ATIVIDADE LÓGICA PARA COMPUTAÇÃO

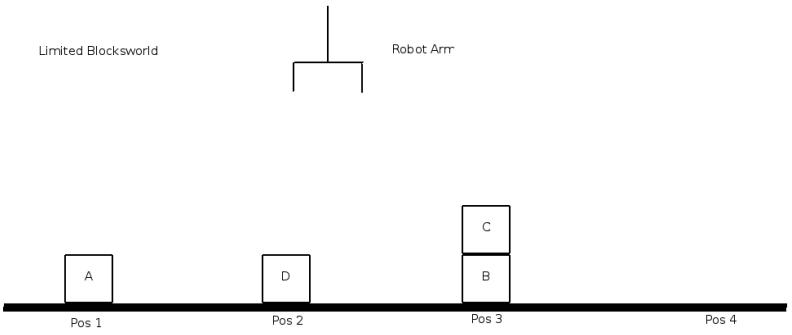
OBSERVAÇÕES IMPORTANTES:

- A atividade proposta deve ser feita com equipes de no máximo 4 membros (mais de 4 membros acarreta penalidade acumulativa de 25% na nota final por integrante a mais);
- Um relatório final com as instâncias resolvidas e o respectivo tempo final e espaço deve ser fornecido;
- Data de entrega: 05 de dezembro de 2023 (Via SIGAA).

SATPLAN

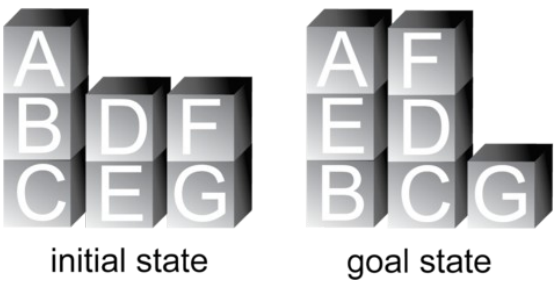
Você irá desenvolver um SATPLAN para resolver o problema do “mundo dos blocos”

O problema do mundo dos blocos consiste em um conjunto de blocos colocados sobre uma mesa (infinita em largura) onde cada bloco pode ser empilhado um sobre o outro através de um braço robótico.



Algumas restrições:
1. O braço robótico só pode segurar um bloco de cada vez;
2. O braço robótico só pode pegar um bloco se ele não possuir nenhum outro bloco acima;
3. O braço robótico, se estiver segurando um bloco, pode colocá-lo sobre a mesa ou sobre outro bloco (que por sua vez não tem outro bloco em cima);
4. Apenas um bloco pode estar imediatamente acima de outro;
5. Podemos ter uma quantidade qualquer de blocos;

Com as restrições acima, o problema do mundo dos blocos consiste em determinar os passos, dada uma configuração inicial de blocos, para se obter uma configuração final.



NOTAÇÃO

Uma descrição comum das ações possíveis, estado inicial e final é através do formato STRIPS (Stanford Research Institute Problem Solver) que é composto dos seguintes elementos:

- Um conjunto de ações, onde para cada ação temos:
    - pré-condições (que precisam ser satisfeitas para a ação poder ser usada)
    - pós-condições (que especifica o cenário que encontramos após a ação ser tomada)
- Um estado inicial
- Um estado final

Como exemplo de entrada no formato STRIPS, considere:

unstack_c_d
on_c_d;clear_c;handempty
holding_c;clear_d;~clear_c;~handempty;~on_c_d

clear_c;clear_a;clear_b;clear_d;ontable_c;ontable_a;ontable_b;ontable_d;handempty
on_d_c;on_c_b;on_b_a

Temos:

- **Nome da ação**: “unstack_c_d” (desempilhar c sobre d) é o nome da ação;
- **Precondições**: “on_c_d;clear_c;handempty” nos diz que para realizar a ação, c deve estar sobre d, c não pode ter nada acima dele e o braço robótico deve estar livre.
- **Poscondições**: “holding_c;clear_d;~clear_c;~handempty;~on_c_d” indica que após executada a ação, o braço robótico estará segurando c, o bloco d estará sem nenhum bloco acima, c não está com a parte de cima livre (por causa do braço robótico), o braço robótico não está livre e não temos mais c sobre d.

- **Estado inicial**:
  “clear_c;clear_a;clear_b;clear_d;ontable_c;ontable_a;ontable_b;ontable_d;handempty” representa o estado inicial que nos indica que c,a,b e d estão com a parte de cima livre, todos sobre a mesa e o braço robótico está vazio.
- **Estado final**: “on_d_c;on_c_b;on_b_a” é o estado final onde queremos que d esteja sobre c, c esteja sobre b e b esteja sobre a.

Observações Importantes:

1. A linha de estado inicial indica as proposições que são verdadeiras no momento no estado inicial do problema, todas as outras proposições são consideradas **falsas**.
2. A linha de estado final indica somente as proposições que queremos como **verdadeiras** (não nos interessa saber a valoração das outras).

ATIVIDADE PROPOSTA:
Dada uma entrada no formato STRIPS, encontrar uma sentença proposicional que cuja valoração que a satisfaz resolva o problema do mundo de blocos dado. Para esse intuito será utilizado um SAT-SOLVER (Use o python-sat).

DICAS:

1. Mapeie cada um dos símbolos proposicionais em inteiros positivos. Exemplo: on_c_d: 1, on_c_b : 2, on_b_a: 3, clear_d: 4, handempty: 5.
2. Na criação da sentença para o SAT-SOLVER, considere valor positivo do mapeamento se a proposição respectiva for verdadeira. Caso contrário, seu valor multiplicado por -1.
   Exemplo: on_c_d ~on_c_b on_b_a ficaria como 1 -2 3.
3. Na sequenciação das ações, considere um índice que enumere as ações a serem tomadas,exemplo:

   1_pick-up_b
   2_stack_b_a
   3_pick-up_c
   4_stack_c_b
   5_pick-up_d
   6_stack_d_c

4. Para cada índice do item anterior somente uma ação pode ser tomada, ou seja, não podemos ter 1_pick-up_b e 1_stack_c_b ambos verdadeiros na valoração-solução.
5. Outras proposições também podem ser indexadas para indicarem se elas são válidas após uma ação. Por exemplo: 1_clear_c pode ser verdadeira e após a ação 1_unstack_c_d, temos 2_clear_c como sendo falso.
6. Proposições que são verdadeiras com um índice e que não sofrem alterações com uma determinada ação, continuam verdadeiras para o índice seguinte. Exemplo: 1_ontable_a é verdadeiro e depois da ação 1_unstack_c_d temos que 2_ontable_a também é verdadeira já que a ação tomada não envolve a proposição ontable_a.
7. Utilize como solver o python-sat:
   a. página oficial: https://pysathq.github.io/
   b. exemplo de uso: https://github.com/pysathq/pysat