ROADEF 2005
Paulo Silveira

Introdução

Apesar de não ser um problema cheio de condições diferentes, o ROADEF apresenta uma grande dificuldade quando queremos modela-lo via planejamento: não possui um estado inicial e um goal bem definido, para que possamos começar a achar uma modelagem dos operadores e funções.

Sugestões das outras pessoas

Conversando com vários colegas da sala, a grande maioria teve a idéia de modelar o problema da seguinte maneira.

- O estado inicial é uma fila de carros vazias (ou com os últimos carros do dia anterior).
- Só existe um operador, o de adicionar carro, que colocar um carro na fila atual, e incrementa os valores das penalizações.
- O goal é quando atingirmos os N carros pedidos em determinada instância do problema.
- A busca é feita por um A*, onde o custo não é o tamanho do plano, mas sim o valor das penalizações. (essa busca deve demorar muito para Ns relativamente pequenos, como visto no EP1).

Como não gosto muito de fazer algo parecido com o que os outros fizeram, resolvi adotar uma abordagem completamente diferente.

Minha idéia

A idéia principal é mudar o paradigma que um estado da nossa busca no ROADEF é uma fila que tem N carros e ainda faltam X carros a serem construídos.

Um estado agora será os N carros já enfileirados, de alguma maneira. Uma otimização interessante é já alimentar o problema com um estado inicial considerado bom, como o professor Alfredo sugeriu em sala de aula. O planejador seria apenas responsável por uma otimização numérica em cima de uma solução relativamente boa já existente.

Essa solução boa é aquela em que agrupamos todos os carros pelas suas cores.

Depois disso, temos apenas um operador de planejamento, que é o SWAP. Ele é responsável por trocar a ordem de dois carros da nossa fila atual.

A melhor técnica a se empregar seria o **hill climbing reforçado**, por uma razão clara. Queremos minimizar um custo. Enquanto estamos com uma fila de carros de custo X, não devemos sair desta fila enquanto não encontrarmos uma segunda fila de custo menor que X. Isto é, fazemos busca em largura no caso dos filhos do estado atual não serem melhores que o pai.

O goal do problema é que fica diferente. Já temos os N carros na fila, queremos apenas que seja minimizado o custo. Então o goal simplesmente perde o seu sentido nesta minha modelagem. O programa deve rodar exaustivamente durante os 5 minutos propostos pela comissão organizadora, e aí imprimir o seu estado atual, que é a melhor solução encontrada até o momento.

O problema aqui foi simplificado, já que não estamos contando com a ordem dos carros do dia anterior.

Participando do campeonato

Seria interessante participar do campeonato com um planejador, mas não precisaríamos necessariamente modelar o problema em PDDL. Em especial por causa das características diferentes apresentadas por tal problema.

Poderia apenas criar um buscador, e os operadores estão dentro do próprio programa. Seria exatamente um planejador dependente de domínio, assim podendo realizar as devidas otimizações.

PDDL resumido

Segue um esboço do PDDL para a sugestão:

```
(define (domain roadef)

        (:requirements :typing :fluents)
        (:types posicionamento)
        (:predicates
                (posicionamento ?car – carro1  ?position – posicao)


        (:functions
                (custo)
                 (position))


        (:action  swap
          :parameters (? posicionamento – carro1,posicao1  ?posicionamento –
carro2, posicao2)

          :precondition (not (carro1)
                                (= (carro2)))

          :effect (and  (not(posocionamento(carro1,posicao1 )
                                not(posicionamento carro2,posicao2))
                                (posicionamento(carro1, posicao2))
                                (posicionamento(carro2, posicao1)))
```

Precisa também somar na varável **global** custo o valor do custo atual, depois de realizado as diferentes trocas. É fácil fazer isso quando precisamos calcular apenas as penalizações em relação aos últimos carros, mas aqui houve uma troca de carros, o que faria com que a gente recalculasse para trás e para frente de cada posição. É melhor recalcular a fila inteira no planejador dependente de domínio.