

**axiom<sup>TM</sup>**

**Generic, open and powerful**

**Marc Moreno Maza**

**September 1, 2006**

**ORCA**

## An introductory example

Z := Integer

(1) Integer

Type: Domain

Time: 0 sec

Q := Fraction Z

(2) Fraction Integer

Type: Domain

Time: 0 sec

P := Polynomial Q

(3) Polynomial Fraction Integer

Type: Domain

Time: 0 sec

K := Fraction P

(4) Fraction Polynomial Fraction Integer

Type: Domain

Time: 0 sec

```
M := SquareMatrix(4, K)
```

```
(5) SquareMatrix(4, Fraction Polynomial Fraction Integer)
                                         Type: Domain
                                         Time: 0 sec
```

```
quaternion(a: K, b: K, c: K, d: K): M ==
matrix ([[d , a , b , c],
        [-a , d , -c , b],
        [-b , c , d , -a],
        [-c , -b , a , d ]])
```

```
Function declaration quaternion : (Fraction Polynomial
Fraction Integer, Fraction Polynomial Fraction Integer,
Fraction Polynomial Fraction Integer, Fraction Polynomial
Fraction Integer) -> SquareMatrix(4, Fraction Polynomial
Fraction Integer) has been added to workspace.
```

q := quaternion(1,2,3,x/y)

$$\begin{array}{cccc}
 & + & x & + \\
 & | & - & 1 & 2 & 3 & | \\
 & | & y & & & & | \\
 & | & & & & & | \\
 & | & & x & & & | \\
 & | & - & 1 & - & 3 & 2 & | \\
 & | & & y & & & & | \\
 (7) & | & & & & & & | \\
 & | & & & x & & & | \\
 & | & - & 2 & 3 & - & - & 1 & | \\
 & | & & & y & & & & | \\
 & | & & & & & & & | \\
 & | & & & & & x & & | \\
 & | & - & 3 & - & 2 & 1 & - & | \\
 & + & & & & & y & + & 
 \end{array}$$

recip(q)

$$\begin{array}{c}
 + \quad \begin{array}{c} 1 \\ \text{-- } x y \\ 14 \end{array} \quad \begin{array}{c} 1 \ 2 \\ \text{-- } y \\ 14 \end{array} \quad \begin{array}{c} 1 \ 2 \\ - y \\ 7 \end{array} \quad \begin{array}{c} 3 \ 2 \\ \text{-- } y \\ 14 \end{array} + \\
 | \text{-----} | \text{-----} | \text{-----} | \text{-----} | \\
 | \begin{array}{c} 2 \ 1 \ 2 \\ y + \text{-- } x \\ 14 \end{array} \quad \begin{array}{c} 2 \ 1 \ 2 \\ y + \text{-- } x \\ 14 \end{array} \quad \begin{array}{c} 2 \ 1 \ 2 \\ y + \text{-- } x \\ 14 \end{array} \quad \begin{array}{c} 2 \ 1 \ 2 \\ y + \text{-- } x \\ 14 \end{array} | \\
 | \begin{array}{c} 1 \ 2 \\ \text{-- } y \\ 14 \end{array} \quad \begin{array}{c} 1 \\ \text{-- } x y \\ 14 \end{array} \quad \begin{array}{c} 3 \ 2 \\ \text{-- } y \\ 14 \end{array} \quad \begin{array}{c} 1 \ 2 \\ - y \\ 7 \end{array} | \\
 | \text{-----} | \text{-----} | \text{-----} | \text{-----} | \\
 | \begin{array}{c} 2 \ 1 \ 2 \\ y + \text{-- } x \\ 14 \end{array} \quad \begin{array}{c} 2 \ 1 \ 2 \\ y + \text{-- } x \\ 14 \end{array} \quad \begin{array}{c} 2 \ 1 \ 2 \\ y + \text{-- } x \\ 14 \end{array} \quad \begin{array}{c} 2 \ 1 \ 2 \\ y + \text{-- } x \\ 14 \end{array} | \\
 (8) \quad | \begin{array}{c} 1 \ 2 \\ - y \\ 7 \end{array} \quad \begin{array}{c} 3 \ 2 \\ \text{-- } y \\ 14 \end{array} \quad \begin{array}{c} 1 \\ \text{-- } x y \\ 14 \end{array} \quad \begin{array}{c} 1 \ 2 \\ \text{-- } y \\ 14 \end{array} | \\
 | \text{-----} | \text{-----} | \text{-----} | \text{-----} | \\
 | \begin{array}{c} 2 \ 1 \ 2 \\ y + \text{-- } x \\ 14 \end{array} \quad \begin{array}{c} 2 \ 1 \ 2 \\ y + \text{-- } x \\ 14 \end{array} \quad \begin{array}{c} 2 \ 1 \ 2 \\ y + \text{-- } x \\ 14 \end{array} \quad \begin{array}{c} 2 \ 1 \ 2 \\ y + \text{-- } x \\ 14 \end{array} | \\
 | \begin{array}{c} 3 \ 2 \\ \text{-- } y \\ 14 \end{array} \quad \begin{array}{c} 1 \ 2 \\ - y \\ 7 \end{array} \quad \begin{array}{c} 1 \ 2 \\ \text{-- } y \\ 14 \end{array} \quad \begin{array}{c} 1 \\ \text{-- } x y \\ 14 \end{array} | \\
 | \text{-----} | \text{-----} | \text{-----} | \text{-----} | \\
 | \begin{array}{c} 2 \ 1 \ 2 \\ y + \text{-- } x \\ 14 \end{array} \quad \begin{array}{c} 2 \ 1 \ 2 \\ y + \text{-- } x \\ 14 \end{array} \quad \begin{array}{c} 2 \ 1 \ 2 \\ y + \text{-- } x \\ 14 \end{array} \quad \begin{array}{c} 2 \ 1 \ 2 \\ y + \text{-- } x \\ 14 \end{array} | \\
 + \quad \begin{array}{c} 14 \\ 14 \end{array} \quad \begin{array}{c} 14 \\ 14 \end{array} \quad \begin{array}{c} 14 \\ 14 \end{array} \quad \begin{array}{c} 14 \\ 14 \end{array} +
 \end{array}$$

Type: Union(SquareMatrix(4, Fraction Polynomial Fraction Integer), ...)

