Esercizi di introduzione alla calcolabilità

Esercizio 1 (nessun limite al tempo). Data un'enumerazione effettiva, esiste una funzione calcolabile totale $t(i, n)$ che maggiora il tempo (> 0) di calcolo di $M_i(n)$?

Svolgimento. Una tale funzione non esiste.[1]

Si definisca la funzione

$$t(i, n) = \begin{cases} k & \text{se } M_i(n) \downarrow \text{ in meno di } k \text{ passi} \\ 0 & \text{altrimenti} \end{cases}$$

e si supponga per assurdo $t(i, n)$ calcolabile totale.

Sia allora T_i la misura *esatta* del tempo di calcolo di M_i. Per ogni n, la funzione test $T_i(n) \overset{?}{\leq} t(i, n)$ è calcolabile totale poiché basta far girare $M_i(n)$ per al più $t(i, n)$ passi: se l'algoritmo si è arrestato allora il test ha risultato affermativo, altrimenti il test ha risultato negativo. Si definisca la funzione ausiliaria

$$\psi(x) = \begin{cases} \varphi_x(x) + 1 & \text{se } T_x(x) \leq t(x, x) \\ 0 & \text{altrimenti} \end{cases}$$

e, per il ragionamento appena svolto, si osservi che $\psi(x)$ è calcolabile totale. Quindi per la tesi di Church-Turing esiste un indice i tale che $\psi = \varphi_i$. Ma da

$$\varphi_i(i) = \psi(i) = \begin{cases} \varphi_i(i) + 1 & \text{se } T_i(i) \leq t(i, i) \\ 0 & \text{altrimenti} \end{cases}$$

e dalla osservazione $\varphi_i(i) \neq \varphi_i(i)+1$, segue che $\varphi_i(i) = 0$ e dunque $T_i(i) > t(i, i)$. Ciò contraddice l'ipotesi $T_i(i) \leq t(i, i)$ e quindi $t(i, n)$ non è calcolabile totale.

Esercizio 2 (nessun limite allo spazio). Esiste una funzione calcolabile totale che determina se un dato programma M_i calcolato su uno specifico dato x all'interno di un calcolatore C con memoria finita è in ciclo?

Svolgimento. Si definisca

$$h(i, x, C) = \begin{cases} 1 & \text{se } M_i(x) \uparrow \text{ sul calcolatore } C \\ 0 & \text{altrimenti} \end{cases}$$

La funzione totale h è calcolabile? Aver limitata la memoria di C ci permette di stimare il numero massimo di configurazioni di C, e dunque di avere una

[1] Si pensi alle *barre di avanzamento* mostrate da molti programmi: il loro movimento procede a scatti. Se esistesse $t(i, n)$ calcolabile totale, dal confronto del tempo già passato con $t(i, n)$ sarebbe possibile far scorrere una barra di avanzamento in modo continuo.

procedura che in un numero finito di passi stabilisce se $M_i(x)$ è in ciclo o meno.[2] Si rappresenti infatti il calcolatore C come una MdT e siano $n = \#\Sigma_C, m-1 = \#Q_C$ e k le celle del nastro di C. Si possono avere al più n^k stringhe diverse sul nastro di C, su cui la testina si può trovare in al più k posizioni diverse, in al più m stati diversi. Dunque il numero massimo di configurazioni differenti che C può assumere è $l = k \cdot n^k \cdot m$. Sia H la MdT a 4 nastri che al principio contengono sul primo $\langle M_i \rangle$, codifica della MdT M_i, sul secondo $\langle x \rangle$, codifica del dato x, sul terzo la codifica dello stato iniziale di M_i e sul quarto la codifica in unario di l. La MdT H simula il comportamento di $M_i(x)$ lavorando come la MdT universale descritta nel corso, e in più decrementa di una tacca il quarto nastro ogni volta che simula una computazione di $M_i(x)$. Se sul terzo nastro compare lo stato di arresto di M_i e sul quarto nastro vi sono ancora tacche allora H si ferma e restituisce $h(i,x,C) = 0$, altrimenti se sul quarto nastro non vi sono più tacche ed M_i non è nello stato di arresto allora H si ferma e restituisce $h(i,x,C) = 1$ poiché M_i è destinata a ritornare in una delle configurazioni già incontrate.

