

# Chapitre 2

## Définition de la logique propositionnelle

## Contenu de ce chapitre

- ▶ Syntaxe de la logique propositionnelle
  - ▶ Définitions inductives
  - ▶ Preuves par induction

## Contenu de ce chapitre

- ▶ Syntaxe de la logique propositionnelle
  - ▶ Définitions inductives
  - ▶ Preuves par induction
- ▶ Sémantique de la logique propositionnelle
  - ▶ Définition de fonctions par récurrence

## Contenu de ce chapitre

- ▶ Syntaxe de la logique propositionnelle
  - ▶ Définitions inductives
  - ▶ Preuves par induction
- ▶ Sémantique de la logique propositionnelle
  - ▶ Définition de fonctions par récurrence
- ▶ Algorithmique de la logique propositionnelle (première approche)

## Vers une définition de la syntaxe

### Première Idée

On définit les formules selon les différents cas de leur forme.

## Vers une définition de la syntaxe

### Première Idée

On définit les formules selon les différents cas de leur forme.

### Une formule propositionnelle

- ▶ peut être une variable propositionnelle (comme  $x$ )
- ▶  $\neg p$ , où  $p$  est une formule
- ▶  $(p \wedge q)$ , où  $p$  et  $q$  sont des formules
- ▶  $(p \vee q)$ , où  $p$  et  $q$  sont des formules

## Vers une définition de la syntaxe

### Première Idée

On définit les formules selon les différents cas de leur forme.

### Une formule propositionnelle

- ▶ peut être une variable propositionnelle (comme  $x$ )
- ▶  $\neg p$ , où  $p$  est une formule
- ▶  $(p \wedge q)$ , où  $p$  et  $q$  sont des formules
- ▶  $(p \vee q)$ , où  $p$  et  $q$  sont des formules

### Une précision importante :

Seulement les chaînes de caractères ainsi formées sont des formules.

## Exemples de formules propositionnelles

Sont des formules :

- ▶  $x$  (car variable propositionnelle)



## Exemples de formules propositionnelles

Sont des formules :

- ▶  $x$  (car variable propositionnelle)
- ▶  $\neg x$  (car négation d'une formule)

## Exemples de formules propositionnelles

Sont des formules :

- ▶  $x$  (car variable propositionnelle)
- ▶  $\neg x$  (car négation d'une formule)
- ▶  $y$  (car variable propositionnelle)

## Exemples de formules propositionnelles

Sont des formules :

- ▶  $x$  (car variable propositionnelle)
- ▶  $\neg x$  (car négation d'une formule)
- ▶  $y$  (car variable propositionnelle)
- ▶  $(\neg x \wedge y)$  (car conjonction de deux formules)

## Exemples de formules propositionnelles

Sont des formules :

- ▶  $x$  (car variable propositionnelle)
- ▶  $\neg x$  (car négation d'une formule)
- ▶  $y$  (car variable propositionnelle)
- ▶  $(\neg x \wedge y)$  (car conjonction de deux formules)

Ne sont **pas** des formules (dans le sens stricte) :

## Exemples de formules propositionnelles

Sont des formules :

- ▶  $x$  (car variable propositionnelle)
- ▶  $\neg x$  (car négation d'une formule)
- ▶  $y$  (car variable propositionnelle)
- ▶  $(\neg x \wedge y)$  (car conjonction de deux formules)

Ne sont **pas** des formules (dans le sens stricte) :

- ▶  $((x$

## Exemples de formules propositionnelles

Sont des formules :

- ▶  $x$  (car variable propositionnelle)
- ▶  $\neg x$  (car négation d'une formule)
- ▶  $y$  (car variable propositionnelle)
- ▶  $(\neg x \wedge y)$  (car conjonction de deux formules)

Ne sont **pas** des formules (dans le sens stricte) :

- ▶  $((x$
- ▶  $x \vee y$

## Les variables propositionnelles

On choisit d'abord un ensemble de variables propositionnelles :

$$V := \{x, x_1, x_2, x_3, \dots, y, y_1, y_2, \dots, z, z_1, z_2, z_3, \dots\}$$

Nous admettons également les décorations habituelles des variables propositionnelles comme par exemple  $x'$ ,  $y''$ .

## Tentative de définition

L'ensemble  $Form$  des formules propositionnelles est défini comme l'ensemble de chaînes de caractères tel que :

1.  $V \subseteq Form$
2. Si  $p \in Form$  alors  $\neg p \in Form$
3. Si  $p, q \in Form$  alors  $(p \wedge q) \in Form$
4. Si  $p, q \in Form$  alors  $(p \vee q) \in Form$

Nous appelons ces quatre conditions (1) - (4) les **propriétés de clôture des formules propositionnelles**.



## Le problème de la tentative de définition

Il y a plusieurs ensembles *Form* qui satisfont la description :

- L'ensemble  $E_1$  des formules propositionnelles dans le sens de la description informelle au-dessus.

## Le problème de la tentative de définition

Il y a plusieurs ensembles *Form* qui satisfont la description :

- ▶ L'ensemble  $E_1$  des formules propositionnelles dans le sens de la description informelle au-dessus.
- ▶ L'ensemble  $E_2$  de toutes les chaînes de caractères qui ont le même nombre de « ( » que de « ) ».  
Mais cet ensemble contient aussi la chaîne « ( ) ».

## Le problème de la tentative de définition

Il y a plusieurs ensembles *Form* qui satisfont la description :

- ▶ L'ensemble  $E_1$  des formules propositionnelles dans le sens de la description informelle au-dessus.
- ▶ L'ensemble  $E_2$  de toutes les chaînes de caractères qui ont le même nombre de « ( » que de « ) ».  
Mais cet ensemble contient aussi la chaîne « ( ) ( ) ».
- ▶ L'ensemble  $E_3$  de toutes les chaînes de caractères.  
Cet ensemble contient aussi la chaîne « ( ( ».

## Le problème de la tentative de définition

Il y a plusieurs ensembles *Form* qui satisfont la description :

- ▶ L'ensemble  $E_1$  des formules propositionnelles dans le sens de la description informelle au-dessus.
- ▶ L'ensemble  $E_2$  de toutes les chaînes de caractères qui ont le même nombre de « ( » que de « ) ».  
Mais cet ensemble contient aussi la chaîne « ( ) ( ) ».
- ▶ L'ensemble  $E_3$  de toutes les chaînes de caractères.  
Cet ensemble contient aussi la chaîne « ( ( ».

Nous avons  $E_1 \subseteq E_2 \subseteq E_3$ .

## La racine du problème

Notre tentative de définition prend en compte

- ▶ le cas de base (les variables propositionnelles)
- ▶ et les constructions autorisées ( $\neg$ ,  $\wedge$ ,  $\vee$ )

mais ne prend absolument pas en compte la restriction qu'une chaîne qui ne peut pas être construite selon les règles précédentes n'est pas une formule.

## La bonne définition

### Definition

L'ensemble  $Form$  des formules propositionnelles est **le plus petit ensemble** de chaînes de caractères tel que :

1.  $V \subseteq Form$
2. Si  $p \in Form$  alors  $\neg p \in Form$
3. Si  $p, q \in Form$  alors  $(p \wedge q) \in Form$
4. Si  $p, q \in Form$  alors  $(p \vee q) \in Form$

## Définitions inductives

### Le principe

- ▶ On a certaines **propriétés de clôture** (ici : les quatre propriétés de clôture des formules propositionnelles)
- ▶ On définit un ensemble (ici : *Form*) comme étant **le plus petit ensemble** ayant ces propriétés de clôture.

## Définitions inductives

### Le principe

- ▶ On a certaines **propriétés de clôture** (ici : les quatre propriétés de clôture des formules propositionnelles)
- ▶ On définit un ensemble (ici :  $Form$ ) comme étant **le plus petit ensemble** ayant ces propriétés de clôture.

### Qu'est-ce que c'est le plus petit ensemble ?

Si un ensemble  $X$  satisfait toutes les propriétés de clôture des formules propositionnelles, alors  $Form \subseteq X$ .



## Définitions inductives

### Le principe

- ▶ On a certaines **propriétés de clôture** (ici : les quatre propriétés de clôture des formules propositionnelles)
- ▶ On définit un ensemble (ici :  $Form$ ) comme étant **le plus petit ensemble** ayant ces propriétés de clôture.

### Qu'est-ce que c'est le plus petit ensemble ?

Si un ensemble  $X$  satisfait toutes les propriétés de clôture des formules propositionnelles, alors  $Form \subseteq X$ .

### As-t-on le droit d'écrire une telle définition ?

1. Est-ce qu'il existe effectivement (au moins) un tel plus petit ensemble ?
2. Cet ensemble, est-il unique ?

## Existence d'un plus petit ensemble

- ▶ A priori, il n'est pas évident qu'il existe un (**unique**) plus petit ensemble avec une certaine propriété donnée.

