Introduction au domaine de recherche : Axiomatique en théorie des langages et des expressions régulières. Un calcul des séquents pour les treillis distributifs avec action star-continus

Florent Bréhard

2 juin 2015

Introduction

La théorie de la démonstration est la branche de la logique qui s'intéresse à la formalisation du raisonnement mathématique à travers ses étapes les plus élémentaires. Son champ d'application va des questions les plus théoriques concernant les fondements mêmes des mathématiques, jusqu'aux enjeux technologiques et industriels d'aujourd'hui comme la sécurité logicielle des systèmes critiques. L'auteur de ces pages a eu l'occasion lors de trois stages de recherche de se familiariser avec plusieurs aspects de cette discipline à l'interface entre mathématiques, logique et informatique. Les deux premiers stages concernèrent une avancée récente et très prometteuse en théorie des types : la théorie homotopique des types (HoTT), l'un plus orienté informatique avec l'implémentation en Coq d'un fragment de la théorie, l'autre centré sur l'étude mathématique des modèles de cette théorie. Le troisième stage eut pour objet un sujet plus ancien dans lequel de nombreuses questions restent ouvertes : la théorie des langages et des expressions régulières du point de vue de l'axiomatique.

La présente introduction au domaine de recherche prend le parti de se concentrer sur le domaine de ce dernier stage. Les expressions régulières sont des objets mathématiques qui furent intensément étudiés aux début de l'informatique de par leur intérêt dans de nombreux problèmes naturels comme la recherche de caractère, le *parsing* de code source et bien d'autres encore. La première partie de de ce rapport présente donc quelques généralités sur la théorie des expressions régulières et leur généralisation à travars la notion d'algèbre de Kleene. Nous présenterons notamment la théorie du premier ordre proposée par Kozen pour axiomatiser la classe des algèbres de Kleene. Ensuite, nous explorerons dans une deuxième partie différentes extensions possibles aux expressions régulières. L'ajout de nouveaux opérateurs permet ainsi à la syntaxe d'être plus expressive et donc de refléter une classe de problèmes plus grande. Enfin, il sera présenté en troisième partie un nouveau calcul des séquents introduit et étudié par l'auteur de ces pages, avec pour but de couvrir la théorie équationnelle des treillis complets avec action et $\star$-continus ; nous en verrons également les limites.

Table des matières

1 Par où tout commença : expressions régulières, algèbres de Kleene et automates finis

1.1 Expressions régulières

Les *expressions régulières* constituent un élément central dans la théorie des langages. Introduites dans les années 1940 et étudiées plus précisément par le logicien Stephen Cole Kleene, elles connurent un essor remarquable dans les années 1960 avec les débuts de l'informatique grâce à leurs nombreuses applications en recherche de chaînes de caractères ([12], [3]), en analyse lexicale de langages de programmation, etc. Formellement, une expression régulière est un terme construit à partir de "lettres" de base appartenant à un alphabet (fini) Σ donné et suivant la syntaxe récursive suivante :

$$e, f, g \quad ::= \quad a \in \Sigma \mid 0 \mid 1 \mid e + f \mid e \cdot f \mid e^\star$$

A chaque expression régulière e, nous pouvons associer un langage $L(e) \subseteq \Sigma^\star$ de mots (finis) sur Σ (En mathématiques, $\Sigma^\star$ s'appelle le monoïde libre sur Σ) :

- $L(a) = \{a\} \quad (a \in \Sigma)$
- $L(0) = \varnothing$
- $L(1) = \{\epsilon\}$
- $L(e + f) = L(e) + L(f) = L(e) \cup L(f)$
- $L(e \cdot f) = L(e) \cdot L(f) = \{u \cdot v \mid u \in L(e) \wedge v \in L(f)\}$
- $L(e^\star) = L(e)^\star = \bigcup_{n \in \omega} L(e)^n$ où $L(e)^n$ désigne la concaténation de n copies de $L(e)$.

Les langages sur Σ obtenus par traduction d'une expression régulière sont appelés les *langages réguliers*. Nous verrons plus loin une autre caractérisation de cette classe de langage par les *automates finis*. Si $L(e) \subseteq L(f)$, on écrit alors $REG \vDash e \leq f$: on dit que le terme e est plus petit que f dans la sémantique des expressions régulières.

On ajoute parfois un opérateur supplémentaire, $^+$, qui est la variante transitive stricte de l'étoile de Kleene *. On peut le définir via $e^+ = e \cdot e^\star$, et récirpoquement $e^\star = 1 + e^+$.

1.2 Algèbres de Kleene

Les algèbres de Kleene constituent une généralisation de la notion d'expression régulière. On identifie désormais Σ (supposé ici infini dénombrable) avec un ensemble de variables. On considère le langage du premier ordre dont la signature se compose des constantes 0 et 1, des symboles d'opérateurs binaires $+$ (somme) et $\cdot$ (concaténation) ansi que de l'opérateur unaire * (étoile de Kleene). Une *algèbre de Kleene* est alors une structure algébrique sur cette signature vérifiant les axiomes de la figure 1 (cette axiomatisation est due à Dexter Kozen). La relation d'ordre $\leq$ est définissable par $a \leq b \Leftrightarrow a + b = b$.

Par ailleurs, une algèbre de Kleene est dite $\star$-continue si elle vérifie l'axiome :

$$a \cdot b^\star \cdot c = \sup_{n \in \omega} a \cdot b^n \cdot c \qquad (\star/\cdot)$$

Il est important de constater que cet axiome n'est en général pas exprimable au premier ordre.

1.3 Notions de sémantique

Il existe en particulier deux classes d'algèbres de Kleene $\star$-continues, correspondant chacune à une notion de sémantique pour les expressions régulières.

FIGURE 1 – Axiomatisation de Kozen de KA

$$
\begin{array}{rcll}
a+(b+c) & = & (a+b)+c & (K1)\\
a+b & = & b+a & (K2)\\
a+0 & = & a & (K3)\\
a+a & = & a & (K4)\\
a\cdot(b\cdot c) & = & (a\cdot b)\cdot c & (K5)\\
1\cdot a & = & a & (K6)\\
a\cdot 1 & = & a & (K7)\\
a\cdot(b+c) & = & a\cdot b+a\cdot c & (K8)\\
(a+b)\cdot c & = & a\cdot c+b\cdot c & (K9)\\
0\cdot a & = & 0 & (K10)\\
a\cdot 0 & = & 0 & (K11)\\
1+a\cdot a^\star & \leq & a^\star & (K12)\\
1+a^\star\cdot a & \leq & a^\star & (K13)\\
b+a\cdot x\leq x & \Rightarrow & a^\star\cdot b\leq x & (K14)\\
b+x\cdot a\leq x & \Rightarrow & b\cdot a^\star\leq x & (K15)
\end{array}
$$

1.3.1 Sémantique des langages

Soit S un alphabet (fini ou infini). L'ensemble $\mathcal{P}(S^\star)$ des langages sur S forme une algèbre de Kleene $\star$-continue de la manière suivante :

$$
\begin{array}{rcl}
0 & = & \varnothing\\
1 & = & \{\epsilon\}\quad\text{(le mot vide)}\\
L+K & = & L\cup K\\
L\cdot K & = & \{u\cdot v\mid u\in L\wedge v\in K\}\\
L^\star & = & \bigcup_{n\in\omega}L^n
\end{array}
$$

Une *valuation* de Σ dans les langages de S est une application $\sigma:\Sigma\to\mathcal{P}(S^\star)$. Ainsi, si e est une expression régulière (sur Σ), la valuation σ permet d'interpréter e comme un langage $[\![e]\!]_{S,\sigma}$ sur S (noté simplement $[\![e]\!]_\sigma$) :

$$
[\![a]\!]_\sigma=\sigma(a)\quad [\![0]\!]_\sigma=\varnothing\quad [\![1]\!]_\sigma=\{\epsilon\}\quad [\![e+f]\!]_\sigma=[\![e]\!]_\sigma+[\![f]\!]_\sigma\quad [\![e\cdot f]\!]_\sigma=[\![e]\!]_\sigma\cdot[\![f]\!]_\sigma\quad [\![e^\star]\!]_\sigma=[\![e]\!]_\sigma^\star
$$

Dans la sémantique des langages, on dira que $LANG\vDash e\leq f$ si et seulement si pour tout alphabet S et toute valuation $\sigma:\Sigma\to\mathcal{P}(S^\star)$, $[\![e]\!]_\sigma\subseteq[\![f]\!]_\sigma$.