Time: 0.05 (EV) + 0.02 (OT) + 0.01 (GC) = 0.08 sec

quaternion(1,2,3,q)

Cannot convert from type SquareMatrix(4,Fraction  
Polynomial Fraction Integer) to Fraction Polynomial

```
+ x          +
| - 1 2 3 |
| y      |
|      |
|      x |
| - 1 - - 3 2 |
|      y |
|      |
|      x |
| - 2 3 - - 1 |
|      y |
|      |
|      x |
| - 3 - 2 1 - |
+          y +
```

## A brief history

- The [Scratchpad](#) projects were developed since 1971 by [IBM](#) under the direction of [Richard Jenks](#).
- [Scratchpad 2](#) was considered as a research platform for developing new ideas in computational mathematics.
- From 1991, [ALDOR](#), an extension language for [Scratchpad 2](#) was developed under the lead of [Stephen Watt](#).
- In 1990s, the [Scratchpad 2](#) project was renamed to [AXIOM](#). It was sold to the [Numerical Algorithms Group \(NAG\)](#) and became a commercial system.
- [AXIOM](#) was withdrawn from the market in October, 2001.
- [NAG](#) agreed to release [AXIOM](#) and [ALDOR](#) as **free software**.
- Today, the efforts of the [AXIOM](#) community are coordinated by [Tim Daly](#). Those around [ALDOR](#) are organized at [ORCCA](#).

## More recently

- Released under [Modified BSD License](#) in Sept. 2002.
- [GNU Common Lisp port](#) from Codemist Common Lisp
- [Literate rewrite](#); Algebra code released in Sept. 2003
- [Electronic Book](#) released in April 2004
- [Graphics](#) released in Sept. 2004
- [Browser](#) released in Dec. 2004
- [Feature complete](#) release in Feb. 2005
- Introducing [fast arithmetic and parallelism](#) in AXIOM/ALDOR (Yuzhen Xie, Xin Li and Akpodigha Filatei)



## A few data

- Approximately **3.6 million** lines of source, including documentation.
- The whole system takes about **8 hours** to build on a 1Ghz machine.
- Free and Open Source software maintained on several servers:
  - **savannah**, by courtesy of the **Free Software Foundation**,
  - **sourceforge**, by courtesy of the **Open Source Technology Group**,
  - **axiom-developer.org**, by Tim Daly, with support from the **Center for Algorithms and Interactive Scientific Software (CAISS)** at the City College of New York.
- AXIOM has **82 registered developers** and 22 of them have write access to the source code.
- Direct contributions, over **35 years**, of approximately 164 people.

## AXIOM contributors

Cyril Alberga

George Andrews

Henry Baker

David R. Barton

Fred Blair

Peter A. Broadbery

Florian Bundschuh

Robert Caviness

David V. Chudnovsky

Don Coppersmith

Gary Cornell

Timothy Daly Sr.

Jean Della Dora

Roy Adler

Stephen Balzac

Gerald Baumgartner

Mark Botch

Martin Brock

William Burge

Bruce Char

Gregory V. Chudnovsky

George Corliss

Timothy Daly Jr.

Michael Dewar

Richard Anderson

Yurij Baransky

Gilbert Baumslag

Alexandre Bouyer

Manuel Bronstein

Cheekai Chin

Josh Cohen

Robert Corless

James H. Davenport

Claire DiCrescendo

Sam Dooley  
Brian Dupee

Lionel Ducos  
Dominique Duval

Martin Dunstan

Robert Edwards

Lars Erickson

Richard Fateman  
Brian Ford  
Timothy Freeman

Bertfried Fauser  
Albrecht Fortenbacher  
Korrinn Fu

Stuart Feldman  
George Frances

Marc Gaetano  
Holger Gollan  
Stephen Gortler  
James Griesmer

Rudiger Gebauer  
Teresa Gomez-Diaz  
Johannes Grabmeier  
Vladimir Grinberg

Patricia Gianni  
Laureano Gonzalez-  
Matt Grayson  
Oswald Gschnitzer

Steve Hague  
Martin Hassner

Vilya Harvey  
Henderson

Satoshi Hamaguchi

Pietro Iglio

Richard Jenks

Grant Keady

Bernhard Kutzler

Larry Lambe

Xin Li

Richard Luczak

Camm Maguire

Ian Meikle

Gerard Milmeister

Michael Monagan

Mark Murray

William Naylor

Godfrey Nolan

Tony Kennedy

Frederic Lehobey

Rudiger Loos

Bob McElrath

David Mentre

Mohammed Mobarak

Marc Moreno Maza

C. Andrew Neff

Arthur Norman

Klaus Kusche

Michel Levaud

Michael Lucks

Michael McGettrick

Victor S. Miller

H. Michael Moeller

Scott Morrison

John Nelder

Michael O'Connor

Julian A. Padget

Michel Petitot

Claude Quitte

Norman Ramsey

Jean Rivlin

Philip Santas

Gerhard Schneider

Fritz Schwarz

Jonathan Steinbach

Moss E. Sweedler

James Thatcher

Bill Page

Didier Pinchon

Michael Richardson

Simon Robinson

Alfred Scheerhorn

Martin Schoenert

Nick Simicich

Christine Sundaresan

Eugene Surowitz

Dylan Thurston

Susan Pelzel

Renaud Rioboo

Michael Rothstein

William Schelter

Marshall Schor

William Sit

Robert Sutor

Barry Trager

Themos T. Tsikas

Bernhard Wall

M. Weller

Thorsten Werther

Clifton J. Williamson

Waldemar Wiwianka

David Yun

Richard Zippel

Stephen Watt

Mark Wegman

John M. Wiley

Shmuel Winograd

Knut Wolf

Bruno Zuercher

Juergen Weiss

James Wen

Berhard Will

Robert Wisbauer

Dan Zwillinger

## AXIOM: Genericity

Being able to implement mechanisms for producing new algebraic structures from existing ones:

