The combstruct Package - New Features

This worksheet is an introduction to the new features of the combstruct package. For those not already familiar with this package, there are two worksheets that give an overview of the previous version of combstruct and explain how to use it.

Several new features have been added to the combstruct package. Powersets, i.e. sets without repetition, have been implemented. The *allstructs* function, which generates all the elements of a given size in a combinatorial class, has been extended to accept grammar definitions, whereas previously it only worked for the pre-defined structures. Three new functions, *gfeqns*, *gfsolve* and *gfseries*, have been introduced to find the counting generating functions of the combinatorial classes defined by grammars.

```
> with(combstruct):
```

PowerSet

The new constructor PowerSet has been added to the description language. A PowerSet is a set without repetition, as opposed to the Set constructor which allows repetition of elements.

Here we use PowerSet to define integer partitions with distinct parts.

```
> sys := {L = PowerSet(Sequence(Z,card>=1)) },unlabelled;
```

$$sys := \{L = \mathbf{PowerSet}(\mathbf{Sequence}(Z, 1 \leq card))\}, unlabelled$$

```
> draw([L,sys],size=5);
```

$$\mathbf{PowerSet}(\mathbf{Sequence}(Z), \mathbf{Sequence}(Z, Z, Z, Z))$$

```
> seq(count([L,sys],size=i),i=1..10);
```

$$1, 1, 2, 2, 3, 4, 5, 6, 8, 10$$

allstructs

The allstructs command now accepts grammars as input. For example, we can generate all the alcohol molecules, $C_n H_{2n} OH$, which have n carbon atoms. We model a molecule by writing a grammar for an alkyl,
$C_n H_{2n+1}$, which is a ternary rooted tree that moves freely in space.

```
molecule := {alkyl = Union(H, Prod(C, Set(alkyl, card=3))),
H=Atom, C=Atom};
```

$$molecule :=$$
$$\{alkyl = \mathbf{Union}(H, \mathbf{Prod}(C, \mathbf{Set}(alkyl, card = 3))), H = Atom, C = Atom\}$$

We find all the alkyls with 4 carbon molecules.

```
> allstructs([alkyl, molecule], size=4+2*4+1);
```

$$[\mathbf{Prod}(C, \mathbf{Set}(\mathbf{Prod}(C, \mathbf{Set}(\mathbf{Prod}(C, \mathbf{Set}(H, H, H)), H, H)),$$
$$\mathbf{Prod}(C, \mathbf{Set}(H, H, H)), H)), \mathbf{Prod}(C, \mathbf{Set}($$
$$\mathbf{Prod}(C, \mathbf{Set}(\mathbf{Prod}(C, \mathbf{Set}(\mathbf{Prod}(C, \mathbf{Set}(H, H, H)), H, H)), H, H)), H, H))$$
$$, \mathbf{Prod}(C, \mathbf{Set}($$

$$\mathrm{Prod}(C, \mathrm{Set}(\mathrm{Prod}(C, \mathrm{Set}(H, H, H)), \mathrm{Prod}(C, \mathrm{Set}(H, H, H)), H)), H, H)), \mathrm{Prod}(C, \mathrm{Set}(\mathrm{Prod}(C, \mathrm{Set}(H, H, H)), \mathrm{Prod}(C, \mathrm{Set}(H, H, H)), \mathrm{Prod}(C, \mathrm{Set}(H, H, H)))))]$$

Generating functions

Three functions are now available to work with the generating functions that count the number of objects described by the grammar. Given a variable z, each non-terminal A in the grammar will be associated with the generating function $\mathrm{A}(z)$,

$$\mathrm{A}(z) = \sum_{n=0}^{\infty} a_n z^n,$$

where a_n is the number of objects in A of size n.

Consider the grammar which describes necklaces with three different colours of beads.

```
> necklace := {N=Cycle(bead),
  bead=Union(red,blue,green),red=Atom,blue=Atom,green=Atom};
```

$$necklace := \{blue = Atom, green = Atom, N = \mathrm{Cycle}(bead), bead = \mathrm{Union}(red, blue, green), red = Atom\}$$

The function *gfeqns* returns the system of generating function equations for this grammar. Note that this function does a direct translation from the grammar productions to the generating function equations, without trying to resolve them.

```
> gfeqns(necklace,unlabelled,z);
```

$$\left[\mathrm{bead}(z) = \mathrm{red}(z) + \mathrm{blue}(z) + \mathrm{green}(z), \mathrm{green}(z) = z, \mathrm{red}(z) = z, \mathrm{blue}(z) = z, \mathrm{N}(z) = \sum_{j_1=1}^{\infty} \frac{numtheory_{\phi}(j_1) \ln\left(\frac{1}{1 - \mathrm{bead}(z^{j_1})}\right)}{j_1}\right]$$

