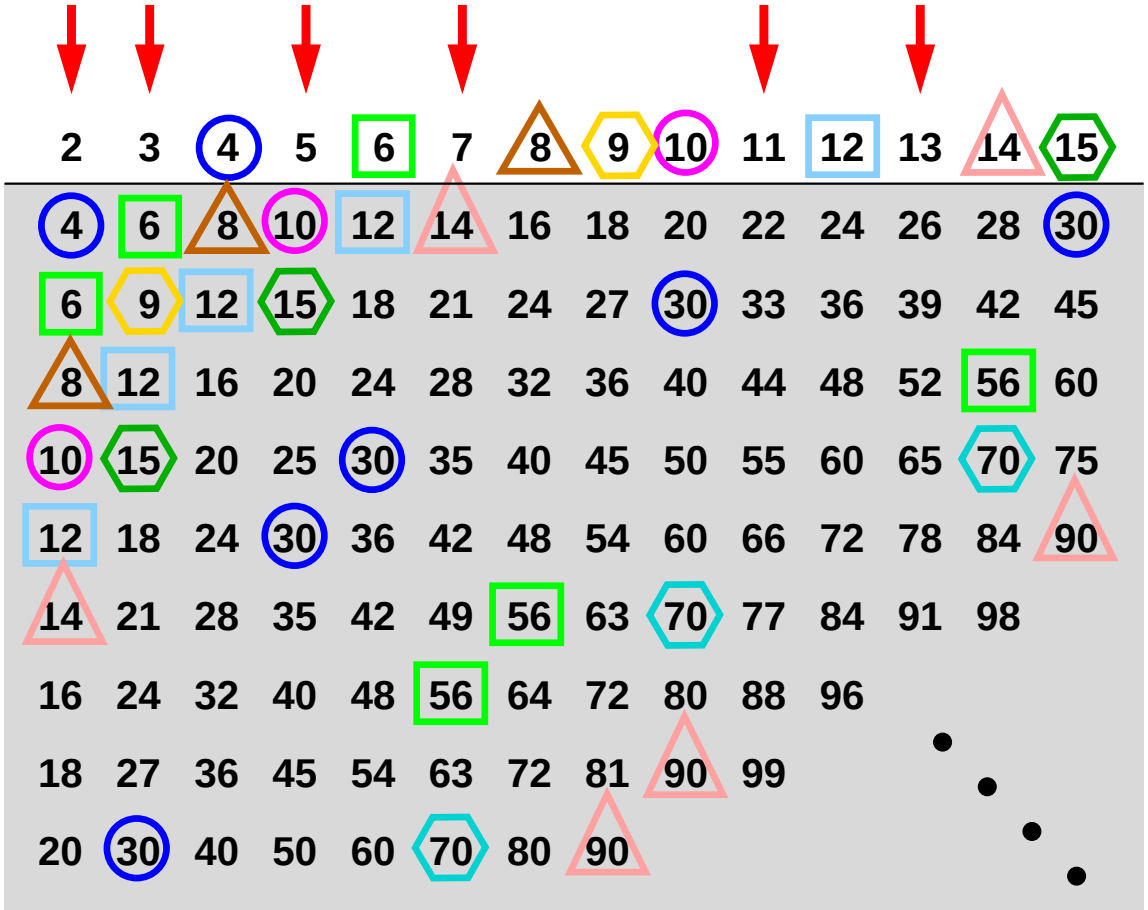
Wie schnell ist schnell genug?
Beispiel Primzahltabelle

