CURRICULUM VITAE E STUDIORUM DI
MARIA CHIARA MEO

NOTE BIOGRAFICHE

TITOLI DI STUDIO

- Diploma di maturità Scientifica, Liceo Scientifico Statale.
- Laurea in Scienze dell'Informazione, Università di Pisa. Votazione di 110/110 e lode. Titolo della tesi "Un calcolo Intensionale di insiemi in programmazione logica". Relatore Prof. Franco Turini.
- Dottorato di Ricerca in Informatica, Dipartimento di Informatica dell'Università di Pisa.

POSIZIONE ACCADEMICA

MACROSETTORE: 01/B Informatica
SETTORE CONCORSUALE: 01/B1 Informatica
SETTORE SCIENTIFICO DISCIPLINARE: INF/01 Informatica
QUALIFICA: Professore ordinario
ANZIANITÀ NEL RUOLO: da marzo 2011
SEDE UNIVERSITARIA: Università di Chieti-Pescara
STRUTTURA DI AFFERENZA: Dipartimento di Economia
POSIZIONI RICOPERTE PRECEDENTEMENTE NEL MEDESIMO ATENEO O IN ALTRI

- SETTEMBRE 2001 - MARZO 2011: Professore associato all'Università di Chieti-Pescara.
- GENNAIO 1995 - SETTEMBRE 2001: Ricercatore all'Università dell'Aquila.

PUBBLICAZIONI

RIVISTE INTERNAZIONALI

1. A. Bossi, M. Gabbrielli, G. Levi and M.C. Meo. A Compositional Semantics for Logic Programs. *Theoretical Computer Science* 122 (1-2): 3—47. Elsevier Science, Amsterdam 1994.

2. M. Gabbrielli, G. Levi and M.C. Meo. Observable Behaviors and Equivalences of Logic Programs. *Information and Computation* 122 (1): 1—29. Academic Press, New York and London, 1995.

3. A. Bossi, M. Bugliesi, M. Gabbrielli, G. Levi and M.C. Meo. Differential Logic Programs: Programming Methodologies and Semantics. *Science of Computer Programming* 27 (3): 217—262. Elsevier Science, Amsterdam, 1996.

4. M. Gabbrielli, G. Levi and M.C. Meo. Resultant Semantics for Prolog. *Journal of Logic and Computation* 6 (4): 491—522. Oxford University Press, Oxford, 1996.

5. M. Comini and M.C. Meo. Compositionality properties of *SLD*-derivations. *Theoretical Computer Science*, 211 (1-2):275—309. Elsevier Science, Amsterdam, 1999.

6. M. Comini, G. Levi, M.C. Meo and G. Vitello. Abstract Diagnosis. *Journal of Logic Programming* 39 (1-3): 43—93. Elsevier Science, Amsterdam, 1999.

7. F.S. de Boer, M. Gabbrielli and M.C. Meo. A Timed Concurrent Constraint Language. *Information and Computation* 161 (1): 45—83. Academic Press, New York and London, 2000.

8. M. Comini, G. Levi and M.C. Meo. A Theory of Observables for Logic Programs. *Information and Computation* 169 (1): 23–80. Academic Press, New York and London, 2001.

9. S. Etalle, M. Gabbrielli and M.C. Meo. Transformations of CCP programs. *ACM Transactions on Programming Languages and Systems (TOPLAS)* 23 (3): 304—395. ACM Press, New York, 2001.

10. F.S. de Boer, M. Gabbrielli and M.C. Meo. A Timed Linda Language and its Denotational Semantic. *Fundamenta Informaticae* 63 (4): 309–330. IOS Press, 2004. ISSN 0169-2968.

11. F.S. de Boer, M. Gabbrielli and M.C. Meo. Proving Correctness of Timed Constraint Programs. *ACM Transactions on Computational Logic (TOCL)* 5 (4): 706–731. ACM Press, New York, 2004.

12. M. Gabbrielli and M.C. Meo. A compositional Semantics for CHR. *ACM Transactions on Computational Logic (TOCL)* 10(2). ACM Press, New York. 2009.

13. M. Gabbrielli, J. Mauro, M.C. Meo and J. Sneyers. Decidability properties for fragments of CHR. *Theory and Practice of Logic Programming (TPLP)* 10(4-6): 611-626. Cambridge University Press. 2010. ISSN: 1471-0684.

14. C. Di Giusto, M. Gabbrielli and M. C. Meo: On the expressive power of multiple heads in CHR. *ACM Transactions on Computational Logic (TOCL)* 13(1): 6. ACM Press, New York. 2012. Codice DOI 10.1145/2071368.2071374

15. M. Gabbrielli, J. Mauro and M. C. Meo. The expressive power of CHR with priorities. *Information and Computation* 228: 62-82 Academic Press, New York and London, 2013. Codice DOI 10.1016/j.ic.2013.05.001.

