# Indução

Método de Prova por Indução

Referência: Language, Proof and Logic
Jon Barwise e John Etchemendy, 2008
Capítulo: 16

# Indução

- Métodos de prova já vistos
  - relacionam-se diretamente com as propriedades das conetivas e quantificadores
- Exceções
  - Prova por contradição: usa-se para qualquer tipo de fórmula
  - Provas para afirmações numéricas
- Provar afirmações da forma

$$\forall \mathbf{x}\ [\mathbf{P(x)} \rightarrow \mathbf{Q(x)}]$$

  - Prova condicional geral já usada para este efeito
  - Indução necessária quando P(x) tem definição indutiva

# Métodos indutivos e indução matemática

- No raciocínio científico
  - indução usada para retirar uma conclusão geral a partir de um número finito de observações
    - em termos lógicos: inferência não é justificada
    - novas observações podem invalidar a conclusão
- Indução matemática
  - conclusão geral, válida para um número **infinito** de instâncias, é justificada com uma prova **finita**
- Aplicação mais usual
  - domínio dos inteiros
  - indução aplicável porque a definição dos inteiros é naturalmente indutiva
  - uso não restrito a este domínio

# Imagem para a indução

- Cadeia de dominós
  - quando se derruba o primeiro: todos caem

- Arranjo dos dominós -- definição indutiva
- Fazer cair todos -- provar teorema por indução
- Requisitos para que os dominós caiam todos:
  - posições tais que quando um cai faz cair o seguinte --passo indutivo
  - o primeiro cai -- passo de base
- Número de peças que uma peça faz cair: sem restrições
  - podem montar-se esquemas complexos

# Definições indutivas

- Exemplos anteriores
  - definição de wff
  - definição de termos aritméticos
- Esquema geral
  - dizer como são os elementos “simples”
  - dizer como gerar novos elementos partindo dos que já se têm
- Exemplo: definir *ambig-wff*

| | |
|---|---|
| $A_1, A_2, \ldots, A_n$ | símbolos proposicionais |
| $\neg \ \wedge \ \vee \ \rightarrow \ \leftrightarrow$ | conetivas |

(1) Cada símbolo proposicional é uma ambig-wff

(2) Se p é ambig-wff, também $\neg$p é ambig-wff

(3) Se p e q são ambig-wff, $p\wedge q$, $p\vee q$, $p\rightarrow q$, $p\leftrightarrow q$ também o são

(4) As únicas ambig-wff são as geradas por aplicação repetida de (1), (2) e (3)

# Definição indutiva

<u>Cláusula de base</u>

especifica os elementos básicos do conjunto a definir

<u>Uma ou mais cláusulas indutivas</u>

descrevem a forma de gerar novos elementos

<u>Cláusula final</u>

estabelece que os elementos ou são básicos ou gerados pelas cláusulas indutivas

# Verificação de definições indutivas

- $A1 \vee A2 \wedge \neg A3$ é ambig-wff
- Prova:

  A1, A2 e A3 são ambig-wff pela cláusula (1)

  $\neg A3$ é ambig-wff pela cláusula (2)

  $A2 \wedge \neg A3$ é ambig-wff pela cláusula (3)

  $A1 \vee A2 \wedge \neg A3$ é ambig-wff pela cláusula (3)

- Porque se chamará esta linguagem ambig-wff?

# Inferência sobre definição indutiva

- **Proposição 1**: Toda a ambig-wff tem pelo menos 1 símbolo proposicional
- Prova:
  - Base:
    - cada símbolo proposicional contém 1 símbolo proposicional
  - Indução:
    - p e q são ambig-wff que contêm pelo menos 1 símbolo proposicional
    - as ambig-wff geradas por (2) e (3) a partir destas também têm pelo menos 1 símbolo proposicional:
      - $\neg p$ tem os símbolos proposicionais de p
      - $p \wedge q$, $p \vee q$, $p \rightarrow q$, $p \leftrightarrow q$ têm os símbolos proposicionais de p e de q
  - Cláusula (4) justifica a conclusão: nada é ambig-wff exceto os elementos base e as fórmulas geradas a partir deles aplicando (2) e (3)

