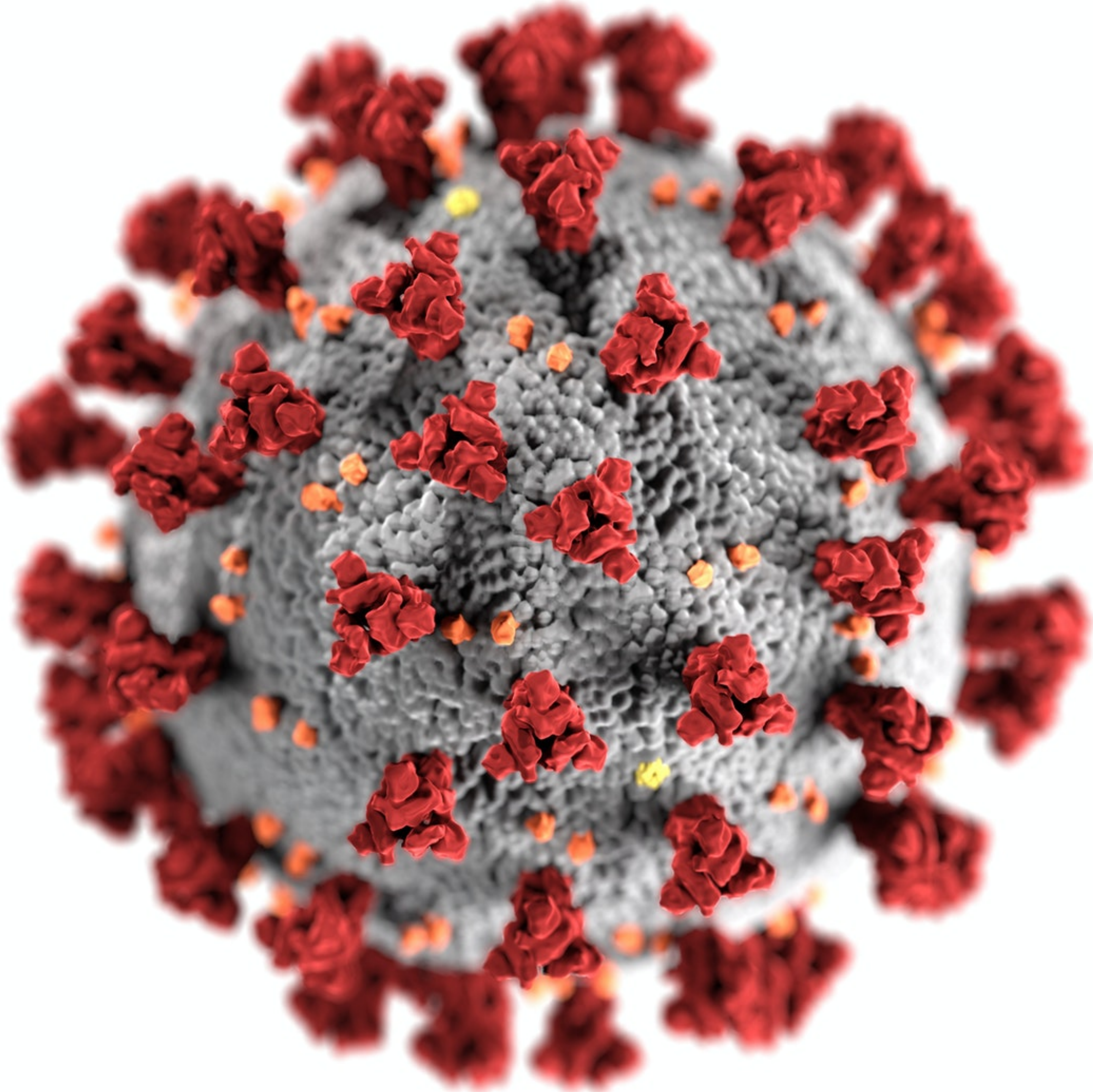
accu 2022

THROWING GRAPHS AT COVID

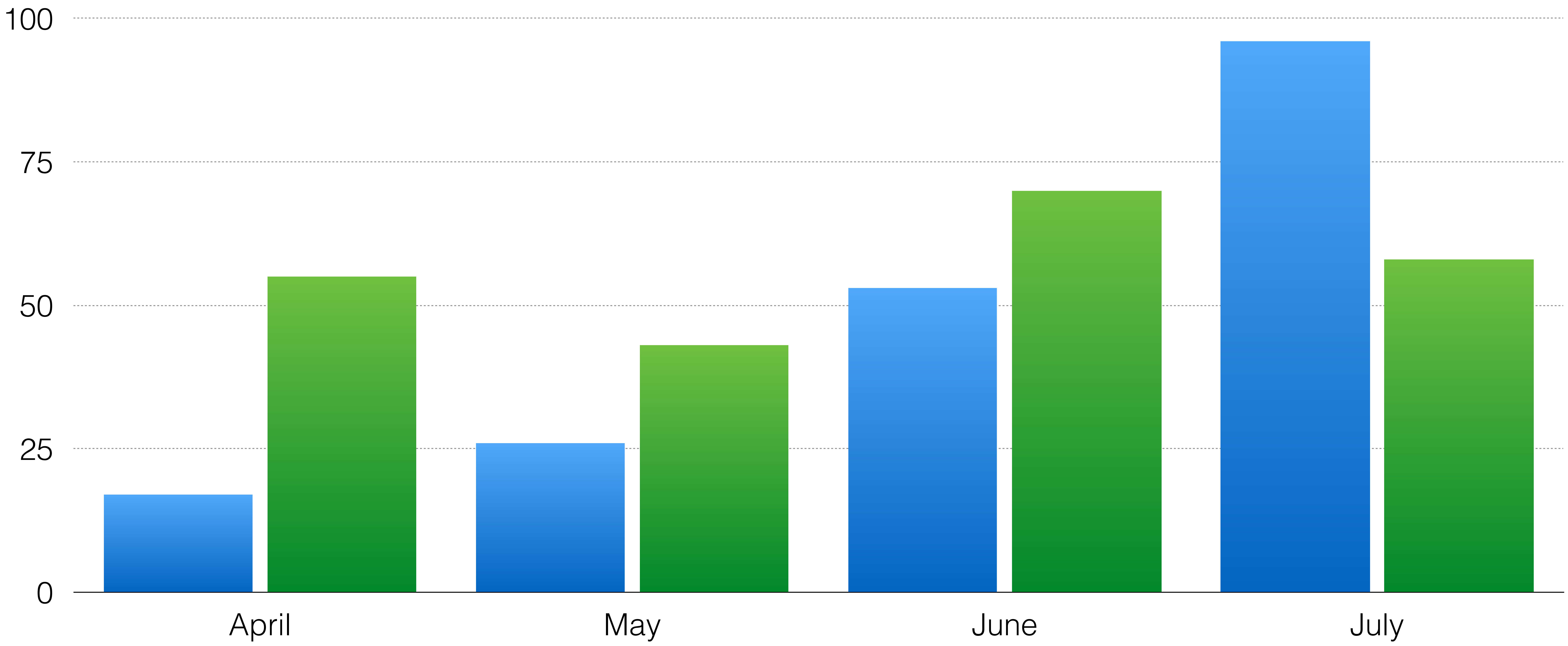
DOM DAVIS

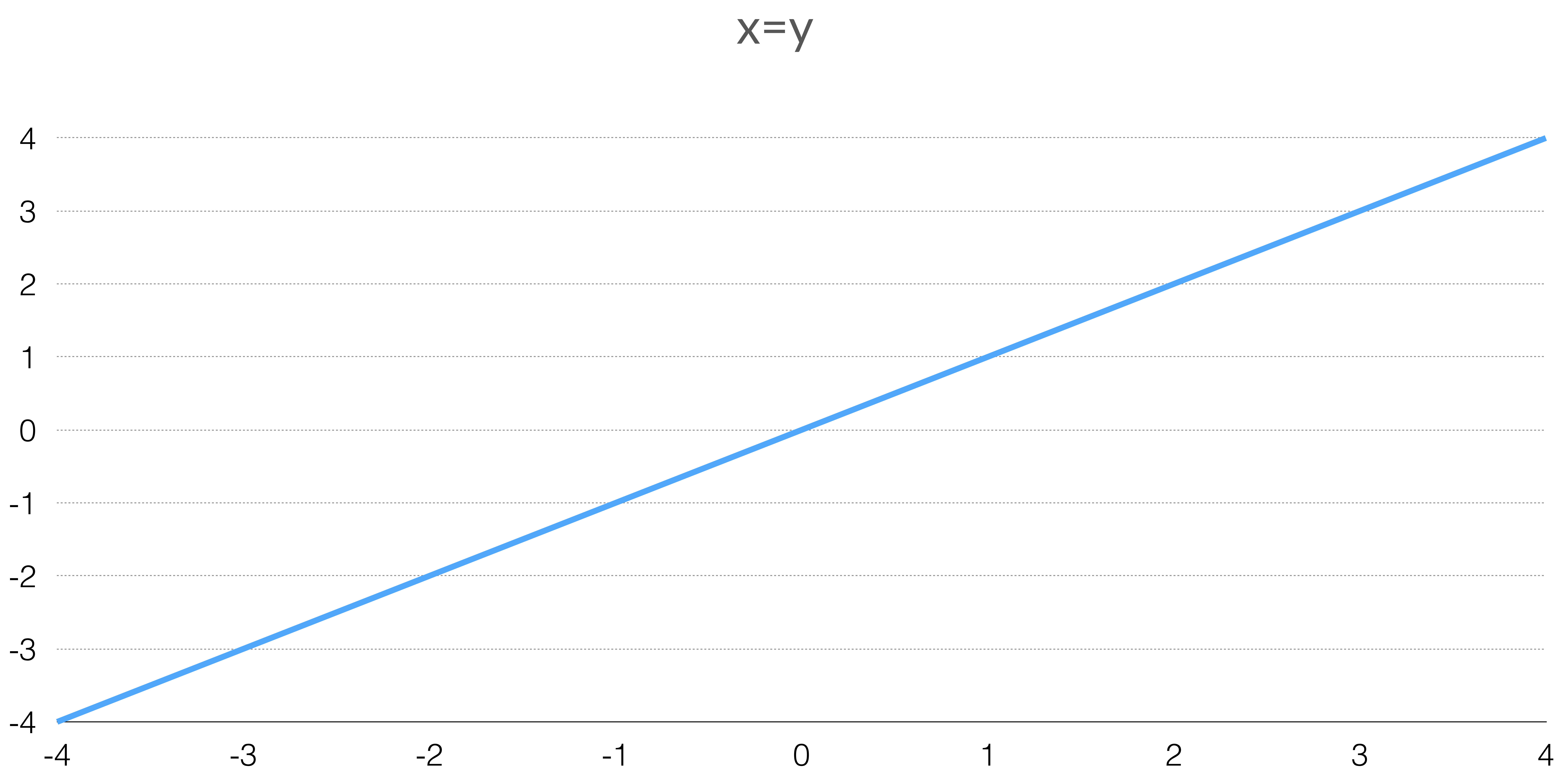


Dom Davis

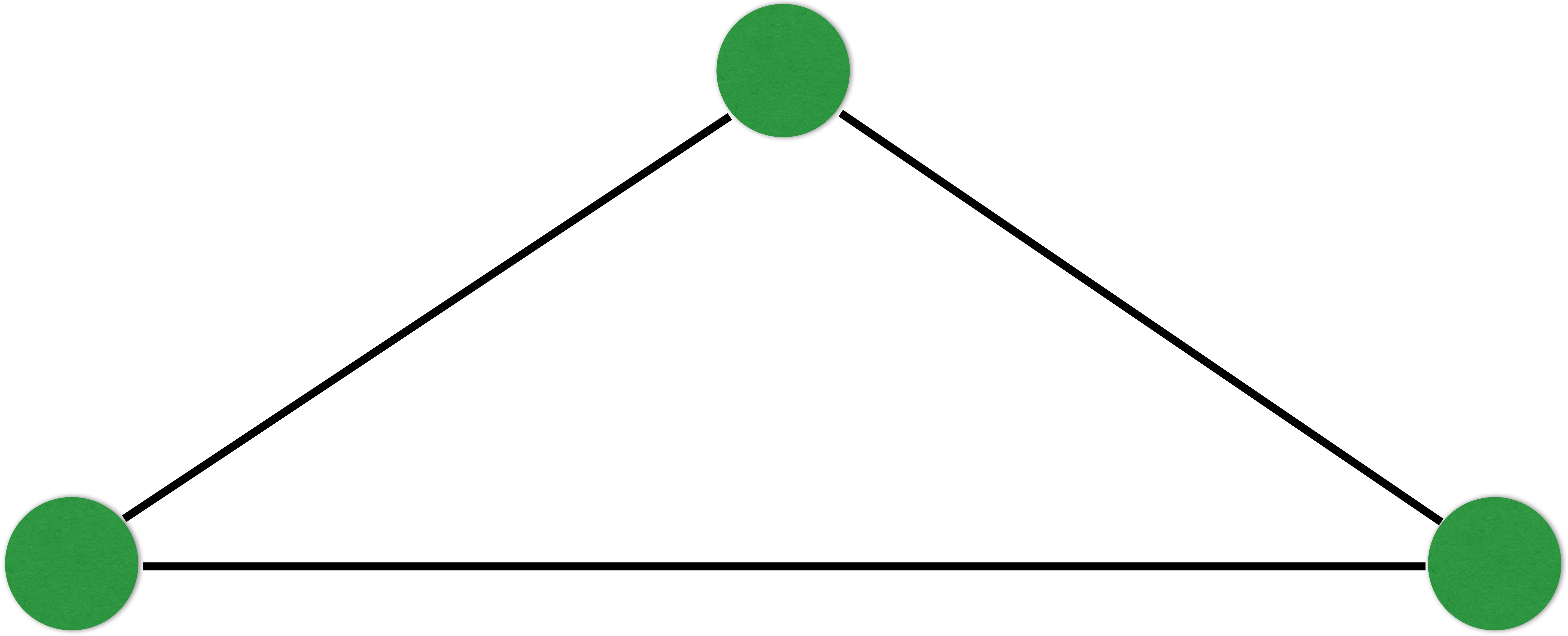
@idomdavis

about.me/idomdavis


100
75
50
25
0
April
May
June
July



x=y
4
3
2
1
0
-1
-2
-3
-4
-4
-3
-2
-1
0
1
2
3
4



USAGE

In its standard use **literally** means ‘in a literal sense, as opposed to a non-literal or exaggerated sense’, as for example in *I told him I never wanted to see him again, but I didn't expect him to take it **literally***. In recent years an extended use of **literally** (and also **literal**) has become very common, where **literally** (or **literal**) is used deliberately in non-literal contexts, for added effect, as in *they bought the car and **literally** ran it into the ground*. This use can lead to unintentional humorous effects (*we were **literally** killing ourselves laughing*) and is not acceptable in formal contexts, though it is widespread.

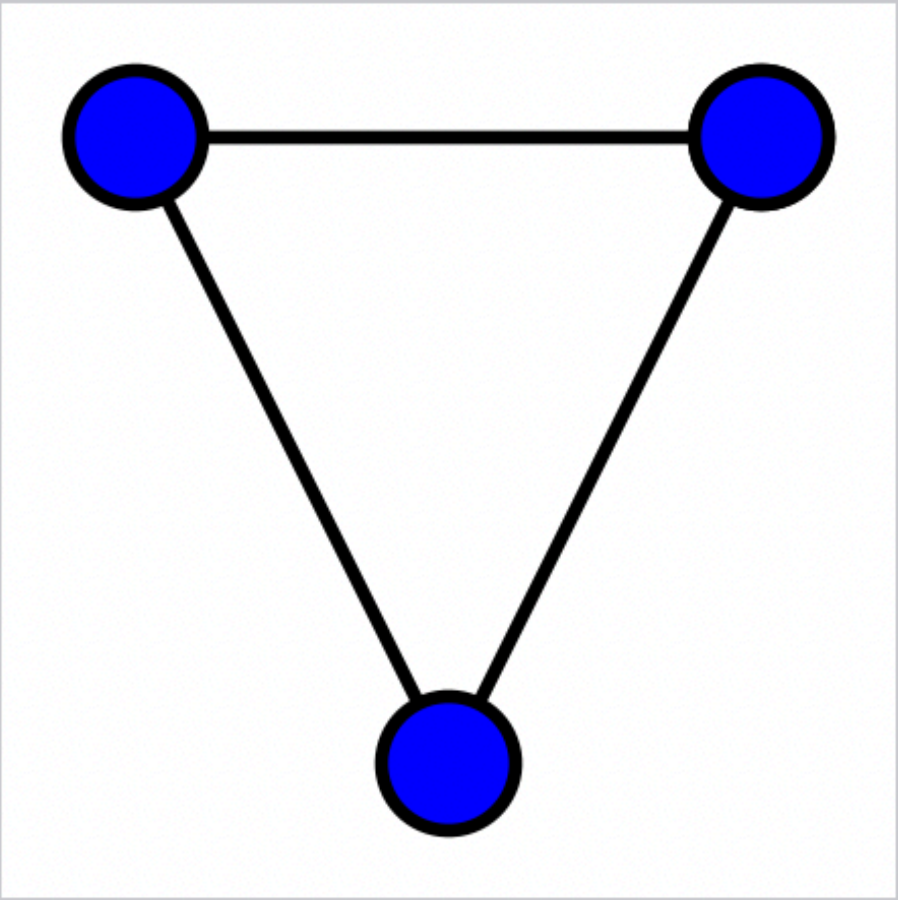
Graph [edit]

In one restricted but very common sense of the term,[1][2] a **graph** is an ordered pair $G = (V, E)$ comprising:

- V, a set of **vertices** (also called **nodes** or **points**);
- $E \subseteq \{\{x, y\} \mid x, y \in V \text{ and } x \neq y\}$, a set of **edges** (also called **links** or **lines**), which are unordered pairs of vertices (that is, an edge is associated with two distinct vertices).

To avoid ambiguity, this type of object may be called precisely an **undirected simple graph**.

In the edge $\{x, y\}$, the vertices x and y are called the **endpoints** of the edge. The edge is said to **join** x and y and to be **incident** on x and on y. A vertex may exist in a graph and not belong to an edge. **Multiple edges**, not allowed under the definition above, are two or more edges that join the same two vertices.



