

Algoritmer og Datastrukturer

Quicksort
[CLRS, kapitel 7]

Sandsynlighedsteori

Baggrund?

- a) Ingen
- b) Gymnasiet
- c) Gymnasiet + Universitetet
- d) Universitetet
- e) Andet steds
- f) Ved ikke



Sandsynligheden for at slå krone
 $\frac{1}{2}$

Eksperiment

Kast en Mønt

- a) Plat**
- b) Krone**
- c) På højkant**
- d) Ingen mønt**

Eksperiment

Antal kast før man får krone ?

- a) 1
- b) 2
- c) 3
- d) 4
- e) 5
- f) 6
- g) 7
- h) 8
- i) 9 eller flere

Quicksort:

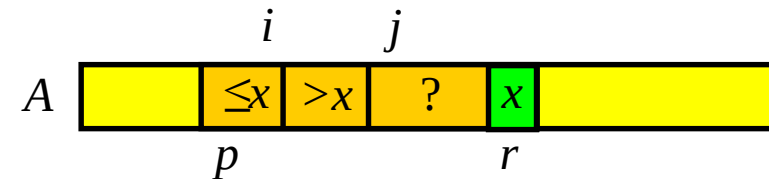
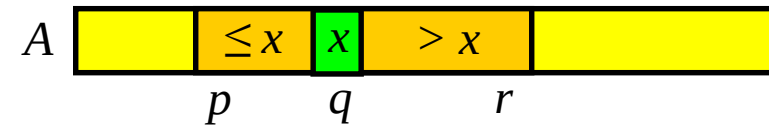
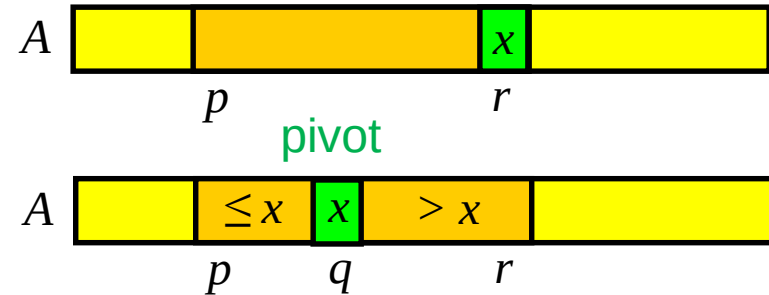
Sorter $A[p..r]$

QUICKSORT(A, p, r)

```
1  if  $p < r$ 
2       $q = \text{PARTITION}(A, p, r)$ 
3      QUICKSORT( $A, p, q - 1$ )
4      QUICKSORT( $A, q + 1, r$ )
```

PARTITION(A, p, r)

```
1   $x = A[r]$ 
2   $i = p - 1$ 
3  for  $j = p$  to  $r - 1$ 
4      if  $A[j] \leq x$ 
5           $i = i + 1$ 
6          exchange  $A[i]$  with  $A[j]$ 
7  exchange  $A[i + 1]$  with  $A[r]$ 
8  return  $i + 1$ 
```



Worst-case time $O(n^2)$

Hoare, 1961

Quicksort på 23 elementer

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
18	10	5	4	20	11	22	15	3	17	14	16	19	8	6	21	7	13	9	12	23	2	1

1	10	5	4	20	11	22	15	3	17	14	16	19	8	6	21	7	13	9	12	23	2	18
---	----	---	---	----	----	----	----	---	----	----	----	----	---	---	----	---	----	---	----	----	---	----

10	5	4	11	15	3	17	14	16	8	6	7	13	9	12	2	18	20	21	23	19	22
----	---	---	----	----	---	----	----	----	---	---	---	----	---	----	---	----	----	----	----	----	----

2	5	4	11	15	3	17	14	16	8	6	7	13	9	12	10	20	21	19	22	23
---	---	---	----	----	---	----	----	----	---	---	---	----	---	----	----	----	----	----	----	----

5	4	3	8	6	7	9	10	15	11	17	13	14	12	16	19	21	20	23
---	---	---	---	---	---	---	----	----	----	----	----	----	----	----	----	----	----	----

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

15	11	13	14	12	16	17
----	----	----	----	----	----	----

20	21
----	----

5	4	3	6	7	8
---	---	---	---	---	---

11	12	13	14	15
----	----	----	----	----

17

21

3	4	5
---	---	---

13	14
----	----

4 5 13

4

Hvor Mange Kald til Partition ?

```
QUICKSORT( $A, p, r$ )
```

```
1  if  $p < r$ 
```

```
2       $q = \text{PARTITION}(A, p, r)$ 
```

```
3      QUICKSORT( $A, p, q - 1$ )
```

```
4      QUICKSORT( $A, q + 1, r$ )
```

