Programação Dinâmica

Fonte: Cormen, Thomas H.; et al. Introduction to Algorithms.MIT Press, 2001.

Programação Dinâmica

- **Divisão e Conquista**: particiona o problema em subproblemas independentes, resolve os subproblemas recursivamente, e então combinam as soluções para resolver o problema original.
- **Programação Dinâmica**:
  - resolve um problema **combinando soluções para subproblemas**.
  - **subproblemas não são independentes** (os subproblemas compartilham subproblemas). Divisão e conquista, resolveria repetidamente os subproblemas comuns.
  - resolve **cada subproblema uma vez só** e **guarda o seu resultado em uma tabela**, consultando-a toda a vez que o subproblema é requerido.

Programação Dinâmica - Aplicação

Problemas de Otimização

- várias soluções possíveis;
- busca-se uma solução com valor ótimo (mínimo ou máximo);

Solução em Programação Dinâmica:

1. Caracteriza-se a estrutura de uma solução ótima.
2. Define-se recursivamente o valor de uma solução ótima.
3. Calcula-se o valor de uma solução ótima em um processo *bottom-up*.
4. Constrói-se a solução ótima a partir das informações já calculadas.

Programação Dinâmica - Aplicação

Problemas de Otimização

- várias soluções possíveis;
- busca-se uma solução com valor ótimo (mínimo ou máximo);

Etapas 1 a 3 formam a BASE da Programação Dinâmica:

1. Caracteriza-se a estrutura de uma solução ótima.
2. Define-se recursivamente o valor de uma solução ótima.
3. Calcula-se o valor de uma solução ótima em um processo *bottom-up*.
4. Constrói-se a solução ótima a partir das informações já calculadas.

Etapa 4 pode ser omitida.

Problemas de Otimização

- Programação de linhas de montagem.
- Multiplicação de matrizes com o menor número de multiplicações escalares.
- Subseqüência comum mais longa.

Elementos da Programação Dinâmica

O que um problema de otimização deve ter para que a programação dinâmica seja aplicável?

- subestrutura ótima;
- superposição de problemas.

Subestrutura Ótima

Uma solução ótima para o problema **contém** em seu interior **soluções ótimas para subproblemas**.

Constrói-se uma solução ótima para um problema a partir de soluções ótimas para subproblemas.

Subestrutura Ótima

Padrão comum na descoberta de sub. ótima:

1. Mostrar que uma solução para o problema consiste em fazer uma escolha $\Rightarrow$ um ou mais subproblemas devem ser resolvidos.
2. Supor que, para um dado problema, temos a escolha que conduz a solução ótima (não importa como esta escolha é determinada; supõe-se dada);
3. Dada esta escolha, determinar quais subproblemas resultam dela e como deve ser caracterizado o espaço de subproblemas resultante.
4. Mostrar que as soluções para os subproblemas usados dentro da solução ótima para o problema devem ser elas próprias ótimas.

Subestrutura Ótima

Mostrar que as soluções para os subproblemas usados dentro da solução ótima para o problema devem ser elas próprias ótimas.

"recortar e colar"

- Suponha que cada uma das soluções dos subproblemas não é ótima;
- "corte" a solução não ótima;
- "cole" a solução ótima;
- obtém-se uma solução melhor para o problema original. Deriva-se uma contradição: contradiz-se a otimalidade da solução do problema (a hipótese de que se tinha uma solução ótima).

Subestrutura Ótima

Quem fornece a escolha que conduz a solução ótima? (supomos dada no passo 2)

Tente todas as escolhas, resolva todos os subproblemas resultante de cada escolha, e pegue a escolha cuja solução é a melhor

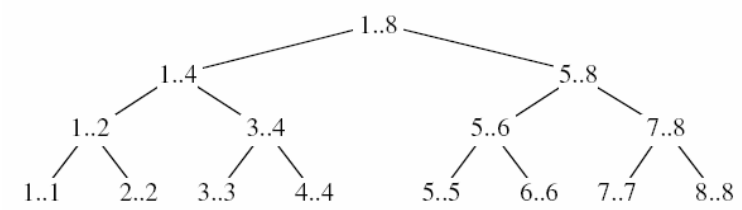
Como caracterizar o espaço de subproblemas?

- Mantenha o espaço tão simples quanto possível;
- Expanda ele se necessário.

Subproblemas superpostos

Ocorre quando um algoritmo recursivo reexamina o mesmo problema inúmeras vezes.

Bons algoritmos de divisão e conquista geram um novo problema a cada estágio da recursão. Ex.: MergeSort



Subproblemas superpostos

Algoritmos de PD:

- Resolve cada subproblema uma única vez;
- Armazena a solução em uma tabela;
- Consulte-a quando necessário.

Agradecimento à Profa. Simone Costa