Algorytmy i struktury danych I
wykład

Karol Szałowski

WYDZIAŁ FIZYKI
i INFORMATYKI STOSOWANEJ
Uniwersytet Łódzki

2019/2020

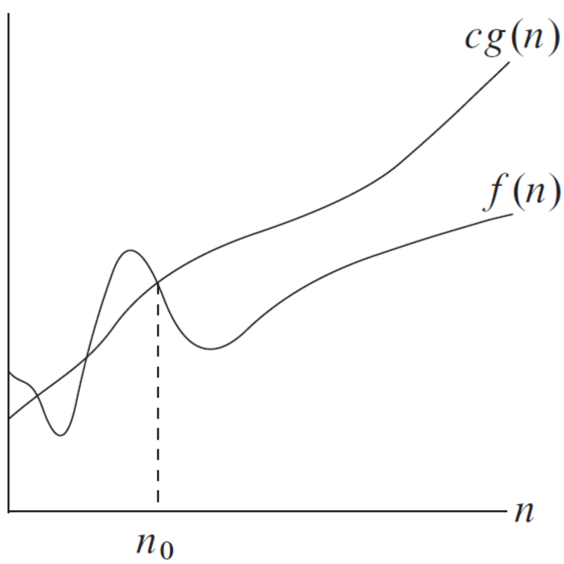
karol.szalowski@uni.lodz.pl

WYDZIAŁ FIZYKI
i INFORMATYKI STOSOWANEJ
Uniwersytet Łódzki

Notacja O

Niech $f(n)$ oznacza funkcję opisującą zależność liczby operacji dominujących od rozmiaru danych wejściowych, $g(n)$ pewną funkcję referencyjną

$$f(n) = O(g(n)) \Leftrightarrow \exists_{c>0,n_0>0} \forall_{n>n_0}\ 0 \leqslant f(n) \leqslant cg(n)$$

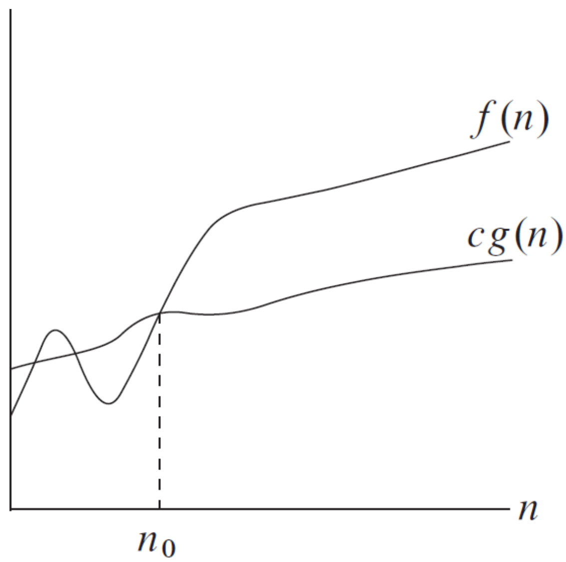


WYDZIAŁ FIZYKI
i INFORMATYKI STOSOWANEJ
Uniwersytet Łódzki

Notacja Ω

Niech $f(n)$ oznacza funkcję opisującą zależność liczby operacji dominujących od rozmiaru danych wejściowych, $g(n)$ pewną funkcję referencyjną

$$f(n) = \Omega(g(n)) \Leftrightarrow \exists_{c>0, n_0>0} \forall_{n>n_0}\ 0 \leqslant cg(n) \leqslant f(n)$$

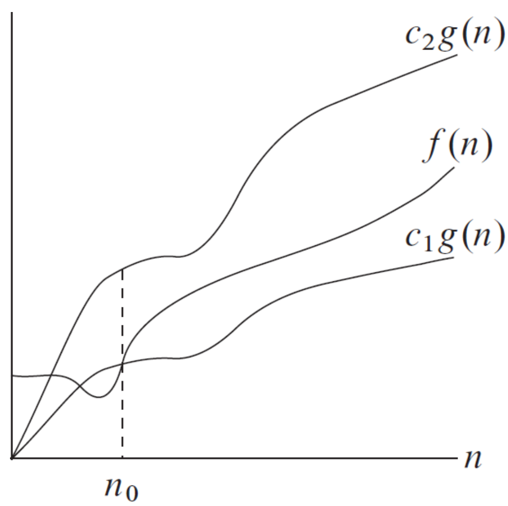


WYDZIAŁ FIZYKI I INFORMATYKI STOSOWANEJ
Uniwersytet Łódzki

Notacja Θ

Niech $f(n)$ oznacza funkcję opisującą zależność liczby operacji dominujących od rozmiaru danych wejściowych, $g(n)$ pewną funkcję referencyjną

$$f(n) = \Theta(g(n)) \Leftrightarrow \exists_{c_1>0, c_2>0, n_0>0} \forall_{n>n_0}\ 0 \leqslant c_1 g(n) \leqslant f(n) \leqslant c_2 g(n)$$



