辅学 lesson2

Zhejiang University, Advanced Data Structrue and Algorithm Analysis

赵一帆

2025 年 11 月 16 日

目录

Backtracking

- n-Queen problem
- Turnpike problem
- Tic-Tac-Toe
- $\alpha - \beta$ pruning

backtracking - Turnpike problem

单选题

The turnpike reconstruction problem is to reconstruct a point set from distances between every pair of points. Given a set of distances 2, 2, 3, 3, 4, 5, 6, 7, 8, 10, there are 5 corresponding points. Assume that X1 is at 0 and X5 is at 10. Which of the following statements is TRUE?

A. X2=3, X3=6, X4=8

B.X2=3, X3=4, X4=8

C.X2=2, X3=4, X4=8

D.X2=2, X3=6, X4=7

backtracking - Turnpike problem

单选题

The turnpike reconstruction problem is to reconstruct a point set from distances between every pair of points. Given a set of distances 2, 2, 3, 3, 4, 5, 6, 7, 8, 10, there are 5 corresponding points. Assume that X1 is at 0 and X5 is at 10. Which of the following statements is TRUE?

A. X2=3, X3=6, X4=8

B.X2=3, X3=4, X4=8

C.X2=2, X3=4, X4=8

D.X2=2, X3=6, X4=7

Backtracking tree

backtracking - Turnpike problem

多选题

In a turnpike reconstruction problem, the distance set is given as 1, 1, 2, 4, 4, 5, 5, 5, 6, 6, 7, 9, 10, 11, 12. In now backtracking state(a node in the backtracking tree), we temporarily identify four points: $x_1 = 0, x_2 = 12, x_3 = 1, x_4 = 2 (x_4 = 10)$, which next try is possible ?
$A. x_5 = 3$ $B. x_5 = 4$ $C. x_5 = 5$ $D. x_5 = 6$ $E. x_5 = 7$ $F. x_5 = 8$ $G. x_5 = 9$

backtracking - Turnpike problem

多选题

In a turnpike reconstruction problem, the distance set is given as 1, 1, 2, 4, 4, 5, 5, 5, 6, 6, 7, 9, 10, 11, 12. In now backtracking state(a node in the backtracking tree), we temporarily identify four points: $x_1 = 0, x_2 = 12, x_3 = 1, x_4 = 2 (x_4 = 10)$, which next try is possible ?
$A.x_5 = 3$ $B.x_5 = 4$ $C.x_5 = 5$ $D.x_5 = 6$ $E.x_5 = 7$ $F.x_5 = 8$ $G.x_5 = 9$

Backtracking tree is a binary search tree, so in every node we will try two different children unless the node is a leaf of the backtracking tree or it is pruned.

backtracking - Turnpike problem

多选题

In a turnpike reconstruction problem, the distance set is given as 1, 1, 2, 4, 4, 5, 5, 5, 6, 6, 7, 9, 10, 11, 12. In now backtracking state(a node in the backtracking tree), we temporarily identify four points: $x_1 = 0, x_2 = 12, x_3 = 1, x_4 = 2 (x_4 = 10)$, which next try is possible ?
$A. x_5 = 3$ $B. x_5 = 4$ $C. x_5 = 5$ $D. x_5 = 6$ $E. x_5 = 7$ $F. x_5 = 8$ $G. x_5 = 9$

Backtracking tree is a binary search tree, so in every node we will try two different children unless the node is a leaf of the backtracking tree or it is pruned.
$x_1 = 0, x_2 = 12, x_3 = 1, x_4 = 2$, we have use the distance 1, 1, 2, 10, 11, 12, and the left biggest distance is 9.

backtracking - Turnpike problem

多选题

In a turnpike reconstruction problem, the distance set is given as 1, 1, 2, 4, 4, 5, 5, 5, 6, 6, 7, 9, 10, 11, 12. In now backtracking state(a node in the backtracking tree), we temporarily identify four points: $x_1 = 0, x_2 = 12, x_3 = 1, x_4 = 2(x_4 = 10)$, which next try is possible ?
$A.x_5 = 3$ $B.x_5 = 4$ $C.x_5 = 5$ $D.x_5 = 6$ $E.x_5 = 7$ $F.x_5 = 8$ $G.x_5 = 9$

