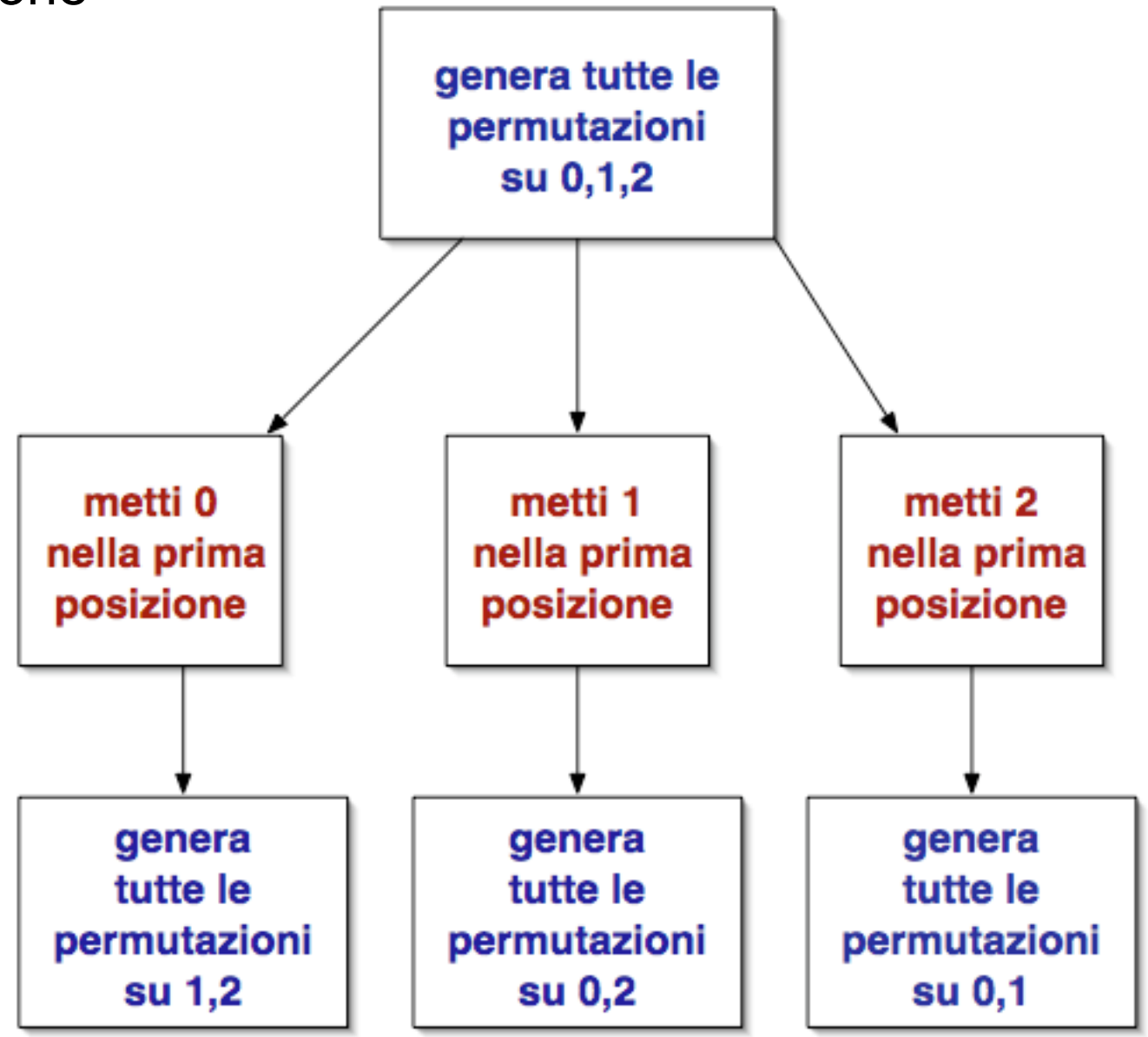
Generare tutte le permutazioni

Le permutazioni su n elementi sono n!
Per n = 3 sono

permutazione numero 1
0 1 2
permutazione numero 2
0 2 1
permutazione numero 3
1 0 2
permutazione numero 4
1 2 0
permutazione numero 5
2 0 1
permutazione numero 6
2 1 0

