Metody numeryczne

EiT 2011/2012

Sprawy organizacyjne

- dr Wojciech Tylman, Katedra Mikroelektroniki i Technik Informatycznych PŁ
- B 18, lp., p. 56
- www.dmcs.p.lodz.pl
- tyl@dmcs.p.lodz.pl
- godziny przyjęć: WWW

Tematyka

- Metody rozwiązywania równań przeznaczone do implementacji komputerowej
- Zagadnienia związane z obliczeniami numerycznymi przeprowadzanymi na współczesnych komputerach

Literatura

- Leon O. Chua , Pen-Min Lin “Komputerowa analiza układów elektronicznych : algorytmy i metody obliczeniowe”, Wydawnictwa Naukowo-Techniczne, Warszawa 1981
- Zenon Fortuna, Bohdan Macukow, Janusz Wąsowski “Metody numeryczne”, Wydawnictwa Naukowo-Techniczne, Warszawa 2006

Organizacja zajęć

- Wykład: 15h (2h co tydzień, pół semestru)
- Ćwiczenia: 15h (1h co tydzień)
- Laboratorium: 15h (2h co drugi tydzień)

- Zaliczenie przedmiotu: zaliczenie wykładu, ćwiczeń i laboratorium, oceny brane z tą samą wagą

Plan

- Metody numeryczne rozwiązywania liniowych układów równań (2h)
- Metody numeryczne rozwiązywania równań nieliniowych i nieliniowych układów równań (2h)
- Komputerowe opracowywanie wyników pomiarów (interpolacja, aproksymacja) (1h)
- Algorytmy przetwarzania sygnałów (1h)
- Ograniczenia i korzyści symulacji komputerowej (1/2h)
- Symulacja i eksperyment komputerowy (1/2h)
- Oprogramowanie do obliczeń i symulacji inżynierskich (4h)
- Zasady tworzenia skryptów do narzędzi programowych (2h)
- Dokumentacja inżynierska (1h)
- Zaliczenie (1h)

Metody numeryczne rozwiązywania liniowych układów równań

- Metoda eliminacji Gaussa
- Metoda rozkładu LU
  - algorytm Crouta
  - algorytm Doolittla

Metoda eliminacji Gaussa

- Efektywna metoda rozwiązywania układów równań liniowych
- Wymaga w przybliżeniu $n^3/3$ mnożeń. Np. dla n=10 daje to 333 mnożeń
- Dla porównania, metoda Cramera wymaga w przybliżeniu $2(n + 1)!$ mnożeń. Dla n=10 daje to 79 833 600 mnożeń

Metoda eliminacji Gaussa

$$\mathbf{Ax} = \boldsymbol{\mu}$$

$$\begin{aligned}
a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n &= \mu_1 \\
a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n &= \mu_2 \\
&\vdots \\
a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n &= \mu_n
\end{aligned}$$

1. **Eliminacja w przód**
2. **Podstawienie wstecz**

Etap I: eliminacja w przód

$$a_{11}x_1 + a_{12}x_2 + a_{13}x_3 = \mu_1 \quad \left| \times \frac{-a_{21}}{a_{11}} + \right. \quad \left| \times \frac{-a_{31}}{a_{11}} + \right.$$

$$a_{21}x_1 + a_{22}x_2 + a_{23}x_3 = \mu_2$$

$$a_{31}x_1 + a_{32}x_2 + a_{33}x_3 = \mu_3$$

$$a_{11}x_1 + a_{12}x_2 + a_{13}x_3 = \mu_1$$

$$a_{22}^{(2)}x_2 + a_{23}^{(2)}x_3 = \mu_2^{(2)} \quad \left| \times \frac{-a_{32}^{(2)}}{a_{22}^{(2)}} + \right.$$

$$a_{32}^{(2)}x_2 + a_{33}^{(2)}x_3 = \mu_3^{(2)}$$

$$
\begin{aligned}
a_{11}x_1 + a_{12}x_2 + a_{13}x_3 &= \mu_1 \\
a_{22}^{(2)}x_2 + a_{23}^{(2)}x_3 &= \mu_2^{(2)} \\
a_{33}^{(3)}x_3 &= \mu_3^{(3)}
\end{aligned}
$$

$$
\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ 0 & a_{22}^{(2)} & a_{23}^{(2)} \\ 0 & 0 & a_{33}^{(3)} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} \mu_1 \\ \mu_2^{(2)} \\ \mu_3^{(3)} \end{bmatrix}
$$

Etap II: podstawienie wstecz

$$x_3 = \frac{\mu_3^{(3)}}{a_{33}^{(3)}}$$

$$\vdots$$

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} \mu_1^{(6)} \\ \mu_2^{(5)} \\ \mu_3^{(4)} \end{bmatrix}$$

$$\mathbf{A_r} = \left[\mathbf{A} \,\vdots\, \boldsymbol{\mu}\right] \quad \Rightarrow \quad \left[\mathbf{1} \,\vdots\, \hat{\boldsymbol{\mu}}\right]$$

Metoda eliminacji Gaussa - przykład (1)

$$\begin{bmatrix} 2 & 6 & 1 \\ 1 & 2 & 2 \\ 4 & 6 & 8 \end{bmatrix} \cdot x = \begin{bmatrix} 13 \\ 3 \\ 8 \end{bmatrix}$$

Metoda eliminacji Gaussa - przykład (2)

$$\begin{bmatrix} 2 & 6 & 1 & 13 \\ 1 & 2 & 2 & 3 \\ 4 & 6 & 8 & 8 \end{bmatrix}$$

Metoda eliminacji Gaussa - przykład (3)

$$\begin{bmatrix} 2 & 6 & 1 & 13 \\ 0 & -1 & 3/2 & -7/2 \\ 0 & -6 & 6 & -18 \end{bmatrix}$$

Metoda eliminacji Gaussa - przykład (4)

$$\begin{bmatrix} 2 & 6 & 1 & 13 \\ 0 & -1 & 3/2 & -7/2 \\ 0 & 0 & -3 & 3 \end{bmatrix}$$

Metoda eliminacji Gaussa - przykład (5)

$$\begin{bmatrix} 2 & 6 & 1 & 13 \\ 0 & -1 & 3/2 & -7/2 \\ 0 & 0 & 1 & -1 \end{bmatrix}$$

Metoda eliminacji Gaussa - przykład (6)

$$\begin{bmatrix} 2 & 6 & 1 & 13 \\ 0 & -1 & 0 & -2 \\ 0 & 0 & 1 & -1 \end{bmatrix}$$

Metoda eliminacji Gaussa - przykład (7)

$$\begin{bmatrix} 2 & 6 & 1 & 13 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & -1 \end{bmatrix}$$

Metoda eliminacji Gaussa - przykład (8)

$$\begin{bmatrix} 2 & 0 & 0 & 2 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & -1 \end{bmatrix}$$

Metoda eliminacji Gaussa - przykład (9)

$$\begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & -1 \end{bmatrix}$$

Metoda eliminacji Gaussa - przykład (10)

$$x = \begin{bmatrix} 1 \\ 2 \\ -1 \end{bmatrix}$$

Metoda eliminacji Gaussa

- Podczas wyznaczania rozwiązania nie jest wyznaczana macierz odwrotna
- Jeśli zachodzi potrzeba wyznaczenia macierzy odwrotnej, można to zrobić za pomocą eliminacji Gaussa poprzez dopisanie do macierzy A macierzy jednostkowej i wykonanie algorytmu jak poprzednio.
- Wymaga to n^3 mnożeń

Metoda eliminacji Gaussa

- Jeśli podczas wykonywania algorytmu okaże się, że element przez którego odwrotność należy mnożyć (czyli dzielić przez) ma wartość 0 (lub nawet wartość bezwzględną bardzo małą w porównaniu z pozostałymi) należy przeprowadzić zamianę wierszy tak, aby uzyskać element o dużej wartości. Aby zapewnić dużą dokładność, należy zawsze zamieniać wiersze (można również kolumny) przed każdym mnożeniem.

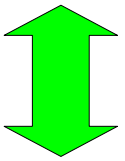
Metoda eliminacji Gaussa - przykład (1)

$$\begin{bmatrix} 2 & 6 & 1 \\ 1 & 3 & 2 \\ 4 & 6 & 8 \end{bmatrix} \cdot x = \begin{bmatrix} 11 \\ -2 \\ -14 \end{bmatrix}$$

$$\left[\begin{array}{ccc|c} 2 & 6 & 1 & 11 \\ 1 & 3 & 2 & -2 \\ 4 & 6 & 8 & -14 \end{array}\right]$$

Metoda eliminacji Gaussa - przykład (2)

$$\begin{bmatrix} 2 & 6 & 1 & 11 \\ 0 & 0 & 1.5 & -7.5 \\ 0 & -6 & 6 & -36 \end{bmatrix}$$



$$\begin{bmatrix} 2 & 6 & 1 & 11 \\ 0 & -6 & 6 & -36 \\ 0 & 0 & 1.5 & -7.5 \end{bmatrix}$$

Rozkład LU

$$\mathbf{A} = \mathbf{LU}$$

$$\mathbf{U} = \begin{bmatrix} 1 & & \mathbf{u_{ij}} \\ & 1 & \\ & & \ddots \\ & \mathbf{0} & & 1 \end{bmatrix}$$

Macierz trójkątna górna

$$\mathbf{L} = \begin{bmatrix} & \ddots & \mathbf{0} \\ & & \ddots \\ & & & \ddots \\ & \mathbf{l_{ij}} & \end{bmatrix}$$

Macierz trójkątna dolna

Zastosowanie rozkładu LU

$$\mathbf{Ax} = \mathbf{LUx} = \boldsymbol{\mu}$$

$$\Downarrow$$

$$\mathbf{Ux} = \mathbf{y}$$

$$\mathbf{Ly} = \boldsymbol{\mu} \quad \textbf{(a)}$$

$$\mathbf{Ux} = \mathbf{y} \quad \textbf{(b)}$$

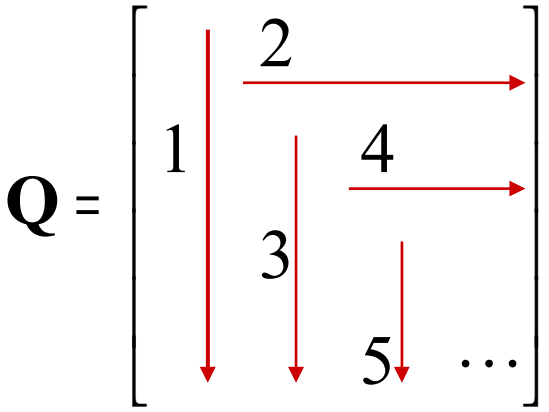
Algorytm Crouta

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = \begin{bmatrix} l_{11} & 0 & 0 \\ l_{21} & l_{22} & 0 \\ l_{31} & l_{32} & l_{33} \end{bmatrix} \begin{bmatrix} 1 & u_{12} & u_{13} \\ 0 & 1 & u_{23} \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{Q} = \mathbf{L} + \mathbf{U} - \mathbf{1}$$

$$\mathbf{Q} = \begin{bmatrix} l_{11} & u_{12} & u_{13} \\ l_{21} & l_{22} & u_{23} \\ l_{31} & l_{32} & l_{33} \end{bmatrix}$$

Algorytm Crouta



$$u_{ij} = \left(a_{ij} - \sum_{k=1}^{i-1} l_{ik} u_{kj} \right) \Big/ l_{ii} \qquad dla \quad i < j$$

$$l_{ij} = a_{ij} - \sum_{k=1}^{j-1} l_{ik} u_{kj} \qquad dla \quad i \geq j$$

Rozkład LU - przykład

$$A = \begin{bmatrix} 2 & 6 & 1 \\ 1 & 2 & 2 \\ 4 & 6 & 8 \end{bmatrix}$$

$$Q = \begin{bmatrix} 2 & 6 & 1 \\ 1 & 2 & 2 \\ 4 & 6 & 8 \end{bmatrix}$$

Rozkład LU - przykład

$$Q = \begin{bmatrix} 2 & 3 & 1/2 \\ 1 & 2 & 2 \\ 4 & 6 & 8 \end{bmatrix}$$

Rozkład LU - przykład

$$Q = \begin{bmatrix} 2 & 3 & 1/2 \\ 1 & -1 & 2 \\ 4 & -6 & 8 \end{bmatrix}$$

Rozkład LU - przykład

$$Q = \begin{bmatrix} 2 & 3 & 1/2 \\ 1 & -1 & -3/2 \\ 4 & -6 & 8 \end{bmatrix}$$

Rozkład LU - przykład

$$Q = \begin{bmatrix} 2 & 3 & 1/2 \\ 1 & -1 & -3/2 \\ 4 & -6 & -3 \end{bmatrix}$$

Rozkład LU - przykład

$$L = \begin{bmatrix} 2 & 0 & 0 \\ 1 & -1 & 0 \\ 4 & -6 & -3 \end{bmatrix}$$

Rozkład LU - przykład

$$U = \begin{bmatrix} 1 & 3 & 1/2 \\ 0 & 1 & -3/2 \\ 0 & 0 & 1 \end{bmatrix}$$

Rozkład LU - przykład

$$\begin{bmatrix} 2 & 0 & 0 \\ 1 & -1 & 0 \\ 4 & -6 & -3 \end{bmatrix} \cdot y = \begin{bmatrix} 13 \\ 3 \\ 8 \end{bmatrix}$$

Rozkład LU - przykład

$$\begin{bmatrix} 2 & 0 & 0 \\ 1 & -1 & 0 \\ 4 & -6 & -3 \end{bmatrix} \cdot y = \begin{bmatrix} 13 \\ 3 \\ 8 \end{bmatrix}$$

$$\left[\begin{array}{ccc|c} 2 & 0 & 0 & 13 \\ 1 & -1 & 0 & 3 \\ 4 & -6 & -3 & 8 \end{array}\right]$$

Rozkład LU - przykład

$$\begin{bmatrix} 1 & 0 & 0 & 13/2 \\ 1 & -1 & 0 & 3 \\ 4 & -6 & -3 & 8 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & 0 & 13/2 \\ 0 & -1 & 0 & -7/2 \\ 0 & -6 & -3 & -18 \end{bmatrix}$$

Rozkład LU - przykład

$$\begin{bmatrix} 1 & 0 & 0 & 13/2 \\ 0 & 1 & 0 & 7/2 \\ 0 & -6 & -3 & -18 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & 0 & 13/2 \\ 0 & 1 & 0 & 7/2 \\ 0 & 0 & -3 & 3 \end{bmatrix}$$

Rozkład LU - przykład

$$\begin{bmatrix} 1 & 0 & 0 & 13/2 \\ 0 & 1 & 0 & 7/2 \\ 0 & 0 & 1 & -1 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 3 & 1/2 \\ 0 & 1 & -3/2 \\ 0 & 0 & 1 \end{bmatrix} \cdot x = \begin{bmatrix} 13/2 \\ 7/2 \\ -1 \end{bmatrix}$$

Rozkład LU - przykład

$$\begin{bmatrix} 1 & 3 & 1/2 & 13/2 \\ 0 & 1 & -3/2 & 7/2 \\ 0 & 0 & 1 & -1 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 3 & 0 & 7 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & -1 \end{bmatrix}$$

Rozkład LU - przykład

$$\begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & -1 \end{bmatrix}$$

$$x = \begin{bmatrix} 1 \\ 2 \\ -1 \end{bmatrix}$$

Algorytm Doolittle'a rozkładu LU

Dana jest macierz A. Wyznaczamy Q:

1. **Przepisz do Q pierwszą kolumnę z A**
2. **W 1-szym wierszu dziel el. niediagonalne przez diag.**
3. **Od elementu o wsk. (*i,j*) {dla *i*>1, *j*>1} odejmij iloczyn elementów o wsk. (*i*,1) i (1,*j*)**
7. **Jeżeli *liczba_kolumn(wierszy)*>=2 oznacz tę podmacierz jako A, idź do 1**
8. **STOP**

Algorytm Doolittle'a rozkładu LU - przykład

$$\mathbf{A} = \begin{bmatrix} 2 & 2 & -4 \\ 4 & 6 & -12 \\ 4 & 3 & 2 \end{bmatrix} \overset{kr.1 i 2}{\Rightarrow} \begin{bmatrix} 2 & 1 & -2 \\ 4 & 6 & -12 \\ 4 & 3 & 2 \end{bmatrix} \overset{kr.3}{\Rightarrow} \begin{bmatrix} 2 & 1 & -2 \\ 4 & 6-4\cdot 1 & -12+4\cdot 2 \\ 4 & 3-4\cdot 1 & 2+4\cdot 2 \end{bmatrix} = \begin{bmatrix} 2 & 1 & -2 \\ 4 & 2 & -4 \\ 4 & -1 & 10 \end{bmatrix}$$

$$\overset{kr.1 i 2}{\Rightarrow} \begin{bmatrix} 2 & 1 & -2 \\ 4 & 2 & -2 \\ 4 & -1 & 10 \end{bmatrix} \overset{kr.3}{\Rightarrow} \begin{bmatrix} 2 & 1 & -2 \\ 4 & 2 & -2 \\ 4 & -1 & 10-2\cdot 1 \end{bmatrix} = \begin{bmatrix} 2 & 1 & -2 \\ 4 & 2 & -2 \\ 4 & -1 & 8 \end{bmatrix} = \mathbf{Q}$$

Metoda iteracji prostej (metoda punktu stałego)

Dla równania $x = 4 - 2x^{\frac{1}{3}} = F(x)$ szukamy $x = x^*$ spełniającego to równanie ($x^*=1,641$)

Ogólna postać: $F(x) = x$

Przyjmując np. $x_0=10$ mamy $F(x_0)= -0,3089 \neq x_0$, ale jest to bliżej szukanego rozwiązania. Przyjmując $x_1 = F(x_0)$ mamy $F(x_1)=5,3519=x_2$. Dalej $F(x_2)=0,5016=x_3$ itd.:

$x_4=2,4109$	$x_8=1,6785$	$x_{12}=1,6430$	$x_{16}=1,6411$
$x_5=1,3182$	$x_9=1,6231$	$x_{13}=1,6400$	$x_{17}=1,6409$
$x_6=1,8071$	$x_{10}=1,6496$	$x_{14}=1,6414$	$x_{18}=1,6410$
$x_7=1,5639$	$x_{11}=1,6369$	$x_{15}=1,6408$	$x_{19}=1,6410$

Metoda iteracji prostej (metoda punktu stałego)

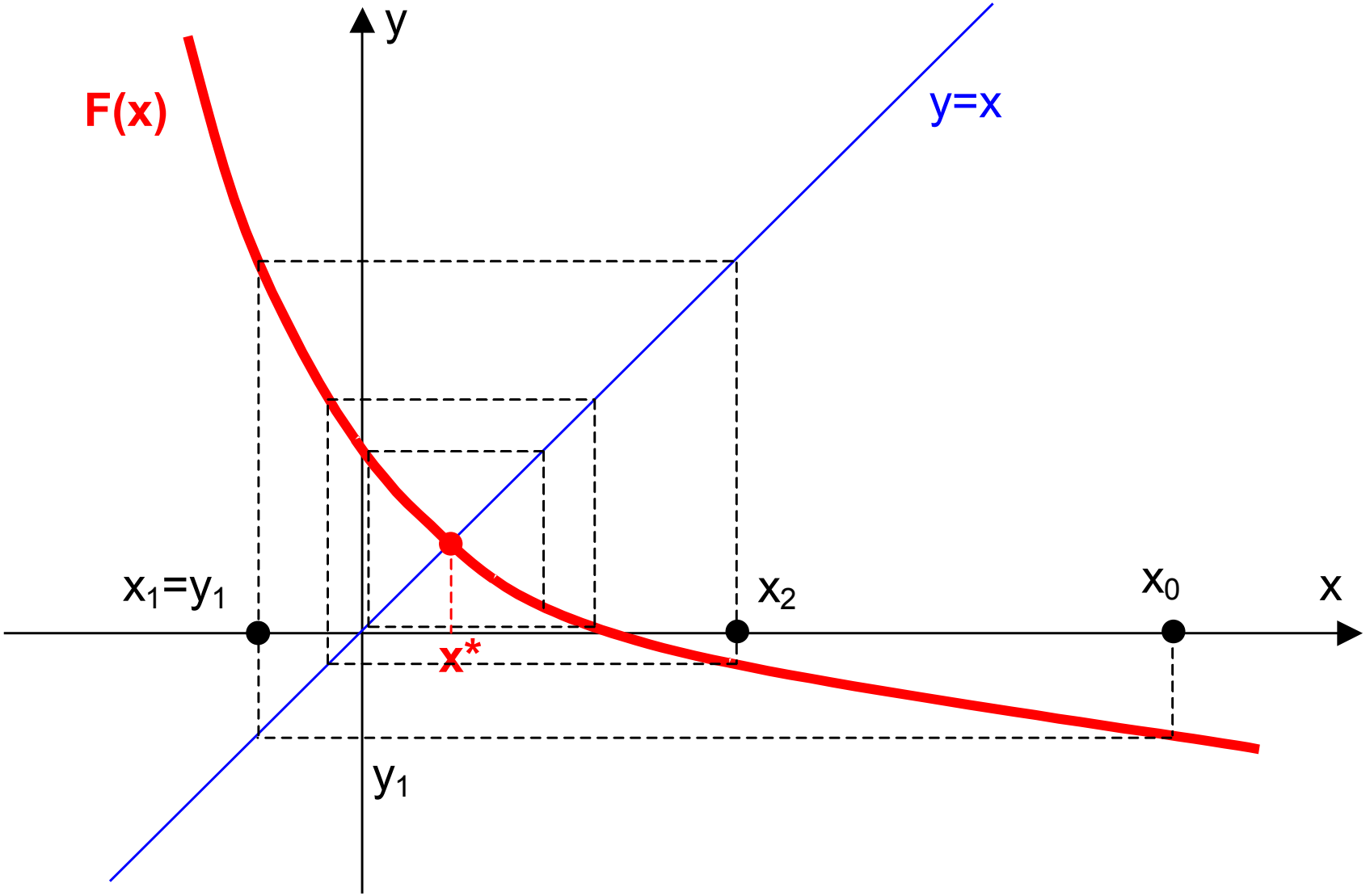
Algorytm:

Wybierz wartość początkową x_0

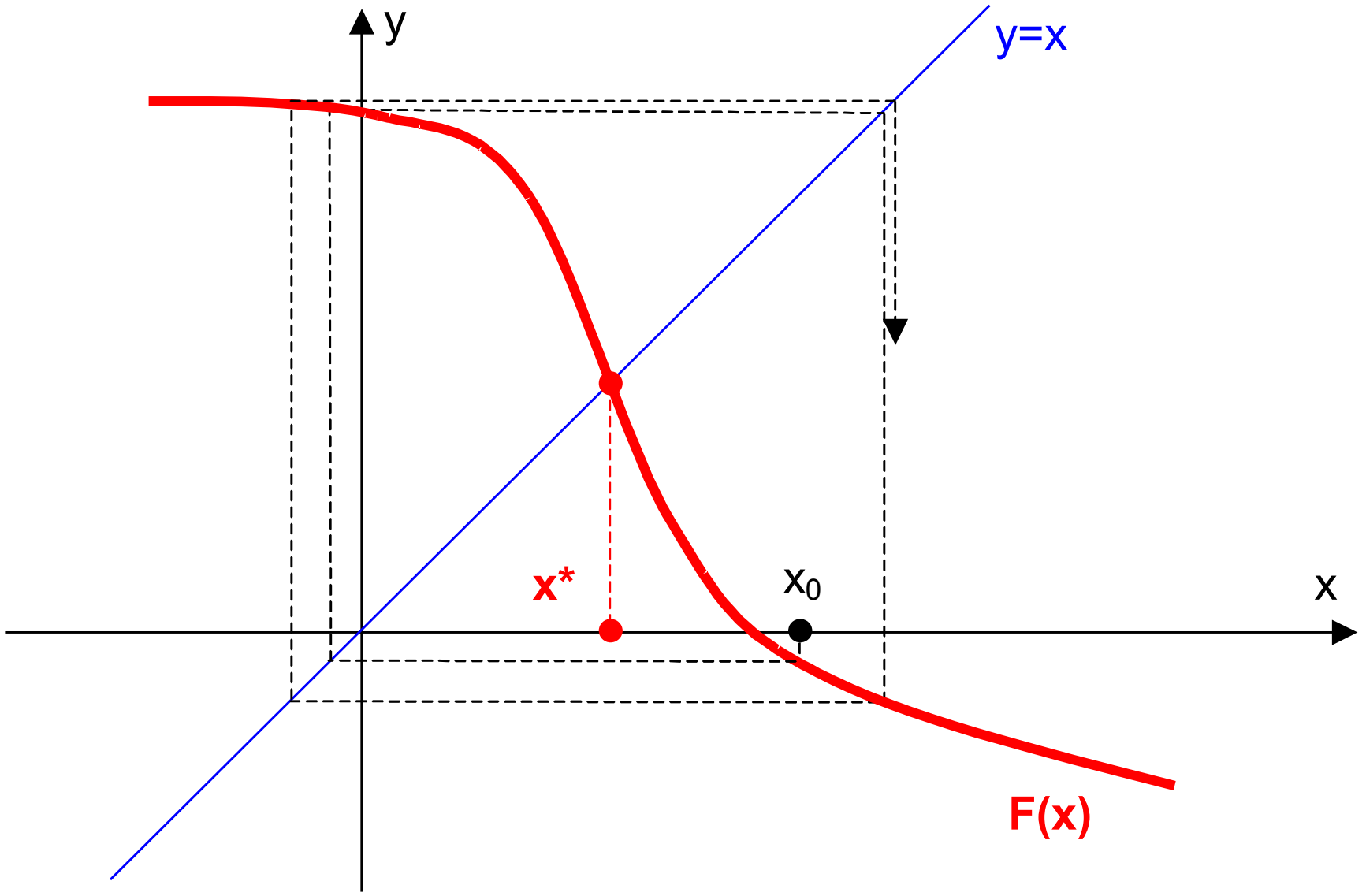
$$x_1 = F(x_0)$$
$$x_2 = F(x_1)$$
$$x_3 = F(x_2)$$
$$\vdots$$
$$x_{n+1} = F(x_n)$$

Jeżeli $|x_{n+1} - x_n| \leq \varepsilon$ **STOP**

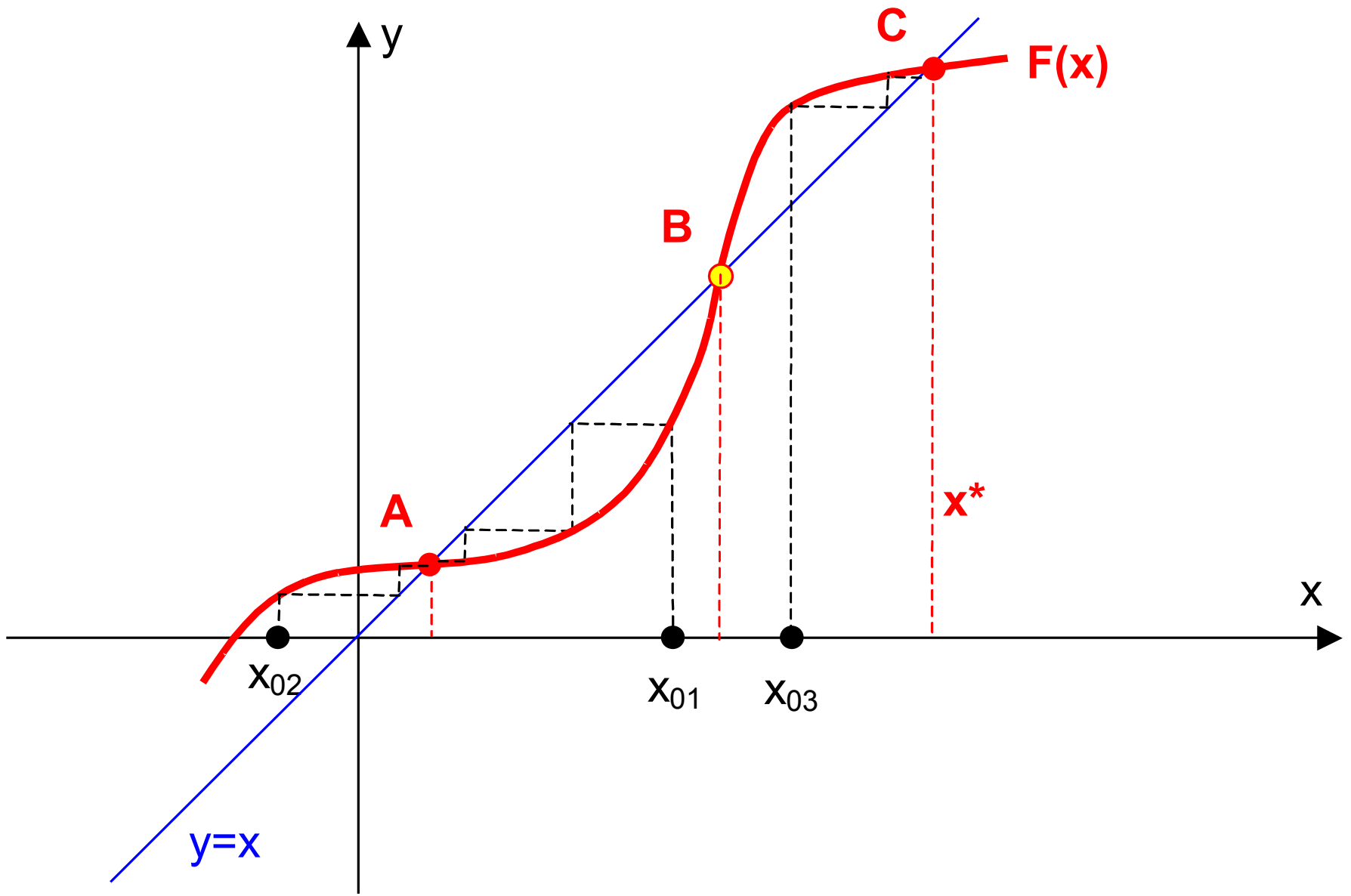
Metoda iteracji prostej - interpretacja geometryczna



Metoda iteracji prostej - brak zbieżności



Metoda iteracji prostej - punkt nieosiągalny



Metoda iteracji prostej dla układu równań

$$\mathbf{x} = \mathbf{F}(\mathbf{x})$$

$$\mathbf{x}_{k+1} = \mathbf{F}(\mathbf{x}_k), \qquad k = 0, 1, 2, \cdots$$

$$\lim_{k \to \infty} \mathbf{x} = \mathbf{x}^*$$

Metoda Newtona-Raphsona

Chcemy rozwiązać równanie:

$$\mathbf{f}\left(\mathbf{x}\right) = \mathbf{0} \qquad \textbf{(a)}$$

Potrzebujemy takiego **F(x)**, dla którego rozwiązanie **x*** równania **F(x) = x** jest jednocześnie rozwiązaniem *(a)*

Metoda Newtona-Raphsona

Można np. przyjąć:

$$\mathbf{F}(\mathbf{x}) = \mathbf{x} - \mathbf{K}(\mathbf{x})\,\mathbf{f}(\mathbf{x}) \qquad \textbf{(b)}$$

gdzie K(x) jest dowolną macierzą n x n

Wtedy równanie ma postać:

$$\mathbf{x} - \mathbf{K}(\mathbf{x})\mathbf{f}(\mathbf{x}) = \mathbf{x}$$

Każde x*, które jest rozwiązaniem równania (a), jest punktem stałym F(x), ponieważ:

L= $\mathbf{F}(\mathbf{x}^*) = \mathbf{x}^* - \mathbf{K}(\mathbf{x}^*)\,\mathbf{0} = \mathbf{x}^*$ P= $\mathbf{x}^*$ L=P

Metoda Newtona-Raphsona

Jednocześnie, jeżeli **x** = **x*** jest punktem stałym dla **F(x)** i jeżeli **K(x)** jest macierzą nieosobliwą dla dowolnego punktu stałego **F(x)**, to x* jest rozwiązaniem równania **f(x)=0**

$$\mathbf{x^* = x^* - K(x^*)\, f(x^*)}$$

$$\mathbf{K(x^*)\, f(x^*) = 0} \quad \Big| * \mathbf{K^{-1}(x^*)} \quad \Rightarrow \quad \mathbf{f(x^*) = 0}$$

Metoda Newtona-Raphsona

Proces zbieżności zależeć będzie od przyjętej macierzy **K**.
Można np. przyjąć:

$$\mathbf{K}(\mathbf{x}) = \mathbf{J}^{-1}(\mathbf{x})$$

$$\mathbf{J}(\mathbf{x}) = \begin{bmatrix} \frac{\partial f_1(\mathbf{x})}{\partial x_1} & \frac{\partial f_1(\mathbf{x})}{\partial x_2} & \cdots & \frac{\partial f_1(\mathbf{x})}{\partial x_n} \\ \frac{\partial f_2(\mathbf{x})}{\partial x_1} & \frac{\partial f_2(\mathbf{x})}{\partial x_2} & \cdots & \frac{\partial f_2(\mathbf{x})}{\partial x_n} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial f_n(\mathbf{x})}{\partial x_1} & \frac{\partial f_n(\mathbf{x})}{\partial x_2} & \cdots & \frac{\partial f_n(\mathbf{x})}{\partial x_n} \end{bmatrix}$$

Metoda Newtona-Raphsona

Równanie (b) w postaci iteracyjnej:

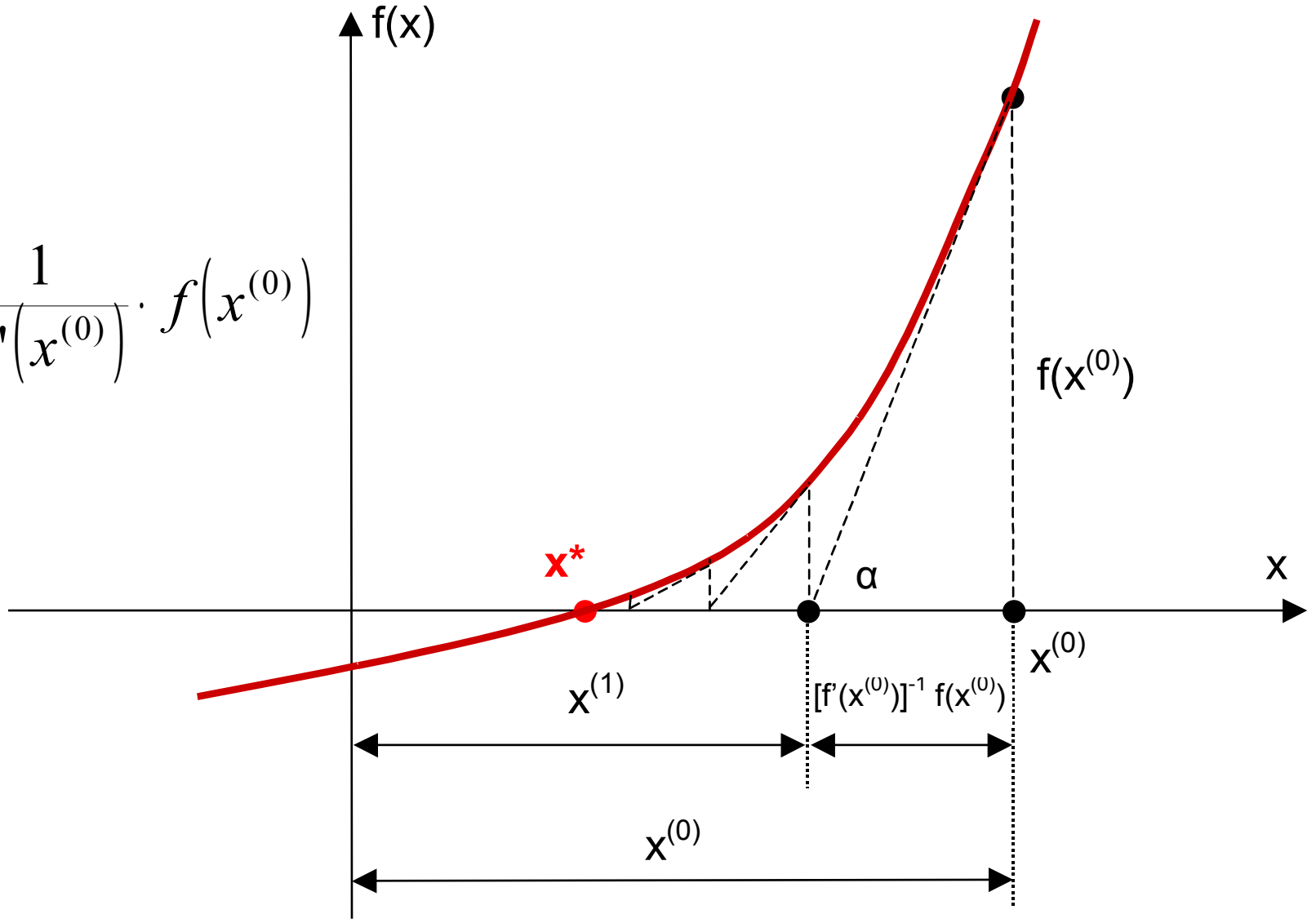
$$\mathbf{F}(\mathbf{x}_j) = \mathbf{x}_j - \mathbf{K}(\mathbf{x}_j)\,\mathbf{f}(\mathbf{x}_j) = \mathbf{x}_{j+1}$$

$$\mathbf{x}_{j+1} = \mathbf{x}_j - \left[\mathbf{J}(\mathbf{x}_j)\right]^{-1}\mathbf{f}(\mathbf{x}_j)$$

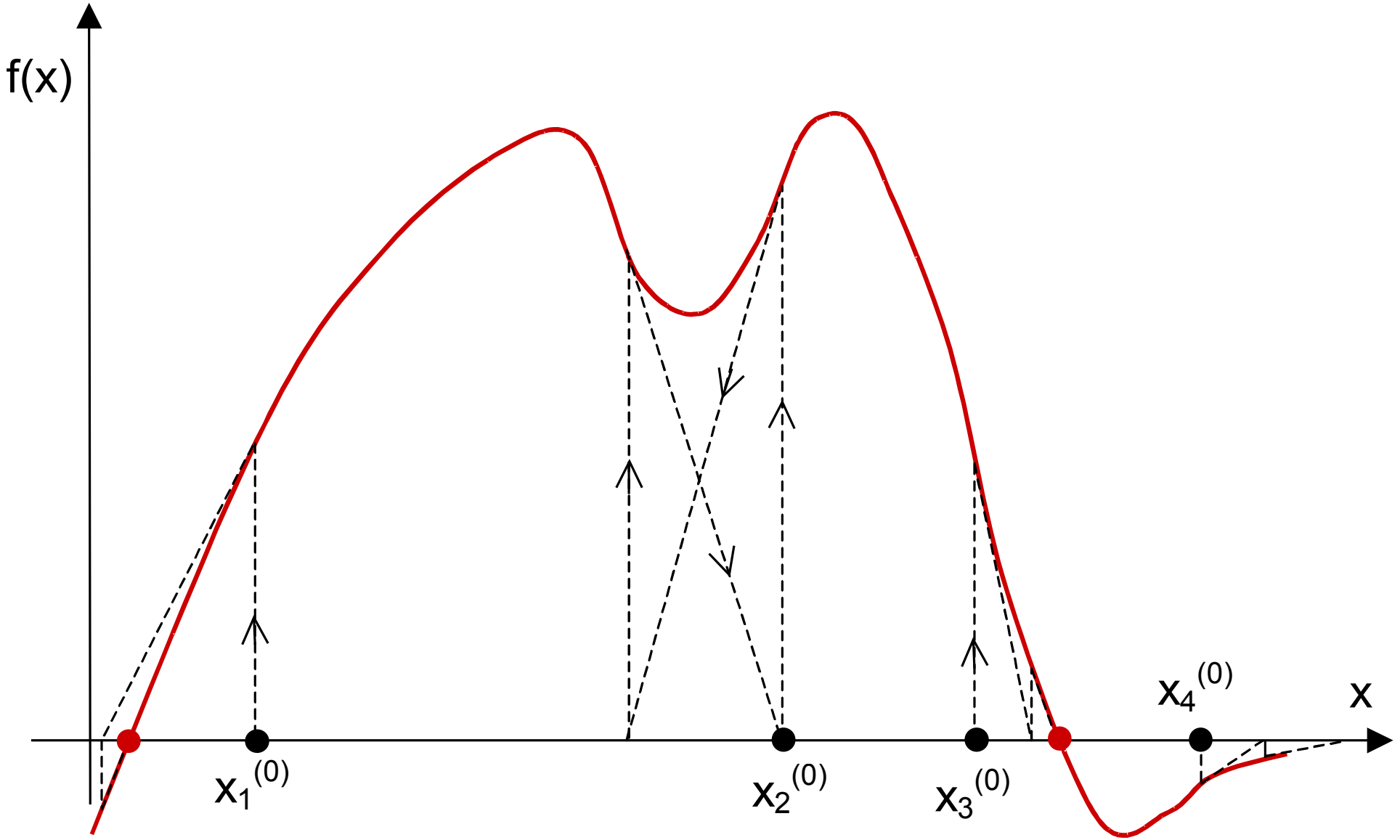
Jest to równanie metody **Newtona-Raphsona**

Metoda Newtona-Raphsona

$$x^{(1)} = x^{(0)} - \frac{1}{f'\left(x^{(0)}\right)} \cdot f\left(x^{(0)}\right)$$



Metoda Newtona-Raphsona



Metoda Newtona-Raphsona - przykład

$$\boldsymbol{F}(\boldsymbol{x}) = \begin{bmatrix} 2x_1^{\,2} + x_2 \\ x_1 + 3x_2^{\,2} \end{bmatrix} \qquad \boldsymbol{F}(\boldsymbol{x}) = \boldsymbol{0}$$

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}^{-1} = \frac{1}{ad - bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$$

Metoda Newtona-Raphsona - przykład

$$\boldsymbol{F}(\boldsymbol{x})=\begin{bmatrix} 2\,x_1^{\,2}+x_2 \\ x_1+3\,x_2^{\,2} \end{bmatrix} \qquad \boldsymbol{F}(\boldsymbol{x})=\boldsymbol{0}$$

$$\boldsymbol{J}(\boldsymbol{x})=\begin{bmatrix} 4\;x_1 & 1 \\ 1 & 6\;x_2 \end{bmatrix} \quad \boldsymbol{J}(\boldsymbol{x})^{-1}=\frac{1}{24\;x_1\,x_2-1}\begin{bmatrix} 6\;x_2 & -1 \\ -1 & 4\;x_1 \end{bmatrix}$$

$$\boldsymbol{x}_{\boldsymbol{nast}}=\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}-\frac{1}{24\;x_1\,x_2-1}\begin{bmatrix} 6\;x_2 & -1 \\ -1 & 4\;x_1 \end{bmatrix}\begin{bmatrix} 2\;x_1^{\,2}+x_2 \\ x_1+3\;x_2^{\,2} \end{bmatrix}$$

Metoda Newtona-Raphsona - przykład

$$\boldsymbol{x}_{nast} = \frac{1}{24\ x_1\ x_2 - 1}\begin{bmatrix} x_2\,(12\ x_1^{\,2} - 3\ x_2) \\ x_1\,(12\ x_2^{\,2} - 2\ x_1) \end{bmatrix}$$

$$\boldsymbol{x}_0 = \begin{bmatrix} -10 \\ -10 \end{bmatrix}$$

$$\boldsymbol{x}_1 = \begin{bmatrix} -5.12714 \\ -5.08545 \end{bmatrix}$$

$$\boldsymbol{x}_2 = \begin{bmatrix} -2.69185 \\ -2.63095 \end{bmatrix}$$

$$\boldsymbol{x}_3 = \begin{bmatrix} -1.47679 \\ -1.40903 \end{bmatrix}$$

$$\boldsymbol{x}_4 = \begin{bmatrix} -0.87518 \\ -0.80803 \end{bmatrix}$$

$$\boldsymbol{x}_5 = \begin{bmatrix} -0.58762 \\ -0.52522 \end{bmatrix}$$

$$\boldsymbol{x}_6 = \begin{bmatrix} -0.46883 \\ -0.41138 \end{bmatrix}$$

$$\boldsymbol{x}_7 = \begin{bmatrix} -0.43892 \\ -0.38352 \end{bmatrix}$$

$$\boldsymbol{x}_8 = \begin{bmatrix} -0.43680 \\ -0.38158 \end{bmatrix}$$

$$\boldsymbol{x}_9 = \begin{bmatrix} -0.43679 \\ -0.38157 \end{bmatrix}$$

$$\boldsymbol{x}_{10} = \begin{bmatrix} -0.43679 \\ -0.38157 \end{bmatrix}$$

$$\boldsymbol{F}(x_{10}) = \begin{bmatrix} -5.5511\mathrm{e}{-17} \\ 0 \end{bmatrix}$$

Algorytm Newtona-Raphsona dla układu n równań

$$f_i\left(x_1, x_2 \cdots x_n\right) = 0 \qquad i = 1, 2, \cdots n$$

$$\mathbf{x}^{(j)} = \left[x_1^{(j)} \;\; x_2^{(j)} \;\; x_n^{(j)} \;\; \right]^T$$

punkt w *j*-tej iteracji

Algorytm Newtona-Raphsona dla układu n równań

Rozwijamy funkcję wielu zmiennych
w szereg Taylora wokół punktu $\mathbf{x}^{(j)}$:

$$f_i\left(x_1, x_2 \cdots x_n\right) = f_i\left(x_1^{(j)}, x_2^{(j)}, \cdots x_n^{(j)}\right) + \left.\frac{\partial f_i(\mathbf{x})}{\partial x_1}\right|_{\mathbf{x}=\mathbf{x}^{(j)}} \left(x_1 - x_1^{(j)}\right) +$$

$$+ \left.\frac{\partial f_i(\mathbf{x})}{\partial x_2}\right|_{\mathbf{x}=\mathbf{x}^{(j)}} \left(x_2 - x_2^{(j)}\right) + \quad \cdots \quad + \left.\frac{\partial f_i(\mathbf{x})}{\partial x_n}\right|_{\mathbf{x}=\mathbf{x}^{(j)}} \left(x_n - x_n^{(j)}\right) +$$

$$+ \frac{1}{2!} \left.\frac{\partial^2 f_i(\mathbf{x})}{\partial x_1^2}\right|_{\mathbf{x}=\mathbf{x}^{(j)}} \left(x_1 - x_1^{(j)}\right)^2 \quad \cdots \qquad dla\ i = 1, 2, \cdots n$$

Algorytm Newtona-Raphsona dla układu n równań

Podstawiamy $\mathbf{x} = \mathbf{x}^{(j+1)}$:

$$f_i\left(x_1^{(j+1)}, x_2^{(j+1)} \cdots x_n^{(j+1)}\right) = f_i\left(x_1^{(j)}, x_2^{(j)}, \cdots x_n^{(j)}\right) + \left.\frac{\partial f_i(\mathbf{x})}{\partial x_1}\right|_{\mathbf{x}=\mathbf{x}^{(j)}} \left(x_1^{(j+1)} - x_1^{(j)}\right) +$$

$$+ \left.\frac{\partial f_i(\mathbf{x})}{\partial x_2}\right|_{\mathbf{x}=\mathbf{x}^{(j)}} \left(x_2^{(j+1)} - x_2^{(j)}\right) + \quad \cdots \quad + \left.\frac{\partial f_i(\mathbf{x})}{\partial x_n}\right|_{\mathbf{x}=\mathbf{x}^{(j)}} \left(x_n^{(j+1)} - x_n^{(j)}\right) +$$

$$+ \frac{1}{2!} \left.\frac{\partial^2 f_i(\mathbf{x})}{\partial x_1^2}\right|_{\mathbf{x}=\mathbf{x}^{(j)}} \left(x_1^{(j+1)} - x_1^{(j)}\right)^2 \quad \cdots \qquad dla\ i = 1, 2, \cdots n$$

Algorytm Newtona-Raphsona dla układu n równań

Jeżeli jesteśmy „blisko” rozwiązania, pomijamy wyrazy wyższych rzędów:

$$f_i\left(\mathbf{x}^{(j+1)}\right) \approx f_i\left(\mathbf{x}^{(j)}\right) + \left.\frac{\partial f_i(\mathbf{x})}{\partial x_1}\right|_{\mathbf{x}=\mathbf{x}^{(j)}} \left(x_1^{(j+1)} - x_1^{(j)}\right) +$$

$$+ \left.\frac{\partial f_i(\mathbf{x})}{\partial x_2}\right|_{\mathbf{x}=\mathbf{x}^{(j)}} \left(x_2^{(j+1)} - x_2^{(j)}\right) + \quad \cdots \quad + \left.\frac{\partial f_i(\mathbf{x})}{\partial x_n}\right|_{\mathbf{x}=\mathbf{x}^{(j)}} \left(x_n^{(j+1)} - x_n^{(j)}\right)$$

$$dla\ i = 1, 2, \cdots n$$

Algorytm Newtona-Raphsona dla układu n równań

Jeżeli $\mathbf{x}^{(j+1)}$ jest już rozwiązaniem naszego równania, czyli $\mathbf{f}(\mathbf{x}^{(j+1)}) = \mathbf{0}$, to:

$$\mathbf{J}\left(\mathbf{x}^{(j)}\right)\left(\mathbf{x}^{(j+1)} - \mathbf{x}^{(j)}\right) = -\mathbf{f}\left(\mathbf{x}^{(j)}\right)$$

Symulacja i eksperyment komputerowy

- Symulacja komputerowa - metoda odtwarzania zjawisk zachodzących w świecie rzeczywistym (lub ich niektórych właściwości i parametrów) za pomocą ich zmatematyzowanych modeli, definiowanych i obsługiwanych przy użyciu programów komputerowych; wykorzystywana do wnioskowania o przebiegu tych zjawisk i procesów, których bezpośrednia obserwacja jest niemożliwa lub zbyt kosztowna. - Encyklopedia PWN

Zalety symulacji komputerowej

- Koszt - nie ma potrzeby budowy rzeczywistego układu, nic nie może ulec zniszczeniu
- Czas - symulacja jest metodą szybszą niż konstrukcja układu (choć sam proces symulacji może być długotrwały)
- Modyfikowalność - zasymulowany układ łatwo jest zmienić
- Łatwość i różnorodność analiz - niektórych analiz wręcz nie da się sensownie zrealizować w układzie rzeczywistym

Zalety symulacji komputerowej

- Wiedza - możliwość wszechstronnego zbadania układu w różnych warunkach pracy, zrozumienia jego działania, możliwość śledzenia zmian normalnie zbyt szybkich do uchwycenia, łatwiejsze znajdowanie błędów
- Alternatywa dla rozwiązań analitycznych - w dużych systemach mogą być one nieosiągalne
- Postprocessing - wygodna prezentacja wyników, np. w postaci graficznej

Wady symulacji komputerowej

- Poprawność wyników ściśle zależy od poprawności modelu
  - modele o małej złożoności (np. aproksymujące działanie przyrządu parametryzowanymi równaniami) mogą:
    - nie uwzględniać pewnych zjawisk, a w rezultacie dawać duże błędy w nietypowych sytuacjach
    - być nieprawdziwe poza założonym obszarem pracy
    - być trudne w modyfikacji przy modyfikacji przyrządu (postępie technologicznym itp.)
    - bezpodstawnie zakładać że wystarczająca jest symulacja jednodomenowa
    - mieć problemy ze stabilnością

Wady symulacji komputerowej

- modele o dużej złożoności (np. wielodomenowe modele wykorzystujące metodę elementów skończonych) mogą:
  - być zbyt skomplikowane aby umożliwić symulację układów w rozsądnym czasie
  - mieć ogromne wymagania pamięciowe
  - być bardzo praco i czasochłonne w przygotowaniu
  - mieć problemy ze stabilnością
- Prawidłowe zaprojektowanie symulacji może być trudne
- Rozrzut parametrów może podważyć wyniki symulacji

Wady symulacji komputerowej

- Interpretacja wyników ostatecznie jest pozostawiona czynnikowi ludzkiemu
- Ograniczenia systemów komputerowych (np. ograniczona precyzja liczb) mogą przekładać się na ograniczenia symulacji

Eksperyment komputerowy

- Eksperyment komputerowy - eksperyment przeprowadzany w całości przy użyciu symulacji komputerowej
- Może służyć np.:
  - testowaniu hipotez
  - przygotowywaniu bądź analizie rzeczywistych eksperymentów
  - weryfikowaniu rozwiązań uproszczonych
  - projektowaniu bądź przeprojektowywaniu urządzeń i układów

Podział modeli

- Modele można dzielić np. na:
  - liniowe i nieliniowe
  - deterministyczne i probabilistyczne
  - statyczne i dynamiczne
  - ciągłe i dyskretne
  - o parametrach skupionych bądź rozłożonych
  - jedno i wielodomenowe

Oprogramowanie

- Modele tworzy się i symuluje z wykorzystaniem oprogramowania
- Oprogramowanie może być:
  - językiem programowania (np. Simula)
  - ogólnego przeznaczenia z elementami specjalizowanymi (np. Matlab)
  - opracowane pod kątem konkretnej dziedziny, np. symulacji układów elektronicznych:
    - Spice
    - Eldo
    - Saber

Komputerowe opracowywanie wyników pomiarów

- Oprócz sytuacji gdy symulacje są przeprowadzane całkowicie przy użyciu komputera, często zachodzi potrzeba wprowadzenia rzeczywistych danych pomiarowych
- Aby maksymalnie wykorzystać zawarte w nich informacje, praktycznie zawsze konieczna jest jakaś forma wstępnego przetworzenia

Komputerowe opracowywanie wyników pomiarów

- Może ono zawierać np.:
  - usuwanie wartości błędnych
  - uzupełnianie wartości brakujących
  - filtrowanie (np. górno/dolno/pasmowowprzepustowe)
  - wygładzanie
  - przewidywanie wartości przyszłych, odtwarzanie historycznych
  - decymację
  - dyskretyzację

Aproksymacja i interpolacja

- Są metodami przy użyciu których znajduje się krzywą f(x), opisaną równaniem (równaniami) analitycznym, na podstawie zbioru punktów
- Punkty mogą pochodzić np. z pomiaru rzeczywistego procesu
- Celem może być np. łatwość dalszej analizy/przetwarzania, weryfikacja hipotez dotyczących charakteru zjawiska, wygładzanie, przewidywanie, uzupełnianie wartości brakujących itd.

