Jacques Pitrat, la métaconnaissance et le bootstrap de l'IA

Marc PORCHERON

Ingénieur de Recherche Senior à EDF R&D

Thésard de Jacques Pitrat (1990)

marc.porcheron@edf.fr

- ***Métaconnaissance*, futur de l'IA**
- Réflexivité/autoréférence, bootstrap/amorçage
- *Métaconnaissance,* clé de la *singularité*
- En guise de conclusion

« Des systèmes doués d'une intelligence supérieure doivent être capables de connaître le domaine dans lequel ils travaillent, mais aussi d'examiner ce qu'ils doivent faire, ce qu'ils peuvent faire, ce qu'ils savent, ce qu'ils sont en train de faire.

Ils opèrent à un niveau "méta ", au-dessus du niveau du problème à traiter ; la connaissance devient pour eux objet d'étude.

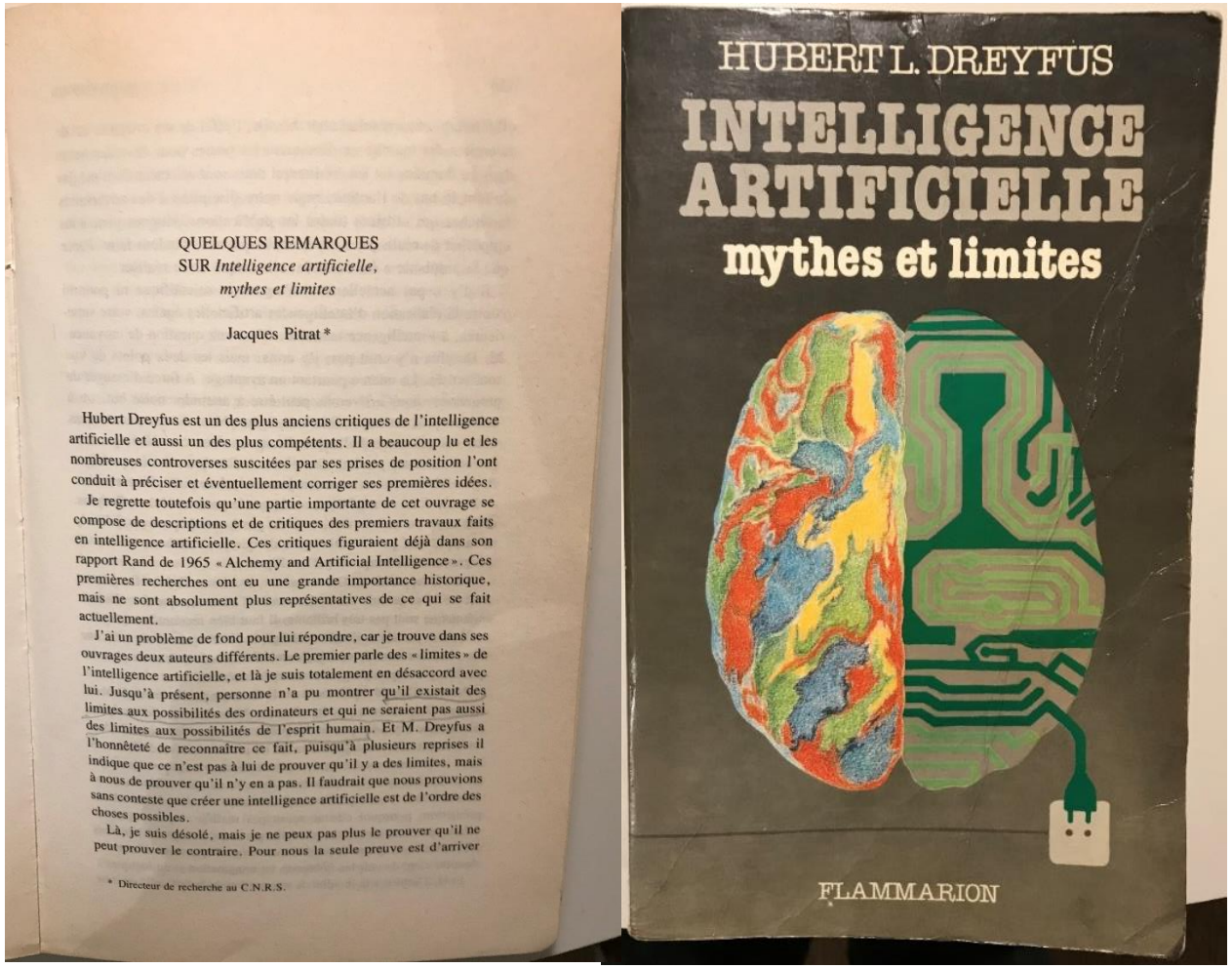
La métaconnaissance est précisément l'outil qui permet de travailler sur la connaissance et de réaliser de tels systèmes supérieurement intelligents. »

Une conviction très forte et très ancienne

- En 1979, en réponse H.L. Dreyfus

P 437 *: « Il n'est pas facile de trouver les connaissances que nous utilisons : elles sont souvent inconscientes. De plus, nous les stockons en mémoire sous une forme telle que nous ne pouvons les donner spontanément : elles sont déclenchées par les problèmes posés, nous ne pouvons les énumérer. Comme nous avons de la peine à extraire les connaissances d'un expert humain,* ***il faudra que nous en venions à la réalisation de programmes capables de les découvrir automatiquement, donc capable d'apprendre. Pour cela, ils devront disposer de connaissances pour trouver des connaissances****.*

