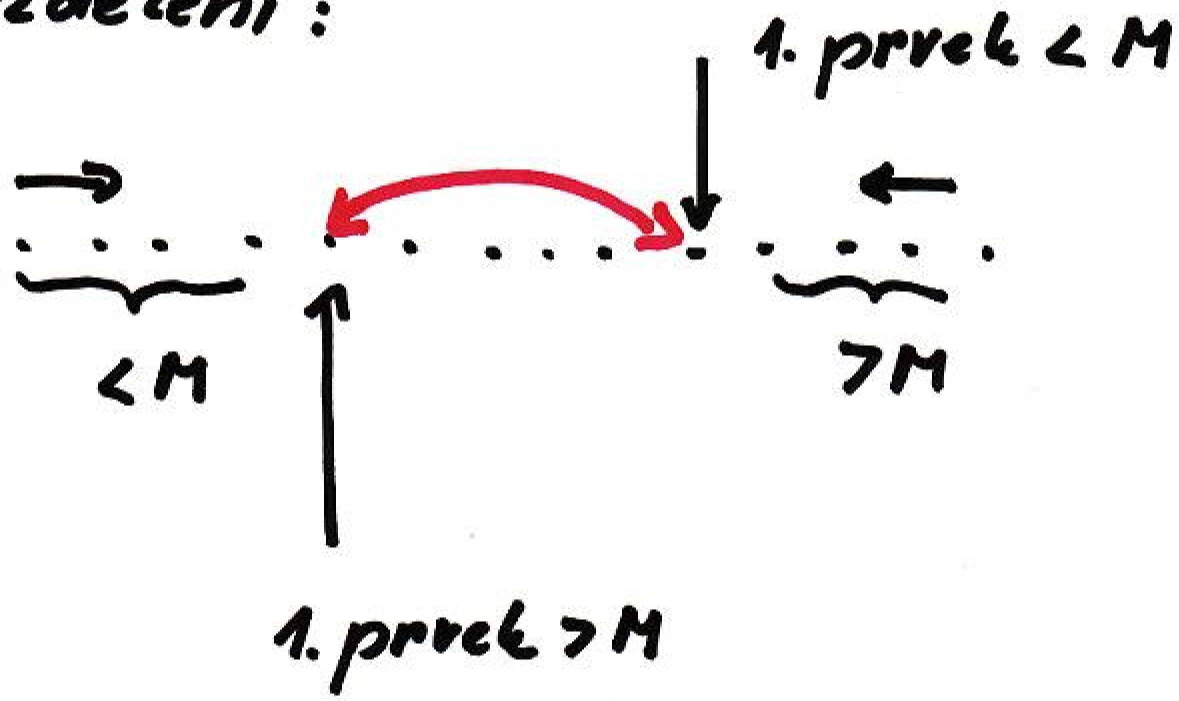
QUICKSORT

Hoare (1961 - 1962)

princip "divide et impera"

1) volba pivota M
2) rozdělení pole na prvky < M a > M
3) rekurzivní volání na obě části

rozdělení:



procedure Quicksort (l, r)

$i := l$, $k := r+1$, $a := a(l)$

while $i < k$ **do**

  repeat $i := i+1$ **until** $a(i) \geq a$

  repeat $k := k-1$ **until** $a(k) \leq a$

  if $i < k$ **then** $x := a(i)$, $a(i) := a(k)$, $a(k) := x$

  endif

enddo

$x := a(l)$, $a(l) := a(k)$, $a(k) := x$

if $l < k-1$ **then** Quicksort $(l, k-1)$ **endif**

if $k+1 < r$ **then** Quicksort $(k+1, r)$ **endif**

volá se Quicksort $(1, n)$

je třeba dodefinovat $a(0)$, $a(n+1)$, aby

$$a(0) \leq a(i) \leq a(n+1) \text{ pro vš. } i$$

potřebuje **zásobník** **velikosti $O(\log n)$**

dá se převést na iteraci

Časová složitost

$$T(n) = 1 + n + T(k-1) + T(n-k)$$

$$T(0) = T(1) = 0$$

za předpokladu, že třídíme permutaci

$k =$ hodnota pivota

nejhorší případ: $T(n) \le 1 + n + \max_k \{T(k-1) + T(n-k)\}$

maximum pro $k = 1$ nebo $k = n$

$$T(n) = 1 + n + T(n-1) = O(n^2)$$

nejlepší případ: pro $k = \frac{n}{2}$

$$T(n) = 1 + n + 2T(\tfrac{n}{2}) = O(n \log n)$$

průměrný případ:

předp. $P(\text{pivot} = k) = \frac{1}{n}$ pro vš. k

$$ET(n) = \frac{1}{n} \sum_{k=1}^{n} (1 + n + ET(k-1) + ET(n-k))$$

$$ET(0) = ET(1) = 0$$

řešení:

$$ET(n) = \frac{1}{n}\sum_{k=1}^{n}\left(1+n+ET(k-1)+ET(n-k)\right) =$$

$$= n+1+\frac{2}{n}\sum_{k=0}^{n-1} ET(k)$$

$$n\,ET(n) = n(n+1) + 2\sum_{k=0}^{n-1} ET(k) \qquad (1.)$$

$$(n+1)\,ET(n+1) = (n+1)(n+2) + 2\sum_{k=0}^{n} ET(k) \qquad (2.)$$

odečteme (1.) od (2.):

$$(n+1)\,ET(n+1) = 2(n+1) + (n+2)\,ET(n)$$

$$ET(n+1) = 2 + \frac{n+2}{n+1}\,ET(n)$$

postupným dosazováním vyjde:

$$ET(n) = 2\sum_{i=2}^{n}\frac{n+1}{i+1} = 2(n+1)\left(H_{n+1}-\frac{3}{2}\right) \leq$$

$$\leq 2(n+1)\log(n+1) = O(n\log n)$$

Volba pivota

1) vždy levý krajní

pro náhodnou permutaci je to volba stejně dobrá či špatná jako kterákoli jiná

pro setříděný soubor nejhorší možná ($O(n^2)$)

