PROGETTO DI UNA UNITÀ DI RICERCA - MODELLO B
Anno 2004 - prot. 2004013015_002

PARTE I

1.1 Tipologia del programma di ricerca

Interuniversitario

Aree scientifico disciplinari

Area 01: Scienze matematiche e informatiche (100%)

1.2 Durata del Programma di Ricerca

24 Mesi

1.3 Coordinatore Scientifico del Programma di Ricerca

GIACOBAZZI ***ROBERTO*** ***roberto.giacobazzi@univr.it***

INF/01 - Informatica

Università degli Studi di VERONA

Facoltà di SCIENZE MATEMATICHE FISICHE e NATURALI

Dipartimento di INFORMATICA

1.4 Responsabile Scientifico dell'Unità di Ricerca

BAGNARA ***ROBERTO***

Professore Associato ***07/12/1963*** ***BGNRRT63T07D458Z***

INF/01 - Informatica

Università degli Studi di PARMA

Facoltà di SCIENZE MATEMATICHE FISICHE e NATURALI

Dipartimento di MATEMATICA

0521/032317	***0521/032350***	***bagnara@cs.unipr.it***
(Prefisso e telefono)	*(Numero fax)*	*(Email)*

1.5 Curriculum scientifico del Responsabile Scientifico dell'Unità di Ricerca

Testo italiano
Roberto Bagnara (laurea in Scienze dell'Informazione, Università di Pisa, 1992; Dottorato di Ricerca in Informatica, Università di Pisa, 1997) è stato Ricercatore in Informatica presso l'Università degli Studi di Parma dal 1997 al 2001, ed è ora Professore associato di Informatica presso la stessa Università. È attivo da oltre un decennio nei settori dell'analisi di programmi, dell'interpretazione astratta e, più in generale, delle tecniche di manipolazione dei programmi basate sulla semantica. È coautore di oltre 25 pubblicazioni su riviste e atti di conferenze internazionali. Ha partecipato a diversi progetti di ricerca nazionali ed internazionali: ESPRIT Basic Research Action n. 6707 ("ParForce"); COFIN'99 "Certificazione automatica di programmi mediante interpretazione astratta"; COFIN'00 "Interpretazione astratta, sistemi di tipo e analisi Control-Flow"; COFIN'01 "Ragionamento su aggregati e numeri a supporto della programmazione e relative verifiche"; COFIN'02 "Verifica di sistemi reattivi basata su vincoli". Per quest'ultimo progetto, e` stato responsabile dell'unità di Parma. Nell'ambito delle "Azioni Integrate Italia-Spagna", è stato il coordinatore italiano del progetto "Ambienti avanzati per lo sviluppo di programmi logici" (IT229, 2001-2002). È l'autore principale di "China", un analizzatore statico altamente innovativo per programmi logici con vincoli (vedi http://www.cs.unipr.it/China/). Guida il gruppo che ha progettato e implementato la "Parma Polyhedra Library" (PPL): la PPL è una libreria C++ moderna, robusta ed efficiente per la manipolazione di poliedri convessi (vedi http://www.cs.unipr.it/ppl/). Guida anche il progetto interdisciplinare "PURRS" (Parma University's Recurrence Relation Solver) in corso presso il Dipartimento di Matematica dell'Università di Parma. L'obiettivo del progetto è quello di avanzare lo stato dell'arte nel campo dell'analisi e della verifica automatica di complessità (vedi http://www.cs.unipr.it/purrs/).

Testo inglese
Roberto Bagnara (laurea degree in Computer Science, University of Pisa, 1992; PhD in Computer Science, University of Pisa, 1997) has been an Assistant Professor of Computer Science at the University of Parma from 1997 to 2001 and is now Associate Professor of Computer Science at the same university. For more that 10 years he has been active in the fields of program analysis, abstract interpretation and, more generally, semantics-based program manipulation. He has coauthored more than 25 papers that have been published in journals and international conference proceedings. He has participated in several international and national research projects: ESPRIT Basic Research Action Project n. 6707 ("ParForce"); COFIN'99 MURST project "Automatic Program Certification by Abstract Interpretation"; COFIN'00 MURST project "Abstract Interpretation, Type Systems and Control-Flow Analysis"; COFIN'01 MURST project "Aggregate- and Number-Reasoning for Computing"; COFIN'02 MURST project "Constraint Based Verification of Reactive Systems." For this last project, he was responsible for the Parma research unit. He was the Italian coordinator for the Italy-Spain bilateral project "Advanced Development Environment for Logic Programs" (IT229, 2001-2002). He is the main author of "China", a state-of-the-art data-flow analyzer for constraint logic programs (see http://www.cs.unipr.it/China/). He is leading the Parma Polyhedra Library (PPL) project: the PPL is a modern, robust and efficient C++ library for the manipulation of convex polyhedra (see http://www.cs.unipr.it/ppl/ for more details). He is also leading the interdisciplinary project PURRS (Parma University's Recurrence Relation Solver) at the Department of Mathematics of the University of Parma. The aim of this project is to advance the state of the art in automatic complexity analysis and verification (see http://www.cs.unipr.it/purrs/ for more details).

1.6 Pubblicazioni scientifiche più significative del Responsabile Scientifico dell'Unità di Ricerca

1. *BAGNARA R. (1998).* ***A Hierarchy of Constraint Systems for Data-Flow Analysis of Constraint Logic-Based Languages*** *SCIENCE OF COMPUTER PROGRAMMING. (vol. 30 pp. 119-155) ISSN: 0167-6423*
2. *BAGNARA R.; HILL P. M.; ZAFFANELLA E. (2002).* ***Set-Sharing Is Redundant for Pair-Sharing*** *THEORETICAL COMPUTER SCIENCE. (vol. 1-2 pp. 3-46) ISSN: 0304-3975*
3. *BAGNARA R.; RICCI E.; ZAFFANELLA E.; HILL P. M. (2002).* ***Possibly Not Closed Convex Polyhedra and the Parma Polyhedra Library*** *9th International Symposium on Static Analysis (Madrid, Spagna). vol. 2477 pp. 213-229 Lecture Notes in Computer Science, Springer-Verlag, Berlin..*
4. *BAGNARA R.; HILL P. M.; RICCI E.; ZAFFANELLA E. (2003).* ***Precise Widening Operators for Convex Polyhedra*** *10th International Symposium on Static Analysis (San Diego, California, USA). vol. 2694 pp. 337-354 Lecture Notes in Computer Science, Springer-Verlag, Berlin..*
5. *BAGNARA R.; HILL P. M.; ZAFFANELLA E. (2003).* ***Widening Operators for Powerset Domains*** *5th Int'l Conf. on Verification, Model Checking and Abstract Interpretation. vol. 2937 pp. 135-148 Lecture Notes in Computer Science, Springer-Verlag, Berlin.*

1.7 Risorse umane impegnabili nel Programma dell'Unità di Ricerca

1.7.1 Personale universitario dell'Università sede dell'Unità di Ricerca

Personale docente

nº	Cognome	Nome	Dipartimento	Qualifica	Settore Disc.	Mesi Uomo 1° anno	Mesi Uomo 2° anno
1.	*BAGNARA*	*Roberto*	*Dip. MATEMATICA*	*Prof. Associato*	*INF/01*	*10*	*6*
2.	*RICCI*	*Gabriele*	*Dip. MATEMATICA*	*Prof. Associato*	*INF/01*	*4*	*2*
3.	*ROSSI*	*Gianfranco*	*Dip. MATEMATICA*	*Prof. Ordinario*	*INF/01*	*5*	*3*
4.	*ZAFFANELLA*	*Enea*	*Dip. MATEMATICA*	*Ricercatore Universitario*	*INF/01*	*11*	*5*
	TOTALE					**30**	**16**