Aproksymacja i interpolacja

- Aby przeprowadzić aproksymację bądź interpolację, należy przyjąć jakąś postać funkcji, możliwą do parametryzacji (np. wielomian, wielomian uogólniony)
- Różnica między aproksymacją a interpolacją polega na wymaganiu dotyczącym przechodzenia krzywej przez dane punkty:
  - aproksymacja nie wymaga przejścia przez dane punkty
  - interpolacja wymaga przejścia przez dane punkty

Aproksymacja

- Zadanie:
  - mając n punktów (par wartości) (x_i, y_i) dla i = 1,..., n
  - znaleźć funkcję postaci $f(x, \beta)$, gdzie β jest wektorem m parametrów, m < n
  - tak aby zminimalizować błąd $S = \sum_{i-1}^{n} r_i^2$, gdzie $r_i = y_i - f\left(x_i, \hat{\beta}\right)$ jest błędem aproksymacji dla pojedynczego punktu

Aproksymacja

- Zadanie to można rozumieć jako próbę przybliżonego rozwiązania równania o postaci

$$X\beta = y$$

  przy czym liczba wierszy X jest większa niż liczba kolumn
- każdy wiersz X odpowiada jednemu narzuconemu punktowi
- kolejne wartości w wierszu to wartości kolejnych składowych wielomianu

Aproksymacja

- Jest to metoda najmniejszych kwadratów (LSQ)
- Zadania aproksymacyjne można podzielić na dwie kategorie:
  - liniowe
  - nieliniowe
- Zadanie liniowe ma rozwiązanie analityczne, zadanie nieliniowe rozwiązuje się numerycznie, metodami iteracyjnymi

Liniowa metoda najmniejszych kwadratów (LLSQ)

- Liniowość nie oznacza, że metoda nadaje się jedynie do liniowych funkcji, a tylko że funkcja jest liniowa względem parametrów β
- Funkcja ma więc postać: $f(x,\beta)=\sum_{j=1}^{m} X_j \beta_j$,

  gdzie X_j jest funkcją x (w tym funkcją nieliniową, stałą itd.)

Liniowa metoda najmniejszych kwadratów (LLSQ)

- Aby znaleźć minimum funkcji błędu, należy znaleźć punkt dla którego pochodne cząstkowe po wszystkich parametrach mają wartość 0

- Pochodne cząstkowe wyrażają się wzorem

$$\frac{\delta S}{\delta \beta_j} = 2 \sum_{i=1}^{n} r_i \frac{\delta r_i}{\delta \beta_j}$$

- Ponieważ $r_i = y_i - \sum_{j=1}^{m} X_{ij} \beta_j$ pochodna we wzorze powyżej ma postać $\frac{\delta r_i}{\delta \beta_j} = -X_{ij}$

Liniowa metoda najmniejszych kwadratów (LLSQ)

- Podstawiając do wzoru na pochodną cząstkową funkcji błędu i przyrównując do 0 mamy

$$\frac{\delta S}{\delta \beta_j} = 2\sum_{i=1}^{n} r_i \frac{\delta r_i}{\delta \beta_j} = -2\sum_{i=1}^{n} \left(y_i - \sum_{k=1}^{m} X_{ik} \beta_k \right) X_{ij} = 0$$

- Po uporządkowaniu otrzymujemy

$$\sum_{i=1}^{n} \sum_{k=1}^{m} X_{ij} X_{ik} \beta_k = \sum_{i=1}^{n} X_{ij} y_i$$

Liniowa metoda najmniejszych kwadratów (LLSQ)

- W postaci macierzowej:

$$\left(X^T X\right)\boldsymbol{\beta} = X^T y$$

- Rozwiązanie względem $\boldsymbol{\beta}$ daje poszukiwane współczynniki

Liniowa metoda najmniejszych kwadratów (LLSQ) - przykład

- Znaleźć współczynniki funkcji aproksymującej następujące punkty:
  1,2; 2, 4; 3, 8; 4, 18; 5, 27
- Przyjąć funkcję postaci $f(x)=\beta_3 x^2+\beta_2 x+\beta_1$

Liniowa metoda najmniejszych kwadratów (LLSQ) - przykład

- Układ równań:

$$\beta_3 1^2+\beta_2 1+\beta_1=2$$
$$\beta_3 2^2+\beta_2 2+\beta_1=4$$
$$\beta_3 3^2+\beta_2 3+\beta_1=8$$
$$\beta_3 4^2+\beta_2 4+\beta_1=18$$
$$\beta_3 5^2+\beta_2 5+\beta_1=27$$

- Odpowiadająca mu macierz $\boldsymbol{X}$:

$$\begin{matrix} 1 & 1 & 1 \\ 4 & 2 & 1 \\ 9 & 3 & 1 \\ 16 & 4 & 1 \\ 25 & 5 & 1 \end{matrix}$$

Liniowa metoda najmniejszych kwadratów (LLSQ) - przykład

- Lewa strona: $X^T X$=

  979 225 55
  225 55 15
  55 15 5

- Prawa strona: $X^T y$=

  1053
  241
  59

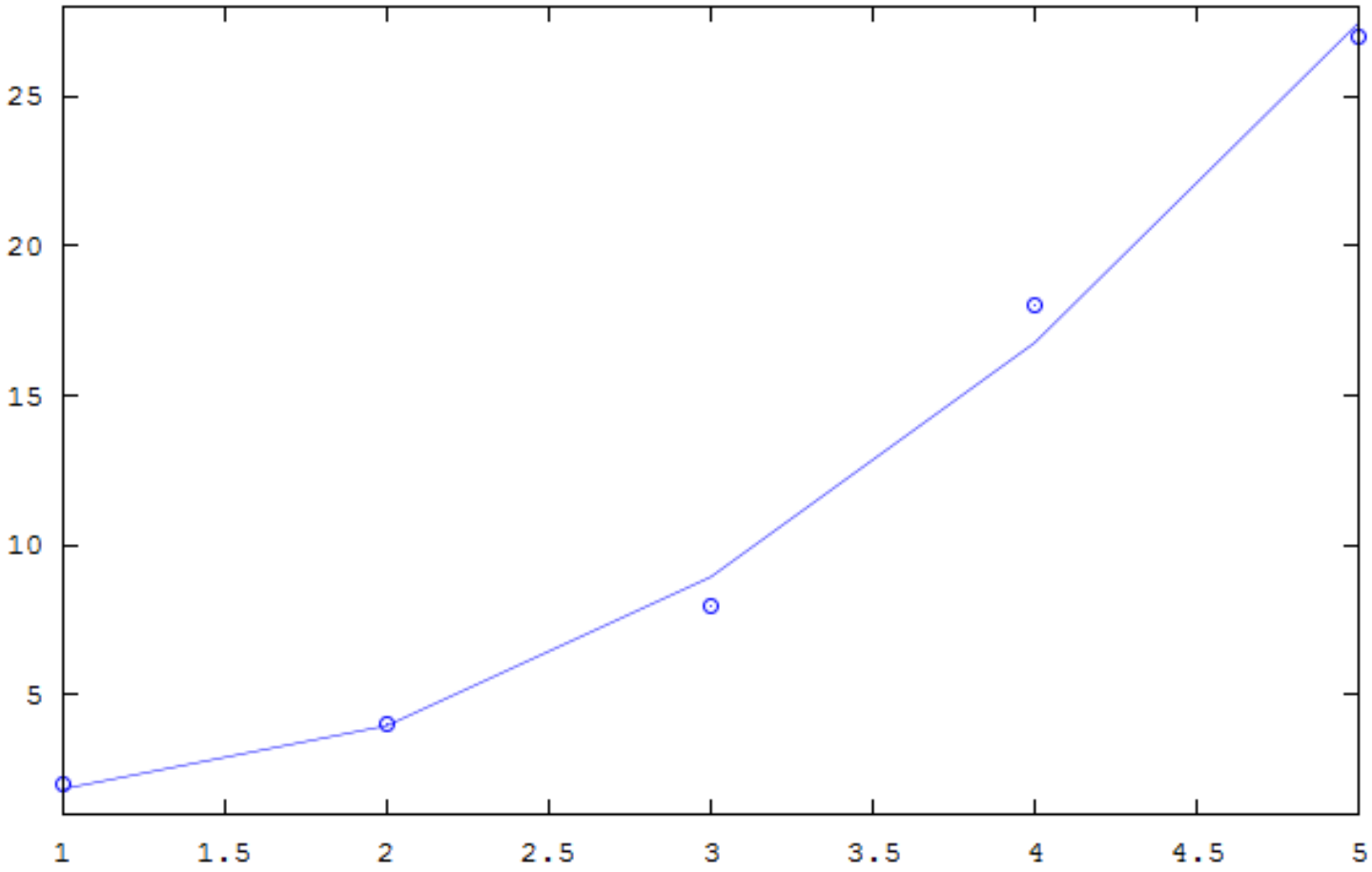
- Rozwiązując otrzymujemy β=

  1.4286
  -2.1714
  2.6

Liniowa metoda najmniejszych kwadratów (LLSQ) - przykład

- Poszukiwane równanie ma więc postać

$$f(x) = 1.4286x^2 - 2.1714x + 2.6$$



Liniowa metoda najmniejszych kwadratów (LLSQ) - przykład

- W Matlabie aproksymację wielomianem dowolnego stopnia można przeprowadzić przy pomocy funkcji `polyfit`
- Argumentami są współrzędne x punktów, współrzędne y punktów, stopień wielomianu

```
octave-3.0.0.exe:7> x=[1 2 3 4 5];
octave-3.0.0.exe:8> y=[2 4 8 18 27];
octave-3.0.0.exe:9> p=polyfit(x, y, 2)
p =

   1.4286  -2.1714   2.6000
```

Liniowa metoda najmniejszych kwadratów (LLSQ)

- Przy stosowaniu aproksymacji należy dobrać stopień złożoności funkcji aproksymującej (np. stopień wielomianu):
  - skomplikowana funkcja może zapewnić mniejszy błąd aproksymacji (będzie przebiegać bliżej punktów), ale
  - może mieć dużo zafalowań i ma mniejszą zdolność “uogólniania” (np. ignorowania błędnych pomiarów)

Interpolacja

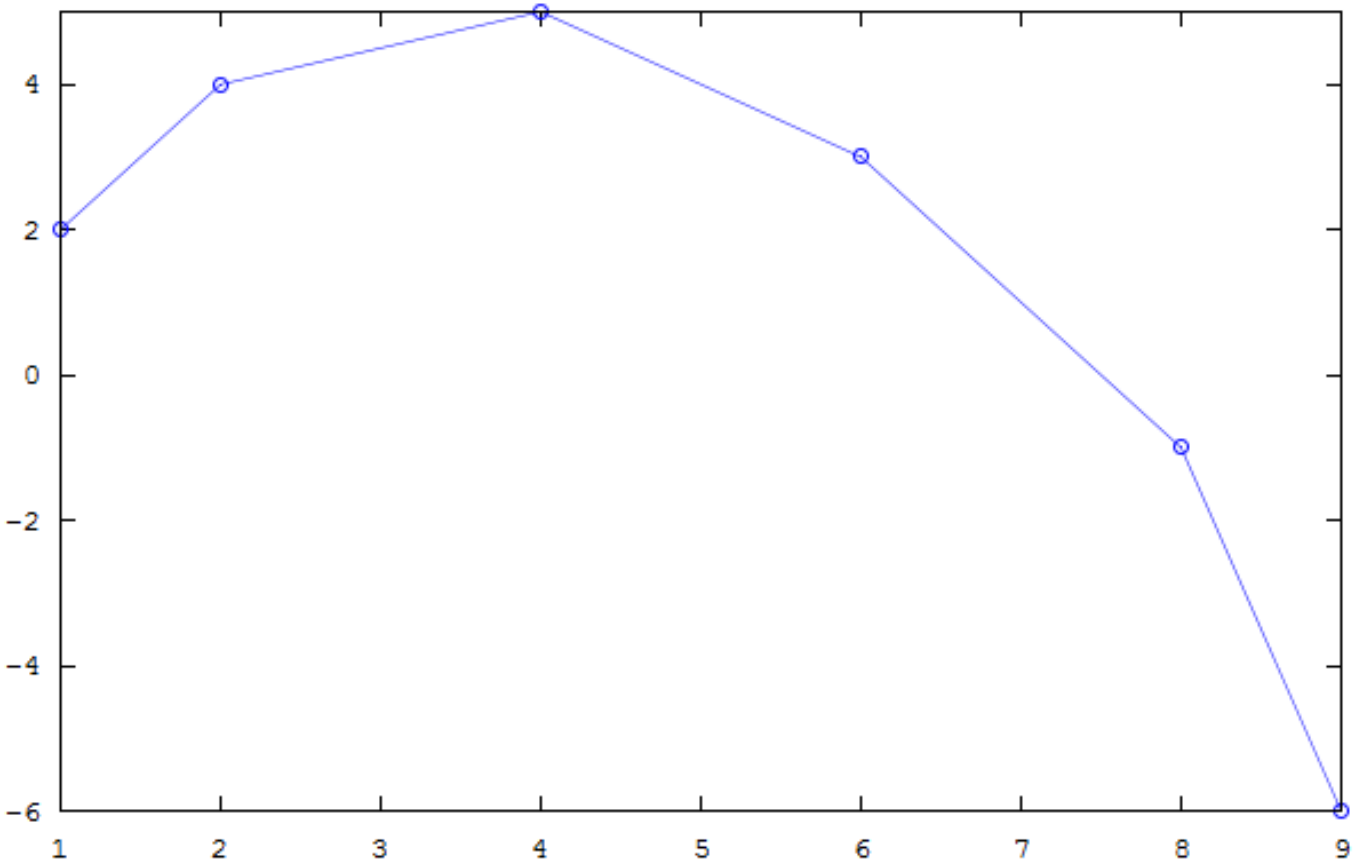
- Interpolacja może zostać użyta do znalezienia wartości (funkcji, hipotetycznego pomiaru) dla argumentu dla którego nie posiadamy tej wartości
- Interpolacja dotyczy znajdowania wartości dla argumentu znajdującego się pomiędzy argumentami dla których wartości są znane
- Jeśli szukamy wartości dla argumentu poza tym przedziałem, jest to ekstrapolacja
- Często interesujący nas obszar dzieli się na mniejsze i interpoluje osobno

Interpolacja liniowa

- Najprostsza (nie licząc interpolacji odcinkami stałej) metoda
- Polega na przeprowadzeniu prostej przez dwa najbliższe znane punkty
- Wartości dla dowolnego argumentu odczytuje się jako wartość funkcji liniowej

```
octave-3.0.0.exe:12> x=[1 2 4 6 8 9];
octave-3.0.0.exe:13> y=[2 4 5 3 -1 -6];
octave-3.0.0.exe:14> plot(x, y)
octave-3.0.0.exe:15> plot(x, y, '-o')
```

Interpolacja liniowa



Interpolacja wielomianowa

- Koncepcja bardzo podobna do aproksymacji, tyle, że liczba punktów jest równa stopniowi wielomianu pomniejszonemu o 1:
  - liniowa: 2 punkty - 2 równania, wielomian 1 stopnia,2 niewiadome
  - kwadratowa: 3 punkty - 3 równania, wielomian 2 stopnia, 3 niewiadome
  - sześcienna: 4 punkty - 4 równania, wielomian 3 stopnia, 4 niewiadome
  - itd.

Interpolacja wielomianowa

- Liczba równań odpowiada liczbie niewiadomych, jeśli wśród przyjętych punktów nie ma punktów o identycznych odciętych, układ równań jest oznaczony i można otrzymać współczynniki wielomianu interpolującego

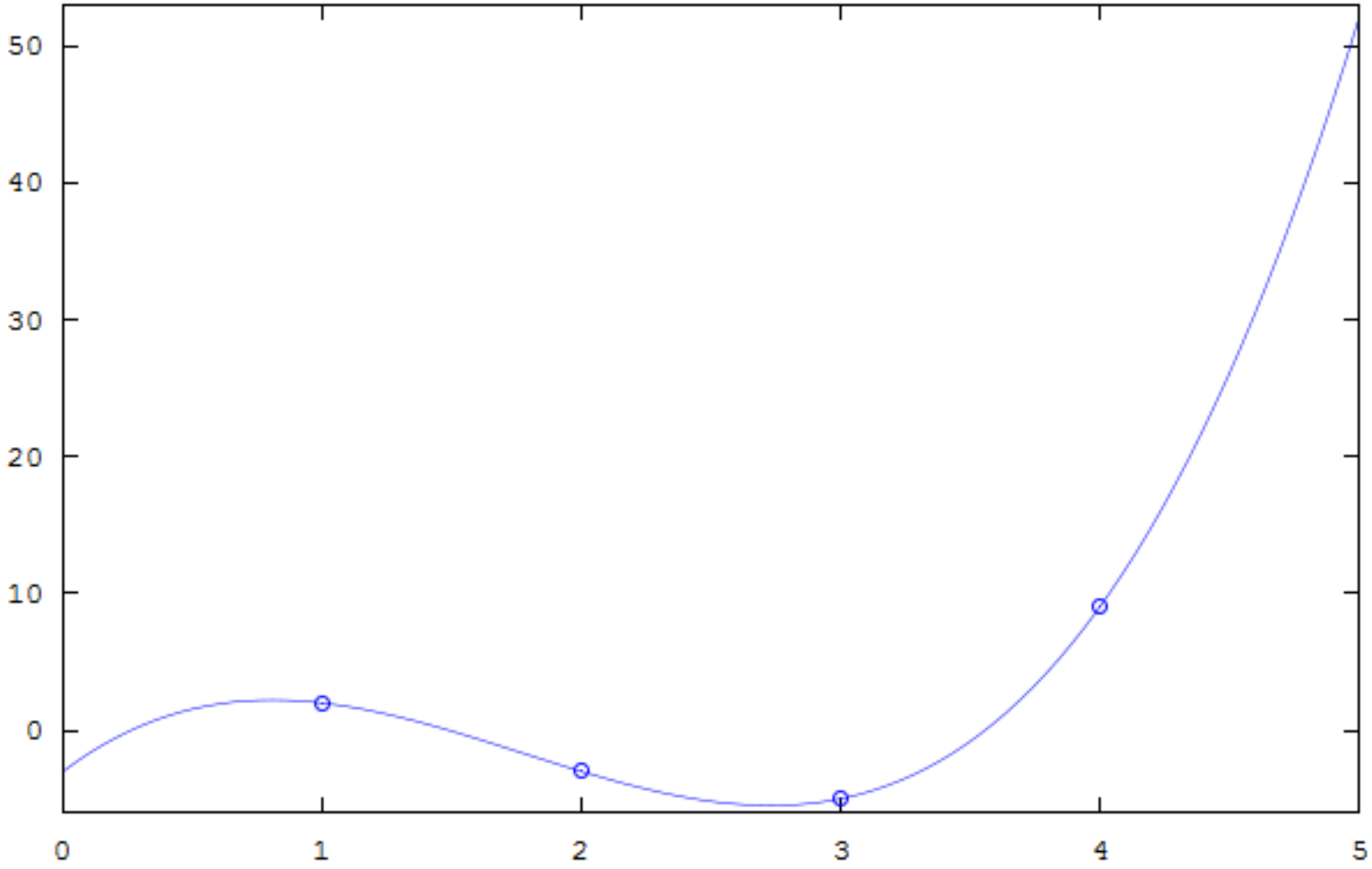
Interpolacja wielomianowa - przykład

- Znaleźć wielomian interpolujący dla zbioru punktów:
  1, 2; 2, -3; 3, -5; 4, 9
- 4 punkty, czyli wielomian 3 stopnia:
  $f(x)=\beta_4 x^3+\beta_3 x^2+\beta_2 x+\beta_1$
- Równania:
  $\beta_4 1^3+\beta_3 1^2+\beta_2 1+\beta_1=2$
  $\beta_4 2^3+\beta_3 2^2+\beta_2 2+\beta_1=-3$
  $\beta_4 3^3+\beta_3 3^2+\beta_2 3+\beta_1=-5$
  $\beta_4 4^3+\beta_3 4^2+\beta_2 4+\beta_1=9$

Interpolacja wielomianowa - przykład

- Macierzowo:
  $\boldsymbol{X\beta=y}$
- $\boldsymbol{X}=\begin{matrix}1 & 1 & 1 & 1\\ 8 & 4 & 2 & 1\\ 27 & 9 & 3 & 1\\ 64 & 16 & 4 & 1\end{matrix}$
- Rozwiązując względem $\boldsymbol{\beta}$ otrzymujemy:
  $\beta_1=-3,\ \beta_2=14.3333,\ \beta_3=-11.5,\ \beta_4=2.1667$

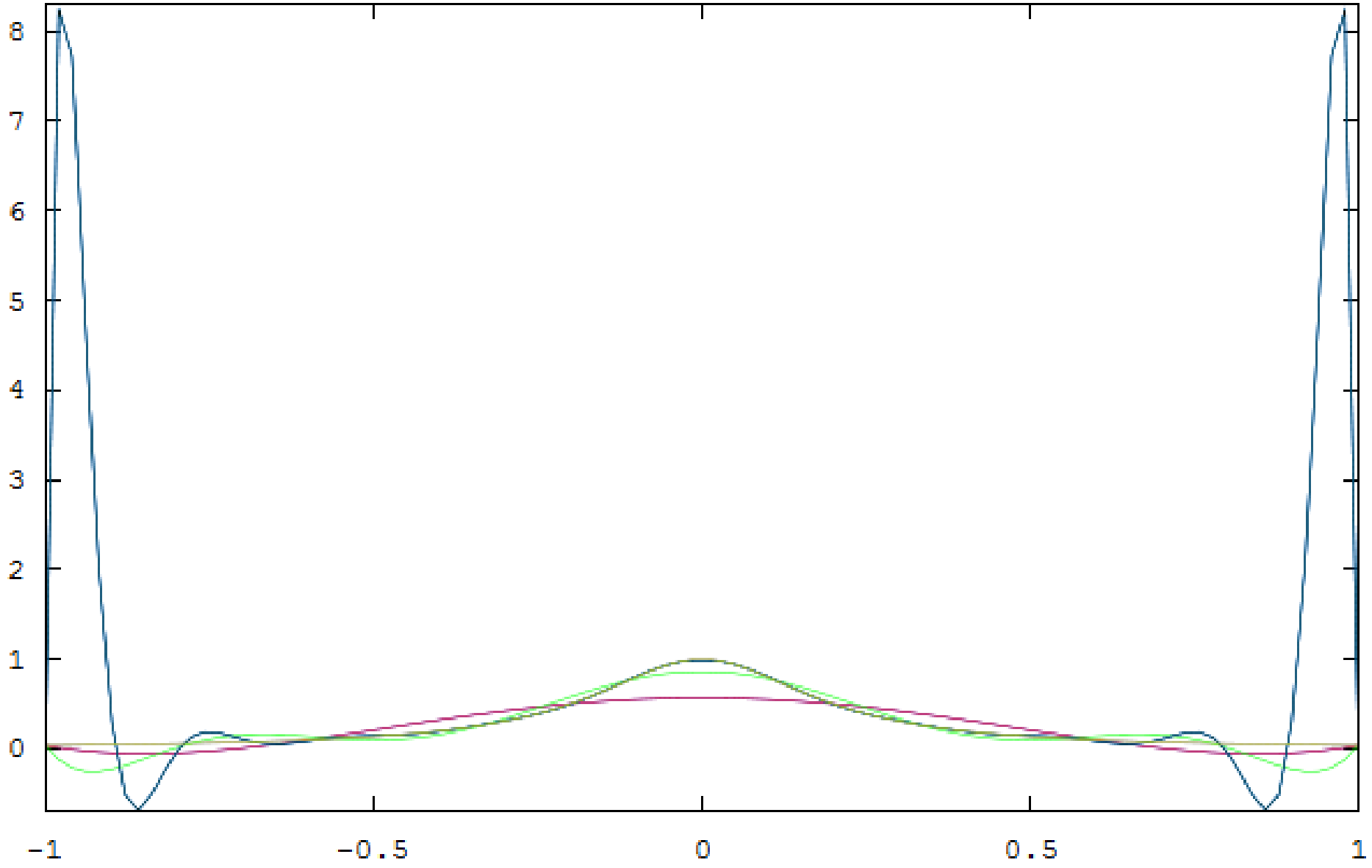
Interpolacja wielomianowa - przykład



Interpolacja wielomianowa - problemy (efekt Rungego)

- Zadanie:
  - przeprowadzić interpolację wielomianową funkcji $1/(1+25x^2)$ w przedziale $<-1; 1>$
  - jako punkty interpolacyjne przyjąć kolejno 6, 10 i 20 równomiernie rozłożonych punktów
- W efekcie przeprowadzona zostanie interpolacja wielomianami 5, 9 i 19 stopnia

Interpolacja wielomianowa - problemy (efekt Rungego)



Interpolacja funkcjami sklejanymi (spline)

- Funkcja sklejana: $S:[a,b]\to\mathbb{R}$
  składa się z wielomianów $P_i:\left[t_i,t_{i+1}\right)\to\mathbb{R}$
  gdzie $a=t_0<t_1<\cdots t_{k-2}<t_{k-1}=b$
- Punkty t nazywane są węzłami
- Inaczej:

$$S(t)=P_0(t),\ t_0\leqslant t<t_1$$
$$S(t)=P_1(t),\ t_1\leqslant t<t_2$$
$$\vdots$$
$$S(t)=P_{k-2}(t),\ t_{k-2}\leqslant t<t_{k-1}$$

Interpolacja funkcjami sklejanymi (spline)

- Jeśli wszystkie wielomiany są stopnia co najwyżej n, funkcja sklejana jest stopnia co najwyżej n
- W węzłach wartości funkcji “wcześniejszej” i “późniejszej” są sobie równe
- Zwykle wymaga się też równych wartości pochodnych

Interpolacja sześciennymi funkcjami sklejanymi

- Funkcje są funkcjami sześciennymi
- W węzłach wymagana jest równość funkcji oraz pierwszej i drugiej pochodnej
- Jeśli mamy n+1 punktów (n wielomianów) to:
  - potrzebujemy 4n parametrów (po 4 dla każdego sześciennego wielomianu)
  - mamy n+1 warunków z własności interpolacyjnej (przechodzenie funkcji przez dane punkty)
  - mamy 3(n-1) warunków z równości funkcji i pochodnych dla punktów “wewnętrznych” (bez punktów skrajnych)

Interpolacja sześciennymi funkcjami sklejanymi

- W sumie mamy n+1 + 3(n-1) = 4n-2 warunków i 4n parametrów - brakuje 2 warunków
- Można przyjąć np. $S''(x_0)=S''(x_n)=0$ - jest to tzw. naturalna sześcienna funkcja sklejana

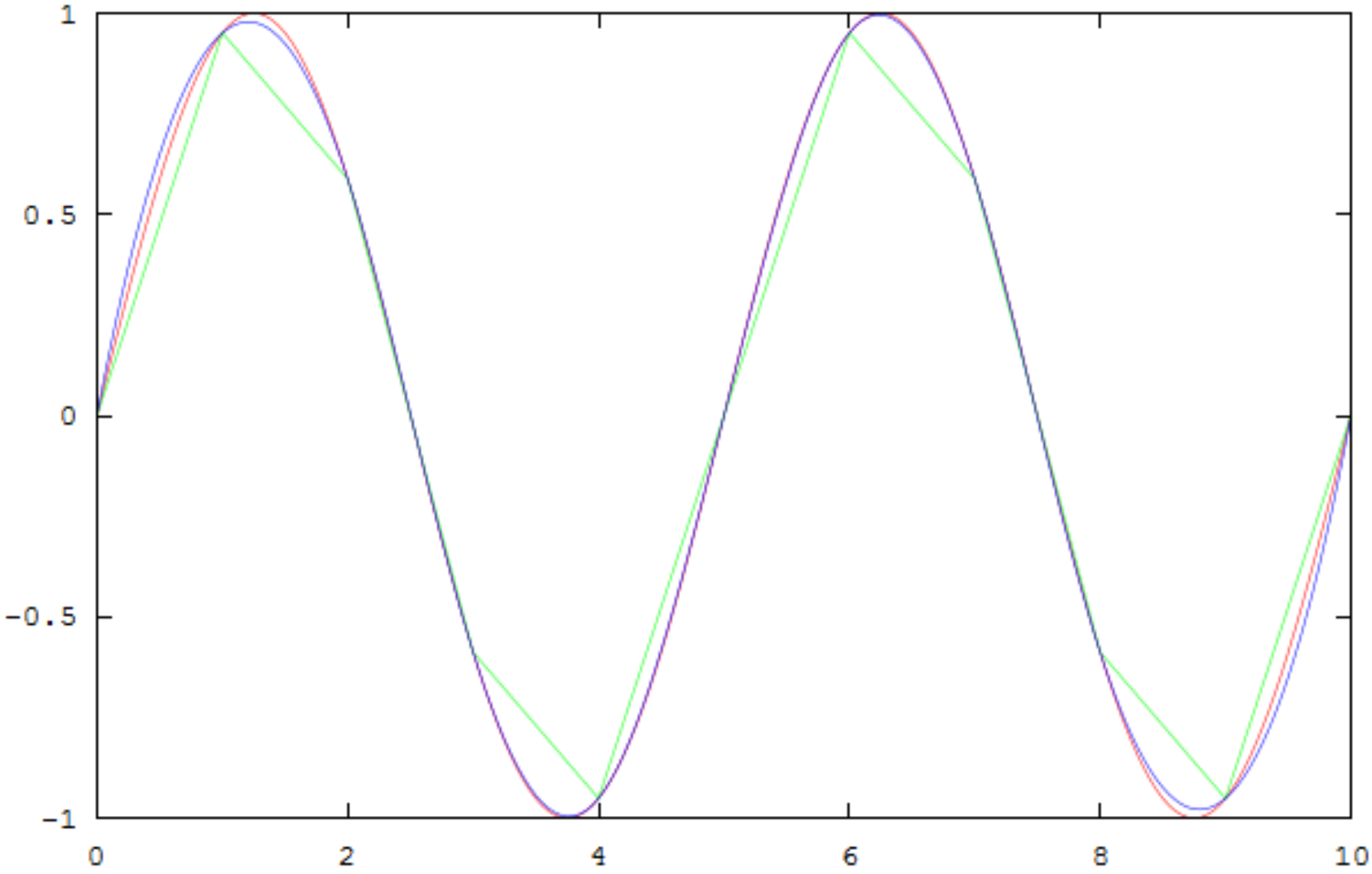
Interpolacja sześciennymi funkcjami sklejanymi - Matlab

- W Matlabie interpolację można uzyskać funkcją interp1(X, Y, XI, METHOD)
- Argumentami są:
  X: dane argumenty
  Y: wartości funkcji dla danych argumentów
  XI: argumenty w których ma być wyznaczona funkcja
  METHOD: sposób interpolacji, np. linear, spline

Interpolacja sześciennymi funkcjami sklejanymi - Matlab

```
xf=[0:0.05:10]; %gdzie okreslic funkcje
xp=[0:10];      %gdzie znamy wartości funkcji
yp = sin(2*pi*xp/5); %znane wartości
lin=interp1(xp,yp,xf); %interpolacja liniowa
spl=interp1(xp,yp,xf,'spline'); %sklejana
yf = sin(2*pi*xf/5); %dla porownania
plot(xf,yf,"r",xf,lin,"g",xf,spl,"b");
```

Interpolacja sześciennymi funkcjami sklejanymi - Matlab



Sygnały

- Sygnał - dowolna wielkość zmieniająca się w czasie lub przestrzeni
- W praktyce wszystko co jest mierzalne może być potraktowane jako sygnał
- W celu przetwarzania sygnały są zwykle konwertowane na inne sygnały, łatwo poddające się obróbce, np. sygnały elektryczne
- Ostatecznie często sygnał przybiera formę reprezentacji matematycznej

Podział sygnałów

- Z czasem ciągłym bądź dyskretnym
- Z wielkościami ciągłymi bądź skwantowanymi
- Analogowe, cyfrowe

Kwantyzacja

- Proces przybliżenia wartości ciągłych poprzez skończoną liczbę wartości
- Może również służyć do przybliżania wartości reprezentowanych przez skończoną, ale dużą, liczbę wartości poprzez mniejszą liczbę wartości
- Zawsze prowadzi do straty informacji
- Pojawia się błąd, zwany błędem kwantyzacji

Dyskretyzacja (próbkowanie)

- Próbkowanie jest przeprowadzane poprzez pomiar sygnału ciągłego w równomiernych odstępach czasu - co T
- T jest okresem próbkowania
- f=1/T to częstotliwość próbkowania
- Musi być spełnione twierdzenie Nyquista-Shannona

Twierdzenie Nyquista-Shannona

- Sygnał analogowy może być idealnie odtworzony z próbek (sygnału spróbkowanego) jeśli częstotliwość próbkowania przekraczała dwukrotną maksymalną częstotliwość sygnału wejściowego
- Sygnał może być odtworzony przy pomocy wzoru interpolacyjnego Whittakera-Shanona:

$$x(t) = \sum_{n=-\infty}^{\infty} x[n] \cdot sinc\left(\frac{t - nT}{T}\right)$$

gdzie

$$sinc(x) = \frac{\sin(\pi x)}{\pi x}$$

Próbkowanie

- Twierdzenie to oznacza, że tylko sygnały o ograniczonej częstotliwości mogą być idealnie spróbkowane
- Żaden sygnał o ograniczonym czasie trwania nie ma ograniczonej częstotliwości
- Jeśli twierdzenie Nyquista-Shannona nie jest spełnione, mamy do czynienia z aliasingiem
- W praktyce stosuje się filtry dolnoprzepustowe przed próbkowaniem sygnału

Kwantyzacja

- Dopuszczalny zakres wartości wejściowych dzieli się na skończoną liczbę przedziałów N
- Granice między przedziałami to poziomy decyzyjne, jest ich N-1
- Wartość wejściowa należąca do danego przedziału jest zastępowana przez tzw. poziom reprezentacji - może być to np.:
  - górna wartość przedziału
  - dolna wartość przedziału
  - wartość środka przedziału

Kwantyzacja

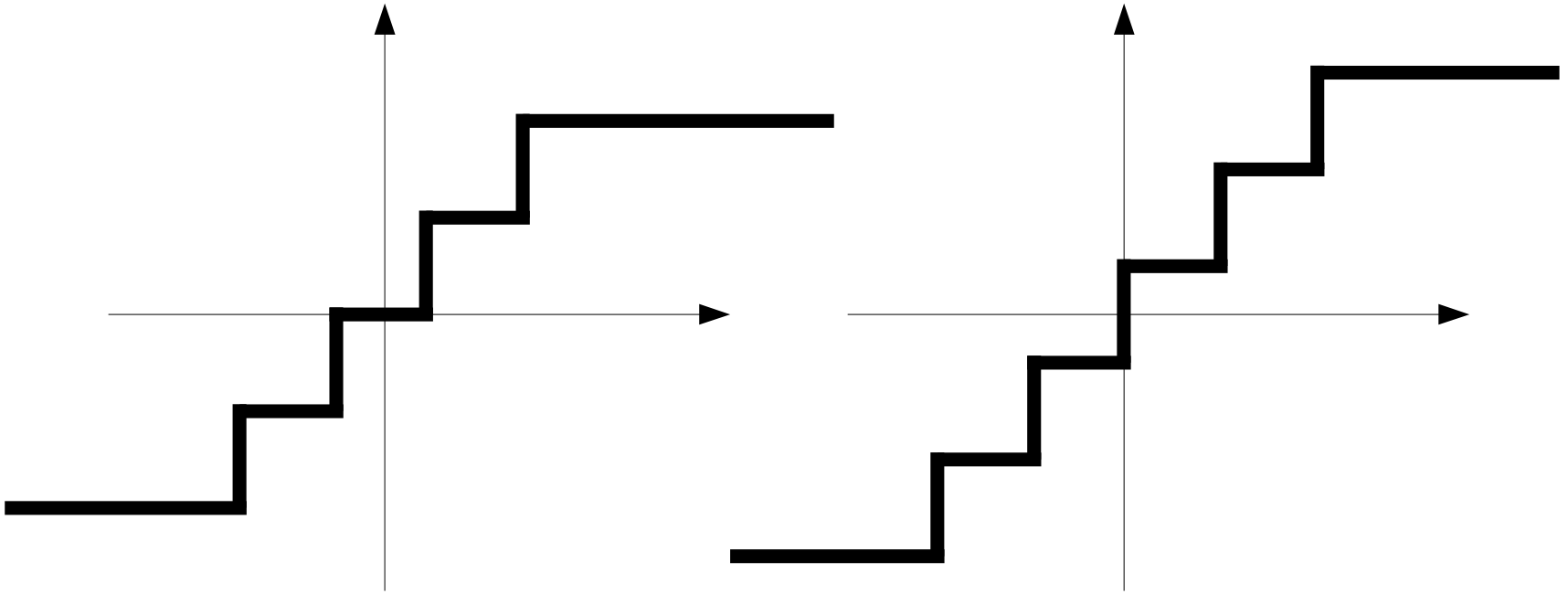
- Typowym przykładem jest kwantyzacja liniowa
- Może być opisana równaniem

$$Q(x)=g(floor(f(x)))$$

- Wartość $i=floor(f(x))$ jest nazywana indeksem kwantyzacji i to właśnie ona jest zwykle wynikiem procesu kwantyzacji

Kwantyzacja

- Typowymi przykładami kwantyzacji liniowej jest:
  - kwantyzacja stała w zerze
  - kwantyzacja ze skokiem w zerze



Kwantyzacja

- Przyjmując reprezentację na M bitach wartości wejściowej z przedziału od -1 do 1 otrzymujemy wzory na kwantyzację:
  - stałą w zerze $$Q(x)=\frac{floor\left(2^{M-1}x+0.5\right)}{2^{M-1}}$$
  - ze skokiem w zerze $$Q(x)=\frac{floor\left(2^{M-1}x\right)+0.5}{2^{M-1}}$$

Próbkowanie i kwantyzacja w praktyce

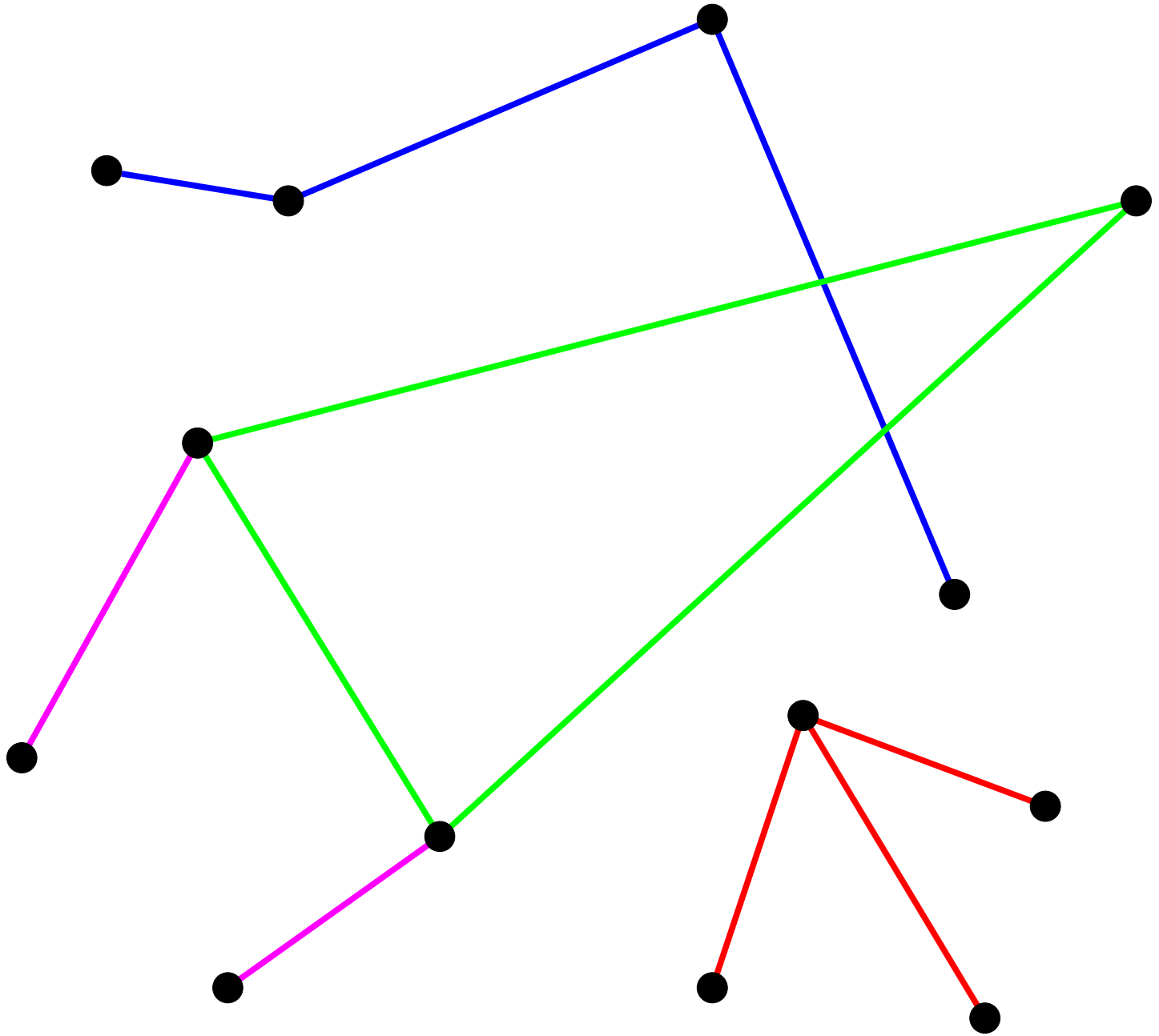
- W praktyce stosuje się przetworniki analogowo-cyfrowe
- Realizacje praktycne odbiegają od założeń teoretycznych:
  - brak idealnych filtrów antyaliasingowych
  - niezerowy czas próbkowania
  - niestałość okresu (częstotliwości) próbkowania
  - ograniczone szybkości narastania
  - nieliniowości układów

Droga

Definicja 1: Droga

Zbiór krawędzi b_1, b_2, ..., b_n w grafie niezorientowanym G_n jest nazywany **drogą** między wierzchołkami V_j i V_k jeżeli krawędzie te mogą być uporządkowane w ten sposób, że:

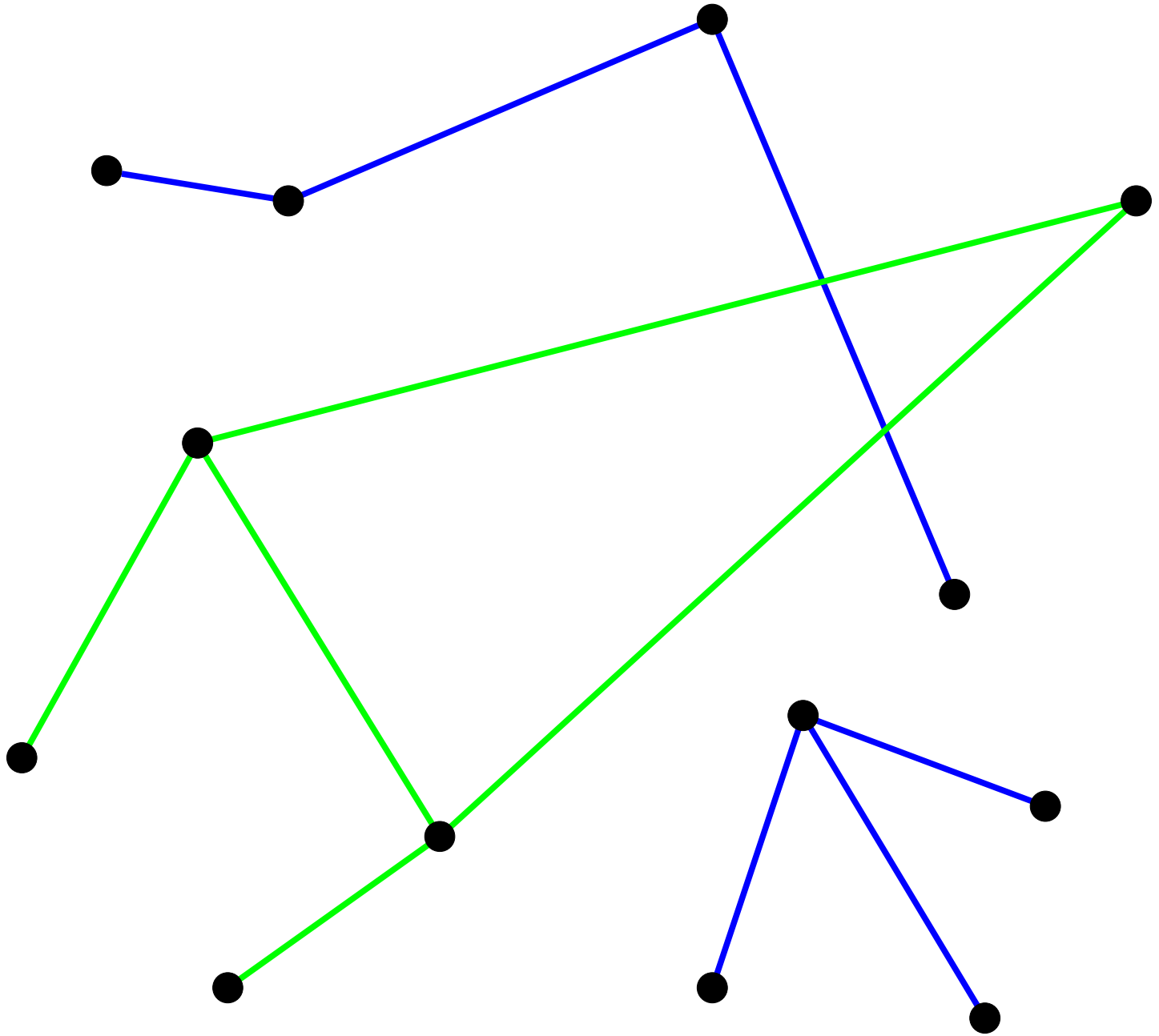
- krawędzie kolejne b_i i b_{i+1} mają zawsze wspólny wierzchołek
- żaden wierzchołek w G_n nie jest wierzchołkiem więcej niż 2 krawędzi zbioru
- V_j jest wierzchołkiem dokładnie jednej krawędzi zbioru oraz V_k jest również wierzchołkiem dokładnie jednej krawędzi zbioru.



