

Analisi dei Sistemi Informatici: Dominio Astratto degli Intervalli

Marco Colognese VR423791

Mattia Rossini VR423614

Università degli Studi di Verona

Corso di Laurea Magistrale in Ingegneria e Scienze Informatiche

Ottobre 2018

1 Background

2 Dominio astratto degli Intervalli

3 Implementazione

4 Conclusioni

L'**analisi statica** permette di calcolare un'approssimazione dell'insieme dei valori o dei comportamenti che si verificheranno durante l'esecuzione di un programma senza avviarlo.

Approssimazione: se denotiamo con $\llbracket \cdot \rrbracket$ il comportamento concreto del programma e con $\llbracket \cdot \rrbracket^\#$ quello approssimato si ha che:

$$\llbracket P \rrbracket \subseteq \llbracket P \rrbracket^\#$$

Uno dei principali approcci dell'analisi statica è l'interpretazione astratta.

L'**interpretazione astratta** è un framework per l'analisi statica che definisce un'astrazione corretta per la semantica concreta di un programma.

Per creare un'**astrazione** occorre:

- un *dominio concreto* C : è il punto di partenza;
- un *dominio astratto* A : è un'approssimazione di C e modella alcune proprietà dei calcoli concreti, tralasciando informazioni superflue;
- una *funzione di astrazione*: $\alpha : C \rightarrow A$;
- eventualmente una *funzione di concretizzazione*: $\gamma : A \rightarrow C$.

Dominio astratto degli Intervalli

Definizione

Il **dominio astratto degli Intervalli** è un dominio numerico non relazionale in cui un insieme di interi viene approssimato dal più piccolo intervallo che li contiene ed è così definito:

$$\mathbb{I} = \{[l, u] \mid l \in \mathbb{Z} \cup \{-\infty\}, u \in \mathbb{Z} \cup \{+\infty\}, l \leq u\}$$

Questo dominio è un **reticolo** completo ed in particolare:

- l'ordinamento \sqsubseteq è tale che $[a, b] \sqsubseteq [c, d]$ solo se l'intervallo $[a, b]$ è interamente contenuto in $[c, d]$, cioè $a \geq c$ e $b \leq d$;
- l'elemento *top* \top è l'intervallo $[-\infty, \infty]$ che contiene tutti gli altri;
- l'elemento *bottom* \perp è l'insieme vuoto che non contiene elementi.

Dominio astratto degli Intervalli

Accelerazione della convergenza

Questo dominio non rispetta la ACC, dunque non garantisce la terminazione: perciò viene introdotto il **widening** ∇ . Funziona come segue:

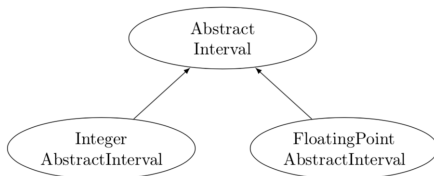
$$\begin{aligned} [a, b] \nabla [c, d] &= [e, f] \quad \text{t.c.} \\ e &= \begin{cases} -\infty & \text{se } c < a \\ a & \text{altrimenti} \end{cases} & f &= \begin{cases} +\infty & \text{se } b < d \\ b & \text{altrimenti} \end{cases} \end{aligned}$$

Utilizzando il widening può capitare di avere eccessive perdite di precisione, perciò viene introdotto il **narrowing** \triangle . Funziona come segue:

$$\begin{aligned} [a, b] \triangle [c, d] &= [e, f] \quad \text{t.c.} \\ e &= \begin{cases} c & \text{se } a = -\infty \\ a & \text{altrimenti} \end{cases} & f &= \begin{cases} d & \text{se } b = +\infty \\ b & \text{altrimenti} \end{cases} \end{aligned}$$

La libreria implementata è composta dalle seguenti classi:

- *AbstractInterval*: classe astratta, rappresenta un intervallo generico;
- *IntegerAbstractInterval* e *FloatingPointAbstractInterval*: estendono *AbstractInterval* per intervalli rispettivamente interi e a virgola mobile;
- *Bound*: rappresenta un *bound* che compone un intervallo;
- *Infinity*: rappresenta un valore infinito con il relativo segno;
- *UndefinedOperationException*: rappresenta l'eccezione per le operazioni non definite ed estende la classe *exception*.



Implementazione

Linguaggio Toy

Per provare la libreria, è stato definito un semplice linguaggio chiamato **Toy**.

<program>	::= {<statement>\n}*
<statement>	::= <assignment> <conditional> <loop>
<assignment>	::= <identifier> = <expression>
<conditional>	::= if <condition>\n {<assignment>\n}* endif
<loop>	::= while <condition>\n {<assignment>\n}* endwhile
<expression>	::= <value> <value> <operator> <value>
<value>	::= <identifier> <number> -<number>
<condition>	::= <value> <comparison> <value> <boolean>
<boolean>	::= true false
<operator>	::= + - * /
<identifier>	::= <letter> <id>*
<comparison>	::= == != <= >= < >
<id>	::= <letter> <digit>
<number>	::= <digit>+ <digit>+.<digit>+
<letter>	::= a b ... z A B ... Z
<digit>	::= 0 1 2 3 4 5 6 7 8 9

L'interprete esegue in modo ordinato le seguenti fasi:

- 1 **cerca** il programma nel file *input.txt*;
- 2 **legge** il file riga per riga, riconoscendo lo **statement** di ognuno: assegnamento, *if* oppure *while*;
- 3 partendo dai valori delle variabili, genera i rispettivi **intervalli**;
- 4 modifica gli intervalli attraverso le **operazioni** della libreria;
- 5 **stampa** a video gli intervalli relativi a ciascuna variabile indicandone nome, *lower bound* e *upper bound*.

Per ogni classe è stato scritto un file di **test**, basandosi sulla libreria *Catch2*.

L'obiettivo è quello di fornire una **copertura** del codice il più vicino possibile al 100%. Questi sono stati utili per verificare la corretta implementazione della libreria anche nei casi limite delle operazioni tra intervalli.

Per compilare ed avviare i test eseguire i seguenti comandi:

- `make` (per compilare il codice);
- `make test` (per eseguire i test).

È stata scritta la **documentazione** attraverso lo standard tool *Doxygen*:

- per ogni **funzione** vengono definiti una breve descrizione (`@brief`), i parametri di input (`@param`) ed il valore di ritorno (`@return`);
- per ogni **classe** viene definita una breve descrizione (`@brief`);
- per ogni **file** vengono definiti il copyright (`@copyright`), la licenza (`@license`), gli autori (`@authors`), la data di produzione (`@date`) e la versione del progetto (`@version`).

Per generare la documentazione eseguire i seguenti comandi:

- `make` (per compilare il codice);
- `make doc` (per generare la documentazione).

Questa viene generata nella directory `/doc` in formato *HTML*.

Sviluppare questo progetto ci ha permesso principalmente di:

- imparare il linguaggio `C++`;
- approfondire a fondo l'analisi statica, l'interpretazione astratta e, soprattutto, il dominio degli Intervalli;
- implementare e successivamente osservare il comportamento ed il funzionamento di tale dominio in ambito pratico;
- mettere in pratica le conoscenze teoriche acquisite durante il percorso di studi triennale, ovvero creare un interprete, organizzare il codice in classi e documentarlo per migliorarne la leggibilità.