Wydział Elektryczny
POLITECHNIKA WARSZAWSKA

1DI1107 - Podstawy informatyki

Mini Projekt 2

dr inż. Konrad Markowski
konrad.markowski@pw.edu.pl

Politechnika Warszawska

Definicja Grafu

Graf (ang. graph) jest strukturą danych składającą się z dwóch zbiorów:

- *zbioru wierzchołków V (ang. vertices),*
- *zbioru krawędzi E (ang. edges),*

co matematycznie zapisujemy w postaci uporządkowanej pary:

$$G = (V, E)$$

$V = \{v_1, v_2, ..., v_n\}$ – *zbiór n ponumerowanych wierzchołków,*

$E = \{e_1, e_2, ... e_m\}$ – *zbiór m ponumerowanych krawędzi.*

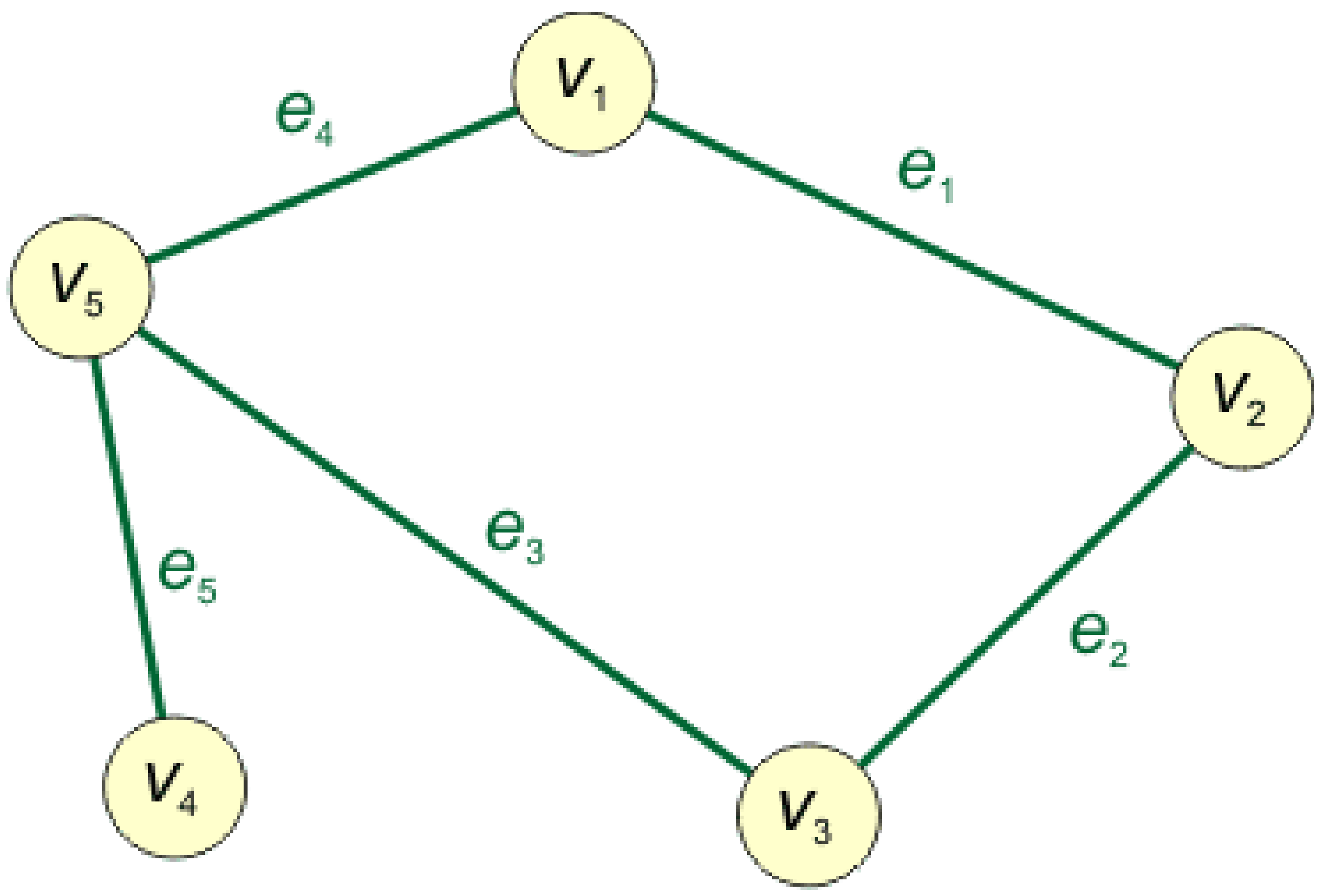
Każda krawędź jest parą (w grafie skierowanym parą uporządkowaną) wierzchołków grafu połączonych tą krawędzią:

$$E = \{ (u, v): u, v \in V \}$$

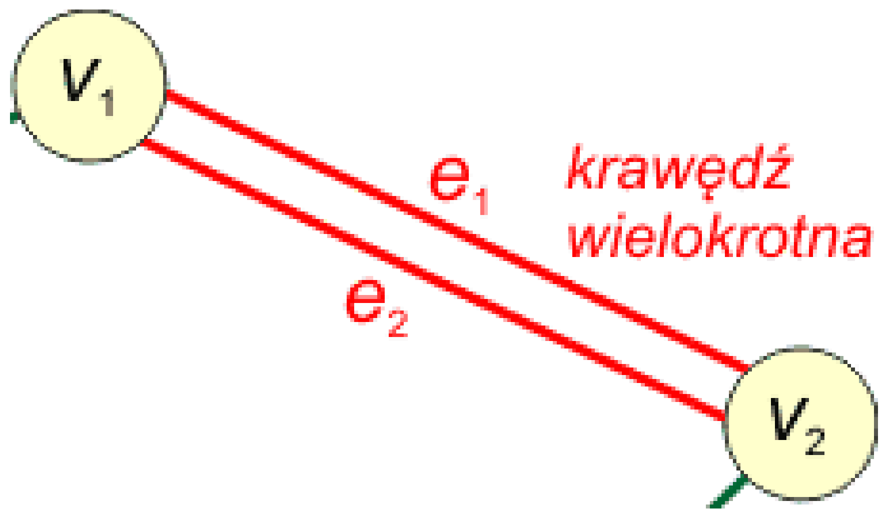
W grafie zwykłym mamy wierzchołki oraz krawędzie, które łączą ze sobą pary wierzchołków grafu:

$G = (V, E)$
$V = \{ v_1, v_2, v_3, v_4, v_5 \}$
$E = \{ e_1, e_2, e_3, e_4, e_5 \}$

$e_1 = (v_1, v_2)$
$e_2 = (v_2, v_3)$
$e_3 = (v_3, v_5)$
$e_4 = (v_1, v_5)$
$e_5 = (v_4, v_5)$



Dane dwa wierzchołki mogą być połączone ze sobą za pomocą więcej niż jednej krawędzi, które nazywamy ***krawędzią wielokrotną*** *(ang. multi-edge).*



$G = (V, E)$

$V = \{ v_1, v_2\}, E = \{ e_1, e_2\}$

$e_1 = (v_1, v_2);$ $e_2 = (v_1, v_2)$

e_1, e_2 – *krawędź wielokrotna*

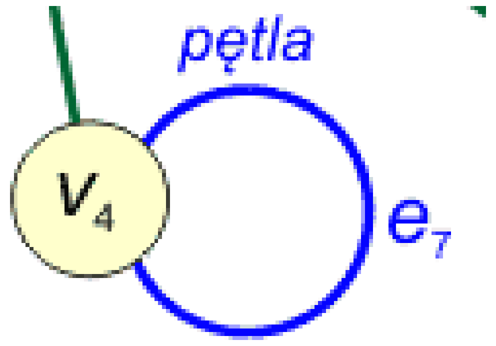
Wierzchołek może łączyć się krawędzią z samym sobą. Otrzymujemy wtedy tzw. ***pętlę*** *(ang. loop).*

$G = (V, E)$

$V = \{ v_4 \};$ $E = \{ e_7 \}$

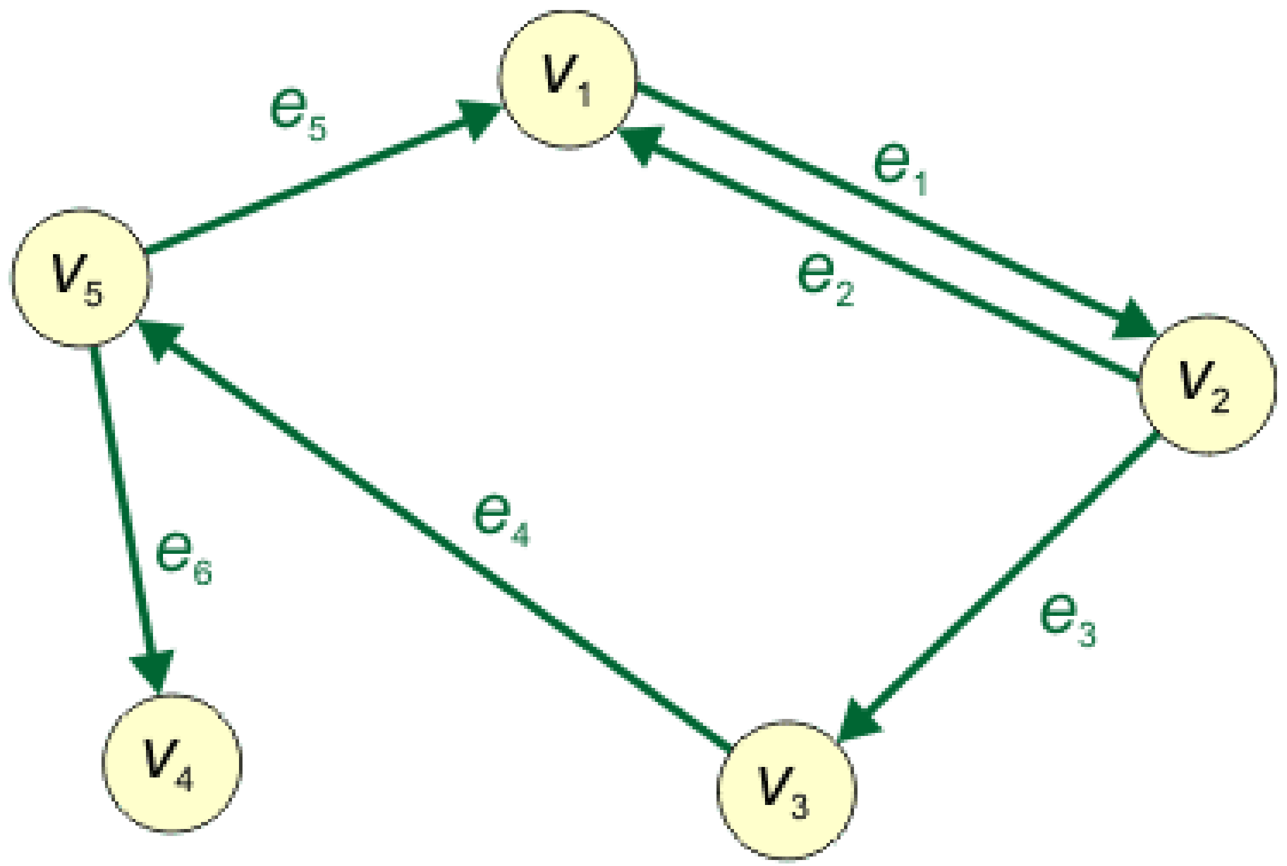
$e_7 = (v_4, v_4)$

e_7 – *pętla*



Graf zawierający pętle lub krawędzie wielokrotne nazywamy ***multigrafem****.*

- *Krawędź, którą można przebywać tylko w określoną stronę, nazywa się **krawędzią skierowaną** (ang. directed edge). Krawędzie skierowane oznaczamy strzałkami.*
- *Graf zawierający krawędzie skierowane nazywamy grafem skierowanym (ang. directed graph) lub w skrócie digrafem (ang. digraph).*