L'autre difficulté est de réaliser des programmes capables d'utiliser efficacement de grandes quantités de connaissances. Il n'est pas possible pratiquement d'incorporer les connaissances au programme : celui-ci serait gigantesque, donc pratiquement impossible à mettre au point et à modifier. Il faut pouvoir les donner « en vrac », de façon à ce que nous puissions en ajouter, les enlever ou les modifier facilement. Mais comment réaliser un programme qui utilise avec efficacité des connaissances données sous cette forme ? ***Je pense que la solution est de prendre pour cela un programme de même nature qui dispose de connaissances pour utiliser efficacement des connaissances. Ce sont des méta- connaissances. »***



QUELQUES REMARQUES
SUR *Intelligence artificielle, mythes et limites*

Jacques Pitrat *

Hubert Dreyfus est un des plus anciens critiques de l'intelligence artificielle et aussi un des plus compétents. Il a beaucoup lu et les nombreuses controverses suscitées par ses prises de position l'ont conduit à préciser et éventuellement corriger ses premières idées.

Je regrette toutefois qu'une partie importante de cet ouvrage se compose de descriptions et de critiques des premiers travaux faits en intelligence artificielle. Ces critiques figuraient déjà dans son rapport Rand de 1965 « Alchemy and Artificial Intelligence ». Ces premières recherches ont eu une grande importance historique, mais ne sont absolument plus représentatives de ce qui se fait actuellement.

J'ai un problème de fond pour lui répondre, car je trouve dans ses ouvrages deux auteurs différents. Le premier parle des « limites » de l'intelligence artificielle, et là je suis totalement en désaccord avec lui. Jusqu'à présent, personne n'a pu montrer qu'il existait des limites aux possibilités des ordinateurs et qui ne seraient pas aussi des limites aux possibilités de l'esprit humain. Et M. Dreyfus a l'honnêteté de reconnaître ce fait, puisqu'à plusieurs reprises il indique que ce n'est pas à lui de prouver qu'il y a des limites, mais à nous de prouver qu'il n'y en a pas. Il faudrait que nous prouvions sans conteste que créer une intelligence artificielle est de l'ordre des choses possibles.

Là, je suis désolé, mais je ne peux pas plus le prouver qu'il ne peut prouver le contraire. Pour nous la seule preuve est d'arriver

* Directeur de recherche au C.N.R.S.

Des connaissances …

- Fait F : « Russel est un logicien »
  - *Logicien(Russel)*
- Règle R : « Tout les logiciens sont bizarres »
  - $\forall(x) Logicien(x) \sqsupset Bizarre(x)$
  - *SI Logicien(x) ALORS Bizarre(x)*

… et des connaissances qui s'appliquent sur des connaissances

- **Pour utiliser efficacement des connaissances**
  - *« Si R_1 et R_2 sont des règles et si le nombre de prémisses de R_1 est inferieur à celui de R_2, alors R_1 est plus facile à évaluer que R_2 »*
  - *« Si R_1 et R_2 sont des règles et si le nombre de conclusions de R_1 est inferieur à celui de R_2, alors R_1 est moins intéressante à évaluer que R_2 »*
- **Pour découvrir de nouvelles connaissances**
  - Une heuristique de découverte en mathématique :

  « Si un concept repose sur une notion d'ensemble, alors étudier les concepts correspondant aux cas « extrêmes », c'est-à-dire ceux où cet ensemble comporte un très petit nombre d'éléments, ou au contraire un très grand nombre d'éléments » (d'après [AM,EURISKO D. Lenat, 1983])
  - Concept : « Ensemble *D* des diviseurs d'un entier »
  - |D| très grand ➔ …
  - |D| très petit ➔ on a découvert les nombres premiers !!!!

- *Métaconnaissance*, futur de l'IA
- **Réflexivité/autoréférence, bootstrap/amorçage**
- *Métaconnaissance,* clé de la *singularité*
- En guise de conclusion

« ***Le but d'un amorçage est de mettre en œuvre un objet à l'aide de lui-même***. *Cette définition apparaît à première vue paradoxale, car comment se servir d'un objet qui n'existe pas encore.*

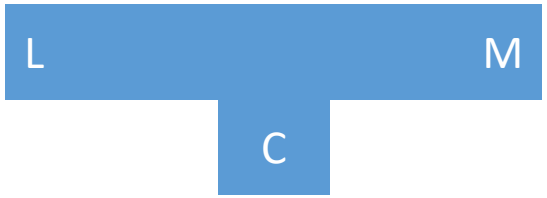
Pour comprendre l'amorçage, nous devons faire intervenir le facteur temps. ***Il y a en réalité une succession d'objets de plus en plus perfectionnés et chacun participe à l'élaboration de son successeur. Vu sous cet angle, le paradoxe disparaît.*** *Nous utilisons très souvent cette méthode pour réaliser des tâches complexes.*

L'amorçage est utile dans toutes les situations où nous nous disons : «Je ferais plus facilement ce que je suis en train de faire si je l'avais déjà fait»».

