Let $A$ and $B$ be sets,

A **binary relation from** $A$ **to** $B$ is a subset of $A \times B$

Let $A$ be a set,

A **binary relation on** $A$ is a subset of $A \times A$
Let $R$ be a relation on $A$.

**R is reflexive iff** $(a,a) \in R$ for every $a \in A$

**R is symmetric iff** $(a,b) \in R$ implies $(b,a) \in R$

**R is antisymmetric iff** $(a,b) \in R$ and $a \neq b$ implies $(b,a) \notin R$

**R is transitive iff** $(a,b) \in R$ and $(b,c) \in R$ implies $(a,c) \in R$
Functions

A function \( f : A \to B \) (\( A \) as input and \( B \) as output) is a special type of relation.

A function \( f \) from \( A \) to \( B \) is a relation from \( A \) to \( B \) such that:
for every \( a \in A \), there is exactly one \( b \in B \) with \( (a, b) \in f \)

i.e., for every input \( a \in A \), there is one output \( b \in B \).
We denote this \( b \) by \( f(a) \).

Function composition: If \( f : A \to B \) and \( g : B \to C \) then their composition \( g \circ f : A \to C \) is defined by
\[
g \circ f (a) = g(f(a)) \]
Composing Relations

Let $R$ be a relation from $A$ to $B$. Let $S$ be a relation from $B$ to $C$.

The composition of $R$ and $S$, $S \circ R$ is the relation from $A$ to $C$ defined by:

$$S \circ R = \{(a, c) : \exists b \text{ such that } (a, b) \in R \text{ and } (b, c) \in S\}$$

Intuitively, a pair is in the composition if there is a “connection” from the first to the second.

The order of writing composition generalizes the function case
Examples

(a,b) ∈ Parent iff b is a parent of a
(a,b) ∈ Sister iff b is a sister of a

When is (x,y) ∈ Sister ∘ Parent?

When is (x,y) ∈ Parent ∘ Sister?

S □ R = {(a, c) | ∃ b such that (a,b)∈ R and (b,c)∈ S}
Powers of a Relation

\[ R^2 = R \circ R \]
\[ = \{(a, c) : \exists b \text{ such that } (a, b) \in R \text{ and } (b, c) \in R \} \]

\[ R^0 = \{(a, a) : a \in A\} \quad \text{“the equality relation on } A\text{”} \]

\[ R^{n+1} = R^n \circ R \quad \text{for } n \geq 0 \]

e.g., \[ R^1 = R^0 \circ R = R \]
\[ R^2 = R^1 \circ R = R \circ R \]
Matrix Representation

Relation $R$ on $A = \{a_1, ..., a_n\}$

\[
m_{ij} = \begin{cases} 
1 & \text{if } (a_i, a_j) \in R \\
0 & \text{if } (a_i, a_j) \notin R 
\end{cases}
\]

\{ (1, 1), (1, 2), (1, 4), (2, 1), (2, 3), (3, 2), (3, 3), (4, 2), (4, 3) \}

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
Properties using matrix representation

reflexive

symmetric

anti-symmetric

Same when rows & columns swapped

No 1-1 pairs
Directed Graphs

\[ G = (V, E) \]

- \( V \) – vertices
- \( E \) – edges, ordered pairs of vertices
Directed Graphs

\[ G = (V, E) \]

- \( V \) – vertices
- \( E \) – edges (relation on vertices)

**Path:** \( v_0, v_1, ..., v_k \) with each \((v_i, v_{i+1})\) in \( E \)
Directed Graphs

\[ G = (V, E) \]

- \( V \) – vertices
- \( E \) – edges (relation on vertices)

**Path:** \( v_0, v_1, ..., v_k \) with each \((v_i, v_{i+1})\) in \( E \)

- **Simple Path:** none of \( v_0, ..., v_k \) repeated
- **Cycle:** \( v_0 = v_k \)
- **Simple Cycle:** \( v_0 = v_k \), none of \( v_1, ..., v_k \) repeated
Directed Graphs

\[ G = (V, E) \]

- \( V \) – vertices
- \( E \) – edges

(\( E \) is a relation on \( V \))

**Path:** \( v_0, v_1, ..., v_k \) with each \((v_i, v_{i+1})\) in \( E \)

- **Simple Path:** none of \( v_0, ..., v_k \) repeated
- **Cycle:** \( v_0 = v_k \)
- **Simple Cycle:** \( v_0 = v_k \), none of \( v_1, ..., v_k \) repeated
Directed Graphs

\[ G = (V, E) \]
- \( V \) – vertices
- \( E \) – edges (relation on vertices)

