Logique formelle & Programmation logique

Dr. Stéphane Lengrand,

Stephane.Lengrand@Polytechnique.edu

Cours 0.5 :

Pré-requis à ce cours

Définition par récurrence (faible) sur les entiers

$$\begin{aligned} f(0) &:= \ldots \\ f(n+1) &:= \ldots f(n) \ldots \end{aligned}$$

définit une fonction sur tous les entiers (aussi appelée suite).

Définition par récurrence (faible) sur les entiers

Exemple : La suite géométrique

$$\begin{array}{lll} f(0) & := \ldots & u_0 := 0 \\ f(n+1) & := \ldots f(n) \ldots & u_{n+1} := 2 + u_n \end{array}$$

définit une fonction sur tous les entiers (aussi appelée suite).

Définition par récurrence (faible) sur les entiers

Exemple : La suite géométrique

$$\begin{array}{llll} f(0) & := \ldots & u_0 & := 0 \\ f(n+1) & := \ldots f(n) \ldots & u_{n+1} & := 2 + u_n \end{array}$$

définit une fonction sur tous les entiers (aussi appelée suite).

Correspond à des programmes récursifs. Le même exemple en Java ou C :

```
int geo2(int n){
   if (n==0) return 0;
   return 2+geo2(n-1);
}
```

définit bien une fonction sur tous les entiers car les appels récursifs terminent.

Définition par récurrence (faible) sur les entiers

Exemple : La suite géométrique

$$\begin{array}{llll} f(0) & := \ldots & u_0 & := 0 \\ f(n+1) & := \ldots f(n) \ldots & u_{n+1} & := 2 + u_n \end{array}$$

définit une fonction sur tous les entiers (aussi appelée suite).

Correspond à des programmes récursifs. Le même exemple en Java ou C :

```
int geo2(int n){
   if (n==0) return 0;
   return 2+geo2(n-1);
}
```

définit bien une fonction sur tous les entiers car les appels récursifs terminent.

Principe de récurrence (faible) sur les entiers

Soit $\mathcal{P}$ une propriété qu'un entier peut avoir ou ne pas avoir.

Principe de récurrence (faible) sur les entiers

Soit $\mathcal{P}$ une propriété qu'un entier peut avoir ou ne pas avoir.

Exemple : $\mathcal{P}$ est "être pair"

Principe de récurrence (faible) sur les entiers

Soit $\mathcal{P}$ une propriété qu'un entier peut avoir ou ne pas avoir.

Exemple : $\mathcal{P}$ est "être pair"

"n satisfait la propriété $\mathcal{P}$" se note $\mathcal{P}(n)$

Principe de récurrence (faible) sur les entiers

Soit $\mathcal{P}$ une propriété qu'un entier peut avoir ou ne pas avoir.

Exemple : $\mathcal{P}$ est "être pair"

"n satisfait la propriété $\mathcal{P}$" se note $\mathcal{P}(n)$

Principe de récurrence "faible"

Si $\mathcal{P}(0)$

et si pour tout entier n, $\mathcal{P}(n)$ implique $\mathcal{P}(n+1)$

alors pour tout entier n, on a $\mathcal{P}(n)$.

Principe de récurrence (faible) sur les entiers

Soit $\mathcal{P}$ une propriété qu'un entier peut avoir ou ne pas avoir.

Exemple : $\mathcal{P}$ est "être pair"

"n satisfait la propriété $\mathcal{P}$" se note $\mathcal{P}(n)$

Principe de récurrence "faible"

Si $\mathcal{P}(0)$

et si pour tout entier n, $\mathcal{P}(n)$ implique $\mathcal{P}(n+1)$

alors pour tout entier n, on a $\mathcal{P}(n)$.

Exemple :

Prouvez que pour tout entier n, on a $u_n = 2 * n$

Principe de récurrence (faible) sur les entiers

Soit $\mathcal{P}$ une propriété qu'un entier peut avoir ou ne pas avoir.

Exemple : $\mathcal{P}$ est "être pair"

"n satisfait la propriété $\mathcal{P}$" se note $\mathcal{P}(n)$

Principe de récurrence "faible"

Si $\mathcal{P}(0)$

et si pour tout entier n, $\mathcal{P}(n)$ implique $\mathcal{P}(n+1)$

alors pour tout entier n, on a $\mathcal{P}(n)$.

Exemple :

Prouvez que pour tout entier n, on a $u_n = 2 * n$

Définition par récurrence (forte) sur les entiers

$$f(n) := \ldots f(i) \ldots \quad \text{où } i < n$$

définit aussi une fonction sur tous les entiers (aussi appelée suite).

Définition par récurrence (forte) sur les entiers

$$f(n) := \ldots f(i) \ldots \quad \text{où } i < n$$

Exemple : La suite

$$\begin{aligned} v_0 &:= 0 \\ v_n &:= v_{n/2} + 1 \quad n \geq 1 \end{aligned}$$

définit aussi une fonction sur tous les entiers (aussi appelée suite).

