Note del corso di Calcolabilità e Linguaggi Formali - Lezione 7

Alberto Carraro

DAIS, Universitá Ca' Foscari Venezia http://www.dsi.unive.it/~acarraro

1 Introduzione

Nelle precedenti lezioni sono stati introdotti vari modelli astratti di automi che implementano dispositivi di calcolo. Per le Macchine di Turing inoltre sono state definite codifiche e "protocolli" per il loro utilizzo allo scopo di calcolare funzioni sui numeri naturali. In questa lezione faremo una cosa diversa: descriveremo una classe di funzioni sui numeri naturali in termini puramente matematici, senza riferimento ad alcun modello di macchina di calcolo e solo successivamente scopriremo che tali funzioni sono esattamente quelle calcolabili dalle Macchine di Turing.

2 Le funzioni primitive ricorsive

Un tipo di funzioni molto importanti sono le cosiddette *funzioni caratteristiche*, che sono funzioni a valori in $\{0,1\}$ da intendersi come valori booleani che indicano l'appartenenza (o non-appartenenza) di n-uple di numeri a sottoinsiemi di $\mathbb{N}^n$.

Definition 1 (Funzione caratteristica). *Sia $R \subseteq \mathbb{N}^n$ una relazione n-aria. La* funzione caratteristica *di R, indicata con $\mathsf{c}_R : \mathbb{N}^n \to \mathbb{N}$, è definita come segue*

$$\mathsf{c}_R(\vec{x}) = \begin{cases} 1 & \text{se } R(\vec{x}) \\ 0 & \text{altrimenti} \end{cases}$$

Definition 2 (Funzioni di base). *Chiamiamo* funzioni di base *quelle qui di seguito elencate:*

(1) la funzione $\mathsf{c}_=$;
(2) la funzione zero *$\mathsf{Z} : \mathbb{N} \to \mathbb{N}$ data da $\mathsf{Z}(x) = 0$, per ogni $x \in \mathbb{N}$;*
(3) la funzione successore *$\mathsf{S} : \mathbb{N} \to \mathbb{N}$ data da $\mathsf{S}(x) = x + 1$, per ogni $x \in \mathbb{N}$;*
(4) per ogni coppia di numeri j, n con $1 \leq j \leq n$ la funzione proiezione *j-esima $I^n_j : \mathbb{N}^n \to \mathbb{N}$ data da $I^n_j(x_1, \ldots, x_n) = x_j$, per ogni n-upla $(x_1, \ldots, x_n) \in \mathbb{N}^n$.*

Definition 3 (Funzioni primitive ricorsive). *L'insieme* **PrimREC** *delle* funzioni ricorsive primitive *è il più piccolo insieme di funzioni sui numeri naturali che soddisfa le seguenti proprietà:*

(**r1**) *Contiene tutte le funzioni di base della Definizione* **??**.

(**r2**) *È chiuso per composizione, ovvero se contiene* $\psi, \gamma_1, \ldots, \gamma_m$ *allora contiene anche la funzione* φ *data da* $\varphi(\vec{x}) = \psi(\gamma_1(\vec{x}), \ldots, \gamma_m(\vec{x}))$.

(**r3**) *È chiuso per ricorsione primitiva, ovvero se contiene* ψ, γ *allora contiene anche la funzione* φ *data da*

$$\varphi(\vec{x}, y) = \begin{cases} \psi(\vec{x}) & \text{se } y = 0 \\ \gamma(\vec{x}, y-1, \varphi(\vec{x}, y-1)) & \text{altrimenti} \end{cases}$$

Le proprietà di chiusura (**r2**) ed (**r3**) della Definizione **??**, sono detti, rispettivamente, *schema di composizione* e *schema di ricorsione primitiva*. Infatti, avendo definito **PrimREC** come il più piccolo insieme di funzioni chiuse rispetto alle costruzioni elencate, esse rappresentano di fatto i soli schemi di definizione, da applicare un numero finito di volte alle funzioni di base, per creare nuove funzioni primitive ricorsive a partire da altre esistenti. In altre parole l'insieme **PrimREC** è definito per induzione, e si possono pertanto fare dimostrazioni e costruzioni per induzione sulla definizione delle funzioni che vi appartengono.

