Section 4.2: Equivalence Relations

What is "equality"?
What is "equivalence"?

Equality is more basic, fundamental concept.

Every element in a set is equal only to itself.
The basic equality relation

$\{(x,x) \mid x \in S\}$

We tend to assume equality is implicitly understood and agreed upon.

Broader Issues:

Is 2+2=4?

No: "2+2" has 3 characters, while "4" has 1 character

Yes: There is an underlying set, namely R and these strings refer to the same object in R.

So 2+2 is "equivalent" to 4.   (Use algebra to show the equivalence.)

We require three properties of any notion of equality:

Reflexive	$x=x$
Symmetric	If $x=y$ then $y=x$
Transitive	If $x=y$ and $y=z$ then $x=z$

Notation:

$=$	Equality
$\sim$	Equivalence

Any equivalence relation should behave
the same as we expect equality to behave.

Reflexive	$x\sim x$
Symmetric	If $x\sim y$ then $y\sim x$
Transitive	If $x\sim y$ and $y\sim z$ then $x\sim z$

Equivalence Relations

A binary relation is an equivalence relation iff it has these 3 properties:

Reflexive	x~x
Symmetric	If x~y then y~x
Transitive	If x~y and y~z then x~z

"RST"

Note: When taking the reflex.,sym. & trans. closures, write tsr(R)

Examples:

Equality on any set

x ~ y iff |x| = |y| over the set of strngs {a,b,c}*

x ~ y iff x and y have the same birthday over the set of people

Another Example:

Consider the set of all arithmetic expressions, such as:

4x+2

The relation $e_1 \sim e_2$ holds iff e1 and e2 have the same value (for any assignment to the variables)

So:

4x+2 ~ 2(2x+1)

<u>Quiz:</u> Which of these relations are RST?

x R y iff $x \le y$ or $x > y$ over Z

x R y iff $|x-y| \le 2$ over Z

x R y iff x and y are both even over Z

Quiz: Which of these relations are RST?

x R y iff x≤y or x>y over Z
  everything is related to everything else
  reflexive, symmetric & transitive → equivalence.

x R y iff |x-y|≤2 over Z

x R y iff x and y are both even over Z

Quiz: Which of these relations are RST?

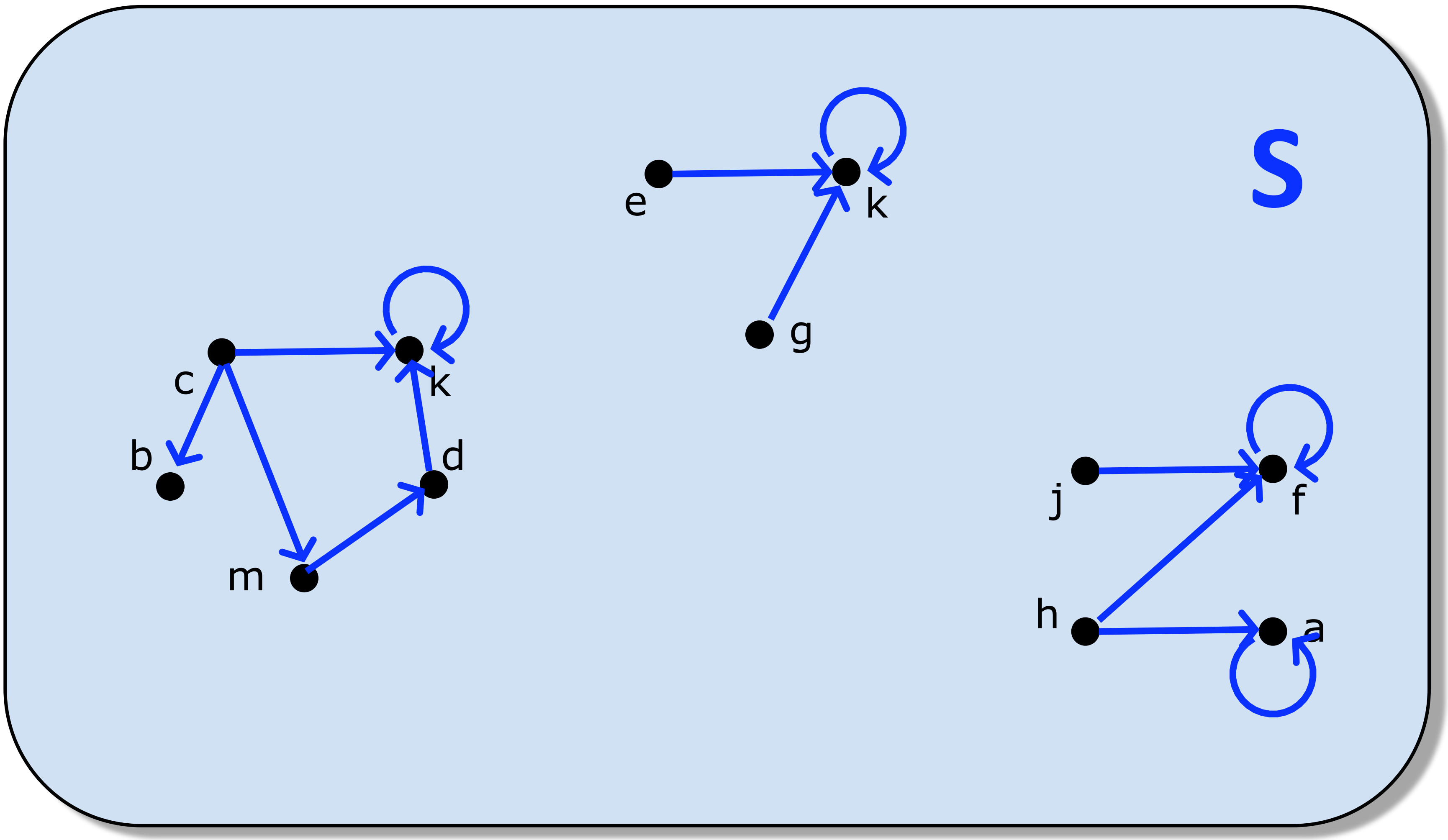
x R y iff x≤y or x>y over Z
 everything is related to everything else
 reflexive, symmetric & transitive → equivalence.
x R y iff |x-y|≤2 over Z
 3~5 and 5~7 but not 3~7 → not transitive
x R y iff x and y are both even over Z

Quiz: Which of these relations are RST?

- x R y iff x≤y or x>y over Z
  - everything is related to everything else
  - reflexive, symmetric & transitive → equivalence.
- x R y iff |x-y|≤2 over Z
  - 3~5 and 5~7 but not 3~7 → not transitive
- x R y iff x and y are both even over Z
  - 7 is not related to 7 --> not reflexive

