

# IA01 : Intelligence Artificielle - Représentation des connaissances

TP3 : Système expert

ASSAF Nora & GAJAN Antoine



# Table des matières

<b>Introduction</b>	<b>2</b>
<b>1 Expertise</b>	<b>3</b>
1.1 Comment définir une personne ? . . . . .	3
1.1.1 Tranche d'âge . . . . .	3
1.1.2 Lien avec la personne . . . . .	3
1.1.3 Goûts de la personne . . . . .	4
1.1.4 Caractère de la personne . . . . .	4
1.2 Base de connaissances . . . . .	5
1.3 Bases de règles . . . . .	7
1.4 Bases de questions . . . . .	7
1.5 Moteur d'inférence . . . . .	8
<b>2 Programmation du système expert</b>	<b>10</b>
2.1 Chaînage avant . . . . .	10
2.2 Chaînage arrière . . . . .	10
2.3 Fonction main . . . . .	11
2.4 Fonctions de service . . . . .	12
2.4.1 Fonctions relatives aux règles . . . . .	13
2.4.2 Fonctions relatives à la base de faits . . . . .	13
2.4.3 Fonctions relatives aux cadeaux . . . . .	13
2.4.4 Fonctions relatives aux questions . . . . .	14
2.5 Exemples de fonctionnement du système expert . . . . .	15
2.5.1 Chainage avant . . . . .	15
2.5.2 Chainage arrière . . . . .	15
<b>Conclusion</b>	<b>17</b>
2.6 Difficultés rencontrées et limites de notre projet . . . . .	17
2.7 Bilan et conclusion . . . . .	18
<b>3 Annexes</b>	<b>19</b>

## Introduction

Ce TP a pour objectif de mobiliser les compétences vues en cours magistraux et en travaux dirigés afin de concevoir un système expert à l'ordre 0+.

Nous avons choisi le sujet : "Quel est le meilleur cadeau de Noël ?". En effet, réalisé pendant les périodes de fêtes, ce TP était pour nous l'occasion de mêler l'utile à l'agréable. Nombreuses sont les personnes qui se demandent quel cadeau offrir le 25 décembre, dont nous-même. En 2014, 65,7 % des Français déclaraient être satisfaits de leurs cadeaux de Noël. Ce système expert nous permettrait de faire monter ce pourcentage et est aussi l'occasion pour nous de mêler ingénierie et vie personnelle.

Notre système expert proposera deux types de recherche. La première, consistant en un chaînage avant, permettra de déterminer quel est le meilleur cadeau de Noël à offrir en fonction des caractéristiques de la personne qui va se faire offrir le cadeau. La deuxième, consistant en un chaînage arrière, déterminera si le cadeau que vous voulez offrir correspond bien aux goûts de la personne.

Nous évoquerons dans un premier temps l'élément indispensable à notre système expert : l'expertise. Ensuite, nous aborderons la programmation de ce système expert ainsi que les limites que nous avons rencontrées au cours du projet.

Ce rapport s'efforcera de présenter la démarche entreprise afin de répondre au problème.



FIGURE 1 – Cadeaux de Noël

# 1 Expertise

## 1.1 Comment définir une personne ?

La première chose à faire est de définir une personne, ses caractéristiques et ses goûts.

Nous avons choisi dans un premier temps de définir une personne en fonction de ses caractéristiques générales. L'âge, le lien avec la personne à qui elle souhaite offrir un cadeau, ainsi que ses goûts et son caractère seront étudiés.

### 1.1.1 Tranche d'âge

En fonction de son âge, une personne peut appartenir à différentes catégories :

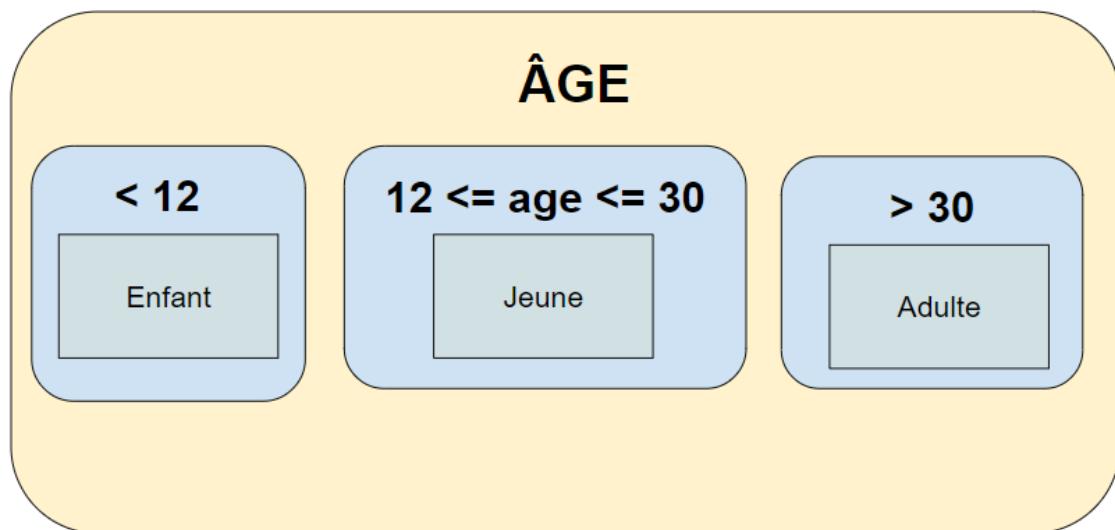


FIGURE 2 – Tranche d'âge pour notre système expert

### 1.1.2 Lien avec la personne

Il est important de prendre en considération le lien avec la personne. En effet, on n'offrira pas le même cadeau à son partenaire qu'à un membre de sa famille. Ainsi, la personne à qui on offre le cadeau peut être :

