L'Informatique : Science et Technique

Kamel Barkaoui

Beït al Hikma, 26 Février 2016

L’Informatique

- INFORmation + autoMATIQUE = INFORMATIQUE

- **son objet :**

  – **Traitement automatique de l'information via l’exécution de programmes informatiques par des machines**

L'évolution de l'Informatique

3 grandes étapes (en cours de progression):

- **Informatique fondamentale**

- **Informatique distribuée**

- **Informatique ubiquitaire** **(en cours de lancement)**

Informatique fondamentale (dés les années 30)

- **Les 4 concepts de base :**
  - **information, algorithme, machine, langage**

Quelles sont les capacités et les limites des ordinateurs ?

- **Les fondements propres et liés à d'autres sciences**
  - **Logique mathématique**
  - **Théorie de la calculabilité (les modèles de calcul)**
  - **Théorie de la complexité**
  - Mathématiques discrètes
  - Physique, électronique solide, science des matériaux
    - technologie des composants
  - Sciences de la vie, génétique, biotechnologies

Informatique distribuée (dés les années 1970)

- **Convergence avec les télécommunications et l'audiovisuel**
  - réseaux numériques : Internet , le Web , les réseaux locaux,..
  - interopérabilité, standardisation,

- **Capacités de stockage de l'information et de traitement des données**
  - microprocesseurs multi-cœur, grid computing
  - SGBD, SID, virtualisation, cloud computing

- **Programmation à large échelle** (La crise du logiciel)
  - analyse (quoi faire), architecture (comment faire) interfaces H/M
  - paradigme objet, l'approche composants, test, vérification formelle
  - plusieurs millions de lignes de code dans un téléphone portable

Les questions clés liées à la sûreté de fonctionnement et qualité du logiciel et à l'environnement (green computing)

L'étendue de ses applications

- **L'économie du savoir et de l'innovation**
  - modélisations et simulations
    - complexe scientifique-militaire-industriel
    - écosystèmes, climatologie, santé,..
  - réseaux de diffusion du savoir et systèmes nationaux d'innovation
    - collaboration universités & industrie (biens et services)
- **La société numérique**
    - le Web social (Web 2.0) : blogs, forums, réseaux sociaux , le crowdsourcing
    - les services Web, BPM
- **Les retombées scientifiques, techniques et économiques**
    - le Web sémantique (Web 3.0) : bases de connaissances interopérables (métadonnées)
    - la science de données (Data science) : recherche de modèles dans les mégadonnées /Big data (data mining, process mining)
      - Big Data représentera 8 % du PIB européen en 2020, soit environ 1 600 milliards d'euros

Informatique ubiquitaire (dés 2010)

La fusion entre le monde numérique et le monde physique

« *A world in which information can be accessed from anywhere, at any time, by anyone and anything* »

« intelligence ambiante »,
Informatique « pervasive, diffuse, omniprésente, invisible »

- **Internet des objets (IoT)**
  - objets de tout genre connectés permettant de fournir des informations en tout genre
    - **puces électroniques enfouies dans les objets** (RFID, micro-capteurs)
    - 15 milliards d'objets connectés à internet (CA de 400 millions d'euros) en 2015
      - 50 et 80 milliards (CA de 1 700 milliards d'euros) en 2020

Informatique ubiquitaire

- **Les systèmes cyber-physiques**
  - Le logiciel embarqué
  - le concept d'Industry 4.0 (smart factories)
- **L'Informatique contextuelle** (context-aware computing)
  - les systèmes orientés services (le Web 3.0)
    - anticiper / prédire en temps réel le comportement des usagers
  - l'industrie de la surveillance : exploitation des donnés personnelles
- **Le concept machine to machine (M2M)**
  - unification homme-machine
- **La « réalité augmentée »**
  - intégration monde réel-monde virtuel en temps réel

Questions clés liées à la vie privée, à la responsabilité des personnes, à la compositionnalité et l'adaptabilité de systèmes hétérogènes

Les fondements de l’Informatique

- Théorie de l'information
  - **codage, compression, transmission**
- Théorie algorithmique de l’information
  - **décidabilité, complexité**
- Modélisation : syntaxe et sémantique
  - **langages de spécification, méthodes formelles**

L’Information

- **L’information** : ensemble de **renseignement**s compréhensibles par l’esprit humain, **permettant** de construire, de reconstruire ou d’enrichir **une connaissance sur un sujet**.
  - **le support des connaissances dans tous les domaines :**
    - **scientifique, technique, économique, social**

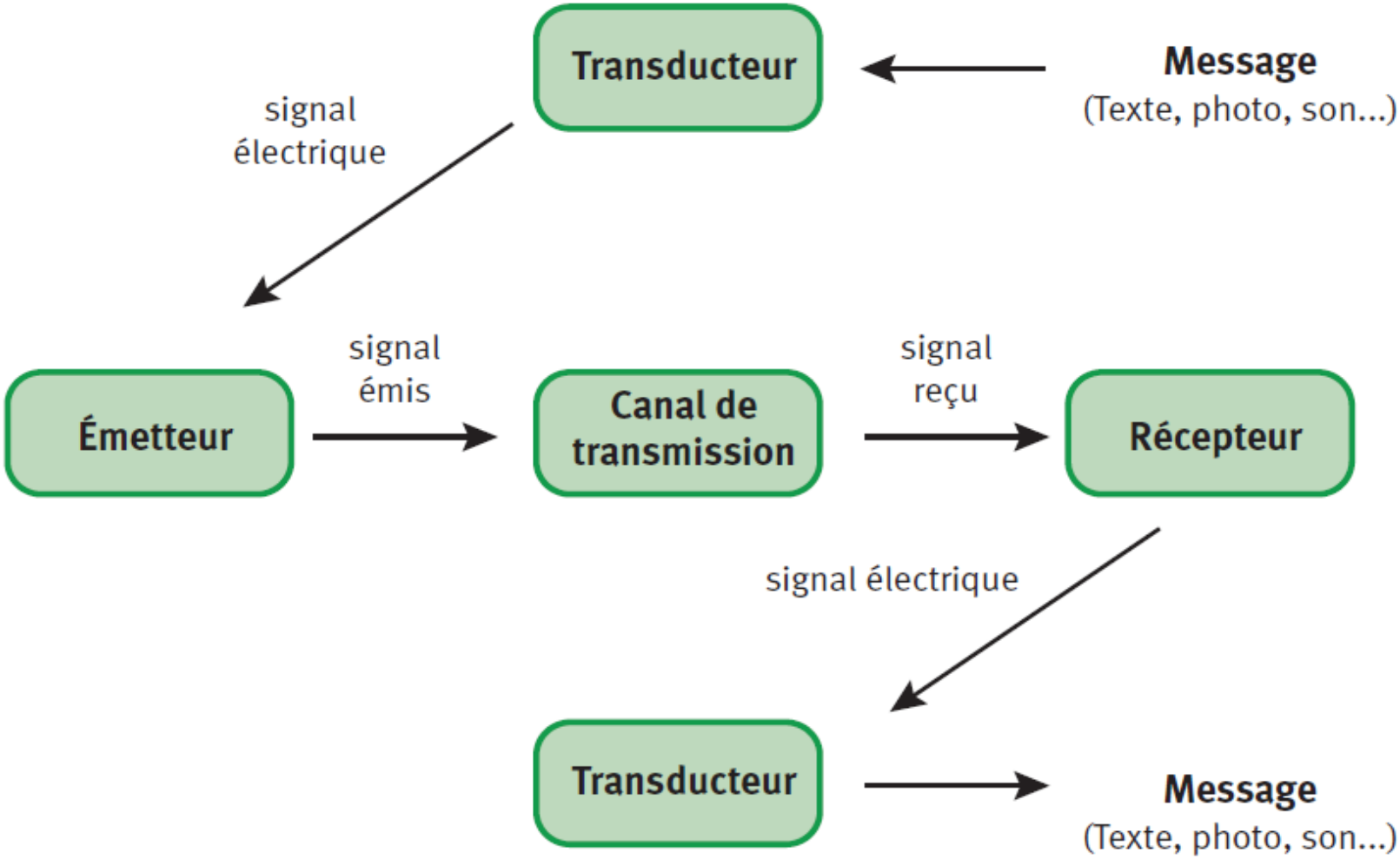
- **L’information « cinétique », l'information en action dans les interactions.**
  - le professeur qui enseigne aux élèves, le téléchargement d'un fichier, l'ordre d'un chef, l'appel d'un service web, …

- **L’information « potentielle » prisonnière de la matière**
  - le contenu d'un disque dur, l'assemblage complexe d'une machine, la structure du cristal ou de l'ADN, la mémoire biologique, l'essence d'une œuvre d'art, …

L'Information

- **Les informations traitées sont de différentes natures** :
  - **textes :** manuscrits, lettres, livres, articles, rapports, pages du web
  - **images** : photographies, cartes, graphiques
  - **documents multimédias** : vidéos , interviews, discours, reportages, émissions, conférences
  - **objets 3D**
  - **documents composites** : textes illustrés, sites webs, diaporamas
  - **programmes**
  - **nombres**
  - etc…

Codage et transmission de l'information



Codage de l’information

- La correspondance qui permet sans ambiguïté de passer d’une représentation (dite externe) d’une information à une autre représentation (dite interne) de la même information, suivant un ensemble de règles précises **(transducteur)**

- **Le codage (nommé base binaire) en trois étapes**
  - **L’information est exprimée par une suite de nombres (numérisation)**
  - **Chaque nombre est codé sous forme binaire** (suite de 0 et 1)
    - **Le bit** « binary digit », c'est-à-dire 0 ou 1 est la plus petite unité d'information manipulable par une machine numérique.
  - **Chaque élément binaire est représenté par un état physique**

Codage binaire de l'information

Pourquoi la base binaire ?

