Capitolo 8

Riassunto in preparazione all'esame

8.1 Definizioni utili / conoscenze base

- **Teorema SMN**: Dato $m, n \geq 1$, esiste $S : \mathbb{N}^{m+1} \to \mathbb{N}$ calcolabile e totale, tale che $\phi_e^{(m+n)}(\vec{x}, \vec{y}) = \phi_{S(e,\vec{x})}^{(n)}(\vec{y})$ per qualche $\vec{x} \in \mathbb{N}^m$ e $\vec{y} \in \mathbb{N}^n$.
- **Insiemi ricorsivi e RE**: dato un sottoinsieme $X \subseteq \mathbb{N}$, se la funzione caratteristica dell'insieme è calcolabile si dice che l'insieme è **ricorsivo**, mentre se è calcolabile solo la funzione semi-caratteristica si dice che è **Ricorsivamente Enumerabile**. Se un insieme è ricorsivo, il predicato $x \in A$ è decidibile, mentre se è RE, il predicato è semi-decidibile.
- **Insieme saturo**: un insieme $A \subseteq \mathbb{N}$ si dice **saturo** se, dati $x \in A$ e $y \in \mathbb{N}$, tali che $\phi_x = \phi_y$, allora anche $y \in A$. Ovvero un insieme è saturo quando contiene tutti gli indici che calcolano una determinata funzione. Alternativamente A è saturo se e solo se esiste $\mathcal{A} \subseteq \mathcal{C}$ tale che $A = \{x \,|\, \phi_x \in \mathcal{A}\}$, cioè l'appartenenza all'insieme dipende da delle caratteristiche della funzione calcolata dal programma e non da come questo è definito.
- $K = \{x \mid \phi_x(x) \downarrow\} = \{x \mid x \in W_x\}$, ovvero l'insieme dei programmi che terminano quando ricevono se stessi in input, **non** è ricorsivo, ma è RE. L'insieme **non è saturo** (si dimostra cercando un e' che calcola la stessa funzione di e ma che non termina su se stesso).
- **Teorema di Rice**: Il teorema di Rice asserisce che qualsiasi proprietà non banale delle funzioni calcolate dai programmi non è decidibile. In altre parole, sia $A \subseteq \mathbb{N}$ l'insieme che contiene tutti i programmi la cui funzione calcolata gode di una determinata proprietà (ovvero A è un insieme saturo), se la proprietà non è banale ($A \neq \emptyset, \neq \mathbb{N}$) allora A non è ricorsivo.
- $T = \{x | \phi_x \text{ è totale}\}$, è saturo, non è ricorsivo, è RE.
- **Funzione finita**: una funzione si dice finita se è definita solamente in un numero finito di elementi.
- **Pezzo di una funzione**: date due funzioni $f, g : \mathbb{N} \to \mathbb{N}$, f è un **pezzo** di g ($f \subseteq g$) se

$$\forall x\, f(x) \downarrow \Rightarrow g(x) \downarrow \text{ e } f(x) = g(x)$$

  Ovvero f è un pezzo di g solo se nei punti in cui è definita è uguale a g.
- $g(x, y)$: durante la **riduzione** $A \leq_m B$ si cerca una funzione $g(x, y)$ calcolabile e tale che vista come funzione di y appartenga a B solo se $x \in A$. Tipicamente viene utilizzata una funzione costante rispetto ad y, tuttavia in alcuni casi, ad esempio quando c'è la condizione $\phi_x(x) > x$ è necessario usare anche y, in modo che venga soddisfatta la condizione, ad esempio $g(x, x) > x$. Alternativamente y viene utilizzato quando la funzione deve avere determinate proprietà per qualche $n \in \mathbb{N}$.
- **Riduzioni tipiche:**
  - $K \leq_m A$ per provare che A non è ricorsivo

  - $A \leq_m K$ per provare che A è RE (se so che A non è ricorsivo, posso definire SC_A per provare che è RE, senza fare la riduzione)
  - $\overline{K} \leq_m A$ per provare che A non è RE.

