# IA1 : Système Expert

Théo Dézé & Charles Mallet

9 Novembre 2018

## Table des matières

**I Introduction** **2**

**1 Faits** **2**

**2 Règles** **2**

**3 Choix du langage** **3**

**II Système Expert** **3**

**4 Base de connaissances** **3**
4.1 Fichier . . . . . 4
4.2 Interface . . . . . 4

**5 Moteur d'inférences** **4**

**6 Interface utilisateur** **4**
6.1 Ligne de commandes . . . . . 5
6.2 Graphique . . . . . 6

**III Conclusion** **6**

**7 Améliorations possibles** **6**

**8 Code source du projet** **7**

## Première partie

# Introduction

Pour le projet, nous avons décidé de travailler sur un problème de diagnostique médical. Nous avons défini différentes maladies qui sont à "vrai" si le patient les a.

## 1 Faits

Nos faits sont de trois natures. Soit il s'agit d'informations sur les symptômes du patient, soit ils portent sur ses attributs ou sur les maladies qu'il a.

Comme nous sommes sur un moteur d'inférences 0+, nos faits peuvent valoir une valeur booléenne, une chaîne de caractères ou un chiffre (pas dans ce cas mais on peut les tester dans le cas Météorologie [1]).

**Exemple**

**Symptômes** La zone des douleurs (Poitrine, Gorge, Abdomen, Aucun) ; le patient a de la fièvre, de la toux ou des vomissements.

**Attributs** Le sexe de la personne.

**Maladies** Maladie que le patient peut avoir (Rhume, Infarctus, Appendicite, etc).

## 2 Règles

Nous avons organisé notre choix des règles selon un arbre décisionnel (voir figure 1). Cela correspond à des questions que pourraient poser un médecin à un patient pour déterminer la maladie du patient.

Il est possible que le patient ait plusieurs maladies. Cela est dû au fait que notre arbre est très simplifié et que nous n'avons pas implémenté le chaînage mixte.

Ce dernier point oblige l'utilisateur à fournir toutes les informations. Au contraire, si on avait eu un chaînage mixte, on aurait pu demander au patient seulement ce qui est utile.

