Engineering for Scientific Computing

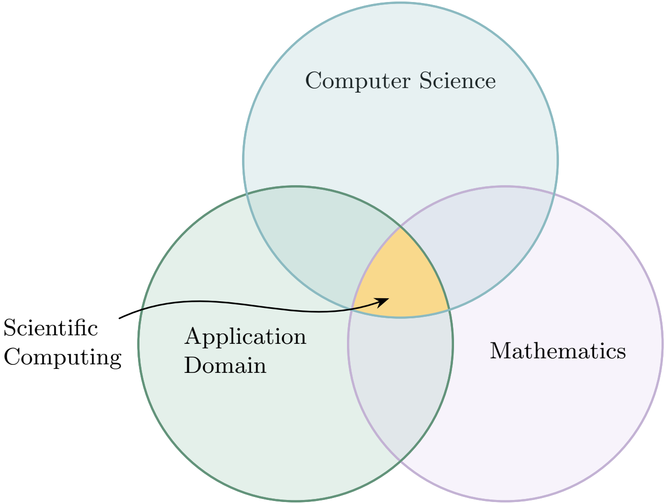
Irregular Parallelism and Dynamically Data-Intensive Computation

Alexander Brandt
University of Western Ontario
Department of Computer Science

February 22, 2023

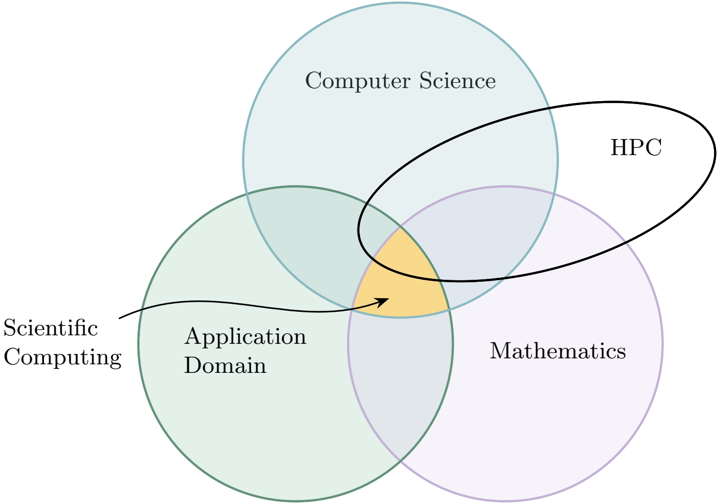
Engineering *for* Scientific Computing

- Scientific computing is interdisciplinary



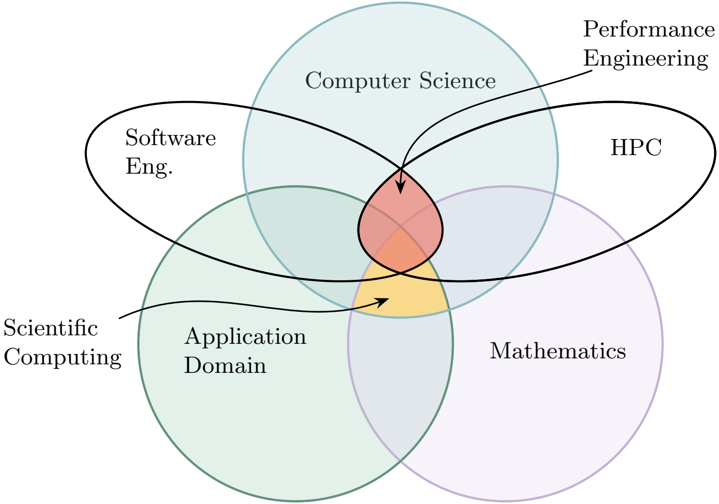
Engineering *for* Scientific Computing

- Scientific computing is interdisciplinary
- High-Performance Computing (HPC) increasingly necessary



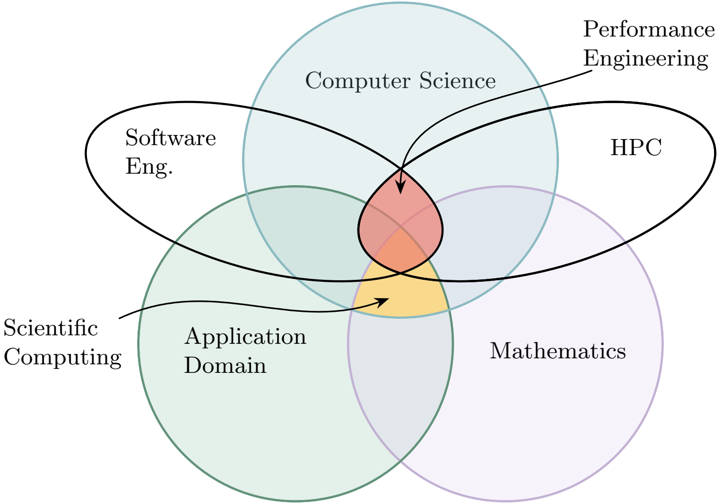
Engineering *for* Scientific Computing

- Scientific computing is interdisciplinary
- High-Performance Computing (HPC) increasingly necessary



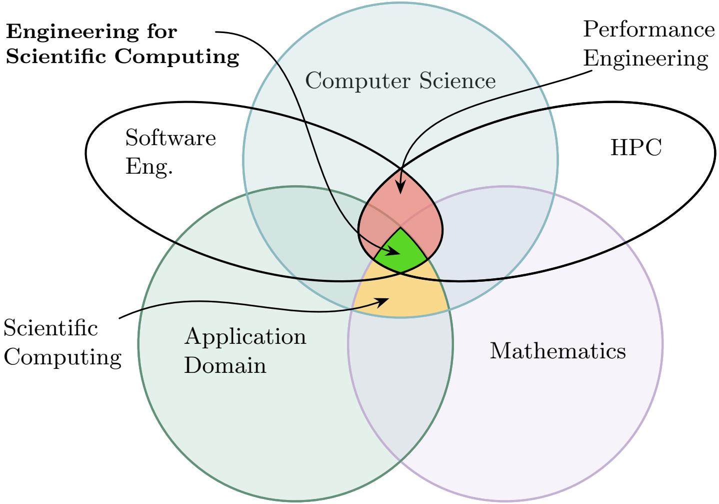
Engineering *for* Scientific Computing

- Scientific computing is interdisciplinary
- High-Performance Computing (HPC) increasingly necessary
- ***Chasm*** between software engineering and scientific computing [15, 19]



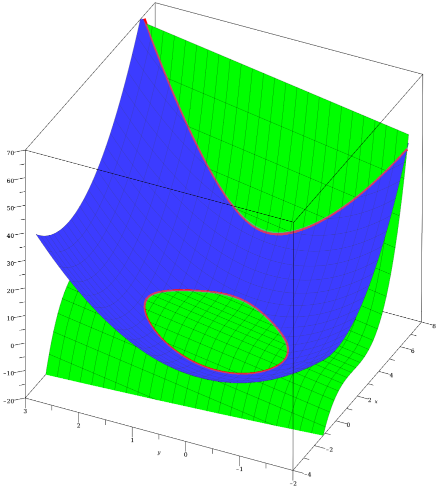
Engineering *for* Scientific Computing

- Scientific computing is interdisciplinary
- High-Performance Computing (HPC) increasingly necessary
- ***Chasm*** between software engineering and scientific computing [15, 19]
- Engineered solutions for performance, productivity, capability, maintainability



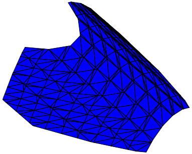
Motivating Application: Solving Systems of Equations

Find x, y, z satisfying $$F = \begin{cases} a(x,y,z) = 0 \\ b(x,y,z) = 0 \\ c(x,y,z) = 0 \end{cases}$$



- A fundamental problem in scientific computing and application domains
- Numerical methods are effective in practice
  - ↳ e.g. Newton's method, Homotopy methods, ...
- **Symbolic Computation** allows computing an **exact** description of **all solutions**
  - ↳ foundational for (elliptic-curve) cryptography, robotics, celestial mechanics, signal processing, ... [14]

