Chapter 1

Directed Graphs

Colored operads are closely related to rooted trees, which are a certain type of directed graphs with specified inputs and output. Before discussing rooted trees, in this chapter we discuss some basic concepts about graphs that will be used throughout the rest of this book. Some of the definitions in this chapter can also be found in standard graph theory books, such as [**Bol98**].

1.1. Set Notations

In this section, we recall some basic notations about sets and functions. The definitions of various types of graphs as well as categories will require the following notations about sets.

Suppose S is a set.

(1) The *empty set* is written as $\varnothing$. It is the set with no elements.

(2) $x \in S$ means x is an element in S.

(3) $x \notin S$ means x is not an element in S.

(4) If S is a finite set, then $|S|$ denotes the number of elements in S.

(5) For an integer $n \geq 1$, the *n-fold product* $S^{\times n}$ denotes the set of n-tuples $(x_1, \ldots, x_n)$ of elements $x_1, \ldots, x_n$ in S.

(6) Denote by

$$S^{\underline{2}} = \Big\{\{x, y\} : x, y \in S,\, x \neq y\Big\} \tag{1.1.1}$$

the set of unordered pairs in S not of the form $\{x, x\}$ for $x \in S$.

Suppose T and U are also sets. The following notations are about functions.

(1) A *function* with domain S and codomain T is written as either

$$f : S \longrightarrow T \qquad \text{or} \qquad S \xrightarrow{\ f\ } T\ .$$

Such a function is a rule that assigns to each element $x \in S$ an element $f(x) \in T$, called the *image* of x under f. A function is also called a *map*. We sometimes write $x \longmapsto f(x)$ to indicate that the image of x is $f(x)$.

(2) The *identity function* on S is written as Id_S or just Id. It is the function $\mathrm{Id}_S(x) = x$ for $x \in S$.

(3) If $f : S \longrightarrow T$ and $g : T \longrightarrow U$ are functions, their *composition* is the function

$$gf = g \circ f : S \longrightarrow U, \quad (gf)(x) = g(f(x)) \quad \text{for} \quad x \in S.$$

(4) A *bijection* $f : S \longrightarrow T$ is a function that is both one-to-one and onto. In other words, there exists a unique function $f^{-1} : T \longrightarrow S$ such that

$$f^{-1}f = \mathrm{Id}_S \quad \text{and} \quad ff^{-1} = \mathrm{Id}_T\,.$$

A bijection is sometimes denoted by the symbol $\cong$. Call f^{-1} the *inverse* of f.

Suppose $S, T, S_1, \ldots, S_n$ are sets for some $n \geq 1$. The following notations are about constructions involving multiple sets.

(1) $S \subseteq T$ means S is a *subset* of T. So $S \subseteq T$ means every element $x \in S$ is also an element in T.

(2) If S is a subset of T, then the *difference* $T \smallsetminus S$ means the subset of elements $t \in T$ such that $t \notin S$.

(3) $S \cap T$ means the *intersection* of S and T. So $x \in S \cap T$ if and only if $x \in S$ and $x \in T$.

(4) S and T are *disjoint* if $S \cap T = \varnothing$.

(5) The *product*

$$S_1 \times \cdots \times S_n = \prod_{i=1}^{n} S_i = \Big\{(x_1, \ldots, x_n) : x_i \in S_i \text{ for } 1 \leq i \leq n\Big\} \tag{1.1.2}$$

is the set of n-tuples in which the ith entry is an element in S_i for each i.

(6) Suppose the sets S_i's are pairwise disjoint. The *coproduct*, also called the *disjoint union*, is the set

$$S_1 \sqcup \cdots \sqcup S_n = \coprod_{i=1}^{n} S_i = \Big\{x : x \in S_i \text{ for some } 1 \le i \le n\Big\}. \tag{1.1.3}$$

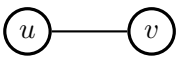
(7) If S and T are disjoint, $x \in S$, and $y \in T$, then the *quotient*

$$\frac{S \sqcup T}{(x \sim y)} \tag{1.1.4}$$

is the set consisting of elements in S and in T, except that x and y are now identified as one element.

