Лекция 9. Моделирование массивов

Цель лекции

Построить аксиоматику для массивов.

Постановка задачи

- Назовем последовательность значений одного типа логическим массивом. Домен массивов – это множество всех логических массивов.
- Элементы логического массива проиндексированы, начиная с 0.
- У переменной с доменом массивов можно читать элемент, изменять элемент. При изменении элемента переменная получает новое значение. При создании массива задается его размер и начальное значение (одно) всех элементов.
- Нужно добавить домен массивов в блок-схемы. Для этого нужно придумать типы-символы, функциональные символы, предикатные символы и аксиомы. Придумать операторы CALL для работы с массивами.

Символы

- `type array 't` – логический массив
- `type arrayVar 't = { mutable items: array 't}` – тип переменных в блок-схеме с доменом массивов
- `function length (a: array 't)` – длина массива, это функциональный символ, значит можем его использовать только в спецификациях
- `predicate valid_index (a: array 't) (i: int) = 0 <= i < length a`
- `function get (a: array 't) (i: int)` – получение элемента с индексом `i`
- `function set (a: array 't) (i: int) (v: 't): array 't` – изменение элемента с индексом `i`

Аксиомы

```
axiom get_set: forall a i j v.
    get (set a i v) j =
    (if i = j then v else get a j)
axiom len_set: forall a i v.
    len (set a i v) = len a
axiom len_nonneg: forall a.
    len a >= 0
```

Примитивы для операторов CALL

```
val ln (a: arrayVar 't): int
    ensures { result = len a.items }
val gt (a: arrayVar 't) (i: int) : 't
    requires { valid_index a.items i }
    ensures { result = get a.items i }
val st (a: arrayVar 't) (i: int) (v: 't): array 't
    requires { valid_index a.items i }
    ensures { result = set a.items i }
val make (sz: int) (v: 't): array 't
    requires { sz >= 0 }
    ensures { len result = sz }
    ensures { forall i. 0 <= i < sz ->
        get result i = v }
```

Пример

- На языке WhyML напишите модель программы, решающей следующую задачу: на вход подается массив и два значения; нужно заменить в массиве все вхождения первого значения на второе. В этой модели входные переменные не должны изменяться.
- Напишите спецификацию задачи на языке Why.
- При помощи методов Флойда докажите полную корректность модели программы относительно этой спецификации.

Конструкции old, at. Метки

- Разрешим изменять входные переменные, т.е. сделаем типом входной переменной тип `arrayVar`.
- Теперь в спецификациях нужно обращаться к значению переменной-массива, которое было у нее при вызове функции. Для этого нужны следующие конструкции:
- `old expr` – это для постусловия
- `at expr 'label` – это для спецификаций внутри функции, надо создать метку в коде функции (`'label: expr`)