Définition par récurrence (forte) sur les entiers

$f(n) := \ldots f(i) \ldots$ où $i < n$

Exemple : La suite

$$\begin{aligned} v_0 &:= 0 \\ v_n &:= v_{n/2} + 1 \quad n \geq 1 \end{aligned}$$

définit aussi une fonction sur tous les entiers (aussi appelée suite).

Correspond à des programmes récursifs. Le même exemple en Java ou C :

```
int suite(int n){
   if (n==0) return 0;
   return suite(n/2)+1;
}
```

définit bien une fonction sur tous les entiers car les appels récursifs terminent.

Définition par récurrence (forte) sur les entiers

$f(n) := \ldots f(i) \ldots$ où $i < n$

Exemple : La suite

$$\begin{aligned} v_0 &:= 0 \\ v_n &:= v_{n/2} + 1 \quad n \geq 1 \end{aligned}$$

définit aussi une fonction sur tous les entiers (aussi appelée suite).

Correspond à des programmes récursifs. Le même exemple en Java ou C :

```
int suite(int n){
   if (n==0) return 0;
   return suite(n/2)+1;
}
```

définit bien une fonction sur tous les entiers car les appels récursifs terminent.

Principe de récurrence (fort) sur les entiers

Si pour tout entier n, $(\forall i < n, \mathcal{P}(i))$ implique $\mathcal{P}(n)$

alors pour tout entier n, on a $\mathcal{P}(n)$.

Principe de récurrence (fort) sur les entiers

Si pour tout entier n, $(\forall i < n, \mathcal{P}(i))$ implique $\mathcal{P}(n)$

alors pour tout entier n, on a $\mathcal{P}(n)$.

Exemple :

Prouvez que pour tout entier n, on a $v_n \leq n$

Principe de récurrence (fort) sur les entiers

Si pour tout entier n, $(\forall i < n, \mathcal{P}(i))$ implique $\mathcal{P}(n)$

alors pour tout entier n, on a $\mathcal{P}(n)$.

Exemple :

Prouvez que pour tout entier n, on a $v_n \leq n$

Les entiers comme structure inductive (libre)

Induction = récurrence en anglais.

Les entiers comme structure inductive (libre)

Induction = récurrence en anglais.

Les 5 axiomes de Peano

- 0 est un entier
- Si n est un entier,

  alors $S(n)$ est un entier.

- 0 n'est le successeur d'aucun entier
- Si $S(n) = S(m)$ alors $n = m$

- Principe de récurrence (faible ou fort)

. . .définissent le comportement des entiers naturels.

Les entiers comme structure inductive (libre)

Induction = récurrence en anglais.

Les 5 axiomes de Peano

- 0 est un entier
- Si n est un entier,

  alors $S(n)$ est un entier.

"Construction inductive" des entiers

- 0 n'est le successeur d'aucun entier
- Si $S(n) = S(m)$ alors $n = m$
- Principe de récurrence (faible ou fort)

. . .définissent le comportement des entiers naturels.

Les entiers comme structure inductive (libre)

Induction = récurrence en anglais.

Les 5 axiomes de Peano

- 0 est un entier
- Si n est un entier,

  alors $S(n)$ est un entier.

"Construction inductive" des entiers

- 0 n'est le successeur d'aucun entier
- Si $S(n) = S(m)$ alors $n = m$

Structure **libre**

= injectivité des constructeurs

- Principe de récurrence (faible ou fort)

. . .définissent le comportement des entiers naturels.

Les entiers comme structure inductive (libre)

Induction = récurrence en anglais.

Les 5 axiomes de Peano

- 0 est un entier
- Si n est un entier,

  alors $S(n)$ est un entier.

"Construction inductive" des entiers

- 0 n'est le successeur d'aucun entier
- Si $S(n) = S(m)$ alors $n = m$

Structure **libre**

= injectivité des constructeurs

- Principe de récurrence (faible ou fort)

. . .définissent le comportement des entiers naturels.

Autres structures inductives (libres)

Ce qu'on peut faire avec les entiers, on peut le faire avec autre chose.

Autres structures inductives (libres)

Ce qu'on peut faire avec les entiers, on peut le faire avec autre chose.

Définition des listes d'entiers :

– `nil`, la liste vide, est une liste.

– Si l est une liste et n un entier, alors $n :: l$ est une liste

(de tête n et de queue l)

Autres structures inductives (libres)

Ce qu'on peut faire avec les entiers, on peut le faire avec autre chose.

Définition des listes d'entiers :

– `nil`, la liste vide, est une liste.

– Si l est une liste et n un entier, alors $n :: l$ est une liste

(de tête n et de queue l)

Définition de la taille d'une liste, "par récurrence sur la liste" :

– la taille de `nil` est 0.