- représentation simple par un signal électrique, magnétique ou lumineux (charge +/-, spin up/down, on/off)
- seule base à unifier les nombres et la logique **(Shannon 1930)**
- **0 / 1 = la dichotomie, centrale en algorithmique**

L'information est manipulée en regroupement de 8 bits appelé Octet

un octet permet de stocker un caractère (lettre ou chiffre)

un mot = 16 bits

un double mot = 32 bits

kilooctet (ko ou KB)=1024 octets ; Mégaoctet (Mo ou MB) 1024 ko = 1 048 576 octets

Gigaoctet (Go ou GB) 1024 Mo = 1 073 741 824 octets

Téraoctet (To) 1024 Go = 1 099 511 627 776 octets

Tout est codé en binaire

- **213 représente un nombre en base 10**
  - En réalisant des divisions successives par 10, on obtient :
  - $213(10) = 3 \times 10^0 + 1 \times 10^1 + 2 \times 10^2$
  - **En base 2, il n'y a que deux chiffres : 0 et 1.**

11010101 $= 1 \times 2^0 + 0 \times 2^1 + 1 \times 2^2 + 0 \times 2^3 + 1 \times 2^4 + 0 \times 2^5 + 1 \times 2^6 + 1 \times 2^7$ = **213 (10)**

- **Texte : le codage ASCII**
  - **on associe une valeur binaire à chaque lettre** 256 possibilités sur un octet
    - **exemple :** A = 01000001 ; B =01000010
  - table universelle « utf-8 » (au plus 32 bits pour un caractère ou idéogramme)
- **Image : le codage RVB**
  - **décomposition de l' image en points : les pixels (critère de qualité)**
  - **une couleur à chaque pixel sur 3 octets :** (**Rouge**, **Vert**, **Bleu**)
    - l'intensité de chaque composante par un nombre entre 0 et 255
    - Exemple : (192, 64, 255) est un pixel violet
- **Son : la numérisation par échantillonnage**

La compression

- **Compression obligatoire pour la distribution**
  - **La taille d'une vidéo numérisée image par image est énorme**
    - méthode de compression = CODEC
      - La vidéo compressée enregistrée dans un fichier : mp4, wmv, .mov
      - mp3 (format audio)
  - **Compression d'archivage** (plusieurs fichiers dans un seul fichier)
  - **TVHD : réseaux haute vitesse à large bande**
    - **5 mégabits par seconde (image 1000 x 1000 pixels : 24 mégabits)**

Transport de l'information

- **Notion centrale de protocole**
  - un ensemble de règles et de procédures permettant la communication entre processus (s'exécutant éventuellement sur différentes machines)

- **Réseau téléphonique commuté** (RTC)
  - **continuité de la ligne de communication** (point à point)
    - **réservation d'un canal de communication pour toute la durée de la communication entre les parties reliées (adapté à la voix/téléphone fixe)**

- **Internet : réseau de réseaux (la suite TCP/IP)**
  - **multiplexage/démultiplexage**
    - **transit sur une même ligne de données provenant d'applications diverses et réparties**
  - **routage par sauts successifs (next-hop routing)**
    - **besoin de ne connaitre que l'adresse IP du prochain routeur directement accessible (et celle du destintaire)**

Internet facilite tout

- **des milliards de milliards de connexions** (mobiles ou M2M)
    - moles (6.023.1023) de bits
- **des services innombrables** (gain de la numérisation)
    - Intégration image, parole, texte et musique ,..
- **accès multiforme**
    - ordinateur, téléphone, PDA, TV,…
- **généralisation du haut débit**
    - autoroutes optiques partout
- **l'arrivée des objets...**
    - adresse IPV6 : 128 bits , une par objet

Le principe de robustesse de la suite TCP/IP (Postel 1981)

Interopérabilité plus importante que le respect de la norme (OSI)

Le prix Jon Postel attribué depuis 1999 par l'Internet Society

Aspects scientifiques liés au codage et transport

- cryptologie, sécurité (IPSEC) : **protection**
- théorie de l'information: **transmission**
- vérification formelle : protocoles **liaison, réseau**
- graphes, ordonnancement: **routage**
- algorithmes distribués: **contrôle de flux**
- statistiques & probabilités: **dimensionnement**
- systèmes dynamiques: **diffusion**

Détection et correction d'erreurs lors de la transmission

- **Faire face à l'instabilité du canal radio :** les évanouissements, les bruits ou les interférences.
    - fiabilité des réseaux mobiles

- **Réduire la probabilité d'erreur par bit :**
  - techniques de codage pour la détection et correction d'erreurs (codes en blocs ou codes convolutifs)
    - ajout de bits de redondance dans le message à émettre
    - vérification de la loi de codage à la réception
  - protocoles coopératifs pour réseaux sans fil (diversité spatiale)
    - technique MIMO, communication coopérative

Côut: accroissement du débit numérique

nécessité du réseau haut débit

Algorithmique

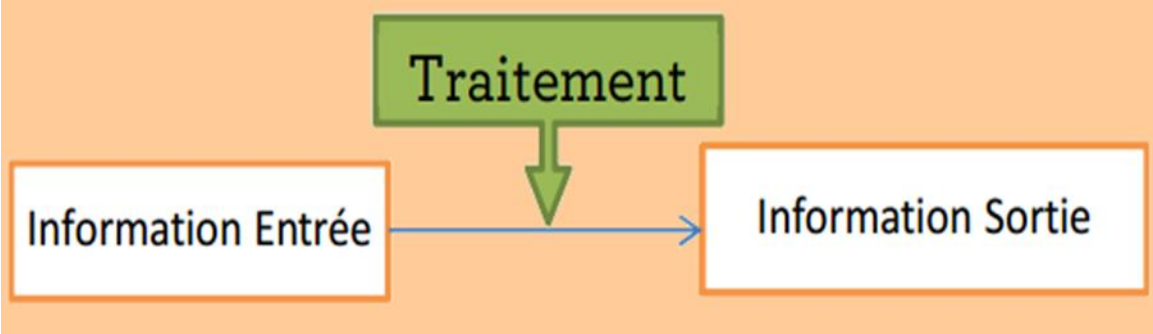
Al Khawarizmi
~ 783 – 850

Algorithme

Art d'organiser un calcul complexe en partant d'opérations simples

Algorithmique

- Le traitement de l'information ou ***ALGORITHME***
  - *Une suite d'opérations* ***transformant une représentation de cette information en une autre représentation plus facile à manipuler ou à interpréter*** *suivant un ensemble de règles fini*



- Algorithme : squelette abstrait d'un programme informatique
  - *Traitement séquentiel : systèmes fonctionnels*
  - *Traitement concurrent : systèmes réactifs*

Exemples de traitement

- 1. **Décider : mot →vrai / faux.**
  - reconnaissance de motifs (traitement de texte) ;
  - fouille de données ;
  - bio-informatique, génomique (ressemblance entre deux segments d'ADN).

- 2. **Calculer : mot →mot'**
  - production de code,
  - compilation (transformer un programme C en langage machine) ;
  - compression /décompressions de données (zip, unzip).

- 3. **Organiser / contrôler :**
  - vérification automatique ;
  - supervision d'ateliers.

Le théorème d'incomplétude de Gödel (1931)

- **Peut-on décider mécaniquement si un énoncé est démontrable ou non ?**
  « <u>préciser "mécaniquement", c'est formaliser la notion d'algorithme</u> »

- **Dixième problème de Hilbert :** Peut-on décider si un polynôme $P \in Z[X_1, \cdots, X_n]$ à coefficients entiers possède une racine entière (équation diophantienne)?
  - **Il n'existe aucun algorithme général décidant l'existence de solutions pour les équations diophantiennes.**

- **Gödel:** « Un système d'axiomes cohérent et suffisamment expressif est susceptible de générer des énoncés dont la validité ne peut être démontrée dans le cadre des règles mêmes qui gouvernent la formulation de ces énoncés et leurs déductions.»

- **L'échec du programme de Hilbert qui a voulu prouver la cohérence de l'arithmétique.**
- **Constat dramatique pour la preuve automatique de théorèmes** (Halting problem)

Les modèles de calcul

- **Diverses formalisations mathématiques de ce que peut être une méthode de calcul :**
  - Fonctions récursives, Kurt Gödel, 1931-34.
  - λ-calcul, Alonzo Church, 1936.
  - **Machines de Turing, Alan Turing, 1936.**
  - Systèmes de Post,
  - Machines RAM, RISC
  - Programmes JAVA, C, CAML,PYTHON, . . .

- **Il s'avère que ces modèles, très différents, ont exactement la même puissance !**

La machine de Turing (1936)

- **Une machine de Turing est un 8-uplet $M = (Q, \Sigma, \Gamma, B, \delta, q_0, qa, qr)$**

où :

1. **Q** est l'ensemble fini des états ;
2. **Σ** est un alphabet fini ;
3. **Γ** est l'alphabet de travail fini : $\Sigma \subset \Gamma$;
4. **$B \in \Gamma$** est le caractère blanc ;
5. **$q_0 \in Q$** est l'état initial ;
6. **$qa \in Q$ est** l'état d'acceptation ;
7. **$qr \in Q$** est l'état de refus (ou d'arrêt) ;
8. **δ** est la fonction de transition : δ est une fonction (possiblement partielle) de $Q \times \Gamma$ dans $Q \times \Gamma \times \{\leftarrow, |, \rightarrow\}$.
   - le symbole ← est utilisé pour signifier un déplacement vers la gauche
     → un déplacement vers la droite, et | aucun déplacement.

La machine de Turing (1936)

- **Restriction à un alphabet binaire**
  - Toute machine de Turing qui travaille sur un alphabet Σ quelconque peut être simulée par une machine de Turing qui travaille sur un alphabet Σ avec uniquement deux lettres, avec $\Gamma = \Sigma \cup \{B\}$.

- **Le modèle de la machine de Turing**
  - peut paraître extrêmement rudimentaire.
  - **extrêmement puissant, capable de capturer la notion de calculable en informatique.**
    - **un modèle abstrait du fonctionnement des appareils mécaniques de calcul, tel un ordinateur et sa mémoire**

La thèse de Church-Turing (1936)

- **Classe des fonctions calculables :**
  - une fonction est dite **calculable** s'il existe une méthode précise **(algorithme** ou procédure mécanique) qui, étant donné un argument, permet d'obtenir l'image en un nombre fini d'étapes.
    - **d*écidabilité algorithmique***

Thèse de Church-Turing :

une fonction est calculable si et seulement si elle est calculable par machine de Turing.

Fonctions incalculables

- Théorème de Cantor (1891) :
  - **les fonctions (partielles ou pas) sur un domaine infini ne sont pas dénombrables** (argument de la diagonale)
- Un algorithme peut toujours être représenté par un mot fini sur un alphabet fini :
  - **il y a seulement un nombre dénombrable de fonctions calculables**.

$\Rightarrow$ **Il existe des fonctions f qui sont incalculables**, c'est-à-dire dont, étant donné x, la valeur f(x) ne peut être calculée en un temps fini par aucun algorithme que l'on aurait associé à f.

$\Rightarrow$ ***indécidabilité algorithmique***

Le Problème de l'arrêt

- **On connaît de nombreux exemples explicites de fonctions incalculables (problèmes indécidables)**

- **Le dixième problème de Hilbert**

- **Le plus courant est celui du problème de l'arrêt (Halting problem):**
  - Il n'existe pas de programme informatique qui prend comme entrée une description d'un programme informatique et une entrée et qui, grâce à la seule analyse de ce code — bien que celui-ci soit de taille finie — répond VRAI si le programme s'arrête sur son entrée et FAUX sinon.

Attention : dire qu'un problème est indécidable ne veut pas dire que la question posée n'a pas et n'aura jamais de solution mais **seulement qu'il n'existe pas de méthode , applicable d'une façon mécanique, pour répondre à la question.**

La théorie algorithmique de l'information

- **Théorie initiée par Kolmogorov et Chaïtin dans les années 1960**
  - **vise à quantifier et qualifier le contenu en information d'un ensemble de données**
    - en utilisant la théorie de la calculabilité et la notion de machine universelle de Turing.
  - **fondements différents de ceux de la théorie de l'information de Shannon**
    - qui 'intéresse aux aspects communicationnels des messages émis et laissant de côté leur contenu sémantique et leur contenant physique

La théorie algorithmique de l'information

binary program → Computer → binary output

Quel est le plus court programme qui produit une sortie donnée ?

