

Intelligence Artificielle : contour, évolutions et tendances

Marie-Christine Rousset

LIG, Université de Grenoble

Avec des extraits de transparents de **Didier Dubois** (IRIT), **Laurent Simon** (LRI), **Olivier Aycard** (LIG), **Thomas Eiter** (TU Wien)

Plan

- Présentation et illustration de l'approche IA
- Contours de l'IA
- Zoom personnel sur quelques évolutions et tendances
- Un avis personnel sur la place de l'IA dans l'enseignement de l'informatique

Qu'est-ce que l'IA ?

- Une branche de l'Informatique, dont le but est :
la **représentation**, l'**extraction** et l'**exploitation** de
connaissances interprétables et manipulables
dans des **formalismes à *dominante* symbolique**,
en vue d'**aider** à la résolution de **problèmes de**
décision.

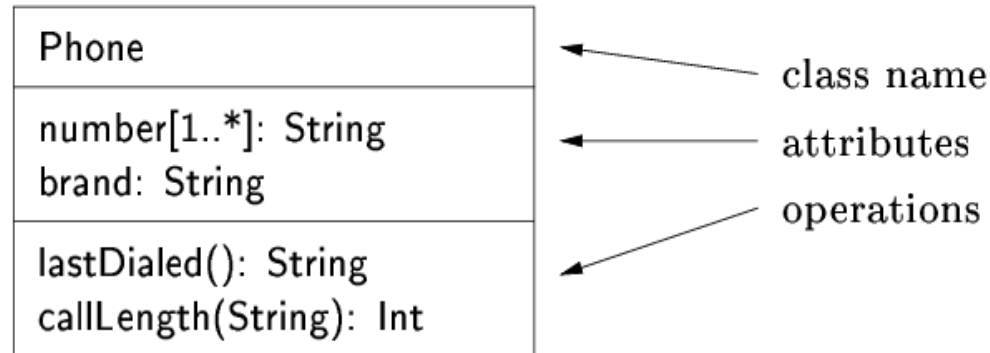
Ambition de l'IA

- Rendre la machine capable :
 - de **raisonner** sur une situation *statique* ou *dynamique*
 - pour faire un *diagnostic*, proposer une *décision*, un *plan* d'action
 - d'**expliquer** et de communiquer ses conclusions
 - d'**abstraire**, d'**apprendre**, de **découvrir** à partir de données
- Par des méthodes **génériques** susceptibles de s'adapter à de larges classes de situations.

Exemple

- Raisonner sur des modèles écrits en UML pour inférer des (bonnes ou mauvaises) propriétés
 - associer une sémantique formelle aux diagrammes UML
 - **facile** de traduire les diagrammes de données UML par des formules de la logique des prédicats avec égalité
 - appliquer des algorithmes d'inférence appropriés
 - **difficile** car la logique des prédicats est semi-décidable
 - **solution pour contourner cette difficulté** :
 - trouver un **fragment décidable** de la logique des prédicats avec égalité suffisamment expressif pour capturer le pouvoir d'expression d'UML:
⇒ **Les logiques de description**

Sémantique logique d'UML



Classes : prédicats unaires Attributs et types : prédicats binaires

Opérations $f(P_1, \dots, P_m):R$ prédicats d'arité $m+2$

+ axiomes logiques pour exprimer la sémantique:

$$\forall x. (\text{Phone}(x) \supset (\exists y. \text{number}(x,y) \wedge \exists z. \text{brand}(x,z)))$$

$$\forall x,y. (\text{Phone}(x) \wedge \text{number}(x,y) \supset \text{String}(y))$$

$$\forall x,r. (\text{Phone}(x) \wedge \text{lastDialed}(x,r) \supset \text{String}(r))$$

$$\forall r,r'. (\text{lastDialed}(x,r) \wedge \text{lastDialed}(x,r') \supset r=r')$$

Vérification de propriétés par inférence logique

- Consistance (satisfiabilité)
 - du diagramme de classes
 - L'ensemble des axiomes admet un modèle dans lequel au moins une classe est non vide
 - d'une classe
 - L'ensemble des axiomes admet un modèle dans lequel la classe est non vide
- Equivalence entre classes
 - Redondance du diagramme
- Inférence de types et de contraintes de cardinalité

