Sintesi automatica di software di controllo per sistemi ciberfisici

Federico Mari, Vadim Alimguzhin
Università degli Studi di Roma "Foro Italico", Sapienza Università di Roma
federico.mari@uniroma4.it, alimguzhin@di.uniroma1.it

Abstract

L'automazione e il controllo di sistemi ciberfisici sono tematiche di grande interesse per l'Industria 4.0. Molti sistemi ciberfisici sono di fatto sistemi di controllo a ciclo chiuso, dove un software guida un impianto affinché vengano soddisfatti requisiti di sicurezza. Visto il contesto altamente critico, tali requisiti diventano sempre più stringenti, motivando la ricerca in metodi di generazione automatica di software di controllo corretto per sistemi ciberfisici. Questo contributo presenta le linee di ricerca attuali del nostro gruppo di ricerca in questo contesto.

1 Introduzione

Con la diffusione crescente dei sistemi ciberfisici nell'Industria 4.0, l'Intelligenza Artificiale (IA) è chiamata ad affrontare nuove sfide nell'ambito del controllo e dell'automazione industriale. I sistemi ciberfisici sono di nevralgica importanza per le tecnologie abilitanti dell'automazione industriale. Alcuni esempi sono i sistemi di movimentazione dei materiali automatici e la robotica avanzata. Visto il contesto *safety-* e *mission-critical*, è necessario che tali sistemi soddisfino requisiti di sicurezza stringenti, motivando così la ricerca in metodi di analisi per sistemi ciberfici.

Molti sistemi ciberfisici sono di fatto sistemi di controllo a ciclo chiuso (figura 1) che consistono in due sottosistemi principali: il controllore e l'impianto (*plant*, *controlled system*). L'impianto è un sistema fisico composto, ad esempio, da dispositivi elettrici o meccanici mentre il controllore è un software di controllo eseguito su un microcontrollore. In un ciclo infinito, il controllore legge la conversione da analogico a digitale, *Analog-to-Digital* (AD), dell'output dei sensori dall'impianto e manda la conversione da digitale ad analogico, *Digital-to-Analog* (DA), dei comandi agli attuatori dell'impianto al fine di garantire che il sistema a ciclo chiuso soddisfi i requisiti di sicurezza.

L'impianto di un sistema ciberfisico è modellato come un sistema ibrido a tempo discreto, *Discrete-Time Hybrid System* (DTHS), la cui dinamica è definita da una combinazione booleana di vincoli (possibilmente non lineari) sulle sue variabili intere e reali. Il passaggio da tempo continuo a tempo discreto è lasciato al progettista (metodi di Runge-Kutta).

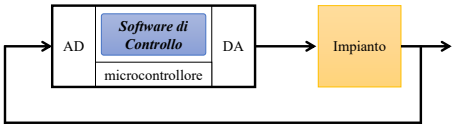


Figura 1: Sistema di controllo a ciclo chiuso

Siano dati (*i*) il modello DTHS dell'impianto di un sistema ciberfisico, (*ii*) le specifiche formali che descrivono i requisiti funzionali del sistema a ciclo chiuso (*safety* e *liveness*), (*iii*) le specifiche di implementazione che descrivono i requisiti non funzionali del software di controllo, quali ad esempio il numero di bit usati per le conversioni AD e DA (detto *schema di quantizzazione*) e il tempo di esecuzione nel caso pessimo, *Worst-Case Execution Time* (WCET). Il problema consiste nella generazione automatica di un software di controllo per il sistema (i) che soddisfi i dati requisiti funzionali (ii) e non funzionali (iii). In [Mari *et al.*, 2012] si dimostra che tale problema è indecidibile.

Nel seguito si presenta *Quantized feedback Kontrol Synthesizer* (QKS), un algoritmo che calcola una condizione sufficiente e necessaria per la soluzione di un problema di generazione di software di controllo per DTHS. QKS sfrutta tecniche di IA e di metodi formali (*Model Checking*). QKS è stato applicato efficacemente a diversi casi d'uso di interesse per il controllo e l'automazione industriale, quali ad esempio il convertitore DC-DC riduttore (Buck), e il pendolo invertito, *Inverted Pendulum* (IP), di figura 3. Figura 4 mostra sull'esempio IP come QKS migliori lo stato dell'arte rappresentato dal software Pessoa [Mazo *et al.*, 2010].

