Mini-projet d'« Intelligence artificielle »

José Martinez

2018-2019

A ▶ Apprentissage de concepts structurés ●●●●○

Certains tests de Q.I. (quotient intellectuel) proposent de trouver un intrus dans un ensemble d'objets. Ces objets peuvent être numériques, textuels, graphiques (cf. figures 1 et 2).

Intéressons-nous à des graphiques mettant en œuvre formes, couleurs, tailles et alignements. Les formes seront « cercle », « carré », « triangle », etc. Les couleurs seront « vert »,

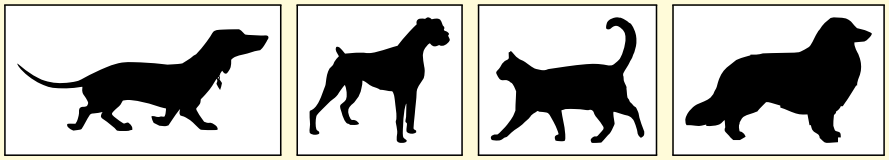

Figure 1 – Un test de Q.I. « Trouvez l'intrus » pour s'échauffer... pour un humain ! [Images issues de svgsilh.com]

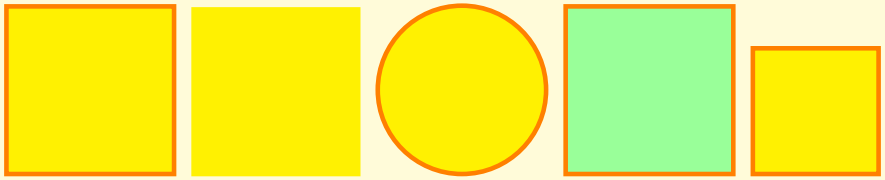

FIGURE 2 – Un test de Q.I. « Trouvez l'intrus » (méta) difficile...

« bleu », « rouge », « jaune », etc. Les tailles seront « petit », « moyen », « grand », etc. Les alignements seront « vertical », « horizontal », etc.

La description d'un tel graphique est récursive. La grammaire correspondante est :

$$
\begin{aligned}
G &\to \text{feuille}(F, C, T) \mid \text{nœud}(A, G, G) \\
F &\to \text{cercle} \mid \text{carré} \mid \text{triangle} \\
C &\to \text{vert} \mid \text{bleu} \mid \text{rouge} \mid \text{jaune} \\
T &\to \text{petit} \mid \text{moyen} \mid \text{grand} \\
A &\to \text{vertical} \mid \text{horizontal}
\end{aligned}
$$

Un graphique élémentaire, ou feuille, se limite donc à un forme colorée d'une certaine taille. Un graphique structuré, ou nœud interne, permet de placer deux sous-graphiques à une certaine position l'un par rapport à l'autre.

La figure 3 illustre le propos par quelques graphiques aléatoires de tailles croissantes. Les descriptions structurées des premiers sont :

1. leaf(square, red, medium) ;
2. node(vertical, leaf(square, green, medium), leaf(circle, red, medium)) ;
3. node(vertical, leaf(square, blue, large), node(vertical, leaf(square, green, medium), leaf(circle, green, small))) ;