Automatisation du raisonnement

- Le problème du raisonnement en logique des prédicats est semi-décidable
 - Traduire les diagrammes de classes en un ensemble de formules logiques ne suffit pas pour pouvoir vérifier **automatiquement** telle ou telle propriété logique
- **Encodage** des formules logiques résultant de la traduction d'un diagramme UML dans un **fragment décidable** de la logique du premier ordre
 - dans une logique de description (ALCQI)

Reasoning on UML Class Diagram using Description Logic Based Systems, *D. Berardi, D. Calvanese and G. De Giacomo*, Applications of Description Logics (ADL'01)

Contours de l'IA

Problématiques fondamentales

- Représentation de connaissances et de raisonnements
- Révision, mise à jour et fusion d'informations
- Problèmes de décision
- Apprentissage automatique
- Algorithmes génériques de résolution de problèmes

Les formalismes

- les logiques classique et non-classiques, la programmation logique et leurs méthodes de preuve.
- Les contraintes
- les graphes (réseaux sémantiques, graphes conceptuels, réseaux Bayésiens, ontologies)

Les thèmes d'applications

- Les jeux
- Planification d'actions
- Ordonnancement
- Diagnostic
- Les systèmes multi-agents
- La fouille de données
- Le web sémantique
- Traitement de la langue naturelle
- Enseignement assisté par ordinateur

Différents types de raisonnement

- Dans l'incertain (non-monotone, probabiliste)
- Tolérant les conflits, argumentation
- Temporel (logiques modales, points, intervalles)
- Sur l'action pour la planification
- Sur le changement (mise à jour, révision)
- Sur les connaissances mutuelles d'agents
- Sur les permissions et obligations (logiques déontiques)
- Spatial qualitatif sur les points et les lignes, sur l'inclusion, le contact entre objets géométriques (méréo-topologie)

Thématiques transversales

Font pour partie appel à des techniques d'IA :

- le traitement d'image
- le traitement du langage naturel
- l'indexation multimédia
- la robotique (cognitive)
- la réalité virtuelle
- la vie artificielle (étude, simulation ou réalisation d'êtres animés artificiels capables d'effectuer ensemble de manière adaptative des tâches spécialisées)
- les paradigmes de résolution de problèmes inspirés du vivant ou de l'humain: neurones artificiels, les méta-heuristiques, la logique floue

Evolutions et tendances: zoom personnel

- Les **réseaux Bayésiens** s'imposent comme modèle quantitatif pour le raisonnement dans l'incertain
- **L'apprentissage statistique** s'impose
- **Conditional Preferences Networks** : un nouveau modèle qualitatif pour l'aide à la décision
- **Answer Set Programming**: un nouveau paradigme de programmation logique
- Un tournant décisif dans la **résolution pratique de SAT**
- Les **logiques de description** s'imposent comme un standard pour la description d'ontologies et le web sémantique

Raisonner dans l'incertain

- Tâche centrale
 - Elaborer des croyances sur une situation à partir de connaissances génériques sur le monde et d'informations contingentes.
- Beaucoup de nos informations sont incertaines et/ou non Booléennes
 - Les connaissances génériques ont des exceptions
 - Les informations sur la situation courante ne sont pas forcément fiables
 - Les mots de la langue naturelle sont peu précis
- La logique classique ne suffit pas.

Historique

- Années 1970-80
 - Systèmes-experts MYCIN et PROSPECTOR avec des coefficients de vraisemblance ad hoc
- Limites formelles des Systèmes Experts
 - Pas de sémantique claire
 - Pas de traitement de l'information incomplète ou contradictoire
 - Confusion entre connaissance générique et contingente
- 3 voies ont principalement émergé
 - Logiques non-monotones
 - Argumentation
 - Réseaux Bayésiens

Exemple classique

A partir de :

$\text{manchot}(x) \Rightarrow \text{oiseau}(x) ; \text{oiseau}(x) \Rightarrow \text{vole}(x)$

$\text{manchot}(x) \Rightarrow \neg \text{vole}(x)$

$\text{manchot}(x_0) ; \text{oiseau}(x_1)$

on veut pouvoir déduire :

$\neg \text{vole}(x_0), \text{vole}(x_1)$

En logique classique : contradiction!

