Lösungen für den GetConnected 2015

Niklas Baumstark Tobias Heuer
Sebastian Ullrich Tung Nguyen Xuan

6. Februar 2015

SimpleSum

Aufgabe

- Gegeben n, $a_1, \ldots, a_n$
- Gesucht Summe $s = \sum_{i=1}^{n} a_i$

Lösung

- Straight-forward : for-Schleife
- $0 \leq s \leq 10^5 \Rightarrow$ int reicht aus

3n+1: Aufgabe

Gegeben

Paare von Zahlen i, j.

Gesucht

Maximale Collatz-Länge der Zahlen in $[i, j]$.

3n+1: Lösung

Collatz-Länge kann direkt durch Simulation berechnet werden:

```
int collatz_length(int n) {
    int len = 1;
    while (n > 1) {
        if (n % 2 == 0)
            n /= 2;
        else
            n = 3*n + 1;
        len++;
    }
    return len;
}
```

3n+1: Lösung

Es genügt nun, das Maximum im gegebenen Bereich auszugeben:

```
cin >> i >> j;
int answer = 0;
for (int n = i; n <= j; ++n)
    answer = max(answer, collatz_length(n));
cout << answer << "\n";
```

ChocoBar

Aufgabe

- Wie oft muss man eine $w \times h$ Tafel Schokolade minimal brechen, um Einzelstücke zu erhalten?

Lösung

- 1 Mal Brechen $\Rightarrow$ 1 Stück mehr
- Am Anfang: 1 Stück
- Am Ende: $w \cdot h$ Stücke

$\Rightarrow$ Man bricht immer genau $w \cdot h - 1$ Mal

Professor Snørbørden: Aufgabe

- Gegeben: Sequenz von korrekten und fehlerhaften Folien
- Schlafbedürfnis des Studenten
- Aufgabe: Schlaf so legen, dass Student möglichst wenig fehlerhafte Folien verschläft

Professor Snørbørden: Lösung

- Lösung: Alle Positionen durchprobieren.
- Beispiel: Student muss 5 Folien schlafen.
- 0110001001

Professor Snørbørden: Lösung

- Lösung: Alle Positionen durchprobieren.
- Beispiel: Student muss 5 Folien schlafen.
- 0110001001 3

Professor Snørbørden: Lösung

- Lösung: Alle Positionen durchprobieren.
- Beispiel: Student muss 5 Folien schlafen.
- 0110001001 -1 +1 3

Professor Snørbørden: Lösung

- Lösung: Alle Positionen durchprobieren.
- Beispiel: Student muss 5 Folien schlafen.
- 0110001001 3

Professor Snørbørden: Lösung

- Lösung: Alle Positionen durchprobieren.
- Beispiel: Student muss 5 Folien schlafen.
- 0110001001 +0 -0 3

Professor Snørbørden: Lösung

- Lösung: Alle Positionen durchprobieren.
- Beispiel: Student muss 5 Folien schlafen.
- 0110001001 3

Professor Snørbørden: Lösung

- Lösung: Alle Positionen durchprobieren.
- Beispiel: Student muss 5 Folien schlafen.
- 01100010 01 -0 +1 3

Professor Snørbørden: Lösung

- Lösung: Alle Positionen durchprobieren.
- Beispiel: Student muss 5 Folien schlafen.
- 0110001001 4

Professor Snørbørden: Lösung

- Lösung: Alle Positionen durchprobieren.
- Beispiel: Student muss 5 Folien schlafen.
- 01100010 01 -1 +1 4

Professor Snørbørden: Lösung

- Lösung: Alle Positionen durchprobieren.
- Beispiel: Student muss 5 Folien schlafen.
- 0110001001 4

Professor Snørbørden: Lösung

- Lösung: Alle Positionen durchprobieren.
- Beispiel: Student muss 5 Folien schlafen.
- 0110001001 -1 +0 4

