Sets

Epp, chapter 5

What is a set?

- A set is a group of "objects"
  - People in a class: { Alice, Bob, Chris }
  - Classes offered by a department: { CS 101, CS 202, … }
  - Colors of a rainbow: { red, orange, yellow, green, blue, purple }
  - States of matter { solid, liquid, gas, plasma }
  - States in the US: { Alabama, Alaska, Virginia, … }
  - Sets can contain non-related elements: { 3, a, red, Virginia }

- Although a set can contain (almost) anything, we will most often use sets of numbers
  - All positive numbers less than or equal to 5: {1, 2, 3, 4, 5}
  - A few selected real numbers: { 2.1, π, 0, -6.32, e }

Set properties 1

- Order does not matter
  - We often write them in order because it is easier for humans to understand it that way
  - {1, 2, 3, 4, 5} is equivalent to {3, 5, 2, 4, 1}

- Sets are notated with curly brackets

Set properties 2

- Sets do not have duplicate elements
  - Consider the set of vowels in the alphabet.
    - It makes no sense to list them as {a, a, a, e, i, o, o, o, o, o, u}
    - What we really want is just {a, e, i, o, u}
  - Consider the list of students in this class
    - Again, it does not make sense to list somebody twice
- Note that a list is like a set, but order does matter and duplicate elements are allowed
  - We won't be studying lists much in this class

Specifying a set 1

- Sets are usually represented by a capital letter (A, B, S, etc.)

- Elements are usually represented by an italic lower-case letter (*a*, *x*, *y*, etc.)

- Easiest way to specify a set is to list all the elements: A = {1, 2, 3, 4, 5}
  - – Not always feasible for large or infinite sets

Specifying a set 2

- Can use an ellipsis (…): B = {0, 1, 2, 3, …}
  - Can cause confusion. Consider the set C = {3, 5, 7, …}. What comes next?
  - If the set is all odd integers greater than 2, it is 9
  - If the set is all prime numbers greater than 2, it is 11

- Can use set-builder notation
  - D = {x | x is prime and x > 2}
  - E = {x | x is odd and x > 2}
  - The vertical bar means "such that"
  - Thus, set D is read (in English) as: "all elements *x* such that *x* is prime and *x* is greater than 2"

Specifying a set 3

- A set is said to "contain" the various "members" or "elements" that make up the set
  - If an element *a* is a member of (or an element of) a set S, we use then notation $a \in S$
    - $4 \in \{1, 2, 3, 4\}$
  - If an element is not a member of (or an element of) a set S, we use the notation $a \notin S$
    - $7 \notin \{1, 2, 3, 4\}$
    - Virginia $\notin \{1, 2, 3, 4\}$

Often used sets

- **N** = {0, 1, 2, 3, …} is the set of natural numbers
- **Z** = {…, -2, -1, 0, 1, 2, …} is the set of integers
- $\mathbf{Z}^+$ = {1, 2, 3, …} is the set of positive integers (a.k.a whole numbers)
  - Note that people disagree on the exact definitions of whole numbers and natural numbers
- **Q** = $\{p/q \mid p \in \mathbf{Z}, q \in \mathbf{Z}, q \neq 0\}$ is the set of rational numbers
  - Any number that can be expressed as a fraction of two integers (where the bottom one is not zero)
- **R** is the set of real numbers

The universal set 1

- U is the universal set – the set of all of elements (or the "universe") from which given any set is drawn
  - For the set {-2, 0.4, 2}, U would be the real numbers
  - For the set {0, 1, 2}, U could be the natural numbers (zero and up), the integers, the rational numbers, or the real numbers, depending on the context

The universal set 2

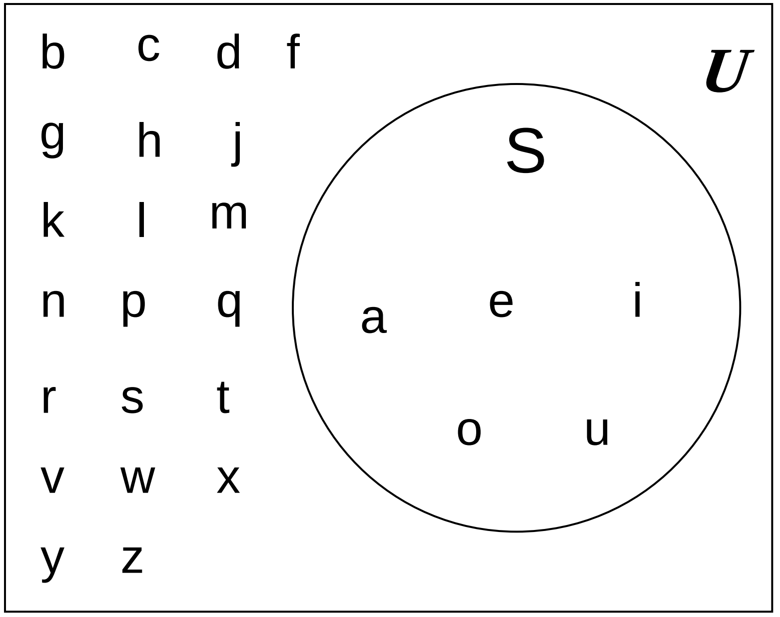
– For the set of the students in this class, ***U*** would be all the students in the University (or perhaps all the people in the world)

– For the set of the vowels of the alphabet, ***U*** would be all the letters of the alphabet

– To differentiate ***U*** from U (which is a set operation), the universal set is written in a different font (and in bold and italics)

Venn diagrams

- Represents sets graphically
  - The box represents the universal set
  - Circles represent the set(s)
- Consider set S, which is the set of all vowels in the alphabet
- The individual elements are usually not written in a Venn diagram



Sets of sets

- Sets can contain other sets
  - S = { {1}, {2}, {3} }
  - T = { {1}, {{2}}, {{{3}}} }
  - V = { {{1}, {{2}}}, {{{3}}}, { {1}, {{2}}, {{{3}}} } }
    - V has only 3 elements!
- Note that 1 ≠ {1} ≠ {{1}} ≠ {{{1}}}
  - They are all different

The empty set 1

- If a set has zero elements, it is called the empty (or null) set
  - Written using the symbol ∅
  - Thus, ∅ = { } ← **VERY IMPORTANT**
  - If you get confused about the empty set in a problem, try replacing ∅ by { }
- As the empty set is a set, it can be a element of other sets
  - { ∅, 1, 2, 3, x } is a valid set

The empty set 1

- Note that $\emptyset \neq \{ \emptyset \}$
  - The first is a set of zero elements
  - The second is a set of 1 element (that one element being the empty set)

- Replace $\emptyset$ by { }, and you get: $\{ \} \neq \{ \{ \} \}$
  - It's easier to see that they are not equal that way

Set equality

- Two sets are equal if they have the same elements
  - {1, 2, 3, 4, 5} = {5, 4, 3, 2, 1}
    - Remember that order does not matter!
  - {1, 2, 3, 2, 4, 3, 2, 1} = {4, 3, 2, 1}
    - Remember that duplicate elements do not matter!
- Two sets are not equal if they do not have the same elements
  - {1, 2, 3, 4, 5} ≠ {1, 2, 3, 4}

Subsets 1

- If all the elements of a set S are also elements of a set T, then S is a subset of T
  - For example, if S = {2, 4, 6} and T = {1, 2, 3, 4, 5, 6, 7}, then S is a subset of T
  - This is specified by $S \subseteq T$
    - Or by $\{2, 4, 6\} \subseteq \{1, 2, 3, 4, 5, 6, 7\}$
- If S is not a subset of T, it is written as such: $S \nsubseteq T$
  - For example, $\{1, 2, 8\} \nsubseteq \{1, 2, 3, 4, 5, 6, 7\}$

Subsets 2

- Note that any set is a subset of itself!
  - Given set S = {2, 4, 6}, since all the elements of S are elements of S, S is a subset of itself
  - This is kind of like saying 5 is less than or equal to 5
  - Thus, for any set S, $S \subseteq S$

Subsets 3

- The empty set is a subset of *all* sets (including itself!)
  - Recall that all sets are subsets of themselves
- *All* sets are subsets of the universal set
- A horrible way to define a subset:

  $\forall x\,(\,x \in A \rightarrow x \in B\,)$

  - English translation: for all possible values of x, (meaning for all possible elements of a set), if x is an element of A, then x is an element of B
  - This type of notation will be gone over later

Proper Subsets 1

- If S is a subset of T, and S is not equal to T, then S is a proper subset of T
  - Let T = {0, 1, 2, 3, 4, 5}
  - If S = {1, 2, 3}, S is not equal to T, and S is a subset of T
  - A proper subset is written as $S \subset T$
  - Let R = {0, 1, 2, 3, 4, 5}. R is equal to T, and thus is a subset (but not a proper subset) or T
    - Can be written as: $R \subseteq T$ and $R \not\subset T$ (or just R = T)
  - Let Q = {4, 5, 6}. Q is neither a subset or T nor a proper subset of T