2.1 Schemi alternativi

Grazie alle funzioni di proiezione e alle proprietà elementari dell'aritmetica, si possono creare nuovi e più flessibili schemi derivati da quello di composizione e di ricorsione primitiva che rendono più comode le definizioni, pur non aggiungendo nuove funzioni a **PrimREC**.

Lemma 1. *(i) La funzione identità è in* **PrimREC**.

(ii) Sia $\psi : \mathbb{N}^m \to \mathbb{N}$ *una funzione in* **PrimREC**, *sia* $n \geq m$ *e siano* $j_1, \ldots, j_m$ *indici presi in* $\{1, \ldots, n\}$. *Allora la funzione* $\varphi : \mathbb{N}^n \to \mathbb{N}$ *data da* $\varphi(x_1, \ldots, x_n) = \psi(x_{j_1}, \ldots, x_{j_m})$ *è anch'essa in* **PrimREC**.

(iii) Ogni funzione costante è in **PrimREC**.

(iv) Se ψ, γ *sono in* **PrimREC**, *anche la funzione* φ *data da*

$$\varphi(\vec{x}, y) = \begin{cases} \psi(\vec{x}) & \text{se } y = 0 \\ \gamma(\varphi(\vec{x}, y-1)) & \text{altrimenti} \end{cases}$$

è in **PrimREC**.

(v) Se ψ, γ *sono in* **PrimREC**, *anche la funzione* φ *data da*

$$\varphi(\vec{x}, y) = \begin{cases} \psi(\vec{x}) & \text{se } y = 0 \\ \gamma(\vec{x}, \varphi(\vec{x}, y-1)) & \text{altrimenti} \end{cases}$$

è in **PrimREC**.

(vi) La funzione predecessore è in **PrimREC**.

Proof. (i) L'identità è la funzione $I^1_1(x)$.

(ii) Chiaramente $\varphi(\vec{x}) = \psi(I^n_{j_1}(\vec{x}), \ldots, I^n_{j_m}(\vec{x}))$, quindi è in **PrimREC** per lo schema di composizione.

(iii) Sia ψ data da $\psi(\vec{x}) = n$, per un certo $n \in \mathbb{N}$. Allora

$$\psi(\vec{x}) = \underbrace{\mathsf{S}(\cdots\mathsf{S}(}_{n \text{ volte}}\mathsf{Z}(x_1))\cdots)$$

quindi $\psi \in \mathbf{PrimREC}$ per il punto (2) e lo schema di composizione.

(iv) Si ponga $\gamma'(\vec{x}, y, z) = \gamma(z)$. Allora γ' è in **PrimREC** per il punto (2). Ora la funzione φ è data da

$$\varphi(\vec{x}, y) = \begin{cases} \psi(\vec{x}) & \text{se } y = 0 \\ \gamma'(\vec{x}, y-1, \varphi(\vec{x}, y-1)) & \text{altrimenti} \end{cases}$$

e quindi, per lo schema di ricorsione primitiva, è in **PrimREC**.

(v) Simile al punto precedente.

(vi) La funzione predecessore sui numeri naturali è data da

$$\mathsf{P}(y) = \begin{cases} 0 & \text{se } y = 0 \\ I_1^2(y-1, \mathsf{P}(y-1)) & \text{altrimenti} \end{cases}$$

e quindi è in **PrimREC** per lo schema di ricorsione primitiva.

□

Notiamo che se $\varphi(\vec{x}, y) : \mathbb{N}^{n+1} \to \mathbb{N}$ è una funzione ricorsiva, allora per ogni numero naturale k fissato la funzione $\varphi(\vec{x}, k) : \mathbb{N}^n \to \mathbb{N}$ è una funzione ricorsiva, per via dello schema di composizione.