Professor Snørbørden: Lösung

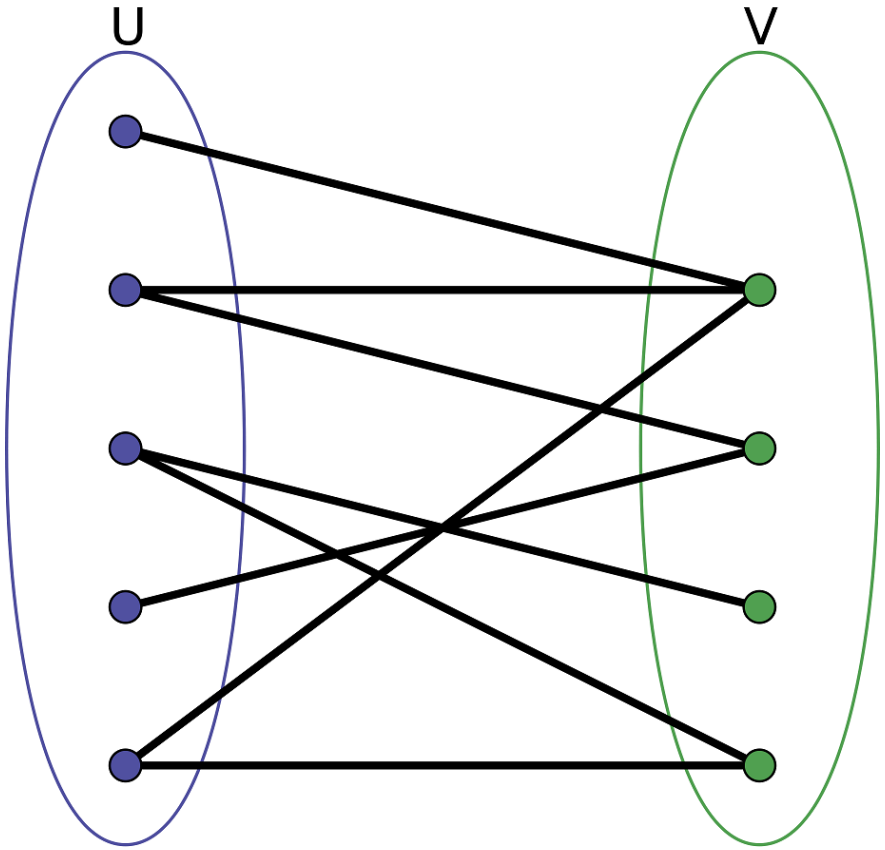
- Lösung: Alle Positionen durchprobieren.
- Beispiel: Student muss 5 Folien schlafen.
- 0110001001 3

Bicoloring

Problem

Gegeben ist ein ungerichteter zusammenhängender Graph $G = (V, E)$. Entscheide, ob die Knoten von G mit zwei Farben einfärbar sind, sodass keine zwei adjazenten Knoten die gleiche Farbe haben (d.h. G ist bipartit).

Bicoloring: Beispiel



Bicoloring: Lösung

Lösung

- Führe Tiefensuche von ein beliebigen Knoten aus.
- Markiere Knoten abwechselnd mit 0 und 1 im Verlauf der Tiefensuche (0 und 1 repräsentieren die Farben).
- Wird ein Knoten besucht, welcher schon vorher besucht wurde und die gleiche Farbe hat wie sein Vorgänger, dann gib „*NOT BICOLORABLE.*" aus.
- Ist jeder Knoten besucht und kein Konflikt trat auf, gib „*BICOLORABLE.*" aus.

Lawnmower: Aufgabe

Gegeben

- Tabelle $A_{n \times m}$
- Anfang : $A_{i,j} = 100 \ \forall i, j$
- 2 Operationen, beliebig oft auszuführen
  - r i h : $\forall 0 \leq j < m$, $A_{i,j} > h \Rightarrow A_{i,j} = h$
  - c j h : $\forall 0 \leq i < n$, $A_{i,j} > h \Rightarrow A_{i,j} = h$
- Endzustand $B_{n \times m}$

Gesucht

- Ist B von A erreichbar?

Lawnmower: Lösung

Was, wenn B erreicht ist?

Lawnmower: Lösung

Was, wenn B erreicht ist?

- rMax[i] $= \max\{\ B_{i,j}, 0 \leq j < m\ \}$
- cMax[j] $= \max\{\ B_{i,j}, 0 \leq i < n\ \}$

Lawnmower: Lösung

Was, wenn B erreicht ist?

- rMax[i] $= \max\{\ B_{i,j}, 0 \leq j < m\ \}$
- cMax[j] $= \max\{\ B_{i,j}, 0 \leq i < n\ \}$
- Letzte Operation an der Stelle i, j ist
  - r i h $\Rightarrow B_{i,j} = rMax[i]$
  - c j h $\Rightarrow B_{i,j} = cMax[j]$
  - $\Rightarrow B_{i,j} \geq min(rMax[i], cMax[j])$ (*)

Lawnmower: Lösung

Was, wenn B erreicht ist?

- rMax[i] $= \max\{\ B_{i,j}, 0 \leq j < m\ \}$
- cMax[j] $= \max\{\ B_{i,j}, 0 \leq i < n\ \}$
- Letzte Operation an der Stelle i, j ist
  - r i h $\Rightarrow B_{i,j} = rMax[i]$
  - c j h $\Rightarrow B_{i,j} = cMax[j]$
  - $\Rightarrow B_{i,j} \geq min(rMax[i], cMax[j])$ (*)

Ist (*) erfüllt, ist B erreichbar ?

Lawnmower: Lösung

Was, wenn B erreicht ist?

- rMax[i] $= \max\{\ B_{i,j}, 0 \leq j < m\ \}$
- cMax[j] $= \max\{\ B_{i,j}, 0 \leq i < n\ \}$
- Letzte Operation an der Stelle i, j ist
  - r i h $\Rightarrow B_{i,j} = rMax[i]$
  - c j h $\Rightarrow B_{i,j} = cMax[j]$
  - $\Rightarrow B_{i,j} \geq min(rMax[i], cMax[j])$ (*)

Ist (*) erfüllt, ist B erreichbar ?

