# Compter sur ordinateur et autres calculs : toujours plus vite

**Nathalie Revol**

INRIA, LIP, ENS-Lyon

Nathalie.Revol@ens-lyon.fr

Die, 19 octobre 2004

# Algorithme. . .

suite d'instructions à effectuer pour accomplir une tâche

Pas d'implicite ni d'imprécision.
Syntaxe rigide quand on s'adresse à un ordinateur.

# Algorithme. . .
# recette du taboulé

# Algorithme. . .
# recette du taboulé (pour un humain)

1. Dans une terrine ajouter dans l'ordre en mélangeant à chaque fois : La semoule, les tomates coupées en dés avec leur jus, les oignons hachés, 3 cuillères de menthe hachée, 3 cuillères de persil haché, le jus des citrons, les olives et les raisins puis ajouter le reste des ingrédients : huile, sel et poivre.

2. Vérifier l'assaisonnement plusieurs fois. Laisser macérer 3 ou 4 heures (voire faire la veille).

3. En décoration quelques feuilles de menthe en bouquet une tomate en quartier et un petit oignon en quartier.

# Recette du taboulé : traduction en langage algorithmique

1. Dans une terrine ajouter dans l'ordre en mélangeant à chaque fois : La semoule, les tomates coupées en dés avec leur jus, les oignons hachés, 3 cuillères de menthe hachée, 3 cuillères de persil haché, le jus des citrons, les olives et les raisins puis ajouter le reste des ingrédients huile , sel et poivre.

   ajouter 250g de semoule dans la terrine
   mélanger
   couper 500g de tomates en dés
   ajouter les 500g de tomates coupées en dés avec leur jus dans la terrine
   mélanger
   hacher 250g d'oignons
   ajouter les 250g d'oignons hachés dans la terrine
   mélanger

hacher 3 cuillères à soupe de menthe
ajouter les 3 cuillères à soupe de menthe hachée dans la terrine
mélanger
hacher 3 cuillères à soupe de persil
ajouter 3 cuillères à soupe de persil haché dans la terrine
mélanger
presser 3 citrons
ajouter le jus des 3 citrons dans la terrine
mélanger
ajouter 150g d'olives dans la terrine
mélanger
ajouter les 50g de raisins dans la terrine
mélanger
ajouter 8 cuillerées à soupe d'huile d'olive
ajouter 1/2 cuillerée à café de sel et 1/2 cuillerée à café de poivre
mélanger

2. Vérifier l'assaisonnement plusieurs fois. Laisser macérer 3 ou 4 heures (voire faire la veille).
   intraduisible !!!

3. En décoration quelques feuilles de menthe en bouquet une tomate en quartier et un petit oignon en quartier.
   poser sur la terrine quelques feuilles de menthe
   découper une tomate en quartier
   poser sur la terrine les quartiers de tomate
   découper un oignon (de 100g) en quartier
   poser sur la terrine les quartiers d'oignon

**Et tout ceci en supposant que "peser", "hacher", "découper" etc ont déjà été précisément définis algorithmiquement !**

# Plan de l'exposé

- **Compter avec des entiers**
  - opérations arithmétiques (école primaire)
  - notation redondante
  - calculer avec des rationnels (collège)
  - calcul modulaire
- **Problèmes de graphes**
  - qu'est-ce qu'un graphe
  - chemin eulérien
  - problèmes difficiles
- **Hiérarchie des problèmes**
  - la classe P
  - la classe NP
  - hiérarchie

# Plan de l'exposé

- **Compter avec des entiers**
  - opérations arithmétiques (école primaire)
  - notation redondante
  - calculer avec des rationnels (collège)
  - calcul modulaire
- **Problèmes de graphes**
  - qu'est-ce qu'un graphe
  - chemin eulérien
  - problèmes difficiles
- **Hiérarchie des problèmes**
  - la classe P
  - la classe NP
  - hiérarchie

# Compter avec des entiers

**Pour des humains ayant 10 doigts :**

représentation des nombres entiers :

$$13 = 1 \times 10 + 3 \times 1 = 1 \times 10^1 + 3 \times 10^0$$

**Pour des ordinateurs ayant 2 poings :**

représentation des nombres entiers :

$$13_{10} = 1 \times 8 + 1 \times 4 + 0 \times 2 + 1 \times 1 = 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 1101_2$$

# Compter avec des entiers : additionner

À la main :

$$\begin{array}{cccccc} & & & {\scriptstyle 1} & {\scriptstyle 1} & \\ & & 1 & 2 & 3 & 4 \\ + & & & 5 & 6 & 7 \\ \hline & & 1 & 8 & 0 & 1 \end{array}$$

# Compter avec des entiers : additionner

Complexité de l'addition de 2 nombres à $n$ chiffres :

$$\mathcal{O}(n)$$

opérations sur des chiffres.

**Hypothèse :**
additionner deux chiffres quelconques prend toujours le même temps
ce qui n'est pas le cas du cerveau humain : $1 + 2$ est plus rapide à effectuer que $8 + 9$.

# Compter avec des entiers : problème sur ordinateur

$$\begin{array}{cccccc} & & {\scriptstyle 1} & {\scriptstyle 1} & & \\ & 1 & 2 & 3 & 4 & \\ + & & 5 & 6 & 7 & \\ \hline & 1 & 8 & 0 & 1 & \end{array}$$

**Problème :** la retenue !

Pourquoi ne pas commencer par la gauche, pour avoir tout de suite l'ordre de grandeur du résultat ?

Pourquoi ne pas additionner chacun deux chiffres simultanément ?

**Solution :** changer de système de numération !

# Compter avec des entiers
# Systèmes redondants de numération (Avizienis, 1961)

**Principe :** un même nombre peut avoir plusieurs représentations.
Avantage pour l'addition : on choisit toujours un chiffre qui permettra d'absorber la retenue (**absorber** au sens d'**empêcher de se propager**).

**Comment :** ne pas se limiter aux chiffres compris entre 0 et 9 :

- autoriser les chiffres supérieurs à 10 (par exemple de 0 à 19),
- autoriser les chiffres négatifs.

**Exemple :** en base 10 avec les chiffres de -6 à 6
1546 s'écrit $1546$, $2(-5)46$ noté $2\bar{5}46$, $2(-4)(-6)6$ noté $2\bar{4}\bar{6}6$, $2\bar{4}\bar{5}\bar{4}$, $16\bar{6}6$, $16\bar{5}\bar{4}$. . .

# Compter avec des entiers
# Systèmes redondants de numération (Avizienis, 1961)

$$\begin{array}{rcccc} & 1 & 2 & 3 & 4 \\ + & & 5 & 6 & 7 \\ \hline & 1 & 7 & 9 & 11 \end{array}$$

soit $1790 + 11 = 1801$.

$$\begin{array}{rcccc} & 1 & 2 & 3 & 4 \\ + & & 5 & 6 & 7 \\ \hline & 2 & \bar{2} & 0 & 1 \end{array}$$

$2\bar{2}01 = 2000 - 200 + 00 + 1 = 1801$.

