Matemática Discreta para Ciência da Computação

P. Blauth Menezes

blauth@inf.ufrgs.br

Departamento de Informática Teórica

Instituto de Informática / UFRGS

Matemática Discreta para Ciência da Computação

P. Blauth Menezes

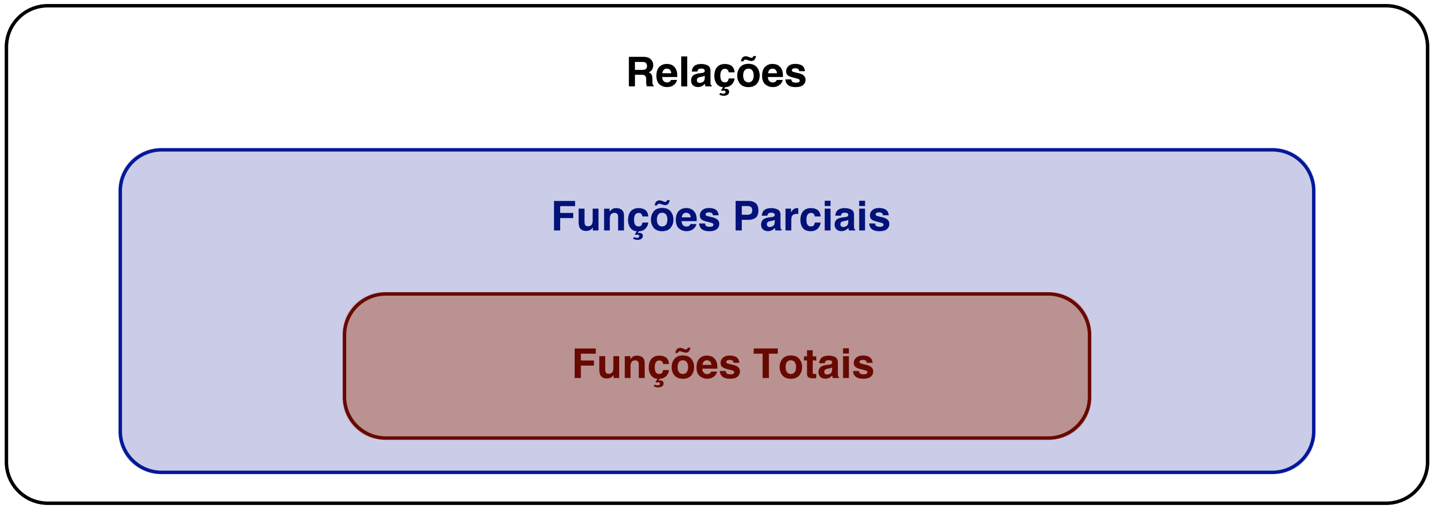
5 – Funções Parciais e Totais

5 Funções Parciais e Totais

- **Função Parcial**
  - simplesmente relação funcional
- **Função Total ou Função**
  - relação funcional total

◆ Portanto:

- *função parcial* é relação
- *função* é função parcial (portanto, é relação)
- *nem* toda relação é função parcial
  - ∗ basta considerar uma relação não-funcional
- *nem* toda função parcial é uma função
  - ∗ basta considerar uma função parcial não-total



◆ Estudo das funções

- destacado do estudo das relações
- importante para a Matemática e Computação e Informática
- maioria das abordagens matemáticas
  - centradas no conceito de função total
- em Computação e Informática
  - função parcial é *tão ou mais importante* que função
- computablidade
  - noção mais fundamental em CC
  - baseada em funções *parciais*

5 – Funções Parciais e Totais

5.1 Função Parcial
- 5.1.1 Definição e Introdução
- 5.1.2 Função Parcial Dual
- 5.1.3 Composição de Funções Parciais
- 5.1.3 Restrição

5.2 Autômato Finito

5.3 Função Total

5.4 Construções Matemáticas como Funções

5.5 Função de Hashing

5.5 Funções nas Linguagens de Programação

5.7 Linguagem de Programação Funcional

5.1 Função Parcial

5.1.1 Definição e Introdução

Def: Função Parcial

Função Parcial é uma relação funcional $f \subseteq A \times B$

- **Portanto, função parcial é relação na qual**
  - cada elemento do domínio está relacionado com, *no máximo*, um elemento do contra-domínio

◆ Notação

- função parcial $f \subseteq A \times B$

$$f\colon A \nrightarrow B$$

- ou, quando é claro que se trata de uma função parcial

$$f\colon A \to B$$

- $\langle a, b \rangle \in f$

$$f(a) = b \quad \text{ou} \quad f\langle a \rangle = b$$

◆ Termos alternativos para função parcial

- operação parcial
- mapeamento parcial
- transformação parcial

◆ Exemplos de relações funcionais

- são exemplos de funções parciais

Exp: Função Parcial

$A = \{ a \}$, $B = \{ a, b \}$ e $C = \{ 0, 1, 2 \}$

- $\varnothing: A \to B$ ✔
- $\{ \langle 0, a \rangle, \langle 1, b \rangle \}: C \to B$ ✔
- $=: A \to B$ ✔
- $x^2: \mathbf{Z} \to \mathbf{Z}$ onde $x^2 = \{ \langle x, y \rangle \in \mathbf{Z}^2 \mid y = x^2 \}$ ✔

- $A \times B: A \to B$ ✘
- $<: C \to C$ ✘

◆ Grafo e matriz de uma função parcial?

◆ Função parcial como matriz ou grafo (endorrelação)

- matriz: no maximo um valor verdadeiro em cada *linha*
- grafo: no máximo uma aresta *partindo* de cada nodo

Exp: Função Parcial

Adição nos naturais. Operação ad: $\mathbf{N} \times \mathbf{N} \to \mathbf{N}$ tal que

$$\mathrm{ad}\langle a, b \rangle = a + b$$

- conjunto imagem?

Divisão nos reais. Operação div: $\mathbf{R} \times \mathbf{R} \to \mathbf{R}$ tal que:

$$\mathrm{div}\langle x, y \rangle = x / y$$

- conjunto imagem?

5 – Funções Parciais e Totais

5.1 Função Parcial
- 5.1.1 Definição e Introdução
- 5.1.2 Função Parcial Dual
- 5.1.3 Composição de Funções Parciais
- 5.1.3 Restrição

5.2 Autômato Finito

5.3 Função Total

5.4 Construções Matemáticas como Funções

5.5 Função de Hashing

5.6 Funções nas Linguagens de Programação

5.7 Linguagem de Programação Funcional

5.1.2 Função Parcial Dual

- **Relação dual de uma função parcial**
  - *não* necessariamente é uma função parcial (por quê?)

Exp: Relação Dual de Função Parcial

$A = \{ 0, 1, 2 \}$ e endofunção parcial $f: A \to A$ tq $f = \{ \langle 0, 2 \rangle, \langle 1, 2 \rangle \}$

$$f^{op} = \{ \langle 2, 0 \rangle, \langle 2, 1 \rangle \}$$

- **Conceito dual de funcional?**

- **Lembre-se: conceito dual de funcional é injetora**
- **Dual de função parcial é função parcial?**
  - deve ser injetora
- **Ou seja**
  - relação dual de uma relação funcional e injetora
  - é relação injetora e funcional
- **Conclusão**
  - relação dual de função parcial injetora
  - é função parcial injetora.

Exp: Relação Dual de Função Parcial

$A = \{ a \}$, $B = \{ a, b \}$ e $C = \{ 0, 1, 2 \}$. Duais são funções parciais?

- $\varnothing: A \to B$ ✔
- $=: A \to B$ ✔
- $\{ \langle 0, a \rangle, \langle 1, b \rangle \}: C \to B$ ✔
- $x^2: \mathbf{Z} \to \mathbf{Z}$ onde $x^2 = \{ \langle x, y \rangle \in \mathbf{Z}^2 \mid y = x^2 \}$ ✘

5 – Funções Parciais e Totais

5.1.3 Composição de Funções Parciais

- **Composição de relações é relação**
  - por definição
- **Composição de funções parciais é função parcial?**
  - basta mostrar que a composição de funcionais é funcional

Teorema: Composição de Funcionais é Funcional

$R: A \to B$ e $S: B \to C$ relações funcionais

Então $S \circ R: A \to C$ *é* uma relação funcional

Prova: Composição de Funcionais é Funcional

Suponha $R: A \to B$ e $S: B \to C$ relações funcionais

Então $S \circ R: A \to C$ é relação e basta provar que

$$(\forall a \in A)(\forall c_1 \in C)(\forall c_2 \in C)(a\,(S \circ R)\,c_1 \wedge a\,(S \circ R)\,c_2 \to c_1 = c_2)$$

Suponha $a \in A$, $c_1 \in C$ e $c_2 \in C$ tais que $a\,(S \circ R)\,c_1 \wedge a\,(S \circ R)\,c_2$

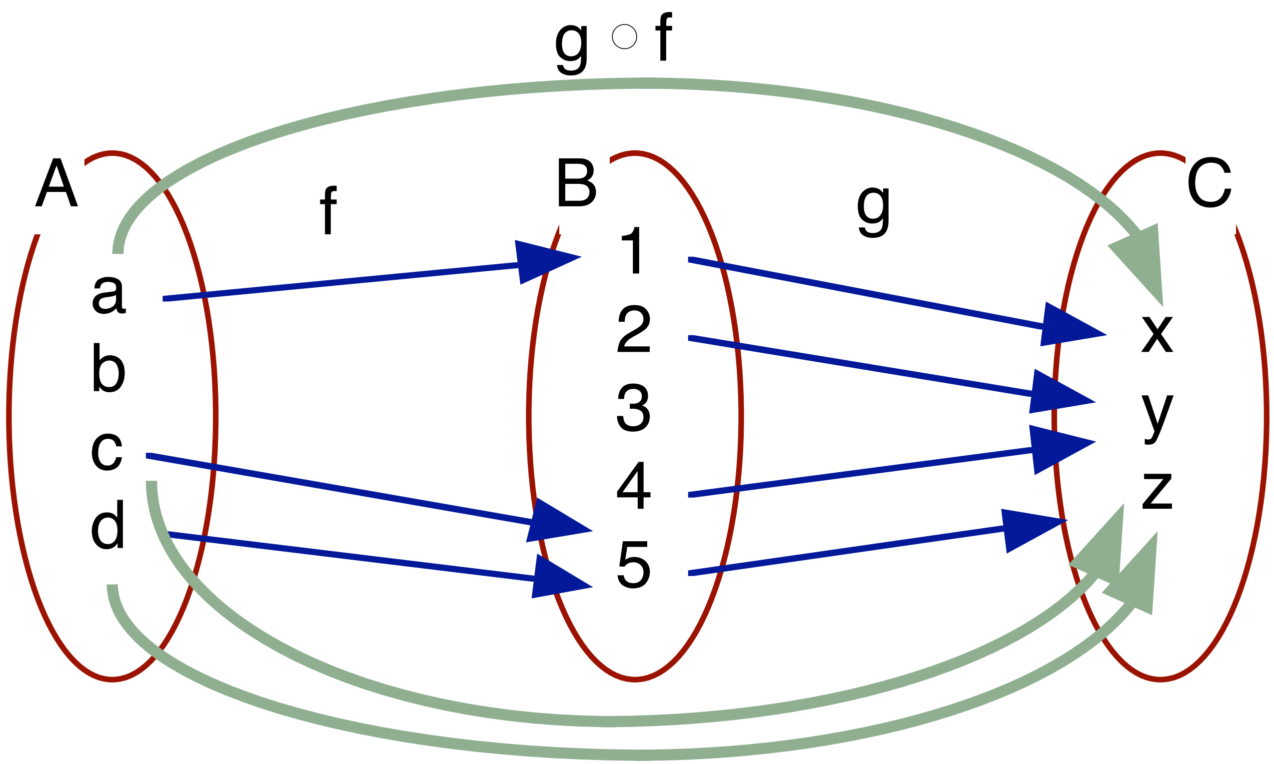
Então (prova *direta*)

