数論における Julia の援用

IMI 共同利用「数学と物理における Julia の活用」

横山 俊一 (東京都立大学)
s-yokoyama@tmu.ac.jp

2023年7月11日　※ 本日38歳になりました

内容の一部は JSPS 科研費・基盤(C) 20K03537
Julia 言語を用いた新しい計算機数論システムの開発とその応用
(代表：横山 俊一) の援助を受けています

科研費
KAKENHI

代数系（数論）における Julia の立ち位置

- 幾何系や解析系に比べると 活用状況・認知度ともに低い 状況
- 既存の CAS 開発においては、1つの分野（代数的対象物）に特化 したものが多く、長期間使い込んだユーザが異なるプラットフォームに移行するのは困難
  例： Magma（C, 次頁）、Sage（Python, 次々頁）、Pari/GP（C）etc.
- そもそも Julia 言語で書かれたパッケージはまだ少数
- それでも開発速度は徐々に上がってきている：
  例： AbstractAlgebra.jl, Gap.jl, GroebnerBasis.jl etc.
- 正しく実装すれば、多項式代数においても高速

Magma

- 豪 シドニー大学 を中心として開発運営が進められている
  計算代数システム。前身は Cayley (1982) で、
  正式ローンチは 1993 年
  有償だがプロプライエタリライセンス（非営利）

- 主として 数論および計算機代数に特化 した関数群
  圏論 に基づいた設計思想

- 2013年から Intel AVX 演算に対応（Linux 64bit OS のみ）

- Bosma-Cannon-Playoust, The Magma algebra system I: the user language (CANT/London, 1997)

Sage/CoCalc/SageMathCell

- 米 ワシントン大学 を中心として開発運営が
  進められて「いた」汎用数式処理システム

- 数論幾何学の研究者 William Stein が開発責任者
  1990年代には Magma の HECKE パッケージ（モジュラー形式の計算）の
  開発に従事、2005年の正式ローンチの際に Python へ移植

- web ベースの UI 開発の変遷：
  Sage Notebook -> salv.us -> SageMathCloud -> CoCalc
  SageMathCell は CoCalc の簡易版

- 2019年に大学を退職し、SageMath Inc. CEO/ファウンダーとして運営

Nemo

- 独 カイザースラウテルン工科大学 を中心として開発運営が進められている、Julia で開発された数論計算パッケージおよび C/C++ wrapper 集

- Singular の開発本拠地でもある（→ Singular.jl pkg ）

- 中心メンバ： Claus Fieker, William Hart, Tommy Hofmann, Frederik Johansson, Marek Kaluba, Carlo Sircana

- FHHJ, Nemo/Hecke: Computer Algebra and Number Theory Packages for the Julia Programming Language, ISSAC’17.

Nemo

- 2018年、Nemo は2つのパッケージに分離：
  Nemo.jl　Flint, Arb, Antic 限定 wrapper
  AbstractAlgebra.jl　上記外の generic なパッケージ

- Julia 単独でも
  - Z, Q, R, C（Flint/Arb）
  - 行列計算, FFT, 並列計算　etc.

  などが扱えるが、これに加えて整数論に特化した機能をもつ：
  - 代数体（Antic）, 有限体（Flint）
  - 1変数／多変数多項式環（generic）
  - p進体・関数体（Flint, finite precision）

