On finding multiplicities of characteristic polynomial factors of black-box matrices

J-G. DUMAS, C. PERNET and B. D. SAUNDERS

Grenoble Univ. France
University of Delaware, USA.

ISSAC 2009, Seoul,
July 31
Graph isomorphism

Problem

$\text{Graph-isomorphism} \in P$?
Graph isomorphism

Problem

$\text{Graph-isomorphism} \in P$?

[Audenaert, & al. 2007]: the spectrum of a symmetric power of the graph determines its isomorphism class???
Graph isomorphism

Problem

Graph-isomorphism ∈ P ?

[Audenaert, & al. 2007]: the spectrum of a symmetric power of the graph determines its isomorphism class ???

Experiments: symmetric powers of families of strongly regular graphs
Graph isomorphism

**Problem**

*Graph-isomorphism* ∈ \( P \)?

[Audenaert, & al. 2007]: the spectrum of a symmetric power of the graph determines its isomorphism class ???

**Experiments:** symmetric powers of families of strongly regular graphs

- \( k = 2 \): wrong ([Godsil, Royle & al. 2006])
- \( k = 3 \): true up to 29 edges (70 cases, \( n = 3654 \))
- \( k = 3 \): true up to 36 edges (36,510 cases, \( n = 7140 \))

Compute characteristic polynomials over \( \mathbb{Z} \)
Graph isomorphism

Problem

$\text{Graph-isomorphism} \in P$?

[Audenaert, & al. 2007]: the spectrum of a symmetric power of the graph determines its isomorphism class???

Experiments: symmetric powers of families of strongly regular graphs

- $k = 2$: wrong ([Godsil, Royle & al. 2006])
- $k = 3$: true up to 29 edges (70 cases, $n = 3654$)
- $k = 3$: true up to 36 edges (36510 cases, $n = 7140$)

Compute characteristic polynomials over $\mathbb{Z}$
Computing the characteristic polynomial

Overview on the main approaches:

**Traces of powers:** Leverrier 1881, Faddeev 59, ... \( O(n^4) \)

⇒ Dense, over a ring, best in parallel

**Determinant expansion:** Samuelson 42, Berkowitz 84 \( O(n^4) \)

⇒ Dense, over a ring
Computing the characteristic polynomial

Overview on the main approaches:

**Traces of powers:** Leverrier 1881, Faddeev 59, ... \(\mathcal{O}(n^4)\)
- Dense, over a ring, best in parallel

**Determinant expansion:** Samuelson 42, Berkowitz 84 \(\mathcal{O}(n^4)\)
- Dense, over a ring

**Elimination based:** Danilevskii 37, Hessenberg 41, ... \(\mathcal{O}(n^3)\)
- Dense, over a field
Computing the characteristic polynomial

Overview on the main approaches:

**Traces of powers:** Leverrier 1881, Faddeev 59, ... $O(n^4)$
⇒ Dense, over a ring, best in parallel

**Determinant expansion:** Samuelson 42, Berkowitz 84 $O(n^4)$
⇒ Dense, over a ring

**Elimination based:** Danilevskii 37, Hessenberg 41, ... $O(n^3)$
⇒ Dense, over a field

**Explicit Krylov:** Keller-Gehrig 85, Giesbrecht 93, ... $O(n^\omega \log n)$
⇒ Dense Black-box, over a field

**Implicit Krylov:** P. & Storjohann 07, ... $O(n^\omega)$
⇒ Dense over a field
Computing the characteristic polynomial

Overview on the main approaches:

**Traces of powers:** Leverrier 1881, Faddeev 59, ... \(\mathcal{O}(n^4)\)

\[\Rightarrow\text{Dense, over a ring, best in parallel}\]

**Determinant expansion:** Samuelson 42, Berkowitz 84 \(\mathcal{O}(n^4)\)

\[\Rightarrow\text{Dense, over a ring}\]

**Elimination based:** Danilevskii 37, Hessenberg 41, ... \(\mathcal{O}(n^3)\)

\[\Rightarrow\text{Dense, over a field}\]

**Explicit Krylov:** Keller-Gehrig 85, Giesbrecht 93, ... \(\mathcal{O}(n^\omega \log n)\)

\[\Rightarrow\text{Dense Black-box, over a field}\]

**Implicit Krylov:** P. & Storjohann 07, ... \(\mathcal{O}(n^\omega)\)

\[\Rightarrow\text{Dense over a field}\]
Black Box linear algebra
Black Box linear algebra

- Matrices viewed as linear operators
- Algorithms based on matrix vector apply only \( \Rightarrow \) cost \( E(n) \)
Black Box linear algebra

