L'informatique derrière les clics

« *Amphi pour tous* »

Guillaume Theyssier

Janvier 2009

Contenu de cet exposé

Contenu de cet exposé

Contenu de cet exposé

Quelques mots...

informatique : (1962) *science du traitement rationnel et automatique de l'information*

Quelques mots...

informatique : (1962) *science du traitement rationnel et automatique de l'information*

information : *action de donner ou de recevoir une forme*

Quelques mots...

informatique : (1962) *science du traitement rationnel et automatique de l'information*

information : *action de donner ou de recevoir une forme*

calcul : du latin CALCULUS "*petit caillou*"

Quelques mots...

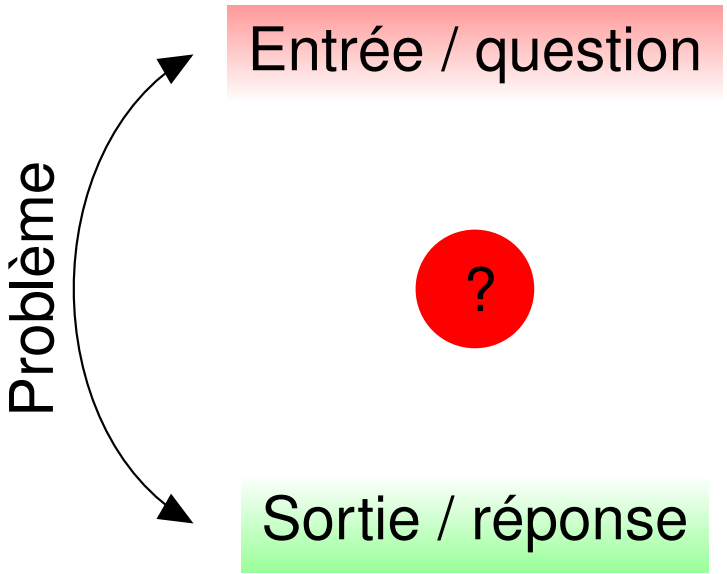
informatique : (1962) *science du traitement rationnel et automatique de l'information*

information : *action de donner ou de recevoir une forme*

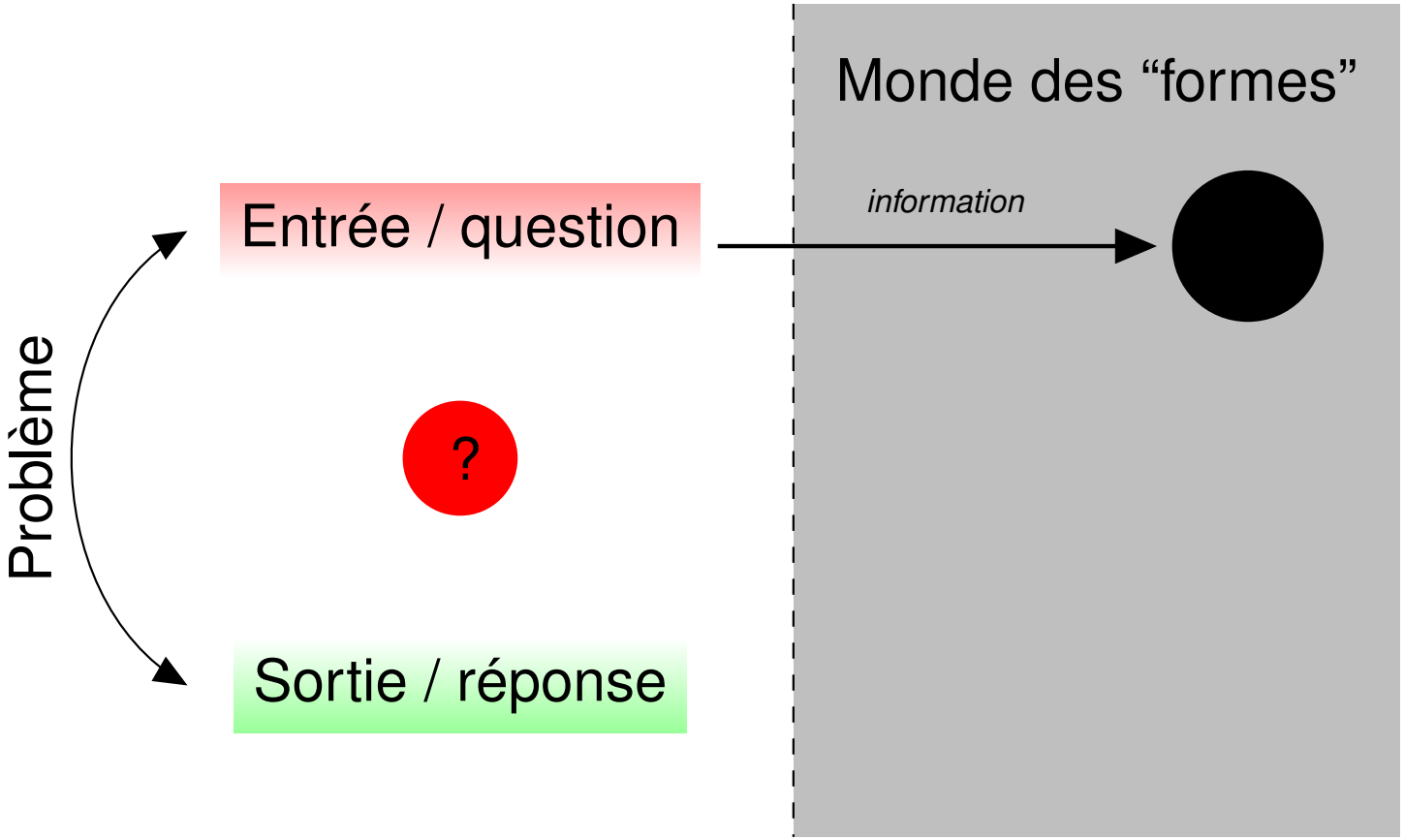
calcul : du latin CALCULUS "*petit caillou*"

algorithme : Al Khwarizmi, IX^{e} siècle

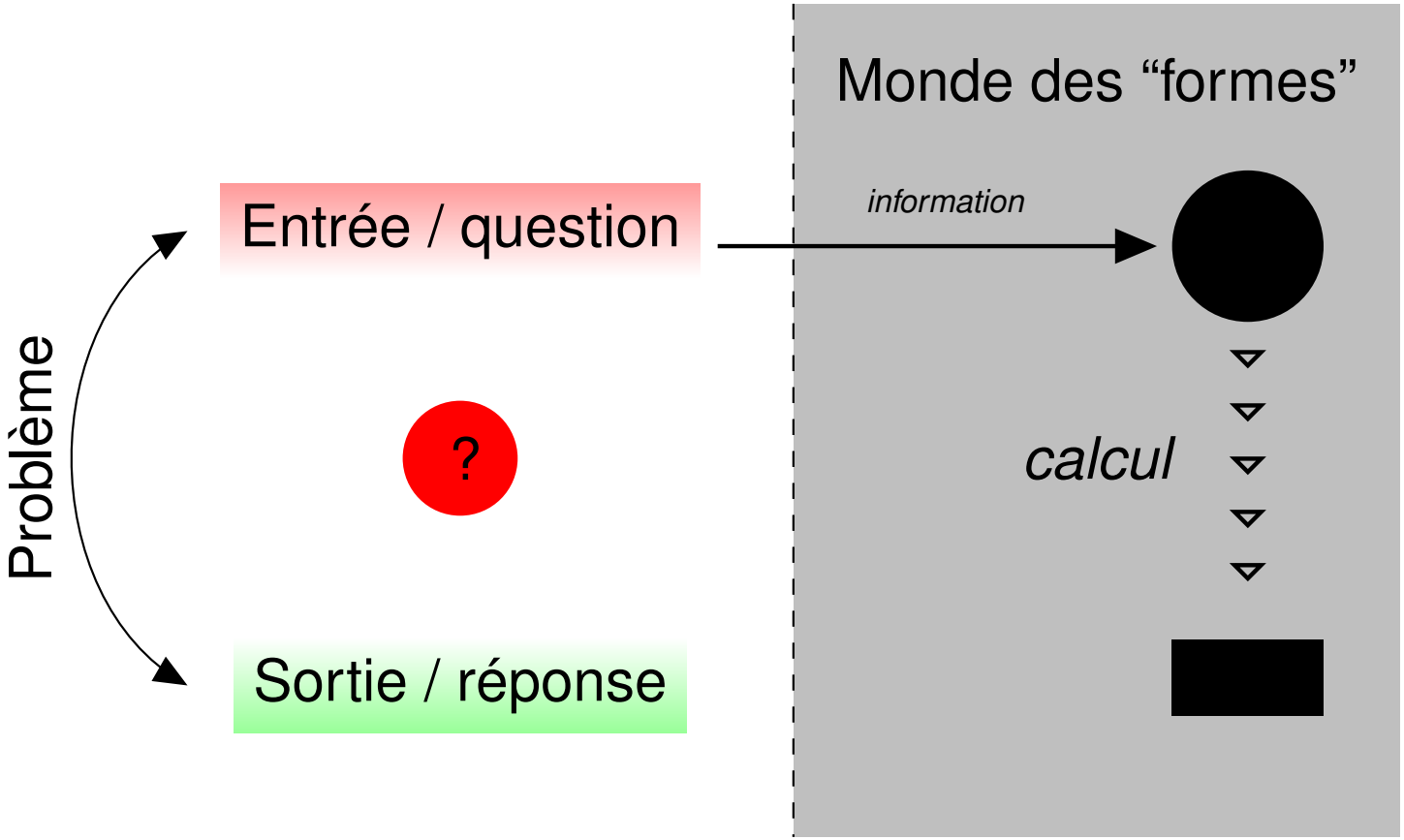
Informatique : de quoi s'agit-il ?



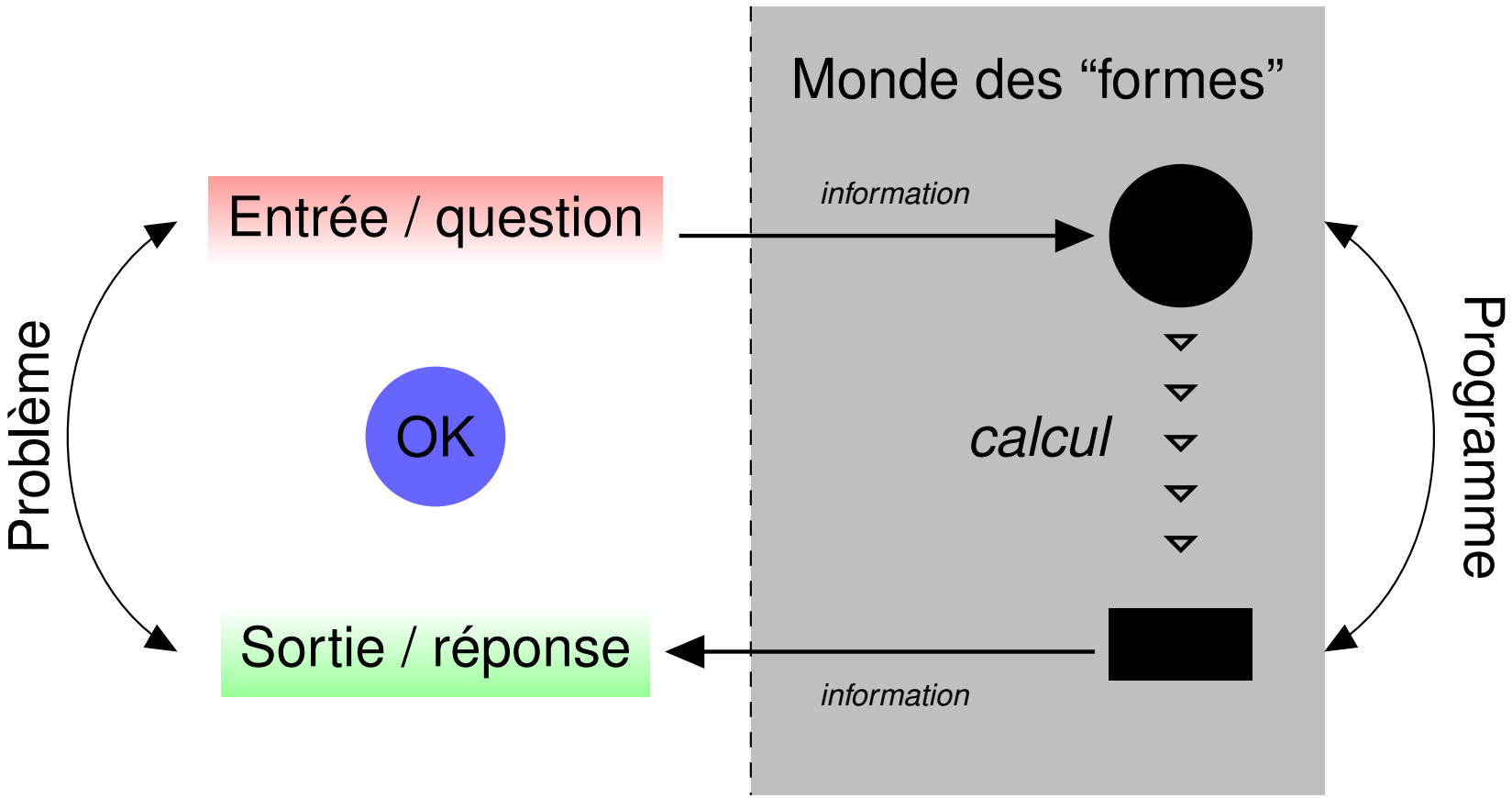
Informatique : de quoi s'agit-il ?



