Generating Permutations and Combinations

CIS 2910

Joe Sawada - University of Guelph

Generating algorithms

So far we have only looked at the counting problem: How many items are there?

Now, we want to handle the question: What are all the possibilities?

Definition

A **generation algorithm** is an algorithm that exhaustively lists all instances of a specific object, so that each instance is listed exactly once.

Considerations for generation algorithms:

- representation (how do we represent the object?)
- efficiency (how fast is the algorithm?)
- order of output (lexicographic, Gray code ...)

Order of output

There are two common ways to order output of strings. Lexicographic and Gray code order.

Definition

A string $a_1 a_2 \cdots a_n$ is said to be **lexicographically larger** than another string $b_1 b_2 \cdots b_n$ if for some k we have $a_k > b_k$ and $a_i = b_i$ for all $1 \leq i \leq k-1$.

Lexicographic order is often what you think of as your standard alphabetic order.

Definition

A listing is said to be in **Gray code order** if each successive string in the listing differs by a constant amount. For example, the swapping of elements, or the flipping of a bit.

Orders of ouput

Lexicographic vs Gray code

Lexicographic	Gray code
000	000
001	001
010	011
011	010
100	110
101	111
110	101
111	100

The second list is known as the Binary Reflected Gray Code. Each binary string differs by a single bit flip from the previous string.

Applications

Some instances where a generation algorithm may be very useful.

Traveling SalesPerson

A salesperson wants to travel to 6 cities while on tour. We know that there are $6! = 120$ ways to visit the cities, but which order is cheapest? or minimizes total time/distance travelled? We could compute these values for each of the 120 possibilities.

Subset Sum

Given a set of 10 integers, does it contain a subset whose elements sum to 100? We could check all possible subsets and check their totals.

Generating permutations

Given a permutation, how can we determine the next permutation in the lexicographic ordering?

Observe that once we have a general way to do this we can generate all permutations by starting with $1234 \cdots n$ and successively finding the next largest permutation.

Permutations in lexicographic order

```
perm = array();

Next()
    Update perm[ ] to the next largest permutation;

Main(n: integer)
    for i = 1 to n do perm[i] := i;
    for i = 1 to n! − 1 do Next();
```

Finding the next permutation in Lex order

Examples

What is the next largest permutation for the following:

- 234156? next: 234165
- 234165? next: 234516
- 234516? next: 234561
- 362541? next: 364125

When can we simply exchange the last two characters? If not, what is a general strategy?

Finding the next permutation in Lex order

Algorithm: Next Permutation

Find the next largest permutation of $a_1 a_2 \cdots a_n$.

1. Find the largest position i such that $a_i < a_{i+1}$.
2. Replace a_i with the smallest value in $a_{i+1} \cdots a_n$ that is bigger than the current value of a_i.
3. Reassign the values from the remaining elements $a_{i+1} \cdots a_n$ (and the original value of a_i) into increasing order.

Example

Let's apply the algorithm to the permutation where $n = 9$: 812369754

1. Largest i is $a_5 = 6$.
2. Next largest value to the right of 6 is 7. So we have $81237 \cdots$.
3. The remaining values are 6954, which we put into increasing order to get: 812374569.

Generating Subsets

Recall our first generation algorithm issue: how to represent an object.

We can represent all subsets of an n-set with binary strings of length n.

So now given such a binary string, how can we find the next largest one?

Algorithm: Next Subset

Find the next largest binary string of $b_1 b_2 \cdots b_n$.

1. Find the largest i such that $b_i = 0$ (scan from right).
2. Set $b_i = 1$.
3. Set $b_j = 0$ for all $i + 1 \leq j \leq n$.

For example: $Next(1101\mathbf{0111}) = 1101\mathbf{1000}$

Generating Combinations

To generate r-combinations of the set $\{1, 2, \ldots, n\}$ we could use binary strings with r ones and length n. (eg. 100110010)

However, for this algorithm we represent the r-elements with a sequence of their values given in increasing order. (eg. 1458)

Algorithm: Next Combination

Finding the next largest combination for $a_1 a_2 \cdots a_r$?

1. Find the largest i such that $a_i \neq n - r + i$.
2. Set $a_i = a_i + 1$.
3. **for** $j = i$ **to** r **set** $a_j = a_{j-1} + 1$.

Examples

Suppose that $n = 9$ and $r = 4$:

- Next(1458) $=$ 1459
- Next(1459) $=$ 1467
- Next(3789) $=$ 4567