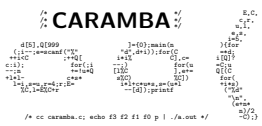
# Paramétrage de CADO-NFS pour le problème du logarithme discret dans les corps finis premiers

**Kevin Trancho**
M1 Informatique

Stage encadré par Pierrick Gaudry

4 septembre 2018





# Au programme de l'exposé

- Paramétrage de CADO-NFS, un problème de recherche.
- Utilisation de mes connaissances en informatique et en mathématiques pour résoudre ce problème.
- Contributions au projet CADO-NFS.

# Le Loria

Né du CRIN (Centre de Recherche en Informatique de Nancy) en 1997.
Unité Mixte de Recherche associant le CNRS, l'INRIA et les universités de Nancy.

- 190 chercheurs / enseignants chercheurs
- 100 doctorants
- 27 équipes de recherche

- Algorithmique, calcul, image et géométrie
- Méthodes formelles
- Réseaux, systèmes et services
- Traitement automatique des langues et des connaissances
- Systèmes complexes, intelligence artificielle et robotique

# Le Loria

Né du CRIN (Centre de Recherche en Informatique de Nancy) en 1997.
Unité Mixte de Recherche associant le CNRS, l'INRIA et les universités de Nancy.

- 190 chercheurs / enseignants chercheurs
- 100 doctorants
- 27 équipes de recherche

- **Algorithmique, calcul, image et géométrie**
- Méthodes formelles
- Réseaux, systèmes et services
- Traitement automatique des langues et des connaissances
- Systèmes complexes, intelligence artificielle et robotique

# L'équipe CARAMBA

Dirigée par Emmanuel Thomé depuis Janvier 2016.
Anciennement CARAMEL dirigée par Pierrick Gaudry.

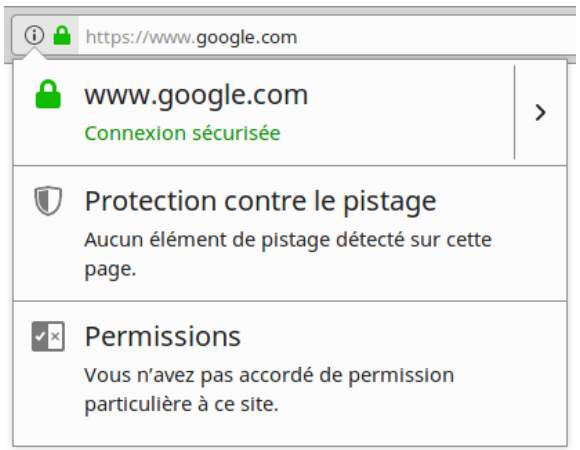
- Attaques Mathématiques
  - Factorisation d'entiers
  - Problème du logarithme discret
- Implémentations
  - Réseaux euclidiens
  - Algèbre linéaire
- Mathématiques
  - Théorie des nombres
  - Courbes elliptiques
  - Systèmes polynomiaux

# L'équipe CARAMBA

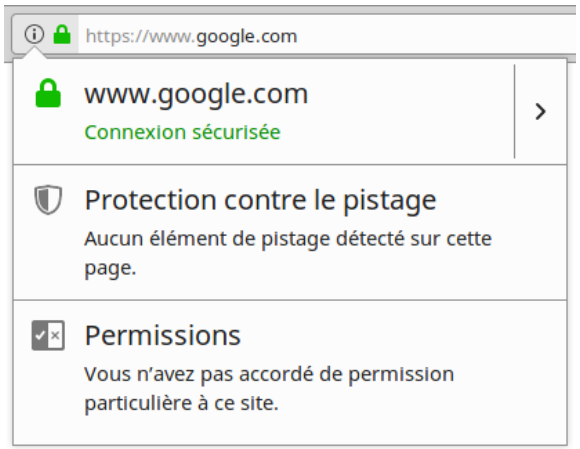
Dirigée par Emmanuel Thomé depuis Janvier 2016.
Anciennement CARAMEL dirigée par Pierrick Gaudry.

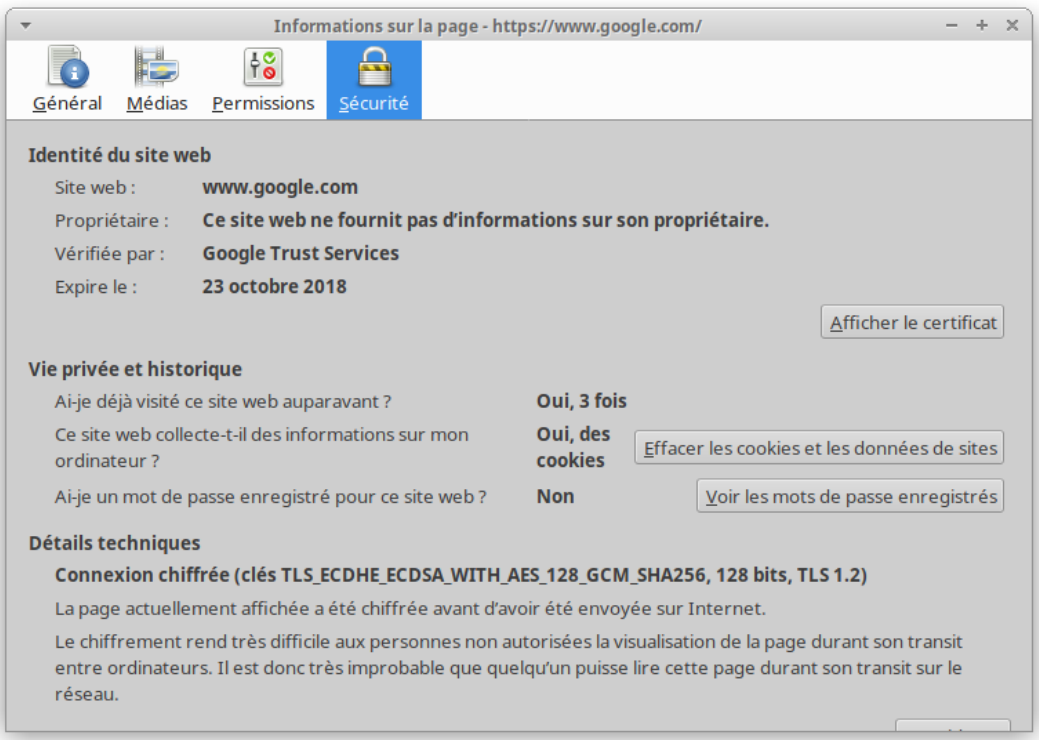
- **Attaques Mathématiques**
  - Factorisation d'entiers
  - **Problème du logarithme discret**
- Implémentations
  - Réseaux euclidiens
  - Algèbre linéaire
- Mathématiques
  - Théorie des nombres
  - Courbes elliptiques
  - Systèmes polynomiaux

# Exemple de contexte cryptographique

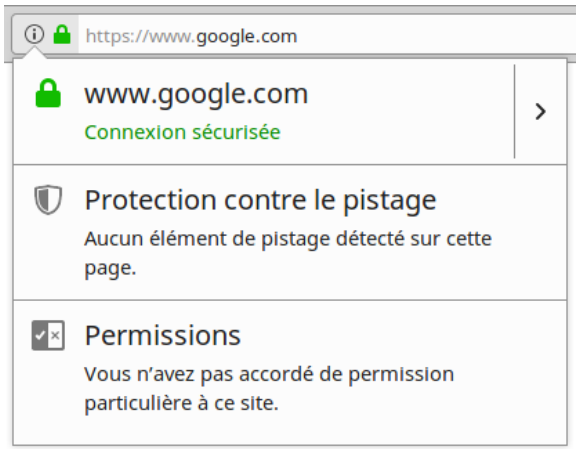


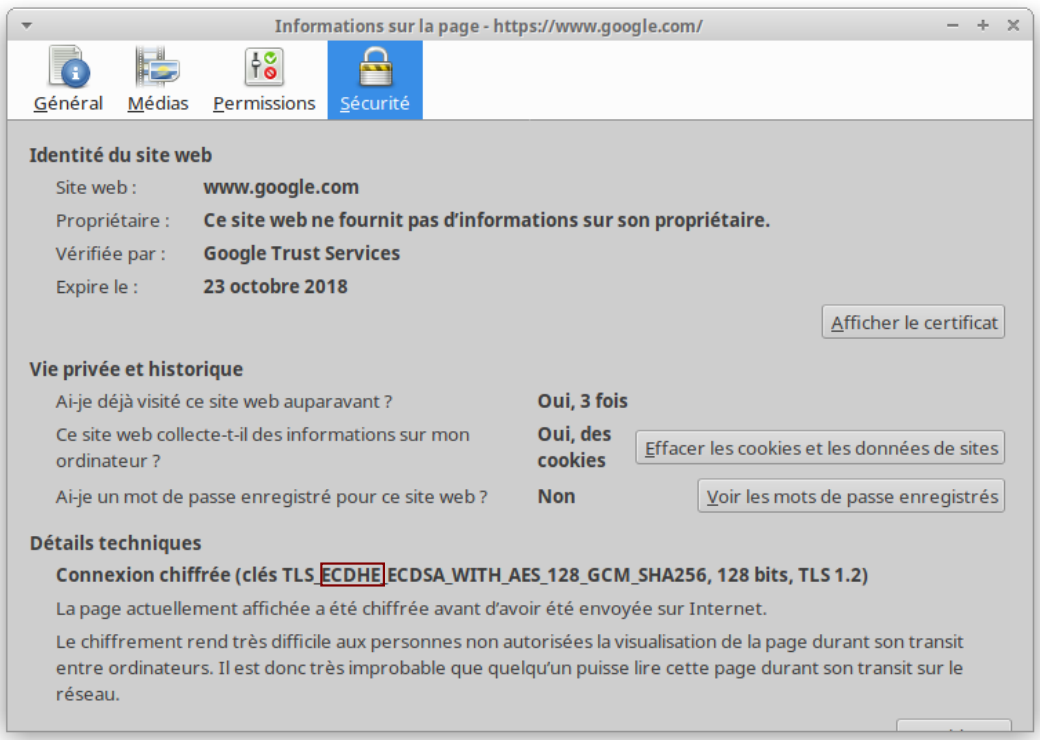
# Exemple de contexte cryptographique



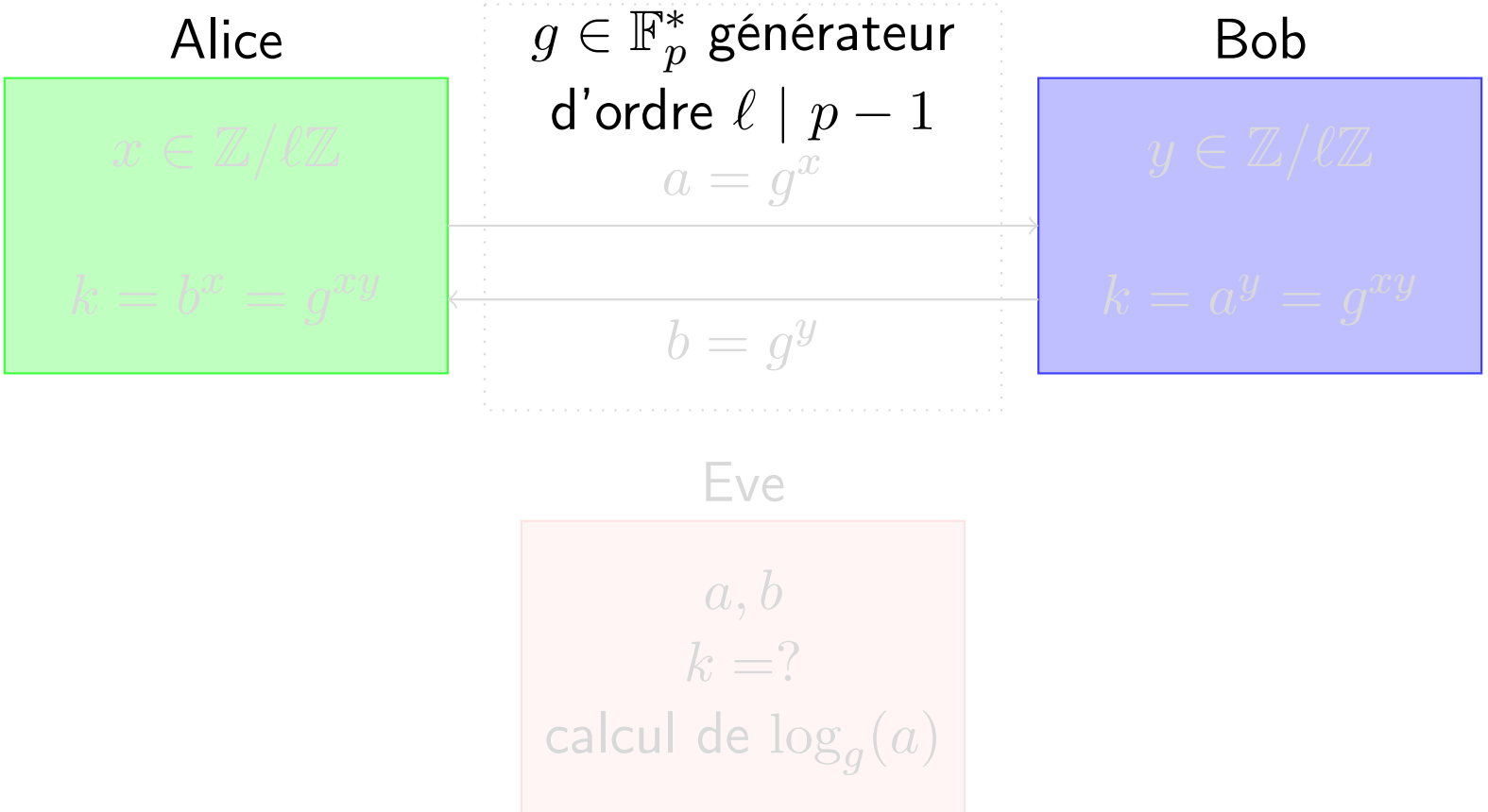



# Exemple de contexte cryptographique

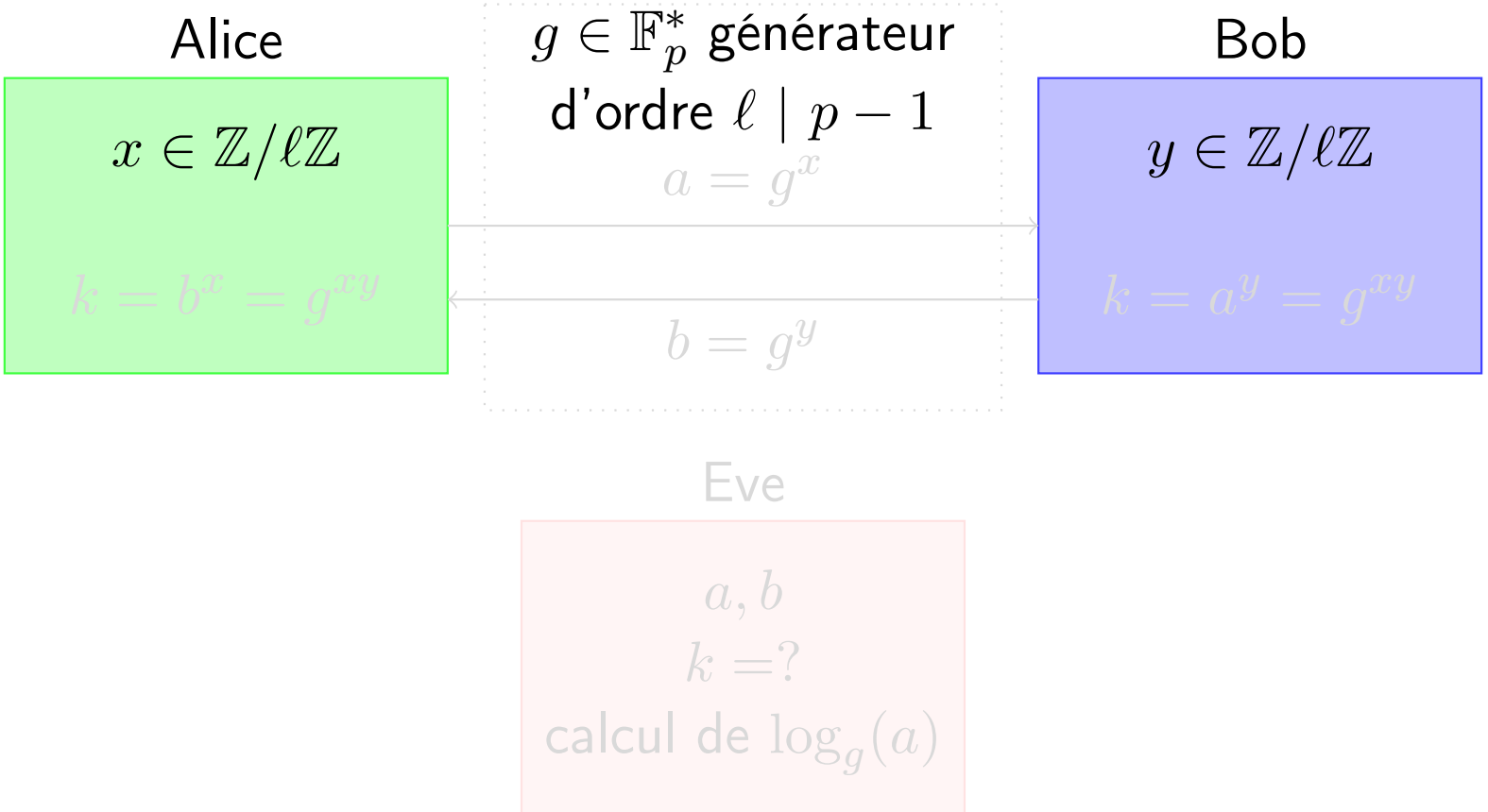


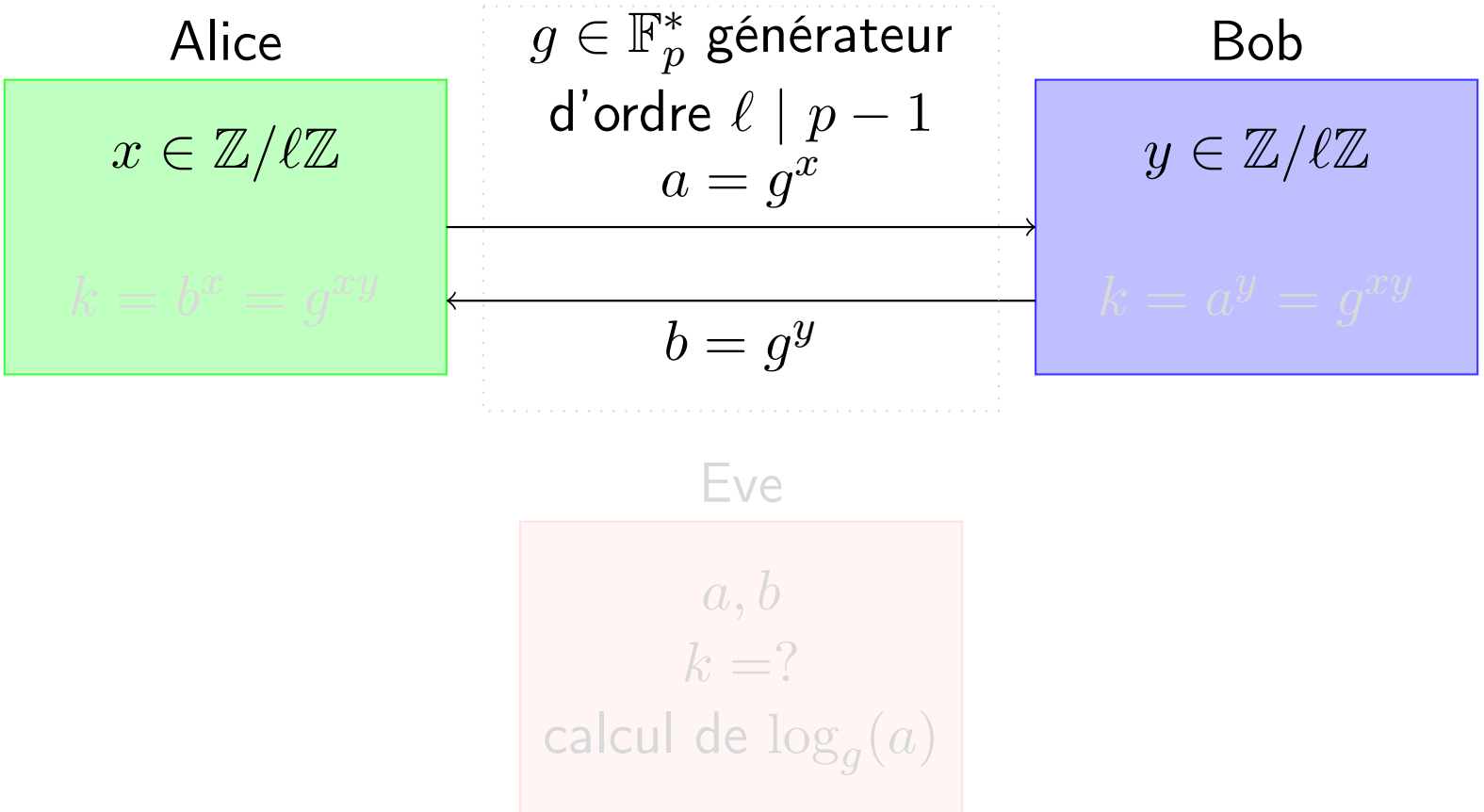
# Un exemple en pratique : l'échange de clés de Diffie-Hellman



