$\mathcal{A} \Rightarrow \mathcal{B}$ „$\mathcal{A}$ hinreichend“

$\mathcal{B} \Rightarrow \mathcal{A}$ „$\mathcal{A}$ notwendig“

Äquivalente Formeln $\Leftrightarrow$		Bezeichnung
$A \wedge B$	$B \wedge A$	Kommutativ
$A \vee B$	$B \vee A$	
$A \wedge (B \wedge C)$	$(A \wedge B) \wedge C$	Assoziativ
$A \vee (B \vee C)$	$(A \vee B) \vee C$	
$A \wedge (B \vee C)$	$(A \wedge B) \vee (A \wedge C)$	Distributiv
$A \vee (B \wedge C)$	$(A \vee B) \wedge (A \vee C)$	
$A \wedge A$	A	Idempotenz
$A \vee A$	A	
$\neg\neg A$	A	Involution
$\neg(A \wedge B)$	$\neg A \vee \neg B$	De-Morgan
$\neg(A \vee B)$	$\neg A \wedge \neg B$	
$A \wedge (A \vee B)$	A	Absorption
$A \vee (A \wedge B)$	A	
$A \Rightarrow B$	$\neg A \vee B$	Elimination
$\neg(A \Rightarrow B)$	$A \wedge \neg B$	
$A \Leftrightarrow B$	$(A \Rightarrow B) \wedge (B \Rightarrow A)$	

$\mathcal{A} \Rightarrow \mathcal{B}$ „$\mathcal{A}$ hinreichend“

$\mathcal{B} \Rightarrow \mathcal{A}$ „$\mathcal{A}$ notwendig“

Äquivalente Formeln $\Leftrightarrow$		Bezeichnung
$A \wedge B$	$B \wedge A$	Kommutativ
$A \vee B$	$B \vee A$	
$A \wedge (B \wedge C)$	$(A \wedge B) \wedge C$	Assoziativ
$A \vee (B \vee C)$	$(A \vee B) \vee C$	
$A \wedge (B \vee C)$	$(A \wedge B) \vee (A \wedge C)$	Distributiv
$A \vee (B \wedge C)$	$(A \vee B) \wedge (A \vee C)$	
$A \wedge A$	A	Idempotenz
$A \vee A$	A	
$\neg\neg A$	A	Involution
$\neg(A \wedge B)$	$\neg A \vee \neg B$	De-Morgan
$\neg(A \vee B)$	$\neg A \wedge \neg B$	
$A \wedge (A \vee B)$	A	Absorption
$A \vee (A \wedge B)$	A	
$A \Rightarrow B$	$\neg A \vee B$	Elimination
$\neg(A \Rightarrow B)$	$A \wedge \neg B$	
$A \Leftrightarrow B$	$(A \Rightarrow B) \wedge (B \Rightarrow A)$	

$\mathcal{A} \Rightarrow \mathcal{B}$ „$\mathcal{A}$ hinreichend“

$\mathcal{B} \Rightarrow \mathcal{A}$ „$\mathcal{A}$ notwendig“

Äquivalente Formeln $\Leftrightarrow$		Bezeichnung
$A \wedge B$	$B \wedge A$	Kommutativ
$A \vee B$	$B \vee A$	
$A \wedge (B \wedge C)$	$(A \wedge B) \wedge C$	Assoziativ
$A \vee (B \vee C)$	$(A \vee B) \vee C$	
$A \wedge (B \vee C)$	$(A \wedge B) \vee (A \wedge C)$	Distributiv
$A \vee (B \wedge C)$	$(A \vee B) \wedge (A \vee C)$	
$A \wedge A$	A	Idempotenz
$A \vee A$	A	
$\neg\neg A$	A	Involution
$\neg(A \wedge B)$	$\neg A \vee \neg B$	De-Morgan
$\neg(A \vee B)$	$\neg A \wedge \neg B$	
$A \wedge (A \vee B)$	A	Absorption
$A \vee (A \wedge B)$	A	
$A \Rightarrow B$	$\neg A \vee B$	Elimination
$\neg(A \Rightarrow B)$	$A \wedge \neg B$	
$A \Leftrightarrow B$	$(A \Rightarrow B) \wedge (B \Rightarrow A)$	

Groß-O-Notation Kosten $C_f(n)$ mit $g : \mathbb{N} \to \mathbb{R} \exists c > 0 \exists n_0 > 0 \forall n \geq n_0$

Untere Schranke $\Omega(f)$
$C_f(n) \geq c * g(n)$

Obere Schranke $O(f)$
$C_f(n) \leq c * g(n)$

Exakte Schranke $\Theta(f)$
$C_f(n) \in \Omega(f) \cap O(f)$
Polynom $k$ten Grades $\in \Theta(n^k)$

(Beweis: g und c finden)

Elementare Operationen, Kontrollstr. $\in \mathbf{O(1)}$

Schleifen $\in i$ Wiederholungen $*$ $O(f)$ teuerste Operation

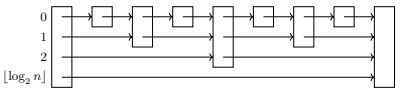
Abfolge $O(g)$ nach $O(f) \in O(\max(f; g))$

