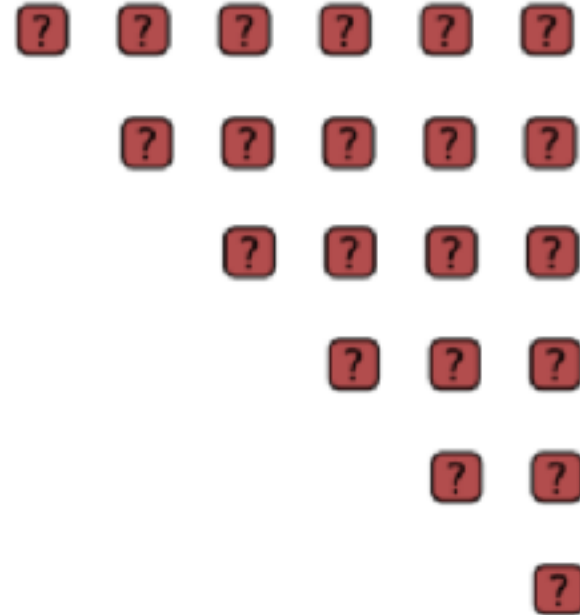
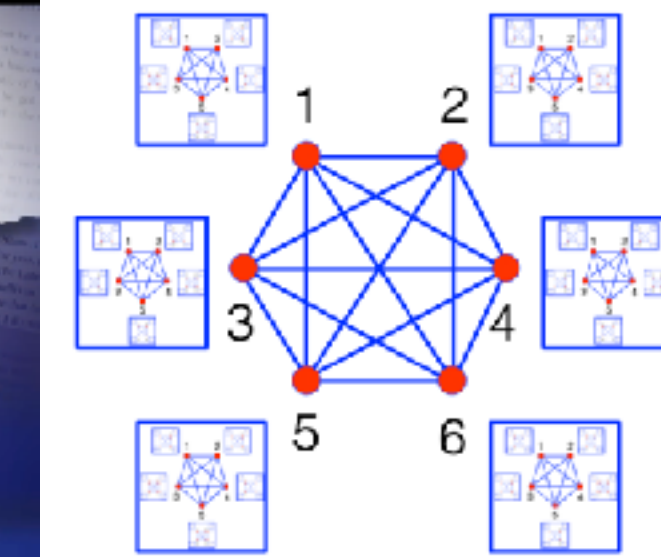
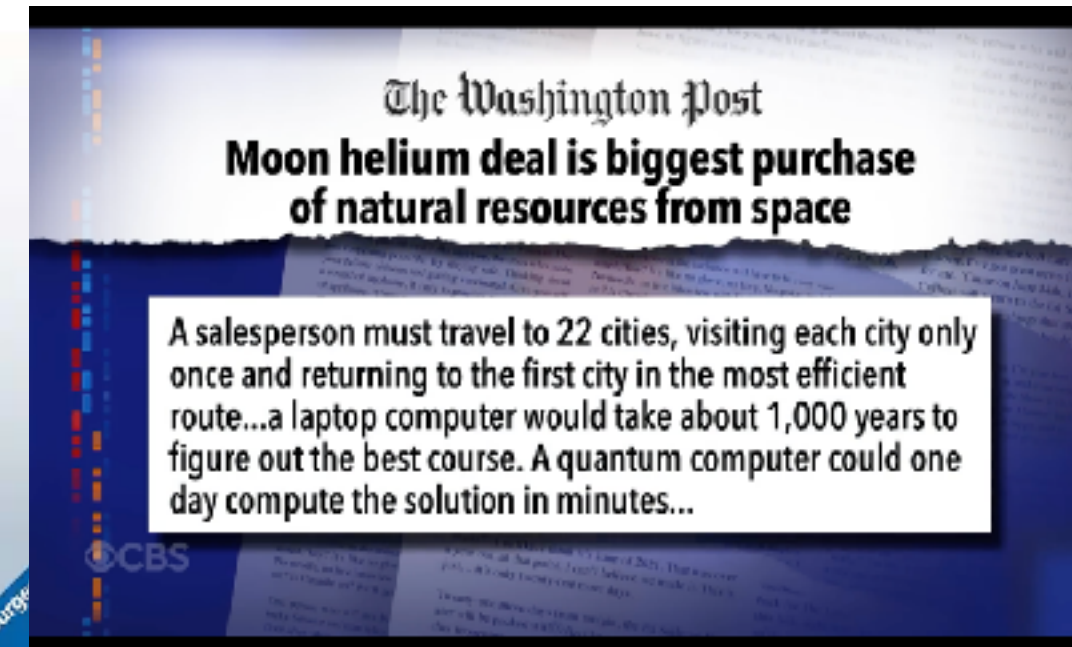
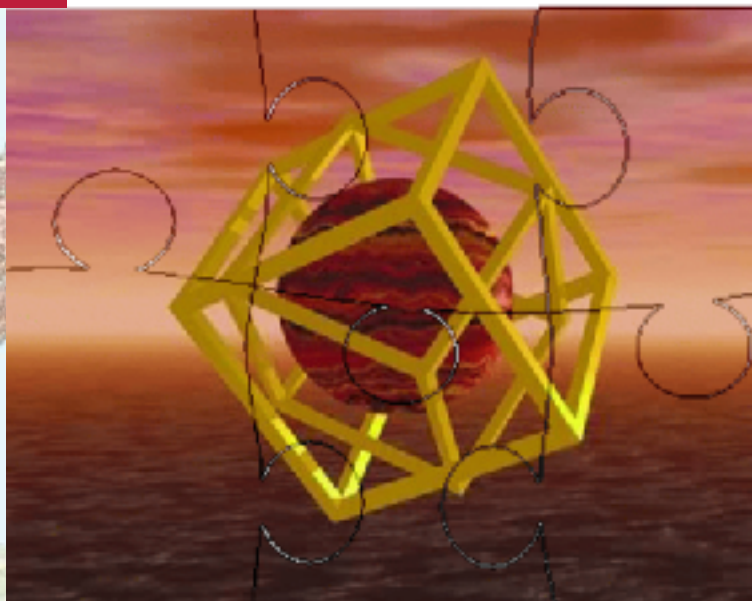
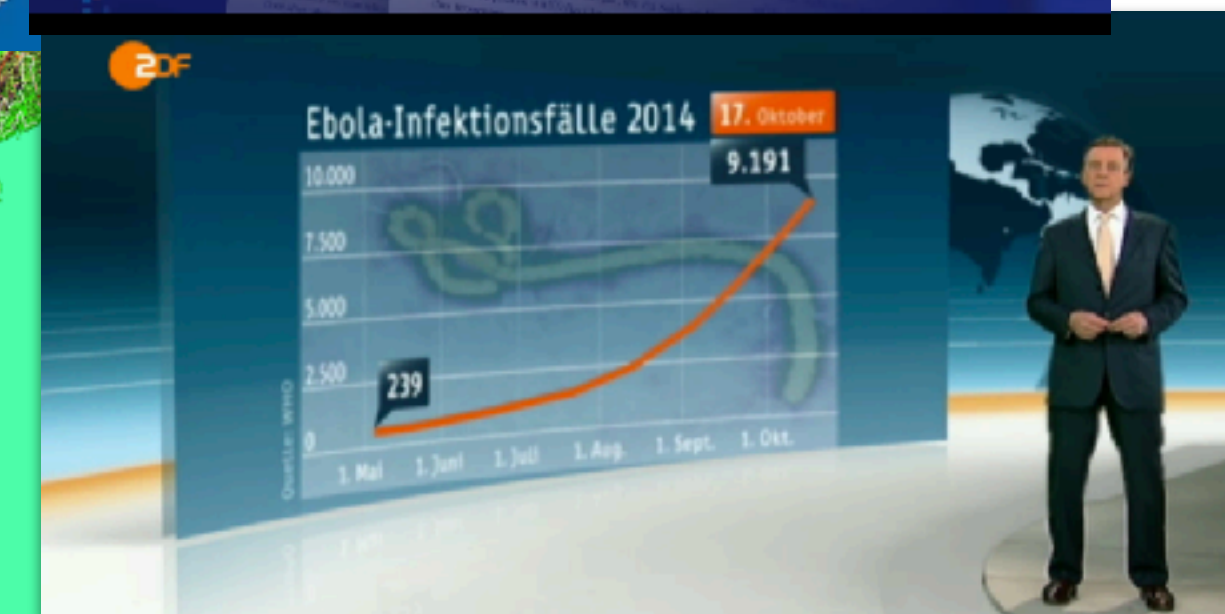
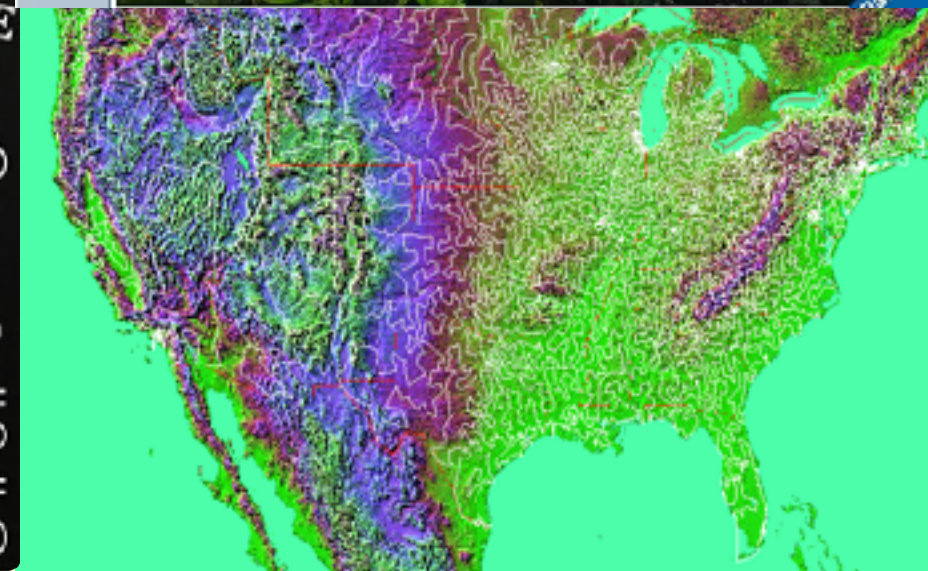




Technische
Universität
Braunschweig



WE OFFER THREE • KINDS OF SERVICE
GOOD • FAST • CHEAP
YOU CAN PICK ANY TWO
★★★★★
GOOD & CHEAP WON'T BE FAST
GOOD & FAST WON'T BE CHEAP
FAST & CHEAP WON'T BE GOOD



Geht's nicht noch etwas schneller?

Eine Rundreise durch gelöste und ungelöste Probleme des algorithmischen Alltags

Sándor P. Fekete

Department of Computer Science

TU Braunschweig

Die beiden Hauptdarsteller

Algorithmen

Datenstrukturen

I get the job done.
What the hell do
you want?

I don't make
things difficult.
That's the way
they get, all by
themselves.

Then don't.



Can you make it
without killing
yourself?

I don't want to
work with you.

Ain't got no choice.

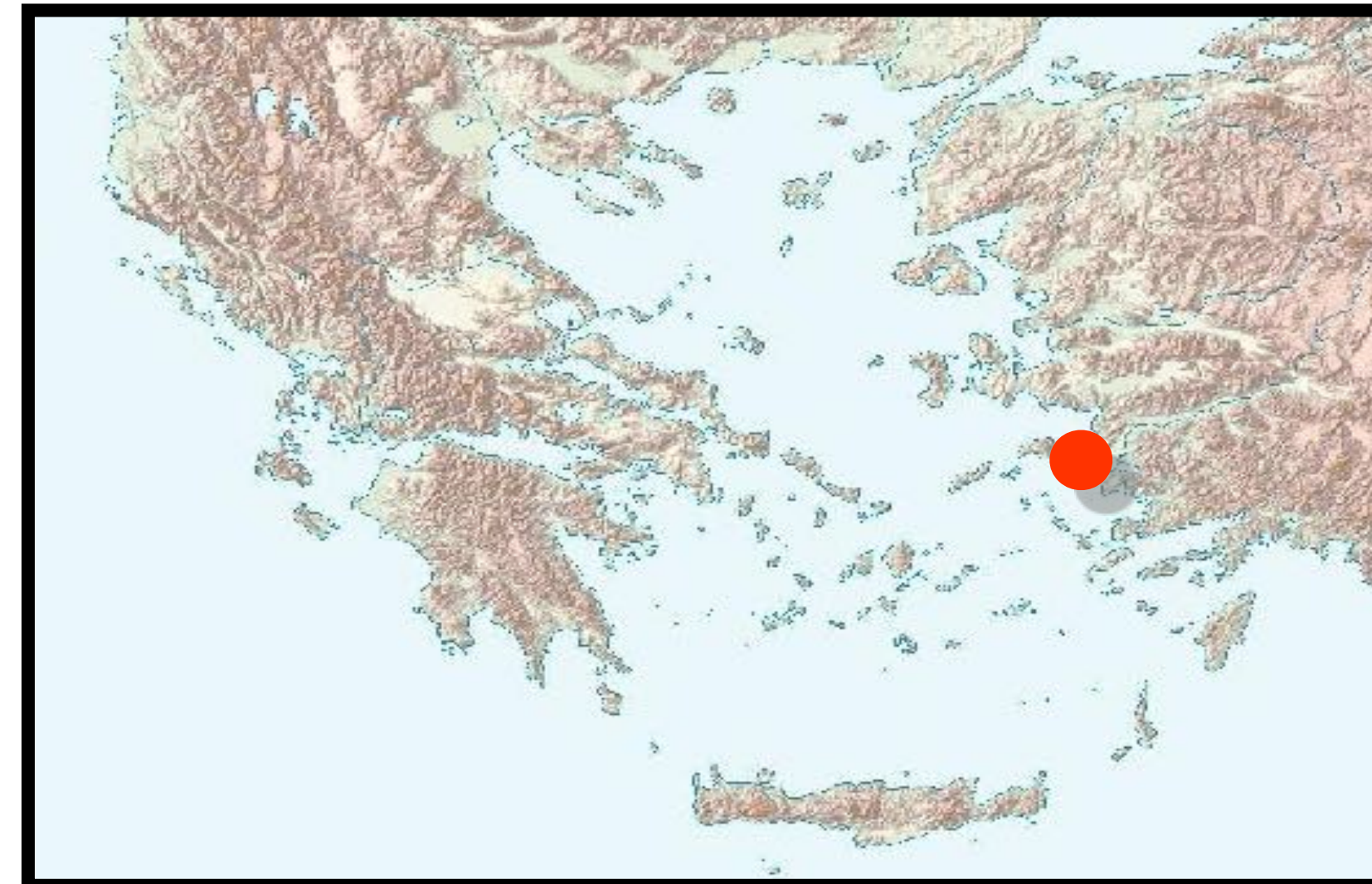
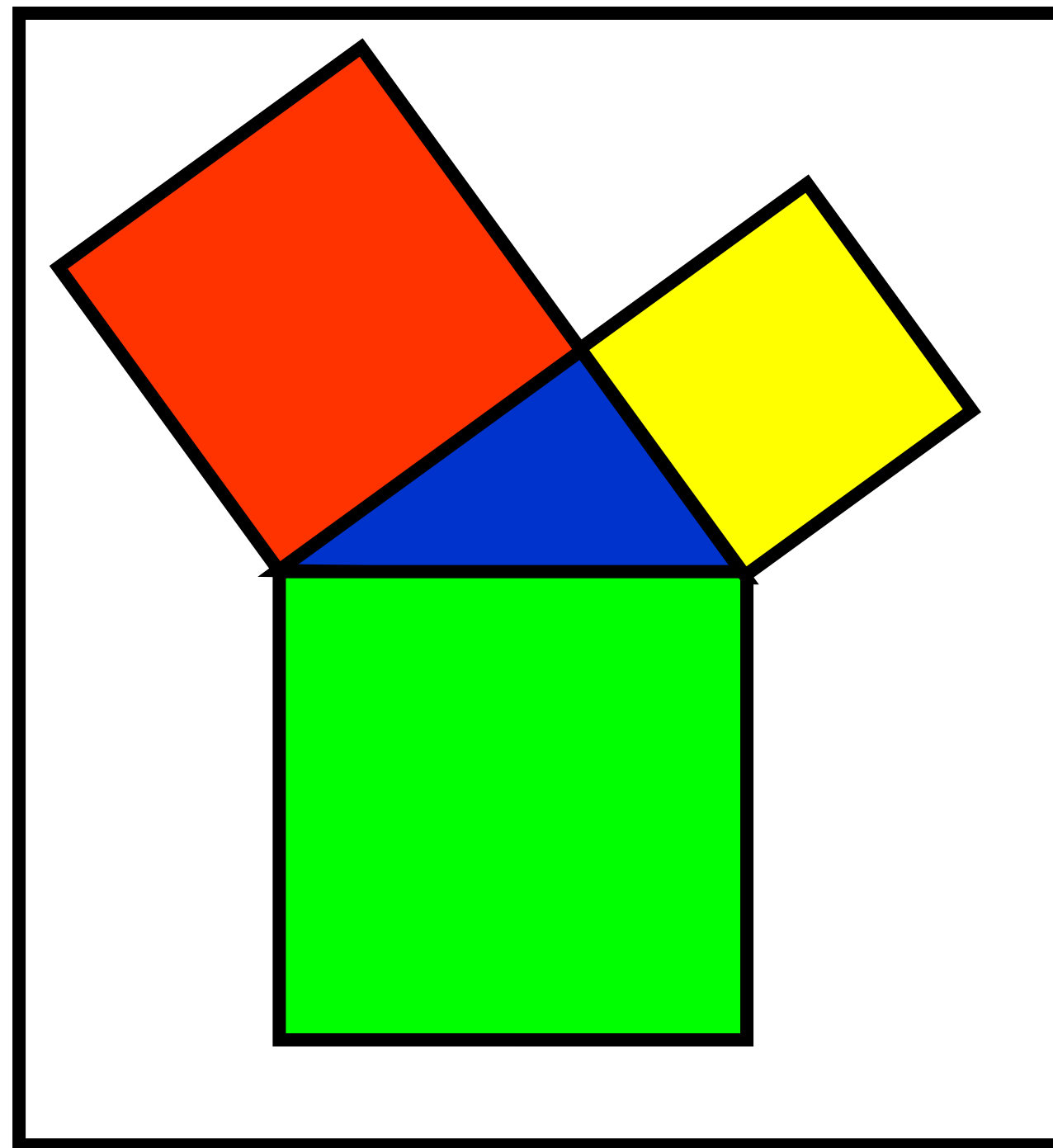
I'm getting too old
for this...

1 Die Laufzeit der Geschichte

Die bekannteste Formel aller Zeiten

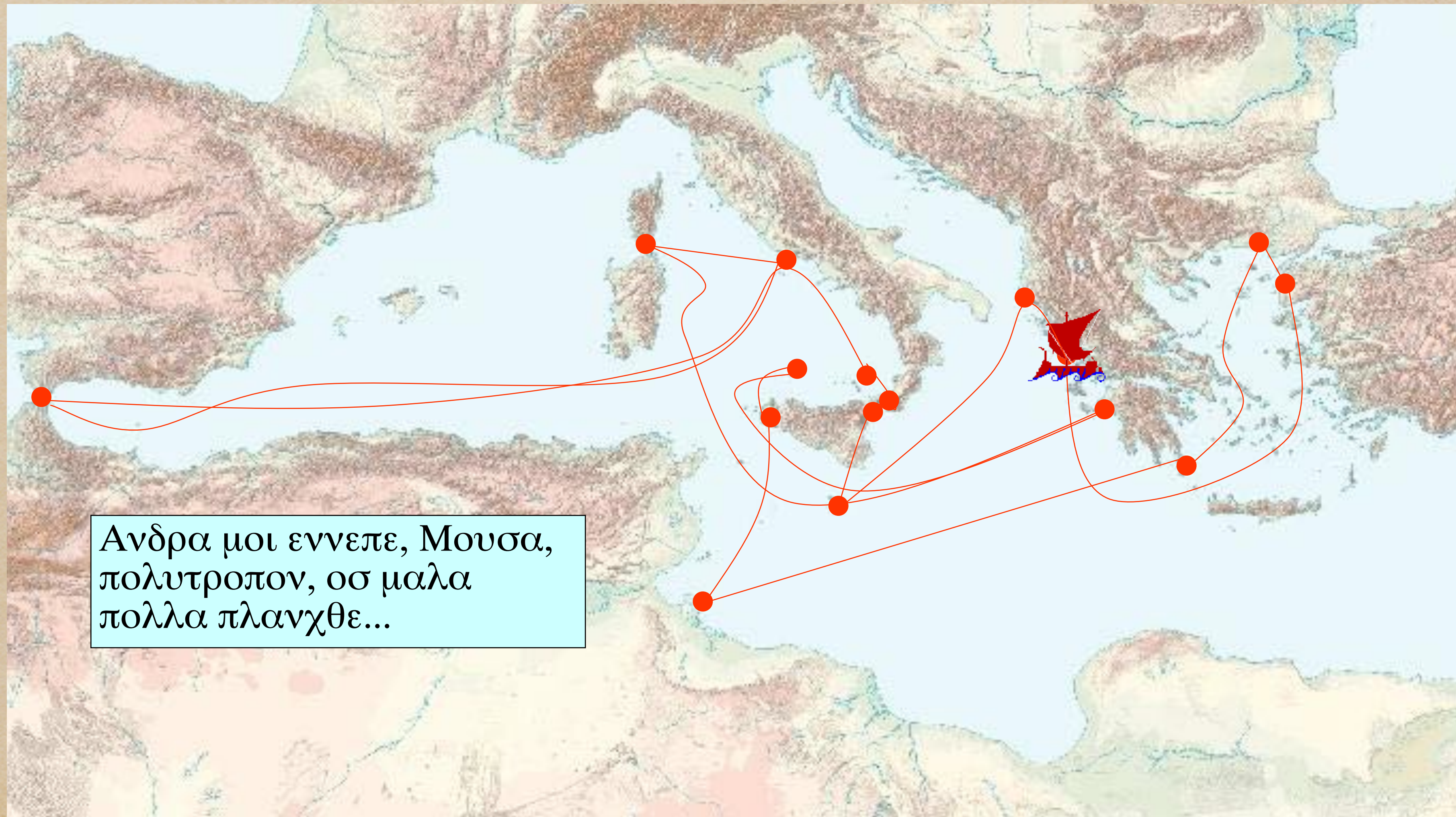
mathematische

$$a^2 + b^2 = c^2$$

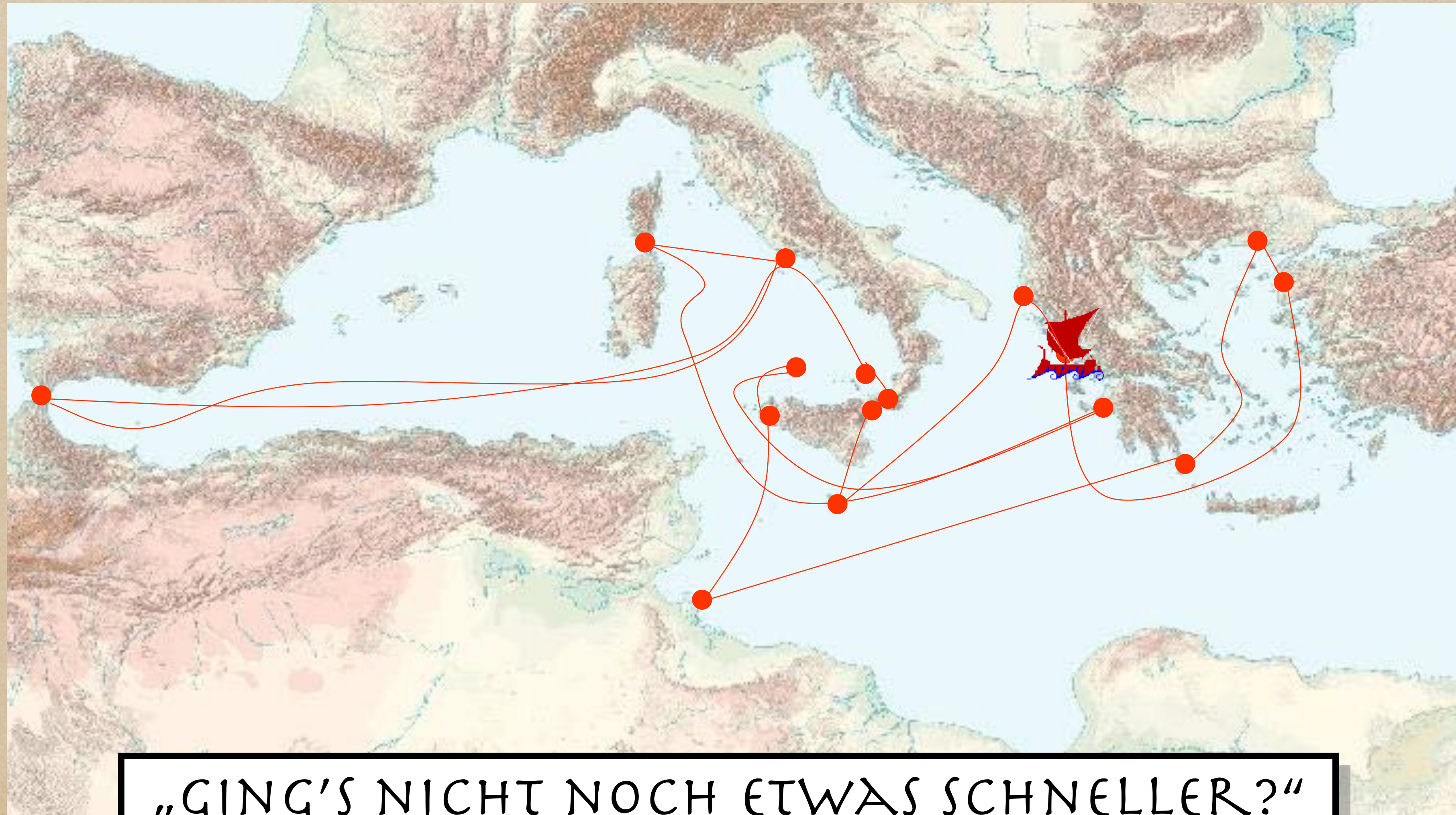


Pythagoras (ca. 500 v.Chr.)

20 JAHRE SPÄTER!



20 JAHRE SPÄTER!



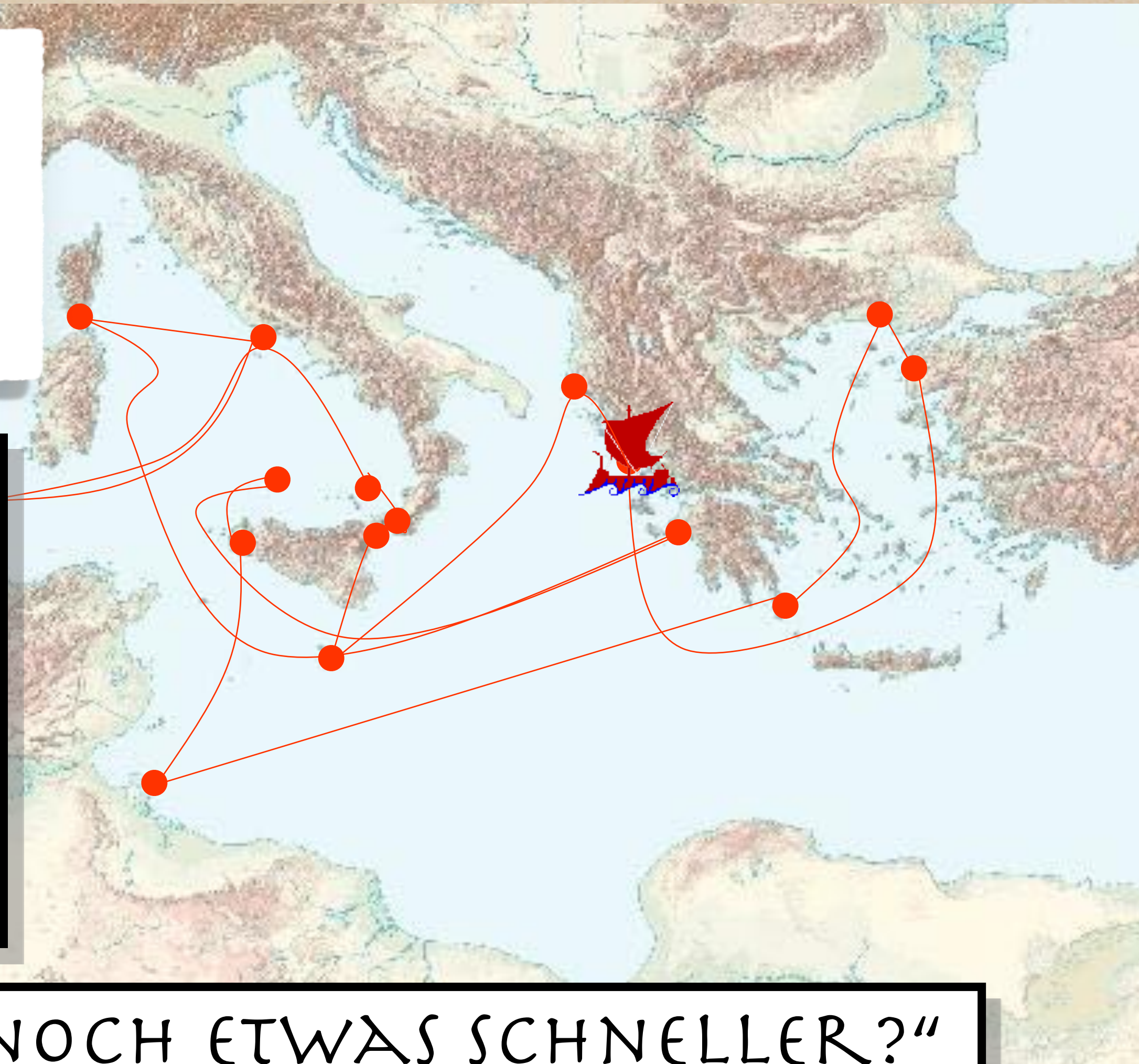
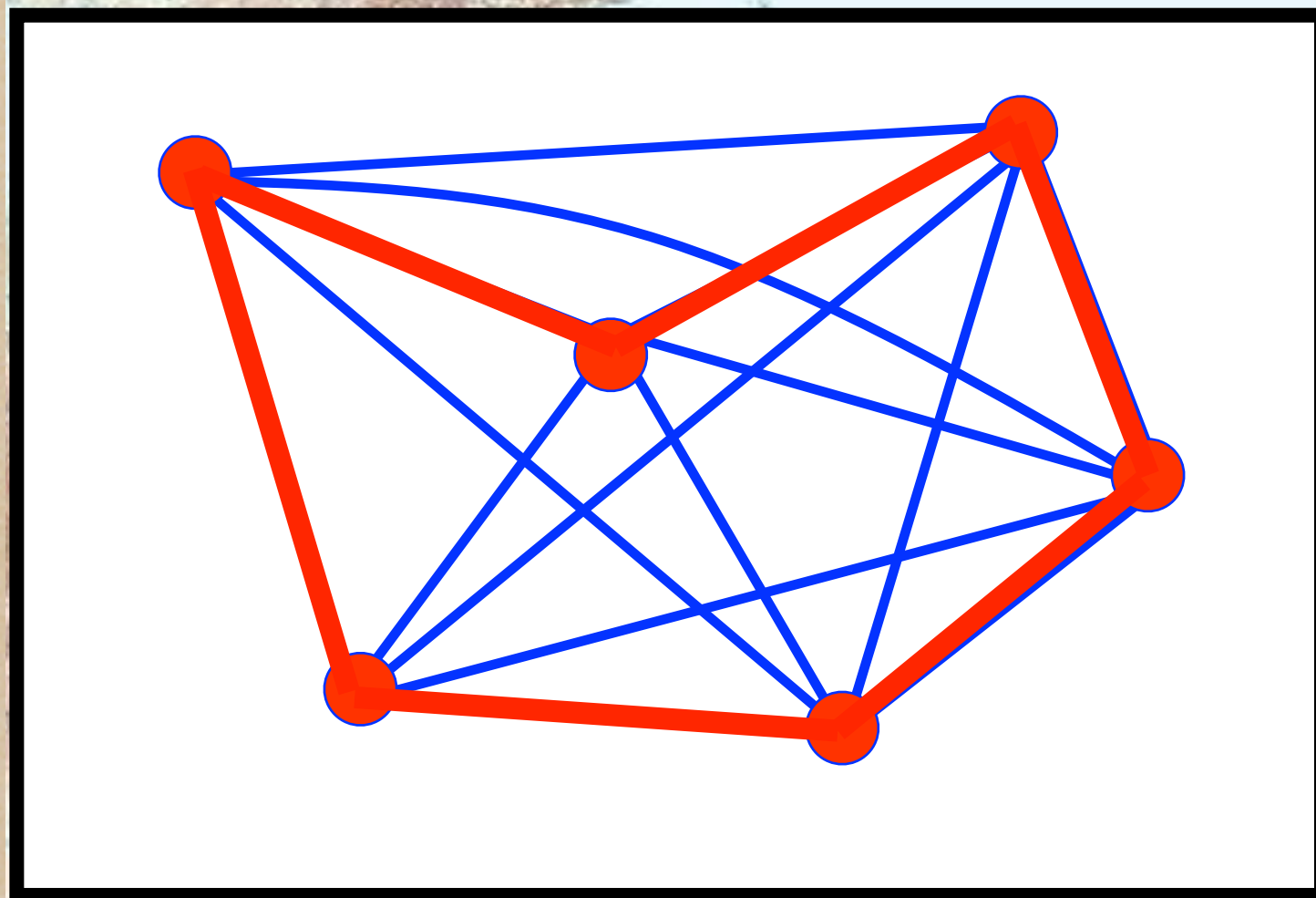
„GING'S NICHT NOCH ETWAS SCHNELLER?“

20 JAHRE SPÄTER!

Das Traveling Salesman Problem

Input: Ein gewichteter Graph $G=(V,E)$

Output: Eine kürzeste Rundreise $\mathcal{T}$
die alle Knoten in V besucht



„GING'S NICHT NOCH ETWAS SCHNELLER?“



Geschäftsmodell

The Washington Post Moon helium deal is biggest purchase of natural resources from space

A salesperson must travel to 22 cities, visiting each city only once and returning to the first city in the most efficient route...a laptop computer would take about 1,000 years to figure out the best course. A quantum computer could one day compute the solution in minutes...

The Washington Post

MONDAY, FEBRUARY 5, 2018

"It would take a laptop computer 1,000 years to compute the most efficient route between 22 cities, for example. A quantum computer could do this within minutes, possibly seconds."

February 5, 2018

- (1) 300 Mio. einsammeln
- (2) Zum Mond fliegen
- (3) Helium-3 abbauen
- (4) Zur Erde bringen
- (5) Quantencomputer bauen
- (6) Beste Tour berechnen

Geschäftsmodell

Bill Cook



The Washington Post

Mostly sunny 52/32 • Tomorrow: Cloudy 47/33 [B](#) MONDAY, FEBRUARY 5, 2018 • B2

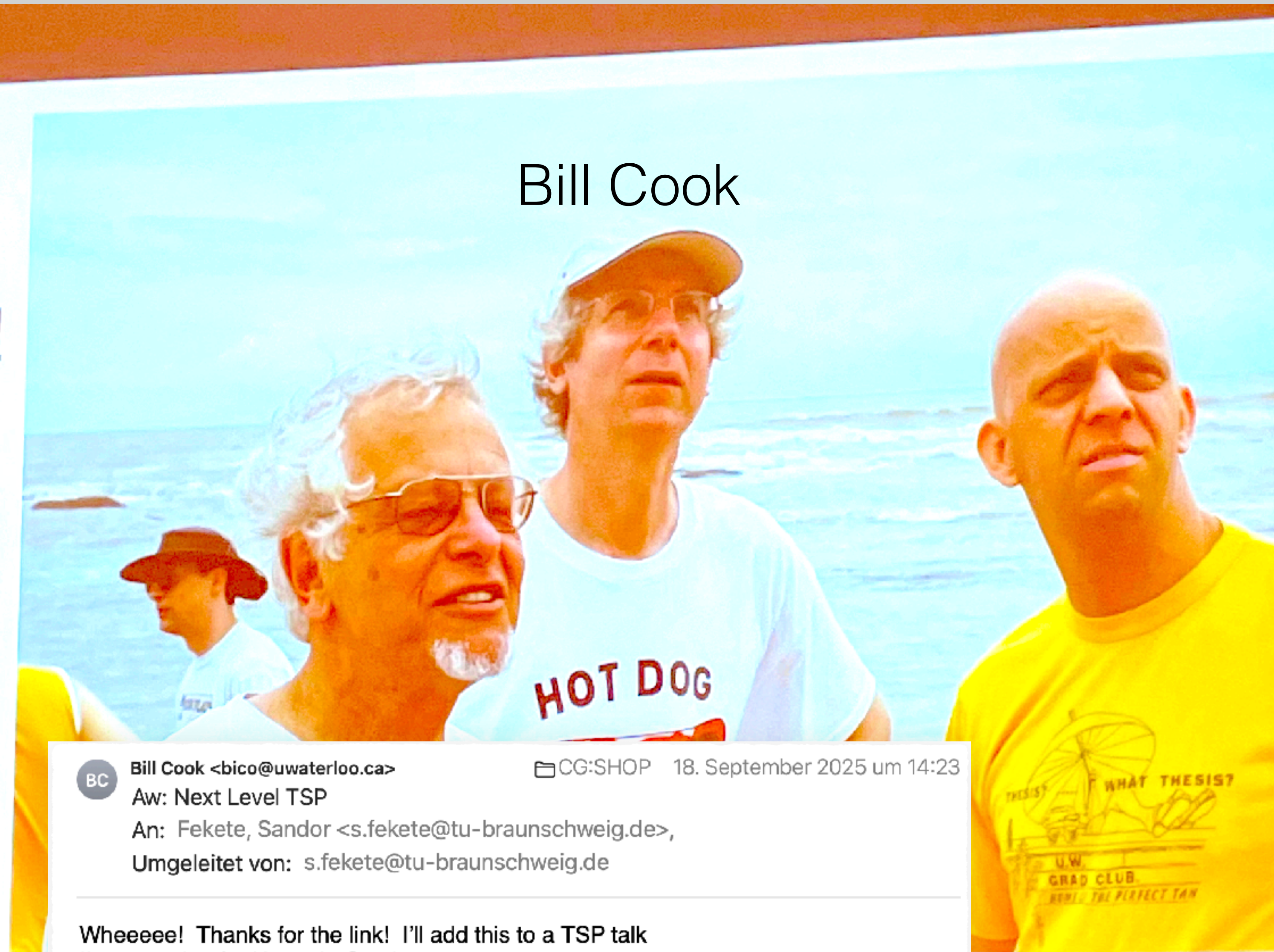
“It would take a laptop computer 1,000 years to compute the most efficient route between 22 cities, for example. A quantum computer could do this within minutes, possibly seconds.”

February 5, 2018

- (1) 300 Mio. einsammeln
- (2) Zum Mond fliegen
- (3) Helium-3 abbauen
- (4) Zur Erde bringen
- (5) Quantencomputer bauen
- (6) Beste Tour berechnen

Geschäftsmodell

Bill Cook



BC

Bill Cook <bico@uwaterloo.ca>

CG:SHOP 18. September 2025 um 14:23

Aw: Next Level TSP

An: Fekete, Sandor <s.fekete@tu-braunschweig.de>,

Umgeleitet von: s.fekete@tu-braunschweig.de

Wheeeee! Thanks for the link! I'll add this to a TSP talk
I'm preparing for Korea in October.

Bico

Google Research

The Washington Post

Mostly sunny 52/37 • Tomorrow: Cloudy 47/33 [B](#) MONDAY, FEBRUARY 5, 2018 • 82

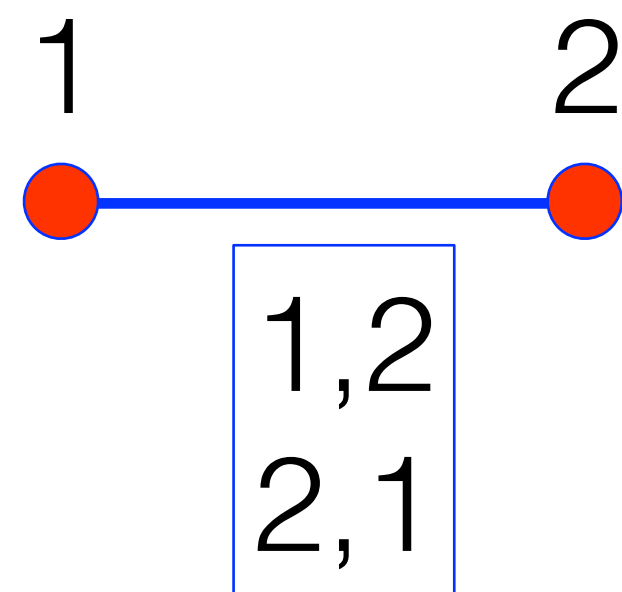
“It would take a laptop computer 1,000 years to compute the most efficient route between 22 cities, for example. A quantum computer could do this within minutes, possibly seconds.”

February 5, 2018

- (1) 300 Mio. einsammeln
- (2) Zum Mond fliegen
- (3) Helium-3 abbauen
- (4) Zur Erde bringen
- (5) Quantencomputer bauen
- (6) Beste Tour berechnen

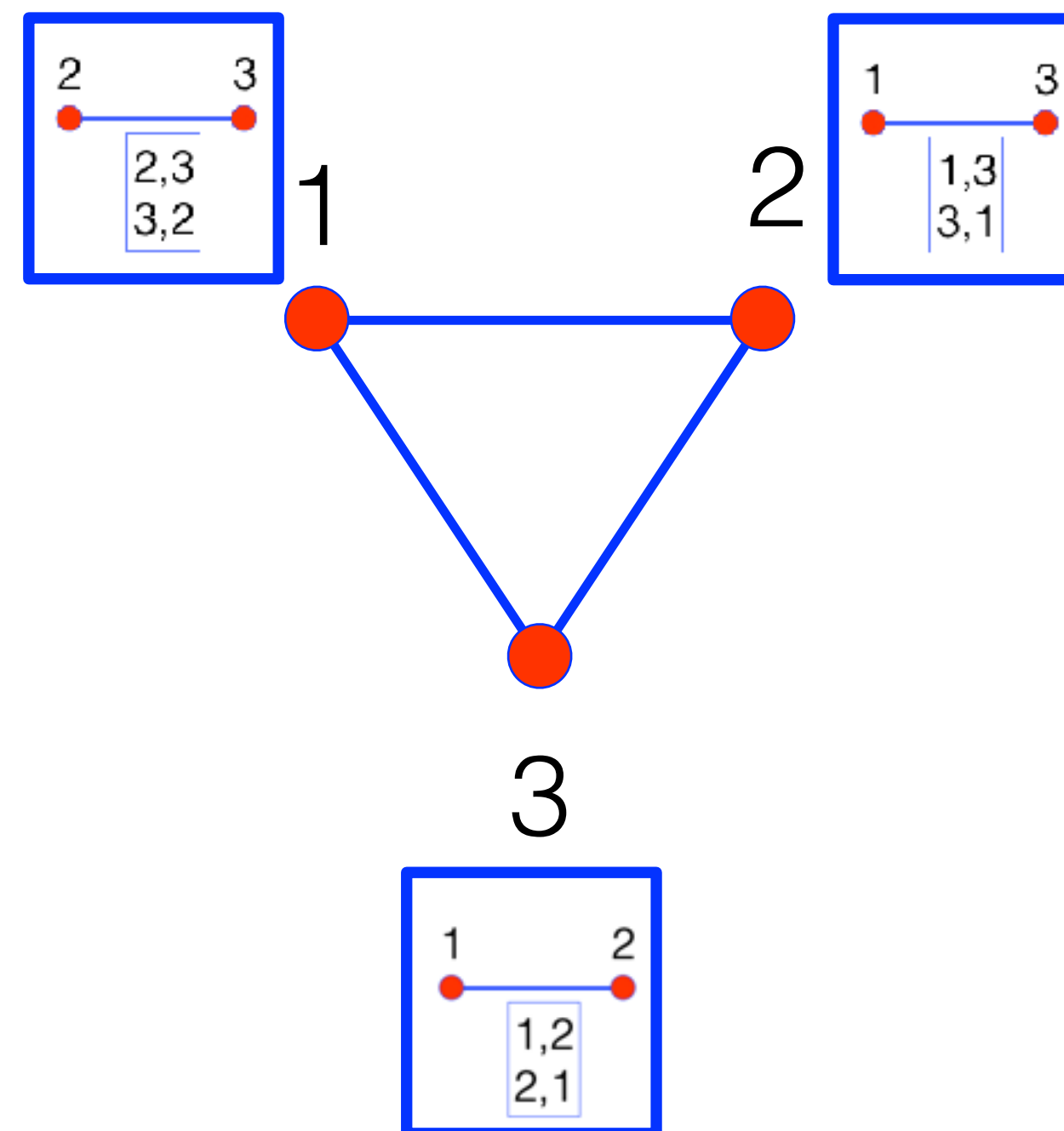
Wie viele mögliche Touren gibt es?

n=2



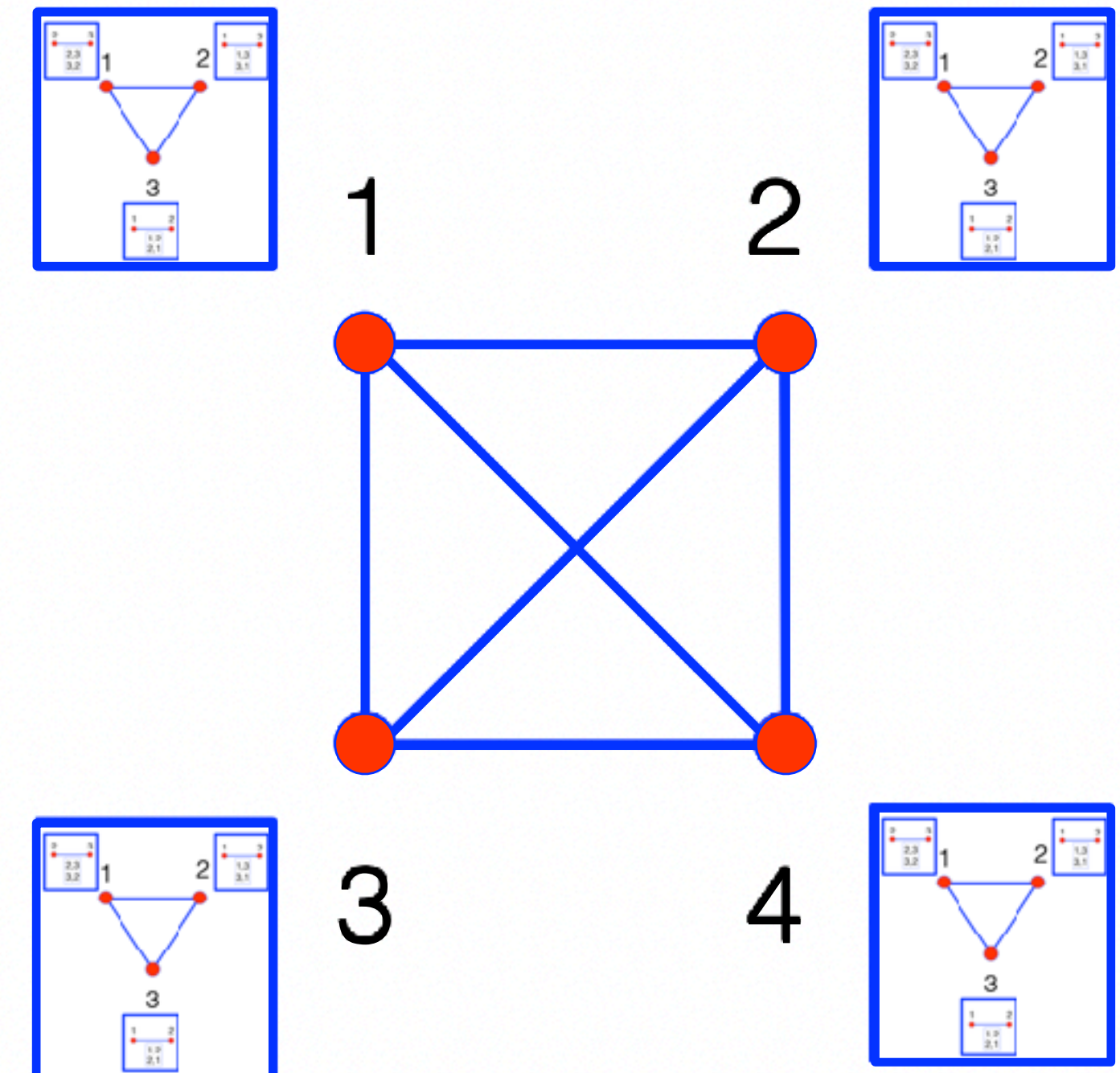
2 Touren

n=3



3*2 = 6 Touren

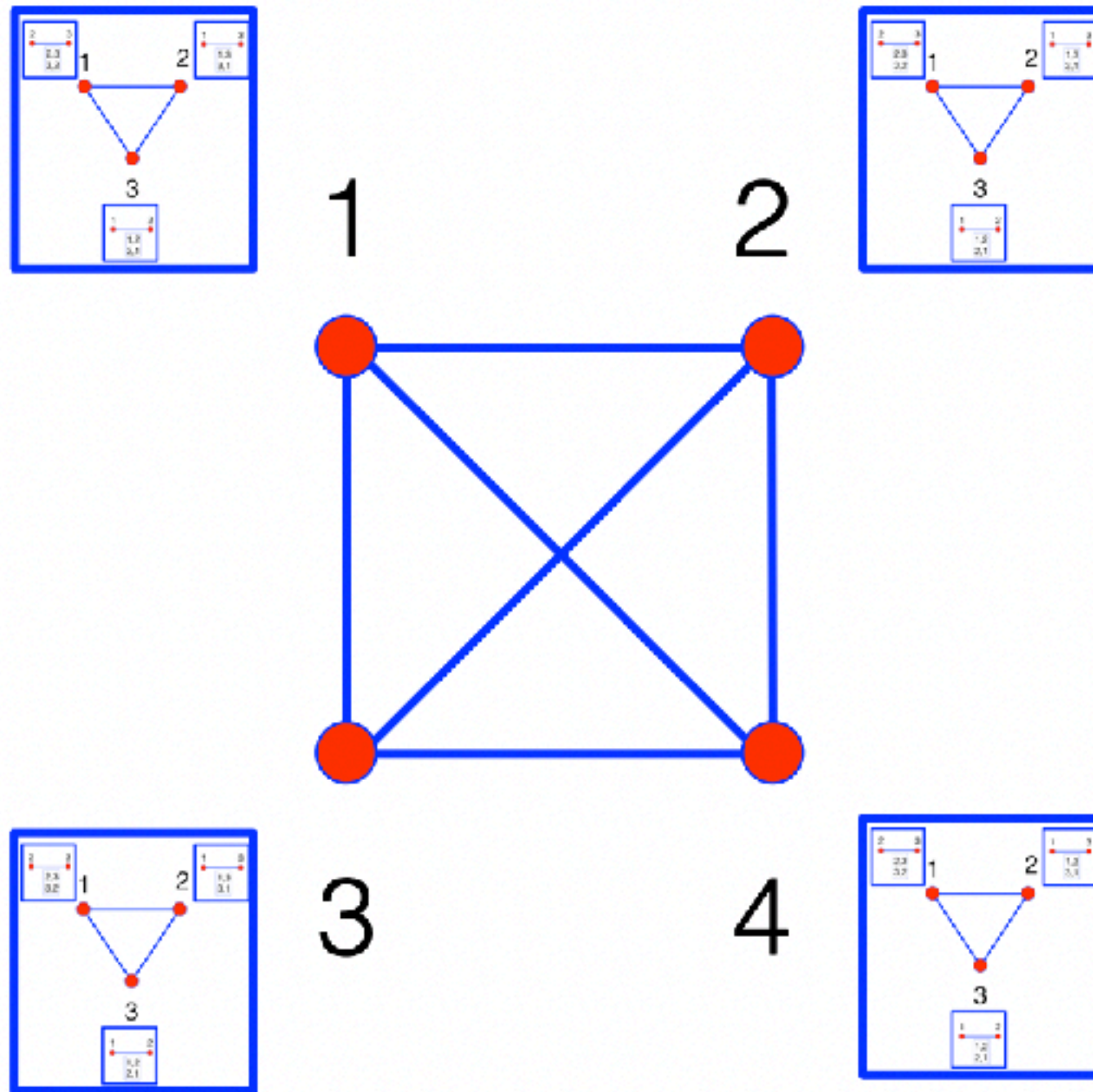
n=4



4*6 = 24 Touren

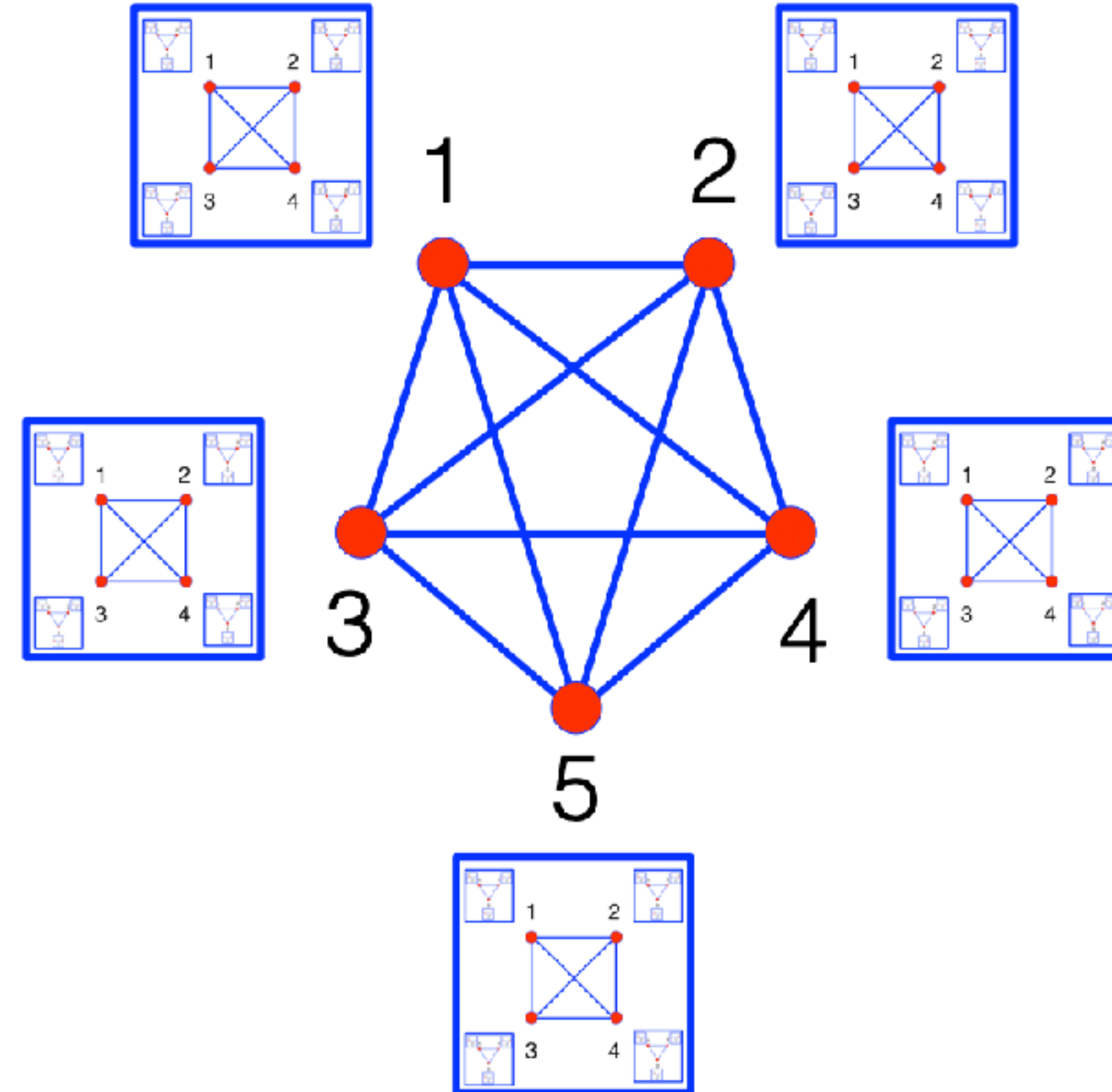
Wie viele mögliche Touren gibt es?