Incremental Decomposition of a Non-Linear System

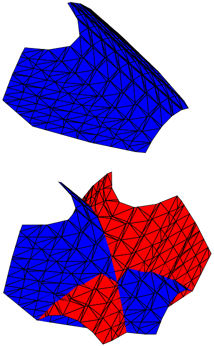


Intersect one equation at a time with the current solution set

$$F = \begin{cases} x^2 + y + z = 1 \\ x + y^2 + z = 1 \\ x + y + z^2 = 1 \end{cases}$$

$$\left\{ x^2 + y + z = 1 \right\}$$

Incremental Decomposition of a Non-Linear System



Intersect one equation at a time with the current solution set

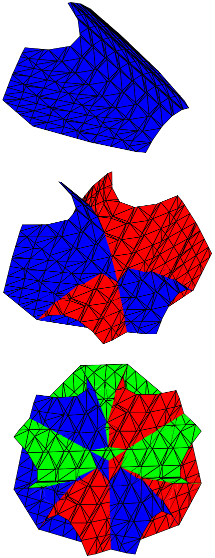
$$F = \begin{cases} x^2 + y + z = 1 \\ x + y^2 + z = 1 \\ x + y + z^2 = 1 \end{cases}$$

$$\left\{ x^2 + y + z = 1 \right\}$$

$$F[2] \qquad \downarrow$$

$$\left\{ \begin{array}{r} x + y^2 + z = 1 \\ y^4 + (2z-2)y^2 + y + (z^2 - z) = 0 \end{array} \right\}$$

Incremental Decomposition of a Non-Linear System



Intersect one equation at a time with the current solution set

$$F = \begin{cases} x^2 + y + z = 1 \\ x + y^2 + z = 1 \\ x + y + z^2 = 1 \end{cases}$$

$$\left\{x^2 + y + z = 1\right\}$$

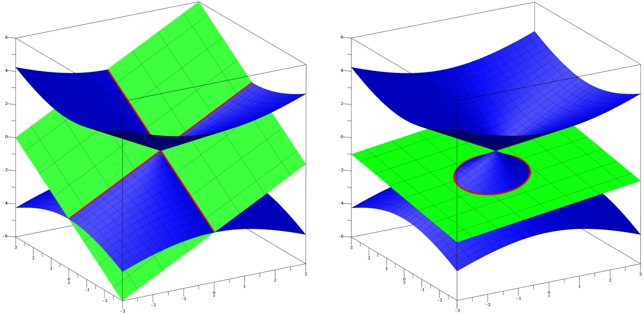
$$F[2] \quad \downarrow$$

$$\left\{ \begin{array}{r} x + y^2 + z = 1 \\ y^4 + (2z-2)y^2 + y + (z^2 - z) = 0 \end{array} \right\}$$

$$F[3] \quad \swarrow \quad \swarrow \quad \searrow \quad \searrow$$

$$\begin{cases} x - z = 0 \\ y - z = 0 \\ z^2 + 2z - 1 = 0 \end{cases}, \quad \begin{cases} x = 0 \\ y = 0 \\ z - 1 = 0 \end{cases}, \quad \begin{cases} x = 0 \\ y - 1 = 0 \\ z = 0 \end{cases}, \quad \begin{cases} x - 1 = 0 \\ y = 0 \\ z = 0 \end{cases}$$

Irregular Parallelism in Solving Systems



Cone: $z^2 = x^2 + y^2$
Plane: $ax + by + cz = d$

- **Irregular Parallelism**: dissimilar tasks with unpredictable dependencies [16]
- An intersection may fall into several different cases depending on particular values of parameters
- An intersection may produce multiple components
- **Discovering** existence of multiple components is **as difficult as solving** those components [A**B**MMX 23]

Expression Swell in Symbolic Computation

$$\{2x^3y^3 - 2x^3 - 4x^2y^3z + 5xy^3z^3 - y^3z^5 = 0\}$$

$$\downarrow \quad F[2] = -2x^3z - 3xy^3z^2 + 2xy^3 + y^3z^4 - 2y^3z^2$$

Expression Swell in Symbolic Computation

$$\{2x^3y^3 - 2x^3 - 4x^2y^3z + 5xy^3z^3 - y^3z^5 = 0\}$$

$$\downarrow \quad F[2] = -2x^3z - 3xy^3z^2 + 2xy^3 + y^3z^4 - 2y^3z^2$$

$$\left\{\begin{array}{r} 9xy^6z^4 - 12xy^6z^2 + 4xy^6 - 26xy^3z^6 + 24xy^3z^5 - 6xy^3z^4 - 16xy^3z^3 + 24xy^3z^2 - 8xy^3 + 21xz^8 + 26xz^6 - 3xz^4 - 12xz^2 + 4x - 3y^6z^6 \\ +8y^6z^4 - 4y^6z^2 + 8y^3z^8 - 8y^3z^7 - 6y^3z^6 + 16y^3z^5 - 16y^3z^4 + 8y^3z^2 - 5z^{10} - 8z^8 + 9z^6 + 8z^4 - 4z^2 = 0 \\ -y^9z^6 - 3y^9z^5 + 6y^9z^4 + 20y^9z^3 - 12y^9z^2 - 12y^9z + 8y^9 + 3y^6z^8 + 6y^6z^7 - 17y^6z^6 - 30y^6z^5 + 42y^6z^4 - 72y^6z^3 + 36y^6z^2 + 24y^6z \\ -24y^6 - 3y^3z^{10} - 7y^3z^9 + 42y^3z^7 + 21y^3z^6 + 33y^3z^5 - 6y^3z^4 + 52y^3z^3 - 36y^3z^2 - 12y^3z + 24y^3 + z^{12} + 3z^{10} - 3z^8 - 11z^6 + 6z^4 + 12z^2 - 8 = 0 \end{array}\right\}$$

$$\downarrow \quad F[3] = -4xz^2 + 4x + 1 + y^2 + z^4 - 2z^2$$

Expression Swell in Symbolic Computation

$$\{2x^3y^3 - 2x^3 - 4x^2y^3z + 5xy^3z^3 - y^3z^5 = 0\}$$

$$\downarrow \quad F[2] = -2x^3z - 3xy^3z^2 + 2xy^3 + y^3z^4 - 2y^3z^2$$

$$\left\{ \begin{array}{r} 9xy^6z^4 - 12xy^6z^2 + 4xy^6 - 26xy^3z^6 + 24xy^3z^5 - 6xy^3z^4 - 16xy^3z^3 + 24xy^3z^2 - 8xy^3 + 21xz^8 + 26xz^6 - 3xz^4 - 12xz^2 + 4x - 3y^6z^6 \\ +8y^6z^4 - 4y^6z^2 + 8y^3z^8 - 8y^3z^7 - 6y^3z^6 + 16y^3z^5 - 16y^3z^4 + 8y^3z^2 - 5z^{10} - 8z^8 + 9z^6 + 8z^4 - 4z^2 = 0 \\ -y^9z^6 - 3y^9z^5 + 6y^9z^4 + 20y^9z^3 - 12y^9z^2 - 12y^9z + 8y^9 + 3y^6z^8 + 6y^6z^7 - 17y^6z^6 - 30y^6z^5 + 42y^6z^4 - 72y^6z^3 + 36y^6z^2 + 24y^6z \\ -24y^6 - 3y^3z^{10} - 7y^3z^9 + 42y^3z^7 + 21y^3z^6 + 33y^3z^5 - 6y^3z^4 + 52y^3z^3 - 36y^3z^2 - 12y^3z + 24y^3 + z^{12} + 3z^{10} - 3z^8 - 11z^6 + 6z^4 + 12z^2 - 8 = 0 \end{array} \right\}$$

$$\downarrow \quad F[3] = -4xz^2 + 4x + 1 + y^2 + z^4 - 2z^2$$

$9xy^6z^4 - 12xy^6z^2 + 4xy^6 - 26xy^3z^6 + 24xy^3z^5 - 6xy^3z^4 - 16xy^3z^3 + 24xy^3z^2 - 8xy^3 + 21xz^8 + 26xz^6 - 3xz^4 - 12xz^2 + 4x - 3y^6z^6 + 8y^6z^4 - 4y^6z^2 + 8y^3z^8 - 8y^3z^7 - 6y^3z^6 + 16y^3z^5 - 16y^3z^4 + 8y^3z^2 - 5z^{1}0 - 8z^8 + 9z^6 + 8z^4 - 4z^2 = 0$

$19972797246309946665991728185028495388912649549470666788546098y^{36}z + 255360896949225564096937270147875972975435590636918196007883733y^{36} - 773405114075532496596681506767164345221720878888299237363327579y^{35}z -$

$2842600320707143088653046402202027581491747757477831354032645337y^{35} + 4597491592171577745281415354161839265867411144225789772330185361y^{34}z + 13007839738889802463721307950046673388655716105599795947228758205y^{34}$