2) náhodná volba (randomizovaný Quicksort)

"obrana" proti nejhoršímu případu, když nevíme nic o rozložení vstupních dat

3) medián

výpočet mediánu : $O(n)$

Quicksort : $T(n) = 2T(\frac{n}{2}) + O(n) = O(n \log n)$

konstanta je příliš vysoká

4) pseudomedián

5) medián z konečného (malého) počtu prvků

MEDIÁN

Hoare (1962) – algoritmus FIND (QUICKSELECT)

nejhorší případ: $O(n^2)$

průměrný případ: $O(n)$

Blum, Floyd, Pratt, Rivest, Tarjan (1973):

algoritmus SELECT

nejhorší případ: $O(n)$ přesně $5.43n$

Schönhage, Paterson, Pippenger (1976):

algoritmus dosti komplikovaný

složitost $3n$

Dor, Zwick (1995):

vylepšení předchozího algoritmu na $2.95n$

teoretická dolní mez pro složitost výpočtu mediánu je $(2+\varepsilon)n$

FIND

hledá i-tý nejmenší prvek v M

procedure Find(M, i)

$s :=$ nějaký prvek M

$M_1 := \{ m \in M;\ m < s \}$

$M_2 := \{ m \in M;\ m > s \}$

case $|M_1|$ **of**

$< i-1$: Find$(M_2,\ i - |M_1| - 1)$

$= i-1$: return s

$> i-1$: Find(M_1, i)

endcase

očekávaná doba výpočtu:

$$T(n,i) \leq cn + \frac{1}{n}\left(\sum_{k=1}^{i-1} T(n-k,\ i-k) + \sum_{k=i+1}^{n} T(k-1,\ i) \right)$$

$$T(n) \leq cn + \frac{1}{n} \max_i \left(\sum_{k=1}^{i-1} T(n-k) + \sum_{k=i}^{n-1} T(k) \right) \leq 4cn$$

$$T(n) = \max_i T(n,i)$$

SELECT

procedure SELECT(M, i)

rozděl M na $\lceil \frac{n}{5} \rceil$ podmnožin $M_1, \dots, M_{\lceil \frac{n}{5} \rceil}$

po 5 prvcích

for $j := 1$ **to** $\lceil \frac{n}{5} \rceil$ **do** $m_j :=$ medián M_j **enddo**

SELECT$(\{m_1, \dots, m_{\lceil \frac{n}{5} \rceil}\}, \lceil \frac{n}{10} \rceil)$

co $\bar{m}$ = medián z mediánů

$M_1 := \{m \in M;\ m < \bar{m}\}$

$M_2 := \{m \in M;\ m > \bar{m}\}$

if $i \le |M_1|$ **then** SELECT(M_1, i)

else SELECT$(M_2, i - |M_1|)$

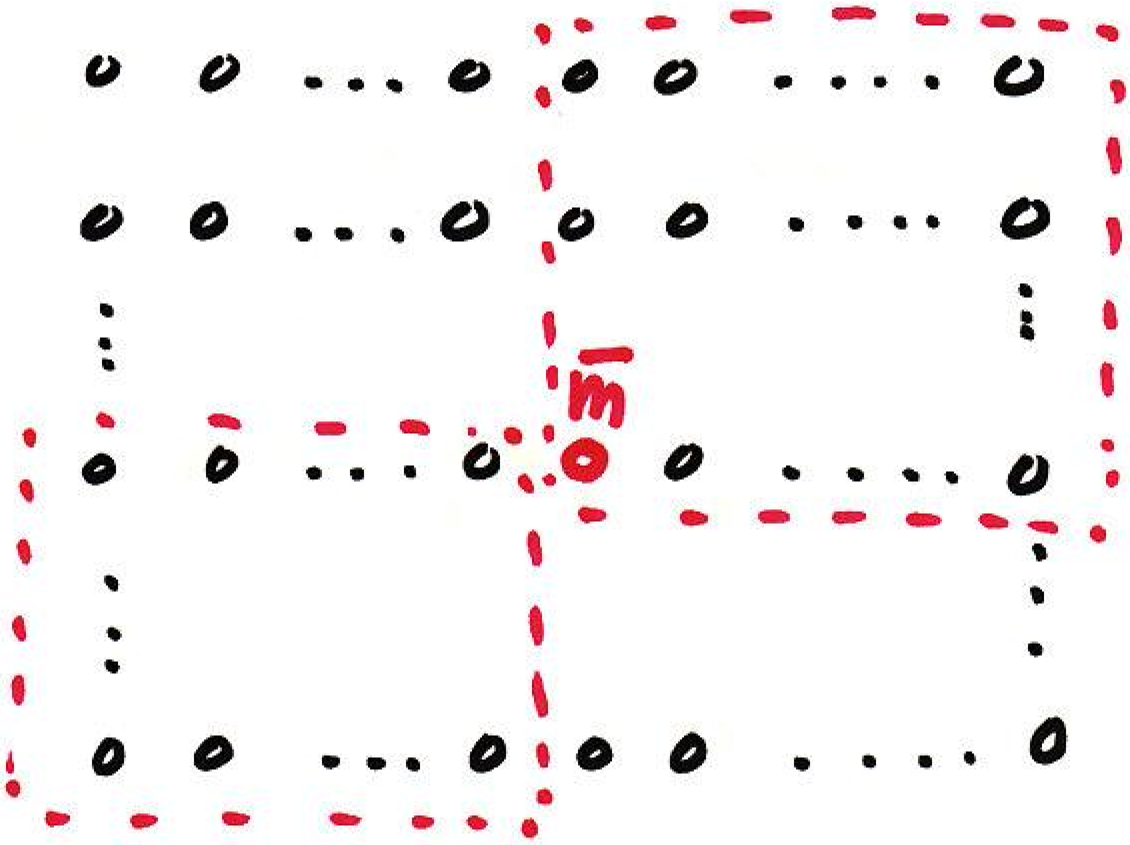
endif

doba výpočtu:

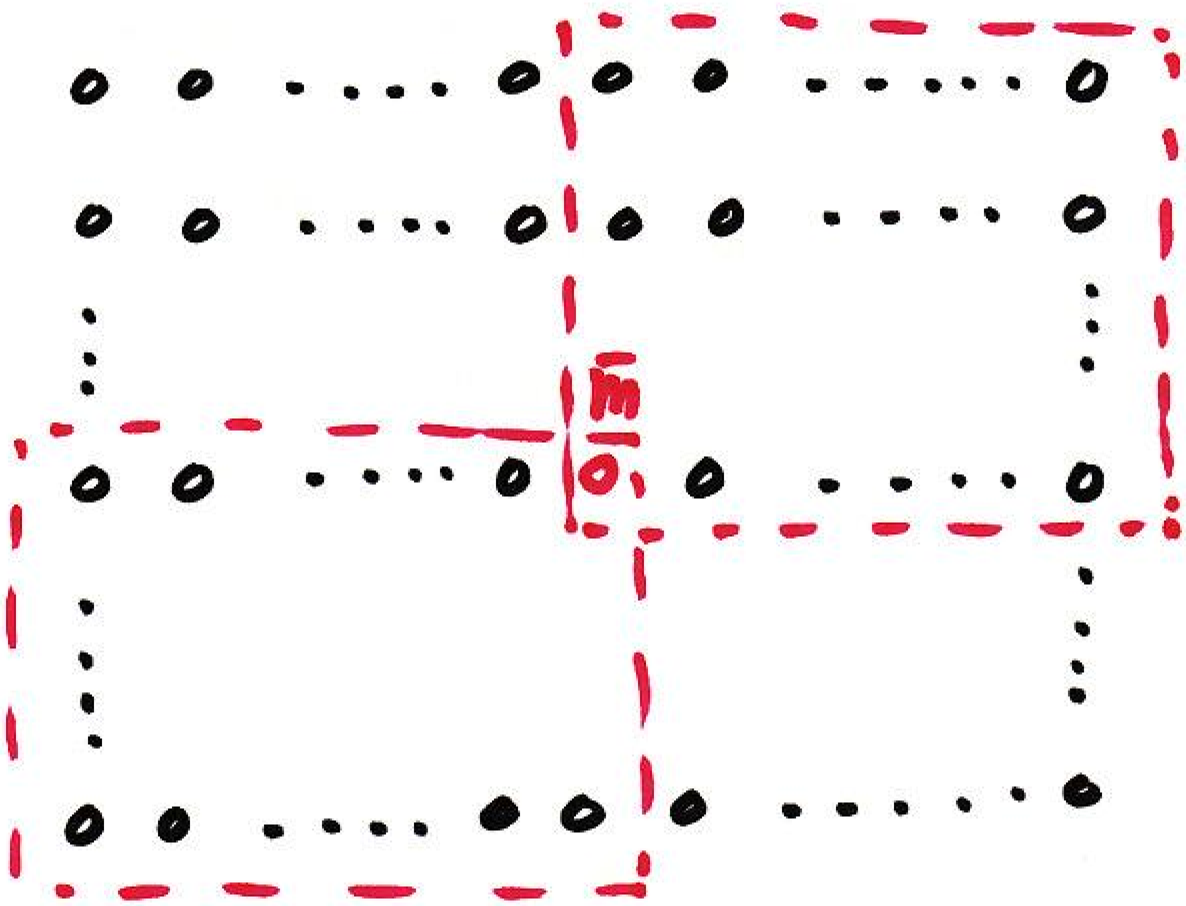
$$T(n) \le T\left(\left\lceil \frac{n}{5} \right\rceil\right) + T\left(\underbrace{\left\lceil \frac{7n}{10} \right\rceil}\right) + O(n) = O(n)$$

např. - uvádí se
různě přesně

proč? představa



nebo



PSEUDOMEDIÁN

Posloupnost se rozdělí na trojice (např.), v každé se určí medián.

Posloupnost těchto mediánů se opět rozdělí na trojice, v každé se určí medián.

atd. ...

počet porovnání:

$$3\left(\frac{n}{3}+\frac{n}{9}+\ldots+\frac{n}{3^k}+\ldots\right) \leq$$

$$\leq n\sum_{k=0}^{\infty}\frac{1}{3^k} = \frac{3}{2}n$$

jako aproximace pro Quicksort to stačí

Medián z konečného počtu prvků

medián ze 3:

očekávaná doba výpočtu:

$$ET(n) = \sum_{k=1}^{n} \frac{(n-k)(k-1)}{\binom{n}{3}} \left(ET(n-k) + ET(k-1)\right) + O(n)$$

$$\sim \frac{12}{7}(n+1)\log(n+1)$$

To platí pro rovnoměrné rozdělení vstupních permutací!

