UNIVERSITY OF Southampton

Logic and set operations

Michael Butler

8 March 2017

Predicate Logic

Basic predicates: $\boxed{x \in S}$ $\boxed{S \subseteq T}$ $\boxed{x \leq y}$

Predicate operators:

- Negation: $\boxed{\neg P}$ P does not hold
- Conjunction: $\boxed{P \ \wedge \ Q}$ both P and Q hold
- Disjunction: $\boxed{P \ \vee \ Q}$ either P or Q holds
- Implication: $\boxed{P \implies Q}$ if P holds, then Q holds
- Universal Quantification: $\boxed{\forall x \cdot P}$ P holds for all x.
- Existential Quantification: $\boxed{\exists x \cdot P}$ P holds for some x.

Defining Set Operators with Logic

Predicate	Definition
$x \notin S$	$\neg\,(x \in S)$
$x \ \in \ S \cup T$	$x \in S \ \ \lor \ \ x \in T$
$x \ \in \ S \cap T$	$x \in S \ \ \land \ \ x \in T$
$x \ \in \ S \setminus T$	$x \in S \ \ \land \ \ x \notin T$
$S \ \subseteq \ T$	$\forall x \cdot x \in S \ \implies \ x \in T$

Relations and Functions

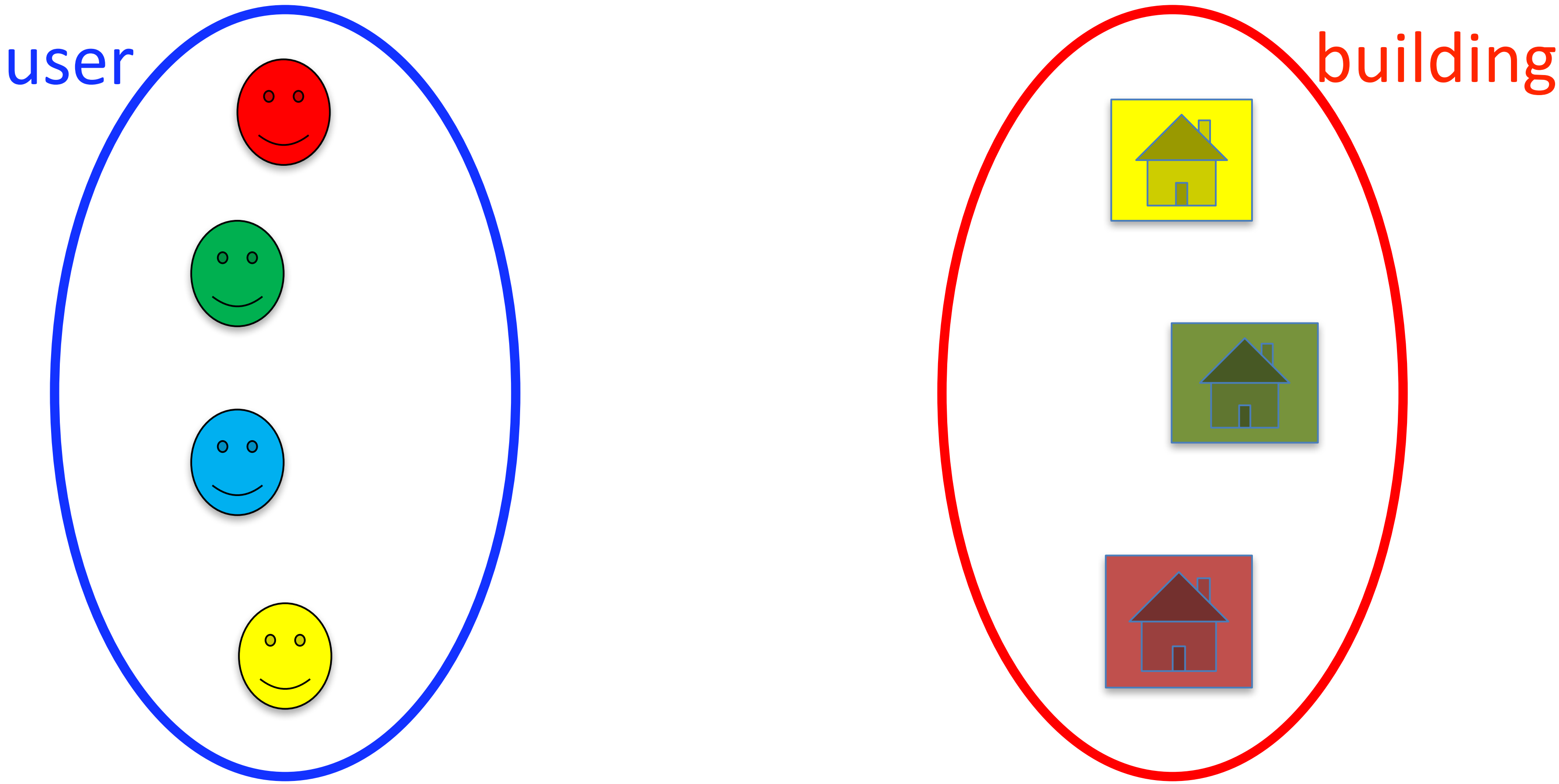
Michael Butler

9 March 2017

Requirements for a Buildings Access System

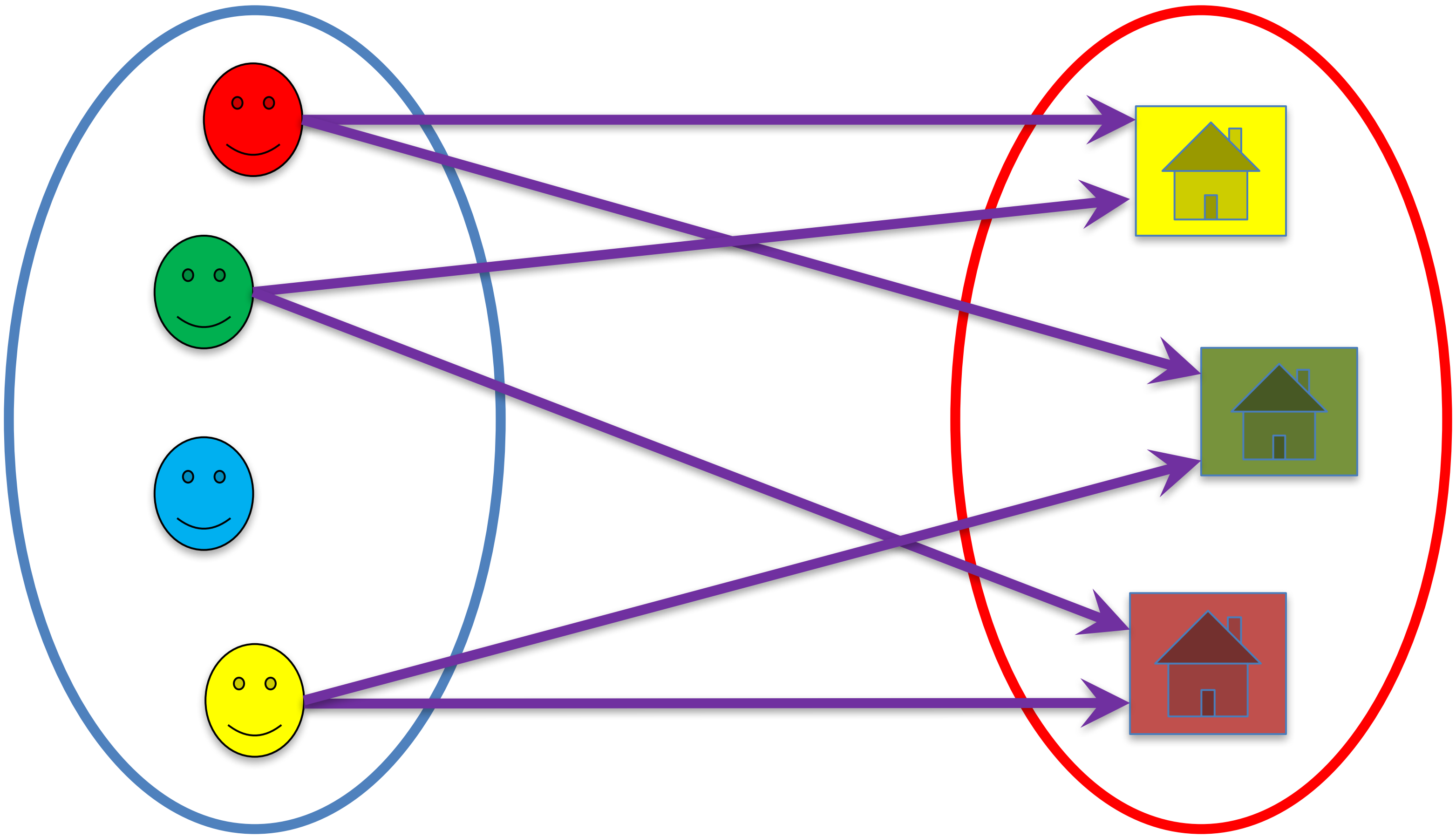
- Specify a system that controls access to a collection of buildings.
- Registered users will have access permission to enter certain buildings.
- A user can only enter buildings that they have access permission for.
- The system should keep track of the location of users.
- The system should manage registration and access permission for users.

