UNIVERSITY OF Southampton
School of Electronics and Computer Science

11. A Very Strict Society – An Exercise in Modelling Using Relations

Thai Son Hoang
(example from Jean-Raymond Abrial)

ECS, University of Southampton, U.K.

COMP1216 - Software Modelling and Design
14th March 2019

Objectives

- Different type of relations functions
- An example of formalisation with relations.
- Example from Jean-Raymond Abrial

Different properties of relations

Consider a binary relation $r : S \leftrightarrow T$

- Totality Every element of S has *at least one relationship*

$$dom(r) = S$$

- Surjectivity Every element of T has *at least one relationship*.

$$ran(r) = T$$

- Functional Every element of S has *at most one relationship*

$$\forall x, y1, y2 \cdot x \mapsto y1 \in r \land y1 \neq y2 \Rightarrow x \mapsto y2 \notin r$$

- Injectivity Every element of T has *at most one relationship*

$$\forall x1, x2, y \cdot x1 \mapsto y \in r \land x1 \neq x2 \Rightarrow x2 \mapsto y \notin r$$

Note how totality vs. surjective, functional vs. injectivity are symmetric properties.

Type of Relations in Event-B

UNIVERSITY OF Southampton
School of Electronics and Computer Science

	None	T	S	T $\wedge$ S
None	$\rel$	$\trel$	$\srel$	$\strel$
F	$\pfun$	$\tfun$	$\psur$	$\tsur$
I				
F $\wedge$ I	$\pinj$	$\tinj$		$\tbij$

Table: Event-B relations

- T means the relation is total,
- S means the relation is surjective,
- F means the relation is functional,
- I means the relation is injective.

E.g., column T $\wedge$ S and row F denotes *total surjective function*, i.e., $\tsur$

Example: a Very Strict Society

- Every person is either a man or a woman
- But no person can be a man and a woman at the same time
- Only women have husbands, who must be a man
- Woman have at most one husband
- Likewise, men have at most one wife
- Moreover, mother are married women

Formal Representation (1/2)


```
context c0
sets PERSON // The set of people
constants
  men // The set of men
  women // The set of women
  husband // The husband relationship
  mother // The mother relationship
axioms
  /*
   * – Every person is either a man or a woman.
   * – But no person can be a man and a woman at the same time.
   */
  @axm1: men ⊆ PERSON
  @axm2: women = PERSON \ men
```

Formal Representation (2/2)

UNIVERSITY OF Southampton
School of Electronics and Computer Science

```
/*
 * – Only women have husbands, who must be a man.
 * – Woman have at most one husband.
 * – Likewise, men have at most one wife.
 */
@axm3: husband ∈ women ⤔ men

// Moreover, mother are married women.
@axm4: mother ∈ PERSON → dom(husband)

end
```

Defining New Concepts

wife, spouse, father

UNIVERSITY OF Southampton
School of Electronics and Computer Science

context c1
extends c0
constants
  wife // The wife relationship
  spouse // The spouse relationship
  father // The father relationship
axioms
  // A is the wife of B if B is the husband of A
  @def−wife: $\mathsf{wife} = \mathsf{husband}^{-1}$
  // A is the spouse of B if A is either the husband or the wife of B.
  @def−spouse: $\mathsf{spouse} = \mathsf{husband} \cup \mathsf{wife}$
  // A is the father of B if A is the husband of the mother of B.
  @def−father: $\mathsf{father} = \mathsf{mother};\mathsf{husband}$
  // Theorems about father and mother relationships
  theorem @thm−mother_father_wife: $\mathsf{mother} = \mathsf{father};\mathsf{wife}$
  theorem @thm−spouse_symmetric: $\mathsf{spouse} = \mathsf{spouse}^{-1}$
  theorem @thm−father_mother: $\mathsf{father};\mathsf{father}^{-1} = \mathsf{mother};\mathsf{mother}^{-1}$
  theorem @thm−father_not_mother: $\mathsf{father};\mathsf{mother}^{-1} = \varnothing$
  theorem @thm−mother_not_father: $\mathsf{mother};\mathsf{father}^{-1} = \varnothing$
end

Defining New Concepts

parents, children, daughter, sibling


```
context c2
extends c1
constants
 parents // The parents relationship
 children // The children relationship
 daughter // The daughter relationship
 sibling // The sibling relationship
axioms
 // A is a parent of B if A is the mother or the father of B.
 @def−parents: parents = mother ∪ father
 // A is a child of B if B is a parent of A.
 @def−children: children = parents⁻¹
 // A is a daughter of B if A is a child of B who is a woman.
 @def−daughter: daughter = children ▷ women
 // A is a sibling of B if A is a child of a parent of B who is not A.
 @def−sibling: sibling = (parents;children) \ id
 // Theorems
 theorem @thm−sibling_symmetric: sibling = sibling⁻¹
 theorem @thm−father_mother_children: father;children = mother;children
end
```

Defining New Concepts

brother, sibling-in-law, nephew-or-niece, uncle-or-aunt, cousin

```
context c3
extends c2
constants
 brother // The brother relationship
 sibling_in_law // The sibling−in−law relationship
 nephew_or_niece // The nephew−or−niece relationship
 uncle_or_aunt // The uncle−or−aunt relationship
 cousin // The cousin relationship
axioms
 @def−brother: brother = sibling ▷ men
 @def−sibling_in_law: sibling_in_law = (sibling;spouse) ∪ (spouse;sibling)
 theorem @thm−sibling_in_law_symmetric: sibling_in_law =
         sibling_in_law^-1
 @def−nephew_or_niece: nephew_or_niece = (sibling ∪ sibling_in_law);
         children
 @def−uncle_or_aunt: uncle_or_aunt = nephew_or_niece^-1
 @def−cousin: cousin = uncle_or_aunt;children
 theorem @thm−cousin_symmetric: cousin = cousin^-1
end
```

Summary

- Modelling using Relations
  - Different types of relations
  - Inverse relation
  - Domain/range retriction/subtraction
  - Forward composition

Further Reading

UNIVERSITY OF Southampton
School of Electronics and Computer Science

- Jean-Raymond Abrial. *Modeling in Event-B: System and Software Engineering*.
  Cambridge University Press, 2010 (Chapter 9)