableau sémantique représente la conjonction des sous-formules qui étiquettent les noeuds de la branche, et un tableau sémantique représente une disjonction de branches. Plus formellement :

DÉFINITION 6.2.– *Une branche d'un tableau sémantique T, notée b_i est une suite finie de noeuds $n_1, \cdots, n_i, n_{i+1}, \cdots, n_k$ telle que n_1 est la racine de T et tout noeud*

n_{i+1} est obtenu à partir d'un noeud n_j, $j \leq i$, par l'application d'une des règles d'expansion (prolongation, ramification, négation ou double négation).

Les tableaux sémantiques s'étendent aux ensembles de formules comme suit :

DÉFINITION 6.3.– *Soit $\{A_1, A_2, \cdots, A_n\}$ un ensemble fini de formules propositionnelles,*

– l'arbre composé d'une seule branche donné dans la figure 6.2 est un tableau sémantique pour $\{A_1, A_2, \cdots, A_n\}$;

– si T est un tableau sémantique pour $\{A_1, A_2, \cdots, A_n\}$ et T^* est le tableau sémantique qui résulte de T par l'application des règles d'expansion alors T^* est un tableau sémantique pour $\{A_1, A_2, \cdots, A_n\}$.

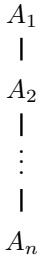

Figure 6.2. *Tableau sémantique pour $\{A_1, A_2, \cdots, A_n\}$.*

DÉFINITION 6.4.– *Soit T un tableau sémantique et b_i une branche de T, on dit que b_i est une branche complète , si pour chaque prolongation α qui figure dans b_i, les deux constituants α_1 et α_2 se trouvent sur la branche b_i, si pour chaque ramification β qui figure dans b_i, au moins l'un des deux constituants β_1 et β_2 figure sur la branche b_i, si pour chaque double négation $\neg\neg\alpha$ qui figure dans b_i, α figure dans b_i, si $\neg\top$ figure dans b_i alors $\bot$ figure dans b_i et si $\neg\bot$ figure dans b_i alors $\top$ figure dans b_i.*

DÉFINITION 6.5.– *Un tableau sémantique T est complet si toutes les branches de T sont complètes.*

Le tableau sémantique de l'exemple 6.1 est un tableau complet.

DÉFINITION 6.6.– *Soit T est un tableau sémantique et b_i une branche de T :*

– une branche b_i est fermée (ou close) s'il existe une formule propositionnelle F telle que les formules F et $\neg F$ figurent à la fois dans b_i ou si $\bot$ figure dans b_i ;

– une branche b_i est atomiquement fermée (ou close) s'il existe une proposition p telle que les propositions p et $\neg p$ figurent à la fois dans b_i ou si $\bot$ figure dans b_i ;

– une branche ouverte est une branche qui n'est pas fermée.

DÉFINITION 6.7.– *Soit T est un tableau sémantique et b_i une branche de T :*

– un tableau sémantique T est fermé (ou clos) si toutes les branches de T sont atomiquement fermées ;

– un tableau sémantique T est ouvert si T possède au moins une branche ouverte.

DÉFINITION 6.8.– *Soit T est un tableau sémantique et b_i une branche de T :*

– une branche b_i d'un tableau sémantique T est satisfaisable (ou cohérent) si l'ensemble des formules qui la composent est satisfaisable (ou cohérent) ;

– un tableau sémantique T est satisfaisable (ou cohérent) si au moins un branche de T est satisfaisable (ou cohérente).

Dans l'exemple 6.1 le tableau sémantique pour $\neg F$ est fermé, les feuilles de l'arbre, les noeuds (11) et (12) sont étiquetés par des formules propositionnelles atomiques et des contradictions sont produites car la branche gauche de l'arbre contient le noeud (6) étiquetés par $\neg p$ et le noeud (10) étiqueté par p et la branche droite de l'arbre contient le noeud (5) étiqueté par $\neg q$ et le noeud (12) étiqueté par q. Toutes les branches du tableau sémantique sont fermées, le tableau sémantique pour $\neg F$ est fermé, la formule $\neg F$ est donc incohérente et par conséquent la formule F est une tautologie.

PROPOSITION 6.1.– *Toute application d'une règle d'expansion sur un tableau sémantique satisfaisable produit un autre tableau satisfaisable.*

La méthode des tableaux sémantiques est utilisée pour prouver l'incohérence d'un ensemble de formules comme suit.

PROPOSITION 6.2.– *S'il existe un tableau sémantique fermé pour un ensemble de formules propositionnelles $\mathcal{F}$, alors $\mathcal{F}$ n'est pas satisfaisable (ou incohérent).*

Cette proposition établit le principe de la méthode des tableaux sémantiques pour établir l'incohérence d'un ensemble de formules $\mathcal{F}$. On construit le tableau sémantique T pour l'ensemble de formules $\mathcal{F}$. La construction du tableau sémantique s'arrête si toutes les branches sont fermées ou s'il n'est plus possible d'utiliser les règles d'expansion sur les sous-formules qui étiquettent les noeuds de l'arbre. Si toutes les branches de T sont fermées alors $\mathcal{F}$ est incohérent. Il n'est pas toujours nécessaire de construire un tableau sémantique complet, comme l'illustre l'exemple 6.2. L'ordre d'application des règles d'expansion permet de produire des branches atomiquement fermées qui ne sont pas forcément complètes, dans ce cas l'arbre construit comporte moins de noeuds et la preuve d'incohérence est plus courte. L'efficacité de la méthode des tableaux sémantiques réside dans le choix judicieux d'heuristiques pour l'ordre d'application des règles d'expansion.

EXEMPLE 6.2.– Soit l'ensemble de formules $\mathcal{F} = \{(p \vee q) \wedge (\neg p \rightarrow r),\ q \rightarrow \neg r,\ \neg p\}$, le tableau sémantique initial pour l'ensemble de formules $\mathcal{F}$ est donné par la figure 6.3. L'application des règles d'expansion sur ce tableau donne le tableau sémantique de la figure 6.4.

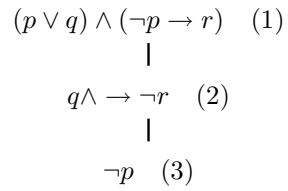

Figure 6.3. *Tableau sémantique pour* $\mathcal{F} = \{(p \vee q) \wedge (\neg p \rightarrow r),\ q \rightarrow \neg r,\ \neg p\}$.

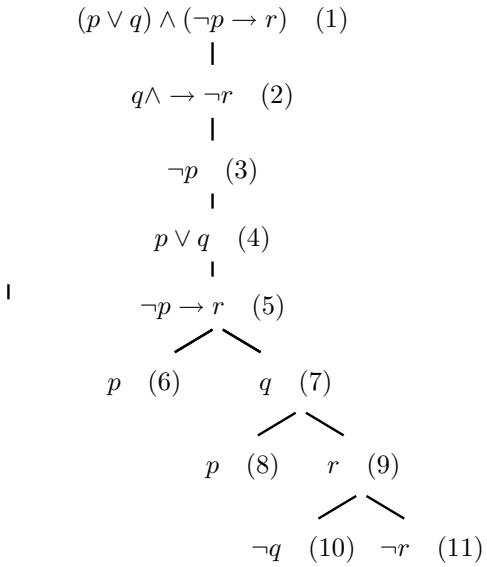

Figure 6.4. *Tableau sémantique après application des règles d'expansion à l'ensemble de formules* $\mathcal{F} = \{(p \vee q) \wedge (\neg p \rightarrow r),\ q \rightarrow \neg r,\ \neg p\}$.

En appliquant successivement les règles d'expansion aux sous formules qui étiquettent respectivement les noeuds (2), (4) et (5) dans cet ordre, on obtient un arbre comportant 13 noeuds. Toutes les branches du tableau sont fermées, l'ensemble de formules $\mathcal{F}$ est incohérent. On peut également construire un tableau comportant 11 noeuds soit en appliquant successivement les règles d'expansion aux sous formules qui étiquettent respectivement les noeuds (4), (2) et (5) dans cet ordre, soit en appliquant successivement les règles d'expansion aux sous formules qui étiquettent respectivement les noeuds (5), (2) et (4) dans cet ordre comme l'illustre l'exemple de la figure 6.4.

