Modellazione e Analisi di Sistemi

Indice

Capitolo 1

Abstract State Machines

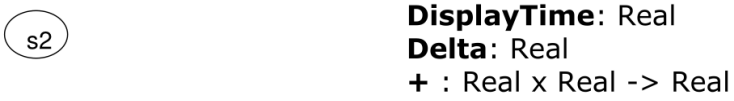
Le ASM sono delle FSM (*Final State Machines*) con stati generalizzati; rappresentano la forma matematica di macchine che estendono la nozione di FSM, **ampliando la definzione di stato** e **modificando la forma delle transizioni**.

Stati

Gli stati di controllo non strutturati vengono sostituiti da stati (strutturati) che modellano:

- **dati** complessi arbitrati (con domini di base e funzioni per la struttura)
- **operazioni** per la manipolazione di dati

Possiamo definire gli stati come delle *algebre*.



Transizioni

Le transizione sono *"regole"* che descrivono il cambiamento di funzioni da uno stato al successivo; permettono di modificare la struttura algebrica durante l'esecuzione della ASM.

if condition then Updates

Negli FSM le transizioni sono rappresentate con delle frecce.

Le ASM sono dotate di un ambiente di tool per:

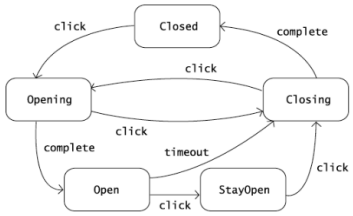
- editing
- simulazione
- validazione
- verifica
- generazioni di casi di test

Un modello ASM può essere visto come pseudocodice su strutture dati astratte.

Da FSM a ASM

Domini:
Stati: insieme degli stati

Funzioni:
ctl_state: Stati
click: boolean
complete: boolean
timeout: boolean



Regole di transizione:

if ctl_state = Opering and click
then
ctl_state:= Closing

if ctl_state = Closing and click
then
ctl_state:= Opening

if ctl_state = Closing and complete
then
ctl_state:= Closed

.....

Inizializzazione:

State = {Opening, Closing, Open, ….}
ctl_state = Open

Possiamo definire ASM = (header, body, main rule, inizialization)

Domini:
State: insieme degli stati

Funzioni:
ctl_State: State
input: String
output: String

Regole di transizione:
r_s1_1 = **if** ctl_State = s1 and
input = "1"
then
ctl_State:= "s1"
output:= "0"
r_s1_0 =
r_s2_1 = ...
r_s2_0 = ...

main rule

initialization

asm FSM
import StandardLibrary
signature:
controlled ctl_State: String
monitored input: String
out output: String

header

definitions:
rule r_s1_1 = **if** ctl_ = "s1" and input = "1" **then**
par ctl_State:= "s1", output:= "0" **endpar**
endif
rule r_s1_0 = **if** ctl_State = "s1" and input = "0" **then** ...
rule r_s2_1 = ...
rule r_s2_0 = ...

body

main rule r_Main = **par** r_s1_1, r_s1_0, r_s2_1, r_s2_0 **endpar**

default init so:
function currentState = "s1"

1.1 Formalismo

1.1.1 Vocabolario

DEF: Un **vocabolario** Σ è una collezione finita di nomi di funzioni.
Le funzioni possono essere dinamiche o statiche, a seconda che l'interpretazione del nome della funzione cambia o no da uno stato al successivo (funzioni in senso matematico).

1.1.2 Costanti

Le funzioni statiche di arietà zero sono dette **costanti**. Ogni vocabolario contiene sempre le costanti *undef, true, false.*
Ad esempio:

- i numeri sono costanti numeriche
- `voto = 30`

1.1.3 Funzioni statiche

Le funzioni statiche (arietà > 0) sono definite tramite una legge fissa.
Ad esempio:

- operazioni tra numeri (+, -, ...)
- operazioni tra booleani (AND, OR, ...)
- `max(m, n)`

1.1.4 Fuzioni dinamiche

Le funzioni dinamiche di arietà zero sono le variabili dei linguaggi di programmazione.

1.1.5 Stato ASM

DEF: Fissato un vocabolario Σ, uno **stato** A del vocabolario Σ è un insieme non vuoto X, detto *superuniverso di* A, con le interpretazioni dei nomi delle funzioni di Σ.
Da questa definizione, segue che:

- se f è un nome di funzione *n-aria* di Σ, allora la sua interpretazione f^A è una funzione da X^n a X
- Se c è un nome di costante di Σ, allora la sua interpretazione c^A è un elemento di X

Possiamo definire il superuniverso come un *"dominio di interpretazione"*; i simboli del vocabolario, presi singoralmente, sono soltanto simboli.

1.1.6 Domini ASM

Il superuniverso di uno stato ASM è suddiviso in *universi*, rappresentati dalle loro funzioni caratteristiche.

Se A è un sottoinsieme dell'insieme X, la funzione caratteristica di A è quella funzione da X all'insieme $\{0,1\}$ che sull'elemento $x \in X$ vale 1 se x appartiene ad A, e vale 0 in caso contrario.

Ogni universo rappresenta un dominio. In base a questa rappresentazione degli insiemi in termini di funzioni caratteristiche, uno stato di una ASM consente di modellare **domini eterogenei**.
Alcuni esempi di domini:

- predefiniti, come `Interi, String, ...`
- definiti dall'utente, come tipi astratti o a partire da altri domini

Esempio

Dominio $X = \{1, 2, a, b, mario, pippo\}$, ripartito in domini:

- $Interi = \{1.2\}$
- $Char = \{a, b\}$
- $String = \{mario, pippo\}$

1.1.7 Termini ASM

DEF: i termini di Σ sono espressioni sintattiche così costruite:

1. Variabili $v_0, v_1, v_2, \ldots$ sono termini
2. Costanti c di Σ sono termini
3. Se f è un nome di funzione *n-aria* di Σ e $t_1, \ldots, t_n$ sono termini
   $\Rightarrow f(t1, \ldots, t_n)$ è un termine

Ad esempio:

- $v_0 + v_1$
- $1 + (v_2 * 0)$

Un termine che non contiene variabili è detto chiuso. I termini sono *oggetti sintattici*. **Assumono significato (o semantica) nello stato**; il suo valore è l'*interpretazione del termine* in A.