# Compter avec des entiers : multiplier

À la main :

$$
\begin{array}{cccccccc}
 & & & & 9 & 8 & 7 & 6 \\
\times & & & 1 & 2 & 3 & 4 & \\
\hline
 & & & & & & & \\
 & & & 3 & 9 & 5 & 0 & 4 \\
 & & 2 & 9 & 6 & 2 & 8 & \\
 & 1 & 9 & 7 & 5 & 2 & & \\
 & 9 & 8 & 7 & 6 & & & \\
\hline
1 & 2 & 1 & 8 & 6 & 9 & 8 & 4
\end{array}
$$

# Compter avec des entiers : multiplier

Sur ordinateur :

$$
\begin{array}{rrrrrrrr}
 & & & & 9 & 8 & 7 & 6 \\
\times & & & 1 & 2 & 3 & 4 & \\
\hline
 & & & & & & & \\
 & & & 3 & 9 & 5 & 0 & 4 \\
 & & + & 2 & 9 & 6 & 2 & 8 \\
\hline
 & & & 3 & 3 & 5 & 7 & 8 & 4 \\
 & + & 1 & 9 & 7 & 5 & 2 & & \\
\hline
 & & 2 & 3 & 1 & 0 & 9 & 8 & 4 \\
 & + & 9 & 8 & 7 & 6 & & & \\
\hline
 & 1 & 2 & 1 & 8 & 6 & 9 & 8 & 4
\end{array}
$$

# Compter avec des entiers : multiplier

## Multiplication des paysans russes

## ou comment éviter d'apprendre ses tables

$$57 * 86 = 4902$$

| | |
|---|---|
| 86 | 57 |
| 172 | 28 |

# Compter avec des entiers : multiplier

## Multiplication des paysans russes

## ou comment éviter d'apprendre ses tables

$$57 * 86 = 4902$$

| | |
|---|---|
| 86 | 57 |
| 172 | 28 |
| 344 | 14 |

# Compter avec des entiers : multiplier

## Multiplication des paysans russes

## ou comment éviter d'apprendre ses tables

$$57 * 86 = 4902$$

| | |
|---|---|
| 86 | 57 |
| 172 | 28 |
| 344 | 14 |
| 688 | 7 |

# Compter avec des entiers : multiplier

## Multiplication des paysans russes

## ou comment éviter d'apprendre ses tables

$$57 * 86 = 4902$$

| | |
|---|---|
| 86 | 57 |
| 172 | 28 |
| 344 | 14 |
| 688 | 7 |
| 1376 | 3 |

# Compter avec des entiers : multiplier

## Multiplication des paysans russes

## ou comment éviter d'apprendre ses tables

$$57 * 86 = 4902$$

| | |
|---|---|
| 86 | 57 |
| 172 | 28 |
| 344 | 14 |
| 688 | 7 |
| 1376 | 3 |
| 2752 | 1 |

# Compter avec des entiers : multiplier

## Multiplication des paysans russes

## ou comment éviter d'apprendre ses tables

$$57 * 86 = 4902$$

| | |
|---|---|
| 86 | 57 |
| ~~172~~ | ~~28~~ |
| ~~344~~ | ~~14~~ |
| 688 | 7 |
| 1376 | 3 |
| 2752 | 1 |

# Compter avec des entiers : multiplier

## Multiplication des paysans russes

## ou comment éviter d'apprendre ses tables

$$57 * 86 = 4902$$

| | |
|---|---|
| 86 | 57 |
| ~~172~~ | ~~28~~ |
| ~~344~~ | ~~14~~ |
| 688 | 7 |
| 1376 | 3 |
| 2752 | 1 |
| 4902 | |

# Multiplication : on peut aller plus vite. . .

Si $A$ et $B$ s'écrivent chacun avec $2n$ chiffres en base $\beta$,

$$A = a_H \beta^n + a_L, \qquad B = b_H \beta^n + b_L$$

$$A \times B = a_H b_H \beta^{2n} + (a_H b_L + a_L b_H)\beta^n + a_L b_L$$

soit 4 multiplications de nombres 2 fois plus court. . .

## Multiplication de Karatsuba

On peut aussi écrire le produit comme

$$A \times B = a_H b_H \beta^{2n} + [(a_H + a_L)(b_H + b_L) - a_H b_H - a_L b_L]\beta^n + a_L b_L$$

avec 3 multiplications de nombres 2 fois plus court. . .
on multiplie 2 nombres de $n$ chiffres en $\mathcal{O}(n^{\log_2 3})$.

# Multiplication : on peut aller plus vite. . .

Si $A = 1234$ et $B = 5678$ ($4 = 2 \times 2$ chiffres en base 10),
$A \times B = 7006652$

$$A = 12 \times 10^2 + 34, \qquad B = 56 \times 10^2 + 78$$

## Multiplication de Karatsuba

$$\begin{aligned} A \times B &= 12 \times 56 \times 10^4 + [(12 + 34) \times (56 + 78) - 12 \times 56 - 34 \times 78] \times \\ &= 672 \times 10^4 + [46 \times 134 - 672 - 2652] \times 10^2 + 2652 \\ &= 672 \times 10^4 + [6164 - 672 - 2652] \times 10^2 + 2652 \\ &= 6720000 + 284000 + 2652 \\ &= 70066652 \end{aligned}$$

avec 3 multiplications de nombres 2 fois plus court. . .

# Multiplication : on peut aller encore plus vite. . .

## Multiplication rapide

Multiplication basée sur la transformée de Fourier rapide (FFT) (ou plutôt discrète : DFT) : la complexité asymptotique de la multiplication est

$$\mathcal{O}(n \log n \log \log n)$$

mais elle est difficile à implanter et ne bat d'autres méthodes plus simples qu'à partir d'environ 57 000 chiffres décimaux.

# Exponentiation

$$3^{13} = 1\,594\,323$$

| | |
|---|---|
| 3 | 13 |
| 9 | 6 |
| 81 | 3 |
| 6561 | 1 |

puis multiplication $3 * 81 * 6561 = 1\,594\,323$.

Autre façon de l'écrire : $13 = 8 + 4 + 1 = 1101_2$
$3^{13} = 3^8 \times 3^4 \times 3^1$ avec $3^4 = \left(3^2\right)^2$ et $3^8 = \left(3^4\right)^2$.
On procède par élévations au carré successives et multiplications finales : il y en a $\mathcal{O}(\log_2 n)$ si $n$ est l'exposant.

# Compter avec des entiers : diviser

## Division

| 990 | 252 |
|---|---|
| −756 | 3 |
| 234 | |

| 252 | 234 |
|---|---|
| −234 | 1 |
| 18 | |

| $\overset{\frown}{234}$ | 18 |
|---|---|
| −18 | 13 |
| 54 | |
| −54 | |
| 0 | |

Complexité de la division de 2 nombres à $n$ chiffres :

$$\mathcal{O}(M(n))$$

où $M(n)$ est la complexité de la multiplication.

# Compter avec des entiers : diviser le bug du Pentium en 1994

**Algorithme utilisé :** celui qu'on apprend à l'école.

**Bug du Pentium :** T. Nicely (Virginie, USA), en octobre 1004, a constaté que son ordinateur donnait un résultat erroné lors du calcul de $1.0/824633702441$ : erreur à partir du 12e chiffre.
T. Coe a découvert le pire cas : $41195835.0/3145727.0$ donnait $1.333739068902$ au lieu de $1.333820449136$ (faux à partir du 5e chiffre).

**Explication :** pour aller plus vite, utilisation de notation redondante :
utilisation de chiffres supérieurs à la base pour les restes partiels
utilisation de chiffres négatifs pour le quotient
et erreur en mélangeant tout ça !

# Calculer à l'école primaire : les entiers

## pgcd

Rappel : pgcd $(a, b)$ = pgcd $(b, r)$ avec $a = bq + r$ si $r \neq 0$.

