Informações suplementares

6 de Novembro de 2012

1. Houve actualização da data de entrega da primeira fase do projecto para 23/11, já anunciada nas aulas.

2. Depois da primeira actualização do enunciado, feita a 3/11, em que se introduziu a necessidade de fazer as funções de procura receber como argumento um problema, de forma a poderem explorar a utilização de diferentes formulações anteriormente referida no enunciado, importa estabelecer a interface a utilizar para fazer a construção do problema.

   Um problema é uma estrutura de nome `problema` com:

   - o campo `estado-inicial`, que descreve o estado inicial;
   - o campo `solucao?`, que vai conter como valor uma função que recebe um estado e reconhece se o estado é ou não uma solução;
   - campo `accoes`, que tem como valor uma função que recebe um estado e retorna as acções executáveis naquele estado;
   - campo `resultado`, que tem como valor uma função que recebe como argumentos uma acção e um estado e devolve o estado que resulta de executar a acção no estado argumento;
   - campo `custo-caminho`, que tem como valor uma função que calcula o custo de um caminho.

   Em termos de implementação, a definição do tipo problema deve ser a seguinte:

   ```
   (defstruct problema
              estado-inicial solucao? accoes resultado custo-caminho)
   ```

   Sugere-se a consulta do livro (3ª Edição, pg. 66-68) para obter esclarecimentos adicionais sobre a estrutura do tipo problema, que deve ser esta de forma a se poder usar os algoritmos do livro com adaptações mínimas.

   No exemplo do enunciado, é considerada a utilização da função `formulacao-problema`, que tem como argumento uma lista de tarefas e produz um problema. Para considerar as diferentes formulações, devem ser utilizadas diferentes funções com nomes `formulacao-problema`, `formulacao-problema1`, `formulacao-problema2`, etc, sendo a função de nome `formulacao-problema` a que vai ser executada em testes.

3. Relativamente à satisfação de restrições, a função psr recebe como argumento um problema de satisfação de restrições, representado por estrutura do tipo csp cuja definição é:

   ```
   (defstruct csp variaveis dominios restricoes)
   ```

   Deve haver a função `faz-csp` que recebe como argumento a lista de tarefas a executar e produz um csp adequado. O exemplo da chamada à função poderia ser obtida substituindo-se no exemplo da pg. 5 `pppa` por `psr` e `formulacao-problema` por `faz-csp`, obtendo-se os mesmos resultados.