

Analisi di Sistemi informatici

Riassunto dei principali argomenti

Autore:

Davide Bianchi

Indice

1	Preliminari matematici	2
1.1	Ordini parziali	2
1.2	Reticoli	2
1.3	Teoremi di punto fisso	2
2	Interpretazione astratta	2
2.1	Introduzione	2
2.2	Connessione di Galois	3
2.3	Famiglie di Moore	4
2.4	Upper closure operator	4
2.5	Reticolo delle interpretazioni astratte	4
2.6	Computazioni astratte e concrete	5
2.7	Accelerazione della convergenza	5
2.7.1	Widening	5
2.7.2	Narrowing	6
2.8	Linguaggio e semantica	6

1 Preliminari matematici

1.1 Ordini parziali

1.2 Reticoli

1.3 Teoremi di punto fisso

2 Interpretazione astratta

2.1 Introduzione

Lo scopo è quello di trovare un'approssimazione di una semantica $\langle P \rangle$ di $\llbracket P \rrbracket$ tale per cui valgano:

- *correttezza*: $\llbracket P \rrbracket \subseteq \langle P \rangle$;
- *decidibilità*: $\langle P \rangle \subseteq Q$ è decidibile (Q è un insieme di semantiche che soddisfa la proprietà di interesse).

Se entrambe le proprietà sono soddisfatte, allora vale che

$$(\langle P \rangle \subseteq Q) \Rightarrow (\llbracket P \rrbracket \subseteq Q)$$

La semantica è data da una coppia $\langle D, f \rangle$ dove D è una coppia $\langle D, \leq_D \rangle$ rappresentante un dominio semantico e $f : D \rightarrow D$ è una funzione di trasferimento con una soluzione a punto fisso.

Dato un oggetto concreto, definiamo:

- un **oggetto astratto** come una rappresentazione matematica sovra-approssimata del corrispondente concreto;
- un **dominio astratto** come un insieme di oggetti astratti con delle operazioni astratte, che approssimano quelle concrete;
- una funzione di **astrazione** α che mappa oggetti concreti in oggetti astratti;
- una funzione di **concretizzazione** γ che mappa oggetti astratti in oggetti concreti.

La caratteristica peculiare delle astrazioni è che solo alcune proprietà vengono osservate con esattezza, le altre vengono solo approssimate. In sostanza, dato un dominio astratto A , gli elementi di A sono osservati con esattezza, gli altri sono approssimati o l'informazione è persa del tutto.

Proprietà. L'insieme delle proprietà $\mathcal{P}(\Sigma)$ di oggetti in Σ è l'insieme di elementi che gode di quella proprietà. Questo insieme di proprietà costituisce un reticolo completo

$$\langle \mathcal{P}(\Sigma), \subseteq, \emptyset, \cup, \cap, \neg \rangle$$

dove:

- \subseteq è l'implicazione logica;
- Σ è **true**;
- \cup è la disgiunzione (oggetti che godono di P o di Q appartengono a $P \cup Q$);
- \cap è la congiunzione (oggetti che godono di P e di Q appartengono a $P \cap Q$);
- \neg è la negazione (oggetti che non godono di P stanno in $\Sigma \setminus P$).

Direzione dell'astrazione. Quando si approssima una proprietà concreta $P \in \mathcal{P}(\Sigma)$ usando una proprietà astratta \bar{P} , deve essere stabilito un criterio per definire quando \bar{P} è un'approssimazione di P .

Si distinguono quindi i seguenti casi:

- approssimazione *da sopra*: $P \subseteq \bar{P}$;
- approssimazione *da sotto*: $P \supseteq \bar{P}$.