Altro personale

Nessuno

1.7.2 Personale universitario di altre Università

Personale docente

Nessuno

Altro personale

Nessuno

1.7.3 Titolari di assegni di ricerca

nº	Cognome	Nome	Dipartimento	Data di inizio del contratto	Durata(in anni)	Mesi Uomo 1° anno	Mesi Uomo 2° anno
1.	*PANEGAI*	*Elio*	*Dip. MATEMATICA*	*01/12/2003*	*2*	*5*	*3*
TOTALE						**5**	**3**

1.7.4 Titolari di borse

Nessuno

1.7.5 Personale a contratto da destinare a questo specifico programma

Qualifica	Costo previsto	Mesi Uomo 1° anno	Mesi Uomo 2° anno	Note
Assegnista	*19.000*	*8*	*3*	
Altre tipologie	*10.000*		*8*	*programmatore*
TOTALE	**29.000**	**8**	**11**	

1.7.6 Personale extrauniversitario indipendente o dipendente da altri Enti

Nessuno

PARTE II

2.1 Titolo specifico del programma svolto dall'Unità di Ricerca

Testo italiano
Domini astratti e calcolo astratto per l'analisi di programmi

Testo inglese
Abstract domains and abstract computations for program analysis

---

2.2 Settori scientifico-disciplinari interessati dal Programma di Ricerca

INF/01 - Informatica

2.3 Parole chiave

Testo italiano
INTERPRETAZIONE ASTRATTA ; DOMINI ASTRATTI ; WIDENING ; DOMINI NUMERICI ; ANALISI STATICA ; ANALISI DI COMPLESSITÀ

Testo inglese
ABSTRACT INTERPRETATION ; ABSTRACT DOMAINS ; WIDENING ; NUMERICAL DOMAINS ; STATIC ANALYSIS ; COMPLEXITY ANALYSIS

2.4 Base di partenza scientifica nazionale o internazionale

Testo italiano
Assicurare la sicurezza, l'affidabilità e l'attendibilità dei sistemi hardware e software è una delle più grandi sfide che la società moderna si trova ad affrontare. Sebbene sia inevitabile fare affidamento su sistemi informatici di complessità sempre crescente, allo stesso tempo dobbiamo constatare che i metodi tradizionali di validazione (come il testing), nonostante il loro costo elevato, non riescono a fornire adeguate garanzie sul reale funzionamento di detti sistemi. Un esempio recentissimo è dato dall'accertamento, nel febbraio 2004, che ``un malfunzionamento software di un diffuso sistema di gestione dell'energia elettrica distribuito dalla General Electric ha contribuito all'impatto devastante del black-out che ha colpito il nord-est degli Stati Uniti lo scorso 14 agosto, lasciando al buio 50 milioni di persone in otto stati degli USA e nel Canada'' (si veda http://www.securityfocus.com/news/8016).

Da questa situazione consegue naturalmente la crescente importanza delle procedure di validazione per l'hardware ed il software: per le cosiddette applicazioni ``mission-critical'', è essenziale garantire che i sistemi informatici siano esenti da malfunzionamenti e non producano risultati errati, i quali potrebbero avere conseguenze catastrofiche. Gli ultimi 25 anni hanno visto lo sviluppo di tecniche di analisi e manipolazione dei programmi basate sulla semantica in grado di offrire, da un lato, il rigore metodologico mancante negli approcci basati su testing e simulazione, dall'altro lato, un livello di scalabilità irraggiungibile sia per i metodi basati su analisi esaustive di modelli concreti, sia per gli strumenti di verifica basati su dimostratori automatici di teoremi. La teoria dell'Interpretazione Astratta [CC77, CC79] è alla base dell'applicazione rigorosa di tecniche semantiche quali l'analisi statica ed il model checking astratto, che condividono l'approccio basato sulla derivazione di informazioni corrette, ancorché approssimate, circa il comportamento di un sistema informatico mediante l'astrazione della sua descrizione, sia essa una specifica formale, un modello o un programma. Questo progetto si focalizza in particolare sullo studio ed implementazione dei domini e delle tecniche di calcolo astratti (come l'iterazione caotica di punto fisso con operatori di widening/narrowing [CC92]), con particolare riferimento all'analisi statica dei programmi.

Lo sviluppo di un analizzatore statico per un linguaggio di programmazione reale, o anche per un suo sottoinsieme specializzato (ad esempio, il codice C generato dalla compilazione dei linguaggi sincroni o di programmi Javacard), è un compito oltremodo complicato. Fortunatamente, questo compito può essere suddiviso in sotto-analisi distinte corrispondenti ai diversi aspetti delle proprietà di interesse, che possono essere implementate individualmente come domini astratti. Un dominio astratto è quindi composto da un insieme di rappresentazioni effettive per le proprietà logiche di interesse, dalle operazioni atte ad estrarre tali proprietà dalle componenti del programma sotto esame, dalle primitive per la propagazione (in avanti o all'indietro) di queste proprietà all'interno della struttura del programma, nonché altre primitive per accelerare la convergenza del calcolo di punto fisso. Quando più domini astratti sono disponibili, essi possono essere combinati, per esempio mediante congiunzione delle proprietà logiche rappresentate. In tal caso è necessario predisporre un opportuno processo di riduzione che consenta la propagazione dell'informazione tra le varie componenti, al fine di consentire eventuali semplificazioni e riduzioni atte ad aumentarne la precisione. Sono stati studiati ed implementati diversi tipi di composizione di domini astratti. Questo schema, formalizzato dall'interpretazione astratta, conduce in modo naturale all'idea di una libreria generica di domini astratti utilizzabile in svariati

analizzatori statici.

Le esperienze fatte durante lo sviluppo della componente per l'analisi di sharing dell'analizzatore statico China [Bag97, BHZ02, BZH05, HBZ02, HZB04, ZBH99, ZHB02] e durante lo sviluppo, tuttora in corso, della ``Parma Polyhedra Library'' [BHRZ03, BHZ03a, BHZ03b, BRZH02] hanno dimostrato che la creazione di un dominio astratto funzionalmente completo e professionale rappresenta, sia dal punto di vista teorico che dal punto di vista dell'implementazione, un obiettivo ambizioso che può coinvolgere una pluralità di persone per più anni.

Molte proprietà dei programmi (come le proprietà invarianti e l'assenza di errori a run-time) riguardano insiemi di stati o relazioni tra insiemi di stati (come le nozioni di correttezza totale e parziale). Le astrazioni di insiemi infiniti di stati rappresentati da proprietà numeriche (espresse, ad esempio, da insiemi di vettori su domini numerici) o proprietà simboliche (espresse, ad esempio, da insiemi di nomi) sono quindi da considerarsi di primaria importanza e, per esse, molti progressi si possono e devono ancora ottenere. Il lavoro richiesto investe sia aspetti teorici che aspetti implementativi.

Un esempio elementare di dominio astratto è dato dal dominio degli intervalli: congiunzioni di vincoli della forma `ax <= b' o `ax < b', dove `a' e `b' sono costanti numeriche ed `x' rappresenta una quantità relativa alla computazione sotto esame (ad esempio, il valore di una variabile numerica del programma oppure la dimensione di un buffer allocato dinamicamente). Nonostante il fatto che gli intervalli sono stati implementati e reimplementati dozzine di volte, nessuna delle implementazioni esistenti risulta essere adeguata per gli scopi dell'interpretazione astratta (si veda più sotto).

Gli ottagoni [Min01] offrono un semplice esempio di dominio relazionale (congiunzioni di vincoli della forma `ax + by <= c', dove `a' e `b' possono assumere i valori -1, 0 o +1 e `c' è una costante numerica, mentre `x' e `y' sono quantità della computazione sotto esame). In questo caso gli algoritmi sono sensibilmente più complicati, in particolare quando occorre tenere in considerazione gli eventuali errori di arrotondamento commessi nei calcoli in virgola mobile, sia nella semantica concreta che nel dominio astratto.

