Calcul des constructions

David Delahaye

David.Delahaye@lirmm.fr

Université de Montpellier
Faculté des Sciences

Licence Informatique L3 2024-2025

Quel cadre logique pour vérifier les programmes ?

Où en sommes-nous ?

- Logique du premier ordre (classique et intuitionniste).
- Systèmes de preuves formelles (LK, LJ).
- Raisonnement avec l'égalité ($\mathrm{LK}_{\mathrm{EQ}}$, $\mathrm{LJ}_{\mathrm{EQ}}$).

Il nous faut plus de théories

- Une théorie générale : théorie des ensembles.
- Permet de tout encoder (fonctions, entiers, etc.).

Quel cadre logique pour vérifier les programmes ?

Où en sommes-nous ?

- Logique du premier ordre (classique et intuitionniste).
- Systèmes de preuves formelles (LK, LJ).
- Raisonnement avec l'égalité (LK_{EQ}, LJ_{EQ}).

Il nous faut plus de théories

- Une théorie générale : théorie des ensembles.
- Permet de tout encoder (fonctions, entiers, etc.).

Théorie « naïve » des ensembles

Un peu d'histoire

- Fin du 19ème siecle, volonté de formaliser certaines théories.
- Par exemple : les ensembles, la géométrie, etc.
- Théorie « naïve » de Georg Cantor en 1878.
- Théorie incohérente : paradoxe de Bertrand Russell en 1902.
  - $R = \{x \mid x \notin x\}$, deux cas :
    - Soit $R \in R$, donc par définition de R, $R \notin R$.
    - Soit $R \notin R$, donc par définition de R, $R \in R$.
  - Le problème vient de l'axiome de compréhension :
    - $\exists x.\forall y.y \in x \Leftrightarrow P(y)$.
    - R se définit en prenant $P(x) = x \notin x$.

Théorie des ensembles et théorie des types

Remédier au paradoxe de Russell

- Deux possibilités qui ont donné deux théories.
- Théorie des ensembles de Zermelo-Fraenkel :
  - Introduite par Ernst Zermelo en 1908, étendue plus tard par Abraham Fraenkel en 1922.
  - Restriction de l'axiome de compréhension : $\forall z.\exists x.\forall y.y \in x \Leftrightarrow y \in z \wedge P(y)$.
- Théorie des types de Whitehead-Russell :
  - Introduite par Alfred Whitehead et Bertrand Russell en 1911.
  - On ne restreint pas l'axiome de compréhension.
  - On type les expressions de manière à éviter $x \notin x$ ou $x \in x$.
- Alternatives : théorie des ensembles de von Neumann-Bernays-Gödel, appelée aussi théorie des classes, introduite en 1925.

Théorie des ensembles et théorie des types

Remédier au paradoxe de Russell

- Deux possibilités qui ont donné deux théories.
- Théorie des ensembles de Zermelo-Fraenkel :
  - Introduite par Ernst Zermelo en 1908, étendue plus tard par Abraham Fraenkel en 1922.
  - Restriction de l'axiome de compréhension : $\forall z.\exists x.\forall y.y \in x \Leftrightarrow y \in z \wedge P(y)$.
- Théorie des types de Whitehead-Russell :
  - Introduite par Alfred Whitehead et Bertrand Russell en 1911.
  - On ne restreint pas l'axiome de compréhension.
  - On type les expressions de manière à éviter $x \notin x$ ou $x \in x$.
- Alternatives : théorie des ensembles de von Neumann-Bernays-Gödel, appelée aussi théorie des classes, introduite en 1925.

Théorie des ensembles et théorie des types

Remédier au paradoxe de Russell

- Deux possibilités qui ont donné deux théories.
- Théorie des ensembles de Zermelo-Fraenkel :
  - Introduite par Ernst Zermelo en 1908, étendue plus tard par Abraham Fraenkel en 1922.
  - Restriction de l'axiome de compréhension : $\forall z.\exists x.\forall y.y \in x \Leftrightarrow y \in z \wedge P(y)$.
- Théorie des types de Whitehead-Russell :
  - Introduite par Alfred Whitehead et Bertrand Russell en 1911.
  - On ne restreint pas l'axiome de compréhension.
  - On type les expressions de manière à éviter $x \notin x$ ou $x \in x$.
- Alternatives : théorie des ensembles de von Neumann-Bernays-Gödel, appelée aussi théorie des classes, introduite en 1925.

Théorie des ensembles et théorie des types

Remédier au paradoxe de Russell

- Deux possibilités qui ont donné deux théories.
- Théorie des ensembles de Zermelo-Fraenkel :
  - Introduite par Ernst Zermelo en 1908, étendue plus tard par Abraham Fraenkel en 1922.
  - Restriction de l'axiome de compréhension : $\forall z.\exists x.\forall y.y \in x \Leftrightarrow y \in z \wedge P(y)$.
- Théorie des types de Whitehead-Russell :
  - Introduite par Alfred Whitehead et Bertrand Russell en 1911.
  - On ne restreint pas l'axiome de compréhension.
  - On type les expressions de manière à éviter $x \notin x$ ou $x \in x$.
- Alternatives : théorie des ensembles de von Neumann-Bernays-Gödel, appelée aussi théorie des classes, introduite en 1925.

Théorie des ensembles ou théorie des types ?

Que choisir ?

- Inconvénients de la théorie des ensembles :
  - Statut des fonctions :
    - Les fonctions ne sont pas primitives (ensembles de couples).
      On doit introduire des notations (lieurs, application, domaines de définitions, etc.).
    - On ne peut pas calculer, tout repose sur le raisonnement.
      Pour démontrer la validité de $2 + 2 = 4$, on doit utiliser les axiomes :

$$\begin{cases} \forall y.o + y = y \\ \forall x, y.s(x) + y = s(x + y) \end{cases}$$

  - Statut des démonstrations :
    Les démonstrations sont à part dans la théorie.
    Ce sont des objets périphériques et non des objets du langage.
    Ainsi, on peut démontrer que 5 est premier mais on ne peut pas exprimer qu'un certain objet est une démonstration de cela.

Théorie des ensembles ou théorie des types ?

Que choisir ?

- Inconvénients de la théorie des ensembles :
  - Statut des fonctions :
    - Les fonctions ne sont pas primitives (ensembles de couples). On doit introduire des notations (lieurs, application, domaines de définitions, etc.).
    - On ne peut pas calculer, tout repose sur le raisonnement. Pour démontrer la validité de $2+2=4$, on doit utiliser les axiomes :

$$\left\{\begin{array}{l}\forall y.o+y=y\\ \forall x,y.s(x)+y=s(x+y)\end{array}\right.$$

  - Statut des démonstrations :
    Les démonstrations sont à part dans la théorie.
    Ce sont des objets périphériques et non des objets du langage.
    Ainsi, on peut démontrer que 5 est premier mais on ne peut pas exprimer qu'un certain objet est une démonstration de cela.

Théorie des ensembles ou théorie des types ?

Que choisir ?

- Inconvénients de la théorie des ensembles :
  - Statut des fonctions :
    - Les fonctions ne sont pas primitives (ensembles de couples). On doit introduire des notations (lieurs, application, domaines de définitions, etc.).
    - On ne peut pas calculer, tout repose sur le raisonnement. Pour démontrer la validité de $2+2=4$, on doit utiliser les axiomes :

$$\begin{cases} \forall y.o + y = y \\ \forall x, y.s(x) + y = s(x+y) \end{cases}$$

  - Statut des démonstrations : Les démonstrations sont à part dans la théorie. Ce sont des objets périphériques et non des objets du langage. Ainsi, on peut démontrer que 5 est premier mais on ne peut pas exprimer qu'un certain objet est une démonstration de cela.

Théorie des ensembles ou théorie des types ?

Que choisir ?

- Inconvénients de la théorie des ensembles :
  - Statut des fonctions :
    - Les fonctions ne sont pas primitives (ensembles de couples). On doit introduire des notations (lieurs, application, domaines de définitions, etc.).
    - On ne peut pas calculer, tout repose sur le raisonnement. Pour démontrer la validité de $2 + 2 = 4$, on doit utiliser les axiomes :

$$\begin{cases} \forall y.o + y = y \\ \forall x, y.s(x) + y = s(x + y) \end{cases}$$

  - Statut des démonstrations :
    Les démonstrations sont à part dans la théorie.
    Ce sont des objets périphériques et non des objets du langage.
    Ainsi, on peut démontrer que 5 est premier mais on ne peut pas exprimer qu'un certain objet est une démonstration de cela.

Théorie des ensembles ou théorie des types ?

Que choisir ?

- Inconvénients de la théorie des ensembles :
  - Statut des fonctions :
    - Les fonctions ne sont pas primitives (ensembles de couples).
      On doit introduire des notations (lieurs, application, domaines de définitions, etc.).
    - On ne peut pas calculer, tout repose sur le raisonnement.
      Pour démontrer la validité de $2 + 2 = 4$, on doit utiliser les axiomes :

$$\left\{ \begin{array}{l} \forall y.o + y = y \\ \forall x, y.s(x) + y = s(x + y) \end{array} \right.$$

  - Statut des démonstrations :
    Les démonstrations sont à part dans la théorie.
    Ce sont des objets périphériques et non des objets du langage.
    Ainsi, on peut démontrer que 5 est premier mais on ne peut pas exprimer qu'un certain objet est une démonstration de cela.

Théorie des ensembles ou théorie des types ?

Que choisir ?

- Inconvénients de la théorie des ensembles :
  - Statut des fonctions :
    - Les fonctions ne sont pas primitives (ensembles de couples).
      On doit introduire des notations (lieurs, application, domaines de définitions, etc.).
    - On ne peut pas calculer, tout repose sur le raisonnement.
      Pour démontrer la validité de $2 + 2 = 4$, on doit utiliser les axiomes :

$$\begin{cases} \forall y.o + y = y \\ \forall x, y.s(x) + y = s(x + y) \end{cases}$$

  - Statut des démonstrations :
    Les démonstrations sont à part dans la théorie.
    Ce sont des objets périphériques et non des objets du langage.
    Ainsi, on peut démontrer que 5 est premier mais on ne peut pas exprimer qu'un certain objet est une démonstration de cela.

Théorie des ensembles ou théorie des types ?

Que choisir ?

- Introduire du calcul dans le langage.
  - On souhaite avoir une représentation intentionnelle des fonctions :
    - Les fonctions sont des objets du langage (contrairement à la théorie des ensembles où elles sont un cas particulier de prédicats).
    - Une fonction peut être définie par un algorithme la calculant.
  - Théorie des types simples de Church en 1940, logique d'ordre supérieur basée sur le λ-calcul typé.

Théorie des ensembles ou théorie des types ?

Que choisir ?

- Introduire du calcul dans le langage.
  - On souhaite avoir une représentation intentionnelle des fonctions :
    - Les fonctions sont des objets du langage (contrairement à la théorie des ensembles où elles sont un cas particulier de prédicats).
    - Une fonction peut être définie par un algorithme la calculant.
  - Théorie des types simples de Church en 1940, logique d'ordre supérieur basée sur le λ-calcul typé.

Théorie des ensembles ou théorie des types ?

Que choisir ?

- Introduire du calcul dans le langage.
  - On souhaite avoir une représentation intentionnelle des fonctions :
    - Les fonctions sont des objets du langage (contrairement à la théorie des ensembles où elles sont un cas particulier de prédicats).
    - Une fonction peut être définie par un algorithme la calculant.
  - Théorie des types simples de Church en 1940, logique d'ordre supérieur basée sur le λ-calcul typé.

Théorie des ensembles ou théorie des types ?

Que choisir ?

- Introduire du calcul dans le langage.
  - On souhaite avoir une représentation intentionnelle des fonctions :
    - Les fonctions sont des objets du langage (contrairement à la théorie des ensembles où elles sont un cas particulier de prédicats).
    - Une fonction peut être définie par un algorithme la calculant.
  - Théorie des types simples de Church en 1940, logique d'ordre supérieur basée sur le λ-calcul typé.

Théorie des ensembles ou théorie des types ?

Que choisir ?

- Les démonstrations comme objets du langage.
  - Interprétation de Brouwer-Heyting-Kolmogorov en logique intuitionniste : les démonstrations sont vues comme des fonctions.
  - Isomorphisme (correspondance) de Curry-Howard :
    - Proposition = type et preuve = fonction.
    - Haskell Curry en 1958, Nicolaas de Bruijn en 1968, William Alvin Howard en 1969.

Théorie des ensembles ou théorie des types ?

Que choisir ?

- Les démonstrations comme objets du langage.
  - Interprétation de Brouwer-Heyting-Kolmogorov en logique intuitionniste : les démonstrations sont vues comme des fonctions.
  - Isomorphisme (correspondance) de Curry-Howard :
    - Proposition = type et preuve = fonction.
    - Haskell Curry en 1958, Nicolaas de Bruijn en 1968, William Alvin Howard en 1969.

Théorie des ensembles ou théorie des types ?

Que choisir ?

- Les démonstrations comme objets du langage.
  - Interprétation de Brouwer-Heyting-Kolmogorov en logique intuitionniste : les démonstrations sont vues comme des fonctions.
  - Isomorphisme (correspondance) de Curry-Howard :
    - Proposition = type et preuve = fonction.
    - Haskell Curry en 1958, Nicolaas de Bruijn en 1968, William Alvin Howard en 1969.

Théorie des ensembles ou théorie des types ?

Que choisir ?

- Calcul des constructions (CC) :
  - Introduit par Thierry Coquand et Gérard Huet en 1985.
  - Basé sur un λ-calcul typé et sur l'isomorphisme de Curry-Howard.
  - Polymorphisme, types dépendants, constructions.
- On aura aussi besoin de l'induction :
  - Calcul des constructions inductives (prochain cours).
  - Introduit par Thierry Coquand et Christine Paulin-Mohring en 1989.
  - Ajout des types inductifs primitifs.
  - Coq est basé sur ce calcul.

Théorie des ensembles ou théorie des types ?

Que choisir ?

- Calcul des constructions (CC) :
  - Introduit par Thierry Coquand et Gérard Huet en 1985.
  - Basé sur un λ-calcul typé et sur l'isomorphisme de Curry-Howard.
  - Polymorphisme, types dépendants, constructions.
- On aura aussi besoin de l'induction :
  - Calcul des constructions inductives (prochain cours).
  - Introduit par Thierry Coquand et Christine Paulin-Mohring en 1989.
  - Ajout des types inductifs primitifs.
  - Coq est basé sur ce calcul.

λ-calcul de Church

Syntaxe (abstraite)

- On fixe un ensemble de variables $x, y, \ldots \in \mathbb{V}$;
- L'ensemble des termes $\mathcal{T}$ est le plus petit ensemble tel que :
  - Si $x \in \mathbb{V}$ alors $x \in \mathcal{T}$ (variable) ;
  - Si $x \in \mathbb{V}$ et $M \in \mathcal{T}$ alors $\lambda x.M \in \mathcal{T}$ (fonction ou λ-terme) ;
  - Si $M, N \in \mathcal{T}$ alors $M\ N \in \mathcal{T}$ (application).

λ-calcul de Church

Notation pointée

- La portée d'un lieur (λ) va jusqu'à la parenthèse fermante du terme du lieur ;
- Si le terme du lieur n'est pas parenthésé, la portée du lieur va jusqu'à la fin du terme ;
- Donc, si on veut arrêter la portée d'un lieur, il suffit d'utiliser des parenthèses pour limiter explicitement la portée du lieur ;
- Exemples :
  - $\lambda x.f\ x\ y \equiv \lambda x.(f\ x\ y)$;
  - Si on veut que le λ ne porte que sur $f\ x$, on doit écrire : $(\lambda x.f\ x)\ y$.
- Notation : $\lambda x, y.M \equiv \lambda x.\lambda y.M$.

Associativité

- L'application associe à gauche :
  - $M\ N\ L \equiv ((M\ N)\ L)$.

λ-calcul de Church

Variables libres, variables liées

- Une variable x est libre dans un terme M si et seulement s'il existe une occurrence de x dans M qui n'est sous la portée d'un lieur ;
- Une variable x est liée dans un terme M si et seulement s'il existe une occurrence de x dans M qui est sous la portée d'un lieur ;
- Occurrence $\equiv$ position d'un terme dans un terme.

λ-calcul de Church

Variables libres (« Free Variables »)

- L'ensemble des variables libres d'un terme M, noté $FV(M)$, est défini par récurrence structurelle par :
  - Si $x \in \mathbb{V}$ alors $FV(x) = \{x\}$;
  - Si $x \in \mathbb{V}$ et $M \in \mathcal{T}$ alors $FV(\lambda x.M) = FV(M) - \{x\}$;
  - Si $M, N \in \mathcal{T}$ alors $FV(M\ N) = FV(M) \cup FV(N)$.

Variables liées (« Bound Variables »)

- L'ensemble des variables liées d'un terme M, noté $BV(M)$, est défini par récurrence structurelle par :
  - Si $x \in \mathbb{V}$ alors $BV(x) = \emptyset$;
  - Si $x \in \mathbb{V}$ et $M \in \mathcal{T}$ alors $BV(\lambda x.M) = BV(M) \cup \{x\}$;
  - Si $M, N \in \mathcal{T}$ alors $BV(M\ N) = BV(M) \cup BV(N)$.

λ-calcul de Church

Variables libres, variables liées : exemples

- y est libre dans $\lambda x.f\ x\ y$;
- x est liée dans $\lambda x.f\ x\ y$;
- Dans la formule $\lambda x.f\ x\ y$:
  - L'ensemble des variables libres est $\{f, y\}$;
  - L'ensemble des variables liées est $\{x\}$.
- Une variable peut être libre et liée à la fois (c'est-à-dire qu'elle possède une occurrence où elle est libre et une autre où elle est liée), par exemple : $(\lambda x.f\ x\ y)\ x$, où x est libre (deuxième occurrence) et liée (première occurrence) à la fois.