La notion de preuve avec la méthode des tableaux sémantiques est définie par :

DÉFINITION 6.9.– *Soit F une formule propositionnelle, on dit que F a une preuve par tableaux sémantiques si le tableau sémantique pour $\neg F$ est fermé.*

Les résultats d'adéquation et de complétude sont données par le théorème suivant :

THÉORÈME 6.1.– *Soit F une formule propositionnelle, F est une tautologie si et seulement si F a une preuve par tableaux sémantiques.*

Cela conduit à la déduction avec la méthode des tableaux sémantiques :

THÉORÈME 6.2.– *Soit F une formule propositionnelle et $\mathcal{F}$ un ensemble de formules propositionnelles, $\mathcal{F} \models F$ si et seulement si le tableau sémantique pour $\mathcal{F} \cup \{\neg F\}$ est fermé.*

La taille d'une preuve par tableaux sémantiques est mesurée en nombre de noeuds internes de l'arbre correspondant au tableau et la complexité théorique de cette méthode a été donnée par F. Massacci [MAS 00b] qui a montré le résultat suivant :

THÉORÈME 6.3.– *Soit Σ_n un ensemble de clauses n-aires, la complexité de la preuve par tableaux de l'incohérence de Σ_n admet comme borne supérieure $\mathcal{O}(2^{n^2})$.*

D'un point de vue pratique, des démonstrateurs de théorèmes utilisant les tableaux sémantiques ont été développés et implantés [OPP 88, SCH 90, HEU 96]. Une conférence internationale, appelée "TABLEAUX" est dédiée aux travaux de recherche sur le raisonnement automatique basé sur les tableaux sémantiques, pour plus de détails consulter le site WEB de la conférence dont l'URL est http ://i12www.ira.uka.de/TABLEAUX/.

L'avantage de la méthode des tableaux sémantiques réside dans le fait que celle-ci n'examine pas toutes les interprétations indistinctement, elle examine seulement les interprétations qui sont susceptibles de conduire à une contradiction. Par ailleurs, elle s'étend aisément à d'autres logiques comme la logique des prédicats ou la logique modale.

6.1.2. *Tableaux sémantiques en logique des prédicats*

La méthode des tableaux sémantiques s'étend à la logique des prédicats par l'ajout de règles pour les formules quantifiées universellement et existentiellement.

6.1.2.1. *Règle d'instanciation universelle*

La règle d'instanciation universelle (appelée également règle γ) représente la satisfaction d'une formule quantifiée universellement et spécifie que si une formule quantifiée universellement est vraie alors elle est vrai pour toute instance d'élément de l'univers de Herbrand. Cette règle est représenté par :

Les règles d'instanciation universelles pour la logique des prédicats sont les suivantes, où a est une constante quelconque du langage de la logique des prédicats :

$$\frac{\gamma}{\gamma_0(a)}$$

Tableau 6.7. *Règle gamma.*

$$\frac{\gamma}{\gamma_0(a)} \quad \frac{\forall x\ F(x)}{F(a)} \quad \frac{\neg\exists x\ F(x)x}{\neg F(a)}$$

Tableau 6.8. *Règles d'instanciation universelle.*

6.1.2.2. *Règle d'instanciation existentielle*

La règle d'instanciation existentielle (appelée également règle δ) représente la satisfaction d'une formule quantifiée existentiellement et spécifie que si une formule quantifiée existentiellement est vraie alors elle est vrai pour une instance d'élément de l'univers de Herbrand. Cette règle est représenté par :

$$\frac{\delta}{\delta_0(a)}$$

Tableau 6.9. *Règle delta.*

Les règles d'instanciation existentielle pour la logique des prédicats sont les suivantes, où a est une nouvelle constante sur la branche :

$$\frac{\delta}{\delta_0(a)} \quad \frac{\exists x\ F(x)}{F(a)} \quad \frac{\neg\forall x\ F(x)}{\neg F(a)}$$

Tableau 6.10. *Règles d'instanciation existentielle.*

Les règles d'instanciation universelle peuvent s'appliquer indéfiniment alors que les règles d'instanciation existentielle ne s'appliquent qu'une fois. L'application des

règles d'instanciation universelle et d'instanciation existentielle dans nimporte quel ordre peut éventuellement conduire à l'écriture d'une infinité d'instances des formules qui ne sont pas des propositions contradictoires (arbre infini). Afin de faire apparaître des contradictions, les règles de prolongation, de ramification, de négation et de double négation et d'instanciation existentielle sont appliquées avant la règle d'instanciation universelle et la même instance est utilisée dans les règles existentielles et universelles. L'exemple suivant illustre la méthode des tableaux sémantiques pour la logique des prédicats.

EXEMPLE 6.3.– Soit la formule $F = \exists x\ (p(x) \wedge q(x)) \rightarrow (\exists x\ p(x) \wedge \exists x\ q(x))$, le tableau sémantique pour $\neg F$ est donné par la figure 6.5. La règle d'instanciation existentielle sur la sous-formule étiquetant le noeud (2) avec la constante a non encore utilisée dans aucune des branches produit les noeuds (6) et (8), étiquetés par $p(a)$ ainsi que les noeuds (7) et (9), étiquetés par $q(a)$. La règle d'instanciation universelle sur la sous-formule étiquetant le noeud (4) avec la constante a produit le noeud (10) étiqueté par $\neg p(a)$, cette branche est fermée. La règle d'instanciation universelle sur la sous-formule étiquetant le noeud (5) avec la constante a produit le noeud (11) étiqueté par $\neg q(a)$, cette branche est fermée. Le tableau sémantique pour $\neg F$ est fermé donc $\neg F$ est incohérente et F est une tautologie.

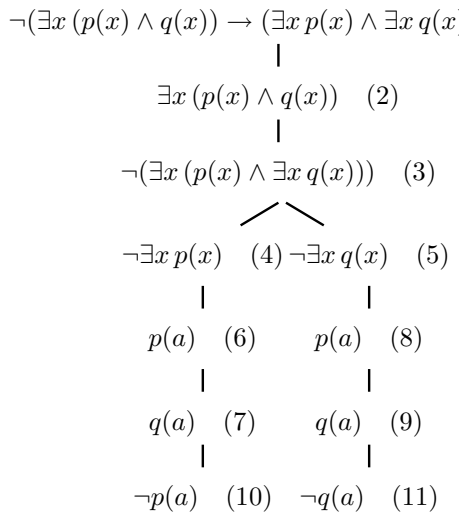

Figure 6.5. *Tableau sémantique pour la formule* $\neg F = \neg(\exists x\ (p(x) \wedge q(x)) \rightarrow (\exists x\ p(x) \wedge \exists x\ q(x)))$.

Comme dans le cas de la logique propositionnelle, la notion de preuve est définie par :

DÉFINITION 6.10.– *Soit F une formule de la logique des prédicats, on dit que F a une preuve par tableaux sémantiques si le tableau sémantique pour $\neg F$ est fermé.*

Les résultats d'adéquation et de complétude sont données par le théorème suivant :

THÉORÈME.– *Soit F une formule de la logique des prédicats, F est une tautologie si et seulement si F a une preuve par tableaux sémantiques.*

La méthode des tableaux sémantiques pour la logique des prédicats n'est pas décidable dans le cas général, car il est possible de construire des arbres infinis, cependant, d'un point de vue pratique, des démonstrateurs de théorèmes basés sur les tableaux sémantiques pour la logique du premier ordre sans égalité ont été développés par exemple, les démonstrateurs LeanTaP [BEC 95] et 3TAP [BEC 96].

6.1.3. *Tableaux sémantiques en logique modale*

L'origine de la méthode des tableaux sémantiques pour la logique modale [GOR 03] est la rencontre des travaux sur l'adaptation des séquents de Gentzen [SZA 69] (Voir 6.2.2) à la logique modale et des travaux de Beth[BET 55] et de Kripke [KRI 59] sur la logique propositionnelle.

Parmi les méthodes de déduction par tableaux sémantiques, la méthode utilisant les formules préfixées est présentée en raison de sa simplicité et de son efficacité [MAS 94, MAS 00a, GOR 99, FIT 98]. Les formules sont préfixées par des étiquettes qui "nomment", en quelque sorte, le monde dans lequel les formules sont supposées satisfaites. Les règles d'expansion tiennent compte des formules et des étiquettes et reflètent la sémantique des mondes possibles.

DÉFINITION.– *Un préfixe est une suite finie d'entiers positifs, notée σ, et une formule préfixée est une expression de la forme $\sigma\ F$ où σ est un préfixe et F une formule de la logique modale.*

On note $\sigma_0.\sigma_1$ le préfixe obtenu par la concaténation des préfixes σ_0 et σ_1, soit n un entier, on note $\sigma.n$ la suite σ suivie de n et $\sigma.n$ nomme un monde accessible par l'un des mondes nommé par σ.

Pour construire une preuve de F par tableaux sémantiques, on construit un arbre dont la racine est étiquetée par la formule préfixée $1\ \neg F$, ce qui signifie que le monde nommé par 1 ne satisfait pas la formule F, et les branches de l'arbre sont construites par l'application des règles d'expansion suivantes.

6.1.3.1. *Règle de prolongation*

Pour tout préfixe σ, la règle de prolongation représente la satisfaction de la formule, notée $\sigma\ \alpha = \sigma\ \alpha_1\ \wedge\ \sigma\ \alpha_2$, composée de la conjonction de deux sous-formules préfixées $\sigma\ \alpha_1$ et $\sigma\ \alpha_2$, et signifie que si le monde nommé par σ satisfait α alors il satisfait α_1 et α_2. Les règles de prolongation de la logique modale sont représentées par la table 6.11.

