ANCORA SULLE FUNZIONI RICORSIVE PRIMITIVE

Corso di Informatica Teorica - modulo 2

Prof. **Settimo Termini**

Commenti sulla funzione produttività

Prima di tornare alle funzioni ricorsive primitive, ricordiamo una conseguenza della dimostrata non Turing calcolabilità della funzione produttività.
Poichè da una breve analisi di quello che dovremmo fare noi (diciamo "a mano") per calcolare il valore della funzione produttività emerge che il problema più difficile col quale ci si scontra è quello di stabilire se una generica MdT si ferma prima o poi su un certo valore della variabile di ingresso, osserviamo che questo è il problema cruciale.

Ricordiamo ancora che i problemi connessi con la funzione produttività sono stati presentati per la prima volta dal matematico ungherese T. Rado nel 1962 e che ad esse si fa riferimento col nome di Busy Beaver Problem.

Funzioni ricorsive primitive

Torniamo adesso alle nostre funzioni ricorsive primitive.

Ricordiamo la definizione.
Una funzione si dice ***ricorsiva primitiva*** se può essere ottenuta a partire dalle ***funzioni iniziali*** mediante un numero finito di applicazioni delle operazioni di ***composizione*** e ***ricorsione***

Ricordiamo ancora che come funzioni iniziali abbiamo preso la ***funzione successore***, la ***funzione costante zero*** e le ***funzioni di selezione***.

(vedere le slides precedenti)

Esempi di funzioni ricorsive primitive

Mostriamo adesso la ricorsività primitiva di alcune funzioni.

ATTENZIONE Questo equivale a **dare la dimostrazione che esse sono ricavabili in modo meccanico dalle funzioni iniziali usando soltanto, un numero finito di volte, le operazioni di composizione e ricorsione**.

ESEMPIO 1 La funzione **somma** $f(x, y) = x + y$

La usuale definizione ricorsiva di somma è $(\alpha) \begin{cases} f(x,0) = x \\ f(x,y+1) = f(x,y) + 1 \end{cases}$

COMMENTO La definizione è equivalente a $\begin{cases} x + 0 = x \\ x + (y+1) = (x+y) + 1 \end{cases}$

Esempi di funzioni ricorsive primitive

Purtroppo la (α) per noi **non** è soddisfacente perché le funzioni scritte a destra del segno = non sono funzioni iniziali (che sono le uniche che - al momento - sappiamo essere ricorsive primitive).

Scriviamo $x = u_1^1(x)$

$$f(x,y) + 1 = s(u_2^3(y, f(x,y), x))$$

per cui la (α) diviene:

$$(\alpha') \begin{cases} f(x,0) = u_1^1(x) \\ f(x,y+1) = s(u_2^3(y, f(x,y), x)) \end{cases}$$

La (α') ci soddisfa perché $u_1^1(x)$ è funzione iniziale e $s(u_2^3)$ è composizione delle due funzioni iniziali s e u_2^3 .

Esempi di funzioni ricorsive primitive

Essendo la (α') una corretta applicazione dell'operazione di ricorsione a funzioni ricorsive primitive possiamo infine concludere che la funzione somma $f(x,\ y) = x + y$ è **ricorsiva primitiva**.

... **COMMENTO** Tutto questo può sembrare estremamente pedante ma è necessario che inizialmente si proceda con estrema cautela. In realtà nella presentazione precedente abbiamo proceduto sempre in modo abbastanza informale.

Esempi di funzioni ricorsive primitive: il prodotto

E ESEMPIO 2 La funzione **prodotto** $f(x,\ y) = x \cdot y$

Le equazioni di ricorsione solite sono

$$\begin{cases} h(x,0) = 0 \\ h(x,y+1) = h(x,y) + x \end{cases}$$

... COMMENTO Nella notazione tradizionale la definizione è equivalente a

$$\begin{cases} x \cdot 0 = 0 \\ x(y+1) = (x \cdot y) + x \end{cases}$$

Per dimostrare la ricorsività primitiva di h dobbiamo di nuovo riscrivere il lato destro delle due equazioni in maniera opportuna.