Algorithme d'Euclide :

$$\text{pgcd}(990,\ 252) = \text{pgcd}(252,\ 234) = \text{pgcd}(234,\ 18) = 18.$$

Complexité du pgcd de 2 nombres à $n$ chiffres : $\mathcal{O}(M(n) \log n)$.

# Calculer à l'école primaire : les entiers

**Savoir si un nombre est premier :** on a longtemps cru que c'était difficile, depuis l'été 2002, on sait grâce à Agrawal, Kayal et Saxena que c'est "facile" : en $\mathcal{O}(n^9)$.

**Factoriser un nombre en ses facteurs premiers :**
$990 = 2 \times 3 \times 3 \times 5 \times 11$ et $252 = 2 \times 2 \times 3 \times 3 \times 7$.

Complexité : difficile ! $\Rightarrow$ certains algorithmes de cryptographie à clé publique, dont RSA.

# Calcul modulaire

**Calculer l'heure ou le jour :** calcul modulo 60, 24, 7 ou 12.

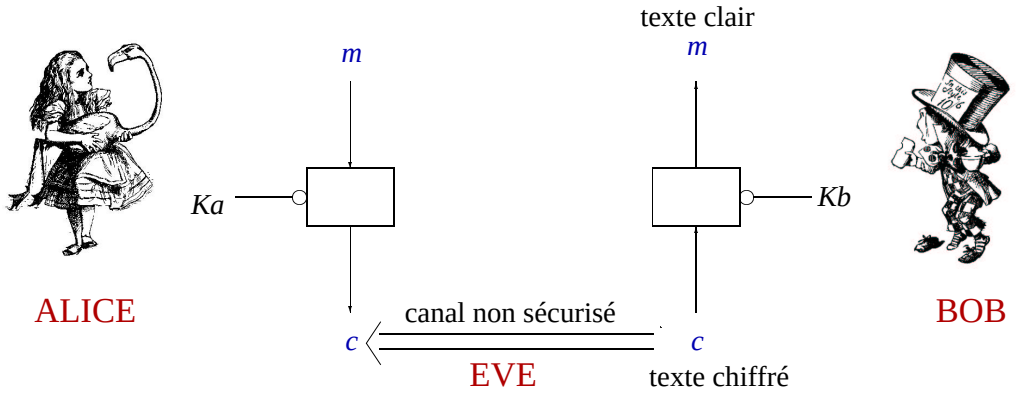
S'il est maintenant 8h45, quelle heure sera-t-il dans 35mn ? 9h20 :
calcul modulo 60 : $(45 + 35) \mod 60 = 80 \mod 60 = 20$
et comme $(45 + 35)/60 = 80/60 = 1$, vous avez rajouté 1h.

Et dans 9h, quelle heure sera-t-il ? 5h45 du matin :
$(20 + 9) \mod 24 = 29 \mod 24 = 5$
ou alors $(8 + 9) \mod 12 = 17 \mod 12 = 5$.

Aujourd'hui nous sommes mardi 19 octobre. Quel jour de la semaine tombera le 29 octobre ?
$(29 - 19) \mod 7 = 10 \mod 7 = 3$
mercredi-un, jeudi-deux, vendredi-trois : ce sera un vendredi !

# Algorithme RSA
# Alice, Bob et la méchante Eve. . .

*Cryptographie à clé publique*




INRIA

4

# Algorithme RSA
# Alice, Bob et la méchante Eve. . .

ALICE

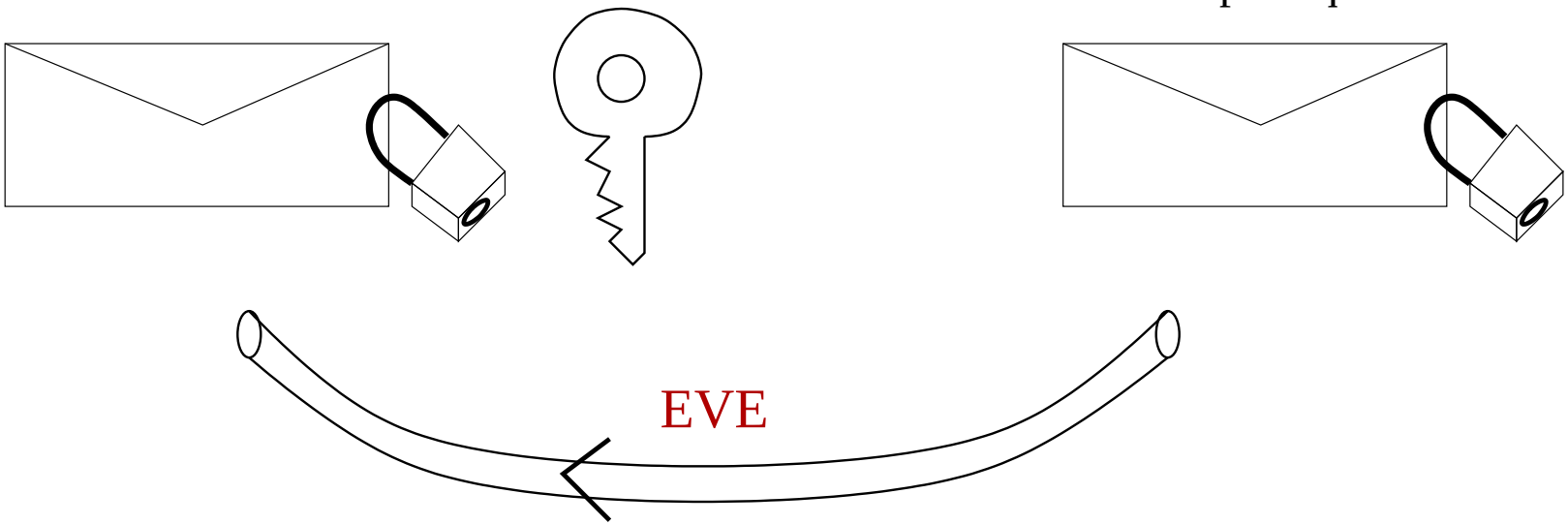
BOB

décrypte le message

avec sa clé privée

encrypte le message

avec la clé publique d'Alice



EVE

communication non sécurisée

# Algorithme RSA
# Alice, Bob et la méchante Eve. . .

## Alice

1. choisit 2 grands nombres premiers $p$ et $q$ : facile
2. calcule $N = p \times q$ ($N$ doit avoir au moins 150 chiffres)
3. choisit $e \in \{2, \cdots, \varphi(N) - 2\}$ premier avec $\varphi(N)$
   où $\varphi$ est la fonction d'Euler : $\varphi(N) = (p-1) \times (q-1)$
4. calcule $d$ tel que $ed \equiv 1 \mod \varphi(N)$
   Bézout : $\exists (d, v)/ed + v.\varphi(N) = 1$.
5. publie $(N, e)$ sa clé publique et cache $(N, d)$ sa clé secrète.

## Euler

$\forall x \in \{0 \cdots N-1\}$, si $x \wedge N = 1$ alors $x^{\varphi(N)} \equiv 1 \mod N$,
càd $x^{\varphi(N)+1} \equiv x \mod N$.

# Alice, Bob et la méchante Eve. . .

## Alice

publie $(N, e)$ sa clé publique et cache $(N, d)$ sa clé secrète.
(Bézout : $\exists (d, v)/ed + v.\varphi(N) = 1$.)

## Euler

$\forall x \in \{0 \cdots N-1\}$, si $x \wedge N = 1$ alors $x^{\varphi(N)+1} \equiv x \mod N$.

