

Algorithmique

Programmation Objet

Python



Andrea G. B. Tettamanzi
Université Nice Sophia Antipolis
Département Informatique
andrea.tettamanzi@unice.fr

CM - Séance 1

Algorithmes

Plan

- Détails sur le fonctionnement de l'UE
- Introduction
- Algorithmes

Fonctionnement

- Structure de l'UE
 - Structures des données + leurs algorithmes
 - CM → TD (pseudo-langage) → TP (Python)
- Contrôle des connaissances
 - TD Noté (25%)
 - TP Noté (25%)
 - Contrôle terminal écrit (50%)

Groupes, EDT et Logistique

- A priori
 - Deux groupes de TD Info + 1 groupe TD M + MI + Elec
Mais il manque deux chargés
 - 3 groupes TP Info + 1 groupe TP M + MI + Elec
 - Mais là aussi il manque un chargé
- Page Web de l'enseignement :
 - <http://www.i3s.unice.fr/~tettaman/Classes/AlgoProgObjet/>

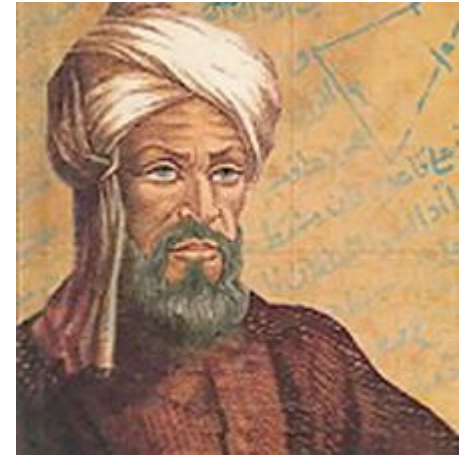
Bibliographie

- Cormen, Leiserson, Rivest. *Introduction to Algorithms*.
- Knuth. *The Art of Computer Programming*.
- Goodrich, Tamassia, Goldwasser. *Data Structures and Algorithms in Python*.
- ...

Remerciements

- Mes prédécesseurs
 - Jean-Charles Régin
 - Carine Fédèle
- Les chargés de TD et de TP de cette année et des années précédentes

Introduction



محمد بن موسى الخوارزمي

Algorithmique : une discipline très ancienne

- Remonte à l'Antiquité
 - Égypte antique : multiplication
 - Euclide : calcul du pgcd de deux nombres
 - Archimède : calcul d'une approximation de π
- "Algorithmes" < Muḥammad ibn Mūsā al-Khwārizmī, scientifique perse du IX^{ème} siècle, actif dans la Maison de la Sagesse de Bagdad
- L'algorithmique reste la base dure de l'Informatique:
 - Dans le software (logiciel)
 - Dans le hardware : un processeur est un câblage d'algorithmes fréquemment utilisés (multiplication ...)

Algorithmique

- L'aspect scientifique de l'informatique

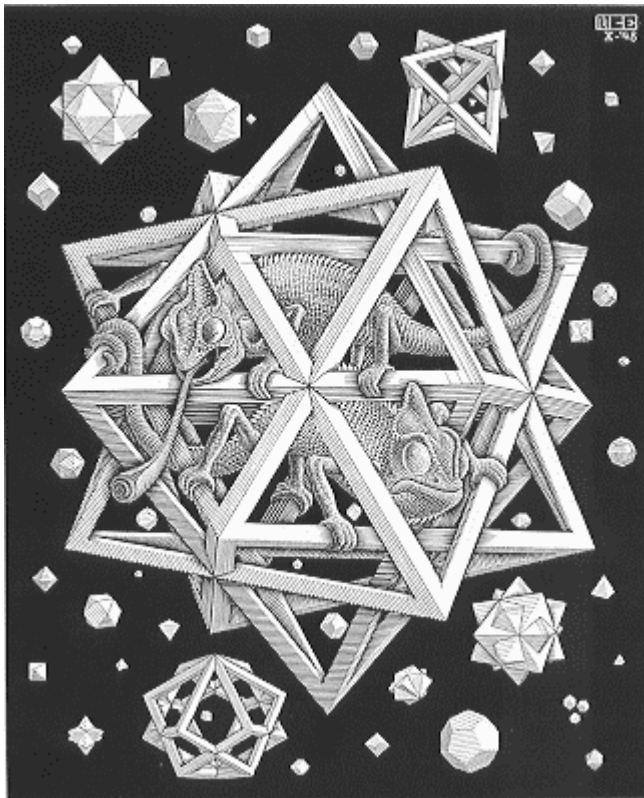
« Computer Science is no more about computers than astronomy is about telescopes » -- Michael R. Fellows