**Path:** \( v_0, v_1, ..., v_k \) with each \((v_i, v_{i+1})\) in \( E \)

**Simple Path:** none of \( v_0, ..., v_k \) repeated
**Cycle:** \( v_0 = v_k \)
**Simple Cycle:** \( v_0 = v_k \), none of \( v_1, ..., v_k \) repeated
Representation of Relations

Directed Graph Representation (Digraph)

\{ (a, b), (a, a), (b, a), (c, a), (c, d), (c, e), (d, e) \}
Directed Graph Representation (Digraph)

\{(a, b), (a, a), (b, a), (c, a), (c, d), (c, e) \ (d, e) \}
If $S = \{(2, 2), (2, 3), (3, 1)\}$ and $R = \{(1, 2), (2, 1), (1, 3)\}$

Compute $S \circ R$
If $S = \{(2, 2), (2, 3), (3, 1)\}$ and $R = \{(1, 2), (2, 1), (1, 3)\}$
Compute $S \circ R$
If \( S = \{(2, 2), (2, 3), (3, 1)\} \) and \( R = \{(1, 2), (2, 1), (1, 3)\} \)
Compute \( S \circ R \)
If \( R = \{(1, 2), (2, 1), (1, 3)\}\) and \( R = \{(1, 2), (2, 1), (1, 3)\}\)

Compute \( R \circ R \)

\[
(a, c) \in R \circ R = R^2 \quad \text{iff} \quad \exists b \ ((a, b) \in R \land (b, c) \in R)
\]

iff \( \exists b \) such that \( a, b, c \) is a path
Relational Composition using Digraphs

If \( R = \{(1, 2), (2, 1), (1, 3)\} \) and \( R = \{(1, 2), (2, 1), (1, 3)\} \)

Compute \( R \circ R \)

\[(a, c) \in R \circ R = R^2 \iff \exists b \ ((a, b) \in R \land (b, c) \in R) \]
\iff \exists b \text{ such that } a, b, c \text{ is a path}
Relational Composition using Digraphs

If $R = \{(1, 2), (2, 1), (1, 3)\}$ and $R = \{(1, 2), (2, 1), (1, 3)\}$

Compute $R \circ R$

Special case: $R \circ R$ is paths of length 2.

- $R$ is paths of length 1
- $R^0$ is paths of length 0 (can’t go anywhere)
- $R^3 = R^2 \circ R$ etc, so is $R^n$ paths of length $n$
Def: The **length** of a path in a graph is the number of edges in it (counting repetitions if edge used > once).

Elements of $R^0$ correspond to paths of length 0.
Elements of $R^1 = R$ are paths of length 1.
Elements of $R^2$ are paths of length 2.

...
Let $R$ be a relation on a set $A$.
There is a path of length $n$ from $a$ to $b$ in the digraph for $R$ if and only if $(a,b) \in R^n$.
Connectivity In Graphs

**Def:** Two vertices in a graph are **connected** iff there is a path between them.

Let $R$ be a relation on a set $A$. The **connectivity** relation $R^*$ consists of the pairs $(a, b)$ such that there is a path from $a$ to $b$ in $R$.

$$R^* = \bigcup_{k=0}^{\infty} R^k$$

Note: The Rosen book uses the wrong definition of this quantity. What the Rosen defines (ignoring $k = 0$) is usually called $R^+$.
How Properties of Relations show up in Graphs

Let $R$ be a relation on $A$.

- ** Reflexive**: $R$ is reflexive iff $(a, a) \in R$ for every $a \in A$.
- ** Symmetric**: $R$ is symmetric iff $(a, b) \in R$ implies $(b, a) \in R$.
- ** Antisymmetric**: $R$ is antisymmetric iff $(a, b) \in R$ and $a \neq b$ implies $(b, a) \notin R$.
- ** Transitive**: $R$ is transitive iff $(a, b) \in R$ and $(b, c) \in R$ implies $(a, c) \in R$. 
How Properties of Relations show up in Graphs

Let R be a relation on A.

R is **reflexive** iff \((a,a) \in R\) for every \(a \in A\)

- at every node

R is **symmetric** iff \((a,b) \in R\) implies \((b,a) \in R\)

- or

R is **antisymmetric** iff \((a,b) \in R\) and \(a \neq b\) implies \((b,a) \notin R\)

- or

R is **transitive** iff \((a,b) \in R\) and \((b,c) \in R\) implies \((a,c) \in R\)
Add the **minimum possible** number of edges to make the relation transitive and reflexive.
Relation with the **minimum possible** number of **extra edges** to make the relation both transitive and reflexive.