Nemo

- Flint を用いることなく generic に生成されたもの
  (例：係数環 R) は AbstractAlgebra.jl に格納

- Flint や Arb 単独で高速処理できるものは、すべて
  Julia 言語で実行するだけ

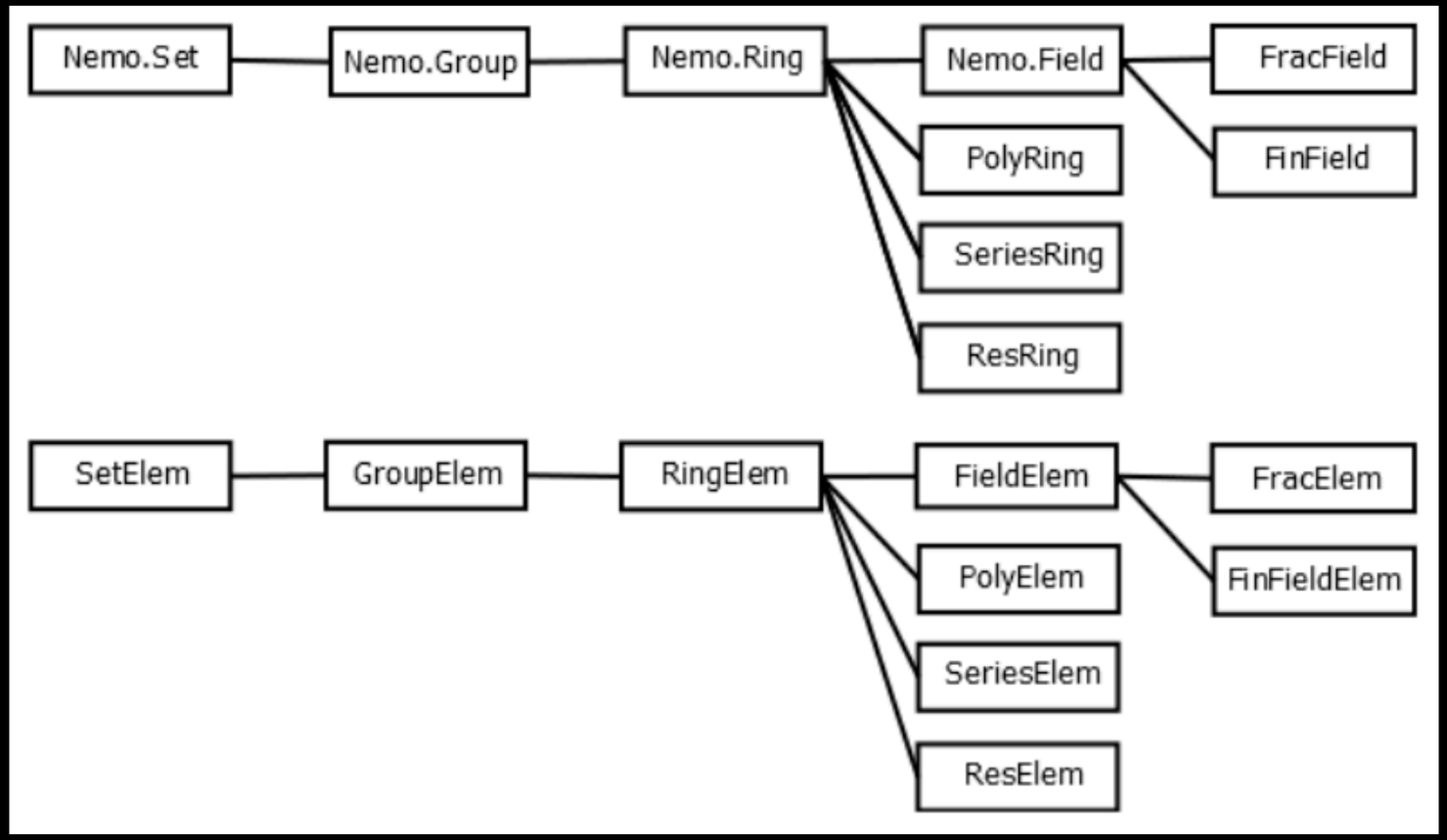
- より高速な実装を目指すならば、まず AbstractAlgebra.jl のリストを確認

- 例： S = Q_p（p進数体, precision m）

```
function O(R::FlintPadicField, m::fmpz)
   if isone(m)
      N = 0
   else
      p = prime(R)
      if m == p
         N = 1
      else
         N = flog(m, p)
         p^(N) != m && error("Not a power of p in p-adic O()")
      end
   end
   d = padic(N)
   d.parent = R
   return d
end
```

m=p^r 以外の prec が代入されるとエラーを返す

Abstract-type hierarchy



- 例えば xxx.Field は xxx.Ring に属する
  - Magma では異なる category として処理
  - Sage では構造が 逆： xxx.ring は xxx.Field の部分集合