WYDZIAŁ FIZYKI I INFORMATYKI STOSOWANEJ
Uniwersytet Łódzki

Notacje asymptotyczne

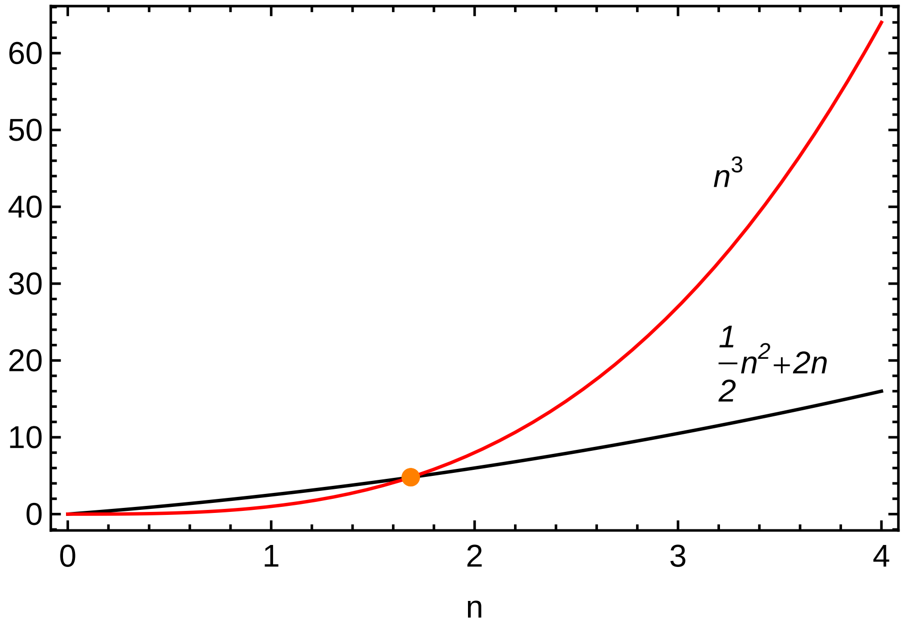
Inny sposób określania złożoności asymptotycznej:

- $$f(n) = O(g(n)) \Leftrightarrow \lim_{n\to\infty} \frac{f(n)}{g(n)} < \infty$$
- $$f(n) = \Omega(g(n)) \Leftrightarrow \lim_{n\to\infty} \frac{f(n)}{g(n)} > 0$$
- $$f(n) = \Theta(g(n)) \Leftrightarrow 0 < \lim_{n\to\infty} \frac{f(n)}{g(n)} < \infty$$

Przykład - notacja O

$$f(n) = O(g(n)) \Leftrightarrow \exists_{c>0, n_0>0} \forall_{n>n_0}\ 0 \leqslant f(n) \leqslant cg(n)$$

Niech $f(n) = \frac{1}{2}n^2 + 2n$. Czy $f(n) = \frac{1}{2}n^2 + 2n = O(n^3)$?



Dla $n > n_0 = 1$, $c = 1$ zachodzi $0 \leqslant \frac{1}{2}n^2 + 2n \leqslant cn^3$.

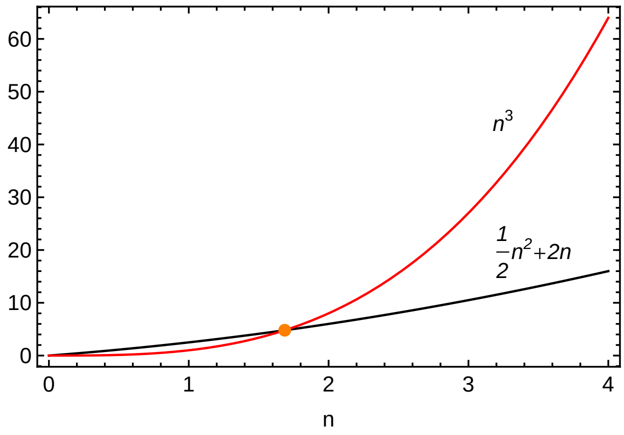
Przykład - notacja O

$$f(n) = O(g(n)) \Leftrightarrow \exists_{c>0, n_0>0} \forall_{n>n_0} \; 0 \leqslant f(n) \leqslant cg(n)$$

Niech $f(n) = \frac{1}{2}n^2 + 2n$. Czy $f(n) = \frac{1}{2}n^2 + 2n = O(n^3)$?

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \lim_{n\to\infty} \frac{\frac{1}{2}n^2 + 2n}{n^3} = \lim_{n\to\infty} \frac{\frac{1}{2n} + \frac{2}{n^2}}{1} = 0$$

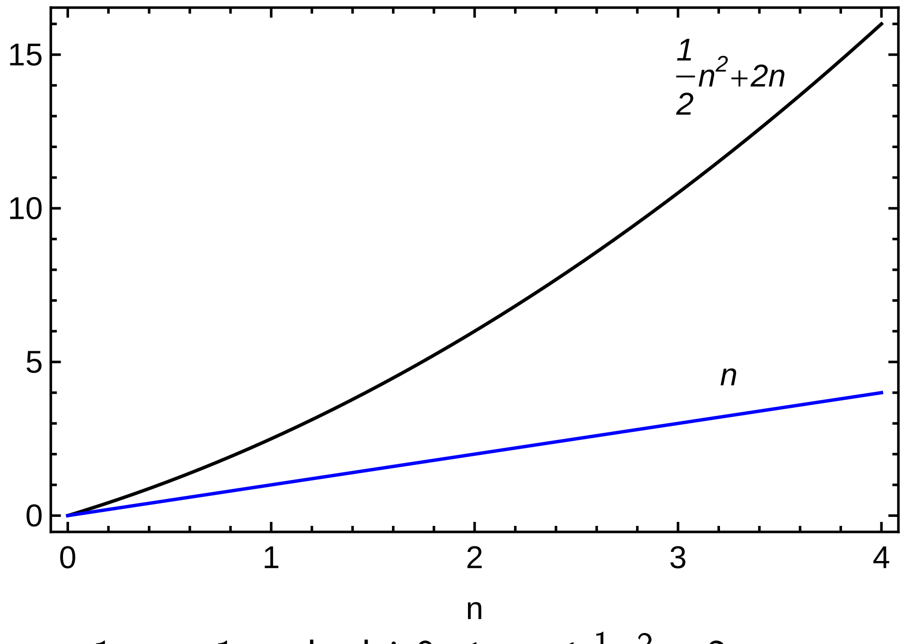
$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0 \Rightarrow f(n) = \frac{1}{2}n^2 + 2n = O(n^3)$$



Przykład - notacja Ω

$$f(n) = \Omega(g(n)) \Leftrightarrow \exists_{c>0, n_0>0} \forall_{n>n_0} \; 0 \leqslant cg(n) \leqslant f(n)$$

Niech $f(n) = \frac{1}{2}n^2 + 2n$. Czy $f(n) = \frac{1}{2}n^2 + 2n = \Omega(n)$?