16. M. Gabbrielli, M. C. Meo, P. Tacchella and H. Wiklicky. Unfolding for CHR programs. *Theory and Practice of Logic Programming (TPLP)* 15(3):264--311. Cambridge University Press, 2015. Codice DOI 10.1017/S1471068413000288.

17. S. Bistarelli, M. Gabbrielli, M. C. Meo and F. Santini. Timed soft concurrent constraint programs: an interleaved and a parallel approach. *Theory and Practice of Logic Programming (TPLP)* 15(6):743--782. Cambridge University Press, 2015. Codice DOI 10.1017/S1471068414000106.

18. G. Amato, M. C. Meo, F. Scozzari. Exploiting Linearity in Sharing Analysis of Object-oriented Programs. *Electronic Notes of Theoretical Computer Science* 322: 3-18, Elsevier Science, Amsterdam, 2016. Codice DOI 10.1016/j.entcs.2016.03.002.

19. G. Amato, S. Di Nardo Di Maio, M. C. Meo, F. Scozzari. Descending chains and narrowing on template abstract domains. *Acta Informatica* 55(6): 521- 545, Springer Berlin Heidelberg, 2018 (pubblicato online il 4 gennaio 2017). Codice DOI 10.1007/s00236-016-0291-0.

20. E. De Angelis, F. Fioravanti, M.C. Meo, A. Pettorossi and M. Proietti, Semantics and Controllability of Time-Aware Business Processes. *Fundamenta Informaticae* 165: 205- 244, IOS Press 2019.

21. G. Amato, M. C. Meo, F. Scozzari. On collecting semantics for program analysis. *Theoretical Computer Science*, 823: 1—25. Elsevier Science, Amsterdam, 2020.

CONFERENZE INTERNAZIONALI

22. A. Bossi, M. Gabbrielli, G. Levi, and M.C. Meo. Contributions to the Semantics of Open Logic Programs. In *Proc. Int'l Conference on Fifth Generation Computer Systems (FGCS 92)*: (570—590). IOS Press 1992. ISBN 90-5199-099-5.

23. M. Gabbrielli, G. Levi, and M.C. Meo. Observational Equivalences for Logic Programs. In Proc. *Joint Conference and Symposium on Logic Programming (JCSLP 92)*: (131—145). The MIT Press, Cambridge, Mass. 1992. ISBN 0-262-51064-2.

24. M. Gabbrielli and M.C. Meo. Fixpoint semantics for Partial answers and Call Patterns. In *Proc. Third Int'l Conference on Algebraic and Logic Programming (ALP 92)*. Lecture Notes in Computer Science, 632: (84—99). Springer-Verlag, Berlin, 1992. ISBN 3-540-55873-X.

25. A. Bossi, M. Bugliesi, M. Gabbrielli, G. Levi and M.C. Meo. Differential Logic Programming. In Proc. *Twentieth Annual ACM SIGACT/SIGPLAN Symposium on Principles of Programming Languages (POPL 93)*: (359—370). ACM Press, New York 1993. ISBN:0-89791-560-7

26. P. Bruscoli, F. Levi, G. Levi and M.C. Meo: Compilative Constructive Negation in Constraint Logic Programs. In Proc of 19th International Colloquium on Trees in Algebra and Programming (CAAP '94). Lecture Notes in Computer Science, 787: (52—67). Springer-Verlag, Berlin, 1994. ISBN 3-540-57879-X.

27. A. Bossi, M. Fabris and M.C. Meo. A Bottom-up Semantics for Constructive Negation. In Pro*c. of Eleventh Int'l Conference on Logic Programming (ICLP 94)*: (520—534). The MIT Press, 1994. ISBN 0-262-72022-1.

28. M. Comini, G. Levi and M.C. Meo: Compositionality in SLD-Derivations and their Abstractions. In *Logic Programming, Proceedings of the 1995 International Symposium (ILPS 1995)*: (561—575), The MIT Press, 1995, ISBN 0-262-62099-5.

29. M. Comini, G. Levi, M.C. Meo and G. Vitello. Proving properties of Logic Programs by Abstract Diagnosis. In Proc of *Fifth Int'l Workshop on Analysis and Verification of Multiple-Agent Languages, (LOMAPS 96)*, Lecture Notes in Computer Science, 1192: (22—50). Springer-Verlag, Berlin, 1996. ISBN 3-540-62503-8.

30. F.S. de Boer, M. Gabbrielli and M.C. Meo. Semantics and expressive power of a timed concurrent constraint language. In Proc. *Third Int'l Conf. on Principles and Practice of*

Constraint Programming (CP 97). Lecture Notes in Computer Science. 1330: (47—61). Springer-Verlag, Berlin, 1997.

31. S. Etalle, M. Gabbrielli and M.C. Meo. Unfold/Fold Transformations of CCP programs. In *Proc. of* 9th *International Conference on Concurrency Theory* (*CONCUR '98)*, Lecture Notes in Computer Science, 1466: (348—365). Springer-Verlag, Berlin, 1998. ISBN 3-540-64896-8.

