Gmail Axel Zedigh <axel.zedigh@gmail.com>

(inget ämne)

1 meddelande

Axel Zedigh <axel.zedigh@gmail.com> 28 mars 2019 15:38
Till: Axel Zedigh <axel.zedigh@gmail.com>

Labb 06 Analys - Axel Zedigh

Del 1

Timeit

- **stmt:** den kod man vill mäta tid på
- **setup:** kod som ska köras innan stmt
- **number:** antal gånger som man vill köra stmt
- Timeit ger den tid (i sekunder) det tar att exekvera *stmt number*-antal gånger.
  Ex:

```
print(timeit(stmt=kod2, setup=kod1, number = 10000)
```

Tidtagning

Linjärsökning i osorteradlista

N (rader)	Varv	Tid(s)	Teori	Praktik	Tid/1000varv
500	100000	3.7192			0.0372
1000	100000	8.3033			0.083
10000	100000	99.48			0.9948
1000000	1	5.9114	$O(n)$	$O(n)$	0.0591

Sökning med binärsökning i sorterad lista

N (rader)	Varv	Tid(s)	Teori	Praktik	Tid/1000varv
500	100000	0.4101			0.0041
1000	100000	0.4504			0.0045
10000	100000	0.5988			0.006
100000	100000	0.6876	$O(log_2(n))$	$O(log_2(n))$	0.0069

Sortering med mergeSort

N (rader)	Varv	Tid(s)	Teori	Praktik	Tid/1000varv
500	100	1.6637			16.6368
1000	100	3.3275			33.2747

N (rader)	Varv	Tid(s)	Teori	Praktik	Tid/1000varv
10000	100	37.1597			371.5973
1000000	1	44.3684	$O(Nlog_2N)$	$O(Nlog_2N)$	44368

Sortering med quickSort

N (rader)	Varv	Tid(s)	Teori	Praktik	Tid/1000varv
500	100	1.5611			15.611
1000	100	3.0649			30.649
10000	100	32.4595			324.5952
1000000	1	51.2488	$O(Nlog_2N)$	$O(Nlog_2N)$	51248

Slår upp element i dictionary

N (rader)	Varv	Tid(s)	Teori	Praktik	Tid/1000varv
494	100000	0.0351			0.0004
977	100000	0.0353			0.0004
8355	100000	0.0369			0.0004
72665	100000	0.0369	$O(1)$	$O(1)$	0.0004

- Finns dubletter av artister, kan påverka snabbhet vid sökning (ex går sökning av sista element för 1miljonlåtar snabbt, ev pga att artisten hittas vid ett tidigare tillfälle i listan och inte sist)

Del 2

Teori

Alternativ1

Följande operationer görs i **linjärsökning** av **osorterad lista**:

- n = n:te längsta låten, N = längd på lista
- *Jämför längd*: $\sum_{i=1}^{n}(N-i) = -(1/2)n(n-2N+1)$
- *Kollar i dict*: $\sum_{i=1}^{n} i(N-i) = \left(-\frac{1}{6}n(n+1)(2n-3N+1)\right)$