- $a\,(S \circ R)\,c_1 \wedge a\,(S \circ R)\,c_2 \Rightarrow$ definição de composição
- $(\exists b_1 \in B)(\exists b_2 \in B)(a\,R\,b_1 \wedge a\,R\,b_2 \wedge b_1\,S\,c_1 \wedge b_2\,S\,c_2) \Rightarrow$ R é func.
- $b_1 = b_2 \wedge b_1\,S\,c_1 \wedge b_2\,S\,c_2 \Rightarrow$ S é funcional
- $c_1 = c_2$

Logo, $S \circ R: A \to C$ é relação funcional

Exp: Composição de Relações

- $f = \{ \langle a, 1 \rangle, \langle c, 4 \rangle, \langle d, 5 \rangle \}$
- $g = \{ \langle 1, x \rangle, \langle 2, y \rangle, \langle 4, y \rangle, \langle 5, z \rangle \}$
- $g \circ f = \{ \langle a, x \rangle, \langle c, z \rangle, \langle d, z \rangle \}$



Obs: Dualidade e Prova de Teoremas

Fato extremamente *importante*

- todo o resultado válido
- tambem é válido para o seu conceito dual
- prova é praticamente a mesma, respeitando as noções duais

Não é tão evidente na Teoria dos Conjuntos

- amplamente explorado na Teoria das Categorias

a noção de dualidade divide o trabalho pela metade

- incluindo definições e provas

◆ Por dualidade

- composição de relações injetoras é uma relação injetora

Teorema: Composição de Injetoras é Injetora

$R: A \to B$ e $S: B \to C$ relações injetoras

Então $S \circ R: A \to C$ é uma relação injetora

◆ Prova

- exercício: *dualizar* prova anterior

5 – Funções Parciais e Totais

5.1 Função Parcial
- 5.1.1 Definição e Introdução
- 5.1.2 Função Parcial Dual
- 5.1.3 Composição de Funções Parciais
- 5.1.3 Restrição

5.2 Autômato Finito

5.3 Função Total

5.4 Construções Matemáticas como Funções

5.5 Função de Hashing

5.6 Funções nas Linguagens de Programação

5.7 Linguagem de Programação Funcional

5.1.4 Restrição

- **Para uma função parcial**
  - restrição, a partir de um subconjunto de seu domínio
- **Operação (sobre funções)**
  - importante quando aplicada sobre sistemas
  - uma das operações fundamentais da álgebra de processos

Def: Restrição do Domínio de uma Função Parcial

$f: A \to B$ função parcial e A_0 tal que $A_0 \subseteq A$

Restrição do Domínio de f relativamente a A_0

$$f \backslash A_0: A_0 \to B$$

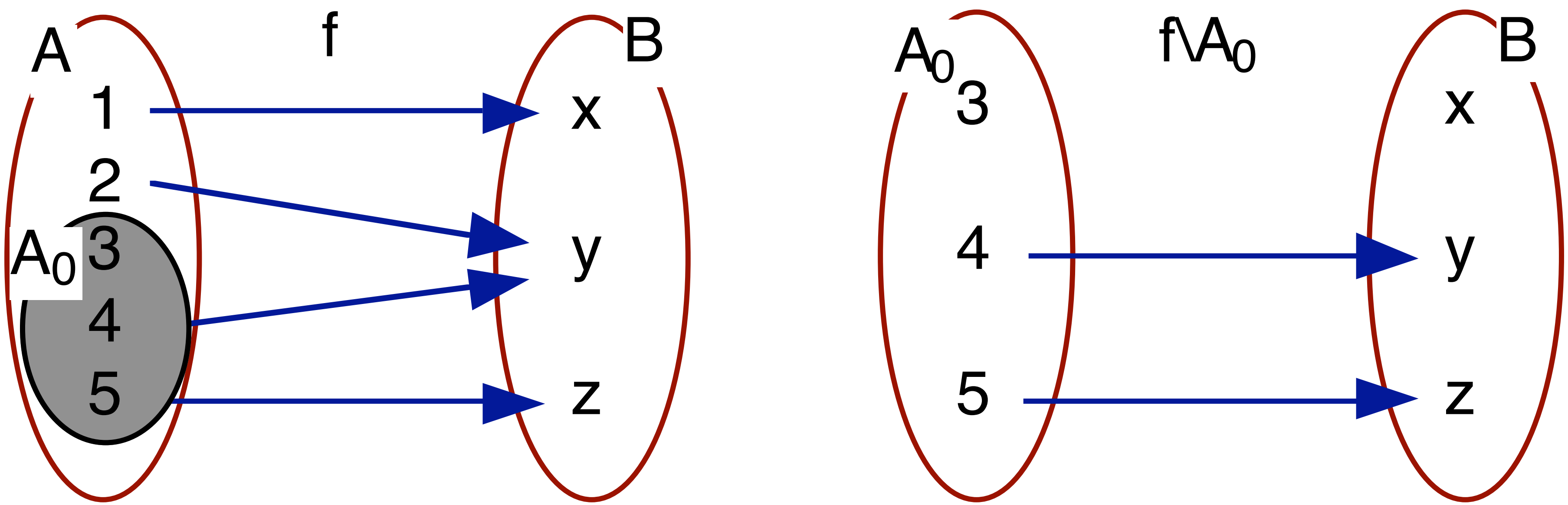
é tal que

$$f \backslash A_0 = f \cap (A_0 \times B)$$

Exp: Restrição do Domínio de uma Função Parcial

$A = \{ 1, 2, 3, 4, 5 \}$, $B = \{ x, y, z \}$ e a função parcial $f: A \rightarrow B$

Para $A_0 = \{ 3, 4, 5 \}$, a função parcial $f \backslash A_0: A_0 \rightarrow B$



Exp: Restrição do Domínio de uma Função Parcial

$A = \{ a \}$, $B = \{ a, b \}$ e $C = \{ 0, 1, 2 \}$

$\varnothing: A \to B$

$\varnothing \backslash A = \varnothing: A \to B$

$R = \{ \langle 0, a \rangle, \langle 1, b \rangle \}: C \to B$

$R \backslash \{ 0 \} = \{ \langle 0, a \rangle \}: \{ 0 \} \to B$

$id_B = \{ \langle a, a \rangle, \langle b, b \rangle \}: B \to B$

$id_B \backslash A = \{ \langle a, a \rangle \}: A \to B$

$x^2 = \{ \langle x, y \rangle \in \mathbf{Z}^2 \mid y = x^2 \}: \mathbf{Z} \to \mathbf{Z}$

$x^2 \backslash \mathbf{N} = \{ \langle x, y \rangle \in \mathbf{N} \times \mathbf{Z} \mid y = x^2 \}$

- **Restrição introduzida foi sobre o domínio**
  - Como seria sobre o contra-domínio? Exercício
- **Restrição de sistemas**
  - exemplificação em autômatos finitos

5 – Funções

5.1 Função Parcial
5.2 Autômato Finito
  5.2.1 Modelo e Exemplo
  5.2.2 Autômato Finito como Função Parcial
  5.2.3 Restrição de um Autômato Finito
  5.2.3 Leitura Complementar
5.3 Função Total
5.4 Construções Matemáticas como Funções
5.5 Função de Hashing
5.6 Funções nas Linguagens de Programação
5.7 Linguagem de Programação Funcional

5.2 Autômato Finito

◆ Autômato Finito

- sistema de estados *finitos*
- modelo computacional do tipo seqüencial muito comum
- usado em diversos estudos
  - ∗ Linguagens Formais, Compiladores
  - ∗ Semântica Formal, Teoria da Concorrência, …
- conceito de autômato finito introduzido (via exemplos)
  - ∗ baseado em Linguagens Formais
  - ∗ usados para verificar se (w - palavra, L - linguagem)

$$w \in L \quad \text{ou} \quad w \notin L$$

5 – Funções

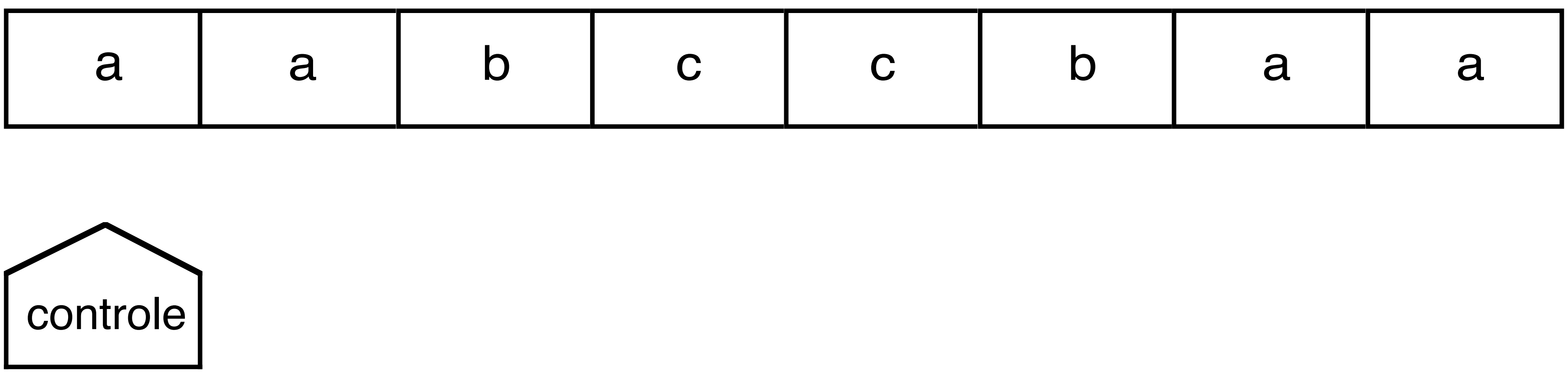
5.2.1 Modelo e Exemplo

- **Autômato Finito: máquina composta por**
  - Fita
  - Unidade de Controle
  - Programa

- **Fita: dispositivo de entrada**
  - contém a informação a ser processada
  - finita, dividida em células: cada célula armazena um símbolo
  - símbolos: pertencem a um alfabeto de entrada
  - *não* é possível gravar sobre a fita

♦ Unidade de Controle: reflete o estado corrente

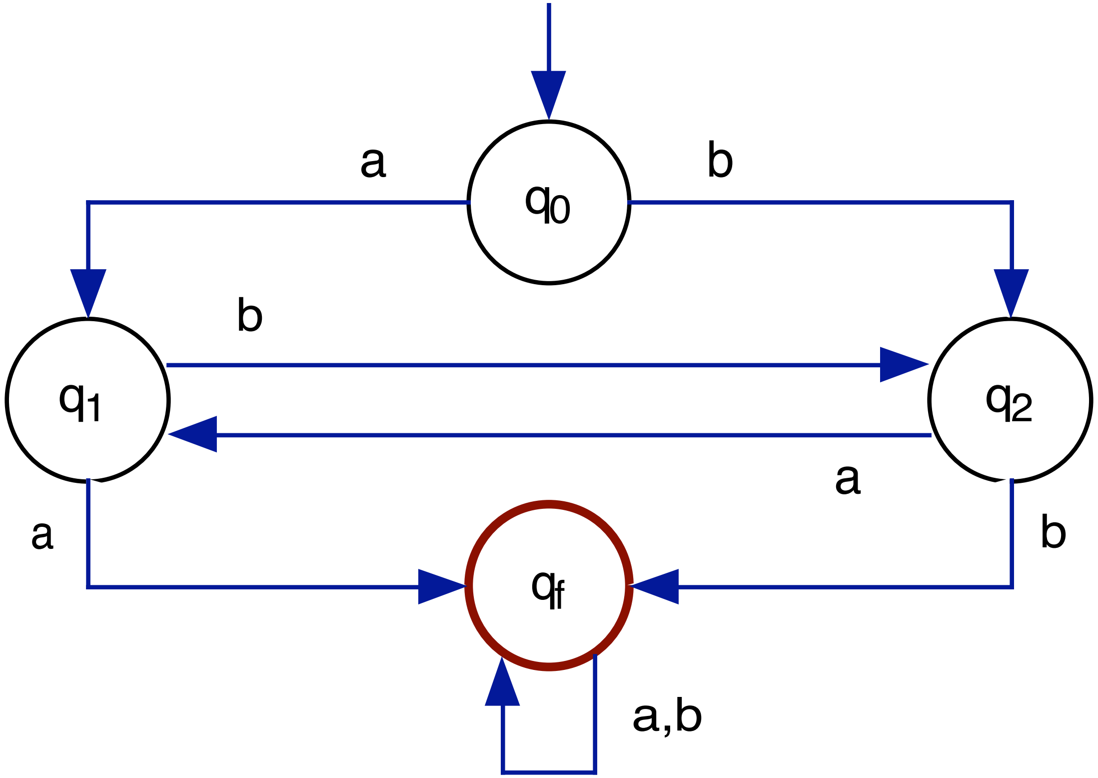
- Estados
  - número de estados: finito e predefinido
- Unidade de Leitura
  - inicialmente: cabeça na célula mais à esquerda da fita
  - lê o símbolo de uma célula de cada vez
  - após a leitura, move a cabeça uma célula para a direita