V nejhorším případě je 2x rychlejší než původní verze.

původní: $T(n) = n + T(n-1) = n + n-1 + T(n-2) = \ldots$

med. ze 3: $T(n) = n + T(n-2) = n + n-2 + T(n-4) = \ldots$

každý 2. člen vypadne

<u>medián z 2t+1</u> : Sedgewick (1976)

$$ET(n) = \sum_{k=1}^{n} \frac{\binom{n-k}{t}\binom{k-1}{t}}{\binom{n}{2t+1}} \left(ET(n-k) + ET(k-1)\right) + O(n)$$

řešení:

$$ET(n) = \frac{1}{H_{2t+2} - H_{t+1}} (n+1) H_{n+1} + O(n)$$

$ET(n) \longrightarrow n \log n + O(n)$ pro $t \to \infty$

ale velmi pomalu

největší zrychlení je pro $t=1$

cena hledání mediánu pro větší t roste

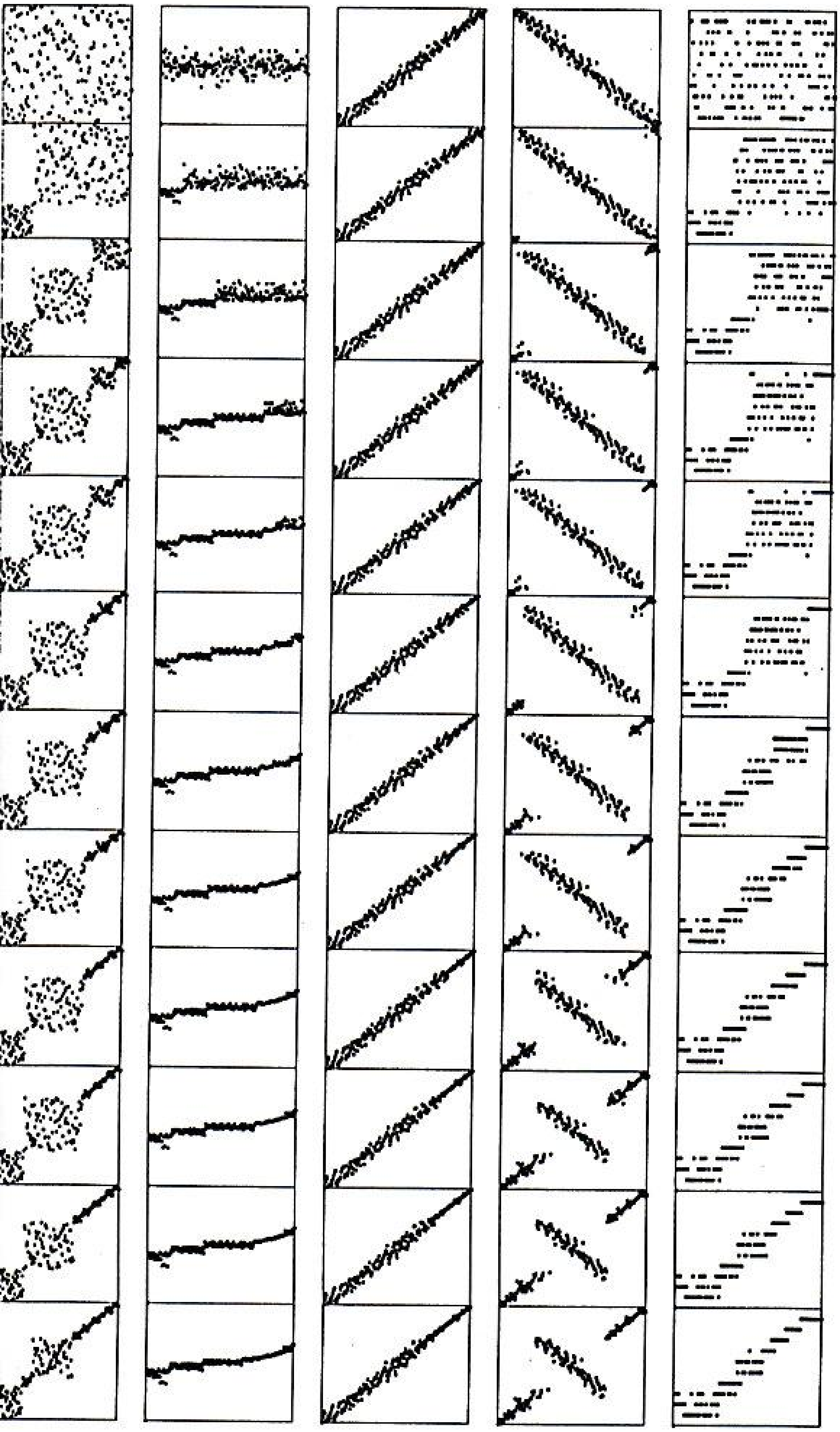


Diagram 7.4 Dynamické charakteristiky quicksortu pro různé typy souborů

Volba libovolného dělicího prvku v quicksortu vyústí v odlišné rozdělovací scénáře pro různé soubory. Tyto diagramy ilustrují výchozí části scénářů pro soubory, jež jsou náhodné, gaussovské, jsou téměř uspořádány, téměř reverzně uspořádány a nahodile uspořádány s 10 různými hodnotami klíčů (zleva doprava), pomocí relativně velké hodnoty přerušení pro malé podsoubory. Prvky nezahmuté v dělení končí podél úhlopříčky, a pole je pak snáze zpracovatelné vkládacím tříděním. Téměř uspořádané soubory vyžadují nadměrný počet oddílů.

