3.5 Aufwand: Wie lange dauert das Ausprobieren?

Leitidee

Eine alte Legende erzählt vom Erfinder des Schachspiels. Als Belohnung wünschte er sich Reis: ein Korn auf das erste Feld des Schachbretts, zwei auf das zweite, vier auf das dritte – auf jedes Feld doppelt so viele wie auf das vorige. Der König hielt das für einen bescheidenen Wunsch. Doch allein auf dem letzten Feld müssten 2^{63}, also über 9 Trillionen Körner liegen. Das ist mehr Reis, als heute auf der ganzen Welt in hundert Jahren geerntet wird. Genauso überraschend wächst die Anzahl der Möglichkeiten, die Brute-Force ausprobieren muss. In diesem Abschnitt rechnest du nach, wie schnell sie wächst, und vergleichst den **Aufwand** von Brute-Force und Heuristik.

Lernziele

Nach diesem Abschnitt kannst du

- erklären, was man anschaulich unter dem Aufwand eines Lösungsansatzes versteht,
- die Anzahl der Möglichkeiten für Brute-Force beim Handlungsreisenden ($(n-1)!$) und beim Rucksackproblem (2^n) berechnen,
- abschätzen, wie lange ein Computer für Brute-Force bei größeren Probleminstanzen brauchen würde,
- faktorielles und exponentielles Wachstum beschreiben und vergleichen,
- den Aufwand von Brute-Force und Greedy-Heuristiken miteinander vergleichen,
- begründen, warum man in der Praxis oft Heuristiken verwendet.

3.5.1 Was ist Aufwand?

Wenn du ein Zimmer aufräumst, kannst du den Aufwand auf zwei Arten angeben: „Ich musste 40 Dinge wegräumen“ oder „Ich habe 25 Minuten gebraucht“. Die erste Angabe hängt nur vom Zimmer ab. Die zweite hängt auch davon ab, wie schnell du bist.

In der Informatik ist es ähnlich. Den **Aufwand** eines Lösungsansatzes beschreibt man anschaulich so:

- durch die **Anzahl der Möglichkeiten**, die geprüft werden müssen (zum Beispiel Rundreisen oder Teilmengen), oder
- durch die **gemessene Rechenzeit** auf einem Computer.

Die Rechenzeit hängt vom Computer ab: Ein neuer Rechner ist schneller als ein alter. Die Anzahl der Möglichkeiten hängt dagegen nur von der Probleminstanz und vom Lösungsansatz ab. Deshalb rechnen wir in diesem Abschnitt vor allem mit ihr.

Definition: Aufwand

Der **Aufwand** eines Lösungsansatzes gibt an, wie viel Arbeit nötig ist, um eine Probleminstanz zu lösen. Anschaulich misst man ihn als Anzahl der zu prüfenden Möglichkeiten oder als Rechenzeit.

Merke

Der **Aufwand** eines Lösungsansatzes lässt sich anschaulich als **Anzahl der zu prüfenden Möglichkeiten** oder als **tatsächlich gemessene Rechenzeit** beschreiben.

3.5.2 Brute-Force beim Handlungsreisenden: faktorielles Wachstum

Ein Programm hat alle Rundreisen für 2 bis 8 Orte erzeugt und gezählt (Startort fest, ohne Berücksichtigung der Symmetrie). Das ist das Ergebnis:

Orte n	Anzahl der Rundreisen
2	1
3	2
4	6
5	24
6	120
7	720
8	5 040

Das sind genau die Werte von $(n-1)!$, die du im Abschnitt über Brute-Force hergeleitet hast. Bei 8 Orten ist es zum Beispiel $7! = 7 \cdot 6 \cdot 5 \cdot 4 \cdot 3 \cdot 2 \cdot 1 = 5040$.