λ-calcul de Church

Substitution

- $M[L/y]$ est la substitution dans le terme M de toutes les occurrences libres de la variable y par le terme L ;
- Elle est définie par récurrence structurelle par :
  - Si $x \in \mathbb{V}$ alors $x[L/y] = \begin{cases} L, \text{ si } x = y \\ x, \text{ sinon} \end{cases}$
  - Si $x \in \mathbb{V}$ et $M \in \mathcal{T}$ alors $(\lambda x.M)[L/y] = \begin{cases} \lambda x.M, \text{ si } x = y \\ \lambda x.M[L/y], \text{ sinon} \end{cases}$
  - Si $M, N \in \mathcal{T}$ alors $(M\ N)[L/y] = M[L/y]\ N[L/y]$.

Exemples

- $(\lambda z.x)[y/x] = \lambda z.y$;
- $((\lambda x.f\ x)\ x\ z)[y/x] = (\lambda x.f\ x)\ y\ z$.

λ-calcul de Church

Substitution

- $M[L/y]$ est la substitution dans le terme M de toutes les occurrences libres de la variable y par le terme L ;
- Elle est définie par récurrence structurelle par :
  - Si $x \in \mathbb{V}$ alors $x[L/y] = \begin{cases} L, \text{ si } x = y \\ x, \text{ sinon} \end{cases}$
  - Si $x \in \mathbb{V}$ et $M \in \mathcal{T}$ alors $(\lambda x.M)[L/y] = \begin{cases} \lambda x.M, \text{ si } x = y \\ \lambda x.M[L/y], \text{ sinon} \end{cases}$
  - Si $M, N \in \mathcal{T}$ alors $(M\ N)[L/y] = M[L/y]\ N[L/y]$.

Exemples

- $(\lambda z.x)[y/x] = \lambda z.y$;
- $((\lambda x.f\ x)\ x\ z)[y/x] = (\lambda x.f\ x)\ y\ z$.

λ-calcul de Church

Substitution

- $M[L/y]$ est la substitution dans le terme M de toutes les occurrences libres de la variable y par le terme L ;
- Elle est définie par récurrence structurelle par :
  - Si $x \in \mathbb{V}$ alors $x[L/y] = \begin{cases} L, \text{ si } x = y \\ x, \text{ sinon} \end{cases}$
  - Si $x \in \mathbb{V}$ et $M \in \mathcal{T}$ alors $(\lambda x.M)[L/y] = \begin{cases} \lambda x.M, \text{ si } x = y \\ \lambda x.M[L/y], \text{ sinon} \end{cases}$
  - Si $M, N \in \mathcal{T}$ alors $(M\ N)[L/y] = M[L/y]\ N[L/y]$.

Exemples

- $(\lambda z.x)[y/x] = \lambda z.y$;
- $((\lambda x.f\ x)\ x\ z)[y/x] = (\lambda x.f\ x)\ y\ z$.

λ-calcul de Church

Substitution sûre

- Capture de variables :
  - $(\lambda y.x)[y/x] = \lambda y.y$: l'occurrence libre de y devient liée.
- La substitution $M[N/x]$ est dite sûre si $BV(M) \cap FV(N) = \emptyset$;
- On ne considérera que des substitutions sûres et on peut toujours se ramener à des substitutions sûres par α-renommage (par renommage des variables liées par des variables « fraîches » non libres).

λ-calcul de Church

Substitution sûre

- Capture de variables :
  - $(\lambda y.x)[y/x] = \lambda y.y$: l'occurrence libre de y devient liée.
- La substitution $M[N/x]$ est dite sûre si $BV(M) \cap FV(N) = \emptyset$;
- On ne considérera que des substitutions sûres et on peut toujours se ramener à des substitutions sûres par α-renommage (par renommage des variables liées par des variables « fraîches » non libres).

λ-calcul de Church

α-renommage

- $\lambda x.M \rightsquigarrow_\alpha \lambda y.M[y/x]$, si $y \notin FV(M)$;
- La relation $\rightarrow_\alpha$ est la plus petite relation contenant la relation $\rightsquigarrow_\alpha$ et qui passe au contexte :
  - $\lambda x.M \rightarrow_\alpha \lambda x.M'$, si $M \rightarrow_\alpha M'$;
  - $M\ N \rightarrow_\alpha M'\ N$, si $M \rightarrow_\alpha M'$;
  - $M\ N \rightarrow_\alpha M\ N'$, si $N \rightarrow_\alpha N'$.
- La relation $=_\alpha$ est la clôture réflexive, symétrique et transitive de la relation $\rightarrow_\alpha$;
- On dira que deux termes M et N sont α-équivalents si et seulement si on a $M =_\alpha N$;
- Dans la suite, on quotientera l'ensemble $\mathcal{T}$ des termes par la relation $=_\alpha$, c'est-à-dire que deux termes α-équivalents seront considérés comme identiques.

λ-calcul de Church

α-renommage

- $\lambda x.M \rightsquigarrow_\alpha \lambda y.M[y/x]$, si $y \notin FV(M)$;
- La relation $\rightarrow_\alpha$ est la plus petite relation contenant la relation $\rightsquigarrow_\alpha$ et qui passe au contexte :
  - $\lambda x.M \rightarrow_\alpha \lambda x.M'$, si $M \rightarrow_\alpha M'$;
  - $M\ N \rightarrow_\alpha M'\ N$, si $M \rightarrow_\alpha M'$;
  - $M\ N \rightarrow_\alpha M\ N'$, si $N \rightarrow_\alpha N'$.
- La relation $=_\alpha$ est la clôture réflexive, symétrique et transitive de la relation $\rightarrow_\alpha$;
- On dira que deux termes M et N sont α-équivalents si et seulement si on a $M =_\alpha N$;
- Dans la suite, on quotientera l'ensemble $\mathcal{T}$ des termes par la relation $=_\alpha$, c'est-à-dire que deux termes α-équivalents seront considérés comme identiques.

λ-calcul de Church

α-renommage

- $\lambda x.M \rightsquigarrow_\alpha \lambda y.M[y/x]$, si $y \notin FV(M)$;
- La relation $\rightarrow_\alpha$ est la plus petite relation contenant la relation $\rightsquigarrow_\alpha$ et qui passe au contexte :
  - $\lambda x.M \rightarrow_\alpha \lambda x.M'$, si $M \rightarrow_\alpha M'$;
  - $M\ N \rightarrow_\alpha M'\ N$, si $M \rightarrow_\alpha M'$;
  - $M\ N \rightarrow_\alpha M\ N'$, si $N \rightarrow_\alpha N'$.
- La relation $=_\alpha$ est la clôture réflexive, symétrique et transitive de la relation $\rightarrow_\alpha$;
- On dira que deux termes M et N sont α-équivalents si et seulement si on a $M =_\alpha N$;
- Dans la suite, on quotientera l'ensemble $\mathcal{T}$ des termes par la relation $=_\alpha$, c'est-à-dire que deux termes α-équivalents seront considérés comme identiques.

λ-calcul de Church

α-renommage

- $\lambda x.M \rightsquigarrow_\alpha \lambda y.M[y/x]$, si $y \notin FV(M)$;
- La relation $\rightarrow_\alpha$ est la plus petite relation contenant la relation $\rightsquigarrow_\alpha$ et qui passe au contexte :
  - $\lambda x.M \rightarrow_\alpha \lambda x.M'$, si $M \rightarrow_\alpha M'$;
  - $M\ N \rightarrow_\alpha M'\ N$, si $M \rightarrow_\alpha M'$;
  - $M\ N \rightarrow_\alpha M\ N'$, si $N \rightarrow_\alpha N'$.
- La relation $=_\alpha$ est la clôture réflexive, symétrique et transitive de la relation $\rightarrow_\alpha$;
- On dira que deux termes M et N sont α-équivalents si et seulement si on a $M =_\alpha N$;
- Dans la suite, on quotientera l'ensemble $\mathcal{T}$ des termes par la relation $=_\alpha$, c'est-à-dire que deux termes α-équivalents seront considérés comme identiques.

λ-calcul de Church

α-renommage

- $\lambda x.M \rightsquigarrow_\alpha \lambda y.M[y/x]$, si $y \notin FV(M)$;
- La relation $\rightarrow_\alpha$ est la plus petite relation contenant la relation $\rightsquigarrow_\alpha$ et qui passe au contexte :
  - $\lambda x.M \rightarrow_\alpha \lambda x.M'$, si $M \rightarrow_\alpha M'$;
  - $M\ N \rightarrow_\alpha M'\ N$, si $M \rightarrow_\alpha M'$;
  - $M\ N \rightarrow_\alpha M\ N'$, si $N \rightarrow_\alpha N'$.
- La relation $=_\alpha$ est la clôture réflexive, symétrique et transitive de la relation $\rightarrow_\alpha$;
- On dira que deux termes M et N sont α-équivalents si et seulement si on a $M =_\alpha N$;
- Dans la suite, on quotientera l'ensemble $\mathcal{T}$ des termes par la relation $=_\alpha$, c'est-à-dire que deux termes α-équivalents seront considérés comme identiques.

λ-calcul de Church

β-réduction

- $(\lambda x.M)\ N \rightsquigarrow_\beta M[N/x]$, si la substitution est sûre ;
- La relation $\rightarrow_\beta$ est la plus petite relation contenant la relation $\rightsquigarrow_\beta$ et qui passe au contexte ;
- La relation $\rightarrow_\beta^*$ est la clôture réflexive et transitive de la relation $\rightarrow_\beta$;
- La relation $=_\beta$ est la clôture symétrique de la relation $\rightarrow_\beta^*$.

Exemples

- $(\lambda x.x)\ y \rightarrow_\beta y$;
- $(\lambda x, y.f\ x\ y)\ t\ u \rightarrow_\beta^* f\ t\ u$;
- $(\lambda x.(\lambda y.f\ y)\ x)\ t \rightarrow_\beta^* f\ t$.

λ-calcul de Church

β-réduction

- $(\lambda x.M)\ N \rightsquigarrow_\beta M[N/x]$, si la substitution est sûre ;
- La relation $\rightarrow_\beta$ est la plus petite relation contenant la relation $\rightsquigarrow_\beta$ et qui passe au contexte ;
- La relation $\rightarrow_\beta^*$ est la clôture réflexive et transitive de la relation $\rightarrow_\beta$;
- La relation $=_\beta$ est la clôture symétrique de la relation $\rightarrow_\beta^*$.

Exemples

- $(\lambda x.x)\ y \rightarrow_\beta y$;
- $(\lambda x, y.f\ x\ y)\ t\ u \rightarrow_\beta^* f\ t\ u$;
- $(\lambda x.(\lambda y.f\ y)\ x)\ t \rightarrow_\beta^* f\ t$.

λ-calcul de Church

β-réduction

- $(\lambda x.M)\ N \rightsquigarrow_\beta M[N/x]$, si la substitution est sûre ;
- La relation $\rightarrow_\beta$ est la plus petite relation contenant la relation $\rightsquigarrow_\beta$ et qui passe au contexte ;
- La relation $\rightarrow^*_\beta$ est la clôture réflexive et transitive de la relation $\rightarrow_\beta$;
- La relation $=_\beta$ est la clôture symétrique de la relation $\rightarrow^*_\beta$.

Exemples

- $(\lambda x.x)\ y \rightarrow_\beta y$;
- $(\lambda x, y.f\ x\ y)\ t\ u \rightarrow^*_\beta f\ t\ u$;
- $(\lambda x.(\lambda y.f\ y)\ x)\ t \rightarrow^*_\beta f\ t$.

λ-calcul de Church

β-réduction

- $(\lambda x.M)\ N \rightsquigarrow_\beta M[N/x]$, si la substitution est sûre ;
- La relation $\rightarrow_\beta$ est la plus petite relation contenant la relation $\rightsquigarrow_\beta$ et qui passe au contexte ;
- La relation $\rightarrow^*_\beta$ est la clôture réflexive et transitive de la relation $\rightarrow_\beta$;
- La relation $=_\beta$ est la clôture symétrique de la relation $\rightarrow^*_\beta$.

Exemples

- $(\lambda x.x)\ y \rightarrow_\beta y$;
- $(\lambda x, y.f\ x\ y)\ t\ u \rightarrow^*_\beta f\ t\ u$;
- $(\lambda x.(\lambda y.f\ y)\ x)\ t \rightarrow^*_\beta f\ t$.

λ-calcul de Church

β-réduction

- $(\lambda x.M)\ N \rightsquigarrow_\beta M[N/x]$, si la substitution est sûre ;
- La relation $\rightarrow_\beta$ est la plus petite relation contenant la relation $\rightsquigarrow_\beta$ et qui passe au contexte ;
- La relation $\rightarrow_\beta^*$ est la clôture réflexive et transitive de la relation $\rightarrow_\beta$;
- La relation $=_\beta$ est la clôture symétrique de la relation $\rightarrow_\beta^*$.

Exemples

- $(\lambda x.x)\ y \rightarrow_\beta y$;
- $(\lambda x, y.f\ x\ y)\ t\ u \rightarrow_\beta^* f\ t\ u$;
- $(\lambda x.(\lambda y.f\ y)\ x)\ t \rightarrow_\beta^* f\ t$.

λ-calcul de Church

Rédex et forme normale

- Un rédex est un terme de la forme $(\lambda x.M)\ N$;
- Un terme est en forme normale s'il ne contient aucun rédex ;
- Un terme en forme normale ne peut plus être β-réduit.

Exemples

- $\lambda x.f\ x$ est en forme normale ;
- $(\lambda x.f\ x)\ y$ n'est pas en forme normale.

λ-calcul de Church

Normalisation

- Un terme E est (faiblement) normalisable s'il existe un terme E' en forme normale tel que $E \rightarrow_\beta^* E'$;
- Un terme E est fortement normalisable si toutes les réductions à partir de E sont finies.

Exemples

- $(\lambda x.f\ x)\ y$ est fortement normalisable, la forme normale est $f\ y$;
- Si $\Delta = \lambda x.x\ x$, $\Omega = \Delta\ \Delta$ n'est pas normalisable,
  $(\lambda x.x\ x)\ (\lambda x.x\ x) \rightarrow_\beta (\lambda x.x\ x)\ (\lambda x.x\ x) \rightarrow_\beta \ldots$;
- $(\lambda x.y)\ \Omega$ est normalisable, la forme normale est y, mais il n'est pas fortement normalisable car $\Omega \rightarrow_\beta \Omega \rightarrow_\beta \ldots$, donc
  $(\lambda x.y)\ \Omega \rightarrow_\beta (\lambda x.y)\ \Omega \rightarrow_\beta \ldots$

λ-calcul de Church

Théorèmes fondamentaux

- Théorème de Church-Rosser : si E_1 et E_2 sont des termes tels que $E_1 =_\beta E_2$, alors il existe un terme E tel que $E_1 \rightarrow_\beta^* E$ et $E_2 \rightarrow_\beta^* E$.
- Théorème de confluence (ou du losange) : Si E, E_1 et E_2 sont des termes tels que $E \rightarrow_\beta^* E_1$ et $E \rightarrow_\beta^* E_2$, alors il existe un terme E' tel que $E_1 \rightarrow_\beta^* E'$ et $E_2 \rightarrow_\beta^* E'$;
- Le théorème de confluence implique l'unicité de la forme normale.

λ-calcul de Church

Théorèmes fondamentaux

- Théorème de Church-Rosser : si E_1 et E_2 sont des termes tels que $E_1 =_\beta E_2$, alors il existe un terme E tel que $E_1 \rightarrow^*_\beta E$ et $E_2 \rightarrow^*_\beta E$.
- Théorème de confluence (ou du losange) : Si E, E_1 et E_2 sont des termes tels que $E \rightarrow^*_\beta E_1$ et $E \rightarrow^*_\beta E_2$, alors il existe un terme E' tel que $E_1 \rightarrow^*_\beta E'$ et $E_2 \rightarrow^*_\beta E'$;
- Le théorème de confluence implique l'unicité de la forme normale.

λ-calcul de Church

Théorèmes fondamentaux

- Théorème de Church-Rosser : si E_1 et E_2 sont des termes tels que $E_1 =_\beta E_2$, alors il existe un terme E tel que $E_1 \rightarrow_\beta^* E$ et $E_2 \rightarrow_\beta^* E$.
- Théorème de confluence (ou du losange) : Si E, E_1 et E_2 sont des termes tels que $E \rightarrow_\beta^* E_1$ et $E \rightarrow_\beta^* E_2$, alors il existe un terme E' tel que $E_1 \rightarrow_\beta^* E'$ et $E_2 \rightarrow_\beta^* E'$;
- Le théorème de confluence implique l'unicité de la forme normale.

λ-calcul de Church

Structures de données : les booléens

- *true* $= \lambda a, b.a$ et *false* $= \lambda a, b.b$;
- On a : *true* $x\ y \rightarrow_\beta^* x$ et *false* $x\ y \rightarrow_\beta^* y$;
- Définition de l'alternative *ifthenelse* $= \lambda b, u, v.b\ u\ v$:
  - *ifthenelse true* $x\ y = (\lambda b, u, v.b\ u\ v)$ *true* $x\ y \rightarrow_\beta^*$ *true* $x\ y \rightarrow_\beta^* x$;
  - *ifthenelse false* $x\ y = (\lambda b, u, v.b\ u\ v)$ *false* $x\ y \rightarrow_\beta^*$ *false* $x\ y \rightarrow_\beta^* y$.

λ-calcul de Church

Structures de données : les booléens

- $true = \lambda a, b.a$ et $false = \lambda a, b.b$;
- On a : $true\ x\ y \rightarrow_\beta^* x$ et $false\ x\ y \rightarrow_\beta^* y$;
- Définition de l'alternative $ifthenelse = \lambda b, u, v.b\ u\ v$:
  - $ifthenelse\ true\ x\ y = (\lambda b, u, v.b\ u\ v)\ true\ x\ y \rightarrow_\beta^* true\ x\ y \rightarrow_\beta^* x$;
  - $ifthenelse\ false\ x\ y = (\lambda b, u, v.b\ u\ v)\ false\ x\ y \rightarrow_\beta^* false\ x\ y \rightarrow_\beta^* y$.