– la taille de $n :: l$ est la taille de l plus 1.

Autres structures inductives (libres)

Ce qu'on peut faire avec les entiers, on peut le faire avec autre chose.

Définition des listes d'entiers :

– `nil`, la liste vide, est une liste.
– Si l est une liste et n un entier, alors $n :: l$ est une liste
  (de tête n et de queue l)

Définition de la taille d'une liste, "par récurrence sur la liste" :

– la taille de `nil` est 0.
– la taille de $n :: l$ est la taille de l plus 1.

On pourrait poser des principes de récurrence sur les listes
mais en général, on se ramène à ceux sur les entiers via la notion de taille.

Autres structures inductives (libres)

Ce qu'on peut faire avec les entiers, on peut le faire avec autre chose.

Définition des listes d'entiers :

– `nil`, la liste vide, est une liste.
– Si l est une liste et n un entier, alors $n :: l$ est une liste
  (de tête n et de queue l)

Définition de la taille d'une liste, "par récurrence sur la liste" :

– la taille de `nil` est 0.
– la taille de $n :: l$ est la taille de l plus 1.

On pourrait poser des principes de récurrence sur les listes
mais en général, on se ramène à ceux sur les entiers via la notion de taille.

Autres structures inductives (libres)

Définition des arbres étiquetés :

– une feuille, étiquetée par un symbole s, est un arbre

– Si A_1,..., A_n sont des arbres, et f est un symbole,

  alors $f(A_1, \ldots, A_n)$ est un arbre.

Voir dessin au tableau.

Autres structures inductives (libres)

Définition des arbres étiquetés :

– une feuille, étiquetée par un symbole s, est un arbre

– Si A_1,..., A_n sont des arbres, et f est un symbole,
  alors $f(A_1, \ldots, A_n)$ est un arbre.

Voir dessin au tableau.

(Exemple de) définition de la taille d'un arbre, "par récurrence sur l'arbre" :

– la taille d'une feuille est 1.

– la taille de $f(A_1, \ldots, A_n)$ est la somme des tailles de A_1,..., A_n,
  plus 1.

Autres structures inductives (libres)

Définition des arbres étiquetés :

– une feuille, étiquetée par un symbole s, est un arbre

– Si A_1,..., A_n sont des arbres, et f est un symbole,
  alors $f(A_1, \ldots, A_n)$ est un arbre.

Voir dessin au tableau.

(Exemple de) définition de la taille d'un arbre, "par récurrence sur l'arbre" :

– la taille d'une feuille est 1.

– la taille de $f(A_1, \ldots, A_n)$ est la somme des tailles de A_1,..., A_n,
  plus 1.

On pourrait poser des principes de récurrence sur les structures inductives mais en général, on se ramène à ceux sur les entiers.

Autres structures inductives (libres)

Définition des arbres étiquetés :

– une feuille, étiquetée par un symbole s, est un arbre

– Si A_1,..., A_n sont des arbres, et f est un symbole,
  alors $f(A_1, \ldots, A_n)$ est un arbre.

Voir dessin au tableau.

(Exemple de) définition de la taille d'un arbre, "par récurrence sur l'arbre" :

– la taille d'une feuille est 1.

– la taille de $f(A_1, \ldots, A_n)$ est la somme des tailles de A_1,..., A_n,
  plus 1.

On pourrait poser des principes de récurrence sur les structures inductives mais en général, on se ramène à ceux sur les entiers.

Cours 1 :

Syntaxe, Sémantique, Logique propositionnelle

Pas de pensée sans langage

Prenons les mathématiques comme objet d'étude

(faisons des meta-mathématiques !), et analysons leur structure !

Pas de pensée sans langage

Prenons les mathématiques comme objet d'étude

(faisons des meta-mathématiques !), et analysons leur structure !

Une **proposition** exprime, dans un langage syntaxique, quelque chose

(qui a un sens).

Pas de pensée sans langage

Prenons les mathématiques comme objet d'étude

(faisons des meta-mathématiques !), et analysons leur structure !

Une **proposition** exprime, dans un langage syntaxique, quelque chose

(qui a un sens).

structure à base de symboles qui permet d'exprimer qq chose = syntaxe,

signification de cette expression = sémantique

Pas de pensée sans langage

Prenons les mathématiques comme objet d'étude

(faisons des meta-mathématiques !), et analysons leur structure !

Une **proposition** exprime, dans un langage syntaxique, quelque chose

(qui a un sens).

structure à base de symboles qui permet d'exprimer qq chose = syntaxe,

signification de cette expression = sémantique

signifiant = syntaxe, signifié = sémantique

Pas de pensée sans langage

Prenons les mathématiques comme objet d'étude

(faisons des meta-mathématiques !), et analysons leur structure !

