



## Théorie des langages

Olivier Ridoux

### ► To cite this version:

| Olivier Ridoux. Théorie des langages. École d'ingénieur. France. 2020. hal-02505877

HAL Id: hal-02505877

<https://hal.science/hal-02505877>

Submitted on 11 Mar 2020

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

comme **texte**.

Ce module est accompagné de travaux pratiques utilisant des outils modernes magique exemple de **héroe du marché** propose à une **bonne ingénierie**. La TL répond à ces questions par des résultats formels puisants qui ont permis de développer des outils qui sortent tous les systèmes informatiques : compilateurs, générateurs de documents structurés, etc. La TL offre donc un moyen de répondre à ces questions par des résultats formels puisants qui ont retrouvé les éléments de signalisation dans un message ?

- « *Formal Languages: Origins and Directions* », Gréebach (Annals Hist. Comp., 1981)
- « *The New Turing Omnibus* », Dewdney (Owl Books, 2001)
- « *Mathematical Techniques for Information* », Arnold (Gesellschaft (Dundod, 2005)
- « *Methods for Information* », Vélu (Dundod, 2005)

## Bibliographie

La TL répond à ces questions par des résultats formels puisants qui ont retrouvé les éléments de signalisation dans un message ?

retracer les messages de leur signalisation, numérotation, etc. ? Comment insister sur la **sous-génération** et des cas négatifs pour détecter la **sur-génération** ?

2. **Ampliante** : une grammaire peut donner plusieurs décomposition structurées à la même phrase, comme dans **Alice a vu Bob avec sa langue-vue** (de quoi le faire décrire par une autre pour la transmettre) ou **le sujet et le violon** (de quoi le faire décrire par une autre pour la transmettre) mais que dans la seconde **Alice** est le sujet et le **violon** est le sujet de la phrase, ce qui rend la grammaire ambiguë.

3. **Complexe** : une grammaire engendre donc des mots possibles comme **Alice aime le violon**, mais aussi **Bob aime apprendre le violon**. L'enjeu de la grammaire est de trouver les mots possibles comme **Alice aime le violon**, mais aussi **Bob aime apprendre le violon**.

2. **Ampliante** : une grammaire peut donner plusieurs décomposition structurées à la même phrase, comme dans **Alice a vu Bob avec sa langue-vue** (de quoi le faire décrire par une autre pour la transmettre) ou **le sujet et le violon** (de quoi le faire décrire par une autre pour la transmettre) mais que dans la seconde **Alice** est le sujet et le **violon** est le sujet de la phrase, ce qui rend la grammaire ambiguë.

3. **Complexe** : une grammaire engendre donc des mots possibles comme **Alice aime le violon**, mais aussi **Bob aime apprendre le violon**. L'enjeu de la grammaire est de trouver les mots possibles comme **Alice aime le violon**, mais aussi **Bob aime apprendre le violon**.

En pratique, utiliser la TL revient à **programmer** avec des expressions régulières, des protocoles, etc., mais aussi architecture, croissance des planètes, traces d'exécution, structures séquentielles organiques : programmes, documents, traces d'exécution, des langages. Ils construisent les outils de base pour formaliser une grande variété de langages, mais pas de la même classe), et les automates **engendrent** des mots et des langages (mais pas de la même classe).

Les expressions régulières et les grammars **engendent** des mots et des langages

des langages, mais pas de la même classe), et les automates **acceptent** des mots et des langages.

Noter que la grammaire est appellée **grammaire algébrique**, note **G<sub>A</sub>**. D'autres formes existent.

Une grammaire **engendre** un langage suivante :

**SEM( $\epsilon$ ) = { $\epsilon$ }**      **SEM( $a$ ) = { $a$ }**      **SEM( $b$ ) = { $b$ }** ...

**SEM( $r_1 \cdot r_2$ ) = SEM( $r_1$ ) U SEM( $r_2$ )**      union de deux langages

**SEM( $r_1 \cdot r_2$ ) = SEM( $r_1$ ) \* SEM( $r_2$ )**      produit de deux langages

**SEM( $r^*$ ) = SEM( $r$ )<sup>\*</sup>**      itération d'un langage

RE constitue donc un langage qui décrit des langages ; c'est un **métalangage**.