Un dominio astratto più accurato è quello dei poliedri convessi [CH78]. Un elemento di questo dominio codifica una proprietà descrivibile mediante un insieme finito di equazioni e disequazioni lineari. Anche quando si considerano proprietà più complesse, la cui descrizione precisa richiederebbe l'utilizzo di vincoli non lineari, l'approssimazione data dai poliedri convessi fornisce spesso risultati di precisione accettabile. Gli strumenti software industriali che intendono utilizzare appieno il dominio astratto dei poliedri convessi devono necessariamente fronteggiare un insieme di problemi. Innanzitutto, le implementazioni attuali devono raggiungere un livello di scalabilità che consenta loro di gestire centinaia di vincoli definiti su centinaia di variabili con un consumo ragionevole delle risorse tempo e spazio di memoria. Tale requisito implica la definizione di implementazioni robuste degli operatori sui poliedri convessi, caratterizzate da un tempo massimo di esecuzione garantito (possibilmente al costo di una perdita controllata di precisione) e dalla disponibilità di una ampia scelta di compromessi tra la velocità e la precisione del calcolo di punto fisso. Tutto ciò ha a che fare principalmente con lo sviluppo, il più sistematico possibile, di tecniche per l'accelerazione del calcolo di punto fisso basate su operatori di widening accurati, che dovranno essere in grado di catturare i pattern di crescita più comuni per le varie applicazioni dei poliedri convessi per l'analisi e la verifica di sistemi complessi.

L'analisi di complessità mira a determinare limitazioni superiori e inferiori alla complessità di algoritmi, processi, stutture dati e programmi. I risultati di questo tipo di analisi, che può essere parzialmente o totalmente automatizzata, possono essere usati, ad esempio, per valutare se debba essere o meno consentita l'esecuzione di agenti mobili in un determinato contesto, per assistere i programmatori nel ragionamento sul comportamento dei programmi, per guidare l'applicazione di trasformazioni di programmi volte all'ottimizzazione, e per scoprire possibili inefficienze dovute ad errori di programmazione, altrimenti molto difficili da individuare. L'analisi automatica di complessità, pur essendo di grande interesse per una serie di tecniche emergenti e per applicazioni quali gli agenti mobili, il data mining ed il commercio elettronico, è tuttora un campo relativamente inesplorato. Inoltre, tranne alcune rilevanti eccezioni quali [DL93, DLGHL97], non si hanno a disposizione strumenti che possano fornire limitazioni di complessità accurate (superiori ed inferiori) per sottoinsiemi significativi dei linguaggi di programmazione comunemenți utilizzati.

Testo inglese

Guaranteeing the reliability, trustworthiness, safety and security of software and hardware systems is a major challenge for modern societies. We rely on systems that are increasingly complex and, at the same time, we experience how spectacularly traditional validation methods, such as testing, fail to provide the required guarantees despite their very high cost. A very recent example is constituted by the recognition, in February 2004, that ``a previously-unknown software flaw in a widely-deployed General Electric energy management system contributed to the devastating scope of the August 14th northeastern U.S. blackout [which] eventually cut off electricity to 50 million people in eight states and Canada'' (see http://www.securityfocus.com/news/8016).

The obvious consequence of this state of affairs is the increasing importance of software and hardware validation: for mission-critical applications, it is crucial to ensure that the computer systems cannot have erroneous behavior or produce wrong results which could have catastrophic consequences. The last 25 years have seen the progressive development of semantics-based program analysis and manipulation techniques that offer a rigor not provided by testing and simulation, and a scalability that is out of reach of methods based on exhaustive checking of concrete models or verification using automated theorem proving. Abstract interpretation [CC77, CC79] provides the ideal foundation for semantics-based techniques such as static analysis and abstract model checking, since what they have in common is the determination of correct though approximate information about the behavior of a system by systematically abstracting the semantics of its specification, model or program. The present project focuses on the study and implementation of abstract domains and abstract computation techniques (such as chaotic fixpoint iteration with widening/narrowing [CC92]) especially targeted at static program analysis.

The development of a static analyzer for a real programming language, or even for a specialized subset (e.g., C code generated from synchronous languages or JavaCard), is a very difficult task. Fortunately, the task can be organized in separate sub-analyses corresponding to various aspects of the properties of interest, which can then be implemented individually as abstract domains. An abstract domain includes a computer representation of the logical properties of interest, operations for extracting this information from program components, primitives for propagating this information forward and/or backward within the program, as well as

others to accelerate the convergence of fixpoint computations and so forth. When several abstract domains are available, they can be composed, e.g., by conjunction of the logical properties that they encode. In this case a reduction process is necessary, to propagate information from one abstract domain to the other, mainly for simplification and precision enhancement. Many other possible compositions of abstract domains have been studied and can be applied. This scheme, formalized by abstract interpretation, naturally leads to the idea of a library of abstract domains that can be used in several static analyzers.

The experience gained with the development of the sharing analysis component of the China analyzer [Bag97, BHZ02, BZH05, HBZ02, HZB04, ZBH99, ZHB02] and with the ongoing development of the ``Parma Polyhedra Library'' [BHRZ03, BHZ03a, BHZ03b, BRZH02] has shown that the creation of a professional, full-fledged abstract domain is a very significant and challenging task (both from the theoretical and the implementation points of view), involving a number of people over several years.

Many program properties (such as invariance and absence of runtime errors) involve sets of states or relations between sets of states (such as partial or total correctness). Abstractions of infinite sets of states such as numerical properties (expressed, for example, by sets of vectors of numbers) or symbolic properties (expressed, for instance, as sets of names) are thus of primary importance and, for them, much progress remains to be done. The required work is both at the theoretical level and at the implementation level.

An elementary example of abstract domain is the domain of intervals: conjunctions of constraints of the form `ax <= b' or `ax < b' where `a' and `b' are numerical constants and `x' is a value involved in the program computation (for instance, the value of a numerical program variable or the length of a dynamically allocated buffer). Despite the fact that intervals have been implemented and reimplemented dozens of times, no existing implementation is really suitable for the purposes of abstract interpretation (see below).

Octagons [Min01] offer a simple example of relational domain (conjunctions of constraints of the form `ax + by <= c' where `a' and `b' are either -1, 0 or +1, `c' is a numerical constant, while `x' and `y' are values involved in the program computation). Here the algorithms are considerably more involved, particularly when rounding errors in floating-point computations have to be taken into account both in the concrete semantics and in the abstract domain.

A more sophisticated abstract domain is the one of convex polyhedra [CH78]. An element of this domain can encode any property that can be described by a finite system of linear equalities and inequalities. Even when considering more complex properties, whose precise description would require the handling of non-linear constraints, the approximation provided by convex polyhedra often yields results of acceptable precision. Industrial tools cannot fully make use of the convex polyhedra abstract domain without considering a number of problems. First of all, the present implementations must scale up to deal with many hundreds of constraints on many hundreds of variables with reasonable time and memory space resources. This means that there should be robust implementations of operators on polyhedra with guaranteed maximal execution times (maybe at the controlled expense of accuracy) and with a wide available range of compromises between the speed and precision of fixpoint computations. This concerns mainly the development, as systematic as possible, of precise widening operators for fixpoint convergence acceleration techniques. These operators will have to capture the growth patterns that are most common in the various applications of convex polyhedra in the analysis and verification of complex systems.

Complexity analysis aims at the derivation of upper and lower bounds to the complexity of algorithms, processes, data structures and programs. The results of such analyses, which may be partially or wholly automated, can be used, say, to decide whether mobile agents should be allowed to run in a given context, assist programmers in reasoning about the behavior of programs, guide applications of optimized program transformations, and discover efficiency bugs that are otherwise very difficult to detect. The field of fully automatic complexity analysis, which is of great interest for a range of emerging techniques and applications such as mobile agents, data mining and electronic commerce, is relatively unexplored. Moreover, with some relevant exceptions such as [DL93, DLGHL97], there is a lack of tools that can provide precise upper and lower complexity bounds for significant fragments of commonly used programming languages.

2.4.a Riferimenti bibliografici

[AK85] J. F. Allen and H. A. Kautz. A model of naive temporal Reasoning. Formal Theories of the Commonsense World, pages 251-268. Ablex, Norwood, NJ, 1985.

