Manuel Kauers
RISC-Linz

Hagenberg, March 12, 2004

Algorithmic Combinatorics

Homework 2

Exercise 1 Find a rank function for $[n]^{[n-2]}$, i.e., find a bijection $r\colon [n]^{[n-2]} \to [n^{n-2}]$ such that r^{-1} is nice to compute.

We do it for $[n]^{[m]}$, $n, m \in \mathbb{N}$. For given $\pi = (\pi_1, \dots, \pi_m) \in [n]^{[m]}$ define

$$r(\pi) = 1 + \sum_{i=1}^{m} (\pi_i - 1) n^{i-1}.$$

Then, given $N \in [n^m]$ we have

$$r^{-1}(N) = (\pi_1, \dots, \pi_m)$$

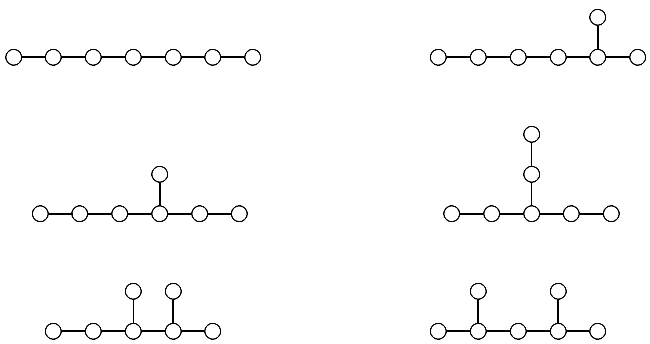
where $\pi_i = ((N-1) \bmod n^i \operatorname{div} n^{i-1})$ $(i = 1, \dots, m)$.

Exercise 2 Suppose we are given a random number generator. Write a Mathematica program that constructs labeled trees over $[n]$ uniformly at random.

We proceed by choosing a random number $r \in [n^{n-2}]$, convert it into a tuple $t \in [n]^{[n-2]}$, to which in turn we apply the the inverse Prüfer transform.
The code for a Mathematica implementation is provided in the appendix to this solution. A Mathematica readable version of it is provided at http://www.risc.uni-linz.ac.at/people/mkauers/exercises/

Exercise 3 How many different unlabeled trees over 7 vertices are there?

We have found the following ones:



Exercise 4 Try to give a constructive definition of the notion "unlabeled graph."

We represent an unlabeled graph by the equivalence class of all isomorphic labeled graphs, where the isomorphy of unlabelled graphs is used.

If the equivalence relation $\sim$ on $\mathrm{LabTrees}(n)$ is defined by

$$T_1 \sim T_2 \;:\Longleftrightarrow\; \exists\, f\colon T_1 \to T_2\ \forall\, v \in V(T_1) : \mathrm{Neighbours}(f(v)) = f(\mathrm{Neighbours}(v))$$

then we may define $\mathrm{UnlabTrees}(n) := \mathrm{LabTrees}(n)/\sim$. This definition is constructive as there are only finitely many candidates $f\colon T_1 \to T_2$ for an isomorphism.

Exercise 5 Prove the recurrence of the Stirling numbers of the second kind:

$$S(n,k) = kS(n-1,k) + S(n-1,k-1) \qquad (n,k \geq 1).$$

$S(n,k)$ is by definition the number of partitions of a set of n elements into k disjoint nonempty subset. For a recurrence of $S(n,k)$, consider a systematic way to enumerate all those partitions. Given $\{1,\ldots,n\}$ we may first compute the partitions of $\{1,\ldots,n-1\}$ into k subsets and add $\{n\}$ to one of those. There are $S(n-1,k)$ partitions of $\{1,\ldots,n-1\}$ and k possible extensions to partitions of $\{1,\ldots,n\}$. There are further partitions where $\{n\}$ is a singleton. These are not included yet because empty subsets are not permitted. There are $S(n-1,k-1)$ partitions $S_1 \cup \cdots \cup S_{k-1}$ of $\{1,\ldots,n-1\}$ and they extend to as many partitions $S_1 \cup \cdots \cup S_{k-1} \cup \{n\}$ of $\{1,\ldots,n\}$. We have found the recurrence

$$S(n,k) = kS(n-1,k) + S(n-1,k-1) \qquad (n \geq 1, k \geq 1)$$

Exercise 6 Prove:

$$x^n = \sum_{k=0}^{n} S(n,k) x^{\underline{k}}$$

For $n = 0$ there is nothing to prove. Suppose the formula holds for some $n \geq 0$. Then

$$\begin{aligned}
&\sum_{k=0}^{n+1} S(n+1,k) x^{\underline{k}} = \sum_{k=0}^{n+1} kS(n,k) x^{\underline{k}} + \sum_{k=0}^{n+1} S(n,k-1) x^{\underline{k}} \\
&= \sum_{k=0}^{n} S(n,k) x^{\underline{k}} + \sum_{k=-1}^{n} S(n,k) x^{\underline{k+1}} = \sum_{k=0}^{n} S(n,k) x^{\underline{k}} + \sum_{k=0}^{n} S(n,k) x^{\underline{k}} (x-k) \\
&= \sum_{k=0}^{n} S(n,k) x^{\underline{k}} + x \sum_{k=0}^{n} S(n,k) x^{\underline{k}} - \sum_{k=0}^{n} S(n,k) x^{\underline{k}} = x^{n+1},
\end{aligned}$$