Tento přesný výsledek je téměř roven sumě, již lze snadno aproximovat integrálem (viz odstavci 2.3):

$$\frac{C_N}{N+1} \approx 2 \sum_{1 \le k < N} \frac{1}{k} \approx 2 \int_1^N \frac{1}{x} dx = 2 \ln N,$$

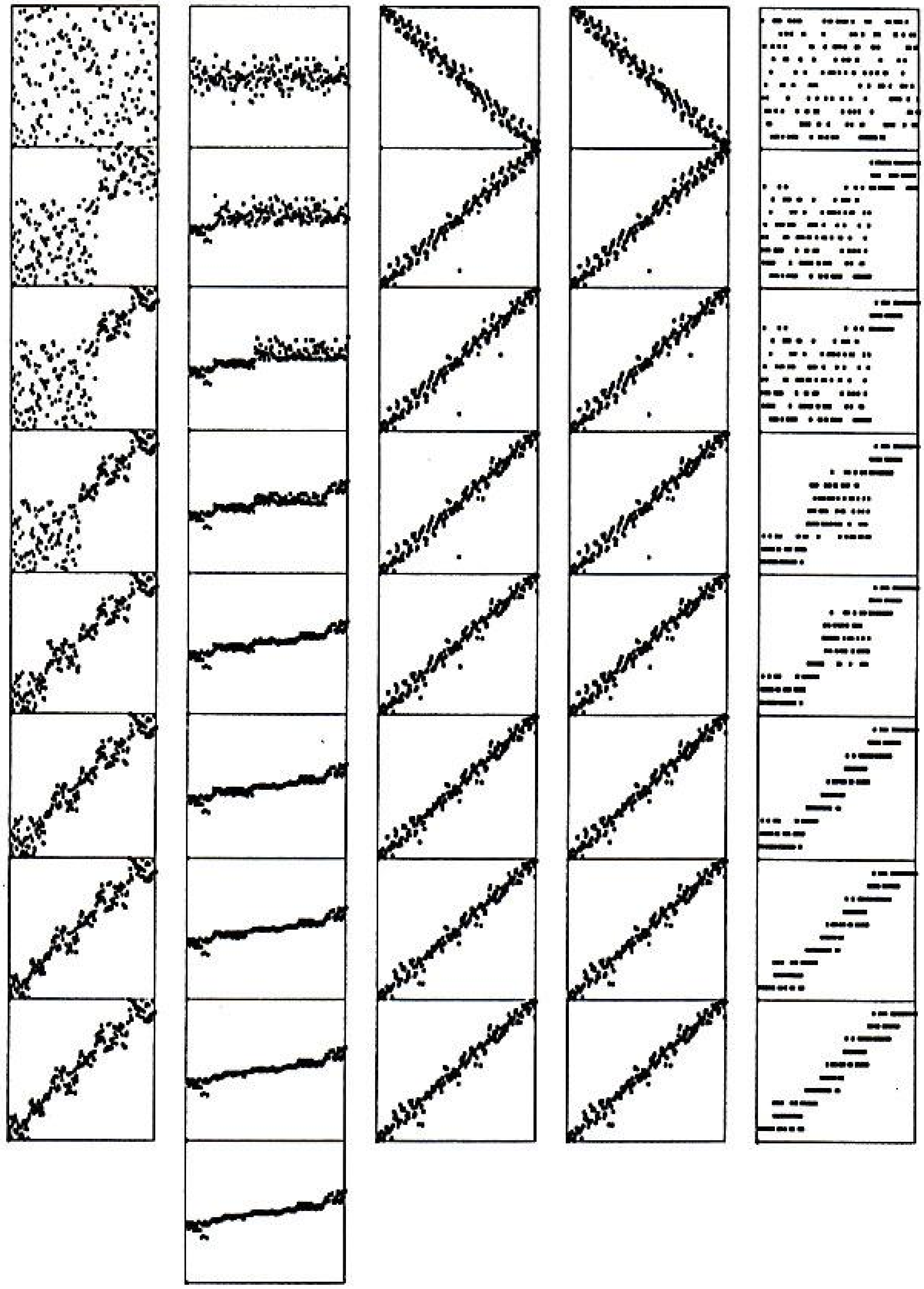


Diagram 7.11 Dynamické charakteristiky quicksortu s mediánem ze tří pro různé typy souborů

Modifikace pro medián ze tří (částečně pomocí středního prvku souboru) provádí dobrou práci tak, že činí dělicí proces robustnějším. Degenerované typy souborů uvedené v diagramu 7.4 jsou zvládány relativně dobře. Další možností, jíž lze dosáhnout stejného cíle, je použití náhodného dělicího prvku.

Quicksort je velmi rozšířen, poněvadž dobře pracuje v nejrůznějších situacích. Může se stát, že pro některé dílčí případy může být vhodnější jiná metoda, ale quicksort zvládá více typů třídění, než zvládají jiné třídicí metody, a je často významně rychlejší než alternativní přístupy. Tabulka 7.1 ukazuje empirické výsledky pro podporu uvedených poznámek.

Martinez, Roura (2001):

Optimální velikost výběru prvků pro výpočet mediánu je $a\sqrt{n} + o(\sqrt{n})$.

a ... konstanta závislá na metodě výpočtu mediánu a na ceně operací porovnání a výměn (jak?)

Jsou-li výměny mnohem dražší než porovnání, není medián vhodnou volbou pro pivota v Quicksortu.

Bentley, Mc Ilroy (1993):

pro $n < 7$ použít jednoduchý Quicksort

pro $8 \leq n \leq 39$ Quicksort s mediánem ze 3

pro $n > 40$ Quicksor s pseudomediánem z 9

tyto postupy kombinovat

očekávaná složitost empiricky:

$$1{,}5783\, n \log_e n - 0{,}74\, n$$

Durand (2003)

předchozí algoritmus teoreticky:

meze pro přepínání 15 a 85 prvků

očekávaná složitost:

$$1{,}5697\, n \log_e n - 1{,}1512\, n + 1{,}5697 \log_e n -$$
$$- 7{,}4633 + o(1)$$