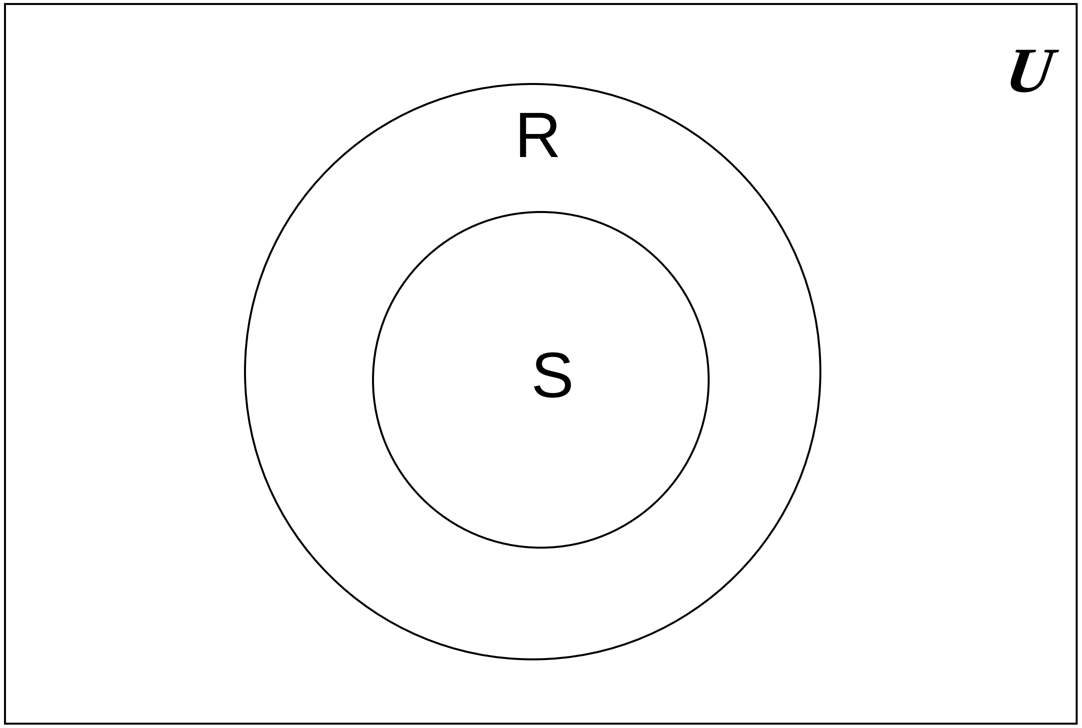
Proper Subsets 2

- The difference between “subset” and “proper subset” is like the difference between “less than or equal to” and “less than” for numbers

- The empty set is a proper subset of all sets other than the empty set (as it is equal to the empty set)

Proper subsets: Venn diagram

$S \subset R$



Set cardinality

- The cardinality of a set is the number of elements in a set
  - Written as |A|
- Examples
  - Let R = {1, 2, 3, 4, 5}. Then |R| = 5
  - |∅| = 0
  - Let S = {∅, {a}, {b}, {a, b}}. Then |S| = 4
- This is the same notation used for vector length in geometry
- A set with one element is sometimes called a singleton set

Power sets 1

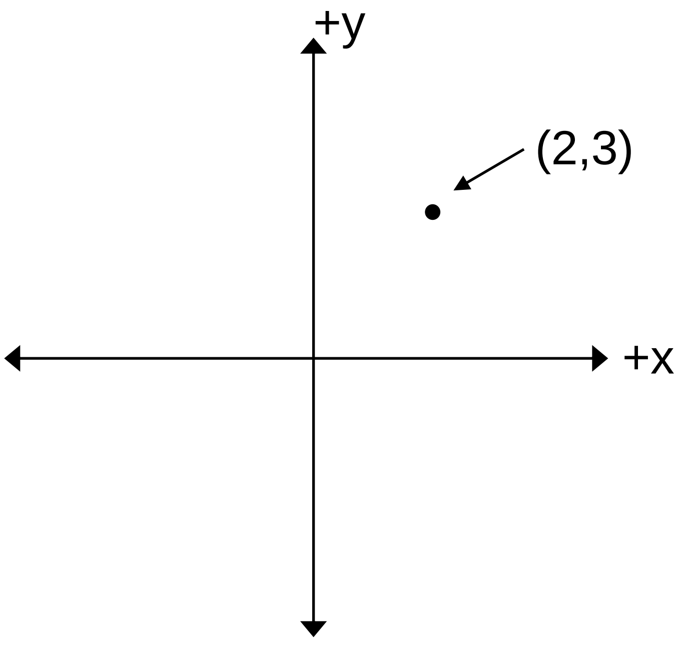
- Given the set S = {0, 1}. What are all the possible subsets of S?
  - They are: ∅ (as it is a subset of all sets), {0}, {1}, and {0, 1}
  - The power set of S (written as P(S)) is the set of all the subsets of S
  - P(S) = { ∅, {0}, {1}, {0,1} }
    - Note that |S| = 2 and |P(S)| = 4

Power sets 2

- Let T = {0, 1, 2}. The P(T) = { ∅, {0}, {1}, {2}, {0,1}, {0,2}, {1,2}, {0,1,2} }
  - Note that |T| = 3 and |P(T)| = 8
- P(∅) = { ∅ }
  - Note that |∅| = 0 and |P(∅)| = 1
- If a set has *n* elements, then the power set will have 2^n elements

Tuples

- In 2-dimensional space, it is a (*x*, *y*) pair of numbers to specify a location
- In 3-dimensional (1,2,3) is not the same as (3,2,1) – space, it is a (*x*, *y*, *z*) triple of numbers
- In *n*-dimensional space, it is a *n*-tuple of numbers
  - Two-dimensional space uses pairs, or 2-tuples
  - Three-dimensional space uses triples, or 3-tuples
- Note that these tuples are **ordered**, unlike sets
  - the *x* value has to come first



Cartesian products 1

- A Cartesian product is a set of all ordered 2-tuples where each “part” is from a given set
  - Denoted by A x B, and uses parenthesis (not curly brackets)
  - For example, 2-D Cartesian coordinates are the set of all ordered pairs **Z** x **Z**
    - Recall **Z** is the set of all integers
    - This is all the possible coordinates in 2-D space
  - Example: Given A = { a, b } and B = { 0, 1 }, what is their Cartiesian product?
    - C = A x B = { (a,0), (a,1), (b,0), (b,1) }

Cartesian products 2

- Note that Cartesian products have only 2 parts in these examples (later examples have more parts)
- Formal definition of a Cartesian product:
  - $A \times B = \{ (a,b) \mid a \in A \text{ and } b \in B \}$

Cartesian products 3

- All the possible grades in this class will be a Cartesian product of the set S of all the students in this class and the set G of all possible grades
  - Let S = { Alice, Bob, Chris } and G = { A, B, C }
  - D = { (Alice, A), (Alice, B), (Alice, C), (Bob, A), (Bob, B), (Bob, C), (Chris, A), (Chris, B), (Chris, C) }
  - The final grades will be a subset of this: { (Alice, C), (Bob, B), (Chris, A) }
    - Such a subset of a Cartesian product is called a relation (more on this later in the course)

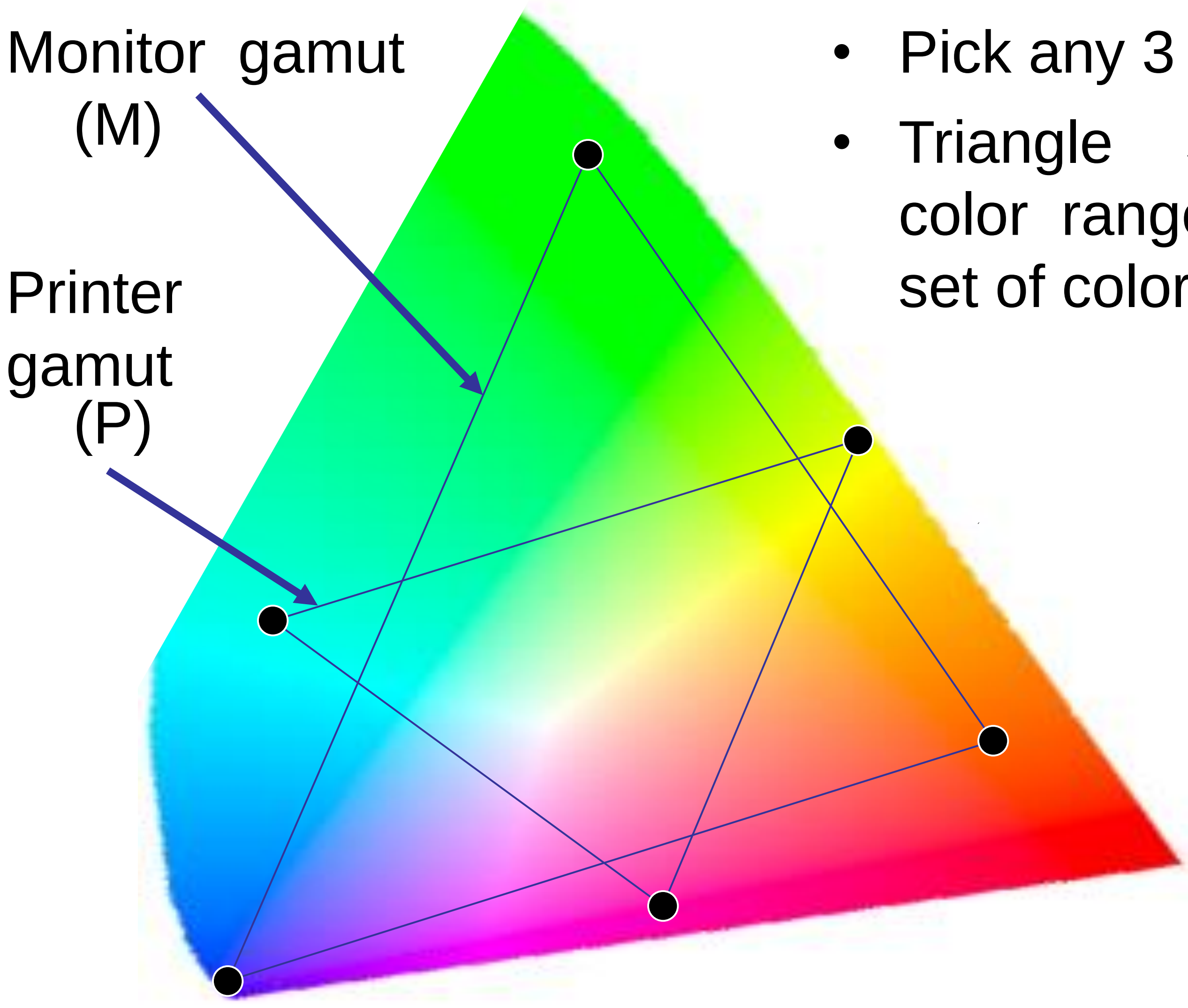
Cartesian products 4

- There can be Cartesian products on more than two sets
- A 3-D coordinate is an element from the Cartesian product of **Z** x **Z** x **Z**