« L'informatique n'est pas plus la science des ordinateurs que l'astronomie n'est celle des télescopes »

(citée par Edsger Dijkstra, Turing award 1972)

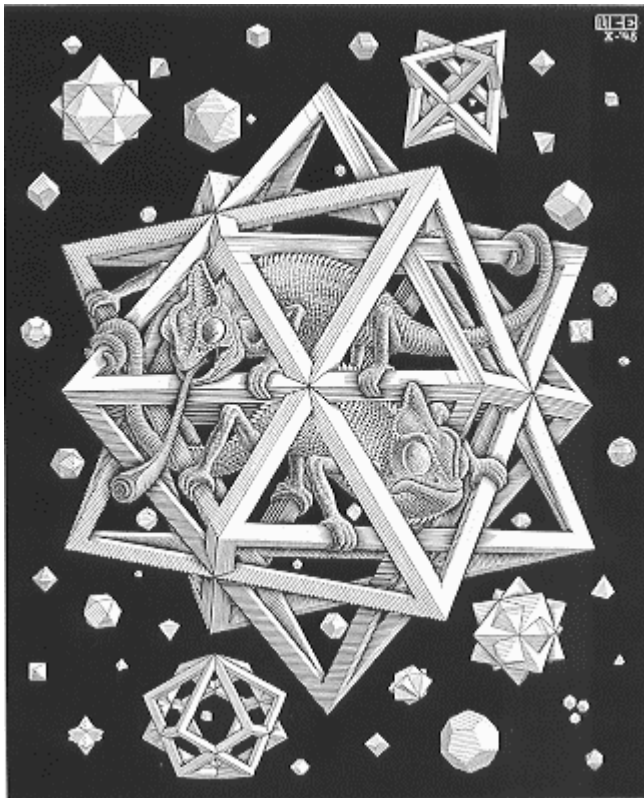
Résolution de problèmes

Problème



Résolution de problèmes

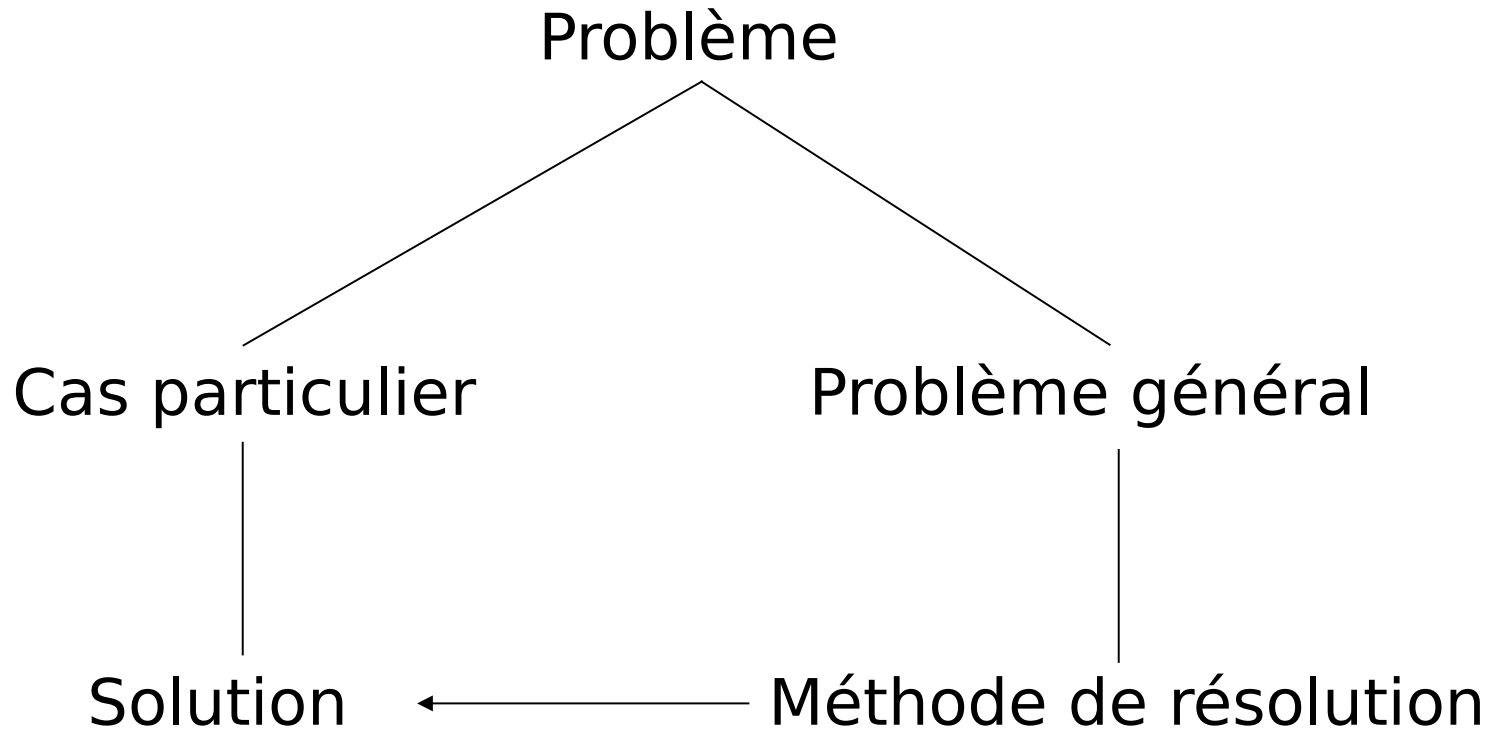
Problème



Solution

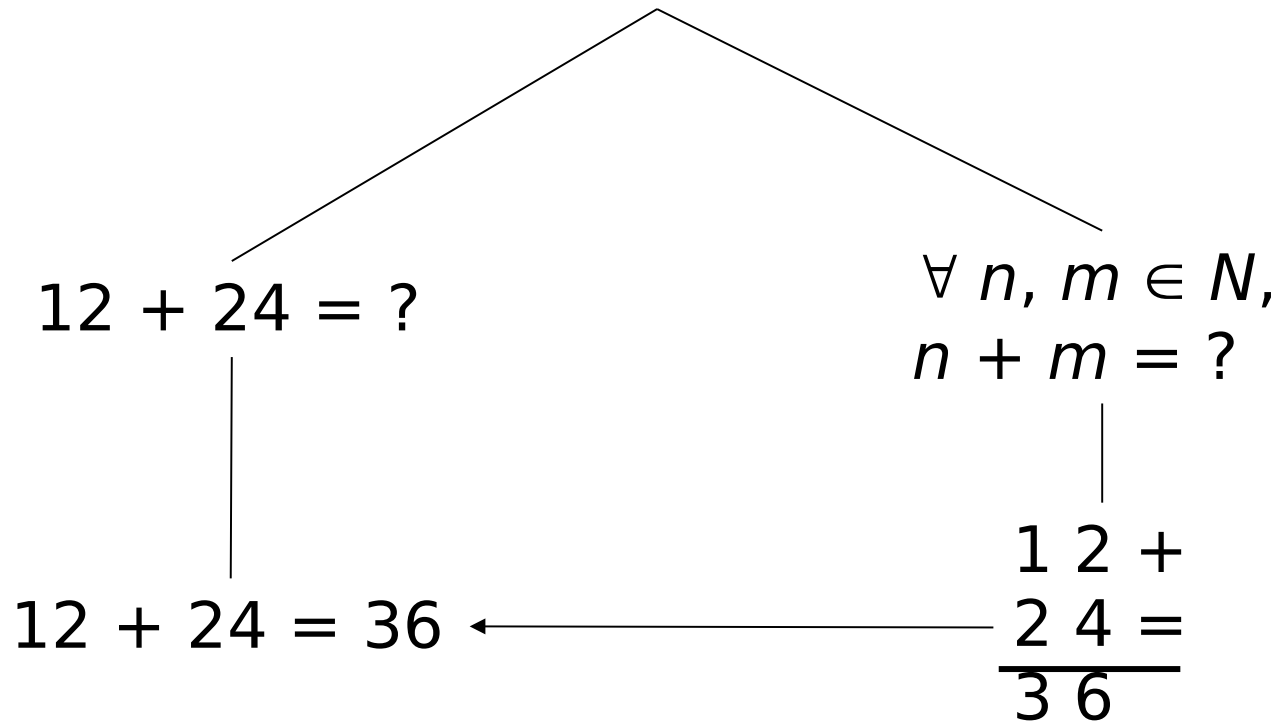


Problème et instance

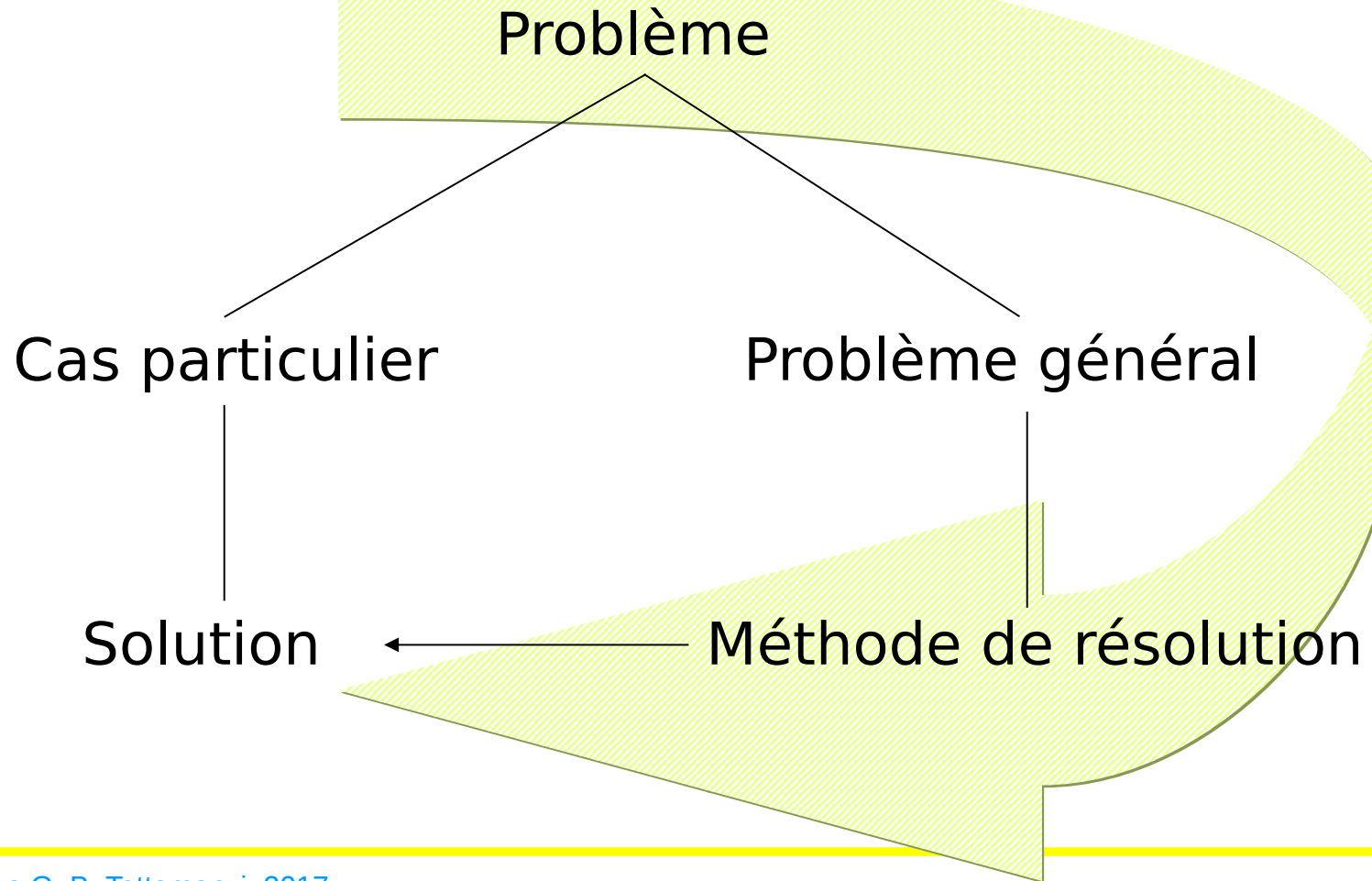


Exemple

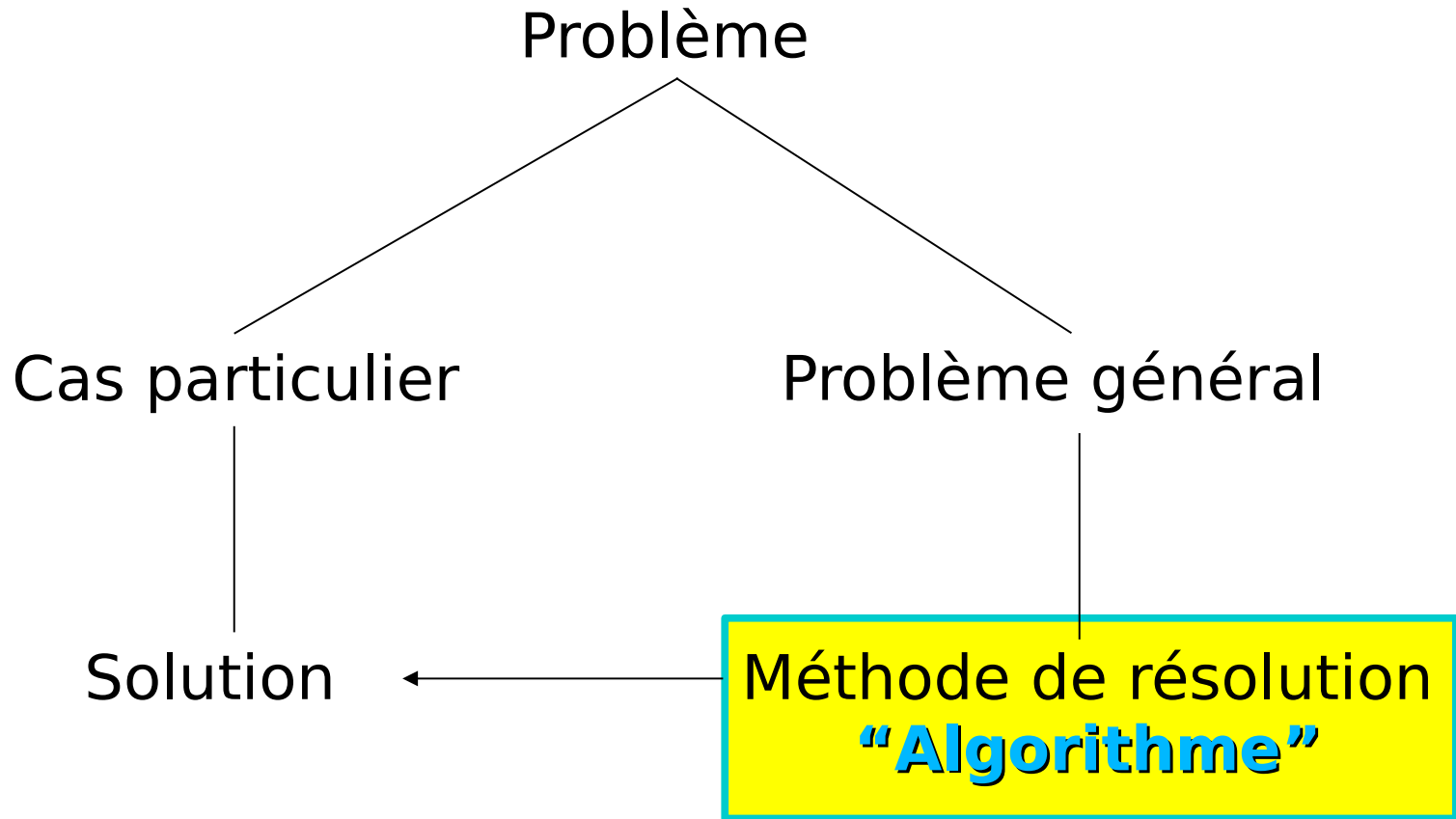
Addition



Résolution de problèmes



Résolution de problèmes



Définition d'algorithme