[Anc91] C. Ancourt. Génération Automatique de Codes de Transfert pour Multiprocesseurs à Mémoires Locales. PhD thesis, Université de Paris VI, 1991.

[Bag97] R. Bagnara. Data-Flow Analysis for Constraint Logic-Based Languages. PhD thesis, Dipartimento di Informatica, Università di Pisa, 1997.

[Bag98] R. Bagnara. A hierarchy of constraint systems for data-flow analysis of constraint logic-based languages. Science of Computer Programming, 30(1-2):119-155, 1998.

[BCC+03] B. Blanchet, P. Cousot, R. Cousot, J. Feret, L. Mauborgne, A. Miné, D. Monniaux, and X. Rival. A static analyzer for large safety-critical software. Proc. of the ACM SIGPLAN 2003 Conference on Programming Language Design and Implementation, pages 196-207, San Diego, CA, 2003. ACM Press.

[BGL92] R. Bagnara, R. Giacobazzi, and G. Levi. Static analysis of CLP programs over numeric domains. Actes ``Workshop on Static Analysis '92'', volume 81-82 of Bigre, pages 43-50, Bordeaux, 1992. Extended abstract.

[BGL93] R. Bagnara, R. Giacobazzi, and G. Levi. An application of constraint propagation to data-flow analysis. In Proc. of 9th Conference on Artificial Intelligence for Applications, pages 270-276, Orlando, FL, 1993. IEEE Computer Society Press.

[BHRZ03] R. Bagnara, P. M. Hill, E. Ricci, and E. Zaffanella. Precise widening operators for convex polyhedra. Static Analysis: Proc. of the 10th International Symposium, volume 2694 of LNCS, pages 337-354, San Diego, CA, 2003. Springer-Verlag.

[BHZ02] R. Bagnara, P. M. Hill, and E. Zaffanella. Set-sharing is redundant for pair-sharing. Theoretical Computer Science, 277(1-2):3-46, 2002.

[BHZ03a] R. Bagnara, P. M. Hill, and E. Zaffanella. A new encoding and implementation of not necessarily closed convex polyhedra. Proc. of the 3rd Workshop on Automated Verification of Critical Systems, pages 161-176, Southampton, UK, 2003.

[BHZ03b] R. Bagnara, P. M. Hill, and E. Zaffanella. Widening operators for powerset domains. Proc. of the Fifth International Conference on Verification, Model Checking and Abstract Interpretation (VMCAI 2004), volume 2937 of LNCS, pages 135-148, Venice, 2003. Springer-Verlag.

[BK97] F. Benoy and A. King. Inferring argument size relationships with CLP(R). Logic Program Synthesis and Transformation: Proc. of the 6th International Workshop, volume 1207 of LNCS, pages 204-223, Stockholm, 1997. Springer-Verlag.

[BRZH02] R. Bagnara, E. Ricci, E. Zaffanella, and P. M. Hill. Possibly not closed convex polyhedra and the Parma Polyhedra Library. Static Analysis: Proc. of the 9th International Symposium, volume 2477 of LNCS, pages 213-229, Madrid, 2002. Springer-Verlag.

[BZ04] R. Bagnara and A. Zaccagnini. Checking and bounding the solutions of some recurrence relations. Quaderno 344, Dipartimento di Matematica, Università di Parma, 2004. Available at http://www.cs.unipr.it/Publications/.

[BZH05] R. Bagnara, E. Zaffanella, and P. M. Hill. Enhanced sharing analysis techniques: A comprehensive evaluation. Theory and Practice of Logic Programming, 5(1&2), 2005. To appear.

[BZZ03] R. Bagnara, A. Zaccagnini, and T. Zolo. The automatic solution of recurrence relations. I. Linear recurrences of finite order with constant coefficients. Quaderno 334, Dipartimento di Matematica, Università di Parma, 2003. Available at http://www.cs.unipr.it/Publications/.

[CC77] P. Cousot and R. Cousot. Abstract interpretation: A unified lattice model for static analysis of programs by construction or approximation of fixpoints. Proc. of the Fourth Annual ACM Symposium on Principles of Programming Languages, pages 238-252, New York, 1977. ACM Press.

[CC79] P. Cousot and R. Cousot. Systematic design of program analysis frameworks. Proc. of the Sixth Annual ACM Symposium on Principles of Programming Languages, pages 269-282, New York, 1979. ACM Press.

[CC92] P. Cousot and R. Cousot. Comparing the Galois connection and widening/narrowing approaches to abstract interpretation. Proc. of the 4th International Symposium on Programming Language Implementation and Logic Programming, volume 631 of LNCS, pages 269-295, Leuven, 1992. Springer-Verlag.

[CH78] P. Cousot and N. Halbwachs. Automatic discovery of linear restraints among variables of a program. Conference Record of the Fifth Annual ACM Symposium on Principles of Programming Languages, pages 84-96, Tucson, AR, 1978. ACM Press.

[CL98] Ph. Clauss and V. Loechner. Parametric analysis of polyhedral iteration spaces, extended version. Journal of VLSI Signal Processing, 19(2):179-194, 1998.

[Dav87] E. Davis. Constraint propagation with interval labels. Artificial Intelligence, 32:281-331, 1987.

[DL93] S. Debray and N.-W. Lin. Cost analysis of logic programs. ACM Transactions on Programming Languages and Systems, 15(5):826-875, 1993.

[DLGHL97] S. K. Debray, P. López-García, M. V. Hermenegildo, and N.-W. Lin. Lower bound cost estimation for logic programs. Logic Programming: Proc. of the 1997 International Symposium, pages 291-305, Port Washington, NY, 1997. The MIT Press.

[DPPR00] A. Dovier, C. Piazza, E. Pontelli, and G. Rossi. Sets and constraint logic programming. ACM Transactions on Programming Languages and Systems, 22(5):861-931, 2000.

[DRS01] N. Dor, M. Rodeh, and S. Sagiv. Cleanness checking of string manipulations in C programs via integer analysis. Static Analysis: 8th International Symposium, SAS 2001, volume 2126 of LNCS, pages 194-212, Paris, 2001. Springer-Verlag.

[Gra91] P. Granger. Static analysis of linear congruence equalities among variables of a program. TAPSOFT'91: Proc. of the International Joint Conference on Theory and Practice of Software Development, volume 493 of LNCS, pages 169-192, Brighton, UK, 1991. Springer-Verlag.

[Gra97] P. Granger. Static analyses of congruence properties on rational numbers (extended abstract). Static Analysis: Proc. of the 4th International Symposium, volume 1302 of LNCS, pages 278-292, Paris, 1997. Springer-Verlag.

[HBZ02] P. M. Hill, R. Bagnara, and E. Zaffanella. Soundness, idempotence and commutativity of set-sharing. Theory and Practice of Logic Programming, 2(2):155-201, 2002.

[HMPV03] N. Halbwachs, D. Merchat, and C. Parent-Vigouroux. Cartesian factoring of polyhedra in linear relation analysis. Static Analysis: Proc. of the 10th International Symposium, volume 2694 of LNCS, pages 355-365, San Diego, CA, 2003. Springer-Verlag.

[HPR97] N. Halbwachs, Y.-E. Proy, and P. Roumanoff. Verification of real-time systems using linear relation analysis. Formal Methods in System Design, 11(2):157-185, 1997.

[HZB04] P. M. Hill, E. Zaffanella, and R. Bagnara. A correct, precise and efficient integration of set-sharing, freeness and linearity for the analysis of finite and rational tree languages. Theory and Practice of Logic Programming, 4(3):289-323, 2004.

[LTW+01] M. Lerch, G. Tischler, J. Wolff von Gudenberg, W. Hofschuster, and W. Krämer. The interval library filib++ 2.0 -- design, features and sample programs. Preprint 2001/4, Universität Wuppertal, Germany, 2001.

[LW97] V. Loechner and D. K. Wilde. Parameterized polyhedra and their vertices. International Journal of Parallel Programming, 25(6):525-549, 1997.