Set Operations

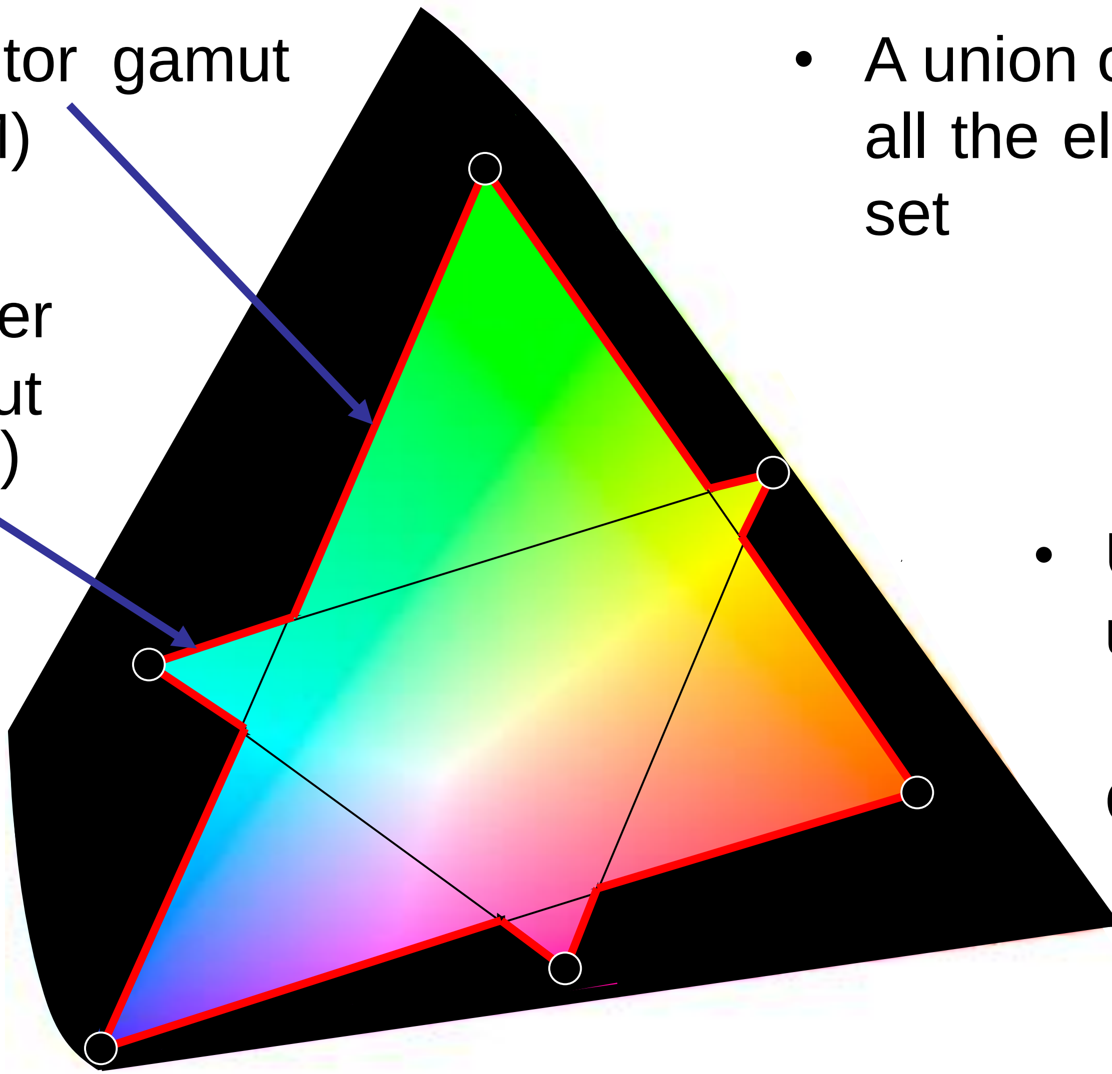
Epp, chapter 3

Sets of Colors



- Pick any 3 “primary” colors
- Triangle shows mixable color range (gamut) – the set of colors

n 1



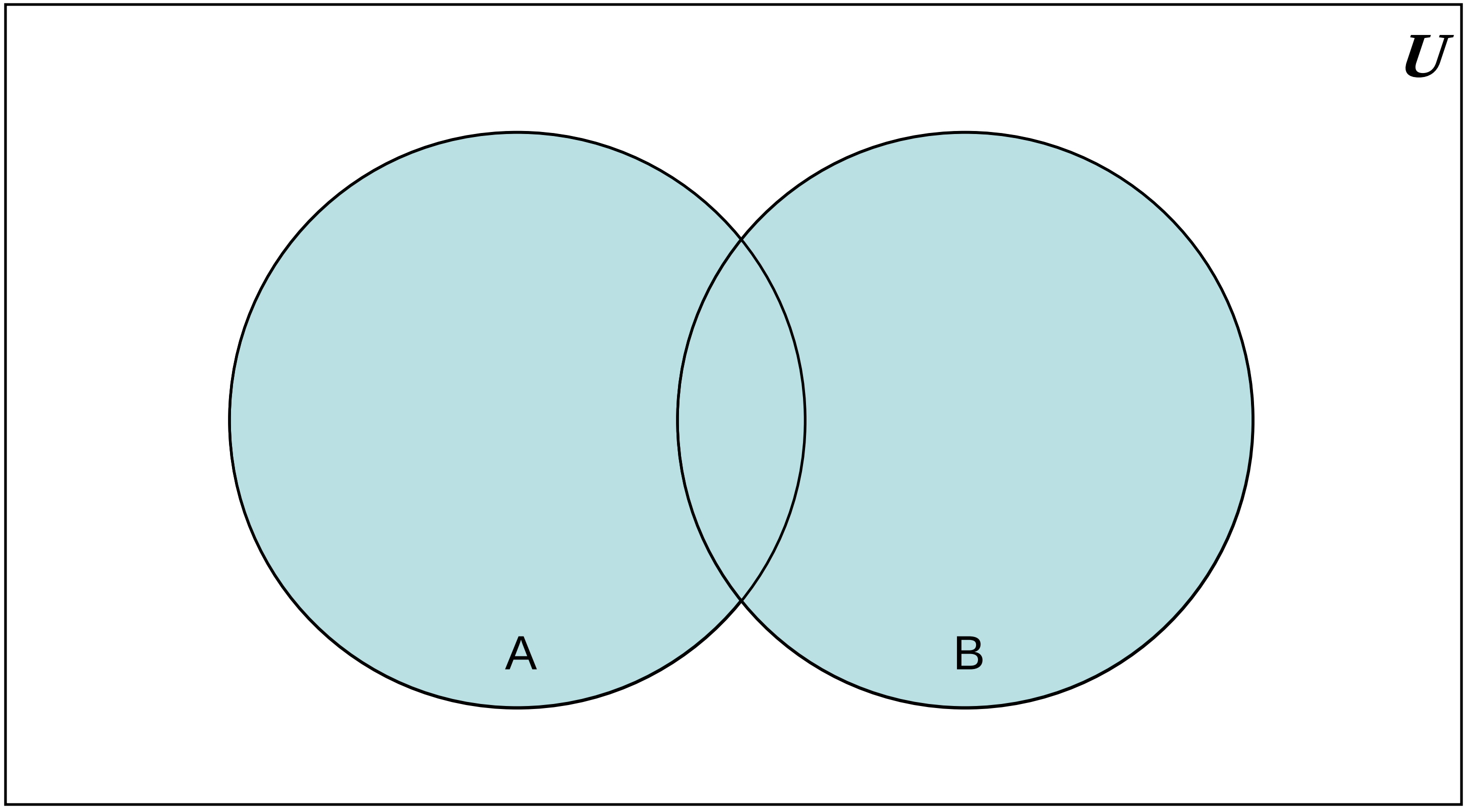
- A union of the sets contains all the elements in EITHER set

- Union symbol is usually a U
  - Example:

C = M U P

Set operations: Union 2

A U B



Set operations: Union 3

- Formal definition for the union of two sets:
  A U B = { x | $x \in$ A **or** $x \in$ B }
- Further examples
  - {1, 2, 3} U {3, 4, 5} = {1, 2, 3, 4, 5}
  - {New York, Washington} U {3, 4} = {New York, Washington, 3, 4}
  - {1, 2} U ∅ = {1, 2}