Ex. la **##4/4(CDFCBCACCBAFFCB)\*(EEEEEBBBBBBBBBAAAAA)\*CDECBCA** décrit les premières notes du **YELLOW SUBMARINE** des Beatles (C=Do, D=Ré, etc.) où **V** est **{#, /, A, B, C, D, E, F, G}**. Ici, la **re** engendre des **mots-partitions**.

On qualifie de **rationnels** les langages qu'il est possible de caractériser par une **re**.

L'ensemble des langages n'étant pas dénombrable mais RE l'étant, les langages ne sont pas tous rationnels ! Mais alors, lesquels le sont ? Lesquels ne le sont pas ?

**Automates finis (fsa, pour finite state automata)** : Ils sont un exemple du style **intensionnel** de caractérisation des langages. La théorie décrit comment former des **fsa**, et comment ceux-ci **acceptent** des mots. On note un **fsa** générique **M** (pour **Machine**). Chaque **fsa** est fini. Les **fsa** constituent un ensemble **dénombrable** : **FSA**.

Un **fsa** est défini par un vocabulaire **V**, un ensemble fini d'états, noté **Q**, contenant un état initial **q<sub>0</sub> ∈ Q**, et des états finaux **F ⊂ Q**. Il est aussi défini par une relation de transition **δ** constituée d'un ensemble de triplets (**q, a, q'**), souvent notés **q →<sup>a</sup> q'**, où **a ∈ V** et **q, q' ∈ Q**. Les **fsa** **acceptent** des mots de la façon suivante :

**ACCEPT( $\epsilon, q$ )** est vraissi **q ∈ F**      les états finaux acceptent le mot vide, ils sont les seuls à le faire

**ACCEPT( $a \cdot m, q$ )** est vraissi **E q' tq q →<sup>a</sup> q'**. **ACCEPT(m, q<sub>0</sub>)**

un état quelconque accepte un mot si il permet une transition

qui accepte sa première lettre et qui conduit à un état qui accepte le reste

• un **fsa** **accepte** un mot m ssi **ACCEPT(m, q<sub>0</sub>)**.

Un **fsa** **reconnait** un langage si il en accepte tous les mots, et seulement eux.

La TL ne s'arrête pas aux langages rationnels. Que certains langages ne soient pas rationnels fait se demander si on peut construire le même genre d'édifice pour eux, ou une partie d'entre eux. Et la réponse est oui. Par exemple, on montre que le langage des parenthèses est l'archétype d'une famille de langages qui sont bien caractérisés par une autre forme d'expression, les **grammaires algébriques**, et par des **automates à pile**, que les deux ont exactement la même puissance d'expression, qui est bien testée par une variante du Lemme de l'étoile, le tout constituant la théorie des **langages algébriques** (on dit aussi **langage sans contexte**). On montre aussi que les langages ne sont pas tous algébriques, et on recommande en définissant des classes de langages toujours plus larges qu'on ordonne dans la **Hierarchie de Chomsky**, du nom de Noam Chomsky (1928...), autre grand nom de la TL. Dans cette hiérarchie, les classes de langages sont désignées par un numéro ; ex. les langages rationnels sont les langages de **type 3** (voir page 7).

Le Lemme de l'étoile se clôt cette présentation des langages rationnels. Il faut surtout en retenir l'architecture générale : l'échafaudage **lettre-mot-langage**, les moyens de **caractériser des langages**, par **expression** (extensionnelle) et par **automate** (intensionnel), l'équivalence formelle entre ces moyens, l'identification de leur **puissance d'expression**, et la formalisation de cette puissance d'expression dans un **critère d'appartenance**.

La TL ne s'arrête pas aux langages rationnels. Que certains langages ne soient pas rationnels fait se demander si on peut construire le même genre d'édifice pour eux, ou une partie d'entre eux. Et la réponse est oui. Par exemple, on montre que

le langage des parenthèses est l'archétype d'une famille de langages qui sont bien

caractérisés par une autre forme d'expression, les **grammaires algébriques**, et par

des **automates à pile**, que les deux ont exactement la même puissance d'expression,

qui est bien testée par une variante du Lemme de l'étoile, le tout constituant

la théorie des **langages algébriques** (on dit aussi **langage sans contexte**). On

montre aussi que les langages ne sont pas tous algébriques, et on recommande

en définissant des classes de langages toujours plus larges qu'on ordonne dans la

**Hierarchie de Chomsky**, du nom de Noam Chomsky (1928...), autre grand nom de

la TL. Dans cette hiérarchie, les classes de langages sont désignées par un numé-

ro ; ex. les langages rationnels sont les langages de **type 3** (voir page 7).

