



Paralleles Sortierung

Björn Rathjen Patrick Winterstein
Freie Universität Berlin

Proseminar Algorithmen, SS14

Inhalt

Motivation

Grundlage des Sortierens

Komparator

Sortiernetzwerk

Aufbau

Korrektheit

Laufzeit

Herleitung

Vergleich mit Software sortieren

Gegenüberstellung

Zusammenfassung

Ausblick

Anhang

Motivation

Grundlage des Sortierens

Sortiernetzwerk

Laufzeit

Gegenüberstellung

Zusammenfassung

Ausblick

ist Basis für :

- ▶ Suche
- ▶ (Sortierung)
 - ▶ Listen
 - ▶ Wörterbücher
 - ▶ ...
- ▶ Ist dies auch in Hardware möglich ?

Motivation

Grundlage des Sortierens Komparator

Sortiernetzwerk

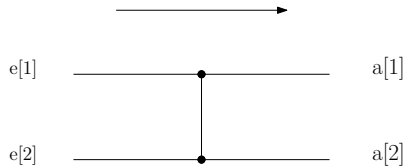
Laufzeit

Gegenüberstellung

Zusammenfassung

Ausblick

- ▶ 2 Eingänge
- ▶ vergleichender Baustein
- ▶ 2 Ausgänge



```
void comp(chan in1, in2, out1 out2){  
    a = <- in1;  
    b = <- in2;  
  
    if (a < b){  
        out1 <- a;  
        out2 <- b;  
        return void;  
    }  
    out1 <- b;  
    out2 <- a;  
    return void;  
}
```

Motivation

Grundlage des Sortierens

Sortiernetzwerk

Aufbau

Korrektheit

Laufzeit

Gegenüberstellung

Zusammenfassung

Ausblick

- ▶ mehrere Eingabeleitungen (gleiche Anzahl an Ausgabeleitungen)
- ▶ mehrere vergleichende Schritte
- ▶ Ausgabe soll sortiert sein

Aufgabe :

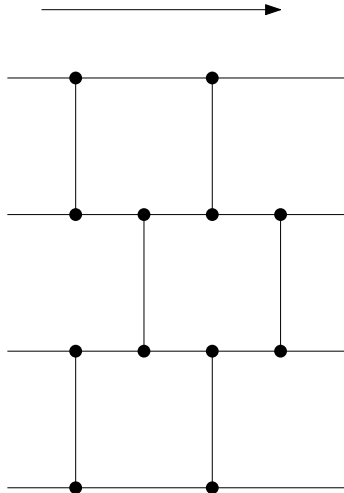
- ▶ Resultat soll sortierte Ausgabe sein

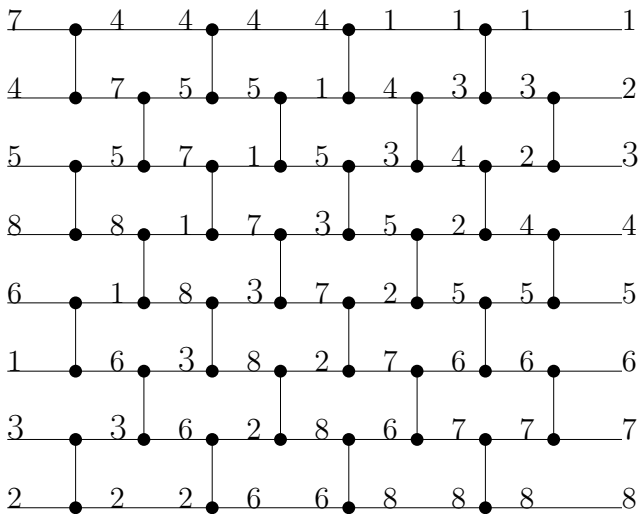
Aufgabe :

- ▶ Resultat soll sortierte Ausgabe sein

grundlegendes Prinzip :