Si le programme est concis mais pas la sortie, on a une théorie

Si la sortie est aléatoire alors il n’y a pas de compression possible et l’entrée est de même taille que la sortie

La notion de complexité

- **La complexité (le contenu en information) d'un objet (une suite de caractères) est définie par la longueur ou le temps de calcul de sa plus courte description algorithmique possible.**

- Exemple: Soient les 2 mots suivants formés de 32 caractères :
  - **abababababababababababababababab**
  - **4c1j5b2p0cv4w1x8rx2y39umgw5q85s7**
  - Le premier a la description courte suivante : "ab 16 fois", formée de 10 caractères.
  - Le second n’a pas de description simple évidente autre que le mot lui même avec 32 caractères.

De la décidabilité à la complexité

- La calculabilité : la question de l'existence (ou de la non-existence) de solutions algorithmiques à des problèmes donnés.

- **La complexité** : La question de l'existence (ou de la non-existence) d'algorithmes efficaces à des problèmes décidables donnés.
  - **Aspect essentiel en pratique**: temps ou mémoire nécessaire sur la machine pour l'exécuter.
  - Indépendante du langage de programmation utilisé (équivalence avec la machine de Turing)

La complexité d'un algorithme

- On associe à l'algorithme A une entrée d

- **On mesure l'efficacité**, c'est-à-dire **la complexité** d'un algorithme en terme d'une mesure élémentaire à valeur entière :
  - le nombre d'instructions effectuées,
  - la taille de la mémoire utilisée,
  - le nombre de comparaisons effectuées,
  - ou toute autre mesure.

- **Complexité au pire des cas :** la valeur de cette mesure dans le pire des cas (des entrées)

La complexité d'un problème

C'est la complexité du meilleur algorithme qui résout un problème donné **i.e. celle de l'algorithme ayant la plus faible complexité (le plus efficace)**

On s'intéresse à l'ordre de grandeur (l'asymptotique) des complexités quand la taille n des entrées devient très grande :

f(n) = O(g(n)) où g(n) est linéaire, n logn, quadratique, cubique, non-polynomiale

Complexité en temps

Temps de calcul nécessaire à l'algorithme pour s'exécuter sur un processeur capable d'exécuter un million d'instructions élémentaires par seconde.

Tableau repris de Kleinberg et Tardos (2005) (∞ signifie un temps > 10^{25} années)

Complexité	n	$n \log_2 n$	n^2	n^3	1.5^n	2^n	$n!$
$n = 10$	< 1 s	< 1 s	< 1 s	< 1 s	< 1 s	< 1 s	4 s
$n = 30$	< 1 s	< 1 s	< 1 s	< 1 s	< 1 s	18 min	10^{25} ans
$n = 50$	< 1 s	< 1 s	< 1 s	< 1 s	11 min	36 ans	∞
$n = 100$	< 1 s	< 1 s	< 1 s	1s	12,9 ans	10^{17} ans	∞
$n = 1000$	< 1 s	< 1 s	1s	18 min	∞	∞	∞
$n = 10000$	< 1 s	< 1 s	2 min	12 jours	∞	∞	∞
$n = 100000$	< 1 s	2 s	3 heures	32 ans	∞	∞	∞
$n = 1000000$	1s	20s	12 jours	31,710 ans	∞	∞	∞

Algorithmes en temps raisonnable

- **Un algorithme est efficace** si sa complexité en temps est polynomiale **$O(n^k)$**pour un entier **k** : **raisonnable en théorie de la complexité**

- **Classe P** : la classe des problèmes décidés par une machine de Turing avec un algorithme en temps polynomial

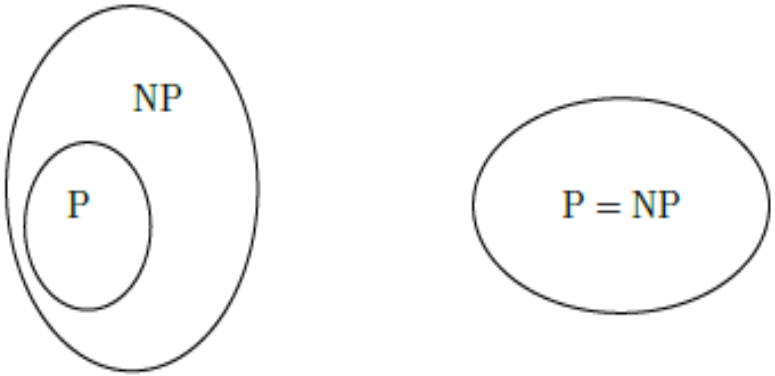
- **Classe NP** : la classe des problèmes décidés par une machine de Turing non déterministe avec un algorithme polynomial

  (i.e. qui possèdent un vérificateur polynomial)

- **$P \subseteq NP$**

La question P = NP ?

Clairement, on a soit P = NP, soit P $\subset$ NP

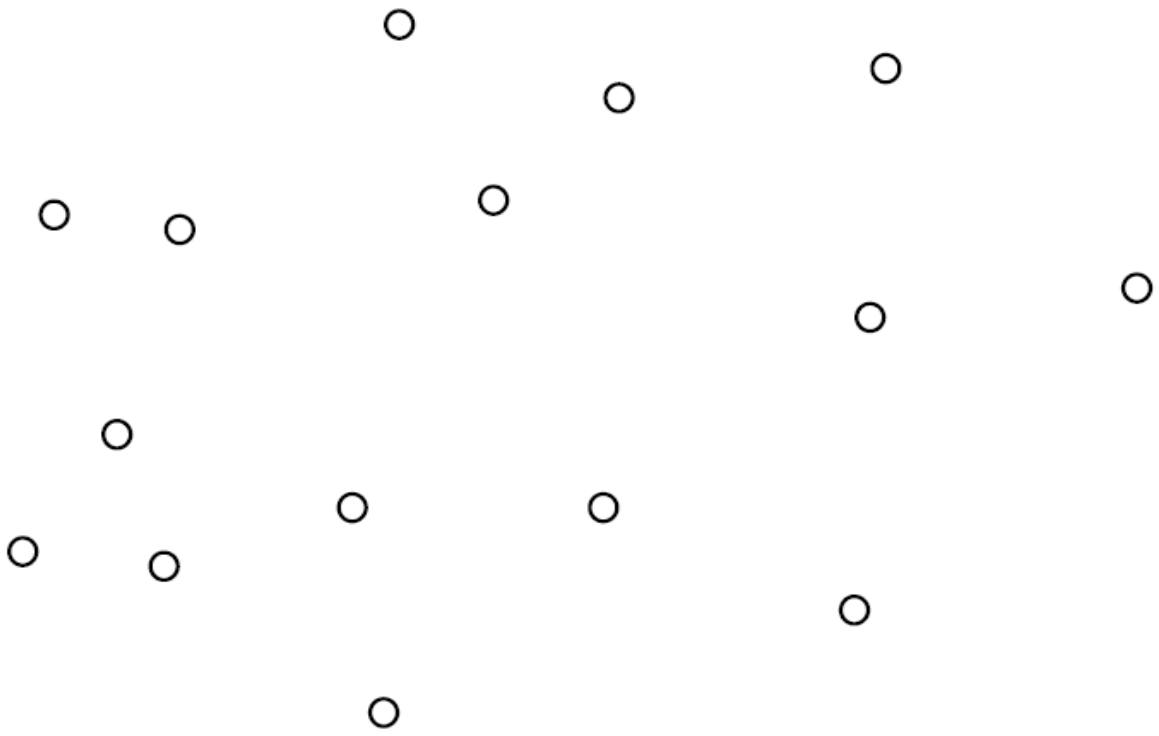


Clay Mathematics Institute offre **1 000 000** de dollars à qui déterminerait la réponse à cette question !

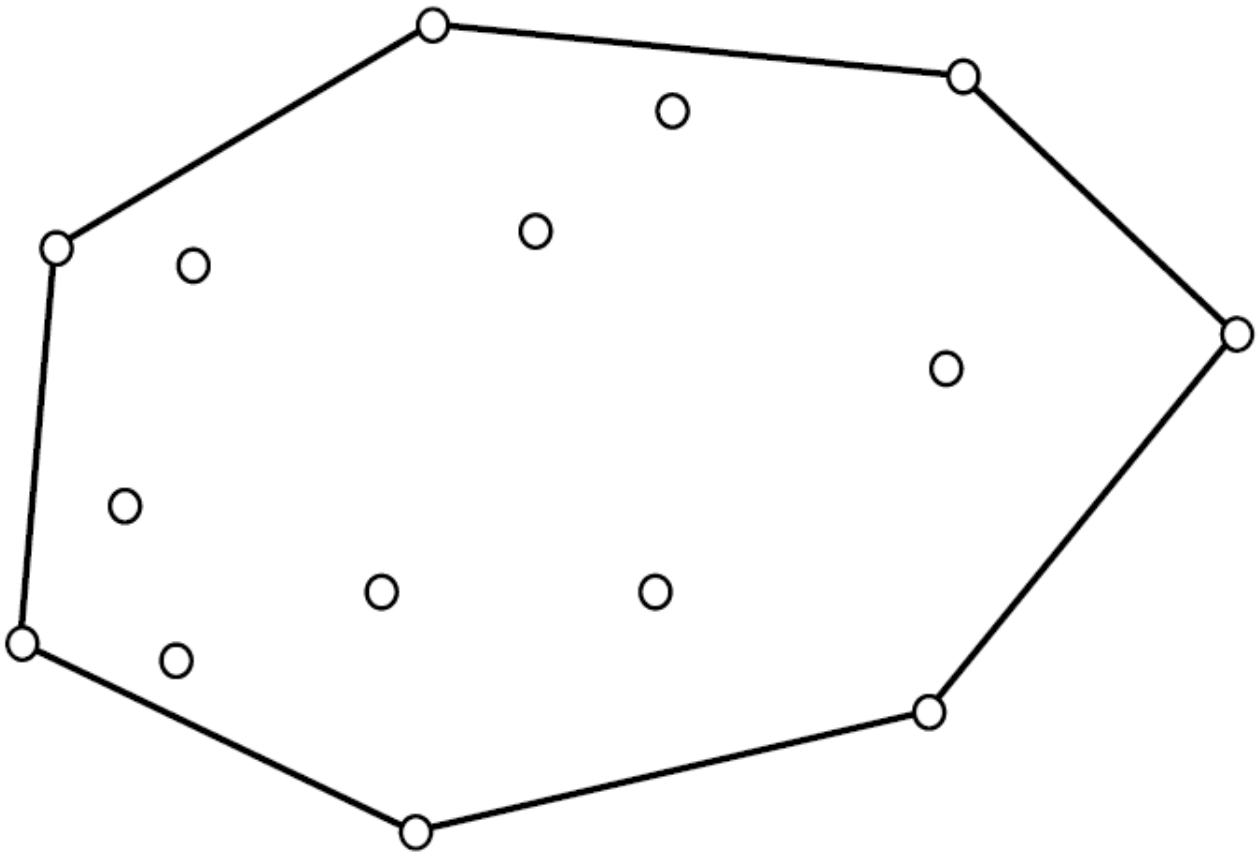
enjeu économique crucial sur les systèmes de cryptographie actuels