È necessario notare come l'aver definito i nastri delle MdT infiniti a destra renda possibile la definizione della macchina H. Se imponessimo un limite di spazio, in particolare al quarto nastro di H, sarebbe possibile trovare un calcolatore C con un l maggiore del limite, e dunque la funzione h(i,x,C) diverrebbe non calcolabile.

Detto in altre parole, se non avessimo il vincolo della memoria finita di C, la funzione $h(i,x)$ (a due posti invece che tre) non sarebbe calcolabile, infatti risulterebbe $h(i,i) = 1 - \chi_K(i)$, che sappiamo non essere calcolabile.

Esercizio 3. Si dimostri che un insieme infinito A è ricorsivo se e solo se può essere generato in modo strettamente crescente, cioè esiste una funzione strettamente crescente g tale che $A = Imm(g)$.

Svolgimento. Se A è ricorsivo allora la sua funzione caratteristica è ricorsiva, dunque possiamo definire (con un po' di libertà) la funzione g come

$$\begin{cases} g(0) & = \mu y[\chi_A(y) = 1] \\ g(n+1) & = \mu y[\chi_A(y) = 1 \wedge g(n) < y] \end{cases}$$

La funzione che verfica $<$ è primitiva ricorsiva, e la costruenda funzione g è strettamente crescente. Per mostrare che $A = Imm(g)$ basta ordinare in modo crescente gli elementi di A, cioè riordinare $A = \{a_i\}_i$ in modo che $a_{i_{n+1}}$ sia l'elemento di A immediatamente successivo ad a_{i_n}, e osservare che per ogni n

[2]Nei primi calcolatori, l'operatore sedeva alla consolle, che era munita di una fila di lampadine che mostravano lo stato di avanzamento del programma; uno dei compiti dell'operatore era proprio quello di interrompere l'esecuzione, avendo rilevato un ciclo dal fatto che che la stessa configurazione si ripeteva.

vale $a_{i_n} = g(n)$. In simboli

$$\begin{aligned} A &= \{a_{i_0}, a_{i_1}, \dots, a_{i_n}, \dots\} \\ &= \{g(0), g(1), \dots, g(n), \dots\} \end{aligned}$$

Se $A = Imm(g)$, con g calcolabile totale strettamente crescente, per dimostrare che A è ricorsivo bisogna far vedere che χ_A è una funzione calcolabile totale. Si definisca allora la seguente funzione che lo è banalmente (B è un insieme finito):

$$\varphi_A(n) = \begin{cases} 1 & \text{se } n \in B = \{g(0), \dots, g(n)\} \subset A \\ 0 & \text{altrimenti} \end{cases}$$

Verifichiamo che $\varphi_A = \chi_A$, cioè che la funzione appena definita è davvero la funzione caratteristica di A:

- se $n \in B$ allora $n \in A$ per definizione
- se $n \notin B$ allora $n < g(n) < g(n+1) < \dots$ e quindi $n \notin A$

Esercizio 4. Si dimostri che ogni insieme ricorsivamente enumerabile non ricorsivo (quindi non vuoto) has almeno un sottoinsieme ricorsivo infinito.

Svolgimento. Sia f la funzione calcolabile totale che enumera l'insieme r.e. dato. A partire da f si costruisca la seguente funzione:

$$\begin{cases} g(0) & = & f(0) \\ g(n+1) & = & f(\mu\, k[f(k) > g(n)]) \end{cases}$$

L'immagine di g è un insieme infinito. Inoltre tale funzione è *strettamente crescente*, e quindi, per l'esercizio 3, la sua immagine è un insieme ricorsivo.

