2

Generating Elementary Combinatorial Objects

2.1 Combinatorial generation

Often it is necessary to find nice algorithms to solve problems such as generating all the subsets of a given set $\mathcal{S}$ of size n, say. Related problems include generating all the permutations of $\mathcal{S}$, or all the k-subsets of $\mathcal{S}$.

Among the generation algorithms we will study are those that generate the desired objects in a *lexicographic order*, and the so-called *minimal change algorithms*, in which each object is generated from the previous one by performing a very small change. Both of these types of *sequential generation* are often accomplished by means of a *successor* algorithm, which is used to find the next object following a given one, with respect to a given ordering.

As well as sequential generation, we will be interested in *ranking* and *unranking* algorithms. A ranking algorithm determines the position (or rank) of a combinatorial object among all the objects (with respect to a given order); an unranking algorithm finds the object having a specified rank. Thus, ranking and unranking can be considered as inverse operations.

Here are slightly more formal mathematical descriptions of these concepts. Suppose that $\mathcal{S}$ is a finite set and $N = |\mathcal{S}|$. A ranking function will be a bijection

$$\mathsf{rank} : \mathcal{S} \rightarrow \{0, \ldots, N-1\}.$$

A rank function defines a total ordering on the elements of $\mathcal{S}$, by the obvious rule

$$s < t \Leftrightarrow \mathsf{rank}(s) < \mathsf{rank}(t).$$

Conversely, there is a unique rank function associated with any total ordering defined on $\mathcal{S}$.

If rank is a ranking function defined on $\mathcal{S}$, then there is a unique unranking function associated with the function rank. This function unrank is also a bijection,

$$\mathsf{unrank} : \{0, \ldots, N-1\} \rightarrow \mathcal{S}.$$