

Curriculum Vitae
di
Salvatore La Torre

Indice

1 INFORMAZIONI GENERALI	1
2 ATTIVITÀ DIDATTICA	2
3 ATTIVITÀ PROFESSIONALI	3
4 ATTIVITÀ DI RICERCA	6
5 ELENCO DELLE PUBBLICAZIONI	18

1 Informazioni Generali

Interessi di Ricerca

Aspetti formali di specifica, analisi e sintesi di sistemi hardware e software. Linguaggi formali, teoria degli automi, sistemi ibridi, logica temporale, teoria dei giochi.

Titoli di Studio

- Laurea in Scienze dell'Informazione, Università degli Studi di Salerno, votazione 110/110 e lode, 28 Aprile 1994.
- Dottorato di Ricerca, Matematica Applicata ed Informatica, Università degli Studi di Napoli "Federico II", 10 Marzo 2000.
- Philosophiae Doctorate (PhD), Computer Science, University of Pennsylvania, Philadelphia (USA), 21 Dicembre 2001.

Borse di Studio

- Borsa di studio CNR per laureandi per lo svolgimento dell'attività di ricerca nell'ambito della Teoria degli Automi presso il Dipartimento di Informatica ed Applicazioni dell'Università degli Studi di Salerno, bando n. 209.01.58 del 19/07/92, dall'1 Settembre 1993 al 31 Agosto 1994.
- Borsa di studio CNR bando n. 201.19.01 del 30/11/94, codice 12.01.08, nell'ambito della tematica "Integrazione di tecniche di fault-tolerance e di diagnosi per il miglioramento dell'affidabilità dei sistemi" presso l'Istituto di Elaborazione dell'Informazione - CNR, Pisa, dall'1 Marzo 1995 al 15 Febbraio 1996.
- Assegno di ricerca, cofinanziato dal Fondo Sociale Europeo, relativo all'area scientifico disciplinare Scienze - Informatica presso il Dipartimento di Informatica ed Applicazioni dell'Università degli Studi di Salerno per il progetto di ricerca "Specializzato in verifica formale di sistemi", dall'1 Novembre 1999 al 30 Novembre 2000.

Formazione

- Visita al Computer and Information Science Department, University of Pennsylvania, Philadelphia, USA, Giugno 2005.
- Visita al Computer Science Department, University of Illinois, Urbana-Champaign, USA, Maggio-Giugno 2005.
- Visita al Computer and Information Science Department, University of Pennsylvania, Philadelphia, USA, Luglio-Agosto 2004.

- Visita all'Institut d'Informatique, University of Namur, Belgio, dal 14 al 19 marzo 2004.
- Visita al Computer and Information Science Department, University of Pennsylvania, Philadelphia, USA, Settembre 2001 - Settembre 2002.
- PhD in Computer Science, Penn Engineering School, University of Pennsylvania, USA, 1999-2001.
- Dottorato di Ricerca in Matematica Applicata ed Informatica - XI ciclo di durata quadriennale presso la Facoltà di Scienze MM. FF. NN. dell'Università degli Studi di Napoli "Federico II", 1995-1999.
- Visita al Computer and Information Science Department, University of Pennsylvania, Philadelphia, USA, Marzo-Luglio, 1998.
- European School of Computer Science su "Methods and Tools for the Verification of Infinite State Systems", Grenoble, France, 23-25 Marzo, 1997.

2 Attività Didattica

Relatore di tesi per il conseguimento della laurea in Informatica.

Membro commissione esami di profitto per i corsi:

Fondamenti di Informatica e Laboratorio, Laboratorio di Informatica I e II, Logica Matematica, Linguaggi di Programmazione I e II, Fondamenti di Programmazione, Linguaggi Formali e Compilatori, Laboratorio di Algoritmi e Strutture Dati.

Corsi insegnati:

- *Laboratorio di Algoritmi e Strutture Dati*, Laurea in Informatica, Università degli Studi di Salerno, anni accademici 2003-04, 2004-05.
- *Linguaggi di Programmazione II*, Laurea in Informatica, Università degli Studi di Salerno, anno accademico 2004-05.
- *Logica Matematica*, Laurea in Informatica, Università degli Studi di Salerno, anni accademici 2002-03, 2003-04.
- *Fondamenti di Informatica e Laboratorio*, Laurea in Matematica, Università degli Studi di Salerno, anni accademici 2002-03, 2003-04, 2004-05.

Assistenza docenza corsi all'estero:

- *Theory of Computation*, CIS511, University of Pennsylvania, anno accademico 1999-2000.

- *Introduction to Algorithms*, CSE320, University of Pennsylvania, anno accademico 1999-2000.

Assistenza docenza corsi in Italia:

- *Laboratorio di Sistemi Operativi*, Corso di Laurea in Informatica, Università degli Studi di Salerno, anno accademico 2000-01.
- *Laboratorio di Algoritmi e Strutture Dati*, Corso di Laurea in Informatica, Università degli Studi di Salerno, anno accademico 2000-01.

3 Attività Professionali

Membro di Comitati di Programma

- 4th International Conference on Verification, Model Checking, and Abstract Interpretation (VMCAI 2003), 9 - 11 Gennaio 2003, New York, USA.
- 16th International Conference on Computer-Aided Verification (CAV 2004), 14 - 17 Luglio 2004, Boston, Massachusetts, USA.
- 4th International Conference on Formal Modelling and Analysis of Timed Systems (FORMATS 2006), 25 - 27 Settembre 2006, Paris, France.
- 11th International Workshop on Formal Methods for Industrial Critical Systems (FMICS 2006), collegato CONCUR 2006, 26-27 Agosto 2006, Bonn, Germany.

Chair di sessione a Workshop e conferenze

- “Workshop on Games in Design and Verification”, GDV’04, Boston, Massachusetts, USA, July 18, 2004.
- “16th International Conference on Computer-Aided Verification”, CAV’04, Boston, Massachusetts, USA, July 13 - 17, 2004.

Revisione articoli per conferenze

CONCUR’99, CAV’00, ICALP’00, CAV’01, FSTTCS’01, HSCC’01, TACAS’01, HSCC’02, VMCAI’02, LICS’02, FTRTFT’02, CSL’02, STACS’03, VMCAI’03, TACAS’03, HSCC’03, ICALP’03, CSL’03, ICTCS’03, FSTTCS’03, VMCAI’04, TACAS’04, CAV’04, ICALP’04, STACS’05, CAV’05, ICALP’05, DLT’05, FSTTCS’05, IEEE-ACC’06, ICALP’06, CONCUR’06, SAS’06.

Revisione articoli per riviste

Formal Methods in System Design, Fundamenta Informaticae, IEEE Transactions on Automatic Control, Information and Computation, International Journal of Foundations of Computer Science, Logical Methods in Computer Science, Theoretical Computer Science.

Progetti di Ricerca Coordinati

- Progetto giovani ricercatori “Model-checking for Open Real-time Systems”, 2002-03. Responsabile scientifico: Salvatore La Torre. Membri: Marco Faella, Salvatore La Torre.

Studenti di Dottorato

- Aniello Murano (co-tutore Margherita Napoli), Tesi: “Decision Problems on Tree Automata and Analysis of Open Timed Systems”, Università degli Studi di Salerno, 2002.
- Gennaro Parlato, Tesi: “On the Model-checking of Hierarchical and Recursive State Machines”, Università degli Studi di Salerno, 2005.

Presentazioni di comunicazioni scientifiche su invito

- “Deciding Games in LTL Fragments”, *Centre Fédéré en Vérification*, Université Libre de Bruxelles, Bruxelles, Belgio, 19 Marzo, 2004.
- “Model-checking of Hierarchical/Recursive State Machines with Context-Dependent Properties”, *Séminaire d’Informatique - Institut d’Informatique*, Namur, Belgio, 15 Marzo, 2004.
- “Timed games with TCTL specifications”, *Workshop on Mathematical Models and Techniques for Analysing Systems*, Montreal, Canada, 30 Settembre - 4 Ottobre, 2002.
- “Parametric Linear Temporal Logic”, *Dagstuhl Seminar: Verification of Infinite-state Systems*, Dagstuhl, Germany, 2-7 Aprile, 2000.

Presentazioni di comunicazioni scientifiche a Convegni e Workshops

- “Decidability and Infinite Precision in Timed Automata”, *43-esimo Congresso Annuale AICA*, AICA’05, Udine, 5-7 Ottobre, 2005.
- “Perturbed Timed Automata”, *8th International Workshop on Hybrid Systems: Computation and Control*, HSCC’05, Zurich, Switzerland, March 9-11, 2005.
- “Playing Games with Boxes and Diamonds”, *14th International Conference on Concurrency Theory*, CONCUR’03, Marseille, France, September 3-5, 2003.