6.1.3.2. *Règle de ramification*

Pour tout préfixe σ, la règle de ramification représente la satisfaction de la formule, notée $\sigma\ \beta = \sigma\ \beta_1\ \vee\ \sigma\ \beta_2$, composée de la disjonction de deux sous-formules

$$\frac{\sigma\ \alpha}{\begin{array}{l}\sigma\ \alpha_1\\ \sigma\ \alpha_2\end{array}} \qquad \frac{\sigma\ x\wedge y}{\begin{array}{l}\sigma\ x\\ \sigma\ y\end{array}} \qquad \frac{\sigma\ \neg(x\vee y)}{\begin{array}{l}\sigma\ \neg x\\ \sigma\ \neg y\end{array}} \qquad \frac{\sigma\ \neg(x\rightarrow y)}{\begin{array}{l}\sigma\ x\\ \sigma\ \neg y\end{array}} \qquad \frac{\sigma\ x\leftrightarrow y}{\begin{array}{l}\sigma\ x\rightarrow y\\ \sigma\ x\rightarrow y\end{array}}$$

Tableau 6.11. *Règles de prolongation pour la logique modale.*

préfixées $\sigma\ \beta_1$ et $\sigma\ \beta_2$ et signifie que si le monde nommé par σ satisfait β alors il satisfait β_1 ou β_2. Les règles de ramification de la logique modale sont représentées par la figure 6.12.

$$\frac{\sigma\ \beta}{\sigma\ \beta_1 \mid \sigma\ \beta_2} \qquad \frac{\sigma\ x\vee y}{\sigma\ x \mid \sigma\ y} \qquad \frac{\sigma\ \neg(x\wedge y)}{\sigma\ \neg x \mid \sigma\ \neg y}$$

$$\frac{\sigma\ \neg(x\rightarrow y)}{\sigma\ \neg x \mid \sigma\ y} \qquad \frac{\sigma\ \neg(x\leftrightarrow y)}{\sigma\ \neg(x\rightarrow y) \mid \sigma\ \neg(y\rightarrow x)}$$

Tableau 6.12. *Règles de ramification pour la logique propositionnelle.*

6.1.3.3. *Règle de double négation*

Pour tout préfixe σ, la règle de double négation représente la satisfaction de la formule, notée $\sigma\ \neg\neg\alpha = \sigma\ \alpha$ composée de la double négation d'une sous-formule propositionnelle α et signifie que si le monde nommé par σ satisfait $\sigma\ \neg\neg\alpha$ alors il satisfait α. La règle de double négation est représentée par la figure 6.13 :

$$\frac{\sigma\ \neg\neg\alpha}{\sigma\ \alpha}$$

Tableau 6.13. *Règle de double négation.* .

De nouvelles règles d'expansion sont introduites pour les opérateurs modaux $\Diamond$ et $\Box$.

6.1.3.4. *Règle de possibilité*

La sémantique de l'opérateur $\Diamond$ spécifie que pour la formule $\sigma\ \Diamond\alpha$, le monde nommé par σ satisfait $\Diamond\alpha$ et il existe au moins un monde accéssible à partir du monde dont le nom est σ qui satisfait α. D'aprés la définition des préfixes, le monde accéssible

à partir de σ a un nom de la forme $\sigma.n$ à ondition que ce nom n'ait jamais été utilisé préalablement, ce qui se traduit par la règle représentée par la Table 6.14 :

$$\frac{\sigma\ \Diamond\alpha}{\sigma.n\ \alpha} \qquad \frac{\sigma\ \neg\ \Box\alpha}{\sigma.n\ \neg\alpha}$$

Tableau 6.14. *Règles de possibilité.*

6.1.3.5. *Règle de nécessité*

La sémantique de l'opérateur $\Box$ règle despécifie que pour la formule $\sigma\ \Box\alpha$, le monde nommé par σ satisfait $\Box\alpha$ et tous les mondes accéssibles à partir du monde dont le nom est σ satisfont α. D'aprés la définition des préfixes, si le préfixe $\sigma.n$ a déjà été utilisé il est le nom d'un monde accéssible à partir du monde dont le nom est σ, ce qui se traduit par la règle et représentée par la table 6.15 :

$$\frac{\sigma\ \Box\alpha}{\sigma.n\ \alpha} \qquad \frac{\sigma\ \neg\ \Diamond\alpha}{\sigma.n\ \neg\alpha}$$

Tableau 6.15. *Règles de nécessité.*

Les définitions données en 6.2.1 pour les formules propositionnelles s'étendent aisément aux formules de la logique modale.

DÉFINITION 6.11.– *Soit F une formule de la logique modale. Une branche d'un tableau est fermée (ou close) si les formules préfixées $\sigma\ F$ et $\sigma\ \neg F$ apparaissent dans la branche.*

DÉFINITION 6.12.– *Un tableau sémantique est fermé (ou clos) si toutes ses branches sont fermées.*

Comme dans le cas de la logique propositionnelle, la notion de preuve est définie par :

DÉFINITION 6.13.– *Soit F une formule de la logique modale. On dit que F a une preuve par tableaux sémantiques si le tableau sémantique pour $1\ \neg F$ est fermé.*

L'exemple suivant illustre la méthode des tableaux sémantiques pour la logique modale.

EXEMPLE 6.4.– Soit F la formule de la logique modale $F = \Box(p \wedge q) \rightarrow (\Box p \wedge \Box q)$ Le tableau sémantique pour la formule préfixée $1\ \neg F$ est donné par la table 6.6. La racine de l'arbre, c'est à dire, le noeud (1), est étiquetée par la formule préfixée $1\ \neg F$. La règle de prolongation produit les noeuds (2) et (3). La règle de disjonction est utilisée avant la règle de nécessité. La règle de disjonction sur la sous-formule

étiquetant le noeud (3) produit les noeuds (4) et 9. La règle de possibilité sur le noeud (5) produit le noeud (6) avec un nouveau préfixe 1.1 sur la branche. La règle de nécessité sur le noeud (2) produit le noeud (6) avec le préfixe 1.1 a déja été utilisé sur la branche. La règle de prolongation sur le noeud (6) produit les noeuds (7) et (8). La règle de possibilité sur le noeud (9) produit le noeud (10) avec un nouveau préfixe 1.1 sur la branche. La règle de nécessité sur le noeud (2) produit le noeud (11) avec le préfixe 1.1 a déja été utilisé sur la branche. La règle de prolongation sur le noeud (11) produit les noeuds (12) et (13). Les feuilles de l'arbre, les noeuds (8) et (13) sont étiquetés par des formules propositionnelles atomiques et plus aucune règle d'expansion ne peut s'appliquer. Le tableau sémantique est fermé car sur la branche gauche les noeuds (5) et (7) donnent respectivement les formules préfixées $1.1\ \neg p$ et $1.1\ p$ et sur la branche droite les noeuds (10) et (13) donnent respectivement les formules préfixées $1.1\ \neg q$ et $1.1\ q$. Par conséquent la formule F est une tautologie.

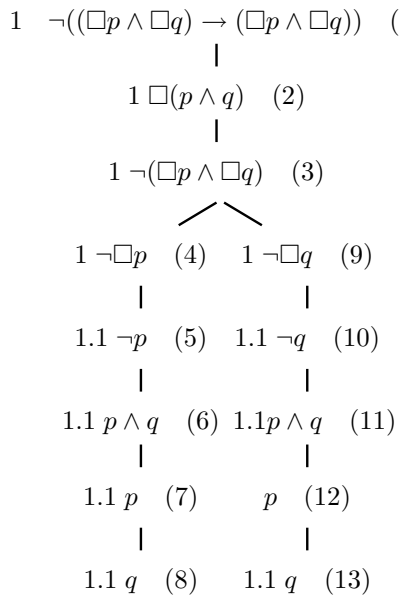

Figure 6.6. *Tableau sémantique pour la formule* $1\neg F = \neg(\Box(p \wedge q) \rightarrow (\Box p \wedge \Box q))$.

REMARQUE.– Les règles de prolongation, de ramification et de possibilité sont utilisées avant la règle de nécessité afin de faire apparaître des contradictions.

La méthode des tableaux sémantiques présentée dans cette section est constituée des règles de prolongation, ramification, nécessité et de possibilié et est conçue pour

le système K (voir C.2). D'autres règles peuvent être utilisées pour obtenir une méthode des tableaux sémantiques efficace pour les autres systèmes formels T, $S4$, $S5$ [MAS 94, FIT 98, GOR 99]

6.1.3.6. *Autres règles de nécessité*

Soit les préfixes σ et $\sigma.n$ apparaissant dans la branche d'un tableau, de nouvelles règles d'expansion sont définies dans la table 6.16.

