Algorithmen 2

Algorithmus	Komplexität
Prioritätslisten	
Pairing Heaps	insert, merge: $\mathcal{O}(1)$, deleteMin, remove: $\mathcal{O}(\log n)$ amortisiert, decreaseKey: $\mathcal{O}(\log \log n) \leq T \leq \mathcal{O}(\log n)$
Fibonacci Heaps	deleteMin, remove: $\mathcal{O}(\log n)$ amortisiert, andere Operationen: $\mathcal{O}(1)$
Radix Heap	insert, deleteMin: $\mathcal{O}(K)$, decreaseKey: $\mathcal{O}(1)$
Graphenalgorithmen	
Dijkstra	$\mathcal{O}(m \cdot T_{\text{decreaseKey}}(n) + n \cdot (T_{\text{deleteMin}}(n) + T_{\text{insert}}(n)))$. Mit Fibonacci Heap: $\mathcal{O}(m+n \log n)$. Erwartete Laufzeit mit Binary Heap: $\mathcal{O}(m + n \log \frac{m}{n} \log n)$. $T_{\text{DijkstraBQ}} = \mathcal{O}(m + nC)$ oder $\mathcal{O}(m+\text{maxPathLength})$. $T_{\text{DijkstraRadix}} = \mathcal{O}(m+ n \cdot \log C)$.
All-Pairs Shortest Paths mit Knotenpotentialen	$\mathcal{O}(nm + n^2 \log n)$
Schrumpfgraph / SCC Berechnung	$\mathcal{O}(m + n)$
Ford-Fulkerson Maximum Flow	$\mathcal{O}(mnU)$, U: largest capacity
Dinic Maximum Flow	$\mathcal{O}(mn^2)$, $\mathcal{O}(mn \log n)$ with dynamic trees, $\mathcal{O}((m + n)\sqrt{m})$ with unit capacities, $\mathcal{O}((m + n)\sqrt{n})$ with unit networks
Preflow push	Arbitrary: $\mathcal{O}(n^2m)$, FIFO: $\mathcal{O}(n^3)$, Highest Level: $\mathcal{O}(n^2\sqrt{m})$
Goldberg-Tarjan	$\mathcal{O}(mn \log(n^2/m))$
Randomisierte Algorithmen	
Sortieren Ergebnisüberprüfung	$\mathcal{O}(n)$
Cuckoo Hashing	?
Externe Algorithmen	
Externe Stapel	$\mathcal{O}(1/B)$ I/Os pro Operation amortisiert
Externes Sortieren	$\mathcal{O}(2\frac{n}{B}(\log_{\frac{M}{B}} \frac{n}{M}))$ I/Os
Mehrwegemischen	Sortieren: $\mathcal{O}(2\frac{n}{B}(1 + \lceil \log_{M/B} \frac{n}{M} \rceil))$ I/Os

Externe Prioritätslisten	deleteMin: $\mathcal{O}(\log m)$, insert: amortisiert $\mathcal{O}(\log m)$. (m: Länge der einzelnen sortierten Sequenzen)
Approximationsalgorithmen	
ListScheduling	?
Rucksackproblem: Dynamische Programmierung	Zeit: $\mathcal{O}(n\hat{P})$ pseudo-polynomiell ($\hat{P}$ ist obere Schranke für den Profit, z.B. $\sum_i p_i$). Platz: $\hat{P} + \mathcal{O}(n)$ Maschinenworte plus $\hat{P}n$ Bits.
FPTAS für Knapsack	Laufzeit $\mathcal{O}(n^3/\varepsilon)$
Fixed-Parameter-Algorithmen	
Vertex Cover	Laufzeit $\mathcal{O}(2^k(n+m))$, k: maximale Kardinalität des Covers
Parallele Algorithmen	
Reduktion	Zeit $T_{\text{seq}}(n/p) + \Theta(\log p)$
Präfixsummen Hyperwürfel	$T_{\text{prefix}} = \mathcal{O}((\alpha + l\beta)\log p)$
Paralleles Quicksort	Erwartete Gesamtzeit: $\mathcal{O}(\alpha \log^2 p)$
Geometrische Algorithmen	
Streckenschnitt	$\mathcal{O}((n+k)\log n)$
Graham's Scan	Sortieren + $\mathcal{O}(n)$
Kleinste einschließende Kugel	Erwartet $\mathcal{O}(n)$
2D Bereichssuche	Vorverarbeitungszeit $\mathcal{O}(n \log n)$, Platz $\mathcal{O}(n)$, Anfragebearbeitung $\mathcal{O}(\log n)$ (Counting), $\mathcal{O}(k + \log n)$ oder wenigsten $\mathcal{O}(k \cdot \log n)$ (Reporting)
Onlinealgorithmen	
Ski Rental	?
Paging	?
Stringalgorithmen	
Multikey Quicksort	$\mathcal{O}(k \log k + N)$, k: Anzahl Strings, N: Länge der Strings
Naive Mustersuche	$\mathcal{O}(n \cdot m)$
Knuth-Morris-Pratt	$\mathcal{O}(n + m)$
Berechnung des Border-Arrays	$\mathcal{O}(m)$
Text Index	$\mathcal{O}(m)$ Zeit für Mustersuche, $\mathcal{O}(n)$ Zeit für Aufbau
Suffix-Baum	$\mathcal{O}(m)$ Suchzeit mit $\mathcal{O}(n\sigma)$ Wörter Platz, $\mathcal{O}(m \cdot \log \sigma)$ Suchzeit mit $\mathcal{O}(n)$ Wörter Platz. Konstruktion $\mathcal{O}(n)$. σ: Länge des Alphabets
DC3	$\mathcal{O}(n)$
Präfix-Doubling	$\mathcal{O}(n \log n)$, Platzbedarf $8n(+n)$ Wörter
LCP-Array-Konstruktion	$\mathcal{O}(n)$
LZ77	$\mathcal{O}(n)$

PSV/NSV	$\mathcal{O}(n)$
RMQ	$\mathcal{O}(n)$ Wörter Platz, Anfrage in $\mathcal{O}(1)$
Wavelet-Tree	Konstruktion (naiv) $\mathcal{O}(n \log \sigma)$, $n\lceil \log \sigma \rceil (1 + o(1))$ Bits Platz. Rank- und Select-Anfragen in $\mathcal{O}(\log \sigma)$, Zeichenzugriff in $\mathcal{O}(\log \sigma)$