32. M.C. Meo. On the Expressiveness of Concurrent Constraint Languages. In Proc of the Workshop on Object-Oriented Technology *(ECOOP'99)*, Lecture Notes in Computer Science, 1743: (261—265). Springer-Verlag, Berlin, 1999. ISBN 3-540-66954-X.

33. F.S. de Boer, M. Gabbrielli and M.C. Meo. A Timed Linda Language. In Proc. 4th *Conference on Coordination Models and Languages (COORDINATION 2000)*, Lecture Notes in Computer Science, 1906: (299—304). Springer-Verlag, Berlin, 2000. ISBN 3-540-41020-1.

34. F.S. de Boer, M. Gabbrielli and M.C. Meo. A Temporal Logic for reasoning about Timed Concurrent Constraint Programs. In Proc. of the 8th International Symposium on Temporal Representation and Reasoning, *TIME 01*: (227—233). IEEE Press, 2001. ISBN 1-58113-388-X

35. F.S. de Boer, M. Gabbrielli, and M.C. Meo. A Denotational Semantics for a Timed Linda Language. In Proc. of the 3rd International ACM SIGPLAN Conference on Principles and Practice of Declarative Programming *(PPDP 01)*: (28—36). ACM Press, New York, 2001. ISBN 1-58113-388-X.

36. F.S. de Boer, M. Gabbrielli, and M.C. Meo. Proving Correctness of Timed Concurrent Constraint Programs. In Proc. of the 5th International Conference on Foundations of Software Science and Computation Structures (*FOSSACS 02)*. Lecture Notes in Computer Science, 2303: (37—51), Springer-Verlag, Berlin, 2002. ISBN 3-540-43366-X.

37. G. Delzanno, M. Gabbrielli and M.C. Meo. Compositional Verification of Infinite State Systems. *Proc. of Logic Programming, 19th International Conference,(ICLP 2003)*. Lecture Notes in Computer Science, 2916: (47—48). Springer-Verlag, Berlin, 2003. ISBN 3-540-20642-6.

38. G. Delzanno, M. Gabbrielli and M.C. Meo: A compositional semantics for CHR. In Proc. of the 7th International ACM SIGPLAN Conference on Principles and Practice of Declarative Programming (*PPDP 05)*: (209—217). ACM Press, New York, 2005. ISBN 1-59593-090-6.

39. P. Tacchella, M. Gabbrielli and M.C. Meo. Unfolding in CHR with propagation rules. In Proc. of the 9th International ACM SIGPLAN Conference on Principles and Practice of Declarative Programming (*PPDP 07)*: (179—186). ACM Press, New York, 2007. ISBN 978-1-59593-769-8.

40. S. Bistarelli, M. Gabbrielli, M.C. Meo and F. Santini. Timed Soft Concurrent Constraint Programs. In Proc. of Coordination Models and Languages, 10th International Conference, (COORDINATION 2008). Lecture Notes in Computer Science, 5052:(50—66). Springer-Verlag, Berlin, 2008. ISBN 978-3-540-68264-6.

41. M. Gabbrielli, M.C. Meo and P. Tacchella. A compositional Semantics for CHR with propagation rules. In Special Issue on Constraint Handling Rules, Current Research Topics. Lecture Notes in Computer Science, 5388 (119—160). Springer-Verlag, Berlin, 2008. ISBN 978-3-540-92242-1. (Versione breve in Proc. of the 3rd Workshop on Constraint Handling Rules (CHR 2006): (93—107), 2006).

42. C. Di Giusto, M. Gabbrielli and M. C. Meo: Expressiveness of Multiple Heads in CHR. In Proc of the 35th Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM 2009). Lecture Notes in Computer Science, 5404: (205—216) Springer-Verlag, Berlin, 2009.

43. M. Gabbrielli, J. Mauro and M.C. Meo. On the expressive power of priorities in CHR. In Proc. of the 11th International ACM SIGPLAN Conference on Principles and Practice of Declarative Programming (*PPDP 09):* (267—276). ACM Press, 2009.

44. M. Gabbrielli, J. Mauro, M.C. Meo and J. Sneyers. Decidability properties for fragments of CHR. In Proc. of the 26th International Conference on Logic Programming (*ICLP 2010)*. 2010.

45. A. Bossi and M.C. Meo. Theoretical foundations and Semantics of Logic Programming. In. A 25-Year Perspective on Logic Programming: Achievements of the Italian Association for Logic Programming. Lecture Notes in Computer Science, 6125: (15—36) Springer-Verlag, Berlin, 2010. Codice DOI 10.1007/978-3-642-14309-0_2.

