The *Combinatorica* Software Package

by Bruce E. Sagan
Department of Mathematics
Michigan State University
East Lansing, MI 48824-1027

Combinatorica, ver. 0.8, 1990; Steven S. Skiena, Department of Computer Science, State University of New York, Stony Brook, NY 11794, (516) 632-9026, skiena@sbcs.sunysb.edu. Requires a computer running version 1.2 or later of *Mathematica*, included with version 2.0 or later; no charge for copies.

Implementing Discrete Mathematics: Combinatorics and Graph Theory with Mathematica$^{\mathbf{r}}$, Addison-Wesley, Reading, MA, 1990; single copy \$43.95.

Combinatorica is a software package for doing combinatorics and graph theory. It consists of over 230 functions for manipulating permutations, combinations, partitions, Young tableaux, graphs, and various other objects. About three-quarters of these functions deal with graphs and graph algorithms, the remaining quarter are devoted to other topics. The contents of this package make it quite suitable for a class in combinatorics and a good companion to the texts of Roberts [4], Stanton and White [5], or Tucker [6]

I. Combinatorica and Mathematica. *Combinatorica* can be run only as part of a *Mathematica* session since it is written using primitives from that algebra package. I will assume the reader is familiar with *Mathematica*, its programming techniques and user interface. More information on *Mathematica* can be found in the reviews by Hoenig [2] and Herman [1].

If your copy of *Mathematica* is version 2.0 or later, then *Combinatorica* can be found in the **Packages/DiscreteMath** directory. If your version is at least 1.2 but does not contain the package, it is available free by anonymous ftp to cs.sunysb.edu (or 130.245.1.15). The files are in the directory pub/Combinatorica and take up only about 100K worth of disk space. Once the package is on your system, it can be loaded using the line

```
In[1]:= <<Combinatorica.m
```

Since *Combinatorica* is based on *Mathematica*, it derives many of its strengths and weaknesses from its parent. All of its functions are logically named using full English words instead of unreadable abbreviations. Thus, if you want a listing of all the subsets of {1,2,3} listed in lexicographic order, merely issue the command

```
In[2]:= LexicographicSubsets[{1,2,3}]
```

and you will be rewarded with

```
Out[2]= {{}, {1}, {1, 2}, {1, 2, 3}, {1, 3}, {2}, {2, 3}, {3}}
```

However, if you want to do more than just call built-in functions, you must be comfortable with writing programs in *Mathematica*. This is a functional programming language akin to Lisp, as opposed to an imperative language such as Fortran, Pascal or C. Although Skiena provides an appendix in his manual with an overview of the necessary background, I found it too brief to understand the functions he had constructed or to make my own without consulting a book on *Mathematica* itself such as [7]. However, once I was familiar with the material, the appendix was an excellent reference for jogging the memory.

One of the outstanding things about *Mathematica* is its graphics capabilities, and *Combinatorica* makes full use of them. *Combinatorica* can make pictures of Ferrers diagrams for integer partitions, Hasse diagrams for partial orders, and various embeddings of graphs. Furthermore, these images can be printed using the *Mathematica* **PSPrint** command. All the figures in this review were generated that way.

II. Combinatorial Functions. *Combinatorica* knows about permutations, permutation groups, certain special classes of permutations (involutions, derangements, Josephus permutations, heaps), combinations, partitions of sets and integers, compositions (ordered integer partitions), and Young tableaux. For each object, there is a corresponding enumeration function so one can evaluate expressions involving multinomial coefficients, Stirling numbers, hooklengths, cycle index polynomials, and the partition function. There are also functions for generating all objects of a certain type, ranking and unranking them, and picking an object at random. Many of these constructions are taken from the book of Nijenhuis and Wilf [3] where they are presented as Fortran programs.

Some of the *Combinatorica* functions were written with an eye towards illustrating certain programming techniques, rather than making them as efficient as possible. For example, **DistinctPermutations**, which lists all the different permutations of a set with repeated elements, makes use of a generalized backtrack procedure. It can be made 7–10 times faster by modifying the algorithm in Roberts' book [4, page 66]. However, this bias can be turned into an educational advantage by challenging the students to find functions which outperform those in the package.

