Sum Of Subsets (Backtracking)

Problem :- There are a 'n' distinct positive numbers (weights) and problem is to find all combination of these numbers whose sums are some value say 'm'. This is called as Sum of Subsets problem.

Solution :- A vector $\{x_1, x_2 \dots x_n\}$ will represent solution using fixed tuple size strategy. In this, x_i of solution vector is either one or zero depending on whether w_i is included or not.

Bounding function :-

* $B_k(x_1, \dots x_k)$ = true iff $\{2, 3, 4, 5\}$

Remaining

$S1 = \{1, 1\}$ $m = 13$

total sum till now $\rightarrow$ $\sum_{i=1}^{k} w_i x_i + \sum_{i=k+1}^{n} w_i \geq m$ $S2 = \{1, 0\}$ $m = 13$

$5 + 11 \geq 10$

* We can make bounding function more strong, if we assume the w_i's are initially in nondecreasing order.

* In this case $x_1 \dots x_k$, cannot lead to an answer node if $\{2, 3, 5, 6\}$ $m = 6$

$$\sum_{i=1}^{k} x_i w_i + w_{k+1} > m$$

$\{2, 3\}$

$\therefore$ The bounding fn are

$B_k(x_1 \dots x_k)$ = true iff

$$\boxed{\sum_{i=1}^{k} w_i x_i + \sum_{i=k+1}^{n} w_i \geq m \quad \text{and} \quad \sum_{i=1}^{k} w_i x_i + w_{k+1} \leq m}$$

Algorithm :-

purpose :- Find all subsets of $w[1 \ldots n]$ that sums to m.

~~SumofSub(s, k, r)~~

Input :- $s = \sum_{j=1}^{k-1} w[j] * x[j]$ and $r = \sum_{j=k}^{n} w[j]$

$s = 0$

r = sum of all elements

Output :- All subsets that sums to m.

Assumption :- (i) $w[i] \leq m$ and $\sum_{i=1}^{n} w[i] \geq m$

(ii) Value of $x[j] = 0$ if $w[j]$ is excluded

else value of $x[j] = 1$, $1 \leq j < n$

Global Array :- $x[j]$ and $w[j]$ are global arrays. where $w[j]$'s are in non decreasing order.

SumofSubset (s, k, r)

total sum, remaining

{

// Generate left child.

// B_{k-1} is true, $s + w[k] \leq m$

including

1. $x[k] = 1$

subset found 2. if $(s + w[k] = m)$ then write $(x[1 \ldots k])$

3. else if $(s + w[k] + w[k+1] \leq m)$

4. then SumofSubset$(s + w[k], k+1, r - w[k])$.

5. // Generate Right child

if $((s + r - w[k] \geq m)$ and $(s + w[k+1] \leq m))$ then

6. $x[k] = 0$

7. Sum of subset $(s, k+1, r - w[k])$;

Scanned by CamScanner

Explanation :

1. By using s and r variables, algorithms avoids computing $\sum_{i=1}^{k-1} w_i x_i$ and $\sum_{i=k}^{n} w_i$ each time.

2. Initial call is SumofSubset $(0, 1, \sum_{i=1}^{n} w_i)$

3. Algorithm handles 3 cases

(a) Solution is found i.e in line 2 add k^{th} element, write solution, and stop search.

(b) line 3 & 4, adds k^{th} element in the subset and search continues further moving to $k+1$ element.

(c) line 5 to 7, is executed when $k+1$ element is not to be included.

