Модели на софтуерни системи

доц. Олга Георгиева

СУ, ФМИ

катедра “Софтуерни технологии”

Модели на софтуерни системи

Лекция 3: Множества - представяне в Z нотацията.
Дефиниции в Z нотацията

Теории

Можем да направим много само с логиката.

Често, обаче, се налага да въведем специални аксиоми, за да опишем специфични явления.

След това се интересуваме от следствията, резултатите, заключенията от тези аксиоми (давайки други правила за дедукция).

Тези следствия се наричат теория.

Примери: Някои общи теории, които се представят в курса.

- Теорията на множествата (set theory)
- Теорията на последователностите (theory for sequences)
- Теорията на едновременните/паралелните процеси (theory for concurrent processes)

Други теории, които се изграждат за характеризиране на софтуерни системи

- *Пр*: theory of telephones for a switching system
- *Пр*: theory of devices and event processing for real-time system

Множества, релации, функции

Ще бъдат представени едни от най-важните теории, прилагани при моделирането на софтуерни системи:

– теория на множествата (set theory)
– релации (relations)
– функции (functions)

! Тези теории са в основата на формални модели като *Z нотацията*, *Машини на състоянието*, *Мрежи на Петри* ...

Използвани са при изграждането на спецификации,
- подобряване и
- доказателство/тестване на софтуерните системи.

Множества – принадлежност и разширение

Множеството (интуитивно) е добре дефинирана колекция от отделни обекти.

?=? {синьо,червено,зелено}

Пр.:1) {1, 3, 5, 7, ... }

2) {червено, зелено, синьо, зелено}

3) {Peter, Doug, Kevin, Eduardo, ... }

4) { yes, no }

- Но НЕ !: {red, 1, 2, 3 }
- *Пр.*:

- the four oceans of the world
- the individuals who have been appointed to the post of secretary-general of the United Nations
- the passwords that may be generated using eight lower-case letters
- the prime numbers
- the collection of programs written in C^{++} that halt if run for a sufficient time on a computer with unlimited storage

Дефиниране & принадлежност

- Множествата могат да бъдат зададени чрез изброяване (by extension):

  Odds == {1, 3, 5, 7, ... }
  Colors == {red, green, blue, blue, green}
  СТ == {Иван, Никола, Мария, ... }

- Синтактична абревиатура == (по дефиниция) { ... }

- Тест за принадлежност: $3 \in$ Odds ; $2 \notin$ Odds

  забел. $\in$ **е предикат** върху множества и елементи.

- Множества с дефинирани имена:

  $\varnothing$ множество без елементи (the “null” or “empty” set) ; също: { }
  N множество на естествените числа { 0, 1, ... }
  Z множество на целите числа {... -2, -1, 0, 1, 2, ... }
  ....

Дефинирани множества

- Можем да дефинираме нови множества от основни елементи:

  – наречени “основни множества”(“basic sets”)
  – записвани “Множеството” (the set)
  – оператор за равенство = между тях (по презумпция).

Пр.

[BookIdentifiers] = [Date, Name, Place]

По-късно ще видим как можем да въведем допълнително твърдение за елементите в дадено нововъведено множество.

Равенство и размер (cardinality) на множество

• **Размер** (**Cardinality**) на множество: **#**

{1, 2, 4} = 3

А следващите?

{ {1,2}, {1,2,3,4,5,6,7} } = ?

{1, 2, 2, 4} = ?

? Защо?

