HOL4-Beagle, de l'ordre supérieur vers le premier ordre

Thibault Gauthier

October 10, 2013

Deux façon différentes de démontrer des théorèmes

	Prouveur interactif	Prouveur automatique
Prouveurs	HOL4, Coq, ...	Beagle, SPASS, ...
Expressivité	Ordre supérieur.	Premier ordre.
Efficacité	Un expert doit les guider.	Ils prouvent une grande diversité de problèmes.
Sûreté	Petit noyau.	Code assez long.

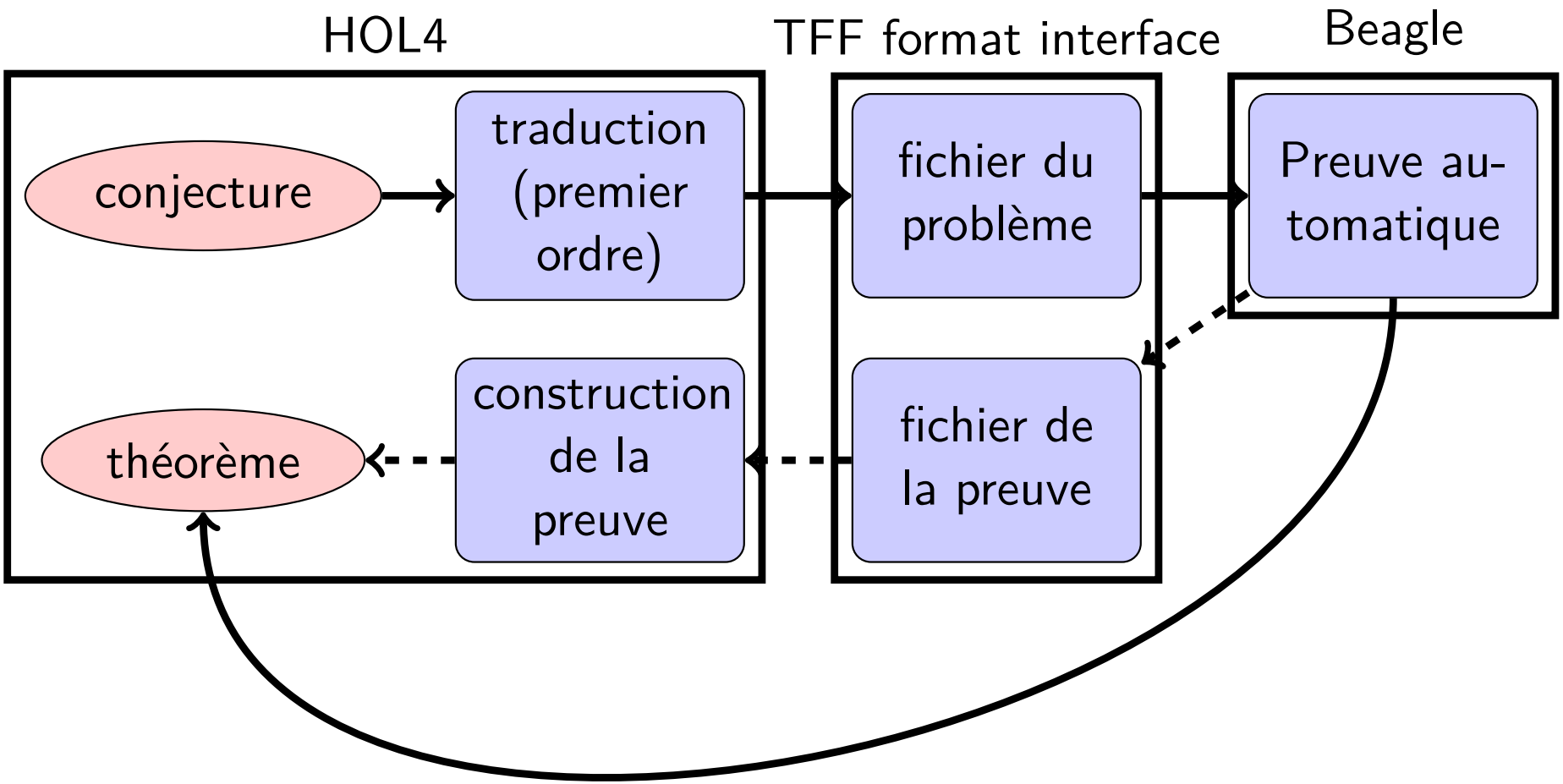
Comment pallier les faiblesses du prouveur interactif HOL4 à l'aide du prouveur automatique Beagle ?

Problème:
Le prouveur interne à HOL4 (METIS_TAC), ne résout pas de problème utilisant l'arithmétique sans l'aide d'un humain.

Comment pallier les faiblesses du prouveur interactif HOL4 à l'aide du prouveur automatique Beagle ?

Problème:
Le prouveur interne à HOL4 (METIS_TAC), ne résout pas de problème utilisant l'arithmétique sans l'aide d'un humain.
Solution:
Nous avons développé une nouvelle fonction BEAGLE_TAC. Elle appelle un prouveur automatique externe Beagle, qui a été conçu pour résoudre des problèmes arithmétiques.

Comment BEAGLE_TAC fonctionne?



Les avantages du format TFF(TPTP)

Voici une courte description du format TFF:

- Utilisation: répandu
- Formules: arithmétiques typées du premier ordre.

Ses avantages sont:

- Lisible par un humain
- Lisible par d'autres prouveurs automatiques

HOL4

Voici les caractéristiques de HOL4:

- Prouveur interactif
- Logique: ordre supérieur et types polymorphes
- Language: écrit en SML
- Sûreté: type [thm] abstrait

Une preuve facile

$$\dfrac{\dfrac{\dfrac{A \vdash A \qquad B \vdash B}{A, B \vdash A \wedge B}\wedge_i}{A \vdash B \Rightarrow (A \wedge B)}\Rightarrow_i}{\vdash A \Rightarrow (B \Rightarrow (A \wedge B))}\Rightarrow_i$$

```
(* forward proof *)
val th1 = ASSUME ``A:bool``;
val th2 = ASSUME ``B:bool``;
val th3 = CONJ th1 th2;
val th4 = DISCH ``B:bool`` th3;
val th5 = DISCH ``A:bool`` th4;
```