(1)  $R = Z,$

(2)  $n \longmapsto Z/nZ,$

(3)  $(R, X) \longmapsto R[X],$

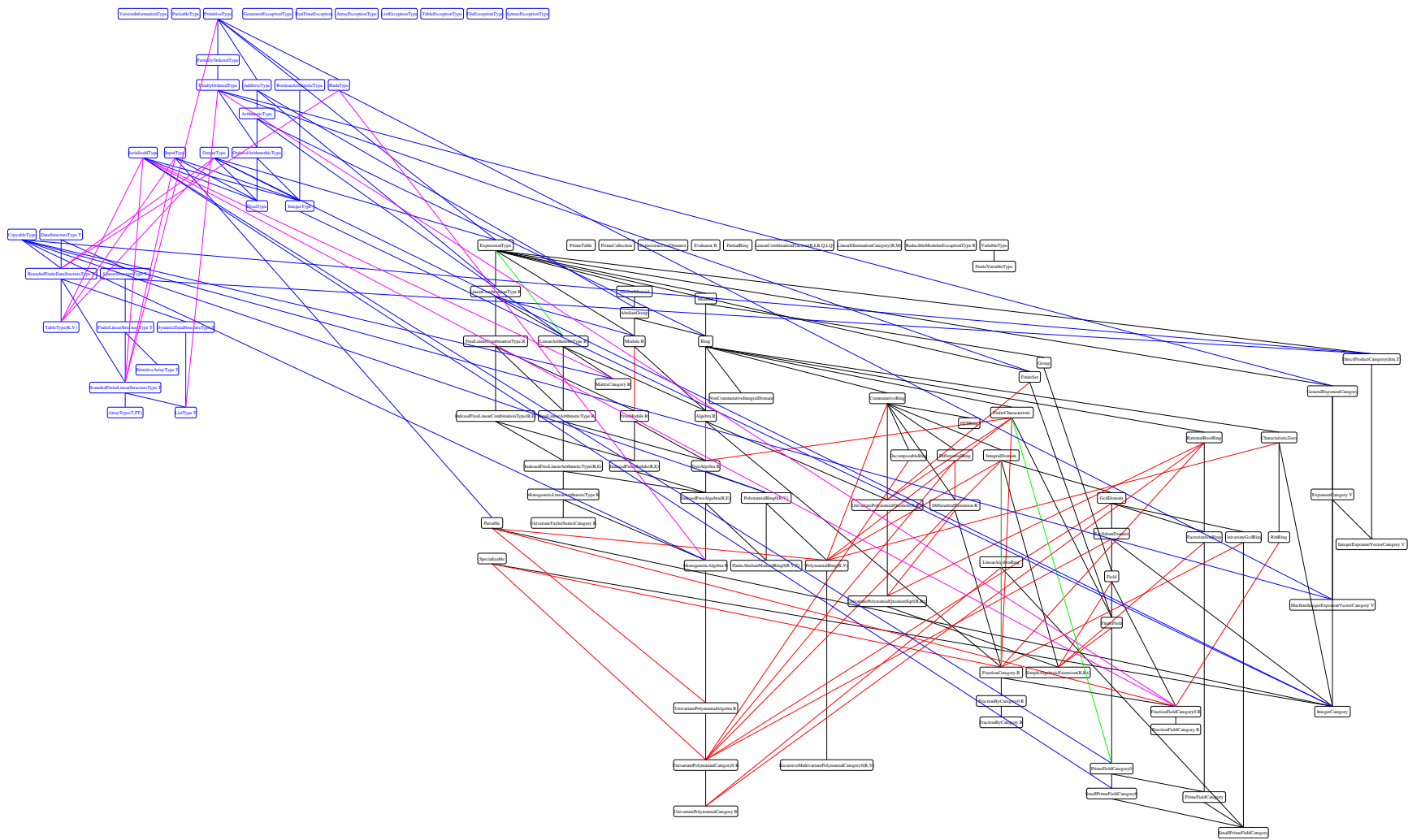
(4)  $(R, p) \longmapsto R/pR$

(5)  $R \longmapsto \mathbf{Fr}(R),$

(6)  $(R_1, R_2) \longmapsto R_1 \times R_2.$

Including conditional statements like:

- If  $R$  is commutative, then  $R[X]$  is commutative.
- If  $R$  is an integral domain, then  $\mathbf{Fr}(R)$  is a field.

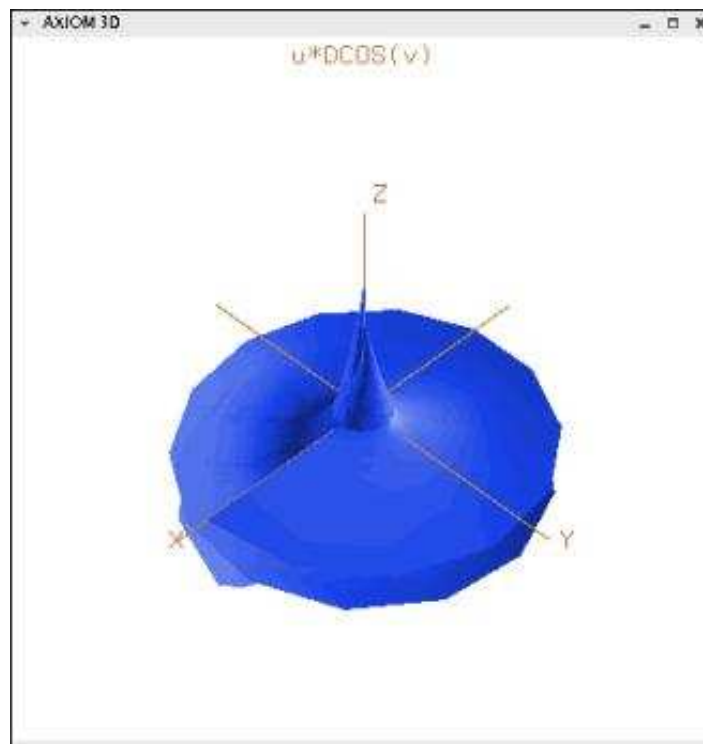




# AXIOM: A laboratory for the mathematician

```

- 1.14. Differential Equations
(1) -> y := operator 'y
(1) y
Type: BasicOperator
(2) -> deq := x**3 * D(y x, x, 3) + x**2 * D(y x, x, 2) - 2 * x * D(y x, x) + 2 * y x = 2 * x**4
(2) x y''' (x) + x y'' (x) - 2xy' (x) + 2y(x) = 2x^4
Type: Equation Expression Integer
(3) -> solve(deq, y, x)
(3)
[particular=  $\frac{x^5 - 10x^3 + 20x + 4}{15x}$ , basis= [ $\frac{2x^3 - 3x + 1}{x}$ ,  $\frac{-1}{x}$ ,  $\frac{3x^2 - 1}{x}$ ]]
Type: Union(Record(particular: Expression Integer, basis: List Expression Integer),...)
(4) -> []
    
```



AXIOM HyperDoc Top Level

**axiom**

What would you like to do?

