0.1 COMBINATORIAL ALGORITHMS

0.1.1 Who?

The subroutines are offered by Albert Nijenhuis and Herbert S. Wilf in their book: *Combinatorial Algorithms*, published in 1978 by Academic Press, Inc.

0.1.2 Where?

The authors were teaching in Department of Mathematics at University of Pennsylvania when writing the book.

0.1.3 When?

Since the book I got is the second edition, the subroutines must be created some time before 1975 when the first edition was issued.

0.1.4 Why?

In this directory is a collection of subroutines, in FORTRAN 77, for the solution of combinatorial problems. It could be used as not only a very helpful tool to teach the algorithms of combinatorics but also a beginning set of building blocks for users to incorporate into their programs to meet their own needs. Therefore, all the subroutines are not just a collection of pretty airfacts to be looked at but basic elements of the growing and working equipment of scientific investigation and learning.

0.1.5 What?

The subroutines contained in this directory are:

backtr.f Supervise the backtrace search.

chromp.f Calculate the chromatic polynomial of a connected graph.

colvrt.f Find possible colors of vertex K.

cycles.f Count cycles, find sign of a permutation, tag and/or invert.

eulcrc.f Find candidates for Kth edge of Euler circuit.

exheap.f Sort a list of any items into linear order.

hamcrc.f Find candidates for Kth vertex in a Hamilton circuit.

hpsort.f Sort a linear array into nondecreasing order.

invert.f Invert upper triangular matrix.

lbltre.f Produce the edge list of a tree from its Priifer codeword.

lexsub.f Generate subsets of {1, 2, ..., N} which succeed input set, in lexicographical order, with optional jumps over supersets.

minspt.f Find spanning tree of minimal length.

mobius.f Find Mobius matrix from covering relation.

netflo.f Find maximum flow in a network.

nexcom.f Next composition of n into k parts.

nexequ.f Generate next equivalence relation on {1, 2, .., n}.

nexksb.f Next k-subset of a n-set, in lexicographic order.

nexpar.f Find next partition of n.

nexper.f Generate next permutation of 1, 2, ..., n.

nexsub.f Generate subsets of {1, 2, ..., N}, in an order specified by the Gray code.

nexytb.f Supplies the sequence of Young tableaux of given shape.

nxksrd.f List k-subsets of an n-set, in RD order.

perman.f Calculate permanent of square matrix.

poly.f Operations on polynomials in power and factorial form.

powser.f Compose power series.

rancom.f Random composition of n into k parts.

ranequ.f Generate random equivalence relation on {1, 2, ..., n}.

ranksb.f Choose a random k-subset of {1, 2, ..., n}.

ranpar.f Generate a random partition of n.

ranper.f Generate random permutation of n letters.

ranrut.f Generate random unlabeled rooted tree.

ransub.f Generate random subset of an n-set.

ranytb.f Selects, u.a.r., a Young tableau of given shape.

renumb.f Renumber rows and coloumns of a matrix.

select.f Implement the algorithms for performing any of the four tasks (sequencing, ranking, unranking, and random selection) on any of the following families:

1. K-subsets of an N-set
2. partitions of N objects into K classes
3. permutations of N objects with K cycles
4. vector subspaces of dimension K of N-dimensional space over GF(Q)[1]
5. permutations of N letters with K runs
6. partitions of N whose largest part is K
7. compositions of N into K parts

spanfo.f Determine connectivity of a graph; find spanning forest.

spntre.f Find candidates for Kth edge of spanning tree.

triangle.f Discover consistent labeling of elements of partially ordered set.

0.1.6 How?

In last section, a list of combinatorial objects have been shown. Two kinds of usages, the exhaustive search and the random sampling, are dealt with. The two categories of use acall for different kinds of algorithms for different problems. Users could check Nijenhuis and Wilf's book to get more information on the mathematical basis of the problems and the algorithms, and even flow charts. The specifications of subroutines are also available.

We haven't tested all the subroutines by writing drivers for them. Users may have to write their own main programs to try them.

The subroutines that have drivers already are hpsort.f, nexytb.f, nexper.f, nexpar.f, and invert.f.

0.1.7 Programming Language?

The source code was written in FORTRAN 77.

We use Fortran 77 compiler to make it.

[1] Q is set to 2 by a DATA statement in the FUNCTION PH1 subprogram. It can easily be changed by user, if desired. Please refer to the source code for more detail.

0.1.8 Operating Environment?

The operating system we use is sparc-sun-sunos4.1.3.

0.1.9 Efficiency?

So far with the subroutines for which we have made drivers, the efficiency is pretty satisfactory. The reason might be that all the calculation needed is not very time consuming.

0.1.10 Portability?

To make the software run, users have to write main program in which the subroutines are called, and may also need to create some other routines to interact with them. For example, in backtr.f, another routine CANDTE might be needed to find the list of candidates and place them at the end of a STACK which is a parameter of backtr().

After the driver is set, just compile it with fortran 77 compiler and offer some test data to run it.

0.1.11 Availability?

Since all the subroutines are extracted from Nijenhuis and Wilf's book, there is no mention about the license issue in the software. Neither it could found in the book.

0.1.12 Documentation?

No document is available for the library of the subroutines. Nijenhuis and Wilf's book is the best reference to understand and use the software.

0.1.13 Robustness?

We just tested a small part of the subroutines and did not find problem with them except that there is a type error which might happen when someone was entering the subroutine. And the Fortran 77 compiler seems to be powerful to avoid any run time crash.

0.1.14 Data Formats?

The data formats are different depending on the problems and algorithms. The details are in Nijenhuis and Wilf's book.

0.1.15 Output and Visualization?

The software doesn't offer any output format. Users could define it in their driver programs. Besides, in Nijenhuis and Wilf's book, there are some hints on which data could be output and input, which is input only or output only. Anyway, users still have flexibility to decide themselves.

0.1.16 Ease of Interface

It is not difficult at all to use the software in case you know how to write some pieces in Fortran 77. All the subroutines are created in order to be incorporated into other applications, therefore they have good interface – explicit declaration of parameters.

0.1.17 Usefulness?

9 – very comprehensive and useful to learn combinatorial algorithms.