- Matrices viewed as linear operators
- Algorithms based on matrix vector apply only $\Rightarrow$ cost $E(n)$

\[
A \in K^{n \times m} \\
v \in K^m \quad \rightarrow \quad Av \in K^n
\]

Structured matrices: Fast apply (e.g. $E(n) = \mathcal{O}(n \log n)$)
Sparse matrices: Fast apply and no fill-in
Black Box linear algebra

- Matrices viewed as linear operators
- Algorithms based on matrix vector apply only $\Rightarrow$ cost $E(n)$

$$A \in K^{n \times m}$$

$\begin{align*}
v \in K^m & \quad \Rightarrow \quad Av \in K^n
\end{align*}$

Structured matrices: Fast apply (e.g. $E(n) = O(n \log n)$)
Sparse matrices: Fast apply and no fill-in

$\Rightarrow$

- Iterative methods
- No access to coefficients, trace, no elimination
- Matrix multiplication $\Rightarrow$ Black-box composition
Black box linear algebra

**Minimal polynomial:** [Wiedemann 86]
- adapts numerical iterative Krylov/Lanczos methods
- $\mathcal{O}(dE(n) + n^2)$ operations

**Rank, Det, Solve:** [Kaltofen & Saunders 90, Chen & Al. 02]
- reduced to minimal polynomial and preconditioners
Black box linear algebra

**Minimal polynomial:** [Wiedemann 86]
⇒ adapts numerical iterative Krylov/Lanczos methods
⇒ $O\left(dE(n) + n^2\right)$ operations

**Rank, Det, Solve:** [Kaltofen & Saunders 90, Chen & Al. 02]
⇒ reduced to minimal polynomial and preconditioners
⇒ $O^\sim(nE(n))$ operations
Black-box characteristic polynomial

Open Problem [Kaltofen 98 Pb. 3]

\[ \text{CharPoly} \in \mathcal{O}(nE(n)) + \mathcal{O}^\sim(n^2) \text{ operations and } \mathcal{O}(n) \text{ memory} \]
Black-box characteristic polynomial

Open Problem [Kaltofen 98 Pb. 3]
$\text{CharPoly in } \mathcal{O}(nE(n)) + \mathcal{O}^\sim(n^2) \ \text{operations and } \mathcal{O}(n) \ \text{memory}$

State of the art:

Eberly 2000 : adaptive in the number $\Phi$ of invariant factors
$\mathcal{O}^\sim(nE(n) + \Phi n^2)$

Villard 2000 : adaptive in the number $\Psi$ of distinct invariant factors
$\mathcal{O}^\sim(n^{1.5}E(n))$
Black-box characteristic polynomial

Open Problem [Kaltofen 98 Pb. 3]

CharPoly in $\mathcal{O}(nE(n)) + \mathcal{O}^\sim(n^2)$ operations and $\mathcal{O}(n)$ memory

State of the art:

Eberly 2000: adaptive in the number $\Phi$ of invariant factors

$\mathcal{O}^\sim(nE(n) + \Phi n^2)$

Villard 2000: adaptive in the number $\Psi$ of distinct invariant factors

$\mathcal{O}^\sim(n^{1.5} E(n))$

Present Contribution:

- algorithms and heuristics efficient in practice
- improving the best complexity by a $\log n$ factor, under a conjectured property
Outline

1. Computing multiplicities
2. Hybrid algorithms
3. Experiments
Outline

1. Computing multiplicities
2. Hybrid algorithms
3. Experiments
Towards a fast heuristic:

\[ \text{MinPoly} \mid \text{Charpoly} \]

\[ \Rightarrow \text{only differ in the multiplicities of irreducible factors.} \]

\[ \text{MinPoly} = \prod_{i} P_{i}^{e_{i}} \]

\[ \text{CharPoly} = \prod_{i} P_{i}^{m_{i}} \text{ with } m_{i} \geq e_{i}, d_{i} = \deg P_{i} \]
Towards a fast heuristic:

MinPoly | Charpoly

⇒ only differ in the multiplicities of irreducible factors.

\[
\begin{align*}
\text{MinPoly} & = \prod_{i} P_i^{e_i} \\
\text{CharPoly} & = \prod_{i} P_i^{m_i} \quad \text{with } m_i \geq e_i, d_i = \deg P_i
\end{align*}
\]

- compute MinPoly
- Factor it
- Determine the multiplicities \( m_i \)
The method of the nullities