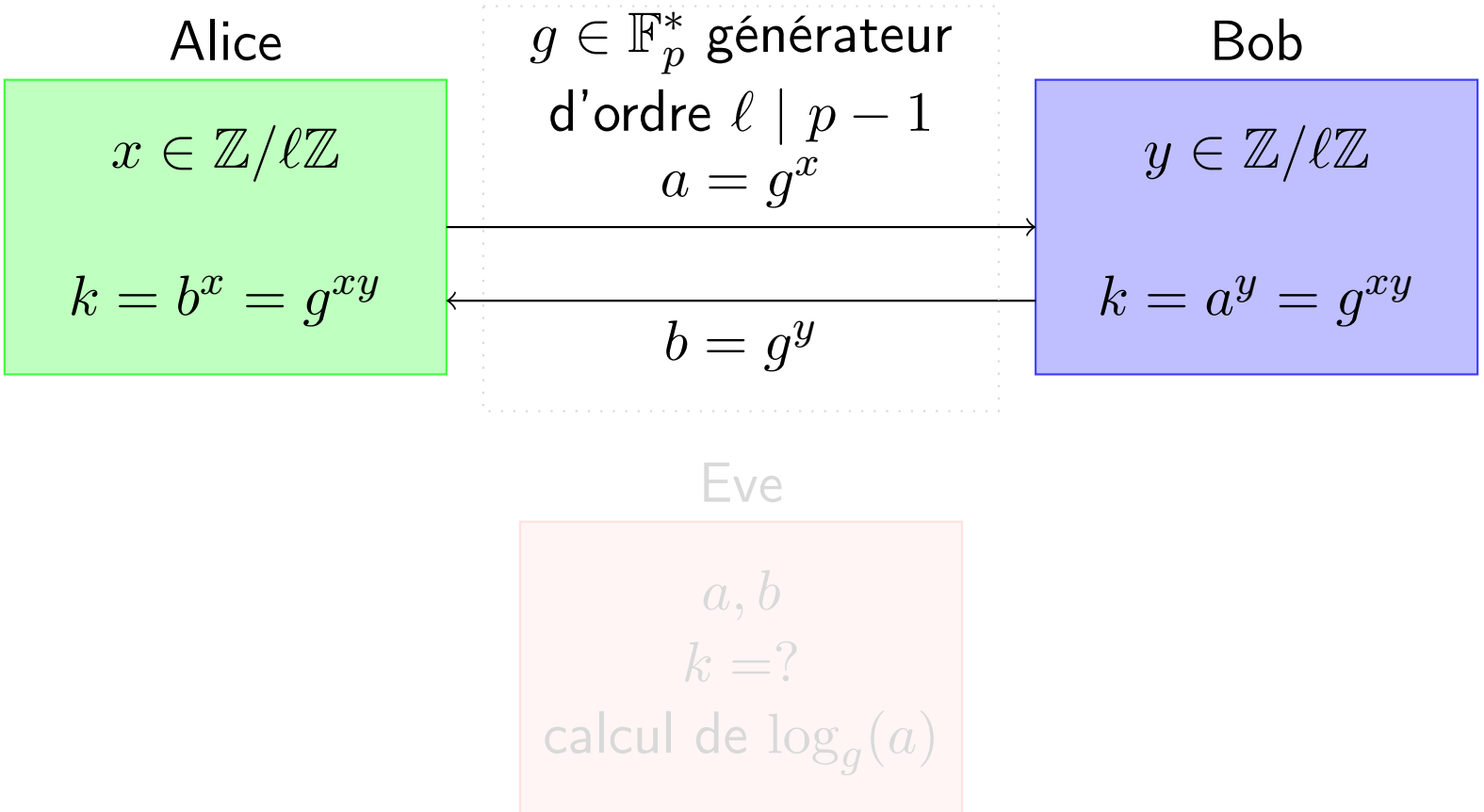
# Un exemple en pratique : l'échange de clés de Diffie-Hellman



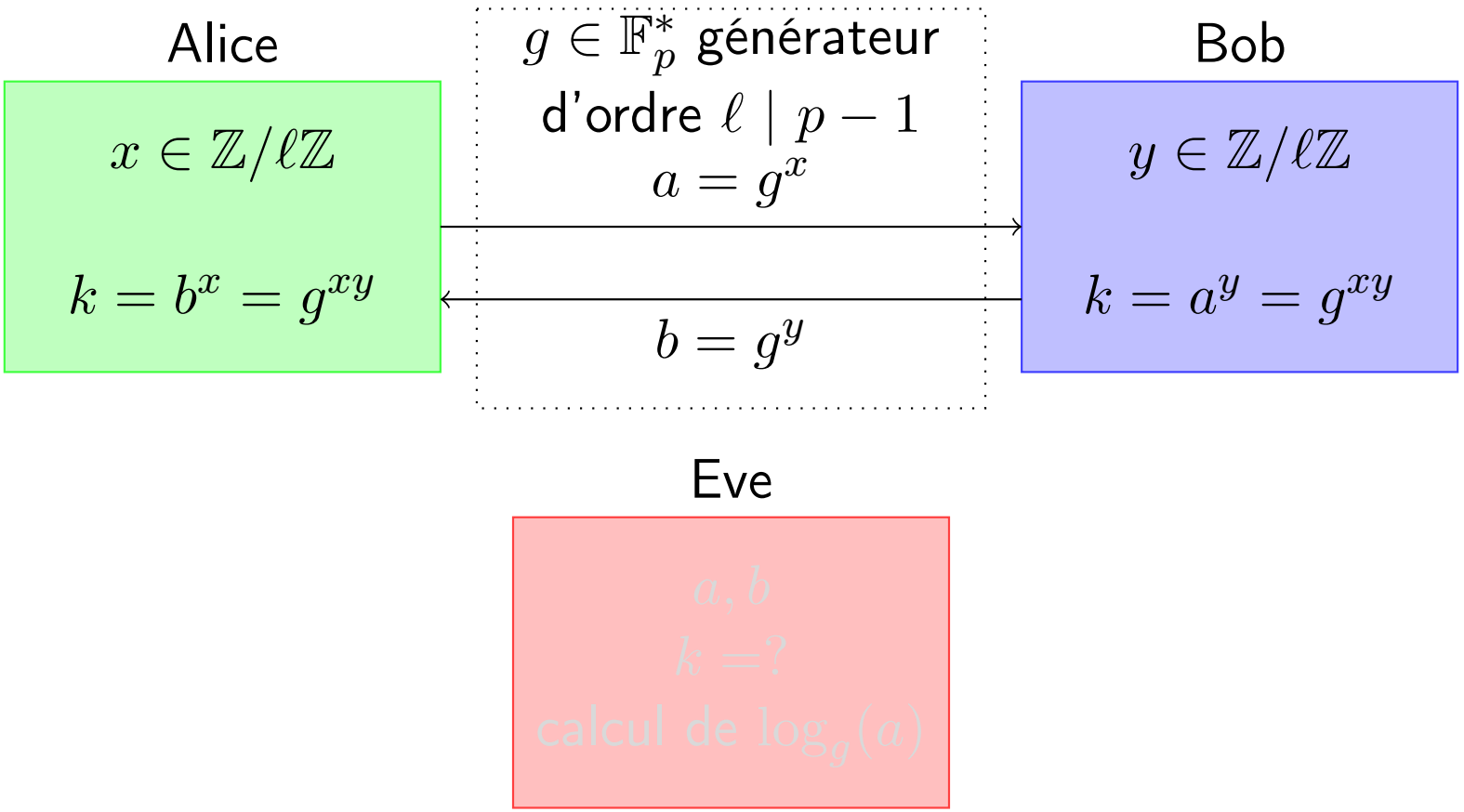
# Un exemple en pratique : l'échange de clés de Diffie-Hellman



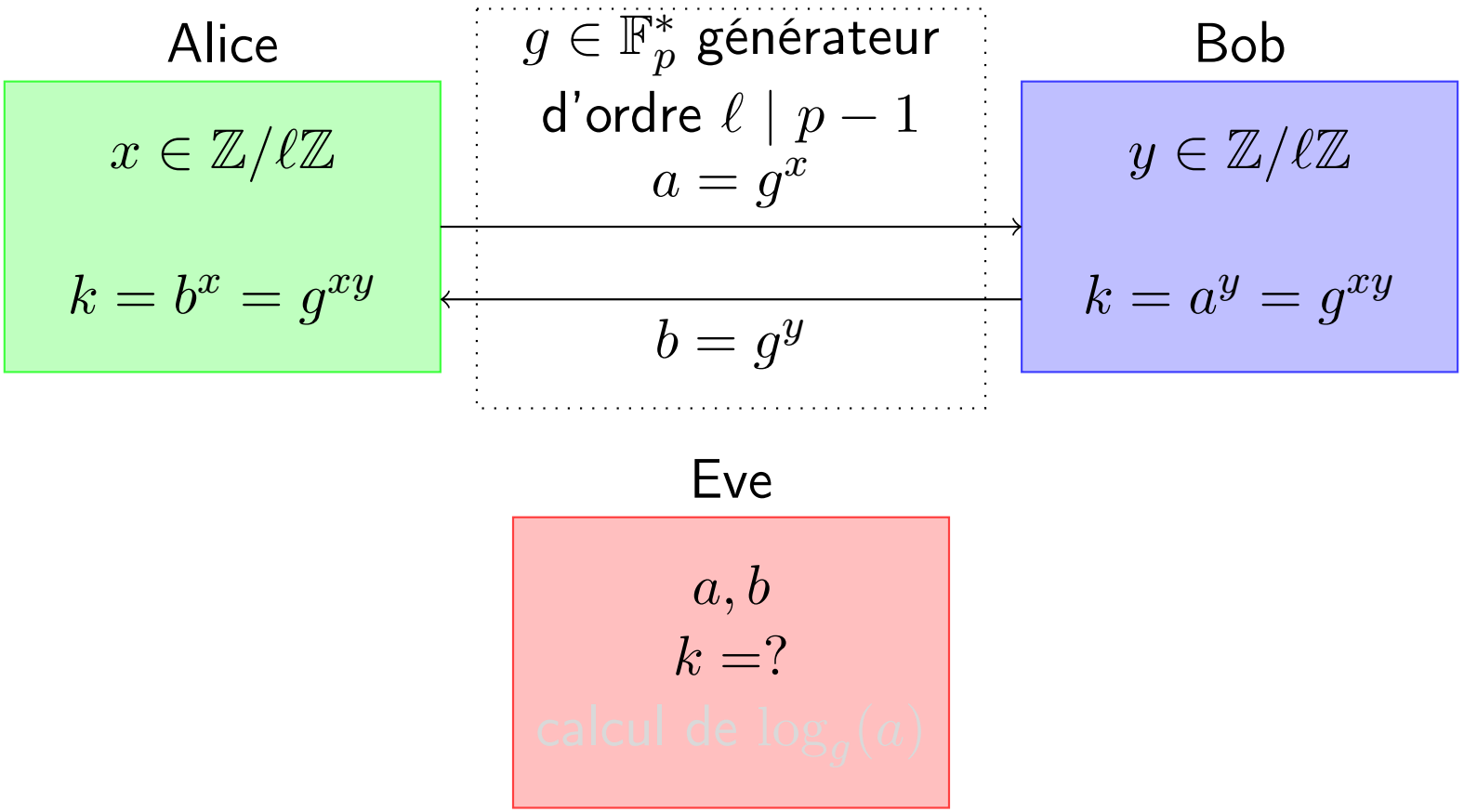
# Un exemple en pratique : l'échange de clés de Diffie-Hellman



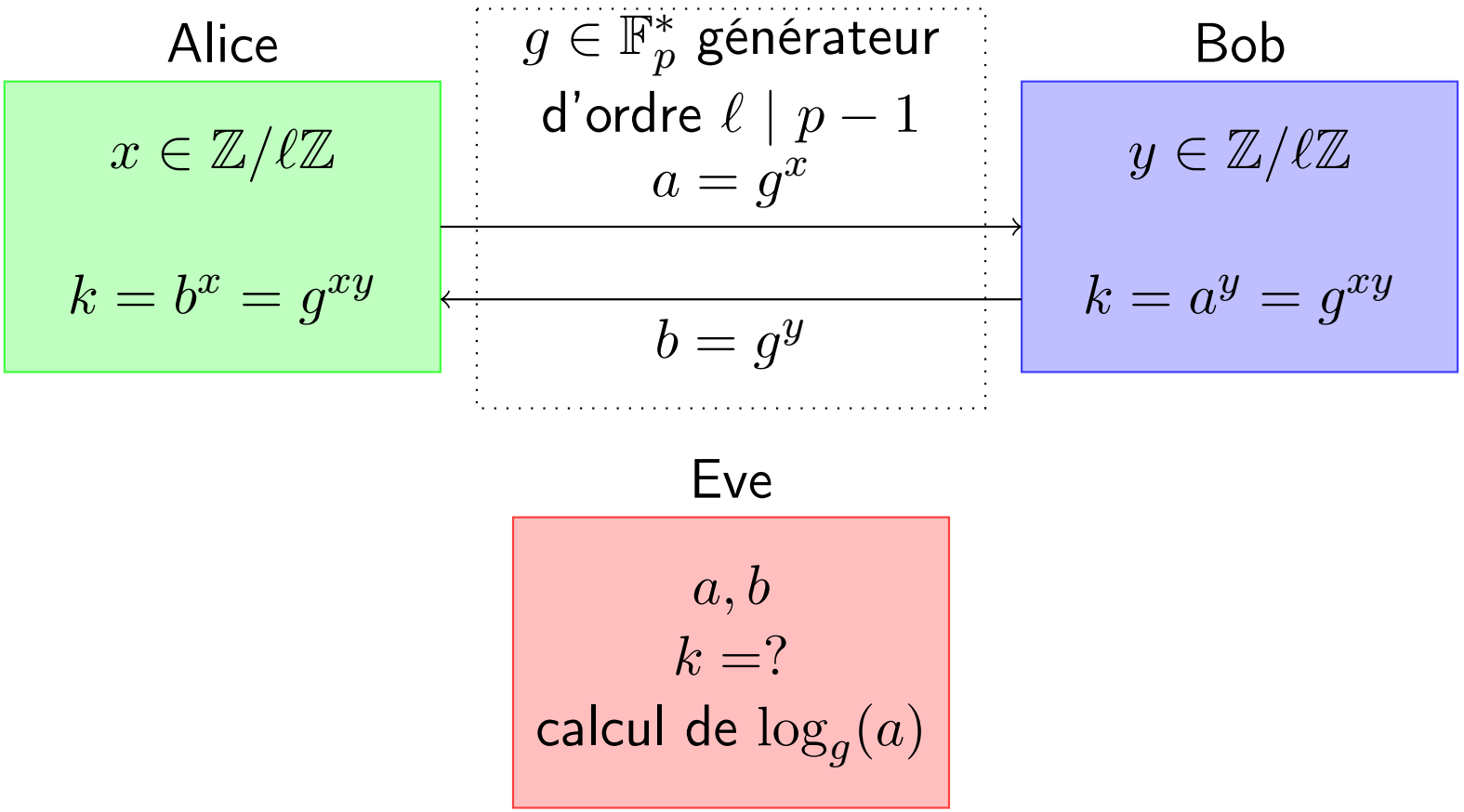
# Un exemple en pratique : l'échange de clés de Diffie-Hellman



# Un exemple en pratique : l'échange de clés de Diffie-Hellman



# Un exemple en pratique : l'échange de clés de Diffie-Hellman



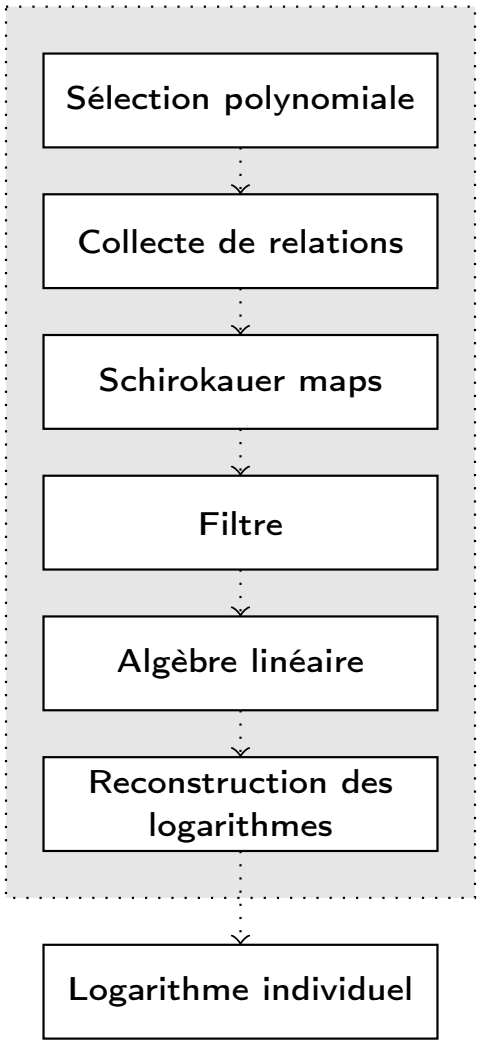
# CADO-NFS

Implémentation robuste de l'algorithme du 'crible algébrique' pour les attaques mathématiques suivantes :

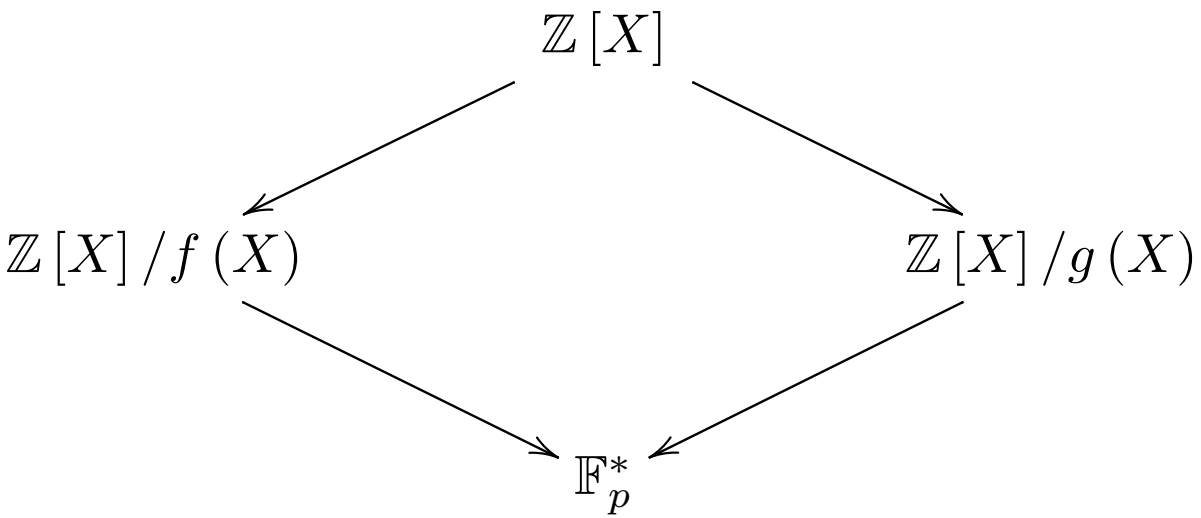
- Factorisation d'entiers.
- Résolution du problème du logarithme discret.

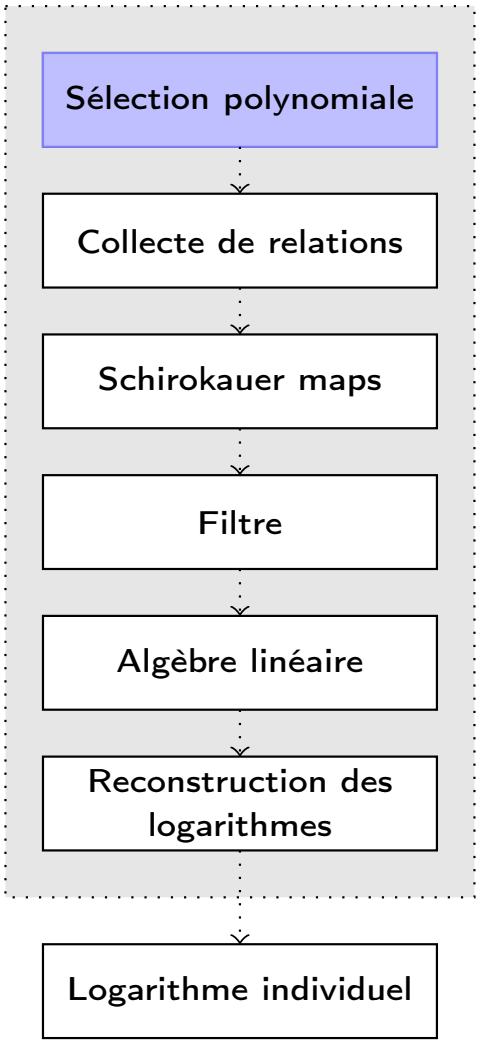
Utilisé à l'origine dans le cas du logarithme discret pour des records (tailles 768 bits ou 1024 bits piégé).