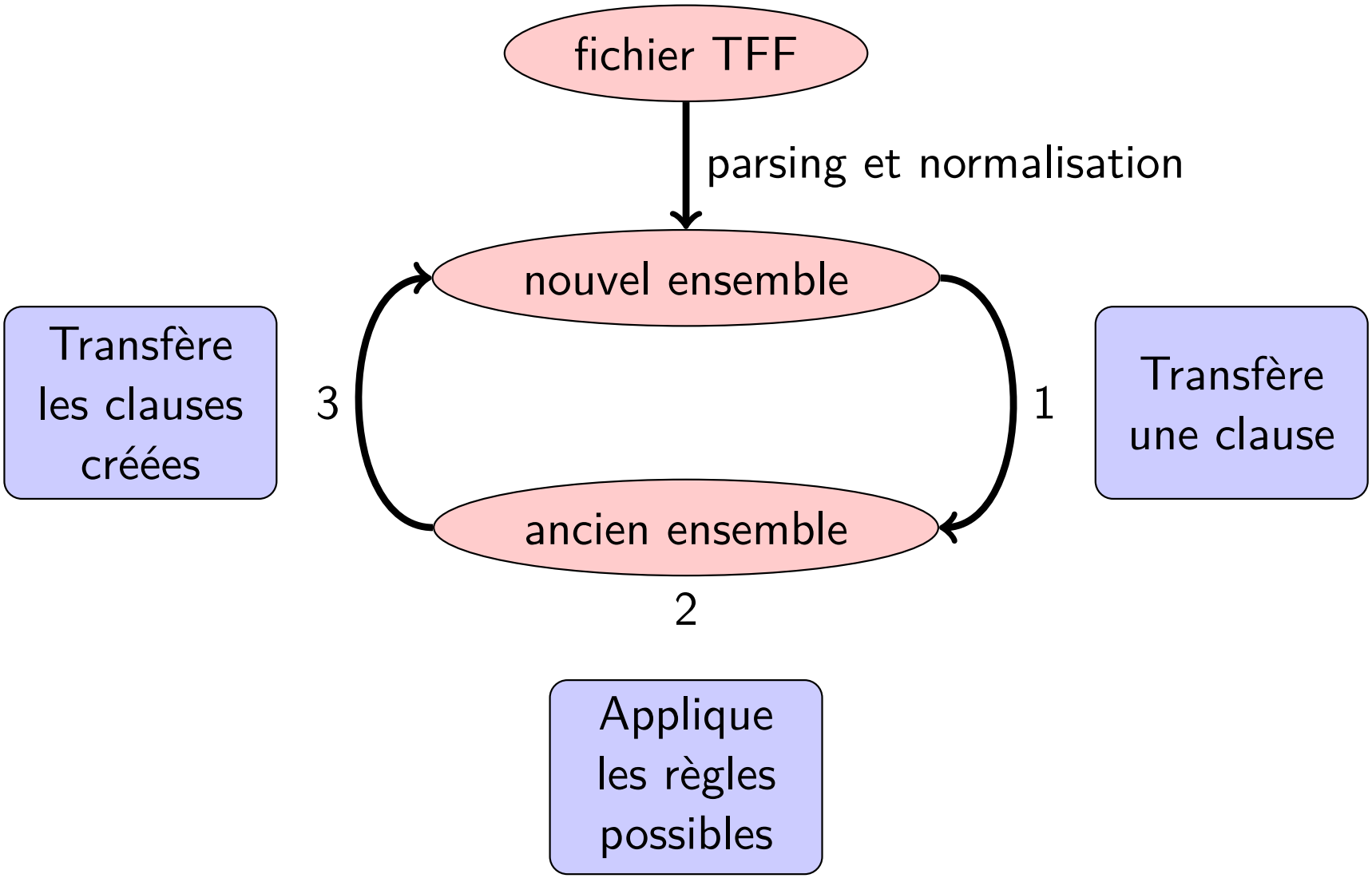
```
(* backward proof *)
g(`A ==> B ==> A /\ B `);
e(DISCH_TAC);
e(DISCH_TAC);
e(CONJ_TAC);
e(ACCEPT_TAC th1);
e(ACCEPT_TAC th2);
```

Beagle

Voici les caractéristiques de Beagle:

- Prouveur automatique
- Logique: premier ordre et types monomorphes
- Spécialité: arithmétique
- Réponse: insatisfaisable, satisfaisable ou inconnu

Une preuve par saturation



Ordre de la traduction vers le premier ordre

Voici les principales étapes de la traduction:

1. Monomorphisation
2. Négation de la conclusion (preuve par l'absurde)
3. Mise en forme normale conjonctive (utilisée plusieurs fois)
4. λ-lifting
5. Elimination des arguments de type booléen
6. Les naturels sont définis comme des entiers positifs
7. Mise sous forme d'un ensemble de clauses
8. Elimination de l'ordre supérieur

Motivation

Instanciation des types polymorphes de HOL4 par des types monomorphes.

Problème

Théorème: $\forall x : a.\ C\ x\ x$
Conjecture: $C\ 42\ 42$

Unification du type des constantes $C : a \to a \to bool$ et $C : num \to num \to bool$

