Construction of a Free Algebra
Chapter 1 Section 2
David Meretzky

In[1610]:=
```
Needs["Combinatorica`"]
```

In[1629]:=
```
Clear[A, arity, a, b, e, j, k, l, i, m, t, T, F, f];

(* Some helper functions *)
```

In[1612]:=
```
partitions[n_, arity_] := Return[
   sp = SetPartitions[n];
   For[i = 1, i ≤ Length[sp], i++,
    If[Length[sp[[i]]] < arity,
      sp = Join[sp, Permutations[PadLeft[sp[[i]], arity]]];
     ];
   ];
   For[i = 1, i ≤ Length[sp], i++,
    If[Length[sp[[i]]] > arity || Length[sp[[i]]] < arity,
      sp = Drop[sp, {i}];
      i = i - 1;
     ];
   ];
   Union[Map[Map[Length, #] &, sp]]
  ];

genCartesianProduct[l_] := Return[
   nl = {};
   For[i = 1, i <= Length[l] - 1, ++i,
    If[i == 1 && Length[l] > 1,
     nl = CartesianProduct[l[[1]], l[[i + 1]]];
    ];

    If[i == 1 && Length[l] == 1,
     Print["broke"];
     nl = l;
     Break;
    ];

    If[i > 1,
     nl = CartesianProduct[nl, l[[i + 1]]];
    ];
```

```
    ];
    Map[Flatten, nl]
   ];

F[depth_, type_] := Return[
    f = {};
    (* For each operation *)
    For[j = 1, j ≤ Length[type[[1]]] , j++,
     arity = type[[1, j]] /. ar;
     cp = CartesianProduct[{{{type[[1, j]]}}}, partitions[depth - 1, arity]];
     f = Join[f, cp];
    ];
    f
   ];

rePartition[tuple_] := Return[
    feed = tuple;
    ari = Map[# /. ar &, feed];
    isNumeric = Map[NumericQ, Map[# /. ar &, feed]];
    For[p = Length[isNumeric], p ≥ 1, --p,
     If[isNumeric[[p]],
       rep = Part[feed, (p) ;; (p + ari[[p]])];
       feed[[p]] = rep;
       feed = Drop[feed, {p + 1, p + ari[[p]]}];
      ];
    ];
    Flatten[feed, 1]
   ];
```

Create a type *T*

```
In[1616]:= T = {
    {t0, t1, t2},
    ar = {t0 → 0, t1 → 1, t2 → 2}
   };

(* Let T_n denote the subset of the type T with arity n. *)
T_n_ := Select[T[[1]], (# /. ar) == n &];

In[529]:= (* For instance, T_1 is a list of the arity 1 operations of the type T*)

In[1137]:= T_0

Out[1137]= {t0}
```

In[1138]:=
```
T[[1]]
```
Out[1138]=
```
{t0, t1, t2}
```

Create a Set X

```
(* We will use the set of two elements *)
```

In[1618]:=
```
X = {a, b}
```
Out[1618]=
```
{a, b}
```

Create F_0

In[1579]:=
```
F0 = Union[X, T0]
```
Out[1579]=
```
{a, b, t0}
```

In[1619]:=
```
FreeAlgebra[set_, type_, depth_] := Return[
   (* T0 has the arity 0 elements *)
   T0 = Select[type[[1]], (# /. type[[2]]) == 0 &];
   (* Define F0 *)
   F0 = Union[set, T0];

   rules = {0 → F0};

   For[k = 1, k ≤ depth, k++,

    (* Initialize with template *)
    Fk = F[k, type];

    (* replace template with previous Fk*)
    Fk = Map[# /. rules &, Fk];

    (* Expand Set Fk to cartesian product*)
    Fk = Flatten[Map[genCartesianProduct[FlattenAt[#, {2}]] &, Fk], 1];

    (* Add a new rule for substituting at the next level *)
    rules = Join[rules, {k → Fk}];
   ];
   (* repartition the set *)
   For[k = 1, k ≤ depth, k++,
    Fk = Map[rePartition, Fk];
   ];

  ];
```