Definition

Companion matrix of

\[ P = x^n - a_{n-1}x^{n-1} - \cdots - a_0 : \]

\[
C_P = \begin{bmatrix}
0 & a_0 \\
1 & a_1 \\
. & . & . \\
1 & a_{n-1}
\end{bmatrix}
\]
The method of the nullities

**Definition**

*Companion matrix of*

\[ P = X^n - a_{n-1}X^{n-1} - \cdots - a_0: \]

\[ C_P = \begin{bmatrix}
0 & a_0 \\
1 & a_1 \\
& \ddots & \ddots \\
& & 1 & a_{n-1}
\end{bmatrix} \]

**Property**

\[ P(C_P) = 0 \]
The method of the nullities

**Definition**

*Companion matrix of*

\[ P = X^n - a_{n-1}X^{n-1} - \cdots - a_0: \]

\[ C_P = \begin{bmatrix}
0 & a_0 \\
1 & a_1 \\
\vdots & \vdots \\
1 & a_{n-1}
\end{bmatrix} \]

**Property**

\[ P(C_P) = 0 \]

\[ P_1 \left( \begin{bmatrix} C_{P_0} & C_{P_1} \end{bmatrix} \right) = \begin{bmatrix} * & 0 \end{bmatrix} \]

\[ \Rightarrow \text{nullity} = \text{deg}(P_1) \]
The method of the nullities

Definition

Companion matrix of

\[ P = X^n - a_{n-1}X^{n-1} - \cdots - a_0 : \]

\[ C_P = \begin{bmatrix}
0 & a_0 \\
1 & a_1 \\
\vdots & \vdots \\
1 & a_{n-1}
\end{bmatrix} \]

Property

\[ P(C_P) = 0 \]

\[ P_1 \left( \begin{bmatrix}
C_{P_0} \\
C_{P_1}
\end{bmatrix} \right) = \begin{bmatrix}
* \\
0
\end{bmatrix} \]

\[ \Rightarrow \text{nullity} = \text{deg}(P_1) \]

Property

\[ P(C_{P_k}) \equiv P \left( \begin{bmatrix}
C_P & 1 \\
\vdots & \vdots \\
1 & C_P
\end{bmatrix} \right) \]

\[ = \begin{bmatrix}
0 & 1 \\
\vdots & \vdots \\
0 & 1
\end{bmatrix} \]

\[ \Rightarrow \text{nullity} (P(C_{P_k})) = n - (k-1)\text{deg}(P) \]

\[ \text{nullity} (P^k(C_{P_k})) = n \]
The method of the nullities

**Theorem**

\[ \text{nullity}(P^{e_i}_i(A)) = m_i d_i \]

\[ \Rightarrow m_i = \left( \frac{n - \text{rank}(P^{e_i}_i(A))}{d_i} \right) \]
The method of the nullities

**Theorem**

\[
\text{nullity } (P_i^{e_i}(A)) = m_id_i
\]

\[
\Rightarrow m_i = \left( \frac{n - \text{rank}(P_i^{e_i}(A))}{d_i} \right)
\]

\[
\begin{bmatrix}
C_{(X+1)^3} & C_{(X+2)^5} \\
C_{(X+1)^2} & C_{X+2} \\
C_{X+1} & C_{X+1}
\end{bmatrix}
\quad \xrightarrow{(X+1)^3} \quad
\begin{bmatrix}
0 & C_{(X+2)^5} \\
0 & C_{X+2} \\
0 & C_{X+2}
\end{bmatrix}
\]
The method of the nullities

Theorem

\[ \text{nullity } (P_i^{e_i}(A)) = m_i d_i \]

\[ \therefore m_i = \left( \frac{n - \text{rank}(P_i^{e_i}(A))}{d_i} \right) \]

Characteristics of the method

Cost : \( \mathcal{O}(e_i d_i E(n)) \)

- only for small factors, with small multiplicity \( e_i \)
- still possible to get partial information applying powers \( k < e_i \) of \( P_i \) to \( A \).
The combinatorial search method

Total degree equation:

$$\sum_i d_i m_i = n$$

- Integer programming problem
- Branch & Bound strategy
  - incrementally increase the multiplicity of each factor
  - list every admissible candidate
- Several candidates are possible \(\Rightarrow\) discriminate them one evaluation at a random value: \(\text{CharPoly} \lambda_0 = \det(\lambda_0 I - A)\)
The combinatorial search method

Total degree equation:

\[ \sum_i d_i m_i = n \]

- Integer programming problem
- *Branch & Bound* strategy
  - incrementally increase the multiplicity of each factor
  - list every admissible candidate
- Several candidates are possible \( \Rightarrow \) discriminate them one evaluation at a random value: \( \text{CharPoly} \lambda_0 = \det(\lambda_0 I - A) \)