$G = (V, E)$

$V = \{ v_1, v_2, v_3, v_4, v_5 \}$

$E = \{ e_1, e_2, e_3, e_4, e_5, e_6 \}$

$e_1 = (v_1, v_2)$

$e_2 = (v_2, v_1)$

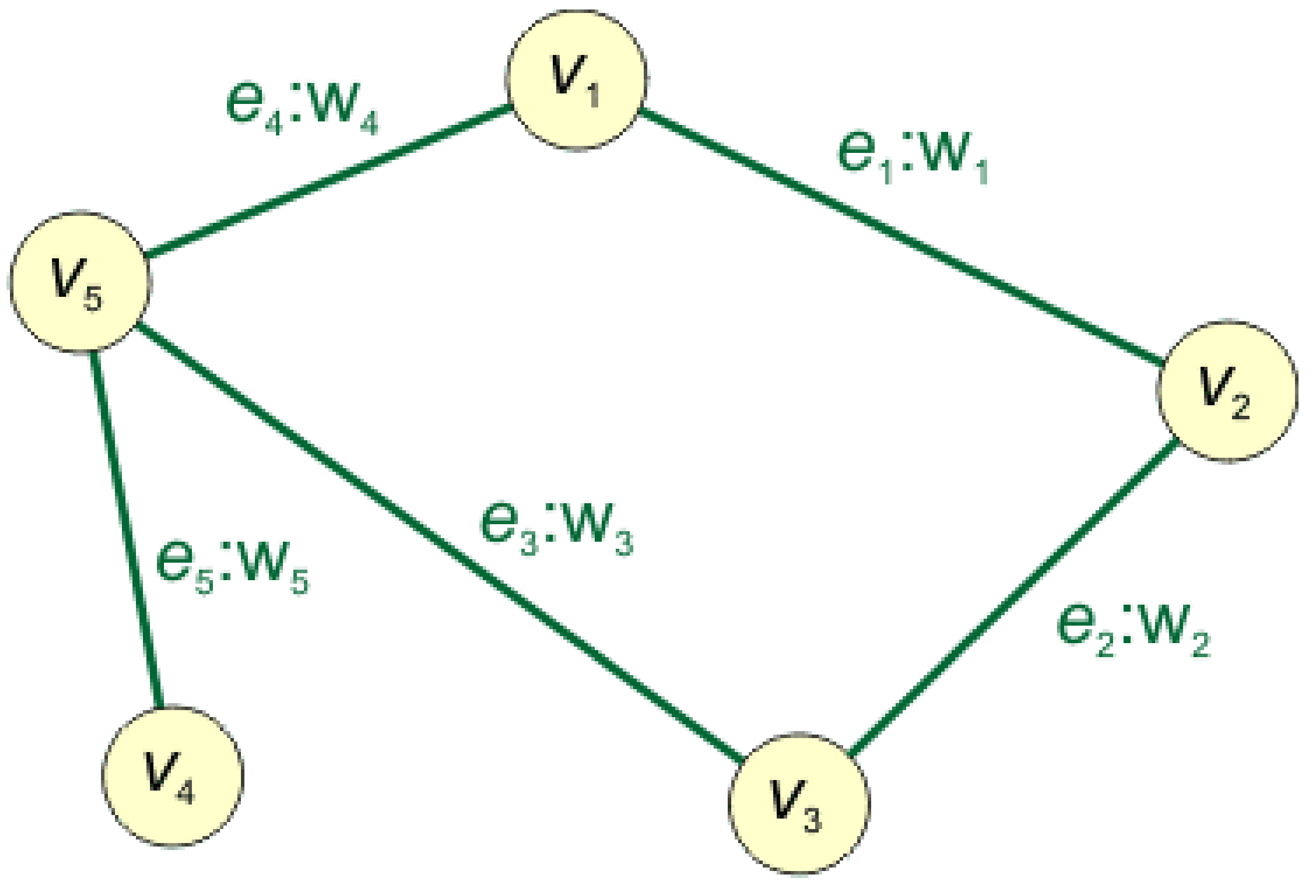
$e_3 = (v_2, v_3)$

$e_4 = (v_3, v_5)$

$e_5 = (v_5, v_1)$

$e_6 = (v_5, v_4)$

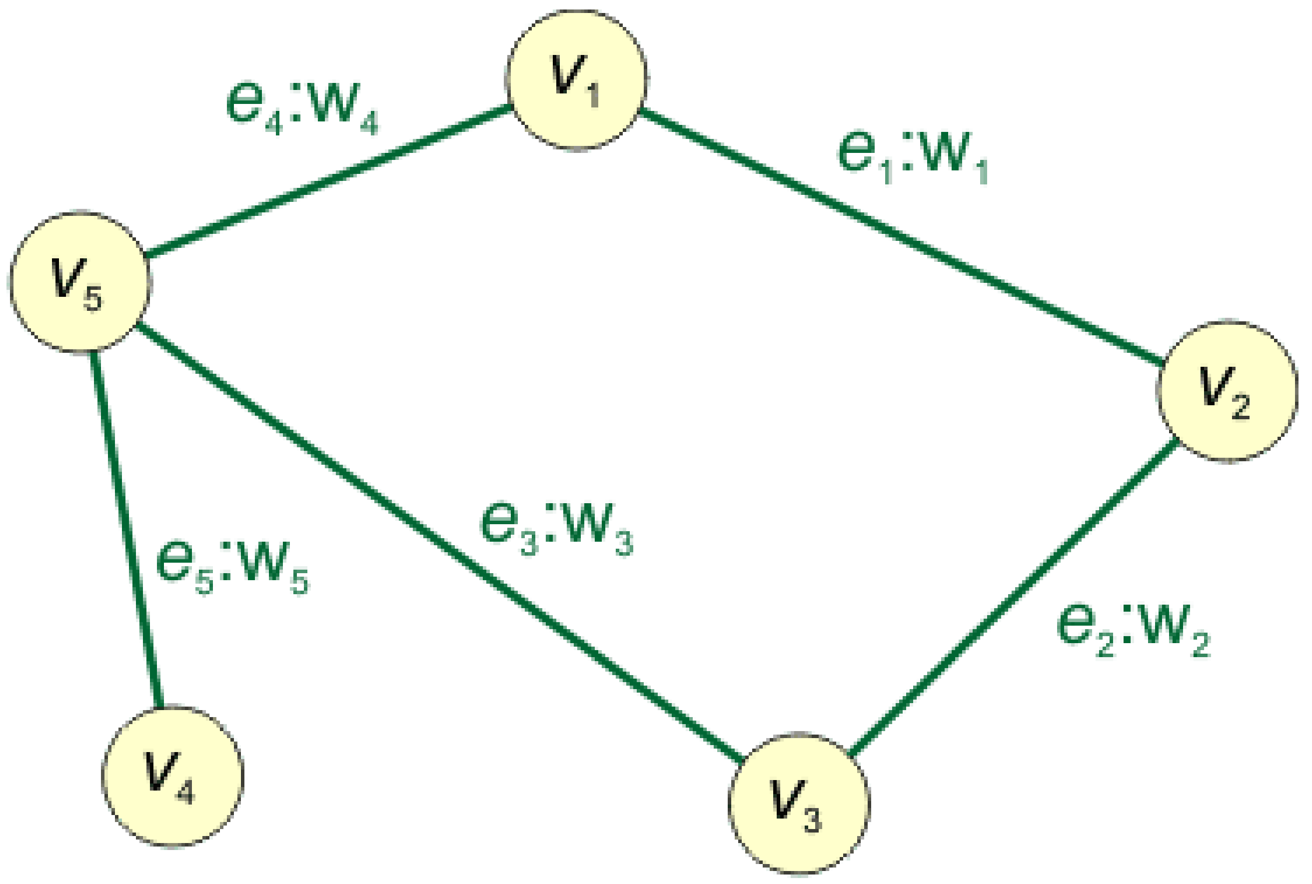
- *Z krawędziami grafu mogą być związane dodatkowe wartości, np. pokonanie drogi z jednego miasta do drugiego może wymagać określonej ilości czasu lub energii.*
- *Wartości te nazywamy wagami (ang. weight), a graf posiadający takie krawędzie nazywamy **grafem ważonym** (ang. weighted graph).*