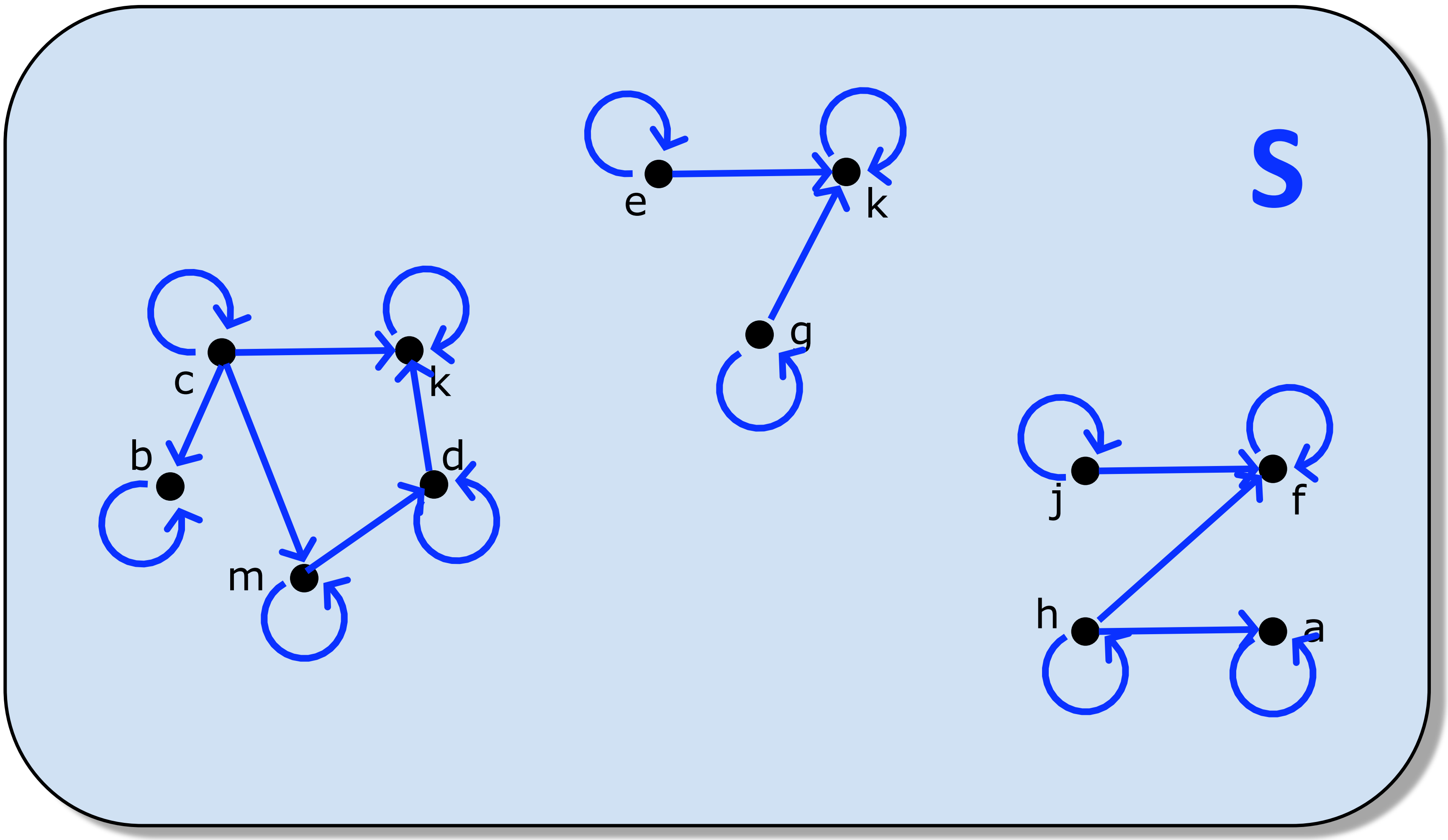
Equivalence Relations – RST

Reflexive
Symmetric
Transitive



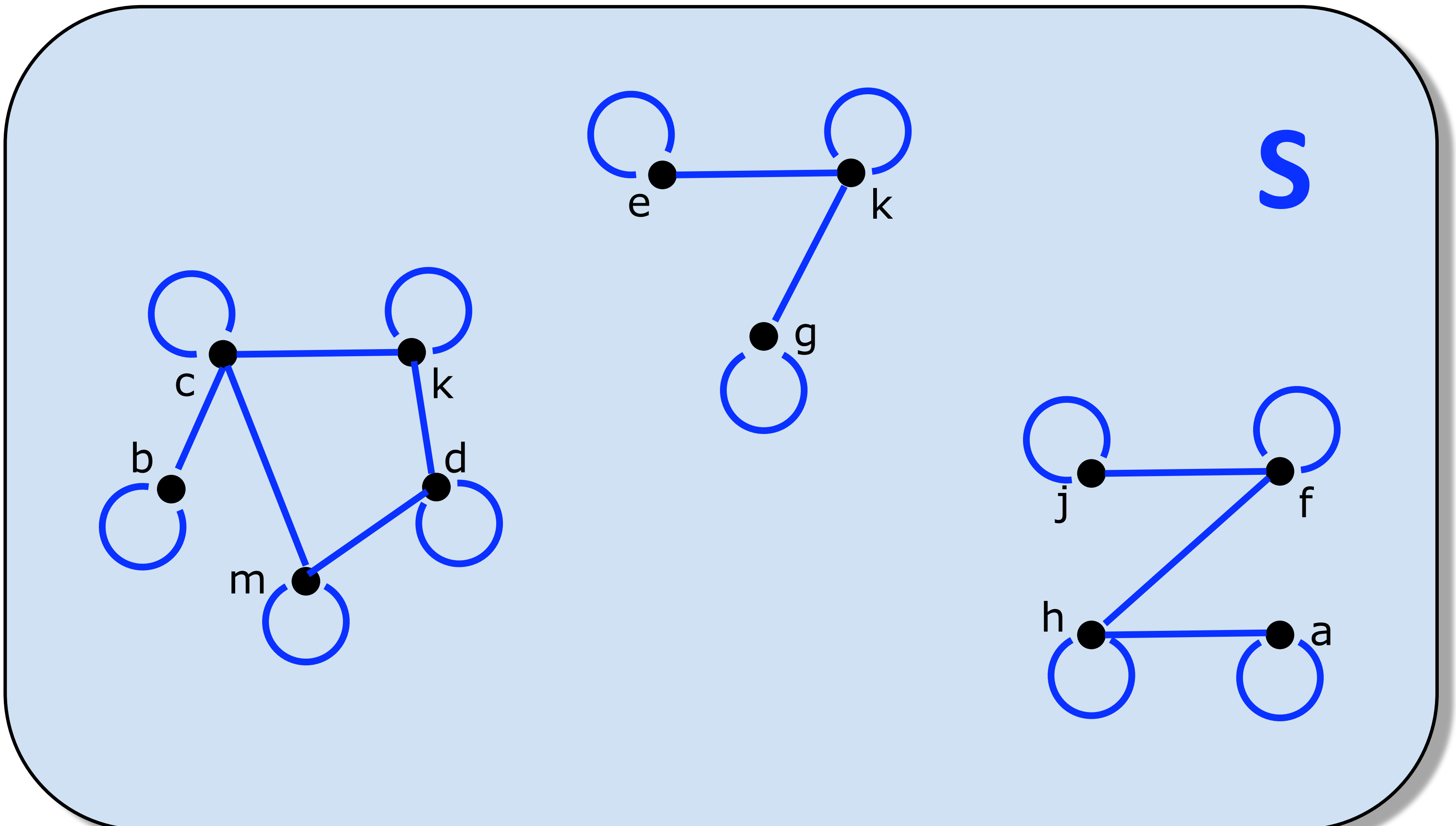
Equivalence Relations – RST

Reflexive
Symmetric
Transitive



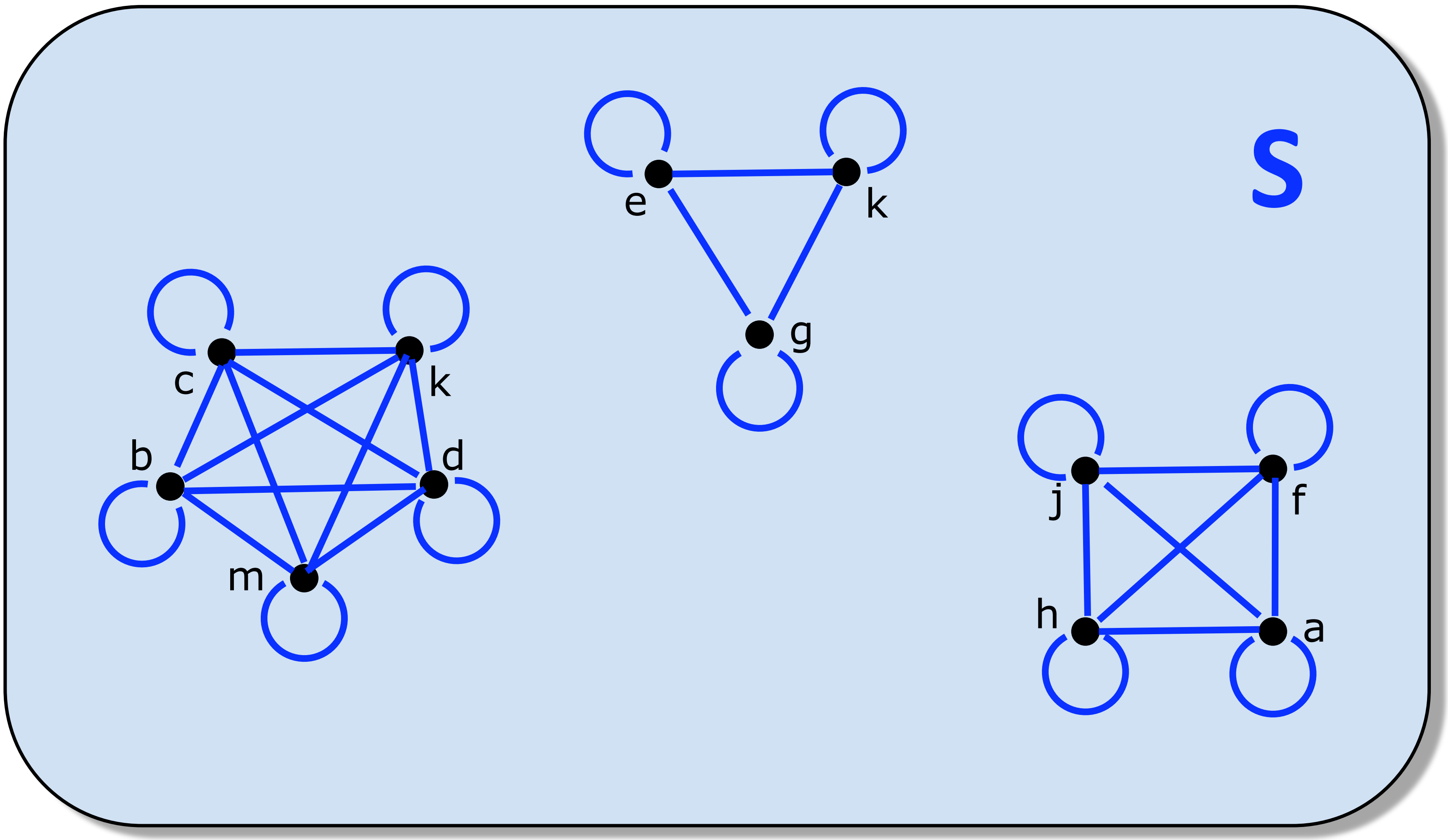
Equivalence Relations – RST

Reflexive
Symmetric
Transitive



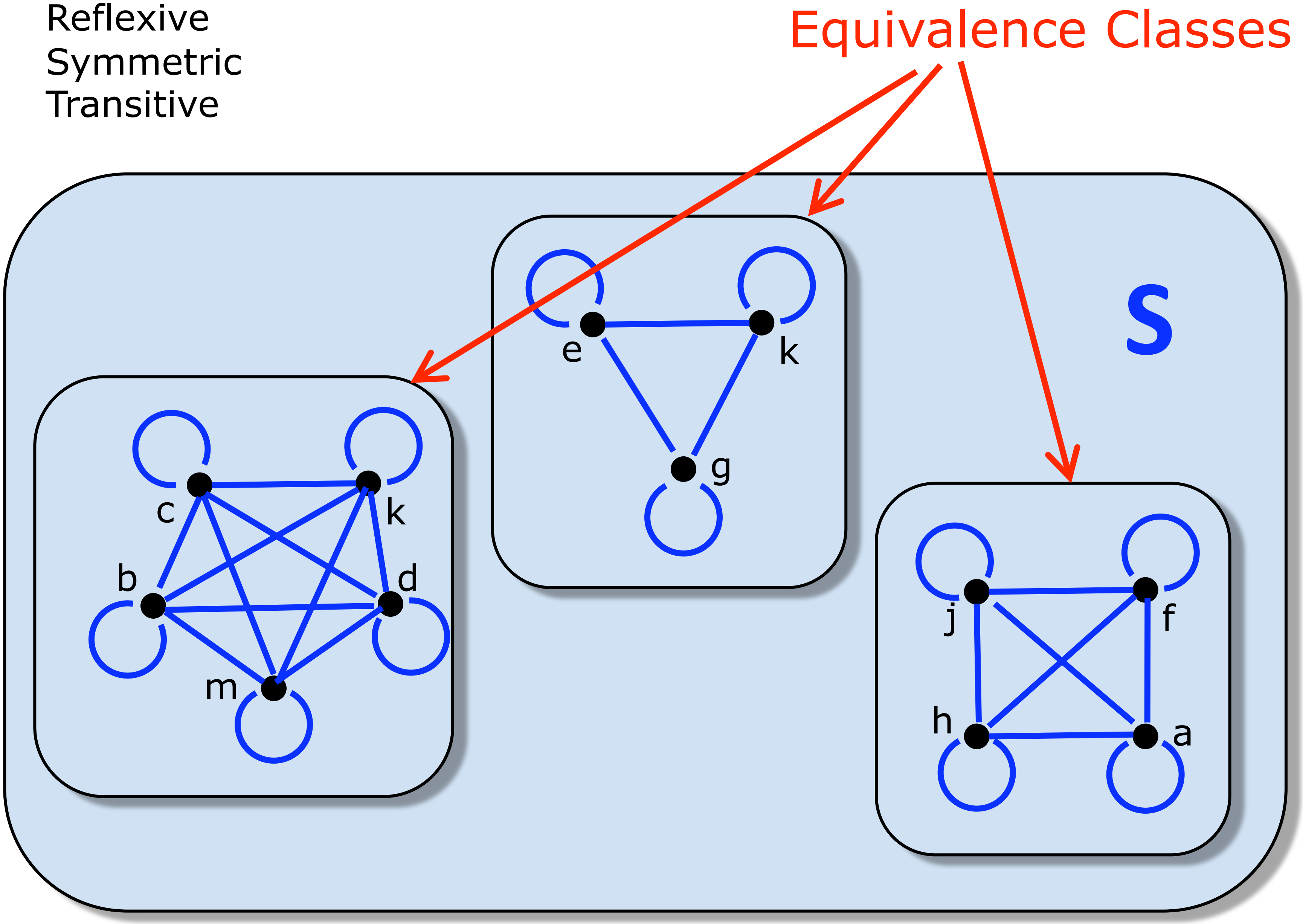
Equivalence Relations – RST

Reflexive
Symmetric
Transitive



Equivalence Relations – RST

Reflexive
Symmetric
Transitive



Partitions

A partition of a set S is a collection of (nonempty) disjoint subsets whose union is S.

Equivalence Classes

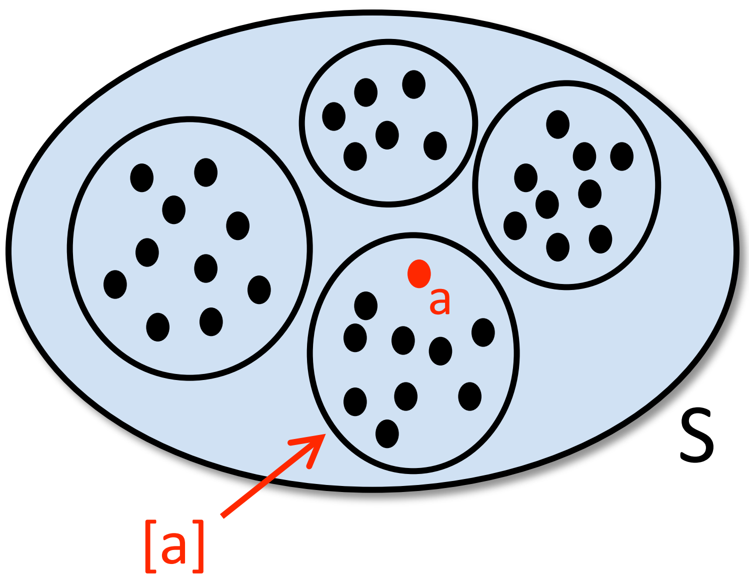
If R is RST over A, then for each $a \in A$, the equivalence class of a is denoted [a] and is defined as the set of things equivalent to a:

$$[a] = \{ x \mid x\,R\,a \}$$

Theorem

Let A be a set…

- The equivalence classes of any RST relation over A form a partition of A.
- Any partition of A yields an RST over A, where the sets of the partition act as the equivalence classes.