# CADO-NFS : grandes étapes

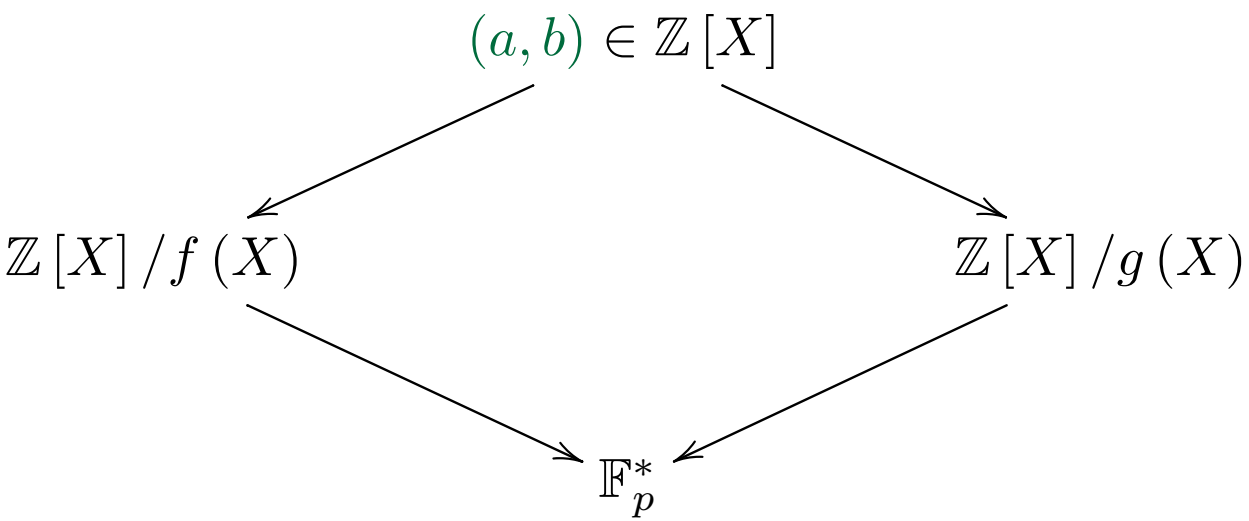


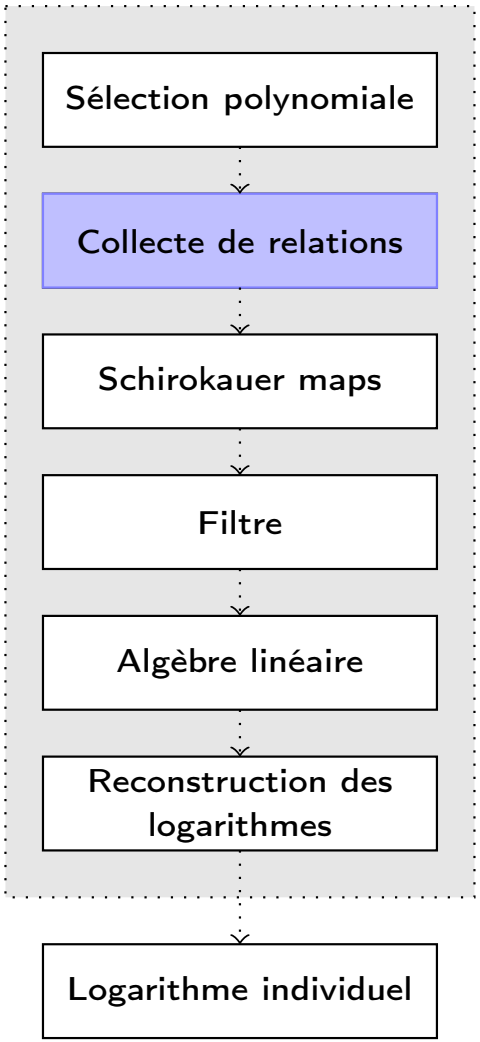
# Algorithme NFS



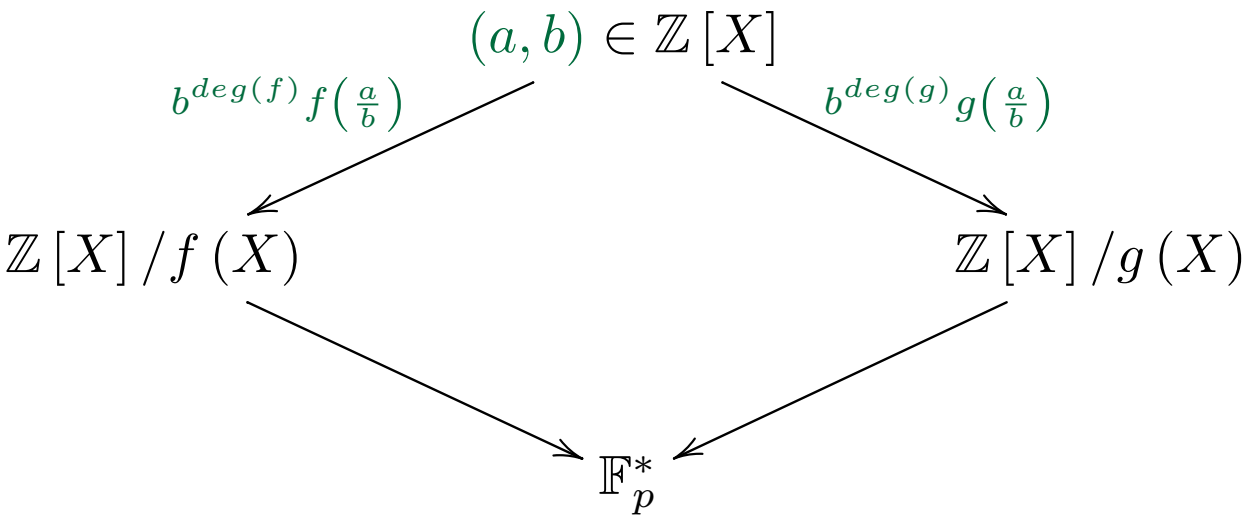



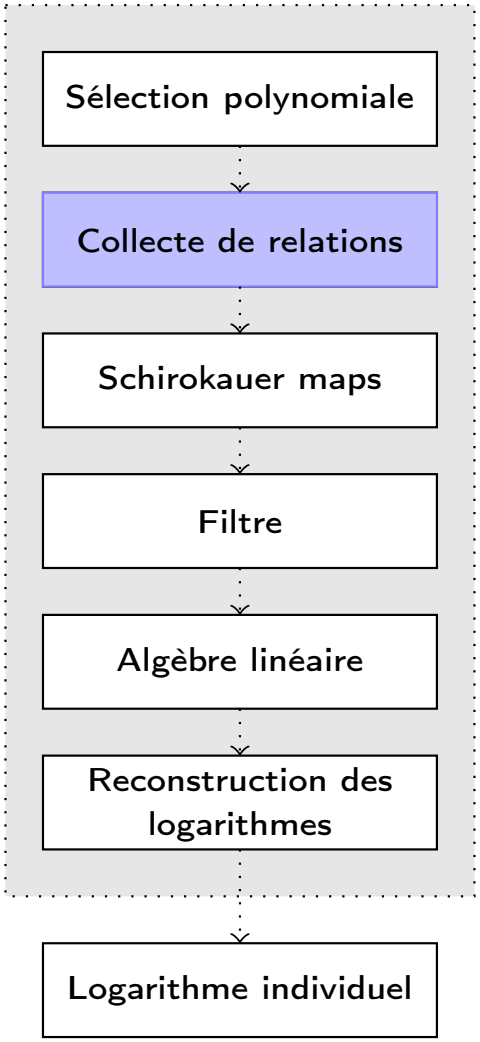
# Algorithme NFS



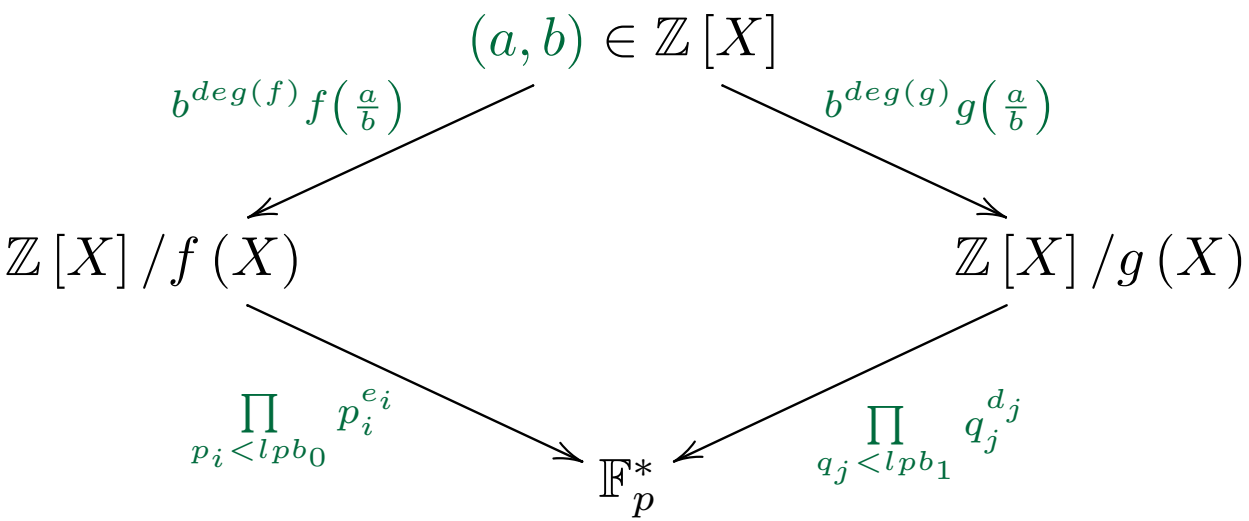



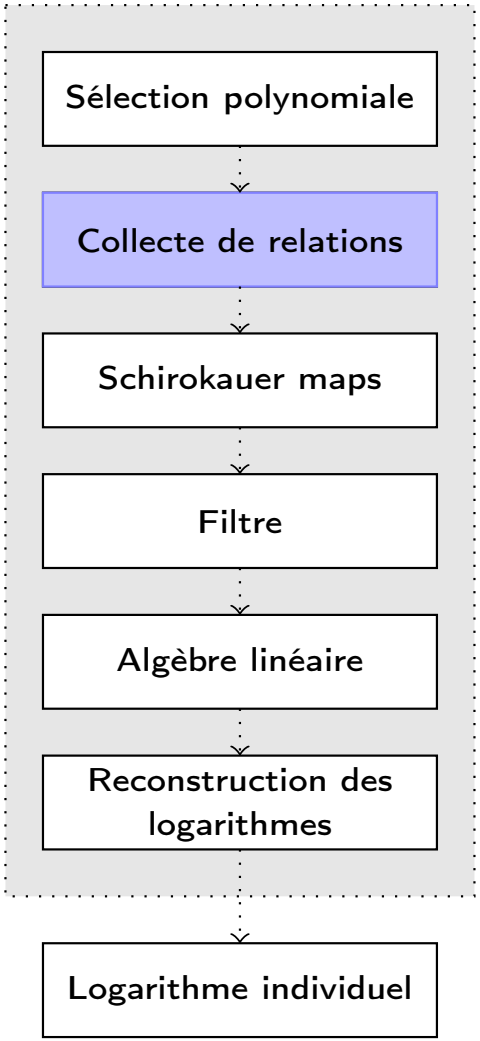
# Algorithme NFS



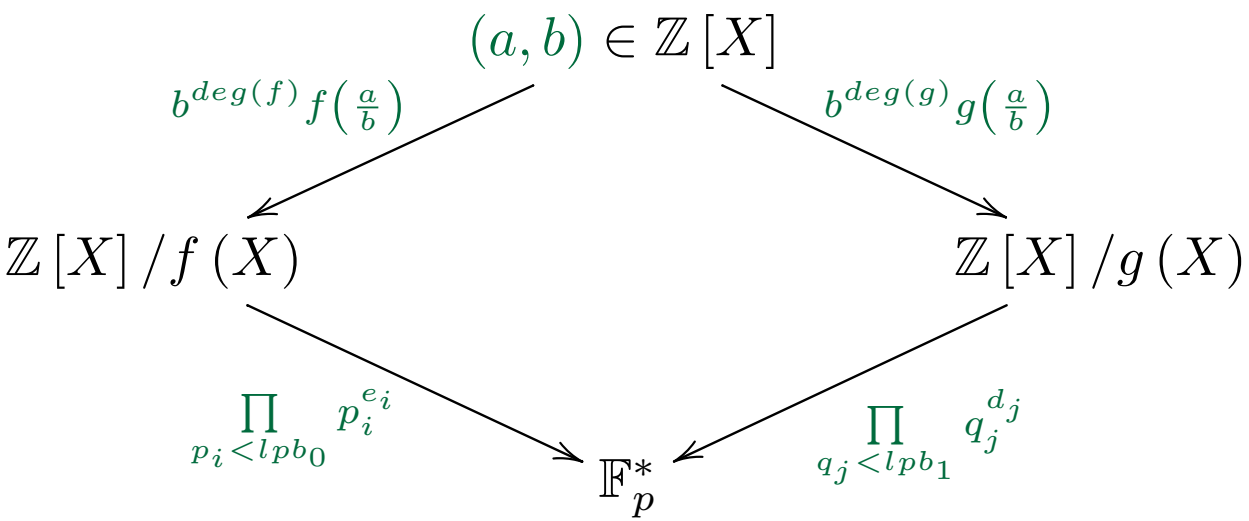



# Algorithme NFS

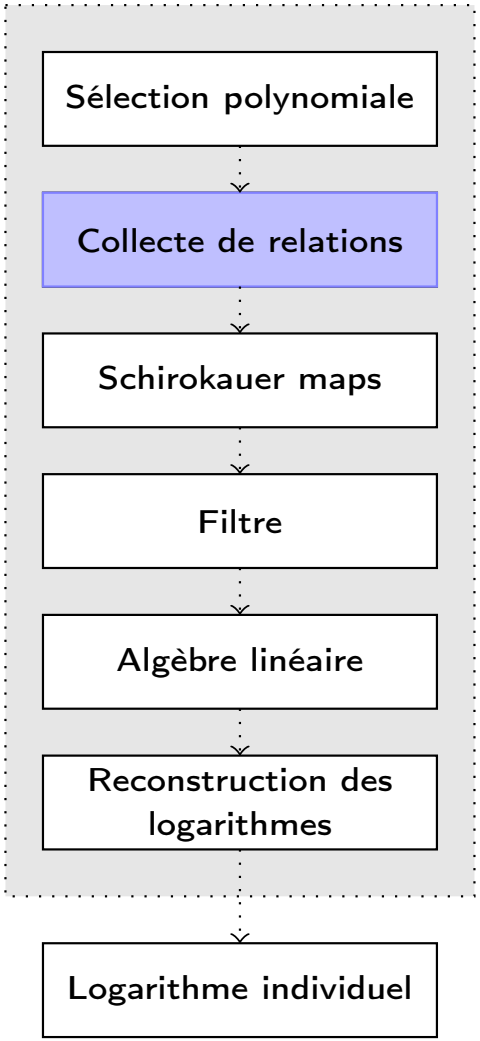


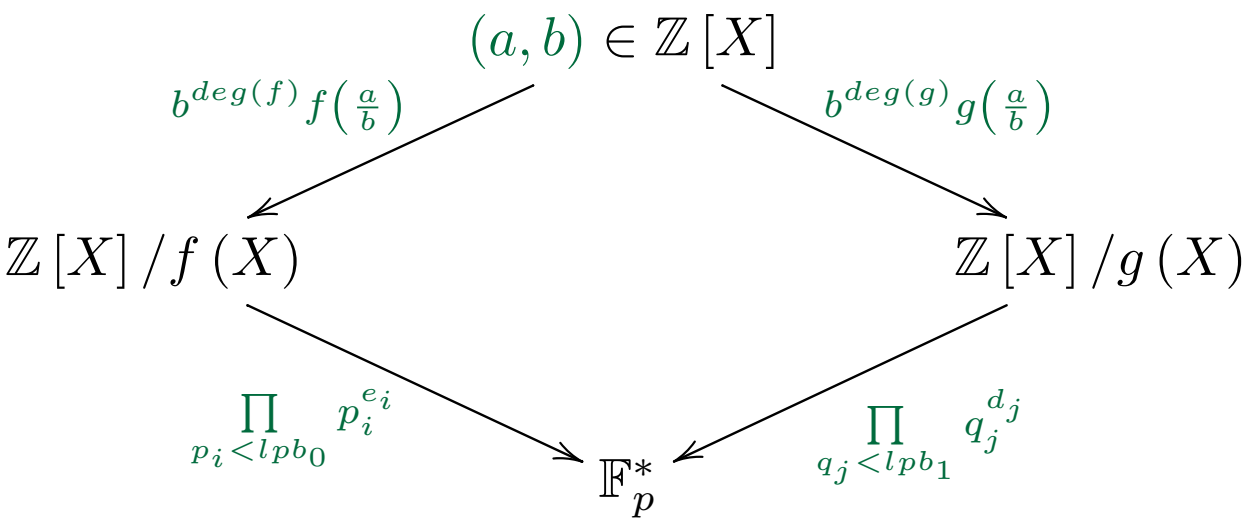
# Algorithme NFS




$$\prod_{p_i < lpb_0} p_i^{e_i} \equiv \prod_{q_j < lpb_1} q_j^{d_j} \ [p]$$

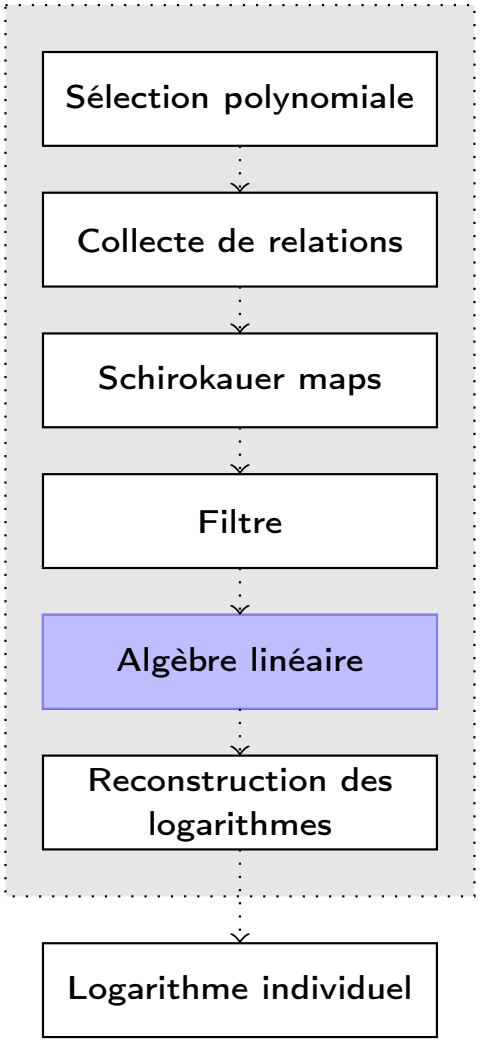


# Algorithme NFS



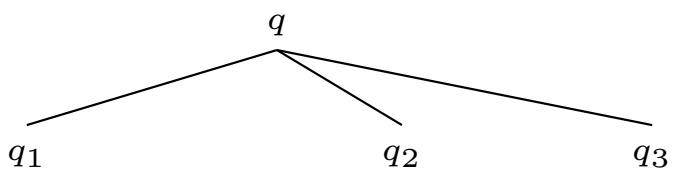

$$\prod_{p_i < lpb_0} p_i^{e_i} \equiv \prod_{q_j < lpb_1} q_j^{d_j} \,[p]$$

$$\sum_{p_i < lpb_0} e_i \log(p_i) - \sum_{q_j < lpb_1} d_j \log(q_j) \equiv 0 \,[\ell]$$