Merke

Beim Problem des Handlungsreisenden (Brute-Force) wächst der Aufwand **faktoriell** mit der Anzahl der Städte: $(n-1)!$

Wie lange braucht ein Computer dafür? Nehmen wir einen sehr schnellen Computer an, der **eine Milliarde** ($1\,000\,000\,000$) Rundreisen pro Sekunde prüfen kann. Das ist sehr großzügig gerechnet. Dann gilt:

$$\text{Zeit in Sekunden} = \frac{\text{Anzahl der Rundreisen}}{1\,000\,000\,000}$$

Bei 8 Orten sind das $5040 \,:\, 1\,000\,000\,000 \;=\; 0{,}000\,005\,04$ Sekunden – nur ein paar Millionstel Sekunden. Kein Grund zur Sorge also? Rechne selbst weiter.

Arbeitsauftrag 52: Tabelle interpretieren

Betrachte die Tabelle mit den Anzahlen der Rundreisen für 2 bis 8 Orte.

1. Um welchen Faktor wächst die Anzahl der Rundreisen jeweils von n zu $n+1$ Orten? Beschreibe das Muster.
2. Schätze anhand des Musters (ohne die Fakultät direkt auszurechnen), wie viele Rundreisen es bei **12 Orten** gibt. Wie lange würde der Computer mit einer Milliarde Rundreisen pro Sekunde dafür brauchen?
3. Und bei **20 Orten**? Berechne die Anzahl mit dem Taschenrechner (Taste $x!$) und die Zeit in Jahren. Ein Jahr hat $365 \cdot 24 \cdot 60 \cdot 60 = 31\,536\,000$ Sekunden.

Deine Antworten:

Die folgende Tabelle setzt die Rechnung fort. Die Zeiten gelten für den Computer mit einer Milliarde Rundreisen pro Sekunde.

Orte n	Rundreisen $(n-1)!$	Rechenzeit
10	362 880	weniger als eine Tausendstelsekunde
12	39 916 800	etwa 0,04 Sekunden
15	87 178 291 200	etwa 87 Sekunden (knapp 1,5 Minuten)
18	355 687 428 096 000	etwa 4,1 Tage
20	121 645 100 408 832 000	etwa 3,9 Jahre
25	620 448 401 733 239 439 360 000	etwa 20 Millionen Jahre

Und die Symmetrie? Sie halbiert die Anzahl der Rundreisen, die man wirklich prüfen muss. Bei 25 Orten wären es dann „nur“ noch etwa 10 Millionen Jahre. Das ändert am Ergebnis nichts: Brute-Force ist für 25 Orte hoffnungslos – dabei hat ein Paketdienst oft über 100 Adressen am Tag.

Stolperstein

Ein doppelt so schneller Computer hilft kaum. Er schafft in derselben Zeit doppelt so viele Rundreisen. Aber schon **ein** zusätzlicher Ort vervielfacht die Anzahl: Von 20 auf 21 Orte wächst sie auf das Zwanzigfache. Schnellere Computer verschieben die Grenze nur um wenige Orte.

3.5.3 Brute-Force beim Rucksack: exponentielles Wachstum

Beim Rucksackproblem verdoppelt jeder zusätzliche Gegenstand die Anzahl der Teilmengen. Bei n Gegenständen sind es 2^n. Das ist das Muster aus der Schachbrett-Legende.

Merke

Beim Rucksackproblem (Brute-Force) wächst der Aufwand **exponentiell** mit der Anzahl der Gegenstände: 2^n

Arbeitsauftrag 53: Brute-Force und Greedy beim Rucksack

1. Berechne für eine wachsende Anzahl von Gegenständen die Anzahl der Teilmengen, die Brute-Force prüfen muss, und trage sie in die Tabelle ein.
2. Vergleiche gedanklich mit dem Aufwand der Greedy-Heuristik: Wie viele Gegenstände muss sie jeweils prüfen, nachdem die Liste sortiert ist?

Gegenstände n	Teilmengen 2^n (Brute-Force)	Prüfungen der Greedy-Heuristik
5		
10		
15		
18		

Gymnasium Ottweiler

Dein Vergleich:

Auch hier lohnt sich ein Blick auf größere Instanzen. Wieder gilt: eine Milliarde Teilmengen pro Sekunde.