46. G. Amato, S. Di Nardo Di Maio, M.C. Meo and F. Scozzari. Narrowing Operators on Template Abstract Domains. In Proc. of Formal Methods, 20th International Symposium (FM 2015): Lecture Notes in Computer Science, 9109: (57—72) Springer-Verlag, Berlin, 2015. Codice DOI 10.1007/978-3-319-19249-9_5.

47. E. De Angelis, F. Fioravanti, M.C. Meo, A. Pettorossi and M. Proietti, Verification of Time-Aware Business Processes Using Constrained Horn Clauses. In Proc. of the 26th International Symposium on Logic-Based Program Synthesis and Transformation (LOPSTR) 2016. Lecture Notes in Computer Science, 10364: (38—55) Springer-Verlag, Berlin, 2016. Codice DOI 10.1007/978-3-319-63139-4_3.

48. E. De Angelis, F. Fioravanti, M.C. Meo, A. Pettorossi and M. Proietti, Verifying Controllability of Time-Aware Business Processes. In Proc. of Rule and Reasoning – International Joint Conference, RuleML+RR 2017. Lecture Notes in Computer Science, 10364: (103—118) Springer-Verlag, Berlin, 2017. Codice DOI 10.1007/978-3-319-61252-2_8.

49. G. Amato, M.C. Meo and F. Scozzari. A Taxonomy of Program Analysis. In Proc. of 19th Italian Conference on Theoretical Computer Science. CEUR Workshop proceedings, 2243: (213—217), 2018.

CAPITOLI DI LIBRI

50. A. Bossi, M. Gabbrielli, G. Levi and M.C. Meo. An OR-compositional Semantics for Logic Programs. In Jean-Marie Jaquet, editor, *Constructing Logic programs*. John Wiley & Sons Ltd, 1993. ISBN 0-471-93789-4.

ATTIVITÀ DI RICERCA

PARTECIPAZIONE A PROGETTI DI RICERCA E VISITE AD ALTRI ISTITUTI

Ha partecipato ai seguenti progetti di ricerca

- Progetto Finalizzato del C.N.R. “Sistemi Informativi e Calcolo Parallelo”.
- Progetto ESPRIT Basic Research Action Project “Integration”.
- Progetto ESPRIT “PARFORCE”.
- Cofin 1997: Tecniche formali per la specifica, l'analisi, la verifica, la sintesi e la trasformazione di sistemi software.
- Cofin 1999: Architetture Software e Linguaggi per Coordinare Componenti Distribuite e Mobili.
- Cofin 2001: Ragionamento su aggregati e numeri a supporto della programmazione e relative verifiche: dagli algoritmi di decisione alla programmazione con vincoli con multi-insiemi, insiemi e mappe.
- Cofin 2002: Verifica di sistemi reattivi basata su vincoli (COVER).
- Cofin 2004: AIDA - Interpretazione Astratta: Progettazione e Applicazioni.
- Prin 2005. Tecniche di Astrazione, Concorrenza e Vincoli Soft per Sicurezza Informatica e studio di Sistemi Informatici.

Ha visitato come Visiting Researcher il CWI di Amsterdam dal 20 Gennaio al 30 maggio 1994.

PARTECIPAZIONE A COMITATI DI PROGRAMMA E AD ORGANI DIRETTIVI

È stata membro dei comitati di programma delle seguenti conferenze

- Joint Conference on Declarative Programming AGP’97. Grado, Italia.
- Joint Conference on Declarative Programming AGP’98. Coruña, Spagna.
- Joint Conference on Declarative Programming AGP’99. L’Aquila, Italia (co-presidente).
- Joint Conference on Declarative Programming AGP’00. La Habana, Cuba (co-presidente).
- Convegno Italiano di Logica Computazionale CILC’04. Parma, Italia.
- PPDP’05, International Conference on Principles and Practice of Declarative Programming. Lisbona, Portogallo, 11-13 Luglio 2005
- PPDP 2009, International Conference on Principles and Practice of Declarative Programming. Coimbra, Portogallo, 7-9 Settembre 2009
- CHR 2010, Seventh International Workshop on Constraint Handling Rules, Edimburgo, Scozia, 20 Luglio 2010
- CILC 2011, 26-esimo Convegno Italiano di Logica Computazionale, Pescara, Italia, 31 Agosto-2 Settembre 2011.
- CILC 2012, 32-esimo Convegno Italiano di Logica Computazionale, Roma, Italia, 6-7 Giugno 2012.
- CHR 2013 10th International Workshop on Constraint Handling Rules, Berlino, Germania, 13 Luglio 2013
- CHR 2014, 11th International Workshop on Constraint Handling Rules, Vienna, Austria, 18 Luglio 2014
- CILC 2016, 31-esimo Convegno Italiano di Logica Computazionale, Milano, Italia, 20-22 Giugno 2016.