Facciamo seguire alcuni esempi di funzioni a noi familiari che sono primitive ricorsive. Per ciascuna di esse cercate di capire quali schemi sono stati utilizzati nella definizione.

$$x \dot{-} y = \begin{cases} x & \text{se } y = 0 \\ \mathsf{P}(x \dot{-} (y-1)) & \text{altrimenti} \end{cases} \qquad x + y = \begin{cases} x & \text{se } y = 0 \\ \mathsf{S}(x + (y-1)) & \text{altrimenti} \end{cases}$$

$$x \cdot y = \begin{cases} 0 & \text{se } y = 0 \\ x + (x \cdot (y-1)) & \text{altrimenti} \end{cases} \qquad x^y = \begin{cases} 1 & \text{se } y = 0 \\ x \cdot (x^{y-1}) & \text{altrimenti} \end{cases}$$

$$!y = \begin{cases} 1 & \text{se } y = 0 \\ y \cdot !(y-1) & \text{altrimenti} \end{cases}$$

Chiaramente, utilizzando il solo schema di rimpiazzamento, otteniamo che se $\psi_1(\vec{x}), \ldots, \psi_k(\vec{x})$ sono funzioni primitive ricorsive, anche le funzioni $\varphi(\vec{x}) = \sum_{i=1}^{k} \psi_i(\vec{x})$ e $\gamma(\vec{x}) = \prod_{i=1}^{k} \psi_i(\vec{x})$ sono primitive ricorsive. Altre importantissime funzioni primitive ricorsive sono le cosiddette *somme e prodotti limitati*, che generalizzano il caso delle somme appena visto, in quanto si vuole permettere al numero degli addendi di variare in funzione dell'argomento passato alla funzione. Data una funzione primitiva ricorsiva $n+1$-aria $\varphi(\vec{x}, y)$, le funzioni $\sum_{y<z} \varphi(\vec{x}, y)$ e $\prod_{y<z} \varphi(\vec{x}, y)$ (anch'esse da $\mathbb{N}^{n+1}$ in $\mathbb{N}$) sono definite come segue:

$$\sum_{y<z} \varphi(\vec{x}, y) = \begin{cases} 0 & \text{se } z = 0 \\ \varphi(\vec{x}, z) + \sum_{y<z-1} \varphi(\vec{x}, y) & \text{altrimenti} \end{cases}$$

$$\prod_{y<z} \varphi(\vec{x}, y) = \begin{cases} 1 & \text{se } z = 0 \\ \varphi(\vec{x}, z) \cdot \prod_{y<z-1} \varphi(\vec{x}, y) & \text{altrimenti} \end{cases}$$

Inoltre per lo schema di rimpiazzamento anche le funzioni $\sum_{y<\psi(\vec{x},z)} \varphi(\vec{x}, y)$ e $\prod_{y<\psi(\vec{x},z)} \varphi(\vec{x}, y)$ sono primitive ricorsive, qualora $\psi(\vec{x}, z)$ e $\varphi(\vec{x}, y)$ lo siano.

3 Insiemi e predicati primitivi ricorsivi

Definition 4. *Sia $R \subseteq \mathbb{N}^n$ una relazione n-aria. Diciamo che R è* primitiva ricorsiva *se la sua funzione caratteristica c_R è una funzione primitiva ricorsiva.*

Ad esempio l'uguglianza è una relazione primitiva ricorsiva. Vediamo come si comportano i predicati primitivi ricorsivi rispetto alle operazioni logiche.

Lemma 2. *Siano $R, P \subseteq \mathbb{N}^n$ due predicati primitivi ricorsivi. Allora $P \vee R$, $P \wedge R$, $P \Rightarrow R$ e $\neg P$ sono tutti predicati primitivi ricorsivi.*

Proof. Abbiamo che $\mathsf{c}_{P\wedge R}(\vec{x}) = \mathsf{c}_P(\vec{x}) \cdot \mathsf{c}_P(\vec{x})$ e $\mathsf{c}_{\neg P}(\vec{x}) = 1 - \mathsf{c}_P(\vec{x})$. Infine $\mathsf{c}_{P\vee R}(\vec{x}) = \mathsf{c}_{\neg(\neg P\wedge\neg R)}(\vec{x})$ e $\mathsf{c}_{P\Rightarrow R}(\vec{x}) = \mathsf{c}_{\neg P\vee R}(\vec{x})$. □