- “Optimal-Reachability and Control for Acyclic Weighted Timed Automata”, *2nd IFIP International Conference on Theoretical Computer Science*, IFIP TCS’02, Montreal, Canada, August 25-30, 2002.
- “Dense Real-time Games”, *17th Annual IEEE Symposium on Logic in Computer Science*, LICS’02, Copenhagen, Denmark, July 22-25, 2002.
- “Deterministic Generators and Games for LTL Fragments”, *16th Annual IEEE Symposium on Logic in Computer Science*, LICS’01, Boston, Massachusetts, USA, 16-19 Giugno, 2001.
- “Firing Squad Synchronization Problem on Bidimensional Cellular Automata with Communication Constraints”, *3rd International Conference on Machines, Computations and Universality*, MCU’01, Chisinau, Moldova, 23-27 Maggio, 2001.
- “Optimal Paths in Weighted Timed Automata”, *4th International Workshop on Hybrid Systems: Computation and Control*, HSCC’01, Roma, Italia, 28-30 Marzo, 2001.
- “A Decidable Dense Branching-time Temporal Logic”, *20th Conference on the Foundations of Software Technology and Theoretical Computer Science*, FSTTCS’00, New Delhi, India, 13-15 Dicembre, 2000.
- “Compositionality in the Synchronization of Processors exchanging a minimal amount of information”, *5th Workshop su Sistemi Distribuiti: Algoritmi, Architetture e Linguaggi*, WSDAAL’00, Ischia, Italy, 18-20 Settembre, 2000.
- “Polyhedral Flows in Hybrid Automata”, *2nd International Workshop on Hybrid Systems: Computation and Control*, HSCC’99, Berg en Dal, The Netherlands, 29-31 Marzo, 1999.
- “Synchronization of 1-Way Connected Processors”, *11th International Symposium on Fundamentals of Computation Theory*, FCT’97, Cracovia, Poland, 1-3 Settembre, 1997.
- “Synchronization of a line of identical processors at a given time”, *7th International Joint Conference on the Theory and Practice of Software Development*, TAPSOFT’97, Lille, France, 14-18 Aprile, 1997.

Altre presentazioni

- “Solving TCTL games with Timed Büchi Automata”, *Validation’02*, Genova, Italia, November 22-23, 2002.
- “CHARON model of the Electric Throttle Control for automotive systems: verification”, *CMU-Penn Meeting on Embedded Systems*, Allenberry, Pennsylvania, USA, 20 Dicembre, 2001.
- “Automati temporizzati su ω -alberi e temporal logic”, *Modelli e Metodi della Computazione e dei Linguaggi di Programmazione*, Anacapri, 30 Ottobre - 1 Novembre, 1995.

4 Attività di Ricerca

L'elevato livello tecnologico e l'economicità dei sistemi hardware e software ha portato al loro utilizzo in molti settori dell'attività umana. In molti casi, questi sistemi sono utilizzati in situazioni in cui il corretto funzionamento è un requisito irrinunciabile, come ad esempio per i sistemi di controllo di un aereo, le apparecchiature mediche salvavita o i sistemi di controllo di una catena di produzione. Un loro cattivo funzionamento, infatti, potrebbe comportare danni ingenti sia in termini economici che di vite umane.

Garantire il corretto funzionamento dei sistemi digitali non è mai stato un compito facile ed oggi è divenuto ancor più difficile a causa della loro notevole complessità. In risposta a questo problema sono state sviluppate diverse metodologie che in molti casi hanno dato luogo ad applicazioni industriali. I metodi formali sono sicuramente tra le metodologie che stanno rivestendo un ruolo sempre più importante nella realizzazione di sistemi affidabili. Uno dei loro vantaggi principali è rappresentato dal fatto che operano su modelli astratti e quindi possono essere utilizzati sin dalle fasi iniziali di un progetto. È possibile in questo modo prevenire l'introduzione di errori che se scoperti in una fase avanzata potrebbero causare notevoli ritardi con conseguente lievitazione dei costi. Per esprimere le proprietà da garantire e per rappresentare i modelli astratti dei sistemi si utilizzano degli opportuni formalismi. Per esempio, una verifica di correttezza si può ridurre a controllare che una proprietà espressa in linguaggio di specifica è soddisfatta da un modello del sistema espresso in forma di sistema di transizione (*model-checking*).

L'attività di ricerca svolta si colloca principalmente in questa area. In particolare, sono stati analizzati modelli formali di sistemi discreti, real-time e ibridi (automi finiti, automi con chiamate ricorsive, automi temporizzati, automi ibridi) e logiche temporali (discrete, real-time e parametriche). Nell'ambito di questi formalismi sono stati definiti e studiati il problema della sintesi, il problema della verifica e i problemi di ottimizzazione collegati. Per i modelli non noti in letteratura si sono anche studiati problemi inerenti i linguaggi formali accettati. Sono state investigate, inoltre, alcune nuove rappresentazioni di grafi infiniti basate su linguaggi regolari e se ne sono studiati i modelli corrispondenti. Un altro aspetto dell'affidabilità dei sistemi che è stato considerato riguarda le tecniche di misurazione della dependability, con l'introduzione di un nuovo modello per la valutazione delle misure di dependability, particolarmente utile per i sistemi iterativi.

Altre ricerche svolte hanno riguardato problemi teorici su sistemi di transizione. In particolare, sono stati presi in considerazione alcuni modelli di reti di processori studiandone il problema della sincronizzazione ad un tempo fissato. Infine, si è analizzato il problema della reversibilità di un testo sotto sostituzione parallela.

Nel seguito sono descritti i risultati ottenuti con i riferimenti agli articoli scientifici in cui sono riportati. Un elenco completo degli articoli scientifici prodotti compare nella parte finale di questo documento.

Linguaggi Formali e Teoria degli Automi:

a) Gli automi finiti non consentono di modellare esplicitamente il tempo. Gli automi temporizzati introdotti da R. Alur e D. Dill estendono gli automi finiti con variabili a valori reali (i cosiddetti clock). I clock vengono usati per esprimere vincoli temporali sui ritardi tra occorrenze di eventi. L'automa temporizzato è oggi uno dei modelli formali di sistemi real-time tra i più usati e studiati nel campo della specifica e verifica di sistemi.

Gli automi temporizzati accettano linguaggi temporizzati che consistono in sequenze di eventi annotati con il loro tempo di occorrenza. La classe dei linguaggi temporizzati definibili attraverso gli automi temporizzati è chiusa rispetto all'unione, all'intersezione e alla proiezione ma non rispetto al complemento. Inoltre, mentre il problema del vuoto è decidibile attraverso degli algoritmi simbolici che manipolano vincoli temporali, il problema dell'universalità e il problema dell'inclusione tra linguaggi temporizzati sono indecidibili. L'indecidibilità del problema dell'inclusione e la non chiusura rispetto al complemento hanno motivato molte ricerche mirate a limitare l'espressività degli automi temporizzati.

In [B3], viene introdotto il modello di automa temporizzato perturbato definito come un automa temporizzato i cui clock sono imperfetti e cambiano ad un tasso di velocità che varia nell'intervallo $[1 - \varepsilon, 1 + \varepsilon]$, per uno specifico errore di perturbazione ε . Si dimostra che il linguaggio di un automa perturbato con un unico clock è accettato da un automa temporizzato deterministico. Questo porta ad una procedura per verificare il problema dell'inclusione per circuiti asincroni, dove l'obiettivo è verificare che l'implementazione di un circuito non include comportamenti che non sono presenti nella sua specifica. In aggiunta, si prova anche che la determinizzazione e la decidibilità dell'equivalenza tra linguaggi non è possibile per automi con più di un clock anche in presenza di perturbazioni.

Automi temporizzati ed automi ibridi in generale rappresentano utili formalismi per la modellazione di sistemi. Tuttavia nelle fasi iniziali di un progetto, principalmente per la limitatezza delle informazioni disponibili, può essere arduo determinare il valore di una costante da utilizzare in un modello e quindi un progettista può trovare estremamente conveniente l'utilizzo di modelli che permettano di non specificare le costanti fin quando non se ne ha una stima adeguata. Il modello formale che viene considerato in [D4] è l'automa temporizzato con vincoli parametrici. Precedenti ricerche mostrano che il problema del vuoto per questi automi è indecidibile se vengono usati almeno tre clock con parametri. L'obiettivo della ricerca in corso è l'individuazione di sottoclassi di automi parametrici con problema del vuoto decidibile.

b) In [A9] vengono introdotti un modello di automa temporizzato su ω -alberi e la corrispondente teoria, allo scopo di rappresentare il tempo esplicitamente nel modello. L'automa da un lato estende agli alberi gli automi temporizzati su parole introdotti da Alur e Dill, dall'altro estende con i tempi gli automi su alberi. L'interesse per gli automi temporizzati su alberi è motivato principalmente dai collegamenti con le logiche temporali branching-time e i giochi temporizzati. Inoltre, la struttura temporale ad albero è utile per rappresentare la concorrenza (in forma di interfogliamento di esecuzioni di processi) e il nondeterminismo. Il modello introdotto è stato studiato nella versione deterministica e non deterministica con criteri di accettazione alla Büchi ed alla Muller. In particolare, sono state studiate le relazioni tra le classi di linguaggi individuate e con le corrispondenti classi di linguaggi non temporizzati. Inoltre, per le classi di linguaggi introdotte sono stati risolti alcuni problemi di decisione (vuoto, equivalenza, equivalenza "debole") e provate alcune proprietà di chiusura (unione, intersezione, complemento, concatenazione ed ω -iterazione).

c) I sistemi ibridi sono sistemi digitali real-time, "incastonati" in un ambiente analogico. Lo sviluppo tecnologico degli ultimi anni ha consentito l'uso di processori digitali in molti campi dell'attività umana, così oggi i sistemi ibridi controllano molti degli strumenti che quotidianamente utilizziamo. In molti casi questi sistemi, sia ad esempio quelli usati nelle automobili che negli aerei, operano in situazioni in cui la sicurezza è un fattore critico. Si

rende quindi necessario l'utilizzo di rigorose tecniche di analisi.

