Décodage adaptatif pour les systèmes multimodulaires redondants

Clément PERNET
en collaboration avec
M. Khonji, J-L. Roch, T. Roche et T. Stalinski
INRIA-MOAIS, LIG, Grenoble Université

Groupe de travail ARENAIRE, LIP, ENS Lyon
jeudi 25 mars 2010

Plan

Plan

Calculs exacts haute performance

Domaines de calcul de base

- $\mathbb{Z}, \mathbb{Q}$ $\Rightarrow$taille variable
- $\mathbb{Z}_p, \mathrm{GF}(p^k)$ $\Rightarrow$arithmétique propre
- $K[X]$ pour $K = \mathbb{Z}, \mathbb{Z}_p, ...$
- ...

Calculs exacts haute performance

Domaines de calcul de base

- $\mathbb{Z}, \mathbb{Q}$ $\Rightarrow$taille variable
- $\mathbb{Z}_p, \mathrm{GF}(p^k)$ $\Rightarrow$arithmétique propre
- $K[X]$ pour $K = \mathbb{Z}, \mathbb{Z}_p, \ldots.$
- ...

Nombreux domaines d'application:

Théorie des nombres:

- calcul de tables de courbes elliptiques, de formes modulaires,
- test de conjectures

Crypto:

- Attaques algébriques (bases de Groebner, cribles quadratiques, calcul d'index,...)
- Recherches de grands nombres premiers

Théorie des graphes: tests de conjectures (isomorphisme,...)

Votre problème favori, ici! ....

Plan

Calcul parallèle

(Renouveau du) calcul parallèle

PC individuels $\Rightarrow$multi/many cores

- Fin de la course à la fréquence CPU
- Multi-core: 2, 4, 6, ...
- Many-cores: futur proche, déjà dans les GPU's
- multi-GPU

Calcul parallèle

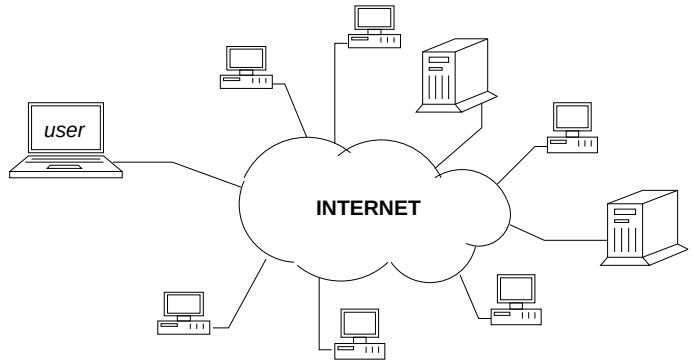
(Renouveau du) calcul parallèle

PC individuels $\Rightarrow$multi/many cores

- Fin de la course à la fréquence CPU
- Multi-core: 2, 4, 6, ...
- Many-cores: futur proche, déjà dans les GPU's
- multi-GPU

HPC $\Rightarrow$Global computing

- servers
- clusters, grilles,
- volunteer computing, Peer to Peer
- cloud computing



$\Rightarrow$Tendance générale vers le calcul massivement parallèle

Calcul volontaire et pair à pair

Projets populaires:

- Mersenne Prime search
- SETI@Home
- Folding@Home/BOINC :
  $\Rightarrow$1 Petaflops (670 000 PS3s) in 2007
- ...

$\Rightarrow$Quel niveau de confiance?

Calcul ambiant

[Above the clouds, a Berkeley view on Cloud Computing, Feb 09]

Accord difficile sur la définition

- Software As A Service
- Centre de calculs fournissant des ressources de type *a pay as you go*

Calcul ambiant

[Above the clouds, a Berkeley view on Cloud Computing, Feb 09]

Accord difficile sur la définition

- Software As A Service
- Centre de calculs fournissant des ressources de type *a pay as you go*

The interesting thing about Cloud Computing is that we've redefined Cloud Computing to include everything that we already do... I don't understand what we would do differently in the light of Cloud Computing other than change the wording of some of our ads.

Larry Ellison

It's stupidity, It's worse than stupidity: it's a marketing hype campaign.

R. Stallman

Calcul ambiant

Intérêt:

- illusion de ressources infinies de calcul
- pas d'investissement initial
- processeur *à l'heure*, stockage *à la journée*

Calcul ambiant

Intérêt:

- illusion de ressources infinies de calcul
- pas d'investissement initial
- processeur *à l'heure*, stockage *à la journée*

$\Rightarrow$pour les calculs massivement parallèles:

Nouvelle *associativité des coûts* :

100 ordinateurs pendant 1h $\equiv$ 1 ordinateur pendant 100h

$\Rightarrow$Quel niveau de confiance?

Plan

Tolérance aux fautes

De différents types:

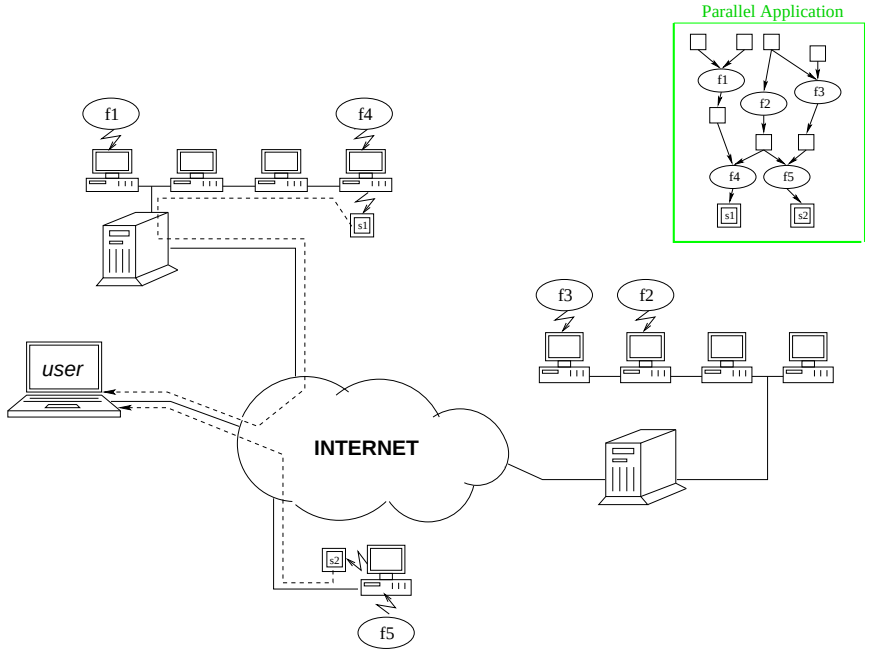
- Pannes franches (crash,...)
- Congestion de réseau
- Attaques malicieuses

$\Rightarrow$Modèle de la faute Byzantine (pas toujours défaillant)

Calcul ambiant

- Grilles et P2P: Allocation transparente des ressources à des utilisateurs authentifiés

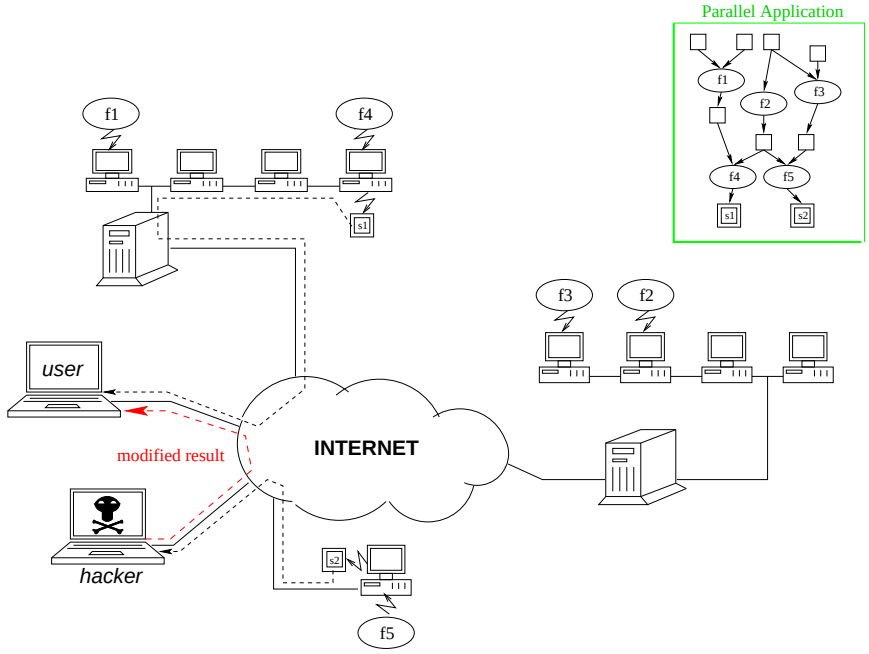
  ordonnancement supportant la connection/déconnection de ressources