♦ Programa: função *parcial*

- comanda as leituras
- define o estado da máquina
  - ∗ dependendo do estado corrente e símbolo lido
  - ∗ determina o novo estado

Exp: Autômato: aa ou bb como subpalavra

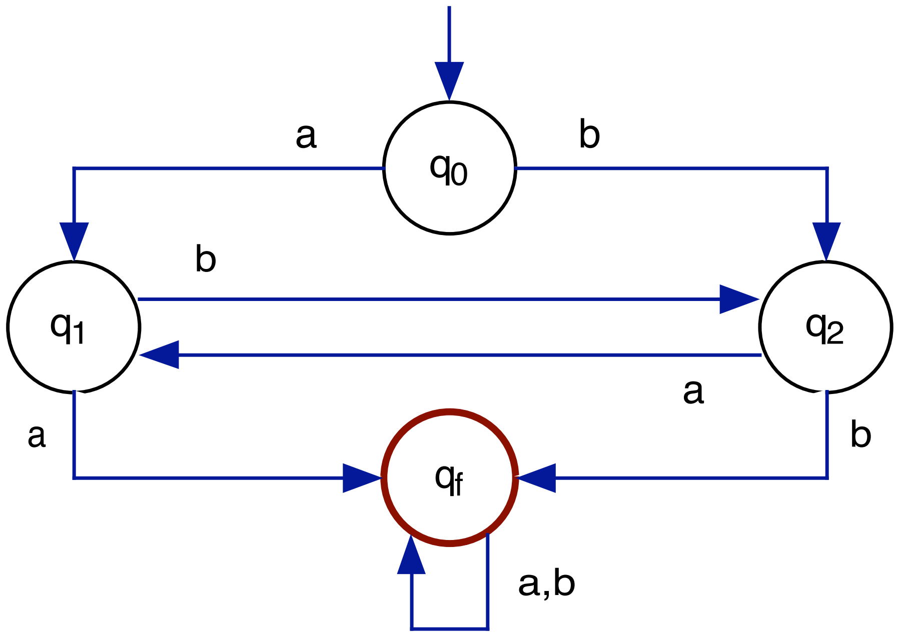


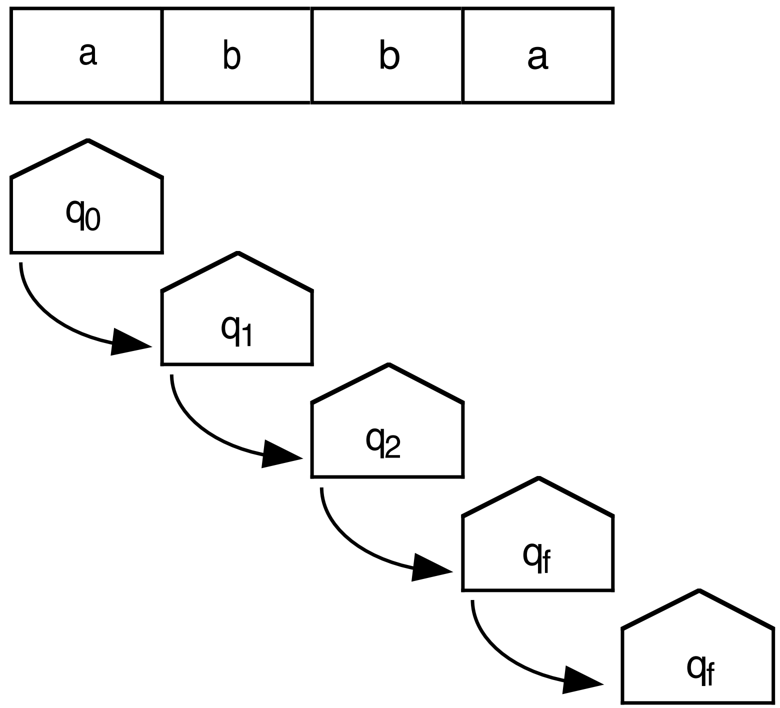
- nodos: estados; q_0 - estado inicial; q_f - estado final
- arcos: transições ou computações atômicas
- processamento: sucessiva aplicação de computações atômicas

Exp: Autômato: aa ou bb como subpalavra

Linguagens Formais: o autômato pára (normalmente)

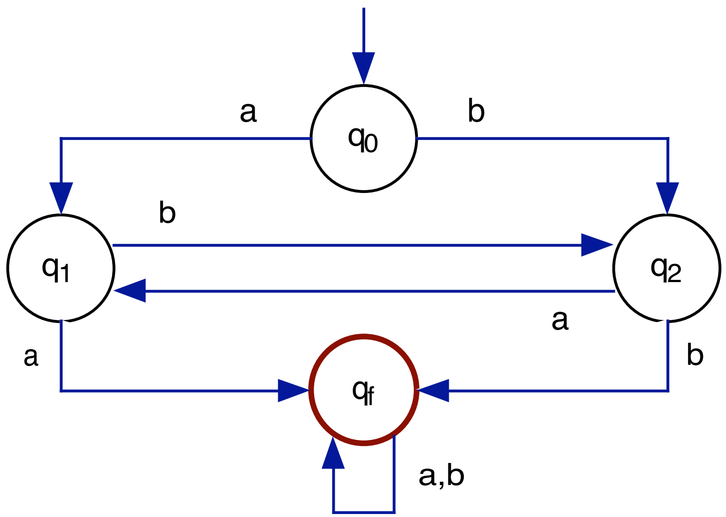
- quando processar *toda* a entrada
- *aceita* a entrada se parar em um estado final





6 – Funções

5.2.2 Autômato Finito como Função Parcial



- **Autômato finito definido como função parcial**
  - no estado q_0, ao ler a, assume o estado q_1
  - no estado q_0, ao ler b, assume o estado q_2

  - $\langle\langle q_0, a\rangle, q_1\rangle$
  - $\langle\langle q_0, b\rangle, q_2\rangle$

- **Assim, em cada par ordenado da forma $\langle\langle q, a\rangle, p\rangle$**

  - componentes
    - primeira: o par $\langle$estado corrente, símbolo lido$\rangle$
    - segunda: novo estado

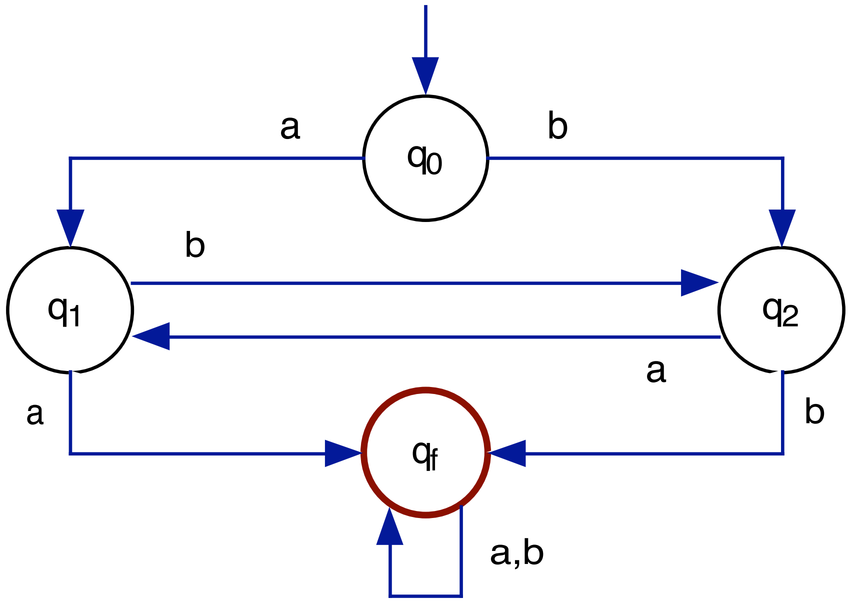
  - função *parcial*

$$\delta: Q \times \Sigma \to Q$$

    - Q - conjunto finito de estados Σ - alfabeto
    - $\langle\langle q, a\rangle, p\rangle$ é tal que $\delta(\langle q, a\rangle) = p$

◆ Todos os pares que definem a função programa

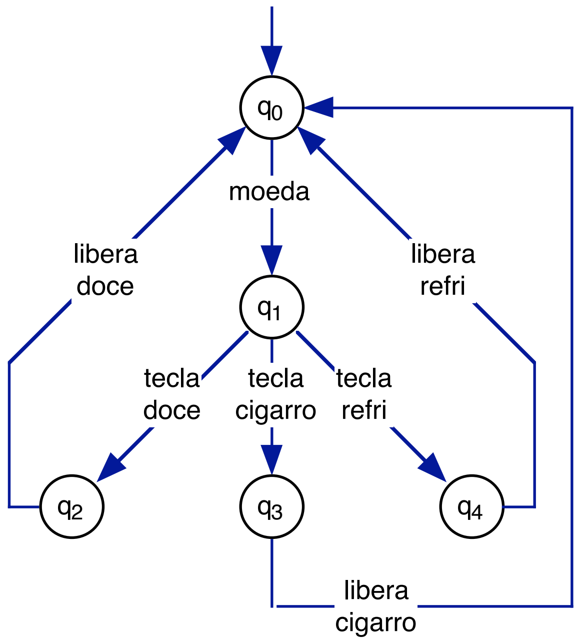
- Total? Injetora? Sobrejetora?