Dato un oggetto o , si vuole quindi sapere se $o \in P$:

$$P \supseteq \bar{P} : \begin{cases} \text{"Si"} & o \in \bar{P} \\ \text{"Non lo so"} & o \notin \bar{P} \end{cases} \quad P \subseteq \bar{P} : \begin{cases} \text{"No"} & o \notin \bar{P} \\ \text{"Non lo so"} & o \in \bar{P} \end{cases}$$

Migliore approssimazione. Definiamo come *migliore approssimazione* di una proprietà P in A il glb delle over-approximation di P in A , ossia:

$$\bar{P} = \bigcap \{ \bar{P}' \in A \mid P \subseteq \bar{P}' \} \in A$$

2.2 Connessione di Galois

Imponiamo il vincolo che α e γ siano monotone, allora concludiamo che:

- $\gamma \circ \alpha : C \rightarrow C$ è **estensiva**: $\gamma(\alpha(c)) \geq c$;
- $\alpha \circ \gamma : A \rightarrow A$ è **riduttiva**: $\alpha(\gamma(a)) \leq a$.

Le definizioni qui sopra dicono rispettivamente che:

- α perde informazione, e γ non la può recuperare;
- γ non perde informazione.

Definizione 2.2.1 (Connessione di Galois). *Dati due poset $\langle A, \leq_A \rangle$ e $\langle C, \leq_C \rangle$, e due funzioni monotone $\alpha : C \rightarrow A$ e $\gamma : A \rightarrow C$, diciamo che $\langle C, \alpha, \gamma, A \rangle$ è una connessione di Galois se:*

- $\forall c \in C : c \leq_C \gamma(\alpha(c))$
- $\forall a \in A : \alpha(\gamma(a)) \leq_A a$

Se inoltre vale che $\forall a \in A : \alpha(\gamma(a)) = a$, allora $\langle C, \alpha, \gamma, A \rangle$ è un'inserzione di Galois.

Una connessione e un'inserzione di Galois sono rappresentate rispettivamente come

$$C \xleftrightarrow[\alpha]{\gamma} A \quad C \xleftrightarrow[\alpha]{\gamma} \gg A$$

La funzione α è detta *aggiunta sinistra*, mentre la funzione γ è detta *aggiunta destra*.

Teorema 2.2.1. *Data una connessione di Galois $C \xleftrightarrow[\alpha]{\gamma} A$, sono equivalenti:*

- $C \xleftrightarrow[\alpha]{\gamma} A$;
- α è suriettiva;
- γ è iniettiva.

Inoltre, dati due domini astratti, non esistono due coppie (α, γ) che formino una connessione di Galois; quindi la connessione di Galois tra due domini è **unica**, e le funzioni sono identificabili attraverso:

$$\alpha(c) = \bigwedge \{ a \in A \mid c \leq_C \gamma(a) \}$$

$$\gamma(a) = \bigvee \{ c \in C \mid \alpha(c) \leq_A a \}$$

2.3 Famiglie di Moore

Definizione 2.3.1 (Famiglia di Moore). Sia L un reticolo completo. $X \subseteq L$ è una famiglia di Moore di L se

$$X = \mathcal{M}(X) = \left\{ \bigwedge S \mid S \subseteq X \right\}$$

dove

$$\bigwedge \emptyset = \top \in \mathcal{M}(X)$$

Da questa definizione segue che, ipotizzando che ogni proprietà concreta abbia una migliore astrazione $\bar{P} \in A$, implica che il dominio A è una famiglia di Moore.

2.4 Upper closure operator

Definizione 2.4.1 (Upper closure operator). Una funzione $f : P \rightarrow P$ su un poset $\langle P, \leq_P \rangle$ è un upper closure operator (uco) se soddisfa le seguenti proprietà:

- *estensività*: $\forall x \in P : x \leq_P \rho(x)$
- *monotonia*: $\forall x, y \in P : (x \leq_P y) \Rightarrow (\rho(x) \leq_P \rho(y))$
- *idempotenza*: $\forall x \in P : \rho(x) = \rho(\rho(x))$

I lower closure operator sono definiti in modo duale, specificando che ρ deve essere *riduttiva*, ovvero che $\forall x \in P : x \geq_P \rho(x)$.

Teorema 2.4.1. Data una connessione di Galois $C \xleftrightarrow[\alpha]{\gamma} A$ si ha che $\gamma \circ \alpha$ è un uco e $\alpha \circ \gamma$ è un lco.