A graph with three vertices and three edges.

In one more general sense of the term allowing multiple edges,[3][4] a **graph** is an ordered triple $G = (V, E, \phi)$ comprising:

- V, a set of **vertices** (also called **nodes** or **points**);
- E, a set of **edges** (also called **links** or **lines**);
- $\phi : E \to \{\{x, y\} \mid x, y \in V \text{ and } x \neq y\}$, an **incidence function** mapping every edge to an unordered pair of vertices (that is, an edge is associated with two distinct vertices).

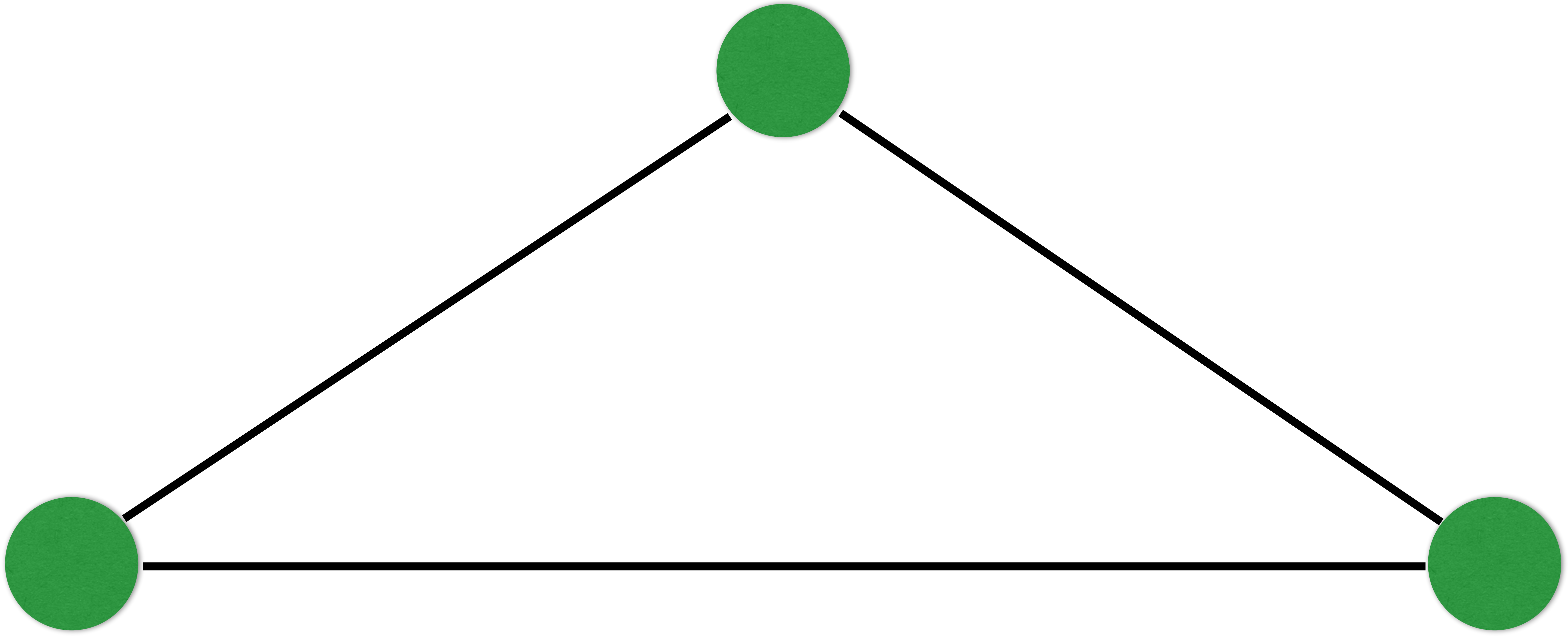
To avoid ambiguity, this type of object may be called precisely an **undirected multigraph**.

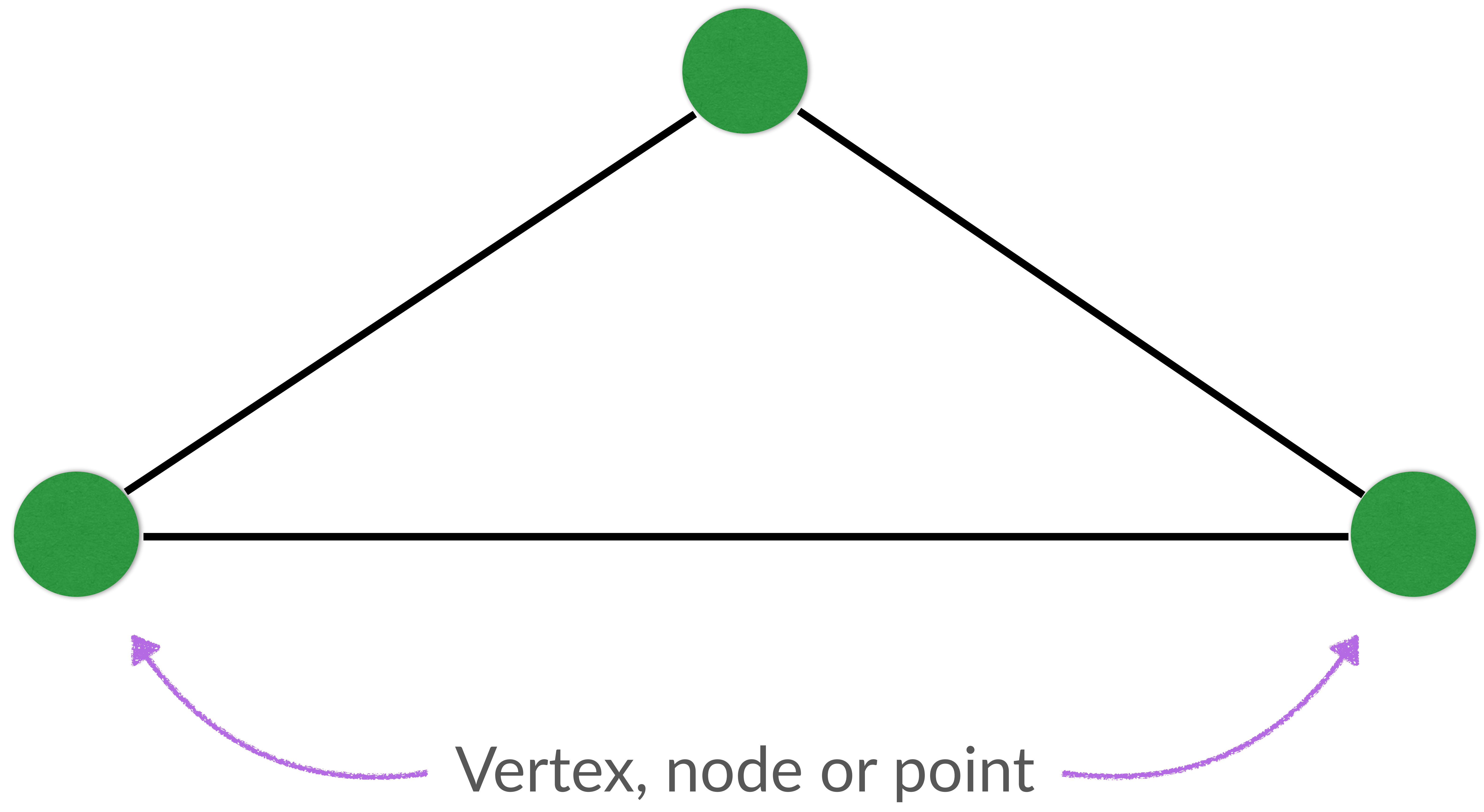
A **loop** is an edge that joins a vertex to itself. Graphs as defined in the two definitions above cannot have loops, because a loop joining a vertex x to itself is the edge (for an undirected simple graph) or is incident on (for an undirected multigraph) $\{x, x\} = \{x\}$ which is not in $\{\{x, y\} \mid x, y \in V \text{ and } x \neq y\}$. So to allow loops the definitions must be expanded. For undirected simple graphs, the definition of E should be modified to $E \subseteq \{\{x, y\} \mid x, y \in V\}$. For undirected multigraphs, the definition of ϕ should be modified to $\phi : E \to \{\{x, y\} \mid x, y \in V\}$. To avoid ambiguity, these types of objects may be called **undirected simple graph permitting loops** and **undirected multigraph permitting loops** (sometimes also **undirected pseudograph**), respectively.