- un(e) ami(e)
- de la famille
- son/sa compagnon/compagne

### 1.1.3 Goûts de la personne

Pour les goûts, nous avons défini plusieurs catégories : les décos, la cuisine, la musique, les jeux, la technologie, le sport, la mode et la télévision. Chacune de ses catégories peut se spécialiser comme suit :

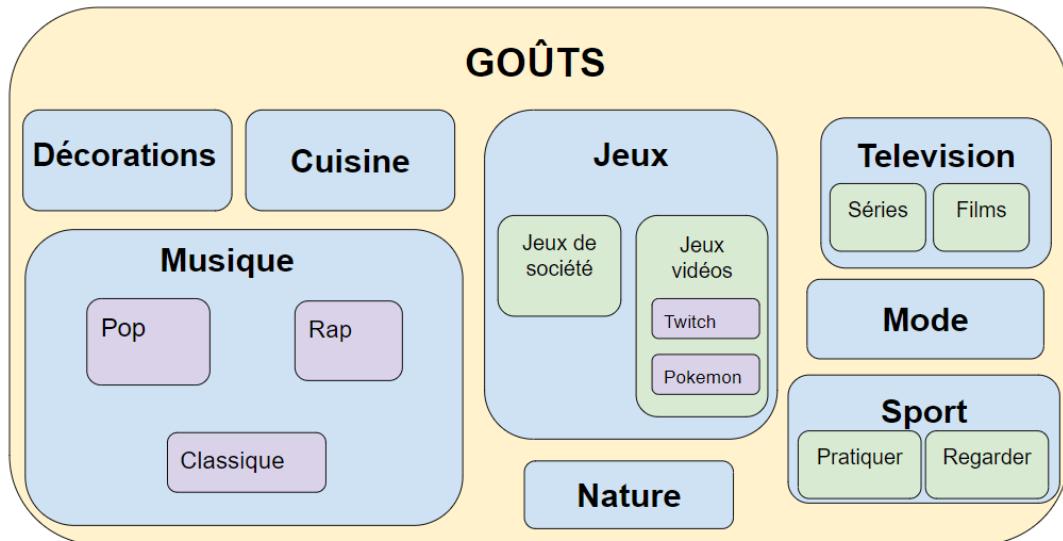


FIGURE 3 – Schéma des goûts possibles d'une personne

### 1.1.4 Caractère de la personne

Le dernier élément permettant de décrire une personne est son caractère. Ainsi, nous avons décidé qu'une personne peut être :

- tête en l'air
- gourmande
- nostalgique
- aventurière
- artistique / sensible à l'art
- casanière
- créative
- curieux

Afin de représenter une personne, ses caractéristiques auront le formalisme suivant dans la base de faits de notre système expert :

```
( variable valeur )
```

Par exemple, si la personne est gourmande, alors on aura dans la base de faits

---

l'information (gourmand oui). Si la relation la liant avec la personne est une amitié, alors on aura (relation ami).

## 1.2 Base de connaissances

Pour établir une liste de cadeaux possibles, nous avons fait des recherches sur les cadeaux les plus offerts au cours des 10 dernières années.

Les liens utiles nous ayant permis de choisir les cadeaux à traiter sont présentés dans la bibliographie.

En nous basant sur les tendances actuelles et nos idées personnelles, nous avons ainsi pu établir une liste d'une cinquantaine de cadeaux. On retrouvera dans cette liste des places de concert, des sweats, des abonnements à des plateformes de streaming, des voyages et d'autres cadeaux tendances.

Voici la liste de tous les cadeaux :

- Un cactus
- Un verre à bière
- Un calendrier personnalisé
- Un porte-clé personnalisé
- Un livre de cuisine pour étudiant
- Un livre de culture générale
- Des goodies Harry Potter
- Une visite de musée
- Deux places de cinéma
- Une carte cadeau de 20 euros
- Un cadre photo personnalisé
- Une boite de chocolats
- Un kit de coloriage
- Des jeux en bois
- Une peluche
- Une place de concert de rap
- Une place de concert de pop
- Une place de concert classique
- Un livre de mode
- Un sweat UTC
- Un sweat
- Une place d'exposition d'art

- Un jeu Pokémon
- Un abonnement Twitch
- Le jeu GTAVI
- Un jeu de société
- Une tenue de sport
- Une place pour un match de foot
- Une carte-cadeau de 50 euros
- Une écharpe
- Des playmobilis
- Des Legos
- Une peluche personnalisée
- Une place de festival
- Une place pour un défilé de mode
- Un dîner dans un restaurant chic
- Un vélo d'appartement
- Un abonnement BeinSport à l'année
- Une visite du musée Harry Potter
- Un abonnement Netflix à l'année
- Une carte-cadeau de 150 euros
- Un stage de cuisine
- Un voyage à deux
- Une montre de luxe
- Un week-end à deux
- Un kit complet de playmobilis
- Un kit complet de Legos
- Un kit d'éveil musical
- Des jeux en bois
- Un voyage en famille