Calcul ambiant et tâches forgées

- Grilles et P2P: Allocation transparente des ressources à des utilisateurs authentifiés

  ordonnancement supportant la connection/déconnection de ressources



- Mais une tâche peut être **forgée** $\iff f(\text{ données }) \neq \hat{f}(\text{ données })$
- la corruption peut affecter *de nombreuses* tâches [e.g. client patché dans SETI@home]

Etat de l'art pour la certification de résultats

- Vise principalement les programmes formés par des tâches indépendantes
- Approche générale: *Réplication*
  - Vote [*e.g.* BOINC, SETI@home]
  - Spot-checking [Germain-Playez03, basé sur Wald test]
  - rendez-vous et redémarrage
    $\Rightarrow$30' à 1h sur les grand clusters!
  - blacklisting, Tolérance aux pannes basées sur la crédibilité [Sarmenta03]
  - Exécution partielle sur ressources sûres [Gao-Malewicz04]

Etat de l'art pour la certification de résultats

- Vise principalement les programmes formés par des tâches indépendantes
- Approche générale: *Réplication*
  - Vote [*e.g.* BOINC, SETI@home]
  - Spot-checking [Germain-Playez03, basé sur Wald test]
  - rendez-vous et redémarrage
    $\Rightarrow$30' à 1h sur les grand clusters!
  - blacklisting, Tolérance aux pannes basées sur la crédibilité [Sarmenta03]
  - Exécution partielle sur ressources sûres [Gao-Malewicz04]
- Approche spécifiques: *vérification de post-condition* sur les résultats
  - Eg: tri $\mathcal{O}(n \log n)$ – Verificateur simple $\mathcal{O}(n)$ [Blum97]
  - **Approche la plus efficace, quand elle est possible!**

  Mais pas de garantie sur le résultat sans hypothèse sur l'attaque

Approche: attaques massives vs localisées

- En pratique:
  - pour la plupart des exécutions, seulement quelques tâches forgées!
    $\hookrightarrow$ réplication totale inutile et trop coûteuse

- Mais, **pas de confiance aveugle**:
  - falsification à grande échelle possible
    $\hookrightarrow$ doit être détectée par une vérification de confiance de tâches choisies aléatoirement:
    Extended Monte-Carlo Certification **EMCT**

    [Krings&al 06]
  - peu de falsifications possibles
    $\hookrightarrow$ peuvent être corrigées efficacement par **ABFT**(algorithm-based fault tolerance)

    [Beckman 93, Plank&al 97, Saha 2006]

Approche: attaques massives vs localisées

- En pratique:
  - pour la plupart des exécutions, seulement quelques tâches forgées!
    $\hookrightarrow$ réplication totale inutile et trop coûteuse

- Mais, **pas de confiance aveugle**:
  - falsification à grande échelle possible
    $\hookrightarrow$ doit être détectée par une vérification de confiance de tâches choisies aléatoirement:
    Extended Monte-Carlo Certification **EMCT**

    [Krings&al 06]
  - peu de falsifications possibles
    $\hookrightarrow$ peuvent être corrigées efficacement par **ABFT**(algorithm-based fault tolerance)

    [Beckman 93, Plank&al 97, Saha 2006]

ABFT: Algorithmic Based Fault Tolerance

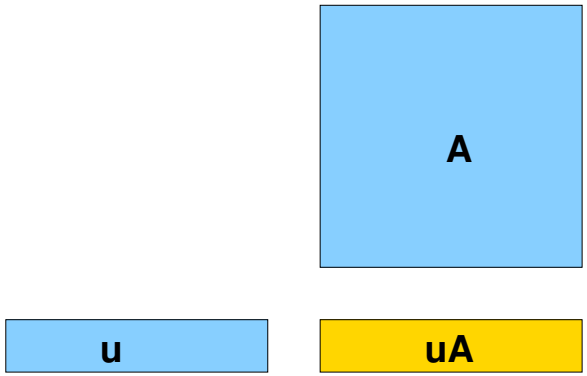
Idée: incorporer la redondance dans l'algorithme

$\Rightarrow$utiliser des propriétés spécifiques au problème

ABFT: Algorithmic Based Fault Tolerance

Idée: incorporer la redondance dans l'algorithme
$\Rightarrow$utiliser des propriétés spécifiques au problème

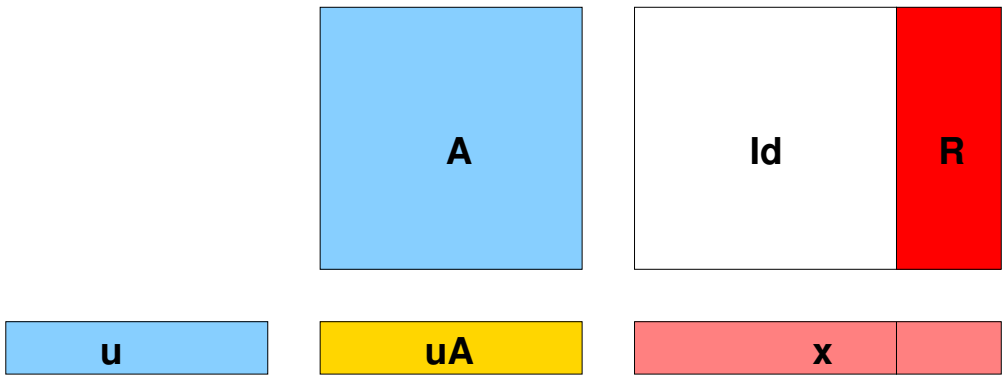
Example: Produit matrice vecteur [Dongarra 2006]



ABFT: Algorithmic Based Fault Tolerance

Idée: incorporer la redondance dans l'algorithme
$\Rightarrow$utiliser des propriétés spécifiques au problème

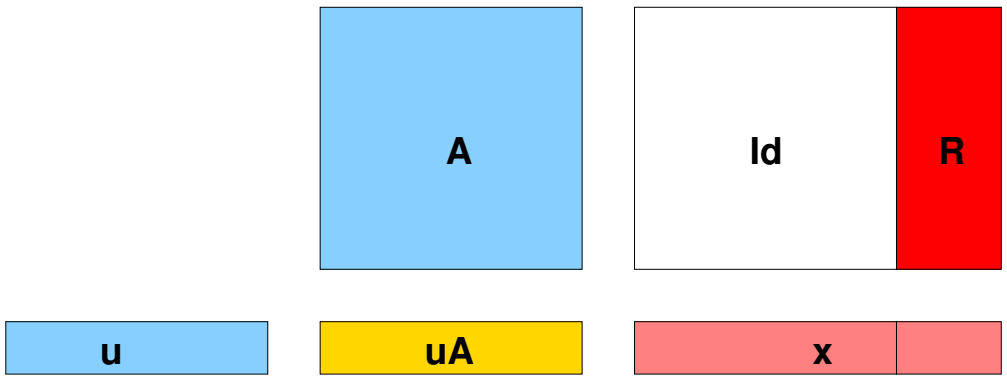
Example: Produit matrice vecteur [Dongarra 2006]



ABFT: Algorithmic Based Fault Tolerance

Idée: incorporer la redondance dans l'algorithme
$\Rightarrow$utiliser des propriétés spécifiques au problème

Example: Produit matrice vecteur [Dongarra 2006]



- précalcule le produit $B = A \times \begin{bmatrix} I & R \end{bmatrix}$
- calcule $x = uB$ en parallèle
- décode/corrige x