Depuis Chomsky, on privilie des formes de caractérisation des langages d'un mot à un langage. Mais

juste la même était de caractériser l'appartenance d'un mot à un langage. Mais

ce n'est pas suffisant car les applications qui lisent des données formates

veulent en plus utiliser le format comme un guide pour leur calcul.

Depuis Chomsky, on privilie des formes de caractérisation des langages d'un mot à un langage. Mais

ce n'est pas suffisant car les applications qui lisent des données formates

veulent en plus utiliser le format comme un guide pour leur calcul.

Depuis Chomsky, on privilie des formes de caractérisation des langages d'un mot à un langage. Mais

ce n'est pas suffisant car les applications qui lisent des données formates

veulent en plus utiliser le format comme un guide pour leur calcul.

Depuis Chomsky, on privilie des formes de caractérisation des langages d'un mot à un langage. Mais

ce n'est pas suffisant car les applications qui lisent des données formates

veulent en plus utiliser le format comme un guide pour leur calcul.

Depuis Chomsky, on privilie des formes de caractérisation des langages d'un mot à un langage. Mais

ce n'est pas suffisant car les applications qui lisent des données formates

veulent en plus utiliser le format comme un guide pour leur calcul.

Depuis Chomsky, on privilie des formes de caractérisation des langages d'un mot à un langage. Mais

ce n'est pas suffisant car les applications qui lisent des données formates

veulent en plus utiliser le format comme un guide pour leur calcul.

Depuis Chomsky, on privilie des formes de caractérisation des langages d'un mot à un langage. Mais

ce n'est pas suffisant car les applications qui lisent des données formates

veulent en plus utiliser le format comme un guide pour leur calcul.

Depuis Chomsky, on privilie des formes de caractérisation des langages d'un mot à un langage. Mais

ce n'est pas suffisant car les applications qui lisent des données formates

veulent en plus utiliser le format comme un guide pour leur calcul.

Depuis Chomsky, on privilie des formes de caractérisation des langages d'un mot à un langage. Mais

ce n'est pas suffisant car les applications qui lisent des données formates

veulent en plus utiliser le format comme un guide pour leur calcul.

Depuis Chomsky, on privilie des formes de caractérisation des langages d'un mot à un langage. Mais

ce n'est pas suffisant car les applications qui lisent des données formates

veulent en plus utiliser le format comme un guide pour leur calcul.

Depuis Chomsky, on privilie des formes de caractérisation des langages d'un mot à un langage. Mais

ce n'est pas suffisant car les applications qui lisent des données formates

veulent en plus utiliser le format comme un guide pour leur calcul.

Depuis Chomsky, on privilie des formes de caractérisation des langages d'un mot à un langage. Mais

ce n'est pas suffisant car les applications qui lisent des données formates

veulent en plus utiliser le format comme un guide pour leur calcul.

Depuis Chomsky, on privilie des formes de caractérisation des langages d'un mot à un langage. Mais

ce n'est pas suffisant car les applications qui lisent des données formates

veulent en plus utiliser le format comme un guide pour leur calcul.

Depuis Chomsky, on privilie des formes de caractérisation des langages d'un mot à un langage. Mais

ce n'est pas suffisant car les applications qui lisent des données formates

veulent en plus utiliser le format comme un guide pour leur calcul.

Depuis Chomsky, on privilie des formes de caractérisation des langages d'un mot à un langage. Mais

ce n'est pas suffisant car les applications qui lisent des données formates

veulent en plus utiliser le format comme un guide pour leur calcul.

Depuis Chomsky, on privilie des formes de caractérisation des langages d'un mot à un langage. Mais

ce n'est pas suffisant car les applications qui lisent des données formates

veulent en plus utiliser le format comme un guide pour leur calcul.

Depuis Chomsky, on privilie des formes de caractérisation des langages d'un mot à un langage. Mais

ce n'est pas suffisant car les applications qui lisent des données formates

veulent en plus utiliser le format comme un guide pour leur calcul.

