Algorithms for Generating Permutations and Combinations

Section 6.3

Prof. Nathan Wodarz

Math 209 - Fall 2008

Contents

1 Listing Permutations and Combinations

1.1 Listing Permutations and Combinations

Listing Permutations and Combinations

- Goal: List all permutations and/or combinations of a set
- Problems:
  - Lots of them
  - How can we be sure all are listed?
  - Idea: Put some sort of *order* on permutations/combinations

1.2 Lexicographic Order

Lexicographic Order

- Will use *lexicographic order* to list all permutations and/or combinations
- Similar to *dictionary (alphabetical) order*
  - If Word A is shorter than Word B, and every letter of Word A occurs in the same place in Word B, Word A comes before Word B (”compute” and ”computer”)
  - If the first letter that differs in Word A comes before the corresponding letter in Word B, then Word A comes before Word B (”math” and ”matter”)
- For strings $\alpha = s_1 s_2 s_3 \cdots s_p$ and $\beta = t_1 t_2 t_3 \cdots t_q$ taken from the set $\{1, 2, 3, \ldots, n\}$
  - For example, $\alpha = 1742$ and $\beta = 18285$ are strings over $\{1, 2, 3, 4, 5, 6, 7, 8\}$
  - We write $\alpha < \beta$ (α is *lexicographically less than* β) provided that
    - $p < q$ and $s_i = t_i$ for $1 \leq i \leq q$ (e.g., $\alpha = 1732$ and $\beta = 173245$)
    - For the first i such that $s_i \neq t_i$, $s_i < t_i$ (e.g., $\alpha = 28473$ and $\beta = 2848$)

Lexicographic Order and Permutations

Example. For the following 4−permutations from the set $\{1, 2, 3, 4, 5, 6, 7\}$, find the permutation that immediately follows them in lexicographic order

1. 1234 is followed by
2. 4567 is followed by
3. 5437 is followed by
4. 7654 is followed by

Lexicographic Order and Combinations

- We will always list a given combinations the order $s_1 < s_2 < \cdots < s_p$

Example. For the following 4−combinations from the set $\{1, 2, 3, 4, 5, 6, 7\}$, find the combination that immediately follows them in lexicographic order

1. 1234 is followed by
2. 3467 is followed by
3. 4567 is followed by

2 Generating Permutations and Combinations

2.1 Generating Combinations

Generating Combinations

- Given a string $\alpha = s_1 \cdots s_r$, to find the next string (as a combination)
  - Find the rightmost element not at its maximum value
  - Don't change anything before that element
  - Increment the element found above
  - Each additional element is one more than the previous
- For 5-combinations of $\{1, 2, 3, 4, 5, 6, 7, 8\}$:
  - We will find successor of 13578
  - What is rightmost element not at its maximum?
  - Increase that by 1
  - List remaining elements in order.
  - Successor is

Algorithm for Generating Combinations

List all r-combinations of $\{1, 2, \ldots, n\}$ in increasing lexicographic order.

```
Input: r, n
Output: All r-combinations of {1,2,...,n}
        in increasing lexicographic order

1.   combination(r,n){
2.    for i = 1 to r
3.      s_i = i
      // Print the first r-combination
4.    print(s_1, s_2, ..., s_r)
5.    for i = 2 to C(n,r) {
6.      m = r
7.      max_val = n
8.      while (s_m == max_val){
          // Find the rightmost element
          not at maximum value
9.        m = m - 1
10.       max_val--
11.     }

        // Increment the above rightmost
        element
12.     s_r++
        // All others are the successors
        of this element
13.     for j = m + 1 to r
14.       s_j = s_{j-1} + 1
        // Print this new combination
15.     print(s_1, s_2, ..., s_r)
16.   }
17. }
```

2.2 Generating Permutations

Generating Permutations

- Given a string $\alpha = s_1 \cdots s_r$, to find the next string (as a permutation)
  - Find the rightmost place where digits increase
  - Don't change anything before that element
  - Make the left element of the pair as small as possible but still larger than it was
  - Each additional element is as small as possible
- For permutations of $\{1, 2, 3, 4, 5, 6\}$:
  - We will find successor of 135642
  - What is rightmost place the digits increase?
  - Increase the leftmost to be smallest possible
  - List remaining elements in smallest to largest.
  - Successor is

Algorithm for Generating Permutations

List all permutations of $\{1, 2, \ldots, n\}$ in increasing lexicographic order.

```
Input: n
Output: All permutations of {1,2,...,n} in increasing lexicographic order

1.   permutation(n){
2.    for i = 1 to r
3.      s_i = i
      // Print the first permutation
4.    print(s_1, s_2, ..., s_r)
5.    for i = 2 to n! {
6.      m = n - 1
7.      while (s_m > s_{m+1})
          // Find the last decrease
8.        m = m - 1
9.      k = n
10.     while (s_m > s_k)
          // Find the last element
          greater than s_m
11.       k = k - 1


12.     swap(s_m, s_k)
13.     p = m + 1
14.     q = n
15.     while (p < q) {
          // swap s_{m+1} and s_n, swap s_{m+2}
           and s_{n-1}, ...
16.       swap(s_p, s_q)
17.       p++
18.       q--
19.     }
        // Print this new permutation
20.     print(s_1, s_2, ..., s_r)
21.   }
22. }
```

Summary

Summary

You should be able to:

- Work with lexicographic ordering
- Find the next combination and/or permutation of a given one