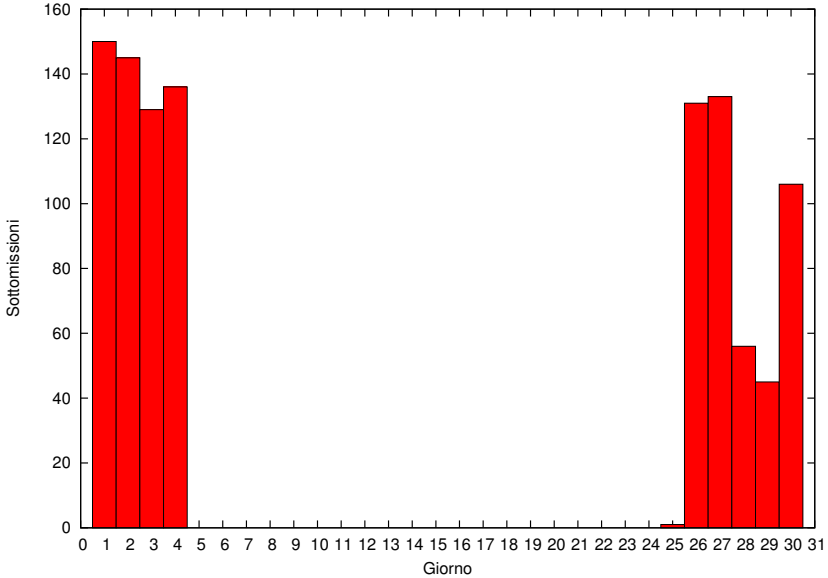
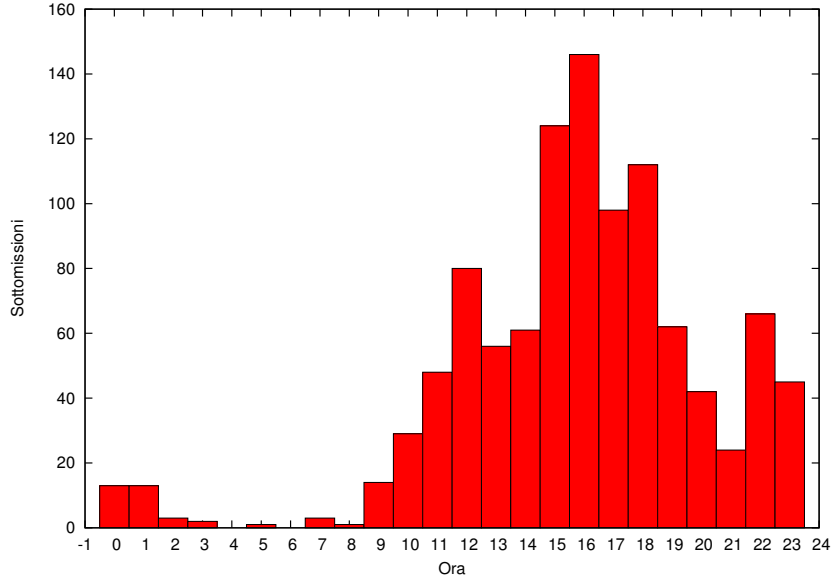
Spiegazioni ASD 2015
Kraken

Risultati

Statistiche

Numero sottoposizioni: 999





Punteggi

- $P < 30 \rightarrow$ progetto non passato
- $30 \leq P \leq 55 \rightarrow$ un punto bonus al voto dello scritto
- $55 \leq P \leq 100 \rightarrow$ due punti bonus al voto dello scritto
- $100 \leq P \rightarrow$ tre punti bonus al voto dello scritto
- Classifica completa sul mio sito

Funzione ricorsiva

- Sia V_i l'insieme di famiglie che possono essere messe subito dopo membri della famiglia i.
- Sia $S(k)$ il numero di sequenze valide di lunghezza k
- Sia $S(k, i)$ il numero di sequenze valide di lunghezza k che iniziano con famiglia i.

$$S(k,i) = \begin{cases} 1 & \text{if } k = 0 \\ \sum_{j \in V_i} S(k-1, j) \end{cases}$$

$$S(k) = \sum S(k,i)$$

Funzione ricorsiva

```
long long int S(int k, int i){
  if(k==1)
    return 1;
  long long int count = 0;
  for(int j=0;j<N;j++){
    if(can[i][j])
      count+=S(k-1,j);
  }
  return count;
}
```

Memoization

```
long long int S(int k, int i){
  if(k==1)
    return 1;
  if(memo[k][i]==-1){
    long long int count = 0;
    for(int j=0;j<N;j++){
      if(can[i][j])
        count+=S(k-1,j);
    }
    memo[k][i]=count;
  }
  return memo[k][i];
}
```

Iterativa

```
long long int found = N;
int size = 1;
memo.push_back(vector<long long int>(N,0));
memo.push_back(vector<long long int>(N,1));
while(found<X){
   size++;
   memo.push_back(vector<long long int>(N));
   for(int i=0;i<N;i++){
      long long int count = 0;
      for(int j=0;j<N;j++){
         if(can[i][j])
            count+=S(size-1,j);
      }
      memo[size][i]=count;
      found+=count;
   }
}
```

Iterativa

```
long long int found = N;
int size = 1;
vector<long long int> memo(N,1);
while(found<X){
   size++;
   vector<long long int> newmemo(N,0);
   for(int i=0;i<N;i++){
      for(int j=0;j<N;j++){
         if(can[i][j])
            newmemo[i]+=memo[j];
      }
      found+=newmemo[i];
   }
   memo=newmemo;
}
```

Complessità

- Sia L la lunghezza dell'output del programma
- Algoritmo è $O(N * L)$
- Visto che S cresce esponenzialmente, si tratta di $O(N * \log K)$
- Esiste un caso in cui S non cresce esponenzialmente. Come lo gestiamo?