λ-calcul de Church

Structures de données : les booléens

- $\mathit{true} = \lambda a, b.a$ et $\mathit{false} = \lambda a, b.b$;
- On a : $\mathit{true}\ x\ y \rightarrow_\beta^* x$ et $\mathit{false}\ x\ y \rightarrow_\beta^* y$;
- Définition de l'alternative $\mathit{ifthenelse} = \lambda b, u, v.b\ u\ v$:
  - $\mathit{ifthenelse}\ \mathit{true}\ x\ y = (\lambda b, u, v.b\ u\ v)\ \mathit{true}\ x\ y \rightarrow_\beta^* \mathit{true}\ x\ y \rightarrow_\beta^* x$;
  - $\mathit{ifthenelse}\ \mathit{false}\ x\ y = (\lambda b, u, v.b\ u\ v)\ \mathit{false}\ x\ y \rightarrow_\beta^* \mathit{false}\ x\ y \rightarrow_\beta^* y$.

λ-calcul de Church

Structures de données : les booléens

- *true* $= \lambda a, b.a$ et *false* $= \lambda a, b.b$;
- On a : *true* $x\ y \rightarrow_\beta^* x$ et *false* $x\ y \rightarrow_\beta^* y$;
- Définition de l'alternative *ifthenelse* $= \lambda b, u, v.b\ u\ v$:
  - *ifthenelse true* $x\ y = (\lambda b, u, v.b\ u\ v)$ *true* $x\ y \rightarrow_\beta^*$ *true* $x\ y \rightarrow_\beta^* x$;
  - *ifthenelse false* $x\ y = (\lambda b, u, v.b\ u\ v)$ *false* $x\ y \rightarrow_\beta^*$ *false* $x\ y \rightarrow_\beta^* y$.

λ-calcul de Church

Structures de données : les booléens

- *true* $= \lambda a, b.a$ et *false* $= \lambda a, b.b$;
- On a : *true* $x\ y \rightarrow_\beta^* x$ et *false* $x\ y \rightarrow_\beta^* y$;
- Définition de l'alternative *ifthenelse* $= \lambda b, u, v.b\ u\ v$:
  - *ifthenelse true* $x\ y = (\lambda b, u, v.b\ u\ v)$ *true* $x\ y \rightarrow_\beta^*$ *true* $x\ y \rightarrow_\beta^* x$;
  - *ifthenelse false* $x\ y = (\lambda b, u, v.b\ u\ v)$ *false* $x\ y \rightarrow_\beta^*$ *false* $x\ y \rightarrow_\beta^* y$.

λ-calcul de Church

Structures de données : les entiers ou itérateurs de Church

- Itérer une fonction f, n fois :
  - $0 = \lambda f, x.x$;
  - $1 = \lambda f, x.f\ x$;
  - $2 = \lambda f, x.f\ (f\ x)$;
  - $\ldots$
  - $n = \lambda f, x.f\ (f\ (\ldots(f\ x))) = \lambda f, x.f^n\ x$, avec f itérée n fois.
- Fonction successeur, deux solutions :
  - Ajouter une itération de f en tête : $\lambda n, f, x.f\ (n\ f\ x)$;
  - Ajouter une itération de f en queue : $\lambda n, f, x.n\ f\ (f\ x)$.

λ-calcul de Church

Structures de données : les entiers ou itérateurs de Church

- Itérer une fonction f, n fois :
  - $0 = \lambda f, x.x$;
  - $1 = \lambda f, x.f\ x$;
  - $2 = \lambda f, x.f\ (f\ x)$;
  - $\ldots$
  - $n = \lambda f, x.f\ (f\ (\ldots (f\ x))) = \lambda f, x.f^n\ x$, avec f itérée n fois.
- Fonction successeur, deux solutions :
  - Ajouter une itération de f en tête : $\lambda n, f, x.f\ (n\ f\ x)$;
  - Ajouter une itération de f en queue : $\lambda n, f, x.n\ f\ (f\ x)$.

λ-calcul de Church

Récursion

- Théorème : tout terme F a un point fixe X, autrement dit un terme X tel que $X =_\beta F\ X$. En fait, il existe un terme Y sans variable libre tel que pour tout F, $Y\ F$ soit un point fixe de F. Un tel terme est appelé un combinateur de point fixe ;
- Combinateur de point fixe de Church/Curry :
$Y = \lambda f.(\lambda x.f\ (x\ x))\ (\lambda x.f\ (x\ x))$;
- On a bien $Y\ F =_\beta F\ (Y\ F)$, mais pas $Y\ F \to^*_\beta F\ (Y\ F)$;
- Combinateur de point fixe de Turing :
$\Theta = (\lambda g, h.h\ (g\ g\ h))\ (\lambda g, h.h\ (g\ g\ h))$;
- On a bien $\Theta\ F =_\beta F\ (\Theta\ F)$ et $\Theta\ F \to^*_\beta F\ (\Theta\ F)$.

λ-calcul de Church

Récursion

- Théorème : tout terme F a un point fixe X, autrement dit un terme X tel que $X =_\beta F\ X$. En fait, il existe un terme Y sans variable libre tel que pour tout F, $Y\ F$ soit un point fixe de F. Un tel terme est appelé un combinateur de point fixe ;
- Combinateur de point fixe de Church/Curry :
  $Y = \lambda f.(\lambda x.f\ (x\ x))\ (\lambda x.f\ (x\ x))$;
- On a bien $Y\ F =_\beta F\ (Y\ F)$, mais pas $Y\ F \rightarrow^*_\beta F\ (Y\ F)$;
- Combinateur de point fixe de Turing :
  $\Theta = (\lambda g, h.h\ (g\ g\ h))\ (\lambda g, h.h\ (g\ g\ h))$;
- On a bien $\Theta\ F =_\beta F\ (\Theta\ F)$ et $\Theta\ F \rightarrow^*_\beta F\ (\Theta\ F)$.

λ-calcul de Church

Récursion

- Théorème : tout terme F a un point fixe X, autrement dit un terme X tel que $X =_\beta F\ X$. En fait, il existe un terme Y sans variable libre tel que pour tout F, $Y\ F$ soit un point fixe de F. Un tel terme est appelé un combinateur de point fixe ;
- Combinateur de point fixe de Church/Curry :
  $Y = \lambda f.(\lambda x.f\ (x\ x))\ (\lambda x.f\ (x\ x))$;
- On a bien $Y\ F =_\beta F\ (Y\ F)$, mais pas $Y\ F \rightarrow_\beta^* F\ (Y\ F)$;
- Combinateur de point fixe de Turing :
  $\Theta = (\lambda g, h.h\ (g\ g\ h))\ (\lambda g, h.h\ (g\ g\ h))$;
- On a bien $\Theta\ F =_\beta F\ (\Theta\ F)$ et $\Theta\ F \rightarrow_\beta^* F\ (\Theta\ F)$.

λ-calcul de Church

Récursion : exemple

- Écrire la fonction qui calcule la somme des n premiers entiers ;
- On suppose qu'on a les fonctions $if0thenelse$, $pred$ et $plus$;
- $F = \lambda f, n.if0thenelse\ n\ 0\ (plus\ n\ (f\ (pred\ n)))$;
- Exemple de calcul :
  $(\Theta\ F\ 2) \rightarrow_\beta^* F\ (\Theta\ F)\ 2 \rightarrow_\beta^*$
  $if0thenelse\ 2\ 0\ (plus\ 2\ ((\Theta\ F)\ (pred\ 2))) \rightarrow_\beta^*$
  $plus\ 2\ ((\Theta\ F)\ (pred\ 2)) \rightarrow_\beta^* plus\ 2\ ((\Theta\ F)\ 1) \rightarrow_\beta^*$
  $plus\ 2\ (F\ (\Theta\ F)\ 1) \rightarrow_\beta^*$
  $plus\ 2\ (if0thenelse\ 1\ 0\ (plus\ 1\ ((\Theta\ F)\ (pred\ 1)))) \rightarrow_\beta^*$
  $plus\ 2\ (plus\ 1\ ((\Theta\ F)\ (pred\ 1))) \rightarrow_\beta^* plus\ 2\ (plus\ 1\ (F\ (\Theta\ F)\ 0)) \rightarrow_\beta^*$
  $plus\ 2\ (plus\ 1\ (if0thenelse\ 0\ 0\ (plus\ 0\ ((\Theta\ F)\ (pred\ 0))))) \rightarrow_\beta^*$
  $plus\ 2\ (plus\ 1\ 0) \rightarrow_\beta^* 3$;
- Il faut une stratégie de réduction car on peut toujours réduire $\Theta\ F$ et partir sur une réduction infinie.

λ-calcul de Church

Récursion : exemple

- Écrire la fonction qui calcule la somme des n premiers entiers ;
- On suppose qu'on a les fonctions $if0thenelse$, $pred$ et $plus$;
- $F = \lambda f, n.if0thenelse\ n\ 0\ (plus\ n\ (f\ (pred\ n)))$;
- Exemple de calcul :
  $(\Theta\ F\ 2) \rightarrow^*_\beta F\ (\Theta\ F)\ 2 \rightarrow^*_\beta$
  $if0thenelse\ 2\ 0\ (plus\ 2\ ((\Theta\ F)\ (pred\ 2))) \rightarrow^*_\beta$
  $plus\ 2\ ((\Theta\ F)\ (pred\ 2)) \rightarrow^*_\beta plus\ 2\ ((\Theta\ F)\ 1) \rightarrow^*_\beta$
  $plus\ 2\ (F\ (\Theta\ F)\ 1) \rightarrow^*_\beta$
  $plus\ 2\ (if0thenelse\ 1\ 0\ (plus\ 1\ ((\Theta\ F)\ (pred\ 1)))) \rightarrow^*_\beta$
  $plus\ 2\ (plus\ 1\ ((\Theta\ F)\ (pred\ 1))) \rightarrow^*_\beta plus\ 2\ (plus\ 1\ (F\ (\Theta\ F)\ 0)) \rightarrow^*_\beta$
  $plus\ 2\ (plus\ 1\ (if0thenelse\ 0\ 0\ (plus\ 0\ ((\Theta\ F)\ (pred\ 0))))) \rightarrow^*_\beta$
  $plus\ 2\ (plus\ 1\ 0) \rightarrow^*_\beta 3$;
- Il faut une stratégie de réduction car on peut toujours réduire $\Theta\ F$ et partir sur une réduction infinie.

λ-calcul de Church

Récursion : exemple

- Écrire la fonction qui calcule la somme des n premiers entiers ;
- On suppose qu'on a les fonctions $if0thenelse$, $pred$ et $plus$;
- $F = \lambda f, n.if0thenelse\ n\ 0\ (plus\ n\ (f\ (pred\ n)))$;
- Exemple de calcul :
  $(\Theta\ F\ 2) \rightarrow_\beta^* F\ (\Theta\ F)\ 2 \rightarrow_\beta^*$
  $if0thenelse\ 2\ 0\ (plus\ 2\ ((\Theta\ F)\ (pred\ 2))) \rightarrow_\beta^*$
  $plus\ 2\ ((\Theta\ F)\ (pred\ 2)) \rightarrow_\beta^* plus\ 2\ ((\Theta\ F)\ 1) \rightarrow_\beta^*$
  $plus\ 2\ (F\ (\Theta\ F)\ 1) \rightarrow_\beta^*$
  $plus\ 2\ (if0thenelse\ 1\ 0\ (plus\ 1\ ((\Theta\ F)\ (pred\ 1)))) \rightarrow_\beta^*$
  $plus\ 2\ (plus\ 1\ ((\Theta\ F)\ (pred\ 1))) \rightarrow_\beta^* plus\ 2\ (plus\ 1\ (F\ (\Theta\ F)\ 0)) \rightarrow_\beta^*$
  $plus\ 2\ (plus\ 1\ (if0thenelse\ 0\ 0\ (plus\ 0\ ((\Theta\ F)\ (pred\ 0))))) \rightarrow_\beta^*$
  $plus\ 2\ (plus\ 1\ 0) \rightarrow_\beta^* 3$;
- Il faut une stratégie de réduction car on peut toujours réduire $\Theta\ F$ et partir sur une réduction infinie.

λ-calcul de Church

Récursion : exemple

- Écrire la fonction qui calcule la somme des n premiers entiers ;
- On suppose qu'on a les fonctions $if0thenelse$, $pred$ et $plus$;
- $F = \lambda f, n.if0thenelse\ n\ 0\ (plus\ n\ (f\ (pred\ n)))$;
- Exemple de calcul :
  $(\Theta\ F\ 2) \rightarrow_\beta^* F\ (\Theta\ F)\ 2 \rightarrow_\beta^*$
  $if0thenelse\ 2\ 0\ (plus\ 2\ ((\Theta\ F)\ (pred\ 2))) \rightarrow_\beta^*$
  $plus\ 2\ ((\Theta\ F)\ (pred\ 2)) \rightarrow_\beta^* plus\ 2\ ((\Theta\ F)\ 1) \rightarrow_\beta^*$
  $plus\ 2\ (F\ (\Theta\ F)\ 1) \rightarrow_\beta^*$
  $plus\ 2\ (if0thenelse\ 1\ 0\ (plus\ 1\ ((\Theta\ F)\ (pred\ 1)))) \rightarrow_\beta^*$
  $plus\ 2\ (plus\ 1\ ((\Theta\ F)\ (pred\ 1))) \rightarrow_\beta^* plus\ 2\ (plus\ 1\ (F\ (\Theta\ F)\ 0)) \rightarrow_\beta^*$
  $plus\ 2\ (plus\ 1\ (if0thenelse\ 0\ 0\ (plus\ 0\ ((\Theta\ F)\ (pred\ 0))))) \rightarrow_\beta^*$
  $plus\ 2\ (plus\ 1\ 0) \rightarrow_\beta^* 3$;
- Il faut une stratégie de réduction car on peut toujours réduire $\Theta\ F$ et partir sur une réduction infinie.

λ-calcul de Church

Récursion : exemple

- Écrire la fonction qui calcule la somme des n premiers entiers ;
- On suppose qu'on a les fonctions $if0thenelse$, $pred$ et $plus$;
- $F = \lambda f, n.if0thenelse\ n\ 0\ (plus\ n\ (f\ (pred\ n)))$;
- Exemple de calcul :
  $(\Theta\ F\ 2) \rightarrow_\beta^* F\ (\Theta\ F)\ 2 \rightarrow_\beta^*$
  $if0thenelse\ 2\ 0\ (plus\ 2\ ((\Theta\ F)\ (pred\ 2))) \rightarrow_\beta^*$
  $plus\ 2\ ((\Theta\ F)\ (pred\ 2)) \rightarrow_\beta^* plus\ 2\ ((\Theta\ F)\ 1) \rightarrow_\beta^*$
  $plus\ 2\ (F\ (\Theta\ F)\ 1) \rightarrow_\beta^*$
  $plus\ 2\ (if0thenelse\ 1\ 0\ (plus\ 1\ ((\Theta\ F)\ (pred\ 1)))) \rightarrow_\beta^*$
  $plus\ 2\ (plus\ 1\ ((\Theta\ F)\ (pred\ 1))) \rightarrow_\beta^* plus\ 2\ (plus\ 1\ (F\ (\Theta\ F)\ 0)) \rightarrow_\beta^*$
  $plus\ 2\ (plus\ 1\ (if0thenelse\ 0\ 0\ (plus\ 0\ ((\Theta\ F)\ (pred\ 0))))) \rightarrow_\beta^*$
  $plus\ 2\ (plus\ 1\ 0) \rightarrow_\beta^* 3$;
- Il faut une stratégie de réduction car on peut toujours réduire $\Theta\ F$ et partir sur une réduction infinie.

Récursion : exemple

- Écrire la fonction qui calcule la somme des n premiers entiers ;
- On suppose qu'on a les fonctions $if0thenelse$, $pred$ et $plus$;
- $F = \lambda f, n.if0thenelse\ n\ 0\ (plus\ n\ (f\ (pred\ n)))$;
- Exemple de calcul :
  $(\Theta\ F\ 2) \rightarrow^*_\beta F\ (\Theta\ F)\ 2 \rightarrow^*_\beta$
  $if0thenelse\ 2\ 0\ (plus\ 2\ ((\Theta\ F)\ (pred\ 2))) \rightarrow^*_\beta$
  $plus\ 2\ ((\Theta\ F)\ (pred\ 2)) \rightarrow^*_\beta plus\ 2\ ((\Theta\ F)\ 1) \rightarrow^*_\beta$
  $plus\ 2\ (F\ (\Theta\ F)\ 1) \rightarrow^*_\beta$
  $plus\ 2\ (if0thenelse\ 1\ 0\ (plus\ 1\ ((\Theta\ F)\ (pred\ 1)))) \rightarrow^*_\beta$
  $plus\ 2\ (plus\ 1\ ((\Theta\ F)\ (pred\ 1))) \rightarrow^*_\beta plus\ 2\ (plus\ 1\ (F\ (\Theta\ F)\ 0)) \rightarrow^*_\beta$
  $plus\ 2\ (plus\ 1\ (if0thenelse\ 0\ 0\ (plus\ 0\ ((\Theta\ F)\ (pred\ 0))))) \rightarrow^*_\beta$
  $plus\ 2\ (plus\ 1\ 0) \rightarrow^*_\beta 3$;
- Il faut une stratégie de réduction car on peut toujours réduire $\Theta\ F$ et partir sur une réduction infinie.

λ-calcul de Church

Récursion : exemple

- Écrire la fonction qui calcule la somme des n premiers entiers ;
- On suppose qu'on a les fonctions $if0thenelse$, $pred$ et $plus$;
- $F = \lambda f, n.if0thenelse\ n\ 0\ (plus\ n\ (f\ (pred\ n)))$;
- Exemple de calcul :
  $(\Theta\ F\ 2) \rightarrow_\beta^* F\ (\Theta\ F)\ 2 \rightarrow_\beta^*$
  $if0thenelse\ 2\ 0\ (plus\ 2\ ((\Theta\ F)\ (pred\ 2))) \rightarrow_\beta^*$
  $plus\ 2\ ((\Theta\ F)\ (pred\ 2)) \rightarrow_\beta^* plus\ 2\ ((\Theta\ F)\ 1) \rightarrow_\beta^*$
  $plus\ 2\ (F\ (\Theta\ F)\ 1) \rightarrow_\beta^*$
  $plus\ 2\ (if0thenelse\ 1\ 0\ (plus\ 1\ ((\Theta\ F)\ (pred\ 1)))) \rightarrow_\beta^*$
  $plus\ 2\ (plus\ 1\ ((\Theta\ F)\ (pred\ 1))) \rightarrow_\beta^* plus\ 2\ (plus\ 1\ (F\ (\Theta\ F)\ 0)) \rightarrow_\beta^*$
  $plus\ 2\ (plus\ 1\ (if0thenelse\ 0\ 0\ (plus\ 0\ ((\Theta\ F)\ (pred\ 0))))) \rightarrow_\beta^*$
  $plus\ 2\ (plus\ 1\ 0) \rightarrow_\beta^* 3$;
- Il faut une stratégie de réduction car on peut toujours réduire $\Theta\ F$ et partir sur une réduction infinie.

