Edge list

All vertex objects are stored in an unordered list V, and all edge objects are stored in an unordered list E.

$$V = \{v_1,\ v_2,\ v_3,\ v_4,\ v_5,\ v_6\}$$
$$E = \{e_1,\ e_2,\ e_3,\ e_4,\ e_5,\ e_6\}$$

Each v_i is an Vertex object that has element x(value) and also position of the vertex in V.

v_1 → (x = v_1.value, pos = 0)
v_2 → (x = v_2.value, pos = 1)
v_3 → (x = v_3.value, pos = 2)
v_4 → (x = v_4.value, pos = 3)
v_5 → (x = v_5.value, pos = 4)
v_6 → (x = v_6.value, pos = 5)

Each e_i is an Edge objest that has element x(value) and references to the vertex objects associated with the endpoint vertices of e_i. Also has A reference to the position of the edge instance in list E.

e_1 → (x = e_1.value, associated = (v_1, v_2), pos = 0)
e_2 → (x = e_2.value, associated = (v_2, v_4), pos = 1)
e_3 → (x = e_3.value, associated = (v_1, v_3), pos = 2)
e_4 → (x = e_4.value, associated = (v_3, v_4), pos = 3)
e_5 → (x = e_5.value, associated = (v_3, v_6), pos = 4)
e_6 → (x = e_6.value, associated = (v_3, v_5), pos = 0)

Adjacency matrix

n = 6
-- = nullptr

v_1 → [--, e_1, e_3, --, --, --],
v_2 → [e_1, --, --, e_2, --, --],
v_3 → [e_3, --, --, e_4, e_6, e_5],
v_4 → [--, e_2, e_4, --, --, --],
v_5 → [--, --, e_6, --, --, --],
v_6 → [--, --, e_5, --, --, --],

Adjacency list

For each vertex v, we maintain a collection I(v), called the incidence collection of v, whose entries are edges incident to v.

$$V = \{v_1, v_2, v_3, v_4, v_5, v_6\}$$

$v_1 \rightarrow (x = v_1.value, I(v) = \{e_1, e_3\}, pos = 0)$
$v_2 \rightarrow (x = v_2.value, I(v) = \{e_1, e_2\}, pos = 1)$
$v_3 \rightarrow (x = v_3.value, I(v) = \{e_3, e_4, e_5, e_6\}, pos = 2)$
$v_4 \rightarrow (x = v_4.value, I(v) = \{e_2 , e_4\}, pos = 3)$
$v_5 \rightarrow (x = v_5.value, I(v) = \{e_6\}, pos = 4)$
$v_6 \rightarrow (x = v_6.value, I(v) = \{e_5\}, pos = 5)$

Adjacency map

hash based map to implement I(v) from above. we let the opposite endpoint of each incident edge serve as a key in the map, with the edge structure serving as the value.

$v_2{:}e_1$ → means that v_2 is key and hash of it will return e_1 as the value.

$v_1 \rightarrow (x = v_1.value, I(v)\_map = \{v_2{:}e_1, v_3{:}e_3\}, pos = 0)$
$v_2 \rightarrow (x = v_2.value, I(v)\_map = \{v_1{:}e_1, v_4{:}e_2\}, pos = 1)$
$v_3 \rightarrow (x = v_3.value, I(v)\_map = \{v_1{:}e_3, v_4{:}e_4, v_5{:}e_6, v_6{:}e_5\}, pos = 2)$
$v_4 \rightarrow (x = v_4.value, I(v)\_map = \{ v_2{:}e_2, v_3{:}e_4\}, pos = 3)$
$v_5 \rightarrow (x = v_5.value, I(v)\_map = \{v_3{:}e_6\}, pos = 4)$
$v_6 \rightarrow (x = v_6.value, I(v)\_map = \{v_3{:}e_5\}, pos = 5)$