

Backtracking

- N Queens
- Subset Sum

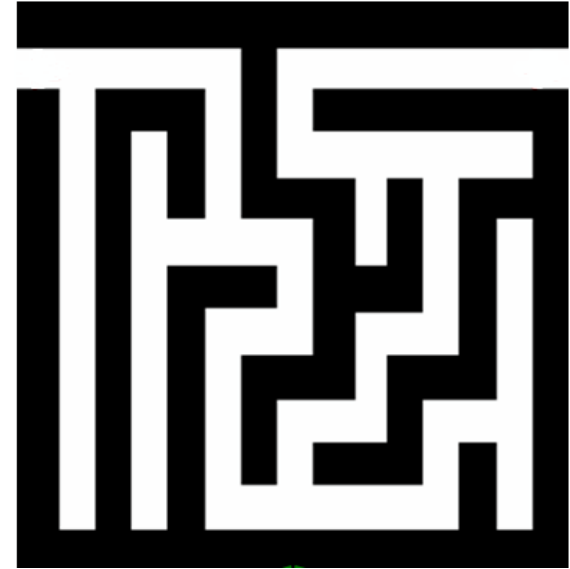
Backtracking

- Suppose you have to make a *series of decisions*, among various *choices*, where
 - You don't have enough information to know what to choose
 - Each decision leads to a new set of choices
 - Some sequence of choices (possibly more than one) may be a solution to your problem
- **Backtracking** is a methodical way of trying out various sequences of decisions, until you find one that "works"

Maze Problem

Problem. Given a maze, find a path from start to finish. At each intersection, you have to decide between three or fewer choices:

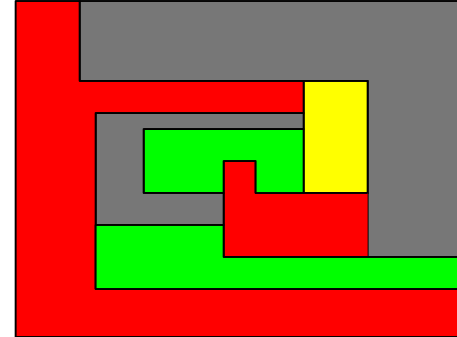
- Go straight
- Go left
- Go right



- You don't have enough information to choose correctly
- Each choice leads to another set of choices
- One or more sequences of choices may (or may not) lead to a solution

Coloring a Map

Problem. Color a map with not more than 4 colors so adjacent countries have different colors.



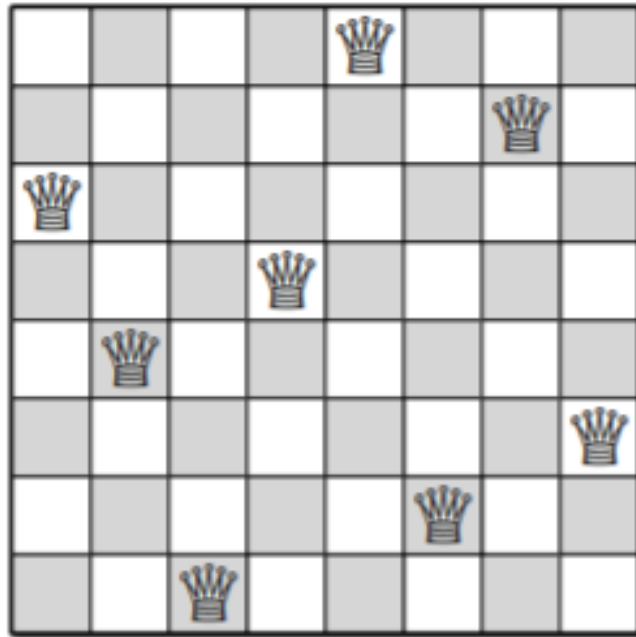
- You don't have enough information to choose colors
- Each choice leads to another set of choices
- One or more sequences of choices may (or may not) lead to a solution

N Queen

N Queens

N Queens.

- The problem is to place n queens on an $n \times n$ chessboard, so that no two queens are attacking each other, i.e., no two queens are in the same row, the same column, or the same diagonal.



