Chasing the blend

F. Bou
M. Schorlemmer
IIIA, CSIC
Barcelona

J. Corneli
Computing
Goldsmiths, London

D. Gómes-Ramírez
Cognitive Science
Osnabrück

E. Maclean
A. Smaill
Informatics
Edinburgh

A. Pease
Computing
Dundee

Abstract

We model the mathematical process whereby new mathematical theories are produced, involving shared and individual creativity. Here we provide rational reconstructions of some developments from mathematical history; our longer-term goal is to support machine and human mathematical creativity.

Introduction

To be written by Alan.

Background

Blending in Mathematics

Alison?

Image Schemas

Marco

Blending and the infinite

Note: *Marco, Ewen, Alan, Felix*

Naturals and Integers

A Simple Example – the Integers

In order to demonstrate the machinery involved in blending mathematical theories, we consider combining a theory of natural numbers with the concept of the inverse of a function to obtain the integers. Let us assume an simple axiomatisation of the natural numbers (without order axioms) as shown in Figure 1, and call this theory $\mathbb{N}$. Now let us also define a simple theory which introduces the concept of a function with an inverse as shown in Figure 2, and call this theory $\mathbb{F}$.

Identifying a Generic Space In order to incorporate the notion of blending here we want to be able to identify a "generic" component of each theory and compute the pushout as discusses in §. We can use the HDTP system (**GustKS2006**; **Schmidt2010**) to discover a common theory and signature morphism between symbols in the two theories $\mathbb{N}$ and $\mathbb{F}$. The Generic theory contains a sort N and a function $func$, and the morphisms from the Generic theory to $\mathbb{N}$ and $\mathbb{F}$ are:

$$s \quad \leftarrow_{g_{\mathbb{N}}} \quad func \quad \rightarrow_{g_{\mathbb{F}}} \quad f \tag{1}$$

$$Nat \quad \leftarrow_{g_{\mathbb{N}}} \quad N \quad \rightarrow_{g_{\mathbb{F}}} \quad X \tag{2}$$

```
spec NAT =
     sort  Nat
     ops   zero : Nat;
           s : Nat → Nat;
           __+__ : Nat × Nat → Nat
     ∀ x, y, z : Nat
     • s(x) = y ∧ s(x) = z ⇒ y = z
     • s(x) = s(y) ⇒ x = y
     • ∃ a : Nat • s(x) = a
     • ¬ s(x) = zero
     • s(x) + y = s(x + y)
     • zero + y = y
end
```

Figure 1: A theory of the natural numbers without order

```
spec FUNC =
     sort  X
     op    f : X → X
     op    finv : X → X
     ∀ x : X
     • f(finv(x)) = x
     • finv(f(x)) = x
end
```

Figure 2: A theory with a function and its inverse defined

Here the successor function is identified in the mapping with the function in the theory $\mathbb{F}$, and g_K is the label for the set of symbol mappings determined by the signature morphism from the Generic space the theory K.

Computing the Colimit The HETS system (**MossakowskiEA06**) can then be exploited to find a new theory by computing the colimit:

$$\begin{tikzcd}
 & \text{Gen} \arrow[dl, "g_{\mathbb{N}}"'] \arrow[dr, "g_{\mathbb{F}}"] & \\
\mathbb{N} \arrow[dr, "b_{\mathbb{N}}"'] & & \mathbb{F} \arrow[dl, "b_{\mathbb{F}}"] \\
 & \text{Blend} &
\end{tikzcd}$$

This generates the theory shown in 3.

```
spec SPEC =
     sort  N
     op    __+__ : N × N → N
     op    p : N → N
     op    s : N → N
     op    zero : N
     ∀ x, y, z : N • s(x) = y ∧ s(x) = z ⇒ y = z    %(Ax1)%
     ∀ x, y : N • s(x) = s(y) ⇒ x = y               %(Ax2)%
     ∀ x : N • ∃ a : N • s(x) = a                   %(Ax3)%
     ∀ x : N • ¬ s(x) = zero                        %(Ax4)%
     ∀ x, y : N • s(x) + y = s(x + y)               %(Ax5)%
     ∀ y : N • zero + y = y                         %(Ax6)%
     ∀ x : N • s(p(x)) = x                        %(Ax1_7)%
     ∀ x : N • p(s(x)) = x                        %(Ax2_8)%
```