## Existence d'un plus petit ensemble

- ▶ A priori, il n'est pas évident qu'il existe un (**unique**) plus petit ensemble avec une certaine propriété donnée.
- ▶ Exemple : **le plus petit** ensemble d'entiers qui contient au moins deux éléments ?

## Existence d'un plus petit ensemble

- ▶ A priori, il n'est pas évident qu'il existe un (**unique**) plus petit ensemble avec une certaine propriété donnée.
- ▶ Exemple : **le plus petit** ensemble d'entiers qui contient au moins deux éléments ?
- ▶ Ensembles  $\{0, 1\}$  et  $\{2, 3\}$

## Existence d'un plus petit ensemble

- ▶ A priori, il n'est pas évident qu'il existe un (**unique**) plus petit ensemble avec une certaine propriété donnée.
- ▶ Exemple : **le plus petit** ensemble d'entiers qui contient au moins deux éléments ?
- ▶ Ensembles  $\{0, 1\}$  et  $\{2, 3\}$
- ▶ Dans notre cas, il y a un plus petit ensemble (mais on ne va pas le montrer).

## Existence d'un plus petit ensemble

- ▶ A priori, il n'est pas évident qu'il existe un (**unique**) plus petit ensemble avec une certaine propriété donnée.
- ▶ Exemple : **le plus petit** ensemble d'entiers qui contient au moins deux éléments ?
- ▶ Ensembles  $\{0, 1\}$  et  $\{2, 3\}$
- ▶ Dans notre cas, il y a un plus petit ensemble (mais on ne va pas le montrer).
- ▶ En bref : c'est dû au fait qu'il s'agit ici de propriétés de clôture.

## Définitions inductives

- Notre définition de la syntaxe de la logique propositionnelle est une **définition inductive**.

## Définitions inductives

- ▶ Notre définition de la syntaxe de la logique propositionnelle est une **définition inductive**.
- ▶ On verra d'autres exemples de définitions inductives.

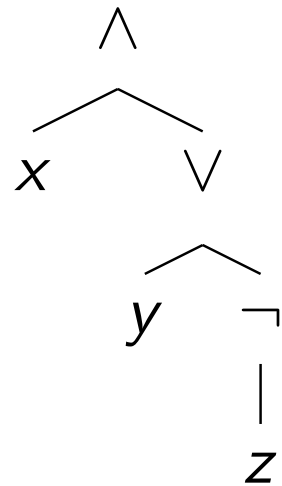


## Définitions inductives

- ▶ Notre définition de la syntaxe de la logique propositionnelle est une **définition inductive**.
- ▶ On verra d'autres exemples de définitions inductives.
- ▶ Ici on ne s'intéresse pas à étudier les définition inductives en général, il suffit de dire qu'elles sont toujours de la forme « **le plus petit ensemble qui satisfait certaines propriétés de clôture** ».

## Syntaxe Abstraite

Syntaxe abstraite de  $(x \wedge (y \vee \neg z))$  :



Structure de la formule, représenté par un **arbre**. On remarque qu'on n'a pas besoin de parenthèses.

## Encodage de l'implication

On peut encoder l'implication de la manière suivante:

$$(p \Rightarrow q) := (\neg p \vee q)$$

$$(p \Leftrightarrow q) := ((p \Rightarrow q) \wedge (q \Rightarrow p))$$

On l'utilisera comme suit:

si $p$ alors $q$	$(p \Rightarrow q)$
$q$ si $p$	$(p \Rightarrow q)$
$q$ seulement si $p$	$(p \Leftarrow q)$ ou $(q \Rightarrow p)$
$q$ uniquement si $p$	$(p \Leftarrow q)$ ou $(q \Rightarrow p)$
$q$ si et seulement si $p$	$(p \Leftrightarrow q)$

## Preuves par induction : Un exemple

### Theorem

*Toute formule propositionnelle a le même nombre de parenthèses ouvrantes que de parenthèses fermantes.*

## Notation

- ▶  $|w|_(_$  dénote le nombre de parenthèses ouvrantes de  $w$ .

## Notation

- ▶  $|w|_{(}$  dénote le nombre de parenthèses ouvrantes de  $w$ .
- ▶  $|w|_{)}$  dénote le nombre de parenthèses fermantes de  $w$ .

## Notation

- ▶  $|w|_{(}$  dénote le nombre de parenthèses ouvrantes de  $w$ .
- ▶  $|w|_{)}$  dénote le nombre de parenthèses fermantes de  $w$ .
- ▶ Par exemple,  $|(x \wedge y)|_{(} = 1$

## Notation

- ▶  $|w|_{(}$  dénote le nombre de parenthèses ouvrantes de  $w$ .
- ▶  $|w|_{)}$  dénote le nombre de parenthèses fermantes de  $w$ .
- ▶ Par exemple,  $|(x \wedge y)|_{(} = 1$
- ▶ Pareil pour des autres symboles :  $|w|_{\neg}$ , etc.



## Notation

- ▶  $|w|_{(}$  dénote le nombre de parenthèses ouvrantes de  $w$ .
- ▶  $|w|_{)}$  dénote le nombre de parenthèses fermantes de  $w$ .
- ▶ Par exemple,  $|(x \wedge y)|_{(} = 1$
- ▶ Pareil pour des autres symboles :  $|w|_{\neg}$ , etc.

À montrer : pour tout  $w \in Form$ ,  $|w|_{(} = |w|_{)}$ .

## Le principe d'une preuve par induction structurelle

Le principe est illustré ici sur l'exemple des formules propositionnelles :

### Theorem

*Soit  $P$  une propriété sur les chaînes de caractères. Supposons que  $P$  vérifie les énoncés de clôture suivants :*

- ▶ *tout élément de  $V$  a la propriété  $P$ ,*
- ▶ *si  $p$  satisfait  $P$  alors  $\neg p$  satisfait  $P$ ,*
- ▶ *si  $p$  et  $q$  satisfont  $P$  alors  $(p \wedge q)$  satisfait  $P$ ,*
- ▶ *si  $p$  et  $q$  satisfont  $P$  alors  $(p \vee q)$  satisfait  $P$ ,*

*Alors tous les éléments de  $\text{Form}$  satisfont  $P$ .*

Exemple de propriété  $P$  : avoir le même nombre de ( que de ).

## Comment rédiger une preuve par induction structurelle ?

À montrer :

pour tout  $w \in Form$ ,  $|w|_{(} = |w|_{)}$ .

## Comment rédiger une preuve par induction structurelle ?

À montrer :

pour tout  $w \in Form$ ,  $|w|_{\zeta} = |w|_{\eta}$ .

(1) Cas des variables :

À montrer :  $|w|_{\zeta} = |w|_{\eta}$  pour tout  $w \in V$ .

## Comment rédiger une preuve par induction structurelle ?

À montrer :

pour tout  $w \in Form$ ,  $|w|_{\zeta} = |w|_{\eta}$ .

(1) Cas des variables :

À montrer :  $|w|_{\zeta} = |w|_{\eta}$  pour tout  $w \in V$ .

Démonstration :

Si  $w \in V$  alors  $|w|_{\zeta} = 0$  et  $|w|_{\eta} = 0$ , donc  $|w|_{\zeta} = |w|_{\eta}$ .

## Rédaction d'une preuve par induction (2)

(2) Cas de la négation :

Soit  $p \in \text{Form}$ . **Hypothèse d'induction** :  $|p|_{\zeta} = |p|_{\eta}$ .

À montrer :  $|\neg p|_{\zeta} = |\neg p|_{\eta}$ .

## Rédaction d'une preuve par induction (2)

(2) Cas de la négation :

Soit  $p \in \text{Form}$ . **Hypothèse d'induction** :  $|p|_{\zeta} = |p|_{\eta}$ .

À montrer :  $|\neg p|_{\zeta} = |\neg p|_{\eta}$ .

Démonstration :

$$|\neg p|_{\zeta} =$$

## Rédaction d'une preuve par induction (2)

(2) Cas de la négation :

Soit  $p \in \text{Form}$ . **Hypothèse d'induction** :  $|p|_{\zeta} = |p|_{\eta}$ .

À montrer :  $|\neg p|_{\zeta} = |\neg p|_{\eta}$ .

Démonstration :

$$|\neg p|_{\zeta} = |p|_{\zeta} \quad \text{par définition } |\cdot|_{\zeta}$$



## Rédaction d'une preuve par induction (2)

(2) Cas de la négation :

Soit  $p \in \text{Form}$ . **Hypothèse d'induction** :  $|p|_{\zeta} = |p|_{\eta}$ .

À montrer :  $|\neg p|_{\zeta} = |\neg p|_{\eta}$ .

Démonstration :

$$\begin{aligned} |\neg p|_{\zeta} &= |p|_{\zeta} && \text{par définition } |\cdot|_{\zeta} \\ &= |p|_{\eta} && \text{par hypothèse d'induction} \end{aligned}$$

## Rédaction d'une preuve par induction (2)

(2) Cas de la négation :

Soit  $p \in \text{Form}$ . **Hypothèse d'induction** :  $|p|_{\zeta} = |p|_{\eta}$ .