Ensuite, nous avons déterminé un formalisme de représentation de ces données dans notre système expert. Pour cela, nous avons décidé de considérer un cadeau comme suit :

```
(conditions cadeau phrase_descriptive)
```

Ainsi, à chaque cadeau, nous sommes en capacité de fournir une phrase d'accroche, qui sera pertinente lors des affichages à l'utilisateur. Plutôt que d'obtenir une phrase du type "Le cadeau le plus adapté est un voyage", il aura le message personnalisé "Sa passion est visiter ! Voyager est la meilleure idée!".

De même, l'ajout de conditions au cadeau permettra de vérifier plus aisément lors du chainage arrière la convenance du cadeau.

Nous avons par exemple la représentation suivante de la place de festival dans notre base de connaissances :

```
(( ( budget gros ) ( relation ami ) ( age jeune ) ( musique t ) ( fetard t ) ) ' place_festival "Une place de festival !")
```

### 1.3 Bases de règles

A présent, nous savons définir de façon précise une personne et nous avons une liste de cadeau définie. Pour savoir quel cadeau est le plus adapté, il est nécessaire de concevoir une base de règles dont chacune est de la forme "Si ... alors ...". Pour concevoir cette base de règles, nous nous sommes fiés à l'expertise issue de la bibliographie. Celle-ci nous a permis de réaliser un moteur d'inférence d'une profondeur 3.

Dans notre formalisme, nous avons choisi de représenter une règle de la manière suivante :

```
(R1 ( condition1 condition2 ...) conclusion)
```

Nous avons opté pour cette représentation pour sa simplicité. En effet, ayant traité des cas similaires en travaux dirigés, nous aurons seulement besoin d'adapter les fonctions de service relatives aux règles.

Nous avons ainsi les règles suivantes liées au budget cadeau :

```
(R1 ((<= budget 30)) '(budget petit))
(R2 ((> budget 30) (<= budget 100)) '(budget moyen))
(R3 ((> budget 100) (<= budget 1500)) '(budget gros))
(R4 ((> budget 1500)) '(budget enorme))
```

### 1.4 Bases de questions

Afin d'en apprendre plus sur la personne, il est important de poser les bonnes questions. Chaque question fait référence à une caractéristique ou un goût de la personne. Par exemple, la question relative à l'âge est "Quelle âge a la personne ?".

Afin de manipuler les expressions les plus faciles possibles, nous avons choisi de représenter les questions de la manière suivante dans notre système expert :

```
(variable_impactee "Enonce de la question" conditions)
```

Ainsi, pour l'âge, nous retrouverons dans la base de questions l'information suivante :

```
(age "Quel age a la personne ?" NIL)
```

Les éventuelles conditions, dernier élément de la représentation, permettent de poser les questions uniquement dans le cas où certaines questions ont été posées préalablement. Par exemple, on veillera à ne pas demander quel est le type de jeux préféré de la personne si elle a indiqué auparavant ne pas aimer les jeux. On aura alors :

```
(type_jeux "La personne prefere t-elle les jeux videos ou les jeux de  
societe ? (video/societe)" (jeux t))
```

## 1.5 Moteur d'inférence

Le moteur d'inférence permet à notre système expert de conduire des raisonnements logiques et de dériver des conclusions à partir d'une base de faits et d'une base de connaissances. Ces dernières ayant été définies précédemment, il nous reste à déterminer quel type de raisonnement appliquer à notre système expert.

Le chainage avant et le chainage arrière semblent tous deux pertinents vis-à-vis de la problématique. En effet, dans un premier temps, on peut chercher un cadeau à offrir en fonction de sa personnalité (chainage avant). Néanmoins, il est aussi possible de vérifier si l'idée de cadeau à offrir semble pertinente ou non (chainage arrière). C'est pourquoi nous avons décidé de concevoir les deux types de raisonnement pour notre système expert.

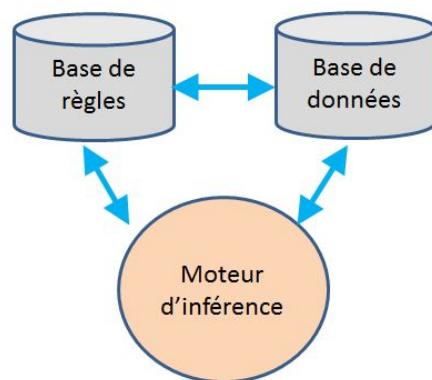


FIGURE 4 – Lien entre moteur d'inférence, base de règles et base de connaissances

Notre système expert est de profondeur 3. En effet, plusieurs exemples de questions et règles le démontrent. Ainsi, en fonction de l'âge, si la personne est jeune, on lui demandera si elle est étudiante, et si elle étudiante, on lui demandera si elle est à l'UTC. Nous avons appliqué le même raisonnement lié à la pratique de sport, de jeux et aux contenus télévisés regardés par la personne. Ceci peut être schématisé comme suit :