Figure 3: An inconsistent version of the integers (without order)

Removal of Inconsistencies This theory is automatically determined to be inconsistent due to the axioms

$$\forall x : \mathbb{Z}.nots(x) = 0 \tag{3}$$

$$s(p(x)) = x \tag{4}$$

Removal of the limiting axiom (3) results in a theory which is very similar to what we understand to be the integers as shown in Figure 4.

```
spec SPEC =
     sort  N
     op    __+__ : N × N → N
     op    p : N → N
     op    s : N → N
     op    zero : N
     ∀ x, y, z : N • s(x) = y ∧ s(x) = z ⇒ y = z    %(Ax1)%
     ∀ x, y : N • s(x) = s(y) ⇒ x = y               %(Ax2)%
     ∀ x : N • ∃ a : N • s(x) = a                   %(Ax3)%
     ∀ x, y : N • s(x) + y = s(x + y)               %(Ax4)%
     ∀ y : N • zero + y = y                         %(Ax5)%
     ∀ x : N • s(p(x)) = x                        %(Ax1_7)%
     ∀ x : N • p(s(x)) = x                        %(Ax2_8)%
end
```

Figure 4: A consistent version of the integers (without order)

Running the Blend Running the blend refers to discovering axioms or definitions which make the blend incomplete. In the example of the version in Figure 4, the definition of plus needs to be extended to understand how to calculate with the predecessor function:

$$p(x) + y = p(x + y)$$

from which theorems such as

$$p(x) + s(y) = x + y$$

can be discovered.

Potential and actual infinity

Prime Ideals as a blend

Introduction

One of the most fundamental concepts of modern mathematics, which is the basis of commutative algebra and a seminal ingredient of the language of schemes in modern algebraic geometry is the one of prime ideal (**EGAI**; **Eisenbud95**).

In this section, we present an implementation done in Hets (**Mossakowskihets**), in the language of CASL for the case of a principal ideal domain (PID). The resulting blending space contains two equivalent definitions of the containing relation for ideals. One of them is the trivial one in terms of elements and the other one is given in terms of product of ideals. It states that an ideal X is contained in an ideal Y if and only if there exists an ideal C such that $X = Y * C$. This is true if the base ring is a PID (it is an elementary exercise). However, if the base ring is not a PID, for example $R = \mathbb{Z}[T]$, then one can check that the ideals $X = (2)$ and $Y = (2, T)$ gives a counterexample. Here, it is important to point out that in this implementation we looked for a minimal set of axioms such that, at the same time, the semantic interpretation can be uniquely determined. It is always possible to construct an implementation with additional axioms given by properties that could be logically derived from the main axioms, (e.g. the set theoretical properties of the containment relation for subsets of a set) but these properties are secondary ones, meanwhile, the ones defining the arithmetic of the ring, of an ideal and of the set of ideals of the ring are the essential ones.

We will recover the concept of prime ideal of a commutative ring with unity as a sort of partial (or weaken) blending (i.e. a blend for just some axioms of the input theories) between the concepts of an ideal of a commutative ring with unity (enriched with the collection of all the ideals of the corresponding ring) and the concept of a prime number of the integers.

In fact, in order to obtain the desired space it is enough to consider a more general version of the prime numbers (in our case a partial version), namely, a monoid $(Z, *, 1)$ with an "special" divisibility relation $\lfloor$. Besides, the generic space would capture just the syntactic correspondences that we wish to identity in the blending space, since the blend would be basically the union of the collection of axioms given on each space, doing the corresponding identifications.

Our approach to blending is the one adopted by Goguen in terms of colimits (**Gog99**; **Goguen01**; **Goguen05c**).

We present the conceptual spaces from the standard "pure" mathematical point of view doing concurrently the corresponding translation into the setting of the Common Algebraic Specification Language (CASL) (**BidoitMosses2004**).