λ-calcul de Church

Récursion : exemple

- Écrire la fonction qui calcule la somme des n premiers entiers ;
- On suppose qu'on a les fonctions $if0thenelse$, $pred$ et $plus$;
- $F = \lambda f, n.if0thenelse\ n\ 0\ (plus\ n\ (f\ (pred\ n)))$;
- Exemple de calcul :
  $(\Theta\ F\ 2) \rightarrow_\beta^* F\ (\Theta\ F)\ 2 \rightarrow_\beta^*$
  $if0thenelse\ 2\ 0\ (plus\ 2\ ((\Theta\ F)\ (pred\ 2))) \rightarrow_\beta^*$
  $plus\ 2\ ((\Theta\ F)\ (pred\ 2)) \rightarrow_\beta^* plus\ 2\ ((\Theta\ F)\ 1) \rightarrow_\beta^*$
  $plus\ 2\ (F\ (\Theta\ F)\ 1) \rightarrow_\beta^*$
  $plus\ 2\ (if0thenelse\ 1\ 0\ (plus\ 1\ ((\Theta\ F)\ (pred\ 1)))) \rightarrow_\beta^*$
  $plus\ 2\ (plus\ 1\ ((\Theta\ F)\ (pred\ 1))) \rightarrow_\beta^* plus\ 2\ (plus\ 1\ (F\ (\Theta\ F)\ 0)) \rightarrow_\beta^*$
  $plus\ 2\ (plus\ 1\ (if0thenelse\ 0\ 0\ (plus\ 0\ ((\Theta\ F)\ (pred\ 0))))) \rightarrow_\beta^*$
  $plus\ 2\ (plus\ 1\ 0) \rightarrow_\beta^* 3$;
- Il faut une stratégie de réduction car on peut toujours réduire $\Theta\ F$ et partir sur une réduction infinie.

λ-calcul de Church

Récursion : exemple

- Écrire la fonction qui calcule la somme des n premiers entiers ;
- On suppose qu'on a les fonctions $if0thenelse$, $pred$ et $plus$;
- $F = \lambda f, n.if0thenelse\ n\ 0\ (plus\ n\ (f\ (pred\ n)))$;
- Exemple de calcul :
  $(\Theta\ F\ 2) \rightarrow_\beta^* F\ (\Theta\ F)\ 2 \rightarrow_\beta^*$
  $if0thenelse\ 2\ 0\ (plus\ 2\ ((\Theta\ F)\ (pred\ 2))) \rightarrow_\beta^*$
  $plus\ 2\ ((\Theta\ F)\ (pred\ 2)) \rightarrow_\beta^* plus\ 2\ ((\Theta\ F)\ 1) \rightarrow_\beta^*$
  $plus\ 2\ (F\ (\Theta\ F)\ 1) \rightarrow_\beta^*$
  $plus\ 2\ (if0thenelse\ 1\ 0\ (plus\ 1\ ((\Theta\ F)\ (pred\ 1)))) \rightarrow_\beta^*$
  $plus\ 2\ (plus\ 1\ ((\Theta\ F)\ (pred\ 1))) \rightarrow_\beta^* plus\ 2\ (plus\ 1\ (F\ (\Theta\ F)\ 0)) \rightarrow_\beta^*$
  $plus\ 2\ (plus\ 1\ (if0thenelse\ 0\ 0\ (plus\ 0\ ((\Theta\ F)\ (pred\ 0))))) \rightarrow_\beta^*$
  $plus\ 2\ (plus\ 1\ 0) \rightarrow_\beta^* 3$;
- Il faut une stratégie de réduction car on peut toujours réduire $\Theta\ F$ et partir sur une réduction infinie.

λ-calcul de Church

Récursion : exemple

- Écrire la fonction qui calcule la somme des n premiers entiers ;
- On suppose qu'on a les fonctions $if0thenelse$, $pred$ et $plus$;
- $F = \lambda f, n.if0thenelse\ n\ 0\ (plus\ n\ (f\ (pred\ n)))$;
- Exemple de calcul :
  $(\Theta\ F\ 2) \rightarrow_\beta^* F\ (\Theta\ F)\ 2 \rightarrow_\beta^*$
  $if0thenelse\ 2\ 0\ (plus\ 2\ ((\Theta\ F)\ (pred\ 2))) \rightarrow_\beta^*$
  $plus\ 2\ ((\Theta\ F)\ (pred\ 2)) \rightarrow_\beta^* plus\ 2\ ((\Theta\ F)\ 1) \rightarrow_\beta^*$
  $plus\ 2\ (F\ (\Theta\ F)\ 1) \rightarrow_\beta^*$
  $plus\ 2\ (if0thenelse\ 1\ 0\ (plus\ 1\ ((\Theta\ F)\ (pred\ 1)))) \rightarrow_\beta^*$
  $plus\ 2\ (plus\ 1\ ((\Theta\ F)\ (pred\ 1))) \rightarrow_\beta^* plus\ 2\ (plus\ 1\ (F\ (\Theta\ F)\ 0)) \rightarrow_\beta^*$
  $plus\ 2\ (plus\ 1\ (if0thenelse\ 0\ 0\ (plus\ 0\ ((\Theta\ F)\ (pred\ 0))))) \rightarrow_\beta^*$
  $plus\ 2\ (plus\ 1\ 0) \rightarrow_\beta^* 3$;
- Il faut une stratégie de réduction car on peut toujours réduire $\Theta\ F$ et partir sur une réduction infinie.

λ-calcul de Church

Récursion : exemple

- Écrire la fonction qui calcule la somme des n premiers entiers ;
- On suppose qu'on a les fonctions $if0thenelse$, $pred$ et $plus$;
- $F = \lambda f, n.if0thenelse\ n\ 0\ (plus\ n\ (f\ (pred\ n)))$;
- Exemple de calcul :
  $(\Theta\ F\ 2) \rightarrow^*_\beta F\ (\Theta\ F)\ 2 \rightarrow^*_\beta$
  $if0thenelse\ 2\ 0\ (plus\ 2\ ((\Theta\ F)\ (pred\ 2))) \rightarrow^*_\beta$
  $plus\ 2\ ((\Theta\ F)\ (pred\ 2)) \rightarrow^*_\beta plus\ 2\ ((\Theta\ F)\ 1) \rightarrow^*_\beta$
  $plus\ 2\ (F\ (\Theta\ F)\ 1) \rightarrow^*_\beta$
  $plus\ 2\ (if0thenelse\ 1\ 0\ (plus\ 1\ ((\Theta\ F)\ (pred\ 1)))) \rightarrow^*_\beta$
  $plus\ 2\ (plus\ 1\ ((\Theta\ F)\ (pred\ 1))) \rightarrow^*_\beta plus\ 2\ (plus\ 1\ (F\ (\Theta\ F)\ 0)) \rightarrow^*_\beta$
  $plus\ 2\ (plus\ 1\ (if0thenelse\ 0\ 0\ (plus\ 0\ ((\Theta\ F)\ (pred\ 0))))) \rightarrow^*_\beta$
  $plus\ 2\ (plus\ 1\ 0) \rightarrow^*_\beta 3$;
- Il faut une stratégie de réduction car on peut toujours réduire $\Theta\ F$ et partir sur une réduction infinie.

Récursion : exemple

- Écrire la fonction qui calcule la somme des n premiers entiers ;
- On suppose qu'on a les fonctions $if0thenelse$, $pred$ et $plus$;
- $F = \lambda f, n.if0thenelse\ n\ 0\ (plus\ n\ (f\ (pred\ n)))$;
- Exemple de calcul :
  $(\Theta\ F\ 2) \rightarrow_\beta^* F\ (\Theta\ F)\ 2 \rightarrow_\beta^*$
  $if0thenelse\ 2\ 0\ (plus\ 2\ ((\Theta\ F)\ (pred\ 2))) \rightarrow_\beta^*$
  $plus\ 2\ ((\Theta\ F)\ (pred\ 2)) \rightarrow_\beta^* plus\ 2\ ((\Theta\ F)\ 1) \rightarrow_\beta^*$
  $plus\ 2\ (F\ (\Theta\ F)\ 1) \rightarrow_\beta^*$
  $plus\ 2\ (if0thenelse\ 1\ 0\ (plus\ 1\ ((\Theta\ F)\ (pred\ 1)))) \rightarrow_\beta^*$
  $plus\ 2\ (plus\ 1\ ((\Theta\ F)\ (pred\ 1))) \rightarrow_\beta^* plus\ 2\ (plus\ 1\ (F\ (\Theta\ F)\ 0)) \rightarrow_\beta^*$
  $plus\ 2\ (plus\ 1\ (if0thenelse\ 0\ 0\ (plus\ 0\ ((\Theta\ F)\ (pred\ 0))))) \rightarrow_\beta^*$
  $plus\ 2\ (plus\ 1\ 0) \rightarrow_\beta^* 3$;
- Il faut une stratégie de réduction car on peut toujours réduire $\Theta\ F$ et partir sur une réduction infinie.

λ-calcul de Church

Récursion : exemple

- Écrire la fonction qui calcule la somme des n premiers entiers ;
- On suppose qu'on a les fonctions $if0thenelse$, $pred$ et $plus$;
- $F = \lambda f, n.if0thenelse\ n\ 0\ (plus\ n\ (f\ (pred\ n)))$;
- Exemple de calcul :
  $(\Theta\ F\ 2) \rightarrow^*_\beta F\ (\Theta\ F)\ 2 \rightarrow^*_\beta$
  $if0thenelse\ 2\ 0\ (plus\ 2\ ((\Theta\ F)\ (pred\ 2))) \rightarrow^*_\beta$
  $plus\ 2\ ((\Theta\ F)\ (pred\ 2)) \rightarrow^*_\beta plus\ 2\ ((\Theta\ F)\ 1) \rightarrow^*_\beta$
  $plus\ 2\ (F\ (\Theta\ F)\ 1) \rightarrow^*_\beta$
  $plus\ 2\ (if0thenelse\ 1\ 0\ (plus\ 1\ ((\Theta\ F)\ (pred\ 1)))) \rightarrow^*_\beta$
  $plus\ 2\ (plus\ 1\ ((\Theta\ F)\ (pred\ 1))) \rightarrow^*_\beta plus\ 2\ (plus\ 1\ (F\ (\Theta\ F)\ 0)) \rightarrow^*_\beta$
  $plus\ 2\ (plus\ 1\ (if0thenelse\ 0\ 0\ (plus\ 0\ ((\Theta\ F)\ (pred\ 0))))) \rightarrow^*_\beta$
  $plus\ 2\ (plus\ 1\ 0) \rightarrow^*_\beta 3$;
- Il faut une stratégie de réduction car on peut toujours réduire $\Theta\ F$ et partir sur une réduction infinie.

λ-calcul de Church

Récursion : exemple

- Écrire la fonction qui calcule la somme des n premiers entiers ;
- On suppose qu'on a les fonctions $if0thenelse$, $pred$ et $plus$;
- $F = \lambda f, n.if0thenelse\ n\ 0\ (plus\ n\ (f\ (pred\ n)))$;
- Exemple de calcul :
  $(\Theta\ F\ 2) \rightarrow_\beta^* F\ (\Theta\ F)\ 2 \rightarrow_\beta^*$
  $if0thenelse\ 2\ 0\ (plus\ 2\ ((\Theta\ F)\ (pred\ 2))) \rightarrow_\beta^*$
  $plus\ 2\ ((\Theta\ F)\ (pred\ 2)) \rightarrow_\beta^* plus\ 2\ ((\Theta\ F)\ 1) \rightarrow_\beta^*$
  $plus\ 2\ (F\ (\Theta\ F)\ 1) \rightarrow_\beta^*$
  $plus\ 2\ (if0thenelse\ 1\ 0\ (plus\ 1\ ((\Theta\ F)\ (pred\ 1)))) \rightarrow_\beta^*$
  $plus\ 2\ (plus\ 1\ ((\Theta\ F)\ (pred\ 1))) \rightarrow_\beta^* plus\ 2\ (plus\ 1\ (F\ (\Theta\ F)\ 0)) \rightarrow_\beta^*$
  $plus\ 2\ (plus\ 1\ (if0thenelse\ 0\ 0\ (plus\ 0\ ((\Theta\ F)\ (pred\ 0))))) \rightarrow_\beta^*$
  $plus\ 2\ (plus\ 1\ 0) \rightarrow_\beta^*$ 3 ;
- Il faut une stratégie de réduction car on peut toujours réduire $\Theta\ F$ et partir sur une réduction infinie.

λ-calcul de Church

Récursion : exemple

- Écrire la fonction qui calcule la somme des n premiers entiers ;
- On suppose qu'on a les fonctions $if0thenelse$, $pred$ et $plus$;
- $F = \lambda f, n.if0thenelse\ n\ 0\ (plus\ n\ (f\ (pred\ n)))$;
- Exemple de calcul :
  $(\Theta\ F\ 2) \rightarrow_\beta^* F\ (\Theta\ F)\ 2 \rightarrow_\beta^*$
  $if0thenelse\ 2\ 0\ (plus\ 2\ ((\Theta\ F)\ (pred\ 2))) \rightarrow_\beta^*$
  $plus\ 2\ ((\Theta\ F)\ (pred\ 2)) \rightarrow_\beta^* plus\ 2\ ((\Theta\ F)\ 1) \rightarrow_\beta^*$
  $plus\ 2\ (F\ (\Theta\ F)\ 1) \rightarrow_\beta^*$
  $plus\ 2\ (if0thenelse\ 1\ 0\ (plus\ 1\ ((\Theta\ F)\ (pred\ 1)))) \rightarrow_\beta^*$
  $plus\ 2\ (plus\ 1\ ((\Theta\ F)\ (pred\ 1))) \rightarrow_\beta^* plus\ 2\ (plus\ 1\ (F\ (\Theta\ F)\ 0)) \rightarrow_\beta^*$
  $plus\ 2\ (plus\ 1\ (if0thenelse\ 0\ 0\ (plus\ 0\ ((\Theta\ F)\ (pred\ 0))))) \rightarrow_\beta^*$
  $plus\ 2\ (plus\ 1\ 0) \rightarrow_\beta^* 3$;
- Il faut une stratégie de réduction car on peut toujours réduire $\Theta\ F$ et partir sur une réduction infinie.

λ-calcul de Church

Récursion : exemple

- Écrire la fonction qui calcule la somme des n premiers entiers ;
- On suppose qu'on a les fonctions $if0thenelse$, $pred$ et $plus$;
- $F = \lambda f, n.if0thenelse\ n\ 0\ (plus\ n\ (f\ (pred\ n)))$;
- Exemple de calcul :
  $(\Theta\ F\ 2) \rightarrow^*_\beta F\ (\Theta\ F)\ 2 \rightarrow^*_\beta$
  $if0thenelse\ 2\ 0\ (plus\ 2\ ((\Theta\ F)\ (pred\ 2))) \rightarrow^*_\beta$
  $plus\ 2\ ((\Theta\ F)\ (pred\ 2)) \rightarrow^*_\beta plus\ 2\ ((\Theta\ F)\ 1) \rightarrow^*_\beta$
  $plus\ 2\ (F\ (\Theta\ F)\ 1) \rightarrow^*_\beta$
  $plus\ 2\ (if0thenelse\ 1\ 0\ (plus\ 1\ ((\Theta\ F)\ (pred\ 1)))) \rightarrow^*_\beta$
  $plus\ 2\ (plus\ 1\ ((\Theta\ F)\ (pred\ 1))) \rightarrow^*_\beta plus\ 2\ (plus\ 1\ (F\ (\Theta\ F)\ 0)) \rightarrow^*_\beta$
  $plus\ 2\ (plus\ 1\ (if0thenelse\ 0\ 0\ (plus\ 0\ ((\Theta\ F)\ (pred\ 0))))) \rightarrow^*_\beta$
  $plus\ 2\ (plus\ 1\ 0) \rightarrow^*_\beta 3$;
- Il faut une stratégie de réduction car on peut toujours réduire $\Theta\ F$ et partir sur une réduction infinie.

Calcul des constructions

Langage

$$
\begin{array}{lcll}
t & ::= & \mathrm{Type} & \\
 & | & \mathrm{Prop} & \textit{Type de toutes les propositions} \\
 & | & x & \textit{Variables} \\
 & | & \Pi x : t.t & \textit{Produits} \\
 & | & \lambda x : t.t & \lambda\textit{-termes/fonctions} \\
 & | & t\ t & \textit{Application}
\end{array}
$$

Règles

- Jugements de typage :
  - De la forme : $x_1 : A_1, \ldots, x_n : A_n \vdash t : B$, où les x_i sont des variables, et les A_i et B des termes.
  - Si les variables x_i ont respectivement les types A_i alors t a le type B.
- Notations : Γ sera une séquence $x_1 : A_1, \ldots, x_n : A_n$; $A, B, C, \ldots$ seront des termes ; K, L seront soit Prop, soit Type.