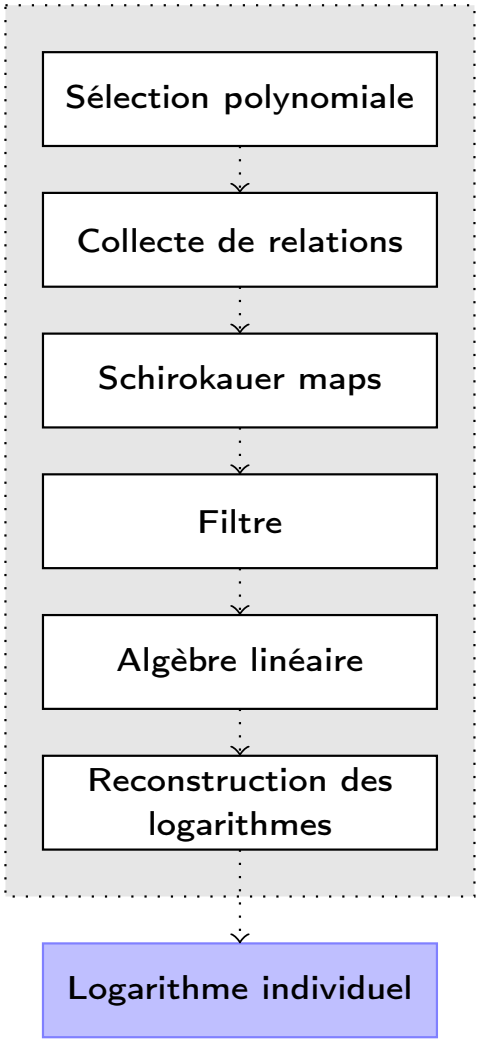


# CADO-NFS : Logarithmes individuels

$$\log(q) \equiv \sum_{p_i < lpb_0} e_i \log(q_i) \, [\ell]$$

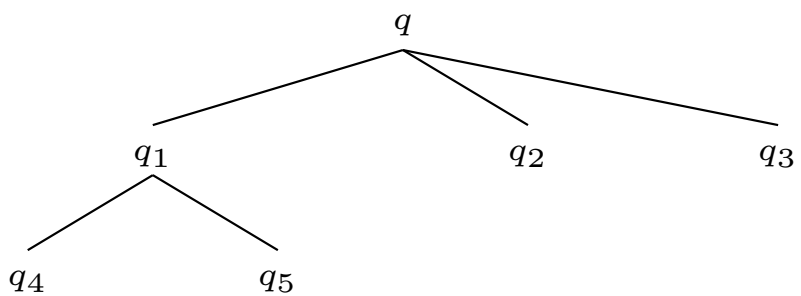



Base de donnée des logarithmes

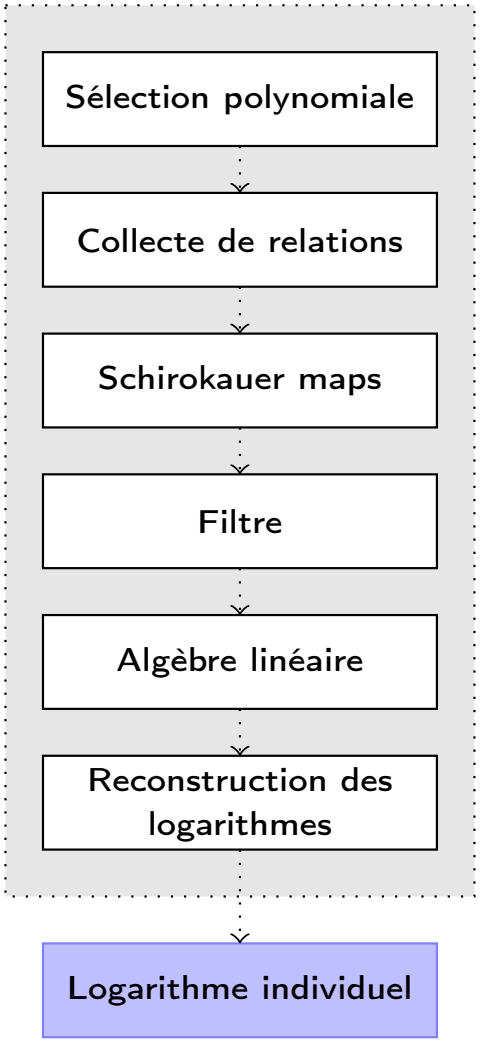


# CADO-NFS : Logarithmes individuels

$$\log(q) \equiv \sum_{p_i < lpb_0} e_i \log(q_i) \, [\ell]$$

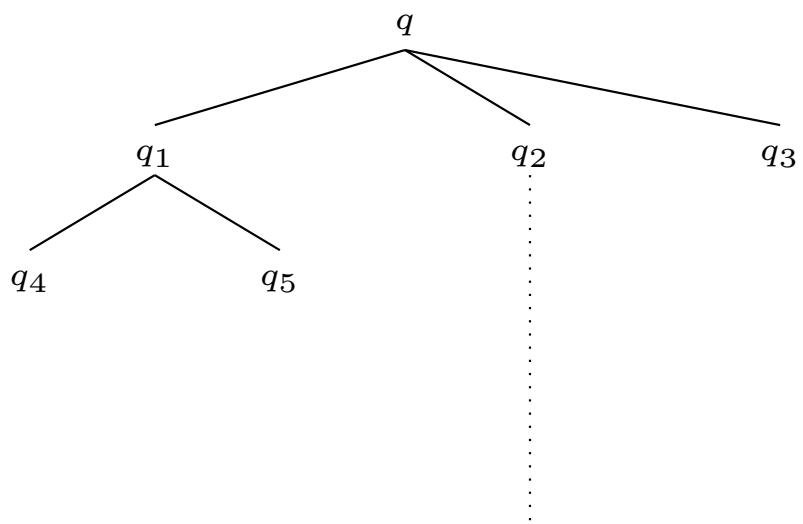



Base de donnée des logarithmes

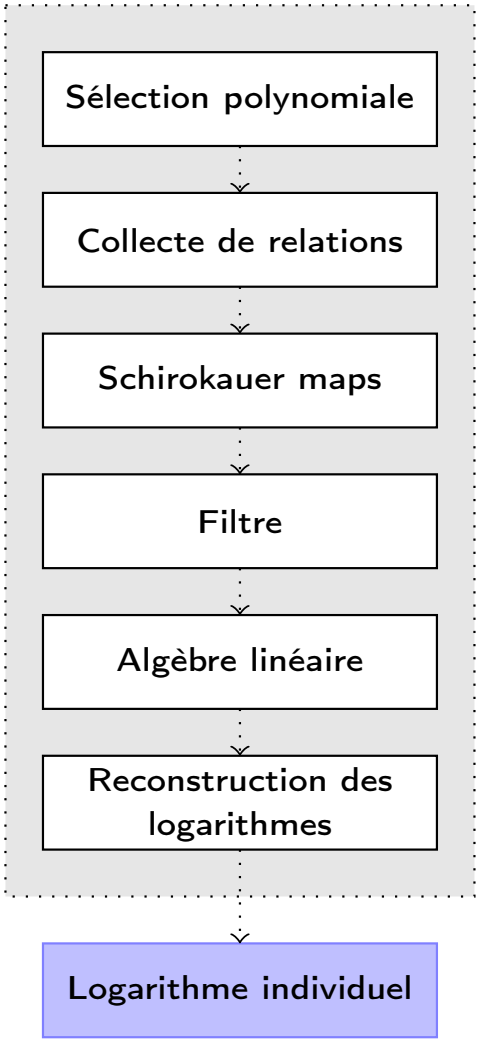


# CADO-NFS : Logarithmes individuels

$$\log(q) \equiv \sum_{p_i < lpb_0} e_i \log(q_i) \; [\ell]$$

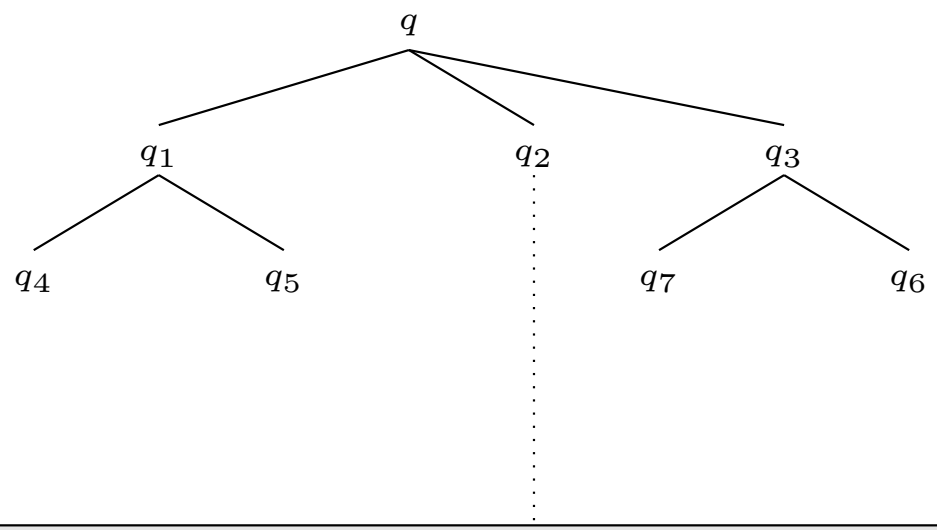



Base de donnée des logarithmes

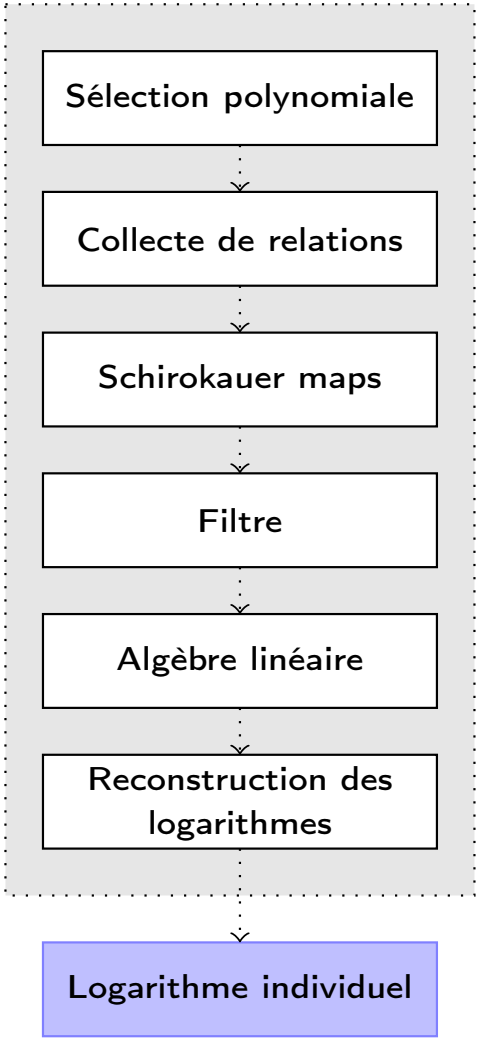


# CADO-NFS : Logarithmes individuels

$$\log(q) \equiv \sum_{p_i < lpb_0} e_i \log(q_i) \ [\ell]$$

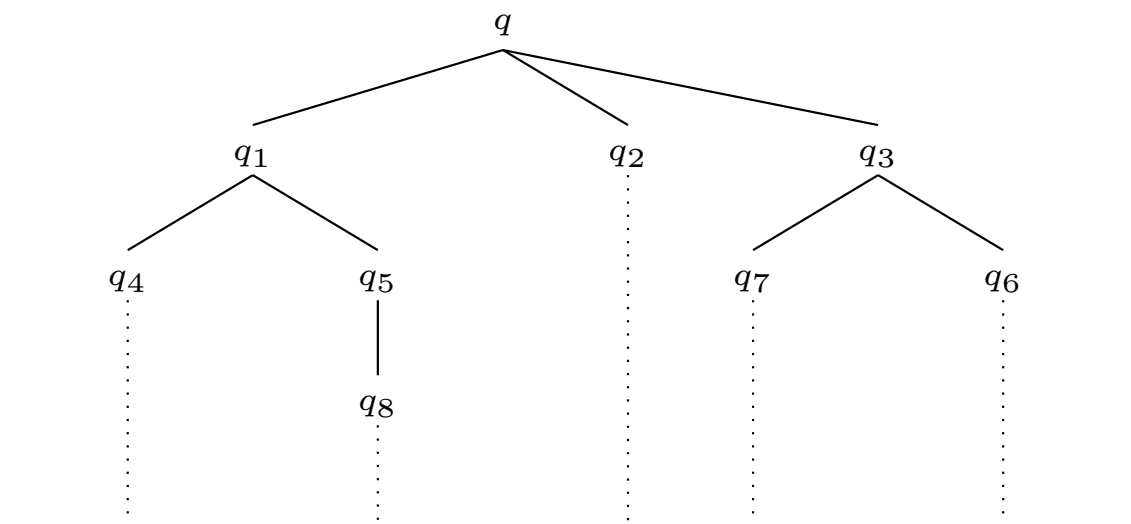


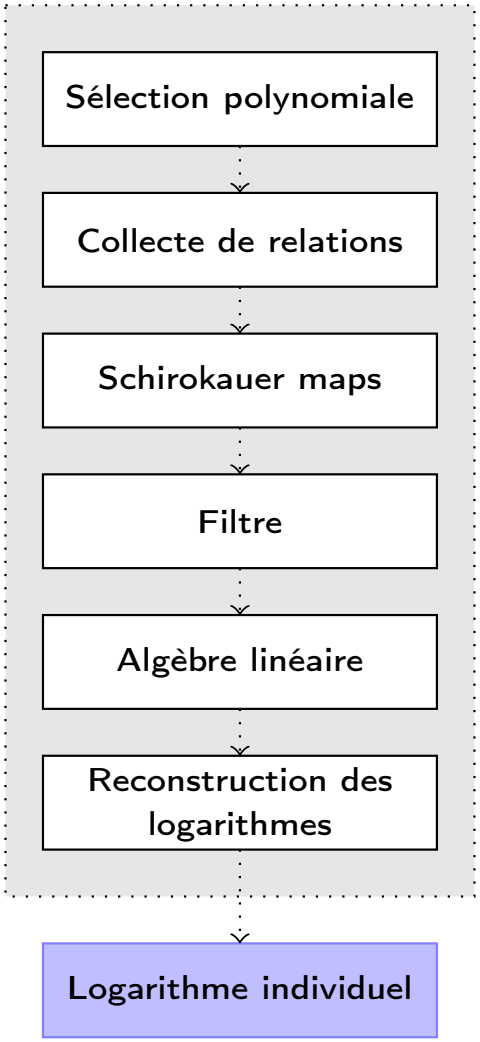

Base de donnée des logarithmes



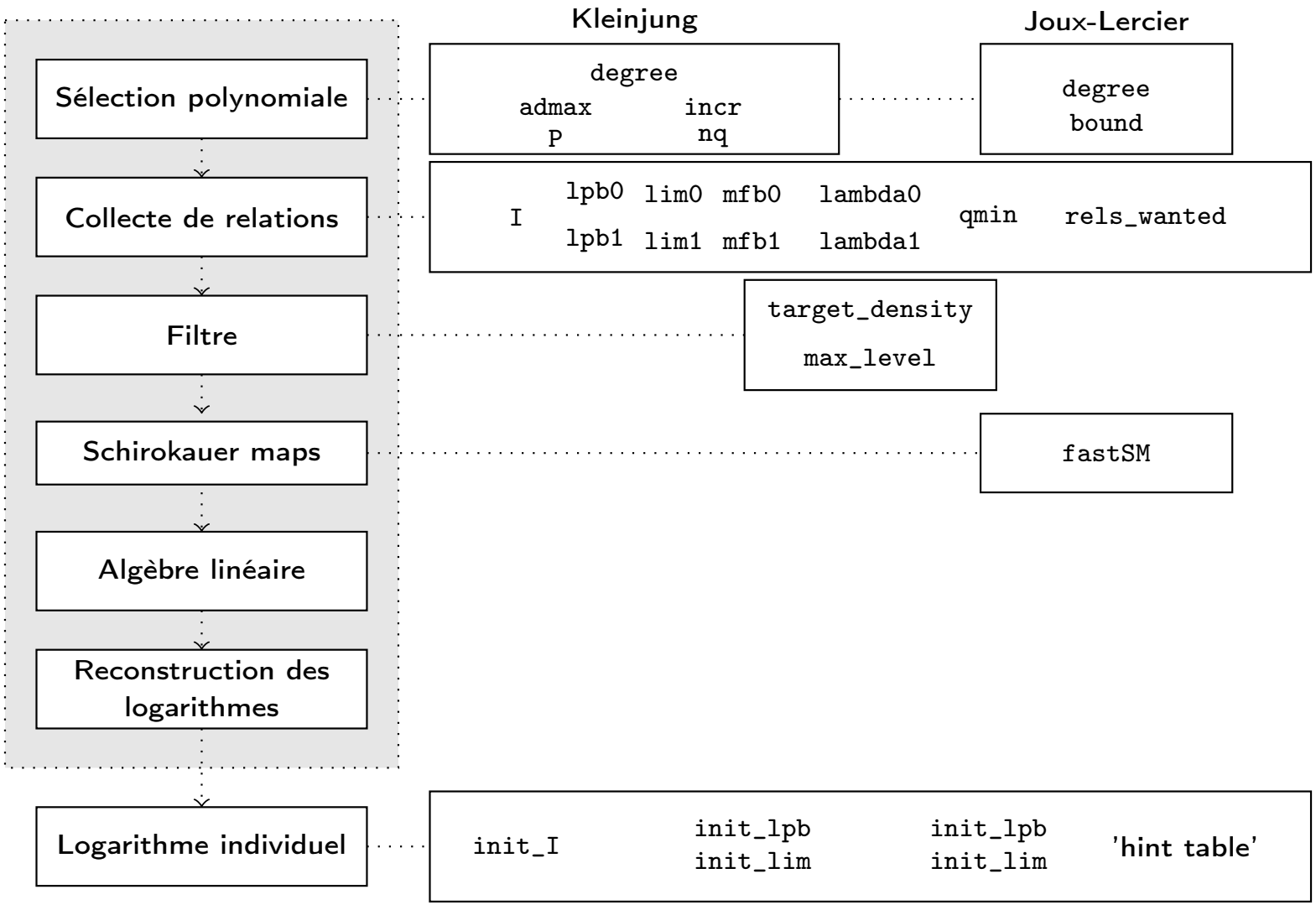
# CADO-NFS : Logarithmes individuels

$$\log(q) \equiv \sum_{p_i < lpb_0} e_i \log(q_i) \, [\ell]$$



# Vue d'ensemble des principaux paramètres



# CADO-NFS : Idées pour le paramétrage

Le pire cas : $p$ nombre premier de Sophie Germain.
C'est-à-dire $p = 2\ell + 1$ avec $\ell$ premier.
Étude de NFS dans le cas $p$ de Sophie Germain et $\ell = \frac{p-1}{2}$.

- Première approche : s'intéresser au temps CPU total.
  - Trop de temps d'attente.
  - Peu d'informations sur la pertinence des paramètres obtenus.
- Seconde approche : s'intéresser aux résultats des différentes étapes.
  - Permet interprétation locale des résultats.
  - Mais pas de garantie sur l'optimalité.

# CADO-NFS : Idées pour le paramétrage

Le pire cas : $p$ nombre premier de Sophie Germain.
C'est-à-dire $p = 2\ell + 1$ avec $\ell$ premier.
Étude de NFS dans le cas $p$ de Sophie Germain et $\ell = \frac{p-1}{2}$.

- Première approche : s'intéresser au temps CPU total.
  - Trop de temps d'attente.
  - Peu d'informations sur la pertinence des paramètres obtenus.

- Seconde approche : s'intéresser aux résultats des différentes étapes.
  - Permet interprétation locale des résultats.
  - Mais pas de garantie sur l'optimalité.

# CADO-NFS : Idées pour le paramétrage

Le pire cas : $p$ nombre premier de Sophie Germain.
C'est-à-dire $p = 2\ell + 1$ avec $\ell$ premier.
Étude de NFS dans le cas $p$ de Sophie Germain et $\ell = \frac{p-1}{2}$.

- Première approche : s'intéresser au temps CPU total.
  - Trop de temps d'attente.
  - Peu d'informations sur la pertinence des paramètres obtenus.
- Seconde approche : s'intéresser aux résultats des différentes étapes.
  - Permet interprétation locale des résultats.
  - Mais pas de garantie sur l'optimalité.

# Connaissances mises en pratique

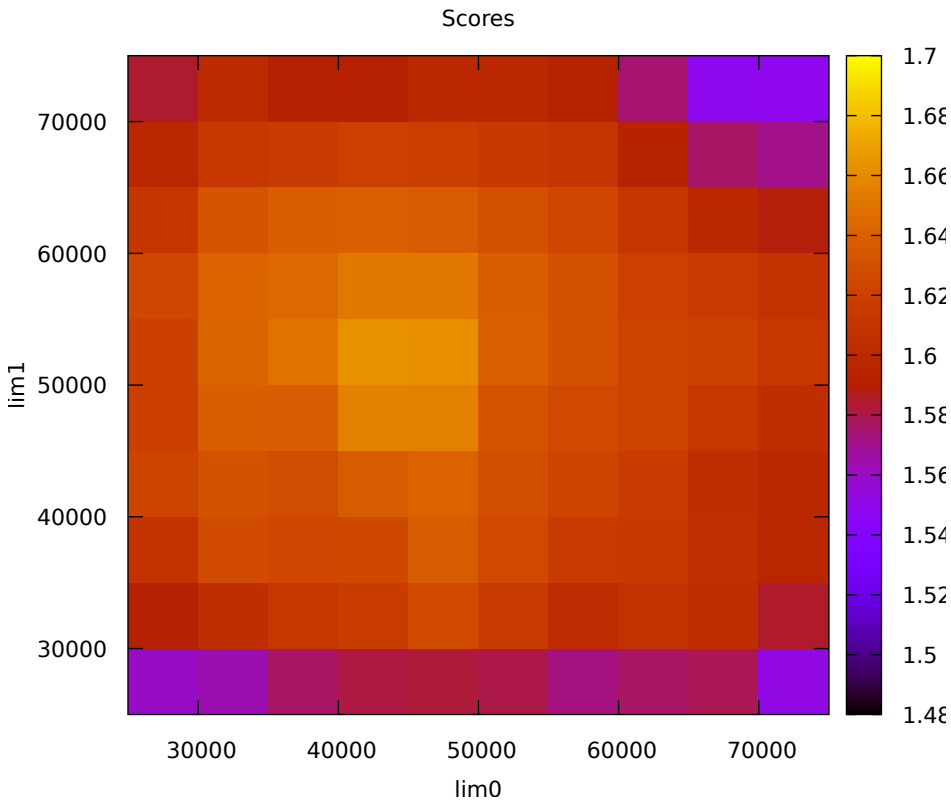
- Connaissances en algèbre : principalement en théorie des nombres ou issues du cours de cryptographie.

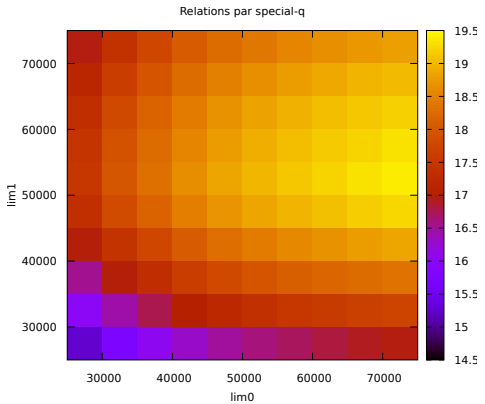
- Outils informatiques : `python3`, `SageMath`.

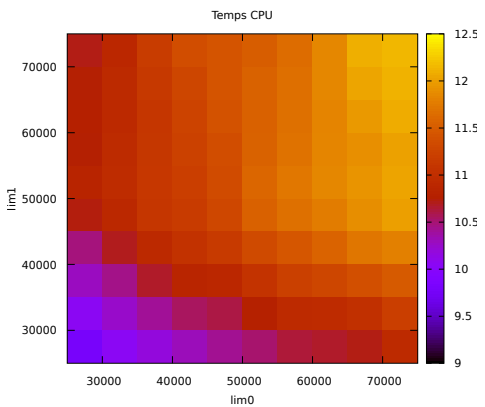
  - Parsing de différentes statistiques issues de CADO-NFS.
  - Génération de graphiques / résultats humainement observables.
  - Automatisation de certains calculs.
  - Calculs mathématiques avec Sage.

# Chercher les paramètres à l'aide d'unités de travail

Utilisations de mes outils pour visualiser des résultats locaux.