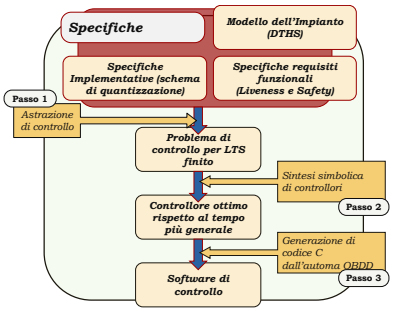


Figura 2: Flusso di QKS

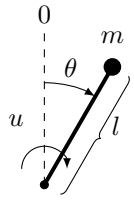
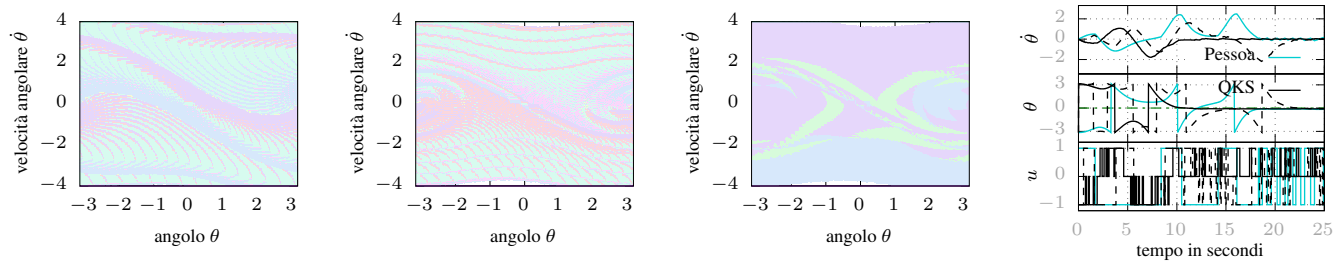


Figura 3: IP



(a) Pessoa: regione controllata. Colori diversi corrispondono a insiemi diversi di azioni u ammesse dal controllore. L'OBDD del controllore ha 19.994 nodi.

(b) QKS: regione controllata. Il confronto con figura 4b mostra una diversa strategia del software di controllo. L'OBDD del controllore ha 21.629 nodi.

(c) QKS compatto: regione controllata. Le azioni permesse sono più omogenee e minimali. L'OBDD del controllore ha 5.031 nodi.

(d) Simulazione del sistema a ciclo chiuso con i software di controllo generati, a partire dall'equilibrio stabile del pendolo in basso. QKS compatto è tratteggiato.

Figura 4: *Confronto tra QKS e Pessoa su IP*. Schema di quantizzazione a 9 bit. *Liveness* e *safety*: raggiungere e mantenere l'equilibrio verticale del pendolo in alto. Tempo di esecuzione: QKS 1h:30m, Pessoa 2h. *Considerazioni sulle performance del software di controllo*: 1) Il controllore compatto di QKS ha una dimensione che è il 23% di quella normale (figura 4c vs figura 4b) a fronte di un peggioramento del tempo di stabilizzazione (23 secondi del compatto, tratteggiato, vs 10 del normale, figura 4d). 2) Il controllore normale di QKS stabilizza il pendolo più velocemente di quello di Pessoa (10 secondi vs 18, figura 4d) ma ha una velocità di cambio di switch più elevata.

2 Il software QKS

Figura 2 mostra il flusso di esecuzione di QKS. **Passo 1.** Dal DTHS di input, si genera una astrazione finita di controllo che dipende dallo schema di quantizzazione e che rappresenta l'impianto come è visto dal software di controllo dopo la conversione AD. **Passo 2.** Data una astrazione finita G, che rappresenta i requisiti di *safety* e *liveness*, si calcola un controllore K per il DTHS di input che, partendo da uno stato iniziale qualsiasi, guidi il sistema a ciclo chiuso in G malgrado il possibile comportamento non deterministico. **Passo 3.** Si traduce l'automa finito K in un software di controllo (codice in linguaggio C).