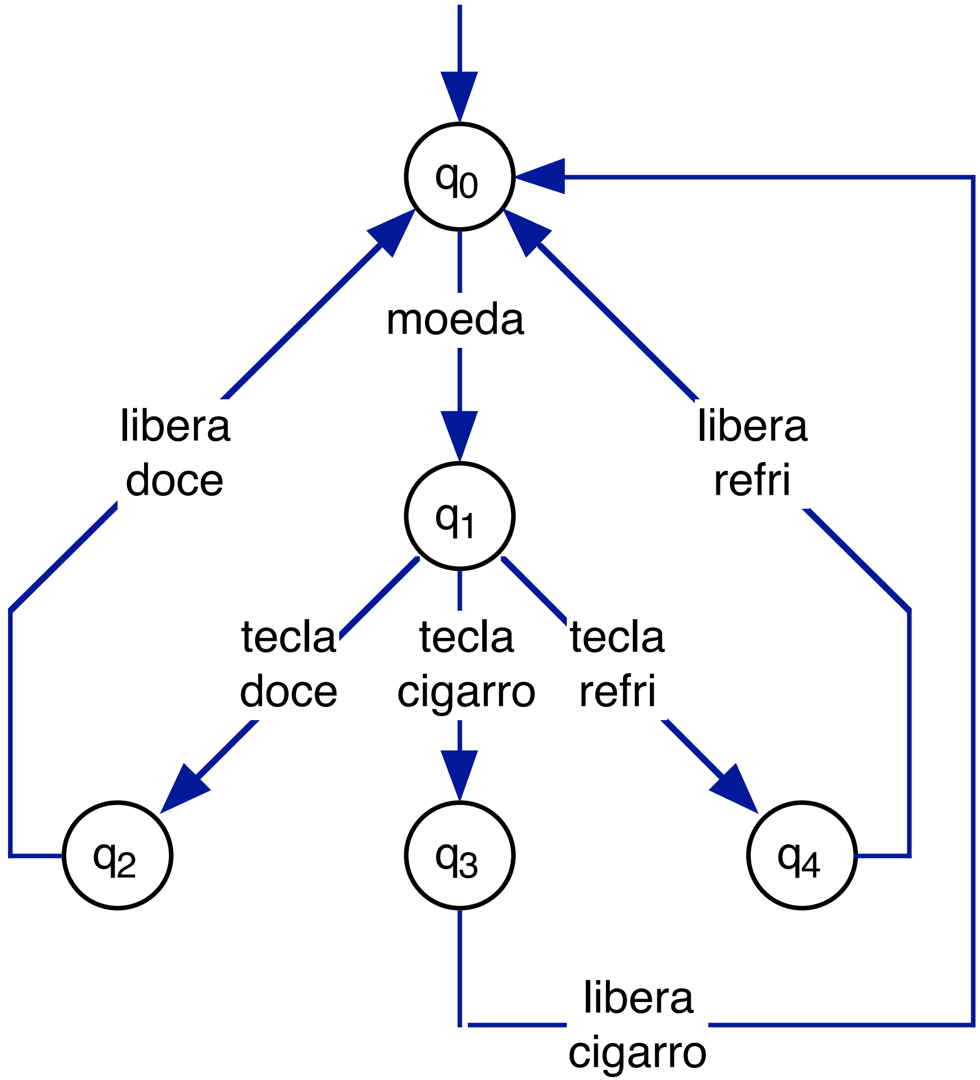
$\langle\langle q_0, a\rangle, q_1\rangle$	$\langle\langle q_1, a\rangle, q_f\rangle$	$\langle\langle q_2, a\rangle, q_1\rangle$	$\langle\langle q_f, a\rangle, q_f\rangle$
$\langle\langle q_0, b\rangle, q_2\rangle$	$\langle\langle q_1, b\rangle, q_2\rangle$	$\langle\langle q_2, b\rangle, q_f\rangle$	$\langle\langle q_f, b\rangle, q_f\rangle$

Exp: Autômato Finito como Interface Homem × Máquina

- $\delta: Q \times \Sigma \to Q$ total? injetora? sobrejetora?
  * $Q = \{ \langle q_0, q_1, q_2, q_3, q_4 \}$
  * Σ = { moeda, tecla_doce, tecla_cigarro, tecla_refri, libera_doce, libera_cigarro, libera_refri }



Exp: Autômato Finito como Interface Homem × Máquina



5 – Funções Parciais e Totais

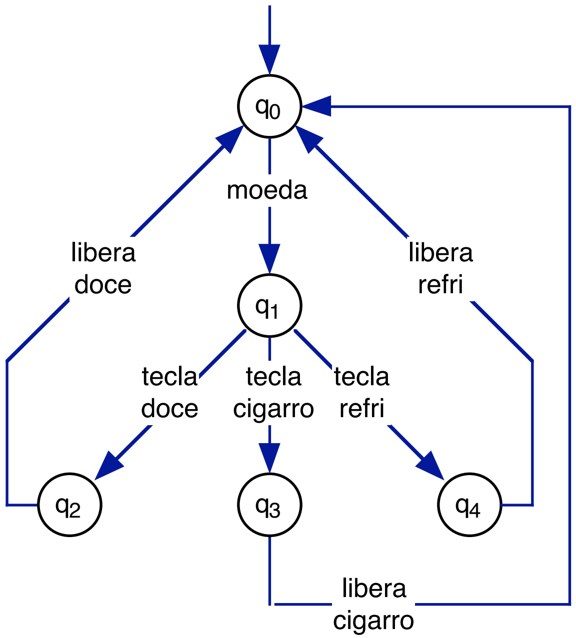
5.1 Função Parcial
5.2 Autômato Finito
    5.2.1 Modelo e Exemplo
    5.2.2 Autômato Finito como Função Parcial
    5.2.3 Restrição de um Autômato Finito
    5.2.3 Leitura Complementar
5.3 Função Total
5.4 Construções Matemáticas como Funções
5.5 Função de Hashing
5.6 Funções nas Linguagens de Programação
5.7 Linguagem de Programação Funcional

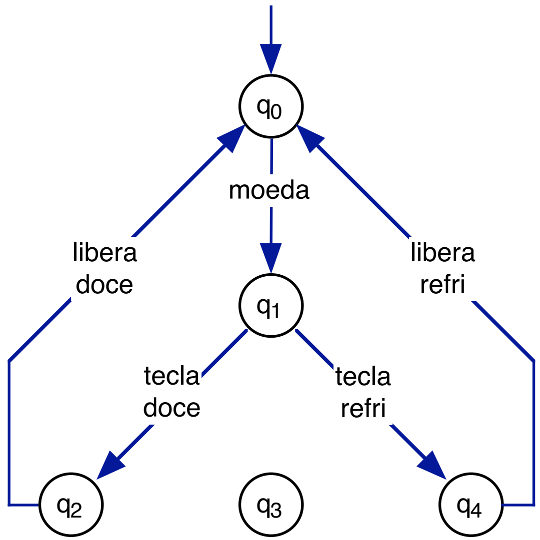
5.2.3 Restrição de um Autômato Finito

◆ Cálculo de restrição de sistemas

- importante aplicação da operação de restrição de funções parciais
- reuso de *software*: importante no estudo de
  - ∗ Engenharia de *Software*
  - ∗ paradigma Orientação a Objetos

Exp: Restrição de Autômato Finito × Reuso de Software





Desejada uma nova máquina, *sem* as funções relacionadas com cigarros

$$\delta \backslash Q \times \Sigma_0 : Q \times \Sigma_0 \rightarrow Q$$

Σ_0 = { moeda, tecla_doce, tecla_refri, libera_doce, libera_refri }

Obs: Manutenção de Software

Restrição de *software* pode facilmente ser implementada

- ferramenta automática de desenvolvimento/manutenção
- programador *não* altera o *software*
  - realiza uma operação sobre este
  - resultado desejado: *garantido*

Custo de manutenção de *software*

- freqüentemente é maior que o de desenvolvimento
- baixa confiabilidade de um *software* alterado (pelo programador)

Portanto, ferramentas automáticas de manutenção/reuso de *software*

- fundamental importância

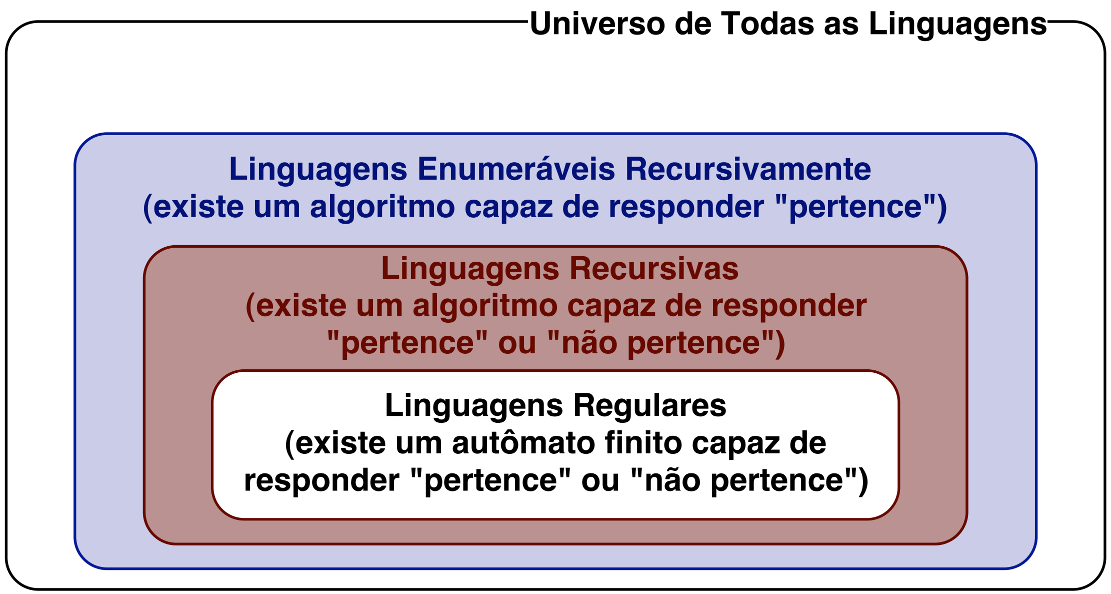
5 – Funções Parciais e Totais

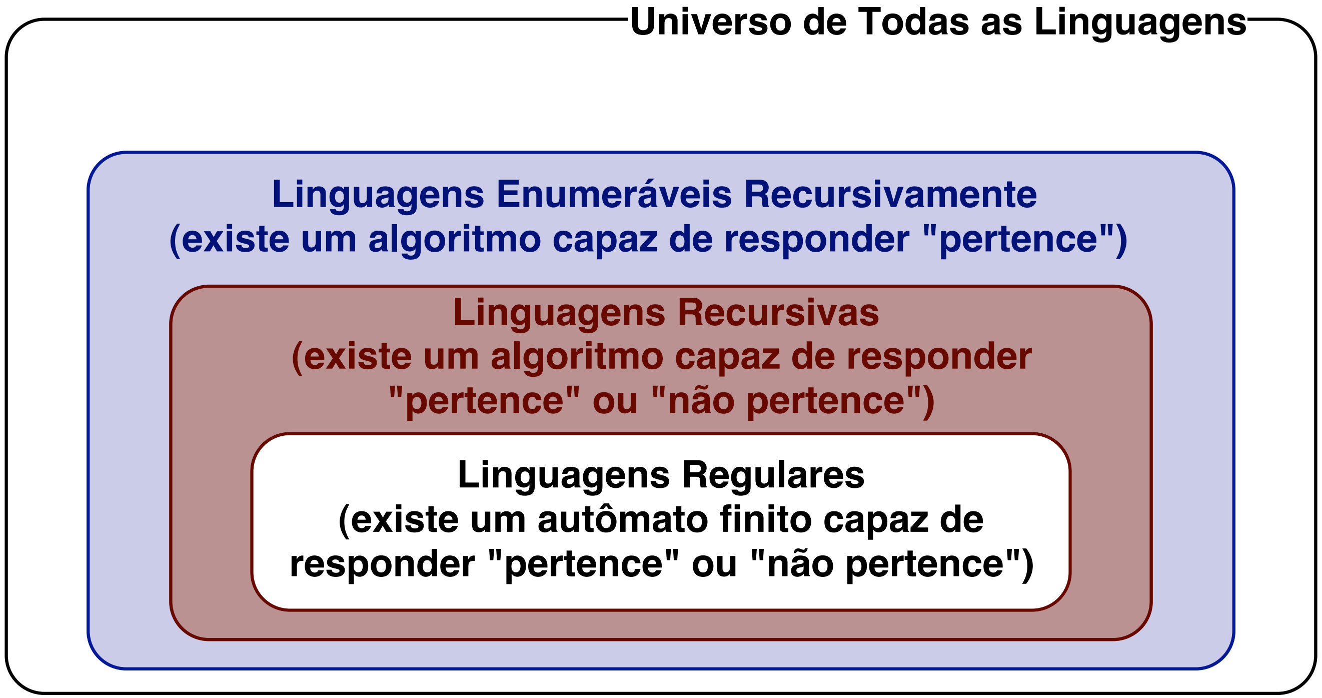
5.1 Função Parcial
5.2 Autômato Finito
  5.2.1 Modelo e Exemplo
  5.2.2 Autômato Finito como Função Parcial
  5.2.3 Restrição de um Autômato Finito
  5.2.3 Leitura Complementar
5.3 Função Total
5.4 Construções Matemáticas como Funções
5.5 Função de Hashing
5.6 Funções nas Linguagens de Programação
5.7 Linguagem de Programação Funcional

5.2.4 Leitura Complementar

◆ Autômatos finitos: memória finita e predefinida

- limitações sérias para solucionar problemas
- linguagens regulares: reconhecidas por autômatos finitos
- hierarquia de linguagens: classe dos problemas mais simples
  - Exemplo: parênteses balanceados – não existe autômato finito





