Intro to Algorithms, COMP-160, Homework #5

Benjamin Tanen, 03/03/2016

1. We want to sort n integers in the range 0 to n^k, where $k = O(1)$, as fast as possible. The integers have length l, and the radix (base) is d. Do not assume these are $O(1)$.

   (a) The time complexity of counting sort is is $\Theta(a) + \Theta(b)$ where $a =$ number of elements and $b =$ range of elements. Thus, complexity is $\Theta(n)+\Theta(n^k+1) = \Theta(n^k)$.

   (b) There is an obvious relationship between d and l (if d increases, l decreases and vice versa). It is also known (commonly used relation) that, because d is the radix or base, we can describe the length l in terms of d by saying $l = \log_d(m)$ where m is the range of our numbers. Because of the specifics of our range (0 to n^k), we can relate l and d with: $l = \lfloor \log_d(n^k) + 1 \rfloor$

   (c) We know the time complexity of radix sort is $\Theta(l \cdot (n + d))$ since we are sorting n elements into the d buckets l times. Therefore, since we know that $l = \lfloor \log_d(n^k) + 1 \rfloor$ and both d and k are constant, the complexity of radix sort is $\Theta(\log_d n^k \cdot (n + d)) = \Theta(\log_{\Theta(1)} n^{\Theta(1)} \cdot (n + \Theta(1))) = \Theta(n \cdot \log n)$.

   (d) Since we know n and k, we can optimize our work by increasing our base d based on these parameters. If we make $d = n$, we can see that $l = \lfloor \log_n(n^k) + 1 \rfloor = k + 1$. Using this new value of l, we can express the time complexity of radix sort as $\Theta((k + 1)(n + d)) = \Theta(\Theta(1)(n + n)) = \Theta(n)$. Thus, if we can optimize our base to be equal to n, the time complexity of radix sort becomes linear.