Esercizio 5. Si definisca l'insieme

$$W_i = \{n \mid \varphi_i(n) \downarrow\} = dom(\varphi_i)$$

e si considerino due funzioni calcolabili φ_i e φ_j tali che $\emptyset \neq W_i \subsetneq W_j$. Si discuta dell'esistenza di una funzione calcolabile totale f tale che per ogni x vale

$$W_{f(x)} = \begin{cases} W_i & \text{se } \varphi_x(x) \uparrow \text{, cioè se } x \notin K \\ W_j & \text{se } \varphi_x(x) \downarrow \text{, cioè se } x \in K \end{cases}$$

Svolgimento. Contrariamente a quanto suggerisce l'intuizione, una tale funzione calcolabile totale esiste. Sappiamo che W_i è r.e. e non vuoto, dunque esiste una funzione calcolabile totale h_i tale che $W_i = Imm(h_i)$, e similmente esiste h_j tale che $W_j = Imm(h_j)$. La seguente procedura genera gli (infiniti) elementi di $W_{f(x)}$ e poiché è definita per ogni x, la f è totale:

```
n:=0;
W_{f(x)} := ∅;
while (φ_x(x) non converge in n passi) do
    W_{f(x)} := W_{f(x)} ∪ h_i(n);
    n:=n+1;
n:=0;
while true do
    W_{f(x)} := W_{f(x)} ∪ h_j(n);
    n:=n+1
```

A parole: essendo $W_i \subsetneq W_j$, posso iniziare a costruire $W_{f(x)}$ generando alcuni elementi di W_i con la funzione h_i e inserendoli in $W_{f(x)}$; non appena mi accorgo che la $\varphi_x(x)$ termina inizio ad emettere gli elementi di W_j (alcuni magari già calcolati) in $W_{f(x)}$ con la funzione h_j.

Si noti che se W_i e W_j fossero disgiunti, la funzione f sarebbe la caratteristica di K e quindi *non* sarebbe calcolabile totale.

Esercizio 6 (William E. Dowling). Si consideri il contesto in cui un programma P viene eseguito sul dato x all'interno di un *ambiente*, creato e gestito da un sistema operativo OS fissato. Un programma è detto *virus* se altera il codice di OS durante la sua esecuzione, ed è ragionevole immaginare che esistano virus per OS.[3]

Diciamo che il programma P lanciato sul dato x, all'interno dell'ambiente generato da OS, *diffonde un virus* se altera OS. Altrimenti diciamo che P è *sicuro sul dato* x ed è *sicuro* se lo è su ogni dato.

Fissato il sistema operativo OS, esiste un programma IS-SAFE che, presi in input un qualsiasi programma P e un dato x, decide se P è sicuro su x?

Svolgimento.
Il programma IS-SAFE con le caratteristiche indicate non esiste. La dimostrazione procede per assurdo. Supporre che il programma

$$\text{IS-SAFE}(P,x) = \begin{cases} \text{SI} & \text{se } P \text{ è sicuro su } x \\ \text{NO} & \text{altrimenti} \end{cases}$$

sia sicuro porta ad un assurdo. Si consideri il programma

$$D(P) = \begin{cases} \text{CIAO} & \text{se IS-SAFE}(P,P)\text{=NO} \\ \text{altera OS} & \text{altrimenti} \end{cases}$$

e si valuti $D(D)$. Se D fosse sicuro sul dato D allora non può essere stata scelta l'opzione "altrimenti" nella definizione di D, dunque IS-SAFE(D,D)=NO, e dunque una contraddizione. Se D alterasse OS allora o IS-SAFE(D,D)=SI, e

[3] Questa definizione di virus si concentra sull'interazione fra un programma ed il sistema operativo ignorando la possibile interazione fra diversi programmi.

dunque una contraddizione, o IS-SAFE(D,D)=NO, e quindi l'unica azione compiuta da D è stata la scrittura di CIAO, operazione innocua, e dunque una contraddizione. Di conseguenza D non può essere calcolabile e sicuro.

Esercizio 7. È decidibile se $\forall i, j.\ \varphi_i = \varphi_j$?