Graf spójny

Definicja 2: Graf spójny

Niezorientowany graf G_n jest **grafem spójnym**, jeżeli istnieje droga między dowolnymi dwoma wierzchołkami grafu. Sieć N lub graf zorientowany G_d są spójne, jeżeli związany z nimi graf G_n jest spójny.

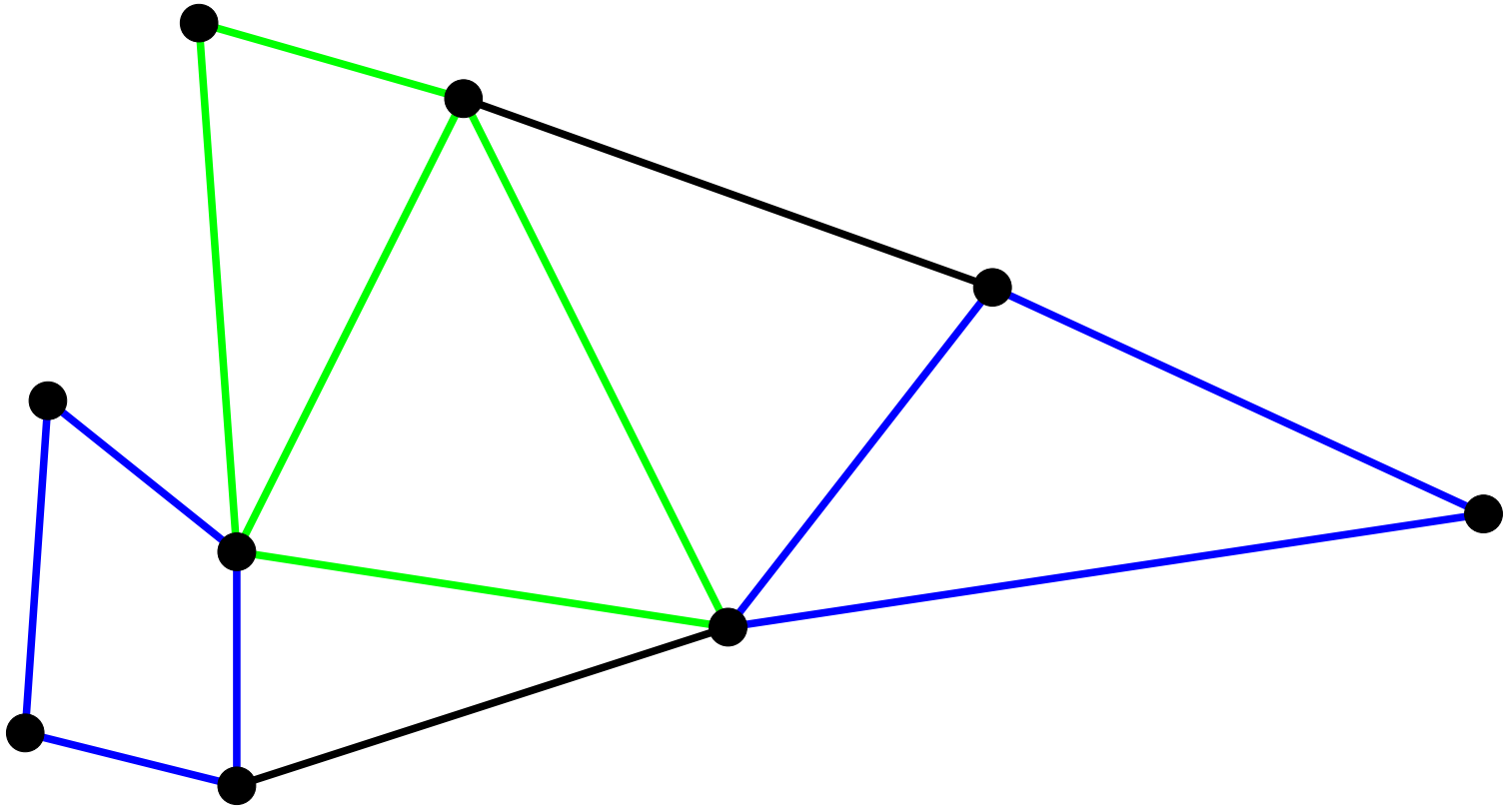


Cykl

Definicja 3: Cykl (obwód)

Podgraf G_s grafu G_n jest nazywany **cyklem**, jeżeli:

- G_s jest spójny
- każdy wierzchołek w G_s ma dokładnie dwie powiązane z nim krawędzie



Drzewo

Definicja 4: Drzewo

Podgraf G_s grafu spójnego G_n jest nazywany **drzewem** T, jeżeli:

- G_s jest spójny
- G_s zawiera wszystkie wierzchołki G_n
- G_s nie zawiera cykli

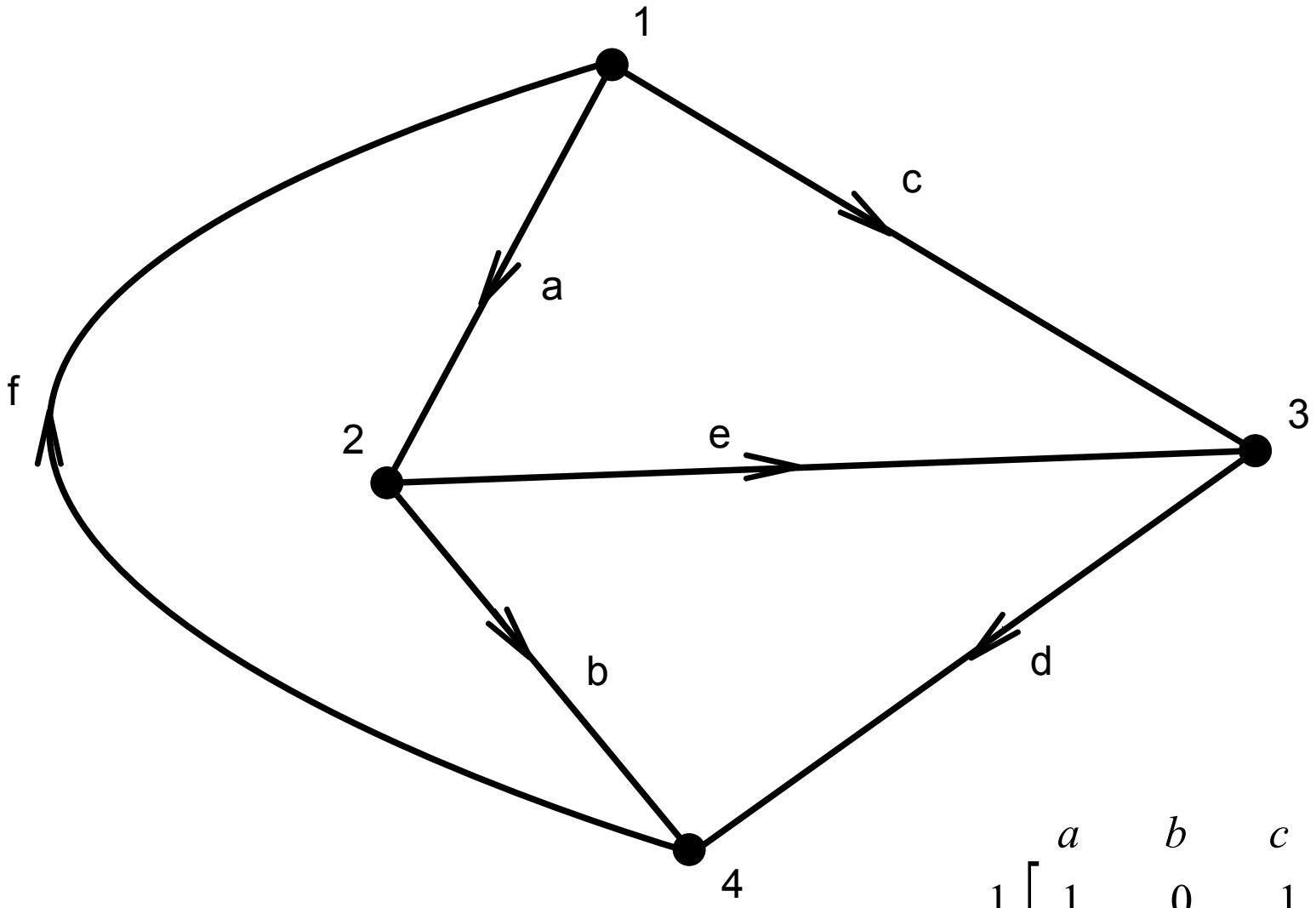
- Krawędzie które należą do drzew T nazywane są krawędziami drzewa, pozostałe - krawędziami przeciwdrzewa (cięciwami)
- Wniosek: jeśli G_n zawiera n wierzchołków, to drzewo zawiera dokładnie n-1 krawędzi.

Macierz incydencji

Definicja 5: Macierz incydencji $\mathbb{A}_a$

Dla grafu zorientowanego G_d zawierającego n wierzchołków i b krawędzi definiujemy **macierz incydencji** jako macierz o wymiarze n x b $\mathbb{A}_a=[a_{ij}]$, przy czym:

- a_{ij} = 1 jeżeli krawędź j jest połączona (incydentna) z i-tym wierzchołkiem i skierowana od wierzchołka
- a_{ij} = -1 jeżeli krawędź j jest połączona (incydentna) z i-tym wierzchołkiem i skierowana do wierzchołka
- a_{ij} = 0 jeżeli krawędź j nie jest jest połączona (incydentna) z i-tym wierzchołkiem.



$$\mathbb{A}a = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{cccccc} a & b & c & d & e & f \end{array} \\ \begin{bmatrix} 1 & 0 & 1 & 0 & 0 & -1 \\ -1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & -1 & 1 & -1 & 0 \\ 0 & -1 & 0 & -1 & 0 & 1 \end{bmatrix} \end{array}$$

I prawo Kirchhoffa

- $\mathbb{i}$ - wektor prądów o wymiarze b x 1, kolejność prądów w tym wektorze odpowiada kolejności kolumn w macierzy $\mathbb{A}_a$
- I prawo Kirchhoffa w odniesieniu do wszystkich węzłów ma postać $\mathbb{A}_a \cdot \mathbb{i} = \mathbb{0}$.
- Ten układ równań jest liniowo zależny, każde równanie wynika z n-1 pozostałych równań.

Twierdzenie 1

- Maksymalny układ niezależnych równań wynikających z I prawa Kirchhoffa dla węzłów spójnej sieci N może być przedstawiony w postaci $\mathbb{A} \cdot \mathbb{i} = \mathbb{0}$, gdzie $\mathbb{A}$ to **zredukowana** macierz incydencji

$$
\begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array}
\begin{array}{c}
\begin{array}{cccccc} a & b & c & d & e & f \end{array} \\
\begin{bmatrix}
1 & 0 & 1 & 0 & 0 & -1 \\
-1 & 1 & 0 & 0 & 1 & 0 \\
0 & 0 & -1 & 1 & -1 & 0 \\
0 & -1 & 0 & -1 & 0 & 1
\end{bmatrix}
\end{array}
\cdot
\begin{bmatrix} i_a \\ i_b \\ i_c \\ i_d \\ i_e \\ i_f \end{bmatrix} = 0
$$

Transformacja węzłowa

Wybierając n-ty wierzchołek grafu jako wierzchołek odniesienia, można zapisać wektor **potencjałów węzłowych** jako:

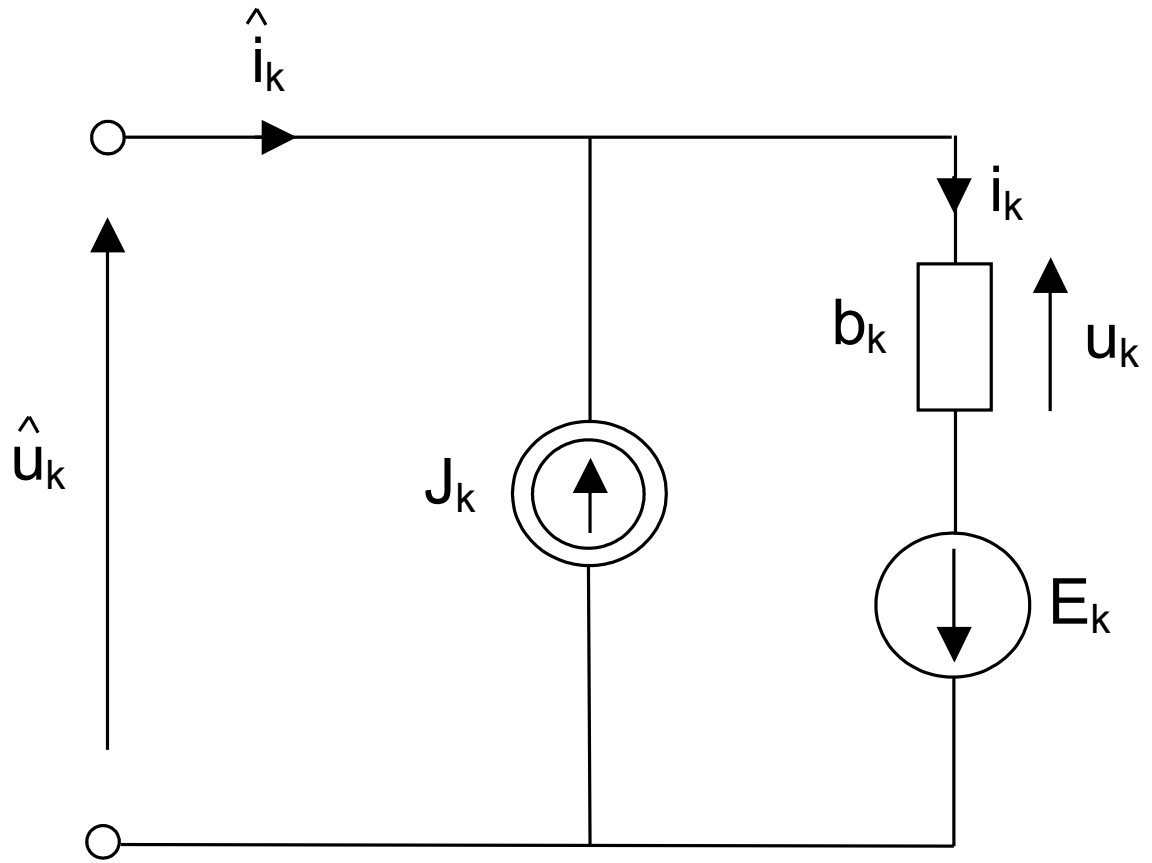
$u_n = [u_{1n}\ u_{2n}\ ...\ u_{(n-1)n}]^T$

wówczas $u = A^T u_n$ - zależność łącząca napięcia gałęziowe i potencjały węzłowe.

Transformacja węzłowa

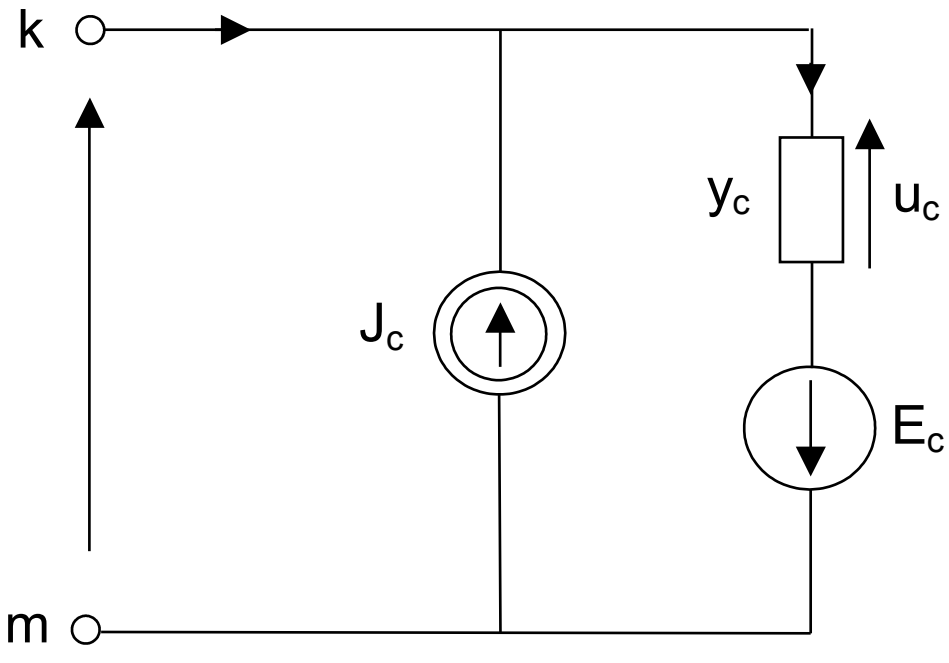
$$A^{T} \cdot u_n = \begin{bmatrix} 1 & -1 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & -1 \\ 0 & 0 & 1 \\ 0 & 1 & -1 \\ -1 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} u_{1n} \\ u_{2n} \\ u_{3n} \end{bmatrix} = \begin{bmatrix} u_{1n} - u_{2n} \\ u_{2n} \\ u_{1n} - u_{3n} \\ u_{3n} \\ u_{2n} - u_{3n} \\ -u_{1n} \end{bmatrix} = \begin{bmatrix} u_a \\ u_b \\ u_c \\ u_d \\ u_e \\ u_f \end{bmatrix}$$

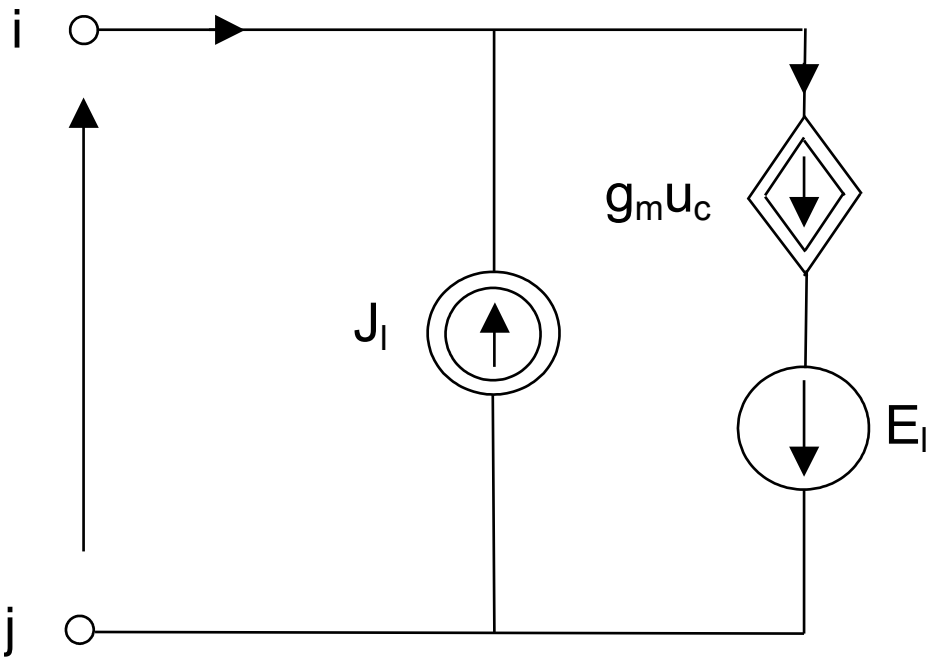
Metoda potencjałów węzłowych



***k*-ta gałąź**

Metoda potencjałów węzłowych





Metoda potencjałów węzłowych

$$\hat{\mathbf{u}} = \begin{bmatrix} \hat{u}_1 & \hat{u}_2 & \cdots & \hat{u}_b \end{bmatrix}^T$$

$$\hat{\mathbf{i}} = \begin{bmatrix} \hat{i}_1 & \hat{i}_2 & \cdots & \hat{i}_b \end{bmatrix}^T$$

$$\mathbf{E} = \begin{bmatrix} E_1 & E_2 & \cdots & E_b \end{bmatrix}^T$$

$$\mathbf{J} = \begin{bmatrix} J_1 & J_2 & \cdots & J_b \end{bmatrix}^T$$

$$\mathbf{u} = \begin{bmatrix} u_1 & u_2 & \cdots & u_b \end{bmatrix}^T$$

$$\mathbf{i} = \begin{bmatrix} i_1 & i_2 & \cdots & i_b \end{bmatrix}^T$$

Metoda potencjałów węzłowych

$$\hat{\mathbf{u}} = \mathbf{u} - \mathbf{E} \qquad (1)$$

$$\hat{\mathbf{i}} = \mathbf{i} - \mathbf{J} \qquad (2)$$

$$\mathbf{A}\hat{\mathbf{i}} = \mathbf{0} \qquad (3) \qquad \left(\mathbf{A} = [a_{ij}]_{n\,x\,b}\right)$$

$$\mathbf{A}(\mathbf{i} - \mathbf{J}) = \mathbf{0} \qquad \mathbf{Ai} = \mathbf{AJ} \qquad (4)$$

Metoda potencjałów węzłowych - opis elementów

$$i_k = \frac{1}{R_k} u_j \qquad R_k \neq 0$$

$$i_k = g_{kj} u_j$$

Metoda potencjałów węzłowych - opis elementów

$$\begin{bmatrix} i_1 \\ i_2 \\ \vdots \\ i_b \end{bmatrix} = \begin{bmatrix} y_{11} & y_{12} & \cdots & y_{1b} \\ y_{21} & y_{22} & \cdots & y_{2b} \\ \vdots & \vdots & \vdots & \vdots \\ y_{b1} & y_{b2} & \cdots & y_{bb} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_b \end{bmatrix}$$

Rezystancje:

$$y_{k\alpha} = \begin{cases} 0 & dla \quad \alpha \neq k \\ \dfrac{1}{R_k} & dla \quad \alpha = k \end{cases}$$

Źródła sterowane:

$$y_{k\alpha} = \begin{cases} 0 & dla \quad \alpha \neq j \\ g_{kj} & dla \quad \alpha = j \end{cases}$$

Metoda potencjałów węzłowych - równanie

$$\mathbf{i} = \mathbf{Y}_b \mathbf{u} \qquad (5)$$

$$\mathbf{A}\mathbf{Y}_b \mathbf{u} = \mathbf{A}\mathbf{J}$$

ponieważ $\hat{\mathbf{u}} = \mathbf{u} - \mathbf{E}$

$$\mathbf{A}\mathbf{Y}_b \left(\hat{\mathbf{u}} + \mathbf{E} \right) = \mathbf{A}\mathbf{J}$$

czyli $$\mathbf{A}\mathbf{Y}_b \hat{\mathbf{u}} = \mathbf{A}\left(\mathbf{J} - \mathbf{Y}_b \mathbf{E} \right) \qquad (6)$$

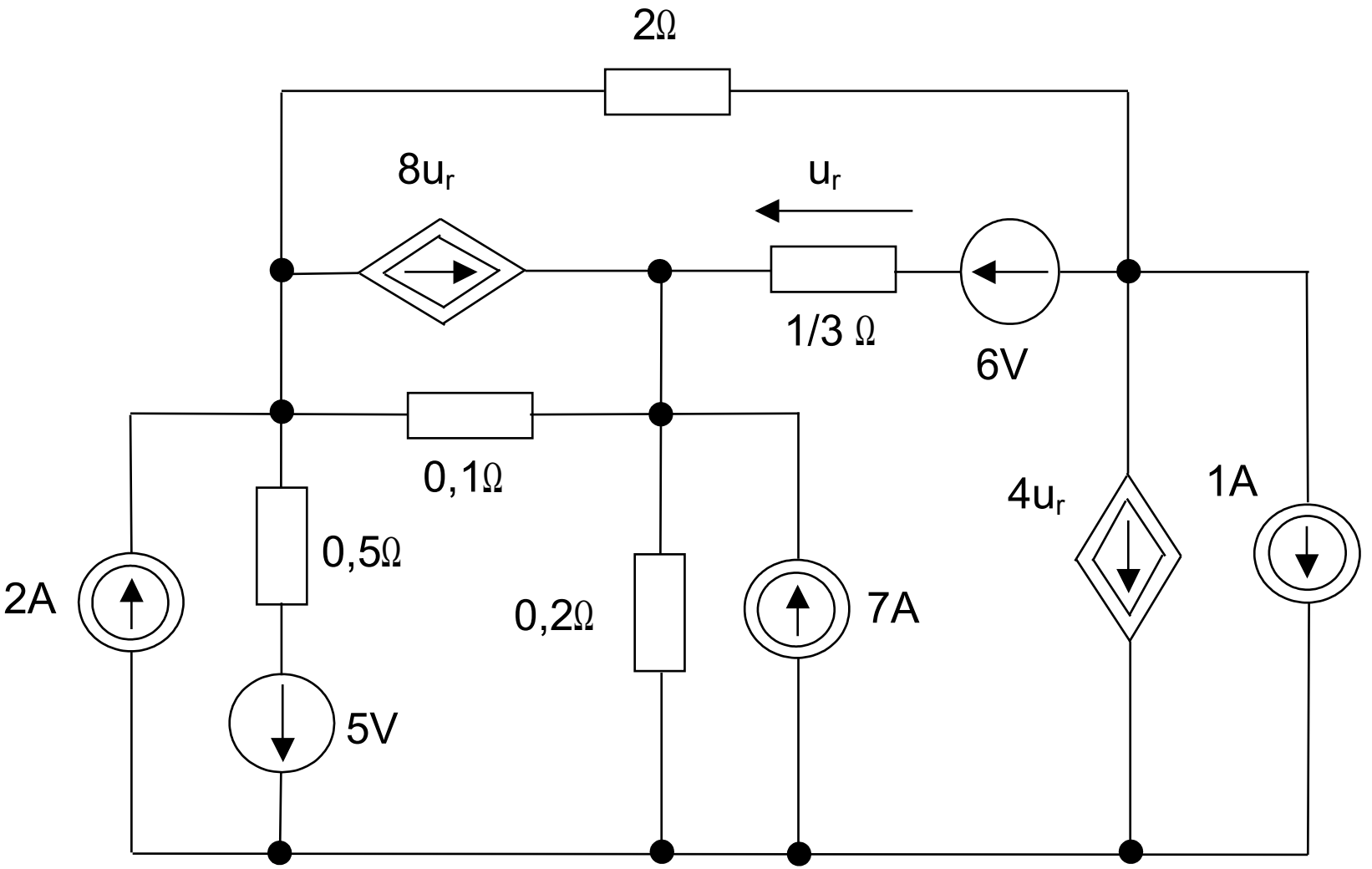
Metoda potencjałów węzłowych - równanie

Transformacja węzłowa: $\hat{\mathbf{u}} = \mathbf{A}^{\mathrm{T}}\mathbf{u}_{\mathbf{n}}$

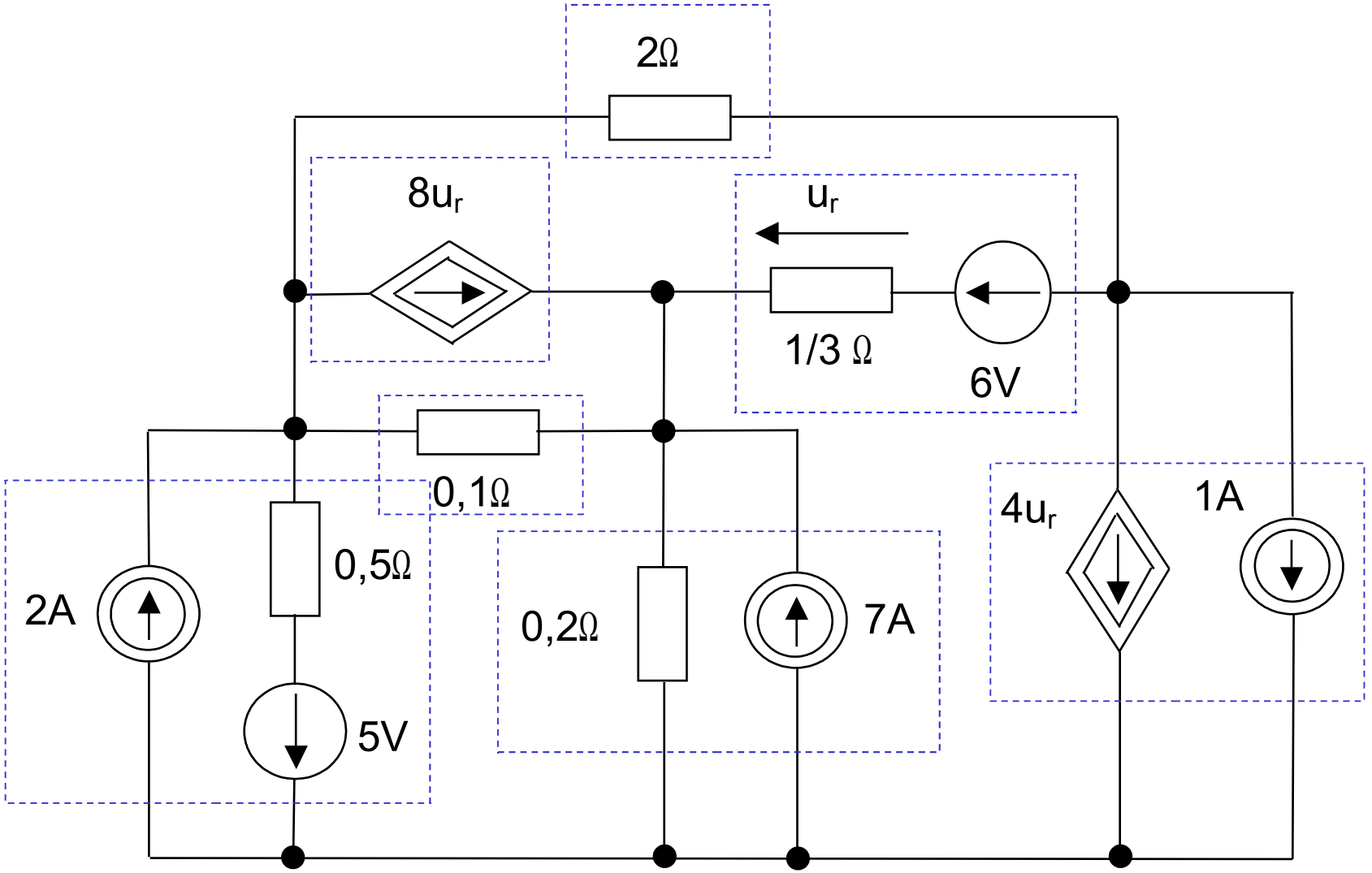
$$\left(\mathbf{A}\mathbf{Y}_{\mathbf{b}}\mathbf{A}^{\mathrm{T}}\right)\mathbf{u}_{\mathbf{n}} = \mathbf{A}\left(\mathbf{J} - \mathbf{Y}_{\mathbf{b}}\mathbf{E}\right) \qquad (7)$$

$$\mathbf{Y}_{\mathbf{n}}\mathbf{u}_{\mathbf{n}} = \mathbf{J}_{\mathbf{n}} \qquad (8)$$

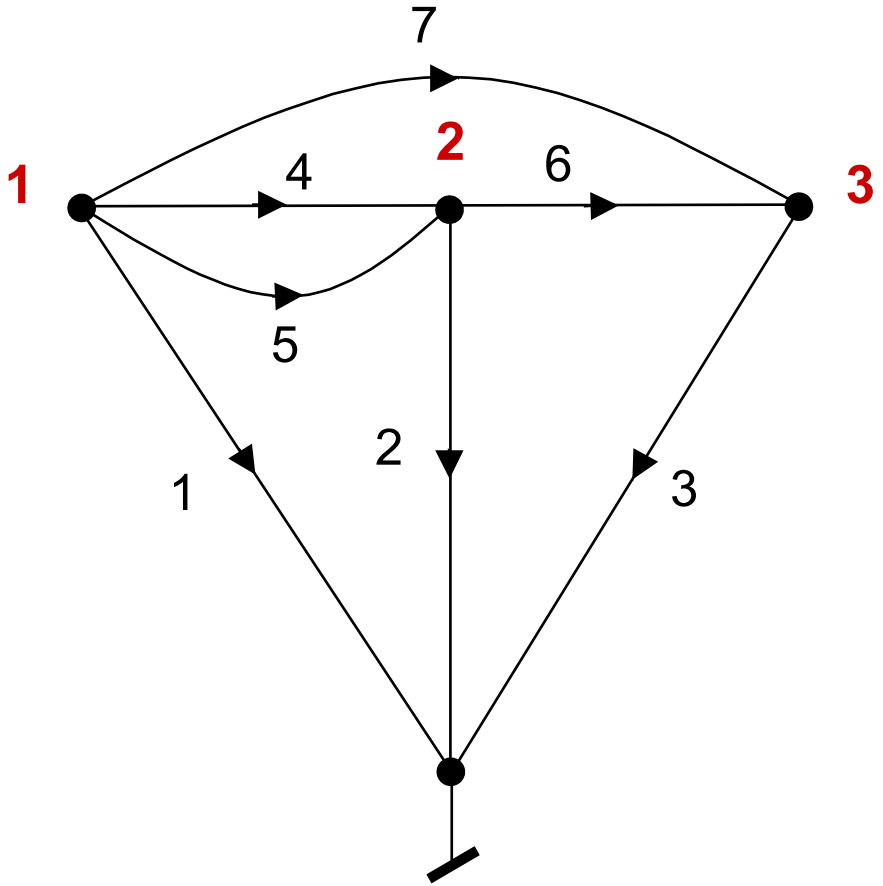
Metoda potencjałów węzłowych - przykład



Metoda potencjałów węzłowych - przykład



Metoda potencjałów węzłowych - przykład



$$\mathbf{A} = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & -1 & -1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & -1 & -1 \end{bmatrix}$$

Metoda potencjałów węzłowych - przykład

$$\mathbf{Y_b} = \begin{bmatrix} 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 5 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & 0 & 0 & 8 & 0 \\ 0 & 0 & 0 & 0 & 10 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0,5 \end{bmatrix}$$

Metoda potencjałów węzłowych - przykład

$$\mathbf{J} = \begin{bmatrix} 2 \\ 7 \\ -1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \qquad \mathbf{E} = \begin{bmatrix} 5 \\ 0 \\ 0 \\ 0 \\ 0 \\ -6 \\ 0 \end{bmatrix}$$

Metoda potencjałów węzłowych - przykład

$$\mathbf{Y_n} = \mathbf{A}\mathbf{Y_b}\mathbf{A}^{\mathbf{T}} = \begin{bmatrix} 12{,}5 & -2 & -8{,}5 \\ -10 & 10 & 5 \\ -0{,}5 & 1 & -0{,}5 \end{bmatrix}$$

$$\mathbf{J_n} = \mathbf{A}\left(\mathbf{J} - \mathbf{Y_b}\mathbf{E}\right) = \begin{bmatrix} 40 \\ -23 \\ 5 \end{bmatrix}$$

Metoda potencjałów węzłowych: sieci liniowe, wymuszenia sinusoidalne

Rezystor *R*	Impedancja *Z*
Konduktancja *G*	Admitancja *Y*
	Cewka: $Z_L = j\omega L$ Kondensator: $Z_C = 1/j\omega C$
Funkcja czasu *u(t)*	Wskaz *U(ω)*
Funkcja czasu *i(t)*	Wskaz *I(ω)*
R, G, u, i są liczbami rzeczywistymi	*Z, Y, V, I* są liczbami zespolonymi

MPW: sieci liniowe, wymuszenia sinusoidalne

$$\left(\mathbf{A}\mathbf{Y}_{\mathbf{b}}\mathbf{A}^{\mathbf{T}}\right)\mathbf{U}_{\mathbf{n}} = \mathbf{A}\left(\mathbf{J} - \mathbf{Y}_{\mathbf{b}}\mathbf{E}\right)$$

$$\mathbf{Y}_{\mathbf{n}}\mathbf{U}_{\mathbf{n}} = \mathbf{J}_{\mathbf{n}}$$

Indukcyjności sprzężone

$$\mathbf{U_L} = j\omega\, \mathbf{L}\,\mathbf{I_L}$$

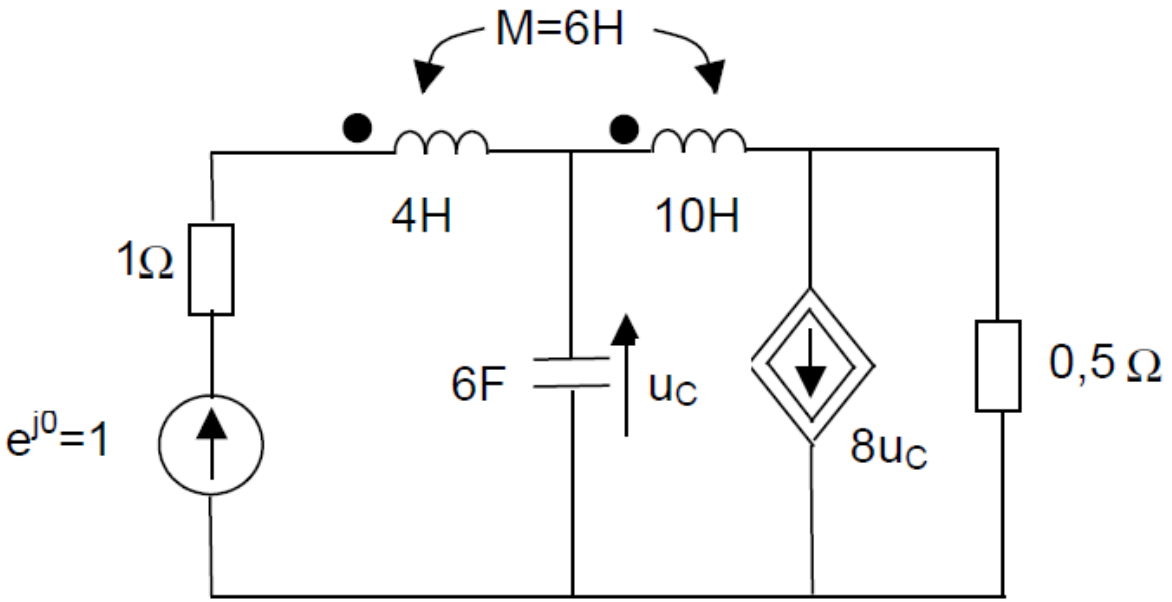
$$k = \frac{|M|}{\sqrt{L_1 L_2}} \leq 1$$

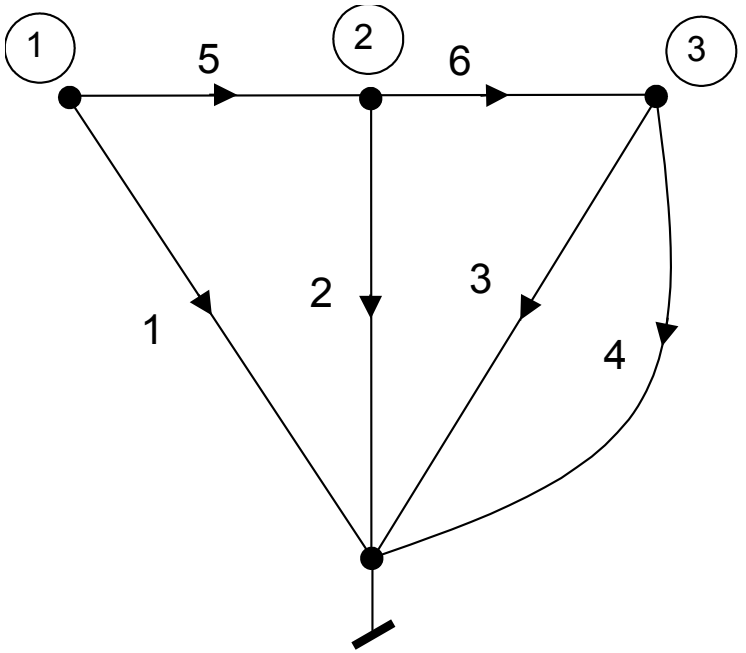
$$u_{L1} = L_1 \frac{di_1}{dt}, \quad u_{L2} = L_2 \frac{di_2}{dt}, \quad u_{M1} = M \frac{di_2}{dt}, \quad u_{M2} = M \frac{di_1}{dt}$$

$$\mathbf{I_L} = \frac{1}{j\omega} \boldsymbol{\Gamma} \mathbf{U_L}$$

$$\boldsymbol{\Gamma} = \mathbf{L}^{-1}$$

MPW: wymuszenia sinusoidalne - przykład





$$Y_C = j\omega C, \qquad Y_L = \frac{1}{j\omega L}$$

MPW: wymuszenia sinusoidalne - przykład

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & -1 & 1 \\ 0 & 0 & 1 & 1 & 0 & -1 \end{bmatrix}$$

$$\mathbf{\Gamma} = \mathbf{L}^{-1} = \begin{matrix} \\ 5 \\ 6 \end{matrix} \begin{matrix} \begin{matrix} 5 & 6 \end{matrix} \\ \begin{bmatrix} 4 & 6 \\ 6 & 10 \end{bmatrix}^{-1} \end{matrix} = \begin{matrix} \\ 5 \\ 6 \end{matrix} \begin{matrix} \begin{matrix} 5 & 6 \end{matrix} \\ \begin{bmatrix} 2{,}5 & -1{,}5 \\ -1{,}5 & 1 \end{bmatrix} \end{matrix}$$

$$\omega = 0{,}5\ rad/s$$

$$\frac{1}{j\omega\,\mathbf{L}} = -2j\mathbf{\Gamma}$$

MPW: wymuszenia sinusoidalne - przykład

$$\mathbf{Y_b} = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & j3 & 0 & 0 & 0 & 0 \\ 0 & 8 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & -j5 & j3 \\ 0 & 0 & 0 & 0 & j3 & -j2 \end{bmatrix} \end{array} \qquad \mathbf{J} = \mathbf{0} \qquad \mathbf{E} = \begin{bmatrix} -1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

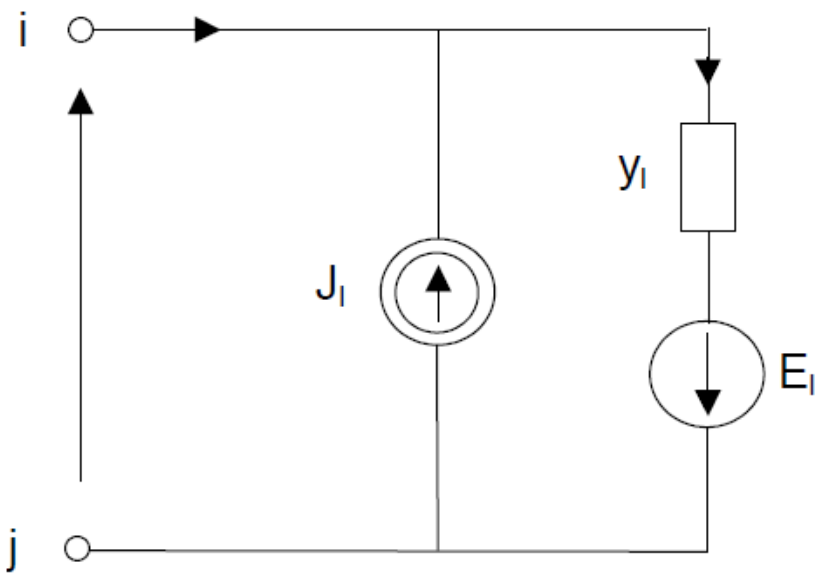
$$\mathbf{Y_n} = \begin{bmatrix} 1-j5 & j8 & -j3 \\ j8 & -j10 & j5 \\ -j3 & 8+j5 & 2-j2 \end{bmatrix} \qquad \mathbf{J_n} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

Bezpośrednie wyznaczanie równania węzłowego

$$\mathbf{Y}_n = \mathbf{A}\mathbf{Y}_b\mathbf{A}^T$$

$$\mathbf{J}_n = \mathbf{A}\left(\mathbf{J} - \mathbf{Y}_b\mathbf{E}\right)$$

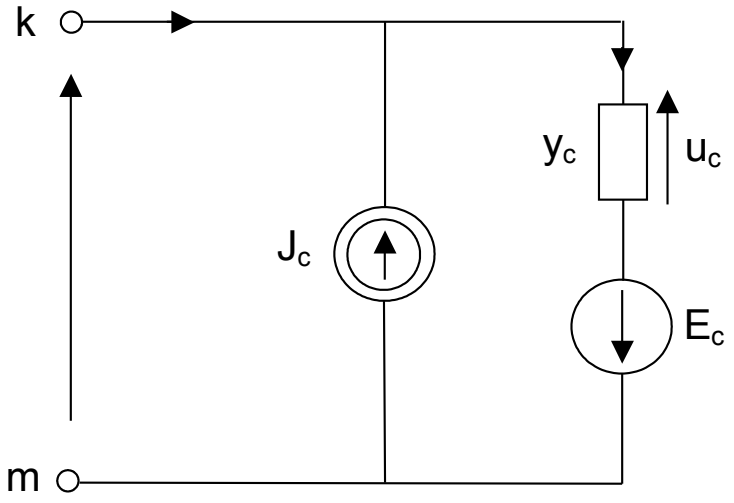
Bezpośrednie wyznaczanie równania węzłowego

