Lecture 4
Structures

David Garlan
Carnegie Mellon University

Models of Software Systems
Fall 2016

The Story So Far

- Formal Systems
  - > Syntax
    - » language: alphabet + grammar
    - » deductive system: axioms + rules of inference
  - > Semantics
    - » interpretation
- Propositional logic
- Predicate logic
- Natural deduction

This Lecture

- **Having developed general ways to talk about *properties of things* and to *deduce consequences* about them, we now need a way to represent (or model) the *“things”* themselves.**
- **In the next two lectures we will discuss ways of modeling software system structures mathematically, using sets, tuples, sequences, relations, functions, and records.**

Sets

- A set is a collection of *distinct* objects
- Elements of well-formed sets must have the same type
- There are many ways to express the same set
- Examples:
  - > {1, 3, 5, 7, ... }
  - > {Doug, Kevin, Peter, Eduardo}
  - > {Peter, Doug, Kevin, Eduardo}
  - > {yes, no}
  - > {red, green, blue, green}

= {blue, red, green} ???

- But not: {red, 1, 2, 3 }

Element Types

- **The *type* of an element in a set is defined to be the maximal set in which it is an element**
- **Examples**
  - > **The type of each element in the set {1, 2, 3, 4} is the set of integers (or $\mathbb{Z}$), but not the set of natural numbers ($\mathbb{N}$), since $\mathbb{Z}$ is larger than $\mathbb{N}$.**
  - > **The type of each element in the set {Bill, John, Joe, Mary} is the set of all names of people. (We will see how to define such sets in a minute.)**
- **Because of this definition, the type of elements of a set is unique**
  - > **there can't be more than one maximal set (why?)**

Basic Sets

- We can define new sets of primitive elements
  - > Called *basic sets* or *given sets*
  - > Written [NewSetName]
  - > We know nothing about the elements, except that there is an = operator to compare them and that different basic sets are disjoint (i.e., do not have any common elements)
- Examples
  - > [Student]
  - > [Topic]
  - > [BookIdentifiers]
  - > [Date,Name,Place] – we can define several basic sets in one place

The Integers

- We include one built-in set, the set of integers, denoted by $\mathbb{Z}$
- This is the set {…, -2, -1, 0, 1, 2, …}
- We will also assume we know the usual facts about integers and operations over them
  - Examples $1 < 2$; $2 + 2 = 4$; $3 – 10 = -7$

Variable Declarations

- We can introduce a new variable names: x, y, …
- When we do so we must also say what set is a member of, as follows:

  x: S

- Here x is the variable we are introducing and S is any *expression* that represents a set
- The *type* of x is the type of elements in S
  - > That is, the maximal set containing elements from S
- We have already seen variable declarations in expressions like $\forall x{:}S \bullet \ldots$

Variable Declarations (2)

- To declare a variable with global scope (i.e., not be part of some other expression, such as a quantified expression) we use a vertical bar

$$\begin{array}{|l} x: S \end{array}$$

Use “\axdef” environment in Latex

- This is called an *axiomatic declaration*
- We can optionally add a constraint on the value of x, as follows:

$$\begin{array}{|l} x:S \\ \hline P(x) \end{array}$$

-- where P is a predicate

- Example: Assuming we have declared [Topic]

$$\begin{array}{|l} logic: Topic \\ models: Topic \\ \hline logic \neq models \end{array}$$

Set Definition & Membership

- **Sets can be defined by enumeration**

  SmallOdds == {1, 3, 5, 7}

  Colors == {red, green, blue, blue, green} Strange?

  15-671 == {Elizabeth, Doug, Kevin, Eduardo }

  > **Note the use of "==" for definition (also ≙)**

- **Membership test: 3 $\in$ Odds ; 2 $\notin$ Odds**

  > **Note that $\in$ and $\notin$ are binary infix *predicates* over elements and sets**

Some Predefined Sets

- Some sets have predefined names
  - $\emptyset$ the set with no elements (the "null" or "empty" set) -- also written $\{\}$
  - $\mathbb{Z}$ the set of integers {... -2, -1, 0, 1, 2, ... }
  - $\mathbb{N}$ the set of natural numbers {0, 1, 2, ... }
  - $\mathbb{N}_1$ the set of positive numbers {1, 2, ... }