- ▶ intuitiver Einsatz von Vergleichen
- ▶ Schrittweises sortieren



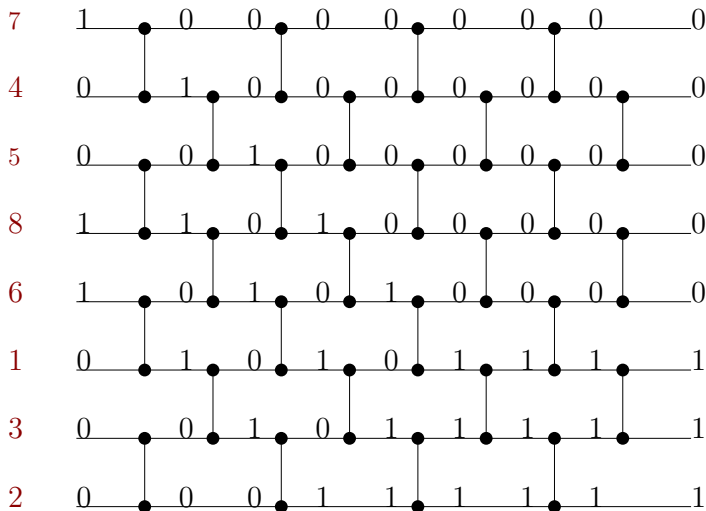


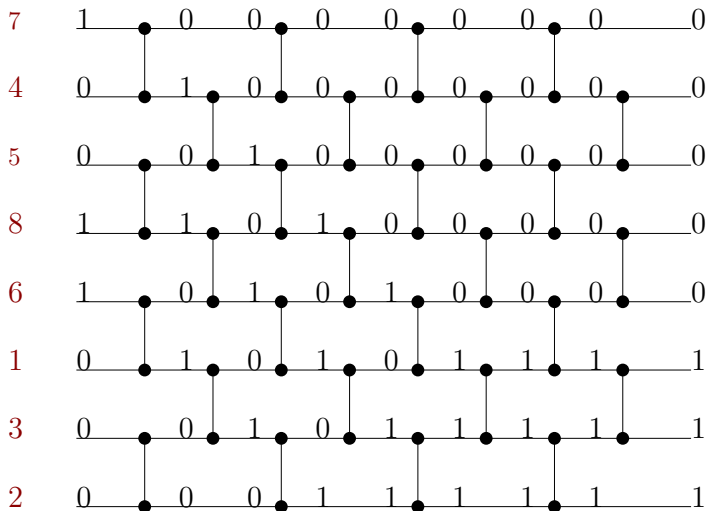
Wenn es eine Folge A gibt, die ein Sortiernetzwerk nicht sortiert, so existiert auch eine 0,1-Folge, die von diesem Netzwerk nicht sortiert wird.

man kann jede Zahlenfolge durch eine 0,1 Folge repräsentieren
Konstante k und Zahlenfolge A mit den Elementen a_i

$$f(a_i) = \begin{cases} 0, & \text{if } a_i < k \\ 1, & \text{if } a_i \geq k \end{cases}$$

0,1- Beispiel





Beispiel an der Tafel ?

Aufgabe :

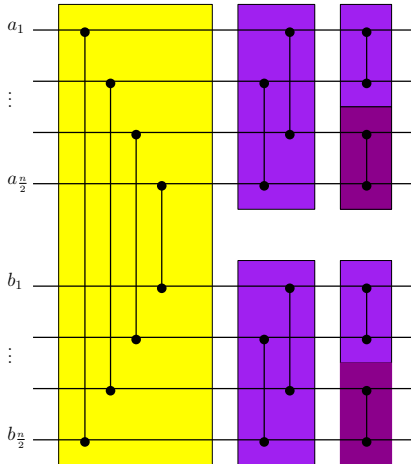
- ▶ Resultat soll sortierte Ausgabe sein
- ▶ **soll effizient sein**

Aufgabe :

- ▶ Resultat soll sortierte Ausgabe sein
- ▶ **soll effizient sein**

grundlegendes Prinzip :

- ▶ intuitiver Einsatz von Vergleichen
+ **Einbezug von Teile und Herrscher**



Ablauf :

- ▶ sortierte Eingabelisten gemischt
 - ▶ untere Hälfte alle größer als in oberer
- ▶ rekursiv die kleineren listen
- ▶ Resultat eine Sortierte Liste