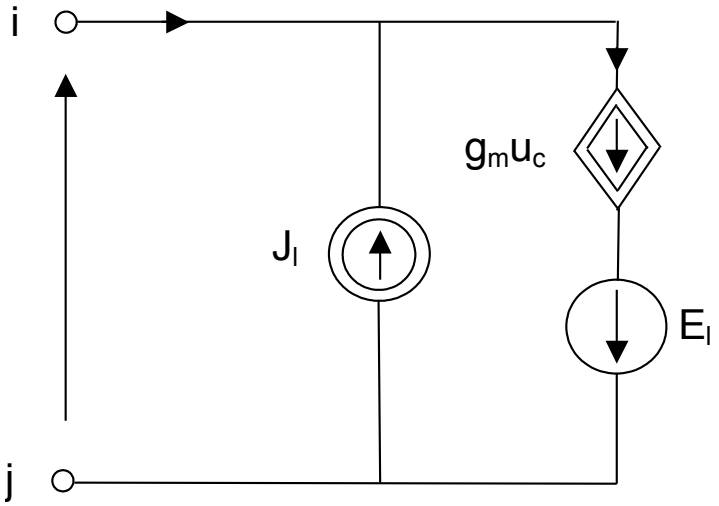


$$\mathbf{Y_n} = \begin{matrix} \\ i \\ j \end{matrix} \begin{matrix} \begin{matrix} i & \quad j \end{matrix} \\ \begin{bmatrix} y_l & -y_l \\ -y_l & y_l \end{bmatrix} \end{matrix}$$

$$\mathbf{J_n} = \begin{matrix} i \\ j \end{matrix} \begin{bmatrix} (J_l - y_l E_l) \\ -(J_l - y_l E_l) \end{bmatrix}$$

Bezpośrednie wyznaczanie równania węzłowego

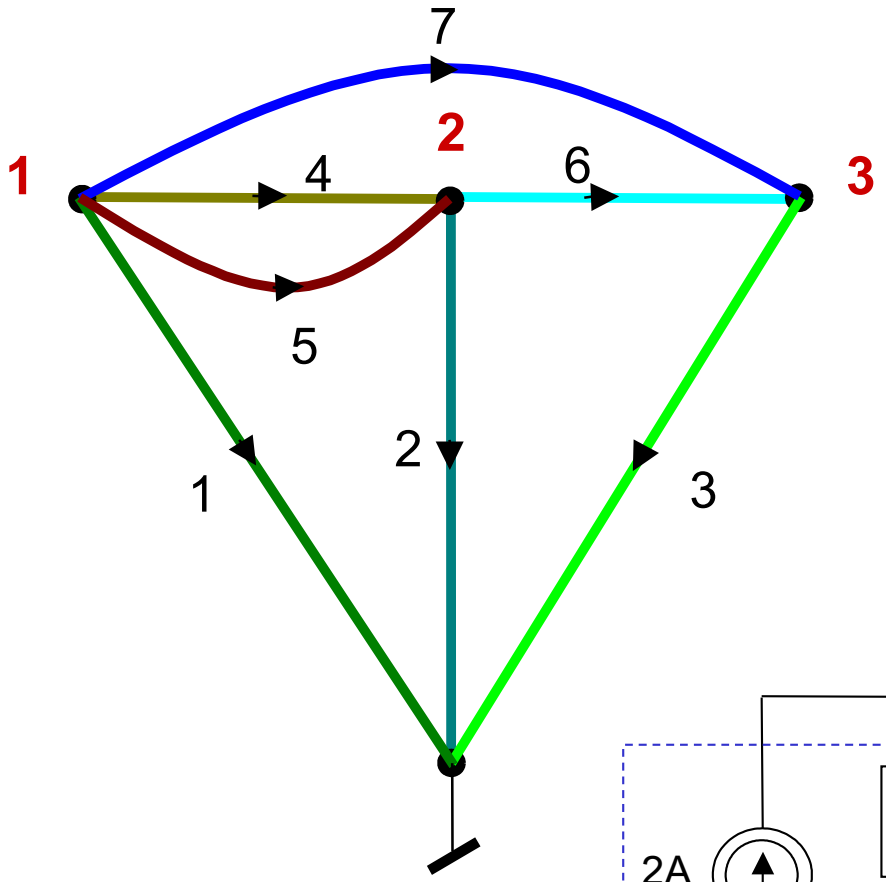


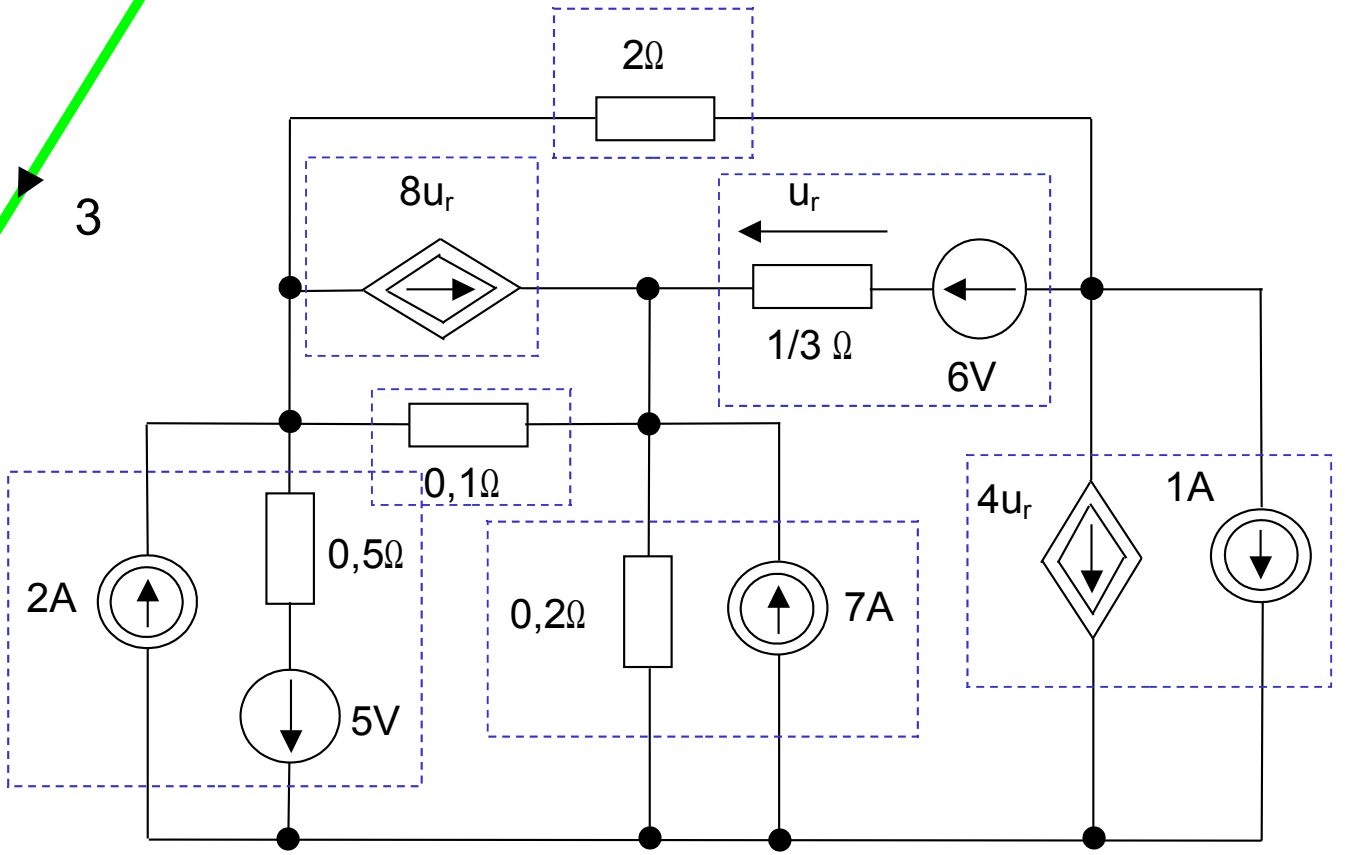


$$\mathbf{Y_n} = \begin{array}{c} \\ \text{od } i \\ \text{do } j \end{array} \begin{array}{c} \begin{array}{cc} \text{od } k & \text{do } m \end{array} \\ \begin{bmatrix} g_m & -g_m \\ -g_m & g_m \end{bmatrix} \end{array}$$

$$\mathbf{J_n} = \begin{array}{c} i \\ j \end{array} \begin{bmatrix} \left(J_l - g_m E_c\right) \\ -\left(J_l - g_m E_c\right) \end{bmatrix}$$

Metoda potencjałów węzłowych - przykład



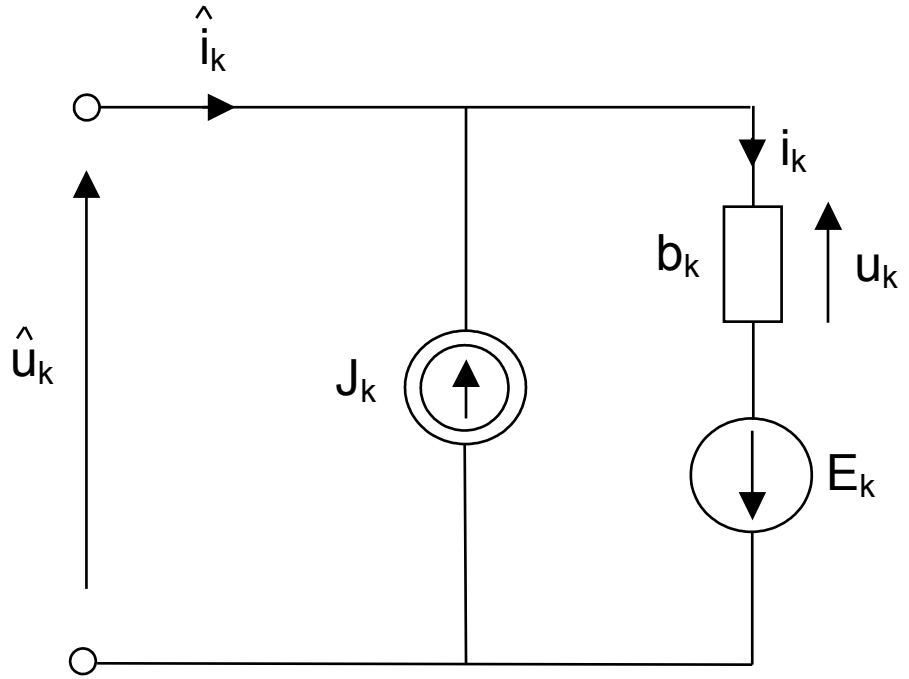


Bezpośrednie wyznaczanie równania węzłowego - przykład

$$Y_N = \begin{matrix} 2+10+0{,}5 & -10+8 & -0{,}5-8 \\ -10 & 5+10+3-8 & -3+8 \\ -0{,}5 & -3+4 & 3+0{,}5-4 \end{matrix}$$

$$J_N = \begin{matrix} 2-10+48 \\ 7+18-48 \\ -1+24-18 \end{matrix}$$

Analiza sieci nieliniowych metodą potencjałów węzłowych



$$i_k = g_k\left(u_k\right)$$

$$i_k = g_k\left(u_j\right)$$

Analiza sieci nieliniowych metodą potencjałów węzłowych

$$\mathbf{i} = \begin{bmatrix} i_1 \\ i_2 \\ \vdots \\ i_b \end{bmatrix} = \begin{bmatrix} g_1(u_\alpha) \\ g_1(u_\beta) \\ \vdots \\ g_1(u_\zeta) \end{bmatrix} = \mathbf{g}(\mathbf{u})$$

$u_\alpha , u_\alpha \cdots$ **dowolne napięcie gałęziowe** $u_1, u_2, \cdots u_b$

Jeżeli nie ma źródeł sterowanych:

$\alpha = 1, \quad \beta = 2, \quad \cdots \quad \zeta = b$

Analiza sieci nieliniowych metodą potencjałów węzłowych

$$\mathbf{A}\hat{\mathbf{i}} = \mathbf{0} \qquad \hat{\mathbf{i}} = \mathbf{i} - \mathbf{J} \qquad \mathbf{Ai} = \mathbf{AJ}$$

$$\mathbf{Ag}(\mathbf{u}) = \mathbf{AJ} \qquad \hat{\mathbf{u}} = \mathbf{u} - \mathbf{E}$$

$$\mathbf{Ag} \circ \left(\hat{\mathbf{u}} + \mathbf{E} \right) = \mathbf{AJ} \qquad \hat{\mathbf{u}} = \mathbf{A}^{\mathrm{T}}\mathbf{u}_{\mathbf{n}}$$

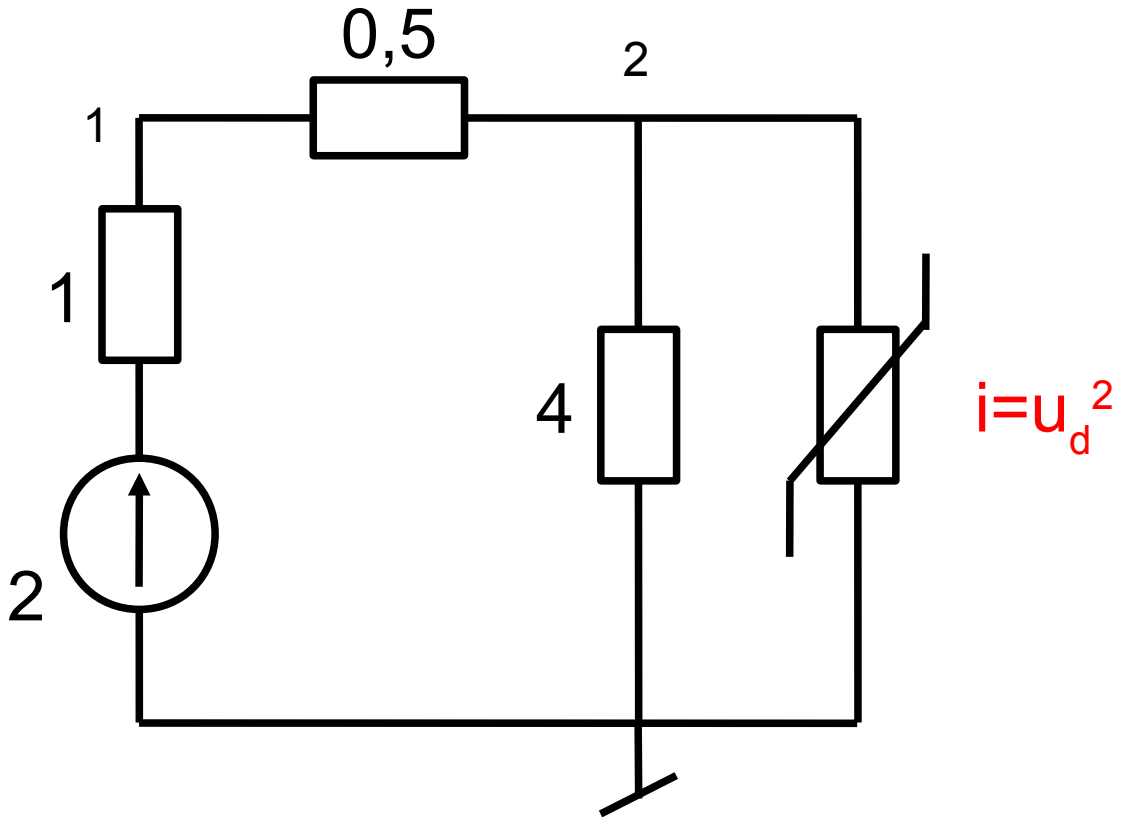
$$\mathbf{Ag} \circ \left(\mathbf{A}^{\mathrm{T}}\mathbf{u}_{\mathbf{n}} + \mathbf{E} \right) = \mathbf{AJ}$$

Analiza sieci nieliniowych metodą potencjałów węzłowych

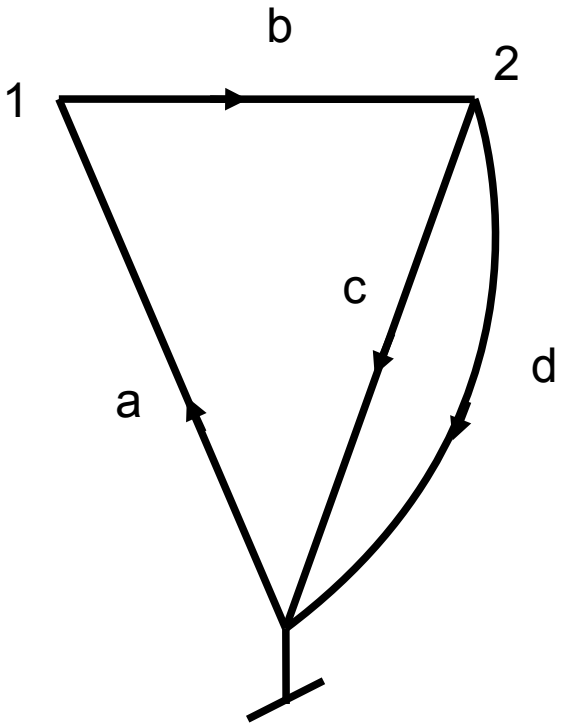
$$\mathbf{f}(\mathbf{x}) = \mathbf{0}$$

$$\mathbf{f}(\mathbf{x}) = \mathbf{f}(\mathbf{u_n}) = \mathbf{Ag} \circ (\mathbf{A}^{\mathbf{T}}\mathbf{u_n} + \mathbf{E}) - \mathbf{AJ}$$

Sieci nieliniowe - przykład



$i=u_d^2$



Sieci nieliniowe - przykład

$$\mathbf{A}= \begin{array}{c} \\ 1 \\ 2 \end{array} \begin{array}{c} \begin{array}{cccc} a & b & c & d \end{array} \\ \begin{bmatrix} -1 & 1 & 0 & 0 \\ 0 & -1 & 1 & 1 \end{bmatrix} \end{array}$$

$$\mathbf{E}= \begin{array}{c} \\ a \\ b \\ c \\ d \end{array} \begin{array}{c} \begin{array}{c} \end{array} \\ \begin{bmatrix} 2 \\ 0 \\ 0 \\ 0 \end{bmatrix} \end{array}$$

J=0

Sieci nieliniowe - przykład

$$\mathbf{g(u)=} \begin{matrix} a \\ b \\ c \\ d \end{matrix} \begin{bmatrix} 1 \cdot u_a \\ 2 \cdot u_b \\ 0.25 \cdot u_c \\ u_d^{\,2} \end{bmatrix}$$

Sieci nieliniowe - przykład

$$\boldsymbol{A}\,\boldsymbol{g}\circ(\boldsymbol{A}^{\boldsymbol{T}}\boldsymbol{u}_{\boldsymbol{n}}+\boldsymbol{E})-\boldsymbol{A}\,\boldsymbol{J}$$

$$\begin{bmatrix} -1 & 1 & 0 & 0 \\ 0 & -1 & 1 & 1 \end{bmatrix} \boldsymbol{g} \circ \left(\begin{bmatrix} -1 & 0 \\ 1 & -1 \\ 0 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} + \begin{bmatrix} 2 \\ 0 \\ 0 \\ 0 \end{bmatrix} \right) - \mathbf{0}$$

Sieci nieliniowe - przykład

$$\begin{bmatrix} -1 & 1 & 0 & 0 \\ 0 & -1 & 1 & 1 \end{bmatrix} \boldsymbol{g} \circ \left(\begin{bmatrix} -u_1 \\ u_1 - u_2 \\ u_2 \\ u_2 \end{bmatrix} + \begin{bmatrix} 2 \\ 0 \\ 0 \\ 0 \end{bmatrix} \right)$$

$$\begin{bmatrix} -1 & 1 & 0 & 0 \\ 0 & -1 & 1 & 1 \end{bmatrix} \boldsymbol{g} \circ \left(\begin{bmatrix} -u_1 + 2 \\ u_1 - u_2 \\ u_2 \\ u_2 \end{bmatrix} \right)$$

Sieci nieliniowe - przykład

$$\begin{bmatrix} -1 & 1 & 0 & 0 \\ 0 & -1 & 1 & 1 \end{bmatrix} \left(\begin{bmatrix} -u_1 + 2 \\ 2(u_1 - u_2) \\ 0.25\,u_2 \\ {u_2}^2 \end{bmatrix} \right)$$

$$\begin{bmatrix} u_1 - 2 + 2\ u_1 - 2\ u_2 \\ -2\ u_1 + 2\ u_2 + 0.25 u_2 + {u_2}^2 \end{bmatrix} = \begin{bmatrix} 3\ u_1 - 2\ u_2 - 2 \\ -2\ u_1 + 2.25 u_2 + {u_2}^2 \end{bmatrix}$$

Zmodyfikowana metoda potencjałów węzłowych

Idea:

I. Do układu równań węzłowych uzyskanych z prądowego prawa Kirchhoffa dołączamy dodatkowe równania, napisane dla następujących gałęzi:

- **zawierających źródła napięciowe: niezależne i sterowane,**
- **gałęzi w postaci zwarcia,**
- **zawierające elementy uzależnione prądowo (od prądu płynącego przez tą samą lub inną gałąź)**

Zmodyfikowana metoda potencjałów węzłowych

II. Prądy gałęzi z p. I. są traktowane jako dodatkowe zmienne pierwotne na równi z potencjałami węzłowymi

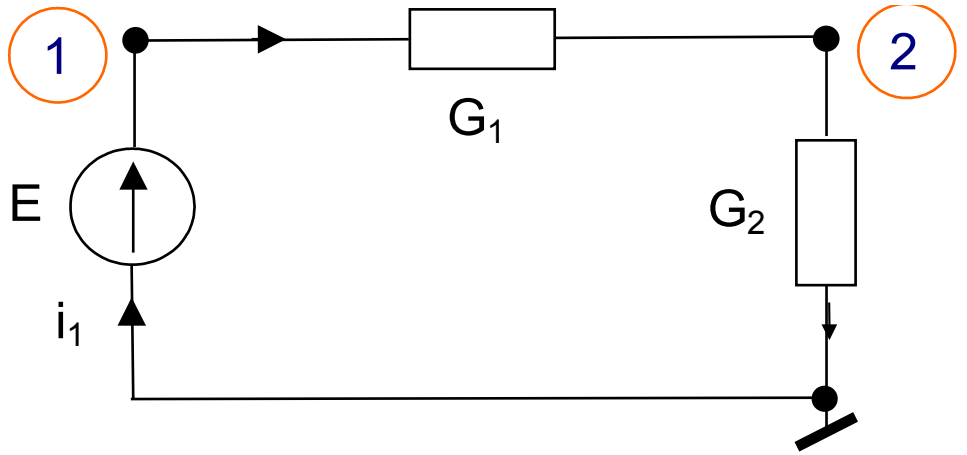
$$\left[\begin{array}{c|c} \mathbf{Y}_n & \mathbf{B} \\ \hline \mathbf{C} & \mathbf{D} \end{array}\right] \left[\begin{array}{c} \mathbf{u}_n \\ \hline \mathbf{i}_d \end{array}\right] = \left[\begin{array}{c} \mathbf{J}_n \\ \hline \mathbf{F}_d \end{array}\right]$$

Zmodyfikowana metoda potencjałów węzłowych

$G_1(u_1-u_2)-i_1=0$

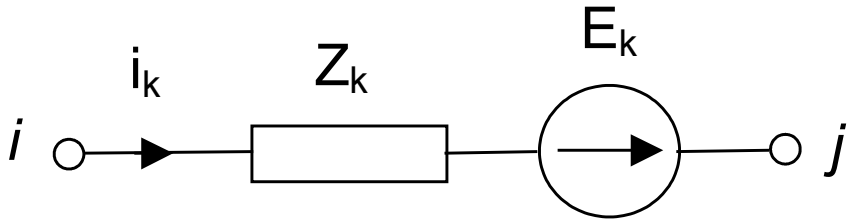
$-G_1(u_1-u_2)+u_2G_2=0$

$u_1 = E$



$$
\begin{array}{c}
 \\
1 \\
2 \\
i_1
\end{array}
\begin{array}{c}
\begin{array}{ccc} 1 & 2 & i_1 \end{array} \\
\left[\begin{array}{cc:c} G_1 & -G_1 & -1 \\ -G_1 & G_1+G_2 & 0 \\ \hdashline -1 & 0 & 0 \end{array}\right]
\end{array}
\left[\begin{array}{c} u_1 \\ u_2 \\ \hdashline i_1 \end{array}\right]
=
\left[\begin{array}{c} 0 \\ 0 \\ \hdashline -E_1 \end{array}\right]
$$

Źródło napięciowe niezależne



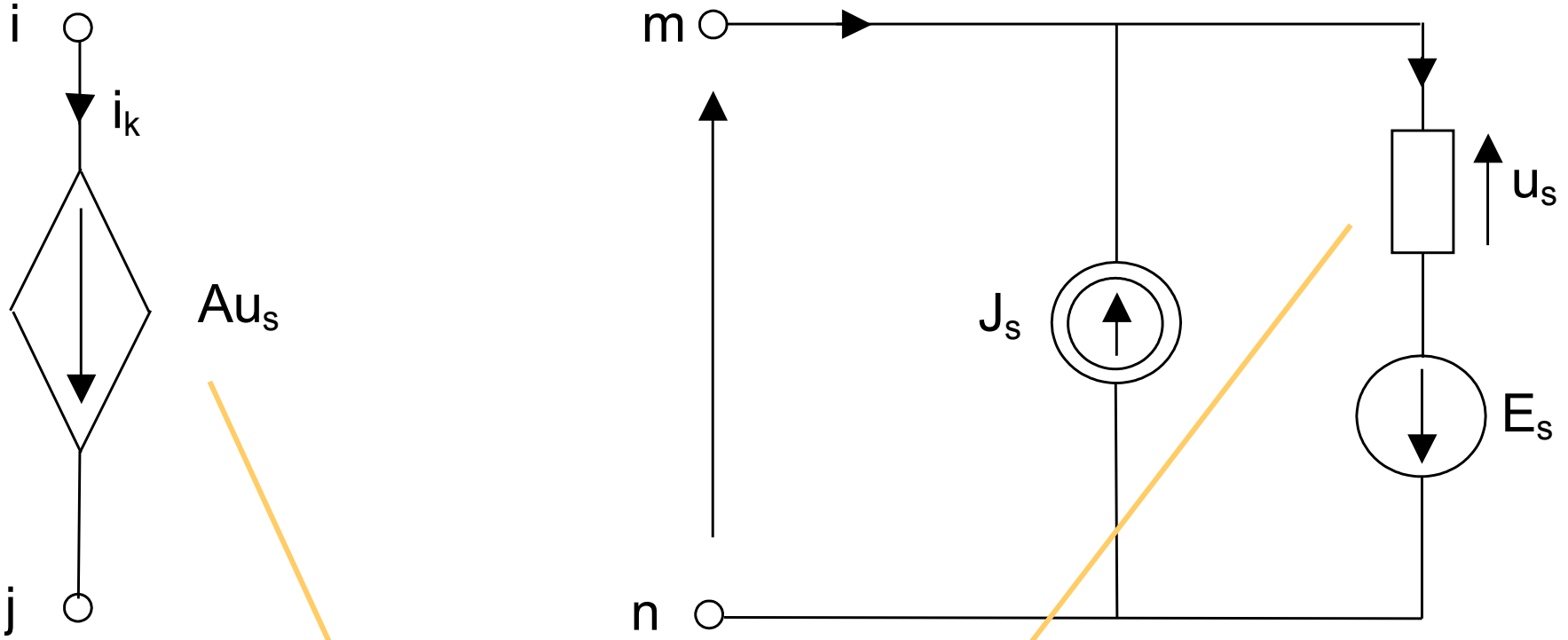
$$u_i - u_j + E_k = Z_k i_k$$

czyli:

$$u_i - u_j - Z_k i_k = -E_k$$

$$\begin{array}{c} \\ i \\ j \\ i_k \end{array} \begin{array}{c} \begin{array}{ccc} i & j & i_k \end{array} \\ \left[\begin{array}{ccc:c} & & & 1 \\ & & & -1 \\ \hdashline 1 & & -1 & -Z_k \end{array}\right] \end{array} \left[\begin{array}{c} u_i \\ u_j \\ \hdashline i_k \end{array}\right] = \left[\begin{array}{c} \\ \\ \hdashline -E_k \end{array}\right]$$

Źródło napięciowe sterowane napięciowo



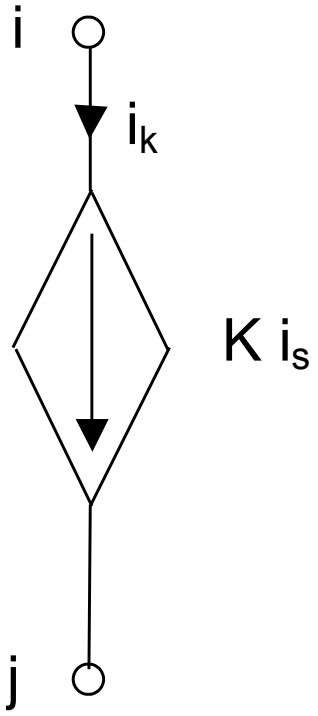
$$u_i - u_j + Au_s = 0 \quad \textbf{ale} \quad u_s = u_m - u_n + E_s$$

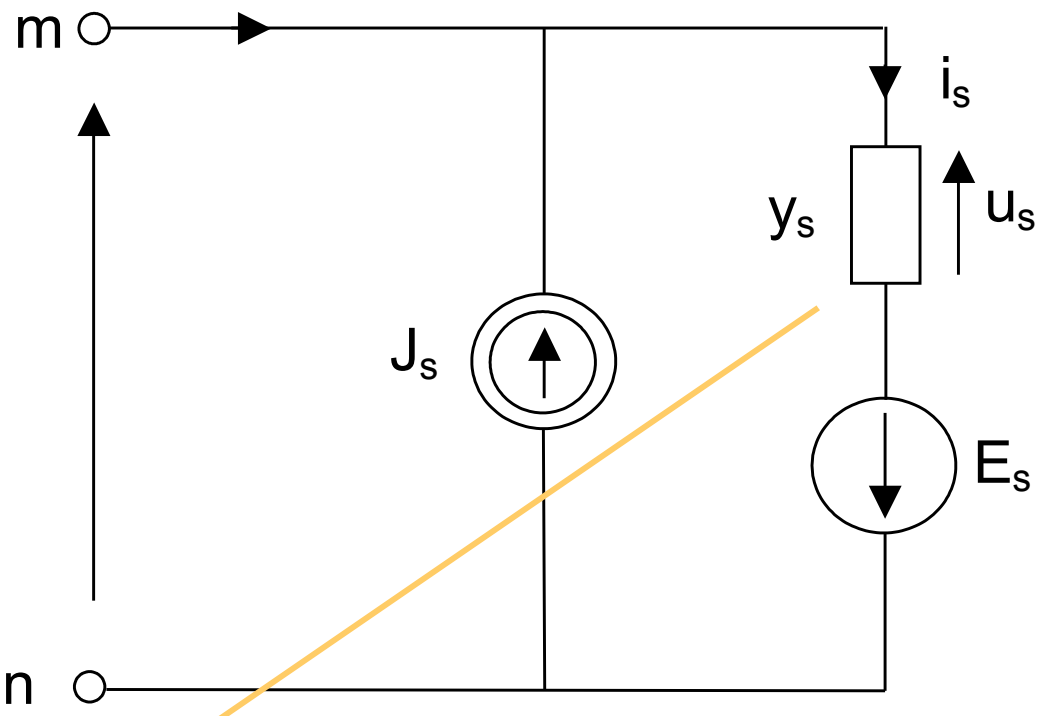
$$\textbf{czyli:} \quad u_i - u_j + Au_m - Au_n = -AE_s$$

Źródło napięciowe sterowane napięciowo

$$
\begin{array}{c}
\\ i \\ j \\ m \\ n \\ i_k
\end{array}
\begin{array}{c}
\begin{array}{cccc} i & j & m & n \end{array} \\
\left[\begin{array}{cccc:c}
 & & & & 1 \\
 & & & & -1 \\
 & & & & \\
 & & & & \\
\hdashline
1 & -1 & A & -A &
\end{array}\right]
\end{array}
\left[\begin{array}{c} u_i \\ u_j \\ u_m \\ u_n \\ \hdashline i_k \end{array}\right]
=
\left[\begin{array}{c} \\ \\ \\ \\ \hdashline -AE_s \end{array}\right]
$$

Źródło napięciowe sterowane prądowo

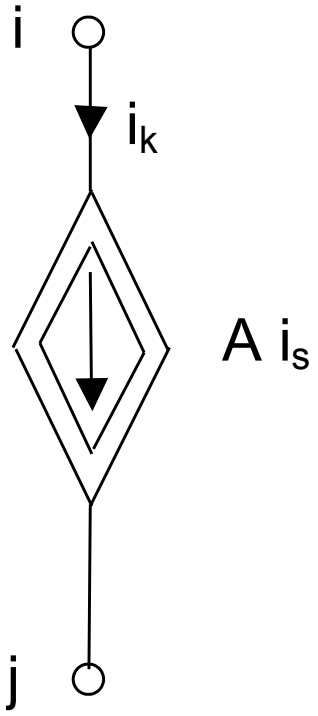


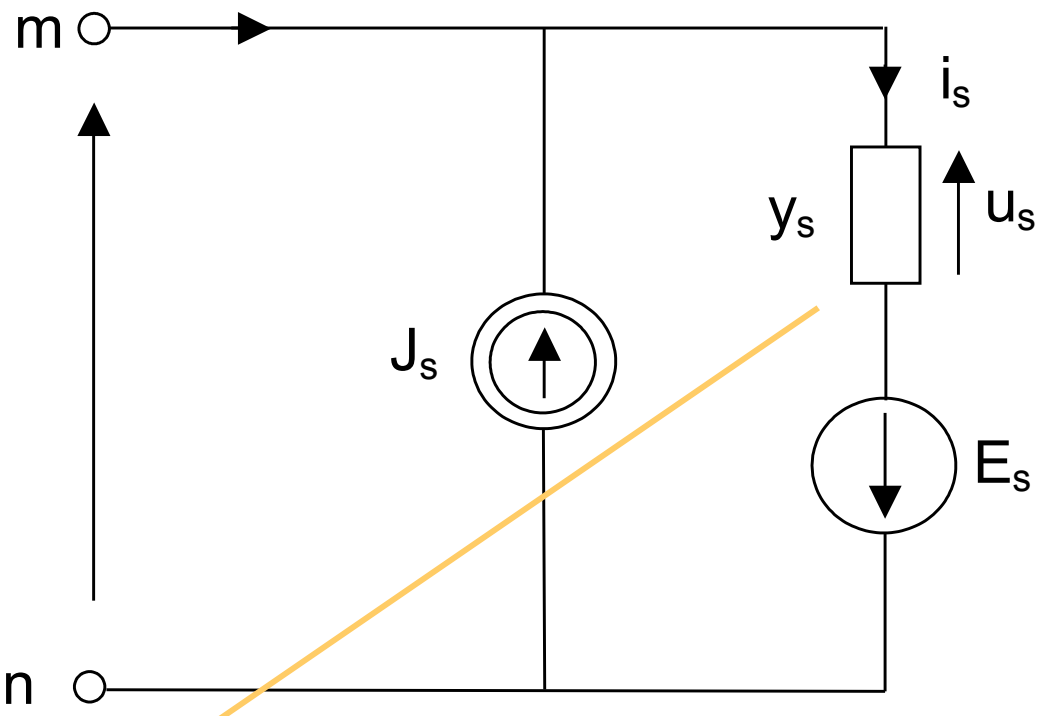


$$y_s \neq \infty, \qquad i_s = y_s u_s, \qquad K i_s = K y_s u_s = K' u_s$$

gdzie: $K' = K y_s$

Źródło prądowe sterowane prądowo

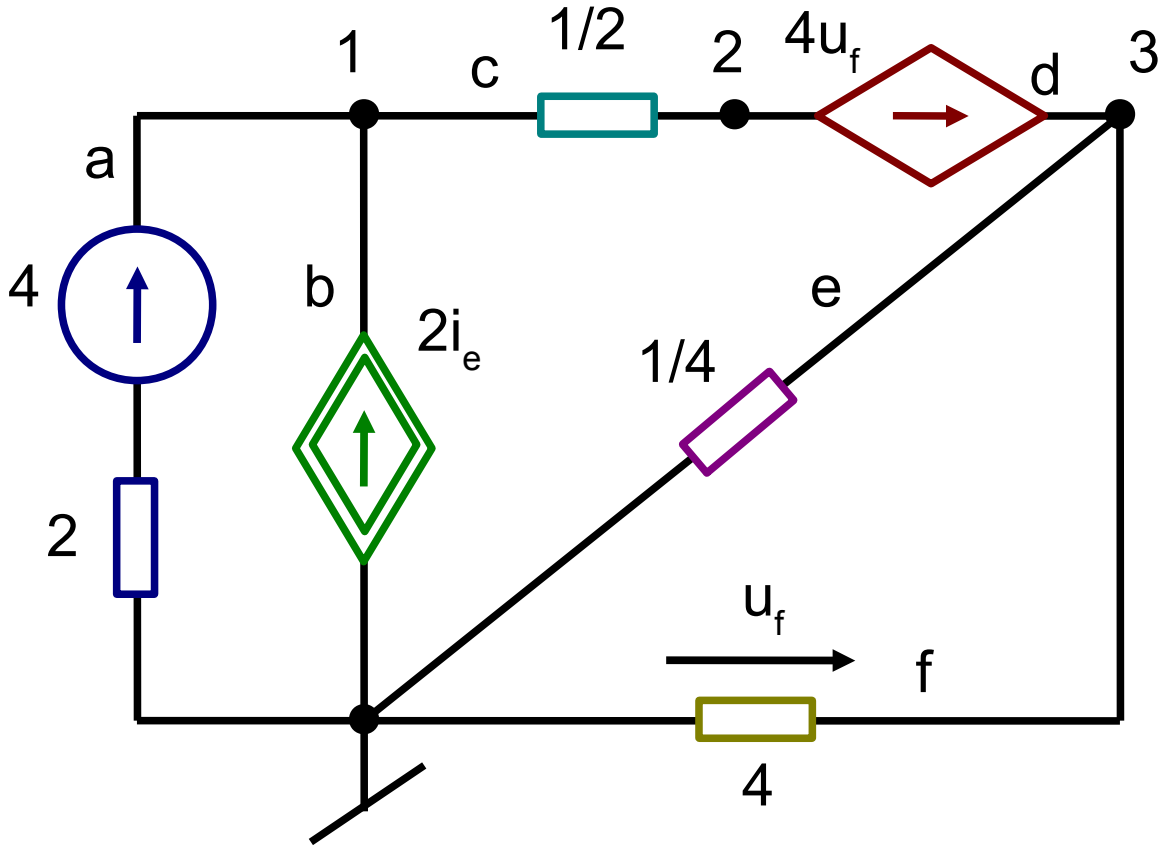


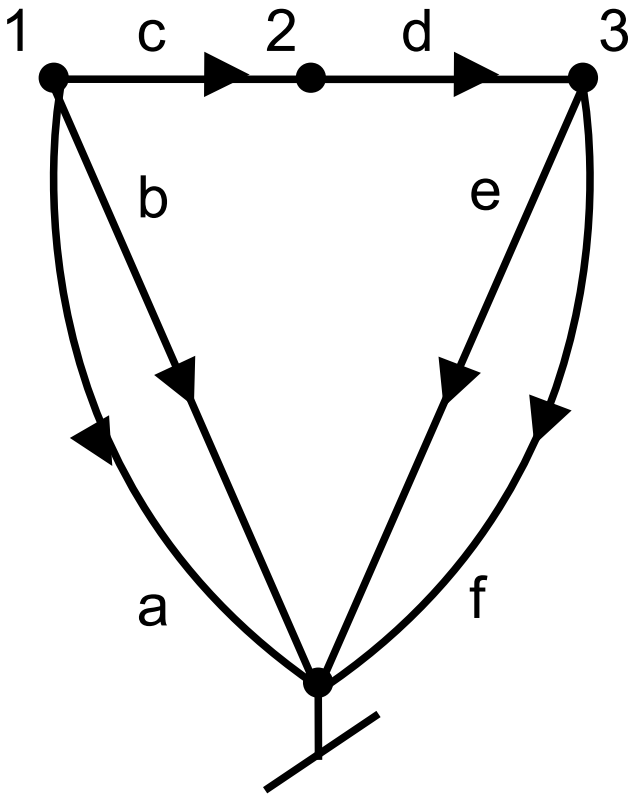


$$y_s \neq \infty, \qquad i_s = y_s u_s, \qquad A i_s = A y_s u_s = G' u_s$$

gdzie: $G' = A y_s$

Zmodyfikowana metoda potencjałów węzłowych - przykład





Zmodyfikowana metoda potencjałów węzłowych - przykład

$$Y_N = \begin{array}{c|ccc} & 1 & 2 & 3 \\ 1 & 2 & 2 & \\ 2 & -2 & 2 & \\ 3 & & & 4+1/4 \end{array} = \begin{array}{ccc} 2 & 2 & 0 \\ -2 & 2 & 0 \\ 0 & 0 & 4.25 \end{array}$$

$$J_N = \begin{array}{c|c} 1 & 0 \\ 2 & 0 \\ 3 & 0 \end{array}$$

Zmodyfikowana metoda potencjałów węzłowych - przykład

a: $2i_a+4=u_1 \rightarrow u_1-2i_a=4$

b: $i_b=-2i_e \quad i_e=4u_3 \rightarrow i_b=-8u_3 \rightarrow i_b+8u_3=0$

d: $u_3-u_2=4u_f \quad u_f=u_3 \rightarrow u_3-u_2=4u_3 \rightarrow u_3-4u_3-u_2=0$

Zmodyfikowana metoda potencjałów węzłowych - przykład

$$
\begin{array}{c}
 \\
1 \\
2 \\
3 \\
i_a \\
i_b \\
i_d
\end{array}
\begin{array}{c}
\begin{array}{cccccc} 1 & 2 & 3 & i_a & i_b & i_d \end{array} \\
\begin{bmatrix}
2 & 2 & 0 & 1 & 1 & \\
-2 & 2 & 0 & & & 1 \\
0 & 0 & 4.25 & & & -1 \\
1 & & & -2 & & \\
 & & 8 & & 1 & \\
 & -1 & 1\text{-}4 & & &
\end{bmatrix}
\end{array}
\begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ i_a \\ i_b \\ i_d \end{bmatrix}
=
\begin{bmatrix} 0 \\ 0 \\ 0 \\ 4 \\ 0 \\ 0 \end{bmatrix}
$$

Zmodyfikowana metoda potencjałów węzłowych - przykład

$$
\begin{array}{c}
 \\ 1 \\ 2 \\ 3 \\ i_a \\ i_b \\ i_d
\end{array}
\begin{array}{c}
\begin{array}{cccccc} 1 & 2 & 3 & i_a & i_b & i_d \end{array} \\
\begin{bmatrix}
\mathbf{2} & \mathbf{2} & \mathbf{0} & \mathbf{1} & \mathbf{1} & \mathbf{0} \\
\mathbf{-2} & \mathbf{2} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} \\
\mathbf{0} & \mathbf{0} & \mathbf{4.25} & \mathbf{0} & \mathbf{0} & \mathbf{-1} \\
\mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{-2} & \mathbf{0} & \mathbf{0} \\
\mathbf{0} & \mathbf{0} & \mathbf{8} & \mathbf{0} & \mathbf{1} & \mathbf{0} \\
\mathbf{0} & \mathbf{-1} & \mathbf{-3} & \mathbf{0} & \mathbf{0} & \mathbf{0}
\end{bmatrix}
\end{array}
\begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ i_a \\ i_b \\ i_d \end{bmatrix}
=
\begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{4} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix}
$$

Zmodyfikowana metoda potencjałów węzłowych - przykład

$$\begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ i_a \\ i_b \\ i_d \end{bmatrix} = \begin{bmatrix} 0.4179 \\ 1.4328 \\ -0.4776 \\ -1.7910 \\ 3.8209 \\ -2.0299 \end{bmatrix}$$

Metody macierzy rzadkich

$$\mathbf{A} = \begin{bmatrix} x & x & 0 & 0 \\ x & x & x & 0 \\ 0 & x & x & x \\ 0 & 0 & x & x \end{bmatrix} \qquad \mathbf{A}^{-1} = \begin{bmatrix} x & x & x & x \\ x & x & x & x \\ x & x & x & x \\ x & x & x & x \end{bmatrix}$$

$$\mathbf{L} = \begin{bmatrix} x & 0 & 0 & 0 \\ x & x & 0 & 0 \\ 0 & x & x & 0 \\ 0 & 0 & x & x \end{bmatrix} \qquad \mathbf{U} = \begin{bmatrix} 1 & x & 0 & 0 \\ 0 & 1 & x & 0 \\ 0 & 0 & 1 & x \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Założenia odnośnie macierzy A

1. $$\mathbf{A} = \begin{bmatrix} x & 0 & x & 0 \\ 0 & x & x & 0 \\ x & x & x & x \\ 0 & 0 & x & x \end{bmatrix}$$

macierz strukturalnie symetryczna

$$a_{ij} = 0 \quad a_{ji} = 0$$

lub

$$a_{ij} \neq 0 \quad a_{ji} \neq 0$$

2. $$a_{ii} \neq 0$$

3. $$l_{ii} \neq 0$$

4. $$l_{ij} = 0 - 2 \cdot (-6) - 3 \cdot 4 = 0$$ - mimo to przyjmujemy że powstał nowy element niezerowy - nie rozważamy konkretnych wartości

Wpływ kolejności operacji na liczbę nowych elementów niezerowych

$$\mathbf{A} = \begin{bmatrix} x & 0 & 0 & x \\ 0 & x & 0 & x \\ 0 & 0 & x & x \\ x & x & x & x \end{bmatrix} \qquad \mathbf{Q} = \begin{bmatrix} x & 0 & 0 & x \\ 0 & x & 0 & x \\ 0 & 0 & x & x \\ x & x & x & x \end{bmatrix}$$

$$\mathbf{A} = \begin{bmatrix} x & x & x & x \\ x & x & 0 & 0 \\ x & 0 & x & 0 \\ x & 0 & 0 & x \end{bmatrix} \qquad \mathbf{Q} = \begin{bmatrix} x & x & x & x \\ x & x & x & x \\ x & x & x & x \\ x & x & x & x \end{bmatrix}$$

Określanie nowych elementów niezerowych metodą grafu

Występuje analogia między rozkładem LU a metodą eliminacji Gaussa. Można dyskutować kolejność eliminacji zmiennych z równań

$$\mathbf{A} \cdot \mathbf{x} = \boldsymbol{\mu}$$

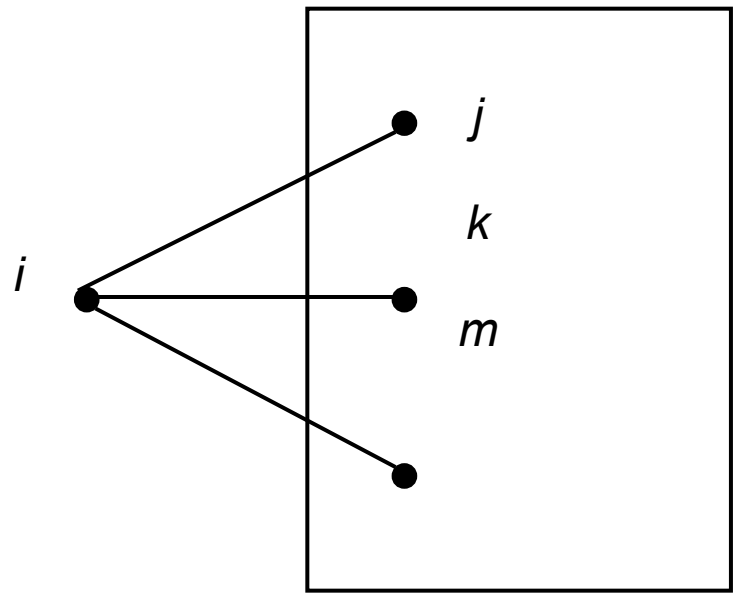
$$x_i = -\frac{1}{a_{ii}} \left(\sum_{\substack{k=1 \\ k \neq i}}^{n} a_{ik} \cdot x_k - \mu_i \right)$$

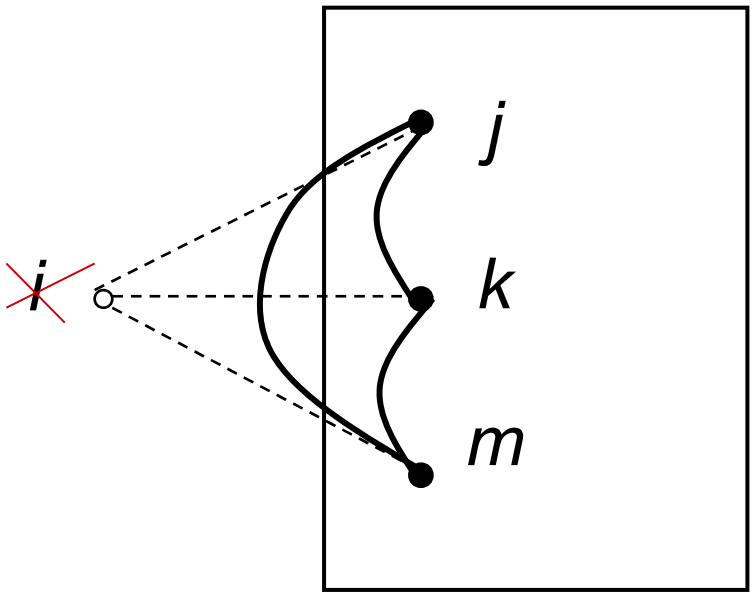
Eliminację i jej wpływ na powstawanie nowych elementów niezerowych można przedstawić poprzez odpowiedni graf odpowiadający macierzy A

Określanie liczby nowych elementów z użyciem grafu

Budujemy graf w ten sposób, że każdej parze niezerowych niediagonalnych elementów (a_{ij}, a_{ji}) odpowiada krawędź łącząca wierzchołki i oraz j.