Ricorsione


genera tutte le permutazioni su 0,1,2
metti 0 nella prima posizione
metti 1 nella prima posizione
metti 2 nella prima posizione
genera tutte le permutazioni su 1,2
genera tutte le permutazioni su 0,2
genera tutte le permutazioni su 0,1

Generare le permutazioni: rappresentazione

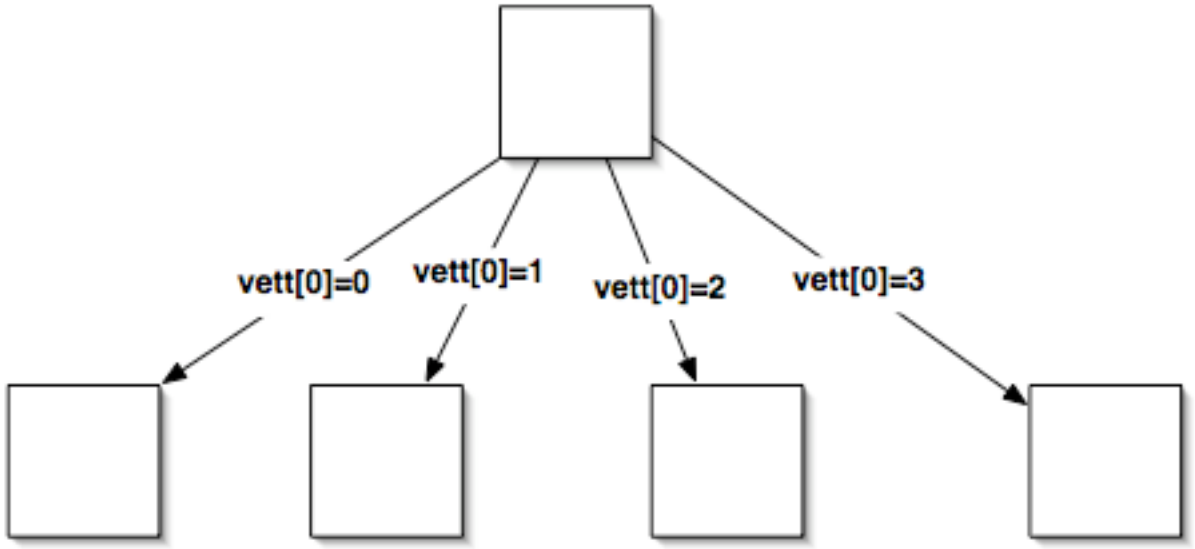
Una permutazione su 0,1,…,n-1 si può rappresentare con un vettore di n elementi:

Per esempio se n = 3

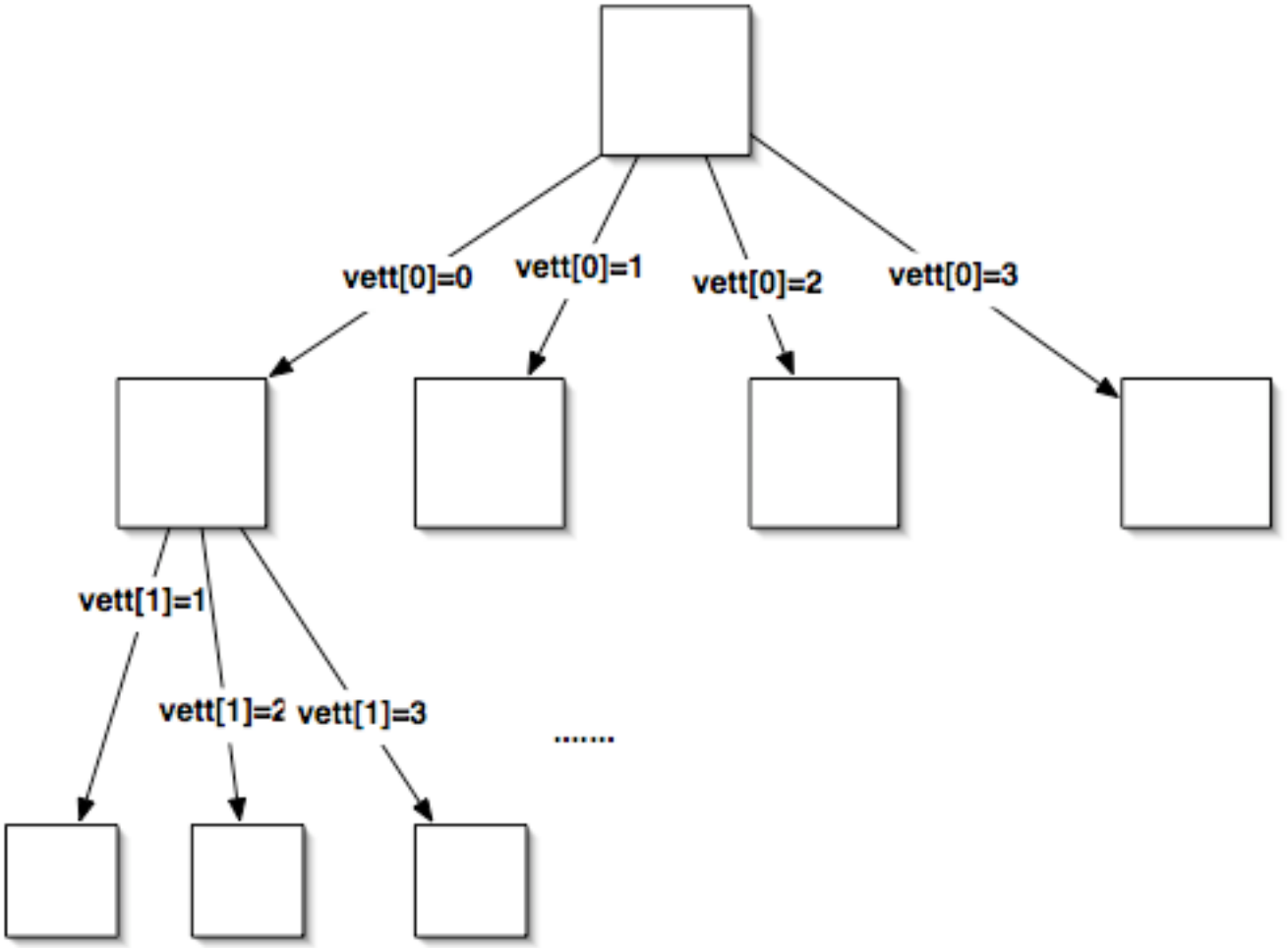
vett[0] = 2, vett[1] = 1 e vett[2]=0