**Characteristics of the method**

- Mostly efficient with factors of large degree \( d_i \)
- exponential complexity
  \( \Rightarrow \) Experimentally: limited to the 5 largest factors
The index calculus method

\[ \prod_{j=1}^{k} P_j(\lambda)^{m_j} = \det(\lambda I - A) \mod p \]
The index calculus method

\[ \prod_{j=1}^{k} P_j(\lambda)^{m_j} = \det(\lambda I - A) \mod p \]

\[ \sum_{j=1}^{k} \log_g(P_j(\lambda))m_j = \log_g(\det(\lambda I - A)) \mod (p - 1) \]

Taking \( t \lambda_i \)'s at random: \( \Rightarrow t \times k \) linear system in the \( m_j \)

\[
\begin{bmatrix}
\log_g P_1(\lambda_1) & \ldots & \log_g P_k(\lambda_1) \\
\vdots & \ddots & \vdots \\
\log_g P_1(\lambda_t) & \ldots & \log_g P_k(\lambda_t)
\end{bmatrix}
\begin{bmatrix}
m_1 \\
\vdots \\
m_k
\end{bmatrix} =
\begin{bmatrix}
\log_g \det(\lambda_1 I - A) \\
\vdots \\
\log_g \det(\lambda_t I - A)
\end{bmatrix}
\]
The index calculus method

\[ \prod_{j=1}^{k} P_j(\lambda)^{m_j} = \det(\lambda I - A) \mod p \]

\[ \sum_{j=1}^{k} \log g(P_j(\lambda)) m_j = \log g(\det(\lambda I - A)) \mod (p - 1) \]

Taking \( t \) \( \lambda_i \)'s at random: \( \Rightarrow t \times k \) linear system in the \( m_j \)

\[
\begin{bmatrix}
\log g P_1(\lambda_1) & \ldots & \log g P_k(\lambda_1) \\
\vdots & \ddots & \vdots \\
\log g P_1(\lambda_t) & \ldots & \log g P_k(\lambda_t)
\end{bmatrix}
\begin{bmatrix}
m_1 \\
m_2 \\
\vdots \\
m_k
\end{bmatrix} =
\begin{bmatrix}
\log g \det(\lambda_1 I - A) \\
\vdots \\
\log g \det(\lambda_t I - A)
\end{bmatrix}
\]

- If non singular: the unique solution gives the \( m_i \).
- Conjecture: the system is likely to be non singular.
Outline

1. Computing multiplicities
2. Hybrid algorithms
3. Experiments
Hybrid algorithms

Combination according to the predilection domain:

- $d_i e_i$ small: nullities
- $d_i$ large: combinatorial search

remaining cases: index calculus
Hybrid algorithms

Combination according to the predilection domain:

- $d_i e_i$ small : nullities
- $d_i$ large : combinatorial search

remaining cases : index calculus

Imagrement

Multiple candidates treated as multiple RHS for the index calculus system
Further improvements

Villard 2000: \( \text{MinPoly}(A + UV), \text{s.t. } \text{rank}(UV) = k \)
\( \Rightarrow \) \( k \)-th invariant factor
\( \Rightarrow \) partial information on the multiplicities
Further improvements

**Villard 2000**: $\text{MinPoly}(A + UV)$, s.t. $\text{rank}(UV) = k$

$\Rightarrow$ $k$-th invariant factor

$\Rightarrow$ partial information on the multiplicities

Combined with index calculus:

- compute the largest invariant factors by decreasing order
- until only $\sqrt{n}$ unknown multiplicities remain
- solve a $\sqrt{n} \times \sqrt{n}$ index calculus system
Further improvements

Villard 2000: \( \text{MinPoly}(A + UV) \), s.t. \( \text{rank}(UV) = k \)

- \( k \)-th invariant factor
- partial information on the multiplicities

Combined with index calculus:

- compute the largest invariant factors by decreasing order
- until only \( \sqrt{n} \) unknown multiplicities remain
- solve a \( \sqrt{n} \times \sqrt{n} \) index calculus system

- at most \( \sqrt{n} \) invariant factors computed
- \( \mathcal{O} \left( n^{1.5} E(n) \right) \), (saving a log factor)

But non singularity is only conjectured
Computing Charpoly over $\mathbb{Z}$

- compute $P_M = \text{MinPoly over } \mathbb{Z}$
- decompose it into irreducible factors $P_i$
- pick a prime $p$ at random
- compute $P_C = \text{CharPoly mod } p$
- compute the multiplicities of the $P_i \mod p$ in $P_C$. 