Classroom demonstrations that illuminate combinatorial concepts are easy to construct using *Combinatorica*. For example, many of my students have difficulty understanding big-Oh notation and its connection with the analysis of algorithms. If I ask them for the computational complexity of generating all 2-element subsets of an n-set, I'm liable to get blank stares. A better approach is to time the *Combinatorica* function **KSubsets** which lists all such combinations. For added effect, we can make a plot of these times for $n = 10, 20, \ldots, 100$ by calling

```
In[3]:= ListPlot[ Table[n,Timing[KSubsets[Range[n],2];][[1,1]],
                  {n,10,100,10}] ];
```

which yields the graph in Figure 1.

FIGURE 1 ABOUT HERE

Now when I ask what function would approximate the data points, I get a response that easily motivates the equation

$$n(n-1)/2 = \mathrm{O}(n^2).$$

III. Graph Functions. *Combinatorica* can utilize all of the standard graphs: complete graphs, bipartite graphs, trees, cycles, stars, wheels, grids, hypercubes and the like. It has functions for performing various operations to form new graphs such as taking unions, joins, products, forming the line graph, or realizing a degree sequence. The graphics capabilities of *Mathematica* are put to good use here, and these graphs can be displayed in a number of different embeddings in the plane. However, the unwary user will be in for some surprises. For example, the complete tripartite graph $K_{2,2,2}$ is produced with the command

```
In[4]:= ShowLabeledGraph[ K[2,2,2] ];
```

whose output is shown in Figure 2. You will note that the edges {1,5} and {2,6} have been obscured.

FIGURE 2 ABOUT HERE

One can rectify this situation by "shaking" the graph and calling

```
In[5]:= ShowLabeledGraph[ ShakeGraph[ K[2,2,2], 0.2] ];
```

which perturbs the vertices at random to eliminate collinearities (Figure 3).

FIGURE 3 ABOUT HERE

It is a pity that the shaken version is not the default.

Skiena provides a number of procedures for extracting graphical invariants. A single function call will find the components of a graph, any Eulerian or Hamiltonian cycles, maximum cliques or minimum covers, the automorphism group, or the chromatic polynomial. The usual graph algorithms covered in a course on the subject are also implemented, including those for shortest paths, minimum spanning trees, network flows, matchings and planarity. These procedures are fairly complicated, and students often have trouble understanding them from a description in a text. Using the *Mathematica* **Trace** function to keep track of the values of various variables at different stages of the algorithm often makes it much clearer what is going on.

IV. The Manual. Skiena's book presents each *Combinatorica* function by providing both the *Mathematica* code and an English language description. It also gives various examples of how procedures can be used, offering a wealth of material to students. The Glossary of Functions is a handy reference as are the many citations of the literature. Exercises and research problems are given at the end of each chapter. As an unexpected bonus, Skiena has a nice sense of humor and has no qualms about letting it show.

The text itself is written with good attention to the issue of computational complexity. Skiena also tries to give simple, intuitive descriptions of the algorithms used. Unfortunately, this leads to numerous small inaccuracies and a few larger errors in the book. Thus I found it was hard to understand a function from his description unless I already knew how it was supposed to work.

V. Summary. *Combinatorica* won a Distinguished Mathematics Software Award in the 1991 EDUCOM Competition. It provides functions for working with most of the topics that are covered in a standard combinatorics and graph theory course. Students will find it hard to do anything substantial with *Combinatorica* unless they are quite familiar with programming in *Mathematica*. However, the instructor can use the package, especially its graphics features, to put together demonstrations that will illuminate nicely some of the more difficult concepts in combinatorics and the theory of algorithms.

References

[1] Eugene A. Herman, *Mathematica* — A Review, *Notices of the American Mathematical Society* 35 (1988) 1333–1349.

[2] Alan Hoenig, Mathematics by machine with *Mathematica*, Software Reviews, *College Mathematics Journal* 21 (1990) 146–149.

[3] Albert Nijenhuis and Herbert S. Wilf, *Combinatorial Algorithms*, 2nd ed., Academic Press, New York, NY, 1978.

[4] Fred S. Roberts, *Applied Combinatorics*, Prentice-Hall, Englewood Cliffs, NJ, 1984.

[5] Dennis Stanton and Dennis White, *Constructive Combinatorics*, Springer-Verlag, New York, NY, 1986.

[6] Alan Tucker, *Applied Combinatorics*, 2nd ed., John Wiley and Sons, New York, NY, 1984

[7] Stephen Wolfram, *Mathematica*, Addison-Wesley, Redwood City, CA, 1988.