Dla $n > n_0 = 1$, $c = 1$ zachodzi $0 \leqslant cn \leqslant \frac{1}{2}n^2 + 2n$.

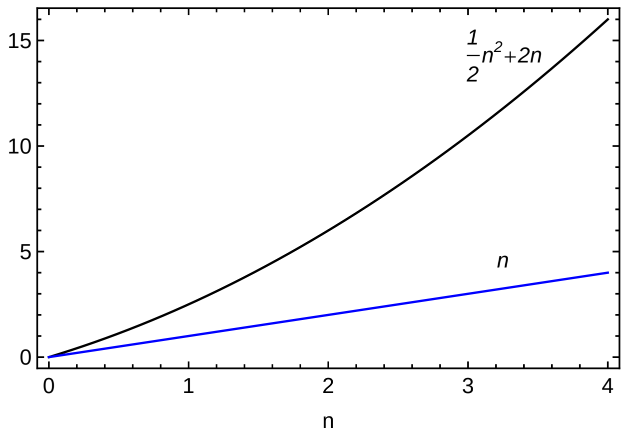
Przykład - notacja Ω

$$f(n) = \Omega(g(n)) \Leftrightarrow \exists_{c>0, n_0>0} \forall_{n>n_0}\ 0 \leqslant cg(n) \leqslant f(n)$$

Niech $f(n) = \frac{1}{2}n^2 + 2n$. Czy $f(n) = \frac{1}{2}n^2 + 2n = \Omega(n)$?

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \lim_{n\to\infty} \frac{\frac{1}{2}n^2 + 2n}{n} = \lim_{n\to\infty} \frac{\frac{1}{2}n + 2}{1} = \infty$$

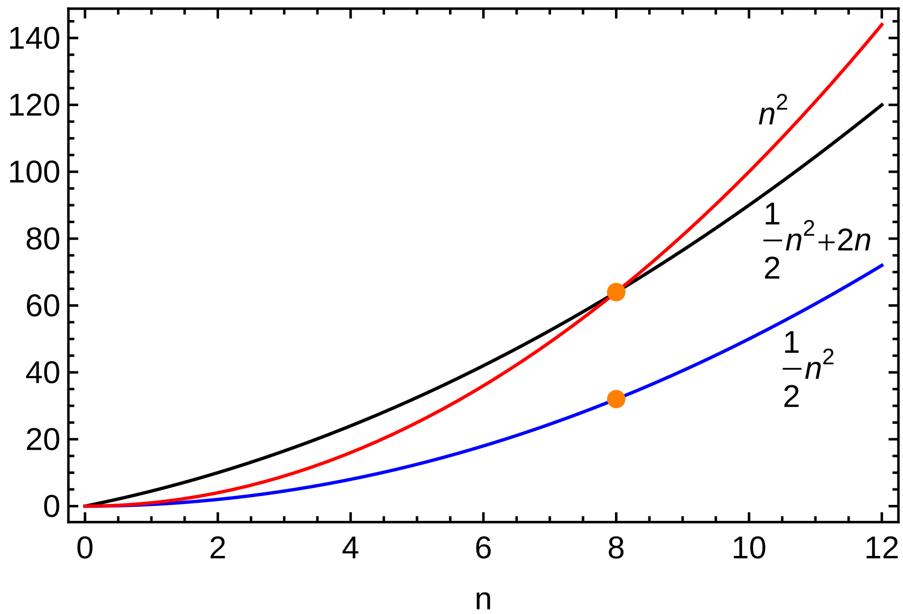
$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty \Rightarrow f(n) = \frac{1}{2}n^2 + 2n = \Omega(n)$$



Przykład - notacja Θ

$$f(n) = \Theta(g(n)) \Leftrightarrow \exists_{c_1>0, c_2>0, n_0>0} \forall_{n>n_0}\ 0 \leqslant c_1 g(n) \leqslant f(n) \leqslant c_2 g(n)$$

Niech $f(n) = \frac{1}{2}n^2 + 2n$. Czy $f(n) = \frac{1}{2}n^2 + 2n = \Theta(n^2)$?



Dla $n > n_0 = 8$, $c_1 = \frac{1}{2}$, $c_2 = 1$ zachodzi
$0 \leqslant c_1 n^2 \leqslant \frac{1}{2}n^2 + 2n \leqslant c_2 n^2$.

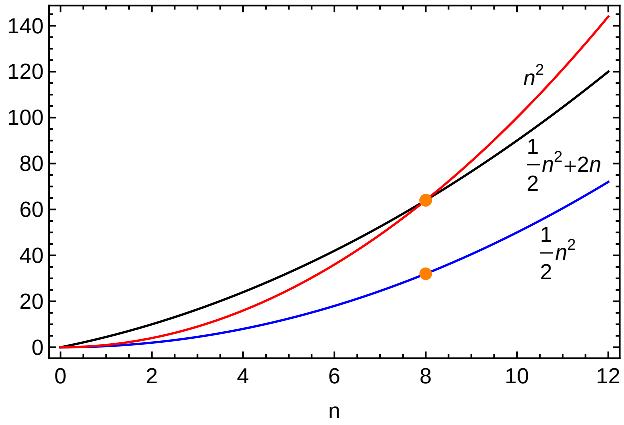
Przykład - notacja Θ

$$f(n) = \Theta(g(n)) \Leftrightarrow \exists_{c_1>0, c_2>0, n_0>0} \forall_{n>n_0} \, 0 \leqslant c_1 g(n) \leqslant f(n) \leqslant c_2 g(n)$$

Niech $f(n) = \frac{1}{2}n^2 + 2n$. Czy $f(n) = \frac{1}{2}n^2 + 2n = \Theta(n^2)$?