À montrer :  $|\neg p|_{\zeta} = |\neg p|_{\eta}$ .

Démonstration :

$$\begin{aligned} |\neg p|_{\zeta} &= |p|_{\zeta} && \text{par définition } |\cdot|_{\zeta} \\ &= |p|_{\eta} && \text{par hypothèse d'induction} \\ &= |\neg p|_{\eta} && \text{par définition } |\cdot|_{\eta} \end{aligned}$$

## Rédaction d'une preuve par induction (3)

### (3) Cas de la conjonction :

Soient  $p, q \in Form$ .

Hypothèse d'induction :  $|p|_{\zeta} = |p|_{\eta}$  et  $|q|_{\zeta} = |q|_{\eta}$

À montrer :  $|(p \wedge q)|_{\zeta} = |(p \wedge q)|_{\eta}$

## Rédaction d'une preuve par induction (3)

### (3) Cas de la conjonction :

Soient  $p, q \in Form$ .

Hypothèse d'induction :  $|p|_{\zeta} = |p|_{\eta}$  et  $|q|_{\zeta} = |q|_{\eta}$

À montrer :  $|(p \wedge q)|_{\zeta} = |(p \wedge q)|_{\eta}$

Démonstration :

$$|(p \wedge q)|_{\zeta} =$$

## Rédaction d'une preuve par induction (3)

### (3) Cas de la conjonction :

Soient  $p, q \in \text{Form}$ .

Hypothèse d'induction :  $|p|_{\zeta} = |p|_{\eta}$  et  $|q|_{\zeta} = |q|_{\eta}$

À montrer :  $|(p \wedge q)|_{\zeta} = |(p \wedge q)|_{\eta}$

### Démonstration :

$$|(p \wedge q)|_{\zeta} = 1 + |p|_{\zeta} + |q|_{\zeta} \quad \text{par définition } |\cdot|_{\zeta}$$

## Rédaction d'une preuve par induction (3)

### (3) Cas de la conjonction :

Soient  $p, q \in \text{Form}$ .

Hypothèse d'induction :  $|p|_{\zeta} = |p|_{\eta}$  et  $|q|_{\zeta} = |q|_{\eta}$

À montrer :  $|(p \wedge q)|_{\zeta} = |(p \wedge q)|_{\eta}$

### Démonstration :

$$\begin{aligned} |(p \wedge q)|_{\zeta} &= 1 + |p|_{\zeta} + |q|_{\zeta} && \text{par définition } |\cdot|_{\zeta} \\ &= |p|_{\eta} + |q|_{\eta} + 1 && \text{par hypothèse d'induction} \end{aligned}$$

## Rédaction d'une preuve par induction (3)

### (3) Cas de la conjonction :

Soient  $p, q \in Form$ .

Hypothèse d'induction :  $|p|_{\zeta} = |p|_{\eta}$  et  $|q|_{\zeta} = |q|_{\eta}$

À montrer :  $|(p \wedge q)|_{\zeta} = |(p \wedge q)|_{\eta}$

### Démonstration :

$$\begin{aligned}
 |(p \wedge q)|_{\zeta} &= 1 + |p|_{\zeta} + |q|_{\zeta} && \text{par définition } |\cdot|_{\zeta} \\
 &= |p|_{\eta} + |q|_{\eta} + 1 && \text{par hypothèse d'induction} \\
 &= |(p \wedge q)|_{\eta} && \text{par définition } |\cdot|_{\eta}
 \end{aligned}$$

## Rédaction d'une preuve par induction (4)

### (4) Cas de la disjonction :

Soient  $p, q \in Form$ .

Hypothèse d'induction :  $|p|_{\zeta} = |p|_{\eta}$  et  $|q|_{\zeta} = |q|_{\eta}$

À montrer :  $|(p \vee q)|_{\zeta} = |(p \vee q)|_{\eta}$



## Rédaction d'une preuve par induction (4)

### (4) Cas de la disjonction :

Soient  $p, q \in \text{Form}$ .

Hypothèse d'induction :  $|p|_{\zeta} = |p|_{\eta}$  et  $|q|_{\zeta} = |q|_{\eta}$

À montrer :  $|(p \vee q)|_{\zeta} = |(p \vee q)|_{\eta}$

Démonstration :

$$|(p \vee q)|_{\zeta} =$$

## Rédaction d'une preuve par induction (4)

### (4) Cas de la disjonction :

Soient  $p, q \in Form$ .

Hypothèse d'induction :  $|p|_{\zeta} = |p|_{\eta}$  et  $|q|_{\zeta} = |q|_{\eta}$

À montrer :  $|(p \vee q)|_{\zeta} = |(p \vee q)|_{\eta}$

### Démonstration :

$$|(p \vee q)|_{\zeta} = 1 + |p|_{\zeta} + |q|_{\zeta} \quad \text{par définition } |\cdot|_{\zeta}$$

## Rédaction d'une preuve par induction (4)

### (4) Cas de la disjonction :

Soient  $p, q \in \text{Form}$ .

Hypothèse d'induction :  $|p|_{\zeta} = |p|_{\eta}$  et  $|q|_{\zeta} = |q|_{\eta}$

À montrer :  $|(p \vee q)|_{\zeta} = |(p \vee q)|_{\eta}$

### Démonstration :

$$\begin{aligned} |(p \vee q)|_{\zeta} &= 1 + |p|_{\zeta} + |q|_{\zeta} && \text{par définition } |\cdot|_{\zeta} \\ &= |p|_{\eta} + |q|_{\eta} + 1 && \text{par hypothèse d'induction} \end{aligned}$$

## Rédaction d'une preuve par induction (4)

### (4) Cas de la disjonction :

Soient  $p, q \in Form$ .

Hypothèse d'induction :  $|p|_{\zeta} = |p|_{\eta}$  et  $|q|_{\zeta} = |q|_{\eta}$

À montrer :  $|(p \vee q)|_{\zeta} = |(p \vee q)|_{\eta}$

### Démonstration :

$$\begin{aligned}
 |(p \vee q)|_{\zeta} &= 1 + |p|_{\zeta} + |q|_{\zeta} && \text{par définition } |\cdot|_{\zeta} \\
 &= |p|_{\eta} + |q|_{\eta} + 1 && \text{par hypothèse d'induction} \\
 &= |(p \vee q)|_{\eta} && \text{par définition } |\cdot|_{\eta}
 \end{aligned}$$

## Remarques

- ▶ On utilise le théorème sur le principe des preuves par induction structurelle.

## Remarques

- ▶ On utilise le théorème sur le principe des preuves par induction structurelle.
- ▶ Toujours dire clairement quelles sont les hypothèses, et qu'est-ce qu'il faut montrer.

## Remarques

- ▶ On utilise le théorème sur le principe des preuves par induction structurelle.
- ▶ Toujours dire clairement quelles sont les hypothèses, et qu'est-ce qu'il faut montrer.
- ▶ Justifier les étapes du raisonnement.

## Remarques

- ▶ On utilise le théorème sur le principe des preuves par induction structurelle.
- ▶ Toujours dire clairement quelles sont les hypothèses, et qu'est-ce qu'il faut montrer.
- ▶ Justifier les étapes du raisonnement.
- ▶ Parfois (mais pas toujours !), le cas  $\vee$  et le cas  $\wedge$  sont très similaires.



## Retour au théorème

### Theorem

*Soit  $P$  une propriété sur les chaînes de caractères. Supposons que  $P$  vérifie les énoncés de clôture suivants :*

- ▶ *tout élément de  $V$  a la propriété  $P$ ,*
- ▶ *si  $p$  satisfait  $P$  alors  $\neg p$  satisfait  $P$ ,*
- ▶ *si  $p$  et  $q$  satisfont  $P$  alors  $(p \wedge q)$  satisfait  $P$ ,*
- ▶ *si  $p$  et  $q$  satisfont  $P$  alors  $(p \vee q)$  satisfait  $P$ ,*

*Alors tous les éléments de  $\text{Form}$  satisfont  $P$ .*

## Démonstration

- Soit  $X$  l'ensemble de toutes les chaînes de caractères avec la propriété  $P$ .

## Démonstration

- ▶ Soit  $X$  l'ensemble de toutes les chaînes de caractères avec la propriété  $P$ .
- ▶ On remarque donc que l'ensemble  $X$  satisfait les propriétés de clôture des formules propositionnelles (par l'hypothèse du théorème).

## Démonstration

- ▶ Soit  $X$  l'ensemble de toutes les chaînes de caractères avec la propriété  $P$ .
- ▶ On remarque donc que l'ensemble  $X$  satisfait les propriétés de clôture des formules propositionnelles (par l'hypothèse du théorème).
- ▶ Donc  $Form \subseteq X$  car  $Form$  est le plus petit ensemble de chaînes de caractères qui satisfait ces propriétés.