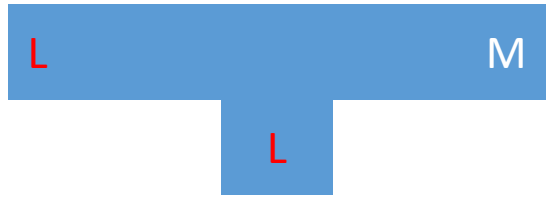
Jacques Pitrat in « Des métaconnaissances pour des systèmes intelligents » Quaderni Année 1995 25 pp. 29-42

Le bootstrap, une technique bien connue des informaticiens …

- … pour écrire un *compilateur* dans *le même langage* que celui qu'il compile
- Soit un langage (*L)* dont on veut écrire un compilateur … en (*L)*
  - A priori impossible, à moins d'avoir les pouvoirs magiques du Baron de Munchausen qui pouvait se soulever en tirant sur les sangles de ses propres bottes …
- *T-Diagrams :*



Un compilateur écrit en C qui compile le langage L en un langage M
(M est supposé exécutable sur la machine, éventuellement après avoir été lui-même compilé)



?

Le bootstrap, une technique bien connue des informaticiens …

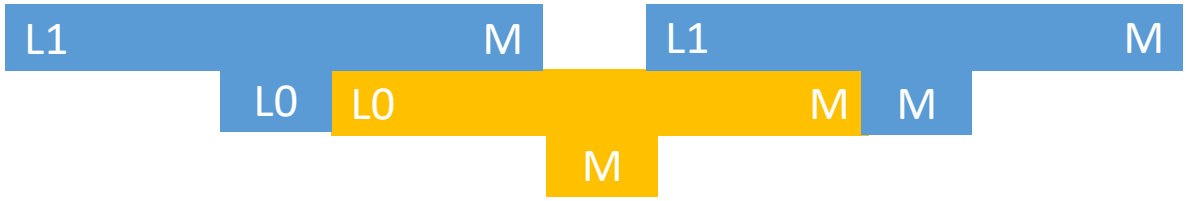
- Amorçage
  - Ecrire en *(M)* un compilateur *C0* pour un *sous-langage (L0)* de *(L) :*
    - *Par exemple (L0) ne contient que des boucles « for » et pas de boucles « while »*

→ *C0*

- Extensions
  - Etendre (L0) en (L1) ⊂ L et écrire le compilateur de (L1) en (L0) :

→ *C1*

  - Appliquer C0 à C1 :

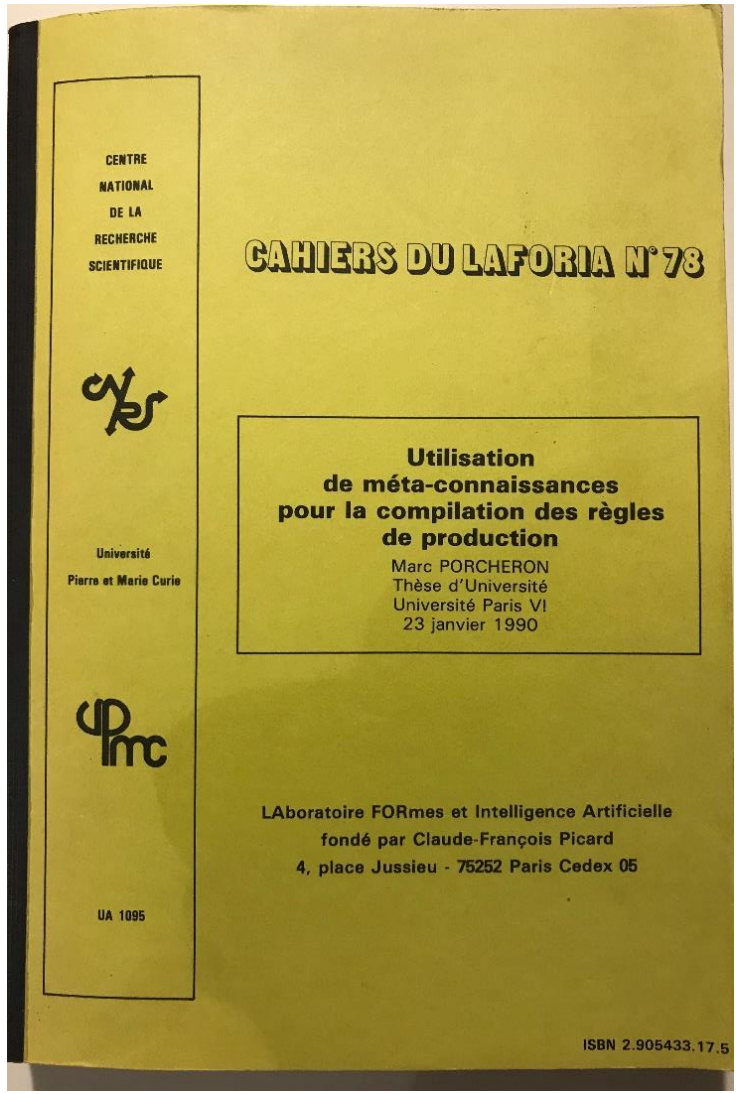


→ *C2*

⋮

  - Répéter l'opération jusqu'à obtenir :

- J. Pitrat a montré que ces principes pouvaient s'appliquer au bootstrap de la métaconnaissance, en utilisant un « moteur de démarrage »
- Ces sont ces idées que j'ai essayé de mettre en pratique dans ma thèse sur une *meta-expertise de compilation de langages à base de règles de production* …



Quelques slides de ma soutenance (il y a 30 ans …)

généralités

Objectifs :

Analyser le système, indépendamment de toute procédure d'interprétation prédéfinie, de manière au contraire, à décider comment il doit être interprété.

