Descrizione del Sistema

Giorgio Mariani

Sommario

In questo documento viene data una descrizione generale del comportamento del sistema interessato, andando a descrivere input, output e componenti interne di questo. Lo scopo di mantenere informazioni più dettagliate è riservato ad altri documenti.

1 Scopo del Sistema

Il sistema si pone come obbiettivo la capacità di generare automaticamente delle librerie *Simulink*® contenenti monitor[1] per formule in *Bounded Linear Temporal Logic* (*BLTL*). Come input il sistema richiede due stringhe; il nome di un file contente formule *BLTL* ed il nome della directory in cui verrà inserita la libreria in output.

2 Input del Sistema

Input formula file

Il primo input del sistema è il nome di un file contente le formule *BLTL* di cui interessa generare i monitor. Questo input è obbligatorio per il corretto funzionamento del sistema. Il file deve essere codificato secondo la codifica **ISO/IEC 8859-1**.

Output directory

Il secondo input del sistema è il nome della directory in cui verrà salvata la libreria in output, assieme a dei file di supporto per questa. Occorre che non esista una directory o un file con il nome specificato, la cartella infatti verrà creata durante l'esecuzione del sistema. Inoltre occorre che la cartella padre a quella indicata sia esistente. Questo input non è necessario, infatti nel caso in cui non venga specificato una cartella con un nome di default verrà creata (nella stessa directory dove si trova locato il sistema).

3 Output del Sistema

Come precedentemente accennato il sistema durante la sua esecuzione crea la *output directory* utilizzando il nome dato in input, dentro a questa vengono inseriti diversi file, ora presentati.

Output library

Al termine di un esecuzione corretta del sistema dovrà essere presente nella *output directory* un file (con nome **bltl_lib.slx**) contenente la libreria generata, in particolare per ogni formula nell' *input file* è presente un blocco sottosistema rappresentante il monitor per quella proprietà. Tali sottosistemi prendono in input un numero di segnali reali non complessi equivalente al numero di variabili presenti nella formula associata e da in output un segnale booleano. Il nome del monitor associato alla i-esima formula in *input file* è indicato con **monitor**_i.

[1] Con monitor si intende un sistema in grado di verificare se una certa proprietà (scritta in una qualche logica) sia vera.

Monitor generati

I monitor generati sono sotto-sistemi con lo scopo di indicare se la formula generatrice, durante l'esecuzione di un modello contenente il monitor, si falsifica per almeno un istante. Se si verificasse una situazione del genere il monitor restituirebbe come output (da quel momento fino al termine della simulazione) il valore **1**, altrimenti **0** è restituito. Per poter utilizzare il monitor occorre copiarlo dalla *output library* nel modello interessato.

Output Support Files

Nella *output directory* oltre alla libreria generata saranno presenti dei file di supporto, appunto utilizzati per il corretto funzionamento di questa. In particolare i file che dovranno necessariamente essere contenuti nella cartella di output sono:

mexSfunction.mex*arch*

Funzione *mex* implementante una S-Function utilizzata nei monitor. Il suffisso *arch* indica l'architettura per cui l'eseguibile è stato compilato.

slblocks.m

Funzione *MATLAB®* contenente meta-dati per la libreria generata, in particolare consente la visualizzazione di questa nel *Library Browser* di *Simulink®*.

4 Logiche Temporali utilizzate

Nel sistema vengono utilizzate due logiche temporali estremamente simili tra loro, entrambe derivate dalla famosa *Linear Temporal Logic*, logica utilizzata per verificare proprietà "dilatate" nel tempo.

Bounded Linear Temporal Logic (*BLTL*)

L'utente del sistema si interfaccerà solo con questo tipo di logica, ha come costrutto base (ovvero le foglie dell'albero sintattico di una formula generica) dei predicati formati da relazioni su combinazioni lineari di variabili reali (oppure delle costanti booleane). Per decidere la validità di una formula per questa logica occorre avere a disposizione una traccia di simulazione che assegni un valore alle variabili utilizzate per un intervallo temporale.

ESEMPIO

Un esempio di formula in *BLTL* è il seguente:

$$\Diamond^1(a \cdot x_1 + b \cdot x_2 \leq \beta \wedge True)$$

Con $a, b, \beta \in \mathbb{R}$ e x_1, x_2 nomi di variabili dipendenti dal tempo e definite dalla traccia di simulazione utilizzata.

*BLTL**

Versione leggermente diversa della *BLTL* dove invece di avere traccie definite su variabili reali si hanno traccie definite direttamente sui predicati nella formula. Questa logica viene utilizzata all'interno del sistema e invisibile all'utente.

ESEMPIO

Un esempio di formula in *BLTL** è il seguente:

$$\Diamond^1(\rho \wedge True)$$

Con ρ un predicato il cui valore dipende dal tempo e definito dalla traccia utilizzata per la valutazione.