In [B21], si affronta un problema inerente la verifica degli automi ibridi. Gli automi ibridi sono modelli matematici di sistemi ibridi in grado di esprimere, in un unico formalismo, sia il loro comportamento discreto (attraverso le transizioni) che continuo (attraverso i vincoli differenziali). La loro verifica formale si basa su procedure di computazione simbolica che manipolano insiemi di stati. Per una sottoclasse di questi automi, i cosiddetti automi ibridi lineari, queste procedure possono essere implementate usando combinazioni booleane di vincoli lineari sulle variabili del sistema. In un automa ibrido lineare, il flusso ad ogni locazione di controllo è descritto da un poliedro (assegnato alla locazione) che dà i valori delle derivate prime delle variabili del sistema. La proprietà chiave di questi flussi è che gli automi ibridi lineari descrivono flussi poliedrici, cioè flussi che mettono in relazione un insieme di stati descritto da un poliedro con un altro poliedro. In questa ricerca si studia la possibilità di generalizzare la sintassi degli automi lineari ibridi mantenendo i flussi poliedrici. In particolare, si considerano flussi descritti da “origin-dependent rate polytopes”, cioè le derivate prime delle variabili del sistema (rates) dipendono non solo dalla locazione corrente ma anche dallo stato specifico visitato dall'automa al momento in cui transisce nella locazione corrente. Vengono individuate condizioni necessarie e sufficienti affinché una classe di flussi descritta da origin-dependent rate polytopes sia poliedrica. Si considera inoltre la classe dei flussi fortemente poliedrici, dove l'insieme degli stati che sono raggiungibili, a partire da un poliedro ed entro un certo tempo, è garantito essere a sua volta un poliedro. La classe dei flussi fortemente poliedrici consente risparmi computazionali nella verifica formale rispetto ai flussi poliedrici.

Questa ricerca è stata estesa e completata in [A5]. In particolare, si studia e si paragona alle precedenti una nuova classe di flussi: i cosiddetti flussi “polyhedrally sliced”, in cui l'insieme degli stati che sono raggiungibili a partire da un poliedro ed esattamente ad un tempo dato è garantito essere a sua volta un poliedro. Questa classe è interessante in quanto consente una simulazione simbolica dell'automa ibrido anche in presenza di flussi non poliedrici. Si discute inoltre un metodo per approssimare dinamiche esponenziali basato sulle classi di automi introdotte. Un'importante novità della ricerca svolta è rappresentata dal fatto che per la prima volta viene condotto un approccio semantico al problema della verifica formale degli automi ibridi.

d) La sostituzione parallela di una parola in un testo è definita nel seguente modo: dato un testo t tutte le occorrenze di una parola w sono sostituite in modo da ottenere un insieme di testi diversi in accordo alle diverse decomposizioni di t rispetto a w . In [A17] vengono fornite delle condizioni necessarie sulla forma di w affinché la sostituzione sia reversibile. Lo studio della reversibilità di un testo sotto sostituzione parallela ha delle ovvie implicazioni riguardanti la crittografia: dato un testo criptato, si vuole essere in grado di recuperare il testo in chiaro di partenza.

e) I grafi infiniti sono stati ampiamente studiati ed usati per modellare programmi concorrenti non terminanti e spesso risultano particolarmente utili per studiare le proprietà di grafi estesi. In questo contesto, l'uso di una rappresentazione adeguata è di primaria importanza.

In [B22] si propone e si studia un nuovo modo per rappresentare grafi infiniti attraverso i linguaggi regolari. Vengono presi in considerazione ipergrafi con iperarchi etichettati permettendo anche di avere più iperarchi tra la stessa tupla ordinata di vertici (multigrafi). I grafi vengono rappresentati attraverso un linguaggio regolare prefix-free L , per l'insieme dei

vertici, e un linguaggio regolare di tuple P , per gli archi, con il significato che tra due o più vertici vi è un arco se il suffisso della tupla¹ di parole rappresentanti i vertici appartiene a P . La classe di grafi così rappresentata risulta essere più grande della classe dei grafi equazionali introdotti da Courcelle.

In [A10], viene introdotta un'equivalente notazione basata sugli infissi di una tupla invece che sul suffisso. Infatti, tra due o più vertici vi è un arco se una delle tuple, costituita da prefissi delle parole ottenute dai vertici eliminando il più lungo prefisso comune, appartiene a P . Principali vantaggi in questa notazione, rispetto alla precedente, sono una rappresentazione più succinta e un maggiore potere espressivo nel caso di un linguaggio P finito. La classe di grafi individuata è stata confrontata con classi significative e contiene strettamente i grafi introdotti da Caucal. Si mostra, inoltre, che alcuni problemi tipici su grafi, quali l'esistenza di una radice, di un vertice isolato, di un pozzo o la determinazione del grado di un vertice, sono decidibili per questa classe di grafi pur non essendo decidibile la teoria monadica del secondo ordine. Infine, appartengono a questa classe anche alcuni grafi infiniti non esprimibili nel formalismo introdotto da Caucal e con teoria monadica del secondo ordine decidibile.

f) In [A3] vengono investigati nuovi criteri di accettazione per automi finiti su alberi infiniti. In particolare si rilassa il criterio di accettazione alla Muller: “un run è di accettazione se su ogni cammino gli stati che si ripetono infinite volte sono esattamente gli stati di un insieme di accettazione”. Si considerano il criterio di accettazione che richiede che “gli stati che si ripetono infinite volte sono contenuti in un insieme di accettazione” (Landweber) e il criterio di accettazione che richiede che “l'insieme degli stati che si ripetono infinite volte contiene tutti gli stati di un insieme di accettazione” (Muller-Superset). Questi criteri vengono confrontati in termini di succintezza ed espressività. Si prova che gli automi con accettazione di tipo Muller-Superset forniscono una caratterizzazione alternativa dei linguaggi di alberi infiniti accettati alla Büchi. Inoltre, nel caso deterministico questo criterio permette di descrivere linguaggi di alberi infiniti utilizzando uno spazio che è al più quello utilizzato dagli automi con accettazione alla Büchi, in generale, ed è esponenzialmente più piccolo, in alcuni casi. Gli automi su alberi infiniti con accettazione alla Landweber definiscono invece una classe di linguaggi che non è confrontabile con quella definita dagli automi alla Büchi. L'aspetto interessante di questa classe è che il problema del vuoto è decidibile in tempo polinomiale. Questo risultato quindi estende la classe di automi su alberi con problema del vuoto trattabile con interessanti ripercussioni nel campo della verifica dei sistemi.

Oltre al problema del vuoto, un'altra caratteristica interessante per una classe di automi nell'ambito della verifica dei sistemi, è la chiusura rispetto al complemento. Infatti dato un automa A_S che esprime la specifica del sistema ed un automa A_I che rappresenta un modello del sistema, il problema della verifica può essere ridotto a testare il vuoto del linguaggio $L(A_I) \cap \overline{L(A_S)}$. In [B4] si studiano le classi di automi che accettano il complemento dei linguaggi di alberi infiniti accettati alla Büchi deterministicamente e non deterministicamente. Si prova che il vuoto del complemento di un linguaggio accettato da un automa deterministico alla Büchi A può essere controllato in tempo polinomiale nella taglia di A . Per la classe nondeterministica invece si caratterizza il complemento con gli automi di Landweber alternanti e si prova che per questi ultimi il problema del vuoto è

¹Per suffisso di una tupla si intende la tupla che si ottiene eliminando da ogni componente della tupla il più lungo prefisso comune.