Další optimalizace – dotřiďování malých polí

Sedgewick (1976)

malá pole dotřídit Insertionsortem

hranice je kolem 9 prvků

doporučení: malá pole nechat nesetříděná

na závěr Insertionsort na celou posloupnost

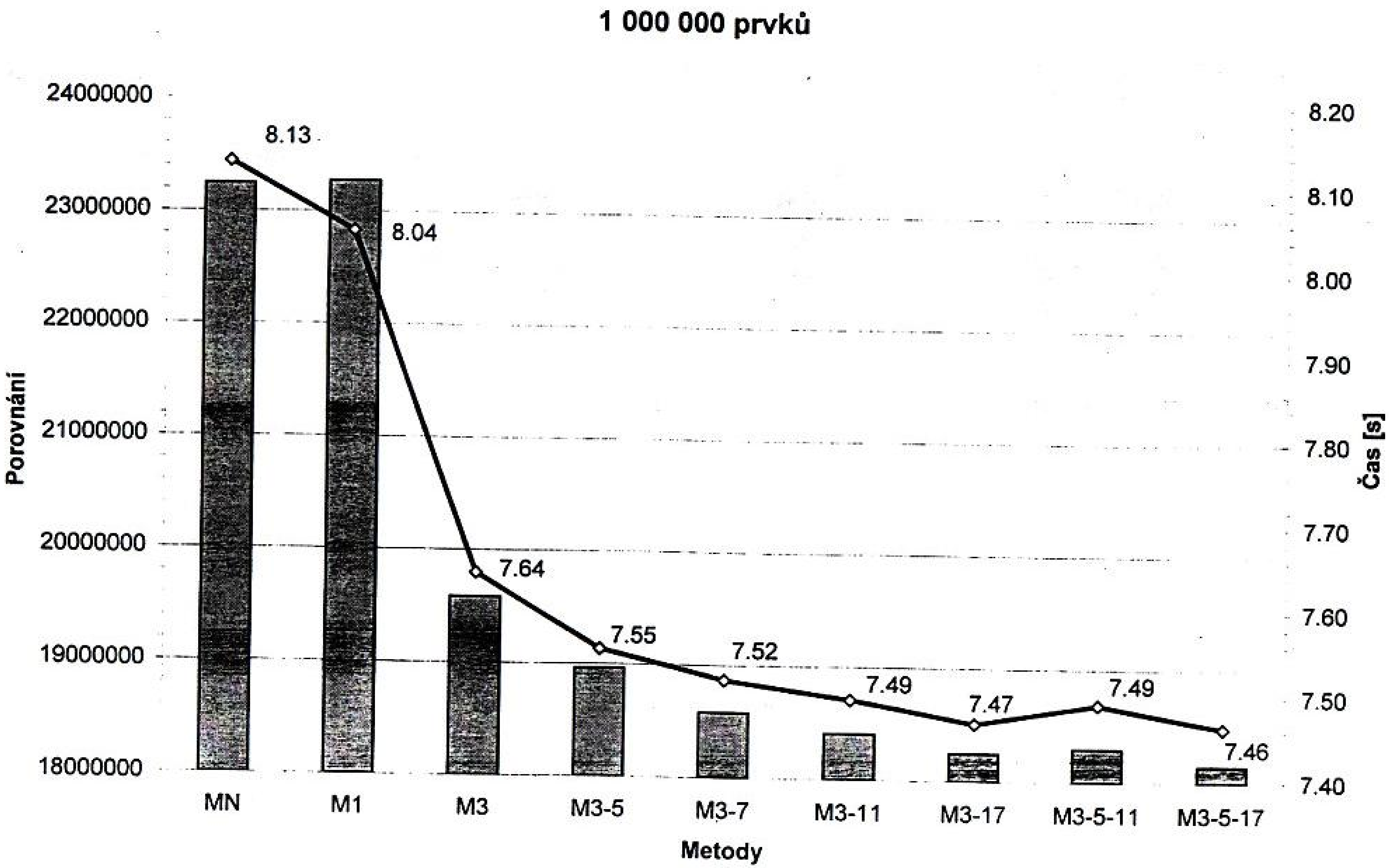
současné doporučení:

v paměti s cachí setřídit každé malé pole zvlášť,

hned jak se k němu dojde

(nenastane výpadek z cache)