Set Equality

- **We would like to be able to determine when two sets definitions represent the "same" set**
- **Informally:**

  Two sets are equal *if and only if* they contain the same elements.
- **Formally:**

  $S = R$ if and only if $\forall x{:}T \bullet x \in S \Leftrightarrow x \in R$

  where T is the type of elements in sets S and R
- **Note this only makes sense if the two sets contain elements of the same type**
- **Note this definition implies that the ordering and arity of elements does not matter.** **(Why?)**

Finiteness and Cardinality

- A set with a finite number of elements is a called a finite set
- For a finite set its cardinality, or size, is the number of distinct elements in the set
  - Undefined for infinite sets
- We use the symbol #
- Examples
  - # {1, 2, 4} = 3
  - # { {1,2}, {1,2,3,4,5,6,7} } = ?
  - # {1, 2, 2, 4} = ?

Set Operators

- **We can form new sets from other sets using set operators:**
  - **$\cap$ (intersection), $\cup$ (union), \ (difference)**
- **Examples: Let A == {1,2,3} and B == {3,4,5}**
  - **$A \cap B = \{3\}$**
  - **$A \cup B = \{1,2,3,4,5\}$**
  - **$A \setminus B = \{1,2\}$**
- **Note the use of "=" versus "==".**

Qn: Why?

- **A is a *subset* of B ($A \subseteq B$) if and only if every element of A is also an element of B.**
- **$A = B$ if and only if $A \subseteq B$ and $B \subseteq A$.**

© Garlan,2016

Other definitions

- $x \in A \cap B$ iff $x \in A \wedge x \in B$
- $x \in A \cup B$ iff $x \in A \vee x \in B$
- $x \in A \setminus B$ iff $x \in A \wedge x \notin B$

Recall "iff" means "if and only if" or $\Leftrightarrow$

© Garlan,2016

Set Axioms & Laws

- **Axioms**
  - > **Set membership:** $\forall x: T \bullet x \in \{x\}$
  - > **Empty set:** $\forall x: T \bullet \sim(x \in \emptyset)$
- **Laws (provable from logic and the axioms)**
  - > $A \cap B = B \cap A$
  - > $A \cup B = B \cup A$
  - > $(A \cup B) \cup C = A \cup (B \cup C)$
  - > $(A \cap B) \cap C = A \cap (B \cap C)$
  - > $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$
  - > **and many others – see the textbook**
- **These can be proved using natural deduction**
  - > **In the next lecture we'll see how to do this using an "equational" style**

Set Comprehension

- **Enumerating all of the elements of a set is not always possible**
- **We would like to describe a set in terms of a distinguishing property of its elements.**
  - **Roster == the set of students in 17-651**
  - **Pgh == the set of residents of Pittsburgh**
  - **Primes == the set of integers that are prime**
- **Each element satisfies some property**

  Qn: How can we define such properties?

  Ans: Predicates!
- **This kind of set specification is called *set comprehension***

Set Comprehension (2)

- Simple form of set comprehension

  $\{x : S \mid P(x)\}$

  "the set of x in S that satisfy P(x)", or

  "the set of x in S such that P(x)"

- Examples
  - > natural numbers less than 20: $\{x : \mathbb{N} \mid x < 20\}$
  - > non-negative integers: $\{x : \mathbb{Z} \mid x \geq 0\}$

    Same as $\mathbb{N}$
  - > integers with squares bigger than 100:

    $\{x : \mathbb{Z} \mid x^2 > 100\}$
  - > even integers: $\{x : \mathbb{Z} \mid (\exists\, y : \mathbb{Z} \bullet x = 2y)\}$
  - > all natural numbers: $\{x : \mathbb{N} \mid true\}$
  - > empty set of natural numbers: $\{x : \mathbb{N} \mid false\}$

© Garlan,2016

Power Sets

- The set of all subsets of S is referred to as the *power set* of S and written $\mathbb{P}$ S.
- Examples:
  - > $\mathbb{P}$ {1, 2, 3 } = { ∅,{1},{2},{3},{1,2},{1,3},{2,3},{1,2,3}}
  - > $\mathbb{P}$ $\mathbb{N}$ = { ∅, {0}, {0, 1}, {0,2}, {0,3} ...}
- Power sets can be used to define new types, and can be used in declarations
  - > class-groups == $\mathbb{P}$ Student
  - > Integer-sets == $\mathbb{P}$ $\mathbb{Z}$
  - > If x : $\mathbb{P}$ $\mathbb{Z}$, then x is a *set of* Integers.
  - > Whenever you see " .. : $\mathbb{P}$ .." read "set of"
  - > Alternatively could write: x: Integer-sets

Note the difference between ":" and "∈"

If a set has N elements, how many elements does its power set have?