Systeme-expert: conclusion « de normand » pour x_0

Solutions:

- Donner la priorité aux informations spécifiques
- Argumenter
- quantifier $P(\text{vole} \mid \text{oiseau})$ etc... => **les réseaux Bayésiens**

Approche Bayésienne

- Hypothèse
 - tout état de connaissance est représentable par une distribution de probabilité unique sur les interprétations du langage
- Démarche
 - Connaissance générique = causalité entre variables + probabilités
 - Information contingente = instantiation de variables
 - Calculer $P(\text{Conclusion} \mid \text{Information Contingente})$
- Propriété utile
 - toute probabilité jointe positive est factorisable par un produit de probabilités conditionnelles formant un **graphe acyclique orienté** (DAG) visualisant des relations d'indépendance conditionnelle

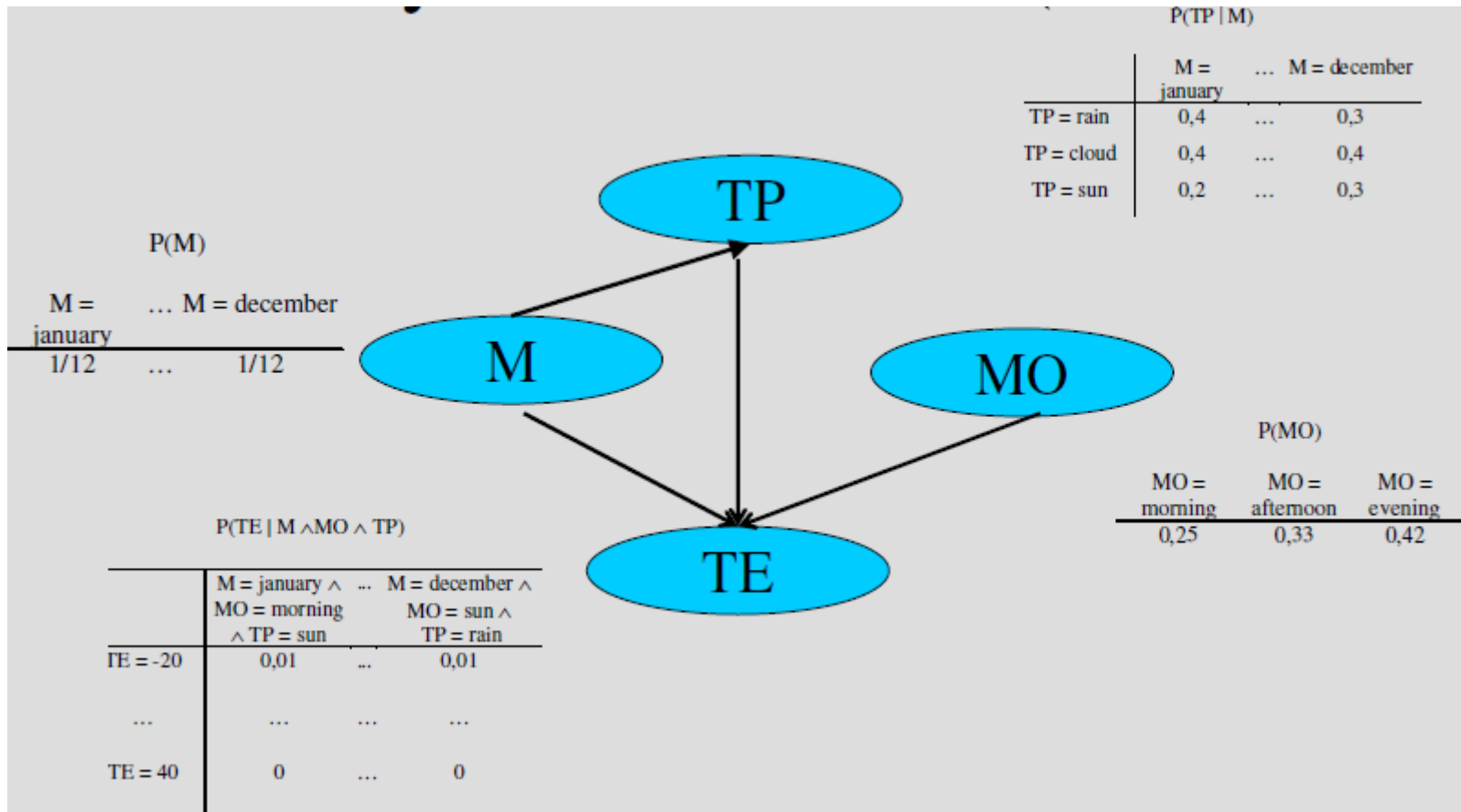
Exemple météo à Grenoble

MO: moment de la journée (variable discrète à 4 valeurs)