## Démonstration

- ▶ Soit  $X$  l'ensemble de toutes les chaînes de caractères avec la propriété  $P$ .
- ▶ On remarque donc que l'ensemble  $X$  satisfait les propriétés de clôture des formules propositionnelles (par l'hypothèse du théorème).
- ▶ Donc  $Form \subseteq X$  car  $Form$  est le plus petit ensemble de chaînes de caractères qui satisfait ces propriétés.
- ▶ Autrement dit, tout élément de  $Form$  a la propriété  $P$ .

## Définition d'une fonction

- Une méthode pour définir une fonction : par une expression close.

## Définition d'une fonction

- ▶ Une méthode pour définir une fonction : par une expression close.
- ▶ Par exemple, la fonction qui envoie un argument, qui est un nombre naturel, vers l'argument incrémenté de 1 :

$$f(x) := x + 1$$

La variable  $x$  est le paramètre formel de la fonction.

## Définition d'une fonction

- ▶ Une méthode pour définir une fonction : par une expression close.
- ▶ Par exemple, la fonction qui envoie un argument, qui est un nombre naturel, vers l'argument incrémenté de 1 :

$$f(x) := x + 1$$

La variable  $x$  est le paramètre formel de la fonction.

- ▶ Pour appliquer une telle fonction : remplacer dans l'expression le paramètre formel par le paramètre actuel, puis évaluer l'expression.



## Définition d'une fonction

- ▶ Une méthode pour définir une fonction : par une expression close.
- ▶ Par exemple, la fonction qui envoie un argument, qui est un nombre naturel, vers l'argument incrémenté de 1 :

$$f(x) := x + 1$$

La variable  $x$  est le paramètre formel de la fonction.

- ▶ Pour appliquer une telle fonction : remplacer dans l'expression le paramètre formel par le paramètre actuel, puis évaluer l'expression.
- ▶ Sur l'exemple :

$$f(5) = 5 + 1 = 6$$

## Définition de fonctions sur *Form*

- Problème : comment définir une fonction sur un ensemble qui est défini par induction ?

## Définition de fonctions sur *Form*

- ▶ Problème : comment définir une fonction sur un ensemble qui est défini par induction ?
- ▶ Exemple : comment définir la fonction qui donne la longueur d'une formule, ou le nombre de parenthèses ?

## Définition de fonctions sur *Form*

- ▶ Problème : comment définir une fonction sur un ensemble qui est défini par induction ?
- ▶ Exemple : comment définir la fonction qui donne la longueur d'une formule, ou le nombre de parenthèses ?
- ▶ En général, l'évaluation d'une telle fonction appliquée à une formule dépend de la **structure** de la formule.

## Définition par récurrence

On peut définir une fonction avec domaine *Form* comme suit :

1. on donne le résultat de la fonction appliquée à un élément quelconque de  $V$ ,

## Définition par récurrence

On peut définir une fonction avec domaine *Form* comme suit :

1. on donne le résultat de la fonction appliquée à un élément quelconque de  $V$ ,
2. on donne le résultat de la fonction appliquée à une formule de la forme  $\neg p$ , sachant quel est le résultat de la fonction appliquée à  $p$ ,

## Définition par récurrence

On peut définir une fonction avec domaine *Form* comme suit :

1. on donne le résultat de la fonction appliquée à un élément quelconque de  $V$ ,
2. on donne le résultat de la fonction appliquée à une formule de la forme  $\neg p$ , sachant quel est le résultat de la fonction appliquée à  $p$ ,
3. on donne le résultat de la fonction appliquée à une formule de la forme  $(p \wedge q)$ , sachant quel est le résultat de la fonction appliquée à  $p$  et le résultat de la fonction appliquée à  $q$ ,

## Définition par récurrence

On peut définir une fonction avec domaine *Form* comme suit :

1. on donne le résultat de la fonction appliquée à un élément quelconque de  $V$ ,
2. on donne le résultat de la fonction appliquée à une formule de la forme  $\neg p$ , sachant quel est le résultat de la fonction appliquée à  $p$ ,
3. on donne le résultat de la fonction appliquée à une formule de la forme  $(p \wedge q)$ , sachant quel est le résultat de la fonction appliquée à  $p$  et le résultat de la fonction appliquée à  $q$ ,
4. on donne le résultat de la fonction appliquée à une formule de la forme  $(p \vee q)$ , sachant quel est le résultat de la fonction appliquée à  $p$  et le résultat de la fonction appliquée à  $q$ .



## Exemple récurrence : *length*

La fonction *length* est définie comme suit :

1.  $length(x) = 1$  si  $x \in V$
2.  $length(\neg p) = 1 + length(p)$
3.  $length((p \wedge q)) = 3 + length(p) + length(q)$
4.  $length((p \vee q)) = 3 + length(p) + length(q)$

## Exemple récurrence : length

Évaluation :

$$\text{length}((x_1 \wedge \neg x_2)) =$$

## Exemple récurrence : length

Évaluation :

$$\text{length}((x_1 \wedge \neg x_2)) = 3 + \text{length}(x_1) + \text{length}(\neg x_2) \quad (3)$$

## Exemple récurrence : length

Évaluation :

$$\begin{aligned} \text{length}((x_1 \wedge \neg x_2)) &= 3 + \text{length}(x_1) + \text{length}(\neg x_2) && (3) \\ &= 3 + 1 + \text{length}(\neg x_2) && (1) \end{aligned}$$

## Exemple récurrence : length

Évaluation :

$$\begin{aligned} \text{length}((x_1 \wedge \neg x_2)) &= 3 + \text{length}(x_1) + \text{length}(\neg x_2) & (3) \\ &= 3 + 1 + \text{length}(\neg x_2) & (1) \\ &= 3 + 1 + 1 + \text{length}(x_2) & (2) \end{aligned}$$

## Exemple récurrence : *length*

Évaluation :

$$\begin{aligned} \textit{length}((x_1 \wedge \neg x_2)) &= 3 + \textit{length}(x_1) + \textit{length}(\neg x_2) && (3) \\ &= 3 + 1 + \textit{length}(\neg x_2) && (1) \\ &= 3 + 1 + 1 + \textit{length}(x_2) && (2) \\ &= 3 + 1 + 1 + 1 && (1) \end{aligned}$$

## Exemple récurrence : length

Évaluation :

$$\begin{aligned} \text{length}((x_1 \wedge \neg x_2)) &= 3 + \text{length}(x_1) + \text{length}(\neg x_2) && (3) \\ &= 3 + 1 + \text{length}(\neg x_2) && (1) \\ &= 3 + 1 + 1 + \text{length}(x_2) && (2) \\ &= 3 + 1 + 1 + 1 && (1) \\ &= 6 \end{aligned}$$

## Exemple de récurrence : $\mathcal{V}$

Notre deuxième exemple est la fonction  $\mathcal{V}$ , définie comme suit :

1.  $\mathcal{V}(x) = \{x\}$  si  $x \in V$
2.  $\mathcal{V}(\neg p) = \mathcal{V}(p)$
3.  $\mathcal{V}((p \wedge q)) = \mathcal{V}(p) \cup \mathcal{V}(q)$
4.  $\mathcal{V}((p \vee q)) = \mathcal{V}(p) \cup \mathcal{V}(q)$



## Exemple de récurrence : $\mathcal{V}$

Notre deuxième exemple est la fonction  $\mathcal{V}$ , définie comme suit :

1.  $\mathcal{V}(x) = \{x\}$  si  $x \in V$
2.  $\mathcal{V}(\neg p) = \mathcal{V}(p)$
3.  $\mathcal{V}((p \wedge q)) = \mathcal{V}(p) \cup \mathcal{V}(q)$
4.  $\mathcal{V}((p \vee q)) = \mathcal{V}(p) \cup \mathcal{V}(q)$

$\mathcal{V}(p)$  est l'ensemble des variables de  $p$ .

## Exemple de récurrence : $\mathcal{V}$

Évaluation :

$$\mathcal{V}((x_1 \wedge (x_2 \vee x_3))) =$$

## Exemple de récurrence : $\mathcal{V}$

Évaluation :

$$\mathcal{V}((x_1 \wedge (x_2 \vee x_3))) = \mathcal{V}(x_1) \cup \mathcal{V}((x_2 \vee x_3)) \quad (3)$$

## Exemple de récurrence : $\mathcal{V}$

Évaluation :

$$\begin{aligned}\mathcal{V}((x_1 \wedge (x_2 \vee x_3))) &= \mathcal{V}(x_1) \cup \mathcal{V}((x_2 \vee x_3)) && (3) \\ &= \{x_1\} \cup \mathcal{V}(x_2) \cup \mathcal{V}(x_3) && (1), (4)\end{aligned}$$

## Exemple de récurrence : $\mathcal{V}$

Évaluation :

$$\begin{aligned}\mathcal{V}((x_1 \wedge (x_2 \vee x_3))) &= \mathcal{V}(x_1) \cup \mathcal{V}((x_2 \vee x_3)) && (3) \\ &= \{x_1\} \cup \mathcal{V}(x_2) \cup \mathcal{V}(x_3) && (1), (4) \\ &= \{x_1\} \cup \{x_2\} \cup \{x_3\} && (1), (1)\end{aligned}$$

## Exemple de récurrence : $\mathcal{V}$

Évaluation :

$$\begin{aligned}\mathcal{V}((x_1 \wedge (x_2 \vee x_3))) &= \mathcal{V}(x_1) \cup \mathcal{V}((x_2 \vee x_3)) && (3) \\ &= \{x_1\} \cup \mathcal{V}(x_2) \cup \mathcal{V}(x_3) && (1), (4) \\ &= \{x_1\} \cup \{x_2\} \cup \{x_3\} && (1), (1) \\ &= \{x_1, x_2, x_3\}\end{aligned}$$

## Une subtilité

- Une fonction doit toujours associer à un argument donné un seul résultat. On doit donc assurer que la définition récursive d'une fonction garantit bien cette unicité du résultat.