Users and Buildings



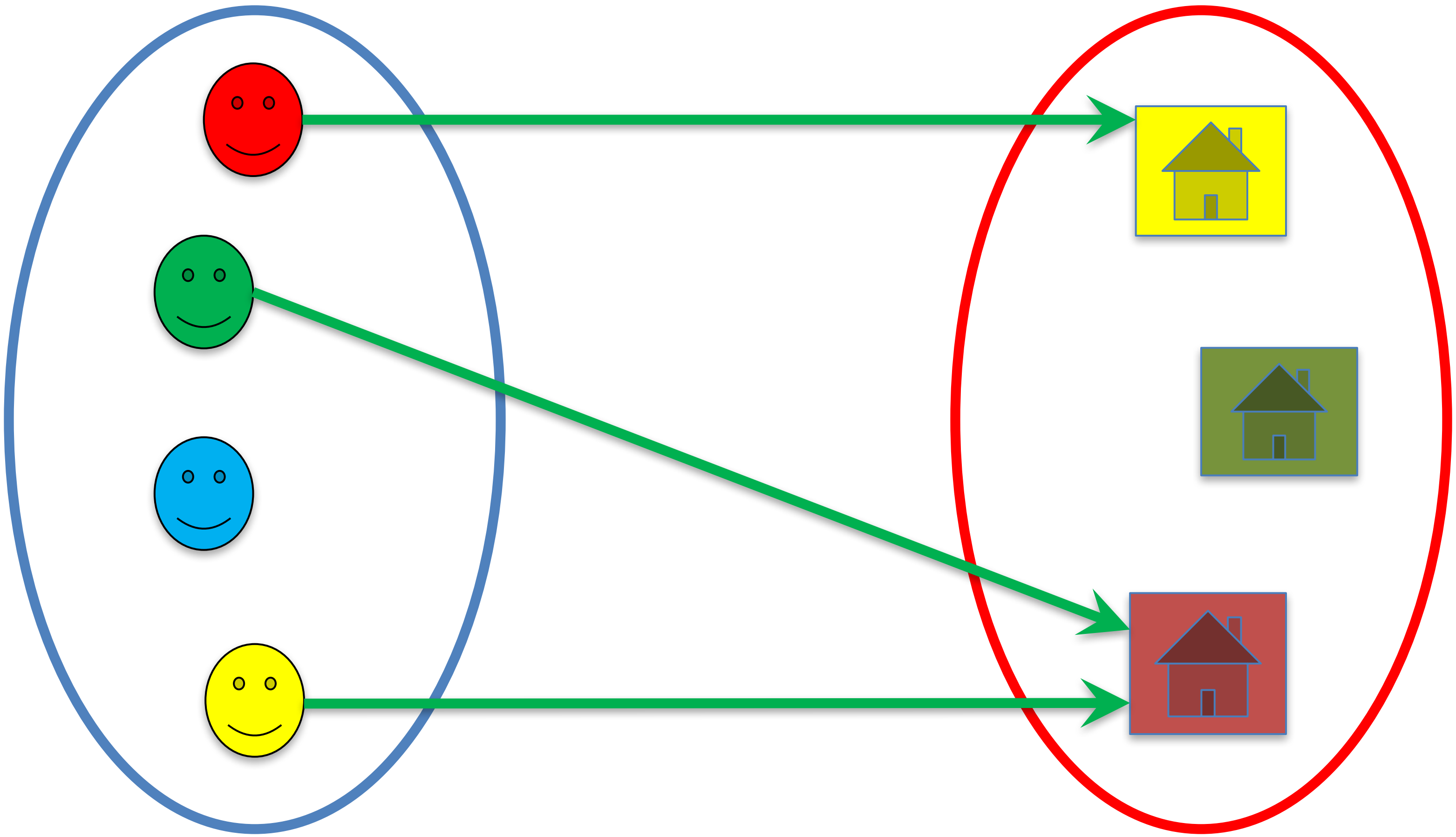
Carrier sets: USER BUILDING

Permission



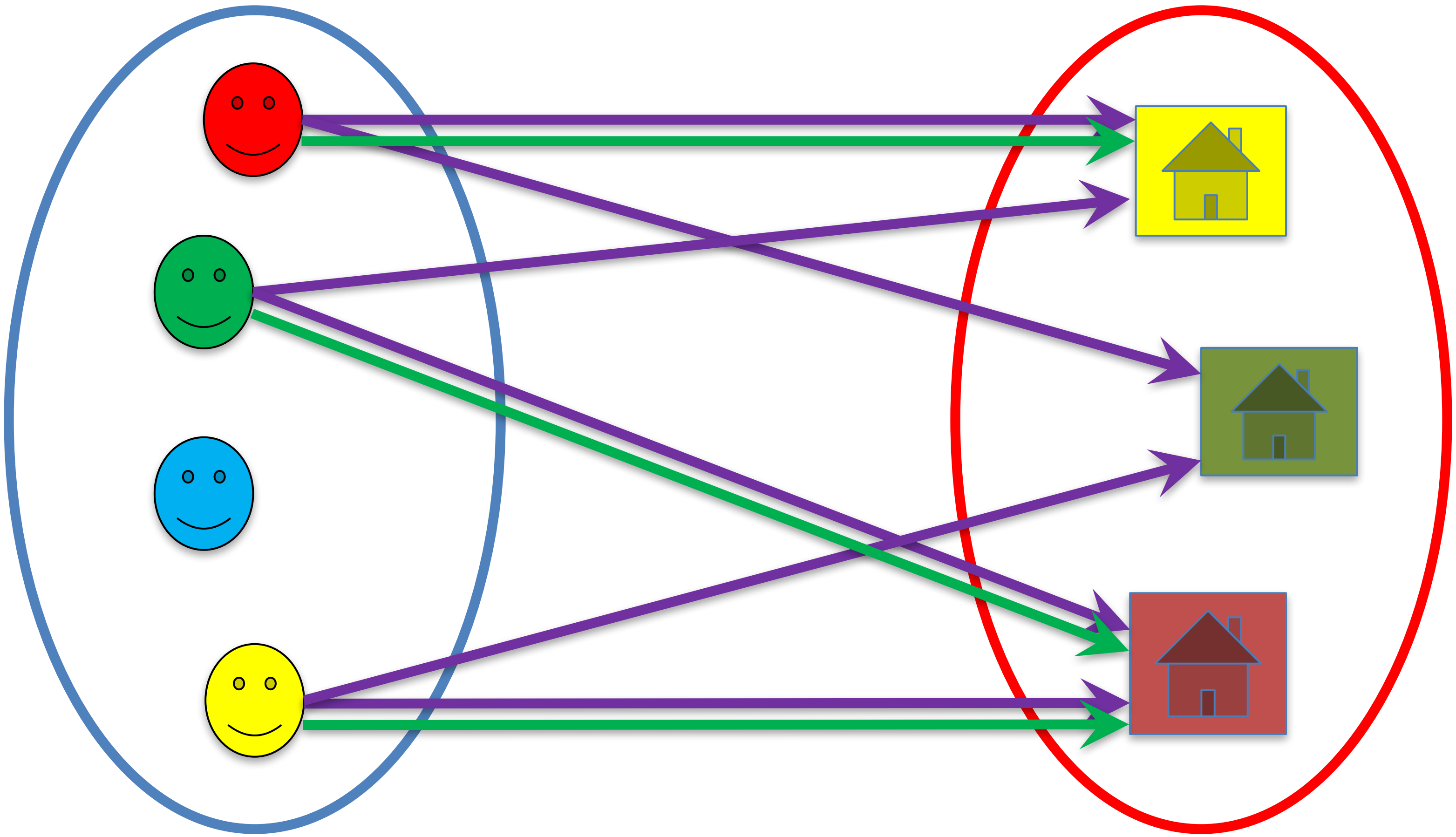
Many-to-many relation

Location



Many-to-one relation

Location conforms to Permission



Location $\subseteq$ Permission

Ordered Pairs and Cartesian Products

An ordered pair is an element consisting of two parts:

a first part and a second part.