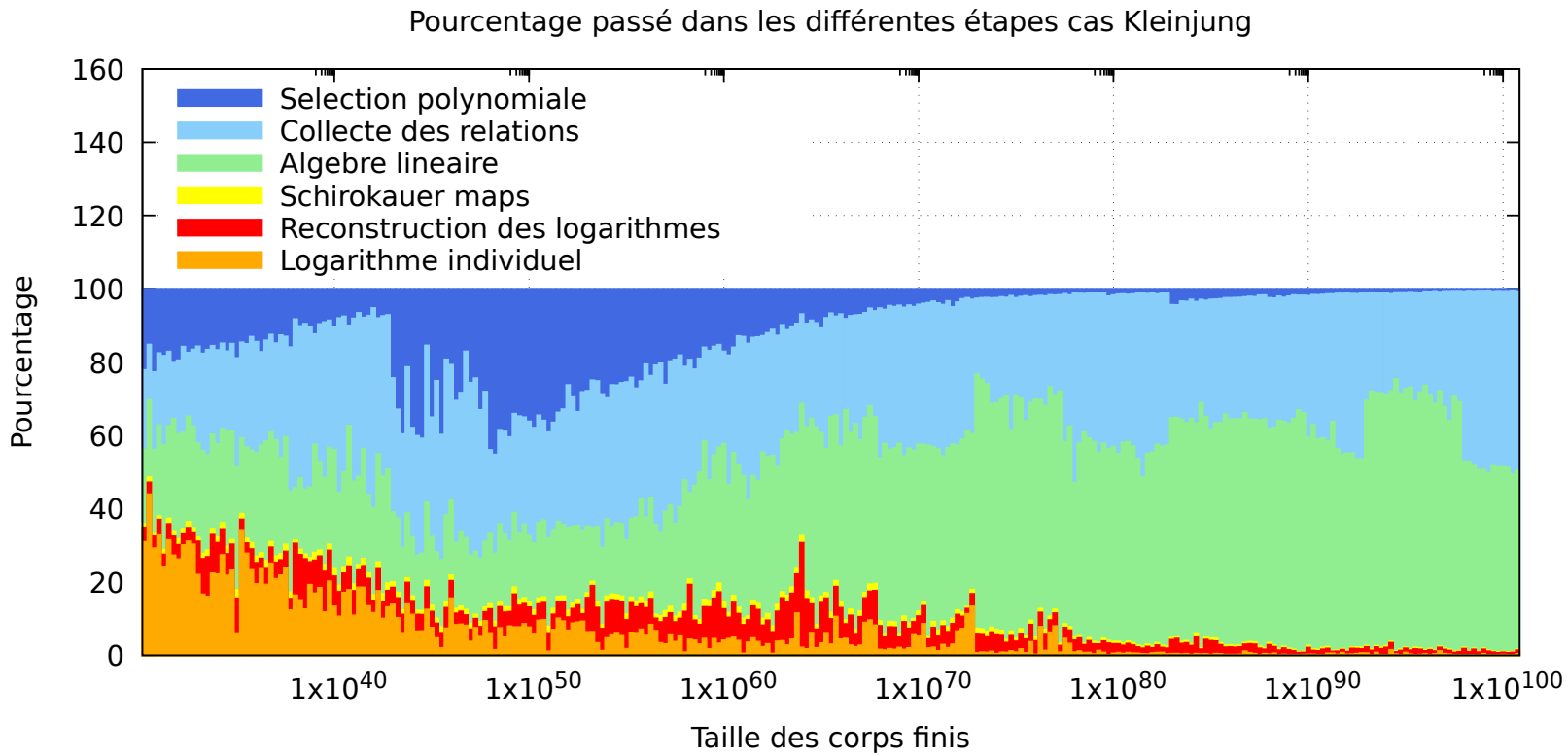


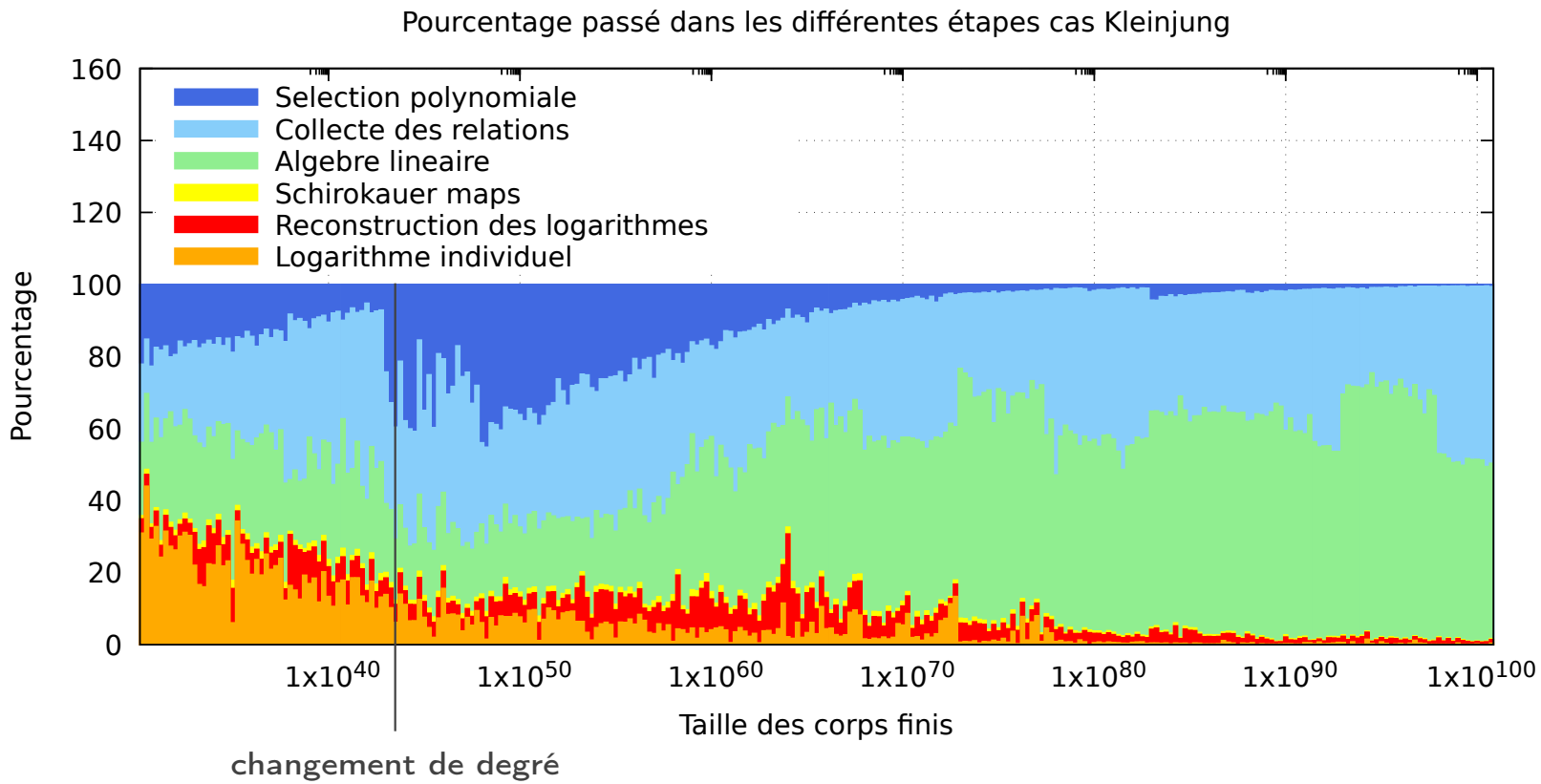
# Statistiques sur les différentes étapes

S'intéresser aux différentes statistiques d'un test.



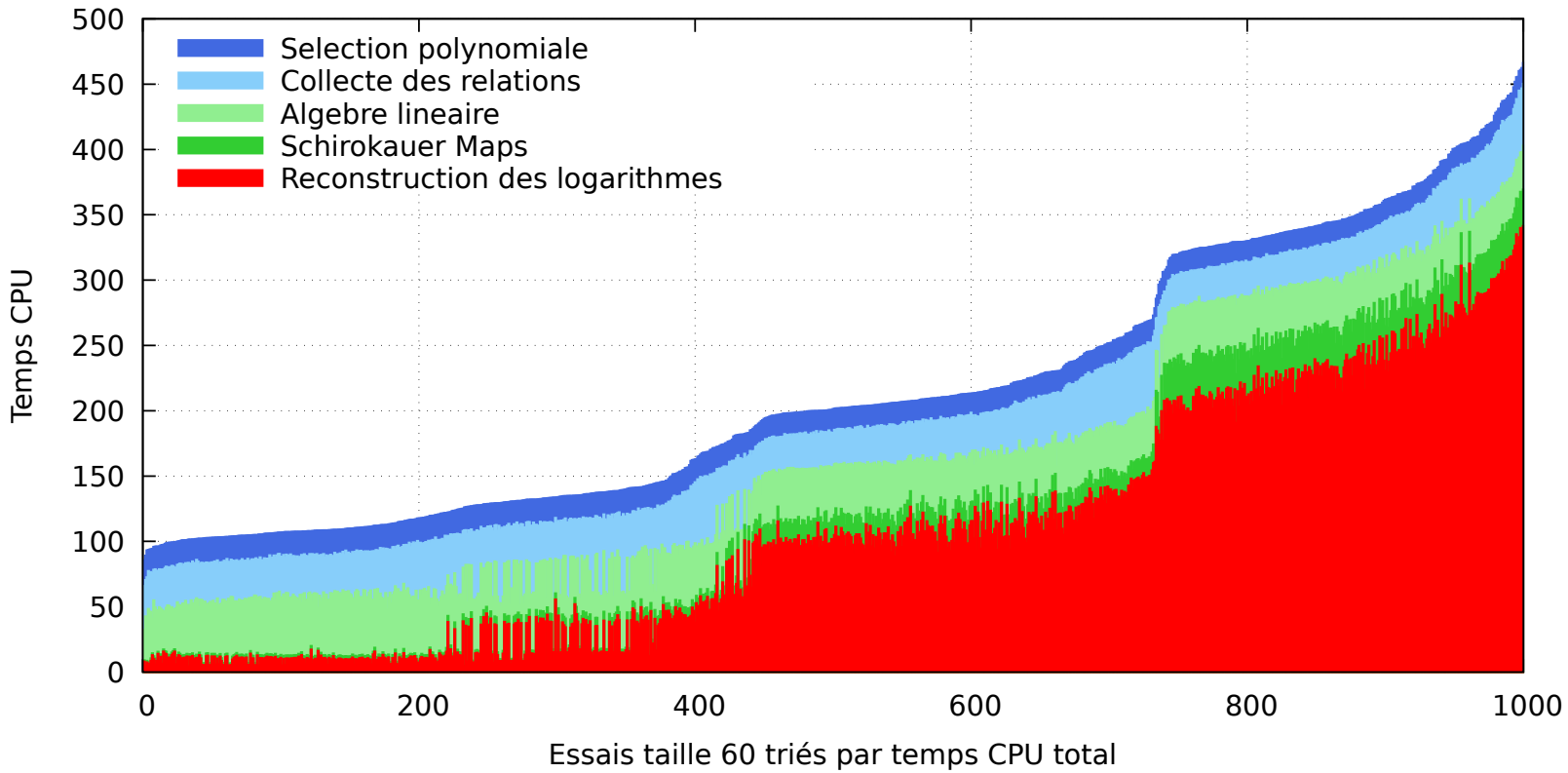
# Statistiques sur les différentes étapes

S'intéresser aux différentes statistiques d'un test.

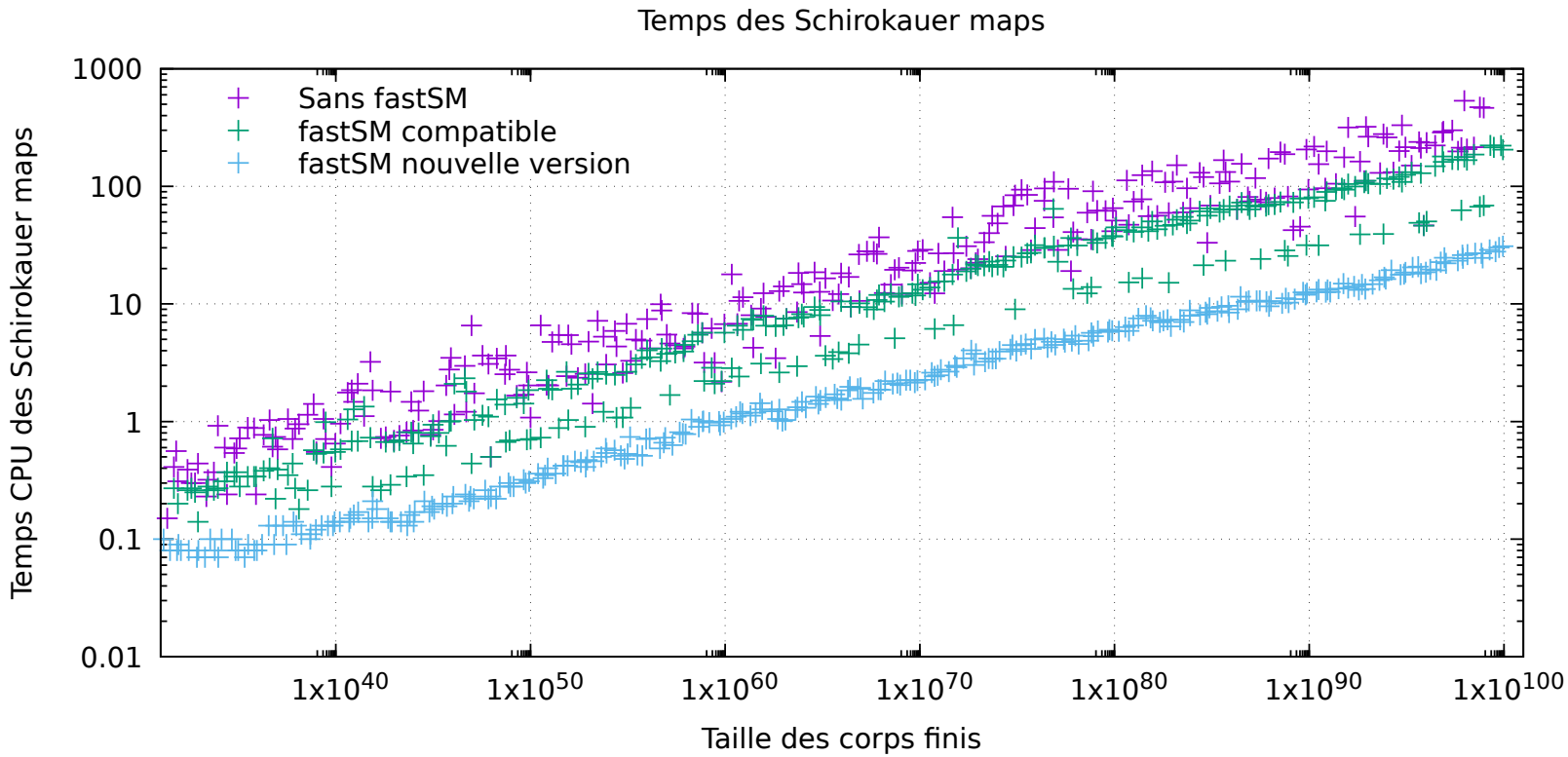


# Problème des Schirokauer maps

Calcul en moyenne : mise en évidence du problème des Schirokauer maps pour les petites tailles.

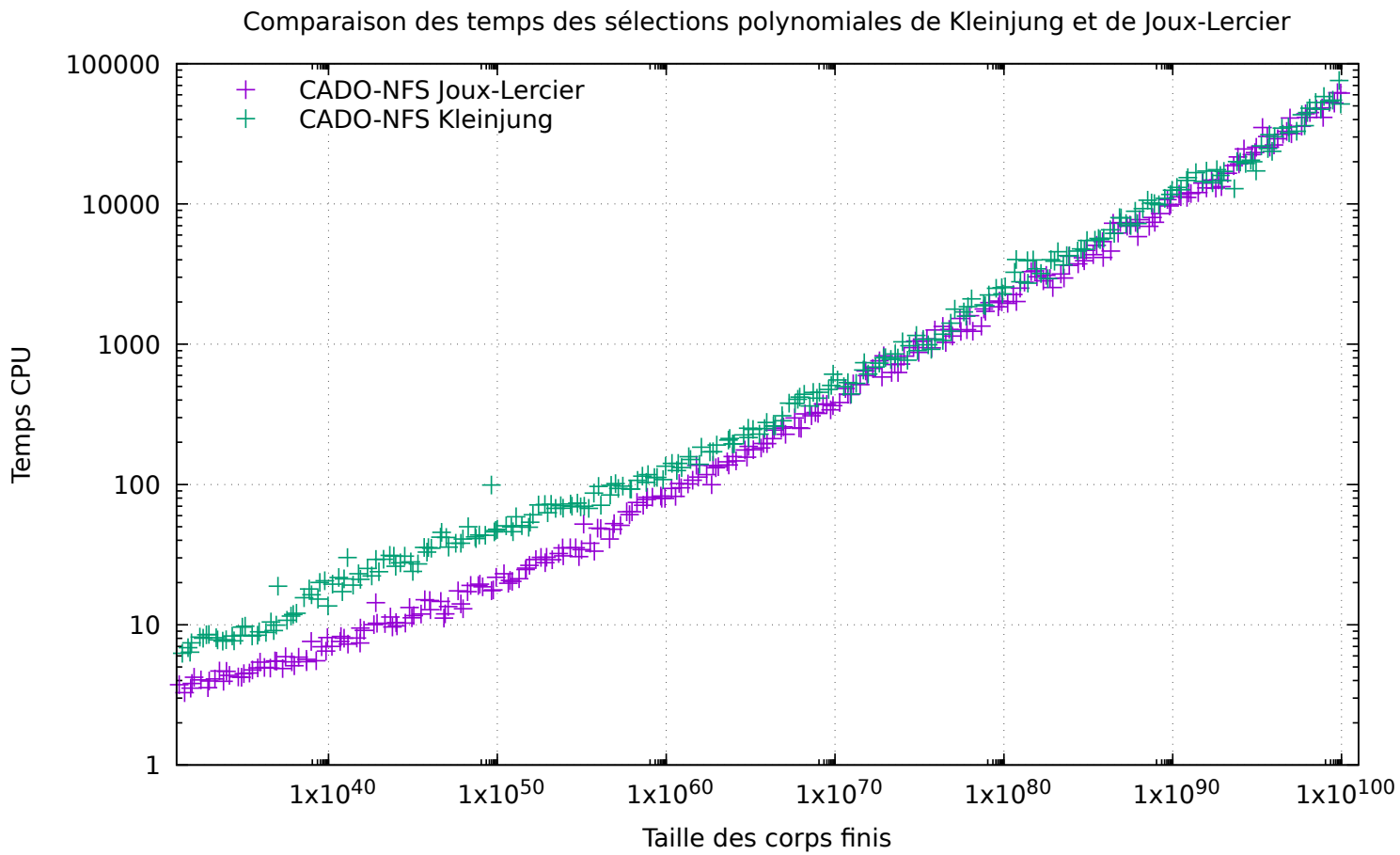


# Solutions au problème des Schirokauer maps

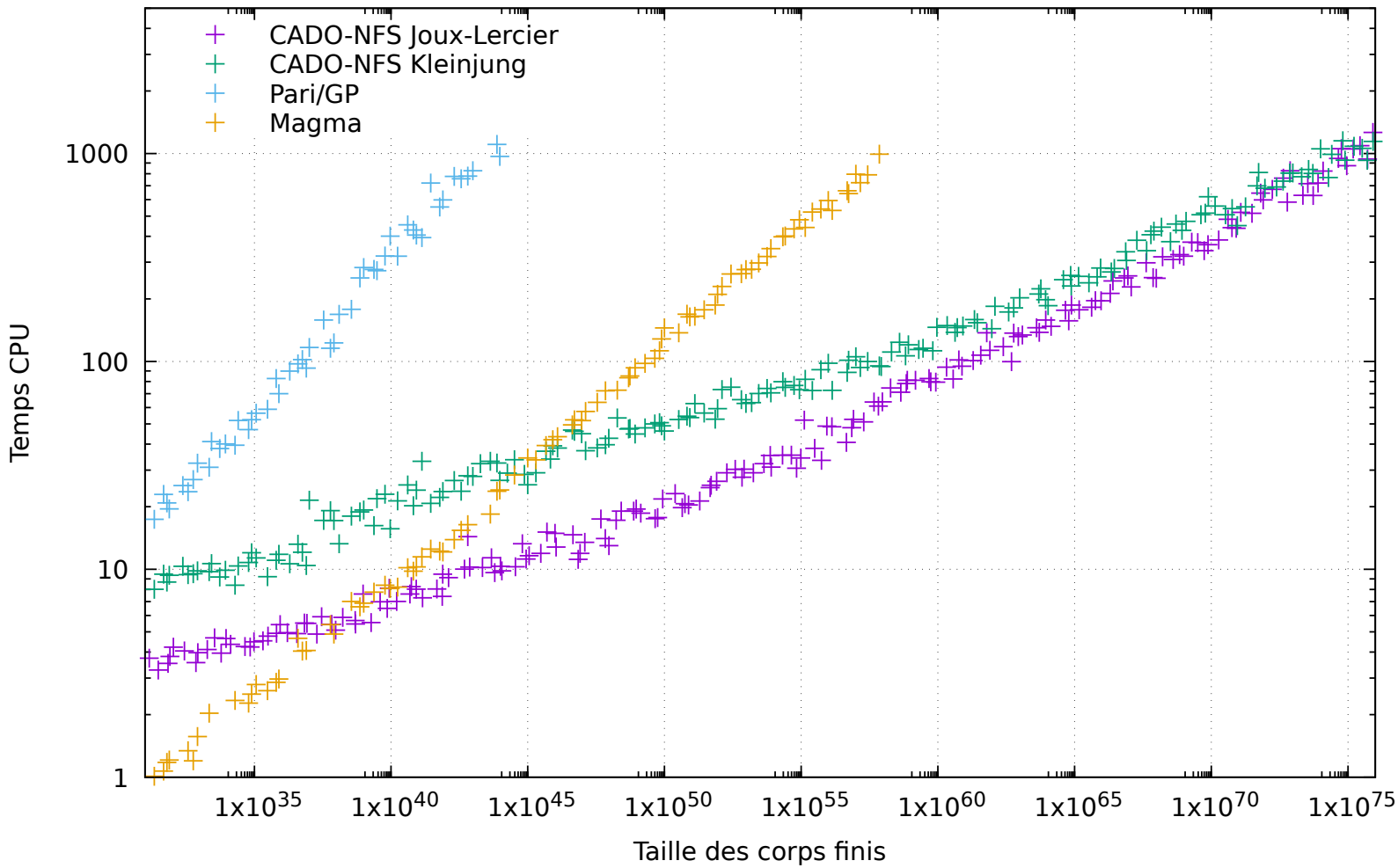


# Sélection polynomiale Joux-Lercier et Kleinjung

Comparaison de Joux-Lercier (expérimental) et Kleinjung (actuellement utilisé).



