Recursão

- A idéia de qualquer algoritmo recursivo é simples:
  - Se a instancia do problema é pequena, resolva-a diretamente.
  - Se a instancia é grande, reduza-a a uma instância menor do mesmo problema
- Embora possa confundir programadores iniciantes, usar recursividade com eficiência é uma marca de um bom programador
- Grande importância na Computação
  - Linguagens de programação puramente recursivas
    - Prolog
    - *Lisp* (Auto-cad é programado em *Lisp*).

Recursão

- Um método recursivo é capaz de resolver apenas os casos básicos
  - Caso mais simples de um problema (caso base ou caso básico)
- Se a chamada do método for um caso básico, o método retorna resultado

Recursão

- Se o método for chamado para um caso mais complexo...
  - Ele divide o problema em duas partes conceituais
    - Uma que o método sabe resolver
    - Outra que o método não sabe resolver
  - A parte do problema que o método não sabe resolver é uma versão mais simples do problema original
  - O método chama uma nova cópia dele mesmo para resolver o problema menor (***chamada recursiva*** ou ***passo de recursão***)
  - O processo se repete até que reste apenas o caso base
  - Ao final de cada *passo de recursão* o método retorna o resultado que é combinado com a parte do problema que o método chamador já havia solucionado para dessa forma chegar a solução do problema todo

Definições Recursivas

- Os predicados em Prolog podem ser definidos recursivamente
- Um predicado é definido recursivamente se uma ou mais regras em sua definição refere-se a ela mesma

Definições Recursivas

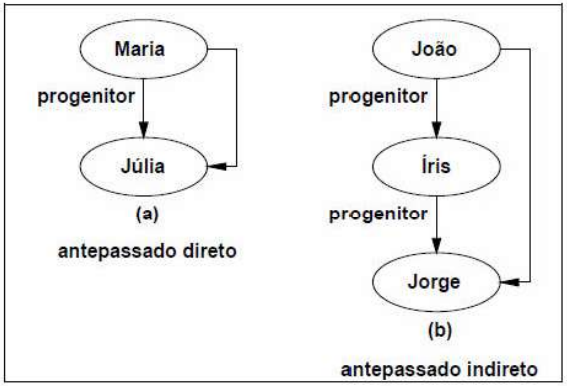

Figura 2.3 Exemplos da relação *antepassado*

Luis, A. M. Palazzo, Introdução à Programação Prolog, Educat, 1997.

Definições Recursivas

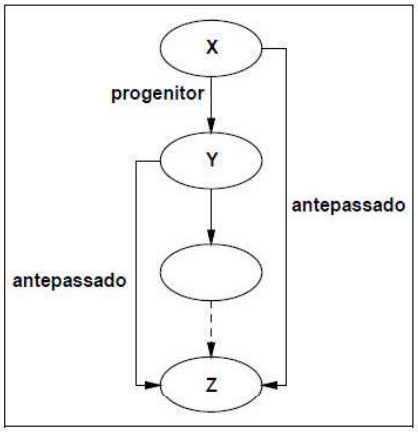

Figura 2.4 Formulação recursiva da relação *antepassado*

Luis, A. M. Palazzo, Introdução à Programação Prolog, Educat, 1997.

Exemplo 1: Descendente

```
filho(brigite,caroline).
filho(caroline,donna).

descende(X,Y):- filho(X,Y).
descende(X,Y):- filho(X,Z), filho(Z,Y).
```

Exemplo 1: Descendente

```
filho(ana,brigite).
filho(brigite,caroline).
filho(caroline,donna).
filho(donna,emilia).


descende(X,Y):- filho(X,Y).
descende(X,Y):- filho(X,Z), filho(Z,Y).
```

Exemplo 1: Descendente

```
filho(ana,brigite).
filho(brigite,caroline).
filho(caroline,donna).
filho(donna,emilia).


descende(X,Y):- filho(X,Y).
descende(X,Y):- filho(X,Z), filho(Z,Y).
```

```
?- descende(ana,donna).
no
?-
```

Exemplo 1: Descendente

```
filho(ana,brigite).
filho(brigite,caroline).
filho(caroline,donna).
filho(donna,emilia).


descende(X,Y):- filho(X,Y).
descende(X,Y):- filho(X,Z), filho(Z,Y).
descende(X,Y):- filho(X,Z), filho(Z,U), filho(U,Y).
```

```
?-
```

Exemplo 1: Descendente

```
filho(ana,brigite).
filho(brigite,caroline).
filho(caroline,donna).
filho(donna,emilia).


descende(X,Y):- filho(X,Y).
descende(X,Y):- filho(X,Z), descende(Z,Y).
```

```
?-
```

Exemplo 1: Descendente

```
filho(ana,brigite).
filho(brigite,caroline).
filho(caroline,donna).
filho(donna,emilia).


descende(X,Y):- filho(X,Y).
descende(X,Y):- filho(X,Z), descende(Z,Y).
```

```
?- descende(ana,donna).
```

Árvore de Busca

- Desenhe a árvore de busca para

  ?- descende(ana,donna).