Backtracking
Sum of Subsets

Sum of subsets

- Given n distinct positive numbers wi, and m, find all subsets whose sums are m.

- Explicit Constraints :
- Xi = { j / j is an interger and 1 <= j <= n }

- Implicit Constraints :

i. no two xi's are same

ii. Sum of the corresponding wi's be m.

iii. To avoid generation of multiple instances of the same subsets)

- We can formulate this problem using either
  - *Fixed*- or *variable* - sized tuples.

```
Algorithm SumOfSub( s, k, r )
// Find all  subsets of w[ 1:n ]  that sum to m
// It is assumed that w[1] ≤ m  and     ∑       wi ≥
   m
{
      x[ k ]=1;  //  left child
      if( s + w[k] = m )  then write( x[ 1: k ] ) ;
   // Subset found
                 else  if ( s + s [ k ] + s[ k+1 ] ≤ m
   )
               then  SumOfSub( s+ w[k], k+1, r- w[k]
   )
      // Generate right child and evaluate Bk
     if ( ( s + r – w[ k ] ≥ m ) and ( s + w[ k+1 ]
   ≤ m ) ) then
      {
            x[ k ] = 0;
      }           SumOfSub( s, k+1, r- w[k] ) ;
}
```

Ex.- n=6, m=30, w [1:6]= { 5,10,12,13,15,18 }

Portion of state space tree generated by SumOfSub.

circular nodes indicate subsets with sums equal to m.