Hecke

- Nemo（と AbstractAlgebra パッケージ）に準じて開発されている整数論 に特化した計算パッケージ
- 1990 年代に William Stein（Sage/CoCalc 開発者）が Magma で開発していた、モジュラー形式に関するパッケージ HECKE とは別物
- https://github.com/thofma/Hecke.jl（ソースコード）
- https://thofma.github.io/Hecke.jl/latest/（ドキュメント）

```
(@v1.5) pkg> add Primes
    Updating registry at `~/.julia/registries/General`
    Updating git-repo `https://github.com/JuliaRegistries/General.git`
   Resolving package versions...
Updating `~/.julia/environments/v1.5/Project.toml`
  [27ebfcd6] + Primes v0.5.0
Updating `~/.julia/environments/v1.5/Manifest.toml`
  [27ebfcd6] + Primes v0.5.0

julia> using Primes
[ Info: Precompiling Primes [27ebfcd6-29c5-5fa9-bf4b-fb8fc14df3ae]

julia> factor(20200811)
17^2 * 69899

julia> @time factor(37837492843298492383902389201839237982374892374)
  0.000457 seconds (49 allocations: 3.492 KiB)
2 * 13 * 14552881862807112455347072769938168454759573999
```

```
julia> @time factor(155091189582461323967422273374060483191006034694003396144697
710870373304529655232110787300599828902108544248049381377511219702428693927879643
3018958702811151575938519984102178816)
^CERROR: InterruptException:
Stacktrace:
 [1] BigInt at ./gmp.jl:56 [inlined]
 [2] sub(::BigInt, ::BigInt) at ./gmp.jl:158
 [3] - at ./gmp.jl:476 [inlined]
 [4] pollardfactors!(::BigInt, ::Primes.Factorization{BigInt}) at /Users/s-yokoy
ama_lab/.julia/packages/Primes/eRBAQ/src/Primes.jl:415
 [5] pollardfactors!(::BigInt, ::Primes.Factorization{BigInt}) at /Users/s-yokoy
ama_lab/.julia/packages/Primes/eRBAQ/src/Primes.jl:431 (repeats 3 times)
 [6] factor!(::BigInt, ::Primes.Factorization{BigInt}) at /Users/s-yokoyama_lab/
.julia/packages/Primes/eRBAQ/src/Primes.jl:271
 [7] factor at /Users/s-yokoyama_lab/.julia/packages/Primes/eRBAQ/src/Primes.jl:
304 [inlined]
 [8] top-level scope at ./timing.jl:174 [inlined]
 [9] top-level scope at ./REPL[9]:0
```

Magma の Pollard's Rho + ECM

```
> Factorization(155091189582461323967422273374060483191006034694003396144697710\
08703733045296552321107873005998289021085442480493813775112197024286939278796430\
3018958702811151575938519984102178816);
Integer main factorization (primality of factors will be proved)
Effort: 3
Seed: 260009057 0
    Number: 155091189582461323967422273374060483191006034694003396144697710870\
    373304529655232110787300599828902108544248049381377511219702428693927879643\
    018958702811151575938519984102178816

Pollard Rho
    Trials: 8191
    Number: 231661664844289682481058010180425143978317160529552433050689942523\
    3740229588704158097520762177388796152 51
    (105 digits)
    Factor: 21727894856911 (14 digits)
    Cofactor: 106619470671179617891148702910743456307136150318626307032909495\
    93900960465823539183339042941 (92 digits)
    Time: 0.009
```

```
Pollard Rho
    Trials: 8191
    Number: 106619470671179617891148702910743456307136150318626307032909495939009\
    096046582353918339042941
    (92 digits)
    No factor found
    Time: 0.000

1 composite number remaining

ECM
    x: 106619470671179617891148702910743456307136150318626307032909495939009604\
    65823539183339042941
    (92 digits)
    Initial B1: 5000, limit: 858248
    Initial Pollard p - 1, B1: 45000
    Step 1; B1: 5000 [858248], digits: 92, elapsed time: 0.020
    Factor: 146234082633259 (15 digits)
    Cofactor: 729101374667703049351065352752289046328983288105030665770554647881\
    70201591799 (77 digits)
    Total ECM time: 0.109
```

```
ECM
    x: 10846044215834722665042065816010288453465430807005605353874509
    (62 digits)
    Initial B1: 6341, limit: 17552
    Step 1; B1: 6341 [17552], digits: 62, elapsed time: 0.000
    Step 10; B1: 7072 [17552], digits: 62, elapsed time: 0.160
    Step 20; B1: 7932 [17552], digits: 62, elapsed time: 0.360
    Step 30; B1: 8842 [17552], digits: 62, elapsed time: 0.580
    Step 40; B1: 9802 [17552], digits: 62, elapsed time: 0.820
    Step 50; B1: 10809 [17552], digits: 62, elapsed time: 1.100
    Step 60; B1: 11864 [17552], digits: 62, elapsed time: 1.399
    Step 70; B1: 12969 [17552], digits: 62, elapsed time: 1.710
    Step 80; B1: 14124 [17552], digits: 62, elapsed time: 2.060
    Step 90; B1: 15329 [17552], digits: 62, elapsed time: 2.430
    Factor: 75045259055838983 (17 digits)
    Cofactor: 144526707646708212356380247022426585248301323 (45 digits)
    Total ECM time: 2.740

Total time: 3.139

[ <2, 206>, <67, 2>, <271, 1>, <5351, 1>, <3726911, 1>, <5830001, 1>,
<146234082633259, 1>, <6722279202985811, 1>, <75045259055838983, 1>,
<144526707646708212356380247022426585248301323, 1> ]
```