◆ Complexidade de algoritmos

- classe de algoritmos *mais* eficientes (tempo de processamento)
- qq autômato que solucione é igualmente eficiente
- qq solução é *ótima*

◆ Implementação computacional de aut. finitos: trivial

5 – Funções Parciais e Totais

5.1 Função Parcial
5.2 Autômato Finito
5.3 Função Total
- 5.3.1 Definição e Introdução
- 5.3.2 Exemplos Imortantes de Funções
- 5.3.3 Função Dual
- 5.3.4 Composição de Funções

5.4 Construções Matemáticas como Funções
5.5 Função de Hashing
5.6 Funções nas Linguagens de Programação
5.7 Linguagem de Programação Funcional

5.3 Função Total

5.3.1 Definição e Introdução

- **Função (total)**
  - função parcial a qual é *total*
  - herda conceitos e terminologias das relações e funções parciais

Def: Função, Aplicação

Aplicação, Função Total ou simplesmente Função

- função parcial $f: A \to B$ a qual é total

- **Portanto, uma função (total)**
  - função parcial definida para *todos* os elementos do domínio

Exp: Função

$A = \{ a \}$, $B = \{ a, b \}$ e $C = \{ 0, 1, 2 \}$

- $=: A \to B$ ✔
- $id_B: B \to B$ ✔
- $x^2: \mathbf{Z} \to \mathbf{Z}$ onde $x^2 = \{ \langle x, y \rangle \in \mathbf{Z}^2 \mid y = x^2 \}$ ✔
- $ad: \mathbf{N} \times \mathbf{N} \to \mathbf{N}$ tal que $ad(a, b) = a + b$ ✔
- $\varnothing: \varnothing \to \varnothing$ ✔

- $\varnothing: A \to B$ ✘
- $\{ \langle 0, a \rangle, \langle 1, b \rangle \}: C \to B$ ✘
- $A \times B: A \to B$ ✘
- $<: C \to C$ ✘
- $div: \mathbf{R} \times \mathbf{R} \to \mathbf{R}$ tal que $div(x, y) = x/y$ ✘

Por que uma relação vazia é função e a outra não?

◆ Função como matriz ou grafo (endorrelação)

- matriz: existe exatamente um valor verdadeiro em cada *linha*
- grafo: existe exatamente uma aresta *partindo* de cada nodo

◆ No contexto das funções

- injetora coincide com monomorfismo
  - ∗ monomorfismo = injetora + total
- sobrejetora coincide com epimorfismo
  - ∗ epimorfismo = sobrejetora + funcional
- isomorfismo também é denominado de função bijetora
  - ∗ como isomorfismo = monomorfismo + epimorfismo
  - ∗ então bijetora = injetora + sobrejetora
- como fica no contexto das funções parciais?

Exp: Função Injetora, Sobrejetora e Bijetora.

$A = \{ a \}$, $B = \{ a, b \}$, $C = \{ 0, 1, 2 \}$ e X um conjunto qualquer

- $=: A \to B$ — injetora, não-sobrejetora
- $id_X: X \to B$ — bijetora
- $x^2: \mathbf{Z} \to \mathbf{Z}$ — não-injetora, não-sobrejetora
- $ad: \mathbf{N} \times \mathbf{N} \to \mathbf{N}$ — epimorfismo, não-monomorfismo
- $\varnothing: \varnothing \to \varnothing$ — bijetora
- $\{ \langle 0, 1 \rangle, \langle 1, 2 \rangle, \langle 2, 0 \rangle \}: C \to C$ — isomorfismo
- $R = \{ \langle x, y \rangle \mid y = \operatorname{sen} x \}$ — monomorfismo, não-epimorfismo

5 – Funções Parciais e Totais

5.3.2 Exemplos Importantes de Funções

◆ Importantes exemplos de funções

- Lembre: se Σ é alfabeto, então Σ^* é conj. das palavras sobre Σ
- Exemplo: para $\Sigma = \{ a, b \}$

$$\Sigma^* = \{ \varepsilon, a, b, aa, ab, ba, bb, aaa, \ldots \}$$

◆ Função constante

- qq valor do domínio, resulta no mesmo valor do contra-domínio

Def: Função Constante

A e B conjuntos. Função constante em $b \in B$

$$\text{const}_b: A \to B$$

- para todo $a \in A$, $\text{const}_b(a) = b$

Exp: Função Constante

$A = \{ a \}$ e $\Sigma = \{ a, b \}$ um alfabeto

- $\text{const}_5: \mathbf{R} \to \mathbf{R}$ $\qquad$ $\text{const}_5 = \{ \langle x, 5 \rangle \in \mathbf{R}^2 \}$
- $\text{palavra\_vazia}: \Sigma^* \to \Sigma^*$ $\qquad$ $\text{palavra\_vazia} = \{ \langle w, \varepsilon \rangle \in \Sigma^* \times \Sigma^* \}$
- $\text{id}_A: A \to A$ $\qquad$ (toda função identidade é uma função constante?)
- $\varnothing: \varnothing \to \varnothing$

◆ Função Concatenação

- especialmente importante para Computação e Informática
- operação binária, definida sobre uma linguagem
  - ∗ associa a cada par de palavras uma palavra
  - ∗ formada pela justaposição da primeira com a segunda

Def: Função Concatenação

$\Sigma = \{ a, b \}$ um alfabeto

$$\text{conc}: \Sigma^* \times \Sigma^* \to \Sigma^*$$

- para todo $\langle u, v \rangle \in \Sigma^* \times \Sigma^*$, $\text{conc}\langle u, v \rangle = u\,v$

Exp: Concatenação

$\Sigma = \{ a, b \}$

- concatenação das palavras aba e bbb de Σ^* resulta em

ababbb palavra de Σ^*.

◆ Importantes funções induzidas por relações ou operações sobre conjuntos

- função inclusão:
  - reflete a continência de conjuntos
  - toda continência induz um função inclusão e vice-versa
- função projeção
  - reflete a relação entre o produto cartesiano e os conjuntos originais
- função imersão
  - reflete a relação a união disjunta e os conjuntos originais
- projeção e imersão: caracterizam a reversabilidade das operações

Def: Função Inclusão

A e B conjuntos tais que $A \subseteq B$

$$inc_{A,B}: A \hookrightarrow B \text{ ou simplesmente } inc: A \hookrightarrow B$$

- para todo $a \in A$, $inc(a) = a$

◆ **Toda função inclusão é injetora**

$$inc_{A,B}: A \rightarrowtail B$$

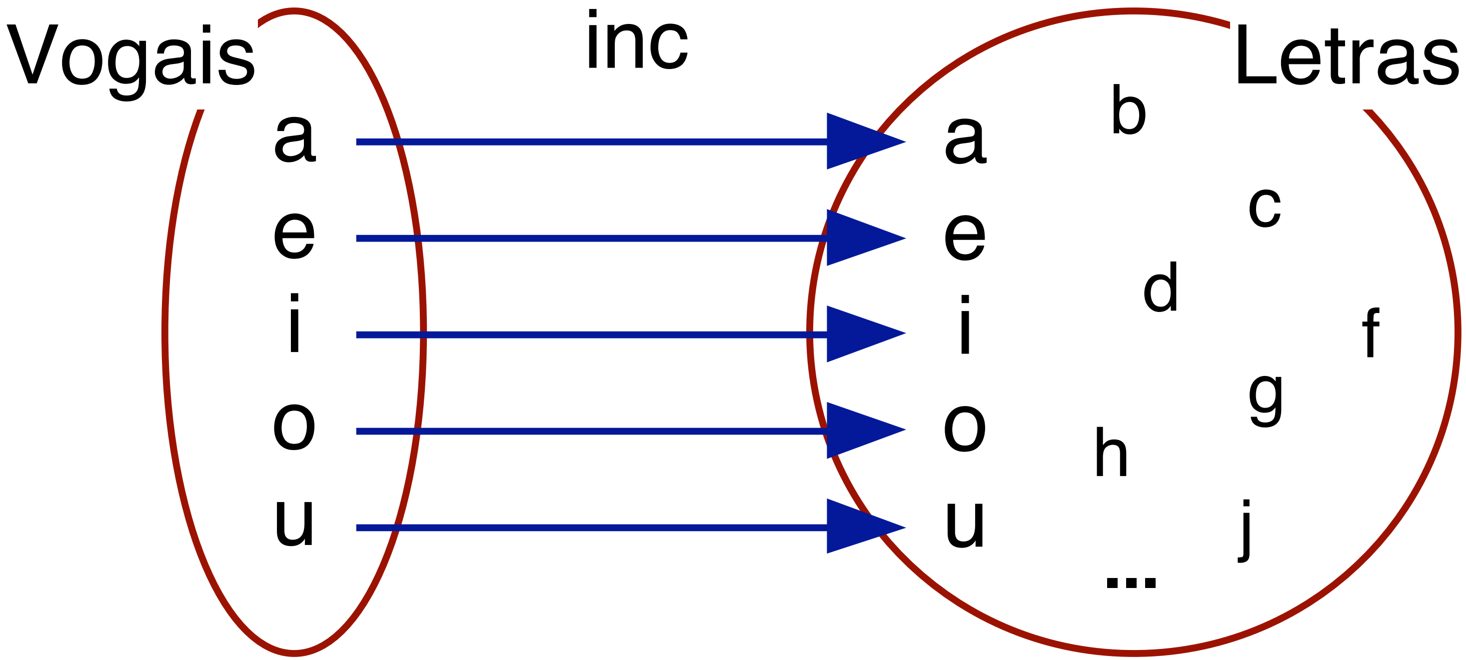
(por quê?)

Exp: Função Inclusão × Continência

Vogais = { a, e, i, o, u } e Letras = { a, b, c,…,z }

- claramente Vogais ⊆ Letras

$$\text{inc}_{\text{Vogais},\text{Letras}}: \text{Vogais} \hookrightarrow \text{Letras}$$



Exp: …Função Inclusão × Continência

As continências $\mathbf{N} \subseteq \mathbf{Z}$, $\mathbf{Z} \subseteq \mathbf{Q}$ e $\mathbf{Q} \subseteq \mathbf{R}$ induzem as funções inclusão

$$\text{inc}_{\mathbf{N},\mathbf{Z}}: \mathbf{N} \rightarrowtail \mathbf{Z} \qquad \text{inc}_{\mathbf{Z},\mathbf{Q}}: \mathbf{Z} \rightarrowtail \mathbf{Q} \qquad \text{inc}_{\mathbf{Q},\mathbf{R}}: \mathbf{Q} \rightarrowtail \mathbf{R}$$

- Pode-se afirmar que existe a função de inclusão $\text{inc}_{\mathbf{N},\mathbf{R}}: \mathbf{N} \rightarrowtail \mathbf{R}$?
- Generalizando: composição de funções inclusão é uma função inclusão?

◆ **Já foi visto: reversabilidade do produto cartesiano**

- como pode ser obtida?