© Garlan,2016

Finite Subsets

- Sometimes we want to talk about the set of all finite subsets of another set
- We use the symbol $\mathbb{F}$ to represent this set (of sets)
- Example: $\mathbb{F}\ \mathbb{Z}$ represents the set of all finite subsets of $\mathbb{Z}$
  - > Includes sets like {-1,1}
  - > Excludes sets like {1, 3, 5, ….}

Cartesian Products

- **Ordered pairs, triples, etc.**
- **Examples:**
  - > **(2, 3) $\neq$ (3, 2)**
  - > **S == { (2, red), (5, blue), (3, red) }**
- **The set of all tuples constructed from two sets is called a *Cartesian Product*, or just *Product* of those sets.**
  - > **If S and T are those sets, this is written S x T**
- **Examples**
  - > **$\mathbb{N}$ x $\mathbb{N}$ the set of pairs of natural numbers**
  - > **(2, red) $\in$ $\mathbb{N}$ x Color**

More Tuples

- In general, tuples can be described using the following form:
  - > {declarations | predicate}
  - > the predicate is sometimes called an invariant over the state space defined by the declarations
- Examples
  - > $\{x: \mathbb{N}; y: \mathbb{N} \mid y = x + 1\} = \{(0,1), (1,2), (2,3), \ldots\}$ or equivalently:

    $\{x, y: \mathbb{N} \mid y = x + 1\} = \{(0,1), (1,2), (2,3), \ldots\}$
  - > $\{x: \mathbb{F}\,\mathbb{Z}; y: \mathbb{N} \mid y = \#x\} = \{ (\{-1,2,3\}, 3), (\varnothing, 0)\ldots\}$

Relations

- A *relation* is a set of pairs.
- Examples:
  - A == { (1,1), (1,2), (2,2) }
  - B == { (2, red), (5, blue), (3, red) }
  - C == { (David, Jun 1), (Mary, Aug 2), (Bill, Feb 5) }
- The set of all relations over sets S, T is indicated by $S \leftrightarrow T$
  - If $r: T_1 \leftrightarrow T_2$ we call T_1 the *source* and T_2 the *target*
  - $S \leftrightarrow T$ is equivalent to $\mathbb{P}$ (S x T)

If S has 3 elements and T has 2 elements how many does $S \leftrightarrow T$ have?

- Examples:

R1: $\mathbb{N} \leftrightarrow \mathbb{N}$ R2: $\mathbb{N} \leftrightarrow$ Color R3: Person $\leftrightarrow$ Date

Maplet Notation

- **A pair (a,b) can be written using the “maplet” symbol “$\mapsto$”, read “maps to”**
- **Example**
  > **R == {(1,2), (1,3), (2,3)}**

  could be written

  > **R == {1$\mapsto$2, 1$\mapsto$3, 2$\mapsto$3}**

  Note: no parentheses are used around maplet pairs

Relations (2)

- **The *domain* of a relation is the set of first elements. ("dom")**
- **The *range* of a relation is the set of second elements. ("ran")**
- **Examples:**

A == { (1,1), (1,2), (2,2) }

dom A = { 1, 2 } and ran A = { 1, 2 }

even though A $\in$ Z $\leftrightarrow$ Z

B == { (2, red), (5, blue), (3, red) }

dom B = { 2,3,5 } and ran B = { red, blue }

C == { (David, Jun 1}, {Mary, Aug 2}, {Bill, Feb 5) }

dom C = { David, Mary, Bill}

ran C = { Jun 1, Aug 2, Feb 5 }

Functions

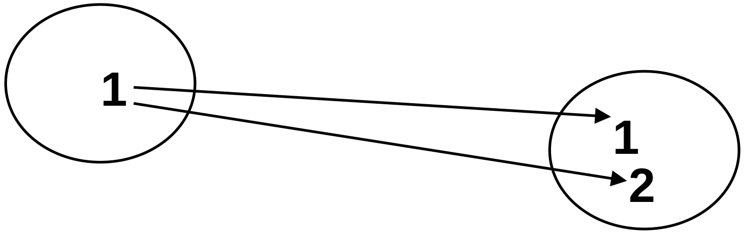
- A *function* is a relation such that no two distinct pairs contain the same first element.
- Equivalently: any element of the source is mapped to at most one element in the target of the relation
- Examples:

  B == { (2, red), (5, blue), (3, red) }

  C == { (David, Jun 1), (Mary, Aug 2), (Bill, Feb 5) }

<u>No</u>:

  A == { (1,1), (1,2), (2,2) }

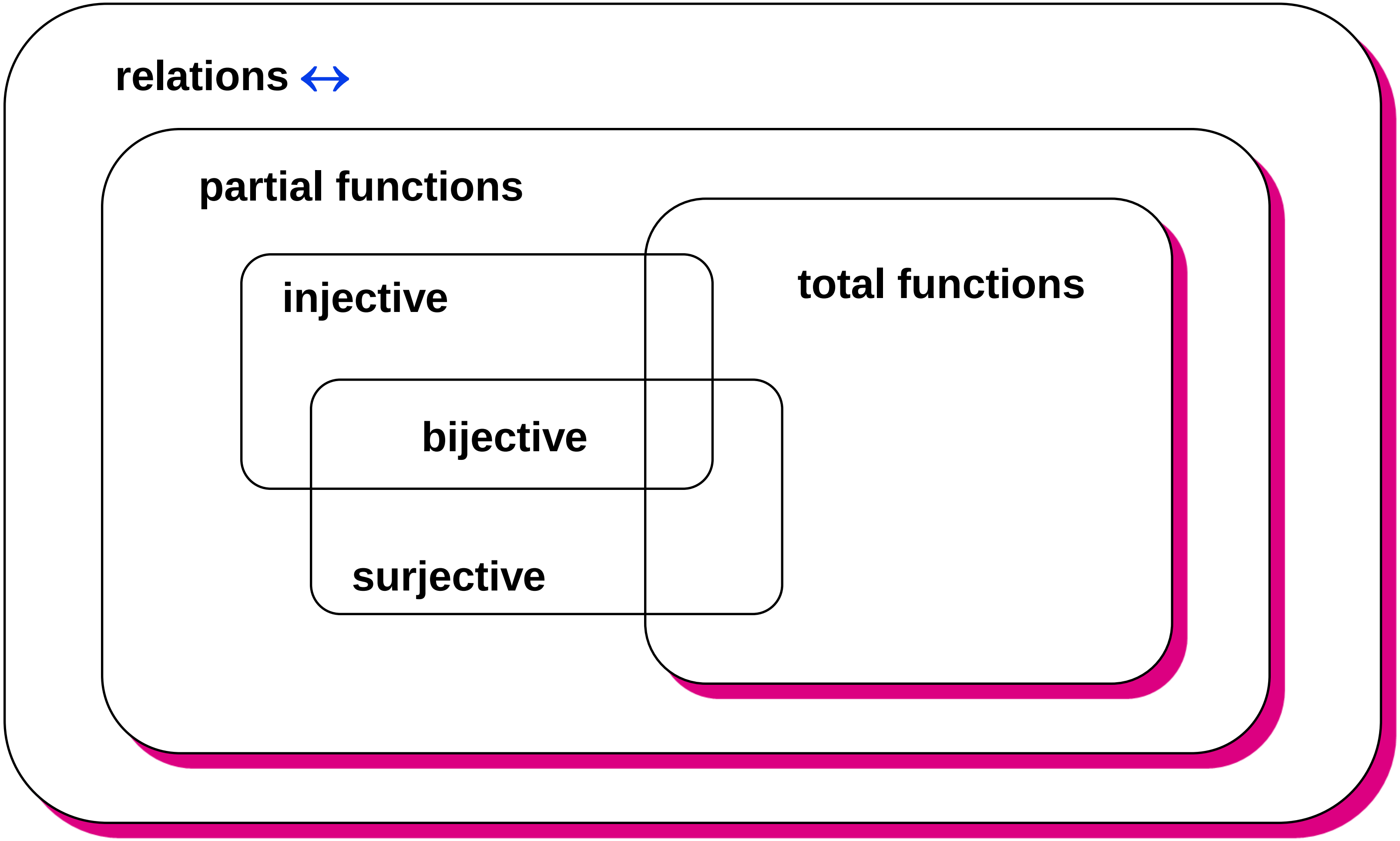


Some special cases

Suppose f: $A \leftrightarrow B$

1. f is a function defined for *all* values of A
   - we say f is a "total" function, and write $A \rightarrow B$
2. f is a function defined for *some* values of A
   - we say f is a "partial" function, and write $A \nrightarrow B$
3. f is a function defined for a *finite set* of values of A
   - we say f is a "finite" function, and write $A \mathrel{+\!\!\!+\!\!\!\rightarrow} B$
4. f is a function for which no element in ran(f) is associated with more than one element in dom(f)