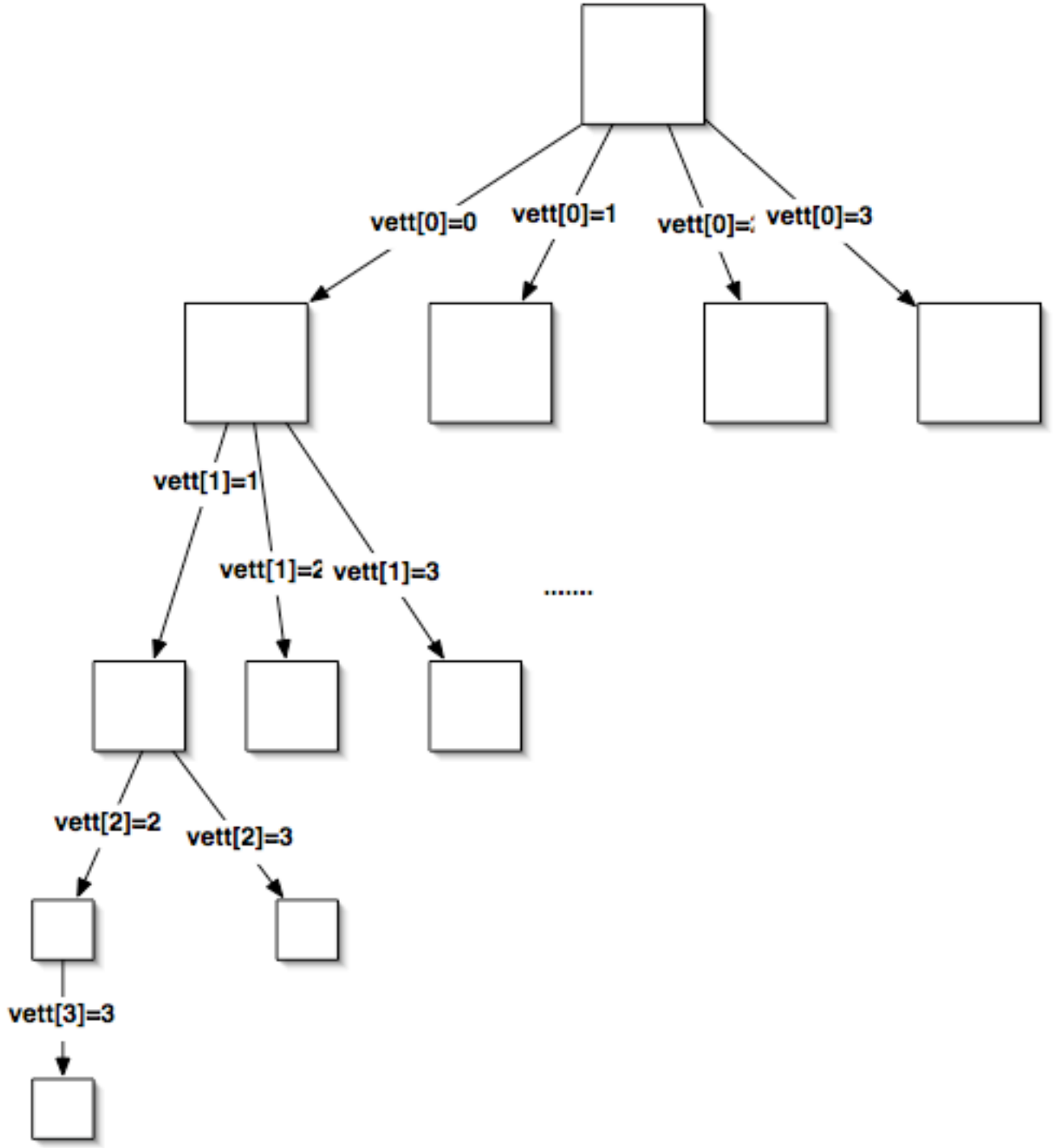
Rappresenta la permutazione 2,1,0.