- **Basic Commands** Solve problems by filling in templates.
- **Reference** Scan on-line documentation for AXIOM.
- **Topics** Learn how to use AXIOM, by topic.
- **Browse** Browse through the AXIOM library.
- **Examples** See examples of use of the library.
- **Settings** Display and change the system environment.
- **WAG Link** Link to WAG Numerical Library.
- **About AXIOM** See some basic information about AXIOM.
- **What's New** Enhancements in this version of AXIOM.

Parametric Surfaces

Graphing a surface defined by  $x = f(u,v)$ ,  $y = g(u,v)$ ,  $z = h(u,v)$ . This page describes the plotting of surfaces defined by the parametric equations of two variables,  $x = f(u,v)$ ,  $y = g(u,v)$ , and  $z = h(u,v)$ , for which the ranges of  $u$  and  $v$  are explicitly defined. The basic draw command for this function utilizes either the uncompiled function or compiled function format and uses the **surface** command to specify the three functions for the  $x$ ,  $y$  and  $z$  components of the surface. The general format for uncompiled functions is:

$$\text{draw}(\text{surface}(f(u,v), g(u,v), h(u,v)), u = a..b, v = c..d)$$

where  $a..b$  and  $c..d$  are segments defining the intervals  $[a,b]$  and  $[c,d]$  over which the parameters  $u$  and  $v$  span. In this case the functions are not compiled until the draw command is executed. Here is an example of a surface plotted using the parabolic cylindrical coordinate system option:

```

draw(surface(u*cos(v),
u*sin(v), v*cos(u)), u=-4..4, v=0..2*pi, coordinates=
    
```

# AXIOM: Free and open source

The screenshot shows the Axiom FrontPage website. The header features the Axiom logo and the title "FrontPage". Below the header, there is a "Welcome to Axiom!" section. The main content area contains several paragraphs of text, including a list of links and a section titled "SOURCE". The browser's address bar shows the URL "http://www.axiom-systems.org/".

The screenshot shows the Axiom MathAction website. The header features the Axiom logo and the title "MathAction". Below the header, there is a "Welcome to MathAction - The Interactive Mathematical Web!" section. The main content area contains several paragraphs of text, including a list of links and a section titled "SOURCE". The browser's address bar shows the URL "http://www.axiom-systems.org/mathaction/".

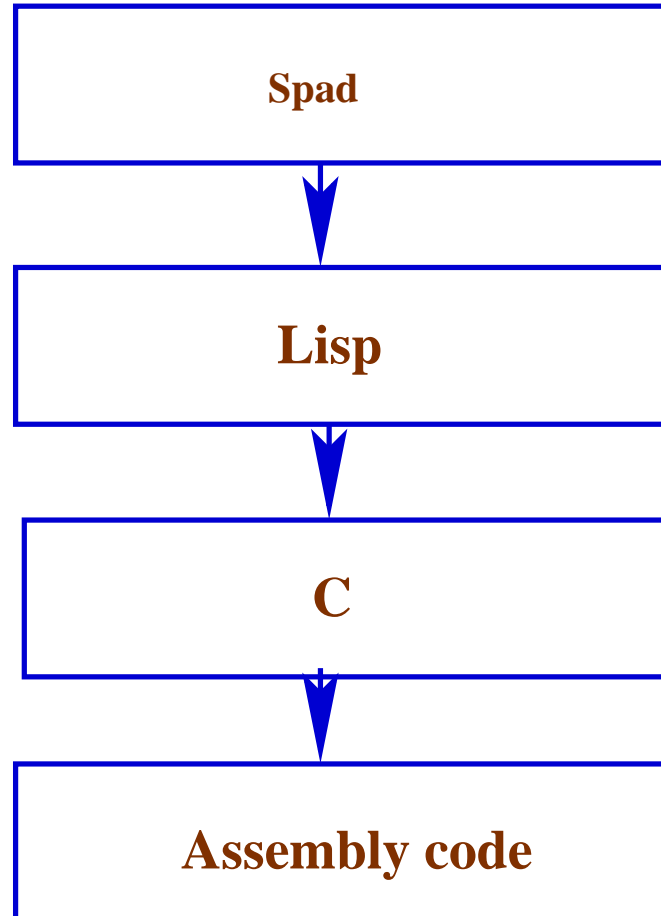
The screenshot shows the Axiom AxiomDownload website. The header features the Axiom logo and the title "AxiomDownload". Below the header, there is a "SOURCE" section. The main content area contains several paragraphs of text, including a list of links and a section titled "SOURCE". The browser's address bar shows the URL "http://www.axiom-systems.org/axiomdownload/".

The screenshot shows the Axiom RosettaStone website. The header features the Axiom logo and the title "RosettaStone". Below the header, there is a "1 Rosetta Translations" section. The main content area contains several paragraphs of text, including a list of links and a section titled "SOURCE". The browser's address bar shows the URL "http://www.axiom-systems.org/rosetta/".

## Literate programming

- A literate program should combine the explanation of the code with the actual code.
- **Programming becomes a task of writing for other people rather than the machine.**
- Literate programming was made famous by Knuth who originally created Web for Pascal programs.
- Norman Ramsey (Harvard) created noweb which is language independent.
- Axiom uses noweb to make **every file into a literate program.**
- As a result all files in Axiom are meant to be document.
- `weave` is a command
  - to extract the latex source code,
  - which can help preparing a technical report.

# AXIOM: a multiple-level programming environment



## The SPAD language

```
ResidueRing(F, Expon, VarSet, FPol, LFPol) : Dom == Body
```

```
where
```

```
F      : Field
```

```
Expon  : OrderedAbelianMonoidSup
```

```
VarSet : OrderedSet
```

```
FPol   : PolynomialCategory(F, Expon, VarSet)
```

```
LFPol  : List FPol
```

```
Dom    == Join(CommutativeRing, Algebra F) with
```

```
  reduce : FPol -> $
```

```
  ++ reduce(f) produces the equivalence class of f in the res
```

```
  coerce : FPol -> $
```

```
  ++ coerce(f) produces the equivalence class of f in the res
```

```
  lift   : $ -> FPol
```

```
  ++ lift(x) return the canonical representative of the equiv
```

```
Body == add
```

```
--representation
```

```
Rep := FPol
```