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \lim_{n\to\infty} \frac{\frac{1}{2}n^2 + 2n}{n^2} = \lim_{n\to\infty} \frac{\frac{1}{2} + \frac{2}{n}}{1} = \frac{1}{2}$$

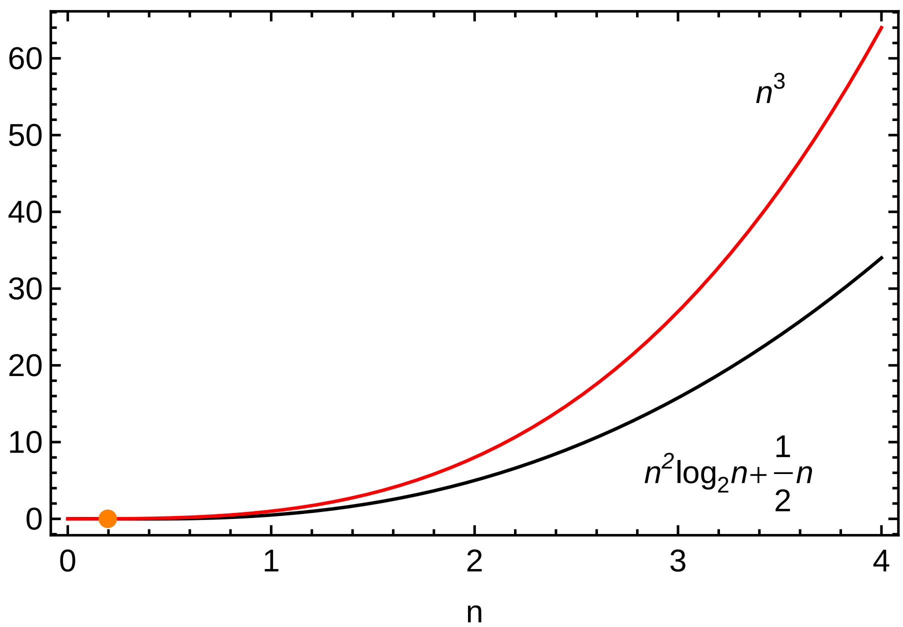
$$0 < \lim_{n\to\infty} \frac{f(n)}{g(n)} < \infty \Rightarrow f(n) = \frac{1}{2}n^2 + 2n = \Theta(n^2)$$



Przykład - notacja O

$$f(n) = O(g(n)) \Leftrightarrow \exists_{c>0, n_0>0} \forall_{n>n_0} \, 0 \leqslant f(n) \leqslant cg(n)$$

Niech $f(n) = n^2 \log_2 n + \frac{1}{2}n$. Czy $f(n) = n^2 \log_2 n + \frac{1}{2}n = O(n^3)$?

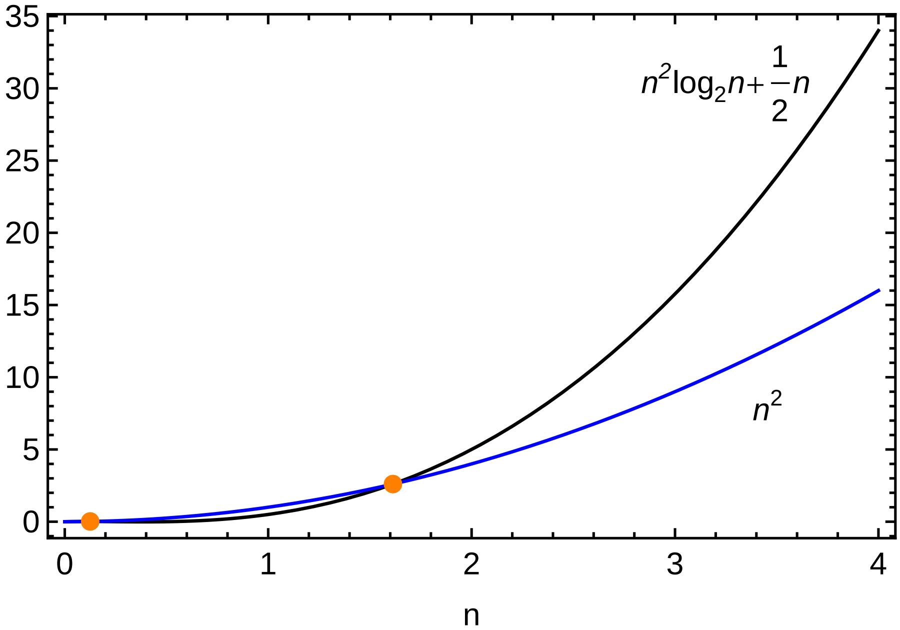


Dla $n > n_0 = 1$, $c = 1$ zachodzi $0 \leqslant n^2 \log_2 n + \frac{1}{2}n \leqslant cn^3$.

Przykład - notacja Ω

$$f(n) = \Omega(g(n)) \Leftrightarrow \exists_{c>0,n_0>0} \forall_{n>n_0}\ 0 \leqslant cg(n) \leqslant f(n)$$

Niech $f(n) = n^2 \log_2 n + \frac{1}{2}n$. Czy $f(n) = n^2 \log_2 n + \frac{1}{2}n = \Omega(n^2)$?



Dla $n > n_0 = 1$, $c = 1$ zachodzi $0 \leqslant cn^2 \leqslant n^2 \log_2 n + \frac{1}{2}n$.