Nemo の Pollard's Rho + ECM

```
julia> using Nemo

Welcome to Nemo version 0.17.6

Nemo comes with absolutely no warranty whatsoever


julia> factor(ZZ(20200811))
1 * 69899 * 17^2

julia> @time factor(ZZ(155091189582461323967422273374060483191006034694003396144
697710870373304529655232110787300599828902108544248049381377511219702428693927879643018958702811151575938519984102178816))
  0.941467 seconds (97.52 k allocations: 9.656 MiB, 1.21% gc time)
1 * 3726911 * 5830001 * 67^2 * 2^206 * 672279202985811 * 5351 * 146234082633259 * 75045259055838983 * 144526707646708212356380247022426585248301323 * 271
```

注： Magma ではさらに MPQS が使える

```
julia> Qx,x=PolynomialRing(FlintQQ,"x");

julia> K1,a=NumberField(x^2-2,"a")
(Number field over Rational Field with defining polynomial x^2-2, a)

julia> K2,b=NumberField(x^2-3,"b")
(Number field over Rational Field with defining polynomial x^2-3, b)

julia> isisomorphic(K1,K2)
(false, Injection of Number field over Rational Field with defining polynomial x
^2-2 into Number field over Rational Field with defining polynomial x^2-3
defined by a -> 0
)

julia> K,g=number_field([x^2-2,x^2-3])
(Non-simple number field with defining polynomials fmpq_mpoly[x1^2 - 2, x2^2 - 3
], NfAbsNSElem[_$1, _$2])

julia> g[2]^2
3

julia> minpoly(g[1]+g[2])
x^4-10*x^2+1
```

```
julia> Qx,x=PolynomialRing(FlintQQ,"x");

julia> f=x^2-10;

julia> k,a=NumberField(f,"a");

julia> @time Ok=maximal_order(k)
 13.223396 seconds (27.29 M allocations: 1.298 GiB, 4.86% gc time)
Maximal order of Number field over Rational Field with defining polynomial x^2-1
0
with basis nf_elem[1, a]

julia> g=x^3-14;

julia> l,b=NumberField(g,"b");

julia> @time Ol=maximal_order(l)
  0.000176 seconds (585 allocations: 29.828 KiB)
Maximal order of Number field over Rational Field with defining polynomial x^3-1
4
with basis nf_elem[1, b, b^2]
```

```
julia> c,mc=class_group(k)
(GrpAb: Z/2, ClassGroup map of
Set of ideals of OK
)

julia> A=ray_class_field(mc)
Class field defined mod (<1, 1>, InfPlc[]) of structure c

julia> K=number_field(A)
non-simple Relative number field over
Number field over Rational Field with defining polynomial x^2-10
 with defining polynomials AbstractAlgebra.Generic.MPoly{nf_elem}[_$1^2-2]
```

```
julia> @time ZK=maximal_order(K)
  0.000007 seconds (1 allocation: 48 bytes)
Relative maximal order of non-simple Relative number field over
Number field over Rational Field with defining polynomial x^2-10
 with defining polynomials AbstractAlgebra.Generic.MPoly{nf_elem}[_$1^2-2]
with pseudo-basis
(1, 1//1 * <1, 1>)
(_$1+a, 1//4 * <2, a>)

julia> discriminant(ZK)
<1, 1>
Norm: 1
Minimum: 1
two normal wrt: 2
```

Magma's NF-CG-RCF-NF

```
> Qx<x>:=PolynomialRing(Rationals());
> k<a>:=NumberField(x^2-10);
> c,mc:=ClassGroup(k); c; mc;
Abelian Group isomorphic to Z/2
Defined on 1 generator
Relations:
    2*c.1 = 0
Mapping from: GrpAb: c to Set of ideals of Maximal Equation Order with defining
polynomial x^2 - 10 over its ground order
> A:=RayClassField(mc); A;
FldAb, defined by (<[1, 0]>, [])
of structure: Z/2
> K:=NumberField(A); K;
Number Field with defining polynomial $.1^2 - 2 over k
> time ZK:=MaximalOrder(K);
Time: 0.020
> ZK;
Maximal Order of Equation Order with defining polynomial x^2 + [-2, 0] over its
ground order
> Discriminant(ZK);
Principal Ideal
Generator:
    [1, 0]
```

ベンチマーク例 (from NemoCas page)

Fateman benchmark

- f = 1 + x + y + z + t
- p = f^30
- time q = p*(p + 1)

This benchmark tests generic polynomial arithmetic. Note that we don't have a dedicated multivariate polynomial module in Nemo yet, so this and the next benchmark are done with nested univariate polynomial rings. Where possible we do the same thing in the other systems we time (Pari/GP is an exception).