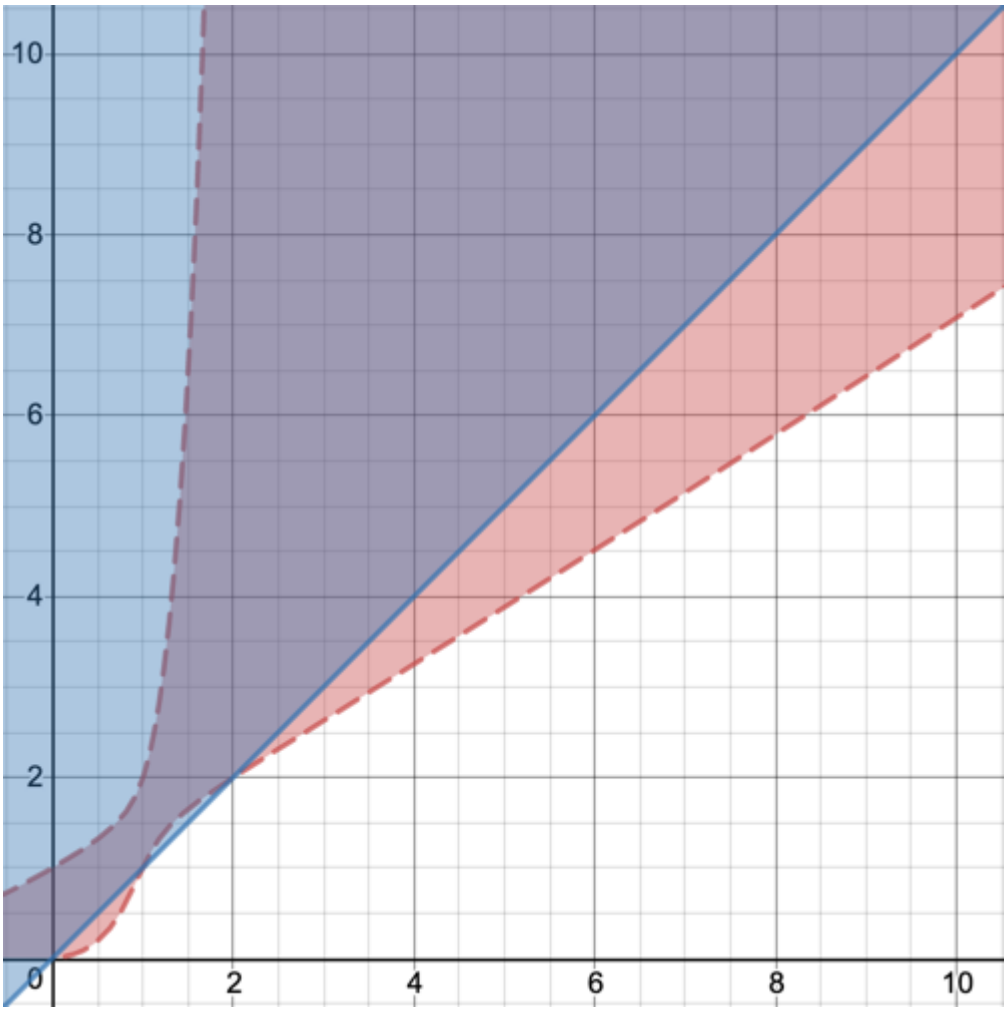
Följande operationer görs då man sorterar lista:

- *Quick/MergeSort*: $O(Nlog_2N)$

För att det ska gå snabbare att sortera listan sen hitta låtlängd måste följande ekvation gälla:

$$-(1/2)n(n-2N+1)+\left(-\frac{1}{6}n(n+1)(2n-3N+1)\right)>N\left(\frac{\log(N)}{\log(2)}\right)$$

Följande graf visar n på x-axel och N på y-axel.
Man kan se att det enligt denna teori alltid är bättre att sortera först



Alternativ2

Följande operationer görs i **linjärsökning** av **osorterad lista**:

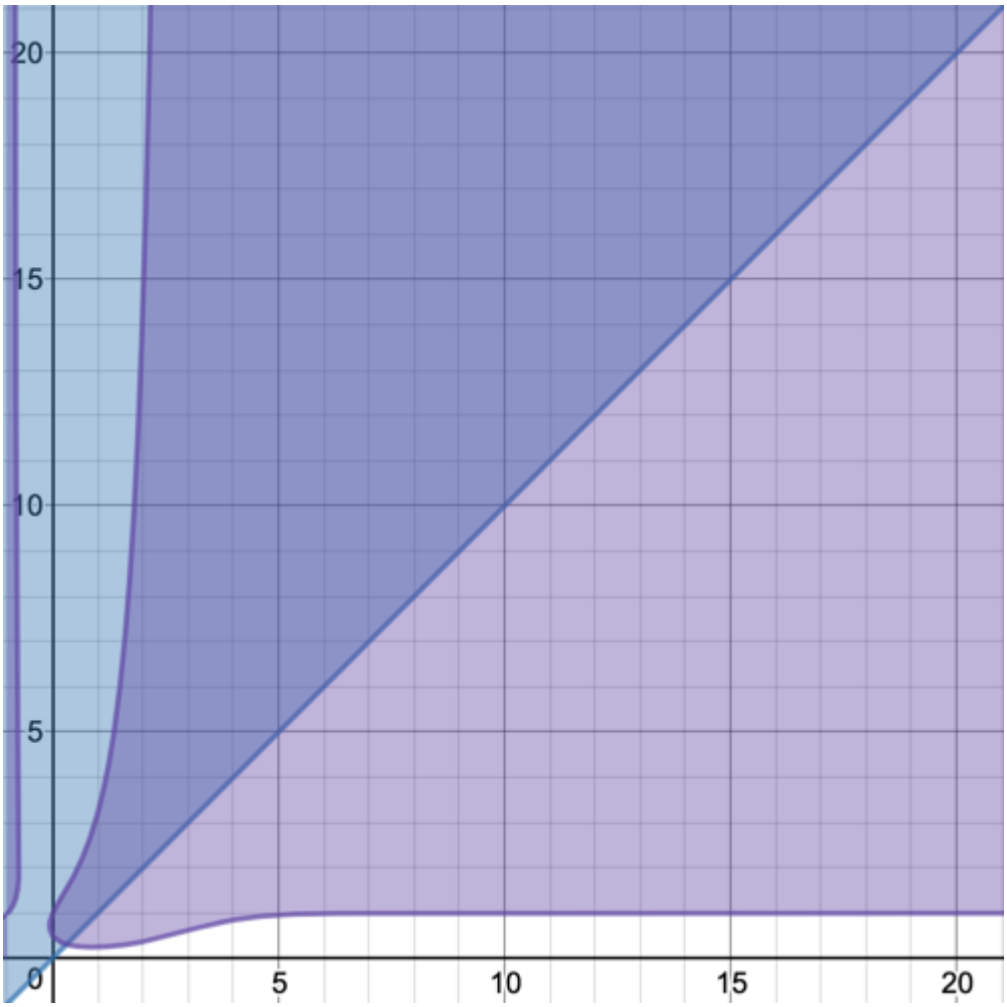
- n = n:te längsta låten, N = längd på lista
- *Jämför längd*: $N \cdot n$
- *Kollar i dict*: $N \cdot n!$

Följande operationer görs då man sorterar lista:

- *Quick/MergeSort*: $O(Nlog_2N)$

För att det ska gå snabbare att sortera listan sen hitta låtlängd måste följande ekvation gälla:
$N \cdot n + N \cdot n! > Nlog_2N$

Följande graf visar n på x-axel och N på y-axel.
Man kan se att det enligt denna teori alltid är bättre att sortera först



Praktik

Linjärsökning i osorterad lista samt Mergesort sen finn låt

N (rader)	Varv	(n:te längst)	Linjärsök	Tid/1000varv	Sortera,sök	Tid/1000varv	n då lika tid
50	1000	1	0.0407		1.4119		
50	1000	2	0.0772		1.4448		
50	1000	3	0.1103		1.4156		
50	1000	5	0.1877		1.4536		
50	1000	10	0.4132		1.4296		
500	10	1	0.0031	0.3146	0.1803	18.0329	
500	10	5	0.0193	1.9265	0.1707	17.0661	
500	10	10	0.0352	3.5168	0.1553	15.5297	
500	10	100	0.4161	41.6091	0.1588	15.8796	ca 50
1000	10	1	0.0086	0.8617	0.3244	32.4374	
1000	10	5	0.0393	3.9346	0.3427	34.2715	
1000	10	10	0.097	9.6972	0.354	35.4047	
1000	10	100	0.7883	78.8348	0.3508	35.0778	ca 50 (0.3321)
10000	10	1	0.0721	7.2141	3.6288	362.8807	

N (rader)	Varv	(n:te längst)	Linjärsök	Tid/1000varv	Sortera,sök	Tid/1000varv	n då lika tid
10000	10	5	0.3492	34.9197	3.6753	367.5306	
10000	10	10	0.7134	71.337	3.7209	372.0922	
10000	10	100	7.1875	718.7514	3.6325	363.2514	ca 50 (3.6154)

Ju mindre listan är desto snabbare går det att linjärsöka direkt i listan.
Ju mindre n:te längsta låt man vill ta ut desto snabbare är linjärsök än sortering.

Man kan se att teorin inte stämmer med praktiken…

Filer i mapp

```
Analys.md
L06-1.py
L06-test.py
README.md
merge.py
sang-artist-data.txt
sang100000rader.txt
sang10000rader.txt
sang1000rader.txt
sang500rader.txt
sang50rader.txt
testresultat.txt
unique100000rader.txt
unique10000rader.txt
unique1000rader.txt
unique500rader.txt
unique_tracks.txt
```