# Princípio da indução matemática

- Forma da afirmação: condicional geral
- Antecedente: definido indutivamente (Dom)

$$\forall p \, [(p \in \mathrm{Dom}) \rightarrow Q(p)]$$

- Forma da prova
  - **Passo base**: mostrar que os elementos base satisfazem Q
  - **Passo indutivo**: admitindo que alguns elementos satisfazem Q mostrar que os elementos que são gerados a partir deles pelas cláusulas indutivas também satisfazem Q

**Hipótese indutiva**

  - **Conclusão**: todos os elementos do domínio satisfazem Q

# Indução na complexidade da fórmula

- S – conjunto de fórmulas ambig-wff, construído a partir de definição indutiva (admitindo que só há 2 símbolos proposicionais A e B)
- $\boxed{S(0) = \{A, B\}}$ **caso base**
- $S(1) = S(0) \cup \{\neg A, \neg B, A \wedge A, A \wedge B, B \wedge A, B \wedge B, A \vee A, A \vee B, B \vee A, B \vee B, A \rightarrow A, A \rightarrow B, B \rightarrow A, B \rightarrow B, A \leftrightarrow A, A \leftrightarrow B, B \leftrightarrow A, B \leftrightarrow B\}$
- $S(2) = S(1) \cup \{\neg\neg A, \neg\neg B, \neg A \wedge A, \neg A \wedge B, \ldots, A \wedge \neg A, A \wedge \neg B, A \wedge A \wedge A, \ldots,$
  $\neg A \wedge A, \neg A \wedge B, \neg A \wedge \neg A, \neg A \wedge \neg B, \neg A \wedge A \wedge A, \ldots\}$
- …

**passo indutivo**

- S(n) = {…} (fórmulas com n níveis de operadores)
- S(n+1) = {…} (fórmulas com n+1 níveis de operadores)
- …

# Árvore de análise

- $A1 \lor A2 \land \neg A3$
  - Duas árvores de análise possíveis
  - Nível 3 e nível 2


Árvore I

Árvore II

# Uso de indução

- **Proposição 2**:

  Nenhuma ambig-wff tem o símbolo $\neg$ imediatamente antes de uma das conetivas $\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$

  $\forall$p [(p é ambig-wff) $\rightarrow$ Q(p)]

  Q: não ter $\neg$ imediatamente antes de uma conetiva binária

- Prova:
  - Passo base: Q(p) verifica-se para as ambig-wff dadas por (1)
  - Passo indutivo:
    - Caso 1: por (2), se p tem propriedade Q, também $\neg$p
    - Caso 2: por (3) se p tem propriedade Q, também p$\wedge$q, p$\vee$q, p$\rightarrow$q, p$\leftrightarrow$q

- Problema: nenhum dos casos se pode provar

# Paradoxo do inventor

- Caso 1: não se pode provar
  - $\rightarrow$A1 verifica Q e $\neg\rightarrow$A1 não verifica
- Caso 2: não se pode provar
  - A1$\neg$ e A2 verificam Q e A1$\neg\vee$A2 não verifica
- Caso em que uma prova indutiva encrava
  - Afirmação a provar é verdadeira
  - Para prová-la tem de se provar algo **mais forte**
- Nova condição na Proposição 2:
  - Q': não começar por conetiva binária, não terminar em $\neg$ nem ter $\neg$ imediatamente antes de uma conetiva binária
  - Caso 1: óbvio
  - Caso2: por considerações acerca das propriedades de p e q

# Cláusula final da definição indutiva

- Qual o estatuto da cláusula

  *(4) Nada é ambig-wff a menos que seja gerado por aplicações sucessivas de (1), (2) e (3)*

  - Refere, para além de objetos que estão a ser definidos, as outras cláusulas da definição
  - usa noção de “aplicação repetida”