TP: temps (variable discrète à 3 valeurs)

M: mois (variable discrète à 12 valeurs)

TE: température (variable à valeurs entières dans [-20,40])



Avantages et inconvénients des réseaux Bayésiens

- $P(\text{Concl}|\text{IC})$ calculable par propagation locale dans les graphes sans circuits
- Rigueur: sémantique claire
- Représentation graphique « simple » à appréhender
- Non monotonie capturée si observations incomplètes
 $P(\text{Vole} | \text{Oiseau} \ \& \ \text{Manchot}) < P(\text{Vole} | \text{Oiseau})$

Mais

- Représentation graphique commode seulement si les connaissances forment un DAG (toujours consistant!)
- Complexité du calcul: NP-dur dans le cas général

Apprentissage artificiel

- Problématique du « Machine Learning »
 - Découvrir des régularités dans un ensemble de données (induction) et les utiliser sur de nouveaux cas
- Problématique reliée : Fouille de données
 - Recherche d'informations remarquables dans une base de données
 - Résumé (de masses) d'informations
- Deux visions
 - Apprentissage *supervisé* de connaissances génériques à partir d'ensembles d'exemples et de contre-exemples, ou plus généralement d'exemples étiquetés.
 - Apprentissage *non supervisé*: regroupement de données semblables en « clusters »

Apprentissage artificiel

- Spécificités de l'approche IA du Machine Learning
 - apprentissage *symbolique* de concepts: souci plus grand d'interprétabilité
 - apprentissage *relationnel*: programmation logique inductive
 - apprentissage de *structures* : réseaux bayésiens...
- Évolution
 - L'apprentissage automatique tend à se rapprocher de la statistique (et s'éloigner de l'IA?)
 - Apprentissage statistique: apprendre la distribution de probabilités des données à partir d'un échantillon

Raisonnement qualitatif pour l'aide à la décision

- Les CP-Nets émergent comme outil de modélisation et de raisonnement sur des préférences conditionnelles
- CP-Nets (Conditional Preferences Networks)
 - Modélisation graphique des préférences d'un utilisateur qui capture l'indépendance conditionnelle entre préférences
 - Analogie aux réseaux Bayésiens
 - Les tables de probabilités conditionnelles sont remplacées par des tables de préférence conditionnelles (information qualitative versus quantitative)
 - Une sémantique simple et rigoureuse de l'inférence de préférences
 - Des algorithmes pour
 - Comparer et ordonner des choix
 - Calculer les meilleurs choix compatibles avec les contraintes sur les préférences

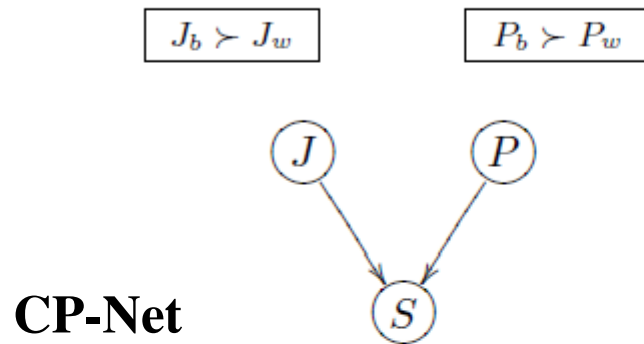
Exemple: comment m'habiller pour sortir ce soir ?