$-14551766654563233069769478299533042169715777774874033785235390770y^{33}z - 39311783847965836463763269677531892442728273759399670604908456620y^{33} + 42220772154794083042256951516806711984142300479410323993077675977y^{32}z$

$+135300918759792868141428977189324990687810898550776498646441963978y^{32} - 1842488278864165263626686429416979983191582708199387540864338503$ … $22746y^{31}$

$+489813093834904632283634107703886178745382333811043928288352420894y^{30}z + 11477452114986578374$ … 7103747 … $7428862y^{29}$

$-2343467409590551935074271467937059774894399065396070365346899765297y^{29} + 27327443138231997420$ … 1719192 … $43102978y$

$-6539036544425242929812484889939849791358217407825704966936886143564y^{27}z - 14247995836448282220$ … 9498925 … 641882467

$+19881741003546369294625878460392613127674196684911418852883519870499y^{26} - 21189209624374395671006259507671739941484193795920341064012836$ … 4197852584

$+4642023721205176827225954745933688646960896804305891438429367220 0926y^{24}z + 94954758126244976269801159566878860562138871385579671941248510333904y^{24} - 309658636370772671591721103764924235767856392915453400024481105722$

$-86816714547872691732407605049343157990564018727388616113576050293536y^{23} + 93712118065567693634752382202821927798161283293549073599065402840600y^{22}z + 235461750304570868800321055011893654069856015117424024431804090588$

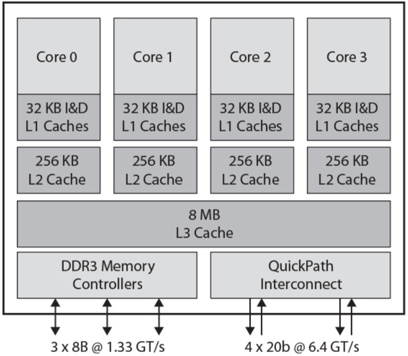
$-18659495591704192058865188469076539181172675929363364517928156716858 0y^{21}z - 351185482476438785264578009806366561807291909193756221896673883306491y^{21} + 2984214263477420058341738185389180546938210618368939009690971662$

[**B**GMY 23]

6202 characters to encode only 37 unique solutions

Research Themes and Interests

1. **Software Performance Engineering**
   - ↳ **Irregular Parallelism** and **Unstructured Parallelism**
   - ↳ **Dynamically Data-Intensive Computation**
   - ↳ Adaptive, dynamic scheduling and cooperation
   - ↳ Data locality, cache-efficient algorithms



Research Themes and Interests

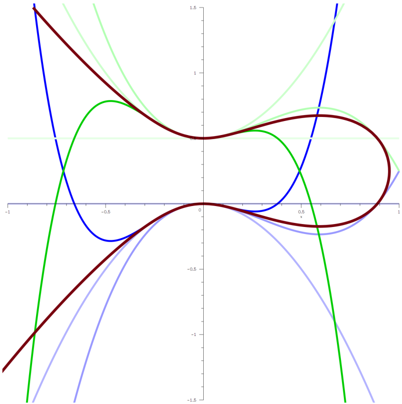
1. **Software Performance Engineering**
   - ↳ **Irregular Parallelism** and **Unstructured Parallelism**
   - ↳ **Dynamically Data-Intensive Computation**
   - ↳ Adaptive, dynamic scheduling and cooperation
   - ↳ Data locality, cache-efficient algorithms

2. **Scientific and Symbolic Computation**
   - ↳ Polynomial system solving
   - ↳ Data structures for large, potentially infinite objects
   - ↳ Making previously intractable problems tractable

Branches of $F(x, y) = 0$ as power series,

$$F(x,y) = y^2 - x^2 + \frac{9}{8}x^3 - \frac{y}{2}$$

[**B**KM 20] [**B**M 21]



Research Themes and Interests

1. **Software Performance Engineering**
   - ↳ **Irregular Parallelism** and **Unstructured Parallelism**
   - ↳ **Dynamically Data-Intensive Computation**
   - ↳ Adaptive, dynamic scheduling and cooperation
   - ↳ Data locality, cache-efficient algorithms
2. **Scientific and Symbolic Computation**
   - ↳ Polynomial system solving
   - ↳ Data structures for large, potentially infinite objects
   - ↳ Making previously intractable problems tractable
3. **Software Engineering for Science**
   - ↳ Mathematical and Scientific Software Design
   - ↳ Supporting **End-User Programming**
   - ↳ Performance, maintainability, adaptability, reproducibility, capability

Solving a Non-Linear System of Equations

Via **Gröbner Basis** we can "solve" a non-linear system

$$\begin{cases} x^2 + y + z = 1 \\ x + y^2 + z = 1 \\ x + y + z^2 = 1 \end{cases} \quad \underset{\text{F4 or F5 Algorithm}}{\overset{\text{Buchberger's Algorithm}}{\Longrightarrow}} \quad \begin{cases} x + y + z^2 - 1 = 0 \\ y^2 - z^2 - y + z = 0 \\ 2yz^2 + z^4 - z^2 = 0 \\ z^6 - 4z^4 + 4z^3 - z^2 = 0 \end{cases}$$

Solving a Non-Linear System of Equations

Via **Gröbner Basis** we can "solve" a non-linear system

$$\begin{cases} x^2 + y + z = 1 \\ x + y^2 + z = 1 \\ x + y + z^2 = 1 \end{cases} \xrightarrow[\text{F4 or F5 Algorithm}]{\text{Buchberger's Algorithm}} \begin{cases} x + y + z^2 - 1 = 0 \\ y^2 - z^2 - y + z = 0 \\ 2yz^2 + z^4 - z^2 = 0 \\ z^6 - 4z^4 + 4z^3 - z^2 = 0 \end{cases}$$

"Solving" a system is not just about finding particular values, rather:

> *"find a description of the solutions from which important data is easily extracted"*

Solving a Non-Linear System of Equations

Via **Gröbner Basis** we can "solve" a non-linear system

$$\begin{cases} x^2 + y + z = 1 \\ x + y^2 + z = 1 \\ x + y + z^2 = 1 \end{cases} \xrightarrow[\text{F4 or F5 Algorithm}]{\text{Buchberger's Algorithm}} \begin{cases} x + y + z^2 - 1 = 0 \\ y^2 - z^2 - y + z = 0 \\ 2yz^2 + z^4 - z^2 = 0 \\ z^6 - 4z^4 + 4z^3 - z^2 = 0 \end{cases}$$

"Solving" a system is not just about finding particular values, rather:

> *"find a description of the solutions from which important data is easily extracted"*

Why?

- A **positive-dimensional system** has an **infinite** number of solutions
- *Underdetermined* linear systems, most non-linear systems
- For polynomials of degree > 4, their solutions (usually) cannot be described in *radicals*

Triangular Decomposition of a Non-Linear System

$$\begin{cases} x^2 + y + z = 1 \\ x + y^2 + z = 1 \\ x + y + z^2 = 1 \end{cases} \xrightarrow{\text{Gröbner Basis}} \begin{cases} x + y + z^2 = 1 \\ (y + z - 1)(y - z) = 0 \\ z^2\left(z^2 + 2y - 1\right) = 0 \\ z^2\left(z^2 + 2z - 1\right)(z - 1)^2 = 0 \end{cases}$$

$\Downarrow$ **Triangular Decomposition**

$$\begin{cases} x - z = 0 \\ y - z = 0 \\ z^2 + 2z - 1 = 0 \end{cases}, \quad \begin{cases} x = 0 \\ y = 0 \\ z - 1 = 0 \end{cases}, \quad \begin{cases} x = 0 \\ y - 1 = 0 \\ z = 0 \end{cases}, \quad \begin{cases} x - 1 = 0 \\ y = 0 \\ z = 0 \end{cases}$$

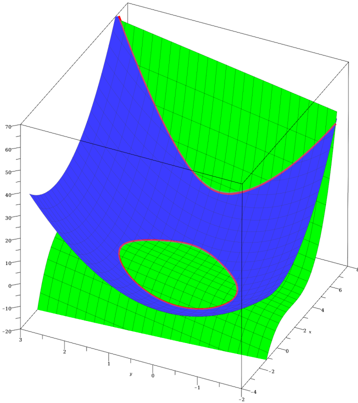
In triangular decomposition, **multiple components** are found, suggesting possible **component-level parallelism** [17], [A**B**MMX 20], [A**B**MMX 23]

