A: Theory

We consider a unigram as a single word and a bigram as an ordered pair of words in the form $\{w_1, w_2\}$ separated by space. We have two dictionaries Pw and $Pw2$ that contains unigrams and bigrams respectively along with their frequencies. We use the following notations.

Notations	Meaning
Pw[w]	frequency of w in Pw dictionary
Pwseg[w]	frequency of w in Pwseg dictionary (extracted from wseg_simplified_cn)
Pw2[w_1, w_2]	frequency of $\{w_1, w_2\}$ bigram in Pw2 dictionary
V	Total number of different unigrams or bigrams
N	Sum of frequencies
$voc[w]$	Number of different bigrams started by w

There can be different cases that might happen when we analyze any bigram. However, when we analyze the probability of the very first word of the document or when we can not find the first word w_1 of an ordered pair $\{w_1, w_2\}$ in the dictionary Pw, we only use the unigram model to assign the probability of the word.

According to the unigram model, The probability of a word w is given by:

$$P(w) = \frac{Pw[w] + 1}{Pw.N + Pw.V + 1} \tag{1}$$

if it is found in Pw or

$$P(w) = \frac{1}{Pw.N + Pw.V + 1} \tag{2}$$

if it is not found in Pw

However, we use bigram model for every ordered pair $\{w_1, w_2\}$ when w_1 is found in Pw. According to the bigram model, the conditional probability of w_2 given that we already observed w_1 is calculated. There can be any of the following three cases.

(i) Bigram $\{w_1, w_2\}$ in Pw2
In this case there is a connection between two known words. We consider conditional probability for w_2 as:

$$P(w_2|w_1) = \frac{Pw2[w_1, w_2] + 1}{Pw[w_1] + Pw2.voc[w_1] + 1} \tag{3}$$

(ii) Bigram not in Pw2, w_2 in Pw
Although there is no observed connection, both are known words. In this case, we assign the conditional probability based on the frequency of w_2 in Pw.

$$P(w_2|w_1) = \frac{Pw[w_2] + 1}{(Pw[w_1] + Pw2.voc[w_1] + 1)(Pw.N + Pw.V + 1)} \tag{4}$$

(iii) Bigram not in Pw2, w_2 is not in Pw
In this case, w_2 is completely unknown. Instead of assigning a zero probability, we assign probability of w_2 as

$$P(w_2|w_1) = \frac{1}{(Pw[w_1] + Pw2.voc[w_1] + 1)(Pw.N + Pw.V + 1)} \tag{5}$$

B: Data Structure

Data Structure	Description
Pw,Pwseg, Pw2	Dictionaries with frequencies of unigrams (Pw and Pwseg) and bigrams (Pw2)
chart	Dynamic programming Table to store the words
entry	A structure {word, start position, log probability, back pointer}
pq	priority queue of entries, priority= lowest starting

C: Algorithm

The algorithm is given below:

Algorithm 1 Word Segmentation

```
Input:  File with non segmented words $Input$,
    unigram dictionary $Pw$,
    wseg unigram dictionary $Pwseg$,
    bigram dictionary $Pw_2$
Output:  File with segmented words
 1: Initialize $chart = \{\}$ and $maxlen \leftarrow$ longest bigram length
 2: Initialize priority queue with starting position as priority
 3: for $l$ in range(1,...,maxlen) do
 4:   $w_1 \leftarrow$ substring input[0,...,l-1]
 5:   if $w_1$ is found in Pw or Pwseg then
 6:     calculate $P(w_1)$ according to unigram model
 7:     add $\{w_1, 0, P[w_1], None\}$ into pq
 8: if nothing inserted in last for loop then
 9:   $w_1 \leftarrow$ input[0]
10:   calculate $P(w_1)$ according to unigram model
11:   add $\{w_1, 0, Pw[w_1], None\}$ into pq
12: while pq not empty do
13:   $item \leftarrow$ highest priority item from pq
14:   $end \leftarrow item.length + item.start$
15:   if $item.probability > chart[end].probability$ then
16:     $chart[end] \leftarrow item$
17:   $nextstart \leftarrow end + 1$
18:   if $w_1$ is found in Pw then
19:     for $l$ in range(1,...,maxlen) do
20:       $w_2 \leftarrow$ substring input[nextstart,...,nextstart+l-1]
21:       if bigram $\{w_1, w_2\}$ is found in Pw2 then
22:         calculate $P(w_2|w_1)$ according to bigram model
23:         add $\{w_2, nextstart, P(w_1) * P(w_2|w_1), item\}$ into pq
24:       else if $w_2$ is found in Pw or Pwseg then
25:         calculate $P(w_2)$ according to unigram model
26:         add $\{w_2, nextstart, P(w_1) * P(w_2|w_1), item\}$ into pq
27:     if nothing inserted in last for loop then
28:       $w_2 \leftarrow$ input[nextstart]
29:       calculate $P(w_2)$ according to unigram model
30:       add $\{w_2, nextstart, P(w_1) * P(w_2|w_1), item\}$ into pq
31:   else
32:     for $l$ in range(1,...,maxlen) do
33:       $w_2 \leftarrow$ substring input[nextstart,...,nextstart+l-1]
34:       if $w_2$ is found in Pw or Pwseg then
35:         calculate $P(w_2)$ according to unigram model
36:         add $\{w_2, nextstart, P(w_1) * P(w_2), item\}$ into pq
37:     if nothing inserted in last for loop then
38:       $w_2 \leftarrow$ input[nextstart]
39:       calculate $P(w_2)$ according to unigram model
40:       add $\{w_2, nextstart, P(w_1) * P(w_2), item\}$ into pq
41: $item \leftarrow$ chart[end]
42: while $itemisnotNone$ do
43:   add item.word to wordlist
44:   $item \leftarrow item.backpointer$
45: print wordlist in reverse order
```