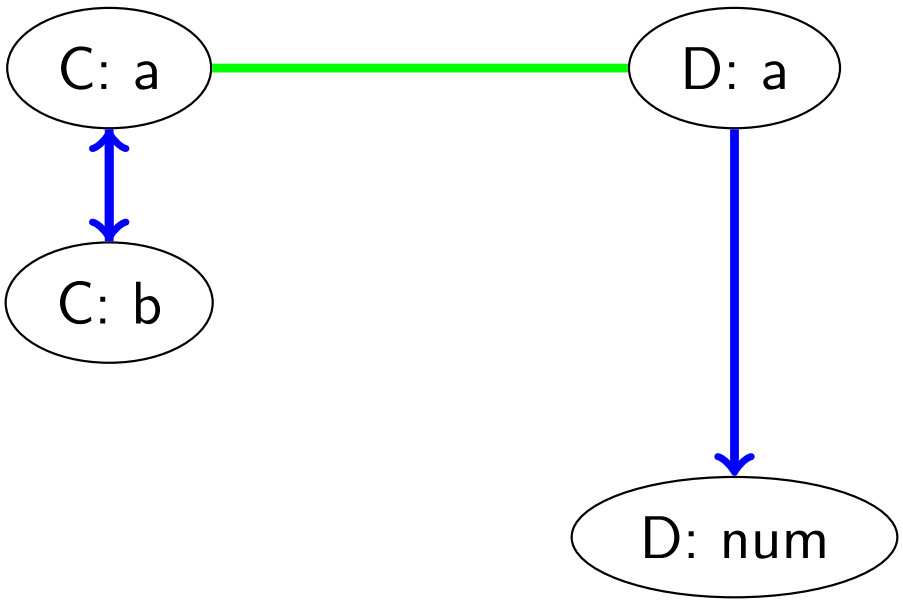
Nouveau problème

Théorème: $\forall x : num.\ C\ x\ x$
Conjecture: $C\ 42\ 42$

Graphe de dépendance

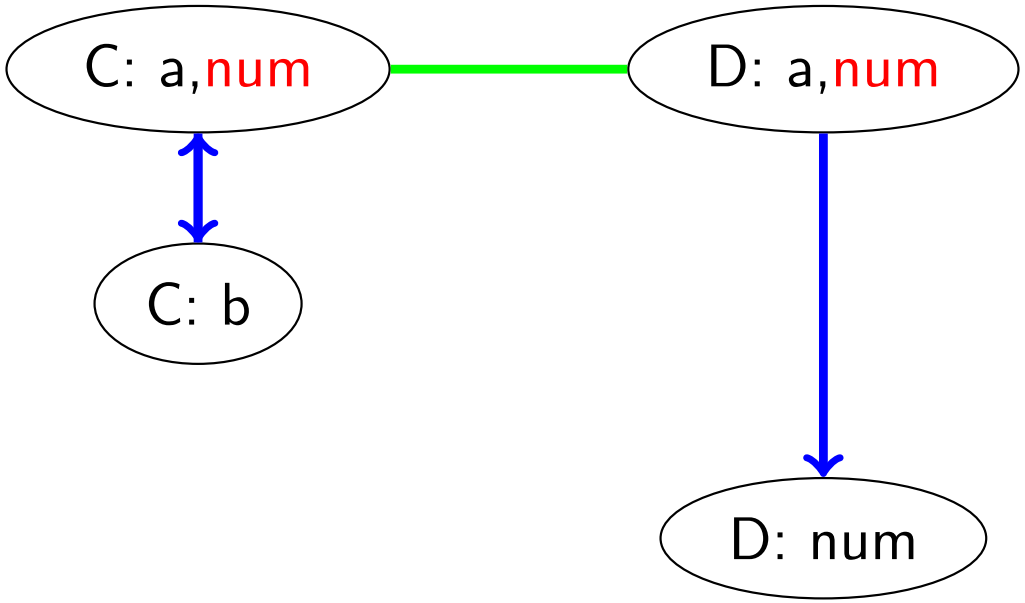
Problème

Théorème 1: $\forall x : a.\ C\ x \Rightarrow D\ x$
Théorème 2: $\forall x : b.\ C\ x$
Conjecture : $D\ 42$



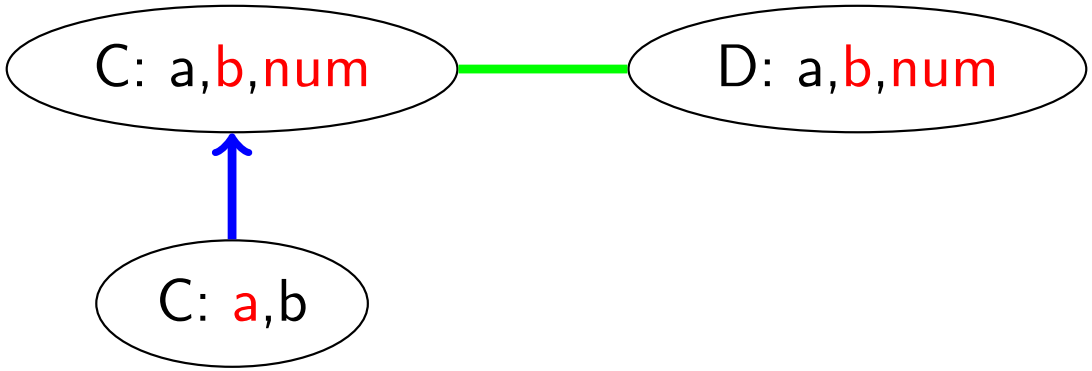
Example 1: Une co-instanciation

La flèche de substitution à droite induit une co-instanciation des constantes du premier théorème:



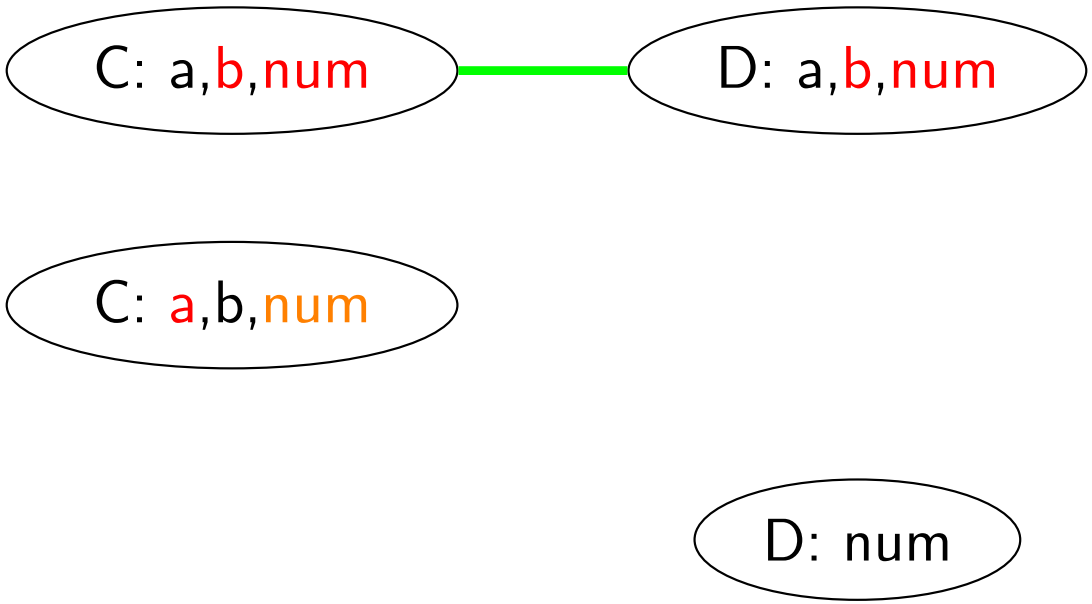
Example 1: Toutes les co-instanciations en parallèles

Nous nommerons $\mathcal{T}$ cette transformation du graphe.



Example 1: Répétition des co-instanciations

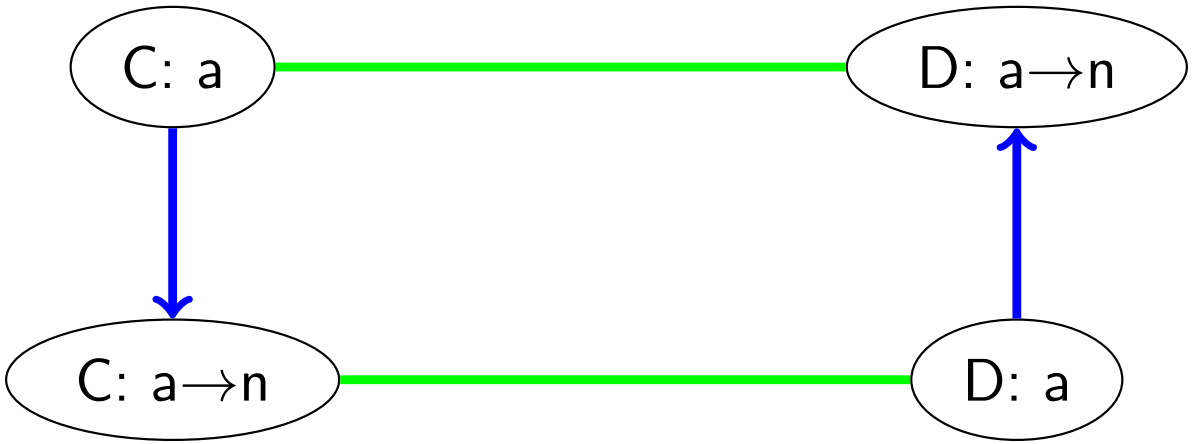
La transformation $\mathcal{T}$ précédente peut être répétée sur le nouveau graphe de dépendance que nous avons obtenu.