- Expressão em LPO:
  - Direta para as cláusulas (1), (2) e (3)
    - ...
    - (2) $\forall p$ [ambig-wff(p) $\rightarrow$ ambig-wff(concat(‘$\neg$‘, p))]
    - ...
  - Não existe tradução deste tipo para (4)

# Definições indutivas em Teoria de Conjuntos

- Definições indutivas: podem exprimir-se na linguagem da Teoria de Conjuntos
- Ambig-wff
  - O conjunto S das ambig-wff é o menor conjunto que verifica
  - (1) Cada símbolo de proposição está em S
  - (2) Se p está em S, $\neg$p está em S
  - (3) Se p e q estão em S, p$\wedge$q, p$\vee$q, p$\rightarrow$q, p$\leftrightarrow$q também estão
- (4) foi substituída pela referência a "o menor conjunto que satisfaz (1), (2) e (3)"

# Provas

- Para provar que todas as ambig-wff estão em Q
  - conjunto S das ambig-wff é subconjunto de Q $S \subseteq Q$
  - Se Q satisfaz (1) - (3)
  - $S \subseteq Q$ pela definição
- Problema na prova da Proposição 2
  - Q não satisfaz (2) ou (3)
  - Q’ é conjunto mais restrito, verifica (1) - (3)
  - $S \subseteq Q' \subseteq Q$
  - logo $S \subseteq Q$ : resultado pretendido
  - O paradoxo do inventor significa ter que inventar uma condição mais forte para provar, que é satisfeita por menos elementos, mas que permite avançar no raciocínio

# Indução sobre os naturais

- Definição indutiva dos números naturais
  1. 0 é um número natural
  2. Se n é natural, n+1 é natural
  3. Nada é um natural excepto os resultados da aplicação repetida de (1) e (2)
- Em teoria de conjuntos

  $\mathbb{N}$, o conjunto dos naturais, é o conjunto mais pequeno que satisfaz

  (1) $0 \in \mathbb{N}$

  (2) Se $n \in \mathbb{N}$, $n+1 \in \mathbb{N}$
- Prova indutiva sobre $\mathbb{N}$

  $$\forall x\ [(x \in \mathbb{N} \rightarrow x \in Q]$$

De:

(1) $0 \in Q$

(2) Se $n \in Q$ então $n+1 \in Q$

pode concluir-se $\mathbb{N} \subseteq Q$

# Exemplo: soma de n naturais

- Para todo o número natural n, a soma dos n primeiros naturais é n(n+1)/2
- Prova:

**ordem da indução**

$\boxed{\forall n\ [n \in \mathbb{N} \rightarrow n \in Q]}$

**afirmação a provar**

$\boxed{\text{Q(n): a soma dos n primeiros naturais é n(n+1)/2}}$

**hipótese**

Caso base: a soma dos 0 primeiros naturais é 0

Passo indutivo: $\boxed{\text{Seja um número natural k para o qual Q(k) se verifica}}$

Soma dos k primeiros naturais é k(k+1)/2

Soma dos primeiros k+1 naturais:

$\boxed{1+2+\ldots+k} + k+1 =$

**usar a hipótese**

$k(k+1)/2 + k+1 = (k+1)(k/2 + 1) = (k+1)(k+2)/2$

$\boxed{\text{portanto Q(k+1) também se verifica.}}$ **conclusão**

# Exemplo: fatorial

- Definição de fatorial
  - $n! = \begin{cases} 1 & se\ n = 0 \\ n(n-1)(n-2)\ldots 2.1 & se\ n \geq 1 \end{cases}$

