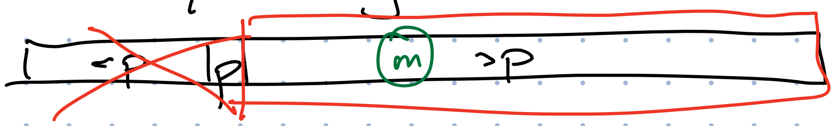
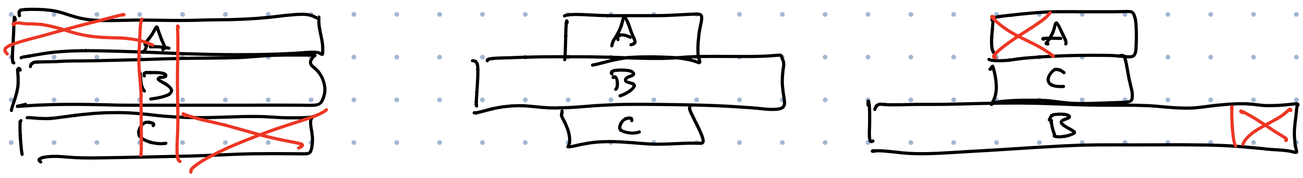
HW5 out — due next Tue

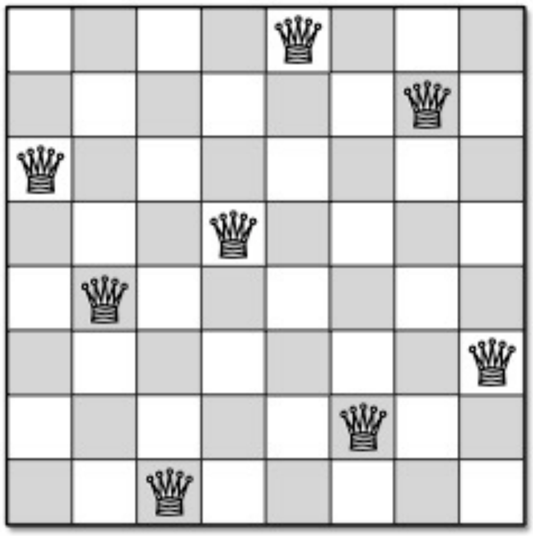
- Arithmetic takes time

```
result ← 1
for i ← 1 to n
    result ← result + result
```

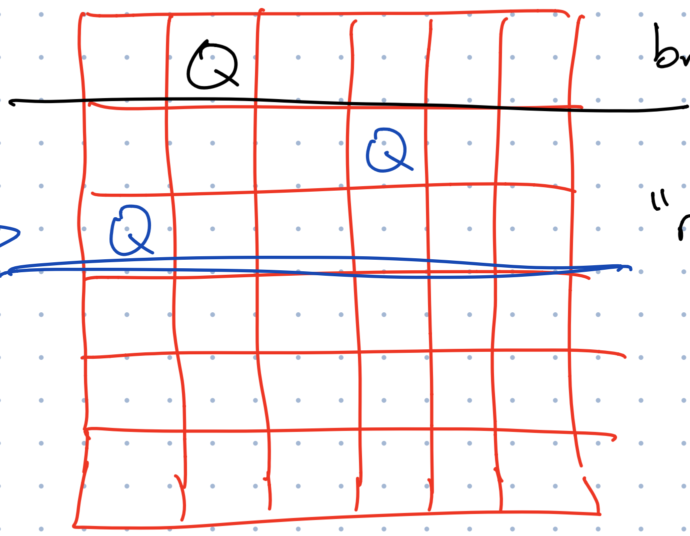
- Recursion sometimes requires generalization.