- ICTCS 2018, 19th Italian Conference on Theoretical Computer Science, Urbino, Italia, 18-20 Settembre 2018
- LOPSTR 2019, 29th International Symposium on Logic-Based Program Synthesis and Transformation, Porto, Portogallo, 8-10 ottobre 2019
- ECAI 2020, 24th European Conference on Artificial Intelligence, Santiago de Compostela, Spagna (svolto in modalità online) 29 Agosto-8 Settembre 2020.

È stata membro dell'organo direttivo dell'Associazione italiana di Programmazione Logica da dicembre 2003 a dicembre 2006.

Editore dei seguenti Atti di Conferenza

- Maria Chiara Meo, Manuel Vilares Ferro: 1999 Joint Conference on Declarative Programming, AGP'99, L'Aquila, Italy, September 6-9, 1999 APPIA-GULP-PRODE 1999
- Agostino Dovier, Maria Chiara Meo, Andrea Omicini: Declarative Programming - Selected Papers from AGP 2000 La Habana (CUBA), December 4-6, 2000. Electronic Notes in Theoretical Computer Science 48: (2001)

DESCRIZIONE DELL'ATTIVITÀ DI RICERCA

SEMANTICA DELLA PROGRAMMAZIONE LOGICA

La parte iniziale dell'attività di ricerca, svolta durante la scuola di dottorato è stata rivolta allo studio di semantiche composizionali per linguaggi logici (puri ed estesi) che permettono di modellare il comportamento osservabile dei programmi logici in modo più preciso di quanto non avvenisse con i modelli semantici esistenti.

L'approccio generale seguito ed i principali risultati ottenuti, contenuti nella tesi di dottorato e pubblicati in [1, 2, 4, 22, 23, 24, 50], riguardano la definizione di varie semantiche e la dimostrazione della loro correttezza e full abstraction. Tali risultati hanno fornito le basi semantiche per la realizzazione di strumenti di analisi statica e di verifica di correttezza dei programmi.
In particolare, in [1, 22] viene definita una semantica per i programmi logici *aperti* che è composizionale rispetto all'unione dei programmi. Come mostrato in [24], tale semantica può essere usata per modellare osservabili non standard, quali risposte parziali e call patterns quando non si considera la regola di calcolo. In [4] ed in [24] è inoltre definita una semantica di punto fisso che permette di modellare tali osservabili anche in presenza di una data regola di calcolo. Anche in questo caso, la rilevanza dello studio di queste semantiche si collega alla possibilità di definire strumenti di analisi che considerano il comportamento operazionale dei programmi nei linguaggi logici realmente implementati (e quindi con regola di calcolo)

Lo schema generale, definito inizialmente per il linguaggio logico puro, è esteso anche al caso di linguaggi logici con meccanismi di ereditarietà [3, 25]. L'intera costruzione è basata sulla nozione di *differential programs*, programmi logici annotati con dichiarazioni che stabiliscono le loro interfacce esterne e che permettono di modellare i vari meccanismi di ereditarietà statici e dinamici, tipici dei linguaggi ad oggetti.

NEGAZIONE COSTRUTTIVA

Parallelamente, sempre durante il periodo di dottorato si è interessata alla semantica della negazione in programmazione logica. In [26] è definita una versione compilativa della

negazione costruttiva (la negazione intensionale) ed è provata la sua correttezza e completezza (non-ground) rispetto al completamento a tre valori. Viene mostrato che la negazione intensionale è essenzialmente equivalente alla negazione costruttiva ed è più efficiente, come ci si potrebbe aspettare dal fatto che essa è una tecnica compilativa, dove la trasformazione ed il processo di normalizzazione associato sono eseguiti una volta per tutte sul programma sorgente.

In [27] viene definita una semantica bottom-up per la negazione costruttiva che è provata corretta e completa rispetto al completamento a tre valori del programma. La semantica descrive sia le risposte che le computazioni indefinite per le interrogazioni positive e negative. La sua costruzione segue l'idea base della negazione costruttiva, dove le risposte ad interrogazioni negative sono ottenute negando una "frontiera" dell'albero di computazione per le corrispondenti interrogazioni positive. La semantica proposta può essere considerata una base per ragionare sulle semantiche operazionali per la negazione costruttiva definite in letteratura.

OSSERVABILI ED INTERPRETAZIONE ASTRATTA

La parte principale dell'attività di ricerca, svolta durante la scuola di dottorato è stata rivolta allo studio di un formalismo semantico utilizzabile per ragionare sulle proprietà composizionali di astrazioni delle SLD-derivazioni. In [5] viene introdotta una semantica per i programmi logici definiti, espressa in termini di SLD-derivazioni e vengono studiate varie proprietà delle SLD-derivazioni, usando tale semantica. In [8, 28] viene definita una formalizzazione delle astrazioni delle SLD-derivazioni. Grazie alla relazione fra le proprietà degli operatori primitivi, definiti in [5] e le proprietà della semantica, si può definire una tassonomia di osservabili (astrazioni). Gli osservabili sono rappresentati da inserzioni di Galois fra le SLD-derivazioni ed un dominio astratto. Ogni classe della tassonomia è caratterizzata da un insieme di proprietà, che correlano gli operatori semantici primitivi e l'inserzione di Galois che definisce l'osservabile. Per ogni classe abbiamo una metodologia per derivare automaticamente la "migliore" semantica astratta (sistema di transizione, semantica denotazionale o entrambi) e la validità per la semantica astratta di alcuni teoremi che valgono per le SLD-derivazioni (AND-composizionalità, correttezza e full abstraction, OR-composizionalità delle denotazioni, costruzioni top-down e bottom-up equivalenti).