Al fine di generare il controllore K al passo 2, QKS applica un noto algoritmo di *search* per la sintesi simbolica basato sulla manipolazione di diagrammi di decisione binari ordinati, *Ordered Binary Decision Diagram* (OBDD). Tale algoritmo trova una soluzione ottima rispetto al tempo, poiché il controllore K guida il sistema a ciclo chiuso in G sempre seguendo cammini minimi. Pertanto, il software di controllo generato da QKS realizza una strategia di controllo vicina all'ottimo rispetto al tempo. Inoltre, grazie alla traduzione da un OBDD che ha profondità di decisione nota, tale software ha un WCET lineare nel numero di bit dello schema di quantizzazione (requisiti non funzionali).

Oltre alla versione sequenziale per DTHS lineari [Mari *et al.*, 2010; 2012; 2014] e nonlineari [Alimguzhin *et al.*, 2012a; 2017] (risultati in figura 4) QKS è disponibile in altre varianti. **Software di controllo compatto.** [Alimguzhin *et al.*, 2012b] sacrifica l'ottimalità temporale a favore della compattezza del software, cercando (passo 3, figura 2) regioni massimali che possano essere controllate eseguendo la stessa azione. **Parallelo.** [Alimguzhin *et al.*, 2013a] presenta QKS basato su *Map-Reduce* (passo 1, figura 2). **Esplorazione efficace dei parametri** [Alimguzhin *et al.*, 2013b] presenta una versione *on-the-fly* di QKS, che velocizza la computazione nei problemi che non hanno soluzione (passi 1 e 2, figura 2).

Riferimenti bibliografici

[Alimguzhin *et al.*, 2012a] V. Alimguzhin, F. Mari, I. Melatti, I. Salvo, e E. Tronci. Automatic control software synthesis for quantized discrete time hybrid systems. In *CDC 2012*. IEEE, 2012.

[Alimguzhin *et al.*, 2012b] V. Alimguzhin, F. Mari, I. Melatti, I. Salvo, e E. Tronci. On model based synthesis of embedded control software. In *EMSOFT 2012*. ACM, 2012.

[Alimguzhin *et al.*, 2013a] V. Alimguzhin, F. Mari, I. Melatti, I. Salvo, e E. Tronci. A map-reduce parallel approach to automatic synthesis of control software. In *SPIN 2013*, volume 7976 of *LNCS*. Springer, 2013.

[Alimguzhin *et al.*, 2013b] V. Alimguzhin, F. Mari, I. Melatti, I. Salvo, e E. Tronci. On-the-fly control software synthesis. In *SPIN 2013*, volume 7976 of *LNCS*. Springer, 2013.

[Alimguzhin *et al.*, 2017] V. Alimguzhin, F. Mari, I. Melatti, I. Salvo, e E. Tronci. Linearizing discrete-time hybrid systems. *IEEE TAC*, 62(10), 2017.

[Mari *et al.*, 2010] F. Mari, I. Melatti, I. Salvo, e E. Tronci. Synthesis of quantized feedback control software for discrete time linear hybrid systems. In *CAV 2010*, volume 6174 of *LNCS*. Springer, 2010.

[Mari *et al.*, 2012] F. Mari, I. Melatti, I. Salvo, e E. Tronci. Undecidability of quantized state feedback control for discrete time linear hybrid systems. In *ICTAC 2012*, volume 7521 of *LNCS*. Springer, 2012.

[Mari *et al.*, 2014] F. Mari, I. Melatti, I. Salvo, e E. Tronci. Model based synthesis of control software from system level formal specifications. *ACM TOSEM*, 23(1), 2014.

[Mazo *et al.*, 2010] M. Mazo, A. Davitian, e P. Tabuada. PESSOA: A tool for embedded controller synthesis. In *CAV 2010*, volume 6174 of *LNCS*. Springer, 2010.