UB Social

CS50AI with Python

3.Knights

Mateus Schwede

Problemática

- Escreva programa para resolver quebra-cabeças lógicos (puzzles);
- Em 1978, lógico Raymond Smullyan publicou "Qual o nome deste livro?", livro de quebra-cabeças lógicos. Entre os quebra-cabeças do livro, havia classe de quebra-cabeças que Smullyan chamou de quebra-cabeças "Knights and Knaves" (cavaleiros e valetes/patifes);
- No quebra-cabeça Knights e Knaves, cada personagem é knight ou knave. Knight sempre dirá verdade: se knight afirma frase, então frase é verdadeira. Por outro lado, knave sempre mentirá: se knave afirma frase, então frase é falsa;
- Objetivo do quebra-cabeça é, dado conjunto de frases ditas por cada personagens, determinar, para cada, se o mesmo é knight ou knave;
- Exemplo, considere puzzle simples com apenas 1 personagem chamado A. A diz "sou knight e knave";
- Logicamente, pode-se raciocinar que se A fosse knight, então frase é verdadeira. Mas sabe-se que frase não pode ser verdadeira, porque A não pode ser knight e knave - sabe-se que cada personagem é knight ou knave, não ambos. Então, conclui-se que A é knave;

Problemática

- Objetivo neste problema é determinar como representar puzzles usando lógica proposicional, de modo que IA execute algoritmo de verificação de modelo (model-checking) possa resolver esses puzzles automaticamente.

Instruções

- Baixe código de https://cdn.cs50.net/ai/2023/x/projects/1/knights.zip e descompacte-o.

Funcionamento

- Arquivo logic.py define classes para diferentes tipos de conectivos lógicos. Tais classes podem ser compostas umas dentro das outras, então expressão como And(Not(A), Or(B, C)) representa sentença lógica afirmando que símbolo A não é verdadeiro, e que B ou C são verdadeiros (onde "OR" aqui se refere a inclusivo, não exclusivo, or);
- logic.py contém função model_check, que pega base de conhecimento e consulta. Base de conhecimento é única sentença lógica: se várias sentenças lógicas forem conhecidas, elas podem ser unidas em expressão And. model_check considera recursivamente todos modelos possíveis e retorna True se base de conhecimento implica consulta, e retorna False caso contrário;
- Arquivo puzzle.py define 6 símbolos proposicionais. AKnight, por exemplo, representa frase que "A é knight", enquanto AKnave representa frase que "A é knave". O mesmo para B e C;
- Tem-se 4 bases de conhecimento diferentes e vazias, knowledge0, knowledge1, knowledge2 e knowledge3, que conterão conhecimento necessário para deduzir soluções para próximos Puzzles 0, 1, 2 e 3, respectivamente;
- Função main de puzzle.py percorre todos puzzles e usa model-checking para calcular, conforme conhecimento daquele puzzle, se cada personagem é knight ou knave, imprimindo quaisquer conclusões que algoritmo de verificação de modelo seja capaz de tirar.

Especificações

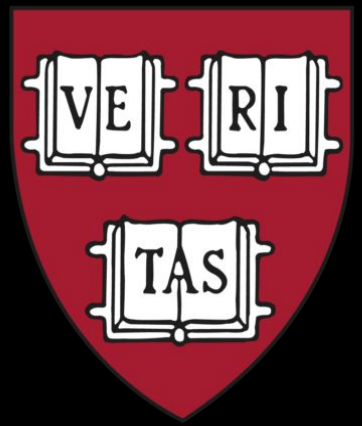
- Adicione conhecimento/conteúdo para bases de conhecimento knowledge0, knowledge1, knowledge2, e knowledge3 para resolver os seguintes puzzles;
- Puzzle 0 é quebra-cabeça do Background, contendo único personagem, A;
    - A diz "sou knight e knave".
- Puzzle 1 tem 2 personagens, A e B;
    - A diz "Nós 2 somos knaves";
    - B não diz nada.
- Puzzle 2 tem 2 personagens, A e B;
    - A diz "Somos da mesma espécie";
    - B diz "Somos de espécies diferentes".

Especificações

- Puzzle 3 tem 3 personagens, A, B e C;
  - A diz "Eu sou knight" ou "Eu sou knave", mas você não sabe qual;
  - B diz "A disse 'Eu sou knave'";
  - B então diz "C é knave";
  - C diz "A é knight".
- Nos puzzles, cada personagem é knight ou knave. Cada frase dita por knight é verdadeira, cada frase dita por knave é falsa;
- Após base de conhecimento para problema, pode-se executar puzzle.py para ver solução do puzzle.

Submissão

- Visual Studio Code online: https://cs50.dev
- Testar precisão da lógica do algoritmo: check50 ai50/projects/2024/x/knights
- Testar estilização do código: style50 puzzle.py
- Para submissão:
  - Em https://submit.cs50.io/invites/d03c31aef1984c29b5e7b268c3a87b7b, entre com GitHub e autorize CS50;
  - Instale pacote Git, Python 3 (e pip), instalando pacotes: pip3 install style50 check50 submit50
  - Submeta o projeto: submit50 ai50/projects/2024/x/knights
- Verificar avaliação: https://cs50.me/cs50ai.



UB Social

Mateus Schwede
HBS ID 202400167108 - DCE ID @00963203