Triangular Decomposition of a Non-Linear System

$$\begin{cases} x^2 + y + z = 1 \\ x + y^2 + z = 1 \\ x + y + z^2 = 1 \end{cases} \quad \overset{\text{Gröbner Basis}}{\Longrightarrow} \quad \begin{cases} x + y + z^2 = 1 \\ (y + z - 1)(y - z) = 0 \\ z^2\left(z^2 + 2y - 1\right) = 0 \\ z^2\left(z^2 + 2z - 1\right)(z - 1)^2 = 0 \end{cases}$$

$\Downarrow$ **Triangular Decomposition**

$$\begin{cases} x - z = 0 \\ y - z = 0 \\ z^2 + 2z - 1 = 0 \end{cases}, \quad \begin{cases} x = 0 \\ y = 0 \\ z - 1 = 0 \end{cases}, \quad \begin{cases} x = 0 \\ y - 1 = 0 \\ z = 0 \end{cases}, \quad \begin{cases} x - 1 = 0 \\ y = 0 \\ z = 0 \end{cases}$$



In triangular decomposition, **multiple components** are found, suggesting possible **component-level parallelism** [17], [A**B**MMX 20], [A**B**MMX 23]

Triangular Sets, Triangular Decomposition

Polynomials in variables $\underline{X} = x_1 < x_2 < \cdots < x_n$ form an algebraic ring $\mathbb{K}[\underline{X}]$

Triangular set $T \subset \mathbb{K}[\underline{X}]$

↳ a set of polynomials with pairwise different maximum ("main") variables

$$\left\{ \begin{array}{r} (2y + ba)x - by + a^2 \\ 2y^2 - by - a^2 \\ a + b \end{array} \right\} \subset \mathbb{Q}[b < a < y < x]$$

Initial: the lead. coeff. w.r.t. the main variable

$$h_T = \prod_{t \in T} \text{initial}(t)$$

Triangular Sets, Triangular Decomposition

Polynomials in variables $\underline{X} = x_1 < x_2 < \cdots < x_n$ form an algebraic ring $\mathbb{K}[\underline{X}]$

Triangular set $T \subset \mathbb{K}[\underline{X}]$

↳ a set of polynomials with pairwise different maximum ("main") variables

$$\left\{ \begin{array}{r} (2y + ba)x - by + a^2 \\ 2y^2 - by - a^2 \\ a + b \end{array} \right\} \subset \mathbb{Q}[b < a < y < x]$$

Initial: the lead. coeff. w.r.t. the main variable

$$h_T = \prod_{t \in T} \text{initial}(t)$$

Algebraic Variety $V(F)$

↳ set of common solutions of a polynomial system $F \subset \mathbb{K}[\underline{X}]$

Quasi-Component

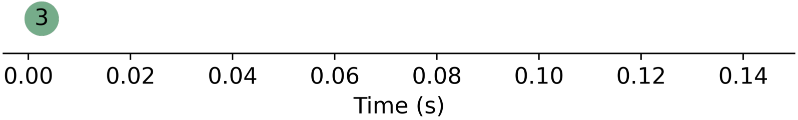
$$W(T) = V(T) \setminus V(h_T)$$

Triangular Decomposition: find T_i s.t.

$$\bigcup_{i=1}^{e} W(T_i) = V(F)$$

$\{y + w\}$

$$F = \begin{cases} y + w \\ 5w^2 + y \\ xz + z^3 + z \\ x^5 + x^3 + z \end{cases}$$



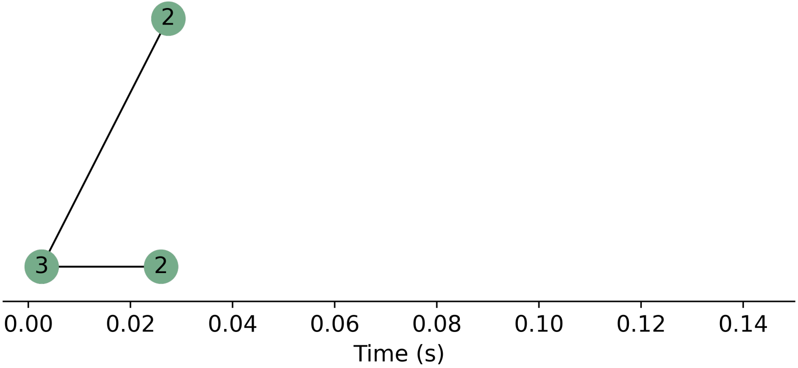
Decomposition of the Solution Space: **Intersect**

$$\{y + w\}$$

$$\downarrow F[2] \qquad \searrow$$

$$\left\{ \begin{matrix} 5y + 1 \\ 5w - 1 \end{matrix} \right\} \qquad \left\{ \begin{matrix} y \\ w \end{matrix} \right\}$$

$$F = \begin{cases} y + w \\ 5w^2 + y \\ xz + z^3 + z \\ x^5 + x^3 + z \end{cases}$$



Decomposition of the Solution Space: **Intersect**

$$\{y + w\}$$

$$\downarrow F[2] \qquad \searrow$$

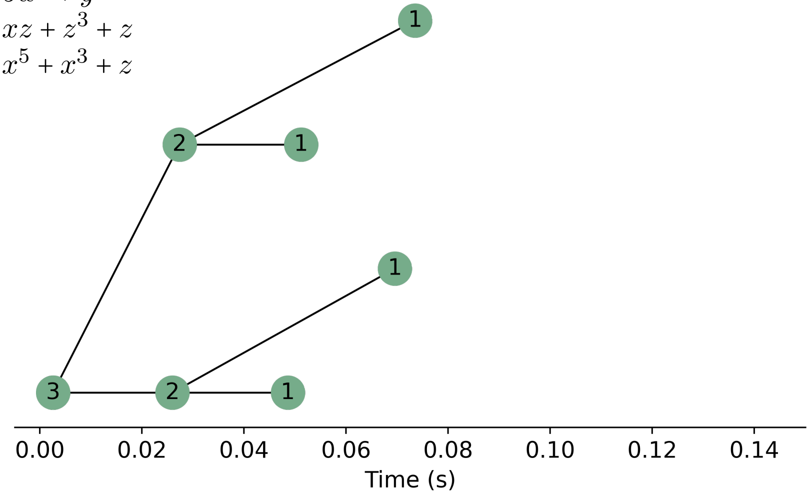
$$\begin{Bmatrix} 5y + 1 \\ 5w - 1 \end{Bmatrix} \qquad \begin{Bmatrix} y \\ w \end{Bmatrix}$$

$$\downarrow \qquad F[3] \qquad \searrow$$

$$\begin{Bmatrix} x + z^2 + 1 \\ 5y + 1 \\ 5w - 1 \end{Bmatrix} \qquad \begin{Bmatrix} x + z^2 + 1 \\ y \\ w \end{Bmatrix}$$

$$\begin{Bmatrix} 5y + 1 \\ z \\ 5w - 1 \end{Bmatrix} \qquad \begin{Bmatrix} y \\ z \\ w \end{Bmatrix}$$

$$F = \begin{cases} y + w \\ 5w^2 + y \\ xz + z^3 + z \\ x^5 + x^3 + z \end{cases}$$



Decomposition of the Solution Space: **Intersect**

$$F = \begin{cases} y + w \\ 5w^2 + y \\ xz + z^3 + z \\ x^5 + x^3 + z \end{cases}$$

$$\{y + w\}$$

$\downarrow F[2]$ $\searrow$

$$\begin{Bmatrix} 5y + 1 \\ 5w - 1 \end{Bmatrix} \qquad \begin{Bmatrix} y \\ w \end{Bmatrix}$$

