Problem 1. Natural language queries [10 points]

To do better than Google, we propose to let the users use natural language to form their queries. So, if the user wants to know who the Prime Minister of Singapore is, the user will type in "Who is the Prime Minister of Singapore?" instead of "prime minister singapore" as is currently done. Instead of displaying some sentence fragments containing the query words together with each returned link, the user will be shown a complete sentence containing the answer together with each returned link.

Give one advantage and one disadvantage (with supporting arguments) of the new system compared to current search engines. Discuss the appropriateness of the new system as a general search engine.

Solution: Advantage: User who are not familiar with current search engine tools, e.g. the elderly who are not familiar with computers may find natural language easier to use. Another advantage is that natural language may allow more precise queries since it is more expressive.

Disadvantage: More computing power is required, resulting in either slower response or higher cost in equipment.

Natural language interface appears to be useful only infrequently. The end-to-end principle would suggest constructing a specialized system for this, rather than using it for a general purpose search engine.

Problem 2. Scoring sentences [20 points]

For our system, we will retrieve documents in the same way as current search engines. We will then further process the top k documents in order to rank each sentence in the documents. We will rank the sentences by their *edit distance* from the query sentence. The edit distance of a sentence s from the query q is the cost of the least expensive sequence of operations required to sequentially transform the query into the sentence. The operations are started at the first position of both the query and sentence. Let i be the current position in the query and j be the current position in the sentence. The operations and effects are:

- Delete: delete the current word in the query - increment i but leave j the same, at a cost of 1.
- Insert: use the word from the current position in the sentence - increment j but leave i the same at a cost of 1.
- Replace: replace the current word in the query with the word in the sentence - increment i and j at a cost of 1.
- Copy: copy the current word in the query to the current word in the sentence - increment both i and j at the cost of 0.

(a) Give a dynamic programming algorithm (pseudo-code) for computing the edit distance from a query to a sentence.

Solution:

$C[0,0] = 0$
$C[i,0] = i$ for $i = 1, \ldots, m$
$C[0,j] = j$ for $j = 1, \ldots, n$
for $i = 1$ to m **do**
  for $j = 1$ to n **do**
    if $q[i] = s[j]$ **then**
      $c = 0$
    else
      $c = 1$
    $C[i,j] = \min\{C[i-1,j]+1, C[i,j-1]+1, C[i-1,j-1]+c\}$

The edit distance is in $C[m,n]$.

(b) Write down the loop invariants for the loops in your code.

Solution: For the outer loop: $C[k,l]$ correctly represents the edit distance for transforming $q[1 \ldots k]$ to $s[1,l]$ for all $k < i$ and $0 \leq l \leq n$.
For the inner loop: $C[k,l]$ correctly represents the edit distance for transforming $q[1 \ldots k]$ to $s[1,l]$ for all $k < i$ and $0 \leq l \leq n$. $C[i,l]$ also correctly represents the edit distance for transforming $q[1 \ldots i]$ to $s[1,l]$ for $l < j$.

Problem 3. Improve speed [20 points] Assume that your algorithm for computing the edit distance has a running time of $\Theta(mn)$ where m is the length of the query and n is the length of the sentence. We would like to improve the running time of the system as much as possible.

We will first derive a lower bound for the edit distance. Let $w_1, \ldots, w_k$ be the words in the vocabulary. Let $c_q(w_i)$ be the number of occurences of w_i in the query and $c_s(w_i)$ be the number of occurences of w_i in the sentence. Let $d(q, s)$ be the edit distance between query q and sentence s.

(a) Argue that $\sum_{i=1}^{k} |c_q(w_i) - c_s(w_i)| \leq 2d(q, s)$.

Solution: Start with an empty sentence construct s from q by following the sequence of operations. Each delete operation on word w_i can increase $|c_q(w_i) - c_s(w_i)|$ by at most 1. Each insert operation of word w_i also increase $|c_q(w_i) - c_s(w_i)|$ by at most 1. Replace operation of w_i by w_j can increase $|c_q(w_i) - c_s(w_i)|$ by at most 1 and $|c_q(w_j) - c_s(w_j)|$ by at most 1. Finally copy operation does not change any cost. Since $d(q, s)$ is the number of delete, insert and replace operations and the cost of each such operation is bounded by 2, we have the required inequality.

(b) Describe how the inequality $\sum_{i=1}^{k} |c_q(w_i) - c_s(w_i)| \leq 2d(q, s)$ can be used to possibly improve the overall running time of the system assuming that you wish to return the top 10 sentences with the smallest edit distances.