## Bob

1. écrit son message sous la forme d'un entier $x \in \{0 \cdots N-1\}$
2. calcule $y = x^e \mod N$ : facile
3. envoie $y$

**Alice** calcule $y^d$ :

$$x \equiv x^{\varphi(N)+1} \equiv x^{-v.\varphi(N)+1} \equiv x^{ed} \equiv y^d \mod N$$

# Les rationnels

$+, -, \times, /$ : on sait faire
(beaucoup de calculs de pgcd).

Exemples de logiciels : Maple, Mathematica, certaines calculatrices (TI).

**Avantage : calcul exact.**

**Inconvénients :**
Nombres grossissent.
Complexité : dépend de la taille des nombres.
$\sqrt{}, \exp, \sin$ : on ne sait pas faire. . .

# Plan de l'exposé

- **Compter avec des entiers**
  - opérations arithmétiques (école primaire)
  - notation redondante
  - calculer avec des rationnels (collège)
  - calcul modulaire
- **Problèmes de graphes**
  - qu'est-ce qu'un graphe
  - chemin eulérien
  - problèmes difficiles
- **Hiérarchie des problèmes**
  - la classe P
  - la classe NP
  - hiérarchie

# Qu'est-ce qu'un graphe ?

C'est un ensemble de sommets reliés par des traits ou des flèches.

**Exemples :**

- carte routière : sommets = villes et traits = routes
- schéma d'orientation : sommets = niveaux d'étude et traits = accès
- etc

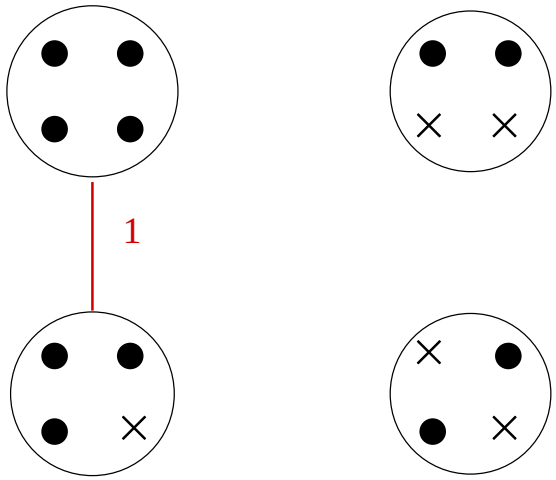
# Intermède

Quatre verres sont placés en carré sur un plateau. Une barmaid aveugle portant des gants de boxe essaie de les mettre tous dans le même sens, en retournant un ou deux verres. À chaque essai, un farceur lui dit sans jamais lui mentir s'il a réussi puis tourne à sa guise le plateau.

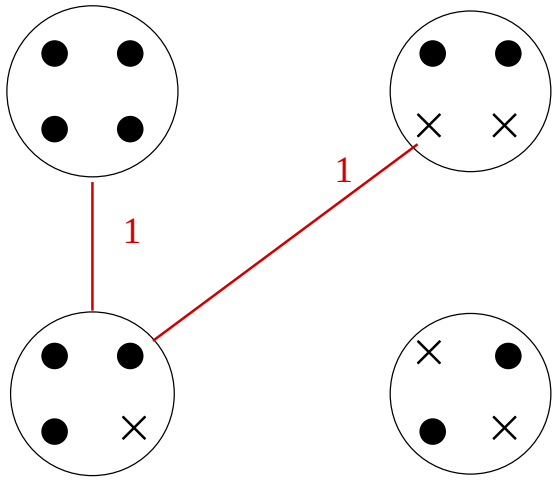
La barmaid va-t-elle réussir à mettre tous les verres dans le même sens ?

Quelle est la stratégie la plus rapide qui permette d'y arriver à coup sûr ?

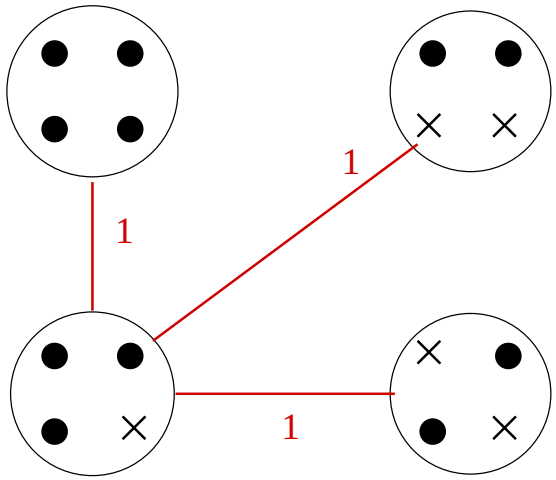
# La serveuse de bar aveugle avec des gants de boxe



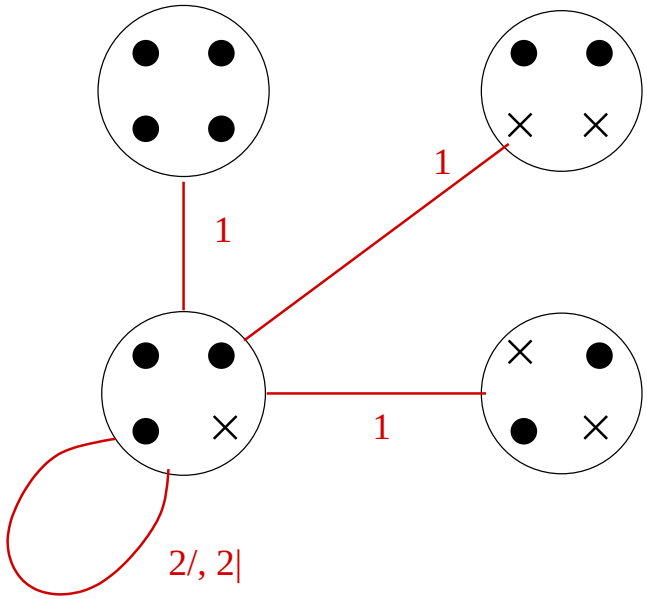
# La serveuse de bar aveugle avec des gants de boxe



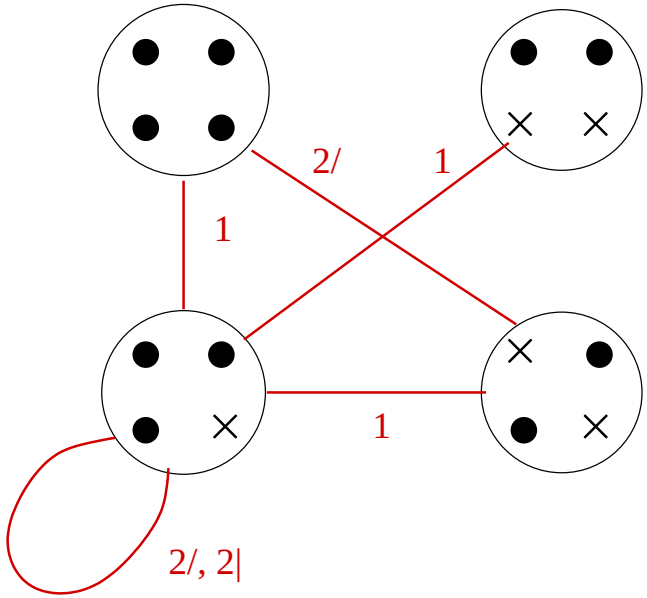
# La serveuse de bar aveugle avec des gants de boxe