R. Monic - diplomka (1997)



R. Moric - diplomka (1997)

Tabulka 6.1a - Hraniční velikost metody M3-5

Velikost pole	500	750	1000	1250	1500
Čas [s]	3,615	3,607	3,598	3,603	3,611

Tabulka 6.1b - Hraniční velikost metody M3-7

Velikost pole	1000	1500	2000	2500	3000
Čas [s]	3,580	3,572	3,576	3,683	3,625

Tabulka 6.1c - Hraniční velikost metody M3-11

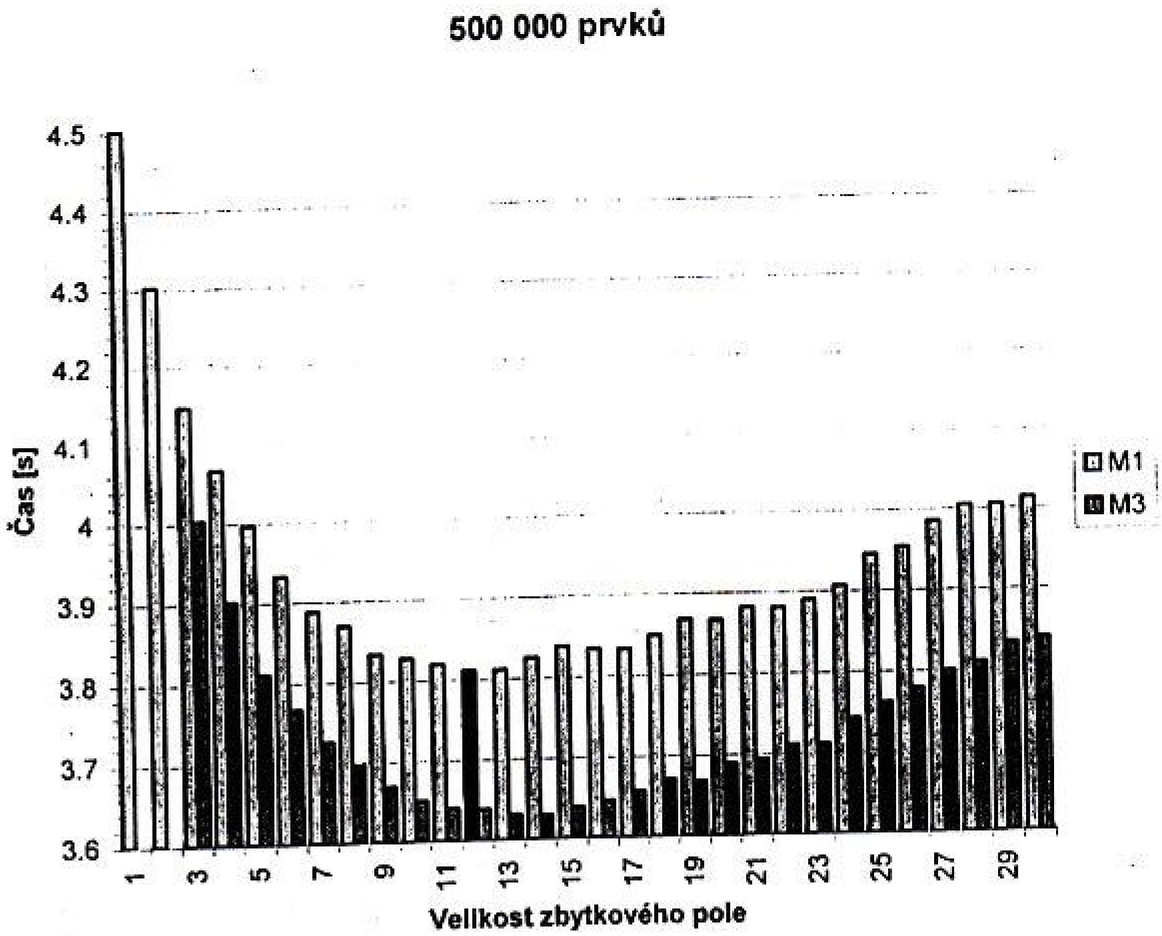
Velikost pole	1000	1500	2000	2500	3000
Čas [s]	3,575	3,570	3,565	3,578	3,583

Tabulka 6.1d - Hraniční velikost metody M3-17

Velikost pole	1500	2000	2500	3000	3500
Čas [s]	3,565	3,553	3,550	3,557	3,563

R. Moric - diplomka (1997)

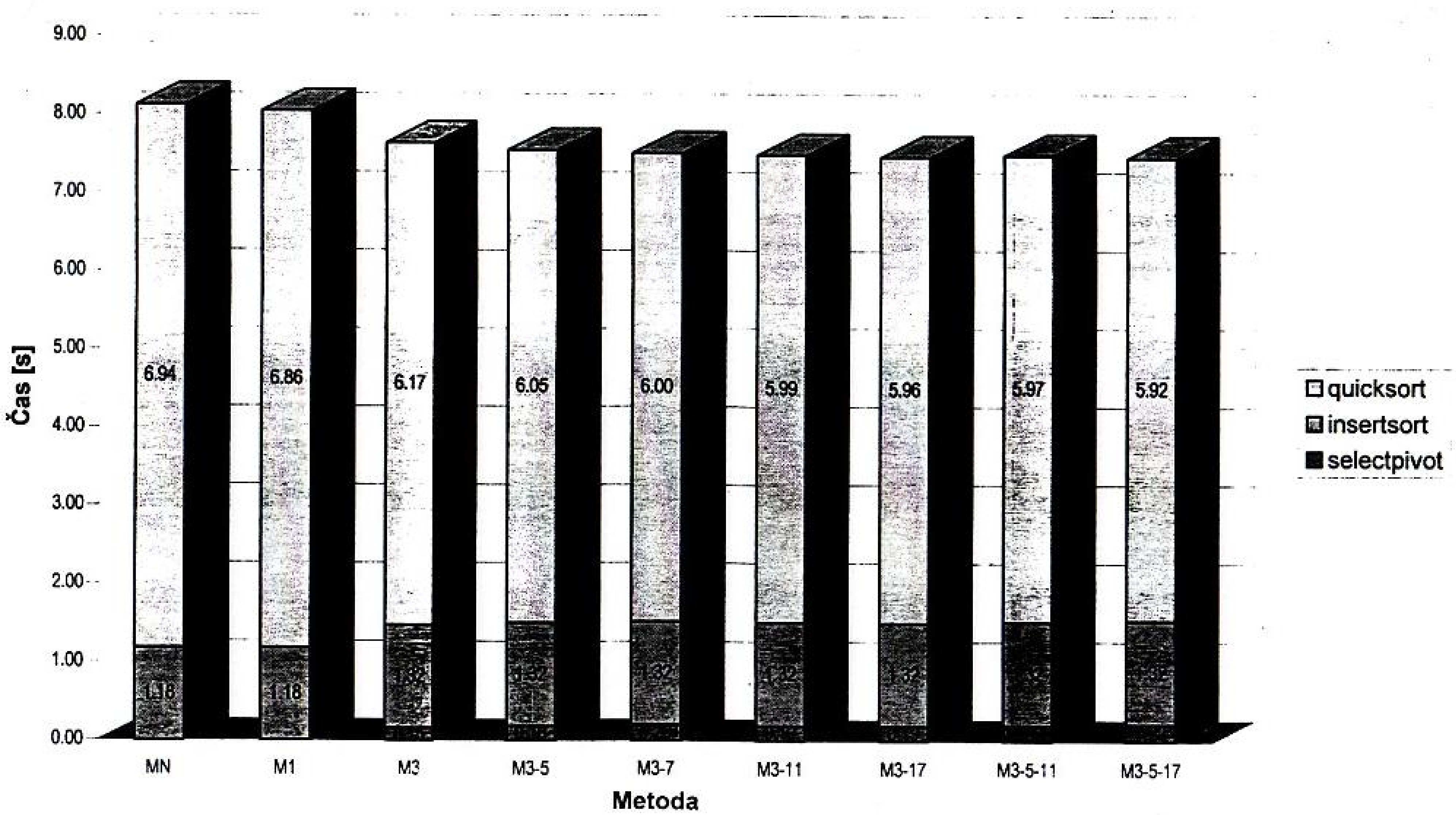
Obrázek 6.1 - Vliv velikosti zbytkového pole na efektivitu třídění