5 Componenti del Sistema

Viene presentato ora un elenco delle componenti che costituisco il sistema, insieme ad una veloce descrizione:

Parser Componente che si occupa della lettura dell' *input file* e della successiva costruzione degli albero sintattici associati alle formule presenti nel file.

Library Generator Componente che gestisce la creazione ed il trasferimento dei file da utilizzare, assieme alla compilazione dei *mex* file utilizzati. Inoltre gestisce gli interfacciamenti tra le altre componenti.

Monitor Maker Componente che si occupa,dato un modello *Simulink®* ed un albero sintattico *BLTL*, di aggiungere un sotto-sistema con funzione di monitor per la formula rappresentata dall'albero.

Monitor Library Componente che mantiene una libreria *C++* utilizzata per il corretto funzionamento dei monitor costruiti.

Parser

Come precedentemente introdotto questa parte del sistema si occupa della lettura di *input file*, e nel caso questo sia esistente e rappresentante un insieme di formule valide, questa costruisce una sequenza di alberi sintattici per ogni formula. Questa componente nell'effettivo genera un array di strutture *MATLAB®* con dei campi precisi, mantenenti le informazioni necessarie per la costruzione dei monitor. In particolare gli alberi costruiti verranno utilizzati da *Monitor Maker* per la generazione dei rispettivi monitor nella *output library*.

Library Generator

Parte del sistema che gestisce e coordina le altre componenti, assieme alla costruzione della *output directory*, inizializzazione e salvataggio della *output library* ed al trasferimento dei file di supporto necessari. Inoltre questa componente gestisce la compilazione dei *mex* file utilizzati dai monitor generati. Questa può essere quindi considerata la componente principale del sistema, che organizza e si interfaccia con le altri parti del sistema.

Monitor Maker

La componente *Monitor Maker* gestisce la costruzione dei monitor nella libreria. In particolare dati un nome *monitor_name*, un modello *Simulink® lib* ed un albero sintattico *tree*, questa si occupa di costruire un sotto-sistema in *lib* chiamato *monitor_name* e di aggiungere a questo i blocchi necessari affinché questo controlli correttamente che la proprietà indicata da *tree* rimanga valida durante la simulazione del sistema contente il monitor.

Dettagli monitor generato Il monitor generato avrà tante porte in input quante sono le variabili utilizzate nei predicati della formula generatrice, il segnale che le porte in input prendono deve essere di tipo scalare reale non complesso. Il monitor è dotato di un unica porta in output, con tipo scalare booleano, tale porta restituisce il valore **1** se durante la simulazione la proprietà associata al monitor si è falsificata per almeno un istante. Più nel dettaglio, sia t l'istante associato ad un *Major-timestep*, allora se la proprietà si è falsificata per almeno un istante tra l'inizio di simulazione e $t - \delta$ (con δ un valore reale dipendente dalla struttura della formula) allora per quel *Major-timestep* il valore restituito in output dal monitor è **1**, altrimenti è **0**.

Monitor Library

Parte non propriamente attiva del sistema, consiste in un insieme di classi e file *C++* utilizzate per calcolare la validità di una formula *BLTL**. Queste file contengono anche un implementazione di una S-function, utilizzata dai monitor generati. Queste classi vengono compilate in un unico *mex* file chiamato **mexSfunction.mex*arch***.

6 Utilizzo tipico

Supponiamo di avere un modello *Simulink®* chiamato *model.slx* di cui siamo interessati a testare se vale o meno una certa formula *BLTL* ϕ, come possiamo per risolvere questo problema? Una soluzione consisterebbe appunto nell'utilizzo del sistema mostrato, seguendo i passaggi ora riportati:

1. Creazione del file *infile.txt* contente una corretta rappresentazione di phi.
2. Esecuzione del sistema con in input *infile.txt* ed il nome della directory in cui vogliamo sia salvato la libreria in output (*output directory*).
3. Aggiunta della *output directory* al *MATLAB®* path.
4. Avvio di *Simulink®*.
5. Apertura di *model.slx*.
6. Apertura del *Library Browser*
7. Apertura della libreria generata (*output library*) nel *Library Browser*.
8. *Drag-and-drop* del monitor associato a phi nel modello *model.slx* (questo causa la copia del blocco in *model.slx*).
9. Assegnare i rispettivi segnali in *model.slx* come input al monitor appena copiato (per ogni variabile in phi dovrebbe esserci una porta con lo stesso nome nel monitor).
10. Collegare output monitor ad un blocco che ne elabori il segnale (ad esempio potremmo voler interrompere la simulazione quando la proprietà diventa falsa collegando il segnale ad un *assertion block*).
11. Simulare.