Università di Roma La Sapienza

Facoltà di Ingegneria dell'informazione, informatica e statistica

Dipartimento di Informatica

L'uso della GPU nel MBSE

Systems, Modelica, MPGOS and modelica2GPU

Riccardo La Marca 1795030

riccardo.lamarca98@gmail.com

lamarca.1795030@studenti.uniroma1.it

Ottobre, 2020

Indice

Cos'è MBSE?

MBSE, o Model-based System Engeenering, è una pratica dell'ingegneria dei sistemi secondo la quale la comunicazione tra i membri del team non avviene tramite documenti cartacei ma ben tramite lo sviluppo di modelli, solitamente tramite il linguaggio *SysML* o *UML*, per evitare abigiutà ed errori di interpretazione a causa del linguaggio naturale utilizzato.

Introduzione

Sempre più, nella nostra realtà, sta diventando importante modellare e simulare *sistemi cyber-fisici*, come droni o sistemi IoT, e *sistemi biologici*, come il ciclo cellulare o ancora lo sviluppo e la propagazione di un virus. Sebbene questi due siano argomenti diversi, hanno principalmente una cosa in comune: entrambi possono essere espressi come un sistema di equazioni differenziali (o *ODE*, *Ordinary Differential Equation*) il quale può essere integrato sul tempo al fine di dare un "vita" al modello stesso il quale risulterà in un dataset con cui si andrà a descrivere l'andamento, in termini di valori sugli stati, quindi la **traiettoria** del sistema stesso. Questa fase di integrazione viene chiamata *simulazione* del modello.

Gli *eventi* sono delle situazioni alternative che deviano il normale corso della traiettoria del sistema facendole toccare punti che altrimenti non avrebbero mai considerato. Esistono quindi due tipologie di eventi: di *stato* e di *tempo*. I primi sorgono da condizioni sugli stati, ad esempio $x > 10$ con x stato del sistema, mentre i secondi da condizioni sul tempo. Su quest'ultimi il discorso diventa più ampio però un classico evento di questo tipo scaturisce inserendo nel sistema una componente detta *Sample-And-Hold*, che agisce creando una condizione sul tempo della simulazione. Una componente di *Sample-And-Hold*, letteralmente "campionamento e mantenimento" è una classica componente che viene inserita all'intero di sistemi continui al fine di discretizzarne una parte di comportamento. Il nome stesso deriva dalle due azioni che questo svolge: (1) campiona il tempo della simulazione in base a dei valori che per semplicità chiameremo *start* ed *interval*, e (2) mantiene fino al prossimo sample il valore di uno stato. Un tipico esempio è l'**Analog to Digital Converter** (ADC) il quale implementa internamente il sottosistema di s&h[1]. Gli eventi, quindi, servono principalmente per immettere una discretizzazione all'interno di un sistema che altrimenti sarebbe continuo.

```
parameter Real sample_time(unit="s")=0.1251231;
when sample(0, sample_time) then
  omega1_measured = omega1; // omega1 è uno stato
end when;
```

Figure 1.1: Esempio di componente SH in Modelica.

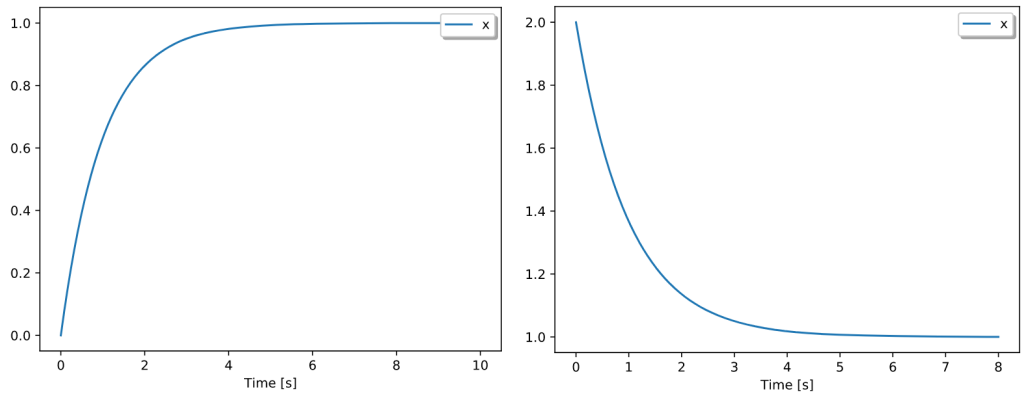
[1] Sample-And-Hold

Esempio1. First order system time event

Si consideri il semplice sistema descritto da una sola equazione differenziale[2].

