Машини на състояния – варианти – Резюме

Проблеми при стандартните машини на състояния

-Големият брой състояния дори при прости системи може да усложни анализа.

-Необходимост от компактно представяне на състояния и действия:

- Използване на предикати за описване на състояния.
- Дефиниране на целите чрез промени в източника.
- Фокус върху текстови описания вместо графични за улеснение на кодирането.

Основни варианти на машини на състояния:

А)Състоянията като функция: Всяко **състояние на М** е **крайна функция от крайно множество Var от променливи (“identifiers”, “names”) към (вероятно) безкрайно множество Val от типови стойности - крайна частична функция**

M = ({S :Var ⇸ Val}, I, A, d)

Пр.:Множеството от състояния на брояч е тотална функция: с функция на преходите: {s:{x}→ int s(x) ≥ 0}

с функция на преходите**:**

dinc={ (s,a,s'): S ×{inc}× S s'(x) = s(x) +1}

Функцията на преходите:

d action ={ (s,a,s'): S {action} x S| Φ [s(v)/ v]) ∧Ψ [s'(v)/ v' ,s(v)/ v]}

v е вектор от променливите на състоянието.

Може да бъде представена по-кратко чрез **нотация “тип програмиране”**: **pre-post** спецификация. За машината M = (S, I, A, d) най-общия запис :

action

    pre **Φ** (v)

    post **Ψ** (v, v′)

action (header) ∈ A

Φ и Ψ са съответно pre and post предикати върху вектора v на променливите на състоянието.

Импликация (→) е основната логическа връзка в pre-post спецификациите, защото **постусловията зависят от предварителните** условия: **Φ(v)→Ψ(v,v′)**

Конюнкция (∧) може да участва в дефинирането на сложни предварителни условия (или постусловия), изискващи едновременно изпълнение на няколко условия.

Б) Действия с аргументи

Преходите могат да зависят от аргументи. Пример: inc(i:int) pre i>0 post x′=x+i

Lambda абстракциите са използвани за кратко и формално описание на действията.

Действия с резултати

Допълнителна структура на действията се дефинира, когато **резултатът от действията са необходими на външния**

наблюдател. Преходът на състоянията е случай, който се описва чрез двойка събития:

а) **извикване** (invocation event): името на действието и стойностите на неговите входни аргументи или

б) отговор (response event): името на условието за приключване и стойността на неговия резултат.

Използват се специални думи като:

-ok: За нормално приключване.

-result: За върнатата стойност на действието.

-извънредно (exceptional) приключване; event

В) Функция на преходите– недетерминизъм

Когато **има повече от едно възможно следващо състояние** при еднакво действие и еднакво изходно състояние, то функцията на преходите трябва да свързва множество от състояния:

d (sk , action) = {s1 ′, s2 ′,…, sn ′ }

Недетерминизмът се представя с **дизюнкция (или) в post състоянието**.

inc(i:inc)

pre i > 0

post x’ = x + i **v** x’ = 2i

Г) Съвместно използване на разгледаните случаи

Общият шаблон, прилаган при описание с pre-post нотация на всяко действие action в A от M = (S, I, A, d), е:

action(inputs)/ $\mathbf{term_1}$ ($\mathbf{output_1}$), …, $\mathbf{term_n}$ ($\mathbf{output_n}$)

pre Φ (v)

post Ψ (v, v′)

където **inputs** е списък на аргументите и техният тип, $\mathbf{term_i}$ - името на i-тото условие за приключване и $\mathbf{output_i}$ е типът и резултатът, съответстващ на $\mathbf{term_i}$.

Φ и **Ψ** са **предикати** на състоянието върху вектора v.

Използваните в това изложение определители:

• ok – използва се в header за нормално прекъсване

• result – използва се в post-condition за върната стойност

• terminates – използва се в post-condition за стойност на условието за приключване. Стойността е указана в header.

• empty

Други машини на състоянието, често използвани в практиката

-Крайни автомати

Детерминистични крайни автомати (DFSA) M = (S, I, F, A, d):

S е крайно (ограничено) множество от възможни състояния;

I, I ⊆ S е множество от начални състояния (**singleton set**);

F ⊆ S е крайно множество от **крайни/последни** състояния;

A е крайно множество от действия;

$d \subseteq S \times A \nrightarrow S$ е детерминистична, частична функция.

trace † - пътека, която завършва с последно състояние.

L(M) може да бъде безкраен (безкраен брой стрингове на M)

-Крайни изпълнения и безкрайно поведение - модел, включващ (само) ограничени пътеки (**finite-trace model**).

-Машините са с поведение, съставено с **безкрайно множество** от **крайни пътеки**:

-Опростява разсъжденията

-Практическа резонност – безкрайното изпълнение не може да се види

-Недостатък – невъзможност да опише “deadlocks”, “ fairness”. За тази цел се изисква усложняване на структурата на пътеките и поведението

Разсъждения върху МС:

• **Най-важната характеристика на МС е инвариантността**: предикати Q, които са верни за всички достижими състояния. (Пр. Броячите)

1/Индукция върху състоянията от изпълненията. Удобна е когато има рекурсивна структура на домейна:

Нека има **изпълнение**: $\langle s_0, a_1 , s_1 , a_2 , \ldots, s_{i-1} , a_i , s_i , \ldots\rangle$. За да се докаже, че характеристиката Ξ е **инварианта**, необходимо за всяко изпълнение:

1/ Основен случай: Показва се валидност за началното състояние s0
2/ Индуктивна стъпка: Приема се валидност за състояние s_{i-1} и се доказва за състояние s_i

2/Предикатът, който формира на състоянията, е по-силен от инвариантността, която се стремим да докажем P ⇒ Ξ

3/Д-во чрез правило върху pre/post спецификацията (най-често използвана): (Case analysis of all actions)

1. Показва се, че Ξ е истинен за всички начални състояния

2. За всяко действие Приема се, че

- pre-условието **Φ** е в сила в pre-състоянието,

- инварианта **Ξ** е валиден за pre-състоянието

- post-условието **Ψ** е в сила в pre и post състоянията

показва се, че **- Ξ е валиден и в post - състоянието Следователно Ξ е инварианта**