Lemma 3. *Le relazioni $<, >, \leq, \geq, \neq$ sono primitive ricorsive.*

Proof. Abbiamo che $x < y \Leftrightarrow (x \dot{-} y = 0) \wedge (y \dot{-} x \neq 0)$. Quindi $\mathsf{c}_<(x, y)$ è una funzione primitiva ricorsiva dato che $\mathsf{c}_<(x, y) = \mathsf{c}_=(x \dot{-} y, 0) \cdot (1 - \mathsf{c}_=(y \dot{-} x, 0))$. Usando le proprietà degli operatori logici ora si possono definire le funzioni caratteristiche delle altre relazioni. □

Lemma 4. *Sia $\varphi(\vec{x})$ in* **PrimREC**. *Allora il predicato $P(\vec{x}, y) := \varphi(\vec{x}) = y$ (oppure $<, >$, ecc...) è primitivo ricorsivo.*

Proof. Immediato, poiché $\mathsf{c}_P(\vec{x}, y) = \mathsf{c}_=(\varphi(\vec{x}), y)$ che è in **PrimREC** per lo schema di rimpiazzamento. □

Lemma 5 (Schema di definizione per casi). *Siano $\psi_1, \ldots, \psi_k$ funzioni primitive ricorsive e siano $R_1, \ldots, R_k$ predicati primitivi ricorsivi n-ari a due a due disgiunti e tali che $\bigcup_{i=1}^{k} R_i = \mathbb{N}^k$. Allora la funzione φ definita dallo schema*

$$\varphi(\vec{x}) = \begin{cases} \psi_1(\vec{x}) & \text{se } R_1(\vec{x}) \\ \vdots & \vdots \\ \psi_k(\vec{x}) & \text{se } R_k(\vec{x}) \end{cases}$$

è primitiva ricorsiva.

Proof. Abbiamo $\varphi(\vec{x}) = \sum_{i=1}^{k} \mathsf{c}_{R_i}(\vec{x}) \cdot \psi_i(\vec{x})$. □

Come conseguenza dei Lemmi **??** e **??** la funzione *massimo* tra due numeri è primitiva ricorsiva poiché abbiamo

$$max(x,y) = \begin{cases} x & \text{se } x \geq y \\ y & \text{se } x < y \end{cases}$$

Quindi anche il massimo di una lista di valori di funzione primitiva ricorsiva φ è una funzione primitiva ricorsiva in quanto

$$max_{y<z}\varphi(\vec{x},y) = \begin{cases} 0 & \text{se } z = 0 \\ max(\varphi(\vec{x},z), max_{y<z-1}\varphi(\vec{x},y)) & \text{altrimenti} \end{cases}$$

Con un ragionamento analogo si dimostra che la funzione $min_{y<z}\varphi(\vec{x},y)$ è anch'essa primitiva ricorsiva se φ lo è.

Vediamo come si comportano i predicati primitivi ricorsivi rispetto alla quantificazione. Anticipando che essi non sono chiusi rispetto a quantificazioni universali ed esistenziali arbitrarie, ci interessiamo ad una particolare forma di quantificazione, detta *quantificazione limitata*. Sia $P(\vec{x},y)$ una relazione $n+1$-aria. Definiamo due relazioni $n+1$-arie

$$\forall y{<}z.P(\vec{x},y) := \forall y.(y \geq z \vee P(\vec{x},y)) \qquad \exists y{<}z.P(\vec{x},y) := \exists y.(y < z \wedge P(\vec{x},y))$$

Lemma 6 (Quantificazione limitata). *Siano $R \subseteq \mathbb{N}^{n+1}$ un predicato primitivo ricorsivo. Allora $\forall y < z.P(\vec{x},y)$ e $\exists y < z.P(\vec{x},y)$ sono predicati primitivi ricorsivi.*

Proof. Siano $R := \forall y{<}z.P(\vec{x},y)$ e $Q := \exists y{<}z.P(\vec{x},y)$. Abbiamo che $\mathsf{c}_R(\vec{x},z) = min_{y<z}\mathsf{c}_P(\vec{x},z)$ e similmente $\mathsf{c}_Q(\vec{x},z) = max_{y<z}\mathsf{c}_Q(\vec{x},y)$. □

Non è difficile immaginare come estendere il Lemma **??** al caso di predicati come $\forall y{\leq}z.P(\vec{x},y)$ e $\forall y{<}\psi(\vec{x},y).P(\vec{x},y)$ (con $\psi(\vec{x},y)$ primitiva ricorsiva).

