Декларативно програмиране в процедурните езици

Формули от първи ред

Владислав Ненчев

Софийски Университет “Св. Климент Охридски”
Факултет по Математика и Информатика
Катедра по Математическа Логика и Приложенията ѝ

6 Декември 2016

Предикатно смятане от първи ред

Език на предикатното смятане от първи ред:

константи, променливи, функции, предикати, $\&$, $\vee$, $\neg$, $\Longrightarrow$, $\Longleftrightarrow$, $\forall$, $\exists$

Предикатно смятане от първи ред

Език на предикатното смятане от първи ред:

константи, променливи, функции, предикати, $\&$, $\vee$, $\neg$, $\Longrightarrow$, $\Longleftrightarrow$, $\forall$, $\exists$

Примери:

- $\forall x \forall y \forall z (x \equiv y \ \& \ y \equiv z \Longrightarrow x \equiv z)$
- $\forall a \forall b \forall c \exists x_1 \exists x_2 (a x_1^2 + b x_1 + c = 0 \ \& \ a x_2^2 + b x_2 + c = 0 \ \& \ x_1 \neq x_2)$
- $(\forall x \in V)(\exists e_1 \in E)(\exists e_2 \in E)(e_1.in = x \ \& \ e_2.out = x)$

Предикатно смятане от първи ред

Език на предикатното смятане от първи ред:

константи, променливи, функции, предикати, $\&$, $\vee$, $\neg$, $\Longrightarrow$, $\Longleftrightarrow$, $\forall$, $\exists$

Примери:

- $\forall x \forall y \forall z (x \equiv y \ \& \ y \equiv z \Longrightarrow x \equiv z)$
- $\forall a \forall b \forall c \exists x_1 \exists x_2 (a x_1^2 + b x_1 + c = 0 \ \& \ a x_2^2 + b x_2 + c = 0 \ \& \ x_1 \neq x_2)$
- $(\forall x \in V)(\exists e_1 \in E)(\exists e_2 \in E)(e_1.in = x \ \& \ e_2.out = x)$

За обхождане на дадено множество с квантори използваме итератори. Понякога се налага дефиниране на нови итератори в текущия обект (например, итератор на пътищата в граф).

Формули в C#

Чрез IEnumerable.All и IEnumerable.Any.

Формули в C#

Чрез IEnumerable.All и IEnumerable.Any.

```
bool IsTransitive() =>
  ! vertices.Any(x => vertices.Any(y =>
    vertices.Any(z =>
      edges.Contains(Edge(x, y)) &&
          edges.Contains(Edge(y, z)) &&
          !edges.Contains(Edge(x, z)))));
```

```
bool AllAccessCentralVertex() =>
  vertices.All(x => vertices.Any(y =>
    AreConnected(x, y) &&
    vertices.All(z => edges.Contains(Edge(y, z)))));
```

Формули в Java

Чрез Stream.allMatch, Stream.anyMatch и Stream.noneMatch.

Формули в Java

Чрез Stream.allMatch, Stream.anyMatch и Stream.noneMatch.

```
class Graph
{
  private Set<Integer> vertices;
  private Set<GraphEdge> edges;
  public boolean IsForwardSerial()
  {
    return vertices.stream().allMatch(
      vertex -> edges.stream().anyMatch(
        edge -> edge.From == vertex));
  }
  public boolean IsBackwardSerial()
  {
    return vertices.stream().allMatch(
      vertex -> edges.stream().anyMatch(
        edge -> edge.To == vertex));
  }
}
```

Формули в JavaScript

Чрез Array.every и Array.some.

Формули в JavaScript

Чрез Array.every и Array.some.

```
Map.prototype.IsMonotonicallyIncreasing =
  function()
  {
    var keys = Array.from(this.keys());
    return keys.every(key1 => keys.every(key2 =>
      key1 >= key2 ||
        this.get(key1) < this.get(key2)));
  }
Map.prototype.IsMonotonicallyDecreasing =
  function()
  {
    var keys = Array.from(this.keys());
    return keys.every(key1 => keys.every(key2 =>
      key1 >= key2 ||
        this.get(key1) > this.get(key2)));
  }
```

Формули в Python

Чрез Iterable.all, Iterable.any и Iterable.map.

Формули в Python

Чрез Iterable.all, Iterable.any и Iterable.map.

```
def IsFunctional(relation):
  return
    all(map(lambda tuple1:
      all(map(lambda tuple2:
        tuple1[0] != tuple2[0] or
        tuple1[1] == tuple2[1],
      relation)),
    relation))
```

Упражнения

- Проверка, че дадена релация образува прост цикъл, ако се разгледа като граф.

Курсова задача III - примери

Задача:
Да се извършат следните проверки за подаден неориентиран граф с оцветени върхове:

- че няма едноцветно ребро;
- че няма четириъгълник, който да е само в два цвята.

Вход: Неориентиран граф с оцветени върхове.
Изход: True/False

Задача:
Да се провери за подаден ориентиран граф с оцветени върхове, че във всеки негов цикъл има поне 3 различни цвята.
Вход: Ориентиран граф с оцветени върхове.
Изход: True/False

Курсова задача III - примери (продължение)

Задача:
Да се извършат следните проверки за подаден ориентиран граф с числа по върховете:

- че графа е функционален (т.е. релацията, образувана от ребрата му, е функционална);
- че графа е монотонен (т.е. ако има ребро от връх n към връх m то $n < m$).

Вход: Ориентиран граф с числа по върховете.
Изход: True/False

Задача:
Да се провери за подаден ориентиран граф, че той е дърво (но не и гора) и широчината му е 3.
Вход: Ориентиран граф.
Изход: True/False

Курсова задача III - примери (продължение)

Задача:
Да се провери за подаден ориентиран граф с оцветени ребра, че няма два едноцветни пътя, които да се пресичат по средата (да образуват X).
Вход: Ориентиран граф с оцветени ребра.
Изход: True/False

Задача:
Да се провери за подаден неориентиран свързан граф, че има поне два различни центъра.
Вход: Неориентиран свързан граф.
Изход: True/False

Курсова задача III - примери (продължение)

Задача:
Дадено е множество от обекти, два двуместни оператора над него и един едноместен оператор на същото множество (операторите са представени с Map обекти). Да се провери, че законите на де Морган важат за операторите.
Вход: Множество от обекти, два двуместни и един едноместен оператори.
Изход: True/False

Задача:
Дадена е крайна група (чрез множество от обекти и двуместен оператор над него, представен с Map обект). Дадено е и подмножество на обектите на групата. Да се провери, дали това подмножество поражда нормална подгрупа на дадената.
Вход: Крайна група и подмножество на обектите на групата.
Изход: True/False

Курсова задача III - примери (продължение)

Задача:
Дадена е двуместна релация (чрез множество от двойки). Да се провери, дали тази релация е частична подредба, за която има поне 20 елемента, които са независими помежду си.
Вход: Двуместна релация.
Изход: True/False

Задача:
Дадено е множество (базис) от три n-местни вектора (векторите се записват като списъци или n-орки) и множество от числа. Да се провери, дали подаден n-местен вектор може да се представи като линейна комбинация на тези от базиса, като се използват само коефициенти от даденото множество от числа.
Вход: Базис от три n-местни вектора, множество от числа и още един n-местен вектор.
Изход: True/False