Induzione Matematica e Verifica di Funzioni Ricorsive

Note Didattiche ad uso degli Studenti di Programmazione
Anno 99/00

Vladimiro Sassone

INDICE

Data: 24 Ottobre 1999. *Versione:* **2.00** vs

Introduzione

Queste note descrivono ed esemplificano il *principio di induzione ben fondata* ed alcune tecniche elementari per l'analisi e la sintesi di definizioni *ricorsive* di funzioni basate su tale principio.

Il termine funzione è usato qui sia nel senso 'semantico' di funzione matematica, cioè una *applicazione* f da un dominio D ad un codominio C, indicata usualmente con $f\colon D \to C$, sia nel senso 'sintattico' di *programma funzionale*. In senso stretto, in quest'ultimo caso sarebbe certamente più corretto parlare di *definizione* (ricorsiva) di funzione piuttosto che di funzione; il lettore è esortato a fare attenzione a questo abuso di linguaggio.

1. Induzione Matematica

Consideriamo il problema di definire una funzione dai naturali nei booleani che valga *true* se e solo se il suo argomento è un numero pari, supponendo di non disporre dell'operazione di divisione. Una possibile definizione di tale funzione è la seguente:

```
pari n =
  if n == 0 then True
    else if n == 1 then False
      else pari(n-2)
```

Come garantire che tale definizione soddisfi la proprietà

$$pari(n) \equiv \text{`}n \text{ è un numero pari'}$$

ovvero che la funzione definita sia corretta rispetto la specifica?

Sfruttando la nostra intuizione sul procedimento meccanico per determinare il valore che una funzione ricorsiva assume su un elemento del suo dominio, possiamo notare immediatamente due fatti importanti:

1. una buona definizione ricorsiva deve essere tale da garantire che la funzione sia definita immediatamente su alcuni casi semplici, detti ***casi base***, o ***casi terminali***, come ad esempio il caso $n = 0$ nella definizione della funzione fattoriale, o i casi $n = 0$ ed $n = 1$ in quella della funzione `pari`;
2. il calcolo della funzione sui casi non base, ovvero i ***casi ricorsivi***, deve poter essere ricondotta al calcolo della funzione stessa su argomenti 'più semplici', cioè argomenti che ci avvicinano ad uno dei casi base, come ad esempio $n - 1$ nel caso della funzione fattoriale, o $n - 2$ nel caso della funzione `pari`.

La nozione di 'essere più semplice' – che in generale può essere niente affatto banale – induce un 'ordinamento' sul dominio di definizione della funzione che facilita notevolmente il nostro compito suggerendo un metodo per dimostrare proprietà di funzioni ricorsive. Infatti, una volta verificato che una certa proprietà valga sugli argomenti 'più semplici' (minimali nell'ordinamento), il secondo fatto permette di ricondurre la validità della proprietà su argomenti complessi alla sua validità su argomenti 'più semplici'. Questo modo di procedere è noto in matematica sotto il nome di ***procedimento induttivo*** e, come vedremo, è imprescindibilmente legato al concetto di definizione ricorsiva.

1.1. Induzione sui Naturali. In matematica, il principio di induzione è la base per dimostrazioni che coinvolgono numeri naturali. In particolare, come vedremo in seguito, è possibile utilizzare opportune versioni del principio di induzione anche per ragionare sulle proprietà di funzioni definite in modo ricorsivo. Il problema, in questo contesto, è analogo a quello affrontato nel paradigma imperativo, laddove si introduce una semantica alternativa a quella operazionale, basata sul calcolo delle *weakest preconditions*, che consente di ragionare sulle proprietà dei programmi.

Per il momento, prima di introdurre una semplice notazione per scrivere programmi funzionali (ovvero definizioni (ricorsive) 'eseguibili' di funzioni), studieremo alcuni concetti base che utilizzeremo poi per progettare funzioni ricorsive e verificarne proprietà.

Sia $\phi(n)$ una data proprietà sui naturali definita per $n \in \mathbb{N}$. Formalmente una proprietà può essere rappresentata come il sottoinsieme $\phi \subseteq \mathbb{N}$ degli n per cui $\phi(n) \equiv \mathit{true}$, cioè $\phi(n) \equiv \mathit{true}$ se e solo se $n \in \phi$. Consideriamo, ad esempio, la proprietà $\phi(n) \equiv$ '$n \cdot (n+3)$ *è pari*' sui numeri naturali e supponiamo di voler dimostrare che $\phi(n)$ è vera per ogni naturale, ovvero che $\forall n \in \mathbb{N}.\ \phi(n) \equiv \mathit{true}$. Possiamo pensare di procedere per induzione, ovvero sfruttando il ***principio di induzione matematica***.

TEOREMA 1.1 *(Induzione).* *Sia ϕ una proprietà sui naturali, e $n_0 \in \mathbb{N}$. Se*

- ▷ $\phi(n_0) \equiv \mathit{true}$; *e*
- ▷ *se per ogni* $n \in \mathbb{N}$, $n \geq n_0$, $\phi(n) \equiv \mathit{true}$ *allora* $\phi(n+1) \equiv \mathit{true}$

allora $\phi(n) \equiv \mathit{true}$ *per ogni* $n \geq n_0$. *Ovvero:*

$$\begin{array}{ll} \textbf{\textit{se}} & \phi(n_0) \wedge \forall n\colon n \geq n_0.\ \phi(n) \Rightarrow \phi(n+1) \\ \textbf{\textit{allora}} & \forall n \in \mathbb{N}.\ n \geq n_0 \Rightarrow \phi(n). \end{array}$$

Al fine di illustrare tale principio, consideriamo ad esempio la seguente sequenza di equazioni:

$$0 = 0^2,\ 1 = 1^2,\ 1+3 = 2^2,\ 1+3+5 = 3^2,\ 1+3+5+7 = 4^2,$$
$$1+3+5+7+9 = 5^2, \ldots$$

Osservando questa sequenza si può pensare che la proprietà

$$\phi(n) \equiv 1+3+\ldots+(2n-1) = n^2$$

possa valere su tutti i naturali, visto che $\phi(0) \equiv \phi(1) \equiv \phi(2) \equiv \phi(3) \equiv \mathit{true}$. Per verificare ciò, ovvero che $\phi = \mathbb{N}$, utilizziamo il principio di induzione come segue.

1. $\phi(0) \equiv 0 = 0^2 \equiv \mathit{true}$.
2. Consideriamo un generico numero naturale $n \geq 1$, e supponiamo che $\phi(n)$ sia vera, ovvero $1+3+\ldots+(2n-1) = n^2$. Consideriamo il caso

$\phi(n+1)$, successore. È facile vedere che:

$$\begin{array}{rl}
 & 1+3+\cdots+(2n-1)+(2(n+1)-1) \\
= & \qquad \{\ \textbf{Ip-Ind:}\ 1+3+\cdots+(2n-1)=n^2\ \} \\
 & n^2+(2(n+1)-1) \\
= & \qquad \{\ \text{aritmetica}\ \} \\
 & n^2+2n+1 \\
= & \qquad \{\ \text{aritmetica}\ \} \\
 & (n+1)^2
\end{array}$$

Ovvero, $\phi(n+1) \equiv \mathit{true}$, per un generico (quindi per tutti gli) $n \geq 1$.

Ne consegue quindi che, per il principio di induzione,

$$\forall n \in \mathbb{N}.\ \phi(n).$$

Si osservi che i punti chiave di questa dimostrazione sono stati:

- ▷ individuazione di una caso base, $n_0 = 0$, da cui far partire la dimostrazione;
- ▷ utilizzazione della *ipotesi induttiva* $\phi(n)$ nella dimostrazione di $\phi(n+1)$, ovvero fissato un generico $n \in \mathbb{N}$ tale che $n > n_0$, si utilizza $\phi(n) \equiv \mathit{true}$ per dimostrare che $\phi(n+1) \equiv \mathit{true}$.

Il primo caso viene detto *caso base*, mentre il secondo viene chiamato *caso induttivo*.

ESERCIZIO 1.1. Si dimostri che $\phi(n) \equiv$ '$n \cdot (n+3)$ *è pari*' è vera per ogni naturale.

Ecco un altro esempio di dimostrazione per induzione.

ESEMPIO 1.2. Dimostriamo che $(\forall n \in \mathbb{N}.\ 2^n \geq 1+n)$ applicando il principio di induzione a $\phi(n) \equiv 2^n \geq 1+n$.

Caso base: $\phi(0) \equiv (1 = 1) \equiv \mathit{true}$.

Caso induttivo: Supponiamo l'ipotesi induttiva, ovvero che, fissato un generico $n \in \mathbb{N}$, $\phi(n) \equiv \mathit{true}$. Consideriamo il caso $\phi(n+1)$.

$$\begin{array}{rl}
 & 2^{n+1} \\
= & \qquad \{\ \text{aritmetica}\ \} \\
 & 2^n\ 2 \\
\geq & \qquad \{\ \textbf{Ip-Ind:}\ \phi(n) \equiv (2^n \geq 1+n)\ \} \\
 & (1+n)2 \\
= & \qquad \{\ \text{aritmetica}\ \} \\
 & 2n+2 \\
\geq & \qquad \{\ \text{aritmetica}\ \} \\
 & 1+(1+n)
\end{array}$$

Per il principo di induzione, dunque: $\forall n \in \mathbb{N}.\ 2^n \geq 1+n$.

L'utilizzo del principio di induzione per fare dimostrazioni di proprietà sui naturali richiede una certa cautela nel verificare le varie ipotesi. La seguente dimostrazione, apparentemente corretta, nasconde un grave errore nell'applicazione del principio di induzione.

ESEMPIO 1.3 *(Erroneo)*. Dimostriamo che tutte le persone hanno lo stesso colore d'occhi. Chiaramente, questo asserto è falso. La seguente dimostrazione però, basata sul principio di induzione, fornisce un apparente giustificazione matematica a questo asserto. Assumendo che le persone siano raggruppate in insiemi, dimostriamo che ogni insieme di n persone, contiene tutte persone aventi il medesimo colore d'occhi. Sia dunque

$$\phi(n) \quad \equiv \quad \text{ogni insieme di } n \text{ persone, contiene tutte persone aventi lo stesso colore d'occhi}$$

Si tratta quindi di dimostrare una proprietà per ogni possibile insieme, quindi, per ogni naturale $n \in \mathbb{N}$, con $n \geq 1$. Il caso base, $n = 1$, è ovvio, poiché in un insieme costituito da una sola persona, c'è un solo possibile colore d'occhi. Supponiamo, per ipotesi induttiva, che in ogni insieme di n ($n \geq 1$) elementi, tutti abbiano lo stesso colore d'occhi. Consideriamo il caso induttivo, ovvero un qualsiasi insieme S di $n+1$ persone. Poiché $|S| = n+1$, e $n \geq 1$, allora, S contiene almeno due persone distinte. Siano esse s_1 ed s_2 ($s_1 \neq s_2$). Consideramo gli insiemi $S \setminus \{s_1\}$ e $S \setminus \{s_2\}$. Essendo $|S \setminus \{s_1\}| = |S \setminus \{s_2\}| = n$ possiamo invocare l'ipotesi induttiva, ed affermare che tutte le persone in $S \setminus \{s_i\}$ hanno lo stesso colore d'occhi, per $i = 1, 2$. Siccome $s_1 \in S \setminus \{s_2\}$ e $s_2 \in S \setminus \{s_1\}$, ne consegue che s_1 ed s_2 hanno gli occhi dello stesso colore e, quindi, tutte le persone in S hanno lo stesso colore d'occhi. Quindi, $\forall n \in \mathbb{N}.\ \phi(n)$.

L'errore commesso qui è che, al fine di dedurre che S contiene tutte persone con il medesimo colore d'occhi, gli insiemi $S \setminus \{s_1\}$ e $S \setminus \{s_2\}$ devono contenere almeno un elemento in comune. Questo perché altrimenti non vi sarebbe alcun legame tra il colore degli occhi di s_1 e di s_2. Deve quindi essere $|S \setminus \{s_i\}| \geq 1$ ($i = 1, 2$), ovvero $|S| \geq 2$. Ne consegue che il caso base considerato nella dimostrazione induttiva non è significativo, in quanto, al fine di ottenere una valida dimostrazione per il passo induttivo, è necessario considerare un insieme S di almeno due persone: il caso base significativo è $n = 2$, e l'asserto $\phi(2)$ è chiaramente falso.

Un modo alternativo per convincersi che il procedimento è erroneo è il seguente. Nella dimostrazione del caso induttivo $\phi(n)$ *non* abbiamo *affatto* considerato un *generico* elemento $n > n_0$: non abbiamo utilizzato solo l'ipotesi induttiva, ma di fatto (ed erroneamente) anche l'ulteriore ipotesi $n \geq 3$, compromettendo la genericità di n e, di conseguenza, la validità del principio di induzione.

1.2. Induzione Completa. La formulazione precedente del principio di induzione non sempre è applicabile proficuamente alla dimostrazione di proprietà di natura induttiva di numeri naturali. Ad esempio, volendo dimostrare che ogni numero naturale $n \geq 2$ è fattorizzabile nel prodotto di numeri primi, non c'è alcun modo di sfruttare l'ipotizzata esistenza di una fattorizzazione per n (ipotesi induttiva) per mostrare una fattorizzazione per $n+1$ (caso successore). Infatti, in generale, la fattorizzazione di $n+1$ non ha nulla a che vedere con quella di n. Al fine di ovviare a questa limitazione del principio di induzione, consideriamo la seguente generalizzazione, detta ***principio di induzione completa***.

TEOREMA 1.4 *(Induzione completa).* *Sia $\phi(n)$ una propretà sui naturali $n \in \mathbb{N}$, tale che $n \geq n_0$. Allora:*

se	$\phi(n_0) \wedge \forall n\colon n > n_0.\ (\forall m\colon n_0 \leq m < n.\ \phi(m)) \Rightarrow \phi(n)$
allora	$\forall n : n \geq n_0.\ \phi(n).$

Analogamente al precedente principio di induzione, le dimostrazioni saranno divise in *caso base* e *caso induttivo*.