Definition 5 (μ-ricorsione limitata). *Sia $P(\vec{x},y)$ un predicato $n+1$-ario primitivo ricorsivo. Allora definiamo la funzione $\mu y < z.P(\vec{x},y) : \mathbb{N}^{n+1} \to \mathbb{N}$ come segue*

$$\mu y{<}z.P(\vec{x},y) = \begin{cases} \textit{il minimo } y \textit{ tale che } P(\vec{x},y) & \textit{se } \exists y{<}z.P(\vec{x},y) \\ z & \textit{altrimenti} \end{cases}$$

Lemma 7 (Schema di definizione per μ-ricorsione limitata). *Sia R un predicato primitivo ricorsivo $n+1$-ario e sia $\psi(\vec{x},z)$ una funzione primitiva ricorsiva $n+1$-aria. Allora*

(i) la funzione φ definita dallo schema $\varphi(\vec{x},z) = \mu y < z.P(\vec{x},y)$ è primitiva ricorsiva.

(ii) la funzione φ definita dallo schema $\varphi(\vec{x}, z) = \mu y{<}\psi(\vec{x}, z).P(\vec{x}, y)$ è primitiva ricorsiva.

Proof. (i) Per i Lemmi **??** e **??** il predicato $R(\vec{x}, y) := P(\vec{x}, y) \wedge (\forall z{<}y.\neg P(\vec{x}, z))$ è primitivo ricorsivo. Infine osserviamo che $\varphi(\vec{x}, z) = \sum_{y<z+1} y \cdot \mathsf{c}_R(\vec{x}, y)$.
(ii) Semplicemente usando il punto (i) e lo schema di rimpiazzamento.

□

Analizziamo ora un importante caso particolare del Lemma **??**.

Lemma 8 (Schema di definizione per μ-ricorsione limitata autorefente). *Siano $\gamma(\vec{x}, z, y)$ e $\delta(\vec{x}, z)$ due funzioni primitive ricorsive. Allora*

(i) la funzione φ definita dallo schema $\varphi(\vec{x}, z) = \mu y{<}z.[\gamma(\vec{x}, \varphi(\vec{x}, z-1), y) = 0]$ è primitiva ricorsiva.
(ii) la funzione φ definita dallo schema $\varphi(\vec{x}, z) = \mu y{<}\delta(\vec{x}, \varphi(\vec{x}, z-1)).[\gamma(\vec{x}, \varphi(\vec{x}, z-1), y) = 0]$ è primitiva ricorsiva.

Proof. (i) Sia $R_\gamma(\vec{x}, z, y) := (\gamma(\vec{x}, z, y) = 0 \wedge (\forall z{<}y.\gamma(\vec{x}, z, y) \neq 0))$. La funzione $\mathsf{c}_R(\vec{x}, z, y)$ è primitiva ricorsiva. Infine osserviamo che

$$\varphi(\vec{x}, z) = \begin{cases} 0 & \text{se } z = 0 \\ \sum_{y<z} y \cdot \mathsf{c}_{R_\gamma}(\vec{x}, \varphi(\vec{x}, z-1), y) & \text{altrimenti} \end{cases}$$

e quindi $\varphi(\vec{x}, z) \in \mathbf{PrimREC}$ per lo schema di ricorsione primitiva e di rimpiazzamento.
(ii) Si tratta di una semplice estensione del punto (i).

□

In sostanza il Lemma **??** dice che quando definiamo una funzione per μ-ricorsione limitata, sia il predicato di controllo che il bound possono essere dati in funzione del valore di $\varphi(\vec{x}, z-1)$.