[Mas93] F. Masdupuy. Array Indices Relational Semantic Analysis Using Rational Cosets and Trapezoids. Thèse d'informatique, École Polytechnique, Palaiseau, France, 1993.

[Min01] A. Miné. The octagon abstract domain. In Proc. of the Eighth Working Conference on Reverse Engineering (WCRE'01), pages 310-319, Stuttgart, Germany, 2001. IEEE Computer Society Press.

[NR00] S. P. K. Nookala and T. Risset. A library for Z-polyhedral operations. Publication interne 1330, IRISA, Campus de Beaulieu, Rennes, France, 2000.

[Pug92] W. Pugh. A practical algorithm for exact array dependence analysis. Communications of the ACM, 35(8):102-114, 1992.

[QRR96] P. Quinton, S. Rajopadhye, and T. Risset. On manipulating Z-polyhedra. Technical Report 1016, IRISA, Campus Universitaire de Bealieu, Rennes, France, 1996.

[ZBH99] E. Zaffanella, R. Bagnara, and P. M. Hill. Widening Sharing. Principles and Practice of Declarative Programming, volume 1702 of LNCS, pages 414-431, Paris, 1999. Springer-Verlag.

[ZHB02] E. Zaffanella, P. M. Hill, and R. Bagnara. Decomposing non-redundant sharing by complementation. Theory and Practice of Logic Programming, 2(2):233-261, 2002.

2.5 Descrizione del programma e dei compiti dell'Unità di Ricerca

Testo italiano
La maggior parte del lavoro dell'unità di ricerca verrà condotta nel contesto del Workpackage 1 (Analisi Statica) del progetto e riguarderà la specifica, progettazione ed implementazione di domini ed operatori astratti, ivi inclusi quegli operatori, come i widening ed i narrowing, che possono essere utilizzati per forzare o accelerare la convergenza del calcolo astratto. In particolare, l'obiettivo principale di questa ricerca è lo studio teorico e l'implementazione effettiva di una libreria di domini astratti numerici funzionalmente completi, robusti e riutilizzabili: intervalli, differenze e divisioni vincolate, ottagoni, poliedri convessi, griglie e Z-poliedri.

Alcune operazioni sul dominio dei poliedri convessi sono caratterizzate da una complessità esponenziale nel caso pessimo, sia in tempo che in spazio. D'altra parte, il dominio degli intervalli, pur essendo molto più semplice da un punto di vista computazionale, spesso restituisce approssimazioni troppo grossolane. Una soluzione intermedia è data dai domini delle differenze vincolate e degli ottagoni. Gli utilizzatori industriali di strumenti di analisi statica hanno spesso bisogno della precisione fornita dai domini astratti più accurati; al tempo stesso, però, hanno bisogno di una qualche forma di controllo sul consumo delle risorse tempo e spazio di memoria, in quanto risultati approssimati ottenuti entro un dato intervallo di tempo sono preferibili rispetto a risultati più precisi, ma disponibile solo ad una data non meglio precisata. Perseguiremo pertanto l'obiettivo di una combinazione dinamica del dominio dei poliedri con questi domini di vincoli più semplici al fine di consentire un buon compromesso tra precisione e scalabilità. L'analisi statica o procedura di verifica, per default, dovrebbe considerare i domini più descrittivi (e costosi), come quello dei poliedri convessi, e specificare per ogni operazione una limitazione sulle risorse (tempo di calcolo, occupazione di memoria, valore massimo dei coefficienti numerici). Quando il calcolo di una operazione eccede i limiti fissati, si utilizza un algoritmo più semplice e meno accurato, ma che fornisce un risultato corretto ai fini di un'interpretazione astratta, ovvero tale che il poliedro risultante è un sovrainsieme del risultato esatto. L'algoritmo meno preciso può operare uno o più cambi di rappresentazione, per esempio approssimando un poliedro utilizzando un sistema di differenze vincolate o di intervalli e calcolando l'operazione su questi domini semplificati. Il risultato può quindi essere convertito all'indietro in un poliedro convesso. L'obiettivo finale è la realizzazione di una libreria robusta di operatori sui poliedri in grado di gestire molte centinaia (potenzialmente migliaia) di vincoli su spazi con un numero elevato di dimensioni. È importante notare che tale scenario è proponibile solo se tutti i domini astratti coinvolti sono progettati ed implementati in maniera da poter consistentemente reagire, senza perdita di risorse di sistema, ad eventi quali la scadenza di un timeout o un'eccezione lanciata dalle routine di allocazione della memoria. Il fatto di aver raggiunto questo obiettivo per la ``Parma Polyhedra Library'' [BRZH02], che sarà la componente chiave nella libreria di domini astratti numerici, è quindi un punto essenziale.

Altre caratteristiche importanti che devono essere possedute da tutte le componenti della libreria sono: il supporto per tutti gli operatori richiesti da un interprete astratto; l'interoperabilità, ovvero la possibilità di costruire una descrizione di un tipo approssimando una descrizione di un altro tipo; la definizione di operatori di decomposizione dello spazio vettoriale al fine di velocizzare l'esecuzione [HMPV03]; la disponibilità, per ogni operatore, di un database di test di non-regressione.

Intervalli

Un buon dominio astratto per gli intervalli è desiderabile non solo come dominio di fall-back, ovvero il dominio più efficiente (anche se non molto preciso) da utilizzare quando tutti gli altri tentativi sono falliti, ma anche come ingrediente chiave per l'implementazione di domini più complessi come le differenze ed le divisioni vincolate [Dav87, Bag97]. Sfortunatamente, nessuna delle librerie di intervalli esistenti è adeguata per costituire la base di un tale dominio astratto. Le librerie disponibili o mancano del supporto per gli intervalli non chiusi (così che non sono in grado di rappresentare vincoli della forma `ax < b'), oppure non forniscono il supporto adeguato per le approssimazioni nel senso della correttezza parziale (ad esempio, la divisione per un intervallo contenente lo zero genera un errore a tempo di esecuzione invece di restituire l'intervallo contenente il risultato sotto l'assunzione che la divisione concreta approssimata non fosse una divisione per zero), oppure ignorano gli errori di arrotondamento e sono pertanto scorrette. Per esempio, la libreria filib++ [LTW+01], che è molto sofisticata e ben considerata all'interno del suo campo di applicazione, soffre dei primi due problemi menzionati.

Verrà pertanto studiato ed implementato un dominio astratto completo basato sugli intervalli. Tali intervalli saranno parametrici rispetto ad alcune caratteristiche e supporteranno sia estremi chiusi che aperti. Sarà possibile scegliere gli estremi in una serie di domini numerici, inclusi gli interi nativi, gli interi a precisione illimitata, i razionali ed i numeri in virgola mobile nativi. Nel caso dei numeri in virgola mobile saranno resi disponibili vari metodi per controllare il tipo di arrotondamento, con diverse caratteristiche relative alla portabilità ed all'efficienza. Indipendentemente dal tipo degli estremi, saranno supportati sia intervalli di numeri reali che di interi.

Differenze e divisioni vincolate

Un dominio di differenze vincolate consente di esprimere la relazione tra due quantità per mezzo di insiemi finiti di vincoli della forma `x-y in S', dove `S' è un sottoinsieme dei reali [Dav87]. La prima proposta di utilizzo per l'analisi statica di questa ed altre descrizioni originate nel campo dell'intelligenza artificiale si è avuta in [BGL92, BGL93] ed è stata ulteriormente elaborata in [Bag97, capitolo 5]. Avendo a disposizione un dominio adeguato per gli intervalli (estesi), l'implementazione delle operazioni di base sulle differenze vincolate è relativamente semplice. Però, come nel caso degli ottagoni (si veda più avanti), per alcune operazioni importanti, come il widening, al momento attuale non sono note realizzazioni efficienti. È quindi richiesto un ulteriore lavoro teorico che consenta di ottenere per questo dominio la completezza funzionale desiderata. Per gli operatori di widening, si prevede di utilizzare le idee presentate in [BHRZ03, BHZ03b].