Une **proposition** exprime, dans un langage syntaxique, quelque chose

(qui a un sens).

structure à base de symboles qui permet d'exprimer qq chose = syntaxe,

signification de cette expression = sémantique

signifiant = syntaxe, signifié = sémantique

Niveau meta (transparent subtile, mais pas vital)

Pendant longtemps : monde réel fournit un cadre pour la sémantique,

on vérifiait qu'une proposition était vraie ou fausse par confrontation avec le réel.

Niveau meta (transparent subtile, mais pas vital)

Pendant longtemps : monde réel fournit un cadre pour la sémantique,

on vérifiait qu'une proposition était vraie ou fausse par confrontation avec le réel.

Maintenant : objets mathématiques trop abstraits

(voyez-vous des nombres complexes dans la rue ?)

$\Longrightarrow$ sémantique donnée par le niveau meta-mathématique

Niveau meta (transparent subtile, mais pas vital)

Pendant longtemps : monde réel fournit un cadre pour la sémantique,

on vérifiait qu'une proposition était vraie ou fausse par confrontation avec le réel.

Maintenant : objets mathématiques trop abstraits

(voyez-vous des nombres complexes dans la rue ?)

$\Longrightarrow$ sémantique donnée par le niveau meta-mathématique

Se placer / raisonner dans une logique X pour étudier la logique Y.

Niveau meta (transparent subtile, mais pas vital)

Pendant longtemps : monde réel fournit un cadre pour la sémantique,

on vérifiait qu'une proposition était vraie ou fausse par confrontation avec le réel.

Maintenant : objets mathématiques trop abstraits

(voyez-vous des nombres complexes dans la rue ?)

$\Longrightarrow$ sémantique donnée par le niveau meta-mathématique

Se placer / raisonner dans une logique X pour étudier la logique Y.

Exemples :

X = l'arithmétique de Péano $\Longrightarrow$ sémantique à base d'entiers

X = théorie des ensembles $\Longrightarrow$ sémantique à base d'ensembles

Niveau meta (transparent subtile, mais pas vital)

Pendant longtemps : monde réel fournit un cadre pour la sémantique, on vérifiait qu'une proposition était vraie ou fausse par confrontation avec le réel.

Maintenant : objets mathématiques trop abstraits

(voyez-vous des nombres complexes dans la rue ?)

$\Longrightarrow$ sémantique donnée par le niveau meta-mathématique

Se placer / raisonner dans une logique X pour étudier la logique Y.

Exemples :

X = l'arithmétique de Péano $\Longrightarrow$ sémantique à base d'entiers

X = théorie des ensembles $\Longrightarrow$ sémantique à base d'ensembles

Remarque : le niveau meta-mathématique étant syntaxique, finalement il n'y a jamais que de la syntaxe (sémantique = syntaxe du niveau meta)

Niveau meta (transparent subtile, mais pas vital)

Pendant longtemps : monde réel fournit un cadre pour la sémantique, on vérifiait qu'une proposition était vraie ou fausse par confrontation avec le réel.

Maintenant : objets mathématiques trop abstraits

(voyez-vous des nombres complexes dans la rue ?)

$\Longrightarrow$ sémantique donnée par le niveau meta-mathématique

Se placer / raisonner dans une logique X pour étudier la logique Y.

Exemples :

X = l'arithmétique de Péano $\Longrightarrow$ sémantique à base d'entiers

X = théorie des ensembles $\Longrightarrow$ sémantique à base d'ensembles

Remarque : le niveau meta-mathématique étant syntaxique, finalement il n'y a jamais que de la syntaxe (sémantique = syntaxe du niveau meta)

Clivage syntaxe/sémantique à deux niveaux

Une proposition parle d'objets (d'études).

Clivage syntaxe/sémantique à deux niveaux

Une proposition parle d'objets (d'études).

Niveau objet : les structures syntaxiques 4 et IV désignent le même objet sémantique.

Clivage syntaxe/sémantique à deux niveaux

Une proposition parle d'objets (d'études).

Niveau objet : les structures syntaxiques 4 et IV désignent le même objet sémantique.

Niveau proposition : les structures syntaxiques $(x \in y) \wedge (x \in z)$ et $(x \in z) \wedge (x \in y)$ ont la même sémantique.

Clivage syntaxe/sémantique à deux niveaux

Une proposition parle d'objets (d'études).

Niveau objet : les structures syntaxiques 4 et IV désignent le même objet sémantique.

Niveau proposition : les structures syntaxiques $(x \in y) \wedge (x \in z)$ et $(x \in z) \wedge (x \in y)$ ont la même sémantique.

Syntaxe de la logique propositionnelle

En logique propositionnelle : pas d'objets !

Syntaxe de la logique propositionnelle

En logique propositionnelle : pas d'objets !

– des variables propositionnelles $p, q, r, \ldots$ désignent des propositions quelconques

Syntaxe de la logique propositionnelle

En logique propositionnelle : pas d'objets !