Svolgimento. La risposta è negativa. Infatti, sia per assurdo calcolabile la seguente funzione

$$g(i,j) = \begin{cases} 1 & \text{se } \varphi_i = \varphi_j \\ 0 & \text{altrimenti} \end{cases}$$

Inoltre sia

$$\psi(i,j) = \begin{cases} 1 & \text{se } i \in K \\ \text{indefinita} & \text{altrimenti} \end{cases}$$

La ψ è intuitivamente calcolabile (perché?) dunque per la tesi di Church-Turing esiste k per cui $\psi = \varphi_k$. Possiamo quindi applicare il teorema del parametro

$$\psi(i,j) = \varphi_k(i,j) = \varphi_{s(k,i)}(j).$$

Poiché tra le vari macchine che calcolano la ψ ne possiamo scegliere una, prendiamo la k-esima e scriviamo $f(i) = \lambda i.s(k,i)$, ottenendo

$$\psi(i,j) = \varphi_k(i,j) = \varphi_{s(k,i)}(j) = \varphi_{f(i)}(j).$$

(Questo modo di individuare l'indice $f(i)$ è piuttosto comune e in seguito non verrà descritto con questo livello di dettaglio, e scriveremo solo $\psi(i,j) = \varphi_{f(i)}(j)$.)
Ora sia n_0 uno degli indici della macchina che restituisce sempre 1, cioè

$$\varphi_{n_0} = \lambda x.1$$

Definiamo adesso

$$g(f(i), n_0) = \begin{cases} 1 & \text{se } \varphi_{f(i)} = \varphi_{n_0} \Rightarrow \psi(f(i), n_0) = 1 \Rightarrow i \in K \\ 0 & \text{se } \varphi_{f(i)} \neq \varphi_{n_0} \Rightarrow \psi(f(i), n_0)\uparrow \Rightarrow i \notin K \end{cases}$$

la quale, poiché $f(x)$ è calcolabile totale per il teorema del parametro, deciderebbe il problema della fermata: assurdo.

NB. Non posso dire che $\{(i,j) \mid \varphi_i \neq \varphi_j\}$ è r.e. mandando le due macchine a coda di rondine in parallelo (p.e. se $\varphi_i = \lambda x.\ 1$ e $\varphi_j = \lambda x.\mathit{indefinita}$ mai potrò concludere che $\varphi_i \neq \varphi_j$).

Condideriamo adesso l'uguaglianza debole: $\varphi_i \sim \varphi_j$ se e solamente se $\forall x \in \mathit{dom}(\varphi_i) \cap \mathit{dom}(\varphi_j).\ \varphi_i(x) = \varphi_j(x)$ Di nuovo $\varphi_i \sim \varphi_j$ non è r.e. mentre il suo complemento è r.e. ($\sim$ non è una relazione di equivalenza: è riflessiva e simmetrica, ma non transitiva).

Esercizio 8. Si dimostri che CONST $= \{x \mid \varphi_x$ totale e costante$\}$ non è ricorsivo.

Svolgimento. Basta ridurre K a CONST. Per farlo, definiamo la funzione:

$$\psi(x,y) = \begin{cases} 1 & \text{se } \exists z > y \text{ tale che } \varphi_x(x)\downarrow \text{ in meno di z passi} \\ \text{indefinita} & \text{altrimenti} \end{cases}$$

che è calcolabile parziale, essendo la funzione semi-caratteristica di K.

Allora, per il teorema s-m-n esiste f calcolabile totale iniettiva tale che $\varphi_{f(x)}(y) = \psi(x,y)$ (anche qui usiamo il solito trucchetto di scegliere uno degli indici che calcolano la ψ per poi ignorarlo). Adesso

$$x \in K \Rightarrow \varphi_{f(x)} = \psi(x,y) = \lambda y.\, 1 \Rightarrow f(x) \in CONST$$

$$x \notin K \Rightarrow \varphi_{f(x)} = \lambda y.\text{ indefinito} \Rightarrow f(x) \notin CONST.$$

Quindi CONST non è r. (e nemmeno r.e.).