Generare le permutazioni

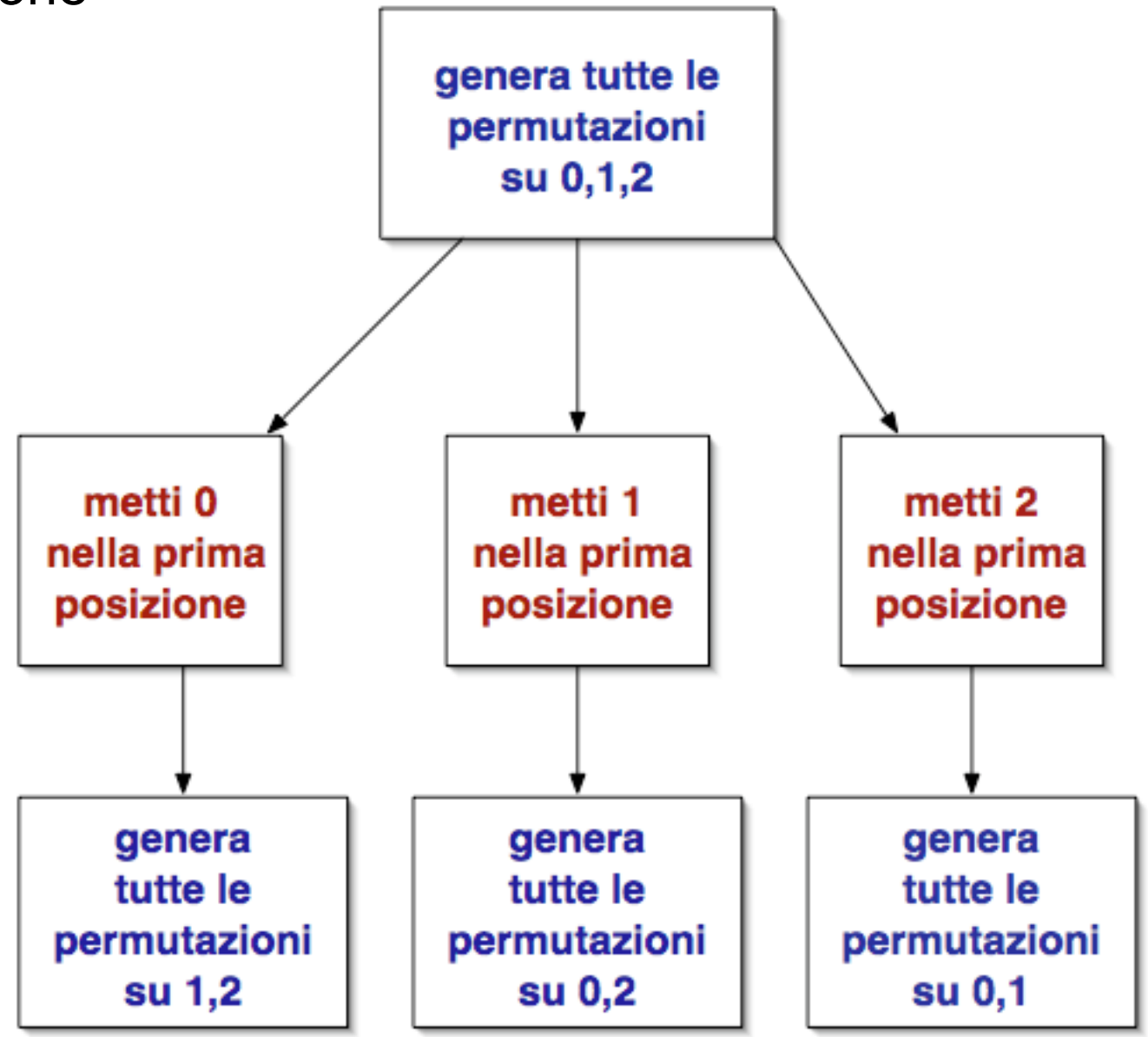


Generare le permutazioni




vett[0]=0
vett[0]=1
vett[0]=
vett[0]=3
vett[1]=1
vett[1]=2
vett[1]=3
.....
vett[2]=2
vett[2]=3
vett[3]=3

Ricorsione


genera tutte le permutazioni su 0,1,2
metti 0 nella prima posizione
metti 1 nella prima posizione
metti 2 nella prima posizione
genera tutte le permutazioni su 1,2
genera tutte le permutazioni su 0,2
genera tutte le permutazioni su 0,1

Algoritmo più dettagliato

Riprendendo il nostro algoritmo: dobbiamo gestire in generale il caso in cui un arbitrario sottoinsieme di m elementi

$$P \subseteq \{0,1,\ldots,n-1\}$$

è stato utilizzato come insieme di valori nella permutazione corrente e dobbiamo ancora sistemare gli elementi in

$$val = \{0,1,\ldots,n-1\} - P.$$

L'insieme P nel nostro schema ricorsivo contiene di volta in volta il primo valore attribuito:

P={0} nel primo ramo, P={1} nel secondo, P={2} nell'ultimo.