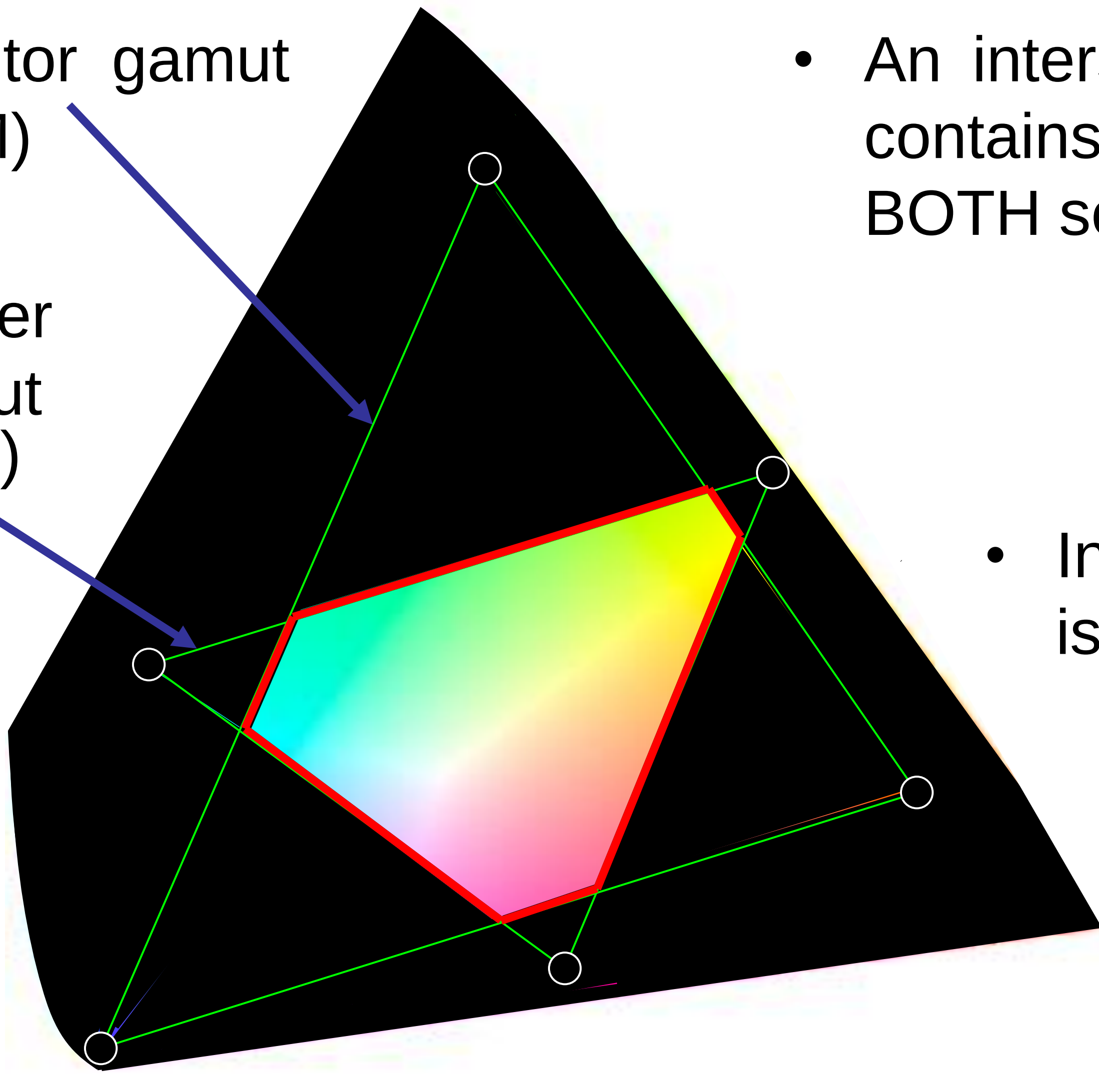
Set operations: Union 4

- Properties of the union operation
  - A U ∅ = A — Identity law
  - A U ***U*** = ***U*** — Domination law
  - A U A = A — Idempotent law
  - A U B = B U A — Commutative law
  - A U (B U C) = (A U B) U C — Associative law

Set operations: Intersection 1

Monitor gamut (M)

Printer gamut (P)



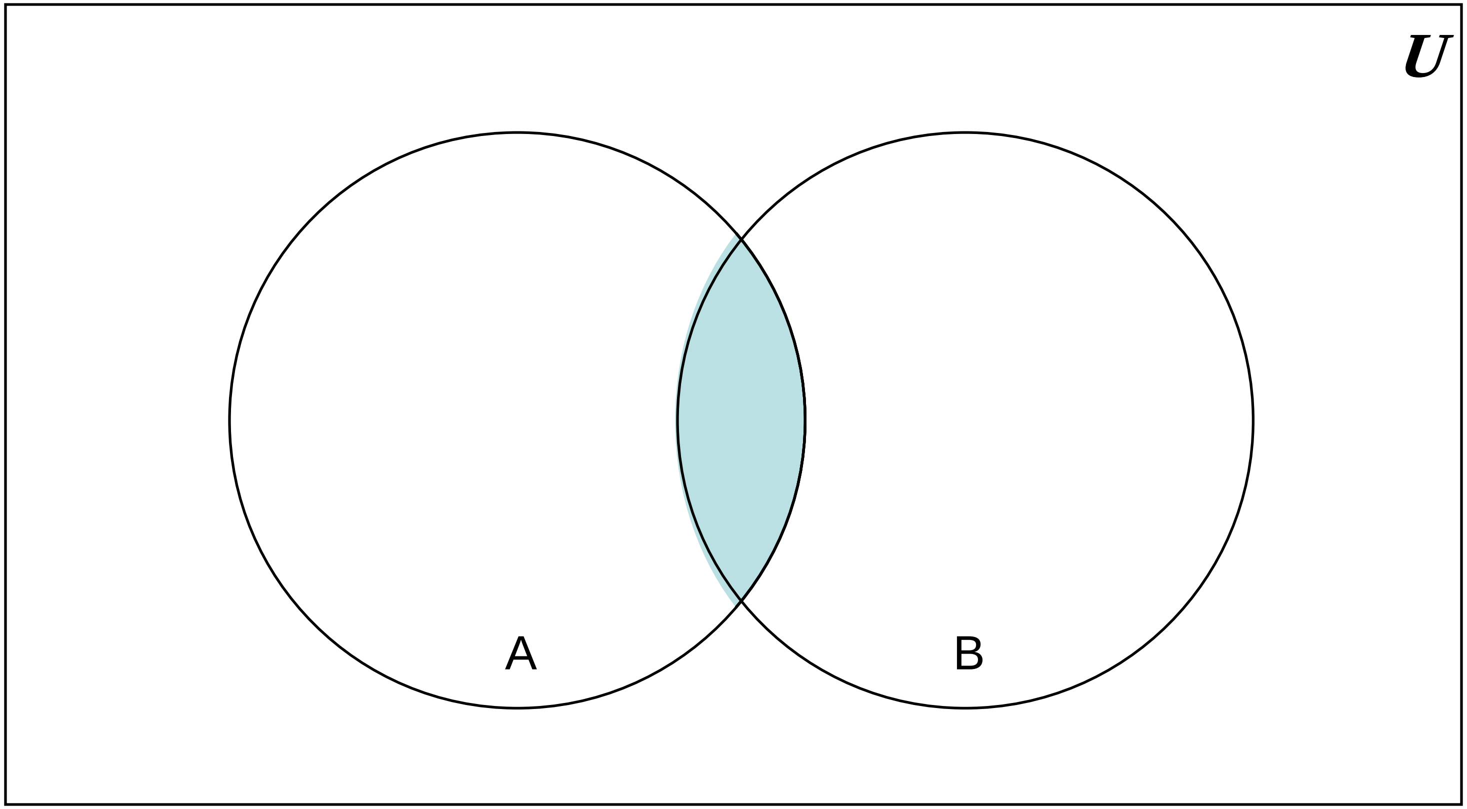
- An intersection of the sets contains all the elements in BOTH sets

- Intersection symbol is a ∩
    - Example:

$C = M \cap P$

Set operations: Intersection 2

$A \cap B$

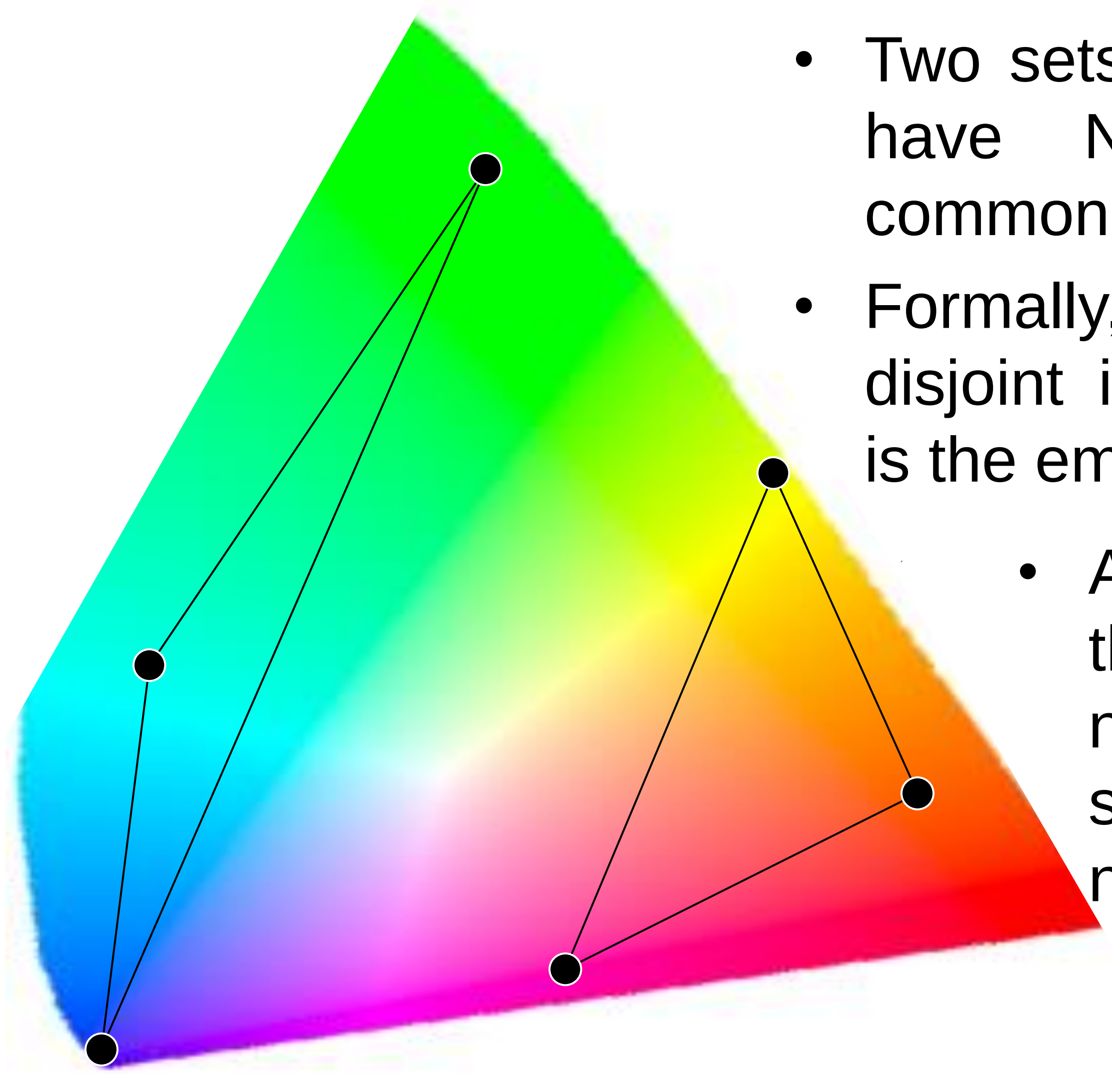


Set operations: Intersection 3

- Formal definition for the intersection of two sets: $A \cap B = \{ x \mid x \in A \textbf{ and } x \in B \}$
- Further examples
  - $\{1, 2, 3\} \cap \{3, 4, 5\} = \{3\}$
  - $\{$New York, Washington$\} \cap \{3, 4\} = \emptyset$
    - No elements in common
  - $\{1, 2\} \cap \emptyset = \emptyset$
    - Any set intersection with the empty set yields the empty set