Median 





methodisches Tattonieren



brute force

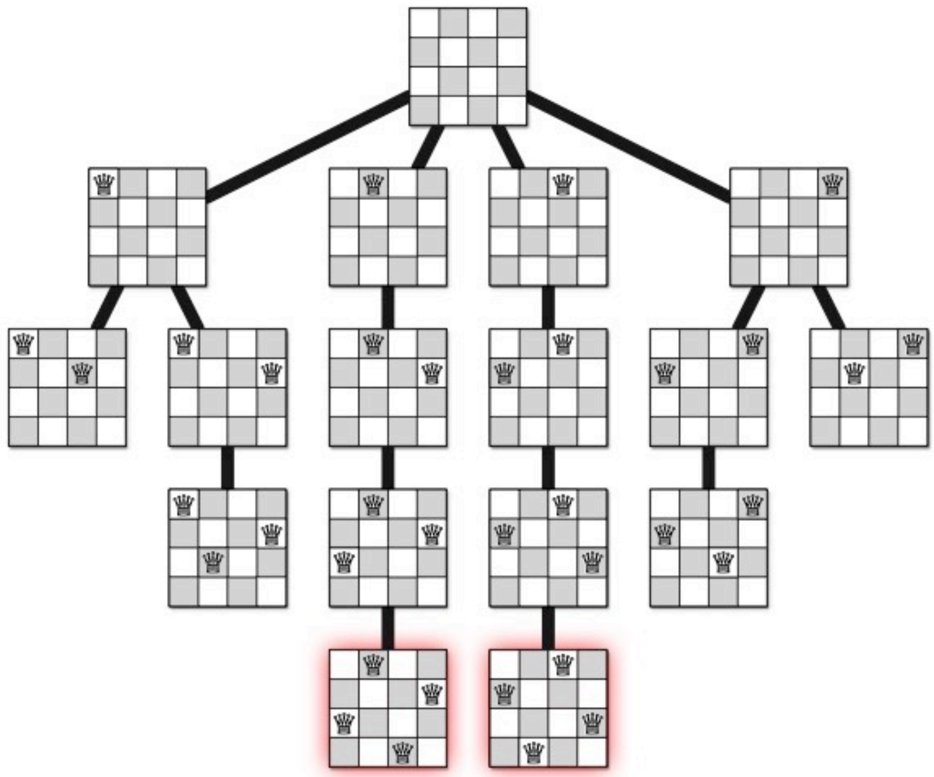
"recursion"