EXPTIME-completo.

g) In [A8] si introducono gli automi finiti deterministici con chiamate ricorsive. Gli automi finiti vengono estesi con transizioni che permettono chiamate ad altri automi. I linguaggi accettati da questi automi sono esattamente i linguaggi contex-free deterministici. L'aspetto interessante è che per caratterizzare tutti questi linguaggi si ha bisogno di automi con punti di uscita multipli. Se si restringono questi automi permettendo un solo punto di entrata e un solo punto di uscita, si caratterizzano tutti i linguaggi deterministici context-free che sono accettati da un automa push-down deterministico con un solo stato. Si analizzano inoltre i collegamenti con rappresentazioni di grafi e il problema della raggiungibilità.

Raggiungibilità e Model-checking:

a) Alcuni problemi interessanti come il job-shop scheduling problem possono essere modellati come problemi di raggiungibilità in automi temporizzati aciclici. Il problema della raggiungibilità è il problema di decidere in un sistema di transizione se esiste un'esecuzione che porta da uno stato iniziale ad un dato stato. Esso è collegato con la verifica delle proprietà di *safety* ("niente di indesiderato succede in un'esecuzione del sistema"), e per gli automi temporizzati è noto essere PSPACE-completo. In [D3] proviamo che per gli automi temporizzati aciclici il problema della raggiungibilità è invece NP-completo. La complessità rimane invariata anche se consentiamo al più un autociclo per ogni locazione dell'automata. Gli algoritmi di soluzione proposti si basano su riduzioni a programmi lineari di tipo misto (cioè programmi lineari sia con variabili intere che reali). Un altro risultato interessante contenuto in [D3], è che il problema della raggiungibilità è invece PSPACE-hard quando si permette un numero arbitrario di autocicli (la prova è ottenuta riducendo il problema della fermata per automi linear-bounded al problema della raggiungibilità in un automa temporizzato con 2 locazioni, n autocicli sulla locazione iniziale, e 6 clock). Sarebbe quindi interessante raffinare questo risultato per verificare quale sia la complessità del problema della raggiungibilità per automi temporizzati aciclici a meno di un numero di autocicli per locazione che sia costante (> 1) o logaritmico nella taglia dell'automata. L'obiettivo di questa ricerca è di estendere questi risultati ad arbitrari automi temporizzati con componenti fortemente connesse di "piccole" dimensioni.

b) In [A6] si introduce il concetto di automa temporizzato pesato e si studia il problema dei cammini (run) ottimali in questo modello. L'ottimalità di un cammino è stabilito relativamente ad alcune funzioni di costo che sono associate all'automata. Il problema viene ridotto allo shortest-path problem in un grafo direzionato pesato attraverso una costruzione che raffina la ben nota costruzione dell'automata delle regioni dovuto ad Alur e Dill. In questo modo si ottiene una procedura per risolvere problema dei cammini ottimali a partire da uno stato dell'automata che richiede tempo esponenziale nella lunghezza dei vincoli di clock. Lo stesso approccio permette di ottenere anche una procedura per risolvere lo stesso problema a partire da tutti gli stati di una regione data che richiede tempo esponenziale nel prodotto del numero dei clock e il massimo tra la lunghezza dei vincoli di clock e la taglia del massimo peso utilizzato. Questi risultati hanno un'interessante implicazione anche nel campo degli automi ibridi in senso stretto, in quanto il problema affrontato può essere riformulato come un problema di raggiungibilità negli automi ibridi le cui variabili sono tutte clocks eccetto una che ha un tasso di crescita che può essere diverso per ogni locazione. Questa classe di automi è più generale della classe degli automi temporizzati e non è inclusa negli automi

ibridi rettangolari inizializzati, che rappresentano le due classi di automi ibridi sinora studiate in cui la raggiungibilità è decidibile.

Diversi problemi di interesse pratico possono essere modellati con il problema della raggiungibilità ottimale in automi aciclici, come ad esempio vari problemi di scheduling ottimale e di controllo del traffico aereo. In [B10] si prova che il problema della raggiungibilità ottimale per automi pesati aciclici è FP^{NP} . Questo è conseguenza del fatto che il corrispondente problema di decisione è NP-completo. L'algoritmo di soluzione proposto consiste in una riduzione (mediante scelta non deterministica) a programmi lineari.

c) Per modellare sistemi di grandi dimensioni è certamente utile poter annidare in diversi livelli i dettagli del sistema. In alcuni casi, l'uso di modelli gerarchici può portare a risparmi significativi in termini di spazio richiesto per la rappresentazione del sistema. Un generico modello formale gerarchico per sistemi a stati finiti è la cosiddetta macchina gerarchica a stati finiti. Il modello che si propone in [B8] differisce da altri modelli gerarchici noti nella letteratura in quanto consente di etichettare tutti i nodi della struttura con proposizioni atomiche e non solo i nodi semplici (nodi senza strutture gerarchiche annidate al loro interno). Per questo motivo è detto macchina gerarchica a stati con proprietà dipendenti dal contesto (CHSM). In [B8] si considera anche la generalizzazione delle CHSM che consente di avere una struttura di chiamate ricorsive tra i vari moduli (CRSM). Per entrambi i modelli si studiano il problema della raggiungibilità e del model checking di formule LTL. La raggiungibilità può essere modellata come il model-checking di una formula del tipo $\diamond\varphi$ dove φ è una formula booleana rappresentante un insieme di stati del modello. Si prova che sia per le CHSM che le CRSM, il problema della raggiungibilità è NP-completo e che il model-checking di una generica formula LTL richiede tempo lineare nella taglia del sistema ed esponenziale nella taglia della formula. Si determina anche una classe di formule booleane per le quali la raggiungibilità può essere risolta in tempo lineare sia nella taglia del modello che nella taglia della formula. I risultati ottenuti migliorano un precedente approccio, nel senso che su alcune classi di sistemi si ottiene un miglioramento di tempo esponenziale.

d) I programmi spesso richiedono l'esecuzione concorrente di diverse threads che interagiscono tra di loro. Ogni thread a sua volta può comportare l'invocazione ricorsiva di procedure e quindi utilizzare uno stack locale. In generale, combinando chiamate ricorsive e sincronizzazione di task si ottengono modelli di calcolo in grado di simulare le macchine di Turing. Quindi un semplice problema di verifica quale è la raggiungibilità diventa indecidibile. In [B1], si introduce un nuovo modello per programmi ricorsivi concorrenti: la macchina a stati ricorsiva comunicante (RSM comunicante). Una RSM comunicante estende una macchina a stati ricorsiva (RSM) permettendo una forma ristretta di concorrenza: uno stato di un modulo può essere raffinato ricorsivamente in una collezione finita di moduli (che funzionano in parallelo). Tuttavia per mantenere il modello decidibile, la comunicazione tra moduli è consentita solo tra moduli invocati con la stessa operazione di fork. In [B1], si studia il model-checking di una CRSM rispetto a specifiche espresse in una logica temporale che estende CARET con un operatore parallelo. Questa nuova logica viene denominata CONCARET. L'algoritmo proposto per risolvere questo problema richiede tempo esponenziale nella taglia della formula e nel massimo numero di moduli che possono essere invocati simultaneamente. Essendo il model-checking per CARET già EXPTIME-completo, si ha che il model-checking di CRSM rispetto a specifiche espresse nella logica CONCARET è EXPTIME-completo.

Logica Temporale:

La Logica Temporale è un utile formalismo per la specifica e la verifica dei sistemi. Nelle logiche temporali tradizionali è possibile formulare semplicemente asserzioni rispetto all'ordinamento temporale degli eventi, come ad esempio "ogni richiesta p deve essere seguita da una risposta q ". Tra queste logiche, LTL e CTL sono sicuramente tra le più studiate ed utilizzate. In alcune circostanze però, questo formalismo risulta essere insufficiente e, di conseguenza, sono state introdotte alcune logiche temporali quantitative o real-time ("ogni p è seguita da una q entro 5 passi"), tra cui l'estensione real-time di CTL, detta TCTL. Il problema del model-checking in TCTL è decidibile mentre il problema della soddisfacibilità non lo è.

In [A12] si affronta il problema di introdurre una logica temporale real-time di tipo branching-time con problema della soddisfacibilità decidibile e che, comunque, sia sufficientemente espressiva. Viene, così, introdotta la logica STCTL che estende CTL introducendo il tempo esplicitamente ma che differisce da TCTL sia per la sintassi che per la semantica. Infatti, l'uso dell'uguaglianza nei vincoli temporali non è consentito e la semantica è definita rispetto ad una struttura più semplice. La soddisfacibilità delle formule in STCTL è decidibile e la prova di questo risultato è ottenuta mediante riduzione al problema del vuoto nella teoria degli automi su ω -alberi temporizzati. Viene, inoltre, presentato un approccio al problema della verifica per sistemi real-time di tipo model-checking, basato sulla nuova logica introdotta. In [B18] viene provato che se si restringe sintatticamente TCTL impedendo l'uso dell'uguaglianza nei vincoli temporali, si ottiene una logica decidibile. Il risultato è ottenuto riducendo il problema della soddisfacibilità in questo frammento di TCTL al problema della soddisfacibilità in STCTL con semantica ristretta ad intervalli del tipo chiuso a sinistra e aperto a destra. Viene così colmato il gap che c'era con le logiche temporali lineari con tempo denso, dove è nota la decidibilità di MITL con la restrizione ai soli intervalli non singolari nei vincoli temporali.