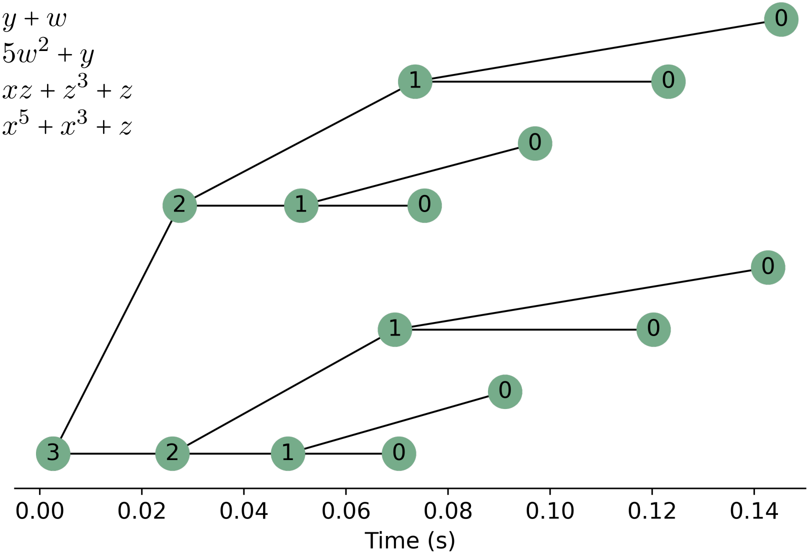
$\downarrow$ $F[3]$ $\searrow$

$$\begin{Bmatrix} x + z^2 + 1 \\ 5y + 1 \\ 5w - 1 \end{Bmatrix} \qquad \begin{Bmatrix} x + z^2 + 1 \\ y \\ w \end{Bmatrix}$$

$$\begin{Bmatrix} 5y + 1 \\ z \\ 5w - 1 \end{Bmatrix} \qquad \begin{Bmatrix} y \\ z \\ w \end{Bmatrix}$$

$\swarrow\swarrow\searrow\searrow$ $F[4]$ $\swarrow\swarrow\searrow\searrow$

$\dots$ $\dots$



Decomposition as a Case Discussion

To compute an intersection, initials must be **regular**: non-zero nor a zero-divisor

Regular Chain: a special triangular set where every initial is regular

Split computations via a **case discussion**: either the initial is zero or it is non-zero

$$f = (y+1)x^2 - x$$

$$T = \begin{cases} y^2 - 1 = 0 \\ z - 1 = 0 \end{cases}$$

Decomposition as a Case Discussion

To compute an intersection, initials must be **regular**: non-zero nor a zero-divisor

Regular Chain: a special triangular set where every initial is regular

Split computations via a **case discussion**: either the initial is zero or it is non-zero

$$f = (y+1)x^2 - x$$

$$T = \begin{cases} y^2 - 1 = 0 \\ z - 1 = 0 \end{cases}$$

$\xrightarrow{y+1=0}$

$$T_1 = \begin{cases} y + 1 = 0 \\ z - 1 = 0 \end{cases}$$

$\xrightarrow{f = x}$

$$T_3 = \begin{cases} x = 0 \\ y + 1 = 0 \\ z - 1 = 0 \end{cases}$$

Decomposition as a Case Discussion

To compute an intersection, initials must be **regular**: non-zero nor a zero-divisor

Regular Chain: a special triangular set where every initial is regular

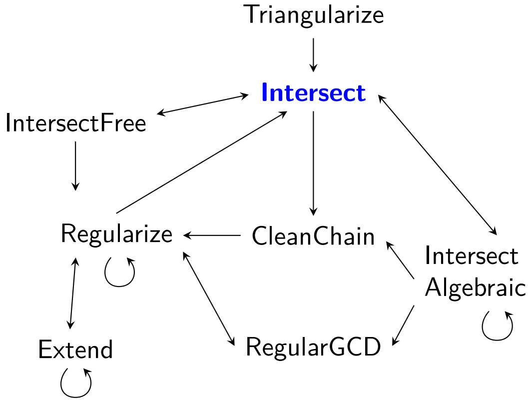
Split computations via a **case discussion**: either the initial is zero or it is non-zero

$$f = (y+1)x^2 - x$$

$$T = \begin{cases} y^2 - 1 = 0 \\ z - 1 = 0 \end{cases}$$

$$\xrightarrow{y+1=0} \quad T_1 = \begin{cases} y + 1 = 0 \\ z - 1 = 0 \end{cases} \quad \xrightarrow{f = x} \quad T_3 = \begin{cases} x = 0 \\ y + 1 = 0 \\ z - 1 = 0 \end{cases}$$

$$\xrightarrow{y+1\neq 0} \quad T_2 = \begin{cases} y - 1 = 0 \\ z - 1 = 0 \end{cases} \quad \xrightarrow{f = 2x^2 - x} \quad T_4 = \begin{cases} 2x^2 - x = 0 \\ y - 1 = 0 \\ z - 1 = 0 \end{cases}$$



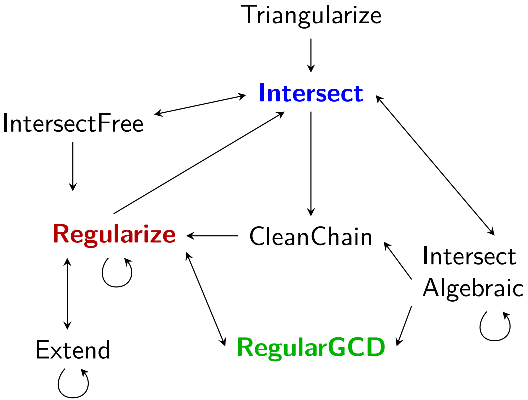
Workpile pattern

Algorithm TriangularizeByTasks(F)

$\textit{Tasks} := \{ (F, \varnothing) \}$; $\mathcal{T} := \varnothing$
while $|\textit{Tasks}| > 0$ **do**
  $(P, T) :=$ pop a task from *Tasks*
  Choose a polynomial $p \in P$; $P' := P \setminus \{p\}$
  for T' in **Intersect**(p, T) **do**
    if $|P'| = 0$ **then** $\mathcal{T} := \mathcal{T} \cup \{T'\}$
    else $\textit{Tasks} := \textit{Tasks} \cup \{(P', T')\}$
return $\mathcal{T}$

Algorithms for Triangular Decomposition

Producer-Consumer, **Pipeline** patterns

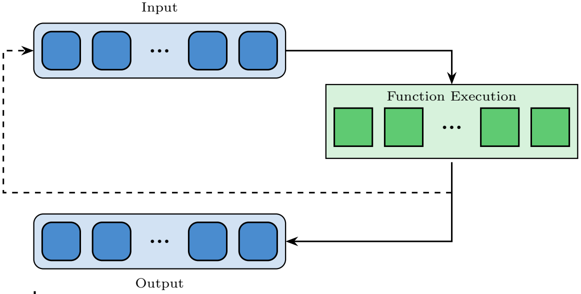


Algorithm Regularize(p, T)

for $(g_i, T_i) \in$ RegularGCD(p, T_v, T_v^-) **do**
  if $\dim(T_i) < \dim(T_v^-)$ **then**
    for $T_j \in$ Regularize(p, T_i) **do**
      yield T_j
  else if $g_i \notin \mathbb{K}$ and $\deg(g_i, v) > 0$ **then**
    yield $T_i \cup g_i$
    yield $T_i \cup \mathrm{pquo}(T_v, g_i)$
    for $T_{i,j} \in$ Intersect$(\mathrm{lc}(g_i, v), T_i)$ **do**
      for $T' \in$ Regularize$(p, T_{i,j})$ **do**
        yield T'
  else
    yield T_i

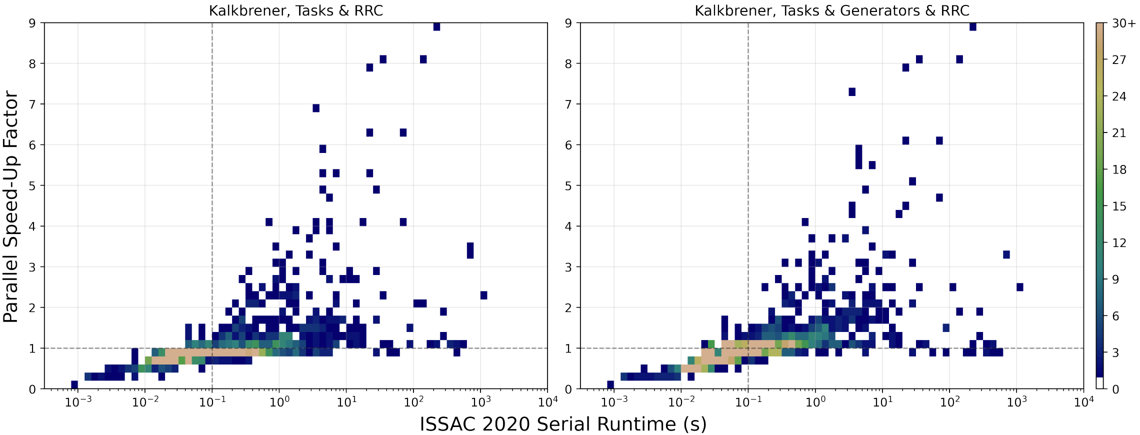
Support for Cooperative Irregular Parallelism