The first conceptual space

Let $(R,+.*,0,1)$ be a commutative ring with unity, i.e. R is a set with two binary operations, $+$ and $*$, and two special elements $0,1\in R$ satisfying the following axioms:

1. $(\forall a\in R)(a+0=0+a=a)$
2. $(\forall a\in R)(\exists b\in R)(a+b=b+a=0)$
3. $(\forall a,b,c\in R)((a+b)+c=a+(b+c)))$
4. $(\forall a,b\in R)(a+b=b+a)$
5. $(\forall a\in R)(a*1=1*a=a)$
6. $(\forall a,b,c\in R)((a*b)*c=a*(b*c)))$
7. $(\forall a,b\in R)(a*b=b*a)$
8. $(\forall a,b,c\in R)(a*(b+c)=a*b+a*c)$

Now, R can be understood as the sort containing the elements of the corresponing commutative ring with unity. An ideal I is a subset of R satisfying the following axiom:

$$(\forall i,j\in I)(\forall r\in R)(i+(-j)\in I\wedge r*i\in I).$$

Let us define a unary relation (predicate) $isideal$ on the set (sort) of subsets of R, $P(R)$, as follows: $isideal(I)$ if and only if I is an ideal of R.

Now, we define

$$\mathrm{Spec}_I R=\{A\subseteq R: isideal(A)\}.$$

Here, $\mathrm{Spec}_I R$ is considered as a subsort of the sort $P(R)$.

There is one natural operation on $\mathrm{Spec}_I R$, let us say $\cdot_\iota$, inherited in a natural way from the corresponding operations $+$ and $\cdot$ on S:

Let $I,J\in\mathrm{Spec}_I R$, then we define

$$I\cdot_\iota J:=\{i_1\cdot j_1+...+i_n\cdot j_n: n\in\mathbb{N}\wedge i_k\in I\wedge j_k\in J\}.$$

With this operation $\mathrm{Spec}_I R$ forms a commutative monoid (i.e. it holds commutativity, associativity and there exists a neutral element (in this case the ring)). However, this fact is irrelevant in our case for the blending process. As a matter of fact, the only property that we want to keep into the blend is the one saying that this operation has a neutral element 1_ι, which can be seen as an additional notation for the ring, but respect to this operation $\cdot_\iota$ instead of being the sort of elements of the ring, i.e., R.

On the other side, we want to see the contention relation $\subseteq$ as a binary relation over the sort $\mathrm{Spec}_I R$.

Summarizing, our first conceptual space consists of sorts $R,\mathrm{Spec}_I R$ and $P(R)$; operations $+_R,*_R,0_R,1_R,1_\iota$ and $\cdot_\iota$; and the relations $\subseteq$ and $isideal$.

Here we add all the corresponding axioms defining R as a commutative ring, the explicit former definition of $isideal$, $\mathrm{Spec}_I R$ and $\cdot_\iota$; and the axiom guaranteeing that 1_ι is the neutral element for $\cdot_\iota$.

Let us denote this space by $\mathbb{I}$.

The second conceptual space

Let $\mathbb{Z}$ be the set of the integer numbers. Here, we can choose any axiomatization of them, since for the (partial) blending we just take into account only the fact that $(\mathbb{Z},*,1)$ is a commutative monoid. Or even simpler, we only use the fact that 1 is the neutral element with the operation $*$. One can, for example, take the simple characterization of $\mathbb{Z}$, given by Martin Brandenburg (**brandenburgdobleinduction**), as the only ordered commutative ring with unity satisfying the following "bi-inductive" property:

$$\forall M\subseteq\mathbb{Z}\,[0\in M\wedge(\forall n\in\mathbb{Z}(n\in M\to n\pm 1\in M)) \\ \to M=\mathbb{Z}].$$

We define also an upside down divisibility relation $\lfloor$ defined as

$$e\lfloor g:=g|e,$$

We re-write the classical divisibility relation on this way in order to obtain the right primality condition on the blend. Let us define a unary relation $isprime$ on $\mathbb{Z}$ as follows: for all $p\in\mathbb{Z}$, $isprime(p)$ holds if $p\neq 1$ and the following (primality) condition holds:

$$(\forall a,b\in\mathbb{Z})(ab\lfloor p)\to(a\lfloor p\vee b\lfloor p).$$

Besides, we define the set (sort) of the prime numbers as

$$Prime=\{p\in\mathbb{Z}/isprime(p)\}$$

Now, it is an elementary fact to see that this condition is an equivalent form of the standard definition of prime number given in the classical number theory books (see for example **Apostol76** In the CASL language, we consider $\mathbb{Z}$ as the sort of the integer numbers, $*$ as a binary operation , $prime$ as a predicate and $\lfloor$ as a binary relation, any of them defined over the sort $\mathbb{Z}$.