• **Равенство** на множества: Две множества са равни, ако имат едни и същи елементи:

$$(S = T) \Leftrightarrow (\forall x \bullet x \in S \Leftrightarrow x \in T)$$

- Подреждането и мултипликативността на елементите няма значение!
- Алтернативно определение (*axiom of extension*):

$$\frac{(\forall x : t \bullet x \in u) \wedge (\forall x : u \bullet x \in t)}{t = u}\ [\text{ext}]$$

Сравнение на множества

• **Обобщение на равенството** на множества – сравнение на множества:

Ако всеки елемент от множество ***s*** е представен и в множество ***t***, то казваме, че множество ***s*** е подмножество на ***t***, т.е. $s \subseteq t$

Example 5.7 Let *Benelux* denote the set of countries in the Benelux economic union, and let *Europe* denote the set of all countries in the European Union. Since the formation of the EU, it has been true that $Benelux \subseteq Europe$. There were other partners when the EU (then the EEC) was formed in 1957, so it is also true that $\neg(Europe \subseteq Benelux)$. □

Можем да формулираме правило

$$\frac{\forall x : s \bullet x \in t}{s \subseteq t} \text{ [subset]}$$

което да използваме в двете посоки:

$$s \subseteq t \land t \subseteq s \Leftrightarrow s = t$$

Операции с множества

• **! Можем да формираме множества от други множества с помощта на операторите**:

∩ **сечение** (intersection); **U обединение** (union), **\ разлика** (difference)

• *Пр.*: Нека A == {1,2,3} и B == {3,4,5}

A ∩ B = {3}

A **U** B = {1,2,3,4,5}

A **** B = {1,2}

• *Забел*. "=" вместо "==". → *? Защо?*

• A е подмножество на B ($A \subseteq B$), ако всеки елемент от A е също и елемент от B.

Тогава: A = B , ако $A \subseteq B$ и $B \subseteq A$ → *Запишете формално?*

Аксиоми и закони

- **Основни аксиоми**
  - Принадлежност към множество (set membership): $\forall x \bullet x \in \{X\}$
  - Съществува празно множество: $\forall x \bullet \neg (x \in \varnothing)$

- **Закони** (могат да бъдат доказани) - комутативен, асоциативен, дистрибутивен

$$A \cap B = B \cap A \qquad A \cup B = B \cup A$$
$$(A \cup B) \cup C = A \cup (B \cup C)$$
$$(A \cap B) \cap C = A \cap (B \cap C)$$
$$A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$$

  - и други

Как могат да бъдат доказани тези закони?

Определяне на множество (set comprehension)

- Изброяването на елементите на дадено множество не винаги е възможно.
- Можем да опишем елементите на множество чрез различаване според определено свойство/качество на тези елементи.

Българи == множеството на гражданите с български паспорт

Primes == множество на целите числа, които са прости

! Всеки елемент удовлетворява описания критерий за принадлежност..

- ?: Как можем да дефинираме такъв критерий?

Определяне на множество (set comprehension)

Проста форма

{x : S | P(x)}

“множеството от всички **x** в **S**, които удовлетворяват **P(x) ”,** или
“множеството от всички **x** в **S** така, че **P(x)”**

x : S - **декларативна част**

P(x) - предикатна част

•*Пр.*:

* Естествените числа по-малки от 20: **{x: N | x < 20}**
* Естествените числа: **{x: N | true}**
* Празно множество на естествените числа: **{x: N | false}**
* Четни цели числа: $\{x: \mathbb{Z} \mid (\exists y: \mathbb{Z} \bullet x = 2y)\}$

• *забел*. Променлива в предиката

Определяне на множество

Специфика в използваните символи

Често се интересуваме от израз, дефиниран от стойности, удовлетворяващи предиката: $\{x: S \bullet f(x)\}$,

където **f(.)** е функция, дефинирана за елементи от множеството **S**.