- Object-oriented library on top of C++11 threads [**B** 22]
- Shared thread pool
- **Parallel patterns** and their **composition**:
  - ↳ Map, Workpile, Fork-Join,
- Asynchronous Generators (coroutines) support
  - ↳ Producer-Consumer, Pipeline
- **Priority threads** allow temporary *oversubscription*
  - ↳ Start **coarse-grained** tasks as soon as possible
  - ↳ Prioritize tasks with **potential to expose more parallelism**
  - ↳ Self-regulating distribution of resources



Basic Polynomial Algebra Subprograms

Parallel Speedup in Triangular Decomposition



Parallel speedup on 12 cores [A**B**MMX 20]

Range of Regularity in Parallelism

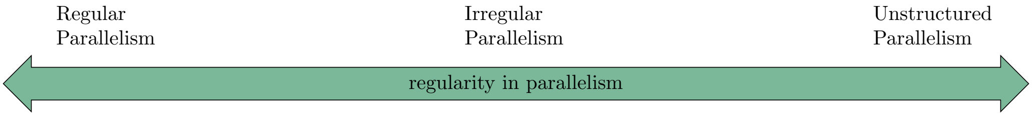
$$\begin{bmatrix} a_{1,1} & a_{2,1} & a_{3,1} & a_{4,1} \\ a_{1,2} & a_{2,2} & a_{3,2} & a_{4,2} \\ a_{1,3} & a_{2,3} & a_{3,3} & a_{4,3} \\ a_{1,4} & a_{2,4} & a_{3,4} & a_{4,4} \end{bmatrix} \times \begin{bmatrix} b_{1,1} & b_{2,1} & b_{3,1} & b_{4,1} \\ b_{1,2} & b_{2,2} & b_{3,2} & b_{4,2} \\ b_{1,3} & b_{2,3} & b_{3,3} & b_{4,3} \\ b_{1,4} & b_{2,4} & b_{3,4} & b_{4,4} \end{bmatrix} = \begin{bmatrix} c_{1,1} & c_{2,1} & c_{3,1} & c_{4,1} \\ c_{1,2} & c_{2,2} & c_{3,2} & c_{4,2} \\ c_{1,3} & c_{2,3} & c_{3,3} & c_{4,3} \\ c_{1,4} & c_{2,4} & c_{3,4} & c_{4,4} \end{bmatrix}$$

Regular parallelism:

↳ algorithms guaranteed to decompose into independent tasks regardless of problem instance

↳ classic **scalable parallelism**

↳ Regular and data parallelism well-supported by AVX, AMX, Cilk, OpenMP, TBB, CUDA



Range of Regularity in Parallelism

Irregular parallelism:

↳ sparse data structures, graphs

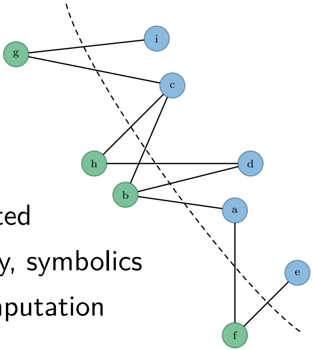
↳ available parallelism changes with each problem instance

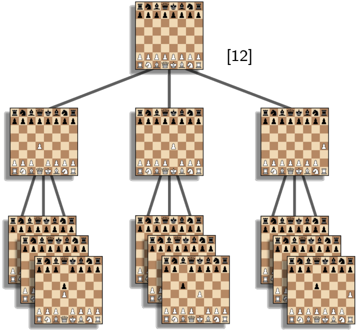
Unstructured parallelism:

↳ tasks/data dynamically generated

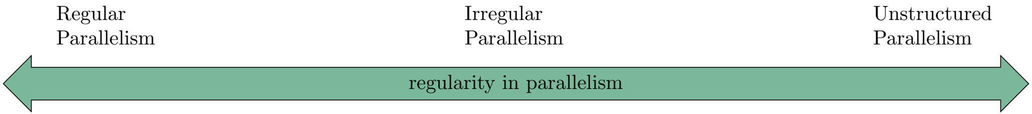
↳ state-space search, game theory, symbolics

↳ dynamically data-intensive computation





[12]



Up Next: Supporting Irregular Parallelism

Goal: computational model for unstructured, data-intensive computation

- *Dataflow* [18], *Codelet* [20] effective for irregular, sparse graph algorithms
- Work, Span, Parallel Speed-Up difficult to quantify for unstructured problems

Up Next: Supporting Irregular Parallelism

Goal: computational model for unstructured, data-intensive computation

- *Dataflow* [18], *Codelet* [20] effective for irregular, sparse graph algorithms
- Work, Span, Parallel Speed-Up difficult to quantify for unstructured problems

Goal: practical support for unstructured parallelism

- composition, cooperation of parallel regions [A**B**MMX 20], [A**B**M 21], [**B**M 21], [**B** 22]
- **locality-aware** scheduling, resource (re-)distribution [21]

Up Next: Supporting Irregular Parallelism

Goal: computational model for unstructured, data-intensive computation

- *Dataflow* [18], *Codelet* [20] effective for irregular, sparse graph algorithms
- Work, Span, Parallel Speed-Up difficult to quantify for unstructured problems

Goal: practical support for unstructured parallelism

- composition, cooperation of parallel regions [A**B**MMX 20], [A**B**M 21], [**B**M 21], [**B** 22]
- **locality-aware** scheduling, resource (re-)distribution [21]

Questions:

- How much parallelism is available for a given problem instance?
- What amount of potential parallelism was exploited?
- Adaptive, auto-tuned scheduling as problem evolves at runtime?

Up Next: Supporting Irregular Parallelism and Exploiting it!

Goal: computational model for unstructured, data-intensive computation

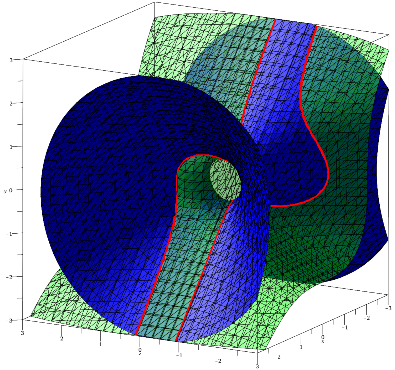
- *Dataflow* [18], *Codelet* [20] effective for irregular, sparse graph algorithms
- Work, Span, Parallel Speed-Up difficult to quantify for unstructured problems

Goal: practical support for unstructured parallelism

- composition, cooperation of parallel regions [A**B**MMX 20], [A**B**M 21], [**B**M 21], [**B** 22]
- **locality-aware** scheduling, resource (re-)distribution [21]

Questions:

- How much parallelism is available for a given problem instance?
- What amount of potential parallelism was exploited?
- Adaptive, auto-tuned scheduling as problem evolves at runtime?



$$x^3 + y^3 + z^2 = \frac{1}{2}$$
$$x^2 - y^2 = z^2 + z$$

The BPAS Library [2]

Basic Polynomial Algebra Subprograms

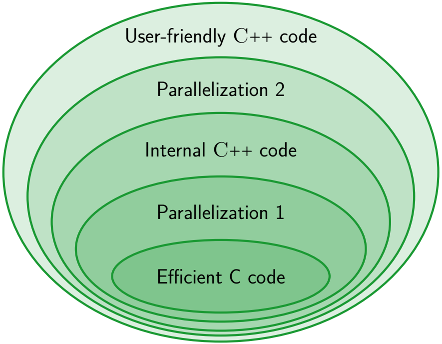
"BLAS for Polynomials"

- Open-source library for fundamental polynomial data structures, arithmetic, operations [**B**AMM 19]
- Finite fields, integers, rational numbers, intervals
- GCDs, factorization [**B** 2022]
- Polynomial system solving [A**B**MMX 20], [A**B**MMX 23]
- Multivariate power series and polynomials over power series [**B**KM 20], [**B**M 21]
- Optimized for cache complexity and multicore execution; generic support for composed parallelism [**B** 22]
- Over 600,000 lines of code

Towards Performance, Maintainability, and Ease-of-Use

Layered Architecture of BPAS [BMM 20]

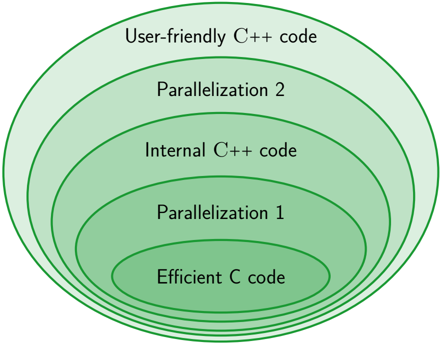
- Performance-critical code implemented in C
- Lightweight C++ class interface for ease of use
- Class templates and abstract classes for extensibility
- Parallel constructs completely encapsulated