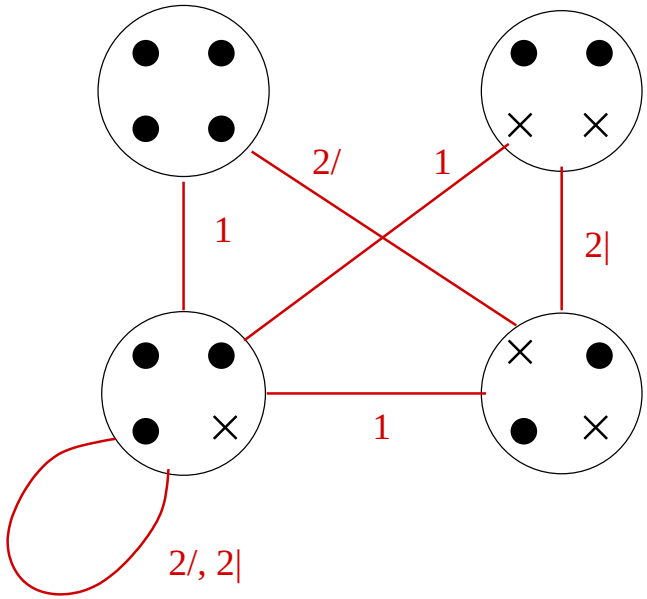
# La serveuse de bar aveugle avec des gants de boxe



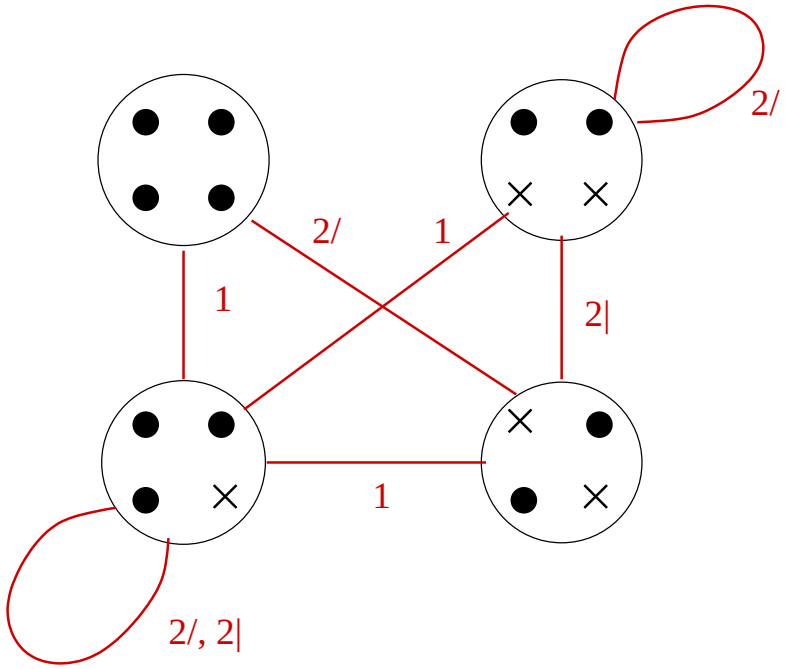
# La serveuse de bar aveugle avec des gants de boxe



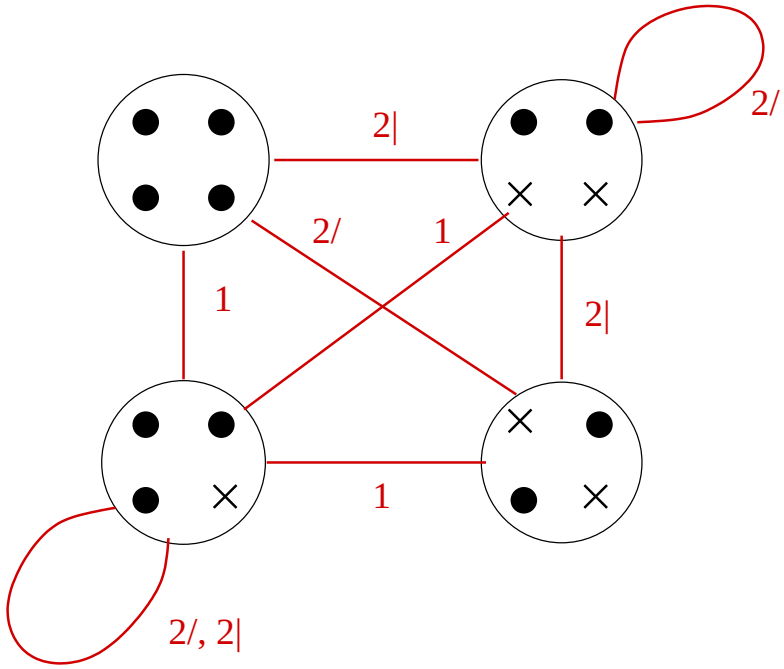
# La serveuse de bar aveugle avec des gants de boxe



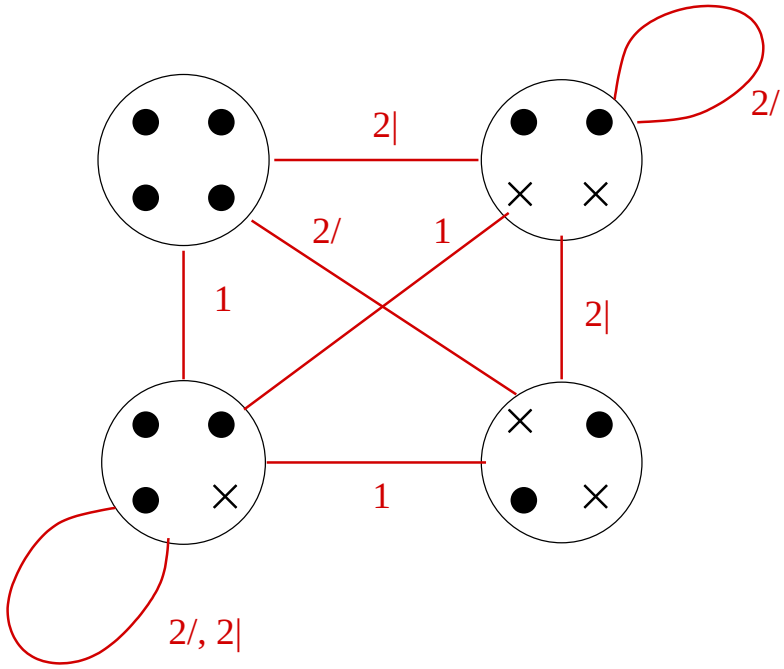
# La serveuse de bar aveugle avec des gants de boxe