Esempi di funzioni ricorsive primitive: il prodotto

OSSERVAZIONE Ricordiamo che la funzione costante zero è una delle funzioni iniziali, indicata con $n(x)$

OSSERVAZIONE Ricordiamo di avere appena dimostrato che la funzione somma è ricorsiva primitiva

Possiamo quindi scrivere $h(x,y) + x = f(u_2^3(y, h(x,y), x), u_3^3(y, h(x,y), x))$

La funzione a destra del segno $=$ è ricorsiva primitiva perché ottenuta per **composizione** di ricorsive primitive (u_2^3 e u_3^3 sono iniziali e la f è la somma).

Le equazioni di ricorsione divengono

$$\begin{cases} h(x,0) = n(x) \\ h(x,y+1) = f(u_2^3(y, h(x,y), x), u_3^3(y, h(x,y), x)) \end{cases}$$

e questo schema ci permette di concludere che il prodotto è una funzione ricorsiva primitiva.

Esempi di funzioni ricorsive primitive: il fattoriale

E **ESEMPIO 3** La funzione **fattoriale** $x! = h(x)$

Le equazioni di ricorsione sono

$$\begin{cases} 0! = 1 \\ (x+1)! = x!(x+1) \end{cases}$$

O **OSSERVAZIONE** $x + 1 = \mathrm{s}(x)$

cioè $\begin{cases} h(0) = 1 \\ h(t+1) = g(t, h(t)) \end{cases}$

dove $g(x_1, x_2) = s(x_1) \cdot x_2 = s(u_1^2(x_1, x_2)) \cdot u_2^2(x_1, x_2)$

O **OSSERVAZIONE** Sappiamo già che la moltiplicazione è ricorsiva primitiva.

Esempi di funzioni ricorsive primitive

E **ESEMPIO 4** La funzione **esponenziale** $e(x, y) = x^y$

Le equazioni di ricorsione sono $\begin{cases} x^0 = 1 \\ x^{y+1} = x^y \cdot x \end{cases}$

O **OSSERVAZIONE** Queste equazioni assegnano il valore 1 all'indeterminato 0^0. Dobbiamo riscrivere in maniera opportuna il secondo membro delle due equazioni.

Si ha $1 = s(n(x))$ (successore della costante zero e quindi composizione di funzioni iniziali)

e $\begin{aligned} x^y \cdot x &= e(x, y) \cdot x \\ &= h(u_2^3(x, e(x, y), y), u_1^1(x)) \end{aligned}$ (e quindi composizione, mediante il prodotto, di funzioni iniziali)

Le equazioni di ricorsione divengono $\begin{cases} x^0 = s(n(x)) \\ e((x, y+1) = h(u_2^3(x, e(x, y), y), u_1^1(x)) \end{cases}$

Elenco di funzioni ricorsive primitive

E ESEMPIO 5 Funzione ***predecessore***

$$p(x) = \begin{cases} x - 1 & \text{se } x \neq 0 \\ 0 & \text{se } x = 0 \end{cases}$$

Le equazioni di ricorsione per la funzione predecessore p sono le seguenti:

$$p(0) = 0$$
$$p(t+1) = t$$

e possiamo concludere che p è *ricorsiva primitiva*

(in effetti dovremmo scrivere esplicitamente i membri di destra usando le funzioni di base)

Elenco di funzioni ricorsive primitive

E **ESEMPIO 6** Funzione differenza modificata

$$x \dot{-} y = \begin{cases} x - y & \text{se } x \geq y \\ 0 & \text{se } x < y \end{cases}$$

È totale mentre $x - y$ è definita per $x < y$ (**ricordiamo che trattiamo solo funzioni definite sugli interi non negativi**)

$$x \dot{-} y = x$$
$$x \dot{-} (t + 1) = p(x \dot{-} t)$$

Elenco di funzioni ricorsive primitive

E ESEMPIO 7 Funzione *differenza assoluta*

$$| x - y |$$

questa funzione è definita come il valore assoluto della differenza tra x *ed* y
Poichè possiamo scriverla in modo molto semplice nel modo che segue:

$$| x - y | = (x \dot{-} y) + (y \dot{-} x)$$

possiamo immediatamente concludere che è ricorsiva primitiva essendo la somma (funzione ricorsiva primitiva) di due funzioni ricorsive primitive