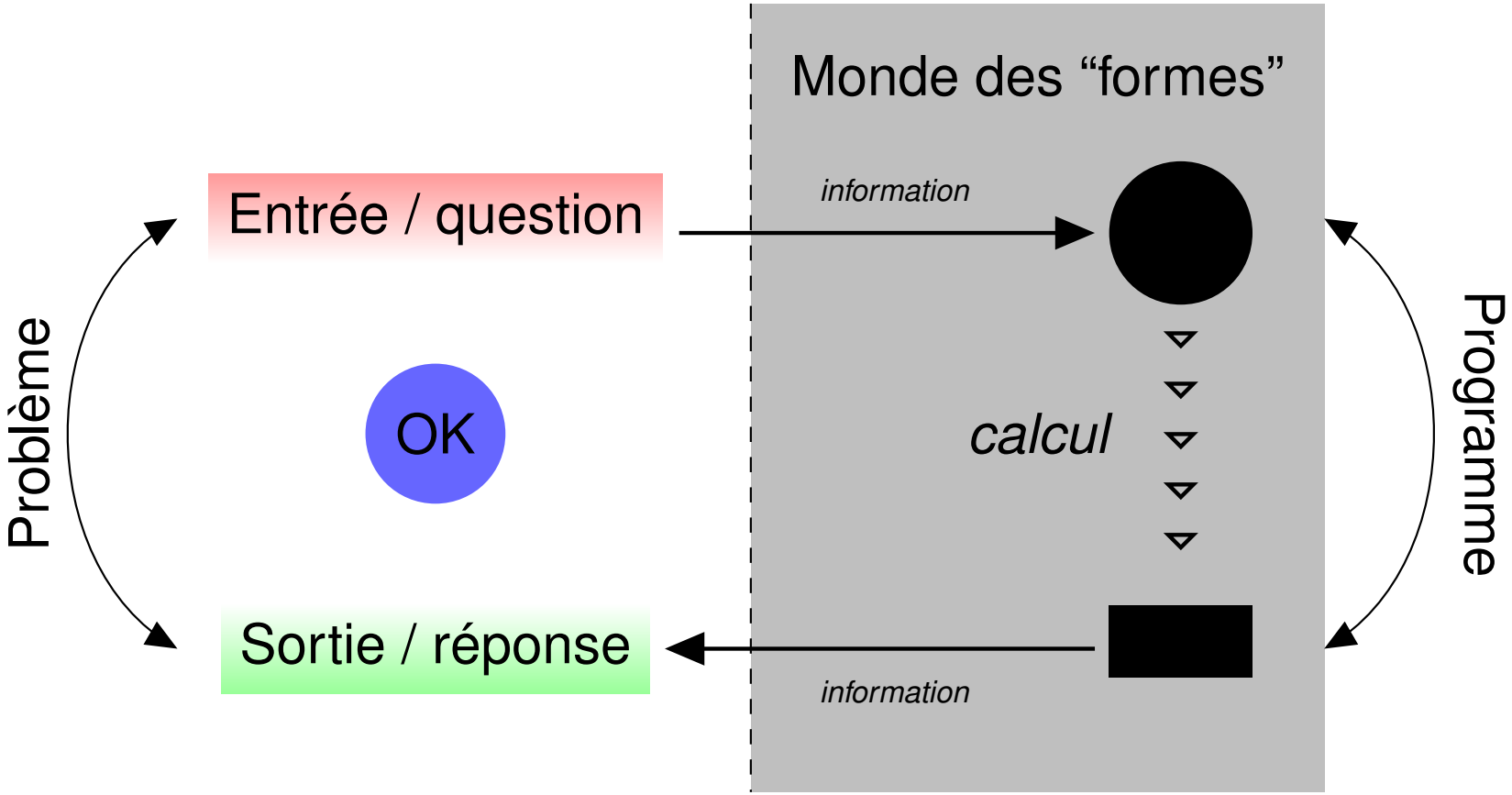
Informatique : de quoi s'agit-il ?



Informatique : de quoi s'agit-il ?



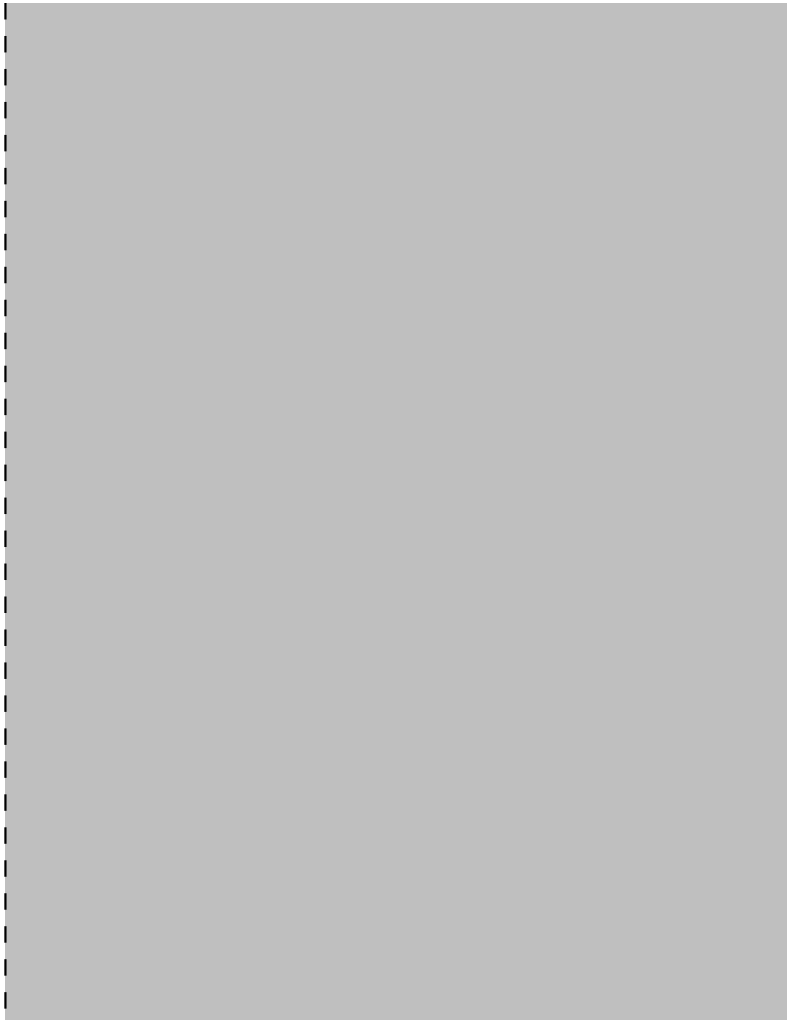
Informatique : de quoi s'agit-il ?



- un problème / des questions
- un programme / des calculs

Exemple

Monde des “formes”



Entrée : $a = 123$, $b = 45$

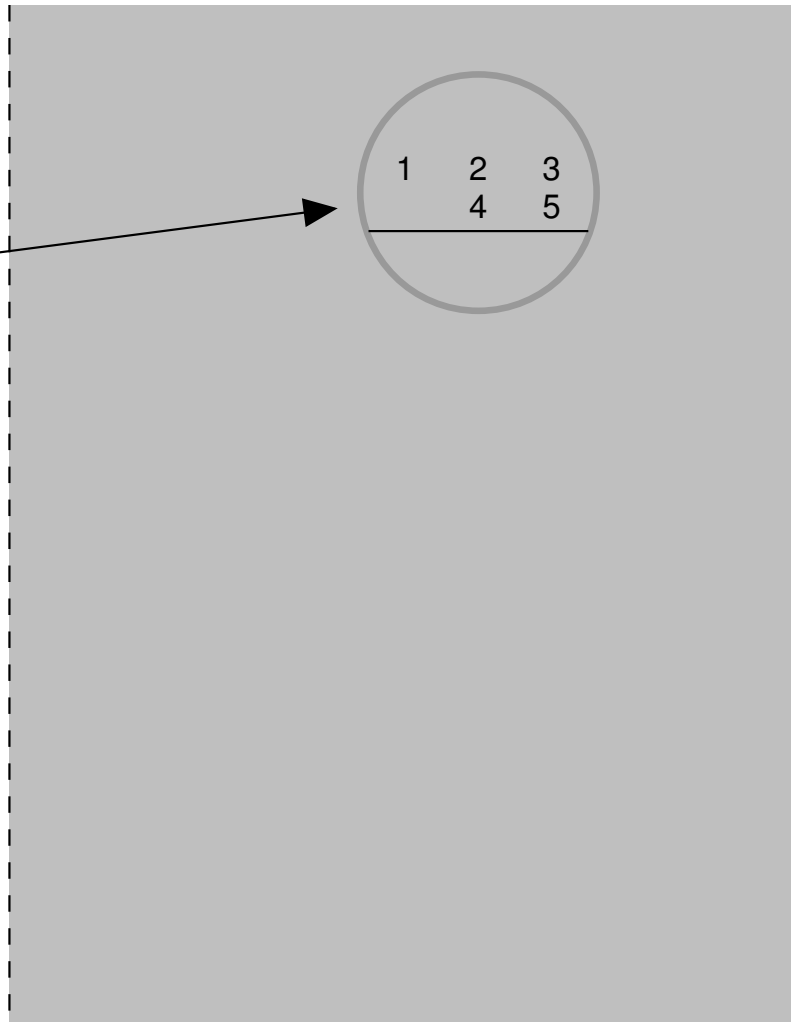
?

Sortie : $a \times b$

Exemple

Monde des “formes”

Entrée : $a = 123$, $b = 45$



?

