Parallel Algorithms in Algebraic Geometry

Anne Frühbis-Krüger

Institut für Algebraische Geometrie
Leibniz Universität Hannover

7ECM, Berlin, 20.7.2016

Classical Parallelization Approaches

Modular Approaches

- Chinese remainder,

Classical Parallelization Approaches

Modular Approaches

- Chinese remainder, rational reconstruction

Classical Parallelization Approaches

Modular Approaches

- Chinese remainder, rational reconstruction
- advantage: splits problem into smaller ones,
  avoids intermediate coefficient swell

Classical Parallelization Approaches

Modular Approaches

- Chinese remainder, rational reconstruction
- advantage: splits problem into smaller ones,
  avoids intermediate coefficient swell
- drawback: existence of bad primes,
  decision on correctness of final result

Classical Parallelization Approaches

Modular Approaches

- Chinese remainder, rational reconstruction
- advantage: splits problem into smaller ones,
  avoids intermediate coefficient swell
- drawback: existence of bad primes,
  decision on correctness of final result
- applications in algebraic geometry: e.g. Gröbner Bases, normalization, integral bases

Classical Parallelization Approaches

Inherently Parallel Structure of Task

- relies on knowledge about the problem

Classical Parallelization Approaches

Inherently Parallel Structure of Task

- relies on knowledge about the problem
- advantage: optimally suits the problem

Classical Parallelization Approaches

Inherently Parallel Structure of Task

- relies on knowledge about the problem
- advantage: optimally suits the problem
- drawback: only applicable for certain tasks

Classical Parallelization Approaches

Inherently Parallel Structure of Task

- relies on knowledge about the problem
- advantage: optimally suits the problem
- drawback: only applicable for certain tasks
- applications in algebraic geometry:
  e.g. ADE-singularities in normalization,

Classical Parallelization Approaches

Inherently Parallel Structure of Task

- relies on knowledge about the problem
- advantage: optimally suits the problem
- drawback: only applicable for certain tasks
- applications in algebraic geometry:
  e.g. ADE-singularities in normalization,
  charts in algorithmic desingularization

Classical Parallelization Approaches

Inherently Parallel Structure of Task

- relies on knowledge about the problem
- advantage: optimally suits the problem
- drawback: only applicable for certain tasks
- applications in algebraic geometry:
  e.g. ADE-singularities in normalization,
  charts in algorithmic desingularization

inherently parallel structure need not be obvious

Recall: Jacobian criterion

$I = \langle f_1, \ldots, f_s \rangle \subseteq \mathbb{C}[\underline{x}]$ equidimensional
$X = V(I)$, $\dim(X) = n - c$

Recall: Jacobian criterion

$I = \langle f_1, \ldots, f_s \rangle \subseteq \mathbb{C}[\underline{x}]$ equidimensional
$X = V(I)$, $\dim(X) = n - c$

$J =$ ideal of $c \times c$ minors of Jacobian matrix of I

Recall: Jacobian criterion

$I = \langle f_1, \ldots, f_s \rangle \subseteq \mathbb{C}[\underline{x}]$ equidimensional
$X = V(I)$, $\dim(X) = n - c$

$J =$ ideal of $c \times c$ minors of Jacobian matrix of I

$$\mathrm{Sing}(X) = V(I + J)$$

Recall: Jacobian criterion

$I = \langle f_1, \ldots, f_s \rangle \subseteq \mathbb{C}[\underline{x}]$ equidimensional
$X = V(I)$, $\dim(X) = n - c$

$J =$ ideal of $c \times c$ minors of Jacobian matrix of I

$$\operatorname{Sing}(X) = V(I + J)$$

in a smoothness test:

$$X \text{ non-singular} \Longleftrightarrow \operatorname{Sing}(X) = \emptyset$$

Jacobian criterion in parallel?

time consuming steps:

- computing $\binom{n}{c} \cdot \binom{s}{c}$ minors

Jacobian criterion in parallel?

time consuming steps:

- computing $\binom{n}{c} \cdot \binom{s}{c}$ minors
- testing whether $1 \in \sqrt{I + J}$

Jacobian criterion in parallel?

time consuming steps:

- computing $\binom{n}{c} \cdot \binom{s}{c}$ minors
- testing whether $1 \in \sqrt{I + J}$

structural considerations on parallelization:

- computing a single minor:
  moderately expensive,

Jacobian criterion in parallel?

time consuming steps:

- computing $\binom{n}{c} \cdot \binom{s}{c}$ minors
- testing whether $1 \in \sqrt{I + J}$

structural considerations on parallelization:

- computing a single minor:
  moderately expensive,
  potentially relatively large input and output

Jacobian criterion in parallel?