Plan

Algèbre linéaire exacte

Mathematics is the art of reducing any problem to linear algebra

W. Stein

$\Rightarrow$La brique de base à optimiser pour de nombreuses applications

Algèbre linéaire exacte

Mathematics is the art of reducing any problem to linear algebra

W. Stein

$\Rightarrow$La brique de base à optimiser pour de nombreuses applications

- Produit de matrices dans GF(p)
- Éliminations: Gauss, Gram-Schmidt (LLL), ...
- Itération de Krylov
- Algorithme des restes chinois

Algèbre linéaire exacte

Mathematics is the art of reducing any problem to linear algebra

W. Stein

$\Rightarrow$La brique de base à optimiser pour de nombreuses applications

- Produit de matrices dans GF(p)
- Éliminations: Gauss, Gram-Schmidt (LLL), ...
- Itération de Krylov
- Algorithme des restes chinois

Plan

Algorithme des restes chinois

$$\mathbb{Z}/(n_1 \dots n_k)\mathbb{Z} \equiv \mathbb{Z}/n_1\mathbb{Z} \times \cdots \times \mathbb{Z}/n_k\mathbb{Z}$$

Calcul de $y = f(x)$ dans $\mathbb{Z}$

begin
  Calculer une borne β sur $\max(|f|)$;
  Tirer $n_1, \dots n_k$, premiers 2 à 2, tq. $n_1 \dots n_k > \beta$;
  for $i = 1 \dots k$ **do**
    Calculer $y_i = f(x \mod n_i) \mod n_i$
  Calculer $y = \mathrm{CRT}(y_1, \dots, y_k)$
end

$$\begin{array}{rccl} \mathrm{CRT}: & \mathbb{Z}/n_1\mathbb{Z} \times \cdots \times \mathbb{Z}/n_k\mathbb{Z} & \to & \mathbb{Z}/(n_1 \dots n_k)\mathbb{Z} \\ & (x_1, \dots, x_k) & \mapsto & \sum_{i=1}^{k} x_i \Pi_i Y_i \mod \Pi \end{array} \quad \text{où}$$

$$\begin{cases} \Pi &= \prod_{i=1}^{k} n_i \\ \Pi_i &= \Pi / n_i \\ Y_i &= \Pi_i^{-1} \mod n_i \end{cases}$$

Algorithme des restes chinois

$$\mathbb{Z}/(n_1 \dots n_k)\mathbb{Z} \equiv \mathbb{Z}/n_1\mathbb{Z} \times \cdots \times \mathbb{Z}/n_k\mathbb{Z}$$

Calcul de $y = f(x)$ dans $\mathbb{Z}$

begin
  Calculer une borne β sur $\max(|f|)$;
  Tirer $n_1, \dots n_k$, premiers 2 à 2, tq. $n_1 \dots n_k > \beta$;
  for $i = 1 \dots k$ **do**
    Calculer $y_i = f(x \mod n_i) \mod n_i$; `/* Évaluation */`
  Calculer $y = \mathrm{CRT}(y_1, \dots, y_k)$; `/* Interpolation */`
end

$$\begin{array}{rccl} \mathrm{CRT}: & \mathbb{Z}/n_1\mathbb{Z} \times \cdots \times \mathbb{Z}/n_k\mathbb{Z} & \to & \mathbb{Z}/(n_1 \dots n_k)\mathbb{Z} \\ & (x_1, \dots, x_k) & \mapsto & \sum_{i=1}^{k} x_i \Pi_i Y_i \mod \Pi \end{array} \quad \text{où}$$

$$\begin{cases} \Pi & = & \prod_{i=1}^{k} n_i \\ \Pi_i & = & \Pi / n_i \\ Y_i & = & \Pi_i^{-1} \mod n_i \end{cases}$$

Analogie avec l'évaluation/interpolation

Évaluer P en a $\leftrightarrow$ Reduire P modulo $X - a$

Analogie avec l'évaluation/interpolation

Évaluer P en a $\leftrightarrow$ Reduire P modulo $X - a$

Polynômes	
Evaluation: $P \mod X - a$ Évaluer P en a	
Interpolation: $P = \sum_{i=1}^{k} \frac{\prod_{j \neq i}(X - a_j)}{\prod_{j \neq i}(a_i - a_j)}$	

Analogie avec l'évaluation/interpolation

Évaluer P en a $\leftrightarrow$ Reduire P modulo $X - a$

Polynômes	**Entiers**
Evaluation: $P \mod X - a$ Évaluer P en a	$N \mod m$ "Évaluer" N en m
Interpolation: $P = \sum_{i=1}^{k} \frac{\prod_{j \neq i}(X - a_j)}{\prod_{j \neq i}(a_i - a_j)}$	$N = \sum_{i=1}^{k} a_i \prod_{j \neq i} m_j (\prod_{j \neq i} m_j)^{-1[m_i]}$

Analogie avec l'évaluation/interpolation

Évaluer P en a $\quad\leftrightarrow\quad$ Reduire P modulo $X - a$

Polynômes	**Entiers**
Evaluation: $P \mod X - a$ Évaluer P en a	$N \mod m$ "Évaluer" N en m
Interpolation: $P = \sum_{i=1}^{k} \frac{\prod_{j\neq i}(X-a_j)}{\prod_{j\neq i}(a_i-a_j)}$	$N = \sum_{i=1}^{k} a_i \prod_{j\neq i} m_j (\prod_{j\neq i} m_j)^{-1[m_i]}$

Analogie: complexités dans $\mathbb{Z}\leftrightarrow$ dans $K[X]$

- taille des coefficients
- $\mathcal{O}\left(\log \|\text{résultat}\| \times T_{\text{algébr.}}\right)$
- degré des polynômes
- $\mathcal{O}\left(\deg(\text{résultat}) \times T_{\text{algébr.}}\right)$

Analogie avec l'évaluation/interpolation

Évaluer P en a $\leftrightarrow$ Reduire P modulo $X - a$

Polynômes	**Entiers**
Evaluation: $P \mod X - a$ Évaluer P en a	$N \mod m$ "Évaluer" N en m
Interpolation: $P = \sum_{i=1}^{k} \frac{\prod_{j \neq i}(X - a_j)}{\prod_{j \neq i}(a_i - a_j)}$	$N = \sum_{i=1}^{k} a_i \prod_{j \neq i} m_j (\prod_{j \neq i} m_j)^{-1[m_i]}$

Analogie: complexités dans $\mathbb{Z}\leftrightarrow$ dans $K[X]$

- taille des coefficients
- $\mathcal{O}\left(\log \|\text{résultat}\| \times T_{\text{algébr.}}\right)$
- $\det(n, \|A\|) = \mathcal{O}^{\sim}(n \log |A\| \times n^{\omega})$
- degré des polynômes
- $\mathcal{O}\left(\deg(\text{résultat}) \times T_{\text{algébr.}}\right)$
- $\det(n, d) = \mathcal{O}^{\sim}(nd \times n^{\omega})$

Terminaison anticipée

Calcul dans $\mathbb{Z}$:

- borne β sur le résultat
- Choix des n_i: tels que $n_1 \dots n_k > \beta$

$\Rightarrow$algorithme déterministe

Terminaison anticipée

Calcul dans $\mathbb{Z}$:

- borne β sur le résultat
- Choix des n_i: tels que $n_1 \ldots n_k > \beta$

$\Rightarrow$algorithme déterministe

Terminaison anticipée:

- Pour chaque nouveau n_i:
  - reconstruire $y_i = f(x) \mod n_1 \ldots n_i$
  - Si $y_i = y_{i-1}$ $\Rightarrow$terminé

$\Rightarrow$probabiliste Monte Carlo

Avantage:

- nombre de moduli adaptatif selon la valeur de sortie
- Intéressant quand
  - borne pessimiste: matrices creuses, structurées, ...
  - pas de borne disponible

Plan

Les codes de résidus redondants

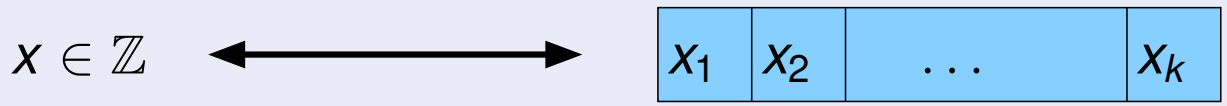
Principe:

- Utilisation des restes chinois pour la parallélisation
- Fautes byzantines affectant certains calculs modulaires
- Reconstruction tolérante aux faute
  $\Rightarrow$Tolérance aux pannes algorithmique (ABFT)

Plan

Algorithme de Mandelbaum, dans $\mathbb{Z}$

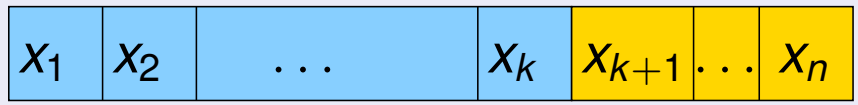
Théorème des restes chinois



où $p_1 \times \cdots \times p_k > x$ et $x_i = x \mod p_i \, \forall i$

Algorithme de Mandelbaum, dans $\mathbb{Z}$

Théorème des restes chinois

$x \in \mathbb{Z}$ 

où $p_1 \times \cdots \times p_n > x$ et $x_i = x \mod p_i \, \forall i$

Algorithme de Mandelbaum, dans $\mathbb{Z}$

Théorème des restes chinois

$x \in \mathbb{Z}$ | x_1 | x_2 | $\ldots$ | x_k | x_{k+1} | $\ldots$ | x_n |

où $p_1 \times \cdots \times p_n > x$ et $x_i = x \mod p_i \, \forall i$

Définition

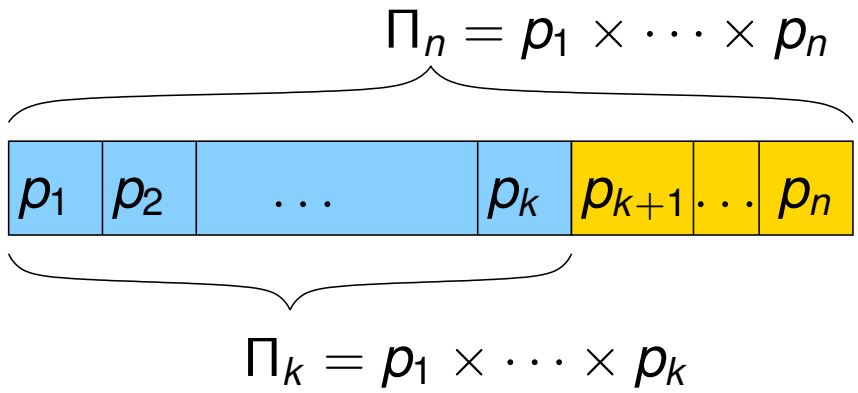
(n,k)-*code:* $C =$

$$\left\{ (x_1, \ldots, x_n) \in \mathbb{Z}_{p_1} \times \cdots \times \mathbb{Z}_{p_n} \; t.q. \; \exists x, \left\{ \begin{array}{lcl} x & < & p_1 \ldots p_k \\ x_i & = & x \mod p_i \, \forall i \end{array} \right\} \right.$$

Principe

Propriété

$$X \in C \text{ ssi } X < \Pi_k.$$



Redondance : $r = n - k$

Principe

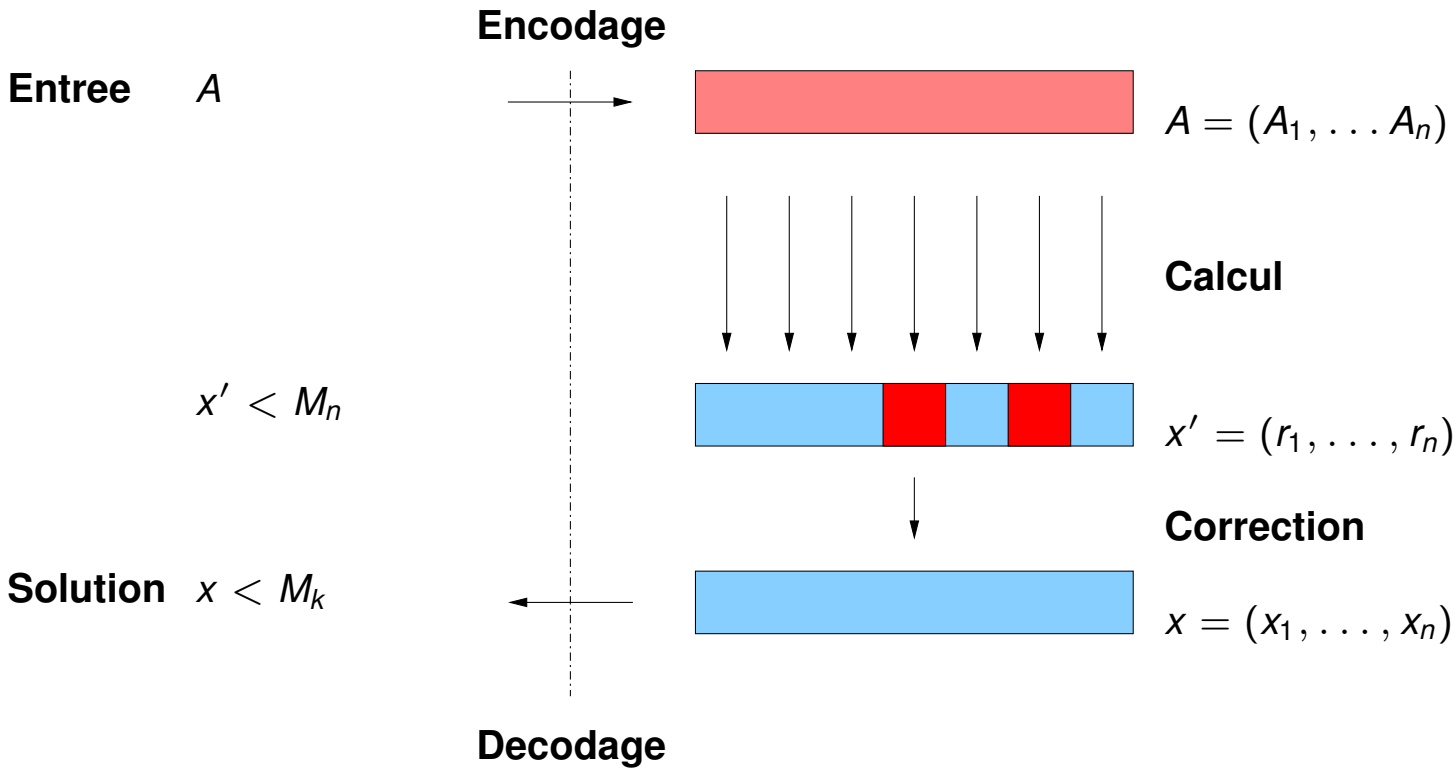
Canal de transmission $\equiv$ Calcul

Principe

Canal de transmission bruité $\equiv$ Calcul non sûr

Principe

Canal de transmission bruité $\equiv$ Calcul non sûr



Propriétés du code

Modèle d'erreur:

- Erreur: $E = X' - X$
- Support de l'erreur: $I = \{i \in 1 \dots n, E \neq 0 \mod p_i\}$
- Impact de l'erreur: $\Pi_F = \prod_{i \in I} p_i$

Propriétés du code

Modèle d'erreur:

- Erreur: $E = X' - X$
- Support de l'erreur: $I = \{i \in 1 \dots n, E \neq 0 \mod p_i\}$
- Impact de l'erreur: $\Pi_F = \prod_{i \in I} p_i$

Détecte jusqu'à r erreurs:

Si $X' = X + E$ avec $X \in C, \#I \leq r$,

$$X' > \Pi_k.$$

- Redondance $r = n - k$, distance: $r + 1$
- $\Rightarrow$peut corriger jusqu'à $\left\lfloor \frac{r}{2} \right\rfloor$ erreurs en théorie
- Plus délicat en pratique...