J: variable à 2 valeurs (black, white) \leftrightarrow choix possibles de couleur de ma veste

P: variable à 2 valeurs (black, white) \leftrightarrow choix possibles de couleur de mon pantalon

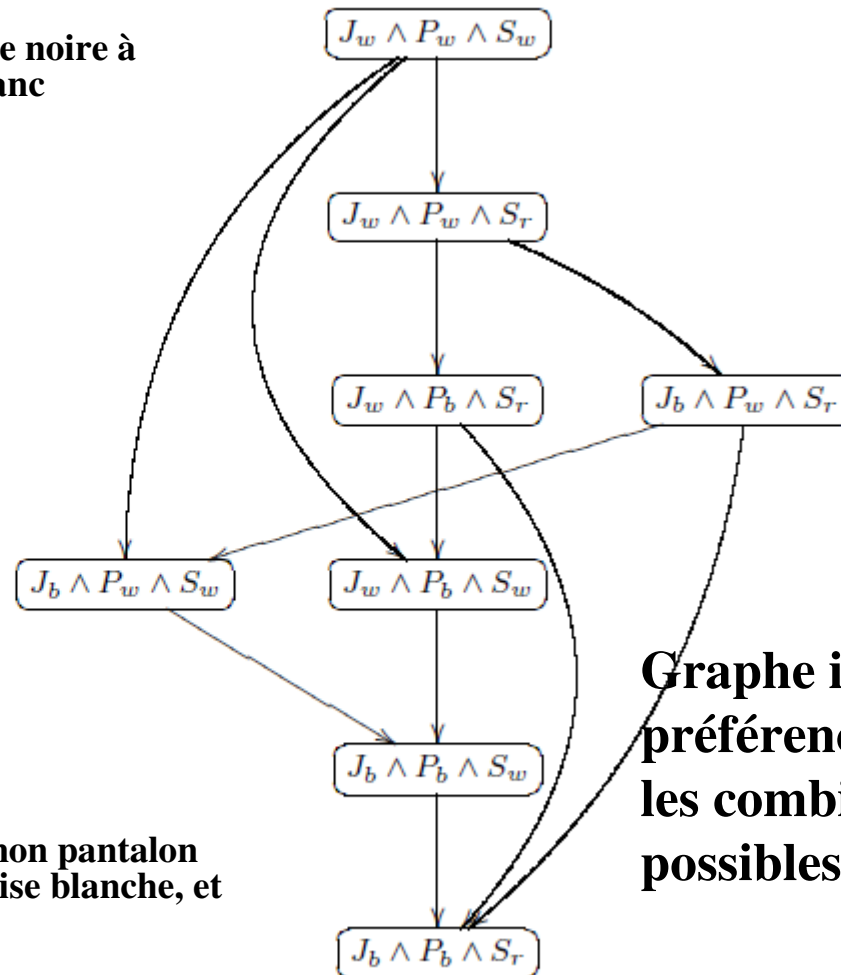
S: variable à 2 valeurs (red, white) \leftrightarrow choix possibles de couleur de ma chemise

Toute chose égale par ailleurs, je préfère une veste noire à une blanche, et un pantalon noir à un pantalon blanc



$J_b \wedge P_b$	$S_r \succ S_w$
$J_w \wedge P_b$	$S_w \succ S_r$
$J_b \wedge P_w$	$S_w \succ S_r$
$J_w \wedge P_w$	$S_r \succ S_w$

Toute chose égale par ailleurs, si ma veste et mon pantalon sont de couleur différente, je préfère une chemise blanche, et s'ils sont unis, je préfère une chemise rouge



Grappe induit de préférences entre les combinaisons possibles

En rouge et noir !

Answer Set Programming (ASP)

- Cadre général de programmation logique traitant de façon rigoureuse et « naturelle » le traitement d'informations incomplètes
- Avec des solveurs fondés sur des algorithmes et techniques proches de ceux pour SAT et les CSP
- De nombreuses applications
- Résultat de l'évolution des nombreux travaux sur le raisonnement non monotone des années 80, sur la sémantique de la négation et Prolog

ASP versus Prolog pour la résolution de problèmes

- La construction de preuves est remplacée par la construction de modèles (answer sets)
 - Un modèle (stable) est une solution d'un problème encodé par un programme logique
- Un programme logique: un ensemble de règles
$$a_1 \vee \dots \vee a_k \leftarrow b_1, \dots, b_m, \text{not } c_1, \dots, \text{not } c_n$$
$$a_i, b_j, c_l \text{ sont des littéraux (sans fonctions)}$$
- Un programme avec variables est remplacé par un programme équivalent sans variables
- M est un modèle stable s'il satisfait toutes les règles :
Si $\{b_1, \dots, b_m\} \subset M$ et $M \cap \{c_1, \dots, c_n\} = \emptyset$
Alors $M \cap \{a_1, \dots, a_k\} \neq \emptyset$