T	$\frac{\sigma\ \Box\alpha}{\sigma\ \alpha}$	$\frac{\sigma\ \neg\Diamond\alpha}{\sigma\ \neg\alpha}$
D	$\frac{\sigma\ \Box\alpha}{\sigma\ \Diamond\alpha}$	$\frac{\sigma\ \neg\Diamond\alpha}{\sigma\ \neg\Box\alpha}$
B	$\frac{\sigma.n\ \Box\alpha}{\sigma\ \alpha}$	$\frac{\sigma.n\ \neg\Diamond\alpha}{\sigma\ \neg\alpha}$
4	$\frac{\sigma\ \Box\alpha}{\sigma.n\ \Box\alpha}$	$\frac{\sigma\ \neg\Diamond\alpha}{\sigma.n\ \neg\Diamond\alpha}$
$4r$	$\frac{\sigma.n\ \Box\alpha}{\sigma\ \Box\alpha}$	$\frac{\sigma.n\ \neg\Diamond\alpha}{\sigma\ \neg\Diamond\alpha}$
D	$\frac{\sigma\ \Box\alpha}{\sigma\ \Diamond\alpha}$	$\frac{\sigma\ \neg\Diamond\alpha}{\sigma\ \neg\Box\alpha}$

Tableau 6.16. *Autres règles d'expansion pour la logique modale.*

La table 6.17 donne les règles d'expansion qui peuvent être ajoutés aux systèmes formels pour rendre la méthode des tableaux plus efficace.

$système\ formel$	$règle$
D	D
T	T
$K4$	4
B	$B,\ 4$
$S4$	$T,\ 4$
$S5$	$T,\ 4,\ 4r$

Tableau 6.17. *Règles supplémentaires à ajouter aux systèmes formels.*

Les résultats d'adéquation et de complétude s'étendent aisément au cas de formules préfixées de la logique modale :

THÉORÈME 6.4.– *Soit F une formule de la logique modale, F est une tautologie si et seulement si F a une preuve par tableaux sémantiques.*

La méthode des tableaux sémantiques préfixés pour la logique modale propositionnelle est décidable, elle s'étend à la logique modale du premier ordre [FIT 98].

La méthode des tableaux sémantiques pour la logique modale, comme le système SST[MAS 00b], est P-SPACE-complet pour les logiques modales K, KD, T, KB, KDB, $S4$ et NP-TIME complète pour les logiques modales $S5$ et $KD45$. D'un point de vue pratique, des démonstrateurs de théorèmes basés sur les tableaux sémantiques ont été développés soit pour des logiques temporelles utilisant des opérateurs modaux, [BES 89], [ENJ 90], [BAL 97a], soit de façon plus générale pour des logiques modales comme LoTREC[Far 01], TWB[ABA 03], KSAT[GIU 96], FaCT [HOR 98].

6.2. Calcul des séquents

Nous présentons le calcul des séquents pour la déduction automatique en logique propositionnelle, logique des prédicats et logique modale et en logique intuitionniste. Ces logiques ainsi que les notations qui s'y réfèrent sont présentées, respectivement dans les annexes A, B, C et D.

Le calcul des séquents est un système de déduction introduit par G. Gentzen [SZA 69]. Ce système repose sur l'idée suivante, de même qu'à toute formule incohérente correspond sa négation qui est une formule cohérente, à tout ensemble de formules incohérent correspond un séquent qui représente en quelque sorte la négation d'un ensemble de formules incohérent. Un séquent exprime intuitivement qu'une conjonction de formules implique une disjonction de formules [LAD 51]. Le calcul des séquents permet de faire des déductions directes ou des réfutations comme dans le cas des tableaux sémantiques.

6.2.1. *Calcul des séquents propositionnel*

DÉFINITION 6.14.– *On appelle séquent une expression de la forme $\Gamma \Rightarrow \Delta$[2] où Γ et Δ sont des suites éventuellement vides de formules propositionnelles. On appelle Γ l'antécédent du séquent et Δ le conséquent du séquent.*

Intuitivement, si $\Gamma = X_1, \cdots X_n$ et $\Delta = Y_1 \cdots Y_m$, où $m > 0$, et $n > 0$ sont des entiers, le séquent $\Gamma \Rightarrow \Delta$ peut être vu comme la formule propositionnelle $X_1 \wedge \cdots \wedge X_n \rightarrow Y_1 \vee \cdots \vee Y_n$.

2. Pour éviter toute confusion le symbole $\Rightarrow$ est préféré au symbole $\vdash$.

Le calcul des séquents est un système axiomatique, où le symbole $\Rightarrow$ est un symbole primitif, défini à partir d'axiomes et de règles de déduction.

DÉFINITION 6.15.– *Un axiome est un séquent dont l'antécédent et le conséquent ont en commun au moins une proposition atomique.*

Les règles de déduction pour les séquents s'écrivent sous la forme $\dfrac{S_1, \cdots, S_n}{S}$ où $S_1, \cdots, S_n$ est une suite éventuellement vide de séquents qui sont appelés séquents prémisses et où S est un séquent unique appelé séquent conclusion. Le nom de la règle de déduction figure à gauche de la ligne horizontale lorsque le connecteur figure dans l'antécédent du séquent conclusion et à droite de la ligne horizontale lorsque le connecteur figure dans le conséquent du séquent conclusion.

Comme l'antécédent et le conséquent d'un séquent sont constitués d'une suite ordonnée de formules, des règles structurales sont définies qui permettent de permuter, de supprimer et de ventiler des formules dans l'antécédent et le conséquent.

$$\text{règles de permutation} : \quad \frac{\Gamma \Rightarrow \Delta, X}{\Gamma \Rightarrow X, \Delta} \qquad \frac{\Gamma \Rightarrow \Delta}{\Gamma, X \Rightarrow \Delta}$$

$$\text{règles de contraction} : \quad \frac{\Gamma \Rightarrow \Delta, X, X}{\Gamma \Rightarrow \Delta, X} \qquad \frac{X, X, \Gamma \Rightarrow \Delta}{X, \Gamma \Rightarrow \Delta}$$

$$\text{règles d'atténuation} : \quad \frac{\Gamma \Rightarrow \Delta}{\Gamma \Rightarrow \Delta, X} \qquad \frac{\Gamma \Rightarrow \Delta}{X, \Gamma \Rightarrow \Delta}$$

D'autres règles sont introduites qui correspondent à l'introduction de connecteurs propositionnels :

$$\frac{\Gamma \Rightarrow \Delta, X}{\neg X, \Gamma \Rightarrow \Delta}\neg\ \Rightarrow \qquad \frac{X, \Gamma \Rightarrow \Delta}{\Gamma \Rightarrow \Delta, \neg X}\Rightarrow\ \neg$$

$$\frac{\Gamma \Rightarrow \Delta, X \quad Y, \Gamma \Rightarrow \Delta}{X \to Y, \Gamma \Rightarrow \Delta}\to\Rightarrow \qquad \frac{X, \Gamma \Rightarrow \Delta, Y}{\Gamma \Rightarrow \Delta, X \to Y}\Rightarrow\to$$

$$\frac{X, Y, \Gamma \Rightarrow \Delta \quad Y, \Gamma \Rightarrow \Delta, X, Y}{X \leftrightarrow Y, \Gamma \Rightarrow \Delta}\leftrightarrow\Rightarrow \qquad \frac{X, \Gamma \Rightarrow \Delta, Y \quad Y, \Gamma \Rightarrow \Delta, X}{\Gamma \Rightarrow \Delta, X \leftrightarrow Y}\Rightarrow\leftrightarrow$$

$$\frac{X_1, \ldots, X_n, \Gamma \Rightarrow \Delta}{(X_1 \wedge \cdots \wedge X_n), \Gamma \Rightarrow \Delta}\wedge\ \Rightarrow \qquad \frac{\Gamma \Rightarrow \Delta, X_1 \ \cdots\ \Gamma \Rightarrow \Delta, X_n}{\Gamma \Rightarrow \Delta, (X_1 \wedge \cdots \wedge X_n)}\Rightarrow\ \wedge$$

$$\frac{X_1, \Gamma \Rightarrow \Delta, \ \cdots\ X_n, \Gamma \Rightarrow \Delta}{(X_1 \vee \cdots \vee X_n), \Gamma \Rightarrow \Delta}\vee\ \Rightarrow \qquad \frac{\Gamma \Rightarrow \Delta, X_1, \ldots, X_n}{\Gamma \Rightarrow \Delta, (X_1 \vee \cdots \vee X_n)}\Rightarrow\ \vee$$

La règle pour les axiomes :

$$\frac{}{A \Rightarrow \Delta, A}$$

Les règle pour les constantes $\top$ et $\bot$:

$$\frac{}{\bot \Rightarrow \Delta}\bot\ \Rightarrow \qquad \frac{\Gamma \Rightarrow \Delta}{\Gamma \Rightarrow \Delta}\Rightarrow\ \bot$$

$$\frac{\Gamma \Rightarrow \Delta}{\Gamma, \top \Rightarrow \Delta}\top\ \Rightarrow \qquad \frac{}{\Gamma \Rightarrow \top}\Rightarrow\ \top$$

Et enfin une règle d'élimination ou de coupure :

$$\frac{\Gamma \Rightarrow \Delta, X \quad X, \Gamma' \Rightarrow \Delta'}{\Gamma, \Gamma' \Rightarrow \Delta, \Delta'}$$

D'un point de vue axiomatique, la déduction d'un séquent S séquent s'effectue par la construction d'un arbre, appelé arbre de dérivation, dont :

– la racine est étiquetée par le séquent S ;

– les noeuds sont étiquetés par des séquents ;

– les descendants de noeuds sont les séquents premisses obtenus par l'application, à rebours, des règles de déduction.