   - we say f is a "one-to-one" or "injective" function, and write $A \rightarrowtail B$
5. f is a function whose range is B
   - we say f is an "onto" or "surjective" function, and write $A \twoheadrightarrow B$
6. f is both one-to-one and onto
   - we say f is a "bijection", and write $A \rightarrowtail\!\!\!\!\!\twoheadrightarrow B$

Special Cases (2)



Relations/Functions as Sets

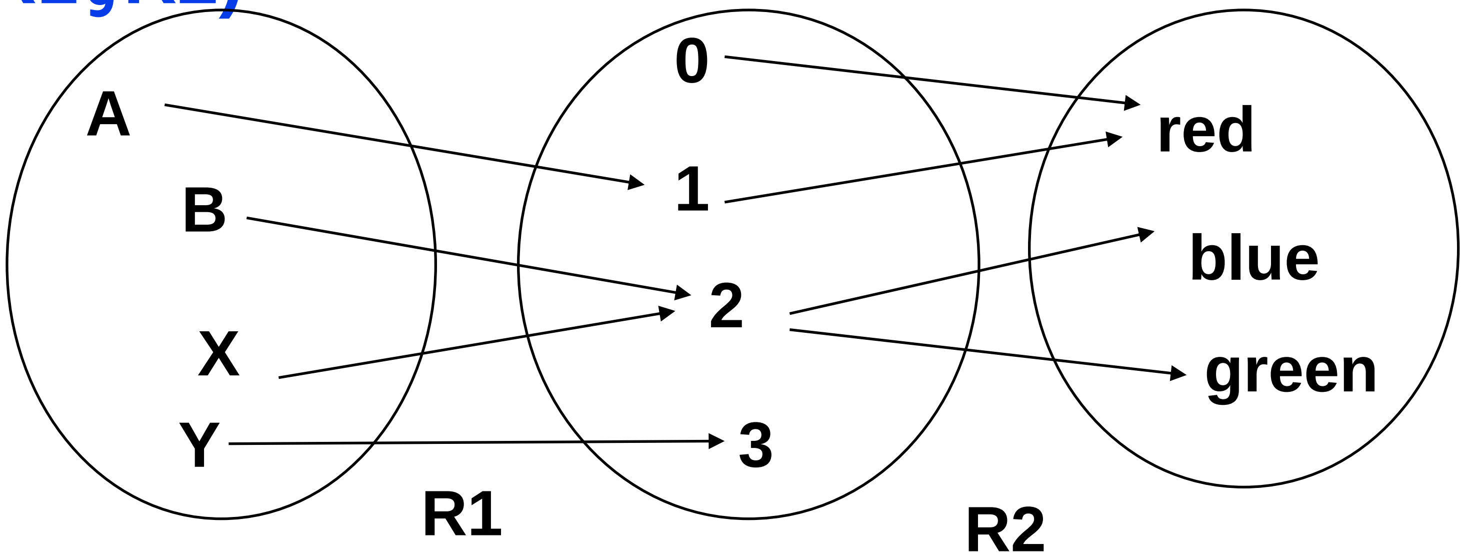
- **Since relations are just sets (of pairs) we can apply set operators to them.**
- **Example:**
  - > **R1 == {(1,red), (2,blue)}**
  - > **R2 == {(3,green), (2,blue)}**
  - > **R1 ∪ R2 = {(1,red), (2,blue), (3,green)}**
  - > **# R1 = 2**

Question: is the union/intersection of two functions a function?