$G = (V, E)$
$V = \{ v_1, v_2, v_3, v_4, v_5 \}$
$E = \{ e_1, e_2, e_3, e_4, e_5 \}$

$e_1 = (v_1, v_2, w_1)$
$e_2 = (v_2, v_3, w_2)$
$e_3 = (v_3, v_5, w_3)$
$e_4 = (v_1, v_5, w_4)$
$e_5 = (v_4, v_5, w_5)$

- *Z krawędziami grafu mogą być związane dodatkowe wartości, np. pokonanie drogi z jednego miasta do drugiego może wymagać określonej ilości czasu lub energii.*
- *Wartości te nazywamy wagami (ang. weight), a graf posiadający takie krawędzie nazywamy **grafem ważonym** (ang. weighted graph).*



G = (V, E)

V = { v_1, v_2, v_3, v_4, v_5 }

E = { e_1, e_2, e_3, e_4, e_5 }

$e_1 = (v_1, v_2, w_1)$

$e_2 = (v_2, v_3, w_2)$

$e_3 = (v_3, v_5, w_3)$

$e_4 = (v_1, v_5, w_4)$

$e_5 = (v_4, v_5, w_5)$

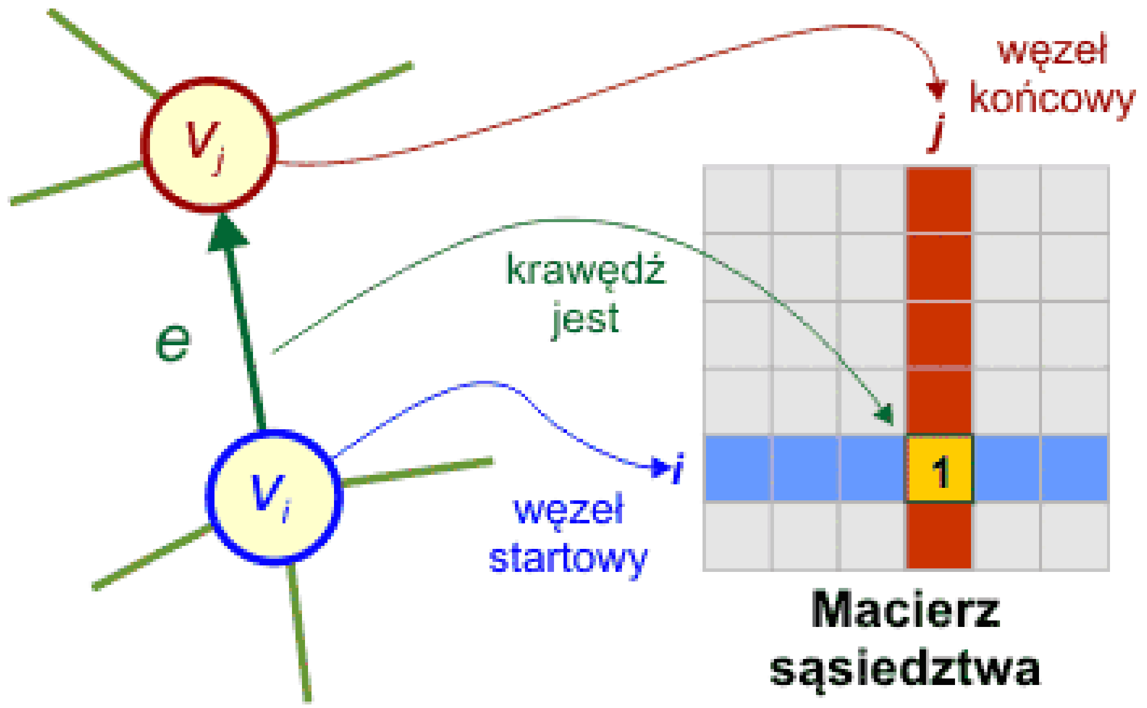
Sposoby reprezentacji grafów

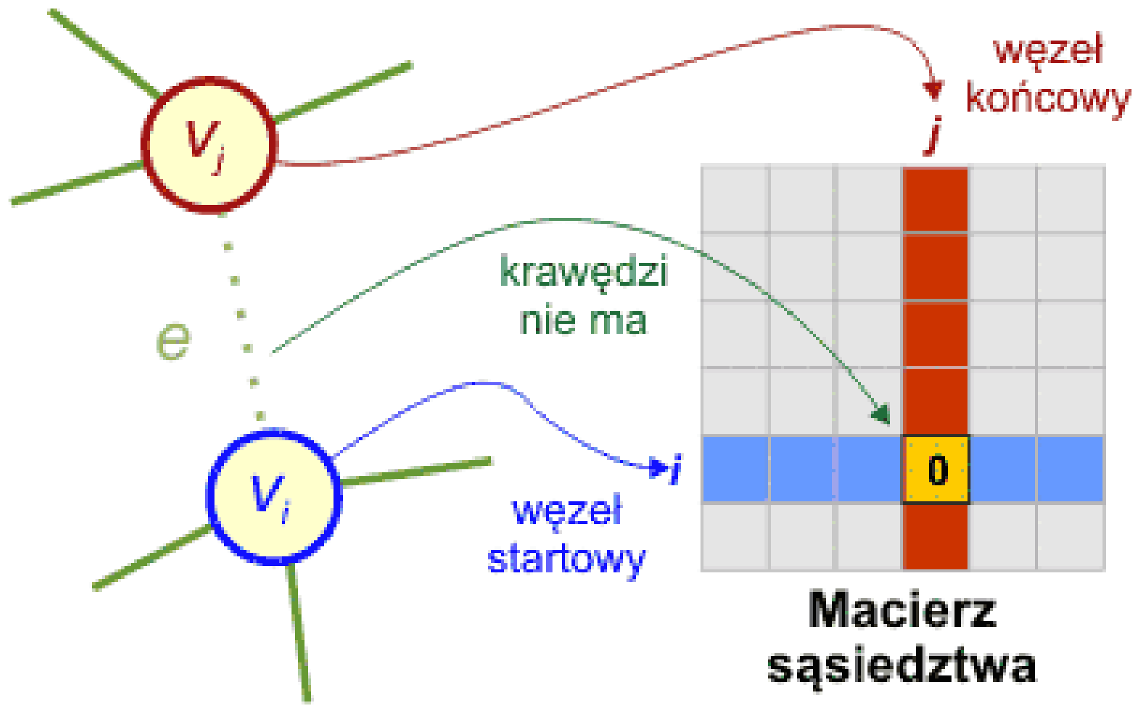
Chcąc realizować algorytmy grafowe, będziemy zmuszeni wprowadzać różne grafy do pamięci komputera:

- *Macierz sąsiedztwa*
- *Macierz incydencji*
- *Lista sąsiedztwa*

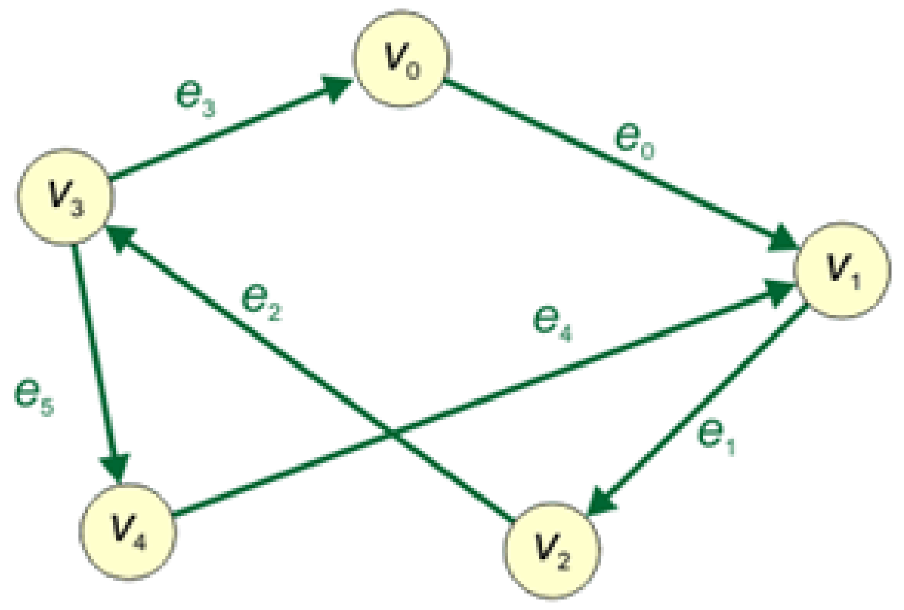
Macierz sąsiedztwa

- *Graf reprezentujemy za pomocą macierzy kwadratowej* **A** *o stopniu n, gdzie n oznacza liczbę wierzchołków w grafie.*
- *Macierz tą nazywamy macierzą sąsiedztwa (ang. adjacency matrix). Odwzorowuje ona połączenia wierzchołków krawędziami.*
- *Komórka* **A**[*i, j*]*, która znajduje się w i-tym wierszu i j-tej kolumnie odwzorowuje krawędź łączącą wierzchołek startowy* v_i *z wierzchołkiem końcowym* v_j*.*





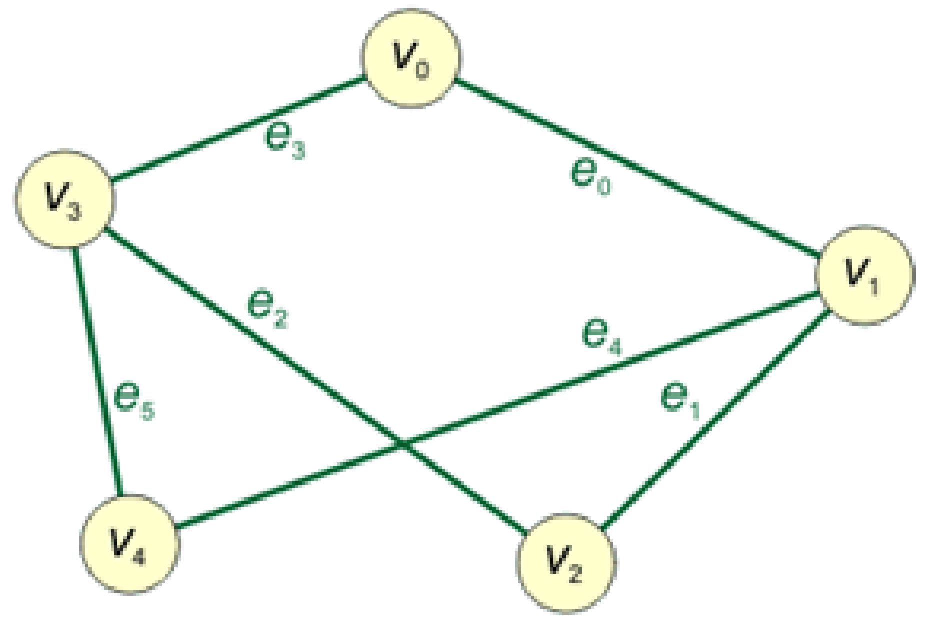
- **<u>Przykład 1:</u>** *Graf skierowany*



	0	1	2	3	4
0	0	1	0	0	0
1	0	0	1	0	0
2	0	0	0	1	0
3	1	0	0	0	1
4	0	1	0	0	0

- *Jeśli* $\mathbf{A}[\,i, j\,]$ *ma wartość 1, to dana krawędź istnieje.*
- *Jeśli* $\mathbf{A}[\,i, j\,]$ *ma wartość 0, to wierzchołek* v_i *nie łączy się krawędzią z wierzchołkiem* v_j.