Gegenstände n	**Teilmengen 2^n**	**Rechenzeit**
20	1 048 576	etwa eine Tausendstelsekunde
30	1 073 741 824	etwa 1 Sekunde
40	1 099 511 627 776	etwa 18 Minuten
50	1 125 899 906 842 624	etwa 13 Tage
60	1 152 921 504 606 846 976	etwa 37 Jahre

Merkst du das Muster? Zehn Gegenstände mehr bedeuten jeweils etwa das Tausendfache, denn $2^{10} = 1024$.

Faktoriell oder exponentiell – was wächst schneller?

Vergleiche die beiden Wachstumsarten für kleine Werte von n:

n	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
$(n-1)!$	1	1	2	6	24	120	720	5 040	40 320	362 880
2^n		2	4	8	16	32	64	128	256	512	1 024

Bei $n = 1$ steht in der Tabelle $(1-1)! = 0! = 1$ (zur Erinnerung: $0! = 1$ ist so festgelegt). Bis $n = 5$ ist 2^n größer. Ab $n = 6$ überholt $(n-1)!$ und lässt 2^n weit hinter sich. Der Grund: Bei 2^n wird bei jedem Schritt mit 2 multipliziert, bei der Fakultät mit einer immer größeren Zahl. Beide Wachstumsarten sind aber so schnell, dass Brute-Force schon bei mittelgroßen Instanzen unbrauchbar wird.

3.5.4 Und die Heuristiken?

Wie viel Arbeit macht die Nearest-Neighbor-Heuristik? Bei n Orten muss sie im ersten Schritt $n-1$ Entfernungen ansehen, im zweiten $n-2$ und so weiter bis 1. Bei 20 Orten sind das

$$19 + 18 + 17 + \ldots + 2 + 1 = 190$$

Entfernungen. Brute-Force bräuchte dagegen $19!$, also etwa $1{,}2 \cdot 10^{17}$ Rundreisen.

Bei der Greedy-Heuristik für den Rucksack mit 20 Gegenständen: 20 Verhältnisse berechnen, die Liste sortieren und 20-mal prüfen, ob ein Gegenstand passt. Auch mit einem einfachen Sortierverfahren sind das nur wenige hundert Schritte statt über einer Million Teilmengen.

bei $n = 20$	Brute-Force	Greedy-Heuristik
Handlungsreisender	$19! \approx 1{,}2 \cdot 10^{17}$ Rundreisen	190 Entfernungen (Nearest Neighbor)
Rucksack	$2^{20} = 1\,048\,576$ Teilmengen	wenige hundert Schritte (Wert-Gewicht-Verhältnis)

Merke

- Greedy-Heuristiken haben dagegen einen **deutlich geringeren** Aufwand. Er wächst nur langsam mit der Größe der Probleminstanz. Beim Rucksack bestimmt vor allem das **Sortieren** den Aufwand.
- Schon bei mäßig größeren Probleminstanzen wird der Unterschied zwischen Brute-Force und Heuristik **drastisch** spürbar.

Deshalb setzt man in der Praxis bei großen Probleminstanzen fast immer Heuristiken ein. Man nimmt in Kauf, dass die Lösung vielleicht nicht optimal ist – dafür bekommt man überhaupt eine Lösung, und zwar rechtzeitig.

So kannst du es sagen

So vergleichst du den Aufwand in Fachsprache:

- *„Brute-Force dauert ewig.“* → *„Der Aufwand des Brute-Force-Ansatzes wächst beim Handlungsreisenden faktoriell mit der Anzahl der Orte.“*
- *„Bei einem Gegenstand mehr wird es doppelt so viel.“* → *„Beim Rucksackproblem verdoppelt jeder zusätzliche Gegenstand die Anzahl der zu prüfenden Teilmengen; der Aufwand wächst exponentiell.“*
- *„Greedy geht schneller.“* → *„Die Greedy-Heuristik hat einen deutlich geringeren Aufwand, liefert aber nicht immer eine optimale Lösung.“*