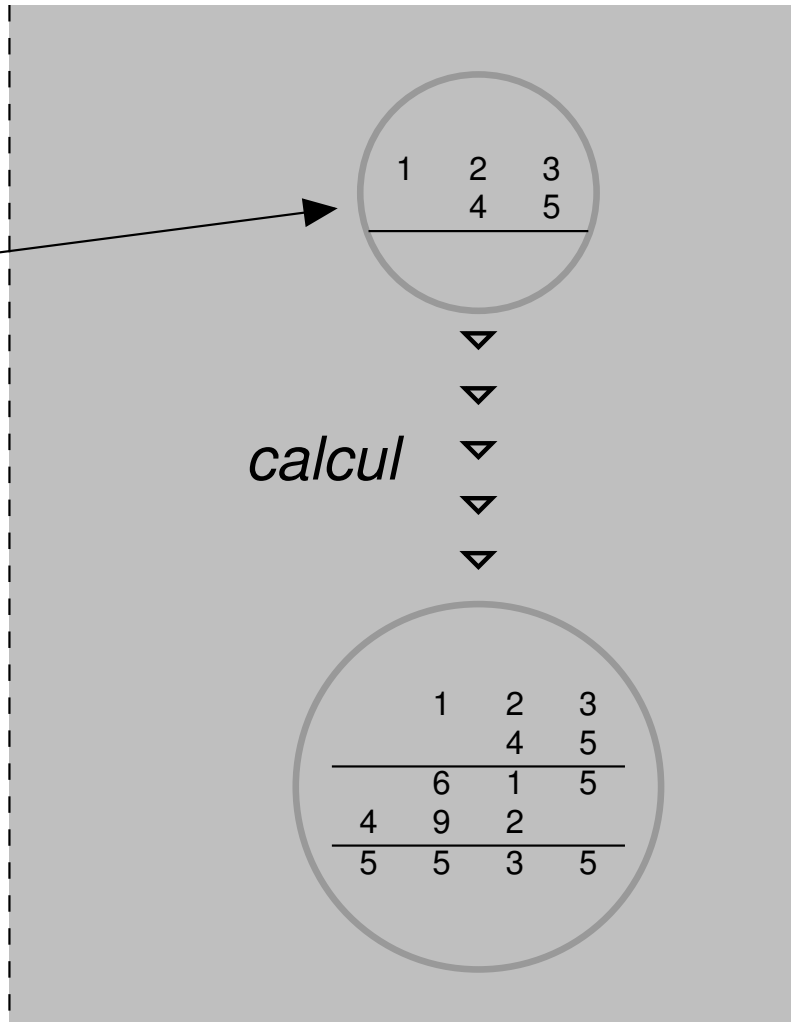
Sortie : $a \times b$

Exemple

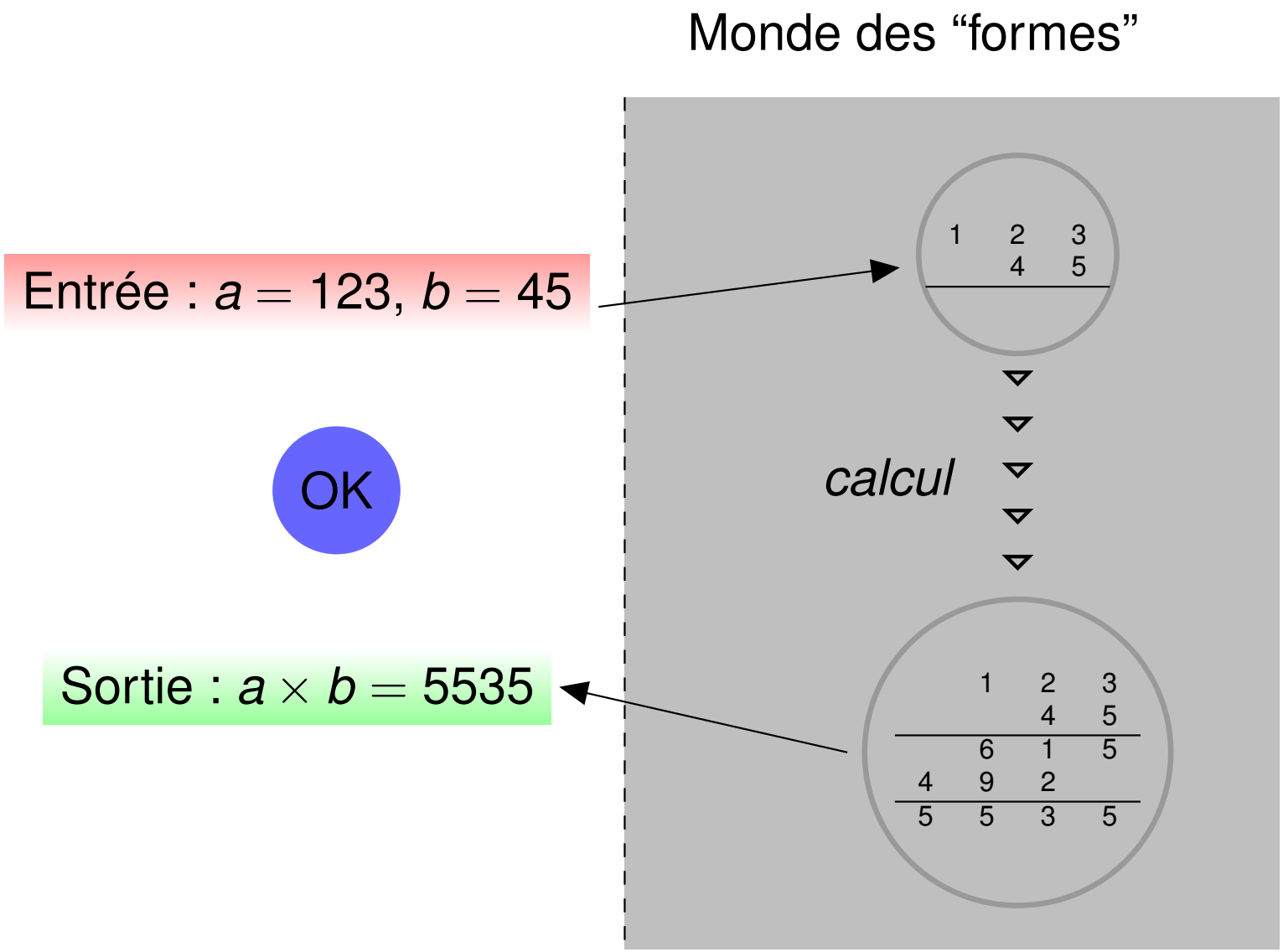
Entrée : $a = 123$, $b = 45$

?

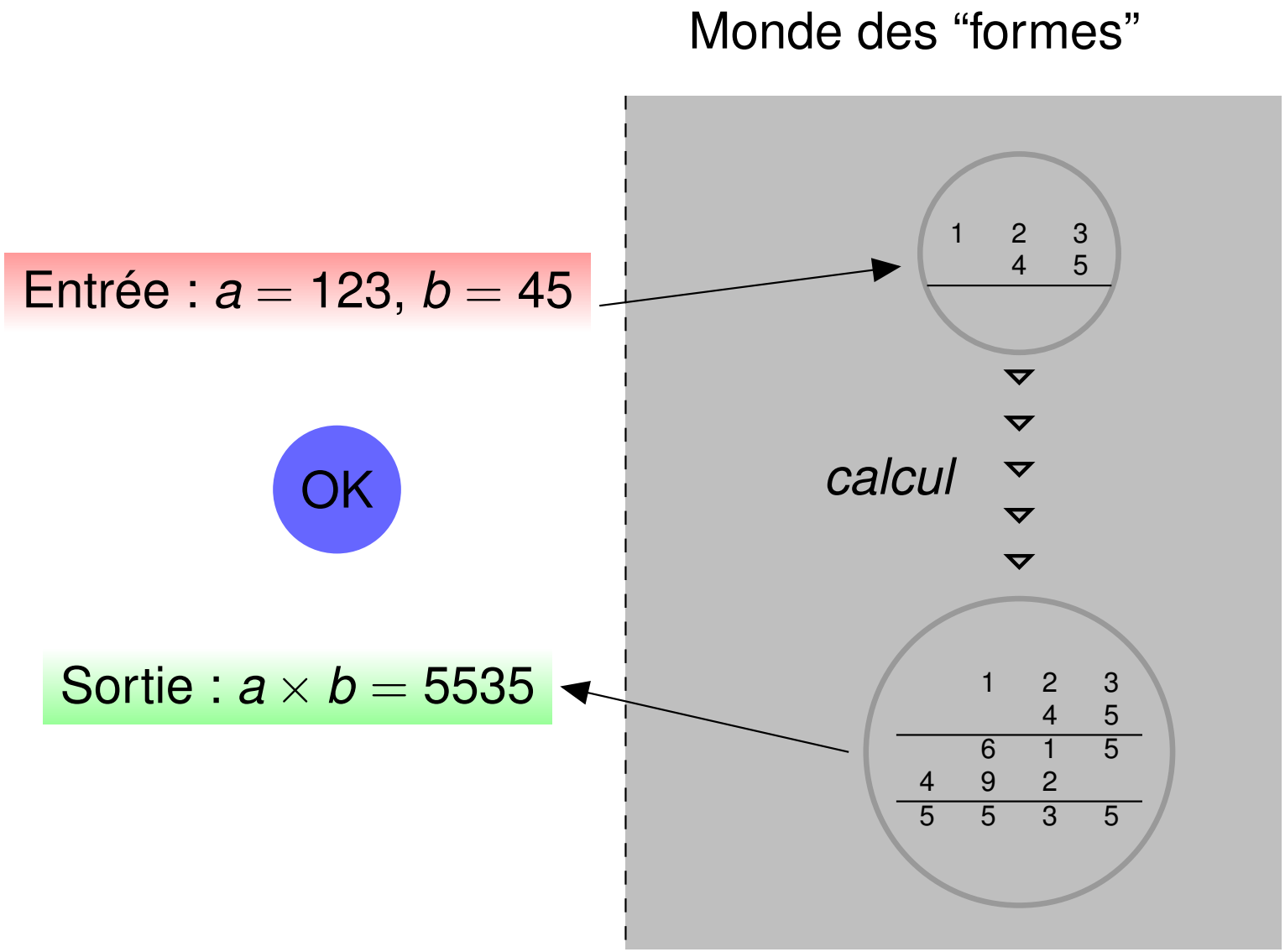
Sortie : $a \times b$



Exemple


Monde des “formes”
Entrée : $a = 123$, $b = 45$
OK
calcul
Sortie : $a \times b = 5535$
1 2 3
4 5
1 2 3
4 5
6 1 5
4 9 2
5 5 3 5

Exemple



Méthode datant du moyen-âge...

Années 1930

A. M. Turing

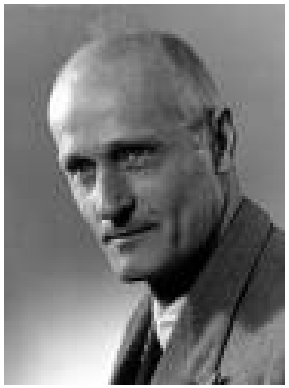
1912–1954

K. Gödel

1906–1978

A. Church

1903–1995



S. C. Kleene

1909–1994

E. Post

1897–1954

J. von Neumann

1903–1957

Modèle standard de l'informatique

- **Les formes ?**

Modèle standard de l'informatique

- **Les formes ?** des mots !

Modèle standard de l'informatique

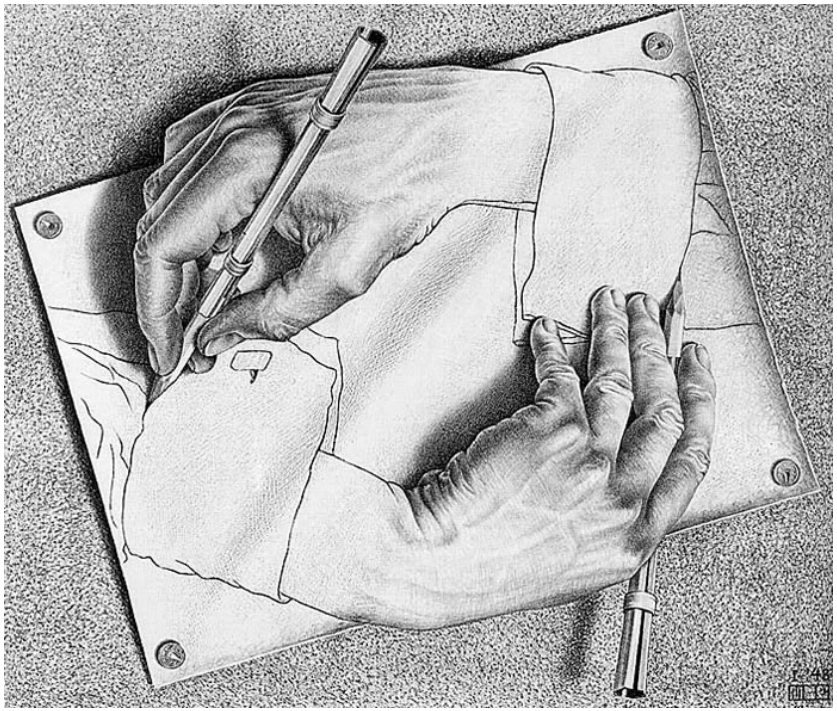
- **Les formes ?** des mots !
- **Les programmes ?**

Modèle standard de l'informatique

- **Les formes ?** des mots !
- **Les programmes ?** encore des mots !

Modèle standard de l'informatique

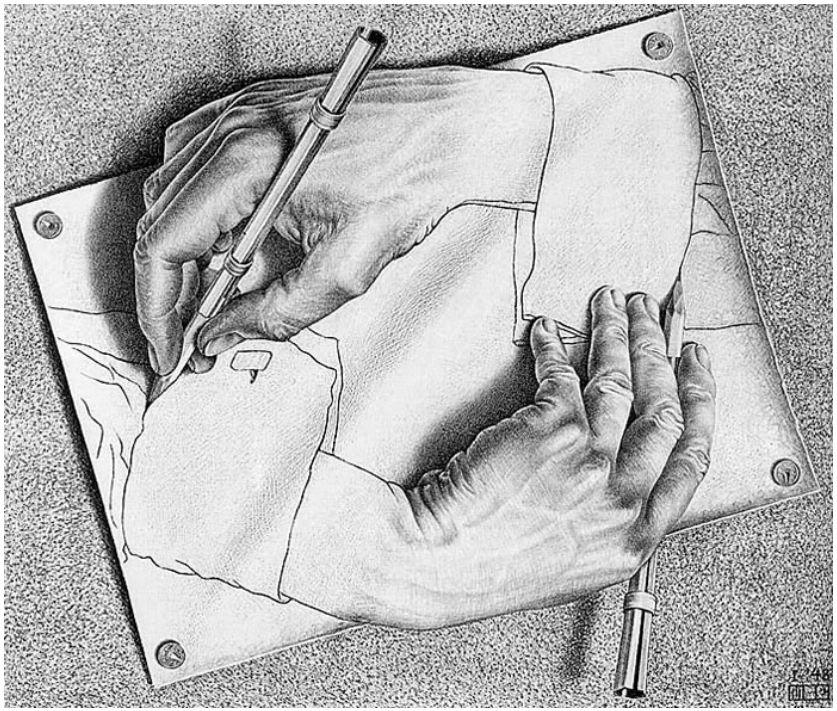
- **Les formes ?** des mots !
- **Les programmes ?** encore des mots !



M. C. Escher, *Drawing Hands*, 1948

Modèle standard de l'informatique

- **Les formes ?** des mots !
- **Les programmes ?** encore des mots !