Depuis Chomsky, on privilie des formes de caractérisation des langages d'un mot à un langage. Mais

ce n'est pas suffisant car les applications qui lisent des données formates

veulent en plus utiliser le format comme un guide pour leur calcul.

Depuis Chomsky, on privilie des formes de caractérisation des langages d'un mot à un langage. Mais

ce n'est pas suffisant car les applications qui lisent des données formates

veulent en plus utiliser le format comme un guide pour leur calcul.

Depuis Chomsky, on privilie des formes de caractérisation des langages d'un mot à un langage. Mais

ce n'est pas suffisant car les applications qui lisent des données formates

veulent en plus utiliser le format comme un guide pour leur calcul.

Depuis Chomsky, on privilie des formes de caractérisation des langages d'un mot à un langage. Mais

ce n'est pas suffisant car les applications qui lisent des données formates

veulent en plus utiliser le format comme un guide pour leur calcul.

Depuis Chomsky, on privilie des formes de caractérisation des langages d'un mot à un langage. Mais

## Prérequis : logique et théorie des ensembles - lire, apprendre, comprendre.

Pliage : recto visible (autre face), traits gris rentrants, traits rouges saillants. Découper selon le trait rouge entre les deux ● de l'autre face, puisachever le pliage.

### 1 Théorie naïve des ensembles finis

Un **ensemble** est une collection d'objets où la position et la multiplicité de comptent pas. On appelle **éléments** les objets de la collection, et on dit qu'un élément **appartient** à un ensemble. Cela se note  $e \in E$  pour «  $e$  appartient à  $E$  » ou  $e, e' \in E$  pour «  $e$  et  $e'$  appartiennent à  $E$  ».

On note une collection entre accolades  $\{\dots\}$ . Ex.  $\{a, b, c\}$  est une collection, et puisque la position ne compte pas, la collection  $\{b, c, a\}$  dénote le même ensemble, et puisque la multiplicité ne compte pas non plus, la collection  $\{a, b, a, c\}$  dénote encore le même ensemble.

Dans la théorie naïve des ensembles on ne considère que des objets qu'il est possible de distinguer les uns des autres, mais à part cela n'importe quoi peut être un objet, y compris un ensemble.

Une collection vide,  $\{\}$ , constitue un ensemble particulier qu'on appelle **l'ensemble vide**, et qu'on note  $\emptyset$  ou  $\{\}$ . On appelle **singleton** un ensemble qui n'a qu'un élément. L'ensemble  $\{\emptyset\}$  est donc un singleton. Ne pas confondre l'élément  $a$  et le singleton  $\{a\}$ . Les confondre revient à commettre une erreur de type en programmation, comme confondre **int** et **int []**. Ne pas confondre non plus  $\emptyset$  et  $\{\emptyset\}$ . Ce serait comme confondre un pointeur nul et un pointeur sur un pointeur nul.

La collection des éléments d'un ensemble s'appelle **l'extension** de l'ensemble. On peut aussi spécifier les éléments d'un ensemble par une propriété qui les distingue de ceux qui n'appartiennent pas à l'ensemble. Cette propriété s'appelle **l'intension** de l'ensemble (l'intension n'a rien à voir avec l'intention, ou dessein). On exprime une intension soit par une phrase en prenant le risque d'être imprécis ou ambigu, soit par une formule logique en prenant le risque d'être fastidieux. N'importe quelle formule logique fausse (contradictoire ou absurde) est une intension de  $\emptyset$ .

Ex. « **les lettres communes aux mots "bancale"** et **"tabac"** » est une intension possible pour l'ensemble constitué de la collection  $\{a, b, c\}$ . Les **3 premières lettres de l'alphabet** est une intension équivalente. Un ensemble peut avoir de nombreuses intensions équivalentes, certaines même difficiles à comparer.

Si tous les éléments d'un ensemble  $E_1$  appartiennent à un ensemble  $E_2$ , on dit que  $E_1$  est **inclus** dans  $E_2$ , et on le note  $E_1 \subseteq E_2$ . On dit aussi que  $E_1$  est un **sous-ensemble** ou une **partie** de  $E_2$ . On note  $E_1 \subsetneq E_2$  pour signifier l'inclusion sans égalité possible (on parle alors de sous-ensemble strict), et  $E_1 \subseteq E_2$  pour insister sur l'égalité possible.  $\emptyset$  est un sous-ensemble de chaque ensemble, et tout ensemble est sous-ensemble (non-strict) de lui-même. Si un ensemble a  $n$  éléments, il a  $2^n$  sous-ensembles (non-strict).