Algorithme de Graham

- **Calcul de l'enveloppe convexe d'un ensemble de points**

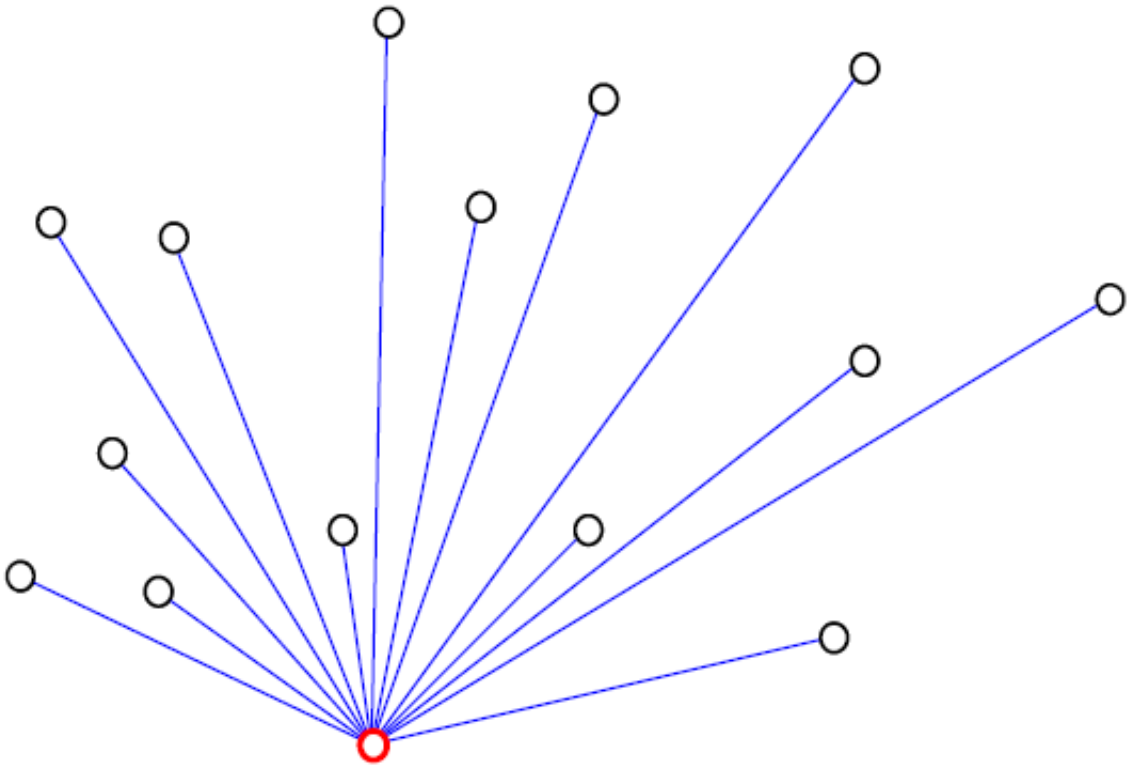


Algorithme



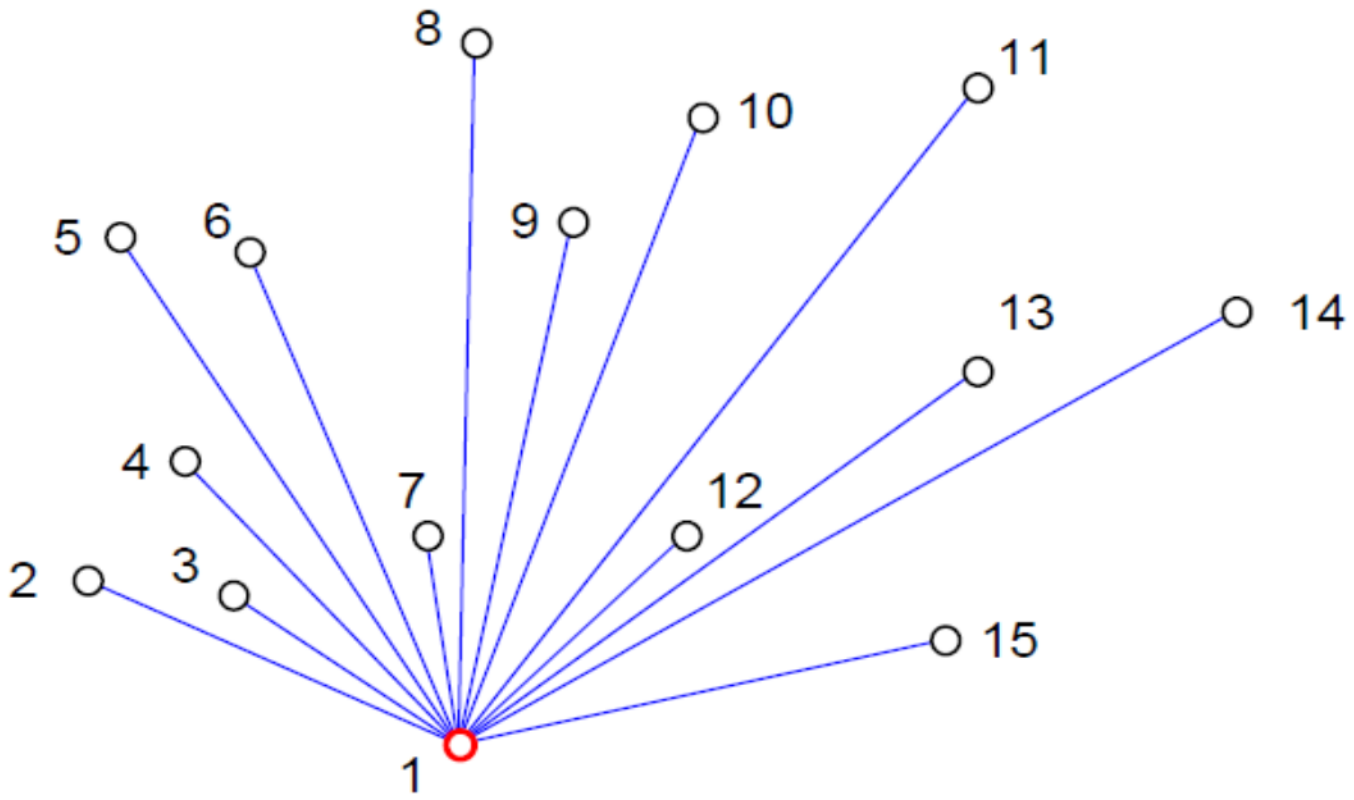
- Un segment est dans l'enveloppe convexe si et seulement si tous les autres points sont du même côté

Point le plus bas, calcul des angles



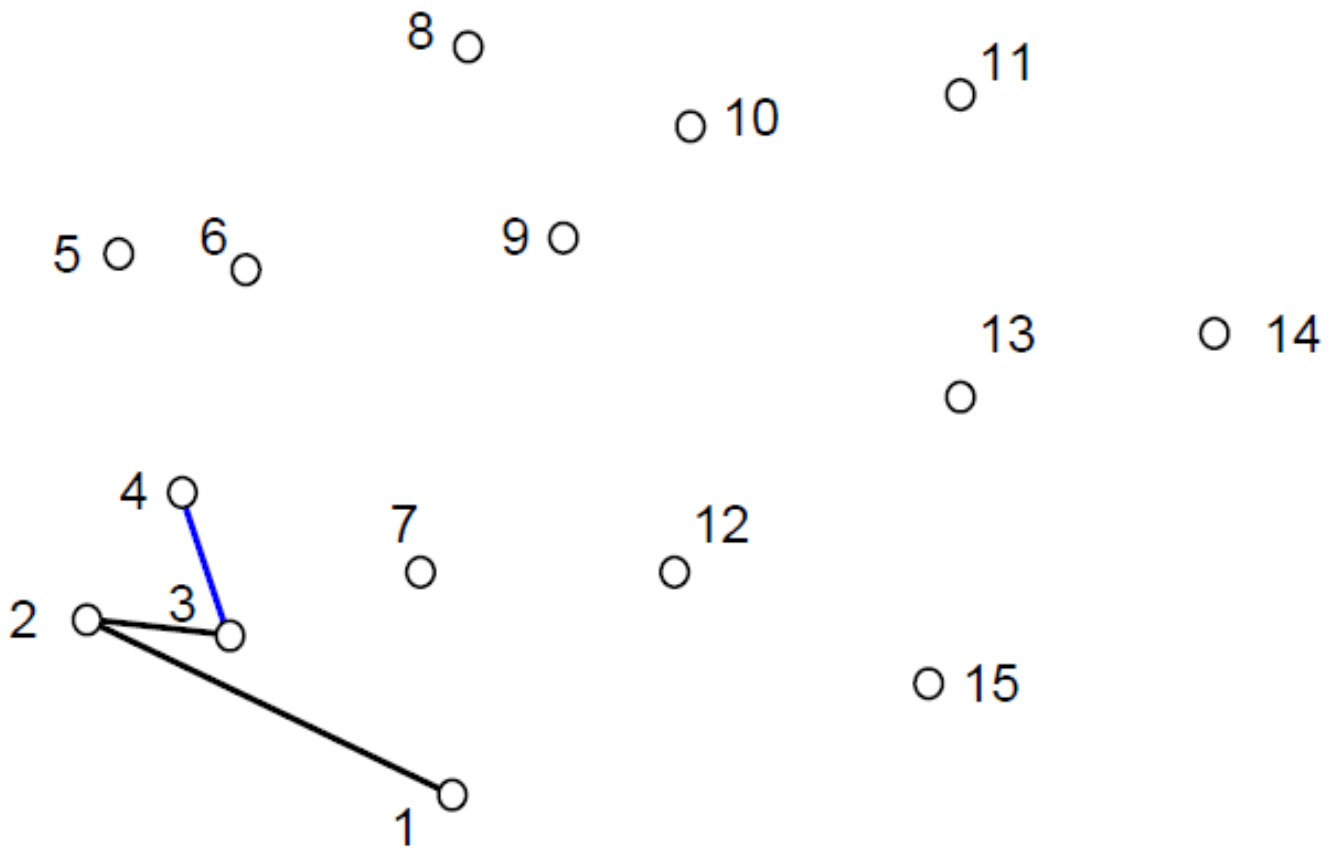
Coût : O(n)