Partitions

A partition of a set S is a collection of (nonempty) disjoint subsets whose union is S.

Equivalence Classes

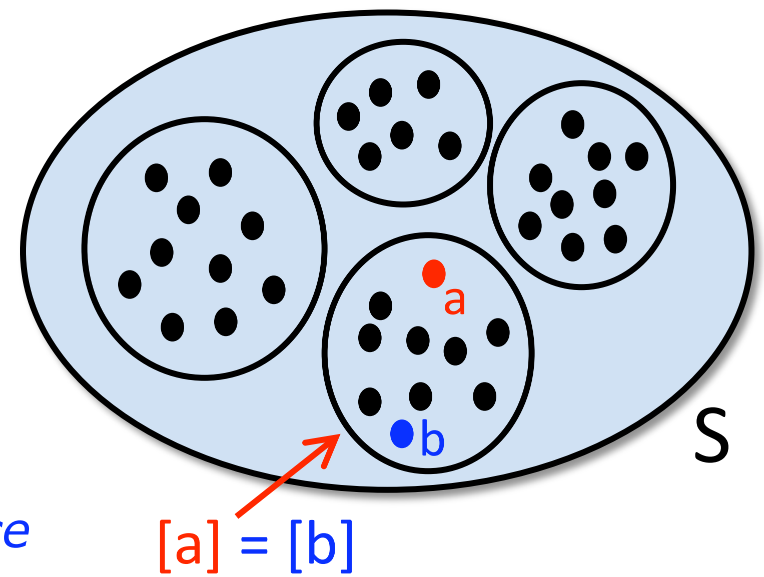
If R is RST over A, then for each a∈A, the equivalence class of a is denoted [a] and is defined as the set of things equivalent to a:

$[a] = \{ x \mid x \, R \, a \}$

Theorem

Let A be a set…

- The equivalence classes of any RST relation over A form a partition of A.
- Any partition of A yields an RST over A, where the sets of the partition act as the equivalence classes.

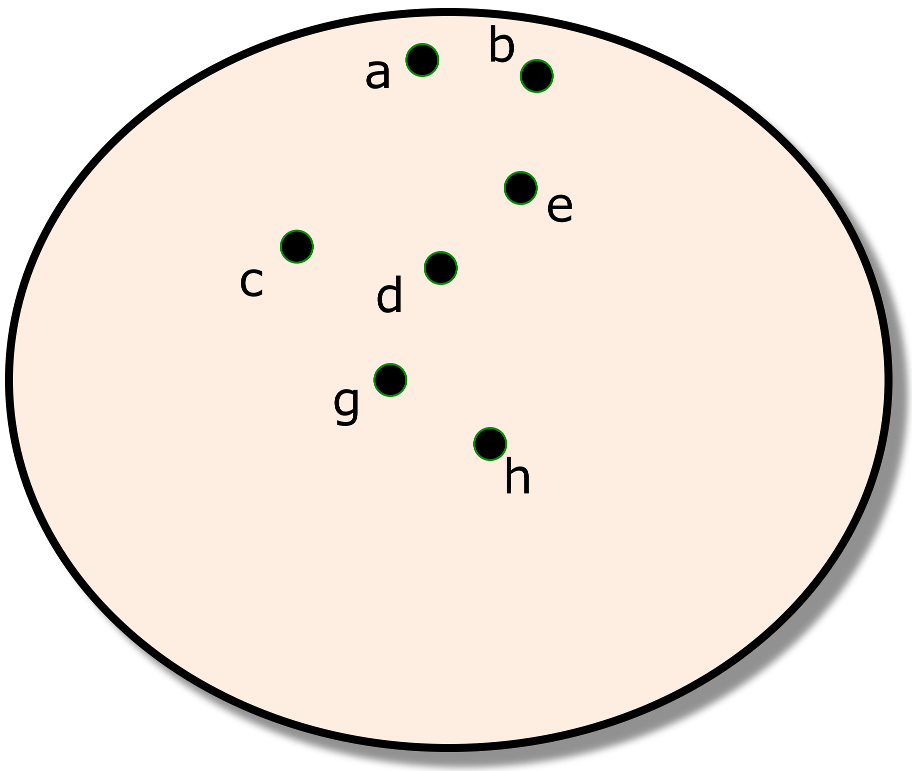


You can use any member of an equivalence class as its representative.

[a] = [b]

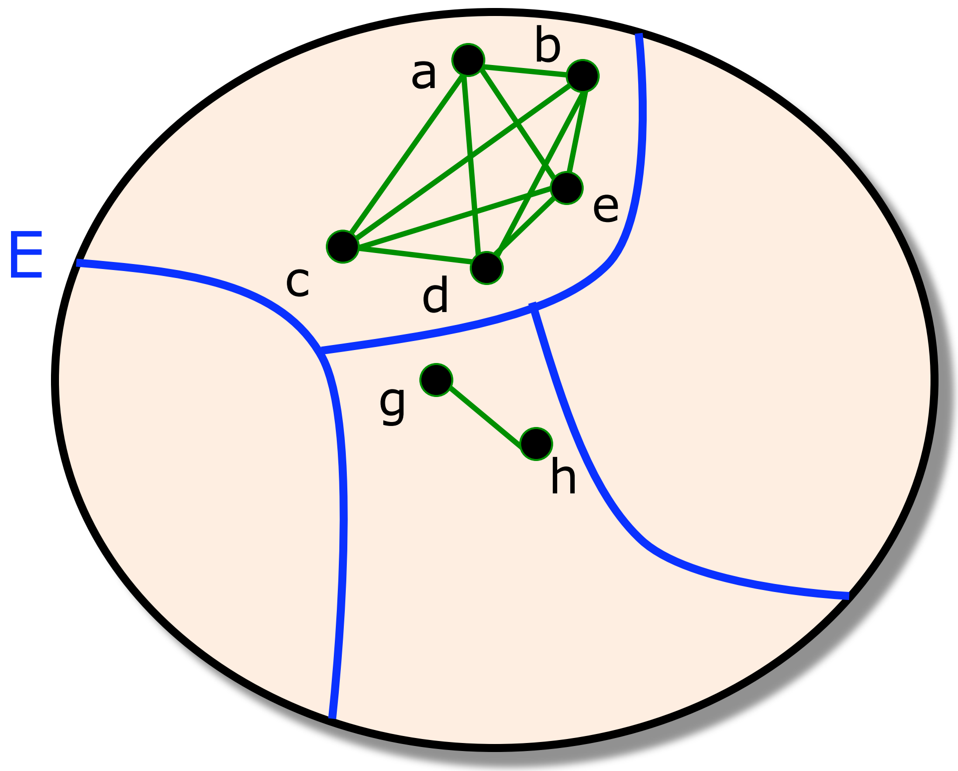
Intersection Property

If E and F are two equivalence relations over A
(i.e., E and F are RST)…
then $E \cap F$ is also an equivalence relation
(i.e., is also RST).



Intersection Property

If E and F are two equivalence relations over A
(i.e., E and F are RST)…
then E ∩ F is also an equivalence relation
(i.e., is also RST).

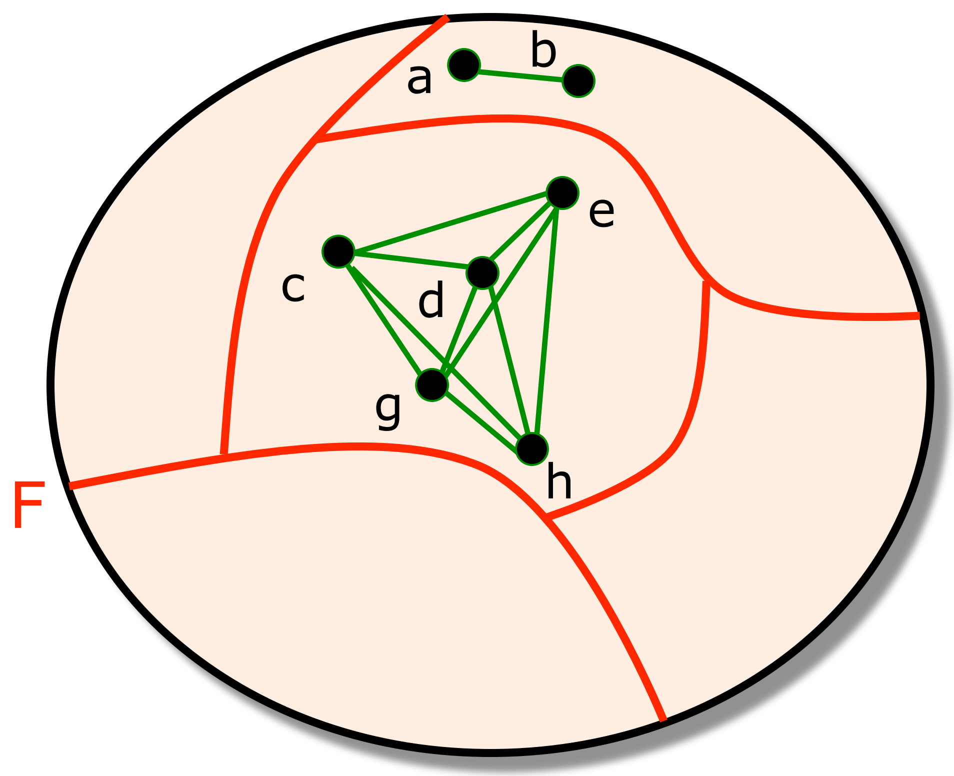


a~b
a~c
a~d
a~e
b~c
b~d
b~e
c~d
c~e
d~e

g~h

Intersection Property

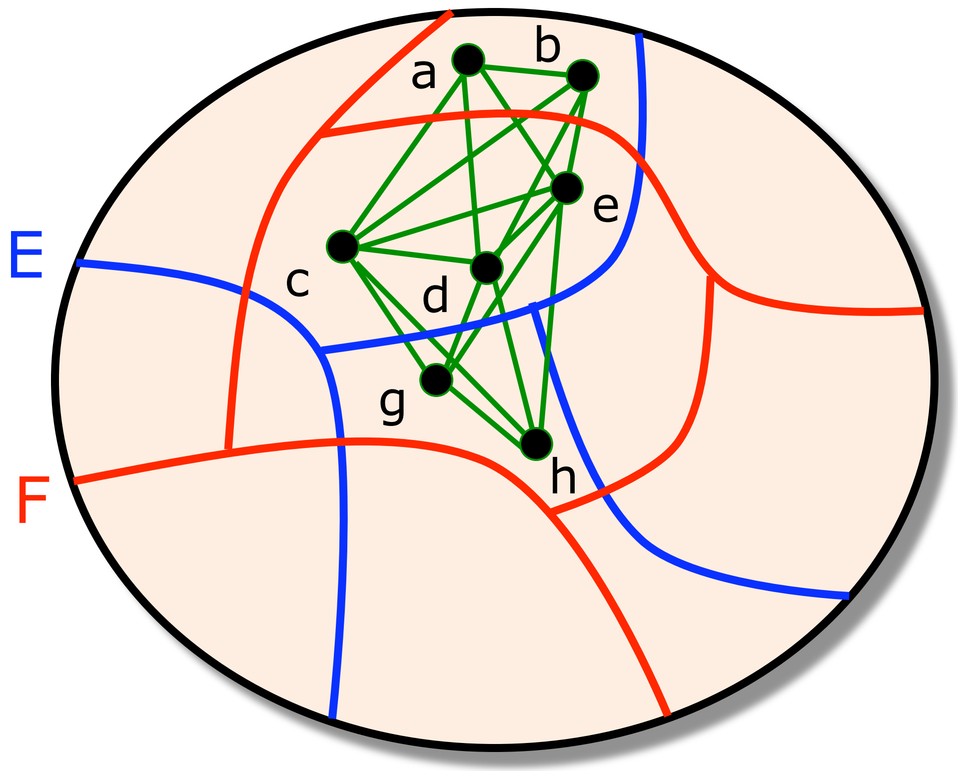
If E and F are two equivalence relations over A
(i.e., E and F are RST)…
then E ∩ F is also an equivalence relation
(i.e., is also RST).