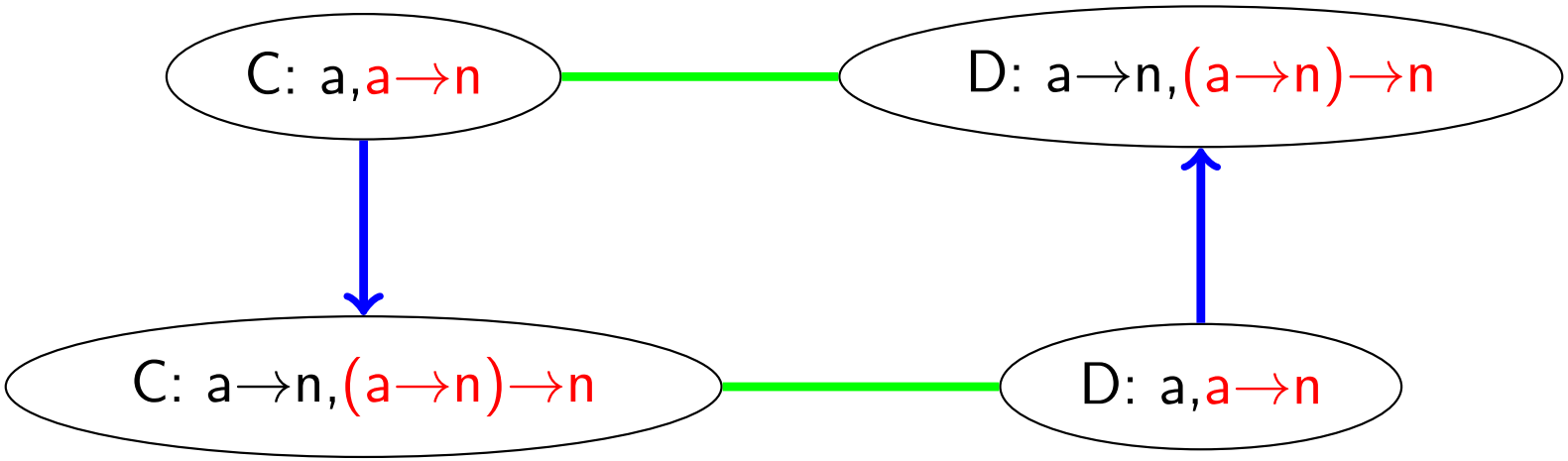
Ce graphe est un point fixe pour $\mathcal{T}$.

Example 2: Un example sans point fixe

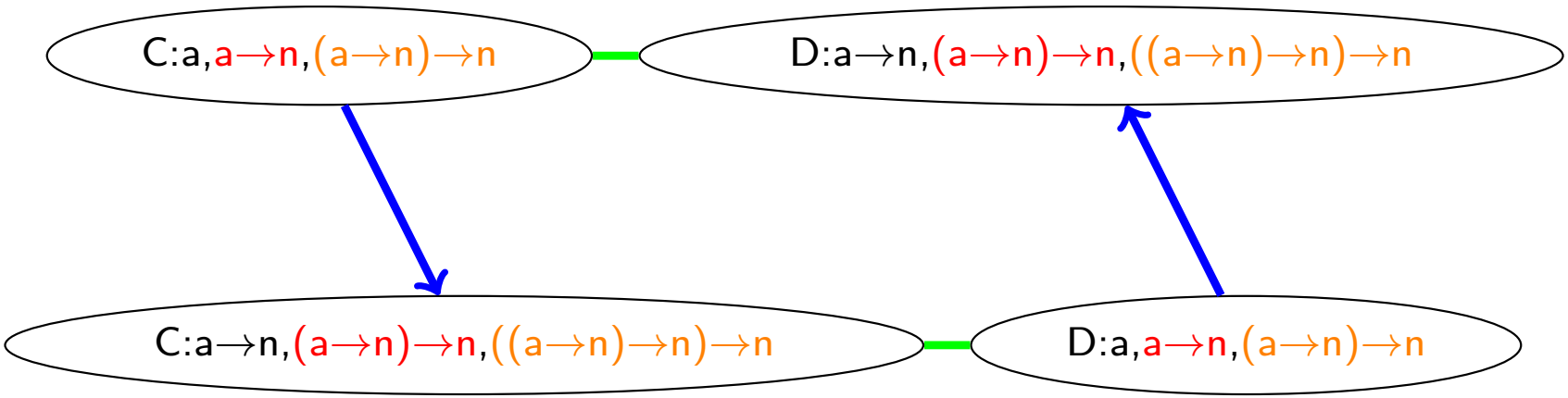
Recommençons avec un autre graphe.



Example 2: Un example sans point fixe



Example 2: Un example sans point fixe

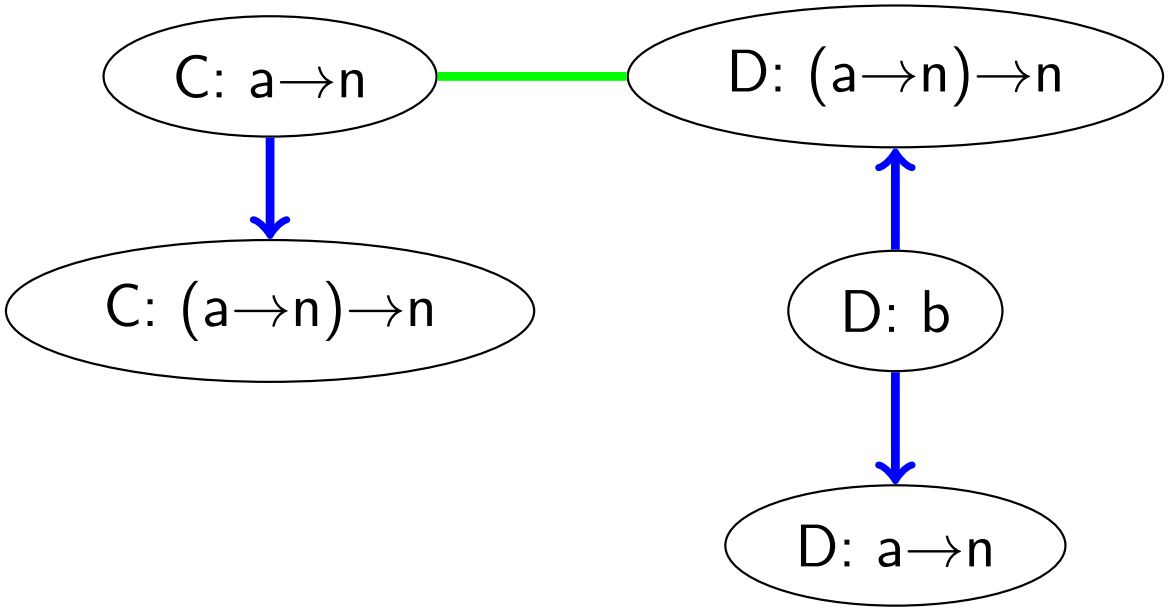


La transformation $\mathcal{T}$ ne trouve pas de point fixe.

Conjecture et contre-example

Conjecture (contre-example ci-dessous)

Soit G un graphe de dépendance. Soit $n > 1$ le nombre de noeuds de G. Si G n'a pas de circuit, alors un point fixe est obtenu en moins de $n - 1$ étapes.