Solution: Note that $\sum_{i=1}^{k} |c_q(w_i) - c_s(w_i)|$ can be computed in time $O(m+n)$ compared to $\Theta(mn)$ for edit distance. For each sentence, we first compare $\sum_{i=1}^{k} |c_q(w_i) - c_s(w_i)|/2$ with the 10th best edit distance found so far. If it is larger, we do not need to compute the actual edit distance. Otherwise, we compute the edit distance. If the result is smaller, we replace the 10th smallest edit distance with the new distance and sentence (can be efficiently maintained using a heap). Sorting the sentences by $\sum_{i=1}^{k} |c_q(w_i) - c_s(w_i)|$ initially so that the smallest values are tested first is also likely to reduce the number of edit distance computations required.

Problem 4. Alternative formulation [20 points] Your colleague pointed out that many sentences may have large edit distance from each other despite having the same meaning. For example, "Lee Hsien Loong is the Prime Minister of Singapore" and "The Prime Minister of Singapore is Lee Hsien Loong". He is worried that the method would not work well because of that.

(a) Give a plausible reason for why the method is likely to work well for some types of queries despite his concern.

Solution: Common facts are often written in different ways in different documents by different people. If the fact is often required (also often queried), is is likely that it is written in a very similar way in some document simply because of the large number of documents available on the web.

(b) Your colleague would like to give a different formulation. He reasons that words that are close together in the query should generally also be close together in an answer sentence but large groups of them may be shifted by a large distance from the corresponding position in the query.

He propose the following measure of goodness for a sentence. Let $s_1 s_2 \ldots s_n$ be the sentence, where s_i denotes word at position i in the sentence. Let $\pi(i)$ be a permutation function such that $s_{\pi(1)} s_{\pi(2)} \ldots s_{\pi(n)}$ is a permutation of the original sentence. The cost of a permutation π is $\sum_{i=1}^{n-1} c_q(s_{\pi(i)}, s_{\pi(i+1)})$ where $c_q(w_i, w_j)$ is the smallest difference in the positions of words w_i and w_j within the query q if both w_i and w_j exists in q, or $c_q(w_i, w_j) = C$ otherwise (for some constant C). The goodness of a sentence (smaller is better) is the cost of the best permutation: $\min_\pi \sum_{i=1}^{n-1} c_q(s_{\pi(i)}, s_{\pi(i+1)})$.

Reduce the problem of computing the goodness of a sentence to a travelling salesman problem (what are the vertices, edges and edge weights?).

Solution: Construct a vertex for each word s_i. For each pair of words s_i and s_j, construct and edge (s_i, s_j) with weight $c_q(s_i, s_j)$. Add a dummy vertex with edges of weight 0 to every vertex. The solution of the travelling salesman problem on this graph gives the goodness of the sentence.

(c) Your colleague said, "Oh no, travelling salesman is NP-complete. Since my problem can be represented as a travelling salesman problem, it is intractable". Comment on that statement.

Solution: The statement is wrong. To prove that it is intractable, need to do a reduction from travelling salesman to the problem.

Problem 5. Does natural language queries work? [20 points] To evaluate the new system, you hired some NUS undergraduate students to help you do an experiment. Each student is asked to search for the answer to an information need scenario by

- First use a normal search engine and measure the amount of time required to find the answer.
- Then use the new system and measure the amount of time required to find the answer for the same information need.

(a) Find two weaknesses in the experiment setup described above.

Solution: One weakness is that the order is fixed - the normal search engine is used before the new system. This gives unfair advantage to the new system. Another weakness is that the system is only tested on undergrads, hence any conclusion is likely to be valid only for undergrads and may not hold for a different population.

(b) Redesign the experiment to remove the weaknesses.

Solution: One possible experiment is to randomly partition the participants into two groups and give a different system to each group. To handle the population issue, one way would be to try to get a sample that is representative of common computer users, rather than to use undergrads.

(c) Describe how hypothesis testing can be done to check whether the measured effect is statistically significant. Describe the null and alternative hypotheses and the type of test to be used.

Solution: A two sample t test (or Z test) can be used. A reasonable null hypothesis, is that there is no difference in the average amount of time required to find the answer for the two systems. A reasonable alternative hypothesis is that a shorter average time is required for the new system.

Problem 6. Does edit distance work well? [10 points] Your colleague is not convinced that edit distance works well for this problem. Design an experiment to evaluate whether it works well. Specify the statistical hypothesis testing procedure that you would use as well.

Solution: To convince him, we probably need to find a reasonable alternative to compare against. If a state of the art system doing the same thing exists, it would serve well as a comparison. If not, we may want to use a simpler method such as ranking with $\sum_{i=1}^{k} |c_q(w_i) - c_s(w_i)|$ to compare against. We can repeat the previous experiment, replacing the conventional search engine with one that uses natural language queries and the alternative ranking method instead of the edit distance. The same hypothesis testing procedure as in the previous question would be okay.