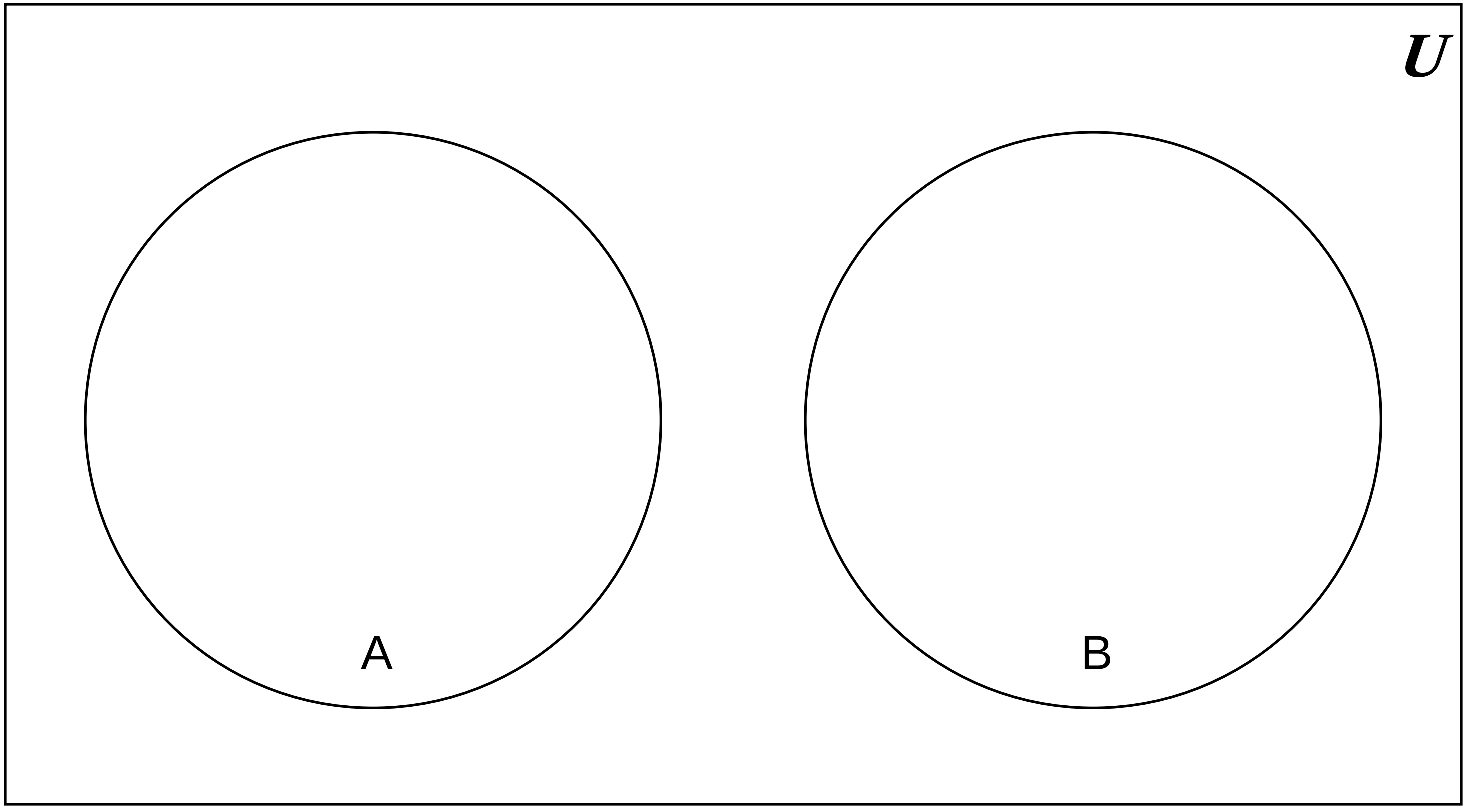
Set operations: Intersection 4

- Properties of the intersection operation
  - $A \cap \boldsymbol{U} = A$ Identity law
  - $A \cap \emptyset = \emptyset$ Domination law
  - $A \cap A = A$ Idempotent law
  - $A \cap B = B \cap A$ Commutative law
  - $A \cap (B \cap C) = (A \cap B) \cap C$ Associative law



- Two sets are disjoint if the have NO elements in common
- Formally, two sets are disjoint if their intersection is the empty set
  - Another example: the set of the even numbers and the set of the odd numbers

Disjoint sets 2



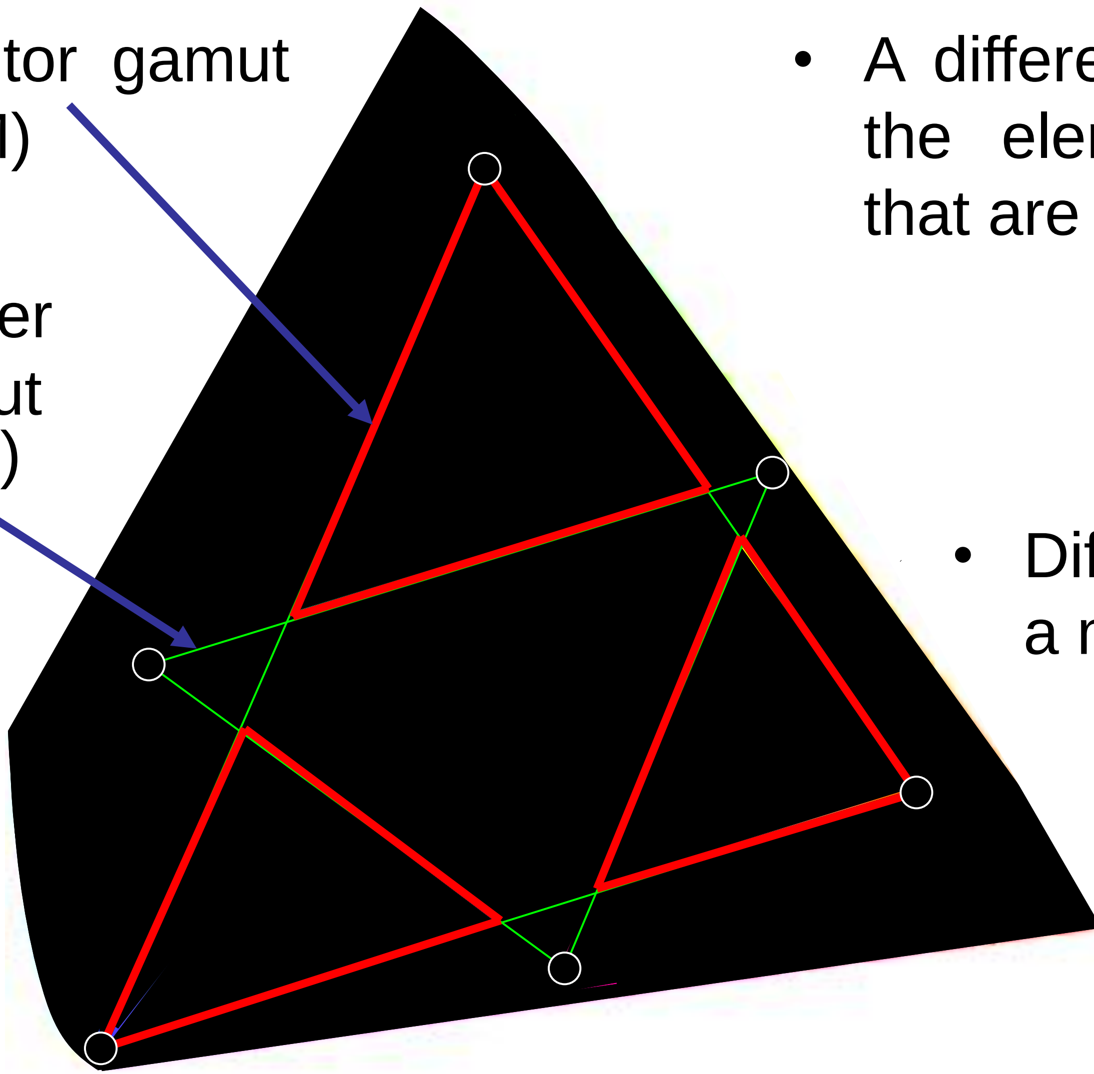
Disjoint sets 3

- Formal definition for disjoint sets: two sets are disjoint if their intersection is the empty set
- Further examples
  - {1, 2, 3} and {3, 4, 5} are not disjoint
  - {New York, Washington} and {3, 4} are disjoint
  - {1, 2} and ∅ are disjoint
    - Their intersection is the empty set
  - ∅ and ∅ are disjoint!
    - Their intersection is the empty set

Set operations:

Monitor gamut (M)

Printer gamut (P)