ESEMPIO 1.5. Dimostriamo per induzione che ogni numero naurale $n \geq 2$ è fattorizzabile in numeri primi. Ovvero, sia:

$$\phi(n) \equiv (n = p_1 \cdots p_k), \text{ con } p_1, \dots, p_k \text{ numeri primi}$$

Ricordiamo che un numero n è primo se è divisibile solo per 1 e n.

Caso Base: $\phi(2) \equiv$ *true*, poiché 2 è un numero primo.
Caso induttivo: Supponiamo valida l'ipotesi induttiva che ogni numero nell'intervallo $[2, n)$ sia rappresentabile come prodotto di numeri primi, ovvero che $\forall m : 2 \leq m < n.\ \phi(m)$. Chiamiamo $p_1^m \dots p_{k_m}^m$ i fattori primi del numero $m \in [2, n)$. Dimostriamo che, sotto l'ipotesi induttiva, $\phi(n) \equiv$ *true*. Possiamo avere due casi:
1. Se n è primo, allora chiaramante $\phi(n) \equiv$ *true*.
2. Supponiamo n non primo. Allora, esiste $a, b \in \mathbb{N}$ tali che $n = a \cdot b$, e $a < n$ e $b < n$. Inoltre, $a \geq 2$ e $b \geq 2$, altrimenti $n = 1$ oppure $n = a$ o $n = b$, che non è possibile secondo le ipotesi. Quindi $a, b \in [2, n)$. Pertanto, per l'ipotesi induttiva, $\phi(a) \equiv \phi(b) \equiv$ *true*. Quindi anche $\phi(n) \equiv$ *true*, poiché, per ipotesi induttiva, la seguente fattorizzazione è in numeri primi:

$$n = p_1^a \cdots p_{k_a}^a \cdot p_1^b \cdots p_{k_b}^b.$$

È importante osservare come, volendo utilizzare il principio di induzione classico, non sia affatto banale ricondurre la dimostrazione di $\phi(n)$, all'ipotesi induttiva $\phi(n-1)$. Al contrario, una ipotesi induttiva più forte, come quella del principio di induzione completa, permetta di risolvere questo problema in modo semplice.

Il seguente esempio, analogo al precedente esempio 1.3, ci fa ricordare l'importanza, nelle dimostrazioni, di una accurata verifica delle corrette ipotesi del principio di induzione. Infatti, anche nel caso dell'induzione completa, è possibile incorrere in dimostrazioni sbagliate, come dimostrato dal seguente esempio.

ESEMPIO 1.6 *(Erroneo).* Sia a un numero naturale fissato. Dimostriamo che $\forall n \in \mathbb{N}.\ n \geq 1 \Rightarrow a^{n-1} = 1$.

Caso base: Per $n = 1$: $a^{1-1} = a^0 = 1$.
Caso Induttivo: Supponiamo che, fissato $n \geq 1$,

$$\forall m : 1 \leq m \leq n.\ a^{m-1} = 1.$$

Consideriamo $a^{(n+1)-1}$:

$$
\begin{array}{ll}
 & a^{(n+1)-1} \\
= & \quad \{\text{ aritmetica }\} \\
 & a^{n} \\
= & \quad \{\text{ aritmetica }\} \\
 & \dfrac{a^{n-1} \cdot a^{n-1}}{a^{n-2}} \\
= & \quad \{\text{ \textbf{Ip-Ind:} } a^{n-1} = 1 \text{ e } a^{n-2} = 1 \} \\
 & \dfrac{1 \cdot 1}{1} \\
= & \quad \{\text{ aritmetica }\} \\
 & 1
\end{array}
$$

Da questa dimostrazione per induzione, conseguirebbe che $\forall n \geq 1.\ a^{n-1} = 1$. Questo è chiaramente falso (salvo che nel caso $a = 1$). Ed infatti questa dimostrazione nasconde un errore. Ancora una volta il caso base scelto non è significativo per l'induzione. Infatti, ad esempio per $n = 2$, $a^{n-1} = a \neq 1$ (a meno che sia $a = 1$). Dove abbiamno sbagliato? Osservando i calcoli notiamo che per poter applicare correttamente l'ipotesi induttiva è necessaro assumere che a^{n-2} sia definita, il che avviene solo se $n \geq 2$: il caso base significativo deve essere $n = 2$. Chiaramente, un corretto uso dell'induzione, evidenzia immediatamente che, già dal caso base, la proprietà che si vuole dimostrare è falsa.

Guardando la questione dall'altro punto di vista, ancora una volta n nella nostra dimostrazione *non* è un *generico* numero maggiore di n_0: abbiamo usato un'ipotesi (a^{n-2} definito) che di fatto restringe n ad essere maggiore di 2.

Il seguente teorema, correla i due principi di induzione visti precedentemente. In particolare, esso ne dimostra l'equivalenza.

TEOREMA 1.7. *Sia* ϕ *una proprietà sui naturali.*

$$(\phi(n_0) \wedge \forall n \in \mathbb{N}.\ n \geq n_0 \wedge \phi(n) \Rightarrow \phi(n+1)) \Rightarrow \forall n \in \mathbb{N}.\ n \geq n_0 \Rightarrow \phi(n)$$

se e solamente se

$$(\phi(n_0) \wedge \forall n\colon n > n_0.\ (\forall m\colon n_0 \leq m < n.\ \phi(m)) \Rightarrow \phi(n)) \Rightarrow \forall n\colon n \geq n_0.\ \phi(n).$$

Questo risultato non sminuisce affatto l'importanza del principio di induzione completa. Infatti, pur essendo equivalenti i due principi, il secondo (induzione completa) supponendo un'ipotesi induttiva più forte, risulta spesso più facilmente utilizzabile del primo, come evidenziato nel precedente esempio 1.5.

1.3. Induzione Ben Fondata. Tipicamente le funzioni ricorsive esprimibili in un qualsiasi linguaggio funzionale trattano oggetti generici quali numeri, caratteri, liste, alberi, *etc.*, e non necessariamente solo numeri naturali. Inoltre, non è affatto detto che la ricorsione calcoli $f(n)$ in funzione di $f(m)$ esclusivamente per numeri $m < n$. In questo senso, il principio di induzione completa enunciato sopra per dimostrazioni di proprietà sui naturali perde efficacia se applicato per dimostrare proprietà di insiemi non costituiti necessariamene da numeri naturali (o comunque usando un 'ordinamento' diverso da quello naturale) quali per esempio quelli manipolati da generiche funzioni ricorsive.

Il principio di induzione può essere generalizzato ulteriormente, considerando, al posto dei numeri naturali con l'ordinamento canonico $\leq$, un insieme qualsiasi ed una relazione generica che esprima una certa 'precedenza', o 'ordine', tra gli elementi dell'insieme. Essenziale, come nelle precedenti formulazioni del principio di induzione, è il riconoscimento corretto di un *caso base* ed una opportuna *ipotesi induttiva*.

Nel seguito, generalizzeremo ulteriormente il principio di induzione completa visto precedentemente. Questa generalizzazone costituirà la base per effettuare dimostrazioni di proprietà di funzioni ricorsive, come vedremo in seguito.

Il primo problema da affrontare è quello di formalizzare in modo generale i concetti di *caso base* e *argomento più semplice* già visti nel caso dell'induzione sui naturali.

Sui naturali è ovvio ricondurre la nozione di caso base e argomento più semplice all'usuale ordinamento tra numeri: n_0 è il caso base, mentre la dimostrazione per il caso n viene ricondotta alla dimostrazione del caso più semplice $m < n$. Per una generalizzazione efficace di questi concetti, in particolare del concetto di *essere più semplice* per un elemento di un insieme rispetto ad un altro, utilizziamo la definizione di *relazione* tra gli elementi di un insieme.

DEFINIZIONE 1.8 ***(Relazioni su insiemi).*** Sia S un insieme. Una relazione su S è un sottoinsieme del prodotto cartesiano $S \times S$.

Detto altrimenti, una relazione $\mathcal{R}$ su S è un insieme di coppie di elementi di S, $\mathcal{R} \subseteq S \times S$. Data $\mathcal{R}$, useremo la usuale notazione infissa $x \ \mathcal{R} \ y$ per denotare il fatto che l'elemento $x \in S$ ***precede*** l'elemento $y \in S$ secondo la relazione $\mathcal{R}$. Poiché questa relazione esprimerà una ***precedenza*** tra x e y, faremo spesso uso di simboli quali $\prec, \sqsubset, < \ldots$ Nel seguito useremo la notazione $(S, \prec)$ per indicare che l'insieme S è fornito di una relazione $\prec$.

DEFINIZIONE 1.9 ***(Elemento Minimale).*** Sia $(S, \prec)$ un insieme con una relazione $\prec$. L'elemento $x \in S$ è minimale rispetto a $\prec$ se $\forall y \in S.\ y \not\prec x$.

Gli elementi massimali di un insieme con relazione di precedenza, quando esistono, sono definiti invertendo la relazione $\prec$ nella definizione 1.9. Ovvero $x \in S$ è massimale rispetto a $\prec$ se $\forall y \in S.\ x \not\prec y$.

ESERCIZIO 1.2 ***(Minimo vs Minimale).*** Si faccia attenzione a non confondere il concetto di elemento *minimale* ($\forall y \in S.\ y \not\prec x$) con quello di elemento *minimo* ($\forall y \in S.\ x \prec y$). Si dimostri che un elemento minimo è anche minimale, ma non viceversa e che, mentre può esistere al più *un* elemento minimo, in genere un insieme $(S, \prec)$ ammette *molti* elementi minimali distinti.

L'idea intuitiva che sta alla base di una relazione $\prec$ su un insieme S è quella di strutturare l'insieme dato introducendo una relazione di 'precedenza' tra i suoi elementi. Nel seguito diremo che $\prec$ è una *relazione di precedenza*, dicendo quindi che x precede y nella relazione $\prec$ se $x \prec y$, evidenziando così l'uso di relazioni di precedenza come generalizzazione del concetto di ordinamento tra naturali. Quindi un elemento di S è minimale rispetto alla relazione $\prec$ se nessun elemento lo precede in $\prec$.

ESEMPIO 1.10. Sia $\mathbb{N}$ l'insieme dei numeri naturali e $\prec$ la relazione definita da $x \prec y \equiv y = x + 1$. In altri termini, $x \prec y$ *se e solamente se* y è il successore di x.

L'unico elemento minimale rispetto a $\prec$ è 0. Analogamente, possiamo estendere $\prec$ all'insieme $\mathbb{Z}$ dei numeri interi stipulando che $x \prec y \equiv y = x + 1$ anche per numeri negativi. Non esistono, però, in $\mathbb{Z}$ elementi minimali rispetto a $\prec$. Un esempio di relazione di precedenza non banale su $\mathbb{N}$ è dato invece dalla seguente definizione di $\sqsubset$:

$$x \sqsubset y \equiv x \neq y \wedge divide(x, y)$$

dove $divide(x, y)$ esprime il fatto che x è un divisore di y. L'unico elemento minimale di $(\mathbb{N}, \sqsubset)$ è 1. (Esiste un elemento massimale in $(\mathbb{N}, \sqsubset)$? Quale?)

Esercizio 1.3. È possibile ricostruire l'usuale ordinamento $\leq$ dei numeri naturali a partire da $\prec$ dato nell'esempio 1.10. (Suggerimento: si consideri che $x \leq y$ se e solo se $\exists k \in \mathbb{N}.\ y = x + k$.)

Se $(S, \prec)$ è un insieme con relazione di precedenza, dati $x, y \in S$ con $x \prec y$, diremo che x è un ***predecessore immediato*** di y. Come vedremo, tale concetto avrà un ruolo rilevante nello studio di funzioni definite ricorsivamente.

Possiamo ora generalizzare il principio di induzione visto precedentemente sui naturali considerando un generico insieme con precedenza $(S, \prec)$.

Definizione 1.11 ***(Catena discendente).*** Una catena discendente di elementi di $(S, \prec)$ è una sequenza, possibilmente *infinita*,

$$s_0 \succ s_1 \succ s_2 \succ s_3 \succ \cdots$$

Si osservi, dunque, che una catena discendente infinita è una successione $\{s_i\}_{i \in \mathbb{N}}$ tale che s_{i+1} è predecessore immediato di s_i, per ogni $i \in \mathbb{N}$.

La nozione *duale* di catena ascendente si ottiene rovesciando la relazione $\prec$ nella definizione 1.11.

Esercizio 1.4. Dimostrare che, se esiste $x \in S$ tale che $x \prec x$, allora $(S, \prec)$ ammette almeno una catena discendente infinita.

In seguito, quando la relazione di precedenza tra gli elementi di un insieme S sia chiara dal contesto, diremo semplicemente 'K è una catena', omettendo di menzionare rispetto quale relazione di precendenza.

Esempio 1.12. Sia $(\mathbb{N}, \prec)$ come definito nell'esempio 1.10. Chiaramente $\mathbb{N}$ è una catena, così come l'insieme $P \subseteq \mathbb{N}$ dei numeri pari lo è rispetto alla precedenza $x \prec y \equiv y = x + 2$.

Sia ora $(\mathbb{N}, \sqsubset)$ l'insieme con precedenza come nell'esempio 1.10, in cui $x \sqsubset y$ se e solo se x è un divisore di y. Sia inoltre K_{20} l'insieme dei divisori di 20, ovvero $K_{20} = \{1, 2, 5, 10, 20\}$. L'insieme K_{20} non è una catena rispetto a $\sqsubset$, in quanto né 5 divide 2, né 2 divide 5.

La nozione di catena è essenziale per caratterizzare la buona fondatezza di un insieme corredato con una relazione di precedenza tra i suoi oggetti.

Definizione 1.13 ***(Insieme Ben Fondato).*** Un insieme $(S, \prec)$ con precedenza è detto ***ben fondato*** se non ammette alcuna catena decrescente infinita,

Esercizio 1.5. Si dimostri che in un insieme ben fondato $(S, \prec)$ non è necessariamente vero che $\{y \prec x\}$ sia finito. In altri termini, il fatto che non esistono catene discendenti infinite non significa che ogni elemento di S abbia un numero finito di 'predecessori immediati'.