Ablauf :

- ▶ bekommen zwei sortierte Listen
- ▶ trennen in geraden und ungeraden Index
- ▶ fassen $a(\text{even})$ $b(\text{odd}) = c$ und $a(\text{odd})$ $b(\text{even}) = d$ zusammen (Resultat muss sortiert sein)
- ▶ c und d werden indexweise verschachtelt
- ▶ aufeinander folgende paare werden verglichen und in richtige Reihenfolge gebracht

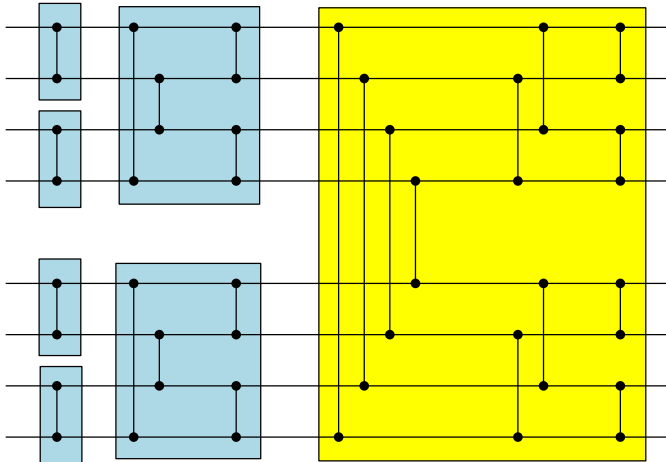


Bild kleiner Zahlenfolge 4-8-16
Beispiel

Motivation

Grundlage des Sortierens

Sortiernetzwerk

Laufzeit

- Herleitung

- Vergleich mit Software sortieren

Gegenüberstellung

Zusammenfassung

Ausblick

N	Anzahl der Schritte
2^1	

N	Anzahl der Schritte
2^1	1

N	Anzahl der Schritte
2^1	1
2^2	

N	Anzahl der Schritte
2^1	1
2^2	$1 + 2$

N	Anzahl der Schritte
2^1	1
2^2	$1 + 2$
2^k	

N	Anzahl der Schritte
2^1	1
2^2	$1 + 2$
2^k	$1 + 2 + 3 + \dots + k - 1 + k = \sum_{i=1}^k i$

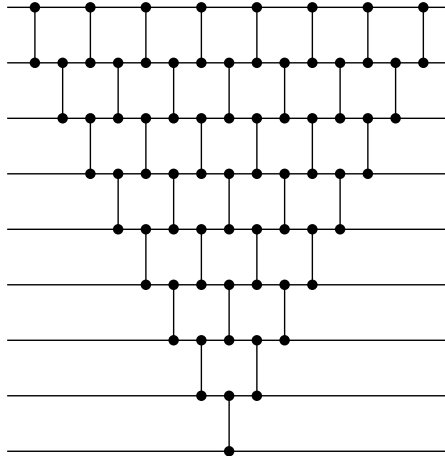
N	Anzahl der Schritte
2^1	1
2^2	$1 + 2$
2^k (kleiner Gauss)	$1 + 2 + 3 + \dots + k - 1 + k = \sum_{i=1}^k i$

N	Anzahl der Schritte
2^1	1
2^2	1 + 2
2^k	$1 + 2 + 3 + \dots + k - 1 + k = \sum_{i=1}^k i$
(kleiner Gauss)	$= \frac{k \cdot (k+1)}{2}$

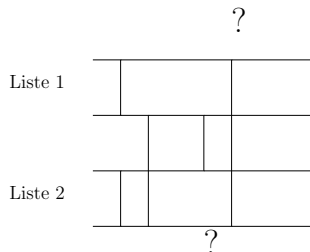
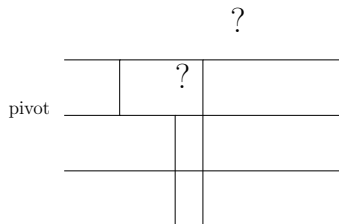
N	Anzahl der Schritte
2^1	1
2^2	1 + 2
2^k	$1 + 2 + 3 + \dots + k - 1 + k = \sum_{i=1}^k i$
(kleiner Gauss)	$= \frac{k \cdot (k+1)}{2}$
$(k = \log_2 n)$	