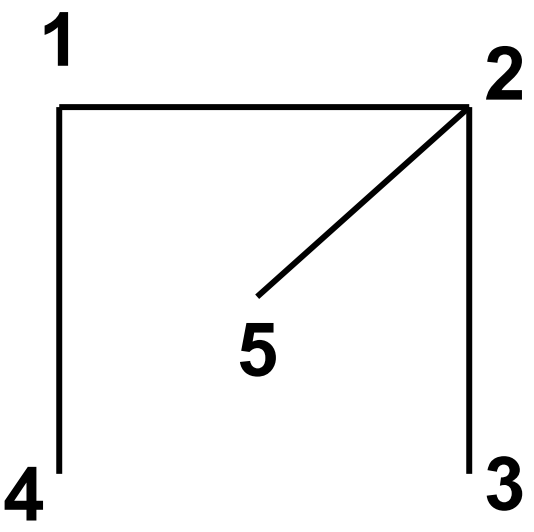
1. Wybieramy wierzchołek i. Dla każdej pary krawędzi (i, j) i (i, k) incydentnych z i, jeśli j i k nie są połączone krawędzią, tworzymy krawędź je łączącą. Utworzenie krawędzi oznacza powstanie nowego elementu niezerowego.
2. Usuwamy i oraz krawędzie do niego dochodzące. Jeśli graf jeszcze istnieje, wracamy do punktu 1.



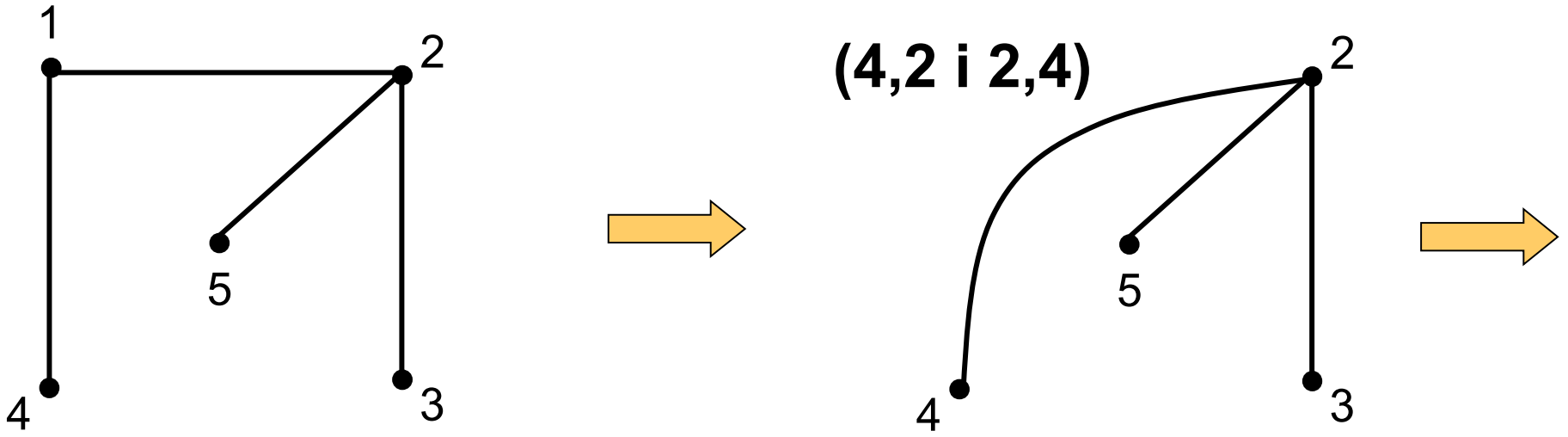


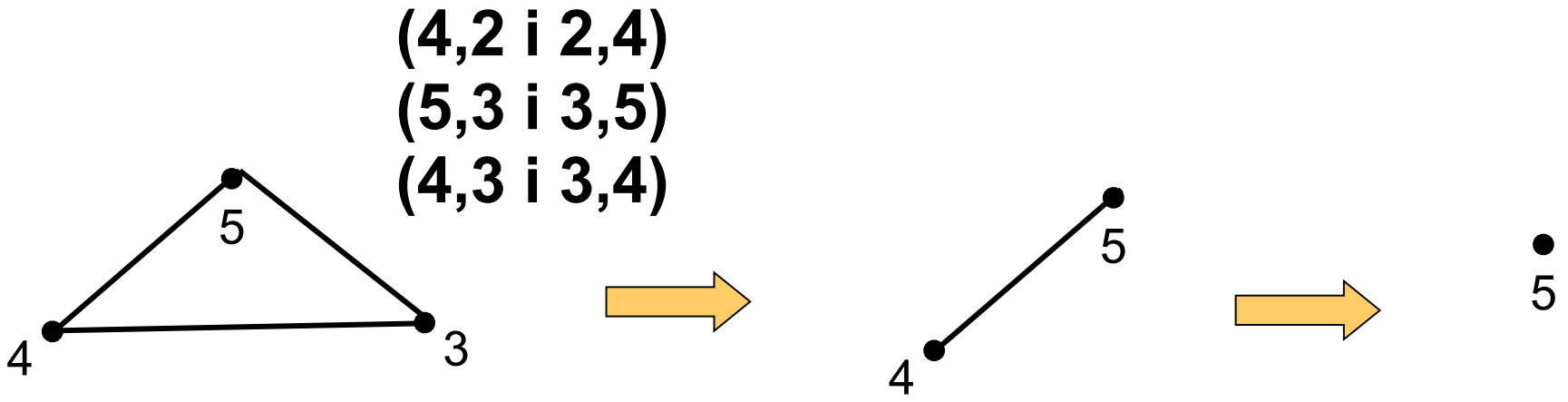
Określanie liczby nowych elementów - przykład

$$\begin{bmatrix} x & x & 0 & x & 0 \\ x & x & x & 0 & x \\ 0 & x & x & 0 & 0 \\ x & 0 & 0 & x & 0 \\ 0 & x & 0 & 0 & x \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \end{bmatrix}$$

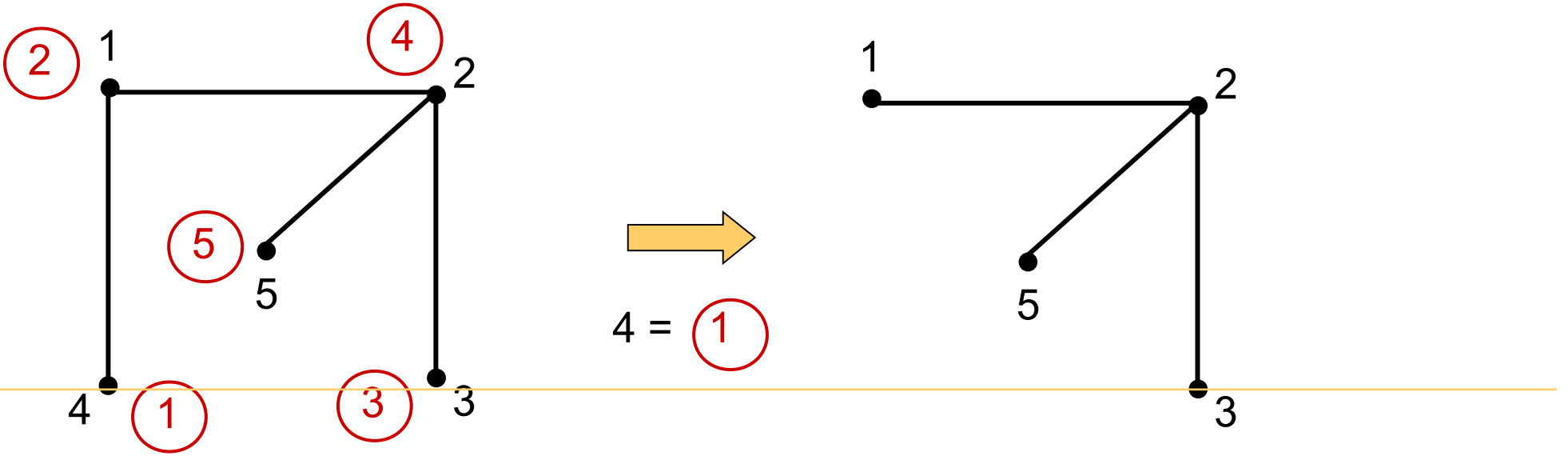


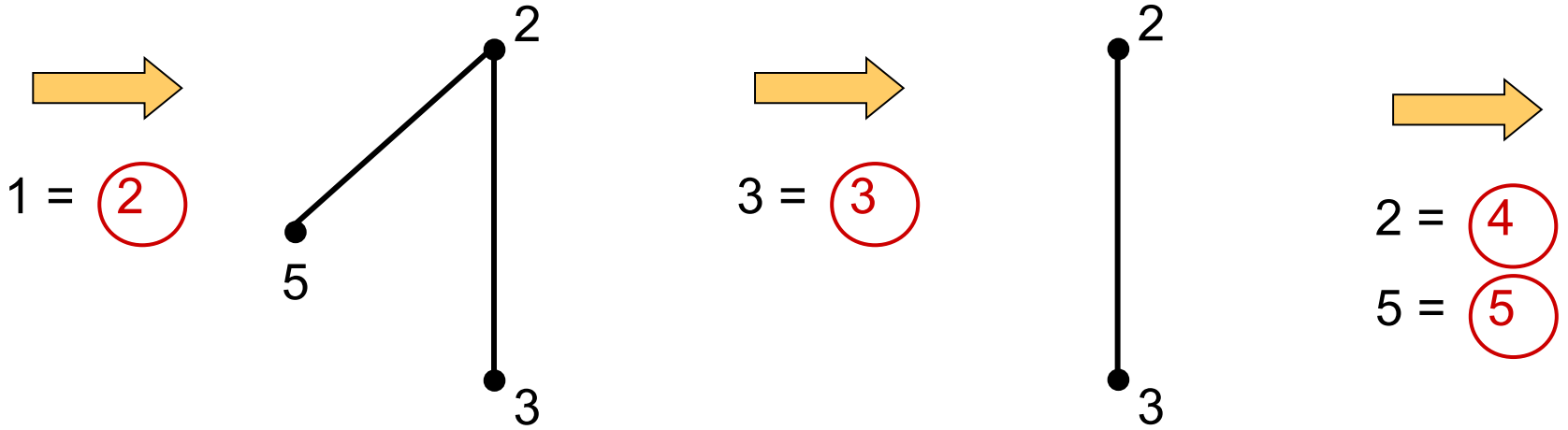
Określanie liczby nowych elementów - przykład





Określanie liczby nowych elementów - przykład





Algorytm porządkowania prawie optymalnego

- Brak jest algorytmów porządkowania optymalnego
- Przedstawiony algorytm w większości przypadków daje optymalny porządek
- Przyjmujemy strukturę macierzy A (współczynników równania) taką jak poprzednio
- Tworzymy na podstawie macierzy A graf nieskierowany tak jak poprzednio

Algorytm porządkowania prawie optymalnego

- Początkowo G jest grafem oryginalnym oraz J=1

1. W G znajdź wierzchołek incydentny z tylko jedną krawędzią. Jeśli jest, nadaj mu numer J, wykonaj 4 i wróć do 1. Jeśli nie ma, idź do 2.
2. W G znajdź wierzchołek którego eliminacja nie wprowadzi nowej krawędzi. Jeśli jest, nadaj mu numer J, wykonaj 4 i wróć do 1. Jeśli nie ma, idź do 3.

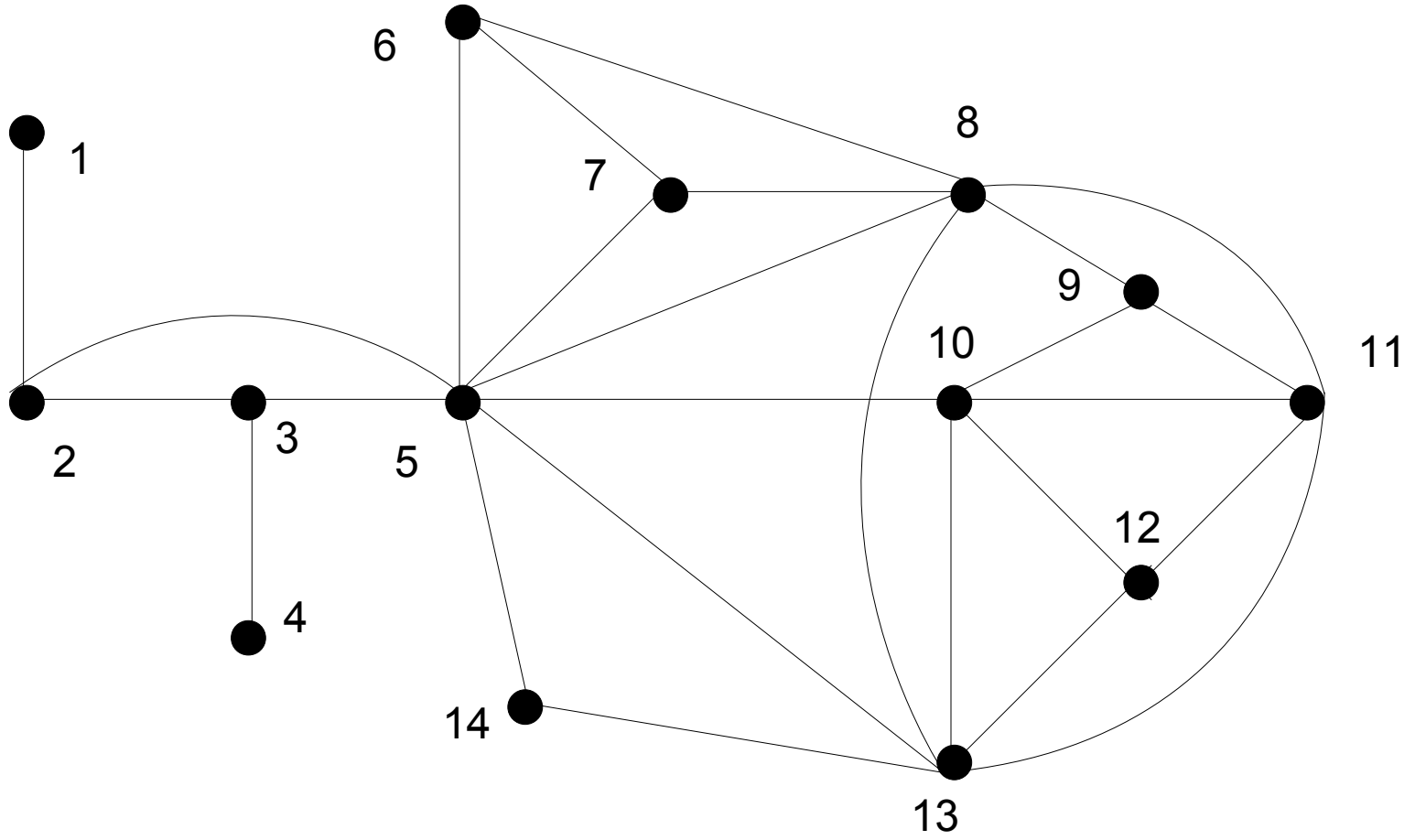
Algorytm porządkowania prawie optymalnego

3. Dla każdego wierzchołka w G określ liczbę n nowych elementów niezerowych przy założeniu, że tylko on będzie wyeliminowany. Wybierz ten, który ma najmniejsze n (jeśli dla kilku n jest takie samo, wybierz ten który jest incydentny z największą liczbą krawędzi). Nadaj mu numer J, wykonaj 4 i wróć do 1.

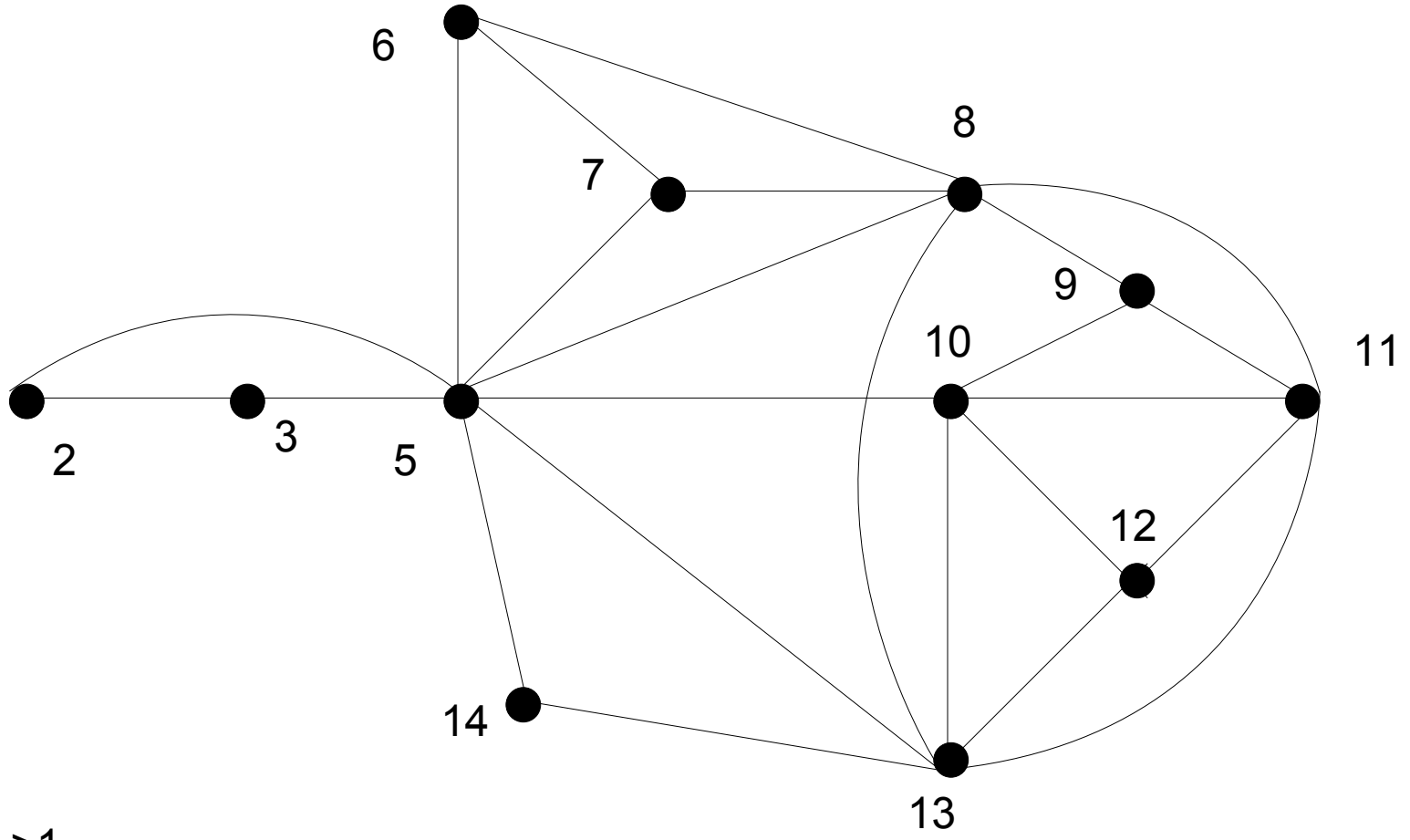
Algorytm porządkowania prawie optymalnego

4. Jeśli został tylko jeden nieprzenumerowany wierzchołek, nadaj mu numer J i koniec algorytmu. W przeciwnym przypadku usuń nowo przenumerowany wierzchołek i krawędzie z nim incydentne, dodaj krawędzie odpowiadające nowym elementom niezerowym i zwiększ J o 1.

Algorytm porządkowania prawie optymalnego - przykład

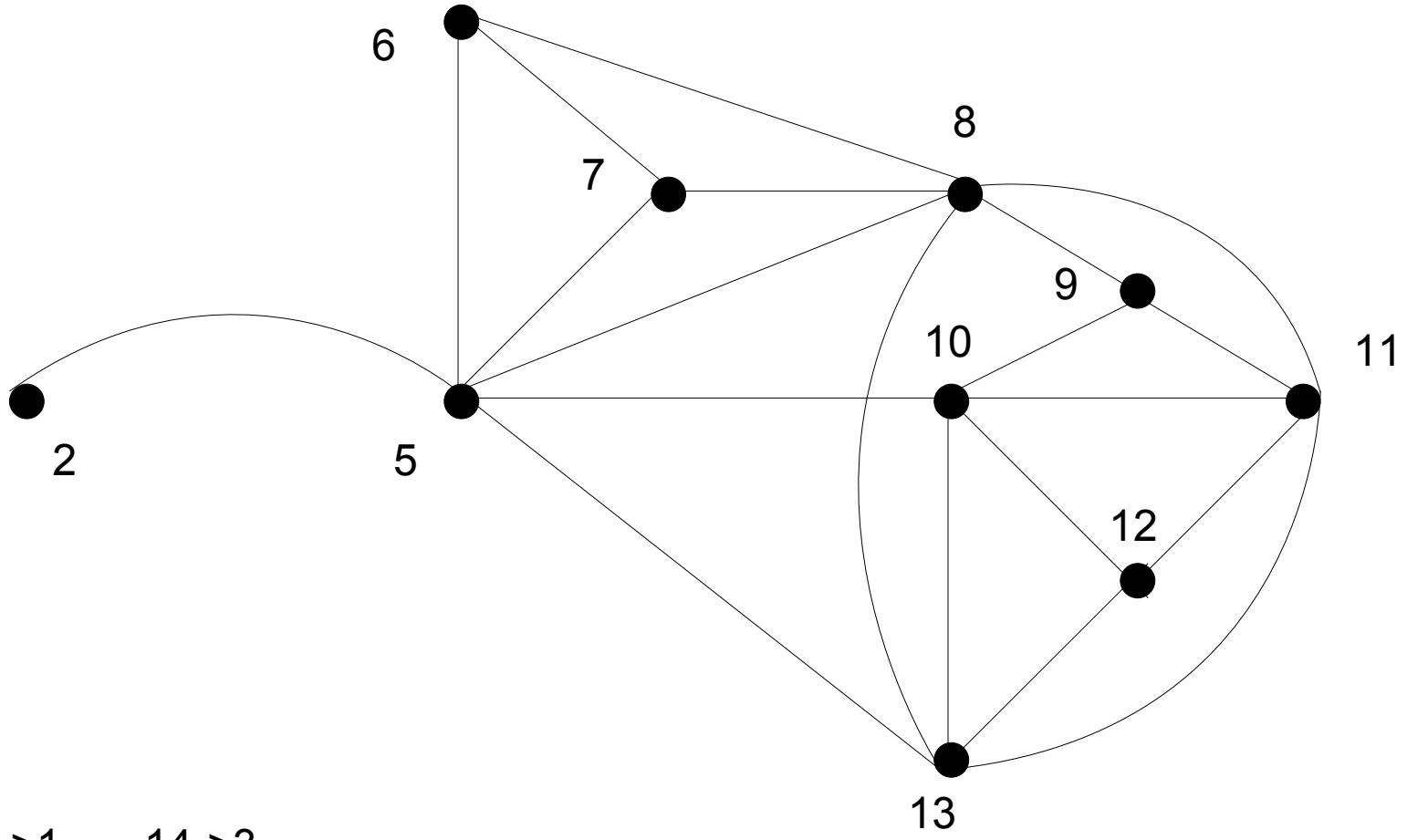


Algorytm porządkowania prawie optymalnego - przykład



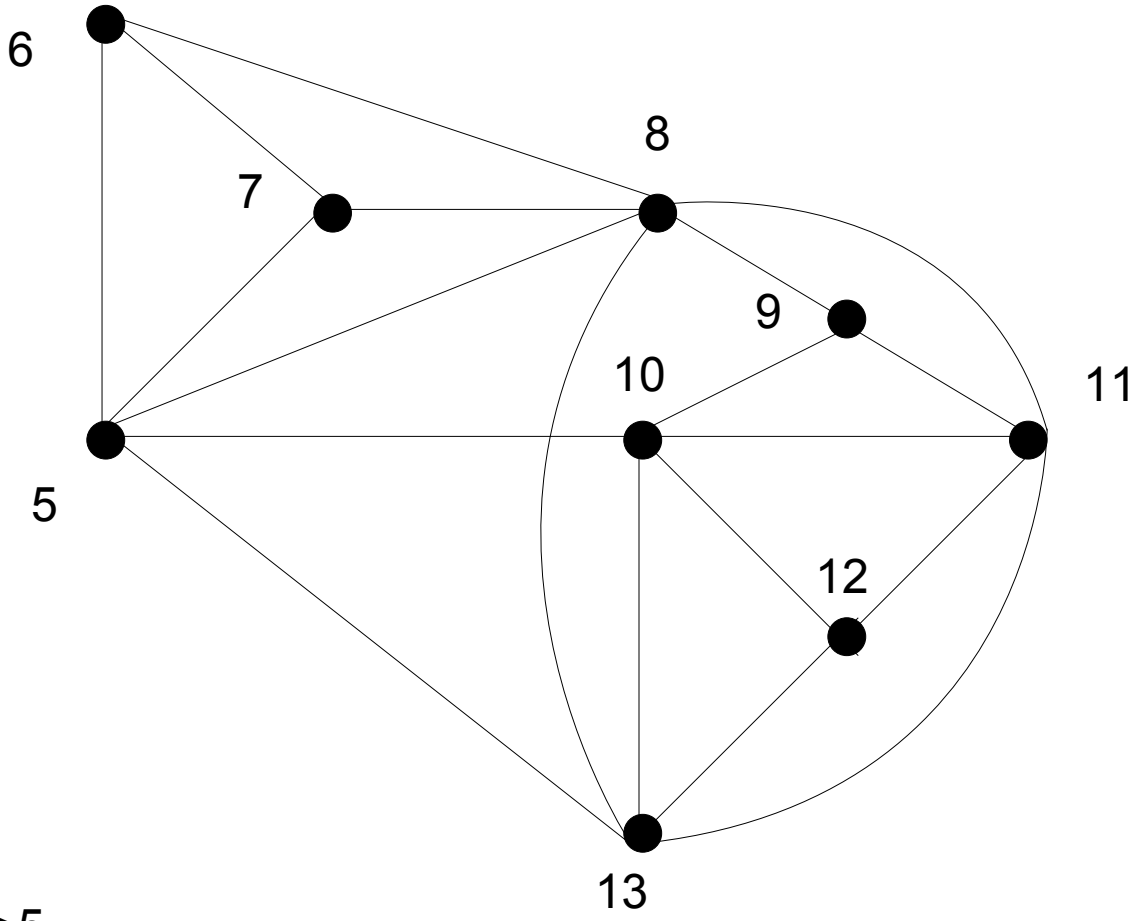
1->1
4->2

Algorytm porządkowania prawie optymalnego - przykład



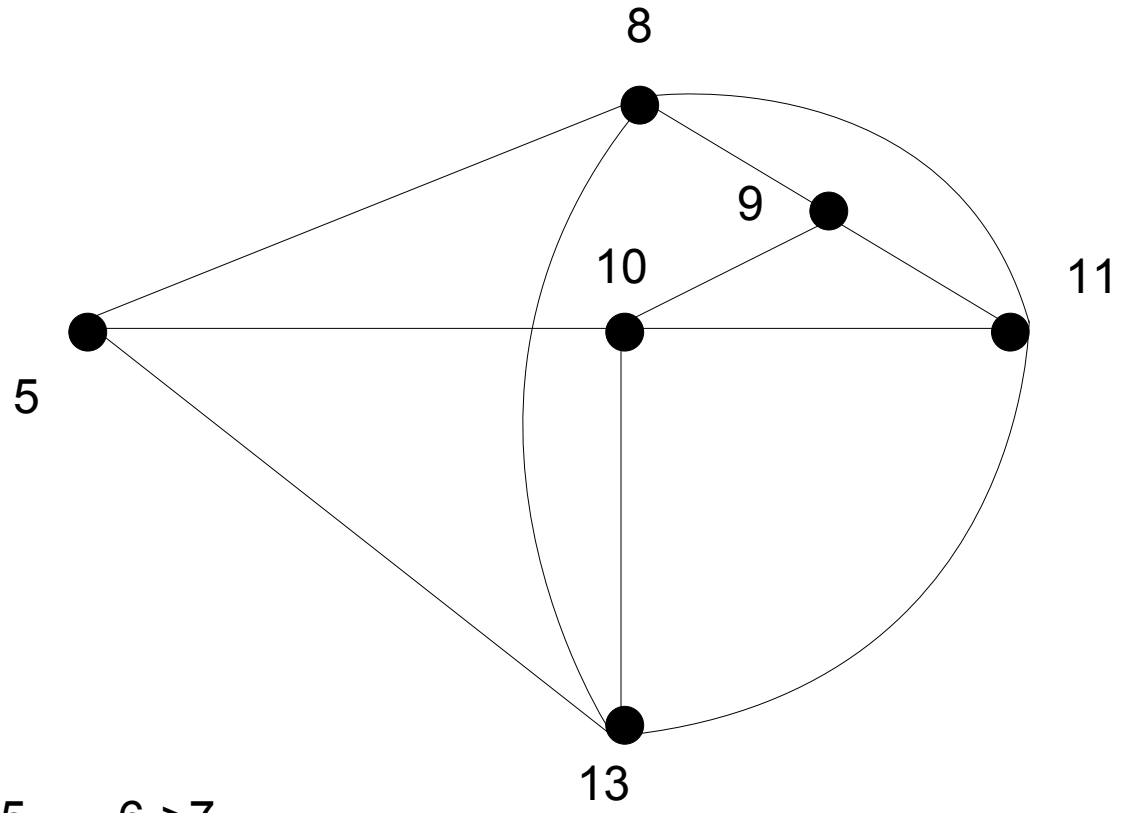
1->1 14->3
4->2 3->4

Algorytm porządkowania prawie optymalnego - przykład



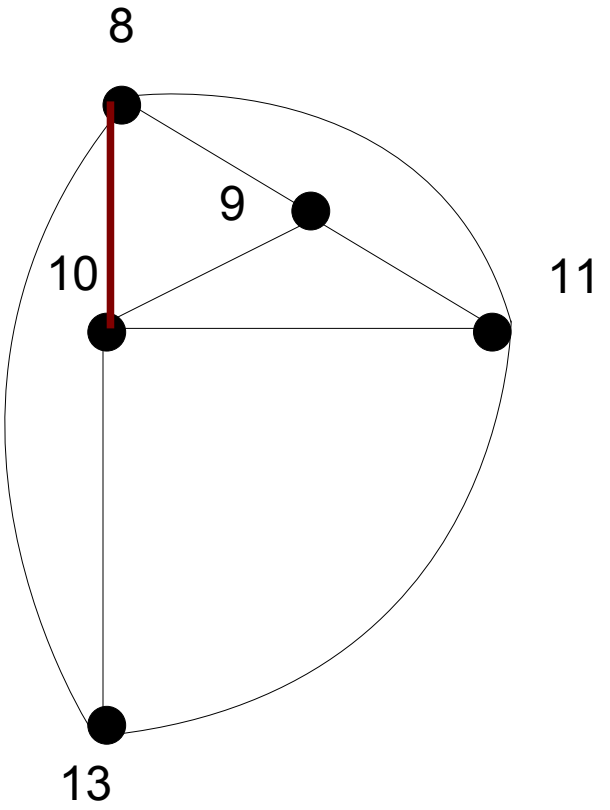
1->1 14->3 2->5
4->2 3->4

Algorytm porządkowania prawie optymalnego - przykład



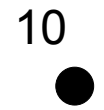
1->1	14->3	2->5	6->7
4->2	3->4	7->6	12->8

Algorytm porządkowania prawie optymalnego - przykład



1->1 14->3 2->5 6->7 5->9
4->2 3->4 7->6 12->8

Algorytm porządkowania prawie optymalnego - przykład



1->1	14->3	2->5	6->7	5->9	9->11	11->13
4->2	3->4	7->6	12->8	13->10	8->12	

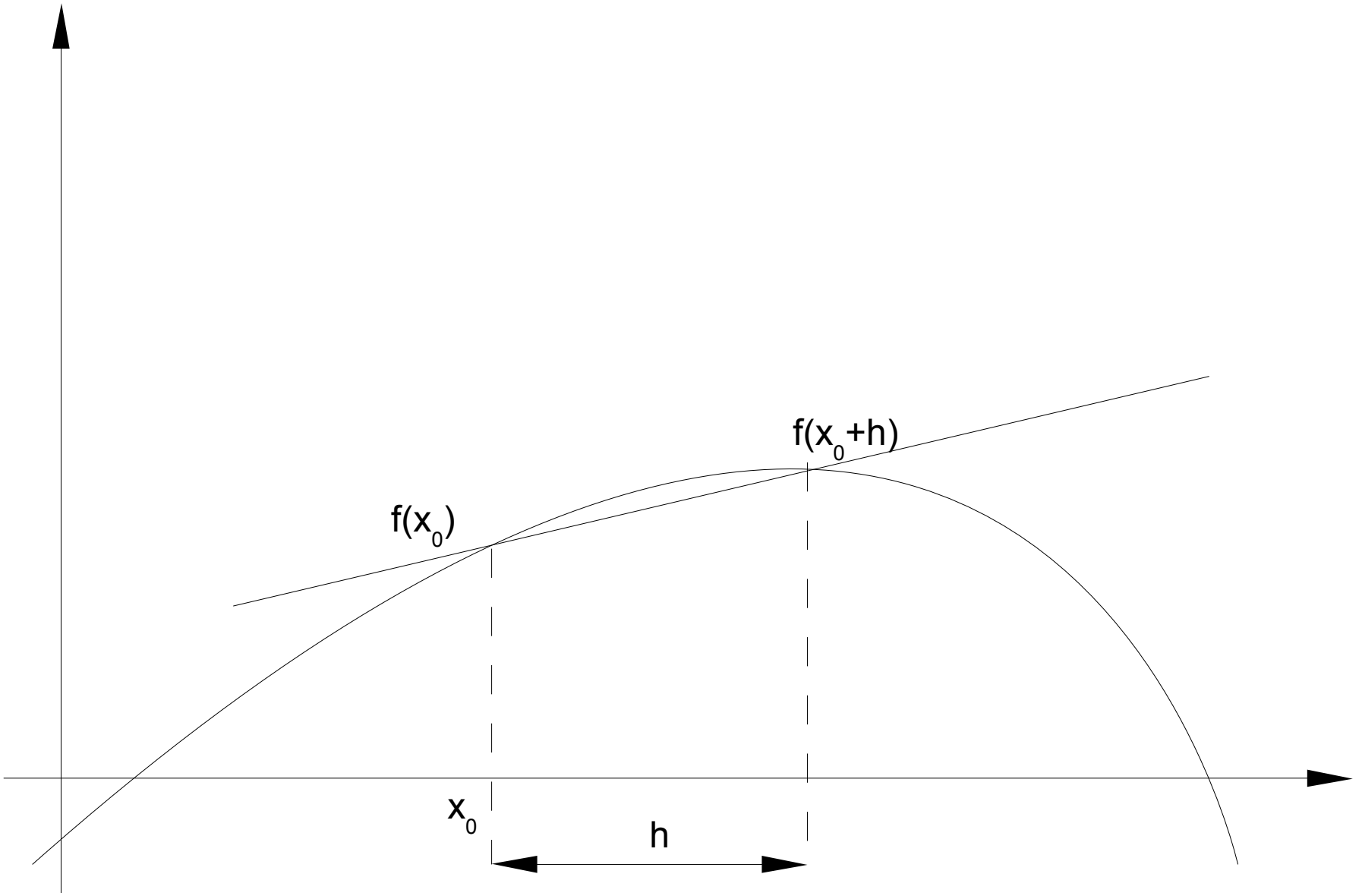
Algorytm porządkowania prawie optymalnego - przykład

1->1	14->3	2->5	6->7	5->9	9->11	11->13
4->2	3->4	7->6	12->8	13->10	8->12	10->14

Różniczkowanie numeryczne

- W wielu przypadkach konieczna jest znajomość wartości pochodnej funkcji – np. metoda Newtona-Raphsona
- Nie zawsze dostępna jest postać analityczna
- Wartość pochodnej w punkcje może być w przybliżeniu wyznaczona metodami numerycznymi

Przybliżenie sieczną



Przybliżenie sieczną

$$f'(x_0) = \frac{f(x_0 + h) - f(x_0)}{h}$$

- Wartość ta różni się od prawidłowej wartości pochodnej
- Różnica ta będzie tym mniejsza, im mniejsze będzie *h*
- Prawidłową wartość otrzymamy dla $h \to 0$

Przybliżenie sieczną

- W implementacji komputerowej nie ma pojęcia "$h \to 0$"
- Bezpośrednie podstawienie wartości 0 da wyrażenie 0/0
- Wybór prawidłowej wartości *h* jest uzależniony od dostępnej precyzji obliczeń zmiennoprzecinkowych
  - Zbyt mała wartość spowoduje błędy obliczeń
  - Zbyt duża wartość będzie złym przybliżeniem pochodnej

Przybliżenie sieczną

- W przypadku małej wartości *h* mogą wystąpić następujące problemy:
  - wartość będzie potraktowana jako 0
  - silny wpływ błędów zaokrągleń przy dzieleniu
  - wartości $f(x_0)$ i $f(x_0+h)$ będą sobie równe, bądź ich różnica będzie obarczona dużym błędem
- Często przyjmuje się h=sqrt(eps)*x, gdzie eps to maksymalny błąd zaokrągleń dla danej platformy sprzętowej

Przybliżenie sieczną

- Alternatywnie można dokonać przybliżenia sieczną przechodzącą przez punkty „przed” i „za” punktem liczenia pochodnej:

$$f'(x_0) = \frac{f(x_0 + h_1) - f(x_0 - h_2)}{h_1 + h_2}$$

- Jeśli odległości *h* są sobie równe daje to:

$$f'(x_0) = \frac{f(x_0 + h) - f(x_0 - h)}{2 \cdot h}$$

Całkowanie numeryczne

- Rodzina algorytmów pozwalających na obliczenie przybliżonej wartości całki oznaczonej
- Alternatywna nazwa – kwadratury, czasem w stosunku do całek wyższych wymiarów używa się nazwy: kubatury

Całkowanie numeryczne

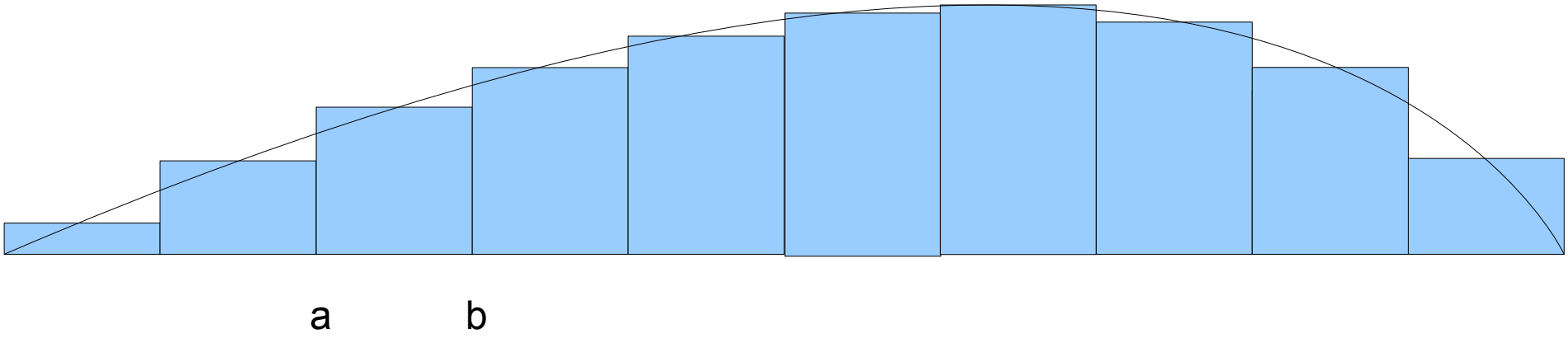
- Konieczność całkowania numerycznego wynika z:
  - braku znajomości analitycznej postaci funkcji całkowanej (np. funkcja jest dana jako zbiór punktów)
  - trudność ze znalezieniem całki analitycznie (nie istnieje zapis w postaci funkcji elementarnej, wymagana funkcja specjalna niezaimplementowana w systemie itp.)