Cardinalità della classe delle funzioni ricorsive primitive

DOMANDA **Quanto è vasta la classe delle funzioni ricorsive primitive?**

Gli strumenti usati per definire tale classe sembrano (e sono) estremamente limitati, tuttavia si può dimostrare che praticamente tutte le funzioni aritmetiche usate ordinariamente in matematica sono ricorsive primitive.
Questo ci indurrebbe allora a credere che la classe delle funzioni ricorsive primitive includa tutti le funzione calcolabili e che pertanto abbiamo trovato la definizione formale cercata di funzione effettivamente calcolabile.
Nonostante la classe delle funzioni ricorsive primitive sia effettivamente amplissima (molto più di quanto uno sia portato inizialmente a pensare) dobbiamo controllare se non sia, tuttavia, possibile costruire funzioni che sono **calcolabili** (sia nel senso di soddisfare i criteri ***intuitivi*** di calcolabilità sia nel senso di corrispondere a metodi effettivi di calcolo) e che, dimostrabilmente, **non** sono ricorsive primitive.

Altre funzioni che sono ricorsive primitive

E ESEMPIO 8

$$\alpha(x) = \begin{cases} 1 & \text{se } x = 0 \\ 0 & \text{se } x \neq 0 \end{cases}$$

Questa funzione non è altro che la differenza modificata di 1 meno x

$$1 \dot{-} x$$

o, ancora si può scrivere informa ricorsiva mediante le semplici equazioni:

$$a(0) = 1$$

$$a(x+1) = 0$$

Altre funzioni che sono ricorsive primitive

- **ESEMPIO 9** $x = y$
- **ESEMPIO 10** $x \neq y$
- **ESEMPIO 11** $x < y$

Altre funzioni che sono ricorsive primitive

E ESEMPIO 12 $x \mid y$ (x è un divisore di y)

E ESEMPIO 13 $Primo(x)$ (x è un numero primo)

E ESEMPIO 14 $\lfloor x/y \rfloor$ (parte intera del quoziente x/y)

E ESEMPIO 15 $R(x,\ y)$ (resto della divisione x/y)

Funzione di Fibonacci

ESEMPIO La funzione definita dalle equazioni

$$\begin{cases} f(0) = 1 \\ f(1) = 1 \\ f(x+2) = f(x+1) + f(x) \end{cases}$$

genera la nota sequenza di Fibonacci nella quale ogni elemento è la somma dei due elementi precedenti:

1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597, ...

Stesso meccanismo di ricorsione?

DOMANDA **La funzione di Fibonacci appartiene alla classe delle funzioni ricorsive primitive?**

Saremmo tentati di rispondere affermativamente per varie ragioni, non ultimo quello (di tipo sociologico e abitudinario) che viene sempre (assieme alla funzione esponenziale) portato come esempio di funzione definita in modo ricorsivo.

Ma la nostra definizione del meccanismo di ricorsione come strumento per comporre funzioni a partire da altre è un po' diverso e più semplice.

La definizione della funzione di Fibonacci NON si riconduce ***immediatamente*** alla definizione di ricorsione che noi abbiamo dato

Stesso meccanismo di ricorsione?

OSSERVIAMO SUBITO: Ovviamente qui ***non*** si avanza alcun dubbio sul fatto che f sia calcolabile e neanche che sia definita usando un qualche meccanismo di ricorsione - inteso in senso intuitivo. La domanda molto semplice e specifica che ci poniamo è la seguente: ***"possiamo definire la funzione di Fibonacci f, usando la definizione di ricorsione che noi abbiamo dato?"***

In caso di risposta affermativa, possiamo concludere che f è ricorsiva primitiva. Se la risposta è negativa, non solo dobbiamo concludere che la funzione f non è ricorsiva primitiva, ma anche porci domande su come estendere questa classe di funzioni.

Stesso meccanismo di ricorsione?

La domanda che ci poniamo, quindi, riguarda più l'adeguatezza della definizione di ricorsione che noi abbiamo dato come operazione per comporre e ottenere nuove funzioni a partire da quelle che già appartengono alla classe da noi definita piuttosto che la natura della funzione di Fibonacci.

Sorge il dubbio che per partire da nozioni e definizioni estremamente semplici alla fine abbiamo dato una definizione troppo restrittiva.

Vedremo che questi dubbi verranno cancellati mostrando che la funzione di Fibonacci è ricorsiva primitiva.