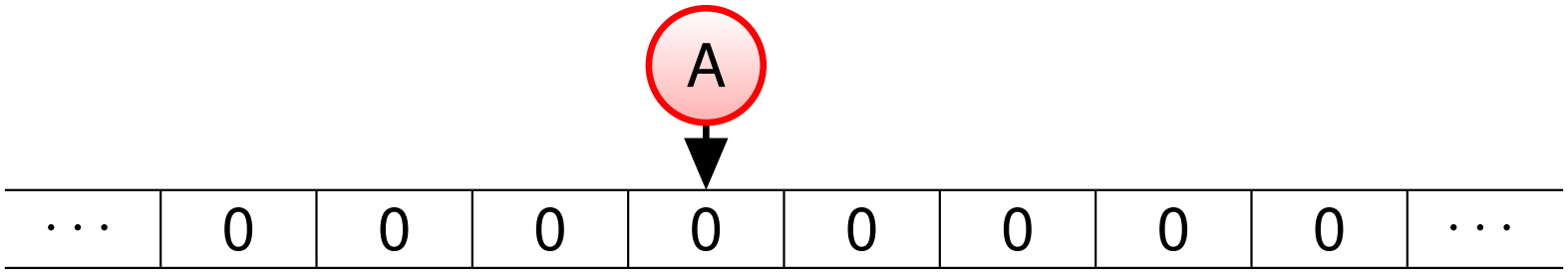
M. C. Escher, *Drawing Hands*, 1948

- mot = suite de symboles (“010101110”, “Père-Noël”, “$\alpha\beta\gamma\delta$”)
- mot $\leftrightarrow$ nombre entier

Les machines de Turing

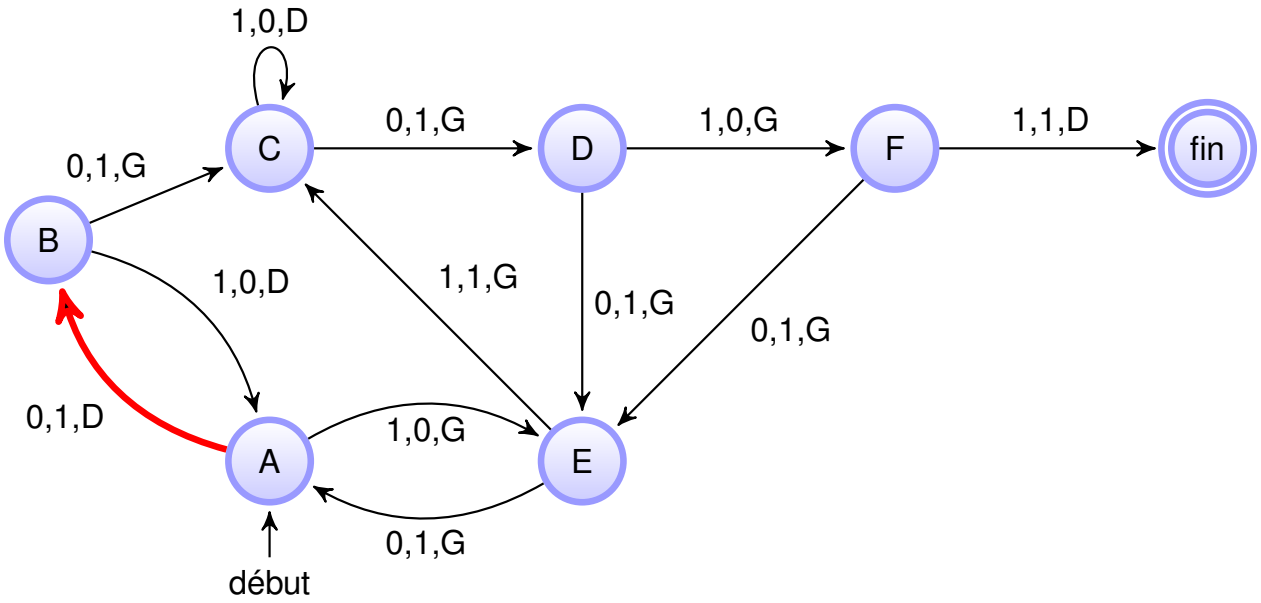
Les machines de Turing

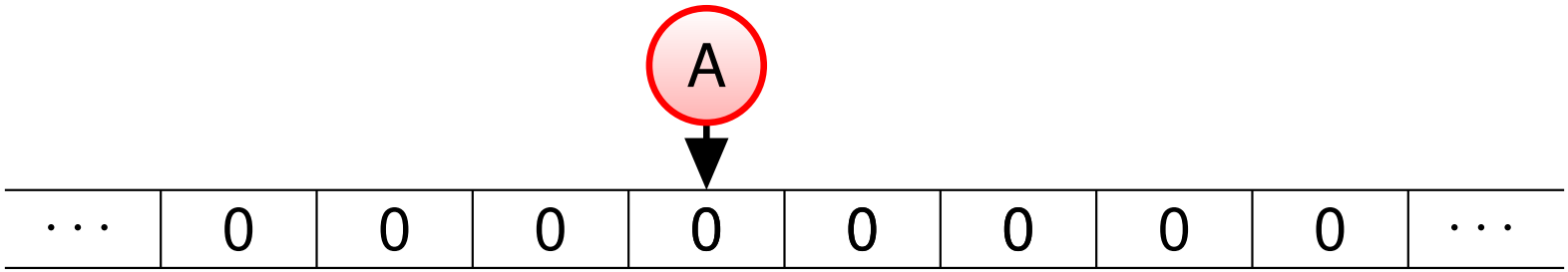
- une sorte de machine à écrire



Les machines de Turing

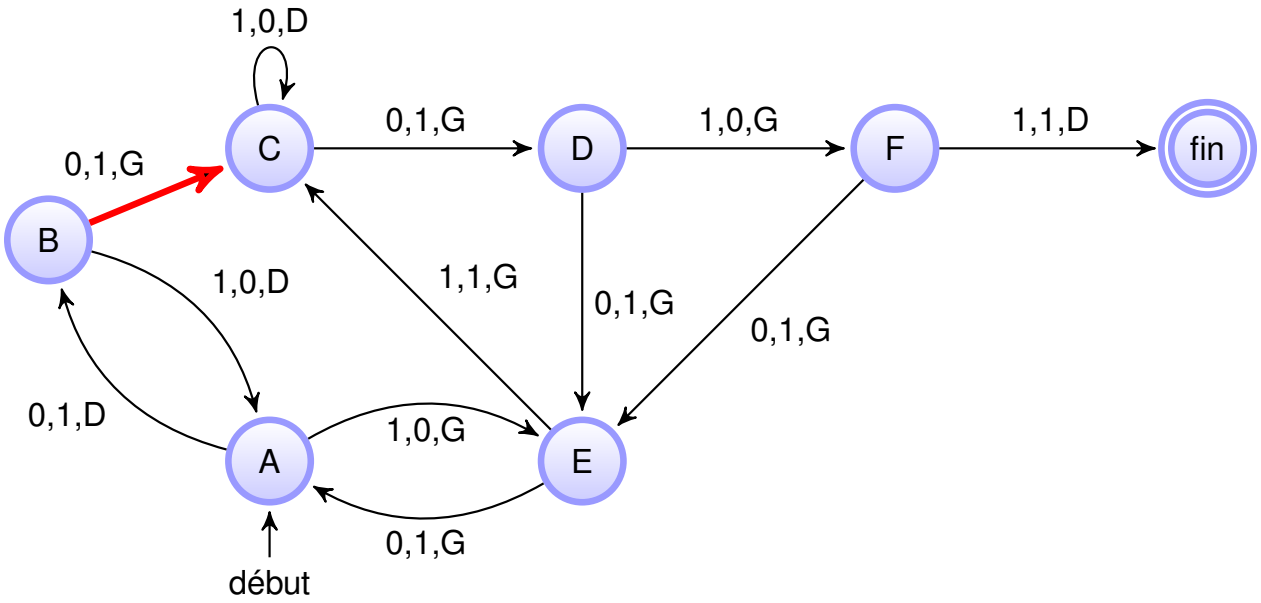
- une sorte de machine à écrire
- guidée par un automate

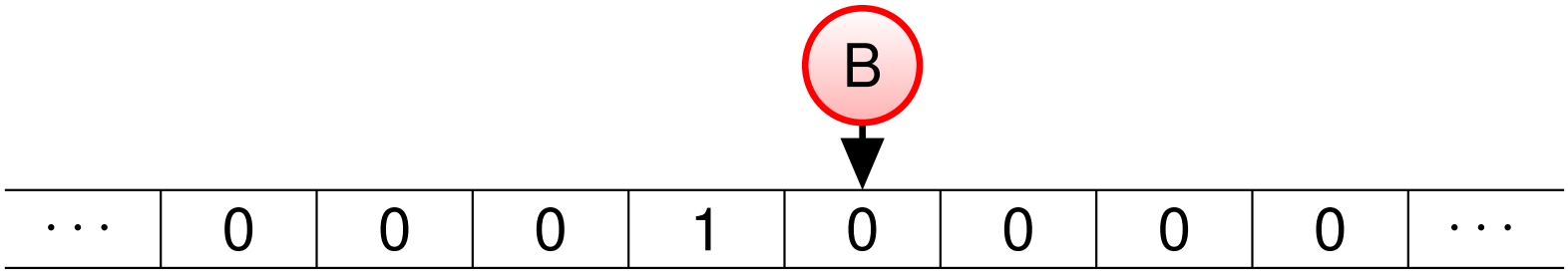




Les machines de Turing

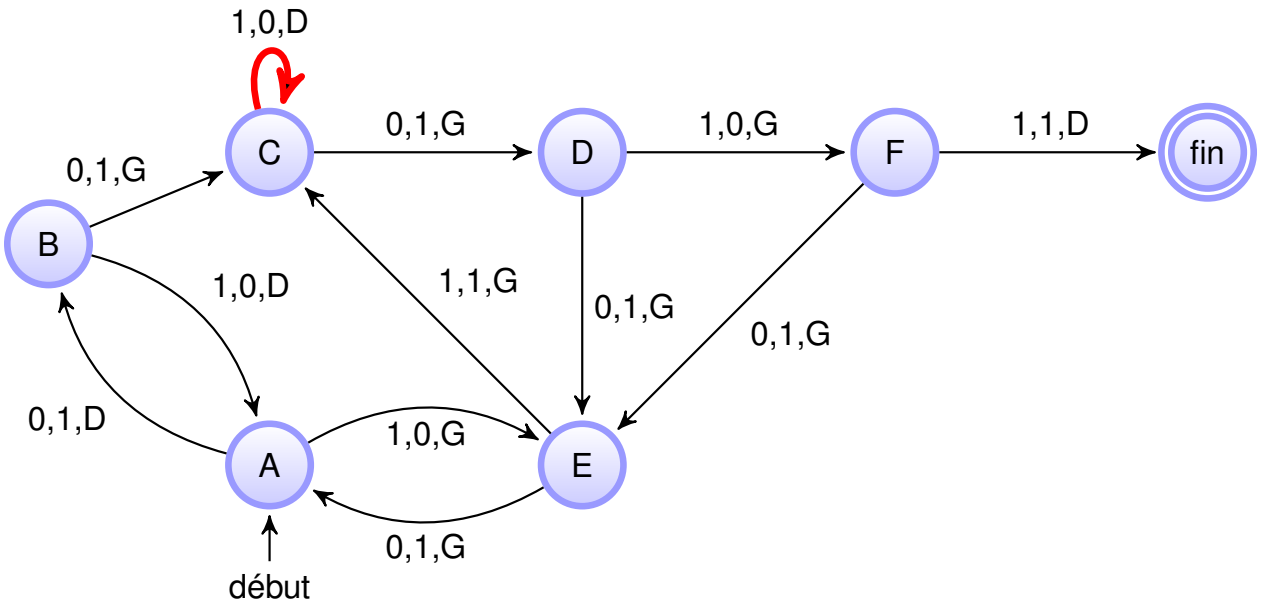
- une sorte de machine à écrire
- guidée par un automate