DÉFINITION 6.16.– *Une dérivation dans le calcul des séquents est un arbre de dérivation où chaque feuille est étiquetée par un séquent qui est un axiome.*

Une dérivation dans le calcul des séquents est une déduction du séquent qui étiquette la racine de l'arbre l'arbre de dérivation à partir des séquents qui étiquettent les feuilles de l'arbre de dérivation.

Il existe une déduction d'une formule F à partir d'un ensemble de formules $\mathcal{F}$ lorsque le séquent $\mathcal{F} \Rightarrow F$ est dérivable dans le calcul des séquents. Lorsque l'ensemble de formules de l'antécédent du séquent est vide, le séquent $\Rightarrow F$ permet de déduire que la formule F est un théorème.

EXEMPLE 6.5.– Soit la formule propositionnelle $F = \neg p \rightarrow (p \rightarrow q)$, l'arbre de dérivation construit à partir du séquent $\Rightarrow \neg p \rightarrow (p \rightarrow q)$ est le suivant :

$$\dfrac{\dfrac{\dfrac{p \Rightarrow\ q, p\ axiome}{p, \neg p \Rightarrow\ q}\neg\ \Rightarrow}{\neg p \Rightarrow\ p \rightarrow q}\Rightarrow \rightarrow}{\Rightarrow\ \neg p \rightarrow (p \rightarrow q)}\Rightarrow \rightarrow$$

Il permet de montrer que la formule F est un théorème.

D'un point de vue sémantique, la notion de validité d'un séquent est introduite comme suit :

DÉFINITION 6.17.– *Un séquent S est validesi toute interprétation de la logique propositionnelle, I qui satisfait tous les antécédents de S satisfait au moins un conséquent de S.*

Dans la déduction naturelle, les règles de déduction (par exemple Modus Ponens) transmettent la validité uniquement des prémisses à la conclusion, en revanche, les règles d'introduction des connecteurs du calcul des séquents sont réversibles, c'est à dire qu'elles transmettent la validité de la conclusion aux prémisses, plus formellement :

PROPOSITION 6.3.– *Pour toutes les règles de déduction du calcul des séquents, à l'exception de la règle pour les axiomes, les prémisses d'un séquent sont valides si et seulement si la conclusion est valide.*

L'adéquation et la complétude du calcul des séquents en logique propositionnelle sont données par le théoréme suivant :

THÉORÈME 6.5.– *Un séquent est valide si et seulement si il est dérivable.*

Le calcul des séquents permet de trouver de façon correcte et complète les conséquences logiques d'un ensemble de formules propositionnelles :

THÉORÈME.– *Soit F une formule propositionnelle et $\mathcal{F}$ un ensemble de formules propositionnelles. Le séquent $\mathcal{F} \Rightarrow F$ est dérivable si et seulement si $\mathcal{F} \models F$*

Pour montrer qu'une formule F est une tautologie un corollaire du théorème précédent assure qu'une formule propositionnelle F est une tautologie si et seulement si le séquent $\Rightarrow F$ est valide. Pour montrer par réfutation qu'une formule F est incohérente il suffit de monter que le séquent $\Rightarrow \neg F$ est valide.

PROPOSITION 6.4.– *Soit $X_1, \cdots, X_n, Y_1, \cdots, Y_n$ des formules propositionnelles, le séquent $X_1, \cdots, X_n \Rightarrow Y_1, \cdots, Y_n$ est valide si et seulement si la formule $X_1 \wedge \cdots \wedge X_n \rightarrow Y_1 \vee \cdots \vee Y_n$ est une tautologie.*

EXEMPLE.– Soit l'ensemble de formules $\mathcal{F} = \{(p \vee q) \wedge (\neg p \rightarrow r),\ q \rightarrow \neg r,\ \neg p\}$. Cet ensemble de formules est transformé en la formule équivalente, $F = ((p \vee q) \wedge (\neg p \rightarrow r)) \wedge (q \rightarrow \neg r) \wedge (\neg p)$. En utilisant la réfutation nous construisons l'arbre de preuve correspondant au séquent $\Rightarrow \neg F = \neg((p \vee q) \wedge (\neg p \rightarrow r) \wedge (q \rightarrow \neg r) \wedge (\neg p))$ comme suit :

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{p \Rightarrow\ \neg(\neg p \rightarrow q), \neg(q \rightarrow \neg r), p\ axiome \qquad q \Rightarrow\ \neg(\neg p \rightarrow q), \neg(q \rightarrow \neg r), p}{(p \vee q) \Rightarrow\ \neg(\neg p \rightarrow q), \neg(q \rightarrow \neg r), p}\vee\ \Rightarrow}{\Rightarrow\ \neg(p \vee q), \neg(\neg p \rightarrow q), \neg(q \rightarrow \neg r), p}\Rightarrow\ \neg}{\Rightarrow\ \neg(p \vee q) \vee \neg(\neg p \rightarrow q) \vee \neg(q \rightarrow \neg r) \vee p}\Rightarrow\ \vee}{\Rightarrow\ \neg((p \vee q) \wedge (\neg p \rightarrow q) \wedge (q \rightarrow \neg r) \wedge (\neg p))}\Rightarrow\ \neg$$

L'arbre de preuve correspondant au séquent $q \Rightarrow\ \neg(\neg p \rightarrow q), \neg(q \rightarrow \neg r), p$ est le suivant :

$$\dfrac{\dfrac{q \Rightarrow\ \neg(\neg p \rightarrow q), p, q\ axiome \qquad \dfrac{\dfrac{\dfrac{p, q \Rightarrow\ p, r\ axiome}{q \Rightarrow\ p, r, \neg p}\Rightarrow\ \neg \qquad \dfrac{r, q \Rightarrow\ p, r\ axiome}{\neg p \rightarrow q, q \Rightarrow\ p, r}\Rightarrow \rightarrow}{\neg p \rightarrow q, \neg r, q \Rightarrow\ p}\neg\ \Rightarrow}{\neg r, q \Rightarrow\ \neg(\neg p \rightarrow q), p}\Rightarrow\ \neg}{q \rightarrow \neg r, q \Rightarrow\ \neg(\neg p \rightarrow q), p}\rightarrow \Rightarrow}{q \Rightarrow\ \neg(\neg p \rightarrow q), \neg(q \rightarrow \neg r), p}\Rightarrow\ \neg$$

Toutes les feuilles de l'arbre de preuve sont étiquetées par un axiome, la formule $\neg F$ correspond à un séquent valide, c'est une tautologie donc la formule F est incohérente et par conséquent, l'ensemble de formules $\mathcal{F}$ est incohérent.

De plus, Gentzen a montré que l'on peut se passer de la règle de coupure :

THÉORÈME 6.6.– *Toute preuve par les séquents peut se transformer en une preuve qui n'utilise pas la règle de coupure.*

Pour plus de détails sur le calcul des séquents en logique propositionnelle consulter [LAF 92, GOR 03].

6.2.2. *Calcul des séquents en logique des prédicats*

Le calcul des séquents s'étend à la logique des prédicats comme suit.

DÉFINITION 6.18.– *On appelle séquent une expression de la forme* $\Gamma \Rightarrow \Delta$ *où* Γ *et* Δ *sont des suites éventuellement vides de formules de la logique des prédicats.*

Les règles de déduction supplémentaires suivantes pour les quantificateurs universels et existentiels sont ajoutées à l'ensemble de règles de déduction propositionnelles. Soit b et t des termes de la logique des prédicats,

$$\frac{\forall x\ A(x), A(t), \Gamma \Rightarrow \Delta}{\forall x\ A(x), \Gamma \Rightarrow} \forall\ \Rightarrow \qquad \frac{\Gamma \Rightarrow \Delta, A(b)}{\Gamma \Rightarrow \Delta, \forall x\ A(x)} \Rightarrow\ \forall$$

$$\frac{A(b), \Gamma \Rightarrow \Delta}{\exists\, xA(x), \Gamma \Rightarrow \Delta} \exists\ \Rightarrow \qquad \frac{\Gamma \Rightarrow \Delta, \exists\, xA(x), A(t)}{\Gamma \Rightarrow \Delta, \exists\, xA(x)} \Rightarrow\ \exists$$

Dans les Les règles $\Rightarrow\ \forall$ et $\exists\ \Rightarrow$, la variable x n'est pas une variable libre dans Γ et Δ. Les règles $\forall\ \Rightarrow$ et $\Rightarrow\ \exists$ sont des règles ré-utilisables, cela s'exprime par la répétition de la formule $\forall x\ A(x)$ dans l'antécédent du séquent prémisse de la règle $\forall\ \Rightarrow$ et par la répétition de la formule $\exists\ xA(x)$ dans le conséquent du séquent prémisse de la règle $\Rightarrow\ \exists$.