a~b	a~b
a~c	
a~d	
a~e	
b~c	
b~d	
b~e	
c~d	c~d
c~e	c~e
d~e	d~e
	c~g
	c~h
	d~g
	d~h
	e~g
	e~h
g~h	g~h

Intersection Property

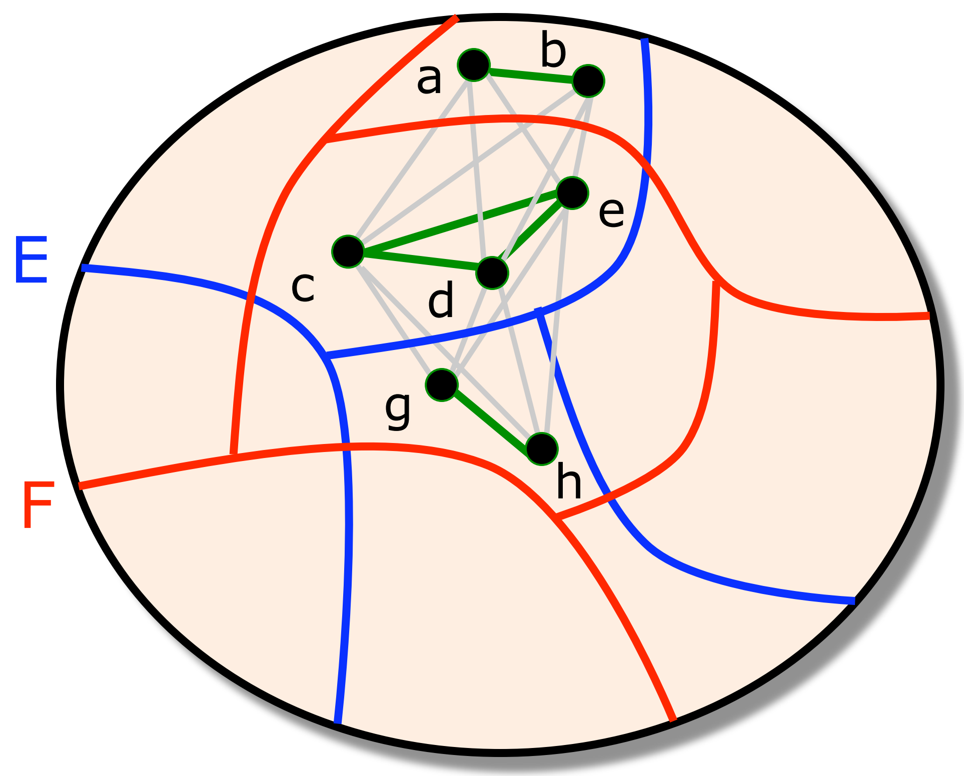
If E and F are two equivalence relations over A
(i.e., E and F are RST)…
then E ∩ F is also an equivalence relation
(i.e., is also RST).



a~b	a~b
a~c	
a~d	
a~e	
b~c	
b~d	
b~e	
c~d	c~d
c~e	c~e
d~e	d~e
	c~g
	c~h
	d~g
	d~h
	e~g
	e~h
g~h	g~h

Intersection Property

If E and F are two equivalence relations over A
(i.e., E and F are RST)…
then E $\cap$ F is also an equivalence relation
(i.e., is also RST).



a~b	a~b
a~c	
a~d	
a~e	
b~c	
b~d	
b~e	
c~d	c~d
c~e	c~e
d~e	d~e
	c~g
	c~h
	d~g
	d~h
	e~g
	e~h
g~h	g~h

Example:

"has same birthday as" is an equivalence relation

All people born on June 1 is an equivalence class

"has the same first name" is an equivalence relation

All people named Fred is an equivalence class

Let x~y iff

x and y have the same birthday <u>and</u>

x and y have the same first name

This relation must be an equivalence relation.

It is the intersection of two equivalence relations.

One class contains all people named Fred who were also born June 1.

Kernel Relations

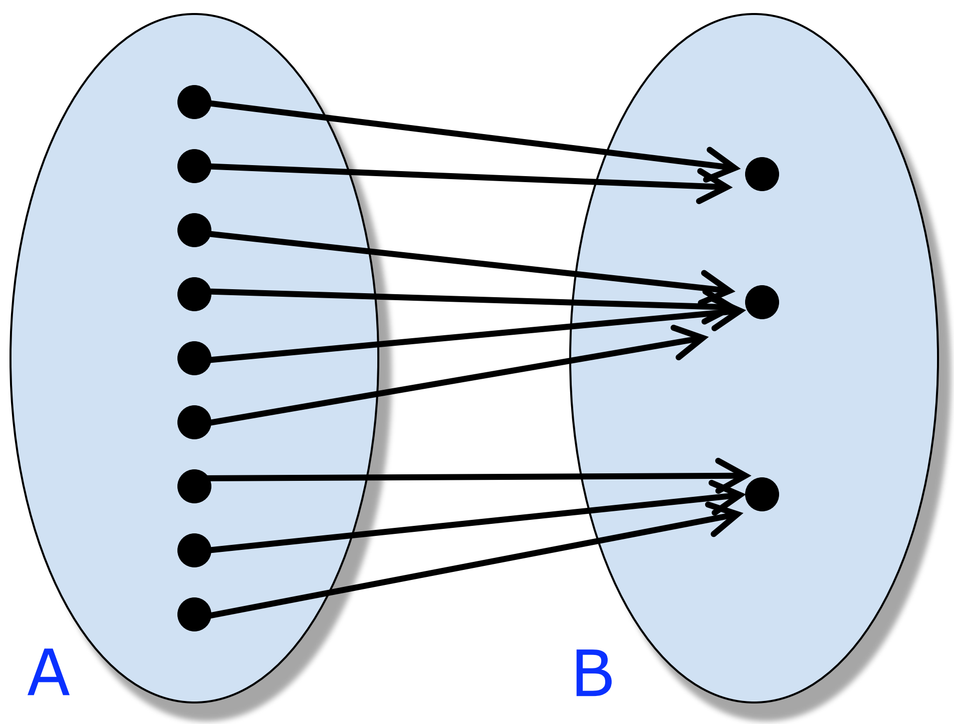
Assume we have a function

$f: A \rightarrow B$

Define a relation on set A by letting

$x \sim y$ iff $f(x)=f(y)$

This is a "kernel relation" and it will be RST: an equivalence relation!



Kernel Relations

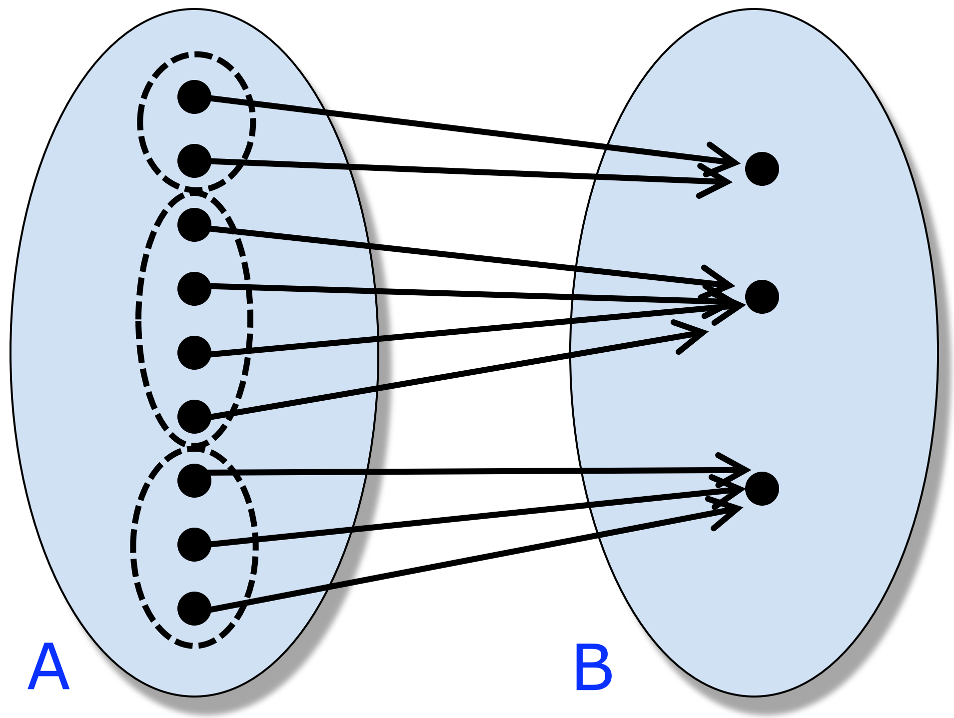
Assume we have a function

    f: A → B

Define a relation on set A by letting

    x ~ y iff f(x)=f(y)

This is a "kernel relation" and it will be RST: an equivalence relation!



Kernel Relations

Example:

Let x~y iff x mod n = y mod n, over any set of integers.

Then ~ is an equivalence relation because it is the kernel relation of function $f: S \to N$ defined by $f(x) = x \bmod n$.

Example:

Let x~y iff x+y is even over Z.

Note that x+y is even

iff x and y are both even or both odd

iff x mod 2 = y mod 2.

Therefore ~ is an equivalence relation

because ~ is the kernel relation of

the function $f: Z \to N$ defined by $f(x) = x \bmod 2$.

Equivalence Classes

Property:

For every pair $a,b \in A$ we must have either:

$[a] = [b]$ or

$[a] \cap [b] = \emptyset$

Example:

Suppose $x \sim y$ iff $x \bmod 3 = y \bmod 3$, over the set N.

The equivalence classes are:

$[0] = \{0,3,6,\ldots\} = \{3k \mid k \in N\}$

$[1] = \{1,4,7,\ldots\} = \{3k+1 \mid k \in N\}$

$[2] = \{2,5,8,\ldots\} = \{3k+2 \mid k \in N\}$

Notice that $[0] = [3] = [6]$.

Notice that $[1] \cap [2] = \emptyset$.

Example:

Suppose $x \sim y$ iff $x \bmod 2 = y \bmod 2$, over the integers Z.

Then $\sim$ is an equivalence relation with equivalence classes

$[0]$=evens, and $[1]$=odds.

Note that $\{[0],[1]\}$ is a partition of Z.

Equivalence Classes

Example:

The set of real numbers $\mathbb{R}$ can be partitioned into the set of half-open intervals $\{(n,n+1] \mid n \in \mathbb{Z}\}$.

… (0,1], (1,2], (2,3], …

Then we have an RST ~ over $\mathbb{R}$, where $x \sim y$ iff $x,y \in (n,n+1]$, for some $n \in \mathbb{Z}$.