The function *gfsolve* will try to solve for the explicit solutions.

```
> gfsolve(necklace,unlabelled,z);
```

$$\left\{\mathrm{bead}(z) = 3z, \mathrm{N}(z) = \sum_{j_1=1}^{\infty} \frac{numtheory_{\phi}(j_1) \ln\left(-\frac{1}{-1 + 3z^{j_1}}\right)}{j_1}, \mathrm{green}(z) = z, \mathrm{red}(z) = z, \mathrm{blue}(z) = z\right\}$$

The truncated series for each generating function can also be found, using the function *gfseries*
.

```
> gfseries(necklace,unlabelled,z);
```

table([

$$N(z) = 3z + 6z^2 + 11z^3 + 24z^4 + 51z^5 + O(z^6)$$

$$\mathrm{red}(z) = z$$

$$\mathrm{bead}(z) = 3z$$

$$\mathrm{green}(z) = z$$

$$\mathrm{blue}(z) = z$$

])

More terms are obtained by increasing the value of the global Order variable.

```
> Order := 20;
```

$$\mathit{Order} := 20$$

```
> gfseries(necklace,unlabelled,z);
```

table([

$$N(z) = 3z + 6z^2 + 11z^3 + 24z^4 + 51z^5 + 130z^6 + 315z^7 + 834z^8 + 2195z^9 + 5934z^{10} + 16107z^{11} + 44368z^{12} + 122643z^{13} + 341802z^{14} + 956635z^{15} + 2690844z^{16} + 7596483z^{17} + 21524542z^{18} + 61171659z^{19} + O(z^{20})$$

$$\mathrm{red}(z) = z$$

$$\mathrm{bead}(z) = 3z$$

$$\mathrm{green}(z) = z$$

$$\mathrm{blue}(z) = z$$

])

The function *gfsolve* will not always be able to find an answer. In these cases, it returns FAIL. Consider the class of non-plane ternary trees.

```
> sys := {G = Union(Z,Set(G,card=3))}:
> gfsolve(sys,unlabelled,x);
```

$$\mathit{FAIL}$$

However, *gfeqns* and *gfseries* will always be able to return answers.

```
> gfeqns(sys,unlabelled,x);
```

$$\left[Z(x) = x,\ G(x) = Z(x) + \frac{1}{3} G(x^3) + \frac{1}{2} G(x)\, G(x^2) + \frac{1}{6} G(x)^3 \right]$$

```
> gfseries(sys,unlabelled,x);
```

table([

$Z(x) = x$

$G(x) =$

$x + x^3 + x^5 + 2x^7 + 4x^9 + 8x^{11} + 17x^{13} + 39x^{15} + 89x^{17} + 211x^{19} + O(x^{21})$

])

Multi-variate generating functions

A multi-variate generating function can be formed by associating an extra variable with any Epsilon in the grammar which has been given a non-terminal name. The extra variable counts the number of occurences of that Epsilon. For example, if z counts atoms, and u marks the epsilons named *leaf* in a combinatorial structure A, then the associated generating function is

$$A(z, u) = \sum_{n=0}^{\infty} \left(\sum_{k=0}^{\infty} a_{n,k} u^k \right) z^n$$

where $a_{n,k}$ is the number of objects of size n which have k leaves.

Here is a general tree where the nodes are atoms and the leaves, defined by the production leaf=Epsilon, have weight zero.

```
> tree1 := {T=Union(L, Prod(N,Set(T))), L=Prod(leaf,Atom),
  leaf=Epsilon,N=Atom}:
```

We can mark the leaves of this unlabelled tree.

```
> gfeqns(tree1,unlabelled,z,[[u,leaf]]);
```

$$\left[L(z, u) = \mathrm{leaf}(z, u)\, z,\ T(z, u) = L(z, u) + N(z, u)\, e^{\left(\sum_{j_1=1}^{\infty} \frac{T\left(z^{j_1}, u^{j_1}\right)}{j_1} \right)},\ \mathrm{leaf}(z, u) = u,\ N(z, u) = z \right]$$