1.3.2 Sémantique relationnelle

Soit U un ensemble. L'ensemble $\mathcal{P}(U^2)$ des relations binaires sur U peut être muni d'une structure d'algèbre de Kleene ($\star$-continue) de la façon suivante :

$$
\begin{array}{rcl}
0 & = & \varnothing\\
1 & = & id=\{(i,i)\mid i\in U\}\\
R+S & = & R\cup S\\
R\cdot S & = & \{(i,j)\mid\exists k\in U\cdot(i,k)\in R\wedge(k,j)\in S\}\\
R^\star & = & \bigcup_{n\in\omega}R^n
\end{array}
$$

Une *valuation* de Σ dans les relations binaires de U est une application $\sigma:\Sigma\to\mathcal{P}(U^2)$. De manière analogue, on peut alors définir pour une expression régulière e la relation binaire $[\![e]\!]_{U,\sigma}$ (abrégé en $[\![e]\!]_\sigma$) qu'elle dénote par σ. En sémantique relationnelle, on écrit $REL\vDash e\leq f$ si et seulement si pour tout ensemble U et toute valuation $\sigma:\Sigma\to\mathcal{P}(U^2)$, $[\![e]\!]_\sigma\subseteq[\![f]\!]_\sigma$.

1.4 Un théorème de complétude

Un théorème important de Dexter Kozen ([9]) dit que l'axiomatisation des algèbres de Kleene qu'il propose est complet pour les sémantiques relationnelle et de langages, ainsi que pour les expressions régulières en tant que langages :

Theorem 1. *Soient e et f deux expressions régulières. On a alors :*

$$KA \vdash e \leq f \quad \Leftrightarrow \quad REG \vDash e \leq f \quad \Leftrightarrow \quad LANG \vDash e \leq f \quad \Leftrightarrow \quad REL \vdash e \leq f$$

En particulier, cela implique que la théorie équationnelle des algèbres de Kleene coïncide avec celles qui sont $\star$-continues :

Corollary 1.

$$Eq(\mathsf{KA}) = Eq(\mathsf{KA}^\star)$$

Notons que ce théorème de Kozen est relativement difficile à démontrer. La preuve s'appuie sur la notion d'automates finis (voir le paragraphe suivant) et de matrices d'expressions régulières.

1.5 Un modèle de calcul pour les langages réguliers : les automates déterministes

On rappelle qu'un langage $L \subseteq \Sigma^\star$ est dit *régulier* s'il est le langage $L(e)$ associé à une expression régulière e sur Σ. Si e et f sont deux expressions régulières, on a vu que $L(e) \subseteq L(f)$ (noté $REG \vDash e \leq f$) si et seulement si $\mathsf{KA} \vdash e \leq f$. En fait, il existe une autre caractérisation de ces langages qui permet même de *décider* l'inclusion $L(e) \subseteq L(f)$: les *automates déterministes*. Cela signifie qu'il existe une machine qui étant données e et f décide si $L(e) \subseteq L(f)$ ou non. L'objectif de ce paragraphe n'est pas de présenter en détail la théorie des automates finis, mais d'en donner les définitions et théorèmes essentiels (le lecteur pourra sa référer à [6] (en français) ou [7]).

Definition 1. *Un automate déterministe sur un alphabet (fini) Σ est un tuple $\mathcal{A} = \langle Q, I, F, \Delta \rangle$ où :*

- *Q est l'ensemble fini des* états *de l'automate.*
- *$I \subseteq Q$ est l'ensemble des* états initiaux.
- *$F \subseteq Q$ est l'ensemble des* états finals.
- *$\Delta \subseteq Q \times \Sigma \times Q$ est la* relation de transitition *de l'automate.*

Par ailleurs, si $I = \{\iota\}$ et si pour tout $p \in Q$ et $a \in \Sigma$, $|\{(q \in Q \mid (p, a, q) \in \Delta\}| \leq 1$ (autrement dit, si Δ est une fonction (partielle) de $Q \times \Sigma$ dans Q), on dit que l'automate fini $\mathcal{A}$ est déterministe*, et on préfère parler de fonction de transition (partielle, mais on peut la compléter avec un état puits) $\delta : Q \times \Sigma \to Q$.*

Etant donné un mot $w = a_1 a_2 \dots a_n \in \Sigma^\star$, un *calcul* de $\mathcal{A}$ sur w est une suite d'états $(q_0, q_1, \dots, q_n)$ telle que $q_0 \in I$ et pour tout $k \in [1, n]$, $(q_{k-1}, a_k, q_k) \in \Delta$. Ce calcul est *acceptant* si $q_n \in F$. On dit alors qu'un mot $w \in \Sigma^\star$ est *reconnu* par $\Sigma^\star$ si et seulement si il existe pour ce mot un calcul acceptant dans $\mathcal{A}$. Cela nous amène à la définition suivante :

Definition 2. *Etant donné un automate fini $\mathcal{A}$ sur l'alphabet $\Sigma^\star$, on appelle $L(\mathcal{A}) \subseteq \Sigma^\star$ le langage des mots reconnus par $\mathcal{A}$. Un langage $L \subseteq \Sigma^\star$ est dit reconnaissable s'il est reconnu par un automate fini.*

La figure 2 présente un automate sur $\Sigma = \{a, b\}$ reconnaissant tous les mots finissant par ab.

Un premier théorème, dont la preuve, omise ici, est effective, établit l'équivalence entre la classe des langages reconnaissables (par automates finis) et celle des langages reconnaissables par automates finis déterministes :

Theorem 2. *Si $L \subseteq \Sigma^\star$ est un langage reconnu par un automate avec n états, alors il existe un automate déterministe avec au plus 2^n états reconnaissant L.*

La figure 3 montre le déterminisé de $\mathcal{A}$ de la figure 2 :

Le théorème fondamental en théorie des automates finis dit que la classe des langages reconnaissables sur Σ est exactement celle des langages réguliers sur Σ :

FIGURE 2 – Un automate $\mathcal{A}$ reconnaissant $L = \Sigma^\star ab$

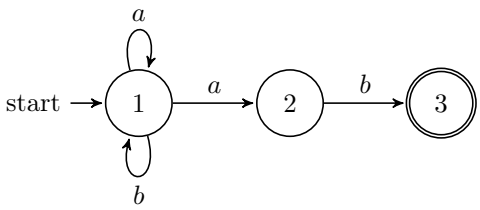
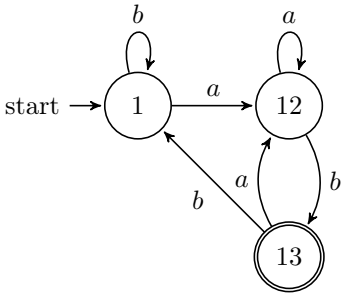


FIGURE 3 – Un automate déterministe $\mathcal{A}'$ reconnaissant $L = \Sigma^\star ab$



Theorem 3. *Soit $L \subseteq \Sigma^\star$ un langage. L est régulier (c'est-à-dire qu'il existe une expression régulière e telle que $L = L(e)$) si et seulement si L est reconnaissable (c'est-à-dire qu'il existe un automate fini $\mathcal{A}$ tel que $L = L(\mathcal{A})$). De plus, ces constructions sont effectives.*