Pur ammettendo asserzioni più forti rispetto alla logica tradizionale, il model checking nelle logiche real-time fornisce solo risposte del tipo "SI/NO". Cosa accadrebbe se ad esempio non sapessimo che 5 è il tempo di risposta adeguato e quindi volessimo determinare un valore appropriato della costante da usare con l'operatore temporale considerato?

In [A11] si fornisce uno strumento per effettuare questo tipo di misure sui sistemi. Viene introdotta la logica PLTL come estensione di LTL ottenuta accoppiando agli operatori temporali dei vincoli temporali contenenti variabili a valori nei numeri naturali ("per almeno x passi p è vera"). In questo tipo di logica la domanda a cui si vuole rispondere è quindi: "per quali valori dei parametri la formula è soddisfacibile?"

Dato un sistema K ed una formula $\varphi(x_1, \dots, x_k)$ è possibile decidere se esiste una valutazione di x_1, \dots, x_k per la quale K soddisfa la proprietà φ e determinare le valutazioni che soddisfano alcuni criteri di ottimalità. In particolare, è possibile individuare valutazioni che minimizzano (o massimizzano) il massimo (o il minimo) dei parametri. I corrispondenti algoritmi presentano la stessa complessità del model checking in LTL (PSPACE) e si dimostra che la sintassi scelta per PLTL è al limite della decidibilità per le logiche temporali parametriche, nel senso che, naturali estensioni di questa sintassi portano a problemi di misurazione, del tipo esposto, indecidibili.

Teoria dei Giochi e Sintesi:

Problemi di sintesi relativi alla progettazione di controllori che soddisfano alcuni requisiti

di correttezza come pure l'analisi di componenti in maniera modulare possono essere formulati come il problema di determinare una strategia vincente in un gioco a due giocatori. Di seguito si riportano i risultati ottenuti in questo ambito.

a) La progettazione e modellazione compositiva dei sistemi reattivi richiede che ogni componente sia vista come un sistema aperto, cioè un sistema che interagisce con l'ambiente e il cui comportamento dipende sia dallo stato del sistema che dall'ambiente. In questa situazione il problema di decisione basilare è determinare l'esistenza di una strategia vincente in un gioco con due partecipanti. In [A4] viene presentato un nuovo algoritmo in spazio $O(d \log n)$ per risolvere giochi alla Büchi, dove n è il numero dei vertici e d è la massima distanza nel grafo di transizione. L'algoritmo viene quindi utilizzato per risolvere giochi con condizione di accettazione espressi con formule in alcuni frammenti di LTL per i quali si forniscono generatori deterministici alla Büchi di taglia asintoticamente ottimale. La complessità dei giochi con condizioni di vincita espresse in LTL è anche studiata in [B6]. Un interessante risultato ottenuto è che se si considerano condizioni di vincita nel frammento di LTL con l'operatore "until" sostituito dal meno espressivo "eventually" e privo dell'operatore "next", i corrispondenti giochi sono 2EXPTIME-hard come nel caso generale. Quindi a differenza del model-checking e della soddisfacibilità per i giochi restringere le condizioni di vincita a questo frammento non comporta una riduzione della complessità computazionale del problema. In [B6] si prova anche che per il frammento di LTL con la sola modalità "eventually", insieme a congiunzioni e disgiunzioni, decidere i giochi è EXPSPACE-hard. Questo risultato chiude il gap computazionale su questa classe di giochi per i quali l'appartenenza alla classe EXPSPACE era stata data in [A4]. Inoltre, in [B6] si mostra che se la condizione di vincita è una combinazione booleana di formule del tipo "eventually p " e "infinitely often P ", dove p è una formula di stato, allora i giochi possono essere decisi in spazio polinomiale, e si mostra anche che questo risultato è completo, nel senso che riusciamo a provare la sua PSACE-hardness. Le condizioni espresse in questo frammento corrispondono a specifiche di safety e reachability su grafi aumentate con condizioni di fairness per entrambi i giocatori.

b) I giochi temporizzati possono essere modellati attraverso un automa temporizzato nondeterministico dove i simboli di input rappresentano le mosse del protagonista e il nondeterminismo modella le mosse dell'avversario. Ognuno dei partecipanti ha la capacità di scegliere la prossima mossa ed il tempo al quale questa verrà eseguita. Se si limita la possibilità di azione dell'avversario alla sola scelta dell'azione discreta (il tempo della mossa è deciso dal protagonista), questi giochi ammettono una traduzione diretta agli automi temporizzati su alberi [A9]. In [B12], si introduce invece un approccio basato sulla teoria degli automi temporizzati che consente di risolvere in maniera sistematica giochi temporizzati con condizioni di vincita espresse con formule di una logica temporale per le quali esiste un automa finito che ne accetta i modelli. Questo approccio viene quindi utilizzato per risolvere (con complessità ottimale) giochi temporizzati con condizioni di vincita espresse nelle logiche LTL e CTL, e attraverso le condizioni di accettazione standard per gli automi su oggetti infiniti, ovvero accettazione alla Büchi, alla Muller, e alla Rabin. In [B11], si risolvono giochi temporizzati in cui anche le condizioni di vincita sono temporizzate. Si osservi che al momento dell'ottenimento di questo risultato non vi erano nella letteratura scientifica lavori che presentassero soluzioni a giochi con condizioni di vincita temporizzate. La classe di condizioni di vincita che si considera è data dalle formule della logica TCTL. L'algoritmo

di soluzione proposto traduce il problema di decidere l'esistenza di una strategia vincente per il protagonista di un TCTL-game al problema del vuoto per automi su alberi infiniti temporizzati. La procedura ottenuta richiede tempo esponenziale e quindi è ottimale visto che giochi temporizzati con condizioni di vincita più semplici (raggiungibilità di un insieme di locazioni di vincita) sono noti essere EXPTIME-hard.

c) Il problema del controllo ottimale e sintesi di un automa temporizzato pesato può essere modellato attraverso un gioco temporizzato con due partecipanti dove un giocatore è il sistema e l'altro giocatore è l'ambiente. Si considera inoltre una funzione che assegna un costo ad ogni mossa del sistema, ad ogni transizione e al tempo di permanenza in una locazione. La condizione di vincita è data dal raggiungimento di alcuni stati di vincita (*reachability games*). Questa funzione assegna quindi un costo ad ogni strategia, ed una strategia è detta ottimale se è quella di costo minimo. In [B10] si risolve la sintesi di un controllo ottimale per automi temporizzati pesati aciclici. L'algoritmo presentato richiede tempo doppiamente esponenziale nella taglia dell'input ed è ottenuto attraverso la riduzione alla soddisfacibilità di una particolare classe di formule della logica del primo ordine dei reali con addizione e relazione d'ordine.

d) In [A2] si studiano i giochi su grafi gerarchici e ricorsivi che possono modellare il flusso di controllo in programmi sequenziali con chiamate ricorsive di funzioni. Sebbene questi giochi possono essere visti come i giochi push-down noti in letteratura, la naturale nozione di vincita in questo contesto richiede che le strategie siano modulari nel senso che possono usare solo una memoria locale (giochi modulari): le scelte all'interno di un modulo non possono tener conto del contesto in cui il modulo stesso è invocato. Mentre la raggiungibilità nei giochi push-down (con memoria globale) è nota essere EXPTIME-completo, in [A2] si prova che la raggiungibilità nei giochi modulari è NP-completo. Viene inoltre data un algoritmo che risolve i giochi modulari calcolando iterativamente (fixpoint computation) l'insieme dei nodi da cui il protagonista ha una strategia vincente. La procedura ottenuta nel caso peggiore converge in un numero di iterazioni esponenziale nel numero di valori che possono essere restituiti dai vari moduli (numero di uscite dei moduli). Se la strategia in un modulo non dipende dalla storia globale, ma può ricordare la storia delle invocazioni passate di questo modulo, cioè se la memoria è locale ma persistente, si mostra che la raggiungibilità diventa indecidibile.

I giochi trattati in [A2] sono stati estesi a condizioni di vincita su giocate di lunghezza infinita in [B7]. In questo lavoro, la condizione di vincita è data come una specifica ω -regolare sugli eventi osservabili. Si considera prima il caso in cui la specifica è data come un automa deterministico alla Büchi. Si prova che il problema considerato è decidibile e si presenta una soluzione basata sulla costruzione di un automa two-way alternante su alberi. L'algoritmo ottenuto impiega tempo al più esponenziale sia nella taglia della specifica che nel numero di uscite delle componenti. Si dimostra che il problema è EXPTIME-completo in generale, e NP-completo per specifiche di taglia fissata. Quindi si dimostra che gli stessi limiti di complessità si applicano anche se la specifica è data come un automa co-Büchi universale. Infine, per specifiche date come formule della logica temporale LTL, si ottiene un algoritmo di sintesi che richiede tempo doppiamente esponenziale nella taglia della formula e esponenziale nel numero di uscite delle componenti.