SageMath 6.8	Pari/GP 2.7.4	Magma V2.21-4	Nemo-0.3
132s	156s	233s	44s

ベンチマーク例 (from NemoCas page)

Magma:

```
> R<x>:=PolynomialRing(Integers());
> S<y>:=PolynomialRing(R);
> T<z>:=PolynomialRing(S);
> U<t>:=PolynomialRing(T);
> time f:=(x+y+z+t+1)^30;
> time g:=f*(f+1);
```

Nemo:

```
> R,x=PolynomialRing(FlintZZ,"x");
> S,y=PolynomialRing(R,"y");
> T,z=PolynomialRing(S,"z");
> U,t=PolynomialRing(T,"t");
> @time f=(x+y+z+t+1)^30;
> @time g=f*(f+1);
```

実はほぼ同一コード

ベンチマーク例 (from NemoCas page)

Pearce benchmark

- f = (x + y + 2z^2 + 3t^3 + 5u^5 + 1)^16
- g = (u + t + 2z^2 + 3y^3 + 5x^5 + 1)^16
- time q = f*g

This benchmark is usually done with sparse multivariate polynomial arithmetic. We use it here with our recursive dense polynomial arithmetic (there is no sparse polynomial representation in Nemo yet).

Note: this is not fair to other packages that implement sparse multivariate polynomial arithmetic, as they could complete this benchmark faster if we used their sparse arithmetic. But that would not be a very meaningful benchmark if the aim is to measure the relative performance of the recursive dense polynomial arithmetic in Nemo. On the other hand, Pari/GP uses a sparse-recursive representation (similar to recursive-dense, but with dedicated zero objects in each ring).

SageMath 6.8	Pari/GP 2.7.4	Magma V2.21-4	Nemo-0.3
2900s	798s	647s	167s

ベンチマーク例 (from NemoCas page)

Resultant benchmark

- R = GF(17^11)
- S = R[y]
- T = S/(y^3 + 3x*y + 1)
- U = T[z]
- f = T(3y^2 + y + x)*z^2 + T((x + 2)*y^2 + x + 1)*z + T(4x*y + 3)
- g = T(7y^2 – y + 2x + 7)*z^2 + T(3y^2 + 4x + 1)*z + T((2x + 1)*y + 1)
- s = f^12
- t = (s + g)^12
- time r = resultant(s, t)

This benchmark is designed to test generics, division and computation of the resultant.

SageMath 6.8	Pari/GP 2.7.4	Magma V2.21-4	Nemo-0.3
179907s	N/A	82s	0.2s

ベンチマーク例 (from NemoCas page)

Polynomials over number fields

- R, x = CyclotomicField(20)
- S = R[y]
- f = (3x^7 + x^4 – 3x + 1)*y^3 + (2x^6-x^5+4x^4-x^3+x^2-1)*y +(-3x^7+2x^6-x^5+3x^3-2x^2+x)
- time r = f^300

This benchmark is designed to test dense polynomials over number fields.

SageMath 6.8	Pari/GP 2.7.4	Magma V2.21-4	Nemo-0.3
6.92s	0.21s	0.70s	0.13s

NemoCas Project

- http://nemocas.org/（Nemo/Hecke 開発ページ）
- Magma, Sage, Pari/GP と比較したいくつかのベンチマーク結果が閲覧可能
- 最新版は開発者用 Git リポジトリから
  Nemo: https://github.com/Nemocas/Nemo.jl
  AbstractAlgebra: https://github.com/Nemocas/AbstractAlgebra.jl
  Hecke: https://github.com/thofma/Hecke.jl
  Singular: https://github.com/oscar-system/Singular.jl/
- 試しに使ってみたいが導入に失敗した方へ：
  Nemo は Jupyter Notebook 経由で Sage 上で利用可能

OSCAR

OSCAR
SYMBOLIC TOOLS

The OSCAR project

Home
About
News
Installation
Tutorials
Examples
Documentation
Community
Talks
Meetings
Credits
Contributors

OSCAR Computer Algebra System

The OSCAR project develops a comprehensive **O**pen **S**ource **C**omputer **A**lgebra **R**esearch system for computations in algebra, geometry, and number theory, written in Julia. In particular, the emphasis is on supporting complex computations which require a high level of integration of tools from different mathematical areas.