Grâce à ce théorème, nous pouvons ramener le problème de décision de l'inclusion de deux langages réguliers (dénotés par deux expressions réguilières) à la décision de l'inclusion de deux langages reconnus par automates. Considérons deux automates $\mathcal{A} = \langle Q_a, \iota_A, F_A, \delta_A \rangle$ et $\mathcal{B} = \langle Q_B, \iota_B, F_B, \delta_B \rangle$, que l'on peut supposer déterministes (quitte à les déterminiser). On va construire une relation $R \subseteq Q_A \times Q_B$, appelée *relation de simulation*, telle que :

- $\iota_A R \iota_B$
- $\forall p \in Q_A, q \in Q_B, a \in \Sigma \cdot pRq \Rightarrow \delta_A(p,a) R \delta_B(q,a)$

La construction effective de R peut-être réalisée de la manière suivante (on ne cherche pas ici à l'optimiser) :

1. Partir de $R = \{(\iota_A, \iota_B)\}$.
2. Tant qu'il existe $p \in Q_A, q \in Q_B, a \in \Sigma$ tels que $(p,q) \in R$ mais $(\delta_A(p,a), \delta_B(q,a)) \notin R$, ajouter $(\delta_A(p,a), \delta_B(q,a))$ à R.

Comme à chaque répétition de l'étape 2, le cardinal de R augmente de 1 et que R est une sous-partie de l'ensemble fini $Q_A \times Q_B$, cet algorithme termine. On peut alors conclure quant à l'inclusion $L(\mathcal{A}) \subseteq L(\mathcal{B})$ grâce au lemme suivant.

Lemma 1. *Les deux énoncés suivants sont équivalents :*

- $L(\mathcal{A}) \subseteq L(\mathcal{B})$
- $\forall p \in F_A, q \in Q_B \cdot pRq \Rightarrow q \in F_B$

La preuve de ce lemme est relativement facile :

Démonstration. Remarquons d'abord que dans un automate déterministe complet, chaque mot engendre exactement un calcul. Par ailleurs, on voit aisément par construction de R que si pRq ($p \in Q_A$ et $q \in Q_B$), alors on a un mot $w = a_1 \dots a_n \in \Sigma^\star$ tel que le calcul associé dans $\mathcal{A}$ $(\iota_A, p_1, \dots, p_n)$ termine en $p_n = p$ et

tel que le calcul associé dans $\mathcal{B}$ $(\iota_B, q_1, \ldots, q_n)$ termine en $q_n = q$. Donc si $L(\mathcal{A}) \subseteq L(\mathcal{B})$, l'hypothèse $p \in F_A$ implique que w est dans $L(\mathcal{A})$ donc dans $L(\mathcal{B})$, donc que $q \in F_B$.

Réciproquement, soit $w \in L(\mathcal{A})$. Considérons dans $\mathcal{A}$ son calcul acceptant $(\iota_A, p_1, \ldots, p_n)$. On a $p_n \in F_A$. Soit $(\iota_B, q_1, \ldots, q_n)$ le calcul de w dans $\mathcal{B}$. Grâce aux hypothèses sur R, on sait que pour tout $k \in [1, n]$, $p_k R q_k$. En particulier, $p_n R q_n$, ce qui par hypothèse implique $q_n \in F_B$, donc que $w \in L(\mathcal{B})$. □

2 Treillis de Kleene, treillis avec action

2.1 Extensions de la syntaxe des expressions régulières

L'étude des algèbres de Kleene se prolonge naturellement vers de possibles extensions du langage de base des expressions régulières. Cela présente à la fois un intérêt théorique (Quels sont les nouveaux axiomes à ajouter ? Quel sens donner à ces nouveaux opérateurs dans les sémantiques décrites précédemment ?) et un intérêt pratique (Etendre l'expressivité des expressions régulières pour généraliser leur usage dans la recherche de chaînes de caractères par exemple). Nous nous pencherons sur deux extensions : la conjonction et la résiduation.

2.1.1 la conjonction $\sqcap$

Si e et f sont deux expressions régulières, on notera $e \sqcap f$ leur conjonction (ou intersection). A l'instar de la constante 0 (le plus petit élément) qui sert d'élément neutre pour l'opérateur somme $+$, on introduit une constante $\top$ (le plus grand élément) comme élément neutre de $\sqcap$. On rajoute les axiomes suivants :

$$\begin{array}{rcll} a \sqcap (b \sqcap c) & = & (a \sqcap b) \sqcap c & (K16) \\ a \sqcap b & = & b \sqcap a & (K17) \\ a \sqcap \top & = & a & (K18) \\ a \sqcap a & = & a & (K19) \\ a + (a \sqcap b) & = & a & (K20) \\ a \sqcap (a + b) & = & a & (K21) \end{array}$$

Ces opérateurs ont une interprétation canonique dans les langages et les relations compatible avec les axiomes $(K16 - 21)$:

- Pour deux langages L et K sur un alphabet S, $L \sqcap K = L \cap K$ est leur intersection ensembliste. La constante $\top$ s'interprète comme le langage plein $S^\star$.
- Pour deux relations binaires R et S sur un ensemble U, $R \sqcap S = R \cap S$ est leur intersection ensembliste et la constante $\top$ représente la relation pleine U^2.

Grâce à cela, il est possible d'étendre les trois sémantiques aux expressions régulières avec intersection :

- Pour deux expressions e et f, $L(e \sqcap f) = L(e) \cap L(f)$, et $L(\top) = \Sigma^\star$.
- Pour deux expressions e et f et une valuation de langages $\sigma : \Sigma \to \mathcal{P}(S^\star)$, $[\![e \sqcap f]\!]_\sigma = [\![e]\!]_\sigma \cap [\![f]\!]_\sigma$ et $[\![\top]\!]_\sigma = S^\star$.
- Pour deux expressions e et f et une valuation relationnelle $\sigma : \Sigma \to \mathcal{P}(U^2)$, $[\![e \sqcap f]\!]_\sigma = [\![e]\!]_\sigma \cap [\![f]\!]_\sigma$ et $[\![\top]\!]_\sigma = U^2$.

2.1.2 La résiduation à gauche $\multimap$ et à droite $\circ\!\!-$

Si e et f sont deux expressions régulières, on définit la résiduation à gauche de f par e comme $e \multimap f$ et la résiduation à droite de f par e comme $f \circ\!\!- e$. Les axiomes régissant ces deux opérateurs en font les adjoints respectivement à gauche et à droite de la concaténation :

$$a \cdot b \leq c \Leftrightarrow b \leq a \multimap c \quad (\multimap /\cdot) \qquad a \cdot b \leq c \Leftrightarrow a \leq c \circ\!\!- b \quad (\circ\!\!- /\cdot)$$

Un des intérêts pointés par Vaughan Pratt ([11]) d'introduire les résiduations est la possibilité de donner une axiomatisation équationnelle finie là où l'axiomatisation de Kozen fait appel à deux formules de Horn (implication d'équations). Cette opération d'adjonction impose une interprétation canonique de ces deux opérateurs dans les langages et les relations :

- Pour deux langages L et K sur un alphabet S,

$$\begin{array}{rcl} L \multimap K & = & \{v \in S^\star \mid \forall u \cdot u \in L \Rightarrow u \cdot v \in K\} \\ K \circ\!\!- L & = & \{u \in S^\star \mid \forall v \cdot v \in L \Rightarrow u \cdot v \in K\} \end{array}$$

- Pour deux relations R et S sur un ensemble U,

$$\begin{array}{rcl} R \multimap S & = & \left\{(i,j) \in U^2 \;\middle|\; \forall k \in U \cdot (k,i) \in R \Rightarrow (k,j) \in S\right\} \\ S \circ\!\!- R & = & \left\{(i,j) \in U^2 \;\middle|\; \forall k \in U \cdot (j,k) \in R \Rightarrow (i,k) \in S\right\} \end{array}$$

D'où une extension naturelle des sémantiques pour la résiduation.