An ordered pair with first part x and second part y is written: $\boxed{x \mapsto y}$

The Cartesian product of two sets is the set of pairs whose first part is in S and second part is in T.

The Cartesian product of S with T is written: $\boxed{S \times T}$

Cartesian Products: Definition and Examples

Defining Cartesian product:

Predicate	Definition
$x \mapsto y \quad \in \quad S \times T$	$x \in S \quad \wedge \quad y \in T$

Examples:

$$\{a, b, c\} \times \{1, 2\} \quad = \quad \{\ a \mapsto 1,\ \ a \mapsto 2,\ \ b \mapsto 1,\ \ b \mapsto 2,\ \ c \mapsto 1,\ \ c \mapsto 2\ \}$$

$$\{a, b, c\} \times \{\} \quad = \quad ?$$

$$\{\ \{a\},\ \{a, b\}\ \} \ \times\ \{1, 2\} \quad = \quad ?$$

Cartesian Products: Definition and Examples

Defining Cartesian product:

Predicate	Definition
$x \mapsto y \quad \in \quad S \times T$	$x \in S \quad \wedge \quad y \in T$

Examples:

$$\{a, b, c\} \times \{1, 2\} \quad = \quad \{\ a \mapsto 1,\ a \mapsto 2,\ b \mapsto 1,\ b \mapsto 2,\ c \mapsto 1,\ c \mapsto 2\ \}$$

$$\{a, b, c\} \times \{\} \quad = \quad \{\}$$

$$\{\ \{a\},\ \{a, b\}\ \} \ \times\ \{1, 2\} \quad = \quad \{\ \{a\} \mapsto 1,\ \{a\} \mapsto 2,\ \{a, b\} \mapsto 1,\ \{a, b\} \mapsto 2\ \}$$

Cartesian Product is a Type Constructor

$S \times T$ is a new type constructed from types S and T.

Cartesian product is the type constructor for ordered pairs.

Given $x \in S, \quad y \in T,$ we have

$$\boxed{x \mapsto y \quad \in \quad S \times T}$$

$$4 \mapsto 7 \quad \in \quad ?$$

$$\{5, 6, 3\} \mapsto 4 \quad \in \quad ?$$

$$\{\ 4 \mapsto 8,\ 3 \mapsto 0,\ 2 \mapsto 9\ \} \ \in \quad ?$$

Cartesian Product is a Type Constructor

$S \times T$ is a new type constructed from types S and T.

Cartesian product is the type constructor for ordered pairs.

Given $x \in S$, $y \in T$, we have

$$\boxed{x \mapsto y \quad \in \quad S \times T}$$

$$4 \mapsto 7 \quad \in \quad \mathbb{Z} \times \mathbb{Z}$$

$$\{5, 6, 3\} \mapsto 4 \quad \in \quad \mathbb{P}(\mathbb{Z}) \times \mathbb{Z}$$

$$\{\ 4 \mapsto 8,\ 3 \mapsto 0,\ 2 \mapsto 9\ \} \quad \in \quad \mathbb{P}(\mathbb{Z} \times \mathbb{Z})$$

Classification of Types in Event-B

Types are sets

Simple Types:

- $\mathbb{Z}$, $\mathbb{B}$
- Basic types (e.g., *WORD*, *NAME*)

Constructed Types:

- $\mathbb{P}(S)$
- $S \times T$

$\mathbb{P}(S)$ is a type that is constructed from S.

$S \times T$ is a type that is constructed from S and T.

Sets of Order Pairs

A database can be modelled as a set of ordered pairs:

$$\begin{aligned} \mathit{directory} \quad = \quad \{ \; & \mathit{mary} \mapsto 287573, \\ & \mathit{mary} \mapsto 398620, \\ & \mathit{john} \mapsto 829483, \\ & \mathit{jim} \mapsto 398620 \; \} \end{aligned}$$

directory has type

$$\mathit{directory} \quad \in \quad \mathbb{P}(\mathit{Person} \times \mathit{PhoneNum})$$

Relations

A relation is a set of ordered pairs.