We denote this conceptual space by $\mathbb{P}$.

The Generic Space

The generic space consists of a set (sort) G with a binary operation $*_G$, a neutral element S and a binary relation $\leq_G$.

Let us denote this space by $\mathbb{G}$.

The Blending Morphisms

Now, let us define the morphisms from the generic space into the two corresponding conceptual spaces. Let $\varphi:\mathbb{G}\to\mathbb{I}$ be the morphism induced by the following syntactic correspondences $\varphi(G)=\mathrm{Spec}_I R,\varphi(*_G)=*_\iota,\varphi(S)=1_\iota$ and $\varphi(\leq_G)=\subseteq$.

Furthermore, let $\delta:=\mathbb{G}\to\mathbb{P}$ be the morphism induced by the syntactic correspondences $\delta(G)=\mathbb{Z},\delta(*_G)=*,\delta(S)=1$ and $\delta(\leq_G)=\lfloor$.

The Axiomatization of the Blending

In the every-day research of the working mathematician it happens frequently that one starts to develop general theoretical frameworks by combining just some aspects of two particular theories but without considering the whole theories. For example, the development of differential geometry was obtained combining just some aspect of general and algebraic topology and some aspects of real analysis (**VelCad05**). The same happens with the methods use in analytic number theory which are a fusion of some components of elementary number theory and some of the real analysis techniques (**Apostol76**).

Therefore, it is more natural in the daily mathematical research to obtain new concepts as "partial" combinations of two former ones, i.e., as combinations (blends) of just some axioms of the corresponding two theories.

Thus, in our case, a partial blend will give us the desired concept. For example, from the properties defining the integers we transfer into the blend only the fact that $\mathbb{Z}$ is a set with a binary operation $*$ having 1 as neutral element.

So, after using the same symbols for denoting the ring as a sort of elements or as the neutral element for product of ideals $\cdot_G$, the blend has the form

$$(S, +_S, *_S, 0_S, 1_S, G = \mathrm{Spec}_I S, isprime, Prime, \cdot_G, S = 1_G, \subseteq)$$

with all the corresponding axioms of the first conceptual space plus the translated version of the axiom defining the primality predicate after doing the corresponding symbolic identifications i.e., an element $P \in G$ (i.e., an ideal of S) satisfied the predicate $isprime$ if and only if

$$\begin{aligned} &P \neq S \wedge (\forall X, Y \in G = \mathrm{Spec}_I S). \\ &\qquad (X \cdot_\alpha Y \subseteq P \rightarrow (X \subseteq P \vee Y \subseteq P)). \end{aligned}$$

Now, it is an elementary exercise to see that this definition is equivalent to the fact that P is a prime ideal of S, i.e. to the condition

$$P \neq S \wedge (\forall a, b \in S)(ab \in B \rightarrow (a \in P \vee b \in P)).$$

Therefore, the predicate $isprime$ turns out to be the predicate characterizing the primality of ideals of S and the set (sort) $Prime$ turns out to be the set of prime ideals of S.

Besides, we just consider the fact that the up-side down divisibility relation is a binary relation without taking into account the formal definition into the blend.

In conclusion, the blending space consists of the axioms assuring that S is a commutative ring with unity, G is the set of ideals of S, $isprime$ is the predicate specifying primality for ideals of S and $Prime$ is the collection of all prime ideals of S.