# La serveuse de bar aveugle avec des gants de boxe



# La serveuse de bar aveugle avec des gants de boxe




Stratégie : $2/,\ 2|,\ 2/,\ 1,\ 2/,\ 2|,\ 2/.$

# Le loup, la chèvre et le chou

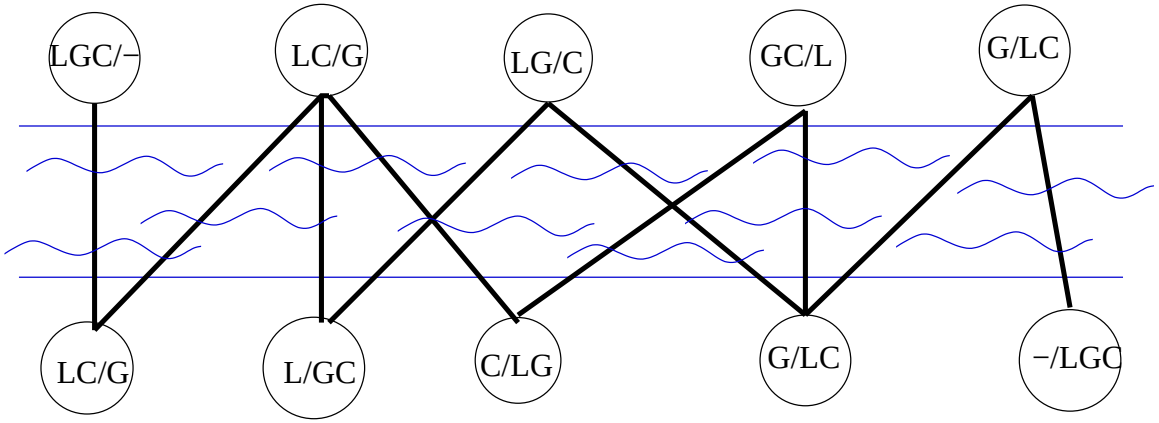
Sur le bord d'une rivière se trouvent un loup, une chèvre et un chou qui désirent traverser ; il y a un bateau si petit que le batelier seul et l'un d'eux peuvent y tenir. Il est question de les passer tous trois, de telle sorte que le loup ne mange pas la chèvre, ni la chèvre le chou, pendant l'absence du batelier.

Comment procéder ?

# Le loup, la chèvre et le chou

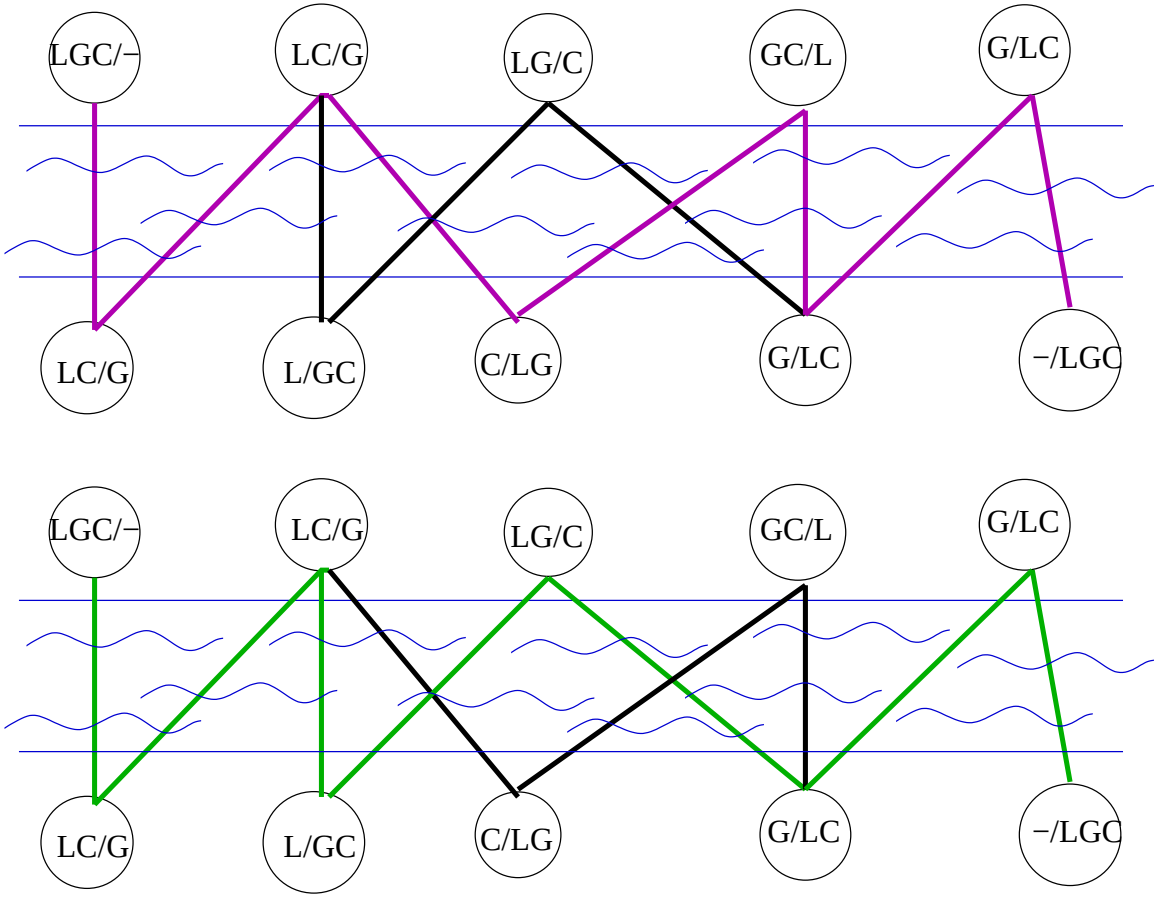
Sur le bord d'une rivière se trouvent un loup, une chèvre et un chou qui désirent traverser ; il y a un bateau si petit que le batelier seul et l'un d'eux peuvent y tenir. Il est question de les passer tous trois, de telle sorte que le loup ne mange pas la chèvre, ni la chèvre le chou, pendant l'absence du batelier.

Comment procéder ?



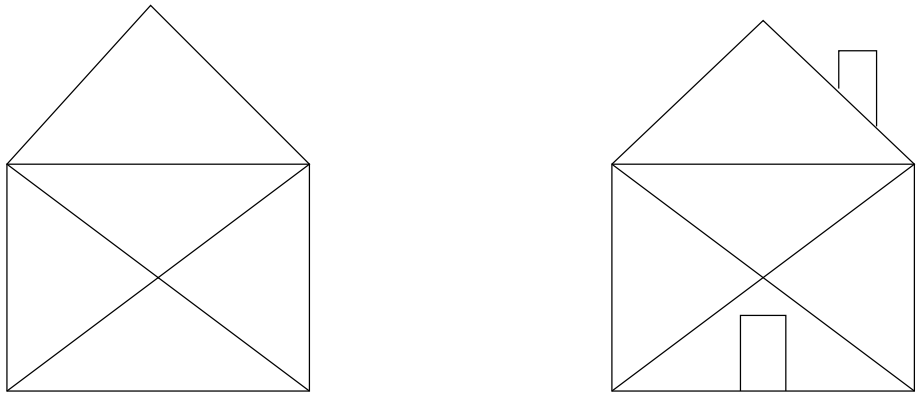
# Le loup, la chèvre et le chou

Deux solutions.