## Une subtilité

- ▶ Une fonction doit toujours associer à un argument donné un seul résultat. On doit donc assurer que la définition récursive d'une fonction garantit bien cette unicité du résultat.
- ▶ En principe, la même formule pourrait être construite de deux façon différentes. Dans ce cas, la définition de la fonction risque de donner deux valeurs différentes selon la construction considérée.



## Une subtilité

- ▶ Une fonction doit toujours associer à un argument donné un seul résultat. On doit donc assurer que la définition récursive d'une fonction garantit bien cette unicité du résultat.
- ▶ En principe, la même formule pourrait être construite de deux façon différentes. Dans ce cas, la définition de la fonction risque de donner deux valeurs différentes selon la construction considérée.
- ▶ Heureusement, cette difficulté n'existe pas pour notre définition des formules propositionnelles : *théorème de lecture unique* (démonstration omise).

## Une subtilité

- ▶ Une fonction doit toujours associer à un argument donné un seul résultat. On doit donc assurer que la définition récursive d'une fonction garantit bien cette unicité du résultat.
- ▶ En principe, la même formule pourrait être construite de deux façon différentes. Dans ce cas, la définition de la fonction risque de donner deux valeurs différentes selon la construction considérée.
- ▶ Heureusement, cette difficulté n'existe pas pour notre définition des formules propositionnelles : *théorème de lecture unique* (démonstration omise).
- ▶ Mais attention dans le cas général des ensembles définis par induction.

## Récurrence et induction

- Un ensemble peut être défini par **induction**: on dit comment construire un nouvel élément de l'ensemble à partir des éléments plus primitifs. Il y donc un sens « ascendant ».

## Récurrence et induction

- ▶ Un ensemble peut être défini par **induction**: on dit comment construire un nouvel élément de l'ensemble à partir des éléments plus primitifs. Il y donc un sens « ascendant ».
- ▶ Les fonctions peuvent être définies par **récurrence** : on définit le résultat d'une fonction appliquée sur un argument composé en faisant référence au résultat de la fonction sur des arguments plus simples. Il y a donc un sens « descendant ».

## Récurrence et induction

- ▶ Un ensemble peut être défini par **induction**: on dit comment construire un nouvel élément de l'ensemble à partir des éléments plus primitifs. Il y donc un sens « ascendant ».
- ▶ Les fonctions peuvent être définies par **récurrence** : on définit le résultat d'une fonction appliquée sur un argument composé en faisant référence au résultat de la fonction sur des arguments plus simples. Il y a donc un sens « descendant ».
- ▶ Finalement, une propriété de tous les éléments d'un ensemble qui est défini par induction est normalement démontrée par **induction structurelle**.

# Sémantique de la logique propositionnelle

# Affectations

Valeurs de vérités : 0 et 1.

# Affectations

Valeurs de vérités : 0 et 1.

## Definition

Une *affectation* est une fonction

$$v : V \rightarrow \{0, 1\}$$

Le **support** d'une affectation  $v$  est défini comme

$$\text{supp}(v) = \{x \in V \mid v(x) = 1\}$$



## Il existe des définitions différentes

- ▶ Pour 0 : False, ff, ...
- ▶ Pour 1 : True, tt, ...

## Il existe des définitions différentes

- ▶ Pour 0 : False, ff, ...
- ▶ Pour 1 : True, tt, ...
- ▶ Les affectations sont parfois définies comme des fonctions **partielles**.

## Notation pour les affectations

► Nous écrivons

$$[x_1 \mapsto 1, x_2 \mapsto 1, x_3 \mapsto 1, \dots, x_n \mapsto 1]$$

pour l'affectation qui aux variables  $x_1, \dots, x_n$  associe la valeur 1, et qui associe à toute autre variable la valeur 0.

## Notation pour les affectations

- Nous écrivons

$$[x_1 \mapsto 1, x_2 \mapsto 1, x_3 \mapsto 1, \dots, x_n \mapsto 1]$$

pour l'affectation qui aux variables  $x_1, \dots, x_n$  associe la valeur 1, et qui associe à toute autre variable la valeur 0.

- On ne peut pas écrire toutes les affectations de cette façon.

## Exemple d'une affectation

$$[x_1 \mapsto 1, x_3 \mapsto 1]$$

est l'affectation qui associe à  $x_1$  et  $x_3$  la valeur 1, et à toute autre variable la valeur 0.

Son support est  $\{x_1, x_3\}$ .

## Interprétation d'une formule

L'interprétation  $\llbracket p \rrbracket v$  d'une formule  $p$  par rapport à l'affectation  $v$  est définie par récurrence sur la structure de  $p$ :

$$\llbracket x \rrbracket v = v(x)$$

$$\llbracket \neg p_1 \rrbracket v = \begin{cases} 0 & \text{si } \llbracket p_1 \rrbracket v = 1 \\ 1 & \text{si } \llbracket p_1 \rrbracket v = 0 \end{cases}$$

$$\llbracket (p_1 \wedge p_2) \rrbracket v = \begin{cases} 0 & \text{si } \llbracket p_1 \rrbracket v = 0 \text{ ou } \llbracket p_2 \rrbracket v = 0 \\ 1 & \text{si } \llbracket p_1 \rrbracket v = 1 \text{ et } \llbracket p_2 \rrbracket v = 1 \end{cases}$$

$$\llbracket (p_1 \vee p_2) \rrbracket v = \begin{cases} 0 & \text{si } \llbracket p_1 \rrbracket v = 0 \text{ et } \llbracket p_2 \rrbracket v = 0 \\ 1 & \text{si } \llbracket p_1 \rrbracket v = 1 \text{ ou } \llbracket p_2 \rrbracket v = 1 \end{cases}$$

*Handwritten note:*  $\neg \llbracket p_1 \rrbracket v = 1 \Rightarrow \llbracket \neg p_1 \rrbracket v = 0$

## Exemples

Pour l'affectation  $v_1 = [y \mapsto 1]$  on a que

►  $\llbracket x \rrbracket v_1 = 0$  (car  $v_1(x) = 0$ )

## Exemples

Pour l'affectation  $v_1 = [y \mapsto 1]$  on a que

- ▶  $\llbracket x \rrbracket v_1 = 0$  (car  $v_1(x) = 0$ )
- ▶  $\llbracket y \rrbracket v_1 = 1$  (car  $v_1(y) = 1$ )



## Exemples

Pour l'affectation  $v_1 = [y \mapsto 1]$  on a que

- ▶  $\llbracket x \rrbracket v_1 = 0$  (car  $v_1(x) = 0$ )
- ▶  $\llbracket y \rrbracket v_1 = 1$  (car  $v_1(y) = 1$ )
- ▶ Donc :  $\llbracket (x \wedge y) \rrbracket v_1 = 0$

## Exemples

Pour l'affectation  $v_1 = [y \mapsto 1]$  on a que

- ▶  $\llbracket x \rrbracket v_1 = 0$  (car  $v_1(x) = 0$ )
- ▶  $\llbracket y \rrbracket v_1 = 1$  (car  $v_1(y) = 1$ )
- ▶ Donc :  $\llbracket (x \wedge y) \rrbracket v_1 = 0$
- ▶ et  $\llbracket (x \vee y) \rrbracket v_1 = 1$ .

## Stratégie d'interprétation

### Theorem

Soient  $p$  et  $q$  des formules propositionnelles et  $v$  une affectation, alors

$$\llbracket (p \wedge q) \rrbracket v = \begin{cases} 0 & \text{si } \llbracket p \rrbracket v = 0 \\ \llbracket q \rrbracket v & \text{si } \llbracket p \rrbracket v = 1 \end{cases}$$

$$\llbracket (p \vee q) \rrbracket v = \begin{cases} 1 & \text{si } \llbracket p \rrbracket v = 1 \\ \llbracket q \rrbracket v & \text{si } \llbracket p \rrbracket v = 0 \end{cases}$$

on peut écrire aussi

$$\llbracket (p \wedge q) \rrbracket v = \begin{cases} 0 & \text{si } \llbracket q \rrbracket v = 0 \\ \llbracket p \rrbracket v & \text{si } \llbracket q \rrbracket v = 1 \end{cases}$$

## Démonstration

Nous démontrons seulement le premier des deux énoncés; le second se montre de façon analogue. Il y a deux cas, selon la valeur de  $\llbracket p \rrbracket v$  :

Cas  $\llbracket p \rrbracket v = 0$ : Nous avons à montrer que dans ce cas  $\llbracket (p \wedge q) \rrbracket v = 0$ . C'est une conséquence immédiate de la définition.