Create a Free Algebra on the set and type up to a certain level

```
In[1620]:= FreeAlgebra[X, T, 5]

In[1621]:= Length[F0]

Out[1621]= 3

In[1622]:= Length[F1]

Out[1622]= 12

In[1623]:= Length[F2]

Out[1623]= 84

In[1624]:= Length[F3]

Out[1624]= 732

In[1625]:= Length[F4]

Out[1625]= 7140

In[1626]:= Length[F5]

Out[1626]= 74 604

In[1627]:= F0

Out[1627]= {a, b, t0}

In[1602]:= Column[F1]

           {t1, a}
           {t1, b}
           {t1, {t0}}
           {t2, a, a}
           {t2, a, b}
           {t2, a, {t0}}
Out[1602]= {t2, b, a}
           {t2, b, b}
           {t2, b, {t0}}
           {t2, {t0}, a}
           {t2, {t0}, b}
           {t2, {t0}, {t0}}

In[1601]:= Column[F2]

           {t1, {t1, a}}
           {t1, {t1, b}}
           {t1, {t1, {t0}}}
           {t1, {t2, a, a}}
           {t1, {t2, a, b}}
           {t1, {t2, a, {t0}}}
           {t1, {t2, b, a}}
           {t1, {t2, b, b}}
           {t1, {t2, b, {t0}}}
           {t1, {t2, {t0}, a}}
           {t1, {t2, {t0}, b}}
```

```
{t1, {t2, {t0}, {t0}}}
{t2, a, {t1, a}}
{t2, a, {t1, b}}
{t2, a, {t1, {t0}}}
{t2, a, {t2, a, a}}
{t2, a, {t2, a, b}}
{t2, a, {t2, a, {t0}}}
{t2, a, {t2, b, a}}
{t2, a, {t2, b, b}}
{t2, a, {t2, b, {t0}}}
{t2, a, {t2, {t0}, a}}
{t2, a, {t2, {t0}, b}}
{t2, a, {t2, {t0}, {t0}}}
{t2, b, {t1, a}}
{t2, b, {t1, b}}
{t2, b, {t1, {t0}}}
{t2, b, {t2, a, a}}
{t2, b, {t2, a, b}}
{t2, b, {t2, a, {t0}}}
{t2, b, {t2, b, a}}
{t2, b, {t2, b, b}}
{t2, b, {t2, b, {t0}}}
{t2, b, {t2, {t0}, a}}
{t2, b, {t2, {t0}, b}}
{t2, b, {t2, {t0}, {t0}}}
{t2, {t0}, {t1, a}}
{t2, {t0}, {t1, b}}
{t2, {t0}, {t1, {t0}}}
{t2, {t0}, {t2, a, a}}
{t2, {t0}, {t2, a, b}}
Out[1601]= {t2, {t0}, {t2, a, {t0}}}
{t2, {t0}, {t2, b, a}}
{t2, {t0}, {t2, b, b}}
{t2, {t0}, {t2, b, {t0}}}
{t2, {t0}, {t2, {t0}, a}}
{t2, {t0}, {t2, {t0}, b}}
{t2, {t0}, {t2, {t0}, {t0}}}
{t2, {t1, a}, a}
{t2, {t1, a}, b}
{t2, {t1, a}, {t0}}
{t2, {t1, b}, a}
{t2, {t1, b}, b}
{t2, {t1, b}, {t0}}
{t2, {t1, {t0}}, a}
{t2, {t1, {t0}}, b}
{t2, {t1, {t0}}, {t0}}
{t2, {t2, a, a}, a}
{t2, {t2, a, a}, b}
{t2, {t2, a, a}, {t0}}
{t2, {t2, a, b}, a}
{t2, {t2, a, b}, b}
{t2, {t2, a, b}, {t0}}
{t2, {t2, a, {t0}}, a}
{t2, {t2, a, {t0}}, b}
{t2, {t2, a, {t0}}, {t0}}
{t2, {t2, b, a}, a}
{t2, {t2, b, a}, b}
{t2, {t2, b, a}, {t0}}
{t2, {t2, b, b}, a}
{t2, {t2, b, b}, b}
```

```
{t2, {t2, b, b}, {t0}}
{t2, {t2, b, {t0}}, a}
{t2, {t2, b, {t0}}, b}
{t2, {t2, b, {t0}}, {t0}}
{t2, {t2, {t0}, a}, a}
{t2, {t2, {t0}, a}, b}
{t2, {t2, {t0}, a}, {t0}}
{t2, {t2, {t0}, b}, a}
{t2, {t2, {t0}, b}, b}
{t2, {t2, {t0}, b}, {t0}}
{t2, {t2, {t0}, {t0}}, a}
{t2, {t2, {t0}, {t0}}, b}
{t2, {t2, {t0}, {t0}}, {t0}}
```