Gli elementi di P sono stati piazzati nelle prime m≥1 posizioni della permutazione.

Allora ora resta da generare tutte le permutazioni degli elementi di val, pseudocodice:

per ogni elemento x in val do

metti x nella posizione m+1

e genera ricorsivamente tutte le permutazioni su val - {x}

Le permutazioni: verso il codice C

Abbiamo bisogno di ricordare gli elementi già piazzati (gli elementi di P) e gli elementi ancora da inserire in una permutazione.

Scegliamo di farlo con un vettore val

val[i] = 1 se i, $0 \leq i < n$, è stato utilizzato nella permutazione corrente

val[i] = 0 altrimenti

Il vettore val sarà quindi passato a ogni chiamata ricorsiva nella quale si cercherà il primo valore disponibile ($\neq 0$) per la corrente posizione nella permutazione.

Al rientro da una chiamata nella quale abbiamo usato val[i], che quindi vale 1, dobbiamo rendere val[i] di nuovo disponibile ponendo val[i] = 0.

I parametri

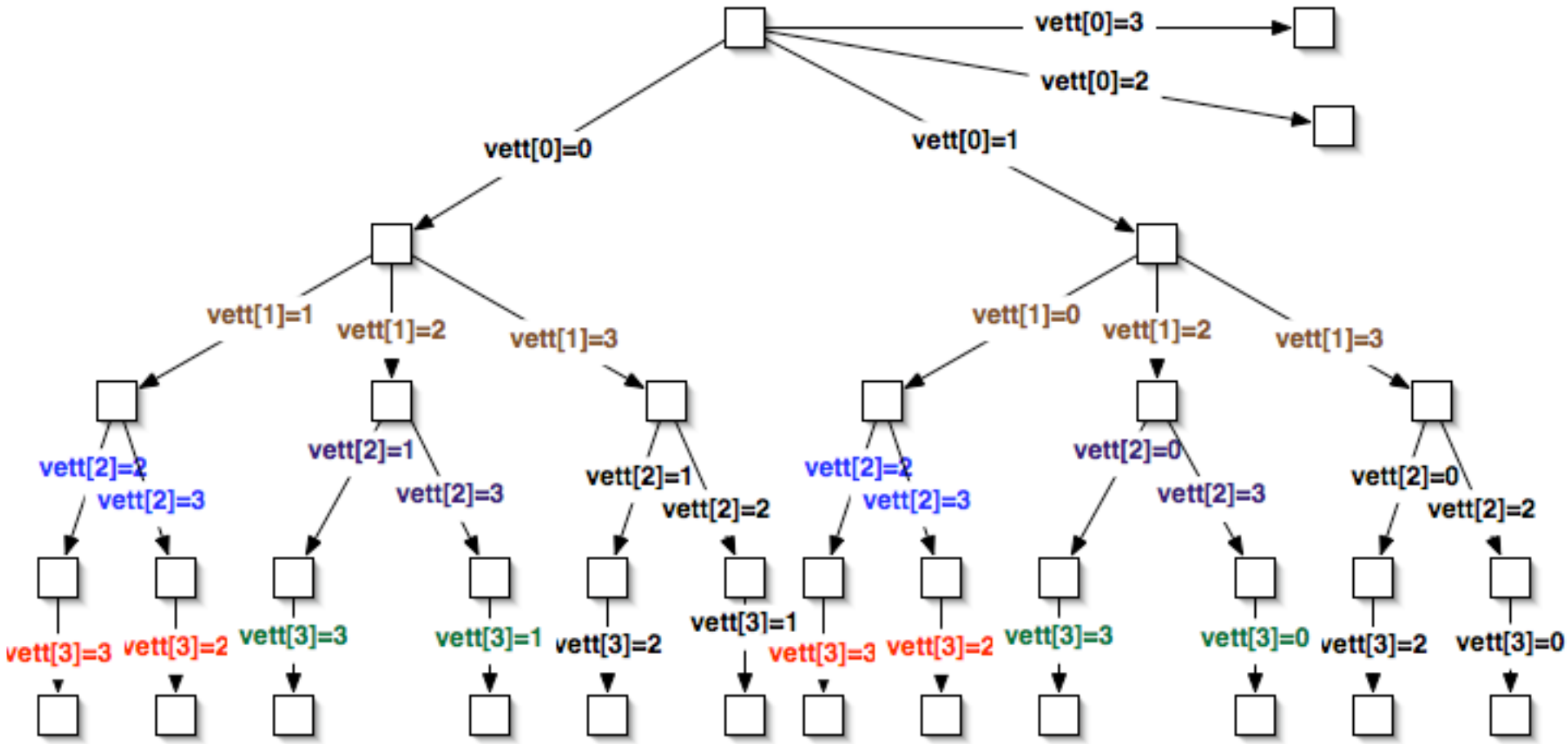
Parametri per la funzione ricorsiva:

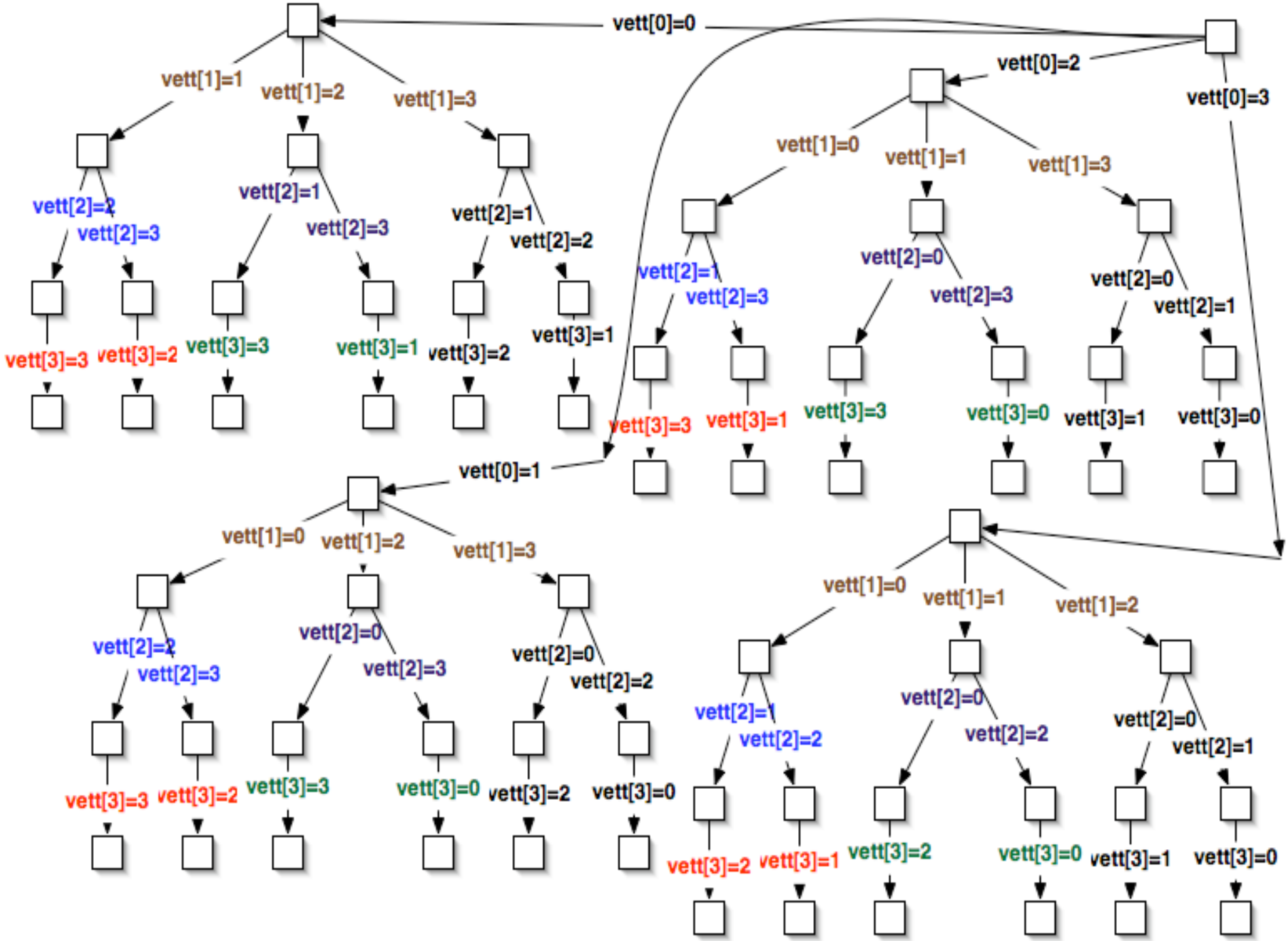
- **un parametro di tipo vettore, vett, di interi per la permutazione,**
- **Un parametro di tipo intero, n, per la lunghezza del vettore vett,**
- **un parametro di tipo vettore,val, di interi per i valori già usati,**
- **un parametro di tipo intero, k, per la posizione corrente nella permutazione**