Fissata una base per i logaritmi, un sistema di divisioni vincolate cattura le relazioni della forma `log |x| - log |y| in S' [Bag97]. Le divisioni vincolate presentano somiglianze con le differenze vincolate e molti aspetti dell'implementazione possono essere riciclati. Rimane però molto lavoro da fare, sia a livello teorico che a livello pratico, per trasformare questa idea in un vero e proprio dominio astratto. Si noti che il dominio delle divisione vincolate coinvolge vincoli non lineari. La necessità, in alcune applicazioni dell'analisi statica, di andare oltre il potere espressivo dei vincoli lineari è stata recentemente sottolineata in [BCC+03], dove si mostra come un dominio basato su ellissoidi sia stato cruciale per il successo di una specifica analisi. Riteniamo che il dominio delle divisioni vincolate sarà utile per analisi quali l'identificazione di overflow a tempo di compilazione.

Ottagoni e oltre

Il dominio degli ottagoni è noto essere caratterizzato da costi computazionali ragionevoli, sebbene il costo cubico (nel numero delle variabili) sia applicabile solo ad una parte delle operazioni di interesse. I limiti dell'attuale definizione (e quindi dell'implementazione) di questo dominio sono: l'aggiunta di un vincolo lineare può essere effettuata efficientemente solamente nel caso di vincoli di forma speciale; il calcolo delle immagini affini dirette ed inverse è definito solo per una classe limitata di trasformazioni affini; non è disponibile una specifica efficiente per l'operazione di time-elapse [HPR97], utile per l'analisi dei sistemi ibridi lineari; non è disponibile un vero e proprio operatore di widening: solo operatori di estrapolazione che non garantiscono la convergenza dell'analisi. In aggiunta, nessuna implementazione supporta i vincoli stretti, per cui non è possibile definire ottagoni non necessariamente chiusi (NNC). In questo progetto intendiamo superare tutte queste limitazioni. In particolare, saranno ricercati algoritmi efficienti per realizzare le operazioni sopra elencate, utilizzando le tecniche descritte in [BHRZ03, BHZ03b] per fornire il dominio di veri e propri operatori di widening ad elevata precisione. In una prima fase, sarà sperimentata un'implementazione degli ottagoni basata su matrici di differenze vincolate [Min01]. Si passerà quindi all'implementazione proposta in [Bag97], che prevede la rappresentazione degli ottagoni come grafi di vincoli sulle classi di intervalli descritte precedentemente. Questo porterà immediatamente alla definizione di un dominio di ottagoni non necessariamente chiusi e consentirà una generalizzazione verso domini astratti più espressivi degli ottagoni, comunque caratterizzati da una complessità computazionale accettabile.

Poliedri convessi

Per quanto riguarda il dominio dei poliedri convessi, ci proponiamo di studiare algoritmi specializzati per classi ristrette di poliedri che sono comunemente utilizzate in molte applicazioni. Un esempio è la classe dei poliedri non negativi (per approssimare i sottoinsiemi dello spazio vettoriale i cui elementi hanno coordinate maggiori od uguali a zero). La ``Parma Polyhedra Library'' sarà anche estesa con un'implementazione efficiente dell'algoritmo del simplesso: questo consentirà, a sua volta, l'aggiunta di alcune

nuove funzionalità quali il calcolo efficiente dell'inviluppo di due poliedri come approssimazione del loro inviluppo poliedrale convesso.

Considereremo anche il problema della manipolazione di poliedri NNC, ovvero poliedri che possono essere descritti da sistemi di vincoli nei quali possono comparire disuguaglianze strette. Tali disuguaglianze sono importanti, ad esempio, per modellare facilmente il caso di regioni temporali che non si intersecano, come accade spesso nel caso di applicazioni in cui siano presenti concorrenza, protocolli di sincronizzazione, interazioni asincrone e vincoli temporali. Le implementazioni disponibili basate sul metodo della doppia descrizione codificano i poliedri NNC come poliedri chiusi in uno spazio vettoriale di dimensione superiore. Sebbene questa codifica consenta di riutilizzare una parte sostanziale del codice (quasi) invariata, alcuni degli operatori risultanti incorrono in perdite di efficienza o precisione evitabili. Per esempio, una potenziale perdita di precisione si ha quando la convergenza del calcolo di punto fisso viene forzata applicando il widening alla rappresentazione chiusa del poliedro NNC. Tale imprecisione può essere evitata definendo un operatore di widening specifico per i poliedri NNC. Come ulteriore esempio, il calcolo di una rappresentazione in forma minima per un poliedro NNC può comportare perdite di efficienza nel prosieguo della computazione, in quanto le procedure di minimizzazione non sono incrementali: la minimizzazione della rappresentazione a vincoli rende invalida la rappresentazione a generatori (e viceversa). Una possibile soluzione potrebbe essere data dall'adozione sistematica di tecniche di computazione lazy. Un'altra possibilità è la specifica di procedure di minimizzazione completamente incrementali, ovvero che consentano non solo l'aggiunta ma anche la rimozione di vincoli/generatori.

Griglie e Z-poliedri

Molte applicazioni dell'analisi statica, come il buffer overflow, l'analisi di terminazione e la parallelizzazione automatica, devono calcolare valori su domini discreti. Candidati per tali applicazioni sono i domini astratti aventi per elementi dei reticoli di vettori a coordinate intere [Anc91, BK97, CL98, DRS01, Gra91, LW97, Pug92]. Per molte applicazioni, però, tali domini sono inadeguati, perché i valori di interesse possono essere non interi oppure le componenti dei reticoli possono essere insiemi di vettori, invece di singoli vettori [Gra97, Mas93]. Ci proponiamo di fornire una definizione generica di dominio di griglie, i cui elementi sono reticoli aventi per componenti altri elementi di domini astratti numerici [Gra97, Mas93]. Tali domini saranno dotati di tutti gli operatori necessari per la loro applicazione all'analisi statica.

Un dominio particolarmente importante ed utile che usa questo tipo di griglie è quello degli Z-poliedri, i cui elementi sono definiti dall'intersezione di un poliedro con una griglia discreta (a coordinate intere) [Anc91, NR00, QRR96]. Le operazioni su questo dominio sono spesso inefficienti o non completamente specificate. Molto lavoro di ricerca è quindi richiesto per la rigorosa formalizzazione ed implementazione dei domini a griglia e, basandosi su questi, per il progetto di un vero e proprio dominio di griglie poliedrali.

Oltre allo studio, progettazione ed implementazione dei domini astratti, l'unità di ricerca di Parma seguirà da vicino la loro applicazione a problemi di analisi statica, in collaborazione con le altre unità del progetto e con altri gruppi di ricerca (si veda http:/www.cs.unipr.it/ppl/Credits/).

Applicazioni all'analisi di complessità

Nel contesto del Workpackage 5 (Vincoli) del progetto ci proponiamo di studiare il problema di fornire un'analisi di complessità completamente automatica, di fronte al quale l'unità di ricerca si trova in una situazione privilegiata. Infatti, al fine di affrontare questo problema, si vorrebbero combinare il lavoro sui domini numerici astratti delineato sopra con il nostro lavoro sulla soluzione automatica di relazioni di ricorrenza (si vedano http://www.cs.unipr.it/purrs/ e [BZZ03, BZ04]). Un modo di esprimere una tale combinazione di informazioni su data-flow e control-flow è per mezzo di un'integrazione ad accoppiamento lasco dei corrispondenti sistemi di vincoli, mantenendo sotto controllo la complessità delle interazioni risultanti. Le relazioni di ricorrenza possono essere viste come vincoli su sequenze di numeri reali che mettono in corrispondenza la dimensione di un problema (una misura di alcuni dei parametri di input) con una limitazione superiore o inferiore della quantità di risorse necessarie per la sua soluzione (ad esempio, il numero di passi della computazione). Oltre ad approfondire lo studio dei metodi automatici per risolvere o approssimare le relazioni di ricorrenza, verrà investigato il problema della loro estrazione dal sistema software analizzato, un'operazione che di solito nasconde ulteriori forme di approssimazione che possono notevolmente influenzare l'accuratezza complessiva dell'analisi. Per fornire una proof-of-concept preliminare, ci concentreremo su di un opportuno frammento di Objective CAML ed estenderemo le tecniche a sottoinsiemi più grandi del linguaggio in fasi successive. Quando saranno considerati gli aspetti object-oriented del linguaggio, un'analisi di complessità accurata dovrà basarsi sulle analisi di classe e di escape, che prevediamo di realizzare per mezzo dell'integrazione di vincoli insiemistici [DPPR00].