Calcul des constructions

Règles

$$\frac{}{\Gamma \vdash \mathrm{Prop} : \mathrm{Type}}\ \mathrm{Prop} \qquad \frac{\Gamma \vdash A : K \qquad (x : A) \in \Gamma}{\Gamma \vdash x : A}\ \mathrm{Var}$$

$$\frac{\Gamma \vdash A : K \qquad \Gamma, x : A \vdash B : L}{\Gamma \vdash \Pi x : A.B : L}\ \mathrm{Prod}$$

$$\frac{\Gamma \vdash A : K \qquad \Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x : A.M : \Pi x : A.B}\ \mathrm{Lam}$$

$$\frac{\Gamma \vdash M : \Pi x : A.B \qquad \Gamma \vdash N : A}{\Gamma \vdash M\ N : B[N/x]}\ \mathrm{App}$$

$$\frac{\Gamma \vdash M : A \qquad A =_\beta B}{\Gamma \vdash M : B}\ \mathrm{Conv}$$

Calcul des constructions

Autres opérateurs logiques

- $A \Rightarrow B \equiv \Pi x : A.B\ (x \notin B)$.
- $A \wedge B \equiv \Pi C : \mathrm{Prop}.(A \Rightarrow B \Rightarrow C) \Rightarrow C$.
- $A \vee B \equiv \Pi C : \mathrm{Prop}.(A \Rightarrow C) \Rightarrow (B \Rightarrow C) \Rightarrow C$.
- $\neg A \equiv \Pi C : \mathrm{Prop}.A \Rightarrow C$.
- $\top \equiv \Pi C : \mathrm{Prop}.C \Rightarrow C$.
- $\bot \equiv \Pi C : \mathrm{Prop}.C$
- $\exists x : A.B \equiv \Pi C : \mathrm{Prop}.(\Pi x : A.B \Rightarrow C) \Rightarrow C$

Calcul des constructions

Fonctions

- On peut écrire des fonctions et les typer.
  Par exemple : $\lambda A, B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x$.

$$
\dfrac{\vdash \mathbf{Prop} : \mathbf{Type} \qquad \dfrac{\Gamma_1 \vdash \mathbf{Prop} : \mathbf{Type} \qquad \Gamma_2 = \Gamma_1, B : \mathbf{Prop} \vdash \begin{array}{l}\lambda x : A.\lambda y : B.x : \\ \Pi x : A.\Pi y : B.A\end{array}}{\Gamma_1 = A : \mathbf{Prop} \vdash \begin{array}{l}\lambda B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \\ \Pi B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A\end{array}}}{\vdash \begin{array}{l}\lambda A, B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \\ \Pi A, B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A\end{array}}
$$

Preuve Π_1 :

$$
\dfrac{\begin{array}{c}\vdots \\ \Gamma_2 \vdash A : \mathbf{Prop}\end{array} \qquad \dfrac{\begin{array}{c}\vdots \\ \Gamma_2 \vdash B : \mathbf{Prop}\end{array} \qquad \dfrac{\begin{array}{c}\vdots \\ \Gamma_3 \vdash A : \mathbf{Prop}\end{array} \qquad (x : A) \in \Gamma_3}{\Gamma_3 = \Gamma_2, y : B \vdash x : A}}{\Gamma_3 = \Gamma_2, x : A \vdash \lambda y : B.x : \Pi y : B.A}}{\Gamma_2 \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A}
$$

Calcul des constructions

Fonctions

- On peut écrire des fonctions et les typer.
  Par exemple : $\lambda A, B : \mathrm{Prop}.\lambda x : A.\lambda y : B.x$.

$$\Gamma_1 \vdash \mathrm{Prop} : \mathrm{Type} \qquad \Gamma_2 = \Gamma_1, B : \mathrm{Prop} \vdash \begin{array}{c}\lambda x : A.\lambda y : B.x : \\ \Pi x : A.\Pi y : B.A\end{array}$$

$$\dfrac{\vdash \mathbf{Prop} : \mathrm{Type} \qquad \Gamma_1 = A : \mathbf{Prop} \vdash \begin{array}{c}\lambda B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \\ \Pi B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A\end{array}}{\vdash \begin{array}{c}\lambda A, B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \\ \Pi A, B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A\end{array}}\ \mathbf{Lam}$$

Preuve Π_1 :

$$\Gamma_3 \vdash A : \mathrm{Prop} \qquad (x : A) \in \Gamma_3$$

$$\Gamma_2 \vdash B : \mathrm{Prop} \qquad \Gamma_3 = \Gamma_2, y : B \vdash x : A$$

$$\Gamma_2 \vdash A : \mathrm{Prop} \qquad \Gamma_3 = \Gamma_2, x : A \vdash \lambda y : B.x : \Pi y : B.A$$

$$\Gamma_2 \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A$$

Calcul des constructions

Fonctions

- On peut écrire des fonctions et les typer.
  Par exemple : $\lambda A, B : \mathrm{Prop}.\lambda x : A.\lambda y : B.x$.

$$\dfrac{\dfrac{}{\vdash \mathbf{Prop} : \mathbf{Type}}\,\mathbf{Prop} \qquad \dfrac{\Gamma_1 \vdash \mathrm{Prop} : \mathrm{Type} \qquad \Gamma_2 = \Gamma_1, B : \mathrm{Prop} \vdash \begin{array}{l}\lambda x : A.\lambda y : B.x : \\ \Pi x : A.\Pi y : B.A\end{array}}{\Gamma_1 = A : \mathbf{Prop} \vdash \begin{array}{l}\lambda B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \\ \Pi B : \mathrm{Prop}.\Pi x : A.\Pi y : B.A\end{array}}}{\vdash \begin{array}{l}\lambda A, B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \\ \Pi A, B : \mathrm{Prop}.\Pi x : A.\Pi y : B.A\end{array}}\,\mathbf{Lam}$$

Preuve Π_1 :

$$\dfrac{\begin{array}{c}\vdots \\ \Gamma_2 \vdash A : \mathrm{Prop}\end{array} \qquad \dfrac{\begin{array}{c}\vdots \\ \Gamma_2 \vdash B : \mathrm{Prop}\end{array} \qquad \dfrac{\begin{array}{c}\vdots \\ \Gamma_3 \vdash A : \mathrm{Prop}\end{array} \qquad (x : A) \in \Gamma_3}{\Gamma_3 = \Gamma_2, y : B \vdash x : A}}{\Gamma_3 = \Gamma_2, x : A \vdash \lambda y : B.x : \Pi y : B.A}}{\Gamma_2 \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A}$$

Calcul des constructions

Fonctions

- On peut écrire des fonctions et les typer.
  Par exemple : $\lambda A, B : \mathrm{Prop}.\lambda x : A.\lambda y : B.x$.

$$\dfrac{\dfrac{}{\vdash \mathbf{Prop} : \mathbf{Type}}\ \mathbf{Prop} \qquad \dfrac{\Gamma_1 \vdash \mathbf{Prop} : \mathrm{Type} \qquad \Gamma_2 = \Gamma_1, B : \mathbf{Prop} \vdash \begin{array}{l}\lambda x : A.\lambda y : B.x :\\ \Pi x : A.\Pi y : B.A\end{array}}{\Gamma_1 = A : \mathbf{Prop} \vdash \begin{array}{l}\lambda B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x :\\ \Pi B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A\end{array}}\ \mathbf{Lam}}{\vdash \begin{array}{l}\lambda A, B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x :\\ \Pi A, B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A\end{array}}\ \mathbf{Lam}$$

Preuve Π_1 :

$$\dfrac{\dfrac{\vdots}{\Gamma_2 \vdash A : \mathrm{Prop}} \qquad \dfrac{\dfrac{\vdots}{\Gamma_2 \vdash B : \mathrm{Prop}} \qquad \dfrac{\dfrac{\vdots}{\Gamma_3 \vdash A : \mathrm{Prop}} \qquad (x : A) \in \Gamma_3}{\Gamma_3 = \Gamma_2, y : B \vdash x : A}}{\Gamma_3 = \Gamma_2, x : A \vdash \lambda y : B.x : \Pi y : B.A}}{\Gamma_2 \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A}$$

Calcul des constructions

Fonctions

- On peut écrire des fonctions et les typer.
  Par exemple : $\lambda A, B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x$.

$$
\dfrac{
\dfrac{}{\vdash \mathbf{Prop} : \mathbf{Type}}\ \mathbf{Prop}
\qquad
\dfrac{
\dfrac{}{\Gamma_1 \vdash \mathbf{Prop} : \mathbf{Type}}\ \mathbf{Prop}
\qquad
\Gamma_2 = \Gamma_1, B : \mathbf{Prop} \vdash \begin{array}{l} \lambda x : A.\lambda y : B.x : \\ \Pi x : A.\Pi y : B.A \end{array}
}{
\Gamma_1 = A : \mathbf{Prop} \vdash \begin{array}{l} \lambda B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \\ \Pi B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A \end{array}
}\ \mathbf{Lam}
}{
\vdash \begin{array}{l} \lambda A, B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \\ \Pi A, B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A \end{array}
}\ \mathbf{Lam}
$$

Preuve Π_1 :

$$
\dfrac{
\begin{array}{c} \vdots \\ \Gamma_2 \vdash A : \mathbf{Prop} \end{array}
\qquad
\dfrac{
\begin{array}{c} \vdots \\ \Gamma_2 \vdash B : \mathbf{Prop} \end{array}
\qquad
\dfrac{
\begin{array}{c} \vdots \\ \Gamma_3 \vdash A : \mathbf{Prop} \end{array}
\qquad
(x : A) \in \Gamma_3
}{
\Gamma_3 = \Gamma_2, y : B \vdash x : A
}
}{
\Gamma_3 = \Gamma_2, x : A \vdash \lambda y : B.x : \Pi y : B.A
}
}{
\Gamma_2 \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A
}
$$

Calcul des constructions

Fonctions

- On peut écrire des fonctions et les typer.
  Par exemple : $\lambda A, B : \mathrm{Prop}.\lambda x : A.\lambda y : B.x$.

$$\dfrac{\dfrac{}{\vdash \mathbf{Prop} : \mathbf{Type}}\,\mathbf{Prop} \qquad \dfrac{\dfrac{}{\Gamma_1 \vdash \mathbf{Prop} : \mathbf{Type}}\,\mathbf{Prop} \qquad \dfrac{\Pi_1}{\Gamma_2 = \Gamma_1, B : \mathbf{Prop} \vdash \begin{array}{l}\lambda x : A.\lambda y : B.x : \\ \Pi x : A.\Pi y : B.A\end{array}}}{\Gamma_1 = A : \mathbf{Prop} \vdash \begin{array}{l}\lambda B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \\ \Pi B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A\end{array}}\,\mathbf{Lam}}{\vdash \begin{array}{l}\lambda A, B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \\ \Pi A, B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A\end{array}}\,\mathbf{Lam}$$

Preuve Π_1 :

$$\genfrac{}{}{0pt}{}{\genfrac{}{}{0pt}{}{\vdots}{\Gamma_2 \vdash A : \mathrm{Prop}} \qquad \genfrac{}{}{0pt}{}{\genfrac{}{}{0pt}{}{\vdots}{\Gamma_2 \vdash B : \mathrm{Prop}} \qquad \genfrac{}{}{0pt}{}{\genfrac{}{}{0pt}{}{\vdots}{\Gamma_3 \vdash A : \mathrm{Prop}} \qquad (x : A) \in \Gamma_3}{\Gamma_3 = \Gamma_2, y : B \vdash x : A}}{\Gamma_3 = \Gamma_2, x : A \vdash \lambda y : B.x : \Pi y : B.A}}{\Gamma_2 \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A}$$

Calcul des constructions

Fonctions

- On peut écrire des fonctions et les typer.
  Par exemple : $\lambda A, B : \mathrm{Prop}.\lambda x : A.\lambda y : B.x$.

$$
\dfrac{
  \dfrac{}{\vdash \mathbf{Prop} : \mathbf{Type}}\,\mathbf{Prop}
  \qquad
  \dfrac{
    \dfrac{}{\Gamma_1 \vdash \mathbf{Prop} : \mathbf{Type}}\,\mathbf{Prop}
    \qquad
    \dfrac{\Pi_1}{\Gamma_2 = \Gamma_1, B : \mathbf{Prop} \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A}
  }{\Gamma_1 = A : \mathbf{Prop} \vdash \lambda B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \Pi B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A}\,\mathbf{Lam}
}{\vdash \lambda A, B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \Pi A, B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A}\,\mathbf{Lam}
$$

Preuve Π_1 :

$$
\dfrac{
  \dfrac{\vdots}{\Gamma_2 \vdash A : \mathrm{Prop}}
  \qquad
  \dfrac{
    \dfrac{\vdots}{\Gamma_2 \vdash B : \mathrm{Prop}}
    \qquad
    \dfrac{
      \dfrac{\vdots}{\Gamma_3 \vdash A : \mathrm{Prop}}
      \qquad
      (x : A) \in \Gamma_3
    }{\Gamma_3 = \Gamma_2, y : B \vdash x : A}
  }{\Gamma_3 = \Gamma_2, x : A \vdash \lambda y : B.x : \Pi y : B.A}
}{\Gamma_2 \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A}
$$

Calcul des constructions

Fonctions

- On peut écrire des fonctions et les typer.
  Par exemple : $\lambda A, B : \mathrm{Prop}.\lambda x : A.\lambda y : B.x$.

$$
\dfrac{
\dfrac{}{\vdash \mathbf{Prop} : \mathbf{Type}}\;\mathbf{Prop}
\qquad
\dfrac{
\dfrac{}{\Gamma_1 \vdash \mathbf{Prop} : \mathbf{Type}}\;\mathbf{Prop}
\qquad
\dfrac{\Pi_1}{\Gamma_2 = \Gamma_1, B : \mathbf{Prop} \vdash \begin{array}{l}\lambda x : A.\lambda y : B.x : \\ \Pi x : A.\Pi y : B.A\end{array}}
}{\Gamma_1 = A : \mathbf{Prop} \vdash \begin{array}{l}\lambda B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \\ \Pi B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A\end{array}}\;\mathbf{Lam}
}{\vdash \begin{array}{l}\lambda A, B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \\ \Pi A, B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A\end{array}}\;\mathbf{Lam}
$$

Preuve Π_1 :

$$
\dfrac{
\begin{array}{c}\vdots \\ \Gamma_2 \vdash A : \mathrm{Prop}\end{array}
\qquad
\dfrac{
\begin{array}{c}\vdots \\ \Gamma_2 \vdash B : \mathrm{Prop}\end{array}
\qquad
\dfrac{\begin{array}{c}\vdots \\ \Gamma_3 \vdash A : \mathrm{Prop}\end{array} \qquad (x : A) \in \Gamma_3}{\Gamma_3 = \Gamma_2, y : B \vdash x : A}
}{\Gamma_3 = \Gamma_2, x : A \vdash \lambda y : B.x : \Pi y : B.A}
}{\Gamma_2 \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A}\;\mathbf{Lam}
$$

Calcul des constructions

Fonctions

- On peut écrire des fonctions et les typer.
  Par exemple : $\lambda A, B : \mathsf{Prop}.\lambda x : A.\lambda y : B.x$.

$$\dfrac{\dfrac{}{\vdash \mathbf{Prop} : \mathbf{Type}}\ \mathbf{Prop} \qquad \dfrac{\dfrac{}{\Gamma_1 \vdash \mathbf{Prop} : \mathbf{Type}}\ \mathbf{Prop} \qquad \dfrac{\Pi_1}{\Gamma_2 = \Gamma_1, B : \mathbf{Prop} \vdash \begin{array}{l}\lambda x : A.\lambda y : B.x : \\ \Pi x : A.\Pi y : B.A\end{array}}}{\Gamma_1 = A : \mathbf{Prop} \vdash \begin{array}{l}\lambda B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \\ \Pi B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A\end{array}}\ \mathbf{Lam}}{\vdash \begin{array}{l}\lambda A, B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \\ \Pi A, B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A\end{array}}\ \mathbf{Lam}$$

Preuve Π_1 :

$$\dfrac{\dfrac{\vdots}{\Gamma_2 \vdash A : \mathbf{Prop}}\ \mathbf{Var} \qquad \begin{array}{c} \begin{array}{c}\vdots \\ \Gamma_2 \vdash B : \mathbf{Prop}\end{array} \qquad \begin{array}{c}\vdots \\ \Gamma_3 \vdash A : \mathbf{Prop} \qquad (x : A) \in \Gamma_3 \\ \Gamma_3 = \Gamma_2, y : B \vdash x : A\end{array} \\ \Gamma_3 = \Gamma_2, x : A \vdash \lambda y : B.x : \Pi y : B.A \end{array}}{\Gamma_2 \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A}\ \mathbf{Lam}$$

Calcul des constructions

Fonctions

- On peut écrire des fonctions et les typer.
  Par exemple : $\lambda A, B : \mathsf{Prop}.\lambda x : A.\lambda y : B.x$.

$$
\dfrac{
\dfrac{}{\vdash \mathsf{Prop} : \mathsf{Type}}\,\mathsf{Prop}
\qquad
\dfrac{
\dfrac{}{\Gamma_1 \vdash \mathsf{Prop} : \mathsf{Type}}\,\mathsf{Prop}
\qquad
\dfrac{\Pi_1}{\Gamma_2 = \Gamma_1, B : \mathsf{Prop} \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A}
}{\Gamma_1 = A : \mathsf{Prop} \vdash \lambda B : \mathsf{Prop}.\lambda x : A.\lambda y : B.x : \Pi B : \mathsf{Prop}.\Pi x : A.\Pi y : B.A}\,\mathsf{Lam}
}{\vdash \lambda A, B : \mathsf{Prop}.\lambda x : A.\lambda y : B.x : \Pi A, B : \mathsf{Prop}.\Pi x : A.\Pi y : B.A}\,\mathsf{Lam}
$$

Preuve Π_1 :

$$
\dfrac{
\dfrac{\vdots}{\Gamma_2 \vdash A : \mathsf{Prop}}\,\mathsf{Var}
\qquad
\dfrac{
\begin{matrix}\vdots \\ \Gamma_2 \vdash B : \mathsf{Prop}\end{matrix}
\qquad
\begin{matrix}\vdots & \\ \Gamma_3 \vdash A : \mathsf{Prop} & (x : A) \in \Gamma_3 \\ \Gamma_3 = \Gamma_2, y : B \vdash x : A & \end{matrix}
}{\Gamma_3 = \Gamma_2, x : A \vdash \lambda y : B.x : \Pi y : B.A}\,\mathsf{Lam}
}{\Gamma_2 \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A}\,\mathsf{Lam}
$$

Calcul des constructions

Fonctions

- On peut écrire des fonctions et les typer.
  Par exemple : $\lambda A, B : \mathsf{Prop}.\lambda x : A.\lambda y : B.x$.