- A difference of two sets is the elements in one set that are NOT in the other

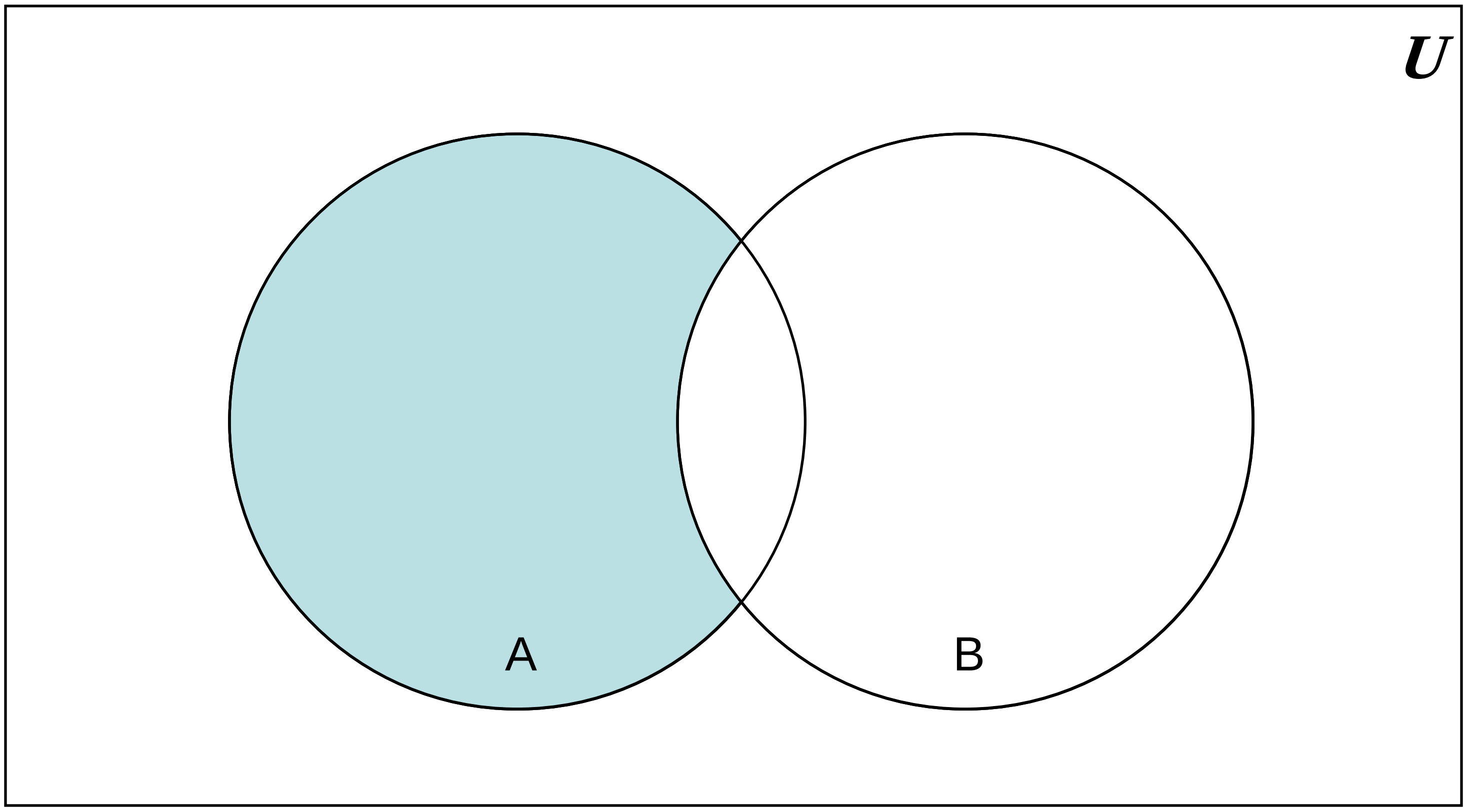
- Difference symbol is a minus sign
  - Example:

C = M - P

- Also visa-versa: C = P - M

Set operations: Difference 2

A - B
B - A



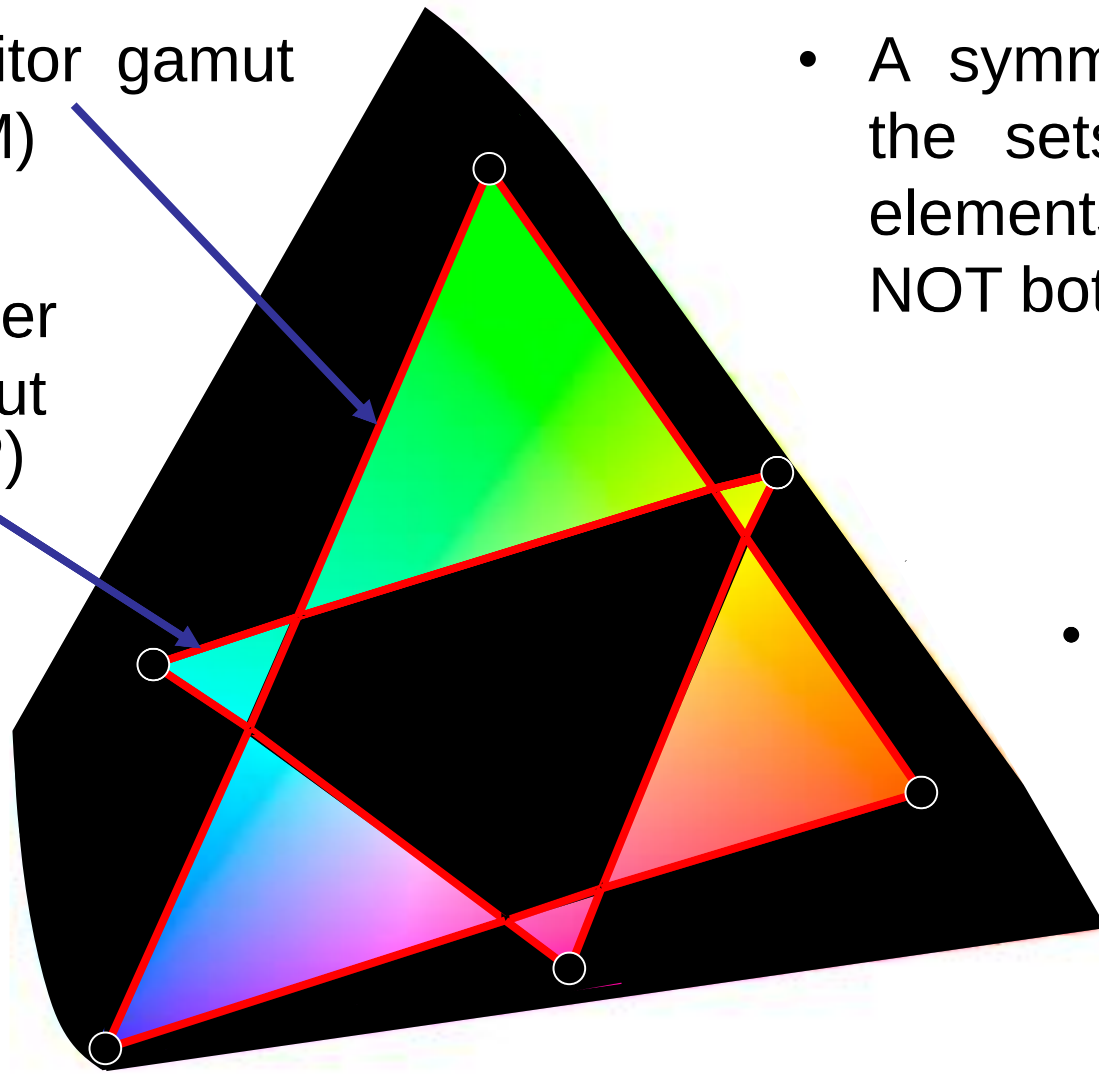
Set operations: Difference 3

- Formal definition for the difference of two sets:
  $A - B = \{ x \mid x \in A \textbf{ and } x \notin B \}$
  $A - B = A \cap \overline{B}$ ← Important!
- Further examples
  - $\{1, 2, 3\} - \{3, 4, 5\} = \{1, 2\}$
  - {New York, Washington} - {3, 4} = {New York, Washington}
  - $\{1, 2\} - \emptyset = \{1, 2\}$
    - The difference of any set S with the empty set will be the set S

Set operations: Symmetric

Monitor gamut (M)

Printer gamut (P)



- A symmetric difference of the sets contains all the elements in either set but NOT both

- Symetric diff. symbol is a ⊕
  - Example:

C = M ⊕ P

Set operations: Symmetric Difference 2

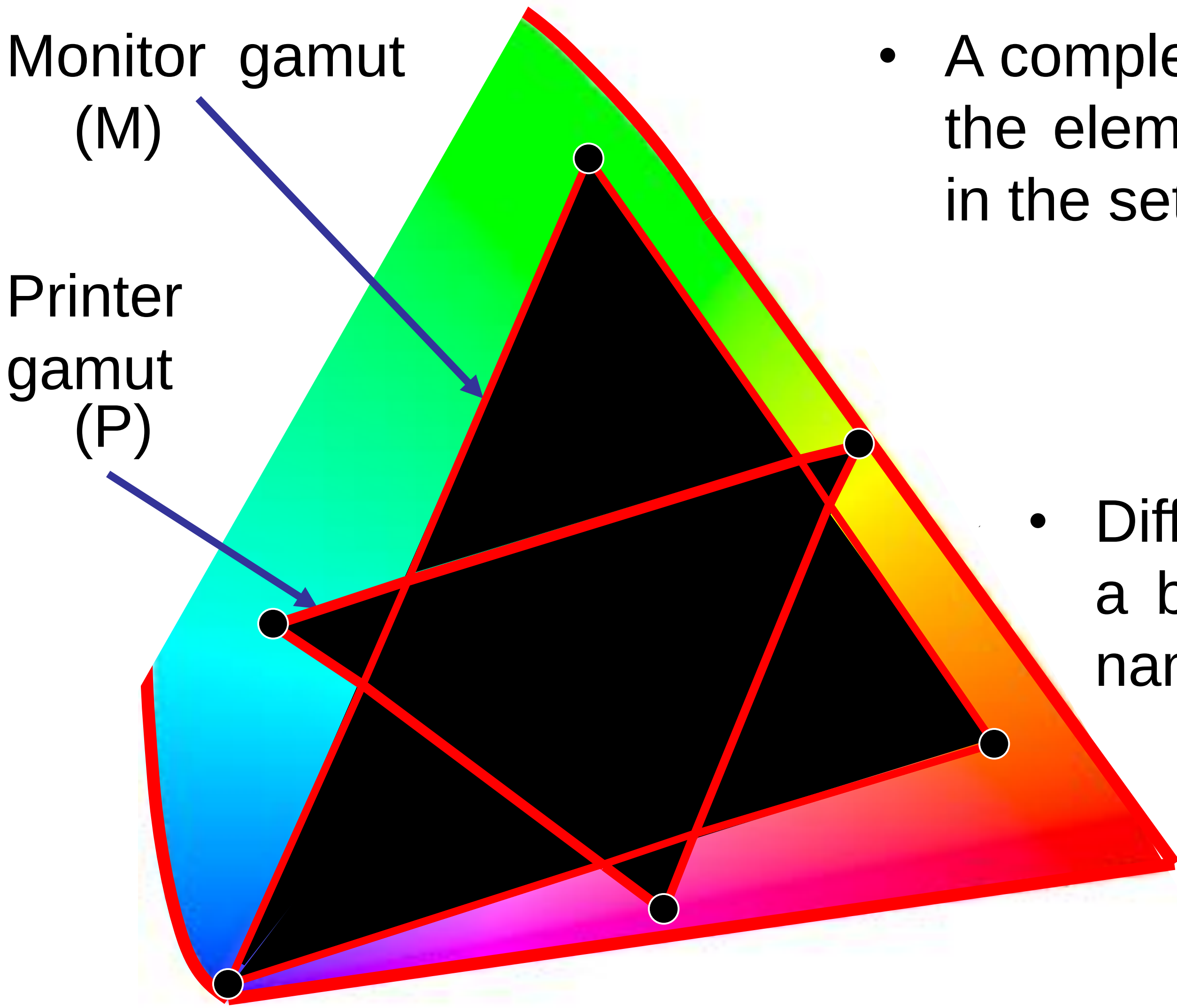
- Formal definition for the symmetric difference of two sets:

  $A \oplus B = \{ x \mid (x \in A \text{ or } x \in B) \text{ and } x \notin A \cap B\}$

  $A \oplus B = (A \cup B) - (A \cap B)$ ← Important!