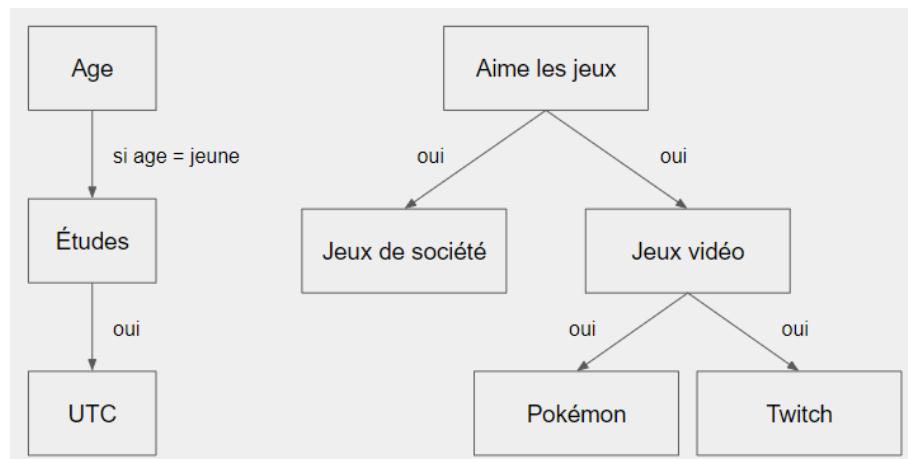


FIGURE 5 – Exemple de règles menant à une profondeur 3

## 2 Programmation du système expert

### 2.1 Chaînage avant

Nous avons dans un premier temps concentré nos efforts sur la réalisation d'un chainage avant. En effet, très généralement, nous ne savons pas quel cadeau offrir et nous aimions, à partir des caractéristiques de la personne, trouver le cadeau idéal.

Pour cela, on commence par poser 3 questions générales, permettant de connaître l'âge, la relation avec la personne ainsi que le budget maximal pour le cadeau. Ensuite, on posera judicieusement des questions jusqu'à ce que l'on trouve le cadeau idéal pour la personne. A chaque question posée, nous la retirons immédiatement de la base de questions afin de ne pas la poser une seconde fois.

Ainsi, nous obtenons le code suivant :

```
; ; Chainage avant : on regarde quel est le meilleur cadeau
(defun chainage-avant ()
    ; ; Variables utiles
    (let ((meilleure-question NIL) (cadeau NIL))
        ; ; Demande des questions basiques
        (ask-question 'age)
        (ask-question 'relation)
        (ask-question 'budget)
        ; ; Tant que l'on n'a pas trouve de cadeau et qu'il
        reste des questions
        (while (AND (eq cadeau NIL) (questions-with-priority))
            ; ; On pose la meilleure question
            (setq meilleure-question (best-question))
            (ask-question meilleure-question)
            ; ; On actualise le cadeau si un cadeau est
            possible
            (setq cadeau (cadeau-possible)))
        )
        ; ; On retourne le cadeau
        cadeau
    )
)
```

### 2.2 Chaînage arrière

Dans un second temps, nous avons réfléchi à la construction d'un moteur en chainage arrière. En effet, nous souhaiterions savoir si le cadeau que nous comptons acheter correspond à la personne. Pour cela, nous demandons à l'utilisateur quel cadeau il souhaite offrir, puis nous regardons s'il existe une règle candidate pour laquelle chacune des prémisses est vérifiée. Ainsi, nous obtenons le code suivant :

```
(defun chainage-arriere (cadeau)
```

```

(let ((ligne_cadeau NIL) (ok NIL) (adapte NIL))
  ;; Recherche de la ligne associee au cadeau
  (dolist (cad *LISTE_CADEAUX*)
    ;; Si c'est bon
    (if (eq (cadr cad) cadeau)
        (setq ligne_cadeau cad)
      )
    )
  ;; Pour chaque condition du cadeau
  (dolist (condition (car ligne_cadeau))
    ;; On pose la question associee
    (ask-question (car condition))
  )
  ;; Pur chaque cadeau de la liste
  (dolist (cad *LISTE_CADEAUX*)
    ;; Si c'est bon
    (if (eq (cadr cad) cadeau)
        (progn
          (setq ligne_cadeau cad)
          (setq ok t)
          ;; On verifie si toutes les
          conditions du cadeau sont verifiees
          (dolist (condition (car
            ligne_cadeau))
            (if (NOT(member
              condition *BASE_DE_FAITS* :test 'equal))
                (setq ok NIL)
              )
            )
          ;; Si toutes les conditions
          sont ok, on approuve le cadeau
          (if ok
              (setq adapte t)
            )
          )
        )
      )
    ;; On retourne si le cadeau est adapte ou NIL
    adapte
  )
)
)

```

## 2.3 Fonction main

La fonction main est le point d'entrée de notre programme d'un point de vue utilisateur. En effet, c'est cette fonction qui s'exécutera en premier pour l'utilisateur. Il nous faut alors lui expliquer brièvement le principe du programme, lui proposer un menu de choix et récupérer l'option qui le satisfait le mieux. Dans notre cas, on s'intéresse aux 2 cas possibles : trouver un cadeau et vérifier si un cadeau convient.