a) $O(\log n)$

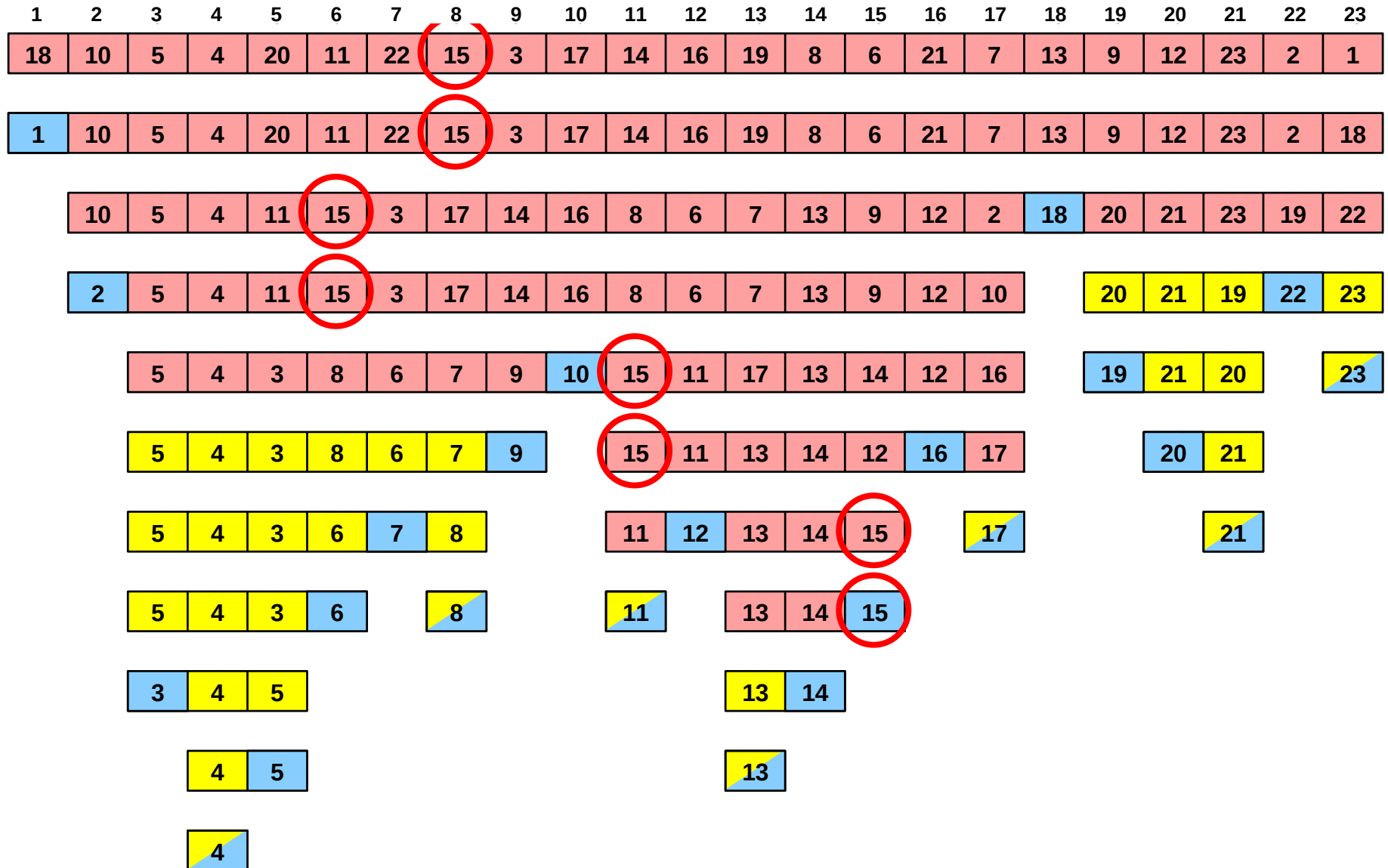
b) $O(n)$

c) $O(n \log n)$

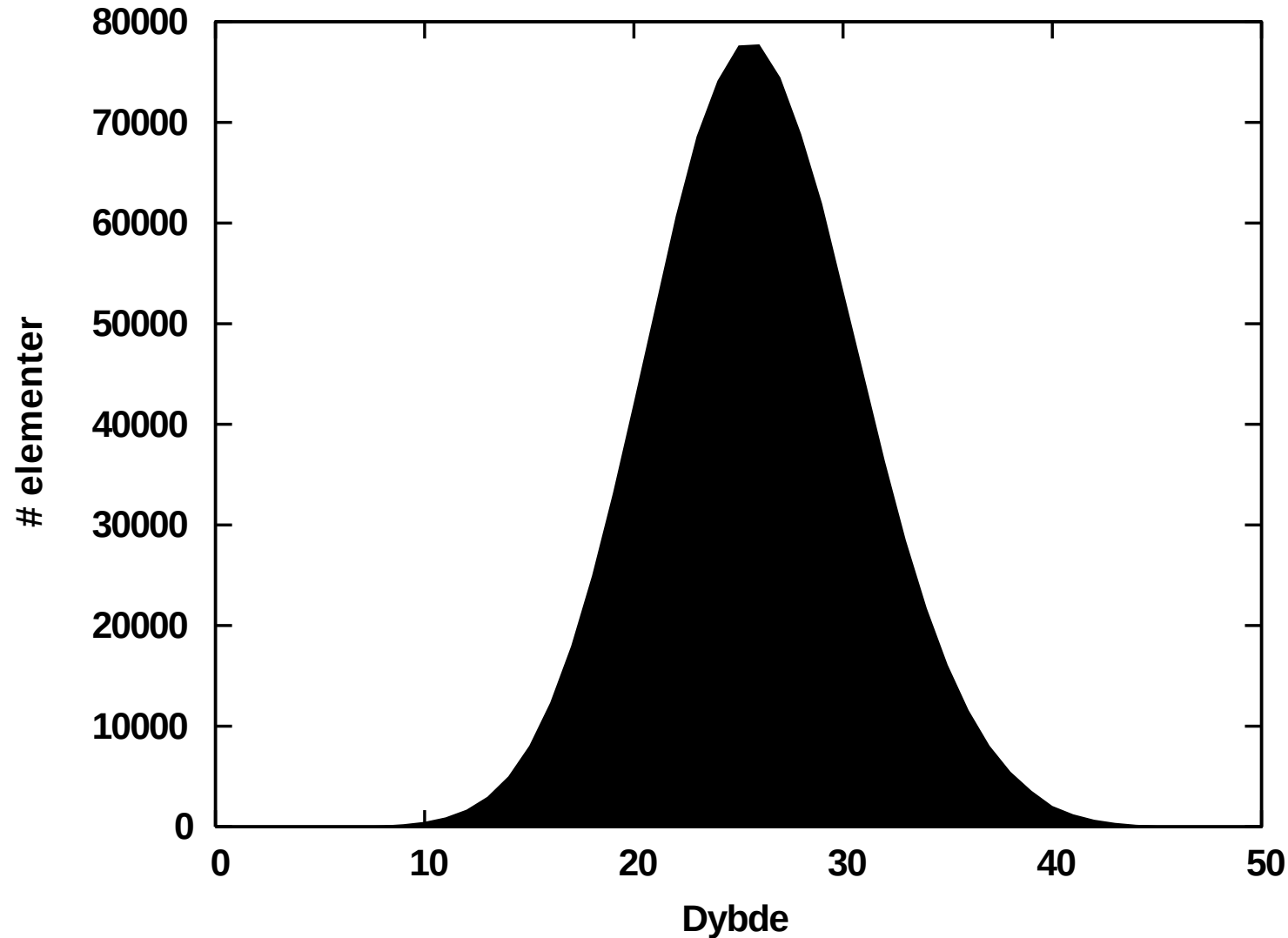
d) $O(n^2)$

e) Ved ikke

Quicksort : Rekursionen for 15



Quicksort : Dybde ved $n \approx 2^{20}$



Randomized Quicksort

RANDOMIZED-QUICKSORT(A, p, r)

```
1  if  $p < r$ 
2       $q = \text{RANDOMIZED-PARTITION}(A, p, r)$ 
3      RANDOMIZED-QUICKSORT( $A, p, q - 1$ )
4      RANDOMIZED-QUICKSORT( $A, q + 1, r$ )
```

RANDOMIZED-PARTITION(A, p, r)

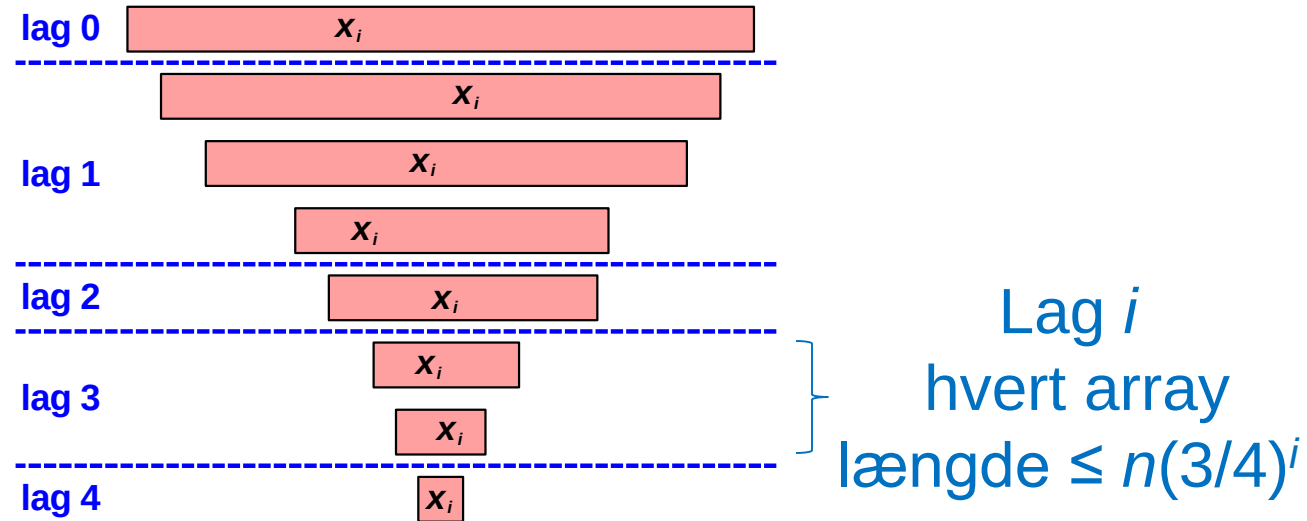
```
1   $i = \text{RANDOM}(p, r)$ 
2  exchange  $A[r]$  with  $A[i]$ 
3  return PARTITION( $A, p, r$ )
```

Forventet tid $O(n \cdot \log n)$

Forventede antal mønt-kast før man får krone 3 gange?

- a) Ingen af nedenstående
- b) Ved ikke
- c) 3
- d) 4
- e) 5
- f) 6
- g) 7
- h) 8
- i) 9

Randomized Quicksort : Analyse



- Et array er i lag j hvis længde $n(3/4)^{j+1}.. n(3/4)^j$
- En opdeling er **god** hvis hver del $\leq 3/4$ elementer (mindst +1 lag) – sker med **sandsynlighed ≥ 0.5**
- x_i forventes ≤ 2 gange i hvert lag
- **Forventede dybde af $x_i \leq 2 \cdot \log_{4/3} n$**