## Démonstration

Nous démontrons seulement le premier des deux énoncés; le second se montre de façon analogue. Il y a deux cas, selon la valeur de  $\llbracket p \rrbracket v$  :

Cas  $\llbracket p \rrbracket v = 0$ : Nous avons à montrer que dans ce cas  $\llbracket (p \wedge q) \rrbracket v = 0$ . C'est une conséquence immédiate de la définition.

Cas  $\llbracket p \rrbracket v = 1$ : Nous avons à montrer que dans ce cas  $\llbracket (p \wedge q) \rrbracket v = \llbracket q \rrbracket v$ . Il y a deux cas, selon la valeur de  $\llbracket q \rrbracket v$ :

## Démonstration

Nous démontrons seulement le premier des deux énoncés; le second se montre de façon analogue. Il y a deux cas, selon la valeur de  $\llbracket p \rrbracket v$  :

Cas  $\llbracket p \rrbracket v = 0$ : Nous avons à montrer que dans ce cas  $\llbracket (p \wedge q) \rrbracket v = 0$ . C'est une conséquence immédiate de la définition.

Cas  $\llbracket p \rrbracket v = 1$ : Nous avons à montrer que dans ce cas  $\llbracket (p \wedge q) \rrbracket v = \llbracket q \rrbracket v$ . Il y a deux cas, selon la valeur de  $\llbracket q \rrbracket v$ :

Cas  $\llbracket q \rrbracket v = 0$ : Nous avons  $\llbracket (p \wedge q) \rrbracket v = 0 = \llbracket q \rrbracket v$

## Démonstration

Nous démontrons seulement le premier des deux énoncés; le second se montre de façon analogue. Il y a deux cas, selon la valeur de  $\llbracket p \rrbracket v$  :

Cas  $\llbracket p \rrbracket v = 0$ : Nous avons à montrer que dans ce cas  $\llbracket (p \wedge q) \rrbracket v = 0$ . C'est une conséquence immédiate de la définition.

Cas  $\llbracket p \rrbracket v = 1$ : Nous avons à montrer que dans ce cas  $\llbracket (p \wedge q) \rrbracket v = \llbracket q \rrbracket v$ . Il y a deux cas, selon la valeur de  $\llbracket q \rrbracket v$ :

Cas  $\llbracket q \rrbracket v = 0$ : Nous avons  $\llbracket (p \wedge q) \rrbracket v = 0 = \llbracket q \rrbracket v$

Cas  $\llbracket q \rrbracket v = 1$ : Nous avons  $\llbracket (p \wedge q) \rrbracket v = 1 = \llbracket q \rrbracket v$

## L'intérêt de ce théorème

- Pour évaluer  $\llbracket (p \wedge q) \rrbracket v$  on évalue d'abord  $\llbracket p \rrbracket v$ , et selon le résultat obtenu il se peut qu'il ne soit plus nécessaire d'évaluer  $\llbracket q \rrbracket v$ , ce qui est bon à savoir quand  $q$  est une très grande expression.



## L'intérêt de ce théorème

- Pour évaluer  $\llbracket (p \wedge q) \rrbracket v$  on évalue d'abord  $\llbracket p \rrbracket v$ , et selon le résultat obtenu il se peut qu'il ne soit plus nécessaire d'évaluer  $\llbracket q \rrbracket v$ , ce qui est bon à savoir quand  $q$  est une très grande expression.
- On aurait pu donner une variante du théorème dans laquelle on interprète d'abord  $q$  au lieu de  $p$ , ou encore des variantes avec des critères plus sophistiqués (par exemple: on commence avec l'interprétation de la formule parmi  $p, q$  qui est la plus petite).

## Évaluation et interprétation

- ▶ En général, une application d'une fonction à des arguments est **évaluée**.
- ▶ Plus spécifiquement, une formule propositionnelle est **interprétée** par rapport à une affectation.

L'interprétation de  $p$  par rapport à  $v$  est obtenue par l'évaluation de  $\llbracket p \rrbracket v$ .

## Définition

Soit  $p$  une formule propositionnelle.

- On écrit  $v \models p$  si  $\llbracket p \rrbracket v = 1$ , et on dit « $p$  est vraie par rapport à  $v$ ».

*v rend vrai p*

## Définition

Soit  $p$  une formule propositionnelle.

- ▶ On écrit  $v \models p$  si  $\llbracket p \rrbracket v = 1$ , et on dit « $p$  est vraie par rapport à  $v$ ».
- ▶ On écrit  $v \not\models p$  si  $\llbracket p \rrbracket v = 0$ , et on dit « $p$  est fausse par rapport à  $v$ ».

Soit l'un ou l'autre

## Définition

Soit  $p$  une formule propositionnelle.

- ▶ On écrit  $v \models p$  si  $\llbracket p \rrbracket v = 1$ , et on dit « $p$  est vraie par rapport à  $v$ ».
- ▶ On écrit  $v \not\models p$  si  $\llbracket p \rrbracket v = 0$ , et on dit « $p$  est fausse par rapport à  $v$ ».
- ▶ On dit que  $p$  est satisfaisable s'il existe une affectation  $v$  telle que  $v \models p$ .

Exemple formule satisfaisable :  $X$

$$v = [x \mapsto 1]$$


ou si  $\neg x$

## Définition

Soit  $p$  une formule propositionnelle.

- ▶ On écrit  $v \models p$  si  $\llbracket p \rrbracket v = 1$ , et on dit « $p$  est vraie par rapport à  $v$ ».
- ▶ On écrit  $v \not\models p$  si  $\llbracket p \rrbracket v = 0$ , et on dit « $p$  est fausse par rapport à  $v$ ».
- ▶ On dit que  $p$  est satisfaisable s'il existe une affectation  $v$  telle que  $v \models p$ .
- ▶ On dit que  $p$  est falsifiable s'il existe une affectation  $v$  telle que  $v \not\models p$ .

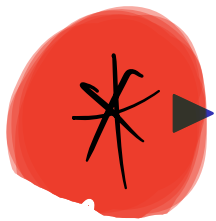
$\exists x$   
 $v = [ ]$



## Définition

Soit  $p$  une formule propositionnelle.

- ▶ On écrit  $v \models p$  si  $\llbracket p \rrbracket v = 1$ , et on dit « $p$  est **vraie par rapport à**  $v$  ».
- ▶ On écrit  $v \not\models p$  si  $\llbracket p \rrbracket v = 0$ , et on dit « $p$  est **fausse par rapport à**  $v$  ».
- ▶ On dit que  $p$  est **satisfaisable** s'il existe une affectation  $v$  telle que  $v \models p$ .
- ▶ On dit que  $p$  est **falsifiable** s'il existe une affectation  $v$  telle que  $v \not\models p$ .



▶ On écrit  $\models p$  si  $v \models p$  pour toute affectation  $v$ , et on dit que  $p$  est **valide** (ou une *tautologie*). *par falsifiable*

$(\times \vee \neg \times)$

## Définition

Soit  $p$  une formule propositionnelle.

- ▶ On écrit  $v \models p$  si  $\llbracket p \rrbracket v = 1$ , et on dit « $p$  est vraie par rapport à  $v$ ».
- ▶ On écrit  $v \not\models p$  si  $\llbracket p \rrbracket v = 0$ , et on dit « $p$  est fausse par rapport à  $v$ ».
- ▶ On dit que  $p$  est satisfaisable s'il existe une affectation  $v$  telle que  $v \models p$ .
- ▶ On dit que  $p$  est falsifiable s'il existe une affectation  $v$  telle que  $v \not\models p$ .
- ▶ On écrit  $\models p$  si  $v \models p$  pour toute affectation  $v$ , et on dit que  $p$  est valide (ou une tautologie).  $\rightarrow$  *très vrai*
- ▶ On écrit  $\not\models p$  si  $v \not\models p$  pour toute affectation  $v$ , et on dit que  $p$  est contradictoire.