Quiz:

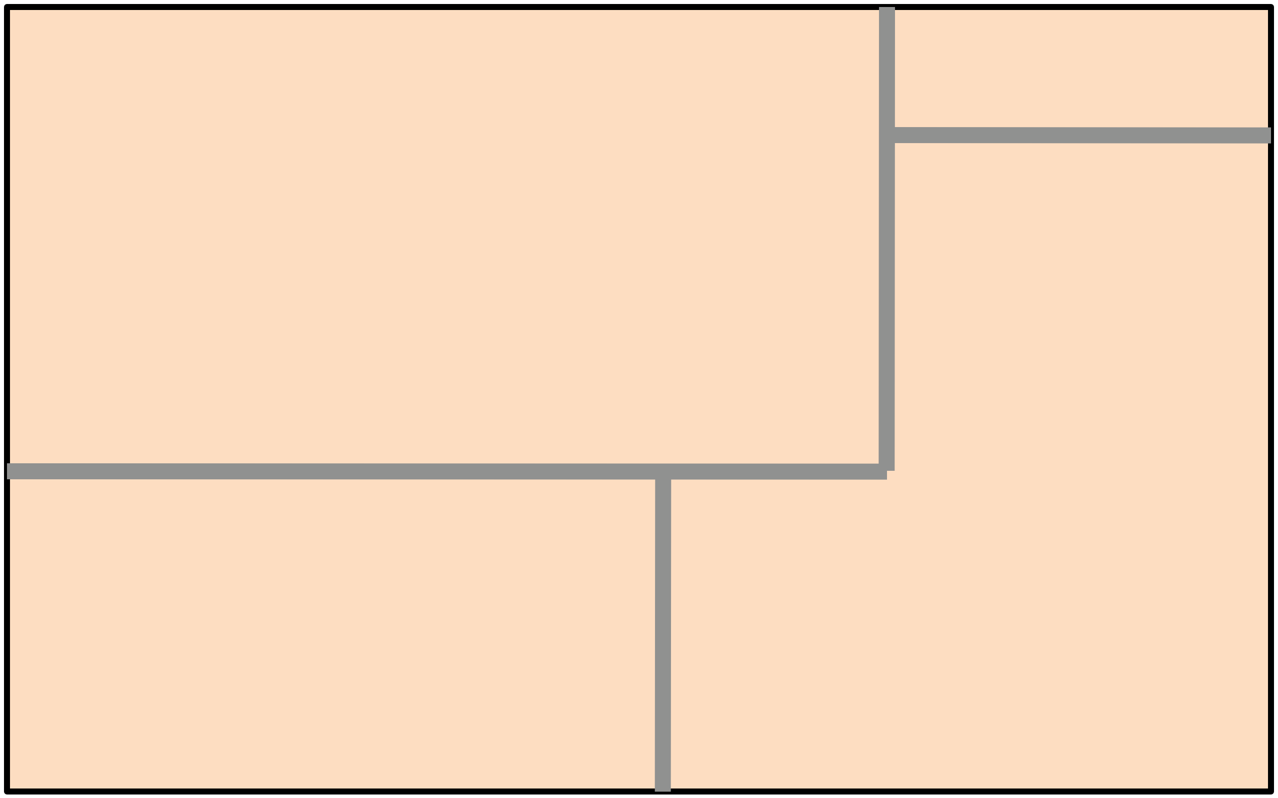
In the preceding example, what is another way to say $x \sim y$?

Answer:

$x \sim y$ iff $\lceil x \rceil = \lceil y \rceil$

Refining Partitions

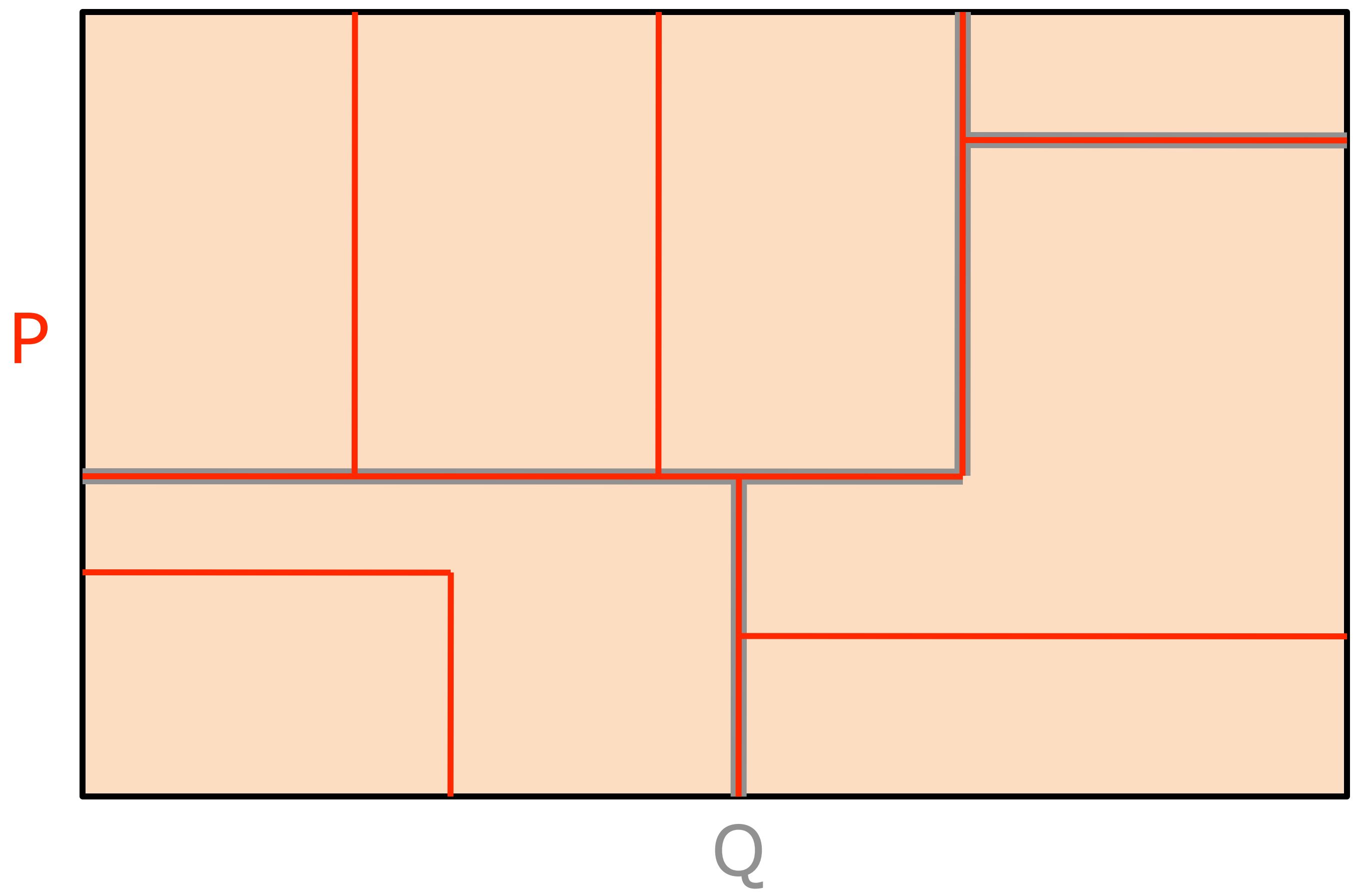
If P and Q are partitions of a set S, then P is a "refinement" of Q if every $A \in P$ is a subset of some $B \in Q$.



Q

Refining Partitions

If P and Q are partitions of a set S, then P is a "refinement" of Q if every $A \in P$ is a subset of some $B \in Q$.



Refining Partitions

Example: Let S = {a,b,c,d,e} and consider the following four partitions of S.

P_1 = { {a,b,c,d,e} }
P_2 = { {a,b}, {c,d,e} }
P_3 = { {a}, {b}, {c}, {d,e} }
P_4 = { {a}, {b}, {c}, {d}, {e} }

Each P_i is a refinement of the previous one.
We can talk about "courser" and "finer" refinements.
P_1 is the "coursest" and P_4 is the "finest" refinement.

Example: Let ~3 and ~6 be the following equivalence relations over $\mathbb{N}$:

$x \sim_3 y$ iff x mod 3 = y mod 3

This relation has the following equivalence classes:

$[0]_3 = \{0,3,6,9,12\ldots\} = \{3k \mid k \in \mathbb{N}\}$

$[1]_3 = \{1,4,7,10,13\ldots\} = \{3k+1 \mid k \in \mathbb{N}\}$

$[2]_3 = \{2,5,8,11,14\ldots\} = \{3k+2 \mid k \in \mathbb{N}\}$

$x \sim_6 y$ iff x mod 6 = y mod 6

This relation has the following equivalence classes:

$[0]_6 = \{0,6,12,\ldots\} = \{6k \mid k \in \mathbb{N}\}$

$[1]_6 = \{1,7,13,\ldots\} = \{6k+1 \mid k \in \mathbb{N}\}$

$[2]_6 = \{2,8,14,\ldots\} = \{6k+2 \mid k \in \mathbb{N}\}$

$[3]_6 = \{3,9,15,\ldots\} = \{6k+3 \mid k \in \mathbb{N}\}$

$[4]_6 = \{4,10,16,\ldots\} = \{6k+4 \mid k \in \mathbb{N}\}$

$[5]_6 = \{5,11,17,\ldots\} = \{6k+5 \mid k \in \mathbb{N}\}$

Example: Let ~3 and ~6 be the following equivalence relations over $\mathbb{N}$:

$x \sim_3 y$ iff x mod 3 = y mod 3

This relation has the following equivalence classes:

$[0]_3 = \{0,3,6,9,12\ldots\} = \{3k \mid k\in\mathbb{N}\}$

$[1]_3 = \{1,4,7,10,13\ldots\} = \{3k+1 \mid k\in\mathbb{N}\}$

$[2]_3 = \{2,5,8,11,14\ldots\} = \{3k+2 \mid k\in\mathbb{N}\}$

$x \sim_6 y$ iff x mod 6 = y mod 6

This relation has the following equivalence classes:

$[0]_6 = \{0,6,12,\ldots\} = \{6k \mid k\in\mathbb{N}\} \quad \subseteq [0]_3$

$[1]_6 = \{1,7,13,\ldots\} = \{6k+1 \mid k\in\mathbb{N}\} \quad \subseteq [1]_3$

$[2]_6 = \{2,8,14,\ldots\} = \{6k+2 \mid k\in\mathbb{N}\} \quad \subseteq [2]_3$

$[3]_6 = \{3,9,15,\ldots\} = \{6k+3 \mid k\in\mathbb{N}\} \quad \subseteq [0]_3$

$[4]_6 = \{4,10,16,\ldots\} = \{6k+4 \mid k\in\mathbb{N}\} \quad \subseteq [1]_3$

$[5]_6 = \{5,11,17,\ldots\} = \{6k+5 \mid k\in\mathbb{N}\} \quad \subseteq [2]_3$

The partition $\sim_6$ is a refinement of $\sim_3$.

Quiz:

Consider the equivalence relations $\sim_2$ and $\sim_3$.
Is either a refinement of the other?

Answer:

$[0]_2$ = set of even numbers = $\{0, 2, 4, 6, 8, \ldots\}$
$[1]_2$ = set of odd numbers = $\{1, 3, 5, 7, 9, \ldots\}$

$[0]_3 = \{0, 3, 6, 9, \ldots\}$
$[1]_3 = \{1, \ldots\}$
$[2]_3 = \{2, \ldots\}$

There is no subset relation between $[0]_3$ and either $[0]_2$ or $[1]_2$.

No.

Theorem: (The intersection property of RSTs)

If E and F are RSTs over A, then the equivalence classes of $E \cap F$ have the form

$$[x]_{E\cap F} = [x]_E \cap [x]_F, \text{ where } x \in A$$

Example: Let $\sim_1$ and $\sim_2$ be the following RSTs over $\mathbb{N}$:

$x \sim_1 y$ iff $\lfloor x/4 \rfloor = \lfloor y/4 \rfloor$

$x \sim_2 y$ iff $\lfloor x/6 \rfloor = \lfloor y/6 \rfloor$

Now define a new RST as $R = \sim_1 \cap \sim_2$
What do the equivalence classes of R look like?