3.5.5 Wachstum im Diagramm

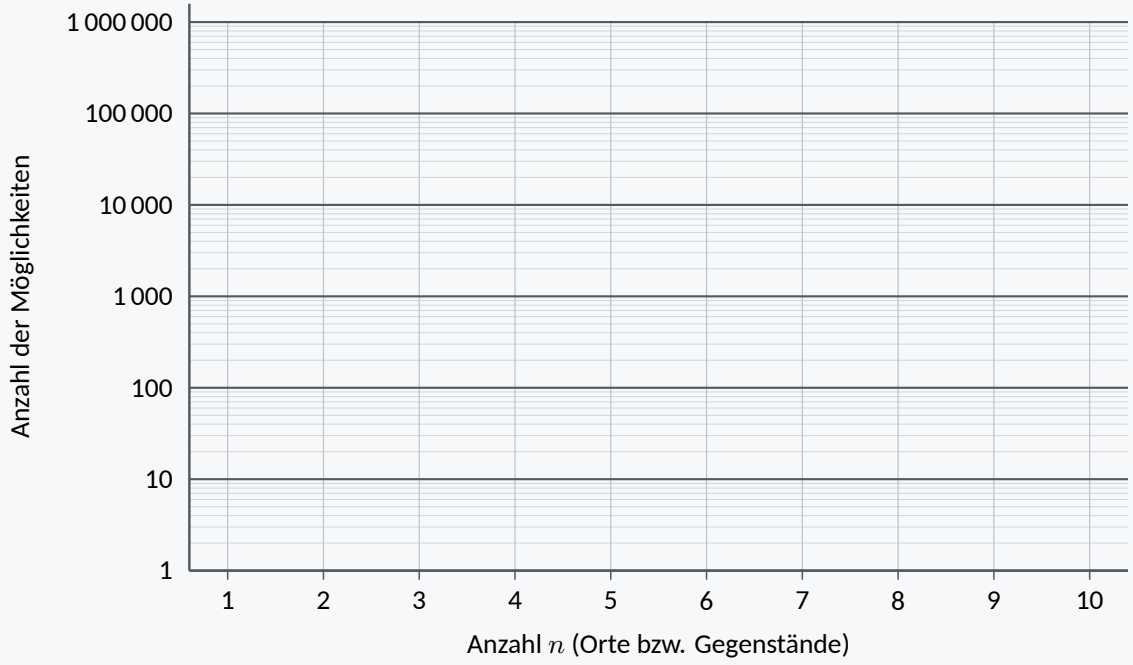
Die Zahlen werden so schnell so groß, dass sie in ein normales Diagramm nicht hineinpassen. Wenn die Hochachse bis 400 000 reicht, verschwinden alle Werte bis 1000 als Punkte auf der Rechtsachse. Man benutzt deshalb eine besondere Einteilung.

Werkzeug

Bei einer **logarithmischen Skala** bedeutet jeder Schritt auf der Hochachse **das Zehnfache**: Auf 1 folgt 10, dann 100, 1000 und so weiter. Die dünnen Linien dazwischen stehen für 2, 3, …, 9 (bzw. 20, 30, …, 90 usw.). Sie liegen nicht gleichmäßig, sondern rücken nach oben immer enger zusammen. Ein Wert wie 24 liegt also zwischen der Linie für 20 und der für 30, etwas näher an 20. So passen sehr kleine und sehr große Zahlen in ein Diagramm.

Arbeitsauftrag 54 (Zusatz): Diagramm zum Wachstum

Stelle das faktorielle Wachstum (Handlungsreisender, $(n-1)!$) und das exponentielle Wachstum (Rucksack, 2^n) für $n = 1$ bis 10 in dem Diagramm mit logarithmischer Skala gegenüber. Nimm die Werte aus der Tabelle im Abschnitt *Faktoriell oder exponentiell*. Zeichne die Punkte ein, verbinde sie und kennzeichne die beiden Kurven. Beschreibe, was du erkennst.