Model-checking e sicurezza:

In [A7], si propone l'uso degli automi temporizzati pesati studiati in [A6,B10, B16] per analizzare proprietà collegate alla sicurezza dei sistemi. Si descrive uno scenario dove un partecipante (Alice) vuole inviare un messaggio ad un altro partecipante (Bob) attraverso una rete di computer e vuole mantenere segreto il contenuto del messaggio fino ad una certa scadenza temporale. Si assume che chiunque voglia tentare di decodificare il messaggio può usare solo risorse locali ai nodi interessati dalla trasmissione del messaggio. Per raggiungere il suo obiettivo Alice prima di spedire il messaggio lo cifra secondo lo schema introdotto da Rivest, Shamir and Wagner noto con il nome di timed-release crypto. In questo scenario, Alice consegna direttamente il messaggio a Bob, il quale per decifrare il messaggio deve ricostruire la chiave facendo un numero di operazioni deciso da Alice al momento della cifratura. Quindi, conoscendo la potenza computazionale del computer usato da Bob, Alice può determinare il tempo necessario a quest'ultimo per scoprire il messaggio con una semplice moltiplicazione. Nello scenario considerato in [A7], invece, si ha una rete con diversi computer, ognuno con una diversa capacità computazionale ed con un diverso intervallo di tempo a disposizione per il calcolo. Si mostra che gli automi temporizzati pesati permettono di rappresentare questo scenario in maniera molto naturale: il sottostante automa temporizzato cattura le caratteristiche della rete e i pesi sulle locazioni esprimono il numero di operazioni per unità di tempo che possono essere eseguite localmente. Quindi, il problema di sicurezza sopra esposto si riduce al problema di determinare il minimo ed il massimo costo sui run di una durata fissata. In [A7] si presentano soluzioni a questi problemi basate sulla riduzione degli stessi a problemi analoghi su grafi direzionati pesati.

Misurazioni di Dependability:

Sono stati condotti studi inerenti la valutazione delle misure di dependability di un software di natura iterativa. Viene definito un nuovo modello in grado di considerare sia le correlazioni tra input di iterazioni consecutive che gli effetti di sequenze di fallimenti consecutivi. Differentemente da modelli precedentemente proposti, questo modello si basa sulle probabilità stazionarie (successo, fallimento benigno e fallimento catastrofico) e su quelle rappresentanti la correlazione. La maggiore facilità nell'ottenere le conoscenze di base per il suo utilizzo rispetto ai modelli precedentemente presentati, lo rendono più utile e maggiormente applicabile. Il modello introdotto è stato poi risolto al fine di ottenere espressioni delle misure di dependability considerate (reliability e performability) e si è condotta, infine, un'analisi della sensibilità delle misure di dependability rispetto ai parametri del modello [A15].

Sincronizzazione di processori:

Nell'ambito dello studio di tecniche per la progettazione e la trasformazione di sistemi, sono stati studiati problemi inerenti la sincronizzazione di processi. L'approccio seguito si basa sulla rappresentazione di reti via automi cellulari. Particolare attenzione è stata data al problema della sincronizzazione di una linea di n processori identici: il cosiddetto "Firing Squad Synchronization Problem" (FSSP). Questo problema fu introdotto nel 1962 da Moore e da allora molte soluzioni sono state presentate. Il problema di determinare una soluzione in tempo minimale con la limitazione per coppie di processori adiacenti in una linea di poter scambiare un solo bit di informazione è stato proposto e risolto da Mazoyer. In [A14] si è condotto uno studio sull'FSSP con queste limitazioni che ha portato all'introduzione di alcune operazioni di composizione di soluzioni ed alla caratterizzazione di alcune significative

famiglie di soluzioni in tempo non minimale. Dal punto di vista teorico, l'interesse nelle soluzioni non minimali in tempo è collegato alla composizione di automi cellulari. In [B19] si pone il problema della sincronizzazione di array di forma quadratica contenenti $(n \times n)$ processori. Come prima, processori adiacenti nella griglia possono scambiare solo un bit di informazione. Si estendono a queste reti di automi cellulari alcune tecniche precedentemente usate nel caso lineare. In [A13] si completa la ricerca cominciata in [B19] fornendo un approccio compositivo all'ottenimento di sincronizzazioni per array rettangolari di $(m \times n)$ processori. Infine, tutti i risultati ottenuti nel caso lineare sono estesi agli automi cellulari bidimensionali.

In [A16] si affronta il problema della sincronizzazione di processori collegati in modo che tra due processori il flusso dell'informazione sia monodirezionale. In questo caso, di solito si usano architetture di tipo toroidale, affinché ogni processore possa comunque ricevere (con un certo ritardo) informazioni da ogni altro processore nella rete. In particolare, sono individuati i limiti inferiori ai tempi delle soluzioni e sono forniti degli algoritmi di sincronizzazione in tempo minimale per il caso lineare (anello) e bidimensionale (matrice toroidale).

In [A1] si studia in maniera sistematica il problema della sincronizzazione in tempo non minimale su reti di processori in forma di linee e matrici bidimensionali con i due tipi di limitazioni descritte precedentemente. In questo articolo si richiamano i risultati studiati nei precedenti articoli e si presenta una notazione unificante. L'approccio compositivo basato sul concetto di segnale è ulteriormente studiato ed arricchito con nuovi risultati. Infine, si provano limiti inferiori alle soluzioni in tempo minimale per tutte le classi di reti che vengono considerate.

In [B5] si presenta una nuova soluzione all'FSSP su array bidimensionali. La nostra soluzione è ottimale sia rispetto al tempo che al numero di bit che due processori adiacenti nella rete possono scambiare ad ogni passo. Infatti, partendo da un processore situato in un angolo, un array di $n \times n$ processori viene sincronizzato in tempo $2n - 1$ consentendo ad ogni processore di inviare ai suoi vicini un solo bit. L'idea principale è di ottenere delle divisioni dell'array in sotto-reti a forma di L con angolo sulla diagonale principale e quindi procedere a sincronizzare ogni sotto-rete a L in completo isolamento rispetto alle altre. In aggiunta, si dimostra che la tecnica sviluppata e i risultati ottenuti permettono di risolvere in tempo ottimale l'FSSP per molte altre varianti su reti a forma di griglia (facendo partire la sincronizzazione da tutti gli angoli della griglia contemporaneamente), griglia circolare e anelli.

Tesi di Dottorato [C2]

Advances in Finite Automata and Temporal Logic for System Verification
(relatore Prof.ssa Margherita Napoli)

L'obiettivo di questa tesi è l'ottenimento di nuovi risultati nel campo dei metodi formali per la verifica automatica dei sistemi. Un primo risultato presentato nella tesi riguarda lo studio della teoria degli automi su ω -oggetti e ω -oggetti temporizzati. Si introducono gli automi temporizzati su alberi infiniti sia nella versione deterministica che non deterministica e con criteri di accettazione alla Büchi ed alla Muller. In particolare, sono state studiate le relazioni tra le classi di linguaggi individuate e con le corrispondenti classi di linguaggi non

temporizzati. Inoltre, per le classi di linguaggi introdotte sono stati risolti alcuni problemi di decisione (vuoto, equivalenza, equivalenza “debole”) e provate alcune proprietà di chiusura (unione, intersezione, complemento, concatenazione ed ω -iterazione). Un secondo aspetto trattato nella tesi ha riguardato l’introduzione della logica real-time branching STCTL e lo studio dei corrispondenti problemi di decisione. STCTL è una logica temporale real-time di tipo branching-time che estende CTL. La principale motivazione per questa nuova logica è la definizione di una logica temporale real-time e branching-time decidibile. L’unica estensione real-time di CTL, antecedente STCTL, è TCTL. Il problema del model-checking in TCTL è decidibile mentre il problema della soddisfacibilità non lo è. STCTL differisce da TCTL sia per la sintassi (mancanza dell’uguaglianza nei vincoli temporali) che per la semantica (definita sugli alberi temporizzati anziché su una struttura ad albero densa). Il risultato di decidibilità in STCTL è ottenuto mediante riduzione al problema del vuoto nella teoria degli automi su ω -alberi temporizzati. Un ultimo aspetto trattato nella tesi è l’estensione della logica temporale LTL con parametri (PLTL). Pur ammettendo asserzioni più forti rispetto alla logica tradizionale, il model checking nelle logiche real-time fornisce solo risposte del tipo “SI/NO”. L’introduzione dei parametri consente di avere informazioni sui possibili valori delle costanti che si devono usare in un sistema. PLTL è definita estendendo LTL accoppiando agli operatori temporali dei vincoli temporali contenenti variabili a valori nei numeri naturali. In questo tipo di logica il model-checking diventa il problema di stabilire per quali valori dei parametri di una formula il sistema costituisce un modello per la stessa. Nella tesi si dimostra che è possibile decidere l’esistenza di tali valori per i parametri e determinare le valutazioni che soddisfano alcuni criteri di ottimalità. Si prova, inoltre, che i corrispondenti algoritmi presentano la stessa complessità del model-checking in LTL (PSPACE) e che la sintassi scelta per PLTL sta al limite della decidibilità per le logiche temporali parametriche, nel senso che naturali estensioni di questa sintassi portano a problemi di misurazione, tipo quelli esposti, indecidibili. La ricerca condotta nella tesi è contenuta negli articoli [A12,A11,B17,B20].

