PRG

09.01.2018

Brute-Force Attack (Rohe Kraft)

→ Passwörter durch Ausprobieren zu entschlüsseln

- Anzahl möglicher Kombinationen – Formel:

Mögliche Kombinationen = mögliche Zeichenzahl **Kennwortlänge**

Passwort besteht aus	Mögliche Kombinationen	Benötigte Zeit zum Entschlüsseln
5 Zeichen (3 Kleinbuchstaben, 2 Zahlen)	36^5= 60.466.176	60.466.176 / 2.000.000.000 = **0,03 Sekunden**
7 Zeichen (1 Großbuchstabe, 6 Kleinbuchstaben)	52^7= 1.028.071.702.528	1.028.071.702.528 / 2.000.000.000 = 514 Sekunden = **ca. 9 Minuten**
8 Zeichen (4 Kleinbuchstaben, 2 Sonderzeichen, 2 Zahlen)	68^8= 457.163.239.653.376	457.163.239.653.376 / 2.000.000.000 = 228.581 Sekunden = **ca. 2,6 Tage**

Brute-Force

- angewendet auf verschlüsselte Passwörter, Dateien, Nachrichten & Informationen

Algorithmen:

- verzichten auf Optimierung

- setzen auf die Kraft des Prozessors

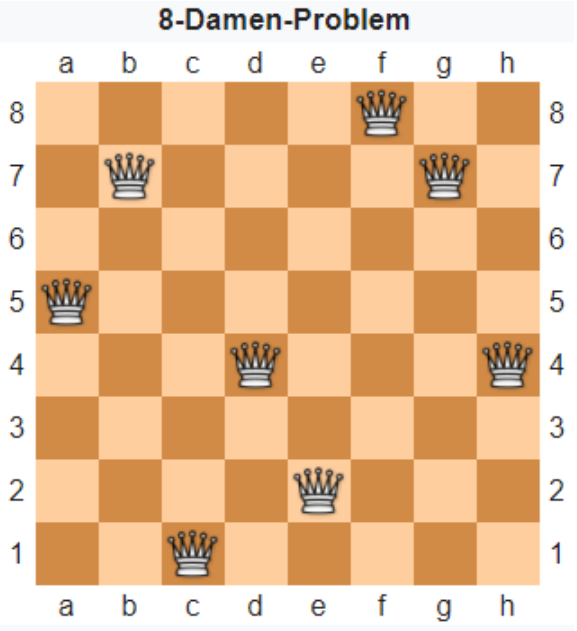
Strategie:

- Aufwand zur besseren Entwicklung lohnt sich nicht / besserer Algorithmus bisher nicht gefunden

- Beispiel: Selection Sort, Bubble Sort

→ Algorithmus verwendet Verfahren direkt an der Problembeschreibung

Beispiel



- n-Dameproblem (DisMod)

-> Damen so platzieren, dass sich keine Dame schlägt (Schachregeln)

Algorithmus:

Gegeben sei ein n*n Feld (Matrix), finde ein Sequenz s von Positionen der Länge n, derart, dass keine Position in s eine andere Position in s nach den Schachregeln bedroht.

Lösung:

Alle Folgen der Länge n von Zeilen- / Spalten Positionen (Paare aus dem Kreuzprodukt {1, .. n} x {1, .. n})

Prozedur „NeueSpalte“ - n-Damenproblem

- für jede Zeile von eins bis letzte Zeile tue folgendes:
  - prüfe ob das Feld von einer bereits gesetzten Dame bedroht ist
  - falls das Feld nicht bedroht sein sollte dann
    - setze eine Dame dort hin
    - setze die Bedrohung, die von der gesetzten Dame ausgeht
    - falls die letzte Spalte erreicht wurde dann
      - gib die Lösung aus
    - falls die letzte Spalte noch nicht erreicht wurde dann
      - Rufe nochmal die gleiche Prozedur auf und führe sie in der nächsten Spalte durch
    - zum Schluss lösche die Bedrohung und die Dame

Greedy (Gierig)

- Ziel: so viel wie möglich zu erreichen (Optimierungsproblem)

- Beispiel: Herausgabe von Wechselgeld

- Algorithmus:

Nimm jeweils immer die größte Münze unter dem Zielwert und ziehe sie von diesem ab. Verfahre derart bis Zielwert gleich null.

-> Rückgabe von 79 Cent: 79 = 50 + 20 + 5 + 2 + 2

- meist nicht so genau

-> weitere Beispiele: Kruskal, Prim, Dijkstra (Datenstrukturen)

Selection Sort

Beispiel: [6, 4, 2, 5, 1, 3, 7]

Sortierverfahren: vergleicht Zahl mit den Nachbarn

Brute-Force Ergebnis: [1, 2, 3, 4, 5, 6, 7]

Greedy Ergebnis: [2, 4, 6, 5, 1, 3, 7]

-> Greedy hat Optimum gefunden

Divide & Conquer (Teile & herrsche)

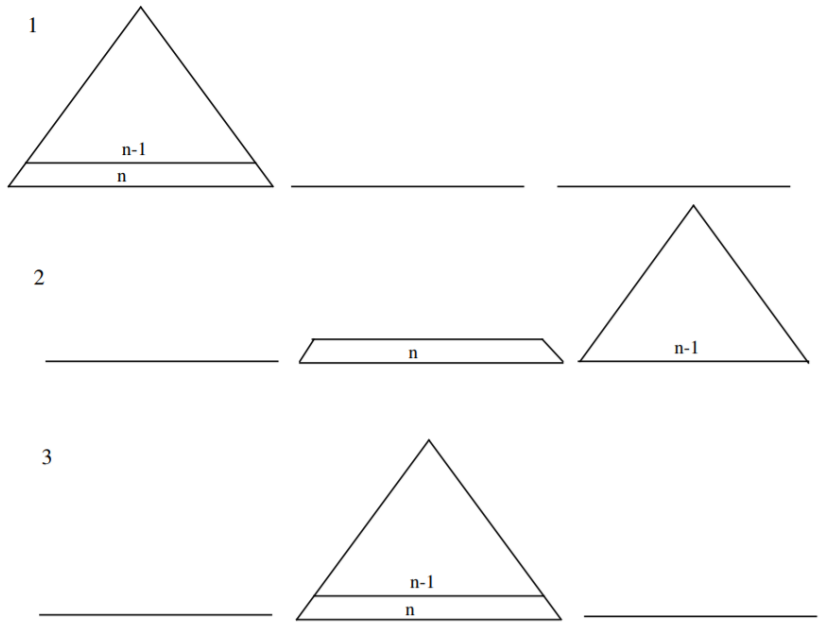
1. Teile das Problem in kleinere Unterprobleme (Divide)
2. Löse rekursiv die entstehenden Unterprobleme (Conquer)
3. Setze die Lösungen zusammen

Beispiel: Mergesort, Quicksort

Türme von Hanoi

Gegeben: Stapel von verschieden großen Scheiben von oben nach unten größer werdend

Aufgabe: Umstapeln auf einen anderen Stapel. Erlaubt ist ein weiterer Hilfsstapel Bedingung: Es darf niemals eine Scheibe auf einer kleineren liegen



Branch-and-Bound(Verzweigung und Schranken)

- lineares Optimierungsproblem der Graphentheorie

Algorithmen: Tiefensuche, Breitensuche

- **Verzweigung**: in mehrere Probleme aufteilen (Breiten- & Tiefensuche)
- **Schranke**: schneidet bestimmte Zweige des Baumes ab, welche nicht mehr betrachtet werden
- > spart Rechenzeit