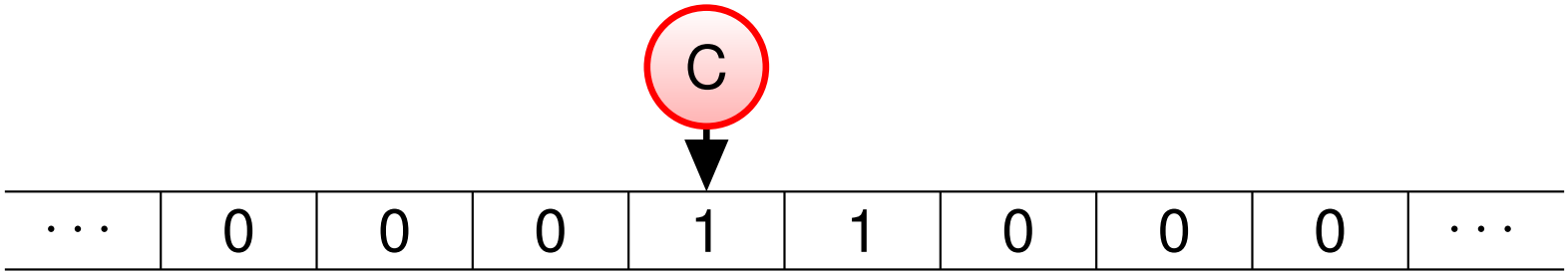




Les machines de Turing

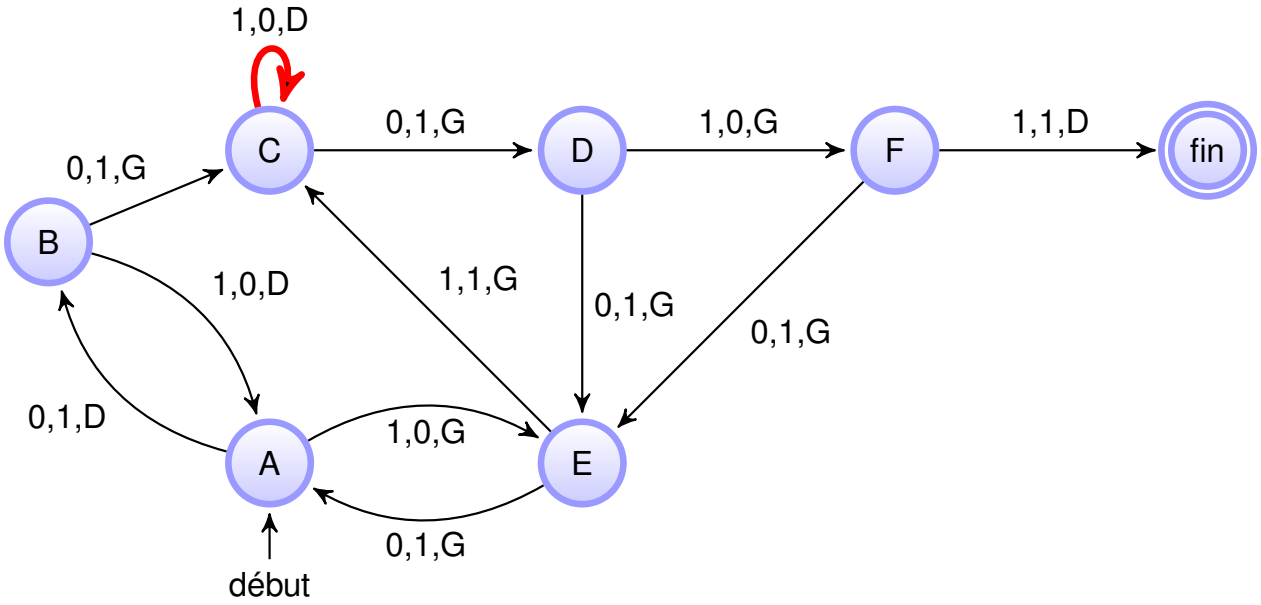
- une sorte de machine à écrire
- guidée par un automate

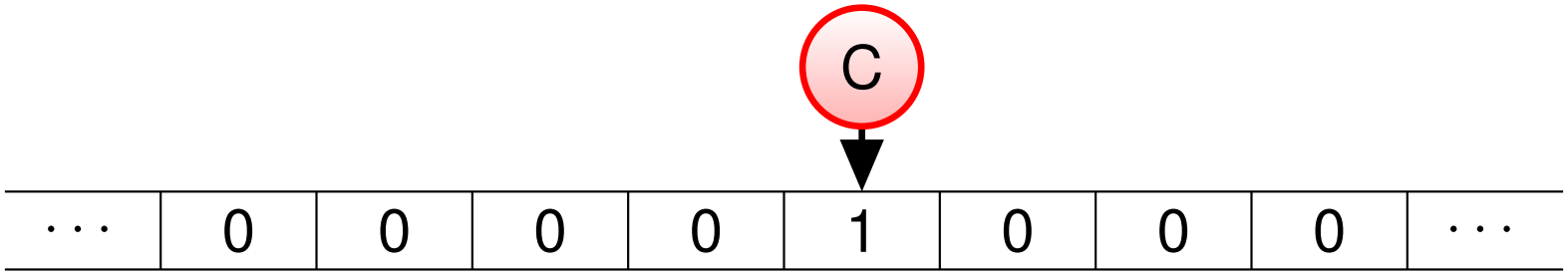




Les machines de Turing

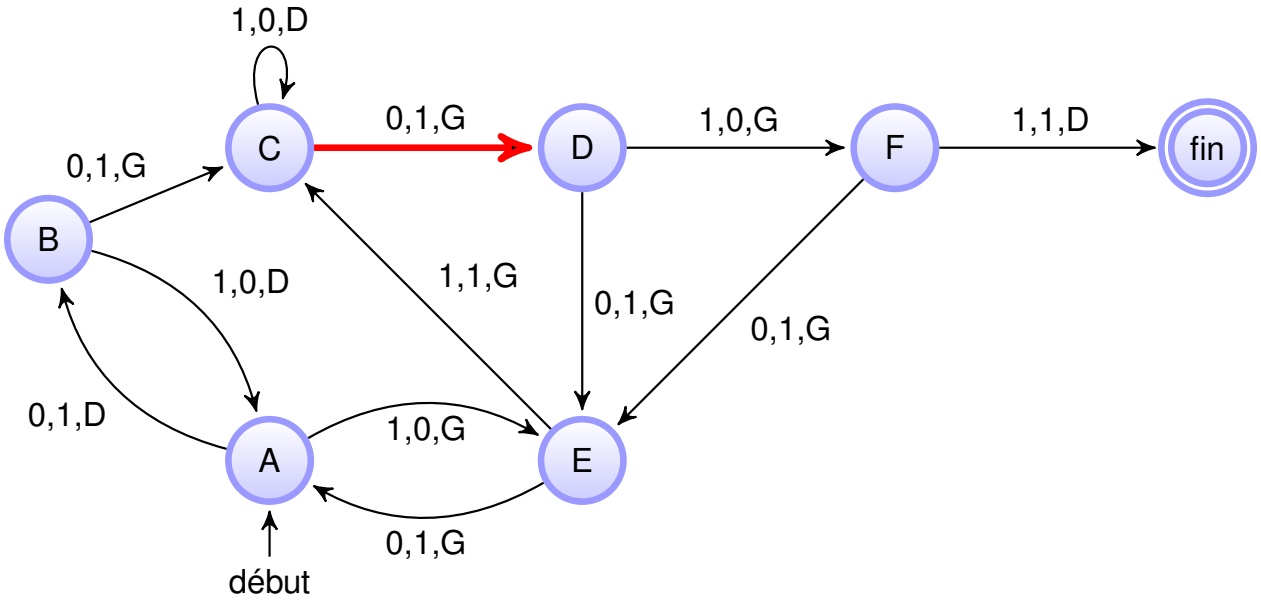
- une sorte de machine à écrire
- guidée par un automate

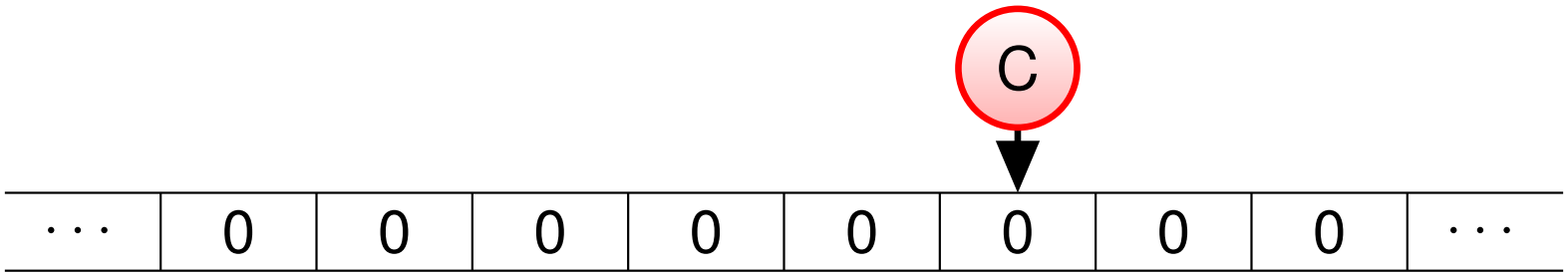




Les machines de Turing

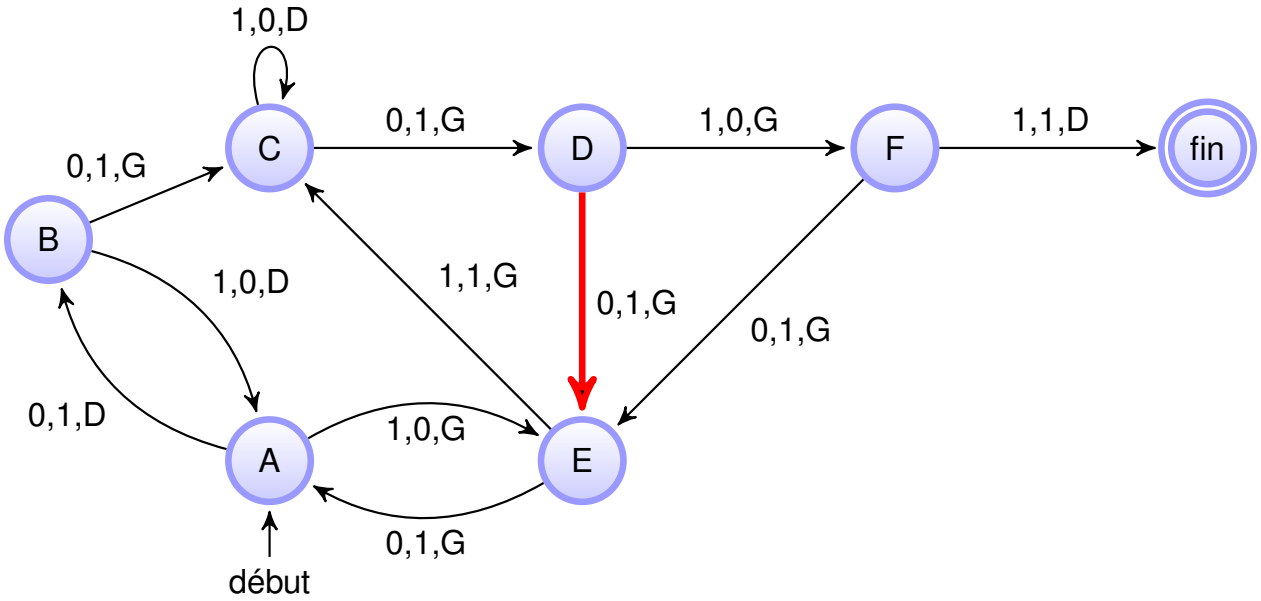
- une sorte de machine à écrire
- guidée par un automate

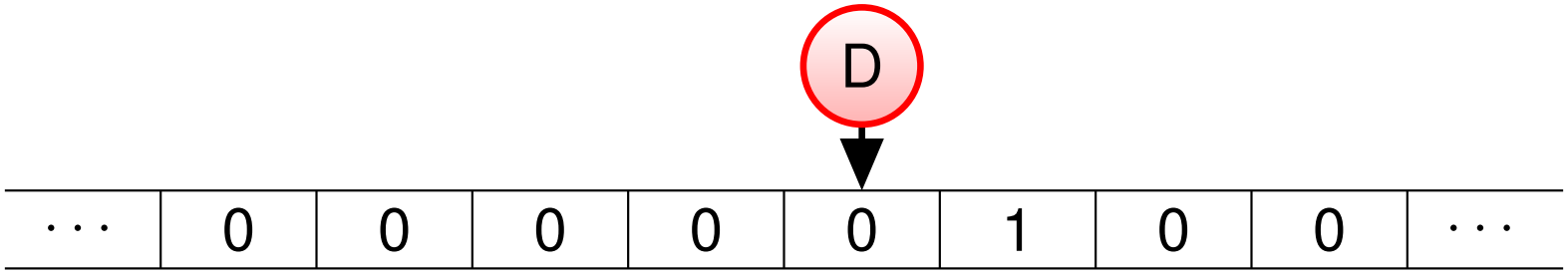




Les machines de Turing

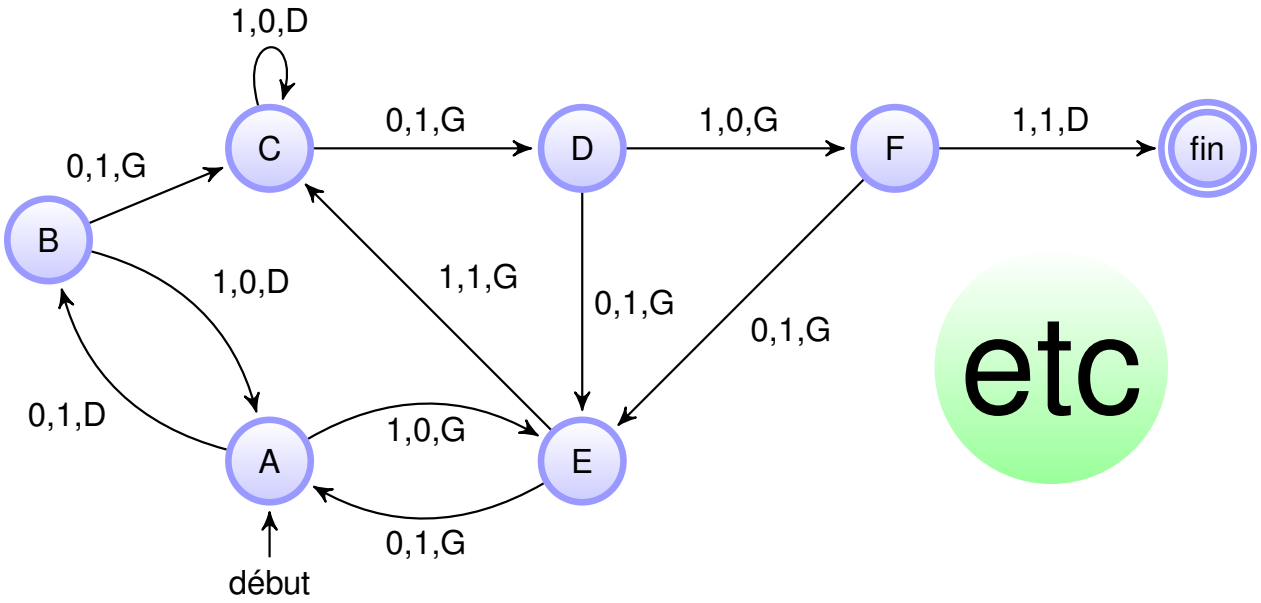
- une sorte de machine à écrire
- guidée par un automate

