Logique

Introduction à Coq/Rocq

Jules Chouquet

UNIVERSITE D'ORLEANS

2025

Contexte

Où on en est : on sait écrire des formules et raisonner dessus pour montrer si elles sont vraies ou fausses.

mais certaines formules peuvent être très longues ou difficiles à prouver.

Contexte

Où on en est : on sait écrire des formules et raisonner dessus pour montrer si elles sont vraies ou fausses.

mais certaines formules peuvent être très longues ou difficiles à prouver.

$\rightarrow$ Que ce soit en déduction naturelle ou en version “rédigée”, comment vérifier qu'une preuve de 15 pages est correcte ?

Contexte

Où on en est : on sait écrire des formules et raisonner dessus pour montrer si elles sont vraies ou fausses.

mais certaines formules peuvent être très longues ou difficiles à prouver.

$\rightarrow$ Que ce soit en déduction naturelle ou en version “rédigée”, comment vérifier qu'une preuve de 15 pages est correcte ?
$\rightarrow$ Comment écrire sans se tromper une preuve avec un raisonnement par cas où il y a 1478 cas importants à vérifier ?

Déjà

Pourquoi on s'intéresse à ce genre de preuves, où est-ce qu'on peut les trouver ?

Déjà

Pourquoi on s'intéresse à ce genre de preuves, où est-ce qu'on peut les trouver ?

- Démontrer qu'un algorithme est correct, quand l'algorithme est compliqué.

Déjà

Pourquoi on s'intéresse à ce genre de preuves, où est-ce qu'on peut les trouver ?

- Démontrer qu'un algorithme est correct, quand l'algorithme est compliqué.
- Démontrer qu'un programme ne va pas planter en vol.

Déjà

Pourquoi on s'intéresse à ce genre de preuves, où est-ce qu'on peut les trouver ?

- Démontrer qu'un algorithme est correct, quand l'algorithme est compliqué.
- Démontrer qu'un programme ne va pas planter en vol.
- Démontrer qu'un programme fera bien ce qu'on attend de lui

Déjà

Pourquoi on s'intéresse à ce genre de preuves, où est-ce qu'on peut les trouver ?

- Démontrer qu'un algorithme est correct, quand l'algorithme est compliqué.
- Démontrer qu'un programme ne va pas planter en vol.
- Démontrer qu'un programme fera bien ce qu'on attend de lui
- Démontrer certains théorèmes mathématiques dont on ne peut pas faire des preuves plus simples

Déjà

Pourquoi on s'intéresse à ce genre de preuves, où est-ce qu'on peut les trouver ?

- Démontrer qu'un algorithme est correct, quand l'algorithme est compliqué.
- Démontrer qu'un programme ne va pas planter en vol.
- Démontrer qu'un programme fera bien ce qu'on attend de lui
- Démontrer certains théorèmes mathématiques dont on ne peut pas faire des preuves plus simples
- ...

Alors

Dans les années 80[1] se développe le logiciel Coq, qui est un **assistant de preuve** . Plusieurs avantages à faire ses preuves avec :

- S'il n'y a pas de message d'erreur, on est **sûr** qu'elles sont correctes
- Si elles sont hyper longues, on en est tout aussi sûr.
- Certaines parties peuvent être prouvées automatiquement.

En 2025, Coq change de nom

ROCQ

1. Première version : 1984

Dans ce cours

Je vais vous donner un aperçu du logiciel, et vous montrer comment il s'utilise.

Puis vous aurez trois séances de TP (avec un petit contrôle à la fin) pour travailler dessus.

Conseil

Installez-le chez vous pour l'essayer (il y a aussi une version en ligne) [2]

2. Infos sur la page du cours

Principe d'utilisation

On veut prouver qu'une formule[3] est **un théorème** .

3. De la logique propositionnelle ou du calcul des prédicats par exemple.

Principe d'utilisation

On veut prouver qu'une formule [3] est **un théorème** .

En calcul des propositions, on sait faire ça avec la déduction naturelle par exemple.