1.2. Graphs

Later we are mostly interested in rooted trees with some extra structures. In this section, we begin with some basic definitions concerning graphs that are needed to define rooted trees. A graph is essentially what one imagines it should be, with some nodes and edges connecting pairs of nodes. So a typical edge looks like this:



As we go further, we will build upon the following set-theoretic definition of a graph.

Definition 1.2.1. A *graph* G is an ordered pair

$$(V, E)$$

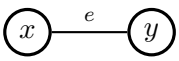
of disjoint sets in which E is a subset of $V^{\underline{2}}$ (1.1.1).

(1) An element in V is called an *abstract vertex*.

(2) An element

$$e = \{x, y\} \in E$$

is called an *edge* with *abstract end-vertices* $x \in V$ and $y \in V$. In this case, we say e is *adjacent to* the abstract vertices x and y. Such an edge is depicted as follows:



(1.2.2)

Since each edge is an unordered pair of abstract vertices, it does not matter whether we write an edge as $\{x, y\}$ or $\{y, x\}$.

(3) Say that a graph G is *finite* (resp., *non-empty*) if both V and E are finite sets (resp., non-empty sets).

(4) A *path* P in a graph G is an ordered list of abstract vertices

$$P = (x_0, x_1, \ldots, x_l) \tag{1.2.3}$$

for some $l \geq 1$ such that

$$e_i = \{x_{i-1}, x_i\} \in E \quad \text{for each} \quad 1 \leq i \leq l.$$

Call l the *length* of the path. We say that such a path is *from* x_0 *to* x_l, that each edge e_i is *in* P, and that P *contains* e_i.

(5) A *trail* is a path $(x_0, \ldots, x_l)$ whose edges $e_i = \{x_{i-1}, x_i\}$ for $1 \leq i \leq l$ are all distinct.

(6) A *cycle* is a path as in (1.2.3) such that
- the abstract vertices x_j for $0 \leq j < l$ are all distinct and
- $x_0 = x_l$.

Note that a cycle has length at least 3.

(7) A *forest* is a graph with no cycles.

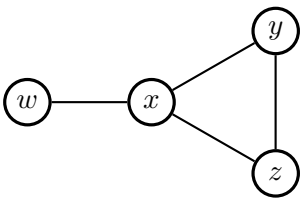
(8) Say that a graph G is *connected* if, for each pair of distinct abstract vertices $x, y \in V$, there exists a path P as in (1.2.3) such that $x_0 = x$ and $x_l = y$.

Convention 1.2.4. Throughout the rest of this book, we will only consider graphs that are both non-empty and finite. So from now on, a graph means a non-empty finite graph.

Example 1.2.5. Consider the graph $G = (V, E)$ with

- $V = \{w, x, y, z\}$;
- E consisting of $\{w, x\}$, $\{x, y\}$, $\{y, z\}$, and $\{x, z\}$.

As in (1.2.2) each abstract vertex will be drawn as a small disk, possibly with the name of the vertex inside of it. Each edge will be drawn as a line connecting its abstract end-vertices. So the graph G may be drawn as follows:



We call such a picture a *presentation* of the graph. Since a presentation uniquely determines a graph, in what follows we will sometimes only draw the graph without explicitly writing down the sets V and E.

In this graph G:

- One path of length 3 is given by $\{w, x, y, x\}$. Note that the edge $\{x, y\}$ is in this path twice, so this path is not a trail.
- One trail of length 4 is given by $\{w, x, y, z, x\}$. This trial is not a cycle.
- One cycle of length 3 is given by $\{x, y, z, x\}$.

Remark 1.2.6. Suppose $G = (V, E)$ is a graph.

(1) We deviate slightly from standard graph theory terminology by calling elements in V *abstract vertices*, instead of just vertices. The reason is that we are reserving the word *vertex* for a more special kind of abstract vertex, which will appear in Definition 1.4.1 below.

(2) By definition, for a pair of abstract vertices $x, y \in V$, there is at most one edge $\{x, y\}$ with abstract end-vertices x and y. In standard graph theory terminology, we do *not* allow multigraphs.