- Exemplos
  - 0! = 1     1! = 1     3! = 3.2.1 = 6     6! = 6.5.4.3.2.1 = 720

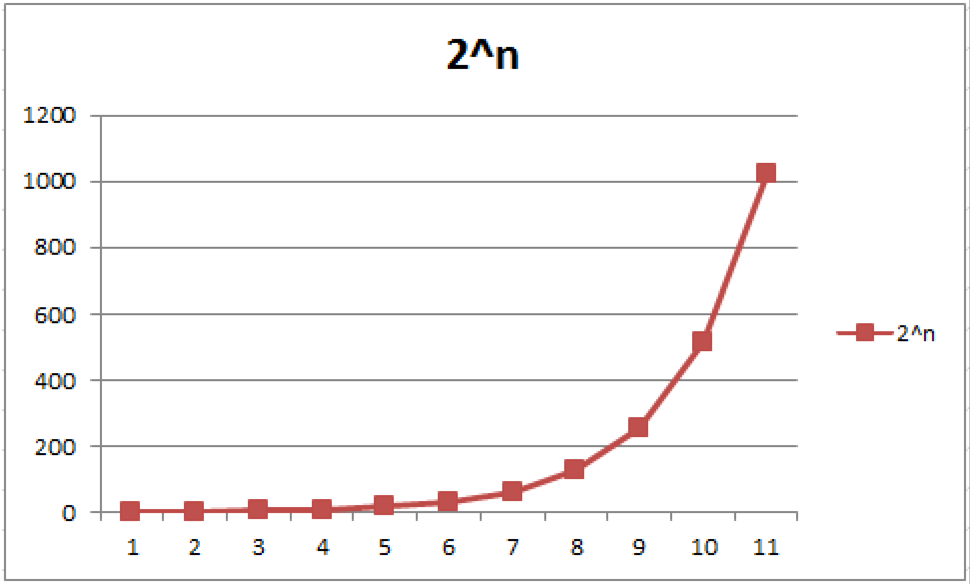
# Propriedade do fatorial

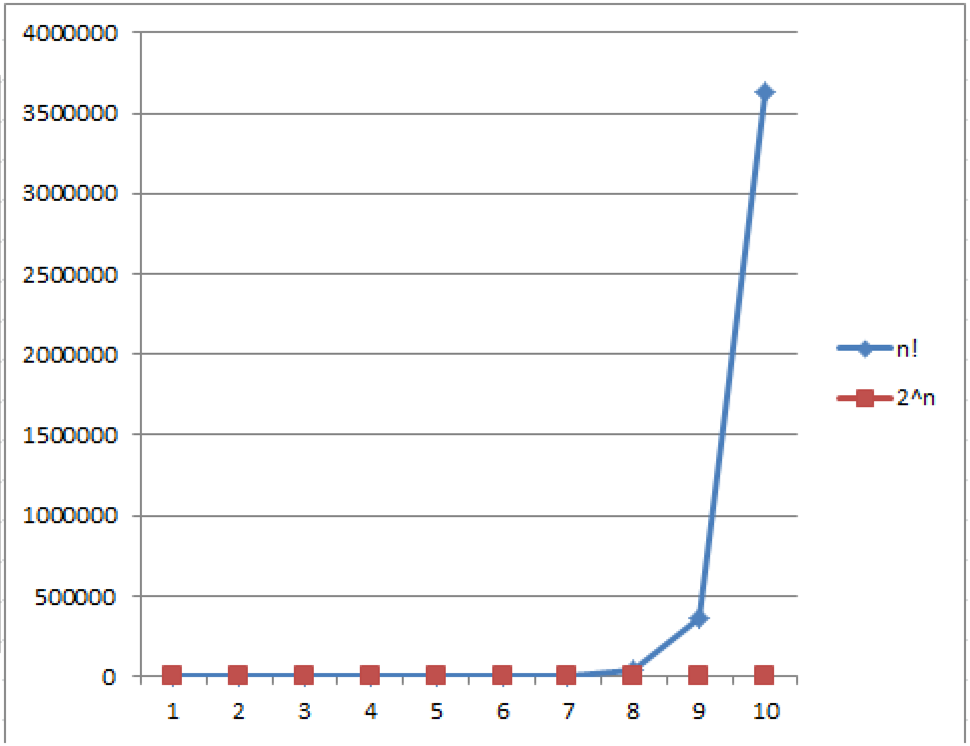
- Use indução para mostrar que o fatorial cresce mais rápido que a exponencial: $n! \geq 2^{n-1}$ para todo o $n \geq 1$
- **Estrutura indutiva**: números naturais
- **Afirmação Q(n)**: $n! \geq 2^{n-1}$
- **Passo base**: Q(1): $1! \geq 2^{1-1} = 2^0 = 1$
- **Passo indutivo**: provar que $(n+1)! \geq 2^n$