Secondo la definizione data noi dovremo potere scrivere una cosa del tipo:

$\boldsymbol{f}(0) = 1$

$\boldsymbol{f}(\mathrm{x}+1) = \boldsymbol{QUALCOSA}(\boldsymbol{f}(\mathrm{x}))$

Invece noi sappiamo che $\boldsymbol{f}(\mathrm{x}+1)$ dipende dai valori assunti da $\boldsymbol{f}$ su DUE valori precedenti. Come possiamo procedere? Facciamo qualche prova.

Introduciamo una funzione ausiliaria ***FA che possiamo definire come segue:***

$\mathrm{FA}(0) = (1, 1)$

$\mathrm{FA}(\mathrm{x} + 1) = (\boldsymbol{destro}(\mathrm{FA}(\mathrm{x})); \boldsymbol{sin}(\mathrm{FA}(\mathrm{x})) + \boldsymbol{destro}(\mathrm{FA}(\mathrm{x})))$

Il problema sembra risolto con questo piccolo accorgimento perchè ***destro*** e ***sin*** sono funzioni di scelta e la somma sappiamo essere ricorsiva primitiva.

Il nostro ***QUALCOSA*** è (***destro***( ), ***sin***( ) + ***destro***( ))

MA ... il tutto funziona su ***coppie*** di numeri !

Allora il problema non è ancora risolto perchè le funzioni ricorsive primitive sono definite sui numeri naturali, non sulle coppie di numeri naturali. Cosa possiamo fare?

Aggiungiamo qualcosa a quanto fatto.

Codifichiamo opportunamente ogni coppia di numeri, mediante un numero $<a, b> = z$*, essendo DS la decodifica sinistra e DD la decodifica destra, si ha che:*

$$a = DS(z),\ b = DD(z);$$

Se riusciamo a trovare una codifica ricorsiva primitiva allora avremo risolto il problema. Infatti ...

Avremo, infatti:

$$Aus(0) = <1, 1>$$

$$Aus(x+1) = <DD\ Aus(y), DS\ Aus(y) + DD\ Aus(y)>$$

Sotto l'ipotesi che <,>, DD e DS siano ricorsive primitive, possiamo concludere che Aus è ricorsiva primitiva, essendo la ricorsione effettuata dalla funzione:

$$G = < DD(\ ), DS\ (\ ) + DD\ (\ )>.$$

Ritorniamo alla funzione di Fibonacci $\boldsymbol{f}$

Aus mi presenta ogni volta una coppia di numeri di Fibonacci consecutivi codificati, è sufficiente dunque che io decodifichi l'elemento più grande della coppia

$$\boldsymbol{DS(Aus\ (x))}$$

essendo questa la composizione di due funzioni ricorsive primitive, risulta anch'essa ricorsiva primitiva.

Siamo riusciti quindi anche se con un po' di fatica (e sotto l'ipotesi che esistono metodi di codifica ricorsivi primitivi) a mostrare che f è ricorsiva primitiva e quindi che il meccanismo di ricorsione che noi abbiamo proposto è sufficientemente potente per esprimere almeno questo tipo di funzioni.

WILHELM ACKERMANN E GABRIEL SUDAN

Esamineremo adesso un altro meccanismo di ricorsione, detto ricorsione a incastro. Questo meccanismo è presente nella definizione di due funzioni dovute a due allievi di Hilbert, Wilhelm Ackermann e Gabriel Sudan.

Una semplice osservazione

Riportiamo qui di seguito la seconda equazione di alcune funzioni ricorsive primitive, scritta in una forma semplificata, opportuna per le prime considerazioni.

Successore	$s(x) = x + 1$	
Somma	$x + (y + 1) = (x + y) + 1$	usa il **successore**
Prodotto	$x \cdot (y + 1) = x \cdot y + x$	usa la **somma**
Esponenziale	$x^{y+1} = x^y \cdot x$	usa il **prodotto**

OSSERVAZIONE La serie di funzioni presenta le due caratteristiche:

- ciascuna funzione è definita usando l'operazione che la precede nella lista;
- ciascuna funzione cresce più velocemente della funzione che la precede nella lista.