In alternativa si verifichi che CONST è un i.i.r.f. e che non è vuoto. Di conseguenza non è r.

Esercizio 9. Si dimostri che
INF $= \{x \mid \text{dominio}(\varphi_x)$ è infinito$\} \leq_{rec}$ CONST $= \{x \mid \varphi_x$ totale e costante$\}$

Svolgimento. Definiamo la funzione:

$$\psi(x,y) = \begin{cases} 1 & \text{se } \exists z > y \text{ tale che } \varphi_x(z)\downarrow \\ \text{indefinita} & \text{altrimenti} \end{cases}$$

che è calcolabile parziale: basta usare opportunamente la tecnica della coda di colomba su φ_x. Allora, per il teorema s-m-n esiste f calcolabile totale iniettiva tale che $\varphi_{f(x)}(y) = \psi(x,y)$ (anche qui usiamo il solito trucchetto di scegliere uno degli indici che calcolano la ψ per poi ignorarlo). Adesso

$$x \in INF \Rightarrow \varphi_{f(x)} = \psi(x,y) = \lambda y.\, 1 \Rightarrow f(x) \in CONST$$

$$x \notin INF \Rightarrow \exists y.\, \varphi_{f(x)}(y) \text{ è indefinito} \Rightarrow f(x) \notin CONST.$$

Esercizio 10. Si dimostri che
$\textsc{tot} = \{x \mid \text{dominio}(\varphi_x) = I\!N\} \leq_{rec} \textsc{inf} = \{x \mid \text{dominio}(\varphi_x) \text{ è infinito}\}$

Svolgimento. Definiamo la seguente funzione, dove abbiamo applicato il teorema *s-m-n*, dato che la ψ è calcolabile parziale (basta usare opportunamente la tecnica della coda di colomba su φ_x su un numero *finito* di argomenti):

$$\varphi_{f(x)}(y) = \psi(x,y) = \begin{cases} 1 & \text{se } \forall z < y.\ \varphi_x(z)\downarrow \\ \text{indefinita} & \text{altrimenti} \end{cases}$$

Adesso

$$x \in TOT \Rightarrow \varphi_{f(x)} = \psi(x,y) = \lambda y.\ 1 \Rightarrow f(x) \in INF$$

$$x \notin TOT \Rightarrow \exists \overline{y}. \forall y > \overline{y}.\ \varphi_{f(x)}(y) \text{ è indefinita} \Rightarrow f(x) \notin INF.$$

Si noti che la funzione identità non riduce $\textsc{tot}$ a $\textsc{inf}$: si consideri la funzione *mezzo* che non è totale ma ha dominio infinito.

Esercizio 11. Si dimostri che

1. $\textsc{fin} = \{x \mid dom\varphi_x \text{ è finito}\}$, cioè l'insieme degli indici delle funzioni calcolabili con dominio finito, non è ricorsivamente enumerabile.
2. $\textsc{inf} = \{x \mid dom\varphi_x \text{ è infinito}\} = \overline{\textsc{fin}}$ non è ricorsivamente enumerabile.

Svolgimento. Per (1) dimostriamo che $\overline{K} \leq_{rec} \textsc{fin}$, da cui la tesi. Definiamo

$$\psi(x,y) = \begin{cases} 1 & \text{se } x \in K \\ \text{indefinita} & \text{altrimenti} \end{cases}$$

che è calcolabile parziale, quindi per il teorema *s-m-n* esiste f calcolabile totale iniettiva, definita con le solite avvertenze, tale che $\varphi_{f(x)}(y) = \psi(x,y)$. Si ha che

$$x \in \overline{K} \text{ sse } x \notin K \Rightarrow \varphi_{f(x)} = \lambda y.\ \text{indefinita} \Rightarrow dom(\varphi_{f(x)}) = \emptyset \Rightarrow f(x) \in \textsc{fin}$$

$$x \notin \overline{K} \text{ sse } x \in K \Rightarrow \varphi_{f(x)} = \lambda y.1 \Rightarrow dom(\varphi_{f(x)}) = I\!N \Rightarrow f(x) \notin \textsc{fin}.$$

