Projet TP3

Expertise dans la création de recette de cocktail alcoolisées

Réalisé dans le cadre de l'UV IA01 intelligence artificielle au sein de l'UTC.

Réalisé par Thomas Habert et Cléa Bordeau

Introduction / Concept :

L'objectif de l'algorithme est de proposer à l'utilisateur des recettes de cocktails alcoolisé en fonction des ingrédients disponibles dans son placard en entrée de système et de certains critères choisis par l'utilisateur comme par exemple le taux d'alcool ou la difficulté de réalisation du cocktails.

Dans ce système expert nous utiliserons, une base de fait, qui contiendra les informations entrées par l'utilisateur, une base de règle pour les recettes et une base de règle pour les ingrédients similaires.

I - Base de Fait et Base de règle

Données de l'algorithme : BASE DE FAIT

Dans la base de fait, l'utilisateur pourra rentrer un certain nombre d'éléments, qui pourront être :

- Données d'entrées : Un ou plusieurs ingrédients

Chaque ingrédient sera représenté par une liste contenant les variables suivantes :

- Son **nom** : celui-ci devra être unique *[Ordre : 0+, les noms sont des strings d'un ensemble fini]*
- Un **nombre entier positif** représentant le nombre total de portion que l'utilisateur possède pour cet ingrédient. *[Ordre : 0+ également]*

- Données d'entrées : Une difficulté

La caractéristique difficulté est représentée par une liste avec les variables suivantes :

- Le mot **difficulte**
- Un **entier** qui représente un niveau de difficulte max que l'utilisateur souhaite réaliser (max 5)

- Données d'entrées : Un préférence de pétillance

La caractéristique petillance est représentée par une liste avec les variables suivantes :

- Le mot **petillant**
- Un attribut qui peut prendre 3 valeurs **(1 ou 0 ou -1)**: 1 l'utilisateur souhaite un cocktail pétillant, 0 il ne veut pas de pétillance, -1 il n'a pas de préférence..

- Données d'entrées : Une préférence de saveure fruitée

La caractéristique fruité est représentée par une liste avec les variables suivantes :

- Le mot **fruite**
- Et un attribut qui peut prendre 3 valeurs**(1 ou 0 ou -1)**: 1 l'utilisateur souhaite un cocktail fruité, 0 non fruité, -1 il n'a pas de préférence.

- Données d'entrées : Un NIVEAU d'alcoolémie

La caractéristique alcoolémie est représentée par une liste avec les variables suivantes :

- Le mot **niveau_alcoolemie**
- Et un **entier entre 0**(non alcoolisé) **et 3** : cela représente la limite d'alcool à ne pas dépasser demandée par l'utilisateur.

Exemple :

Données de l'algorithme : BASES DE RÈGLES

Base de règle des recettes

Chaque règle portera le nom d'une recette *(unique, sert d'identifiant)* et sera définit par une sous liste contenant tout un ensemble de données définies sous forme de liste de 2 arguments qui pourront être :

- Un nom d'ingrédient et sa quantité
- Le mot difficulte et un entier compris entre 1 et 5
- Le mot niveau_alcoolemie et un entier compris entre 0 et 3
- Le mot petillant et un booléen
- Le mot fruite et un booléen

Format d'une règle de recette : *Une règle se présente ainsi sous le format suivant :*

(nomRecette ((ingredient1 dose1) … (ingredientN doseN) (difficulte entier) (petillant booléen) (fruite booléen) (niveau_alcoolemei entier))

Remarque : Il peut y avoir un nombre illimité de listes avec un nom d'ingrédient et sa quantité, en revanche tout autre type de donnée énumérée ci-dessus sera présente strictement une fois dans la règle.

Exemple :

(glucose ((sirop_sucre 2) (difficulte 2) (petillant 0) (fruite 1) (niveau_alcoolemie 2)))

Représente le fait que : si on a au moins 2 doses de sirop_sucre, qu'on souhaite une difficulté maximal d'au moins 2, qu'on souhaite un niveau d'alcoolémie d'au moins 2, qu'on a une préférence pour fruité ou une indifférence, et qu'on a soit une préférence pour non pétillant ou une indifférence, alors on a le cocktail "glucose" de réalisable.

Base de règle des ingrédients remplaçants

Chaque règle est constitué de la manière suivante :

- Le nom de l'ingrédient à remplacer en premier
- Une sous-liste contenant des potentiels remplaçant par ordre de pertinence ainsi que les quantités nécessaires pour remplacer une portion de l'ingrédient à remplacer.

Format d'une règle pour un ingrédients à remplacer : On peut représenter les différentes règles de la manière suivante :

(Ingrédient_a_remplacer ((Ingrédient_remplaçant1 Quantite1) … (Ingrédients_remplaçantN QuantiteN)))

Cela représente le fait que: Si on a au moins une fois l'ingrédient d'une des différentes listes en quantité suffisante alors on peut remplacer l'ingrédient.

II - Fonctionnement :

Les fonctions principales :

- **Main** : menu principal qui quand il s'ouvre pose les questions, et appelle les fonctions nécessaires.
- **Base de fait, base de règle**
- **IsKnown:** test si un ingrédient donné est présent dans la base de règles.
- **AskBF :** demande tous les ingrédients et leur quantité, (STOP pour arrêter les ingrédients) puis les caractéristiques (si 0 est entrée == l'utilisateur ne veut pas de condition particulière)
- **TestValidity :** qui pour une recette en entrée retourne la recette si la recette correspond aux critères et NIL sinon
- **RecettesValides :** Parcourt BR et pour chaque règle test sa validité avec **TestValidity** et ajoute la recette à une liste RecettesValides si elle est valide.
- **UpdateBF :** Prend en argument une recette sélectionnée par l'utilisateur et supprime de la base de faits les ingrédients utilisés.
- **SearchReplacement :** Prend en argument un ingrédient à remplacer et la recette avec cet ingrédient, cherche un remplaçant de l'ingrédient dans la base de remplaçants et renvoi la nouvelle recette possible par l'utilisateur ou nil.

Conclusion

En définitive, ce projet a été l'occasion pour nous d'appliquer les connaissances accumulées en TD au cours du semestre ainsi que d'approfondir certaines notions en effectuant nos propres recherches.

Pour réaliser ce problème nous avons suivi les étapes suivantes :

1- Etablir deux bases de règles : une pour les recettes et une seconde pour les remplacements

2- Créer différentes fonctions constituant le moteur d'inférences (Recettes valides, Search Replacement,...)

3- Créer une interface pour que l'utilisateur puisse interagir avec les systèmes.

Il était intéressant d'arriver à la réalisation d'un projet imaginé par nos soins et d'y ajouter différentes particularités liées à notre sujet. Nous avons dû envisager plusieurs possibilités et parfois revenir en arrière pour finalement arriver à un résultat convaincant.

A noter que notre système a été conçu de manière à être capable de recevoir une infinité de règles supplémentaires. Il également perméable à d'autres utilisations telles qu'un chaînage arrière permettant de donner à un utilisateur entrant un nom de recette, la liste des ingrédients de celle-ci. Celà présente des pistes d'amélioration que nous n'avons pas ajouté dans le cadre du TP3 de IA01.