Testo inglese
Most of the work of this Research Unit will be done in the context of Workpackage 1 (Static Analysis) of the project and will look at the specification, development and implementation of abstract domains and operators, including widenings and narrowings operators, that can be used to enforce or accelerate the convergence of the abstract computation. More specifically, the main objective of this research is the theoretical study and practical implementation of a complete and robust library of reusable numerical abstract domains: intervals, bounded differences and quotients, octagons, convex polyhedra, grids and Z-polyhedra.

The domain of convex polyhedra has several operations that are characterized by an exponential worst-case complexity for both time and space resources. On the other hand, the interval domain, while being much simpler from a computational point of view, often yields approximations that are too coarse. An intermediate solution is given by the domains of bounded differences and octagons. Industrial users of static analysis tools often need the accuracy provided by the most precise abstract domains; however, they also

need some form of control over time and space consumption, since approximate results within a given time frame are preferred to more precise results at an unknown date. We will thus pursue the possibility of dynamically combining polyhedra with these simpler constraint domains in order to solve the precision and scalability trade-off. The static analysis or verification procedure, by default, should consider the more descriptive (and costly) domains, such as convex polyhedra, yet specify maximal resource bounds (computation time, memory occupation, magnitude of the numerical coefficient involved) for each operation. When the computation of an operation exceeds the imposed space or time limits, a simpler, less accurate algorithm is used, which still provides correct results for an abstract interpretation, i.e., such that the resulting polyhedron is a superset of the exact result. The less accurate algorithm will perform a change of representation, for instance by approximating the polyhedra by a system of bounded differences and computing the required operation on that simpler domain. If the cost of computing with bounded differences still happens to be unaffordable, then the system should perform a further change of representation, e.g., scaling down to the domain of intervals, where most of the interesting operations have linear complexity. After the execution of the requested operation on the less precise domain, the result may eventually be converted back into, say, a convex polyhedron. The final objective is the realization of a robust library of operators on polyhedra able to deal with large sets of many hundreds (or, possibly, thousands) of constraints in high dimensional spaces.

It is important to notice that such a scenario is feasible only if all the involved abstract domains have been designed and implemented so that they consistently react, without any loss of system resources, to events such as a timeout or an exception thrown by the memory allocation routines. The fact that we have achieved this in the ``Parma Polyhedra Library'' [BRZH02], which will be one of the key and more complex components in our library of numerical abstract domains, is thus essential.

Other features that are important for all members of the library are: support for every operator needed by an abstract interpreter; support for interoperability, that is, the ability to build any kind of description starting from any other kind of description; support for space decomposition operators that can help speed the execution [HMPV03]; availability of a database of difficult non-regression tests for each operator.

Intervals

A good abstract domain for intervals is highly desirable not only as the fall-back domain, the most efficient (though not very precise) domain to be used when everything else has failed, but as a key ingredient for the realization of more complex domains such as bounded differences and quotients [Dav87, Bag97]. Unfortunately, no existing interval library is really adequate to constitute the basis of such an abstract domain. In fact, the available libraries either lack support for non-closed intervals (so that they are unable to represent constraints of the form `ax < b'), or they do not provide the right support for approximation in the sense of partial correctness (e.g., division by an interval containing zero gives rise to a runtime error instead of yielding an interval containing the result under the assumption that the concrete division approximated was not a division by zero), or they disregard rounding errors and are therefore unsafe. For instance, the filib++ library [LTW+01], which is very sophisticated and highly regarded in its own application field, suffers from the first two problems mentioned above.

We will therefore study and implement a complete abstract domain based on intervals. Such intervals will be parametric on a number of features and will support open as well as closed boundaries. It will be possible to choose boundaries within one of several families of numbers, including native integers, unlimited precision integers, rationals, and native floating point types. When boundaries are floating point numbers, several methods of controlling the rounding directions will be provided with different characteristics concerning portability and efficiency. Independently from the type of the boundaries, both intervals of real numbers and intervals of integer numbers will be supported.

Bounded Differences and Quotients

A domain of bounded differences allows to express the relative values of two quantities by means of finite sets of constraints of the form `x-y in S', where `S' is a subset of the reals [Dav87]. The first proposal for the use in static analysis of this and other descriptions originated in the field of artificial intelligence is due to [BGL92, BGL93] and was further elaborated in [Bag97, Chapter 5]. With a suitable domain for (extended) intervals the implementation of the basic operations for bounded differences is relatively easy. However, as is the case for octagons (see below), for several important operations, including a proper widening, no efficient realization is known to date. Further theoretical work is thus required in order to obtain a fully fledged abstract domain based on bounded differences. For the widening operators, we plan to use the ideas presented in [BHRZ03, BHZ03b].

Once one has fixed a base for logarithms, a system of bounded quotients captures relations of the form `log |x| - log |y| in S' [AK85, Bag97]. Bounded quotients are quite similar to bounded differences and several aspects of the implementations can be reused. However, much work remains to be done, both at the theoretical and at the practical levels, in order to turn this idea into an usable abstract domain. Notice that, differently from all the other domains we want to study in this project, the domain of bounded quotients involves non-linear constraints. The need, in some application of static analysis, to go beyond the expressive power of linear constraints has been recently highlighted in [BCC+03], where the definition and use of an abstract domain based on ellipsoids proved to be crucial for the success of the analysis. We conjecture that the domain of bounded quotients we propose will be quite powerful for analyses such as compile-time overflow detection.

Octagons and Beyond

While the octagon abstract domain is characterized by a quadratic memory cost (in the number of variables), the cubic time cost that is advertised for them is currently only available for a fraction of the operations of interest. The limitations of the current definition (and, of course, of the available implementations) of the octagon domain include the following: the efficient addition of a new linear constraint is only available for constraints that are of the special form above; the computation of affine images and preimages has only been defined for a very limited class of affine transformation; an efficient definition of the time-elapse operation [HPR97], which is useful for the analysis of linear hybrid systems, is unavailable; no proper widening has been defined for octagons: only extrapolation operators that do not provide any convergence guarantee are available. In addition, no currently available implementation supports strict constraints, so that they do not support octagons that are not necessarily closed (NNC). In this project we aim at overcoming all these limitations of the octagon domain. In particular, we will research efficient algorithms to realize the operations mentioned above, using the techniques described in [BHRZ03, BHZ03b] to endow the domain with very precise, proper widening operators. In a first phase we will experiment with the octagon implementation based on difference-bound matrices proposed in [Min01]. We will then bring forward the proposal in [Bag97] and study the realization of octagons as constraint networks based on the class of intervals outlined above. This will immediately result in a class of not necessarily closed octagons and, more generally will allow the generalization to abstract domains that are more expressive than octagons yet characterized by an acceptable computational complexity.

Convex Polyhedra

Regarding the domain of convex polyhedra, we plan to investigate specialized algorithms working on restricted classes of polyhedra that are commonly used in several application fields. An example is the class of non-negative polyhedra (for approximating those subsets of the vector space whose elements have all coordinates greater than or equal to zero). The ``Parma Polyhedra Library'' will also be extended with an efficient version of the simplex algorithm: this will, in turn, enable the addition of several new features such as the efficient computation of the envelope of two polyhedra as an approximation of their convex polyhedral hull.