Martin Oellrich

TU Berlin

6. Mai 2006



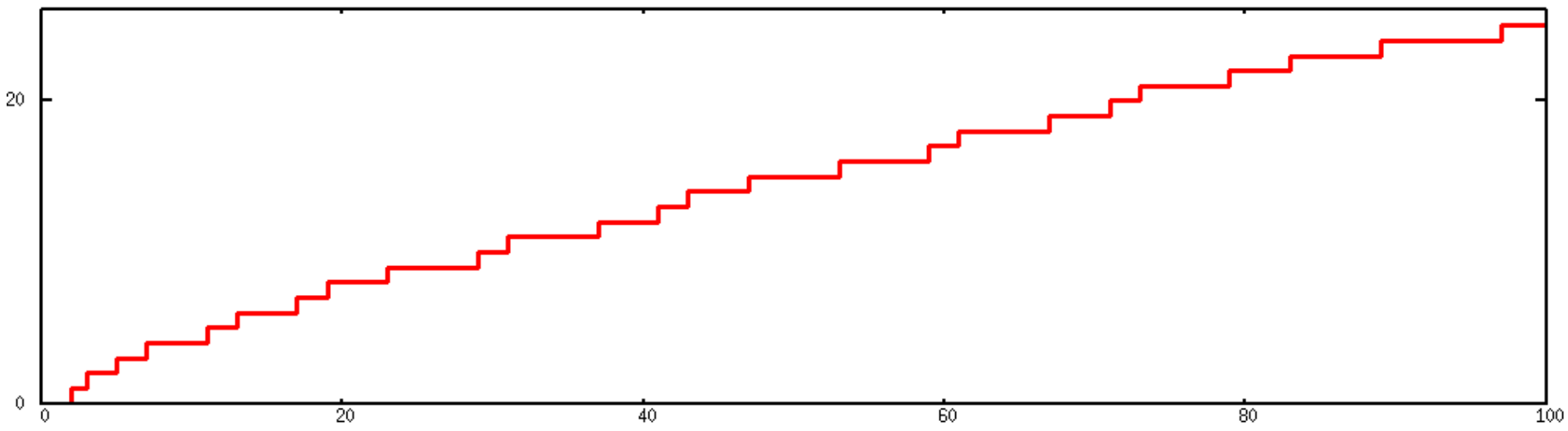
Übersicht

1. Wozu eine Primzahltabelle?
2. Primzahlen direkt prüfen
3. Sieb des Eratosthenes
4. Ergebnisse und Ausblick

Verteilung der Primzahlen

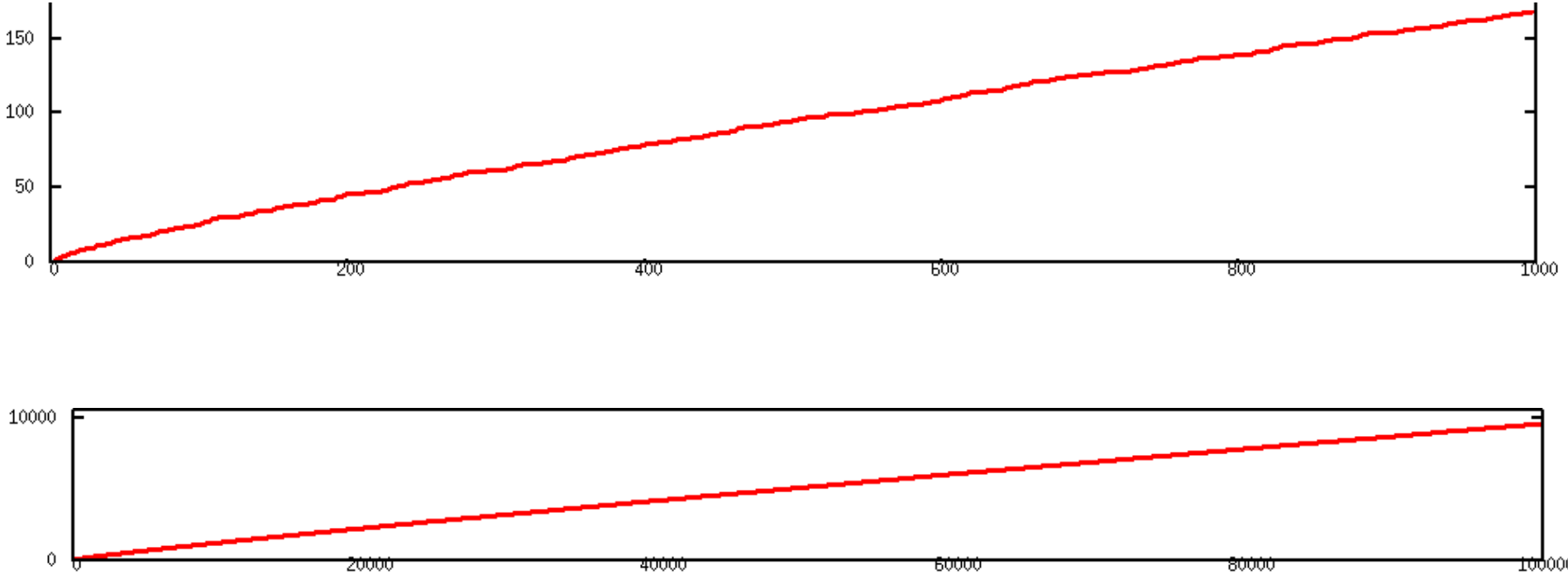
2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 ...

$\pi(n) :=$ Anzahl Primzahlen bis einschließlich n



Verteilung der Primzahlen

Funktion $\pi(n)$



Von weitem sieht $\pi(n)$ immer „glatter“ aus. Welche stetige Funktion approximiert $\pi(n)$ am besten (bis ins Unendliche)?

$\longrightarrow$ `http://de.wikipedia.org/wiki/Primzahlsatz`

Goldbachsche Vermutung

Goldbach (1742, ungelöst): jede gerade Zahl $n \geq 4$ ist darstellbar als Summe zweier Primzahlen