En Coq, ça y ressemble beaucoup :

- Il y a des règles qu'on peut essayer d'appliquer (il faut bien connaître leurs noms)
- On a un ensemble d'hypothèses que l'on peut utiliser dans la preuve.
- Si on veut prouver une formule qui est dans les hypothèses (axiome), ça termine la preuve.

3. De la logique propositionnelle ou du calcul des prédicats par exemple.

Mais

Quelques différences notables :

4. H sont les hypothèses, A la formule à prouver.
5. On lit donc la preuve de bas en haut pour la construire, comme d'habitude.

Mais

Quelques différences notables :

- Le locigiel est intéractif, donc on ne voit pas l'évolution de la preuve et des hypothèses *en entier*.

4. H sont les hypothèses, A la formule à prouver.
5. On lit donc la preuve de bas en haut pour la construire, comme d'habitude.

Mais

Quelques différences notables :

- Le locigiel est intéractif, donc on ne voit pas l'évolution de la preuve et des hypothèses *en entier*.
- Il faut imaginer qu'à tout moment, on se situe sur une ligne de preuve en déduction naturelle, de la forme $H_1, \ldots, H_n \vdash A$ [4].

4. H sont les hypothèses, A la formule à prouver.
5. On lit donc la preuve de bas en haut pour la construire, comme d'habitude.

Mais

Quelques différences notables :

- Le locigiel est intéractif, donc on ne voit pas l'évolution de la preuve et des hypothèses *en entier*.
- Il faut imaginer qu'à tout moment, on se situe sur une ligne de preuve en déduction naturelle, de la forme $H_1, \ldots, H_n \vdash A$[4].
- Quand on indique à Coq la règle que l'on veut utiliser pour prouver A[5], on se retrouve dans une nouvelle configuration, où les hypothèses et la formule à prouver peuvent avoir changé.

4. H sont les hypothèses, A la formule à prouver.
5. On lit donc la preuve de bas en haut pour la construire, comme d'habitude.

Exemple schématique

On veut prouver $A \rightarrow (A \vee B)$, par exemple.

Exemple schématique

On veut prouver $A \to (A \vee B)$, par exemple.

Ce qu'on écrit

Ce qui s'affiche

Hypothèses :

But : $A \to (A \vee B)$

$$\dfrac{\dfrac{\dfrac{}{A \vdash A}\,\text{ax}}{A \vdash A \vee B}\,\vee\text{i-g}}{\vdash A \to A \vee B}\,\to\text{i}$$

Exemple schématique

On veut prouver $A \to (A \vee B)$, par exemple.

Ce qu'on écrit

```
intro H.
```

Ce qui s'affiche

Hypothèses

H : A

But : $A \vee B$

$$\dfrac{\dfrac{\dfrac{}{A \vdash A}\,\text{ax}}{A \vdash A \vee B}\,\vee\text{i-g}}{\vdash A \to A \vee B}\,\to\text{i}$$

Exemple schématique

On veut prouver $A \to (A \vee B)$, par exemple.

Ce qu'on écrit

```
left.
```

Ce qui s'affiche

Hypothèses

H : A

But : A

$$\dfrac{\dfrac{\dfrac{}{A \vdash A}\,\text{ax}}{A \vdash A \vee B}\,\vee\text{i-g}}{\vdash A \to A \vee B}\,\to\text{i}$$

Exemple schématique

On veut prouver $A \to (A \vee B)$, par exemple.

Ce qu'on écrit

```
assumption.
```

Ce qui s'affiche

No more goals

$$\dfrac{\dfrac{\dfrac{}{A \ \vdash A}\,\text{ax}}{A \ \vdash A \vee B}\,\vee\text{i-g}}{\vdash A \to A \vee B}\,\to\text{i}$$

Différence importante avec la déduction naturelle

On a le droit d'utiliser des règles pour modifier les hypothèses. Exemple avec $(A \wedge B) \to A$.

Différence importante avec la déduction naturelle

On a le droit d'utiliser des règles pour modifier les hypothèses. Exemple avec $(A \wedge B) \to A$.

