Probabilistiska algoritmer

Algoritmer som använder slump

Exempel: Quicksort med slump

```
RandomQuicksort(v[i..j]) =
  if i<j then
    x ← v[random(i,j)]
    m ← Partition(v[i..j], x)
    RandomQuicksort(v[i..m-1])
    RandomQuicksort(v[m+1..j])
```

Analys: (beviset skippat)

Med sannolikhet $> 1-\frac{1}{n^k}$ tar RandomQuicksort tid $O(n \log n)$, för varje konstant k.

Las Vegas-algoritm — probabilistisk algoritm som alltid svarar rätt, men kan ta olika lång tid på sej.

EXEMPEL: MAX 3CNFSAT

GIVET EN MÄNGD $\{C_i\}_{i=1}^{n}$ MED KLAUSULER ÖVER BOOLESKA VARIABLER $X_1, \ldots, X_m$, HITTA EN VARIABELTILLDELNING SOM SATISFIERAR SÅ MÅNGA KLAUSULER SOM MÖJLIGT.

```
RandomMax3CNFSAT({C_i}) =
    FOR i ← 1 TO m DO
        x_i ← RANDOM(0,1)
    RETURN {x_i}
```

ANALYS:

TID: $O(n)$

$P[C_i \text{ INTE SATISFIERAD}] = \frac{1}{2} \cdot \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{8}$

Y = ANTAL SATISFIERADE KLAUSULER.

$E[Y] = n \cdot P[C_i \text{ SATISFIERAD}] = n\left(1 - \frac{1}{8}\right) = \frac{7}{8} n$

$\Rightarrow$ APPROXIMERAR MAX 3CNFSAT INOM $\frac{8}{7}$.

MONTE CARLO-ALGORITM — PROBABILISTISK ALGORITM SOM ALLTID GÅR SNABBT, MEN SOM BARA IBLAND GER BRA RESULTAT.

SLUMPELIMINERING

GÖR EN PROBABILISTISK ALGORITM DETERMINISTISK GENOM ATT ARBETA BORT SLUMPEN.

METOD 1: PRÖVA ALLA UTFALL

OM k SLUMPBITAR BEHÖVS I EN ALGORITM FINNS DET 2^k MÖJLIGA UTFALL. TESTA ALLA DESSA OCH PLOCKA DET BÄSTA!

PROBLEM: TAR LÅNG TID OM k ÄR STORT.

METOD 2: BETINGADE VÄNTEVÄRDEN

BERÄKNA $E[Y \mid x_1 = 0]$ OCH $E[Y \mid x_1 = 1]$

VÄLJ DET VÄRDE PÅ x_1 SOM GER STÖRST BETINGAT VÄNTEVÄRDE.

FORTSÄTT PÅ SAMMA SÄTT MED RESTERANDE x_i.

PROBLEM: BETINGADE VÄNTEVÄRDENA KAN VARA SVÅRA ATT BERÄKNA.

FLER METODER FINNS!

> GÅR DET ALLTID ATT SLUMPELIMINERA EN PROBABILISTISK POLYNOMISK ALGORITM TILL EN DETERMINISTISK POLYNOMISK ALGORITM?