– des variables propositionnelles $p, q, r, \ldots$ désignent des propositions quelconques

– des connecteurs $\wedge$ (et) $\vee$ (ou) $\Rightarrow$ (implique), $\neg$ (non-),. . .construisent des propositions complexes à partir de propositions simples, ou sont des constantes logiques $\top$ (vrai) $\bot$ (faux),. . .

Syntaxe de la logique propositionnelle

En logique propositionnelle : pas d'objets !

– des variables propositionnelles $p, q, r, \ldots$ désignent des propositions quelconques

– des connecteurs $\wedge$ (et) $\vee$ (ou) $\Rightarrow$ (implique), $\neg$ (non-),. . .construisent des propositions complexes à partir de propositions simples, ou sont des constantes logiques $\top$ (vrai) $\bot$ (faux),. . .

Une manière rapide d'écrire les règles de construction des propositions :

$$A, B, C, \ldots ::= p \mid (A \wedge B) \mid (A \vee B) \mid (A \Rightarrow B) \mid (\neg A) \mid \top \mid \bot$$

Syntaxe de la logique propositionnelle

En logique propositionnelle : pas d'objets !

– des variables propositionnelles $p, q, r, \ldots$ désignent des propositions quelconques

– des connecteurs $\wedge$ (et) $\vee$ (ou) $\Rightarrow$ (implique), $\neg$ (non-),. . .construisent des propositions complexes à partir de propositions simples, ou sont des constantes logiques $\top$ (vrai) $\bot$ (faux),. . .

Une manière rapide d'écrire les règles de construction des propositions :

$$A, B, C, \ldots ::= p \mid (A \wedge B) \mid (A \vee B) \mid (A \Rightarrow B) \mid (\neg A) \mid \top \mid \bot$$

On appelle ça une définition inductive

Syntaxe de la logique propositionnelle

En logique propositionnelle : pas d'objets !

– des variables propositionnelles $p, q, r, \ldots$ désignent des propositions quelconques

– des connecteurs $\wedge$ (et) $\vee$ (ou) $\Rightarrow$ (implique), $\neg$ (non-),. . .construisent des propositions complexes à partir de propositions simples, ou sont des constantes logiques $\top$ (vrai) $\bot$ (faux),. . .

Une manière rapide d'écrire les règles de construction des propositions :

$$A, B, C, \ldots ::= p \mid (A \wedge B) \mid (A \vee B) \mid (A \Rightarrow B) \mid (\neg A) \mid \top \mid \bot$$

On appelle ça une définition inductive

Propositions = chaînes de symbôles bien-parenthésées, ou arbres ?

Syntaxe de la logique propositionnelle

En logique propositionnelle : pas d'objets !

– des variables propositionnelles $p, q, r, \ldots$ désignent des propositions quelconques

– des connecteurs $\wedge$ (et) $\vee$ (ou) $\Rightarrow$ (implique), $\neg$ (non-),. . .construisent des propositions complexes à partir de propositions simples, ou sont des constantes logiques $\top$ (vrai) $\bot$ (faux),. . .

Une manière rapide d'écrire les règles de construction des propositions :

$$A, B, C, \ldots ::= p \mid (A \wedge B) \mid (A \vee B) \mid (A \Rightarrow B) \mid (\neg A) \mid \top \mid \bot$$

On appelle ça une définition inductive

Propositions = chaînes de symbôles bien-parenthésées, ou arbres ?

les 2 visions sont équivalentes

Syntaxe de la logique propositionnelle

En logique propositionnelle : pas d'objets !

– des variables propositionnelles $p, q, r, \ldots$ désignent des propositions quelconques

– des connecteurs $\wedge$ (et) $\vee$ (ou) $\Rightarrow$ (implique), $\neg$ (non-),. . .construisent des propositions complexes à partir de propositions simples, ou sont des constantes logiques $\top$ (vrai) $\bot$ (faux),. . .

Une manière rapide d'écrire les règles de construction des propositions :

$$A, B, C, \ldots ::= p \mid (A \wedge B) \mid (A \vee B) \mid (A \Rightarrow B) \mid (\neg A) \mid \top \mid \bot$$

On appelle ça une définition inductive

Propositions = chaînes de symbôles bien-parenthésées, ou arbres ?

les 2 visions sont équivalentes

Sémantique de la logique propositionnelle

Pour donner une sémantique aux propositions, il faut

Sémantique de la logique propositionnelle

Pour donner une sémantique aux propositions, il faut

– un ensemble $\mathcal{B}$ dans lequel on va interpréter les propositions.

Sémantique de la logique propositionnelle

Pour donner une sémantique aux propositions, il faut

– un ensemble $\mathcal{B}$ dans lequel on va interpréter les propositions.

