Algoritmy a jejich složitost

Ω Θ O

A0B36PRI-Programování

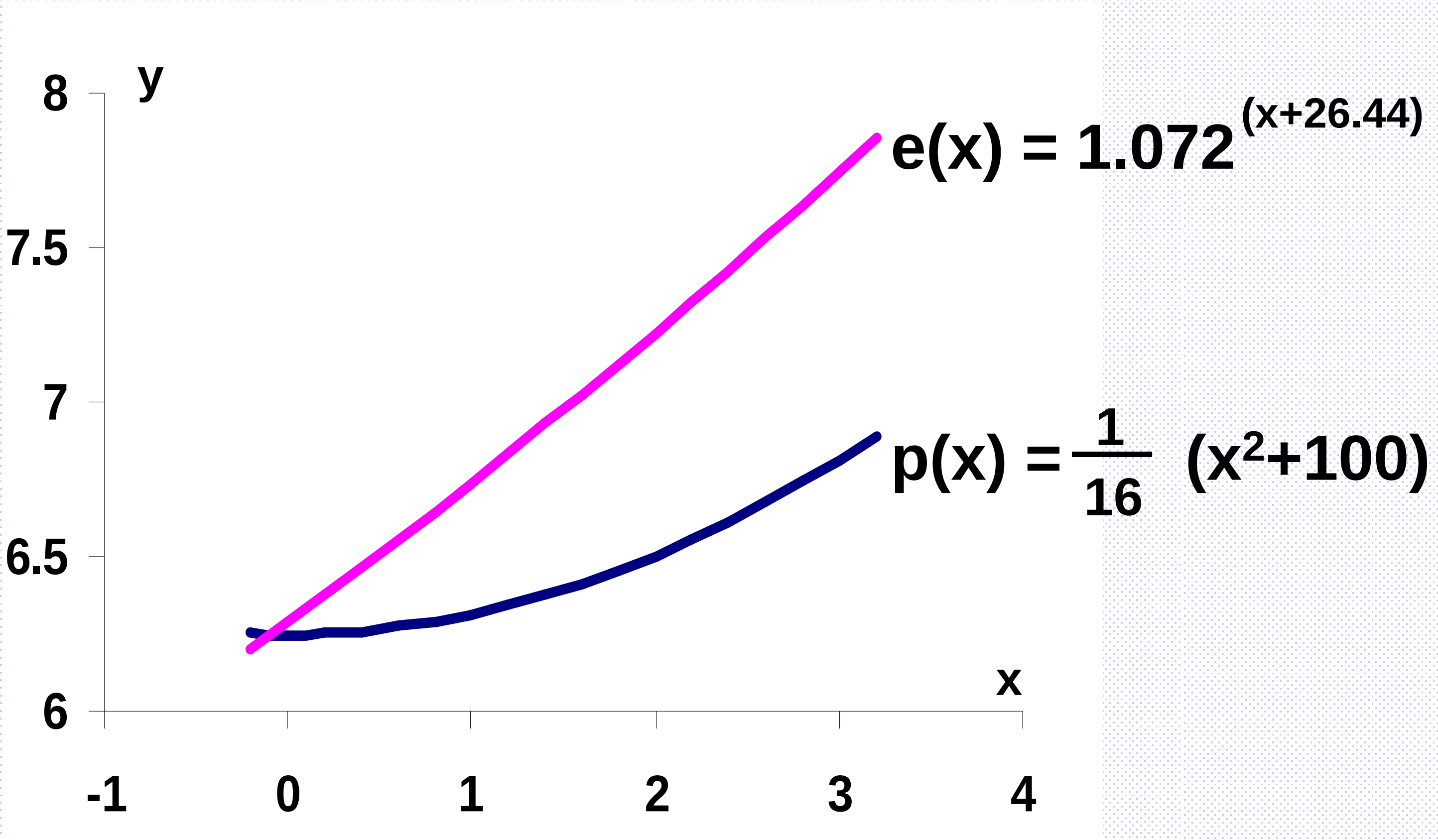
Fakulta elektrotechnická
České vysoké učení technické