Metoda prostokątów

- Jest to najprostsza metoda należąca do grupy opartej o funkcje interpolacyjne
- Interpolacja przeprowadzana wielomianem stopnia 0 (funkcją stałą)
- Jako wartość do obliczeń brana wartość funkcji w środku przedziału interpolowanego (stąd alternatywna nazwa: metoda punku środkowego

Metoda prostokątów

$$\int_a^b f(x)dx \approx (b-a) f\left(\frac{a+b}{2}\right)$$



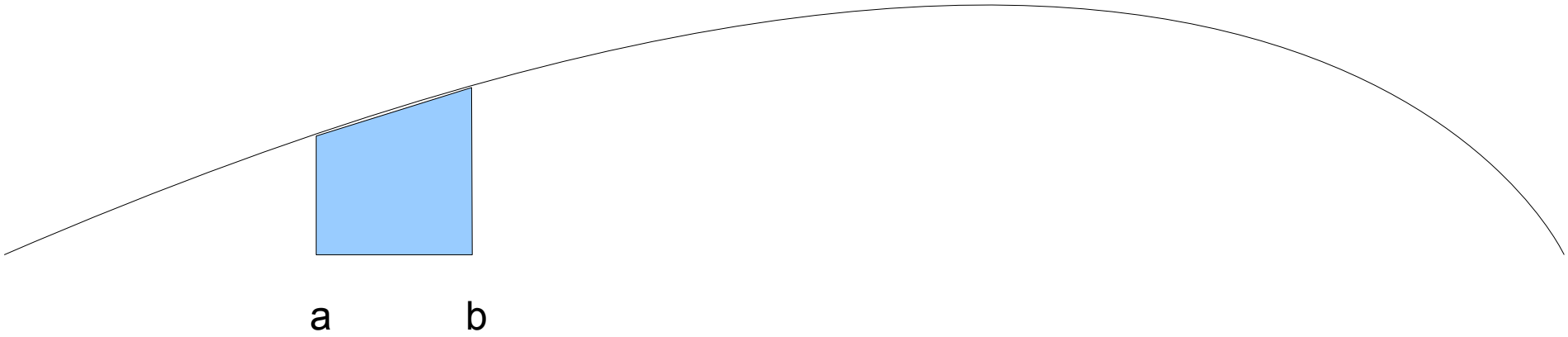
Metoda prostokątów

- Wzór traktuje przedział <a; b> jako jedną całość
- W praktyce rozbija się przedział <a; b> na n podprzedziałów
- Metoda prostokątów przyjmuje jednakową szerokość podprzedziałów
- Wraz ze wzrostem n rośnie dokładność przybliżenia

Metoda trapezów

- Stosując interpolację wielomianem I stopnia (funkcja liniowa) otrzymujemy metodę trapezów

$$\int_a^b f(x)dx \approx (b-a)\frac{f(a)+f(b)}{2}$$



Metoda Simpsona

- Stosując interpolację funkcją kwadratową otrzymujemy metodę Simpsona (parabol)
- Jest ona też równoważna ważonej średniej z metod prostokątów i trapezów.

$$\int_a^b f(x)dx \approx \frac{b-a}{6}\left[f(a)+4\cdot f\left(\frac{a+b}{2}\right)+f(b)\right]$$

Metody Newtona-Cotesa

- Przedstawione metody należą do rodziny metod Newtona-Cotesa
- Przybliżają one całkę poprzez średnią ważoną wartości funkcji całkowanej w równomiernie rozłożonych punktach należących do przedziału całkowania

$$\int_a^b f(x)dx \approx \sum_{i=0}^{n} w_i \cdot f(x_0 + h \cdot i)$$

przy czym $h = (b - a)/n$ jest nazywane krokiem

Metody Newtona-Cotesa

- Wagi wyznacza się całkując bazowe wielomiany Lagrange'a
- Można w ten sposób skonstruować metodę dowolnego rzędu, należy jednak pamiętać, że dla wyższych rzędów mogą pojawić się niestabilności (efekt Rungego)

Oprogramowanie do obliczeń i symulacji inżynierskich

- Matlab
- Octave
- SciLab

Matlab - podstawowe informacje

- Skrót od **Mat**rix **Lab**oratory
- Opracowany pod koniec lat 70 XXw. na uniwersytecie New Mexico
- W 1984 powstaje firma MathWorks; komercjalizacja programu
- W 2004 używany przez ponad 1 000 000 użytkowników; de facto standard dla numerycznych obliczeń inżynierskich
- http://www.mathworks.com/products/matlab/

Matlab - podstawowe informacje

- Dokonuje jedynie obliczeń numerycznych, choć można dołączyć interfejs dla obliczeń algebraicznych
- Niemalże wszystko w Matlabie jest macierzą
- Dostępnych jest bardzo wiele tzw. toolboxów, rozszerzających funkcjonalność podstawowego programu o np. optymalizację, obliczenia statystyczne, obliczenia AI, przetwarzanie sygnałów
- Istotnym dodatkiem jest Simulink, służący do wielodomenowego modelowania systemów

Matlab - podstawy pracy

- Można wyróżnić dwa podstawowe tryby pracy:
  - interaktywny
  - skryptowy
- Oba tryby są interpretowane
- W obu trybach można używać tych samych poleceń
- Skrypty Matlaba mają rozszerzenie `.m`
- Jest możliwość skompilowania kodu Matlaba tak, aby uzyskać osobną aplikację

Matlab - zmienne

- Zmienne nie muszą być deklarowane przed przypisaniem wartości (ale muszą mieć przypisaną wartość przed użyciem w wyrażeniu)
- Wartości przypisuje się operatorem `=`
- Jeśli nie został przypisany do konkretnej zmiennej, ostatni wynik obliczeń jest zapisywany w zmiennej `ans`
- Aby odczytać wartość zmiennej, należy podać jej nazwę
- Aby wynik działania nie był wyprowadzany na ekran, polecenie należy zakończyć średnikiem `;`

```
octave-3.0.0.exe:21> myval=7
myval =  7
octave-3.0.0.exe:22> anotherval=8
anotherval =  8
octave-3.0.0.exe:23> myval=anotherval
myval =  8
octave-3.0.0.exe:24> myval=thirdval
error: `thirdval' undefined near line 24 column 7
error: evaluating assignment expression near line 24, column 6
octave-3.0.0.exe:24> myval
myval =  8
octave-3.0.0.exe:25> myval=9;
octave-3.0.0.exe:26> myval
myval =  9
```

Matlab - proste obliczenia

- Można używać typowych operatorów arytmetycznych, nawiasów itd.
- Kolejność obliczeń jest zgodna z kolejnością operatorów arytmetycznych
- można używać operatorów porównań: `==, <, >, <=, >=, !=`. Wynikiem porównań jest 1 (prawda) lub 0 (fałsz). Matlab traktuje zawsze wartości nie-zerowe jako prawdę
- można używać operatorów logicznych: `&, I, ~`

```
octave-3.0.0.exe:40> anotherval=2+myval*8
anotherval =  74
octave-3.0.0.exe:41> 1>2
ans = 0
octave-3.0.0.exe:42> 5!=6
ans =  1
octave-3.0.0.exe:43> a=4
a =  4
octave-3.0.0.exe:44> b=3
b =  3
octave-3.0.0.exe:45> c=4*(a>b)
c =  4
```

Matlab - wywołanie funkcji

- Funkcje w Matlabie mogą by:
  - wbudowane
  - zawarte w m-plikach
- Funkcję wywołuje się podając jej nazwę i argumenty w nawiasach okrągłych

```
octave-3.0.0.exe:48> x=sin(3.14)
x =  0.0015927
octave-3.0.0.exe:49> y=cos(1)
y =  0.54030
octave-3.0.0.exe:50> abs(-9)
ans =  9
```

Macierze i wektory

- Nawet pojedyncza wartość w Matlabie może być traktowana jako macierz/wektor
- Matlab zawiera szereg operatorów umożlwiających budowanie macierzy oraz odwoływanie się do poszczególnych ich fragmentów
- Najprostszą metodą utworzenia macierzy jest podanie jej wartości w nawiasach kwadratowych; wartości w rzędzie rozdziela się spacją lub przecinkiem, w kolumnie - średnikiem

```
octave-3.0.0.exe:53> a = [1 2 3; 4 5 6]
a =

   1   2   3
   4   5   6

octave-3.0.0.exe:54> b = [1 2; 3 4; 5 6]
b =

   1   2
   3   4
   5   6
```

Macierze i wektory

- Dostęp do elementu/elementów macierzy umożliwia operator （）(nawiasy okrągłe)
- W nawiasach podaje się wymagane indeksy, bądź zakres indeksów - używając operatora : (dwukropek)
- Jeśli odwołanie ma dotyczyć wszystkich indeksów w danym wymiarze, można użyć samego dwukropka
- Najpierw podaje się indeks wiersza, potem kolumny

```
octave-3.0.0.exe:57> a(1, 1)
ans =  1
octave-3.0.0.exe:58> a(2, 1)
ans =  4
octave-3.0.0.exe:59> a(1, 2:3)
ans =

   2   3

octave-3.0.0.exe:60> a(1,:)
ans =

   1   2   3

octave-3.0.0.exe:61> a(:,:)
ans =

   1   2   3
   4   5   6
```

Odwołanie do elementu a wywołanie funkcji

- Matlab stosuje ten sam operator do odwołania do fragmentu macierzy i do wywołania funkcji
- Powoduje to problemy jeśli nazwa zmiennej jest identyczna z nazwą funkcji

```
octave-3.0.0.exe:75> sin(1)
ans =  0.84147
octave-3.0.0.exe:76> sin = 7
sin =  7
octave-3.0.0.exe:77> sin(1)
ans =  7
```

Konstruowanie macierzy

- Macierze można konstruować nie tylko ze skalarów, ale też z innych macierzy
- Stosuje się te same operatory co poprzednio, ale zamiast skalarów wstawia macierze

```
octave-3.0.0.exe:80> d = [1 2; 3 4]
d =
   1   2
   3   4
octave-3.0.0.exe:81> e = [5 6; 7 8]
e =
   5   6
   7   8
octave-3.0.0.exe:82> f = [d e]
f =
   1   2   5   6
   3   4   7   8
octave-3.0.0.exe:83> g = [d; e]
g =
   1   2
   3   4
   5   6
   7   8
octave-3.0.0.exe:84> h= [d; -1 -2]
h =
   1   2
   3   4
  -1  -2
```

Operacje na macierzach

- W przypadku macierzy dostępnych jest więcej operacji niż w przypadku skalarów
- Ten sam operator może mieć nieco inne działanie w przypadku kiedy działa na skalarach, macierzach, bądź skalarach i macierzach jednocześnie
- Różne może być też działanie funkcji w zależności czy argumentem jest skalar, wektor (macierz jednowymiarowa) czy macierz (wielowymiarowa)

Operacje na macierzach

- * mnożenie macierzy
- / mnożenie przez odwrotność macierzy z prawej strony
- \ mnożenie odwrotności macierzy z lewej strony
- .* ./ .\ mnożenie i dzielenie tablicowe
- * / mnożenie i dzielenie przez skalar
- + - dodawanie i odejmowanie macierzy
- + - dodawanie i odejmowanie macierzy i skalara
- ' transpozycja

```
octave-3.0.0.exe:26> a = [1 2; 3 4]
a =

   1   2
   3   4

octave-3.0.0.exe:27> b = [5 6; 7 8];
octave-3.0.0.exe:28> c = [1 2 3; 4 5 6];
octave-3.0.0.exe:29> a*b
ans =

   19   22
   43   50

octave-3.0.0.exe:30> a.*b
ans =

    5   12
   21   32

octave-3.0.0.exe:31> a*c
ans =

    9   12   15
   19   26   33
```

```
octave-3.0.0.exe:32> a.*c
error: product: nonconformant arguments (op1 is 2x2, op2 is 2x3)
error: evaluating binary operator `.*' near line 32, column 2
octave-3.0.0.exe:32> c*a
error: operator *: nonconformant arguments (op1 is 2x3, op2 is 2x2)
error: evaluating binary operator `*' near line 32, column 2
```

```
octave-3.0.0.exe:34> c'*a
ans =

   13   18
   17   24
   21   30

octave-3.0.0.exe:35> a+b
ans =

    6    8
   10   12

octave-3.0.0.exe:36> a+5
ans =

   6   7
   8   9

octave-3.0.0.exe:37> a*5
ans =

    5   10
   15   20
```

Funkcje macierzy

- Funkcje normalnie działające na skalarach po podaniu macierzy obliczą swoją wartość dla każdego elementu macierzy (np. sin)
- Funkcje z zasady działające na macierzach/wektorach mogą się zachowywać odmiennie (np. sum)

```
octave-3.0.0.exe:43> a = [0.1 0.2; 0.3 0.4];
octave-3.0.0.exe:44> sin(a)
ans =

   0.099833   0.198669
   0.295520   0.389418

octave-3.0.0.exe:45> b = [1 2 3];
octave-3.0.0.exe:46> sum(b)
ans =  6
octave-3.0.0.exe:47> sum(b')
ans =  6
octave-3.0.0.exe:48> sum(a)
ans =

   0.40000   0.60000

octave-3.0.0.exe:49> sum(a')
ans =

   0.30000   0.70000
```

Informacje o macierzach

- Funkcja size zwraca rozmiar macierzy, jako wektor n-elementowy
- size(x, 1) zwróci liczbę wierszy macierzy
- size(x, 2) zwróci liczbę kolumn macierzy

```
octave-3.0.0.exe:52> size(b)
ans =

   1   3

octave-3.0.0.exe:53> size(b, 1)
ans =  1
octave-3.0.0.exe:54> size(b, 2)
ans =  3
```

Manipulacje macierzami

- Macierz można rozszerzyć podając wartość nieistniejącego elementu

```
octave-3.0.0.exe:73> a = [1 2]
a =

   1   2

octave-3.0.0.exe:74> a(9) = 8
a =

   1   2   0   0   0   0   0   0   8
```

Manipulacje macierzami

- Macierz można zawęzić wpisując macierz pustą

```
octave-3.0.0.exe:80> a
a =

   1   2   0   0   0   0   0   0   8

octave-3.0.0.exe:81> a(3:8)=[]
a =

   1   2   8
```

Manipulacje macierzami

- Macierz można:
  - przekształcić w macierz o innej liczbie wierzy i kolumn: reshape
  - obrócić: rot90
  - odbić wzdłuż osi: fliplr, flipud
  - przekształcić w macierz o elementach przesuniętych kołowo: circshift
  - sortować: sort, sortrows

m-pliki

- m-pliki mogą zawierać
  - skrypty
  - funkcje
- Skrypty pracują na zmiennych aktualnie istniejących w środowisku
- Funkcje pracują na przekazanych argumentach; zmienne tworzone w funkcji są lokalne dla tej funkcji
- Wywołanie skryptu lub funkcji następuje po podaniu nazwy m-pliku (bez rozszerzenia) oraz argumentów (tylko dla funkcji)

Funkcje

- m-plik o nazwie nazwa_funkcji zawierający na początku deklarację:
  ```
  function [rezultat1,
  rezultat2, ...]=
  nazwa_funkcji(argument1,argument2,
  ...)
  ```
  i opcjonalnie kończący się instrukcją `return`

Funkcje - przykład

- Zawartość pliku circle.m

```
function [area, circumference] = circle(radius)
area = pi * radius^2;
circumference = 2 * pi * radius;
```

  - Wywołanie

```
octave-3.0.0.exe:5> [a, c] = circle(5)
a =  78.540
c =  31.416
octave-3.0.0.exe:6> x = circle(5);
octave-3.0.0.exe:7> x
x =  78.540
```

Funkcje

- Aby funkcja mogła zostać wywołana, musi być
  - w aktualnym katalogu
  - w katalogu na ścieżce dostępu Matlaba
- Aktualny katalog można sprawdzić komendą pwd, zmienić komendą cd
- Ścieżkę dostępu można sprawdzić komendą path, modyfikować komendami addpath, rmpath
- Aktualny katalog jest zawsze przeszukiwany pierwszy

Komentarze

- Komentarz można umieścić znakiem %
- Tekst od % do końca linii jest traktowany jako komentarz
- Komentarz zawarty na początku pliku funkcji (przed deklaracją) może być wyświetlany komendą help

```
% function calculating area and circumference of a circle
% inputs: circle radius

function [area, circumference] = circle(radius)
area = pi * radius^2;   % compute area
circumference = 2 * pi * radius; % compute circumference
```

```
octave-3.0.0.exe:14> [a, c] = circle(5)
a =  78.540
c =  31.416
octave-3.0.0.exe:15> help circle
function calculating area and circumference of a circle
inputs: circle radius
```

Instrukcje sterujące

```
if warunek1
   instrukcje1
elseif warunek2
   instrukcje2
else
   instrukcje3
end
```

```
switch zmienna
    case wartość1
        instrukcje1
    case wartość2
        instrukcje2
    ...
    otherwise
        instrukcjeN
end
```

```
function result = testif(first, second)

if first > second
   result = first;
else
   result = second;
end
```

```
function result = testswitch(condition, first, second, third)

switch condition
   case 1
      result = first;
   case 2
      result = second;
   case 3
      result = third;
   otherwise
      result = 0;
end
```

Pętle

- Matlab pozwala korzystać z pętli typu for i while

```
for zmienna = wartość_początkowa:przyrost:wartość_końcowa
   instrukcje
end
```

```
while warunek
       instrukcje
end
```

```
function result = testfor(start, stop)

result = [];
for i = start : 1 : stop
   result = [result circle(i)];
end
```

```
octave-3.0.0.exe:25> testfor(5, 10)
ans =

    78.540   113.097   153.938   201.062   254.469   314.159

octave-3.0.0.exe:26> x = testfor(5, 10);
octave-3.0.0.exe:27> x
x =

    78.540   113.097   153.938   201.062   254.469   314.159
```

```
function result = testwhile(start, max)

result = [];
r = start;
area = circle(r);
while area <= max
   result = [result area];
   r = r + 1;
   area = circle(r);
end
```

```
octave-3.0.0.exe:33> testwhile(5, 1000)
ans =

 Columns 1 through 7:

    78.540   113.097   153.938   201.062   254.469   314.159
380.133

 Columns 8 through 13:

   452.389   530.929   615.752   706.858   804.248   907.920

octave-3.0.0.exe:34> testwhile(50, 1000)
ans = [](0x0)
```

Pętle

- Matlab umożliwia stosowanie słów kluczowych break i continue

Typy

- Podstawowym typem jest wartość zmiennoprzecinkowa
- Format wyprowadzania danych (ale nie precyzję obliczeń) można zmienić poleceniem format

Łańcuchy tekstowe

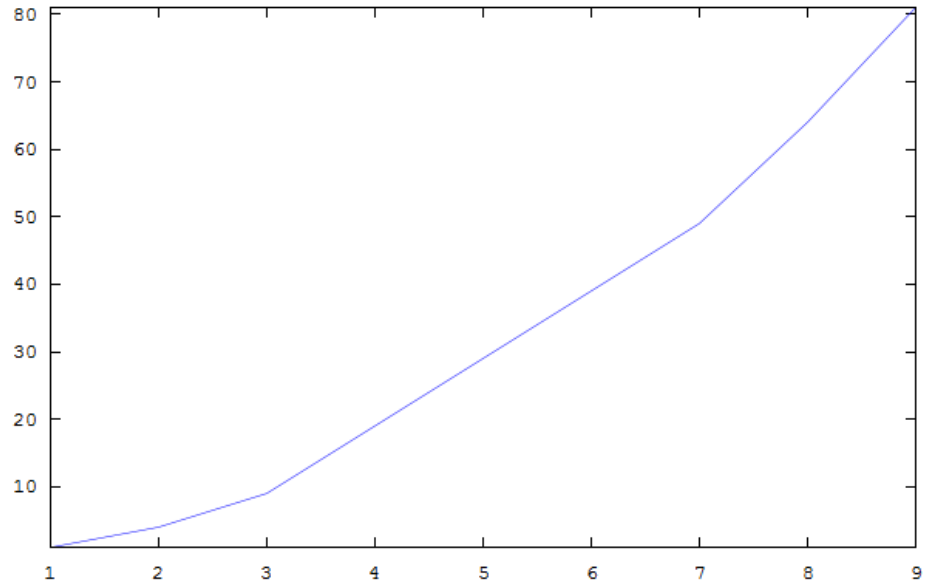
- Łańcuchy tekstowe są traktowane jako wektory
- Literały tekstowe umieszcza się w pojedynczych cudzysłowach

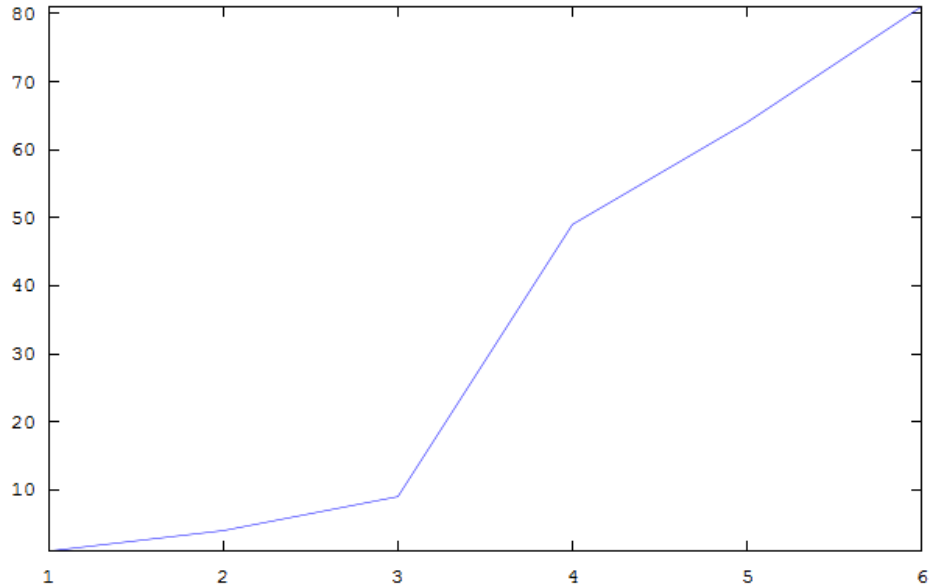
```
octave-3.0.0.exe:49> text = 'ala ma kota'
text = ala ma kota
octave-3.0.0.exe:50> text(3)
ans = a
octave-3.0.0.exe:51> text(5:11)
ans = ma kota
octave-3.0.0.exe:52> 2 * text
ans =

   194   216   194    64   218   194    64   214   222   232
194
```

Grafika

- Wykres we współrzędnych kartezjańskich tworzy się instrukcją plot
- Jeśli argument jest jeden, jako odcięte przyjmuje się indeksy wektora
- Jeśli argumenty są dwa, pierwszy to wektor odciętych, drugi - rzędnych

```
octave-3.0.0.exe:59> x = [1 2 3 7 8 9];
octave-3.0.0.exe:60> y = x.^2;
octave-3.0.0.exe:61> plot(x, y);
octave-3.0.0.exe:62> plot(y);
```

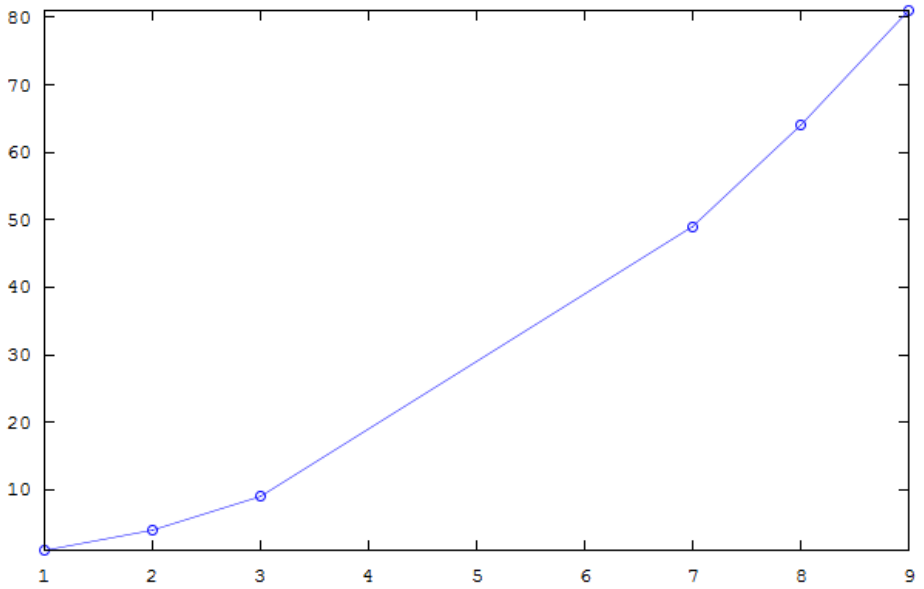


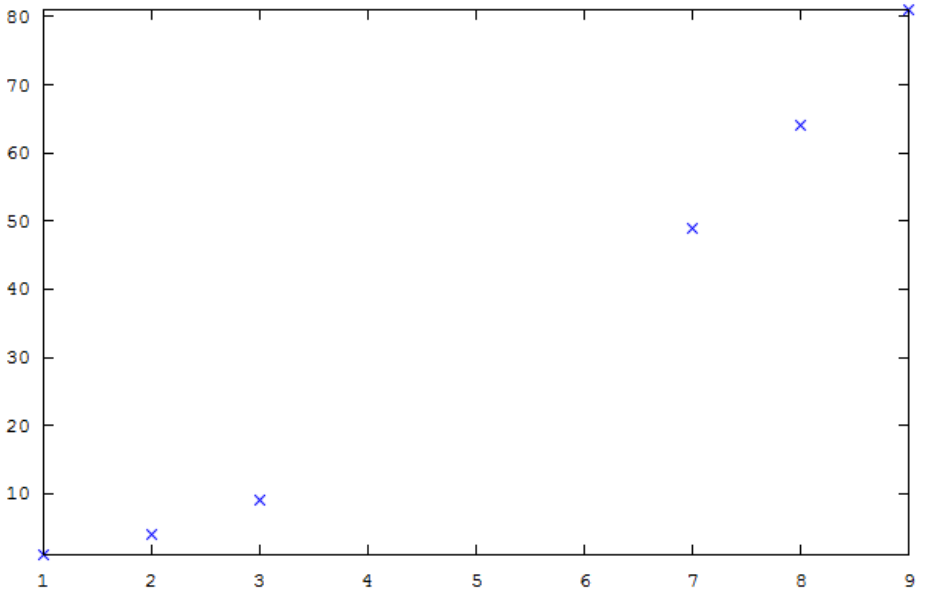


Grafika

- Aby zmienić wygląd linii można użyć argumentu format, np.

```
octave-3.0.0.exe:71> plot(x, y, '-o');
octave-3.0.0.exe:72> plot(x, y, 'x');
```

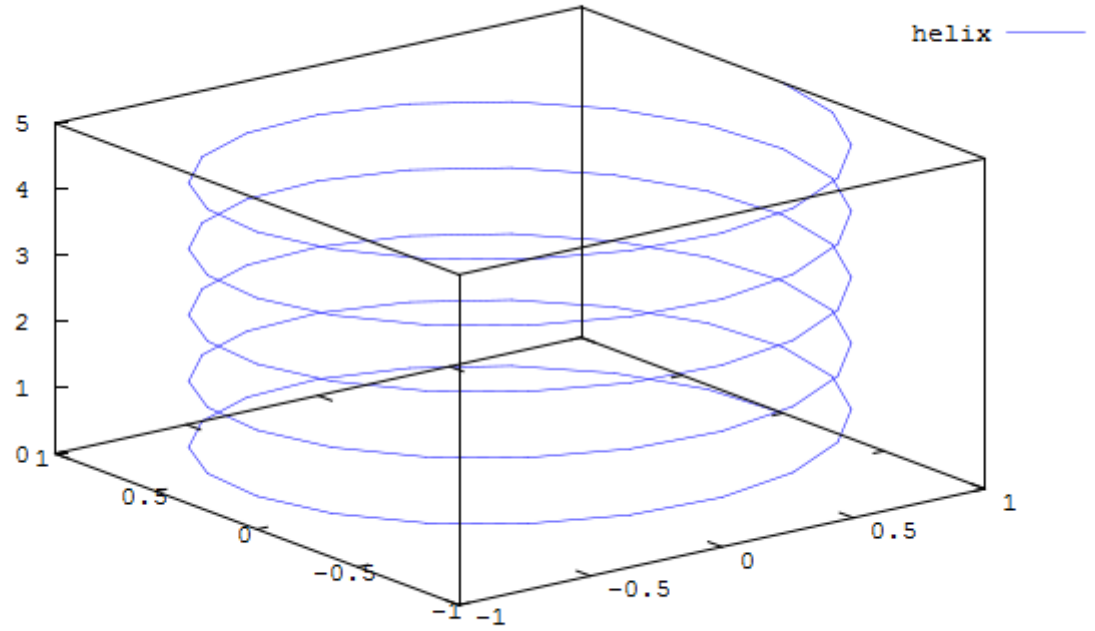




Grafika 3D

- Wykres w 3 wymiarach można uzyskać komendą plot3
- Argumentami są wektory zawierające współrzędne (x, y, z) kolejnych punktów wykresu

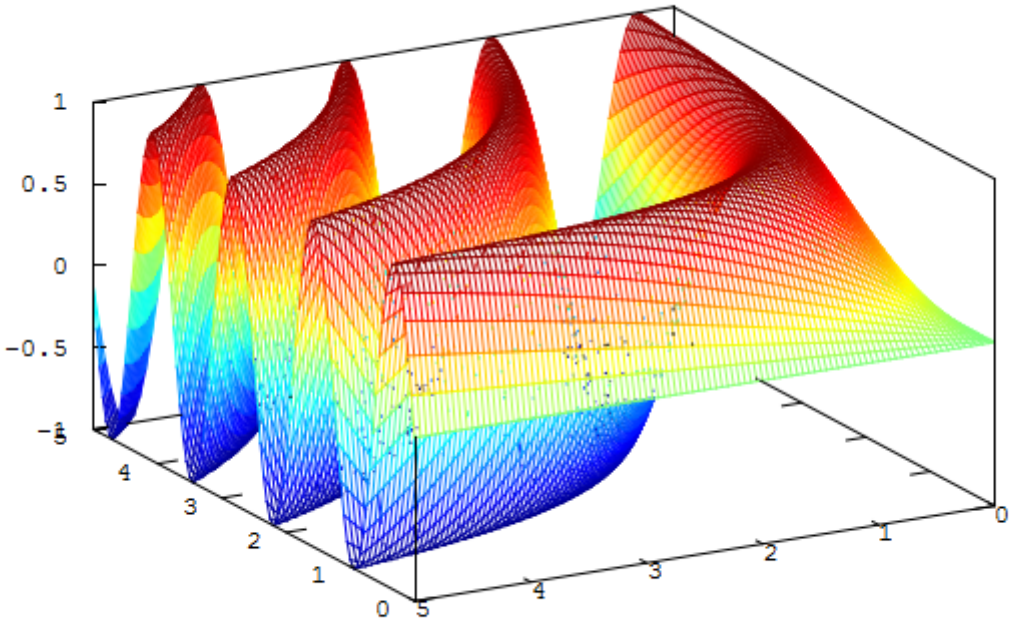
```
octave-3.0.0.exe:52> z = [0:0.05:5];
octave-3.0.0.exe:53> plot3 (cos(2*pi*z), sin(2*pi*z), z,
";helix;");
```


helix
5
4
3
2
1
0
1
0.5
0
-0.5
-1
-1
-0.5
0
0.5
1

Grafika 3D

- Aby wykreślić powierzchnię w trzech wymiarach, należy użyć instrukcji mesh
- Argumentami są:
  - wektor indeksów dla osi x (rozmiar m)
  - wektor indeksów dla osi y (rozmiar n)
  - macierz zawierająca wartości dla wszystkich par indeksów x i y (rozmiar m na n)

```
octave-3.0.0.exe:65> x = [0:0.05:5]';
octave-3.0.0.exe:66> y = x';
octave-3.0.0.exe:67> z = sin(x*y);
octave-3.0.0.exe:68> mesh(x, y, z)
```


1
0.5
0
-0.5
5
4
3
2
1
0
5
4
3
2
1
0

Operacje na plikach

- Matlab może czytać i zapisywać wiele formatów plików
- Podstawowym plikiem jest plik binary, w którym znajdują się zmienne (macierze)
- Do zapisu służy polecenie save, do odczytu - load
- save nazwa_pliku zmienna1 zmienna2 zmienna3
- load nazwa_pliku

```
octave-3.0.0.exe:17> a = [1 2; 3 4];
octave-3.0.0.exe:18> b = 7;
octave-3.0.0.exe:19> save test.dat a b
octave-3.0.0.exe:20> clear
octave-3.0.0.exe:21> a
error: `a' undefined near line 21 column 1
octave-3.0.0.exe:21> b
error: `b' undefined near line 21 column 1
octave-3.0.0.exe:21> load test.dat
octave-3.0.0.exe:22> a
a =

   1   2
   3   4

octave-3.0.0.exe:23> b
b =  7
```

Operacje na plikach

- Alternatywnie można stosować zapis/odczyt w formacie tekstowym
- Jest to wygodne przy imporcie danych z innych źródeł
- W przypadku zapisu nie są zapisywane nazwy zmiennych, poszczególne zmienne nie są od siebie oddzielane

```
octave-3.0.0.exe:36> save -ascii test.dat a b
octave-3.0.0.exe:37> clear
octave-3.0.0.exe:38> a
error: `a' undefined near line 38 column 1
octave-3.0.0.exe:38> b
error: `b' undefined near line 38 column 1
octave-3.0.0.exe:38> load test.dat
error: load: test.dat: inconsistent number of
columns near line 3
error: load: unable to extract matrix size
from file `test.dat'
```

```
1.00000000e+000 2.00000000e+000
3.00000000e+000 4.00000000e+000
7.00000000e+000
```

```
octave-3.0.0.exe:45> a = [1 2; 3 4];
octave-3.0.0.exe:46> c = [5 6; 7 8];
octave-3.0.0.exe:47> save -ascii test.dat a c
octave-3.0.0.exe:48> clear
octave-3.0.0.exe:49> a
error: `a' undefined near line 49 column 1
octave-3.0.0.exe:49> c
error: `c' undefined near line 49 column 1
octave-3.0.0.exe:49> load test.dat
octave-3.0.0.exe:50> ans
octave-3.0.0.exe:51> test
test =

   1   2
   3   4
   5   6
   7   8
```

```
1.00000000e+000 2.00000000e+000
3.00000000e+000 4.00000000e+000
5.00000000e+000 6.00000000e+000
7.00000000e+000 8.00000000e+000
```

Model

- Model jest uproszczeniem rzeczywistości
- Uproszczenie pozwala pominąć nieistotne (w danym momencie, aspekcie) szczegóły
- Jednocześnie pomaga uwypuklić kwestie istotne

W stronę UML

- Prace nad UML rozpoczęły się w 1994, kiedy Rumbaugh i Booch, obaj zatrudniani przez Rational Software Corporation, rozpoczęli prace nad unifikacją OMT i OOAD. Rezultat, Unified Method (UM) 0.8, został zaprezentowany w '95. W tym samym roku, Jacobson dołączył do Rational Software Corporation i wzbogacił UM elementami własnego OOSE, co zaowocowało UM 0.9 i UM 0.91 (oba w '96). Od tego momentu język ten jest znany jako **UML**.

Rozwój UML

- Wysiłki Rational Software Corporation zostały szybko wsparte przez istotne firmy, między innymi: IBM, DEC, HP, Oracle, Unisys, Microsoft. Doprowadziło to do dalszego rozwoju - wersji 1.0 w 1997. Wersja ta została później przekazana do Object Management Group (OMG). Wersja 1.1 powstała w tym samym roku. Była to wersja oficjalna do 2001 (wersja 1.4). Wersja 1.5 stała się oficjalna w 2003.

Diagramy UML

- Model w UML jest graficzną reprezentacją systemu
- Reprezentacja składa się z logicznie połączonych diagramów
- Wersja 2.0 zawiera 13 typów diagramów
- Istotnym pojęciami są pojecia **klasyfikatora** (classifier) – abstrakcyjnej kategorii która uogólnia kolekcję wystąpień mających te same cechy, oraz **wystąpienia** (instance) – egzemplifikacji klasyfikatora

Diagramy przypadków użycia

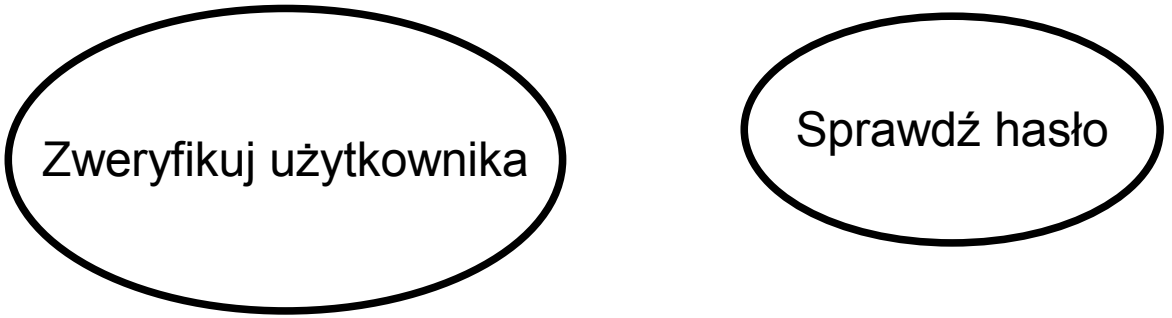
- Umożliwiają:
  - Identyfikację i dokumentację wymogów
  - Analizę zakresu aplikacji
  - Komunikację pomiędzy twórcami, właścicielami, klientami itd.
  - Opracowanie projektu przyszłego systemu, organizacji
  - Określenie procedur testowych dla systemu
- Są dwa typy diagramów przypadku użycia:
  - biznesowe
  - systemowe

Diagramy przypadków użycia

- Zawierają:
  - Przypadki użycia
  - Aktorów
  - Związki (relacje)

Przypadki użycia

- Specyfikacja sekwencji akcji (i ich wariantów) które system może wykonać poprzez interakcję z aktorami tego systemu
- Przypadek użycia jest spójnym fragmentem funkcjonalności systemu
- Nazwą jest rozkaz wypełnienia określonej funkcji. Nazwa jest umieszczona w owalu

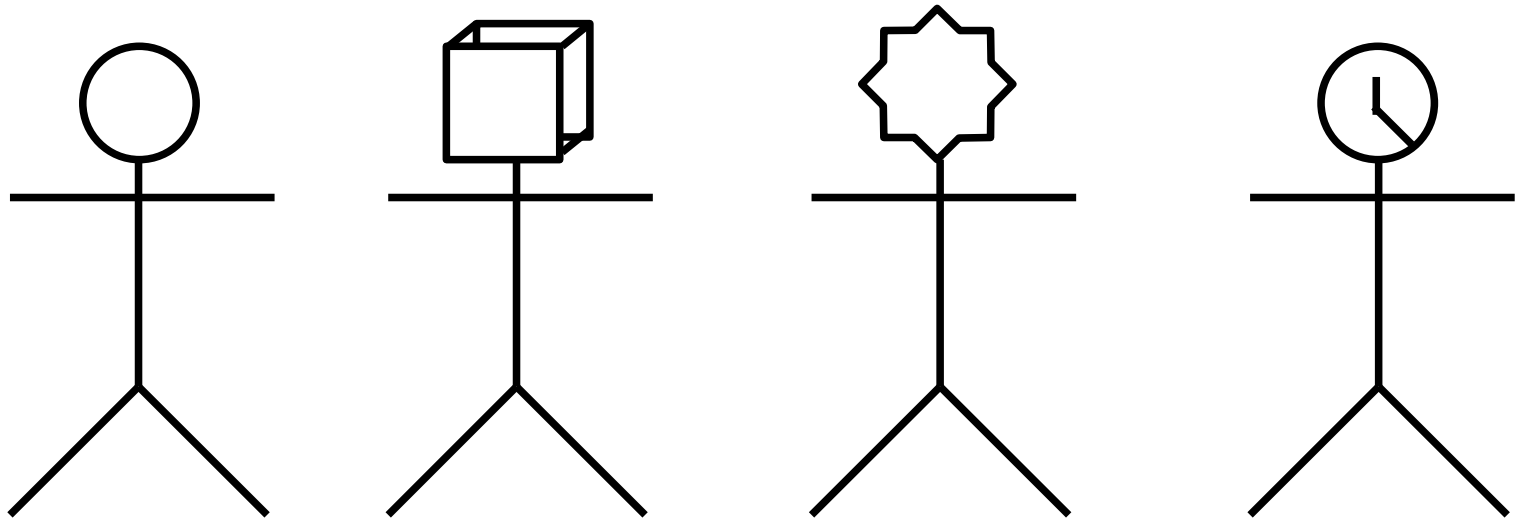


Aktor

- Aktor jest spójnym zbiorem ról odgrywanych przez użytkowników podczas interakcji z przypadkami użycia
- Aktorami mogą być:
  - Osoby (pojedyncza osoba, grupa, organizacja itp.)
  - Zewnętrzne systemy (programowe bądź sprzętowe)
  - Czas
- Nazwa to rzeczownik odzwierciedlający rolę odgrywaną w systemie
- Aktor może uzywać wielu przypadków użycia, przypadek użycia może być używany przezwielu aktorów

Stereotypy aktorów

- Klasyczny symbol aktora może być stereotypowany aby rozróżnić między różnymi typami aktorów



klasyczny / człowiek

system zewnętrzny

urządzenie

czas

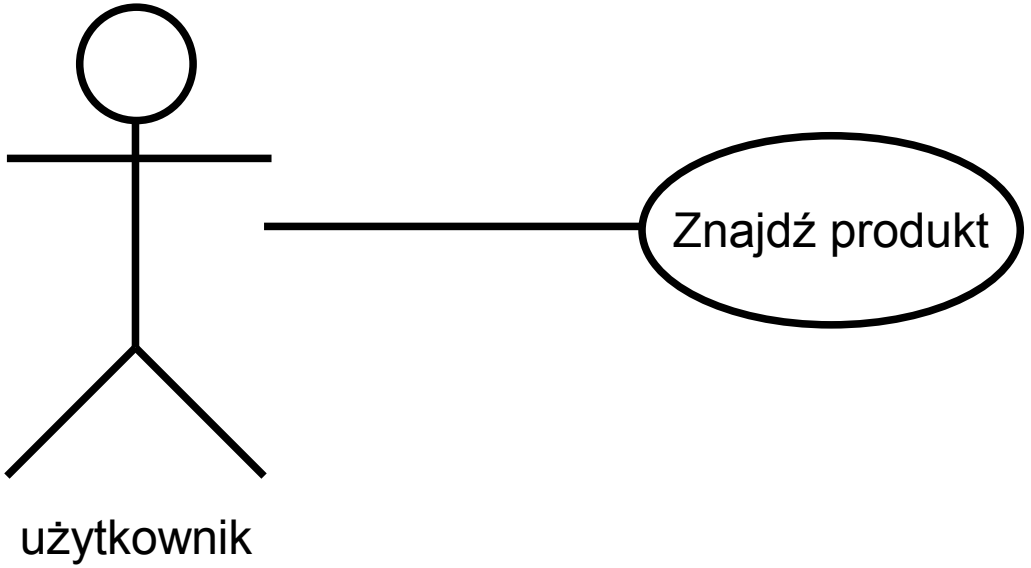
Związek (relationship)

- Wiąe ze sobą elementy diagramu (np. aktorów i przypadki użycia)
- Są 4 rodzaje związków:
  - Asocjacja (association)
  - Uogólnienie (generalisation)
  - Zależność (dependence)
  - Realizacja (realisation)

Asocjacja

- Asocjacja opisuje związek pomiędzy (dwiema lub więcej) wystąpieniami klasyfikatorów
- W diagramie przypadków użycia reprezentuje dwukierunkową komunikację pomiędzy aktorem i przypadkiem uzycia
- Jej reprezentacją graficzną jest ciągła linia
- Zazwyczaj nie są nazywane

Asocjacja



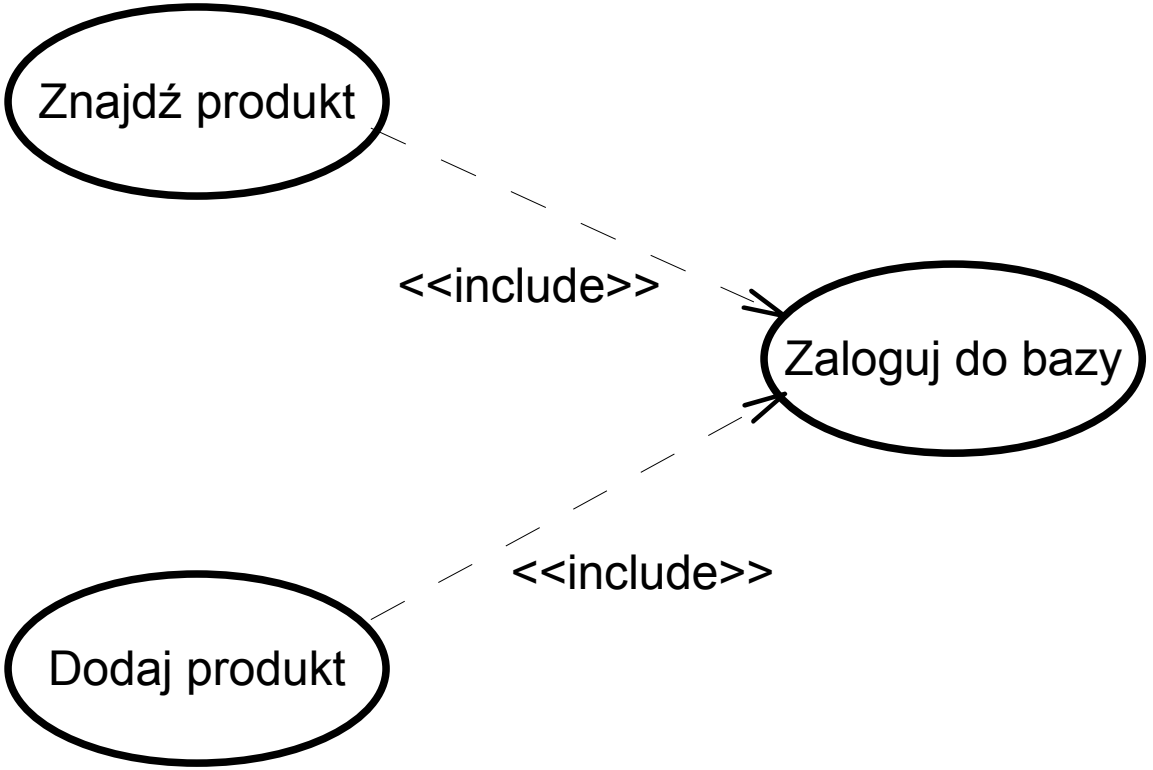
Zależność (dependency)

- Zależność jest związkiem pomiędzy dwoma elementami modelu gdzie zmiana w jednym elemencie (niezależnym one) ma wpływ na drugi element (zależny)
- Jest obrazowana jako linia przerywana zakończona otwartą strzałką
- W diagramach przypadków uzycia zależność jest stereotypowana w:
  - Zależność <<include>>
  - Zależność <<extend>>

Zależność <<include>>

- Związek między przypadkiem zawierającym i zawieranym
- Przypadek zawierany jest wykonywany zawsze gdy wykonywany jest przypadek zawierający – i tylko wtedy
- Jest przydatna gdy kilka przypadków użycia zawiera tę sama wspólną część
- Strzałka skierowana jest od przypadku zawierającego do zawieranego

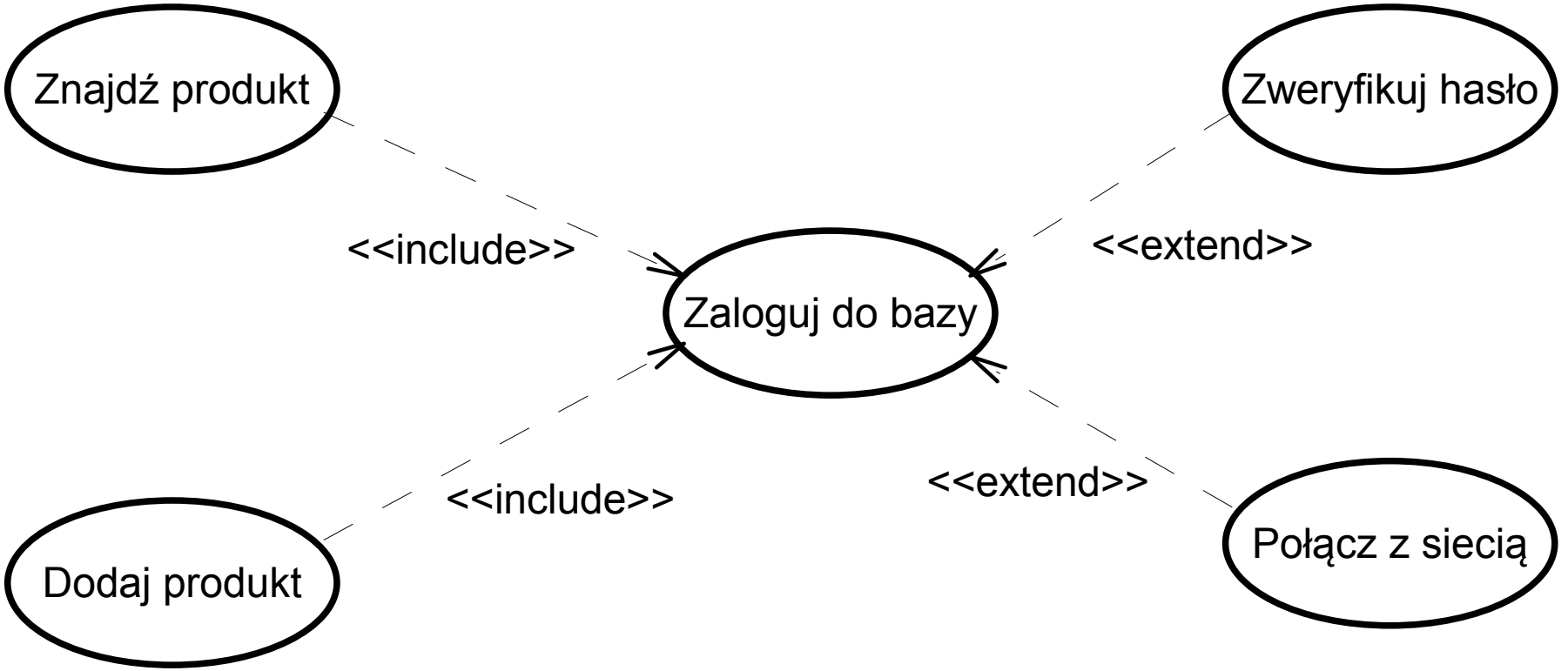
Zależność <<include>>



Zależność <<extend>>

- Zależność między przypadkiem podstawowym i przypadkiem który opcjonalnie może wprowadzić dodatkową funkcjonalność do przypadku podstawowego
- Jest przydatna gdy przypadek może, w pewnych warunkach, być uzależniony od innych przypadków
- Strzałka wskazuje od rozszerzenia do przypadku podstawowego

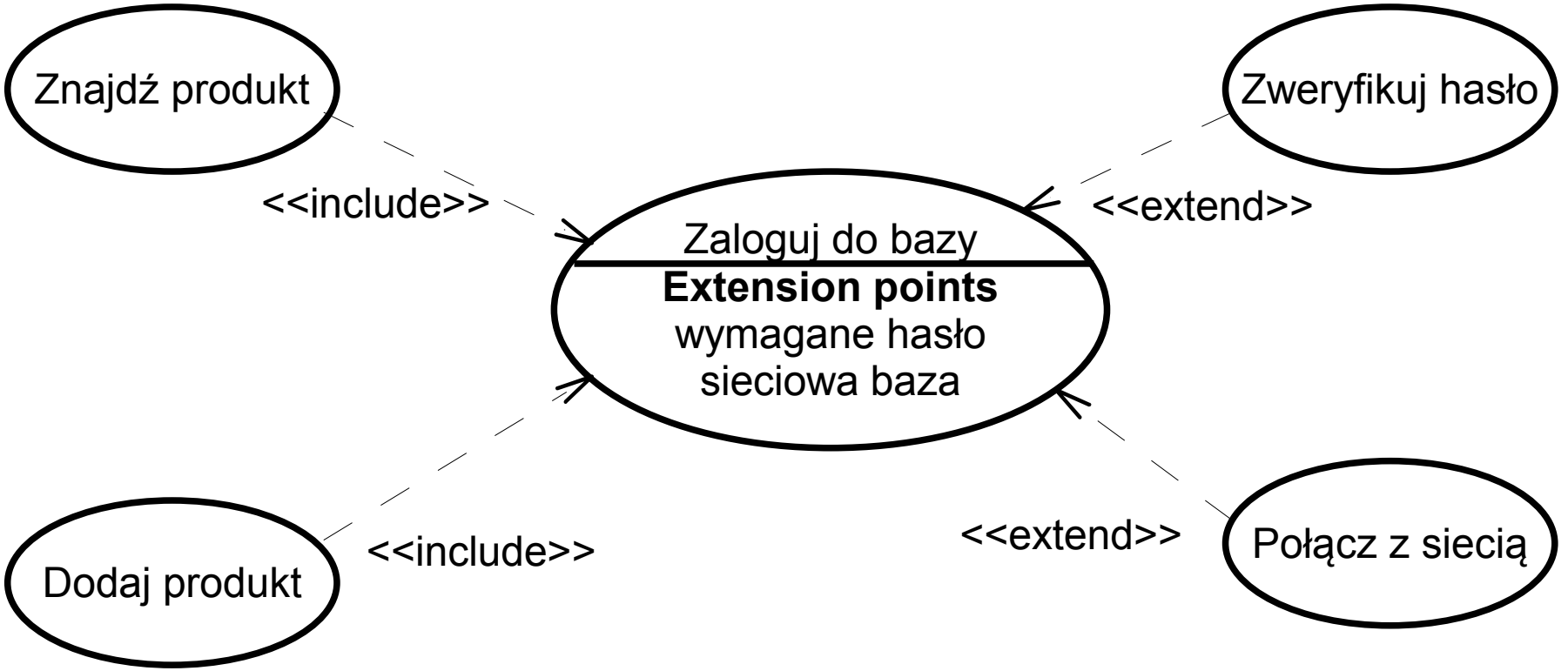
Zależność <<extend>>



Punkty rozszerzenia

- Jest możliwe określenie sytuacji (warunków) w których musi nastąpić dołączenie przypadków rozszerzających
- Są one wyszczególnione w rozszerzanym przypadku, pod poziomą linią