- Una parte finitaria di una funzione totale è sempre calcolabile

8.2 Minimalizzazione illimitata

Data $f : \mathbb{N}^{k+1} \to \mathbb{N}$, si vuole definire $h : \mathbb{N}^k \to \mathbb{N}$ tale che calcoli il minimo z che azzera la funzione f, ovvero:

$$h(\vec{x}) = \mu z.f(\vec{x}, z)$$

Ci sono dei problemi se $f(\vec{x}, z)$ è sempre diversa da 0, perché in questo caso la funzione h è $\uparrow$. In modo simile c'è un problema se esiste un valore $z' < z$ per il quale la funzione f è indefinita perché, anche in questo caso, la funzione h risulta essere $\uparrow$.

Più formalmente, la minimalizzazione illimitata può essere vista come:

$$h(\vec{x}) = \mu z.f(\vec{x}, z) = \begin{cases} \text{minimo } z \text{ tale che } f(\vec{x}, z) = 0 \text{ se esiste, e } \forall z' < z, f(\vec{x}, z') \downarrow \neq 0. \\ \uparrow \ \text{altrimenti} \end{cases}$$

La classe $\mathcal{C}$ delle funzioni URM calcolabili è chiusa rispetto all'operazione di minimalizzazione illimitata, ovvero, se $f : \mathbb{N}^{k+1} \to \mathbb{N}$ è in $\mathcal{C}$, allora anche $\mu z.f(\vec{x}, z)$ è in $\mathcal{C}$.

Dimostrazione: Sia P il programma URM in forma normale che calcola f. L'idea è quella di eseguire P incrementando via via un contatore, e quando viene trovato un valore che azzera la funzione, questo viene ritornato.

Si tratta quindi di scrivere un programma che esegua il programma P, ogni volta su un valore diverso e in modo che non ci siano conflitti sui registri. Come prima cosa è quindi necessario stabilire il numero di registri utilizzato da P:

$$m = \max\big(\rho(P), k\big)$$

Dopodiché è possibile definire un programma *P'* che:

1. Copia il contenuto dei registri $r_1, \ldots, r_k$ in $r_{m+1}, \ldots, r_{m+k}$
2. Esegue il programma P con in input il valore dei registri $r_{m+1}, \ldots, r_{m+k}$ e pone l'output del programma nel registro r_1
3. Controlla se il contenuto del registro è 0. Se è 0 ritorna z, altrimenti continua incrementando z.

```
T([1..k], [m+1..m+k])
P[m+1, ..., m+k+1 -> 1]
J(1, m+k+2, END) #LOOP
S(m+k+1)
J(1,1,LOOP)
#END
```

Dal momento che è possibile trovare un programma che calcola la minimalizzazione illimitata, questa è calcolabile.

8.3 Funzioni primitive ricorsive

La classe delle funzioni primitive ricorsive $\mathcal{PR}$ è la **minima** classe di funzioni che contiene le funzioni

(a) $zero(x) = 0$

(b) $succ(x) = x + 1$

(c) $proj(x) = U_i^k(x_1, \ldots, x_k) = x_i$

ed è chiusa rispetto a

1. composizione
2. ricorsione primitiva

Si ha che $\mathcal{PR} \subsetneq \mathcal{R} = \mathcal{C}$ perché la classe $\mathcal{R}$ delle funzioni parziali ricorsive è chiusa anche rispetto la minimalizzazione illimitata.

Da notare che la minimalizzazione illimitata permette di definire delle funzioni che non sono totali e che $\mathcal{PR} \subsetneq \mathcal{R} \cap$ Totali perché esiste la funzione di Ackermann che è totale e calcolabile ma non è primitiva ricorsiva.

8.4 Riduzione

Si hanno due problemi $\mathcal{A}$ e $\mathcal{B}$ e si vuole poter dire se $\mathcal{A}$ è più facile o più difficile di $\mathcal{B}$.

