CSI 5165 Combinatorial Algorithms — Fall 2010
Computer Science — University of Ottawa

Homework Assignment #1 (100 points, weight 15%)
Due: October 20 at 10:00 a.m. (in lecture)

Part 1: Generating elementary combinatorial objects

1. (10 points) **Simple practice with combinatorial generation algorithms**
   Calculate the result for the following operations. Show your work.

   - Subsets:
     Give the SUCCESSOR and the RANK of 01010110 in the Gray code G^8.
   - k-subsets:
     Give RANK of $\{3, 6, 7, 9\}$ considered as a 4-subset of $\{0, 1, \ldots, 12\}$ in lexicographic and revolving-door order. What is the SUCCESSOR in each of these orders?
   - Permutations:
     Find the rank and successor of the permutation $[2, 4, 6, 7, 5, 3, 1]$ in lexicographic and Trotter-Johnson order.

     UNRANK the rank $r = 56$ as a permutation of $\{1, 2, 3, 4, 5\}$, using the lexicographic and Trotter-Johnson order.

2. (30 points) **Correctness of SUCCESSOR algorithm for Graycodes**
   Prove Theorem 2.2 of the textbook which states that Algorithm 2.3 (also given in class) correctly computes SUCCESSOR for the binary reflected Gray code. You need to state and prove several facts, which will formalize the informal statements given in page 38 of the textbook:

   > Algorithm 2.3 works as follows. If $w(A)$ is even, then the last bit of A (namely a_0) is flipped; if $w(A)$ is odd, then we find the first "1" from the right, and flip the next bit (to the left). The last vector in G^n, which has no successor (added by editor: or sucessor $[0, \ldots, 0]$ in circular order) is $[1, 0, \ldots, 0]$ This corresponds to the set $\{1\}$.

   Hint: Prove the facts by induction on n.

3. (30 points) **Generating k-multisets of an n-set** (Exercise 2.13)
   A *multiset* is a set with (possibly) repeated elements. A k-multiset is one that contains k elements (counting repetitions). Thus, for example, $\{1, 2, 3, 1, 1, 3\}$ is a 6-multiset. The k-multisets of an n-set can be ordered lexicographically, by sorting the elements in each multiset in non-decreasing order and storing the result as a list of length k.

   When writing your algorithms, you will need to consider the following quantity:
   L^n_k = number of k-multisets of an n-set.

   (a) (3 points) Give a recurrence formula for L^n_k.
   (b) (2 points) Give a brief algorithm which computes and stores L^m_i, for all $i \leq k$ and $m \leq n$, on a table.
   (c) (25) Develop successor, ranking and unranking algorithms for the k-multisets of an n-set. You may simply refer to the table values calculated in the previous part.

   Hint: Try to adapt the algorithms for lexicographical ordering of k-subsets of an n-set. Note that you will use quantities L^m_i in place of quantities $\binom{p}{s}$.

   Example: $k = 3$, $n = 4$, $L^4_3 = 20$

rank	T	$\vec{T}$
0	$\{1,1,1\}$	$[1,1,1]$
1	$\{1,1,2\}$	$[1,1,2]$
2	$\{1,1,3\}$	$[1,1,3]$
3	$\{1,1,4\}$	$[1,1,4]$
4	$\{1,2,2\}$	$[1,2,2]$
5	$\{1,2,3\}$	$[1,2,3]$
6	$\{1,2,4\}$	$[1,2,4]$
7	$\{1,3,3\}$	$[1,3,3]$
8	$\{1,3,4\}$	$[1,3,4]$
9	$\{1,4,4\}$	$[1,4,4]$
10	$\{2,2,2\}$	$[2,2,2]$
11	$\{2,2,3\}$	$[2,2,3]$
12	$\{2,2,4\}$	$[2,2,4]$
13	$\{2,3,3\}$	$[2,3,3]$
14	$\{2,3,4\}$	$[2,3,4]$
15	$\{2,4,4\}$	$[2,4,4]$
16	$\{3,3,3\}$	$[3,3,3]$
17	$\{3,3,4\}$	$[3,3,4]$
18	$\{3,4,4\}$	$[3,4,4]$
19	$\{4,4,4\}$	$[4,4,4]$

Part 2: Backtracking Algorithm Design and Implementation

4. (30 points) **Steiner Triple Systems by backtracking**

   A Steiner Triple System of order n, STS(n), is pair $(X, \mathcal{B})$ where X is an n-set and $\mathcal{B}$ is a colection of $n(n-1)/6$ 3-subsets (triples) of X, and every pair of elements of X is contained in exactly one triple.

   Examples: of STS(7) and STS(9), respectively:
   $([1,7], \{\{1,2,3\},\{1,4,5\},\{1,6,7\},\{2,4,6\},\{2,5,7\},\{3,4,7\},\{3,5,6\}\}$
   $([1,9], \{\{1,2,3\},\{1,4,7\},\{1,5,9\},\{1,6,8\},\{2,4,9\},\{2,5,8\},\{2,6,7\},\{3,4,8\},\{3,5,7\},\{3,6,9\},\{4,5,6\},\{7,8,9\}\})$

   (a) Write a backtracking algorithm (pseudocode) to find all STS(n).

   (b) Implement your algorithm and use it to find the number of different STS(7). Can you also find the number of different STS(9) and STS(13)? For each run of your algorithm, specify n, number of nodes in the backtracking tree and number of STS(n) found. Also, please show some of the STS's generated for verification.

   It may be useful to use the algorithm for generating 3-sets in lexicographical order (rank, unrank, successor). This is implemented in C, and available from the textbook's web page:
   `http://www.math.mtu.edu/~kreher/cages/Src.html`