Notacja Θ

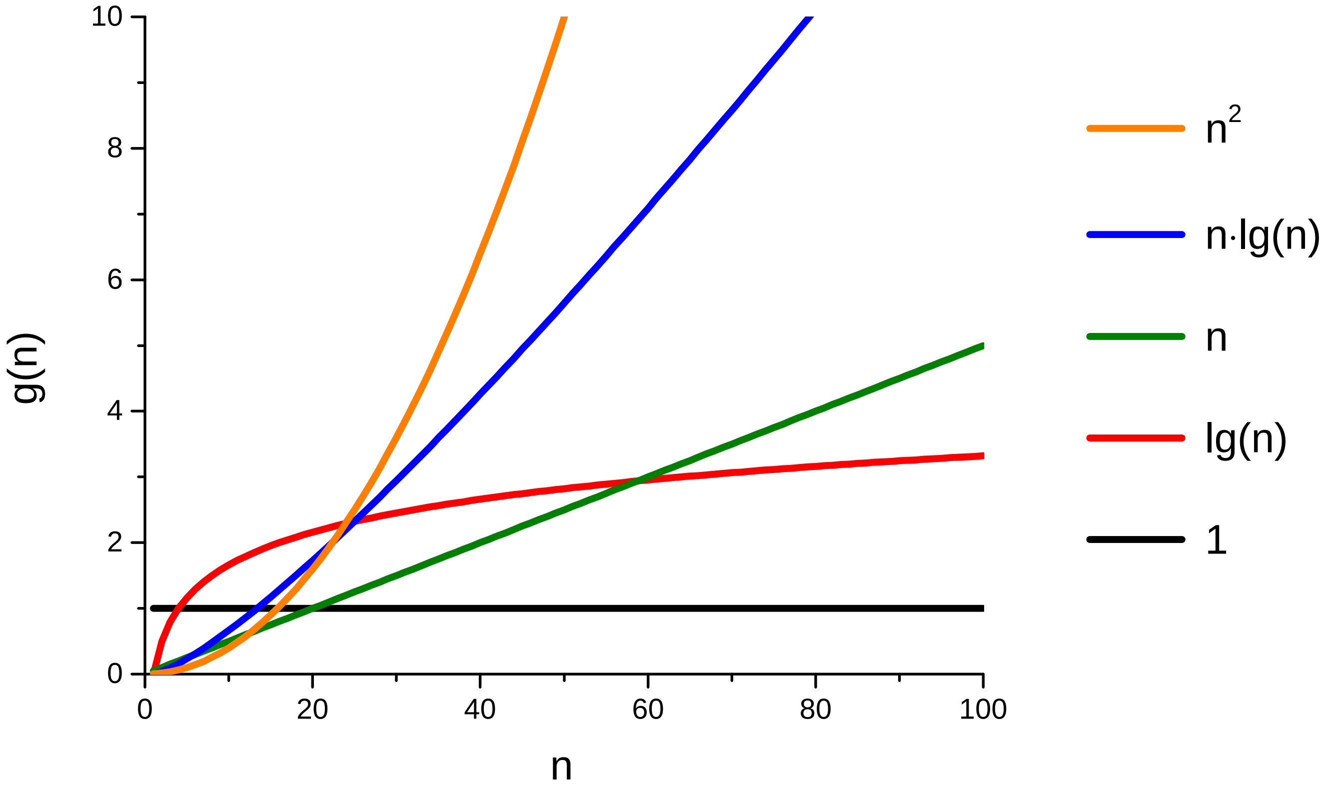
Dla dowolnych funkcji $f(n)$ i $g(n)$ zachodzi:

$$f(n) = \Theta(g(n)) \Leftrightarrow (f(n) = O(g(n)) \wedge f(n) = \Omega(g(n)))$$

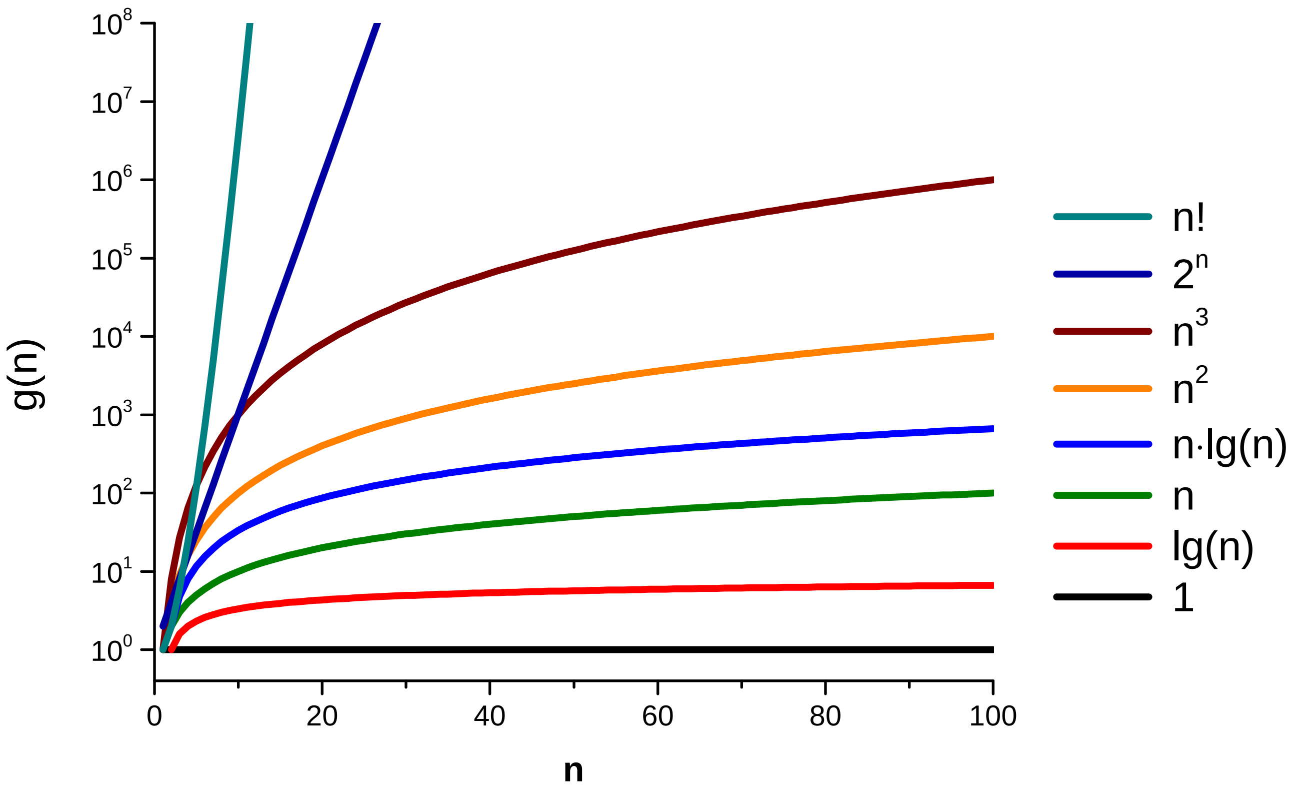
- $n^2 - 3n + 2 = \Theta(n^2)$, bo zarówno O, jak i Ω
- $n^2 - 3n + 2 \neq \Theta(n^3)$, bo tylko O
- $n^2 - 3n + 2 \neq \Theta(n)$, bo tylko Ω