- **<u>Przykład 2:</u>** *Graf nieskierowany*



	0	1	2	3	4
0	0	1	0	1	0
1	1	0	1	0	1
2	0	1	0	1	0
3	1	0	1	0	1
4	0	1	0	1	0

Dla grafu nieskierowanego macierz sąsiedztwa $\mathbf{A}$ *jest symetryczna względem głównej przekątnej, ponieważ jeśli istnieje krawędź od* v_i *do* v_j $(\mathbf{A}[\,i, j\,] = 1)$, *to również musi istnieć krawędź w kierunku odwrotnym, od* v_j *do* v_i $(\mathbf{A}[\,j, i\,] = 1)$.

Macierz incydencji

- *Macierz incydencji (ang. incidence matrix) jest macierzą* **A** *o wymiarze* $n \times m$, *gdzie* n *oznacza liczbę wierzchołków grafu, a* m *liczbę jego krawędzi.*

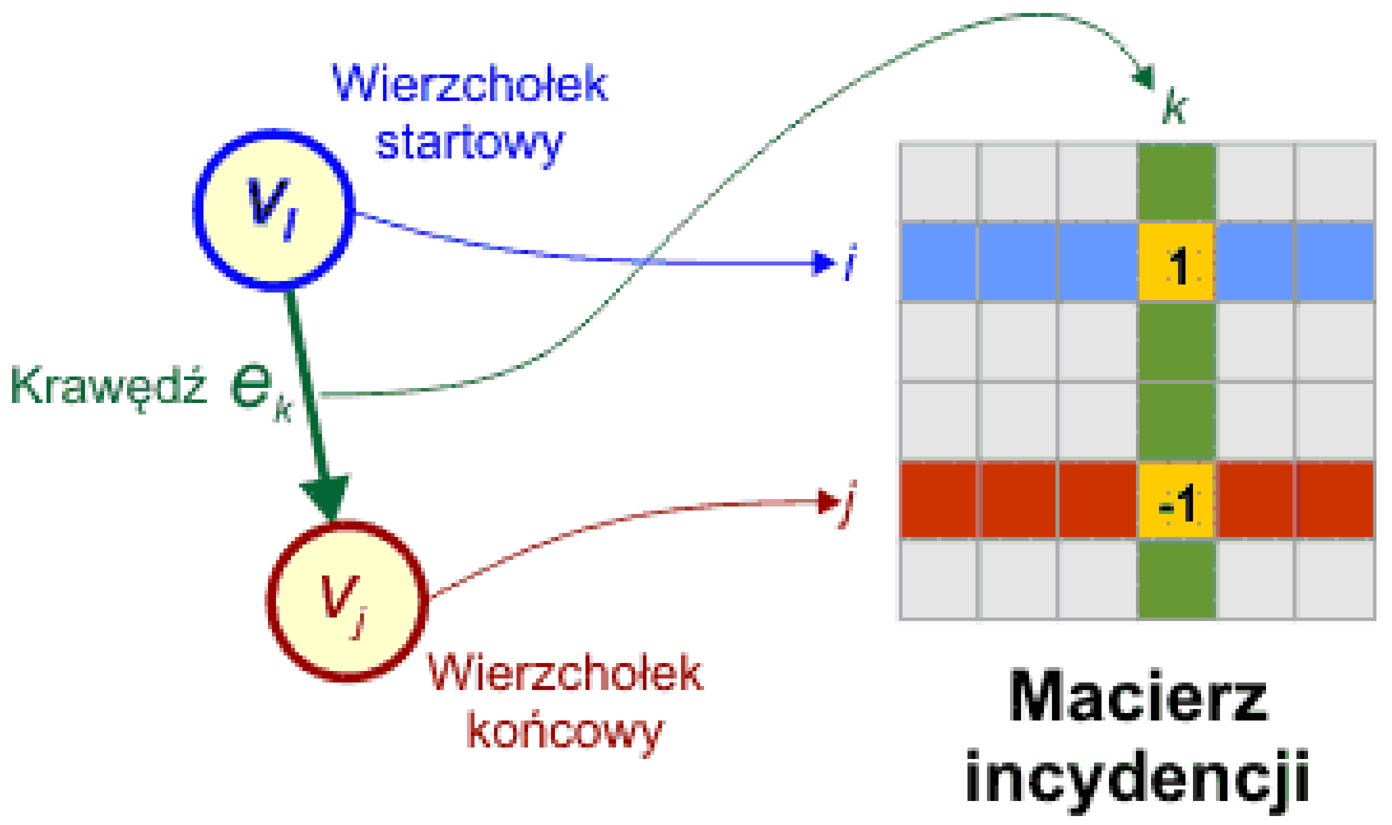
- *Każdy wiersz tej macierzy odwzorowuje jeden wierzchołek grafu.*

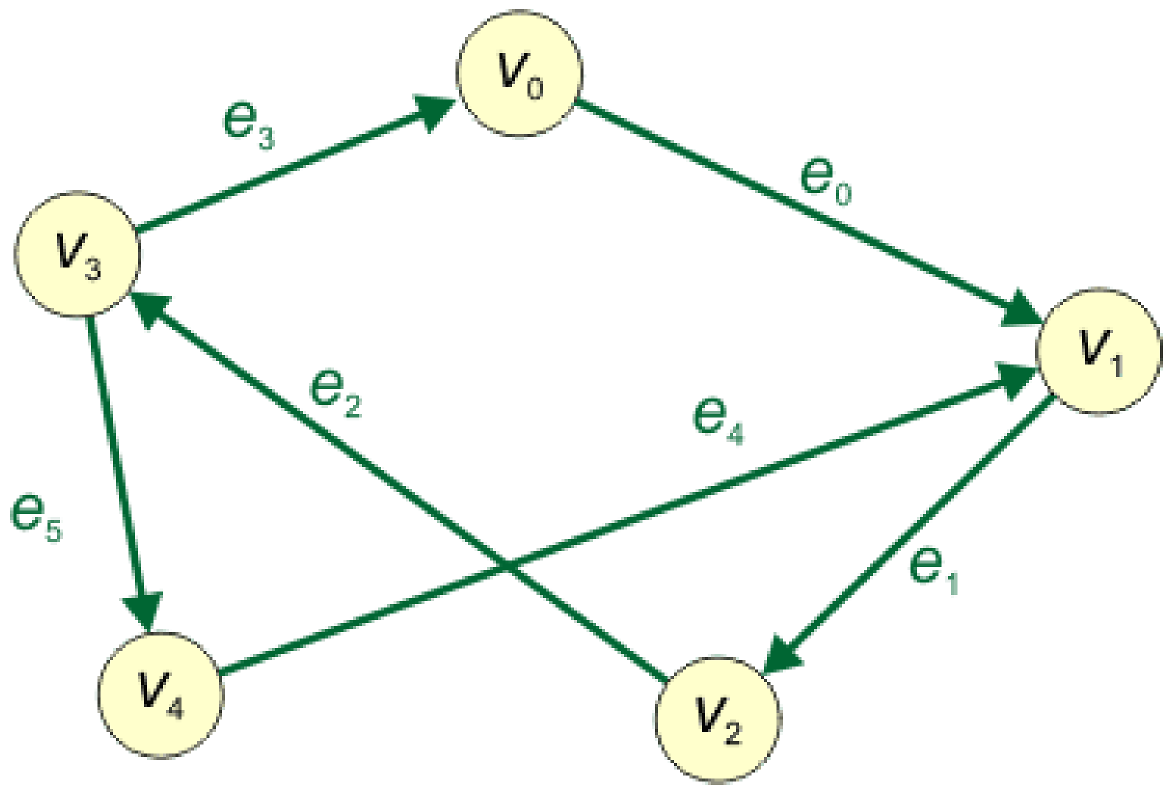
- *Każda kolumna odwzorowuje jedną krawędź.*