Randomized Quicksort : Analyse

Forventede tid for randomized quicksort

$$= O(\sum_{i=1..n} \text{forventede dybde af input } x_i)$$

$$= O(\sum_{i=1..n} \log n)$$

$$= O(n \cdot \log n)$$



Sorterings-algoritmer

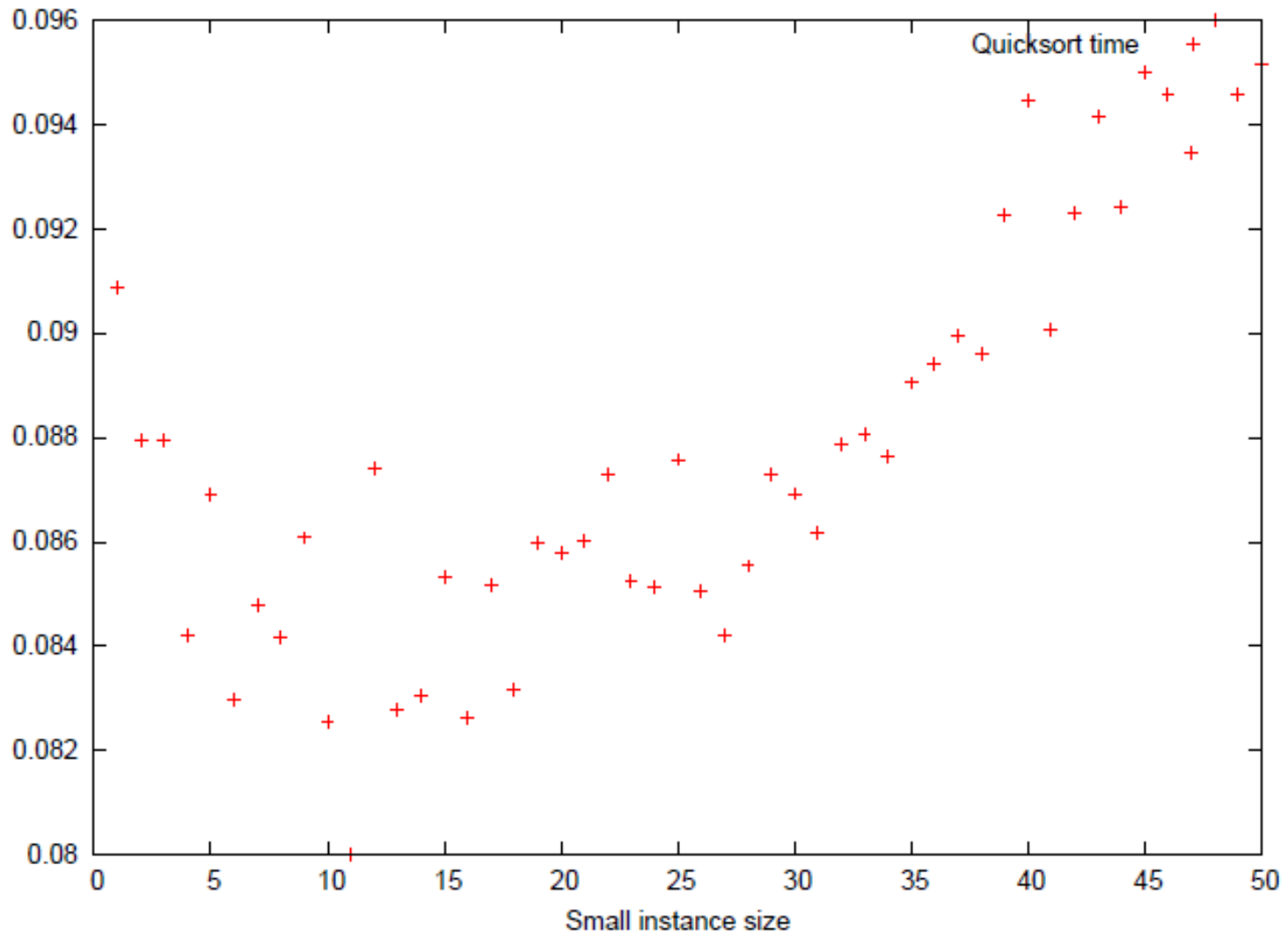
Algoritme	Worst-Case Tid
Heap-Sort	$O(n \cdot \log n)$
Merge-Sort	
Insertion-Sort	$O(n^2)$
QuickSort (Deterministic og randomiseret)	$O(n^2)$

Algoritme	Forventet tid
Randomiseret QuickSort	$O(n \cdot \log n)$

Sortering:

Eksperimentelle resultater

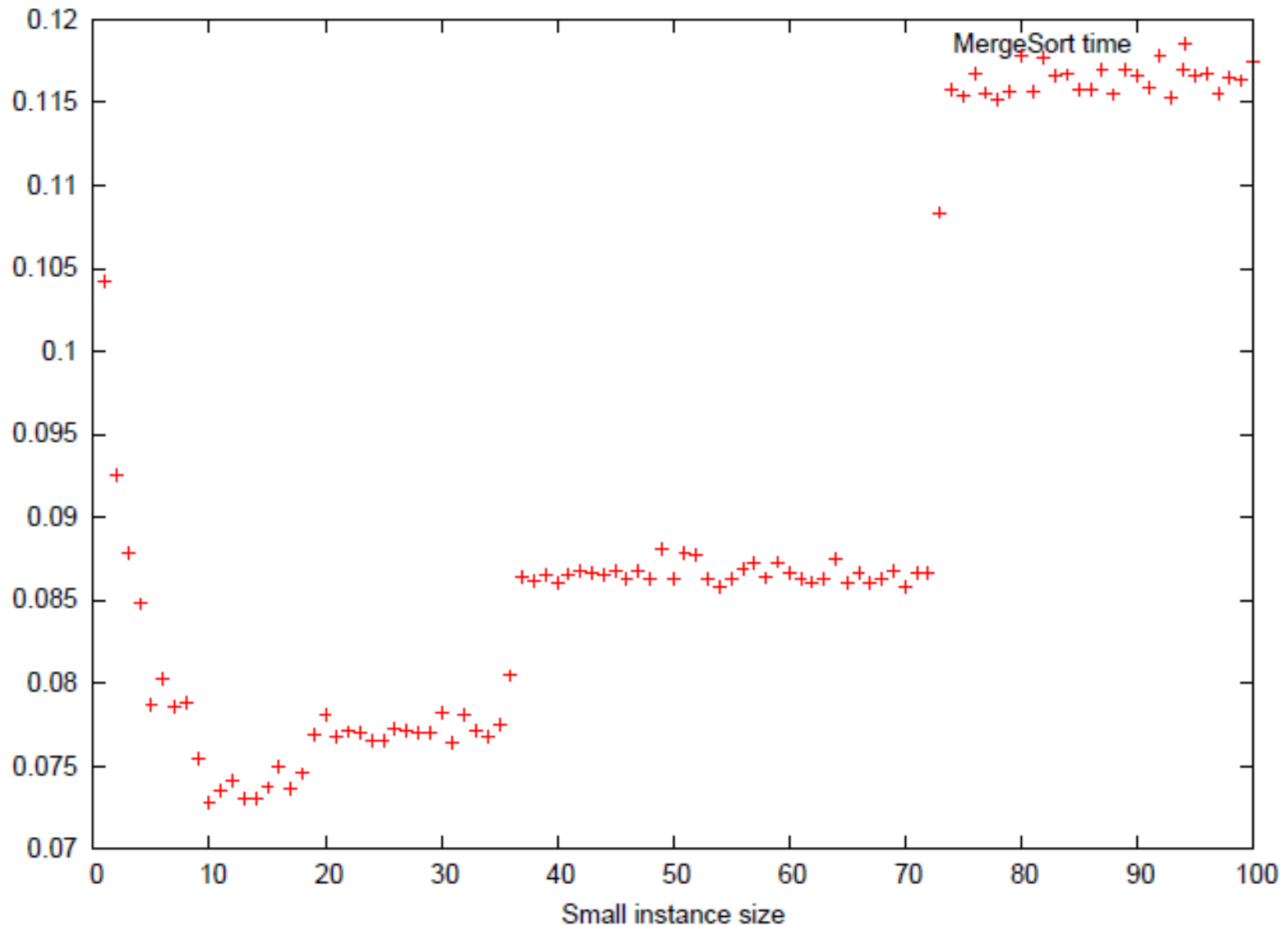
Quicksort med skift til Insertion-sort



Skift til insertion-sort ved små problemstørrelser

$n = 300.000$ elementer

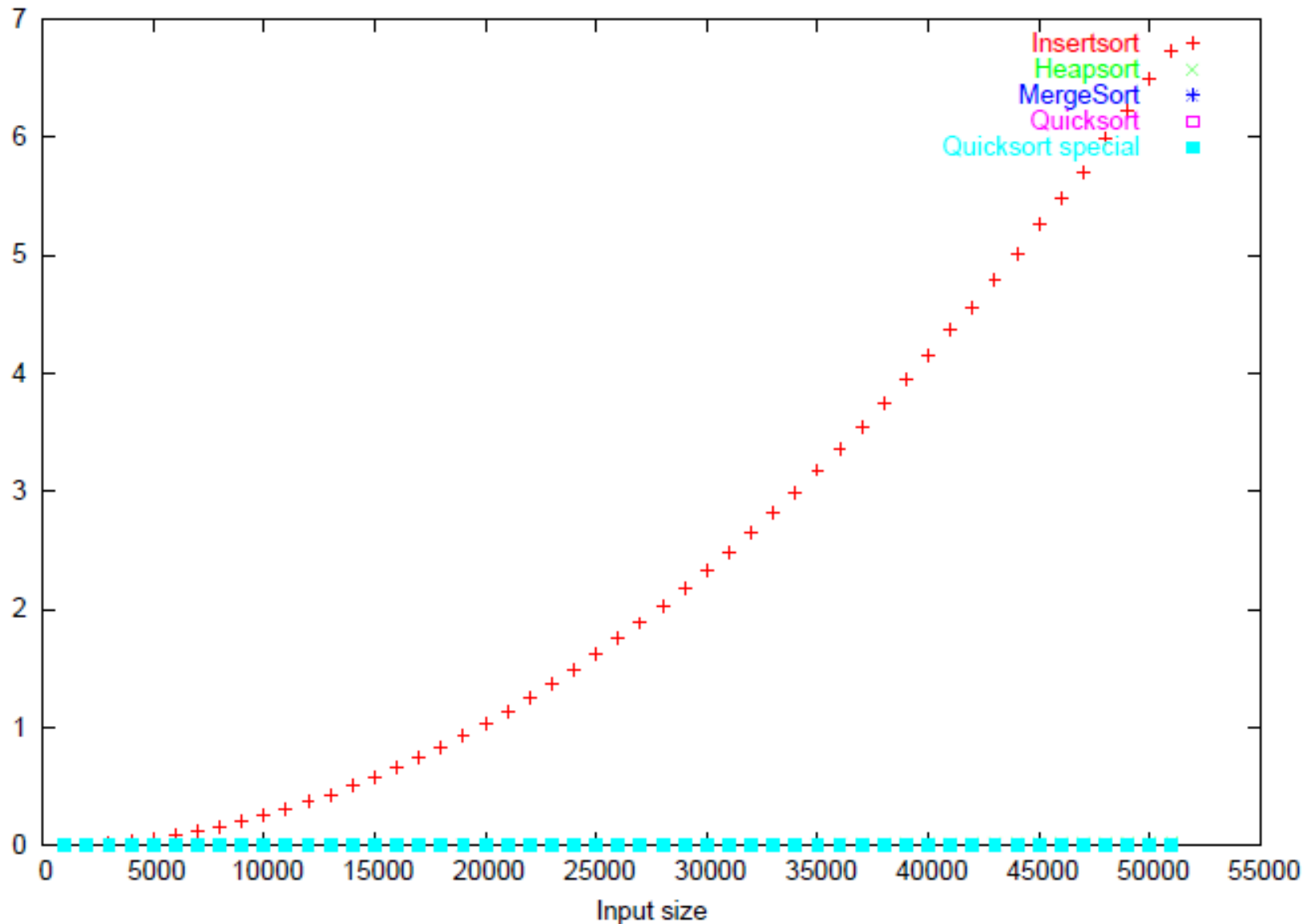
Mergesort med skift til Insertion-sort