The project builds on and extends the four cornerstone systems

- GAP - computational discrete algebra (via GAP.jl Julia package)
- Singular - commutative and non-commutative algebra, algebraic geometry (via Singular.jl Julia package)
- Polymake - polyhedral geometry (via Polymake.jl Julia package)
- Antic (Hecke, Nemo) - number theory

as well as further libraries and packages, which are tied together using the Julia language into the Oscar.jl Julia package.

The development of OSCAR is supported by the Deutsche Forschungsgemeinschaft DFG within the Collaborative Research Center TRR 195.

OSCAR

- 主として NemoCas プロジェクトから派生した、
  Julia を用いた新計算代数システム開発プロジェクト

- GAP（Gap.jl）, Singular（Singular.jl）, Polymake（Polymake.jl）,
  Antic（Nemo.jl / Hecke.jl）の4ライブラリを主軸に構成

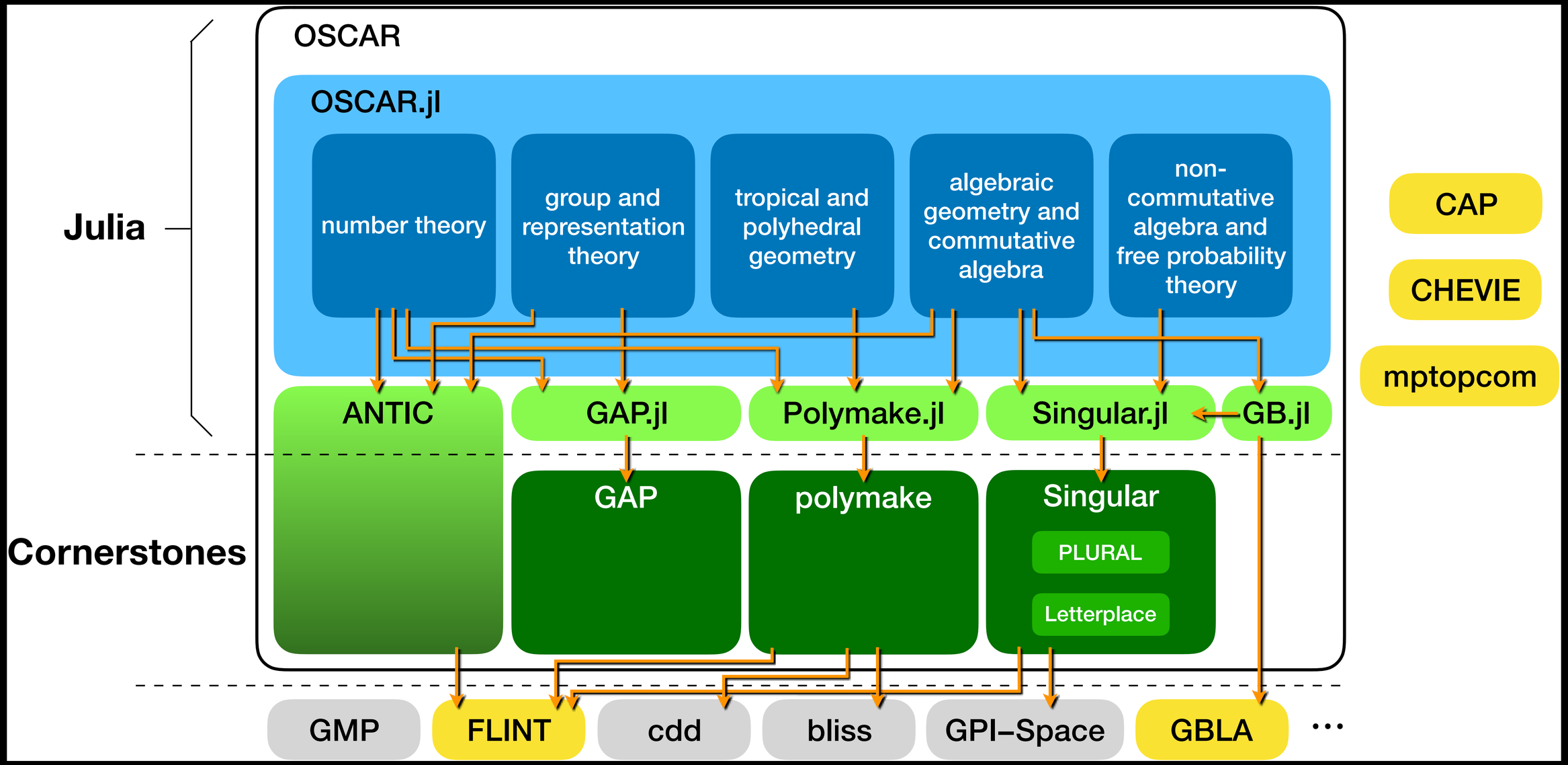
- プロジェクトリーダ：
  - Wolfram Decker（TU Kaiserslautern：NemoCas メンバ）
  - Claus Fieker（TU Kaiserslautern：NemoCas メンバ）
  - Max Horn（TU Kaiserslautern）
  - Michael Joswig（TU Berlin）