Correction

- $\forall i \notin I : E \mod p_i = 0$
- E est un multiple de Π_V: $E = Z\Pi_V = Z\prod_{i\notin I}$
- $\text{pgcd}(E, \Pi) = \Pi_V$

Mandelbaum 78: reconstruction rationnelle

$$\begin{aligned} X = X' - E &= X' - Z\Pi_v \\ \frac{X}{\Pi} &= \frac{X'}{\Pi} - \frac{Z}{\Pi_F} \end{aligned}$$

$\Rightarrow |\frac{X'}{\Pi} - \frac{Z}{\Pi_F}| \leq \frac{1}{2\Pi_F^2}$

$\Rightarrow \frac{Z}{\Pi_F} = \frac{E}{\Pi}$ est une réduite de $\frac{X'}{\Pi}$

$\Rightarrow$reconstruction rationnelle de $X' \mod \Pi$

Capacité de correction

Mandelbaum 78:

- 1 symbole = 1 résidu
- Algo polynomial si $e \leq (n-k)\frac{\log p_{\min} - \log 2}{\log p_{\max} + \log p_{\min}}$
- exponentiel en pire cas (perturbation aléatoire)

Goldreich Ron Sudan 99 pondération des résidus

$\Rightarrow$équivalent

Guruswami Sahai Sudan 00 polynomial invariablement

Capacité de correction

Mandelbaum 78:

- 1 symbole = 1 résidu
- Algo polynomial si $e \leq (n-k)\frac{\log p_{\min} - \log 2}{\log p_{\max} + \log p_{\min}}$
- exponentiel en pire cas (perturbation aléatoire)

Goldreich Ron Sudan 99 pondération des résidus

$\Rightarrow$équivalent

Guruswami Sahai Sudan 00 polynomial invariablement

Interprétation:

- Les erreurs ont des poids variables selon leur impact
  $\Pi_F = \prod_{i \in I} p_i$
- Exemple: $X = 20, p_1 = 2, p_2 = 3, p_3 = 101$
  - 1 erreur sur $X \mod 2$, ou $X \mod 3$, peut être corrigée
  - mais pas sur $X \mod 101$

Plan

Analogie avec les code de Reed Solomon

Gao02 Décodage des codes RS par Euclide étendu:

- Restes Chinois dans $K[X]$
- $p_i = X - a_i$
- Encodage = évalutation en a_i
- Decodage = interpolation
- Correction = Euclide étendu

Analogie avec les code de Reed Solomon

Gao02 Décodage des codes RS par Euclide étendu:

- Restes Chinois dans $K[X]$
- $p_i = X - a_i$
- Encodage = évalutation en a_i
- Decodage = interpolation
- Correction = Euclide étendu

$\Rightarrow$Généralisable pour des p_i de degrés quelconques
$\Rightarrow$notion d'impact variable selon le degré de p_i
$\Rightarrow$unification nécessaire [Sudan 01,...]

Plan

Point de vue généralisé: code d'amplitude

- Dans un anneau euclidien $\mathcal{A}$ muni d'un fonction euclidienne ν
- Distance

$$\begin{array}{rccc} \Delta : & \mathcal{A} \times \mathcal{A} & \rightarrow & \mathbb{R}_+ \\ & (x, y) & \mapsto & \sum_{i | x \neq y [P_i]} \log_2 \nu(P_i) \end{array}$$

Définition

(n,k)-code d'amplitude $C = \{x \in \mathcal{A} : \nu(x) < \kappa\}$,
$n = \log_2 \Pi, k = \log_2 \kappa$.

Point de vue généralisé: code d'amplitude

- Dans un anneau euclidien $\mathcal{A}$ muni d'un fonction euclidienne ν
- Distance

$$\begin{array}{rccc} \Delta : & \mathcal{A} \times \mathcal{A} & \to & \mathbb{R}_+ \\ & (x, y) & \mapsto & \sum_{i | x \neq y [P_i]} \log_2 \nu(P_i) \end{array}$$

Définition

(n,k)-code d'amplitude $C = \{x \in \mathcal{A} : \nu(x) < \kappa\}$, $n = \log_2 \Pi, k = \log_2 \kappa$.

Propriété (Borne Singleton)

$d > n - k$ *en général, et* $d \geq n - k + 1$ *dans* $K[X]$.

$\Rightarrow$taux de correction = amplitude maximal d'une erreur pouvant être corrigée

Intérêt

- Généralisation sur tout anneau euclidien
- Représentation naturelle des quantités d'information
- Plus besoin de trier les moduli
- Capacités de correction plus fines

Intérêt

- Généralisation sur tout anneau euclidien
- Représentation naturelle des quantités d'information
- Plus besoin de trier les moduli
- Capacités de correction plus fines
- Décodage adaptatif: tirant parti de toute la redondance disponible

Interprétation de Mandelbaum

Remarque

Reconstruction rationnelle $\Rightarrow$ *Algorithme d'Euclide Etendu partiel*

Propriété

L'algorithme d'Euclide étendu, appliqué à (E, Π) *et à* $(X' = X + E, \Pi)$, *effectue les même premières itérations jusqu'à* $r_i < \Pi_V$.

$$\begin{array}{l|l} u_{i-1}E + v_{i-1}\Pi = \Pi_v & u_{i-1}X' + v_{i-1}\Pi = r_{i-1} \\ u_iE + v_i\Pi = 0 & u_iX' + v_i\Pi = r_i \end{array}$$

$$\Rightarrow u_iX = r_i$$

Décodage par amplitude, à taux borné

Amplitude based decoder over R

Donnée: Π, X'
Donnée: $\tau \in \mathbb{R}_+ \mid \tau < \frac{\nu(\Pi)}{2}$: borne sur l'amplitude max de correction
Résultat: $X \in R$: message corrigé t.q. $\nu(X)4\tau^2 \leq \nu(\Pi)$
begin
  $\alpha_0 = 1, \beta_0 = 0, r_0 = \Pi$;
  $\alpha_1 = 0, \beta_1 = 1, r_1 = X'$;
  $i = 1$;
  while $(\nu(r_i) > \nu(\Pi)/2\tau)$ **do**
    Soit $r_{i-1} = q_i r_i + r_{i+1}$ la div. euclidienne de r_{i-1} par r_i;
    $\alpha_{i+1} = \alpha_{i-1} - q_i \alpha_i$;
    $\beta_{i+1} = \beta_{i-1} - q_i \beta_i$;
    $i = i + 1$;
  return $X = -\frac{r_i}{\beta_i}$
end

- atteint le taux de correction maximal

Décodage par amplitude, à taux borné

Amplitude based decoder over R

Donnée: Π, X'
Donnée: $\tau \in \mathbb{R}_+ \mid \tau < \frac{\nu(\Pi)}{2}$: borne sur l'amplitude max de correction
Résultat: $X \in R$: message corrigé t.q. $\nu(X)4\tau^2 \leq \nu(\Pi)$
begin
  $\alpha_0 = 1, \beta_0 = 0, r_0 = \Pi;$
  $\alpha_1 = 0, \beta_1 = 1, r_1 = X';$
  $i = 1;$
  while $(\nu(r_i) > \nu(\Pi)/2\tau)$ **do**
    Soit $r_{i-1} = q_i r_i + r_{i+1}$ la div. euclidienne de r_{i-1} par r_i;
    $\alpha_{i+1} = \alpha_{i-1} - q_i \alpha_i;$
    $\beta_{i+1} = \beta_{i-1} - q_i \beta_i;$
    $i = i + 1;$
  return $X = -\frac{r_i}{\beta_i}$
end

- atteint le taux de correction maximal
- nécessite une connaissance à priori de τ

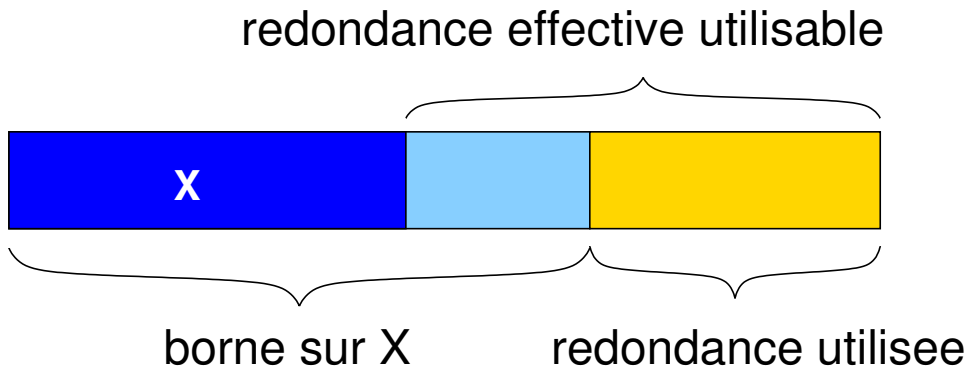
  $\Rightarrow$Comment rendre le taux de correction adaptatif?