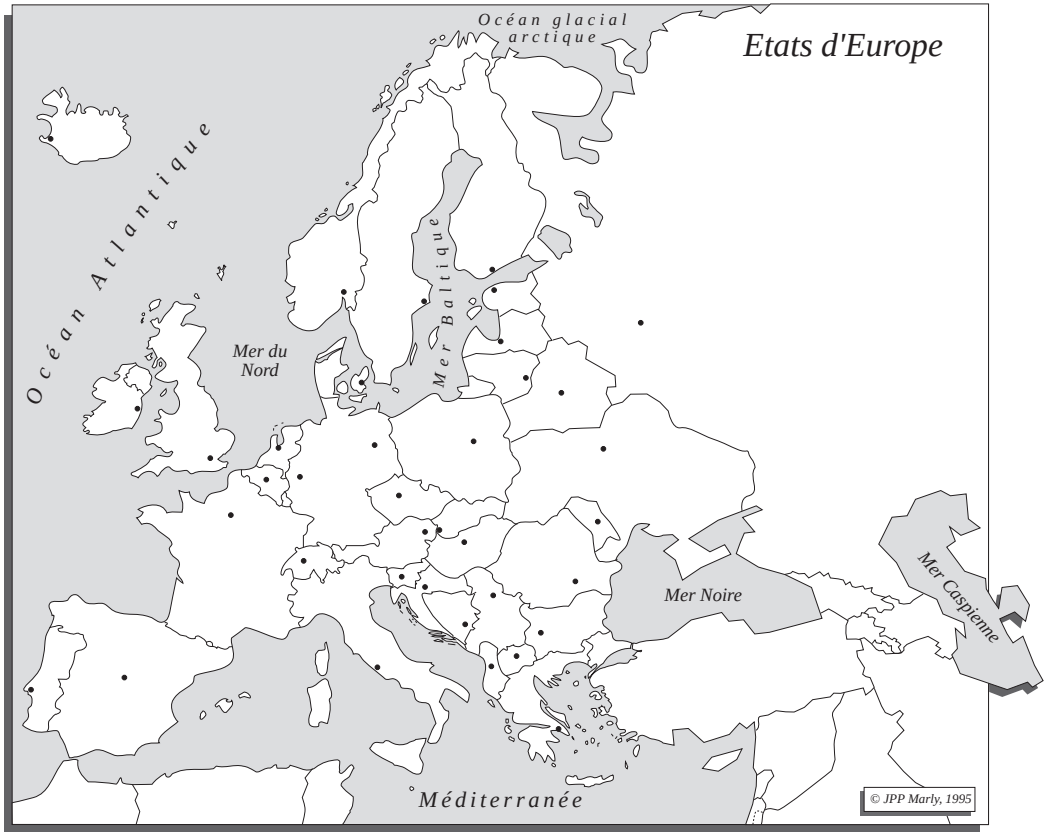
# Chemin eulérien

Peut-on dessiner le dessin de gauche sans lever le crayon et sans passer deux fois par la même arête ? Et celui de droite ?



Oui pour celui de gauche, non pour celui de droite : au plus 2 points peuvent avoir un degré impair.

# Graphes




carte : http://www.cfdp.ch/geofri/

# Complexité

## Reachability

Sur une carte routière avec $n$ villes, peut-on aller de Paris à Copenhague ? de Berlin à Dublin ?

Algorithme pour trouver la réponse en $\mathcal{O}(n^2)$.

## Problème du voyageur de commerce

Sur une carte de France métropolitaine avec $n$ villes, comment établir une tournée la plus courte possible qui passe une fois et une seule par chacune des $n$ villes ?

Aucun algorithme pour trouver la réponse en $\mathcal{O}(n^\alpha)$ (?).

# Plan de l'exposé

- **Compter avec des entiers**
  - opérations arithmétiques (école primaire)
  - notation redondante
  - calculer avec des rationnels (collège)
  - calcul modulaire
- **Problèmes de graphes**
  - qu'est-ce qu'un graphe
  - chemin eulérien
  - problèmes difficiles
- **Hiérarchie des problèmes**
  - la classe P
  - la classe NP
  - hiérarchie

# La classe P

## Définition

$P = \{$ problèmes pour lesquels il existe un algorithme de résolution en temps polynomial en la longueur des entrées $\}$.

## Exemples

addition : $\mathcal{O}(n)$ pour 2 entiers à $n$ chiffres

multiplication : $\mathcal{O}(n \log n \log \log n)$ pour 2 entiers à $n$ chiffres

division : $\mathcal{O}(M(n))$ pour 2 entiers à $n$ chiffres

pgcd : $\mathcal{O}(M(n) \log n)$ pour 2 entiers à $n$ chiffres

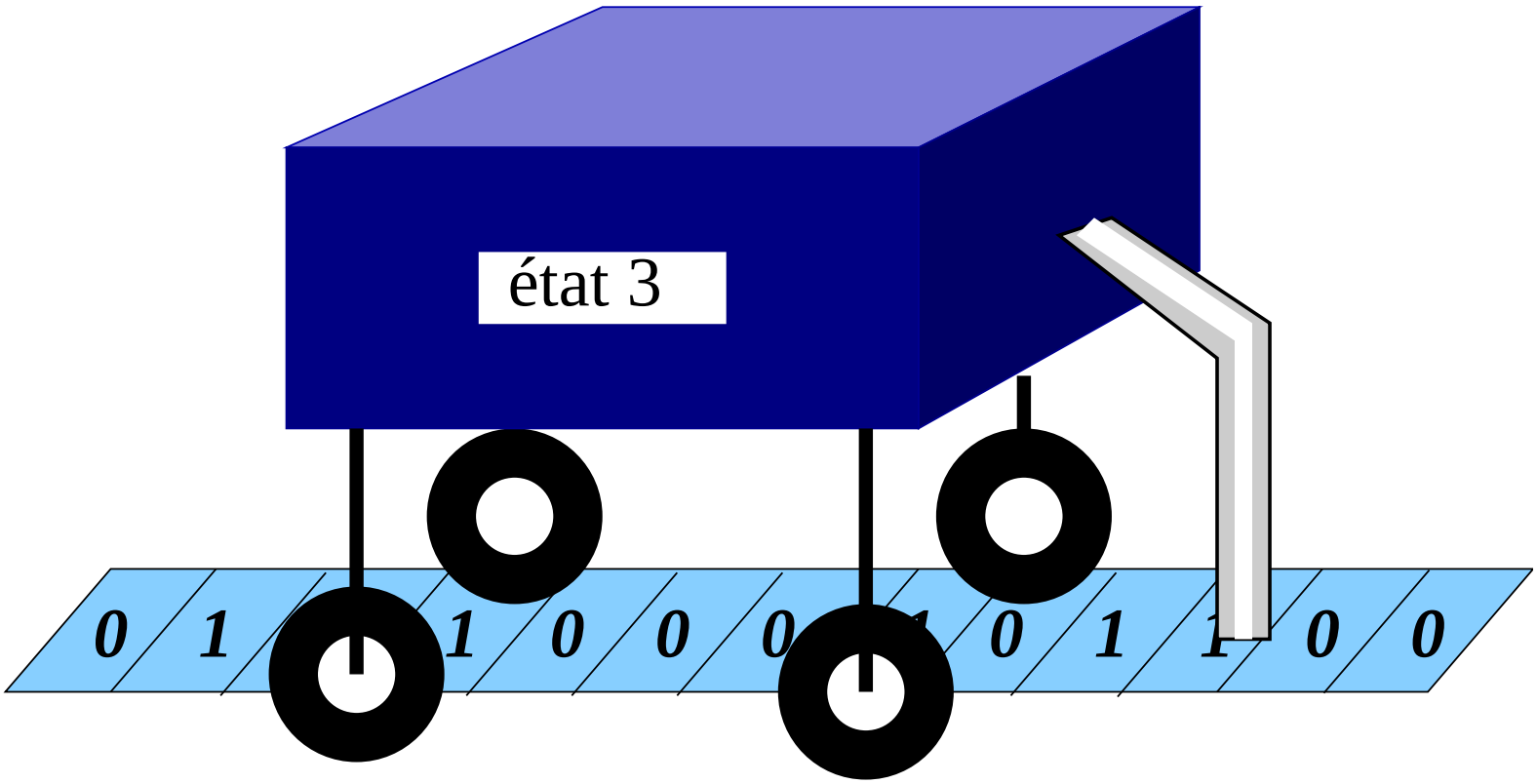
tester la primalité d'un entier à $n$ chiffres (août 2002)

## Sur quel ordinateur ?

sur un modèle théorique de machine appelé machine de Turing. . .

c'est la même sur tout ordinateur réel et avec tout langage de programmation !