Più formalmente siano $A, B \subseteq \mathbb{N}$ gli insiemi contenenti le istanze dei problemi. Si ha che il problema $\mathcal{A}$ si riduce al problema $\mathcal{B}$ se

$$\forall x \in A \Leftrightarrow f(x) \in B$$

Quindi, se $A \leq_m B$:

1. Se B è ricorsivo, anche A è ricorsivo
2. Se A non è ricorsivo, anche B non è ricorsivo
3. Se B è RE, anche A è RE
4. Se A non è RE, anche B non è RE

La dimostrazione viene effettuata ragionando sulle funzioni caratteristiche/semi-caratteristiche dei due insiemi e combinandole con la funzione f di riduzione.

Ad esempio, si ha che A è RE $\Leftrightarrow A \leq_m K = \{x \,|x \in W_x \}$ (K è RE). questo perché

- ($\Leftarrow$): K è RE e quindi SC_K è calcolabile. Dal momento che f è funzione di riduzione, questa è calcolabile e $x \in A \Leftrightarrow f(x) \in K$, quindi si può definire $SC_A(x) = SC_K(f(x))$. La funzione semi-caratteristica di A è calcolabile perché è ottenuta componendo funzioni calcolabili e quindi anche A è RE.
- ($\Rightarrow$): A è RE e quindi $SC_A(x)$ è calcolabile. Bisogna trovare una funzione di riduzione $f : \mathbb{N} \to \mathbb{N}$ tale che $x \in A \Leftrightarrow f(x) \in K$.

  Definiamo

  $$g(x, y) = SC_A(x) = \begin{cases} 1 & x \in A \\ \uparrow & \text{altrimenti} \end{cases}$$

  dato che g è calcolabile, per il teorema SMN esiste $f : \mathbb{N} \to \mathbb{N}$ tale che $\phi_{f(x)}(y) = g(x, y)$.

  Si ha quindi che f è proprio la funzione di riduzione $A \leq_m K$ perché:

  - Se $x \in A$: $\phi_{f(x)}(y) = 1$ e quindi $f(x) \in W_{f(x)}$, pertanto $f(x) \in K$.
  - Se $x \notin A$: $\phi_{f(x)}(y) =\uparrow$ e quindi $f(x) \notin W_{f(x)}$, pertanto $f(x) \notin K$.

8.5 Teorema di Rice

Ogni proprietà del comportamento di un programma non è decidibile.

Sia $A \subseteq \mathbb{N}$ una proprietà del comportamento di un programma (insieme saturo). Se la proprietà non è banale, l'insieme non è ricorsivo. Con proprietà banale si intende $A \neq \mathbb{N}$ e $A \neq \emptyset$.

8.5.1 Dimostrazione per riduzione

L'obiettivo è quello di dimostrare che $K = \{x \mid x \in W_x\} \leq_m A$. K è noto non essere ricorsivo.

Vogliamo quindi trovare una funzione $f : \mathbb{N} \to \mathbb{N}$ calcolabile e totale, tale che $x \in K \Leftrightarrow f(x) \in A$.

Consideriamo un programma $e_0 \in \mathbb{N}$ tale che $\phi_{e_0} = \emptyset$ (ovvero che non termina mai). Questo programma può trovarsi in A o in $\overline{A}$.

- $e_0 \in \overline{A}$: Sia $e_1 \in A$ un programma qualsiasi, ed esiste perché $A \neq \emptyset$.

  Possiamo definire

  $$g(x, y) = \begin{cases} \phi_{e_1}(y) & x \in K \\ \phi_{e_0}(y) = \uparrow & x \notin K \end{cases} = \phi_{e_1}(y) \cdot \mathbb{1}\big(\phi_x(x)\big)$$

  Questa funzione è calcolabile e quindi è possibile utilizzare il teorema SMN per trovare una funzione $f : \mathbb{N} \to \mathbb{N}$ calcolabile e totale, tale che

  $$g(x, y) = \phi_{f(x)}(y) \forall x, y$$

  e che può essere usata come funzione di riduzione.