Ce qu'on écrit

Ce qui s'affiche

Hypothèses :

But : $(A \wedge B) \to A$

$$\dfrac{\dfrac{\dfrac{}{A \wedge B \vdash A \wedge B}\text{ax}}{A \wedge B \ \vdash A}\wedge\text{e-g}}{\vdash (A \wedge B) \to A}\to\text{i}$$

Différence importante avec la déduction naturelle

On a le droit d'utiliser des règles pour modifier les hypothèses. Exemple avec $(A \land B) \to A$.

Ce qu'on écrit

```
intro H.
```

Ce qui s'affiche

Hypothèses :

H : $A \land B$

But : A

$$\dfrac{\dfrac{\dfrac{}{A \land B \vdash A \land B}\ \text{ax}}{A \land B \ \vdash A}\ \land\text{e-g}}{\vdash (A \land B) \to A}\ \to\text{i}$$

On a le droit d'utiliser des règles pour modifier les hypothèses. Exemple avec $(A \wedge B) \to A$.

Ce qu'on écrit

```
destruct H as
(H1,H2).
```

Ce qui s'affiche

Hypothèses :
H1 : A
H2 : B

But : A

$$\cfrac{\cfrac{\cfrac{}{\cancel{A \wedge B \vdash A \wedge B}}\ \cancel{\text{ax}}}{A \wedge B \ \vdash A}\ \cancel{\wedge\text{e-g}}}{\vdash (A \wedge B) \to A}\ \to\text{i}$$

Différence importante avec la déduction naturelle

On a le droit d'utiliser des règles pour modifier les hypothèses. Exemple avec $(A \land B) \to A$.

Ce qu'on écrit

```
assumption.
```

Ce qui s'affiche

No more goals

$$\dfrac{\dfrac{\dfrac{}{\cancel{A \land B \vdash A \land B}}\ \cancel{\text{ax}}}{A \land B \ \vdash A}\ \cancel{\land\text{e-g}}}{\vdash (A \land B) \to A}\ \to\text{i}$$

Différence importante avec la déduction naturelle

On a le droit d'utiliser des règles pour modifier les hypothèses. Exemple avec $(A \wedge B) \to A$.

Ce qu'on écrit

```
assumption.
```

Ce qui s'affiche

No more goals

$$\dfrac{\dfrac{\dfrac{}{\cancel{A \wedge B \vdash A \wedge B}}\ \cancel{\text{ax}}}{A \wedge B \ \vdash A}\ \cancel{\wedge\text{e-g}}}{\vdash (A \wedge B) \to A}\ \to\text{i}$$

Attention

Il faut connaître les **tactiques** de Coq pour écrire des preuves ! [a]

---

a. Certaines, comme `intro` ressemble à la ded. nat. D'autres comme `destruct` sont un peu différentes.

Branches

Que se passe-t-il quand on doit prouver plusieurs choses en même temps ?
(Exemple avec $A \rightarrow (A \wedge A)$)

Branches

Que se passe-t-il quand on doit prouver plusieurs choses en même temps ?
(Exemple avec $A \to (A \wedge A)$)

Ce qu'on écrit

Ce qui s'affiche

[un but]
Hypothèses :

But : $A \to (A \wedge A)$

$$\dfrac{\dfrac{\dfrac{}{A \vdash A}\,\text{ax} \qquad \dfrac{}{A \vdash A}\,\text{ax}}{A \vdash A \wedge A}\,\wedge\text{i}}{\vdash A \to (A \wedge A)}\,\to\text{i}$$

Branches

Que se passe-t-il quand on doit prouver plusieurs choses en même temps ?
(Exemple avec $A \to (A \wedge A)$)

Ce qu'on écrit

```
intro H.
```

Ce qui s'affiche

[un but]
Hypothèses :
H : A

But : $A \wedge A$

$$\dfrac{\dfrac{\dfrac{}{A \vdash A}\,\text{ax} \qquad \dfrac{}{A \vdash A}\,\text{ax}}{A \vdash A \wedge A}\,\wedge\text{i}}{\vdash A \to (A \wedge A)}\,\to\text{i}$$