– une sémantique pour chaque connecteur $\star$,

c'est-à-dire une fonction $f_\star$ de $\mathcal{B}^n$ dans $\mathcal{B}$

($n = 2$ pour les connecteurs binaires, $n = 1$ pour les connecteurs unaires, $n = 0$ pour les constantes, ...)

Sémantique de la logique propositionnelle

Pour donner une sémantique aux propositions, il faut

– un ensemble $\mathcal{B}$ dans lequel on va interpréter les propositions.

– une sémantique pour chaque connecteur $\star$,

c'est-à-dire une fonction $f_\star$ de $\mathcal{B}^n$ dans $\mathcal{B}$

($n = 2$ pour les connecteurs binaires, $n = 1$ pour les connecteurs unaires, $n = 0$ pour les constantes, ...)

Pour que la sémantique de $\wedge$, $\vee$, ... corresponde à notre intuition, il faut que $\mathcal{B}$ soit une algèbre de Boole

Sémantique de la logique propositionnelle

Pour donner une sémantique aux propositions, il faut

– un ensemble $\mathcal{B}$ dans lequel on va interpréter les propositions.

– une sémantique pour chaque connecteur $\star$,

c'est-à-dire une fonction $f_\star$ de $\mathcal{B}^n$ dans $\mathcal{B}$

($n = 2$ pour les connecteurs binaires, $n = 1$ pour les connecteurs unaires, $n = 0$ pour les constantes, ...)

Pour que la sémantique de $\wedge, \vee, \ldots$ corresponde à notre intuition, il faut que $\mathcal{B}$ soit une algèbre de Boole

Exemple d'algèbre de Boole :

$\mathcal{B}$ est l'ensemble des parties d'un ensemble X (quelconque), avec

$$f_\wedge(x, y) = x \cap y \quad f_\top = X \quad f_\neg(x) = \overline{x}$$

$$f_\vee(x, y) = x \cup y \quad f_\bot = \emptyset$$

Sémantique de la logique propositionnelle

Pour donner une sémantique aux propositions, il faut

– un ensemble $\mathcal{B}$ dans lequel on va interpréter les propositions.

– une sémantique pour chaque connecteur $\star$,

c'est-à-dire une fonction $f_\star$ de $\mathcal{B}^n$ dans $\mathcal{B}$

($n = 2$ pour les connecteurs binaires, $n = 1$ pour les connecteurs unaires, $n = 0$ pour les constantes, ...)

Pour que la sémantique de $\wedge, \vee, \ldots$ corresponde à notre intuition, il faut que $\mathcal{B}$ soit une algèbre de Boole

Exemple d'algèbre de Boole :

$\mathcal{B}$ est l'ensemble des parties d'un ensemble X (quelconque), avec

$$f_\wedge(x, y) = x \cap y \quad f_\top = X \quad f_\neg(x) = \overline{x}$$

$$f_\vee(x, y) = x \cup y \quad f_\bot = \emptyset$$

Autre exemple d'algèbre de Boole : les booléens

L'ensemble des booléens $\mathcal{B} = \{\mathsf{T}, \mathsf{F}\}$ à 2 éléments.

x	y	$f_\wedge(x,y)$
T	T	T
T	F	F
F	T	F
F	F	F

x	y	$f_\vee(x,y)$
T	T	T
T	F	T
F	T	T
F	F	F

x	y	$f_\Rightarrow(x,y)$
T	T	T
T	F	F
F	T	T
F	F	T

x	$f_\neg(x)$
T	F
F	T

$f_\top()$
T

$f_\bot()$
F

Sémantique de la logique propositionnelle

Une fois que l'on s'est donné ces fonctions, on peut alors calculer l'interprétation dans $\mathcal{B}$ des propositions

Sémantique de la logique propositionnelle

Une fois que l'on s'est donné ces fonctions, on peut alors calculer l'interprétation dans $\mathcal{B}$ des propositions

Un paramètre : l'interprétation $\mathcal{I}$ des variables propositionelles $p, q, r, \ldots$, dite valuation.

Sémantique de la logique propositionnelle

Une fois que l'on s'est donné ces fonctions, on peut alors calculer l'interprétation dans $\mathcal{B}$ des propositions

Un paramètre : l'interprétation $\mathcal{I}$ des variables propositionelles $p, q, r, \ldots$, dite valuation. Exemple avec les booléens : $\mathcal{I}(p) =$ T ou $\mathcal{I}(p) =$ F.

Sémantique de la logique propositionnelle

Une fois que l'on s'est donné ces fonctions, on peut alors calculer l'interprétation dans $\mathcal{B}$ des propositions

Un paramètre : l'interprétation $\mathcal{I}$ des variables propositionelles $p, q, r, \ldots$, dite valuation. Exemple avec les booléens : $\mathcal{I}(p) =$ T ou $\mathcal{I}(p) =$ F.