In generale:

Somma $f_1(x,0) = x; f_1(x,y+1) = f_0(f_1(x,y)) \longrightarrow f_1(x,y)+1$

Prodotto $f_2(x,0) = 0; f_2(x,y+1) = f_1(x, f_2(x,y))$

Esponenziale $f_3(x,0) = 1; f_3(x,y+1) = f_2(x, f_3(x,y))$

Vista la regolarità di formazione delle equazioni del lato destro, possiamo pensare di estendere questo procedimento e definire, in generale, una successione di funzioni tali che la seconda equazione definitoria sia:

$$f_{n+1}(x,y+1) = f_n(x, f_{n+1}(x,y))$$

Dato il meccanismo di costruzione possiamo estendere l'osservazione precedentemente fatta a proposito dei primi quattro elementi della successione a tutte le funzioni e concludere che ogni f_n cresce più rapidamente di tutte le funzioni che la precedono nella successione.

Proviamo a scrivere il quarto termine della successione, sia pure in modo non del tutto rigoroso, usando un simbolismo familiare.
È facile convincersi che si ha

$$f_4(x,y) = \underbrace{\left(\ldots\left((x)^x\right)\ldots\right)^x}_{y \text{ volte}} = x^{x^{\cdot^{\cdot^{\cdot^{x}}}}\}y \text{ volte}}$$

Se tentiamo di andare oltre, ci accorgiamo che gli strumenti di rappresentazione soliti perdono di efficacia; cioè - detto altrimenti - non siamo in grado di scrivere f_5 o funzioni con indice maggiore di 5 in un formalismo familiare.

Concentriamoci sul termine generale:

$$f_{n+1}(x, y+1) = f_n(x, f_{n+1}(x, y))$$

La funzione di Ackermann

ATTENZIONE Ogni funzione della successione non solo è perfettamente definita da un punto di vista matematico ma è calcolabile in modo effettivo e, proprio per come è stata costruita, è ***dimostrabilmente*** ricorsiva primitiva.

Usiamo adesso tale successione di funzioni ricorsive primitive per definire una nuova funzione di tre variabili che tenga conto di tutta la successione; poniamo:

$$F(n, x, y) = f_n(x, y)$$

Osserviamo intanto che il termine generale diviene

$$F(n+1,\ x,\ y+1)\ =\ F(n,\ x,\ F(n+1,\ x,\ y))$$

e si vede subito che la **ricorsione** è effettuata su **due variabili** e questo presenta delle novità rispetto a ciò che abbiamo incontrato e studiato fino ad ora.

DOMANDA $F(n,\ x,\ y)$ è ricorsiva primitiva?

La funzione di Ackermann

DETTA ANCHE "ESPONENZIALE GENERALIZZATO" O "FUNZIONE A INCASTRO"

L'idea base di Ackermann è molto semplice. Consideriamo il modo nel quale alcune funzioni sono state **definite** nonché il modo in cui è stata dimostrata la loro ricorsività primitiva.

Passiamo in rassegna le funzioni **successore**, **somma**, **prodotto**, **esponenziale**. Tranne la prima che è una funzione iniziale, tutte le altre sono definite, come abbiamo visto, mediante una operazione di ricorsione.

Costruiamo una lista di funzioni tutte ricorsive primitive ciascuna delle quali usa nella sua equazione definitoria di ricorsione come funzione ausiliaria quella appena costruita.

La funzione di Ackermann

Passiamo infine da questa lista infinita di funzioni tutte ricorsive primitive ad una sola funzione, trasformando l'indice in variabile.

Ci accorgiamo che il meccanismo di ricorsione che emerge alla fine di questa costruzione è diverso e più complesso di quello base.

Ci poniamo di nuovo , come abbiamo fatto per la funzione di Fibonacci, la domanda, se questo meccanismo sia riconducibile a quello base. Ovvero, in modo equivalente, se la funzione così definita, la funzione di Ackermann sia ricorsiva primitiva.

La funzione di Ackermann di due variabili

La definizione completa della funzione di Ackermann è la seguente:

$$
\begin{aligned}
&F(0, y) = y + 1 \\
&F(x + 1, 0) = F(x, 1) \\
&F(x + 1, y + 1) = F(x, F(x + 1, y))
\end{aligned}
$$

OSSERVAZIONE Osserviamo che è stata effettuata una semplificazione eliminando una variabile che non interveniva nei meccanismi di ricorsione e ribattezzando le variabili. All'equazione già nota sono state poi premesse due equazioni che forniscono la base dell'induzione, equazioni che nella costruzione precedente erano state trascurate per semplificare la presentazione del meccanismo che si voleva mettere in evidenza ma che adesso sono necessarie per calcolare effettivamente il valore di F.