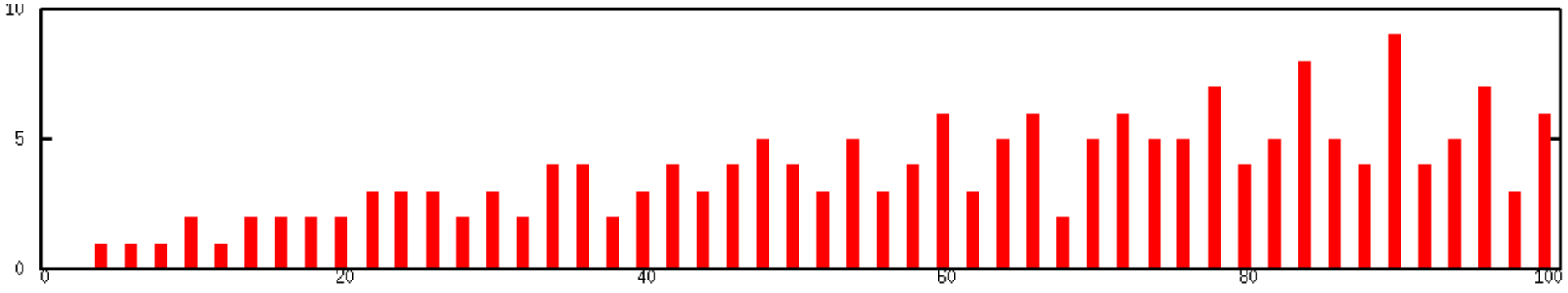
$$4 = 2 + 2$$

$$8 = 3 + 5$$

$$10 = 3 + 7 = 5 + 5$$

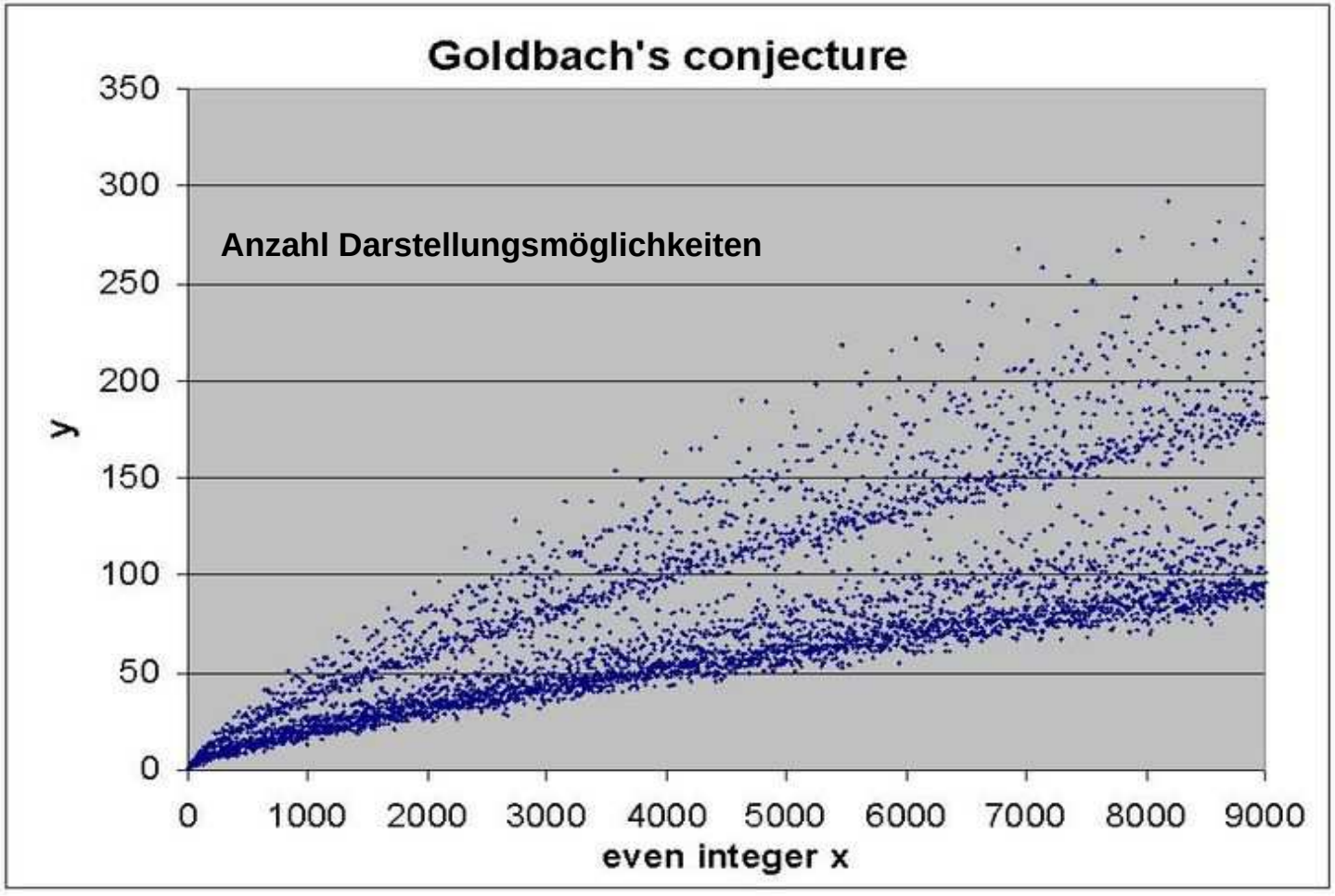
$$24 = 5 + 19 = 7 + 17 = 11 + 13$$

Anzahl Darstellungsmöglichkeiten:



Goldbachsche Vermutung

Goldbach (1742, ungelöst): jede gerade Zahl $n \geq 4$ ist darstellbar als Summe zweier Primzahlen



Primzahltabelle

$\rightarrow$ Es gibt viele Zwecke, für die eine Primzahltabelle nötig ist.

Ziel:

Rechenverfahren entwickeln, das eine Primzahltabelle für $\boxed{n = 10^9}$ möglichst schnell berechnet!

Übersicht

1. Wozu eine Primzahltabelle?
2. Primzahlen direkt prüfen
3. Sieb des Eratosthenes
4. Ergebnisse und Ausblick

Primzahlen direkt prüfen

ist_prim(n)

Eingabe: natürliche Zahl n

Ausgabe: *ja*, falls n prim ist, sonst *nein*

falls $n = 2$ **dann**
    gebe aus *ja*
falls n durch 2 teilbar ist **dann**
    gebe aus *nein*

für $i := 3 \ldots n - 1$ **führe aus:** $\{$ bis $\lfloor \sqrt{n} \rfloor$ reicht $\}$
    falls n durch i teilbar ist **dann**
        gebe aus *nein*

gebe aus *ja*

Primzahlen direkt prüfen

Idee:

prüfe alle Zahlen $2 \ldots n$, ob sie prim sind

für $k := 2 \ldots n$ **führe aus:**

ist_prim(k)

Laufzeiten (LINUX-PC 3.2 GHz):

n	10^3	10^4	10^5	10^6	10^7	10^8
Zeit [s]	0.00	0.00	0.02	0.5	11.9	312.9

$\rightarrow$ **Nachteil:** rechnet jedesmal neu, nutzt keine Vorergebnisse

Übersicht

1. Wozu eine Primzahltabelle?
2. Primzahlen direkt prüfen
3. Sieb des Eratosthenes
4. Ergebnisse und Ausblick

Wer war Eratosthenes?

Eratosthenes von Kyrene
ca. 276–194 v. Chr.

Mathematiker
Astronom
Geograph

Beschäftigte sich mit Dingen wie

- Berechnung des Erdumfangs
- Verdopplung des Würfels
- Dreiteilung eines Winkels
- Primzahlen

`http://www-groups.dcs.st-andrews.ac.uk/~history/Mathematicians/Eratosthenes.html`

Primzahlen als Tabelle generieren

Grundidee:

erzeuge alle zusammengesetzten Zahlen $\leq n$
streiche sie ($\rightarrow$ *aussieben*)

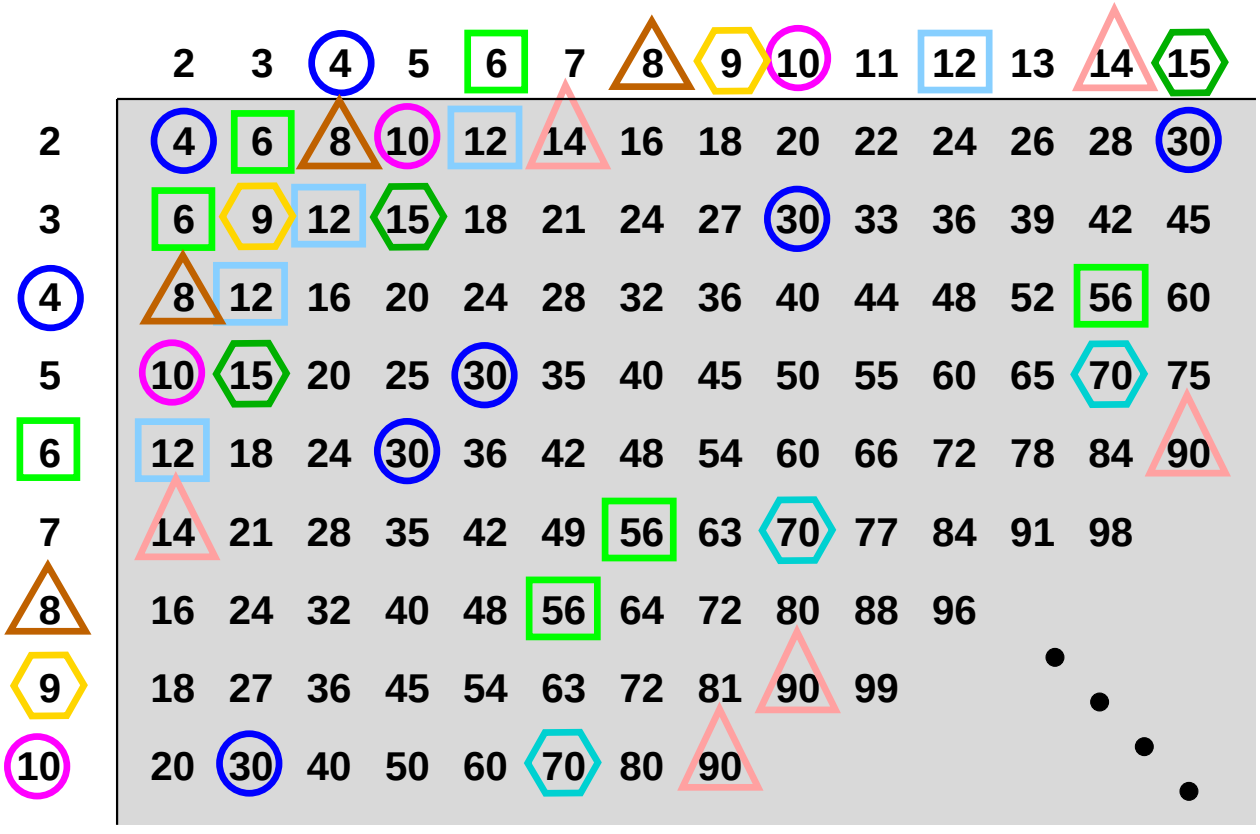
schreibe die Zahlen $2 \ldots n$ in eine Liste
für $i := 2 \ldots n$ **führe aus:**
  für $k := 2 \ldots n$ **führe aus:**
    streiche die Zahl $i \cdot k$ aus der Liste