Dans le premier cas, on appelle la fonction chainage-avant qui renverra le cadeau idéal. Ainsi, nous pourrons alors afficher la phrase d'accroche liée au cadeau pour attirer son attention.

Dans le deuxième cas, on demandera d'abord à l'utilisateur quel cadeau souhaite-t-il offrir. Ensuite, nous appellerons la fonction de chainage arrière afin de vérifier si le cadeau convient. On pourra alors lui indiquer si le cadeau choisi est pertinent ou s'il vaut mieux changer d'idée.

Ainsi, on obtient le code de la fonction main suivant :

```
(defun main ()
  (let ((reponse NIL) (cadeau NIL) (convient NIL))
    ;; Affichage du menu
    (format t "~~ SYSTEME EXPERT : LE MEILLEUR CADEAU DE
NOEL ~~")
    (format t "~~~Que voulez-vous faire ?")
    (format t "~~~ 1) Trouver le cadeau ideal")
    (format t "~~~ 2) Verifier si le cadeau convient")
    ;; Demande du choix
    (format t "~~Votre choix : ")
    (setq reponse (read))
    (if (eq reponse 1)
        ;; Si trouver le cadeau ideal
        (progn
          (setq cadeau (chainage-avant))
          (format t "~~~s" (phrase-accroche
cadeau)))
        )
    ;; Si verifier si un cadeau convient
    (progn
      (format t "~~Quel cadeau souhaitez-vous
offrir ? ")
      (setq cadeau (read))
      (setq convient (chainage-arriere cadeau
)))
    (if convient
        ;; Si le cadeau convient
        (format t "~~Le cadeau ~s
convient !" cadeau)
        ;; Si le cadeau ne convient pas
        (format t "~~Il est encore
temps de changer d'idee. Le cadeau ~s ne semble pas adapte..." cadeau)
      )
    )
  )
)
```

## 2.4 Fonctions de service

Afin de coder le plus efficacement, nous avons conçu de nombreuses fonctions de service, nous permettant de gagner un temps considérable.

### 2.4.1 Fonctions relatives aux règles

Dans le souhait de mieux manipuler les règles, nous avons défini des fonctions permettant de récupérer le numéro de règle, les conditions et la conclusion de la règle. Egalement, grâce à add-regle(), nous pouvons ajouter une nouvelle règle à la base de règles.

Comme évoqué dans le TD sur les systèmes experts, nous avons conçu une fonction permettant de récupérer les règles candidates associées à un but précis.

### 2.4.2 Fonctions relatives à la base de faits

En ce qui concerne les faits, nous avons créé une fonction appartient(but) qui vérifié si un but est dans la base de faits. En effet, en passant à un système à l'ordre 0+, nous ne pouvons plus vérifier avec member la présence ou non d'un fait. Ainsi, notre fonction fera appel à funcall pour tester si le but est dans la base de faits ou non. Nous obtenons ainsi le code suivant :

```
; ; Fonction qui regarde si un but est dans la base de faits
(defun appartient (but)
  ;; Definition des variables
  (let ((operateur (car but)) (variable (cadr but)) (val2 (caddr but))
    ) val1)
    ;; On trouve val1 a partir de la base de faits
    (setq val1 (cadr (assoc variable *
      BASE_DE_FAITS*)))
    (if val1
      ;; On evalue si la condition entre les valeurs et l'
      operateur est correcte
      (funcall operateur val1 val2)
      ;; Sinon, on renvoie NIL
      NIL
    )
  )
)
```

La base de faits sera modifiée au fur et à mesure des questions posées. En effet, à chaque question posée, on ajoutera dans la base de faits une liste composée de la variable associée à la question et de la valeur donnée par l'utilisateur. Ces données nous permettront par la suite de tester si une règle est déclenchable ou non et de trouver le cadeau idéal.

### 2.4.3 Fonctions relatives aux cadeaux

Afin de gérer au mieux les nombreux cadeaux proposés par notre système expert, nous avons décidé de concevoir des fonctions pour manipuler les cadeaux. Ainsi, nous

avons défini des fonctions permettant d'ajouter ou de supprimer un cadeau de la liste des cadeaux. Ces dernières permettent une plus grande souplesse et une plus grande adaptabilité. En effet, si l'on souhaite rajouter un nouveau cadeau pour les prochaines fêtes de fin d'année, il est possible de faire appel à la fonction add-cadeau() qui ajoutera un cadeau dans la variable globale associée à la liste de cadeaux.

Nous avons mis en place des accesseurs permettant de retourner les conditions et les phrases d'accroche liées à un cadeau. Celles-ci nous permettent d'améliorer la lisibilité et la compréhension du code.

#### 2.4.4 Fonctions relatives aux questions

Nous avons d'abord conçu des fonctions pour ajouter et retirer des questions de la base de questions. Ces deux fonctions nous seront utiles respectivement à l'initialisation du programme et lorsque nous poserons des questions. En effet, cette technique de supprimer les questions nous permettra de ne pas poser deux fois une même question.