The **transitive-reflexive closure** of a relation $R$ is the connectivity relation $R^*$. 

Transitive-Reflexive Closure
Let $A_1, A_2, \ldots, A_n$ be sets. An $n$-ary relation on these sets is a subset of $A_1 \times A_2 \times \cdots \times A_n$. 
<table>
<thead>
<tr>
<th>Student_Name</th>
<th>ID_Number</th>
<th>Office</th>
<th>GPA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Knuth</td>
<td>328012098</td>
<td>022</td>
<td>4.00</td>
</tr>
<tr>
<td>Von Neuman</td>
<td>481080220</td>
<td>555</td>
<td>3.78</td>
</tr>
<tr>
<td>Russell</td>
<td>238082388</td>
<td>022</td>
<td>3.85</td>
</tr>
<tr>
<td>Einstein</td>
<td>238001920</td>
<td>022</td>
<td>2.11</td>
</tr>
<tr>
<td>Newton</td>
<td>1727017</td>
<td>333</td>
<td>3.61</td>
</tr>
<tr>
<td>Karp</td>
<td>348882811</td>
<td>022</td>
<td>3.98</td>
</tr>
<tr>
<td>Bernoulli</td>
<td>2921938</td>
<td>022</td>
<td>3.21</td>
</tr>
</tbody>
</table>
AND NOW BACK TO OUR REGULARLY SCHEDULED PROGRAMMING
Selecting strings using labeled graphs as “machines”
Finite State Machines

The circles are called “states”
We’re only in a single state at any point in time...

The “double circle” means “the input is good if it ends here”

“If I get this symbol, follow the arrow…”

“Start here”
Which strings does this machine say are OK?
Which strings does this machine say are OK?

The set of all binary strings that end in 0
Finite State Machines

- States
- Transitions on input symbols
- Start state and final states
- The “language recognized” by the machine is the set of strings that reach a final state from the start

<table>
<thead>
<tr>
<th>Old State</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s_0$</td>
<td>$s_0$</td>
<td>$s_1$</td>
</tr>
<tr>
<td>$s_1$</td>
<td>$s_0$</td>
<td>$s_2$</td>
</tr>
<tr>
<td>$s_2$</td>
<td>$s_0$</td>
<td>$s_3$</td>
</tr>
<tr>
<td>$s_3$</td>
<td>$s_3$</td>
<td>$s_3$</td>
</tr>
</tbody>
</table>
Finite State Machines

- Each machine designed for strings over some fixed alphabet $\Sigma$.

- Must have a transition defined from each state for every symbol in $\Sigma$.

<table>
<thead>
<tr>
<th>Old State</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s_0$</td>
<td>$s_0$</td>
<td>$s_1$</td>
</tr>
<tr>
<td>$s_1$</td>
<td>$s_0$</td>
<td>$s_2$</td>
</tr>
<tr>
<td>$s_2$</td>
<td>$s_0$</td>
<td>$s_3$</td>
</tr>
<tr>
<td>$s_3$</td>
<td>$s_3$</td>
<td>$s_3$</td>
</tr>
</tbody>
</table>
What language does this machine recognize?

<table>
<thead>
<tr>
<th>Old State</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>(s_0)</td>
<td>(s_0)</td>
<td>(s_1)</td>
</tr>
<tr>
<td>(s_1)</td>
<td>(s_0)</td>
<td>(s_2)</td>
</tr>
<tr>
<td>(s_2)</td>
<td>(s_0)</td>
<td>(s_3)</td>
</tr>
<tr>
<td>(s_3)</td>
<td>(s_3)</td>
<td>(s_3)</td>
</tr>
</tbody>
</table>
What language does this machine recognize?

The set of all binary strings that contain 111 or don’t end in 1

<table>
<thead>
<tr>
<th>Old State</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s_0$</td>
<td>$s_0$</td>
<td>$s_1$</td>
</tr>
<tr>
<td>$s_1$</td>
<td>$s_0$</td>
<td>$s_2$</td>
</tr>
<tr>
<td>$s_2$</td>
<td>$s_0$</td>
<td>$s_3$</td>
</tr>
<tr>
<td>$s_3$</td>
<td>$s_3$</td>
<td>$s_3$</td>
</tr>
</tbody>
</table>