24 Октября 2013

Инсерционное моделирование 1

Лекция 9

Атрибутные среды

Атрибутные среды

логическая база

Система простых типов с областями значений (int, real, …)

Функциональные символы: набор типизированных функциональных символов $(\xi_1, \xi_2, ..., \xi_m) \to \xi, m \geq 0$

Предикат – функциональный символ типа $(\xi_1, \xi_2, ..., \xi_m) \to \text{Bool}, m \geq 0$

Интерпретированные функциональные символы (+, *, …)

Интерпретированный символ арности 0 – **константа**

Атрибуты – неинтерпретированные функциональные символы

Атрибут арности 0 – **простой атрибут**, арности > 0 – **функциональный**.

сигнатура, словарь, класс

Ограничения реализации

Только простые типы аргументов и значений

Интерпретация функциональных типов

Ограничения на области значений атрибутов (массивы)

Базовый логический язык

типизированный (многосортный) язык первого порядка

Атрибутные выражения

$$f(t_1, t_2, ..., t_m), m \geq 0$$

Алгебраические выражения (термы)

константы

атрибутные выражения

функциональные выражения

$$f(t_1, t_2, ..., t_m), m > 0, x + y, a.x, ...$$

Универсальность языка первого порядка

Литералы

истинностные значения 0,1

интерпретированные

Формулы

пропозициональные связки: $\wedge, \vee, \neg, ...$

кванторы $\forall x P(x), \exists x P(x)$

ограниченные кванторы $\forall (x \in A) P(x), \exists (x \in A) P(x)$

Может быть расширен модальностями темпоральной или нечеткой логики

Конкретные атрибутные среды

Конечная разложимость

Константное атрибутное выражение:

$f(t_1, t_2, ..., t_m), m \geq 0, \quad t_1, t_2, ..., t_m$ не содержат атрибутов

Неразложимое состояние конкретной среды – отображение

$$s : A \to D$$

множества A константных атрибутных выражений
в множество их значений D, сохраняющее типы.

Ограничения реализации

Почти для всех $t \in A, s(t) = \perp$

Состояние эквивалентно формуле $t_1 = a_1 \wedge ... \wedge t_n = a_n$

или присваиванию $t_1 := a_1 \wedge ... \wedge t_n := a_n$

Переходы конкретной атрибутной среды

Действия

– проверка условий, формулы базового языка

– присваивания $(x_1 := t_1, x_1 := t_2, ..., x_1 := t_m)$

Переходы

$$s \xrightarrow{check(\alpha)} s, \alpha(s) = 1$$

$$s \xrightarrow{y} y(s)$$

$$f(t_1,...,t_m) \notin A \Rightarrow s(f(t_1,...,t_m)) = s(f(s(t_1),...,s(t_m)))$$

$$s(f(t_1,...,t_m) := t) = f(s(t_1),...,s(t_m)) := s(t)$$

$$s(x_1 := t_1,..., x_m := t_m) = (s(x_1 := t_1),..., s(x_m := t_m))$$

$$y(s) = s * s(y)$$

Задача

определить композицию y*z
операторов параллельного присваивания

Агенты конкретных атрибутных сред
(императивные агенты)

Действия: $\mathrm{check}\,\alpha, x := y$

Системы уравнений – программы с переходами

Параметрические системы уравнений – программы с процедурами

Процедуры функции ?

Последовательная композиция $uv = \sum_{u \xrightarrow{a} u'} a.(u'v) + \sum_{u=u+\varepsilon} \varepsilon v$

$$0v = 0, \Delta v = v, \bot v = \bot$$

Условная композиция и цикл

$$\text{if } \alpha \text{ then } P \text{ else } Q = \mathrm{check}\ \alpha.P + \mathrm{check}\ \neg\alpha.Q$$

$$\text{while } \alpha \text{ do } P = \text{if } \alpha \text{ then}(P; \text{while } \alpha \text{ do } P)\text{else } \Delta$$

Параллельная композиция – параллельные программы над общей памятью

Функция погружения

$$\frac{E \xrightarrow{a} E', u \xrightarrow{a} u'}{E[u] \xrightarrow{a} E'[u']}$$

E неразложимое состояние

$$E[u, v] = E[u; v]$$

$$E[u, v] = E[u \parallel v]$$

Операционная семантика императивного программирования

(включая недетерминированные и параллельные программы над общей памятью)

Машины Гуревича (ASM)

Словарь: конечное множество функциональных символов (арность ≥ 0)
Реляционные символы: подмножество словаря
Выделены: равенство =, true, false, undef, унарный символ Bool

Структура X **над словарем** Y: базовое множество структуры X + интерпретация символов словаря на базовом множестве структуры X.

Программа ASM:
Присваивание $t := u$
Условное правило if φ then R else S endif
Параллельная композиция: par $R_1, R_2, ..., R_n$ endpar

Машина: состояния – структуры над словарем Y, замкнутость относительно изоморфизма структур и программы ASM.
Отношение переходов определяется программой.

Пример

```
if b=0 then d:=a
else if b=1 then d:=1
else if a ≥ b then par(a:=b, b:= a mod b)
else par(b:=a, a:= b mod a)
```

Словарь:
a, b, d: 0-арные ф-ции,
$\geq$, mod: бинарные

Состояния:
Базовое множество Nat,
a, b, d:Nat
$\geq$: (Nat,Nat) $\rightarrow$ Bool
mod: (Nat,Nat) $\rightarrow$ Nat

Почему произвольный алгоритм эквивалентен ASM?

Всякий **последовательный алгоритм**
должен удовлетворять следующим постулатам:

Последовательное время (sequential time)

Транзиционная система с функциональным отношением переходов и выделенным множеством начальных состояний

Постулат абстрактных состояний (abstract states)

Состояние есть структура первого порядка определенная с точностью до изоморфизма

Ограниченность проверок (bounded exploration)

присваивания на одном шаге зависят от конечного числа значений термов (алгебраических выражений) на текущем состоянии.