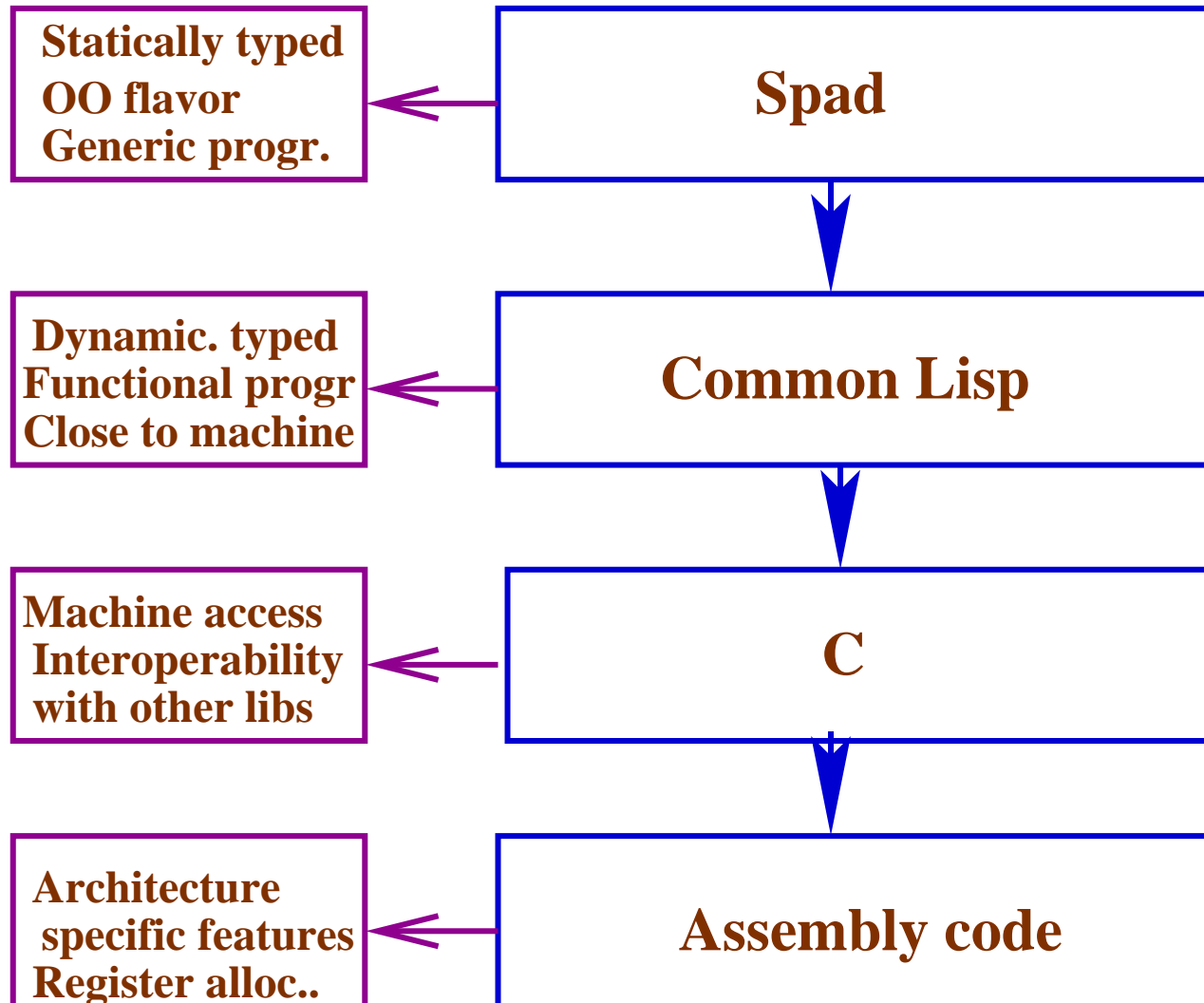
```
import GroebnerPackage(F, Expon, VarSet, FPol)
```

```

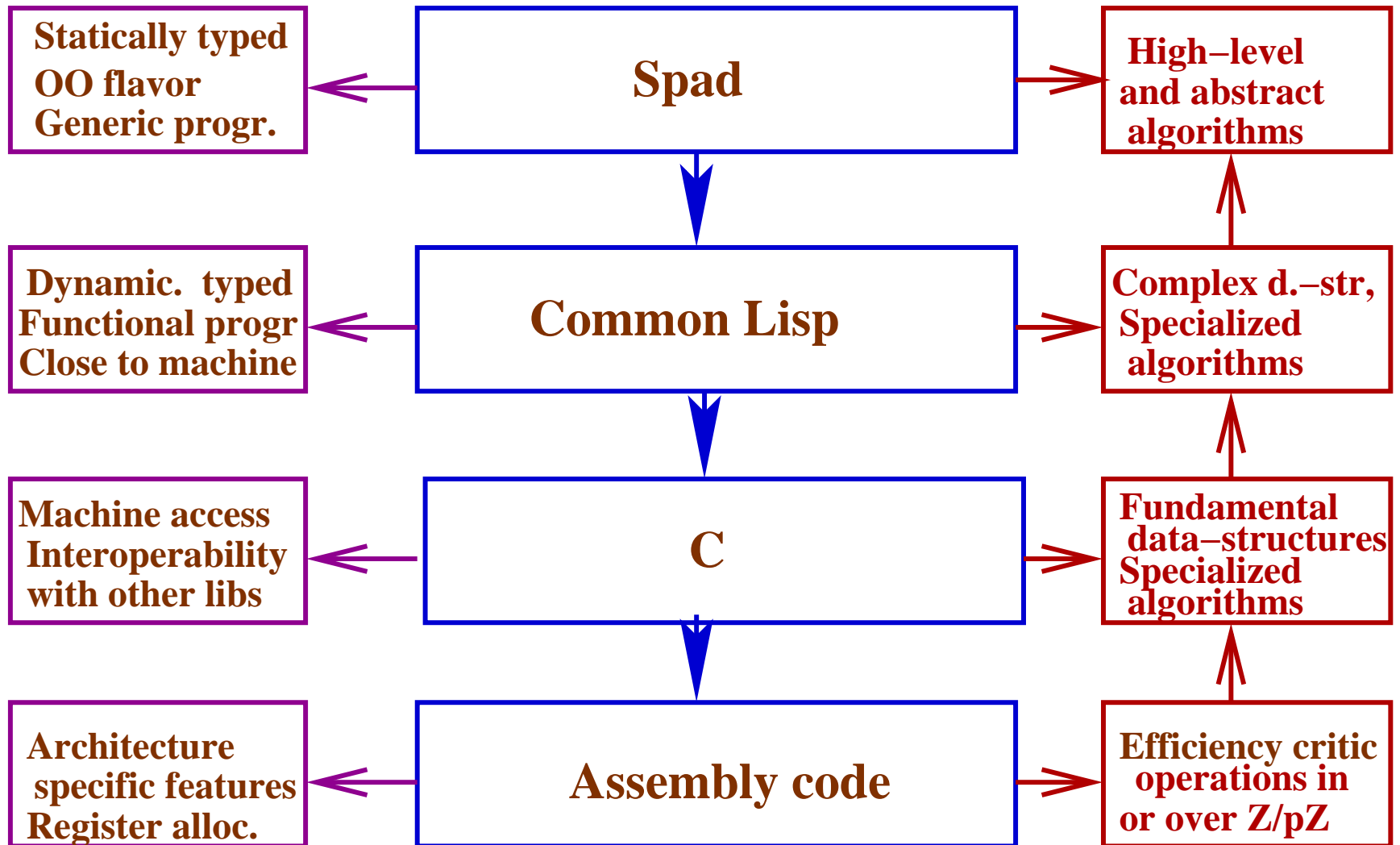
relations:= groebner(LFPol)
relations = [1] => error "the residue ring is the zero ring"
--declarations
x,y: $
--definitions
0 == 0$Rep
1 == 1$Rep
reduce(f : FPol) : $ == normalForm(f,relations)
coerce(f : FPol) : $ == normalForm(f,relations)
lift x == x :: Rep :: FPol
x + y == x +$Rep y
-x == -$Rep x
x*y == normalForm(lift(x *$Rep y),relations)
(n : Integer) * x == n *$Rep x
(a : F) * x == a *$Rep x
x = y == x =$Rep y
characteristic() == characteristic()$F
coerce(x) : OutputForm == coerce(x)$Rep