Plan

Approche adaptative

Buts multiples:

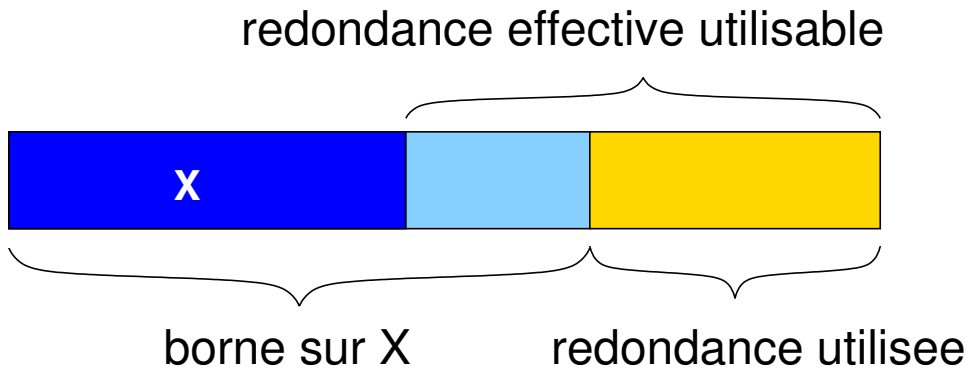
- A n fixé, taux de correction fixé par la borne sur X
  $\Rightarrow$taux de correction pessimiste
  $\Rightarrow$comment tirer parti de toute la redondance disponible?



Approche adaptative

Buts multiples:

- A n fixé, taux de correction fixé par la borne sur X
  $\Rightarrow$taux de correction pessimiste
  $\Rightarrow$comment tirer parti de toute la redondance disponible?



- Permettre la terminaison anticipée: n variable et borne inconnue

Plan

Première approche adaptative

Critère de terminaison dans Euclide Etendu:

- $\alpha_{i+1}\Pi - \beta_{i+1}E = 0$
  $\Rightarrow E = \alpha_{i+1}\Pi/\beta_{i+1}$
  $\Rightarrow$tester si β_j divise Π
- tester la cohérence des bornes:$\nu(X) \leq \frac{\nu(\Pi)}{4\nu(\beta_j)^2}$
- Mais plusieurs candidats possibles
  $\Rightarrow$discrimination par une post-condition sur le résultat

Première approche adaptative

Critère de terminaison dans Euclide Etendu:

- $\alpha_{i+1}\Pi - \beta_{i+1}E = 0$
  $\Rightarrow E = \alpha_{i+1}\Pi/\beta_{i+1}$
  $\Rightarrow$tester si β_j divise Π
- tester la cohérence des bornes:$\nu(X) \leq \frac{\nu(\Pi)}{4\nu(\beta_j)^2}$
- Mais plusieurs candidats possibles
  $\Rightarrow$discrimination par une post-condition sur le résultat

Exemple

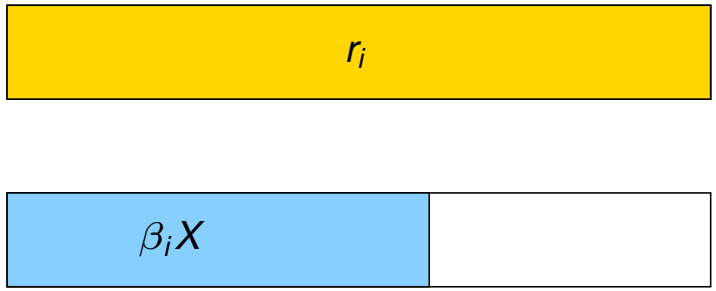
p_i	3	5	7
x_i	2	3	2

- $x = 23$ *avec 0 erreur*
- $x = 2$ *avec 1 erreur*

Plan

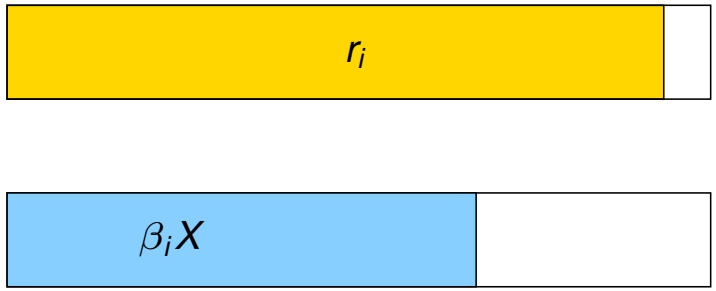
Détection d'un saut

$$\alpha_i \Pi - \beta_i X = r_i$$



Détection d'un saut

$$\alpha_i \Pi - \beta_i X = r_i$$

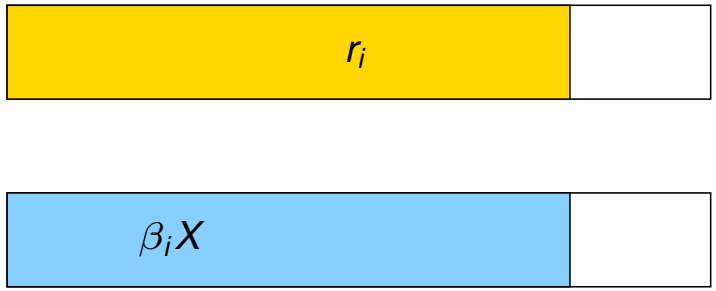


Détection d'un saut

$$\alpha_i \Pi - \beta_i X = r_i$$

Détection d'un saut

$$\alpha_i \Pi - \beta_i X = r_i$$

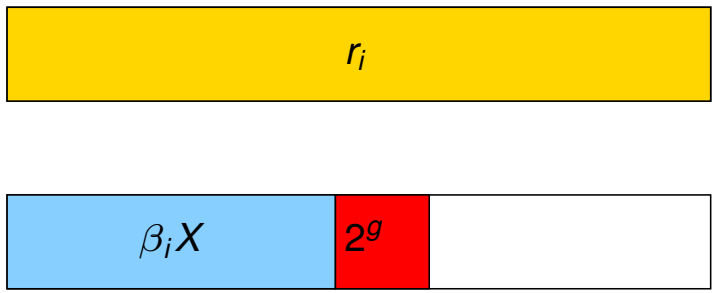


$X = -r_i/\beta_i$

- A l'itération finale: $\nu(r_i) \approx \nu(\beta_i X)$
- Si besoin, saut entre r_{i-1} et r_i.

Détection d'un saut

$$\alpha_i \Pi - \beta_i X = r_i$$

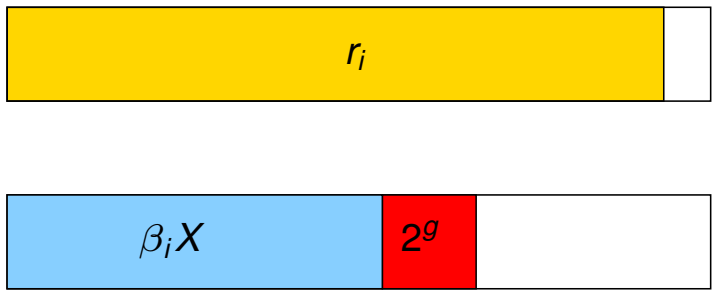


$X = -r_i/\beta_i$

- A l'itération finale: $\nu(r_i) \approx \nu(\beta_i X)$
- Si besoin, saut entre r_{i-1} et r_i.
- $\Rightarrow$Ajout d'un *blanc* 2^g pour détecter le saut de 2^g

Détection d'un saut

$$\alpha_i \Pi - \beta_i X = r_i$$

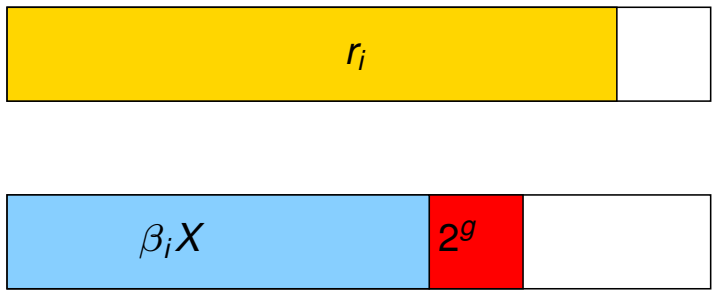


$X = -r_i/\beta_i$

- A l'itération finale: $\nu(r_i) \approx \nu(\beta_i X)$
- Si besoin, saut entre r_{i-1} et r_i.
- $\Rightarrow$Ajout d'un *blanc* 2^g pour détecter le saut de 2^g