# CADO-NFS et d'autres implémentations du logarithme discret



# Contributions

Jeux de paramètres Kleinjung et Joux-Lercier ajoutés à CADO-NFS.

Exposé scientifique devant l'équipe CARAMBA pour présenter mes résultats et peut-être donner des idées de nouvelles pistes de recherche.

Grand nombre de cas testés : mise en évidence et rapport de quelques bugs corrigés.

# Conclusion

- Paramétrage de CADO-NFS : un problème de recherche intéressant, approfondissement de mes connaissances en mathématiques et expérimentation de méthodes en développement (Joux-Lercier).

- L'occasion de mettre en œuvre à la fois mes connaissances en mathématiques et mes compétences informatiques.

- Nouveaux jeux de paramètres pour CADO-NFS, mise en évidence de la pertinence de CADO-NFS face aux autres implémentations.

- Pour aller plus loin dans le stage : approfondir les méthodes que j'ai trouvées pour créer un outil automatique sans intervention humaine ?

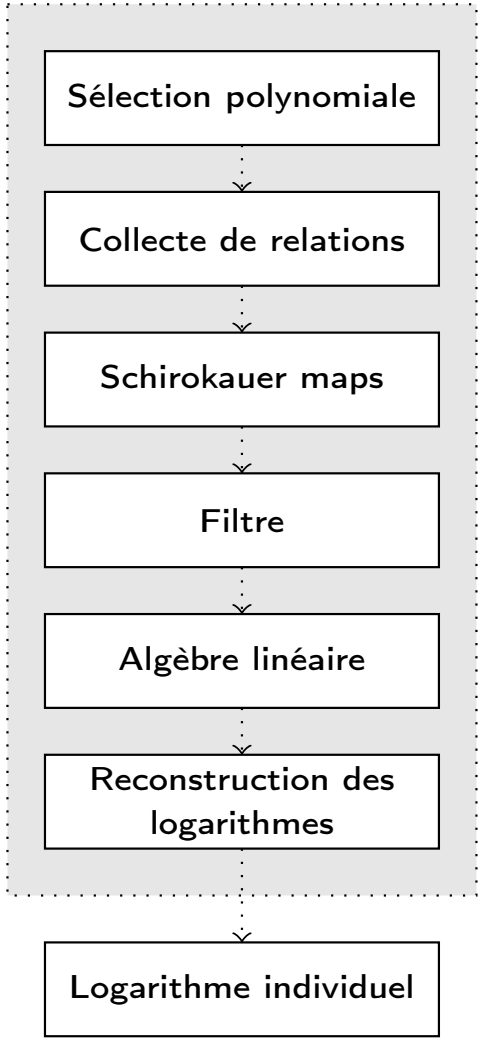
# Questions

Merci de votre attention.

Posez vos questions.

# CADO-NFS en bref

- On sélectionne des polynômes.
- On factorise les normes pour obtenir des relations.
- On fabrique un système linéaire où les logarithmes sont les inconnues.
- On résout le système par l'algèbre linéaire.
- On fabrique un arbre pour exprimer un logarithme cible en fonction de logarithmes connus et précalculés.



# CADO-NFS : Sélection polynomiale

À un paire $(a, b) \in \mathbb{Z}^2$, on associe les normes :

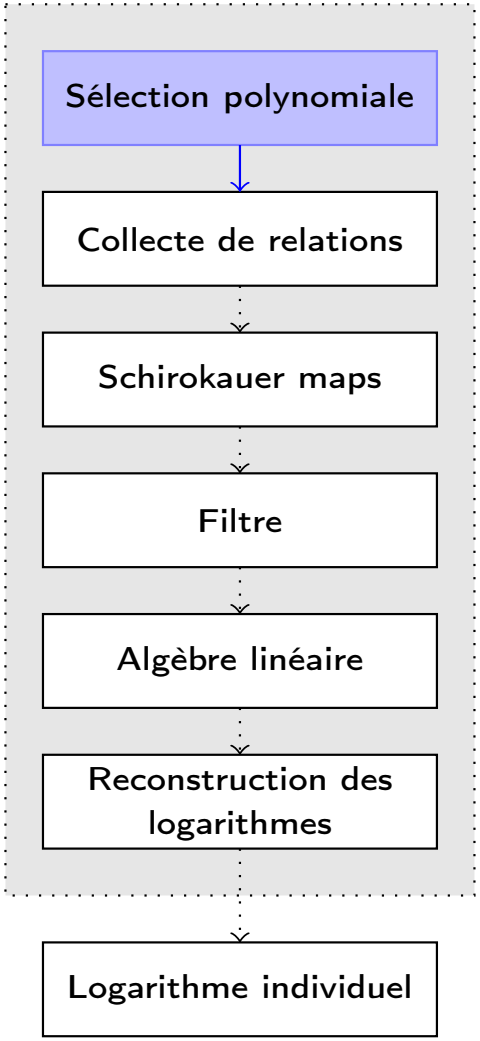
$$Norm_f\left((a,b)\right) = b^{deg(f)} f(\frac{a}{b})$$

$$Norm_g\left((a,b)\right) = b^{deg(g)} g(\frac{a}{b})$$

$N$ est B-friable s'il se décompose en facteurs premiers plus petits que $B$ :

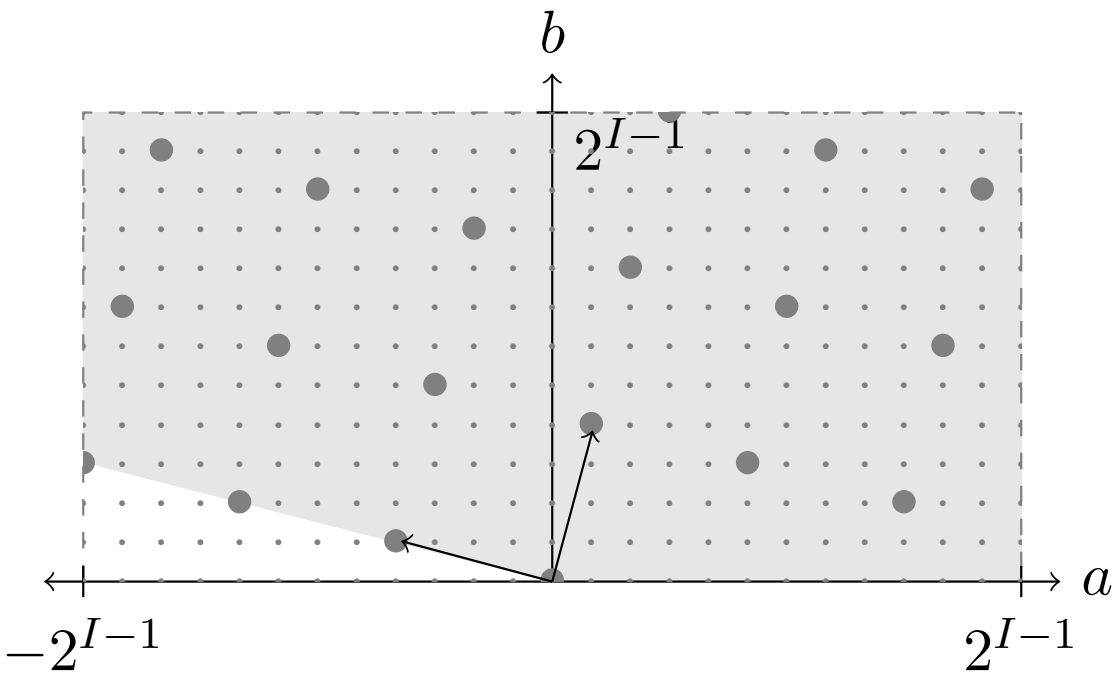
$$N = \prod_{q<B} q^{e_q}.$$

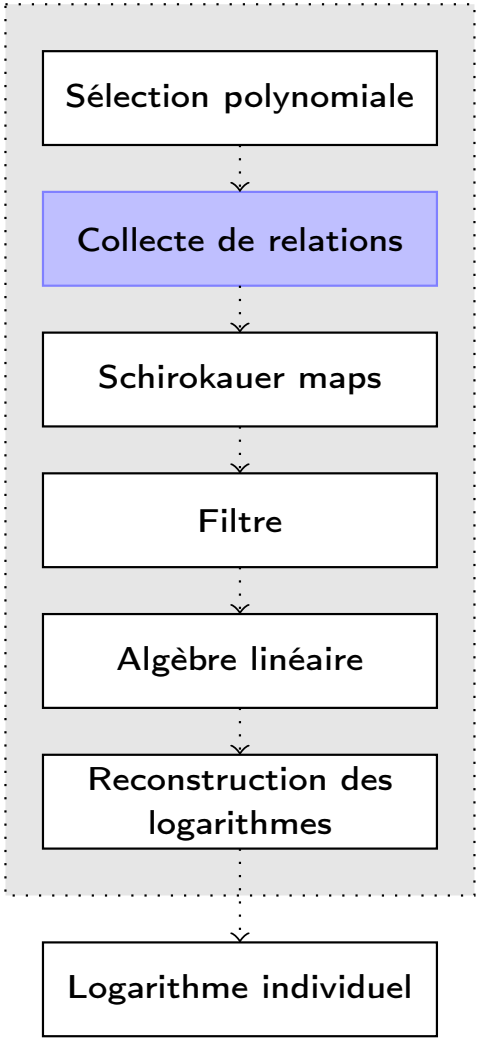
On veut exploiter la friabilité des normes.
Qualité des polynômes : quantité-E de Murphy.



# CADO-NFS : Collecte de relations

On impose un spécial-$\mathfrak{q}$ : facteur premier dans une relation.
On crible sur les normes des paires $(a, b)$ dans un espace à deux dimensions :



# CADO-NFS : Collecte de relations

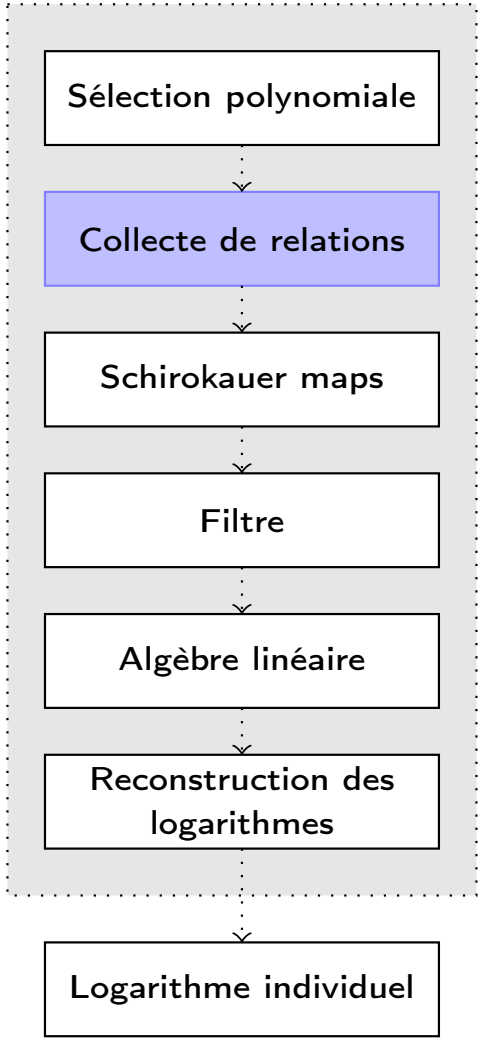
(a, b) = (201098, 29)
Exemple : $5109426692651231679705 39$

- Spécial-$\mathfrak{q}$
- Crible d'Eratosthène.

$$Norm\left((a,b)\right) = \left( \prod_{Norm(\mathfrak{q}) < LIM} Norm\left(\mathfrak{q}\right)^{e_{\mathfrak{q}}} \right) \times R.$$

- Cofactorisation par l'Elliptic Curve Method.

$$Norm\left((a,b)\right) = \prod_{Norm(\mathfrak{q}) < LPB} Norm\left(\mathfrak{q}\right)^{e_{\mathfrak{q}}}.$$



# CADO-NFS : Collecte de relations

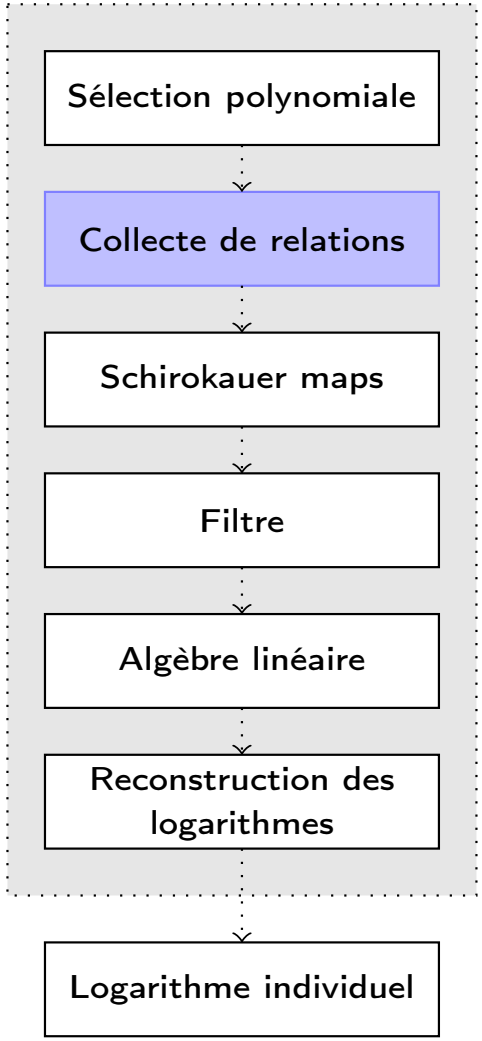
(a, b) = (201098, 29)
Exemple : $\mathbf{6217}.821847626291013620 67$

- Spécial-$\mathfrak{q}$
- Crible d'Eratosthène.

$$Norm\left((a,b)\right) = \left( \prod_{Norm(\mathfrak{q})<LIM} Norm\left(\mathfrak{q}\right)^{e_{\mathfrak{q}}} \right) \times R.$$

- Cofactorisation par l'Elliptic Curve Method.

$$Norm\left((a,b)\right) = \prod_{Norm(\mathfrak{q})<LPB} Norm\left(\mathfrak{q}\right)^{e_{\mathfrak{q}}}.$$



# CADO-NFS : Collecte de relations

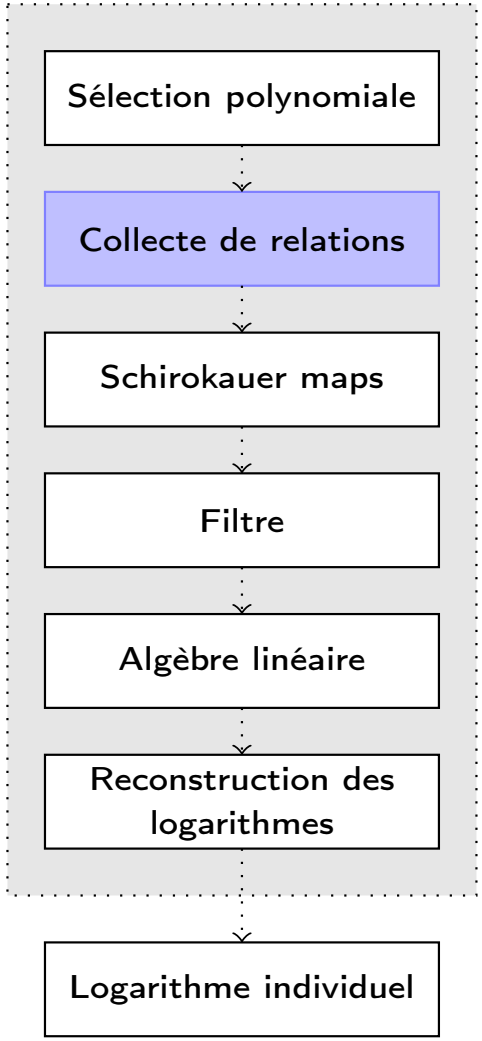
(a, b) = (201098, 29)
Exemple : $6217.\mathbf{3^3}.\mathbf{307}.\mathbf{1033}.\mathbf{11519}.833247589$

- Spécial-$\mathfrak{q}$
- Crible d'Eratosthène.

$$Norm\left((a,b)\right) = \left(\prod_{Norm(\mathfrak{q})<LIM} Norm\left(\mathfrak{q}\right)^{e_{\mathfrak{q}}}\right) \times R.$$

- Cofactorisation par l'Elliptic Curve Method.

$$Norm\left((a,b)\right) = \prod_{Norm(\mathfrak{q})<LPB} Norm\left(\mathfrak{q}\right)^{e_{\mathfrak{q}}}.$$



# CADO-NFS : Collecte de relations

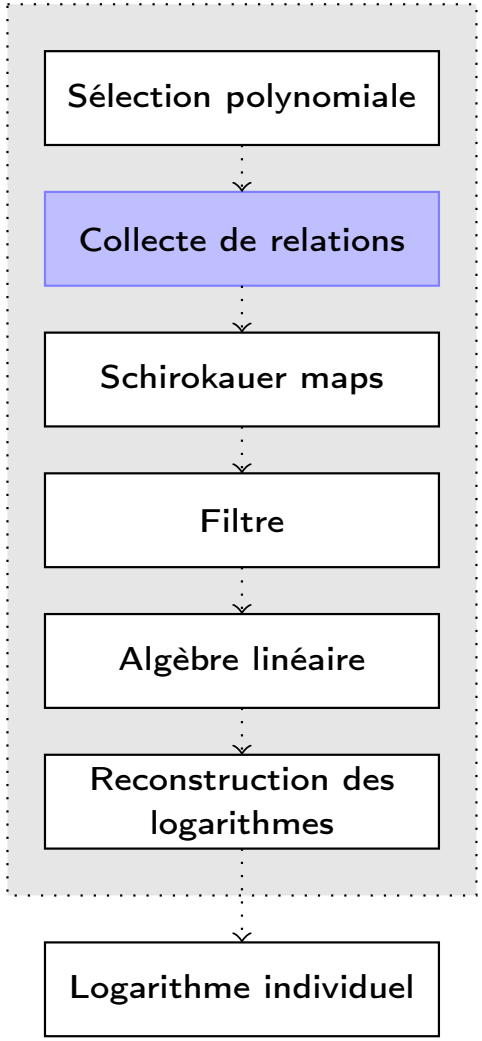
(a, b) = (201098, 29)
Exemple : $6217.3^3.307.1033.11519.\mathbf{27893.29873}$

- Spécial-$\mathfrak{q}$
- Crible d'Eratosthène.

$$Norm\left((a,b)\right) = \left(\prod_{Norm(\mathfrak{q})<LIM} Norm\left(\mathfrak{q}\right)^{e_{\mathfrak{q}}}\right) \times R.$$

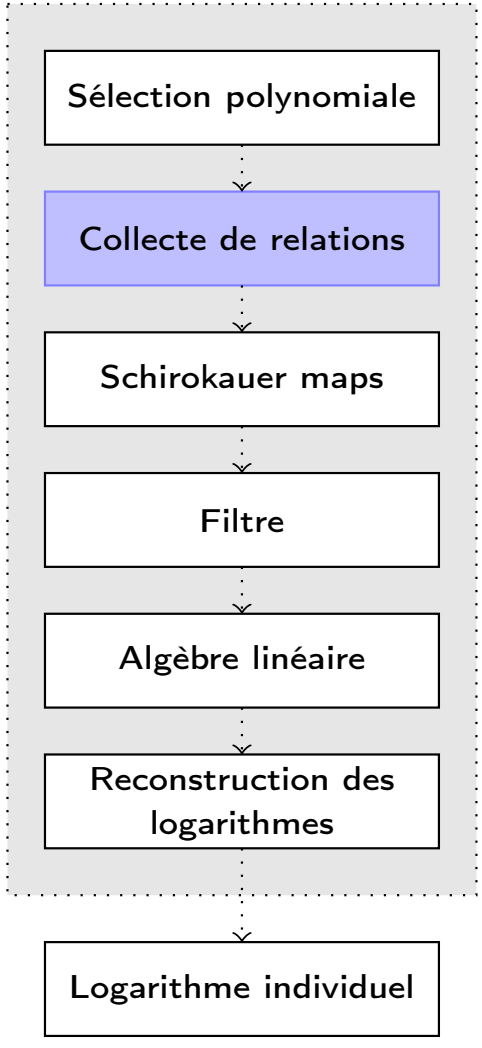
- Cofactorisation par l'Elliptic Curve Method.

$$Norm\left((a,b)\right) = \prod_{Norm(\mathfrak{q})<LPB} Norm\left(\mathfrak{q}\right)^{e_{\mathfrak{q}}}.$$