WYDZIAŁ FIZYKI I INFORMATYKI STOSOWANEJ
Uniwersytet Łódzki

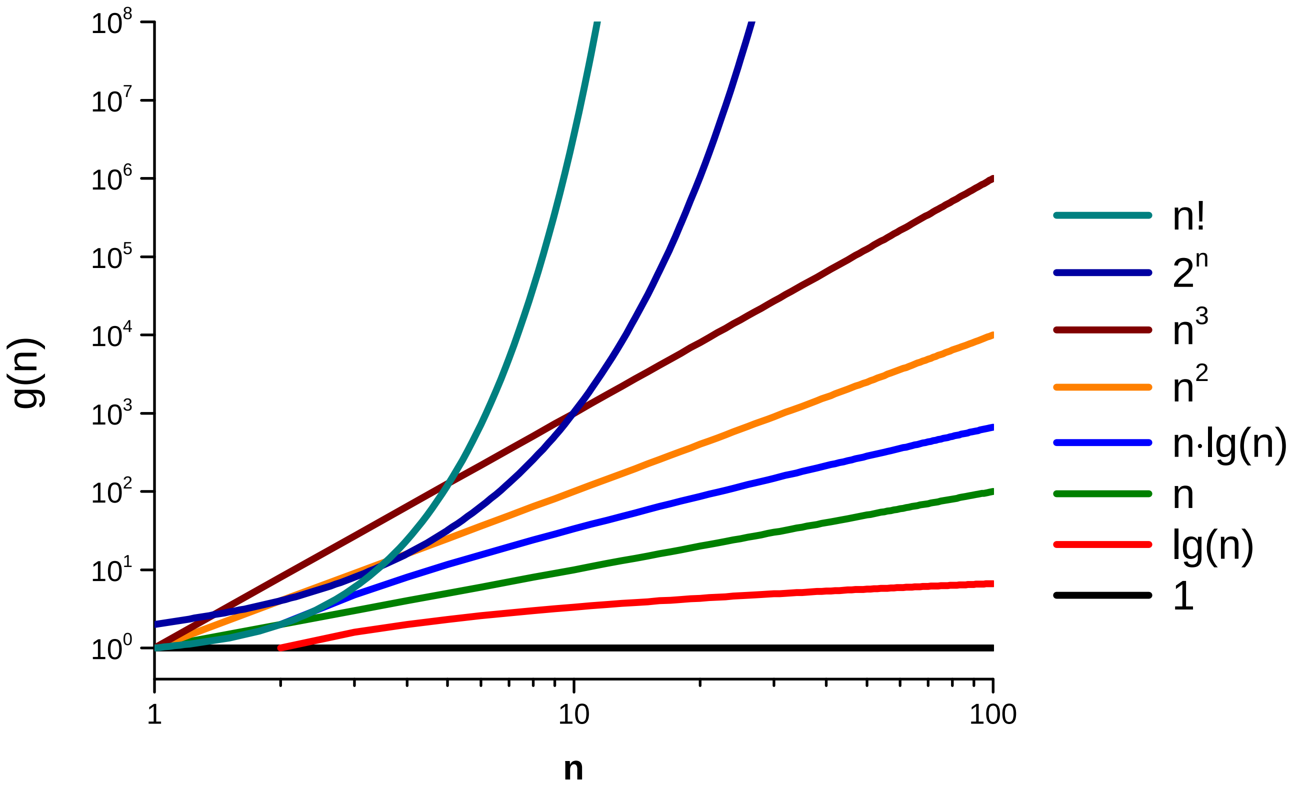
Przykłady złożoności obliczeniowej (skala liniowa)



Przykłady złożoności obliczeniowej (skala logarytmiczna)



Przykłady złożoności obliczeniowej (podwójna skala logarytmiczna)



Metody projektowania algorytmów

- Metoda zachłanna
- Metoda „dziel i zwyciężaj”
- Metoda dynamiczna

WYDZIAŁ FIZYKI I INFORMATYKI STOSOWANEJ
Uniwersytet Łódzki

Metoda zachłanna

- Stosowana przy algorytmach optymalizacji.
- Algorytm zachłanny (*greedy algorithm*) w każdym kroku dokonuje zachłannego (lokalnie optymalnego) wyboru rozwiązania częściowego.

- Lokalnie optymalny wybór $\Rightarrow$ globalnie optymalne rozwiązanie?

