1. **The Multiple-stack Algorithm as A* Search**: Show that the multiple-stack algorithm of Section 6.6.1 is an instance of the generic A* algorithm.

Specifically, the A* search in Section 6.3 is concerned with finding the best hypothesis \( \hat{w}_n \) among all hypotheses \( w_n \) of a given length \( n \). The parameter \( k \) is associated with the length of a partial hypothesis \( w_k \), and a goodness function \( F(w_k) = g(w_k) + d(w_k) \) is defined for each partial hypothesis. Starting with the empty hypothesis in a stack, the current best partial hypothesis \( \hat{w}_k \) is extended at each iteration to \( \hat{w}_k+1 = \hat{w}_k \| \hat{w}_{k+1} \), until the best partial hypothesis attains a length \( n \).

However, given the acoustics \( a_m = a_1, \ldots, a_m \), the goal in Section 6.6.1 is to find \( \hat{w}_n = \arg \max_w P(a_m, w) = \arg \max_w \log P(a_m, w) \), among all word sequences \( w \) regardless of their length, and \( n \) is not provided a priori.

Let us contemplate how to extended the notation of Section 6.3 to this problem. Define a partial hypothesis to be a pair \( \langle l, w_k \rangle \), where \( 0 \leq l \leq m \), and let

\[
\begin{align*}
g(\langle l, w_k \rangle) &= \max_{a_l} \sum_i \log P(a_i, w_i), \quad 0 = l_0 \leq l_1 \leq \ldots \leq l_k = l. \\
\end{align*}
\]

Define the length of a partial hypothesis \( \langle l, w_k \rangle \) to be \( l \), not \( k \), so that the length of a complete hypothesis is \( m \). Extending an \( l \) length hypothesis to a \( l + 1 \) length hypothesis therefore does not necessarily entail extending \( w_k \) to \( w_{k+1} \). we could have \( \langle l, w_k \rangle \rightarrow \langle l + 1, w_k \rangle \) just as easily as \( \langle l, w_k \rangle \rightarrow \langle l + 1, w_{k+1} \rangle \). To see that the multiple-stack algorithm is an A* algorithm, let

\[
\begin{align*}
d(\langle l, w_k \rangle) &= \max_{z} \sum_{l_0, \ldots, l_{k-1}} \log P(a_{l_i-1}^i, z_i), \quad l = l_0 \leq l_1 \leq \ldots \leq l_{|z|} = m, \\
\end{align*}
\]

where \( |z| \) is the length of the word sequence \( z \). Note that \( d(\langle l, w_k \rangle) \) does not depend on \( w_k \)!
(a) Show that if an A* search is conducted using a single stack, with \( g(⟨l, w^k⟩) + d(⟨l, w^k⟩) \) as the goodness of a partial hypothesis \( ⟨l, w^k⟩ \), and is stopped the first time an \( m \) length hypothesis \( ⟨m, \hat{w}^1⟩ \) percolates to the top, then \( \hat{w}^1 \) is indeed the most likely word sequence.

(b) Derive a relationship between \( d(⟨l, w^k⟩) \) and \( g^*(l) \) as defined in Equation (11) on p100.

(c) Show that sorting the stack entries during this A* search according to

\[
F^*(⟨l, w^k⟩) = g(⟨l, w^k⟩) - g^*(l),
\]

will also lead to the discovery of the same \( \hat{w}^1 \), i.e. \( ⟨m, \hat{w}^1⟩ \) will be the first \( m \) length hypothesis to percolate to the top of the (single) stack.

Argue based on these results that when the multiple-stack algorithm on p101 stops, the top entry in the \( m \)-th stack is the most likely word sequence \( \hat{w}^1 \).

2. N-best Paths Using the Multiple-stack Algorithm: Modify the multiple-stack algorithm of Section 6.6 to obtain the \( N \) best hypotheses instead of only the best. In particular, assume that the conditions (a), (b) and (c) of Section 6.6.1 on p100 are satisfied.

(a) Minimally modify the algorithm of Section 6.6.1 to obtain the two best paths.

(b) Check if the extension of Section 6.6.2 will hold for your new algorithm.

(c) Generalize your modification for any \( N \geq 2 \) best paths.

Discuss whether the extension to the actual multiple-stack algorithm of Section 6.6.3, when the assumptions (a), (b) and (c) do not hold, will be possible for your new algorithm. What are the pitfalls for \( N \geq 2 \) that were not there for \( N = 1 \)?