L'interprétation $[A]_{\mathcal{I}}$ d'une d'une proposition A selon la valuation $\mathcal{I}$ est définie par récurrence sur A :

$$\begin{aligned} [p]_{\mathcal{I}} &:= \mathcal{I}(p) \\ [\star(A_1, \ldots, A_n)]_{\mathcal{I}} &:= f_\star([A_1]_{\mathcal{I}}, \ldots, [A_n]_{\mathcal{I}}) \end{aligned}$$

Sémantique de la logique propositionnelle

Une fois que l'on s'est donné ces fonctions, on peut alors calculer l'interprétation dans $\mathcal{B}$ des propositions

Un paramètre : l'interprétation $\mathcal{I}$ des variables propositionelles $p, q, r, \ldots$, dite valuation. Exemple avec les booléens : $\mathcal{I}(p) = \mathsf{T}$ ou $\mathcal{I}(p) = \mathsf{F}$.

L'interprétation $[A]_{\mathcal{I}}$ d'une d'une proposition A selon la valuation $\mathcal{I}$ est définie par récurrence sur A :

$$\begin{aligned} [p]_{\mathcal{I}} &:= \mathcal{I}(p) \\ [\star(A_1, \ldots, A_n)]_{\mathcal{I}} &:= f_\star([A_1]_{\mathcal{I}}, \ldots, [A_n]_{\mathcal{I}}) \end{aligned}$$

Exemple

$$\begin{aligned} [A \wedge B]_{\mathcal{I}} &:= f_\wedge([A]_{\mathcal{I}}, [B]_{\mathcal{I}}) \\ [\neg A]_{\mathcal{I}} &:= f_\neg([A]_{\mathcal{I}}) \end{aligned}$$

Sémantique de la logique propositionnelle

Une fois que l'on s'est donné ces fonctions, on peut alors calculer l'interprétation dans $\mathcal{B}$ des propositions

Un paramètre : l'interprétation $\mathcal{I}$ des variables propositionelles $p, q, r, \ldots$, dite valuation. Exemple avec les booléens : $\mathcal{I}(p) = \mathsf{T}$ ou $\mathcal{I}(p) = \mathsf{F}$.

L'interprétation $[A]_{\mathcal{I}}$ d'une d'une proposition A selon la valuation $\mathcal{I}$ est définie par récurrence sur A :

$$\begin{aligned} [p]_{\mathcal{I}} &:= \mathcal{I}(p) \\ [\star(A_1, \ldots, A_n)]_{\mathcal{I}} &:= f_\star([A_1]_{\mathcal{I}}, \ldots, [A_n]_{\mathcal{I}}) \end{aligned}$$

Exemple

$$\begin{aligned} [A \wedge B]_{\mathcal{I}} &:= f_\wedge([A]_{\mathcal{I}}, [B]_{\mathcal{I}}) \\ [\neg A]_{\mathcal{I}} &:= f_\neg([A]_{\mathcal{I}}) \end{aligned}$$

Satisfiable, Valide

A partir de maintenant, $\mathcal{B} = \{\mathsf{T}, \mathsf{F}\}$

Satisfiable, Valide

A partir de maintenant, $\mathcal{B} = \{\mathsf{T}, \mathsf{F}\}$

Définitions :

– $\mathcal{I}$ est un modèle de A si $[A]_{\mathcal{I}} = \mathsf{T}$

– A est satisfiable s'il y a une valuation qui est un modèle de A

– A est valide si toute valuation est un modèle de A

Satisfiable, Valide

A partir de maintenant, $\mathcal{B} = \{\mathsf{T}, \mathsf{F}\}$

Définitions :

– $\mathcal{I}$ est un modèle de A si $[A]_{\mathcal{I}} = \mathsf{T}$

– A est satisfiable s'il y a une valuation qui est un modèle de A

– A est valide si toute valuation est un modèle de A

Théorème : A est satisfiable (resp. valide) si et seulement si $\neg A$ n'est pas valide (resp. satisfiable)

Satisfiable, Valide

A partir de maintenant, $\mathcal{B} = \{\mathsf{T}, \mathsf{F}\}$

Définitions :

– $\mathcal{I}$ est un modèle de A si $[A]_{\mathcal{I}} = \mathsf{T}$

– A est satisfiable s'il y a une valuation qui est un modèle de A

– A est valide si toute valuation est un modèle de A

Théorème : A est satisfiable (resp. valide) si et seulement si $\neg A$ n'est pas valide (resp. satisfiable)

Pour vérifier qu'une proposition est valide ou satisfiable, on regarde toutes les valuations $\mathcal{I}$ possibles sous la forme d'une table de vérité

Satisfiable, Valide

A partir de maintenant, $\mathcal{B} = \{\mathsf{T}, \mathsf{F}\}$

Définitions :

– $\mathcal{I}$ est un modèle de A si $[A]_{\mathcal{I}} = \mathsf{T}$

– A est satisfiable s'il y a une valuation qui est un modèle de A

– A est valide si toute valuation est un modèle de A

Théorème : A est satisfiable (resp. valide) si et seulement si $\neg A$ n'est pas valide (resp. satisfiable)

Pour vérifier qu'une proposition est valide ou satisfiable, on regarde toutes les valuations $\mathcal{I}$ possibles sous la forme d'une table de vérité

Si A possède n variables propositionnelles, on a 2^n cas à tester !