Implementation for the Principal Ideal Domain Case

On this section we present an implementation done in Hets (**Mossakowskihets**), in the language of CASL for the case of a principal ideal domain (PID). The resulting blending space contains two equivalent definitions of the containing relation for ideals. One of them is the trivial one in terms of elements and the other one is given in terms of product of ideals. It is an elementary exercise to see this equivalence in the PID case.

```
library ideal_blend

logic CASL

%% Prime Ideals over Principal Ideal Domains (PID) as
%% Blends in Casl

spec IDEALSOFRING =
     sort RingElt
              %% sort of Ring Elts
     sort SubSetOfRing
              %% not further defined for the moment
     pred IsIdeal : SubSetOfRing
              %% when a subset is an ideal
     op   0 : RingElt
     op   1 : RingElt
     op   __*__ : RingElt × RingElt → RingElt
     op   __+__ : RingElt × RingElt → RingElt
     pred __isIn__ : RingElt × SubSetOfRing
     sort Ideal = {I : SubSetOfRing • IsIdeal(I)}
     op   R : Ideal
              %% the Ring as an ideal
     op   __**__ : Ideal × Ideal → Ideal, unit R
     pred __issubsetOf__ : Ideal × Ideal
              %% partial order
              %%Definition of the containment predicate
     ∀ A, B : Ideal
     • A issubsetOf B ⇔ ∀ a : RingElt • a isIn A ⇒ a isIn B
     %% axioms for Ring
     ∀ x : RingElt; y : RingElt
     • x + y = y + x
     %Commutativity with +%
     ∀ x : RingElt; y : RingElt; z : RingElt
     • (x + y) + z = x + (y + z)
     %% Associativity with +
     ∀ x : RingElt
     • x + 0 = x ∧ 0 + x = x
     %%unit with +
     ∀ x : RingElt
     • ∃ x' : RingElt • x' + x = 0
     %%Inverse with +
     ∀ x : RingElt; y : RingElt
     • x * y = y * x
     %%Commutativity with *
     ∀ x : RingElt; y : RingElt; z : RingElt
     • (x * y) * z = x * (y * z)
     %%Associativity with *
     ∀ x : RingElt
     • x * 1 = x ∧ 1 * x = x
     %%unit with *
     ∀ x, y, z : RingElt
     • (x + y) * z = (x * z) + (y * z)
     %%Left_Distributivity
     ∀ x, y, z : RingElt
     • z * (x + y) = (z * x) + (z * y)
     %%Right_Distributivity
     %%axioms for Ideal
     ∀ I : SubSetOfRing
     • IsIdeal(I)
       ⇔ ∀ a, b, c : RingElt
           • ((a isIn I ⇒ a isIn R) ∧ 0 isIn I)
             ∧ (a isIn I ∧ c isIn R ⇒ c * a isIn I)
             ∧ (a isIn I ∧ b isIn I ∧ c isIn R ∧ b + c = 0
            ⇒ a + c isIn I)

    %%axioms for PID−s
    pred  __generates__ : RingElt × Ideal
          %%an ideal is generated by an element
    ∀ A : Ideal • ∃ a : RingElt • a generates A
    ∀ a : RingElt
    • ∀ A : Ideal
      • a generates A
        ⇔ ∀ c : RingElt • c isIn A ⇒ ∃ d : RingElt • c = a ∗ d
    %% Definition of the product of ideals
    ∀ A, B : Ideal
    • ∀ a, b : RingElt • a isIn A ∧ b isIn B ⇒ a ∗ b isIn A ∗∗ B
    • ∀ D : Ideal
      • (∀ a, b : RingElt • a isIn A ∧ b isIn B ⇒ a ∗ b isIn A ∗∗ B)
        ⇒ A ∗∗ B issubsetOf D
end
%% axioms defining a very simple version of the integers,
%% considered with an operation ∗, a binary relation ||
%% (upside−down divisibility relation) and a primality axiom.

spec SimpleInt =
    sort  SimpleElem
    ops   1 : SimpleElem;
          __x__ : SimpleElem × SimpleElem → SimpleElem,
          comm, assoc, unit 1
    preds __||__ : SimpleElem × SimpleElem;
          %% division gives partial order
          IsPrime : SimpleElem
    ∀ x, y : SimpleElem
    • x || y ⇔ ∃ c : SimpleElem • x = y x c
    %Def_upsidedownDivisilityRelation%
    %% subsort of primes
    sort  SimplePrime = {p : SimpleElem • IsPrime(p)}
    ∀ p : SimpleElem
    • IsPrime(p)
      ⇔ (∀ a, b : SimpleElem • a x b || p ⇒ a || p ∨ b || p) ∧ ¬ p = 1
    %Def_primality%
end
%% Generic space

spec Gen =
    sort  Generic
    ops   S : Generic;
          __gpr__ : Generic × Generic → Generic, unit S
    pred  gcont : Generic × Generic
end

view I1 :
    Gen to IdealsOfRing =
    Generic ↦ Ideal, S ↦ R, __gpr__ ↦ __∗∗__,
    gcont ↦ __issubsetOf__
end

view I2 :
    Gen to SimpleInt =
    Generic ↦ SimpleElem, S ↦ 1, __gpr__ ↦ __x__,
    gcont ↦ __||__
end

spec Colimit =
    combine I1, I2
end
```