ESERCIZIO 1.6. Dimostrare che $(S, \prec)$ è ben fondato se e solamente se ogni sottoinsieme non vuoto di S contiene almeno un elemento minimale. (Suggerimento: si osservi che da un insieme non vuoto senza alcun elemento minimale si può *sempre* estrarre una catena infinita. Viceversa ...)

ESEMPIO 1.14. Gli insiemi $(\mathbb{N}, \prec)$ e $(\mathbb{N}, \sqsubset)$ dell'esempio 1.10 sono ben fondati. Non lo è, invece, l'insieme $(\mathbb{Z}, \prec)$ dello stesso esempio. Infatti, $\mathbb{Z}$ contiene una catena infinita decrescente data dall'insieme $\mathbb{Z}^- = \{x \in \mathbb{Z} \mid x \leq 0\}$ dei numeri negativi, ovvero $\mathbb{Z}$ non ammette elementi minimali rispetto a $\prec$.

Si noti come uno stesso insieme possa risultare ben fondato o meno a seconda della relazione di precedenza che si considera.

La buona fondatezza di un insieme rispetto ad una relazione di precedenza su di esso generalizza il fatto che, come nei naturali con l'ordinamento $\leq$, esiste un estremo inferiore in grado di rappresentare il caso base dell'induzione. In questo senso la buona fondatezza di un insieme con una relazione di precedenza assicura l'esistenza di casi base (ovvero elementi minimali) su cui fondare l'induzione.

ESEMPIO 1.15. Consideriamo ora la relazione $x \prec y \equiv x = y + 1$ su $\mathbb{N}$. Si osservi che il ruolo di x e y sono scambiati rispetto alla relazione di precedenza definita all'esempio 1.10. In questo caso $(\mathbb{N}, \prec)$ non è ben fondato. Infatti,

$$\forall n \in \mathbb{N}.\ \exists m \in \mathbb{N}.\ m \prec n$$

È invece ben fondato l'insieme con precedenza $(\mathbb{Z}, \sqsubset)$, dove $\sqsubset$ è definito da

$$x \sqsubset y \equiv (x \geq 0 \wedge y \geq 0 \wedge y > x) \ \vee \ (x \leq 0 \wedge y \leq 0 \wedge x > y)$$

e $\leq$ e $\geq$ denotano le usuali relazioni d'ordine tra numeri interi.

1.4. Diagrammi di Precedenza. Un'utile rappresentazione grafica di un insieme S con relazione di precedenza $\prec$ si ottiene tracciando il cosiddetto ***diagramma di precedenza***.[1] Questa tecnica grafica si rivela particolarmente interessante nello studio di proprietà, per esempio la buona fondatezza, della relazione di precedenza tra gli elementi di un generico insieme. Infatti, mentre ragionare sull'ordinamento $\leq$ dei naturali (su cui si basano i principi di induzione matematica e completa), a causa della loro 'estraneità' può risultare non banale capire e sfruttare la struttura di insiemi e relazioni di precendenza diversi da $\mathbb{N}$ e $\leq$. I diagrammi di precedenza aiutano, fornendo una interpretazione grafica di una relazione di precedenza su un insieme.

Sia $(S, \prec)$ un insieme con relazione di precedenza. Nel diagramma di precedenza per $(S, \prec)$, dati due elementi generici $x, y \in S$, il fatto che $x \prec y$ è rappresentato graficamente dal seguente diagramma:

$$\begin{array}{c} y \\ \Big\uparrow \\ x \end{array}$$

Detto altrimenti, nel diagramma di precedenza, tracciamo una freccia come la precedente *da x a y* se e soltanto se x è un *precedessore immediato* di y rispetto

[1]Questa nozione è una semplice variazione dei ben noti diagrammi di Hasse per insiemi parzialmente ordinati.

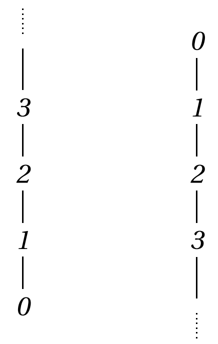


FIGURA 1. Diagrammi di precedenza di $(\mathbb{N}, \prec)$ (Es. 1.10 e 1.15)

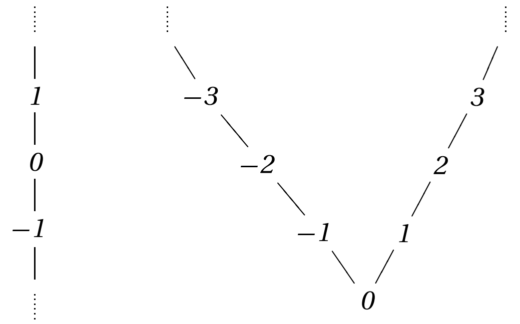


FIGURA 2. Diagramma di precedenza di $(\mathbb{Z}, <)$ e $(\mathbb{N}, \sqsubset)$ (Es. 1.15)

$\prec$. In realtà, laddove questo non crei ambiguità, useremo linee anziché frecce, assumendo la convenzione di leggere il diagramma dal basso verso l'altro. Ad esempio, la situazione precedente verrà rappresentata con il diagramma:

$$\begin{array}{c} y \\ | \\ x \end{array}$$

In questo caso, il fatto che sia x a precedere y, e non viceversa, è rappresentato dal fatto che y è sopra x. Viceversa, scrivendo

$$\begin{array}{c} x \\ | \\ y \end{array}$$

intendiamo dire che y è un predecessore immediato di x.

I diagrammi di precedenza degli insiemi con relazione di precedenza $(\mathbb{N}, \prec)$ degli esempi 1.10 e 1.15 sono rappresentati in Figura 1.

Il seguente esempio evidenzia l'utilità dei diagrammi di precedenza per rappresentare precedenze diverse da quelle standard per i naturali o gli interi.

ESEMPIO 1.16. I diagrammi di precedenza di $(\mathbb{Z}, <)$ di cui all'esempio 1.10, e di $(\mathbb{Z}, \sqsubset)$ dell'esempio 1.15 sono illustrati in Figura 2.

La rappresentazione mediante il diagramma di precedenza consente spesso di verificare facilmente la ben fondatezza o meno di un insieme rispetto ad una relazione di precedenza. Nell'esempio precedente si vede chiaramente che, nel primo caso, l'insieme non è ben fondato (esiste infatti una catena discendente infinita $\{0, -1, -2, \ldots\}$), mentre lo è nel secondo caso (pur essendoci catene infinite ascendenti, non ci sono catene infinite discendenti).

Nel seguito, dato un insieme con relazione di precedenza $(S, \prec)$, per verificarne la ben fondatezza ci accontenteremo di tracciarne il diagramma di precedenza. Quando il diagramma di precedenza fosse costituito da un insieme possibilmente infinito di (sotto)diagrammi, aventi una struttura comune, il diagramma di precedenza dell'intero insieme verrà dato in modo simbolico, ovvero tracciando il diagramma di precedenza di alcune generiche componenti significative, come nell'esempio seguente.

ESEMPIO 1.17. Sia $(\mathbb{N}, \sqsubset)$ l'insieme con precedenza tale che

$$x \sqsubset y \;\equiv\; \exists k\colon k \neq 0.\; y = x + k \cdot A$$

dove A è un numero naturale non nullo, cioè $A \neq 0$. In questo caso, il diagramma di precedenza è costituito da un insieme infinito di catene crescenti. Un modo per rappresentare graficamente tale insieme con precedenza è il seguente:

$$\begin{array}{cccc}
\vdots & \vdots & & \vdots \\
| & | & & | \\
3A & 3A+1 & & 4A-1 \\
| & | & & | \\
2A & 2A+1 & \cdots\cdots & 3A-1 \\
| & | & & | \\
A & A+1 & & 2A-1 \\
| & | & & | \\
0 & 1 & & A-1
\end{array}$$

In questa rappresentazione, vengono evidenziate solo alcune generiche catene del diagramma. Chiaramente, tutte le rimanenti catene hanno la medesima struttura (sono isomorfe). E' evidente pertanto da questo diagramma, che l'insieme dato è ben fondato.

1.5. Induzione Ben Fondata. Veniamo all'enunciato del principio di induzione ben fondata, generalizzando i principi di induzione sui naturali visti precedentemente. Sia S un insieme. Analogamente al principio di induzione completa, il seguente principio di induzione ben fondata permette di dimostrare proprietà di insiemi, generici, ovvero non solo di naturali, equipaggiati con una relazione di precedenza e ben fondati.

TEOREMA 1.18 *(Induzione Ben Fondata).* *Sia $(S, \prec)$ un insieme ben fondato e sia $\phi(x)$ una proprietà su S. Allora vale il seguente asserto:*

$$\begin{array}{ll}
\textbf{\textit{se}} & \forall x \in S.\ (\forall y\colon y \prec x.\ \phi(y)) \Rightarrow \phi(x) \\
\textbf{\textit{allora}} & \qquad \forall x \in S.\ \phi(x).
\end{array}$$

Una lettura informale del principio di induzione ben fondata può essere data come segue: per dimostrare che un asserto $\phi(x)$ vale per ogni elemento x di un insieme ben fondato $(S, \prec)$ è sufficiente dimostrare che ϕ vale su un generico elemento x di S, nell'ipotesi che ϕ valga su ogni elemento y di S che precede x secondo $\prec$. Fissato un generico x in S, la formula

$$\forall y \colon y \prec x.\ \phi(y)$$

viene detta ipotesi induttiva rispetto a x.

Si noti che, in pratica, la dimostrazione di un asserto ϕ basata sul principio di induzione ben fondata avviene in due passi. Analogamente al caso dell'induzione sui naturali, il primo passo della dimostrazione viene comunemente detto caso base, mentre il secondo viene detto caso induttivo.

Caso Base: si dimostra che ϕ vale su tutti gli elementi di S minimali rispetto a $\prec$ (ricordiamo che dato $(S, \prec)$, un elemento $x \in S$ si dice minimale se non esiste nessun elemento $y \in S$ che preceda x). Infatti, se x è minimale, l'insieme degli $y \in S$ tali che $y \prec x$ è vuoto, e dunque l'ipotesi induttiva rispetto a x si riduce a *true*. Di conseguenza la formula $(\forall y \colon y \prec x.\ \phi(y)) \Rightarrow \phi(x)$ si riduce semplicemente a $\phi(x)$: non è possibile usare alcuna ipotesi induttiva.

Caso induttivo: detto x un generico elemento di S, non minimale rispetto a $\prec$, si dimostra la validità di $\phi(x)$ utilizzando, laddove si renda necessaria, l'ipotesi induttiva rispetto a x, ovvero l'ipotesi che ϕ stessa valga su tutti gli elementi di S che precedono x.

È facile osservare come il principio di induzione ben fondata enunciato sopra generalizzi entrambi i principi di induzione matematica sui naturali, ed induzione completa. Il principio di induzione matematica sui naturali, coincide con il principio di induzione ben fondata rispetto all'insieme con precedenza ben fondato $(\mathbb{N} \setminus \{0, \ldots, n_0 - 1\}, \prec)$, dove $\prec$ è definita come segue:

$$x \prec y \equiv y = x + 1.$$

Il principio di induzione completa corrisponde invece al principio di induzione ben fondata rispetto alla chiusura riflessiva e transitiva di $\prec$, ovvero rispetto all'insieme ben fondato $(\mathbb{N} \setminus \{0, \ldots, n_0 - 1\}, <)$ (vedi esempio 1.10).

Prima di vedere alcuni esempi di uso del principio di induzione ben fondata ne diamo una dimostrazione informale.

***Dimostrazione*:** Sia $(S, \prec)$ un insieme ben fondato e sia ϕ una proprietà su S che verifica l'ipotesi del teorema di induzione ben fondata. Consideriamo l'insieme $X = \{x \mid \phi(x) \equiv \mathit{false}\}$. Ovviamente, $X \neq \varnothing$ se e solo se esiste un elemento $x_0 \in S$ per cui non vale ϕ, cioè se e solo se $\neg\forall x \in S.\ \phi(x)$. Dobbiamo dunque dimostrare che $X = \varnothing$.
Siccome $(S, \prec)$ è ben fondata se $X \neq \varnothing$ allora esiste $x \in X$ minimale rispetto $\prec$ (si veda l'esercizio 1.6). Ciò significa che nessun predecessore immediato di x appartiene ad X e quindi, per definizione di X, che ϕ vale su ogni predecessore di x, ovvero

$$\forall y : y \prec x.\ \phi(y).$$

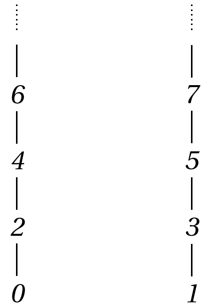


FIGURA 3. Diagramma di precedenza di $(\mathbb{N}, \sqsubset)$

Ma allora, usando l'ipotesi $(\forall y : y \prec x.\ \phi(y)) \Rightarrow \phi(x)$ del principio di induzione ben fondata abbiamo (modus ponens) che $\phi(x) \equiv$ *true*, il che significa $x \notin X$, contraddicendo l'ipotesi $x \in X$. Dobbiamo dunque concludere che X non può contenere alcun elemento, cioè $X = \varnothing$. ✓

ESERCIZIO 1.7. Si dimostri che ogni numero naturale $n \geq 2$ può essere fattorizzato come prodotto di primi usando l'induzione ben fondata su

$$(\mathbb{N} \setminus \{0, 1\}, \sqsubset), \qquad \text{dove } n \sqsubset m \equiv \exists k \geq 2.\ m = k \cdot n.$$

Si noti che prima di applicare il principio di induzione ben fondata è necessario giustificare l'assunzione che $\sqsubset$ sia ben fondata.