time consuming steps:

- computing $\binom{n}{c} \cdot \binom{s}{c}$ minors
- testing whether $1 \in \sqrt{I + J}$

structural considerations on parallelization:

- computing a single minor:
  moderately expensive,
  potentially relatively large input and output
- ideal of minors:
  expensive due to **combinatorial** complexity

Jacobian criterion in parallel?

time consuming steps:

- computing $\binom{n}{c} \cdot \binom{s}{c}$ minors
- testing whether $1 \in \sqrt{I + J}$

structural considerations on parallelization:

- computing a single minor:
  moderately expensive,
  potentially relatively large input and output
- ideal of minors:
  expensive due to **combinatorial** complexity
- radical membership test:
  Gröbner Basis, expensive,

Jacobian criterion in parallel?

time consuming steps:

- computing $\binom{n}{c} \cdot \binom{s}{c}$ minors
- testing whether $1 \in \sqrt{I + J}$

structural considerations on parallelization:

- computing a single minor:
  moderately expensive,
  potentially relatively large input and output
- ideal of minors:
  expensive due to **combinatorial** complexity
- radical membership test:
  Gröbner Basis, expensive,
  very large input, small output

Hironaka's ν^*-Invariant

Defintion[Hironaka]
$(X, 0) \subset (\mathbb{A}^n_{\mathbb{C}}, 0)$ germ,
$I_{X,0} = \langle f_1, \ldots, f_s \rangle \subset \mathbb{C}\{\underline{x}\}$
$f_1, \ldots, f_s$ SB of $I_{X,0}$, sorted by increasing order

$$\nu^*(X, 0) := (ord_0(f_1), \ldots, ord_0(f_s))$$

Lemma[Hironaka]

$$(X, 0) \text{ non-singular} \Longleftrightarrow \nu^*(X, 0) = (\underbrace{1, \ldots, 1}_{\mathrm{codim}(X)})$$

Hironaka's criterion: computational viewpoint

Comparison of the criteria:

Jacobian criterion computing singular locus

Hironaka's criterion testing non-singularity

Hironaka's criterion: computational viewpoint

Comparison of the criteria:

Jacobian criterion computing singular locus

Hironaka's criterion testing non-singularity

Tasks for Hironaka's criterion:

- locus of order at least 2 (if $= \emptyset$: non-singular)

Hironaka's criterion: computational viewpoint

Comparison of the criteria:

Jacobian criterion computing singular locus

Hironaka's criterion testing non-singularity

Tasks for Hironaka's criterion:

- locus of order at least 2 (if $= \emptyset$: non-singular)
- determine hypersurface of maximal contact (**local object**)

Hironaka's criterion: computational viewpoint

Comparison of the criteria:

Jacobian criterion computing singular locus

Hironaka's criterion testing non-singularity

Tasks for Hironaka's criterion:

- locus of order at least 2 (if $= \emptyset$: non-singular)
- determine hypersurface of maximal contact (**local object**)
- descent in ambient dimension for considering next ideal

The Main Algorithm

Input:

- g polynomial
- $I_W = \langle g_1, \ldots, g_r \rangle$ non-singular CI on $D(g)$
- $I_X = \langle f_1, \ldots, f_s \rangle$, $I_W \subseteq I_X$

Output:

- `True` (X non-singular) or `False` (X singular)

1. `if` ($I_W == I_X$ on $D(g)$) `return(True)`
2. `if (not CheckOrder(`I_W,I_X,g`))return(False)`
3. list L =`DescendOneStep(`I_W , I_X , g`)`
4. `for` $(I_U, I_{X|U}, g_U) \in L$ `do`
   - `if (not SmoothnessTest(`$I_U, I_{X|U}, g_U$`))` `return(False)`

The workhorses: CheckOrder

`CheckOrder` – only general idea

- Find regular system of parameters $\underline{y}$ for $W \cap D(g)$ locally

The workhorses: CheckOrder

`CheckOrder` – only general idea

- Find regular system of parameters $\underline{y}$ for $W \cap D(g)$ locally
- compute
  $$J = I_X + \langle \frac{\partial f_i}{\partial y_j} \rangle$$
  (ideal of locus of order at least 2)
- test $V(J) == \emptyset$

The workhorses: CheckOrder

`CheckOrder` – only general idea

- Find regular system of parameters $\underline{y}$ for $W \cap D(g)$ locally
- compute
$$J = I_X + \langle \frac{\partial f_i}{\partial y_j} \rangle$$
(ideal of locus of order at least 2)
- test $V(J) == \emptyset$

in general: no 'global' regular system of parameters

The workhorses: CheckOrder

`CheckOrder` – only general idea

- Find regular system of parameters $\underline{y}$ for $W \cap D(g)$ locally
- compute
$$J = I_X + \langle \frac{\partial f_i}{\partial y_j} \rangle$$
(ideal of locus of order at least 2)
- test $V(J) == \emptyset$

in general: no 'global' regular system of parameters
$\Rightarrow$ covering by principal open subsets