After computing the corresponding colimit in HETS and after interpreting “RingEl” as the sort containing the elements of the ring S, the theory defining the blend corresponds to the axioms defining a PID (S), the set of all its ideals (Generic) and the set all its prime ideals (SimplePrime):

```
library ideal_colim

logic CASL.SulFOL=

spec Spec =
    sorts Generic, RingElt, SimplePrime, SubSetOfRing
    sorts SimplePrime < Generic;
          Generic < SubSetOfRing
    op    0 : RingElt
    op    1 : RingElt
    op    S : Generic
    op    __∗__ : RingElt × RingElt → RingElt
    op    __+__ : RingElt × RingElt → RingElt
    op    __x__ : Generic × Generic → Generic
    pred  IsIdeal : SubSetOfRing
    pred  IsPrime : Generic
    pred  __generates__ : RingElt × Generic
    pred  __isIn__ : RingElt × SubSetOfRing
    pred  gcont : Generic × Generic
    ∀ I : SubSetOfRing • I ∈ Generic ⇔ IsIdeal(I) %(Ax1)%
    ∀ x : Generic
    • x x S = x
                                  %(ga_right_unit___∗∗__)%
    ∀ x : Generic
    • S x x = x
                                   %(ga_left_unit___∗∗__)%
    ∀ A, B : Generic
    • gcont(A, B) ⇔ ∀ a : RingElt • a isIn A ⇒ a isIn B %(Ax4)%
    ∀ x, y : RingElt • x + y = y + x              %(Ax5)%
    ∀ x, y, z : RingElt • (x + y) + z = x + (y + z) %(Ax6)%
    ∀ x : RingElt • x + 0 = x ∧ 0 + x = x         %(Ax7)%
    ∀ x : RingElt • ∃ x' : RingElt • x' + x = 0   %(Ax8)%
    ∀ x, y : RingElt • x ∗ y = y ∗ x              %(Ax9)%
    ∀ x, y, z : RingElt • (x ∗ y) ∗ z = x ∗ (y ∗ z) %(Ax10)%
    ∀ x : RingElt • x ∗ 1 = x ∧ 1 ∗ x = x         %(Ax11)%
    ∀ x, y, z : RingElt • (x + y) ∗ z = (x ∗ z) + (y ∗ z) %(Ax12)%
    ∀ x, y, z : RingElt • z ∗ (x + y) = (z ∗ x) + (z ∗ y) %(Ax13)%
    ∀ I : SubSetOfRing
    • IsIdeal(I)
      ⇔ ∀ a, b, c : RingElt
        • ((a isIn I ⇒ a isIn S) ∧ 0 isIn I)
          ∧ (a isIn I ∧ c isIn S ⇒ c ∗ a isIn I)
          ∧ (a isIn I ∧ b isIn I ∧ c isIn S ∧ b + c = 0
             ⇒ a + c isIn I)
                                                 %(Ax14)%
    ∀ A : Generic • ∃ a : RingElt • a generates A %(Ax15)%
    ∀ a : RingElt; A : Generic
    • a generates A
    ⇔ ∀ c : RingElt • c isIn A ⇒ ∃ d : RingElt • c = a ∗ d
                                                %(Ax16)%
  ∀ A, B : Generic; a, b : RingElt
  • a isIn A ∧ b isIn B ⇒ a ∗ b isIn A x B      %(Ax17)%
  ∀ A, B, D : Generic
  • (∀ a, b : RingElt • a isIn A ∧ b isIn B ⇒ a ∗ b isIn A x B)
    ⇒ gcont(A x B, D)
                                                %(Ax18)%
  ∀ x, y : Generic
  • gcont(x, y) ⇔ ∃ c : Generic • x = y x c      %(Ax3)%
  ∀ p : Generic • p ∈ SimplePrime ⇔ IsPrime(p) %(Ax4_19)%
  ∀ p : Generic
  • IsPrime(p)
    ⇔ (∀ a, b : Generic
         • gcont(a x b, p) ⇒ gcont(a, p) ∨ gcont(b, p))
       ∧ ¬ p = S
                                               %(Ax5_20)%
end
```

Related Example

Galois Theory

This example is developed in a less formal manner than the previous. It is included both to highlight some outstanding technical issues, and to develop some broader theoretical points about the future prospects and applications of our approach.