ESEMPIO 1.19. Dimostriamo, per induzione ben fondata, la ben nota formula di Gauss

$$\forall n \in \mathbb{N}.\ \left(\sum i : i \in [0, n].i = \frac{n(n+1)}{2}\right)$$

Indichiamo con $\phi(n)$ la proprietà in esame, ovvero:

$$\phi(n) \equiv \left(\sum i : i \in [0, n].i = \frac{n(n+1)}{2}\right)$$

Anziché considerare l'usuale ordinamento $\leq$ su $\mathbb{N}$, consideriamo l'insieme ben fondato $(\mathbb{N}, \sqsubset)$, in cui la relazione di precedenza è data dalla relazione definita da:

$$x \sqsubset y \equiv (pari(x) \wedge pari(y) \wedge x < y) \vee (dispari(x) \wedge dispari(y) \wedge x < y)$$

Il diagramma di tale relazione di precedenza può essere disegnato come in Figura 3. Questo diagramma mostra come $(\mathbb{N}, \sqsubset)$ sia ben fondato, e come gli unici elementi minimali rispetto a $\sqsubset$ siano 0 ed 1.

Dimostriamo ora, per induzione ben fondata, la precedente proprietà della somma di naturali.

Caso base: Dobbiamo dimostrare che la proprietà ϕ vale sugli elementi minimali della precedenza, ovvero dobbiamo dimostrare $\phi(0)$ e $\phi(1)$.

$$
\begin{array}{ll|ll}
 & \sum i : i \in [0, 0].i & & \sum i : i \in [0, 1].i \\
= & \quad \{ \text{intervallo} \} & = & \quad \{ \text{intervallo} \} \\
 & \sum i : i = 0.i & & \sum i : i = 0 \vee i = 1.i \\
= & \quad \{ \text{singoletto} \} & = & \quad \{ \textstyle\sum : \vee \} \\
 & 0 & & 0 + 1 \\
= & \quad \{ \text{aritmetica} \} & = & \quad \{ \text{aritmetica} \} \\
 & \frac{0(0+1)}{2} & & \frac{1(1+1)}{2}
\end{array}
$$

Caso Induttivo: Sia ora $n \in \mathbb{N}$ un generico elemento non minimale di $\mathbb{N}$ rispetto a $\sqsubset$, ovvero sia $n \geq 2$. L'ipotesi induttiva, che indichiamo con $IND(n)$, è:

$$IND(n) \equiv \forall m\colon\ m \sqsubset n,\ \ \phi(m)$$

Dunque, supponendo $IND(n)$ vera, abbiamo che:

$$
\begin{array}{ll}
 & \sum i : i \in [0, n].i \\
= & \quad \{ \textbf{Ip: } n \geq 2, \text{intervallo} \} \\
 & (\sum i : i \in [0, n-2].i) + (n-1) + n \\
= & \quad \{ \textbf{Ip-Ind: } IND(n), (n-2) \sqsubset n \} \\
 & \frac{(n-2)(n-1)}{2} + (n-1) + n \\
= & \quad \{ \text{aritmetica} \} \\
 & \frac{(n-2)(n-1) + 2(n-1) + 2n}{2} \\
= & \quad \{ \text{aritmetica} \} \\
 & \frac{(n-2)(n-1) + 2(2n-1)}{2} \\
= & \quad \{ \text{aritmetica} \} \\
 & \frac{(n^2 - n - 2n + 2 + 4n - 2)}{2} \\
= & \quad \{ \text{aritmetica} \} \\
 & \frac{n(n+1)}{2}
\end{array}
$$

È importante osservare come, nel primo passo di dimostrazione, si sia utilizzata l'ipotesi che n è un elemento non minimale, ovvero $n \geq 2$, per l'applicazione delle leggi sugli intervalli. Inoltre il secondo passo è un'applicazione dell'ipotesi induttiva: poiché $(n-2)$ precede n nella relazione di precedenza considerata, l'ipotesi induttiva garantisce che vale $\phi(n-2)$, ovvero che vale l'uguaglianza

$$\sum i : i \in [0, n-2].i = \frac{(n-2)(n-1)}{2}.$$

1.6. Esercizi.

ESERCIZIO 1.8. Dato un insieme X, sia $\mathcal{P}(X)$ l'insieme delle sue parti. Si dimostri per induzione su n che se la cardinalità di X è n, allora la cardinalità di $\mathcal{P}(X)$ è 2^n.

ESERCIZIO 1.9. Una stringa è una sequenza finita $\alpha = \alpha_0\alpha_1 \cdots \alpha_n$ ($n \geq 0$) di simboli α_i appartenenti ad un qualche alfabeto Σ. Dimostrare per induzione su n che non esiste alcuna stringa α su Σ tale che $\beta\alpha = \gamma\alpha$, per $\beta \neq \gamma \in \Sigma$.

ESERCIZIO 1.10. Si dimostri per induzione (ad esempio su m) che, fissato un qualsiasi numero x, $\forall n, m \in \mathbb{N}.\ x^{n+m} = x^n \cdot x^m$.

ESERCIZIO 1.11. Si dimostri per induzione che $\forall k, n, m \in \mathbb{N}.\ k \cdot (n + m) = k \cdot n + k \cdot m$. (Si faccia attenzione ad usare la precedente proprietà distributiva solo come ipotesi induttiva.)

ESERCIZIO 1.12. Sia $\{f_k\}_{k \in \mathbb{N}}$ la successione di Fibonacci (cioè $f_0 = 0, f_1 = 1$, $f_{k+2} = f_k + f_{k+1}$). Si considerino i numeri

$$\varphi = \frac{1 + \sqrt{5}}{2} \qquad \hat{\varphi} = \frac{1 - \sqrt{5}}{2}$$

Si dimostri per induzione su k che $f_k = \frac{1}{\sqrt{5}} \cdot (\varphi^k - \hat{\varphi}^k)$

ESERCIZIO 1.13. Sia $\{f_k\}_{k \in \mathbb{N}}$ la successione di Fibonacci. Si dimostri per induzione che per ogni $n \geq 1$ e $m \geq 0$,

$$f_{n+1} \cdot f_{n-1} - (f_n)^2 = (-1)^n$$

$$f_{n+m} = f_n \cdot f_{m+1} + f_{n-1} \cdot f_m$$

ESERCIZIO 1.14 ***(Precedenza Immagine Inversa).*** Sia X un generico insieme, $(S, \prec)$ un insieme ben fondato, e $f \colon X \to S$ una funzione. Si dimostri che, definendo

$$x_1 \prec_X x_2 \equiv x_1 \neq x_2 \wedge f(x_1) \prec f(x_2),$$

l'insieme con precedenza $(X, \prec_X)$ è ben fondato.

ESERCIZIO 1.15 ***(Precedenza Prodotto).*** Siano $(S_1, \prec_1)$ e $(S_2, \prec_2)$ insiemi ben fondati. Si consideri la precedenza $\prec$ su $S_1 \times S_2$ definita da

$$(s_1, s_2) \prec (s_1', s_2') \equiv s_1 \prec_1 s_1' \wedge s_2 \prec_2 s_2'.$$

Si dimostri che $(S_1 \times S_2, \prec)$ è ben fondato.

ESERCIZIO 1.16 ***(Precedenza Lessicografica).*** Dati insiemi con precedenza $(S_1, \prec_1)$ ed $(S_2, \prec_2)$, la ***precedenza lessicografica*** $\prec_L$ su $S_1 \times S_2$ è la relazione $\prec$ definita da

$$(s_1, s_2) \prec (s_1', s_2') \equiv s_1 \prec_1 s_1' \vee (s_1 = s_1' \wedge s_2 \prec_2 s_2').$$

Si dimostri che se $(S_1, \prec_1)$ e $(S_2, \prec_2)$ sono ben fondati, allora $(S_1 \times S_2, \prec_L)$ è ben fondato.

ESERCIZIO 1.17 ***(Precedenza Lessicografica, II).*** Si discuta la generalizzazione del concetto di precedenza lessicografica di cui all'esercizio precedente a triple di insiemi con precedenza (ben fondati) e, in generale, a n-ple di tali insiemi.

ESERCIZIO 1.18. Sia $(S, \prec)$ un insieme ben fondato. Si denoti con $\prec^+$ la chiusura transitiva di $\prec$ e con $\prec^*$ la sua chiusura riflessiva e transitiva. Si dimostri che $(S, \prec^+)$ è ben fondato e che $(S, \prec^*)$ è un ordinamento parziale.

2. Un Semplice Linguaggio Funzionale

Fissiamo ora una notazione semplice e minimale, ispirata dal linguaggio Haskell, per descrivere funzioni e definizioni ricorsive delle stesse. La base sarà, ovviamente, costituita dalle usuali espressioni a valori naturali, parametriche rispetto identificatori su tali valori. Ci prenderemo la libertà di usare nei nostri esempi le usuali operazioni aritmetiche `+`, `-`, `*`, `‘div‘` e `‘mod‘`. Per quanto riguarda il tipo booleano, oltre alle costanti `True` e `False`, ed i connettivi `not`, `&&` (`and`) e `||` (`or`), useremo i soliti operatori di confronto ==, /= (non uguale), <, <=, > e >=.

Avremo inoltre bisogno di un meccanismo per scrivere ***espressioni condizionali***, cioè espressioni in cui valore dipende dall'ambiente. Useremo il costrutto

`if` E `then` E_1 `else` E_2

dove E è un espressione *booleana*, la ***guardia***, e E_1 ed E_2 rappresentano generiche espressioni, come ad esempio in

```
if x > 0 then 1 else if x == 0 then 0 else -1
m ‘div‘ (if n == 0 then 1 else n)
```

Si noti che tale scrittura non esprime un meccanismo di controllo di flusso usato per eseguire un gruppo di comandi piuttosto di un altro. Essa è una vera e propria *espressione*, indefinita quando la guardia è indefinita, e che invece assume il valore di E_1 o quello di E_2 rispettivamente quando la guardia vale *True* o *False*. Come conseguenza, oltre al fatto che la guardia deve essere un'espressione booleana, è necessario che:

1. ci sia sempre un ramo `else` (in altri termini, il costrutto `if` E `then` E_1 *non* esiste; quale sarebbe altrimenti il valore dell'espressione nel caso di guardia falsa?); e
2. il tipo delle espressioni nei due rami deve essere lo stesso; cioè la scrittura `if` E `then 1 else True` è un irrimediabile errore sintattico.

Per associare un nome ad una funzione useremo ***definizioni*** introdotte dal simbolo =, adottando un modo semplice e naturale per descrivere tale funzione: a destra del simbolo = porremo un'espressione parametrica che, al variare dei parametri nel loro dominio di definizione, assumerà opportuni valori, come ad esempio in:

```
sqr x = x * x
```

Qui, come reso esplicito dalla sintassi `sqr x`, `sqr` è il ***nome*** della funzione, `x` è un ***parametro*** nella espressione `x * x` che costituisce il ***corpo*** della funzione. In questo contesto un parametro è niente altro che un 'segnaposto' per il quale dovrà essere fornito *esplicitamente* un valore reale al momento di applicazione la funzione. Infatti, non è molto utile 'valutare' `sqr` in quanto tale; piuttosto, si valuta l'applicazione di `sqr` ad un preciso argomento, come ad esempio `sqr 3`.

Consideriamo alcuni altri esempi di definizione di funzioni.

```
doppio x = x + x
triplo x = 3 * x
sestuplo x = doppio(triplo x)
pari x = x mod 2 = 0
min(x,y) = if x < y then x else y
```

Un esempio più interessante è la funzione che restituisce il massimo comun divisore di due naturali.[2]

```
gcd m n =
  if m == 0 then n
    else gcd (n mod m) m
```

Rivolgiamo ora la nostra attenzione alla valutazione di espressioni. Procederemo sempre semplificando sottoespressioni e sostituendo a quest'ultime il loro valore, secondo i seguenti principi.

- ▷ Valutare l'applicazione di funzioni, immediatamente se la funzione è predefinita, o usando l'espressione che ne costituisce il corpo se essa è definita mediante =. In questo caso, rimpiazzare i parametri formali nel corpo con gli argomenti.
- ▷ L'ordine di valutazione delle sottoespressoni di una espressione è irrilevante, salvo che questa sia un `if-then-else`. In quest'ultimo caso, valuteremo completamente la guardia: il valore dell'espressione sarà quindi ottenuto valutando l'espressione nel ramo `then` se essa è vera, quella nel ramo `else` se essa è falsa.

Per illustrare le regole cominciamo con la semplice derivazione del valore dell'espressione `sestuplo 2` in cui indicheremo la motivazione di ciascun passaggio. Nel seguito tralasceremo tali dettagli, e anche alcuni dei passaggi meno significativi, lasciando al lettore il compito di completare e giustificare le catene di uguaglianze.

```
sestuplo 2 = doppio(triplo 2)      (definizione di sestuplo)
           = triplo 2 + triplo 2   (definizione di doppio)
           = 3*2 + 3*2             (definizione di triplo)
           = 6 + 6                 (definizione di *)
           = 12                    (definizione di +)
```

Semplice, no? Proseguiamo con un esempio di valutazione di espressioni condizionali.

```
min(3,6) = if 3 < 6 then 3 else 6 (definizione di min)
         = if true then 3 else 6  (siccome 3 < 6)
         = 3                      (definizione di if-then-else)
```

Ovviamente, avendo a che fare con definizioni ricorsive, e cioè con *funzioni parziali*, non è detto che, in generale, il procedimento valutazione termini.

Useremo un'altro tipo di dato: le ***liste*** di numeri naturali. Una lista è struttura lineare consistente di una sequenza *finita* (ma di lunghezza arbitraria) di naturali. Ad esempio

`[]`   e   `[1,2,3]`   e   `[2,6,18,2]`

sono liste, dove `[]` è la ***lista vuota***. Esistono tre importanti funzioni su liste: `null`, che restituisce `true` se e solo se il suo argomento è la lista vuota, `head` che restituisce il primo elemento della sua lista argomento, e `tail` che restituisce il

---

[2]Si osservi che `h(m,n) = E` e `k m n = E` definiscono due funzioni formalmente diverse e non immediatamente interscambiabili: h ha tipo $a \times b \to c$, un solo argomento di tipo coppia, mentre k ha tipo $a \to b \to c$, ed prende quindi i due argomenti l'uno dietro l'altro. Ad ogni modo, le due funzioni sono strettamente correlate. Anzi, sono '*moralmente*' la stessa.

suo argomento dopo aver eliminato da esso il primo elemento. Un esempio di definizione di funzioni su è il seguente:

```
lung xs = if null xs then 0 else 1 + lung(tail xs)
```

Ecco un esempio di valutazione di `lung`.

```
lung [1,2,3] = 1+lung [2,3]
             = 1+(1+lung [3])
             = 1+(1+(1+lung []))
             = 1+(1+(1+0))
             = 1+(1+1)
             = 1+2
             = 3
```

Completiamo il nostro linguaggio introducendo una notazione ispirata alla definizione per casi usuale in matematica. Ad esempio, per definire le funzioni logiche `and` e `or` potremmo procedere per casi sulla forma del primo argomento.

```
and False y = False
and True  y = y

or True  y  = True
or False y  = y
```

Quindi, invece che definire le funzioni con una unica espressione parametrica in due generiche variabili `x` e `y`), abbiamo usato espressioni distinte (parametriche nella sola variabile `y`) nei due casi discriminati dal valore del primo argomento (una delle constanti `True` o `False`).