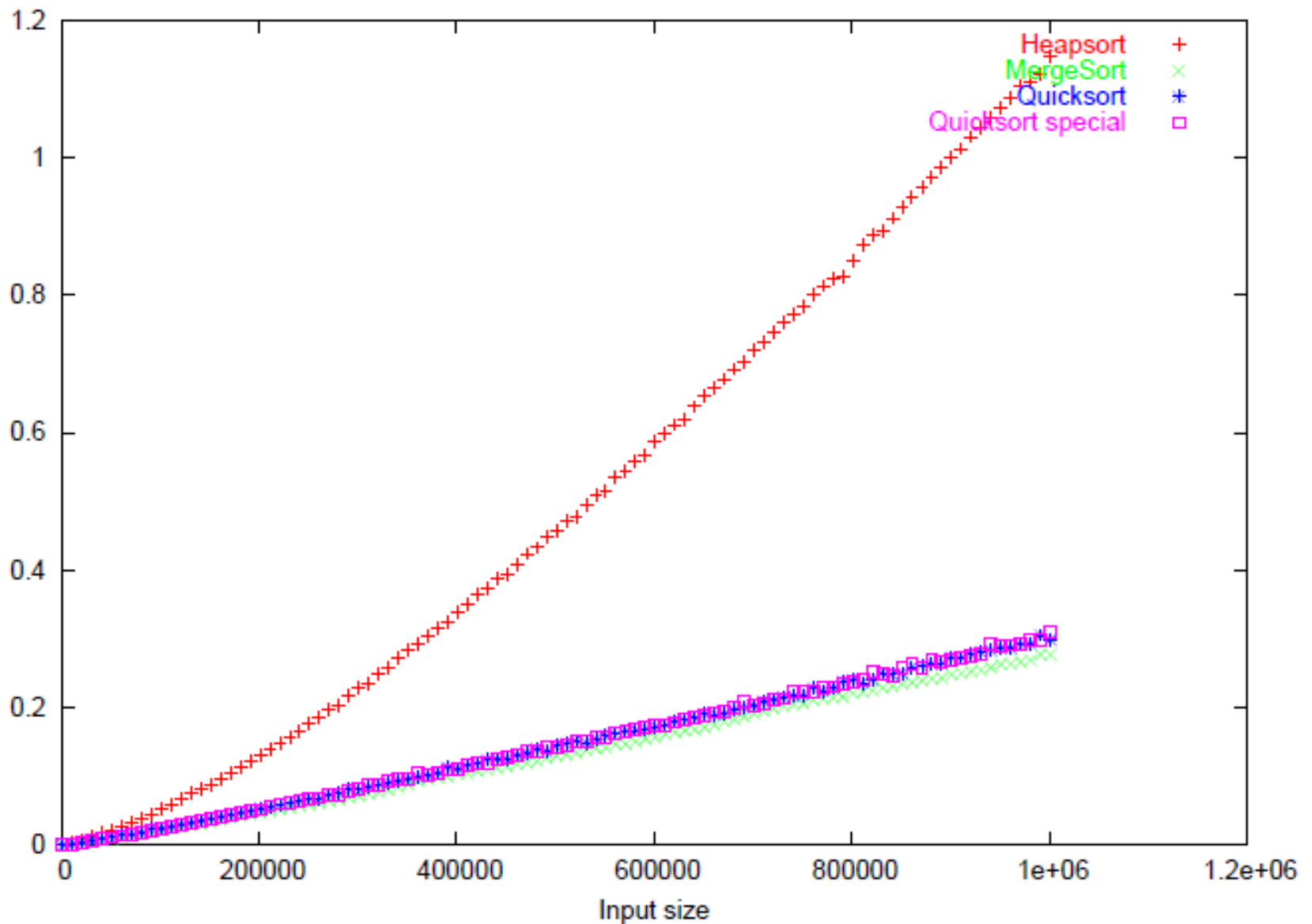
Skift til insertion-sort ved små problemstørrelser

$n = 300.000$ elementer

Tiden for Sorterings Algoritmer



Tiden for Sorterings Algoritmer



Sortering i Praksis

Typisk anvendes **QuickSort** (C, C++, Java)