Przykład: algorytm zachłanny wydawania reszty (należy wydać konkretną kwotę przy wykorzystaniu jak najmniejszej liczby monet/banknotów o danych nominałach):

- $91 = 50 + 20 + 20 + 1$, nominały $50, 20, 10, 5, 2, 1$ (rozwiązanie optymalnie)
- $62 = 50 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1$, nominały $50, 20, 1$ (optymalne rozwiązanie: $20 + 20 + 20 + 1$)

WYDZIAŁ FIZYKI I INFORMATYKI STOSOWANEJ Uniwersytet Łódzki

Metoda „dziel i zwyciężaj”

- **Struktura rekurencyjna**
- Problem jest dzielony na podproblemy o mniejszym rozmiarze podobne do początkowego problemu.
- Podproblemy są rozwiązywane rekurencyjnie
- Rozwiązania podproblemów są łączone w celu uzyskania rozwiązania całego problemu
- Podproblemy są od siebie niezależne.
  - Dziel: dzielimy problem na podproblemy
  - Zwyciężaj: rozwiązujemy podproblemy rekurencyjnie (chyba że rozmiar problemu jest już tak mały, że można go rozwiązać bezpośrednio-inaczej rekurencja nie zatrzyma się)
  - Połącz: łączymy rozwiązania podproblemów w rozwiązanie problemu
- Metoda zstępująca (top-down).

WYDZIAŁ FIZYKI I INFORMATYKI STOSOWANEJ Uniwersytet Łódzki

Metoda dynamiczna

- Problem jest dzielony na podproblemy o mniejszym rozmiarze podobne do początkowego problemu.
- Podproblemy nie są od siebie niezależne.
- Podejście "dziel i zwyciężaj" powodowałoby wielokrotne niezależne rozwiązywanie tych samych problemów, co byłoby nieefektywne.
- Zamiast tego każdy z podproblemów jest rozwiązywany jednokrotnie, a rozwiązanie jest zapamiętywane w tablicy.
- Metoda wstępująca (bottom-up).

"Dziel i zwyciężaj"

- Obliczanie wartości symbolu Newtona:

$$C(n,m) = \binom{n}{m} = \frac{n!}{m!\,(n-m)!}$$

$$C(n,m) = C(n-1,m) + C(n-1,m-1) \qquad C(n,n) = 1 \qquad C(n,0) = 1$$

BinomCoeff(n,m)

Wejście: $n, m;\ 0 \leqslant m \leqslant n$
Wyjście: $C(n,m)$

```
if m = n or m = 0 then
    return 1
else
    return BinomCoeff(n − 1, m) + BinomCoeff(n − 1, m − 1)
end if
```

"Dziel i zwyciężaj"

- Obliczanie wartości symbolu Newtona:

$$C(n,m) = \binom{n}{m} = \frac{n!}{m!\,(n-m)!}$$

$$C(n,m) = C(n-1,m) + C(n-1,m-1) \qquad C(n,n) = 1 \qquad C(n,0) = 1$$

$$C(5,2) = C(4,2) + C(4,1)$$

$$C(4,2) = C(3,2) + C(3,1)$$

$$C(3,2) = \underbrace{C(2,2)}_{=1} + C(2,1)$$

$$C(2,1) = \underbrace{C(1,1)}_{=1} + \underbrace{C(1,0)}_{=1}$$

$$C(3,1) = \mathbf{C(2,1)} + \underbrace{C(2,0)}_{=1}$$

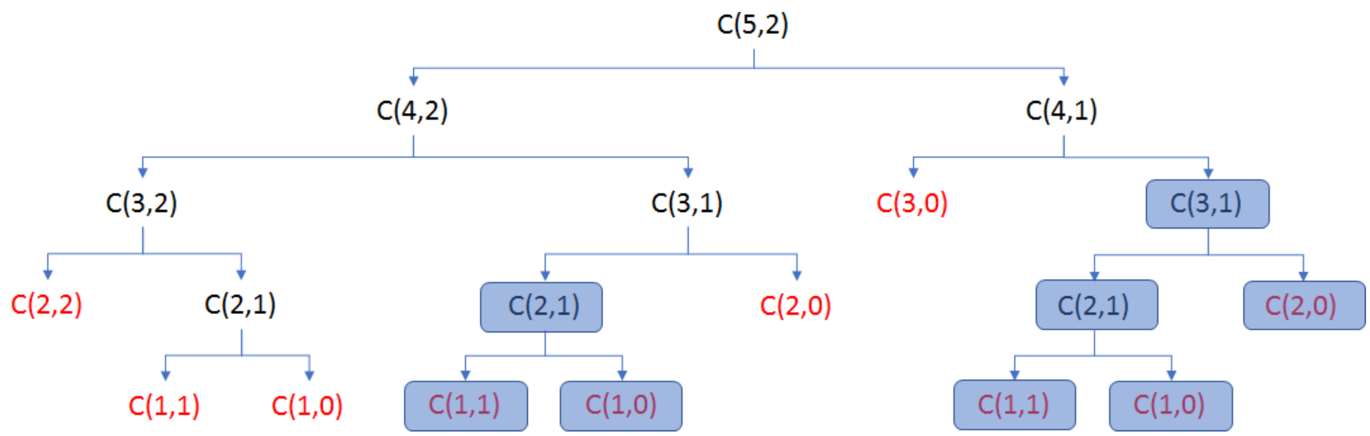
$$C(4,1) = \mathbf{C(3,1)} + \underbrace{C(3,0)}_{=1}$$

"Dziel i zwyciężaj"

- Obliczanie wartości symbolu Newtona:

$$C(n,m) = \binom{n}{m} = \frac{n!}{m!\,(n-m)!}$$

$$C(n,m) = C(n-1,m) + C(n-1,m-1) \qquad C(n,n) = 1 \qquad C(n,0) = 1$$



- $2\binom{n}{m} - 1$ wywołań rekurencyjnych

"Dziel i zwyciężaj" $\longrightarrow$ programowanie dynamiczne

- Obliczanie wartości symbolu Newtona:

$$C(n,m) = \binom{n}{m} = \frac{n!}{m!\,(n-m)!}$$

BinomCoeff(n,m)