### N'oubliez jamais !

Un **modèle** est toujours **imparfait**, et tout ce qu'on peut lui demander est d'être **utile** (d'après George Box 1978, voir aussi a contrario les cartes à l'échelle 1/1 de Jorge Luis Borges 1946 et Lewis Carroll 1893 qui sont parfaites mais inutiles).

### 2 Ensembles infinis

Quand un ensemble comporte un nombre fini d'éléments on appelle ce nombre la **cardinalité** de l'ensemble, mais ce concept doit être revisité pour les ensembles infinis car il n'y a pas de nombre naturel pour mesurer l'infini.

Si on peut établir une bijection entre deux ensembles on dit qu'ils ont la même **cardinalité**. Si un ensemble a la même cardinalité qu'une partie stricte de lui-même, on dit qu'il est **infini**. Les autres ensembles sont dits **finis**. Par exemple, la fonction  $x \mapsto 2x$  forme une bijection entre l'ensemble des **entiers naturels** et l'ensemble des **entiers pairs**, qui en est une partie stricte. La cardinalité modélise la quantité d'éléments d'un ensemble, mais ne peut être assimilée à un entier naturel que pour les ensembles finis.

Les ensembles infinis n'ont pas tous la même cardinalité ; certains sont « plus gros que d'autres ». Si un ensemble a la même cardinalité que celui des entiers naturels, on dit qu'il est **dénombrable**. Il est **indénombrable** sinon.

Les principaux résultats sont les suivants :

- L'ensemble des entiers naturels est dénombrable (par définition), ainsi que l'ensemble des paires d'entiers naturels, des rationnels, des triplets et nuplets, des suites finies de nombres et de symboles, et des textes (et donc des expressions et des programmes), par un **théorème de Georg Cantor (1878)**.

- L'ensemble des fonctions sur les entiers naturels n'est pas dénombrable, ainsi que l'ensemble des parties des entiers naturels, et l'ensemble des nombres réels, par un autre **théorème de Georg Cantor (1874 et 1891)**.

- L'ensemble des parties d'un ensemble  $E$  infini est toujours « **plus gros** » que  $E$ .

On constate que beaucoup d'ensembles intéressants (ex. les réels ou les fonctions) ne sont pas dénombrables, alors que les programmes le sont. Cela indique que ces ensembles intéressants ne sont pas calculables par des programmes.

### 1 Logique des prédictats

Dans sa définition la plus naïve, la **logique des prédictats** (ou **calcul des prédictats**) est la formalisation de la logique employée tous les jours dans les activités un tant soit peu scientifiques. Elle permet d'exprimer des **jugements** sous forme de formules, et de décider formellement si elles sont **vraies** ou **fausses**. Les formules de la logique des prédictats sont définies de la façon suivante :

**Prédicats atomiques** ou (**atomes**) : Ce sont des formules élémentaires, c-à-d. pas composées de sous-formules, qui expriment des jugements sur des objets. Nous ne nous prononçons pas sur la notation à employer dans les atomes ; cela n'a pas vraiment d'importance. Par exemple,

- Rennes est une capitale, où Rennes est l'objet et être une capitale est le prédictat.
- Rennes est la capitale de la France, où Rennes et la France sont des objets et être la capitale de est le prédictat.
- $x > y$ , où  $x$  et  $y$  sont les objets, et où  $>$  (être plus grand que) est le prédictat.
- 1273 + 556 est un nombre premier, où 1273 + 556 est l'objet et être un nombre premier est le prédictat.
- Le Grand théorème de Fermat est vrai, où le Grand théorème de Fermat est l'objet et être vrai est le prédictat.
- $P \neq NP$ , où  $P$  et  $NP$  sont les objets et  $\neq$  (être différent) le prédictat.