Rekursion $\in k$ Aufrufe $*$ $O(f)$ teuerste Operation

Mastertheorem $a \geq 1$, $b > 1$, $\Theta \geq 0$

$$T(n) = a * T(\frac{n}{b}) + \Theta(n^k)$$
$$\Rightarrow \begin{cases} \Theta(n^k) & a < b^k \\ \Theta(n^k \log n) & a = b^k \\ \Theta(n^{\log_b a}) & a > b^k \end{cases}$$

Skip



- Zeiger auf Ebene i zeigt zu nächstem 2^i Element
- Suchen $\in O(\log n)$

(Perfekt) Einfügen, Löschen $\in \mathbf{O(n)}$ (Vollst. Reorga.)

Randomisiert Höhe zufällig (keine vollst. Reorga.)
$P(h) = \frac{1}{2^{h+1}}$: Einfügen, Löschen $\in \mathbf{O(\log n)}$

Sortierproblem

Gegeben (endliche) Folge von Schlüsseln (von Daten) $(K_i)_{i \in I}$

Gesucht Bijektive Abbildung $\pi : I \to I$ (Permutation), sodass $K_{\pi(i)} \leq K_{\pi(i+1)} \quad \forall i \in I$

Ordnung *Allgemein* vs. *speziell*: Ordnung wird nur über Schlüsselvergleiche hergestellt

Relation *Stabil* vs. *instabil*: Vorherig relative Reihenfolge bleibt erhalten

Speicher *In situ* vs. *ex situ*: Zusätzlicher Speicher notwendig

Lokal *Intern* vs. *extern*: Alles im RAM oder Mischung vorsortierter externer Teilfolgen

Anzahl der Inversionen Anzahl kleinerer Nachfolger für jedes Element:

$$\begin{aligned} \text{inv}(L) := |\{(i,j) \mid \\ 0 \leq i < j \leq n-1, \\ L[i] \geq L[j]\}| \end{aligned}$$

Anzahl der Runs Ein *Run* ist eine sortierte Teilliste, die nicht nach links oder rechts verlängert werden kann. Die Anzahl der Runs ist:

$$\begin{aligned} \text{runs}(L) := |\{i \mid \\ 0 \leq i < n-1, \\ L[i+1] < L[i]\}| + 1 \end{aligned}$$

Längster Run Anzahl der Elemente der längsten sortierten Teilliste:

$$\begin{aligned} \text{las}(L) := \max\{r.\text{len} \mid \\ r \text{ ist Run in } L\} \\ \text{rem}(L) := L.\text{len} - \text{las}(L) \end{aligned}$$

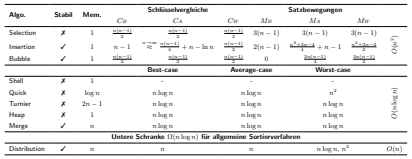
Jedes allgemeine Sortierverfahren benötigt im Worst- und Average-case Schlüsselvergleiche von mindestens:

$$\Omega(n \log n)$$

Lexikographische Ordnung $\leq$ Sei $A = \{a_1, \ldots, a_n\}$ ein Alphabet, dass sich mit gegebener Ordnung $a_1 < \cdots < a_n$ wie folgt auf dem Lexikon $A* = \bigcup_{n \in \mathbb{N}_0} A^n$ fortsetzt:

$$\begin{aligned} & v = (v_1, \ldots, v_p) \leq w = (w_1, \ldots, w_q) \\ \Leftrightarrow & \forall 1 \leq i \leq p : v_i = w_i \quad p \leq q \\ \vee & \forall 1 \leq j \leq i : v_j = w_j \quad v_i < w_i \end{aligned}$$

Fachverteilen Sortieren von n k-Tupeln in k Schritten: Sortieren nach letztem Element, vorletzem usw.



Groß-O-Notation Kosten $C_f(n)$ mit $g : \mathbb{N} \to \mathbb{R} \exists c > 0 \exists n_0 > 0 \forall n \geq n_0$

Untere Schranke $\Omega(f)$
$C_f(n) \geq c * g(n)$

Obere Schranke $O(f)$
$C_f(n) \leq c * g(n)$

Exakte Schranke $\Theta(f)$
$C_f(n) \in \Omega(f) \cap O(f)$
Polynom $k$ten Grades $\in \Theta(n^k)$

(Beweis: g und c finden)

Elementare Operationen, Kontrollstr. $\in \mathbf{O(1)}$

Schleifen $\in i$ Wiederholungen $*$ $O(f)$ teuerste Operation

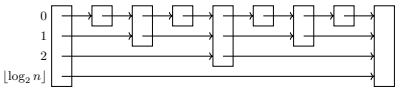
Abfolge $O(g)$ nach $O(f) \in O(\max(f; g))$

Rekursion $\in k$ Aufrufe $*$ $O(f)$ teuerste Operation

Mastertheorem $a \geq 1$, $b > 1$, $\Theta \geq 0$

$$T(n) = a * T(\frac{n}{b}) + \Theta(n^k)$$
$$\Rightarrow \begin{cases} \Theta(n^k) & a < b^k \\ \Theta(n^k \log n) & a = b^k \\ \Theta(n^{\log_b a}) & a > b^k \end{cases}$$

Skip



- Zeiger auf Ebene i zeigt zu nächstem 2^i Element
- Suchen $\in O(\log n)$

(Perfekt) Einfügen, Löschen $\in \mathbf{O(n)}$ (Vollst. Reorga.)