- Inplace (kræver ingen yderligere arrays)
- Hurtig

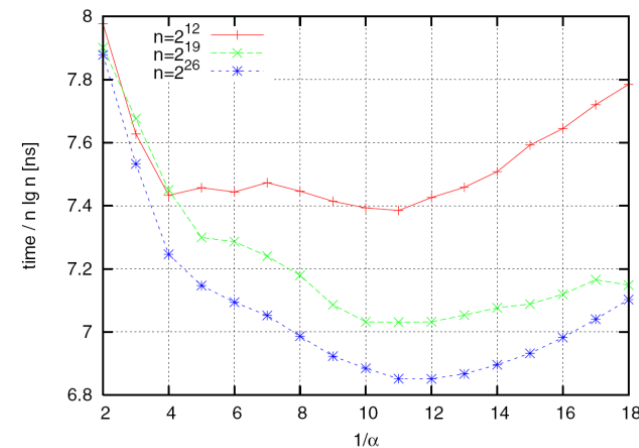
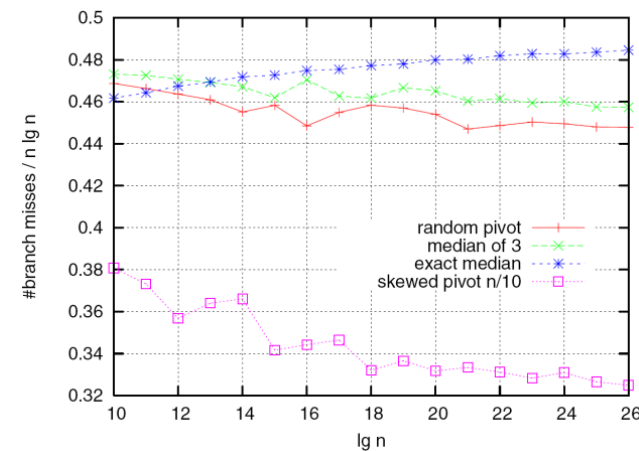
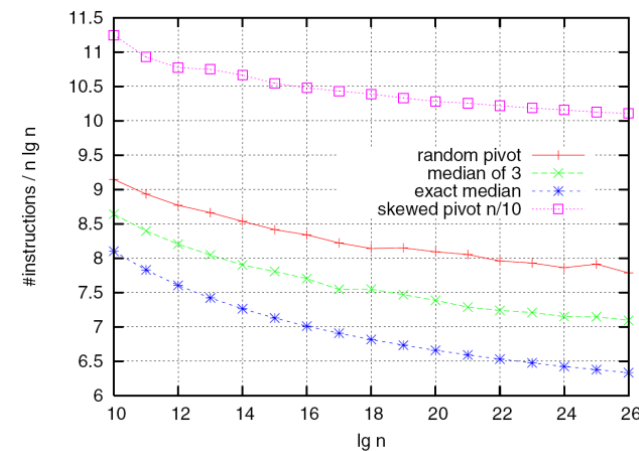
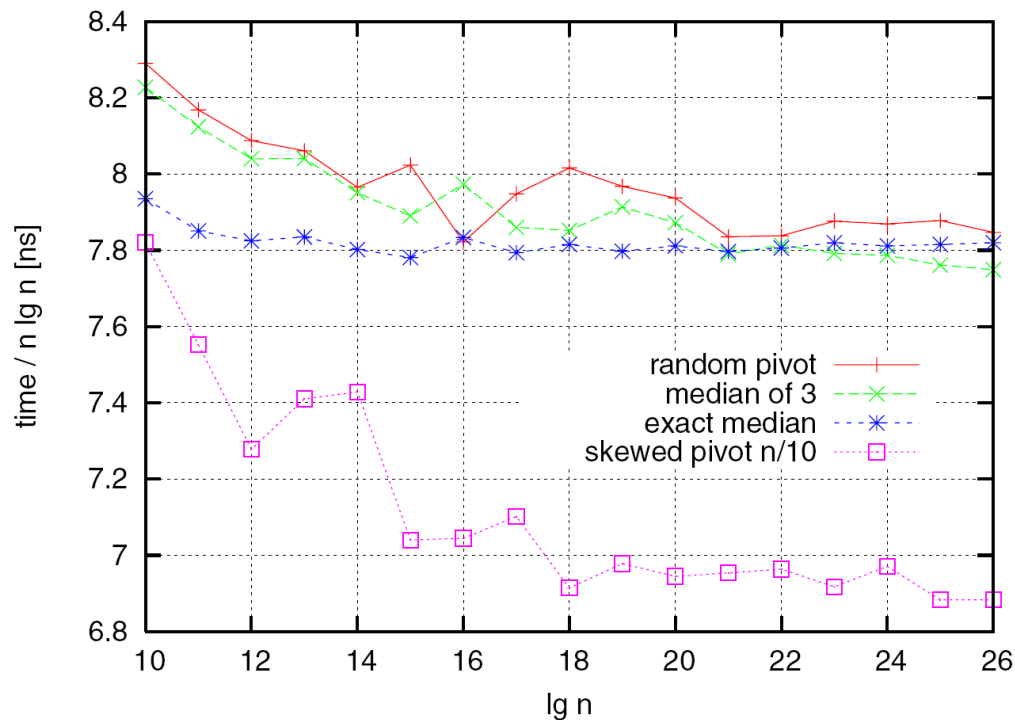
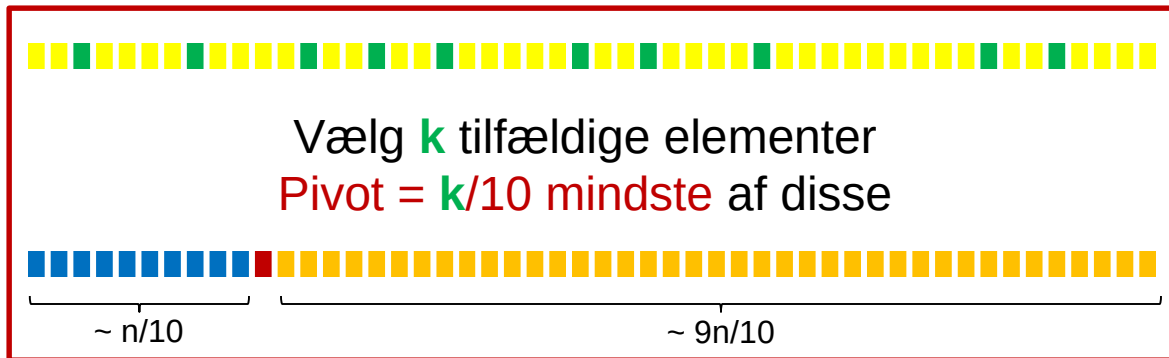
...dog med et par twists

- skift til **InsertionSort** i bunden af rekursionen
- skift til **MergeSort** hvis QuickSort tager for lang tid
- Vælg Pivot'et som medianen ud af $O(1)$ (tilfældige) elementer

"How Branch Mispredictions Affect Quicksort"

Kanela Kaligosi and Peter Sanders

14th Annual European Symposium on Algorithms (ESA 2006)



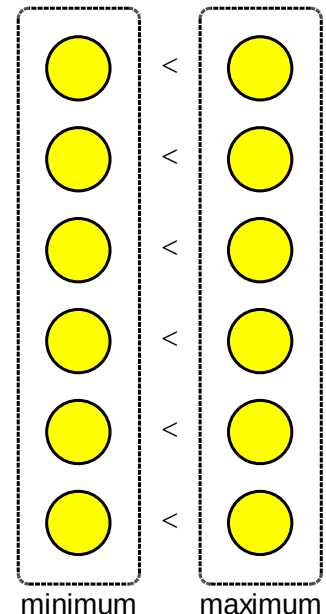
3GHz Pentium 4 Prescott, GCC 3.3, PAPI, gns. 100 kørsler med tilfældige permutationer af $[1..n]$