Un théorème plus faible

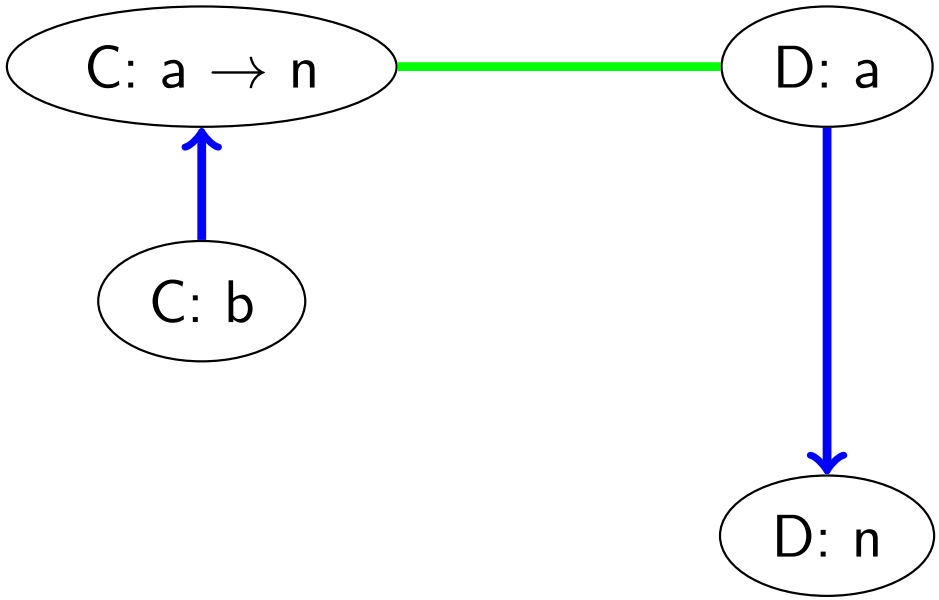
Théorème

Soit p la longueur du plus long chemin dans G. Soit $G_p = \mathcal{T}^g(G)$. Un point fixe pour G est atteint en p étapes, si les conditions suivantes sont réalisées:
- G_p n'a pas de circuit.
- SG_p n'a pas de flèches de substitutions.

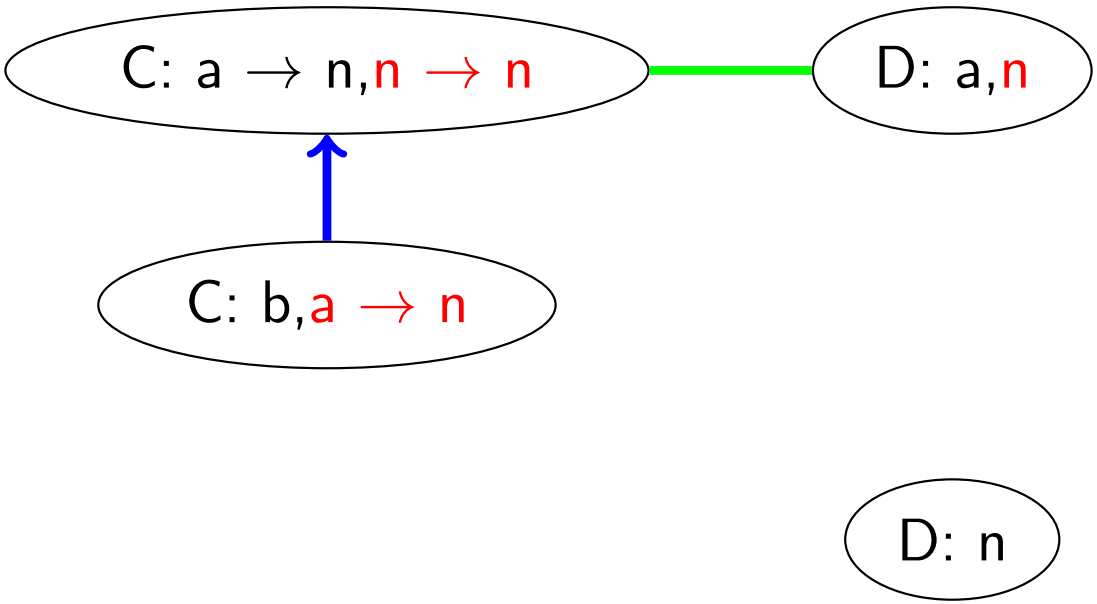
Conjecture

Soit p la longueur du plus long chemin dans G. Soit $G_p = \mathcal{T}^g(G)$. Si G_p n'est pas un point fixe alors, pour tout $i >= 0$, G_i n'est pas un point fixe.

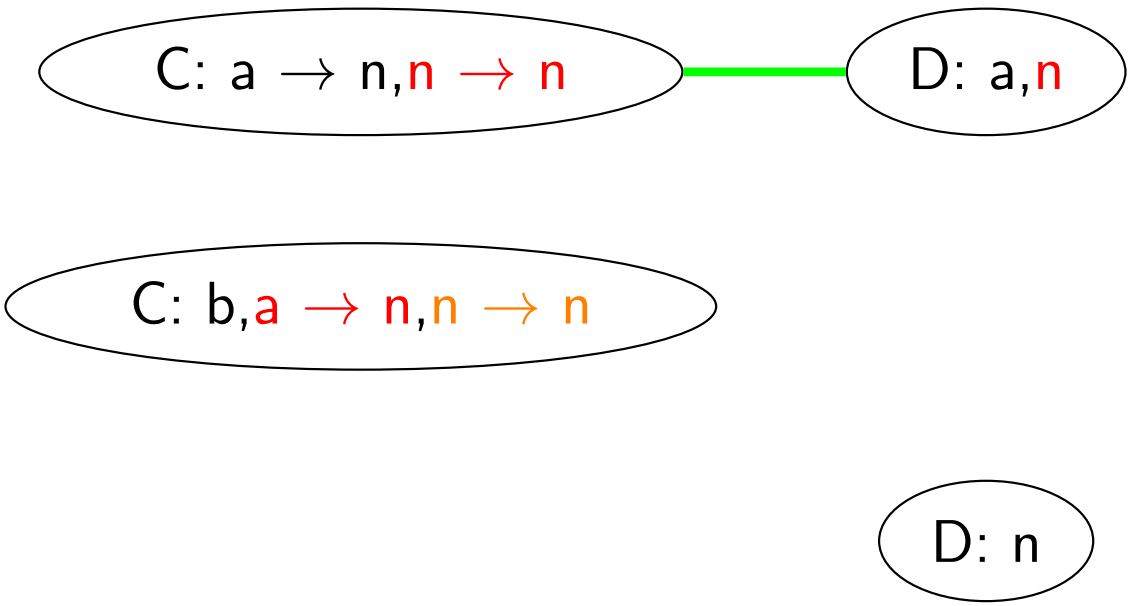
Example 3



Example 3



Example 3



Ce graphe est un point fixe pour $\mathcal{T}$.

Instanciation des théorèmes données par l'utilisateur

1. Terminaison: un point fixe ou une borne arbitraire
2. Extraction d'un ensemble de substitutions pour chaque théorème
3. Instanciation

Rêve

Si un point fixe est trouvé et une preuve polymorphe existe, alors il existe une fonction constructive de cette preuve vers une preuve monomorphe utilisant les théorèmes instanciés.