$$
\dfrac{
\dfrac{}{\vdash \mathsf{Prop} : \mathsf{Type}}\,\mathsf{Prop}
\qquad
\dfrac{
\dfrac{}{\Gamma_1 \vdash \mathsf{Prop} : \mathsf{Type}}\,\mathsf{Prop}
\qquad
\dfrac{\Pi_1}{\Gamma_2 = \Gamma_1, B : \mathsf{Prop} \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A}
}{\Gamma_1 = A : \mathsf{Prop} \vdash \lambda B : \mathsf{Prop}.\lambda x : A.\lambda y : B.x : \Pi B : \mathsf{Prop}.\Pi x : A.\Pi y : B.A}\,\mathsf{Lam}
}{\vdash \lambda A, B : \mathsf{Prop}.\lambda x : A.\lambda y : B.x : \Pi A, B : \mathsf{Prop}.\Pi x : A.\Pi y : B.A}\,\mathsf{Lam}
$$

Preuve Π_1 :

$$
\dfrac{
\dfrac{\vdots}{\Gamma_2 \vdash A : \mathsf{Prop}}\,\mathsf{Var}
\qquad
\dfrac{
\dfrac{\vdots}{\Gamma_2 \vdash B : \mathsf{Prop}}\,\mathsf{Var}
\qquad
\begin{array}{c} \vdots \\ \Gamma_3 \vdash A : \mathsf{Prop} \qquad (x : A) \in \Gamma_3 \\ \Gamma_3 = \Gamma_2, y : B \vdash x : A \end{array}
}{\Gamma_3 = \Gamma_2, x : A \vdash \lambda y : B.x : \Pi y : B.A}\,\mathsf{Lam}
}{\Gamma_2 \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A}\,\mathsf{Lam}
$$

Calcul des constructions

Fonctions

- On peut écrire des fonctions et les typer.
  Par exemple : $\lambda A, B : \mathrm{Prop}.\lambda x : A.\lambda y : B.x$.

$$
\dfrac{
\dfrac{}{\vdash \mathbf{Prop} : \mathbf{Type}}\;\mathbf{Prop}
\qquad
\dfrac{
\dfrac{}{\Gamma_1 \vdash \mathbf{Prop} : \mathbf{Type}}\;\mathbf{Prop}
\qquad
\dfrac{\Pi_1}{\Gamma_2 = \Gamma_1, B : \mathbf{Prop} \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A}
}{\Gamma_1 = A : \mathbf{Prop} \vdash \lambda B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \Pi B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A}\;\mathbf{Lam}
}{\vdash \lambda A, B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \Pi A, B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A}\;\mathbf{Lam}
$$

Preuve Π_1 :

$$
\dfrac{
\dfrac{\vdots}{\Gamma_2 \vdash A : \mathbf{Prop}}\;\mathbf{Var}
\qquad
\dfrac{
\dfrac{\vdots}{\Gamma_2 \vdash B : \mathbf{Prop}}\;\mathbf{Var}
\qquad
\dfrac{\overset{\vdots}{\Gamma_3 \vdash A : \mathrm{Prop}} \qquad (x : A) \in \Gamma_3}{\Gamma_3 = \Gamma_2, y : B \vdash x : A}\;\mathbf{Var}
}{\Gamma_3 = \Gamma_2, x : A \vdash \lambda y : B.x : \Pi y : B.A}\;\mathbf{Lam}
}{\Gamma_2 \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A}\;\mathbf{Lam}
$$

Calcul des constructions

Fonctions

- On peut écrire des fonctions et les typer.
  Par exemple : $\lambda A, B : \mathrm{Prop}.\lambda x : A.\lambda y : B.x$.

$$\dfrac{\dfrac{}{\vdash \mathrm{Prop} : \mathrm{Type}}\,\mathrm{Prop} \qquad \dfrac{\dfrac{}{\Gamma_1 \vdash \mathrm{Prop} : \mathrm{Type}}\,\mathrm{Prop} \quad \dfrac{\Pi_1}{\Gamma_2 = \Gamma_1, B : \mathrm{Prop} \vdash \begin{array}{l}\lambda x : A.\lambda y : B.x :\\ \Pi x : A.\Pi y : B.A\end{array}}}{\Gamma_1 = A : \mathrm{Prop} \vdash \begin{array}{l}\lambda B : \mathrm{Prop}.\lambda x : A.\lambda y : B.x :\\ \Pi B : \mathrm{Prop}.\Pi x : A.\Pi y : B.A\end{array}}\,\mathrm{Lam}}{\vdash \begin{array}{l}\lambda A, B : \mathrm{Prop}.\lambda x : A.\lambda y : B.x :\\ \Pi A, B : \mathrm{Prop}.\Pi x : A.\Pi y : B.A\end{array}}\,\mathrm{Lam}$$

Preuve Π_1 :

$$\dfrac{\dfrac{\vdots}{\Gamma_2 \vdash A : \mathrm{Prop}}\,\mathrm{Var} \quad \dfrac{\dfrac{\vdots}{\Gamma_2 \vdash B : \mathrm{Prop}}\,\mathrm{Var} \quad \dfrac{\dfrac{\vdots}{\Gamma_3 \vdash A : \mathrm{Prop}}\,\mathrm{Var} \quad (x : A) \in \Gamma_3}{\Gamma_3 = \Gamma_2, y : B \vdash x : A}\,\mathrm{Var}}{\Gamma_3 = \Gamma_2, x : A \vdash \lambda y : B.x : \Pi y : B.A}\,\mathrm{Lam}}{\Gamma_2 \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A}\,\mathrm{Lam}$$

Calcul des constructions

Fonctions

- On peut écrire des fonctions et les typer.
  Par exemple : $\lambda A, B : \mathsf{Prop}.\lambda x : A.\lambda y : B.x$.

$$
\dfrac{
\dfrac{}{\vdash \mathsf{Prop} : \mathsf{Type}}\,\mathsf{Prop}
\qquad
\dfrac{
\dfrac{}{\Gamma_1 \vdash \mathsf{Prop} : \mathsf{Type}}\,\mathsf{Prop}
\qquad
\dfrac{\Pi_1}{\Gamma_2 = \Gamma_1, B : \mathsf{Prop} \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A}
}{\Gamma_1 = A : \mathsf{Prop} \vdash \lambda B : \mathsf{Prop}.\lambda x : A.\lambda y : B.x : \Pi B : \mathsf{Prop}.\Pi x : A.\Pi y : B.A}\,\mathsf{Lam}
}{\vdash \lambda A, B : \mathsf{Prop}.\lambda x : A.\lambda y : B.x : \Pi A, B : \mathsf{Prop}.\Pi x : A.\Pi y : B.A}\,\mathsf{Lam}
$$

Preuve Π_1 :

$$
\dfrac{
\dfrac{\vdots}{\Gamma_2 \vdash A : \mathsf{Prop}}\,\mathsf{Var}
\qquad
\dfrac{
\dfrac{\vdots}{\Gamma_2 \vdash B : \mathsf{Prop}}\,\mathsf{Var}
\qquad
\dfrac{
\dfrac{\vdots}{\Gamma_3 \vdash A : \mathsf{Prop}}\,\mathsf{Var}
\qquad
(x : A) \in \Gamma_3
}{\Gamma_3 = \Gamma_2, y : B \vdash x : A}\,\mathsf{Var}
}{\Gamma_3 = \Gamma_2, x : A \vdash \lambda y : B.x : \Pi y : B.A}\,\mathsf{Lam}
}{\Gamma_2 \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A}\,\mathsf{Lam}
$$

Calcul des constructions

Fonctions

- On peut écrire des fonctions et les typer.
  Par exemple : $\lambda A, B : \mathrm{Prop}.\lambda x : A.\lambda y : B.x$.

$$
\dfrac{
\dfrac{}{\vdash \mathbf{Prop} : \mathbf{Type}}\ \mathbf{Prop}
\qquad
\dfrac{
\dfrac{}{\Gamma_1 \vdash \mathbf{Prop} : \mathbf{Type}}\ \mathbf{Prop}
\qquad
\dfrac{\Pi_1}{\Gamma_2 = \Gamma_1, B : \mathbf{Prop} \vdash \begin{array}{l}\lambda x : A.\lambda y : B.x : \\ \Pi x : A.\Pi y : B.A\end{array}}
}{\Gamma_1 = A : \mathbf{Prop} \vdash \begin{array}{l}\lambda B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \\ \Pi B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A\end{array}}\ \mathbf{Lam}
}{\vdash \begin{array}{l}\lambda A, B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \\ \Pi A, B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A\end{array}}\ \mathbf{Lam}
$$

Preuve Π_1 :

$$
\dfrac{
\dfrac{\vdots}{\Gamma_2 \vdash A : \mathbf{Prop}}\ \mathbf{Var}
\qquad
\dfrac{
\dfrac{\vdots}{\Gamma_2 \vdash B : \mathbf{Prop}}\ \mathbf{Var}
\qquad
\dfrac{
\dfrac{\vdots}{\Gamma_3 \vdash A : \mathbf{Prop}}\ \mathbf{Var}
\qquad
(x : A) \in \Gamma_3
}{\Gamma_3 = \Gamma_2, y : B \vdash x : A}\ \mathbf{Var}
}{\Gamma_3 = \Gamma_2, x : A \vdash \lambda y : B.x : \Pi y : B.A}\ \mathbf{Lam}
}{\Gamma_2 \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A}\ \mathbf{Lam}
$$

Calcul des constructions

Fonctions

- On peut écrire des fonctions et les typer.
  Par exemple : $\lambda A, B : \mathrm{Prop}.\lambda x : A.\lambda y : B.x$.

$$
\dfrac{
\dfrac{}{\vdash \mathbf{Prop} : \mathbf{Type}}\;\mathbf{Prop}
\qquad
\dfrac{
\dfrac{}{\Gamma_1 \vdash \mathbf{Prop} : \mathbf{Type}}\;\mathbf{Prop}
\qquad
\dfrac{\Pi_1}{\Gamma_2 = \Gamma_1, B : \mathbf{Prop} \vdash \begin{array}{l}\lambda x : A.\lambda y : B.x : \\ \Pi x : A.\Pi y : B.A\end{array}}
}{\Gamma_1 = A : \mathbf{Prop} \vdash \begin{array}{l}\lambda B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \\ \Pi B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A\end{array}}\;\mathbf{Lam}
}{\vdash \begin{array}{l}\lambda A, B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \\ \Pi A, B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A\end{array}}\;\mathbf{Lam}
$$

Preuve Π_1 :

$$
\dfrac{
\dfrac{\vdots}{\Gamma_2 \vdash A : \mathbf{Prop}}\;\mathbf{Var}
\qquad
\dfrac{
\dfrac{\vdots}{\Gamma_2 \vdash B : \mathbf{Prop}}\;\mathbf{Var}
\qquad
\dfrac{
\dfrac{\vdots}{\Gamma_3 \vdash A : \mathbf{Prop}}\;\mathbf{Var}
\qquad
(x : A) \in \Gamma_3
}{\Gamma_3 = \Gamma_2, y : B \vdash x : A}\;\mathbf{Var}
}{\Gamma_3 = \Gamma_2, x : A \vdash \lambda y : B.x : \Pi y : B.A}\;\mathbf{Lam}
}{\Gamma_2 \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A}\;\mathbf{Lam}
$$

Calcul des constructions

Fonctions

- On peut écrire des fonctions et les typer.
  Par exemple : $\lambda A, B : \mathsf{Prop}.\lambda x : A.\lambda y : B.x$.

$$\dfrac{\dfrac{}{\vdash \mathsf{Prop} : \mathsf{Type}}\,\mathsf{Prop} \qquad \dfrac{\dfrac{}{\Gamma_1 \vdash \mathsf{Prop} : \mathsf{Type}}\,\mathsf{Prop} \qquad \dfrac{\Pi_1}{\Gamma_2 = \Gamma_1, B : \mathsf{Prop} \vdash \begin{array}{c}\lambda x : A.\lambda y : B.x : \\ \Pi x : A.\Pi y : B.A\end{array}}}{\Gamma_1 = A : \mathsf{Prop} \vdash \begin{array}{l}\lambda B : \mathsf{Prop}.\lambda x : A.\lambda y : B.x : \\ \Pi B : \mathsf{Prop}.\Pi x : A.\Pi y : B.A\end{array}}\,\mathsf{Lam}}{\vdash \begin{array}{l}\lambda A, B : \mathsf{Prop}.\lambda x : A.\lambda y : B.x : \\ \Pi A, B : \mathsf{Prop}.\Pi x : A.\Pi y : B.A\end{array}}\,\mathsf{Lam}$$

Preuve Π_1 :

$$\dfrac{\dfrac{\vdots}{\Gamma_2 \vdash A : \mathsf{Prop}}\,\mathsf{Var} \qquad \dfrac{\dfrac{\vdots}{\Gamma_2 \vdash B : \mathsf{Prop}}\,\mathsf{Var} \qquad \dfrac{\dfrac{\vdots}{\Gamma_3 \vdash A : \mathsf{Prop}}\,\mathsf{Var} \qquad (x : A) \in \Gamma_3}{\Gamma_3 = \Gamma_2, y : B \vdash x : A}\,\mathsf{Var}}{\Gamma_3 = \Gamma_2, x : A \vdash \lambda y : B.x : \Pi y : B.A}\,\mathsf{Lam}}{\Gamma_2 \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A}\,\mathsf{Lam}$$

Calcul des constructions

Fonctions

- On peut écrire des fonctions et les typer.
  Par exemple : $\lambda A, B : \mathrm{Prop}.\lambda x : A.\lambda y : B.x$.

$$\dfrac{\dfrac{}{\vdash \mathbf{Prop} : \mathbf{Type}}\ \mathbf{Prop} \qquad \dfrac{\dfrac{}{\Gamma_1 \vdash \mathbf{Prop} : \mathbf{Type}}\ \mathbf{Prop} \qquad \dfrac{\Pi_1}{\Gamma_2 = \Gamma_1, B : \mathbf{Prop} \vdash \begin{array}{c}\lambda x : A.\lambda y : B.x : \\ \Pi x : A.\Pi y : B.A\end{array}}}{\Gamma_1 = A : \mathbf{Prop} \vdash \begin{array}{c}\lambda B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \\ \Pi B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A\end{array}}\ \mathbf{Lam}}{\vdash \begin{array}{c}\lambda A, B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \\ \Pi A, B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A\end{array}}\ \mathbf{Lam}$$

Preuve Π_1 :

$$\dfrac{\dfrac{\vdots}{\Gamma_2 \vdash A : \mathbf{Prop}}\ \mathbf{Var} \qquad \dfrac{\dfrac{\vdots}{\Gamma_2 \vdash B : \mathbf{Prop}}\ \mathbf{Var} \qquad \dfrac{\dfrac{\vdots}{\Gamma_3 \vdash A : \mathbf{Prop}}\ \mathbf{Var} \qquad (x : A) \in \Gamma_3}{\Gamma_3 = \Gamma_2, y : B \vdash x : A}\ \mathbf{Var}}{\Gamma_3 = \Gamma_2, x : A \vdash \lambda y : B.x : \Pi y : B.A}\ \mathbf{Lam}}{\Gamma_2 \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A}\ \mathbf{Lam}$$

Calcul des constructions

Fonctions

- On peut écrire des fonctions et les typer.
  Par exemple : $\lambda A, B : \mathrm{Prop}.\lambda x : A.\lambda y : B.x$.

$$\dfrac{\dfrac{}{\vdash \mathbf{Prop} : \mathbf{Type}}\,\mathbf{Prop} \qquad \dfrac{\dfrac{}{\Gamma_1 \vdash \mathbf{Prop} : \mathbf{Type}}\,\mathbf{Prop} \qquad \dfrac{\Pi_1}{\Gamma_2 = \Gamma_1, B : \mathbf{Prop} \vdash \begin{array}{c}\lambda x : A.\lambda y : B.x : \\ \Pi x : A.\Pi y : B.A\end{array}}}{\Gamma_1 = A : \mathbf{Prop} \vdash \begin{array}{c}\lambda B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \\ \Pi B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A\end{array}}\,\mathbf{Lam}}{\vdash \begin{array}{c}\lambda A, B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \\ \Pi A, B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A\end{array}}\,\mathbf{Lam}$$

Preuve Π_1 :

$$\dfrac{\dfrac{\vdots}{\Gamma_2 \vdash A : \mathbf{Prop}}\,\mathbf{Var} \qquad \dfrac{\dfrac{\vdots}{\Gamma_2 \vdash B : \mathbf{Prop}}\,\mathbf{Var} \qquad \dfrac{\dfrac{\vdots}{\Gamma_3 \vdash A : \mathbf{Prop}}\,\mathbf{Var} \qquad (x : A) \in \Gamma_3}{\Gamma_3 = \Gamma_2, y : B \vdash x : A}\,\mathbf{Var}}{\Gamma_3 = \Gamma_2, x : A \vdash \lambda y : B.x : \Pi y : B.A}\,\mathbf{Lam}}{\Gamma_2 \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A}\,\mathbf{Lam}$$

Calcul des constructions

Fonctions

- Cette fonction est polymorphe (paramétrée par A et B de type Prop).
- Prop est le type des propositions mais on s'aperçoit qu'il est également habité par les types de données (Coq utilise plutôt la sorte Set pour les types de données).
- On pourrait rajouter la sorte Set : elle serait au même niveau que Prop, à savoir de type Type, et Prop et Set seraient incomparables.