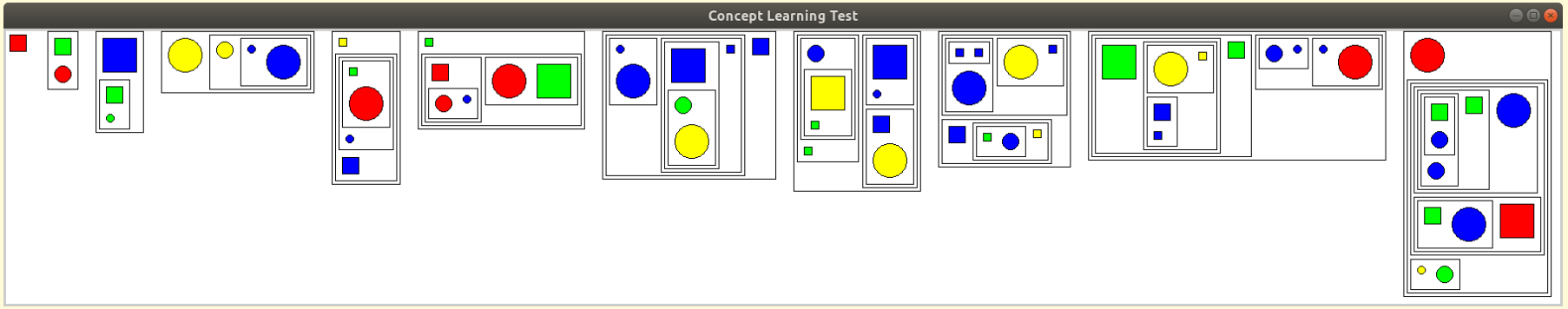

FIGURE 3 – Onze graphiques aléatoires de tailles croissantes

4. node(horizontal, leaf(circle, yellow, large), node(horizontal, leaf(circle, yellow, medium), node(horizontal, leaf(circle, blue, small), leaf(circle, blue, large)))) ;
5. etc.

Le code fourni reprend la grammaire de l'énoncé pour définir des graphiques structurés en Prolog. Il permet alors de générer (i) tous les graphiques d'une taille donnée ou (ii) un graphique aléatoire de taille donnée ou aléatoire. Accessoirement, un ou plusieurs graphiques peuvent être affichés à l'écran comme illustré par les figures.

1 ▶ La taille n d'un graphique est le nombre d'éléments graphiques de base, ou feuilles, dont il est constitué.

Déterminez le nombre d'objets différents de taille n que l'on peut construire avec f formes, c couleurs, t tailles et a alignements :

(i) dans le cas où la structure de l'arbre binaire est fixée ;

(ii) dans le cas général.

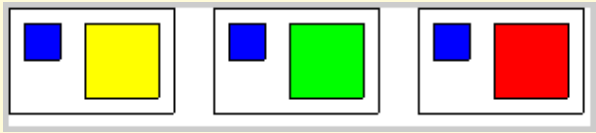

FIGURE 4 – Un ensemble de graphiques similaires

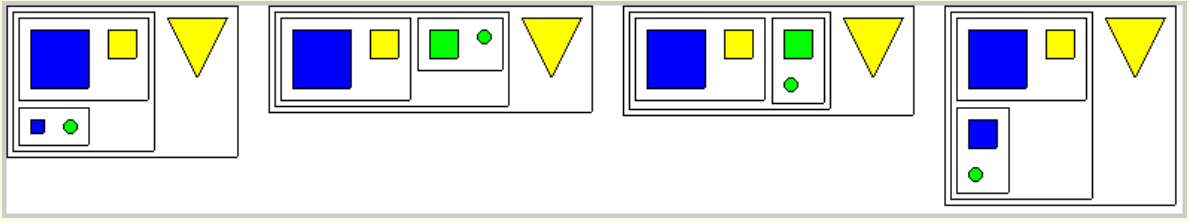

FIGURE 5 – Un autre ensemble de graphiques similaires

Indication. Le décompte d'arbres binaires complets est un problème bien connu.

2 ▶ Comme le démontre la figure 2, il est difficile d'envisager et même d'imaginer toutes les formes de similitudes entre des objets...

Proposez :

(i) des exemples de similitudes entre les figures graphiques proposées, sans vous limiter *a priori* ;

(ii) des exemples de similitudes entre sous-ensembles d'objets ;

(iii) le type de similitude relevant des exemples des figures 4, 5 et 6.

3 ▶ On se limitera aux similitudes correspondants aux figures 4, 5 et 6.

Exemple. Le concept associé à la figure 4 pourrait se traduire en français par :