2.1.3 Un peu de nomenclature

Voici comment on nommera par la suite les différentes extensions de KA :

- Un ensemble muni d'une structure d'algèbre de Kleene augmentée d'une intersection $\sqcap$ avec élément un plus grand élément $\top$ et satisfaisant les axiomes correspondant s'appelle un *treillis de Kleene* (*Kleene lattice*). On notera KL la classe des treillis de Kleene. On notera $\mathsf{KL}^\star$ la classe des treillis de Kleene $\star$-continus (axiome $\star/\cdot$)).
- Un ensemble muni d'une structure d'algèbre de Kleene augmentée des opérateurs adjoints à la concaténation que sont la résiduation à gauche $\multimap$ et à droite $\circ\!\!-$ (axiome P) s'appelle une *algèbre avec action* (*action algebra*). Les algèbres avec action forment une classe nommée ACTA. Les algèbres avec action $\star$-continues forment la classe $\mathsf{ACTA}^\star$.
- Enfin, en combinant tous ces opérateurs, on obtient la classe ACTL des *treillis avec action*. De même, on notera $\mathsf{ACTL}^\star$ la classe des treillis avec action $\star$-continus.

2.2 Incomplétude vis-à-vis des sémantiques

Contrairement au cas initial des algèbres de Kleene, il n'y a plus d'équivalence entre les différentes sémantiques et a fortiori plus de théorème de complétude des axiomes proposés vis-à-vis de ces sémantiques. Voyons quelles sont ces différences :

- La sémantique des langages dénotés par les expressions régulières montre ici clairement ses limitations. En effet, le langage dénoté par une expression régulière n'est qu'un cas particulier de la sémantique des langages où la valuation σ associe à tout $a \in \Sigma$ le singleton $\{a\}$. Dans le cadre des algèbres de Kleene, ce choix était "suffisamment général" pour maintenir l'équivalence. Mais avec les nouveaux opérateurs, cela tombe en défaut. Par exemple, si $a, b \in \Sigma$, $L(a \sqcap b) = \varnothing$ et $L(a \multimap b) = \varnothing$. Or ce ne sont clairement pas des théorèmes dans $LANG$ ou REL. Dans la suite, nous abandonnerons donc le point de vue du langage dénoté par une expression régulière.
- Tous les modèles de REL et $LANG$ sont des treillis distributifs. Par là, on entend que l'on a une distributivité mutuelle entre $+$ et $\sqcap$:

$$(a+b) \sqcap c = (a \sqcap c) + (b \sqcap c) \quad \text{et} \quad (a \sqcap b) + c = (a+c) \sqcap (b+c)$$

  En revanche, sous les axiomes de ACTL, nous ne pouvons montrer que des règles de semi-distributivité :

$$\begin{array}{lll} \mathsf{ACTL} \vDash (a \sqcap c) + (b \sqcap c) \leq (a+b) \sqcap c & \text{mais} & \mathsf{ACTL} \nvDash (a+b) \sqcap c \leq (a \sqcap c) + (b \sqcap c) \\ \mathsf{ACTL} \vDash (a \sqcap b) + c \leq (a+c) \sqcap (b+c) & \text{mais} & \mathsf{ACTL} \nvDash (a+c) \sqcap (b+c) \leq (a \sqcap b) + c \end{array}$$

  C'est ce point que nous allons tenter de corriger par la suite.
- Dans les modèles de REL ou $LANG$, il apparait que si un élément e vérifie $e \leq 1$, alors $e \cdot e = e$. Avec les axiomes de ACTL en revanche, on peut seulement montrer $e \cdot e \leq e$. Or, grâce aux connecteurs $\sqcap$, $\multimap$ ou $\circ\!\!-$ et à la contante 1, on peut créer des termes plus petits que 1 et ainsi créer des contre-exemples :

$$\begin{array}{lll} LANG, REL \vDash 1 \sqcap a \sqcap b = (1 \sqcap a) \cdot (1 \sqcap b) & \text{mais} & \mathsf{ACTL} \nvDash 1 \sqcap a \sqcap b \leq (1 \sqcap a) \cdot (1 \sqcap b) \\ LANG, REL \vDash (a \multimap a) \multimap 1 = ((a \multimap a) \multimap 1)^2 & \text{mais} & \mathsf{ACTL} \nvDash (a \multimap a) \multimap 1 \leq ((a \multimap a) \multimap 1)^2 \end{array}$$

De plus, $LANG$ diffère de REL dans le fait qu'il interprète 1 de manière atomique ($1 = \{\epsilon\}$). On obtient ainsi des théorèmes supplémentaires dans $LANG$:

$$LANG \vDash (1 \sqcap a) \cdot b = b \cdot (1 \sqcap a) \quad \text{mais} \quad REL \nvDash (1 \sqcap a) \cdot b = b \cdot (1 \sqcap a)$$

- La constante $\top$ pose elle aussi problème. On peut grâce à elle exhiber des théorèmes de REL qui sont en général faux dans ACTL :

$$REL \vDash (((\top \multimap a) \circ\!\!- \top) \multimap b) + a = \top \quad \text{mais} \quad \mathsf{ACTL} \nvDash (((\top \multimap a) \circ\!\!- \top) \multimap b) + a = \top$$

On peut même obtenir des théorèmes vrais dans REL mais faux dans $LANG$:

$$REL \vDash \top \cdot a \cdot \top \cdot b \cdot \top = \top \cdot b \cdot \top \cdot a \cdot \top \quad \text{mais} \quad LANG \nvDash \top \cdot a \cdot \top \cdot b \cdot \top = \top \cdot b \cdot \top \cdot a \cdot \top$$

3 Un calcul de séquent pour les treillis avec action distributifs et $\star$-continus : dACTL_ω

3.1 Présentation du calcul de séquents dACTL_ω

Nous venons de voir qu'un des manques les plus évidents des axiomes de ACTL vis-à-vis des sémantiques relationnelle et de langages est la distributivité entre $+$ et $\sqcap$. On ajoute donc cet axiome de distributivité :

$$(a + b) \sqcap c = (a \sqcap c) + (b \sqcap c)$$

La classe obtenue est celle des *treillis distributifs avec action*, notée dACTL. La $\star$-continuité dans cette classe se définit par deux conditions (puisque $+$ distribue avec $\cdot$ et $\sqcap$) :

$$a \cdot b^\star \cdot c = \sup_{n \in \omega} a \cdot b^n \cdot c \quad \text{et} \quad a \sqcap b^\star = \sup_{n \in \omega} a \sqcap b^n$$

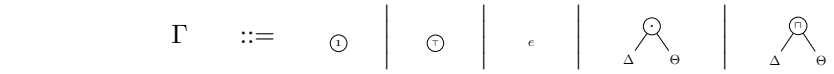
Les treillis distributifs avec action $\star$-continus forment la classe $\mathsf{dACTL}^\star$. De même, on notera dKL pour les treillis de Kleene distributifs, $\mathsf{dKL}^\star$ pour ceux qui sont $\star$-continus (axiomes ..+..), dACTA pour les algèbres avec action distributives et $\mathsf{dACTA}^\star$ pour celles qui sont $\star$-continues (axiome ..).