Tri des angles, numérotation

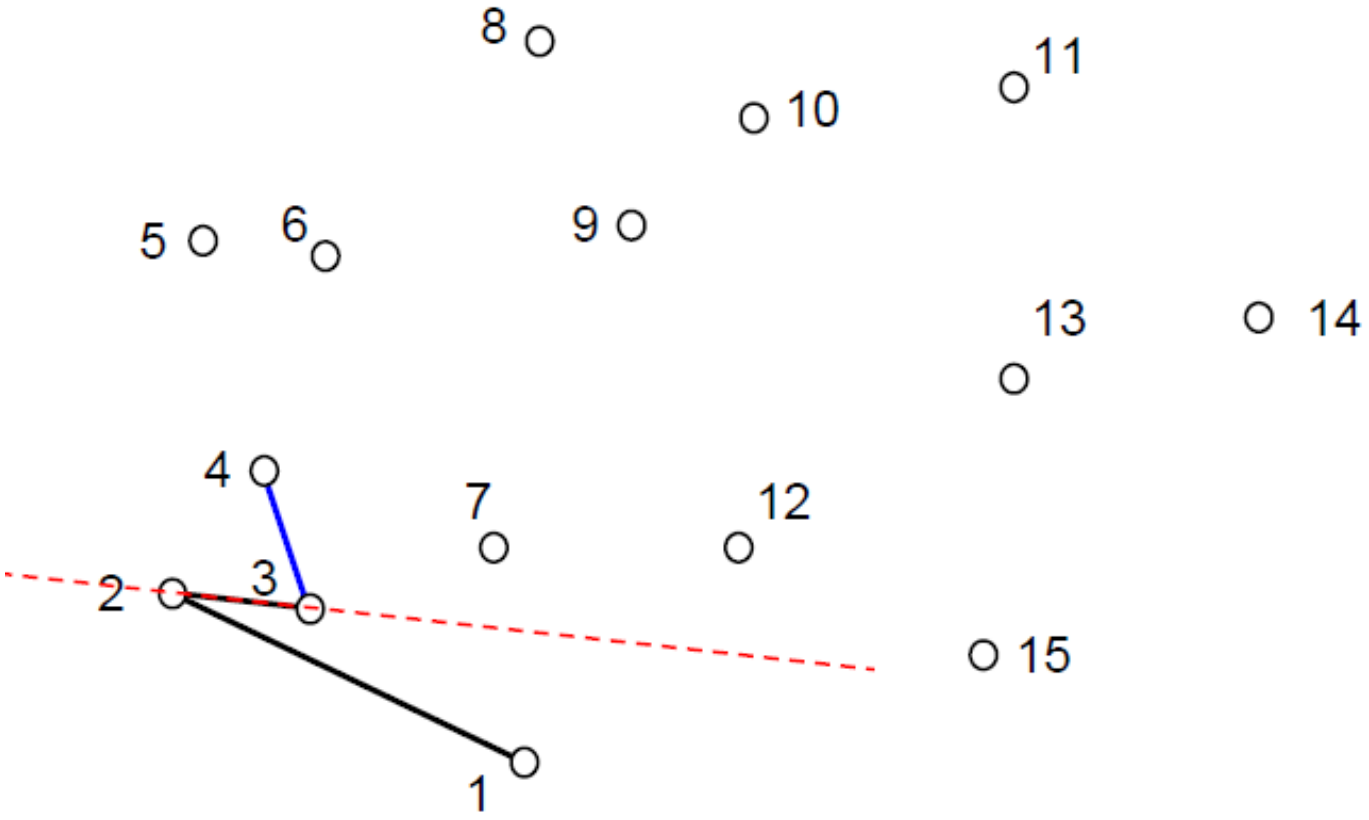


Coût : O(n logn)

Construction de l'enveloppe

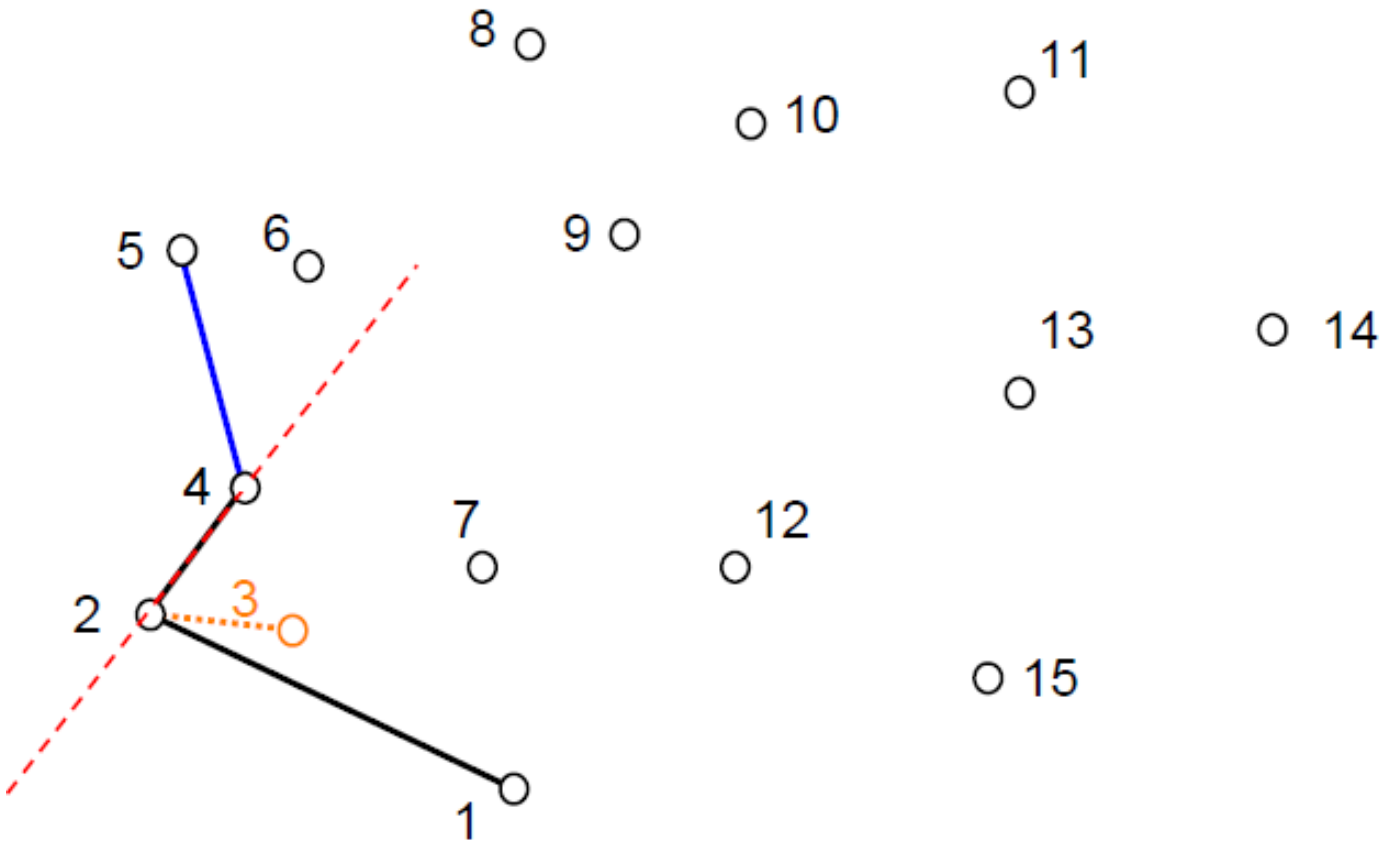


Ajout de 4



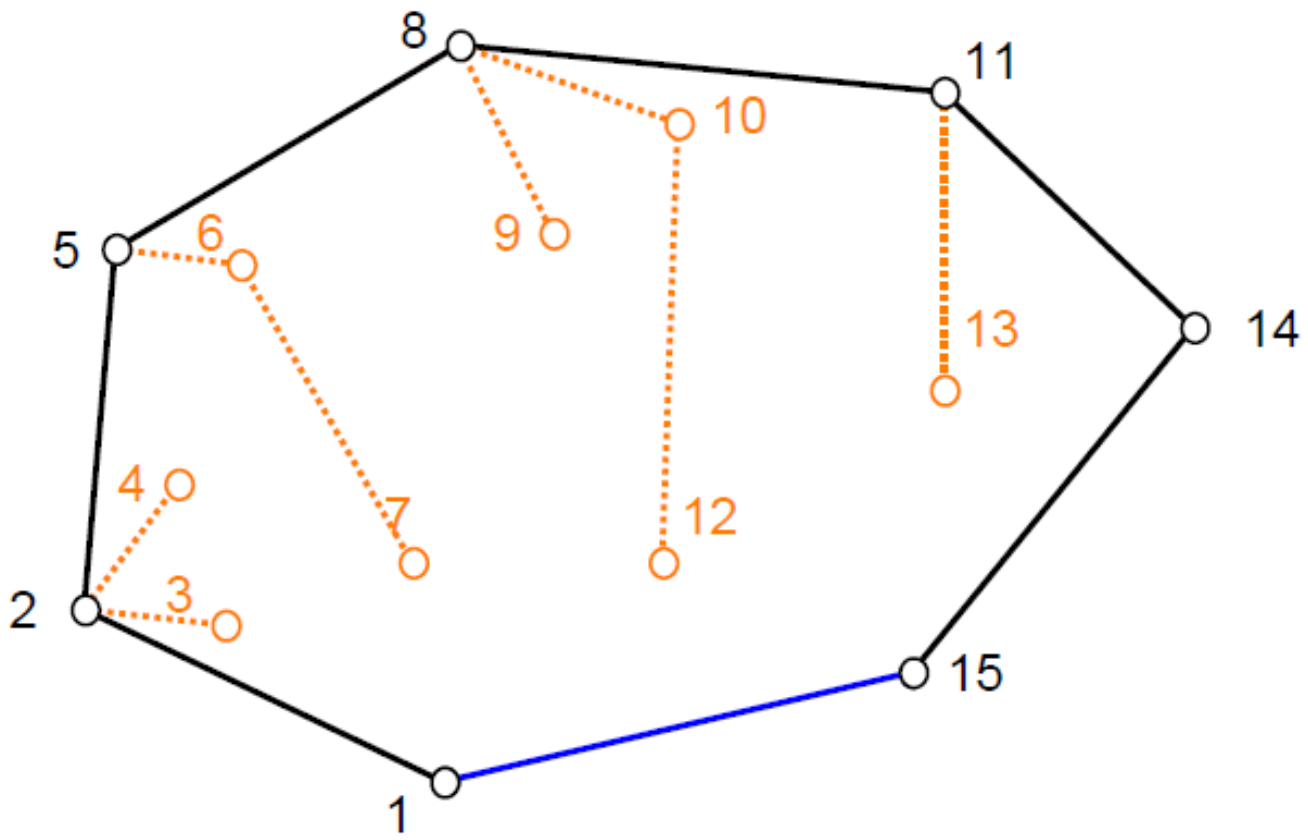
Suppression de 3 car 1 et 4 de chaque côté de 3-2

Ajout de 5



Suppression de 4 car 1 et 5 de chaque côté de 4-2

La solution



Coût de la construction : **O(n)**

car chaque point est introduit et éliminé au plus une fois

Coût total : 0(n logn)

Le problème SAT

- **Une formule propositionnelle est-elle toujours vraie? peut-elle être rendue vraie?**

$$(A \Rightarrow (B \Rightarrow C)) \Rightarrow ((A \Rightarrow B) \Rightarrow (A \Rightarrow C)) \quad \text{toujours}$$

$$(A \Leftrightarrow B) \wedge (C \Leftrightarrow D) \quad \text{parfois}$$

$$(A \Rightarrow B) \Rightarrow (A \wedge \neg B)) \quad \text{jamais}$$

- Applications industrielles et scientifiques:
  - CAO : routage et vérification de circuits,
  - problèmes emplois du temps, tournées, planification de tâches
  - Etude de réactions biochimiques
  - Mais sur beaucoup de variables : 50000; 1000000 et plus!