Calcul des constructions

Preuves

- On peut écrire des preuves de propositions.
  Par exemple : $\Pi A, B : \mathrm{Prop}.\Pi x : A.\Pi y : B.A$.

$$\dfrac{\vdash \mathrm{Prop} : \mathrm{Type} \qquad \dfrac{\Gamma_1 \vdash \mathrm{Prop} : \mathrm{Type} \qquad \Gamma_2 = \Gamma_1, B : \mathrm{Prop} \vdash \begin{array}{l}\lambda x : A.\lambda y : B.x : \\ \Pi x : A.\Pi y : B.A\end{array}}{\Gamma_1 = A : \mathrm{Prop} \vdash \begin{array}{l}\lambda B : \mathrm{Prop}.\lambda x : A.\lambda y : B.x : \\ \Pi B : \mathrm{Prop}.\Pi x : A.\Pi y : B.A\end{array}}}{\vdash \begin{array}{l}\lambda A, B : \mathrm{Prop}.\lambda x : A.\lambda y : B.x : \\ \Pi A, B : \mathrm{Prop}.\Pi x : A.\Pi y : B.A\end{array}}$$

Preuve Π_1 :

$$\dfrac{\dfrac{\vdots}{\Gamma_2 \vdash A : \mathrm{Prop}} \qquad \dfrac{\dfrac{\vdots}{\Gamma_2 \vdash B : \mathrm{Prop}} \qquad \dfrac{\dfrac{\vdots}{\Gamma_3 \vdash A : \mathrm{Prop}} \qquad (x : A) \in \Gamma_3}{\Gamma_3 = \Gamma_2, y : B \vdash x : A}}{\Gamma_3 = \Gamma_2, x : A \vdash \lambda y : B.x : \Pi y : B.A}}{\Gamma_2 \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A}$$

Calcul des constructions

Preuves

- On peut écrire des preuves de propositions.
  Par exemple : $\Pi A, B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A$.

$$\Gamma_1 \vdash \mathbf{Prop} : \mathbf{Type} \qquad \Gamma_2 = \Gamma_1, B : \mathbf{Prop} \vdash \begin{array}{c} \lambda x : A.\lambda y : B.x : \\ \Pi x : A.\Pi y : B.A \end{array}$$

$$\dfrac{\vdash \mathbf{Prop} : \mathbf{Type} \qquad \Gamma_1 = A : \mathbf{Prop} \vdash \begin{array}{c} \lambda B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \\ \Pi B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A \end{array}}{\vdash \begin{array}{c} \lambda A, B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \\ \Pi A, B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A \end{array}} \ \mathbf{Lam}$$

Preuve Π_1 :

$$\begin{array}{c}
\vdots \\
\Gamma_3 \vdash A : \mathbf{Prop} \qquad (x : A) \in \Gamma_3 \\
\vdots \qquad\qquad \vdots \\
\Gamma_2 \vdash B : \mathbf{Prop} \qquad \Gamma_3 = \Gamma_2, y : B \vdash x : A \\
\vdots \\
\Gamma_2 \vdash A : \mathbf{Prop} \qquad \Gamma_3 = \Gamma_2, x : A \vdash \lambda y : B.x : \Pi y : B.A \\
\Gamma_2 \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A
\end{array}$$

Calcul des constructions

Preuves

- On peut écrire des preuves de propositions.
  Par exemple : $\Pi A, B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A$.

$$\Gamma_1 \vdash \mathbf{Prop} : \mathbf{Type} \qquad \Gamma_2 = \Gamma_1, B : \mathbf{Prop} \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A$$

$$\dfrac{\dfrac{}{\vdash \mathbf{Prop} : \mathbf{Type}}\,\mathbf{Prop} \qquad \Gamma_1 = A : \mathbf{Prop} \vdash \lambda B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \Pi B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A}{\vdash \lambda A, B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \Pi A, B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A}\,\mathbf{Lam}$$

Preuve Π_1 :

$$\dfrac{\dfrac{\vdots}{\Gamma_2 \vdash A : \mathbf{Prop}} \qquad \dfrac{\dfrac{\vdots}{\Gamma_2 \vdash B : \mathbf{Prop}} \qquad \dfrac{\dfrac{\vdots}{\Gamma_3 \vdash A : \mathbf{Prop}} \qquad (x : A) \in \Gamma_3}{\Gamma_3 = \Gamma_2, y : B \vdash x : A}}{\Gamma_3 = \Gamma_2, x : A \vdash \lambda y : B.x : \Pi y : B.A}}{\Gamma_2 \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A}$$

Calcul des constructions

Preuves

- On peut écrire des preuves de propositions.
  Par exemple : $\Pi A, B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A$.

$$
\dfrac{
  \dfrac{}{\vdash \mathbf{Prop} : \mathbf{Type}}\ \mathbf{Prop}
  \qquad
  \dfrac{
    \Gamma_1 \vdash \mathbf{Prop} : \mathrm{Type}
    \qquad
    \Gamma_2 = \Gamma_1, B : \mathbf{Prop} \vdash \begin{array}{c} \lambda x : A.\lambda y : B.x : \\ \Pi x : A.\Pi y : B.A \end{array}
  }{
    \Gamma_1 = A : \mathbf{Prop} \vdash \begin{array}{c} \lambda B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \\ \Pi B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A \end{array}
  }\ \mathbf{Lam}
}{
  \vdash \begin{array}{c} \lambda A, B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \\ \Pi A, B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A \end{array}
}\ \mathbf{Lam}
$$

Preuve Π_1 :

$$
\dfrac{
  \begin{array}{c} \vdots \\ \Gamma_2 \vdash A : \mathrm{Prop} \end{array}
  \qquad
  \dfrac{
    \begin{array}{c} \vdots \\ \Gamma_2 \vdash B : \mathrm{Prop} \end{array}
    \qquad
    \dfrac{
      \begin{array}{c} \vdots \\ \Gamma_3 \vdash A : \mathrm{Prop} \end{array}
      \qquad
      (x : A) \in \Gamma_3
    }{
      \Gamma_3 = \Gamma_2, y : B \vdash x : A
    }
  }{
    \Gamma_3 = \Gamma_2, x : A \vdash \lambda y : B.x : \Pi y : B.A
  }
}{
  \Gamma_2 \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A
}
$$

Calcul des constructions

Preuves

- On peut écrire des preuves de propositions.
  Par exemple : $\Pi A, B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A$.

$$
\dfrac{\dfrac{}{\vdash \mathbf{Prop} : \mathbf{Type}}\ \mathbf{Prop} \qquad \dfrac{\dfrac{}{\Gamma_1 \vdash \mathbf{Prop} : \mathbf{Type}}\ \mathbf{Prop} \qquad \Gamma_2 = \Gamma_1, B : \mathbf{Prop} \vdash \begin{array}{l} \lambda x : A.\lambda y : B.x : \\ \Pi x : A.\Pi y : B.A \end{array}}{\Gamma_1 = A : \mathbf{Prop} \vdash \begin{array}{l} \lambda B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \\ \Pi B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A \end{array}}\ \mathbf{Lam}}{\vdash \begin{array}{l} \lambda A, B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \\ \Pi A, B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A \end{array}}\ \mathbf{Lam}
$$

Preuve Π_1 :

$$
\dfrac{\begin{array}{c} \vdots \\ \Gamma_2 \vdash A : \mathbf{Prop} \end{array} \qquad \dfrac{\begin{array}{c} \vdots \\ \Gamma_2 \vdash B : \mathbf{Prop} \end{array} \qquad \dfrac{\begin{array}{c} \vdots \\ \Gamma_3 \vdash A : \mathbf{Prop} \end{array} \qquad (x : A) \in \Gamma_3}{\Gamma_3 = \Gamma_2, y : B \vdash x : A}}{\Gamma_3 = \Gamma_2, x : A \vdash \lambda y : B.x : \Pi y : B.A}}{\Gamma_2 \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A}
$$

Calcul des constructions

Preuves

- On peut écrire des preuves de propositions.
  Par exemple : $\Pi A, B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A$.

$$
\dfrac{\dfrac{}{\vdash \mathbf{Prop} : \mathbf{Type}}\ \mathbf{Prop} \qquad \dfrac{\dfrac{}{\Gamma_1 \vdash \mathbf{Prop} : \mathbf{Type}}\ \mathbf{Prop} \qquad \dfrac{\Pi_1}{\Gamma_2 = \Gamma_1, B : \mathbf{Prop} \vdash \begin{array}{l}\lambda x : A.\lambda y : B.x : \\ \Pi x : A.\Pi y : B.A\end{array}}}{\Gamma_1 = A : \mathbf{Prop} \vdash \begin{array}{l}\lambda B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \\ \Pi B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A\end{array}}\ \mathbf{Lam}}{\vdash \begin{array}{l}\lambda A, B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \\ \Pi A, B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A\end{array}}\ \mathbf{Lam}
$$

Preuve Π_1 :

$$
\dfrac{\dfrac{\vdots}{\Gamma_2 \vdash A : \mathbf{Prop}} \qquad \dfrac{\dfrac{\vdots}{\Gamma_2 \vdash B : \mathbf{Prop}} \qquad \dfrac{\dfrac{\vdots}{\Gamma_3 \vdash A : \mathbf{Prop}} \qquad (x : A) \in \Gamma_3}{\Gamma_3 = \Gamma_2, y : B \vdash x : A}}{\Gamma_3 = \Gamma_2, x : A \vdash \lambda y : B.x : \Pi y : B.A}}{\Gamma_2 \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A}
$$

Calcul des constructions

Preuves

- On peut écrire des preuves de propositions.
  Par exemple : $\Pi A, B : \mathrm{Prop}.\Pi x : A.\Pi y : B.A$.

$$
\dfrac{
\dfrac{}{\vdash \mathbf{Prop} : \mathbf{Type}}\,\mathbf{Prop}
\qquad
\dfrac{
\dfrac{}{\Gamma_1 \vdash \mathbf{Prop} : \mathbf{Type}}\,\mathbf{Prop}
\qquad
\dfrac{\Pi_1}{\Gamma_2 = \Gamma_1, B : \mathbf{Prop} \vdash \begin{array}{c}\lambda x : A.\lambda y : B.x :\\ \Pi x : A.\Pi y : B.A\end{array}}
}{\Gamma_1 = A : \mathbf{Prop} \vdash \begin{array}{c}\lambda B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x :\\ \Pi B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A\end{array}}\,\mathbf{Lam}
}{\vdash \begin{array}{c}\lambda A, B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x :\\ \Pi A, B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A\end{array}}\,\mathbf{Lam}
$$

Preuve Π_1 :

$$
\dfrac{
\dfrac{\vdots}{\Gamma_2 \vdash A : \mathrm{Prop}}
\qquad
\dfrac{
\dfrac{\vdots}{\Gamma_2 \vdash B : \mathrm{Prop}}
\qquad
\dfrac{
\dfrac{\vdots}{\Gamma_3 \vdash A : \mathrm{Prop}}
\qquad
(x : A) \in \Gamma_3
}{\Gamma_3 = \Gamma_2, y : B \vdash x : A}
}{\Gamma_3 = \Gamma_2, x : A \vdash \lambda y : B.x : \Pi y : B.A}
}{\Gamma_2 \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A}
$$

Calcul des constructions

Preuves

- On peut écrire des preuves de propositions.
  Par exemple : $\Pi A, B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A$.

$$
\dfrac{\dfrac{}{\vdash \mathbf{Prop} : \mathbf{Type}}\ \mathbf{Prop} \qquad \dfrac{\dfrac{}{\Gamma_1 \vdash \mathbf{Prop} : \mathbf{Type}}\ \mathbf{Prop} \qquad \dfrac{\Pi_1}{\Gamma_2 = \Gamma_1, B : \mathbf{Prop} \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A}}{\Gamma_1 = A : \mathbf{Prop} \vdash \lambda B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \Pi B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A}\ \mathbf{Lam}}{\vdash \lambda A, B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \Pi A, B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A}\ \mathbf{Lam}
$$

Preuve Π_1 :

$$
\dfrac{\begin{array}{c}\vdots\\ \Gamma_2 \vdash A : \mathbf{Prop}\end{array} \qquad \begin{array}{c} \begin{array}{c}\vdots\\ \Gamma_2 \vdash B : \mathbf{Prop}\end{array} \qquad \begin{array}{c} \begin{array}{c}\vdots\\ \Gamma_3 \vdash A : \mathbf{Prop}\end{array} \qquad (x : A) \in \Gamma_3 \\ \Gamma_3 = \Gamma_2, y : B \vdash x : A \end{array} \\ \Gamma_3 = \Gamma_2, x : A \vdash \lambda y : B.x : \Pi y : B.A \end{array}}{\Gamma_2 \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A}\ \mathbf{Lam}
$$

Calcul des constructions

Preuves

- On peut écrire des preuves de propositions.
  Par exemple : $\Pi A, B : \text{Prop}.\Pi x : A.\Pi y : B.A$.

$$\dfrac{\dfrac{}{\vdash \text{Prop} : \text{Type}}\ \text{Prop} \qquad \dfrac{\dfrac{}{\Gamma_1 \vdash \text{Prop} : \text{Type}}\ \text{Prop} \qquad \dfrac{\Pi_1}{\Gamma_2 = \Gamma_1, B : \text{Prop} \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A}}{\Gamma_1 = A : \text{Prop} \vdash \lambda B : \text{Prop}.\lambda x : A.\lambda y : B.x : \Pi B : \text{Prop}.\Pi x : A.\Pi y : B.A}\ \text{Lam}}{\vdash \lambda A, B : \text{Prop}.\lambda x : A.\lambda y : B.x : \Pi A, B : \text{Prop}.\Pi x : A.\Pi y : B.A}\ \text{Lam}$$

Preuve Π_1 :

$$\dfrac{\dfrac{\vdots}{\Gamma_2 \vdash A : \text{Prop}}\ \text{Var} \qquad \dfrac{\begin{matrix}\vdots \\ \Gamma_2 \vdash B : \text{Prop}\end{matrix} \qquad \dfrac{\begin{matrix}\vdots \\ \Gamma_3 \vdash A : \text{Prop}\end{matrix} \qquad (x : A) \in \Gamma_3}{\Gamma_3 = \Gamma_2, y : B \vdash x : A}}{\Gamma_3 = \Gamma_2, x : A \vdash \lambda y : B.x : \Pi y : B.A}}{\Gamma_2 \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A}\ \text{Lam}$$

Calcul des constructions

Preuves

- On peut écrire des preuves de propositions.
  Par exemple : $\Pi A, B : \text{Prop}.\Pi x : A.\Pi y : B.A$.

$$\dfrac{\dfrac{}{\vdash \textbf{Prop} : \textbf{Type}}\,\textbf{Prop} \qquad \dfrac{\dfrac{}{\Gamma_1 \vdash \textbf{Prop} : \textbf{Type}}\,\textbf{Prop} \qquad \dfrac{\Pi_1}{\Gamma_2 = \Gamma_1, B : \textbf{Prop} \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A}}{\Gamma_1 = A : \textbf{Prop} \vdash \lambda B : \textbf{Prop}.\lambda x : A.\lambda y : B.x : \Pi B : \textbf{Prop}.\Pi x : A.\Pi y : B.A}\,\textbf{Lam}}{\vdash \lambda A, B : \textbf{Prop}.\lambda x : A.\lambda y : B.x : \Pi A, B : \textbf{Prop}.\Pi x : A.\Pi y : B.A}\,\textbf{Lam}$$

Preuve Π_1 :

$$\dfrac{\dfrac{\vdots}{\Gamma_2 \vdash A : \textbf{Prop}}\,\textbf{Var} \qquad \dfrac{\begin{array}{c}\vdots\\ \Gamma_2 \vdash B : \textbf{Prop}\end{array} \qquad \begin{array}{c}\vdots\\ \Gamma_3 \vdash A : \textbf{Prop} \qquad (x : A) \in \Gamma_3\\ \Gamma_3 = \Gamma_2, y : B \vdash x : A\end{array}}{\Gamma_3 = \Gamma_2, x : A \vdash \lambda y : B.x : \Pi y : B.A}\,\textbf{Lam}}{\Gamma_2 \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A}\,\textbf{Lam}$$

Calcul des constructions

Preuves

- On peut écrire des preuves de propositions.
  Par exemple : $\Pi A, B : \mathrm{Prop}.\Pi x : A.\Pi y : B.A$.

$$
\dfrac{
\dfrac{}{\vdash \mathbf{Prop} : \mathbf{Type}}\ \mathbf{Prop}
\qquad
\dfrac{
\dfrac{}{\Gamma_1 \vdash \mathbf{Prop} : \mathbf{Type}}\ \mathbf{Prop}
\qquad
\dfrac{\Pi_1}{\Gamma_2 = \Gamma_1, B : \mathbf{Prop} \vdash \begin{array}{l}\lambda x : A.\lambda y : B.x :\\ \Pi x : A.\Pi y : B.A\end{array}}
}{\Gamma_1 = A : \mathbf{Prop} \vdash \begin{array}{l}\lambda B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x :\\ \Pi B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A\end{array}}\ \mathbf{Lam}
}{\vdash \begin{array}{l}\lambda A, B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x :\\ \Pi A, B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A\end{array}}\ \mathbf{Lam}
$$

Preuve Π_1 :

$$
\dfrac{
\dfrac{\vdots}{\Gamma_2 \vdash A : \mathbf{Prop}}\ \mathbf{Var}
\qquad
\dfrac{
\dfrac{\vdots}{\Gamma_2 \vdash B : \mathbf{Prop}}\ \mathbf{Var}
\qquad
\genfrac{}{}{0pt}{}{\dfrac{\vdots}{\Gamma_3 \vdash A : \mathbf{Prop}} \qquad (x : A) \in \Gamma_3}{\Gamma_3 = \Gamma_2, y : B \vdash x : A}
}{\Gamma_3 = \Gamma_2, x : A \vdash \lambda y : B.x : \Pi y : B.A}\ \mathbf{Lam}
}{\Gamma_2 \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A}\ \mathbf{Lam}
$$

Calcul des constructions

Preuves

- On peut écrire des preuves de propositions.
  Par exemple : $\Pi A, B : \mathrm{Prop}.\Pi x : A.\Pi y : B.A$.

$$
\dfrac{
\dfrac{}{\vdash \mathbf{Prop} : \mathbf{Type}}\ \mathbf{Prop}
\qquad
\dfrac{
\dfrac{}{\Gamma_1 \vdash \mathbf{Prop} : \mathbf{Type}}\ \mathbf{Prop}
\qquad
\dfrac{\Pi_1}{\Gamma_2 = \Gamma_1, B : \mathbf{Prop} \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A}
}{\Gamma_1 = A : \mathbf{Prop} \vdash \lambda B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \Pi B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A}\ \mathbf{Lam}
}{\vdash \lambda A, B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \Pi A, B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A}\ \mathbf{Lam}
$$

Preuve Π_1 :

$$
\dfrac{
\dfrac{\vdots}{\Gamma_2 \vdash A : \mathbf{Prop}}\ \mathbf{Var}
\qquad
\dfrac{
\dfrac{\vdots}{\Gamma_2 \vdash B : \mathbf{Prop}}\ \mathbf{Var}
\qquad
\dfrac{
\begin{matrix}\vdots \\ \Gamma_3 \vdash A : \mathbf{Prop}\end{matrix}
\qquad
(x : A) \in \Gamma_3
}{\Gamma_3 = \Gamma_2, y : B \vdash x : A}\ \mathbf{Var}
}{\Gamma_3 = \Gamma_2, x : A \vdash \lambda y : B.x : \Pi y : B.A}\ \mathbf{Lam}
}{\Gamma_2 \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A}\ \mathbf{Lam}
$$