Backtracking tree is a binary search tree, so in every node we will try two different children unless the node is a leaf of the backtracking tree or it is pruned.
$x_1 = 0, x_2 = 12, x_3 = 1, x_4 = 2$, we have use the distance 1, 1, 2, 10, 11, 12, and the left biggest distance is 9.
$x_1 = 0, x_2 = 12, x_3 = 1, x_4 = 10$, we have use the distance 1, 2, 9, 10, 11, 12, and the left biggest distance is 7.

Backtracking - Tic-Tac-Toe

单选题

In the Tic-tac-toe game, a "goodness" function of a position is defined as $f(P) = W_X - W_O$, where W is the number of potential wins at position P. Player X tries to maximize the "goodness" function while player O tries to minimize it. Consider the following position. Now player O can choose one blank from a, b, c, d, e and f to play. which of the following statements is correct?

A.In this position, $W_X = 3$, $W_O = 2, f(P) = 1$.

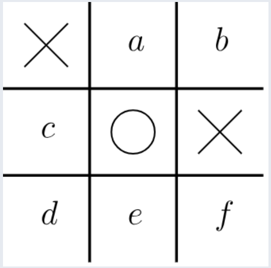
B,After player O plays a piece on b, d or e, the "goodness" function decreases by 2.

C.Player O has a strategy to win the game.

D.Suppose that player O plays a piece on a, c or f. For at least one of these three cases, player X has a strategy to win the game.

Backtracking - Tic-Tac-Toe

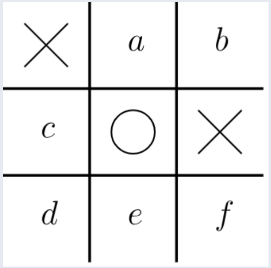
单选题



A.In this position, $W_X = 3, W_O = 2, f(P) = 1$.
B,After player O plays a piece on b, d or e, the "goodness" function decreases by 2.
C.Player O has a strategy to win the game.
D.Suppose that player O plays a piece on a, c or f. For at least one of these three cases, player X has a strategy to win the game.

Backtracking - Tic-Tac-Toe

单选题



A.In this position, $W_X = 3, W_O = 2, f(P) = 1$.
B,After player O plays a piece on b, d or e, the "goodness" function decreases by 2.
C.Player O has a strategy to win the game.
D.Suppose that player O plays a piece on a, c or f. For at least one of these three cases, player X has a strategy to win the game.

backtracking - $\alpha - \beta$ pruning

单选题

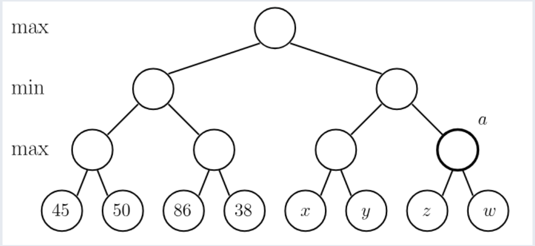
Consider the α-β pruning algorithm on the following game tree. Which of the following statements is/are correct?

A.We DO NOT prune any node in the left subtree of the root.

B.If $x = 65, z = 55, w = 33$, then node a should NOT be pruned whatever the value of y.

C.If $x = 35, z = 77, w = 33$, then node a should be pruned when and only when $35 \leq y \leq 50$.

D.a should be pruned when either $x \leq 50$ or $y \leq 50$.



solving recurrence function

- Substitution method
- Recursion tree method
- Master Theorem

solving recurrence function

Master Theorem

The solution of equation

$$T(N) = aT(\frac{N}{b}) + \Theta(N^k \log^p N)$$

where $a \geq 1, b > 1$ and $p \geq 0$ is

$$T(N) = \begin{cases} O(N^{\log_b a}) & \textit{if } a > b^k \\ O(N^k \log^{p+1} N) & \textit{if } a = b^k \\ O(N^k \log^p N) & \textit{if } a < b^k \end{cases}$$

solving recurrence function

$$T(N) = 2T(\frac{N}{2}) + N \log N$$
$$T(N) = N \log^2 N$$

solving recurrence function

$$T(N) = 2T(\frac{N}{2}) + N\log N$$
$$T(N) = N\log^2 N$$

$$T(n) = T(\sqrt{n}) + T(\sqrt[3]{n}) + T(\sqrt[6]{n}) + \log n$$
$$T(N) = \log N \log\log N$$