Per (2) si consideri la funzione (ottenuta come la $\varphi_{f(x)}$ di sopra da un'opportuna ψ')

$$\varphi_{g(x)}(x,y) = \begin{cases} 1 & \text{se } \varphi_x(x) \text{ non converge in } y \text{ passi} \\ \text{indefinita} & \text{altrimenti} \end{cases}$$

Si ha che

$$x \in \overline{K} \Rightarrow \nexists y.\ \varphi_{g(x)}(x)\downarrow \Rightarrow \varphi_{g(x)} = \lambda y.1 \Rightarrow dom(\varphi_{g(x)}) \text{ è infinito} \Rightarrow g(x) \in \textsc{inf}$$

$$x \notin \overline{K} \text{ sse } x \in K \Rightarrow \varphi_{g(x)} = \lambda y.\ \text{indefinita} \Rightarrow dom(\varphi_{f(x)}) \text{ è finito} \Rightarrow f(x) \in \textsc{fin}.$$

Nota: è immediato vedere che $\emptyset \neq \textsc{fin} \neq I\!N$, da cui, per il lemma usato per dimostrare il teorema di Rice, segue che non è r.

Esercizio 12. L'insieme $I = \{i \mid dom(\varphi_i) = \{3\}\}$ è ricorsivo? ricorsivamente enumerabile? nemmeno ricorsivamente enumerabile? e se $dom(\varphi_i)$ fosse l'insieme dei numeri pari?

Svolgimento Banalmente $\emptyset \neq I \neq I\!N$ e inoltre I è un insieme di indici che rispettano le funzioni, pertanto I non è r.

Inoltre $K \leq_{rec} I$. Infatti, si consideri

$$\varphi_{f(x)}(y) = \psi(x,y) = \begin{cases} 1 & \text{se } x \in K \ \wedge \ y = 3 \\ \text{indefinita} & \text{altrimenti} \end{cases}$$

che è calcolabile parziale. Adesso

$$x \in K \Rightarrow \varphi_{f(x)}(y) = \begin{cases} 1 & \text{se } y = 3 \\ \text{indefinita} & \text{se } y \neq 3 \end{cases} \Rightarrow dom(\varphi_{f(x)}) = \{3\} \Rightarrow f(x) \in I$$

$x \notin K \Rightarrow \varphi_{f(x)} = \lambda y.$ indefinito $\Rightarrow f(x) \notin I$.

Infine, possiamo dimostrare che I non è nemmeno r.e., perché $\overline{K} \leq_{rec} I$; infatti $K \leq_{rec} \overline{I}$, attraverso la seguente funzione di riduzione che modifica leggermente quella usata sopra:

$$\varphi_{g(x)}(y) = \psi(x,y) = \begin{cases} 1 & \text{se } x \in K \ \vee \ y = 3 \\ \text{indefinita} & \text{altrimenti} \end{cases}$$

Abbiamo che

$$x \in K \Rightarrow \varphi_{g(x)}(y) = 1 \Rightarrow dom(\varphi_{g(x)}) = I\!N \Rightarrow g(x) \notin I$$

$$x \notin K \Rightarrow \varphi_{g(x)}(y) = \begin{cases} 1 & \text{se } y = 3 \\ \text{indefinita} & \text{altrimenti} \end{cases} \Rightarrow dom(\varphi_{g(x)}) = \{3\}$$

$\Rightarrow g(x) \in I$

Esercizio 13. Esistono due insiemi A e B non ricorsivi la cui intersezione è un insieme ricorsivo infinito?

Svolgimento Sí: si consideri un insieme r. infinito, per esempio $P = \{2n\}$ e si pongano $A = K \cup P$ e $B = \bar{K} \cup P$.

Esercizio 14. La classe degli insiemi di indici che rappresentano le funzioni forma un'algebra booleana? Si giustifichi la risposta.