Exemple en exercice avec $((p \Rightarrow q) \Rightarrow p) \Rightarrow p$

Satisfiable, Valide

A partir de maintenant, $\mathcal{B} = \{\mathsf{T}, \mathsf{F}\}$

Définitions :

– $\mathcal{I}$ est un modèle de A si $[A]_{\mathcal{I}} = \mathsf{T}$

– A est satisfiable s'il y a une valuation qui est un modèle de A

– A est valide si toute valuation est un modèle de A

Théorème : A est satisfiable (resp. valide) si et seulement si $\neg A$ n'est pas valide (resp. satisfiable)

Pour vérifier qu'une proposition est valide ou satisfiable, on regarde toutes les valuations $\mathcal{I}$ possibles sous la forme d'une table de vérité

Si A possède n variables propositionnelles, on a 2^n cas à tester !

Exemple en exercice avec $((p \Rightarrow q) \Rightarrow p) \Rightarrow p$

Conséquence sémantique (parfois dite logique)

Définition

– B est une conséquence sémantique de A, noté $A \models B$

si tout modèle de A est aussi un modèle de B

– A et B sont sémantiquement équivalents, noté $A \equiv B$,

si $A \models B$ et $B \models A$

Conséquence sémantique (parfois dite logique)

Définition

– B est une conséquence sémantique de A, noté $A \models B$

si tout modèle de A est aussi un modèle de B

– A et B sont sémantiquement équivalents, noté $A \equiv B$,

si $A \models B$ et $B \models A$

voir exercice sur les lois de De Morgan

Conséquence sémantique (parfois dite logique)

Définition

– B est une conséquence sémantique de A, noté $A \models B$

si tout modèle de A est aussi un modèle de B

– A et B sont sémantiquement équivalents, noté $A \equiv B$,

si $A \models B$ et $B \models A$

voir exercice sur les lois de De Morgan

Notez que

– $\top \models A$, aussi noté $\models A$, si et seulement si A est valide.

Conséquence sémantique (parfois dite logique)

Définition

– B est une conséquence sémantique de A, noté $A \models B$

si tout modèle de A est aussi un modèle de B

– A et B sont sémantiquement équivalents, noté $A \equiv B$,

si $A \models B$ et $B \models A$

voir exercice sur les lois de De Morgan

Notez que

– $\top \models A$, aussi noté $\models A$, si et seulement si A est valide.

– $A \models B$ si et seulement si $A \Rightarrow B$ est valide (voir TD).

Conséquence sémantique (parfois dite logique)

Définition

– B est une conséquence sémantique de A, noté $A \models B$

si tout modèle de A est aussi un modèle de B

– A et B sont sémantiquement équivalents, noté $A \equiv B$,

si $A \models B$ et $B \models A$

voir exercice sur les lois de De Morgan

Notez que

– $\top \models A$, aussi noté $\models A$, si et seulement si A est valide.

– $A \models B$ si et seulement si $A \Rightarrow B$ est valide (voir TD).

– si $A \models B$, alors

  – A est valide implique B est valide

  – A est satisfiable implique B est satisfiable

mais l'inverse n'est pas vrai !

Conséquence sémantique (parfois dite logique)

Définition

– B est une conséquence sémantique de A, noté $A \models B$

si tout modèle de A est aussi un modèle de B

– A et B sont sémantiquement équivalents, noté $A \equiv B$,

si $A \models B$ et $B \models A$

voir exercice sur les lois de De Morgan

Notez que

– $\top \models A$, aussi noté $\models A$, si et seulement si A est valide.

– $A \models B$ si et seulement si $A \Rightarrow B$ est valide (voir TD).

– si $A \models B$, alors

  – A est valide implique B est valide

  – A est satisfiable implique B est satisfiable

mais l'inverse n'est pas vrai !

Conclusions

Toutes ces propriétés des propositions sont des

constatations sémantiques

Conclusions

Toutes ces propriétés des propositions sont des

constatations sémantiques

Cours suivant : on verra comment on peut prouver la conséquence ou la validité par une démonstration, c'est-à-dire par un

raisonnement syntaxique

Conclusions

Toutes ces propriétés des propositions sont des

constatations sémantiques

Cours suivant : on verra comment on peut prouver la conséquence ou la validité par une démonstration, c'est-à-dire par un

raisonnement syntaxique

Questions?