Interprétation booléenne

- voir les connecteurs logiques comme des opérations : faux = 0 vrai = 1

0	1
1	0

$\wedge$	0	1
0	0	0
1	0	1

$\vee$	0	1
0	0	1
1	1	1

$\Rightarrow$	0	1
0	1	1
1	0	1

$\Leftrightarrow$	0	1
0	1	0
1	0	1

- méthode naïve : essayer toutes les valeurs de vérité

$A=0 \quad B=0 \quad C=1 \quad D=1 : (A \Leftrightarrow B) \wedge (C \Leftrightarrow D) = 1$

$A=0 \quad B=1 \quad C=0 \quad D=1 : (A \Leftrightarrow B) \wedge (C \Leftrightarrow D) = 0$

temps d'éxécution exponentiel en nombre de variables

SAT est un problème NP-Complet

- **on ne dispose pas d'algorithme polynomial**
  - difficile de trouver une bonne solution parmi les choix en nombre exponentiel

- **en revanche, il est facile (polynomial) de vérifier qu'une solution proposée en est bien une.**

- **NP-Complet : parmi les plus difficiles dans la classe NP**

- **Progrès dans le structures de données :** BDD , DPLL, ATPG,..

Aspects algorithmiques des systèmes répartis

- protocoles de communication
- gestion de mémoire partagée multiprocesseurs
- gestion de cache / pipeline de microprocesseurs
- bases de données réparties
- gestion de trafic aérien ou automobile
- …

Caractéristiques des systèmes répartis

- **Absence d'un état global**

- **Pas de « temps absolu »** :
  - évènements perçus par un processus :
    - ceux dont il est le siège
    - ceux issus des relations de causalité entre l'émission et la réception d'un même message

⇒ **Impossible d'ordonner deux événements quelconques**

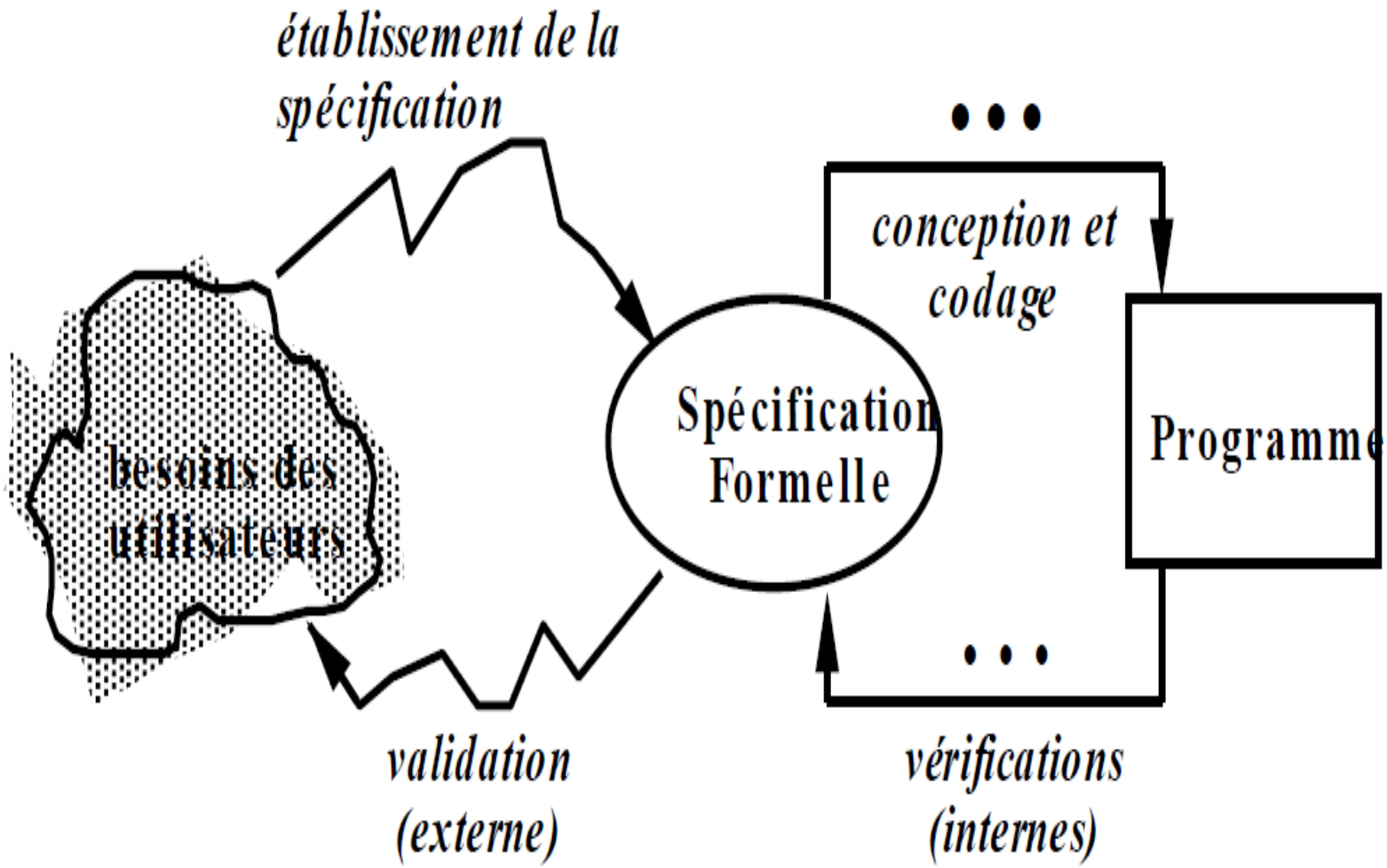
Caractéristiques des systèmes répartis

- **Distribution des données**
  - **Duplication des données :**
    - La donnée x est recopiée en n exemplaires x1,…, xn sur n sites
    - **Assurer cohérence mutuelle des copies** : x1 =…= xn = x
  - **Partitionnement des données sur différents sites**
- **Distribution du contrôle**
  - **Pas de relation hiérarchique entre les processus**
    - privilège tournant : « maître » temporaire
    - résistance aux pannes

Concepts et techniques de l'algorithmique distribuée

- synchronisation de processus
- allocation de ressources
- terminaison
- exclusion mutuelle
- interblocage
- raisonnement par induction

Méthode de développement de logiciel



Méthode de développement de logiciel

- Consensus sur la manière d'utiliser l'informatique pour résoudre un problème : **"spécifier, puis valider, enfin implémenter"**

- Conceptions/Raffinements/Implémentations : $\mathbf{SP_0 \rightarrow SP_1 \rightarrow \cdots \rightarrow SP_n}$

- Codage/Programmation : $\mathbf{SP_n \rightarrow Prog}$

- Vérifications : $\mathbf{SP_0 \leftarrow SP_1 \leftarrow \cdots \leftarrow SP_n \leftarrow Prog}$

Faire de sorte que chaque étape préserve les propriétés requises par la précédente

SP_i est un raffinement correct (implémentation) de SP_{i-1}

référence de correction : la spécification de départ

Besoin de méthodes formelles

- **La vérification de propriétés comportementales nécessite une analyse globale — statique et dynamique — du système.**

- **Besoin crucial pour les systèmes critiques** :
  - les *systèmes de transport* : logiciels embarqués pour l'avionique, trains, automobiles ;
  - la *production d'énergie* : dispositifs de surveillance des centrales nucléaires ;
  - la *santé* : chaînes de production de médicaments, appareil médicaux ;
  - le *système financier* : paiement électronique;

- **Exigence de preuve formelle de l'absence de bug dans les logiciels**
  - **Test prouve la présence d'erreurs, pas leur absence**

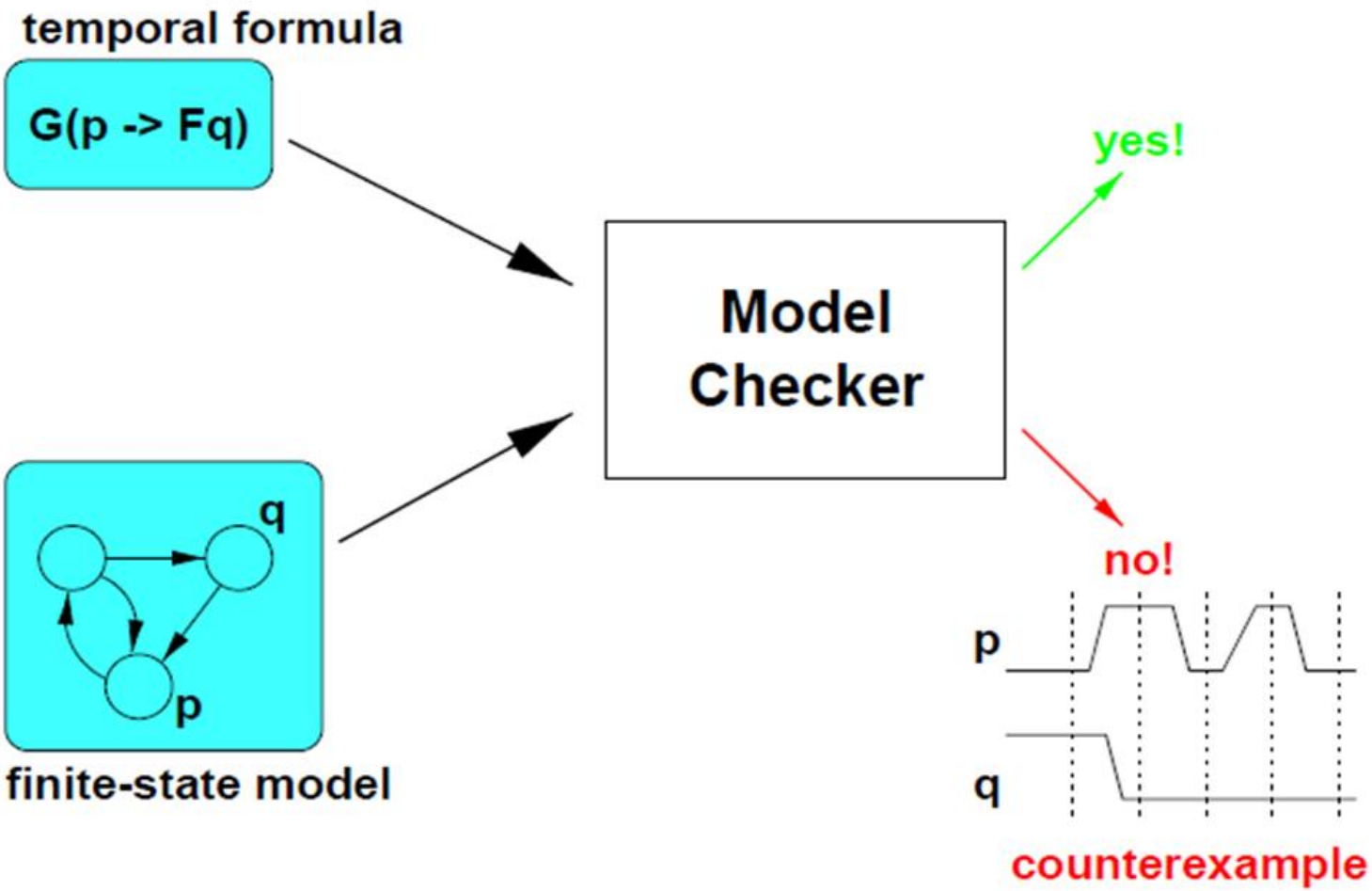
- Construction d'un **modèle formel** du système matériel/logiciel sur lequel on peut **démontrer,** par une **technique formelle, la satisfaction d'une propriété (spécification) du système.**
  - **propriétés de sûreté ou de vivacité**