Notre objectif ici est d'introduire un nouveau calcul de séquents dACTL_ω qui reflète les axiomes de $\mathsf{dACTL}^\star$. C'est en fait une extension du calcul des séquents ACTA_ω de Palka ([10]) dans lequel la conjonction $\sqcap$ ne distribuait pas sur la somme $+$. Un *calcul de séquents* est un système formel dans lequel les *séquents* représentent des propositions. Leur enchainement logique (processus de déduction) est régi par des *règles d'inférence* dont la forme générique est :

$$\frac{\psi_1 \qquad \psi_2 \qquad \dots \qquad \psi_n}{\phi} \text{ (règle } R\text{)}$$

Les séquents $\psi_1, \psi_2, \dots, \psi_n$ sont appelés les *prémisses* : ce sont les hypothèses que l'on devra démontrer pour appliquer la règle. Le séquent ϕ est la *conclusion* de la règle. Si $n = 0$ (pas de prémisse), la règle se comporte donc comme un axiome. Une preuve de ϕ a donc la structure d'un arbre où toutes les branches sont finies : chaque prémisse de la dernière règle est elle-même conclusion d'un sous-arbre qui la prouve. On peut également envisager des règles avec une infinité de prémisses (une telle règle est dite infinitaire). Dans ce cas, un arbre de preuve avec cette règle est infini bien que chaque branche soit finie.

Dans le cas du calcul de séquents dACTL_ω, les séquents ont la forme générique $\Gamma \vdash e$ où e est une expression régulière de ACTL et Γ est un arbre binaire dont les noeuds internes (binaires) sont de deux types (conjonction et concaténation) et dont les feuilles sont soit une expression régulière de ACTL, soit un noeud spécial représentant 1 ou $\top$:

$\Gamma \quad ::= \quad$ 

Un tel arbre Γ s'interprète trivialement comme une expression régulière $[\Gamma]$:

$$[\textcircled{1}] = 1 \quad [\textcircled{\top}] = \top \quad [\,e\,] = e \quad [\ \Delta \cdot \Theta\ ] = [\Delta] \cdot [\Theta] \quad [\ \Delta \sqcap \Theta\ ] = [\Delta] \sqcap [\Theta]$$

Le séquent $\Gamma \vdash e$ se lit alors : $[\Gamma] \leq e$. On va maintenant définir différents groupes de règles d'inférence pour prouver les séquents.

3.1.1 Règles identité

On a tout d'abord la règle axiome disant que e est toujours contenu dans lui-même. C'est la règle dite axiome (ax) :

$$\frac{}{e \vdash e}\, ax$$

L'autre règle du groupe identité est la règle de coupure (cut) :

$$\frac{\Gamma \vdash e \qquad \Delta, e, \Theta \vdash f}{\Delta, \Gamma, \Theta \vdash f}\, cut$$

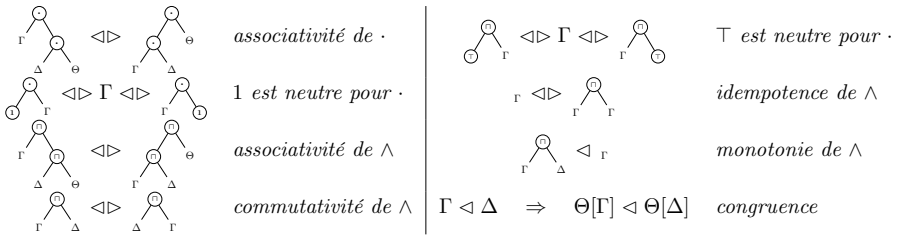
3.1.2 Règles structurelles

Les règles structurelles agissent sur la structure de l'arbre à gauche du séquent et reflètent les propriétés de base de $\top$, 1, $\sqcap$ et $\cdot$. On peut tout regrouper sous une unique règle structurelle gauche :

$$\frac{\Delta \vdash e \qquad \Gamma \lhd \Delta}{\Gamma \vdash e}\, \lhd$$

où la relation binaire $\lhd$ sur les arbres reflète les propriétés des connecteurs 1, $\top$, $\sqcap$ et $\cdot$. (voir figure 4).

FIGURE 4 – règles structurelles usuelles pour $\lhd$



3.1.3 Règles logiques

Les règles logiques spécifient le comportement des connecteurs logiques de ACTL. La plupart des connecteurs disposent d'une (ou plusieurs) règles gauche pour l'introduire à gauche (dans l'arbre) comme connecteur principal d'une formule et d'une ou plusieurs règles droite pour l'introduire comme connecteur principal de l'expression régulière à droite. Elles sont données dans la figure 4.

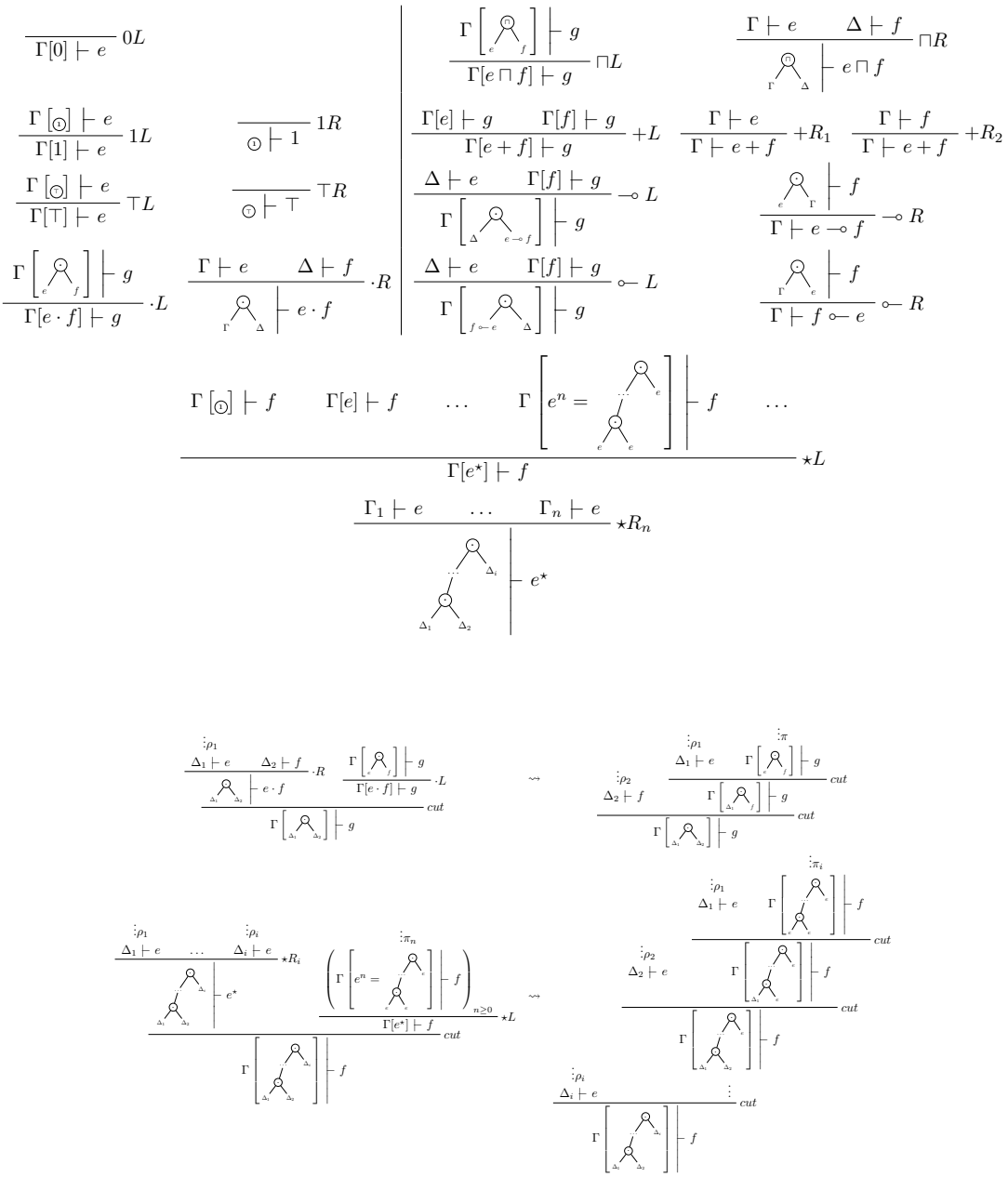
3.2 Elimination des coupures

L'avantage d'un système de calcul des séquents est qu'il procure une forme très structurée aux preuves, bien plus qu'un système à la Hilbert. Il permet une étude syntaxique de la preuve plus poussée et plus agréable. Un des théorèmes majeur que l'on souhaite établir à propos d'un calcul de séquents est l'*élimination des coupures*. Un système qui vérifie l'élimination des coupures est un système dans lequel tout séquent prouvable admet une preuve qui à aucun moment ne fait appel à une règle de coupure. Cela signifie que les règles du calcul sont en quelque sorte "closes par composition" de sorte que la règle de coupure devienne redondante.