Adapter l'interprétation à ce qui risque effectivement de se passer :

Réaliser une partie des opérations normalement effectuées par le moteur.

Supprimer des opérations inutiles.

Déterminer de manière heuristique la méthode de mémorisation à utiliser .

Générer, au moins partiellement, le moteur.

généralités

Connaissances générales sur le modèle d'exécution des langages de règles :

Si une prémisse P1 est plus restrictive qu'une prémisse P2, alors il est intéressant d'instancier P1 avant P2

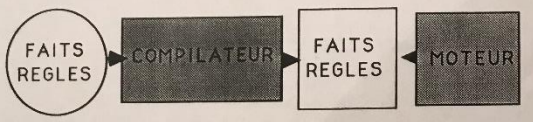
Si la mémoire de travail évolue rapidement, alors il est dangeureux de mémoriser beaucoup d'information entre les cycles.

Si deux règles comportent des conditions contradictoires, alors elles ne peuvent être candidates en meme temps

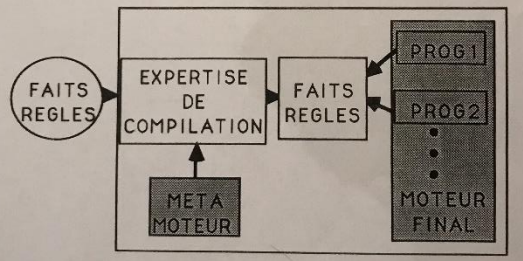
• • •

généralités

Comparaison

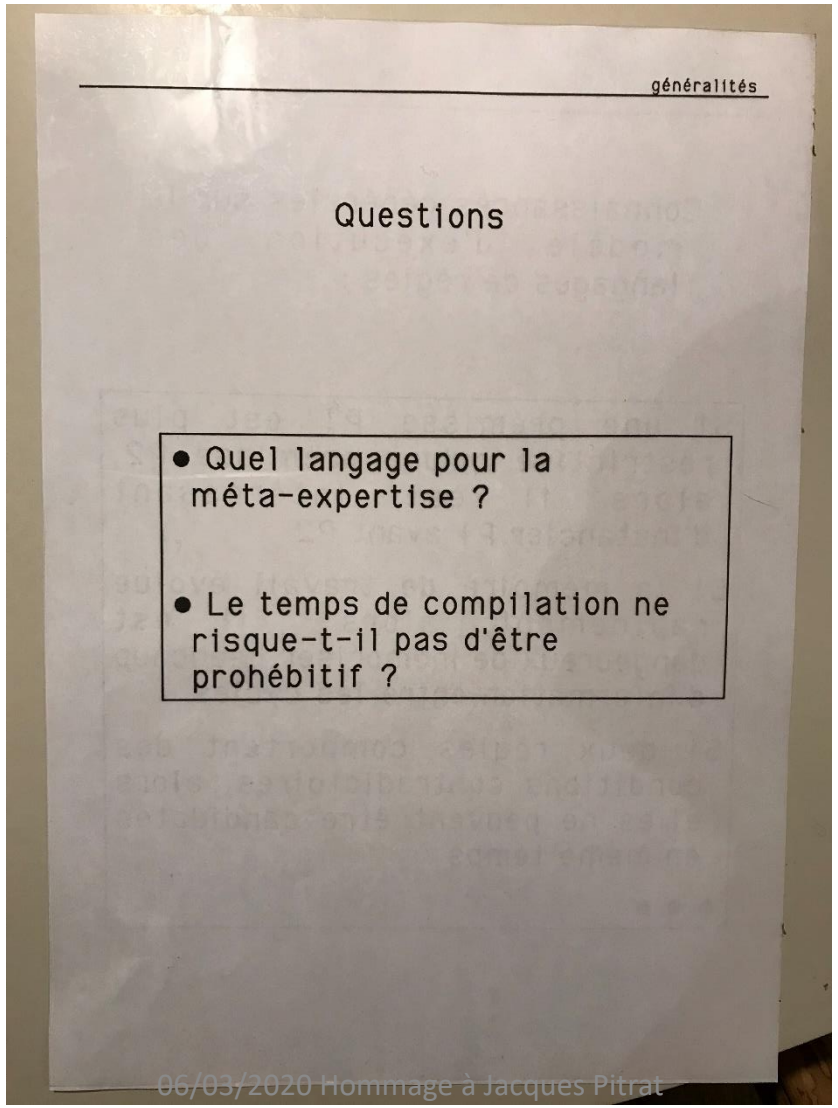


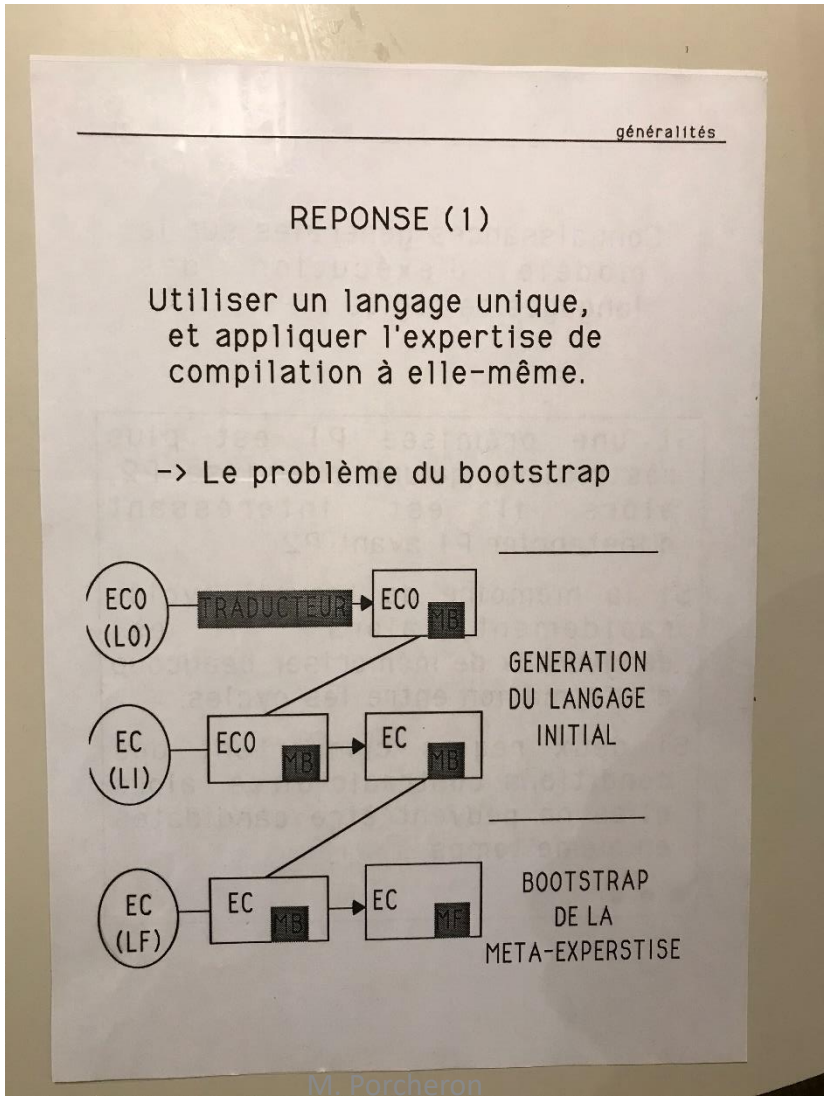
Interprétation classique

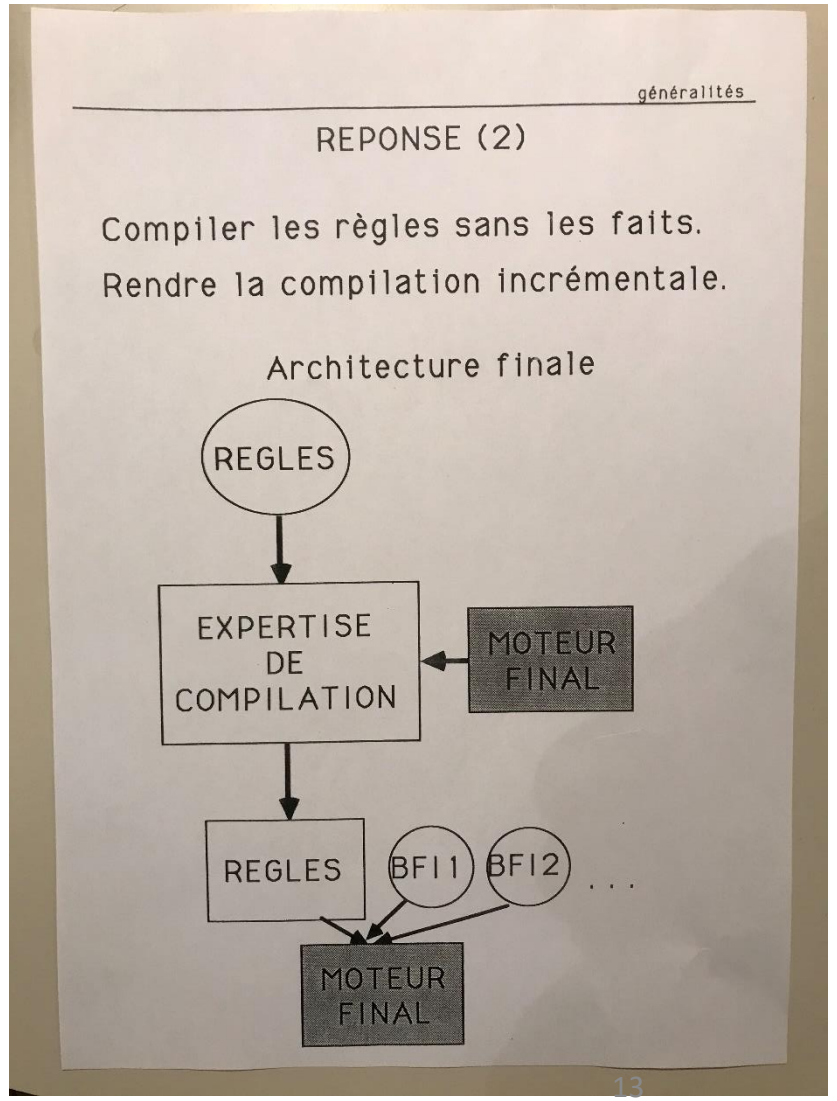


Interprétation utilisant des connaissances de compilation

Quelques slides de ma soutenance (il y a 30 ans …)