Le permutazioni:il programma

```
void genTPerm(int *vett,int k,int *val,unsigned int n)
/*genera tutte le permutazioni sui valori 0,1,...,n-1,
 inizialmente  k = -1 e val[i] = 0, per 0≤i<n.
prec: vett != NULL && val != NULL;
*postc: stampa a video tutte le permutazioni su 0,1,...,n-1, memorizzate in vett.*/
{int i;
if (k == n - 1)
     stampaVett(vett,n);
else
     for(i=0;i<n;i++)
          if (val[i] == 0)
                    {vett[k+1] = i;
                    val[i] = 1;
                    genTPerm(vett,k+1,val,n);
                    val[i] = 0;
                    }
}
```

K inizialmente è -1 in modo che quando k= n-1 si sono fatte n chiamate e quindi la permutazione è completa


vett[0]=3
vett[0]=2
vett[0]=0
vett[0]=1
vett[1]=1
vett[1]=2
vett[1]=3
vett[1]=0
vett[1]=2
vett[1]=3
vett[2]=2
vett[2]=3
vett[2]=1
vett[2]=3
vett[2]=1
vett[2]=2
vett[2]=2
vett[2]=3
vett[2]=0
vett[2]=3
vett[2]=0
vett[2]=2
vett[3]=3
vett[3]=2
vett[3]=3
vett[3]=1
vett[3]=2
vett[3]=1
vett[3]=3
vett[3]=2
vett[3]=3
vett[3]=0
vett[3]=2
vett[3]=0


vett[0]=0
vett[1]=1
vett[1]=2
vett[1]=3
vett[2]=2
vett[2]=3
vett[2]=1
vett[2]=3
vett[2]=1
vett[2]=2
vett[3]=3
vett[3]=2
vett[3]=3
vett[3]=1
vett[3]=2
vett[3]=1
vett[0]=2
vett[0]=3
vett[1]=0
vett[1]=1
vett[1]=3
vett[2]=1
vett[2]=3
vett[2]=0
vett[2]=3
vett[2]=0
vett[2]=1
vett[3]=3
vett[3]=1
vett[3]=3
vett[3]=0
vett[3]=1
vett[3]=0
vett[0]=1
vett[1]=0
vett[1]=2
vett[1]=3
vett[2]=2
vett[2]=3
vett[2]=0
vett[2]=3
vett[2]=0
vett[2]=2
vett[3]=3
vett[3]=2
vett[3]=3
vett[3]=0
vett[3]=2
vett[3]=0
vett[1]=0
vett[1]=1
vett[1]=2
vett[2]=1
vett[2]=2
vett[2]=0
vett[2]=2
vett[2]=0
vett[2]=1
vett[3]=2
vett[3]=1
vett[3]=2
vett[3]=0
vett[3]=1
vett[3]=0