Ordre de la traduction vers le premier ordre

Voici les principales étapes de la traduction:

1. Monomorphisation
2. Négation de la conclusion (Preuve par l'absurde)
3. Mise en forme normale conjonctive (Arrive plusieurs fois)
4. λ-lifting
5. Elimination des arguments de type booléen
6. Les naturels sont définis comme des entiers positifs
7. Mise sous forme d'un ensemble de clauses
8. Elimination de l'ordre supérieur

λ-lifting

Cet étape élimine les λ-abstractions restantes *abs* dans une formule f. Le λ-lifting $\mathcal{L}$ est défini par:

$$\mathcal{L}(f[abs]) = (\forall x_1 \ldots x_n.\ g\ x_1 \ldots x_n = t) \Rightarrow f[abs := g]$$

Exemple: $\mathcal{L}(P\ \lambda x.x + 1) = (\forall x.\ g\ x = x + 1) \Rightarrow P\ g$

Elimination des arguments de type booléen

Soit t un argument de type *bool*. La conversion des arguments booléens $\mathcal{B}_a$ est définie par:

$$\mathcal{B}_a(f[t]) = (t \Rightarrow f[t := \mathit{true}]) \wedge (\neg t \Rightarrow f[t := \mathit{false}])$$

Exemple: $\mathcal{B}_a(P(a \wedge b)) = (a \wedge b \Rightarrow P\ \mathit{true}) \wedge (\neg(a \wedge b) \Rightarrow P\ \mathit{false})$

Elimination de l'ordre supérieur

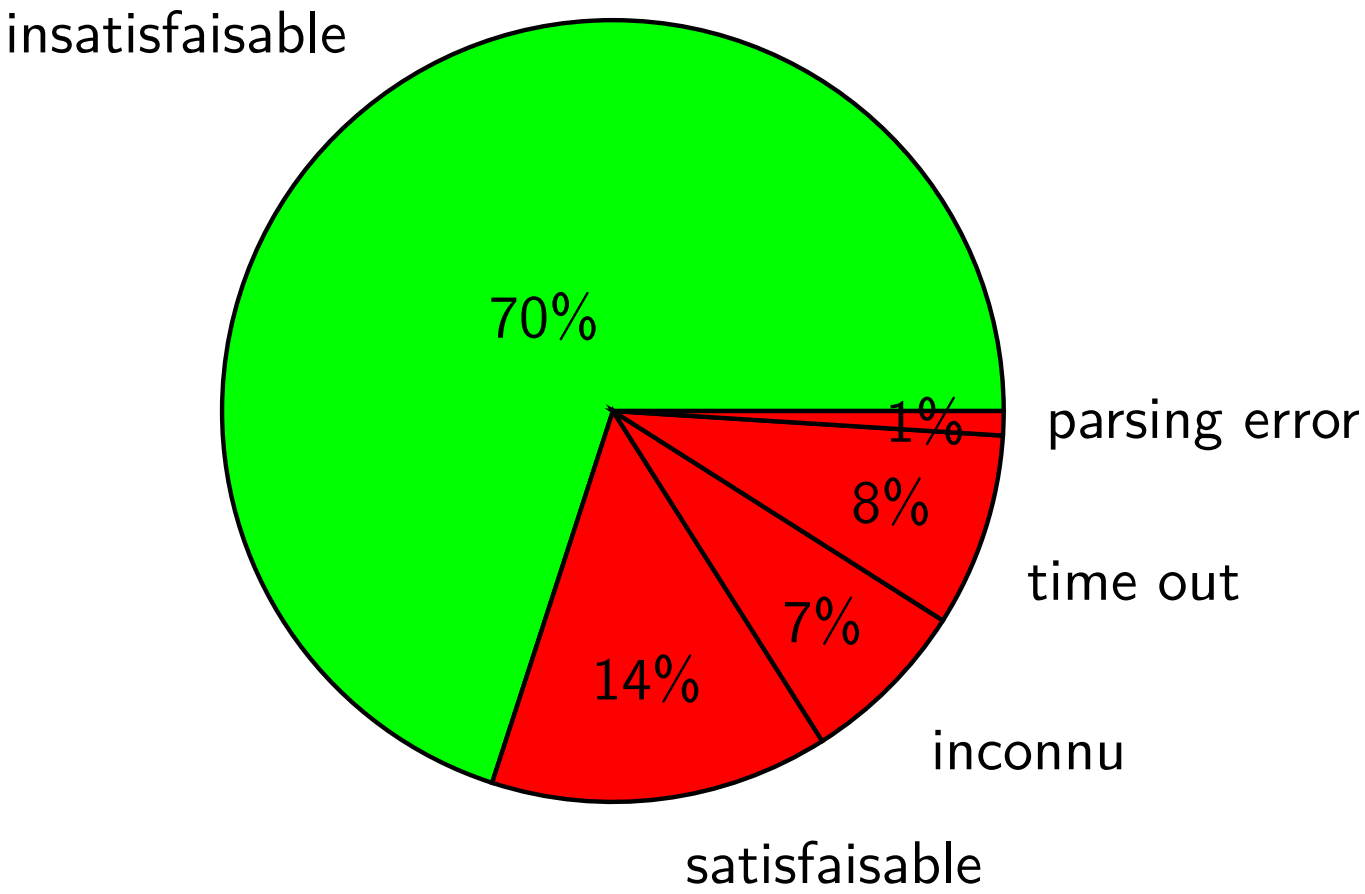
Soit c_a une constante arithmétique. Soit App une nouvelle variable, qui vérifie $App\ x\ y = x\ y$.
L'élimination de l'ordre supérieur $\mathcal{H}$ est definie récursivement par:

$$\mathcal{H}(x) = x$$
$$\mathcal{H}((c_a t)t_2) = (c_a \mathcal{H}(t))\mathcal{H}(t_2)$$
$$\mathcal{H}(t_1 t_2) = App\ \mathcal{H}(t_1)\mathcal{H}(t_2)$$

Exemple: $\mathcal{H}(\forall h.\ (h\ x) + 1 = 0) = \forall h.\ ((App\ h\ x) + 1 = 0)$