Řád růstu funkcí

Ω Θ O

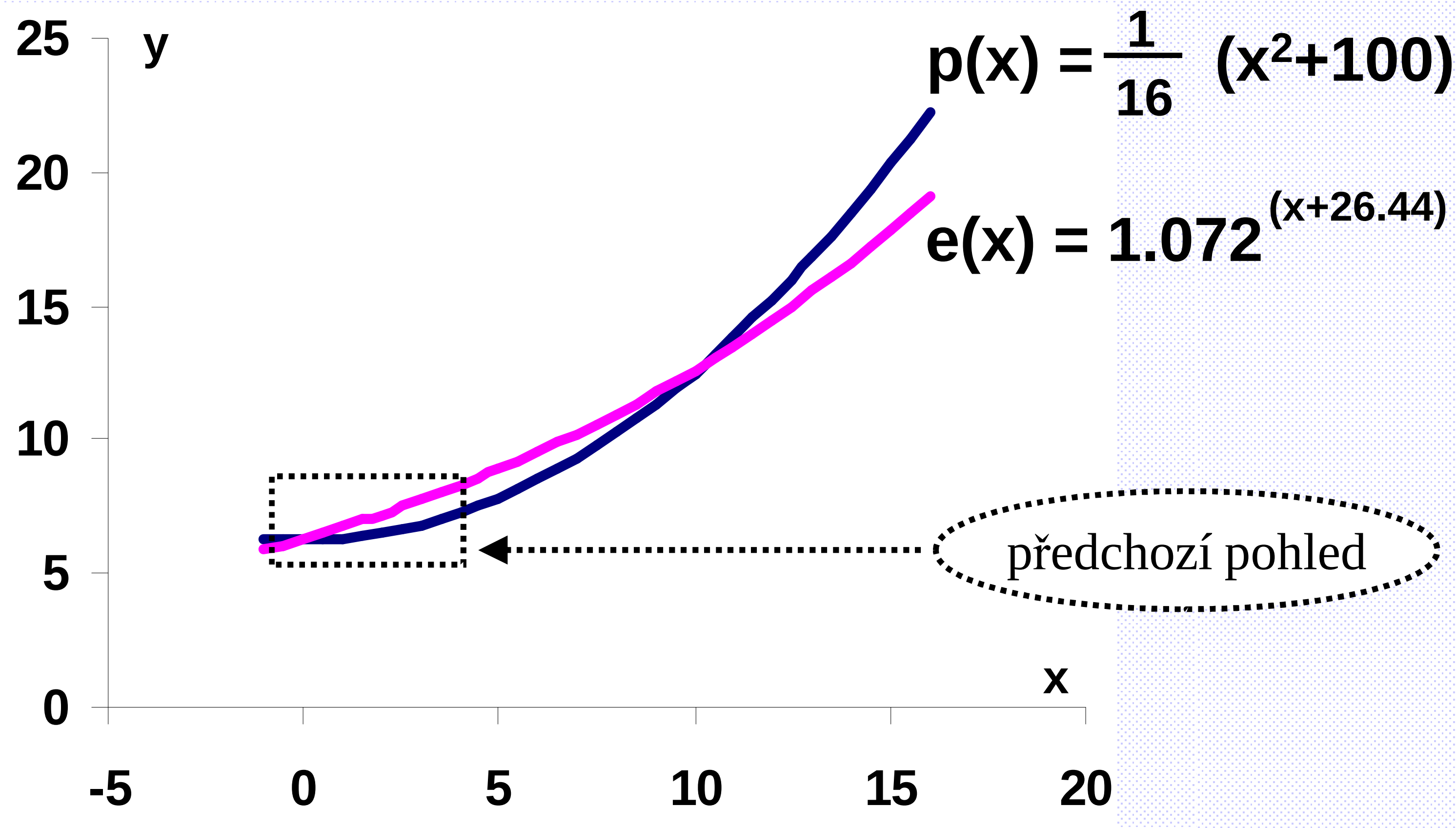
Řád růstu funkcí

Řád růstu funkcí



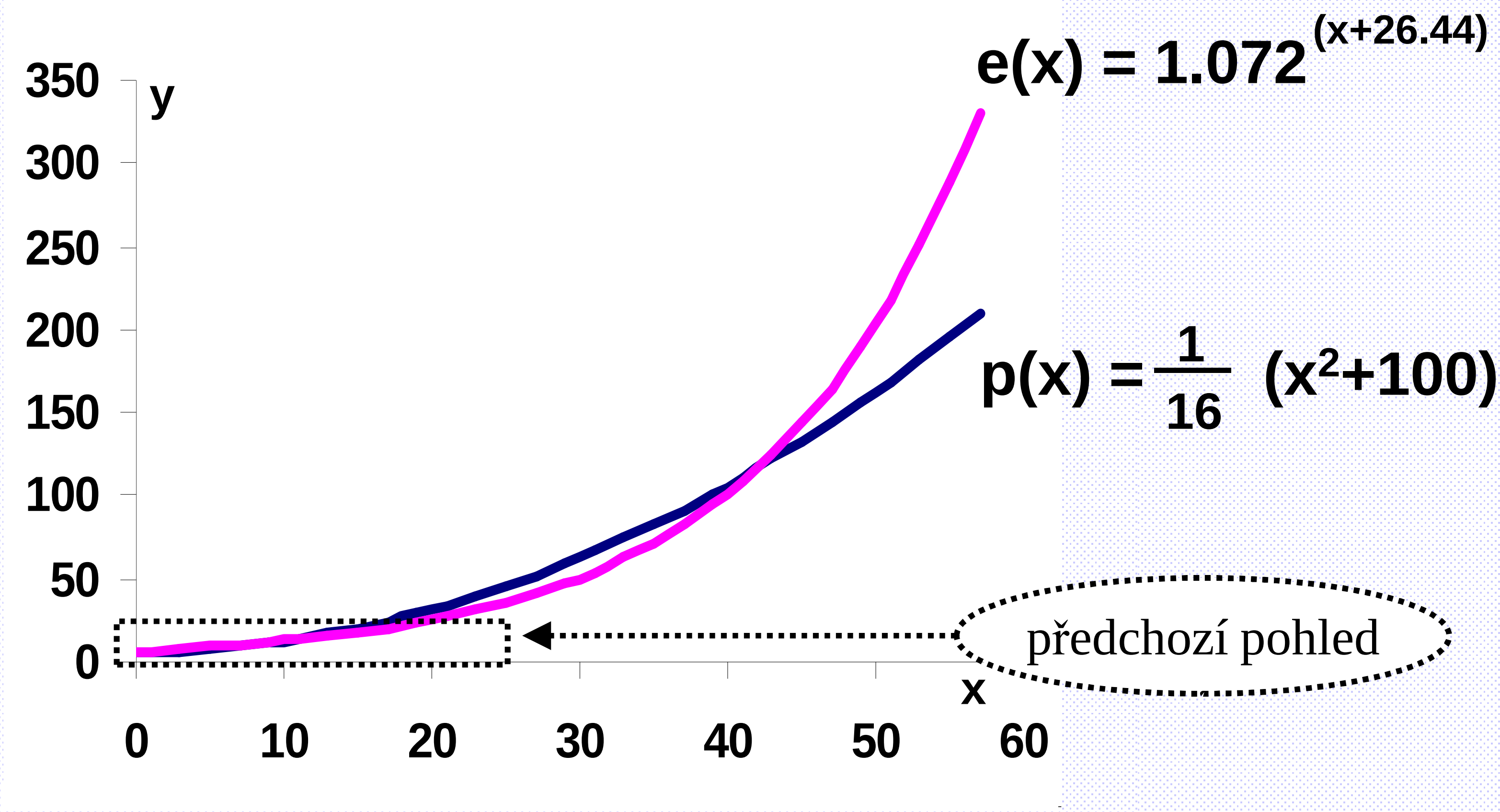
Řád růstu funkcí

Zoom out! :