$[12n]_R = \{ 12n, 12n+1, 12n+2, 12n+3 \}$
$[12n+4]_R = \{ 12n+4, 12+5 \}$
$[12n+6]_R = \{ 12n+6, 12n+7 \}$
$[12n+8]_R = \{ 12n+8, 12n+9, 12n+10, 12n+11 \}$

Example: Let $\sim_1$ and $\sim_2$ be the following RSTs over $\mathbb{N}$:

$x \sim_1 y$ iff $\lfloor x/4 \rfloor = \lfloor y/4 \rfloor$

$[0]_1 = \{0,1,2,3\}$ $[4]_1 = \{4,5,6,7\}$ etc…

$x \sim_2 y$ iff $\lfloor x/6 \rfloor = \lfloor y/6 \rfloor$

$[0]_2 = \{0,1,2,3,4,5\}$ $[6]_2 = \{6,7,8,9,10,11\}$ etc…

Now define a new RST as $R = \sim_1 \cap \sim_2$

What do the equivalence classes of R look like?

$[12n]_R = \{ 12n, 12n+1, 12n+2, 12n+3 \}$
$[12n+4]_R = \{ 12n+4, 12+5 \}$
$[12n+6]_R = \{ 12n+6, 12n+7 \}$
$[12n+8]_R = \{ 12n+8, 12n+9, 12n+10, 12n+11 \}$

Example: Let $\sim_1$ and $\sim_2$ be the following RSTs over $\mathbb{N}$:

$x \sim_1 y$ iff $\lfloor x/4 \rfloor = \lfloor y/4 \rfloor$ The equivalence classes are:

$[4n]_1 = \{4n, 4n+1, 4n+2, 4n+3\}$

$[0]_1 = \{0,1,2,3\}$ $[4]_1 = \{4,5,6,7\}$ etc…

$x \sim_2 y$ iff $\lfloor x/6 \rfloor = \lfloor y/6 \rfloor$ The equivalence classes are:

$[6n]_2 = \{6n, 6n+1, 6n+2, 6n+3, 6n+4, 6n+5\}$

$[0]_2 = \{0,1,2,3,4,5\}$ $[6]_2 = \{6,7,8,9,10,11\}$ etc…

Now define a new RST as $R = \sim_1 \cap \sim_2$

What do the equivalence classes of R look like?

$[12n]_R = \{ 12n, 12n+1, 12n+2, 12n+3 \}$

$[12n+4]_R = \{ 12n+4, 12+5 \}$

$[12n+6]_R = \{ 12n+6, 12n+7 \}$

$[12n+8]_R = \{ 12n+8, 12n+9, 12n+10, 12n+11 \}$

Example: Let $\sim_1$ and $\sim_2$ be the following RSTs over $\mathbb{N}$:

$x \sim_1 y$ iff $\lfloor x/4 \rfloor = \lfloor y/4 \rfloor$ The equivalence classes are:

$[4n]_1 = \{4n, 4n+1, 4n+2, 4n+3\}$

$[0]_1 = \{0,1,2,3\}$ $[4]_1 = \{4,5,6,7\}$ etc…

$x \sim_2 y$ iff $\lfloor x/6 \rfloor = \lfloor y/6 \rfloor$ The equivalence classes are:

$[6n]_2 = \{6n, 6n+1, 6n+2, 6n+3, 6n+4, 6n+5\}$

$[0]_2 = \{0,1,2,3,4,5\}$ $[6]_2 = \{6,7,8,9,10,11\}$ etc…

Now define a new RST as $R = \sim_1 \cap \sim_2$

What do the equivalence classes of R look like?

$[0]_R = [0]_1 \cap [0]_2 = \{0,1,2,3\} \cap \{0,1,2,3,4,5\} = \{0,1,2,3\}$

$[12n]_R = \{ 12n, 12n+1, 12n+2, 12n+3 \}$

$[12n+4]_R = \{ 12n+4, 12+5 \}$

$[12n+6]_R = \{ 12n+6, 12n+7 \}$

$[12n+8]_R = \{ 12n+8, 12n+9, 12n+10, 12n+11 \}$

Example: Let $\sim_1$ and $\sim_2$ be the following RSTs over $\mathbb{N}$:

$x \sim_1 y$ iff $\lfloor x/4 \rfloor = \lfloor y/4 \rfloor$ The equivalence classes are:

$[4n]_1 = \{4n, 4n+1, 4n+2, 4n+3\}$

$[0]_1 = \{0,1,2,3\}$ $[4]_1 = \{4,5,6,7\}$ etc…

$x \sim_2 y$ iff $\lfloor x/6 \rfloor = \lfloor y/6 \rfloor$ The equivalence classes are:

$[6n]_2 = \{6n, 6n+1, 6n+2, 6n+3, 6n+4, 6n+5\}$

$[0]_2 = \{0,1,2,3,4,5\}$ $[6]_2 = \{6,7,8,9,10,11\}$ etc…

Now define a new RST as $R = \sim_1 \cap \sim_2$

What do the equivalence classes of R look like?

$[0]_R = [0]_1 \cap [0]_2 = \{0,1,2,3\} \cap \{0,1,2,3,4,5\} = \{0,1,2,3\}$

$[4]_R = [4]_1 \cap [4]_2 = \{4,5,6,7\} \cap \{0,1,2,3,4,5\} = \{4,5\}$

$[12n]_R = \{ 12n, 12n+1, 12n+2, 12n+3 \}$

$[12n+4]_R = \{ 12n+4, 12+5 \}$

$[12n+6]_R = \{ 12n+6, 12n+7 \}$

$[12n+8]_R = \{ 12n+8, 12n+9, 12n+10, 12n+11 \}$

Example: Let $\sim_1$ and $\sim_2$ be the following RSTs over $\mathbb{N}$:

$x \sim_1 y$ iff $\lfloor x/4 \rfloor = \lfloor y/4 \rfloor$ The equivalence classes are:

$[4n]_1 = \{4n, 4n+1, 4n+2, 4n+3\}$

$[0]_1 = \{0,1,2,3\}$ $[4]_1 = \{4,5,6,7\}$ etc…

$x \sim_2 y$ iff $\lfloor x/6 \rfloor = \lfloor y/6 \rfloor$ The equivalence classes are:

$[6n]_2 = \{6n, 6n+1, 6n+2, 6n+3, 6n+4, 6n+5\}$

$[0]_2 = \{0,1,2,3,4,5\}$ $[6]_2 = \{6,7,8,9,10,11\}$ etc…

Now define a new RST as $R = \sim_1 \cap \sim_2$

What do the equivalence classes of R look like?

$[0]_R = [0]_1 \cap [0]_2 = \{0,1,2,3\} \cap \{0,1,2,3,4,5\} = \{0,1,2,3\}$

$[4]_R = [4]_1 \cap [4]_2 = \{4,5,6,7\} \cap \{0,1,2,3,4,5\} = \{4,5\}$

$[6]_R = [6]_1 \cap [6]_2 = \{4,5,6,7\} \cap \{6,7,8,9,10,11\} = \{6,7\}$

$[12n]_R = \{ 12n, 12n+1, 12n+2, 12n+3 \}$

$[12n+4]_R = \{ 12n+4, 12+5 \}$

$[12n+6]_R = \{ 12n+6, 12n+7 \}$

$[12n+8]_R = \{ 12n+8, 12n+9, 12n+10, 12n+11 \}$

Example: Let $\sim_1$ and $\sim_2$ be the following RSTs over $\mathbb{N}$:

$x \sim_1 y$ iff $\lfloor x/4 \rfloor = \lfloor y/4 \rfloor$ The equivalence classes are:

$[4n]_1 = \{4n, 4n+1, 4n+2, 4n+3\}$

$[0]_1 = \{0,1,2,3\}$ $[4]_1 = \{4,5,6,7\}$ etc…

$x \sim_2 y$ iff $\lfloor x/6 \rfloor = \lfloor y/6 \rfloor$ The equivalence classes are:

$[6n]_2 = \{6n, 6n+1, 6n+2, 6n+3, 6n+4, 6n+5\}$

$[0]_2 = \{0,1,2,3,4,5\}$ $[6]_2 = \{6,7,8,9,10,11\}$ etc…

Now define a new RST as $R = \sim_1 \cap \sim_2$

What do the equivalence classes of R look like?

$[0]_R = [0]_1 \cap [0]_2 = \{0,1,2,3\} \cap \{0,1,2,3,4,5\} = \{0,1,2,3\}$
$[4]_R = [4]_1 \cap [4]_2 = \{4,5,6,7\} \cap \{0,1,2,3,4,5\} = \{4,5\}$
$[6]_R = [6]_1 \cap [6]_2 = \{4,5,6,7\} \cap \{6,7,8,9,10,11\} = \{6,7\}$
$[8]_R = [8]_1 \cap [8]_2 = \{8,9,10,11\} \cap \{6,7,8,9,10,11\} = \{8,9,10,11\}$

$[12n]_R = \{ 12n, 12n+1, 12n+2, 12n+3 \}$
$[12n+4]_R = \{ 12n+4, 12+5 \}$
$[12n+6]_R = \{ 12n+6, 12n+7 \}$
$[12n+8]_R = \{ 12n+8, 12n+9, 12n+10, 12n+11 \}$

Example: Let $\sim_1$ and $\sim_2$ be the following RSTs over $\mathbb{N}$:

$x \sim_1 y$ iff $\lfloor x/4 \rfloor = \lfloor y/4 \rfloor$ The equivalence classes are:

$[4n]_1 = \{4n, 4n+1, 4n+2, 4n+3\}$

$[0]_1 = \{0,1,2,3\}$ $[4]_1 = \{4,5,6,7\}$ etc…

$x \sim_2 y$ iff $\lfloor x/6 \rfloor = \lfloor y/6 \rfloor$ The equivalence classes are:

$[6n]_2 = \{6n, 6n+1, 6n+2, 6n+3, 6n+4, 6n+5\}$

$[0]_2 = \{0,1,2,3,4,5\}$ $[6]_2 = \{6,7,8,9,10,11\}$ etc…

Now define a new RST as $R = \sim_1 \cap \sim_2$

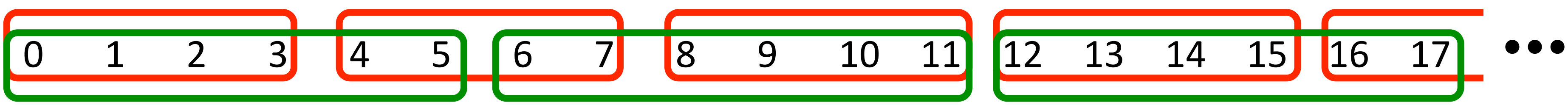
What do the equivalence classes of R look like?