(3) Furthermore, by definition E does not contain any *loops* $\{x, x\}$ at any abstract vertex x. The reason we are making such restrictions from the beginning is that we are only considering colored operads in this book, and such graphs will suffice. On the other hand, if one wants to study bigger cousins of colored operads—such as wheeled operads and (wheeled) props—then more general graphs and more sophisticated graph-theoretic machinery are needed. The reader may consult [**YJ15**] for a general framework for studying (wheeled) props, their related graphs, and so forth.

(4) In a given path $(x_0, \ldots, x_l)$, some edges $e_i = \{x_{i-1}, x_i\}$ may be repeated. When that happens, one may remove some edges from the path to make it into a trail with the same x_0 and x_l. In Exercise (1) below, the reader is asked to prove this statement precisely.

We will consider isomorphism classes of graphs with the following notion of isomorphisms.

Definition 1.2.7. Suppose $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ are two graphs. An *isomorphism*

$$G_1 \xrightarrow{\ \zeta\ } G_2$$

consists of two bijections,

$$V_1 \xrightarrow[\cong]{\zeta_V} V_2 \quad \text{and} \quad E_1 \xrightarrow[\cong]{\zeta_E} E_2\ ,$$

that preserve adjacency relations, in the sense that

$$e = \{x, y\} \in E_1 \quad \text{if and only if} \quad \zeta_E(e) = \big\{\zeta_V(x), \zeta_V(y)\big\} \in E_2.$$

To simplify the notation, we will usually write both ζ_V and ζ_E as just ζ.

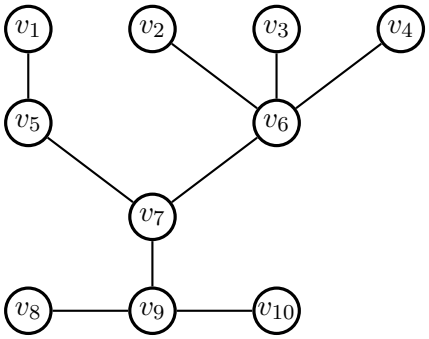
Example 1.2.8. Being a forest and being connected are both properties that are preserved by isomorphisms (Exercise (2)).

Example 1.2.9. Consider the connected forest $G = (V, E)$ with

- $V = \{v_i : 1 \leq i \leq 10\}$;
- E consisting of $\{v_1, v_5\}$, $\{v_2, v_6\}$, $\{v_3, v_6\}$, $\{v_4, v_6\}$, $\{v_5, v_7\}$, $\{v_6, v_7\}$, $\{v_7, v_9\}$, $\{v_8, v_9\}$, and $\{v_9, v_{10}\}$.

This connected forest has a presentation:

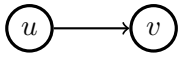
(1.2.10)



We will revisit this example later as further graph-theoretic notions are defined.

1.3. Directed Graphs

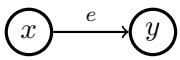
To define rooted trees later, we will need the concept of a directed graph, which is a graph in which every edge has a specified direction. So an edge in a directed graph looks like



We make this precise as follows.

Definition 1.3.1. A *directed graph* is a graph $G = (V, E)$ (Definition 1.2.1) in which each edge has a chosen order, called an *orientation*; i.e., each edge is an ordered pair of abstract vertices.

(1) Suppose $e = (x, y)$ is an edge in a directed graph. It will be depicted as



   (a) Call x and y its *initial vertex* and *terminal vertex*, respectively.
   (b) Call e an *outgoing edge of* x and an *incoming edge of* y.

(2) For an abstract vertex v in a directed graph, the *set of incoming edges* and *the set of outgoing edges* are written as

$$\mathsf{in}(v) \quad \text{and} \quad \mathsf{out}(v), \tag{1.3.2}$$

respectively.