```

## A multiple-level programming environment (I)



## A multiple-level programming environment (II)

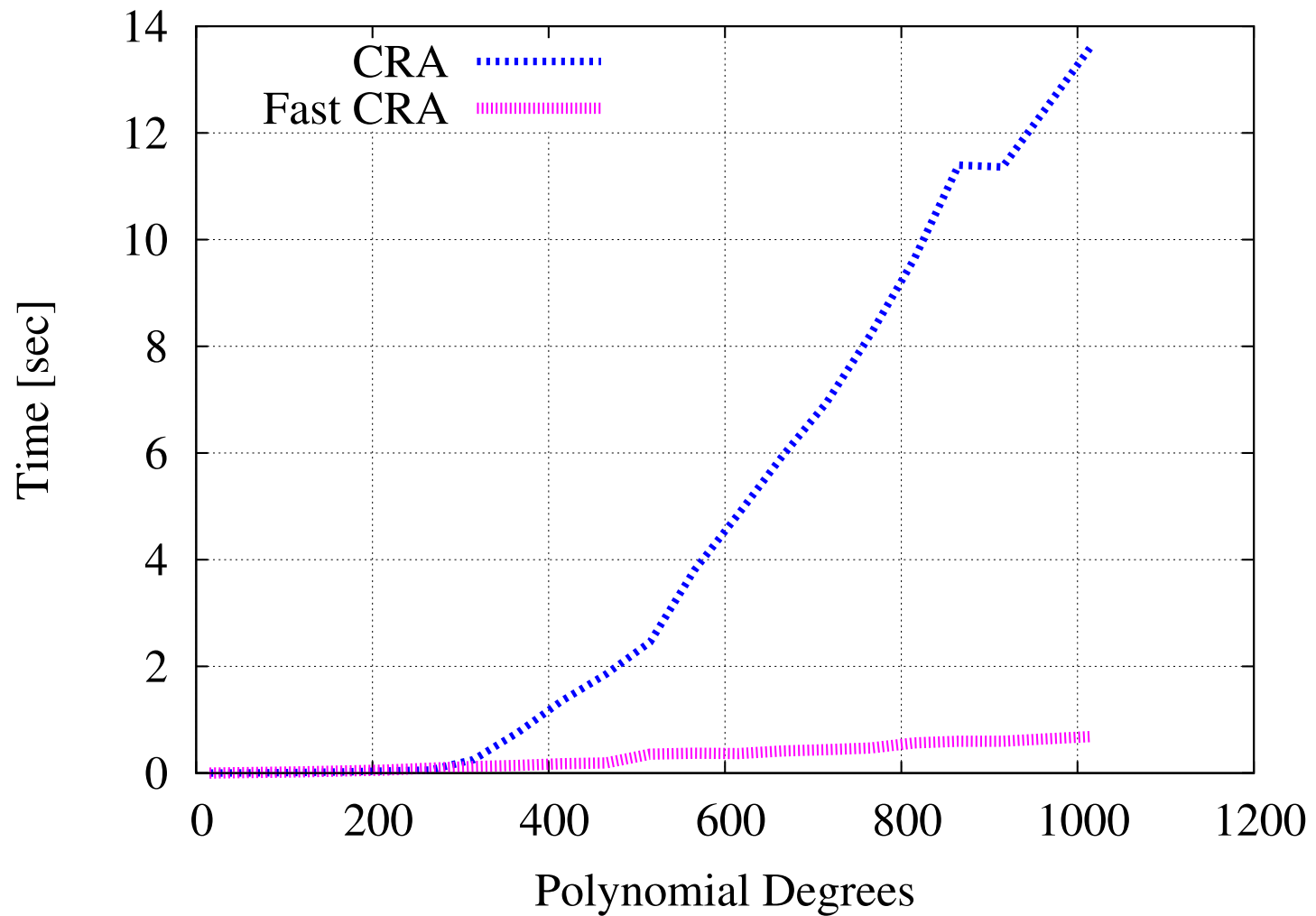




## A few words on ALDOR

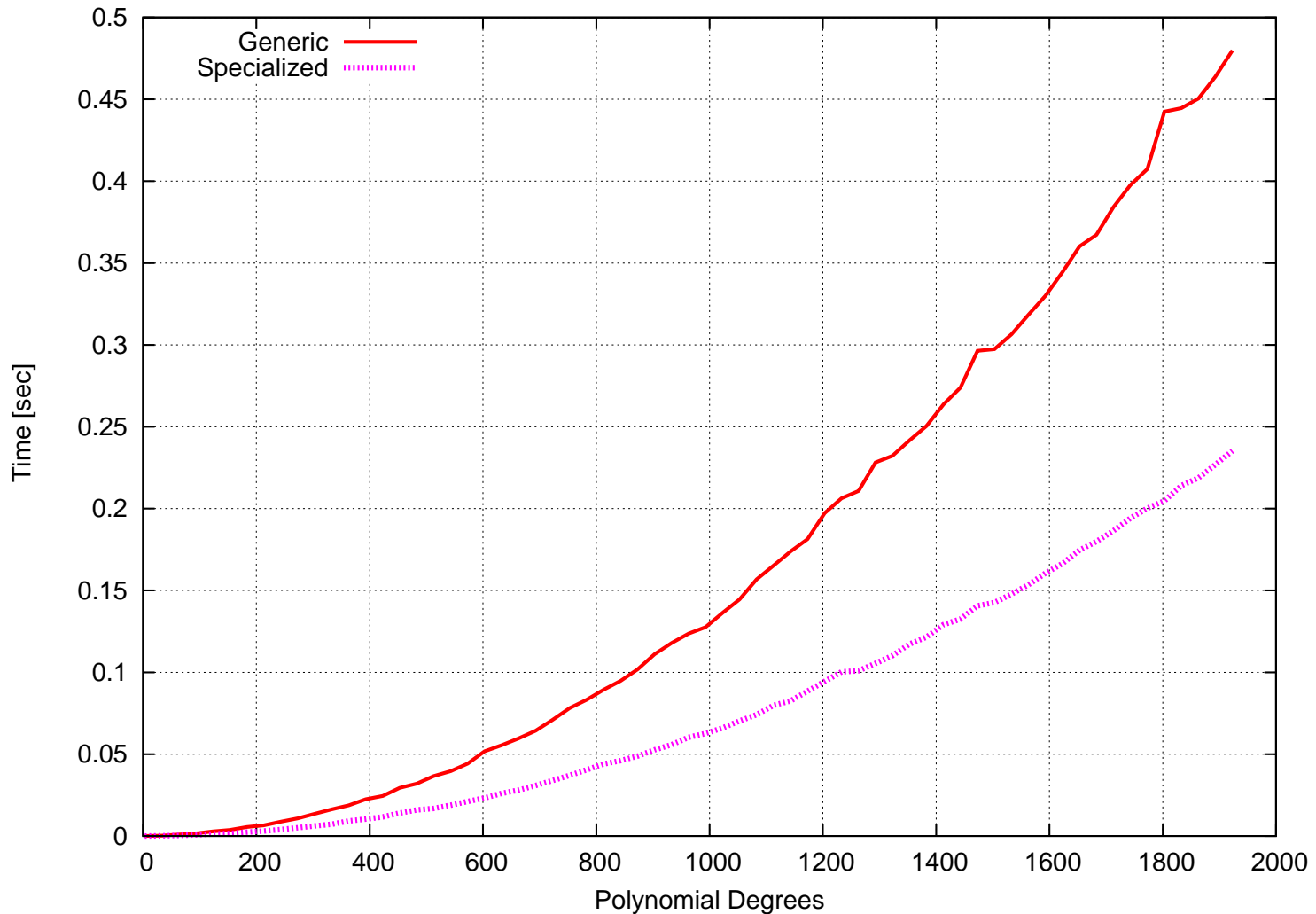
- Some expertise is required in order to interface AXIOM with external packages.
- Developing stand-alone binaries (e.g. servers in a distributed/parallel application) is not purpose of AXIOM.
- ALDOR was initially developed to extend AXIOM via LISP and solve these difficulties.
- ALDOR has also its own libraries:
  - `aldorlib` and `algebra` general purpose libraries for computer algebra,
  - `sumit` for manipulating and solving linear ordinary differential and difference equations and systems
  - `BasicMath` initiated at `NAG` in the 1990's and developed today at `ORCCA`; `BasicMath` supports a **parallel symbolic polynomial system solver**
  - `Calix` for involutive bases in commutative and non-commutative polynomial rings.
  - etc.

## Fast vs standard arithmetic



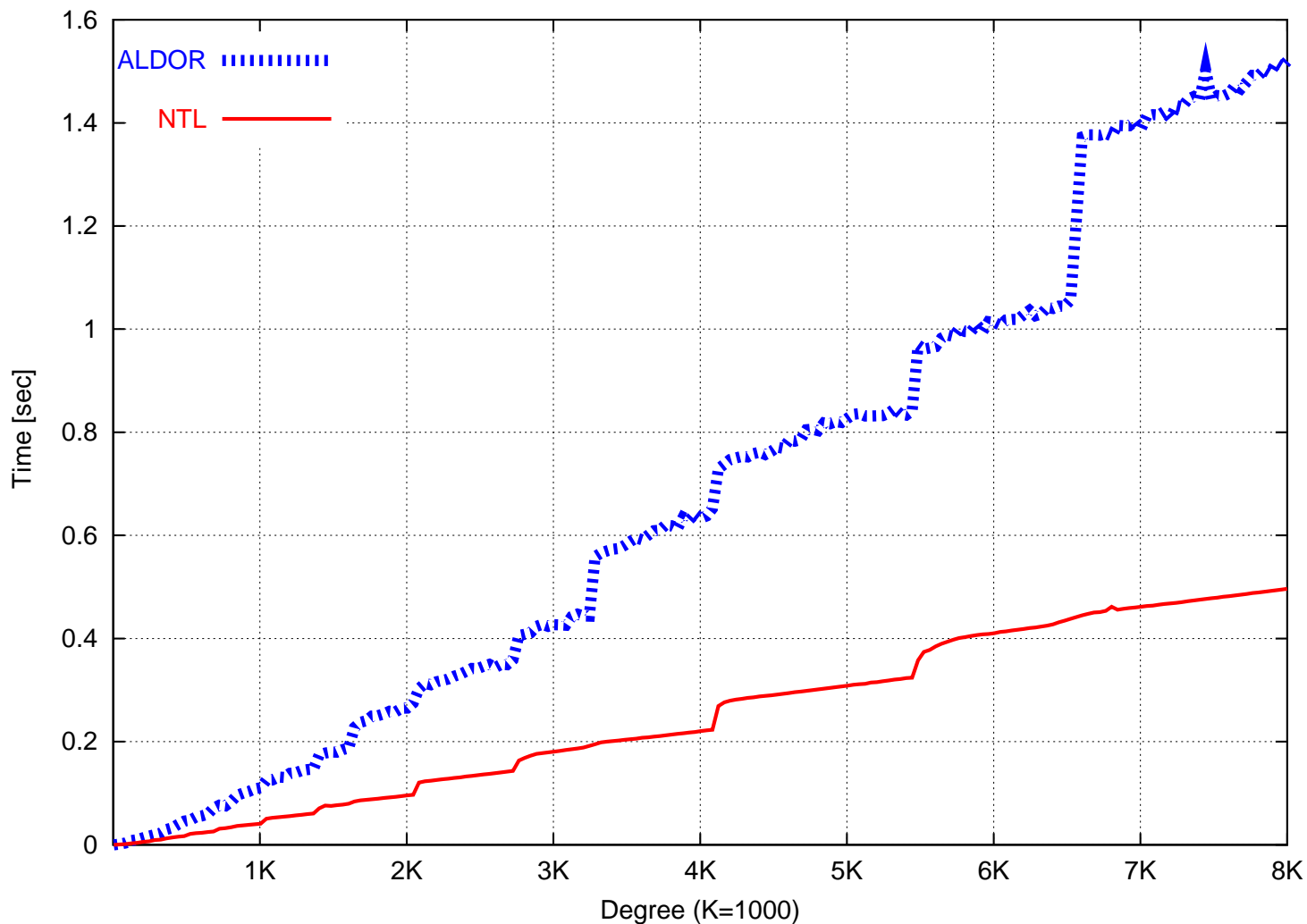
CRA for univariate polynomials in  $\mathbb{Z}/p\mathbb{Z}[x]$  for  $p$  a 27-bit prime in ALDOR.