Prendiamo ad esempio la definizione $\varphi(\vec{x}, z) = \mu y{<}z+1.[y > \varphi(\vec{x}, z-1)]$. Tale funzione si può definire come segue:

$$\varphi(\vec{x}, z) = \begin{cases} 0 & \text{se } z = 0 \\ z+1 & \text{altrimenti} \end{cases}$$

3.1 Codaggio delle sequenze e lo schema "course-of-values"

A questo punto ci si può domandare se una funzione famosa come quella di Fibonacci, data da

$$F(x) = \begin{cases} 0 & \text{se } x = 0 \\ 1 & \text{se } x = 1 \\ F(x-1) + F(x-2) & \text{se } x \geq 2 \end{cases}$$

sia o meno primitiva ricorsiva. A prima vista diremmo che non vi è uno schema che permette di definirla poichè nella sua definizione induttiva il calcolo può richiedere più di un valore della funzione stessa su argomenti più piccoli. Tuttavia stiamo per vedere che la funzione di Fibonacci è primitiva ricorsiva, ma per dimostrarlo abbiamo bisogno di un meccanismo di codifica (primitivo ricorsivo!) per codificare le sequenze di numeri naturali nei numeri naturali stessi.

Prima di procedere abbiamo bisogno di dimostrare che alcune funzioni sono primitive ricorsive. Ricordiamo che per il Teorema Fondamentale dell'Aritmetica ogni numero naturale n si scrive in maniera unica come prodotto di potenze di numeri primi: tale prodotto è chiamato *decomposizione prima* del numero n.

Lemma 9. *I seguenti predicati e funzioni sono primitivi ricorsivi:*

(i) il predicato $x \mid y :=$ *"x divide y"*
(ii) il predicato $\mathsf{prim}(x) :=$ *"x è primo"*
(iii) la funzione $\mathsf{p}(x) =$ *"l'x-esimo numero primo"*
(iv) la funzione $\mathsf{exp}(x, y) =$ *"il più grande numero k tale che* x^k *divide y"*

Proof. (i) Abbiamo $x \mid y \Leftrightarrow \exists z{<}y+1.x \cdot z = y$.
(ii) Abbiamo $\mathsf{prim}(x) \Leftrightarrow x \geq 2 \wedge \forall y{<}x+1.y \mid x \Rightarrow (y = 1 \vee y = x)$.
(iii) Osserviamo che, per la famosa dimostrazione di Euclide sull'infinità dei numeri primi, l'x-esimo numero primo è minore o uguale al prodotto dei primi $x-1$ numeri primi aumentato di uno. Pertanto abbiamo $\mathsf{p}(x) = \mu y{<}!\mathsf{p}(x-1)+2.[y > \mathsf{p}(x-1) \wedge \mathsf{prim}(y)]$.
(iv) Abbiamo $\mathsf{exp}(x, y) = \mu z{<}y+1.[x^z \mid y \wedge \neg(x^{z+1} \mid y)]$.

□

Utilizzando questi strumenti possiamo progettare funzioni primitive ricorsive di codifica e decodifica delle sequenze di numeri naturali.

Indicheremo nel seguito con p_i l'$i+1$-esimo numero primo (pertanto $p_0 = 2$). Definiamo la codifica della sequenza $(x_1, \ldots, x_n)$ come il numero $\prec x_1, \ldots, x_n \succ = p_0^n \cdot \prod_{i=1}^{n} p_i^{x_i}$. Per la decodifica, dato un numero x poniamo:

- $Seq(x) \Leftrightarrow \forall n \leq x.[(n > 0 \wedge (x)_n \neq 0) \Rightarrow n \leq (x)_0]$
- $(x)_n = \mathsf{exp}(p_n, x)$
- $ln(x) = (x)_0$

Dunque $Seq(x)$ è vero sse x è la codifica di una qualche sequenza; se $Seq(x)$ vale, allora $x = \prec (x)_1, \ldots, (x)_{ln(x)} \succ$. I numeri naturali n tali che $Seq(n)$ sono detti *numeri di sequenza*.