Toutefois, le sujet est plus subtil. En effet, il nous a amené à réfléchir sur la manière de poser des questions de manière intelligente. Au départ, nous posons les questions de manière aléatoire. Toutefois, cela ne nous semblait pas être la solution la plus optimale. C'est pourquoi nous avons décidé d'implémenter des fonctions de service permettant d'attribuer une priorité à chacune des questions. Si la variable impliquée dans une question est présente dans les conditions de 4 règles, alors la priorité de la question est 4. Ainsi, nous définissons la meilleure question comme la question ayant la plus grande priorité. Autrement dit, la meilleure question est celle qui a le plus de chance d'aboutir rapidement à une conclusion. Ainsi, nous avons l'algorithme suivant pour attribuer une priorité à chacune des questions :

```
Algorithme questions-with-priority () :
    ;; Pour chaque question
        ;; On suppose la question possible et on
        verifie si c'est le cas
            ;; Si la condition n'est pas verifiee
            dans la base de faits
                ;; Si toutes les conditions pour poser la
                question sont verifiees
                    ;; priorite = 0
                    ;; Pour chaque regle
                        ;; Si toutes les conditions sont verifiees ,
                        alors on incremente de 1 la priorite
                            ;; Ajout a la liste de priorites la
                            liste (question priorite)
                    ;; Retourne la liste
```

## 2.5 Exemples de fonctionnement du système expert

Ainsi, nous avons conçu un système expert à l'ordre 0+ fonctionnant aussi bien en chaînage avant qu'en chaînage arrière. En voici des exemples :

### 2.5.1 Chainage avant

Dans un premier temps, nous présenterons l'exemple de chainage avant. A partir de questions posées à l'utilisateur de manière interactive, le système expert est parvenu à proposer une solution adéquate. Ayant compris que le cadeau était destiné à un enfant créatif, le système expert en a déduit que le cadeau idéal serait des Playmobil. Ce cadeau permettra à l'enfant de s'amuser et d'exprimer sa créativité.

```
== SYSTEME EXPERT : LE MEILLEUR CADEAU DE NOEL ==
Que voulez-vous faire ?
 1) Trouver le cadeau idéal
 2) Vérifier si le cadeau convient
Votre choix : 1
"Quel âge a la personne ?"
>>> 8
"Quel est votre relation avec la personne ? (ami/famille/compagnon)"
>>> famille
"Quel est votre budget cadeau ?"
>>> 50
"La personne aime-t-elle les cadeaux humoristiques ?"
>>> non
"La personne aime-t-elle construire des choses ?"
>>> non
"La personne est-elle créative ?"
>>> oui
"Des playmobil !"
```

FIGURE 6 – Exemple de chaînage avant

Vous pourrez retrouver d'autres exemples en annexe de ce rapport.

### 2.5.2 Chainage arrière

Nous présentons ici un exemple de chainage arrière. A partir du cadeau entré par l'utilisateur, le système expert récupère les questions à poser (grâce aux conditions du cadeau dans la liste de cadeaux) et lui les pose. Toutes les réponses données correspondant exactement aux conditions, le système expert nous indique que le cadeau convient parfaitement !

```
==== SYSTEME EXPERT : LE MEILLEUR CADEAU DE NOEL ====
```

```
Que voulez-vous faire ?
```

- 1) Trouver le cadeau idéal
- 2) Vérifier si le cadeau convient

```
Votre choix : 2
```

```
Quel cadeau souhaitez-vous offrir ? jeux_de_societe
```

```
"Quel est votre budget cadeau ?"
```

```
>>> 35
```

```
"Quel est votre relation avec la personne ? (ami/famille/compagnon)"
```

```
>>> famille
```

```
"Quel âge a la personne ?"
```

```
>>> 14
```

```
"La personne aime-t-elle les moments en famille ?"
```

```
>>> oui
```

```
"La personne aime t'elle les jeux en général ?"
```

```
>>> oui
```

```
Le cadeau JEUX_DE_SOCIETE convient !
```

FIGURE 7 – Exemple de chaînage arrière

# Conclusion

## 2.6 Difficultés rencontrées et limites de notre projet

Lors de ce TP, nous avons identifié plusieurs difficultés. Dans un premier temps, il fallait s'assurer de faire des choix certes subjectifs mais qui restaient cohérents pour décrire une personne et établir les règles aboutissants à chaque cadeau tout en se basant sur notre expertise et nos recherches. Le nombre conséquent de règles était aussi une difficulté à gérer. De plus, il fallait veiller à ce que notre TP soit adapté au chainage avant et arrière.

Ensuite, pendant notre démonstration, notre chaînage avant a fonctionné mais même en posant plusieurs questions, il n'a pas trouvé de cadeaux adapté. Cette démonstration montre les limites de notre programme puisqu'il fonctionne pour certaines conditions bien précises mais pas pour tous les cas envisageables. Cela nous amène à imaginer des solutions telles que des algorithmes qui établiraient plus de règles pour un même cadeau ou même plus de cadeaux en apprenant à partir de données entrées par l'utilisateur, (par exemple en demandant à plusieurs personnes ce qu'elles achètent pour Noël et en déduisant de leurs réponses de nouvelles règles) ou qui proposerait un cadeau même si toutes les conditions ne sont pas remplies en proposant le cadeau le plus proche des données entrées par l'utilisateur ou encore en proposant de changer un paramètre comme le budget pour trouver un cadeau adapté à la personne.

