

Mathematik für Informatiker
Kombinatorik, Stochastik und Statistik
Übungsblatt 1

Tom Paßberg , Iain Dorsch

Aufgabe 1

Zu zeigen: Für alle $n \in \mathbb{N}_0$ gilt

$$\sum_{j=0}^n \binom{n}{j}^2 = \binom{2n}{n}$$

Beweis:

$$\sum_{j=0}^n \binom{n}{j}^2 = \sum_{j=0}^n \binom{n}{j} \cdot \binom{n}{j} \quad (1)$$

$$= \sum_{j=0}^n \binom{n}{j} \cdot \binom{n}{n-j} \quad (2)$$

$$= \binom{n+n}{n} \quad (3)$$

$$= \binom{2n}{n} \quad (4)$$

(2) folgt aus Skript 1.2.5

(3) folgt aus Skript 1.2.12.

Aufgabe 2

Die Anzahl der kürzesten Pfade c wird gegeben durch die Formel:

$$\begin{aligned} c &= \frac{(n-1+m-1)!}{(n-1)! \cdot (m-1)!} \\ &= \binom{n+m-2}{n-1} \end{aligned}$$

Beweis:

Induktionsanfang: Für $n = 1$ und $m = 1$ ist die Position von Start und Ziel identisch. Es gibt trivial einen kürzesten Pfad.

$$c = \binom{0}{0} = 1$$

Induktionsschritt: Hallo

Aufgabe 3

$$a = |\{x \mod 3 = 0 \mid 0 \leq x \leq 100000\}| = \left\lfloor \frac{100000}{3} \right\rfloor$$

$$b = |\{x \mod 5 = 0 \mid 0 \leq x \leq 100000\}| = \left\lfloor \frac{100000}{5} \right\rfloor$$

$$c = |\{x \mod 7 = 0 \mid 0 \leq x \leq 100000\}| = \left\lfloor \frac{100000}{7} \right\rfloor$$

$$d = |\{x \mod 11 = 0 \mid 0 \leq x \leq 100000\}| = \left\lfloor \frac{100000}{11} \right\rfloor$$

$$ab = |\{x \mod 3 = 0 \wedge x \mod 5 = 0 \mid 0 \leq x \leq 100000\}| = \left\lfloor \frac{100000}{3 \cdot 5} \right\rfloor$$

$$ac = |\{x \mod 3 = 0 \wedge x \mod 7 = 0 \mid 0 \leq x \leq 100000\}| = \left\lfloor \frac{100000}{3 \cdot 7} \right\rfloor$$

$$ad = |\{x \mod 3 = 0 \wedge x \mod 11 = 0 \mid 0 \leq x \leq 100000\}| = \left\lfloor \frac{100000}{3 \cdot 11} \right\rfloor$$

$$bc = |\{x \mod 5 = 0 \wedge x \mod 7 = 0 \mid 0 \leq x \leq 100000\}| = \left\lfloor \frac{100000}{5 \cdot 7} \right\rfloor$$

$$bd = |\{x \mod 5 = 0 \wedge x \mod 11 = 0 \mid 0 \leq x \leq 100000\}| = \left\lfloor \frac{100000}{5 \cdot 11} \right\rfloor$$

$$cd = |\{x \mod 7 = 0 \wedge x \mod 11 = 0 \mid 0 \leq x \leq 100000\}| = \left\lfloor \frac{100000}{7 \cdot 11} \right\rfloor$$

$$abc = |\{x \mod 3 = 0 \wedge x \mod 5 = 0 \wedge x \mod 7 = 0 \mid 0 \leq x \leq 100000\}| = \left\lfloor \frac{100000}{3 \cdot 5 \cdot 7} \right\rfloor$$

$$abd = |\{x \mod 3 = 0 \wedge x \mod 5 = 0 \wedge x \mod 11 = 0 \mid 0 \leq x \leq 100000\}| = \left\lfloor \frac{100000}{3 \cdot 5 \cdot 11} \right\rfloor$$

$$acd = |\{x \mod 3 = 0 \wedge x \mod 7 = 0 \wedge x \mod 11 = 0 \mid 0 \leq x \leq 100000\}| = \left\lfloor \frac{100000}{3 \cdot 7 \cdot 11} \right\rfloor$$