Exercise 7 Prove:

$$S(n,k) = \frac{1}{k!} \sum_{j=0}^{k} (-1)^{k-j} \binom{k}{j} j^n$$

Note that the exponential generating function is

$$f_k(x) = \sum_{n=0}^{\infty} \frac{S(n,k)}{k!} x^n = \frac{(\mathrm{e}^x - 1)^k}{k!}.$$

Using this we have

$$S(n,k) = n! \langle x^n \rangle \frac{(\mathrm{e}^x - 1)^k}{k!} = \frac{1}{k!} \frac{\partial^n}{\partial x^n} (\mathrm{e}^x - 1)^k \Big|_{x=0}$$

$$\frac{\partial^n}{\partial x^n} (\mathrm{e}^x - 1)^k = \frac{\partial^n}{\partial x^n} \sum_{i=0}^{k} \binom{k}{i} \mathrm{e}^{ix} (-1)^{k-i} = \sum_{i=0}^{k} \binom{k}{i} i^n \mathrm{e}^{ix} (-1)^{k-i}$$

and therefore

$$S(n,k) = \frac{1}{k!}\sum_{i=0}^{k}\binom{k}{i} i^n (-1)^{k-i}.$$

Exercise 8 Let $B(n) := \sum_{k=0}^{n} S(n,k)$. Prove: $B(n+1) = \sum_{k=0}^{n} \binom{n}{k} B(k)$ for $n \geq 0$.

We assume that the exponential generating function $b(x) = \exp(\mathrm{e}^x + 1)$ is known. (See Exercise 24 of the last semester.) Then

$$\begin{aligned}
\sum_{n=0}^{\infty} \frac{B(n+1)}{n!} x^n &= \frac{\partial}{\partial x} b(x) \\
\sum_{n=0}^{\infty} \frac{1}{n!} \sum_{k=0}^{n} \binom{n}{k} B(k) x^n &= \sum_{n=0}^{\infty} \sum_{k=0}^{n} \frac{B(k)}{k!} \frac{1}{(n-k)!} x^n \\
&= \sum_{n=0}^{\infty} \frac{B(n)}{n!} x^n \cdot \sum_{n=0}^{\infty} \frac{1}{n!} x^n \\
&= b(x)\mathrm{e}^x.
\end{aligned}$$

Thus it has to be shown that

$$b'(x) = b(x)\mathrm{e}^x$$

which is indeed evident.

Appendix

```
(**
 * RandomTree[n]
 *
 * Generates a labelled tree with n vertices at random.
 * Every tree has the same (pseudo)probability of being
 * the result of this method.
 **)
RandomTree[n_Integer] :=
  Random[Integer, {1, n^(n-2)}] // (* create a random number *)
  Number2Tuple[n, n-2, #]& // (* convert it to a tuple *)
  UnPruefer[n, #]& // (* convert the tuple to a tree *)
  Tree2Expr; (* express the tree as an expression tree *)

(**
 * Number2Tuple[n, m, t]
 *
 * Maps the number t to the element of [n]^[m] it encodes.
 * t is assumed to be an integer between 1 and n^m inclusively.
 **)
Number2Tuple[n_Integer, m_Integer, t_Integer] :=
  Module[ {i},

  Table[1 + Quotient[Mod[t-1, m^i], m^(i-1)], {i, 1, m}]
];
```

```
(**
 * UnPruefer[n, pi]
 *
 * Given a tuple pi in [n]^[n-2], this method generates the tree
 * for which Pruefer gives pi.
 *
 * The tree is returned as a set of pairs encoding the edges of the
 * tree. The vertex set is always [n].
 **)
UnPruefer[n_Integer, pi_List] :=
  Module[ {edges, e, i, N, found},

  N = Range[1, n-1];
  edges = {{Last[pi], n}};
  found = {};

  Do[
    e = Min @@ Complement[N, found, Drop[pi, i-1]];
    AppendTo[found, e];
    AppendTo[edges, Sort[{pi[[i]], e}]];
  , {i, 1, n-2}];

  Sort[edges]
];

(**
 * Tree2Expr[e]
 *
 * This is a pretty printer of for trees which converts a list
 * of edges into an expression tree.
 * The nodes will become Heads of the resulting expression.
 **)
Tree2Expr[e_List] :=
  Module[ {i, a, t, m},

  If[ e === {}, Return[] ];

  m = Max[Flatten[e]];

  (* we implicitly assume that an edge {a,b} is directed from
     min(a,b) to max(a,b) *)
  1 @@ Table[
    Tree2Expr[Cases[e, {a_, b_} /; a>=i && b>=i]
       /. a_Integer -> a-i+1]
    /. {a_Integer -> a+i-1, Null -> i},
    {i, 2, m}][[Cases[Sort /@ e, {1, a_} -> a-1]]]
];
```