Towards Performance, Maintainability, and Ease-of-Use

Layered Architecture of BPAS [BMM 20]

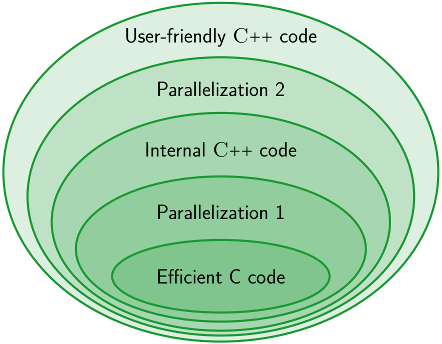
- Performance-critical code implemented in C
- Lightweight C++ class interface for ease of use
- Class templates and abstract classes for extensibility
- Parallel constructs completely encapsulated
- ***Make it hard to do the wrong thing***



Towards Performance, Maintainability, and Ease-of-Use

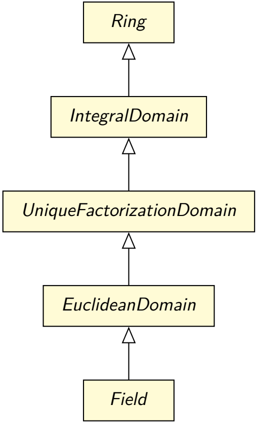
Layered Architecture of BPAS [BMM 20]

- Performance-critical code implemented in C
- Lightweight C++ class interface for ease of use
- Class templates and abstract classes for extensibility
- Parallel constructs completely encapsulated
- ***Make it hard to do the wrong thing***
- ***Encapsulate all complexity on the developer's side***



Working with Algebraic Types

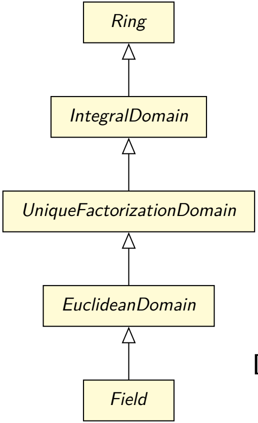
Algebraic types naturally form a hierarchy



- *Rings* define addition and multiplication of elements
- *Integral domains* add the notion of divisibility
- *Unique Factorization Domains*: every element is a product of primes
- *Euclidean domain* have division with remainder
- *Field*: every element has an inverse

Working with Algebraic Types

Algebraic types naturally form a hierarchy



- *Rings* define addition and multiplication of elements
- *Integral domains* add the notion of divisibility
- *Unique Factorization Domains*: every element is a product of primes
- *Euclidean domain* have division with remainder
- *Field*: every element has an inverse

Different *concrete types* of rings are **mathematically incompatible**
↳ $\mathbb{Z}$ is a Euclidean domain; so are univariate polynomials over a finite field

Make it Hard to do the Wrong Thing: Algebraic Type Safety

In existing algebra software, type safety is only a **runtime trait**

- One class for the ring itself
- One class for elements of a ring
- *Singular*: instance variables of Booleans and enumeration [13]
- *CoCoALib*: method overrides return Booleans; `IamField()` [1]

Make it Hard to do the Wrong Thing: Algebraic Type Safety

In existing algebra software, type safety is only a **runtime trait**

- One class for the ring itself
- One class for elements of a ring
- *Singular*: instance variables of Booleans and enumeration [13]
- *CoCoALib*: method overrides return Booleans; `IamField()` [1]

BPAS enforces **static type safety**

- **Algebraic class hiearchy of class templates**
- Rings: classes. Ring Elements: objects of that class
- **Curiously Recurring Template Pattern**
- Compile-time function resolution enforces safety
- [**B**MM 20]

Make it Hard to do the Wrong Thing: Algebraic Type Safety

```
template <class Derived>
class EuclidDomain : GCDDomain<Derived> {
        Derived remainder(const Derived div);
    };

class Integer : EuclidDomain<Integer> {};
//Integer remainder(const Integer div);

class RationalPoly : EuclidDomain<RatonalPoly> {};
//RationalPoly remainder(const RationalPoly div);

Integer x;  RationalPoly p;

//compiler error: EuclidDomain<RationalPoly>::
    remainder takes RationalPoly as parameter
RationalPoly r = p.remainder(x);
```

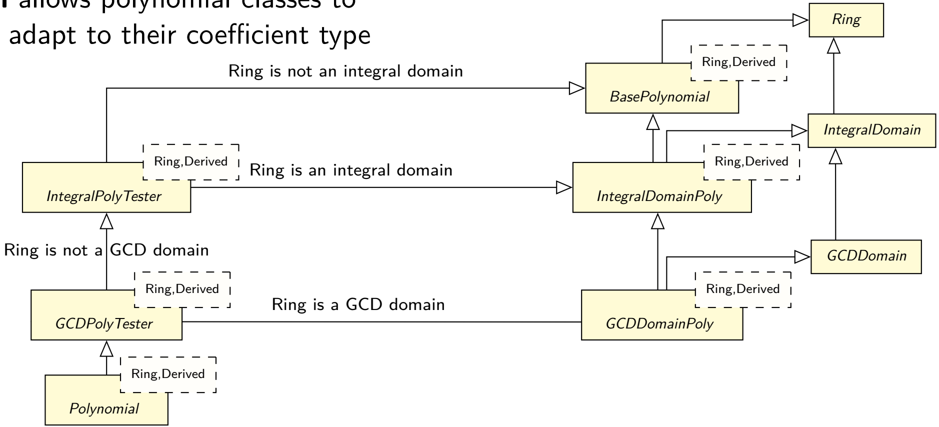
BPAS enforces **static type safety**

- **Algebraic class hiearchy of class templates**
- Rings: classes. Ring Elements: objects of that class
- **Curiously Recurring Template Pattern**
- Compile-time function resolution enforces safety
- [**B**MM 20]

Extensibility and Flexibility

At present:

- User-defined types automatically integrate into hierarchy
- **Template metaprogramming** and **compile-time introspection** allows polynomial classes to automatically adapt to their coefficient type



Up Next: Extensibility and Flexibility

At present:

- User-defined types automatically integrate into hierarchy
- **Template metaprogramming** and **compile-time introspection** for polynomial classes

Goals:

- Integrate with interactive environments for rapid prototyping, accessibility, usability
- High-performance bi-directional symbolic computing software stack, **polynomial system solving**

Up Next: Extensibility and Flexibility

At present:

- User-defined types automatically integrate into hierarchy
- **Template metaprogramming** and **compile-time introspection** for polynomial classes

Goals:

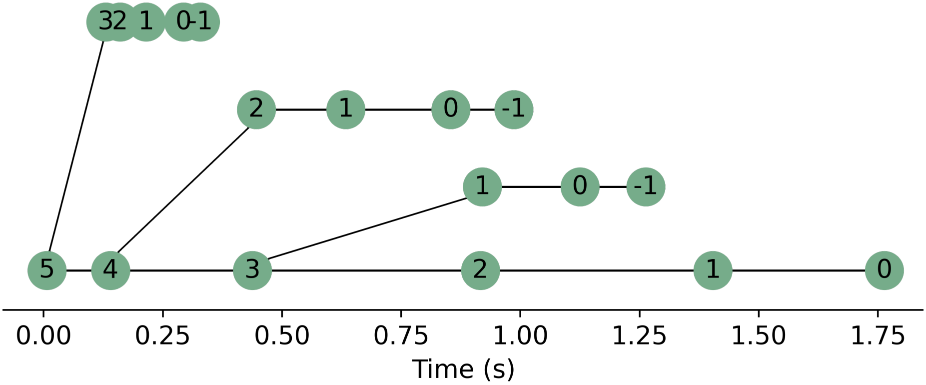
- Integrate with interactive environments for rapid prototyping, accessibility, usability
- High-performance bi-directional symbolic computing software stack, **polynomial system solving**

Questions:

- How to handle performance, data locality in software stack with **expression swell**?
- Can types defined in the interactive environment be used in algebraic class hierarchy?