Пр.: В рамките на криминално разследване властите изискват от базата данни да бъдат извадени адреси на престъпници, карали червена кола:

А) $\{ x : Person \mid x \text{ drives a red car} \bullet address(x) \}$

Б) ако няма сведение за цвета на колата:

$$\{ x : Person \bullet address(x) \}$$

Определяне на множество

Най-общото описание комбинира двете форми – декларативна и предикатна:

$$\{x: S \mid P(s) \bullet f(x)\}$$

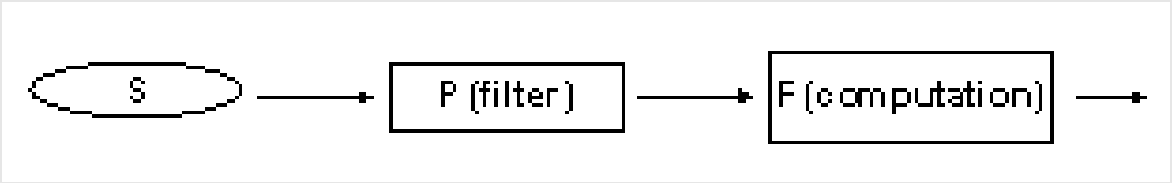
Пр.: а) множество от квадрати на естествените числа по-малки от 20

$$\{x: N \mid x < 20 \bullet x^2\}$$

б) множество от квадрати на четни цели числа

$$\{x: Z \mid (\exists y: Z \bullet x = 2y) \bullet x^2\}$$

Или илюстративно – формиране на принципа на “вложените тръби”*:*



Определяне на множество

Кратки форми:

а) определяне без терм част

$$\{ x : S \bullet e \} = \{ x : S \mid true \bullet e \}$$

б) определяне без предикатна част

$$\{ x : S \mid p \} = \{ x : S \mid p \bullet x \}$$

Повече от една променлива

$$\{ x : A; y : B \mid p \bullet e \}$$

Пр. за самостоятелна работа: An eyewitness account has established that the driver of the red car had an accomplice, and that this accomplice left a copy of the Daily Mail at the scene. The authorities are now interested in tracing the *set of potential criminals*:

$$\{ x : Person;\ y : Person \mid x \text{ is associated with } y \land x \text{ drives a red car} \land y \text{ reads the Daily Mail} \bullet x \}$$

Степенно множество (Power sets)

• (*конструктор над множество*): Множеството на всички подмножества на **S** се нарича **степенно множество** **S** и се отбелязва с **P S**

• *Пр.*: **P** {1, 2, 3 } =
{∅, {1}, {2}, {3}, {1,2}, {1,3}, {2,3}, {1,2,3}}

P N = {∅, {1},{1,2}, {1,3} ...}

• Степенното множеството се използва за дефиниране на нови типове.

class-groups == **P** Student

Integer-sets == **P N**

Ako x : **P N**, то x е множество от цели числа.

Забел.: Когато е написано **“ .. : P ..**” се чете “**множество от ...**”

(! Разлика между “:” и “∈”)

• Алтернативен символ

F ***A*** в случай на крайно множество на подмножествата на ***A***

Декартово произведение (Cartesian product) - 1

• (*конструктор над множества*): Начин за структуриране на информацията, при който асоциираме обекти от различен вид.

Декартовото произведение е множество от *n*-торки т.е. подреден списък от елементи.

Пр.: S == { (2, red), (5, blue), (3, red) }

• **Множеството от всички наредени двойки (x, y)**, конструирани от две множества ***a*** и ***b***, се нарича Декартово произведение или само произведение на ***a*** и ***b*** и се бележи ***a*** x ***b***.

$$(x, y) \in a \times b \Leftrightarrow x \in a \wedge y \in b$$

Редът на компонентите в Декартовото произведение е важен т.е.

a x ***b*** ≠ ***b*** x ***a*** !

Пр. Какво представят следните записи?

