What's new in Flint (2.6)

Fredrik Johansson[1]
Inria Bordeaux

International Congress on Mathematical Software (ICMS) 2020

[1] Credits to William Hart for most of the slides

Introduction to Flint – http://flintlib.org

- C library for number theory
- Used by Singular, Hecke, Nemo, Sage, Macaulay 2, Arb, Calcium, Antic and others
- Basic rings: $\mathbb{Z}$, $\mathbb{Q}$, $\mathbb{Z}/n\mathbb{Z}$, $\mathbb{F}_q$, $\mathbb{Q}_p$
- Univariate polynomials over the above rings
- Multivariate polynomials over the above except p-adics
- Linear algebra over the above rings
- Primality testing and integer factorisation
- Polynomial factorisation over $\mathbb{Z}$, $\mathbb{Z}/n\mathbb{Z}$, $\mathbb{F}_q$
- LLL, HNF, SNF

Using Flint in Julia

julialang.org/downloads/

Current stable release: v1.4.2 (May 23, 2020)

Checksums for this release are available in both MD5 and SHA256 formats.

Windows (.exe) [help]	32-bit	64-bit
macOS 10.8+ (.dmg) [help]		64-bit
Generic Linux Binaries for x86 [help]	32-bit (GPG)	64-bit (GPG)
Generic Linux Binaries for ARM [help]		64-bit (AArch64) (GPG)
Generic FreeBSD Binaries for x86 [help]		64-bit (GPG)
Source	Tarball (GPG)	Tarball with dependencies (GPG)

```
julia-1.4.2/etc/
julia-1.4.2/etc/julia/
julia-1.4.2/etc/julia/startup.jl
wbhart@MSI:~$
wbhart@MSI:~$ julia-1.4.2/bin/julia
               _
   _       _ _(_)_     |  Documentation: https://docs.julialang.org
  (_)     | (_) (_)    |
   _ _   _| |_  __ _   |  Type "?" for help, "]?" for Pkg help.
  | | | | | | |/ _` |  |
  | | |_| | | | (_| |  |  Version 1.4.2 (2020-05-23)
 _/ |\__'_|_|_|\__'_|  |  Official https://julialang.org/ release
|__/                   |

julia> using Pkg

julia> Pkg.add("Nemo")
```

```
julia> using Pkg

julia> Pkg.add("Nemo")
   Updating registry at `~/.julia/registries/General`
   Updating git-repo `https://github.com/JuliaRegistries/General.gi
  Resolving package versions...
   Updating `~/.julia/environments/v1.4/Project.toml`
 [no changes]
   Updating `~/.julia/environments/v1.4/Manifest.toml`
 [no changes]

julia> using Nemo

Welcome to Nemo version 0.17.5

Nemo comes with absolutely no warranty whatsoever
```

```
julia> ZZ(12)
12

julia> QQ(12)
12

julia> ResidueRing(ZZ, 7)
Integers modulo 7

julia> FiniteField(7, 2, "a")
(Finite field of degree 2 over F_7, a)

julia> PadicField(7, 20)
Field of 7-adic numbers
```

```
julia> R, x = PolynomialRing(ZZ, "x")
(Univariate Polynomial Ring in x over Integer Ring, x)

julia> R, (s, t, u) = PolynomialRing(ZZ, ["s", "t", "u"])
(Multivariate Polynomial Ring in s, t, u over Integer Ring,
fmpz_mpoly[s, t, u])

julia> matrix(ZZ, 2, 2, [1, 2, 3, 4])
[1  2]
[3  4]

julia> S, y = PuiseuxSeriesRing(ZZ, 10, "y")
(Puiseux series ring in y over Integer Ring, y+O(y^11))
```

```
julia> f = x^3 + 2x + 1
x^3+2*x+1

julia> f^10
x^30+20*x^28+10*x^27+180*x^26+180*x^25+1005*x^24+1440*x^23+
080*x^22+6840*x^21+13104*x^20+21840*x^19+33810*x^18+50400*x
17+68280*x^16+87612*x^15+104760*x^14+115800*x^13+119570*x^1
+113760*x^11+99184*x^10+79160*x^9+56880*x^8+36240*x^7+20205
x^6+9504*x^5+3540*x^4+970*x^3+180*x^2+20*x+1

julia> factor(f^10)
1 * (x^3+2*x+1)^10
```

```
julia> R, x = PolynomialRing(ZZ, "x")
(Univariate Polynomial Ring in x over Integer Ring, x)

julia> T = MatrixSpace(ZZ, 80, 80)
Matrix Space of 80 rows and 80 columns over Integer Ring

julia> M = rand(T, -20:20);

julia> @time p = charpoly(R, M);
  0.017597 seconds (259 allocations: 1.105 MiB)

julia> @time q = minpoly(R, M);
  0.027888 seconds (334 allocations: 2.820 MiB)
```