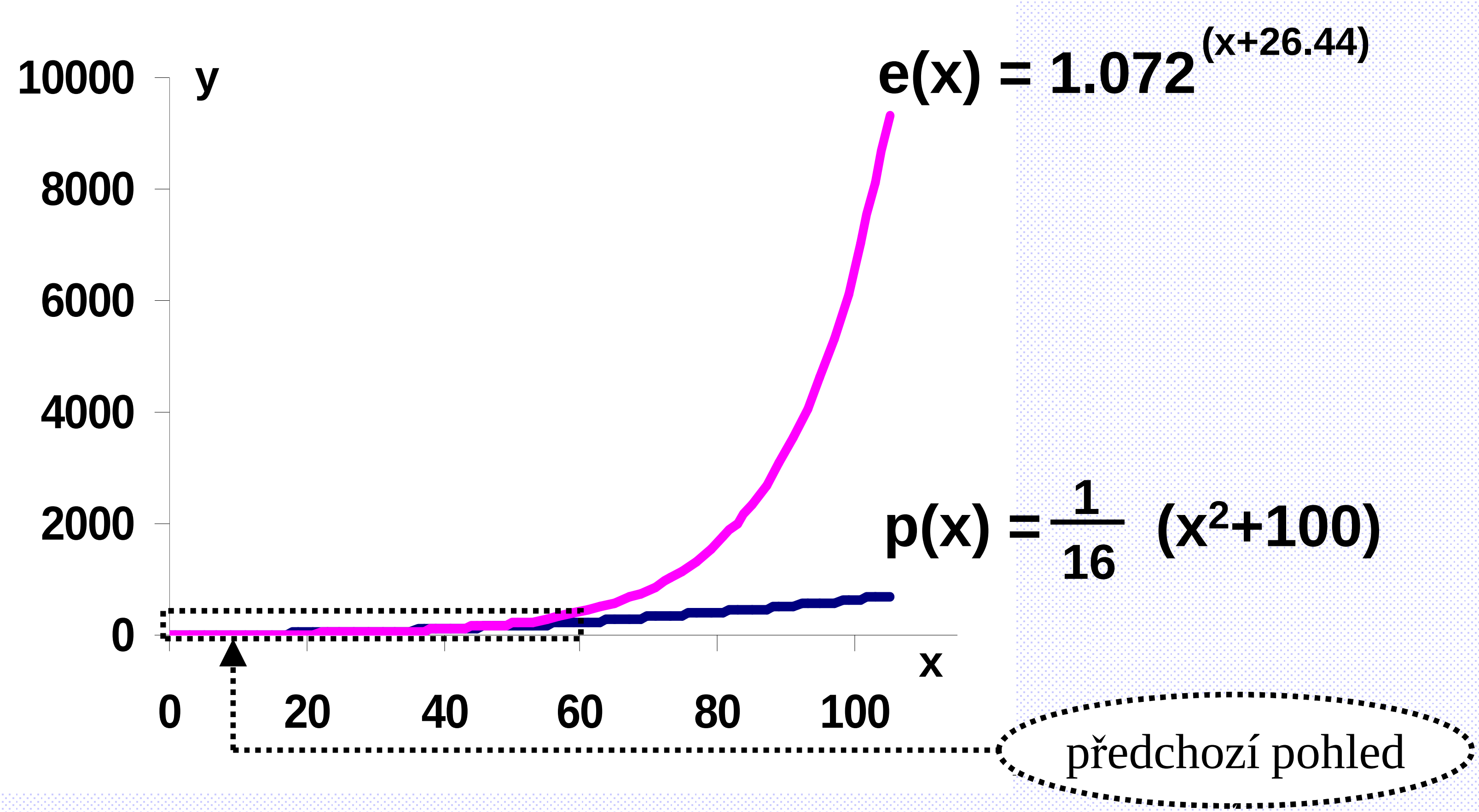
Řád růstu funkcí

Zoom out!:



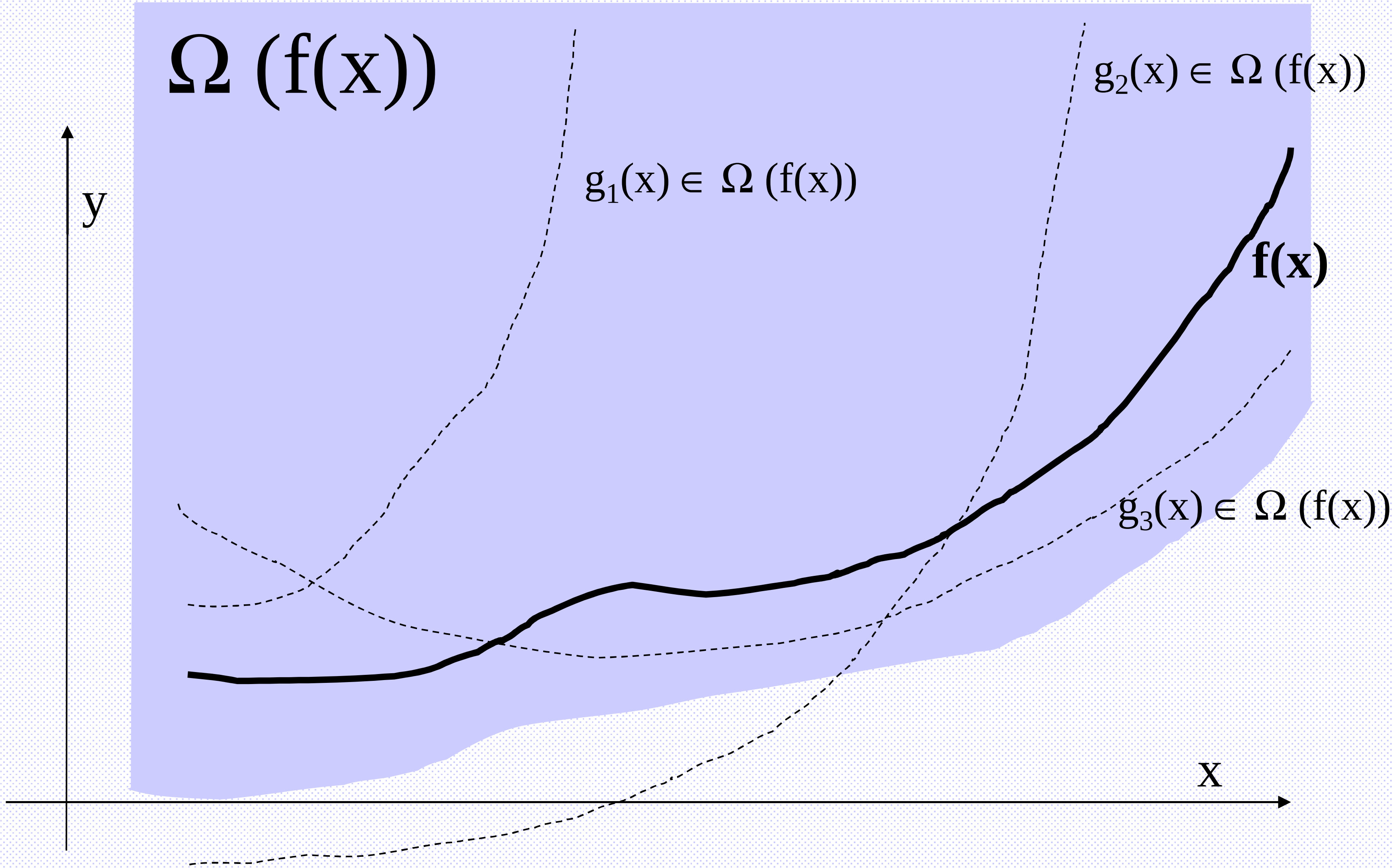
Řád růstu funkcí

Zoom out! :



atd:… e(1000) = 9843181236605408906547628704342.9 p(1000) = 62506.25 …

Řád růstu funkcí



Řád růstu funkcí

Ω Omega

Ω(f(x)) Množina všech funkcí rostoucích rychleji nebo stejně rychle jako f(x).

V množině Ω(f(x))
se octne každá funkce g(x), která od určitého bodu x_0
(není nijak předem předepsáno, kde by x_0 měl být)

a) – je už vždy větší než funkce f(x)

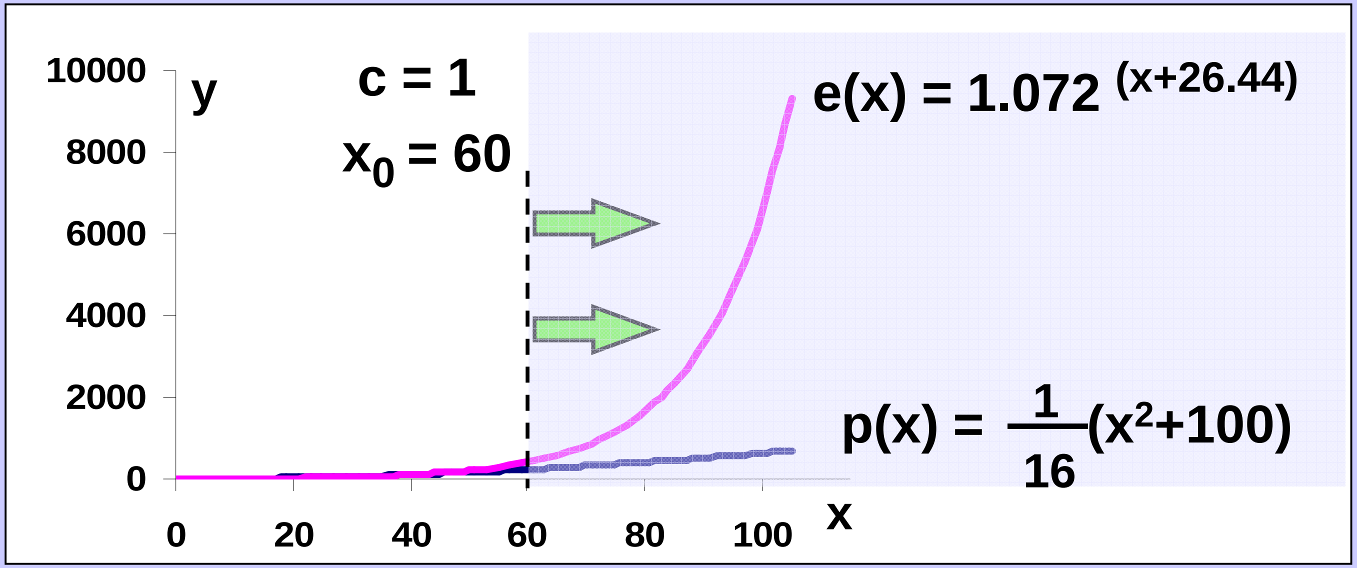
b) – sice větší než f(x) není, ale po vynásobení
f(x) nějakou kladnou konstantou c (c < 1)
(hodnota konstanty ale také není nijak předepsána)
je už g(x) vždy větší než "zmenšená" funkce c·f(x).