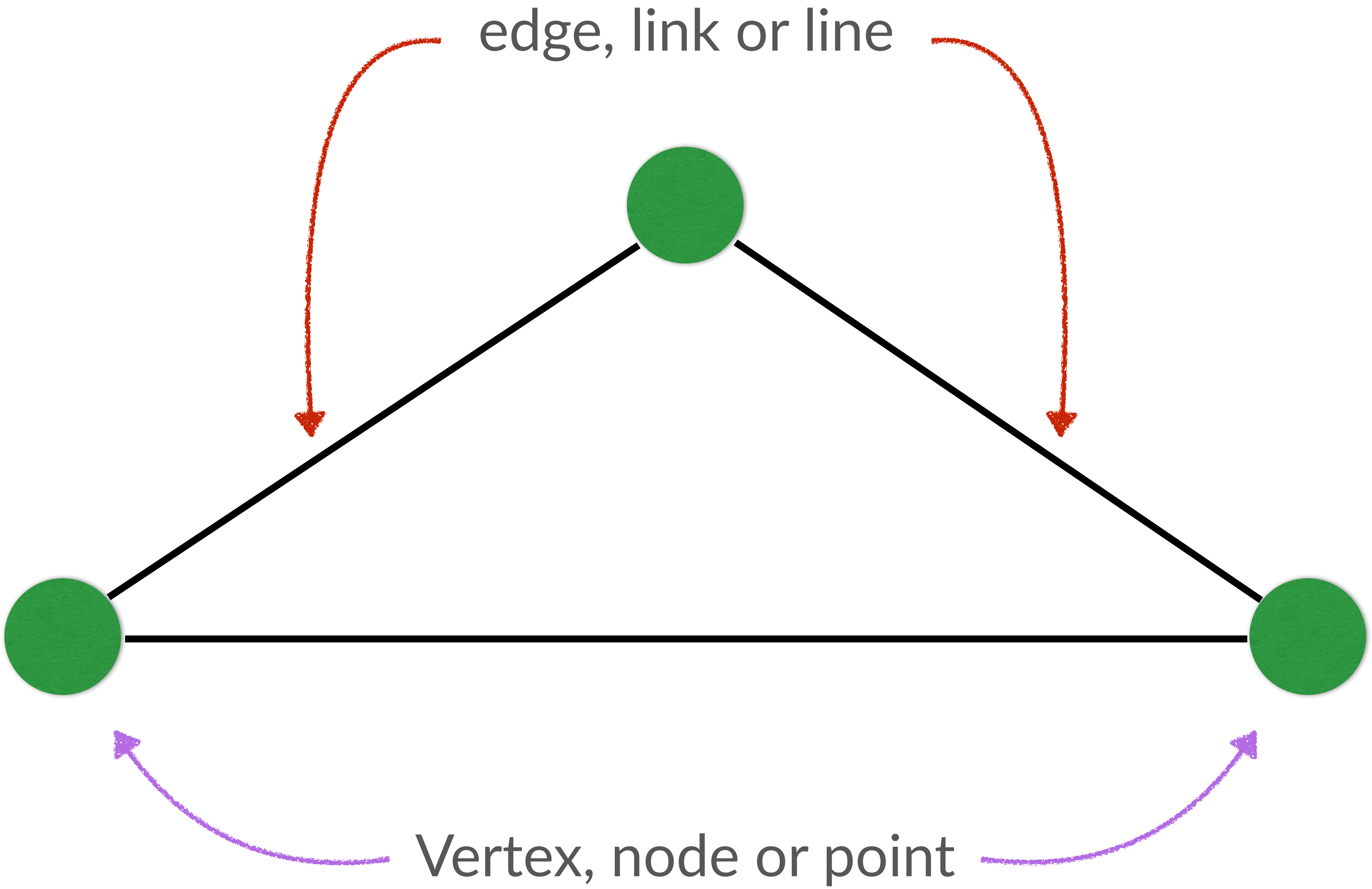
V and E are usually taken to be finite, and many of the well-known results are not true (or are rather different) for infinite graphs because many of the arguments fail in the infinite case. Moreover, V is often assumed to be non-empty, but E is allowed to be the empty set. The **order** of a graph is $|V|$, its number of vertices. The **size** of a graph is $|E|$, its number of edges. The **degree** or **valency** of a vertex is the number of edges that are incident to it, where a loop is counted twice. The **degree** of a graph is the maximum of the degrees of its vertices.