Branches

Que se passe-t-il quand on doit prouver plusieurs choses en même temps ?
(Exemple avec $A \to (A \wedge A)$)

Ce qu'on écrit

```
split.
```

Ce qui s'affiche

[deux buts]

Hypothèses :

H : A

But : A

Le but n.2 sera : A

$$\dfrac{\dfrac{\dfrac{}{A \vdash A}\,\text{ax} \qquad \dfrac{}{A \vdash A}\,\text{ax}}{A \vdash A \wedge A}\,\wedge\text{i}}{\vdash A \to (A \wedge A)}\,\to\text{i}$$

Branches

Que se passe-t-il quand on doit prouver plusieurs choses en même temps ? (Exemple avec $A \to (A \wedge A)$)

Ce qu'on écrit

```
assumption.
```

Ce qui s'affiche

[un but]

Hypothèses :

H : A

But : A

$$\dfrac{\dfrac{}{A \vdash A}\,\text{ax} \qquad \dfrac{}{A \vdash A}\,\text{ax}}{\dfrac{A \vdash A \wedge A}{\vdash A \to (A \wedge A)}\,{\to}\text{i}}\,{\wedge}\text{i}$$

Branches

Que se passe-t-il quand on doit prouver plusieurs choses en même temps ?
(Exemple avec $A \to (A \wedge A)$)

Ce qu'on écrit

```
assumption.
```

Ce qui s'affiche

No more goals

$$\dfrac{\dfrac{\dfrac{}{A \vdash A}\,\text{ax} \qquad \dfrac{}{A \vdash A}\,\text{ax}}{A \vdash A \wedge A}\,\wedge\text{i}}{\vdash A \to (A \wedge A)}\,\to\text{i}$$

Branches

Que se passe-t-il quand on doit prouver plusieurs choses en même temps ? (Exemple avec $A \to (A \land A)$)

Ce qu'on écrit

```
assumption.
```

Ce qui s'affiche

No more goals

$$\dfrac{\dfrac{}{A \vdash A}\,\text{ax} \qquad \dfrac{}{A \vdash A}\,\text{ax}}{\dfrac{A \vdash A \land A}{\vdash A \to (A \land A)}\,\to\text{i}}\,\land\text{i}$$

Remarque

Quand la preuve est la même pour différentes branches, on peut faire en sorte de ne l'écrire qu'une fois, c'est l'un des avantages de Coq.

Branches

Que se passe-t-il quand on doit prouver plusieurs choses en même temps ? (Exemple avec $A \to (A \wedge A)$)

Ce qu'on écrit

```
assumption.
```

Ce qui s'affiche

No more goals

$$\dfrac{\dfrac{\dfrac{}{A \vdash A}\,\text{ax} \qquad \dfrac{}{A \vdash A}\,\text{ax}}{A \vdash A \wedge A}\,\wedge\text{i}}{\vdash A \to (A \wedge A)}\,\to\text{i}$$

Remarque

Quand la preuve est la même pour différentes branches, on peut faire en sorte de ne l'écrire qu'une fois, c'est l'un des avantages de Coq.

Attention

Parfois, dans les différents buts, les hypothèses ne seront pas les mêmes (pensez au raisonnement par cas).

Regardons Coq d'un peu plus près

[demo sur jsCoq]

Quelques tactiques à connaître

Introduire les hypothèses (et donc changer le but) :

Quelques tactiques à connaître

Introduire les hypothèses (et donc changer le but) :

- `intro H.`
- `intros Ha Hb Hc....`

Quelques tactiques à connaître

Introduire les hypothèses (et donc changer le but) :

- `intro H.`
- `intros Ha Hb Hc....`

Transformer un objectif $A \vee B$ en objectif A ou en objectif B :

Quelques tactiques à connaître

Introduire les hypothèses (et donc changer le but) :

- `intro H.`
- `intros Ha Hb Hc....`

Transformer un objectif $A \vee B$ en objectif A ou en objectif B :