Funzione di Sudan

In un contesto diverso il matematico rumeno Gabriel Sudan aveva introdotto nel 1927, [Sur le nombre transfini ω^ω, Bull. Math. Soc. Roumaine de Sciences, 30, pp 11-30] una funzione definita attraverso un meccanosmo di ricorsione strutturalmente simile a quello della funzione di Ackermann. Il suo risultato quindi venne pubblicato un anno prima di Ackermann che lo pubblicò nel 1928, ma al momento non venne osservato nel contesto che stiamo studiando.

Tale funzione è stata ripresentata nell'ambito della teoria della calcolabilità da alcuni matematici rumeni nel corso degli anni '80.

Funzione di Sudan

D DEFINIZIONE

1. $S(x, y, 0) = x + y$
2. $S(x, 0, n+1) = x$
3. $S(x, y+1, n+1) = S(S(x, y, n+1), S(\underbrace{S(x, y, n+1)}, \underbrace{y+1}, 0), n)$

O OSSERVAZIONE In base alla 1, la 3 si può scrivere in modo equivalente:

3'. $S(x, y+1, n+1) = S(S(x, y, n+1), S(x, y, n+1) + y + 1), n)$

Ricorsione ad incastro

ATTENZIONE Si osservi che un modo di verificare la calcolabilità di queste due funzioni è quello di usare le tre equazioni definitorie come "regole meccaniche" per il calcolo effettivo di S, cioè come istruzioni rigide che potrebbe effettuare una macchina. Guardiamo un attimo insieme le due funzioni di Ackermann e Sudan. Limitiamoci all'equazione generale che è la più significativa:

Ackermann $$F(x+1, y+1) = F(x, F(x+1, y))$$

Sudan $$S(x, y+1, n+1) = S(S(x, y, n+1), S(x, y, n+1) + y + 1), n)$$

In entrambe è evidente un meccanismo di ricorsione a incastro non immediatamente riconducibile ad una ricorsione semplice.

A mo' di conclusione provvisoria: facciamo il punto

Cosa abbiamo fatto?

1. Intanto ricordiamo di essere reduci da un risultato negativo. La funzione produttività non è Turing calcolabile. Peggio, come conseguenza di un'analisi delle ragioni di questo risultato negativo siamo giunti alla conclusione che non può esistere nessuna MdT che mi dice se un'altra MdT si fermerà prima o poi a partire da condizioni iniziali date.

2. Abbiamo seguito il suggerimento di Turing, di trovare anche altri modelli di computo e ne abbiamo proposto uno (il più semplice e semplificato possibile.

A mo' di conclusione provvisoria: facciamo il punto

Cosa abbiamo fatto?

3. Abbiamo avuto qualche piccola difficoltà a mostrare che la funzione di Fibonacci rientri in questo schema definitorio, ma alla fine ci siamo riusciti, sotto l'ipotesi che esistano metodi di codifica (e decodifica) ricorsivi primitivi.

4. Dobbiamo trovare questi meccanismi di codifica. Ma siamo abbastanza fiduciosi su questo punto

Quindi: ...

A mo' di conclusione provvisoria: facciamo il punto

5. Abbiamo poi definito una nuova funzione anch'essa definita mediante meccanismi di ricorsione, ma molto più complesso dei precedenti anche se presenta qualche somiglianza anche con quello presente nella funzione di Fibonacci.

6. Scartabbellando nella letteratura scientifica abbiamo trovato che in un altro contesto era stata definita per tutt'altri scopi una funzione che usava lo stesso meccanismo di ricorsione.

7. si vede che queste due funzioni si possono in ogni caso calcolare "a mano" (con carta e penna) in modo molto semplice.

Allora ...

A mo' di conclusione provvisoria: facciamo il punto

... allora:

(- INTANTO NON DIMENTICHIAMO che dobbiamo cercare meccanismi di codifica ricorsivi primitivi.)

- Pensiamo a strategie da seguire per dimostrare che la funzione di Ackermann è ricorsiva primitiva (e, se non dovessimo riuscirci, vedremo che conclusioni si devono trarre).

- Vediamo se possiamo prendere spunto dalle regole che si usano per calcolare "a mano" le funzioni di Ackermann e Sudan per delineare un modello di computo (ancora un altro modello!)