Goal: Irregular and unstructured parallelism in theory and practice

- Continued development of cooperative parallelism for irregular applications
- Metrics to quantify success of per-problem irregular parallelism speed-up

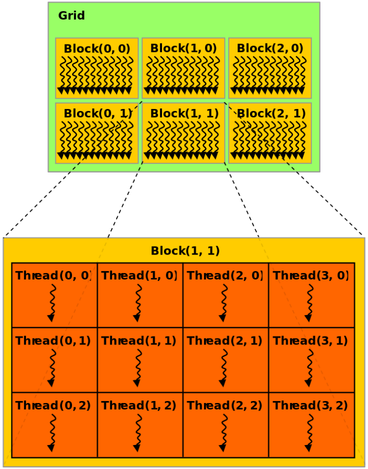


The Next Five Years

Goal: Irregular and unstructured parallelism in theory and practice

Goal: Automatic parallel program parameter optimization

- **Program parameters**: values which control how tasks and mapped to resources
- Particularly important for CUDA as **thread block configurations** [**B**MMPW 19]
- Extend to dynamic scheduling of irregular parallelism on multicores



The Next Five Years

Goal: Irregular and unstructured parallelism in theory and practice

Goal: Automatic parallel program parameter optimization

Goal: SOFTWARE MODULARIZATION is NP-complete?

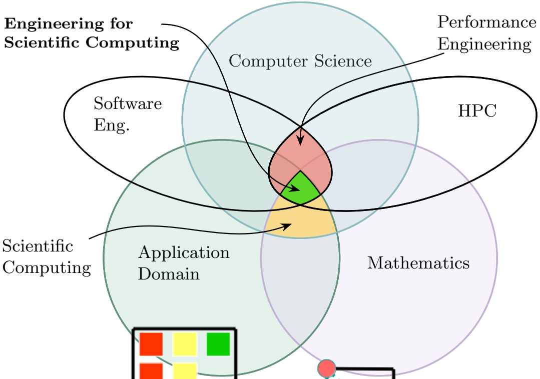
- No formal proof of tractability of SOFTWARE MODULARIZATION exists
- Model as a graph problem which considers inter-module edges **and** intra-module edges
- Provably near-optimal approximation algorithms

Research Vision in a *Post-Moore Era of Computing*

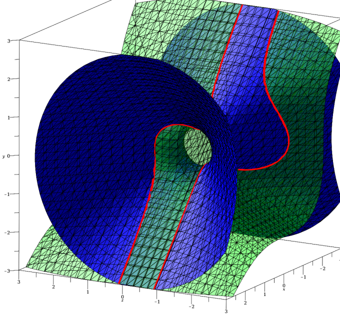
With the end of *Moore's Law*:

1. Performance engineering in symbolic and scientific computation
2. Design and engineer lasting, high-performance, easy-to-use **mathematic software**
3. Software engineering practices for scientific research software, end-user programming

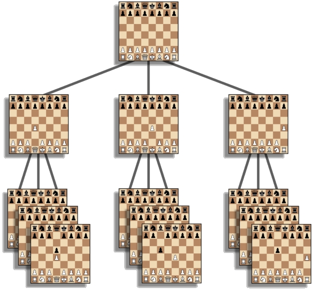
> *"Only by building on top of high-quality, high-performing, reusable code can researchers hope for the continued development of novel software solutions to scientific problems."*

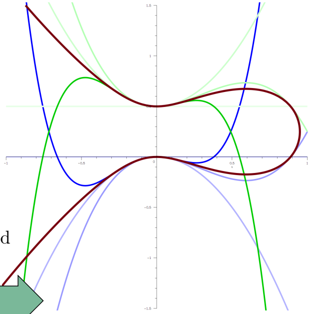


Thank you!



Basic Polynomial Algebra Subprograms





Regular Parallelism

Irregular Parallelism

Unstructured Parallelism

regularity in parallelism

References

[1] J. Abbott and A. M. Bigatti. *CoCoALib: a C++ library for doing Computations in Commutative Algebra*. Available at http://cocoa.dima.unige.it/cocoalib.

[2] M. Asadi, A. Brandt, C. Chen, S. Covanov, J. González Trochez, F. Mansouri, D. Mohajerani, R. H. C. Moir, M. Moreno Maza, D. Talaashrafi, L. Wang, N. Xie, Y. Xie, and H. Yuan. *Basic Polynomial Algebra Subprograms (BPAS)*. www.bpaslib.org. 2023.

[3] M. Asadi, A. Brandt, R. H. C. Moir, M. Moreno Maza, and Y. Xie. "Parallelization of triangular decompositions: Techniques and implementation". In: *J. Symb. Comput.* 115 (2023), pp. 371–406.

[4] M. Asadi, A. Brandt, R. H. C. Moir, and M. Moreno Maza. "Algorithms and Data Structures for Sparse Polynomial Arithmetic". In: *Mathematics* 7.5 (2019), p. 441.

[5] M. Asadi, A. Brandt, R. H. C. Moir, M. Moreno Maza, and Y. Xie. "On the parallelization of triangular decompositions". In: *Proc. of ISSAC 2020*. ACM, 2020, pp. 22–29.

[6] M. Asadi, A. Brandt, and M. Moreno Maza. "Computational Schemes for Subresultant Chains". In: *Proc. of CASC 2021*. Vol. 12865. Lecture Notes in Computer Science. Springer, 2021, pp. 21–41.

[7] A. Brandt. "The Design and Implementation of a High-Performance Polynomial System Solver". PhD thesis. University of Western Ontario, 2022.

[8] A. Brandt, M. Kazemi, and M. Moreno Maza. "Power Series Arithmetic with the BPAS Library". In: *Proc. of CASC 2020*. Vol. 12291. Lecture Notes in Computer Science. Springer, 2020, pp. 108–128.

[9] A. Brandt, D. Mohajerani, M. M. Maza, J. Paudel, and L. Wang. "KLARAPTOR: A Tool for Dynamically Finding Optimal Kernel Launch Parameters Targeting CUDA Programs". In: *CoRR* (2019). arXiv: 1911.02373.

[10] A. Brandt, R. H. C. Moir, and M. Moreno Maza. "Employing C++ Templates in the Design of a Computer Algebra Library". In: *Proc. of ICMS 2020*. Vol. 12097. Lecture Notes in Computer Science. Springer, 2020, pp. 342–352.

[11] A. Brandt and M. Moreno Maza. “On the Complexity and Parallel Implementation of Hensel’s Lemma and Weierstrass Preparation”. In: *Computer Algebra in Scientific Computing, CASC 2021, Proceedings*. Vol. 12865. Lecture Notes in Computer Science. Springer, 2021, pp. 78–99.

[12] C. Butner. *ChessCoach*. https://chrisbutner.github.io/ChessCoach/. 2023.

[13] W. Decker, G.-M. Greuel, G. Pfister, and H. Schönemann. SINGULAR *4-1-1 — A computer algebra system for polynomial computations*. http://www.singular.uni-kl.de. 2018.

[14] J. Grabmeier, E. Kaltofen, and V. Weispfenning, eds. *Computer algebra handbook*. Springer-Verlag, 2003.

[15] D. Kelly. “A Software Chasm: Software Engineering and Scientific Computing”. In: *IEEE Software* 24.6 (2007), pp. 118–120.

[16] M. McCool, J. Reinders, and A. Robison. *Structured parallel programming*. Elsevier, 2012.

[17] M. Moreno Maza and Y. Xie. “Component-level parallelization of triangular decompositions”. In: *Proc. of PASCO 2007*. ACM. 2007, pp. 69–77.

[18] K. Pingali, D. Nguyen, M. Kulkarni, M. Burtscher, M. A. Hassaan, R. Kaleem, T.-H. Lee, A. Lenharth, R. Manevich, M. Méndez-Lojo, et al. “The tao of parallelism in algorithms”. In: *Proc. of SIGPLAN conference on Programming language design and implementation*. 2011, pp. 12–25.

[19] T. Storer. “Bridging the Chasm: A Survey of Software Engineering Practice in Scientific Programming”. In: *ACM Computing Surveys* 50.4 (2017), 47:1–47:32.

[20] J. Suettlerlein, S. Zuckerman, and G. R. Gao. “An implementation of the codelet model”. In: *Proc. of Euro-Par 2013 Parallel Processing*. Springer. 2013, pp. 633–644.

[21] R. M. Yoo, C. J. Hughes, C. Kim, Y.-K. Chen, and C. Kozyrakis. “Locality-Aware Task Management for Unstructured Parallelism: A Quantitative Limit Study”. In: *Proc. of Symposium on Parallelism in Algorithms and Architectures*. SPAA ’13. ACM, 2013, pp. 315–325.