- `left`
- `right`

Quelques tactiques à connaître

Introduire les hypothèses (et donc changer le but) :

- `intro H.`
- `intros Ha Hb Hc....`

Transformer un objectif $A \vee B$ en objectif A ou en objectif B :

- `left`
- `right`

Transformer un but $A \wedge B$ en deux buts : A, et B :

Quelques tactiques à connaître

Introduire les hypothèses (et donc changer le but) :

- `intro H.`
- `intros Ha Hb Hc....`

Transformer un objectif $A \vee B$ en objectif A ou en objectif B :

- `left`
- `right`

Transformer un but $A \wedge B$ en deux buts : A, et B :

- `split`

Quelques tactiques à connaître

Introduire les hypothèses (et donc changer le but) :

- `intro H.`
- `intros Ha Hb Hc....`

Transformer un objectif $A \vee B$ en objectif A ou en objectif B :

- `left`
- `right`

Transformer un but $A \wedge B$ en deux buts : A, et B :

- `split`

Transformer une hypothèse $H : A \wedge B$ en deux hypothèses :

Quelques tactiques à connaître

Introduire les hypothèses (et donc changer le but) :

- `intro H.`
- `intros Ha Hb Hc....`

Transformer un objectif $A \lor B$ en objectif A ou en objectif B :

- `left`
- `right`

Transformer un but $A \land B$ en deux buts : A, et B :

- `split`

Transformer une hypothèse $H : A \land B$ en deux hypothèses :

- `destruct H as (Ha,Hb)`

Le *Modus Ponens* en Coq

Élimination de l'implication

Comment on utilise une hypothèse `H` : $A \to B$?

Comment on utilise une hypothèse `H` : $A \to B$?
$\to$ Pour cela, il faut regarder quel est notre objectif. Si l'on veut que notre hypothèse serve à quelque chose, il faut que le but actuel soit B, sinon c'est impossible.

Comment on utilise une hypothèse `H` : $A \to B$?

$\to$ Pour cela, il faut regarder quel est notre objectif. Si l'on veut que notre hypothèse serve à quelque chose, il faut que le but actuel soit B, sinon c'est impossible.

Si on a `H` : $A \to B$ et qu'on veut montrer B ?

On utilise la tactique `apply H`.

Comment on utilise une hypothèse `H` : $A \to B$?
$\to$ Pour cela, il faut regarder quel est notre objectif. Si l'on veut que notre hypothèse serve à quelque chose, il faut que le but actuel soit B, sinon c'est impossible.

Si on a `H` : $A \to B$ et qu'on veut montrer B ?

On utilise la tactique `apply H`.
Et alors, seul l'objectif change, maintenent le nouveau but est A.

Comment on utilise une hypothèse `H` : $A \rightarrow B$?
$\rightarrow$ Pour cela, il faut regarder quel est notre objectif. Si l'on veut que notre hypothèse serve à quelque chose, il faut que le but actuel soit B, sinon c'est impossible.

Si on a `H` : $A \rightarrow B$ et qu'on veut montrer B ?

On utilise la tactique `apply H`.
Et alors, seul l'objectif change, maintenent le nouveau but est A.

C'est presque comme en déduction naturelle : Pour prouver B, si on connaît $A \rightarrow B$, il reste encore à prouver A !

La négation en Coq

$\neg A$ s'écrit `~A`

Rappelez-vous que la formule $\neg A$ peut être vue comme un raccourci pour $A \to \bot$.

La négation en Coq

$\neg A$ s'écrit `~A`

Rappelez-vous que la formule $\neg A$ peut être vue comme un raccourci pour $A \to \bot$.

En Coq, c'est exactement la même chose. Si le but est ~A, alors `intro` mettra le nouveau but à `False`, avec A comme nouvelle hypothèse.

La négation en Coq

$\neg A$ s'écrit `~A`

Rappelez-vous que la formule $\neg A$ peut être vue comme un raccourci pour $A \to \bot$.