Per valutare l'applicazione di una funzione `f` così definita ad un argomento `E` si procede sostanzialmente come illustrato in precedenza, salvo valutare preliminarmente l'espressione `E` per quanto necessario a confrontare il suo valore con i ***patterns*** (gli schemi) nella definizione. Nel far questo si considerano le varie espressioni che definiscono la funzione procedendo tassativamente *dall'alto verso il basso*. Infine il valore della applicazione è ottenuto valutando l'espressione corrispondente al *primo pattern* che corrisponde all'argomento, usando per i suoi parametri i valori individuati da tale corrispondenza.

Nel caso dei naturali, dunque, potremo scrivere la seguente definizione per la funzione fattoriale.

```
fatt 0 = 1
fatt n = n * fatt(n-1)
```

Il primo pattern è la costante `0`, che può corrispondere solo ad argomenti il cui valore è *0*; il secondo è l'identificatore `n`, che corrisponde a qualsiasi valore. Applicando questa definizione si cade nel secondo caso per *tutti* e *soli* gli argomenti con valore non nullo.

Per quanto riguarda le liste, assumeremo di poter scrivere $x : xs$ per la lista ys la cui `head` (*testa*) è x e la cui `tail` (*coda*) è xs, ed useremo (spesso, ma *non* esclusivamente) il seguente schema di definizione per casi:

```
flist  []    = E1
flist(x:xs)  = E2
```

Ad esempio, la seguente definizione di `lung` è esattamente equivalente a quella data più sopra.

```
lung  []    = 0
lung(x:xs) = 1 + lung xs
```

Concludiamo rilevando due limitazioni nell'uso di patterns in linguaggi di programmazione: è illegale che un identificatore appaia più volte in un pattern, come ad esempio in `fun f x x = ...`, o che vi appaiano funzioni (salvo `:`), come ad esempio in `fun f(x+1) = ...`.

2.1. Esercizi.

ESERCIZIO 2.1. Si definisca ricorsivamente la funzione $f(n) = \sum_{i=0}^{n} i$, $n \in \mathbb{N}$.

ESERCIZIO 2.2. Si definisca una funzione ricorsiva $h\colon \mathbb{N} \times \mathbb{N} \to \{\mathit{false}, \mathit{true}\}$ tale che $h(y, x) \equiv \mathit{divide}(y, x)$.

ESERCIZIO 2.3. Si definisca per ricorsione la funzione $\mathit{conta}\colon \mathbb{N} \times \mathit{List} \to \mathbb{N}$, dove $\mathit{conta}(x, xs)$ è il il numero di occorrenze di x in xs.

ESERCIZIO 2.4. Si definisca ricorsivamente la funzione $\mathit{take}\colon \mathbb{N} \times \mathit{List} \to \mathit{List}$, dove $\mathit{take}(n, xs)$ è la lista formata dai primi $\mathit{min}(n, |xs|)$ elementi di xs.
Analogamente per la funzione $\mathit{drop}\colon \mathbb{N} \times \mathit{List} \to \mathit{List}$, dove $\mathit{drop}(n, xs)$ è la lista ottenuta da xs rimuovendone i primi $\mathit{min}(n, |xs|)$ elementi.

ESERCIZIO 2.5. Si definisca la funzione $\mathit{alterna}\colon \mathit{List} \to \mathit{List}$, dove $\mathit{alterna}(xs)$ è la lista formata dagli elementi di posizione dispari in xs.

ESERCIZIO 2.6. Si definisca una funzione $\mathit{revit}\colon \mathit{List} \times \mathit{List} \to \mathit{List}$ tale che

$$\begin{aligned} \mathit{revit}([\,], xs) &= xs \\ \mathit{revit}(xs, [\,]) &= \mathit{reverse}(xs) \end{aligned}$$

dove *reverse* è la funzione che inverte una lista.

ESERCIZIO 2.7. Si definisca una funzione $\mathit{merge}\colon \mathit{List} \times \mathit{List} \to \mathit{List}$, dove $\mathit{merge}(xs, ys)$ è una fusione zs di xs e ys tale che, se xs ed ys sono ordinate, allora tale è zs.

ESERCIZIO 2.8. Si definisca una funzione $\mathit{insert}\colon \mathbb{N} \times \mathit{List} \to \mathit{List}$ tale che $\mathit{insert}(x, xs)$ sia una lista risultante dalla inserzione di x in xs in modo che, se xs è ordinata, allora tale è $\mathit{insert}(x, xs)$.
Usando *insert*, si definisca la funzione $\mathit{insertsort}\colon \mathit{List} \to \mathit{List}$ che ordina la sua lista argomento. (Suggerimento: si definisca prima $\mathit{insertq}\colon \mathit{List} \times \mathit{List} \to \mathit{List}$ in modo che $\mathit{insertq}(xs, ys)$ inserisca gli elementi di xs in ys ad uno ad uno usando *insert*. Si prenda poi $\mathit{insertsort(xs)} = \mathit{insertq}(xs, [\,])$.)

3. Verifica di Funzioni Ricorsive mediante Induzione

Nei linguaggi imperativi la correttezza di comandi rispetto ad una specifica data può essere valutata mediante *triple di Hoare*: se C è un comando e P e Q sono descrizioni di stati che specificano il comportamento atteso da C, il problema si riduce a verificare la validità della tripla

$$\{P\}C\{Q\},$$

ovvero che da P, eseguendo C segue Q. Nel seguito di queste note definiremo un metodo per verificare la corretta definizione di funzioni ricorsive. Considerando una semplice notazione funzionale (come quella descritta sopra) come base per la programmazione di funzioni, avremo quindi bisogno di sviluppare tecniche per verificare che una data definizione ricorsiva di una funzione f soddisfi una certa proprietà ϕ, per ogni possibile ambiente di calcolo, ovvero per ogni valore possibile per i suoi argomenti. Trattandosi di funzioni, ovvero di applicazioni da un dominio ad un codominio, possiamo affermare che ciò di cui avremo bisogno sarà essenzialmente poter verificare che, per una $f\colon D \to C$, se $x \in D$ soddisfa una proprietà ψ su D, allora $f(x)$ soddisfa una proprietà ϕ su C. Un caso particolare molto rilevante sarà quello in cui si voglia verificare che una data definizione ricorsiva di f corrisponda ad una funzione matematica nota g, ovvero (prendendo $\psi(x) \equiv \mathit{true}$ e $\phi(x) \equiv f(x) = g(x)$, che

$$\forall x \in D.\ f(x) = g(x).$$

La validità della formula precedente implica, in genere, una proprietà a cui faremo nel seguito molta attenzione e che, tipicamente, vorremmo garantire su tutte le funzioni che definiamo all'interno dei nostri programmi: la ***totalità*** di f nel caso g sia una funzione che sappiamo essere totale.

Si osservi che, per dimostrare proprietà di funzioni ***non*** sempre possiamo fare appello ad una semantica operazionale per funzioni, ovvero ad un procedimento costruttivo che ne calcoli il significato per un fissato ambiente di calcolo. Infatti, verificare una proprietà di una funzione calcolandola in ogni ambiente significherebbe, in genere, verificare un numero infinito di asserzioni. Ad esempio, per verificare che f gode della proprietà $\phi(n) \equiv 1+3+\ldots+(2n-1) = n^2 = f(n)$, dovremmo verificarla su ogni elemento del dominio: $\phi(0), \phi(1), \phi(3), \ldots$, il che è chiaramente impraticabile nei termini proposti. Dobbiamo pertanto fare appello ad un metodo formale che ci consenta di dimostrare proprietà di funzioni generalizzando il calcolo (induttivamente) su tutti i possibili argomenti.

Analogamente al caso dei numeri naturali, è il principio di induzione (ben fondata) che fornisce lo strumento principale per questo tipo di analisi. Sia f una funzione da un dominio D ad un codominio C ($f\colon D \to C$), e supponiamo di voler verificare che

$$\forall x \in D.\ f(x) = g(x).$$

Possiamo allora procedere come segue:

- ▹ individuiamo una relazione $\prec$ su D che esprima una relazione di precedenza tra gli elementi di D, in modo che $(D, \prec)$ sia ben fondato;
- ▹ dimostriamo l'asserto precedente per induzione ben fondata, in riferimento alla relazione di precedenza $\prec$ individuata.

Come già accennato, sono due i principali tipi di proprietà di definizioni ricorsive di funzioni cui siamo interessati:

- ▹ la funzione definita è ***totale***;
- ▹ la funzione è ***corretta*** rispetto ad una data specifica.

Come evidenziato precedentemente, in tutti i casi che vedremo, la dimostrazione della seconda proprietà è implicitamente anche una dimostrazione della prima. Più precisamente, vedremo che il semplice fatto che la funzione sia definita mediante una particolare relazione di precedenza *ben fondata* – la ***precedenza indotta*** su D dalla definizione (cf. sezione 4.1) – garantisce che essa sia totale.

ESEMPIO 3.1. Consideriamo la funzione pari data all'inizio del capitolo, e dimostriamone la correttezza utilizzando il principio di induzione ben fondata.

```
pari 0 = True
pari 1 = False
pari n = pari(n-2)
```

Il dominio della funzione è $\mathbb{N}$. Il codominio è l'inseme dei valori booleani $\mathbb{B} = \{false, true\}$. Consideriamo la precedenza $\sqsubset$ su $\mathbb{N}$ dell'esempio 1.19, che, come noto, rende $(\mathbb{N}, \sqsubset)$ un insieme ben fondato.

La proprietà sulla funzione che vogliamo dimostrare è dunque che *pari* corrisponda alla funzione dai naturali ai booleani: $g(n) \equiv n \; mod \; 2 = 0$. Quindi dobbiamo verificare che

$$\forall n \in \mathbb{N}.\; pari(n) \equiv n \; mod \; 2 = 0$$

Caso base: I due elementi minimali sono 0 e 1 (vedi Figura 3).

$pari(0)$	$pari(1)$
$\equiv$ { definizione di *pari* }	$\equiv$ { definizione di *pari* }
$true$	$false$
$\equiv$ { identità }	$\equiv$ { identità }
$0 = 0$	$1 = 0$
$\equiv$ { $0 \; mod \; 2 = 0$ }	$\equiv$ { $1 \; mod \; 2 = 1$ }
$0 \; mod \; 2 = 0$	$1 \; mod \; 2 = 0$

Caso induttivo: Sia ora $n \in \mathbb{N}$ un elemento non minimale rispetto a $\sqsubset$, ovvero sia $n \geq 2$. L'ipotesi induttiva, che indichiamo con $IND(n)$ è in questo caso:

$$IND(n) \equiv \forall m \sqsubset n.\; pari(m) \equiv m \; mod \; 2 = 0$$

Dunque:

$pari(n)$

$\equiv$ { **Ip:** $n \geq 2$, definizione di *pari* }

$pari(n-2)$

$\equiv$ { **Ip-Ind:** $IND(n)$, $(n-2) \sqsubset n$ }

$(n-2) \; mod \; 2 = 0$

$\equiv$ { $n \geq 2 \Rightarrow (n-2) \; mod \; 2 = 0 \equiv n \; mod \; 2 = 0$ }

$n \; mod \; 2 = 0$

Pertanto, per induzione ben fondata, (la definizione de) la funzione *pari* implementa correttamente la funzione matematica $g(n) \equiv n \; mod \; 2 = 0$. Essendo g totale, anche *pari* lo è.

ESEMPIO 3.2. Consideriamo ora l'ordinamento $<$ sui numeri naturali. Come sappiamo $(\mathbb{N},<)$ è ben fondato. Dimostriamo la correttezza di *pari* per induzione ben fondata in riferimento a tale precedenza. Questo, come noto, corrisponde a dimostrare la correttezza di *pari* per induzione completa, poiché per $(\mathbb{N},<)$, l'induzione completa è un caso particolare dell'induzione ben fondata.

Caso base: L'unico elemento minimale rispetto a $<$ è 0, e dunque abbiamo una dimostrazione analoga a quella dell'esempio 3.1e per il corrispondente caso base.

Caso induttivo: Sia ora n un elemento non minimale rispetto a $<$, ovvero sia $n > 0$. Osserviamo immediatamente che la dimostrazione del caso induttivo deve essere fatta per casi, a seconda che l'elemento n considerato sia 1 oppure sia un numero maggiore di 1. Ciò è essenziale nel primo passo della dimostrazione per poter rimpiazzare $pari(n)$ con il suo valore, dato dalla definizione di *pari* stessa.

Una volta osservato questo, le due dimostrazioni corrispondono, rispettivamente, a quella del secondo caso base a quella del caso induttivo nell'esempio 3.1. Si noti, in particolare, come nel caso $n = 1$ non viene utilizzata l'ipotesi induttiva.

La differenza principale tra la dimostrazione dell'esempio 3.1 e quella dell'esempio 3.2 è che nella prima, la precedenza considerata fa sì che i casi base corrispondano esattamente ai casi in cui la funzione è definita immediatamente, mentre nella seconda ciò non succede.

Questo è un esempio di una situazione generale: nella dimostrazione di proprietà di funzioni ricorsive, è molto spesso decisamente conveniente considerare una precedenza sul dominio della funzione che corrisponda esattamente alla struttura della definizione della funzione stessa. In questo modo, nella dimostrazione della proprietà, il caso base ed il caso induttivo (come nell'esempio 3.1) corrisponderanno in modo diretto ai vari casi della definizione della funzione, rendendo più semplici e meno soggette ad errori le dimostrazione per induzione ben fondata.