  Questo perché:

  - Se $x \in K$ allora $\phi_{f(x)}(y) = \phi_{e_1}(y)\ \forall y$ e dato che A è saturo e che $e_1 \in A$, allora anche $f(x) \in A$ perché essendo A saturo contiene tutti gli indici che calcolano ϕ_{e_1}.
  - Se $x \notin K$ allora $\phi_{f(x)}(y) = \phi_{e_0}(y)\ \forall y$ e allo stesso modo dato che $e_0 \notin A$ anche $f(x) \notin A$, perché se $f(x) \in A$, allora anche e_0 dovrebbe appartenere ad A, ma questo è assurdo per ipotesi.

  Si ha quindi che f è la funzione di riduzione $K \leq_m A$ ed essendo K non ricorsivo, anche A **non è ricorsivo**.

- $e_0 \in A$: Sia $B = \overline{A}$, $B \neq \emptyset$ $B \neq \mathbb{N}$ e B è saturo perché è l'insieme complementare di un insieme saturo, ed infine $e_0 \notin B$. Per B valgono quindi tutte le condizioni del punto precedente, e quindi anche $B = \overline{A}$ non è ricorsivo.

8.6 Relazione tra R e RE

Un insieme $A \subseteq \mathbb{N}$ è ricorsivo se la sua funzione caratteristica $\mathcal{X}_A$ è calcolabile

$$\mathcal{X}_A(x) = \begin{cases} 1 & x \in A \\ 0 & x \notin A \end{cases}$$

Invece è ricorsivamente enumerabile se la funzione semi-caratteristica SC_A è calcolabile:

$$SC_A(x) = \begin{cases} 1 & x \in A \\ \uparrow & x \notin A \end{cases}$$

Si ha quindi che $A \subseteq \mathbb{N}$ è ricorsivo $\Leftrightarrow$ A, $\overline{A}$ sono RE.

8.6.1 A ricorsivo $\Rightarrow A, \overline{A}$ RE.

Se A è ricorsivo, la sua funzione caratteristica è calcolabile ed è definita come

$$\mathcal{X}_A(x) = \begin{cases} 1 & x \in A \\ 0 & x \notin A \end{cases}$$

Si può quindi definire la funzione semi-caratteristica come minimalizzazione illimitata di $\mathcal{X}_A$:

$$SC_A(x) = \nVdash \left(\mu w . \overline{sg}\Big(\mathcal{X}_A(x) \Big) \right)$$

così facendo si ottiene una funzione calcolabile che vale 1 se $x \in A$ (perché il segno negato vale 0) e risulta indefinita se $x \notin A$, perché la minimalizzazione non termina dato che il segno negato sarà sempre uguale a 1.

Per dimostrare che anche $\overline{A}$ è RE basta osservare che se A è ricorsivo, anche il suo complementare $\overline{A}$ è ricorsivo e quindi è possibile ripetere la stessa dimostrazione utilizzando $\mathcal{X}_{\overline{A}}$ per dimostrare che anche $\overline{A}$ è RE.

8.6.2 $A, \overline{A}$ RE $\Rightarrow A$ ricorsivo

Le due funzioni SC_A e $SC_{\overline{A}}$ sono calcolabili e quindi esistono due programmi e_1 ed e_2 le cui funzioni coincidono con le due funzioni semi-caratteristiche.

Si può quindi definire un programma che esegue i due programmi in parallelo e in base a quello che termina ritorna 0 oppure 1.

Questa funzione può essere definita come

$$\mathcal{X}_A(x) = \left(\mu w . S\Big(e_1, x, (w)_1, (w)_2 \Big) \vee S\Big(e_2, x, (w)_1, (w)_2 \Big) \right)_1$$

ed è calcolabile perché definita per minimalizzazione illimitata.