Grundlæggende Algoritmer og Datastrukturer

**Randomized-select
[CLRS, kapitel 9.1-9.2]**

Beregning af Minimum of Maximum

- At finde *minimum* af n elementer kræver $n-1$ sammenligninger
- At finde *minimum* og *maximum* af n elementer kræver $3/2 \cdot n - 2$ sammenligninger

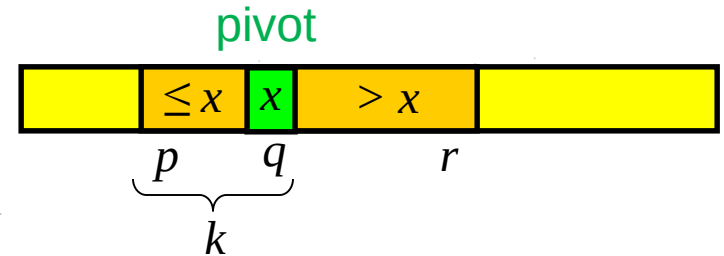


Randomized Select:

Find the i 'th smallest element in $A[p..r]$ ($1 \leq i \leq r-p+1$)

RANDOMIZED-SELECT(A, p, r, i)

```
1  if  $p == r$ 
2      return  $A[p]$ 
3   $q = \text{RANDOMIZED-PARTITION}(A, p, r)$ 
4   $k = q - p + 1$ 
5  if  $i == k$            // the pivot value is the answer
6      return  $A[q]$ 
7  elseif  $i < k$ 
8      return RANDOMIZED-SELECT( $A, p, q - 1, i$ )
9  else return RANDOMIZED-SELECT( $A, q + 1, r, i - k$ )
```



Randomized-Select 15

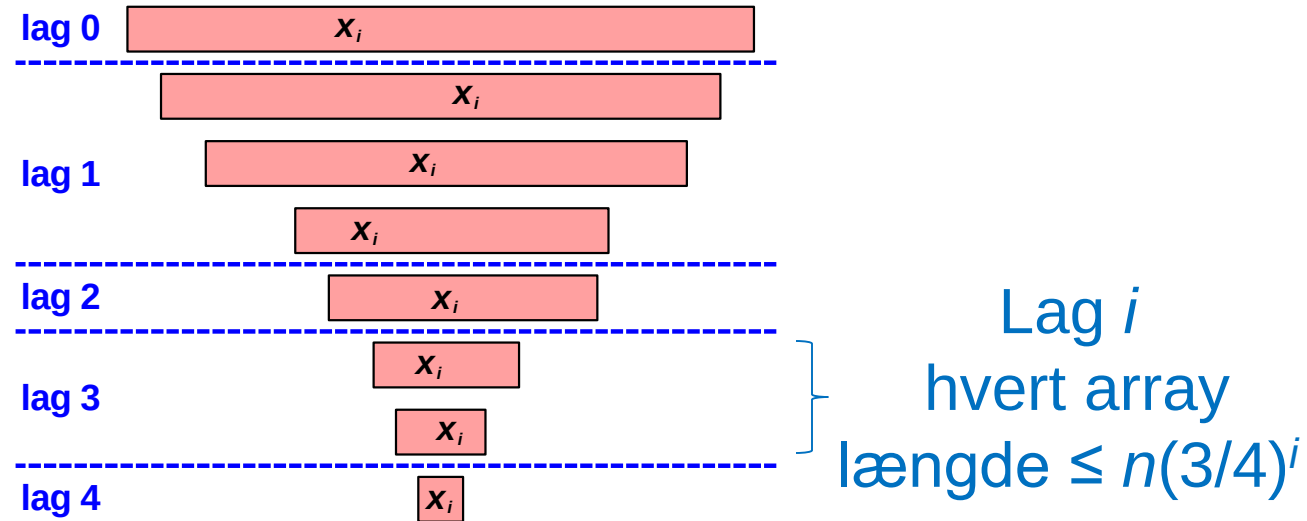
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
18	10	5	4	20	11	22	15	3	17	14	16	19	8	6	21	7	13	9	12	23	2	1
1	10	5	4	20	11	22	15	3	17	14	16	19	8	6	21	7	13	9	12	23	2	18
10	5	4	11	15	3	17	14	16	8	6	7	13	9	12	2	18	20	21	23	19	22	
2	5	4	11	15	3	17	14	16	8	6	7	13	9	12	10							
5	4	3	8	6	7	9	10	15	11	17	13	14	12	16								
15	11	13	14	12	16	17																
11	12	13	14	15																		
13	14	15																				

Worst-case tid for Randomized-Select ?

```
RANDOMIZED-SELECT( $A, p, r, i$ )
1  if  $p == r$ 
2      return  $A[p]$ 
3   $q = \text{RANDOMIZED-PARTITION}(A, p, r)$ 
4   $k = q - p + 1$ 
5  if  $i == k$            // the pivot value is the answer
6      return  $A[q]$ 
7  elseif  $i < k$ 
8      return  $\text{RANDOMIZED-SELECT}(A, p, q - 1, i)$ 
9  else return  $\text{RANDOMIZED-SELECT}(A, q + 1, r, i - k)$ 
```

- a) $O(\log n)$
- b) $O(n)$
- c) $O(n \log n)$
- d) $O(n^2)$
- e) Ved ikke

Randomized Select : Analyse



Forventede tid for randomized select

$$= O(\sum_j \text{forventede tid i lag } j)$$

$$= O(\sum_j n \cdot (3/4)^j \cdot \# \text{ forventede arrays i lag } j)$$

$$= O(\sum_j n \cdot (3/4)^j \cdot 2) \leftarrow \sum_{i=0}^{\infty} c^i = \frac{1}{1-c} \text{ for } 0 < c < 1$$

$$= \mathbf{O(n)}$$



Selektion

Algoritme	Tid
Randomized-Select [CLRS, Kap. 9.2]	$O(n)$ forventet $O(n^2)$ worst-case
Deterministic-Select [CLRS, Kap. 9.3]	$O(n)$ worst-case