Projeto Inteligência Computacional

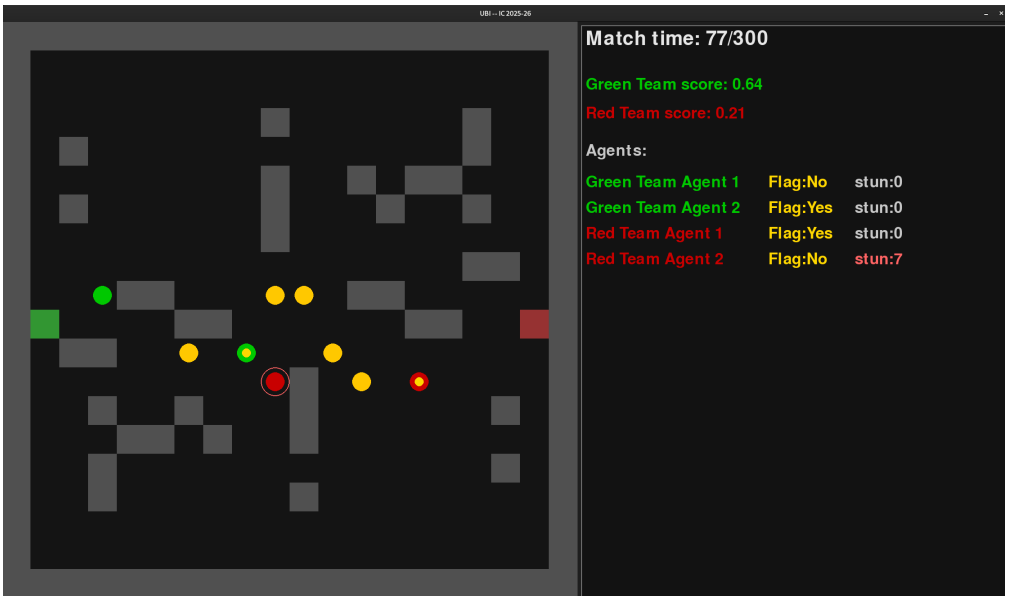
2025-26

1 Introdução

O objetivo do projeto é explorar técnicas de IC para criar uma política para uma equipa constituída por dois agentes conseguir ganhar um jogo de captura-a-bandeira.

Cada um dos elementos do grupo vai desenvolver uma das abordagens para um dos agentes.

2 Descrição do jogo



O jogo decorre num tabuleiro com duas equipas a competir ao mesmo tempo. Cada equipa tem dois jogadores. Cada equipa tem uma cor diferente e uma base.

O objetivo é encontrar no tabuleiro as bandeiras (marcas amarelas) e retorná-las à base. Cada bandeira trazida para a base vale 3 pontos. Cada jogador só pode transportar uma bandeira de cada vez. Após a entrega de uma bandeira na base, ela volta a aparecer no campo. Quando um jogador transporta uma bandeira, o centro do seu marcador fica amarelo. Enquanto transporta a bandeira vai ganhando alguns (poucos) pontos.

Os jogadores podem atorduar os adversários se estiverem no quadrado ao lado. Isto faz com que o adversário fique parado durante 10 instantes. O jogo tem a duração de 300 instantes (cerca de 40 segundos). Quando um agente fica atordoado, aparece um círculo à sua volta. Se estiver a transportar uma bandeira, ela volta à sua posição inicial.

Diferentes valores indicam os vários elementos do jogo: célula vazia: 0, parede: 50, bandeira: 100, agente da mesma equipa: 150, agente adversário: 200, base: 250. Estes valores aparecem na matriz de observação que cada agente recebe. Esta matriz é quadrada e tem lado igual a 2*VISION+1. Isto significa que cada agente não vê todo o tabuleiro mas apenas esta região em torno da sua posição (o problema é parcialmente observável). A constante VISION pode ser ajustada para se fazerem testes. Tipicamente terá um valor de 3.