Alcune osservazioni:

- La funzione $S(e, \vec{x}, y, t)$ vale 1 se il programma e su input x termina entro t passi producendo come output y.
- La minimalizzazione su w viene effettuata per simulare l'avanzamento dei passi. Vengono utilizzati gli esponenti della rappresentazione binaria di w perché così durante la minimalizzazione vengono provati tutti i possibili valori.
- Perché tutto funzioni è necessario che la funzione ϕ_{e_2} sia uguale a $\overline{\text{sg}}(SC_{\overline{A}}(x))$, perché deve valere 0 quando $x \in \overline{A}$.

8.7 Teorema di struttura

Sia $P(x)$ un predicato k-ario, $P(x)$ è semi-decidibile se e solo se esiste $Q(t, x)$ decidibile e tale che

$$P(\vec{x}) \equiv \exists t \, Q(t, \vec{x})$$

Ovvero il predicato P è una generalizzazione di un predicato decidibile che viene calcolato su più punti. In generale, quantificare esistenzialmente un predicato decidibile lo rende semi-decidibile.

Questo perché la funzione semi-caratteristica di P non farà altro che applicare la minimalizzazione illimitata alla funzione caratteristica di Q per cercare un valore che soddisfa Q:

$$
\begin{aligned}
SC_P(x) &= \begin{cases} 1 \text{ se } P(x) \\ \uparrow \text{ altrimenti} \end{cases} \\
&= \mathbb{1}\Big(\mu t.\text{``}Q(t,x)\text{''}\Big) \\
&= \mathbb{1}\Big(\mu t. | \mathcal{X}_Q(t,x) - 1 |\Big)
\end{aligned}
$$

Viceversa, assumendo $P(x)$ semi-decidibile, si ha che SC_P è calcolabile, ovvero esiste un certo programma e tale che $\phi_e = SC_P$.

Quando il predicato $P(x)$ è vero, si ha che $SC_P(x) = 1 = \phi_e(x)$, ma questo vuol dire che il programma e termina su input x dopo un certo numero di passi t. Formalmente:

$$\phi_e(x) = 1 \Leftrightarrow \phi_e(x) \downarrow \Leftrightarrow \exists t \, H(e, x, t) = 1$$

Dove H è la funzione che ritorna 1 se la macchina e termina entro t passi sull'input x che sappiamo essere calcolabile.

Si ha quindi che, indicando il predicato $Q(t, x) \equiv \exists t \, H(e, x, t)$ è decidibile e $P(x) \equiv \exists t \, Q(t, \vec{x})$.

Utilizzando questo teorema è poi possibile andare a dimostrare il **teorema di proiezione**, ovvero che la classe RE è chiusa rispetto la quantificazione esistenziale. Lo si fa estraendo dal predicato semi-decidibile quello decidibile e poi si ragiona su come esprimere il predicato semi-decidibile come l'altro predicato semi-decidibile:

$$P'(\vec{y}) \equiv \exists x. \exists t. Q(t, x, \vec{y}) \equiv \exists w. \underbrace{Q((w)_1, (w)_2, \vec{y})}_{Q'}$$

8.8 Teorema di Rice-Shapiro

Le proprietà del comportamento (funzione calcolata) dei programmi possono essere decidibili se sono finitarie, ovvero dipendono da un numero finito di argomenti (es: *la funzione termina su 5*).

Più formalmente sia $\mathcal{A} \subseteq \mathbb{C}$ e $A = \{x \mid \phi_x \in \mathcal{A}\}$ (A saturo). Se A è RE, allora

$$\forall f \ \in \mathcal{C}. \Big(f \in A \Leftrightarrow \exists \vartheta \subseteq f \text{ e } \vartheta \in \mathcal{A} \Big)$$

A parole, sia $\mathcal{A}$ un sottoinsieme delle funzioni calcolabili e A l'insieme che contiene gli indici che calcolano queste funzioni. Se A è RE, allora una funzione f calcolabile appartiene ad $\mathcal{A}$ solo se esiste una sua parte finita che appartiene ad $\mathcal{A}$.