Divide and Conquer

多选题

The n-th Fibonacci number can be computed by divide and conquer method of computing x^n, where x is the matrix $\begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}$. Which of the following statements is/are correct? **A. The nth Fibonacci number F_n can be computed in $O(\log n)$ time.**
B. The n^2th Fibonacci number F_n can be computed in $O(\log n)$ time.
C. The n^3th Fibonacci number F_n can be computed in $O(\log n)$ time.
D. None of the other options is correct.

Divide and Conquer

最近点对问题

- 二维最近点对问题
- 三维最近点对问题

Dynamic Programming

- 运行时间 = 表项数量 × 计算每个表项的时间
- 最经典的动态规划问题：背包问题（Richard Bellman 1957）

背包问题

给定 n 个物品的价值和尺寸，以及背包的容量，选择出其中尺寸之和不超过背包容量的一部分物品，使得这部分物品的价值之和最大.

table[a][b] = max{table[a - 1][b], value[a] + table[a - 1][b - size[a]]}

T = O(nW)

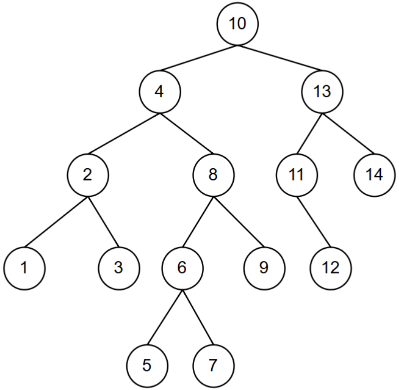
Dynamic Programming

- Fibonacci Numbers
- Ordering Matrix Multiplications
- Optimal Binary Search Tree
- All-Pairs Shortest Path
- Product Assembly

Review - Data Structure

- n - ary search tree: insert, delete
- avl tree
- splay tree
- b - plus tree
- red - black tree
- heap(priority queue): insert, merge, decrease - key, delete - min
- leftist heap
- skew heap
- binomial queue

Review - Avl tree

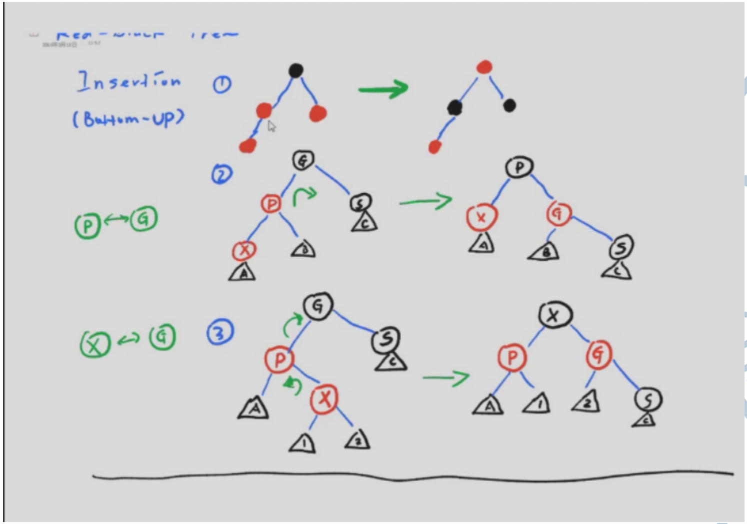


- delete 14.