Analysis :-

Running time complexity of the recurrence is

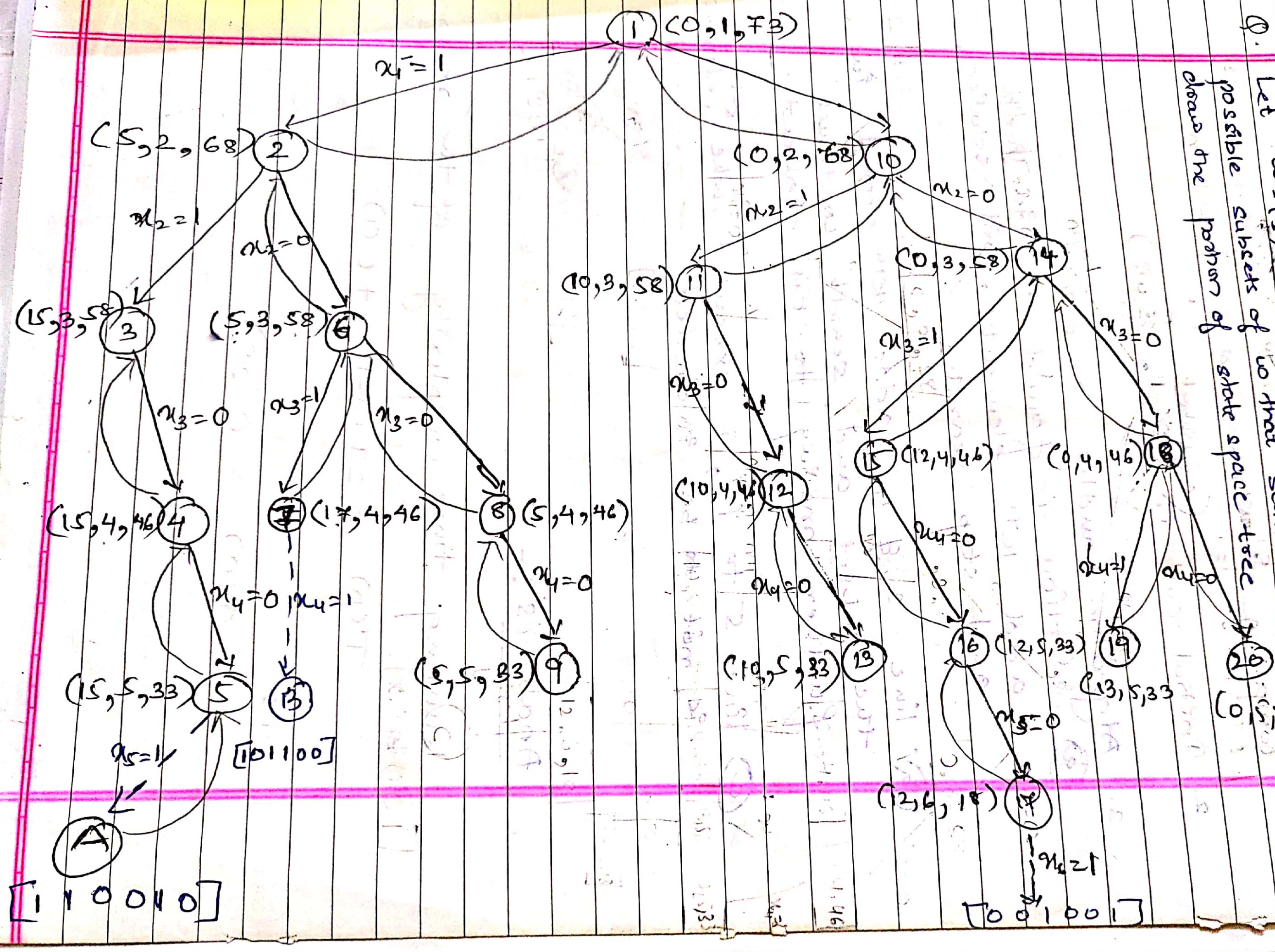
$$T(n) \leq 2T(n-1) + O(1)$$

is

$$T(n) = O(2^n)$$

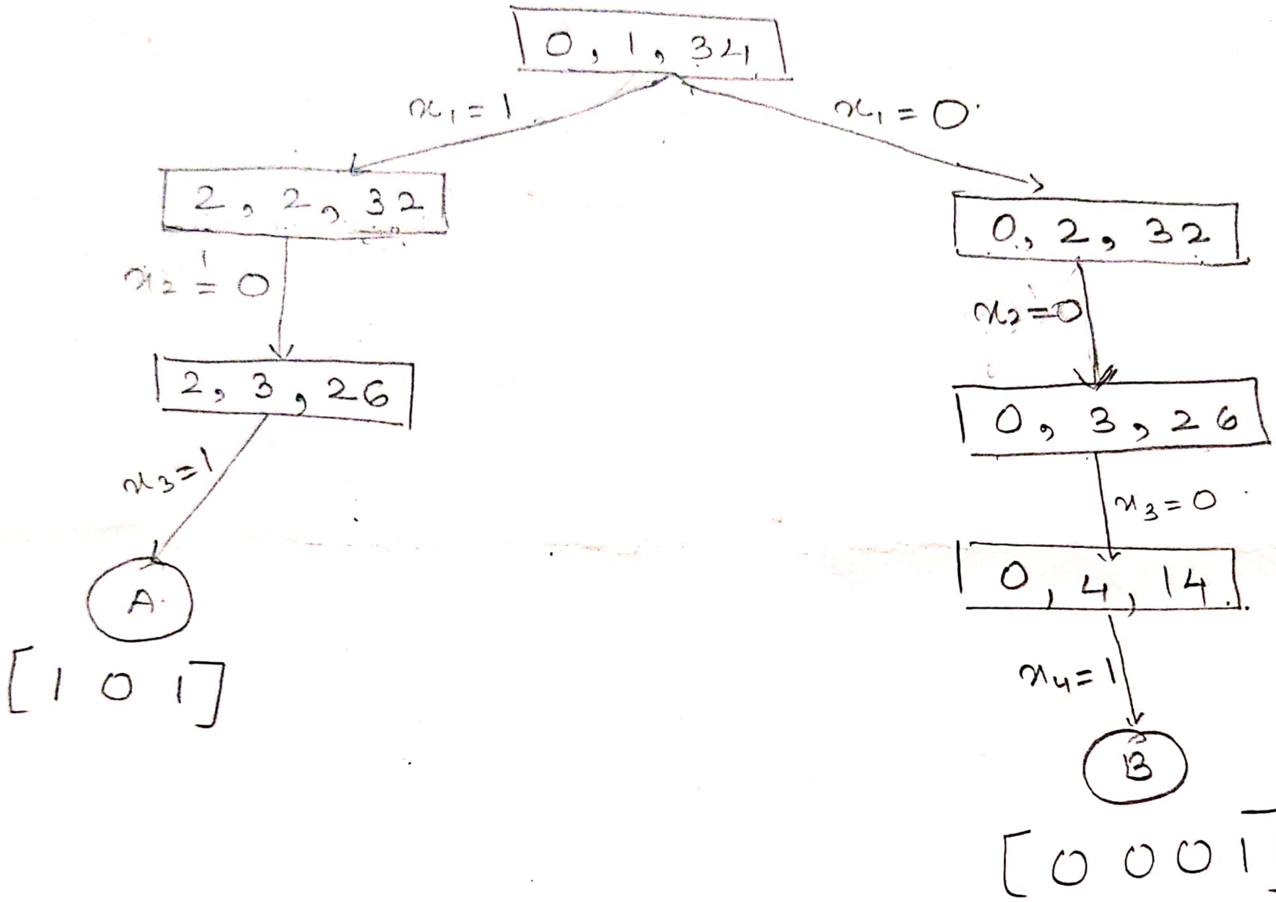