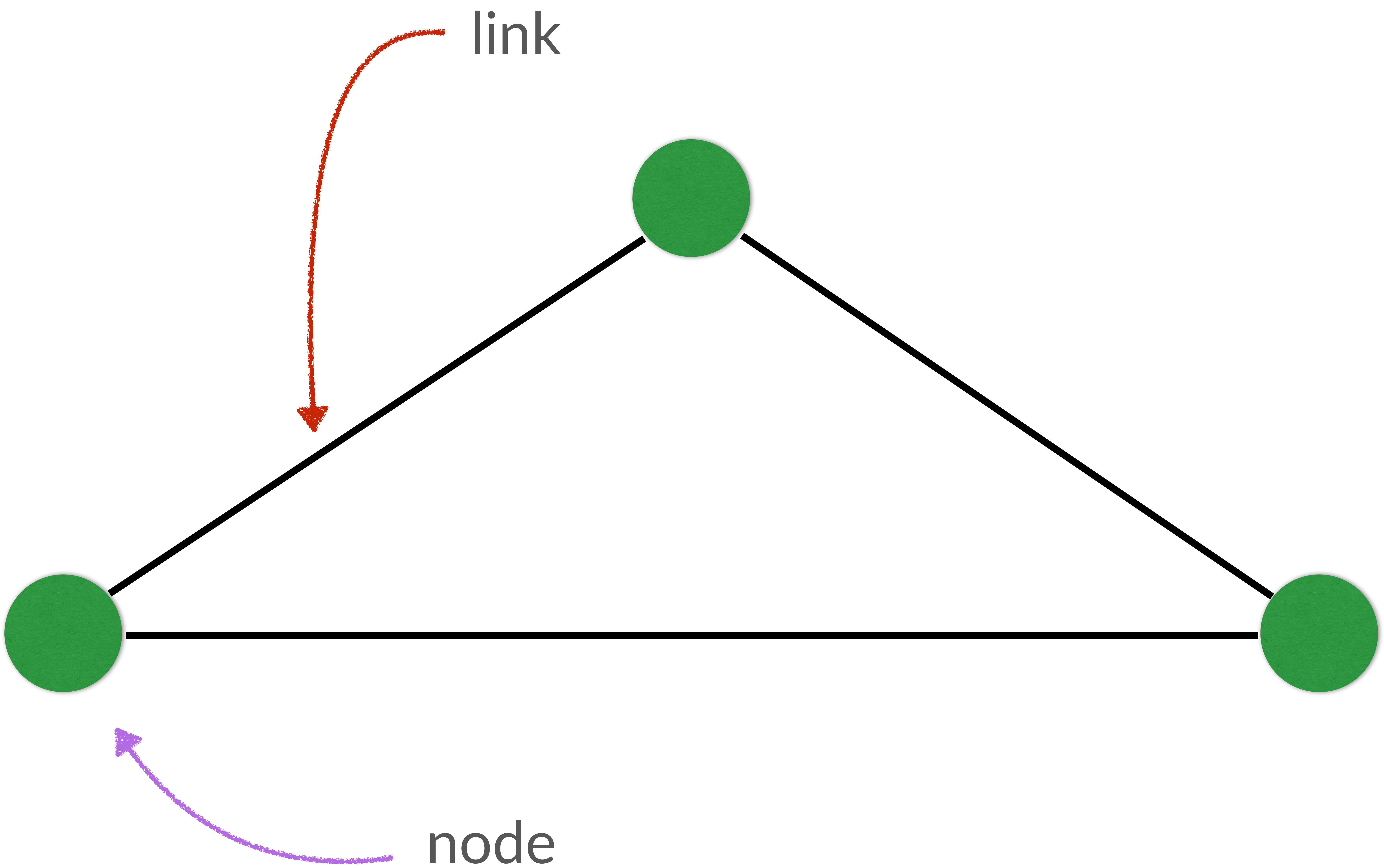
In an undirected simple graph of order *n*, the maximum degree of each vertex is *n* – 1 and the maximum size of the graph is *n*(*n* – 1)/2.

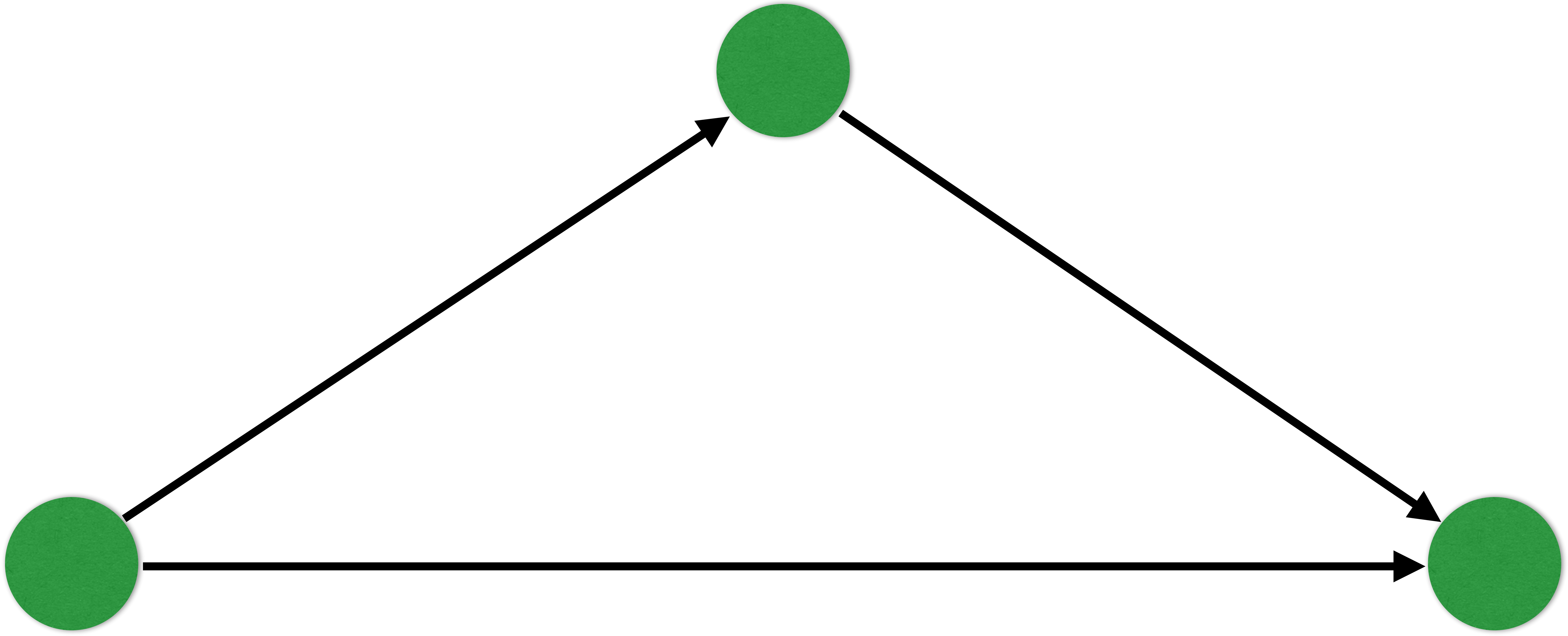
The edges of an undirected simple graph permitting loops G induce a symmetric homogeneous relation ~ on the vertices of G that is called the **adjacency relation** of G. Specifically, for each edge (x, y), its endpoints x and y are said to be **adjacent** to one another, which is denoted $x \sim y$.