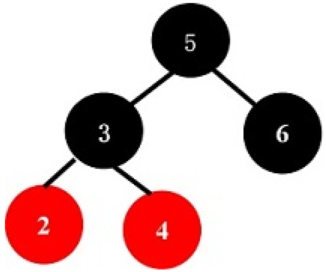
Review - Amortized cost

- Splay tree $\Phi(T) = \sum_{i \in T} R_i = \sum_{i \in T} log(S_i)$
- Deletion of Splay tree need two splay operations.
- Skew heap $\Phi(H) = \#\ of\ heavy\ nodes$
- Binomial queue $\Phi(H) = \#\ of\ heaps$
- $\widehat{c_i} = c_i + \Phi_i - \Phi_{i-1}$
- $\sum_i c_i = \sum_i \widehat{c_i} + \Phi_0 - \Phi_n$

Review - Red black tree



Review - Red black tree

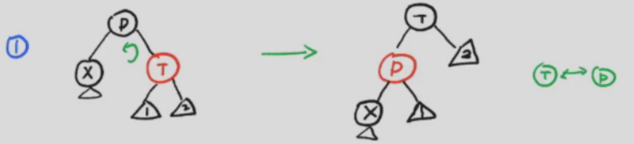


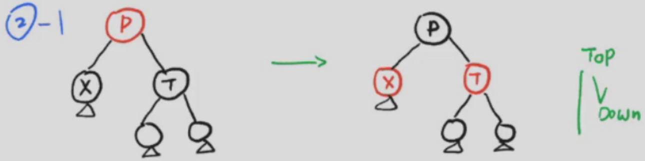
- insert 1.

Review - Red black tree

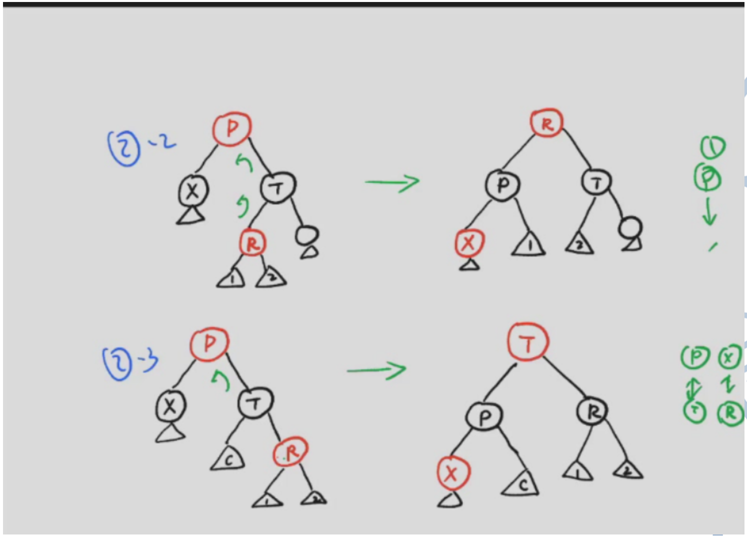
Deletion (Top-Down)

- It is possible to reduce all cases to deleting a leaf node.
- The leaf node to be deleted is black, otherwise it is trivial.
- With smart tree rotations and color changes, make the target node become red.

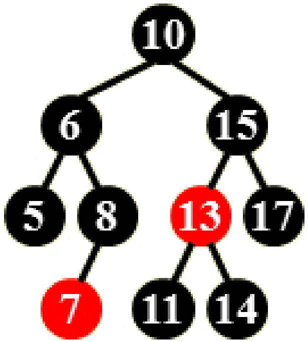




Review - Red black tree



Review - Red black tree



- delete 10.

Review - Heap

- npl vs. heavy(light)
- why can't we use $\Phi(H) = \# \; of \; npl - heavy \; nodes$ as the potential function of skew heap?
- speed up merge from $O(N)$ to $O(\log n)$: Leftist heap and skew heap(amortized).
- speed up insert from $O(\log n)$ to $O(1)$: Binomial queue.
- dijkstra shortest path algorithm(1959)