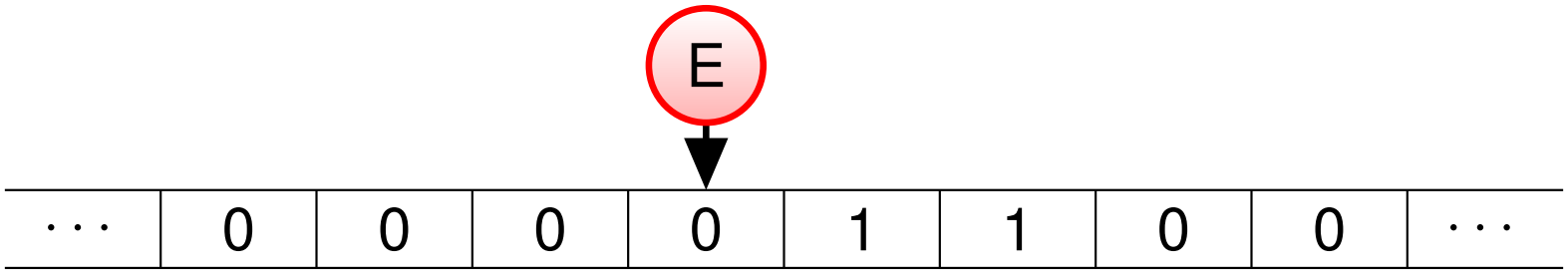




Les machines de Turing

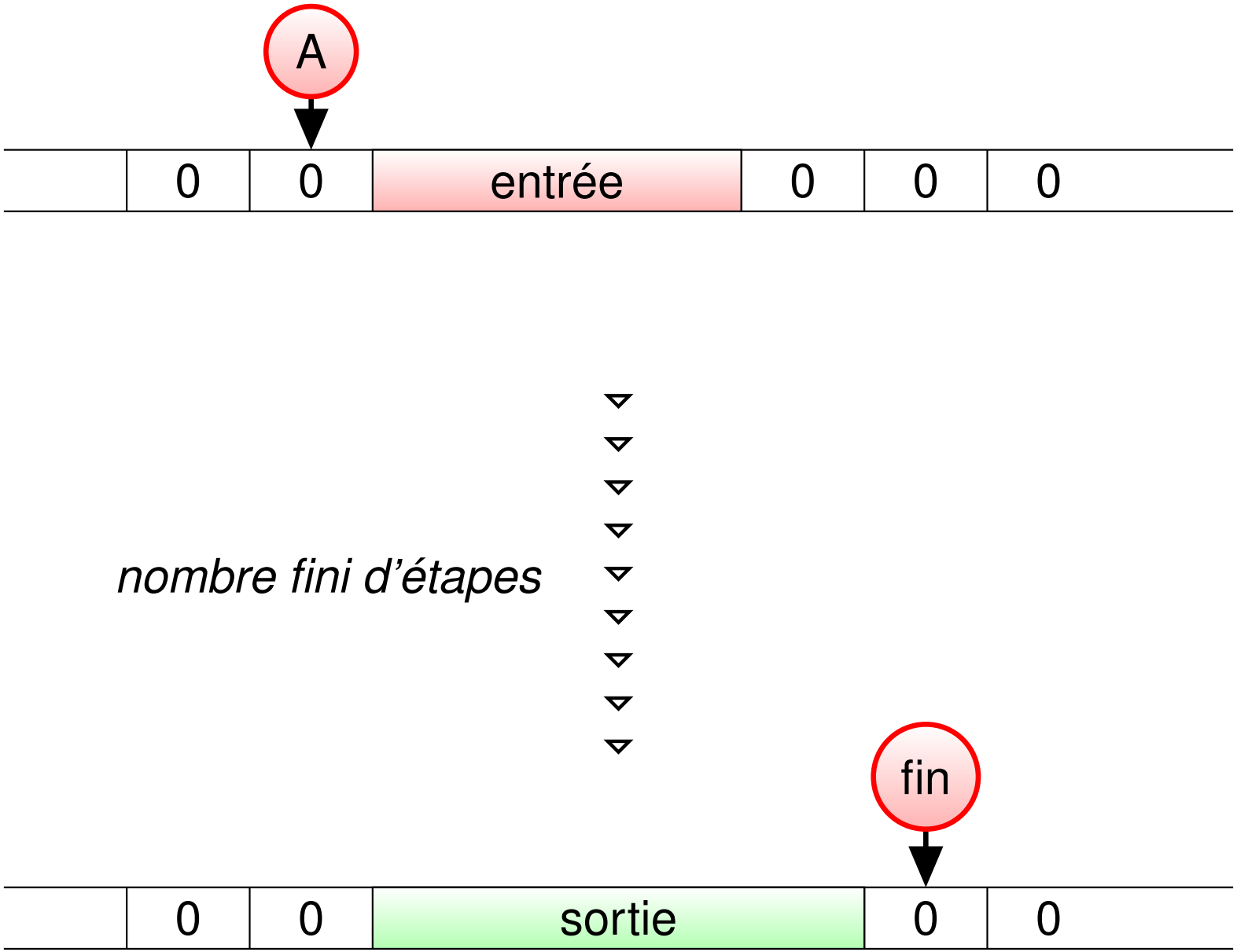
- une sorte de machine à écrire
- guidée par un automate





Calculer avec une machine de Turing

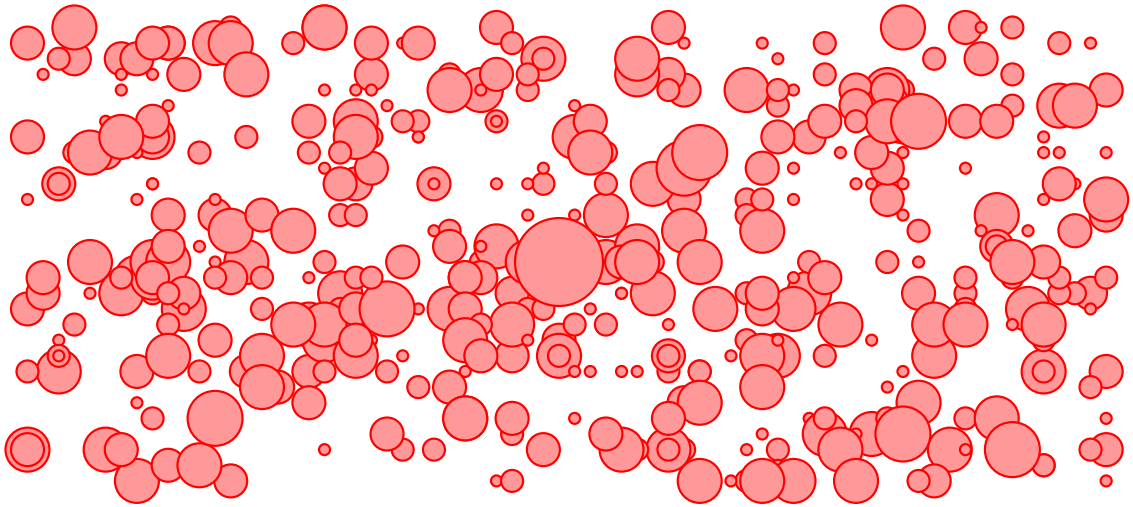
Calculer avec une machine de Turing



Thèse de Church-Turing

« Toute méthode finie et précise qu'un humain peut appliquer avec un papier et un crayon peut être exécutée par une machine de Turing. »

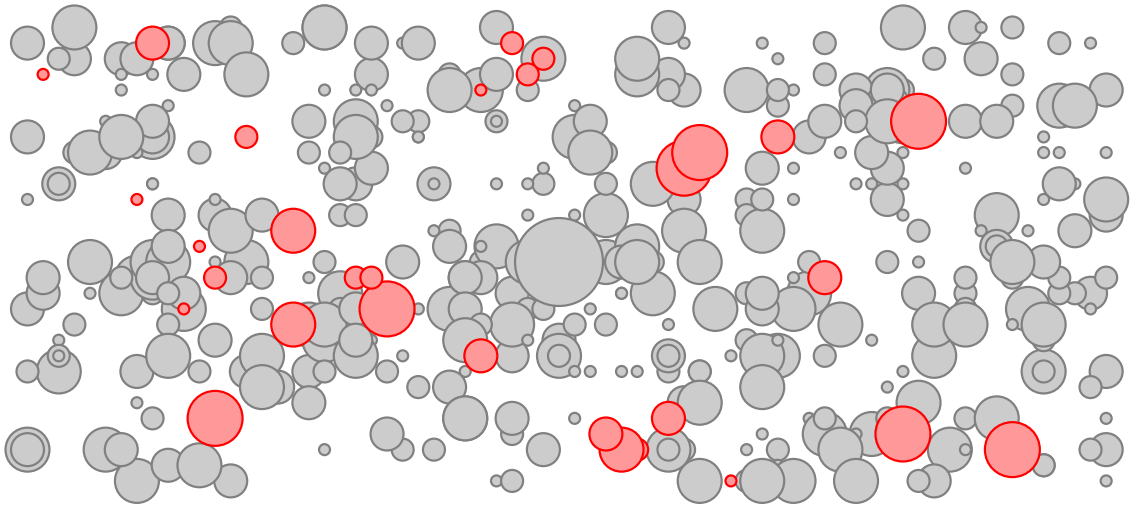
Peut-on tout résoudre par le calcul ?



- trouver le plus court itinéraire,
- prouver un théorème mathématique,
- résoudre un Sudoku,
- gérer le trafic aérien,
- détecter une tumeur dans une image médicale,
- . . .

- une galaxie de problèmes

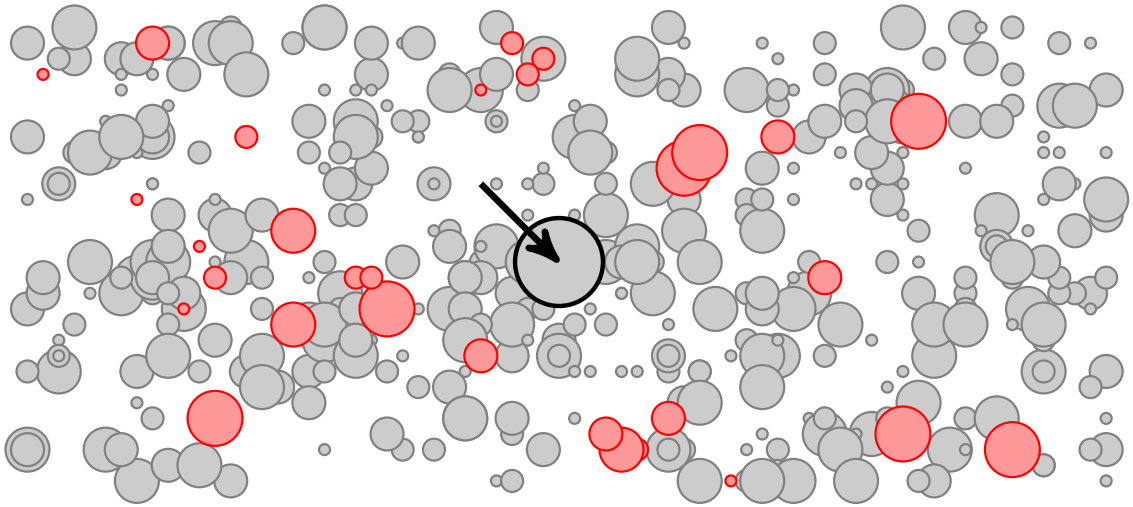
Peut-on tout résoudre par le calcul ?