Si può anche definire l'operazione di *concatenazione di numeri di sequenza* come segue:

$$x * y = \begin{cases} p_0^{ln(x)+ln(y)} \cdot \prod_{i<ln(x)} p_{i+1}^{(x)_{i+1}} \cdot \prod_{j<ln(y)} p_{ln(x)+j+1}^{(y)_{j+1}} & \text{se } Seq(x) \wedge Seq(y) \\ 0 & \text{altrimenti} \end{cases}$$

Qiondi $x{*}y$ è il numero di sequenza della concatenazione delle sequenze di cui x ed y sono le codifiche. In altre parole $\langle x_1, \ldots, x_n \rangle * \langle y_1, \ldots, y_m \rangle = \langle x_1, \ldots, x_n, y_1, \ldots, y_m \rangle$.

Con il prossimo teorema, che fa uso della codifica delle sequenze, dimostriamo che la classe delle funzioni primitive ricorsive è chiusa per definizioni ricorsive in cui il calcolo di $f(\vec{x}, y+1)$ possa coinvolgere non solo il valore $f(\vec{x}, y)$, ma anche possibilmente tutti i valori nell'insieme finito $\{f(\vec{x}, z) : z < y\}$.

Allo scopo del teorema definiamo la *funzione storia* $\hat{f}$ di f come seque:

$$\hat{f}(\vec{x}, y) = \langle f(\vec{x}, 0), \ldots, f(\vec{x}, y)\rangle$$

Theorem 1 (Course-of-values recursion). *Siano g, h funzioni primitive ricorsive e sia f definita dallo schema:*

$$f(\vec{x}, 0) = g(\vec{x})$$
$$f(\vec{x}, y+1) = h(\vec{x}, y, \hat{f}(\vec{x}, y))$$

Allora f e $\hat{f}$ sono entrambe primitive ricorsive.

Proof. È sufficiente dimostrare che $\hat{f}$ è primitiva ricorsiva, poiché f lo sarà di conseguenza. Ora notiamo che

$$\begin{aligned}\hat{f}(\vec{x}, 0) &= \prec f(\vec{x}, 0)\succ \\ &= \prec g(\vec{x})\succ\end{aligned}$$

$$\begin{aligned}\hat{f}(\vec{x}, y+1) &= \prec f(\vec{x}, 0), \ldots, f(\vec{x}, y), f(\vec{x}, y+1)\succ \\ &= \hat{f}(\vec{x}, y) * \prec f(\vec{x}, y+1)\succ \\ &= \hat{f}(\vec{x}, y) * \prec h(\vec{x}, y, \hat{f}(\vec{x}, y))\succ\end{aligned}$$

Pertanto il calcolo di $\hat{f}(\vec{x}, y+1)$ utilizza solo il valore $\hat{f}(\vec{x}, y)$ e le funzioni primitive ricorsive $*$ e h. Quindi $\hat{f}$ è primitiva ricorsiva dato che è definita con lo schema [(**r3**). Infine f è primitiva ricorsiva perché $f(\vec{x}, y) = (\hat{f}(\vec{x}, y))_{y+1}$.

Come conseguenza del Teorema **??** la funzione di Fibonacci (e tante altre) è primitiva ricorsiva.

4 Le funzioni ricorsive totali e parziali

Un'altra famosa funzione è la *funzione di Ackermann*, definita come segue:

$$A(x, y) = \begin{cases} y+1 & \text{se } x = 0 \\ A(x-1, 1) & \text{se } x > 0 \text{ e } y = 0 \\ A(x-1, A(x, y-1)) & \text{se } x > 0 \text{ e } y > 0 \end{cases}$$

È abbastanza immediato intuire che si possa scrivere un programma per calcolatore che computi la funzione di Ackermann. Molto meno immediato è intuire che questa funzione non è primitiva ricorsiva, eppure ciò si può dimostrare. Questo

vuol dire che le funzioni ricorsive primitive non sono sufficienti a catturare tutte quelle calcolabili dalle macchine di Turing. Un'altra osservazione importante è che alcune funzioni calcolate da macchine di Turing sono *parziali*, ovvero vi sono degli input su cui esse non danno alcun risulato: questo perchè alcune macchine di Turing non si arrestano mai su certi input.