On peut attribuer une valeur de vérité, **Vrai** ou **Faux**, à un atome, en se référant à une réalité de terrain, comme pour **Rennes**, à des théories et des calculs, comme pour la primalité de **1273 + 556**, ou à des preuves externes, comme celle du **Grand théorème de Fermat**. Parfois on ne peut pas, par manque d'information, comme pour  $x > y$ , ou parce qu'on ne sait vraiment pas, comme pour  $P \neq NP$ .

**Formules propositionnelles** : Ce sont des formules qui sont constituées de sous-formules, élémentaires ou non, qui sont reliées par des connecteurs logiques, généralement  $\wedge, \vee, \neg$ , ou  $\Rightarrow$ .

- conjonction**,  $\phi_1 \wedge \phi_2$  : relie deux formules pour en former une troisième qui n'est vraie que si les deux premières le sont. Le connecteur  $\wedge$  est **commutatif**, **associatif**, et a pour **élément neutre** la valeur de vérité **Vrai**. Cela permet la notation de conjonction étendue comme  $\wedge_{i \in [1,n]} \phi_i$ . Si on convenait que **Faux < Vrai**,  $\phi_1 \wedge \phi_2$  serait le minimum de  $\phi_1$  et  $\phi_2$ . Si on convenait que **Faux = 0** et **Vrai=1**,  $\phi_1 \wedge \phi_2$  serait  $\phi_1 \times \phi_2$ .
- disjonction**,  $\phi_1 \vee \phi_2$  : relie deux formules pour en former une troisième qui n'est vraie que si au moins une des deux premières l'est. On parle de disjonction **exclusive** si une et une seule de ces formules peut être vraie. Le connecteur  $\vee$  est **commutatif**, **associatif**, et a pour **élément neutre** la valeur de vérité **Faux**. Cela permet la notation de disjonction étendue comme  $\vee_{i \in [1,n]} \phi_i$ . Si on convenait que **Faux < Vrai**,  $\phi_1 \vee \phi_2$  serait le maximum de  $\phi_1$  et  $\phi_2$ . Si on convenait que **Faux = 0** et **Vrai=1**,  $\phi_1 \vee \phi_2$  serait  $1 - (1 - \phi_1) \times (1 - \phi_2)$ .
- implication**,  $\phi_1 \Rightarrow \phi_2$  : relie deux formules pour en former une troisième qui est fausse si  $\phi_1$  est vraie alors que  $\phi_2$  est fausse. Le connecteur  $\Rightarrow$  n'est ni **commutatif** ni **associatif**. Si on convenait que **Faux < Vrai**,  $\phi_1 \Rightarrow \phi_2$  serait **Vrai** si  $\phi_1 \leq \phi_2$ .
- négation**,  $\neg \phi$  : constitue une formule qui est fausse si  $\phi$  est vraie. Si on convenait que **Faux = 0** et **Vrai=1**,  $\neg \phi$  serait  $1 - \phi$ .

**Formules quantifiées** : Ce sont des formules, élémentaires ou non, dont la valeur de vérité s'évalue par rapport à un ensemble, plutôt que par rapport à des objets.

- quantification universelle**,  $\forall x . \phi(x)$  : constitue une formule qui est vraie si  $\phi$  est vraie de tous les éléments du domaine. Si le domaine est fini, la quantification universelle est juste une conjonction des applications de  $\phi$  à tous les éléments du domaine :  $\forall x \in \{e_1, e_2, \dots, e_n\}. \phi(x)$ ssi  $\wedge_{i \in [1,n]} \phi(e_i)$ .
- quantification existentielle**,  $\exists x . \phi(x)$  : constitue une formule qui est vraie si  $\phi$  est vraie d'au moins un élément du domaine. Si le domaine est fini, la quantification existentielle est juste une disjonction des applications de  $\phi$  à tous les éléments du domaine :  $\exists x \in \{e_1, e_2, \dots, e_n\}. \phi(x)$ ssi  $\vee_{i \in [1,n]} \phi(e_i)$ .

**Parenthèses et priorité des opérateurs** : Il est courant d'assigner des priorités d'opérateurs aux différents connecteurs et quantificateurs afin de spécifier comment se lisent les formules complexes en l'absence de parenthèses. Cependant, il n'est jamais très prudent de se poser trop lourdement sur les priorités des opérateurs quand le coût de quelques parenthèses est si faible devant le coût d'une grosse erreur. Il vaut mieux ne pas être avare de parenthèses, et nous proposons d'utiliser les parenthèses rondes (...) pour structurer les connecteurs, et les parenthèses carrées (ou crochets, [...] ) pour les quantificateurs.