Wejście: $n, m;\ 0 \leqslant m \leqslant n$
Wyjście: $C(n,m)$

for $i = 0$ to n **do**
  for $j = 0$ to $\min(i, m)$ **do**
    if $j = 0$ or $j = i$ **then**
      $B[i][j] \leftarrow 1$
    else
      $B[i][j] \leftarrow B[i-1][j] + B[i-1][j-1]$
    end if
  end for
end for
return $B[n][m]$

"Dziel i zwyciężaj" $\longrightarrow$ programowanie dynamiczne

- Obliczanie wartości symbolu Newtona:

$$C(n,m) = \binom{n}{m} = \frac{n!}{m!\,(n-m)!}$$

C(5,2)

j

i

B[0][0]=1		
B[1][0]=1	B[1][1]=1	
B[2][0]=1	B[2][1]=2	B[2][2]=1
B[3][0]=1	B[3][1]=3	B[3][2]=3
B[4][0]=1	B[4][1]=4	B[4][2]=6
B[5][0]=1	B[5][1]=5	**B[5][2]=10**

BinomCoeff(n,m)

Wejście: $n, m;\ 0 \leqslant m \leqslant n$
Wyjście: $C(n,m)$

for $i = 0$ to n **do**
  for $j = 0$ to $\min(i, m)$ **do**
    if $j = 0$ or $j = i$ **then**
      $B[i][j] \leftarrow 1$
    else
      $B[i][j] \leftarrow B[i-1][j] + B[i-1][j-1]$
    end if
  end for
end for
return $B[n][m]$

WYDZIAŁ FIZYKI I INFORMATYKI STOSOWANEJ
Uniwersytet Łódzki

"Dziel i zwyciężaj" → programowanie dynamiczne

- Obliczanie wartości symbolu Newtona:

$$C(n,m) = \binom{n}{m} = \frac{n!}{m!\,(n-m)!}$$

BinomCoeff(n,m)

Wejście: $n, m;\ 0 \leqslant m \leqslant n$
Wyjście: $C(n,m)$

```
B[0] ← 1
for i = 1 to n do
    if i ≤ m then
        B[i] ← 1
    end if
    for j = min(i − 1, m) downto 1 do
        B[j] ← B[j] + B[j − 1]
    end for
end for
return B[m]
```


"Dziel i zwyciężaj" → programowanie dynamiczne

- Obliczanie wartości symbolu Newtona:

$$C(n,m) = \binom{n}{m} = \frac{n!}{m!\,(n-m)!}$$

C(5,2)

j

i

B[0]=1		
B[0]=1	B[1]=1	
B[0]=1	B[1]=2	B[2]=1
B[0]=1	B[1]=3	B[2]=3
B[0]=1	B[1]=4	B[2]=6
B[0]=1	B[1]=5	**B[2]=10**

BinomCoeff(n,m)

Wejście: $n, m;\ 0 \leqslant m \leqslant n$
Wyjście: $C(n,m)$

```
B[0] ← 1
for i = 1 to n do
    if i ≤ m then
        B[i] ← 1
    end if
    for j = min(i − 1, m) downto 1 do
        B[j] ← B[j] + B[j − 1]
    end for
end for
return B[m]
```

"Dziel i zwyciężaj" $\rightarrow$ programowanie dynamiczne

- Obliczanie wartości symbolu Newtona:

$$C(n,m) = \binom{n}{m} = \frac{n!}{m!\,(n-m)!}$$

C(5,2)

j

i

B[0][0]=1		
B[1][0]=1	B[1][1]=1	
B[2][0]=1	B[2][1]=2	B[2][2]=1
B[3][0]=1	B[3][1]=3	B[3][2]=3
B[4][0]=1	B[4][1]=4	B[4][2]=6
B[5][0]=1	B[5][1]=5	**B[5][2]=10**

C(5,2)

j

i

B[0]=1		
B[0]=1	B[1]=1	
B[0]=1	B[1]=2	B[2]=1
B[0]=1	B[1]=3	B[2]=3
B[0]=1	B[1]=4	B[2]=6
B[0]=1	B[1]=5	**B[2]=10**

Metody analizy algorytmów

$T(n)$ - czas potrzebny dla wykonania pewnego algorytmu

$T(n) \propto$ liczba operacji dominujących

Analiza równań rekurencyjnych:

- Metoda podstawień (Substitution)
- Metoda drzew rekurencyjnych (Recurrence-tree)
- Metoda rekurencji uniwersalnej (Master method)

WYDZIAŁ FIZYKI
I INFORMATYKI STOSOWANEJ
Uniwersytet Łódzki

Metoda podstawień

Przykład 1:

$$T(n) = \begin{cases} 1, & n = 1 \\ T(n-1) + n, & n > 1 \end{cases}$$

- Zgadujemy: $$T(n) = O\left(n^2\right)$$

- Czy $T(n) \leqslant cn^2$ dla pewnego $c > 0$ oraz $n \geqslant n_0$?
- Załóżmy, że jest to spełnione dla $T(n-1)$:

$T(n-1) \leqslant c(n-1)^2$.

$$\begin{aligned} T(n) &= T(n-1) + n \leqslant c(n-1)^2 + n = cn^2 - 2cn + c + n = \\ &= cn^2 + c(1-2n) + n \end{aligned}$$

$cn^2 + c(1-2n) + n \leqslant cn^2$ jeżeli
$c(1-2n) + n \leqslant 0 \Rightarrow c \geqslant \frac{n}{2n-1} = \frac{1}{2-\frac{1}{n}}$, co zachodzi dla $n \geqslant 1$ i $c \geqslant 1$

- Dla $n = 1$: $T(1) = 1 \leqslant c \cdot 1^2 = c$. Można wybrać dowolne $c \geqslant 1$.

WYDZIAŁ FIZYKI I INFORMATYKI STOSOWANEJ
Uniwersytet Łódzki