$$
\begin{aligned}
(n+1)! &= (n+1)n(n-1)(n-2)\ldots2.1 \\
&= (n+1)\,(n!) \\
&\geq (n+1)\,2^{n-1} && \text{pela hipótese} \\
&\geq 2.2^{n-1} && (n+1) \geq 2 \\
&= 2^n.
\end{aligned}
$$

**descobrir Q(n) em Q(n+1) para usar a hipótese**

# Exponencial e fatorial




- O fatorial aumenta mais rapidamente que a exponencial

# Provar propriedades de programas

- Programa

```
void Funtion(int n)
    {int x,y,z;
            x = n;
            z = 1;
            y = 0;
        while( x>0 ) {
            y = z+y;
            z = z+2;
            x = x-1;}
        printf(‘‘n^2 = %d, 2n+1 = %d’’, y, z);
    }
```

- Provar: quando executado, o programa imprime os valores de $n^2$ e 2n+1 para uma entrada n

# Prova

- Para provar a propriedade do programa
- **Lema 1**: Dada uma entrada n, haverá exatamente n iterações do ciclo while
  - Prova: por indução

$\forall n$ [(n é entrada $\rightarrow$ Q(n)]

Q: há exatamente n iterações do ciclo

<u>Caso base</u>: n= 0 para x=0 não se entra no ciclo while

<u>Passo indutivo</u>: Seja um número natural k para o qual Q(k) se verifica

Se a entrada for k+1: x fica com k+1

Entra-se no ciclo, é executado 1ª vez e x decrementado

Agora x=k e o ciclo é executado k vezes

No total: ciclo executado k+1 vezes

# Prova

- **Lema 2**: Depois de k iterações do ciclo while, y e z têm os valores $k^2$ e $2k+1$, respetivamente
  - Prova: por indução

**Invariante do ciclo**

$\forall k\ [k \in \mathbb{N} \rightarrow Q(k)]$

Q: depois de k iterações do ciclo while, y e z têm os valores $k^2$ e $2k+1$

<u>Caso base</u>: $k= 0$ ciclo não é executado, $y=0= k^2$ e $z=1=2k+1$

<u>Passo indutivo</u>: Seja um número natural k para o qual Q(k) se verifica

Após mais uma iteração do ciclo while:

$y = z+y = k^2 + 2k+1 = (k+1)^2$

$z = z+2 = 2k+1 +2 = 2(k+1) +1$

# Cálculo do fatorial

```
Fatorial(n){
i=1
fat=1
while (i<n) {
   i= i+1
   fat= fat*i
   }
}
```

- Mostrar que, no final, fat=n!
- Invariante (afirmação a provar): no final de cada ciclo fat=i!
- Base: antes do ciclo i=1 e fat =1=i!
- Indutivo: assumir que fat=i!; se i<n executa-se outro ciclo e i passa a i+1 e fat passa a fat*(i+1)=i!*(i+1)= (i+1)!