Lo schema definito in [8] ha come caso particolare tutte le semantiche considerate in letteratura, ma è anche un importante strumento concettuale per la definizione di domini astratti, che possono essere usati per l'analisi di varie proprietà di programmi. Inoltre un contributo importante del lavoro è la chiarificazione della struttura algebrica delle prove, che permette di analizzare in modo fine proprietà dipendenti dalla struttura delle dimostrazioni. A tale scopo si rivela particolarmente utile un approccio di sapore computazionale alla semantica denotazionale, secondo quanto suggerito da Dana Scott.

DIAGNOSI DI PROGRAMMI

Lo studio di nuove semantiche, affrontato durante la tesi di dottorato, mediante l'utilizzo di tecniche di interpretazione astratta, ha avuto come motivazione principale la realizzazione di strumenti di analisi, verifica ed ottimizzazione dei programmi logici. In [6, 29] si mostra come le tecniche di diagnosi dichiarativa possono essere estese per trattare la verifica di proprietà operazionali, come le risposte calcolate, e di proprietà astratte, come i tipi e le dipendenze ground. L'estensione è ottenuta usando una semplificazione del formalismo semantico dato in [8]. La tecnica risultante (la diagnosi astratta) porta ad eleganti metodi di verifica bottom-up e top-down, che non richiedono di determinare i sintomi a priori e che sono effettivi nel caso di proprietà descritte da domini finiti.

LINGUAGGI CONCORRENTI

Successivamente allo studio dei linguaggi logici si è interessata alla semantica della concorrenza con particolare attenzione ai linguaggi concorrenti con vincoli (*ccp*) ed ai linguaggi di coordinazione (*Linda*).

In [7, 30] sono definite due semantiche per i linguaggi concorrenti con vincoli e per un'estensione del formalismo ccp, chiamata *tccp*, che permette applicazioni real-time. In particolare viene definita una semantica operazionale del tccp e una semplice semantica denotazionale che risulta essere fully abstract rispetto la nozione standard di osservabile (la relazione di input-output associata ad un programma). La semantica denotazionale proposta è la base per la definizione in [11, 34, 36] di una logica temporale per ragionare sulla correttezza di programmi tccp. La logica è basata su modalità che permettono di specificare cosa un processo produce in dipendenza a ciò che l'ambiente gli fornisce come input. Queste modalità forniscono uno stile di specifica di tipo *azione/reazione*, che permette un'assiomatizzazione composizionale corretta e completa del comportamento reattivo di programmi tccp.

In [30, 32] viene poi studiato il potere espressivo di vari linguaggi concorrenti con vincoli con e senza estensioni temporali.

Il lavoro in [7, 30] viene esteso in [17, 40] per lavorare con vincoli soft. I vincoli soft estendono i vincoli classici fornendo un modo per esprimere preferenze, incertezza, conoscenza parziale, etc. Il linguaggio introdotto, il *Timed Soft Concurrent Constraint* (*tscc*), unitamente alla sua semantica, permette di definire un formalismo in cui è possibile modellare e risolvere problemi correlati alla "Qualità dei Servizi".

In [10, 33] viene introdotto un linguaggio Linda-like (*T-Linda*) che è ottenuto per mezzo di un'interpretazione temporizzata dei costrutti usuali del linguaggio Linda e mediante l'aggiunta di un semplice costrutto che permette di specificare vincoli di tempo. In [10, 35] viene poi definita la semantica operazionale di T-Linda per mezzo di un sistema di transizione ed una semantica denotazionale che è basata su *sequenze reattive temporizzate*. La correttezza di questo modello è provata rispetto ad una nozione di osservabile che include le tracce finite di azioni e le coppie di input/output.

VERIFICA COMPOSIZIONALE DI SISTEMI A STATI INFINITI

La verifica di sistemi a stati infiniti è una delle aree più interessanti e problematiche nel campo della verifica automatica di sistemi. Gli approcci più frequentemente utilizzati consistono nell'estensione al caso infinito di tecniche già utilizzate per i sistemi a stati finiti. In questo contesto è particolarmente interessante la proprietà di composizionalità: infatti una metodologia composizionale permette di ridurre la complessità della procedura di verifica suddividendo un grande sistema in pezzi più piccoli, ognuno dei quali verificabile separatamente. In [37] sono illustrate varie tecniche di verifica basate sui vincoli e sulla riscrittura di multinsiemi (MSR) per la verifica automatica di sistemi e di protocolli, parametriche sotto diversi aspetti. Sono poi discusse alcune ricerche che hanno come obiettivo lo sviluppo di un modello composizionale per ragionare su specifiche MSR.