- DFG, TUK に研究所 TRR-195 を2017年に開設

OSCAR

```
  Installed Oscar ──────────────────────── v0.12.1
 Downloaded artifact: libsingular_julia
 Downloaded artifact: Antic
 Downloaded artifact: Ncurses
 Downloaded artifact: MongoC
 Downloaded artifact: PPL
 Downloaded artifact: GLPK
 Downloaded artifact: Zstd
 Downloaded artifact: Perl
 Downloaded artifact: MUMPS_seq
 Downloaded artifact: SPRAL
 Downloaded artifact: FLINT
 Downloaded artifact: OpenSSL
 Downloaded artifact: bliss
 Downloaded artifact: Hwloc
 Downloaded artifact: AtkinModularPolynomialsDB
 Downloaded artifact: ZLatDB
 Downloaded artifact: ClassicalModularPolynomialsDB
 Downloaded artifact: QuadLatDB
 Downloaded artifact: HermLatDB
 Downloaded artifact: SmallGroupDB
 Downloaded artifact: Bzip2
Downloading artifact: polymake
   Downloading [=================================>     ]  85.3 %
```

OSCAR


Julia
Cornerstones
OSCAR
OSCAR.jl
number theory
group and representation theory
tropical and polyhedral geometry
algebraic geometry and commutative algebra
non-commutative algebra and free probability theory
ANTIC
GAP.jl
Polymake.jl
Singular.jl
GB.jl
GAP
polymake
Singular
PLURAL
Letterplace
GMP
FLINT
cdd
bliss
GPI–Space
GBLA
...
CAP
CHEVIE
mptopcom