En Coq, c'est exactement la même chose. Si le but est `~`A, alors `intro` mettra le nouveau but à `False`, avec A comme nouvelle hypothèse.

Et si vous avez un but `False`, vous pouvez donc faire `apply H` si vous avez une hypothèse `H` de la forme `~`A.

`False` et l'absurde

`False` est l'équivalent de $\bot$

On peut, comme en déduction naturelle, prouver n'importe quelle formule à partir d'une contradiction. Cette règle était, pour toute formule A :

$$\frac{\Gamma \vdash \bot}{\Gamma \vdash A}\ \bot$$

False et l'absurde

False est l'équivalent de $\perp$

On peut, comme en déduction naturelle, prouver n'importe quelle formule à partir d'une contradiction. Cette règle était, pour toute formule A :

$$\frac{\Gamma \vdash \perp}{\Gamma \vdash A} \perp$$

En Coq, elle permet de faire la même chose, mais se présente un peu différemment :

La tactique `contradiction`

Si dans mon ensemble d'hypothèses, j'ai `H: False`, alors je peux utiliser `contradiction`.

False et l'absurde

False est l'équivalent de $\perp$

On peut, comme en déduction naturelle, prouver n'importe quelle formule à partir d'une contradiction. Cette règle était, pour toute formule A :

$$\frac{\Gamma \vdash \perp}{\Gamma \vdash A} \perp$$

En Coq, elle permet de faire la même chose, mais se présente un peu différemment :

La tactique `contradiction`

Si dans mon ensemble d'hypothèses, j'ai `H: False`, alors je peux utiliser `contradiction`.
Que se passe-t-il alors : le but est immédiatement démontré !

`False` et l'absurde

`False` est l'équivalent de $\perp$

On peut, comme en déduction naturelle, prouver n'importe quelle formule à partir d'une contradiction. Cette règle était, pour toute formule A :

$$\frac{\Gamma \vdash \bot}{\Gamma \vdash A}\,\bot$$

En Coq, elle permet de faire la même chose, mais se présente un peu différemment :

La tactique `contradiction`

Si dans mon ensemble d'hypothèses, j'ai `H: False`, alors je peux utiliser `contradiction`.
Que se passe-t-il alors : le but est immédiatement démontré !

c'est un peu comme si au lieu de la règle, on avait un nouvel axiome

$$\frac{}{\Gamma, \bot \vdash A}\,\texttt{contr.}$$

False et l'absurde II

Coq est basé sur la logique intuitionniste.

`False` et l'absurde II

Coq est basé sur la logique intuitionniste.

Qu'est-ce que ça peut bien nous faire ?

Il est **impossible** d'utiliser **le raisonnement par l'absurde** !

Il n'y a pas de tactique équivalente à cette règle.

`False` et l'absurde II

Coq est basé sur la logique intuitionniste.

Qu'est-ce que ça peut bien nous faire ?

Il est **impossible** d'utiliser **le raisonnement par l'absurde** !

Il n'y a pas de tactique équivalente à cette règle.

Il est impossible de montrer $A \vee \neg A$, ou $\neg\neg A \to A$, par exemple.

Le raisonnement par cas en Coq

À nouveau, légèrement différent de la déduction naturelle, mais c'est le même principe : Si on sait que $A \vee B$, et que l'on est capable de montrer C depuis A **et** depuis B, alors on peut en déduire C.

Le raisonnement par cas en Coq

À nouveau, légèrement différent de la déduction naturelle, mais c'est le même principe : Si on sait que $A \vee B$, et que l'on est capable de montrer C depuis A **et** depuis B, alors on peut en déduire C.

En Coq : Si j'ai dans mes hypothèses `H` : $A \vee B$, et que mon but actuel est C, alors je peux utiliser la tactique "`case H`".

`case H` : que se passe-t-il ?

Deux objectifs sont créés :

- $A \rightarrow C$
- $B \rightarrow C$

(C'est donc bien à peu près la même chose qu'en déduction naturelle).

Pour finir

Quelques mots et démonstrations sur

- les nombres
- les mathématiques
- la récurrence

en Coq