On pourrait aussi émettre une critique sur la manière dont nos questions sont choisies. En effet, dans notre TP nous avons choisi de mettre une plus grande priorité sur les règles qui déclenchent le plus d'autres règles, hors ce n'est pas forcément cohérent avec les réponses données avant et le programme peut poser beaucoup de questions avant de trouver un cadeau. On pourrait imaginer travailler sur des questions propres à chaque catégorie de budget/age/relation par exemple, même si cela rajouteraient beaucoup de règles et rendrait le code plus lourd.

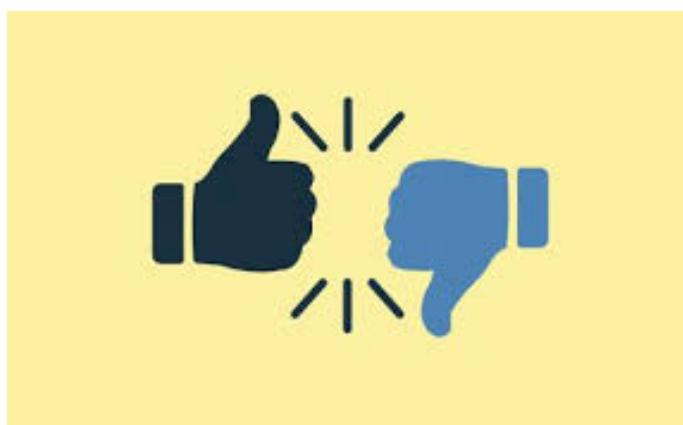


FIGURE 8 – Limites de notre projet

## 2.7 Bilan et conclusion

Ainsi, ce sujet nous a permis de mettre en pratique les connaissances théoriques apportées par le cours magistral et les travaux dirigés. Plus précisément, ce TP a été pour nous l'occasion de réfléchir à des problématiques de développement d'intelligence artificielle telle que la manière adéquate de poser des questions. Notre système expert à l'ordre 0+ nous a permis de mêler l'utile à l'agréable pendant la période des fêtes.

La réalisation de deux types de moteur d'inférence nous a permis de maîtriser les similarités et différences des deux modèles, tout en mettant en exergue des cas d'application.

Ce TP aura de multiples impacts dans notre future vie d'ingénieurs. En effet, il nous permettra de concevoir des systèmes experts performants, aussi bien en chaînage avant qu'en chaînage arrière. Nous serons en capacité de solliciter un expert, qui nous fera part de son expertise, et d'adapter ses connaissances dans notre système, pour le rendre opérationnel.

