Background

PROBLEM SOLVED:

Sometimes it is necessary to list combinations[1] of non-repeating elements, either all possible combinations or a subset. This invention lists such combinations in an order expected by humans rather than in a machine order.

KNOWN SOLUTION:

There is an algorithm that lists combinations of non-repeating elements in a machine order. It involves forming an integer where each bit-position indicates the presence or absence of one of the elements in the set. When the integer is incremented, the bit pattern at each step represents one unique combination.

Here is a simple example. Suppose we wish to list all possible combinations of the elements A, B, C and D. In this example, the number of elements 'n' is four. We form an integer 'i' that has n bit-positions. The lowest-order bit in position zero indicates the presence of element A, position one the presence of element B, position two the presence of C and position three the presence of D. The total number of possible combinations[2] including the empty or null combination is 2^n or in this example 16. We increment 'i' from zero to 16 yielding the following bit patterns and combinations:

i bits combination

---------------------

0 0000 - - - -

1 0001 A - - -

2 0010 - B - -

3 0011 A B - -

4 0100 - - C -

5 0101 A - C -

6 0110 - B C -

7 0111 A B C -

8 1000 - - - D

9 1001 A - - D

10 1010 - B - D

11 1011 A B - D

12 1100 - - C D

13 1101 A - C D

14 1110 - B C D

15 1111 A B C D

This is a fast and foolproof method for listing all possible combinations of a set of non-repeating elements. It can also be modified to list subsets of combinations. For example, it can be used to list all combinations containing a specified number 'm' of the elements from the set of 'n' available elements. This is done by counting the number of set bits at each iteration and keeping only those iterations where 'm' bits are set.

We can illustrate by extending the previous example. Suppose we wish to list all combinations that contain two elements of the set A, B, C and D. In this example m=2 and n=4. We iterate 'i' as before but this time we keep only those iterations where the number of set bits equals two:

i bits combination

---------------------

3 0011 A B - -

5 0101 A - C -

6 0110 - B C -

9 1001 A - - D

10 1010 - B - D

12 1100 - - C D

DRAWBACKS OF KNOWN SOLUTION:

The problem with this method is that combinations are listed in a machine order and not in an order that is easy for humans to use. For example in the first illustration a human would likely list the combinations in this order:

combination

-----------

- - - -

A - - -

- B - -

- - C -

- - - D

A B - -

A - C -

A - - D

- B C -

- B - D

- - C D

A B C -

A B - D

A - C D

- B C D

A B C D

And, as the number of elements in the set increases the disparity between machine and human ordering becomes more pronounced.

Summary of Invention

This invention is an algorithm that lists combinations of non-repeating elements in an order that is easy for humans to use. It does this by making several improvements to a machine order algorithm. First, it exploits the fact that when an integer is repeatedly incremented or decremented the less significant bits change more rapidly than the more significant bits. Second, the integer that is incremented in the machine order algorithm is instead decremented and the orders of the set of elements and the combinations formed from them are reversed. Finally, subsets of the set of elements to combine are listed in order from a subset containing zero elements to one containing all the elements. For the previous illustration this invention would list the combinations in this order:

i bits combination

---------------------

0 0000 - - - -

8 1000 A - - -

4 0100 - B - -

2 0010 - - C -

1 0001 - - - D

12 1100 A B - -

10 1010 A - C -

9 1001 A - - D

6 0110 - B C -

5 0101 - B - D

3 0011 - - C D

14 1110 A B C -

13 1101 A B - D

11 1011 A - C D

7 0111 - B C D

15 1111 A B C D

Description

A mathematical combination is a selection of mutually distinguishable elements from a set of elements without regard to their order of arrangement[1]. For example from the set of elements A, B, C and D, the combination (A,B,C,D) is the same as (A,C,B,D). However, it is sometimes desirable to present combinations in an order that is rational and useful to humans. In that case the representation (A,B,C,D) is preferred over (A,C,B,D) because the elements have an order understood by humans (alphabetic in this case) where B precedes C. Likewise with a list of many combinations from the same set of elements: (A,-,C,-) precedes (A,-,-,D); (A,-,C,-) precedes (-,B,C,\_); (A,-,C,-) precedes (A,-,C,D); etc. A further restriction is that we consider only combinations of non-repeating elements (CNE). We do not include combinations of repeating elements like (A,A,C,D), for example, where the element A is repeated twice. The task then is to present CNE in an order that is rational and useful to humans.