$$bcd = |\{x \mod 5 = 0 \wedge x \mod 7 = 0 \wedge x \mod 11 = 0 \mid 0 \leq x \leq 100000\}| = \left\lfloor \frac{100000}{5 \cdot 7 \cdot 11} \right\rfloor$$

$$\begin{aligned} abcd &= |\{x \mod 3 = 0 \wedge x \mod 5 = 0 \wedge x \mod 7 = 0 \wedge x \mod 11 = 0 \mid 0 \leq x \leq 100000\}| \\ &= \left\lfloor \frac{100000}{3 \cdot 5 \cdot 7 \cdot 11} \right\rfloor \end{aligned}$$

$$\begin{aligned} \text{result} &= a + b + c + d - ab - ac - ad - bc - bd - cd + abc + abd + acd + bcd - abcd \\ &= 58441 \end{aligned}$$

Aufgabe 4

a)

Für jede weitere Stufe wird die vorherige Menge kopiert und um das nächste Element erweitert.

Die neuen und die alten Elemente werden in einer neuen Menge zusammengefasst.

- $n = 0$
 $\{\{\}\}$
- $n = 1$
 $\{\{\}, \{1\}\}$
- $n = 2$
 $\{\{\}, \{1\}, \{2\}, \{1, 2\}\}$
- $n = 3$
 $\{\{\}, \{1\}, \{2\}, \{1, 2\}, \{3\}, \{1, 3\}, \{2, 3\}, \{1, 2, 3\}\}$
- $n = 4$
 $\{\{\}, \{1\}, \{2\}, \{1, 2\}, \{3\}, \{1, 3\}, \{2, 3\}, \{1, 2, 3\}, \{4\}, \{1, 4\}, \{2, 4\}, \{1, 2, 4\}, \{3, 4\}, \{1, 3, 4\}, \{2, 3, 4\}, \{1, 2, 3, 4\}\}$

b)

Beschreibung

Für $n = 0$ gibt der Algorithmus eine Liste mit einer leeren Liste zurück. (Eine Menge in der sich eine leere Menge befindet)

Für $n > 0$ wird die Funktion rekursiv aufgerufen um die Teilmengen von $\{1, \dots, n-1\}$ zu berechnen.

Die Teilmengen von $\{1, \dots, n\}$ ergeben sich aus den Teilmengen von $\{1, \dots, n-1\}$, indem von jeder Teilmenge aus $\{1, \dots, n-1\}$ eine Kopie erstellt wird, in die n hinzugefügt wird.

Funktion zur Berechnung der Teilmengen von $\{1, \dots, n\}$

```
fn teilmengen_rec(n: u8) -> Vec<Vec<u8>> {
    if n == 0 {
        return vec![vec![]];
    }
    teilmengen_rec(n - 1)
        .into_iter()
        .flat_map(|old| {
            let mut new = old.clone();
            new.push(n);
            [old, new]
        })
        .collect()
}
```

Funktionsaufrufe für $n = 1, \dots, 20$

```
fn main() {
    for i in 1..=20 {
        let teilmengen = teilmengen_rec(i);
        println!(
            "Anzahl der Teilmengen von  $\{\{1, \dots, \{\}\}\}$ :  $\{\}$ ",
            i, teilmengen.len()
        );
        if i <= 4 {
            println!("Teilmengen von  $\{\{1, \dots, \{\}\}\}$ :  $\{:\}$ ", i, teilmengen);
        }
    }
}
```

Output:

Anzahl der Teilmengen von $\{1, \dots, 1\}$: 2
Teilmengen von $\{1, \dots, 1\}$: $[[], [1]]$
Anzahl der Teilmengen von $\{1, \dots, 2\}$: 4
Teilmengen von $\{1, \dots, 2\}$: $[[], [2], [1], [1, 2]]$
Anzahl der Teilmengen von $\{1, \dots, 3\}$: 8
Teilmengen von $\{1, \dots, 3\}$: $[[], [3], [2], [2, 3], [1], [1, 3], [1, 2], [1, 2, 3]]$
Anzahl der Teilmengen von $\{1, \dots, 4\}$: 16
Teilmengen von $\{1, \dots, 4\}$: $[[], [4], [3], [3, 4], [2], [2, 4], [2, 3], [2, 3, 4], [1], [1, 4], [1, 3], [1, 3, 4], [1, 2], [1, 2, 4], [1, 2, 3], [1, 2, 3, 4]]$
Anzahl der Teilmengen von $\{1, \dots, 5\}$: 32
Anzahl der Teilmengen von $\{1, \dots, 6\}$: 64
Anzahl der Teilmengen von $\{1, \dots, 7\}$: 128
Anzahl der Teilmengen von $\{1, \dots, 8\}$: 256
Anzahl der Teilmengen von $\{1, \dots, 9\}$: 512
Anzahl der Teilmengen von $\{1, \dots, 10\}$: 1024
Anzahl der Teilmengen von $\{1, \dots, 11\}$: 2048
Anzahl der Teilmengen von $\{1, \dots, 12\}$: 4096
Anzahl der Teilmengen von $\{1, \dots, 13\}$: 8192
Anzahl der Teilmengen von $\{1, \dots, 14\}$: 16384
Anzahl der Teilmengen von $\{1, \dots, 15\}$: 32768
Anzahl der Teilmengen von $\{1, \dots, 16\}$: 65536
Anzahl der Teilmengen von $\{1, \dots, 17\}$: 131072
Anzahl der Teilmengen von $\{1, \dots, 18\}$: 262144
Anzahl der Teilmengen von $\{1, \dots, 19\}$: 524288
Anzahl der Teilmengen von $\{1, \dots, 20\}$: 1048576

Aufgabe 5

Funktion um die Anzahl der Zahlen zwischen 1 und n zu berechnen, die durch mindestens einen der Teiler teilbar sind.

```
use rayon::iter::{IntoParallelIterator, ParallelIterator};
```

```
fn count_numbers(n: u64, teiler: &Vec<u64>) -> usize {
    (1..=n).into_par_iter()
        .filter(|&n| teiler.iter().any(|&t| n % t == 0))
        .count()
}
```

Funktionsaufrufe für $n = 10, 100, \dots, 10000000000$:

```
fn main() {
    let teiler: Vec<u64> = vec![3, 5, 7, 11];
    for n in (1..=10).scan(1, |n, _| { *n *= 10; Some(*n) }) {
        println!(
            "{:11} gerade Zahlen zwischen 1 und {n:11}
            sind durch mindestens einen der Teiler {} teilbar.",
            count_numbers(n, &teiler),
            teiler.iter().map(|&t|
                t.to_string()).collect::<Vec<String>>().join(", ")
        );
    }
}
```

Output:

```
6 Zahlen zwischen 1 und 10 sind durch mindestens einen
der Teiler 3, 5, 7, 11 teilbar.
59 Zahlen zwischen 1 und 100 sind durch mindestens einen
der Teiler 3, 5, 7, 11 teilbar.
585 Zahlen zwischen 1 und 1000 sind durch mindestens einen
der Teiler 3, 5, 7, 11 teilbar.
5845 Zahlen zwischen 1 und 10000 sind durch mindestens einen
der Teiler 3, 5, 7, 11 teilbar.
58441 Zahlen zwischen 1 und 100000 sind durch mindestens einen
der Teiler 3, 5, 7, 11 teilbar.
584416 Zahlen zwischen 1 und 1000000 sind durch mindestens einen
der Teiler 3, 5, 7, 11 teilbar.
5844156 Zahlen zwischen 1 und 10000000 sind durch mindestens einen
der Teiler 3, 5, 7, 11 teilbar.
58441559 Zahlen zwischen 1 und 100000000 sind durch mindestens einen
der Teiler 3, 5, 7, 11 teilbar.
584415585 Zahlen zwischen 1 und 1000000000 sind durch mindestens einen
```

der Teiler 3, 5, 7, 11 teilbar.
5844155845 Zahlen zwischen 1 und 10000000000 sind durch mindestens einen
der Teiler 3, 5, 7, 11 teilbar.