$[0]_R = [0]_1 \cap [0]_2 = \{0,1,2,3\} \cap \{0,1,2,3,4,5\} = \{0,1,2,3\}$

$[4]_R = [4]_1 \cap [4]_2 = \{4,5,6,7\} \cap \{0,1,2,3,4,5\} = \{4,5\}$

$[6]_R = [6]_1 \cap [6]_2 = \{4,5,6,7\} \cap \{6,7,8,9,10,11\} = \{6,7\}$

$[8]_R = [8]_1 \cap [8]_2 = \{8,9,10,11\} \cap \{6,7,8,9,10,11\} = \{8,9,10,11\}$



$[12n]_R = \{ 12n, 12n+1, 12n+2, 12n+3 \}$

$[12n+4]_R = \{ 12n+4, 12+5 \}$

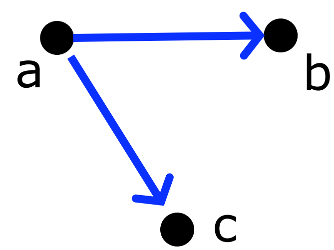
$[12n+6]_R = \{ 12n+6, 12n+7 \}$

$[12n+8]_R = \{ 12n+8, 12n+9, 12n+10, 12n+11 \}$

Generating Equivalence Relations

The smallest equivalence relation containing binary relation R
(i.e., the "equivalence closure of R") is
tsr(R)

Example: Let R = {(a,b),(a,c)} be a relation over {a,b,c}.
Let's turn it into an equivalence relation by computing tsr(R):

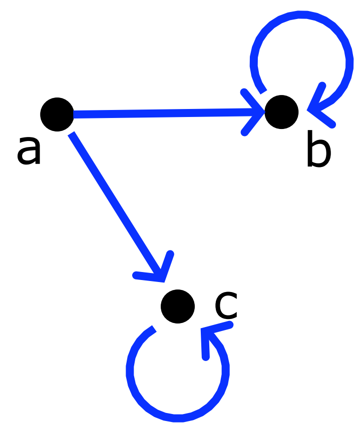


Generating Equivalence Relations

The smallest equivalence relation containing binary relation R
(i.e., the "equivalence closure of R") is
tsr(R)

Example: Let R = {(a,b),(a,c)} be a relation over {a,b,c}.
Let's turn it into an equivalence relation by computing tsr(R):

Add reflexive closure:

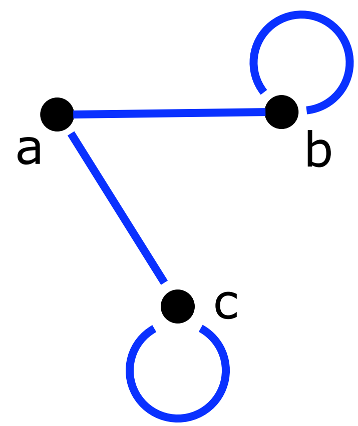


Generating Equivalence Relations

The smallest equivalence relation containing binary relation R
(i.e., the "equivalence closure of R") is
tsr(R)

Example: Let R = {(a,b),(a,c)} be a relation over {a,b,c}.
Let's turn it into an equivalence relation by computing tsr(R):

Add symmetric closure:

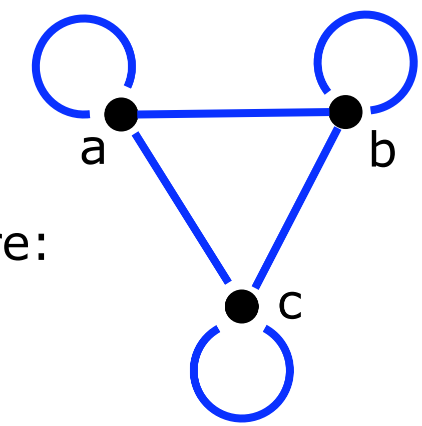


Generating Equivalence Relations

The smallest equivalence relation containing binary relation R
(i.e., the "equivalence closure of R") is
tsr(R)

Example: Let R = {(a,b),(a,c)} be a relation over {a,b,c}.
Let's turn it into an equivalence relation by computing tsr(R):

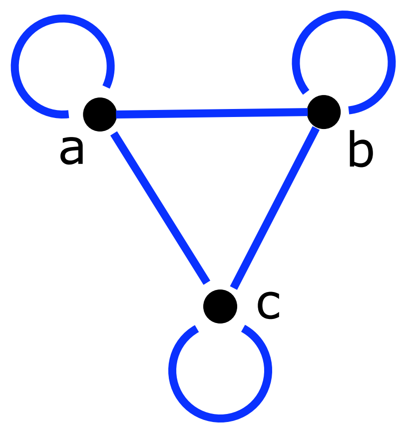
Take transitive closure:



Generating Equivalence Relations

The smallest equivalence relation containing binary relation R
(i.e., the "equivalence closure of R") is
tsr(R)

Example: Let R = {(a,b),(a,c)} be a relation over {a,b,c}.
Let's turn it into an equivalence relation by computing tsr(R):

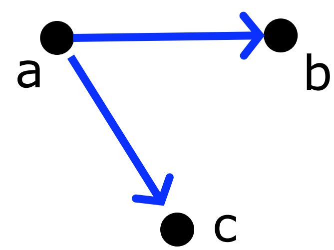


Is order important? Will str(R) work just as well?

Generating Equivalence Relations

The smallest equivalence relation containing binary relation R
(i.e., the "equivalence closure of R") is
tsr(R)

Example: Let R = {(a,b),(a,c)} be a relation over {a,b,c}.
Let's turn it into an equivalence relation by computing tsr(R):



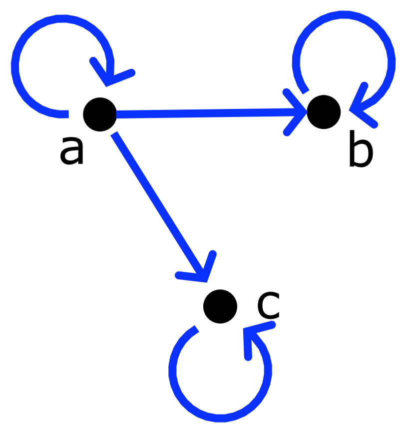
Is order important? Will str(R) work just as well?

Generating Equivalence Relations

The smallest equivalence relation containing binary relation R
(i.e., the "equivalence closure of R") is
tsr(R)

Example: Let R = {(a,b),(a,c)} be a relation over {a,b,c}.
Let's turn it into an equivalence relation by computing tsr(R):

Add reflexive closure:



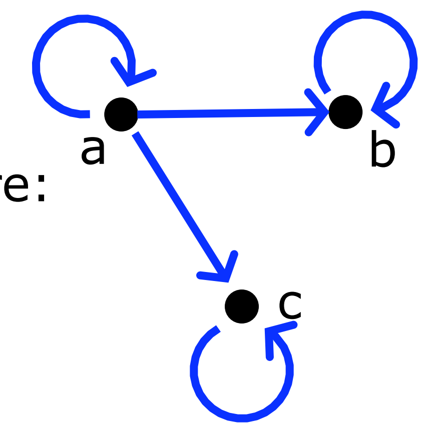
Is order important? Will str(R) work just as well?

Generating Equivalence Relations

The smallest equivalence relation containing binary relation R
(i.e., the "equivalence closure of R") is
tsr(R)

Example: Let R = {(a,b),(a,c)} be a relation over {a,b,c}.
Let's turn it into an equivalence relation by computing tsr(R):

Take transitive closure:
(No change)



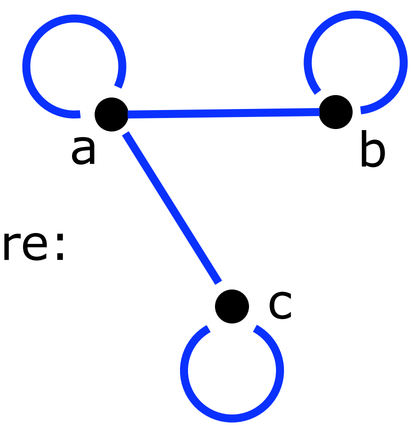
Is order important? Will str(R) work just as well?

Generating Equivalence Relations

The smallest equivalence relation containing binary relation R
(i.e., the "equivalence closure of R") is
tsr(R)

Example: Let R = {(a,b),(a,c)} be a relation over {a,b,c}.
Let's turn it into an equivalence relation by computing tsr(R):

Take symmetric closure:



Is order important? Will str(R) work just as well?

NO!

Equivalence and Meaning

For sets where the elements have no "meaning"…
  just use BASIC EQUALITY on S.
For sets where the elements do have some "meaning"…

We need another set of "values", V, and a "meaning function", m.

$$m: S \rightarrow V$$

Now we can say that two elements in set S are equivalent/equal if they mean the same thing.

$x \sim y$ iff $m(x)=m(y)$ $x = y$ iff $m(x)=m(y)$

Example:

S = strings from {1,+}* that are well-formed expressions, e.g.,

111+111 , 111111 , 1+1+1+1+1+1 , …

$V = \mathbb{N}$

Define m as follows: $m(1^k) = k$, for $k>0$

$m(e_1+e_2) = m(e_1) + m(e_2)$

$m(111+111) = 6$

So our equivalence relation is the kernel relation of m.

111+111 ~ 11+11+11

Kruskal’s Algorithm

To compute a minimal spanning tree.

Overview:

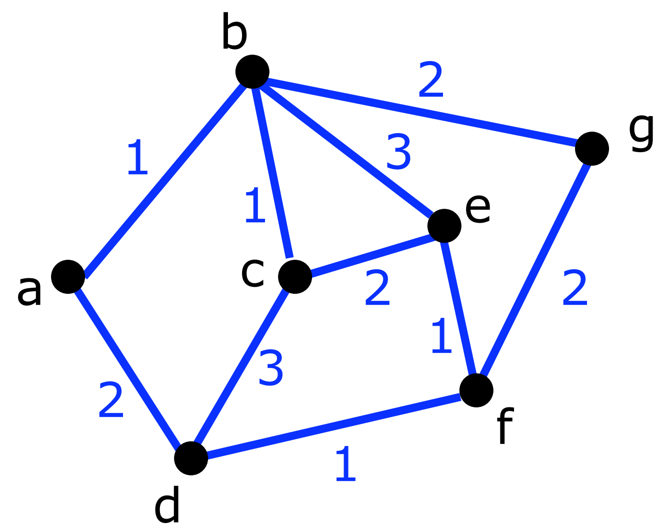
- Look at the set of all vertices in the graph.
- Create a partitioning of the set.
  - Start with the finest possible partitioning.
    - Every node is in an equivalence set by itself.
- Gradually merge partitions.
  - Until there is only one partition, containing all the nodes.
- We are constructing the spanning tree by adding edges.
- Two nodes x and y are in the same subset if
  - there is a path from x to y in the current spanning tree.

Kruskal's Algorithm

Order the edges by weight and put them into a list, L.
T will be the set of egdes representing the spanning tree.
T := Ø.
Create the initial (finest) partitioning.
[v] := {v} for each vertex v in the graph.
while there are two or more equivalence classes **do**
    {x,y} := head(L)
    L := tail(L)
    if [x] ≠ [y] **then**
        T := T ∪ {{x,y}}
        Merge the sets [x] and [y]
            i.e., replace [x] and [y] by [x] ∪ [y]
    endIf
endWhile

Kruskal's Algorithm

List L =
{a,b} {b,c} {d,f} {e,f} {a,d}
{c,e} {f,g} {b,g} {c,d} {b,e}



T (Set of edges):

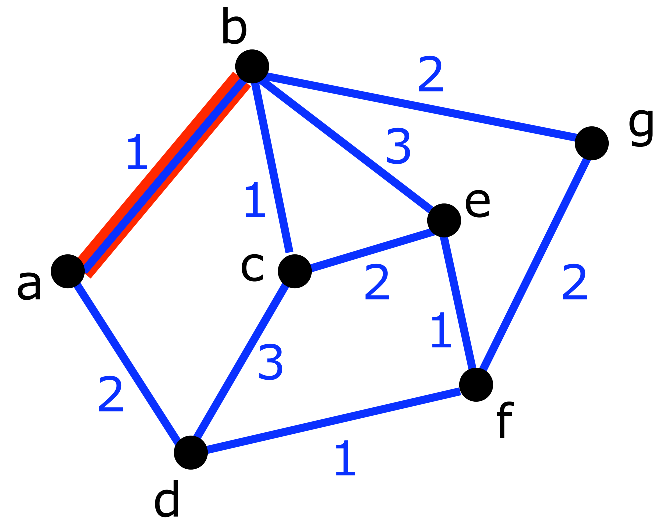
T = {}

Equivalence Classes:

{a} {b} {c} {d} {e} {f} {g}

Kruskal's Algorithm

List L =
{a,b} {b,c} {d,f} {e,f} {a,d}
{c,e} {f,g} {b,g} {c,d} {b,e}



T (Set of edges):