---

Introduction Computing multiplicities Hybrid algorithms Experiments Perspectives
Outline

1. Computing multiplicities
2. Hybrid algorithms
3. Experiments
## Experiments

<table>
<thead>
<tr>
<th>Matrix</th>
<th>EX1</th>
<th>EX2</th>
<th>EX3</th>
<th>EX4</th>
<th>EX5</th>
</tr>
</thead>
<tbody>
<tr>
<td>(n): dimension</td>
<td>560</td>
<td>560</td>
<td>2600</td>
<td>2600</td>
<td>6545</td>
</tr>
<tr>
<td>(d): (\deg(P_{\min}))</td>
<td>54</td>
<td>103</td>
<td>1036</td>
<td>1552</td>
<td>2874</td>
</tr>
<tr>
<td>(\omega): sparsity</td>
<td>15.6</td>
<td>15.6</td>
<td>27.6</td>
<td>27.6</td>
<td>45.2</td>
</tr>
<tr>
<td>(\mathbb{Z})-Minpoly</td>
<td>0.11s</td>
<td>0.26s</td>
<td>117s</td>
<td>260s</td>
<td>5002s</td>
</tr>
<tr>
<td>(\mathbb{Z}[X]) factorize</td>
<td>0.02s</td>
<td>0.07s</td>
<td>9.4</td>
<td>18.15</td>
<td>74.09s</td>
</tr>
<tr>
<td>Nullity/comb.</td>
<td>3.37s</td>
<td>5.33s</td>
<td>33.2s</td>
<td>30.15s</td>
<td>289s</td>
</tr>
<tr>
<td>Total</td>
<td>3.51s</td>
<td>5.66s</td>
<td>159.4s</td>
<td>308.1s</td>
<td>5366s</td>
</tr>
<tr>
<td>Index calc.</td>
<td>3.46s</td>
<td>4.31s</td>
<td>64.0s</td>
<td>57.0s</td>
<td>647s</td>
</tr>
<tr>
<td>Total</td>
<td>3.59s</td>
<td>4.64s</td>
<td>190.4s</td>
<td>336.4s</td>
<td>5641s</td>
</tr>
</tbody>
</table>

Pentium4 (x86 3.2 GHz; 1 Gb)
## Experiments

<table>
<thead>
<tr>
<th>Matrix</th>
<th>n</th>
<th>$\omega$</th>
<th>dense</th>
<th>null-comb</th>
<th>index</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A$</td>
<td>300</td>
<td>1.9</td>
<td>0.32s</td>
<td>0.08s</td>
<td>0.07s</td>
</tr>
<tr>
<td>$AA^T$</td>
<td>300</td>
<td>2.95</td>
<td>0.81s</td>
<td>0.12s</td>
<td>0.12s</td>
</tr>
<tr>
<td>$B$</td>
<td>600</td>
<td>4</td>
<td>4.4s</td>
<td>1.52s</td>
<td>1.97s</td>
</tr>
<tr>
<td>$BB^T$</td>
<td>600</td>
<td>13</td>
<td>2.15s</td>
<td>3.96</td>
<td>7.48s</td>
</tr>
<tr>
<td>TF12</td>
<td>552</td>
<td>7.6</td>
<td>6.8s</td>
<td>5.53s</td>
<td>5.75s</td>
</tr>
<tr>
<td>mk9b3</td>
<td>1260</td>
<td>3</td>
<td>31.25s</td>
<td>10.51s</td>
<td>177s</td>
</tr>
<tr>
<td>Tref500</td>
<td>500</td>
<td>16.9</td>
<td>65.14s</td>
<td>25.14s</td>
<td>25.17s</td>
</tr>
</tbody>
</table>

Athlon (1.8 GHz; 2 Gb)

dense: $BB \mathbb{Z}$-Minpoly + 1 dense charpoly mod $p$
null-comb: $BB \mathbb{Z}$-Minpoly + nullities & Comb. search
index: $BB \mathbb{Z}$-Minpoly + index calculus
Perspectives

- Conjectured behaviour of the index calculus
- Can these computations provide a certificate for the MinPoly (Wiedemann algorithm is only Monte-Carlo)?
- Block-Wiedemann techniques for computing the k-th invariant factor
Conjectured behaviour of the index calculus

Can these computations provide a certificate for the MinPoly (Wiedemann algorithm is only Monte-Carlo) ?

Block-Wiedemann techniques for computing the k-th invariant factor

Thank you