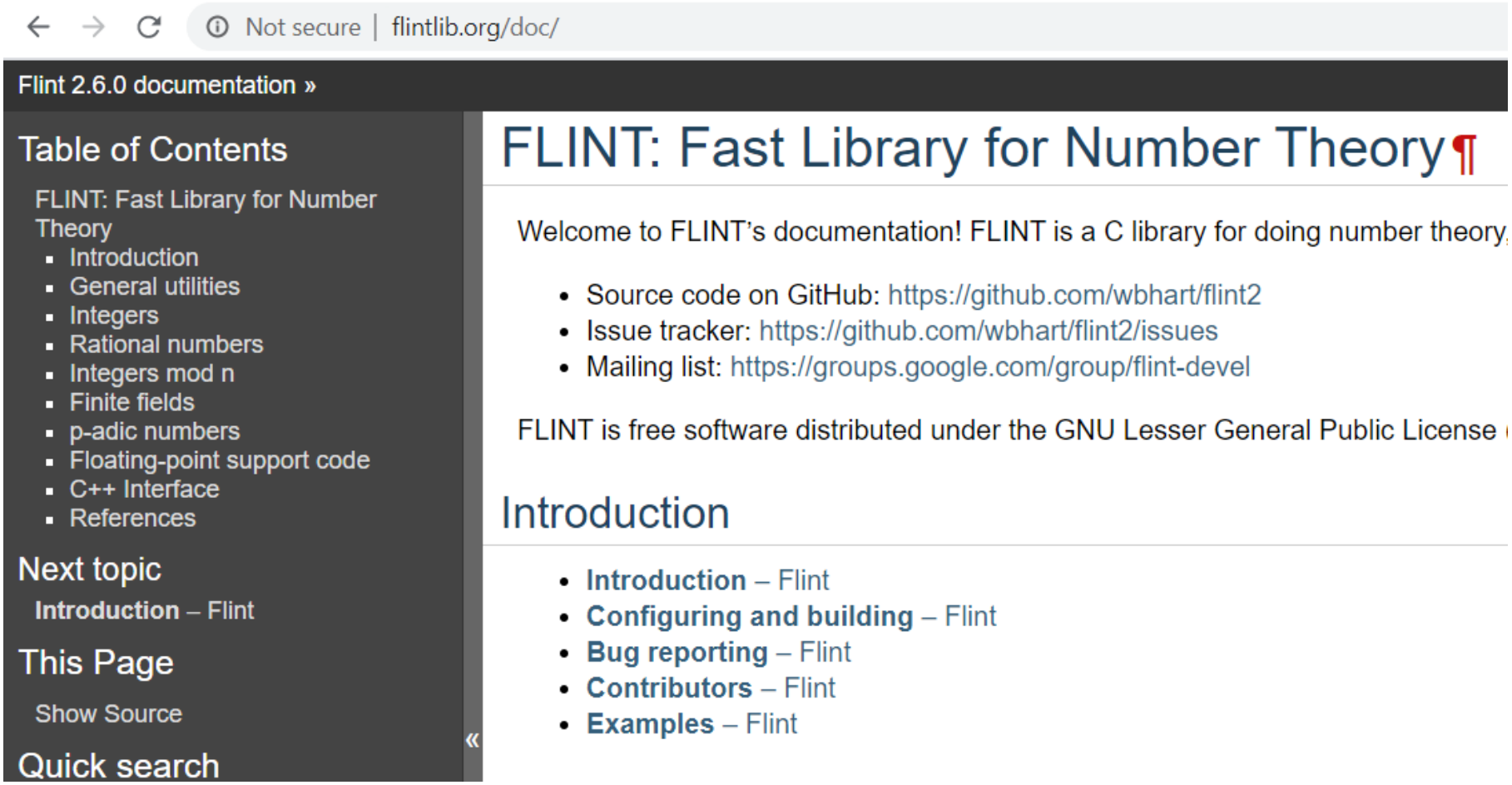
Flint 2.6

- The first release in 5 years
- 300,000 $\rightarrow$ 500,000 lines of code
- Simultaneous development of Nemo
- Over 50 contributors
  - **William Hart** - maintenance, integer factoring, polynomial factoring, FFT, . . .
  - **Daniel Schultz** - multivariate polynomials, thread pool, . . .
  - Fredrik Johansson - faster linear algebra, online docs, . . .
  - Vladimir Glazachev - APRCL
  - Kushagra Singh, Nitin Kumar - qsieve improvements
  - Tommy Hofmann - matrix normal forms
  - Isuru Fernando - testing, build issues, bug fixes
  - Luca De Feo, Edouard Rousseau - Finite field embeddings
  - . . .