The workhorses: CheckOrder

`CheckOrder` – only general idea

- Find regular system of parameters $\underline{y}$ for $W \cap D(g)$ locally
- compute
$$J = I_X + \langle \frac{\partial f_i}{\partial y_j} \rangle$$
(ideal of locus of order at least 2)
- test $V(J) == \emptyset$

in general: no 'global' regular system of parameters
$\Rightarrow$ covering by principal open subsets
$\Rightarrow$ recombination step

The workhorses: DescendOneStep

`DescendOneStep` – only general idea

- use
$$\bigcap \mathrm{Sing}(f_i) = \emptyset$$
to cover $W \cap D(g)$

The workhorses: DescendOneStep

`DescendOneStep` – only general idea

- use
$$\bigcap \mathrm{Sing}(f_i) = \emptyset$$
to cover $W \cap D(g)$

Hironaka's descend in ambient dimension

The workhorses: DescendOneStep

`DescendOneStep` – only general idea

- use
$$\bigcap \mathrm{Sing}(f_i) = \emptyset$$
to cover $W \cap D(g)$

Hironaka's descend in ambient dimension

- covering by principal open subsets, splitting up the problem

The workhorses: DescendOneStep

`DescendOneStep` – only general idea

- use
$$\bigcap \mathrm{Sing}(f_i) = \emptyset$$
to cover $W \cap D(g)$

Hironaka's descend in ambient dimension

- covering by principal open subsets, splitting up the problem
- no recombination step possible

A Hybrid Approach

expensive parts of Hironaka style approach:

- descent in ambient dimension
- differentiation w.r.t. regular system of parameters

A Hybrid Approach

expensive parts of Hironaka style approach:

- descent in ambient dimension
- differentiation w.r.t. regular system of parameters

hybrid approach:

- descend several dimension steps (number of steps tunable)
- use Jacobian criterion for $(I_U, I_{X|U}, g_U)$

The timings

Tabelle : Timings and Memory Usage

		smoothtst			hybrid			Jacobian	
	smooth	time	'parallel'	mem	time	'parallel'	mem	time	mem
$\mathcal{I}_1(6)$	yes	0.24	0.07	0.22	0.18	0.05	0.22	2.5	34
$\mathcal{I}_1(7)$	yes	0.60	0.17	0.24	0.35	0.10	0.22	310	1300
$\mathcal{I}_1(8)$	yes	0.86	0.22	0.32	0.64	0.15	0.23	—	$>$ 20000
$\mathcal{I}_2(3)$	yes	0.22	0.04	0.14	0.08	0.02	0.14	0.05	4.2
$\mathcal{I}_2(4)$	yes	160	9.1	27	40	4.9	190	15	450
$\mathcal{I}_2(5)$	yes	—	—	—	1200	14	510	4000	16000
$\mathcal{I}_3(4)$	no	0.30	0.05	0.22	0.15	0.03	0.22	1.0	8.6
$\mathcal{I}_3(5)$	yes	0.72	0.10	0.22	0.38	0.07	0.22	110	300
$\mathcal{I}_3(6)$	yes	1.3	0.18	0.22	0.83	0.11	0.22	2500	2300
$\mathcal{I}_4(6,3)$	no	0.02	0.01	0.14	0.02	0.01	0.14	3.1	34
$\mathcal{I}_4(7,3)$	no	0.04	0.02	0.14	0.04	0.01	0.14	1600	4000
$\mathcal{I}_4(7,4)$	no	0.10	0.02	0.14	0.10	0.02	0.14	4300	4000

Hironaka's smoothness test:

Hironaka's smoothness test:

- inherently parallel through use of charts

Hironaka's smoothness test:

- inherently parallel through use of charts
- powerful where combinatorics impedes Jacobian criterion
- hybrid approach most beneficial

Hironaka's smoothness test:

- inherently parallel through use of charts
- powerful where combinatorics impedes Jacobian criterion
- hybrid approach most beneficial

Applications:
whereever explicit smoothness tests are necessary

Hironaka's smoothness test:

- inherently parallel through use of charts
- powerful where combinatorics impedes Jacobian criterion
- hybrid approach most beneficial

Applications:
whereever explicit smoothness tests are necessary

In general:

- parallel methods based on charts well-known in algebraic geometry
- up to now rarely exploited for parallel algorithms

Thank you