# CADO-NFS : Collecte de relations

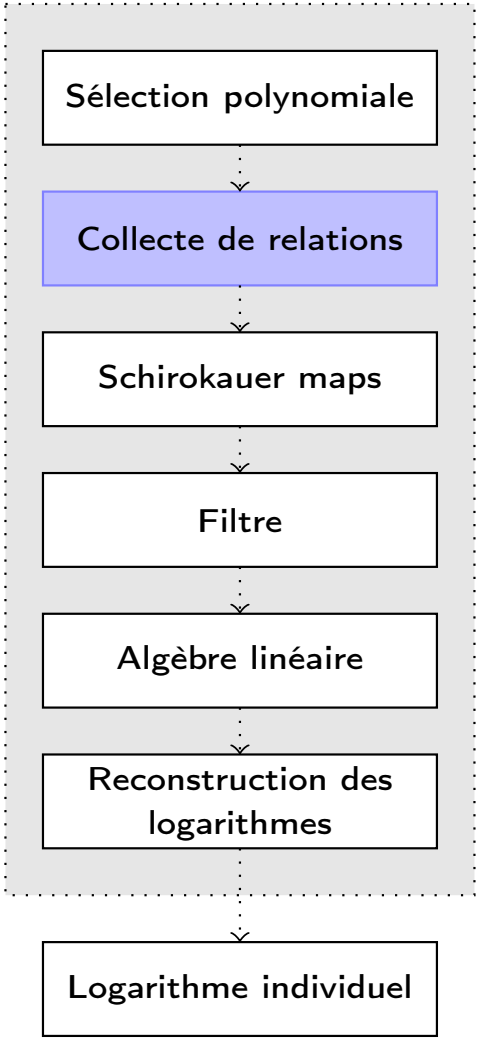
$$Norm\left((a,b)\right) = \prod_{Norm(\mathfrak{q})<LPB} Norm\left(\mathfrak{q}\right)^{e_{\mathfrak{q}}},$$



# CADO-NFS : Collecte de relations

$$Norm\left((a,b)\right) = \prod_{Norm(\mathfrak{q})<LPB} Norm\left(\mathfrak{q}\right)^{e_{\mathfrak{q}}},$$

$$\text{“}\prod_i p_i^{e_i} \equiv \prod_j q_j^{d_j}\ [p]\text{”}^{1}.$$



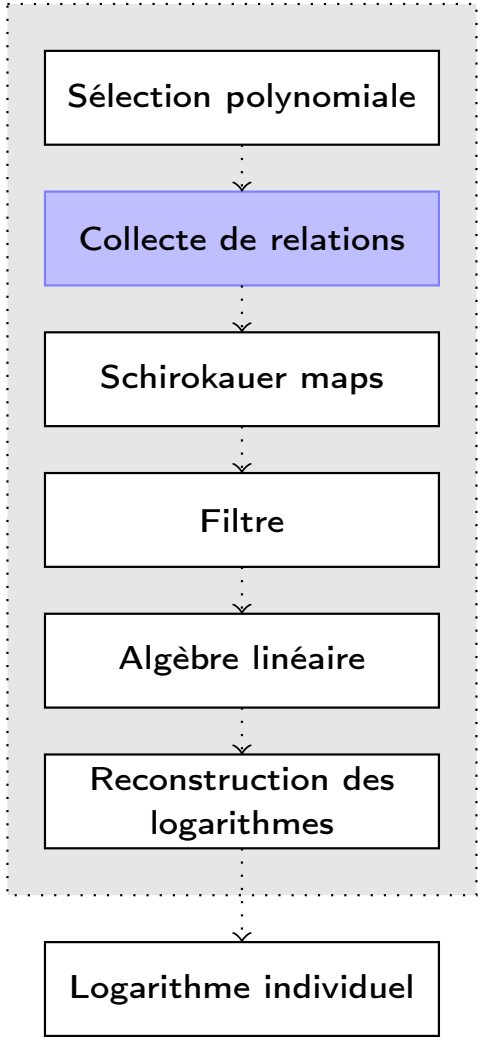

1 . C'est faux, pas de théorème de factorisation unique.

# CADO-NFS : Collecte de relations

$$Norm\left((a,b)\right) = \prod_{Norm(\mathfrak{q})<LPB} Norm\left(\mathfrak{q}\right)^{e_{\mathfrak{q}}},$$

$$\text{“}\prod_i p_i^{e_i} \equiv \prod_j q_j^{d_j}\ [p]\text{”}^{1}.$$

$$\text{“}\sum_i e_i \log_g(p_i) - \sum_j d_j \log_g(q_j) \equiv 0\ [\ell]\text{”}^{1}.$$




1 . C'est faux, pas de théorème de factorisation unique.

# CADO-NFS : Collecte de relations

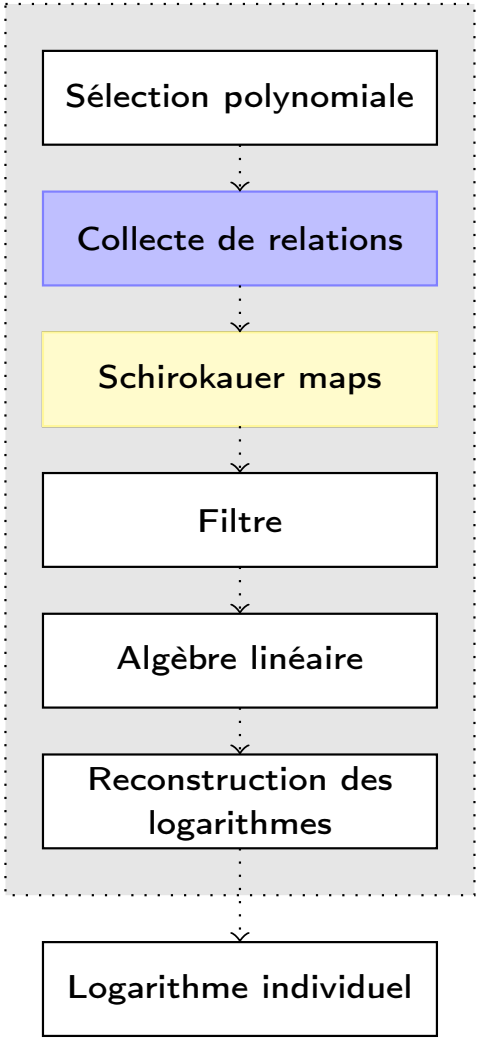
$$Norm\left((a,b)\right) = \prod_{Norm(\mathfrak{q})<LPB} Norm\left(\mathfrak{q}\right)^{e_{\mathfrak{q}}},$$

$$\text{“}\prod_i p_i^{e_i} \equiv \prod_j q_j^{d_j}\ [p]\text{”}^1.$$

$$\text{“}\sum_i e_i \log_g(p_i) - \sum_j d_j \log_g(q_j) \equiv 0\ [\ell]\text{”}^1.$$

$$\sum_i e_i \mathrm{vlog}_g(\mathfrak{p}_i) - \sum_j d_j \mathrm{vlog}_g(\mathfrak{q}_j) +$$

$$\sum_k SM_k\left(a,b\right) \mathrm{vlog}_g(SM_k) \equiv 0\ [\ell]\,.$$

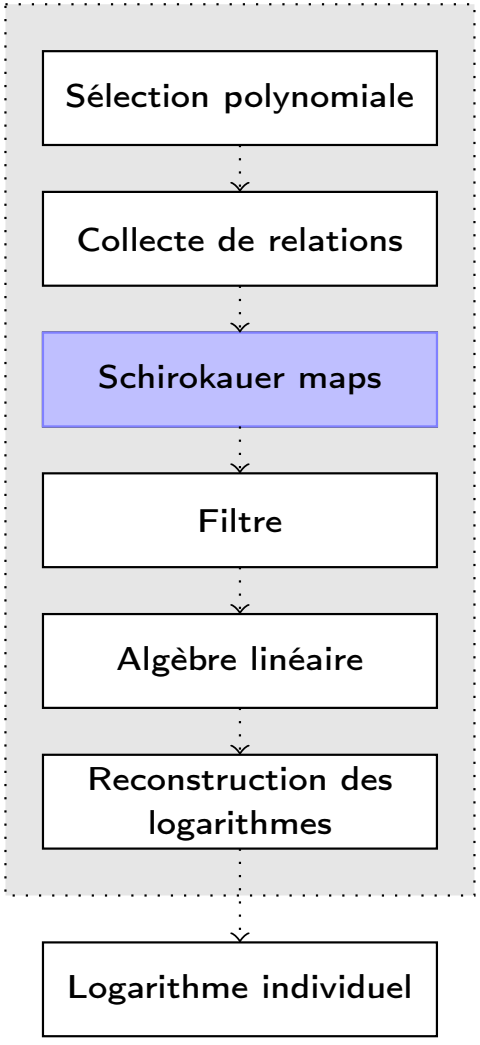


# CADO-NFS : Algèbre linéaire

$$\sum_i e_i \mathrm{vlog}_g(\mathfrak{p}_i) - \sum_j d_j \mathrm{vlog}_g(\mathfrak{q}_j) + \sum_k SM_k\,(a,b)\,\mathrm{vlog}_g(SM_k) \equiv 0\ [\ell]$$

Système d'équations linéaires.
Les logarithmes sont les inconnues.
Les exposants sont les coefficients de la matrice.

$$\begin{pmatrix}
a_{1,1} & a_{1,2} & \cdots & a_{1,n-1} & a_{1,n} & SM_{1,1} & \cdots & SM_{1,m} \\
a_{2,1} & a_{2,2} & \cdots & a_{2,n-1} & a_{2,n} & SM_{2,1} & \cdots & SM_{2,m} \\
\vdots & & & & & & & \vdots \\
\vdots & & & & & & & \vdots \\
\vdots & & & & & & & \vdots \\
a_{k,1} & a_{k,2} & \cdots & a_{k,n-1} & a_{k,n} & SM_{k,1} & \cdots & SM_{k,m}
\end{pmatrix}$$

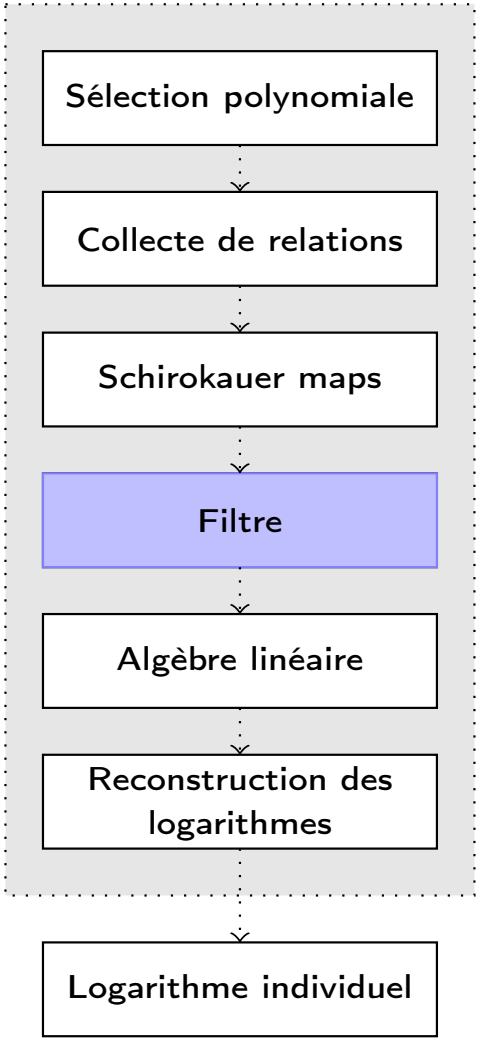


# CADO-NFS : Algèbre linéaire

$$\sum_i e_i \mathrm{vlog}_g(\mathfrak{p}_i) - \sum_j d_j \mathrm{vlog}_g(\mathfrak{q}_j) + \sum_k SM_k\,(a,b)\,\mathrm{vlog}_g(SM_k) \equiv 0\ [\ell]$$

Système d'équations linéaires.
Les logarithmes sont les inconnues.
Les exposants sont les coefficients de la matrice.
Élimination Gaussienne.

$$\begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n-1} & a_{1,n} & SM_{1,1} & \cdots & SM_{1,m} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n-1} & a_{2,n} & SM_{2,1} & \cdots & SM_{2,m} \\ \vdots & & & & & & & \vdots \\ \vdots & & & & & & & \vdots \\ \vdots & & & & & & & \vdots \\ a_{k,1} & a_{k,2} & \cdots & a_{k,n-1} & a_{k,n} & SM_{k,1} & \cdots & SM_{k,m} \end{pmatrix}$$

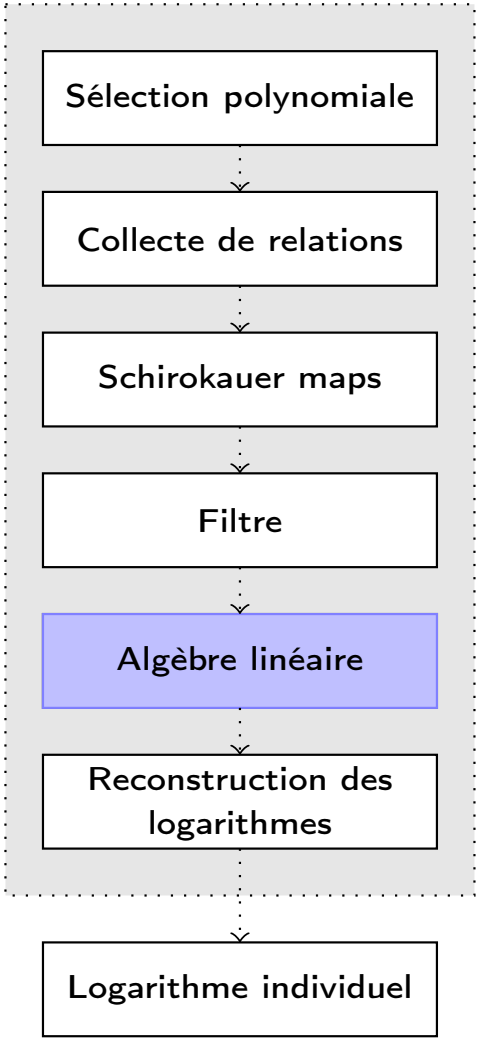


# CADO-NFS : Algèbre linéaire

$$\sum_i e_i \mathrm{vlog}_g(\mathfrak{p}_i) - \sum_j d_j \mathrm{vlog}_g(\mathfrak{q}_j) + \sum_k SM_k\,(a,b)\,\mathrm{vlog}_g(SM_k) \equiv 0\ [\ell]$$

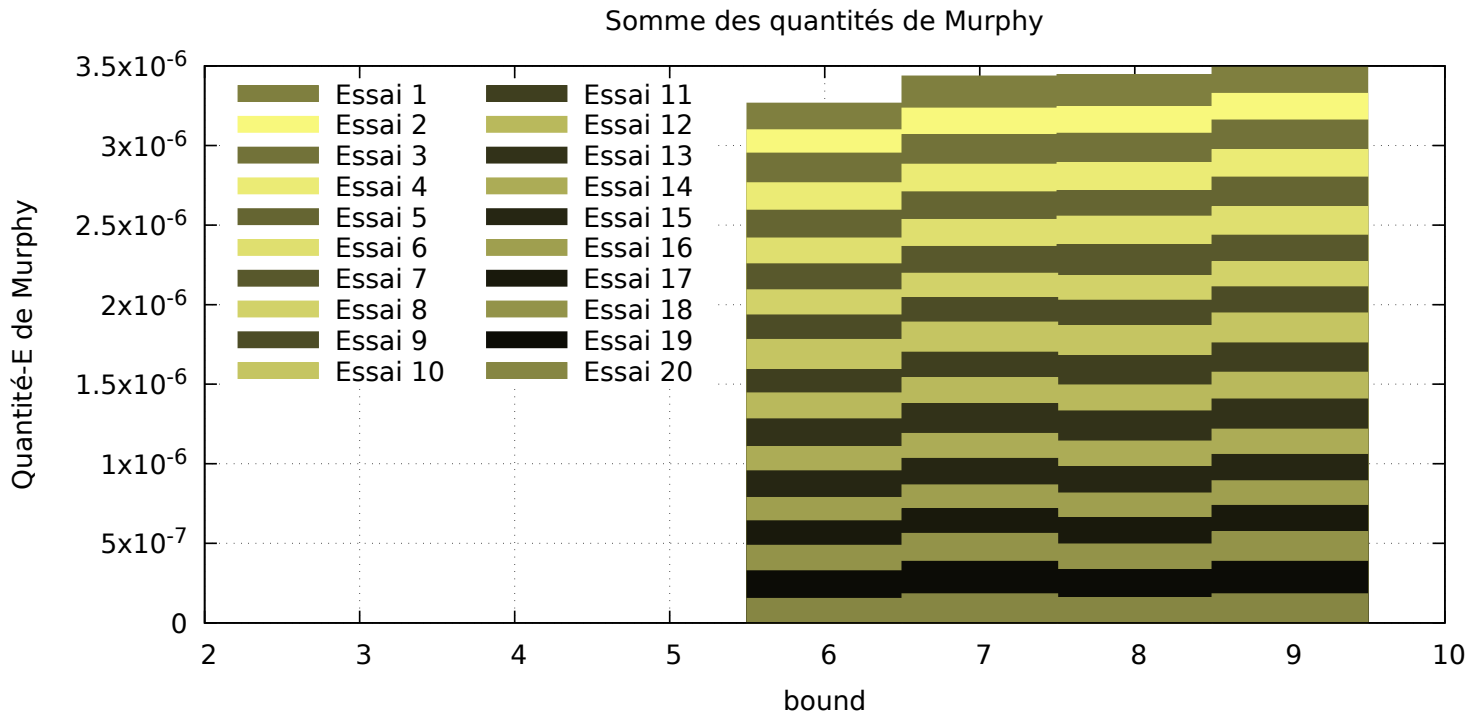
Système d'équations linéaires.
Les logarithmes sont les inconnues.
Les exposants sont les coefficients de la matrice.
Élimination Gaussienne.
Block-Wiedemann.

$$\begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n-1} & a_{1,n} & SM_{1,1} & \cdots & SM_{1,m} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n-1} & a_{2,n} & SM_{2,1} & \cdots & SM_{2,m} \\ \vdots & & & & & & & \vdots \\ \vdots & & & & & & & \vdots \\ \vdots & & & & & & & \vdots \\ a_{k,1} & a_{k,2} & \cdots & a_{k,n-1} & a_{k,n} & SM_{k,1} & \cdots & SM_{k,m} \end{pmatrix}$$

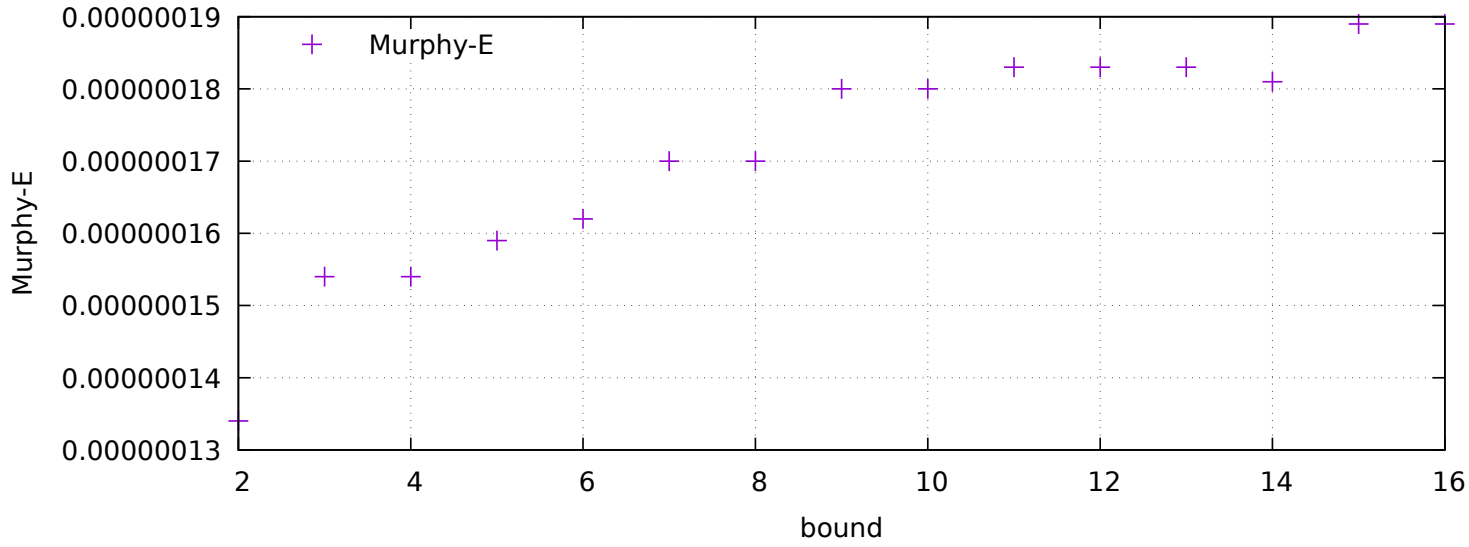


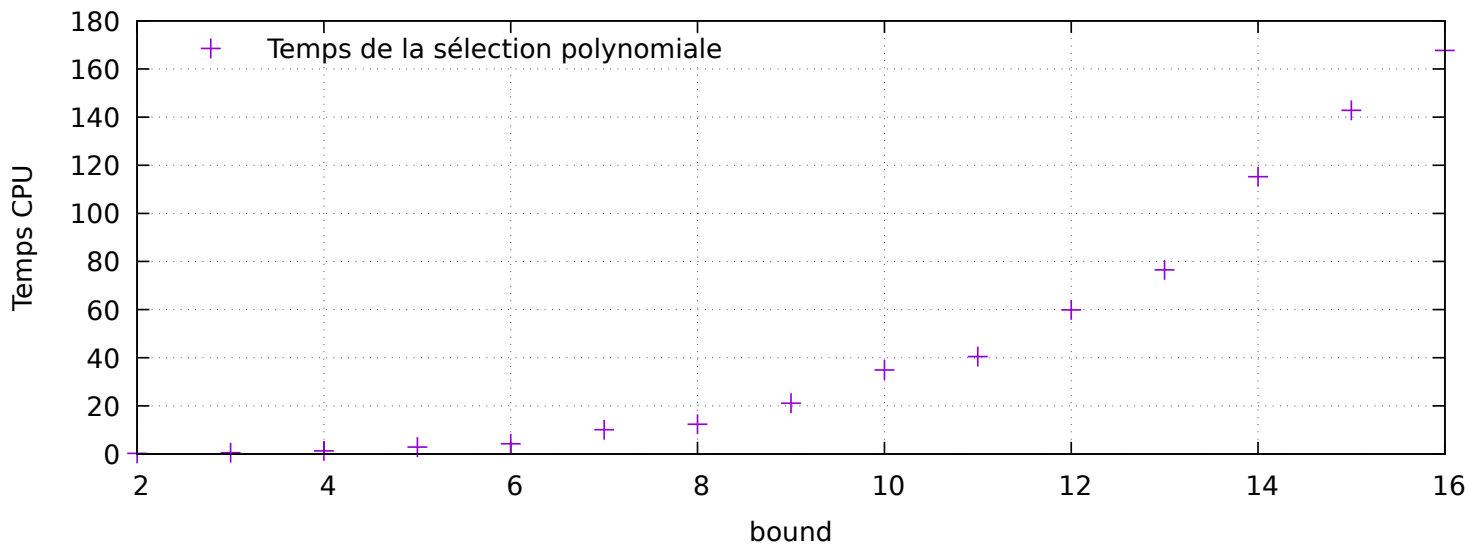
# Calculs en moyenne

Calculs en moyenne pour adapter les paramètres à un maximum de cas ou vérifier la pertinence du jeu de paramètres sur un grand nombre de cas.