In[1628]:= **Column[F_3[[1 ;; 75]]]**

```
{t1, {t1, {t1, a}}}
{t1, {t1, {t1, b}}}
{t1, {t1, {t1, {t0}}}}
{t1, {t1, {t2, a, a}}}
{t1, {t1, {t2, a, b}}}
{t1, {t1, {t2, a, {t0}}}}
{t1, {t1, {t2, b, a}}}
{t1, {t1, {t2, b, b}}}
{t1, {t1, {t2, b, {t0}}}}
{t1, {t1, {t2, {t0}, a}}}
{t1, {t1, {t2, {t0}, b}}}
{t1, {t1, {t2, {t0}, {t0}}}}
{t1, {t2, a, {t1, a}}}
{t1, {t2, a, {t1, b}}}
{t1, {t2, a, {t1, {t0}}}}
{t1, {t2, a, {t2, a, a}}}
{t1, {t2, a, {t2, a, b}}}
{t1, {t2, a, {t2, a, {t0}}}}
{t1, {t2, a, {t2, b, a}}}
{t1, {t2, a, {t2, b, b}}}
{t1, {t2, a, {t2, b, {t0}}}}
{t1, {t2, a, {t2, {t0}, a}}}
{t1, {t2, a, {t2, {t0}, b}}}
{t1, {t2, a, {t2, {t0}, {t0}}}}
{t1, {t2, b, {t1, a}}}
{t1, {t2, b, {t1, b}}}
{t1, {t2, b, {t1, {t0}}}}
{t1, {t2, b, {t2, a, a}}}
{t1, {t2, b, {t2, a, b}}}
{t1, {t2, b, {t2, a, {t0}}}}
{t1, {t2, b, {t2, b, a}}}
{t1, {t2, b, {t2, b, b}}}
{t1, {t2, b, {t2, b, {t0}}}}
{t1, {t2, b, {t2, {t0}, a}}}
{t1, {t2, b, {t2, {t0}, b}}}
{t1, {t2, b, {t2, {t0}, {t0}}}}
{t1, {t2, {t0}, {t1, a}}}
```
Out[1628]=
```
{t1, {t2, {t0}, {t1, b}}}
{t1, {t2, {t0}, {t1, {t0}}}}
{t1, {t2, {t0}, {t2, a, a}}}
{t1, {t2, {t0}, {t2, a, b}}}
{t1, {t2, {t0}, {t2, a, {t0}}}}
{t1, {t2, {t0}, {t2, b, a}}}
{t1, {t2, {t0}, {t2, b, b}}}
```

```
{t1, {t2, {t0}, {t2, b, {t0}}}}
{t1, {t2, {t0}, {t2, {t0}, a}}}
{t1, {t2, {t0}, {t2, {t0}, b}}}
{t1, {t2, {t0}, {t2, {t0}, {t0}}}}
{t1, {t2, {t1, a}, a}}
{t1, {t2, {t1, a}, b}}
{t1, {t2, {t1, a}, {t0}}}
{t1, {t2, {t1, b}, a}}
{t1, {t2, {t1, b}, b}}
{t1, {t2, {t1, b}, {t0}}}
{t1, {t2, {t1, {t0}}, a}}
{t1, {t2, {t1, {t0}}, b}}
{t1, {t2, {t1, {t0}}, {t0}}}
{t1, {t2, {t2, a, a}, a}}
{t1, {t2, {t2, a, a}, b}}
{t1, {t2, {t2, a, a}, {t0}}}
{t1, {t2, {t2, a, b}, a}}
{t1, {t2, {t2, a, b}, b}}
{t1, {t2, {t2, a, b}, {t0}}}
{t1, {t2, {t2, a, {t0}}, a}}
{t1, {t2, {t2, a, {t0}}, b}}
{t1, {t2, {t2, a, {t0}}, {t0}}}
{t1, {t2, {t2, b, a}, a}}
{t1, {t2, {t2, b, a}, b}}
{t1, {t2, {t2, b, a}, {t0}}}
{t1, {t2, {t2, b, b}, a}}
{t1, {t2, {t2, b, b}, b}}
{t1, {t2, {t2, b, b}, {t0}}}
{t1, {t2, {t2, b, {t0}}, a}}
{t1, {t2, {t2, b, {t0}}, b}}
{t1, {t2, {t2, b, {t0}}, {t0}}}
```

Draft Code

```
(*


FreeAlgebra[set_,type_,depth_]:=Return[
  free = {};
  T0 =Select[type[[1]], (#/.ar)==0&];
  F0 = Union[set,T0];
  For[k =1 ,k≤ depth,++k,
   f ={};
   f= F[k,type];

   Print[k];
   Print[f];
   AppendTo[free,f];
  ];
  free
 ];
```

```
FreeAlgebra[set_,type_,depth_]:=Return[
  free = {};
  T0 =Select[type[[1]], (#/.ar)==0&];
  F0 = Union[set,T0];
  For[k =1 ,k≤ depth,++k,
   Fk = Flatten[Map[genCartesianProduct,F[k,T]],1];
   AppendTo[free,Fk];
  ];
  free
 ];

For[k = 1, k ≤ Length[ft1],k++,
 set = Union[ft1[[k,2]]];
 For[p = 1, p ≤ Length[set],++p,
  ft1[[k,2]] =
    ft1[[k,2]]/.set[[p]]→ Fset[[p]]
 ];
];



freeAlgebra[depth_,type_,set_]:= Return[
   F = {};
  For[n = 1, n≤ depth,n++,
   f={};
   (* For each operatior *)
   For[j = 1,j ≤ Length[type[[1]]] ,j++,
    Print[type[[1,j]]];
    a = type[[1,j]]/. ar;
    Print[a];
    Print[partitions[depth-1, a]];
    cp = CartesianProduct[{type[[1,j]]},partitions[depth-1, a]];
    Print[cp];
    AppendTo[f,cp];
   ];
   AppendTo[F,f];
  ];
 ];

F[depth_,type_]:= Return[
   f= {};
```

```
  (* For each operatior *)
  For[j = 1,j ≤ Length[type[[1]]] ,j++,
   arity = type[[1,j]]/. ar;

   cp = {{type[[1,j]]},Map[F#&,partitions[depth-1, arity][[1]]]};
   cp = FlattenAt[cp,2];
   f = AppendTo[f,cp];

  ];
  f
 ];



*)
```