Leeres Diagramm mit logarithmischer Hochachse

Das erkenne ich:

3.5.6 Für Interessierte: Die Tabelle mit Python berechnen

Dieser Unterabschnitt ist freiwillig. Python kann mit beliebig großen ganzen Zahlen rechnen. Die Fakultät berechnest du, indem du bei jedem Schritt mit der nächsten Zahl multiplizierst – genau wie im Auftrag *Tabelle interpretieren*.

```
1 rundreisen = 1
2 for n in range(2, 13):
3     if n > 2:
4         rundreisen = rundreisen * (n - 1)
5     print(n, rundreisen, 2 ** n)
```

AUSGABE

```
2 1 4
3 2 8
4 6 16
5 24 32
6 120 64
7 720 128
8 5040 256
9 40320 512
10 362880 1024
11 3628800 2048
12 39916800 4096
```

- **Zeile 1:** Bei $n = 2$ Orten gibt es $(2-1)! = 1$ Rundreise. Damit beginnt die Variable.
- **Zeilen 2–4:** Für jedes weitere n wird die bisherige Anzahl mit $n - 1$ multipliziert. Aus $(n-2)!$ wird so $(n-1)!$.
- **Zeile 5:** Ausgegeben werden n, die Anzahl der Rundreisen und die Anzahl der Teilmengen 2^n (mit `**` für „hoch“).

Versuch: Weiter rechnen

Ändere `range(2, 13)` in `range(2, 26)`. Vergleiche die letzte Zeile mit der Tabelle im Abschnitt über das faktorielle Wachstum. Ergänze dann eine Ausgabe der Rechenzeit in Sekunden: `rundreisen / 1000000000`. Python schreibt sehr kleine Kommazahlen in einer Kurzform: `1e-09` bedeutet $1 \cdot 10^{-9}$, also eine Milliardstel.

3.5.7 Weitere Übungen

Hier kannst du das Gelernte weiter üben. Die Punkte im Titel zeigen, wie schwer eine Aufgabe ist: ●○○ leicht, ●●○ mittel, ●●● knifflig. Fang mit den leichten an.

Arbeitsauftrag 55: Sätze zum Aufwand ●○○

Ergänze die Sätze.

Satzanfang	Ergänzung
Den Aufwand eines Lösungsansatzes kann man angeben als Anzahl der …	
… oder als gemessene …	
Beim Handlungsreisenden prüft Brute-Force bei n Orten so viele Rundreisen: …	
Dieses Wachstum nennt man …	
Beim Rucksackproblem prüft Brute-Force bei n Gegenständen so viele Teilmengen: …	
Dieses Wachstum nennt man …	
Eine Greedy-Heuristik prüft im Vergleich zu Brute-Force …	

Arbeitsauftrag 56: Zweierpotenzen ●○○

1. Ergänze die Tabelle. Tipp: Jede Zahl ist doppelt so groß wie die davor.
2. Wie viele Teilmengen muss Brute-Force bei 12 Gegenständen prüfen, wie viele bei 13?

n	5	6	7	8	9	10	11	12
2^n	32							

Deine Antwort zu 2:

Gymnasium Ottweiler

Arbeitsauftrag 57: Rundreisen oder Teilmengen? ●○○

Entscheide bei jeder Situation, ob sie zum Problem des Handlungsreisenden oder zum Rucksackproblem passt. Gib an, wie viele Möglichkeiten Brute-Force prüfen muss, und ob das Wachstum faktoriell oder exponentiell ist.

Situation	Problem	Anzahl der Möglichkeiten	Wachstum
Ein Bäcker beliefert morgens 6 Cafés und kehrt zur Backstube zurück (7 Orte).			
Für eine Wanderung stehen 7 Gegenstände zur Auswahl, der Rucksack hat eine Tragkraft.			
Auf dem Handy ist Speicher frei. Von 10 Apps soll Lina einige installieren, die zusammen möglichst nützlich sind.			
Ein Schulbus holt Kinder an 9 Haltestellen ab und fährt zur Schule zurück, von der er gestartet ist (10 Orte).			

Arbeitsauftrag 58: Wer muss mehr prüfen? ●○○

Vergleiche den Aufwand von Brute-Force beim Handlungsreisenden mit n Orten und beim Rucksack mit n Gegenständen. Ergänze die Tabelle und gib an, bei welchem Problem Brute-Force mehr prüfen muss.

n	**Rundreisen** $(n-1)!$	**Teilmengen** 2^n	**mehr muss prüfen …**
4			
5			
6			
8			

Gymnasium Ottweiler

Das fällt mir auf:

Arbeitsauftrag 59: Wie lange rechnet der Computer? ●●○

Ein Laptop prüft **eine Million** Rundreisen pro Sekunde.