$$\dot{x} = (1 - x) \tag{1.1}$$

Vediamo come questo sistema descrive semplicemente l'andamento dello stato x fino al punto 1, sul quale troverà l'equilibrio dal momento che per tutti i successivi istanti di tempo non cambierà mai di valore. Difatti il grafico della traiettoria sarà il seguente



(a) Traiettoria del sistema con valore iniziale $x = 0$

(b) Traiettoria del sistema con valore iniziale $x = 2$

Se volessimo aggiungere una componente SH all'interno del sistema per vedere il valore di x a determinati istanti di tempo, decidiamo prima il tempo di sampling (es. 0.3) e implementiamo tale evento, ottenendo

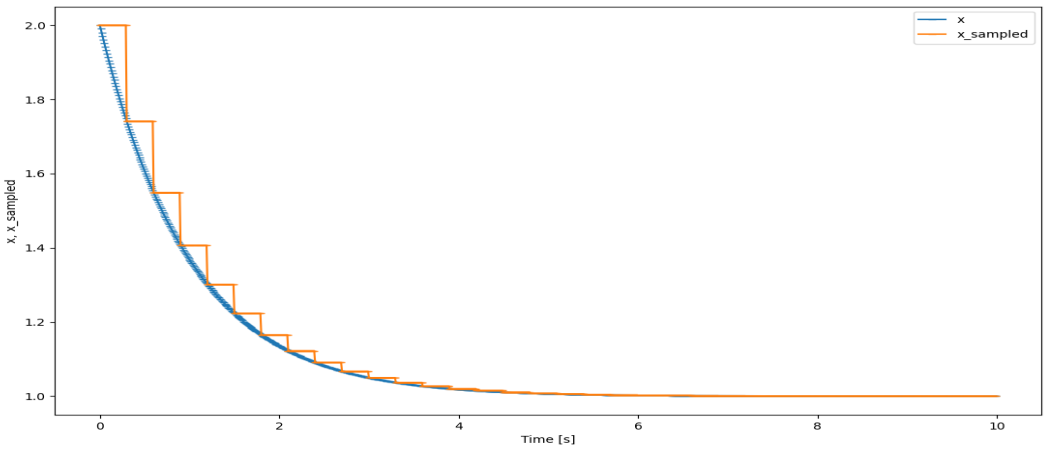


Figure 1.3: FirstOrderSystem con sample-and-hold e sample time di 0.3

[2]Scriveremo $\dot{x}$ al posto di $\frac{\mathrm{d}}{\mathrm{d}t}x(t)$

Questo è il codice Modelica che descrive il modello

```
model FirstOrderSystemWithSH "Semplice sistema di primo ordine"
    parameter Real x0=2.0;
    parameter Real sample_time=0.3;
    Real x;
    Real x_sampled;
initial equation
    x = x0;
    x_sampled = x0;
equation
    der(x) = (1 - x)
    when sample(0, sample_time) then
        x_sampled = x
    end when;
end FirstOrderSystemWithSH;
```

Esempio2. FirstOrderSystem with state event

Consideriamo adesso un secondo sistema descritto dalla seguente equazione differenziale

$$\dot{x} = -\sqrt{x} \tag{1.2}$$

Se provassimo ad eseguire una simulazione per 5 secondi, essa terminerebbe invece a 2. Questo perché le fasi di integrazioni precedenti hanno portato ad una traiettoria per lo stato x che a istanti tempi maggiori di 2 il valore potrebbe essere negativo generando quindi una *floating point exception*. Per ovviare a questo problema possiamo semplicemente introdurre una condizione sullo stato x che nel momento in cui $x < 0$ allora $\dot{x} = 0$, ottenendo la seguente equazione

$$\dot{x} = \begin{cases} -\sqrt{x}, & x \geq 0 \\ 0, & altrimenti \end{cases} \tag{1.3}$$