INF9340 Devoir 1

Ce devoir est à remettre sur papier au secrétariat ou par courriel en format PDF avant 23h59 le dimanche 8 février 2026. Assurez-vous d'inclure votre nom et code permanent. Un gabarit LaTeX est fourni. Tous les problèmes sont assujettis à la politique du tableau noir (voir le site web du cours pour les détails de cette politique). N'oubliez pas que si vous utilisez la politique du tableau noir, vous *devez* indiquer avec qui vous avez collaboré.

Modification le 26 janvier : ajout de la question 11 pour ceux et celles qui souhaitent faire de l'implémentation.

Modification le 28 janvier : fix d'un mismatch entre down et les notes de cours dans la question 11.

1 Déduction naturelle (1 question au choix)

Question 1 (10 points). *Même si nous n'avons pas l'élimination de double négation* $\neg\neg A \supset A$ true *en général, nous avons néanmoins l'élimination de double négation pour les propositions de la forme* $\neg A$. *Démontrez que* $\neg\neg\neg A \supset \neg A$ true.

Question 2 (10 points). *Démontrez qu'une implication implique sa contraposée :* $(A \supset B) \supset (\neg B \supset \neg A)$ true. *Expliquez informellement pourquoi une proposition n'est pas équivalente à sa contraposée en logique constructiviste, c'est-à-dire, pourquoi on ne peut pas démontrer que* $(\neg B \supset \neg A) \supset (A \supset B)$ true.

2 Témoins (2 questions au choix)

Question 3 (10 points). *Démontrez* $(A \supset C) \supset ((B \supset D) \supset ((A \wedge B) \supset (C \wedge D)))$ true. *Donnez le témoin (term annotation) correspondant.*

Question 4 (10 points). *Pour chacun des témoins suivants, déterminez s'il est le témoin d'une proposition. Si oui, donnez la proposition correspondante. Si non, expliquez pourquoi une telle proposition n'existe pas.*

(1) $\lambda x.\lambda y.\lambda z.x(yz)$

(2) $\lambda x.xx$

(3) $\lambda x.\lambda y.\lambda z.xz(yz)$

(4) $\lambda x.\mathsf{case}\ ((\mathsf{fst}(x))(\mathsf{snd}(x)))\ \{\ \}$

Question 5 (10 points). *Concevez une affectation de témoins pour l'opérateur* $\otimes$ *défini par les règles suivantes.*

$$\frac{A\ \text{true} \quad B\ \text{true}}{A \otimes B\ \text{true}}\ (\otimes\text{I}) \qquad \frac{A \otimes B\ \text{true} \qquad \begin{array}{c}\overline{A\ \text{true}}\ u \quad \overline{B\ \text{true}}\ v \\ \vdots \\ C\ \text{true}\end{array}}{C\ \text{true}}\ (\otimes\text{E}^{u,v})$$

Si vos témoins lient des variables, spécifiez quelles variables sont liées et la portée. Donnez les règles de réduction et d'expansion pour vos témoins qui correspondent à la réduction locale et l'expansion locale des démonstrations. Décrivez informellement le sens de $A \otimes B$ true.

3 Correction et complétude locale (1 question au choix)

Question 6 (10 points). *Soit l'opérateur d'équivalence logique $\equiv$ défini par la règle suivante :*

$$\dfrac{\begin{array}{cc}\overline{A\ \mathsf{true}}^{\,u} & \overline{B\ \mathsf{true}}^{\,v} \\ \vdots & \vdots \\ B\ \mathsf{true} & A\ \mathsf{true}\end{array}}{A \equiv B\ \mathsf{true}}\ (\equiv \mathrm{I}^{u,v})$$

Donnez zéro, une ou plusieurs règles d'élimination qui sont localement correctes et complètes par rapport à cette règle. Concevez une affectation de témoins, et les règles de réduction et d'expansion correspondantes.