2 grandes catégories de méthodes formelles

- **La preuve automatique de théorèmes**
  - Recherche automatique de preuve de propriétés partant d'une description formelle du système : un ensemble d'axiomes et un ensemble de règles d'inférences
    - **problème non décidable en général** (théorème d'incomplétude)
    - si décidable : application semi-automatique d'algorithmes couteux en temps et ressources

- **La vérification de modèles (Model-checking)**
  - **Vérification automatique** des propriétés (formulées en logique temporelle) par une énumération **exhaustive** des états accessibles du modèle formel (**automate, STE, RdP)**
    - limitation de l'explosion combinatoire de l'espace d'état
    - techniques d'ordre partiel améliorent grandement l'efficacité
    - Succès industriels - Prix Turing 2007 pour les pionniers dans ces travaux
      - Clarke, Emerson et Sifakis

Model-checker



Les langages de programmation

- Un circuit ne fait que des choses très simples
  - Il en fait des milliards par seconde, et sans erreur

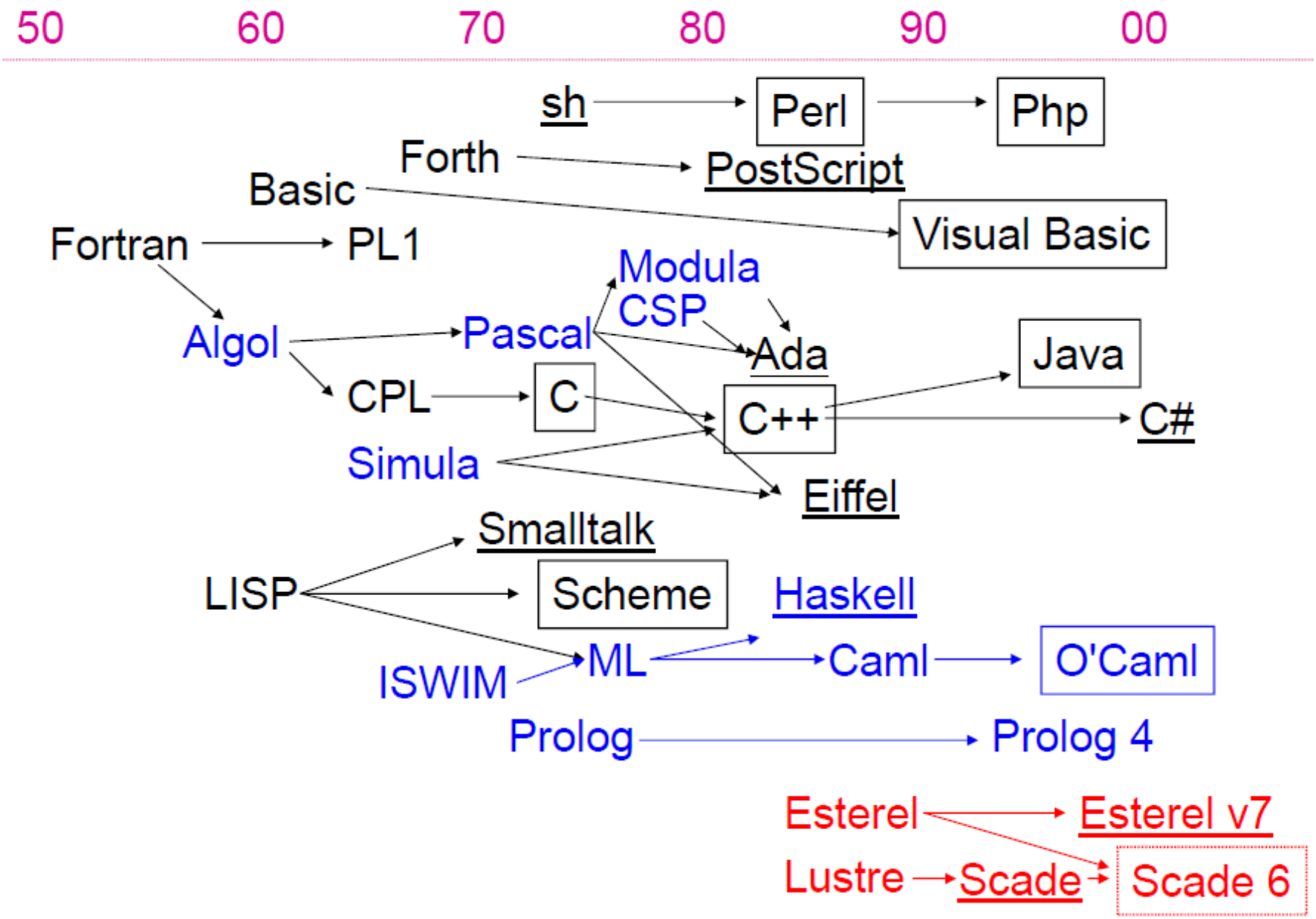
- **Le logiciel** : **spécifier quoi faire, dans tous les détails**
  - Très long texte dans un langage très technique
  - Circuits peu variés, logiciels très variés
  - Circuits très rigides, logiciels très souples
  - Mais de plus en plus gros (millions de lignes)
  - Et pas de loi de Moore....

  - Utiliser la machine pour programmer plus sûr

Les fonctions du langage

- **L'outil d'écriture des programmes**
  - par des hommes ou des programmes
  - pour la machine ou les autres hommes
  - à différents niveaux d'abstraction

- **Le support d'interaction avec la pensée**
  - les langages induisent des styles distincts
  - impératif, descriptif, fonctionnel, logique, temporel, etc...


50
60
70
80
90
00
sh
Perl
Php
Forth
PostScript
Basic
Visual Basic
Fortran
PL1
Modula
CSP
Algol
Pascal
Ada
Java
CPL
C
C++
C#
Simula
Eiffel
Smalltalk
LISP
Scheme
Haskell
ISWIM
ML
Caml
O'Caml
Prolog
Prolog 4
Esterel
Esterel v7
Lustre
Scade
Scade 6

Pourquoi tant de langages ?

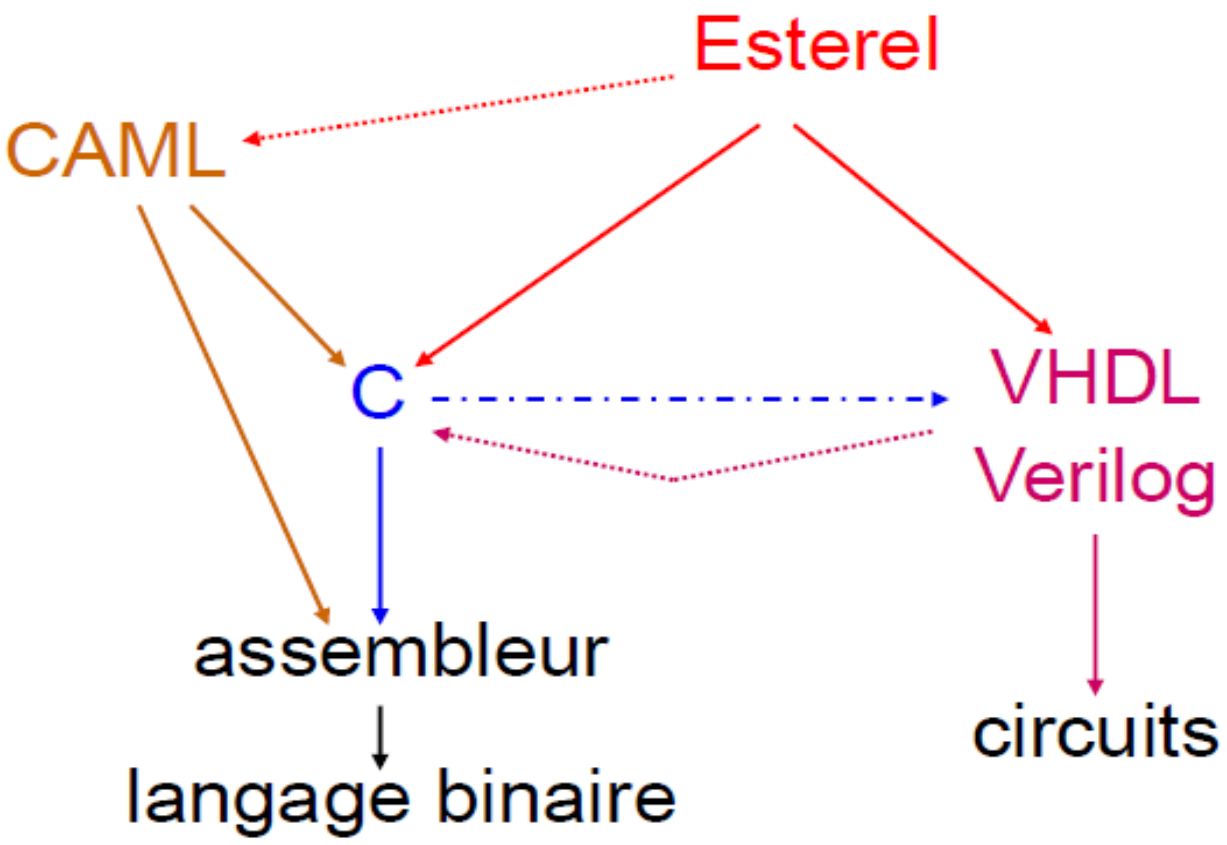
- **beaucoup d'aspects à traiter ensemble**
  - données, actions = programming in the small
  - architecture = programming in the large
  - comment réduire et détecter les bugs
- **beaucoup d'innovations successives**
  - polymorphisme, modules, objets, concurrence,...
- **énormément de compromis possibles**
  - plusieurs grandes classes et beaucoup de dialectes
- **heureusement, des théories solides**
  - automates, grammaires, lambda-calcul, logiques

Les composants d'un langage

- Des principes de calcul et d'architecture
  - calcul : **impératif, fonctionnel, logique, temporel**, etc.
  - architecture : **modularité, héritage, polymorphisme**, etc.
- Une syntaxe et un style syntaxique
  - **détermine les programmes bien construits**
- Un systèmes de types
  - **évite d'additionner des choux et des carottes**
- Une sémantique plus ou moins formelle
  - **définit le sens des programmes**
- Un outillage
  - **compilateurs, débogueurs, manuels, exemples, librairies interfaces avec d'autres langages ou systèmes (API)**
- Une communauté d'utilisateurs

Les compilateurs

Transformation du code source (haut niveau) en langage machine (bas niveau)



Une exigence fondamentale : la portabilité

exécutable indépendant de l'architecture de la machine réelle où il tourne

La naissance de l'ordinateur

- **L'Allemagne : le Plankalkul de Konrad Zuse (1941)**
  - **Machine électromécanique à relais**
- **l'Angleterre où le premier ordinateur électronique est construit (1945)**
  - **Première machine digitale à utiliser les tubes à grande échelle**
- **Les Etats-Unis où la première machine totalement électronique du type de Von Neumann (1949) est construite**
  - **Le concept de programme enregistré en mémoire**
  - **EDSAC** (Electronic Delay Storage Automatic Calculator)

Le Plankalkul de Konrad Zuse

- **Le passage des machines analogiques et des cartes perforées aux ordinateurs**

- **En 1941, construction de la machine électromécanique pour** la résolution de calculs complexes dans le domaine du génie civil
  - première machine à fonctionner avec un programme
  - utilisation de système binaire
    - passage des signaux par des portes logiques plutôt que par roues à additionner
  - unité arithmétique et logique:
    - additionner, soustraire, diviser, multiplier, extraire des racines carrées
    - conversions binaire-décimale et vice versa
  - 2600 relais électromécaniques
    - **très rapide dans le standard des machines mécaniques**
    - **seulement trois additions par seconde et une multiplication en six secondes**

La machine digitale

|

- **Le développement des technologies de l'électronique**

- En 1943 , construction en Angleterre du **Colossus :**

première machine digitale à utiliser les tubes à grande échelle pour accélérer la vitesse de calcul pour déchiffrer les messages des allemands.