Per questi motivi andiamo ad esaminare una classe più ampia di funzioni, che riesca a comprendere ad esempio anche la funzione di Ackermann ed in generale anche le funzioni parziali calcolate dalle macchine di Turing. Per procedere su questa strada abbiamo bisogno di un nuovo fondamentale strumento, la *μ-ricorsione* che andiamo a definire qui di seguito.

Definition 6 (μ-ricorsione). *Sia $P(\vec{x}, y)$ un predicato (o sottoinsieme di $\mathbb{N}^{n+1}$). Allora definiamo la funzione $\mu y.P(\vec{x}, y) : \mathbb{N}^n \to \mathbb{N}$ come segue*

$$\mu y.P(\vec{x}, y) = \begin{cases} \textit{il minimo } y \textit{ tale che } P(\vec{x}, y) & \textit{se un tale } y \textit{ esiste} \\ \uparrow & \textit{altrimenti} \end{cases}$$

Appare immediatamente evidente che lo schema della Definizione **??** può produrre funzioni parziali. Tale tipo di μ-ricorsione è anche detta *illimitata* perchè esistono alcuni importanti casi paticolari della μ-ricorsione, ottenuti ponendo qualche vincolo, che producono esclusivamente funzioni totali:

- quando il predicato P è tale che $\forall \vec{x}.\exists y.P(\vec{x}, y)$, allora lo schema diventa la *μ-ricorsione totale*,
- quando il predicato P è della forma $P(\vec{x}, z, y) := P'(\vec{x}, y) \vee y = z$, allora lo schema diventa la *μ-ricorsione limitata*.

Nei due casi appena elencati lo schema di μ-ricorsione produce solo funzioni totali.

Gli schemi di μ-ricorsione qui sopra descritti non producono però necessariamente delle funzioni adatte ai nostri scopi. Difatti è necessario per noi che il predicato $P(\vec{x}, y)$ usato nella definizione di $\mu y.P(\vec{x}, y)$ abbia una particolare forma.

Definition 7 (Funzioni ricorsive parziali). *L'insieme* **PR** *delle* funzioni ricorsive parziali *è il più piccolo insieme di funzioni sui numeri naturali che soddisfa le proprietà (***r1***)-(***r3***) della Definizione* **??** *ed in più la seguente proprietà:*

(**r4**) *Se $\psi \in$* **PR** *e $P(\vec{x}, y) := (\forall z \leq y.\ \psi(\vec{x}, z)\downarrow) \wedge \psi(\vec{x}, y) = 0$, allora la funzione φ data da $\varphi(\vec{x}) = \mu y.P(\vec{x}, y)$ appartiene a* **PR**.

Chiamiamo **REC** l'insieme delle funzioni ricorsive parziali che sono totali. Abbiamo evidentemente **REC** $\subset$ **PR**.

Lemma 10. *L'insieme* **REC** *delle funzioni ricorsive totali è il più piccolo insieme di funzioni sui naturali che soddisfa le proprietà (***r1***)-(***r3***) della Definizione* **??** *e la seguente restrizione della proprietà (***r4***):*

(**r4′**) *Se* $\psi \in$ **REC** *e* $\forall \vec{x}.\exists y.\psi(\vec{x}, y) = 0$, *allora la funzione* φ *data da* $\varphi(\vec{x}) = \mu y.[\psi(\vec{x}, y) = 0]$ *appartiene a* **REC**.

Si noti che l'unica causa possibile di parzialità per una funzione in **PR** è proprio lo schema di μ-ricorsione illimitata, per cui con la μ-ricorsione totale si catturano tutte le funzioni ricorsive totali.

Riportiamo, senza dimostrarlo, il fatto che la funzione di Ackermann appartiene a **REC**. Siccome essa non appartiene a **PrimREC**, abbiamo **PrimREC** $\subset$ **REC**.