Teorema 2.4.2. $C \xleftrightarrow[\alpha]{\gamma} A$ se e solo se A è isomorfo¹ ad una Moore family di C .

Teorema 2.4.3. Sia $\rho \in \text{uco}(C)$. Allora $\forall A \simeq \rho(C)$ si ha che $\exists \alpha, \gamma : C \xleftrightarrow[\alpha]{\gamma} A$

2.5 Reticolo delle interpretazioni astratte

I vari domini astratti possono essere comparati sulla base della loro precisione. In generale si può dire che un dominio astratto A_1 è più preciso di A_2 (indicato attraverso $A_1 \sqsubseteq A_2$) quando

$$\forall a_2 \in A_2, \exists a_1 \in A_1 \quad \text{tali che} \quad \gamma_1(a_1) = \gamma_2(a_2)$$

ovvero quando

$$\gamma(A_2) \subseteq \gamma(A_1)$$

Collegando agli uco, possiamo dire che

$$A_1 \sqsubseteq A_2 \Leftrightarrow \rho_1 \sqsubseteq \rho_2 \Leftrightarrow \rho_2(C) \subseteq \rho_1(C)$$

Definizione 2.5.1 (Reticolo delle int. astratte). Se C è un reticolo completo o un cpo, allora

$$\langle \text{uco}(C), \sqsubseteq, \sqcup, \sqcap, \lambda x. \top, \lambda x. x \rangle$$

è un reticolo completo dove $\forall \rho, \eta \in \text{uco}(C), \{\rho_i\}_{i \in I} \subseteq \text{uco}(C)$ e $x \in C$:

- $\rho \sqsubseteq \eta \Leftrightarrow \forall y \in C. \rho(y) \leq \eta(y) \Leftrightarrow \eta(C) \subseteq \rho(C)$
- $\left(\bigcap_{i \in I} \rho_i \right)(x) = \bigwedge_{i \in I} \rho_i(x)$
- $\left(\bigsqcup_{i \in I} \rho_i \right)(x) = x \Leftrightarrow \forall i \in I. \rho_i(x) = x$
- $\lambda x. \top, \lambda x. x$ sono rispettivamente top e bottom.

¹Con isomorfismo si intendono reticoli con la stessa struttura.

2.6 Computazioni astratte e concrete

Definizione 2.6.1 (Correttezza). *Data un'inserzione di Galois $C \xleftrightarrow[\alpha]{\gamma} A$, una funzione concreta $f : C \rightarrow C$ e una funzione astratta $f^\sharp : A \rightarrow A$ diciamo che f^\sharp è un'approssimazione corretta di f se*

$$\forall c \in C : \alpha(f(c)) \leq_A f^\sharp(\alpha(c)) \quad \text{backward}$$

o equivalentemente

$$\forall a \in A : f(\gamma(a)) \leq_C \gamma(f^\sharp(a)) \quad \text{forward}$$

Rinforzando la definizione e imponendo uguaglianza si perde l'equivalenza delle due espressioni sopra.

Definizione 2.6.2 (Completezza). *Data un'inserzione di Galois $C \xleftrightarrow[\alpha]{\gamma} A$, una funzione concreta $f : C \rightarrow C$ e una funzione astratta $f^\sharp : A \rightarrow A$ diciamo che f^\sharp è:*

- *backward-completa per f se $\forall c \in C : \alpha(f(c)) = f^\sharp(\alpha(c))$*
- *forward-completa per f se $\forall a \in A : f(\gamma(a)) = \gamma(f^\sharp(a))$*

La definizione rappresenta una situazione ideale in cui non si ha perdita di precisione durante il calcolo astratto. Inoltre la backward-completezza lavora sull'astrazione dell'input delle operazioni, la forward-completezza sull'output.

Le definizioni di completezza possono essere date anche usando gli uco:

- $\rho \in \text{uco}(C)$ è backward-completo per f se $\rho \circ f = \rho \circ f \circ \rho$
- $\rho \in \text{uco}(C)$ è forward-completo per f se $f \circ \rho = \rho \circ f \circ \rho$

Inoltre quando ρ è sia backward che forward-completo allora vale che $\rho \circ f = f \circ \rho$.