- trouver le plus court itinéraire,
- prouver un théorème mathématique,
- résoudre un Sudoku,
- gérer le trafic aérien,
- détecter une tumeur dans une image médicale,
- . . .

- une galaxie de problèmes
- certains sont **incalculables** (en gris) !

Peut-on tout résoudre par le calcul ?



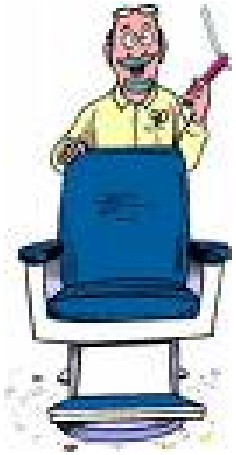
- trouver le plus court itinéraire,
- prouver un théorème mathématique,
- résoudre un Sudoku,
- gérer le trafic aérien,
- détecter une tumeur dans une image médicale,
- . . .

- une galaxie de problèmes
- certains sont **incalculables** (en gris) !

Problème de l'arrêt

Est-ce qu'un programme exécuté sur une entrée donnée finit par renvoyer une réponse ?

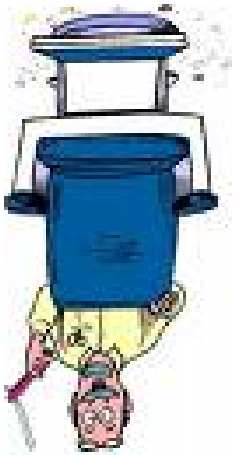
Le problème de l'arrêt : incalculabilité



Dans un petit village, il y a un barbier.
Il rase les gens qui ne se rasent pas eux-mêmes, et seulement ces gens là.

- le paradoxe du barbier

Le problème de l'arrêt : incalculabilité



Dans un petit village, il y a un barbier.
Il rase les gens qui ne se rasent pas
eux-mêmes, et seulement ces gens là.

Par qui est rasé le barbier ?

- le paradoxe du barbier

Le problème de l'arrêt : incalculabilité

*Dans un petit village il y a un **SuperProg**. Il peut déterminer si un programme s'arrête ou pas.*

- le paradoxe du barbier
- un **SuperProg** ?

Le problème de l'arrêt : incalculabilité

Dans ce village, il y a alors aussi le programme suivant :

ProgBarbier (*sur l'entrée "P"*)

Demander à **SuperProg** si **P** s'arrête sur l'entrée "**P**" :

- si oui $\rightsquigarrow$ tourner indéfiniment en rond
- si non $\rightsquigarrow$ s'arrêter

- le paradoxe du barbier
- un **SuperProg** ?
- si oui, alors un **ProgBarbier**

Le problème de l'arrêt : incalculabilité

Dans ce village, il y a alors aussi le programme suivant :

ProgBarbier (*sur l'entrée "P"*)

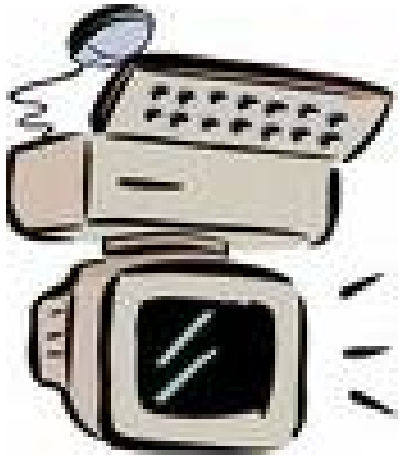
Demander à **SuperProg** si **P** s'arrête sur l'entrée "**P**" :

- si oui $\rightsquigarrow$ tourner indéfiniment en rond
- si non $\rightsquigarrow$ s'arrêter

Que fait ProgBarbier sur l'entrée "ProgBarbier" ?

- le paradoxe du barbier
- un **SuperProg** ?
- si oui, alors un **ProgBarbier**

Le problème de l'arrêt : incalculabilité



Dans ce village, il y a alors aussi le programme suivant :

ProgBarbier (*sur l'entrée "ProgBarbier"*)

Demander à **SuperProg** si **ProgBarbier** s'arrête sur l'entrée "**ProgBarbier**" :

- si oui $\rightsquigarrow$ tourner indéfiniment en rond
- si non $\rightsquigarrow$ s'arrêter

Que fait ProgBarbier sur l'entrée "ProgBarbier" ?

- le paradoxe du barbier
- un **SuperProg** ?
- si oui, alors un **ProgBarbier**
- **SuperProg** n'existe pas !

EFFERALGAN

L'importance du problème de l'arrêt

- difficile même pour les programmes de 10 lignes ?

L'importance du problème de l'arrêt

- difficile même pour les programmes de 10 lignes ? OUI !

```
open Num;;
let est_premier x =
  let rec test_diviseur d =
    d*/d>/x or (mod_num x d<>/(Int 0) && test_diviseur (succ_num d))
  in test_diviseur (Int 2) in
let rec essai n = function
    Int 1 -> raise Not_found
  | i -> if est_premier i && est_premier (n-/i)
         then essai (n+/(Int 2)) n else essai n (pred_num i)
in essai (Int 4) (Int 2);;
```

Exécution

L'importance du problème de l'arrêt

- difficile même pour les programmes de 10 lignes ? OUI !
- difficulté d'écrire des programmes corrects $\rightsquigarrow$ "bugs"

L'importance du problème de l'arrêt

- difficile même pour les programmes de 10 lignes ? OUI !
- difficulté d'écrire des programmes corrects $\rightsquigarrow$ "bugs"

Théorème de Rice

Il n'y a **pas de méthode générale** pour vérifier si un **programme** fait bien ce que l'on veut.

L'importance du problème de l'arrêt

- difficile même pour les programmes de 10 lignes ? OUI !
- difficulté d'écrire des programmes corrects $\rightsquigarrow$ "bugs"
- l'indécidabilité logique (K. Gödel)

L'importance du problème de l'arrêt

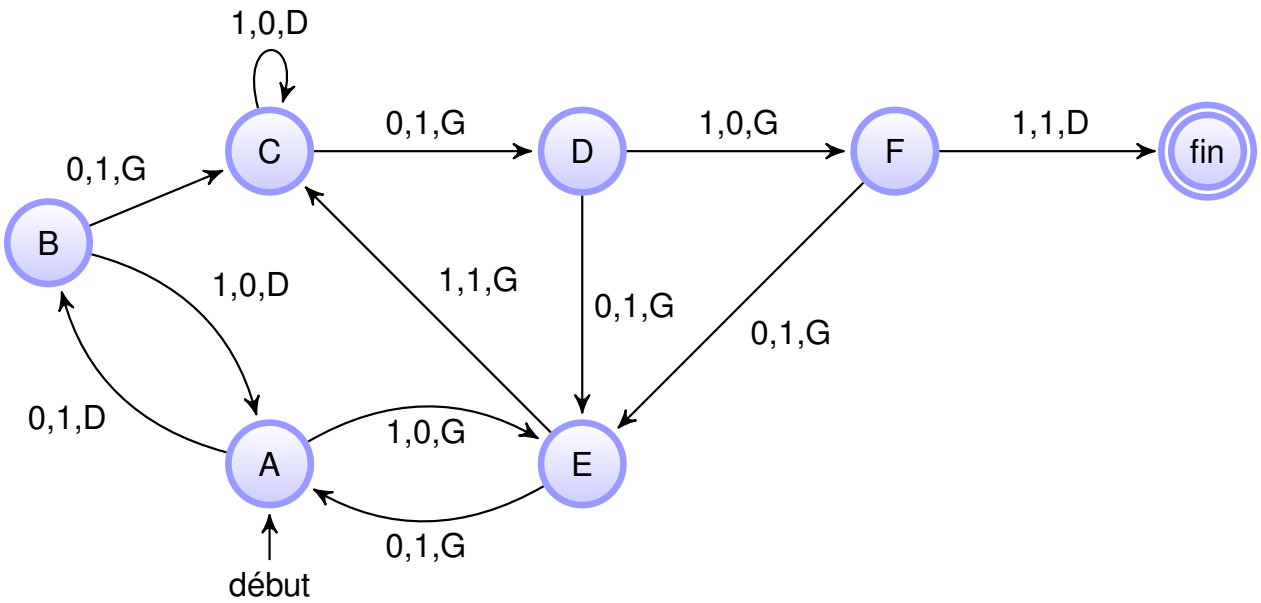
- difficile même pour les programmes de 10 lignes ? OUI !
- difficulté d'écrire des programmes corrects $\rightsquigarrow$ "bugs"
- l'indécidabilité logique (K. Gödel)

En mathématiques...

1. il existe des énoncés **indécidables** :
   - on ne pourra jamais prouver qu'ils sont **vrai**
   - on ne pourra jamais prouver qu'ils sont **faux**
2. même si on les "**décide**" un à un, il reste toujours des indécidables

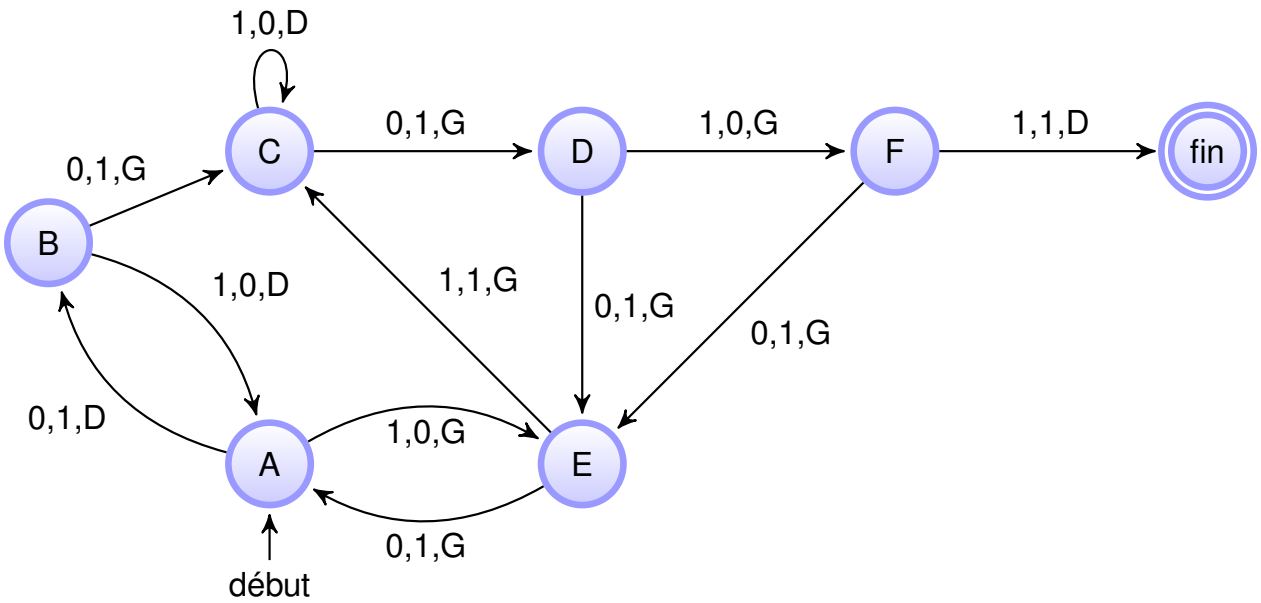
L'importance du problème de l'arrêt

- difficile même pour les programmes de 10 lignes ? OUI !
- difficulté d'écrire des programmes corrects $\rightsquigarrow$ "bugs"
- l'indécidabilité logique (K. Gödel)
- les monstres combinatoires



L'importance du problème de l'arrêt

- difficile même pour les programmes de 10 lignes ? OUI !
- difficulté d'écrire des programmes corrects $\rightsquigarrow$ "bugs"
- l'indécidabilité logique (K. Gödel)
- les monstres combinatoires



La machine finit-elle par s'arrêter ?

La machine finit-elle par s'arrêter ?

La machine finit-elle par s'arrêter ?

OUI ! mais seulement après plus de

$10^{2\,879}$ étapes ! ! !

- **petite** machine (2 symboles, 6 états)

La machine finit-elle par s'arrêter ?

OUI ! mais seulement après plus de

$10^{2\,879}$ étapes ! ! !

- **petite** machine (2 symboles, 6 états)
- comportement **monstrueux**...