- Ja, konstruiere wie folgt : $\forall i, j$
  - $B_{i,j} = rMax[i] \Rightarrow$ r i $B_{i,j}$
  - $B_{i,j} = cMax[i] \Rightarrow$ c j $B_{i,j}$

Lawnmower: Lösung

Was, wenn B erreicht ist?

- rMax[i] $= \max\{\ B_{i,j}, 0 \leq j < m\ \}$
- cMax[j] $= \max\{\ B_{i,j}, 0 \leq i < n\ \}$
- Letzte Operation an der Stelle i, j ist
  - r i h $\Rightarrow B_{i,j} = rMax[i]$
  - c j h $\Rightarrow B_{i,j} = cMax[j]$
  - $\Rightarrow B_{i,j} \geq min(rMax[i], cMax[j])$ (*)

Ist (*) erfüllt, ist B erreichbar ?

- Ja, konstruiere wie folgt : $\forall i, j$
  - $B_{i,j} = rMax[i] \Rightarrow$ r i $B_{i,j}$
  - $B_{i,j} = cMax[i] \Rightarrow$ c j $B_{i,j}$
- rMax, cMax linear berechnenbar $\Rightarrow \mathcal{O}(n \cdot m)$

Elevator: Aufgabe

Gegeben

- $a, b, c \in \mathbb{N}_{>0}$
- $1 \leq x \leq 10^{12}$

Gefragt

- Ist x darstellbar als $x = 1 + \alpha \cdot a + \beta \cdot b + \gamma \cdot c$ mit $\alpha, \beta, \gamma \in \mathbb{N}_{\geq 0}$?

- Standardansatz mit BFS oder DP $= \Omega(x) \rightarrow$ zu langsam

Elevator: Lösung

Ansatz

- Falls y erreichbar, dann auch $y + kc$ erreichbar für alle $k \in \mathbb{N}_{\geq 0}$
- Definiere $f(n) :=$ niedrigstes erreichbares Stockwerk y mit $y \equiv m \pmod{c}$ oder ∞.
- Rekurrenz:
  $f(1) = 1$
  $f(n) = \min\{f((n-a) \textbf{ mod } c) + a, f((n-b) \textbf{ mod } c) + b\}$
- Lösung: x ist erreichbar, falls $x \geq f(x \textbf{ mod } c)$

Elevator: Lösung (2)

Ansatz

- Rekurrenz:
  $f(1) = 1$
  $f(n) = \min\{f(n-a) + a, f(n-b) + b\}$

- Löse mit Dijkstra-Algorithmus:
  - Knotenmenge $V = \{0, \ldots, c-1\}$
  - Kante von n zu $(n+a)$ **mod** c mit Gewicht a. Analog für b.
- Verarbeite Knoten $n \in V$ nach aufsteigendem $f(n)$.
- Laufzeit: $\mathcal{O}(c \cdot \log c + q)$, $q =$ Anzahl Queries

Sample: $a = 3, b = 4, c = 8$.

n	$f(n)$
0	∞
1	1
2	∞
3	∞
4	∞
5	∞
6	∞
7	∞

Sample: $a = 3, b = 4, c = 8$.

n	$f(n)$
0	∞
1	1
2	∞
3	∞
4	∞
5	∞
6	∞
7	∞

Sample: $a = 3, b = 4, c = 8$.

n	$f(n)$
0	∞
1	1
2	∞
3	∞
4	4
5	5
6	∞
7	∞

Sample: $a = 3, b = 4, c = 8$.

n	$f(n)$
0	∞
1	2
2	∞
3	∞
4	4
5	5
6	∞
7	∞

Sample: $a = 3, b = 4, c = 8$.

n	$f(n)$
0	8
1	2
2	∞
3	∞
4	4
5	5
6	∞
7	7

Sample: $a = 3, b = 4, c = 8$.

n	$f(n)$
0	8
1	2
2	∞
3	∞
4	4
5	5
6	∞
7	7

Sample: $a = 3, b = 4, c = 8$.

n	$f(n)$
0	8
1	2
2	∞
3	∞
4	4
5	5
6	∞
7	7

Sample: $a = 3, b = 4, c = 8$.

n	$f(n)$
0	8
1	2
2	10
3	11
4	4
5	5
6	∞
7	7

Sample: $a = 3, b = 4, c = 8$.

n	$f(n)$
0	8
1	2
2	10
3	11
4	4
5	5
6	∞
7	7

Sample: $a = 3, b = 4, c = 8$.

n	$f(n)$
0	8
1	2
2	10
3	11
4	4
5	5
6	∞
7	7

Sample: $a = 3, b = 4, c = 8$.

n	$f(n)$
0	8
1	2
2	10
3	11
4	4
5	5
6	14
7	7

Sample: $a = 3, b = 4, c = 8$.

n	$f(n)$
0	8
1	2
2	10
3	11
4	4
5	5
6	14
7	7

Sample: $a = 3, b = 4, c = 8$.

n	$f(n)$
0	8
1	2
2	10
3	11
4	4
5	5
6	14
7	7

Sample: $a = 3, b = 4, c = 8$.

n	$f(n)$
0	8
1	2
2	10
3	11
4	4
5	5
6	14
7	7