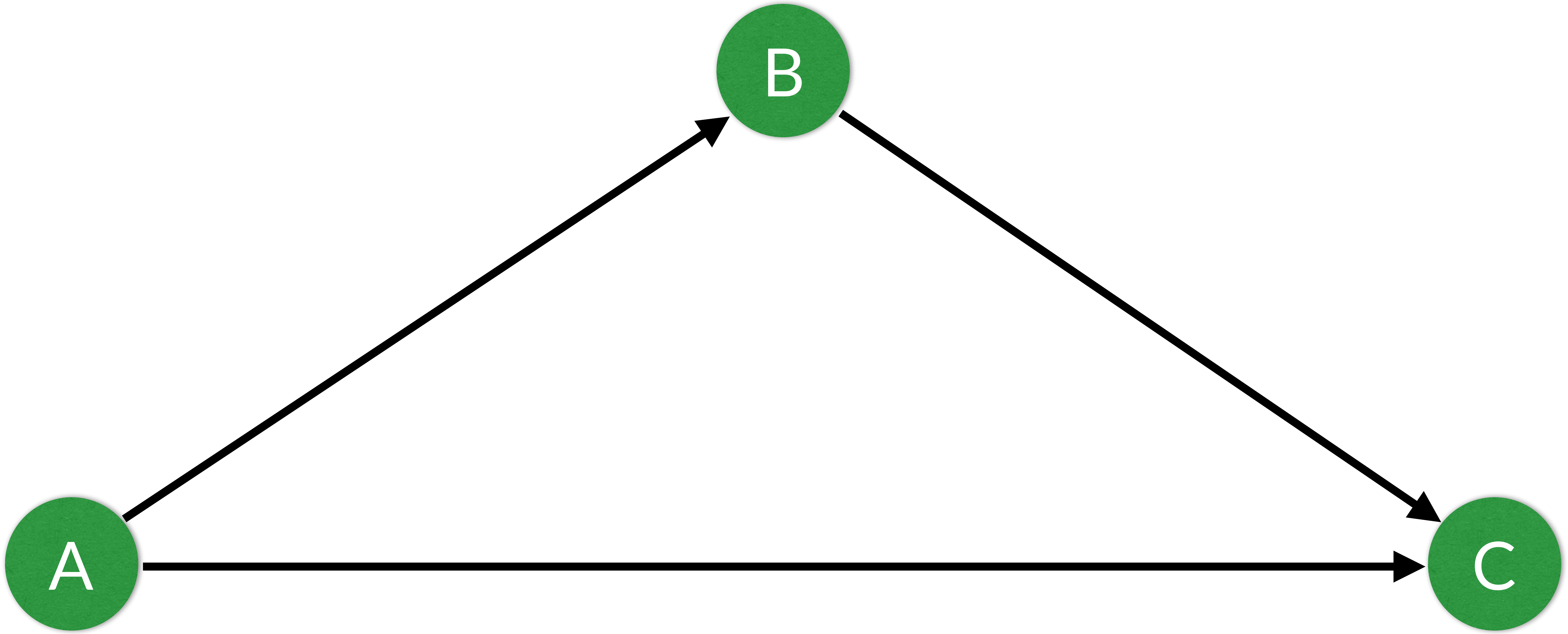



Vertex, node or point

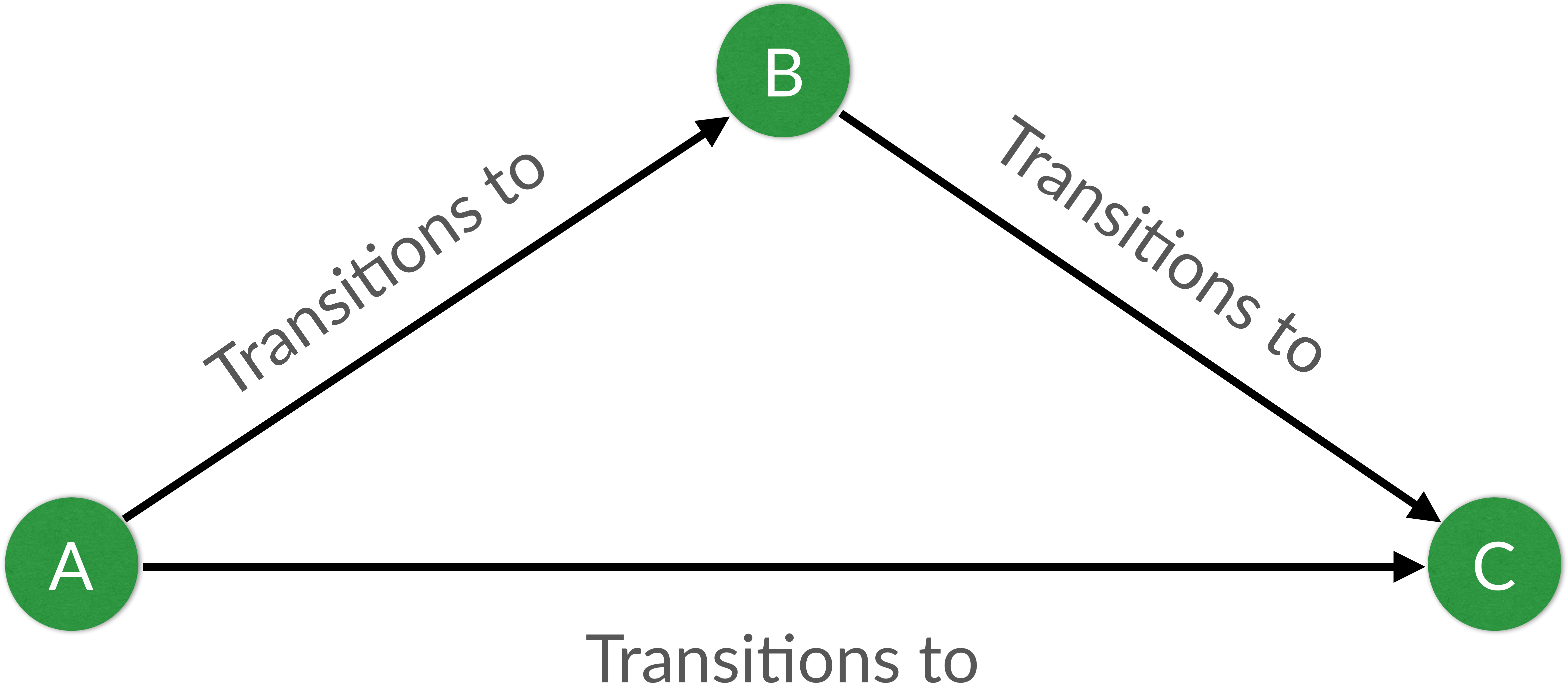

edge, link or line
Vertex, node or point


link
node

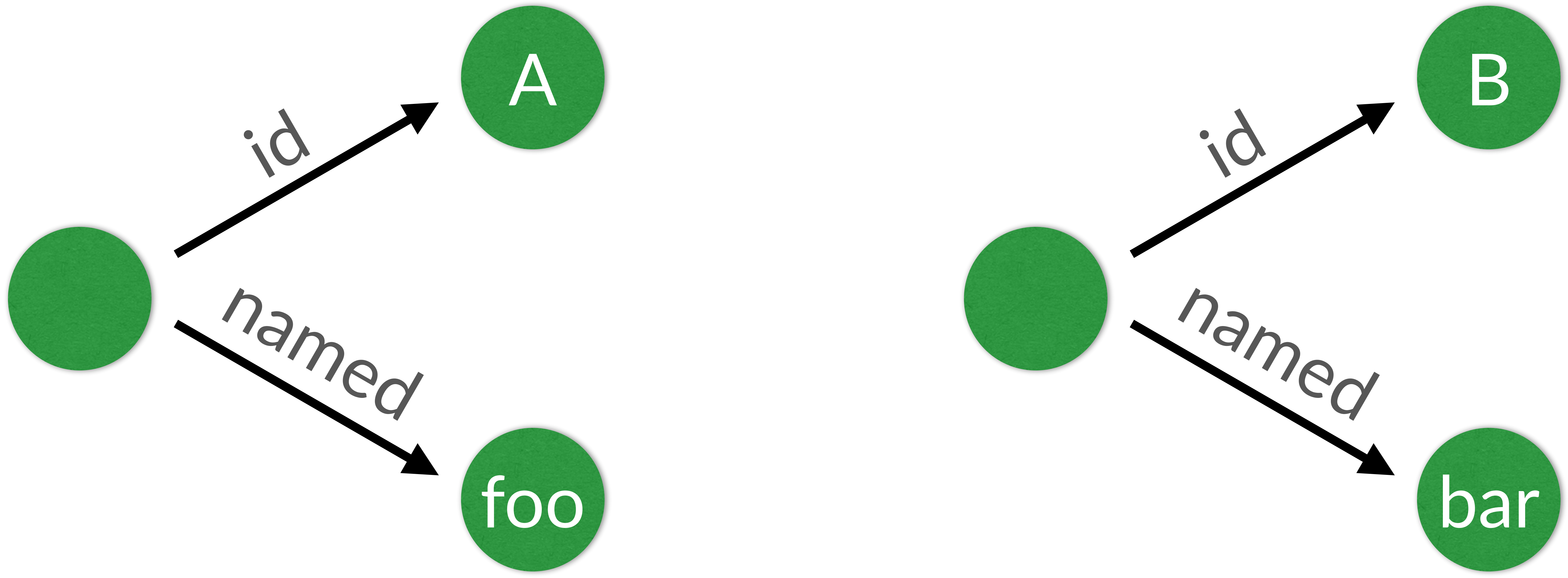



B
A
C

```go
package main

import "fmt"

func main() {
    fmt.Println("We should probably write some code!")
}
```


B
Transitions to
Transitions to
A
C
Transitions to

```go
package main

import "fmt"

func main() {
    fmt.Println("Let's add relationships")
}
```


A
id
named
foo
B
id
named
bar

```
{
  id: "A"
  name: "foo"
}
```

```
{
  id: "B"
  name: "bar"
}
```

```go
package main

import "fmt"

func main() {
    fmt.Println("Let's write more code")
}
```

```go
package main

import "fmt"

func main() {
    fmt.Println("Introducing Neo4j")
}
```

```go
package main

import "fmt"

func main() {
    fmt.Println("A basic population")
}
```

```go
package main

import "fmt"

func main() {
	fmt.Println("\"Contact\"")
}
```

```go
package main

import "fmt"

func main() {
	fmt.Println("Infection")
}
```

```go
package main

import "fmt"

func main() {
    fmt.Println("Location")
}
```

```go
package main

import "fmt"

func main() {
	fmt.Println("Trace")
}
```

Dom Davis
@idomdavis
about.me/idomdavis