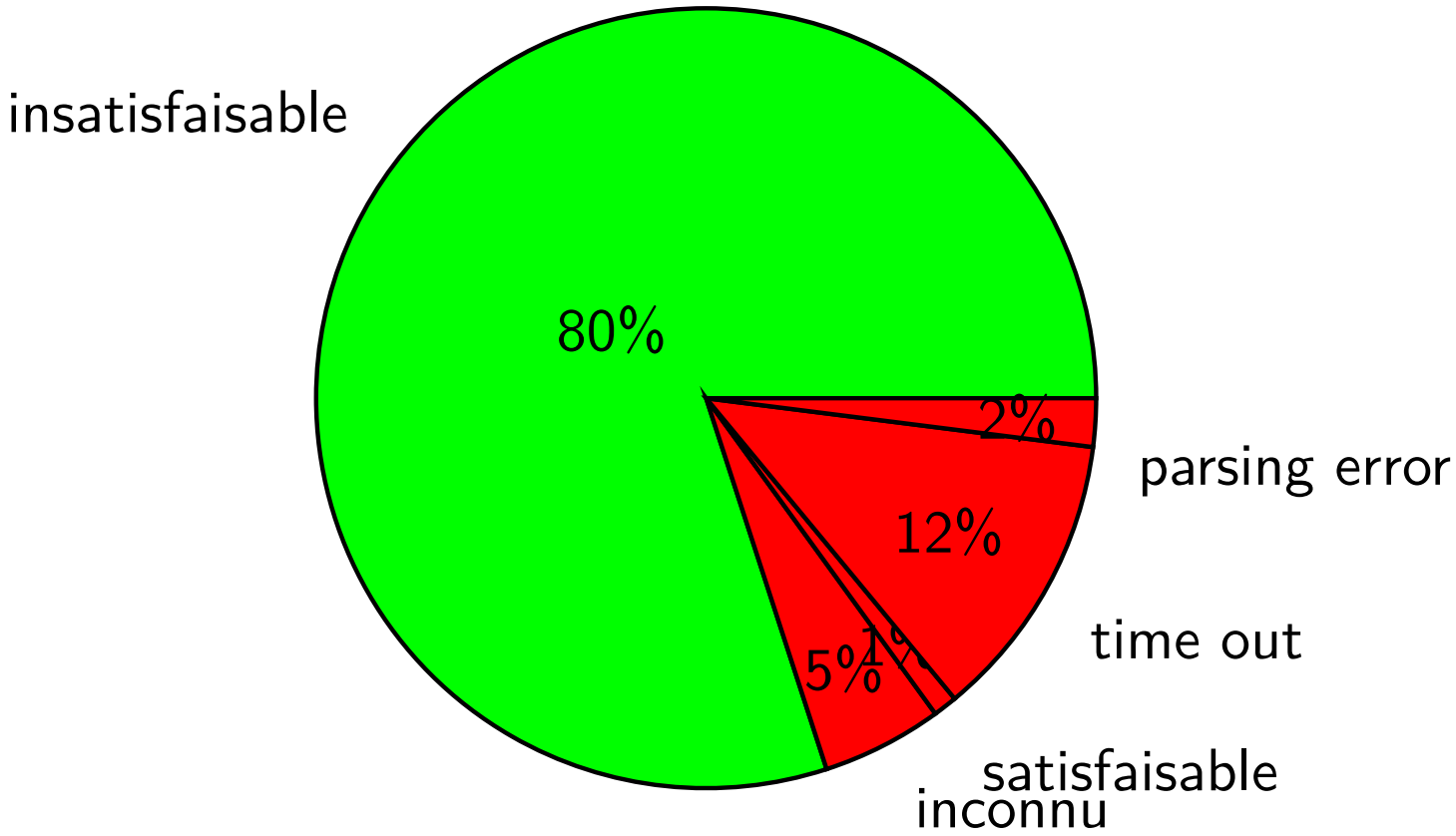
Résultats sans monomorphisation

Nombre total de problèmes: 271

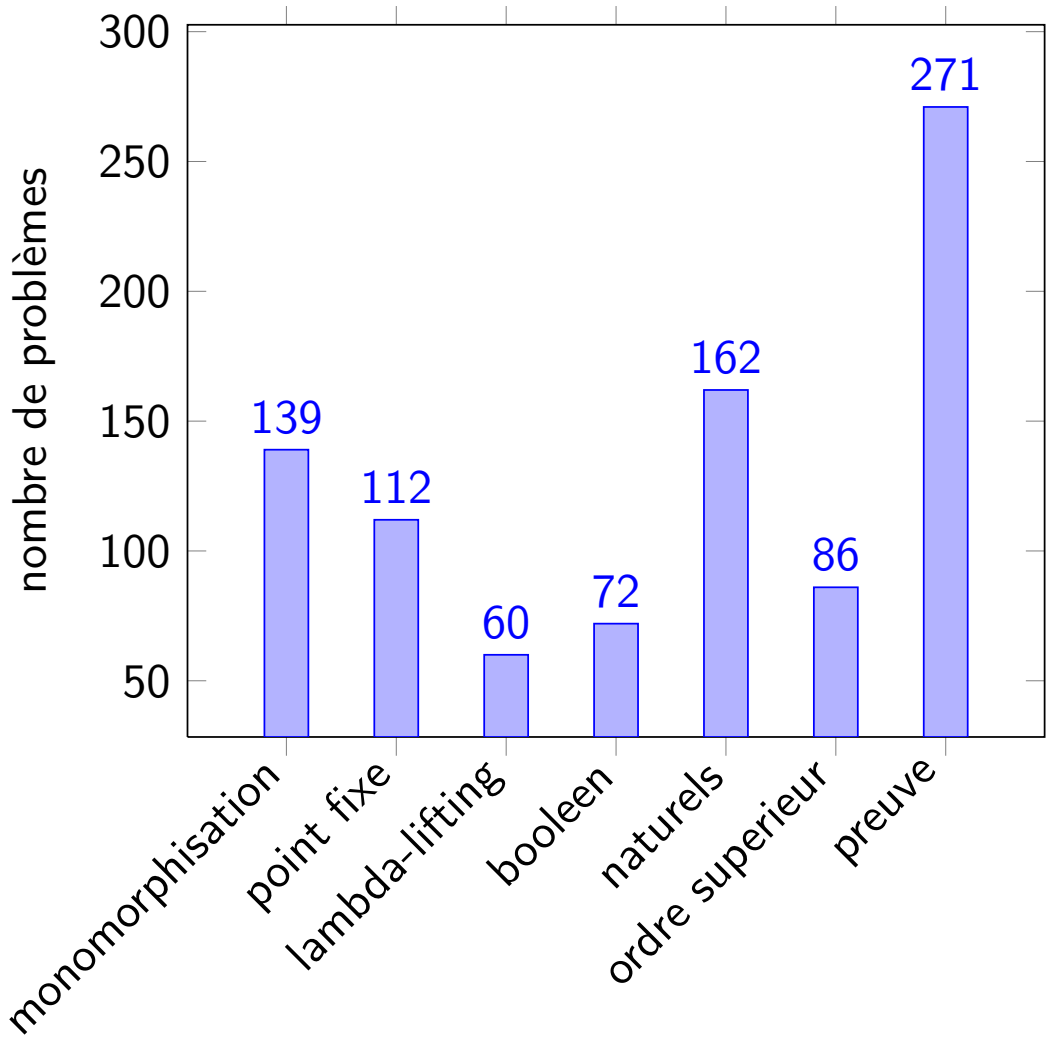


Résultats avec monomorphisation

Nombre total de problèmes: 271 (mêmes problèmes que précédemment)