Takže: pokud najdeme nějaké x_0 a c>0 takové, že
g(x) > c·f(x) všude napravo od x_0, (někdy stačí c=1)
je jisto, že g(x) ∈ Ω(f(x))

Řád růstu funkcí

Ω Omega

Takže: pokud najdeme nějaké x_0 a c>0 takové, že
g(x) > c·f(x) všude napravo od x_0, (někdy stačí c=1)
je jisto, že $g(x) \in \Omega(f(x))$



$x > 60 \Rightarrow$ e(x) > p(x), tj. $1.072^{(x+26.44)} > \frac{1}{16}(x^2+100)$

tudíž platí $\boxed{e(x) \in \Omega(p(x))}$ (ověřte!)

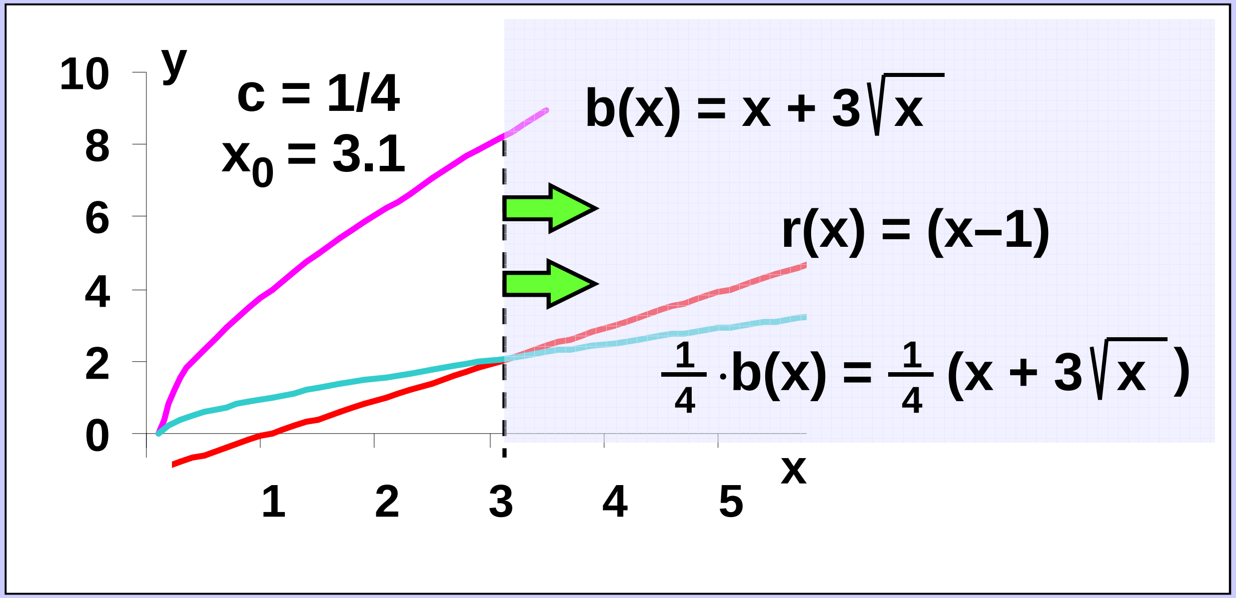
Řád růstu funkcí

Ω Omega

Takže: pokud najdeme nějaké x_0 a c>0 takové, že
$g(x) > c \cdot f(x)$ všude napravo od x_0, (někdy stačí c=1)
je jisto, že $g(x) \in \Omega(f(x))$

$b(x) = x + 3\sqrt{x}$

$r(x) = x - 1$



$x > 3.1 \quad \Rightarrow \quad r(x) > \frac{1}{4} \cdot b(x)$, tj. $4(x-1) > x + 3\sqrt{x}$ (ověřte!)

tudíž platí $\boxed{r(x) \in \Omega(b(x))}$

Řád růstu funkcí

Ω Omega

Typické ukázky

$x^2 \in \Omega(x)$ $\quad x^3 \in \Omega(x^2)$ $\quad x^{n+1} \in \Omega(x^n)$

$2^x \in \Omega(x^2)$ $\quad 2^x \in \Omega(x^3)$ $\quad 2^x \in \Omega(x^{5000})$

$x \in \Omega(\log(x))$ $\quad x \cdot \log(x) \in \Omega(x)$ $\quad x^2 \in \Omega(x \cdot \log(x))$

$2^x \in \Omega(x^{20000})$ $\quad x^{20000} \in \Omega(x)$ $\quad x \in \Omega(1)$

