Trabalho 1 - Mundo dos Blocos

Josival S.Monteiro júnior[1]

[1]Instituto de Computação – Universidade Federal do Amazonas (UFAM)
Caixa Postal 69080-900 – Manaus – AM – Brazil

josival.salvador@icomp.ufam.edu.br

1. Respostas às Questões do Mundo dos Blocos

1.1. Descrição Geral

O projeto em Prolog implementa um ”Mundo dos Blocos”, onde blocos de diferentes tamanhos são manipulados por um agente. A estrutura do código é baseada nas técnicas descritas no capítulo 17 do livro do Bratko. O objetivo é organizar blocos em determinadas posições, respeitando restrições de estabilidade e utilizando um planejador para encontrar soluções eficientes.

1.2. Questão 1: Estado Inicial e Estado Final

No contexto do trabalho, os estados do mundo são representados pelas posições dos blocos. O estado inicial é definido pelos predicados `em(Bloco, Posicao)`, que descrevem a posição atual de cada bloco. O estado final é a meta desejada, que o planejador deve alcançar.

Por exemplo, no código Prolog:

```
em(a, 1).
em(b, 2).
em(c, 3).
```

Este código descreve o estado inicial onde o bloco `a` está na posição 1, `b` na posição 2 e `c` na posição 3.

Para o estado final, podemos definir novas posições para os blocos, como:

```
estado_final([em(a, 3), em(b, 1), em(c, 2)]).
```

O planejador, então, tem como objetivo gerar um plano de ações que mova os blocos até que o estado final seja atingido.

1.3. Questão 2: Ações de Movimento

As ações de movimento são implementadas com o predicado `move(Bloco, De, Para)`, que move um bloco de uma posição para outra, desde que certas condições sejam satisfeitas, como a posição de destino estar livre e o bloco ser suportado.

Por exemplo:

```
move(Bloco, De, Para) :-
    pode_mover(Bloco, De, Para),
    retract(em(Bloco, De)),
```

```
    assert(em(Bloco, Para)),
    write(Bloco), write(' movido de '), write(De), write('
       para '), write(Para), nl.
```

Este predicado remove o bloco da posição atual (`De`) e o coloca na nova posição (`Para`), garantindo que todas as condições de suporte e estabilidade sejam respeitadas.

1.4. Questão 3: Planejamento Heurístico

O planejamento no código utiliza uma heurística simples baseada na distância entre a posição atual do bloco e a meta. A função `heuristica(Bloco, Meta, Custo)` calcula a distância absoluta entre a posição atual do bloco e a posição final desejada.

```
heuristica(Bloco, Meta, Custo) :-
    em(Bloco, P),
    Custo is abs(P - Meta).
```

Essa função serve como uma heurística para estimar o custo de mover o bloco da posição atual para a posição final.

1.5. Questão 4: Ações de Percepção

O código inclui ações de percepção utilizando o predicado `look(Posicao, Objeto)`, que simula uma câmera ou sensor que verifica o que está presente em uma determinada posição.

```
look(Posicao, Objeto) :-
    em(Objeto, Posicao),
    write('Na posi  o '), write(Posicao), write(' h  o
       objeto: '), write(Objeto), nl.
```

Esse predicado é útil quando o robô não tem informações completas sobre o ambiente e precisa perceber o que está em cada posição.

1.6. Questão 5: Modificações no Planejador

Conforme indicado na Seção 17.5 do Bratko, o planejador foi modificado para lidar com variáveis em metas e ações. Uma das modificações é permitir metas não instanciadas inicialmente, como mover um bloco para qualquer posição livre.

No código, isso foi implementado na função `pode_mover`, que verifica se um bloco pode ser movido para uma posição livre.

```
pode_mover(Bloco, De, Para) :-
    bloco(Bloco, _),
    posicao(De), posicao(Para),
    em(Bloco, De),
    livre(Para),
    suportado(Bloco),
    De \= Para.
```

1.7. Questão 6: Tabela de Desempenho

A tabela de desempenho segue o exemplo do agente ”Taxi Driver”do livro do Russell. Aqui estão os principais componentes do ambiente e o desempenho esperado:

Componente	Descrição	Sensores	Atuadores
Desempenho	Minimizar o número de movimentos	Câmera para percepção	Braço robótico para mover blocos
Ambiente	Mundo dos blocos	Percepção de posições ocupadas	Movimentação de blocos
Sensores	Câmera e sensor tátil	Verificar estabilidade dos blocos	Confirmar posições
Atuadores	Braço robótico	Mover blocos	Posicionar blocos corretamente

1.8. Questão 7: Eficiência do Planejador

Para aumentar a eficiência do planejador, utilizamos variáveis não instanciadas em metas e ações, permitindo que a busca por soluções seja mais direcionada e eficiente, evitando a geração de alternativas desnecessárias.

2. Conclusão

Este trabalho implementa um Mundo dos Blocos em Prolog, utilizando um planejador com heurística e suporte a percepções sensoriais. A lógica foi desenvolvida para garantir que os blocos sejam movidos de forma eficiente e estável, respeitando as restrições do ambiente. A tabela de desempenho mostra os principais componentes envolvidos na execução das tarefas.