Definition 1.3.3. Suppose $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ are directed graphs. An *isomorphism*

$$G_1 \xrightarrow{\ \zeta\ } G_2$$

is a graph isomorphism (Definition 1.2.7) that preserves edge orientations; i.e.,

$$e = (x, y) \in E_1 \quad \text{if and only if} \quad \zeta(e) = \big(\zeta(x), \zeta(y)\big) \in E_2.$$

Example 1.3.4. Consider the connected forest

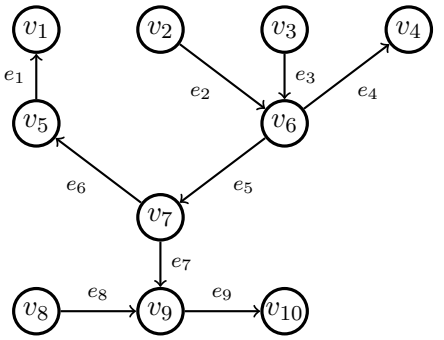
$$G = \big(V = \{v_i\}_{1 \le i \le 10}, E\big)$$

in Example 1.2.9. One way to make it into a directed graph is by specifying the following orientations for its edges:

$$e_1 = (v_5, v_1),\ e_2 = (v_2, v_6),\ e_3 = (v_3, v_6),\ e_4 = (v_6, v_4),\ e_5 = (v_6, v_7),$$
$$e_6 = (v_7, v_5),\ e_7 = (v_7, v_9),\ e_8 = (v_8, v_9),\ e_9 = (v_9, v_{10}).$$

A presentation of this directed connected forest is

(1.3.5)



Here we have

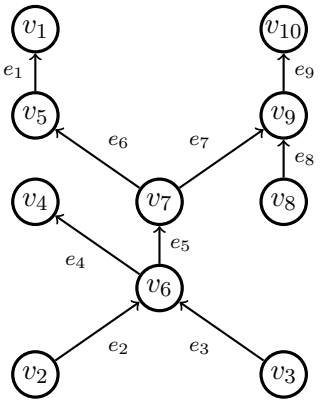
$$\mathsf{in}(v_1) = \{e_1\}, \quad \mathsf{out}(v_1) = \varnothing, \quad \mathsf{in}(v_6) = \{e_2, e_3\}, \quad \mathsf{out}(v_6) = \{e_4, e_5\},$$
$$\mathsf{in}(v_7) = \{e_5\}, \quad \mathsf{out}(v_7) = \{e_6, e_7\},$$

and so forth.

In the presentation (1.3.5), the edges are pointing in a number of different directions, so it is a bit hard to understand. The following is another

presentation of the same directed connected forest that conveys a better sense of direction:

(1.3.6)



In this presentation, all the edges are pointing upward. In what follows, whenever possible we will draw directed graphs this way, with edges pointing upward. For a directed connected forest, this is indeed always possible. The proof of this statement is Exercise (3) below.

1.4. Directed (m, n)-Graphs

Rooted trees are directed graphs in which there are specified inputs and a specified output, satisfying some further conditions. An input is an abstract vertex that has no incoming edges and has exactly one outgoing edge. Likewise, an output is an abstract vertex that has no outgoing edges and has exactly one incoming edge. So an input and an output look like this:

We make this precise as follows.

Definition 1.4.1. Suppose $m, n \geq 0$.

(1) A *directed* (m, n)*-graph* is a quadruple

$$G = (V, E, \mathsf{in}_G, \mathsf{out}_G)$$

consisting of

(a) a directed graph (V, E) (Definition 1.3.1) and

(b) disjoint subsets

$$\mathsf{in}_G \subseteq V \quad \text{and} \quad \mathsf{out}_G \subseteq V$$

such that the following three conditions hold.

- $|\mathsf{in}_G| = m$ and $|\mathsf{out}_G| = n$.
- For each $v \in \mathsf{in}_G$,
$$|\mathsf{in}(v)| = 0 \quad \text{and} \quad |\mathsf{out}(v)| = 1,$$
where $\mathsf{in}(v)$ and $\mathsf{out}(v)$ are the sets of incoming edges and of outgoing edges of v (Definition 1.3.1).
- For each $v \in \mathsf{out}_G$,
$$|\mathsf{in}(v)| = 1 \quad \text{and} \quad |\mathsf{out}(v)| = 0.$$

(2) In such a directed (m,n)-graph G, define the subset

$$\mathsf{Vt}_G = \Big\{ v \in V : v \notin (\mathsf{in}_G \sqcup \mathsf{out}_G) \Big\}. \tag{1.4.2}$$

(3) An abstract vertex $v \in V$ in a directed (m,n)-graph G is called
- an *input* if $v \in \mathsf{in}_G$;
- an *output* if $v \in \mathsf{out}_G$;
- a *vertex* if $v \in \mathsf{Vt}_G$.