2.1 Ações

Cada agente pode fazer uma de seis ações:

- 0: nada acontece, o agente permanece no mesmo local;
- 1: sobe
- 2: desce
- 3: esquerda
- 4: direita
- 5: atordoar o adversário: ganha 0.2 pontos e o adversário perde 0.1. Se o adversário estiver a transportar uma bandeira, ela volta à sua posição inicial.
- 6: largar a bandeira. Se for na base, ganha 3 pontos. Fora da base, a bandeira volta à posição inicial.

2.2 Ficheiro agenteX.py

Cada elemento do grupo tem que desenvolver o código de um agente. Cada agente tem o seu código num ficheiro individual: `agente1.py` e `agente2.py`. Isto permite que o desenvolvimento seja feito em paralelo e um estudante não precise de esperar pelo outro. Podem testar carregando o mesmo ficheiro para os dois agentes, se um dos elementos ainda não tem código feito. Ou então usar o código da equipa adversária que é muito simplesmente uma ação aleatória (ver função).

Dentro deste ficheiro existe pelo menos a função `politica(obs, agentID)` que recebe uma observação na variável `obs` e devolve a ação que o agente quer fazer.

3 Inteligência Computacional

Como é que entra aqui a IC? Vamos explorar as técnicas de IC para criar a política que indica ao agente o que deve fazer. Ficam aqui algumas **sugestões**:

- uma rede neuronal (RN) pode aprender uma política fazendo um mapeamento do espaço das observações para probabilidade de executar ações.
- A computação evolucionária (CE) pode desenvolver políticas representando-as como conjuntos de parâmetros (por exemplo, pesos de uma rede neuronal ou de um sistema baseado em regras) e utilizando algoritmos genéticos (AG) ou estratégias de evolucionárias (EE) para as otimizar. A política pode ser uma estrutura fixa (por exemplo, uma pequena rede neuronal ou árvore de decisão) cujos parâmetros são desenvolvidos.
- IE: A inteligência de enxame (IE) cria políticas modelando os agentes como partículas que otimizam a sua estratégia de seleção de ações num ambiente partilhado. A OEP pode otimizar os parâmetros de uma política, enquanto a otimização de colónias de formigas (OCF) pode construir um modelo probabilístico de seleção de ações com base em pesos semelhantes a feromonas.
- SD: Os sistemas difusos (SD) podem criar políticas baseadas em regras que lidem com a incerteza nas observações (por exemplo, visibilidade parcial, posições inimigas imprecisas). As regras mapeiam conjuntos difusos (por exemplo, "junto à bandeira", "longe da base") para ações, com funções de pertença definidas para as características observadas.
- RN+CE: ideal para otimizar políticas complexas.
- RN+SD: balanço entre interpretabilidade e características aprendidas.
- CE+IE: promove especialização e o trabalho em equipa.
- SD+IE: lida com a incerteza usando regras adaptativas.
- RN+IE: estimula a aprendizagem coordenada.
- CE+SD: cria políticas interpretáveis e otimizadas.
- RN+CE+IE: cria uma política altamente adaptável. A CE desenvolve a arquitetura da RN, a IE otimiza os pesos da rede e a RN processa as observações para a seleção de ações.

4 Entrega

Cada grupo tem que entregar um ficheiro zip com:

- Um relatório em PDF. Máximo 12 páginas.
- Os dois ficheiros `agente1.py` e `agente2.py`.
- Uma apresentação em PDF do trabalho (não é powerpoint é PDF!). Máximo 10 slides.

O envio deve ser feito por email para `luis.alexandre@ubi.pt`. O prof. confirma a correta receção no prazo de 24 horas.

5 Campeonato

Na última aula será feito um campeonato onde todas as equipas irão jogar contra todas as outras e será criado uma ordenação final. Cada jogo ganho vale 3 pontos, um empate vale 1 ponto e uma derrota vale 0 pontos. A ordenação final vai dar valores para a nota no projeto: sendo n as equipas, cada uma ganha parte de 1 valor na nota final em função da sua posição na ordenação deste campeonato. A equipa que ficar em primeiro lugar ganha 1 valor na nota do trabalho. A que ficar em segundo lugar ganha $1 - 1/n$ valores, a que ficar no i-ésimo lugar ganha $1 - (i - 1)/n$ valores.