# Paramétrage sélection polynomiale Joux-Lercier temps



# Exemple : paramétrage de la collecte de relations

Étude avec `las` : unités de travail indépendantes.
Paramètres :

```
lpb0      lpb1
lim0      lim1
mfb0      mfb1
lambda0   lambda1
```

```
Total 9054 reports [0.00128s/r, 18.9r/sq] in 13.6
elapsed s [85.2% CPU]
```

Score :

$$\frac{\text{relations par spécial-q}}{\text{temps écoulé} \times \text{pourcentage de temps CPU}}$$

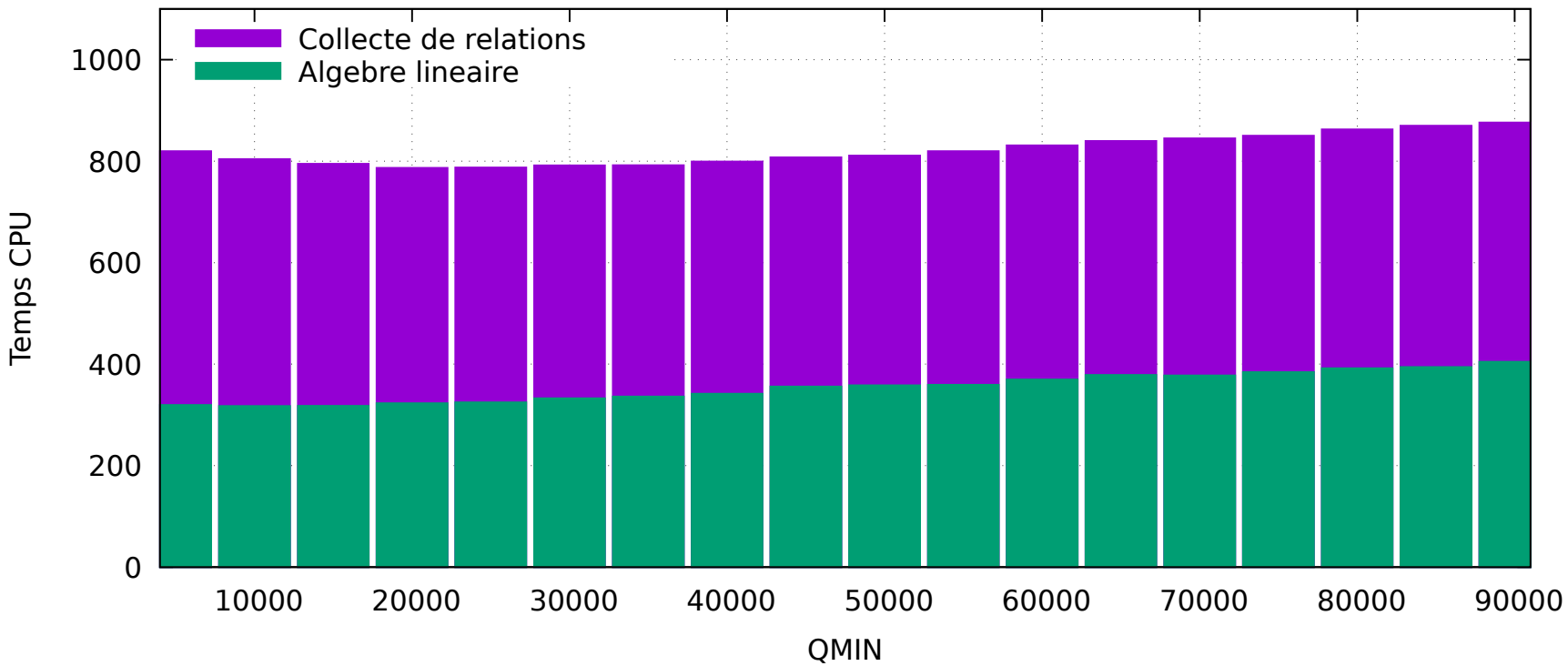
# Exemple : paramétrage de la collecte de relations

Score général :

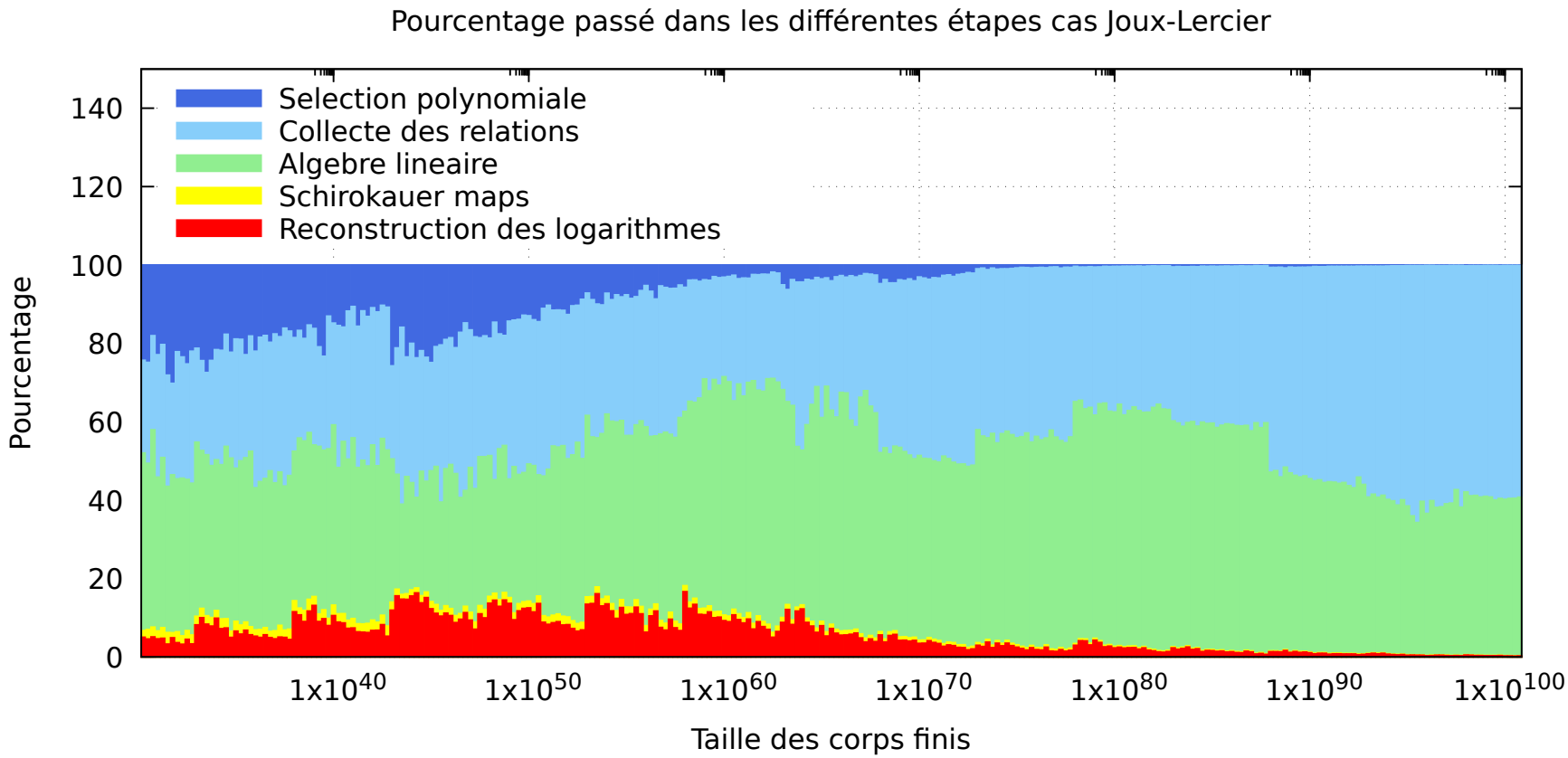
$$\frac{\text{score}}{\pi\left(2^{\texttt{lpb0}}\right) + \pi\left(2^{\texttt{lpb1}}\right)}.$$

| Rang | `lpb` | `lim` | rels/spécial-q | score | score général |
|---|---|---|---|---|---|
| 1 | 18, 18 | 52428, 52428 | 19 | 1.66138 | $3.61169 \times 10^{-5}$ |
| 2 | 19, 19 | 52428, 52428 | 39.8 | 2.95939 | $3.41023 \times 10^{-5}$ |
| 3 | 18, 19 | 52428, 52428 | 27.9 | 2.24091 | $3.37537 \times 10^{-5}$ |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |
| 9 | 17, 17 | 26214, 39321 | 6 | 0.65337 | $2.66659 \times 10^{-5}$ |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |
| 15 | 20, 20 | 314572, 104857 | 72.5 | 3.46641 | $2.11302 \times 10^{-5}$ |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |
| 23 | 16, 16 | 26214, 32768 | 1.6 | 0.19216 | $1.46863 \times 10^{-5}$ |

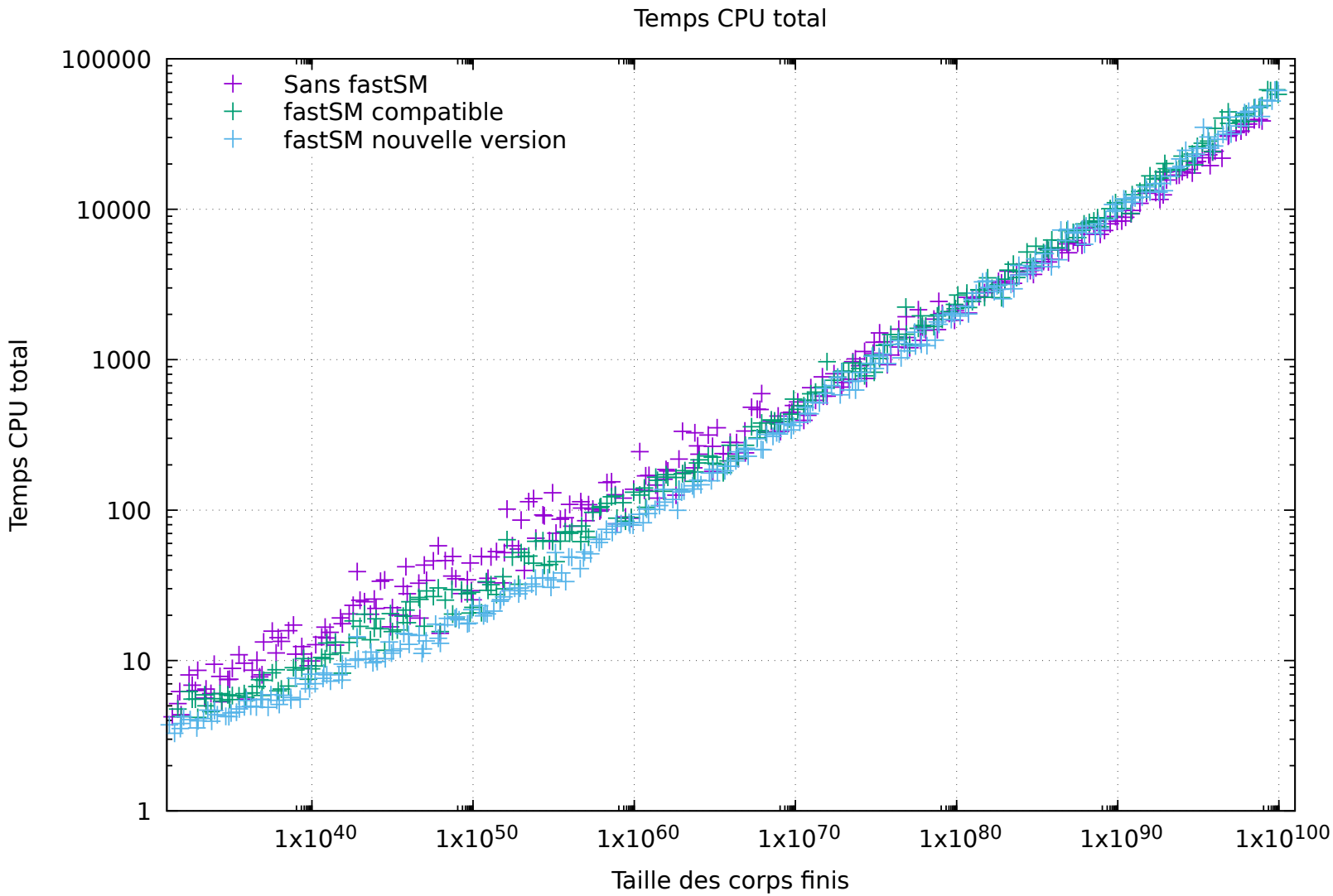
# Exemple : Collecte pour l'algèbre linéaire



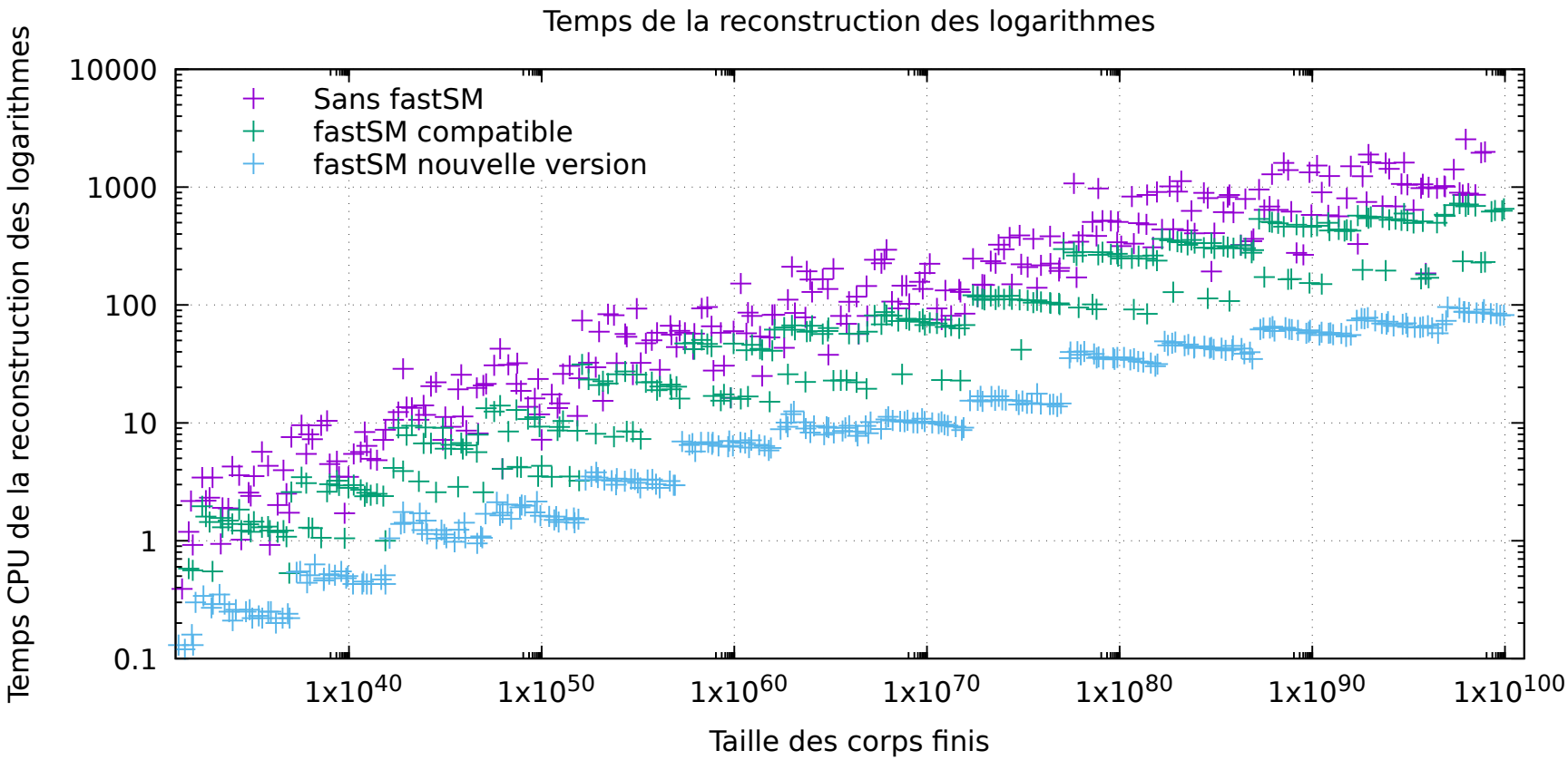
# Exemple : Temps et petites tailles



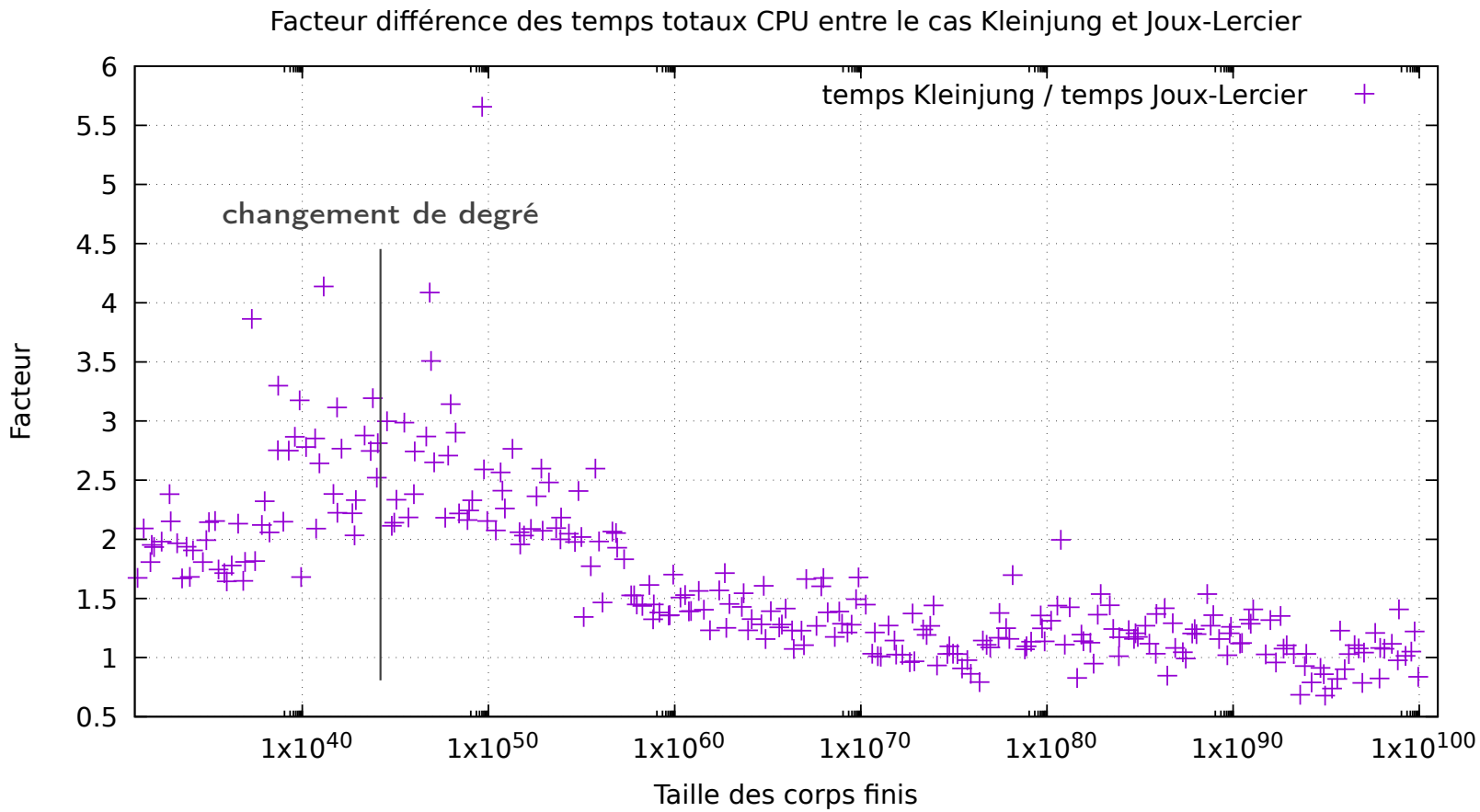
# Solutions au problème des Schirokauer maps



# Temps reconstruction des logarithmes



# Sélection polynomiale Joux-Lercier et Kleinjung



# Comparaison sans biais