- Further examples
  - $\{1, 2, 3\} \oplus \{3, 4, 5\} = \{1, 2, 4, 5\}$
  - {New York, Washington} $\oplus$ {3, 4} = {New York, Washington, 3, 4}
  - $\{1, 2\} \oplus \varnothing = \{1, 2\}$
    - The symmetric difference of any set S with the empty set will be the set S

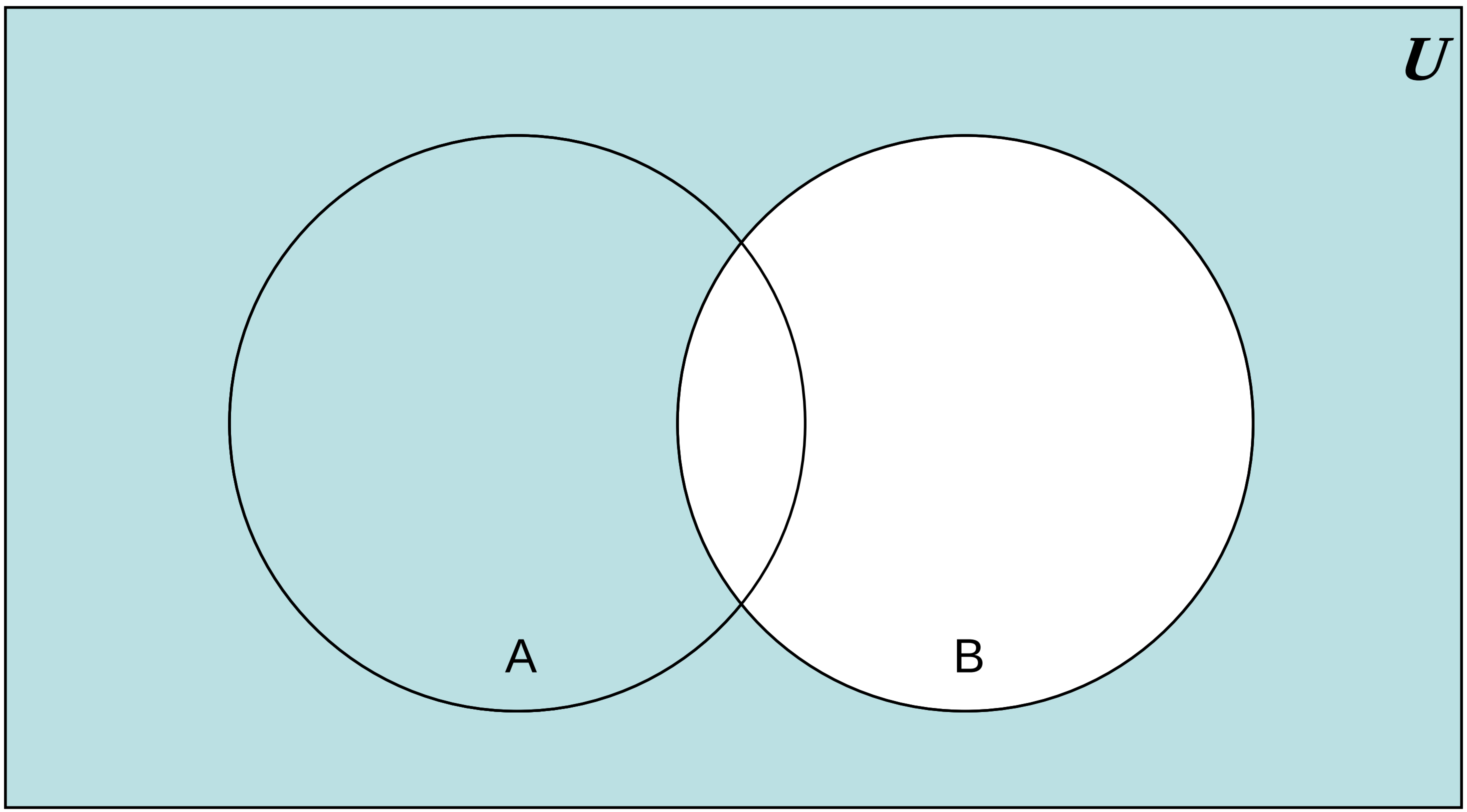
1



- A complement of a set is all the elements that are NOT in the set
- Difference symbol is a bar above the set name: $\overline{P}$ or $\overline{M}$

Complement sets 2

$\overline{A}$
$\overline{B}$



Complement sets 3

- Formal definition for the complement of a set: $\overline{A} = \{ x \mid x \notin A \}$
  - Or $\boldsymbol{U}$ – A, where $\boldsymbol{U}$ is the universal set
- Further examples (assuming $\boldsymbol{U} = \mathbf{Z}$)
  - $\overline{\{1, 2, 3\}} = \{ \ldots, -2, -1, 0, 4, 5, 6, \ldots \}$

Complement sets 4

- Properties of complement sets
  - $\bar{\bar{A}} = A$ Complementation law
  - $A \cup \bar{A} = \boldsymbol{U}$ Complement law
  - $A \cap \bar{A} = \emptyset$ Complement law

Set identities

- Set identities are basic laws on how set operations work
  - Many have already been introduced on previous slides
- Just like logical equivalences!
  - Replace U with $\vee$
  - Replace $\cap$ with $\wedge$
  - Replace $\varnothing$ with F
  - Replace $\boldsymbol{U}$ with T

Set identities: DeMorgan again

- These should look very familiar…

$$\overline{A \cap B} = \overline{A} \cup \overline{B}$$

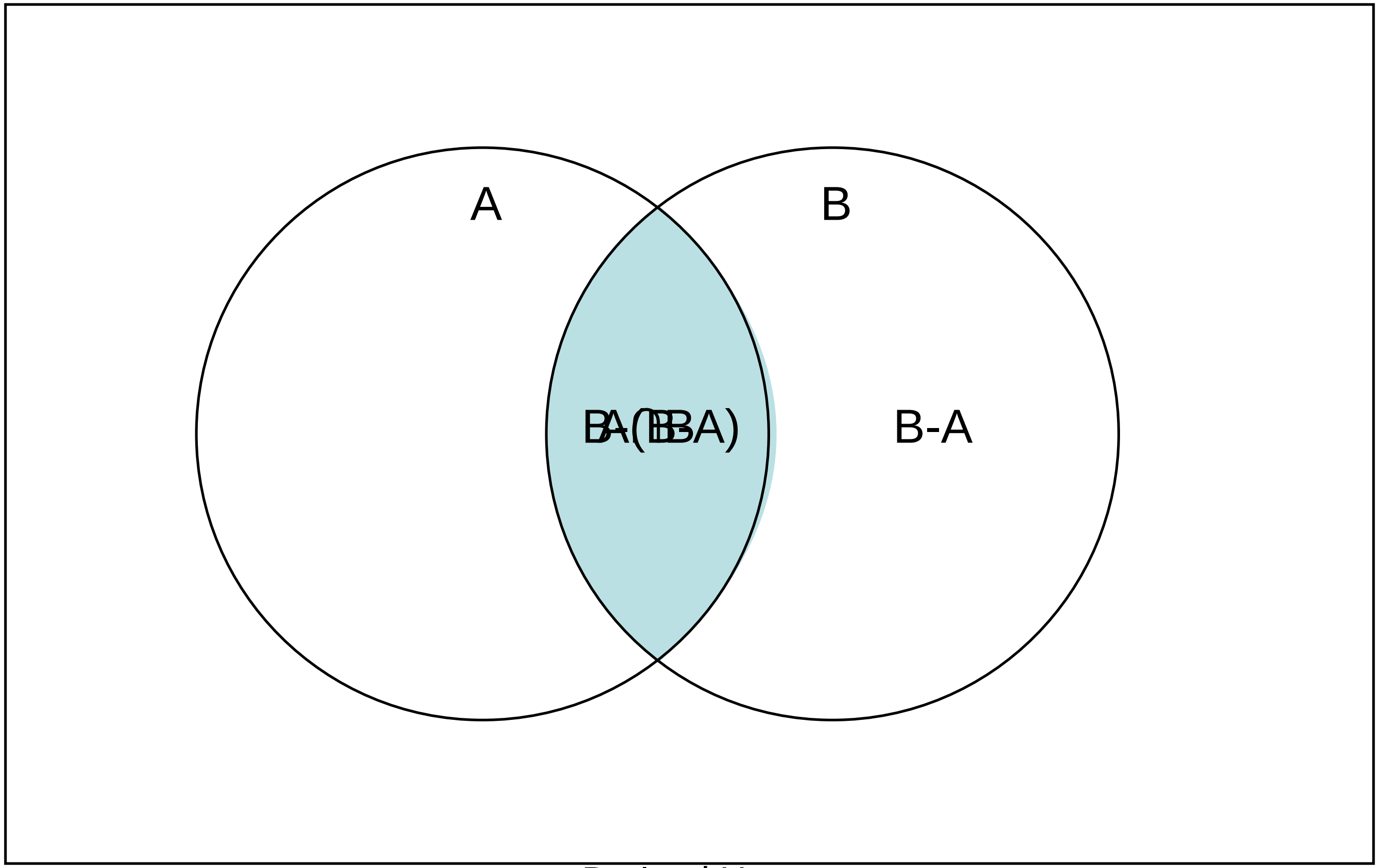
$$\overline{A \cup B} = \overline{A} \cap \overline{B}$$

