A BIJECTION FROM ORDERED TREES TO BINARY TREES THAT SENDS THE PRUNING ORDER TO THE STRAHLER NUMBER

Doron Zeilberger *

Appeared in: Discrete Math **82** (1990), 89-92.

In accordance with the principle from other branches of mathematics that it is better to exhibit an explicit isomorphism between two objects than merely to prove that they are isomorphic, we adopt the general principle that it is better to exhibit an explicit one-to-one correspondence (bijection) between two sets than merely to prove that they have the same number of elements. (Richard Stanley [S], p.11)

It is well known (e.g. [S-W], p. 60) that the number of ordered trees with n vertices equals the number of complete binary trees with n leaves. Mireille Vauchaussade de Chaumont and Ge'rard Viennot[V-V1][Va] (see also [S-W], ch.3, ex.6 (p. 103)) discovered an interesting refinement of this fact. They proved that for any integers n and k, the number of ordered trees with n vertices and pruning order k equals the number of complete binary trees with n leaves and Strahler number k. In this communication I construct a bijection whose "shadow" is this result, thus giving a "bijective proof" of the Vauchaussade-Viennot result and thereby solving their ten-bottles-of-wine problem([V-V2]). This problem was also solved, independently, by Bender and Canfield [B-C].

First, definitions! It will be convenient to adhere to Schutzenberger's philosophy of viewing combinatorial objects such as trees as words in an appropriate formal language. Let T and B stand for "an ordered tree" and "a complete binary tree" respectively. An ordered tree is a finite word in the 2-letter alphabet (,) defined recursively by:

$$T = () or T = (T_1 T_2 ... T_k) for some k \geq 1,$$

where the T_i are themselves ordered trees. The *number of vertices*, ver[T], is defined recursively by $ver[()] := 1$, and

$$ver[(T_1 ... T_k)] := ver[T_1] + ... + ver[T_k] + 1.$$

A *forest* of ordered trees is a concatenation $T_1 T_2 ... T_k$ of ordered trees. Thus every ordered tree can be written as (F), where F is a forest.

A *complete binary tree* is a word in the alphabet (,) defined recursively by:

$$B = () or B = (B_1 B_2),$$

where B_1 and B_2 are complete binary trees on their own right. The *number of leaves* of B, leaf[B], may be defined recursively by $leaf[()] = 1$ and $leaf[(B_1 B_2)] = leaf[B_1] + leaf[B_2]$.

The *Strahler number*, s[B], of a complete binary tree B is defined recursively as follows:

* Department of Mathematics and Computer Science, Drexel University, Philadelphia, PA19104.
Supported in part by the NSF.

$$s[()] = 0, s[(B_1B_2)] = \{ \begin{matrix} max[s[B_1], s[B_2]], s[B_1] \neq s[B_2] \\ s[B_1]+1, s[B_1] = s[B_2] \end{matrix} .$$

As usual, a *factor* of a word $w = w_1...w_n$ is any of the words $w_iw_{i+1}...w_{j-1}w_j$, for $1 \leq i \leq j \leq n$. A *hanging branch* of an ordered tree is a proper factor of the form $(^r)^r$ for some $r \geq 1$ [i.e. r consecutive occurrences of the letter "(" followed by r consecutive occurrences of the letter ")"]. A hanging branch is a *maximal hanging branch* if it is not a proper factor of another hanging branch.

A pruning of an ordered tree is the act of deleting all its maximal hanging branches. The *pruning order* of an ordered tree is the number of prunings required to reduce it to (). For example after one pruning $(((()(()()))(()()(())))$ becomes $((())())$, and after two prunings it shrinks to the one-vertex tree (), so its pruning order is 2.

A k-skewer is an ordered tree of pruning order k that after $k-1$ prunings becomes a tree of the form $(^r)^r$, $(r \geq 2)$. Note that any ordered tree of pruning order k becomes a tree of pruning order 1 after $k-1$ prunings, i.e. a tree of the form $(H_1...H_m)$, where $H_1, ..., H_m$ are maximal hanging branches. By looking at the stuff that disappeared *before* the last pruning, it is easily seen that every ordered tree T of order k has the following form, for some $m \geq 1$

$$T = (F_1S_1F_2...F_mS_mF_{m+1}),$$

where F_i $(i = 1, ..., m)$ are possibly empty *forests* of trees such that (F_i) have order $\leq k-1$ and the S_i are trees such that (S_i) are $k-skewers$. We say that *T has m k-skewers.*

We are now ready to describe the algorithm.