Les règles $\Rightarrow\ \forall$ et $\exists\ \Rightarrow$ correspondent aux règles δ de la méthode des tableaux sémantiques 6.1.2, le terme b est un terme inédit et comme dans la méthode des tableaux elles ne s'appliquent qu'une fois, alors que les règles Les règles $\forall\ \Rightarrow$ et $\Rightarrow\ \exists$ correspondent aux règles γ de la méthode des tableaux sémantiques, le terme t n'est pas un terme inédit et comme dans la méthode des tableaux elles peuvent s'appliquer indéfiniment. Afin d'aboutir rapidement à des axiomes dans l'arbre de dérivation, les règles $\Rightarrow\ \forall$ et $\exists\ \Rightarrow$ s'appliquent avant les règles $\forall\ \Rightarrow$ et $\Rightarrow\ \exists$ sur une même instance.

Les notions de dérivation, de validité d'un séquent sétendent à la logique des prédicats et l'adéquation et la complétude du calcul des séquents en logique des prédicats sont données par le théoréme suivant :

THÉORÈME 6.7.– *Un séquent est valide si et seulement si il est dérivable.*

Le calcul des séquents permet de trouver de façon correcte et complète les conséquences logiques d'un ensemble de formules de la logique des prédicats :

THÉORÈME 6.8.– *Soit* F *une formule de la logique des prédicats et* $\mathcal{F}$ *un ensemble de formules de la logique des prédicats. Le séquent* $\mathcal{F} \Rightarrow F$ *est dérivable si et seulement si* $\mathcal{F} \models F$

Pour montrer qu'une formule F est une tautologie un corollaire du théorème précédent assure qu'une formule de la logique des prédicats F est une tautologie si et seulement si le séquent $\Rightarrow F$ est valide.

L'exemple suivant illustre le calcul des séquents pour la logique des prédicats.

EXEMPLE.– Soit la formule de la logique des prédicats $F = \exists x\ (p(x) \wedge q(x)) \rightarrow (\exists x\ p(x) \wedge \exists x\ q(x))$, l'arbre de dérivation construit à partir du séquent $\Rightarrow \exists x\ (p(x) \wedge q(x)) \rightarrow (\exists x\ p(x) \wedge \exists x\ q(x))$ est le suivant :

$$\dfrac{\dfrac{\dfrac{p(a) \wedge q(a) \Rightarrow \exists x\ p(x) \qquad p(a) \wedge q(a) \Rightarrow \exists x\ q(x)}{p(a) \wedge q(a) \Rightarrow (\exists x\ p(x) \wedge \exists x\ q(x))} \Rightarrow\ \wedge}{\exists x\ (p(x) \wedge q(x)) \Rightarrow (\exists x\ p(x) \wedge \exists x\ q(x))} \exists\ \Rightarrow}{\Rightarrow\ \exists x\ (p(x) \wedge q(x)) \rightarrow (\exists x\ p(x) \wedge \exists x\ q(x))} \Rightarrow \rightarrow$$

Le séquent $p(a) \wedge q(a) \Rightarrow \exists x\ p(x)$ donne l'arbre de dérivation suivant :

$$\dfrac{\dfrac{p(a), q(a) \Rightarrow \exists x\ p(x), p(a) \quad axiome}{p(a), q(a) \Rightarrow \exists x\ p(x)} \Rightarrow\ \exists}{p(a) \wedge q(a) \Rightarrow \exists x\ p(x)} \wedge\ \Rightarrow$$

et le séquent $p(a) \wedge q(a) \Rightarrow \exists x\ q(x)$ donne l'arbre de dérivation suivant :

$$\dfrac{\dfrac{p(a), q(a) \Rightarrow \exists x\ q(x), q(a) \quad axiome}{p(a), q(a) \Rightarrow \exists x\ q(x)} \Rightarrow\ \exists}{p(a) \wedge q(a) \Rightarrow \exists x\ q(x)} \wedge\ \Rightarrow$$

La règle $\exists\ \Rightarrow$ est appliquée avant la règle $\Rightarrow\ \exists$ sur la même instance a qui est une constante de la logique des prédicats. Le séquent est dérivable donc valide et la formule F est un théorème donc une tautologie.

Pour plus de détails sur le calcul des séquents en logique des prédicats consulter [AVR 93].

6.2.3. *Le calcul des séquents pour la logique modale*

Le calcul des séquents de Gentzen s'étend également à la logique modale.

DÉFINITION.– *On appelle séquent une expression de la forme $\Gamma \Rightarrow \Delta$ où Γ et Δ sont des suites éventuellement vides de formules de la logique modale.*

Des règles de déduction supplémentaires pour les modalités $\Box$ et $\Diamond$ sont ajoutées à l'ensemble de règles de déduction propositionnelles.

Pour la logique modale K les règles π sont ajoutées à l'ensemble des règles :

$$\dfrac{\Gamma^{\star}, X \Rightarrow \Delta^{\star}}{\Diamond X \Rightarrow \Delta} \Diamond\ \Rightarrow$$

$$\frac{\Gamma^\star \Rightarrow X, \Delta^\star}{\Gamma \Rightarrow \Box X, \Delta} \Rightarrow \Box$$

avec $\Gamma^\star = \{X, \Box X \in \Gamma\}$ et $\Delta^\star = \{X, \Diamond X \in \Delta\}$. Pour la logique modale T les règles ν sont ajoutées :

$$\frac{\Gamma, X \Rightarrow \Delta}{\Gamma, \Box X \Rightarrow \Delta} \Box \Rightarrow$$

$$\frac{\Gamma \Rightarrow \Diamond X, \Delta}{\Gamma \Rightarrow X, \Delta} \Rightarrow \Diamond$$

avec $\Gamma^\star = \{X, \Box X \in \Gamma\}$ et $\Delta^\star = \{X, \Diamond X \in \Delta\}$. Pour la logique modale $S4$ les règles π et ν avec $\Gamma^\star = \{\Box X, \Box X \in \Gamma\}$ et $\Delta^\star = \{\Diamond X, \Diamond X \in \Delta\}$ sont ajoutées en l'ensemble des règles pour le calcul des séquents en logique propositionnelle.

Les notions de dérivation, validité, d'adéquation et de complétude s'étendent au calcul des séquents en logique modale. La complexité des procédures de décision utilisant le calcul des séquents en logique modale est généralement P-space complet, une procédure de décision utilisant le calcul des séquents en logique modale $S4$ $O(n^4)$-space complète a été donnée par Ladner [LAD 77].

L'exemple suivant illustre le calcul des séquents pour la logique K.

EXEMPLE 6.6.– Soit la formule $F = \Box(p \wedge q) \rightarrow (\Box p \wedge \Box q)$ de la logique modale K, l'arbre de dérivation construit à partir du séquent $\Rightarrow F$ est le suivant :

$$\dfrac{\dfrac{\dfrac{\dfrac{p, q \Rightarrow p \quad axiome}{p \wedge q \Rightarrow p} \wedge \Rightarrow}{\Box(p \wedge q) \Rightarrow \Box p} \Rightarrow \Box \qquad \dfrac{\dfrac{\dfrac{p, q \Rightarrow q \quad axiome}{p \wedge q \Rightarrow q} \wedge \Rightarrow}{\Box(p \wedge q) \Rightarrow \Box q} \Rightarrow \Box}{\Box(p \wedge q) \Rightarrow \Box p \wedge \Box q} \Rightarrow \wedge}{\Rightarrow \Box(p \wedge q) \rightarrow (\Box p \wedge \Box q)} \Rightarrow \rightarrow$$

Le séquent $\Rightarrow \Box(p \wedge q) \rightarrow (\Box p \wedge \Box q)$ est dérivable, donc valide et la formule $\Box(p \wedge q) \rightarrow (\Box p \wedge \Box q)$ est un théorème donc une tautologie.

Pour plus de détails sur le calcul des séquents en logique modale consulter [LAD 77].