Def: Função Projeção

A e B conjuntos *não*-vazios e $A \times B$ o produto cartesiano

$$\pi_1 : A \times B \rightarrow A \quad \text{e} \quad \pi_2 : A \times B \rightarrow B$$

- para todo $\langle a, b \rangle \in A \times B$

$$\pi_1\langle a, b \rangle = a \quad \text{e} \quad \pi_2\langle a, b \rangle = b$$

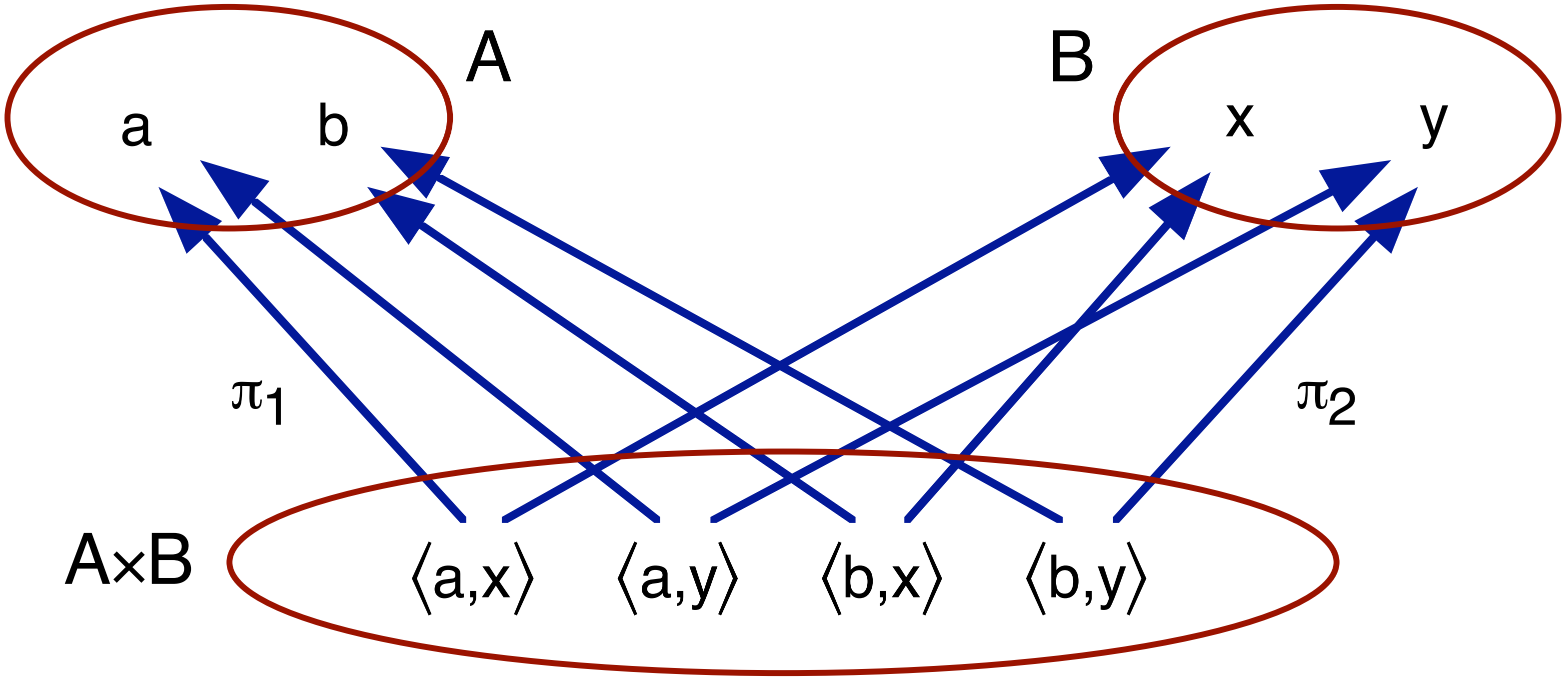
Recuperação dos operandos originais

- $A = \{ \pi_1\langle a, b \rangle \mid \langle a, b \rangle \in A \times B \}$ primeiro operando
- $B = \{ \pi_2\langle a, b \rangle \mid \langle a, b \rangle \in A \times B \}$ segundo operando

Exp: Função Projeção

$A = \{ a, b \}$ e $B = \{ x, y \}$ conjuntos e $A \times B$ o produto cartesiano

- funções projeção $\pi_1: A \times B \to A$ e $\pi_2: A \times B \to B$



Toda função projeção é sobrejetora (por quê?)

Def: Função Imersão

A e B conjuntos e $A + B$ a união disjunta

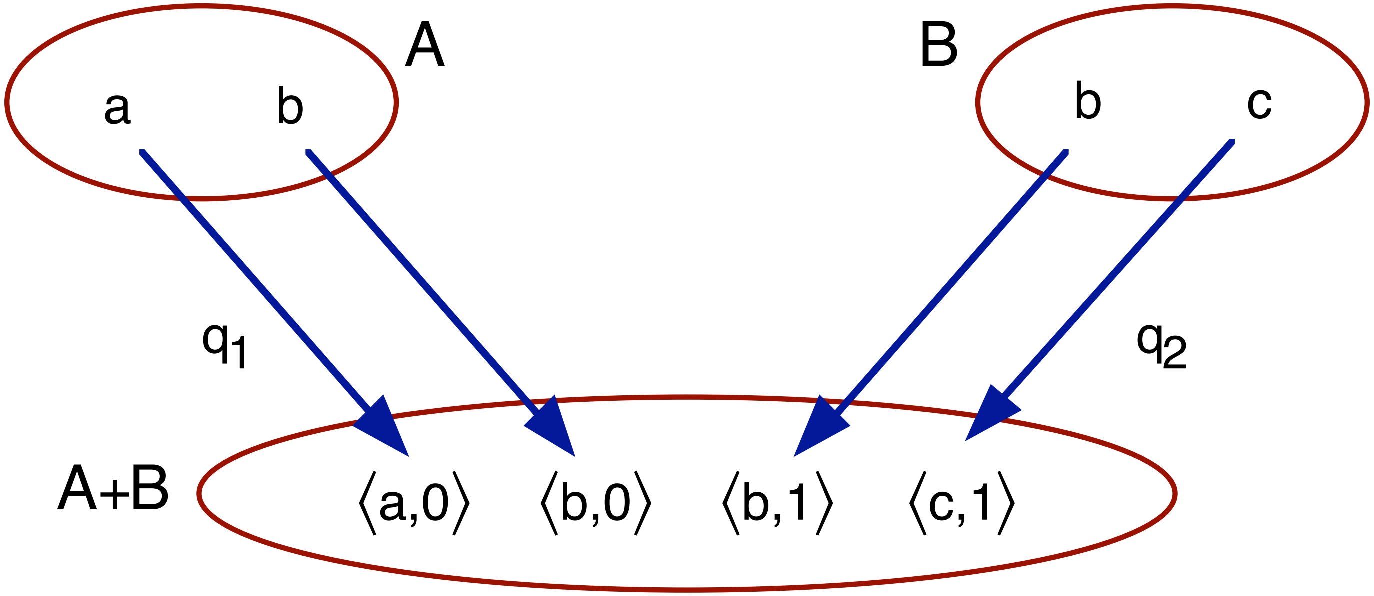
$$q_1 : A \to A + B \quad \text{e} \quad q_2 : B \to A + B$$

- para todo $a \in A$, tem-se que $q_1(a) = \langle a, 0 \rangle$
- para todo $b \in B$, tem-se que $q_2(b) = \langle b, 1 \rangle$

Exp: Função Imersão

$A = \{ a, b \}$ e $B = \{ b, c \}$

- $A + B$ união disjunta
- $q_1 : A \to A + B$ e $q_2 : B \to A + B$ funções de imersão



Toda função imersão é injetora (por quê?).

5 – Funções

5.3.3 Função Dual

◆ Relação dual de uma função

- *não* necessariamente é uma função (por quê?)

Exp: Relação Dual de Função

$f: \{ 0, 1, 2 \} \to \{ 0, 1 \}$ tal que $f = \{ \langle 0, 0 \rangle, \langle 1, 1 \rangle, \langle 2, 0 \rangle \}$

$f^{op} = \{ \langle 0, 0 \rangle, \langle 1, 1 \rangle, \langle 0, 2 \rangle \}$ não-funcional

$g: \{ 0, 1 \} \to \{ 0, 1, 2 \}$ tal que $g = \{ \langle 0, 0 \rangle, \langle 1, 1 \rangle \}$

$g^{op} = \{ \langle 0, 0 \rangle, \langle 1, 1 \rangle \}$ não-total

- conjuntos correspondentes a g e g^{op} são *iguais*
- por que g *é* função e g^{op} *não* é função?

◆ Condições para que dual de função seja função?

- função = total + funcional
- lembre-se o dual de
  - total é sobrejetora
  - funcional é injetora
- conclusão: sobrejetora + injetora, ou seja, bijetora

Exp: Relação Dual de Função

$A = \{ a \}$, $B = \{ a, b \}$ e $C = \{ 0, 1, 2 \}$. Duais são funções?

- $id_B: B \to B$ ✔
- $\varnothing: \varnothing \to \varnothing$ ✔
- $\{ \langle 0, 1 \rangle, \langle 1, 2 \rangle, \langle 2, 0 \rangle \}: C \to C$ ✔

- $=: A \to B$ ✘
- ad: $\mathbf{N} \times \mathbf{N} \to \mathbf{N}$ tal que $ad(a, b) = a + b$ ✘
- $x^2: \mathbf{Z} \to \mathbf{Z}$ onde $x^2 = \{ \langle x, y \rangle \in \mathbf{Z}^2 \mid y = x^2 \}$ ✘
- $R = \{ \langle x, y \rangle \mid y = \text{sen}\, x \}$ ✘

- Em que condições a dual de uma função inclusão é função?

5 – Funções Parciais e Totais

5.3.4 Composição de Funções

- **Composição de funções parciais é função parcial**
  - já foi visto
- **Composição de funções é função?**
  - basta provar que a composições de relações totais *é* total

Teorema: Composição de Totais é Total

$R: A \rightarrow B$ e $S: B \rightarrow C$ relações totais

Então $S \circ R: A \rightarrow C$ é total

Prova: *Composição de Totais é Total*

Suponha $R: A \to B$ e $S: B \to C$ relações totais

Então $S \circ R: A \to C$ é relação

Basta provar que composição de totais é total

$$(\forall a \in A)(\exists c \in C)(a\,(S \circ R)\,c)$$

De fato, suponha $a \in A$. Então:

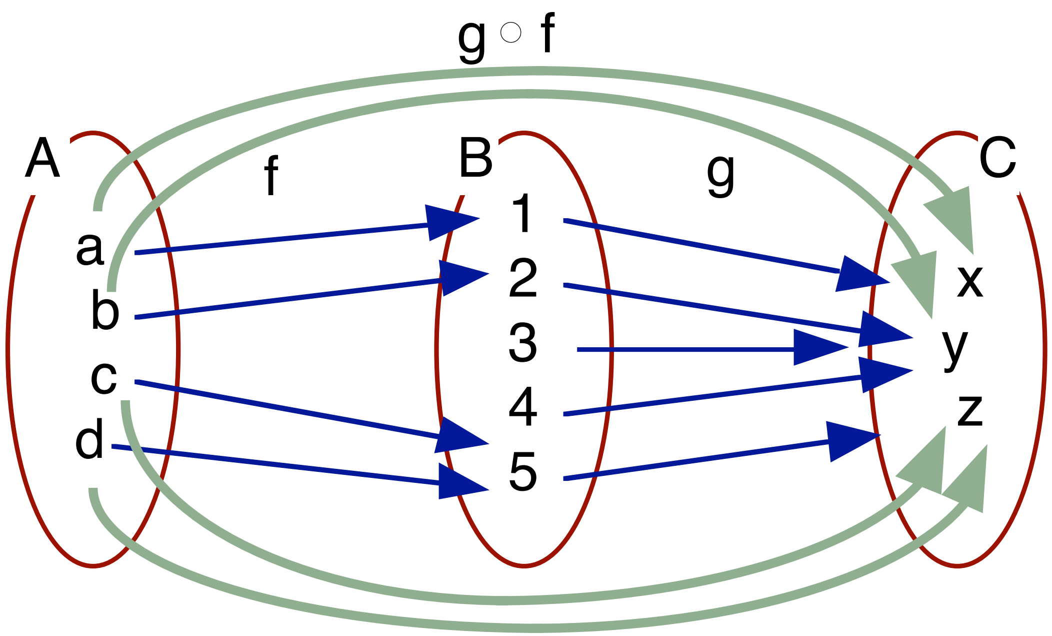
- $a \in A \Rightarrow$ — R é total
- $(\exists b \in B)(a\,R\,b) \Rightarrow$ — S é total
- $(\exists b \in B)(\exists c \in C)(a\,R\,b \land b\,S\,c) \Rightarrow$ — definição de composição
- $(\exists c \in C)(a\,(S \circ R)\,c)$