New and improved features in Flint 2.6

- Integer factorisation
  - Quadratic sieve integer factorisation
  - Elliptic curve integer factorisation
  - APRCL primality test
- Polynomials
  - Multivariate polynomial arithmetic $\mathbb{Z}[x, y, z, \ldots]$
  - van Hoeij factorisation for $\mathbb{Z}[x]$
  - Parallelised FFT
- Linear algebra
  - Howell form
  - Characteristic and minimal polynomial
  - Speed improvements
- Other
  - Embeddings of finite fields
  - Thread pool
  - Online documentation

Online documentation



Integer factorisation

- `fmpz_factor`, `fmpz_factor_smooth`
- Best of breed algorithm: trial division (with tree reduction), perfect power test, ECM, quadratic sieve
- APRCL primality proving
- Quadratic sieve is parallelised
- About twice as fast as latest stable Pari/GP

Integer factorisation : Quadratic sieve

Table: Quadratic sieve timings

Digits	Pari/GP	Flint (1 core)	Flint (4 cores)
50	0.43	0.55	0.39
59	3.8	3.0	1.7
68	38	21	14
77	257	140	52
83	2200	1500	540

Multivariate polynomials

- Main author: Daniel Schultz
- Over $\mathbb{Z}$, $\mathbb{Q}$, $\mathbb{Z}/n\mathbb{Z}$, $\mathbb{F}_q$ so far...
- Sparse: heap based algorithms (Monagan and Pearce)
- "Quasi-dense": big array method, "dense": FFT
- Parallel multiplication, division, GCD
- Fast assembly for accumulation into 3 limbs
- Pack monomials using Kronecker segmentation
- Support lex, deglex and degrevlex
- Multiprecision exponents
- Factorisation coming soon....

Multivariate multiplication

Table: "Dense" Fateman multiply bench

n	Sage	Singular	Magma	Giac	Trip	Flint
5	0.0063s	0.0048s	0.0018s	0.00023s	0.00057s	0.00023s
10	0.51s	0.11s	0.12s	0.0056s	0.023s	0.0043s
15	9.1s	1.4s	1.9s	0.11s	0.21s	0.045s
20	75s	21s	16s	0.62s	2.3s	0.48s
25	474s	156s	98s	2.8s	12s	2.3s
30	1667s	561s	440s	14s	41s	10s

4 variables

Multivariate multiplication

Table: Sparse multiply benchmark

n	Sage	Singular	Magma	Giac	Trip	Flint
4	0.0066s	0.0050s	0.0062s	0.0046s	0.0015s	0.0014s
6	0.15s	0.11s	0.080s	0.030s	0.016s	0.016s
8	1.6s	0.79s	0.68s	0.28s	0.10s	0.10s
10	8s	3.6s	3.0s	1.5s	0.40s	0.48s
12	43s	14s	11s	4.8s	2.2s	2.0s
14	173s	63s	37s	14s	12s	7.2s
16	605s	201s	94s	39s	39s	19s

5 variables

Multivariate multiplication

Table: Sparse Pearce 4 core

n	Giac	Piranha	Trip	Flint
4	0.0062s	0.0034s	0.0015s	0.0013s
6	0.034s	0.025s	0.016s	0.011s
8	0.31s	0.078s	0.093s	0.047s
10	1.2s	0.23s	0.32s	0.19s
12	3.6s	0.71s	1.2s	0.70s
14	10.5s	2.0s	5.5s	2.5s
16	25s	5.7s	10.3s	6.7s

4 variables

FFT: Integer and polynomial multiplication

Table: FFT timings

Words	1 core	4 cores	8 cores
110k	0.07s	0.05s	0.05s
360k	0.3s	0.1	0.1s
1.3m	1.1s	0.4s	0.3s
4.6m	4.5s	1.5s	1.0s
26m	28s	9s	6s
120m	140s	48s	33s
500m	800s	240s	150s

Thank you!

http://flintlib.org/