Theorem 4. *Le système de calcul de séquents* dACTL_ω *admet l'élimination des coupures : si* $\Gamma \vdash e$ *est prouvable dans* dACTL_ω*, alors il en existe une preuve sans la règle cut.*

Comme dans la plupart des calculs de séquents, la preuve de l'élimination des coupures se présente sous la forme d'un algorithme qui transforme une preuve avec coupure en une preuve sans coupure. Bien souvent, c'est la preuve de terminaison de l'algorithme qui renferme toute la difficulté du problème. Nous passons ici sous silence toutes les commutations de règles utilisées dans cet algorithme et nous nous contentons de donner deux exemples de réduction de coupures :

FIGURE 5 – règles logiques de dACTL_ω



L'élimination des coupures induit un corollaire très intéressant. Etant donné que la règle de coupure est la seule règle qui puisse faire apparaitre dans ses prémisses une expression régulière qui n'est pas une sous-formule d'une expression régulière de la conclusion, on peut établir la propriété de sous-formule pour dACTL_ω :

Corollary 2. *Si $\Gamma \vdash e$ est prouvable dans dACTL_ω, alors il en existe une preuve où toutes les expressions régulières qui y apparaissent sont des sous-formules des expressions régulières initialement présentes dans $\Gamma \vdash e$.*

Notons qu'en général pour un calcul de séquents, la propriété de sous-formule se révèle cruciale pour la recherche de preuves.

3.3 Vers une possible complétude ?

Se pose maintenant la question de la complétude. Le système qu'on a proposé se donnait comme objectifs d'axiomatiser la théorie équationnelle de $\mathsf{dACTL}^\star$. On a en effet un théorème de complétude pour cette classe :

Theorem 5. *Soit un séquent $\Gamma \vdash e$. On a l'équivalence suivante :*

$$\mathsf{dACTL}_\omega \vdash (\Gamma \vdash e) \quad \Leftrightarrow \quad \mathsf{dACTL}^\star \vDash [\Gamma] \leq e$$

Démonstration. L'implication directe (théorème de correction) est évidente et dit simplement que les règles de dACTL_ω sont justifiées par les axiomes de $\mathsf{dACTL}^\star$.

La réciproque est le théorème de complétude à proprement parler. Le principe de la preuve est relativement élémentaire. On va construire ce que l'on nomme *algèbre de Lindenbaum* $\mathcal{L}$ des termes de ACTL. On considère l'ensemble des expressions de ACTL que l'on munit du préordre $e \leq f$ si et seulement si $\mathsf{dACTL}_\omega \vdash (e \vdash f)$. On considère alors la relation d'équivalence $e \sim f$ si et seulement si $e \leq f$ et $f \leq e$, et on note $\mathcal{L}$ l'ensemble des termes quotienté par $\sim$. Désormais, $\ulcorner e \urcorner$ désigne la classe d'équivalence de la formule e pour $\sim$, on le préordre $\leq$ devient un ordre (partiel) dans $\mathcal{L}$.

$\mathcal{L}$ peut par ailleurs être muni d'une structure canonique de treillis avec action où opérateurs de ACTL (0, 1, $\top$, $\sqcap$, $+$, $\cdot$, *, $\multimap$ et $\circ\!\!-$) servent à définir les opérations correspondantes. Par exemple, $\ulcorner e \urcorner \multimap \ulcorner f \urcorner$ vaut par définition $\ulcorner e \multimap f \urcorner$. Cette définition fait sens car la relation d'équivalence $\sim$ est compatible aux opérateurs de ACTL. Il faut ensuite s'assurer que $\mathcal{L}$ vérifie tous les axiomes de $\mathsf{dACTL}^\star$. On présente quelques cas parmi les plus délicats :

- $(K14)$: $b + a \cdot x \leq x \Rightarrow a^\star \cdot b \leq x$. On suppose donc trois termes e, f et g de ACTL vérifient $\ulcorner f \urcorner + \ulcorner e \urcorner \cdot \ulcorner g \urcorner \leq \ulcorner g \urcorner$, c'est-à-dire : $\mathsf{dACTL}_\omega \vdash f + e \cdot g \vdash g$. On veut montrer que $\ulcorner e \urcorner^\star \cdot \ulcorner f \urcorner \leq \ulcorner g \urcorner$, c'est-à-dire $\mathsf{dACTL}_\omega \vdash (e^\star \cdot f \vdash g)$. Par $\cdot L$ et $\star L$, cela revient à donner une preuve de :

$$\begin{array}{c} \cdot \\ e^n \quad f \end{array} \Big| \vdash g$$

  pour tout $n \geq 0$. Pour $n = 0$, il s'agit de montrer que $f \vdash g$, ce qui est clair par l'hypothèse. Maintenant, une fois acquis que $e^n \cdot f \vdash g$, on obtient $e^{n+1} \cdot f \vdash g$ en combinant la preuve donnée au rang n avec le fait que $e \cdot g \vdash g$.

- $(\multimap /\cdot)$: $a \cdot b \leq c \Leftrightarrow b \leq a \multimap c$. Dans un sens, on suppose que $\ulcorner e \urcorner \cdot \ulcorner f \urcorner \leq \ulcorner g \urcorner$, c'est-à-dire que $\mathsf{dACTL}_\omega \vdash (e \cdot f \vdash g)$. Par $\multimap R$, on a alors une preuve de $f \vdash e \multimap g$, ce qu'on voulait. Réciproquement, si on dispose d'une preuve de $f \vdash e \multimap g$, on obtient alors en concaténant par la gauche dans les deux membres $e \cdot f \vdash e \cdot (e \multimap g)$. Ne reste plus qu'à couper avec une preuve de $e \cdot (e \multimap g) \vdash g$, qui est triviale.

- (d) : $(a+b) \sqcap c = (a \sqcap c) + (b \sqcap c)$. Pour trois expressions e, f et g, le sens non trivial est $(\ulcorner e \urcorner + \ulcorner f \urcorner) \sqcap \ulcorner g \urcorner \leq (\ulcorner e \urcorner \sqcap \ulcorner g \urcorner) + (\ulcorner f \urcorner \sqcap \ulcorner g \urcorner)$:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{e \vdash e}\,ax \quad \dfrac{}{g \vdash g}\,ax}{\begin{array}{c}\sqcap \\ e \quad g\end{array} \Big| \vdash e \sqcap g}\,\sqcap R}{\begin{array}{c}\sqcap \\ e \quad g\end{array} \Big| \vdash (e \sqcap g) + (f \sqcap g)}\,+R_1 \qquad \dfrac{\dfrac{\dfrac{}{f \vdash f}\,ax \quad \dfrac{}{g \vdash g}\,ax}{\begin{array}{c}\sqcap \\ f \quad g\end{array} \Big| \vdash f \sqcap g}\,\sqcap R}{\begin{array}{c}\sqcap \\ f \quad g\end{array} \Big| \vdash (e \sqcap g) + (f \sqcap g)}\,+R_2}{\begin{array}{c}\sqcap \\ e+f \quad g\end{array} \Big| \vdash (e \sqcap g) + (f \sqcap g)}\,+L}{(e+f) \sqcap g \vdash (e \sqcap g) + (f \sqcap g)}\,\sqcap L$$