Laufzeiten (LINUX-PC 3.2 GHz):

n	10^3	10^4	10^5	10^6
Zeit [s]	0.00	0.2	19.4	1943.4

Analyse des Verfahrens

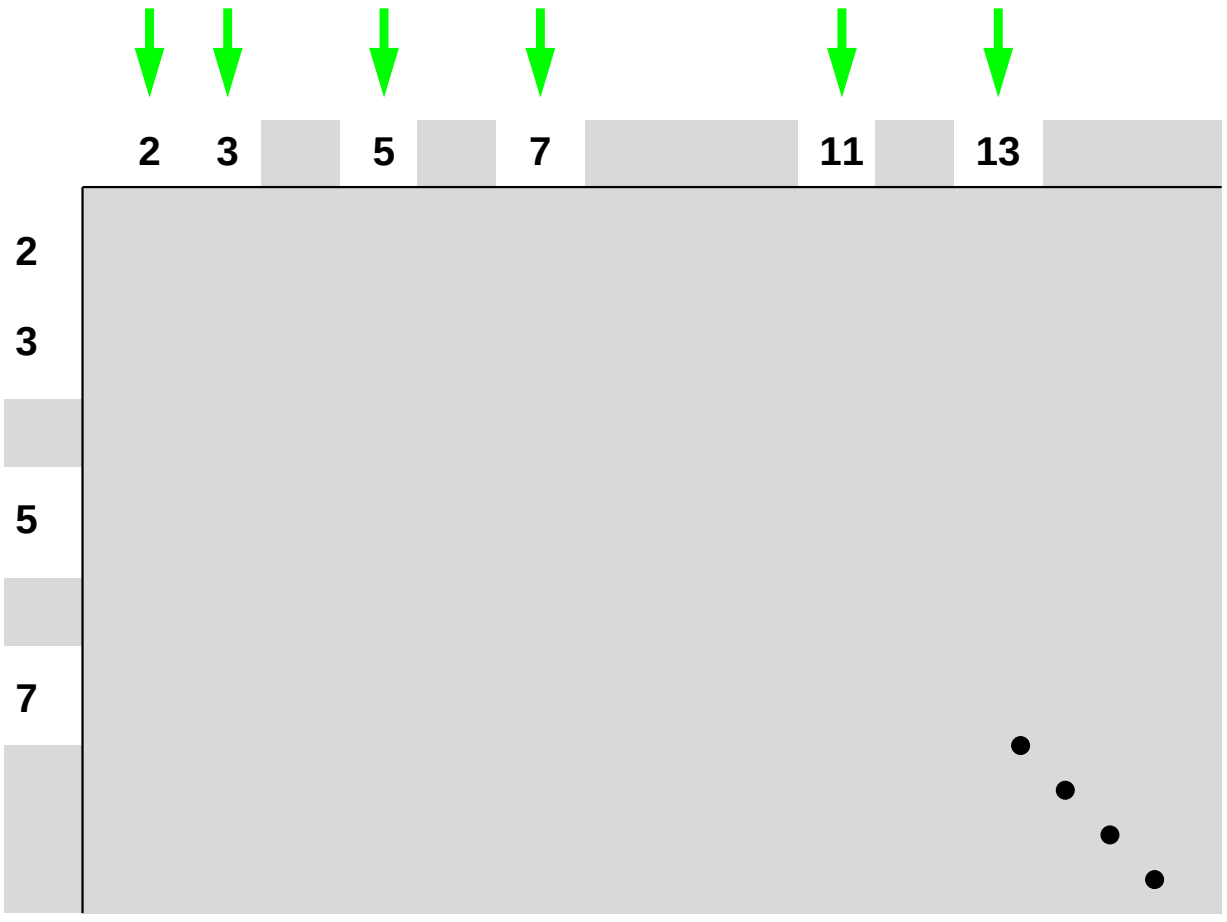
Feld der erzeugten Zahlen $i \cdot k$ (grau):



	2	3	4	5	6	7	8	9	10	11	12	13	14	15
2	4	6	8	10	12	14	16	18	20	22	24	26	28	30
3	6	9	12	15	18	21	24	27	30	33	36	39	42	45
4	8	12	16	20	24	28	32	36	40	44	48	52	56	60
5	10	15	20	25	30	35	40	45	50	55	60	65	70	75
6	12	18	24	30	36	42	48	54	60	66	72	78	84	90
7	14	21	28	35	42	49	56	63	70	77	84	91	98	
8	16	24	32	40	48	56	64	72	80	88	96			
9	18	27	36	45	54	63	72	81	90	99				
10	20	30	40	50	60	70	80	90						

Analyse des Verfahrens

nach Ende weggestrichene Zahlen (grau):



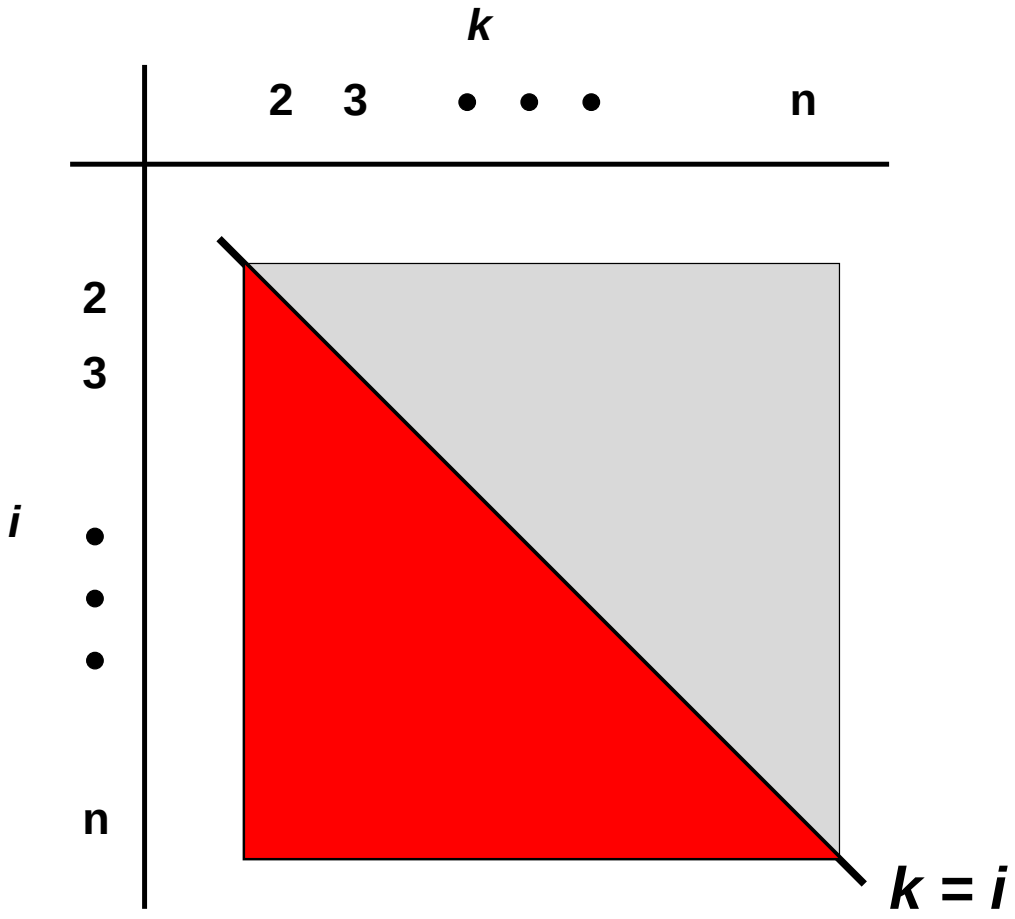
alle Zahlen k
mit der einzigen
Darstellung
$k \cdot 1 = 1 \cdot k$
bleiben übrig:

Primzahlen

1. Verbesserung

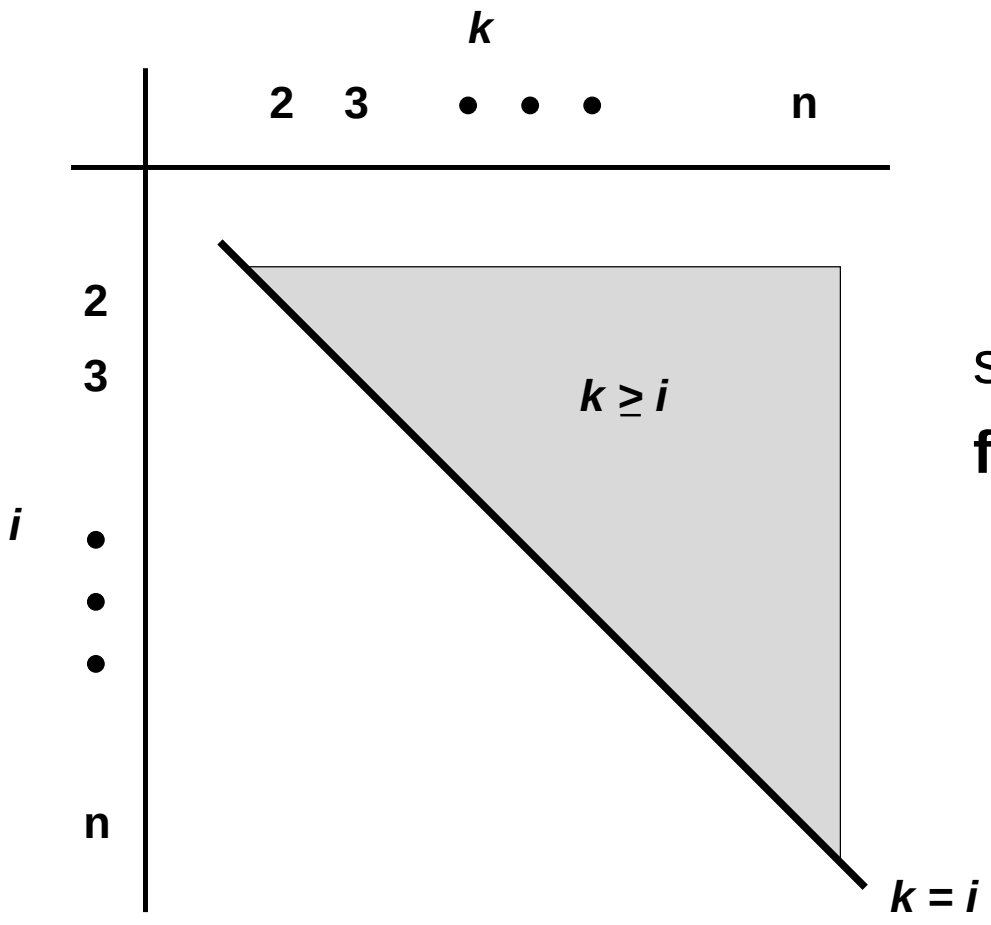
Idee:

Symmetrie $i \cdot k = k \cdot i$ ausnutzen



$\Rightarrow \quad \boxed{k \geq i}$

1. Verbesserung



schreibe $2 \ldots n$ in Liste

für $i := 2 \ldots n$ **führe aus:**

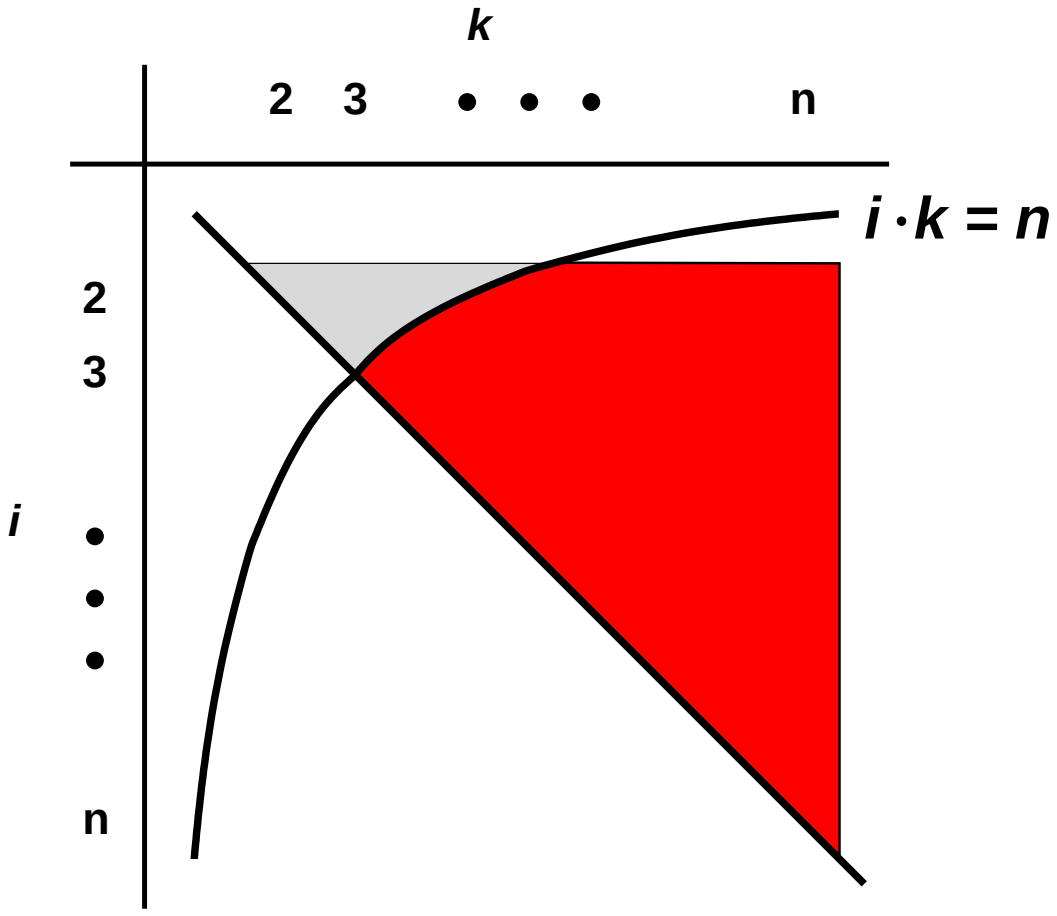
  für $k :=$ i $\ldots n$ **führe aus:**

    streiche $i \cdot k$ aus Liste

2. Verbesserung

Idee:

Bedingung $i \cdot k \leq n$ benutzen

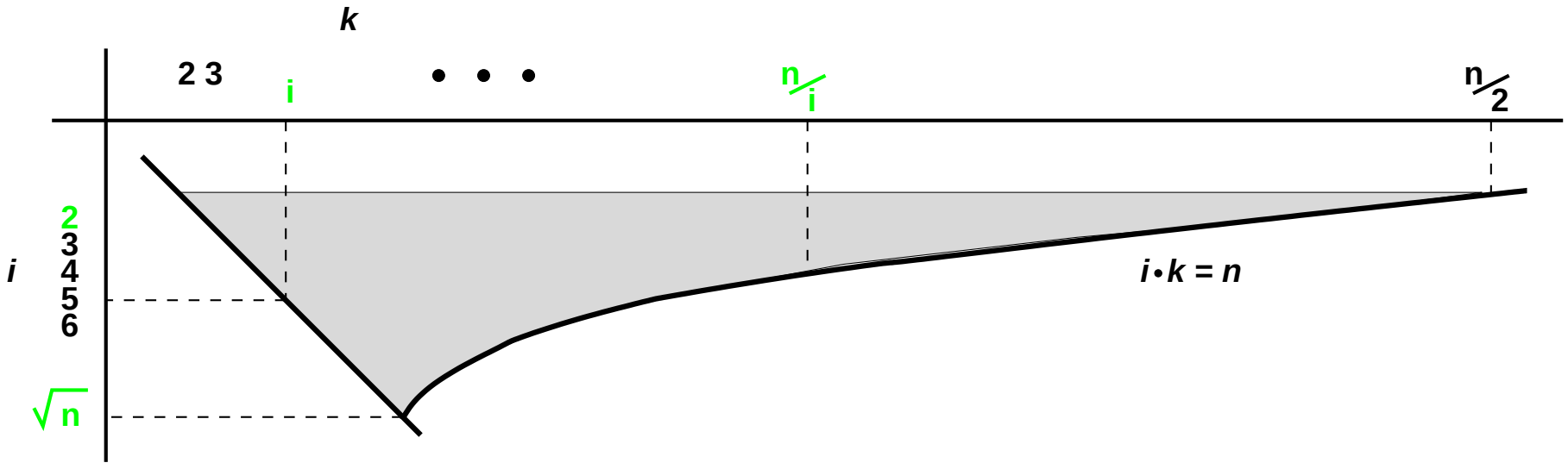


$$\Rightarrow \quad \boxed{k \leq \frac{n}{i}}$$

und mit $k \geq i$

$$\boxed{i \leq \sqrt{n}}$$

2. Verbesserung



schreibe $2 \ldots n$ in Liste

für $i := 2 \ldots \boxed{\lfloor \sqrt{n} \rfloor}$ **führe aus:**

für $k := i \ldots \boxed{\lfloor n/i \rfloor}$ **führe aus:**

streiche $i \cdot k$ aus Liste

Auswirkungen der Verbesserungen

Laufzeiten (LINUX-PC 3.2 GHz) in Sekunden:

n	10^4	10^5	10^6	10^7	10^8	10^9
ist_prim	0.00	0.02	0.5	11.9	312.9	
Grundversion	0.19	19.4	1943.4			
1. *Verbesserung*	0.10	9.8	972.4			
2. *Verbesserung*	0.00	0.01	0.01	2.3	32.7	452.9

3. Verbesserung

Idee:

im Produkt $i \cdot k$ braucht i nur Primzahl zu sein

Beweis: Ist i nicht prim, hat es einen Primteiler $p < i$ und es gibt eine Darstellung $i = p \cdot q$. Dann ist

$$i \cdot k \;=\; (p \cdot q) \cdot k \;=\; p \cdot (q \cdot k) \;=:\; i' \cdot k'$$

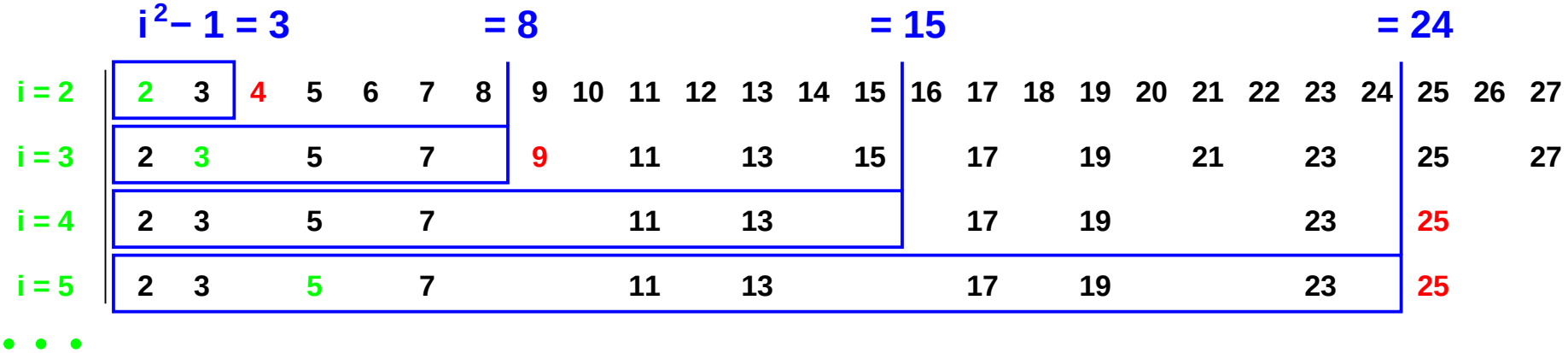
mit $i' = p < i$. Also wurde $i \cdot k$ bereits vorher als Vielfaches von i' gestrichen $\Rightarrow$ nichts mehr zu tun. □

Verbesserung: ***geht das auch?***

schreibe $2 \ldots n$ in Liste
für $i := 2 \ldots \lfloor \sqrt{n} \rfloor$ prim **führe aus:** ← in Liste
  für $k := i \ldots \lfloor n/i \rfloor$ **führe aus:**
    streiche $i \cdot k$ aus Liste

Analyse der 3. Verbesserung

Satz: Vor der i. Iteration stimmt die Tabelle (mindest.) bis $i^2 - 1$.