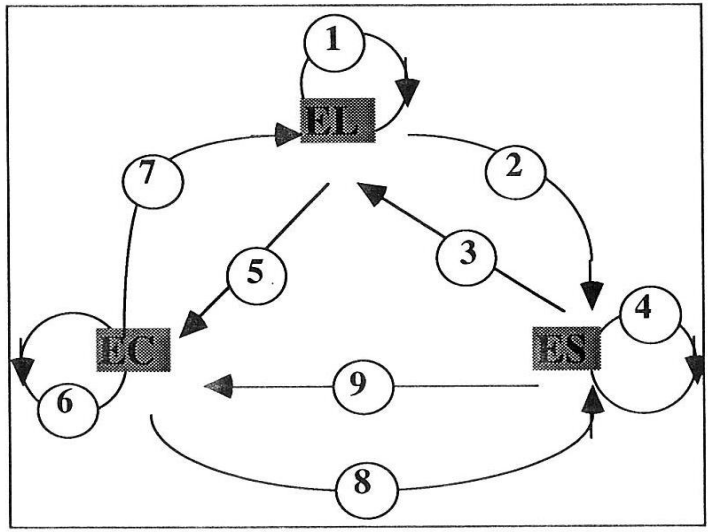






Quelques slides de ma soutenance (il y a 30 ans …)

Génération du langage initial(3)



Le langage Final

- Bases de faits : LORE
- Bases de Règles :

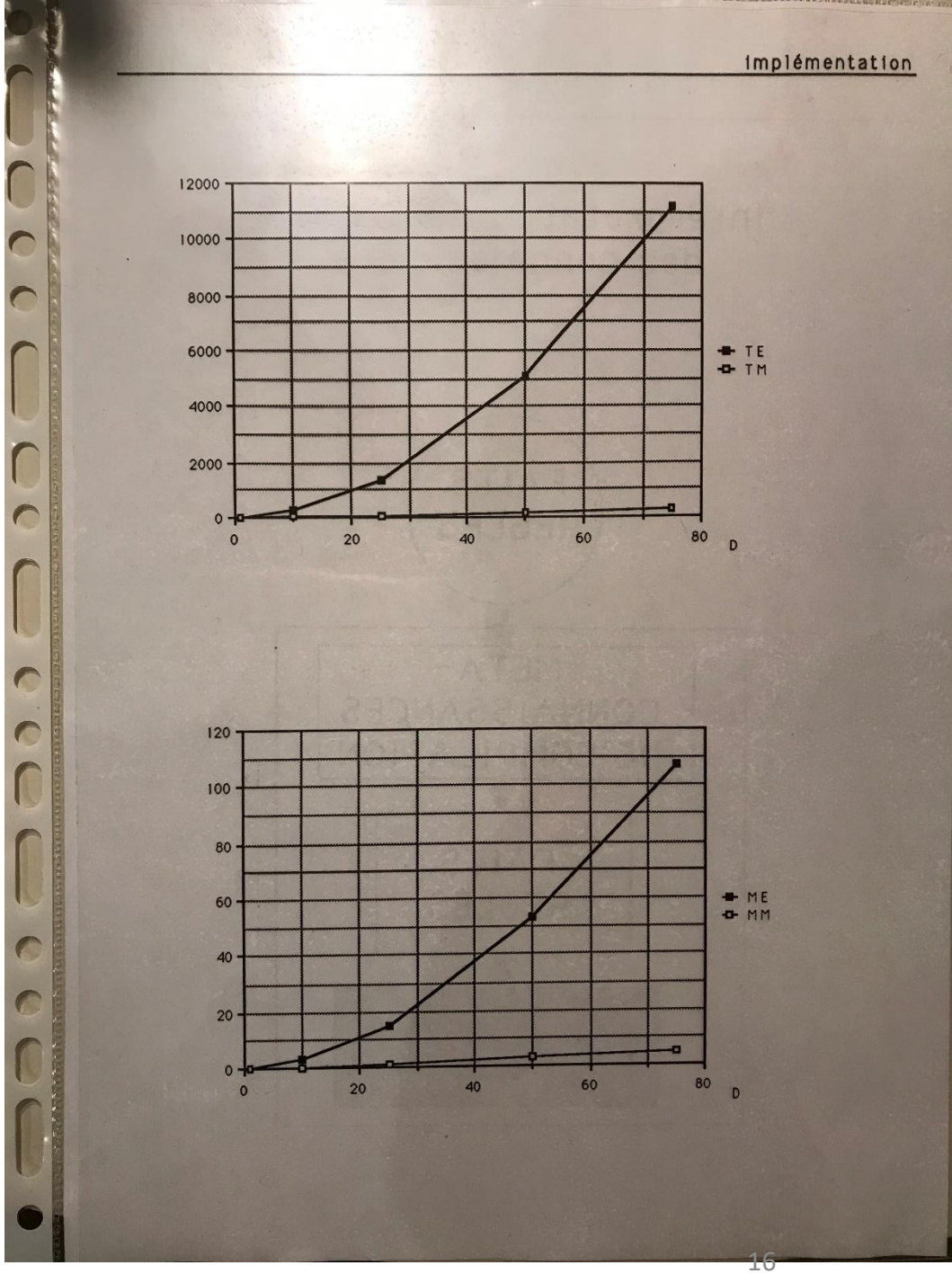
```
[R1 isa rule
lhs
  [?y belongs [?x friends]]
  [?z belongs [?y friends]]
  [?u = [?z age] ]
  [?u > 20]
rhs
  [?x friends add ?z]
]
```