N x N двойки от естествените числа
(2, red) ∈ N x Color
S: P (N x Color)

Декартово произведение - 2

- В общия случай *n*-торките се дефинират чрез променливи (аналогично на record constructor of Pascal)

- Обобщена форма:
  - { декларация | предикат }

Note!: the predicate is sometimes called an invariant over the state space defined by the declarations

- Примери

  {(x, y): N x N | y = x + 1} = {(0,1), (1,2), (2,3), …}

  { x, y : N | y = x + 1} = {{0,1}, {1,2}, {2,3}, …}

  { x: P Z, y: N | y = #x} = { ({-1,2,3}, 3), { Ø, 0}…}

Декартово произведение: обобщение

n-торката $(x_1, ..., x_n)$ (***n*** - tuple) е представена в Декартовото произведение $a_1 \times a_2 \times \ldots \times a_n$, ако и само ако всеки елемент x_i е елемент от съответното множество a_i :

$$\frac{x_1 \in a_1 \wedge \ldots \wedge x_n \in a_n}{(x_1, \ldots, x_n) \in a_1 \times \ldots \times a_n} \; [\text{cart} - \text{mem}]$$

Две ***n***-торки са еднакви, ако са равни във всеки елемент:

$$\frac{x_1 = y_1 \wedge \ldots \wedge x_n = y_n}{(x_1, \ldots, x_n) = (y_1, \ldots, y_n)} \; [\text{cart}-\text{eq}]$$

За да отбележим/дефинираме един елемент от Декартовото произведение, например от t – торката, използваме нотация за проекция: $t.1$, $t.2$, …

$$\frac{t.1 = x_1 \wedge \ldots \wedge t.n = x_n}{t = (x_1, \ldots, x_n)} \; [\text{cart}-\text{proj}]$$

Дефиниции в Z нотацията

Z нотация. Дефиниции

В *Z* нотацията съществуват няколко начина за **дефиниране на обект**:

декларирane;
чрез абревиатура;
чрез аксиома;

и специални механизми за:

свободен тип;
схема.

1) Дeклариране:

а) Ако обектът е множество или основен тип:

[Type]

име

Дефиниране: Декларации - Тип

Дефиницията за **типа** на обектите – **максималното множество в границите на разглежданата спецификация**.

Така всяка стойност на променливата ***x*** се асоциира с един тип: най-голямото множество, на което ***x*** принадлежи.

[Type] или [X], X е дефинирано множество

В **Z нотацията** съществува един дефиниран тип – множество на целите числа **Z**:

- Всички други типове се изграждат чрез този тип
- чрез **основни типове** от стойности;
- допълнителни типове, дефинирани чрез *конструкторите :* степенно множество и Декартово произведение**.**

Забел. Можем да приложим “type-checking” алгоритми към математическия текст на документацията, за да се справим с непоследователност в използването на имена и изрази.

Дефиниране: Декларации - променлива

б) Декларация - обектът е ***променлива*** (алтернативно подпис (signature))

x : A - въвежда нова променлива **x** от множество **A.**

- *Пр.:*

A hotel switchboard uses a software package to maintain a record of call charges to current guests. A formal specification of this system could include the declaration

$$[Guest, Room]$$

introducing two basic types to represent the set of all guests and the set of all rooms. A variable of the type *Guest* is introduced by the following declaration:

$$x : Guest$$

x : A | x ∈ s - подпис и ограничение

Глобална декларация изисква аксиоматично деклариране

Абревиатури

2. **Абревиатури**: Обектът е еднакъв с вече въведен математически обект

symbol == term

Пр.: 1.

The abbreviation definition

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

introduces a set *Additive*, as another name for the set described in enumeration above. The names *red*, *green*, and *blue* must be defined elsewhere, they are not introduced by the abbreviation. If they are declared as elements of a type *Colours*, then *Additive* is a constant of type $\mathbb{P}\, Colours$. □

Пр.: 2.

$$English == \{p : Person \mid p \text{ drinks tea} \land p \text{ takes sugar}\}$$

Абревиатури: Общи абревиатури

Общи абревиатури (generic abbreviations): дефинира *фамилия* от символи, отнасящи се до отделни индекси или параметри.

symbol parameters == term

Дефинира глобална константа *symbol* , параметризирана чрез списък от множества, всеки от които може да се представи в израза *term*.