N	Anzahl der Schritte
2^1	1
2^2	1 + 2
2^k	$1 + 2 + 3 + \dots + k - 1 + k = \sum_{i=1}^k i$
(kleiner Gauss)	$= \frac{k \cdot (k+1)}{2}$
$(k = \log_2 n)$	$\Rightarrow \frac{1}{2} \cdot \log_2 n (\log_2 n + 1)$

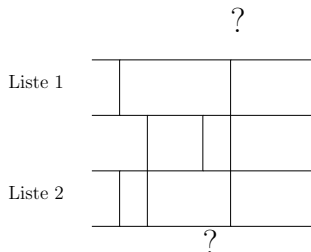
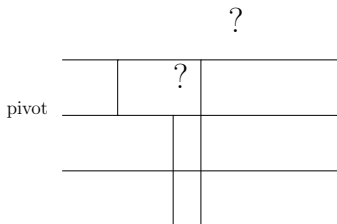
- ▶ Schritte gegen Vergleiche
- ▶ Abhängigkeit von der Eingabe
- ▶ Bezug zum vorherigen Vergleich



Mergesort Quicksort

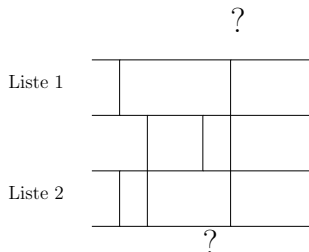
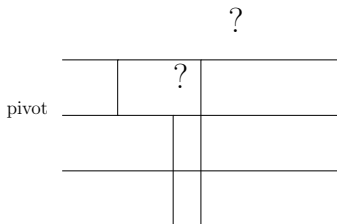


Mergesort Quicksort



Quicksort : wo ist das Pivot Element ?
mit welchem Element müssen wir nun vergleichen?

Mergesort Quicksort



Quicksort : wo ist das Pivot Element ?
mit welchem Element müssen wir nun vergleichen?
Mergesort : Wo ist nun das größte Element ?
welcher Vergleich kommt nun?)

Motivation

Grundlage des Sortierens

Sortiernetzwerk

Laufzeit

Gegenüberstellung

Zusammenfassung

Ausblick

- ▶ Geschwindigkeit vs Variabilität
 - ▶ hohe Geschwindigkeit durch direkte Hardware Implementierung
 - ▶ starre Struktur , bildet Rahmen der Möglichkeiten
 - ▶ stark typisierte Eingabe
- ▶ Hardwareaufwand vs Softwareaufwand
 - ▶ Software zur Auswertung keine zum sortieren
 - ▶ geringe Skalierbarkeit
 - ▶ hoher Aufwand wenn Eingabelimit überschritten wird
 - ▶ nur lokal
 - ▶ Hardware Konzeption eventuell aufwendiger

Motivation

Grundlage des Sortierens

Sortiernetzwerk

Laufzeit

Gegenüberstellung

Zusammenfassung

Ausblick

- ▶ paralleles sortieren ist schnell und effizient
- ▶ problemabhängige Lösung
- ▶ starr, nicht universell

Motivation

Grundlage des Sortierens

Sortiernetzwerk

Laufzeit

Gegenüberstellung

Zusammenfassung

Ausblick

Anhang

- ▶ andere Arten von Netzwerken
- ▶ Hypercubes
- ▶ Simulation von Maschinenmodellen
- ▶ ...



A. Author.

Taschenbuch der Algorithmen.

Springer Verlag , 2008.



Tom Leighton.

Einführung in Parallele Algorithmen und Architekturen

Gitter, Bäume und Hypercubes.

Thomsom Publisching , 1997.



S. Someone.

<http://www.iti.fh->

[flensburg.de/lang/algorithmen/sortieren/networks/nulleins.htm](http://www.iti.fh-flensburg.de/lang/algorithmen/sortieren/networks/nulleins.htm)