Algoritmer & Datastrukturer
Aflevering 1

Anders Bjerg Pedersen, Hold 2

14. februar 2007

Opgave 2-1

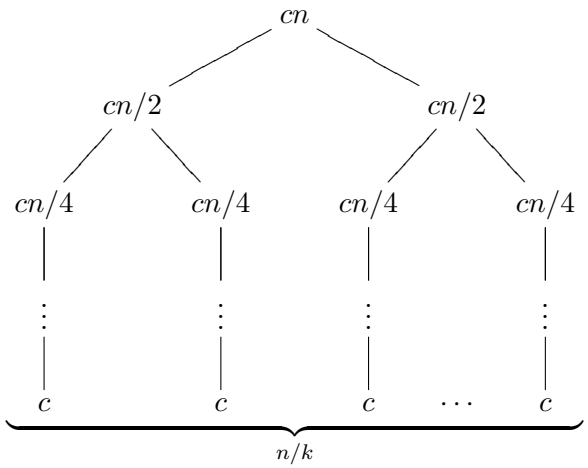
a)

Vi ved fra tidligere, at insertion sort i værste tilfælde vil bruge $an^2 + bn + c$ tid. Dette svarer til, at insertion sort er $\Theta(n^2)$. Denne proces skal gentages $\frac{n}{k}$ gange på lister af fast længde k:

$$\frac{n}{k} \cdot (ak^2 + bk + c) = akn + b + \frac{c}{k} = \Theta(kn).$$

b)

Vi ved, at merge-funktionen er $\Theta(n)$. Vi splitter nu vores liste op i $\frac{n}{k}$ dellister af længde k. Det antages, at insertion sort nu har sorteret listerne. Efter første merge-omgang, der tager tiden cn at flette, er der $\frac{n/k}{2}$ lister tilbage og så videre deropad. Vi skal altså i alt flette lister $\lg(n/k)$ gange. Flettealgoritmen skal altså bruge tiden $cn * \lg(n/k)$, dvs. den er $\Theta(n \lg(n/k))$. Rekursionstræet med de $\lg(n/k)$ rækker er angivet i Figur 1 nedenfor.



Figur 1: Rekursionstræ til opgave b).

c)

Sættes $k = \lg(n)$, får vi, at udførselstiden bliver $n \lg(n) + n \lg\left(\frac{n}{\lg(n)}\right)$. Men da $\frac{n}{\lg(n)} < \lg(n)$ for tilpas store n, vil leddet $n \lg(n)$ dominere, derfor vil algoritmen for $k = \lg(n)$ blive $\Theta(n \lg(n))$, altså det samme som den umodificerede version af flettesortering. Da dette er den eneste mulige værdi af k, for hvilket de to algoritmer har samme udførselstid, har vi løst opgaven.

d)

Man ville vælge $k \leq \lg(n)$, da den modificerede flettesorteringsalgoritme ved $k > \lg(n)$ vil have en udførselstid, der er længere end ved almindelig flettesortering.