Résultats

- Le (méta-)moteur d'inférences « MILORE » et son expertise de compilation
- 500 méta-règles
- 100 pour la génération du langage initial d'amorçage
- 400 pour l'analyse et la compilation du langage final :
  - Analyse des interactions entre règles
  - Construction d'un « graphe de propagation » modélisant la relation d'influence entre règles (une action de R1 est susceptible de valider une condition de R2 …)
  - Décomposition du graphe en composantes connexes et ordonnancement de celles-ci par la fonction de rang
  - Heuristiques d'exécution efficace en fonction de l'analyse (degré de mémorisation d'information utile, génération de code exécutant les parties « procédurales » éventuellement détectées dans la base de connaissances …)

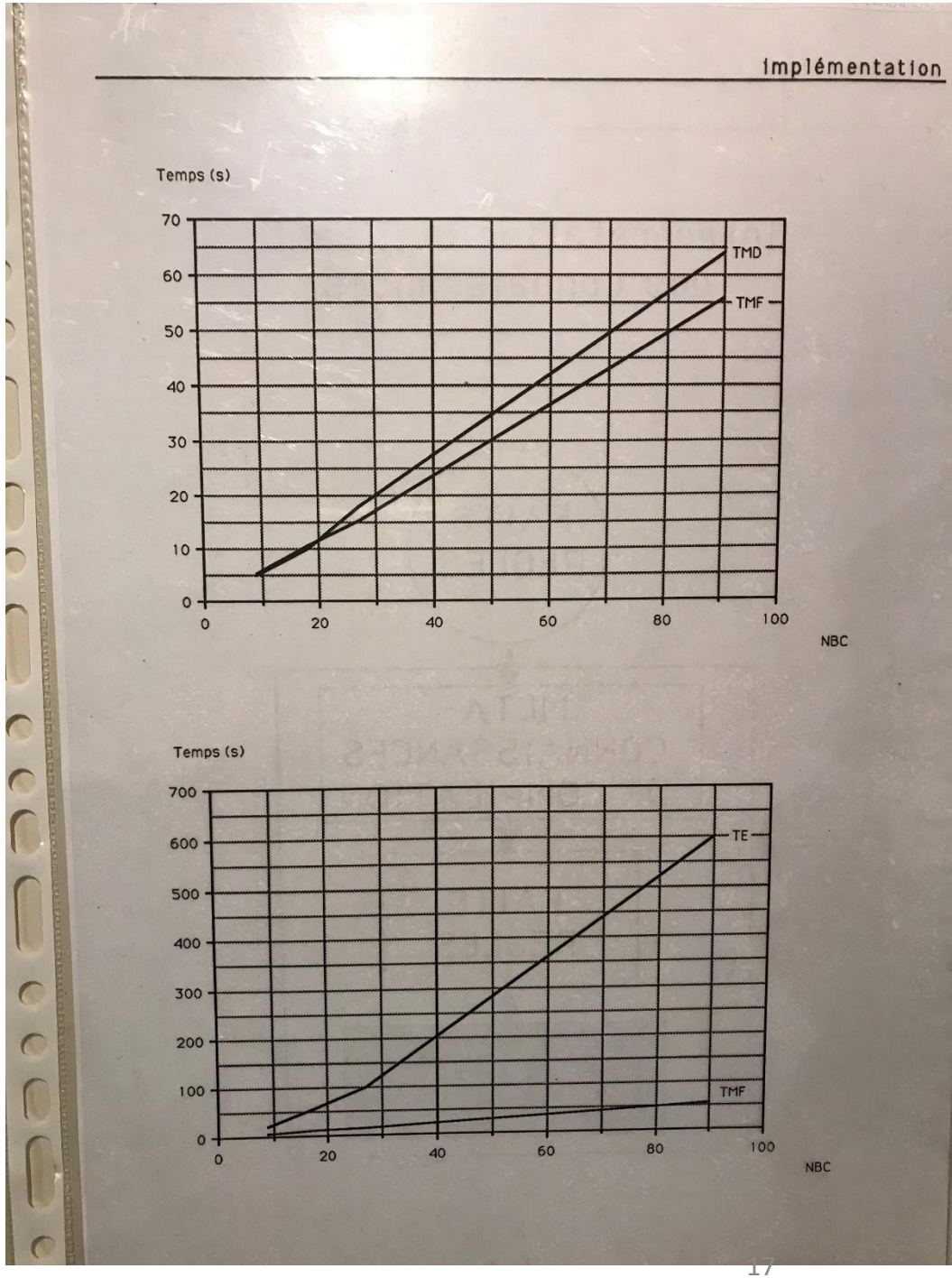
Résultats

- Application à une base de connaissances traitant un problème de diagnostic de pannes dans un système informatique (une quinzaine de règles)
  - TE (rep. ME) temps d'exécution (resp. mémoire consommée) par un moteur d'inférences classique
  - TM (rep. MM) temps d'exécution (resp. mémoire consommée) par MILORE
  - Abscisse D : taille des données (nombre de machines dans le système diagnostiqué)