SEMANTICHE COMPOSIZIONALI E POTERE ESPRESSIVO DI LINGUAGGI PER LA RISOLUZIONE DI VINCOLI

Il CHR (Constraint Handling Rules) è un linguaggio logico dichiarativo disegnato per la scrittura di risolutori di vincoli. Un programma CHR consiste di regole guardate a testa multipla, che permettono di riscrivere vincoli in vincoli più semplici fino a quando non si arriva ad una forma risolta.

Il linguaggio CHR ha ricevuto molta attenzione sia dal punto di vista pratico che teorico. Nonostante ciò, a causa delle clausole a testa multipla, esistono diversi aspetti della semantica del CHR che non sono stati ancora chiariti in modo formale. In [12, 38] viene introdotta una semantica di punto fisso che caratterizza il comportamento di input/output di un programma CHR e che è AND-composizionale, cioè che permette di costruire la semantica di una interrogazione composta a partire dalla semantica delle sue componenti.

La semantica di punto fissa definita in [12, 38] usa come la semantica originale (o standard) del CHR come semantica operazionale di riferimento. In accordo a tale semantica standard, le regole di propagazione del CHR possono introdurre computazioni infinite banali. Infatti poiché tali regole non rimuovono alcun vincolo dal goal corrente (ovvero dalla congiunzione dei vincoli da valutare), una volta che una regola di propagazione può essere applicata, essa può essere applicata un numero infinito di volte

In [39] viene quindi esteso il lavoro in [12, 38] considerando come semantica operazionale di riferimento una semantica più raffinata che permette di evitare le computazioni infinite introdotte dalle regole di propagazione utilizzando un cosiddetto *token store*.

La nuova semantica composizionale definita in [41] è provata corretta rispetto ad una nozione di osservabile diversa da quella utilizzata in [12, 38], poiché ora viene considerato anche il token store.

In [42, 44] viene poi studiato il potere espressivo del CHR con e senza la presenza di regole con testa multipla.

SISTEMI DI TRASFORMAZIONE DI PROGRAMMI

La trasformazione di programmi è stata inizialmente sviluppata come una tecnica per assistere i programmatori nello scrivere programmi efficienti e corretti. Tale tecnica consiste di vari passi di trasformazione. Ogni programma trasformato è equivalente (dà gli stessi risultati) di quello iniziale, una volta fissato l'input.

In [9, 31] viene introdotto un sistema di trasformazione per ottimizzare programmi concorrenti con vincoli (ccp). Sono definite opportune condizioni di applicabilità per le trasformazioni che garantiscono che la semantica di input-output dei programmi è preservata. In aggiunta ai benefici tradizionali che si possono osservare per i linguaggi sequenziali, la trasformazione di programmi concorrenti può anche portare all'eliminazione di canali di comunicazione e di punti di sincronizzazione, alla trasformazione di computazioni non deterministiche in computazioni deterministiche ed al risparmio di spazio computazionale. Inoltre poiché il sistema di trasformazione definito preserva il comportamento di deadlock dei programmi, esso può essere usato per provare l'assenza di deadlock di un dato programma rispetto ad una classe di interrogazioni.

In [16,39] viene invece definita una regola di trasformazione (l'unfolding) per programmi CHR e viene mostrato che essa preserva la semantica dei programmi CHR, in termini di qualified answers, una nozione già definita in letteratura. Viene anche definita una condizione sintattica che permette di rimpiazzare una regola in un programma con la sua versione unfoldata, preservando le qualified answers. Anche se l'idea dell'unfolding è semplice, il suo sviluppo tecnico è complicato dalla presenza nei programmi CHR delle guardie, delle teste multiple e delle sostituzioni matching. In particolare, non è immediato identificare le condizioni che permettono di rimpiazzare la regola originale con la sua versione unfoldata.

COMPITI ORGANIZZATIVI

Attuali:

- Da Novembre 2020: Presidente del Corso di Laurea “Economia e Informatica per l’Impresa” della Facoltà di Economia e del Dipartimento di Economia dell’Università di Chieti-Pescara.
- Da Novembre 2020: Membro della Commissione didattica del Dipartimento di Economia dell’Università di Chieti-Pescara.
- Da Novembre 2020: Membro della Scuola delle Scienze Economiche, Aziendali, Giuridiche e Sociologiche dell’Università di Chieti-Pescara.