Illustration sur l'affectation de reviewers

paper(p1); paper(p2)

cand(« Thomas », p1) ; cand (« Enrico », p2); cand(« Marco », p2);

assign(X,P) ← cand(X,P), not -assign(X,P);

-assign(X,P) ∨ -assign(Y,P) ← cand(Y,P), cand(X,P), X≠Y;

is-assigned(P) ← assign(X,P);

← paper(P), not is-assigned(P);

Answers sets (solutions):

M1= {..., assign(« Thomas », p1), assign(« Enrico », p2), -assign(« Marco »,p2)}

M2= {..., assign(« Thomas », p1), assign(« Marco », p2), -assign(« Enrico »,p2)}

Des progrès majeurs dans la résolution pratique de SAT

LE GRAAL DE L'INFORMATIQUE THÉORIQUE



HISTORIQUE

SAT est le premier problème ayant été montré comme *difficile*.
Il caractérise tous les problèmes qui peuvent s'exprimer ainsi :

Trouver X t.q. $\exists X f(X)$

avec $f()$ une fonction qui se calcule *vite*.

SAT: le problème NP-complet prototypique

POURQUOI VAUT-IL SI CHER ?

SI VOUS SAVEZ LE RÉSOUDRE, VOUS SAUREZ AUSSI :

- Marier des couples suivant leurs affinités (beaucoup de couples)
- Minimiser votre trajet pour aller ramasser les enfants à leurs écoles respectives (beaucoup d'enfants) et revenir à la maison
- Décoder les informations cryptées circulant sur internet
- Vérifier qu'Ariane n'a pas (plus ?) de bugs à boum.
- Remplir de manière optimale votre coffre avant de partir en vacances
- Identifier les différences d'ADN caractéristiques entre les populations

LE CRAY INSTITUTE

Vous offre un million de dollars si vous y arrivez !

...au cœur de l'IA

L'UNE DES DÉFINITIONS DE L'I.A.

Résoudre (en pratique) des problèmes NP-Complets

Un tournant décisif

AVANT 2001

Si on butait sur un problème nouveau, on le transformait en SAT pour montrer qu'il était impossible à résoudre.

APRÈS 2001

Si on bute sur un problème difficile, on le transforme en SAT pour le résoudre efficacement. Les progrès dans la résolution pratique de SAT permettent de résoudre des instances gigantesques !

MAINTENANT

Si vous avez un problème, et que vous pouvez l'exprimer en SAT. Faites-le. Vous irez plus vite.

Illustration: le model checking

PRINCIPES DU BOUNDED MODEL CHECKING

Étant donné un automate décrivant les états possibles d'un système

VÉRIFIER QU'UN ÉTAT N'EST JAMAIS ATTEINT

C'est généralement la recherche de bug. Un état spécial « erreur » est utilisé dans la modélisation.

VÉRIFIER QUE LE SYSTÈME NE BLOQUE JAMAIS

Tout état doit être accessible depuis tout état, à n'importe quel moment du futur (impossible de construire une boucle infinie).

A des liens étroits avec les logiques temporelles.

Avant SAT, les BDD étaient utilisés pour résoudre ces problèmes.

Les logiques de description

Nombreux résultats de décidabilité et de complexité
+ Solveurs

⇒ la logique du premier ordre devient un standard
pour la description d'ontologies, au cœur du Web
sémantique et de l'intégration d'information

L'avenir dira si cela aura autant d'impact en
pratique que le choix dans les années 70 de la
logique comme fondement des SGBDs relationnels

**Quelle place de l'IA dans
l'enseignement de l'Informatique ?**

Point de vue personnel

- Des rudiments de logique ... le plus tôt possible (en L)
 - Pour l'IA, mais aussi les BD, la programmation, le génie logiciel, la preuve de programme, la théorie de la complexité et de la calculabilité
- Une approche déclarative et générique de la résolution informatique de problèmes (en M1, par des UEs au choix)
 - Réseaux Bayésiens (en plus probabilités)
 - Planification
 - Programmation logique
- Techniques de base de l'apprentissage automatique et de la fouille de données (en M1)

Merci de votre attention