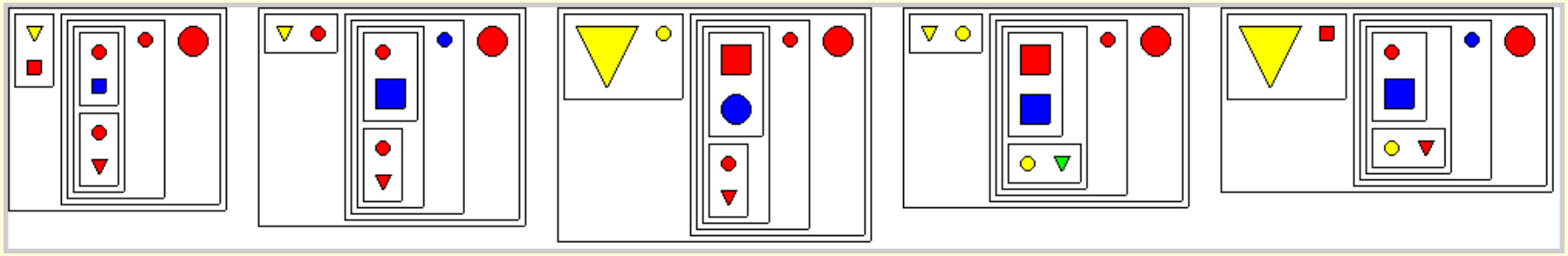

FIGURE 6 – Un dernier ensemble de graphiques similaires

« Un carré moyen bleu à gauche d'un grand carré d'une autre couleur. »

Celui de la figure 6 semble inhumain. Il relève donc bien de l'Intelligence artificielle !

Formalisez le problème de la reconnaissance du *concept* caché :

(i) sans la présence d'un intrus (comme c'est le cas sur les figures mentionnées) ;

> **Indication.** Placez-vous au niveau très général des *concepts* abstraits. Ce n'est qu'ensuite que vous décrirez plus précisément quels concepts concrets vous manipulerez.

(ii) avec la présence d'un unique intrus ;

(iii) avec la présence de plusieurs intrus, voire d'une majorité !

4 ▶ Dans le cas de la recherche du concept minimal avec exactement un intrus, proposez une modélisation de l'exploration du graphe de recherche selon l'approche préconisée (E, i, F, O, P, C).

5 ▶ Implémentez vos solutions en Prolog.

Mettez également en place les tests qui permettent de vérifier automatiquement que l'algorithme réalise bien la tâche d'apprentissage d'un concept à partir d'exemples et de contre-exemple(s).

Évaluation

Comme tout problème relevant de l'Intelligence artificielle, ce mini-projet est complexe. Il peut donner lieu à des développements extrêmement poussés, y compris jusqu'au niveau d'un problème de recherche. Il est donc nécessaire de procéder par étapes en évaluant les niveaux de complexité de solutions de plus en plus générales, en sélectionnant le niveau adéquat en fonction de vos compétences et du temps alloué, et en répartissant de manière adéquate et équilibrée la charge de travail entre les deux membres d'un binôme. Il est également conseillé d'anticiper les modifications ultérieures afin de minimiser les changements dans le code de l'étape précédente – et donc les erreurs.

L'ordre correspond à un développement raisonné et progressif depuis le problème jusqu'à différentes solutions de complexités croissantes.

Éléments d'appréciation

Les éléments d'appréciation sont fonction de chaque partie :

1. La définition formalisée du problème (questions 1 à 4) est évaluée sur :
   (a) les propositions formelles et leurs justifications ;
   (b) la « simplicité » des algorithmes (non déterministes).

2. La résolution du problème est évaluée sur :
   (a) la clarté de la traduction des algorithmes en Prolog ;
   (b) la prise en compte d'heuristiques et leur argumentation ;
   (c) les performances observées.

Compte-rendus

Les codes sources ne suffisent pas. Le compte-rendu comportera trois parties :

1. Une première partie fournira la résolution générale du problème (questions 1 à 4). Cette partie est à rendre après la deuxième séance de TP. *Le dépôt sur Madoc devra avoir été effectué au plus tard le 16 décembre 2018 à 23 h 55.*
2. Le rapport final y ajoutera les détails de réalisation en décrivant les difficultés rencontrées, les solutions proposées pour chaque partie et en illustrant le fonctionnement du système.
3. Enfin, les codes sources et des exemples d'utilisation seront aussi déposés sur Madoc. *Le seul dialecte de Prolog autorisé est SWI-Prolog.*

   Chaque partie du code source devra mentionner l'historique des modifications depuis la création en précisant *le nom de l'auteur, la date et la nature de la modification.*

   Chaque prédicat devra contenir la description explicite et commentée de *sa précondition et de sa post-condition*, et de préférence son évaluation dans le code lui-même.

   Chaque prédicat devra être accompagné de ses *tests unitaires*, le cas échéant d'intégration. Enfin, l'ensemble du programme devra être accompagné au moins d'*une instance réaliste* à faire résoudre par le programme.