(4) An edge $e = (x,y) \in E$ in a directed (m,n)-graph G is called
- an *input edge* if $x \in \mathsf{in}_G$;
- an *output edge* if $y \in \mathsf{out}_G$;
- an *internal edge* if $x, y \in \mathsf{Vt}_G$.

An *external edge* is an edge that is an input edge, an output edge, or both.

(5) The *set of internal edges* in G is denoted by Int_G.

Definition 1.4.3. Suppose

$$G_1 = (V_1, E_1, \mathsf{in}_{G_1}, \mathsf{out}_{G_1}) \quad \text{and} \quad G_2 = (V_2, E_2, \mathsf{in}_{G_2}, \mathsf{out}_{G_2})$$

are directed (m,n)-graphs for some $m, n \geq 0$. An *isomorphism*

$$G_1 \xrightarrow{\ \zeta\ } G_2$$

is a directed graph isomorphism (Definition 1.3.3) that preserves the inputs and the outputs, in the sense that the restrictions of ζ,

$$\mathsf{in}_{G_1} \xrightarrow[\cong]{\ \zeta\ } \mathsf{in}_{G_2} \quad \text{and} \quad \mathsf{out}_{G_1} \xrightarrow[\cong]{\ \zeta\ } \mathsf{out}_{G_2}\ ,$$

are bijections.

Remark 1.4.4. Suppose $G = (V, E, \mathsf{in}_G, \mathsf{out}_G)$ is a directed (m,n)-graph.

(1) The set V of abstract vertices decomposes into a disjoint union

$$V = \mathsf{in}_G \sqcup \mathsf{out}_G \sqcup \mathsf{Vt}_G. \tag{1.4.5}$$

In other words, an abstract vertex is an input, an output, or a vertex.

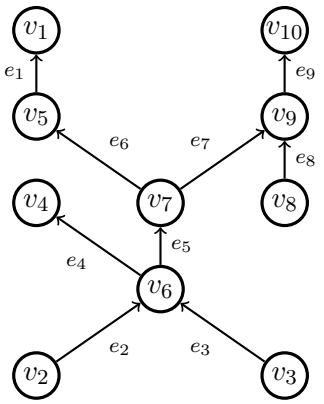
(2) An edge is either an external edge or an internal edge, but it cannot be both.

Convention 1.4.6. By definition every input $v \in \mathsf{in}_G$ is adjacent to a unique input edge $e_v \in E$, which has v as its initial vertex. In what follows, we will often identify an input v with the unique input edge e_v adjacent to it, although technically they are not the same. We will also identify the set in_G of inputs with the set

$$\{e_v : v \in \mathsf{in}_G\} \subseteq E$$

of input edges. Likewise, the set out_G of outputs will often be identified with the set of output edges.

Example 1.4.7. A directed (m, n)-graph can be drawn by *omitting* the inputs and the outputs, drawing only the vertices and the edges. Input edges and output edges then appear as edges not adjacent to a vertex in one end (or both). For example, consider once again the directed connected forest $G = (V, E)$

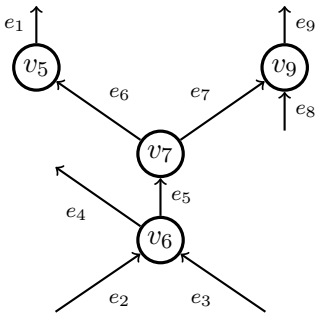


in (1.3.6). There are many directed (m, n)-graphs that can be made from it. Only the abstract vertices $\{v_2, v_3, v_8\}$ are eligible to be inputs. Only the abstract vertices $\{v_1, v_4, v_{10}\}$ are eligible to be outputs. Below are some examples.