Question: how would you prove it?

Relational Composition

- If the target of one relation is the source of another we can form the composition (R1 ⨾ R2)



Note: in some texts ran(R1) must be the same as dom(R2) for ⨾ to be well-defined

Overwriting

- Frequently we will want to create a new function similar to an old one but which has a different result for one or more values
- The *overriding* operator, ⊕, does this:
- Example:
  - > f == {(1,red), (2,blue), (3,green)}
  - > g == {(1,pink), (4,mauve)}
  - > f ⊕ g = {(1,pink), (2,blue), (3,green), (4,mauve)}

Note: replacement only occurs on values in the first function's domain

Domain/Range Restriction

- Suppose
  > R is a relation in S $\leftrightarrow$ T (i.e., R: S $\leftrightarrow$ T)
  > s is a set of elements with type of elements of S
  > t is a set of elements with type of elements of T
- Operators
  > ***Domain restriction:*** **s $\lhd$ R is the set of tuples in R whose first element is in s**
  > ***Range restriction:*** **R $\rhd$t is the set of tuples in R whose second element is in t**
  > ***Domain anti-restriction:*** **s $⩤$R is the set of tuples in R whose first element is NOT in s**
  > ***Range anti-restriction:*** **R $⩥$t is the set of tuples in R whose second element is NOT in t**

Example of Domain/Range Restriction

Suppose:

$R == \{(2, red),(5, blue),(3, red),(5, pink),(4, azure)\}$

$Primary == \{red, blue, green\}$

$Even == \{n: \mathbb{N} \mid \exists\, k: \mathbb{N} \bullet n = 2k\}$

Then:

$Even \lhd R = \{(2, red), (4, azure)\}$

$R \rhd Primary = \{(2, red),(5, blue),(3, red)\}$

Relations and Functions

- We can turn any relation into an equivalent function, and vice versa in the following way:

  $r: S \leftrightarrow T$ {(2,red), (2,blue), (3,red)} #r = 3

  $f: S \rightarrow \mathbb{P}\, T$ {(2,{red,blue}), (3,{red})} #f = 2

Other Set Building Definitions

- **There is a rich collection of other auxiliary definitions.**
- **We'll introduce these as we proceed through the class**
- **Some examples:**
  - $\mathbb{N}_1$ **the set of positive natural numbers**
  - .. **the "between" operator: 2..5 = {2,3,4,5}**

Records

- A record is similar to a tuple, except that sub-components have names
- Example:

  $[\ x{:}\ \mathbb{P}\ \mathbb{Z};\ y, z{:}\ \mathbb{N}\ ]$

  defines the set of records with three components
- We call the labels on components *fields*

  x, y, and z are fields in the above example
- In contrast to tuples, ordering is not important, so

  $[\ x{:}\ \mathbb{P}\ \mathbb{Z};\ y, z{:}\ \mathbb{N}\ ] = [\ z, y{:}\ \mathbb{N}\ ;\ x{:}\ \mathbb{P}\ \mathbb{Z}\ ]$
- Note that a set of records is empty (i.e., has no elements) if one of the field sets is empty

Record Types (continued)

- **Each record type has a set of *projection functions,* one for each field**
  - > **Example: Let r : [x: $\mathbb{Z}$; y, z: $\mathbb{P}\,\mathbb{N}$] be an element of the record type with fields x, y, and z.**
    Then r.x is the value of the component x;

    .x : [x: $\mathbb{Z}$; y, z: $\mathbb{P}\,\mathbb{N}$] $\rightarrow \mathbb{Z}$

- **To build a record from individual field values, we use the following notation:**

  [x= 4 ; y={1,2,3} ; z={1,5}]

  Again, ordering is not significant

Summary of Set-Building Operators

If R, S, and T are sets

$S \cap R$ (intersection), $\cup$ (union), \ (difference)	
$\mathbb{P}$ S	(power set)
{x: S | P(x)}	(set comprehension}
S x R	(cartesian product)
$S \leftrightarrow R$	(relations)
$S \rightarrow R$	(functions), etc.
$\fatsemi$	(relational composition)
$\oplus$	(overwriting)
[a: S; b: R]	(records)

Next Time

- **Reading**
  - > **Other proof techniques and styles.**
  - > **Structural induction.**