Calcul des constructions

Preuves

- On peut écrire des preuves de propositions.
  Par exemple : $\Pi A, B : \text{Prop}.\Pi x : A.\Pi y : B.A$.

$$
\dfrac{
  \dfrac{}{\vdash \mathbf{Prop} : \mathbf{Type}}\ \mathbf{Prop}
  \qquad
  \dfrac{
    \dfrac{}{\Gamma_1 \vdash \mathbf{Prop} : \mathbf{Type}}\ \mathbf{Prop}
    \qquad
    \dfrac{\Pi_1}{\Gamma_2 = \Gamma_1, B : \mathbf{Prop} \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A}
  }{\Gamma_1 = A : \mathbf{Prop} \vdash \lambda B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \Pi B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A}\ \mathbf{Lam}
}{\vdash \lambda A, B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \Pi A, B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A}\ \mathbf{Lam}
$$

Preuve Π_1 :

$$
\dfrac{
  \dfrac{\vdots}{\Gamma_2 \vdash A : \mathbf{Prop}}\ \mathbf{Var}
  \qquad
  \dfrac{
    \dfrac{\vdots}{\Gamma_2 \vdash B : \mathbf{Prop}}\ \mathbf{Var}
    \qquad
    \dfrac{
      \dfrac{\vdots}{\Gamma_3 \vdash A : \mathbf{Prop}}\ \mathbf{Var}
      \qquad
      (x : A) \in \Gamma_3
    }{\Gamma_3 = \Gamma_2, y : B \vdash x : A}\ \mathbf{Var}
  }{\Gamma_3 = \Gamma_2, x : A \vdash \lambda y : B.x : \Pi y : B.A}\ \mathbf{Lam}
}{\Gamma_2 \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A}\ \mathbf{Lam}
$$

Calcul des constructions

Preuves

- On peut écrire des preuves de propositions.
  Par exemple : $\Pi A, B : \text{Prop}.\Pi x : A.\Pi y : B.A$.

$$\dfrac{\dfrac{}{\vdash \textbf{Prop} : \textbf{Type}}\,\textbf{Prop} \qquad \dfrac{\dfrac{}{\Gamma_1 \vdash \textbf{Prop} : \textbf{Type}}\,\textbf{Prop} \qquad \dfrac{\Pi_1}{\Gamma_2 = \Gamma_1, B : \textbf{Prop} \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A}}{\Gamma_1 = A : \textbf{Prop} \vdash \lambda B : \textbf{Prop}.\lambda x : A.\lambda y : B.x : \Pi B : \textbf{Prop}.\Pi x : A.\Pi y : B.A}\,\textbf{Lam}}{\vdash \lambda A, B : \textbf{Prop}.\lambda x : A.\lambda y : B.x : \Pi A, B : \textbf{Prop}.\Pi x : A.\Pi y : B.A}\,\textbf{Lam}$$

Preuve Π_1 :

$$\dfrac{\dfrac{\vdots}{\Gamma_2 \vdash A : \textbf{Prop}}\,\textbf{Var} \qquad \dfrac{\dfrac{\vdots}{\Gamma_2 \vdash B : \textbf{Prop}}\,\textbf{Var} \qquad \dfrac{\dfrac{\vdots}{\Gamma_3 \vdash A : \textbf{Prop}}\,\textbf{Var} \qquad (x : A) \in \Gamma_3}{\Gamma_3 = \Gamma_2, y : B \vdash x : A}\,\textbf{Var}}{\Gamma_3 = \Gamma_2, x : A \vdash \lambda y : B.x : \Pi y : B.A}\,\textbf{Lam}}{\Gamma_2 \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A}\,\textbf{Lam}$$

Calcul des constructions

Preuves

- On peut écrire des preuves de propositions.
  Par exemple : $\Pi A, B : \text{Prop}.\Pi x : A.\Pi y : B.A$.

$$\dfrac{\dfrac{}{\vdash \mathbf{Prop} : \mathbf{Type}}\,\mathbf{Prop} \qquad \dfrac{\dfrac{}{\Gamma_1 \vdash \mathbf{Prop} : \mathbf{Type}}\,\mathbf{Prop} \qquad \dfrac{\Pi_1}{\Gamma_2 = \Gamma_1, B : \mathbf{Prop} \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A}}{\Gamma_1 = A : \mathbf{Prop} \vdash \lambda B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \Pi B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A}\,\mathbf{Lam}}{\vdash \lambda A, B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \Pi A, B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A}\,\mathbf{Lam}$$

Preuve Π_1 :

$$\dfrac{\dfrac{\vdots}{\Gamma_2 \vdash A : \mathbf{Prop}}\,\mathbf{Var} \qquad \dfrac{\dfrac{\vdots}{\Gamma_2 \vdash B : \mathbf{Prop}}\,\mathbf{Var} \qquad \dfrac{\dfrac{\vdots}{\Gamma_3 \vdash A : \mathbf{Prop}}\,\mathbf{Var} \qquad (x : A) \in \Gamma_3}{\Gamma_3 = \Gamma_2, y : B \vdash x : A}\,\mathbf{Var}}{\Gamma_3 = \Gamma_2, x : A \vdash \lambda y : B.x : \Pi y : B.A}\,\mathbf{Lam}}{\Gamma_2 \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A}\,\mathbf{Lam}$$

Calcul des constructions

Preuves

- On peut écrire des preuves de propositions.
  Par exemple : $\Pi A, B : \mathrm{Prop}.\Pi x : A.\Pi y : B.A$.

$$\dfrac{\dfrac{}{\vdash \mathbf{Prop} : \mathbf{Type}}\,\mathbf{Prop} \qquad \dfrac{\dfrac{}{\Gamma_1 \vdash \mathbf{Prop} : \mathbf{Type}}\,\mathbf{Prop} \qquad \dfrac{\Pi_1}{\Gamma_2 = \Gamma_1, B : \mathbf{Prop} \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A}}{\Gamma_1 = A : \mathbf{Prop} \vdash \lambda B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \Pi B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A}\,\mathbf{Lam}}{\vdash \lambda A, B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \Pi A, B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A}\,\mathbf{Lam}$$

Preuve Π_1 :

$$\dfrac{\dfrac{\vdots}{\Gamma_2 \vdash A : \mathbf{Prop}}\,\mathbf{Var} \qquad \dfrac{\dfrac{\vdots}{\Gamma_2 \vdash B : \mathbf{Prop}}\,\mathbf{Var} \qquad \dfrac{\dfrac{\vdots}{\Gamma_3 \vdash A : \mathbf{Prop}}\,\mathbf{Var} \qquad (x : A) \in \Gamma_3}{\Gamma_3 = \Gamma_2, y : B \vdash x : A}\,\mathbf{Var}}{\Gamma_3 = \Gamma_2, x : A \vdash \lambda y : B.x : \Pi y : B.A}\,\mathbf{Lam}}{\Gamma_2 \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A}\,\mathbf{Lam}$$

Calcul des constructions

Preuves

- On peut écrire des preuves de propositions.
  Par exemple : $\Pi A, B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A$.

$$
\dfrac{
  \dfrac{}{\vdash \mathbf{Prop} : \mathbf{Type}}\,\mathbf{Prop}
  \qquad
  \dfrac{
    \dfrac{}{\Gamma_1 \vdash \mathbf{Prop} : \mathbf{Type}}\,\mathbf{Prop}
    \qquad
    \dfrac{\Pi_1}{\Gamma_2 = \Gamma_1, B : \mathbf{Prop} \vdash \begin{array}{c}\lambda x : A.\lambda y : B.x : \\ \Pi x : A.\Pi y : B.A\end{array}}
  }{\Gamma_1 = A : \mathbf{Prop} \vdash \begin{array}{c}\lambda B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \\ \Pi B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A\end{array}}\,\mathbf{Lam}
}{\vdash \begin{array}{c}\lambda A, B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \\ \Pi A, B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A\end{array}}\,\mathbf{Lam}
$$

Preuve Π_1 :

$$
\dfrac{
  \dfrac{\vdots}{\Gamma_2 \vdash A : \mathbf{Prop}}\,\mathbf{Var}
  \qquad
  \dfrac{
    \dfrac{\vdots}{\Gamma_2 \vdash B : \mathbf{Prop}}\,\mathbf{Var}
    \qquad
    \dfrac{
      \dfrac{\vdots}{\Gamma_3 \vdash A : \mathbf{Prop}}\,\mathbf{Var}
      \qquad
      (x : A) \in \Gamma_3
    }{\Gamma_3 = \Gamma_2, y : B \vdash x : A}\,\mathbf{Var}
  }{\Gamma_3 = \Gamma_2, x : A \vdash \lambda y : B.x : \Pi y : B.A}\,\mathbf{Lam}
}{\Gamma_2 \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A}\,\mathbf{Lam}
$$

Calcul des constructions

Preuves

- On peut écrire des preuves de propositions.
  Par exemple : $\Pi A, B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A$.

$$
\dfrac{
\dfrac{}{\vdash \mathbf{Prop} : \mathbf{Type}}\;\mathbf{Prop}
\qquad
\dfrac{
\dfrac{}{\Gamma_1 \vdash \mathbf{Prop} : \mathbf{Type}}\;\mathbf{Prop}
\qquad
\dfrac{\Pi_1}{\Gamma_2 = \Gamma_1, B : \mathbf{Prop} \vdash \begin{array}{c}\lambda x : A.\lambda y : B.x :\\ \Pi x : A.\Pi y : B.A\end{array}}
}{\Gamma_1 = A : \mathbf{Prop} \vdash \begin{array}{c}\lambda B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x :\\ \Pi B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A\end{array}}\;\mathbf{Lam}
}{\vdash \begin{array}{c}\lambda A, B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x :\\ \Pi A, B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A\end{array}}\;\mathbf{Lam}
$$

Preuve Π_1 :

$$
\dfrac{
\dfrac{\vdots}{\Gamma_2 \vdash A : \mathbf{Prop}}\;\mathbf{Var}
\qquad
\dfrac{
\dfrac{\vdots}{\Gamma_2 \vdash B : \mathbf{Prop}}\;\mathbf{Var}
\qquad
\dfrac{
\dfrac{\vdots}{\Gamma_3 \vdash A : \mathbf{Prop}}\;\mathbf{Var}
\qquad
(x : A) \in \Gamma_3
}{\Gamma_3 = \Gamma_2, y : B \vdash x : A}\;\mathbf{Var}
}{\Gamma_3 = \Gamma_2, x : A \vdash \lambda y : B.x : \Pi y : B.A}\;\mathbf{Lam}
}{\Gamma_2 \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A}\;\mathbf{Lam}
$$

Calcul des constructions

Preuves

- On peut écrire des preuves de propositions.
  Par exemple : $\Pi A, B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A$.

$$
\dfrac{
  \dfrac{}{\vdash \mathbf{Prop} : \mathbf{Type}}\ \mathbf{Prop}
  \qquad
  \dfrac{
    \dfrac{}{\Gamma_1 \vdash \mathbf{Prop} : \mathbf{Type}}\ \mathbf{Prop}
    \qquad
    \dfrac{\Pi_1}{\Gamma_2 = \Gamma_1, B : \mathbf{Prop} \vdash \begin{array}{c}\lambda x : A.\lambda y : B.x : \\ \Pi x : A.\Pi y : B.A\end{array}}
  }{\Gamma_1 = A : \mathbf{Prop} \vdash \begin{array}{c}\lambda B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \\ \Pi B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A\end{array}}\ \mathbf{Lam}
}{\vdash \begin{array}{c}\lambda A, B : \mathbf{Prop}.\lambda x : A.\lambda y : B.x : \\ \Pi A, B : \mathbf{Prop}.\Pi x : A.\Pi y : B.A\end{array}}\ \mathbf{Lam}
$$

Preuve Π_1 :

$$
\dfrac{
  \dfrac{\vdots}{\Gamma_2 \vdash A : \mathbf{Prop}}\ \mathbf{Var}
  \qquad
  \dfrac{
    \dfrac{\vdots}{\Gamma_2 \vdash B : \mathbf{Prop}}\ \mathbf{Var}
    \qquad
    \dfrac{
      \dfrac{\vdots}{\Gamma_3 \vdash A : \mathbf{Prop}}\ \mathbf{Var}
      \qquad
      (x : A) \in \Gamma_3
    }{\Gamma_3 = \Gamma_2, y : B \vdash x : A}\ \mathbf{Var}
  }{\Gamma_3 = \Gamma_2, x : A \vdash \lambda y : B.x : \Pi y : B.A}\ \mathbf{Lam}
}{\Gamma_2 \vdash \lambda x : A.\lambda y : B.x : \Pi x : A.\Pi y : B.A}\ \mathbf{Lam}
$$

Calcul des constructions

Preuves

- On peut identifier deux modes :
  - On a programmé t et on recherche son type : $\Gamma \vdash t :?$.
  - On a une proposition A et on recherche sa preuve : $\Gamma \vdash ? : A$.
  - On obtient dans les deux cas un arbre de preuve/typage.
  - Dans le cas des preuves, on gagne un objet preuve (un terme).
- Cela montre bien la dualité de l'isomorphisme de Curry-Howard :
  - propositions $\equiv$ types
  - preuves $\equiv$ programmes

Calcul des constructions

Preuves

- On peut identifier deux modes :
  - On a programmé t et on recherche son type : $\Gamma \vdash t :?$.
  - On a une proposition A et on recherche sa preuve : $\Gamma \vdash ? : A$.
  - On obtient dans les deux cas un arbre de preuve/typage.
  - Dans le cas des preuves, on gagne un objet preuve (un terme).
- Cela montre bien la dualité de l'isomorphisme de Curry-Howard :
  - propositions $\equiv$ types
  - preuves $\equiv$ programmes

Calcul des constructions

Preuves

- On peut identifier deux modes :
  - On a programmé t et on recherche son type : $\Gamma \vdash t :?$.
  - On a une proposition A et on recherche sa preuve : $\Gamma \vdash ? : A$.
  - On obtient dans les deux cas un arbre de preuve/typage.
  - Dans le cas des preuves, on gagne un objet preuve (un terme).
- Cela montre bien la dualité de l'isomorphisme de Curry-Howard :
  - propositions $\equiv$ types
  - preuves $\equiv$ programmes

Calcul des constructions

Preuves

- On peut identifier deux modes :
  - On a programmé t et on recherche son type : $\Gamma \vdash t :?$.
  - On a une proposition A et on recherche sa preuve : $\Gamma \vdash ? : A$.
  - On obtient dans les deux cas un arbre de preuve/typage.
  - Dans le cas des preuves, on gagne un objet preuve (un terme).
- Cela montre bien la dualité de l'isomorphisme de Curry-Howard :
  - propositions $\equiv$ types
  - preuves $\equiv$ programmes

Calcul des constructions

Preuves

- On peut identifier deux modes :
  - On a programmé t et on recherche son type : $\Gamma \vdash t :?$.
  - On a une proposition A et on recherche sa preuve : $\Gamma \vdash ? : A$.
  - On obtient dans les deux cas un arbre de preuve/typage.
  - Dans le cas des preuves, on gagne un objet preuve (un terme).
- Cela montre bien la dualité de l'isomorphisme de Curry-Howard :
  - propositions $\equiv$ types
  - preuves $\equiv$ programmes

Calcul des constructions en Coq

Syntaxe

CC	Coq
Type	Type
Prop	Prop
x	`x`
$\Pi x : A.B$	`forall (x : A),B`
$\lambda x : A.M$	`fun (x : A) => M`
$M\ N$	`M N`

Typage

- Typage avec la commande `Check`.

```
Coq < Check (fun (A : Prop) (a : A) => a).
fun (A : Prop) (a : A) => a
     : forall A : Prop, A -> A
```

Calcul des constructions en Coq

Environnement

- Le contexte Γ des dérivations peut être enrichi avec la commande `Paramater(s)` (contexte valable pour tout le reste de la session).

```
Coq < Parameter T : Prop.
T is declared

Coq < Parameter I : T.
I is declared
```

Évaluation

- Évaluation avec la commande `Eval compute in`.

```
Coq < Eval compute in
        ((fun (A : Prop) (a : A) => a) T I).
     = I
     : T
```

Calcul des constructions en Coq

Définitions

- On peut rajouter des définitions, qui sont constituées d'un nom, d'un type et d'un corps.
- Il faut une règle de réduction supplémentaire pour réduire les définitions (c'est-à-dire expanser le corps des définitions) : c'est la δ-réduction en Coq.
- La δ-réduction est capturée par `Eval compute in`.

```
Coq < Definition my_true (A : Prop) (a : A) := a.
my_true is defined

Coq < Eval compute in (my_true T I).
      = I
      : T
```

Calcul des constructions en Coq

Preuves

- On travaille sur la partie des types des jugements de typage.
- La partie terme peut-être :
  - Soit donnée directement par un λ-terme avec la tactique `exact`.
  - Soit construite progressivement par des tactiques (voir tableau).

Règle CC	Tactique Coq
Var	`assumption`
Lam	`intro`
App	`apply`
Conv	`compute`

Les règles Prop et Prod sont implicites.

Calcul des constructions en Coq

Preuves

- Exemple : on veut démontrer $\Pi A : \text{Prop}.A \Rightarrow A$.
  On utilise la notation "$\Rightarrow$" car pas de dépendance.

```
Lemma proof_1 : forall (A : Prop), A -> A.
Proof.
  exact (fun (A : Prop) (a : A) => a).
Qed.

Lemma proof_2 : forall (A : Prop), A -> A.
Proof.
  intro. (* Règle Prod *)
  intro. (* Règle Prod *)
  assumption. (* Règle Var *)
Qed.
```

Calcul des constructions en Coq

Preuves

- On peut afficher le contenu des preuves réalisées :

```
Print proof_1.
Print proof_2.
```

- Les deux commandes donnent exactement le même résultat :

```
proof_1 =
fun (A : Prop) (H : A) => H
    : forall A : Prop, A → A

proof_2 =
fun (A : Prop) (H : A) => H
    : forall A : Prop, A → A
```