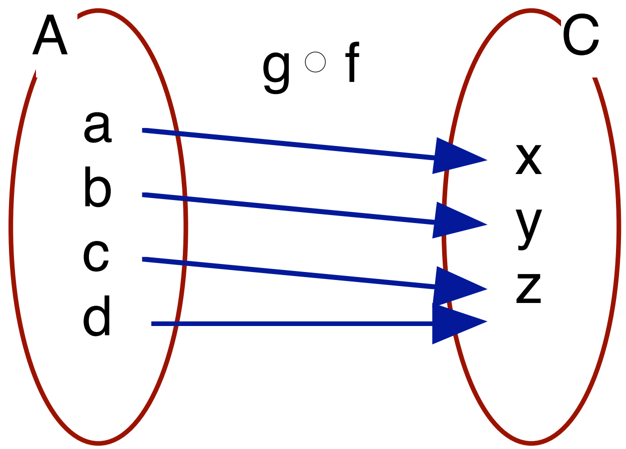
Logo, $S \circ R: A \to C$ é uma relação total

Exp: Composição de Funções

$f: A \to B$, $g: B \to C$ e $g \circ f: A \to C$

- $f = \{ \langle a, 1 \rangle, \langle b, 2 \rangle, \langle c, 5 \rangle, \langle d, 5 \rangle \}$
- $g = \{ \langle 1, x \rangle, \langle 2, y \rangle, \langle 3, y \rangle, \langle 4, y \rangle, \langle 5, z \rangle \}$
- $g \circ f = \{ \langle a, x \rangle, \langle b, y \rangle, \langle c, z \rangle, \langle d, z \rangle \}$





◆ Por dualidade do Teorema da Composição de Totais

- composição de relações sobrejetoras é relação sobrejetora
- exercício: prova do corolário, "dualizando" a prova do teorema

Corolário: Composição de Sobrejetoras é Sobrejetora

$R: A \to B$ e $S: B \to C$ relações sobrejetoras

Então $S \circ R: A \to C$ é sobrejetora

5 – Funções Parciais e Totais

5.4 Construções Matemáticas como Funções

◆ **Funções são freqüentemente usadas para**

- definir outras construções matemáticas
- exemplos
  - * seqüência
  - * multiconjunto
  - * conjunto indexado
  - * relação

5 – Funções Parciais e Totais

5.1 Função Parcial
5.2 Autômato Finito
5.3 Função Total
5.4 Construções Matemáticas como Funções
 5.4.1 Relação como Função
 5.4.2 Multiconjunto
 5.4.3 Seqüência
 5.4.3 Conjunto Indexado
5.5 Função de Hashing
5.6 Funções nas Linguagens de Programação
5.7 Linguagem de Programação Funcional

5.4.1 Relação como Função

- **Relação** $R: A \to B$

$$R \subseteq A \times B$$

- **Como todo subconjunto define uma função inclusão**
  - Qq relação pode ser vista como

$$inc_{R,A\times B}: R \rightarrowtail A \times B$$

  - definição alternativa para relação
  - na Matemática e na Computação e Informática
    - usual definições alternativas equivalentes
    - para uma mesma construção

5 – Funções Parciais e Totais

5.4.2 Multiconjunto

- **Informalmente, um conjunto é**

  uma coleção, sem repetições e sem qualquer ordenação, de objetos denominados elementos

- **Formalmente**

  uma coleção de zero ou mais objetos distintos, chamados elementos do conjunto os quais não possuem qualquer ordem associada

- **Característica fundamental**

  - elementos distintos, *não* podem ser repetidos

- **Pela definição de igualdade de conjuntos**

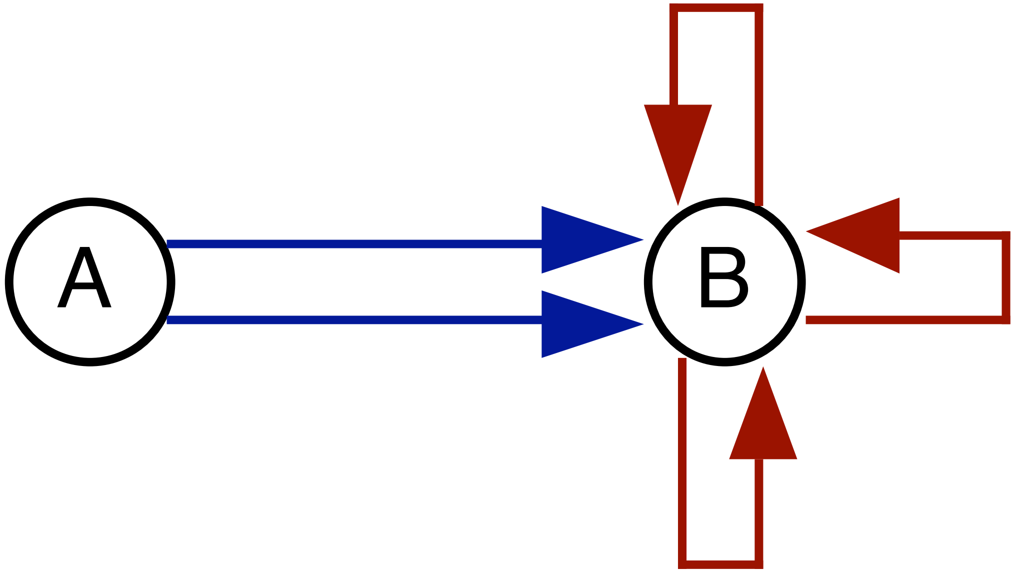
$$\{ 1, 2, 3 \} = \{ 3, 3, 3, 2, 2, 1 \}$$

- **Multiconjuntos**
  - em alguns momentos é necessário tratar conjuntos com repetições
  - exemplo
    - união disjunta
    - grafos

- **Na união disjunta, foi apresentada uma solução**
  - garante uma identidade única de cada elemento
  - noção de "sobre-nome"
  - *vantagem*: reversabilidade da operação
  - multiconjunto: solução alternativa (*não* permite a reversabilidade)

◆ Grafo

- toda endorrelação pode ser vista como um grafo
- *nem* todo grafo é uma relação
  - ∗ um motivo: grafos podem possuir arcos paralelos



- quais arcos são paralelos?

Def: Multiconjunto

X conjunto.

Multiconjunto A de objetos de X é uma função

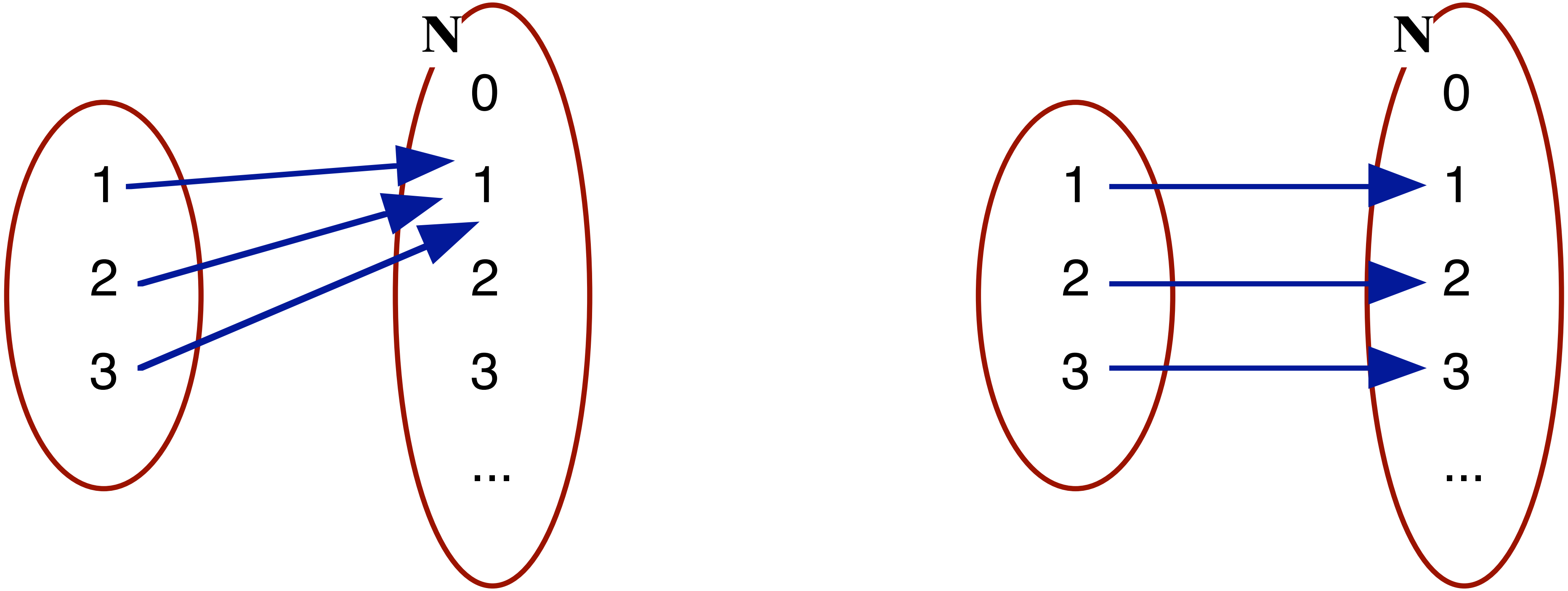
$$A: X \to \mathbf{N}$$

◆ Notação

- como um conjunto, explicitando as repetições
- destacando que trata-se de um *multiconjunto*

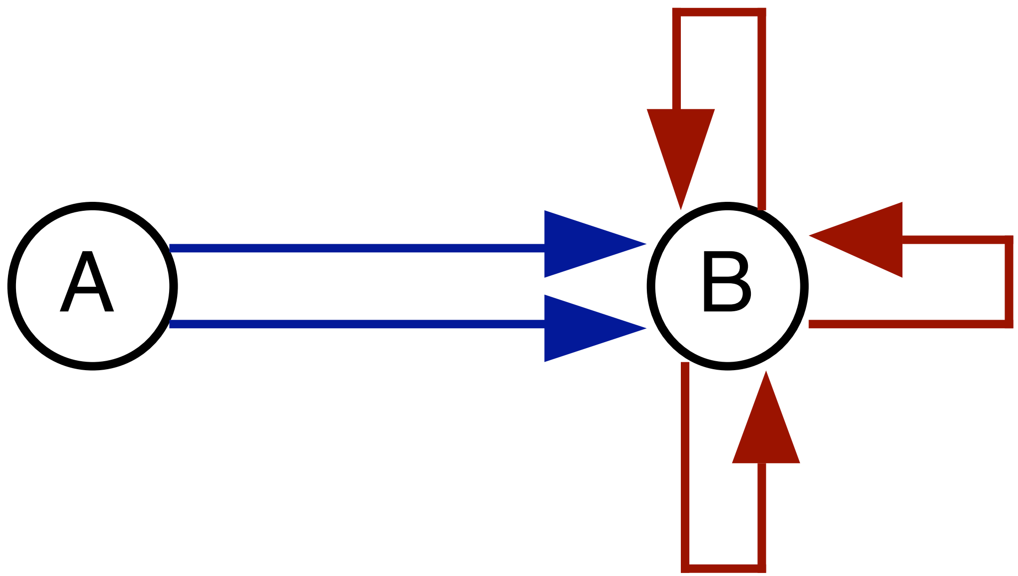
Exp: Multiconjunto

$$\{ 1, 2, 3 \} \neq \{ 3, 3, 3, 2, 2, 1 \}$$



Qual a interpretação quando o número de repetições é zero?

Exp: Grafo com Arcos Paralelos



$$G = \{ \langle A, B \rangle, \langle A, B \rangle, \langle B, B \rangle, \langle B, B \rangle, \langle B, B \rangle \}$$

É uma relação?

Qual a função que define o multiconjunto?

