What are decomposable objects?
Decomposable Objects

What are decomposable objects?
Decomposable objects include trees, graphs, functions, relations, permutations, sets, subsets, cycles, lists, and much more...
Decomposable Objects

What do we want to do with decomposable objects?

- Count them.
- Generate them.
- Generate random ones.
- ...
What do we want to do with decomposable objects?

- Count them.
- Generate them.
- Generate random ones.
- ...

(in both the labeled and unlabeled cases)
Existing Software

- combstruct in Maple (Project Algo, Mishna, Murray, and Zimmermann)
- CS in MuPAD (Project Algo, Corteel, Denis, Dutour, Sarron, and Zimmerman)
- decomposableObject in MuPAD-Combinat (Cellier, Hivert, and Thiéry)
- Aldor-Combinat in Aldor/FriCAS (Hemmecke and Rubey)
Aldor-Combinat

- Started in 2006 by Ralf Hemmecke and Martin Rubey.
- Written as a fully literate program in the language of Aldor that tries to stay as close as possible to the theory of species as outlined in “Combinatorial Species and Tree-like Structures” by Bergeron, Labelle, and Leroux.
- Can be found at http://www.risc.uni-linz.ac.at/people/hemmecke/aldor/combinat/
What are species?

Let $\mathcal{B}$ be the category of finite sets with bijections. A species is simply a functor

$$F : \mathcal{B} \to \mathcal{B}.$$
What are species?

- For every finite set $A$, we get a finite set $F[A]$ whose elements are said to be the *structures* of $F$ on the underlying set $A$. 
What are species?

- For each bijection $\sigma : A \rightarrow B$, we have a bijection

$$F[\sigma] : F[A] \rightarrow F[B]$$

which is called the transport of $F$-structures along $\sigma$. 

\[A = \{a, b, c, d, e, f\}\]

\[\sigma\]

\[A = \{a, b, c, d, e, f\}\]
What are species?

- F is *functorial*, which means that
  1. $F[\text{Id}_A] = \text{Id}_{F[A]}$

Mike Hansen
Decomposable Objects and Combinatorial Species
Example: Partition Species

We define the species of partitions $P$ by letting $P[A]$ be all set partitions of $A$. 

Let $\sigma: \{1, 2, 3\} \to \{1, 2, 3\}$ be the bijection which sends 2 to 3 and 3 to 2. Then, $P[\sigma](\{\{1, 3\}, \{2\}\}) = \{\{1, 2\}, \{3\}\}$. 

Mike Hansen
Decomposable Objects and Combinatorial Species
We define the species of partitions $P$ by letting $P[A]$ be all set partitions of $A$. For example,

$$P[\{1, 2, 3\}] = \left[\{\{1, 2, 3\}\}, \{\{1, 3\}, \{2\}\}, \{\{1, 2\}, \{3\}\}, \{\{2, 3\}, \{1\}\}, \{\{1\}, \{2\}, \{3\}\}\right].$$
We define the species of partitions $P$ by letting $P[A]$ be all set partitions of $A$. For example,

$$P[\{1, 2, 3\}] = \[\{\{1, 2, 3\}\}, \{\{1, 3\}, \{2\}\}, \{\{1, 2\}, \{3\}\}, \{\{2, 3\}, \{1\}\}, \{\{1\}, \{2\}, \{3\}\}\].$$

Let $\sigma : \{1, 2, 3\} \to \{1, 2, 3\}$ be the bijection which sends 2 to 3 and 3 to 2. Then,

$$P[\sigma](\{\{1, 3\}, \{2\}\}) = \{\{1, 2\}, \{3\}\}$$
Example: Partition Species

```python
sage: P = species.PartitionSpecies()
sage: P.structures([1,2,3]).list()
[[{1, 2, 3}],
  {{1, 3}, {2}},
  {{1, 2}, {3}},
  {{2, 3}, {1}},
  {{1}, {2}, {3}}]
```

```python
sage: a = _[[1]]
a
{{1, 3}, {2}}

sage: a.transport(PermutationGroupElement((2,3)))
{{1, 2}, {3}}
```
Example: Partition Species

```python
sage: P = species.PartitionSpecies()
sage: P.structures([1,2,3]).list()
[{{1, 2, 3}},
 {{1, 3}, {2}},
 {{1, 2}, {3}},
 {{2, 3}, {1}},
 {{1}, {2}, {3}}]
```

```python
sage: a = _[1]; a
{{1, 3}, {2}}
sage: a.transport(PermutationGroupElement((2,3)))
{{1, 2}, {3}}
```