Teorema 2.6.1. *Data $C \xleftrightarrow[\alpha]{\gamma} A$, una funzione concreta $f : C \rightarrow C$ e una funzione astratta $f^\sharp : A \rightarrow A$ allora*

$$\forall c \in C : \alpha(f(c)) \leq_A f^\sharp(\alpha(c)) \Leftrightarrow \alpha \circ f \circ \gamma \sqsubseteq f^\sharp$$

Definizione 2.6.3 (Best correct approximation). *Data $C \xleftrightarrow[\alpha]{\gamma} A$ e una funzione concreta $f : C \rightarrow C$ allora $\alpha \circ f \circ \gamma : A \rightarrow A$ è la best correct approximation di f in A .*

2.7 Accelerazione della convergenza

2.7.1 Widening

Un widening

$$\nabla : P \times P \rightarrow P$$

su un poset $\langle P, \leq_P \rangle$ è una funzione che soddisfa:

- $\forall x, y \in P : x \sqsubseteq (x \nabla y) \wedge y \sqsubseteq (x \nabla y)$
- per ogni catena ascendente $x_0 \sqsubseteq x_1 \sqsubseteq \dots \sqsubseteq x_n$ la catena definita come $y_0 = x_0, \dots, y_{n+1} = y_n \nabla x_{n+1}$ non è strettamente crescente.

Dato che in interpretazione astratta è necessario garantire/accelerare la convergenza, viene usato il widening (che si sostituisce al least upper bound), dal momento che anche il calcolo astratto può divergere. Il risultato di un widening è un post-puntofisso di F^∇ , ovvero una sovrapprossimazione del punto fisso più piccolo di $\text{flfp} F$.

Ad esempio, il widening su intervalli funziona come segue:

$$[a, b] \nabla [c, d] = [e, f] \quad \text{tale che}$$

$$e = \begin{cases} -\infty & \text{se } c < a \\ a & \text{altrimenti} \end{cases} \quad e \quad f = \begin{cases} +\infty & \text{se } b < d \\ b & \text{altrimenti} \end{cases}$$

2.7.2 Narrowing

Dato che il widening raggiunge un post-fixpoint, può capitare che si abbiano eccessive perdite di informazione, in questo caso viene usato il narrowing.

Definizione 2.7.1. *Il narrowing è una funzione $\triangle : P \times P \rightarrow P$ tale che:*

- $\forall x, y \in P : y \leq x \implies y \leq x \triangle y \leq x$
- Per ogni catena discendente $x_0 \geq x_1 \geq \dots$, la catena discendente $y_0 = x_0, \dots, y_{i+1} = y_i \triangle x_{i+1}$ non è strettamente decrescente.

Per gli intervalli il narrowing funziona come segue:

$$[a, b] \triangle [c, d] = [e, f] \quad \text{tale che}$$

$$e = \begin{cases} c & \text{se } a = -\infty \\ a & \text{altrimenti} \end{cases} \quad e \quad f = \begin{cases} d & \text{se } b = +\infty \\ b & \text{altrimenti} \end{cases}$$

2.8 Linguaggio e semantica

Introduciamo in questa sezione il linguaggio che verrà usato nel resto della dispensa e la sua semantica.

Statement	Codice
Variabili	x
Espressioni aritmetiche	e
Assegnamenti	x <- e
Lettura da memoria	x <- M[e]
Scrittura in memoria	M[e] ₁ <- e ₂
Condizionali	if (e) S ₁ else S ₂
Salto incondizionato	goto L

La memoria M è vista come un array arbitrariamente lungo dove i valori possono essere inseriti e letti successivamente.

Ogni passo di computazione della semantica operativa trasforma stati del programma (ρ, μ) dove $\rho : Var \rightarrow int$ e $\mu : \mathbb{N} \rightarrow int$. La funzione ρ mappa variabili di programma al loro valore attuale, mentre la funzione μ mappa ogni cella dell'array al suo contenuto.