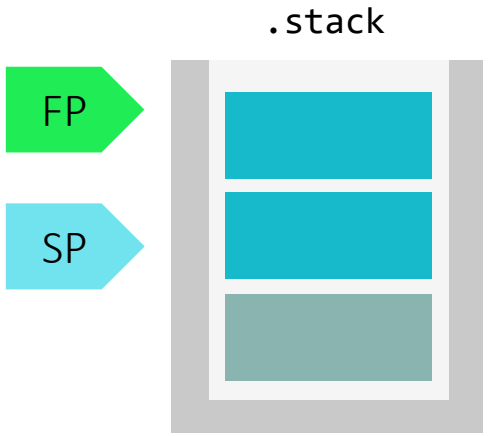
Einführung in die Programmierung



Präsenzblatt 8

Wintersemester 2025/2026

Arbeitsgruppe Systemsoftware
Angewandte Informatik 12

technische universität dortmund

Arbeitsgruppe Systemsoftware

Kolmogorov vermutete um 1956 folgendes:
Zur Berechnung des Produktes zweier Binärzahlen existieren keine Algorithmen die schneller sind als $\Theta(n^2)$

- Tatsächlich widerlegt der Karazuba-Algorithmus diese Vermutung!
- Der Karazuba-Algorithmus besitzt eine Laufzeit von $\Theta\left(n^{\log_2 3}\right) \approx \Theta(n^{1.58})$
- Nach dem Karazuba-Algorithmus wurden viele weitere noch schnellere Algorithmen für die Multiplikation zweier Zahlen gefunden

- Toom-Cook-**Algorithmus** (1965): Bitvektoren werden nicht nur in zwei, sondern in mehrere Teile zerlegt → Eine **Optimale Zerlegung** erfordert dann für $v + 1$ Teile gerade einmal $2v + 1$ Multiplikationen
  - **Variante mit fester Teilung**: $\mathcal{O}\left(n^{c'+1}\right)$ mit von Teilung abhängigem c'
  - **Variante mit variabler Teilung**: $\mathcal{O}\left(n \cdot \log_2(n) \cdot 2^{\sqrt[2]{2\log_2 n}}_{+}\right)$ im Worst-Case

Zerlege x und y in k Teile

```
37  for (size_t i = 0; i < k; i++) {
38    xParts[i] = (x >> addShift(i, partSize)) & mask;
39    yParts[i] = (y >> addShift(i, partSize)) & mask;
40  }
```

Berechne die Interpolationspunkte

```
41  for (size_t i = 0; i < k; i++) {
42    for (size_t j = 0; j < k; j++) {
43      p[i + j] += toomCook(xParts[i], yParts[j], partSize);
44    }
45  }
```

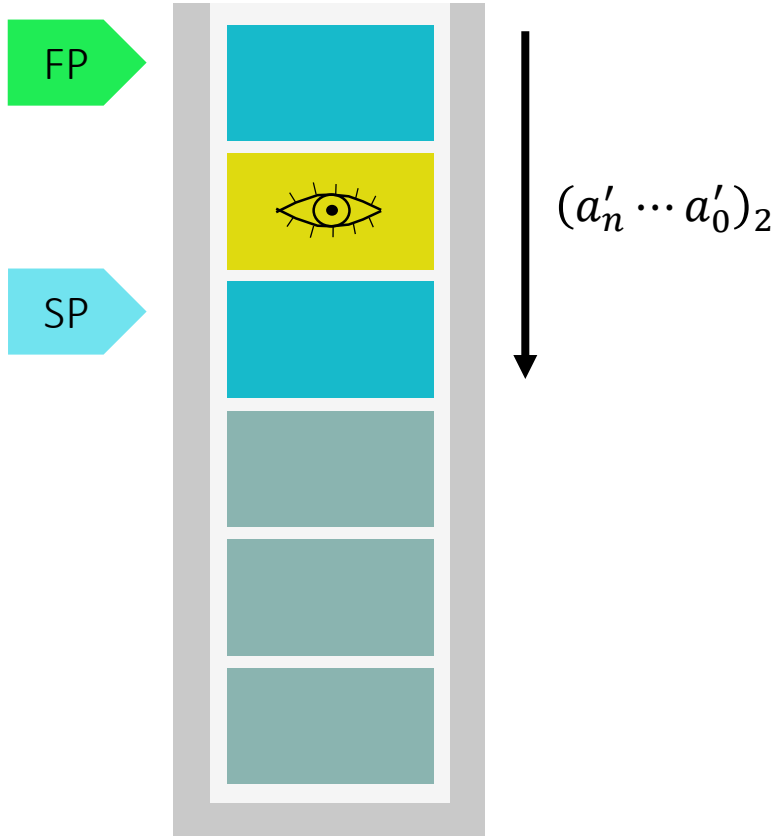
Rekonstruiere das Ergebnis durch Interpolation

```
46  for (size_t i = 0; i < v; i++) {result += (p[i]<<addShift(i, partSize)); }
```

Jahr	Algorithmus	Laufzeit
?	Russische Bauernmultiplikation (Add-Shift)	$\mathcal{O}(n^2)$
1962	Karazuba-Algorithmus	$\mathcal{O}\left(n^{\log_2 3}\right) \approx \mathcal{O}(n^{1.58})$
1969	Toom-Cook-Algorithmus (Knuth)	$\mathcal{O}\left(n \cdot \log_2(n) \cdot 2^{\sqrt[2]{2\log_2 n}_{+}}\right)$
1971	Schönehagen-Strassen-Algorithmus	$\mathcal{O}(n \log n \log\log n)$
2007	Fürer-Algorithmus	$\mathcal{O}\left(n \log n \cdot K^{\log^* n}\right)$
2017	Van der Hoeven-Lecerf-Algorithmus	$\mathcal{O}\left(n \log n \cdot 8^{\log^* n}\right)$
2019/21	Van der Hoeven-Algorithmus	$\Theta(n \log n)$

- Schauen Sie sich die Stack-Animation (`stack.pdf`) im Moodle an
- Welchen Bereich grenzen der **Stackpointer** und **Framepointer** im Stack ein?
  - Stackpointer und Framepointer grenzen jeweils den Arbeitsbereich der aktuell gerufenen Funktion ein
  - Dieser Bereich wird als **Stapelrahmen** (**Stackframe**) bezeichnet
  - In den Arbeitsbereichen liegen jeweils die lokalen Variablen der Funktion
- Wodurch wird die maximale Tiefe einer Rekursion beschränkt?
  - Die Rekursionstiefe ist schlicht durch die Tatsache beschränkt, dass der Stack irgendwann ggf. in den Heap hineinwächst und dort Daten überschreibt → **Stapelüberlauf** (**Stack overflow**)

Stapelrahmen (Stackframe)



Stapelüberlauf