### 2 Idiomes logiques

En pratique, on n'utilise pas n'importe quelle formule de la logique des prédictats. On a tendance à utiliser des expressions idiomatiques qu'il convient de reconnaître et interpréter correctement au premier coup d'œil.

**Quantifications dans un domaine** : Très souvent, on écrit des formules comme  $\forall x \in E . \phi(x)$  ou  $\forall x \in E . \psi(x) . \phi(x)$ . C'est une façon de dire dans quel domaine doit être évaluée la quantification. Ces formules doivent être lues  $\forall x . [x \in E \Rightarrow \phi(x)]$  et  $\forall x . [\psi(x) \Rightarrow \phi(x)]$ . Une conséquence directe est qu'une quantification universelle sur un domaine vide est trivialement vraie. Cela paraît une situation étrange, mais c'est banal en informatique, spécialement quand on considère les cas d'initialisation dans les programmes : ex. **Tous les utilisateurs sont ...** lorsqu'il n'y a pas encore d'utilisateurs. De la même façon, on écrit des formules comme  $\exists x \in E . \phi(x)$  ou  $\exists x \in E . \psi(x) . \phi(x)$ . Ces formules doivent être lues  $\exists x . [x \in E \wedge \phi(x)]$  et  $\exists x . [\psi(x) \wedge \phi(x)]$ . On voit alors qu'une quantification existentielle sur un domaine vide est trivialement fausse.

**Cascades d'implications** : On écrit parfois des formules comme  $\phi_1 \Rightarrow \phi_2 \Rightarrow \phi_3$ , qui pourrait être lue  $(\phi_1 \Rightarrow \phi_2) \Rightarrow \phi_3$  ou  $\phi_1 \Rightarrow (\phi_2 \Rightarrow \phi_3)$ . Dans le premier cas, on exprime que  $\phi_1 \Rightarrow \phi_2$  est une hypothèse qui suffit à démontrer  $\phi_3$ . Dans le second cas, la formule est équivalente à  $(\phi_1 \wedge \phi_2) \Rightarrow \phi_3$ , donc  $(\phi_2 \wedge \phi_3) \Rightarrow \phi_3$ , et donc  $\phi_2 \Rightarrow (\phi_1 \Rightarrow \phi_3)$ . Conclusion, faire des économies de bouts de parenthèses avec discernement !

**Formules de De Morgan (Augustus De Morgan, 1806-1871)** : La négation a à voir avec le complémentaire d'une situation, mais le complémentaire d'une situation compliquée est souvent encore plus compliquée que la situation. Les formules de De Morgan (qui ne sont pas toutes dues à De Morgan) peuvent être mises à profits pour pousser les négations vers l'intérieur des formules où elles s'appliqueront à des formules plus petites donc plus simples.

- $\neg(\phi_1 \wedge \phi_2)$  est équivalent à  $\neg \phi_1 \vee \neg \phi_2$ , et  $\neg(\phi_1 \vee \phi_2)$  est équivalent à  $\neg \phi_1 \wedge \neg \phi_2$ .
- $\neg(\phi_1 \Rightarrow \phi_2)$  est équivalent à  $\phi_1 \wedge \neg \phi_2$ .
- $\neg \neg \phi$  est équivalent à  $\phi$ . Pose beaucoup plus de difficultés que la taille de l'identité ne le laisse penser, mais est vrai pour l'usage usuel de la logique. Se rappeler combien nous humains sommes peu doués pour les doubles négations.
- $\neg \forall x . \phi(x)$  est équivalent à  $\exists x . \neg \phi(x)$ , et  $\neg \exists x . \phi(x)$  est équivalent à  $\forall x . \neg \phi(x)$ .

**Cascades de quantifications** : On imbrique souvent les quantifications. Certaines imbrications doivent faire réfléchir, et d'autres moins.