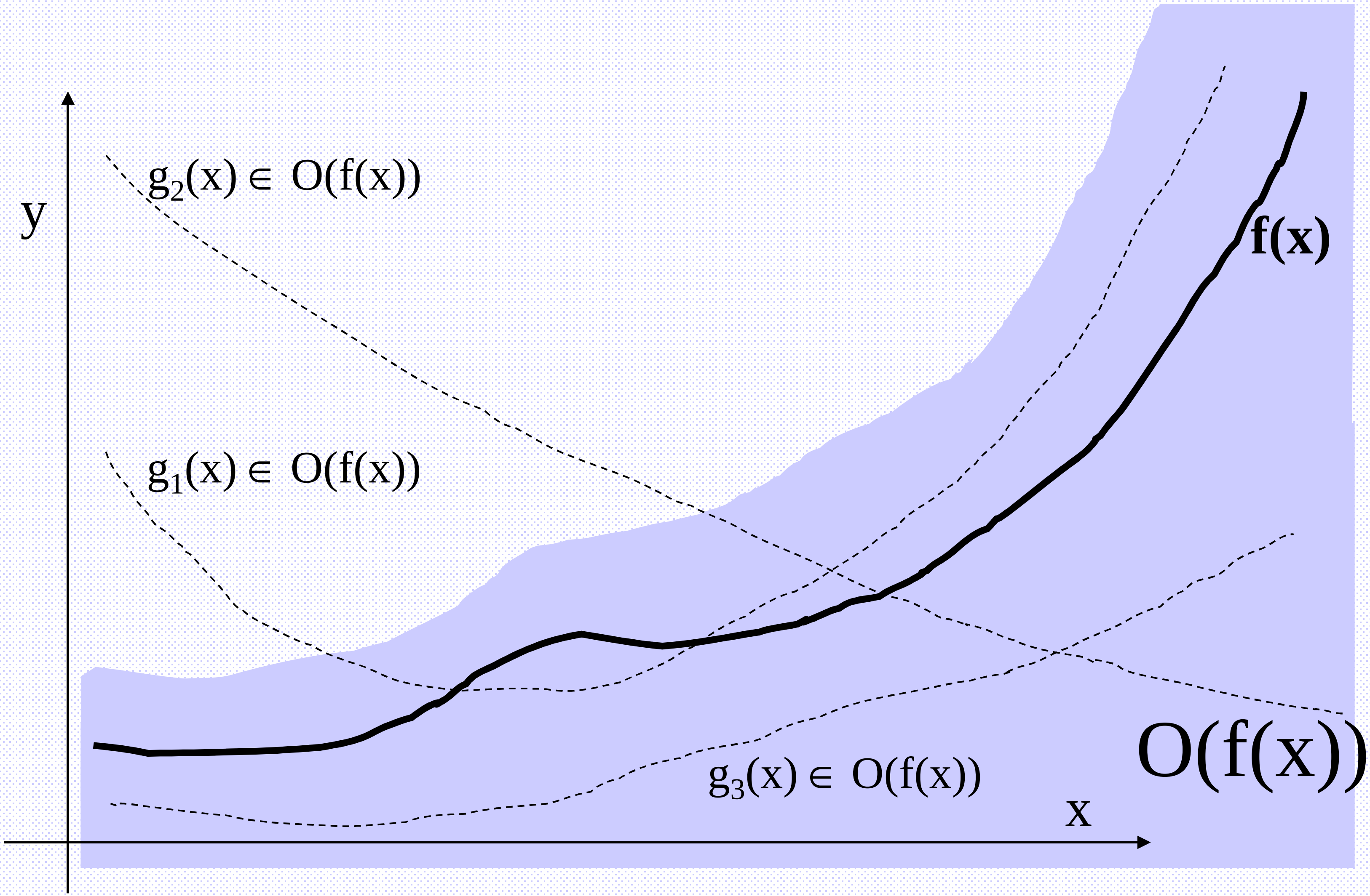
vždy

$$f(x) > 1 \quad \Rightarrow \quad f(x) \in \Omega(1)$$

těžko uvěřitelné

$$\sqrt[200\,000]{x} \in \Omega(\log(x)^{200\,000})$$

Řád růstu funkcí



Řád růstu funkcí

O Omikron

O(f(x)) Množina všech funkcí rostoucích pomaleji nebo stejně rychle jako f(x).

V množině O(f(x))
se octne každá funkce g(x), která od určitého bodu x_0
(není nijak předem předepsáno, kde by x_0 měl být)

a) – je už vždy menší než funkce f(x)

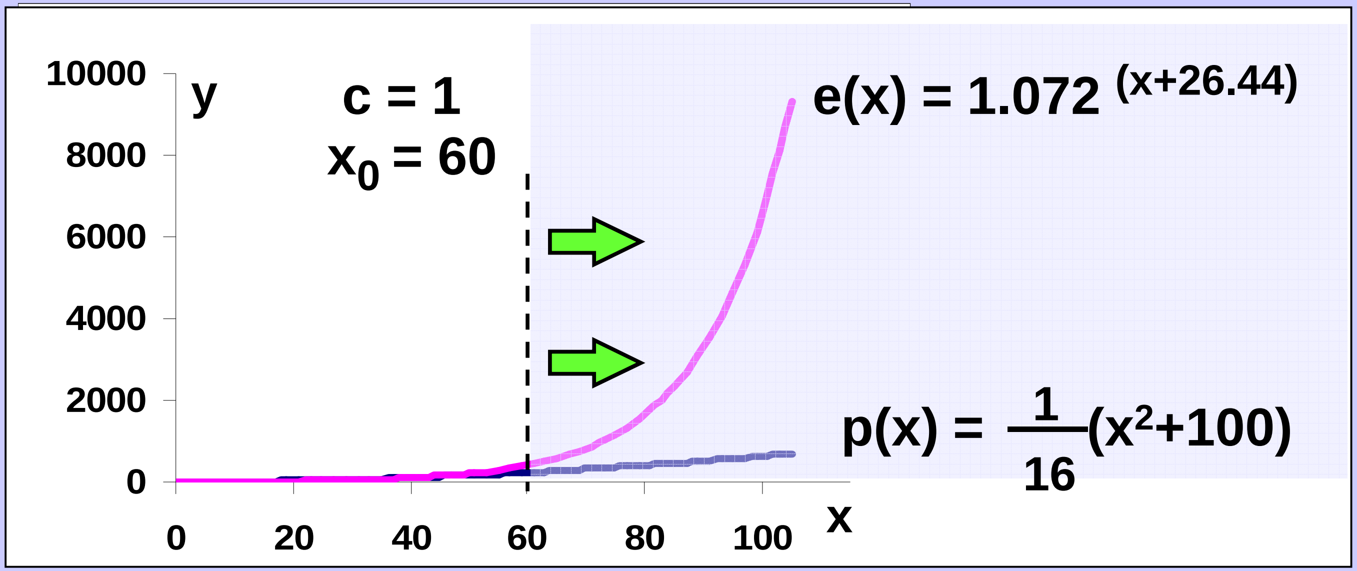
b) – sice menší než f(x) není, ale po vynásobení
nějakou kladnou konstantou (asi < 1 ☺)
(hodnota konstanty také není nijak předepsána)
je už vždy menší než funkce f(x).

Takže: pokud najdeme nějaké x_0 a c>0 takové, že
$c \cdot g(x) < f(x)$ všude napravo od x_0, (někdy stačí c=1)
je jisto, že $g(x) \in O(f(x))$

Řád růstu funkcí

O Omikron

Takže: pokud najdeme nějaké x_0 a c>0 takové, že
$c \cdot g(x) < f(x)$ všude napravo od x_0, (někdy stačí c=1)
je jisto, že $g(x) \in O(f(x))$



$x > 60 \quad \Rightarrow \quad p(x) < e(x)$, tj. $\frac{1}{16}(x^2+100) < 1.072^{(x+26.44)}$

tudíž platí $\boxed{p(x) \in O(e(x))}$ ověřte!