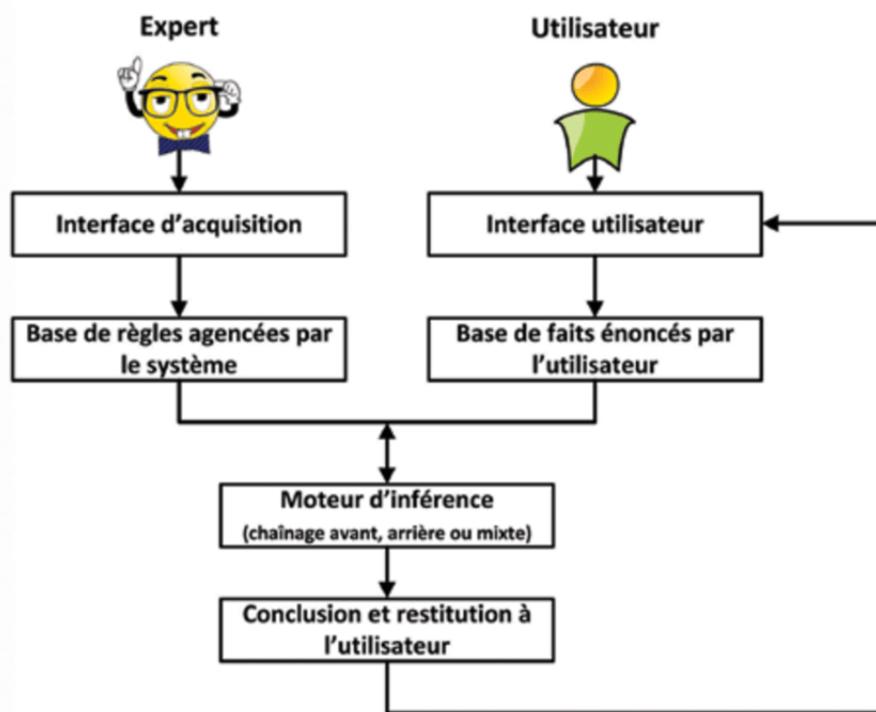


FIGURE 9 – Résumé du fonctionnement du système expert

### 3 Annexes

En annexes, nous avons décidé de mettre des exemples d'exécution de notre code pour montrer son fonctionnement. Pour lancer le programme, il faut l'exécuter (en sélectionnant tout le code en faisant **ctrl + E**) et appeler (**main**) dans l'invite de commande. A noter qu'il faut recompiler le code entièrement entre chaque appel de la fonction **main**.

```
CG-USER(307) : (main)
=====
SYSTEME EXPERT : LE MEILLEUR CADEAU DE NOEL =====

Que voulez-vous faire ?
 1) Trouver le cadeau ideal
 2) Verifier si le cadeau convient
Votre choix : 1
"Quel age a la personne ?"
>>> 18
"Quel est votre relation avec la personne ? (ami/famille/compagnon)"
>>> famille
"Quel est votre budget cadeau ?"
>>> 25
"La personne aime-t-elle les cadeaux humoristiques ?"
>>> oui
"La personne fait-elle des etudes ?"
>>> oui
"La personne est-elle a l'UTC ?"
>>> non
"La personne aime-t-elle la musique ?"
>>> non
"La personne aime t'elle les jeux en general ?"
>>> non
"La personne aime-t-elle le sport ?"
>>> non
"La personne aime-t-elle la mode ?"
>>> oui
"La personne est-elle casaniere ?"
>>> oui
"La personne aime-t-elle lire ?"
>>> oui
"La personne aime-t-elle apprendre de nouvelles choses ?"
>>> non
"La personne aime-t-elle regarder la television en general ?"
>>> non
"La personne est-elle tete en l'air ?"
>>> oui

"Un calendrier personnalisé pour qu'il ou elle n'oublie aucune date !"
```

```
CG-USER(614) : (main)
=====
SYSTEME EXPERT : LE MEILLEUR CADEAU DE NOEL =====

Que voulez-vous faire ?
 1) Trouver le cadeau ideal
```

```

2) Verifier si le cadeau convient
Votre choix : 1
"Quel age a la personne ?"
>>> 18
"Quel est votre relation avec la personne ? (ami/famille/compagnon)"
>>> ami
"Quel est votre budget cadeau ?"
>>> 40
"La personne aime-t-elle les cadeaux humoristiques ?"
>>> oui
"La personne fait-elle des etudes ?"
>>> non
"La personne aime-t-elle la musique ?"
>>> oui
"Quel type de musique la personne prefere t-elle ? (classique/rap/pop)"
>>> rap
"La personne aime t'elle les jeux en general ?"
>>> non
"La personne aime-t-elle le sport ?"
>>> oui
"La personne pratique-t-elle du sport ou regarde-t-elle uniquement ? (pratiquer/regarder)"
>>> pratiquer

"Une tenue de sport pour continuer le sport en 2023 !"

```

```

CG-USER(921): (main)

===== SYSTEME EXPERT : LE MEILLEUR CADEAU DE NOEL =====

Que voulez-vous faire ?
1) Trouver le cadeau id al
2) V rifier si le cadeau convient
Votre choix : 2

Quel cadeau souhaitez-vous offrir ? sweat
"Quel est votre budget cadeau ?"
>>> 35
"Quel est votre relation avec la personne ? (ami/famille/compagnon)"
>>> ami
"Quel age a la personne ?"
>>> 18
"La personne aime-t-elle la mode ?"
>>> oui

Le cadeau SWEAT convient !

```

```

CG-USER(307): (main)

===== SYSTEME EXPERT : LE MEILLEUR CADEAU DE NOEL =====

Que voulez-vous faire ?
1) Trouver le cadeau ideal
2) Verifier si le cadeau convient
Votre choix : 1
"Quel age a la personne ?"

```

>>> 45  
"Quel est votre relation avec la personne ? (ami/famille/compagnon)"  
>>> ami  
"Quel est votre budget cadeau ?"  
>>> 170  
"La personne aime-t-elle les cadeaux humoristiques ?"  
>>> non  
"La personne aime-t-elle la musique ?"  
>>> oui  
"Quel type de musique la personne prefere t-elle ? (classique/rap/pop)"  
>>> pop  
"La personne aime t'elle les jeux en general ?"  
>>> non  
"La personne aime-t-elle le sport ?"  
>>> oui  
"La personne pratique-t-elle du sport ou regarde-t-elle uniquement ? (pratiquer/regarder)"  
>>> regarder  
"La personne aime-t-elle la mode ?"  
>>> oui  
"La personne est-elle casaniere ?"  
>>> non  
"La personne aime-t-elle lire ?"  
>>> oui  
"La personne aime-t-elle apprendre de nouvelles choses ?"  
>>> oui  
"La personne aime-t-elle regarder la television en general ?"  
>>> oui  
"La personne aime-t-elle regarder des series ?"  
>>> oui  
"La personne est-elle tete en l'air ?"  
>>> non  
"La personne est-elle sensible a l'art ?"  
>>> oui  
"La personne est-elle fetarde ?"  
>>> oui  
"La personne aime-t-elle la decoration ?"  
>>> non  
"La personne aime-t-elle les moments en famille ?"  
>>> non  
"La personne aime-t-elle les choses fantastiques (ex : Harry Potter) ?"  
>>> non  
"La personne aime-t-elle cuisiner ?"  
>>> oui  
  
"Un stage de cuisine !"

## Références

- [1] [Sondage Ipsos Décembre 2021](#)
- [2] [Classement des cadeaux vendus](#)
- [3] [Quel est le cadeau le plus offert à Noël](#)
- [4] [Les 55 cadeaux de Noël préférés des Français](#)
- [5] [Cadeaux les plus offerts sur Amazon](#)
- [6] [Offrir un livre à Noël](#)