n=4



$4 \cdot 6 = 24$ Touren

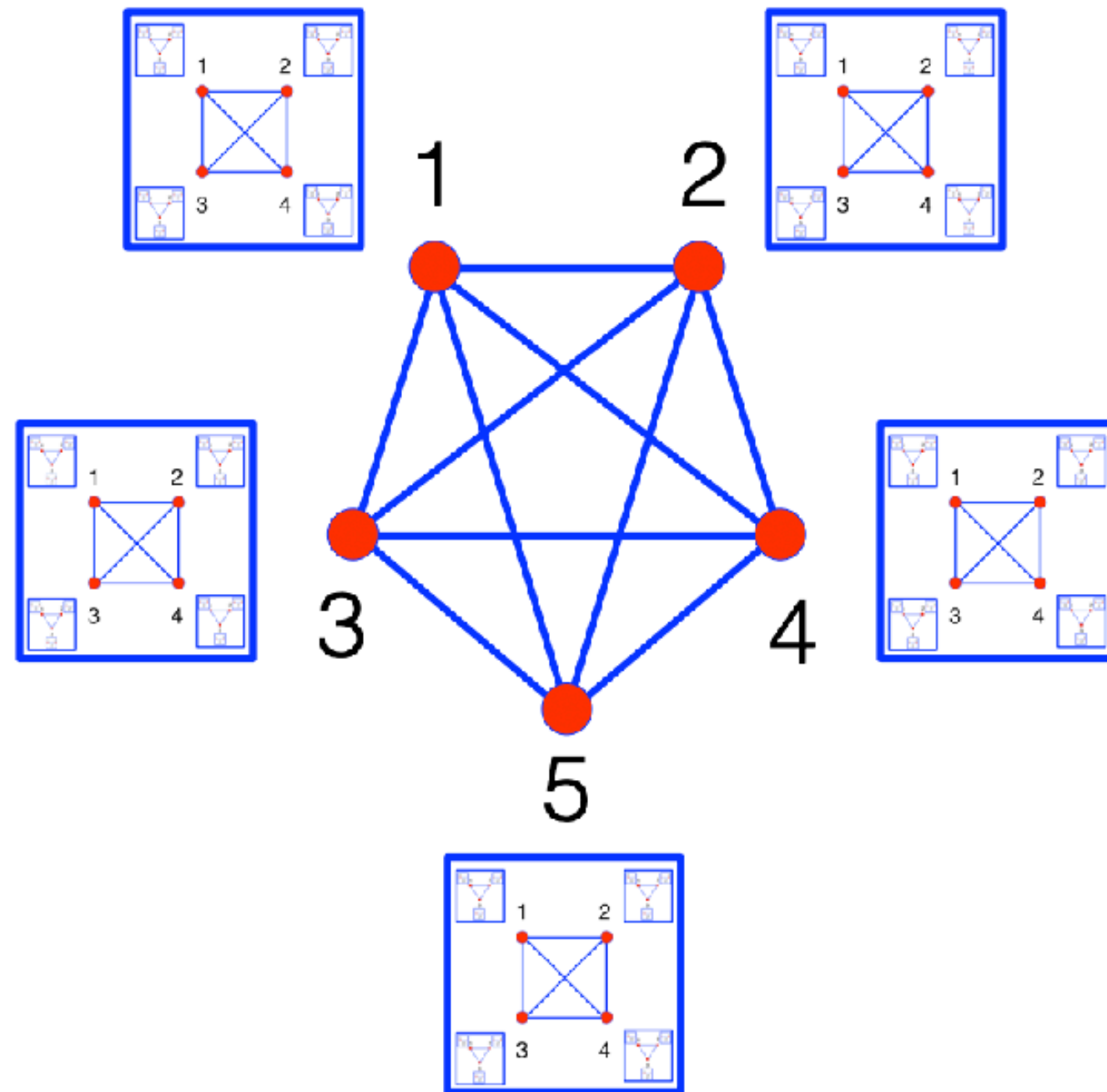
n=5



$5 \cdot 24 = 120$ Touren

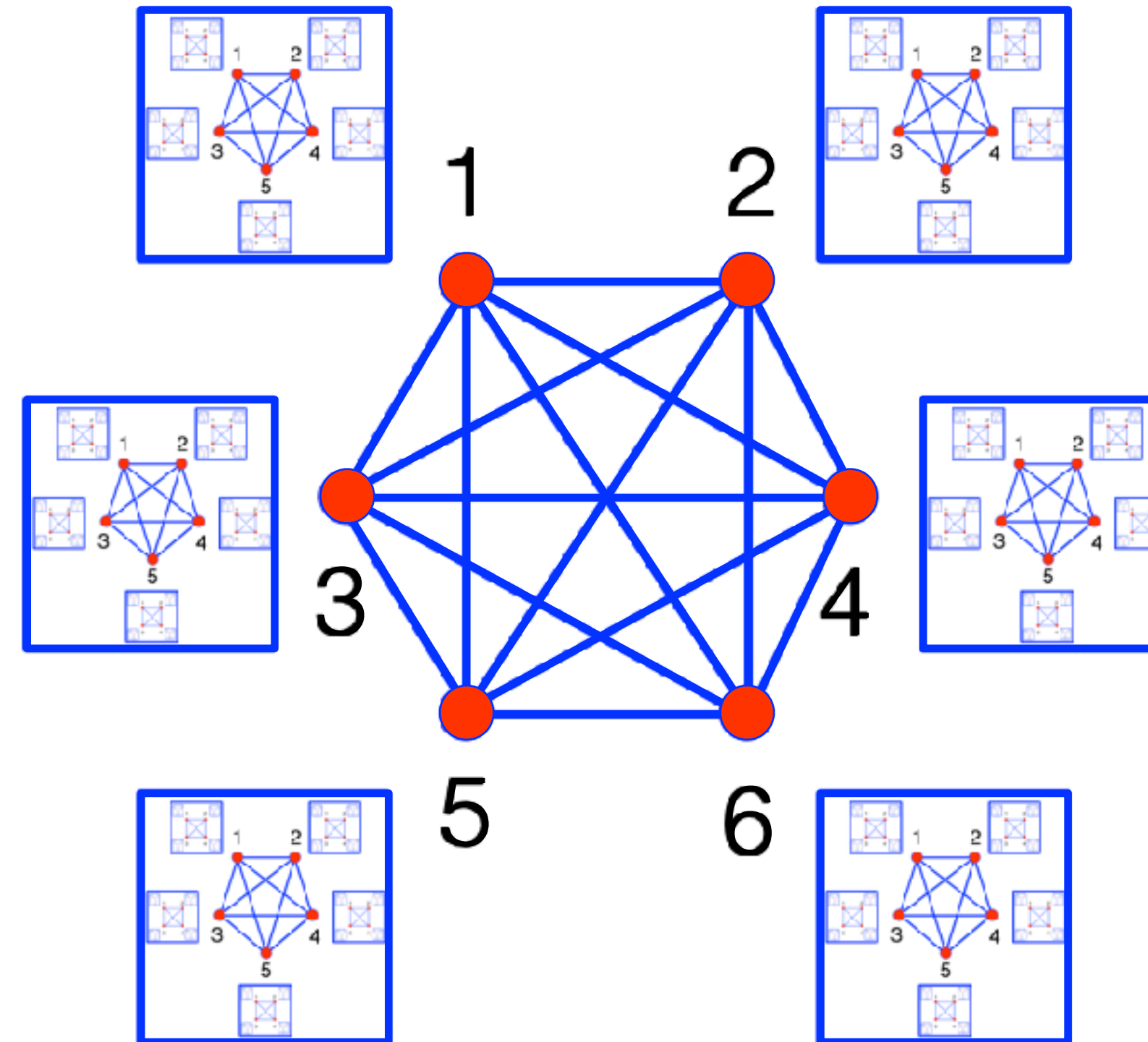
Wie viele mögliche Touren gibt es?

n=5



$5 \cdot 24 = 120$ Touren

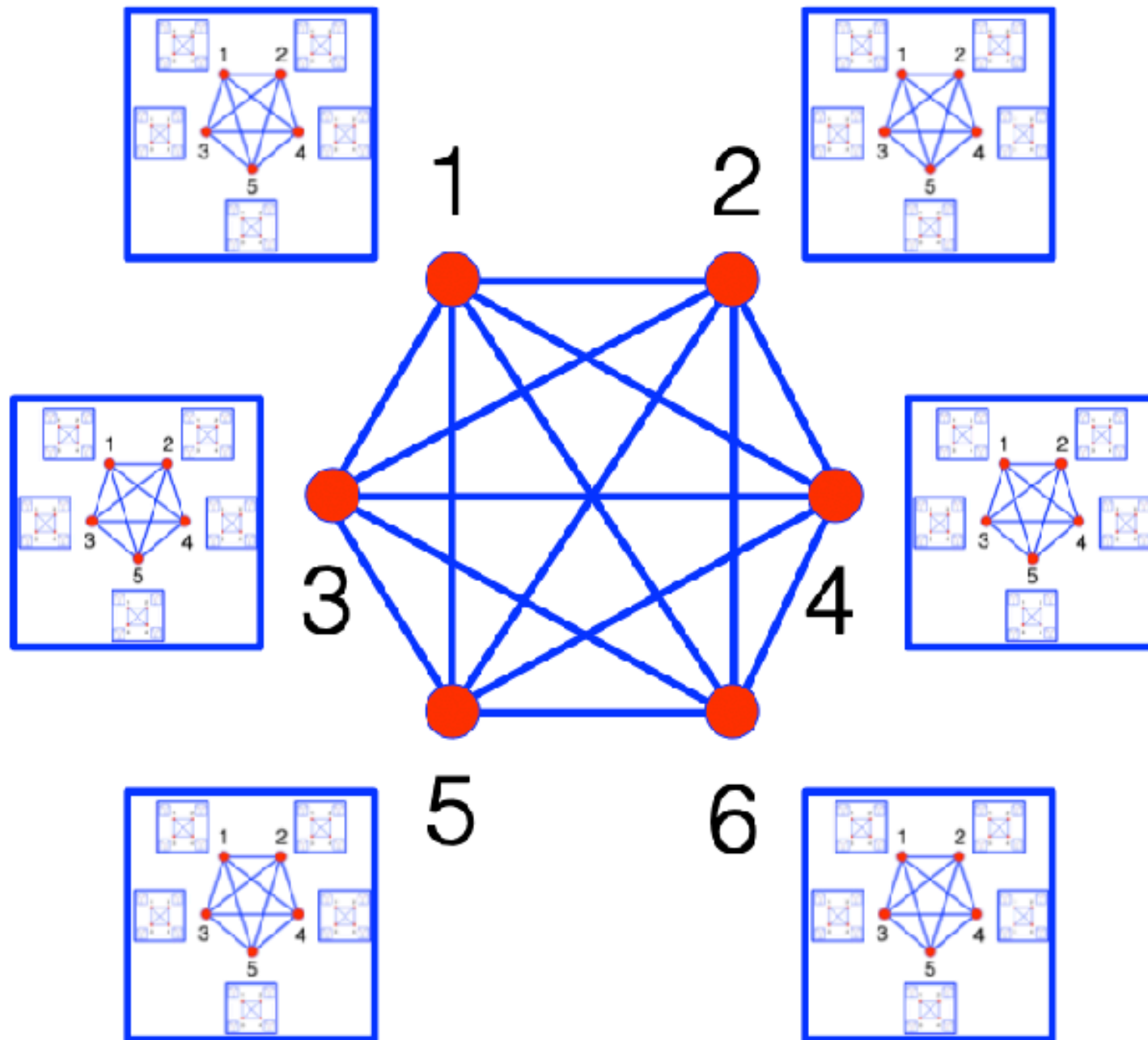
n=6



$6 \cdot 120 = 720$ Touren

Wie viele mögliche Touren gibt es?

n=6



$6 \cdot 120 = 720$ Touren

n=2

$$2 * 1 = 2$$

n=3

$$3 * 2 = 3 * 2 * 1 = 6$$

n=4

$$4 * 6 = 4 * 3 * 2 * 1 = 24$$

n=5

$$5 * 24 = 5 * 4 * 3 * 2 * 1 = 120$$

n=6

$$6 * 120 = 6 * 5 * 4 * 3 * 2 * 1 = 720$$

n=22

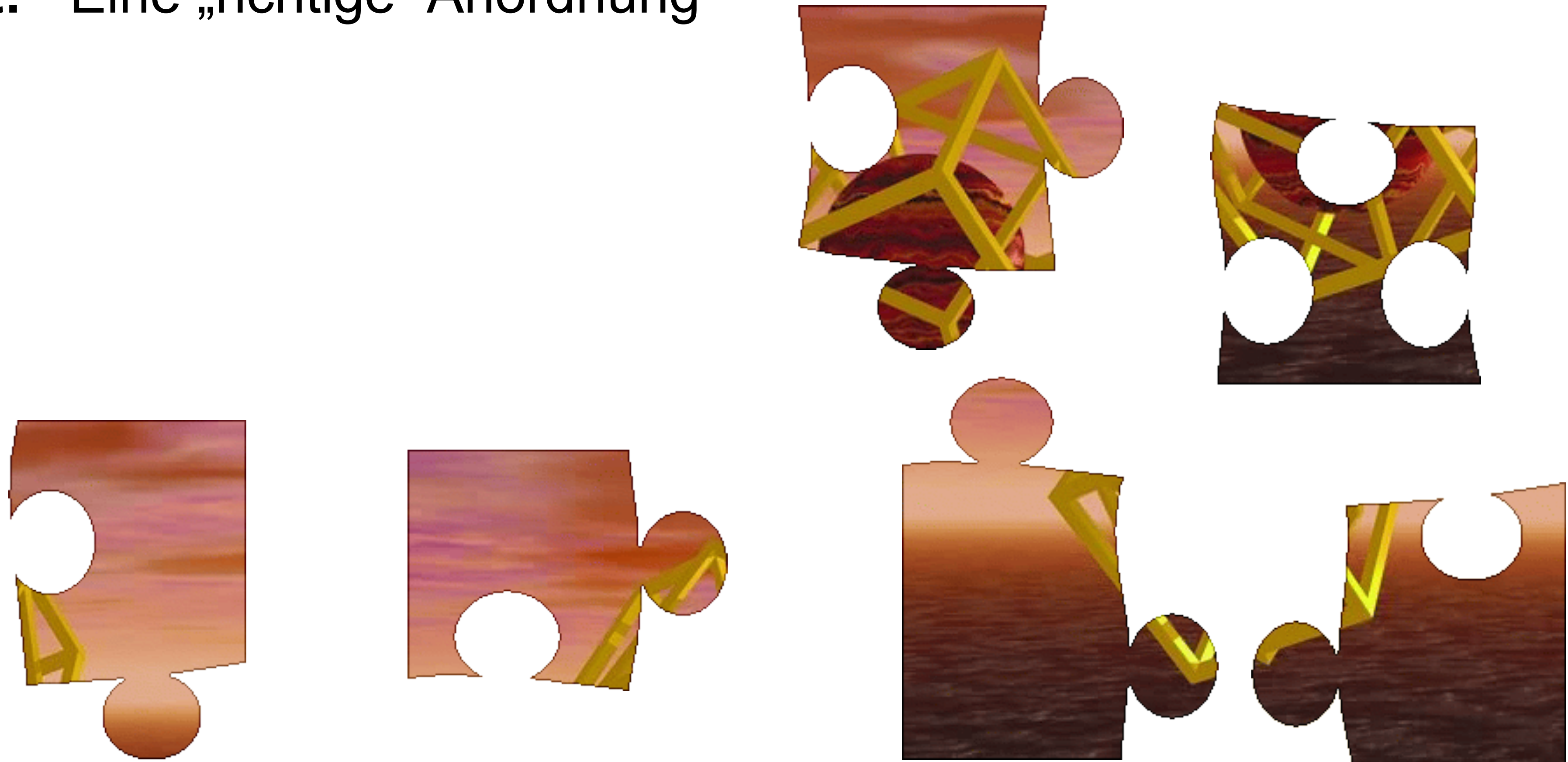
$$22 * 21 * \dots * 2 * 1 = 1.124.000.727.777.607.680.000$$

**Bei 3,5 GHz:
31.557.600.000 Sekunden
~ 10.000 Jahre**

Wie lange dauert es, ein Puzzle zu lösen?

Gegeben: Eine Menge von n Puzzleteilen

Gesucht: Eine „richtige“ Anordnung

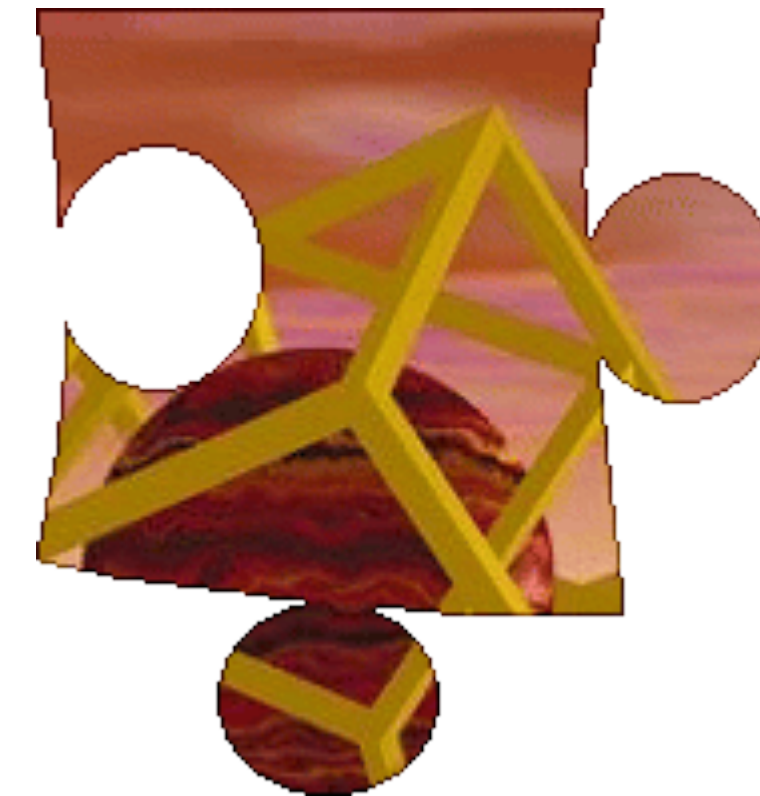


Wie lange dauert es, ein Puzzle zu lösen?

Gegeben: Eine Menge von n Puzzleteilen

Gesucht: Eine „richtige“ Anordnung

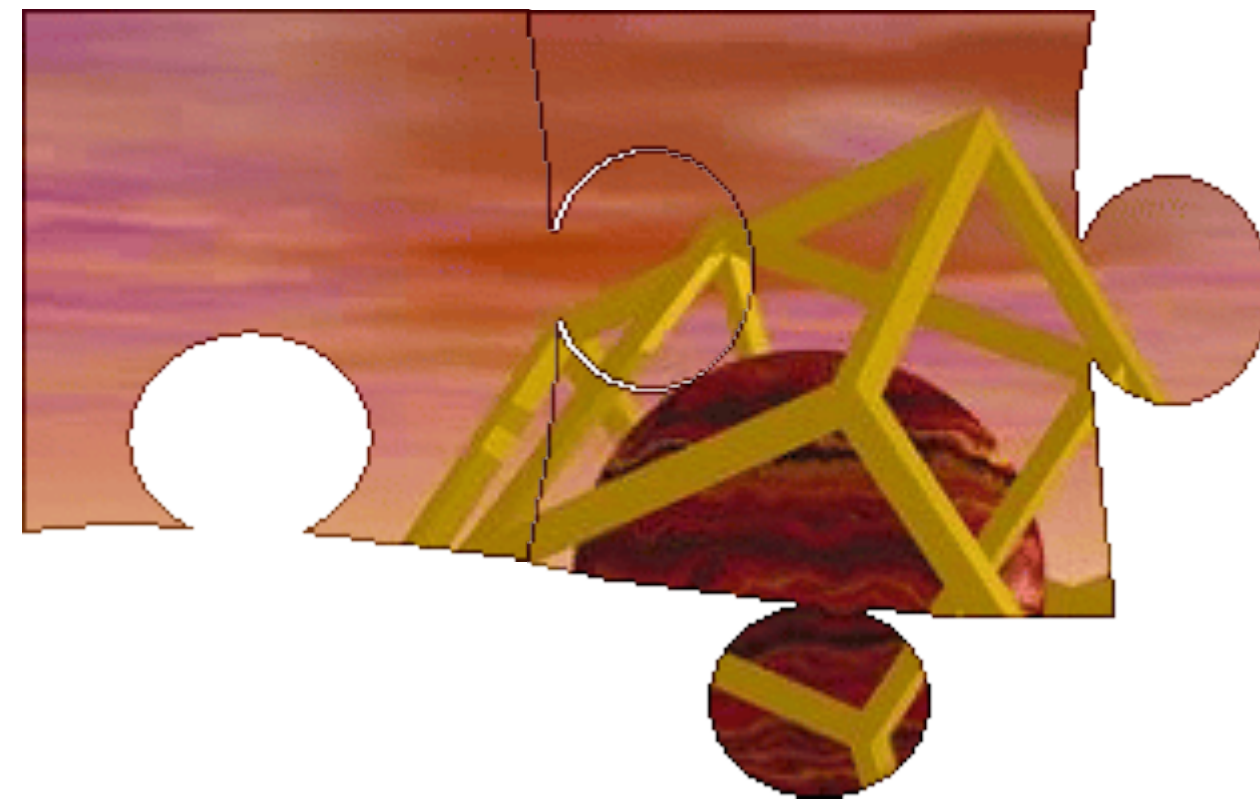
? ? ? ? ? ?



Wie lange dauert es, ein Puzzle zu lösen?

Gegeben: Eine Menge von n Puzzleteilen

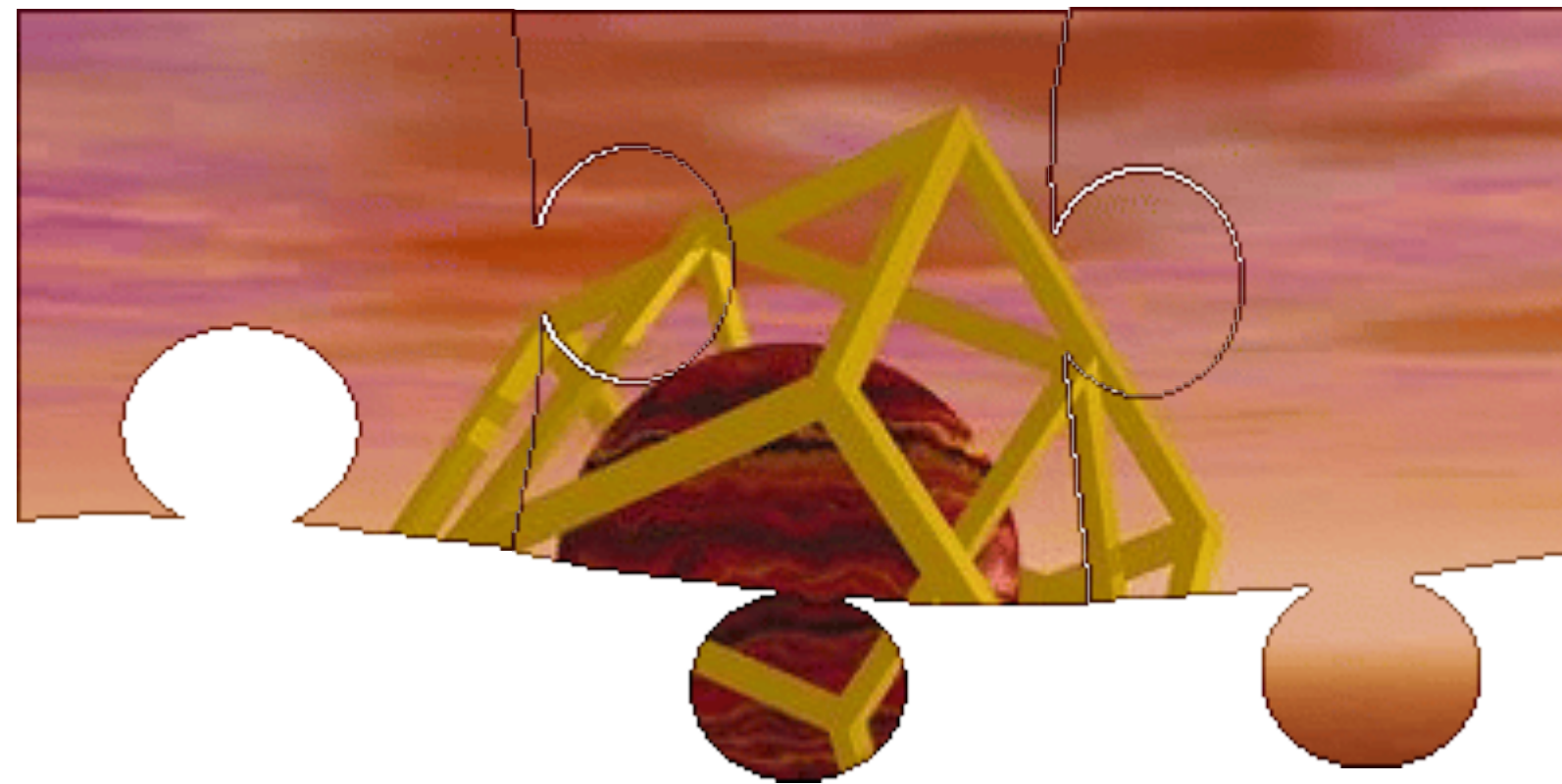
Gesucht: Eine „richtige“ Anordnung



Wie lange dauert es, ein Puzzle zu lösen?

Gegeben: Eine Menge von n Puzzleteilen

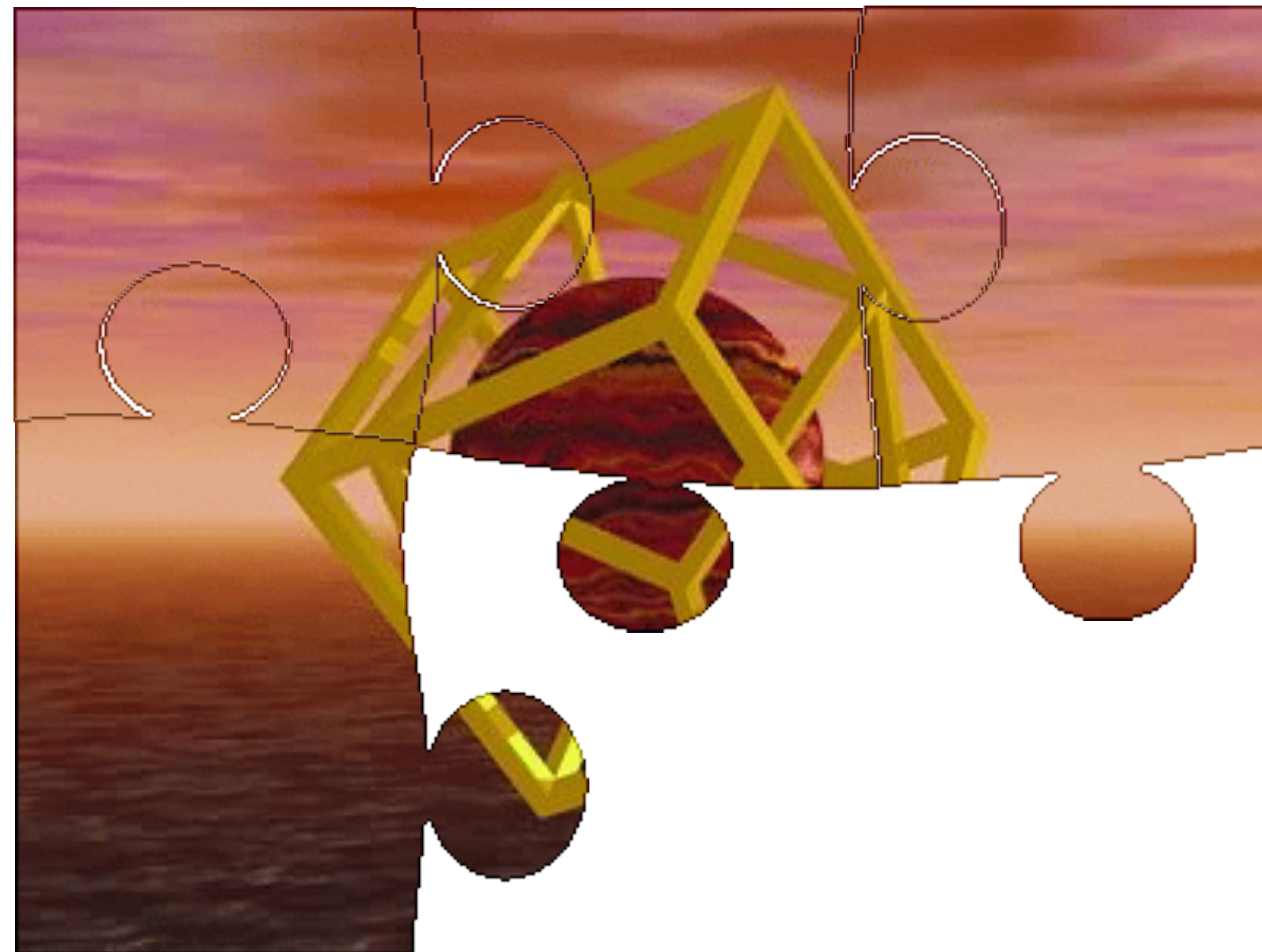
Gesucht: Eine „richtige“ Anordnung



Wie lange dauert es, ein Puzzle zu lösen?

Gegeben: Eine Menge von n Puzzleteilen

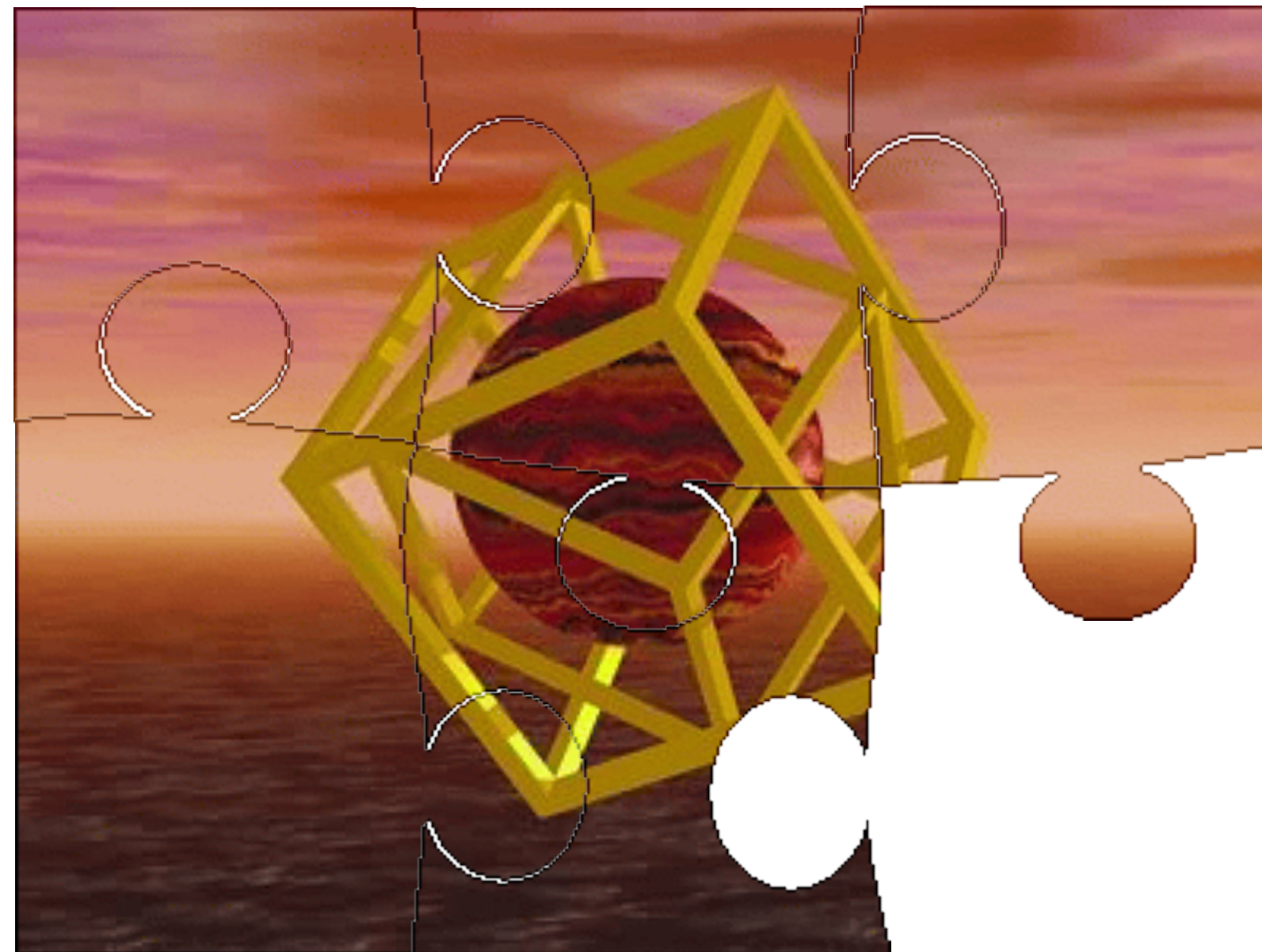
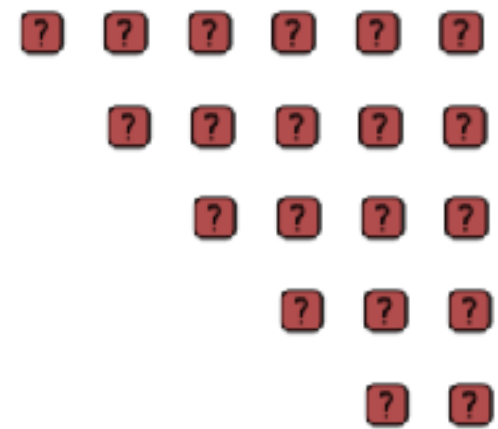
Gesucht: Eine „richtige“ Anordnung



Wie lange dauert es, ein Puzzle zu lösen?

Gegeben: Eine Menge von n Puzzleteilen

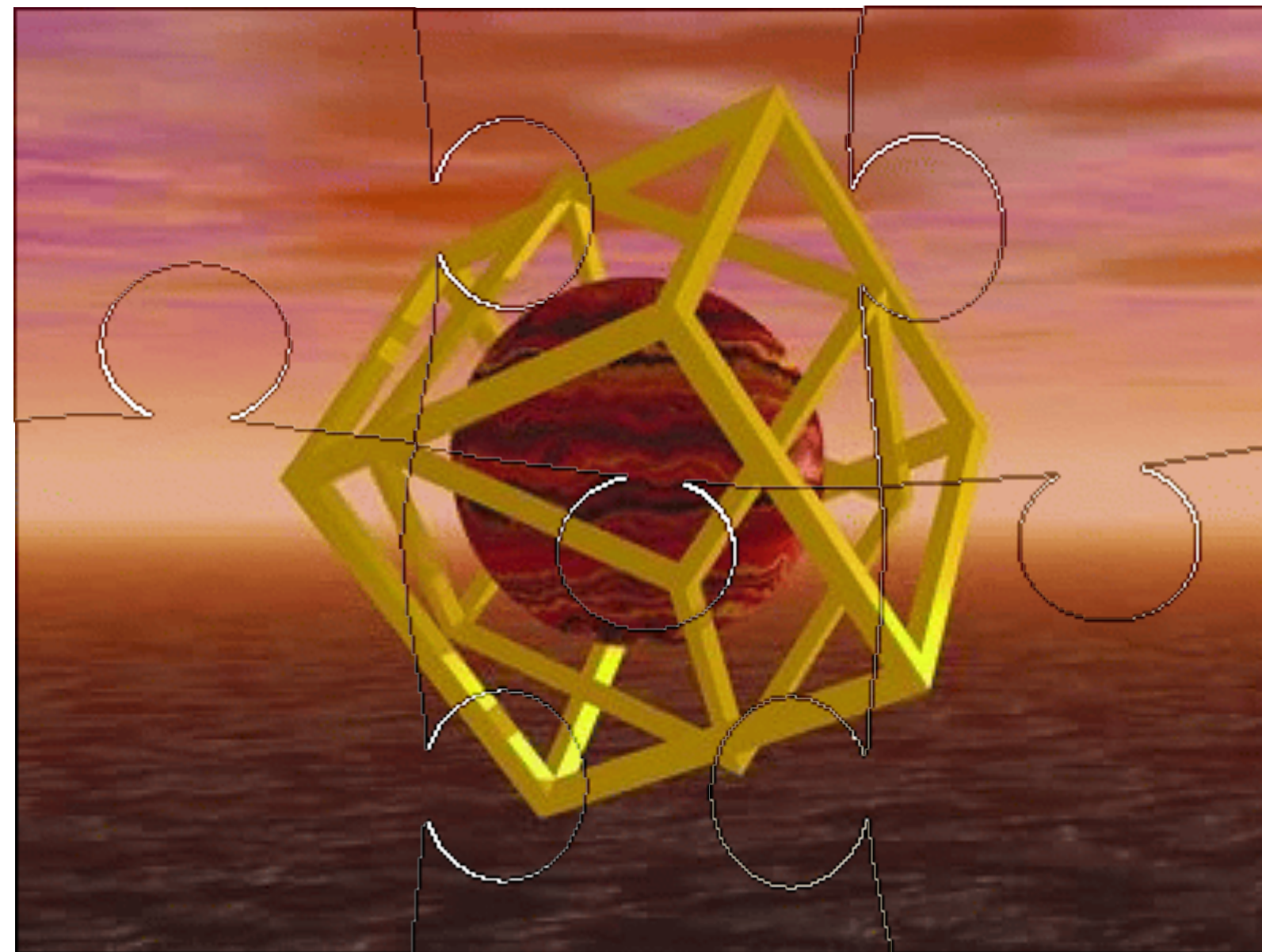
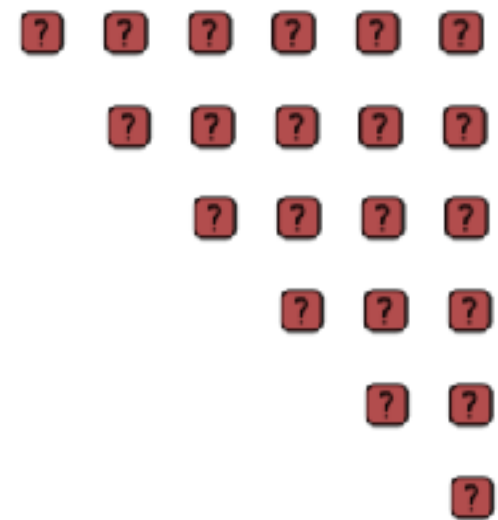
Gesucht: Eine „richtige“ Anordnung



Wie lange dauert es, ein Puzzle zu lösen?

Gegeben: Eine Menge von n Puzzleteilen

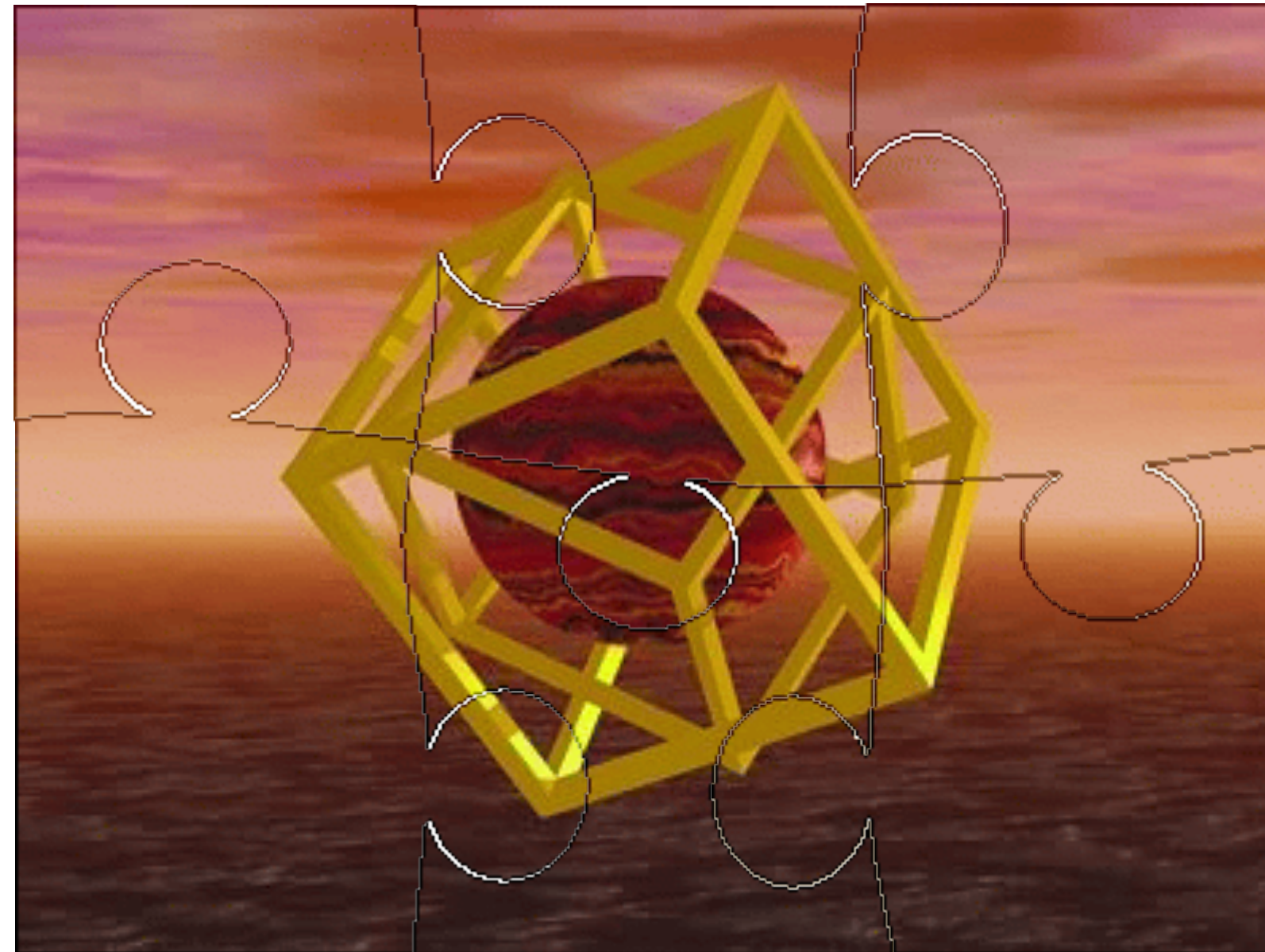
Gesucht: Eine „richtige“ Anordnung



Wie lange dauert es, ein Puzzle zu lösen?

Gegeben: Eine Menge von n Puzzleteilen

Gesucht: Eine systematische Lösungsmethode



Einfach so:

$$6+5+4+3+2+1 = 21$$

$$1+2+3+4+5+6 = 21$$

$$\hline 7+7+7+7+7+7 = 42$$

Wie lange braucht man für n Teile?

$$\boxed{n(n+1)/2}$$

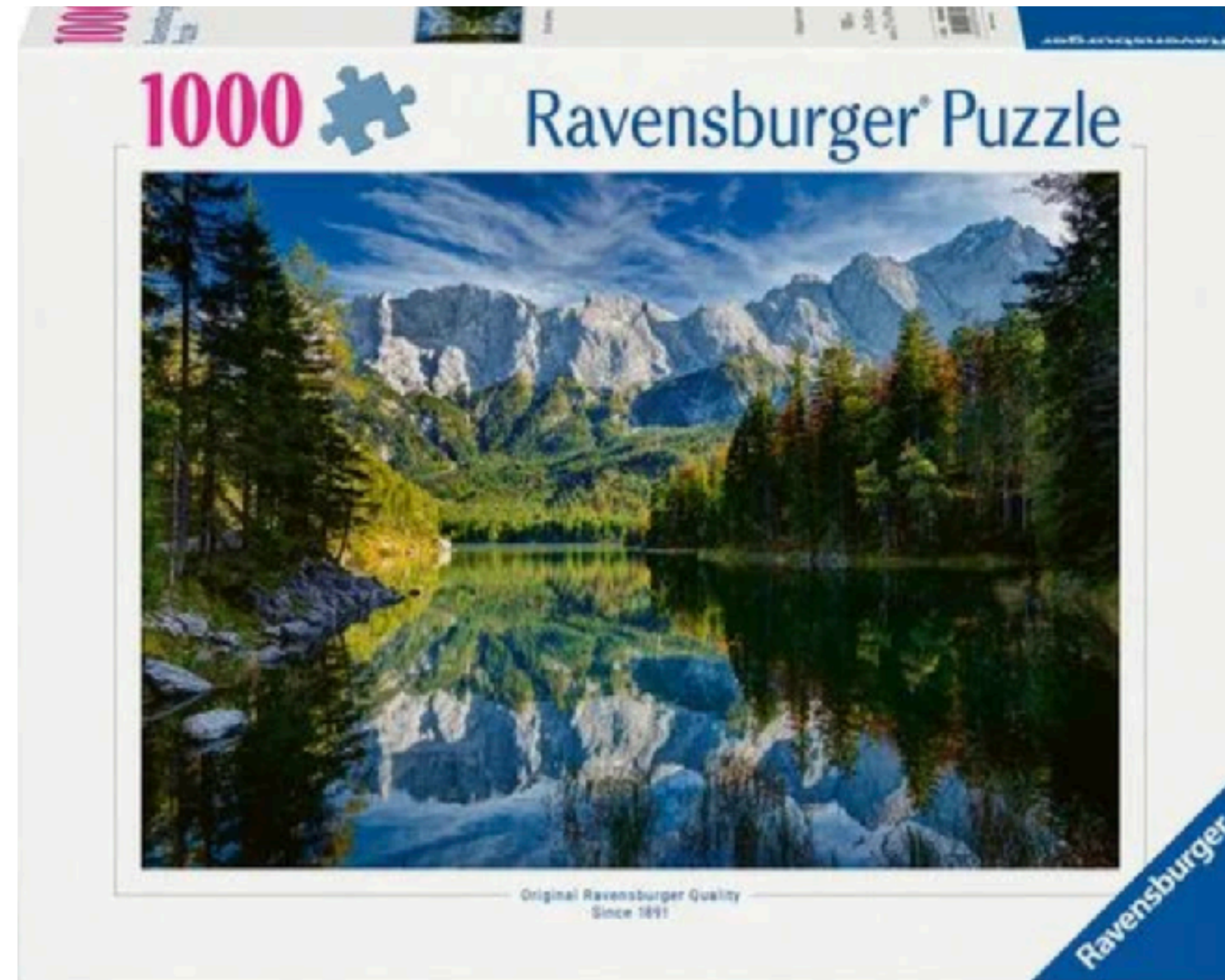
$n = 6:$ 21 Schritte

$n = 100:$ 5050 Schritte

Wie lange dauert es, ein Puzzle zu lösen?

Gegeben: Eine Menge von n Puzzleteilen

Gesucht: Eine systematische Lösungsmethode



Einfach so:

$$6+5+4+3+2+1 = 21$$

$$1+2+3+4+5+6 = 21$$

$$7+7+7+7+7+7 = 42$$

Wie lange braucht man für n Teile?

$$\boxed{n(n+1)/2}$$

$n = 6$: 21 Schritte

$n = 100$: 5050 Schritte

$n = 1000$: 500.500 Schritte

Wie lange dauert es, ein Puzzle zu lösen?