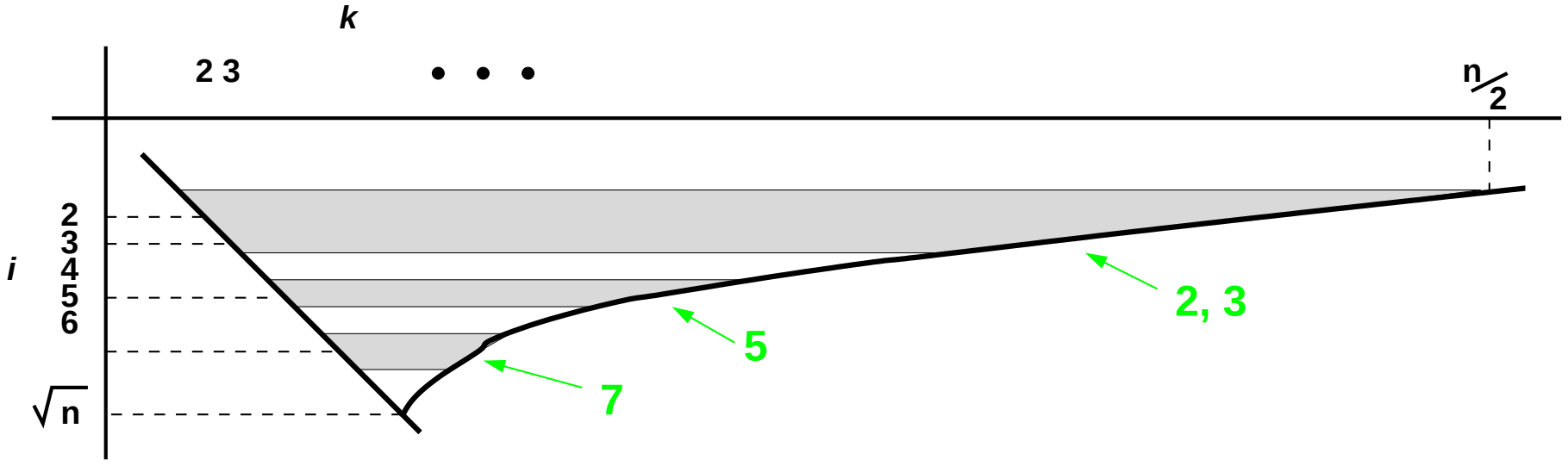
D.h. die Tabelle wird *quadratisch* schnell fertig

Beweis: In der i. Iteration werden nur Zahlen $\geq i^2$ gestrichen. Alle Zahlen $2 \ldots i^2 - 1$ bleiben bis zum Schluss unverändert

$\Rightarrow$ sie müssen hier schon korrekt sein. □

$\Rightarrow$ kann für die speziellen i einfach in Liste nachschauen!

3. Verbesserung



schreibe $2 \ldots n$ in Liste

für $i := 2 \ldots \lfloor \sqrt{n} \rfloor$ in Liste **führe aus:**

  für $k := i \ldots \lfloor n/i \rfloor$ **führe aus:**

    streiche $i \cdot k$ aus Liste

Sieb des Eratosthenes

4. Verbesserung

Idee:

während der i. Iteration brauche nur solche k, die noch in der Liste stehen

Beweis: Steht k nicht in der Liste, so ist es in einer Voriteration $i' < i$ als Vielfaches $k = i' \cdot k'$ gestrichen worden. Daher wurde $i \cdot k = i' \cdot (i \cdot k')$ ebenfalls als Vielfaches von i' gestrichen $\Rightarrow$ nichts mehr zu tun. □

Verbesserung?

schreibe $2 \ldots n$ in Liste

für $i := 2 \ldots \lfloor \sqrt{n} \rfloor$ in Liste **führe aus:**

  für $k := i \ldots \lfloor n/i \rfloor$ in Liste **führe aus:**

    streiche $i \cdot k$ aus Liste

es wird aber folgende Tabelle erzeugt:

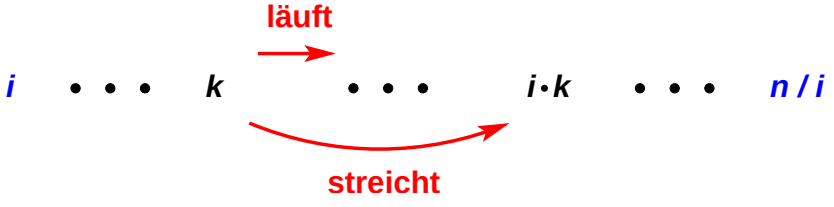
2 3 5 7 8 11 12 13
17 19 20 23 27 28
29 31 32 37 41 ...

Analyse der 4. Verbesserung

Iteration $i = 2$ in Zeitlupe:

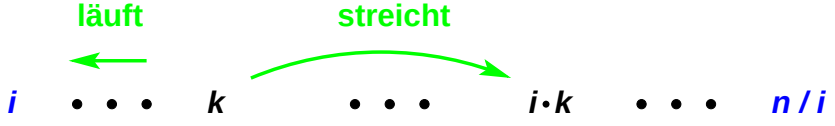
	2	3	4	5	6	7	8	9	10	11	...
$k = 2$:	2	3	**4**	5	6	7	8	9	10	11	...
$k = 3$:	2	3	–	5	**6**	7	8	9	10	11	...
$k = 5$:	2	3	–	5	–	7	**8**	9	**10**	11	...

Problem: *k* läuft nach rechts *und* streicht nach rechts → Verfahren „behindert“ sich selbst.

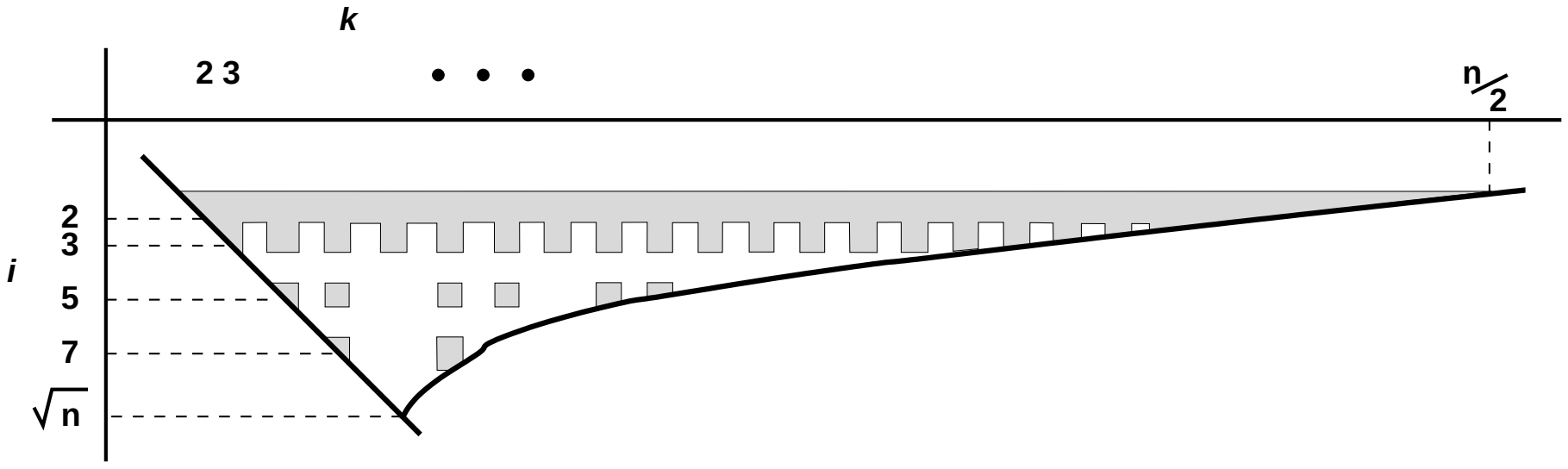


Lösung: muss alle *k* nehmen, die *zu Anfang* der *i*. Iteration in der Liste stehen.