Ma come determinare una precedenza sul dominio della funzione che rispecchi la definizione della funzione stessa? A questa domanda troveremo una risposta nella seguente sezione mediante il già citato concetto di ***precedenza indotta*** sul dominio dalla definizione ricorsiva.

3.1. Esercizi.

ESERCIZIO 3.1. Si dimostri che la definizioni delle funzione date in risposta agli esercizi 2.1–2.4 sono corrette rispetto le loro specifiche.

ESERCIZIO 3.2. Si dimostri che la definizione della funzione *alterna* data in risposta all'esercizio 2.5 è corretta. (Suggerimento: si proceda per induzione ben fondata usando la relazione $xs \prec x : y : xs$.)

ESERCIZIO 3.3. Si dimostri che la funzione *reverse* definita da

```
reverse   []    = []
reverse(x:xs) = reverse(xs)++[x]
```

dove ++ è la funzione che concatena due liste una di seguita all'altra, inverte la lista suo argomento. (Suggerimento: si dimostri, ad esempio, che

$$\forall xs, \forall 1 \leq i \leq |xs|.\ nth(xs, i) = nth(reverse(xs), |xs| - i + 1),$$

dove *nth*(xs,i) restituisce l'*i*esimo elemento di *xs*, procedendo, ad esempio, per induzione ben fondata su $(xs, i) \prec (x : xs, i + 1)$.)

ESERCIZIO 3.4. Si dimostri che $reverse(reverse(xs)) = xs$. (Suggerimento: si proceda per induzione strutturale sulle liste,[3] dopo aver dimostrato il seguente lemma: $reverse(xs +\!+\, ys) = reverse(ys) +\!+\, reverse(xs)$.)

ESERCIZIO 3.5. Si dimostri che la definizione della funzione *revit* data in risposta all'esercizio 2.6 è corretta. (Suggerimento: si proceda per induzione strutturale, dopo aver provato il lemma: $revit(xs, ys) +\!+\, zs = revit(xs, ys +\!+\, zs)$.)

ESERCIZIO 3.6. Si consideri la seguente possibile definizione di $+\!+$.

```
 []    ++ ys = ys
(x:xs) ++ ys = x:(xs ++ ys)
```

Si dimostri che ++ è associativa, ovvero che $(xs +\!+\, ys) +\!+\, zs = xs +\!+\, (ys +\!+\, zs)$. (Suggerimento: si proceda per induzione strutturale su *xs*.)

ESERCIZIO 3.7. Si consideri la definizione

```
last = head(drop(lung xs - 1,xs))
```

Si dimostri che $last(xs +\!+\, [x]) = x$.

ESERCIZIO 3.8. Si consideri la definizione

```
ricerca y   []   = False
ricerca y (x:xs) = if x == y then True
                       else ricerca y xs
```

Si dimostri che $ricerca\ x\ xs \equiv x \in xs$.

ESERCIZIO 3.9. Si consideri la seguente definizione di una funzione che determina il massimo elemento di una lista non vuota.

```
maxl [x] = x
maxl (x:y:xs) = if x <= y then maxl(y:xs)
                  else maxl(x:xs)
```

Si dimostri che la definizione di *maxl* è corretta. (Suggerimento: si osservi che non si può procedere per induzione strutturale, visto che nel ramo else si usa $x : xs$ che *non* è un 'sottotermine' di $x : y : xs$. Si proceda allora per induzione (ben fondata) sulla lunghezza della lista argomento.)

ESERCIZIO 3.10. Si dimostri che la funzione *merge* definita in risposta all'esercizio 2.7 è corretta, cioè che $ORD(xs) \wedge ORD(ys) \Rightarrow ORD(merge(xs, ys))$. (Suggerimento: anche in questo caso l'induzione strutturale non sarà sufficiente. Si consideri la relazione di precedenza

$$(xs, ys) \prec (xs', ys') \quad \text{se e solo se} \quad xs' = x : xs \vee ys' = y : ys$$

e si proceda per induzione ben fondata.)

ESERCIZIO 3.11. Si dimostri che la funzione *insert* definita in risposta all'esercizio 2.8 è corretta, cioè che $ORD(xs) \Rightarrow ORD(insert(x, xs))$. Usando tale risultato, si dimostri la correttezza della funzione *insertsort*, cioè che

$$\forall xs.\ ORD(insertsort(xs)).$$

[3] Cioè per induzione ben fondata su $xs \prec x : xs$.

(Suggerimento: si noti che la formula precedente non è sufficiente: è anche necessario mostrare che *insertsort*(xs) è una *permutazione* di xs. Questo può ottenersi facilmente mostrando che

$$\forall x.\ conta(x, insertq(xs, ys)) = conta(x, xs) + conta(x, ys),$$

da cui segue che $\forall x.\ conta(x, xs) = conta(x, insertsort(xs))$.)

4. Progetto di Funzioni Ricorsive

In quest'ultima sezione studieremo come applicare le tecniche appena viste come ausilio nella fase di progetto di funzioni ricorsive. Esattamente come nel caso della programmazione imperativa in cui lo sviluppo dei programmi (in particolare di cicli **while**) procede di pari passo con l'analisi delle corrispondenti *weakest preconditions*, nel caso della programmazione di funzioni ricorsive, il progetto procederà mano nella mano con l'analisi del dominio della funzione mediante relazioni di precedenza e principio di induzione ben fondata. Una parte essenziale del nostro lavoro sarà dunque riuscire ad associare un'opportuna relazione di precedenza ad una funzione e, viceversa, una funzione ad una relazione di precedenza.

Come possiamo ragionare in maniera *sistematica* per derivare funzioni ricorsive, ovvero programmi nella semplice notazione funzionale che stiamo adoperando? L'idea, come già accennato, è aiutarsi strutturando opportunamente il dominio della funzione mediante una relazione di precedenza. Tenteremo cioè, conoscendo la funzione (matematica) g da definire (*programmare*), di 'inventare' una relazione di precedenza sul dominio di g, ovvero sul dominio della definizione che vogliamo individuare. Se avremo successo in questa fase, potremo interpretare tale relazione come una definizione implicita della funzione incognita f, ottenendo quindi una chiave per la soluzione del problema.

Al fine di rendere più chiara la relazione esistente tra relazioni di precedenza e funzioni ricorsive, vediamo ora come associare un relazione di precedenza ad una funzione ricorsiva data. Ragionando tentando di invertire in maniera opportuna questo procedimento, è la base del ***progetto di funzioni ricorsive***.

4.1. Relazioni di Precedenza Indotte da Definizioni Ricorsive. Gli esempi visti finora mettono in luce il fatto che una buona definizione ricorsiva di una funzione deve essere tale che:

1. su alcuni argomenti la funzione sia definita immediatamente;
2. sui restanti argomenti la funzione possa essere ricondotta ad argomenti 'più semplici'.

L'apparato formale sviluppato nei precedenti paragrafi ci mette a disposizione una serie di strumenti per formalizzare il concetto di argomento 'più semplice':

> *Un argomento è 'più semplice' di un altro, se lo precede in una relazione di precedenza <u>ben fondata</u> sul <u>dominio</u> della funzione.*

DEFINIZIONE 4.1 *(Precedenza Indotta).* Sia f una funzione da D in C definita possibilmente per ricorsione. La ***relazione di precedenza indotta*** su D dalla definizione di f è la relazione $\sqsubset$ su D tale che

$$\forall x, y \in D.\ x \sqsubset y \quad \textit{se e solo se} \quad f(y)\ \textit{è definita in termini di}\ f(x).$$

Si noti che, in genere, la relazione di precedenza indotta da una funzione ricorsiva sul suo dominio ***non*** è un ordinamento parziale, in quanto non è né riflessiva né transitiva. Più importante dal nostro attuale punto di vista è che $\sqsubset$ ***non*** è necessariamente ***ben fondata***. Sarà quest'ultima proprietà di $\sqsubset$ a distinguere le *buone* definizioni da quelle cattive. Tale proprietà – che di fronte ad ogni definizione ricorsiva sarà nostro primo compito verificare – come infatti noto dalle precedenti sezioni, ci permetterà di applicare il principio di induzione ben fondata per ragionare sulla funzione f.

È facile osservare che, nel caso $(D, \sqsubset)$ sia ben fondato, applicando ad esso il principio di induzione ben fondata per dimostrare una certa proprietà ϕ, il caso base corrisponderà esattamente alla dimostrazione di ϕ per tutti gli argomenti su cui la funzione è definita *direttamente*, mentre il caso induttivo corrisponderà alla dimostrazione di ϕ per gli argomenti su cui la funzione è definita in per *ricorsione*.

Come abbiamo già avuto modo di accennare, il fatto che la relazione di precedenza indotta da una definizione di f sia ben fondata garantisce che f sia una funzione totale, come enunciato più precisamente dal seguente teorema.

TEOREMA 4.2 *(Terminazione)*. *Sia $f\colon C \to D$ una funzione definita per ricorsione e sia $\sqsubset$ la precedenza indotta da tale definizione. Allora*

*f è **totale** se e solo se $\sqsubset$ è **ben fondata** e f è definita su tutti i minimali*

ESEMPIO 4.3. Si consideri la seguente definizione di f.

```
f x = f(x+1) + 1
```

Evidentemente il diagramma della relazione di precedenza $\prec$ indotta su $\mathbb{N}$ da tale definizione è il *secondo* diagramma di Figura 1. Altrettanto ovviamente, $(\mathbb{N}, \prec)$ *non* è ben fondato, ed infatti f è indefinita su ogni $n \in \mathbb{N}$.

Analizziamo ora una variazione della definizione di *pari* in cui omettiamo la definizione del caso $n = 1$.

```
pari 0 = True
pari n = pari(n-2)
```

Se consideriamo *pari* una funzione dal dominio degli *interi*, il diagramma della precedenza indotta dalla definizione è quello illustrato in Figura 4; $\sqsubset$ *non* è ben fondata e la funzione definita *non* è totale: essa è definita solo sui numeri interi pari e positivi (perché definita sul corrispondente minimale 0).

Cosa succederebbe considerando, come al solito, *pari* una funzione sui naturali? La precedenza indotta è in questo caso quella mostrata in Figura 3. Ne risulta che $(\mathbb{N}, \sqsubset)$ è ben fondato. Si osservi, però, che ora *pari non* è definita esplicitamente sull'elemento minimale 1. Dunque, *pari non* è totale neanche come funzione sui naturali: è indefinita sui numeri dispari, mentre è definita sui pari (perché è definita sul corrisponde elemento minimale 0).

ESEMPIO 4.4. Consideriamo di nuovo la funzione `pari` dell'esempio 3.1. La relazione di precedenza indotta dalla funzione è la seguente:

$$\forall x, y \in \mathbb{N}.\ x \sqsubset y \equiv y \neq 0 \wedge y \neq 1 \wedge x = y - 2$$

che può essere anche scritta come:

$$\forall x, y \in \mathbb{N}.\ x \sqsubset y \equiv y > 1 \wedge x = y - 2.$$

$$
\begin{array}{cc}
\vdots & \vdots \\
| & | \\
4 & 5 \\
| & | \\
2 & 3 \\
| & | \\
0 & 1 \\
 & | \\
-2 & -3 \\
| & | \\
-4 & -5 \\
| & | \\
\vdots & \vdots
\end{array}
$$

FIGURA 4. Diagramma di precedenza di $(\mathbb{N}, \sqsubset)$

Si noti come la condizione $y \neq 0 \wedge y \neq 1$ escluda tutti i casi in cui $f(y)$ è definita immediatamente. Abbiamo dunque il seguente diagramma:

$$
\begin{array}{cc}
\vdots & \vdots \\
| & | \\
6 & 7 \\
| & | \\
4 & 5 \\
| & | \\
2 & 3 \\
| & | \\
0 & 1
\end{array}
$$

Si noti come, nella relazione di precedenza indotta, gli elementi minimali coincidano proprio con gli elementi del dominio su cui la funzione è definita immediatamente.

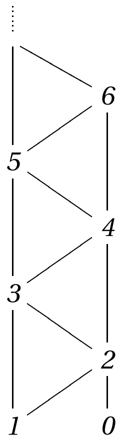
ESEMPIO 4.5. Consideriamo la funzione che definisce i numeri di Fibonacci introdotta all'inizio di queste note:

```
fib 0 = 0
fib 1 = 1
fib n = fib(n-1) + fib(n-2)
```

In questo caso, la relazione di precedenza indotta sul dominio della funzione $\mathbb{N}$ deve esprimere il fatto che i predecessori di un numero naturale m non minimale (ovvero $m > 1$) sono sia $m - 1$ che $m - 2$. Formalmente, se indichiamo con $\sqsubset$ questa relazione, abbiamo:

$$\forall n, m \in \mathbb{N}.\ n \sqsubset m \equiv m > 1 \wedge (n = m - 1 \vee n = m - 2).$$

Il diagramma di precedenza corrispondente è il seguente:



da cui, in base al teorema 4.2, possiamo concludere che *fib* è ben definita.

Vediamo ora un esempio più complesso.

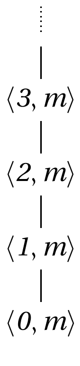
ESEMPIO 4.6. Consideriamo la seguente definizione ricorsiva della funzione *times* da coppie di naturali in naturali. Il dominio della funzione sarà dunque $\mathbb{N} \times \mathbb{N}$.

```
times(0,y) = 0
times(x,y) = times(x-1, y) + y
```

La relazione di precedenza indotta sul dominio della funzione $\mathbb{N} \times \mathbb{N}$ da questa definizione ricorsiva è data dalla seguente definizione:

$$\forall n, m, n', m' \in \mathbb{N}.\ \langle n, m\rangle \sqsubset \langle n', m'\rangle \equiv n' > 0 \land n = n' - 1 \land m = m'.$$

Il diagramma di precedenza di questa relazione può essere così rappresentato:



dove m è un qualunque numero naturale. Si tratterà quindi di un insieme infinito di catene ascendenti del tipo mostrato in figura, al variare di m in $\mathbb{N}$. Tale diagramma mostra che:

- ▷ gli elementi minimali della relazione di precedenza sono infiniti (tutte e sole le coppie $\langle n, m\rangle$, in cui $n = 0$);
- ▷ il dominio della funzione è un insieme ben fondato.