# Machine de Turing



# Réduction d'un problème à un autre

## Ex : la division se ramène à la multiplication.

Pour calculer $A/B$ avec $A$ et $B$ deux entiers à $n$ chiffres : on calcule $X = \beta^n/B$ un inverse approché de $B$ (multiplié par $\beta^n$) puis $Q = (A \times X)/\beta^n$.
Si on veut le reste : $R - A - B \times Q$.

## Newton modifié pour calculer $X$ :

$$X_{2k} = X_k\beta^k + X_k\left(\beta^{2k} - (B_{2k} \times X_k)/\beta^k\right)/\beta^k$$

avec $X_k$ entier à $k$ chiffres, $B_k$ l'entier constitué à partir des $k$ chiffres " de gauche " de $B$.
Arrêt au bout de $\lceil \log_2 n \rceil$ étapes.

# Réduction d'un problème à un autre

## Complexité :

- à l'étape $k$ : deux multiplications de 2 nombres à $k$ chiffres et deux $+/-$
  complexité : $2M(k) + 2k$

- au total :

$$\sum_{i=1}^{\log n} 2M(2^i) \leq CM(n)$$

  puisque $M(n) \leq D.n^2$, on a $\sum_{i=1}^{\log n} M(i) \leq 2M(2n)$.

# La classe NP

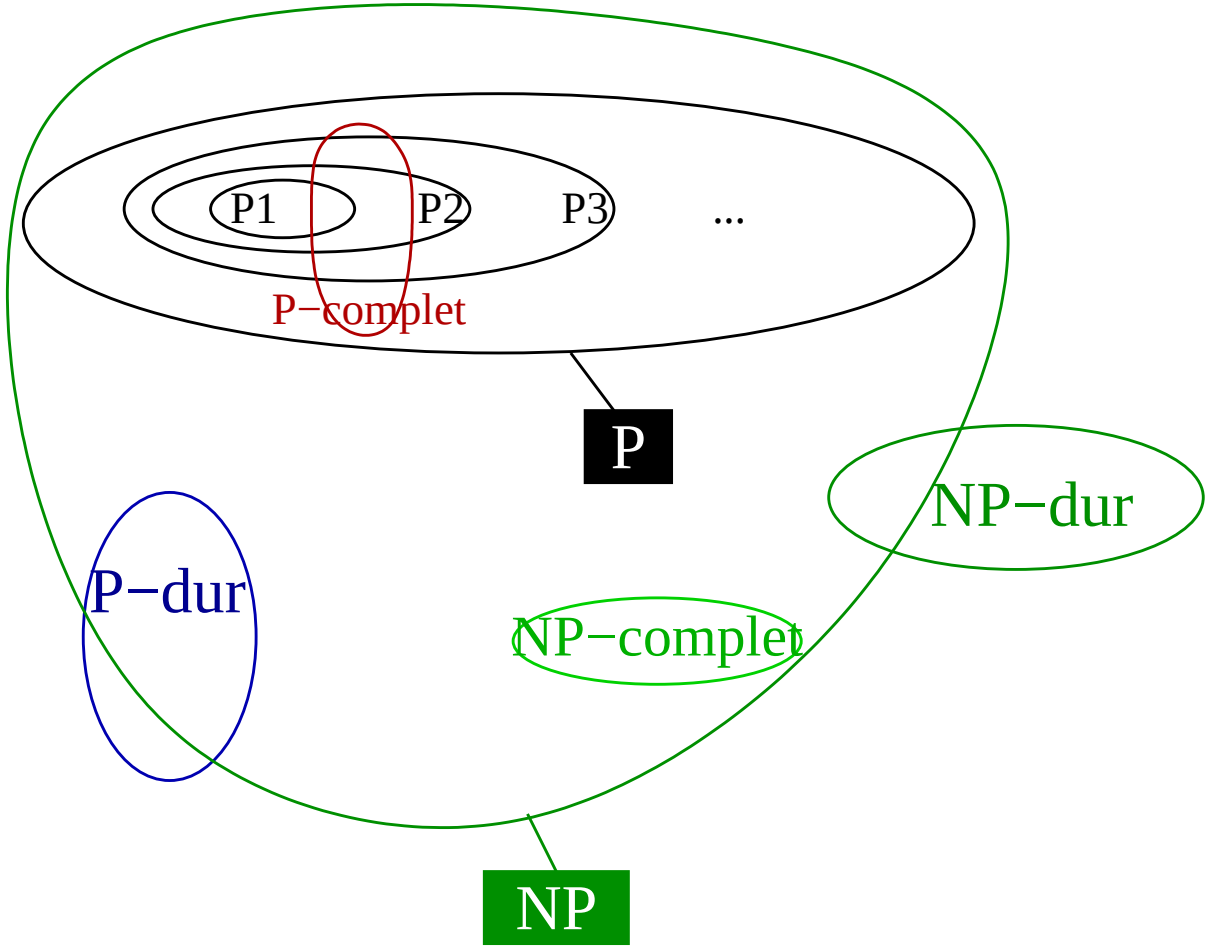
## Définition

$NP = \{$ problèmes pour lesquels il existe un algorithme de vérification de la solution en temps polynomial en la longueur des entrées $\}$.

## Exemples

factorisation d'entiers : il est facile de vérifier une décomposition
$990 = 2 \times 3 \times 3 \times 5 \times 11$ et $252 = 2 \times 2 \times 3 \times 3 \times 7$.
isomorphisme de graphes : si la bijection candidate est donnée, facile !

# Hiérarchies



# Conclusions

## On sait faire

les opérations arithmétiques exactes et flottantes

l'algèbre linéaire

. . .

## On ne sait pas faire

la factorisation d'entiers (?)

l'isomorphisme de graphes

. . .

## On essaie de classer les problèmes

les uns par rapport aux autres : P versus NP

P, P-complet, P-dur

NP, NP-complet, NP-dur

pour justifier pourquoi on ne sait pas faire ! (sauf si P = NP)

# Références

## Vulgarisation


- *Histoire d'algorithmes - Du caillou à la puce*, J.-L. Chabert et al., Belin 1994
- *Logique, informatique et paradoxes*, J.-P. Delahaye, Belin 1995
- *Histoire des codes secrets*, S. Singh, Le livre de poche 2001


## Modérément spécialisés


- *Qualité des calculs sur ordinateurs - Vers des arithmétiques plus fiables ?*, coordonné par M. Daumas et J.-M. Muller, Masson 1997
- *La cryptologie moderne*, A. Canteau et F. Lévy-dit-Véhel, Revue Armements, 2001, http://www-rocq.inria.fr/codes/Anne.Canteau/crypto-moderne.pdf
- *Arithmétique en précision arbitraire*, P. Zimmermann, Réseaux et systèmes répartis, vol. 13, no 4-5, 2001, http://www.inria.fr/rrrt/rr-4272.html
- *Arithmétique par intervalles*, N. Revol, Réseaux et systèmes répartis, vol. 13, no 4-5, 2001, http://www.inria.fr/rrrt/rr-4297.html

## Spécialisés

- *Modern computer algebra*, J. von zur Gathen and J. Gerhard, CUP, 1999
- *Arithmétique des ordinateurs*, J.-M. Muller, Masson 1989
- *Elementary functions*, J.-M. Muller, Birkhauser, 1997.
- *Interval methods for systems of equations*, A. Neumaier, CUP, 1990
- *Computational complexity*, C. Papadimitriou, Addison Wesley, 1994