Umsetzung: *k* streicht immer nach rechts → muss also nach *links* laufen.



4. Verbesserung



schreibe $2 \ldots n$ in Liste

für $i := 2 \ldots \lfloor \sqrt{n} \rfloor$ in Liste **führe aus:**

für $k := \lfloor n/i \rfloor \ldots i$ in Liste absteigend **führe aus:**

streiche $i \cdot k$ aus Liste

Auswirkungen der Verbesserungen

Laufzeiten (LINUX-PC 3.2 GHz) in Sekunden:

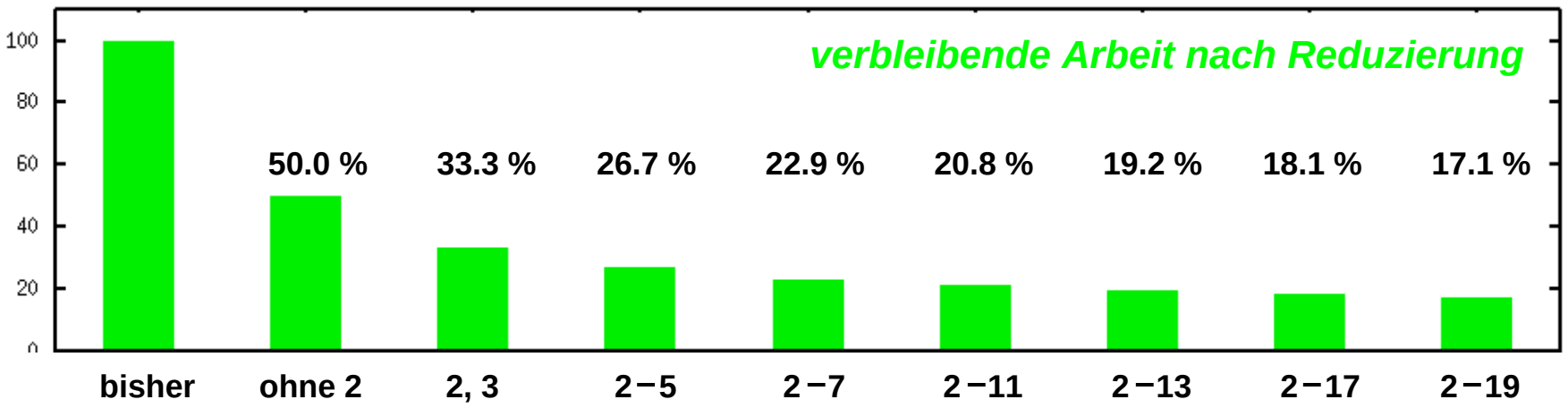
n	10^6	10^7	10^8	10^9
ist_prim	0.5	11.9	312.9	
2. *Verbesserung*	0.01	2.3	32.7	452.9
3. *Verbesserung*	0.02	0.43	5.4	66.5
4. *Verbesserung*	0.01	0.15	1.6	**17.6**

5. Verbesserung

Idee:

starte mit *reduzierter* Liste: ohne die Vielfachen von 2, 3, 5, 7, . . .

bisher:	2	3	4	5	6	7	8	9	10	11	12	13
ohne 2:	2	3	5	7	9	11	13	15	17	19	21	23
2, 3:	2	3	5	7	11	13	17	19	23	25	29	31
2, 3, 5:	2	3	5	7	11	13	17	19	23	29	31	37



Übersicht

1. Wozu eine Primzahltabelle?
2. Primzahlen direkt prüfen
3. Sieb des Eratosthenes
4. Ergebnisse und Ausblick

Ergebnisse

$n = 10^9$

reduz. Liste	Rechenzeit	Tabellengröße
keine	17.6 s	119.2 MByte
ohne 2	33.0 s	59.6 MByte
2, 3	22.6 s	39.7 MByte
2–5	17.8 s	31.8 MByte
2–7	14.7 s	27.3 MByte
2–11	13.3 s	24.8 MByte
2–13	**12.6 s**	22.9 MByte
2–17	24.0 s	21.6 MByte
2–19	25.9 s	20.4 MByte

Anmerkungen: Die Tabelle wird hier als Array von Bits gespeichert. Bei Reduzierung kommt Rechenaufwand durch Indexumrechnungen dazu.

so gut wie möglich?

Idee:

finde ein Mindestmaß für den Arbeitsaufwand
vergleiche Verfahren damit

Was muß das Verfahren am Ende getan haben?

Alle Nichtprimzahlen müssen aus der Tabelle gestrichen sein.

$\rightarrow$ verwende *Anzahl Nichtprimzahlen* als Maßstab

$n = 10^9$	Produkte $i \cdot k$	*Nichtprimzahlen*	*Verhältnis*
Grundversion	10^{18}	$9.49 \cdot 10^8$	$1.1 \cdot 10^9$
1. Verbesserung	$5 \cdot 10^{17}$		$5.3 \cdot 10^8$
2. Verbesserung	$9.44 \cdot 10^9$		9.9
3. Verbesserung	$2.55 \cdot 10^9$		2.7
4. Verbesserung	$9.49 \cdot 10^8$		1.0
5. Verb.: ohne 2	$4.49 \cdot 10^8$	$4.49 \cdot 10^8$	1.0
5. Verb.: 2, 3	$2.82 \cdot 10^8$	$2.82 \cdot 10^8$	1.0
5. Verb.: 2–13	$1.41 \cdot 10^8$	$1.41 \cdot 10^8$	1.0

Ausblick

Was lehrt uns dieses Beispiel?

1. einfache Rechenverfahren sind nicht automatisch *effizient*,
2. zu ihrer Beschleunigung muß man sie *gut verstehen*,
3. es sind oft *viele Verbesserungen* möglich,
4. ihre *Bewertung* erfordert gutes Gefühl für Aufwand
5. *mathematische Ideen* sind oft weitreichender als computertechnische,
6. Verfahrensoptimierung hat etwas *Sportliches* $\rightarrow$ **Spaß!**

Vielen Dank für eure Aufmerksamkeit!