Programação Dinâmica Progressiva Aplicada ao Problema de Alocação de Unidades

1. Princípio da Otimalidade em Programação Dinâmica

O princípio da otimalidade que fornece o suporte teórico à Programação Dinâmica é devido a Bellman, e é enunciado como:

> Uma estratégia é ótima se, em qualquer estágio, quaisquer que sejam as decisões precedentes, as decisões a serem ainda tomadas constituem-se em uma estratégia ótima *quando os resultados das decisões anteriores são incluídos.*

Um enunciado alternativo mais sucinto é:

> Uma estratégia ótima contém somente sub-estratégias ótimas.

2. Definições, Convenções e Hipóteses

- Uma dada *combinação de unidades* consideradas para alocação será identificada por uma cadeia binária de comprimento igual ao número de unidades, na qual **0** indica que a respectiva unidade está desligada e **1** indica que a unidade está em operação.

  Ex. - **0111** indica que, das 4 unidades consideradas, a primeira estará desligada e as unidades 2, 3 e 4 estarão em operação.

- Um *estado* é identificado por uma data combinação i em operação no instante k, e é denotado por (k, i);

- O custo de partida de uma unidades é suposto ser independente do tempo de desligamento, isto é, supõe-se que seja fixo;

- São desprezados os custos de desligamento;
- A cada intervalo de tempo considerado, uma capacidade mínima de geração deverá estar disponível;
- Uma ordem de prioridade estrita poderá ser utilizada para restringir a dimensão do problema (isto será feito no *Caso 1* a seguir).

3. Algoritmo

Sejam as seguintes componentes de custo:

$F_{cost}(k,i)$	:	Custo mínimo total para alcançar o estado (k,i);
$S_{cost}[(k-1,\ell):(k,i)]$	:	Custo de transição do estado $(k-1,\ell)$ para o estado (k,i);
$P_{cost}(k,i)$	:	Custo de produção para o estado (k,i).

O algoritmo para calcular o *custo mínimo na hora k com a combinação i* baseia-se na solução do seguinte problema de otimização via programação dinâmica:

$$F_{cost}(k,i) = \min_{\{\ell\}} \{F_{cost}(k-1,\ell) + S_{cost}[(k-1,\ell):(k,i)] + P_{cost}(k,i)\}$$

Os passos do algoritmo são dados abaixo. Considera-se que:

M	:	Número total de horas a serem consideradas;
N_x	:	Número de estados a serem analisados a cada período;
N_e	:	Número de estratégias a serem analisadas no próximo intervalo de tempo, $N_e \leq N_x$.

Algoritmo:

1. $k = 1$;
2. Para cada $i = 1 : N_x$ no período k, faça:

$$F_{cost}(k,i) = \min_{\{\ell\}} \{S_{cost}[(k-1,\ell) : (k,i)] + P_{cost}(k,i)\}$$

3. $k := k + 1$;
4. $\{\ell\} = N_e$ estados *viáveis* no intervalo $k - 1$;
5. Para cada $i = 1 : N_x$ no período k, faça:

$$\begin{aligned} F_{cost}(k,i) \quad = \quad & \min_{\{\ell\}} \{F_{cost}(k-1,\ell) + S_{cost}[(k-1,\ell) : (k,i)] \\ & P_{cost}(k,i)\} \end{aligned}$$

6. Armazene as N_e estratégias de menor custo;
7. Se $k < M$, volte para o passo 3. Se $k = M$, vá para o passo 8;
8. Recupere retroativamente a alocação ótima. **FIM.**