Ce nombre est physiquement hors d'atteinte

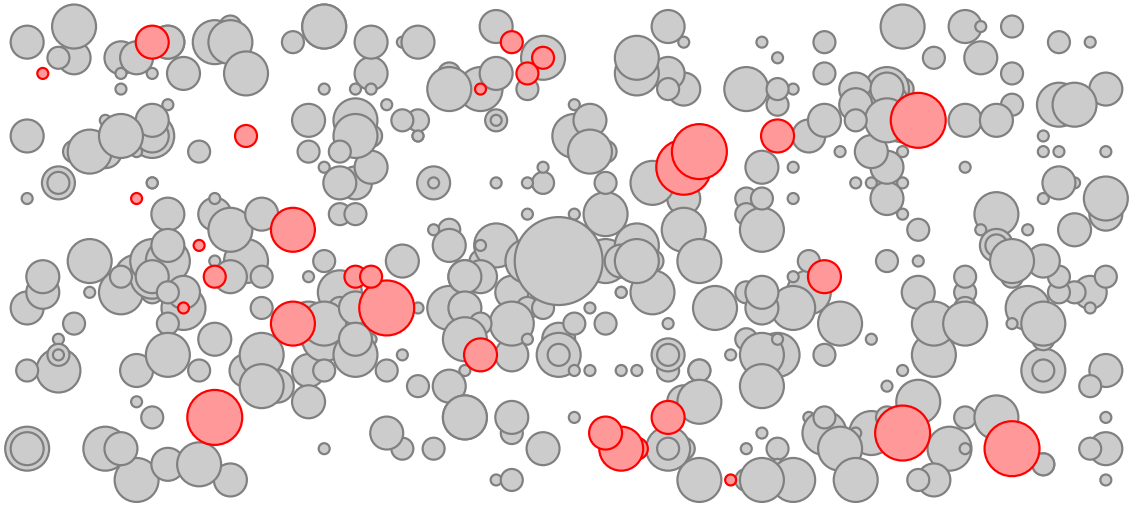
A : âge estimé de l'univers $\approx 10^{18}$ secondes

T : temps de Planck $\approx 10^{-44}$

N : nombre de particules dans l'univers observable $\approx 10^{90}$

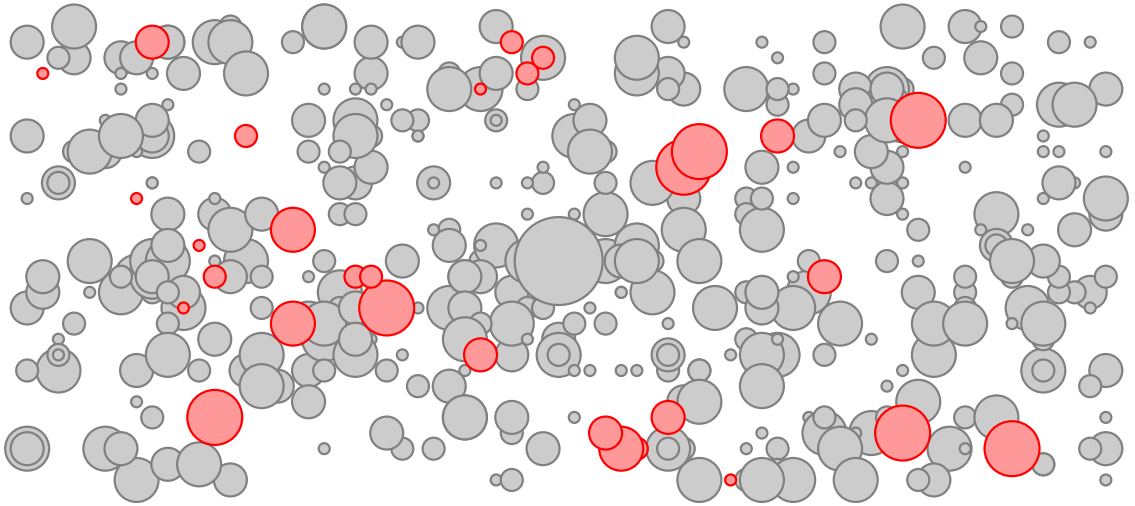
$N \times A/T$: $\approx 10^{152}$

Le monde du calculable



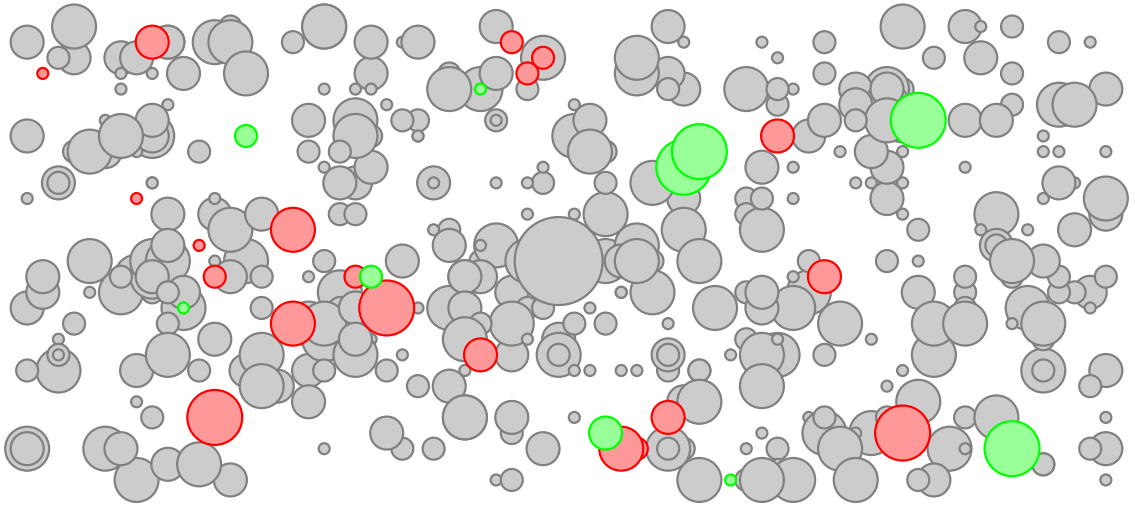
- il y a quand même des problèmes calculables !

Le monde du calculable



- il y a quand même des problèmes calculables !
- peut-on les résoudre efficacement ?

Le monde du calculable



- il y a quand même des problèmes calculables !
- peut-on les résoudre efficacement ?

P : problèmes dont on peut **trouver** efficacement la solution

Problème du plus court chemin

Problème du plus court chemin

- *Données :*
  1. un réseau routier
  2. un point **A**, un point **B**

Problème du plus court chemin

- *Données :*
  1. un réseau routier
  2. un point **A**, un point **B**
- **Question :** quel est le plus court chemin de **A** vers **B** ?

Problème du plus court chemin

- *Données :*
  1. un réseau routier
  2. un point **A**, un point **B**
- **Question :** quel est le plus court chemin de **A** vers **B** ?
- une solution efficace existe ! (années 1960)

E. W. Dijkstra
(1930–2002)

Problème du plus court chemin

- *Données :*
  1. un réseau routier
  2. un point **A**, un point **B**
- **Question :** quel est le plus court chemin de **A** vers **B** ?
- une solution efficace existe ! (années 1960)
- des applications concrètes

E. W. Dijkstra
(1930–2002)

Problème du sac à dos (du cambrioleur)

Problème du sac à dos (du cambrioleur)

Dans une maison, il y a

1. des objets de différentes tailles et différentes valeurs
2. un voleur avec un sac à dos d'une certaine capacité

Problème du sac à dos (du cambrioleur)

100 000 € ?

Dans une maison, il y a

1. des objets de différentes tailles et différentes valeurs
2. un voleur avec un sac à dos d'une certaine capacité

Peut-il remplir son sac d'objets pour une valeur d'au moins 100 000 € ?

Problème du sac à dos (du cambrioleur)

100 000 € ?

Dans une maison, il y a

1. des objets de différentes tailles et différentes valeurs
2. un voleur avec un sac à dos d'une certaine capacité

Peut-il remplir son sac d'objets pour une valeur d'au moins 100 000 € ?

- Pas d'algorithme efficace connu à ce jour !

Problème du sac à dos (suite)

Problème du sac à dos (suite)

- on peut **vérifier facilement** si une solution est bonne :
  1. vérifier que les objets rentrent dans le sac
  2. vérifier que leur valeur totale est $\geq 100\,000$ €

Problème du sac à dos (suite)

- on peut **vérifier facilement** si une solution est bonne :
  1. vérifier que les objets rentrent dans le sac
  2. vérifier que leur valeur totale est $\geq 100\,000$ €

NP : problèmes dont on peut **vérifier** efficacement les solutions

Exemples

- le médecin peut il faire sa tournée de visites en 4h ?
- y a-t-il ici 10 personnes qui se connaissent mutuellement ?
- y a-t-il un emploi du temps qui satisfait tout le monde ?

Problème du sac à dos (suite)

- on peut **vérifier facilement** si une solution est bonne :
  1. vérifier que les objets rentrent dans le sac
  2. vérifier que leur valeur totale est $\geq 100\,000$ €

NP : problèmes dont on peut **vérifier** efficacement les solutions

Exemples

- le médecin peut il faire sa tournée de visites en 4h ?
- y a-t-il ici 10 personnes qui se connaissent mutuellement ?
- y a-t-il un emploi du temps qui satisfait tout le monde ?

- peut-on **trouver** leurs solutions efficacement ?

Trouver / Vérifier

Trouver / Vérifier

P = NP ?

$1 000 000 ! ! !

$1 000 000 ! ! !

Trouver / Vérifier

P = NP ?

$1 000 000 ! ! !

$1 000 000 ! ! !

- La plupart des spécialistes pensent que **P** $\neq$ **NP**

Trouver / Vérifier

P = NP ?

$1 000 000 ! ! !

$1 000 000 ! ! !

- La plupart des spécialistes pensent que **P** $\neq$ **NP**

Théorème

Il y a un algorithme efficace pour le problème du sac à dos *si et seulement si* **P=NP**.

- **sac-à-dos** concentre toute la difficulté de **NP**

Trouver / Vérifier

P = NP ?

$1 000 000 ! ! !

$1 000 000 ! ! !

- La plupart des spécialistes pensent que **P** $\neq$ **NP**

Théorème

Il y a un algorithme efficace pour le problème du sac à dos *si et seulement si* **P=NP**.

- **sac-à-dos** concentre toute la difficulté de **NP**
- moralité : est-il raisonnable de lui chercher un algorithme efficace ?

Complexité

Complexité

« Les machines sont bêtes : elles ne font que répéter mécaniquement des choses simples. »

Complexité

« Les machines sont bêtes : elles ne font que répéter mécaniquement des choses simples. »

oui mais...

Complexité

« Les machines sont bêtes : elles ne font que répéter mécaniquement des choses simples. »

oui mais...

- *est-ce que les interactions **microscopiques** permettent toujours de comprendre les effets **macroscopiques** ?*
- *est-ce que le **tout** peut toujours se déduire de l'analyse de ses **parties** ?*
- *est-ce qu'une **loi d'évolution simple** permet toujours de **prédire le phénomène** qu'elle décrit ?*

La mode vestimentaire

La mode vestimentaire

La mode en 2 mots :

1. c'est cyclique
2. il faut suivre la mode, sinon on est ringard

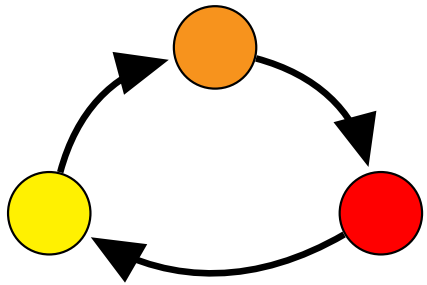
La mode vestimentaire

La mode en 2 mots :

1. c'est cyclique
2. il faut suivre la mode, sinon on est ringard

Une modélisation simple

1. cycle de 3 modes :



2. autour de moi, combien sont passés à la "nouvelle" mode ?
   - si 3 ou plus, je passe aussi à la "nouvelle" mode
   - sinon, je ne change pas

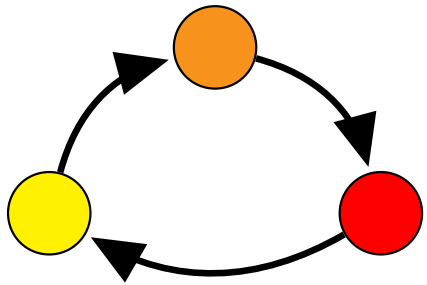
La mode vestimentaire

La mode en 2 mots :

1. c'est cyclique
2. il faut suivre la mode, sinon on est ringard

Une modélisation simple

1. cycle de 3 modes :



2. autour de moi, combien sont passés à la "nouvelle" mode ?
   - si 3 ou plus, je passe aussi à la "nouvelle" mode
   - sinon, je ne change pas

▷ **Automate de Griffeath** Démo

Fourmi de Langton

Démo

Plaidoyer pour l'informatique

Plaidoyer pour l'informatique

1. **une science jeune et utile**

   en mouvement, accessible, applications omniprésentes

2. **un pont entre l'abstrait et le concret**

   raisonnements logiques $\leftrightarrow$ *programmation concrète*

3. **un carrefour de disciplines**

   mathématiques, physique, biologie,...

Bibliographie

1. *« Une histoire de l'informatique. »*
   Philippe BRETON (Points)

2. Jean-Paul DELAHAYE
   `http://www2.lifl.fr/~delahaye/`

3. Revue en ligne *Interstices*
   `http://interstices.info`

4. *« Computer Science Unplugged »*
   `http://csunplugged.org/`