Question 7 (10 points). *Pour cet exercice, nous adoptons une perspective pragmatiste et supposons que la signification d'un connecteur est déterminée par ses règles d'élimination. La sûreté locale garantit alors que les règles d'introduction sont suffisamment fortes par rapport aux règles d'élimination, tandis que la complétude locale garantit qu'elles ne sont pas trop fortes. Considérez le connecteur ternaire ϕ, avec la règle d'élimination*

$$\dfrac{\phi(A,B,C)\ \mathsf{true} \qquad \begin{array}{c}\overline{A\ \mathsf{true}}^{\,u}\ \ \overline{B\ \mathsf{true}}^{\,v} \\ \vdots \\ D\ \mathsf{true}\end{array} \qquad \begin{array}{c}\overline{A\ \mathsf{true}}^{\,w}\ \ \overline{C\ \mathsf{true}}^{\,x} \\ \vdots \\ D\ \mathsf{true}\end{array}}{D\ \mathsf{true}}\ (\phi\,\mathrm{E}^{u,v,w,x})$$

Donnez zéro, une ou plusieurs règles d'introduction, et assurez-vous qu'elles sont localement correctes et complètes par rapport à la règle d'élimination. Concevez une affectation de témoins, et les règles de réduction et d'expansion correspondantes.

4 Vérifications et intercalation (2 questions théoriques au choix OU la question 11)

Question 8 (10 points). *Donnez toutes les vérifications de $(A \wedge B) \supset A$ et de $(\top \supset \bot) \supset \bot$, et les témoins correspondants.*

Question 9 (10 points). *Soit l'opérateur logique $\psi(A,B,C)$ défini par :*

$$\dfrac{\begin{array}{cc}\overline{A\ \mathsf{true}}^{\,u} & \overline{B\ \mathsf{true}}^{\,v} \\ \vdots & \\ C\ \mathsf{true} & \end{array}}{\psi(A,B,C)\ \mathsf{true}}\ (\psi\,\mathrm{I}^{u,v}) \qquad \dfrac{\psi(A,B,C) \quad A\ \mathsf{true} \quad B\ \mathsf{true}}{C\ \mathsf{true}}\ (\psi\,\mathrm{E})$$

Donnez des règles qui déterminent ses utilisations et ses vérifications.

Question 10 (10 points). *Donnez au moins 4 témoins qui correspondent à des vérifications de $(A \supset A) \supset (A \supset A)$. Quelle est la structure générale des témoins ? Sauriez-vous faire une conjecture sur quelle structure mathématique peut être représentée par les témoins de cette proposition ?*

Tâche 11 (20 points). *Implémentez l'algorithme d'intercalation. Vous avez l'option de définir une procédure de décision, qui prend comme entrée une proposition et retourne une valeur booléenne indiquant si la proposition est démontrable. Pour un plus grand défi, vous pouvez implémenter un démonstrateur automatique certifié qui retourne le témoin qui correspond à la vérification de la proposition, au lieu d'une valeur booléenne. Référez-vous à l'article [SB98] cité dans les notes de cours pour plus de détails sur l'algorithme et pour des conseils d'implémentation. Veuillez bien commenter votre code et inclure au moins une vingtaine de tests variés.*

Ma procédure de démonstration certifiée non optimisée fait environ 85 lignes d'OCaml. Pour vous donner un point de départ, voici les structures de données que j'ai utilisées pour représenter les propositions et les témoins, ainsi qu'une fonction auxiliaire pour générer des noms de variables uniques.

```
type atom = string
type prop =
  | Atomic of atom (* propositional variables *)
  | True
  | Absurd
  | And of prop * prop
  | Or of prop * prop
  | Implies of prop * prop

type varname = string
(* We annotate variables with types to make it *)
(* easier to interpret proof terms *)
type up = (* M, N = ... *)
  | Triv (* <> *)
  | Pair of up * up (* <M, N> *)
  | Fun of varname * prop * up (* fun (x:A) -> N *)
  | Inl of up (* inl M *)
  | Inr of up (* inr M *)
  (* case R of { inl(x:A) -> M | inr(y:B) -> N } *)
  | Case of down * (varname * prop * up)
                 * (varname * prop * up)
  | Abort of down (* abort R *)
  | Shift of down (* R *)
and down = (* R = ... *)
  | Var of varname
  | Projl of down (* projl R *)
  | Projr of down (* projr R *)
  | App of down * up  (* RM *)

(* Generates helper function for generating fresh *)
(* variable names that start by `base` *)
let strgengen base =
  let state = ref 0 in
  fun () ->
    let () = state := !state + 1 in
    base ^ (string_of_int !state)

let fresh = strgengen "x"

(* Implémenter au choix: *)
val decide : prop -> bool = ...
val certify : prop -> up option = ...
```