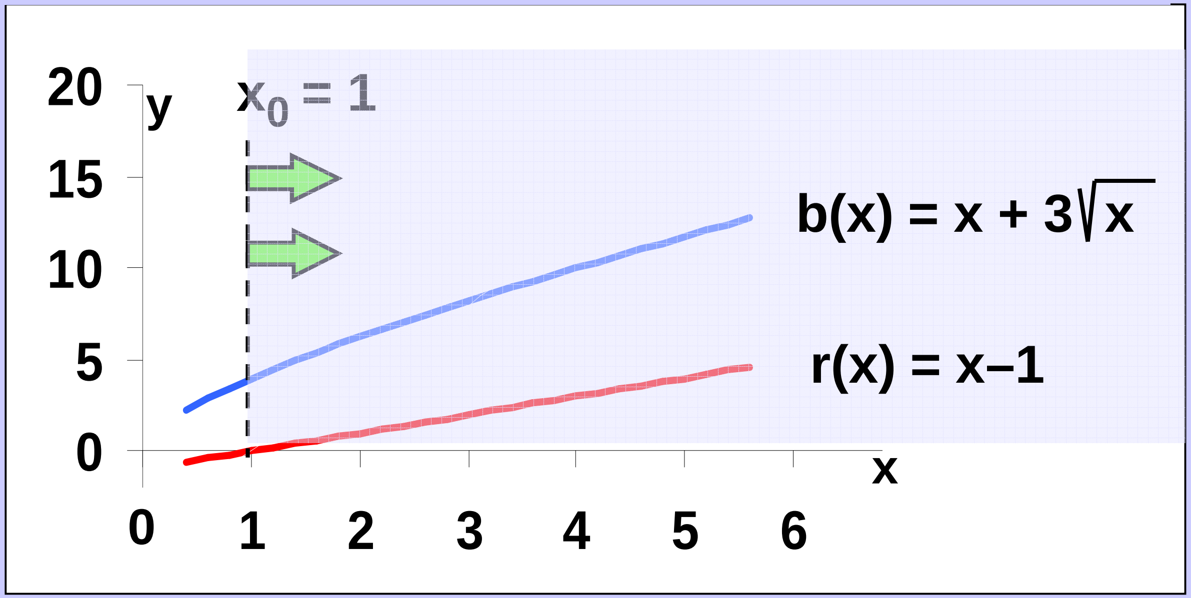
Řád růstu funkcí

O Omikron

Takže: pokud najdeme nějaké x_0 a c>0 takové, že $c \cdot g(x) < f(x)$ všude napravo od x_0, (někdy stačí c=1) je jisto, že $g(x) \in O(f(x))$

$b(x) = x + 3\sqrt{x}$

$r(x) = x-1$



$x > 1 \Rightarrow r(x) < b(x)$, tj. $x-1 < x + 3\sqrt{x}$

tudíž platí $\boxed{r(x) \in O(b(x))}$

Řád růstu funkcí

O Omikron

$$f \in \Omega(g) \iff g \in O(f)$$

$x \in O(x^2)$ $\quad$ $x^2 \in O(x^3)$ $\quad$ $x^n \in O(x^{n+1})$

$x^2 \in O(2^x)$ $\quad$ $x^3 \in O(2^x)$ $\quad$ $x^{5000} \in O(2^x)$

$\log(x) \in O(x)$ $\quad$ $x \in O(x \cdot \log(x))$ $\quad$ $x \cdot \log(x) \in O(x^2)$

$x^{20000} \in O(2^x)$ $\quad$ $x \in O(x^{20000})$ $\quad$ $1 \in O(x)$

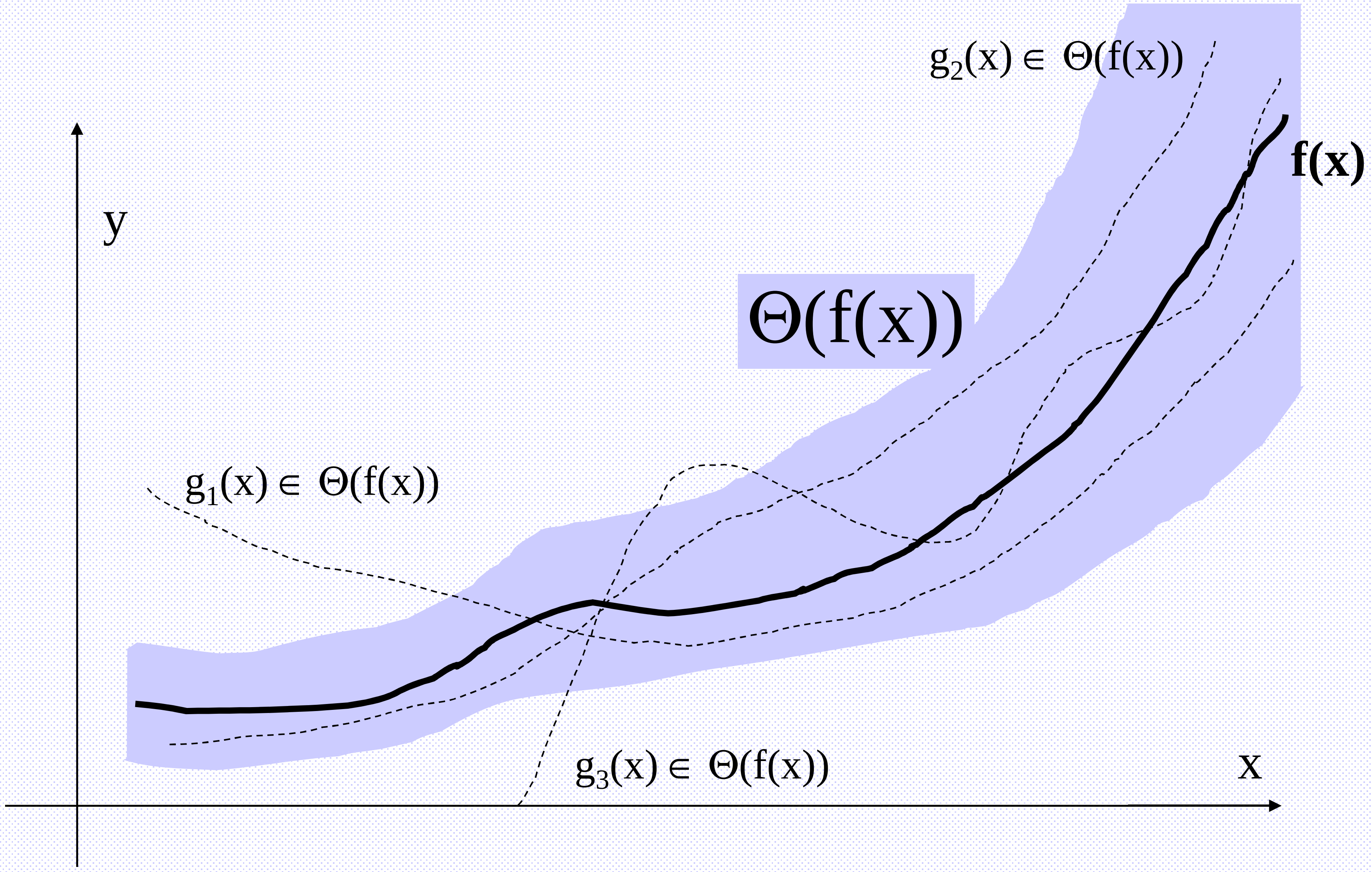
vždy

$$f(x) > 1 \Rightarrow 1 \in O(f(x))$$

těžko uvěřitelné

$$\log(x)^{200\,000} \in O(\sqrt[200\,000]{x})$$

Řád růstu funkcí



Řád růstu funkcí

Θ Theta

$\Theta(f(x))$ Množina všech funkcí rostoucích stejně rychle jako f(x).

$$\Theta(f(x)) = \Omega(f(x)) \cap O(f(x))$$

V množině $\Theta(f(x))$ se octne každá funkce g(x), která spadá jak do $\Omega(f(x))$ tak do $O(f(x))$.