# Exemplo

- **Prove que $5^n$-1 é divisível por 4, para $n \geq 1$.**

- Afirmação Q(n): $5^n - 1 = 4k$ (k inteiro)
- Passo base: n=1, $5^1-1=4$ é divisível por 4
- Passo indutivo:

$$
\begin{aligned}
5^{n+1}-1 &= 5.5^n-1 \\
&= (4+1)5^n-1 \\
&= 4(5^n) + 5^n-1 \\
&= 4(5^n)+4k && \text{pela hipótese} \\
&= 4(5^n+k) && \text{é divisível por 4}
\end{aligned}
$$

# Números harmónicos

- Número harmónico de ordem k
- $H_k = 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{k}$
- Mostre que $H_{2^n} \geq 1 + \frac{n}{2}$
  - isto é, os números harmónicos podem ser arbitrariamente grandes

- $H_1 = 1$
- $H_2 = 1 + \frac{1}{2}$
- $H_3 = 1 + \frac{1}{2} + \frac{1}{3} = H_2 + \frac{1}{3}$
- $H_4 = 1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} = H_2 + \frac{1}{3} + \frac{1}{4} = H_3 + \frac{1}{4}$

• Analisar alguns exemplos iniciais para mais tarde abstrair
• Manter a estrutura toda para evidenciar relações

# Números harmónicos (cont.)

- **Afirmação Q(n):** $\boldsymbol{H_{2^n} \geq 1 + \frac{n}{2}}$
- Passo base: $H_{2^0} = 1 \geq 1 = 1 + \frac{0}{2}$
- Passo indutivo
- $H_{2^{n+1}} = 1 + \frac{1}{2} + \cdots + \frac{1}{2^n} + \frac{1}{2^n+1} + \cdots + \frac{1}{2^{n+1}}$
- $\qquad = H_{2^n} + \frac{1}{2^n+1} + \cdots + \frac{1}{2^{n+1}}$
- $\qquad \geq 1 + \frac{n}{2} + \frac{1}{2^n+1} + \cdots + \frac{1}{2^{n+1}}$ pela hipótese
- $\qquad \geq 1 + \frac{n}{2} + \frac{1}{2^{n+1}} + \cdots + \frac{1}{2^{n+1}}$ porque $\frac{1}{2^n+1} \geq \frac{1}{2^{n+1}}$
- $\qquad = 1 + \frac{n}{2} + 2^n \frac{1}{2^{n+1}} = 1 + \frac{n}{2} + \frac{1}{2} = 1 + \frac{n+1}{2}$

# Números harmónicos (cont.)

- **Afirmação Q(n):** $\boldsymbol{H_{2^n} \leq 1 + n}$

- Passo base: $H_{2^0} = 1 \leq 1 = 1 + 0$
- Passo indutivo
- $H_{2^{n+1}} = 1 + \frac{1}{2} + \cdots + \frac{1}{2^n} + \frac{1}{2^n+1} + \cdots + \frac{1}{2^{n+1}}$
- $\qquad = H_{2^n} + \frac{1}{2^n+1} + \cdots + \frac{1}{2^{n+1}}$
- $\qquad \leq 1 + n + \frac{1}{2^n+1} + \cdots + \frac{1}{2^{n+1}}$ pela hipótese
- $\qquad \leq 1 + n + \frac{1}{2^{n+1}} + \cdots + \frac{1}{2^{n+1}}$ porque $\frac{1}{2^n+1} \geq \frac{1}{2^{n+1}}$
- $\qquad = 1 + n + 2^n \frac{1}{2^{n+1}} \leq 1 + (n + 1)$

# Forma forte da indução matemática

- Dada uma afirmação Q(n) suponha que

1. Q é verdade para um inteiro $n_0$
2. Se $k>n_0$ é um inteiro qualquer e P é verdade para todos os inteiros l na gama $n_0 \leq l < k$, então também é verdade para k

Então Q(n) é verdade para todos os inteiros $n \geq n_0$.

- Aplica-se por exemplo nas expressões ambig-wff para permitir usar numa prova subexpressões de todas as iterações anteriores da definição indutiva
- Prova-se que esta forma é **equivalente** à forma normal do princípio da indução matemática

# Princípio da boa ordenação

- Princípio da boa ordenação para inteiros não negativos

> Qualquer conjunto de inteiros não negativos tem um elemento mínimo

- É também equivalente às duas formas da indução, usando a definição indutiva na forma de conjunto de inteiros