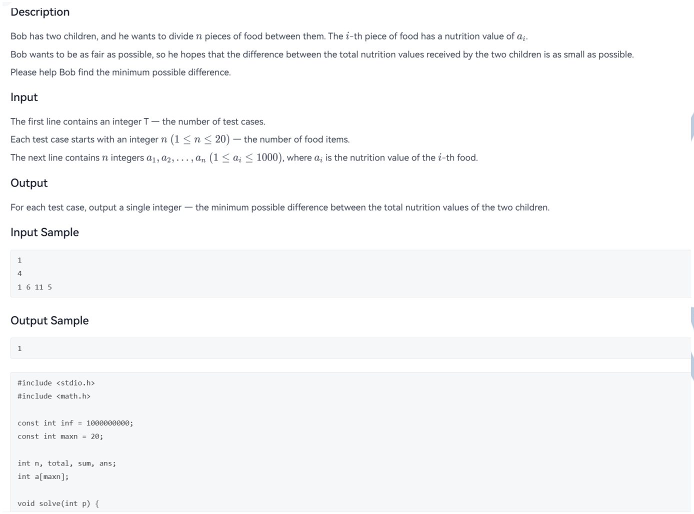
Review - Inverted File Index

- word -> document
- Word stemming and stop words.
- Use search trees and hashing to search items.
- Term-partitioned index vs. Document-partitioned index
- Dynamic indexing.
- Precision vs. recall

Review - Algorithm

- backtracking: n-queen problem, turnpike, tic-tac-toe
- divide and conquer: Closest Points Problem
- dynamic programming: programming, ...
- greedy: The Activity Selection Problem, Huffman code

Review - completion



Description

Bob has two children, and he wants to divide n pieces of food between them. The i-th piece of food has a nutrition value of a_i.

Bob wants to be as fair as possible, so he hopes that the difference between the total nutrition values received by the two children is as small as possible.

Please help Bob find the minimum possible difference.

Input

The first line contains an integer T — the number of test cases.

Each test case starts with an integer n $(1 \leq n \leq 20)$ — the number of food items.

The next line contains n integers $a_1, a_2, \ldots, a_n$ $(1 \leq a_i \leq 1000)$, where a_i is the nutrition value of the i-th food.

Output

For each test case, output a single integer — the minimum possible difference between the total nutrition values of the two children.

Input Sample

```
1
4
1 6 11 5
```

Output Sample

```
1
```

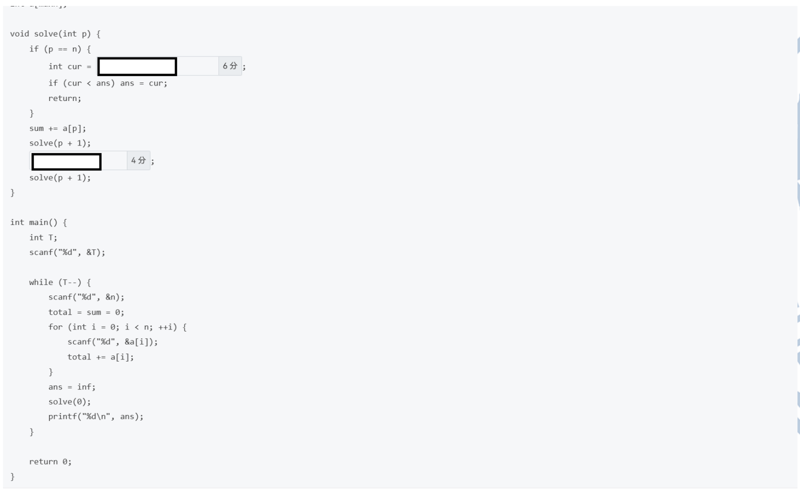
```
#include <stdio.h>
#include <math.h>

const int inf = 1000000000;
const int maxn = 20;

int n, total, sum, ans;
int a[maxn];

void solve(int p) {
```

Review - completion



```
void solve(int p) {
    if (p == n) {
        int cur = ____________ 6分;
        if (cur < ans) ans = cur;
        return;
    }
    sum += a[p];
    solve(p + 1);
    ____________ 4分;
    solve(p + 1);
}

int main() {
    int T;
    scanf("%d", &T);

    while (T--) {
        scanf("%d", &n);
        total = sum = 0;
        for (int i = 0; i < n; ++i) {
            scanf("%d", &a[i]);
            total += a[i];
        }
        ans = inf;
        solve(0);
        printf("%d\n", ans);
    }

    return 0;
}
```