Gegeben: Eine Menge von n Puzzleteilen

Gesucht: Eine systematische Lösungsmethode

Einfach so:

$$6+5+4+3+2+1 = 21$$

$$1+2+3+4+5+6 = 21$$

$$7+7+7+7+7+7 = 42$$

Wie lange braucht man für n Teile?

$$n(n+1)/2$$

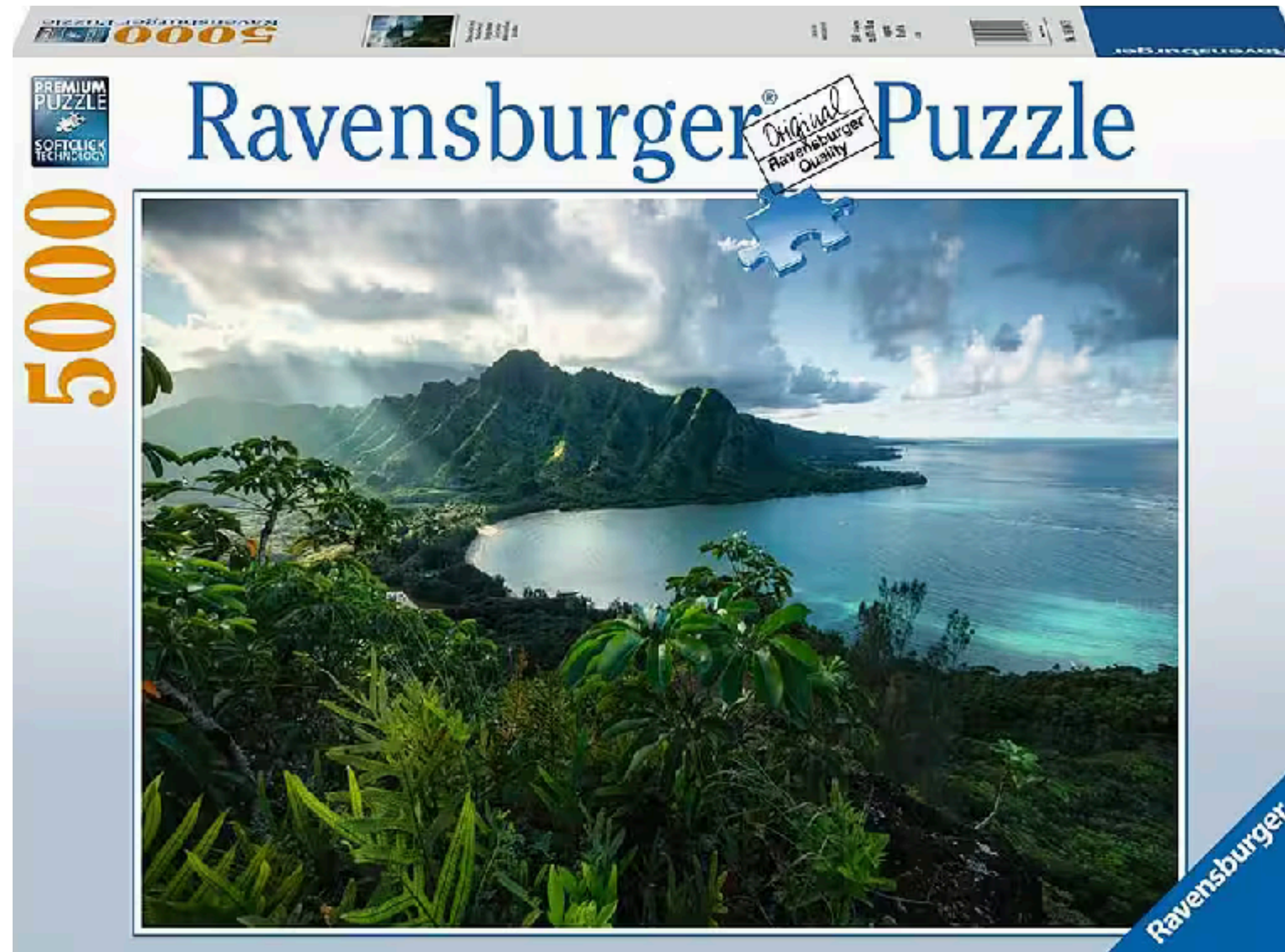
$$\Theta(n^2)$$

$n = 6$: 21 Schritte

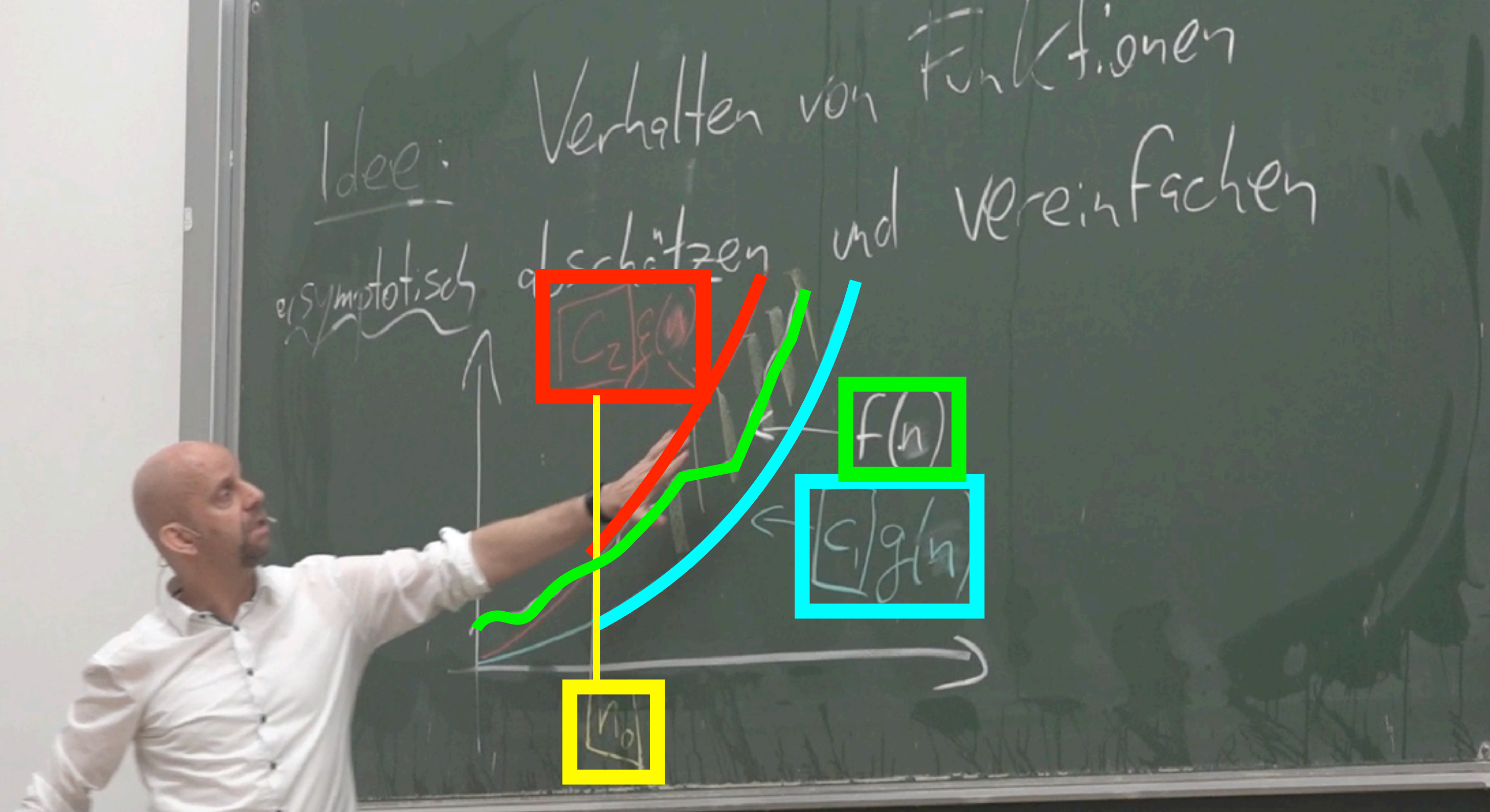
$n = 100$: 5050 Schritte

$n = 1000$: 500.500 Schritte

$n = 5000$: 12.502.500 Schritte



„Geht's nicht noch etwas schneller?“



DEFINITION 3.9 (Θ -Notation)

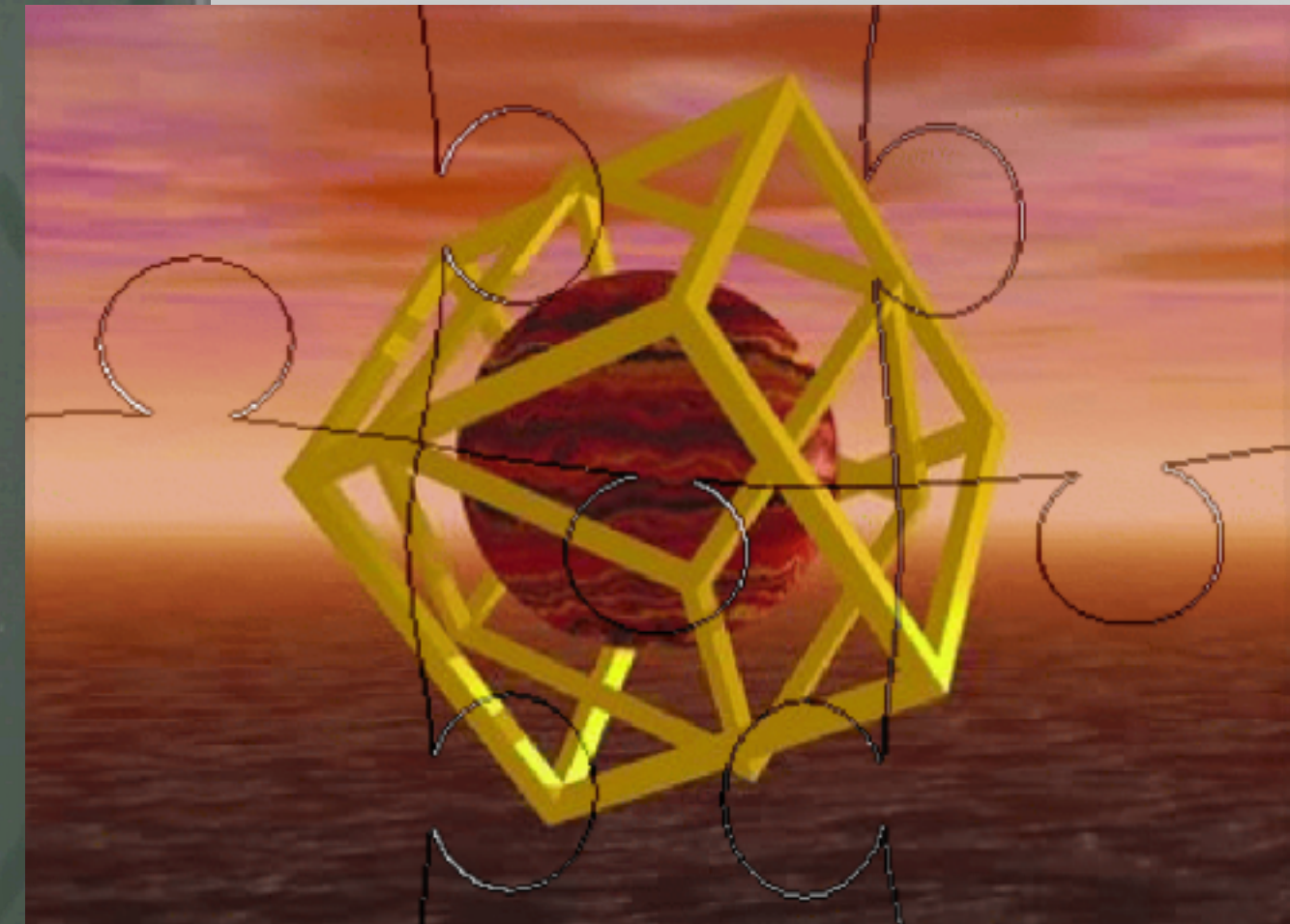
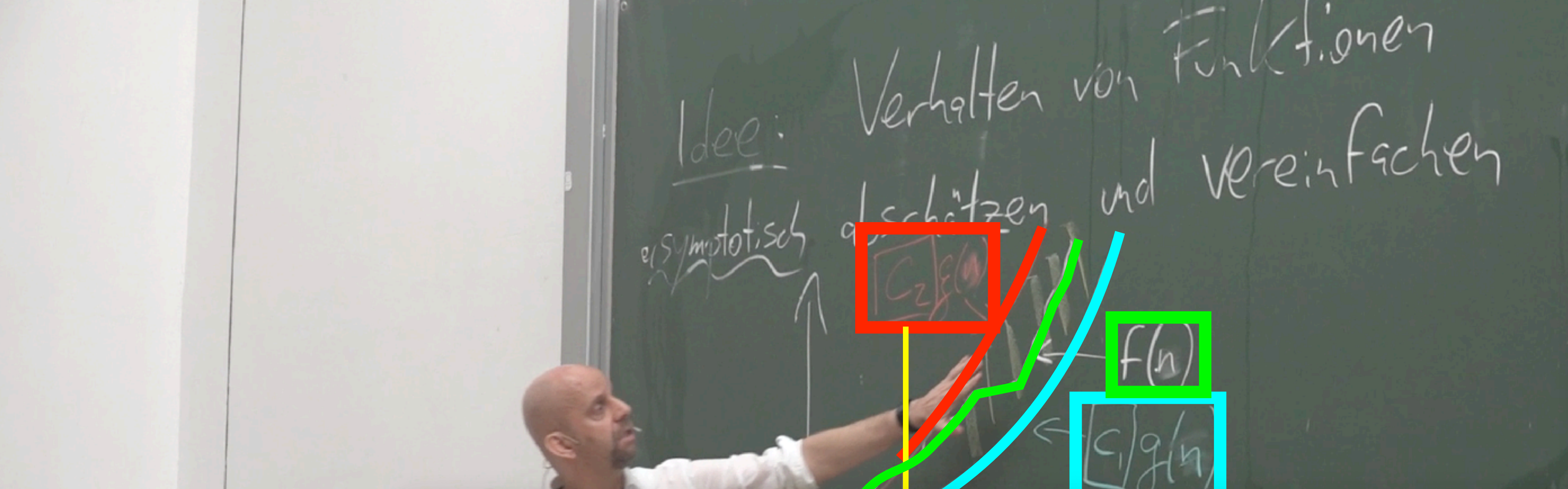
Seien $f, g: \mathbb{N} \rightarrow \mathbb{R}$ Funktionen.

Dann gilt

$$f \in \Theta(g) \Leftrightarrow \text{Es gibt positive Konstanten } c, c_2, n_0 \text{ mit}$$

$$0 \leq c g(n) \leq f(n) \leq c_2 g(n) \text{ für alle } n \geq n_0$$

Man sagt: f wächst asymptotisch in derselben Größenordnung wie g .



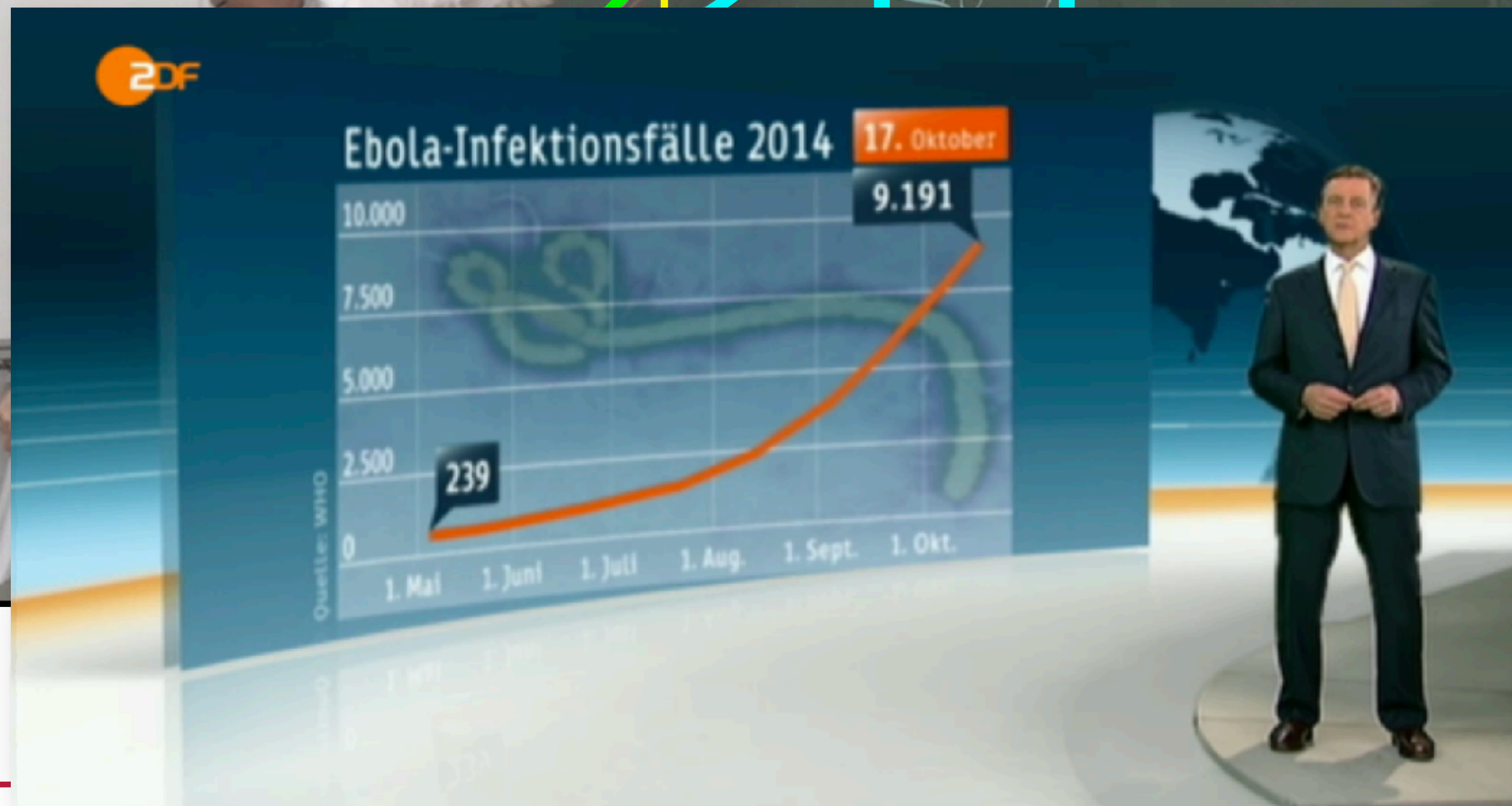
Wie lange braucht man für n Teile?

$$\boxed{n(n+1)/2} \quad \Theta(n^2)$$

$n = 6$: 21 Schritte
 $n = 100$: 5050 Schritte
 $n = 1000$: 500.500 Schritte
 $n = 5000$: 12.502.500 Schritte

$$\Theta(n^2) \quad (2n)^2 = 4 \cdot n^2$$

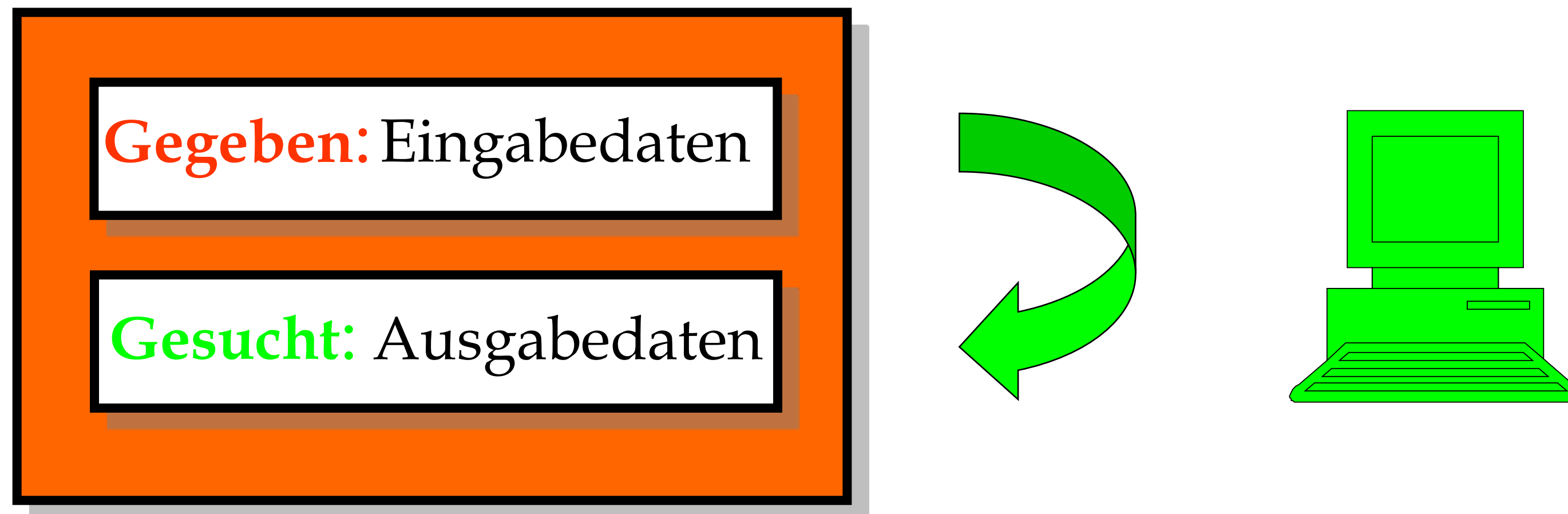
$$\Theta(2^n) \quad 2^{2n} = 2^n \cdot 2^n$$



22.10.25 | Sándor P. Fekete | AuD VL0: Geht's nicht noch etwas schneller?

Konstruktives Lösen von Problemen

Gegeben: Eine Problembeschreibung



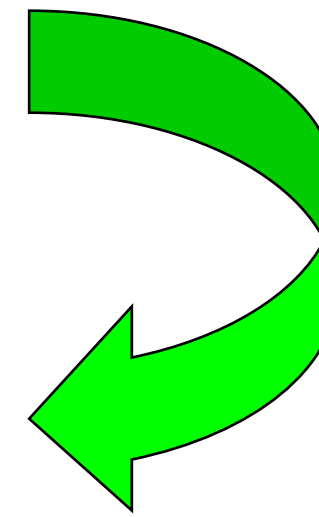
Gesucht: Ein Algorithmus, der das Problem löst

Die Laufzeit von Algorithmen

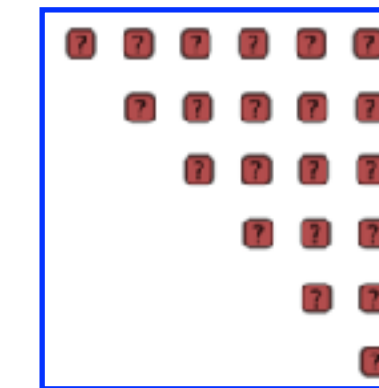
Gegeben: Eine Problembeschreibung der Größe $O(n)$



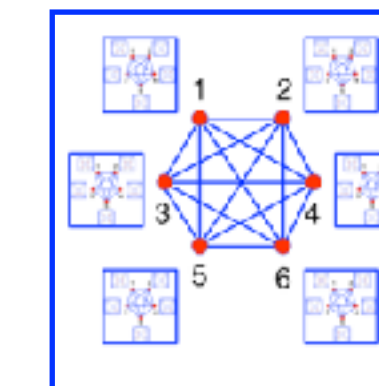
Die Klasse P



Gesucht: Ein polynomialer Algorithmus



$$\Theta(n^2)$$



$$\Theta(2^n)$$

$$\Theta(n!)$$

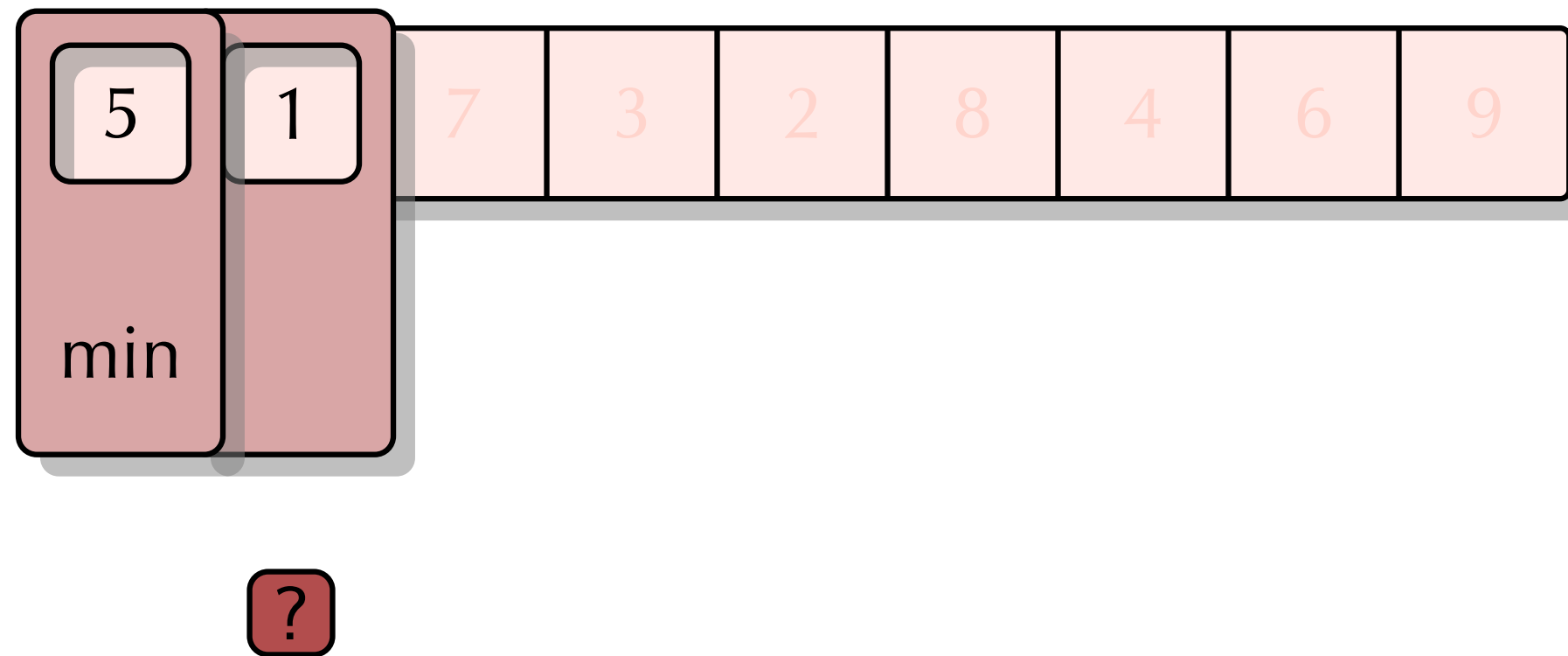


2 Sortieren geht über Probieren

Zwei Sortialgorithmen

Selection Sort:

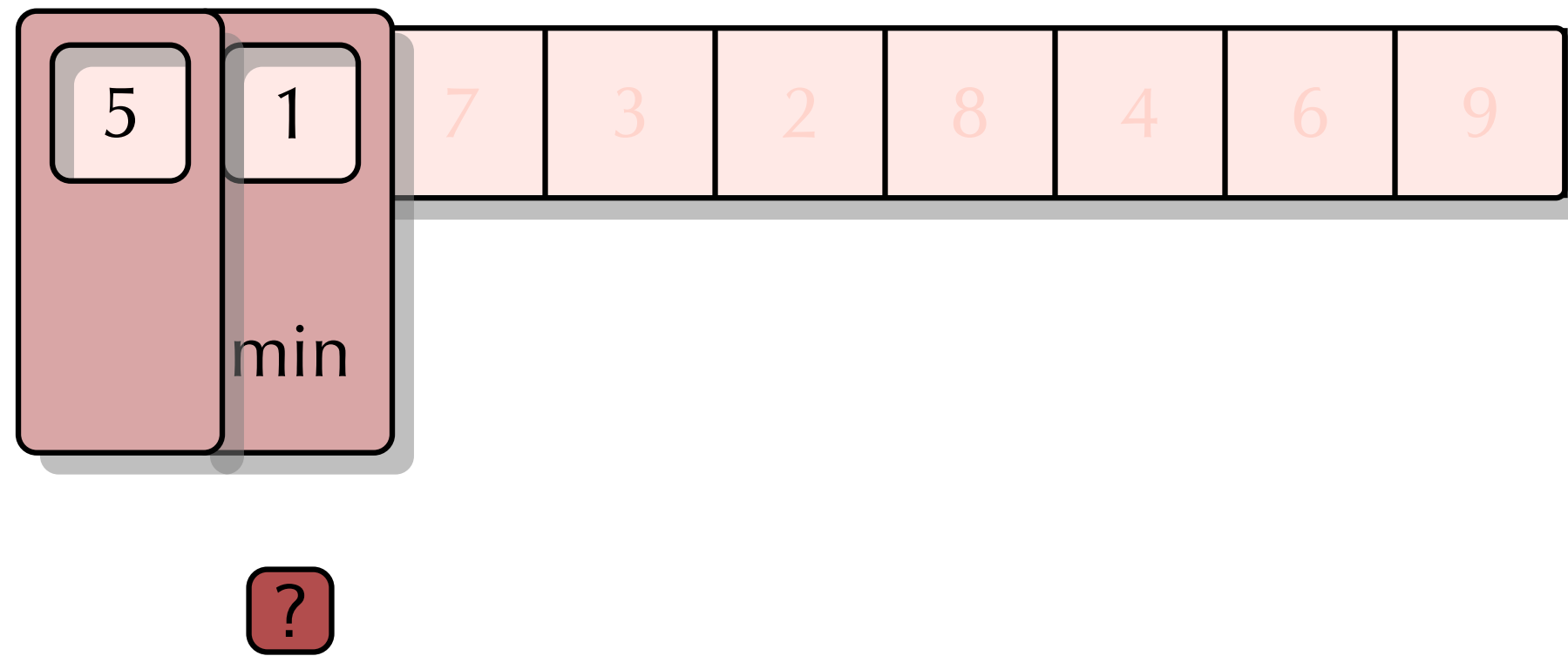
- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle



Zwei Sortialgorithmen

Selection Sort:

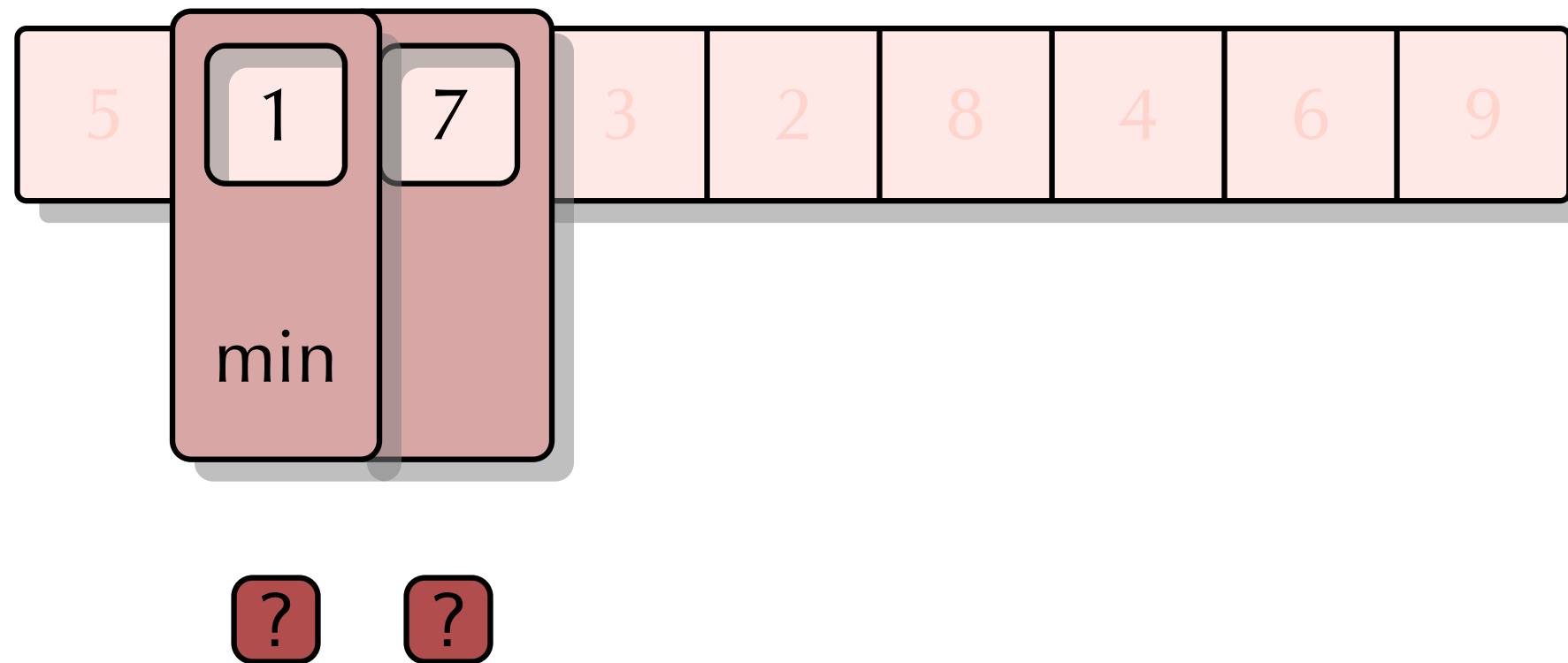
- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle



Zwei Sortialgorithmen

Selection Sort:

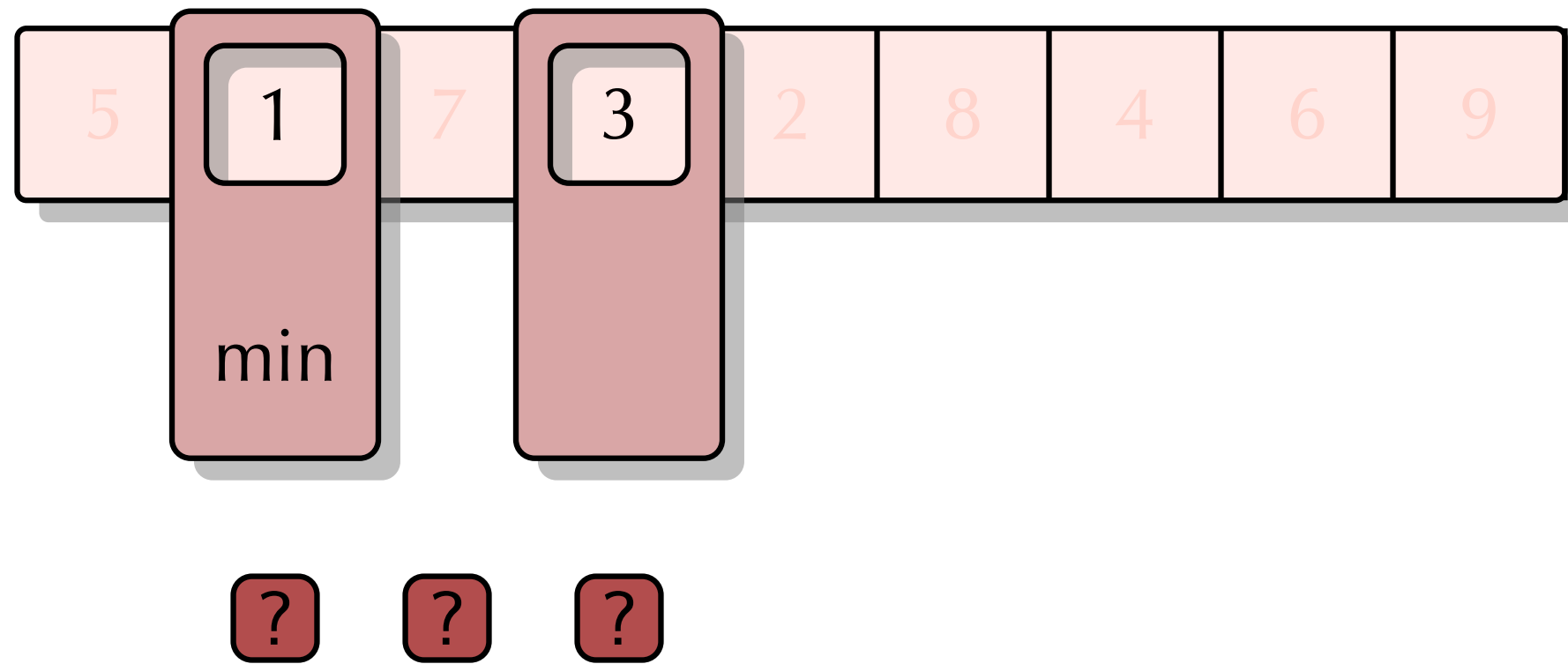
- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle



Zwei Sortialgorithmen

Selection Sort:

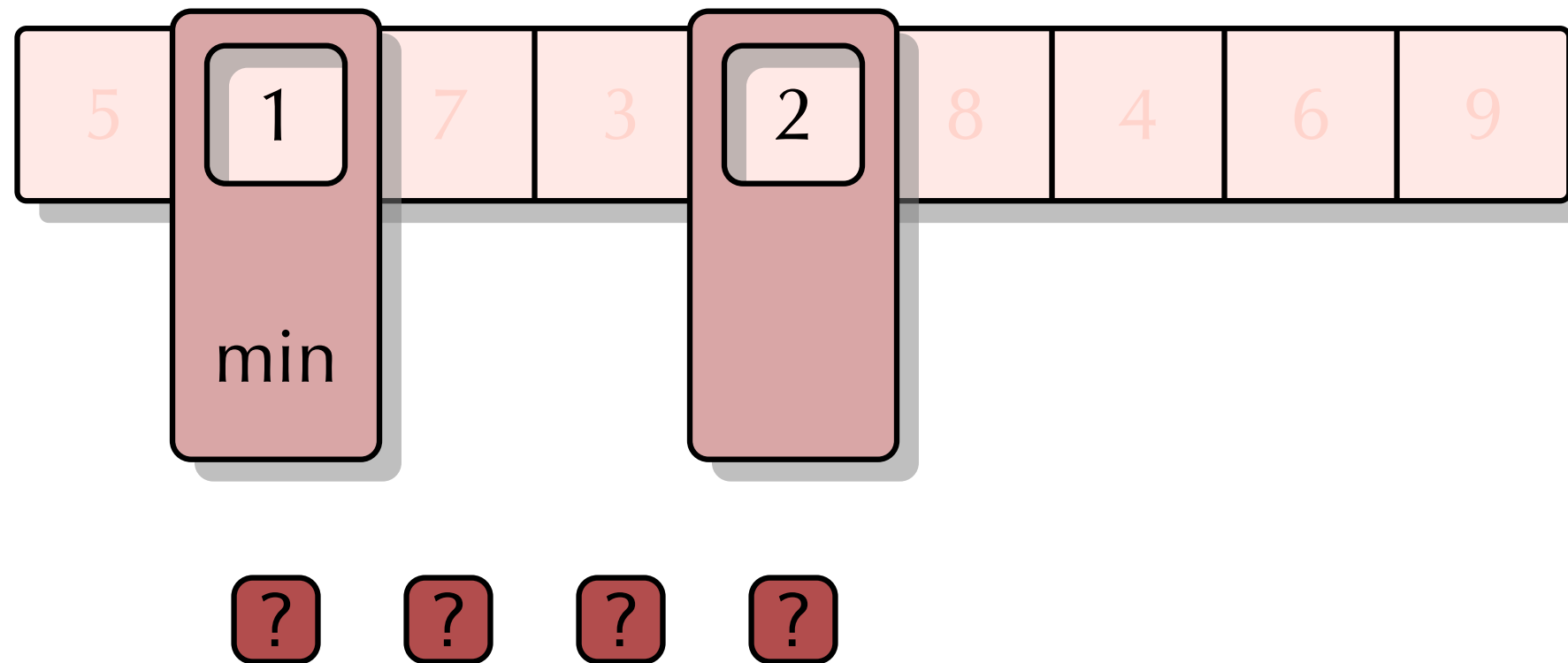
- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle



Zwei Sortialgorithmen

Selection Sort:

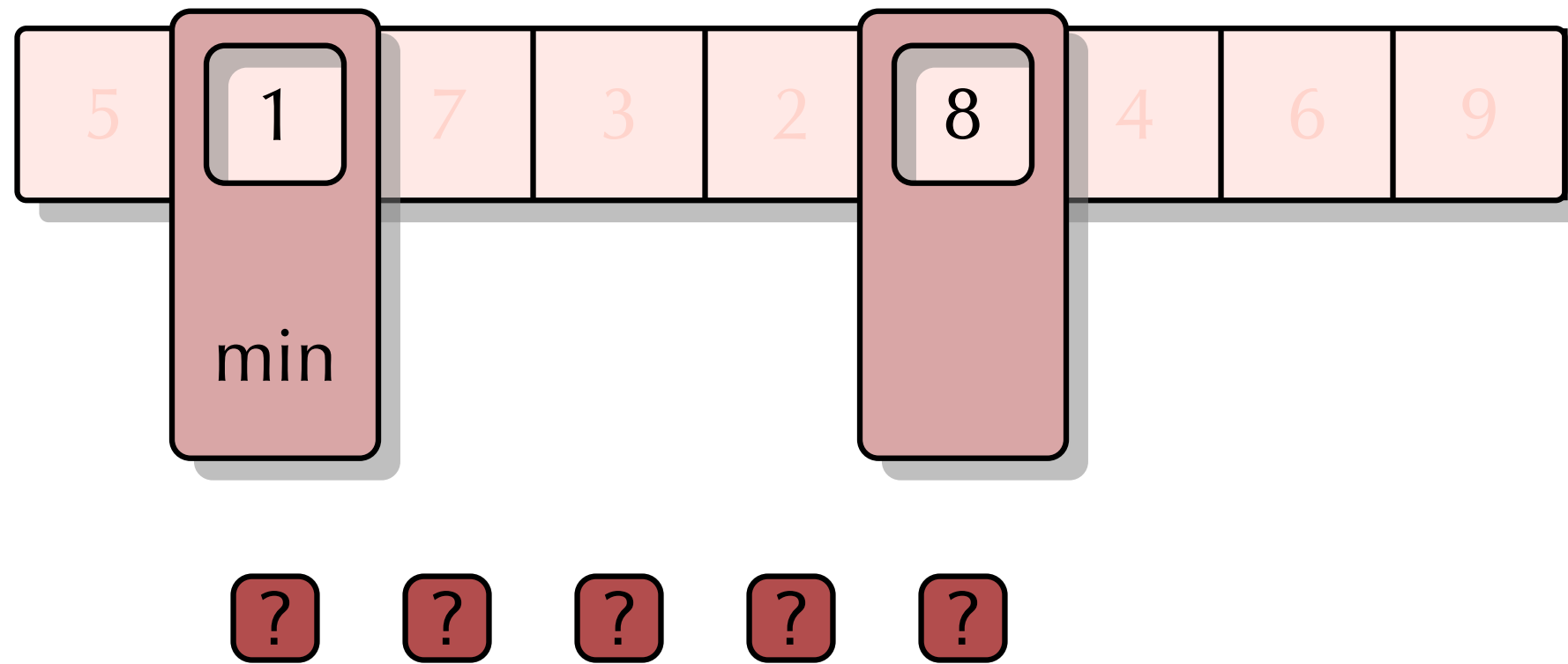
- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle



Zwei Sortialgorithmen

Selection Sort:

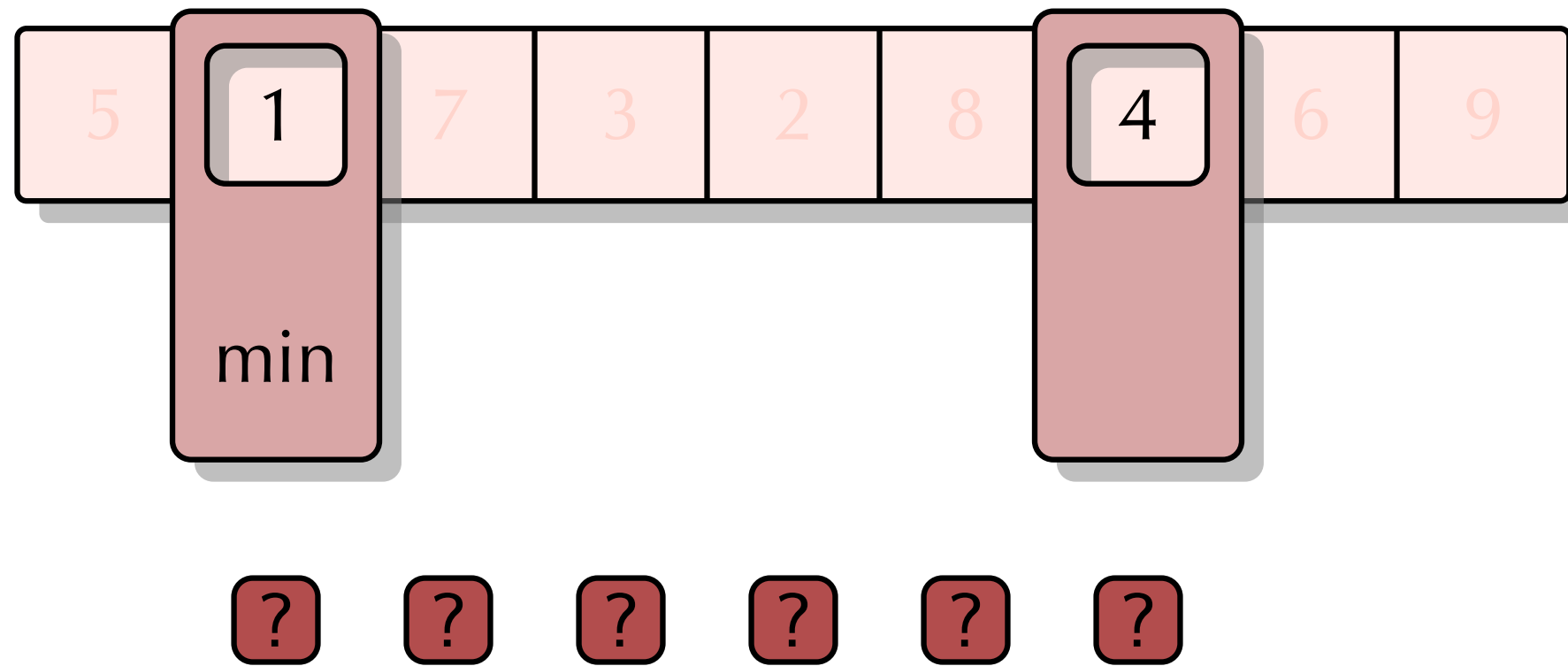
- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle



Zwei Sortialgorithmen

Selection Sort:

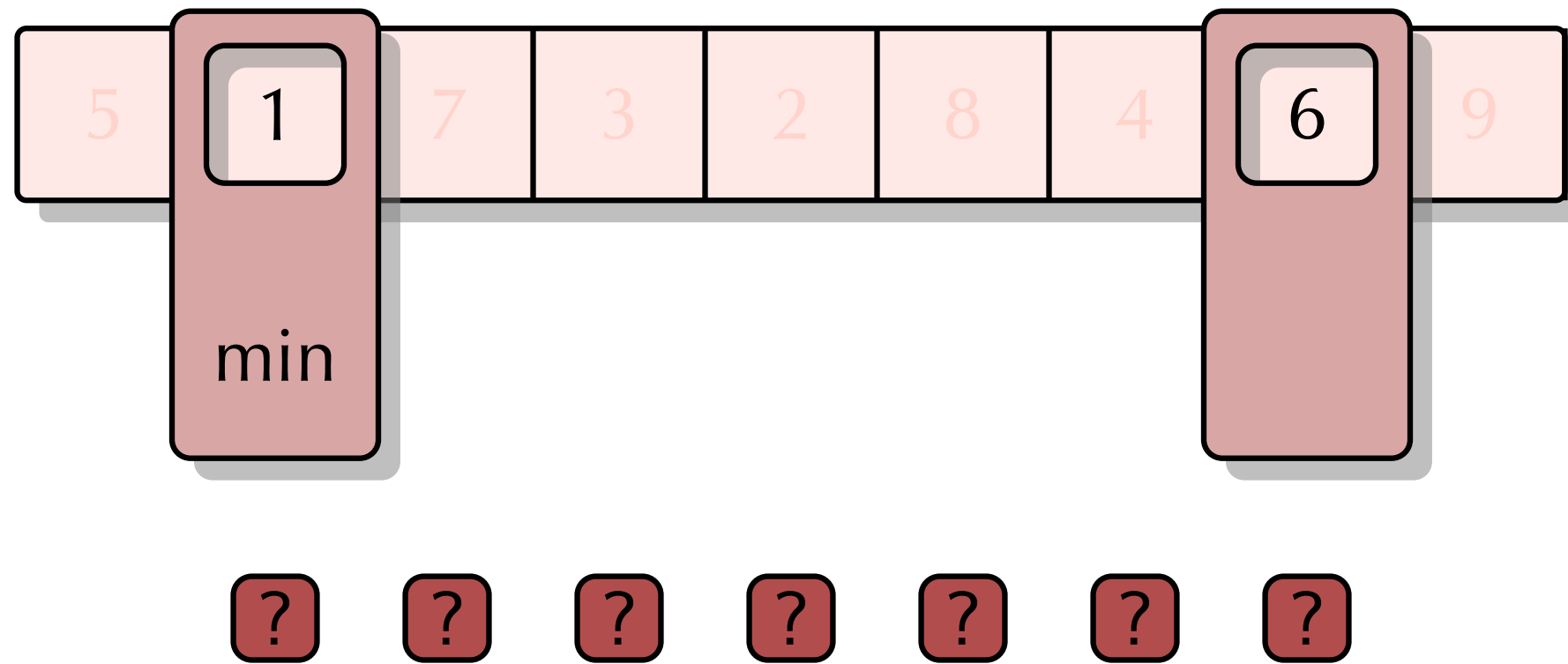
- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle



Zwei Sortialgorithmen

Selection Sort:

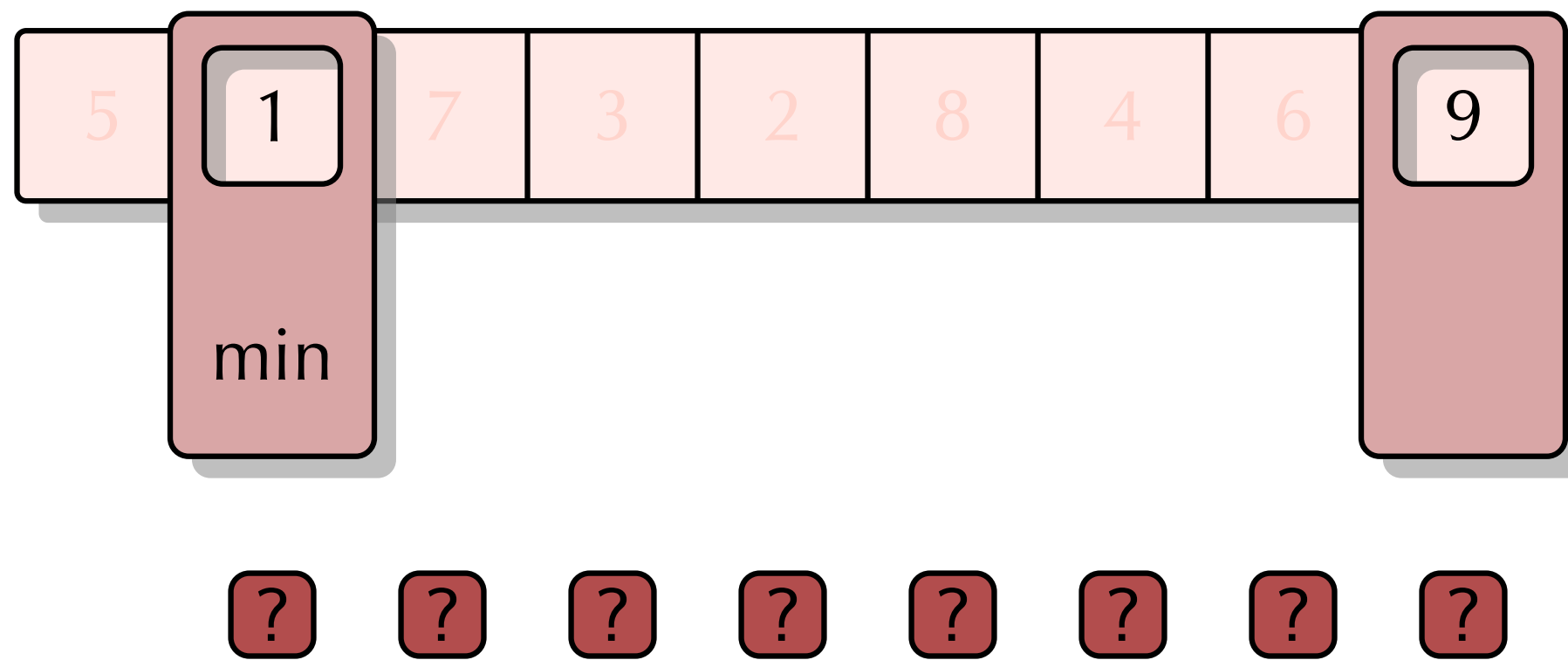
- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle



Zwei Sortialgorithmen

Selection Sort:

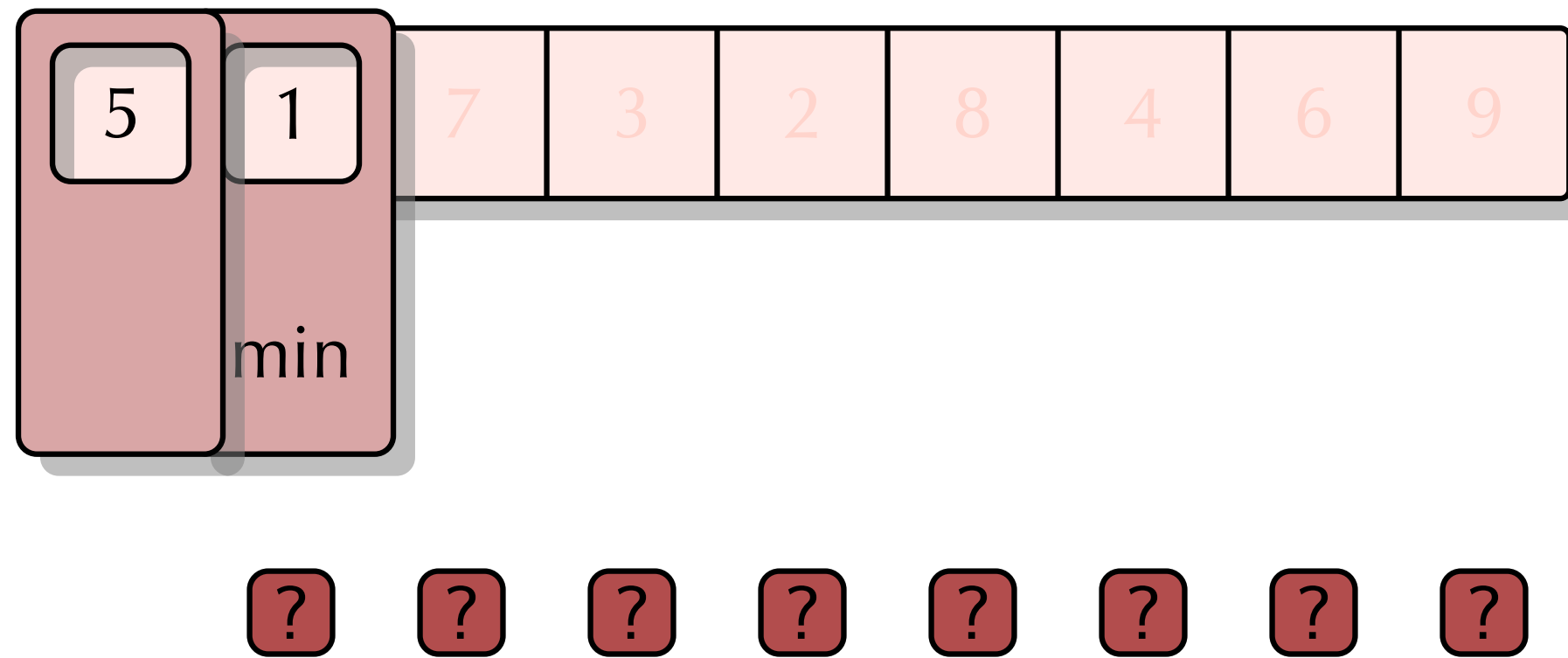
- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle



Zwei Sortialgorithmen

Selection Sort:

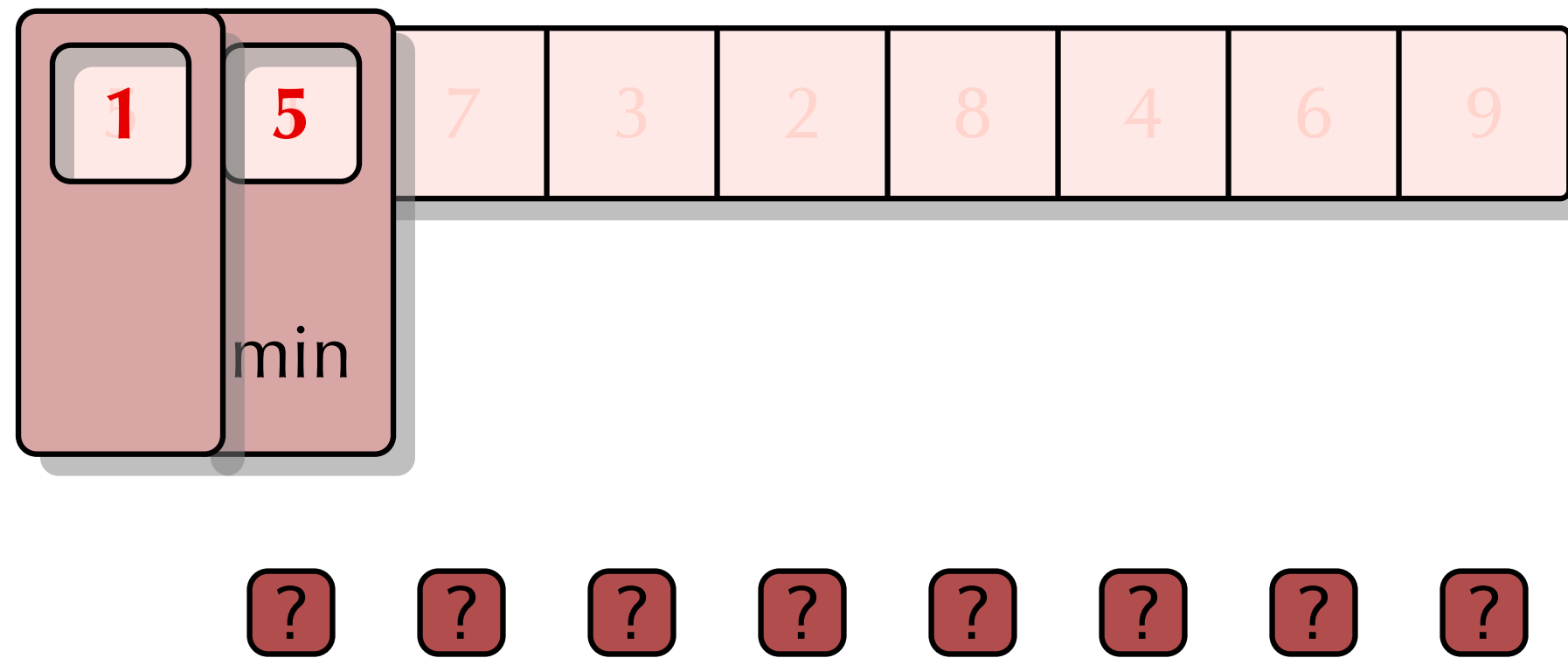
- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle



Zwei Sortialgorithmen

Selection Sort:

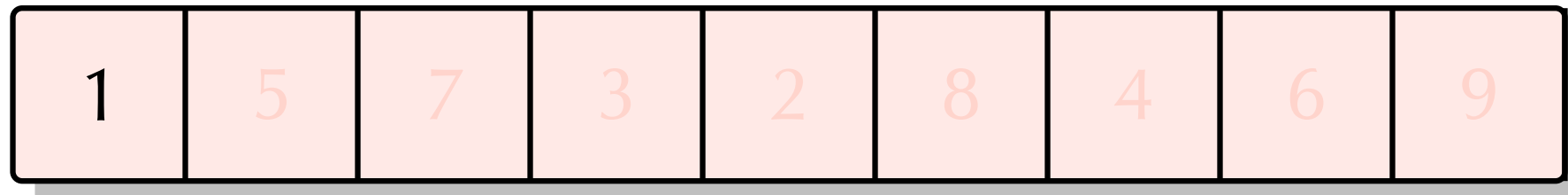
- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle



Zwei Sortialgorithmen

Selection Sort:

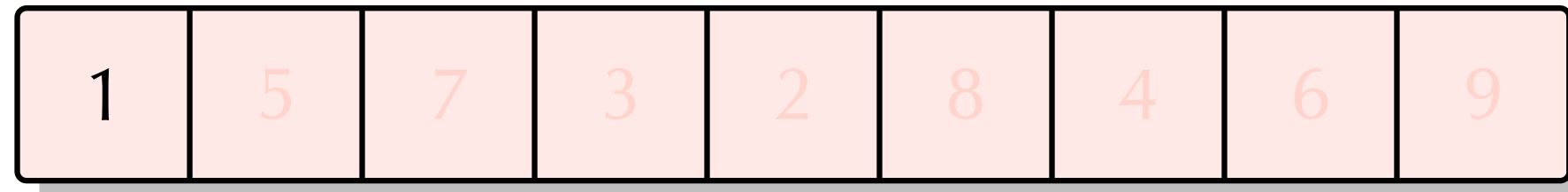
- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle



Zwei Sortialgorithmen

Selection Sort:

- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle



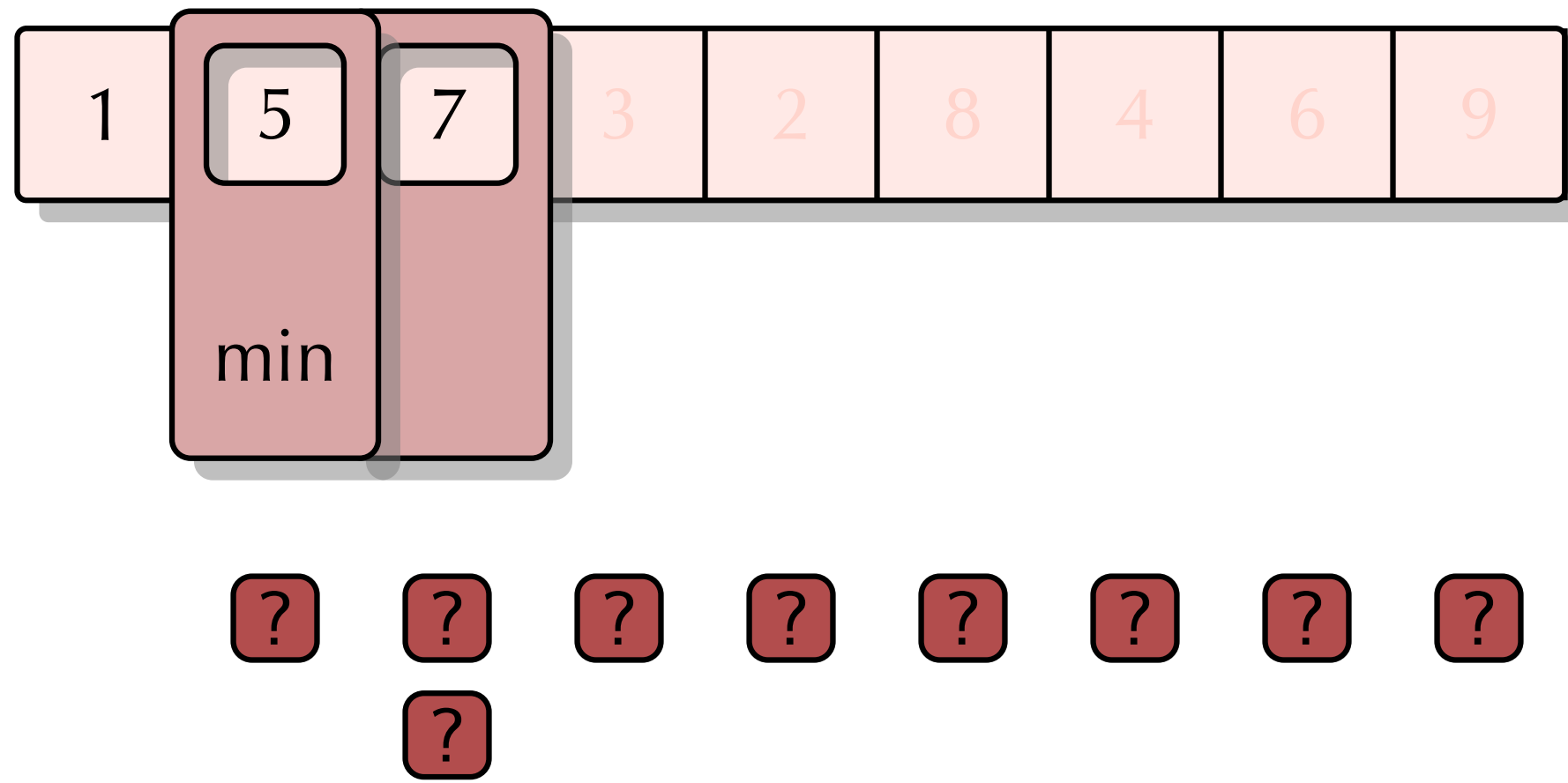
to do (by same procedure)



Zwei Sortialgorithmen

Selection Sort:

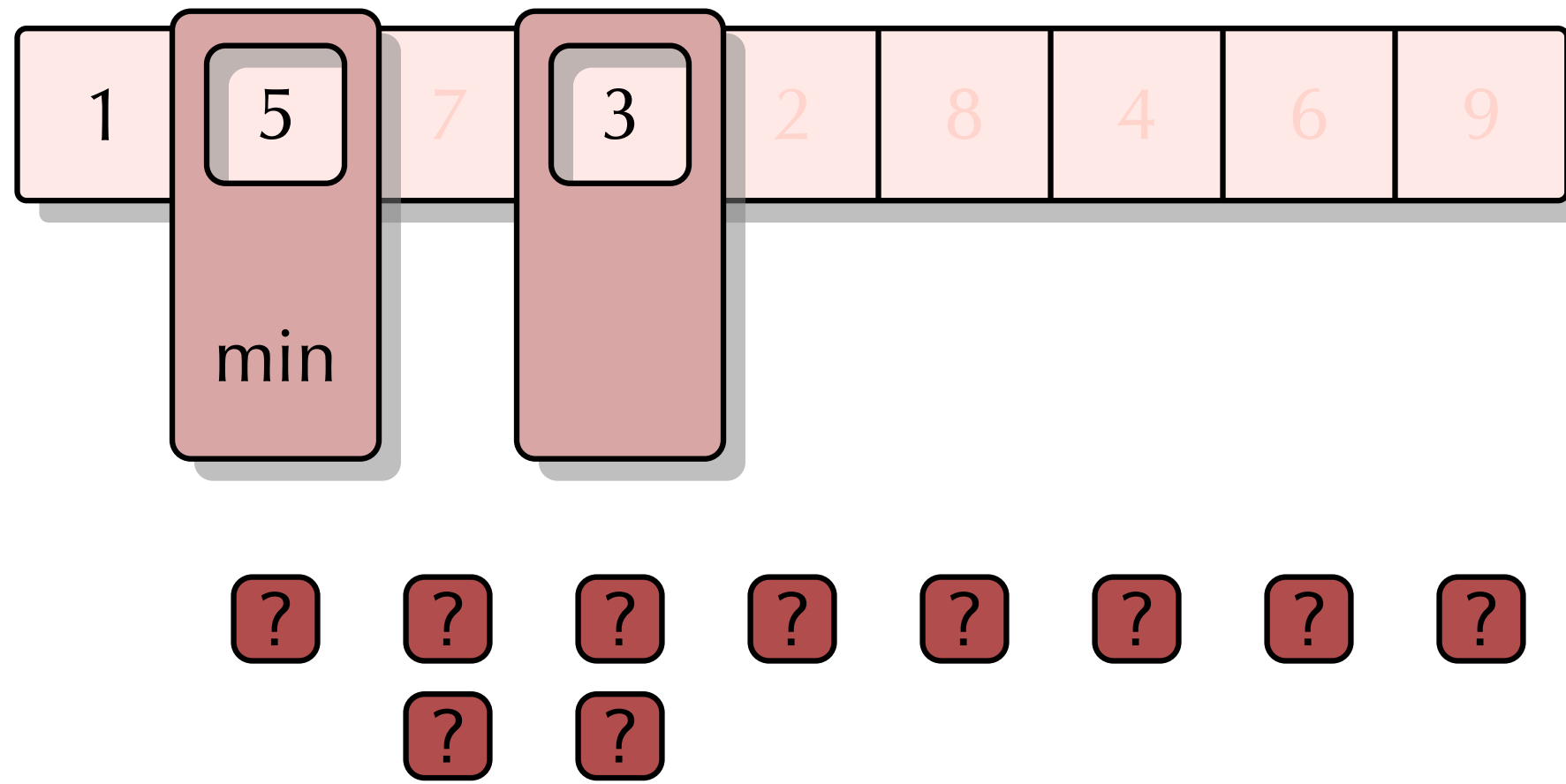
- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle



Zwei Sortialgorithmen

Selection Sort:

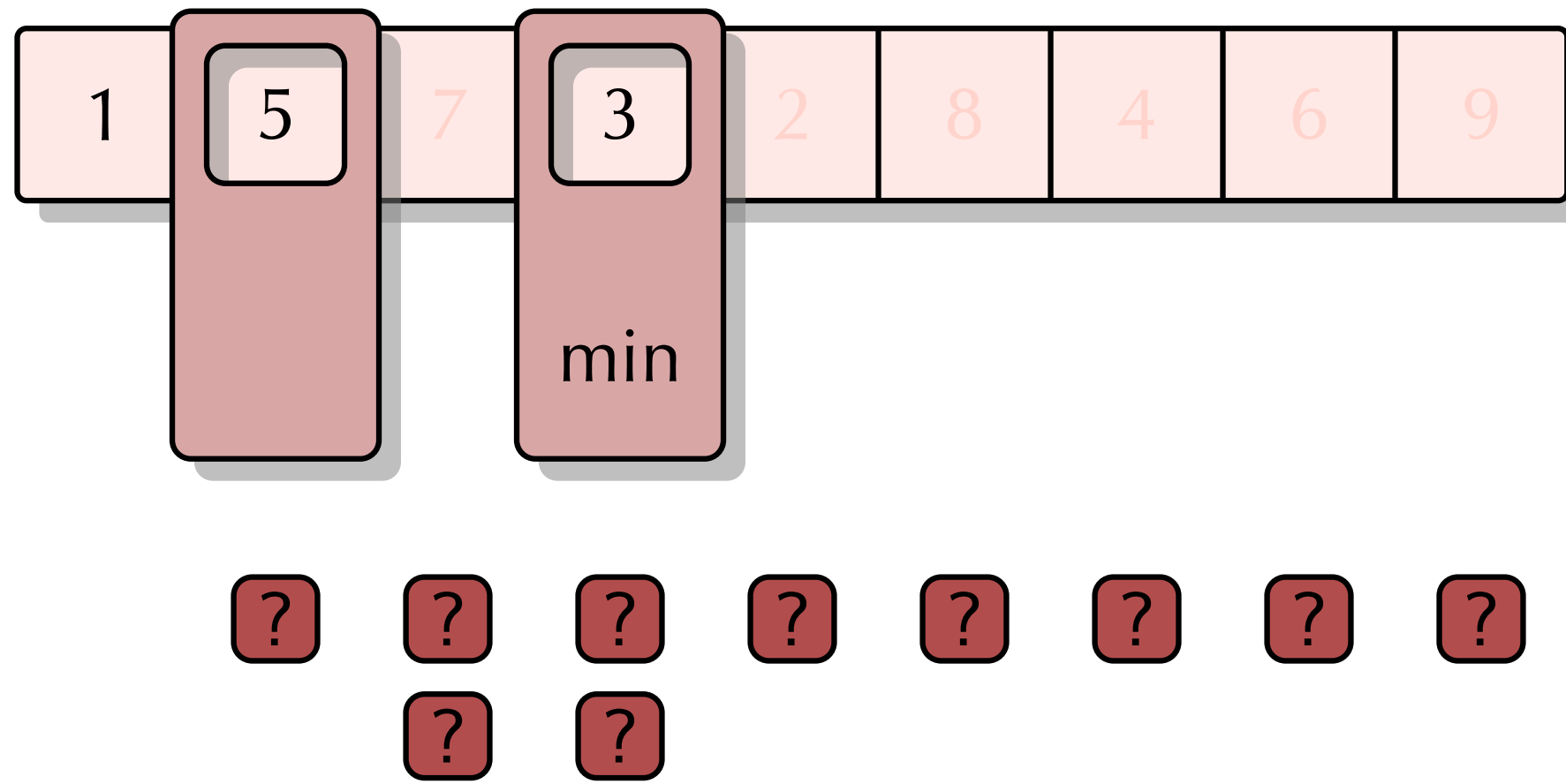
- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle



Zwei Sortialgorithmen

Selection Sort:

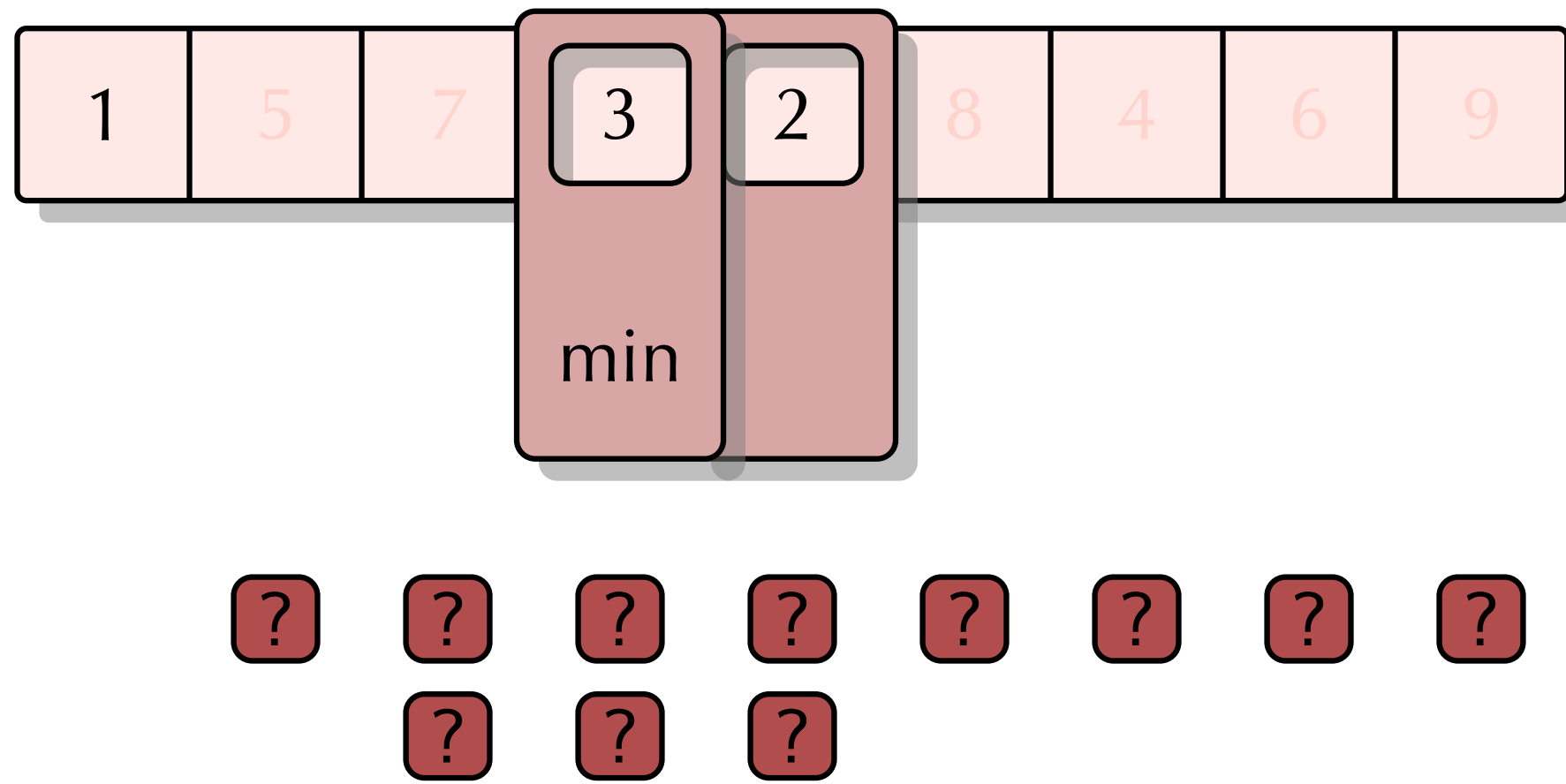
- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle



Zwei Sortialgorithmen

Selection Sort:

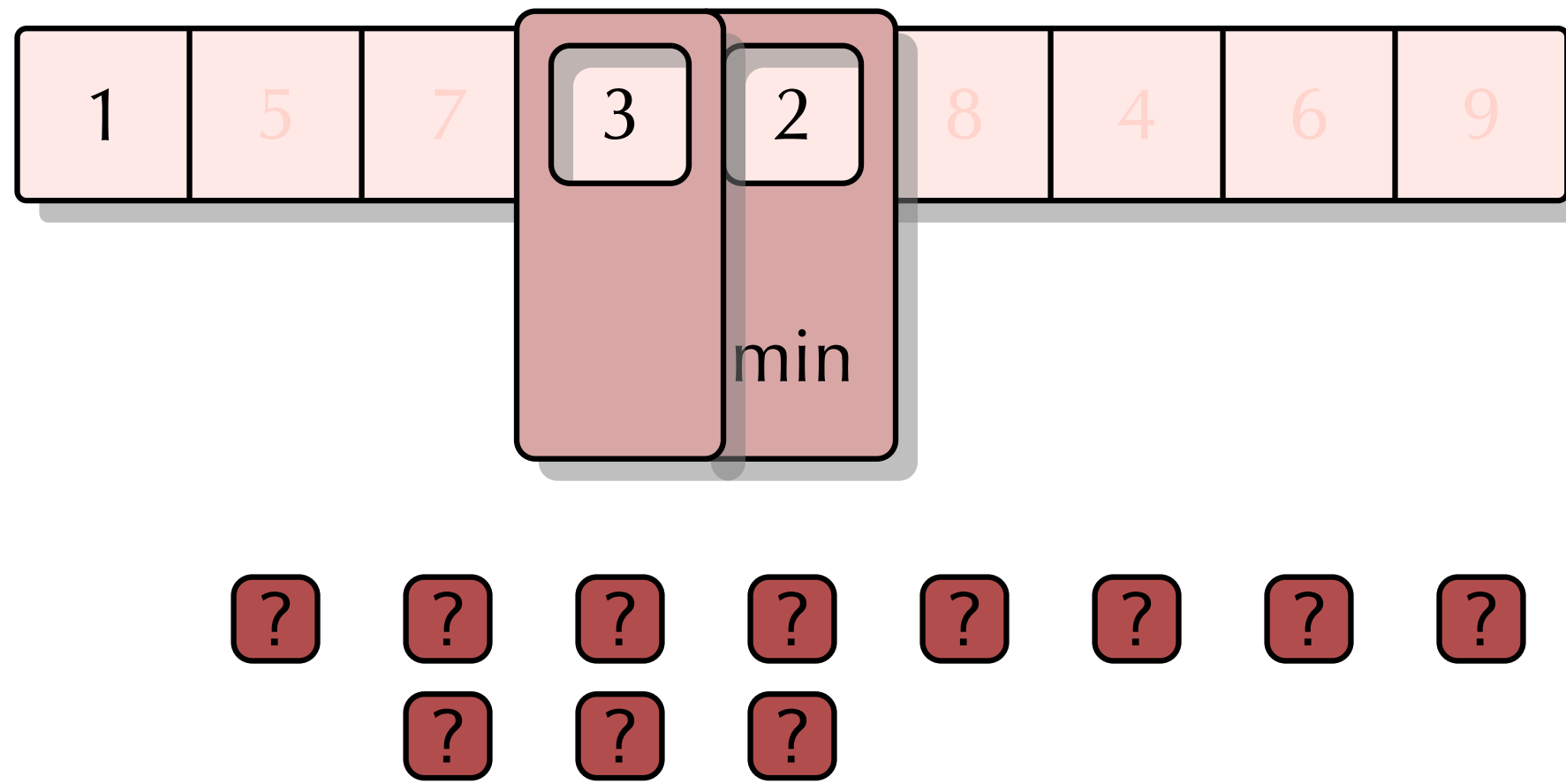
- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle



Zwei Sortialgorithmen

Selection Sort:

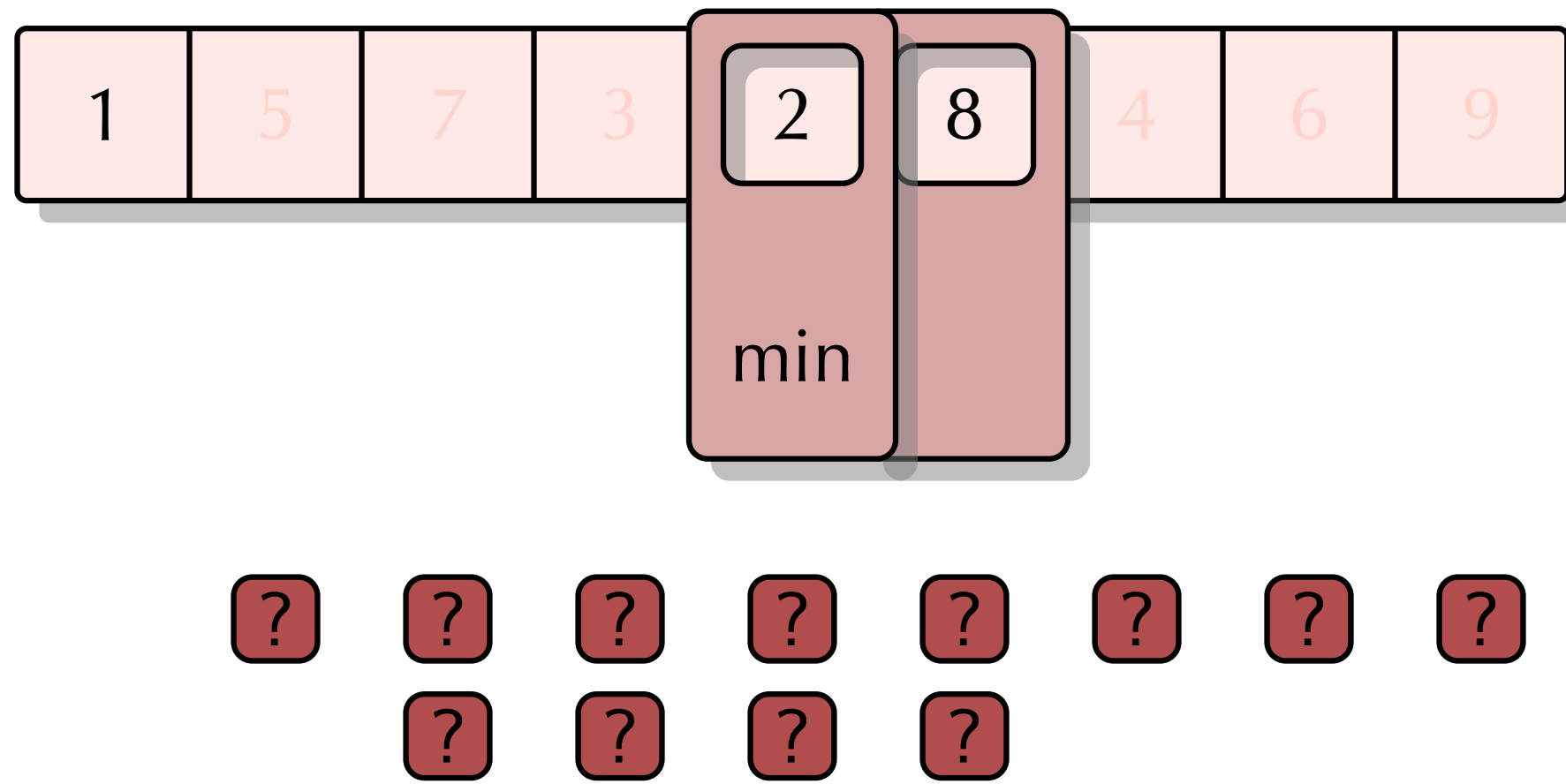
- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle



Zwei Sortialgorithmen

Selection Sort:

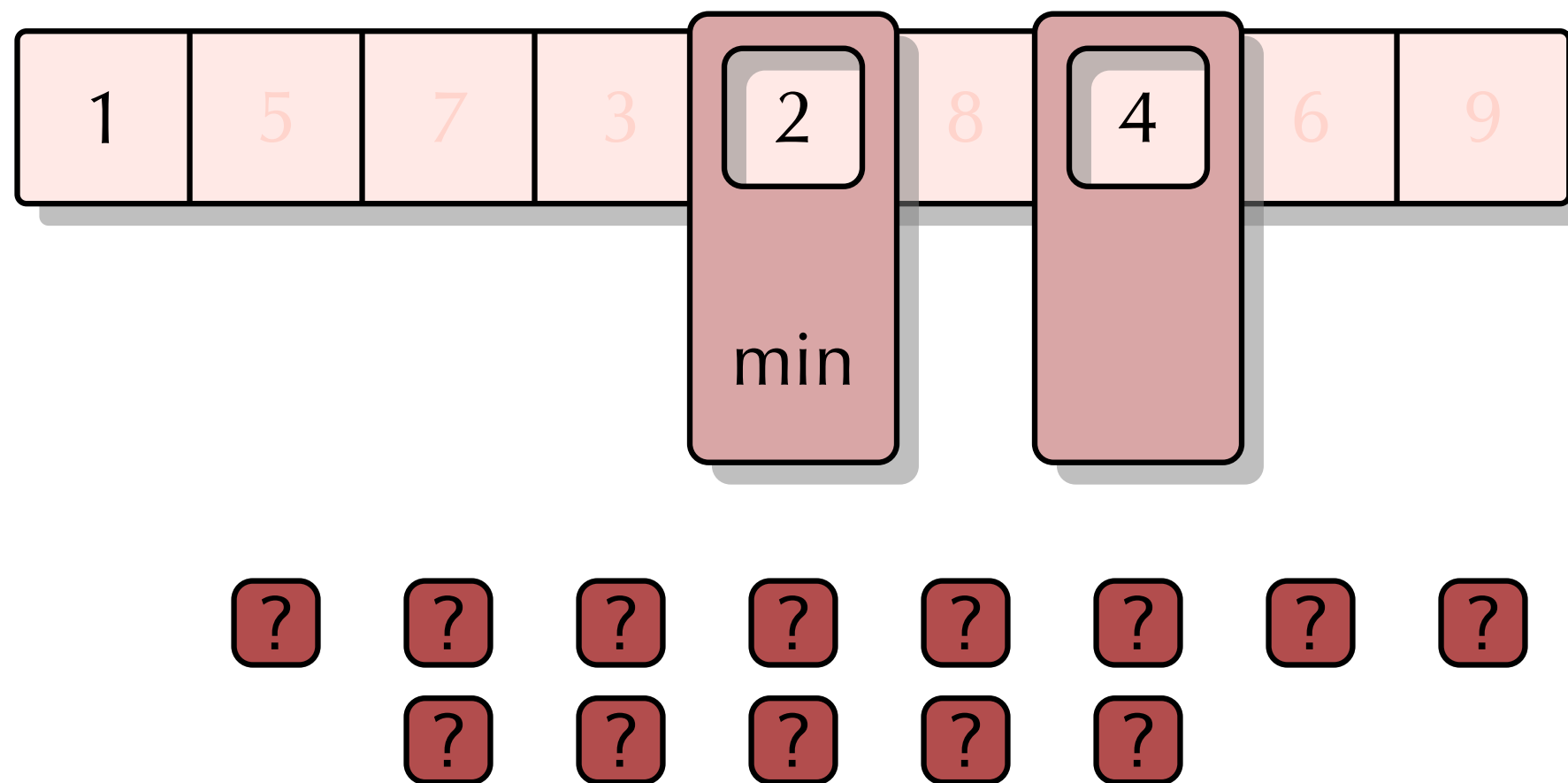
- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle



Zwei Sortialgorithmen

Selection Sort:

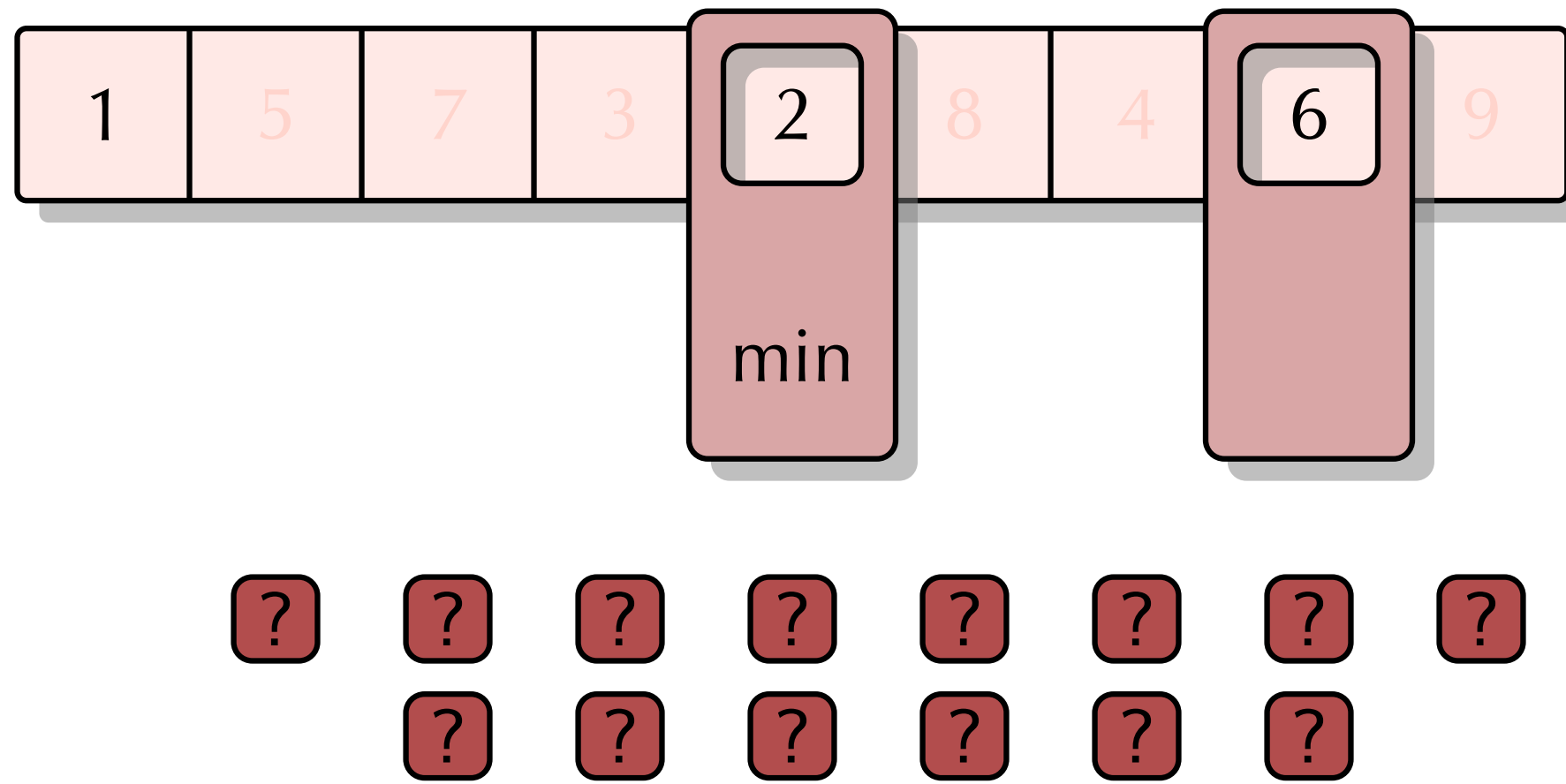
- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle



Zwei Sortialgorithmen

Selection Sort:

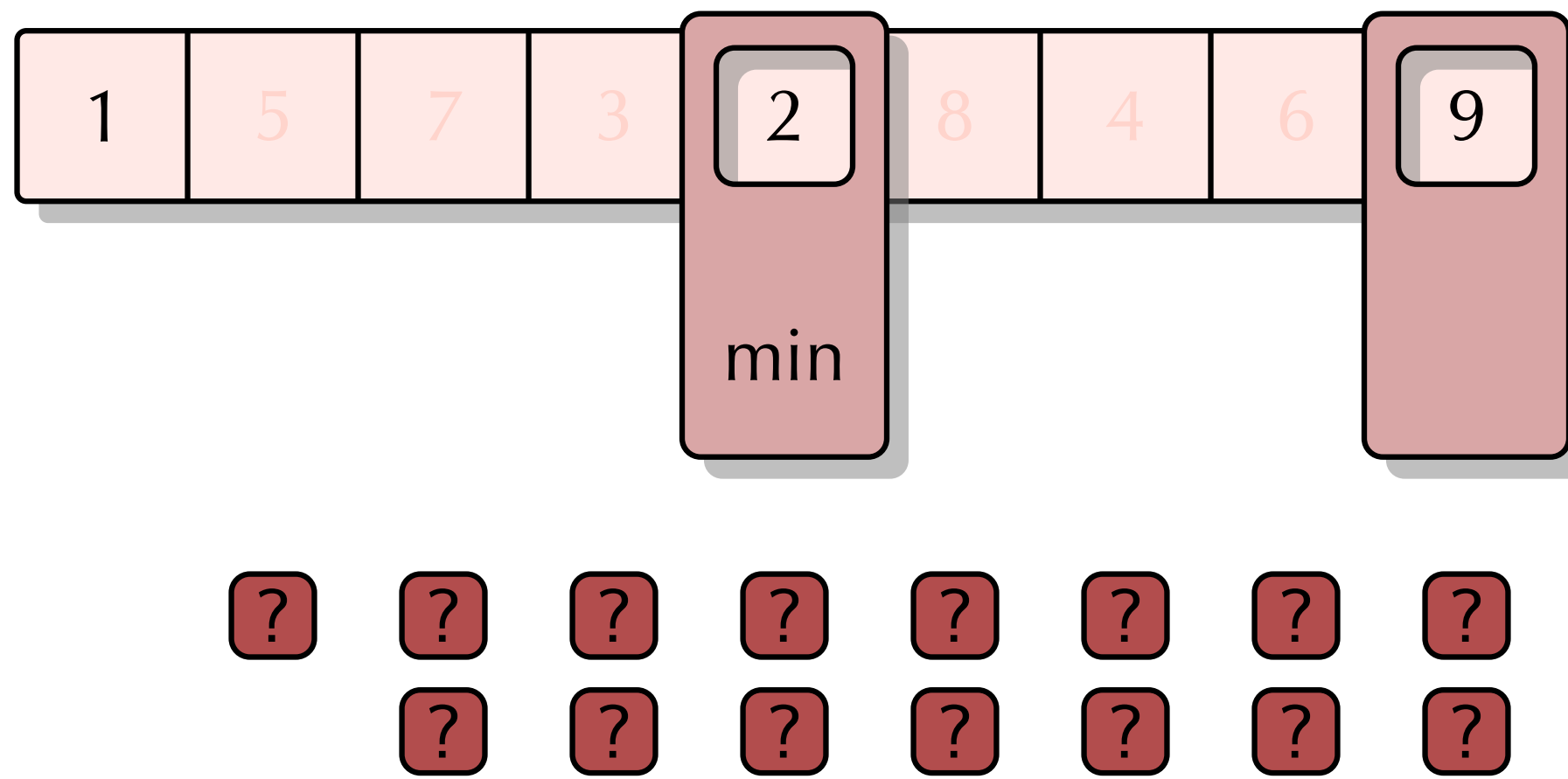
- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle



Zwei Sortialgorithmen

Selection Sort:

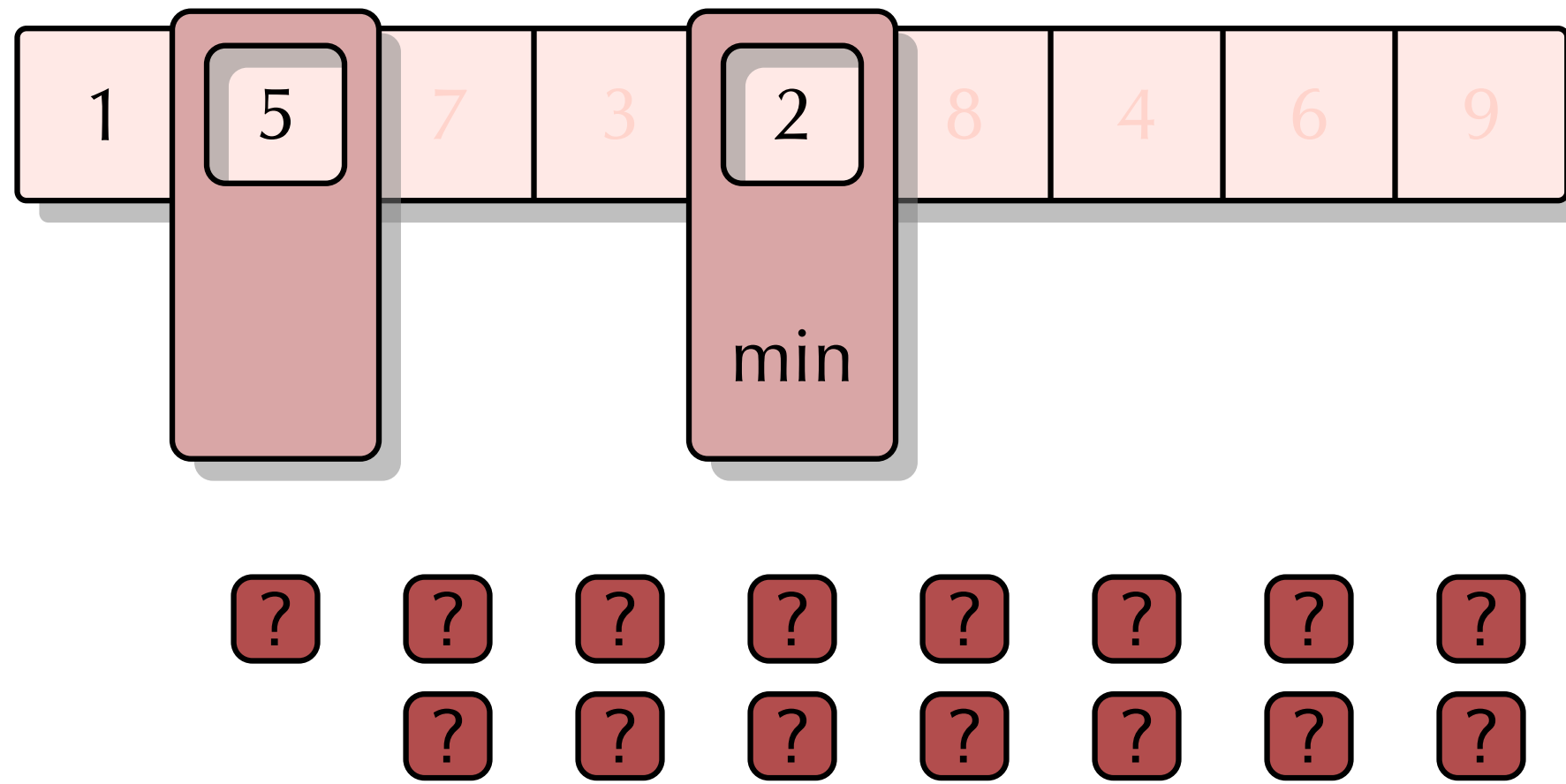
- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle



Zwei Sortialgorithmen

Selection Sort:

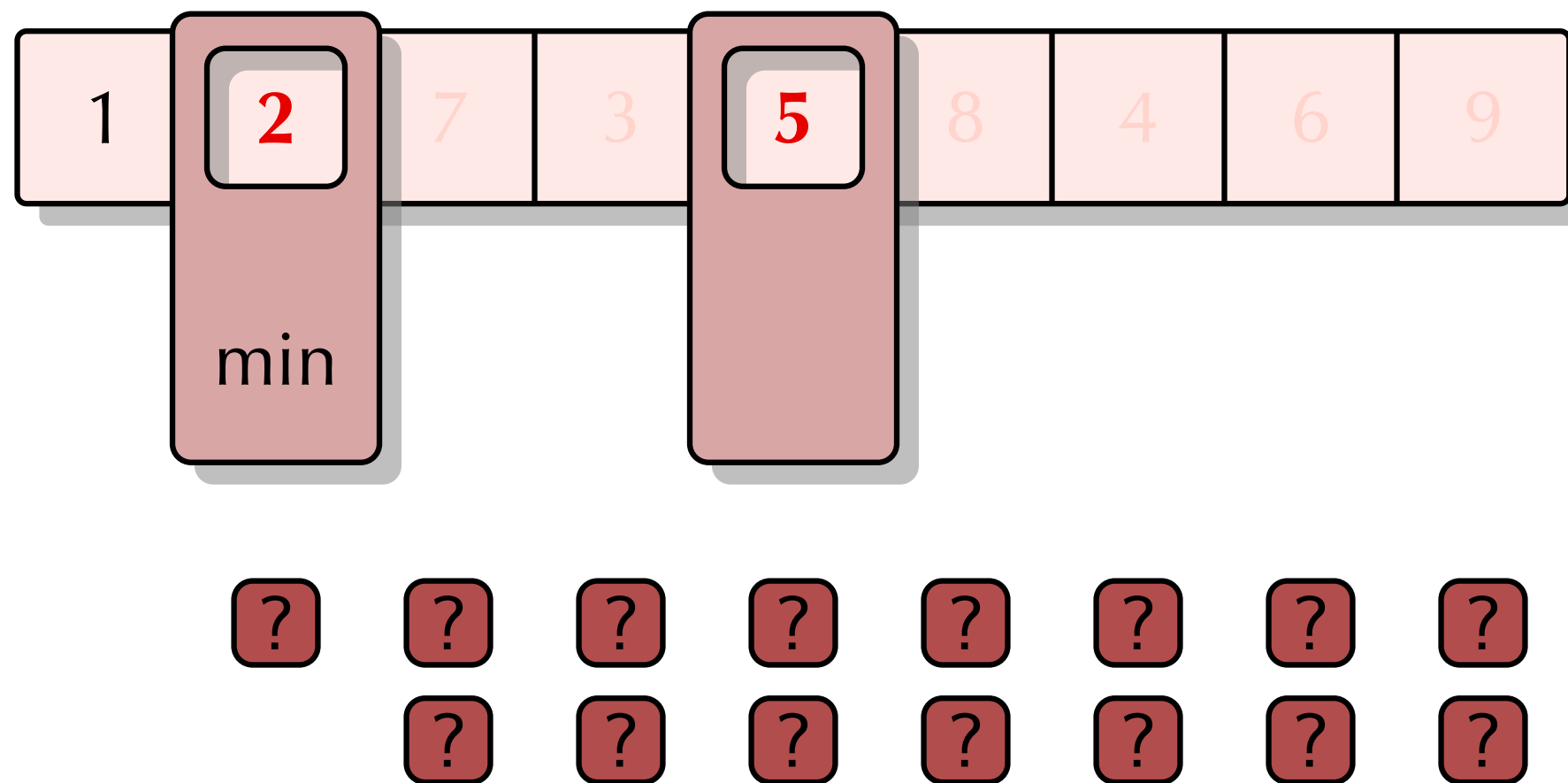
- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle



Zwei Sortialgorithmen

Selection Sort:

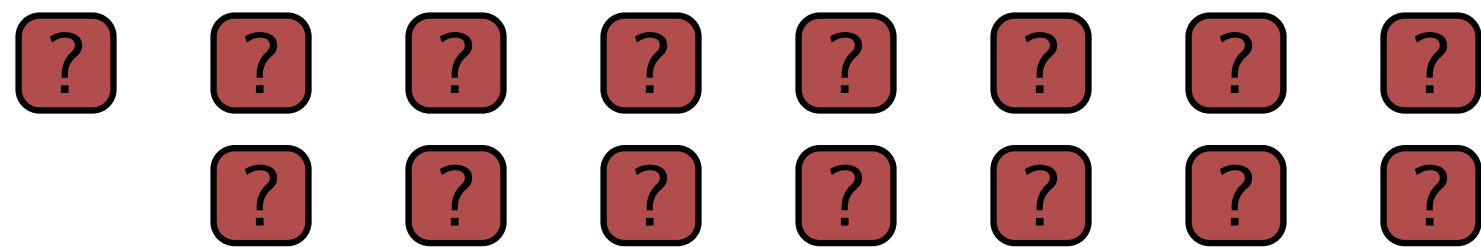
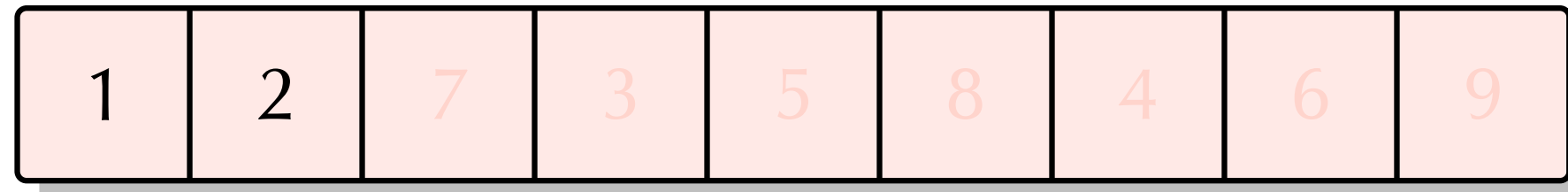
- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle



Zwei Sortialgorithmen

Selection Sort:

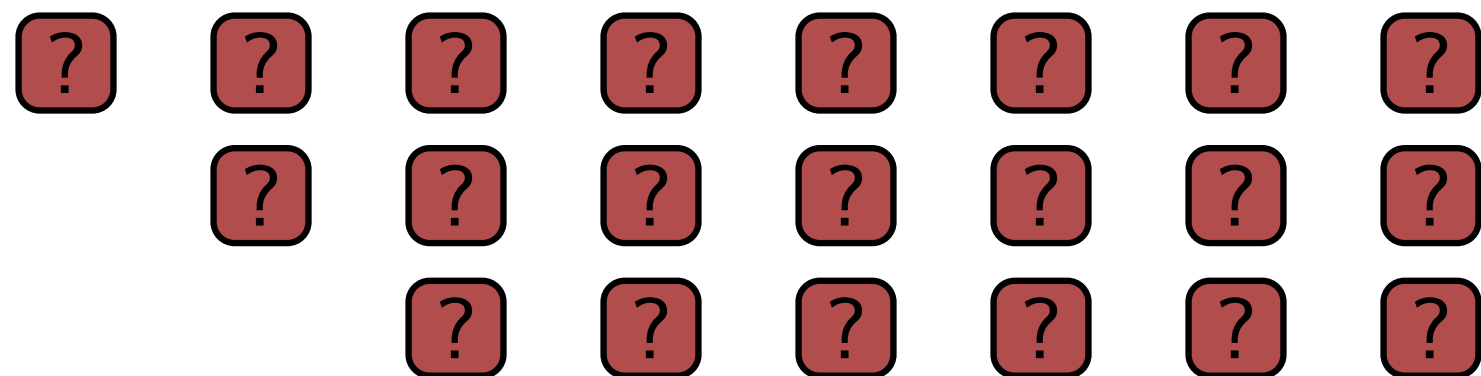
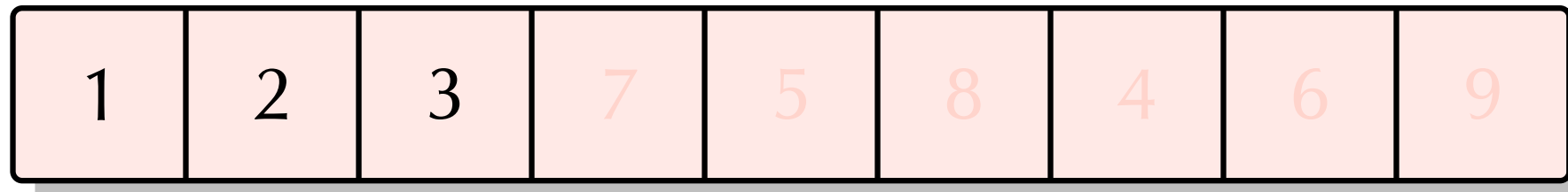
- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle



Zwei Sortialgorithmen

Selection Sort:

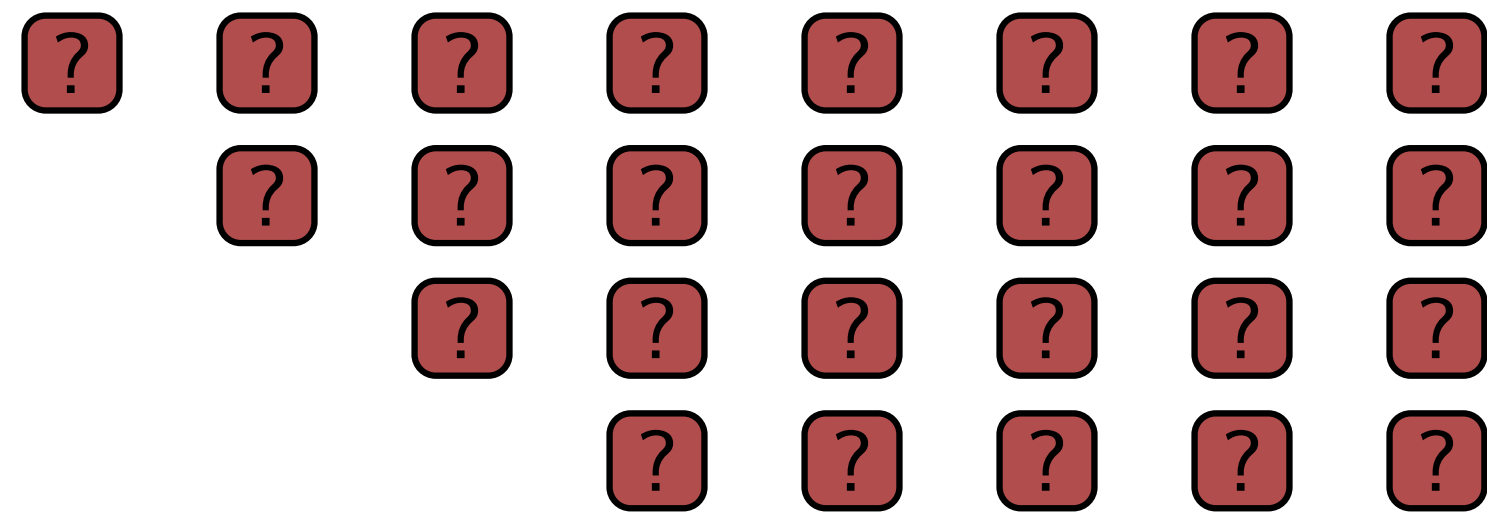
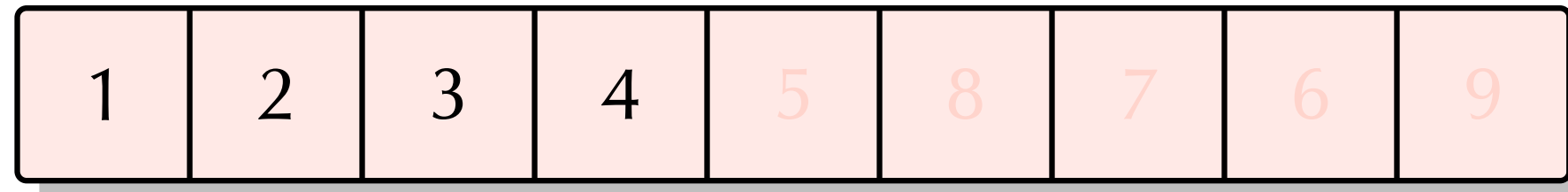
- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle



Zwei Sortialgorithmen

Selection Sort:

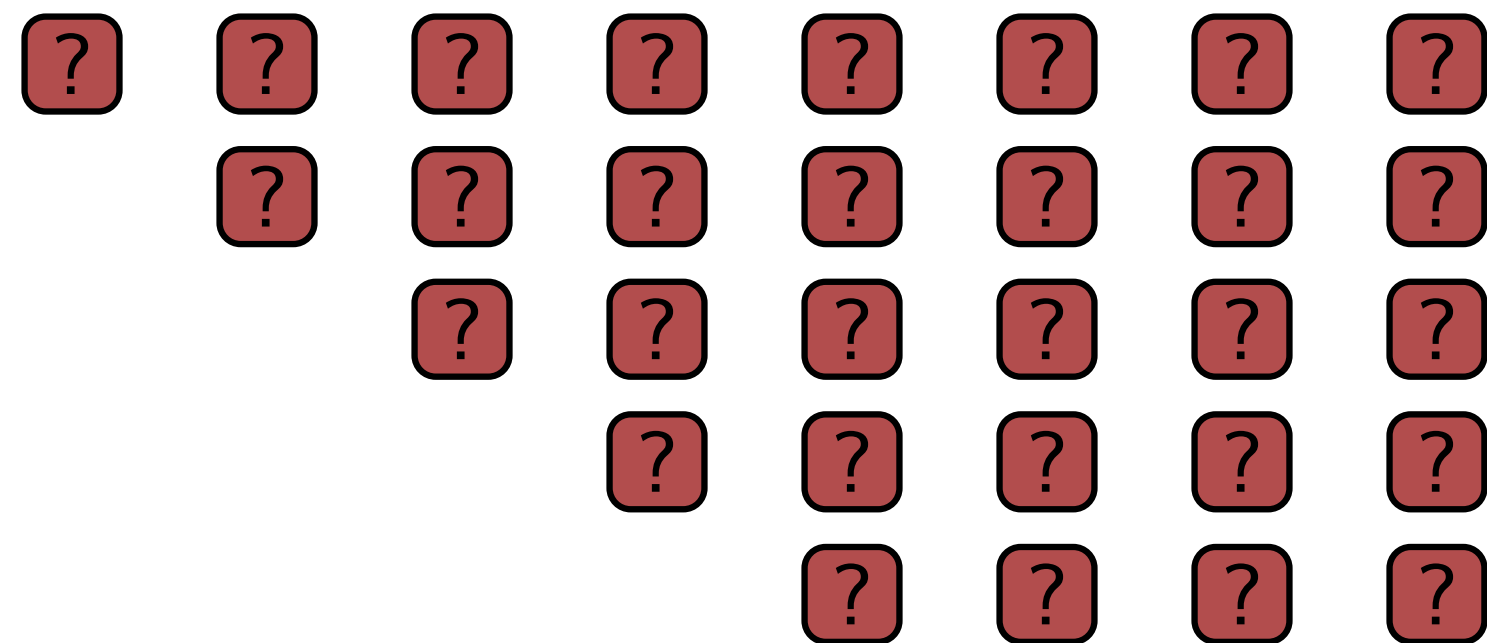
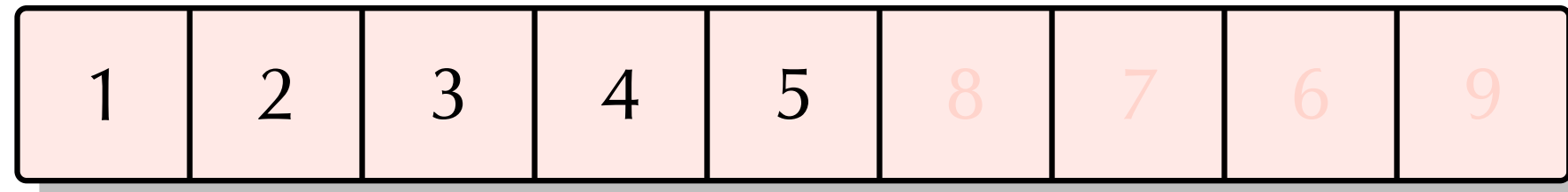
- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle



Zwei Sortialgorithmen

Selection Sort:

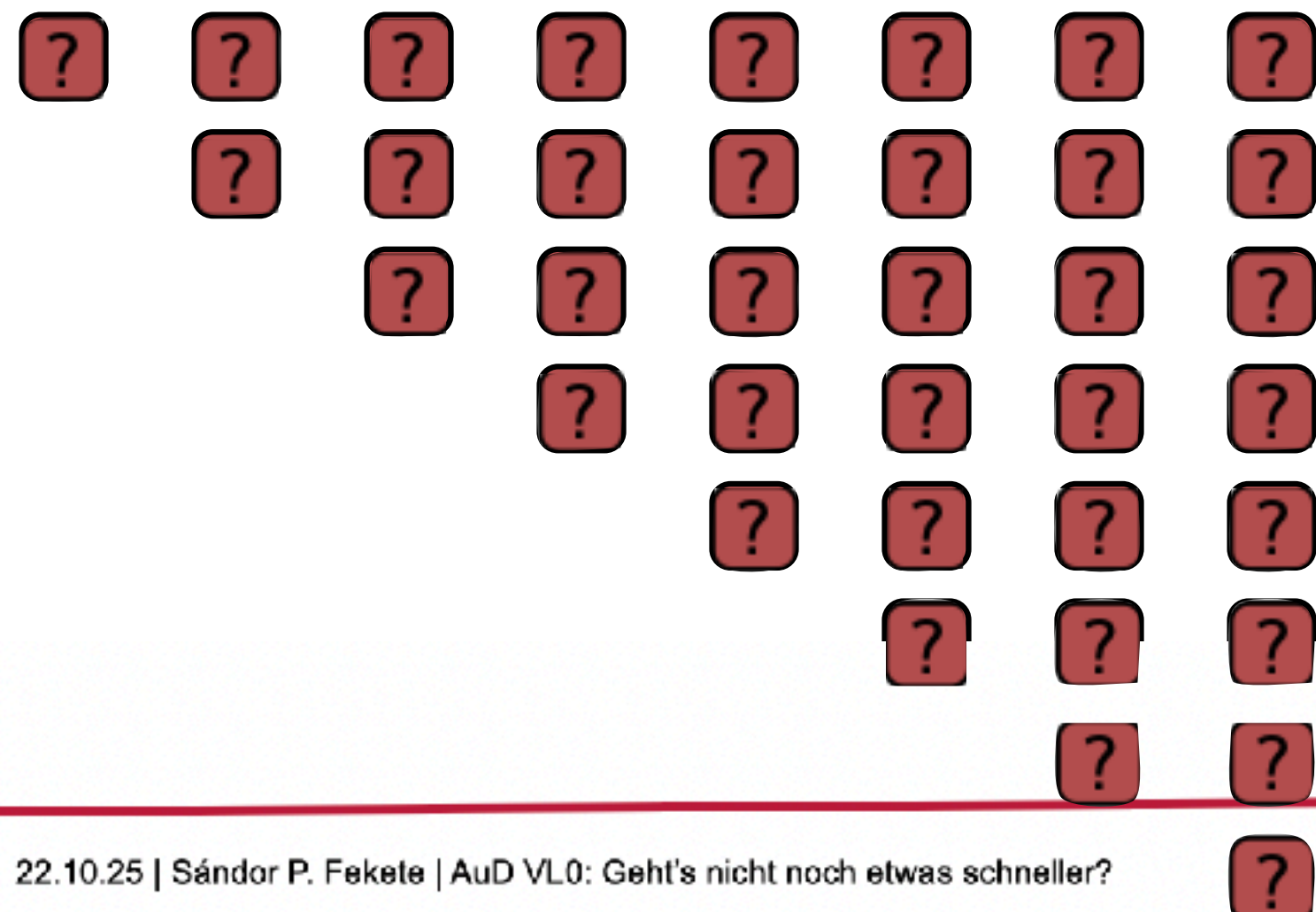
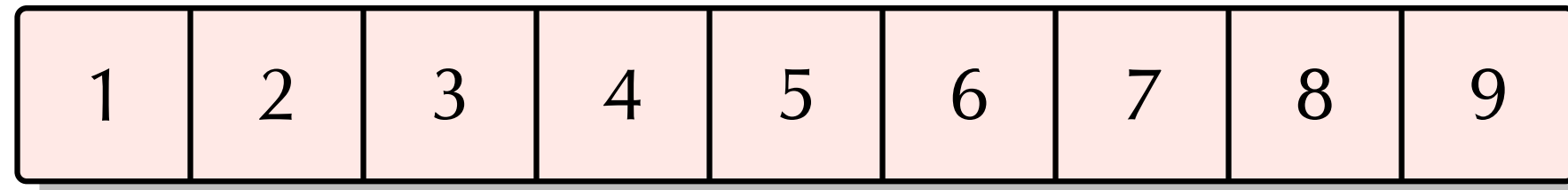
- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle



Zwei Sortialgorithmen

Selection Sort:

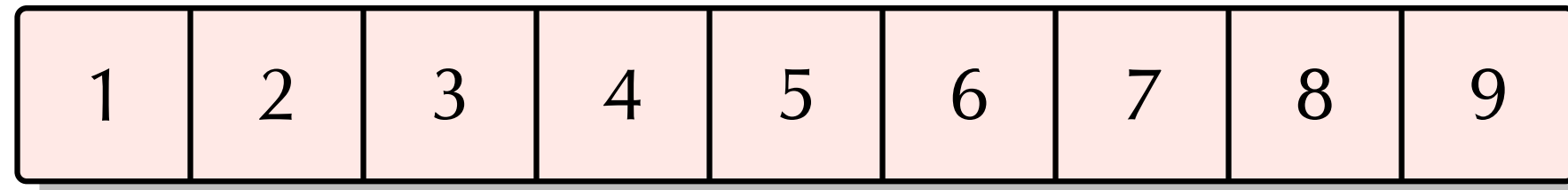
- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle



Zwei Sortialgorithmen

Selection Sort:

- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle



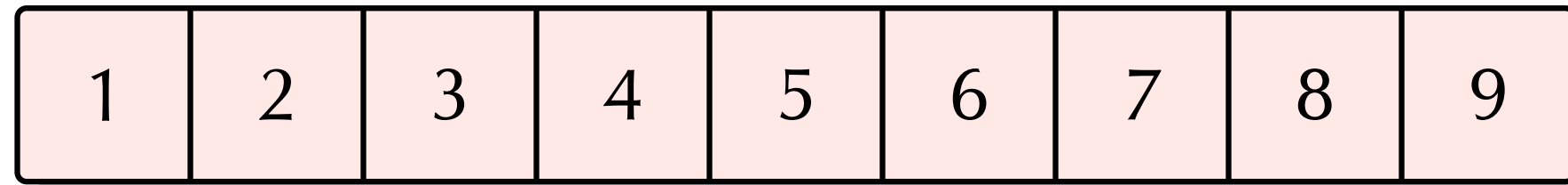
36 Vergleiche



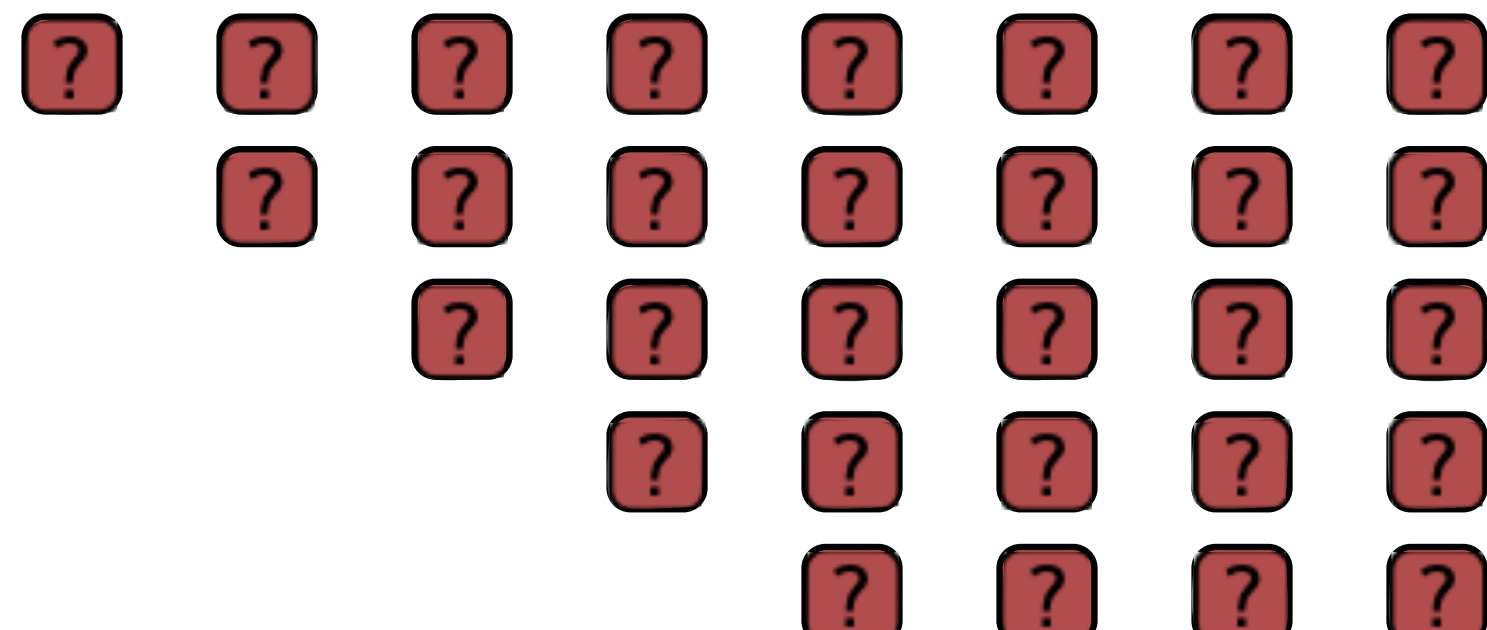
Zwei Sortialgorithmen

Selection Sort:

- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle



36 Vergleiche

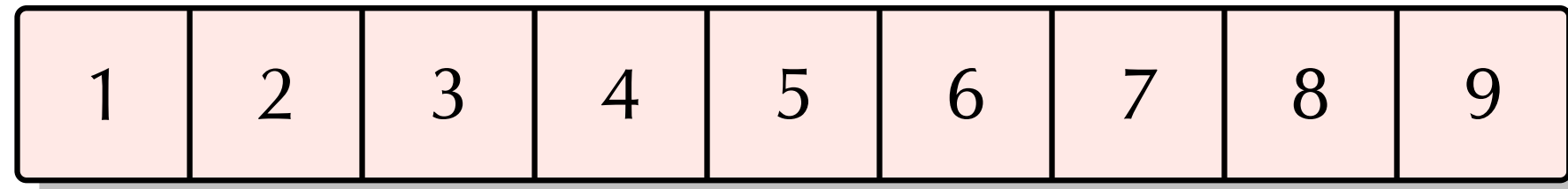


„Geht's nicht noch etwas schneller?“

Zwei Sortialgorithmen

Selection Sort:

- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle



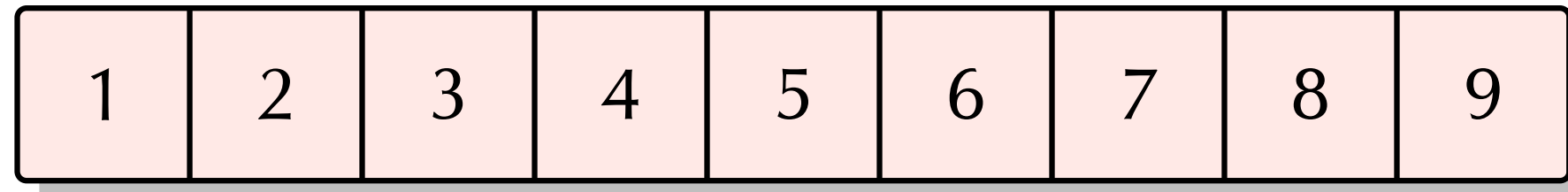
36 Vergleiche



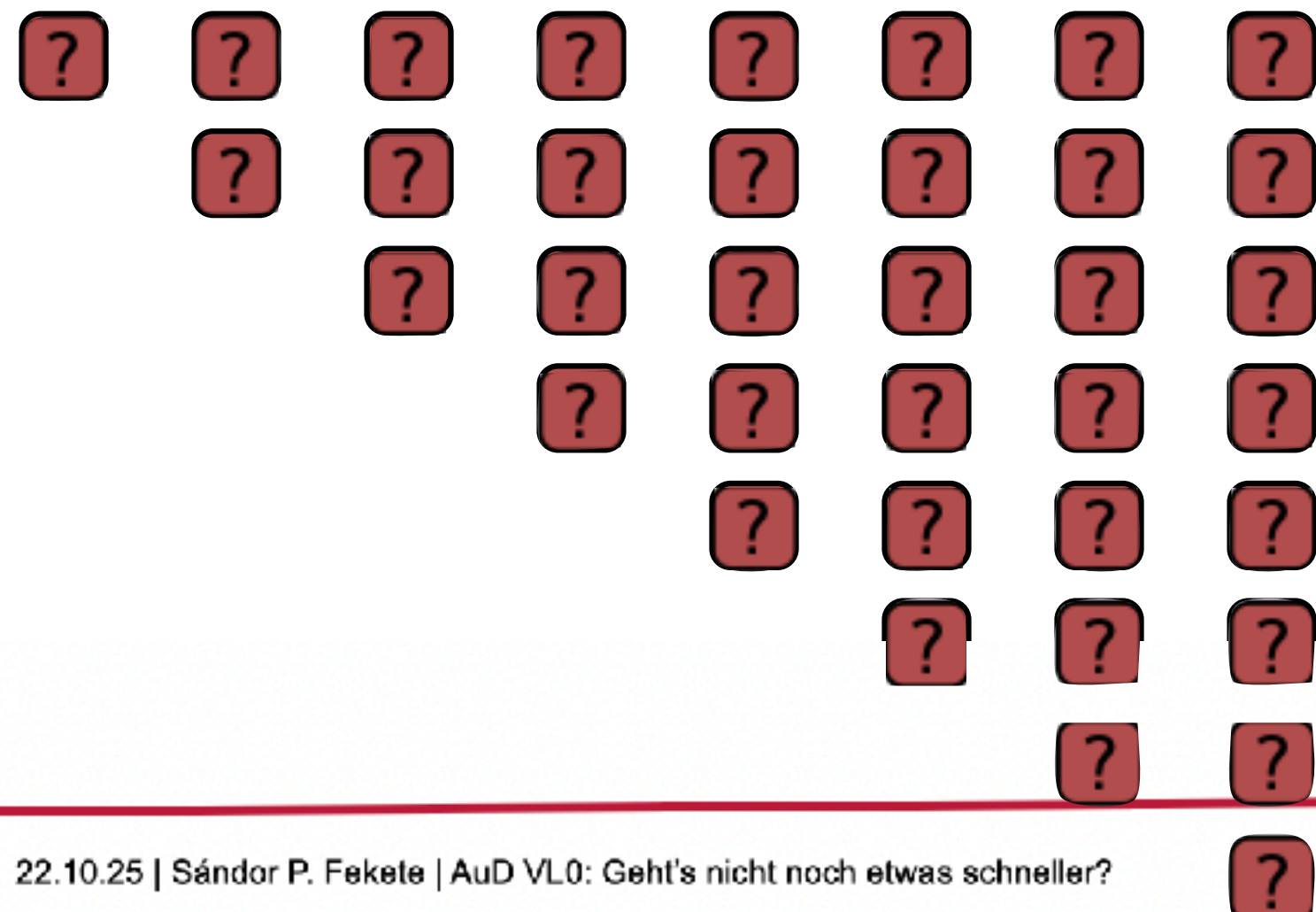
Zwei Sortialgorithmen

Selection Sort:

- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle



36 Vergleiche



Quicksort:

- Bringe ein Objekt an die richtige Stelle
- Bringe die anderen nach links und rechts

Zwei Sortialgorithmen

Selection Sort:

- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle

1	2	3	4	5	6	7	8	9
---	---	---	---	---	---	---	---	---

36 Vergleiche



Quicksort:

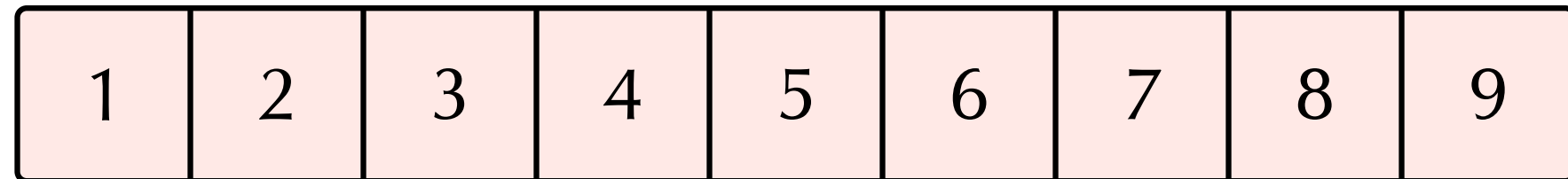
- Bringe ein Objekt an die richtige Stelle
- Bringe die anderen nach links und rechts

5	1	7	3	2	8	4	6	9
---	---	---	---	---	---	---	---	---

Zwei Sortialgorithmen

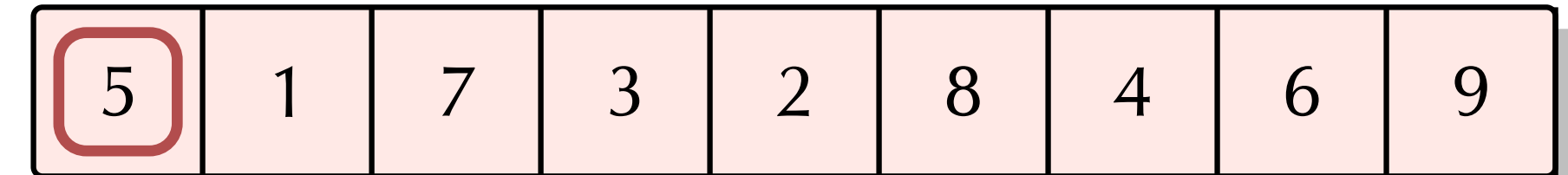
Selection Sort:

- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle

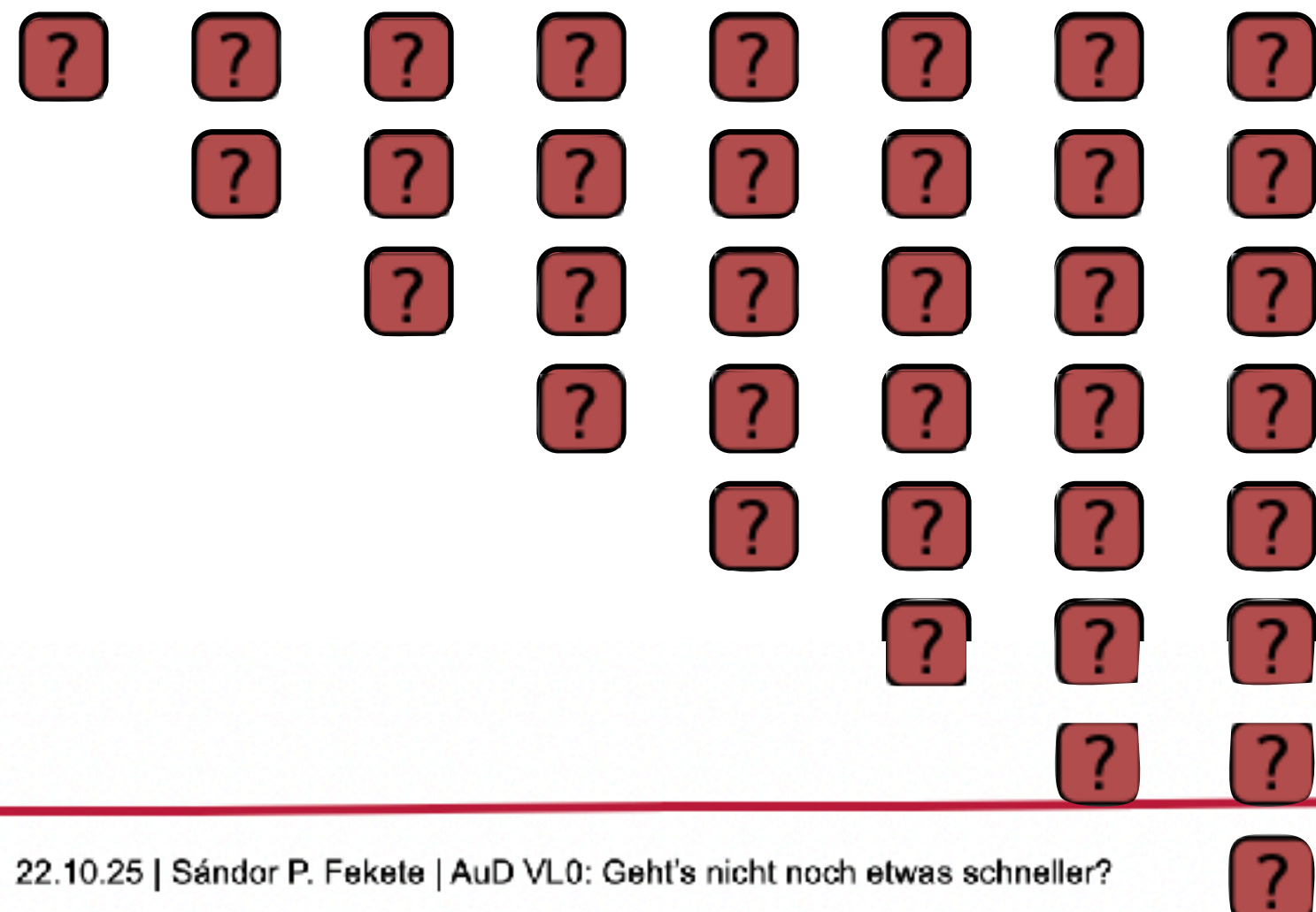


Quicksort:

- Bringe ein Objekt an die richtige Stelle
- Bringe die anderen nach links und rechts



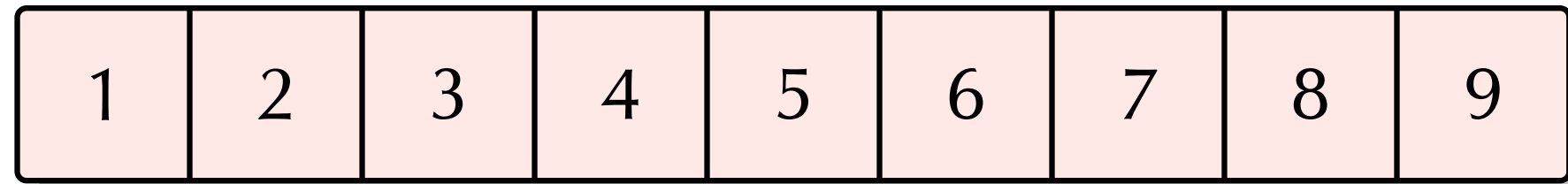
36 Vergleiche



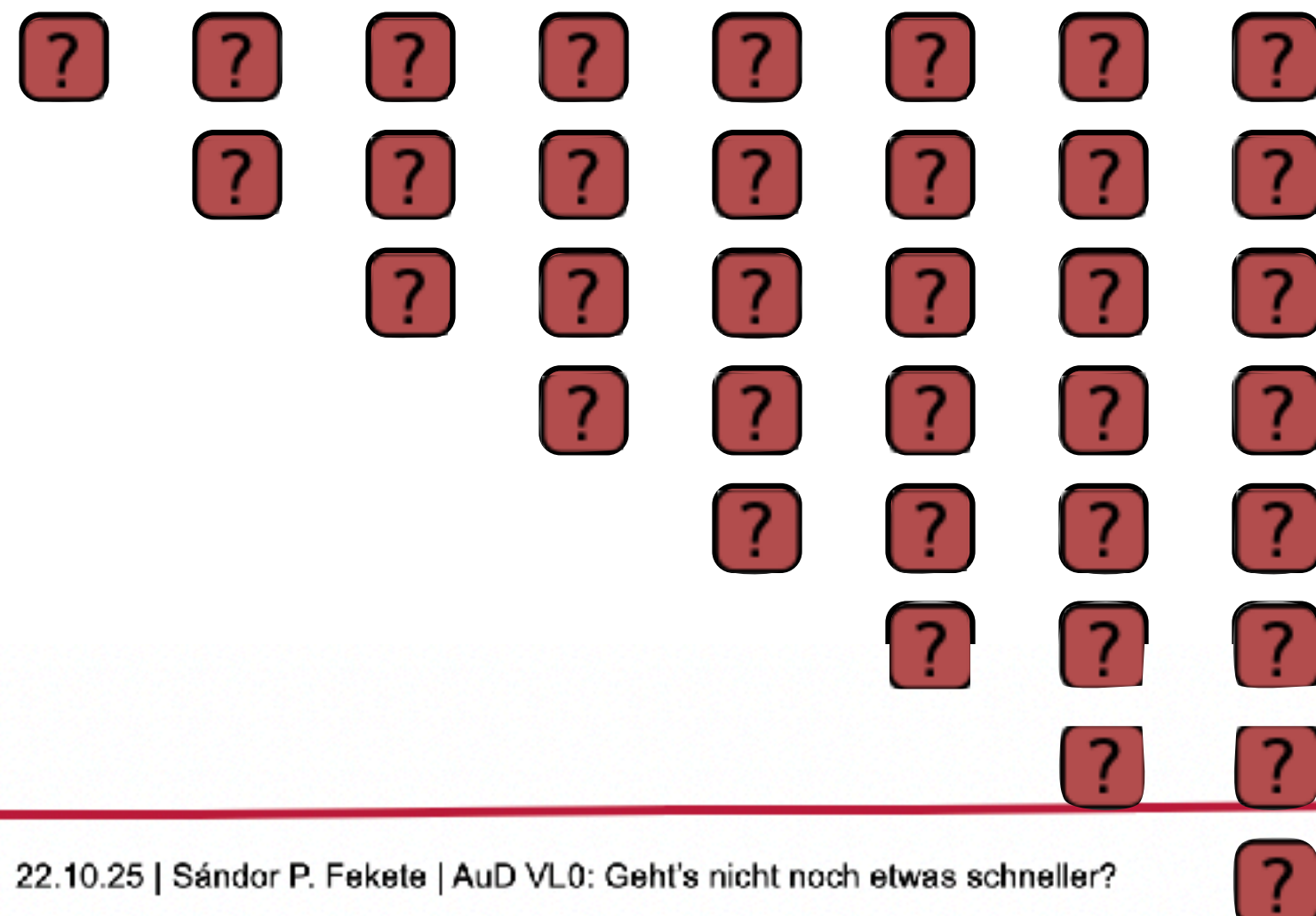
Zwei Sortialgorithmen

Selection Sort:

- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle

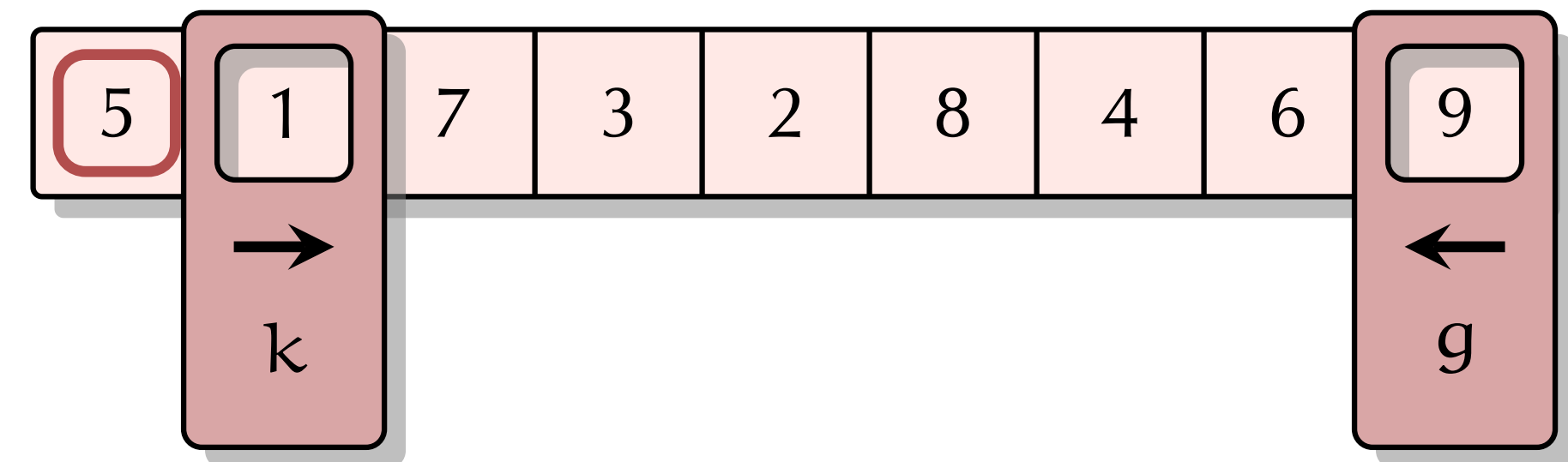


36 Vergleiche



Quicksort:

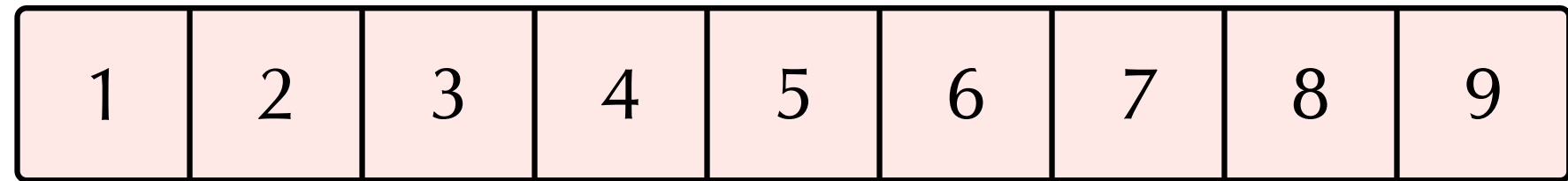
- Bringe ein Objekt an die richtige Stelle
- Bringe die anderen nach links und rechts



Zwei Sortialgorithmen

Selection Sort:

- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle

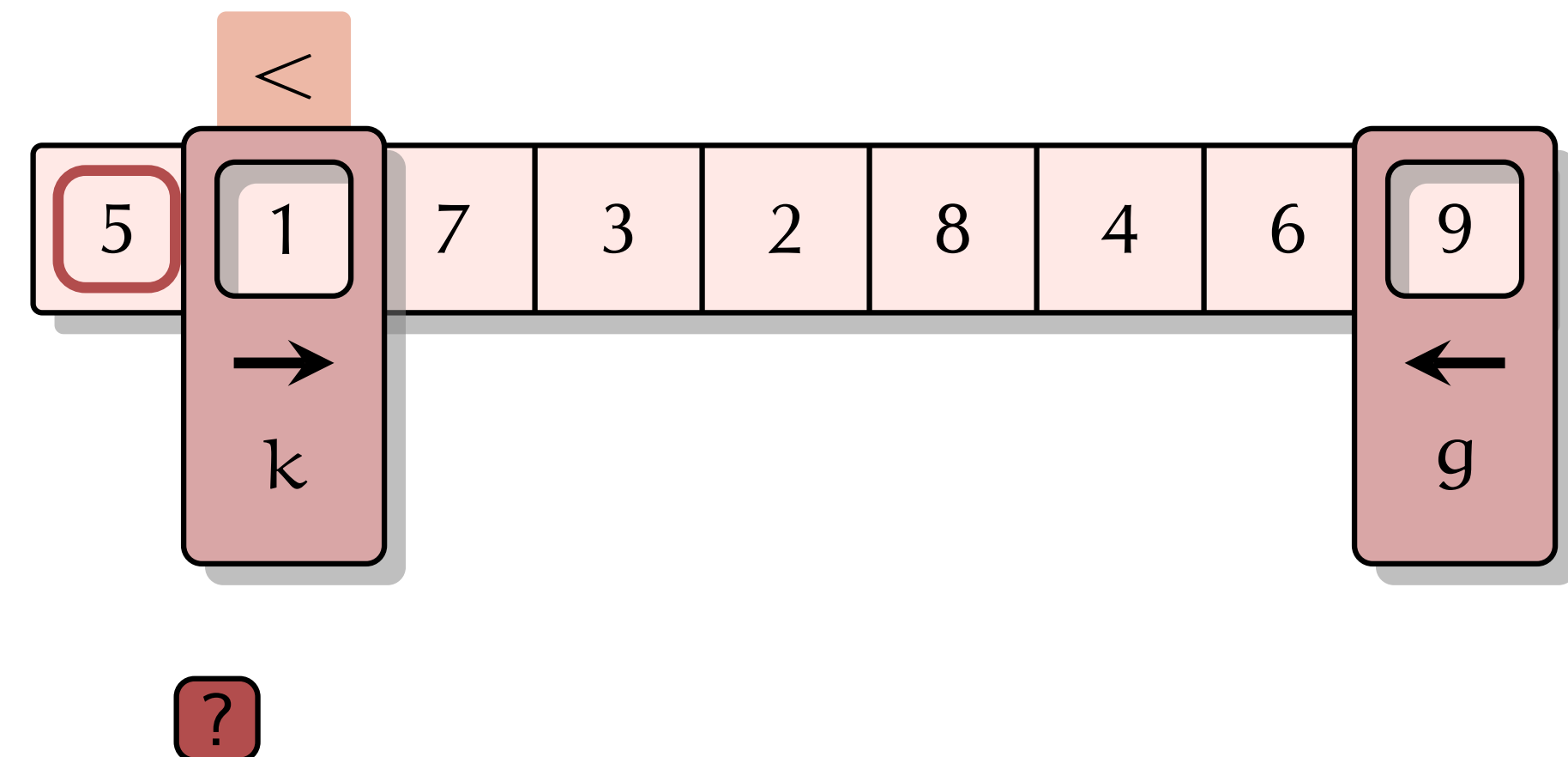


36 Vergleiche



Quicksort:

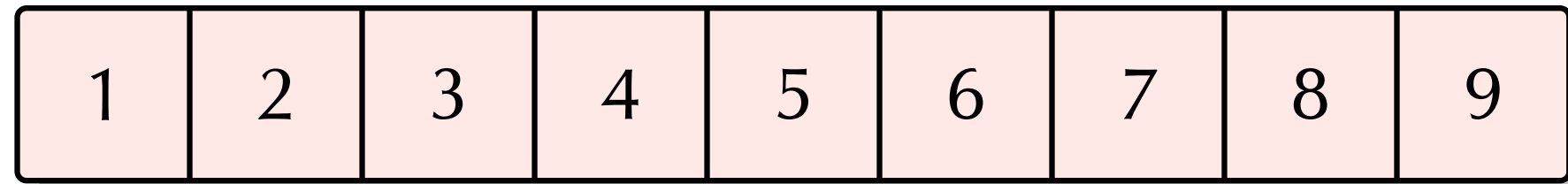
- Bringe ein Objekt an die richtige Stelle
- Bringe die anderen nach links und rechts



Zwei Sortialgorithmen

Selection Sort:

- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle

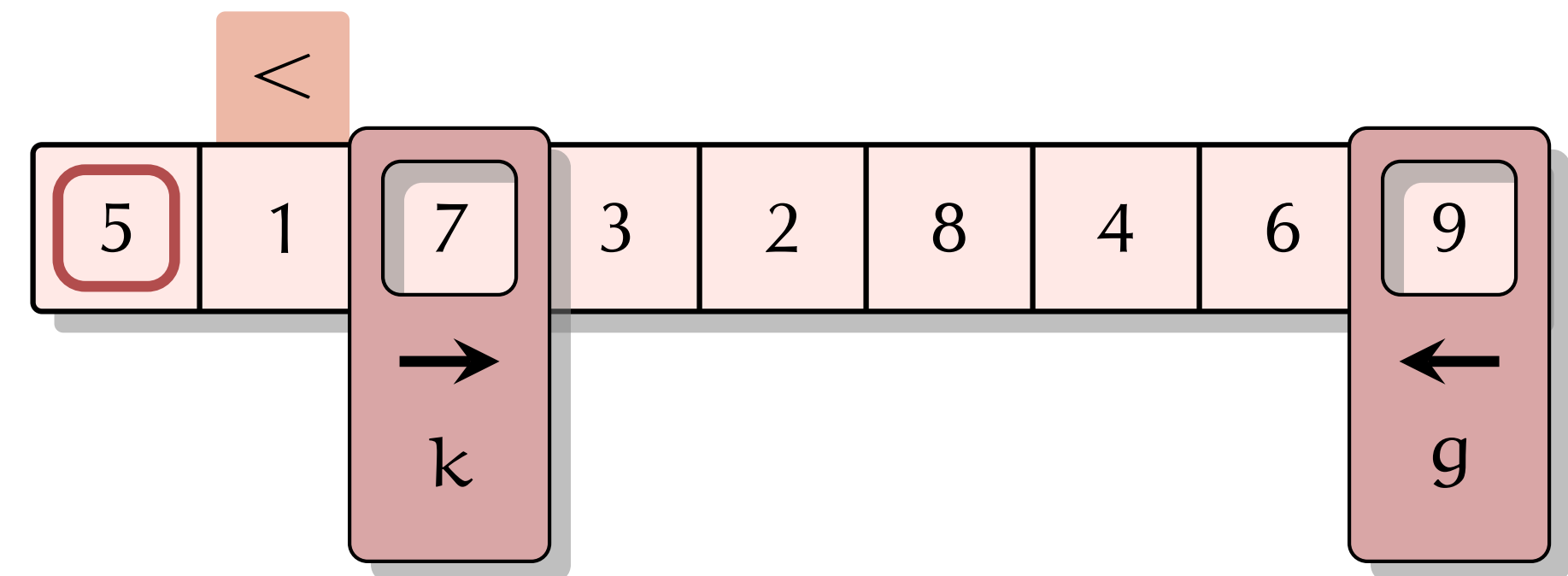


36 Vergleiche



Quicksort:

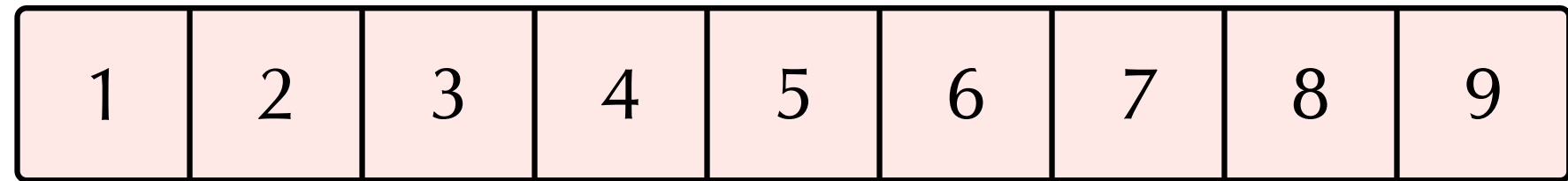
- Bringe ein Objekt an die richtige Stelle
- Bringe die anderen nach links und rechts



Zwei Sortialgorithmen

Selection Sort:

- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle

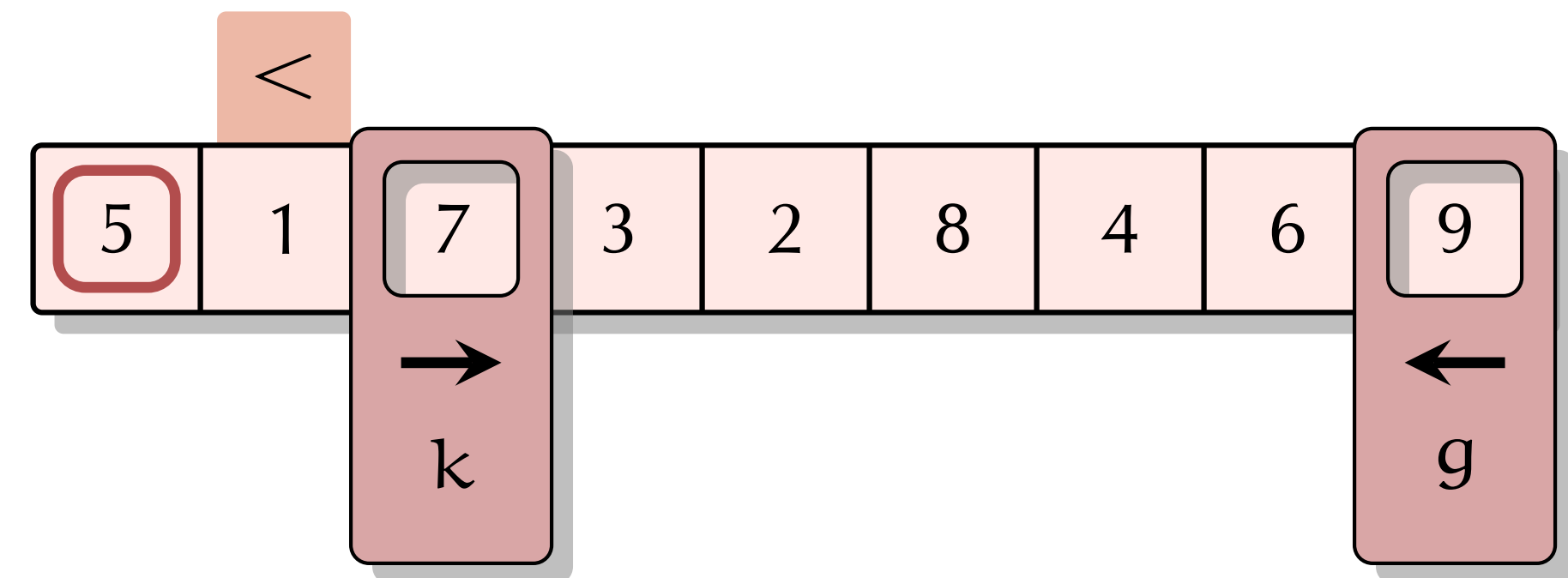


36 Vergleiche



Quicksort:

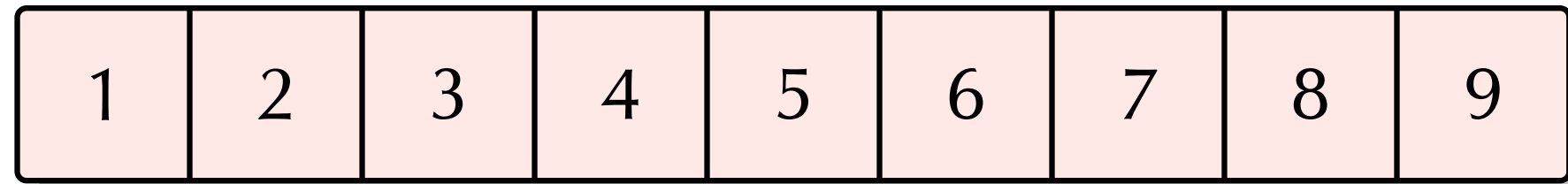
- Bringe ein Objekt an die richtige Stelle
- Bringe die anderen nach links und rechts



Zwei Sortialgorithmen

Selection Sort:

- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle

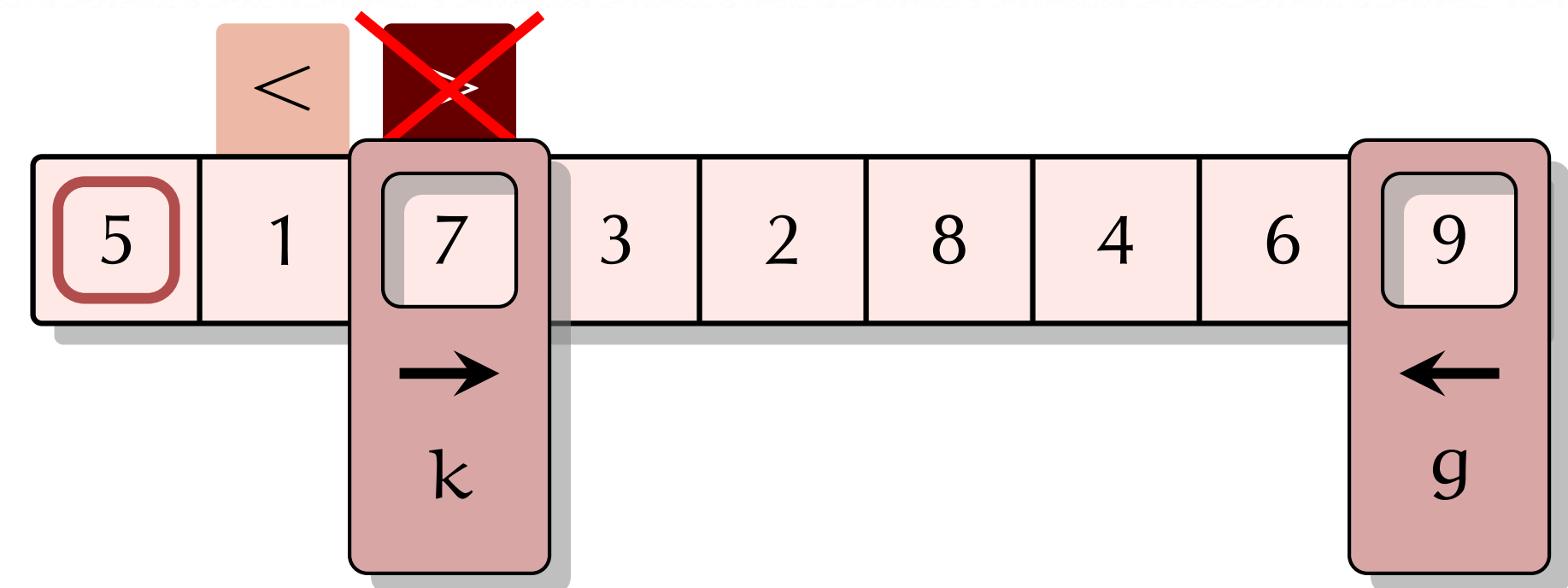


36 Vergleiche



Quicksort:

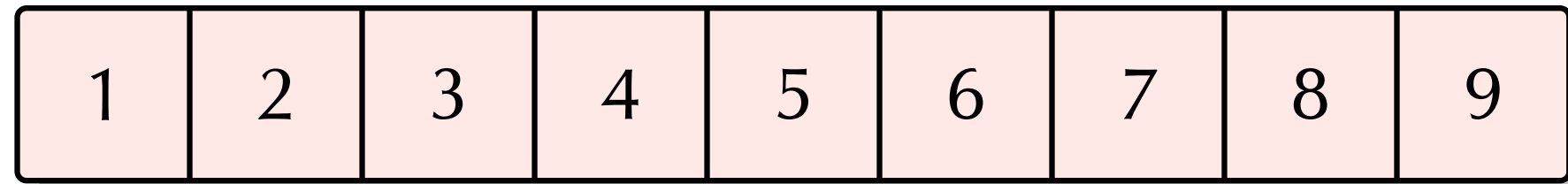
- Bringe ein Objekt an die richtige Stelle
- Bringe die anderen nach links und rechts



Zwei Sortialgorithmen

Selection Sort:

- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle

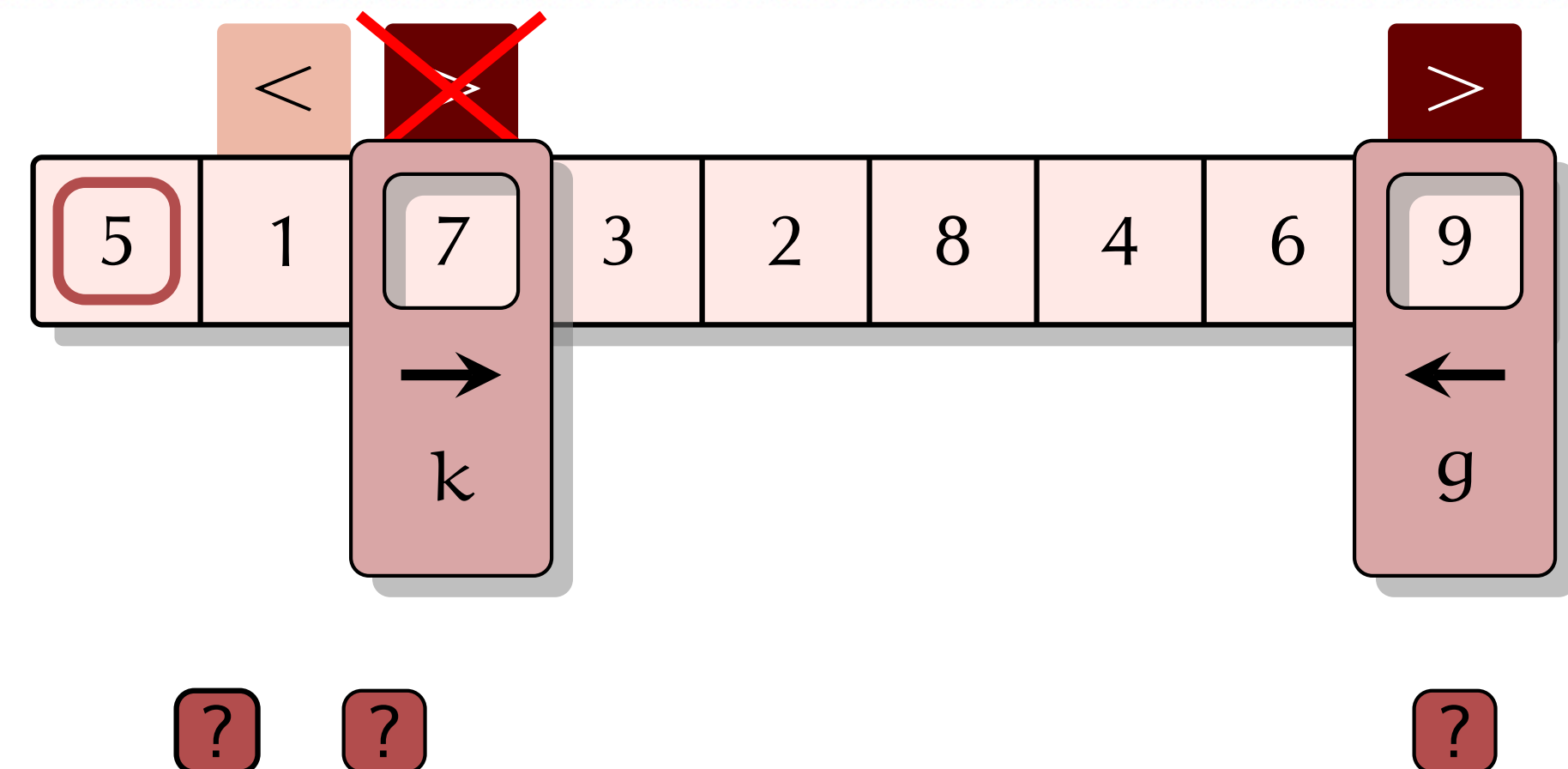


36 Vergleiche



Quicksort:

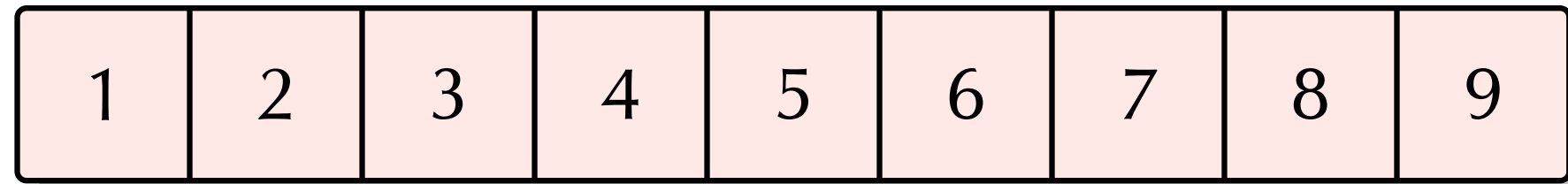
- Bringe ein Objekt an die richtige Stelle
- Bringe die anderen nach links und rechts



Zwei Sortialgorithmen

Selection Sort:

- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle

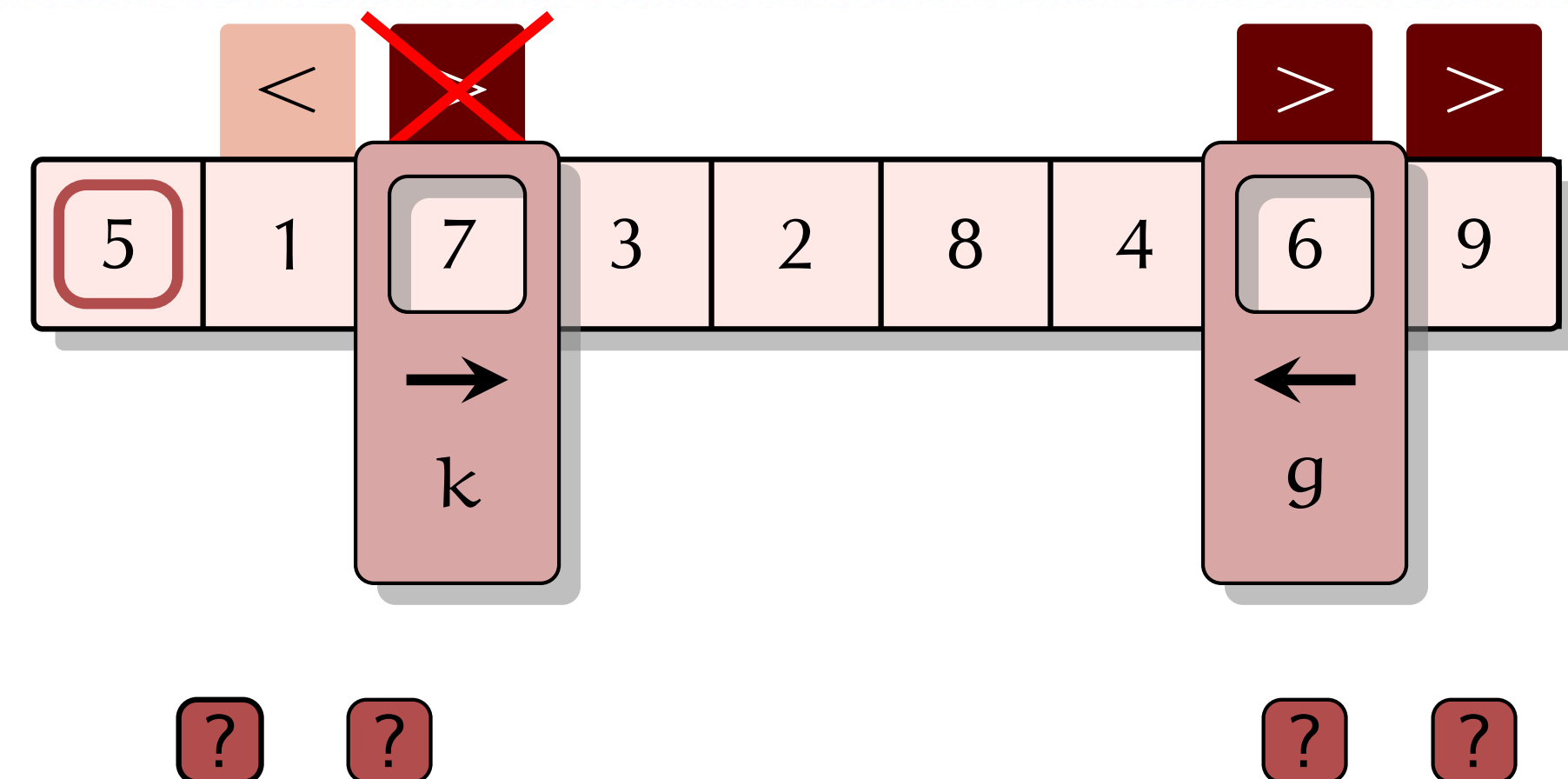


36 Vergleiche



Quicksort:

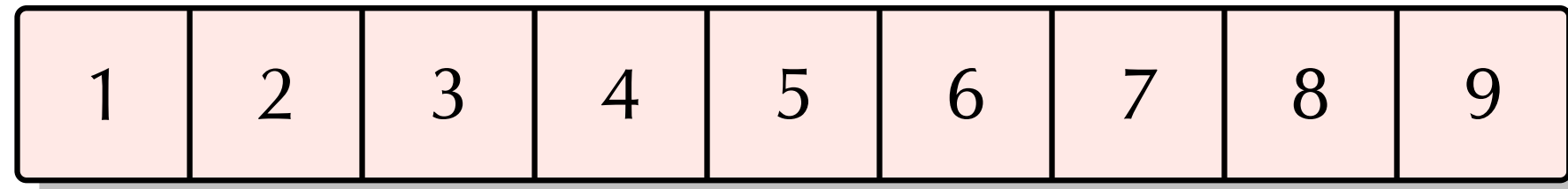
- Bringe ein Objekt an die richtige Stelle
- Bringe die anderen nach links und rechts



Zwei Sortialgorithmen

Selection Sort:

- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle

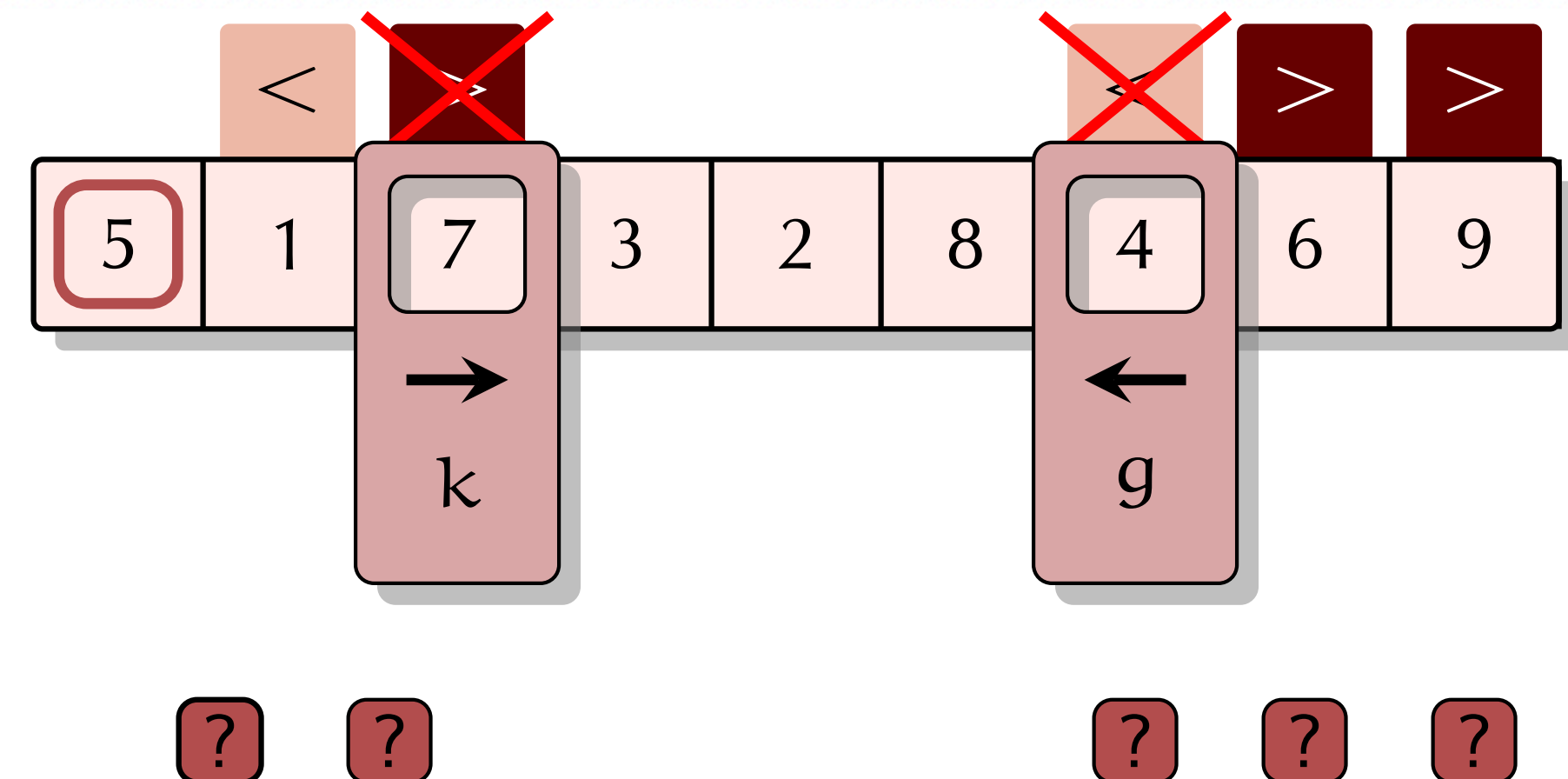


36 Vergleiche



Quicksort:

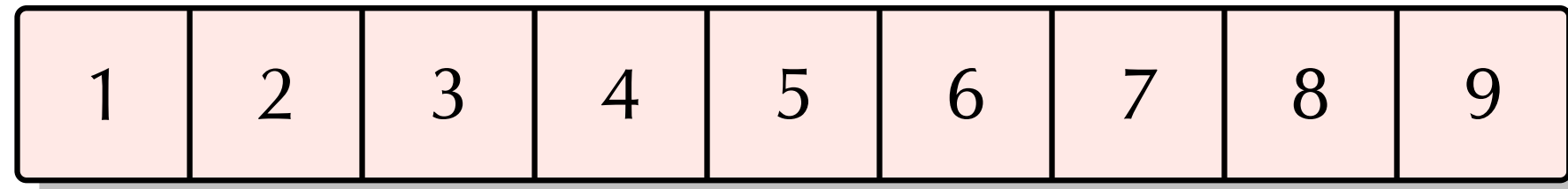
- Bringe ein Objekt an die richtige Stelle
- Bringe die anderen nach links und rechts



Zwei Sortialgorithmen

Selection Sort:

- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle

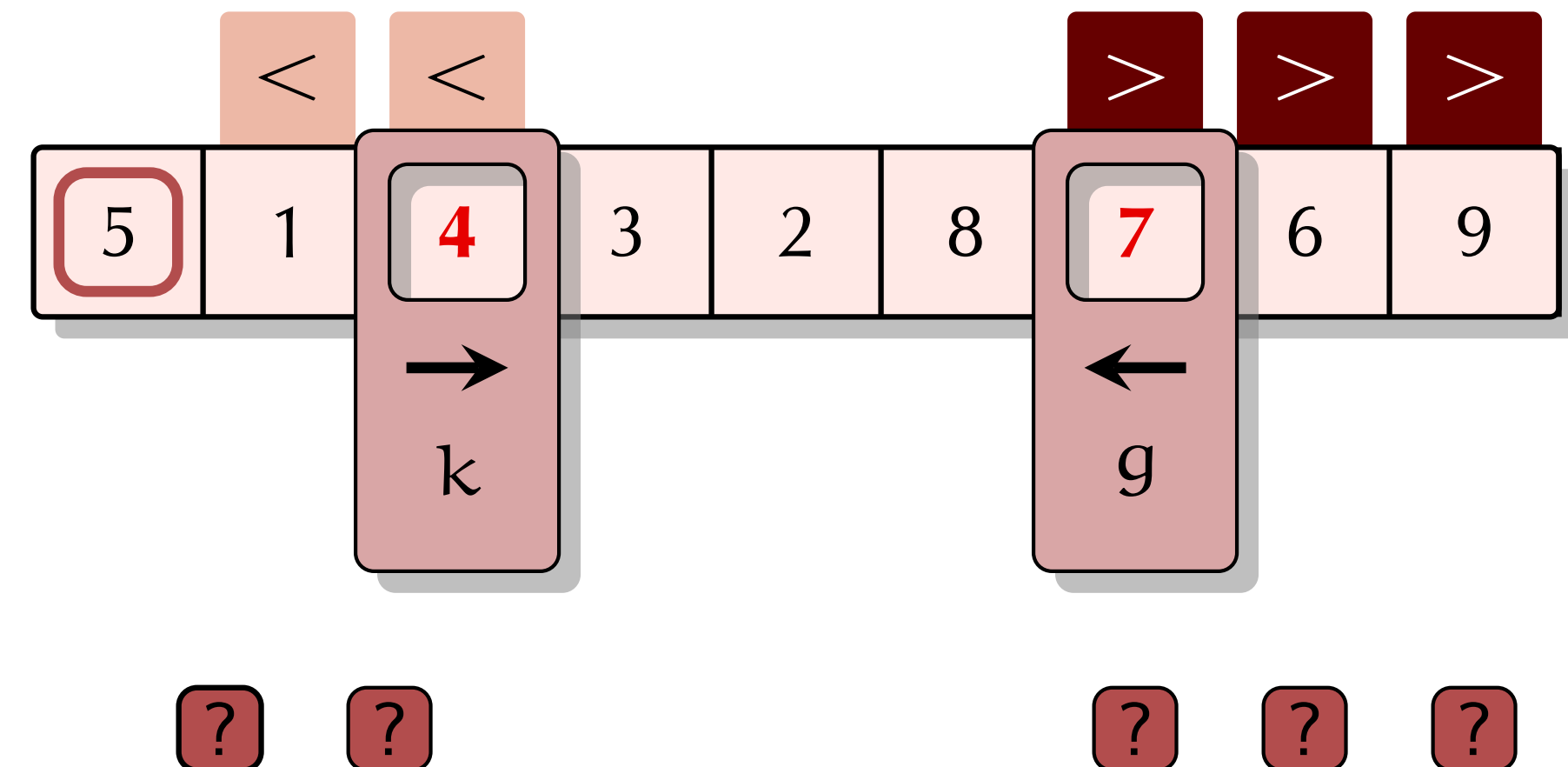


36 Vergleiche



Quicksort:

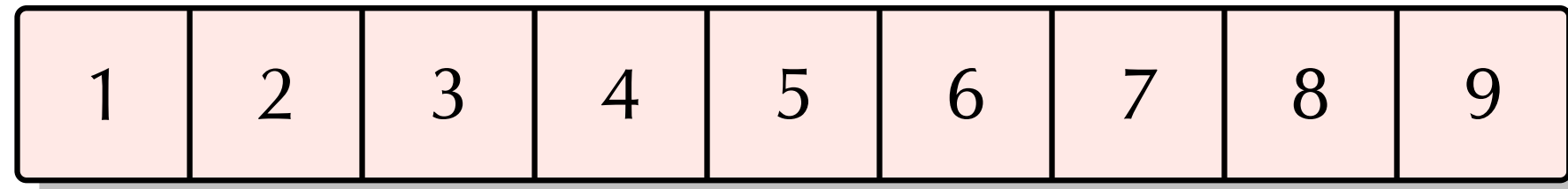
- Bringe ein Objekt an die richtige Stelle
- Bringe die anderen nach links und rechts



Zwei Sortialgorithmen

Selection Sort:

- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle

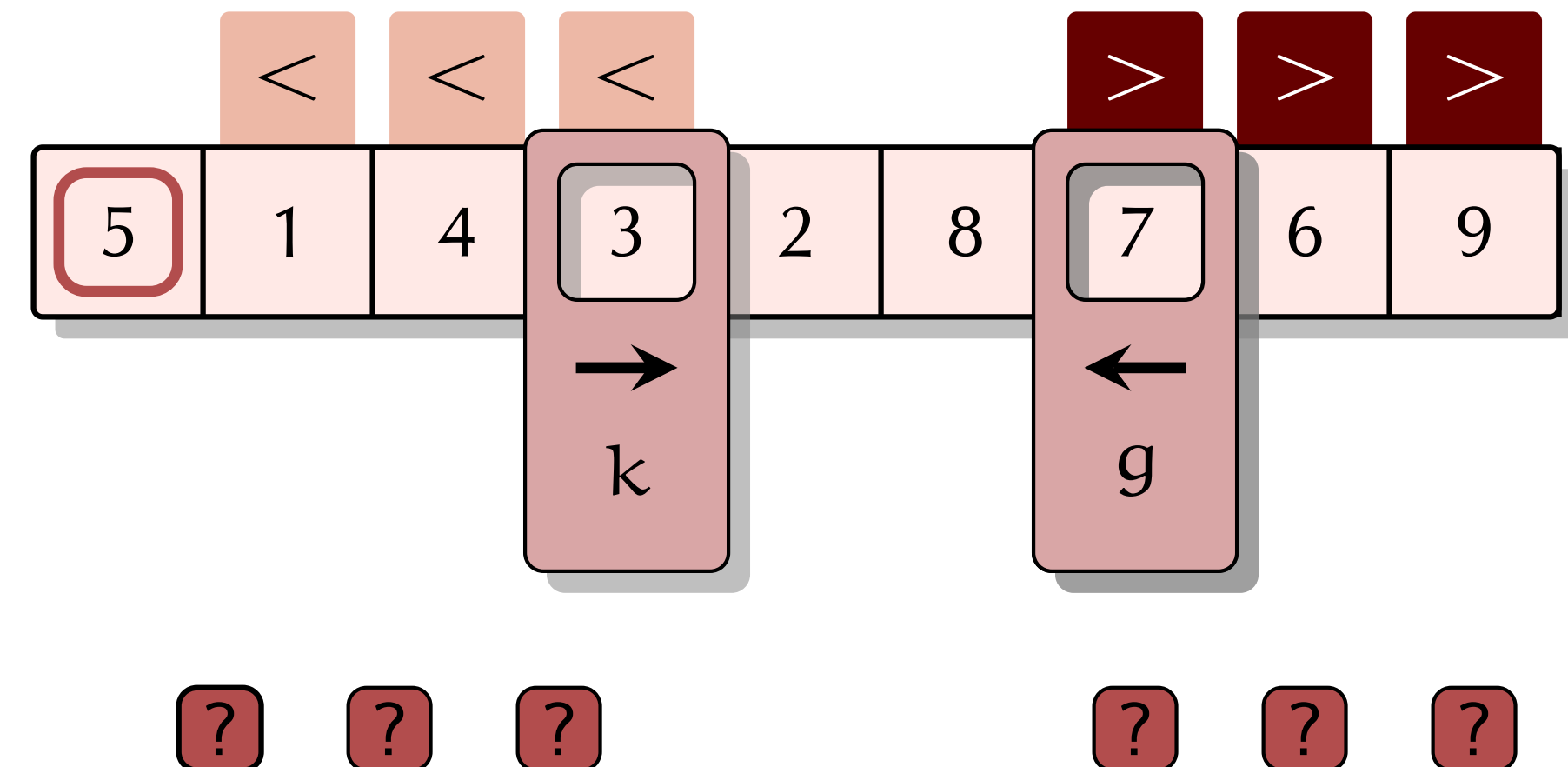


36 Vergleiche



Quicksort:

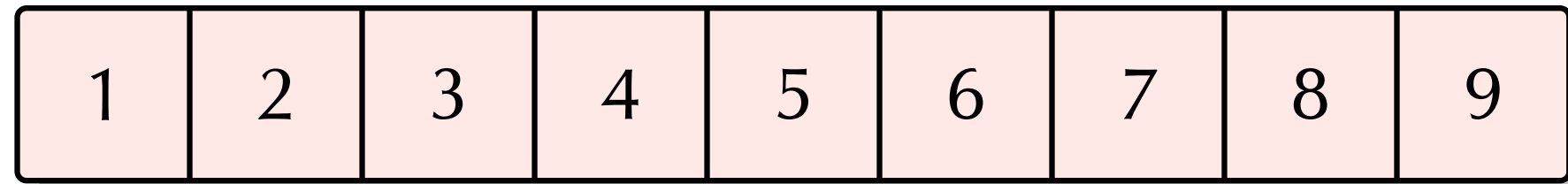
- Bringe ein Objekt an die richtige Stelle
- Bringe die anderen nach links und rechts



Zwei Sortialgorithmen

Selection Sort:

- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle

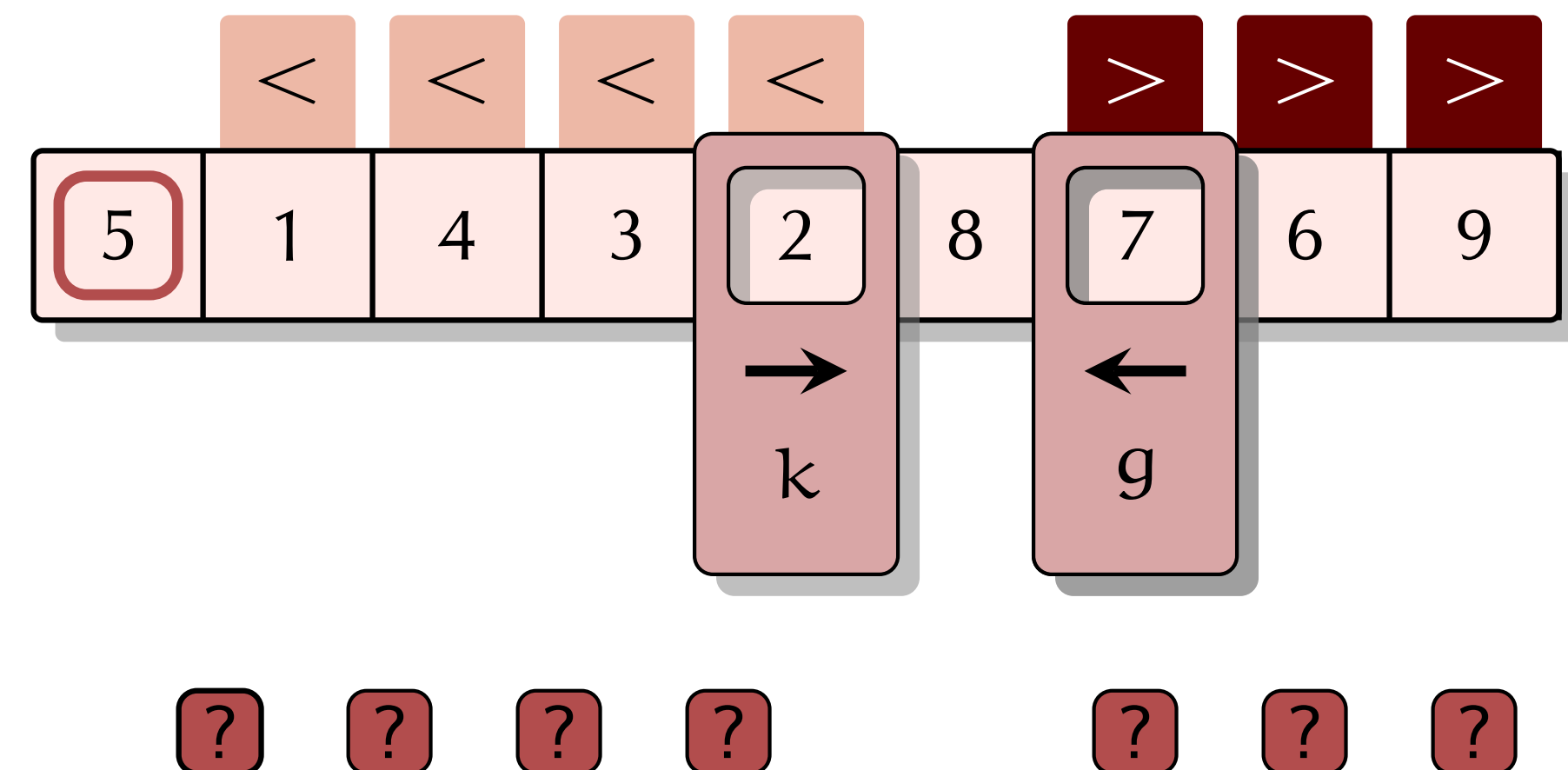


36 Vergleiche



Quicksort:

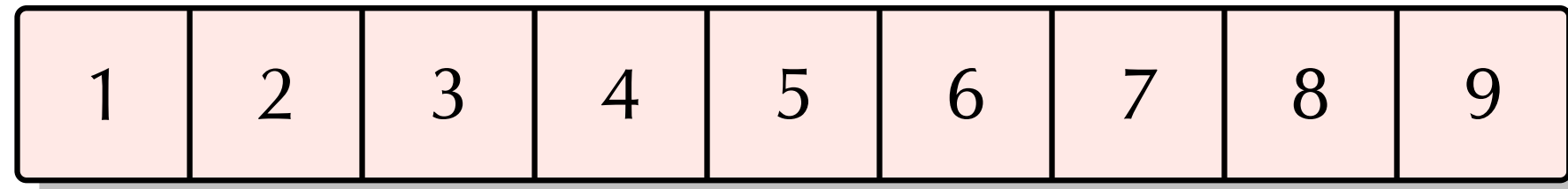
- Bringe ein Objekt an die richtige Stelle
- Bringe die anderen nach links und rechts



Zwei Sortialgorithmen

Selection Sort:

- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle

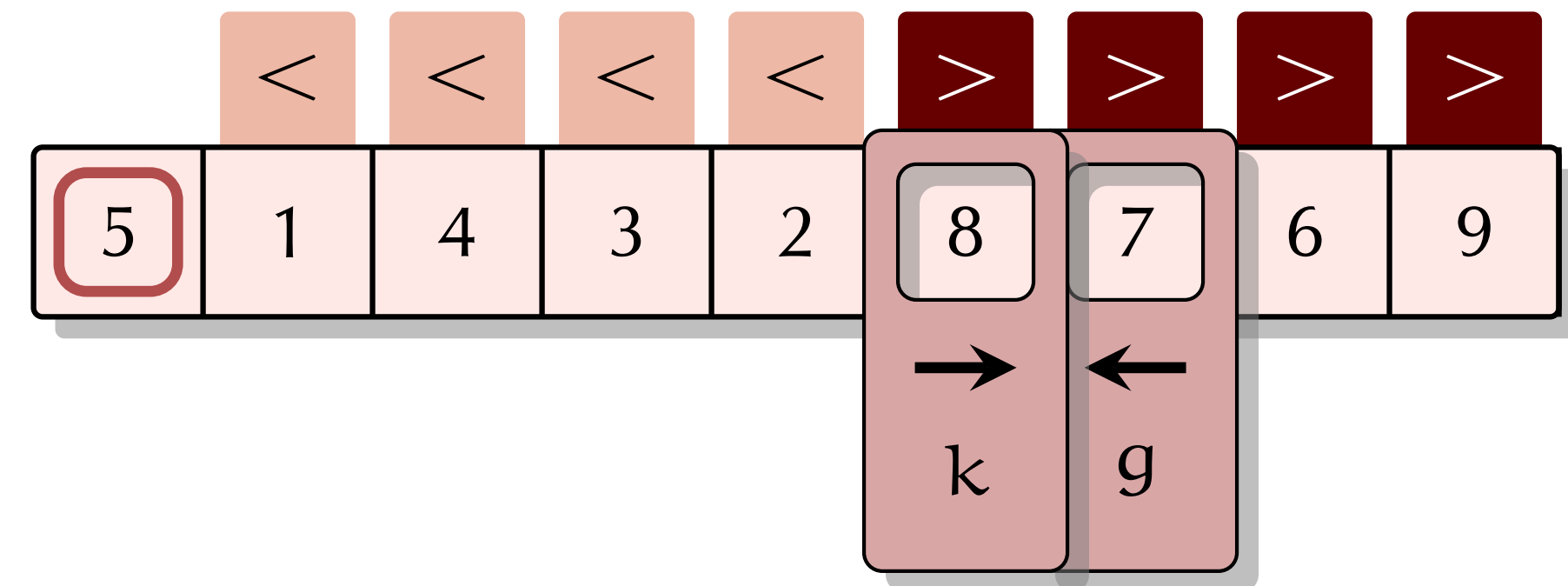


36 Vergleiche



Quicksort:

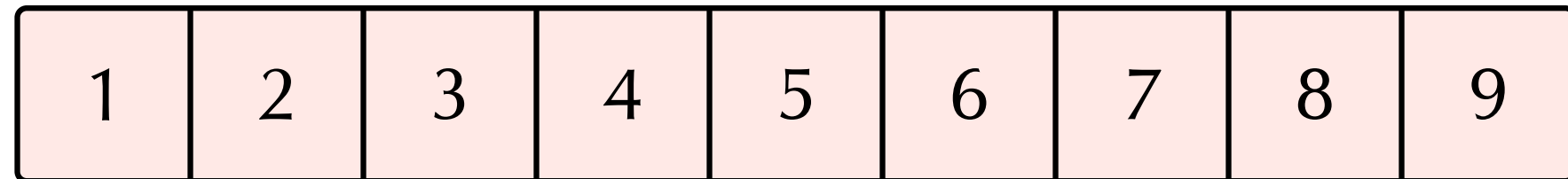
- Bringe ein Objekt an die richtige Stelle
- Bringe die anderen nach links und rechts



Zwei Sortialgorithmen

Selection Sort:

- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle

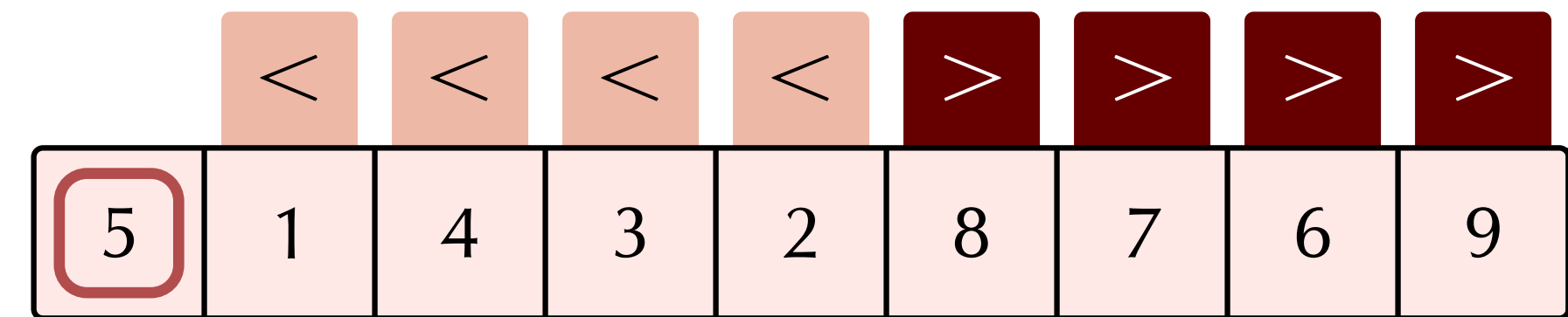


36 Vergleiche



Quicksort:

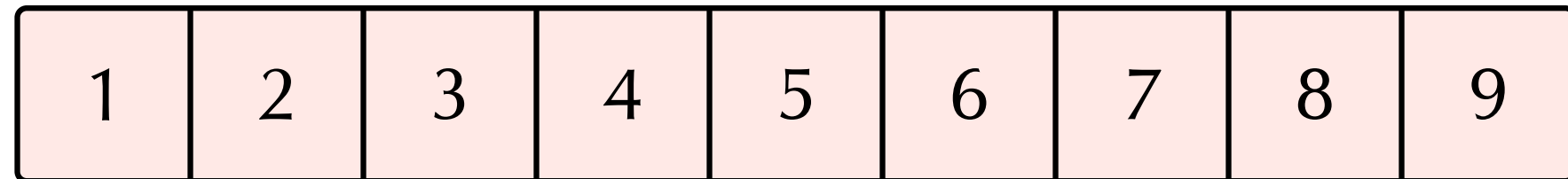
- Bringe ein Objekt an die richtige Stelle
- Bringe die anderen nach links und rechts



Zwei Sortialgorithmen

Selection Sort:

- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle

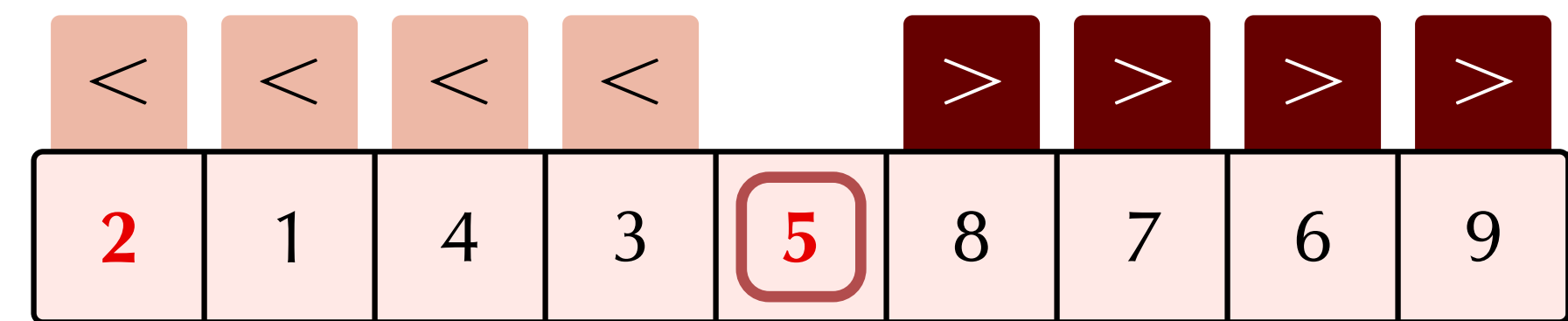


36 Vergleiche



Quicksort:

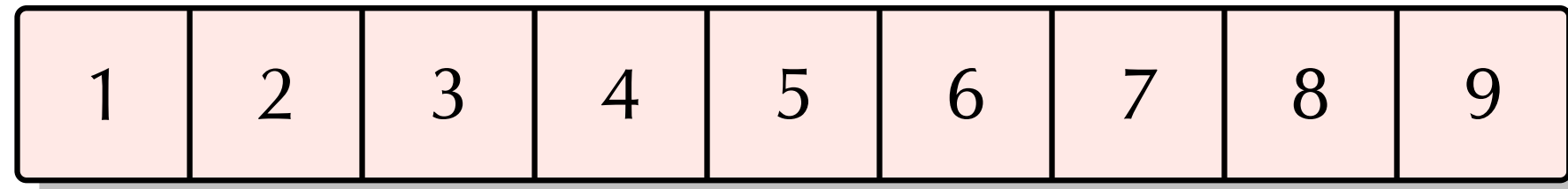
- Bringe ein Objekt an die richtige Stelle
- Bringe die anderen nach links und rechts



Zwei Sortialgorithmen

Selection Sort:

- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle

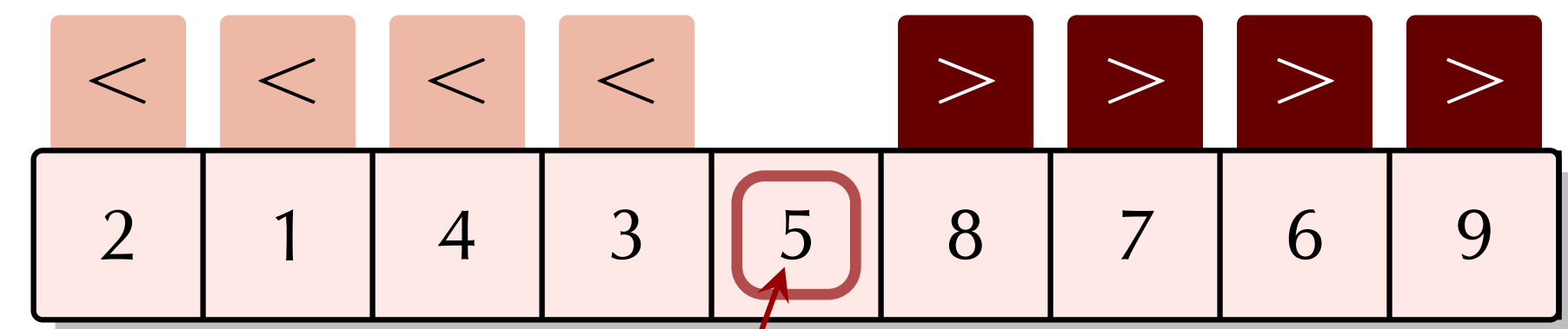


36 Vergleiche



Quicksort:

- Bringe ein Objekt an die richtige Stelle
- Bringe die anderen nach links und rechts



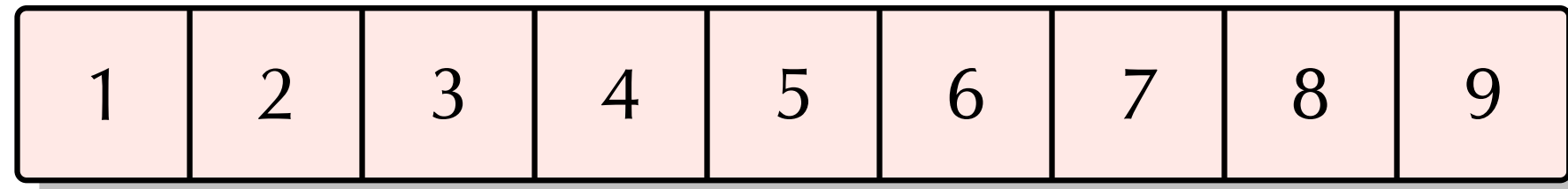
reached final place!



Zwei Sortialgorithmen

Selection Sort:

- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle

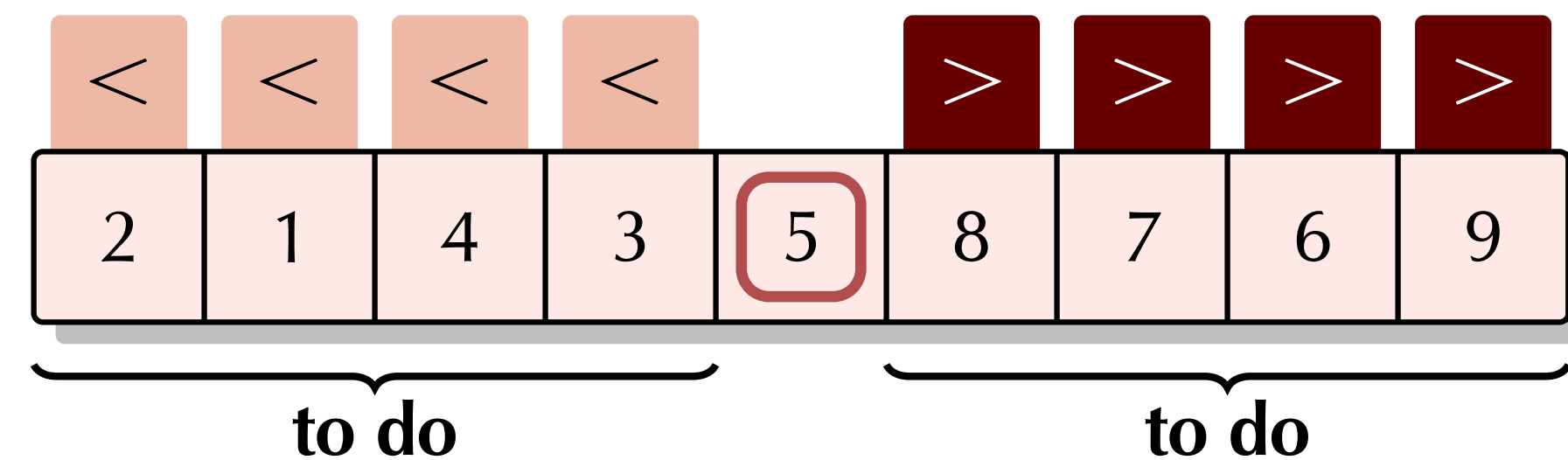


36 Vergleiche



Quicksort:

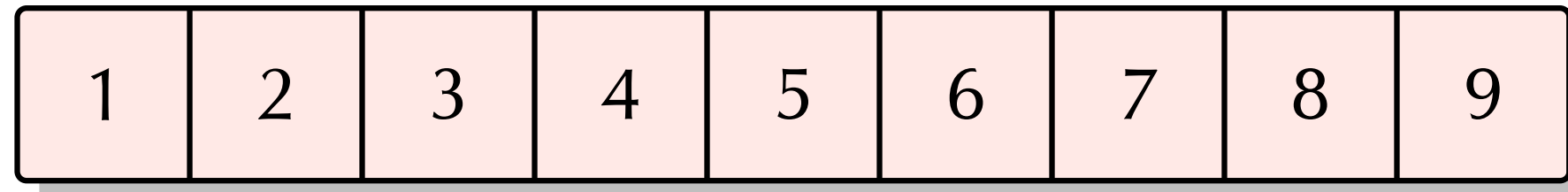
- Bringe ein Objekt an die richtige Stelle
- Bringe die anderen nach links und rechts



Zwei Sortialgorithmen

Selection Sort:

- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle



36 Vergleiche



Quicksort:

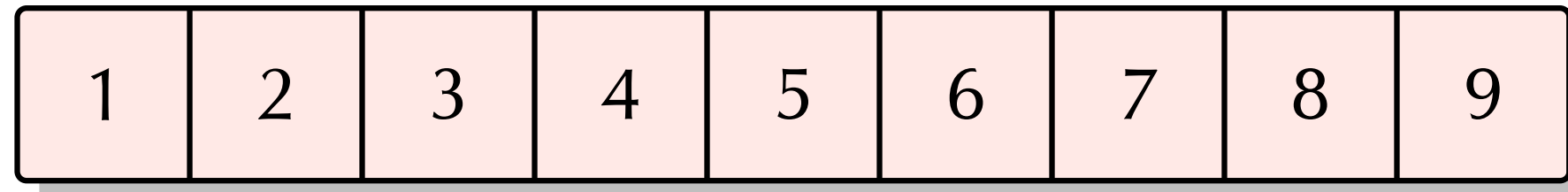
- Bringe ein Objekt an die richtige Stelle
- Bringe die anderen nach links und rechts



Zwei Sortialgorithmen

Selection Sort:

- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle



36 Vergleiche



Quicksort:

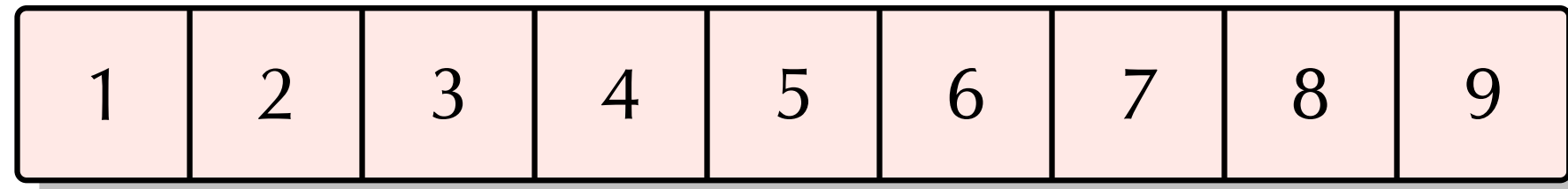
- Bringe ein Objekt an die richtige Stelle
- Bringe die anderen nach links und rechts



Zwei Sortialgorithmen

Selection Sort:

- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle

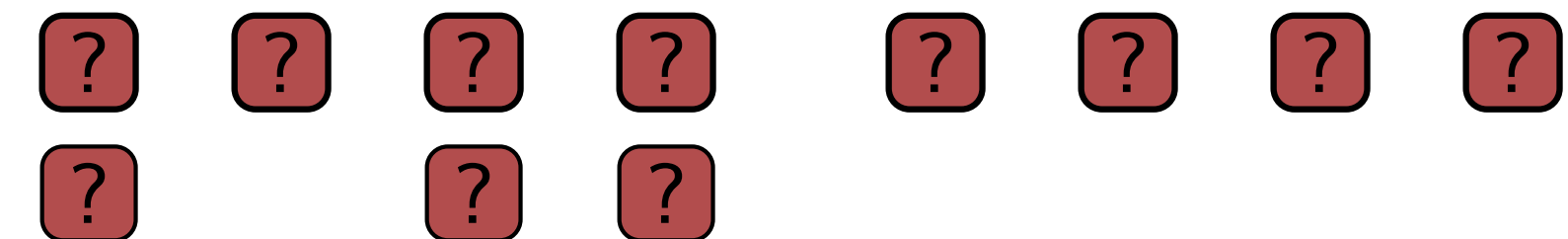
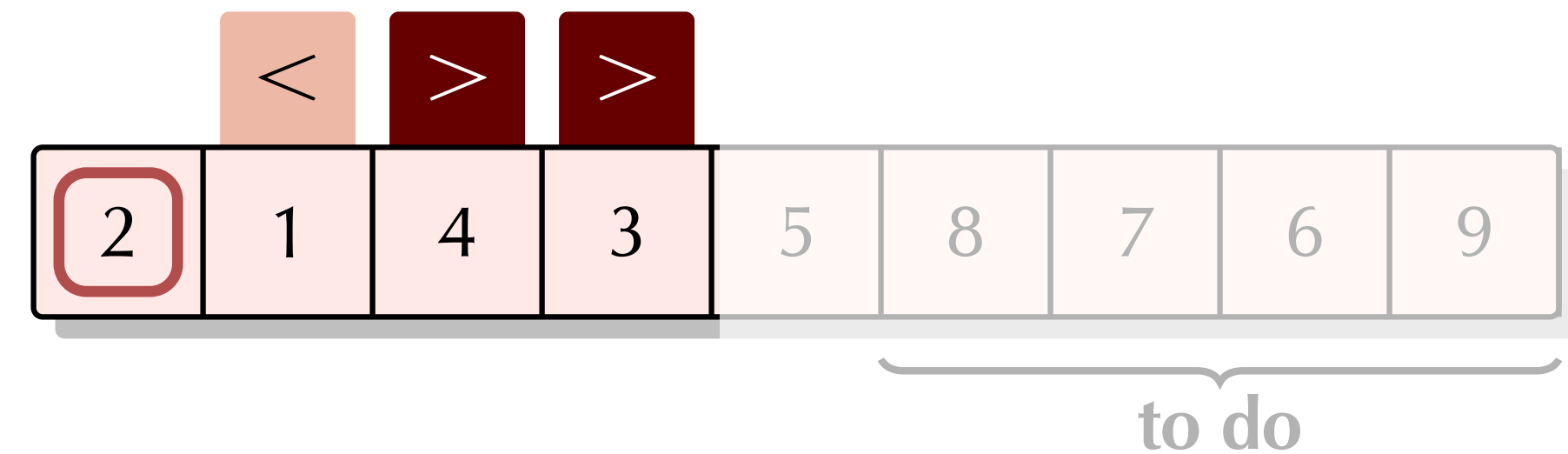


36 Vergleiche



Quicksort:

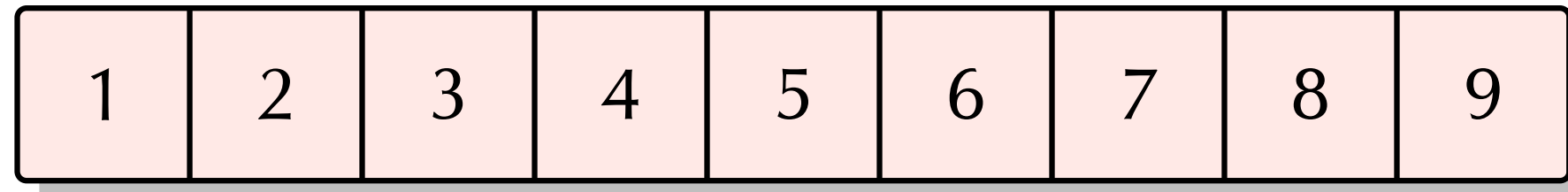
- Bringe ein Objekt an die richtige Stelle
- Bringe die anderen nach links und rechts



Zwei Sortialgorithmen

Selection Sort:

- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle

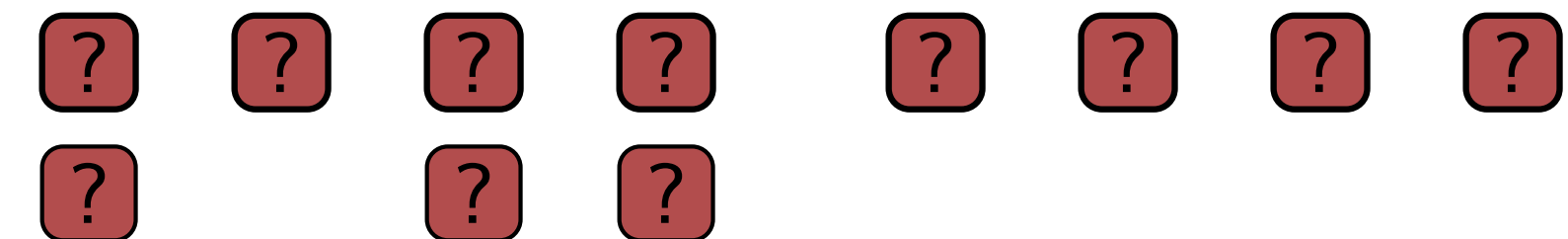
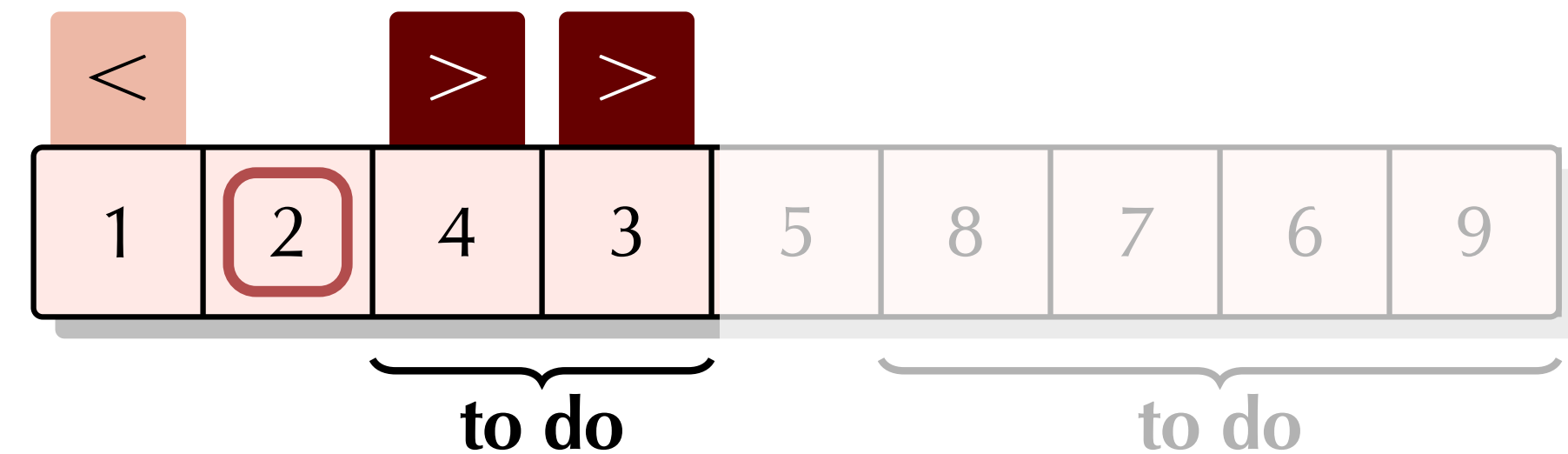


36 Vergleiche



Quicksort:

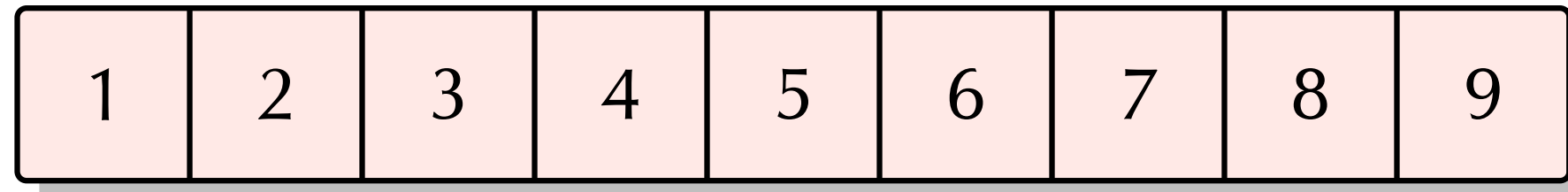
- Bringe ein Objekt an die richtige Stelle
- Bringe die anderen nach links und rechts



Zwei Sortialgorithmen

Selection Sort:

- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle

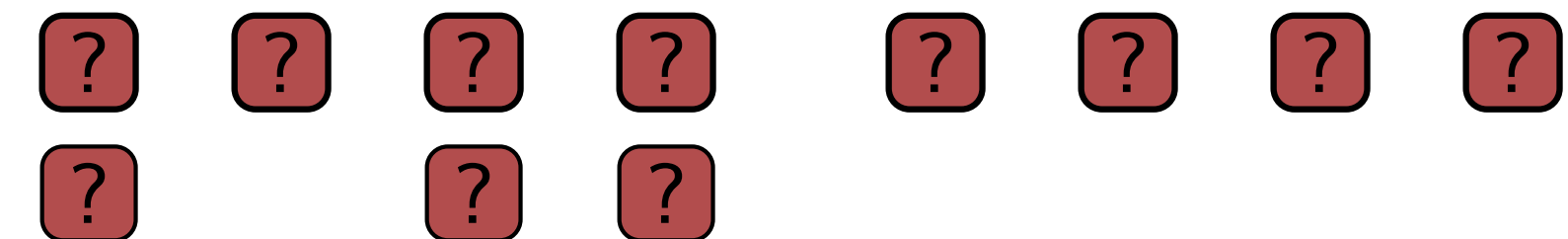


36 Vergleiche



Quicksort:

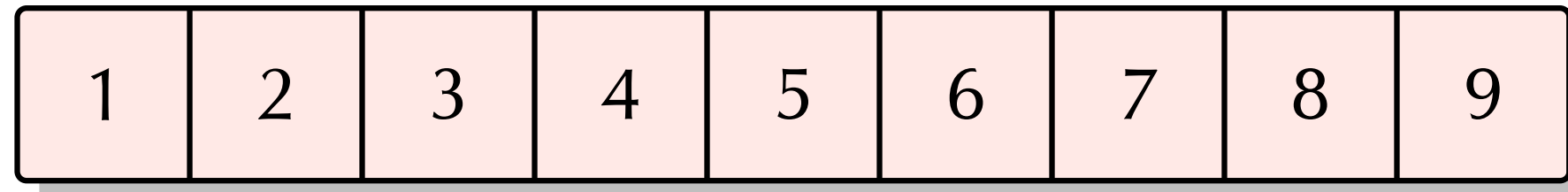
- Bringe ein Objekt an die richtige Stelle
- Bringe die anderen nach links und rechts



Zwei Sortialgorithmen

Selection Sort:

- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle

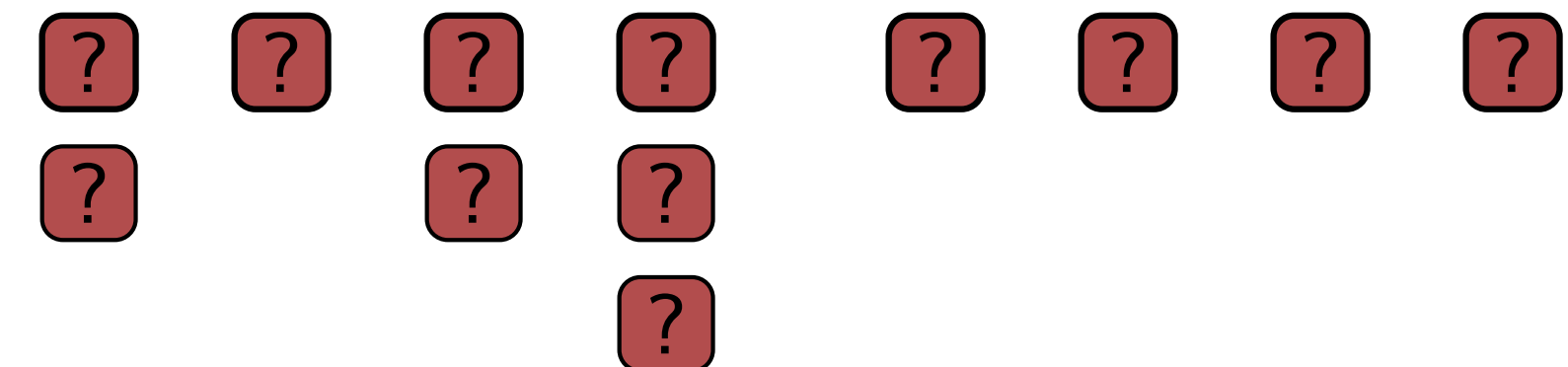
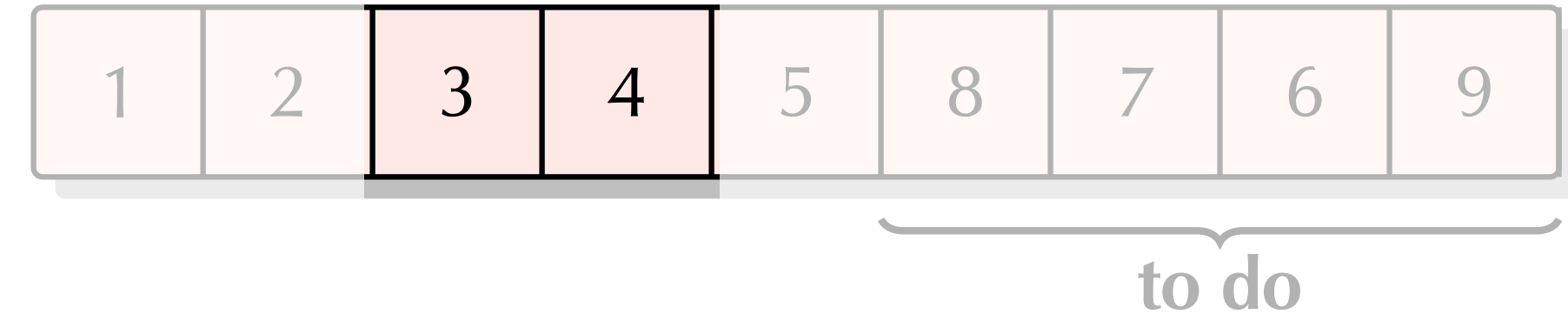


36 Vergleiche



Quicksort:

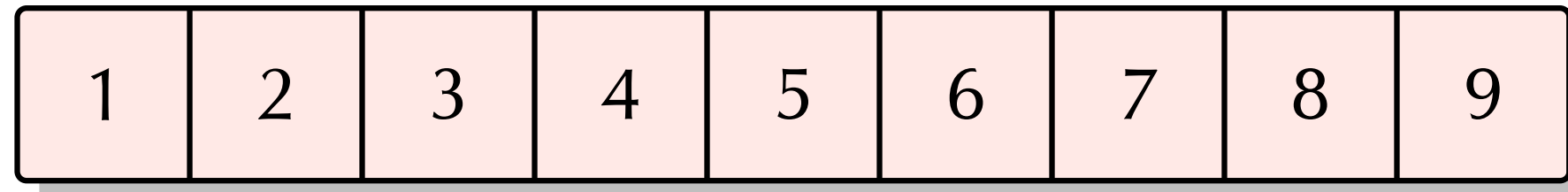
- Bringe ein Objekt an die richtige Stelle
- Bringe die anderen nach links und rechts



Zwei Sortialgorithmen

Selection Sort:

- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle

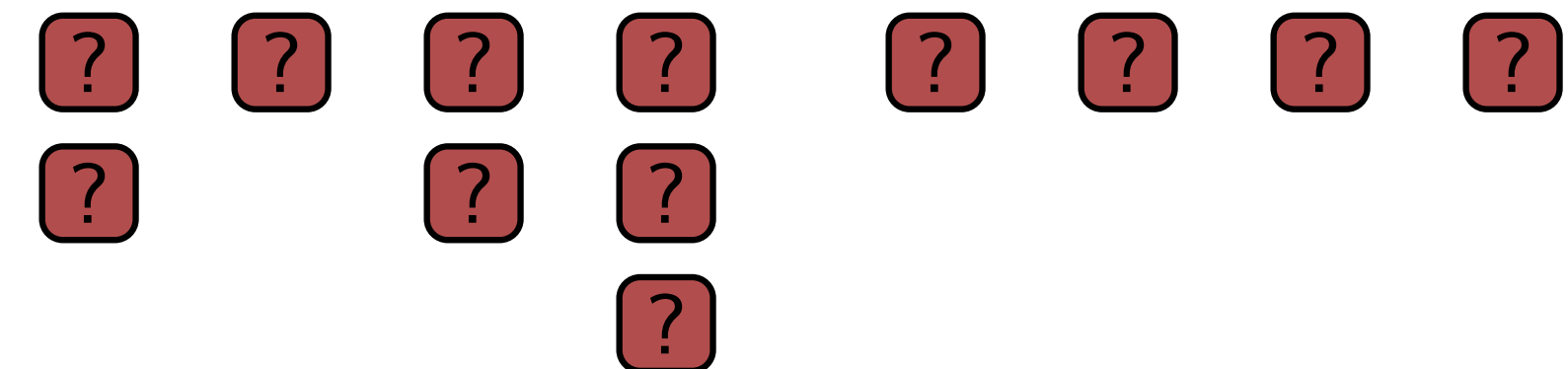


36 Vergleiche



Quicksort:

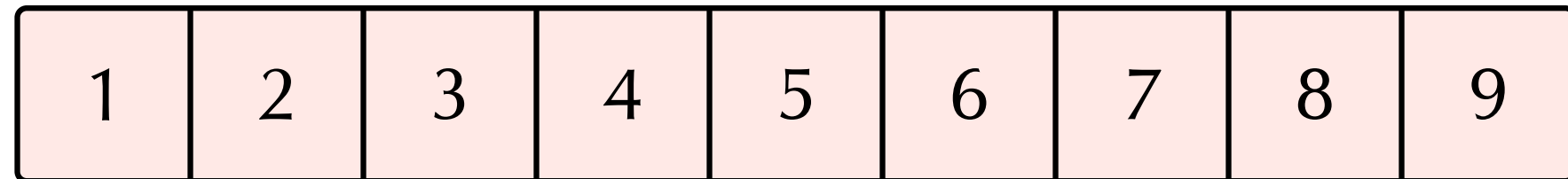
- Bringe ein Objekt an die richtige Stelle
- Bringe die anderen nach links und rechts



Zwei Sortialgorithmen

Selection Sort:

- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle

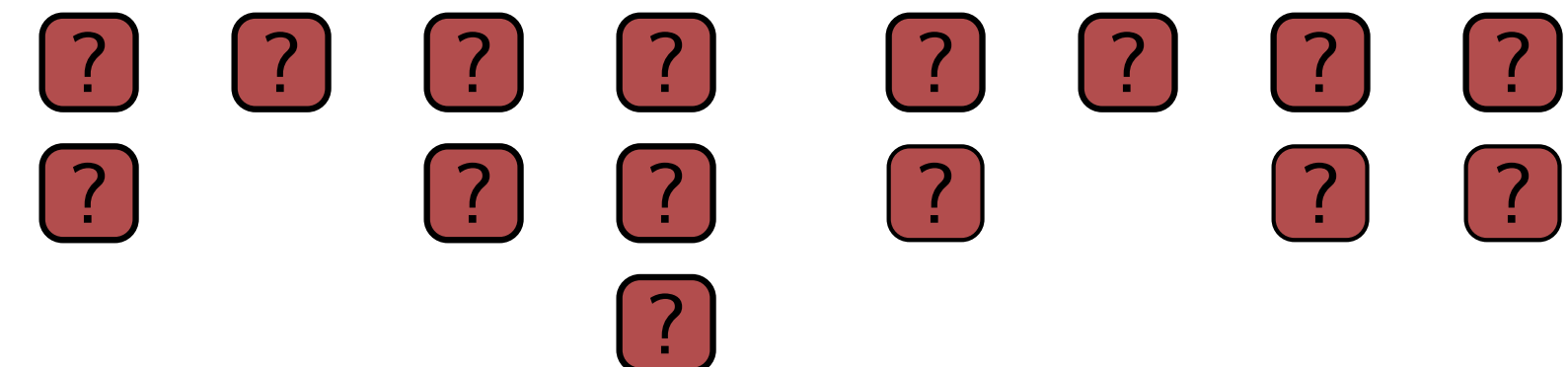
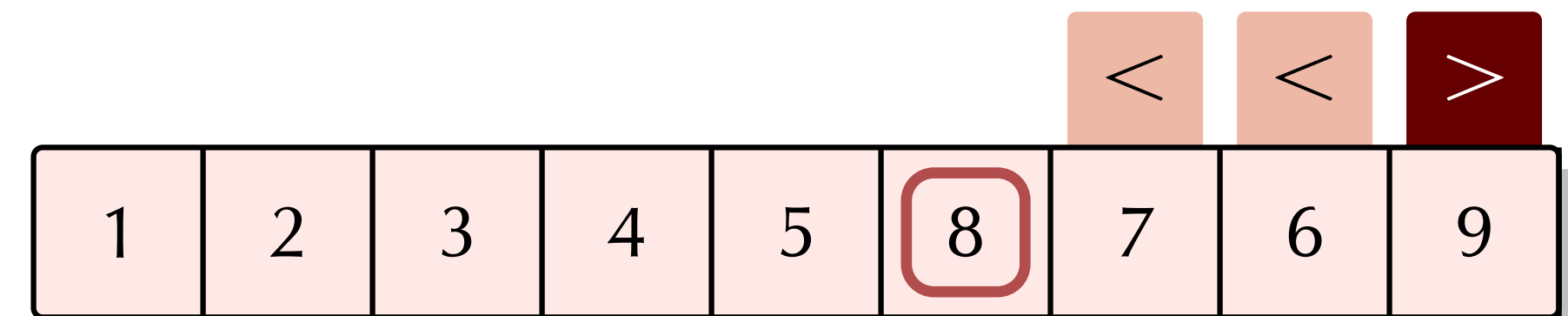


36 Vergleiche



Quicksort:

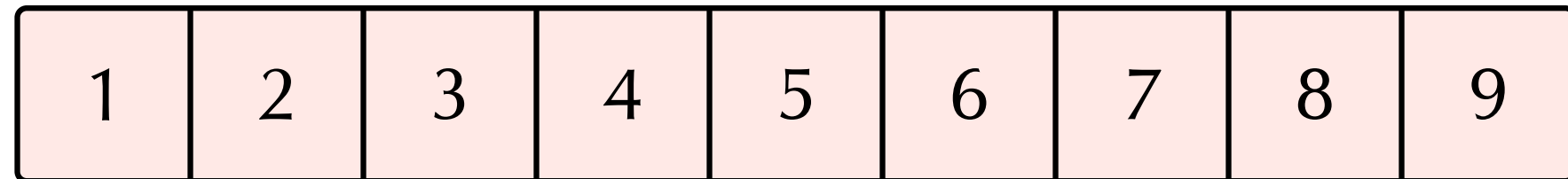
- Bringe ein Objekt an die richtige Stelle
- Bringe die anderen nach links und rechts



Zwei Sortialgorithmen

Selection Sort:

- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle

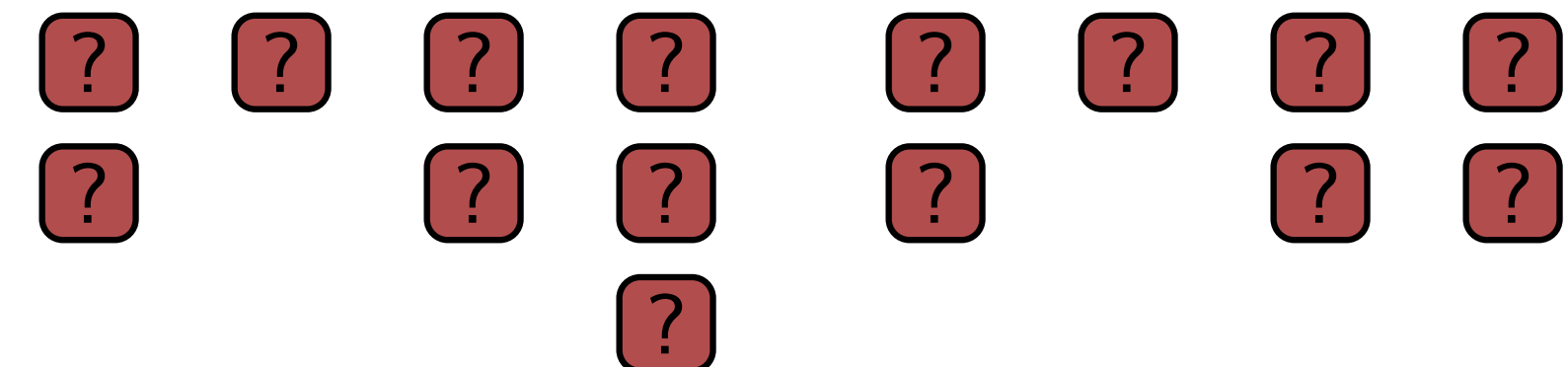
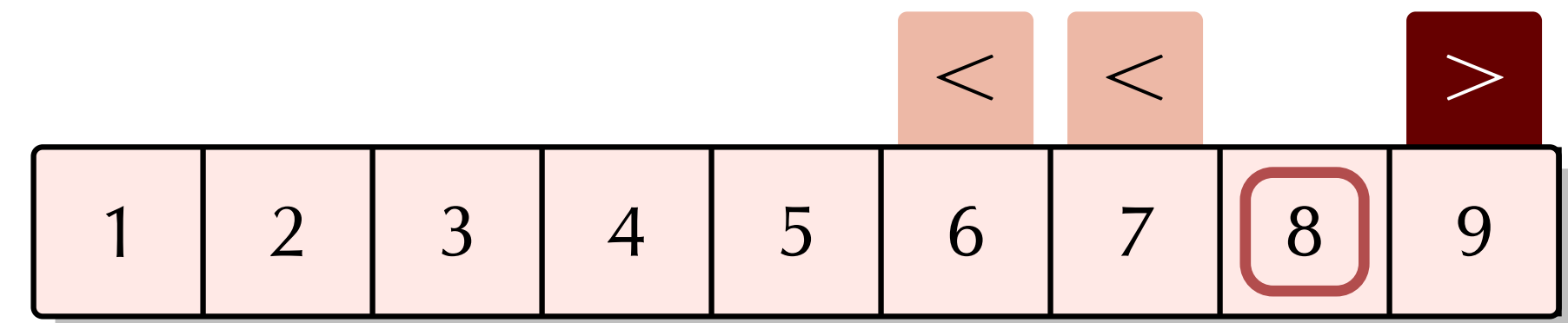


36 Vergleiche



Quicksort:

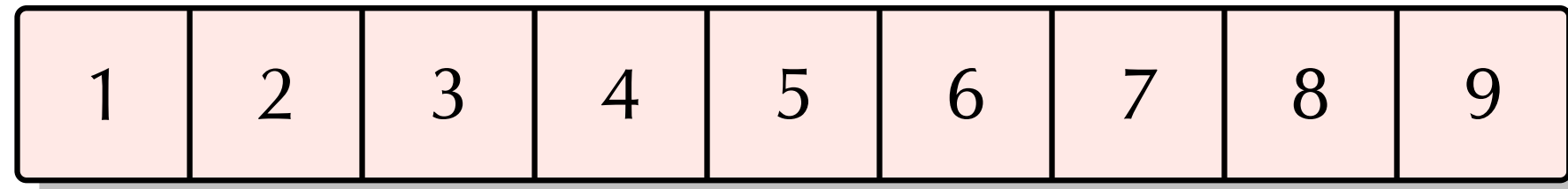
- Bringe ein Objekt an die richtige Stelle
- Bringe die anderen nach links und rechts



Zwei Sortialgorithmen

Selection Sort:

- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle

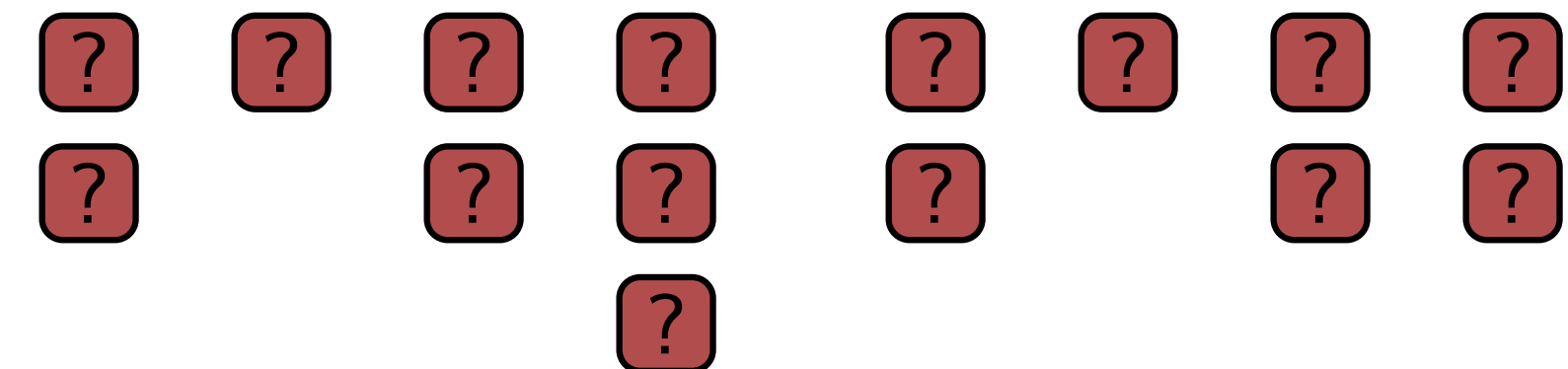
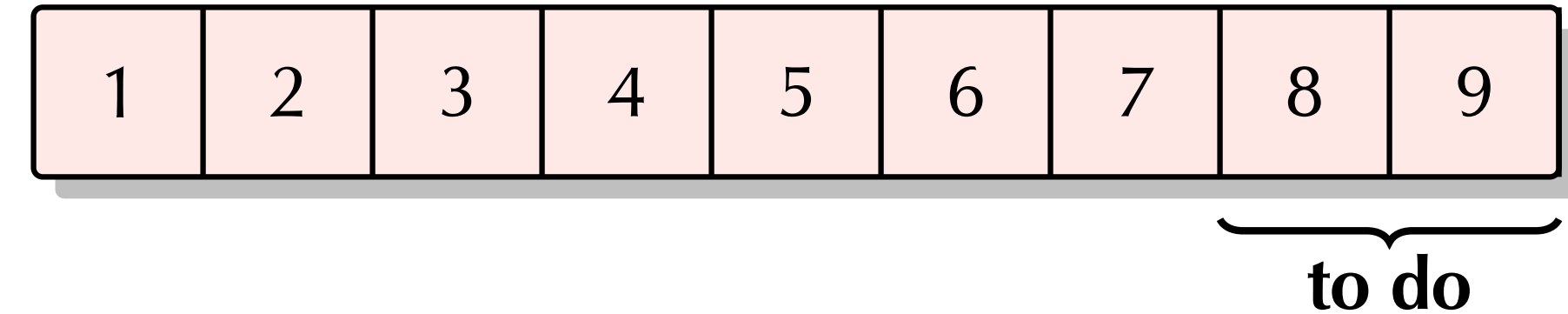


36 Vergleiche



Quicksort:

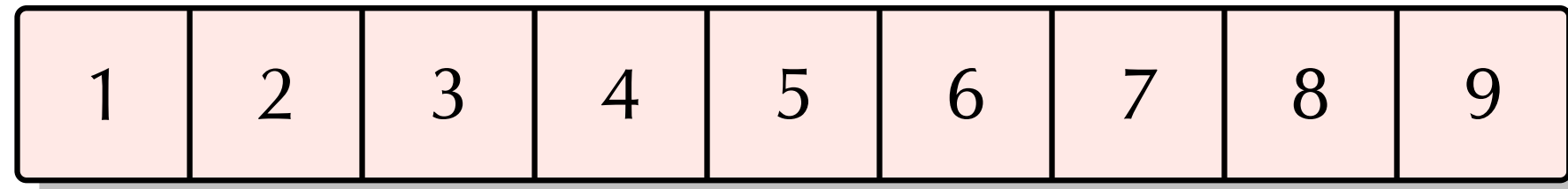
- Bringe ein Objekt an die richtige Stelle
- Bringe die anderen nach links und rechts



Zwei Sortialgorithmen

Selection Sort:

- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle

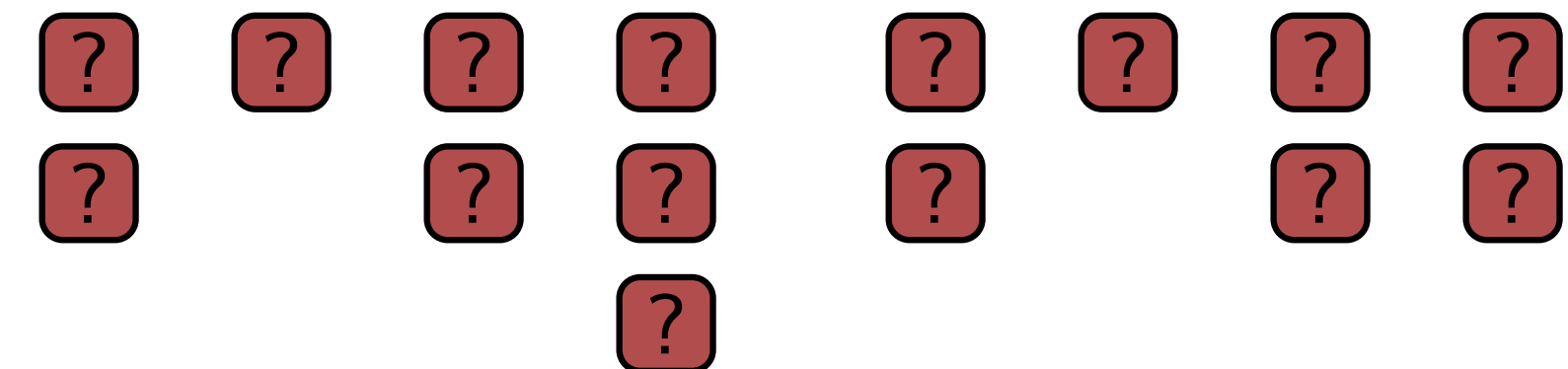
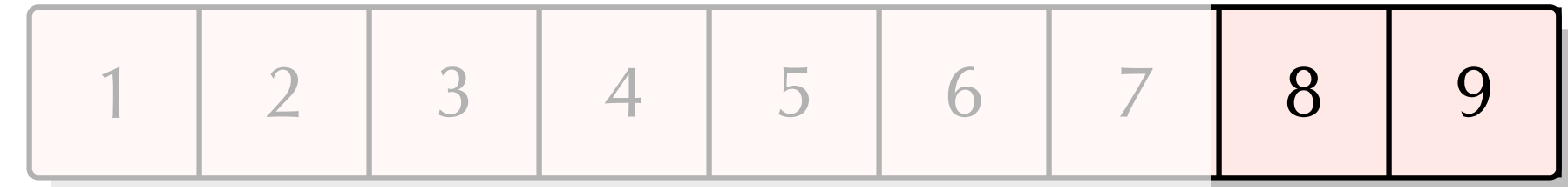


36 Vergleiche



Quicksort:

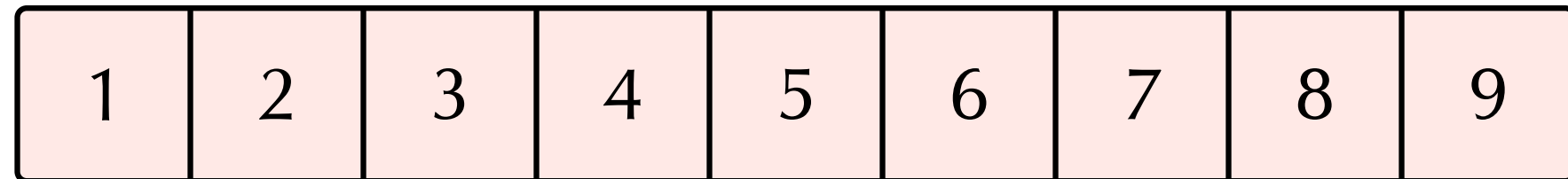
- Bringe ein Objekt an die richtige Stelle
- Bringe die anderen nach links und rechts



Zwei Sortialgorithmen

Selection Sort:

- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle

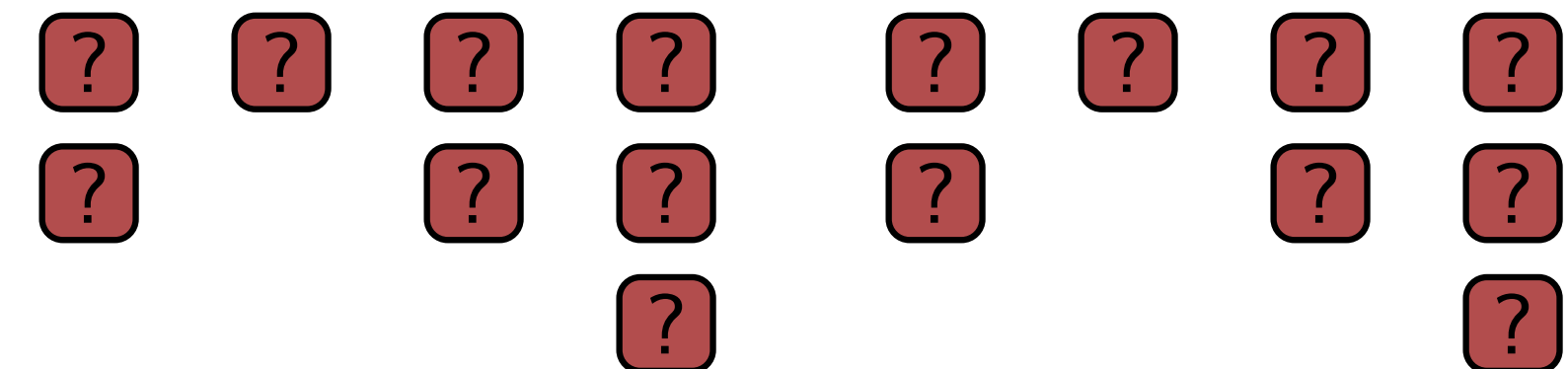
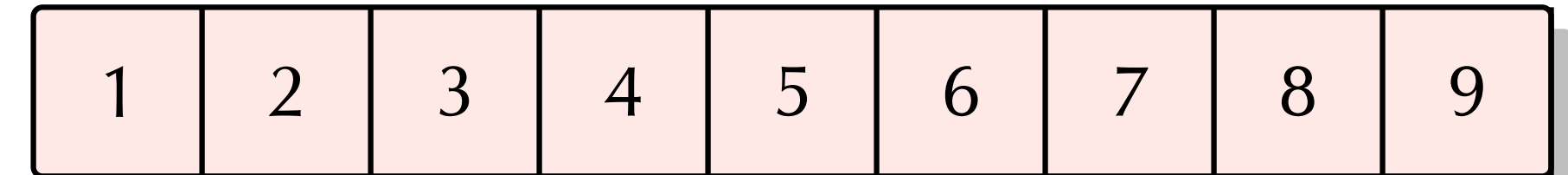


36 Vergleiche



Quicksort:

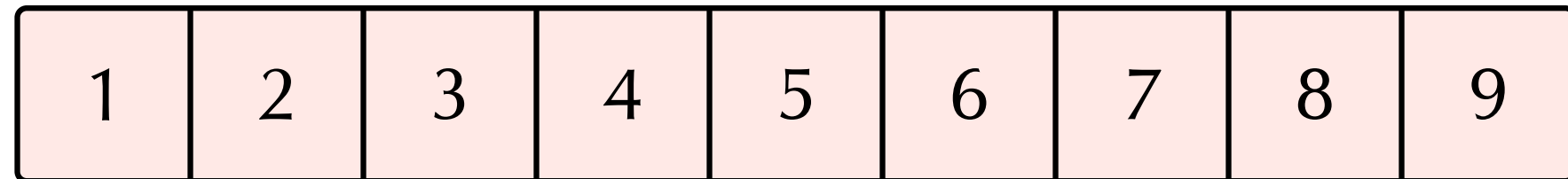
- Bringe ein Objekt an die richtige Stelle
- Bringe die anderen nach links und rechts



Zwei Sortialgorithmen

Selection Sort:

- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle

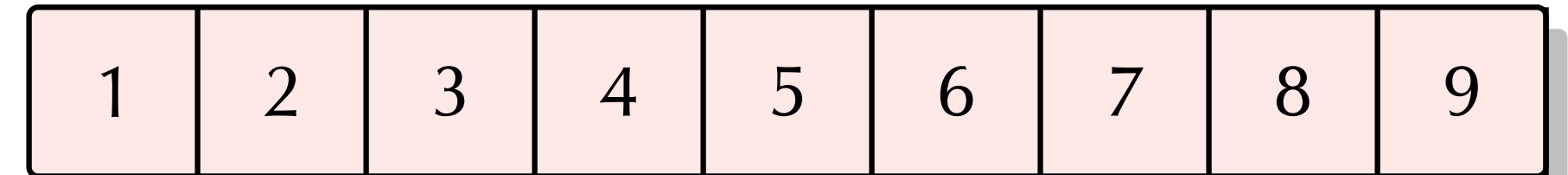


36 Vergleiche

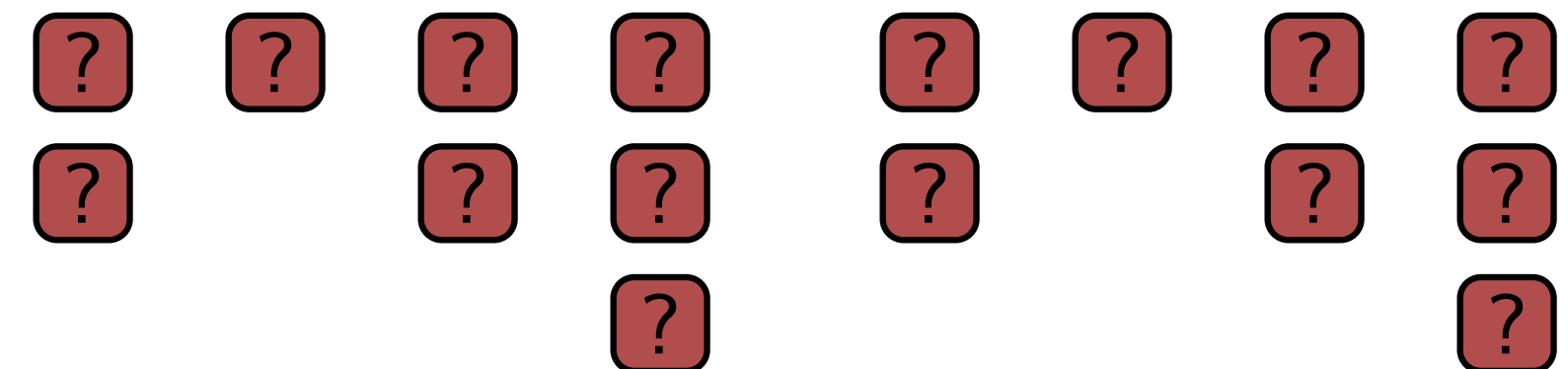


Quicksort:

- Bringe ein Objekt an die richtige Stelle
- Bringe die anderen nach links und rechts



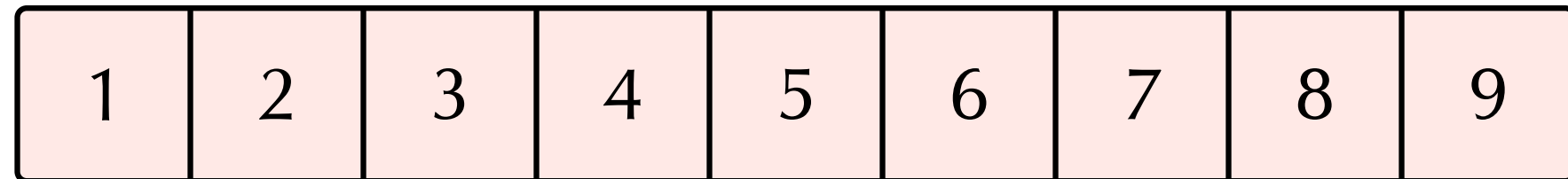
16 Vergleiche



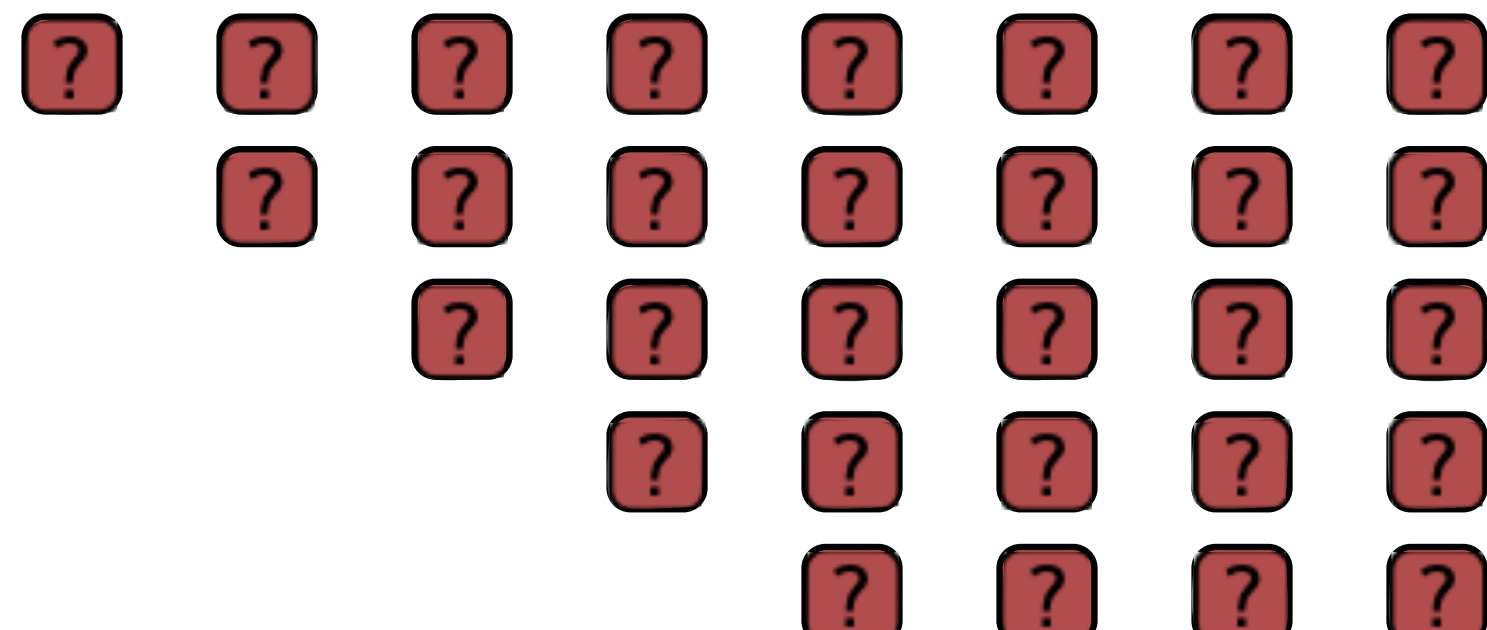
Zwei Sortialgorithmen

Selection Sort:

- Finde das kleinste verbleibende Objekt
- Bringe es an die richtige Stelle

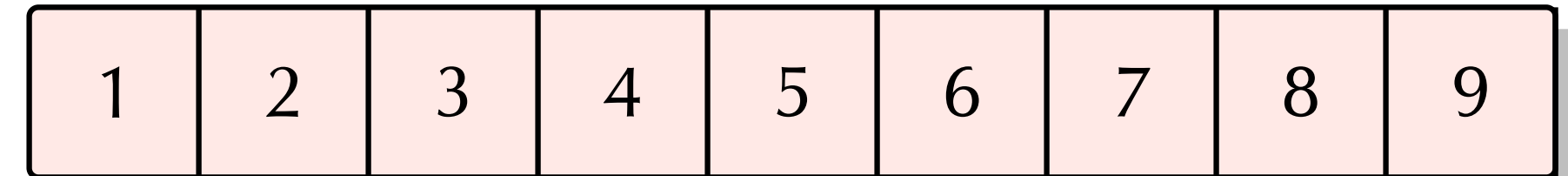


36 Vergleiche

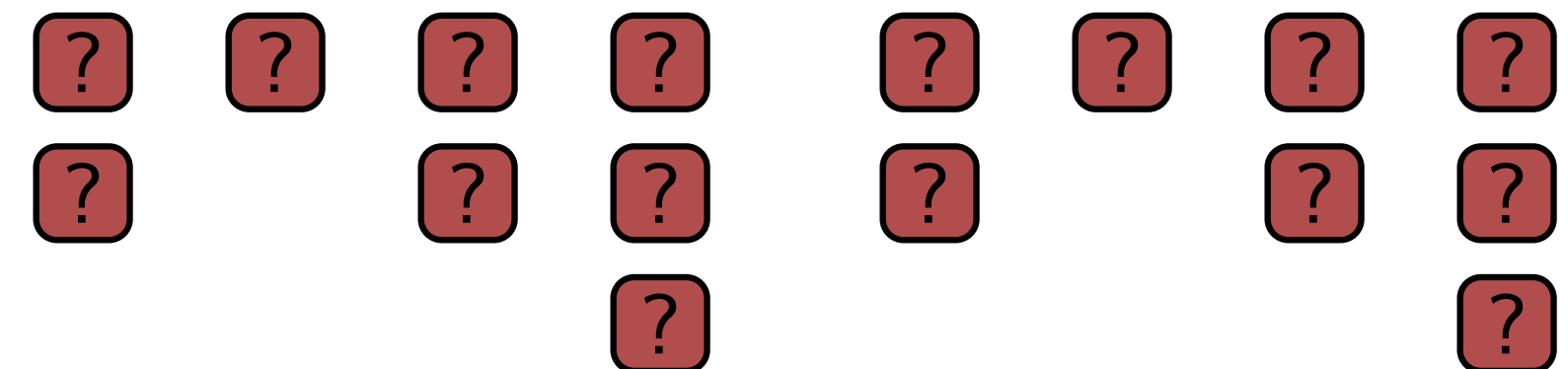


Quicksort:

- Bringe ein Objekt an die richtige Stelle
- Bringe die anderen nach links und rechts



16 Vergleiche



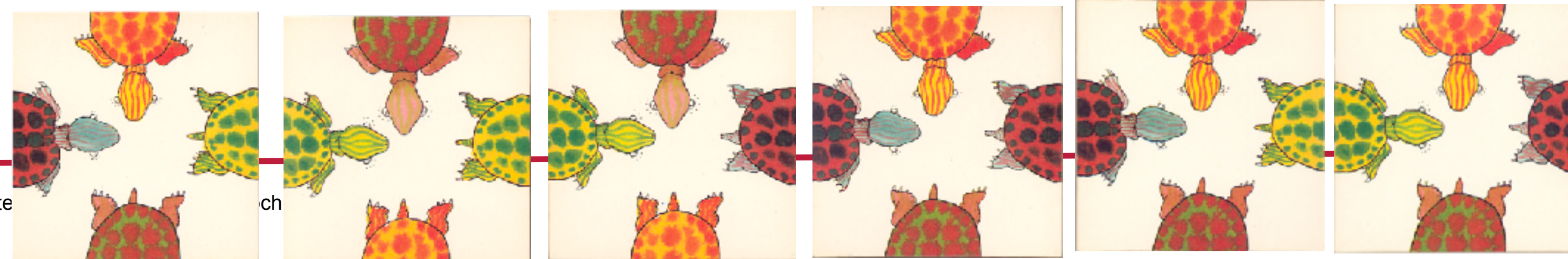
„Geht's nicht noch etwas schneller?“

3 P versus NP

Noch ein Puzzle

Gegeben: Eine Menge von n Puzzleteilen

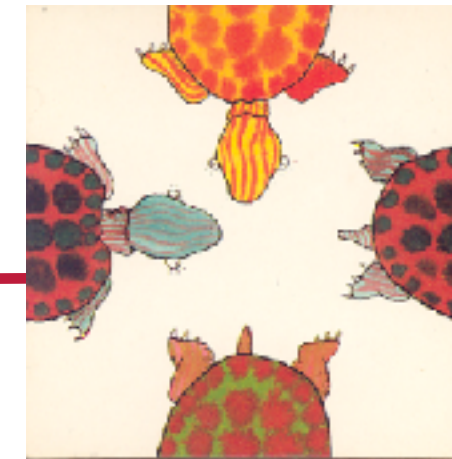
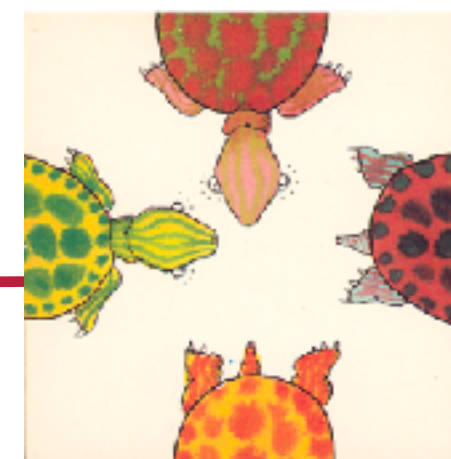
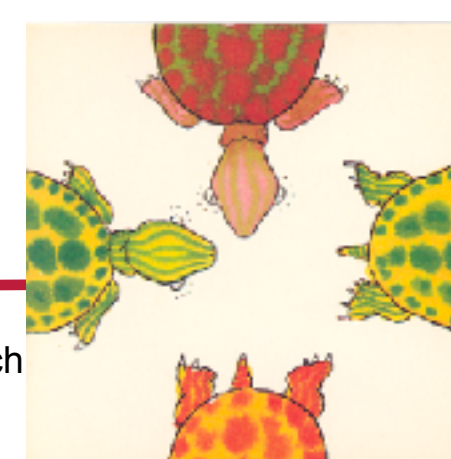
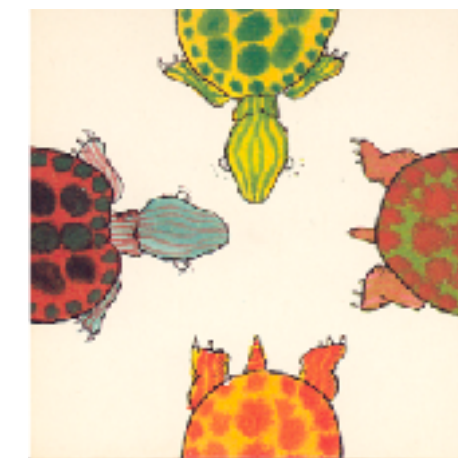
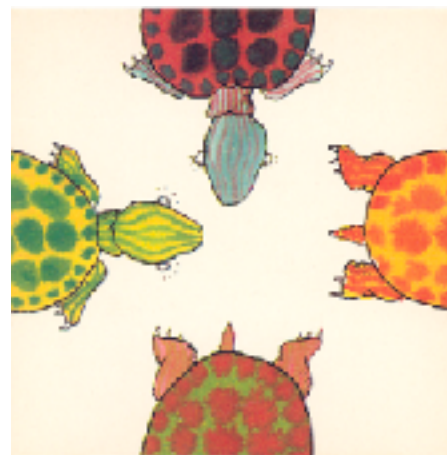
Gesucht: Eine „richtige“ Anordnung



Noch ein Puzzle

Gegeben: Eine Menge von n Puzzleteilen

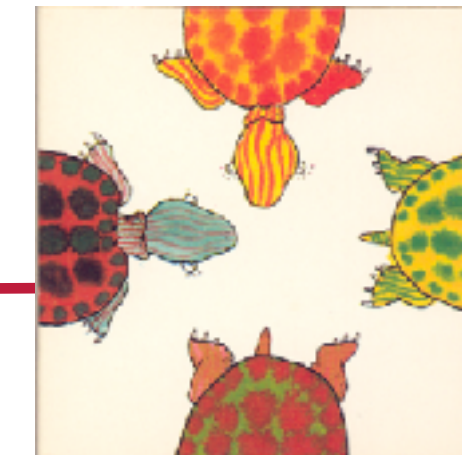
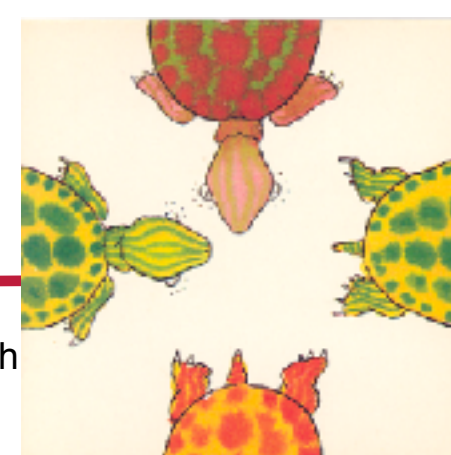
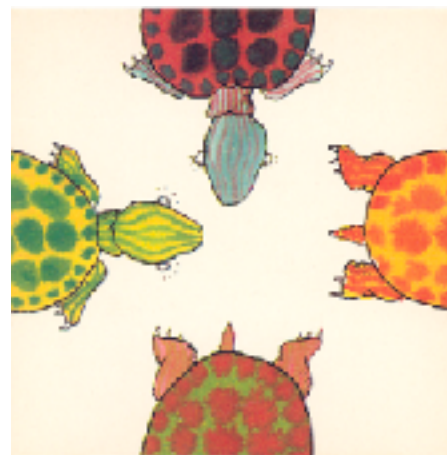
Gesucht: Eine „richtige“ Anordnung



Noch ein Puzzle

Gegeben: Eine Menge von n Puzzleteilen

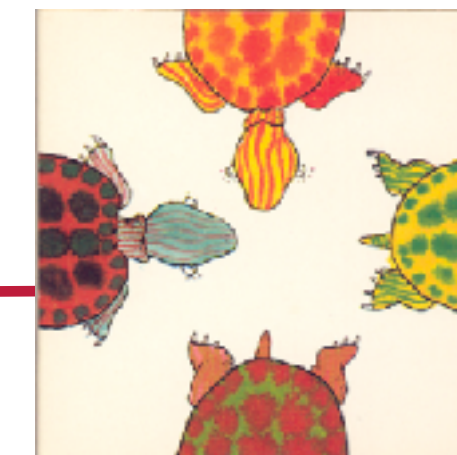
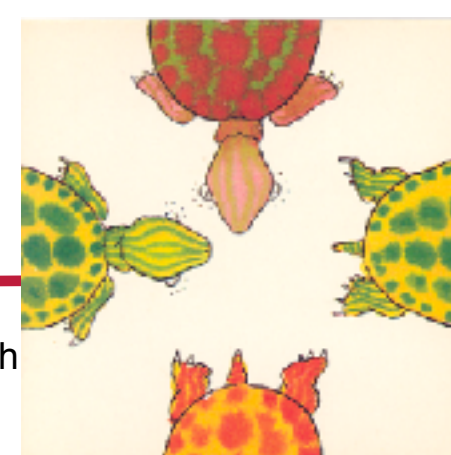
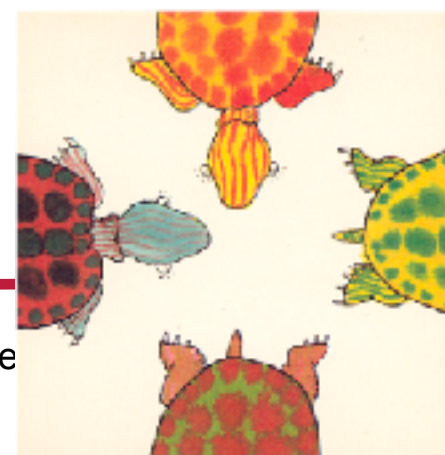
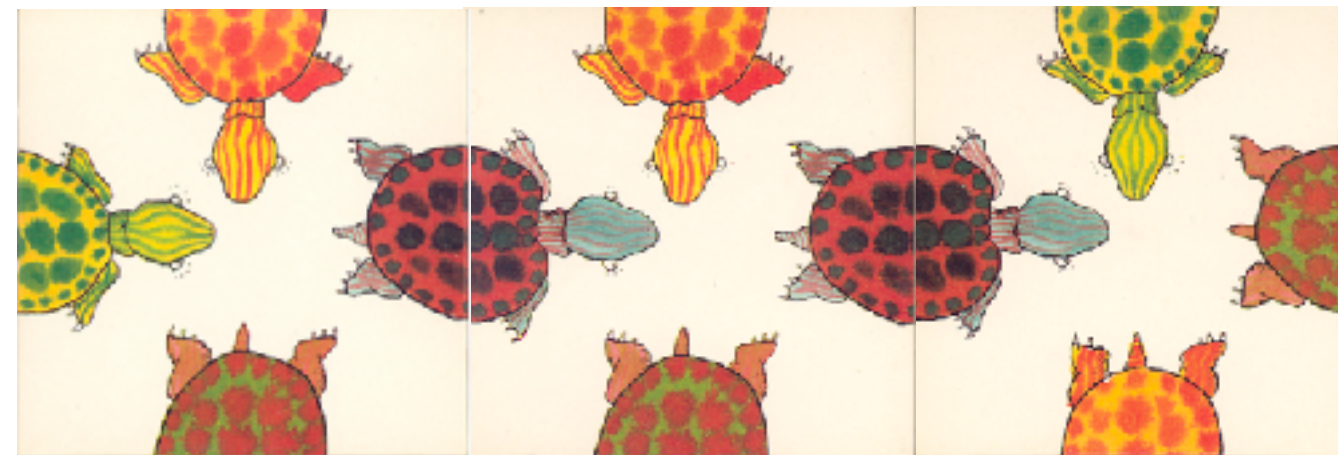
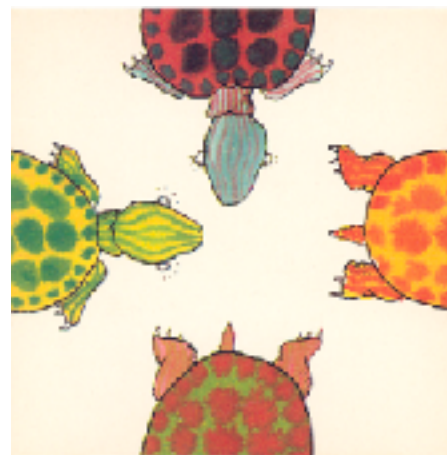
Gesucht: Eine „richtige“ Anordnung