Scanned by CamScanner

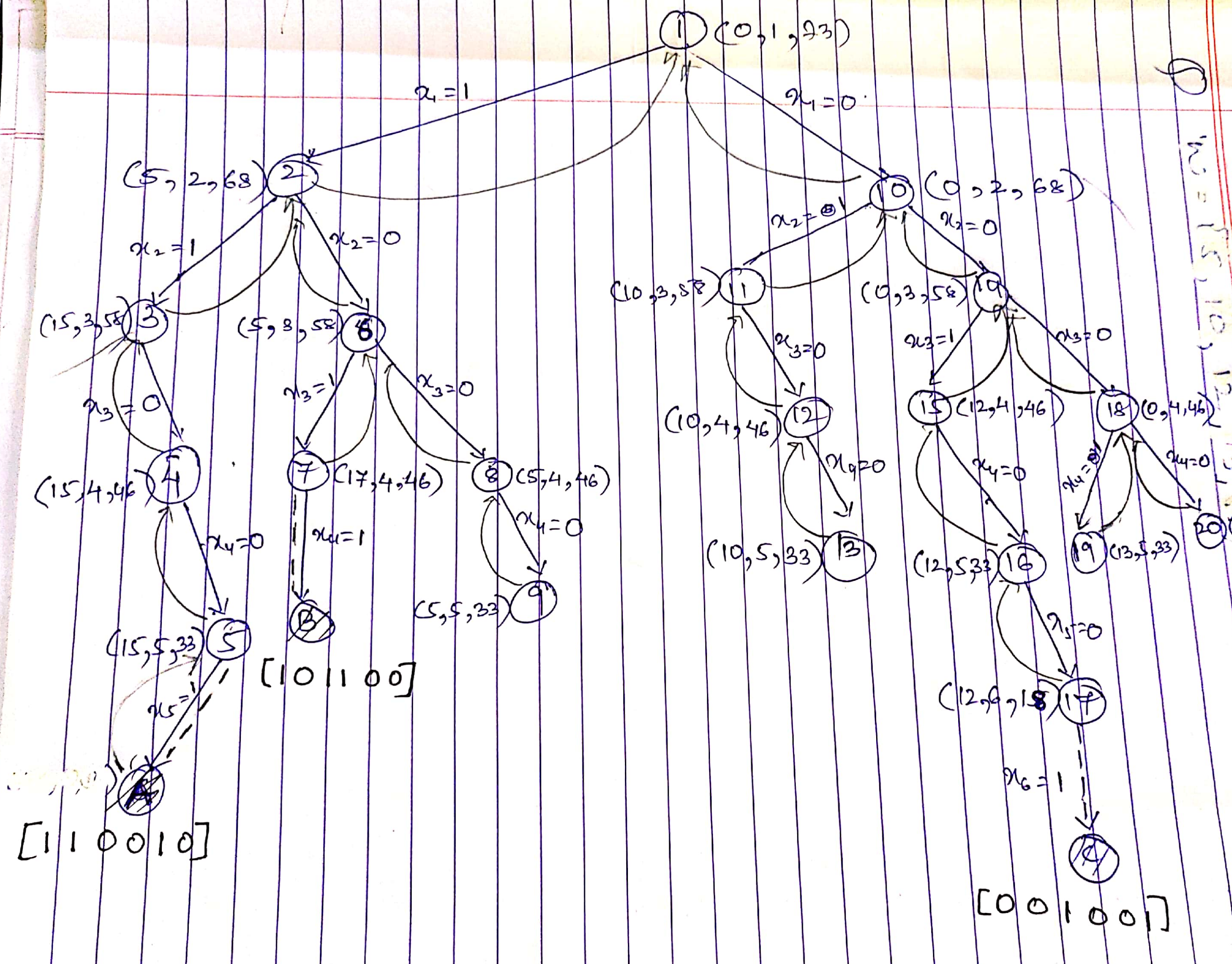
Q. Let $w = \{5, 10, 12, 13, 15, 18\}$ and $m = 30$, Find all possible subsets of w that sums to m and draw the portion of state space tree



Q Let $w = \{2, 6, 13, 14\}$ and $m = 14$, find all possible subsets of w that sums to m and draw state space tree.

$\boxed{S, K, r}$





$w = \{5, 10, 12, 13, 15, 18\}$

$m = 30$