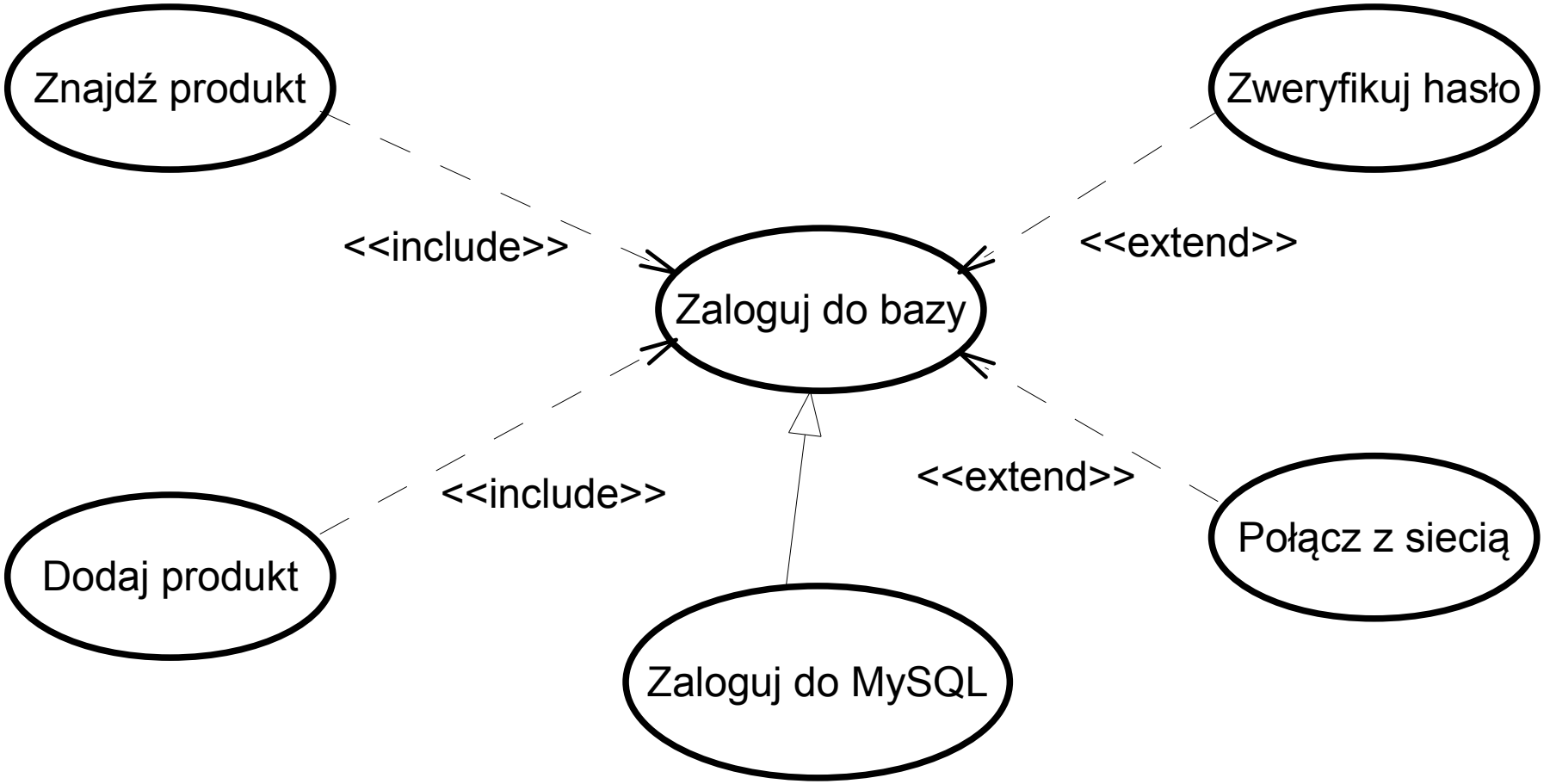
Punkty rozszerzenia



Uogólnienie

- Uogólnienie jest taksonomiczną relacją pomiędzy ogólnym i specjalizowanym klasyfikatorem
- Specjalizowany klasyfikator dziedziczy wszystkie cechy klasyfikatora ogólnego
- Jest obrazowana linią zakończoną zamkniętą strzałką, wskazującą w stronę klasyfikatora ogólnego

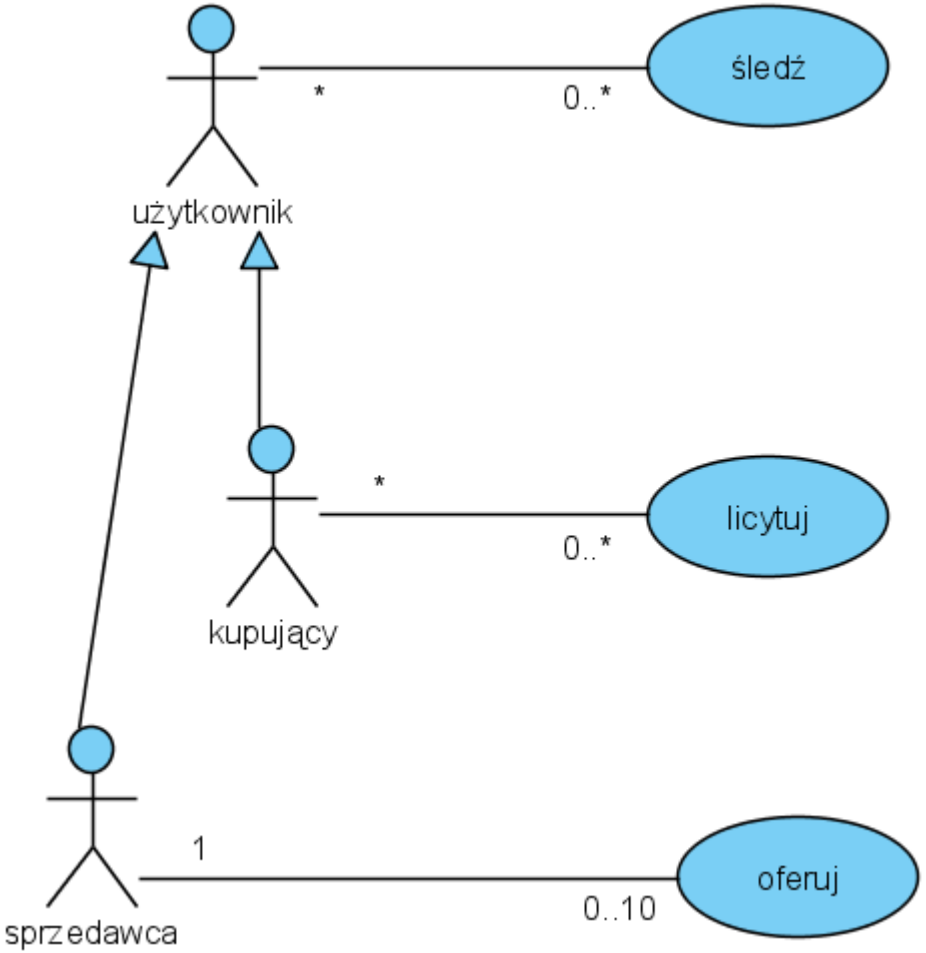
Uogólnienie



Liczebność

- Umożliwia określenie liczby elementów biorących udział w tej asocjacji, na każdym jej końcu
- Możliwe przypadki:
  - n (n > 0) dokładnie n
  - n..* (n ≥ 0) n lub więcej
  - n..m (m > n ≥ 0) od n do m
  - * wiele (nieznana liczba)
  - n, m, o..p, q (q > p...) lista wartości

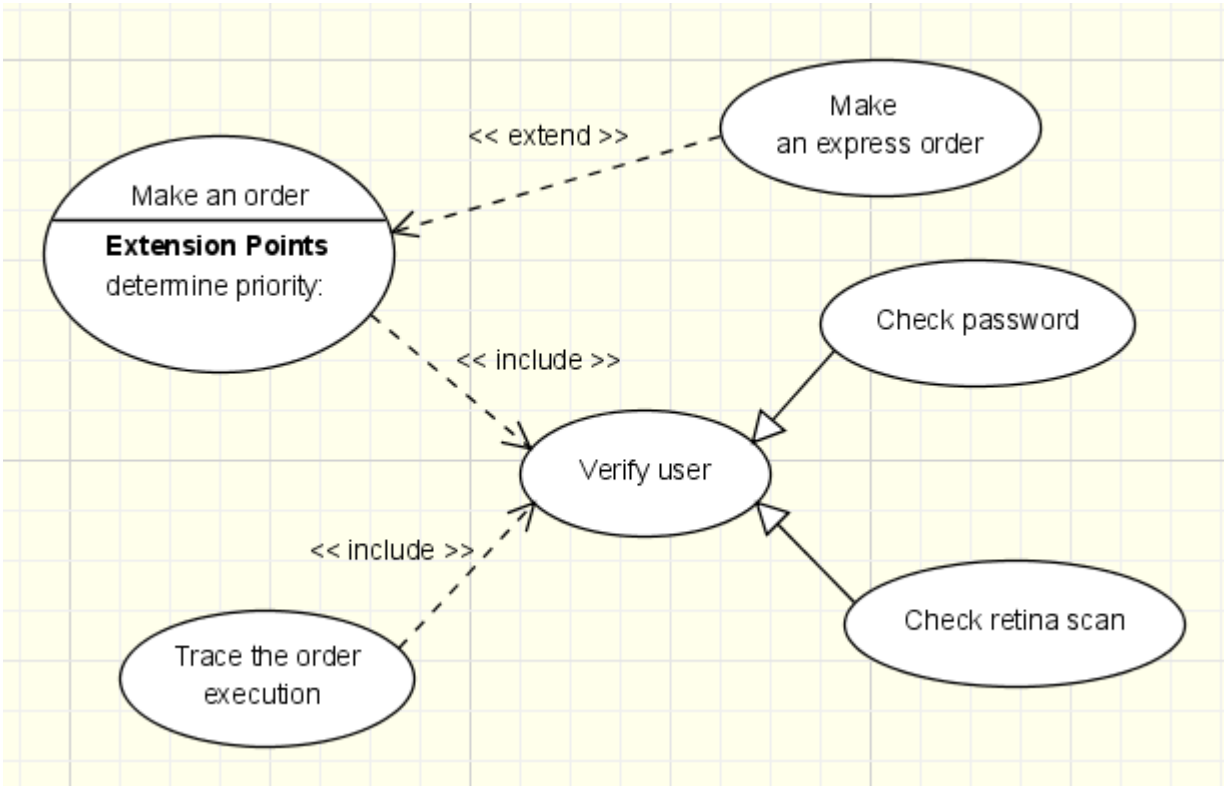
Liczebność


śledź
użytkownik
*
0..*
licytuj
kupujący
*
0..*
oferuj
sprzedawca
1
0..10

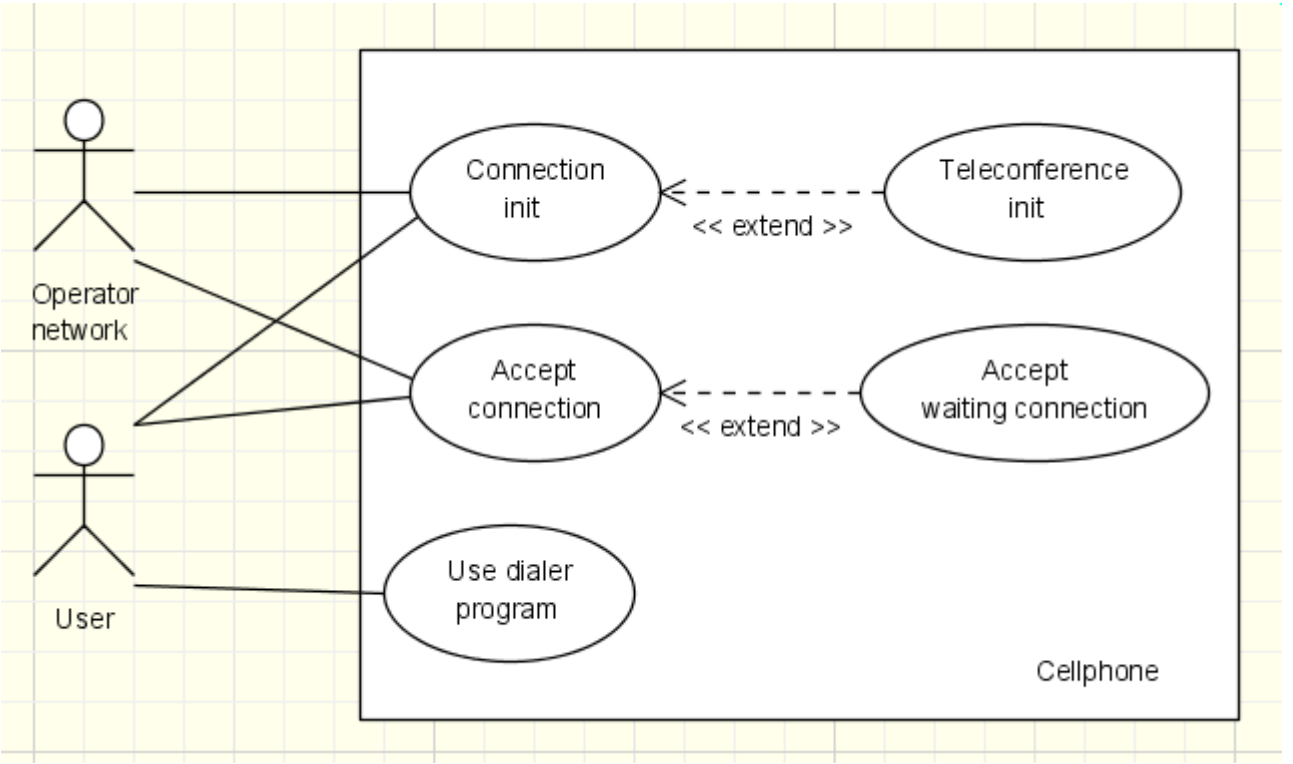
Dokumentowanie przypadków użycia

- Diagram przypadków użycia sam w sobie jest bardzo ogólnikowy
- Aby precyzyjnie określić pożądane zachowanie systemu, kazdy przypadkek uzycia powinien posiadać dodatkowa informaję, tzw. scenariusz
- Scenariusz jest sekwencją akcji, okreslająca zachowanie
- Dla złożonych przypadków można zdefiniować główny oraz alternatywne scenariusze
- Scenariusz moze zostaćzapisany w jezyku naturalnym, pseudo-kodzie, tabeli itp.

Przykładowe diagramy



Przykładowe diagramy

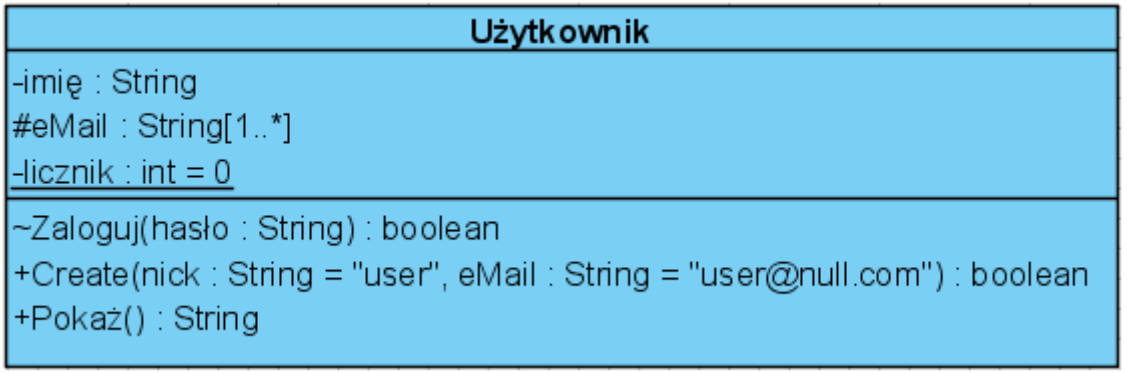


Diagramy klas

- Zawiera informacje o statycznych związkach między elementami (klasami)
- Są ściśle powiązane z technikami programowania zorientowanego obiektowo
- Są jednymi z istotniejszych diagramów w UML

Symbol klasy

- Symbolem klasy jest prostokąt, zwykle podzielony poziomymi liniami na trzy sekcje:
  - nazwy
  - atrybutów
  - operacji
- W razie potrzeby może zostać uzupełniony dodatkowymi sekcjami (np. wyjątków)

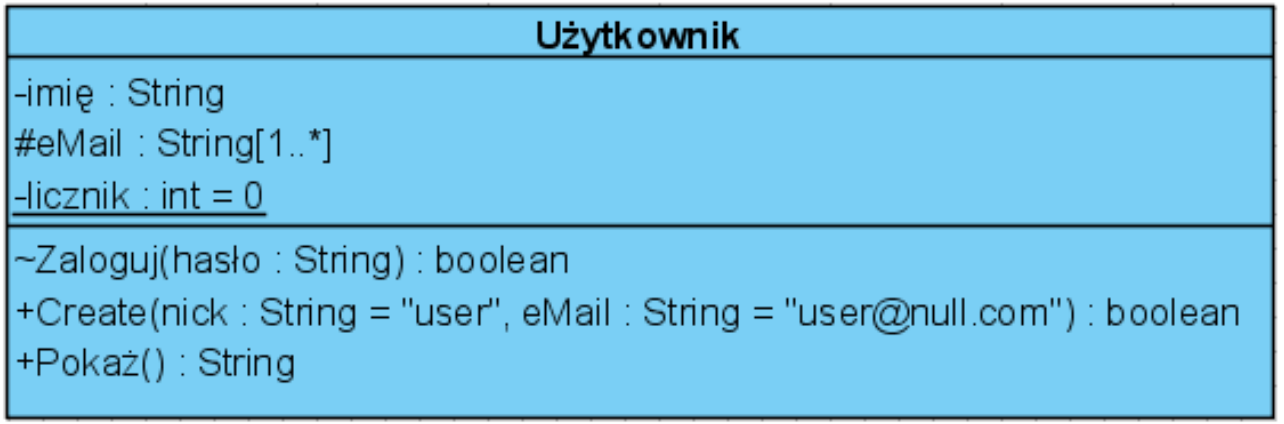


Symbol klasy

- Przy złożonych klasach wyswietlenie wszystkich atrybutów i operacji może zabrać zbyt dużo miejsca
- Możliwe rozwiązania to:
  - Wyświetlenie tylko nazwy klasy, bez sekcji atrybutów i operacji
  - Wyświetlenie tylko nazwy klasy, z pustymi sekcjami atrybutów i operacji
  - Wyświetlenie tylko części atrybutów lub operacji, zaznaczając kontynuację poprzez wielokropek
  - Ukrycie niektórych atrybutów lub operacji

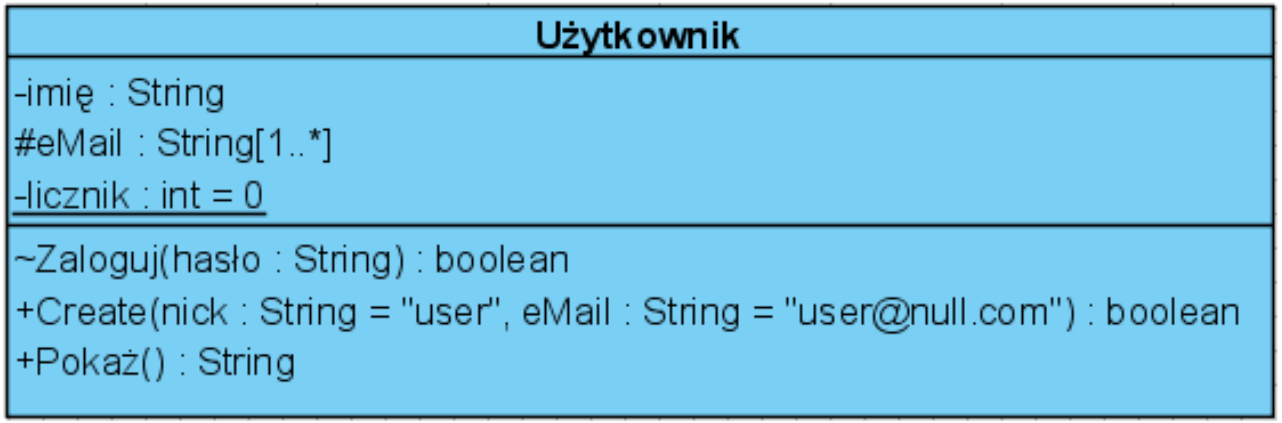
Kontrola dostępu

- Można określić modyfikatory dostepu dla składowych
- Są one ściśle powiązane z koncepcjami programowania zorientowanego obiektowo
- Możliwe rodzaje dostępu:
  - \+ publiczny
  - \- prywatny
  - # chroniony
  - ~ pakietu



Składniki statyczne

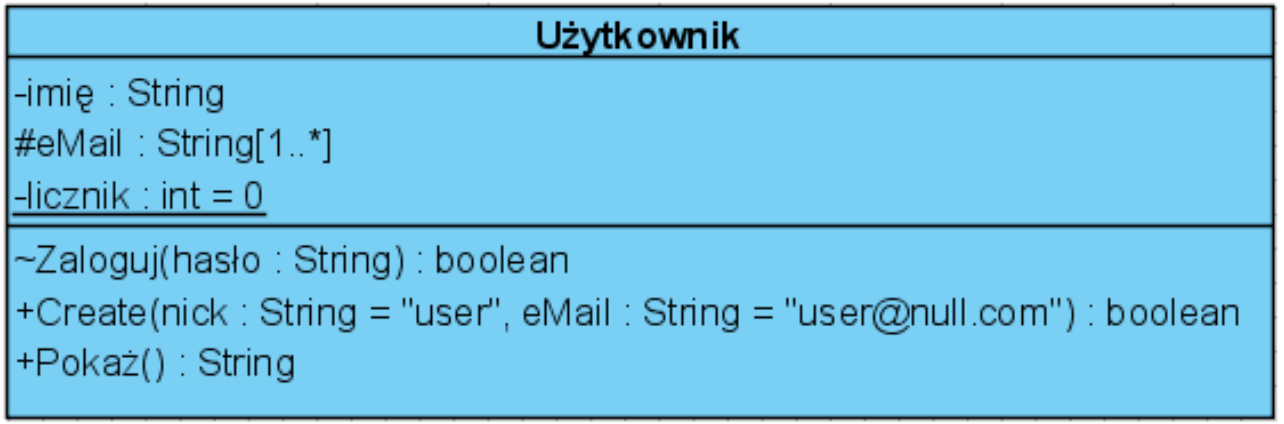
- Składniki mozna zadeklarować jako statyczne – działające na rzecz klasy, nie obiektu
- Koncepcja identyczna jak w językach zorientowanych obiektowo
- Reprezentacją graficzną jest podkreślenie



Specyfikacja składników

- Atrybuty mogą mieć określone:
  - Typ. Typ jest umieszczany po nazwie, oddzielony dwukropkiem
  - Liczebność
  - Wartość początkową
- Operacje mogą mieć określone:
  - Typ zwracany. Typ jest umieszczany po nazwie, oddzielony dwukropkiem
  - Argumenty. Każdy argument może być określony tak jak atrybut, z dodatkowym oznaczeniem kierunku przekazywania wartości (domyślnie “in”)

Specyfikacja składników



Związki

- Wszystkie 4 typy związków są używane
- Głównym typem jest asocjacja
- Może mieć następujące cechy (pogrubiono nowe w stosunku do diagramów przypadków użycia):
  - **nazwa**
  - **role**
  - kierunek nawigacji
  - liczebność
  - **agregacja**

Nazwa

- Można nazwać asocjację aby doprecyzować jej znaczenie
- Nazwa może zawierać kierunek

Role

- Inny sposób doprecyzowania asocjacji
- Rola klasy jest określona przez tekst umieszczony w pobliżu tej klasy
- Można określić jednocześnie nazwę i role

Kierunek nawigacji

- Domyślnie asocjacja jest dwukierunkowa
- Aby była jednokierunkowa, dodaje się strałkę
- Oznacza to że **komunikacja jest** jednokierunkowa (inaczej niż diagramy przypadków użycia)

Liczebność

- Znaczenie identyczne jak w diagramach przypadków użycia

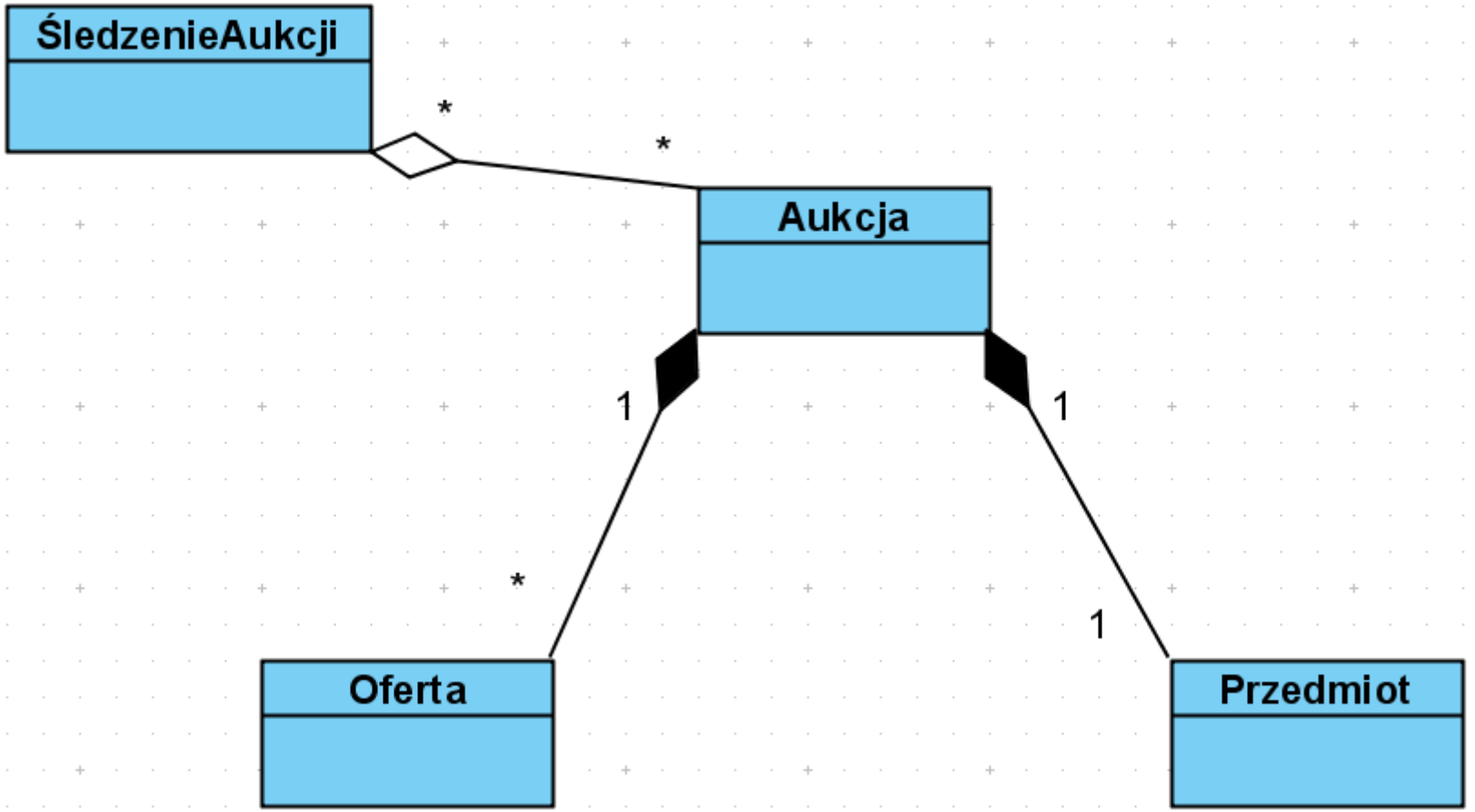
Agregacja

- Określa związek między całością i częścią
- Są dwa typy:
  - całkowita (kompozycja, silna agregacja)
  - częściowa (agregacja, słaba agregacja)
- Jest obrazowana przez romb umieszczony przy symbolu określającym całość
- Silna agregacja jest zobrazowana przez pełen romb, słaba – przez pusty

Silna i słaba agregacja

- W przypadku silnej agregacji części składowe nie mogą istnieć jeśli symbol określający całość jest usunięty
- W przypadku słabej agregacji jest to możliwe. Jeden obiekt może być też zawierany przez wiele innych.

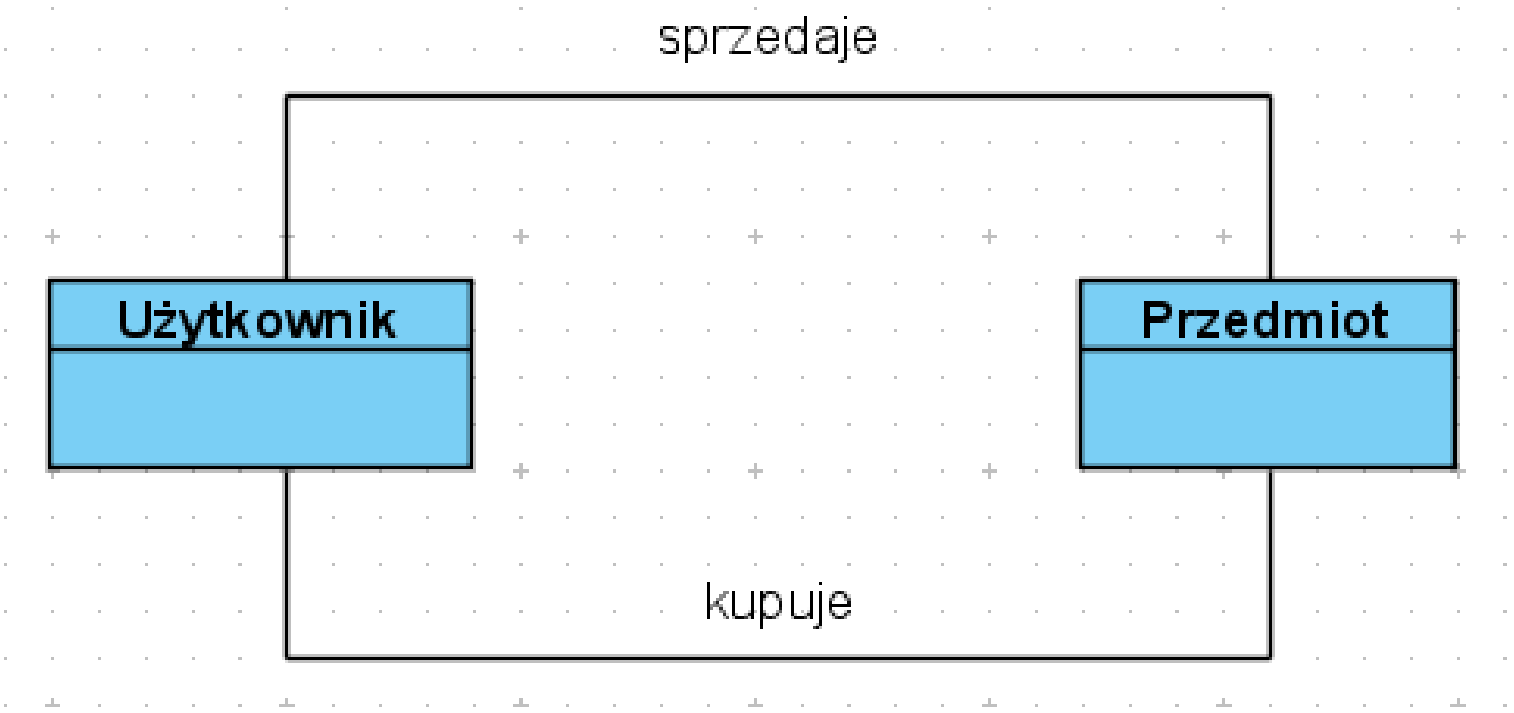
Silna i słaba agregacja



Asocjacja wielokrotna

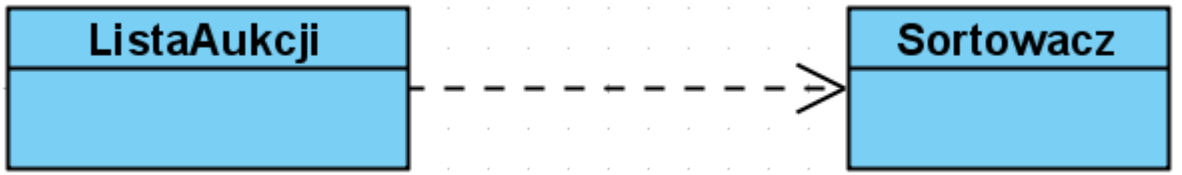
- Dwie klasy mogą być w odmienny sposób związane ze sobą w różnych kontekstach
- W efekcie może być więcej niż jedna asocjacja między klasami
- W takim wypadku każda powinna być nazwana

Multiple associations

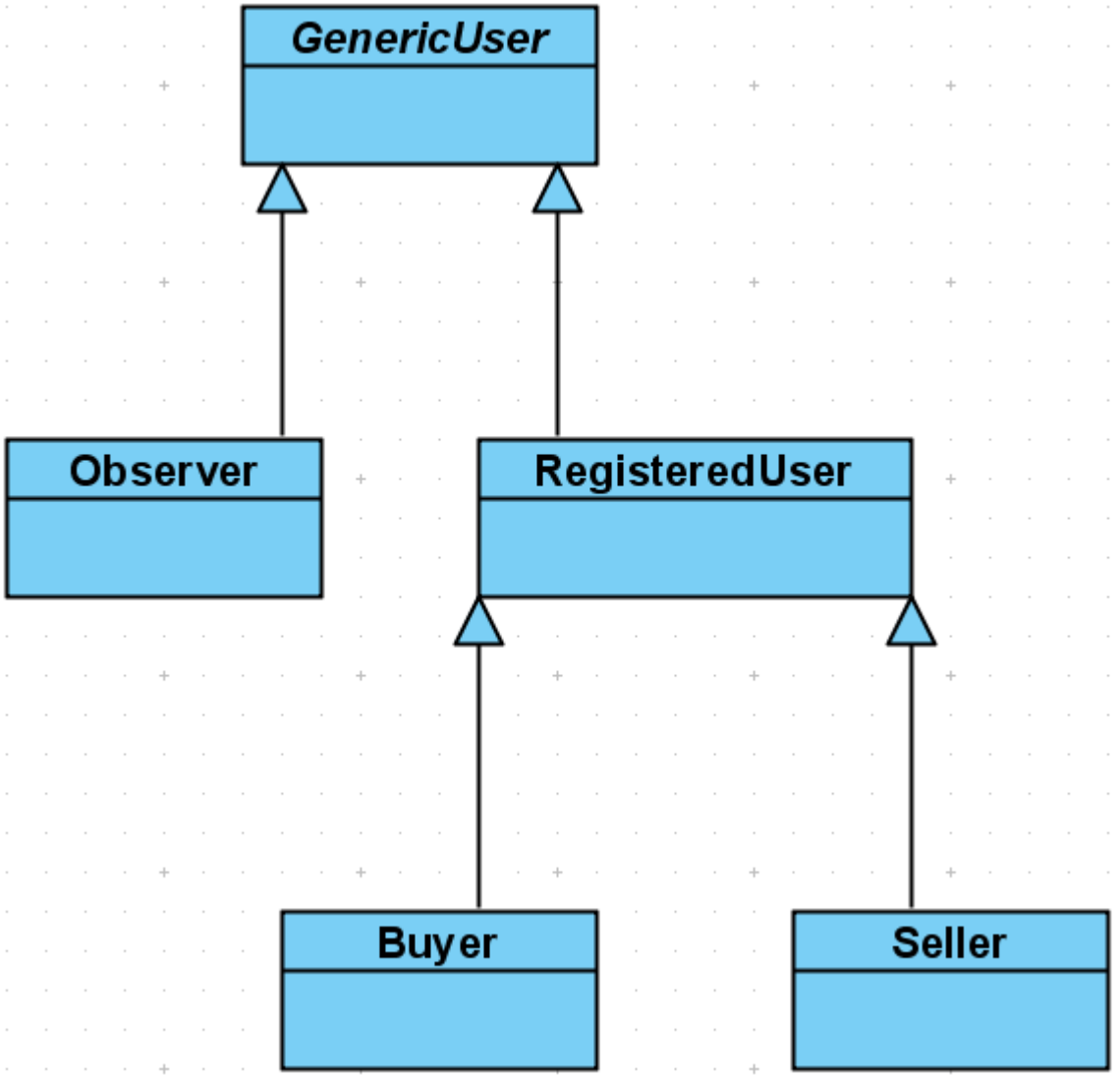


Zależność

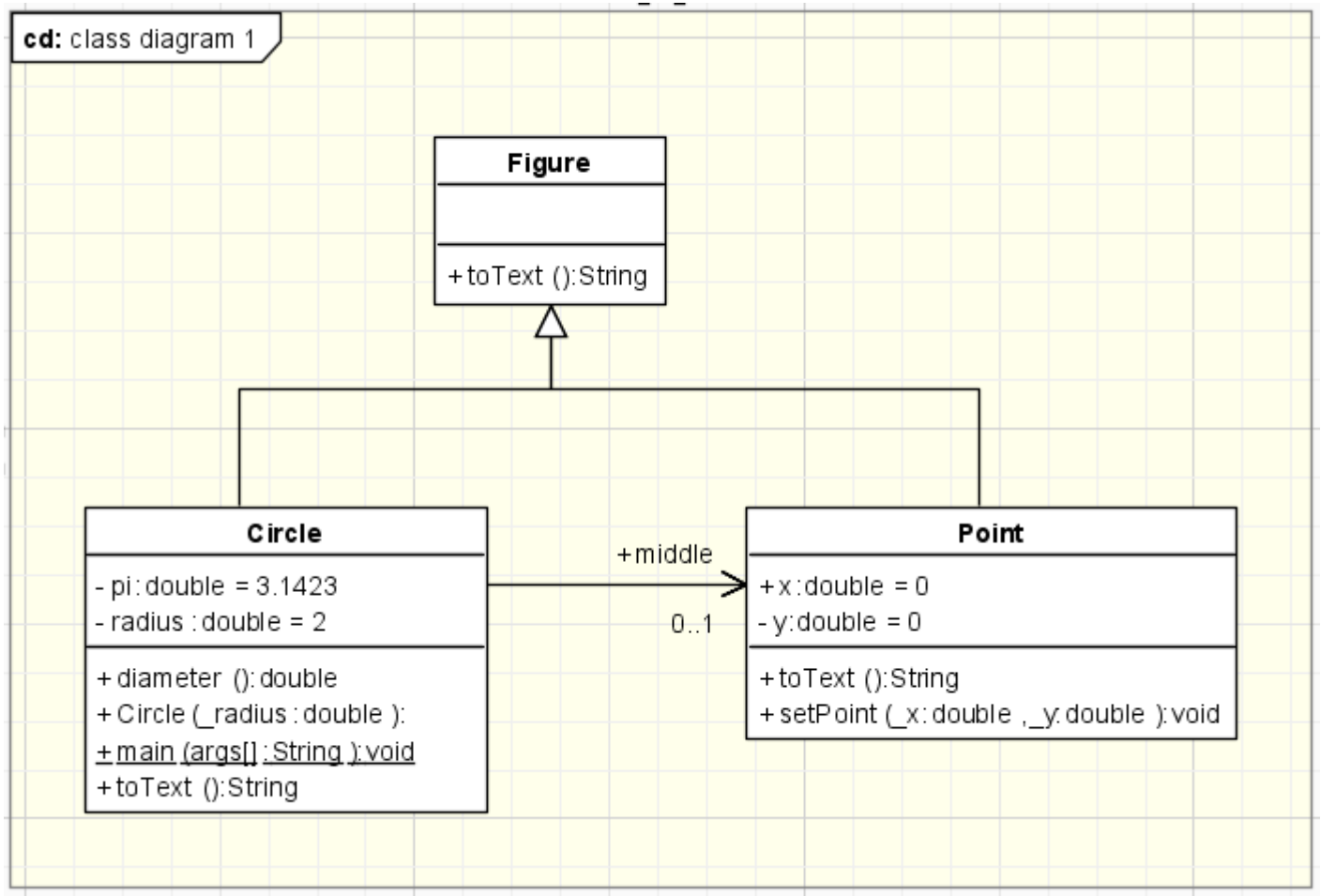
- Oznacza że jedna klasa (klient) w jakiś sposób używa innej klasy (dostawca)
- Jest obrazowana linią przerywaną zakończoną strzałką wskazującą na dostawcę



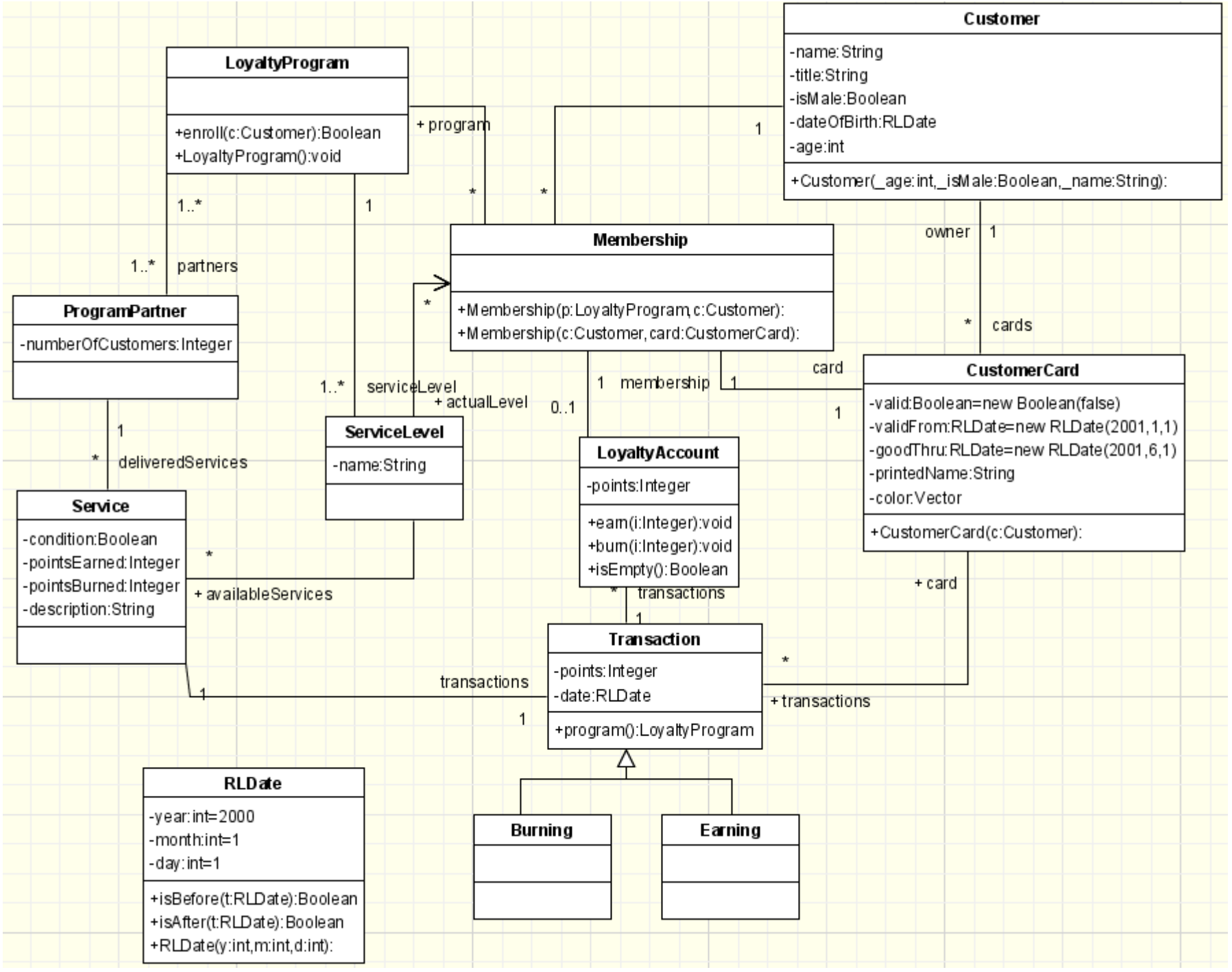
Uogólnienie



Przykładowe diagramy



Przykładowe diagramy



Diagramy czynności (activity)

- Opisują dynamikę systemu (klas- statykę)
- Prezentuja przepływ danych i sterowania
- Mogą być stosowane do modelowania:
  - Procesów biznesowych
  - Scenariuszy przypadków użycia
  - Algorytmów

Elementy składowe

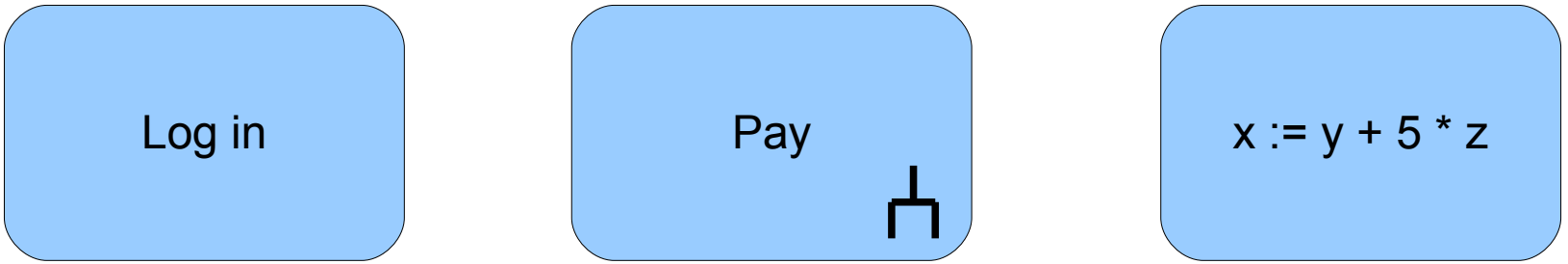
- Czynność
- Akcja
- Przepływ sterowania
- Początek
- Koniec
- Zakończenie przepływu

Czynność

- Reprezentuje złożony proces
- Może zostać (na osobnym diagramie) doprecyzowana, poprzez kolejny diagram czynności, opisujący jej “wnętrze” - powstaje struktura hierarchiczna
- Dekompozycja może przebiegać do poziomu akcji – najmniejszej, niepodzielnej jednostki

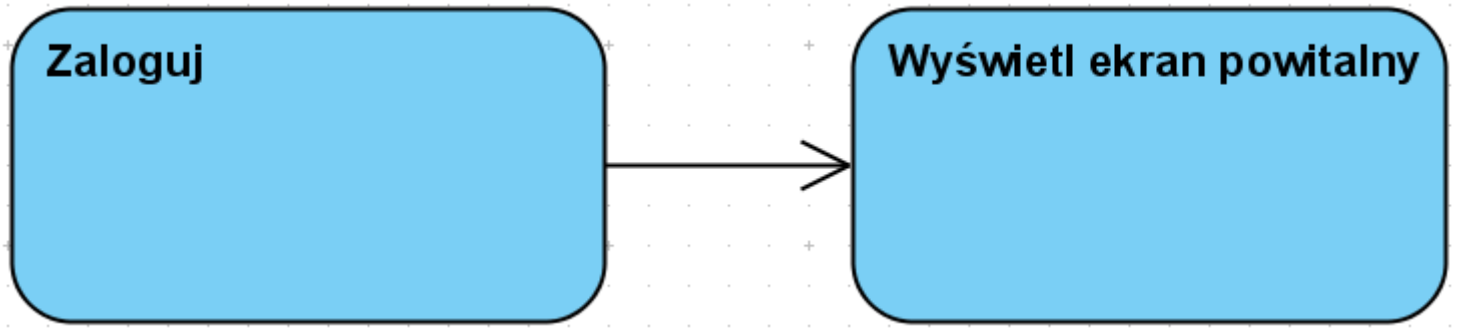
Czynność i akcja

- Symbolem jest prostokąt z zaokrąglonymi rogami
- Czasesm czynności posiadające powiązane z nimi poddiagramy posiadają specjalny znacznik



