More Operations on Relations

Composition: For $R: A \rightarrow B$ and $S: B \rightarrow C$, 

$$a (R \circ S) c ::= \exists b \in B \ a R b \land b S c$$

Inverse: The inverse of a relation $R$, denoted $R^{-1}$, is given by $b R^{-1} a$ iff $a R b$.

Inverse of functions can be defined in the same way. But, the inverse of a function $f$ may not always be a function. $f^{-1}$ is a function iff $f$ is injective.
More Operations on Relations

Composition: For $R: A \rightarrow B$ and $S: B \rightarrow C$,
$$a (R \circ S) c ::= \exists b \in B \ a R b \land b S c$$

Inverse: The inverse of a relation $R$, denoted $R^{-1}$, is given by
$$b R^{-1} a \iff a R b$$

- Inverse of functions can be defined in the same way.
- But, the inverse of a function $f$ may not always be a function
More Operations on Relations

Composition: For \( R: A \rightarrow B \) and \( S: B \rightarrow C \),
\[
a (R \circ S) c ::= \exists b \in B \ a R b \land b S c
\]

Inverse: The inverse of a relation \( R \), denoted \( R^{-1} \), is given by
\[
b R^{-1} a \iff a R b
\]

- Inverse of functions can be defined in the same way.
- But, the inverse of a function \( f \) may not always be a function
  - \( f^{-1} \) is a function iff \( f \) is injective
A relation $R$ from a set to itself is often more interesting than a relation from one set to another.

- $R$ can be composed with itself
- can be represented using a directed graph or digraph.
A Digraph $G = (V, E)$ where
- $V$ is the set of vertices or nodes, and
- $E$ is a set of directed edges of the form $(u, v)$ where $u, v \in V$. 

$G$ is nothing but a relation from $V$ to $V$. In fact, we have already called the "arrows" in visual representation of relations as graphs!
A Digraph $G = (V, E)$ where
- $V$ is the set of vertices or nodes, and
- $E$ is a set of directed edges of the form $(u, v)$ where $u, v \in V$.
  - $u$ is called the tail and $v$ the head of the edge.
  - An edge with the same head and tail is called a self-loop.

Sometimes we call these $V(G)$ and $E(G)$. 
A Digraph $G = (V, E)$ where

- $V$ is the set of vertices or nodes, and
- $E$ is a set of directed edges of the form $(u, v)$ where $u, v \in V$.
  - $u$ is called the tail and $v$ the head of the edge.
  - An edge with the same head and tail is called a self-loop.

Sometimes we call these $V(G)$ and $E(G)$.

$G$ is nothing but a relation from $V$ to $V$

In fact, we have already called the “arrows” in visual representation of relations as graphs!
Degree: number of arrows coming into ("in" degree) or the number of arrows going out ("out" degree)

Property of Degrees in a Graph

\[ \sum_{v \in V(G)} \text{indeg}(v) = \sum_{v \in V} \text{outdeg}(v) \]
Walks and Paths

A walk in a graph $G$ is a sequence of vertices $v_1, v_2, \ldots, v_n$ such that

- Every $v_i \in V(G)$, and
- $(v_j, v_{j+1}) \in E(G)$
- We say that the walk starts at $v_1$ and ends at $v_n$
- The length of the walk is $n - 1$
Walks and Paths

- A walk in a graph $G$ is a sequence of vertices $v_1, v_2, \ldots, v_n$ such that
  - Every $v_i \in V(G)$, and
  - $(v_j, v_{j+1}) \in E(G)$
  - We say that the walk starts at $v_1$ and ends at $v_n$
  - The length of the walk is $n - 1$

- A closed walk is a walk with $v_1 = v_n$.

- A path is a walk where all the vertices are distinct.

- A cycle is a closed walk where $v_1, \ldots, v_{n-1}$ form a path.
Some Properties of Walks

Theorem

- The shortest walk between two vertices \( u \) and \( v \) is a path.
- The shortest closed walk through a vertex \( v \) is a cycle.
Some Properties of Walks

Theorem

- The shortest walk between two vertices $u$ and $v$ is a path.
- The shortest closed walk through a vertex $v$ is a cycle.

$\text{dist}(u, v)$: is the shortest path between $u$ and $v$. 
Some Properties of Walks

**Theorem**

- The shortest walk between two vertices \( u \) and \( v \) is a path.
- The shortest closed walk through a vertex \( v \) is a cycle.

\( \text{dist}(u, v) \): is the shortest path between \( u \) and \( v \).

**The Triangle Inequality**

\[
\text{dist}(u, v) \leq \text{dist}(u, w) + \text{dist}(w, v)
\]
Euler Tour: A closed walk that visits every edge in the graph exactly once.

Hamiltonian Tour: A cycle that visits every vertex exactly once.
Euler Tour Example
Euler Tour: Necessary and Sufficient Condition

\[ \forall v \in V \quad indeg(v) = outdeg(v) \]
Hamiltonian Tour: Example

Also known as *Traveling Salesman Problem*
Think of $E$ as defining a relation from $V$ to $V$

- What do walks of length 2 denote?
- What about walks of length $n$?
Properties of a Relation $R : V \rightarrow V$

**Reflexive:** $\forall a \ aRa$
- Graph has self-loops at every vertex

**Irreflexive:** $\forall a \ a \not\in R a$
- No self-loops
Properties of a Relation $R : V \rightarrow V$

**Symmetric:** $\forall a, b \ aRb \rightarrow bRa$
- Edges come in pairs: we can merge them into one and remove arrows, leading to undirected graphs

**Anti-symmetric:** $\forall a, b \ aRb \rightarrow (a = b \ \lor \ b \not\sim a)$
Properties of a Relation $R : V \rightarrow V$

Transitive: $\forall a, b, c \ aRb \land bRc \rightarrow aRc$

- Any vertex $b$ reachable from $a$ is reachable in a single step.
Properties of a Relation $R : V \rightarrow V$

**Reflexive:** $\forall a \ aRa$

**Irreflexive:** $\forall a \ a \not\in R a$

**Symmetric:** $\forall a, b \ aRb \rightarrow bRa$

**Anti-symmetric:** $\forall a, b \ aRb \rightarrow (a = b \lor b \not\in R a)$

**Transitive:** $\forall a, b, c \ aRb \land bRc \rightarrow aRc$
Closure Operations

- Start with a relation, introduce additional edges implied by properties discussed before:
  - Reflexive closure
  - Symmetric closure
  - Transitive closure
Reflexive Closure

- Add self-loops at every vertex
- Examples
Symmetric Closure

- Add edge from $b$ to $a$ whenever there is an edge from $a$ to $b$
- Examples
Transitive Closure

- Add edge from $a$ to $c$ iff there is an edge from $a$ to $b$ and another edge from $b$ to $c$.
- Examples
Properties of a Relations

Partial Orders: Anti-symmetric and transitive. Forms Directed Acyclic Graphs (DAGs)

Linear order: Partial order where every pair of elements is comparable
  i.e., either $aRb$ or $bRa$ holds.
DAGs, Dependencies, Topological Sort and Scheduling ...
DAGs, Dependencies, Topological Sort and Scheduling ...
Properties of a Relations: Equivalence Relations

- Reflexive, Symmetric and Transitive
- Partition a domain into *Equivalence Classes*

\[ EC(a) = \{ b \mid aRb \} \]

- Examples
  - \( aRb \iff a, b \in \mathbb{N}, a \mod n = b \mod n \)
  - Connected components of a graph