- En 1945, construction de L'ENIAC :

premier ordinateur électronique à grande vitesse prévu pour la résolution de problèmes de balistique de l'armée américaine

L'ENIAC

L'ENIAC (Electronic Numerical Integrated Analyser and Computer) :

- 18000 tubes :
    - 2 fois plus que tous les calculateurs construits avant lui,
    - 1000 fois plus rapide que les calculateurs électromécaniques à relais
    - 100 μs pour faire une addition , 2.8 ms pour une multiplication.
- nécessitait une puissance de 150 kwatts
- occupait une salle de 167 mètres carrés
- commença à fonctionner en 1945 (jusqu'à 1955) pour travaux sur:
    - la météorologie, la mécanique des fluides
    - calculs sur les chaînes de réaction thermonucléaire (la bombe à hydrogène)

- **Limitations : Incapacité de stocker beaucoup d'informations**
    - doit être recâblé pour exécuter un nouveau programme
    - utilise des registres décimaux et non binaires

La machine de Von Neumann

- **Pour faire face à l'incapacité de stocker beaucoup d'informations de l'ENIAC**

- Description logique et théorique par **par John Von Neumann (1946)** d'une machine capable de stocker des programmes et des données en mémoire :
  - **Le concept de programme enregistré en mémoire**
  - **EDSAC** (Electronic Delay Storage Automatic Calculator) : **première machine totalement électronique du type de Von Neumann** (1949).

Des ordinateurs de plus en plus fiables, de plus en plus rapides

- **1950-1960 : deux importantes découvertes d'ingénierie ont changé l'image de l'ordinateur électronique :**
  - **l'utilisation des tores de ferrite comme mémoire**
    - La capacité de la mémoire vive (RAM) a augmenté de 8000 à 64000 mots
  - **l'utilisation du transistor au niveau des unités de traitement.**
    - un temps d'accès de 2 ms.

- Des machines très chères à l'achat et même à la location.
  - les grands centres informatiques du gouvernement, des industries, des universités et de certains grands laboratoires privés.
  - traitement par lot (batch) , time sharing

La naissance des microprocesseurs

- **Les années 60 -70:**
  - **des ordinateurs ultra-rapides grâce aux circuits intégrés**.
  - **la révolution micro-informatique**, mais aussi à la fabrication de supercalculateurs et de stations de travail.

- **Le génie logiciel prend une importance capitale**
  - l'achat d'un ordinateur ne se fait plus en fonction simplement de la puissance du matériel, mais aussi de sa possibilité de faire tourner les logiciels existants.

- **Les années 80 : généralisation de l'informatique personnelle**
  - **des ordinateurs plus conviviaux** (interfaces graphiques, la souris)
  - de moins en moins chers

- **Les années 90 :** c'est aussi l'ère du multimédia, de la réalité virtuelle et d'Internet.

Limites de la miniaturisation électronique ?

- En 1965, Gordon Moore établit sa fameuse "loi" qui prévoyait, en électronique,
  « un doublement du nombre de transistors par puce tous les deux ans »
- En 1971, le premier microprocesseur comptait environ 2000 transistors ;
- 40 ans plus tard, les puces intègrent deux milliards de transistors
  - **la puissance de calcul d'une puce multipliée par plus d'un million** !
- La taille des transistors approcherait celle de l'atome à l'horizon 2020.
  - **Intel promet pour 2018 des puces à 120 milliards de transistors**

- **premier retard constaté sur la célèbre loi.**
  - Dès 2015, Intel s'est heurté à des difficultés inattendues lui faisant retarder de six mois sa série Skylake gravée en 14 nanomètres

- **En dessous de 8 nanomètres, on prévoit que les effets quantiques perturberont le fonctionnement des composants électroniques**

Les futurs ordinateurs quantiques

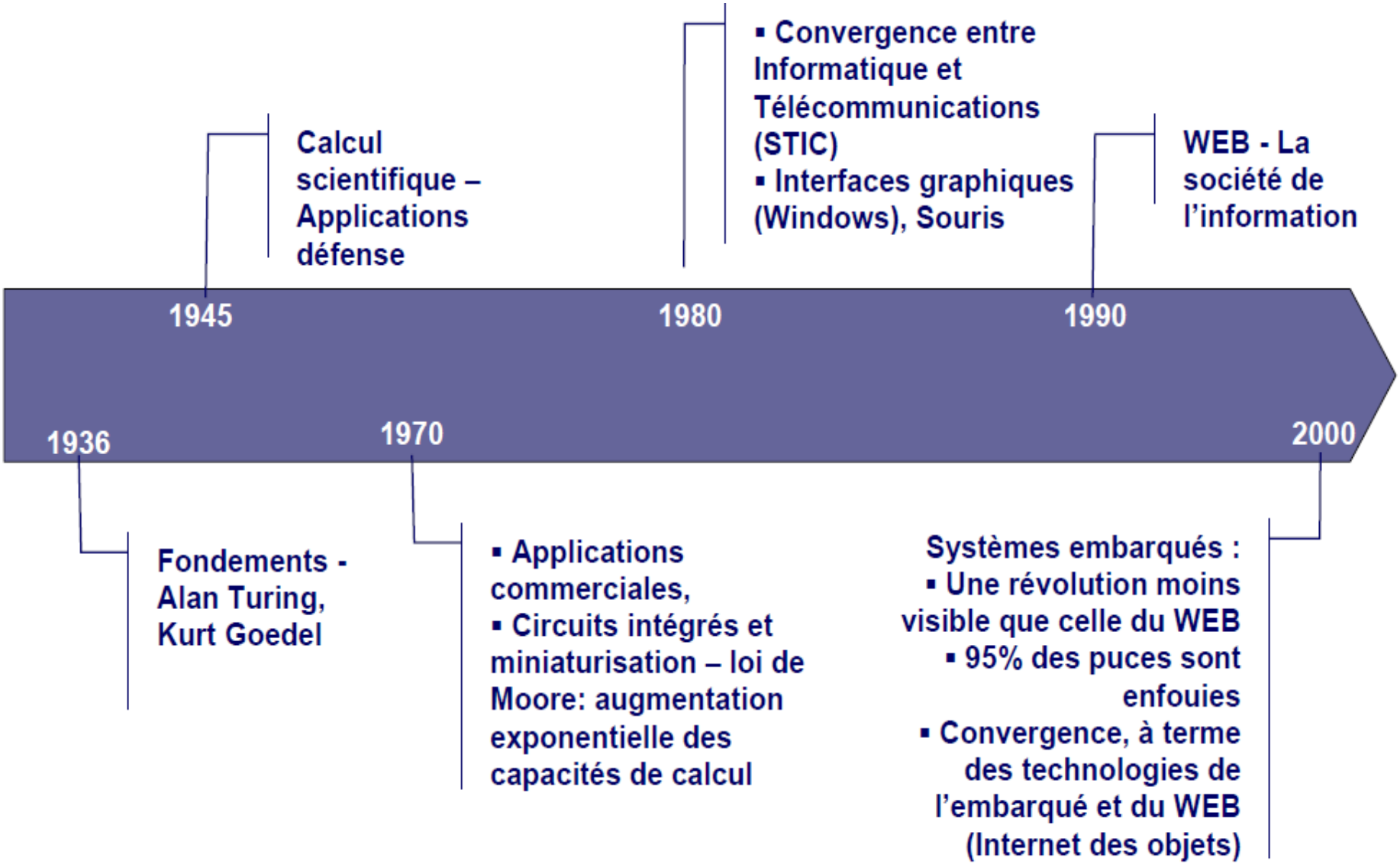
- **Utilisation des lois de la physique quantique pour s'affranchir du mode de calcul binaire**
  - dans un ordinateur classique, la valeur d'un bit est soit 1, soit 0, en revanche, un **"Qbit"** peut aussi prendre simultanément les deux valeurs, ce qui permet d'effectuer des calculs en parallèle.
  - Equivalent à une machine de Turing dont les bits (0 ou1) sont remplacés par des qubits
    - la puissance de calcul théorique d'un ordinateur quantique double à chaque fois qu'on lui adjoint un qubit.
  - pourraient aborder des problèmes hors de portée des machines de Turing :
    - Avec 10 qubits, on a 1024 états superposables, avec n qubits, 2^n
    - l'Algorithme de Shor permet de calculer en un temps polynomial la décomposition en produit de facteurs premiers

Les futurs ordinateurs quantiques

- **réalisation en quelques secondes de simulations numériques** qui demandent aujourd'hui des semaines ou des mois de travail aux meilleurs superordinateurs.

- **domaines d'applications visées :**
  - **Biologie :** simuler le comportement des interactions moléculaires ou cellulaires complexes
    - conception rapide des molécules thérapeutiques et des nouveaux médicaments
  - **Géophysique et climatique** : prévision météorologique voire même des tremblements de terre.
  - **Informatique :** vérification formelle de systèmes hétérogènes complexes (IoT)

L'évolution de l'Informatique



L’évolution de l’Informatique

- L'informatique n'est plus la science (ou la technique) des ordinateurs, elle devient la science de l'information numérique.
- C'est aussi une grande industrie qui irrigue toutes les autres.

- **Un progrès fulgurant en relation avec l’électronique**
  - capacités de traitement via la miniaturisation des composants électroniques (puces , nanotechnologies)
  - transmission de l’information par des réseaux
- **mais en décalage avec le savoir-faire en ingénierie du logiciel**
  - Intégration « correcte » de systèmes de systèmes hétérogènes

Convergence Internet des objets / systèmes cyber-physiques

Nouveaux défis scientifiques

- Développer des paradigmes, des théories et des méthodes pour :
  - garantir **l'interopérabilité** (intégration) des composants
  - passer de la vérification *a posteriori* à la **vérification par construction** (compositionnelle)
  - assurer **l'adaptativité** de systèmes en interaction avec des environnements de plus en plus incertains et imprévisibles
  - **exploiter efficacement les composants existants et futurs** (processeurs multi-cœur) dans **la programmation concurrente**

De grands défis sociétaux

- L'Informatique : une avancée technologique en si peu de temps et sans précédent dans l'histoire de l'Humanité.

- **L'informatisation de notre société** (a laquelle tout chacun participe, ne serait-ce qu'en tant que consommateur) **n'est pas sans danger** :
  - l'indépendance aujourd'hui entre l'information et son support veut dire que :
    - **l'information peut être manipulée, transformée automatiquement, transportée, copiée à l'infini.**
    - **Les questions de sécurité, d'éthique, de droit, prennent de plus en plus d'importance.**

Merci pour votre attention.