## We care about a factor of 2



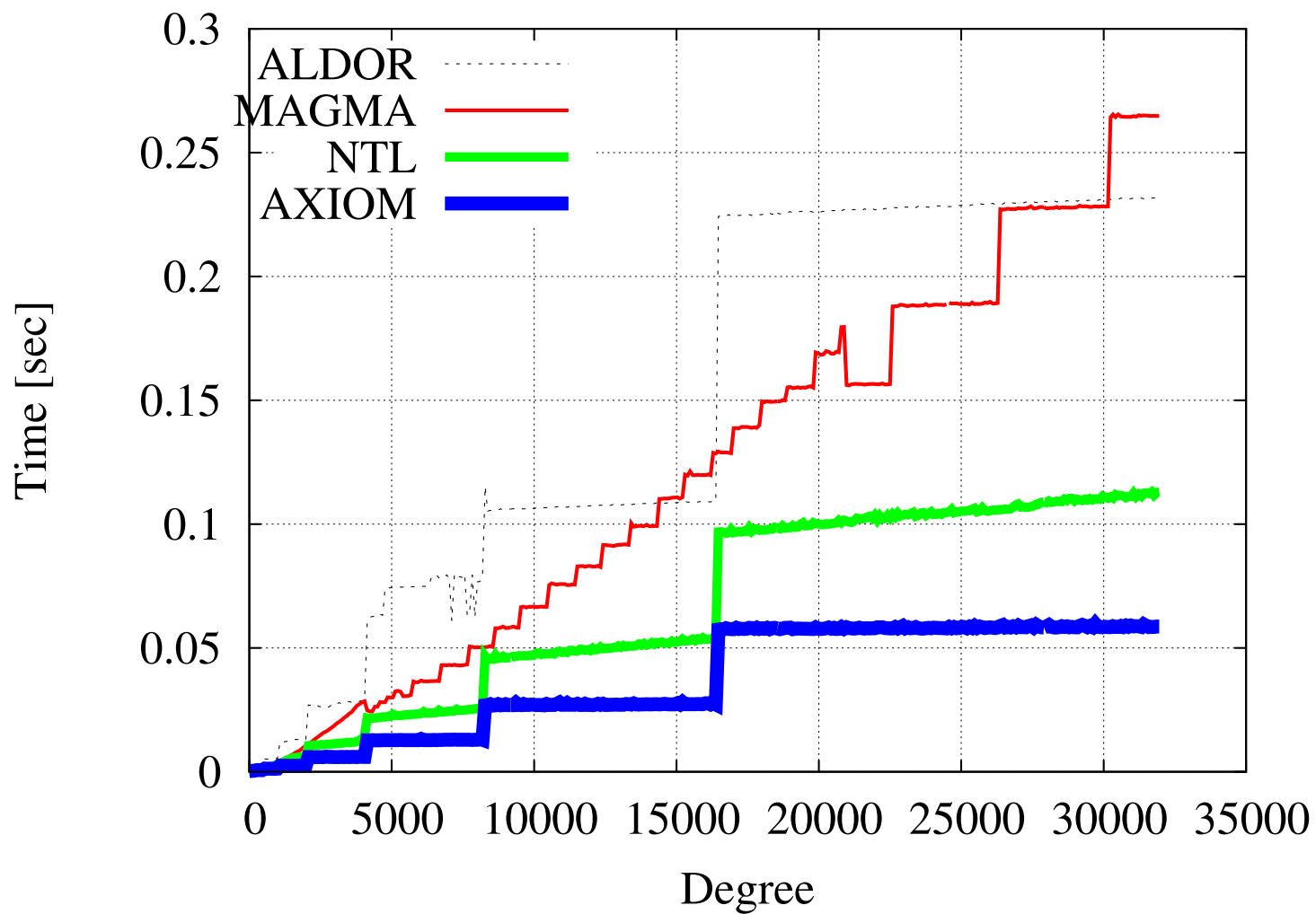
Standard EEA in  $\mathbb{Z}/p\mathbb{Z}[x]$  for a 27-bit prime: generic vs specialized code in ALDOR.

# C-level vs ALDOR implementations



Fast EEA in  $Z/pZ[x]$  for  $p$  a 27-bit prime: generic ALDOR vs specialized C code.

## With AXIOM



FFT-based multiplication in  $\mathbb{Z}/p\mathbb{Z}[x]$  for a 27-bit prime.

## Conclusions and future work

- The AXIOM/ALDOR language remains a beautiful tool for implementing abstract and generic mathematical algorithms.
- An active community of people has turned AXIOM into a free and open computer algebra system.
- AXIOM and ALDOR appear today as platforms for high-performance computing.

## Bibliography

- [AXIOM](http://page.axiom-developer.org) `http://page.axiom-developer.org`
- [ALDOR](http://www.aldor.org) `http://www.aldor.org`
- [GMP](http://swox.com/gmp/) `http://swox.com/gmp/`
- [GNU Common Lisp](http://www.gnu.org/software/gcl) `http://www.gnu.org/software/gcl`
- [MAGMA](http://magma.maths.usyd.edu.au/magma/) `http://magma.maths.usyd.edu.au/magma/`
- [NTL](http://www.shoup.net/ntl) `http://www.shoup.net/ntl`