1. Wie lange braucht er für Brute-Force bei 10, 12 und 14 Orten? Nutze die Werte $9! = 362\,880$, $11! = 39\,916\,800$ und $13! = 6\,227\,020\,800$.
2. Eine Lieferfirma muss ihre Route jeden Morgen in höchstens 5 Minuten berechnen. Für wie viele Orte reicht Brute-Force auf diesem Laptop? Begründe.

Deine Rechnungen:

Arbeitsauftrag 60: Wie viele Gegenstände schafft der Laptop? ●●○

Derselbe Laptop prüft **eine Million** Teilmengen pro Sekunde.

1. Wie lange braucht er bei 20, 30 und 40 Gegenständen? Nutze $2^{20} = 1\,048\,576$, $2^{30} = 1\,073\,741\,824$ und $2^{40} = 1\,099\,511\,627\,776$.
2. Der Laptop darf **eine Stunde** rechnen. Das sind 3 600 000 000 Teilmengen. Wie viele Gegenstände kann er höchstens untersuchen? Tipp: Beginne bei 2^{30} und verdopple.

Deine Rechnungen:

Arbeitsauftrag 61: Ein zehnmal schnellerer Computer ●●●

Der Laptop aus den letzten beiden Aufträgen braucht für Brute-Force beim Handlungsreisenden mit 12 Orten etwa 40 Sekunden und beim Rucksack mit 20 Gegenständen etwa 1 Sekunde. Ein neuer Computer ist **zehnmal so schnell**.

1. Wie viele Orte kann der neue Computer in 40 Sekunden untersuchen? Begründe mit dem Faktor, um den die Anzahl der Rundreisen wächst.
2. Wie viele Gegenstände kann er in 1 Sekunde untersuchen? Begründe ebenso.
3. Was folgt daraus: Lässt sich das Problem des großen Aufwands mit schnelleren Computern lösen?

Gymnasium Ottweiler

Deine Antworten:

3.5.8 Das Wichtigste in Kürze

- Der **Aufwand** eines Lösungsansatzes wird anschaulich als Anzahl der zu prüfenden Möglichkeiten oder als Rechenzeit angegeben.
- Brute-Force beim **Handlungsreisenden** prüft $(n-1)!$ Rundreisen: **faktorielles** Wachstum. Von n auf $n+1$ Orte wächst die Anzahl um den Faktor n.
- Brute-Force beim **Rucksackproblem** prüft 2^n Teilmengen: **exponentielles** Wachstum. Jeder weitere Gegenstand verdoppelt die Anzahl.
- Selbst bei einer Milliarde Möglichkeiten pro Sekunde dauert Brute-Force bei 20 Orten etwa 3,9 Jahre und bei 25 Orten etwa 20 Millionen Jahre.
- **Greedy-Heuristiken** haben einen viel geringeren Aufwand, der nur langsam wächst (z. B. 190 Entfernungen bei Nearest Neighbor mit 20 Orten).
- Deshalb verwendet man bei großen Probleminstanzen Heuristiken – auch wenn ihre Lösung nicht immer optimal ist.

Selbstcheck

- ☐ Ich kann erklären, was man unter dem Aufwand eines Lösungsansatzes versteht.
- ☐ Ich kann die Anzahl der Rundreisen $(n-1)!$ und der Teilmengen 2^n für ein gegebenes n berechnen.
- ☐ Ich kann abschätzen, wie lange ein Computer für Brute-Force braucht.
- ☐ Ich kann faktorielles und exponentielles Wachstum beschreiben und vergleichen.
- ☐ Ich kann den Aufwand von Brute-Force und Greedy-Heuristik an einem Beispiel vergleichen.
- ☐ Ich kann begründen, warum man in der Praxis oft Heuristiken einsetzt.