5 – Funções Parciais e Totais

5.4.3 Seqüência

- **Seqüência**
  - termo sendo usado intuitivamente
- **Quando da definição de produto cartesiano**
  - noção mais formal de seqüência (finita) ou de n-upla ordenada

uma seqüência de n componentes, denominada de n-upla ordenada consiste de n objetos (não necessariamente distintos) em uma ordem fixa

$$\langle x_1, x_2, x_3, \ldots, x_n \rangle \neq \{ x_1, x_2, x_3, \ldots, x_n \}$$

Def: Seqüência Infinita, Seqüência Finita

X conjunto

Seqüência Infinita de X é uma função

$$\underline{x}: \mathbf{N} - \{ 0 \} \to X$$

Seqüência Finita ou n-Upla Ordenada com n componentes de objetos de X é uma função

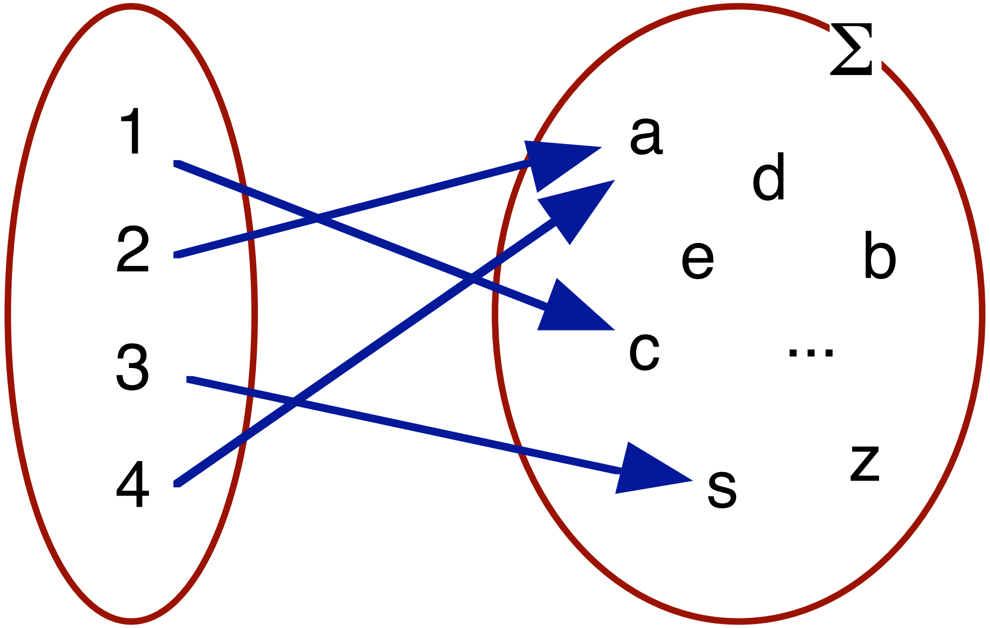
$$\underline{x}: \{ 1, 2, 3,\ldots,n \} \to X$$

Exp: Palavra como Função

$\Sigma = \{ a, b, c, \ldots, z \}$ alfabeto

Palavra casa como função (não-injetora)

$$\text{casa}: \{ 1, 2, 3, 4 \} \to \Sigma$$



- $\text{casa} = \{ \langle 1, c \rangle, \langle 2, a \rangle, \langle 3, s \rangle, \langle 4, a \rangle \}$

5 – Funções Parciais e Totais

5.4.4 Conjunto Indexado

uma variável do tipo arranjo é uma seqüência finita de variáveis, todas do mesmo tipo

Exemplo

```
dados = array[1..10] of char
```

Correspondente função

$$\text{dados}: \{ 1, 2, 3,\ldots,10 \} \to X$$

- X é um conjunto de variáveis do tipo **`char`**
- função dados é
  - ∗ Injetora?
  - ∗ Sobrejetora?

$$\text{dados}: \{1, 2, 3, \ldots, 10\} \to X$$

Injetora: Cada componente de um arranjo é distinta

$$\text{dados}: \{1, 2, 3, \ldots, 10\} \rightarrowtail X$$

- no exemplo
  - $\ast$ dados(1) e dados(8) são variáveis do tipo **`char`** *distintas*
  - $\ast$ correspondem às componentes **`dados[1]`** e **`dados[8]`**

Sobrejetora. Cada componente do arranjo é indexável por algum índice

$$\text{dados}: \{1, 2, 3, \ldots, 10\} \twoheadrightarrow X$$

Logo, a função é bijetora

$$\text{dados}: \{1, 2, 3, \ldots, 10\} \leftrightarrow X$$

Generalização do raciocínio, define conjunto indexado

Def: Conjunto Indexado

I e X conjuntos. Então, para uma função bijetora

$$f: I \leftrightarrow X$$

X é um Conjunto Indexado pelo conjunto I

- **Portanto**
  - qualquer função bijetora define um conjunto indexado
- **Para $f: I \leftrightarrow X$**
  - I - conjunto de índices
  - $x \in X$ é genericamente denotado usando o seu índice $i \in I$

$$f(i) \quad \text{é denotado por} \quad x_i$$

5 – Funções Parciais e Totais

5.5 Função de Hashing

- **Armazenamento e recuperação informações**
  - Eficiente: espaço de armazenamento e tempo de recuperação
- **Armazenamento e recuperação pode ser em**
  - tabela: variável do tipo arranjo
  - arquivo de acesso direto: arquivo
    - ∗ cada entrada/registro acessável diretamente
  - suponha que se trata de tabela

◆ Solução simples e eficiente

- chave de identificação (exemplo, número de matrícula de alunos)
  - índice da tabela

- se os valores para chave >> número provável de entradas?
  - exemplo: cadastro de clientes de loja sendo CIC a chave
  - espaço de armazenamento excessivamente grande e esparço

◆ Para uma tabela com poucas entradas

- usando uma chave relativamente grande
- como endereçar a correspondente entrada na tabela ??

$$f\colon \text{Chaves} \to \{\, 1,\ 2,\ 3, \ldots, n \,\}$$

- **Função para obter o endereço de instalação**
  - função de cálculo de endereço
  - função de aleatorização
  - função de randomização
  - função de hashing
- **Função ideal**
  - *Injetora* (por quê?)
- **No entanto, é difícil conseguir um monomorfismo**
  - funções de *hashing* geralmente geram colisões
    - mesmo endereço a chaves *diferentes*

$$c_1 \neq c_2 \wedge f(c_1) = f(c_2)$$ colisão

Exp: Função de *Hashing*

Chave: entre 0 e 1000

Tabela: entradas indexadas de 1 a 23

Função de hashing relativamente simples e razoavelmente eficiente

$$f: \{ 0, 1, \ldots, 1000 \} \rightarrow \{ 1, 2, \ldots, 23 \}$$

$$f(c) = (c \bmod 23) + 1$$

Exemplo de cálculos e colisões:

Chave	452	623	766	564	825	387	237	360	134	285
Endereço	16	3	8	13	21	20	8	16	20	10

◆ Como objetr uma função de hashing injetora?

- métodos de tratamento de colisões
- política para a escolha de uma entrada disponível
- estudo das Estruturas de Dados

5 – Funções Parciais e Totais

5.6 Funções nas Linguagens de Programação

- **Maioria das linguagens de programação**
  - manipula construções similares ou baseadas nas funções matemáticas
  - Pascal: declaração **`function`**
    - introduzida via exemplos
    - permite implementar algumas funções matemáticas
    - algumas diferenças: proxima secção

Exp: Função em Pascal: *Hashing*

$$f(c) = (c \bmod 23) + 1$$

Declaração de dois tipos intervalos

```
type  interv_0_1000 = 0..1000
      interv_1_23 = 1..23
```

Declaração da função *hashing* f: $\{ 0,1,\ldots,1000 \} \rightarrow \{ 1,2,\ldots,23 \}$

```
function hash(c: interv_0_1000): interv_1_23;
  begin
  hash := (c mod 23) + 1
  end
```

```
function hash(c: interv_0_1000): interv_1_23;
```

- domínio da função
  - **`c`** do tipo **`interv_0_1000`**
  - parâmetro formal

- contra-domínio da função
  - **`hash`**, do tipo **`interv_1_23`**
  - contem valor resultante do cálculo da chamada da função

```
if hash(766) = hash(237) then …
```

- exemplo de chamada da função
  - valores **`766`** e **`237`**: parâmetros atuais
  - comando após a palavra **`then`** é executado?

Exp: Função em Pascal: **EXOR**

Ou-Exclusivo denominado de EXOR (do inglês, exclusive or)

- usual em Computação e Informática

p	q	p EXOR q
V	V	F
V	F	V
F	V	V
F	F	F

EXOR pode ser reescrito, usando os conetivos usuais

$$p \text{ EXOR } q \Leftrightarrow (p \land \neg q) \lor (\neg p \land q)$$

Função em Pascal que implementa o conetivo EXOR

```
function exor(p, q: boolean): boolean;
   begin
   exor := (p and not q) or (not p and q)
   end
```

◆ Um problema das funções em Pascal

- *não* permitem contra-domínio resultante de produto cartesiano
- exemplo: equação polinomial do segundo grau

$$a x^2 + b x + c = 0$$

- fórmula de Baskara: *duas* raízes
  - ⁎ função do tipo

$$\text{baskara: } \mathbf{R}^3 \rightarrow \mathbf{R}^2$$

◆ Solução mais adequada

linguagem de programação funcional

5 – Funções Parciais e Totais

5.7 Linguagem de Programação Funcional

◆ Programação funcional

- estilo de programação baseada em
  - ∗ funções
  - ∗ composição de funções (constituindo um programa)

◆ Programa: expressão funcional

- é avaliada
- em vez de comandos que são executados

◆ Linguagem de programação funcional

- linguagem que suporta e encoraja este estilo programação

- **Linguagem de programação funcional "pura"**
  - *não* possui variáveis, *nem* atribuições.
- **Linguagem de programação funcional é composta por**
  - tipos primitivos de dados
  - constantes de cada tipo primitivo
  - operações: funções sobre os tipos primitivos
  - construtores que permitem definir novos tipos e operações

5 – Funções Parciais e Totais

5.1 Função Parcial
5.2 Autômato Finito
5.3 Função Total
5.4 Construções Matemáticas como Funções
5.5 Função de Hashing
5.6 Funções nas Linguagens de Programação
5.7 Linguagem de Programação Funcional
    5.7.1 Haskell
    5.7.2 Leitura Complementar

5.7.1 Haskell

◆ Haskell

- linguagem de programação funcional pura

- usa o conceito matemático de função
  - ∗ mesmo valores do domínio (parâmetros atuais)
  - ∗ resultam nos mesmos valores do codomínio (mesmas saídas)

- portanto
  - ∗ resultado da aplicação de uma função
  - ∗ independente de qualquer estado implícito do sistema.

◆ Exemplo em Haskell

```
x = 1
```

- função constante: sempre retorna o valor `1`

```
f x = x + 1
```

- função **`f`**: para o parâmetro **`x`**, retorna o valor **`x + 1`**

◆ Contra-exemplo em Pascal (`x` variável `integer`)

```
function contador: integer;
   begin
   x := x + 1;
   contador := x
   end
```

◆ Exemplo em Haskell: $ax^2 + bx + c = 0$

```
baskara a b c =
let delta = b*b - 4*a*c
in ( (-b + sqrt(delta))/(2*a),
   (-b - sqrt(delta))/(2*a) )
```

- para **`a`**, **`b`** e **`c`**, retorna o par ordenado correspondendo as raízes
- **`let`** é usada para declarar **`delta`**
  - ⁕ acessível apenas no escopo da função **`baskara`**

◆ Redução (avaliação) para os valores 1, -5 e 6

```
baskara 1 -5 6
```

```
let delta = (-5)*(-5) - 4*1*6
in ( (5 + sqrt(delta))/(2*1),
     (5 - sqrt(delta))/(2*1) )
```

```
let delta = 1
in ( (5 + sqrt(delta))/2,
     (5 - sqrt(delta))/2 )
```

```
( (5 + 1)/2,
(5 - 1)/2 )
```

```
( 3, 2 )
```

5 – Funções Parciais e Totais

5.1 Função Parcial
5.2 Autômato Finito
5.3 Função Total
5.4 Construções Matemáticas como Funções
5.5 Função de Hashing
5.6 Funções nas Linguagens de Programação
5.7 Linguagem de Programação Funcional
    5.7.1 Haskell
    5.7.2 Leitura Complementar

Matemática Discreta para Ciência da Computação

P. Blauth Menezes

blauth@inf.ufrgs.br

Departamento de Informática Teórica

Instituto de Informática / UFRGS