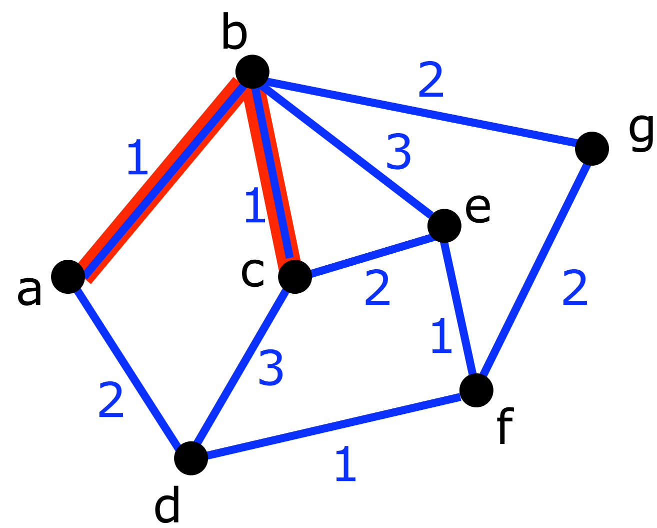
T = {}
∪ {{a,b}}

Equivalence Classes:

{a} {b} {c} {d} {e} {f} {g}
{a,b} {c} {d} {e} {f} {g}

Kruskal's Algorithm

List L =
{a,b} {b,c} {d,f} {e,f} {a,d}
{c,e} {f,g} {b,g} {c,d} {b,e}



T (Set of edges):

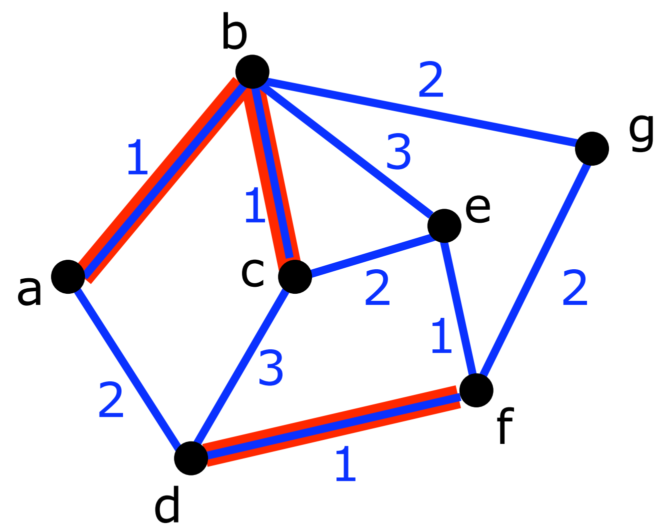
T = {}
∪ {{a,b}}
∪ {{b,c}}

Equivalence Classes:

{a}	{b}	{c}	{d}	{e}	{f}	{g}
{a,b}		{c}	{d}	{e}	{f}	{g}
{a,b,c}			{d}	{e}	{f}	{g}

Kruskal’s Algorithm

List L =
{a,b} {b,c} {d,f} {e,f} {a,d}
{c,e} {f,g} {b,g} {c,d} {b,e}



T (Set of edges):

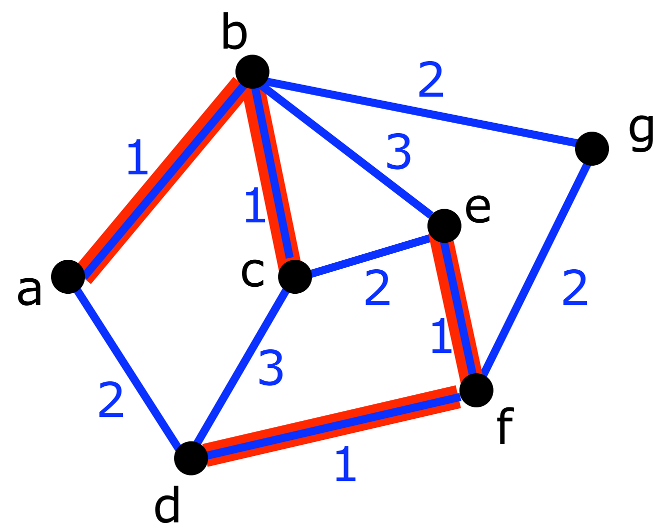
T = {}
∪ {{a,b}}
∪ {{b,c}}
∪ {{d,f}}

Equivalence Classes:

{a}	{b}	{c}	{d}	{e}	{f}	{g}
{a,b}		{c}	{d}	{e}	{f}	{g}
{a,b,c}			{d}	{e}	{f}	{g}
{a,b,c}			{d,f}	{e}		{g}

Kruskal's Algorithm

List L =
{a,b} {b,c} {d,f} {e,f} {a,d}
{c,e} {f,g} {b,g} {c,d} {b,e}



T (Set of edges):

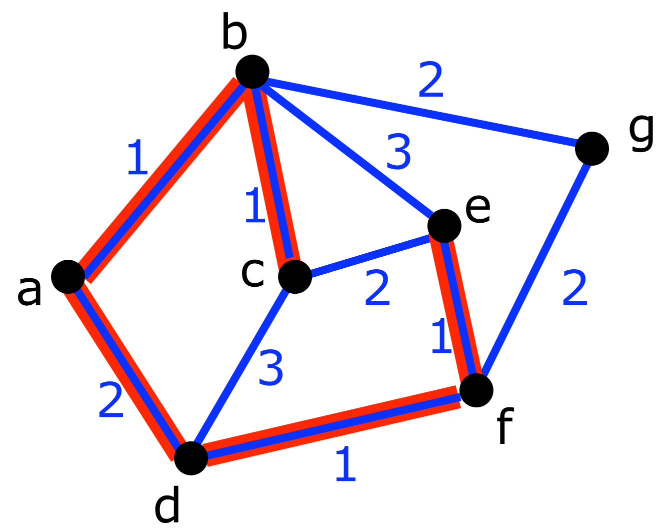
T = {}
∪ {{a,b}}
∪ {{b,c}}
∪ {{d,f}}
∪ {{e,f}}

Equivalence Classes:

{a}	{b}	{c}	{d}	{e}	{f}	{g}
{a,b}		{c}	{d}	{e}	{f}	{g}
{a,b,c}			{d}	{e}	{f}	{g}
{a,b,c}			{d,f}	{e}		{g}
{a,b,c}			{d,e,f}			{g}

Kruskal's Algorithm

List L =
{a,b} {b,c} {d,f} {e,f} {a,d}
{c,e} {f,g} {b,g} {c,d} {b,e}



T (Set of edges):

T = {}
∪ {{a,b}}
∪ {{b,c}}
∪ {{d,f}}
∪ {{e,f}}
∪ {{a,d}}

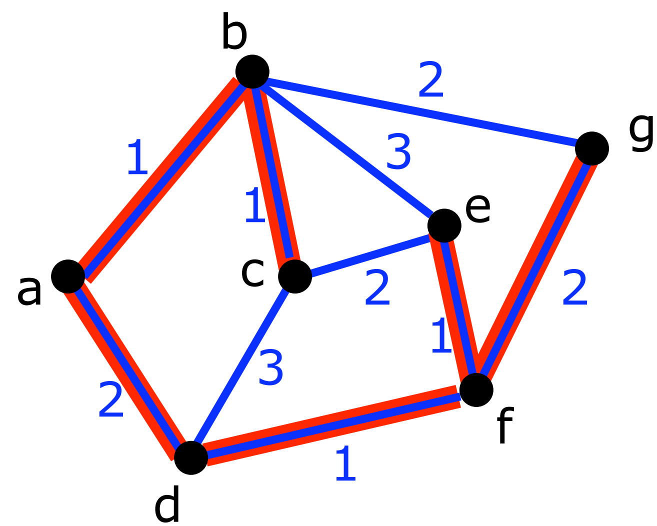
Equivalence Classes:

{a}	{b}	{c}	{d}	{e}	{f}	{g}
{a,b}		{c}	{d}	{e}	{f}	{g}
{a,b,c}			{d}	{e}	{f}	{g}
{a,b,c}			{d,f}	{e}		{g}
{a,b,c}			{d,e,f}			{g}
{a,b,c,d,e,f}						{g}

Kruskal's Algorithm

List L =

{a,b} {b,c} {d,f} {e,f} {a,d}
{c,e} {f,g} {b,g} {c,d} {b,e}



T (Set of edges):

T = {}
∪ {{a,b}}
∪ {{b,c}}
∪ {{d,f}}
∪ {{e,f}}
∪ {{a,d}}
∪ {{f,g}}

Equivalence Classes:

{a}	{b}	{c}	{d}	{e}	{f}	{g}
{a,b}		{c}	{d}	{e}	{f}	{g}
{a,b,c}			{d}	{e}	{f}	{g}
{a,b,c}			{d,f}	{e}		{g}
{a,b,c}			{d,e,f}			{g}
{a,b,c,d,e,f}						{g}
{a,b,c,d,e,f,g}						

Problem: Given an bunch of explicit equivalences

1~8   4~5   9~2   4~10   3~7   6~3   4~9   *(the "generators")*

build up the full equivalence relation, by constructing a partioning.

Approach: Start with a collection of singleton sets

{1}   {2}   {3}   {4}   {5}   {6}   {7}   {8}   {9}   {10}

Process the generators one at a time.
Merge the partitions.

1~8   merge {1} and {8} to produce {1,8}

Result:

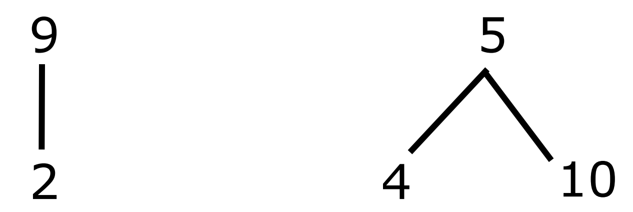
1~8   4~5   9~2   4~10   3~7   6~3   4~9

{ {1,8} , {2,4,5,9,10} , {3,6,7} }

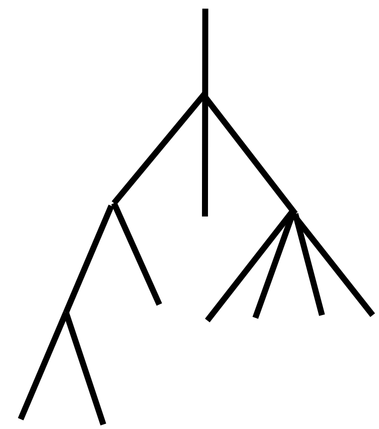
How to represent sets of nodes?

As trees!

Each node may have a parent.



{2,9} {4,5,10}



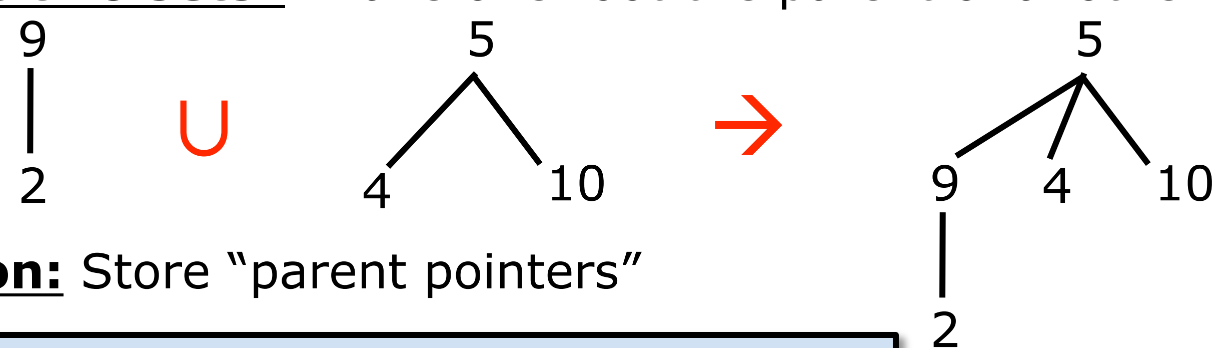
We don't care which node happens to be the root.

<u>Are two nodes in the same set?</u>

Follow the parent links to the roots…

Are they the same root?

<u>How to merge two sets?</u> Make one root the parent of another.



<u>Representation:</u> Store "parent pointers"

node:	1	2	3	4	5	6	7	8	9	10
parent:	8	9	6	5	0	0	6	0	~~0~~ 5	5