Building Blocks

- Partitions
- Permutations
- Cycles
- Sets
- Subsets
- Linear orders (sequences)
- Singleton and empty set species
Addition


The sum on the right side corresponds to a disjoint union.
Addition


The sum on the right side corresponds to a disjoint union.

**Example:**

```python
sage: P = species.PartitionSpecies()
sage: P.structures([1,2]).list()
[{{1, 2}}, {{1}, {2}}]

sage: F = P+P
sage: F.structures([1,2]).list()
[{{1, 2}}, {{1}, {2}}, {{1, 2}}, {{1}, {2}}]
```
Multiplication

\[(F \cdot G)[A] = \sum_{B+C=A} F[B] \times G[C]\]
Multiplication

Example:

```sage
sage: P = species.PartitionSpecies()
sage: F = P*P
sage: F.structures([1,2]).list()

[{{}}*{{1, 2}},
  {{}}*{{1}, {2}},
  {{{1}}}*{{2}},
  {{{2}}}*{{1}},
  {{{1, 2}}}*{},
  {{{1}, {2}}}*{}]
```
When $G[\emptyset] = \emptyset$, 

$$(F \circ G)[A] = \sum_{\pi \in P[A]} F[\pi] \times \prod_{B \in \pi} G[B]$$
Example:

```
sage: E = species.SetSpecies()
sage: Eplus = species.SetSpecies(min=1)
sage: F = E(Eplus)
sage: F.structures([1,2,3]).list()
```

```
[F-structure: {{1, 2, 3}}; G-structures: [{1, 2, 3}],
 F-structure: {{1, 3}, {2}}; G-structures: [{1, 3}, {2}],
 F-structure: {{1, 2}, {3}}; G-structures: [{1, 2}, {3}],
 F-structure: {{2, 3}, {1}}; G-structures: [{2, 3}, {1}],
 F-structure: {{1}, {2}, {3}}; G-structures: [{1}, {2}, {3}]]
```
Other Operations

- Functorial composition
- Derivative
- Pointing
- ...
Recursive definition / Implicit Equations

“A rooted tree is a root which is attached to a set of rooted trees.”

\[ A = X \cdot E(A) \]
Recursive definition / Implicit Equations

“A binary tree is either a leaf or a pair of binary trees.”

\[ B = X + B \ast B \]
Recursive definition / Implicit Equations

“A binary tree is either a leaf or a pair of binary trees.”

\[ B = X + B \times B \]

Example:

```python
sage: B = species.CombinatorialSpecies()
sage: X = species.SingletonSpecies()
sage: B.define(X+B*B)
sage: B.structures([1,2,3]).list()
[1*(2*3),
  1*(3*2),
  ...
  (2*3)*1,
  (3*2)*1]
```
Generating Series

The primary tools used in the theory of species are generating series. With each species, we can associate three different generating series:

1. (Exponential) Generating Series
2. Isomorphism Type Generating Series
3. Cycle Index Series
(Exponential) Generating Series

The (exponential) generating series of a species $F$ is given by

$$F(x) = \sum_{n \geq 0} f_n \frac{x^n}{n!}$$

where $f_n$ is the number of elements of $F[A]$ for any $A$ with $n$ elements.
(Exponential) Generating Series

The (exponential) generating series of a species $F$ is given by

$$F(x) = \sum_{n \geq 0} f_n \frac{x^n}{n!}$$

where $f_n$ is the number of elements of $F[A]$ for any $A$ with $n$ elements.