Si osservi che essendo *times* definita esplicitamente su tutti gli elementi minimali, possiamo appellarci al teorema 4.2 per concludere che essa è ben definita, ovvero totale. Possiamo inoltre dimostrare, per induzione ben fondata su $(\mathbb{N} \times \mathbb{N}, \sqsubset)$, che *times* soddisfa la sua specifica, cioè vale la seguente proprietà:

$$\forall n, m \in \mathbb{N}.\ times(n, m) = n \cdot m.$$

Caso base: Sia m un qualunque numero naturale. Abbiamo:

$$\begin{array}{ll} & times(0, m) \\ = & \quad \{ \text{definizione di } times \} \\ & 0 \\ = & \quad \{ \forall x.\ 0 \cdot x = 0 \} \\ & 0 \cdot m \end{array}$$

Si noti che, essendo m arbitrario, possiamo concludere che la proprietà richiesta vale per tutte le coppie minimali.

Caso induttivo: Sia ora $\langle n, m \rangle$ una coppia non minimale. L'ipotesi induttiva è allora:

$$IND(n, m) \equiv \forall n', m' \in \mathbb{N}.\ \langle n', m' \rangle \sqsubset \langle n, m \rangle \Rightarrow times(n', m') = n' \cdot m'.$$

Abbiamo quindi:

$$\begin{array}{ll} & times(n, m) \\ = & \quad \{ \textbf{Ip:}\ \langle n, m \rangle \text{ non minimale, definizione di } times \} \\ & times(n - 1, m) + m \\ = & \quad \{ \textbf{Ip-Ind:}\ IND(n, m),\ \langle n - 1, m \rangle \sqsubset \langle n, m \rangle \} \\ & (n - 1) \cdot m + m \\ = & \quad \{ \text{aritmetica} \} \\ & n \cdot m + m - m \\ = & \quad \{ \text{aritmetica} \} \\ & n \cdot m \end{array}$$

Si noti che, nel passo di dimostrazione che utilizza l'ipotesi induttiva, il fatto che $\langle n-1, m \rangle \sqsubset \langle n, m \rangle$ è garantito dal fatto che stiamo considerando la relazione di precedenza indotta.

Come già accennato in precedenza, la totalità di *times* può essere anche derivata dalla precedente dimostrazione: il fatto che $\forall n, m \in \mathbb{N},\ times(n, m) = n \cdot m$, implica che $times(n, m)$ ha un valore definito ($n \cdot m$ appunto), qualunque siano n ed m.

Vediamo, per concludere, un esempio di funzione che non è totale. Essa è una banale variazione del precedente analogo esempio.

ESEMPIO 4.7. Sia *foo* la funzione ricorsiva dai naturali nei naturali definita come segue:

```
foo 0 = 0
foo n = n + f(n+1)
```

La relazione di precedenza indotta da tale funzione $\sqsubset$ è definita come segue:

$$\forall n, m \in \mathbb{N}.\ n \sqsubset m \equiv m \geq 1 \wedge m = n + 1.$$

Il diagramma di precedenza è ancora il secondo di Figura 1, e cioè il seguente:

$$\begin{array}{c} 0 \\ | \\ 1 \\ | \\ 2 \\ | \\ 3 \\ | \\ \vdots \end{array}$$

Questo diagramma mostra come la precedenza indotta non renda il dominio della funzione un insieme ben fondato (esiste una catena decrescente infinita, data dall'insieme $\mathbb{N}$ stesso). La funzione è chiaramente non definita per qualsiasi valore (naturale) passato come parametro.

Si noti che, nel diagramma di precedenza corrispondente alla relazione di precedenza indotta da una definizione ricorsiva per la funzione f, le catene che si ottengono seguendo il diagramma dall'alto verso il basso, e a partire da un elemento x del dominio, corrispondono agli elementi del dominio sui quali deve essere calcolato (ricorsivamente) il valore di f per ottenere il valore di $f(x)$. Di qui la necessità che il dominio della funzione, rispetto alla relazione di precedenza indotta, sia un insieme ben fondato. Ad esempio, nel diagramma dell'esempio 4.6, la catena che si ottiene a partire dall'elemento $\langle 3, 5\rangle$ è la seguente:

$$\begin{array}{c} \langle 3, 5\rangle \\ | \\ \langle 2, 5\rangle \\ | \\ \langle 1, 5\rangle \\ | \\ \langle 0, 5\rangle \end{array}$$

e ciò corrisponde al seguente calcolo di $\mathit{times}(3, 5)$:

$$\begin{aligned} \mathit{times}(3,5) &= \mathit{times}(2,5) + 5 \\ &= \mathit{times}(1,5) + 5 + 5 \\ &= \mathit{times}(0,5) + 5 + 5 + 5 \\ &= 0 + 5 + 5 + 5 \\ &= 15. \end{aligned}$$

D'altra parte, nel diagramma dell'esempio 4.7, la catena (discendente infinita) che si ottiene a partire dall'elemento 3 è la seguente:

$$\begin{array}{c} 3 \\ | \\ 4 \\ | \\ 5 \\ | \\ \vdots \end{array}$$

Essa corrisponde al seguente calcolo di *foo*(*3*):

$$foo(3) = 3 + foo(4) = 3 + 4 + foo(5) = \cdots$$

che come si vede non è definito.

4.2. Dalla Precedenza alle Funziona Ricorsiva: il Progetto di Funzioni. Il principio di induzione ben fondata ci suggerisce anche un modo per ragionare metodicamente nel progettare una funzione ricorsiva che soddisfi una certa proprietà (tipicamente che calcoli una data funzione). L'idea è quella di determinare gli argomenti su cui la funzione può essere definita in modo immediato, e poi, per tutti gli altri argomenti, definire la funzione in termini del suo valore su argomenti più semplici (ricordando che ciò dovrà essere giustificato una relazione di precedenza indotta che renda il dominio della funzione ben fondato). Come nel caso dei linguaggi imperativi, per risolvere un problema del tipo $\{P\}?\{Q\}$, possiamo sia 'inventare' un comando C che pensiamo soddisfi la specifica, e poi verificare che $\{P\}C\{Q\}$ sia valida. Ovviamente, 'inventare C' significa cercare di derivarlo il più sistematicamente possibile dai dati in nostro possesso, ad esempio dalla precondizione e postcondizione.

Analogamente, nel caso del progetto di funzioni ricorsive, si tratta dunque di determinare f tale che $f = g$ per una data funzione matematica g che si vuole 'programmare'. L'unico dato in nostro possesso è la proprietà che la funzione deve soddisfare in ogni ambiente di calcolo: essere equivalente ad una funzione nota g. Vediamo, attraverso alcuni esempi, come derivare ulteriori informazioni sulla struttura della funzione f da progettare, analizzando le proprietà di g.

ESEMPIO 4.8. Si vuole definire una funzione ricorsiva $gte(x, y)$ da coppie di naturali in booleani che sia equivalente alle seguente funzione $g(x, y) \equiv x \geq y$, ovvero che goda della seguente proprietà:

$$\forall x, y \in \mathbb{N}.\ \ gte\,(x, y) \equiv x \geq y,$$

supponendo di poter utilizzare solo le operazioni di decremento unitario e di confronto con *0*. Le seguenti ben note proprietà della relazione $\geq$ tra numeri naturali suggeriscono quali devono essere i casi base ed i casi ricorsivi nella definizione della funzione:

- ▹ $\forall x \in \mathbb{N}.\ x \geq 0$;
- ▹ $\forall x \in \mathbb{N}.\ x \neq 0 \Rightarrow x > 0)$;
- ▹ $\forall x, y \in \mathbb{N}.\ x \neq 0 \wedge y \neq 0 \Rightarrow x \geq y \equiv x - 1 \geq y - 1.$

Osservando che ciascuna coppia di naturali cade in uno dei casi precedenti, che sembrano definire una precedenza ben fondata su di essi, possiamo dunque tentativamente definire $gte(x, y)$ come segue:

```
gte(x,0) = True
gte(0,y) = False
gte(x,y) = gte(x-1,y-1)
```

Si noti come, nella definizione di $gte(x, y)$ si siano utilizzate, come richiesto, solo le operazioni di confronto con *0* ($y = 0$, $x = 0$ e $y \neq 0$ nei primi due casi) e di decremento unitario ($x - 1$ e $y - 1$ nel caso ricorsivo).

Dobbiamo ora verificare di aver 'intuito' la definizione di *gte* correttamente. La relazione di precedenza indotta dalla definizione di $gte(x, y)$ sul suo dominio

$(\mathbb{N} \times \mathbb{N})$ è la seguente relazione tra coppie di naturali:

$$\forall x, y, x', y' \in \mathbb{N}.\ \langle x, y\rangle \sqsubset \langle x', y'\rangle \equiv x \neq 0 \wedge y \neq 0 \wedge x = x' - 1 \wedge y = y' - 1.$$

Il diagramma di precedenza corrispondente può essere disegnato come segue:

$$\begin{array}{ccc}
\vdots & \qquad & \vdots \\
| & & | \\
\langle k, y+k\rangle & & \langle x+k, k\rangle \\
| & & | \\
\vdots & & \vdots \\
| & & | \\
\langle 2, y+2\rangle & & \langle x+2, 2\rangle \\
| & & | \\
\langle 1, y+1\rangle & & \langle x+1, 1\rangle \\
| & & | \\
\langle 0, y\rangle & & \langle x, 0\rangle
\end{array}$$

Abbiamo dunque un insieme infinito di catene ascendenti al variare di $x, y \in \mathbb{N}$. Tale diagramma mostra che la relazione di precedenza indotta dalla funzione rende il dominio della funzione un insieme ben fondato e, al tempo stesso, che gli elementi minimali sono tutte e sole le coppie in cui una delle due componenti è 0. Ciò ci lascia concludere che *gte* è ben definita. La dimostrazione per induzione ben fondata su $\sqsubset$ della correttezza della funzione $gte(x, y)$ rispetto la sua specifica $g(x, y)$ è elementare e lasciata per esercizio.

Il precedente esempio evidenzia come, dalle proprietà di base della funzione che si vuole calcolare, in questo caso $\leq$, si possa ricostruire una funzione ricorsiva che calcoli *gte*. Nel caso di funzioni ricorsive ***lineari***, ovvero contenenti al più una sola chiamata ricorsiva in ogni caso del pattern matching ed in ramo `if-then-else`, è facile fornire una procedura sistematica per costruire la funzione che calcoli una funzione data a partire da una relazione di precedenza sul suo dominio. Chiaramente, non tutte le funzioni ricorsive sono lineari, come ad esempio la funzione fattoriale o di Fibonacci viste sopra.

Supponendo di sapere che la funzione da determinare sia di tipo lineare, e conoscendo sia la funzione g che si vuole 'programmare' che una relazione di precedenza $\sqsubset$ sul dominio, possiamo procedere come segue e determinare una funzione f che abbia quella relazione di precedenza come relazione indotta, e che coincida con g.

Determinazione dei casi base: I casi base si determinano trovando quei valori $m \in D$ tali che $\forall n \in D.\ n \sqsubset m \equiv \mathit{false}$. Per un tale argomento m, si pone $f(m) = g(m)$.

Determinazione dei casi induttivi: Sia $\psi(m)$ una data proprietà sul dominio D della funzione, tale che se $\psi(m)$ è vera, allora m non è un caso base, ovvero non è un elemento minimale rispetto alla relazione di precedenza data, ovvero:

$$\psi(m) \Rightarrow (\exists n \in D.\ n \sqsubset m)$$

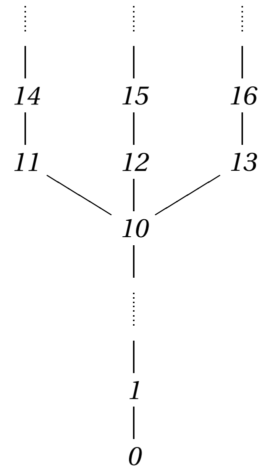
> Se vale che: $n \sqsubset m \wedge \psi(m) \Rightarrow n = E_f$ per una qualche espressione E_f, allora poniamo:
>
> $$f(m) = c \cdot f(E_f) + E$$
>
> dove c ed E sono espressioni incognite. Le espressioni incognite E_f, c ed E saranno determinate verificando per induzione ben fondata che la funzione così trovata soddisfa la specifica.

Il punto chiave è la determinazione della proprietà ψ per il caso induttivo. Questa dipende fortemente dalla struttura della relazione di precedenza data, e da essa va derivata. Il seguente esempio mostra come trovare una funzione, data una proprietà ed una relazione di precedenza sul dominio. È necessario fare particolare attenzione alla determinazione della proprietà ψ, ed al calcolo delle espressioni incognite c e E.