All Combinations:

The first algorithm lists all combinations of non-repeating elements (CNE) in an order that is easy for humans to use. It takes as input a set of 'n' elements sorted from most to least prominent. The algorithm is shown as a flow chart in diagram 1. It can be compared to diagram 2, an algorithm that lists all CNE in a machine order. The highlighted nodes in diagram 1 are those that are added or differ from their counterparts in diagram 2. The set of n elements is labeled 's' in diagrams 1 and 2. Similar to the machine algorithm, this algorithm uses the binary representation of an integer labeled 'i' in diagrams 1 and 2 to identify the elements of s that form the combination associated with i.

The algorithm is comprised of three loops. The outer loop increments 'm', the number of elements in the CNE, from zero to n. The m is compared to the count 'p' of bits in the binary representation of i that are set to one. If m matches the count then this i is examined. Otherwise, it is skipped. This filtering using m causes the CNE with zero elements to be presented first, followed by all the CNE with one element, then two elements, etc. until finally the CNE that contains all the elements is presented. This order of presentation is easy for humans to comprehend. It is not the order of the machine algorithm in diagram 2.

The middle loop decrements the integer i. The upper limit of i is calculated in the first step of the algorithm. This limit is the total number 't' of all possible CNE for the given n, which is t=2^n. The loop decrements i from a maximum value of t-1 to zero. The machine algorithm in diagram 2 increments rather than decrements i.

The inner loop increments a loop counter labeled 'j' in diagrams 1 and 2 from zero to n. In this loop each bit of the binary representation of i is examined to identify the elements of the combination 'c' associated with i. If bit j is one and not zero then element (n-1)-j from set s is added to c at position (n-1)-j. Otherwise, the position (n-1)-j is left empty. The index (n-1)-j effectively reverses the order of s and c from their given order. The machine algorithm in diagram 2 does not reverse s and c.

The three necessary aspects of the algorithm are filtering using m, decrementing i and reversing s and c. Together, they present the CNE in an order pleasing to humans and easy to use.

There is a computational cost for filtering using m compared to the machine order algorithm. For situations where n is large and where performance is critical an additional calculation and check can be added to improve performance. The number of possible CNE for a given m is calculated as the binomial coefficient 'b', which is n! divided by m!(n-m)! where '!' is the factorial operator. The number 'q' of CNE with m elements is accumulated and compared with b. Processing of i in the middle loop continues only while q is less than b. Otherwise, m is incremented and the procedure continues. This is shown in diagram 3 where the highlighted nodes are those that are missing from diagram 1.

Combinations of m Elements:

The second algorithm lists the combinations of non-repeating elements (CNE) when 'm', the number of elements in the CNE, is a given. It is identical to the algorithm described above but with the outer loop incrementing over all m removed. This algorithm is shown as a flow chart in diagram 4. A comparable machine order algorithm that adds filtering using m is shown in diagram 5. The highlighted nodes in diagram 4 are those that differ from their counterparts in diagram 5.

The two loops of this algorithm are identical to the middle and inner loops of the first algorithm. The two necessary aspects of this algorithm are decrementing i and reversing s and c. Together, they present the CNE for a given m in an order pleasing to humans and easy to use. There is no computational cost for this algorithm compared to the machine order algorithm because of the extra steps required in the machine order algorithm in order to filter using m. However, this algorithm can also benefit from the performance enhancement described for the first algorithm. This is shown in diagram 6 where the highlighted nodes are those that are missing from diagram 4.

[1] Combinations in the strict mathematical sense, not permutations. For example see: Jeffrey, Alan; page 59; "Handbook of Mathematical Formulas and Integrals"; 1995; Academic Press, Inc.

[2] From a theory of combinatorics, ibid.

Diagrams

diagram1.emf

Diagram 1: Human Order, All Combinations

diagram2.emf

Diagram 2: Machine Order, All Combinations

diagram3.emf

Diagram 3: Human Order, All Combinations with Optimization

diagram4.emf

Diagram 4: Human Order, Combinations of m Elements

diagram5.emf

Diagram 5: Machine Order, Combinations of m Elements

diagram6.emf

Diagram 6: Human Order, Combinations of m Elements with Optimization