Algorithm V

Input: An ordered tree T.

Output: A complete binary tree $V[T]$ whose Strahler number equals the pruning order of T and whose number of leaves equals the number of vertices of T.

1. (initialize) V[()]=() .

2. If $T \neq ()$ write $T = (T_1T_2...T_a)$. Let $T^{(1)} = T_1$ and $T^{(2)} = (T_2...T_a)$.

$(ver[T] = ver[T^{(1)}] + ver[T^{(2)}].)$

3. Let $U^{(1)}$ and $U^{(2)}$ be certain trees obtained *below* out of $T^{(1)}$ and $T^{(2)}$ then

$$V[T] := (V[U^{(1)}]V[U^{(2)}]).QED$$

Procedure*below* :

k :=the pruning order of T.

Case 1a: $order[T^{(1)}] \leq k-1$, $order[T^{(2)}] = k$, and

Case 1b: $order[T^{(1)}] = k-1$, $order[T^{(2)}] = k-1$.

$U^{(1)} := T^{(1)}$ and $U^{(2)} := T^{(2)}$.

Case 2: $order[T^{(1)}] = k-1$ and $order[T^{(2)}] \leq k-2$. [In both cases 1b and 2 $T^{(1)}$ must have at least two $(k-1)$-skewers or else T would be of order $\leq k-1$.]

Write $T^{(1)}$ in the form

$$T^{(1)} = (F_1 S_1 F_2 ... F_m S_m F_{m+1}), m \geq 2,$$

where (S_i) are $(k-1)$-skewers and F_i are possibly empty forests of trees such that (F_i) have order $\leq k-2$. Write $T^{(2)} = (F)$, where F is the forest resulting from $T^{(2)}$ by deleting its first and last letter.

$$U^{(1)} := (F_1 S_1 F_2), U^{(2)} := (F S_2 F_3 ... F_m S_m F_{m+1}).$$

Case 3: $order[T^{(1)}] = k$.

Now $T^{(1)}$ has exactly one $k-skewer$ [or else T would be of order $> k$], and thus has the form: $(F_1 S_1 F_2)$, where F_1 and F_2 are (possibly empty) forests such that (F_1) and (F_2) are trees of order $\leq k-1$ and (S_1) is a $k-skewer$. As before write $T^{(2)} = (F)$.

$$U^{(1)} := (F_1 S_1 F), U^{(2)} := (F_2).QED$$

In procedure *below* all the cases give mutually exclusive outputs. Indeed we have the following outcomes.

For Case 1a: $order[U^{(1)}] \leq k-1$, $order[U^{(2)}] = k$.

For case 1b: $order[U^{(1)}] = k-1$, $order[U^{(2)}] = k-1$, and $U^{(1)}$ has at least two skewers.

For case 2: $order[U^{(1)}] = k-1$, $order[U^{(2)}] = k-1$, *and* $U^{(1)}$ has exactly one skewer.

For case 3: $order[U^{(1)}] = k$, $order[U^{(2)}] \leq k-1$.

Not only do the four cases give mutually disjoint outputs, they also exhaust all the conceivable possibilities. It thus follows by induction that the mapping is both one-to-one and onto.

REFERENCES

[B-C] E. Bender and E. R. Canfield, *private communication.*

[S] R. Stanley, *"Enumerative combinatorics"*, vol.1, Wadsworth, Monterey, 1986.

[S-W] D. Stanton and D. White, *"Constructive Combinatorics"*, Springer, New York, 1986.

[V-V1] M. Vauchaussade de Chaumont and G. Viennot, *Enumeration of RNA's secondary structure by complexity*, in " Mathematics in biology and medicine" (proceedings, Bari, 1983), Lecture Notes in Biomathematics #57, (V.Capasso et al., editors), Springer Verlag, 1985.

[V-V2] M. Vauchaussade de Chaumont and G. Viennot , in *"Séance de problemes"* (p.386) of *"Combinatoire énumérative"*(proceedings, Montreal 1985), G.Labelle and P.Leroux, editors, Lecture Notes in Mathematics #1234, Springer Verlag, 1986.

[Va] M. Vauchaussade de Chaumont, *"Nombre de Strahler des arbres, languages algébrique et dénombrement de structures secondaires en Biologie moléculaire"*, Thesis, Bordeaux I, 1985.