Przepływ sterowania

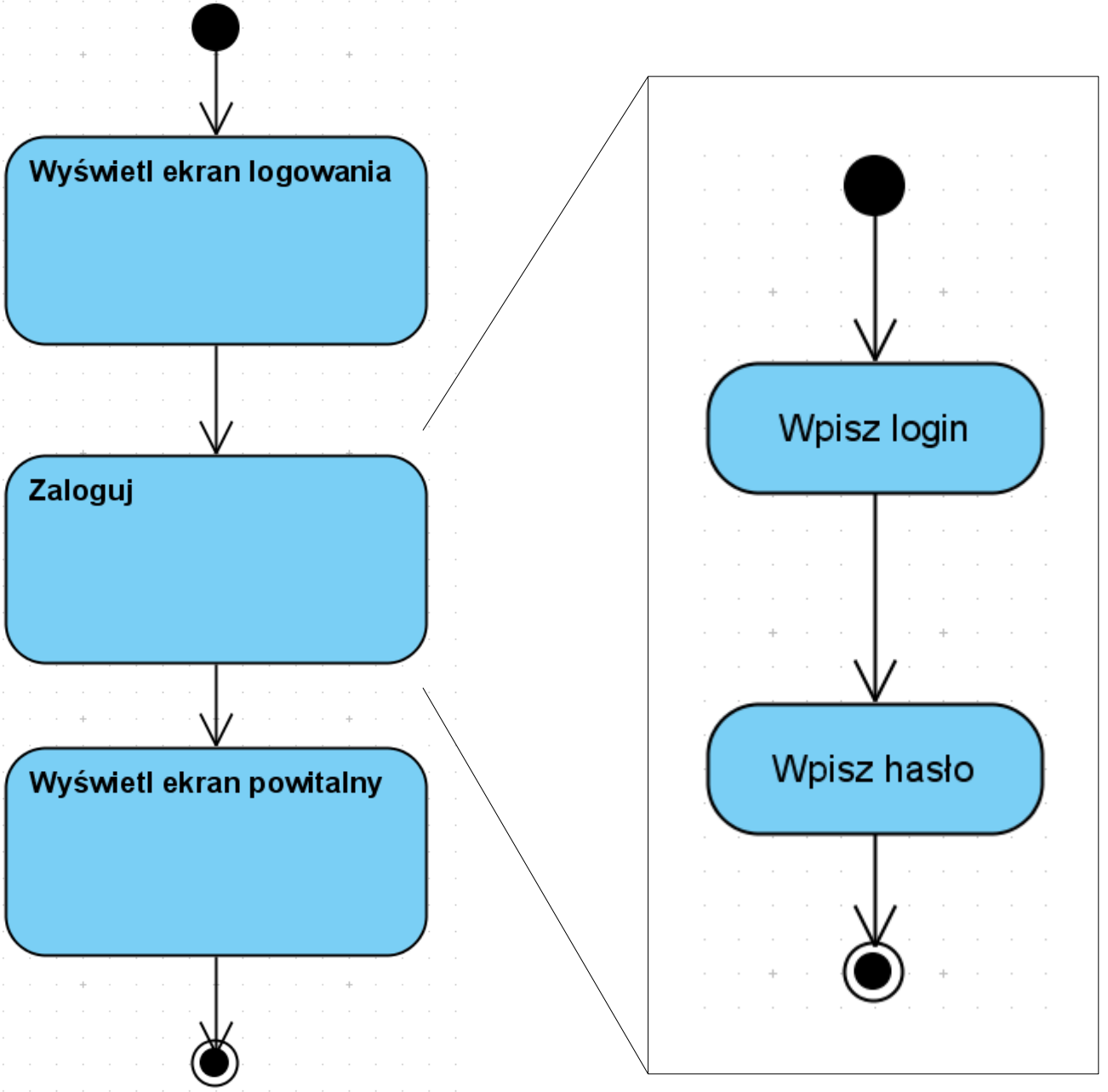
- Jest to związek między dwiema czynnościami/akcjami mówiacy, że po zakończeniu jednej sterowanie będzie przekazane do drugiej
- Symbolem jest strzałka



Początek, koniec, zakończenie przepływu

- Początek określa początek przepływu sterowania. Zwykle jeden na diagram. Symbolem jest pełne kółko
- Koniec oznacza zakończenie wszystkich przepływów w diagramie. Może być więcej niż jeden. Symbolem jest małe pełne kółko wewnątrz większego pustego
- Zakończenie przepływu oznacza koniec jednego przepływu. Może być więcej niż jedno. Symbolem jest kółko z krzyżykiem (X)

Prosty diagram



Decyzja i złączenie

- Alternatywne ścieżki przepływu sterowania mogą zostać opisane przy pomocy węzłów decyzji i złącznia
- Węzeł decyzji ma jeden przepływ wejściowy i wiele wyjściowych. Tylko jedno wyjście może być wybrane w danej chwili
- Węzeł złączenia ma wiele wejść i jedno wyjście
- Symbolem obu węzłów jest romb

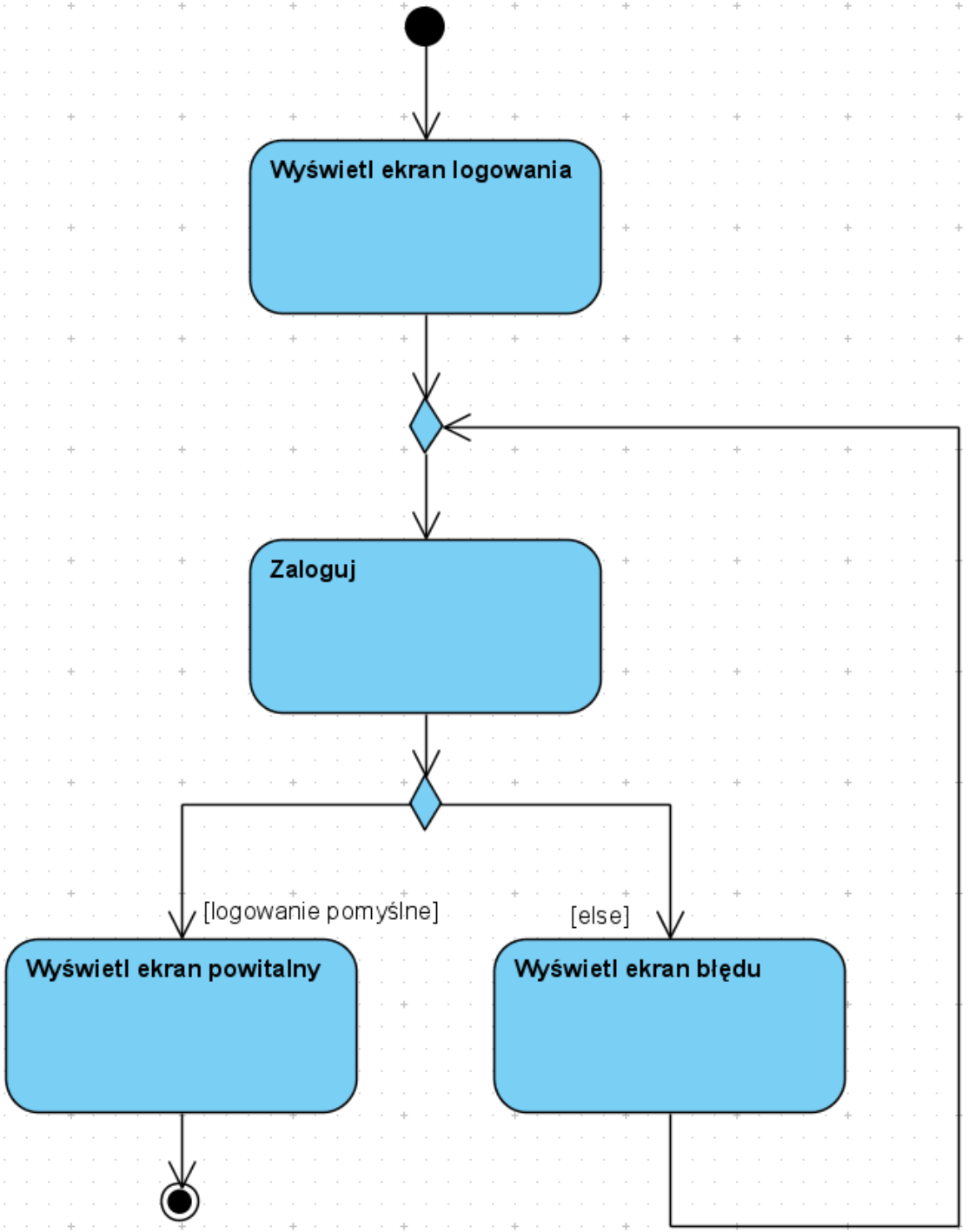
Decyzja

- Wybór wyjścia następuje na podstawie warunku
- Warunek jest umieszczany przy wyjściu, w nawiasach kwadratowych
- Wszystkie warunki muszą sę wzajemnie wykluczać
- Jeden z warunków może zostać zastąpiony słowem kluczowym *else*

Złączenie

- Każdy przepływ pojawiający się na wejściu zostanie natychmiast przekazany na wyjście – brak synchronizacji

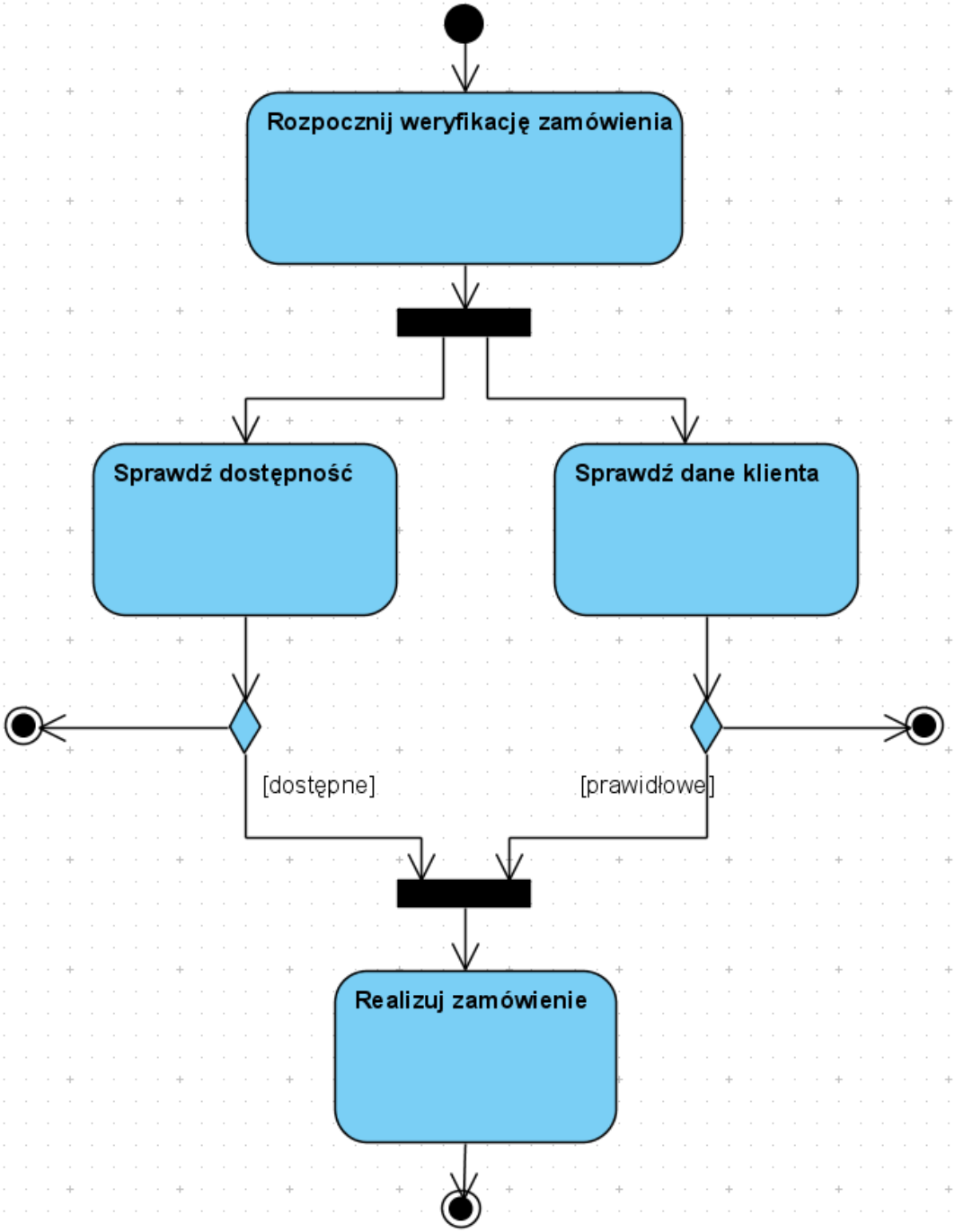
Przykładowy diagram



Przepływy równoległe

- Przepływy mogą być wykonywane równolegle
- Umożliwiają to węzły rozwidlenia i scalenia
- Rozwidlenie ma jedno wejście i wiele wyjść – przepływ wchodzący jest rozdzielany
- Scalenie ma wiele wejść i jedno wyjście. Przepływ zostanie przekazany do wyjścia tylko jeśli przepływy dotarły do wszystkich wejść – następuje synchronizacja

Sample diagram



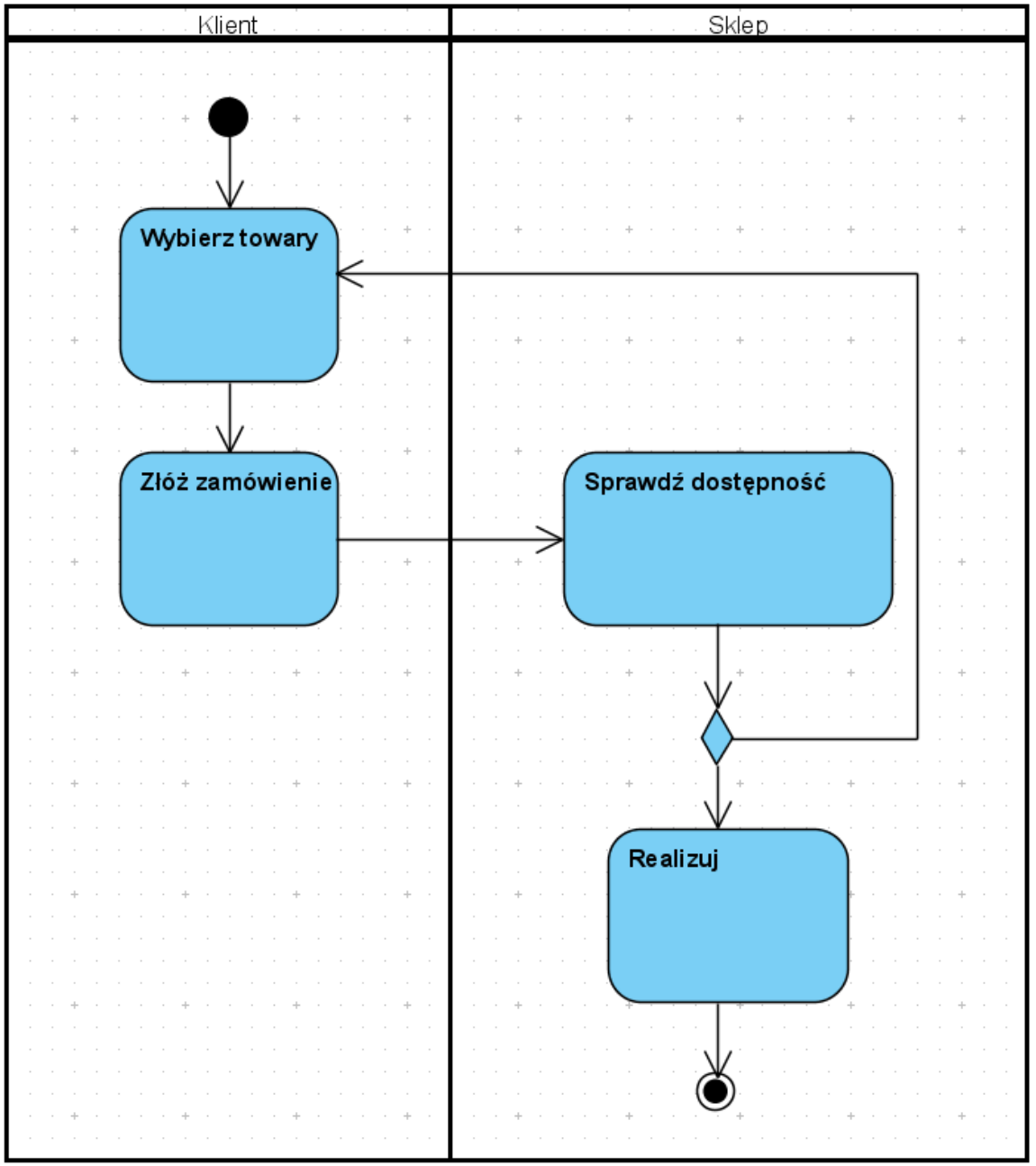
Przepływ obiektów

- Jako uzupełnienie można zaznaczyć przepływ obiektów
- Może być przydatny np. kiedy obiekt jest tworzony i przekazywany dalej do przetwarzania
- Obiekt musi być połączony z czynnością/akcją
- Symbolem jest prostokąt z nazwą
- Można też użyć tzw. przekaźników danych

Przepływ obiektów

Partycje

- Elementy diagramu mogą być pogrupowane w partycje

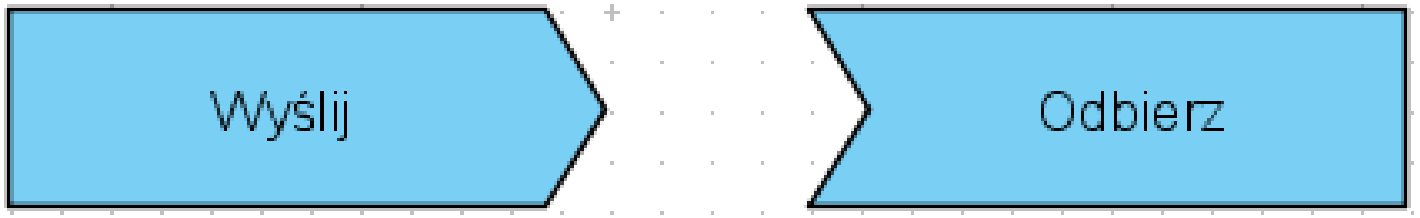


Obszar przerwania

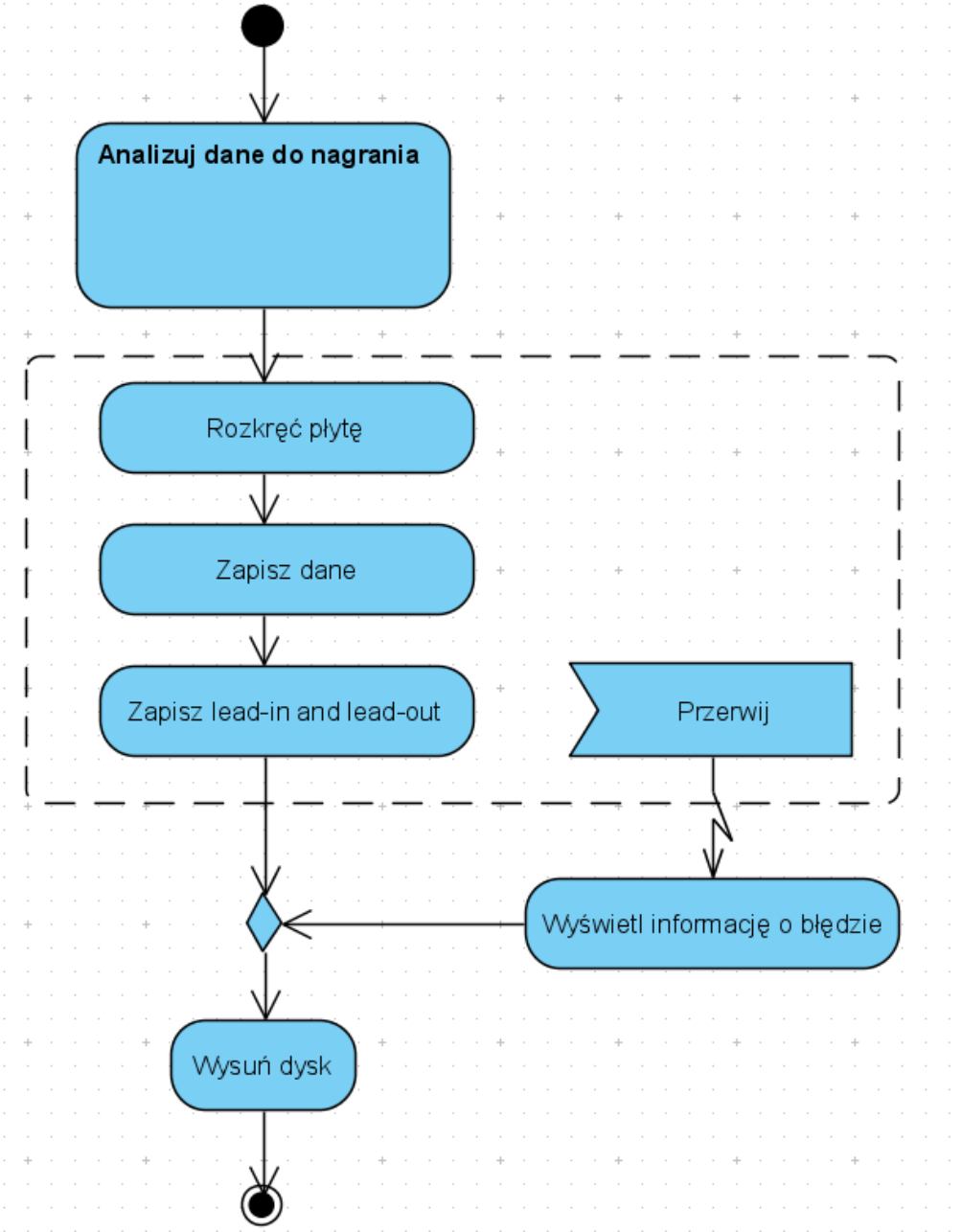
- Umożliwia wyznaczenie fragmentu diagramu którego wykonywanie może zostać natychmiast przerwane przy wystąpieniu zewnętrznego zdarzenia. Zostaje wtedy wykonany przepływ przerwania
- Wszystkie przepływy w regionie są przerywane, z wyjątkiem przepływu przerwania
- Przepływ przerwania zawsze zaczyna się w obszarze przerwania, kończy poza
- Użyteczne są symbole sygnałów

Sygnały

- Mogą służyć do opisu przetwarzania asynchronicznego
- Można:
  - Wysłać sygnał
  - Odebrać sygnał



Obszar przerwania



Przykładowe diagramy

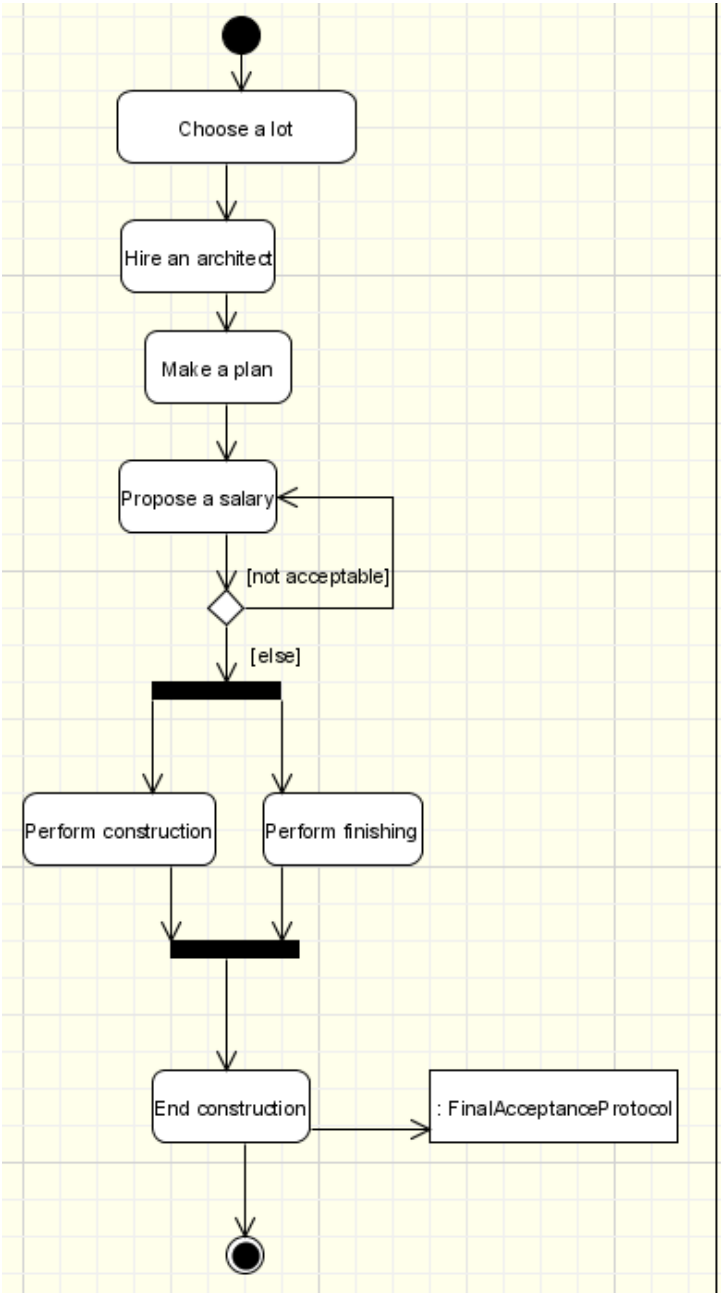


Diagram sekwencji (sequence)

- Opisuje interakcje między obiektami przy pomocy sekwecji wiadomości
- Dobrze nadaje się do dokumentowania przypadków użycia
- Diagram jest zorientowany w dwu wymiarach:
  - Oś pozioma związan jest z kolejnymi obiektami biorącymi udział w wymianie wiadomości
  - Oś pionowa związana jest z upływem czasu

Główne elementy

- Obiekt
- Linia życia
- Wiadomość

Obiekt

- Obiekty klas są podstawowymi elemntami w diagramie sekwencji
- Diagram może również zawierać instancję innych klasyfikatorów: przypadków użycia, aktorów, sygnałów itp.
- Obiekt jest przedstawiany jako prostokąt z nazwą (niekiedy podkreśloną)
- W prostych diagramach wszystkie obiekty umieszczone są przy górnej krawędzi diagramu

Linia życia

- Reprezentuje przedział czasu w którym obiekt istnieje
- Jest zobrazowna przez przerywana pionową linię, zaczynającą się na obiekcie i idącą w dół

Wiadomości

- Wiadomości reprezentują przepływ informacji między obiektami. Wiadomość jest poleceniem dla obiektu aby wykonać pewne operacje
- Najważniejszym (i koniecznym) składnikiem opisu wiadomości jest sygnatura

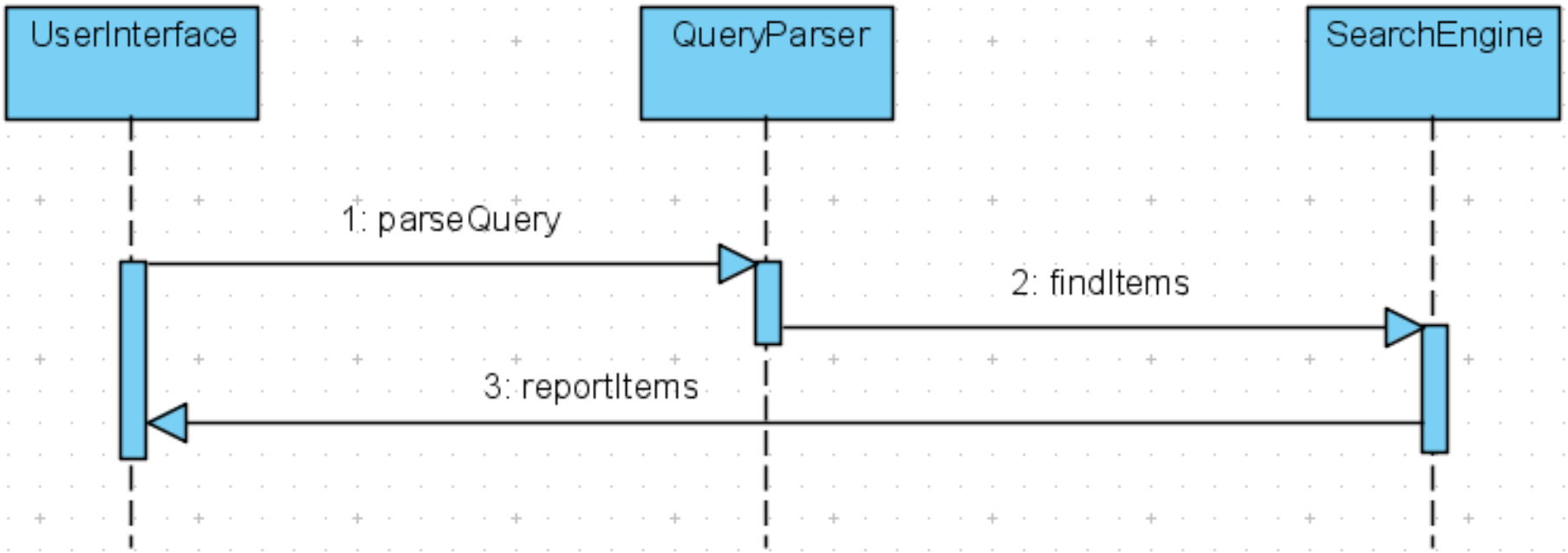
Wiadomość - sygnatura

- Może składać się z:
  - Nazwy (obowiązkowa)
  - Listy argumentów
  - Wartości zwracanej

Specyfikacja wykonania

- Obrazuje okres aktywaności obiektu (obliczenia, przekazywanie widomości z/do)
- Jest przedstawiany jako prostokąt umieszczony na linii życia, jego wysokość określa okres aktywności
- Początek jest związany z aktywacją obiektu (zwykle wiadomością przekazaną od innego obiektu)

Przykładowy diagram



Rodzaje wiadomości

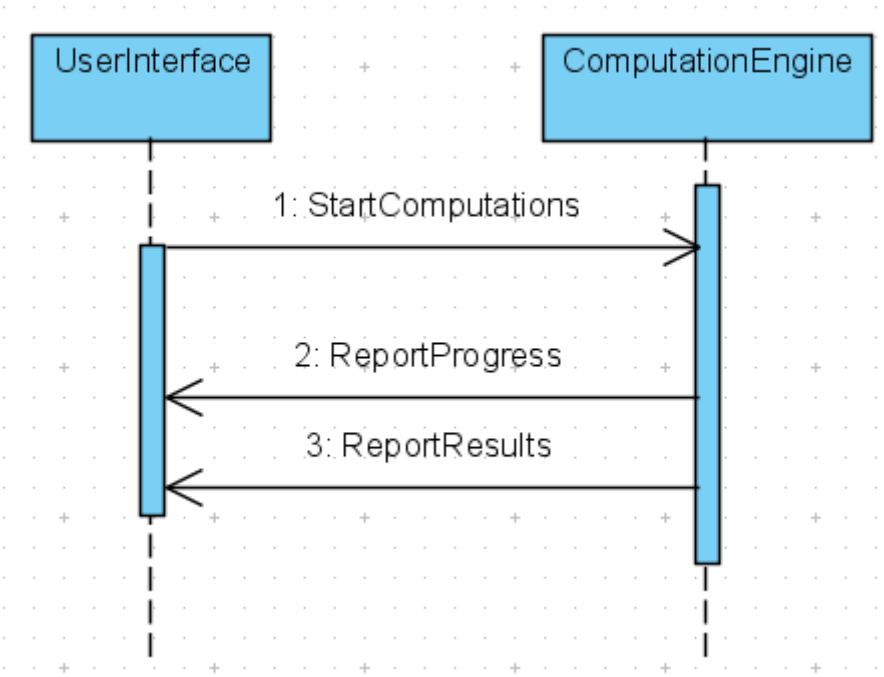
- Synchroniczna
- Asynchroniczna
- Zwrotna
- Zgubiona
- Znaleziona

Wiadomość synchroniczna

- Sterowanie jest przekazane do obiektu wywoływanego
- Przetwarzanie w obiekcie wywołującym jest wstrzymywane do momentu zakończenia wywołanej czynności
- Jest obrazowane pełną strzałką
- Jest odpowiednikiem wywołania funkcji

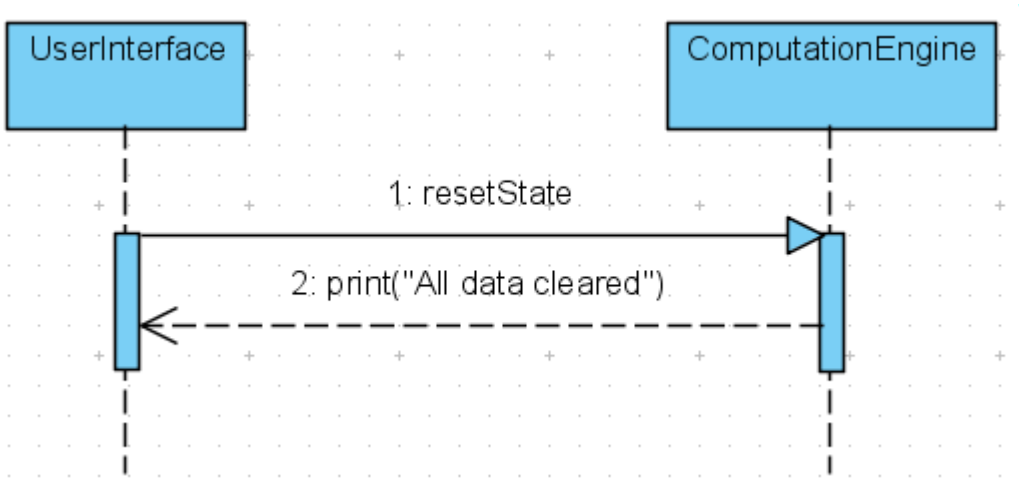
Wiadomość asynchroniczna

- Przetwarzanie w obiekcie wywołującym nie jest przerwane
- Jest obrazowane otwartą strzałką



Wiadomość zwrotna

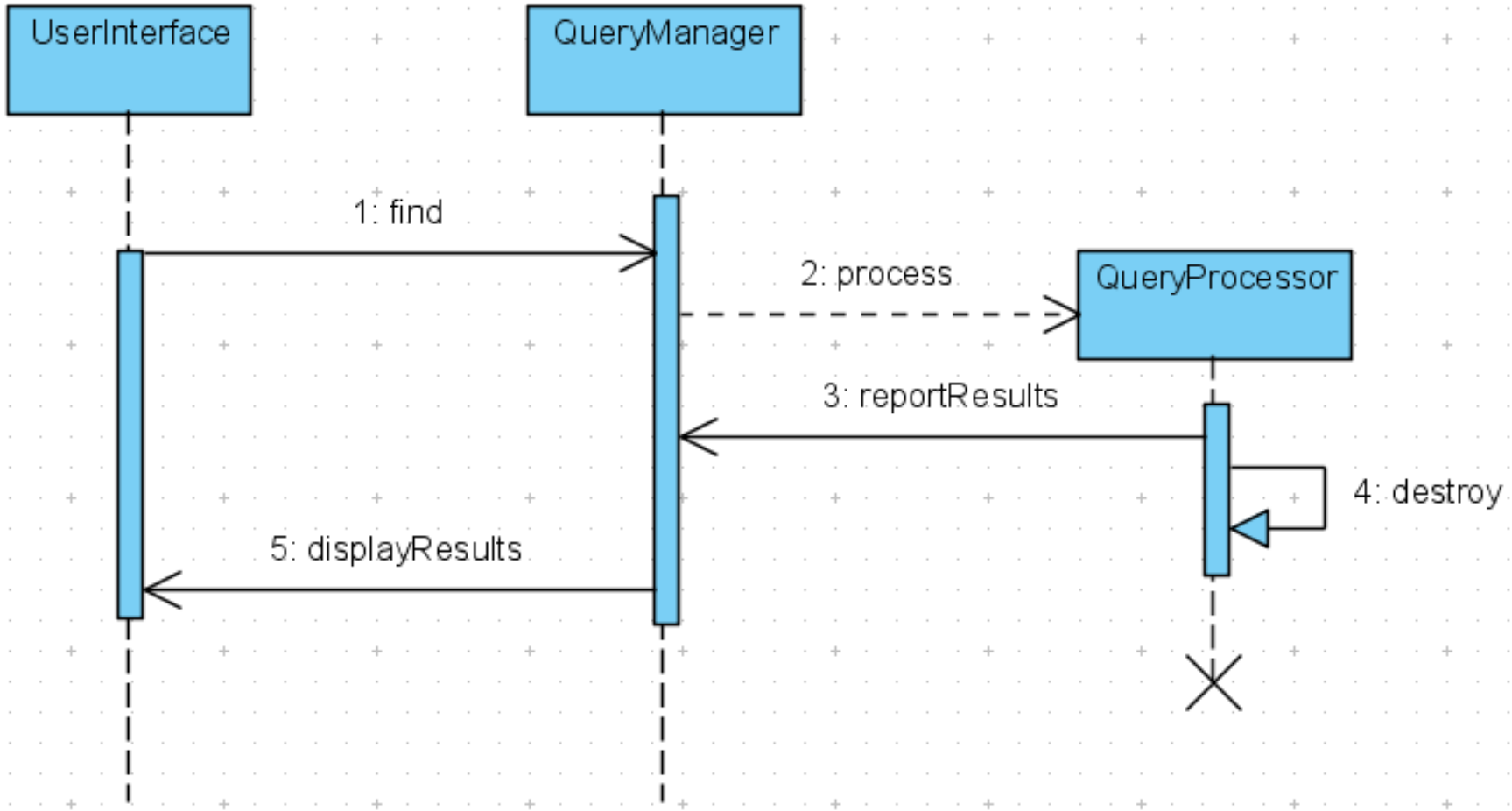
- Obrazuje oddanie sterowania do obiektu wywołującego
- Jest opcjonalna
- Jest przedstawiona jako strzałka z linią przerywaną



Tworzenie i niszczenie obiektów

- Tworzenie i niszczenie są powodowane przez odpowiednie wiadomości
- Wiadomości te mają stereotypy, odpowiedni “create” i “destroy”
- Na końcu wiadomości “create” umieszcza się tworzony obiekt (co powoduje że znajduje się on poniżej innych obiektów)
- Po otrzymaniu wiadomości “destroy” linia życia obiektu kończy się. Jest to dodatkowo wyróżnione umieszczeniem znaku X na końcu linii.

Przykładowe tworzenie i niszczenie



Fragmenty złożone

- Są to wybrane fragmenty diagramu sekwencji, do których odnosi się odpowiedni operator interakcji
- Są zobrazowane ramą otaczającą wybrany region. Rama ma nagłówek w lewym górnym rogu zawierający operator interakcji
- Niektóre operatory wymagają wyodrębnienia podfragmetów regionu. Są one wyodrębniane linią kropkowo-kreskową

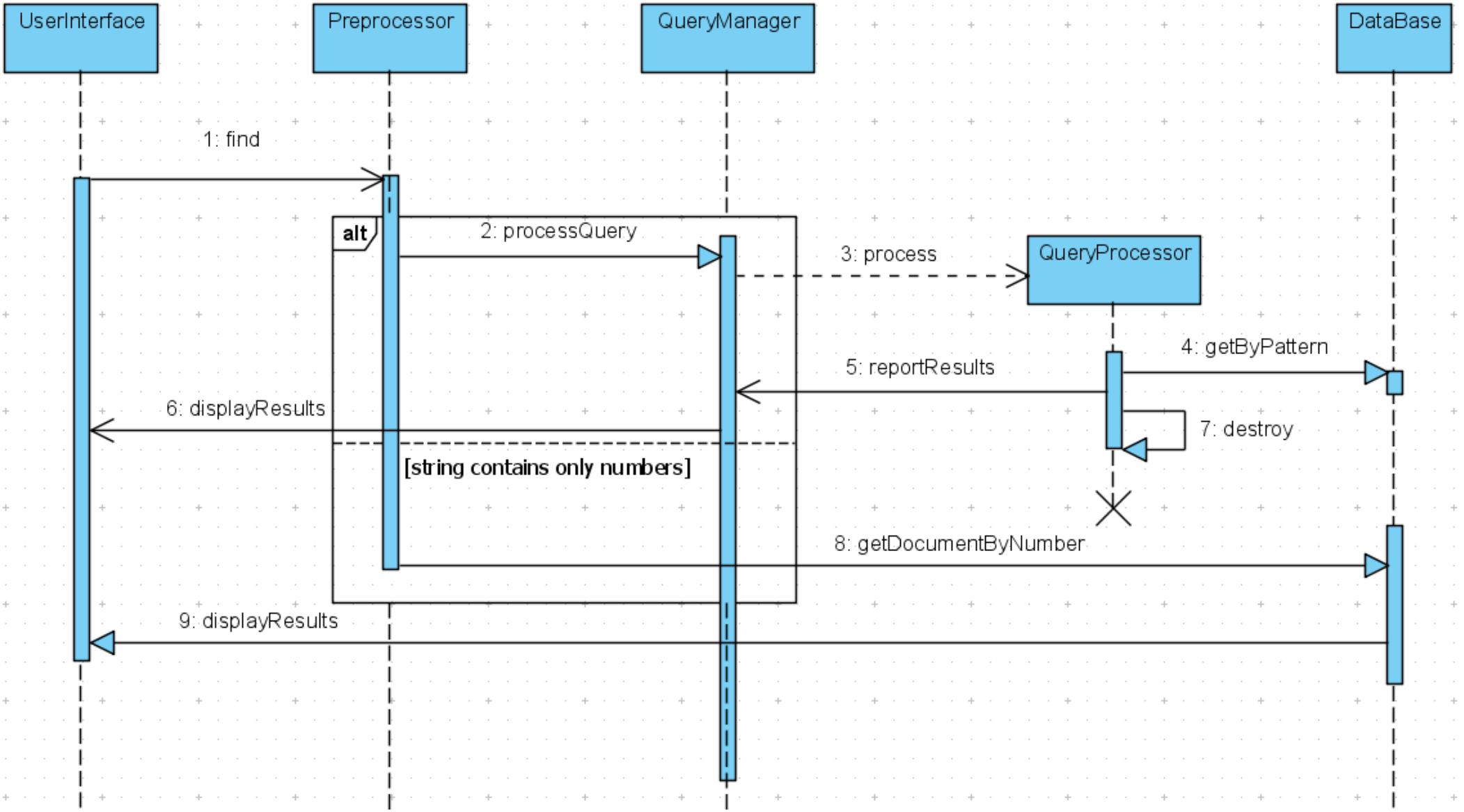
Wybrane operatory

- Alt
- Opt
- Loop
- Par

Operator alt - alternatywa

- Oznacza, że tylko jeden z podobszarów (operandów) obszaru objętego ramą może być wybrany
- Wybór ten zależy od od warunków umieszczonych w podfragmentach
- Podobszar bez warunku jest wyborem domyślnym

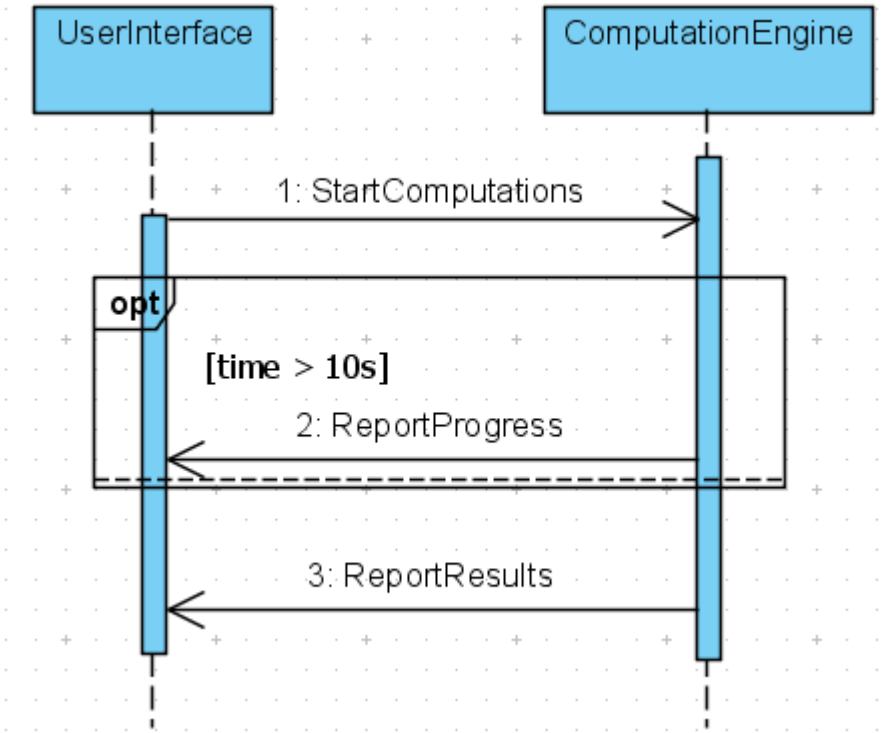
Przykład operatora alt



Operator opt – fragment opcjonalny

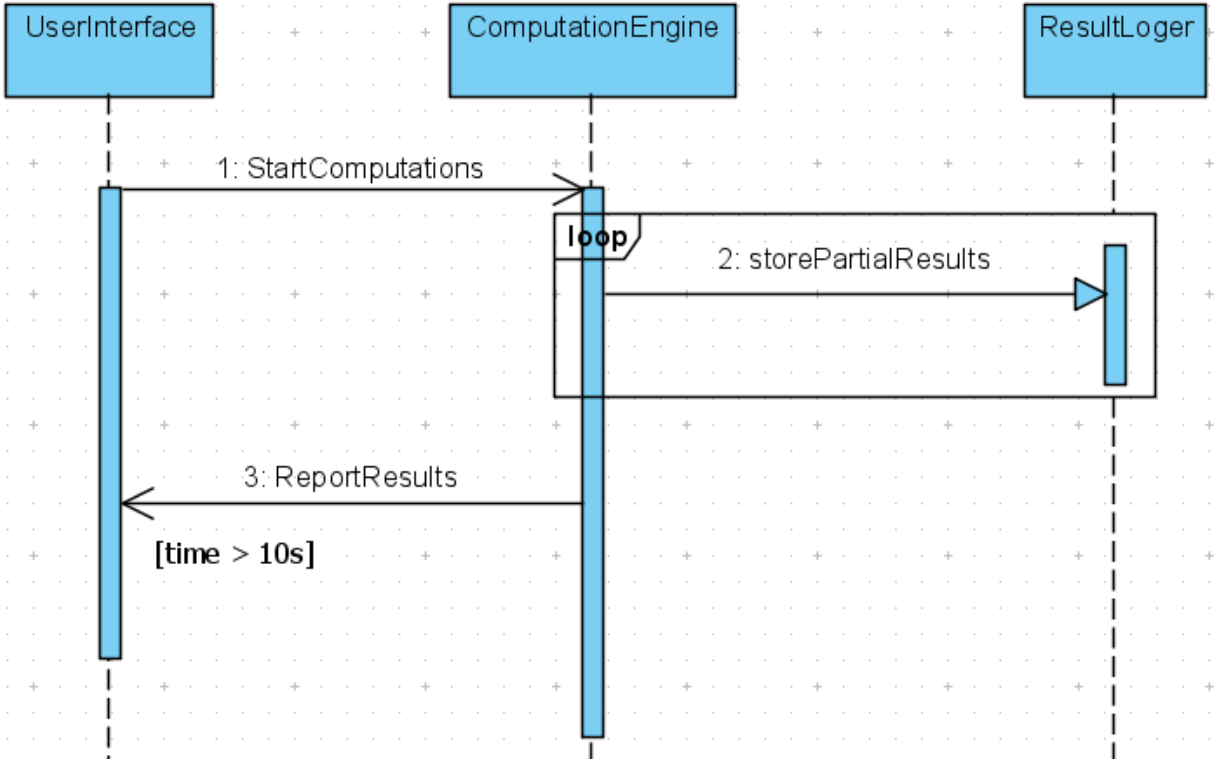
- Część diagramu zostanie wykonana tylko jeśli spełniony będzie warunek
- Odpowiada operatorowi alt z pustym domyslnym operandem

Przykład operatora opt



Operator loop - iteracja

- Umożliwia wielokrotne powtórzenie wybranego fragmentu
- Liczba interacji moze zostać określona



Operator par – wykonanie równoległe

- Oznacza, że podfragmenty fragmentu objetego ramą są wykonywane równolegle