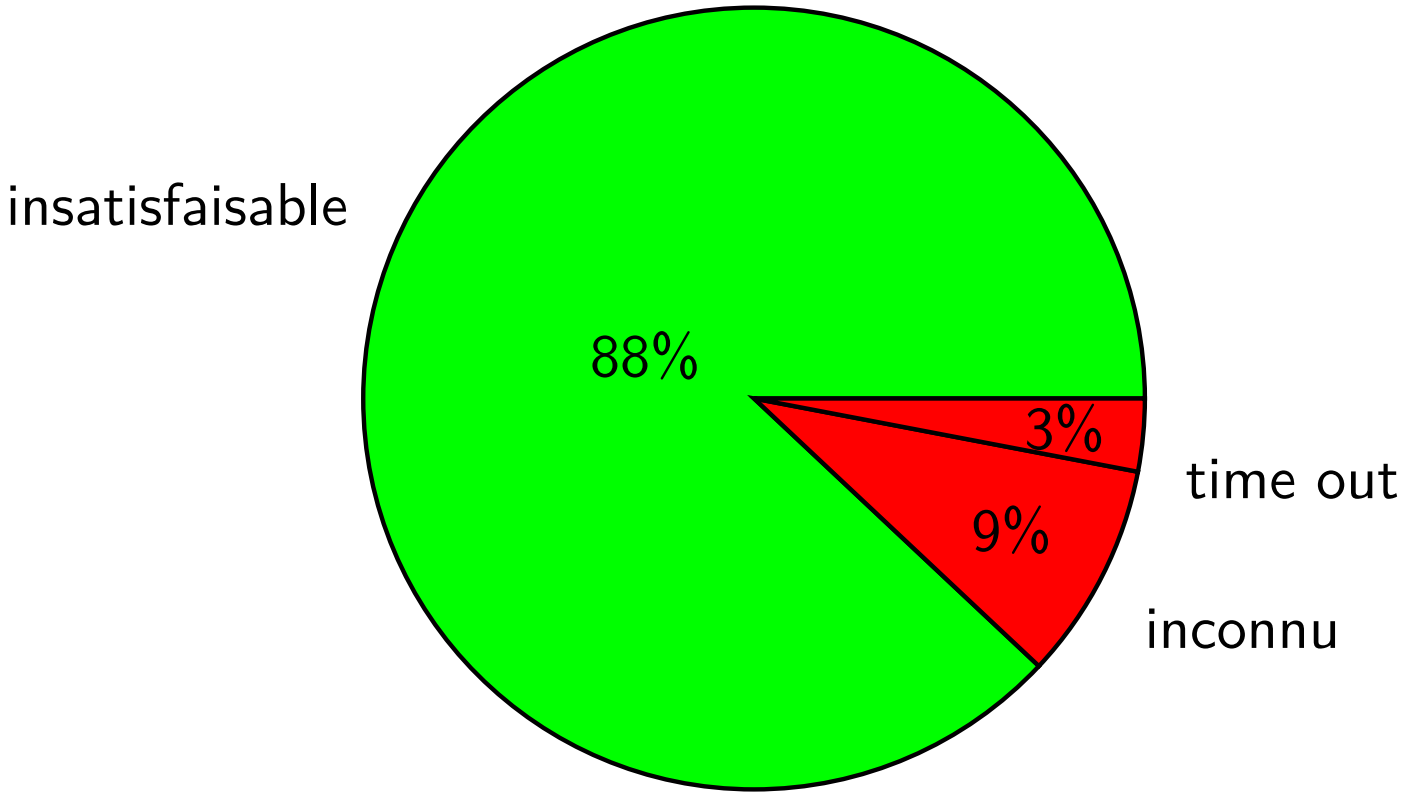


Utilisation des différentes parties de la traduction

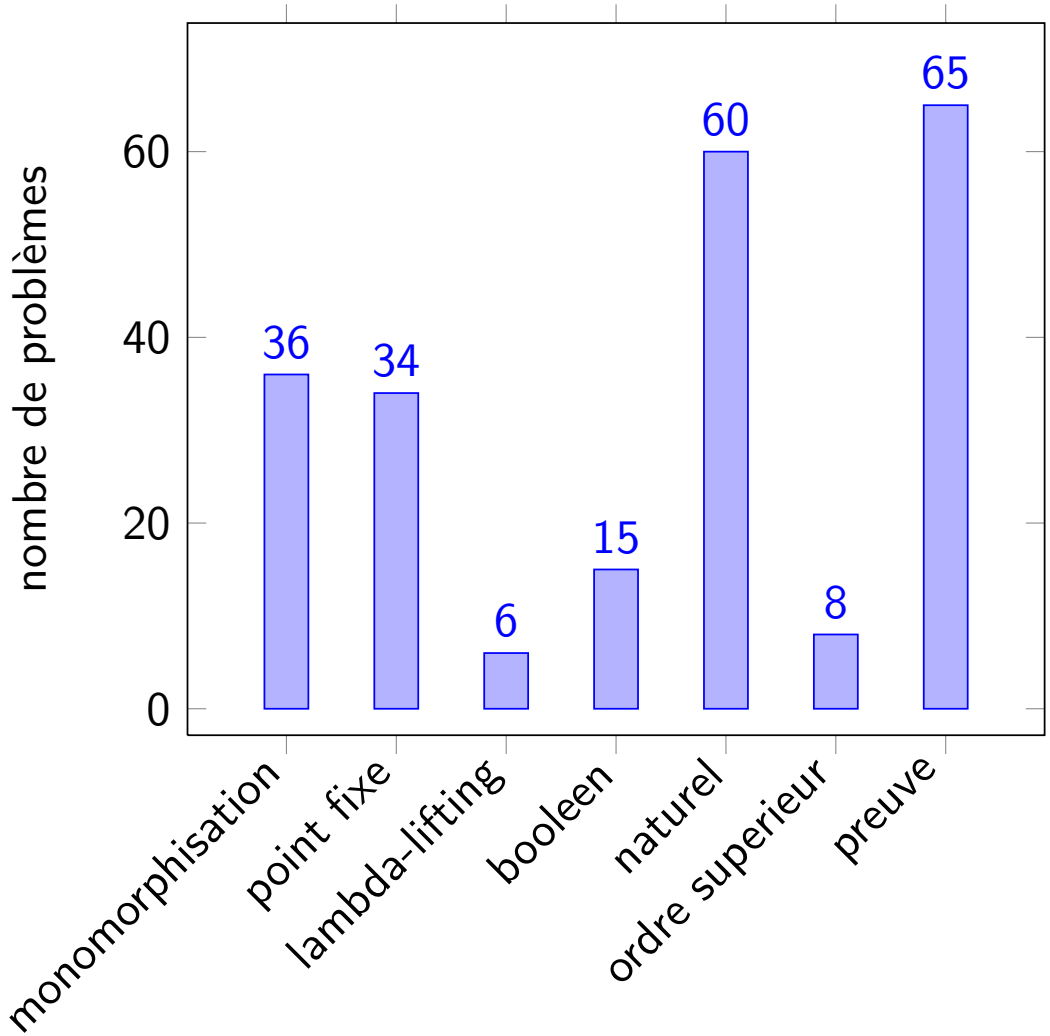


Résultats sur des problèmes arithmétiques

Nombre de problèmes : 65.
Dans ces problèmes, tous les théorèmes (79) ne concernant que l'arithmétique ont été effacés.



Utilisation des différentes parties de la traduction



Quelques améliorations possibles de la traduction

Trois idées d'amélioration:

- Générer automatiquement des théorèmes aidant à prouver la conjecture
- Eliminer l'ordre supérieur plus efficacement
- Traduire les entiers, les rationnels et les réels

Résumé des qualités et des limites de l'interaction HOL4-BEAGLE

Qualités:

- Résout des problèmes arithmétiques sans guidage
- Un format de communication répandu
- Une traduction correcte (préservant l'insatisfaisabilité)

Limites:

- Une preuve non rejouée (BEAGLE_TAC est un oracle)
- Une traduction incomplète
- Une traduction ne préservant pas la satisfaisabilité (Impossible de générer des contre-exemples)

Quelques questions?

Merci d'avoir écouté ma présentation. J'ai beaucoup apprécié cette année passée au LMFI.