**Rapport du TP2 : Knights and Knaves**

**Introduction**

Ce rapport présente l'implémentation et l'analyse du problème "Knights and Knaves", inspiré des énigmes logiques de Raymond Smullyan. L'objectif est de déterminer la nature de chaque personnage (chevalier ou filou) en utilisant la logique propositionnelle et l'algorithme de vérification de modèle.

**Contexte du Problème**

Dans ces énigmes, chaque personnage est soit :

* **Un chevalier** : dit toujours la vérité.
* **Un filou** : ment toujours.

Nous devons traduire ces règles en expressions logiques et utiliser une IA pour déduire la solution.

**Implémentation**

Le code suivant définit les symboles logiques pour représenter les personnages et leurs déclarations :

AKnight = Symbol("A is a Knight")

AKnave = Symbol("A is a Knave")

BKnight = Symbol("B is a Knight")

BKnave = Symbol("B is a Knave")

CKnight = Symbol("C is a Knight")

CKnave = Symbol("C is a Knave")

Chaque personnage est soit un chevalier, soit un filou, mais pas les deux à la fois :

Or(AKnight, AKnave), Not(And(AKnight, AKnave))

**Puzzle 0**

A dit : "Je suis à la fois un chevalier et un filou."

knowledge0 = And(

Or(AKnight, AKnave),

Not(And(AKnight, AKnave)),

Implication(AKnight, And(AKnight, AKnave)),

Implication(AKnave, Not(And(AKnight, AKnave)))

)

**Résultat obtenu :**

* A est un **filou**.

**Puzzle 1**

A dit : "Nous sommes tous les deux des filous."

knowledge1 = And(

Or(AKnight, AKnave),

Or(BKnight, BKnave),

Not(And(AKnight, AKnave)),

Not(And(BKnight, BKnave)),

Implication(AKnight, And(AKnave, BKnave)),

Implication(AKnave, Not(And(AKnave, BKnave)))

)

**Résultat obtenu :**

* A est un **filou**.
* B est un **chevalier**.

**Puzzle 2**

A dit : "Nous sommes du même type."  
B dit : "Nous sommes de types différents."

knowledge2 = And(

Or(AKnight, AKnave),

Or(BKnight, BKnave),

Not(And(AKnight, AKnave)),

Not(And(BKnight, BKnave)),

Implication(AKnight, Biconditional(AKnight, BKnight)),

Implication(AKnave, Not(Biconditional(AKnight, BKnight))),

Implication(BKnight, Biconditional(AKnight, BKnave)),

Implication(BKnave, Not(Biconditional(AKnight, BKnave)))

)

**Résultat obtenu :**

* A est un **filou**.
* B est un **chevalier**.

**Puzzle 3**

A dit soit "Je suis un chevalier." soit "Je suis un filou." B dit : "A a dit 'Je suis un filou'." B dit ensuite : "C est un filou." C dit : "A est un chevalier."

knowledge3 = And(

Or(AKnight, AKnave),

Or(BKnight, BKnave),

Or(CKnight, CKnave),

Not(And(AKnight, AKnave)),

Not(And(BKnight, BKnave)),

Not(And(CKnight, CKnave)),

Implication(AKnight, Or(AKnight, AKnave)),

Implication(AKnave, Not(Or(AKnight, AKnave))),

Implication(BKnight, Biconditional(AKnave, True)),

Implication(BKnave, Biconditional(AKnave, False)),

Implication(BKnight, CKnave),

Implication(BKnave, Not(CKnave)),

Implication(CKnight, AKnight),

Implication(CKnave, Not(AKnight))

)

**Résultat obtenu :**

* A est un **chevalier**.
* B est un **filou**.
* C est un **chevalier**.

**Conclusion**

Nous avons utilisé la logique propositionnelle pour modéliser et résoudre des énigmes logiques en déterminant si chaque personnage est un chevalier ou un filou. L'utilisation de la vérification de modèle a permis d'obtenir automatiquement les résultats souhaités.