Backtracking Solution

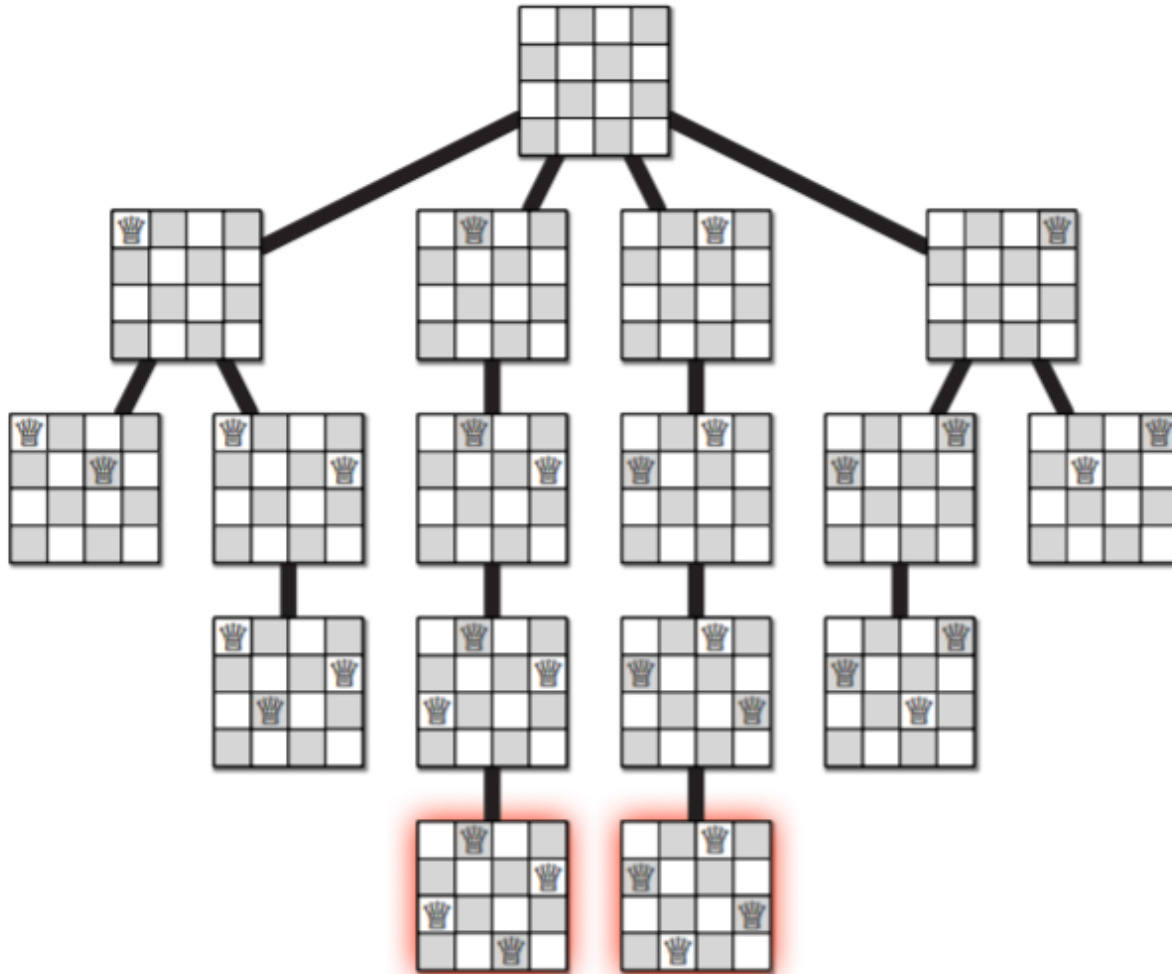
$Q[i]$ indicates which square in row i contains a queen

```
PlaceQueens (Q[1..n], r)

if r = n+1 then
    Print Q[1..n]
else
    for j = 1 to n do
        legal = true
        for i = 1 to r-1 do
            if (Q[i] = j) or (Q[i]=j+r-i) or (Q[i]=j-i+r) then
                legal = false
            if legal then
                Q[r] = j
                PlaceQueens (Q[1..n], r+1)
```

See [here](#) for the demo

Recursion Tree



Subset Sum

Subset Sum

Subset Sum.

- Given a set X of positive integers and target integer T , is there a subset of elements in X that add up to T ?

Example.

- If $X = \{8, 6, 7, 5, 3, 10, 9\}$ and $T = 15$, the answer is true, because the subsets $\{8, 7\}$ and $\{7, 5, 3\}$ and $\{6, 9\}$ and $\{5, 10\}$ all sum to 15.
- if $X = \{11, 6, 5, 1, 7, 13, 12\}$ and $T = 15$, the answer is false.

Backtracking Solution

```
SubsetSum(X,T)
  if T = 0 then
    return True
  else if (T < 0) or (X is empty) then
    return false
  else
    x = any element of X
    with = SubsetSum(X\{x}, T-x)
    wout = SubsetSum(X \{x}, T)
  return (with or wout)
```

Avoiding passing the whole set (X: a global array)

```
SubsetSum(i,T)
  if T = 0 then
    return True
  else if (T < 0) or (i = 0) then
    return false
  else
    with = SubsetSum(i-1, T-X[i])
    wout = SubsetSum(i-1, T)
  return (with or wout)
```

General Pattern

General Pattern

Process at a node u of the recursion tree

```
if (u is a leaf) and (a solution is found) then
    return solution
```

```
else
```

```
    for any child  $v$  of  $u$  (i.e. any possible choice) do
        if no evidence that a solution does not exist at
            the subtree rooted at  $v$  then
            process  $v$ 
```

```
backtrack to the parent of  $u$ 
```

References

References

- Section 2.1 and 2.3 of the text book "algorithms" by Jeff Erikson