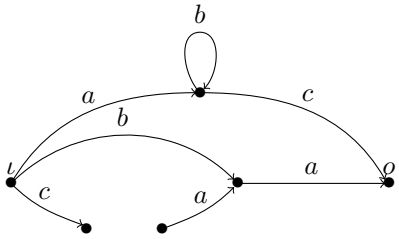
- $(\star/\cdot)$: $a \cdot b^\star \cdot c = \sup_{n \in \omega} a \cdot b^n \cdot c$. Pour trois expressions e, f et g, il est clair que pour tout n, $\ulcorner e \urcorner \cdot \ulcorner f \urcorner^n \cdot \ulcorner g \urcorner \leq \ulcorner e \urcorner \cdot \ulcorner f \urcorner^\star \cdot \ulcorner g \urcorner$ car $\ulcorner f^n \urcorner \leq \ulcorner f^\star \urcorner$. Maintenant, si on a un élément $\ulcorner h \urcorner$ tel que pour tout n, $\ulcorner e \urcorner \cdot \ulcorner f \urcorner^n \cdot \ulcorner g \urcorner \leq \ulcorner h \urcorner$ (c'est-à-dire pour chaque n une preuve de $e \cdot f^n \cdot g \vdash h$), alors on obtient par $\star L$ une preuve de $e \cdot f^\star \cdot g \vdash h$, ce qui montre bien que $\ulcorner e \urcorner \cdot \ulcorner f \urcorner^\star \cdot \ulcorner g \urcorner$ est le supremum des $\ulcorner e \urcorner \cdot \ulcorner f \urcorner^n \cdot \ulcorner g \urcorner$ dans $\mathcal{L}$. Le même raisonnement marche pour l'axiome $(\star/\sqcap)$.

Ainsi $\mathcal{L}$ est dans $\mathsf{dACTL}^\star$. Supposons maintenant qu'on ait deux expressions e et f telles que $\mathsf{dACTL}^\star \vDash e \leq f$. Alors dans $\mathcal{L}$ en particulier, où on interpète les variables libres $a \in \Sigma$ de e et f par $\ulcorner a \urcorner \in \mathcal{L}$, on a $\ulcorner e \urcorner \leq \ulcorner f \urcorner$, ce qui par définition signifie que $\mathsf{dACTL}_\omega \vdash (e \vdash f)$.

□

On s'intéresse à présent à la complétude de dACTL_ω vis-à-vis de la sémantique relationnelle sur certains fragments de ACTL.

FIGURE 6 – $\mathcal{G}(e)$ pour $e = (a \cdot (1 \sqcap b) \cdot c) \sqcap ((b \sqcap (c \cdot \top \cdot a)) \cdot a)$



3.3.1 Complétude sur KL et $\mathsf{KL}^{-\{1,\top\}}$

Un *terme primitif* de KL est un terme sans connecteur disjonctif 0, + ou *. Un terme primitif u s'identifie avec l'unique arbre Γ de dACTL_ω où tous les termes (au niveau des feuilles) sont des symboles de Σ tel que $[\Gamma] = u$. Etant donnée une expression régulière e de KL, on définit par induction sur e son ensemble de termes primitifs, noté $\mathfrak{gt}(e)$:

$$\mathfrak{gt}(a) = \{a\} \quad (a \in \Sigma) \quad \mathfrak{gt}(1) = \{1\} \quad \mathfrak{gt}(\top) = \{\top\}$$
$$\mathfrak{gt}(f \cdot g) = \{s \cdot t \mid s \in \mathfrak{gt}(f), t \in \mathfrak{gt}(g)\} \quad \mathfrak{gt}(f \wedge g) = \{s \wedge t \mid s \in \mathfrak{gt}(f), t \in \mathfrak{gt}(g)\}$$
$$\mathfrak{gt}(0) = \varnothing \quad \mathfrak{gt}(f + g) = \mathfrak{gt}(f) \cup \mathfrak{gt}(g) \quad \mathfrak{gt}(f^\star) = \bigcup_{n \in \omega} \mathfrak{gt}(f^n)$$

Il est facile de voir que dans n'importe quel treillis de Kleene $\star$-continu $\mathcal{A}$ où les symboles de Σ sont interprétés par des éléments de $\mathcal{A}$, on a :

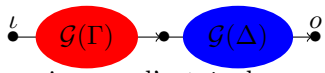
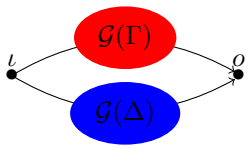
$$e = \sup \{u \mid u \in \mathfrak{gt}(e)\}$$

Il est par ailleurs possible de montrer sans trop de difficulté le lemme suivant :

Lemma 2. *Soient e et f deux expressions régulières de* KL*. On a alors :*

$$\mathsf{dACTL}_\omega \vdash (e \vdash f) \quad \Leftrightarrow \quad \forall u \in \mathfrak{gt}(e) \cdot \exists v \in \mathfrak{gt}(f) \cdot \mathsf{dACTL}_\omega \vdash (u \vdash v)$$

Pour obtenir un théorème de complétude sur le fragment KL, il suffit donc de voir que pour tous u, v termes primitifs de KL, $REL \vDash u \leq v$ entraine $\mathsf{dACTL}_\omega \vdash (u \vdash v)$. Pour ce faire, on recourt à une interprétation classique ([8] et [2]) faisant d'un arbre Γ de dACTL_ω un graphe orienté série-parallèle avec arcs étiquetés par des expressions régulières, noté $\mathcal{G}(\Gamma)$, avec deux points distingués ι (l'entrée) et o (la sortie) :

- $\mathcal{G}(\textcircled{1}) =$  Un seul point qui est à la fois l'entrée et la sortie.
- $\mathcal{G}(\textcircled{\top}) =$  Une entrée et une sortie déconnectées.
- $\mathcal{G}(e) =$  Une entrée et une sortie reliées par un arc étiqueté par e.
- $\mathcal{G}($  $) =$  $\mathcal{G}(\Gamma)$ et $\mathcal{G}(\Delta)$ sont concaténés dans cet ordre en fusionnant la sortie du premier avec l'entrée du second.
- $\mathcal{G}($  $) =$  $\mathcal{G}(\Gamma)$ et $\mathcal{G}(\Delta)$ sont mis en parallèle en fusionnant leur entrée et leur sortie.