In its most straightforward formulation, Galois theory develops a relationship between a polynomial $f(x)$ with coefficients in some field F, the extension of K of F (written "K/F") containing all of the roots of $f(x)$, and the group $\mathbf{Gal}(K)$ of automorphisms of K/F that fix the elements of F. The fundamental theorem of Galois theory states that there is a bijection between the subfields of K/F and the subgroups of $\mathbf{Gal}(K)$; namely, subgroups correspond to their fixed fields. Using this correspondence, properties of polynomials can be derived, most famously the fact that quintic polynomials cannot be solved by algebraic operations and the extraction of roots.

We do not propose to reconstruct much of the theory here, but note that already in this basic account there are several steps that seem compellingly "blend-like."

In the first place, that would describes the notion of a field extension quite well. E is an extension of F if $E \supseteq F$. We could derive the extension relationship from the input concepts E and F by "taking everything additional from E and adding it to F." This is made specific in the process of *adjoining* elements to a field, which simply means to augment the field with all formal finite sums and products of the adjoined elements with coefficients in the base field.

Second, the notion of the *splitting field* of a polynomial, namely the special extension K/F containing all of the roots of $f(x)$. This could be formed conceptually by combining the concept "*the roots of a polynomial f with coefficients in a field F*" and the concept "*a field extension E/F formed by adjoining certain elements to F*." Formally, the roots of f are not part of in the second concept, and they must be put in correspondence with the "certain elements." Note that formulating the concept of a splitting field in this way is different from proving that a splitting field always exists. It does, however, and the proof (by induction on the degree of the polynomial) works by successively adjoining elements to F. This gives an inkling of the idea that blending could be used as a proof step.

As above, we could then form the concept of $\mathbf{Gal}(K)$ by blending at the conceptual level. This time, there would be several constituent pieces: "*the roots of a polynomial f with coefficients in a field F*," "*the splitting field of f*," "*the group of automorphisms of a field extension E*," "*the automorphisms that fix F*."

Finally, assuming that we have built $\mathbf{Gal}(K)$ in this fashion, we would like to know some of its properties. Consider the claim that *elements of* $\mathbf{Gal}(K)$ *permute the roots of* f. This time, instead of being purely conceptual, we want to work at the *process* level, and consider before-and-after descriptions of the result of applying $\varphi \in \mathbf{Gal}(K)$ to some r with the property $f(r) = 0$. This is similar in some ways to the "Riddle of the Buddhist Monk" **FaTu98b** which is cited as an example of the power of blending.[1] However, this time the generic space is not a simple geometric machine, but rather an algebraic machine with several moving parts.

The proof of the claim is as follows. If $f(r) = 0$, then $\varphi f(r) = \varphi 0$. Since φ is an automorphism, $\varphi 0 = 0$; and furthermore φ distributes over the sums and products that make up the polynomial f and fixes its coefficients, therefore $\varphi f(r) = f(\varphi r)$. Chaining the equalities together, we have $f(\varphi r) = 0$.

In short, the proof is a fairly direct result of combining "what it means to be a root," "what it means to be an automorphism," "what it means to say that the automorphism fixes elements of F," and "what it means to be a polynomial with coefficients in the field F." Indeed, the proof is in some sense the only thing it could be if one knows the definitions.

Go99c suggests that "combination is colimit." Can we realise the proof through (one or several) colimit operations? I.e. is the proof a blend? And is there anything special about this proof?

Issues raised

There are various technical questions **(@Alan, @Ewen, @Felix)**, but from a naive mathematical standpoint the first issue is: is it always clear how to combine the relevant facts? And a related question: is it always clear what the relevant facts actually are?

If blending is the realisation of "combinatorial creativity" why are we not swamped by the combinatorial explosion of possible things to combine?

[1] Fauconnier and Turner cite Arthur Koestler: "A Buddhist monk begins at dawn one day walking up a mountain, reaches the top at sunset, meditates at the top for several days until one dawn when he begins to walk back to the foot of the mountain, which he reaches at sunset. Making no assumptions about his starting or stopping or about his pace during the trips, prove that there is a place on the path which he occupies at the same hour of the day on the two separate journeys."

Evaluation and Outlook

Joe et al.

Conclusions

(and references) – everyone!