A relation is a common modelling structure so Event-B has a special notation for it:

$$\boxed{T \leftrightarrow S} \quad = \quad \mathbb{P}(T \times S)$$

So we can write:

$$directory \quad \in \quad Person \leftrightarrow PhoneNum$$

Do not confuse the arrow symbols:

$\leftrightarrow$ combines two sets to form a set.

$\mapsto$ combines two elements to form an ordered pair.

Domain and Range

$$
\begin{aligned}
\mathit{directory} \quad = \quad \{\; & \mathit{mary} \mapsto 287573, \\
& \mathit{mary} \mapsto 398620, \\
& \mathit{john} \mapsto 829483, \\
& \mathit{jim} \mapsto 398620 \;\}
\end{aligned}
$$

$$
\begin{aligned}
\mathrm{dom}(\mathit{directory}) \quad &= \quad \{\mathit{mary}, \mathit{john}, \mathit{jim}\} \\
\mathrm{ran}(\mathit{directory}) \quad &= \quad \{287573, 398620, 829483\}
\end{aligned}
$$

Domain and Range Definition

- The domain of a relation R is the set of first parts of all the pairs in R, written $\boxed{dom(R)}$
- The range of a relation R is the set of second parts of all the pairs in R, written $\boxed{ran(R)}$

Predicate	Definition
$x \in dom(R)$	$\exists y \cdot \ x \mapsto y \ \in \ R$
$y \in ran(R)$	$\exists x \cdot \ x \mapsto y \ \in \ R$

Telephone Directory Model

- Phone directory relates people to their phone numbers.
- Each person can have zero or more numbers.
- People can share numbers.

context *PhoneContext*
sets *Person* *PhoneNum*
end

machine *PhoneBook*
variables *dir*
invariants $dir \in Person \leftrightarrow PhoneNum$

initialisation $dir := \{\}$

Extending the Directory

Add an entry to the directory:

$$
\begin{array}{lcl}
AddEntry & \widehat{=} & \textbf{any } p, n \textbf{ where} \\
 & & \quad p \in Person \\
 & & \quad n \in PhoneNum \\
 & & \textbf{then} \\
 & & \quad dir \ := \ dir \cup \{p \mapsto n\} \\
 & & \textbf{end}
\end{array}
$$

Relational Image

$$
\begin{aligned}
directory \quad = \quad \{ \; & mary \mapsto 287573, \\
& mary \mapsto 398620, \\
& john \mapsto 829483, \\
& jim \mapsto 398620 \; \}
\end{aligned}
$$

Relational image examples:

$$
\begin{aligned}
directory[\,\{mary\}\,] \quad &= \quad \{\, 287573,\; 398620 \,\} \\
directory[\,\{\, john, jim \,\}\,] \quad &= \quad \{\, 829483,\; 398620 \,\}
\end{aligned}
$$

Relational Image Definition

Assume $R \in S \leftrightarrow T$ and $A \subseteq S$

The relational image of set A under relation R is written $\boxed{R[A]}$

Predicate	Definition
$y \in R[A]$	$\exists x \cdot x \in A \wedge x \mapsto y \in R$

Modelling Queries using Relational Image

Determine all the numbers associated with a person in the directory:

$$
\begin{array}{lll}
GetNumbers & \hat{=} & \textbf{any}\ p, result\ \textbf{where} \\
& & \quad p \in Person \\
& & \quad result = dir[\ \{p\}\ ] \\
& & \textbf{end}
\end{array}
$$

Determine all the numbers associated with a set of people:

$$
\begin{array}{lll}
GetMultiNumbers & \hat{=} & \textbf{any}\ ps, result\ \textbf{where} \\
& & \quad ps \subseteq Person \\
& & \quad result = dir[\ ps\ ] \\
& & \textbf{end}
\end{array}
$$

Event-B Lecture Notes

- For overview of modelling with sets in Event-B see Notes:
- http://eprints.soton.ac.uk/402239/
- (also linked from COMP1216 web page)

- Read Sections 1-6