OSCAR

- 計算代数／数論における連携処理をシームレスに行う Oscar.jl を提供

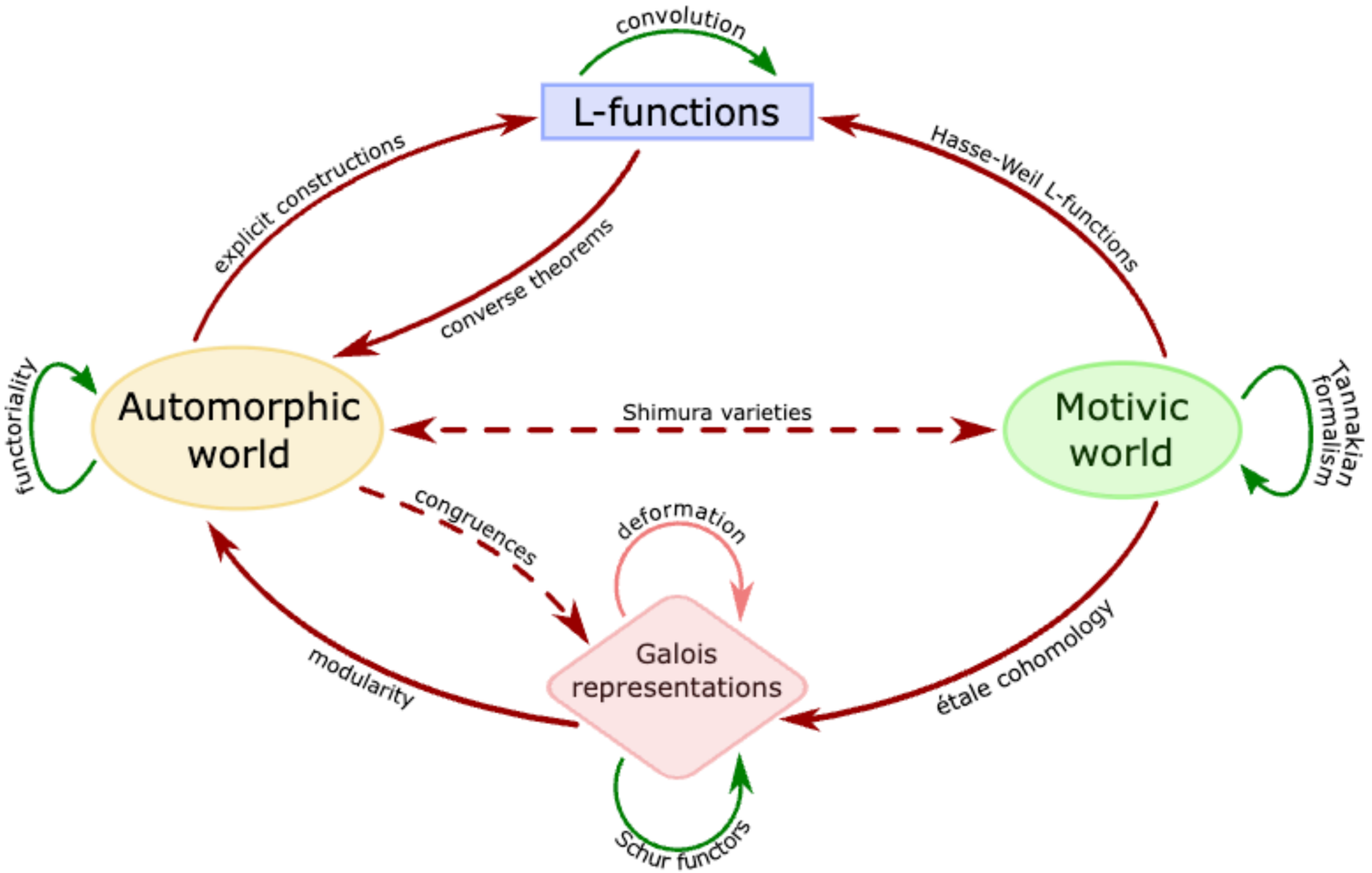
```
Precompiling project...
  74 dependencies successfully precompiled in 118 seconds. 8 already precompiled
.

julia> using Oscar
 -----    -----    -----       _      -----
|     |  |     |  |     |    | |    |     |
|     |  |        |         |   |   |     |
|     |   -----   |        |     |  |-----
|     |         |  |        |-----|  |   |
|     |  |     |  |     |  |     |  |    |
 -----    -----    -----   -     -  -     -

...combining (and extending) ANTIC, GAP, Polymake and Singular
Version 0.12.1 ...
 ... which comes with absolutely no warranty whatsoever
Type: '?Oscar' for more information
(c) 2019-2023 by The OSCAR Development Team

julia>
```

- 数論統合システム LMFDB 上のデモプログラムに OSCAR コードが新規採用

LMFDB

The LMFDB universe

The top half of the diagram is based on the Langlands program, which predicts that any motivic object corresponds to an automorphic object via their L-functions.



LMFDB

Elliptic curve with LMFDB label 100.a2 (Cremona label 100a4)

Minimal Weierstrass equation

Show commands: Magma / Oscar / PariGP / SageMath

$$y^2 = x^3 - x^2 - 908x - 15688$$ (homogenize, simplify)

```
oscar: E = EllipticCurve([0, -1, 0, -908, -15688])
```

```
oscar: short_weierstrass_model(E)
```

Mordell-Weil group structure

$\mathbb{Z}/2\mathbb{Z}$

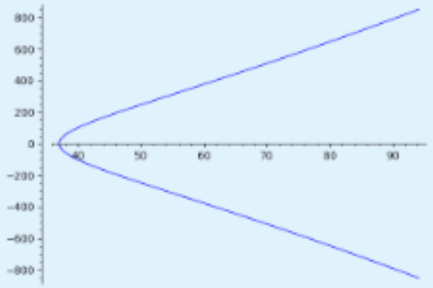
Torsion generators

$(37, 0)$

```
oscar: torsion_structure(E)
```

Properties

Label	100.a2



Conductor	100
Discriminant	-62500000000
j-invariant	$-\frac{20720464}{15625}$
CM	no
Rank	0
Torsion structure	$\mathbb{Z}/2\mathbb{Z}$

Related objects

OSCAR

- すべての情報は OSCAR の web ページから参照：
  https://www.oscar-system.org/

- インストールが少し難しいので注意：
  - Julia に加えて Xcode ツール が必要（Mac OS の場合）
  - Julia を立ち上げてから Pkg ツール を用いる
  - インストール・ダウンロード (アーティファクト)・プリコンパイルに
    少々時間がかかるので気長に待つ

- 終わったら using Oscar で、あとは楽しむ
  ただしパス設定がシビアなので、慣れるまで頑張る
  ※ Jupyter Notebook 環境を使うと少しは導入がラク