Randomisiert Höhe zufällig (keine vollst. Reorga.)
$P(h) = \frac{1}{2^{h+1}}$: Einfügen, Löschen $\in \mathbf{O(\log n)}$

Sortierproblem

Gegeben (endliche) Folge von Schlüsseln (von Daten) $(K_i)_{i \in I}$

Gesucht Bijektive Abbildung $\pi : I \to I$ (Permutation), sodass $K_{\pi(i)} \leq K_{\pi(i+1)} \quad \forall i \in I$

Ordnung *Allgemein* vs. *speziell*: Ordnung wird nur über Schlüsselvergleiche hergestellt

Relation *Stabil* vs. *instabil*: Vorherig relative Reihenfolge bleibt erhalten

Speicher *In situ* vs. *ex situ*: Zusätzlicher Speicher notwendig

Lokal *Intern* vs. *extern*: Alles im RAM oder Mischung vorsortierter externer Teilfolgen

Anzahl der Inversionen Anzahl kleinerer Nachfolger für jedes Element:

$$\begin{aligned}\mathsf{inv}(L) := |\{(i,j) \mid \\ 0 \le i < j \le n-1, \\ L[i] \ge L[j]\}|\end{aligned}$$

Anzahl der Runs Ein *Run* ist eine sortierte Teilliste, die nicht nach links oder rechts verlängert werden kann. Die Anzahl der Runs ist:

$$\begin{aligned}\mathsf{runs}(L) := |\{i \mid \\ 0 \le i < n-1, \\ L[i+1] < L[i]\}| + 1\end{aligned}$$

Längster Run Anzahl der Elemente der längsten sortierten Teilliste:

$$\begin{aligned}\mathsf{las}(L) := \max\{r.\mathsf{len} \mid \\ r \text{ ist Run in } L\} \\ \mathsf{rem}(L) := L.\mathsf{len} - \mathsf{las}(L)\end{aligned}$$

Jedes allgemeine Sortierverfahren benötigt im Worst- und Average-case Schlüsselvergleiche von mindestens:

$$\Omega(n \log n)$$

Lexikographische Ordnung $\le$ Sei $A = \{a_1, \dots, a_n\}$ ein Alphabet, dass sich mit gegebener Ordnung $a_1 < \cdots < a_n$ wie folgt auf dem Lexikon $A* = \bigcup_{n \in \mathbb{N}_0} A^n$ fortsetzt:

$$\begin{aligned}&v = (v_1, \dots, v_p) \le w = (w_1, \dots, w_q) \\ \Leftrightarrow &\forall 1 \le i \le p : v_i = w_i \quad p \le q \\ \vee &\forall 1 \le j \le i : v_j = w_j \quad v_i < w_i\end{aligned}$$

Fachverteilen Sortieren von n k-Tupeln in k Schritten: Sortieren nach letztem Element, vorletzem usw.

Algo.	Stabil	Mem.	Schlüsselvergleiche		Satzbewegungen		
			C_E	C_A	C_W	M_B	M_A	M_W	
Selection	✗	1	$\frac{n(n-1)}{2}$	$\frac{n(n-1)}{2}$	$\frac{n(n-1)}{2}$	$3(n-1)$	$3(n-1)$	$3(n-1)$	
Insertion	✓	1	$n-1$	$n-1+\frac{n(n-1)}{4}+n-\ln n$	$\frac{n(n-1)}{2}$	$2(n-1)$	$\frac{n^2+3n-4}{2}+n-1$	$\frac{n^2+3n-4}{2}$	$O(n^2)$
Bubble	✓	1	$\frac{n(n-1)}{2}$	$\frac{n(n-1)}{2}$	$\frac{n(n-1)}{2}$	0	$\frac{3n(n-1)}{2}$	$\frac{3n(n-1)}{2}$	
			Best-case		**Average-case**		**Worst-case**		
Shell	✗	1	-		-		-		
Quick	✗	$\log n$	$n \log n$		$n \log n$		n^2		$O(n \log n)$
Turnier	✗	$2n-1$	$n \log n$		$n \log n$		$n \log n$		
Heap	✗	1	$n \log n$		$n \log n$		$n \log n$		
Merge	✓	n	$n \log n$		$n \log n$		$n \log n$		
Untere Schranke $\Omega(n \log n)$ für allgemeine Sortierverfahren									
Distribution	✓	n	n		n		$n \log n$, n^2		$O(n)$

$$(AB)_{ij} = \sum_{k=1}^{m} a_{ik} b_{kj}$$

(Reihe $\times$ Spalte)