6.2.4. *Le calcul des séquents intuitionniste*

Le calcul des séquents s'étend également à logique intuitionniste [LAF 92], cependant la notion de séquent est modifiée. Le conséquent d'un séquent est soit vide, soit constitué d'une seule formule, plus formellement :

DÉFINITION 6.19.– *On appelle séquent une expression de la forme* $\Gamma \Rightarrow$ *ou de la forme* $\Gamma \Rightarrow X$ *où* Γ *est une suite éventuellement vide de formules de la logique intuitionniste et* X *une formule de la logique intuitionniste.*

On note D l'ensemble constitué d'une ou de zéro formule et les règles de déduction sont définies comme suit :

$$\text{règles de permutation} : \quad \frac{X, \Gamma \Rightarrow D}{\Gamma, X \Rightarrow D}$$

$$\text{règles de contraction} : \quad \frac{X, X, \Gamma \Rightarrow D}{X, \Gamma \Rightarrow D}$$

$$\text{règles d'atténuation} : \quad \frac{\Gamma \Rightarrow}{\Gamma \Rightarrow X} \qquad \frac{\Gamma \Rightarrow D}{X, \Gamma \Rightarrow D}$$

Les règles qui correspondent aux connecteurs propositionnels sont :

$$\frac{\Gamma \Rightarrow X}{\neg X, \Gamma \Rightarrow} \neg \Rightarrow \qquad \frac{X, \Gamma \Rightarrow}{\Gamma \Rightarrow \neg X} \Rightarrow \neg$$

$$\frac{\Gamma \Rightarrow X \quad Y, \Gamma \Rightarrow D}{X \to Y, \Gamma \Rightarrow D} \to \Rightarrow \qquad \frac{X, \Gamma \Rightarrow Y}{\Gamma \Rightarrow X \to Y} \Rightarrow \to$$

$$\frac{\Gamma, X \Rightarrow D}{\Gamma, X \wedge Y \Rightarrow D} \leftrightarrow \Rightarrow \qquad \frac{\Gamma, Y \Rightarrow D}{\Gamma, X \wedge Y \Rightarrow D} \Rightarrow \leftrightarrow$$

$$\frac{\Gamma \Rightarrow X \quad \Gamma \Rightarrow Y}{\Gamma \Rightarrow X \wedge Y} \wedge \Rightarrow$$

$$\frac{\Gamma \Rightarrow X}{\Gamma \Rightarrow X \vee Y} \vee \Rightarrow \qquad \frac{\Gamma \Rightarrow Y}{\Gamma \Rightarrow X \vee Y} \Rightarrow \vee$$

$$\frac{\Gamma, X \Rightarrow D \quad \Gamma, Y \Rightarrow D}{\Gamma, X \vee Y \Rightarrow D} \vee \Rightarrow$$

Les notions de dérivation, de validité, d'adéquation et de compétude sétendent au calcul des séquents en logique intuitionniste. La complexité des procédures de décision utilisant le calcul des séquents en logique intuitionniste est généralement P-space complet, une procédure de complexité $O(nlogn)$-space complete est donnée dans [HUD 93].

6.3. Le temps : programmation logique

La programmation logique a été étendue à des logiques temporelles, plusieurs langages de programmation logique ont été définis, ils étendent la programmation logique à des opérateurs temporels et utilisent, le plus souvent, un version temporelle de la résolution. Ils diffèrent dans par les représentations du temps qu'ils utilisent et par les classes de problèmes qu'ils peuvent représenter. Nous allons brièvement passer en revue les langages les plus significatifs, pour plus de détails le lecteur pourra se référer à [ORG 94].

6.3.1. *Le langage Templog*

Le langage **Templog** [ABA 87], [ABA 89], utilise une représentation discrète du temps, au moyen d'un axe linéaire du temps, non borné pour le futur. Les instants sont représentés des entiers naturels. Trois opérateurs temporels sont utilisés : $\bigcirc$ (l'instant suivant), $\Box$ (les instants à partir de maintenant), $\Diamond$ (les instants dans le futur). Ces trois opérateurs induisent une extension de la syntaxe Prolog comme suit :

– L'*atome-suivant* ou *next-atome* est un atome préfixé par l'opérateur $\bigcirc$, et $\bigcirc^k\ A$ où A est un atome et $k \geq 0$, signifie que l'opérateur suivant est appiqué k fois à l'atome A. Une *formule-suivante* ou *next-formule* est une formule composée d'atome-suivants.

– Une *clause initiale* est définie par $\forall x_1, \cdots, \forall x_n(A_1 \wedge \cdots \wedge A_n \rightarrow B)$ ou $B \leftarrow A_1 \wedge \cdots \wedge A_n$ où $A_1, \cdots, A_n$ et B sont des formule-suivantes.

– Une *clause permanente* est définie par $\forall x_1, \cdots, \forall x_n \Box(A_1 \wedge \cdots \wedge A_n \rightarrow B)$ ou $B \Leftarrow A_1 \wedge \cdots \wedge A_n$ où $A_1, \cdots, A_n$ et B sont des formule-suivantes.

Un programme Templog est une conjonction de clauses initiales et permanentes. Un but est une conjonction de formule-suivantes. Le mécanisme de déduction en Templog repose sur une forme restreinte d'un système de déduction non-clausal temporel, appelée TSLD-résolution, cette méthode est correcte et complète.

6.3.2. *Le langage Chronolog*

Le langage **Chronolog** [ORG 88], [ORG 92], utilise la même représentation du temps que le langage Templog, cependant deux opérateurs temporels seulement sont utilisés, **first** (le premier) et **next** (le suivant). La syntaxe de Chronolog accepte celle d'un langage du premier ordre et deux extensions, si A est un formule alors **first**A et **next**A sont des formules. Les opérateurs temporels s'appliquent uniquement aux

formules. Toute formule A a une séquence de valeurs dans le temps, la formule **first**A représente la valeur de A à l'instant initial et la formule **next**A représente la valeur de A à l'instant suivant.

Un programme Chronolog est une conjonction de clauses qui peuvent être considérées comme permanentes selon la terminologie de Templog (un opérateur implicite $\Box$ s'applique aux clauses du programme Chronolog) et bien qu'il n' y ait pas de clause initiale à la Templog dans Chronolog, les clauses dont tous les atomes sont préfixés par l'opérateur **first** peuvent être considérées comme des clauses initiales. Le mécanisme de déduction en Chronolog, appellé TiSLD-résolution repose sur extension temporelle de la SL-résolution.

6.3.3. *Le langage Temporal Prolog de Gabbay*

Le langage Temporal Prolog, proposé par D. Gabbay [GAB 87] est une extension de la programmation logique qui autorise des opérateurs modaux et temporels tels que P (à un instant donné dans le passé), F (à un instant donné dans le futur), et $\Box$ (toujours). Ce langage est très expressif et permet des implications imbriquées cependant il comporte des restrictions, l'opérateur $\Box$ ne peut pas préfixer les têtes des clauses d'un programme. La syntaxe de Temporal Prolog est définie comme suit :

– un programme est un ensemble de clauses ;

– une clause est soit une clause ordinaire, soit une clause ordinaire préfixée par l'opérateur $\Box$;

– une clause ordinaire est de la forme H ou de la forme $A \rightarrow H$, où H et A sont respectivement, la tête et le corps de la clause ;

– une tête de clause est soit une formule atomique, soit une formule $F\,A$ ou $P\,A$, où A est une conjonction de clauses ordinaires ;

– un corps de clause est soit une formule atomique, soit une conjonction de corps de clauses, soit une formule $F\,A$ ou $P\,A$, où A est un corps de clause.

Une procédure de déduction pour Temporal Prolog a été proposée par Gabbay, qui est correcte.

Le langage a également été étendu pour traiter la négation par l'echec et traiter la négation dans le cas propositionnel. Par ailleurs, Gabbay [GAB 89] a également proposé un système de déduction "étiquetté" comme base d'une machine de programmation logique temporelle. Le langage considéré est le langage Temporal Prolog enrichi des nouveaux opérateurs temporels, G (toujours dans le futur), H (toujours dans le passé), $\bigcirc$ (l'instant suivant) et $\otimes$ (l'instant précédent). De plus, des indicateurs temporels sont utilisés pour représenter des données temporelles. Par exemple, l'assertion "Si $A(x)$ est vrai au temps t, alors $A(x)$ continuera d'être vrai" est représenté par : $t : A(x) \rightarrow GA(x)$. Quelques restrictions aux clauses d'un programme, en particulier des fonctions de Skolem sont ajoutées pour éliminer les connecteurs existentiels F et P afin de rendre le programme traitable. Les systèmes de déductions "étiquettés" sont

adaptés pour traiter trois représentations du temps : les ordres partiels généraux, les ordres linéaires, les nombres entiers. Chaque représentation nécessite un ensemble différent de règles temporelles et la correction de ces règles a été établi. La complexité du problème de satisfaisabilité d'un ensemble de clauses de Horn temporelles a été étudié dans le cas propositionnel [CHE 93] pour un fragment de Temporal Prolog qui utilise l'opérateur $\bigcirc$ de Templog et ne comporte pas d'opérateurs pour le passé. Le problème de satisfaisabilité pour un un fragment de Temporal Prolog qui utilise seulement les opérateurs $\Box$ et $\Diamond$ est NP-complet et le problème de satisfaisabilité pour un un fragment de Temporal Prolog qui utilise les opérateurs $\Box$, $\Diamond$ et $\bigcirc$ est PSPACE-complet.