Noch ein Puzzle

Gegeben: Eine Menge von n Puzzleteilen

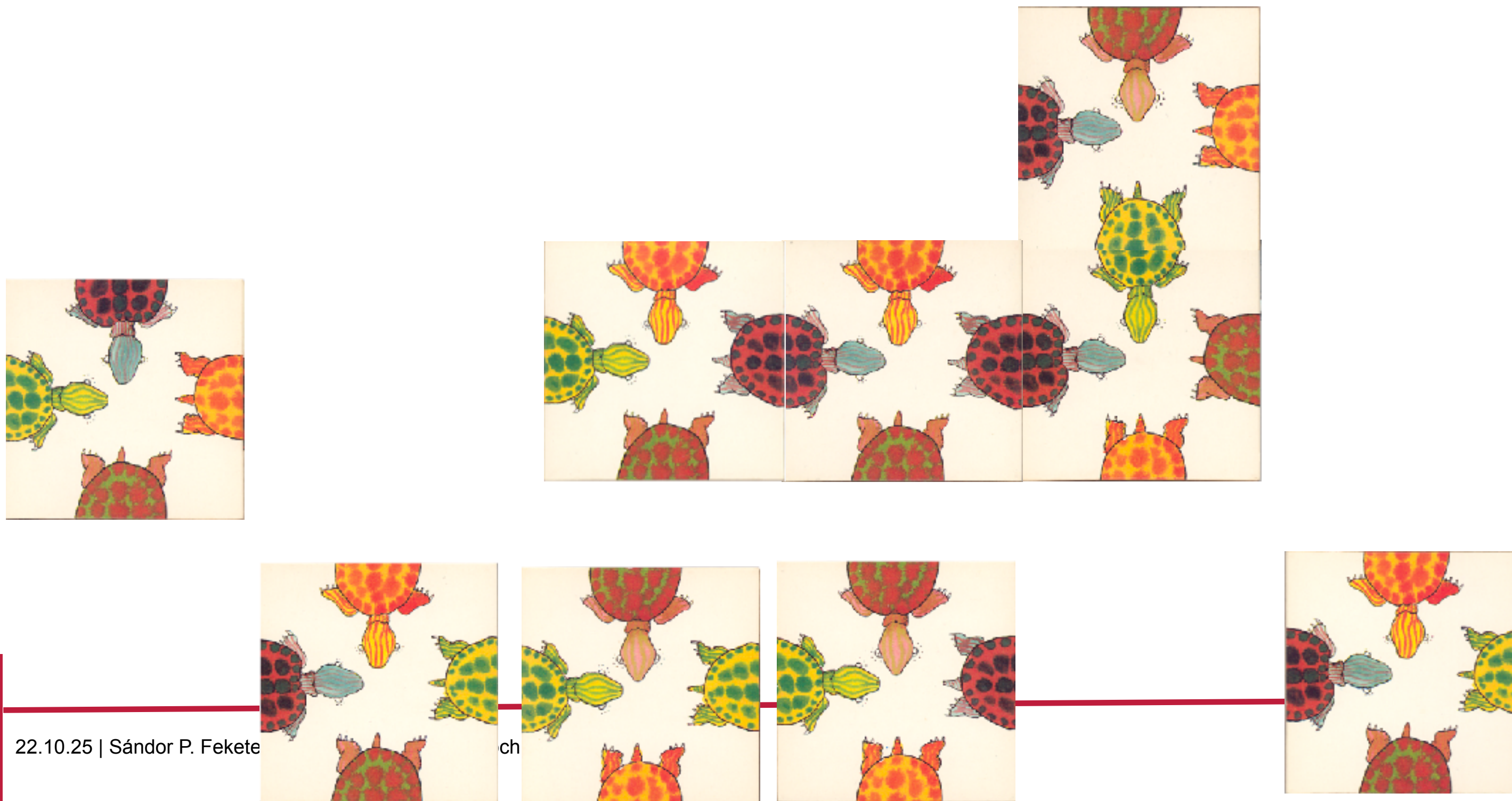
Gesucht: Eine „richtige“ Anordnung



Noch ein Puzzle

Gegeben: Eine Menge von n Puzzleteilen

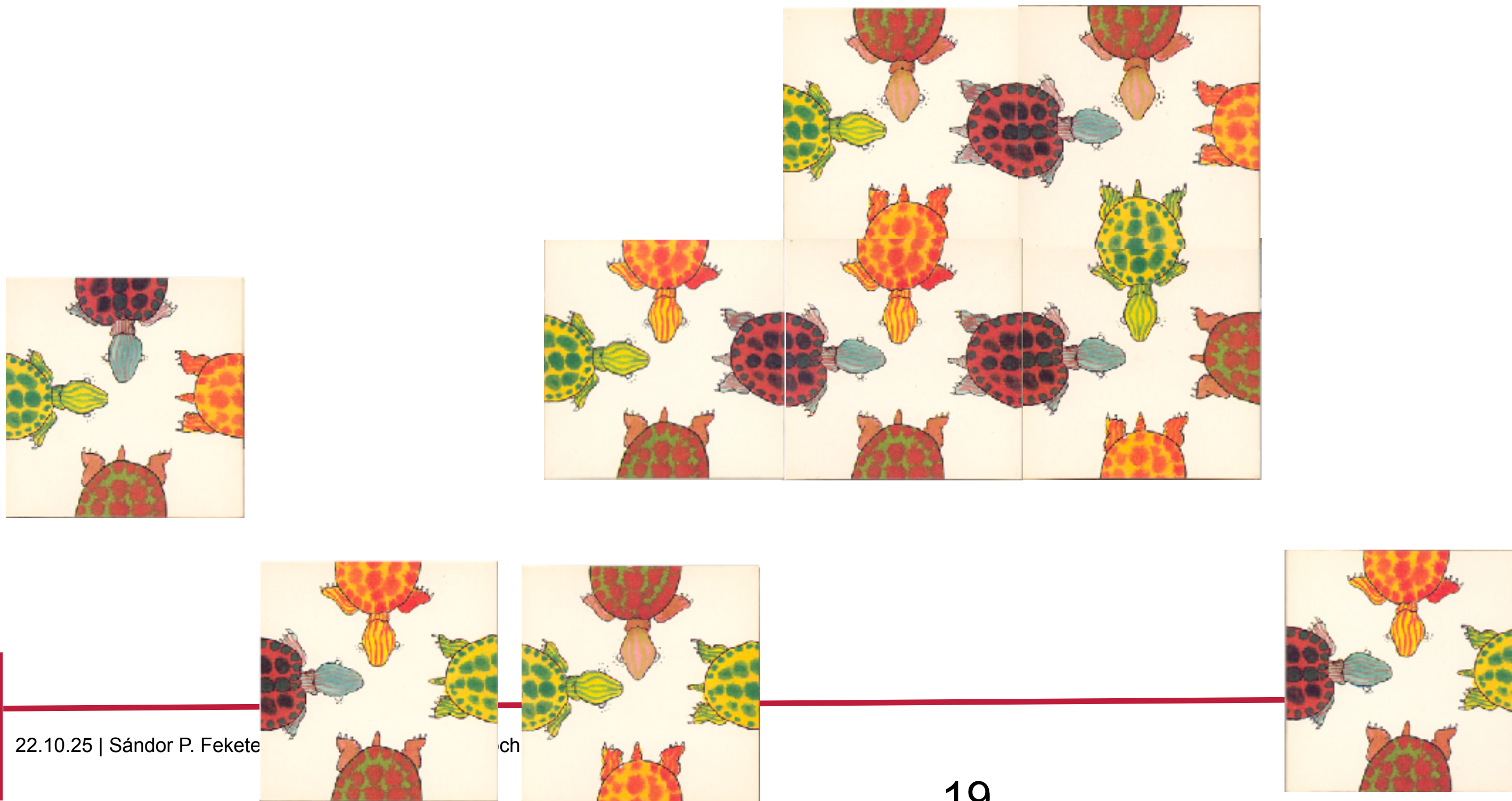
Gesucht: Eine „richtige“ Anordnung



Noch ein Puzzle

Gegeben: Eine Menge von n Puzzleteilen

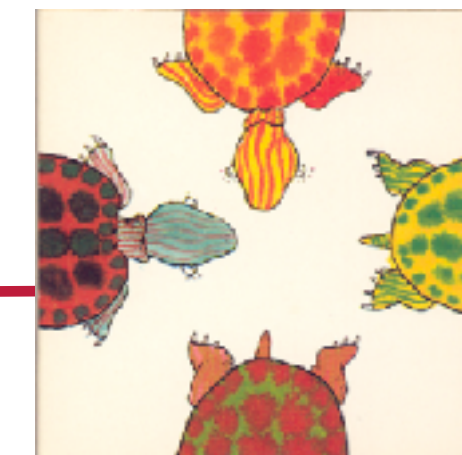
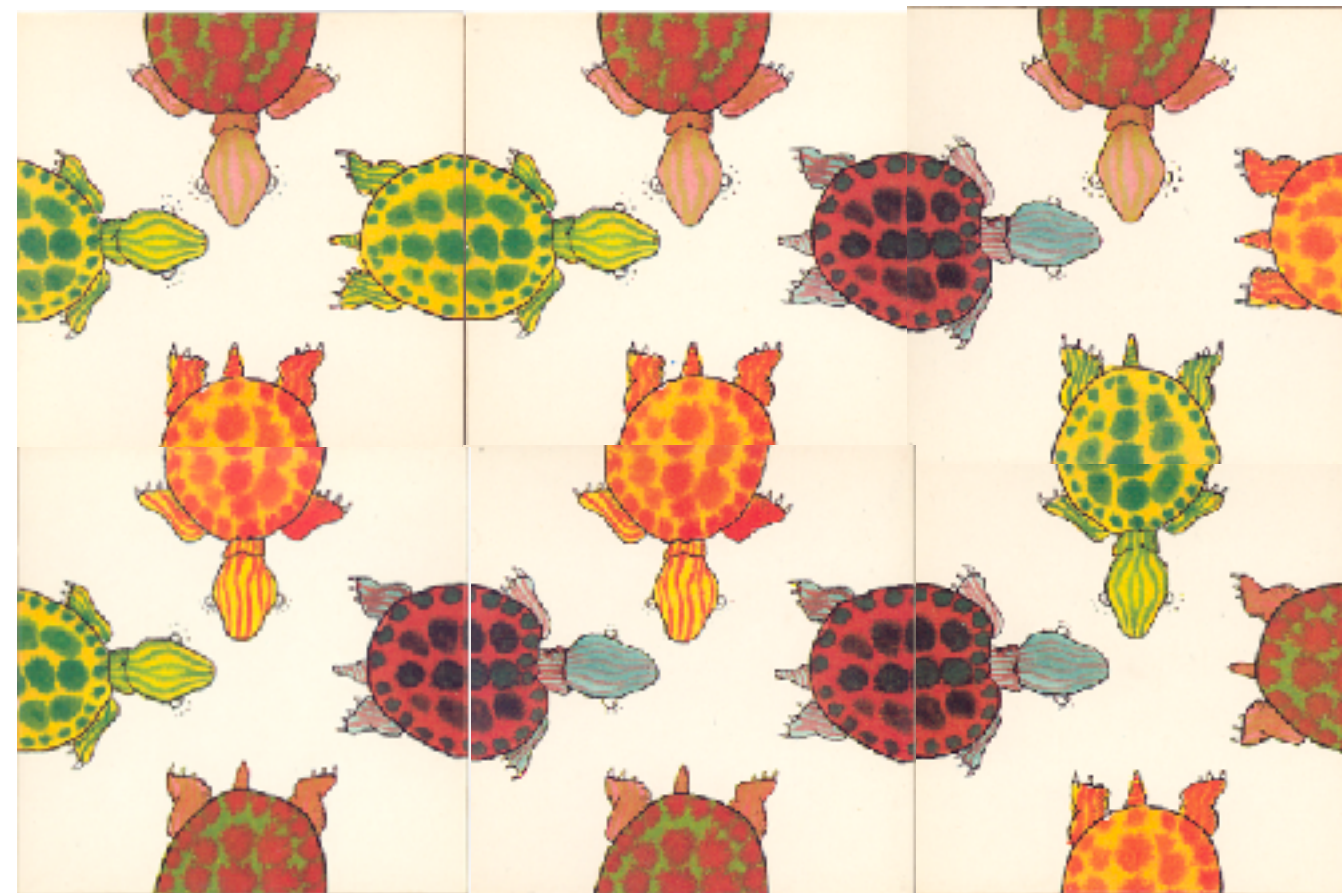
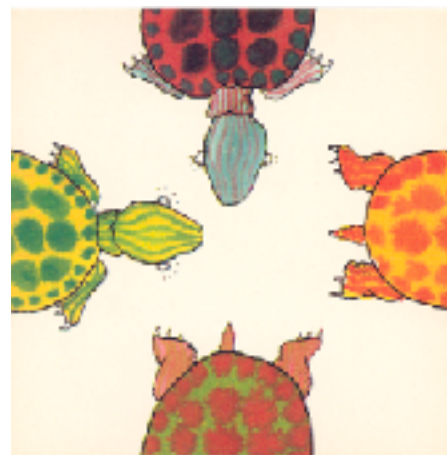
Gesucht: Eine „richtige“ Anordnung



Noch ein Puzzle

Gegeben: Eine Menge von n Puzzleteilen

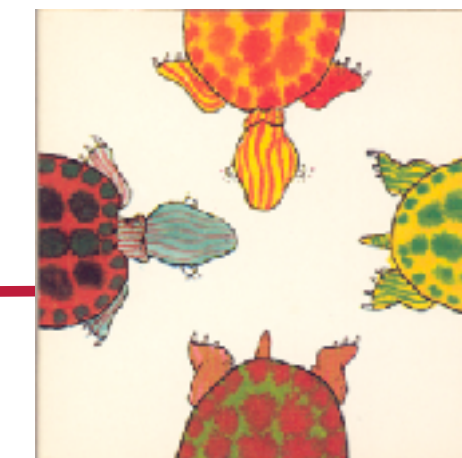
Gesucht: Eine „richtige“ Anordnung



Noch ein Puzzle

Gegeben: Eine Menge von n Puzzleteilen

Gesucht: Eine „richtige“ Anordnung



Noch ein Puzzle

Gegeben: Eine Menge von n Puzzleteilen

Gesucht: Eine „richtige“ Anordnung



Noch ein Puzzle

Gegeben: Eine Menge von n Puzzleteilen

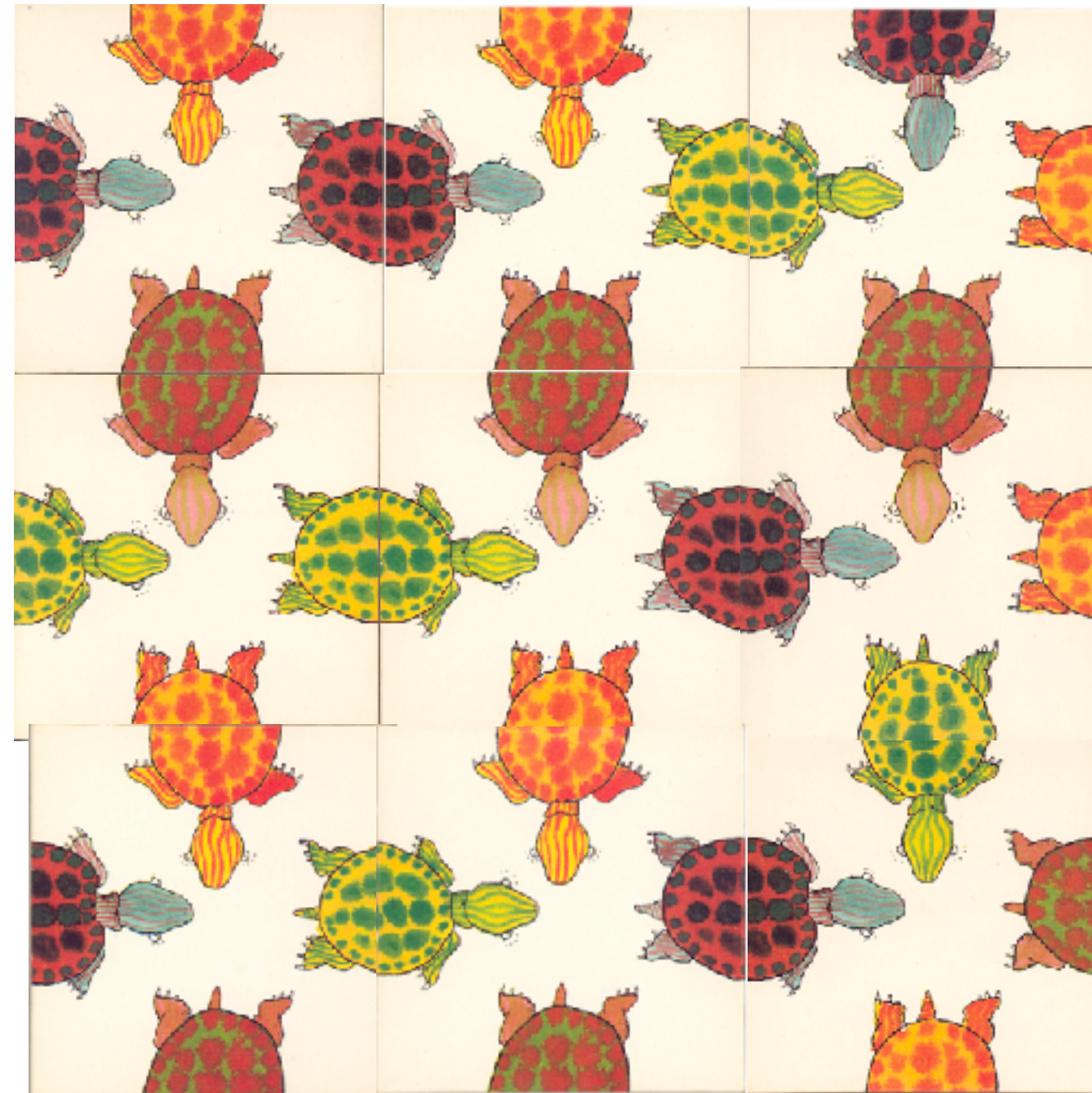
Gesucht: Eine „richtige“ Anordnung



Noch ein Puzzle

Gegeben: Eine Menge von n Puzzleteilen

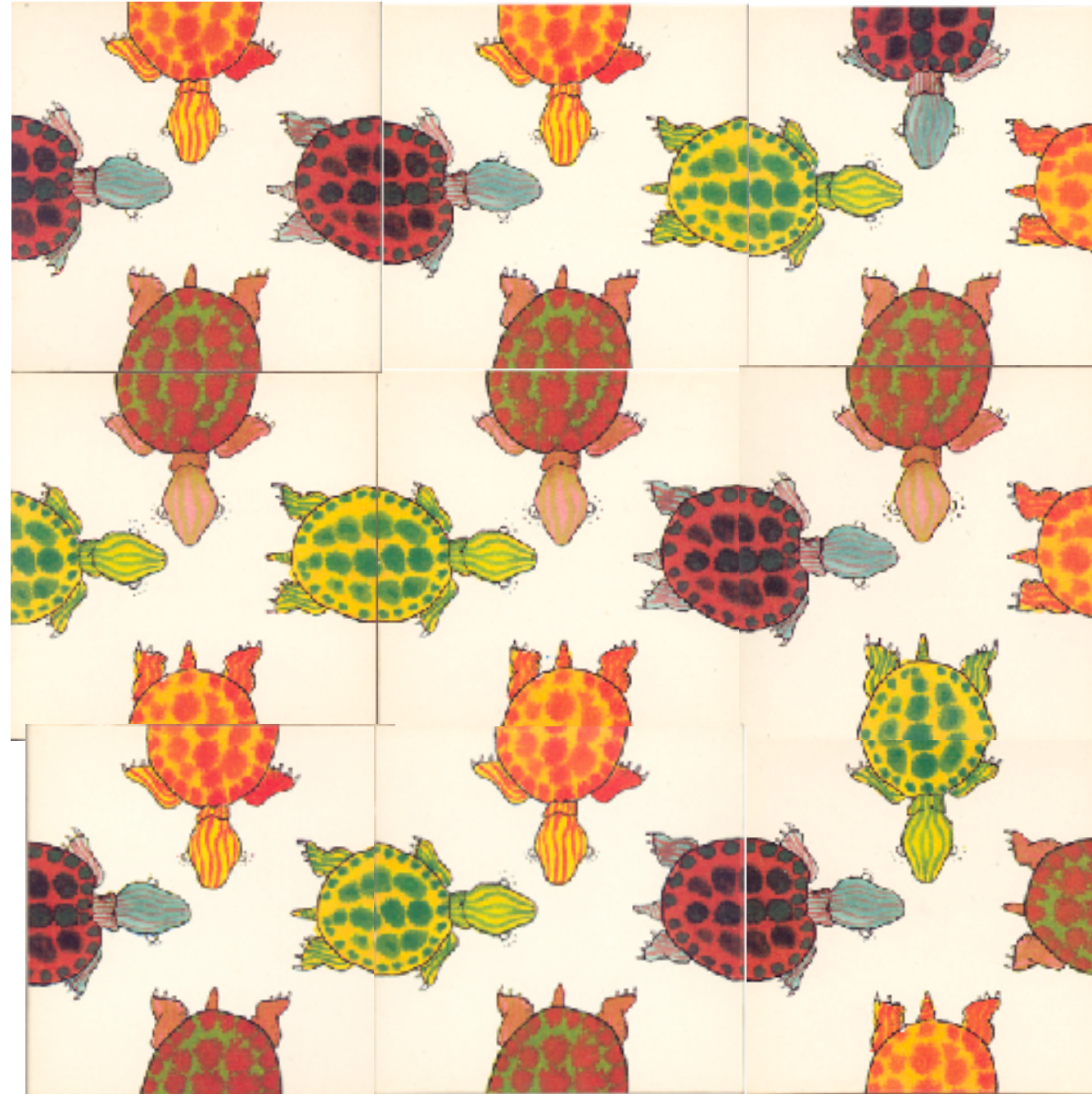
Gesucht: Eine „richtige“ Anordnung



Noch ein Puzzle

Gegeben: Eine Menge von n Puzzleteilen

Gesucht: Eine systematische Lösungsmethode



Problem: „Sackgassen“

Für $n=9$:

$9! * 4^8 =$

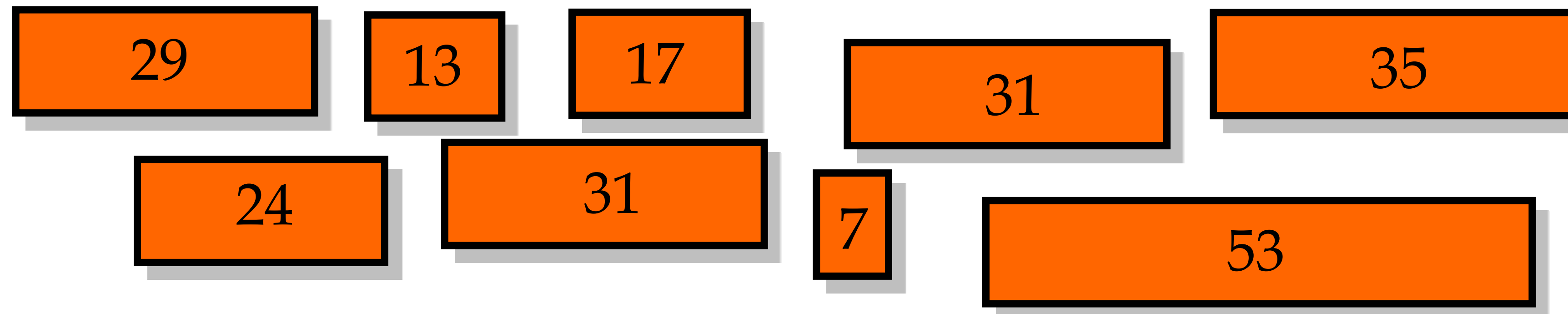
23.781.703.680

mögliche Anordnungen

Kofferpacken für die Reise

Gegeben: Zahlen {7, 13, 17, 24, 29, 31, 31, 35, 53}

Gesucht: Eine Aufteilung in zwei Teilmengen mit jeweils Summe 120



120

120

„Geht's nicht noch etwas schneller?“

Kofferpacken für die Reise

Gegeben: Zahlen {7, 13, 17, 24, 29, 31, 31, 35, 53}

Gesucht: Eine Aufteilung in zwei Teilmengen mit jeweils Summe 120

Existenz: Eine existierende Lösung kann man einfach **NachPrüfen**

Konstruktion: Wie **P**findet man eine Lösung in kurzer Zeit?

Nichtexistenz: Wie beweist man, dass es gar keine Lösung gibt?

24	13	17	31	35
29	7	31	53	

Eigenschaften des Problems

(1)

Wenn es eine Lösung gibt, so lässt sie sich ziemlich schnell *NachPrüfen*.

Die Klasse NP

(2)

Wenn es *keine* Lösung gibt, lässt sich das anscheinend nur schlecht *beweisen*.

Probleme, für die man pfiffig „schnell“ eine Lösung *Pfinden* kann:

Die Klasse P

(Echt pfundig - gepfundes Pfressen für Algorithmiker!)

$$\Theta(n^2)$$



$$\Theta(2^n)$$



Wo ist mein Schlüssel??

Ein wiedergefundener Schlüssel lässt sich relativ leicht identifizieren.



Wo ist mein Schlüssel??

Ein wiedergefundener Schlüssel lässt sich relativ leicht identifizieren.

Aber wie weist man nach, dass ein Schlüssel NICHT in einem Raum ist?

Wo ist mein Schlüssel??

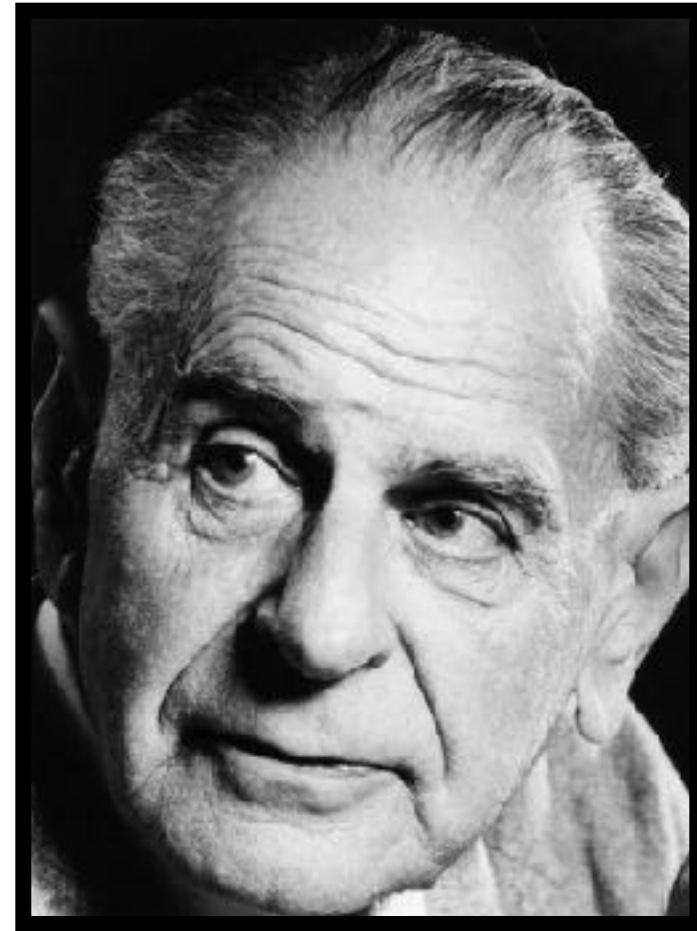
Alltagserfahrungen:

Verlieren geht
schneller als
Wiederfinden.

Ja sagen
geht schneller als
Nein begründen.

Fragen stellen
geht schneller als
Fragen beantworten.

Wo ist mein Schlüssel??



Karl Popper (1902-1994)

Universalaussagen
lassen sich nicht
verifizieren, sondern
nur falsifizieren.

Wo ist mein Schlüssel??

Ein
komplexitätstheoretischer
Glaubenssatz

$$P \neq NP$$

Also: NachPrüfen allein ist noch lange nicht Pfindig!
(Oder Pfiffig)

Schlechte Nachrichten

Satz von Cook (1973):

Es gibt schwerste
Probleme in NP,
sogenannte
„NP-vollständige
Probleme“

Schlechte Nachrichten

Noch schlechtere Nachrichten

Schlechte Nachrichten

Noch schlechtere Nachrichten

Ein polynomieller Algorithmus für

Rundreiseproblem

Partitionsproblem

die meisten interessanten
Optimierungsprobleme

bedeutet auch schon

$$P = NP$$

Schlechte Nachrichten

Noch schlechtere Nachrichten

Ein polynomieller Algorithmus für

The danger if $P = NP$

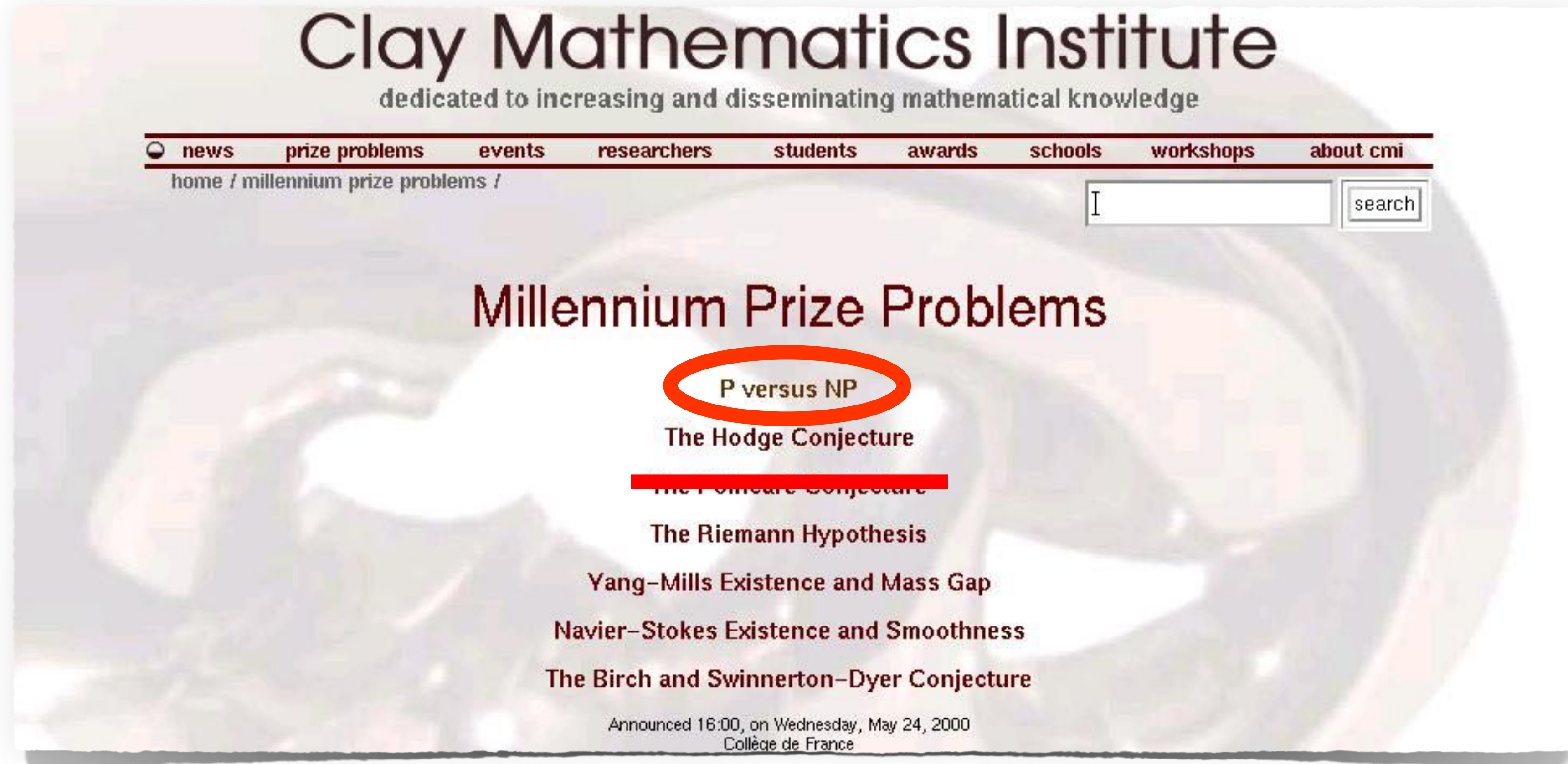
- **Breakdown of cryptography:** Modern encryption, which relies on the difficulty of problems like factoring large numbers (an NP problem), would become obsolete. This would compromise the security of online communication, financial transactions, and data storage.

Optimierungsprobleme

bedeutet auch schon

$$P = NP$$

Ein Milleniumproblem



Preisgeld: 1.000.000 \$

Geht's nicht noch etwas schneller?



Bill Cook

BC

Bill Cook <bico@uwaterloo.ca>
Aw: Next Level TSP
An: Fekete, Sandor <s.fekete@tu-braunschweig.de>,
Umgeleitet von: s.fekete@tu-braunschweig.de

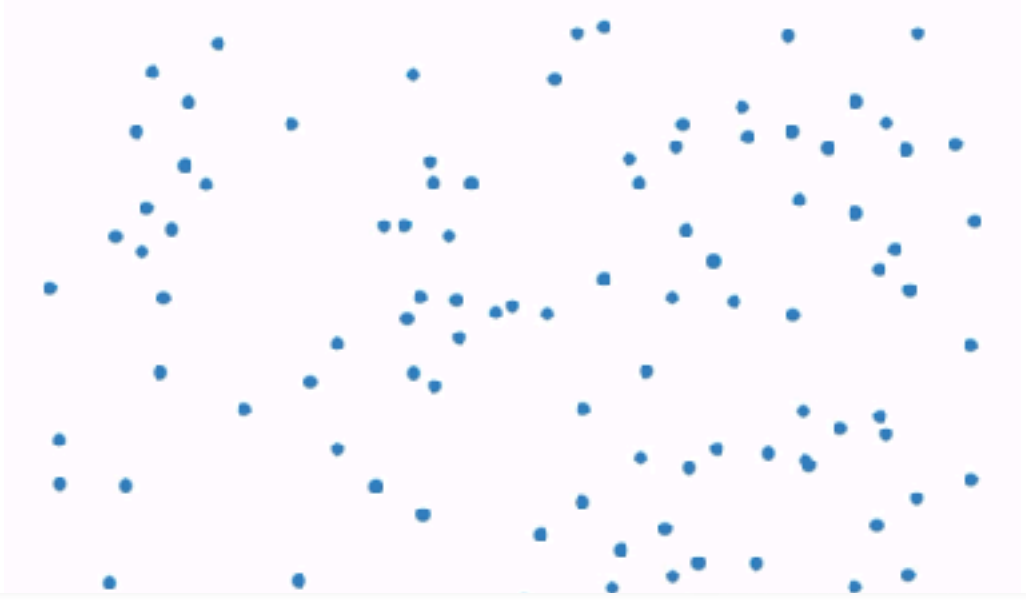
CG:SHOP 18. September 2025

Wheeeee! Thanks for the link! I'll add this to a TSP talk I'm preparing for Korea in October.

Bico

Exact Solver

Random #Nodes 200





Bill Cook @wjcook · Feb 12

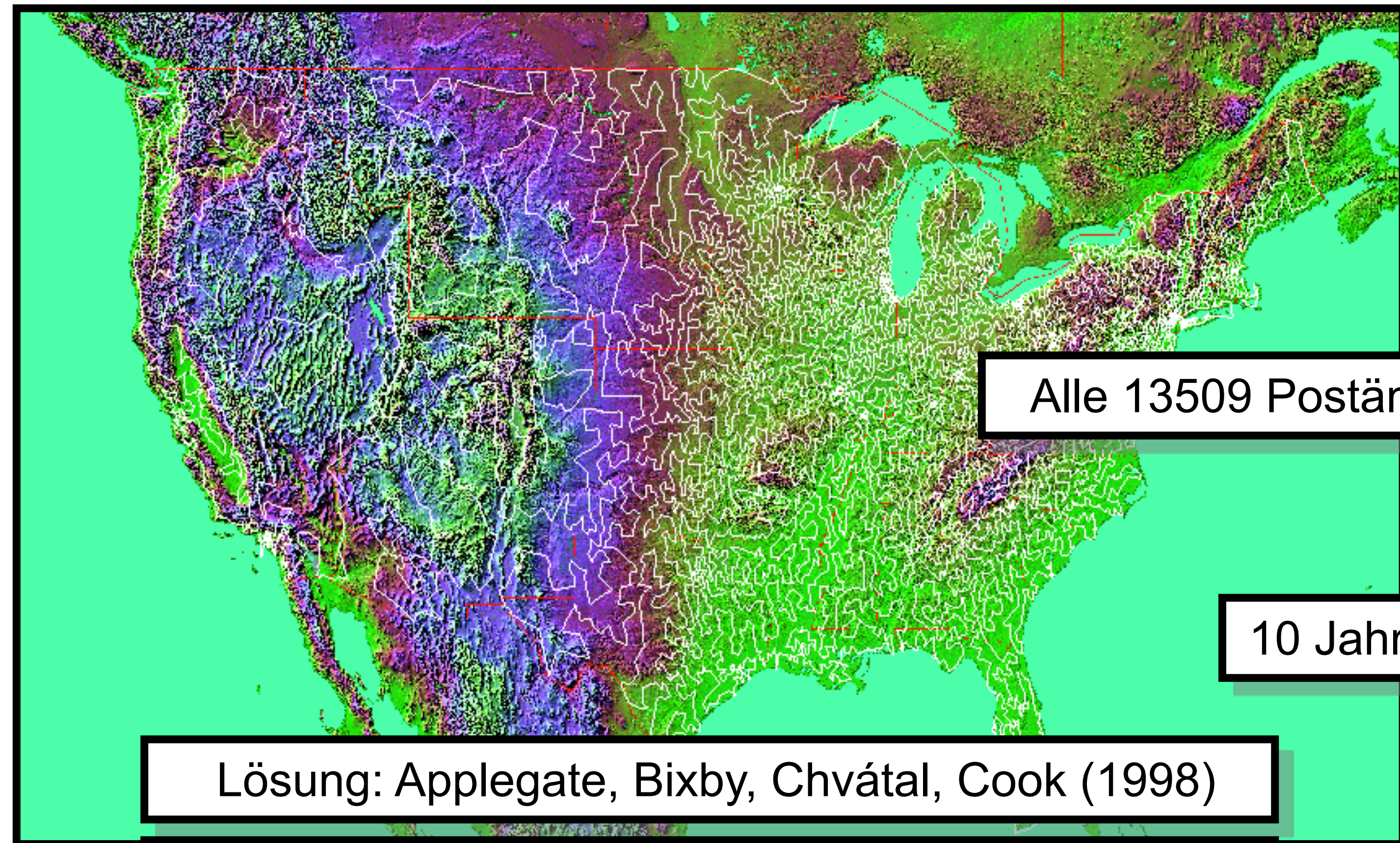
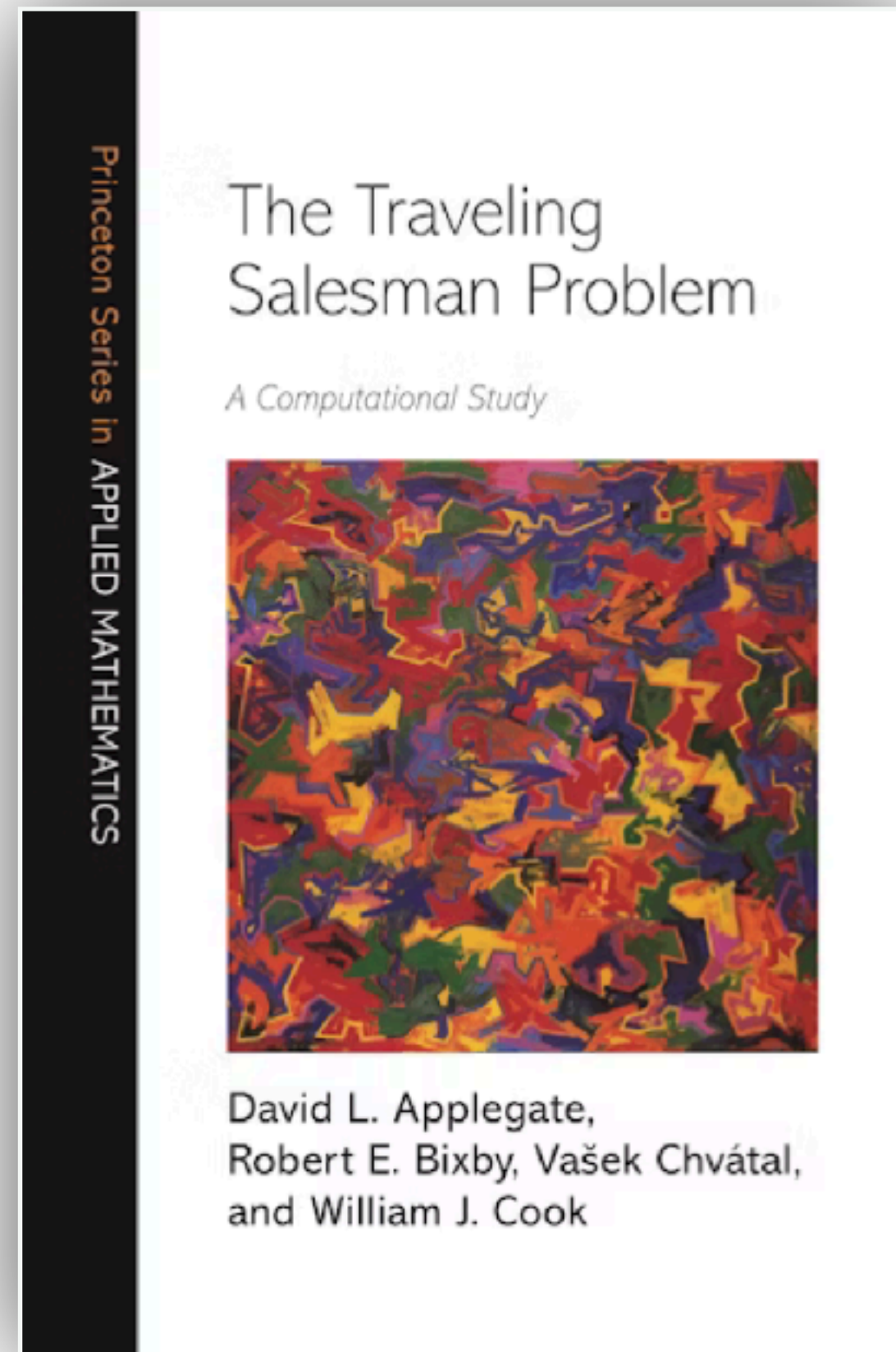
To clarify, computing an optimal TSP route on 22 points takes 0.005 seconds on my iMac, not the 1000 years reported in @washingtonpost An error factor of 6 trillion. Like reporting the US National Debt is \$4.
[washingtonpost.com/news/innovatio...](https://www.washingtonpost.com/news/innovatio...)

LK Run 4: 650380

Stop View Load

$200! \approx 10^{370}$

Eine optimale Rundreise in der Neuen Welt



Alle 13509 Postämter der USA

10 Jahre (!) CPU-Zeit

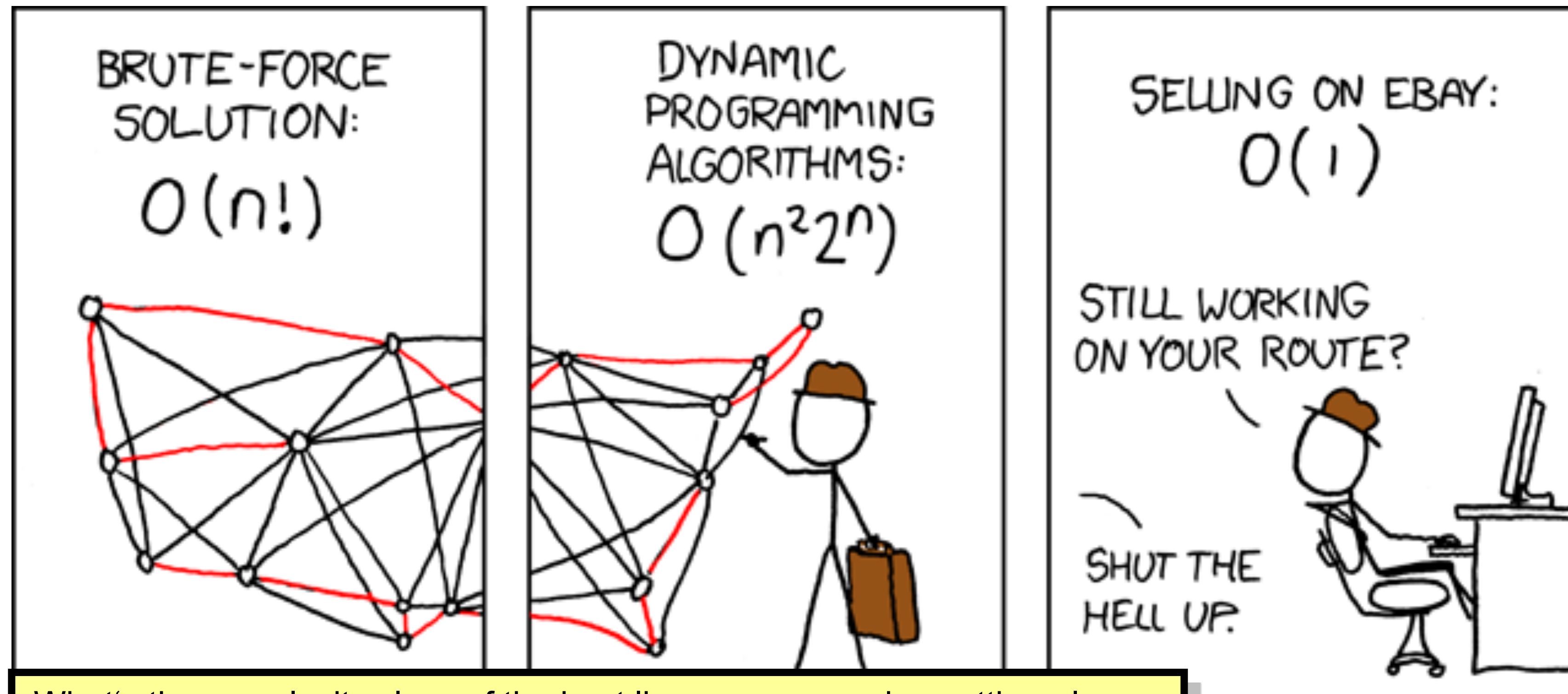
Lösung: Applegate, Bixby, Chvátal, Cook (1998)

Beale-Orchard-Hays-Preis 2000

„Geht's nicht noch etwas schneller?“

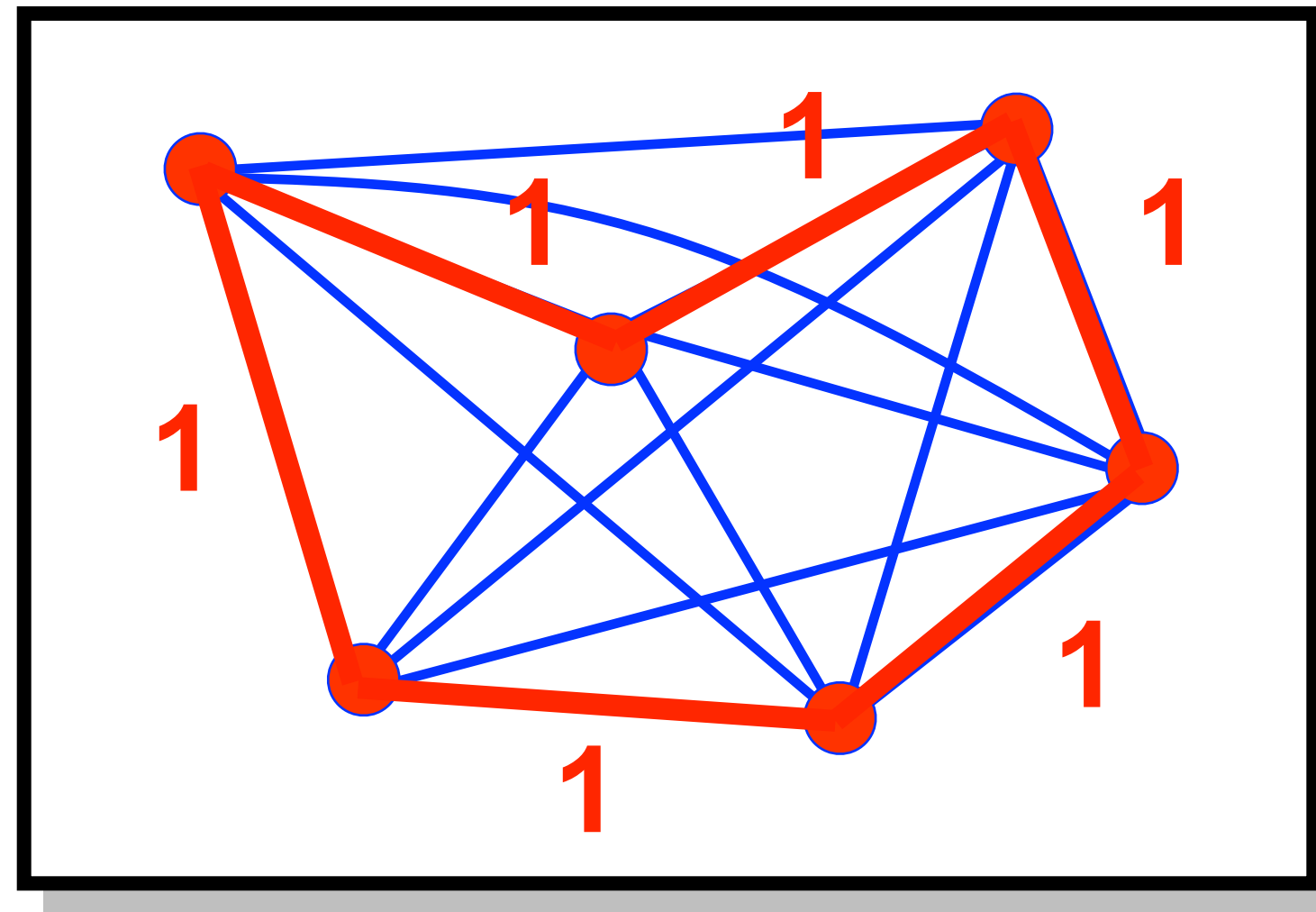
Geht's nicht noch etwas schneller?

xkcd



What's the complexity class of the best linear programming cutting-plane techniques? I couldn't find it anywhere. Man, the Garfield guy doesn't have these problems...

Von Optimierung zu Linearer Algebra



Dantzig–Fulkerson–Johnson formulation

$$x_{ij} = \begin{cases} 1 & \text{the path goes from city } i \text{ to city } j \\ 0 & \text{otherwise.} \end{cases}$$

$$\begin{aligned} \min \quad & \sum_{i=1}^n \sum_{j \neq i, j=1}^n c_{ij} x_{ij} : \\ & \sum_{i=1, i \neq j}^n x_{ij} = 1 & j = 1, \dots, n; \\ & \sum_{j=1, j \neq i}^n x_{ij} = 1 & i = 1, \dots, n; \\ & \sum_{i \in Q} \sum_{j \neq i, j \in Q} x_{ij} \leq |Q| - 1 & \forall Q \subsetneq \{1, \dots, n\}, |Q| \geq 2. \end{aligned}$$

Ganzzahlige und lineare Optimierungsprobleme

Ganzzahliges Optimierungsproblem

$$\max \sum_{i=1}^n x_i p_i$$

$$\begin{array}{ccccccc} a_{11}x_1 & + \dots & + a_{1n}x_n & \leq & b_1 \\ a_{21}x_1 & + \dots & + a_{2n}x_n & \leq & b_2 \\ \vdots & \vdots & \vdots & \vdots & \\ a_{m1}x_1 & + \dots & + a_{mn}x_n & \leq & b_m \end{array}$$

$$x_i \in \{0, 1\}$$

Beweisbar schwer zu lösen

Lineares Optimierungsproblem

$$\max \sum_{i=1}^n x_i p_i$$

$$\begin{array}{ccccccc} a_{11}x_1 & + \dots & + a_{1n}x_n & \leq & b_1 \\ a_{21}x_1 & + \dots & + a_{2n}x_n & \leq & b_2 \\ \vdots & \vdots & \vdots & \vdots & \\ a_{m1}x_1 & + \dots & + a_{mn}x_n & \leq & b_m \end{array}$$

$$x_i \in [0, 1]$$

Beweisbar effizient lösbar



Ausblicke

Algorithmen und Datenstrukturen

Semester Wintersemester 2025/2026 ▾

Studiengänge Informations-Systemtechnik Bachelor, Informatik Bachelor, Wirtschaftsinformatik Bachelor

Algorithmen und Datenstrukturen 2

Semester Sommersemester 2025 ▾

Modulnummer INF-ALG-23

Veranstaltungsnummer INF-ALG-042, INF-ALG-043, INF-ALG-044

Studiengänge Wirtschaftsinformatik Bachelor, Informations-Systemtechnik Bachelor, Informatik Bachelor



Algorithmik-Praktikum: *Solving NP-hard Problems in Practice*

Semester Wintersemester 2025/2026 ▾

Modulnummer INF-ALG-10

Studiengänge Informatik Bachelor, Wirtschaftsinformatik Bachelor

IBR Gruppe ALG (Prof. Fekete)

Art Praktikum

Mathematische Methoden der Algorithmik

Semester Wintersemester 2025/2026 ▾

Studiengänge Wirtschaftsinformatik Master, Informations-Systemtechnik Master, Informatik Master

Vielen Dank!

Algorithmen und Datenstrukturen

Semester Wintersemester 2025/2026 ▾

Studiengänge Informations-Systemtechnik Bachelor, Informatik Bachelor, Wirtschaftsinformatik Bachelor



Algorithmik-Praktikum: *Solving NP-hard Problems in Practice*

Semester Wintersemester 2025/2026 ▾

Modulnummer INF-ALG-10

Studiengänge Informatik Bachelor, Wirtschaftsinformatik Bachelor

IBR Gruppe ALG (Prof. Fekete)

Art Praktikum

Algorithmen und Datenstrukturen 2

Semester Sommersemester 2025 ▾

Modulnummer INF-ALG-23

Veranstaltungsnummer

INF-ALG-042, INF-ALG-043, INF-ALG-044

Studiengänge Wirtschaftsinformatik Bachelor, Informations-Systemtechnik Bachelor, Informatik Bachelor

Mathematische Methoden der Algorithmik

Semester Wintersemester 2025/2026 ▾

Studiengänge Wirtschaftsinformatik Master, Informations-Systemtechnik Master, Informatik Master