Passati:

- Giugno 2007 – Novembre 2010: Presidente del Corso di Laurea “Economia Informatica Specialistica” della Facoltà di Economia dell’Università di Chieti-Pescara.
- Novembre 2010 – Novembre 2017: Presidente del Corso di Laurea “Economia e Informatica per l’Impresa” della Facoltà di Economia e del Dipartimento di Economia dell’Università di Chieti-Pescara.
- Giugno 2014 – Novembre 2017: Membro della Giunta del Dipartimento di Economia dell’Università di Chieti-Pescara.
- Giugno 2014 – Novembre 2017: Membro della Commissione Ricerca del Dipartimento di Economia dell’Università di Chieti-Pescara.
- Giugno 2014 – Novembre 2017: Membro della Commissione didattica del Dipartimento di Economia dell’Università di Chieti-Pescara.
- Novembre 2018 – Novembre 2020: Componente del Senato Accademico dell’Università di Chieti-Pescara.
- Novembre 2017 – Novembre 2020: Direttore del Dipartimento di Economia dell’Università di Chieti-Pescara.
- Membro del Collegio dei Docenti del Dottorato di Ricerca in Scienze del Dipartimento di Scienze (Università di Chieti Pescara).
- Membro del Collegio dei Docenti del Dottorato di Ricerca in Sistemi terrestri ed Ambienti costruiti. (Università di Chieti Pescara).

Tutor degli studenti di Dottorato Dott.ssa Pasqualina Potena (XX ciclo), Dott. Giacomo Di Tollo (XXI ciclo) e Dott.ssa Paola Campli (XXIV ciclo) del Dottorato di Ricerca in Scienze del Dipartimento di Scienze (Università di Chieti Pescara).

Attività di referee

Referee per varie riviste e conferenze internazionali fra cui Theoretical Computer Science, Journal of Logic and Computation, Information and Computation,, Journal of logic programming; Theory and Practice of Logic Programming; ACM SIGPLAN Conference on Principles and Practice of Declarative Programming, ICALP, AMAST, International Conference on Logic Programming, ILPS , Algebraic and Logic Programming, International Static Analysis Symposium; International ACM Conference on Principles and Practice of Declarative Programming; International Symposium on Programming Languages, Implementations, Logics and Programs; Joint Conference on Declarative Programming.

Partecipazione a scuole

- Advanced school on Foundations of Logic Programming, Alghero, Italia, 10-14 Settembre 1990, organizzato da ALP (the Association for Logic Programming) e GULP (Italian Association for Logic Programming).
- Fourth International School for Computer Science, Acireale, Italia, 22 giugno – 4 luglio 1992.
- International Summer School in Logic for Computer Science, Proof Theory and Foundatons of Programming. Università di Chambéry, Francia, 28 giugno – 9 luglio 1993.

ATTIVITÀ DIDATTICA

ULTIMI 5 ANNI

AA 2015-2016

CORSO DI LAUREA IN ECONOMIA E INFORMATICA PER L'IMPRESA, PESCARA

- Programmazione II Modulo 1 (4 CFU)
- Basi di Dati Aziendali (9 CFU)

CORSO DI LAUREA IN IGIENE DENTALE, CHIETI

- Informatica (2 CFU)

CORSO DI LAUREA IN TECNICHE DI LABORATORIO BIOMEDICO, CHIETI

- Sistemi di Elaborazione delle informazioni (2 CFU)

AA 2016-2017

CORSO DI LAUREA IN ECONOMIA E INFORMATICA PER L'IMPRESA, PESCARA

- Programmazione II Modulo 1 (4 CFU)
- Basi di Dati Aziendali (9 CFU)

CORSO DI LAUREA IN IGIENE DENTALE, CHIETI

- Informatica (2 CFU)

CORSO DI LAUREA IN TECNICHE DI LABORATORIO BIOMEDICO, CHIETI

- Sistemi di Elaborazione delle informazioni (2 CFU)

AA 2017-2018

CORSO DI LAUREA IN ECONOMIA E INFORMATICA PER L'IMPRESA, PESCARA

- Programmazione (6 CFU)
- Basi di Dati Aziendali (9 CFU)
- Laboratorio (2 CFU)

AA 2018/2019

CORSO DI LAUREA IN ECONOMIA E INFORMATICA PER L'IMPRESA, PESCARA

- Programmazione e Algoritmi 1 (6 CFU)
- Basi di Dati (3 CFU)
- Laboratorio di Basi di Dati (3 CFU)

AA 2019/2020

CORSO DI LAUREA IN ECONOMIA E INFORMATICA PER L'IMPRESA, PESCARA

- Programmazione e Algoritmi 1 (6 CFU)
- Basi di Dati (3 CFU)
- Laboratorio di Basi di Dati (3 CFU)

CORSO DI LAUREA IN INGEGNERIA BIOMEDICA, PESCARA

- Informatica- mutuo su Programmazione e Algoritmi 1 (6 CFU)

Pescara, 8 novembre 2020 Il dichiarante