Пр.:

Разлика между празни множества на различни типове !

$$\varnothing[S] == \{x : S \mid false\}$$

Абревиатури: Общи абревиатури

Пр.:

For any set T, we may define the set of all non-empty subsets of T as follows:

$$\mathbb{P}_1 T == \{ a : \mathbb{P} T \mid a \neq \varnothing \}$$

We are happy to omit the brackets from the parameter list in the definition and in instantiations:

$$\mathbb{P}_1 \{0, 1\} = \{\{0\}, \{1\}, \{0, 1\}\}$$

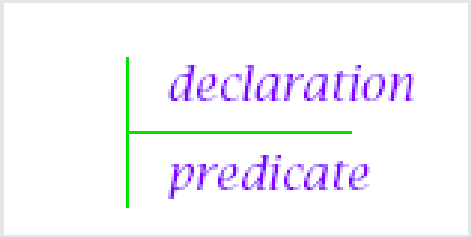
Глобална константа *symbol* между списък от параметри.

parameters symbol parameters == term

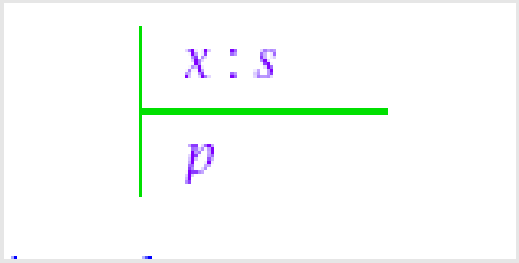
Пр. ***s*** rel ***t*** == **P**(***s*** x ***t***)

3. Аксиоматични дефиниции

1) **Аксиоматични дефиниции**: Описанието на обекта включва ограничения – **аксиома за дадения обект**.



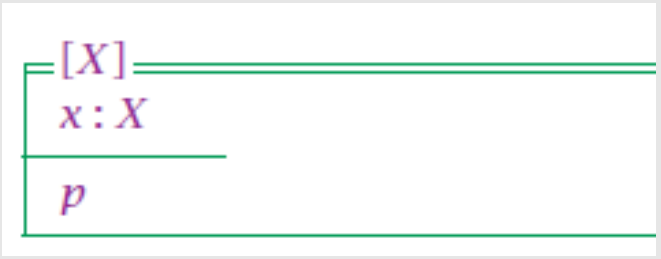
Пр.:



We may use an axiomatic definition to define the set of natural numbers:

Общи аксиоматични дефиниции

2) **Общи аксиоматични дефиниции**: Описание на *фамилия от глобални* константи (свободен тип), параметризирани чрез някое множество ***X***.



x обща константа от типа *X*

p предикат

X м-во, разглеждано като основен тип

Пр.: Дефиниране на символа за подмножество:

$$[X]$$

$$\_ \subseteq \_ : \mathbb{P} X \leftrightarrow \mathbb{P} X$$

$$\forall s, t : \mathbb{P} X \bullet$$

$$s \subseteq t \Leftrightarrow \forall x : X \bullet x \in s \Rightarrow x \in t$$

The $\subseteq$ symbol denotes a relation between two sets of the same type $\mathbb{P} X$

Забел. Дефиницията не трябва да вкарва неконсистентност в спецификацията!

Множества и предикати

2) *Предикатът* се дефинира чрез множество от обекти, които удовлетворяват дадения предикат.

If p is a predicate with a free variable x of type t, and

$$c = \{ x : t \mid p \}$$

then we say that c is the characteristic set of p: it is the set of values of x for which p is true.

Example 6.15 We wish to formalise the predicate 'is a crowd' upon sets of people. To do this, we introduce a set of sets:

$$crowds : \mathbb{P}(\mathbb{P}\, Person)$$
$$crowds = \{ s : \mathbb{P}\, Person \mid \#s \geq 3 \}$$

. With this definition of *crowds*, we may make statements such as

$$\{Alice, Bill, Claire\} \in crowds$$

and

$$\{Dave, Edward\} \in crowds$$

The first of these propositions is true, the second is false. □