TP numéro 5

Intelligence artificielle, ensIIE

Semestre 4, 2025–26

L'objectif de ce TP est de développer un programme en ProLog permettant de gérer des paquets, comme les gestionnaires de paquets sous linux, ou les outils comme pip, npm, opam, etc. Chaque paquet possède une version et un ensemble de dépendances et de conflits. Le but est de trouver quels paquets doivent être installés pour pouvoir en installer un de façon à ce que ses dépendances soient installées et qu'il n'y ait pas de conflit.

Votre code devra être déposé avant le 17 mai 2026 (inclus) sur `exam.ensiie.fr` dans le dépôt `ia-fisa-tp5`.

1 Représentation des paquets

Un paquet sera donné par un triplet

```
package(Name, Version, Constraints)
```

où :

- **`Name`** est le nom du paquet.
- `Version` est son numéro de version.
- `Constraints` est une liste de contraintes qui sont soient des dépendances, soient des conflits.

Une dépendance sera donnée par un couple

```
d(Name, Inequations)
```

où :

- **`Name`** est le nom d'un paquet
- `Inequation` est une liste d'inéquations et d'équations qui doivent être satisfaite par la version du paquet **`Name`** installé.

Un conflit sera donné par un couple

```
c(Name, Inequations)
```

où :

- **`Name`** est le nom d'un paquet
- `Inequation` est une liste d'inéquations et d'équations qui donne lieu au conflit si la version du paquet **`Name`** installé le satisfait.

Les inéquations et équations seront de la forme

```
leq(Ver)
lt(Ver)
geq(Ver)
gt(Ver)
neq(Ver)
eq(Ver)
```

pour respectivement $\leq$ $<$ $\geq$ $>$ $\neq$ $=$.

On considérera aussi des listes de paquets installés ou à installer, qui contiendront des éléments de la forme `p(Name, Version)`.

Vous trouverez dans le fichier `base.pl` un ensemble de paquets prédéfini.

2 Installation d'un paquet

1. Définir un prédicat `satisfy_equations(V, L)` qui est vrai ssi la version `V` satisfait les inéquations et équations de la liste `L`.
2. Définir un prédicat `is_installed_under_equations(N, C, I)` qui est vrai ssi il y a un paquet `p(N, V)` dans la liste `I` telle que `V` vérifie les inéquations dans la liste `C`.
3. Définir un prédicat `update(I, T, R)` tel que :
   - `I` est une liste de paquets installés ;
   - `T` est une liste de paquets à installer ;
   - `R` est obtenu en remplaçant les paquets de `I` par les paquets de `T` du même nom, à condition que la version dans `T` soit plus grande que dans `I` ; et en ajoutant les paquets de `T` dont le nom n'est pas dans `I`. Si pour un nom donné, la version dans `T` est plus petite que celle dans `I`, le prédicat sera faux.
4. Définir un prédicat `check_conflict(L, I)` qui vérifie qu'aucun des paquets donnés dans `L` n'a de conflit avec les paquets de `I`.
5. Définir un prédicat `install(Name, Eq, I, R)` tel que
   - `Name` est le nom d'un paquet que l'on veut installer ;
   - `Eq` est une liste d'inéquations et d'équation que doit satisfaire la version du paquet à installer ; en général, au départ cette liste sera vide ;
   - `I` est une liste de paquets déjà installés ;
   - `R` est la liste des paquets à installer pour que `Name` et ses dépendances récursives soient installés, sans qu'il n'y ait de conflit.

3 Mise-à-jour

6. Définir un prédicat `upgrade(I, N)` tel que `I` est une liste de paquets installés et `N` est la liste des paquets dont il existe une version plus grande et qui peuvent être installés, avec leurs nouvelles dépendances si besoin, et en s'assurant qu'il n'y a pas de conflit.

4 Recommandations

On ajoute une nouvelle contrainte de la forme `r(Name, Inequations)` qui indique qu'un paquet recommande l'installation d'un autre, sans que cela soit strictement nécessaire.

7. Définir un prédicat `install_with_recom(Name, Eq, I, R)` qui fonctionne comme `install(Name, Eq, I, R)` mais en essayant d'installer les paquets recommandés si c'est possible sans conflit. Les recommandations seront satisfaites récursivement. Si une des recommandations ne peut pas être installée sans conflit, elle ne le sera pas.
8. Modifier la base de paquets pour pouvoir tester de façon appropriée `install_with_recom`.