605.744 Information Retrieval
Paul McNamee

Short Problem Set (Module 2)

Note: in general, when I ask you to calculate something, please show work; it often allows me to give partial credit for answers that are close by not entirely correct. And I may not give full credit for answers that are actually correct, but where there is insufficient or no justification.

1. [10%] What advantage(s) do balanced binary trees have over hashtables when used to store our dictionary of indexing terms?

2. [30%] Character n-gram overlap is used for both automated spelling correction and name matching (i.e., deciding whether two names might be the same, a common database problem known as "record linkage"). Using a character 3-gram representation, how many distinct n-grams do "CHEONGSONG" and "CHEONMACHONG" have in common? What is the Dice-coefficient score for these two strings using 3-grams? What is the Dice score using 4-grams instead? Which score is higher? Note: although there is nothing conceptually wrong in doing so, for this problem, do not use "padded" n-grams (*e.g.,* '$' or '_' symbols marking the beginning and end of the strings).

2. [30%] Compute the edit distance (or Levenshtein distance) for these two pairs of strings: (a) "CHEBYSHEV" and "TSCHEBYSCHEF"; and (b) "LEVINSTINE" and "LEVENSHTEIN". Then report a sequence of transformations for that cost that converts one string into the other. You should use unit costs for each operation: insertion, deletion, or substitution; that is, each step has a cost of 1. Note, you do not need to write a program or produce any code for this problem – these examples can be easily determined by pencil and paper – you do not need to construct a table as the example in the textbook.

Trivia: Computing edit distance is a classic dynamic programming problem with an $O(N^2)$ complexity. However, the decision problem "Do strings *x* and *y* (say each of length N) have an edit distance <= *k* can be solved in O(*k*N), which is subquadratic. See: Esko Ukkonen's paper 'Algorithms for approximate string matching' (1985).

3. [30%] Following the method described in the textbook (or lecture materials), calculate Soundex codes for the strings: (a) "Stanford" and (b) "Georgetown"? Show intermediate steps to produce the final code.