8.8.1 Dimostrazione

La dimostrazione si fa in due passi:

1. $\exists f, f \notin \mathcal{A}$ e $\exists \vartheta \subseteq f, \vartheta \in \mathcal{A} \Rightarrow A$ non è RE.
2. $\exists f, f \in \mathcal{A}$ e $\forall \vartheta \subseteq f, \vartheta \notin \mathcal{A} \Rightarrow A$ non è RE.

Caso 1 Sia $f \notin \mathcal{A}$ e $\vartheta \subseteq f, \vartheta \in \mathcal{A}$. Per provare che A non è RE viene fatta la riduzione $\overline{K} \leq_m A$.

Definiamo quindi

$$g(x,y) = \begin{cases} \vartheta(y) & x \in \overline{K} \\ f(y) & x \in K \end{cases} = \begin{cases} \uparrow & x \in \overline{K} \text{ e } y \notin dom(\vartheta) \\ f(y) = \vartheta(y) & x \in \overline{K} \text{ e } y \in dom(\vartheta) \\ f(y) & x \in K \end{cases}$$

$$= \begin{cases} f(y) & x \in K \text{ oppure } y \in dom(\vartheta) \\ \uparrow & \text{altrimenti} \end{cases}$$

Essendo $x \in K$ semi-decidibile, il predicato $Q(x,y) \equiv x \in K$ oppure $y \in dom(\vartheta)$ è semi-decidibile e pertanto la funzione g è calcolabile perché può essere definita come:

$$g(x,y) = f(y) \cdot SC_Q(x,y)$$

Essendo g calcolabile, per il teorema SMN esiste $S : \mathbb{N} \to \mathbb{N}$ calcolabile e totale, tale che $g(x,y) = \phi_{S(x)}(y)$ che può funzionare da funzione di riduzione:

- $x \in \overline{K}$: $\phi_{S(x)}(y) = g(x,y)\ \forall y$ ed in particolare $= \vartheta(y)\ \forall y \Rightarrow \phi_{S(x)} = \vartheta$ ed essendo A saturo, $S(x) \in A$.
- $x \in K$: $\phi_{S(x)}(y) = g(x,y) = f(y)\ \forall y$ da ui segue che $\phi_{S(x)} = f \notin A$ e quindi dato che A è saturo, anche $S(x) \notin A$.

Caso 2 Sia $f \in \mathcal{A}$ e tutte le sue parti finite non sono in $\mathcal{A}$, voglio arrivare a dire che A non è RE. Anche in questo caso si fa la stessa riduzione.

$$\begin{aligned} g(x,y) &= \begin{cases} f(y) & \text{se } \neg H(x,x,y) \\ \uparrow & \text{altrimenti} \end{cases} \\ &= f(y) \cdot \not\Vdash\big(\mu w. \mathcal{X}_H(x,x,y)\big) \end{aligned}$$

Essendo g calcolabile, per SMN esiste S che funziona da funzione riduzione:

- $x \in \overline{K}$: $\forall y\ \neg H(x,x,y) = 0$ e quindi $\phi_{S(x)}(y) = f(y)\ \forall y$, ovvero $\phi_{S(x)} = f$ e dato che A è saturo e $f \in A$, anche $S(x) \in A$.
- $x \in K$: $\exists y_0$ tale che $H(x,x,y_0) = 1$, perché x termina in un certo numero di passi quando riceve se stesso in input. Tra tutti i possibili valori, prendo il minimo y_0 tale che $\forall y < y_0\ \neg H(x,x,y) = 1$ e $\forall y \geq y_0\ H(x,x,y) = 1$.

  Si ha quindi che $\phi_{S(x)}(y) = g(x,y) = \begin{cases} f(y) & y < y_0 \\ \uparrow & \text{altrimenti} \end{cases}$.

  Essendo che $\phi_{S(x)}$ è una parte finita di f, si ha che $\phi_{S(x)} \notin A$ e quindi $S(x) \notin A$.

8.9 Secondo teorema di Ricorsione

Sia $h : \mathbb{N} \to \mathbb{N}$ calcolabile e totale. Allora esiste $e \in \mathbb{N}$ tale che $\phi_e = \phi_{h(e)}$.