1. Disponible dans le répertoire docs du projet.

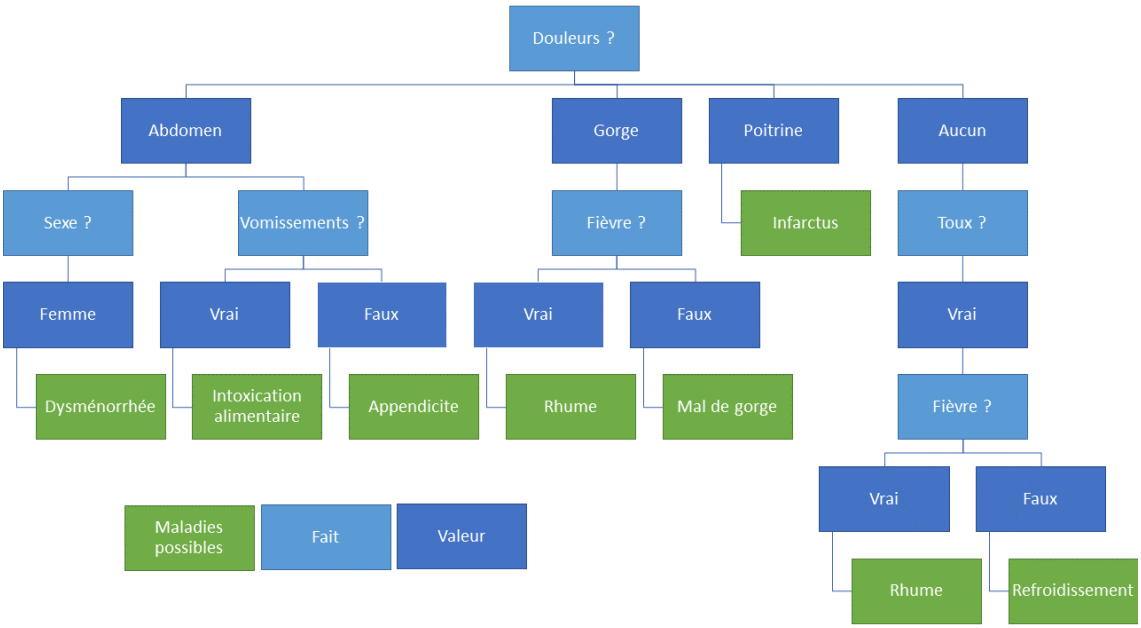


FIGURE 1 – Arbre décisionnel

## 3 Choix du langage

Nous avons choisi de partir sur du python car il présente l'avantage d'être fortement typé mais également d'être dynamique ce qui nous permet facilement de gérer les valeurs qui sont de natures différentes.

Pour l'interface, nous nous sommes servis du package pyside2 qui permet d'utiliser, en python, Qt que nous avons déjà vus en C++, et qui en est le package officiel.

# Deuxième partie
# Système Expert

## 4 Base de connaissances

Notre base de connaissances est constituée de la base de faits et la base de règles qui correspondent à des tableaux de faits et de règles.

Elle peut être enrichie à l'aide de fichiers ou bien de l'interface.

### 4.1 Fichier

Le fichier permet de déclarer des faits et des règles (voir en-dessous). On ne peut avoir qu'un fait ou une règle par ligne mais aussi écrire une ligne de commentaires avec le symbole '#'.

Pour ce qui est des propositions, on peut aussi ajouter des ou (exprimés ainsi '||') et des parenthèses. Les parenthèses nous ont posé un problème. En effet, nous avons été obligés de convertir nos expressions in-fixe en post-fixe pour pouvoir les calculer.

```
Douleur = "Gorge"
Fièvre = Vrai
Sexe = "Homme"

Appendicite = Vrai := Douleur == "Abdomen" & Vomissements == Faux
Intoxication alimentaire = Vrai := Douleur == "Abdomen" & Vomissements == Vrai
Dysménorrhée  = Vrai := Douleur == "Abdomen" & Sexe == "Femme"
Rhume = Vrai := Douleur == "Gorge" & Fièvre == Vrai
Infarctus = Vrai := Douleur == "Poitrine"
Mal de gorge = Vrai := Douleur == "Gorge" & Fièvre == Faux
Rhume = Vrai := Douleur == "Aucun" & Toux == Vrai & Fièvre == Vrai
Refroidissement = Vrai := Douleur == "Aucun" & Toux == Vrai & Fièvre == Faux
```

### 4.2 Interface

Selon l'interface, plusieurs méthodes sont disponibles seulement, la plus simple est de taper une ligne comme vous pouvez le faire dans le fichier et de valider.

## 5 Moteur d'inférences

Nous avons implémenté le chaînage avant avec but ou, juste pour sature, la base de connaissances et le chaînage arrière.

À la fin de l'exécution, il affiche un résultat et le cheminement selon la configuration (changeable). Il est possible de voir le log en tapent la commande "log" et pour lire toutes les commandes disponibles, il suffit de taper "aide".

## 6 Interface utilisateur

Nous avons développé deux interfaces, une en ligne de commandes et une graphique.