“Une **suite finie** et **non-ambiguë** d'opérations élémentaires (ou d'instructions) permettant de **résoudre un problème** sans aucun exercice d'intelligence et donc, par exemple, par une **machine**”

Algorithme

- Un algorithme prend en entrée des données et fournit un résultat permettant de donner la réponse à un problème
- Pour chaque instance
- Après un nombre fini d'opérations élémentaires

Langage

Comment peut-on exprimer un algorithme ?

Dans un langage qui doit être :

- adapté pour décrire la méthode de résolution ;
- précis et non ambigu ;
- compréhensible à l'exécuteur.

Exemples d'algorithmes

- recettes de cuisine ;
- instructions pour le montage d'un meuble ;
- les règles arithmétiques pour exécuter l'addition, la soustraction, la multiplication et la division de deux nombres ;
- les démarches bureaucratiques pour l'obtention d'un passeport.

pgdc(n, m) : Algorithme d'Euclide

1. $r \leftarrow n \pmod{m}$
2. si $r = 0$, le résultat est m , FIN
3. $n \leftarrow m, m \leftarrow r$
4. retourner au Pas 1.

Un faux algorithme

DECIDER SI n EST UN NOMBRE PREMIER

1. $N \leftarrow \{0, 1\}$
2. pour chaque couple de nombres entiers $i > 1$ et $j > 1$,
 $N \leftarrow N \cup \{i \times j\}$;
3. n est premier si et seulement si $n \notin N$.

Algorithme \neq Programme

- Un programme réalise un algorithme
- **Thèse de Turing-Church** : les problèmes ayant une solution algorithmique sont ceux solvables par une machine de Turing (théorie de la calculabilité)
- On ne peut pas résoudre tous les problèmes avec des algorithmes (indécidabilité)
- Problème de la halte
- Savoir de façon indépendante si un algorithme est correct

Merci de votre attention