Détection d'un saut

$$\alpha_i \Pi - \beta_i X = r_i$$

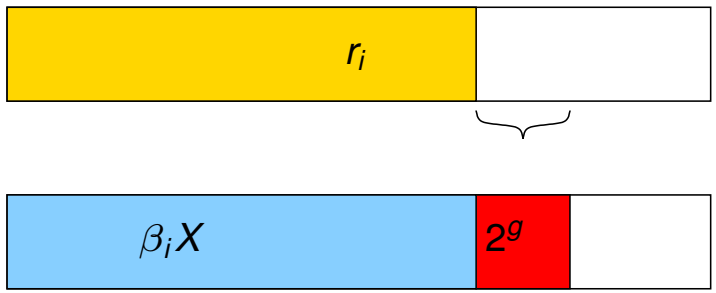


$X = -r_i/\beta_i$

- A l'itération finale: $\nu(r_i) \approx \nu(\beta_i X)$
- Si besoin, saut entre r_{i-1} et r_i.
- $\Rightarrow$Ajout d'un *blanc* 2^g pour détecter le saut de 2^g

Détection d'un saut

$$\alpha_i \Pi - \beta_i X = r_i$$

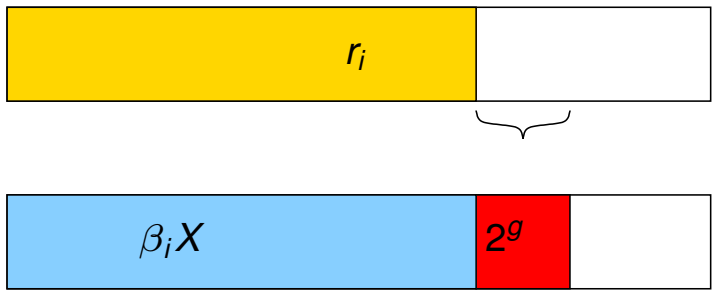


$X = -r_i/\beta_i$

- A l'itération finale: $\nu(r_i) \approx \nu(\beta_i X)$
- Si besoin, saut entre r_{i-1} et r_i.
- $\Rightarrow$Ajout d'un *blanc* 2^g pour détecter le saut de 2^g

Détection d'un saut

$$\alpha_i \Pi - \beta_i X = r_i$$



$X = -r_i/\beta_i$

- A l'itération finale: $\nu(r_i) \approx \nu(\beta_i X)$
- Si besoin, saut entre r_{i-1} et r_i.
- $\Rightarrow$Ajout d'un *blanc* 2^g pour détecter le saut de 2^g

Propriété

- *Perte de capacité de correction: très faible en pratique*
- *Test de la divisibilité pour les candidats restants*
- *Réduit fortement le nombre de tests de divisibilité*

Plan

Expérimentations

Taille de l'erreur	10	50	100	200	500	1000
$g = 2$	$1/446$	$1/765$	$1/1118$	$2/1183$	$2/4165$	$1/7907$
$g = 3$	$1/244$	$1/414$	$1/576$	$2/1002$	$2/2164$	$1/4117$
$g = 5$	$1/53$	$1/97$	$1/153$	$2/262$	$1/575$	$1/1106$
$g = 10$	$1/1$	$1/3$	$1/9$	$1/14$	$1/26$	$1/35$
$g = 20$	$1/1$	$1/1$	$1/1$	$1/1$	$1/1$	$1/1$

Table: Nombre de candidats dans l'algorithme du saut: c/d signifie que d candidats sont apparus avec un saut $> 2^g$, et c parmis eux ont passé le test de divisibilité. $n \approx 6001$ (3000 moduli), $\kappa \approx 201$ (100 moduli).

Expérimentations

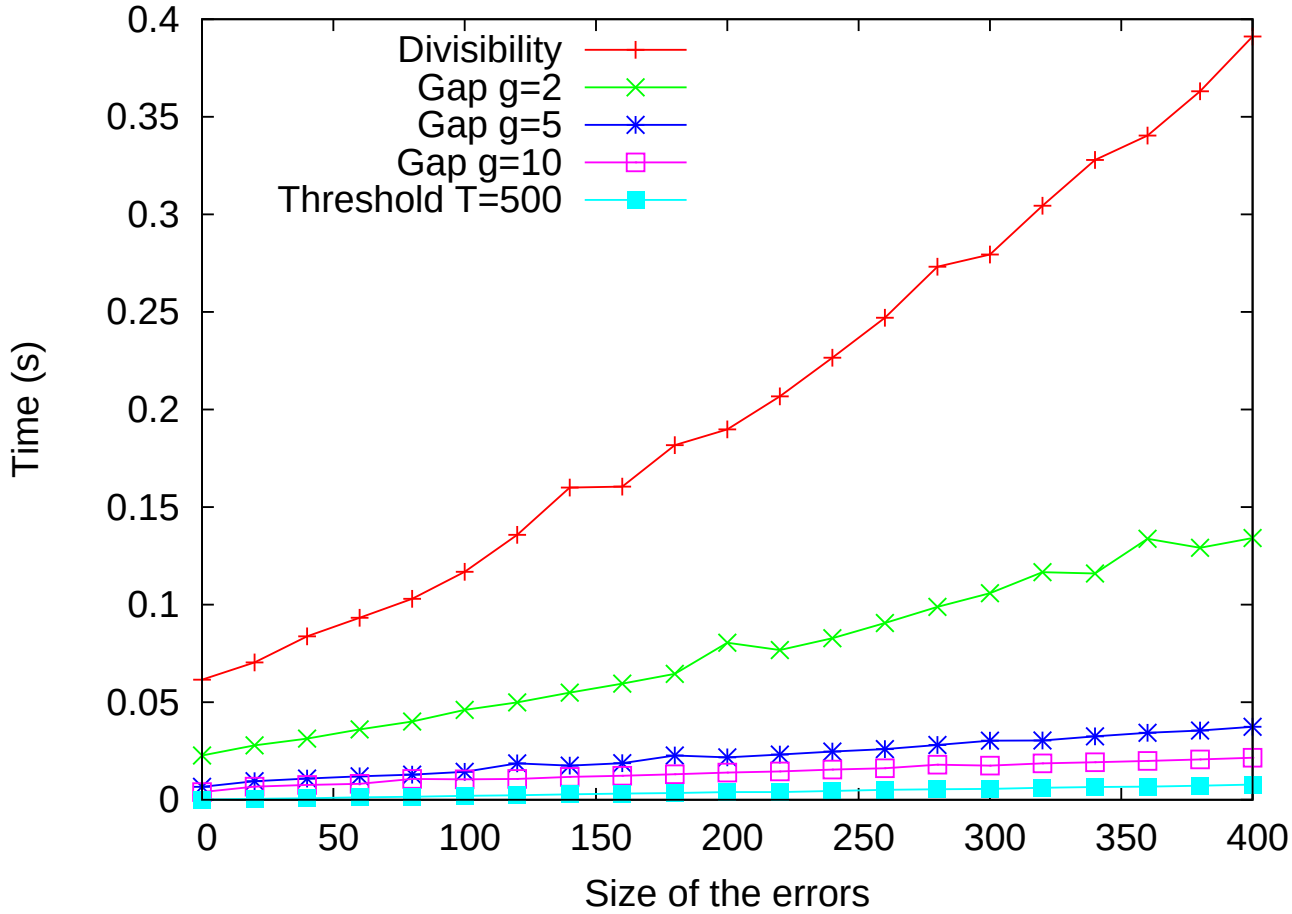


Figure: Comparaison pour $n \approx 26\,016$ ($m = 1300$ moduli de 20 bits), $\kappa \approx 6001$ (300 moduli) et $\tau \approx 10007$ (environ 500 moduli).