Tesi di PhD [C1]

Verification of Reactive Systems and Decision Problems in Temporal Logic
(relatore Prof. Rajeev Alur)

In questa tesi si studiano alcuni problemi di decisione della teoria degli automi e della logica temporale. Il primo problema che si considera è il problema della raggiungibilità ottimale per automi temporizzati, rispetto a una generica funzione di costo lineare nel tempo (*weighted timed automata*). La nostra soluzione consiste nel ridurre questo problema alla determinazione dei cammini minimi in un grafo direzionato con pesi parametrici. L’algoritmo sviluppato impiega tempo doppiamente esponenziale quando si considera una generica regione dello spazio degli stati come regione di stati iniziali, e tempo esponenziale se si considera un unico stato iniziale. Il secondo problema che viene studiato nella tesi consiste in una estensione della sintassi degli automi ibridi lineari senza però rinunciare alla poliedricità dei flussi, cioè dato un insieme di stati descritto da un poliedro si vuole che l’insieme degli stati che possono essere raggiunti al passare del tempo sia ancora un poliedro. Viene introdotta quindi una classe di automi ibridi in cui i flussi sono descritti da flussi che sono dipendenti dallo stato di ingresso in una locazione. Un altro problema

studiato nella tesi è la determinazione di algoritmi ottimali per la soluzione di giochi a due giocatori con condizione di vincita espressa attraverso la logica LTL (linear temporal logic). È noto che determinare l'esistenza di una strategia vincente per questi giochi è un problema 2Exptime-completo. Noi analizziamo frammenti di LTL per cui il problema ha una complessità inferiore e proponiamo un algoritmo di soluzione basato su una riduzione ai giochi alla Büchi. Il passo principale di questa riduzione è la traduzione di una formula LTL in un automa deterministico alla Büchi. Proviamo che la nostra costruzione è ottimale rispetto alla taglia e alla distanza massima (lunghezza del cammino semplice più lungo nel grafo di transizione). Quindi diamo un algoritmo in spazio $O(d \log(n))$ per risolvere i giochi alla Büchi con n vertici e distanza massima d . Un ultimo contributo di questa tesi riguarda la soddisfacibilità delle formule nella logica TCTL (timed computation tree logic). La semantica di TCTL è definita su alberi densi e la soddisfacibilità è nota essere indecidibile anche se ci restringiamo ai soli alberi densi ottenuti da grafi temporizzati (finita soddisfacibilità). Ci sono due possibili cause per questa indecidibilità: il dominio di tempo denso e l'uguaglianza nei vincoli di clock di TCTL. Noi proviamo che se l'uguaglianza non è consentita nei vincoli di clock la soddisfacibilità di una formula TCTL può essere ridotta al problema del vuoto per gli automi temporizzati su alberi infiniti, e quindi è decidibile. La ricerca condotta nella tesi è contenuta negli articoli [A9,B14,B16,B18,B21].

5 Elenco delle Pubblicazioni

Riviste Internazionali:

- A1. “Different Time Solutions for the Firing Squad Synchronization Problem on Basic Grid Networks”, accettato per la pubblicazione su *Theoretical Informatics and Applications*, 2006.
(Con J. Gruska, M. Napoli, M. Parente)
- A2. “Modular Strategies for Recursive Game Graphs”, *Theoretical Computer Science*, 354, pp. 230–249, 2006.
(Con R. Alur e P. Madhusudan)
- A3. “Weak Muller Acceptance Conditions for Tree Automata”, *Theoretical Computer Science*, 332 (1-3), pp. 233–250, 2005.
(Con M. Napoli e A. Murano)
- A4. “Deterministic Generators and Games for LTL Fragments”, *ACM Transactions on Computational Logic*, 5 (1), pp. 1–25, 2004.
(Con R. Alur)
- A5. “Polyhedral Flows in Hybrid Automata”, *Formal Methods in System Design*, 24 (3), pp. 261–280, 2004.
(Con R. Alur e S. Kannan)

- A6. “Optimal Paths in Weighted Timed Automata”, *Theoretical Computer Science*, 318 (3), pp. 297–322, 2004.
(Con R. Alur e G.J. Pappas)
- A7. “Model-checking the Secure Release of a Time-locked Secret over a Network”, *Electronic Notes in Theoretical Computer Science*, 99, pp. 229–243, 2004.
(Con A. Murano e D. Parente)
- A8. “Deterministic Finite Automata with Recursive Calls”, *Information Processing Letters*, 87(4), pp. 187–193, 2003.
(Con J. Gallier e S. Mukhopadhyay)
- A9. “Finite Automata on Timed ω -trees”, *Theoretical Computer Science*, 293 (3), pp. 479–505, 2003.
(Con M. Napoli)
- A10. “Automata-based Representations for Infinite Graphs”, *Theoretical Informatics and Applications*, 35 (4), pp. 311–330, 2001.
(Con M. Napoli)
- A11. “Parametric temporal logic for model measuring”, *ACM Transactions on Computational Logic*, 2 (3), pp. 388–407, 2001.
(Con R. Alur, K. Etessami e D. Peled)
- A12. “Timed Tree Automata with an Application to Temporal Logic”, *Acta Informatica*, 38 (2), pp. 89–116, 2001.
(Con M. Napoli)
- A13. “A Compositional Approach to Synchronize Two Dimensional Networks of Processor”, *Theoretical Informatics and Applications*, 34 (6), pp. 549–564, 2000.
(Con M. Napoli e D. Parente)
- A14. “Synchronization of a line of identical processors at a given time”, *Fundamenta Informaticae*, 34 (1,2), pp. 103–128, 1998.
(Con M. Napoli e D. Parente)
- A15. “Modelling the effects of input correlation in iterative software”, *Reliability Engineering and System Safety*, 57 (3), pp. 189–202, 1997.
(Con A. Bondavalli, S. Chiaradonna e F. Di Giandomenico)
- A16. “Synchronization of One-Way Connected Processors”, *Complex Systems*, 10 (4), pp. 239–255, 1996.
(Con M. Napoli e D. Parente)
- A17. “Parallel Word Substitution”, *Fundamenta Informaticae*, 27, pp. 27–36, 1996.
(Con M. Napoli e D. Parente)

Atti di Conferenze Internazionali:

- B1. “Verification of well-formed Communicating Recursive State Machines”, Proc. of the 7th International Workshop on Verification, Model Checking, and Abstract Interpretation, **VMCAI’06**, Charleston, South Carolina, USA, January 8 - 10, 2002. **Lecture Notes in Computer Science** (2006).
(*Con L. Bozzelli e A. Peron*)
- B2. “Decidability and Infinite Precision in Timed Automata”, Proc. of the 43rd Congresso Annuale AICA, Udine, Italia, October 5-7, 2005. Parte 2, pp. 465–474.
- B3. “Perturbed Timed Automata”, Proc. of the 8th International Workshop on Hybrid Systems: Computation and Control, **HSCC’05**, Zurich, Switzerland, March 9-11, 2005. **Lecture Notes in Computer Science** **3414** (2005), pp. 70–85.
(*Con R. Alur e P. Madhusudan*)
- B4. “Reasoning about co-Buchi Tree Automata”, Proc. of the 1st International Colloquium on Theoretical Aspects of Computing, **ICTAC’04**, Guiyang, China, September 20 - 24, 2004. **Lecture Notes in Computer Science** **3407** (2005), pp. 527–542.
(*Con A. Murano*)
- B5. “Optimal Time & Communication Solutions of Firing Squad Synchronization Problems on Square Arrays, Toruses and Rings”, Proc. of the 8th International Conference on Developments in Language Theory, **DLT’04**, Auckland, New Zealand, December 13 - 17, 2004. **Lecture Notes in Computer Science** **3340** (2004), pp. 200–211.
(*Con J. Gruska e M. Parente*)
- B6. “Playing Games with Boxes and Diamonds”, Proc. of the 14th International Conference on Concurrency Theory, **CONCUR’03**, Marseille, France, September 3-5, 2003. **Lecture Notes in Computer Science** **2761** (2003), pp. 128–143.
(*Con R. Alur e P. Madhusudan*)
- B7. “Modular Strategies for Infinite Games on Recursive Graphs”, Proc. of the 15th International Conference on Computer-Aided Verification, **CAV’03**, Boulder, Colorado, USA, July 8 - 12, 2003. **Lecture Notes in Computer Science** **2725** (2003), pp. 67–79.
(*Con R. Alur e P. Madhusudan*)
- B8. “Hierarchical and Recursive State Machines with Context-Dependent Properties”, Proc. of the 30th International Colloquium on Automata, Languages and Programming, **ICALP’03**, Eindhoven, The Netherlands, June 30 - July 4, 2003. **Lecture Notes in Computer Science** **2719** (2003), pp. 776–789.
(*Con M. Napoli, M. Parente e G. Parlato*)
- B9. “Modular Strategies for Recursive Game Graphs”, Proc. of the 9th International Conference on Tools and Algorithms for the Construction and the Analysis of Systems, **TACAS’03** (a member conference of the European Joint Conferences on Theory and Practice of Software, **ETAPS’03**), Warsaw, Poland, April 7-11, 2003. **Lecture Notes**