$$f(x) \in \Theta(g(x)) \quad \Leftrightarrow \quad g(x) \in \Theta(f(x))$$

Řád růstu funkcí

Θ Theta

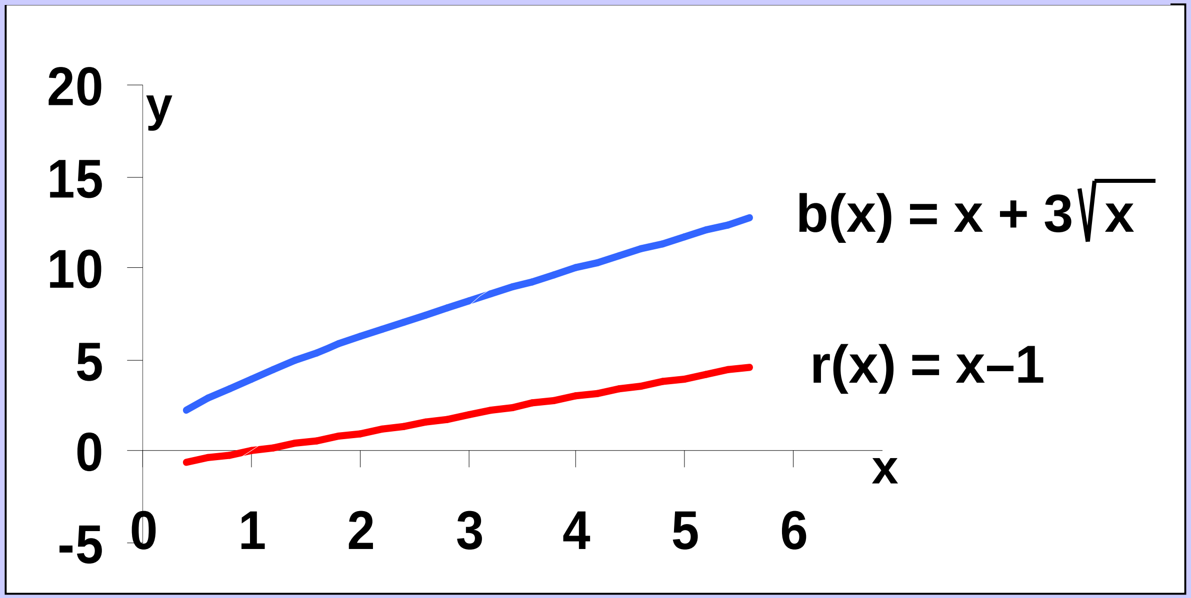
$$f(x) \in \Theta(g(x)) \quad \Leftrightarrow \quad g(x) \in \Theta(f(x))$$

$b(x) = x + 3\sqrt{x}$

$r(x) = x-1$

$r(x) \in \Omega(b(x))$

$r(x) \in O(b(x))$



$r(x) \in \Theta(b(x))$ $\quad$ $b(x) \in \Theta(r(x))$

Řád růstu funkcí

Pravidla

1. $(a > 0) \Leftrightarrow \Theta(f(x)) = \Theta(a \cdot f(x))$

2. $g(x) \in O(f(x)) \Leftrightarrow \Theta(f(x)) = \Theta(f(x) + g(x))$

Slovy

1. Multiplikativní konstanta
   nemá vliv na náležení do množiny $\Theta(f(x))$.
2. Přičtení/odečtení „menší“ funkce
   nemá vliv na náležení do množiny $\Theta(f(x))$.

Příklady

$1.8x + 600 \cdot \log_2(x) \in \Theta(x)$

$x^3 + 7x^{1/2} + 5(\log_2(x))^4 \in \Theta(x^3)$

$13 \cdot 3^x + 9x^{12} + 42x^{-4} + 29 \in \Theta(3^x)$

$4 \cdot 2^n + 3 \cdot 2^{n-1} + 5 \cdot 2^{n/2} \in \Theta(2^n)$

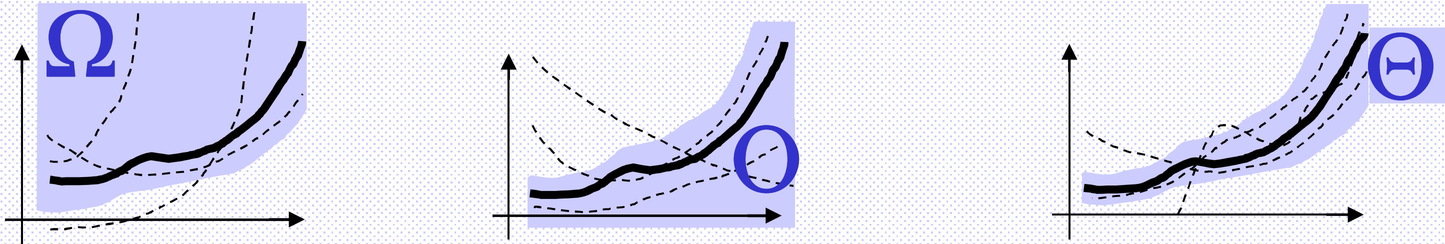
$0.1x^5 + 200x^4 + 7x^2 - 3 \in \Theta(x^5)$

–”– $\in O(x^5)$

–”– $\in \Omega(x^5)$

Řád růstu funkcí

Množiny Ω, O, Θ



Pozor, obrázky jsou jen ilustrační

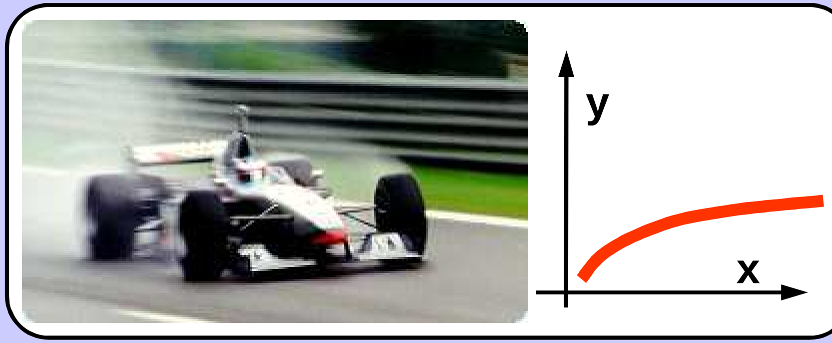
Exaktní definice

$\Omega(f(x)) = \{\, g(x) \;;\; \exists x_0>0, c>0 \;\forall x>x_0 : c \cdot f(x) < g(x) \,\}$

$O(f(x)) = \{\, g(x) \;;\; \exists x_0>0, c>0 \;\forall x>x_0 : c \cdot f(x) > g(x) \,\}$

$\Theta(f(x)) = \{\, g(x) \;;\; \exists x_0>0, c_1>0, c_2>0 \;\forall x>x_0 : c_1 \cdot f(x) < g(x) < c_2 \cdot f(x) \,\}$