## Exemples

La formule  $(x \vee y)$  est

- ▶ satisfaisable (elle est vraie par rapport à  $[x \mapsto 1]$ )

## Exemples

La formule  $(x \vee y)$  est

- ▶ satisfaisable (elle est vraie par rapport à  $[x \mapsto 1]$ )
- ▶ falsifiable (elle est fausse par rapport à  $[]$ )

## Exemples

La formule  $(x \vee y)$  est

- ▶ satisfaisable (elle est vraie par rapport à  $[x \mapsto 1]$ )
- ▶ falsifiable (elle est fausse par rapport à  $[]$ )
- ▶ pas une tautologie

## Exemples

La formule  $(x \vee y)$  est

- ▶ satisfaisable (elle est vraie par rapport à  $[x \mapsto 1]$ )
- ▶ falsifiable (elle est fausse par rapport à  $[]$ )
- ▶ pas une tautologie
- ▶ pas contradictoire

## Exemples

- ▶  $(x \vee \neg x)$  est une tautologie.

## Exemples

- ▶  $(x \vee \neg x)$  est une tautologie.
- ▶  $(x \wedge \neg x)$  est contradictoire.

## Notions de sémantique

Ne pas confondre les notations:

- ▶ Une formule est **vraie** ou **fausse** toujours par rapport à une affectation.
- ▶ Une formule peut être **satisfaisable** ou **falsifiable** tout court. Il n'y a pas de «satisfaisable par une affectation ».
- ▶ Une formule peut être **valide** ou **contradictoire**.

## Proposition :

Une formule  $p$  est valide si et seulement si  $\neg p$  n'est pas satisfaisable.



## Proposition :

Une formule  $p$  est valide si et seulement si  $\neg p$  n'est pas satisfaisable.

## Démonstration :

On a la chaîne d'équivalences suivante :  
 $p$  est valide

## Proposition :

Une formule  $p$  est valide si et seulement si  $\neg p$  n'est pas satisfaisable.

## Démonstration :

On a la chaîne d'équivalences suivante :

$p$  est valide  
ssi  $v \models p$  pour toute affectation  $v$   *$p$  est vrai par rapport à  $v$*

## Proposition :

Une formule  $p$  est valide si et seulement si  $\neg p$  n'est pas satisfaisable.

## Démonstration :

On a la chaîne d'équivalences suivante :

$p$  est valide

ssi  $v \models p$  pour toute affectation  $v$

ssi  $\llbracket p \rrbracket v = 1$  pour toute affectation  $v$

## Proposition :

Une formule  $p$  est valide si et seulement si  $\neg p$  n'est pas satisfaisable.

## Démonstration :

On a la chaîne d'équivalences suivante :

$p$  est valide

ssi  $v \models p$  pour toute affectation  $v$

ssi  $\llbracket p \rrbracket v = 1$  pour toute affectation  $v$

ssi  $\llbracket \neg p \rrbracket v = 0$  pour toute affectation  $v$

## Proposition :

Une formule  $p$  est valide si et seulement si  $\neg p$  n'est pas satisfaisable.

## Démonstration :

On a la chaîne d'équivalences suivante :

$p$  est valide

ssi  $v \models p$  pour toute affectation  $v$

ssi  $\llbracket p \rrbracket v = 1$  pour toute affectation  $v$

ssi  $\llbracket \neg p \rrbracket v = 0$  pour toute affectation  $v$

ssi  $v \models \neg p$  pour **aucune** affectation  $v$

## Proposition :

Une formule  $p$  est valide si et seulement si  $\neg p$  n'est pas satisfaisable.

## Démonstration :

On a la chaîne d'équivalences suivante :

$p$  est valide

ssi  $v \models p$  pour toute affectation  $v$

ssi  $\llbracket p \rrbracket v = 1$  pour toute affectation  $v$

ssi  $\llbracket \neg p \rrbracket v = 0$  pour toute affectation  $v$

ssi  $v \models \neg p$  pour **aucune** affectation  $v$

ssi  $\neg p$  n'est pas satisfaisable

## Proposition :

Une formule  $p$  est contradictoire si et seulement si  $\neg p$  est valide.

(Exercice !)

## Décider validité etc.

- Pour savoir si une formule propositionnelle donnée est satisfaisable ou valide il faut donc en principe évaluer la formule sur toutes les affectations possibles.



## Décider validité etc.

- ▶ Pour savoir si une formule propositionnelle donnée est satisfaisable ou valide il faut donc en principe évaluer la formule sur toutes les affectations possibles.
- ▶ Problème : il y a un nombre infini d'affectations possibles car il y a un nombre infini de variables propositionnelles !

## Décider validité etc.

- ▶ Pour savoir si une formule propositionnelle donnée est satisfaisable ou valide il faut donc en principe évaluer la formule sur toutes les affectations possibles.
- ▶ Problème : il y a un nombre infini d'affectations possibles car il y a un nombre infini de variables propositionnelles !
- ▶ Heureusement, le théorème suivant dit que seulement les variables qui apparaissent dans les formules sont pertinentes.

## D'abord une proposition

### Proposition :

Soit  $p$  une formule propositionnelle et  $v_1, v_2$  des affectations telles que  $v_1(x) = v_2(x)$  pour toute variable  $x \in \mathcal{V}(p)$ . Alors

$$\llbracket p \rrbracket v_1 = \llbracket p \rrbracket v_2.$$

Exercice (sera fait en TD) !

toutes les variables dans  
 $p$

## Le théorème de coïncidence

### Theorem

*Une formule  $p$  est*

- 1. satisfaisable si et seulement s'il existe une affectation  $v$  telle que  $\text{supp}(v) \subseteq \mathcal{V}(p)$  et  $v \models p$ .*
- 2. valide si et seulement si  $v \models p$  pour toute affectation  $v$  avec  $\text{supp}(v) \subseteq \mathcal{V}(p)$ .*

## Démonstration du premier énoncé

Si  $v \models p$  avec  $\text{supp}(v) \subseteq \mathcal{V}(p)$  alors  $p$  est, par définition satisfaisable.

## Démonstration du premier énoncé

Si  $v \models p$  avec  $\text{supp}(v) \subseteq \mathcal{V}(p)$  alors  $p$  est, par définition satisfaisable.

Si  $p$  est satisfaisable il y a une affectation  $w$  telle que  $w \models p$ . Nous construisons une nouvelle affectation  $v$  comme suit:

$$v(x) = \begin{cases} w(x) & \text{si } x \in \mathcal{V}(p) \\ 0 & \text{si } x \notin \mathcal{V}(p) \end{cases}$$

## Démonstration du premier énoncé

Si  $v \models p$  avec  $\text{supp}(v) \subseteq \mathcal{V}(p)$  alors  $p$  est, par définition satisfaisable.

Si  $p$  est satisfaisable il y a une affectation  $w$  telle que  $w \models p$ . Nous construisons une nouvelle affectation  $v$  comme suit:

$$v(x) = \begin{cases} w(x) & \text{si } x \in \mathcal{V}(p) \\ 0 & \text{si } x \notin \mathcal{V}(p) \end{cases}$$

On a que  $\text{supp}(v) \subseteq \mathcal{V}(p)$ , et  $v \models p$  par la proposition précédente.

# Une méthode pour décider la satisfaisabilité d'une formule $p$

1. Calculer  $\mathcal{V}(p)$



## Une méthode pour décider la satisfaisabilité d'une formule $p$

1. Calculer  $\mathcal{V}(p)$
2. Engendrer l'ensemble  $A$  des affectations  $v$  avec  
 $\text{supp}(v) \subseteq \mathcal{V}(p)$

## Une méthode pour décider la satisfaisabilité d'une formule $p$

1. Calculer  $\mathcal{V}(p)$
2. Engendrer l'ensemble  $A$  des affectations  $v$  avec  $\text{supp}(v) \subseteq \mathcal{V}(p)$
3. Évaluer  $\llbracket p \rrbracket v$  pour toute affectation  $v \in A$ . Dès qu'on tombe sur un  $v$  tel que  $\llbracket p \rrbracket v = 1$  on sait que  $p$  est satisfaisable, si on n'en trouve pas alors  $p$  n'est pas satisfaisable.

## Une méthode pour décider la satisfaisabilité d'une formule $p$

1. Calculer  $\mathcal{V}(p)$
2. Engendrer l'ensemble  $A$  des affectations  $v$  avec  $\text{supp}(v) \subseteq \mathcal{V}(p)$
3. Évaluer  $\llbracket p \rrbracket v$  pour toute affectation  $v \in A$ . Dès qu'on tombe sur un  $v$  tel que  $\llbracket p \rrbracket v = 1$  on sait que  $p$  est satisfaisable, si on n'en trouve pas alors  $p$  n'est pas satisfaisable.

Combien d'affectations est-ce qu'il y a à tester, si la formule a  $n$  variables ?

## Décider la validité d'une formule $p$

1. Calculer  $\mathcal{V}(p)$

## Décider la validité d'une formule $p$

1. Calculer  $\mathcal{V}(p)$
2. Engendrer l'ensemble  $A$  des affectations  $v$  avec  $\text{supp}(v) \subseteq \mathcal{V}(p)$

## Décider la validité d'une formule $p$

1. Calculer  $\mathcal{V}(p)$
2. Engendrer l'ensemble  $A$  des affectations  $v$  avec  $\text{supp}(v) \subseteq \mathcal{V}(p)$
3. Évaluer  $\llbracket p \rrbracket v$  pour tout  $v \in A$ . Dès qu'on tombe sur un  $v$  tel que  $\llbracket p \rrbracket v = 0$  on sait que  $p$  n'est pas valide, si on n'en trouve pas alors  $p$  est valide.