Svolgimento Basta far vedere, ed è banale, che

- $\emptyset$ è un i.i.r.f. (nessuna)
- se I è un i.i.r.f. lo è anche il suo complemento
- se I e J sono due i.i.r.f., lo è anche $I \cap J$

Esercizio 15. Si dica se esiste una funzione calcolabile totale f tale che

$$\forall x.\, dom(\varphi_{f(x)}) = I\!N$$

Svolgimento Sia i uno degli indici della funzione $\lambda\, x.\, 0$ e si ponga $f = \lambda\, x.\, i$. Chiaramente $dom(\varphi_{f(x)}) = dom(\varphi_i) = I\!N$.

Esercizio 16. Si dimostri che per ogni enumerazione effettiva esiste $i \in K$ tale che

$$\varphi_i(x) = \begin{cases} 0 & \text{se } x = i \\ \text{indef} & \text{altrimenti} \end{cases}$$

Svolgimento Si definisca la seguente funzione $\psi(x, y)$ che è intuitivamente calcolabile, quindi possiamo applicare il teorema *s-n-m* al suo indice e a x, e poi ottenere con il solito metodo, la seguente funzione calcolabile totale f:

$$\varphi_{f(x)} = \psi(x, y) = \begin{cases} 0 & \text{se } x = y \\ \text{indef} & \text{altrimenti} \end{cases}$$

Per il teorema di ricorsione, $\exists n$ tale che $\varphi_n = \varphi_{f(n)}$ e quindi $\forall y$ si ha che

$$\varphi_n(y) = \varphi_{f(n)}(y) = \psi(n, y) = \begin{cases} 0 & \text{se } y = n \\ \text{indef} & \text{altrimenti} \end{cases}$$

Quindi esiste una funzione calcolabile che converge solamente sul suo indice, indipendentemente dall'enumerazione scelta.

Esercizio 17. Sia A_i l'insieme di indici costruito nel padding lemma e si dimostri che $A_i \leq_{rec} K$

Svolgimento. L'insieme A_i è generato da una funzione ricorsiva primitiva, per cui è ricorsivamente enumerabile. La tesi segue dalla completezza di K.

Esercizio 18. Dato i, si dica se l'insieme $K@i = \{j \mid j \in K \land j \leq i\}$ è r., r.e. o non r.e. e si giustifichi la risposta.

Svolgimento. Poiché $K \downarrow i$ è finito è ricorsivo.

Esercizio 19. Si dimostri che $A = \{x \mid \forall y.\, \varphi_x(y) \uparrow\}$ non è r.e.

Svolgimento Per farlo basta dimostrare che $\overline{K} \leq A$ ovvero che $K \leq \overline{A} = K_1$. Si definisca allora la seguente funzione $\psi(x, y)$ che è intuitivamente calcolabile, quindi possiamo applicare il teorema del parametro al suo indice e a x, e poi ottenere con il solito metodo, la seguente funzione calcolabile totale f:

$$\varphi_{f(x)} = \psi(x, y) = \begin{cases} 1 & \text{se } x \in K \wedge \exists z.\varphi_x(z) \downarrow \\ \text{indef} & \text{altrimenti} \end{cases}$$

Adesso abbiamo che

$$x \in K \implies \varphi_{f(x)} = \lambda y.1 \implies f(x) \in \overline{A} \quad \text{(basta prendere } z = x\text{)}$$

$$x \notin K \implies \varphi_{f(x)} = \lambda y.\text{indef} \implies f(x) \in A$$

Esercizio 20. Un insieme di indici che rappresenta una singola funzione può essere sempre costruito usando il padding lemma?

Svolgimento. No, perché ogni insieme di indici A_x costruito come nel padding lemma a partire dall'indice x è r.e., in quanto immagine di una funzione primitiva ricorsiva (e quindi calcolabile totale) mentre ci sono insiemi di indici che rappresentano le funzioni, p.e. quello che rappresenta la funzione ovunque indefinita, che non sono r.e. (v. esercizio 19).