6.3.4. *Le langage Temporal Prolog de Sakuragawa*

Un autre langage appelé Temporal Prolog a été proposé par Sakuragawa [SAK 87b]. Dans ce langage la signification des prédicats dans le futur dépend de leur signification dans le passé. Toutes les clauses d'un programme en Temporal Prolog sont *causales* (voir ci-après le langage Starlog). Toutes les clauses d'un programme sont permanentes. Un opérateur sur le passé peut se trouver dans le corps d'une clause et un opérateur sur le futur peut figurer dans la tête d'une clause. Ceci rend le langage Temporal Prolog de Sakuragawa très expressif.

Le mécanisme de déduction de Temporal Prolog repose sur celui de Prolog, une traduction préalable des clauses sous forme normale est effectuée, où le seul opérateur temporel autorisé est l'opérateur représentant l'instant précédent.

6.3.5. *Le langage MTL*

Le langage MTL, programmation logique temporelle avec métrique et opérateurs pour le passé a été proposé par Brzoska [BRZ 92]. Le temps est représenté par un axe non borné dans le passé et le futur, les instants sont représentés par un ensemble d'entiers, noté $\mathcal{Z}$. Les formules du langage MTL sont construites à partir des opérateurs logiques usuels et des opérateurs temporels de base suivants qui s'appliquent uniquement aux formules : $\Box_t$ (toujours pour $t \in \mathcal{Z} \cup \{-\infty, +\infty\}$), $\Diamond_t$ (quelquefois pour $t \in \mathcal{Z} \cup \{-\infty, +\infty\}$), $\circ$ (l'instant suivant), et $\bullet$ (l'instant précédent).

A partir de ces opérateurs de bases d'autres opérateurs temporels peuvent être définis comme, par exemple, : $squareA \equiv_{def} \Box_{+\infty}\Box_{-\infty}A$ (toujours sans restriction), $\Diamond A \equiv_{def} \Diamond_{+\infty}\Diamond_{-\infty}A$ (quelquefois sans restriction), $\Box_+ A \equiv_{def} \Box_{+\infty}A$ (toujours dans le futur), $\Box_- A \equiv_{def} \Box_{-\infty}A$ (toujours dans le passé), $\Diamond_+ A \equiv_{def} \Diamond_{+\infty}A$ (quelquefois dans le futur), $\Diamond_- A \equiv_{def} \Diamond_{-\infty}A$ (quelquefois dans le passé).

Un programme MTL est un ensemble de clauses dont les formules peuvent être préfixées par des opérateurs temporels et un but est une formule préfixée ou pas par un opérateur temporel. Tout comme dans le langage Templog l'opérateur $\Box$ ne s'applique pas au corps d'une règle et l'opérateur $\Diamond$ ne s'applique à une tête de règle, cependant le langage MTL est plus expressif que les langanges Templog et Chronolog.

Le mécanisme de déduction de MTL repose sur la MTL-résolution qui est une restriction de la CLP-résolution sur une certaine algèbre. La la MTL-résolution est correcte et complète.

6.3.6. *Le langage Starlog*

Le langage Starlog [CLE 91] est un langage de programmation logique est un langage qui n'est pas basé sur une logique temporelle et n'a pas d'opérateur temporel. Des arguments temporels sont ajoutés à chaque prédicat Prolog. Le premier argument de chaque prédicat est réservé pour un argument temporel. Le temps est représenté par un axe de réels orienté. Les clauses du programme doivent être causale, c'est à dire que les valeurs des arguments temporels figurant dans une tête de clause doivent être supérieurs aux valeurs des arguments temporels figurant dans leur corps. Le mécanisme de déduction de Starlog ne repose pas sur la résolution mais utilise un démonstrateur de théorèmes basé sur les graphes de connection couplé à solveur de contraintes arithmétiques.

6.4. Programmation logique modale

Les langages de programmation logique temporelle ont été conçus pour traiter des aspects temporels ou dynamiques de certains problèmes, les langages de programmation logique modale ont pour objectif de traiter des problèmes où interviennent les notions de connaissances, de croyances et d'hypothèses.

6.4.1. *Le langage Molog*

Le langage Molog a été introduit par L. Fariñas del Cerro [Far 86], il étend Prolog à la logique modale de façon à exprimer les concepts de connaissances, de croyances et d'hypothèses. L'utilisateur choisit une logique modale et définit les règles d'utilisation des opérateurs modaux correspondants. Le langage Molog fournit un cadre général qui peut être utilisé avec différentes logiques modales. Les opérateurs modaux sont groupés en deux catégories les opérateurs universels, comme $\Box$ et les opérateurs existentiels, comme $\Diamond$, pour lesquels une sémantique de Kripke est utilisée.

La structure de base de Molog repose sur les clauses de Horn modales qui sont des clauses de Horn préfixées par des opérateurs modaux, et la forme générale d'une clause de Horn modale est $M(A \leftarrow B_1 \wedge \cdots \wedge B_n)$ où M est une modalité, A est une formule modale atomique et $B_1, \cdots, B_n$ sont des formules modales.

Une séquence d'opérateurs modaux est appelée *modalité*, elle peut s'appliquer une formule modale atomique, une conjonction de formules modales ou à une clause de Horn modale.

Un programme Molog est un ensemble de clauses de Horn modales et un but est une conjonction de formules modales. Le mécanisme de déduction de Molog repose sur la résolution modale qui est une extension de la résolution à la logique modale et

dépend de la logique modale utilisée. Pour la logique modale S_5 la résolution modale est complète. Un autre mécanisme de déduction, appelé *compilation*, a été également proposé dans [Far 86] pour S_5, il transforme les clauses de Horn modales en clauses de Horn sans modalité et utilise l'unification et la résolution standard de Prolog. P. Balbiani *et al.* [Far 88] ont proposé une méthode de résolution pour la logique modale Q, appelée SLD-résoltion modale qui est complète.

La complexité du problème de satisfaisabilité d'un ensemble de clauses de Horn modales a été étudiée, un algorithme a été proposé [Far 87], pour la logique modale S_5, cet algorithme a une complexité polynomiale en temps, en revanche, pour les logiques modales K, Q, T et S_4 la complexité dans le pire des cas est nettement supérieure.

P. Balbiani *et al.* [HER 91] ont ensuite proposé une extension du langage Molog, appelé TIM (Toulouse Inference Machine) qui fournit un cadre général pour la programmation logique modale.

6.4.2. *Le langage Modal Prolog*

Le langage Modal Prolog [SAK 87a] étend Prolog de façon à exprimer modularité, hiérarchie et/ou structure. Un programme logique modal est constitué de deux parties, une description des mondes possibles et une description des relations entre les mondes possibles.

La description d'un monde possible est un programme Prolog dont les atomes sont préfixés par les opérateurs modaux $\Box$ (nécessité) et $\Diamond$ (possibilité). Les opérateurs modaux ne peuvent pas être utilisés dans la tête d'une clause.

La description des relations entre mondes donne les relations d'accessibilité entre mondes possibles et leurs attributs comme, par exemple, réflexivité, symétrie, transitivité. Une clause de relation est une clause Prolog qui décrit les attributs d'une relation d'accessibilité.

Chaque description d'un monde possible correspond à un module du programme, les interactions entre modules du programme sont décrit dans la définition des relations et le langage Modal Prolog introduit une structure hiérarchique dans la programmation logique.

Le mécanisme de déduction de Modal Prolog repose sur l'unification et la résolution et ce mécanisme est correct et complét.

6.4.3. *Le langage Modal Logic Programming*

S. Akama [AKA 86] a proposé une extension de Prolog reposant la logique modale S_5 qui utilise une sémantique de Kripke standard pour les opérateurs modaux. Ce langage de programmation utilise des *clauses définies modales* qui sont une conjonction de littéraux modaux, avec un seul littéral positif. Un littéral modal est un littéral

est de la forme $\bigtriangledown A$ ou $\neg(\bigtriangledown A)$ où $\bigtriangledown$ est une modalité, c'est-à-dire une séquence d'opérateurs modaux $\Box$ et $\Diamond$ et A est une formule atomique.

Un programme est un ensemble de clauses définies modales et le mécanisme de déduction de Modal Logic Programming est une extension de la SLD-résolution standard avec deux règles d'inférences pour les opérateurs modaux. Cette méthode de résolution est complète.