## 6.1 Ligne de commandes

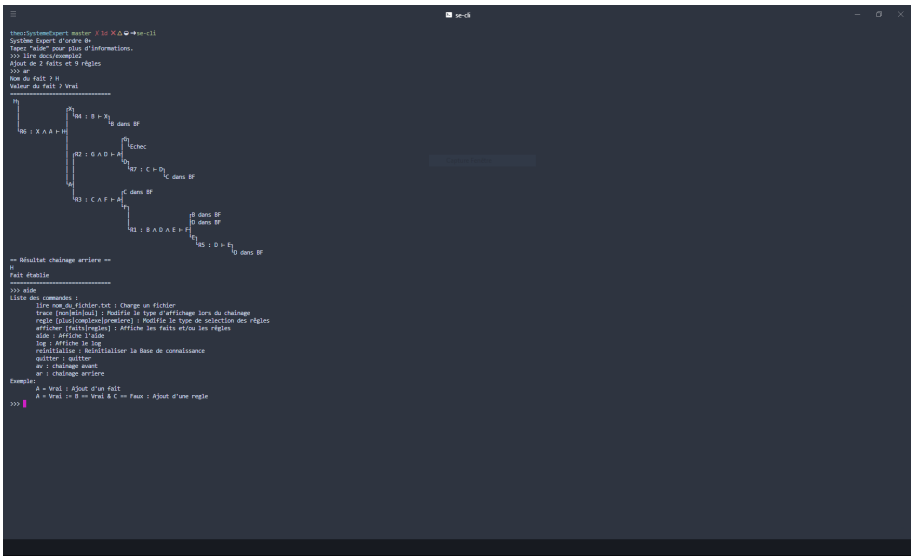


FIGURE 2 – SE-CLI

L'interface une ligne de commandes s'inspire de l'interpréteur de python. Elle permet de faire tout ce que propose l'interface graphique. Comme l'interpréteur de python, il lit chaque ligne comme si c'était une ligne d'un fichier.

## 6.2 Graphique

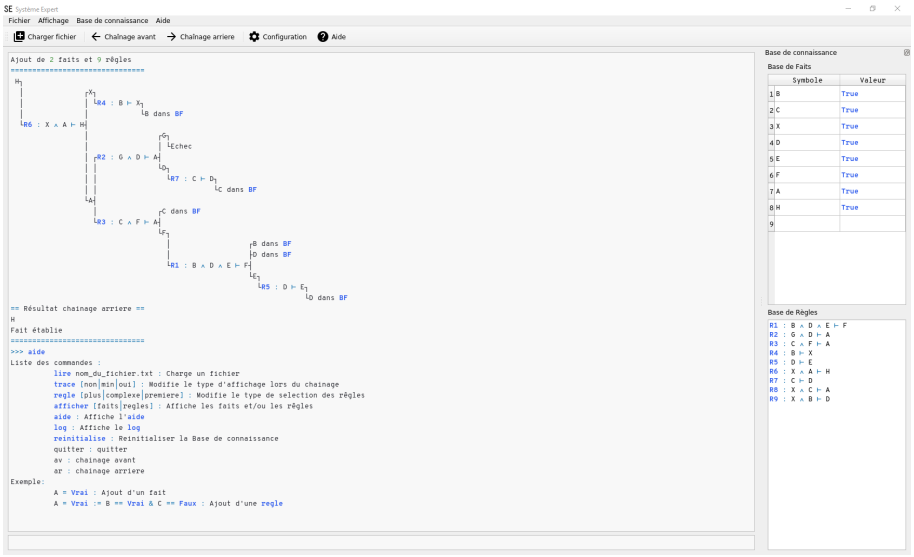


FIGURE 3 – SE-GUI

L'interface permet de simplifier l'utilisation pour un utilisateur qui n'est pas habitué à manipuler un terminal.

La plupart des commandes peuvent être exécutées à l'aide de boutons. Bien sûr, il est toujours possible de taper les commandes comme sur la version en ligne de commandes.

### Composition de l'interface

**Terminal** Il est constitué d'un affichage et d'une ligne de saisie. Il s'utilise exactement comme la version en ligne de commandes.

**Dock** Il permet d'afficher la base de connaissances et d'ajouter/de modifier un fait.

**Barre d'outils** Elle donne accès aux commandes importantes.

# Troisième partie
# Conclusion

## 7 Améliorations possibles

Voici une liste des améliorations possibles :

— Ajouter des coefficients de certitude, pour représenter des événements incertains.

— Représenter les faits sous la forme de n-uplet pour faciliter les regroupements des faits. Exemple : AgeBob = 25 devient (Bob,age,25) ou ParentsBob = "Léo Léa" devient (Bob,parents,Léo,Léa).
— Transformer le moteur 0+ en 1.

# 8 Code source du projet

Toutes les sources du projet et ce document sont disponibles sur https://github.com/theodeze/SystemeExpert

De plus, le projet est disponible sur pypi à l'adresse https://pypi.org/project/sysexpert. Il suffit de taper cette commande pour le télécharger.

```
pip install sysexpert
se-cli # Interface en ligne de commandes
se-gui # Interface graphique
```