(1) $G_1 = (V, E, \mathsf{in}_{G_1}, \mathsf{out}_{G_1})$ is a directed $(3, 3)$-graph with
- $\mathsf{in}_{G_1} = \{v_2, v_3, v_8\} \cong \{e_2, e_3, e_8\}$;
- $\mathsf{out}_{G_1} = \{v_1, v_4, v_{10}\} \cong \{e_1, e_4, e_9\}$;
- $\mathsf{Vt}_{G_1} = \{v_5, v_6, v_7, v_9\}$;
- $\mathsf{Int}_{G_1} = \{e_5, e_6, e_7\}$.

A presentation of it is as follows:

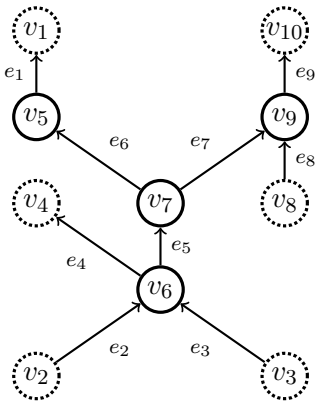
(1.4.8)



In such a presentation, an edge that is shown without a vertex at one end indicates that that end is in fact an input or an output. For example, in (1.4.8), e_1 is shown with only v_5 at its initial end, so the terminal end of e_1 is actually an output. Likewise, e_8 is shown with only v_9 at its terminal end, so its initial end is actually an input.

Compare the presentation (1.4.8) with the original directed connected forest G in (1.3.6). Here we draw the inputs and the outputs in G_1 as abstract vertices with dotted boundaries for easier comparison:

(1.4.9)

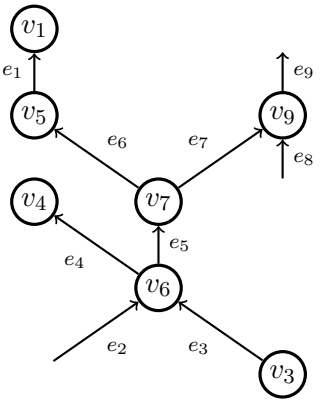


As can be seen from this presentation of G_1, it is actually easier to recognize the input edges and the output edges when we do *not* draw the inputs and the outputs, which are the dotted abstract vertices above.

(2) $G_2 = (V, E, \mathsf{in}_{G_2}, \mathsf{out}_{G_2})$ is a directed $(2,1)$-graph with
- $\mathsf{in}_{G_2} = \{v_2, v_8\} \cong \{e_2, e_8\}$;
- $\mathsf{out}_{G_2} = \{v_{10}\} \cong \{e_9\}$;
- $\mathsf{Vt}_{G_2} = \{v_1, v_3, v_4, v_5, v_6, v_7, v_9\}$;
- $\mathsf{Int}_{G_2} = \{e_1, e_3, e_4, e_5, e_6, e_7\}$.

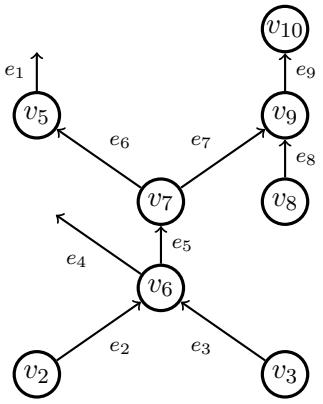
A presentation of it is as follows:

(1.4.10)



(3) $G_3 = (V, E, \mathsf{in}_{G_3}, \mathsf{out}_{G_3})$ is a directed $(0,2)$-graph with
- $\mathsf{in}_{G_3} = \varnothing$;
- $\mathsf{out}_{G_3} = \{v_1, v_4\} \cong \{e_1, e_4\}$;
- $\mathsf{Vt}_{G_3} = \{v_2, v_3, v_5, v_6, v_7, v_8, v_9, v_{10}\}$;
- $\mathsf{Int}_{G_3} = \{e_2, e_3, e_5, e_6, e_7, e_8, e_9\}$.

A presentation of it is as follows:



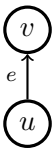
In general a given directed graph may be regarded as a directed (m,n)-graph in a number of different ways, depending on m and n and on which abstract vertices are chosen as inputs and outputs.

Example 1.4.11. Consider the directed forest $G = (V, E)$ with

- $V = \{u, v\}$;
- $E = \{e = (u, v)\}$.