En particulier, un terme primitif u de KL se voit comme arbre et donc comme un graphe série-parallèle $\mathcal{G}(u)$ étiqueté par des symboles de Σ. (voir par exemple la figure 6). En fait, ces graphes peuvent être vus comme une généralisation des mots finis sur Σ lorsqu'on ajoute la conjonction $\sqcap$ et la constante $\top$. Il existe d'ailleurs une généralisation des automates finis, les automates de Pétri, capables de reconnaitre des langages de graphes série-parallèles (Pour plus de détails, voir)

On dispose du lemme crucial suivant :

Lemma 3. *Soient u et v deux termes primitifs de KL. Alors $REL \vDash u \leq v$ si et seulement si il existe un morphisme de graphes préservant les étiquettes des arcs $h : \mathcal{G}(u) \to \mathcal{G}(v)$.*

Si les termes u et v ne contiennent ni 1 ni $\top$, on peut montrer qu'un morphisme $h : \mathcal{G}(u) \to \mathcal{G}(v)$ se relève en une preuve $\mathsf{dACTL}_\omega \vdash (u \vdash v)$, de sorte que sur le fragment $\mathsf{KL}^{-\{1,\top\}}$ (où l'on interdit les constantes 1, $\top$ et où l'étoile de Kleene est remplacée par sa variante transitive stricte $^+$), le système dACTL_ω est complet pour la sémantique relationnelle :

Theorem 6. *Soient e et f deux expressions régulières de $\mathsf{KL}^{-\{1,\top\}}$. Alors on a :*

$$REL \vDash e \leq f \quad \Leftrightarrow \quad \mathsf{dACTL}_\omega \vdash (e \vdash f)$$

En revanche, on continue de trouver des contre-exemples à cette complétude en présence des constantes $1, \top$, comme $\top \cdot a \cdot \top \cdot b \cdot \top = \top \cdot b \cdot \top \cdot a \cdot \top$ et $1 \sqcap a \sqcap b = (1 \sqcap a) \cdot (1 \sqcap b)$. La solution consiste alors à renforcer dACTL_ω en prenant pour la relation $\lhd$ de la règle structurelle gauche non pas les règles structurelles usuelles mais la relation :

$$\Gamma \lhd \Delta \quad \Leftrightarrow \quad \exists h : \mathcal{G}(\Delta) \to \mathcal{G}(\Gamma)$$

On note $\mathsf{dACTL}_\omega^{\blacktriangleleft}$ le système de séquents ainsi obtenu, qui est complet pour la sémantique relationnelle sur KL :

Theorem 7. *Soient e et f deux expressions régulières de KL. Alors :*

$$REL \vDash e \leq f \quad \Leftrightarrow \quad \mathsf{dACTL}_\omega^{\blacktriangleleft} \vdash (e \vdash f)$$

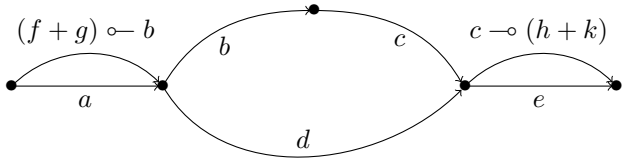
3.3.2 Questions ouvertes pour ACTL

Nous avions vu auparavant comme frein à la complétude l'égalité $((a \multimap a) \multimap 1)^2 = (a \multimap a) \multimap 1$. Cependant, avec le calcul $\mathsf{dACTL}_\omega^{\blacktriangleleft}$, cela devient prouvable. En revanche, d'autres problèmes plus profond rendent la complétude impossible, comme $(((\top \multimap a) \circ\!\!- \top) \multimap b) + a = \top$. En effet, cette égalité est classiquement vraie mais pas intuitionistement. Or dACTL_ω et $\mathsf{dACTL}_\omega^{\blacktriangleleft}$ sont par essence intuitionistes (Les lecteur intéressé par ce point pourra consulter le rapport de stage à ce sujet. L'idée essentielle est que notre calcul de séquents est limité par le fait de n'avoir à droite qu'une formule et non un arbre). Il convient donc de retirer $\top$, et par la même occation 0 puisque $0 \multimap a$ est équivalent à $\top$.

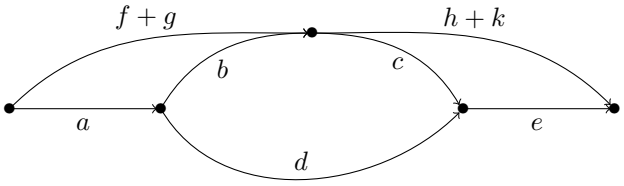
Il semble néanmoins que $\mathsf{dACTL}_\omega^{\blacktriangleleft}$ demeure incomplet sur le fragment sans constantes $\mathsf{ACTL}^{-\{0,1,\top\}}$. Voici un exemple qui semble invalider l'hypothèse de complétude :

$$REL \vDash (((f+g) \circ\!\!- b) \sqcap a) \cdot ((b \cdot c) \sqcap d) \cdot ((c \multimap (h+i)) \sqcap e) \leq (f \cdot h) + (a \cdot ((b \cdot i) \sqcap (d \cdot e))) + (((g \cdot c) \sqcap (a \cdot d)) \cdot e)$$

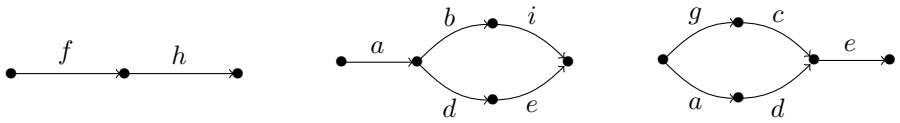
où $a, b, c, d, e, f, g, h, k \in \Sigma$. La preuve de ceci peut s'effectuer de manière "visuelle" ; voici le graphe associé au terme de gauche :



En instantiant la variable du quantificateur universel des formules $(f+g) \circ\!\!- b$ et $c \multimap (h+i)$ sur le point le plus élevé du graphe, les hypothèses à montrer sont triviales ($b \leq b$ et $c \leq c$) et on obtient alors le graphe suivant, qui remarquons-le au passage n'est pas un graphe série-parallèle :



Il est alors clair que dans chacun des quatre cas induits par les deux sommes $f+g$ et $h+k$ (remplacer dans le graphe $f+g$ par f ou g et $h+k$ par h ou k), au moins l'un des trois graphes suivants s'envoie par morphisme dans le graphe obtenu :



D'où l'inclusion $REL \vDash (((f+g) \circ\!\!- b) \sqcap a) \cdot ((b \cdot c) \sqcap d) \cdot ((c \multimap (h+i)) \sqcap e) \leq (f \cdot h) + (a \cdot ((b \cdot i) \sqcap (d \cdot e))) + (((g \cdot c) \sqcap (a \cdot d)) \cdot e)$.

Il semble en revanche raisonnable de penser qu'une telle inclusion n'admet pas de preuve dans $\mathsf{dACTL}^{\blacktriangleleft}_{\omega}$, puisque dans ce systèmes tous les graphes correspondants aux arbres doivent être série-parallèle. Or il semble que quelles que soient les règles morphismes appliquées avant d'être contraint d'utiliser $\multimap L$ ou $\circ\!\!- L$, la structure série-parallèle du graphe ne puisse être conservée si l'on souhaite instantier les variables de $(f+g) \circ\!\!- b$ et $c \multimap (h+k)$ "au bon point" : autrement dit cette structure série-parallèle impose trop de contraintes de sorte que les règles $\multimap L$ et $\circ\!\!- L$ ne sont pas assez expressives.

La question de la complétude de dACTL_{ω} vis-à-vis d'autres fragments de ACTL doit encore être étudiée et complèterait de nombreux résultats déjà connus dans le domaine (par exemple, [4], [5] et [1]).

Références

[1] H. Andréka and S. Mikulás. Lambek calculus and its relational semantics : Completeness and incompleteness. *Journal of Logic, Language and Information*, 1994.

[2] P. Brunet and D. Pous. Petri automata for kleene allegories. 2015.

[3] J. Brzozowski. Derivatives of regular expressions. 1964.

[4] W. Buszkowski. On action logic : Equational theories of action algebras. *Journal of Logic and Computation*.

[5] W. Buszkowski. On the complexity of the equational theory of relational action algebras. *Journal of Logic and Computation*.

[6] O. Carton. *Langages formels, calculabilité et complexité*. 2008.

[7] J. Conway. *Regular Automata and Finite Machines*. 1966.

[8] P. Freyd and A. Scedrov. *Categories, Allegories*. 1990.

[9] D. Kozen. A completeness theorem for kleene algebras and the algebra of regular events. 1993.

[10] E. Palka. An infinitary sequent system for the equational theory of star-continuous action lattices. *Fundamenta Informaticae*, 2007.

[11] V. Pratt. Action logic and pure induction. *Logics in AI*, 1990.

[12] K. Thomson. Regular expression search algorithm. 1968.