# Tables de vérité

[illegible]

## Tables de vérité

$x_1$	$x_2$	$x_3$	$\neg x_2$	$(x_1 \wedge x_2)$	$(x_3 \wedge \neg x_2)$	$((x_1 \wedge x_2) \vee (x_3 \wedge \neg x_2))$
(1)	(2)	(3)	(4)	(5)	(6)	(7)
			$\neg(2)$	$(1) \wedge (2)$	$(3) \wedge (4)$	$(5) \vee (6)$
0	0	0	1	0	0	0
1	0	0				
0	1	0				
1	1	0				
0	0	1				
1	0	1				
0	1	1				
1	1	1				

pas valide



# Tables de vérité

$x_1$	$x_2$	$x_3$	$\neg x_2$	$(x_1 \wedge x_2)$	$(x_3 \wedge \neg x_2)$	$((x_1 \wedge x_2) \vee (x_3 \wedge \neg x_2))$
(1)	(2)	(3)	(4)	(5)	(6)	(7)
			$\neg(2)$	$(1) \wedge (2)$	$(3) \wedge (4)$	$(5) \vee (6)$
0	0	0	1			
1	0	0				
0	1	0				
1	1	0				
0	0	1				
1	0	1				
0	1	1				
1	1	1				

# Tables de vérité

$x_1$	$x_2$	$x_3$	$\neg x_2$	$(x_1 \wedge x_2)$	$(x_3 \wedge \neg x_2)$	$((x_1 \wedge x_2) \vee (x_3 \wedge \neg x_2))$
(1)	(2)	(3)	(4)	(5)	(6)	(7)
			$\neg(2)$	$(1) \wedge (2)$	$(3) \wedge (4)$	$(5) \vee (6)$
0	0	0	1	0	0	
1	0	0				
0	1	0				
1	1	0				
0	0	1				
1	0	1				
0	1	1				
1	1	1				

# Tables de vérité

$x_1$	$x_2$	$x_3$	$\neg x_2$	$(x_1 \wedge x_2)$	$(x_3 \wedge \neg x_2)$	$((x_1 \wedge x_2) \vee (x_3 \wedge \neg x_2))$
(1)	(2)	(3)	(4)	(5)	(6)	(7)
			$\neg(2)$	$(1) \wedge (2)$	$(3) \wedge (4)$	$(5) \vee (6)$
0	0	0	1	0	0	0
1	0	0				
0	1	0				
1	1	0				
0	0	1				
1	0	1				
0	1	1				
1	1	1				

# Tables de vérité

$x_1$	$x_2$	$x_3$	$\neg x_2$	$(x_1 \wedge x_2)$	$(x_3 \wedge \neg x_2)$	$((x_1 \wedge x_2) \vee (x_3 \wedge \neg x_2))$
(1)	(2)	(3)	(4)	(5)	(6)	(7)
			$\neg(2)$	$(1) \wedge (2)$	$(3) \wedge (4)$	$(5) \vee (6)$
0	0	0	1	0	0	0
1	0	0	1	0	0	0
0	1	0				
1	1	0				
0	0	1				
1	0	1				
0	1	1				
1	1	1				

# Tables de vérité

$x_1$	$x_2$	$x_3$	$\neg x_2$	$(x_1 \wedge x_2)$	$(x_3 \wedge \neg x_2)$	$((x_1 \wedge x_2) \vee (x_3 \wedge \neg x_2))$
(1)	(2)	(3)	(4)	(5)	(6)	(7)
			$\neg(2)$	$(1) \wedge (2)$	$(3) \wedge (4)$	$(5) \vee (6)$
0	0	0	1	0	0	0
1	0	0	1	0	0	0
0	1	0	0	0	0	0
1	1	0				
0	0	1				
1	0	1				
0	1	1				
1	1	1				

## Tables de vérité

$x_1$	$x_2$	$x_3$	$\neg x_2$	$(x_1 \wedge x_2)$	$(x_3 \wedge \neg x_2)$	$((x_1 \wedge x_2) \vee (x_3 \wedge \neg x_2))$
(1)	(2)	(3)	(4)	(5)	(6)	(7)
			$\neg(2)$	$(1) \wedge (2)$	$(3) \wedge (4)$	$(5) \vee (6)$
0	0	0	1	0	0	0
1	0	0	1	0	0	0
0	1	0	0	0	0	0
1	1	0	0	1	0	1
0	0	1				
1	0	1				
0	1	1				
1	1	1				

## Tables de vérité

$x_1$	$x_2$	$x_3$	$\neg x_2$	$(x_1 \wedge x_2)$	$(x_3 \wedge \neg x_2)$	$((x_1 \wedge x_2) \vee (x_3 \wedge \neg x_2))$
(1)	(2)	(3)	(4)	(5)	(6)	(7)
			$\neg(2)$	$(1) \wedge (2)$	$(3) \wedge (4)$	$(5) \vee (6)$
0	0	0	1	0	0	0
1	0	0	1	0	0	0
0	1	0	0	0	0	0
1	1	0	0	1	0	1
0	0	1	1	0	1	1
1	0	1	1	0	1	1
0	1	1	0	0	0	0
1	1	1	0	1	0	1

## Efficacité de notre algorithme

- ▶ Si la formule a  $n$  variables :  $2^n$  affectations possibles à essayer (dans le pire des cas)



## Efficacité de notre algorithme

- ▶ Si la formule a  $n$  variables :  $2^n$  affectations possibles à essayer (dans le pire des cas)
- ▶ Temps d'exécution **exponentiel**

## Efficacité de notre algorithme

- ▶ Si la formule a  $n$  variables :  $2^n$  affectations possibles à essayer (dans le pire des cas)
- ▶ Temps d'exécution **exponentiel**
- ▶ Acceptable seulement pour des petites formules.

## Efficacité de notre algorithme

- ▶ Si la formule a  $n$  variables :  $2^n$  affectations possibles à essayer (dans le pire des cas)
- ▶ Temps d'exécution **exponentiel**
- ▶ Acceptable seulement pour des petites formules.
- ▶ Comment faire pour des formules avec 10.000 variables ? Voir dans quelques semaines !

## Raccourcis

- On a le droit d'enchaîner des applications de l'opérateur  $\wedge$  :

$$(p_1 \wedge p_2 \wedge \dots \wedge p_n)$$

ainsi que de l'opérateur  $\vee$  :

$$(p_1 \vee p_2 \vee \dots \vee p_n)$$

## Raccourcis

- On a le droit d'enchaîner des applications de l'opérateur  $\wedge$  :

$$(p_1 \wedge p_2 \wedge \dots \wedge p_n)$$

ainsi que de l'opérateur  $\vee$  :

$$(p_1 \vee p_2 \vee \dots \vee p_n)$$

- On se permet d'omettre la paire de parenthèses qui est autour de la formule **entière**.

## Récupérer la syntaxe stricte

► Remplacer

$$(p_1 \wedge p_2 \wedge p_3 \wedge \dots \wedge p_n)$$

par

$$(p_1 \wedge (p_2 \wedge (p_3 \dots \wedge p_n) \dots))$$

et pareil pour les chaînes  $\vee$ .

## Récupérer la syntaxe stricte

- Remplacer

$$(p_1 \wedge p_2 \wedge p_3 \wedge \dots \wedge p_n)$$

par

$$(p_1 \wedge (p_2 \wedge (p_3 \dots \wedge p_n) \dots))$$

et pareil pour les chaînes  $\vee$ .

- Mettre une paire de parenthèses extérieures autour de la formule si nécessaire

## Exemple

$$(x_1 \wedge x_2 \wedge x_3) \vee y_1 \vee (z_1 \wedge z_2 \wedge z_3 \wedge z_4)$$



## Exemple

$$(x_1 \wedge x_2 \wedge x_3) \vee y_1 \vee (z_1 \wedge z_2 \wedge z_3 \wedge z_4)$$

s'écrit en syntaxe stricte comme

$$\left( (x_1 \wedge (x_2 \wedge x_3)) \vee \left( y_1 \vee \left( z_1 \wedge \left( z_2 \wedge \left( z_3 \wedge z_4 \right) \right) \right) \right) \right)$$

## Syntaxe stricte ou raccourcie ?

- ▶ On autorise la syntaxe raccourcie dans les exemples.

## Syntaxe stricte ou raccourcie ?

- ▶ On autorise la syntaxe raccourcie dans les exemples.
- ▶ Par contre, quand on vous demande de démontrer une propriété de la syntaxe (comme:  $|w|_{\lrcorner} = |w|_{\rceil}$  pour toute  $w \in Form$ ) c'est toujours la syntaxe stricte !