The same value can be given to more than one Epsilon name. Consider this description of arithmetic expressions in x using addition and multiplication.

```
> sys := {expression =
  Union(x,Prod(plus,expression,expression),Prod(times,expressio
  n,expression)), x = Atom, times = Epsilon, plus = Epsilon}:
```

We can ask that the operations plus and times both be marked with the variable u.

```
> Order := 10:
  gfseries(sys,labelled,z,[[u,times,plus]]);
```

table([

$\mathrm{expression}(z, u) = z + 2uz^2 + 8u^2z^3 + 40u^3z^4 + 224u^4z^5 + 1344u^5z^6 +$

$8448\, u^6 z^7 + 54912\, u^7 z^8 + 366080\, u^8 z^9 + O(z^{10})$

$\mathrm{plus}(z, u) = u$

$\mathrm{x}(z, u) = z$

$\mathrm{times}(z, u) = u$

])

Or we can give them different marks.

```
> gfseries(sys,labelled,z,[[u,times],[v,plus]]);
```

table([

$\mathrm{plus}(z, v, u) = v$

$\mathrm{x}(z, v, u) = z$

$$\mathrm{expression}(z, v, u) = z + (v + u)\, z^2 + (2\, v^2 + 4\, v\, u + 2\, u^2)\, z^3 + (5\, v^3 + 15\, v^2\, u + 15\, v\, u^2 + 5\, u^3)\, z^4 + (56\, u\, v^3 + 84\, v^2\, u^2 + 56\, v\, u^3 + 14\, u^4 + 14\, v^4)\, z^5 + (42\, v^5 + 210\, u\, v^4 + 420\, v^3\, u^2 + 420\, v^2\, u^3 + 210\, v\, u^4 + 42\, u^5)\, z^6 + (792\, u\, v^5 + 1980\, u^2\, v^4 + 2640\, v^3\, u^3 + 1980\, v^2\, u^4 + 792\, v\, u^5 + 132\, u^6 + 132\, v^6)\, z^7 + (429\, u^7 + 9009\, u^2\, v^5 + 15015\, u^3\, v^4 + 15015\, v^3\, u^4 + 9009\, v^2\, u^5 + 3003\, v\, u^6 + 3003\, u\, v^6 + 429\, v^7)\, z^8 + (1430\, u^8 + 80080\, u^3\, v^5 + 100100\, u^4\, v^4 + 80080\, v^3\, u^5 + 40040\, v^2\, u^6 + 11440\, v\, u^7 + 40040\, u^2\, v^6 + 11440\, u\, v^7 + 1430\, v^8)\, z^9 + O(z^{10})$$

$\mathrm{times}(z, v, u) = u$

])

We can also associate an expression, rather than a name, to the leaves. For instance, consider the following tree.

```
> sys := { tree=Union(Atom,tree2,tree3,tree4),
  tree2=Prod(node2,tree,tree),
  tree3=Prod(node3,tree,tree,tree),
  tree4=Prod(node4,tree,tree,tree,tree),node2=Epsilon,node3=Eps
  ilon,node4=Epsilon};
```

$$sys := \{tree = \mathrm{Union}(Atom, tree2, tree3, tree4),\ tree2 = \mathrm{Prod}(node2, tree, tree),\ tree3 = \mathrm{Prod}(node3, tree, tree, tree),\ tree4 = \mathrm{Prod}(node4, tree, tree, tree, tree),\ node2 = \mathrm{E},\ node3 = \mathrm{E},\ node4 = \mathrm{E}\}$$

This tree has nodes with two, three or four branches, with atoms for leaves. We use the variable u to mark the binary nodes and v to mark all the internal nodes.

```
> gfsolve(sys,labelled,z,[[u*v,node2],[v,node3,node4]]);
```

$$\{\mathbf{tree3}(z,v,u)=v\ \%1^3,\ \mathbf{tree2}(z,v,u)=v\,u\ \%1^2,\ \mathbf{node4}(z,v,u)=v,\ \mathbf{node2}(z,v,u)=v\,u,\ \mathbf{node3}(z,v,u)=v,\ \mathbf{tree4}(z,v,u)=\%1-z-v\,u\ \%1^2-v\ \%1^3,\ \mathbf{tree}(z,v,u)=\%1\}$$

$$\%1 := \mathrm{RootOf}(-\_Z+z+v\,u\ \_Z^2+v\ \_Z^3+v\ \_Z^4)$$

Then the coefficient of $u^p\ v^q\ z^n$ in tree(v, u, z) is the number of trees with n leaves that have p binary nodes and q internal nodes.

```
>
```