- $\exists x . \exists y . \phi(x, y)$  est équivalent à  $\exists y . \exists x . \phi(x, y)$  qu'on note souvent  $\exists x, y . \phi(x, y)$ .
- $\forall x . \forall y . \phi(x, y)$  est équivalent à  $\forall y . \forall x . \phi(x, y)$  qu'on note souvent  $\forall x, y . \phi(x, y)$ .
- $\exists x . \exists y . \phi(x, y)$  exprime que pour chaque  $x$  il y a un  $y$ , qui peut dépendre de  $x$ , qui a la propriété désirée. Ex. dans  $\forall x . \exists y . x \times y = 0$ , le même  $y$  convient pour tous les  $x$ , mais dans  $\forall x . \exists y . x \times y = 1$ , à tout  $x$  correspond un  $y$  différent. Une variable existentielle est donc implicitement une fonction de toutes les variables universelles qui la précédent.
- $\exists x . \forall y . \phi(x, y)$  exprime qu'un même  $x$  a la propriété  $\phi$  pour tous les  $y$ . Ex.  $\exists x . \forall y . x \times y = 0$  est vrai, mais pas  $\exists x . \forall y . x \times y = 1$ .
- $\exists x . [\psi \wedge \phi(x)]$  est équivalent à  $\psi \wedge \exists x . \phi(x)$  et  $\forall x . [\psi \wedge \phi(x)]$  est équivalent à  $\psi \wedge \forall x . \phi(x)$  si  $x$  n'apparaît pas dans  $\psi$ , et  $\psi$  représente l'un des connecteurs  $\wedge$  ou  $\vee$ .

### 3 Algèbre des ensembles

Étant donné un **univers**  $\mathcal{U}$  d'objets de référence, on peut former une structure algébrique sur les sous-ensembles de  $\mathcal{U}$  avec les opérations suivantes :

**Union** (notée  $\cup$ ) :  $A \cup B$  est l'ensemble formé de la collection des éléments de  $A$  et de ceux de  $B$ . Comme la multiplicité ne compte pas, il est sans conséquence qu'un élément appartienne aux deux ensembles ou seulement à l'un d'entre eux, et alors auquel des deux. Par exemple,  $\{a, b\} \cup \{c, b\} = \{a, b, c\}$ . L'union est **commutative**, **associative**, **idempotente** et a  $\emptyset$  pour **élément neutre**. Cela autorise la notation d'union étendue comme  $\cup_{i \in [1,n]} E_i$ . L'intension d'une union d'ensembles est la disjonction des intensions de ces ensembles. Noter que  $A \cup B = B$  et que  $\cup_{i \in \emptyset} E_i = \emptyset$ .

**Intersection** (notée  $\cap$ ) :  $A \cap B$  est l'ensemble formé de la collection des éléments de  $A$  qui appartiennent aussi à  $B$ . Par exemple,  $\{a, b\} \cap \{c, b\} = \{b\}$ . L'intersection est **commutative**, **associative**, **idempotente** et a  $\mathcal{U}$  pour **élément neutre**. Cela autorise la notation d'intersection étendue comme  $\cap_{i \in [1,n]} E_i$ . L'intension d'une intersection d'ensembles est la conjonction des intensions de ces ensembles. Noter que  $A \cap B$ ssi  $A \cap B = A$  et que  $\cap_{i \in \emptyset} E_i = \mathcal{U}$ .

L'intersection et l'union sont **distributives** l'une par rapport à l'autre ; pour tout triplet d'ensembles  $A$ ,  $B$  et  $C$ ,  $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$  et  $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$ . C'est une propriété fondamentale de la théorie des ensembles.

**Complémentation** (notée  $\complement$ ) :  $\complement_A B$  est l'ensemble formé de la collection des éléments de  $A$  qui n'appartiennent pas à  $B$  : le **complémentaire** de  $B$  dans  $A$ . On note parfois  $A \setminus B$  et même  $A - B$ . Il n'est pas nécessaire que  $B$  soit inclus dans  $A$  ! Quand l'ensemble de référence est  $\mathcal{U}$  on peut se dispenser de le noter :  $\complement B = \complement_{\mathcal{U}} B$ . Noter que  $\complement \complement B = B$ . L'intension du complémentaire d'un ensemble est la conjonction de l'intension de l'ensemble de référence et de la négation de l'intension de l'ensemble. La complémentation est une sorte de négation et suit les **lois de De Morgan** :  $\complement(A \cup B) = \complement A \cap \complement B$  et  $\complement(A \cap B) = \complement A \cup \complement B$ .