Example:
The generating series for the species of partitions is given by

$$P(x) = \sum_{n \geq 0} B_n \frac{x^n}{n!}$$

where $B_n$ are the Bell numbers.
Example:

sage: P = species.PartitionSpecies()
sage: gs = P.generating_series()
sage: gs.coefficients(5)
[1, 1, 1, 5/6, 5/8]
sage: gs
1 + x + x^2 + 5/6*x^3 + 5/8*x^4 + O(x^5)

sage: gs.counts(5)
[1, 1, 2, 5, 15]
Isomorphic Structures

Two structures $a \in F[A]$ and $b \in F[B]$ are said to be isomorphic if there exists a bijection $\sigma : A \rightarrow B$ such that

$$F[\sigma](a) = b.$$ 

Example:

```
sage: a
{{1, 3}, {2}}
sage: b
{{1, 2}, {3}}
sage: a.transport(PermutationGroupElement((2,3)))
{{1, 2}, {3}}
sage: a.is_isomorphic(b)
True
```
sage: P.isotypes([1,2,3,4]).list()

[{{1, 2, 3, 4}},
 {{1, 2, 3}, {4}},
 {{1, 2}, {3, 4}},
 {{1, 2}, {3}, {4}},
 {{1}, {2}, {3}, {4}}]
Isomorphic Structures

sage: P.isotypes([1,2,3,4]).list()
[[{1, 2, 3, 4}],
 {{1, 2, 3}, {4}},
 {{1, 2}, {3, 4}},
 {{1, 2}, {3}, {4}},
 {{1}, {2}, {3}, {4}}]

sage: B.isotypes([1,2,3,4]).list()
[1*(2*(3*4)),
 1*((2*3)*4),
 (1*2)*(3*4),
 (1*(2*3))*4,
 ((1*2)*3)*4]
The isomorphism type generating series of $F$ is defined to be

$$\tilde{F}(x) = \sum_{n \geq 0} \tilde{f}_n x^n$$

where $\tilde{f}_n$ is the number of non-isomorphic elements of $F[A]$ for any $A$ with $n$ elements.
Isomorphism Type Generating Series

Example:
The isomorphism type generating series for the species of partitions is given by

\[ \tilde{P}(x) = \sum_{n \geq 0} p_n x^n \]

where \( p_n \) is the number of integer partitions of \( n \).
Isomorphism Type Generating Series

Example:

```
sage: P = species.PartitionSpecies()
sage: itgs = P.isotype_generating_series()
sage: itgs.coefficients(5)
[1, 1, 2, 3, 5]
sage: itgs
1 + x + 2*x^2 + 3*x^3 + 5*x^4 + O(x^5)
```
Generating Series

The generating series play nicely with the operations on species:

\[(F + G)(x) = F(x) + G(x), (\tilde{F} + \tilde{G})(x) = \tilde{F}(x) + \tilde{G}(x)\]

\[(F \cdot G)(x) = F(x) \cdot G(x), (\tilde{F} \cdot \tilde{G})(x) = \tilde{F}(x) \cdot \tilde{G}(x)\]

\[(F \circ G)(x) = F(G(x))\]
Putting It Together

Rooted Trees

sage: E = species.SetSpecies()
sage: X = species.SingletonSpecies()
sage: A = species.CombinatorialSpecies()
sage: A.define(X*E(A))
sage: A.isotype_generating_series().coefficients(10)
[0, 1, 1, 2, 4, 9, 20, 48, 115, 286]
sage: sloane_find(_)[0][1]
Searching Sloane’s online database...

’Number of rooted trees with n nodes
(or connected functions with a fixed point).’
Weighted Species: Ordered Trees

```
sage: q = QQ['q'].gen()
sage: leaf = species.SingletonSpecies()
sage: internal_node = species.SingletonSpecies(weight=q)
sage: L = species.LinearOrderSpecies(min=1)
sage: T = species.CombinatorialSpecies()

sage: T.define(leaf + internal_node*L(T))
sage: T.isotype_generating_series().coefficient(4)
q^3 + 3*q^2 + q
```
Future Work (This Week!)

- Automatically recognizing recurrence relations
- More efficient random generation
- Multisort species
- Plugging in data structures into the generation routines
Thanks!