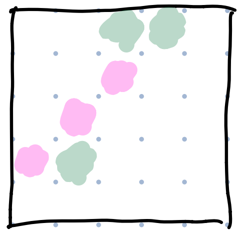
brute force →

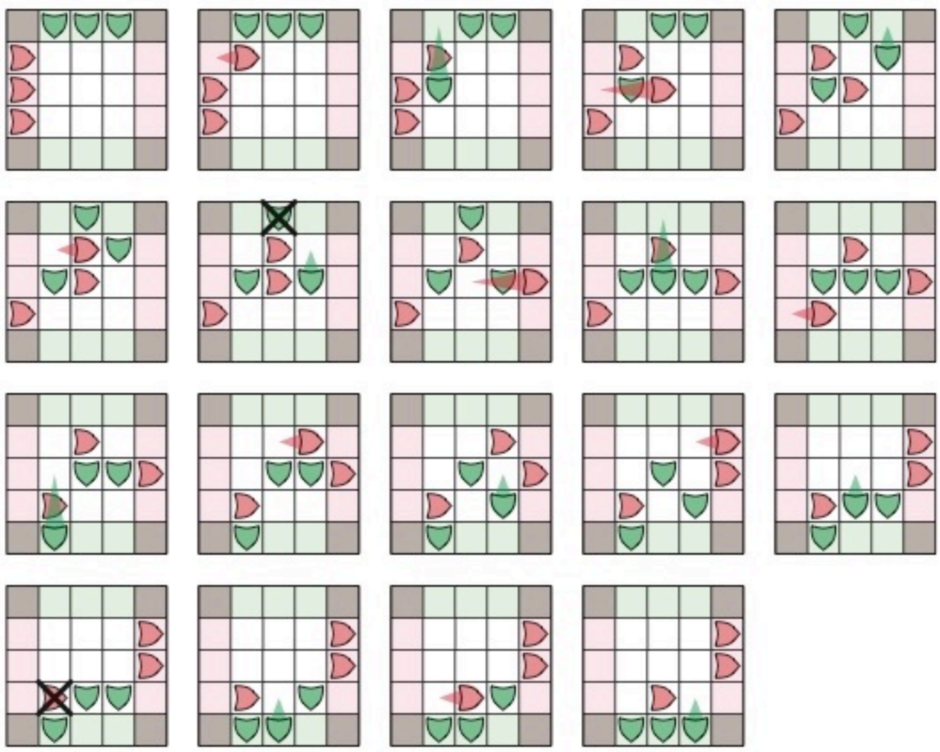
recurse

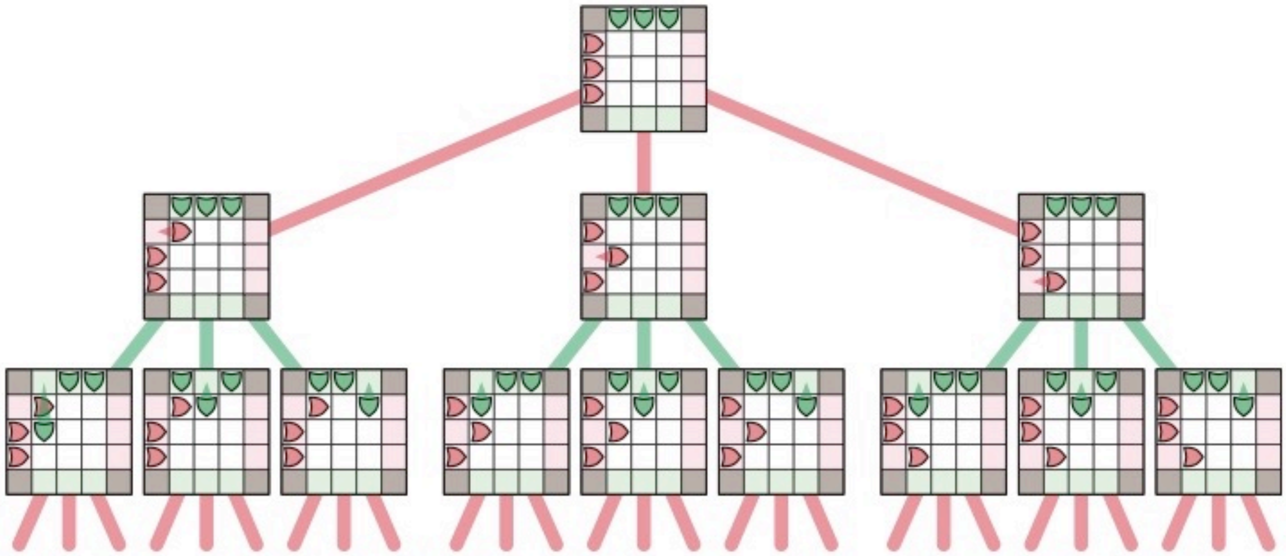
```
PlaceQueens(Q[1..n], r):
  if r = n+1
      print Q[1..n]
  else
      for j ← 1 to n
          legal ← True
          for i ← 1 to r − 1
              if (Q[i] = j) or (Q[i] = j + r − i) or (Q[i] = j − r + i)
                  legal ← False
          if legal
              Q[r] ← j
              PlaceQueens(Q[1..n], r + 1)      ⟨⟨Recursion!⟩⟩
```

Figure 2.2. Gauss and Laquière's backtracking algorithm for the n queens problem.









```
PlayAnyGame(X, player):
    if player has already won in state X
        return Good
    if player has already lost in state X
        return Bad
    for all legal moves X ⇝ Y
        if PlayAnyGame(Y, ¬player) = Bad
            return Good          ⟨⟨X ⇝ Y is a good move⟩⟩
    return Bad                   ⟨⟨There are no good moves⟩⟩
```

PRIMVS|DIGNITAS|IN|TAM|TENVISCIENTIANONPOTEST
ESSERESENIMSVNTPARVAEPROPEINSINGVLISLITTERIS
ATQVEINTERPVNCTIONIBUSVERBORVMOCCVPATAE

PRIMUS·DIGNITAS·IN·...

Input: string $A[1..n]$

Fixed subroutine IsWord(w) → true if w is a "word" / → false

Question: Can we partition A into a sequence of words?

HEARTHANDSATURNSPIN

HE|ARTHANDSATURNSPIN...

HEAR|THANDSATURNSPIN...

HEART|HANDSATURNSPIN...

HEARTH|ANDSATURNSPIN...

What problem are you really solving?

What do we need to remember about past decisions?

What are the recursive subproblems?

Is there a simple encoding of subproblems?

```
SPLITTABLE(A[1..n]):
    if n = 0
        return TRUE
    for i ← 1 to n
        if ISWORD(A[1..i])
            if SPLITTABLE(A[i+1..n])
                return TRUE
    return FALSE
```

```
⟨⟨Is the suffix A[i..n] Splittable?⟩⟩
SPLITTABLE(i):
    if i > n
        return TRUE
    for j ← i to n
        if ISWORD(i, j)
            if SPLITTABLE(j + 1)
                return TRUE
    return FALSE
```

$$T(n) \le O(n) + \sum_{i=1}^{n-1} T(i)$$

$$= O(2^n)$$