Asymptotická složitost



Každému algoritmu lze jednoznačně přiřadit

rostoucí funkci

zvanou

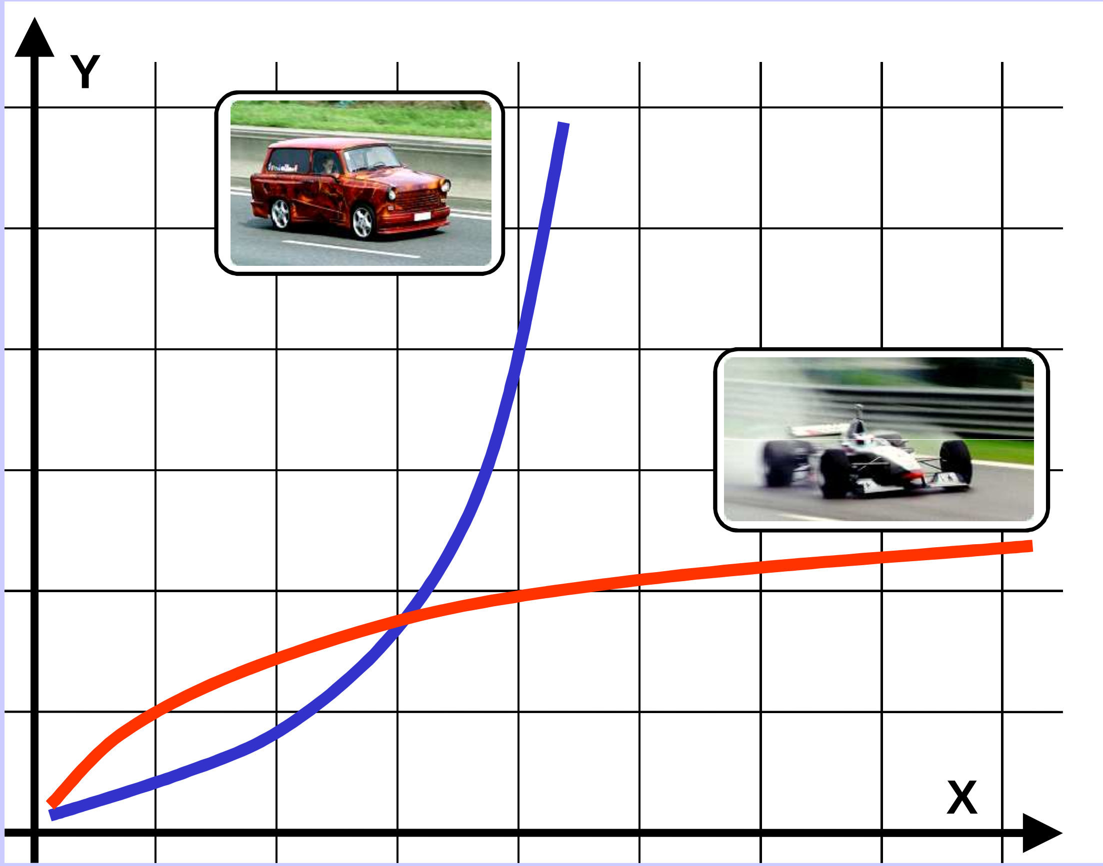
asymptotická složitost,

která charakterizuje počet elementárních operací algoritmu v závislosti na rostoucím rozsahu vstupních dat.

Čím pomaleji tato funkce roste, tím je algoritmus rychlejší.

Asymptotická složitost

Y ~ trvání výpočtu
(počet
elementárních
operací)



X ~ naše požadavky
(rozsah vstupních dat)

Řád růstu funkcí

Porovnání rychlosti růstu funkcí

Funkce f(x) roste asymptoticky rychleji než funkce g(x), když

$$f(x) \in \Omega(g(x)) \;\; \& \;\; f(x) \notin \Theta(g(x))$$

Pozor!

Porovnání rychlosti algoritmů

Algoritmus A je asymptoticky pomalejší než algoritmus B, když

$$f_A(n) \in \Omega(f_B(n)) \;\; \& \;\; f_A(n) \notin \Theta(f_B(n)),$$

kde $f_A(n)$, resp. $f_B(n)$ je funkce určující počet operací,
které provede algoritmus A, resp. B,
při zpracování dat o rozsahu n.

Řád růstu funkcí

Řád růstu funkce

Řád růstu funkce f je taková "co nejjednodušší" funkce g, pro kterou platí

$$g(x) \in \Theta(f(x))$$

Manipulace

Řád růstu funkce f získáme většinou tak, že zanedbáme

1. Aditivní členy rostoucí pomaleji nebo stejně rychle
2. Multiplikativní konstantu

Příklady

$ff(n) = 4 \cdot 2^n + 3 \cdot 2^{n-1} + 5 \cdot 2^{n/2} \in \Theta(2^n)$ Řád růstu ff(n) je 2^n

$hh(x) = x + \log_2(x) - \sqrt{x} \in \Theta(x)$ Řád růstu hh(x) je x

Asymptotická složitost

Asymptotická složitost algoritmu

Asymptotická složitost algoritmu A
je řád růstu funkce f(n), která charakterizuje
počet elementárních operací algoritmu A
při zpracování dat o rozsahu n.

(rozsah dat = celkový počet čísel a znaků v nich)

Ve většině případů nehraje roli, zda uvažujeme

A) počet všech elementárních operací
B) počet všech elementárních operací nad daty
C) počet testů nad daty

Asymptotická složitost vychází táž.

Asymptotická složitost

Asymptotická složitost předložených ukázek

Za předpokladu, že pole má délku n, platí:

Asymptotická složitost hledání minima a maxima v poli je n. V obou uvedených případech.

Asymptotická složitost pomalého zjišťování, kolik čísel v poli je rovno součtu jiného pole, je n^2.

Asymptotická složitost rychlého zjišťování, kolik čísel v poli je rovno součtu jiného pole, je n.

Asymptotická složitost lineárního hledání prvku v poli je n.

Asymptotická složitost hledání prvku uspořádaném poli pomocí půlení intervalu je log(n).

Asymptotická složitost

Úmluvy

Zjednodušení

Běžně se zkráceným termínem „složitost algoritmu“ rozumí právě výraz „asymptotická složitost algoritmu“

Zmatení

Běžně se v literatuře neříká f(x) <u>náleží do</u> $\Theta(g(x))$,

ale f(x) <u>je</u> $\Theta(g(x))$.

Rovněž se to tak značí: $f(x) = \Theta(g(x))$

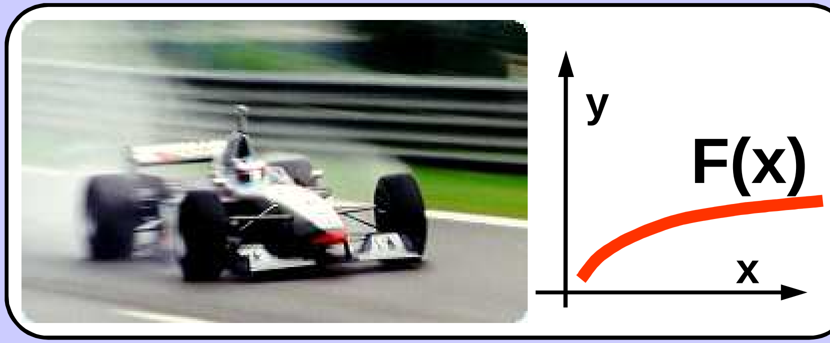
namísto $f(x) \in \Theta(g(x))$

(a stejně pro O a Ω).

Uklidnění

Chápe se to ale beze změny, v původním smyslu <u>náležení</u>.

Asymptotická složitost



$$F(x) \in O(T(x))$$
$$F(x) \notin \Theta(T(x))$$
$$F(x) \notin \Omega(T(x))$$

$$T(x) \in \Omega(F(x))$$
$$T(x) \notin \Theta(F(x))$$
$$T(x) \notin O(F(x))$$

Klidné vánoce a veselé zdraví v novém roce

přeje Ivan Jelínek & kolektiv učitelů PR1