- in **Computer Science 2619** (2003), pp. 363–378.
(Con R. Alur e P. Madhusudan)
- B10. “Optimal-Reachability and Control for Acyclic Weighted Timed Automata”, Proc. of the 2nd IFIP International Conference on Theoretical Computer Science, **IFIP TCS’02**, Montreal, Canada, August 25-30, 2002. **Kluwer Academic Publishers**, pp. 485–497.
(Con S. Mukhopadhyay e A. Murano)
- B11. “Dense Real-time Games”, Proc. of the 17th Annual IEEE Symposium on Logic in Computer Science, **LICS’02**, Copenhagen, Denmark, July 22-25, 2002. **IEEE Computer Society Press**, pp. 167–176.
(Con M. Faella e A. Murano)
- B12. “Automata-theoretic Decision of Timed Games”, Proc. of the 3rd International Workshop on Verification, Model Checking, and Abstract Interpretation, **VMCAI’02**, Venezia, Italia, January 21 - 22, 2002. **Lecture Notes in Computer Science 2294** (2002), pp. 94–108.
(Con M. Faella e A. Murano)
- B13. “Weak Muller Acceptance Conditions for Tree Automata”, Proc. of the 3rd International Workshop on Verification, Model Checking, and Abstract Interpretation, **VMCAI’02**, Venezia, Italia, January 21 - 22, 2002. **Lecture Notes in Computer Science 2294** (2002), pp. 240–254.
(Con M. Napoli e A. Murano)
- B14. “Deterministic Generators and Games for LTL Fragments”, Proc. of the 16th Annual IEEE Symposium on Logic in Computer Science, **LICS’01**, Boston, Massachusetts, USA, June 16 - 19, 2001. **IEEE Computer Society Press**, pp. 291–300.
(Con R. Alur)
- B15. “Firing Squad Synchronization Problem on Bidimensional Cellular Automata with Communication Constraints”, Proc. of the 3rd International Conference on Machines, Computations and Universality, **MCU’01**, Chisinau, Moldova, May 23-27, 2001. **Lecture Notes in Computer Science 2055** (2001), pp. 264–275.
(Con M. Napoli e D. Parente)
- B16. “Optimal Paths in Weighted Timed Automata”, Proc. of the 4th International Workshop on Hybrid Systems: Computation and Control, **HSCC’01**, Roma, Italia, March 28-30, 2001. **Lecture Notes in Computer Science 2034** (2001), pp. 49–62.
(Con R. Alur e G.J. Pappas)
- B17. “A Model of Finite Automata on Timed ω -trees”, Proc. of the 7th Computing: The Australasian Theory Symposium, **CATS’01**, Gold Coast, Queensland, Australia, January 29-February 2, 2001. **Electronic Notes in Theoretical Computer Science 42**, Elsevier Science, 2001.
(Con M. Napoli)

- B18. “A Decidable Dense Branching-time Temporal Logic”, Proc. of the 20th Conference on the Foundations of Software Technology and Theoretical Computer Science, **FSTTCS’00**, New Delhi, India, December 13-15, 2000. **Lecture Notes in Computer Science 1974** (2000), pp. 139–150.
(*Con M. Napoli*)
- B19. “Compositionality in the Synchronization of Processors exchanging a minimal amount of information”, Proc. of the 5th Workshop su Sistemi Distribuiti: Algoritmi, Architetture e Linguaggi, **WSDAAL’00**, Ischia, Italy, 18-20 Settembre, 2000.
(*Con M. Napoli e M. Parente*)
- B20. “Parametric Temporal Logic for Model Measuring”, Proc. of the 26th International Colloquium on Automata, Languages and Programming, **ICALP’99**, Prague, Czech Republic, July 11-15, 1999. **Lecture Notes in Computer Science 1644** (1999), pp. 159–168.
(*Con R. Alur, K. Etessami e D. Peled*)
- B21. “Polyhedral Flows in Hybrid Automata”, Proc. of the 2th International Workshop on Hybrid Systems: Computation and Control, **HSCC’99**, J. Van Schuppen and F. Vaandrager eds., Berg en Dal, The Netherlands, March 29-31, 1999. **Lecture Notes in Computer Science 1569** (1999), pp. 5–18.
(*Con R. Alur e S. Kannan*)
- B22. “Representing Hyper-Graphs by Regular Languages”, Proc. of the 23th International Symposium on Mathematical Foundations of Computer Science, **MFCS’98**, Lubos Brim, Jozef Gruska and Jirm Zlatuska eds., Brno, Czech Republic, August 24-28, 1998. **Lecture Notes in Computer Science 1450** (1998), pp. 571–579.
(*Con M. Napoli*)
- B23. “Synchronization of 1-Way Connected Processors”, Proc. of the 11th International Symposium on Fundamentals of Computation Theory, **FCT’97**, B.Chelbus and L.Czaja eds., Krakow, Poland, September 1 - 3, 1997. **Lecture Notes in Computer Science 1279** (1997), pp. 93–304.
(*Con M. Napoli e D. Parente*)
- B24. “Synchronization of a line of identical processors at a given time”, Proc. of the Colloquium on Trees in Algebra and Programming, **CAAP’97**, in the framework of the 7th International Joint Conference on the Theory and Practice of Software Development, **TAPSOFT’97**, M.Bidoit and M.Dauchet eds., Lille, France, April 14 - 18, 1997. **Lecture Notes in Computer Science 1214** (1997), pp. 405–416.
(*Con M. Napoli e D. Parente*)
- B25. “Dependability of Iterative Software: a Model for Evaluating the Effects of Input Correlation”, Proc. of the 14th International Conference on Computer Safety, Reliability and Security, **SAFECOMP’95**, Belgirate, Italy, **Springer**, Berlin, 1995, pp. 489–503.
(*Con A. Bondavalli, S. Chiaradonna e F. Di Giandomenico*)

Tesi:

- C1. S. La Torre, “Verification of Reactive Systems and Decision Problems in Temporal Logic”, PhD Thesis, Department of Computer and Information Science, University of Pennsylvania, Philadelphia, USA, pp. 1–236, 2001.
- C2. S. La Torre, “Advances in Finite Automata and Temporal Logic for System Verification”, Tesi di Dottorato, Università di Napoli “Federico II”, pp. 1–132, 1999.
- C3. S. La Torre, “Automati su ω -alberi ed ω -alberi temporizzati”, Tesi di Laurea, Dipartimento di Informatica ed Applicazioni, Università di Salerno, 1994.

Rapporti Tecnici e Manoscritti:

- D1. “The Effects of Input Correlation on the Dependability of Iterative Software”, IEI-CNR, Pisa, Italy, Internal Report B4-24, May 1995.
(*Con A. Bondavalli, S. Chiaradonna e F. Di Giandomenico*)
- D2. “Approximating a Finite Metric by Tree Metrics”, Internal Report, University of Pennsylvania, 2000.
- D3. “Subclasses of Timed Automata with NP-complete Reachability Problem”, manoscritto, 2002.
(*Con R. Alur e S. Mukhopadhyay*)
- D4. “Timed Automata with Parametric Timed Constraints”, in preparazione.
(*Con R. Alur*)
- D5. “Compositional Verification of Timed Circuits”, in preparazione.
(*Con R. Alur e P. Madhusudan*)
- D6. “Game-based Decision of MITL Specifications”, in preparazione.
(*Con M. Faella e A. Murano*)

Dichiaro che il contenuto di questo curriculum corrisponde a verità ai sensi degli artt. 2 e 4 della legge 15/1968 e degli artt. 1 e 2 del D.P.R. 403/1998.

11 Maggio 2006

Salvatore La Torre