Pro metodu M1 nabývá funkce velikosti zbytkového pole minima při 12 prvcích přičemž považujeme-li 1% relativní odchylku za přijatelnou, pak kterákoliv volba mezi 9 a 18 prvky se jeví jako dobrá. Pro metodu M3 se jeví jako optimální hranice 14 prvků a pokud uvažujeme stejné kritérium přijatelnosti, pak volba mezi 10 a 17 prvky je dobrá.

R. Moric – diplomka (1997)

1 000 000 prvků



MEANSORT

1. volba pivota – 1. člen posloupnosti.

dále: při rozdělování se prvky v každém úseku sčítají

po rozdělení se v každém úseku vypočte průměr – ten je pivotem pro další rozdělování

B-sort

spojení Quicksortu s Bubblesortem

pivot = prostřední prvek

při rozdělování:

po každé výměně se nový prvek v levé části porovná se svým předchůdcem a případně se vymění

analogicky v pravé části s následníkem

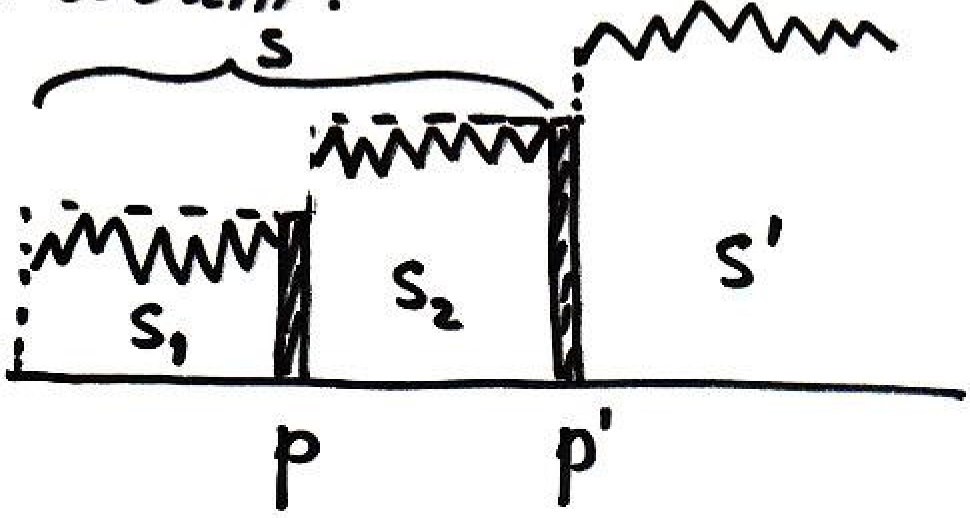
není-li v některé části žádná výměna, je setříděná a dále se nedělí

je vhodný pro skoro setříděné posloupnosti.

Zlepšení prostorové složitosti

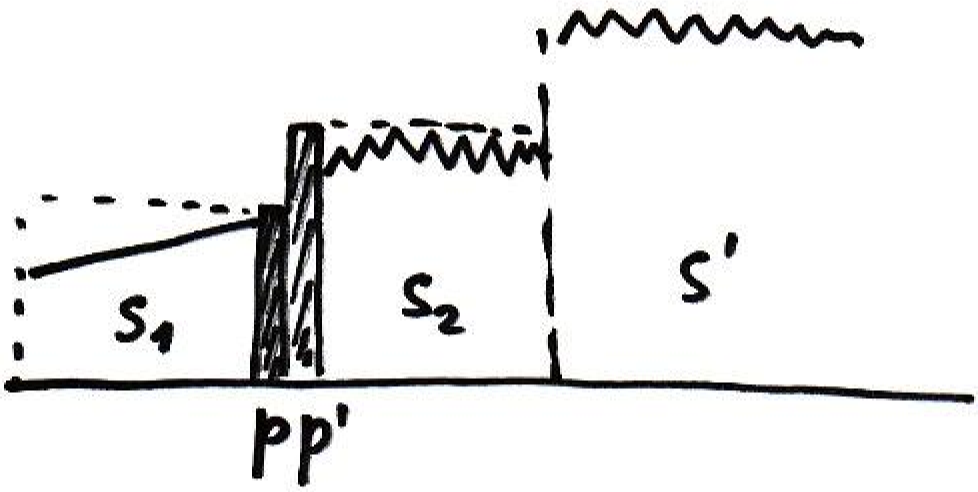
Ďurian (1986) – Quicksort bez zásobníku

rozdělení:



rozdělím podle p' na S a S', p' si zapamatuji
rozdělím podle p na S_1 a S_2, p si zapamatuji
prohodím levý krajní prvek S_2 s p', p' zapomenu
dělím S_1 atd.

návrat z rekurze:



S_1 je setříděná, budu třídit S_2, neznám její meze
najdu místo, kam patří p', a prohodím
Experimentálně je to 4–8% pomalejší než se zásobníkem