Expérimentations

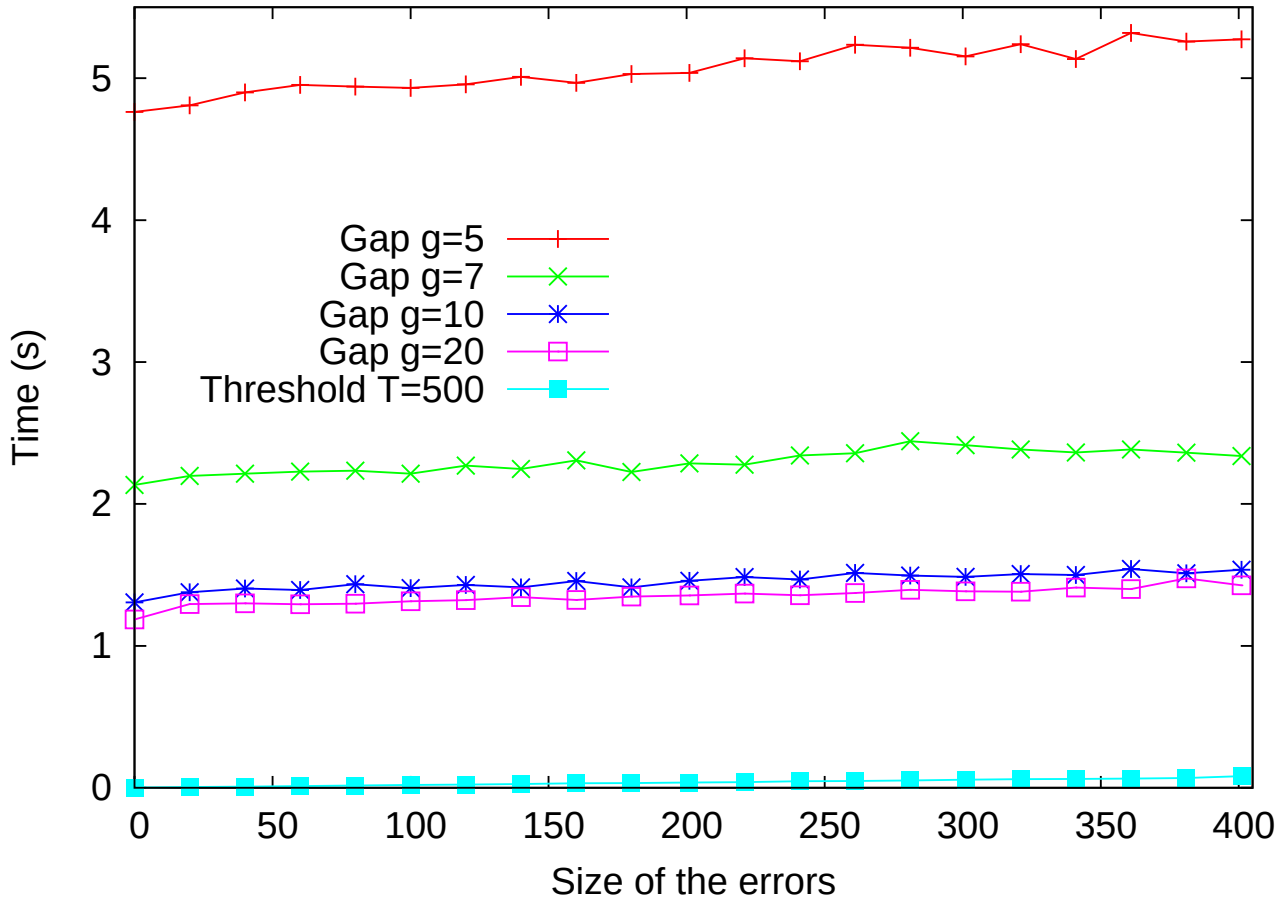


Figure: Comparaison pour $n \approx 200\,917$ ($m = 10000$ moduli de 20 bits), $\kappa \approx 17\,0667$ (8500 moduli) et $\tau \approx 10498$ (500 moduli).

Gap: algorithme d'Euclide jusqu'au bout $\Rightarrow$surcoût

Plan

Terminaison anticipée

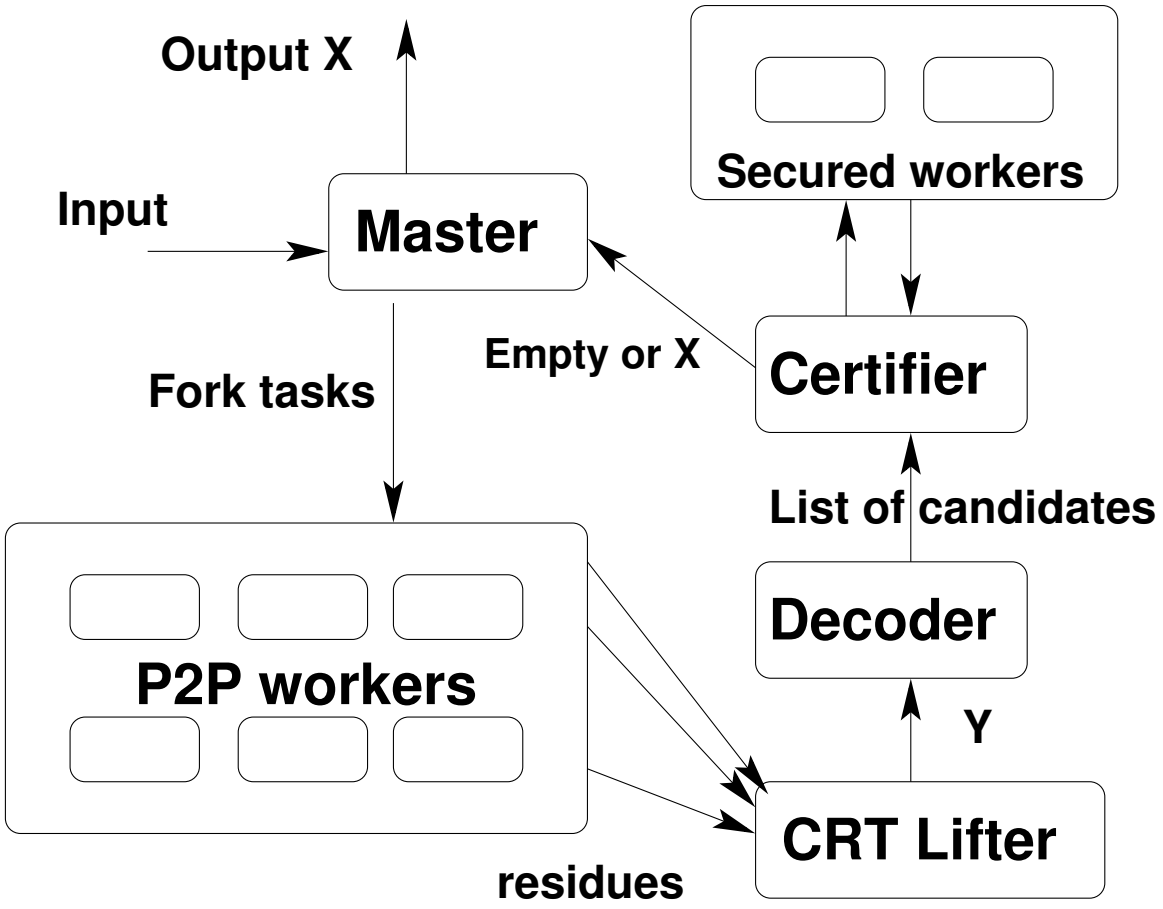


Figure: Principe d'un calcul distribué tolérant aux fautes avec terminaison anticipée

Conclusion

Nouvelle métrique pour les codes de résidus redondants:

- Unification
- Bornes plus fines sur les capacités de correction
- permet le décodage adaptatif au maximum des capacités

Décodage adaptatif et terminaison anticipée

- Plusieurs approches
- Méthode du saut: surcoût limité, performances meilleures
- Insertion dans un schéma global de terminaison anticipée

Perspective

- Explication théorique de l'efficacité du saut (distribution moyenne des q_i dans Euclide Étendu).
- Meilleurs taux de corrections pour $\mathbb{Z}$et $K[X]$ seulement.
- Appliquer au décodage par liste [Sudan, Guruswami]