We will also target the adequate handling of NNC polyhedra, i.e., polyhedra that can be described by systems of constraints where strict inequalities are allowed to occur. Strict inequalities are important, for instance, in order to directly represent non-intersecting temporal regions, as it is often the case when modeling applications where concurrency, synchronization protocols, asynchronous interactions and temporal constraints come into play. Current implementations based on the double description method represent NNC polyhedra as closed polyhedra in a higher dimension vector space. While this encoding allows for a substantial fraction of code to be reused (almost) unchanged, some of the resulting operators suffer from unnecessary performance and/or precision losses. A precision loss may be incurred, for instance, when enforcing the convergence of the fixpoint computation by widening the closed representations of NNC polyhedra. This could be avoided by providing a widening operator specifically designed for NNC polyhedra. As another example, a performance penalty can be met after the computation of a description in minimal form for an NNC polyhedron. The currently available minimization procedures are not incremental so that, when minimizing the constraint (resp., generator) description, they invalidate the dual generator (resp., constraint) description. A possible solutions to this problem is the wider adoption of lazy computation techniques, which would allow to preserve both descriptions until, by looking at the following computation, it would be possible to make a more reasoned choice. Another possibility is the specification of a fully-incremental minimization procedure, allowing for both the addition and the removal of constraints/generators.

Grids and Z-Polyhedra

Many applications of static analysis such as buffer overflow detection, verification of termination or automatic parallelization are concerned with discrete numerical values. Relevant candidates for these applications are those domains whose elements are lattices of vectors having integer coordinates [Anc91, BK97, CL98, DRS01, Gra91, LW97, Pug92]. However, for many applications these domains are inadequate as the values may not be integral or the lattice components may be sets of vectors rather than single vectors [Gra97, Mas93]. Thus we plan to develop a truly generic grids domain whose elements are lattices having components in other numerical abstract domains [Gra97, Mas93]. These grids will be provided with all the basic operations needed in a static analyzer.

One important and useful analysis domain that uses grids is that of Z-polyhedra whose elements can be defined as the intersection of a polyhedron and a grid [Anc91, NR00, QRR96]. The existing operations on this domain are often inefficient or not fully specified. Research is therefore needed in formalizing and implementing the grid domain and then, building on this work, to develop a grid-polyhedral domain.

In collaboration with the other units and with other research groups (see http:/www.cs.unipr.it/ppl/Credits/), the Parma research unit will closely follow any further work on static analysis applications relevant to these and similar abstract domains.

Application to Complexity Analysis

Another line of research will be pursued in the context of Workpackage 5 (Constraints) of the project. We plan to study the problem of fully automatic complexity analysis, for which the research unit is in a unique position. In fact, in order to attack this problem, we would combine the work on numerical abstract domains outlined above with our work on the automatic solution and approximation of recurrence relations (see http://www.cs.unipr.it/purrs/ and [BZZ03, BZ04]). One way to express such a combination of data-flow

and control-flow information is by a loosely coupled integration of the corresponding constraint systems, so as to keep the complexity of the resulting interactions under control. In fact, recurrence relations can be seen as constraints on sequences of real numbers, therefore relating the size of a problem (e.g., a measure of some of its input parameters) with the upper or lower bounds on the amount of resources needed for solving it (e.g., an upper bound on the number of computation steps). Besides deepening the study of automatic methods to solve or approximate recurrence relations, we will study the problem of their extraction from the analyzed software systems, an operation that usually hides other forms of approximation that can greatly influence the overall accuracy of such an analysis. In order to provide a first proof-of-concept, we will initially focus on a suitable fragment of Objective CAML and extend the techniques to larger fragments in later phases. When the object-oriented aspects of the language will be considered, precise complexity analysis will require class and escape analyses, which we plan to realize with the integration of set-constraints [DPPR00].

2.6 Descrizione delle attrezzature già disponibili ed utilizzabili per la ricerca proposta con valore patrimoniale superiore a 25.000 Euro

Testo italiano

Nessuna

Testo inglese

Nessuna

2.7 Descrizione delle Grandi attrezzature da acquisire (GA)

Testo italiano

Nessuna

Testo inglese

Nessuna

2.8 Mesi uomo complessivi dedicati al programma

		Numero	Mesi uomo 1° anno	Mesi uomo 2° anno	Totale mesi uomo
Personale universitario dell'Università sede dell'Unità di Ricerca		*4*	*30*	*16*	*46*
Personale universitario di altre Università		*0*	*0*	*0*	*0*
Titolari di assegni di ricerca		*1*	*5*	*3*	*8*
Titolari di borse	*Dottorato*	*0*			
	Post-dottorato	*0*			
	Scuola di Specializzazione	*0*			
Personale a contratto	*Assegnisti*	*1*	*8*	*3*	*11*
	Borsisti	*0*			
	Dottorandi	*0*			
	Altre tipologie	*1*	*0*	*8*	*8*
Personale extrauniversitario		*0*			
TOTALE		**7**	**43**	**30**	**73**

PARTE III

3.1 Costo complessivo del Programma dell'Unità di Ricerca

Testo italiano

Voce di spesa	Spesa in Euro	Descrizione
Materiale inventariabile	*6.000*	*3 personal computer "PC compatibili", uno dei quali in configurazione "server"; 1 gruppo di continuità; 1 stampante laser; libri*
Grandi Attrezzature		
Materiale di consumo e funzionamento	*1.000*	*spese telefoniche; fotocopie; CD e DVD masterizzabili; spese di manutenzione*
Spese per calcolo ed elaborazione dati		
Personale a contratto	*29.000*	*1 assegno di ricerca annuale (11 mesi/uomo); 1 programmatore (8 mesi/uomo)*
Servizi esterni		
Missioni	*10.000*	*incontri di lavoro con altri componenti del progetto e spese di viaggio e soggiorno per la partecipazione a conferenze e workshop di interesse per le ricerche relative al progetto*
Pubblicazioni		
Partecipazione / Organizzazione convegni	*3.000*	*tasse di iscrizione a conferenze e workshop di interesse per le ricerche relative al progetto*
Altro	*1.000*	*seminari di interesse per il programma di ricerca*
TOTALE	**50.000**	

Testo inglese

Voce di spesa	Spesa in Euro	Descrizione
Materiale inventariabile	*6.000*	*3 PC compatible personal computers, one of those in server configuration; 1 uninterruptible power supply; 1 laser printer; books*
Grandi Attrezzature		
Materiale di consumo e funzionamento	*1.000*	*telephone expenses; xerox-copies; writable CDs and DVDs; maintenance expenses*
Spese per calcolo ed elaborazione dati		
Personale a contratto	*29.000*	*1 one-year research grant (11 man-months); 1 programmer (8 man-months)*
Servizi esterni		
Missioni	*10.000*	*working meetings with other members of the project and travel and living expenses for the participation to conferences and workshops of interest for the research topics of the project*
Pubblicazioni		
Partecipazione / Organizzazione convegni	*3.000*	*registration fees for conferences and workshops of interest for the research topics of the project*
Altro	*1.000*	*seminars of interest for the research program*
TOTALE	**50.000**	

3.2 Costo complessivo del Programma di Ricerca

Costo complessivo del Programma dell'Unità di Ricerca	*50.000*	
Fondi disponibili (RD)	*9.100*	*Fondi FIL, Dipartimento di Matematica*
Fondi acquisibili (RA)	*5.900*	*Fondo locale di cofinanziamento FIN, Università di Parma*
Cofinanziamento di altre amministrazioni		
Cofinanziamento richiesto al MIUR	*35.000*	

3.3.1 Certifico la dichiarata disponibilità e l'utilizzabilità dei fondi di Ateneo (RD e RA)

SI

Occorre precisare che la quota di cofinanziamento MIUR più la quota di cofinanziamento di altre amministrazioni cofinanziatrici del Programma di Ricerca non potrà superare il 70% per programmi Interuniversitari e il 50% per programmi Intrauniversitari del costo totale ammissibile del Programma stesso.

(per la copia da depositare presso l'Ateneo e per l'assenso alla diffusione via Internet delle informazioni riguardanti i programmi finanziati e la loro elaborazione necessaria alle valutazioni; legge del 31.12.96 n° 675 sulla "Tutela dei dati personali")

Firma _____ Data 18/03/2004 ore 17:40