How to prove a set identity

- For example: $A \cap B = B - (B - A)$
- Four methods:
  - Use the basic set identities
  - Use membership tables
  - Prove each set is a subset of each other
    - This is like proving that two numbers are equal by showing that each is less than or equal to the other
  - Use set builder notation and logical equivalences

What we are going to prove…

$A \cap B = B - (B - A)$



Proof by using basic set identities

- Prove that A∩B=B-(B-A)

$A \cap B = B - (B \cap \overline{A})$	Definition of difference
$= B \cap \overline{(B \cap \overline{A})}$	Definition of difference
$= B \cap (\overline{B} \cup \overline{\overline{A}})$	DeMorgan's law
$= B \cap (\overline{B} \cup A)$	Complementation law
$= (B \cap \overline{B}) \cup (B \cap A)$	Distributive law
$= \varnothing \cup (B \cap A)$	Complement law
$= (B \cap A)$	Identity law
$= A \cap B$	Commutative law

What is a membership table

- Membership tables show all the combinations of sets an element can belong to
  - 1 means the element belongs, 0 means it does not
- Consider the following membership table:

A	B	A U B	A ∩ B	A - B

- The [illegible]
  - Thus, [illegible]

Proof by membership tables

- The following membership table shows that A∩B=B-(B-A)

A	B		B-A	
1	1		0	
1	0		0	
0	1		1	
0	0		0	

- Because the two indicated columns have the same values, the two expressions are identical
- This is similar to Boolean logic!

Proof by showing each set is a subset of the other 1

- Assume that an element is a member of one of the identities
  - Then show it is a member of the other
- Repeat for the other identity

- We are trying to show:
  - $(x \in A \cap B \rightarrow x \in B-(B-A)) \wedge (x \in B-(B-A) \rightarrow x \in A \cap B)$
  - This is the biconditional:
  - $x \in A \cap B \leftrightarrow x \in B-(B-A)$

- Not good for long proofs
- Basically, it’s an English run-through of the proof

Proof by showing each set is a subset of the other 2

- Assume that $x \in B-(B-A)$
  - By definition of difference, we know that $x \in B$ and $x \notin B-A$
- Consider $x \notin B-A$
  - If $x \in B-A$, then (by definition of difference) $x \in B$ and $x \notin A$
  - Since $x \notin B-A$, then only one of the inverses has to be true (DeMorgan's law): $x \notin B$ or $x \in A$
- So we have that $x \in B$ and ($x \notin B$ or $x \in A$)
  - It cannot be the case where $x \in B$ and $x \notin B$
  - Thus, $x \in B$ and $x \in A$
  - This is the definition of intersection
- Thus, if $x \in B-(B-A)$ then $x \in A \cap B$

Proof by showing each set is a subset of the other 3

- Assume that $x \in A \cap B$
  - By definition of intersection, $x \in A$ and $x \in B$
- Thus, we know that $x \notin B-A$
  - B-A includes all the elements in B that are also not in A not include any of the elements of A (by definition of difference)
- Consider B-(B-A)
  - We know that $x \notin B-A$
  - We also know that if $x \in A \cap B$ then $x \in B$ (by definition of intersection)
  - Thus, if $x \in B$ and $x \notin B-A$, we can restate that (using the definition of difference) as $x \in B-(B-A)$
- Thus, if $x \in A \cap B$ then $x \in B-(B-A)$

Proof by set builder notation and logical equivalences 1

- First, translate both sides of the set identity into set builder notation
- Then massage one side (or both) to make it identical to the other
  - Do this using logical equivalences

Proof by set builder notation and logical equivalences 2

$B-(B-A)$	Original statement
$=\{x \mid x \in B \wedge x \notin (B-A)\}$	Definition of difference
$=\{x \mid x \in B \wedge \neg(x \in (B-A))\}$	Negating "element of"
$=\{x \mid x \in B \wedge \neg(x \in B \wedge x \notin A)\}$	Definition of difference
$=\{x \mid x \in B \wedge (x \notin B \vee x \in A)\}$	DeMorgan's Law
$=\{x \mid (x \in B \wedge x \notin B) \vee (x \in B \wedge x \in A)\}$	Distributive Law
$=\{x \mid (x \in B \wedge \neg(x \in B)) \vee (x \in B \wedge x \in A)\}$	Negating "element of"
$=\{x \mid F \vee (x \in B \wedge x \in A)\}$	Negation Law
$=\{x \mid x \in B \wedge x \in A\}$	Identity Law
$= A \bigcap B$	Definition of intersection

Proof by set builder notation and logical equivalences 3

- Why can't you prove it the "other" way?
  - I.e. massage A∩B to make it look like B-(B-A)

- You can, but it's a bit annoying
  - In this case, it's not simplifying the statement

Computer representation of sets 1

- Assume that U is finite (and reasonable!)
  - Let U be the alphabet

- Each bit represents whether the element in U is in the set

- The vowels in the alphabet:
  abcdefghijklmnopqrstuvwxyz
  10001000100000100000100000

- The consonants in the alphabet:
  abcdefghijklmnopqrstuvwxyz
  01110111011111011111011111

Computer representation of sets 2

- Consider the union of these two sets:

$$
\begin{array}{r}
10001000100000100000100000 \\
\vee \underline{01110111011111011111011111} \\
11111111111111111111111111
\end{array}
$$

- Consider the intersection of these two sets:

$$
\begin{array}{r}
10001000100000100000100000 \\
\wedge \underline{01110111011111011111011111} \\
00000000000000000000000000
\end{array}
$$

Subset problems

- Let *A*, *B*, and *C* be sets. Show that:

a) $(A \cup B) \subseteq (A \cup B \cup C)$

b) $(A \cap B \cap C) \subseteq (A \cap B)$

c) $(A-B)-C \subseteq A-C$

d) $(A-C) \cap (C-B) = \varnothing$

Russell’s paradox

- Consider the set:
  - $S = \{ A \mid A \text{ is a set and } A \notin A \}$
- Is S an element of itself?
- Consider:
  - Let $S \in S$
    - Then S can not be in itself, by the definition
  - Let $S \notin S$
    - Then S is in itself by the definition
  - Contradiction!

Russell's paradox

- Consider the set:
  - $S = \{ A \mid A \text{ is a set and } A \notin A \}$
- This shows a problem with set theory!
  - Meaning we can define a set that is not viable
- The solution:
  - Restrict set theory to not include sets which are subsets of themselves

The Halting problem

- Given a program P, and input I, will the program P ever terminate?
  – Meaning will P(I) loop forever or halt?

- Can a computer program determine this?
  – Can a human?

- First shown by Alan Turing in 1936
  – Before digital computers existed!

A few notes

- To “solve” the halting problem means we create a function CheckHalt(P,I)
  - P is the program we are checking for halting
  - I is the input to that program
- And it will return “loops forever” or “halts”
- Note it must work for *any* program, not just some programs
  - Or simple programs

Can a human determine if a program halts?

- Given a program of 10 lines or less, can a human determine if it halts?
  - Assuming no tricks – the program is completely understandable
  - And assuming the computer works properly, of course
- And we ignore the fact that an int will max out at 4 billion

Halting problem tests

```
function haltingTest1()
    print "Alan Turing"
    print "was a genius"
    return
```

```
function haltingTest2()
    for factor from 1 to 10
        print factor
    return
```

```
function haltingTest3()
    while ( true )
        print "hello world"
    return
```

```
function haltingTest4()
    int x = 10
    while ( x > 0 )
        print "hello world"
        x := x + 1
    return
```

Perfect numbers

- Numbers whose divisors (not including the number) add up to the number
  - 6 = 1 + 2 + 3
  - 28 = 1 + 2 + 4 + 7 + 14
- The list of the first 10 perfect numbers:
  6, 28, 496, 8128, 33550336, 8589869056, 137438691328, 2305843008139952128, 2658455991569831744654692615953842176, 191561942608236107294793378084303638130997321548169216
  - The last one was 54 digits!
- All known perfect numbers are even; it's an open (i.e. unsolved) problem if odd perfect numbers exist
- Sequence A000396 in OEIS

Odd perfect number search

```
function searchForOddPerfectNumber()
    int n = 1     // arbitrary-precision integer
    while (true) {
        var int sumOfFactors = 0
        for factor from 1 to n - 1
            if factor is a factor of n
                sumOfFactors = sumOfFactors + factor
        if sumOfFactors = n then
            break
        n = n + 2
    }
    return
```

- Will this program ever halt?

Where does that leave us?

- If a human can't figure out how to do the halting problem, we can't make a computer do it for us

- It turns out that it is impossible to write such a CheckHalt() function
  - But how to prove this?

CheckHalt()’s non-existence

- Consider P(I): a program P with input I
- Suppose that CheckHalt(P,I) exists
  - Tests if P(I) will either “loop forever” or “halt”
- A program is a series of bits
  - And thus can be considered data as well
- Thus, we can call CheckHalt(P,P)
  - It’s using the bytes of program P as the input to program P

CheckHalt()'s non-existence

- Consider a new function:

  Test(P):

    loops forever if CheckHalt(P,P) prints "halts"

    halts if CheckHalt(P,P) prints "loops forever"

- Do we agree that Test() is a valid function?
- Now run Test(Test)
  - If Test(Test) halts…
    - Then CheckHalt(Test,Test) returns "loops forever"…
    - Which means that Test(Test) loops forever
    - Contradiction!
  - If Test(Test) loops forever…
    - Then CheckHalt(Test,Test) returns "halts"…
    - Which means that Test(Test) halts
    - Contradiction!

Why do we care about the halting problem?

- It was the first algorithm that was shown to not be able to exist
  - You can prove an existential by showing an example (a correct program)
  - But it's much harder to prove that a program can *never* exist