A presentation of G is

(1.4.12)



The abstract vertex u is the only candidate for an input, and the abstract vertex v is the only candidate for an output. So there are four directed (m,n)-graphs that can be associated to G.

(1) $G_1 = (V, E, \mathsf{in}_{G_1}, \mathsf{out}_{G_1})$ is a directed $(0,0)$-graph with

(1.4.13) $$\mathsf{in}_{G_1} = \varnothing = \mathsf{out}_{G_1}, \quad \mathsf{Vt}_{G_1} = \{u, v\}, \quad \mathsf{Int}_{G_1} = \{e\},$$

and the same presentation as in (1.4.12). We call this the *dumbbell graph*.

(2) $G_2 = (V, E, \mathsf{in}_{G_2}, \mathsf{out}_{G_2})$ is a directed $(1,0)$-graph with
- $\mathsf{in}_{G_2} = \{u\} \cong \{e\}$;
- $\mathsf{out}_{G_2} = \varnothing$;
- $\mathsf{Vt}_{G_2} = \{v\}$;
- $\mathsf{Int}_{G_2} = \varnothing$.

A presentation of G_2 is as follows:

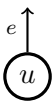
(1.4.14)

We call this the *lollipop graph*.

(3) $G_3 = (V, E, \mathsf{in}_{G_3}, \mathsf{out}_{G_3})$ is a directed $(0,1)$-graph with
- $\mathsf{in}_{G_3} = \varnothing$;
- $\mathsf{out}_{G_3} = \{v\} \cong \{e\}$;
- $\mathsf{Vt}_{G_3} = \{u\}$;
- $\mathsf{Int}_{G_3} = \varnothing$.

A presentation of G_3 is as follows:

(1.4.15)



We call this the *noon graph*. It is an example of a corolla (3.4.4).

(4) $G_4 = (V, E, \mathsf{in}_{G_4}, \mathsf{out}_{G_4})$ is a directed $(1,1)$-graph with
- $\mathsf{in}_{G_4} = \{u\} \cong \{e\}$;
- $\mathsf{out}_{G_4} = \{v\} \cong \{e\}$;
- $\mathsf{Vt}_{G_4} = \varnothing$;
- $\mathsf{Int}_{G_4} = \varnothing$.

A presentation of G_4 is as follows:

(1.4.16)

In this case e is both an input edge and an output edge (Definition 1.4.1). We call this the *exceptional edge*. It will play a prominent role in later chapters.

1.5. Exercises

(1) Suppose $G = (V, E)$ is a graph (Definition 1.2.1) and $x, y \in V$ such that there exists a path P (1.2.3) from x to y. Prove that there exists a trail Q from x to y such that all the edges in Q are also in P.

(2) Suppose G_1 and G_2 are graphs and $\zeta : G_1 \longrightarrow G_2$ is an isomorphism (Definition 1.2.7). Prove that:
   (a) G_1 is a forest if and only if G_2 is a forest.
   (b) G_1 is connected if and only if G_2 is connected.

(3) This exercise originated in Example 1.3.4. Make the following statement precise and then prove it:

   A directed connected forest can always be drawn with all the edges pointing upward.

   Here is one way to prove this statement. Suppose $G = (V, E)$ is a directed connected forest.
   - Define a *directed path* in G as an ordered list of edges
   $$e_1 = (x_0, x_1), \ldots, e_l = (x_{l-1}, x_l)$$
   for some $l \geq 1$ such that x_j for $0 \leq j < l$ are all distinct. Call l its *length*, x_0 its *initial vertex*, and x_l its *terminal vertex*.
   - Define the *height* of an abstract vertex $v \in V$ as the maximal lengths among all the directed paths with terminal vertex v.

   Now assign each abstract vertex $v \in V$ a point in the plane such that vertices of height n are assigned points on the horizontal line $y = n$. Next draw the edges. Prove that all the edges are pointing upward.

1.6. Notes

Our convention of drawing directed (m, n)-graphs with an upward flow follows Markl [**Mar08**]. Some authors use the reverse orientation, especially for rooted trees, which we will discuss in Chapter 3.