Graf skierowany:

Zawartość komórki **A** [*i, j*] *określa powiązanie (incydencję) wierzchołka* v_i *z krawędzią* e_j *w sposób następujący:*

$$A[i,j] = \begin{cases} 0, \text{ jeśli } v_i \text{ nie należy do } e_j \\ 1, \text{ jeśli } v_i \text{ jest początkiem } e_j \\ -1, \text{ jeśli } v_i \text{ jest końcem } e_j \end{cases}$$



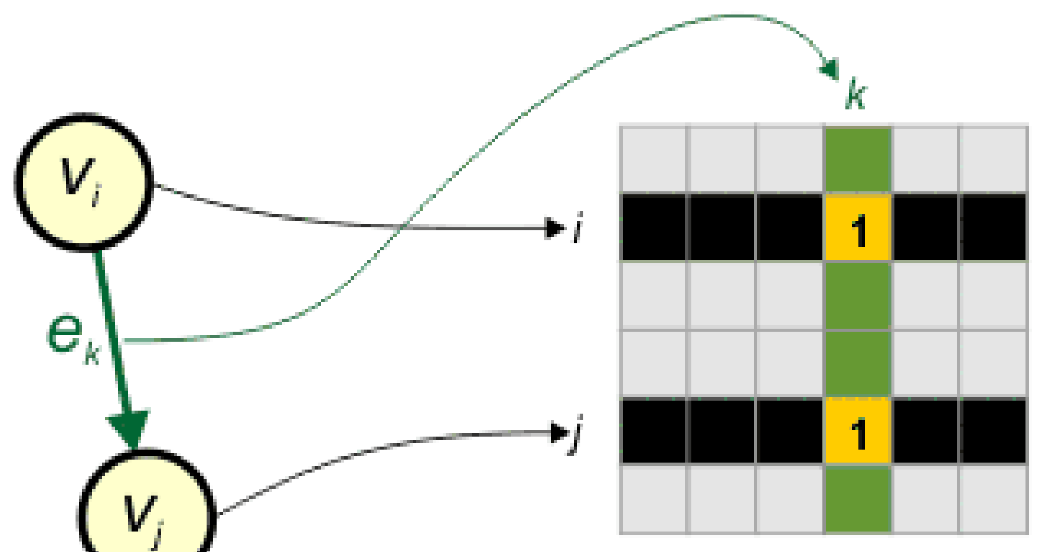


	0	1	2	3	4	5
0	1	0	0	-1	0	0
1	-1	1	0	0	-1	0
2	0	-1	1	0	0	0
3	0	0	-1	1	0	1
4	0	0	0	0	1	-1

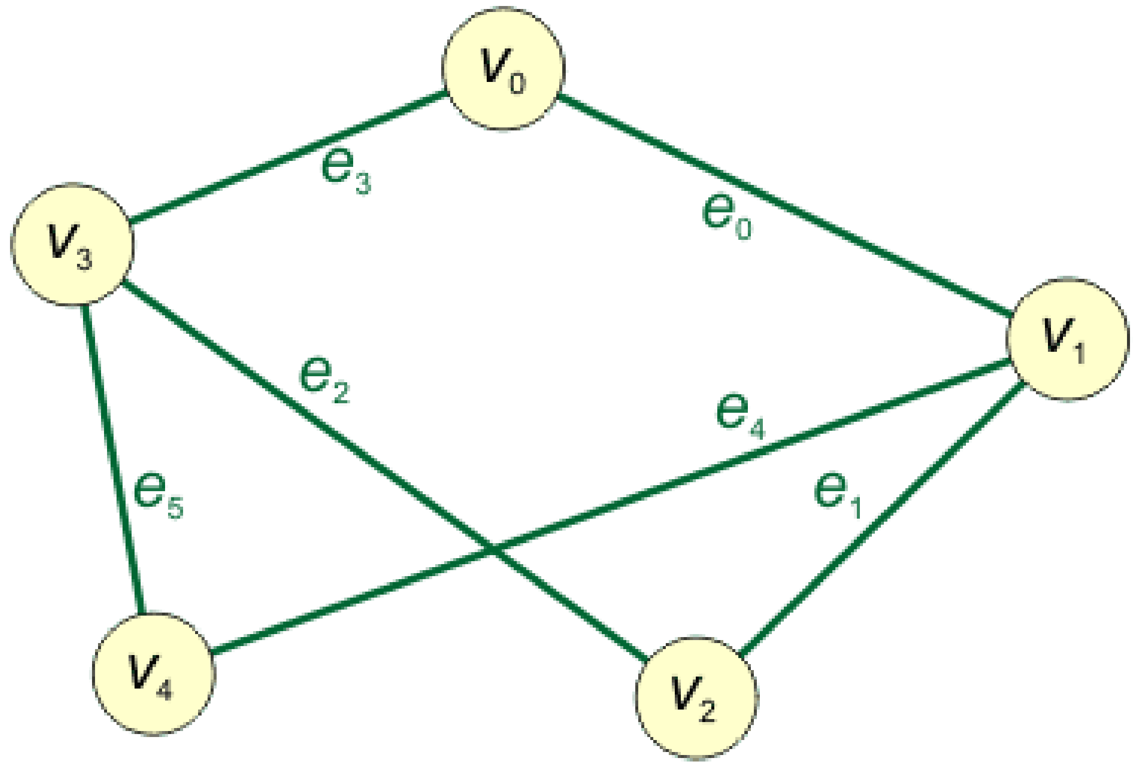
Graf nieskierowany:

Zawartość komórki **A** [*i, j*] *określa powiązanie (incydencję) wierzchołka* v_i *z krawędzią* e_j *w sposób następujący:*

$$A[i,j]=\begin{cases}0, \text{ jeśli } v_i \text{ nie należy do } e_j\\ 1, \text{ jeśli } v_i \text{ należy do } e_j\end{cases}$$



Macierz incydencji

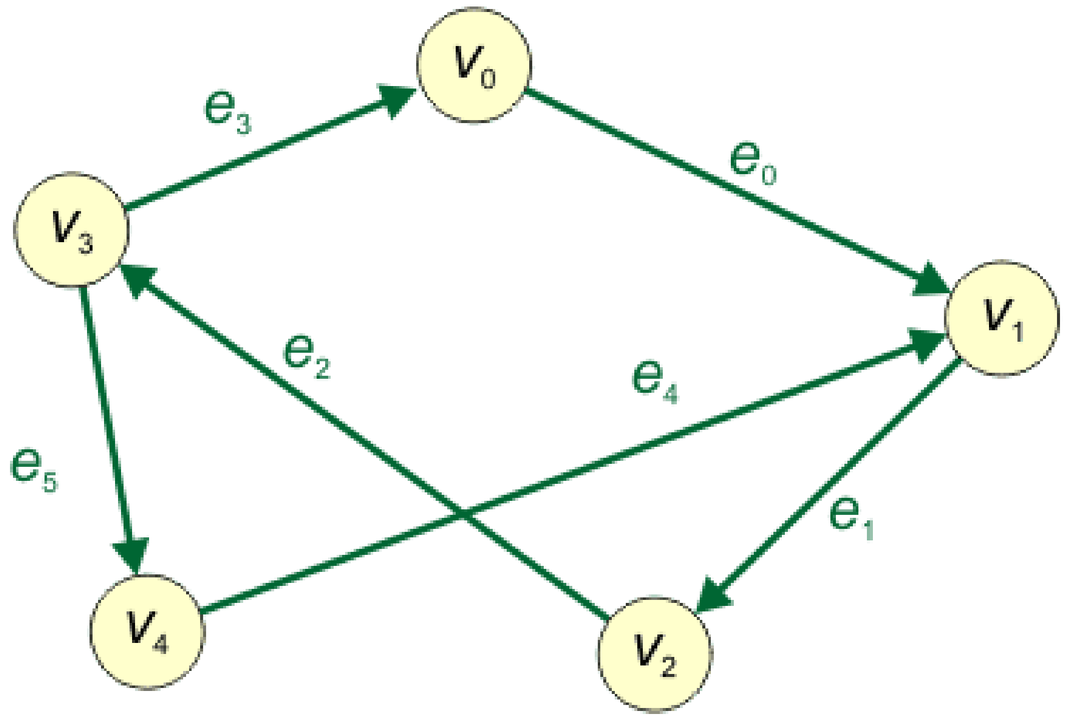


	0	1	2	3	4	5
0	1	0	0	1	0	0
1	1	1	0	0	1	0
2	0	1	1	0	0	0
3	0	0	1	1	0	1
4	0	0	0	0	1	1

Lista sąsiedztwa

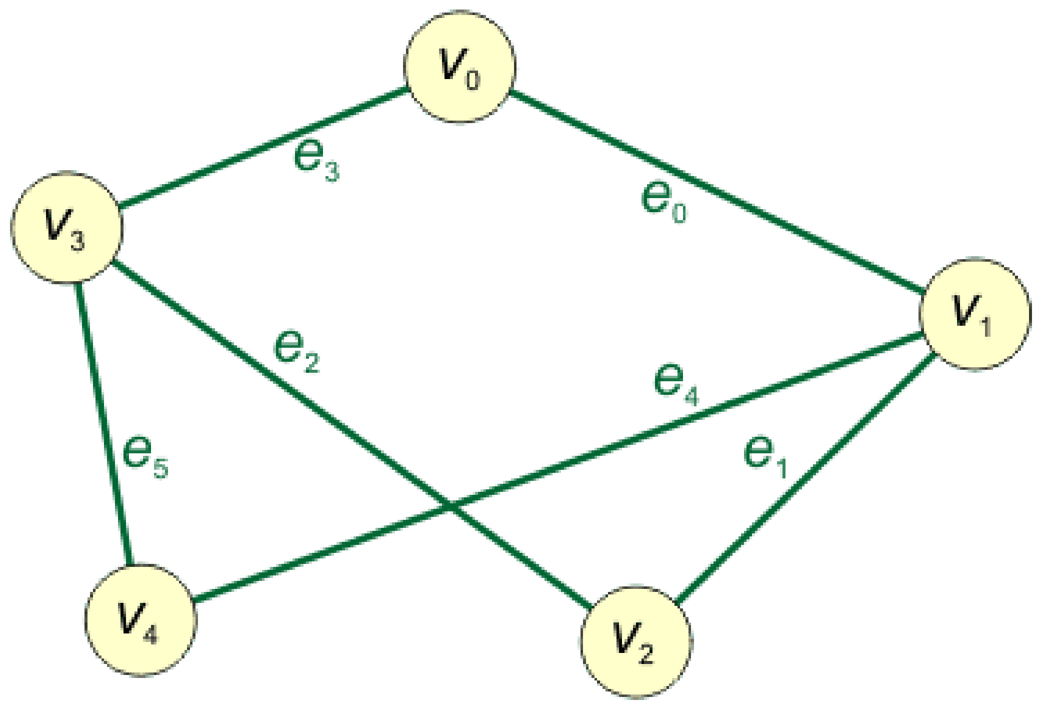
- *Do reprezentacji grafu wykorzystujemy tablicę n elementową* **A**, *gdzie n oznacza liczbę wierzchołków.*
- *Każdy element tej tablicy jest listą.*
- *Lista reprezentuje wierzchołek startowy.*
- *Na liście są przechowywane numery wierzchołków końcowych, czyli sąsiadów wierzchołka startowego, z którymi jest on połączony krawędzią.*
- *Tablica ta nosi nazwę list sąsiedztwa* (*ang. adjacency lists*).

Graf skierowany:



0	1	
1	2	
2	3	
3	0	4
4	1	

Graf nieskierowany:



0	1	3	
1	0	4	2
2	1	3	
3	0	2	4
4	1	3	