ESEMPIO 4.9. Si determini una funzione ricorsiva lineare f sui naturali tale che $\forall n \in \mathbb{N}.\ f(n) = n + 3$ e tale che la seguente relazione sia indotta da f su $\mathbb{N}$:

$$\begin{aligned} n \sqsubset m \quad \equiv \quad & m \neq 0 \wedge \\ & ((m > 13 \ \wedge\ n = m - 3) \vee \\ & \quad (n = 10 \ \wedge\ (m = 11 \ \vee\ m = 12 \ \vee\ m = 13)) \vee \\ & \quad\quad (m \leq 10 \ \wedge\ n = m - 1)) \end{aligned}$$

Il diagramma di precedenza della relazione $\sqsubset$ è:



Determinamo innanzitutto i casi base, ponendo $g(n) = n + 3$. Dalla relazione di precedenza è chiaro che l'unico caso base, ovvero l'unico naturale m tale che per ogni $n \in \mathbb{N}$, $n \sqsubset m \equiv \mathit{false}$, è $m = 0$. Pertanto si pone $f(0) = 0 + 3 = 3$. Quindi abbiamo parzialmente determinato la funzione, sul singolo (in questo caso) caso base.

```
f 0 = 3
f m = ?
```

Determiniamo ora i casi induttivi. Guardando la relazione di precedenza, si vede chiaramente che ponendo

$$\begin{aligned} \psi_1(m) &\equiv m > 13 \\ \psi_2(m) &\equiv m = 11 \vee m = 12 \vee m = 13 \\ \psi_3(m) &\equiv 0 < m \leq 10 \end{aligned}$$

Si ha che, per $i = 1, 2, 3$:

1. $$\boxed{\begin{array}{ll} & n \sqsubset m \wedge \psi_1(m) \\ \equiv & \\ \ldots & \quad \{ \text{Distr.+ zero} \} \\ \equiv & \\ & (m > 13 \ \wedge \ n = m - 3) \\ \Rightarrow & \quad \{ \text{Sempl.-}\wedge \} \\ & n = m - 3 \end{array}}$$

2. $$\boxed{\begin{array}{ll} & n \sqsubset m \wedge \psi_2(m) \\ \equiv & \\ \ldots & \quad \{ \text{Distr.+ zero} \} \\ \equiv & \\ & (11 \leq m \leq 13 \ \wedge \ n = 10) \\ \Rightarrow & \quad \{ \text{Sempl.-}\wedge \} \\ & n = 10 \end{array}}$$

3. $$\boxed{\begin{array}{ll} & n \sqsubset m \wedge \psi_3(m) \\ \equiv & \\ \ldots & \quad \{ \text{Distr.+ zero} \} \\ \equiv & \\ & (0 < m \leq 10 \ \wedge \ n = m - 1) \\ \Rightarrow & \quad \{ \text{Sempl.-}\wedge \} \\ & n = m - 1 \end{array}}$$

Da cui si ottengono le seguenti espressioni per le chiamate induttive: $E_f^1 = m-3$, $E_f^2 = 10$ e $E_f^3 = m - 1$. Pertanto si ottiene la funzione, con espressioni incognite E, K, ed H:

```
f 0 = 3
f m = if (m<10 || m==10) then a * f(m-1) + E
        else if (m==11 || m==12 || m==13) then b * f 10 + K
          else c * f(m-3) + H
```

La determinazione delle rimanenti espressioni incognite ha luogo applicando l'induzione per determinare la correttezza della funzione sin qui specificata, rispetto alla proprietà data. Dimostriamo per induzione che $\forall n \in \mathbb{N}.\ f(n) = n+3$.

Caso Base: ($n = 0$):

$$f(0) = 3 = 0 + 3.$$

Casi Induttivi:: Abbiamo tre casi induttivi:

1.
$$\begin{array}{ll} & f(n) \\ = & \quad \{\,\textbf{Ip: } n \le 10\,\} \\ & f(n-1)+E \\ = & \quad \{\,\textbf{Ip-Ind: } f(n-1) = (n-1)+3\,\} \\ & a(n-1+3)+E = n+3 \\ = & \quad \{\text{ calcolo }\} \\ & E = n+3-an-2a \end{array}$$

2.
$$\begin{array}{ll} & f(n) \\ = & \quad \{\,\textbf{Ip: } 10 < n \le 13\,\} \\ & f(10)+K \\ = & \quad \{\,\textbf{Ip-Ind: } f(10) = 10+3\,\} \\ & b(10+3)+K = n+3 \\ = & \quad \{\text{ calcolo }\} \\ & K = n+3-13b \end{array}$$

3.
$$\begin{array}{ll} & f(n) \\ = & \quad \{\,\textbf{Ip: } n > 13\,\} \\ & f(n-3)+H \\ = & \quad \{\,\textbf{Ip-Ind: } f(n-3) = (n-3)+3\,\} \\ & c(n-3+3)+H = n+3 \\ = & \quad \{\text{calcolo}\} \\ & H = n+3-cn \end{array}$$

Abbiamo quindi ottenuto: $E = n+3-an-2a$, $K = n+3-13b$ e $H = n+3-cn$. Ovviamente, esistono infinite funzioni che possono verificare la proprietà data, ed al tempo stesso hanno $\sqsubset$ come precedenza indotta. Quelle, per così dire più semplici, si possono ottenere cercando di determinare a, b, c in modo che E, K e H siano espressioni il più semplici possibile. È facile vedere che, per $a = b = c = 1$, si ottiene: $E = 1$, $K = n - 10$ e $H = 3$, da cui una possibile soluzione all'esercizio è data dalla seguente funzione:

```
f 0 = 3
f m = if (m<10 || m==10) then f(m-1) + 1
             else if (m==11 || m==12 || m==13) then f 10 + n - 10
                    else f(m-3) + 3
```

4.3. Esercizi.

Esercizio 4.1. Si consideri la definizione di funzione

```
f 100 = 0
f  n  = if n > 100 then n + f 0 else f(n+1) - 1
```

Dimostrare che f è totale e che coincide con la funzione intera $g(x) = x - 100$.

Esercizio 4.2. Si consideri la definizione di funzione

```
f n = if n > 100 then n - 10 else f(f(x+11))
```

Dimostrare che f è totale e che coincide con la funzione intera

$$g(x) = \begin{cases} x - 10 & \text{se } x > 100 \\ 91 & \text{altrimenti.} \end{cases}$$

ESERCIZIO 4.3. Si definisca una funzione $f\colon \mathbb{N} \to List$ tale che

$$f(n) = [\underbrace{a, \ldots, a}_{n \text{ volte}}, \underbrace{b, \ldots, b}_{n \text{ volte}}]$$

e tale che la precedenza indotta dalla definizione sia $n \sqsubset m \equiv m = n + 2$.

ESERCIZIO 4.4. Si definisca una funzione $f\colon \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ in modo tale che $f(n, m) = 2n + 3m$ e che la precedenza indotta dalla definizione sia

$$\begin{aligned} (n, m) \sqsubset (n', m') \quad \equiv \quad & (n' = n + 1 \land m' = m + 1) \\ & \lor (n' = n + 1 \land m = m' = 0) \\ & \lor (n = n' = 0 \land m' = m + 1) \end{aligned}$$

ESERCIZIO 4.5. Si definisca una funzione $f\colon \mathbb{N} \to \mathbb{N}$ in modo tale che sia $f(n) = 3n+5$ e che la precedenza indotta dalla definizione sia $n \sqsubset m \equiv m = n+3$.

ESERCIZIO 4.6. Si consideri la definizione di funzione

```
f 0 = 1
f 1 = 0
f n = 3 * f(n-2)
```

1. Si determini la precedenza indotta da tale definizione e si mostri che essa è ben fondata disegnandone il diagramma.
2. Si dimostri che f coincide con la funzione

$$g(x) = \begin{cases} 0 & \text{se } \neg pari(x) \\ 3^{x \ div\ 2} & \text{altrimenti.} \end{cases}$$

ESERCIZIO 4.7. Si consideri la funzione

$$f(n) = \begin{cases} n & \text{se } n \text{ è primo} \lor n = 0 \\ f(n-1) + 1 & \text{altrimenti.} \end{cases}$$

1. Si determini la precedenza indotta da tale definizione e si mostri che essa è ben fondata disegnandone il diagramma.
2. Si dimostri che $\forall n \in \mathbb{N}.\ f(n) = n$.
3. Supponendo di disporre di una funzione booleana `primo(n)` si scriva un programma funzionale che realizzi f.

ESERCIZIO 4.8. Si consideri la funzione

```
abs 0 = 0
abs x = if x>0 then 1 + abs(1-x) else 1+ abs(-1-x)
```

Definire la precedenza indotta e disegnarne il diagramma. Dimostrare inoltre per induzione ben fondata che $\forall x \in \mathbb{Z}.\ abs(x) = |x|$.

ESERCIZIO 4.9. Data la definizione

```
f(0,m) = m
f(n,0) = n
f(n,m) = f(n-1,m-1) + 3
```

dare la precedenza indotta e disegnarne il diagramma, e dimostrare per induzione ben fondata che $\forall n, m \in \mathbb{N}.\ f(n, m) = |n - m| + 2 \cdot min(n, m)$.

ESERCIZIO 4.10. Si consideri la seguente successione numerica

$$\begin{array}{ll} f_n = f_{n+1} + 1 & \text{se } 0 \leq n < 100 \\ f_n = n & \text{altrimenti.} \end{array}$$

1. Si determini la precedenza indotta da tale definizione su $\mathbb{N}$ e se ne disegni il diagramma, giustificando il fatto che essa è ben fondata.
2. Si scriva una funzione ricorsiva *foo* che calcoli l'n-esimo termine di tale successione.
3. Si verifichi per induzione che $\forall n \in \mathbb{N}.\ foo(n) = 100 + |100 - n|$.

ESERCIZIO 4.11. Si consideri la funzione

```
f(0,m) = m
f(n,m) = n * f(n-1,m+1)
```

Definire la precedenza indotta e disegnarne il diagramma, mostrando che rende il dominio di f ben fondato. Dimostrare per induzione ben fondata che, per ogni n ed m, si ha $f(n, m) = (n + m) * n!$.

ESERCIZIO 4.12. Sia $\mathbb{D}$ l'insieme dei numeri naturali maggiori di 1. Date le funzioni

```
somma(2,m) = m + 2
somma(n,m) = 1 + somma(n-1,m)
prod(2,m)  = 2 * m
prod(n,m)  = m + prod(n-1,m)
```

dimostrare per induzione che $\forall n, m \in \mathbb{D}.\ somma(n, m) \leq prod(n, m)$.

ESERCIZIO 4.13. Si dica qual'è la relazione di precedenza indotta dalla seguente definizione di funzione

```
f(n,m) = if n == m then true
               else if n > m then not(f(n-1,m))
                      else not(f(n,m-1))
```

Si verifichi che tale relazione rende $\mathbb{N} \times \mathbb{N}$ ben fondato e si dimostri per induzione ben fondata che $\forall n, m.\ f(n, m) = pari(|n - m|)$.

ESERCIZIO 4.14. Data la seguente definizione di funzione $ln\colon \mathbb{N} \setminus \{0\} \to \mathbb{N}$, si dimostri per induzione ben fondata che $\forall n \in \mathbb{N} \setminus \{0\}.\ ln(n) \leq n$.

```
ln 1 = 0
ln n = ln(n `div` 2) + 1
```

ESERCIZIO 4.15. Sia *Prod* il predicato booleano $\{(x, y, z) \mid z = x * y\}$. Assumendo le proprietà

(1) $\quad x > 0 \wedge z \geq y \quad \Rightarrow \quad Prod(x, y, z) \equiv Prod(x - 1, y, z - y)$

(2) $\quad x = 0 \wedge z > 0 \quad \Rightarrow \quad Prod(x, y, z) \equiv false$

(3) $\quad x > 0 \wedge z < y \quad \Rightarrow \quad Prod(x, y, z) \equiv false$

(4) $\quad x = 0 \wedge z = 0 \quad \Rightarrow \quad Prod(x, y, z) \equiv true$

1. Scrivere una funzione ricorsiva f che calcoli il predicato *Prod*.

2. Dare il diagramma della relazione di precedenza indotta dalla definizione e mostrare che esso rende il dominio ben fondato.
3. Dimostrare per induzione la correttezza di f.

ESERCIZIO 4.16. Data la seguente definizione ricorsiva di funzione

```
f 0 = 0
f 1 = 2
f n = if n > 2 && pari n then f(n-3) + 1
        else f(n+1) + 1
```

1. dare il diagramma della relazione di precedenza indotta dalla definizione;
2. dimostrare per induzione che f coincide con la seguente funzione g.

$$g(n) = \begin{cases} 0 & \text{se } n = 0 \\ n-1 & \text{se } n > 0 \text{ e } pari(n) \\ n+1 & \text{altrimenti.} \end{cases}$$

ESERCIZIO 4.17. Data la seguente definizione ricorsiva di funzione

```
f 100 = 1
f  n  = n + 1 * f(n+1)
```

dire su quali sottoinsiemi di $\mathbb{N}$ la funzione è totale e perché. Verificare inoltre per induzione che per ogni n su cui f è definita si ha $f(n) = 100!/n!$

ESERCIZIO 4.18. Facendo uso esclusivamente di confronto per uguaglianza con zero, incrementi e decrementi unitari, scrivere una funzione ricorsiva che ritorni *0*, *1*, o *2* a seconda che i suoi due argomenti siano, rispettivamente, uguali, il primo maggiore del secondo, o il secondo maggiore del primo.

ESERCIZIO 4.19. Date le seguenti proprietà del predicato booleano *Multiplo(x,y)*, (dove x ed y sono naturali non nulli)

$$x = y \Rightarrow Multiplo(x, y) \tag{1}$$

$$x < y \Rightarrow \neg Multiplo(x, y) \tag{2}$$

$$x > y \Rightarrow Multiplo(x, y) \equiv Multiplo(x - y, y) \tag{3}$$

scrivere una funzione ricorsiva $\mathbb{N} \setminus \{0\} \times \mathbb{N} \setminus \{0\} \to \mathbb{B}$ che calcoli *Multiplo* e verificarne la correttezza per induzione.

ESERCIZIO 4.20. Si consideri la seguente definizione ricorsiva.

```
f(0,m) = m
f(n,0) = n
f(n,m) = f(n-1,m-1)
```

Si dimostri per induzione che $\forall n, m \in \mathbb{N}.\ f(n, m) = |n - m|$.

ESERCIZIO 4.21. Data la seguente definizione ricorsiva

```
f(0,0) = 1
f(0,m) = 2 * f(0,m-1)
f(n,m) = 1 + f(n-1,m+1)
```

1. disegnare il diagramma della relazione di precedenza indotta dalla definizione;
2. dimostrare per induzione che $\forall n, m \in \mathbb{N}.\ f(n, m) = n + 2^{n+m}$.

ESERCIZIO 4.22. Data la seguente definizione ricorsiva sul dominio dei numeri interi

```
f  0    = 100
f 100   = 0
f(-100) = 0
f n     = if abs n > 100 then 100 - abs n
              else if n > 0 then 1 + f(-n-1)
                else 1 + f(-n+1)
```

1. determinare la relazione di precedenza indotta dalla definizione e disegnare il diagramma;
2. dimostrare per induzione che $\forall n \in \mathbb{N}.\ f(n) = 100 - |n|$.