Résultats

- **… Et bien sûr application du système à lui-même, en l'occurrence à une quarantaine de méta-règles de compilation des parties conditions de règles**
  - TMD temps d'exécution par le *moteur de démarrage*, les règles étant données dans le *langage initial*, i.e. avec des informations procédurales de contrôle (ordre …)
  - TMF par le *moteur final,* les règles étant données dans le *langage final,* i.e. de manière purement déclarative, « en vrac »
  - TME : temps d'exécution par un moteur d'inférences classique
  - Abscisse NBC : taille des données (nombre de conditions de règles à compiler)
- NB : Le moteur final est aussi efficace que le moteur de bootstrap (et même un peu meilleur) : on s'est bien « débarrassé » du langage initial au profit d'un langage purement déclaratif, sans perte de performances …



- *Métaconnaissance*, futur de l'IA
- Réflexivité/autoréférence, bootstrap/amorçage
- ***Métaconnaissance,* clé de la *singularité***
- En guise de conclusion

« Supposons qu'existe une machine surpassant en intelligence tout ce dont est capable un homme, aussi brillant soit-il. La conception de telles machines faisant partie des activités intellectuelles, cette machine pourrait à son tour créer des machines meilleures qu'elle-même; cela aurait sans nul doute pour effet une réaction en chaîne de développement de l'intelligence, pendant que l'intelligence humaine resterait presque sur place.

Il en résulte que la machine ultra intelligente sera la dernière invention que l'homme aura besoin de faire, à condition que ladite machine soit assez docile pour constamment lui obéir »

Good, I. J. (1965), membre de l'équipe d'A. Turing sur le projet Enigma de 1941 à 1942.

« ***J'avais suggéré que l'IA pouvait nous aider à mettre en œuvre l'IA. Ce n'est possible que si nous rendons l'IA réflexive, capable de s'appliquer à elle-même.***

Cela est le cas à partir du moment où l'on introduit les métaconnaissances.

Nous trouverons de nouvelles connaissances pour découvrir des connaissances à l'aide des connaissances pour découvrir des connaissances déjà existantes.

Nous pouvons ainsi amorcer l'IA ».

Jacques Pitrat in « Des métaconnaissances pour des systèmes intelligents » Quaderni Année 1995 25 pp. 29-42

« UN AVENIR À TRES LONG TERME

[…] ***Une critique classique, mais mal fondée, contre l'IA est la constatation que l'ordinateur ne fait jamais que ce qu'on lui dit de faire. Tout le monde en est convaincu ! Dans l'approche classique de l'IA, on veut justement lui dire de se comporter intelligemment. Le problème est que nous ne savons pas encore bien comment le lui dire. Avec l'approche basée sur la métaconnaissance, nous voulons lui dire de se comporter de plus en plus intelligemment. Nous espérons amasser assez de métaconnaissances pour arriver à des systèmes capables de s'améliorer. Cette progression sera lente au départ. Mais, dans ce schéma, tout progrès amène une accélération des progrès ultérieurs. Notre but est que l'IA assure elle-même la plus grande partie de la construction de son état final »***

Jacques Pitrat *in « Des métaconnaissances pour des systèmes intelligents » Quaderni Année 1995 25 pp. 29-42*

- *Métaconnaissance*, futur de l'IA
- Réflexivité/autoréférence, bootstrap/amorçage
- *Métaconnaissance,* clé de la *singularité*
- **En guise de conclusion**

- ***Métaconnaissance, bootstrap,*** deux concepts fructueux pour l'IA que J. Pitrat a découverts et inlassablement investigués
- Ses travaux démontrent leur pertinence dans différents domaines, essentiellement ceux liés à **l'amélioration des performances des systèmes d'inférences par leur auto-analyse**
- Mais J. Pitrat insistait toujours sur l'extrême difficulté à découvrir l'ensemble « minimal » de métaconnaissances d'apprentissage qui permettrait d'initier le bootstrap et de déclencher cette **auto-construction de l'IA** qu'il voyait comme la clé d'une potentielle **singularité**
- Bien que reposant sur des principes différents de ceux sur lesquels J. Pitrat a concentré ses efforts, **l'IA connexionnistes/neuronale** est aussi un champ d'application de ses idées: ***AlphaGo Zero*** **est devenu le meilleur joueur de Go au monde en ne « connaissant » au départ que les règles du jeu et en jouant uniquement contre lui-même ….**
- **Faire collaborer des IA logiques/symboliques et connexionnistes/neuronales pour aller vers le « bootstrap » d'une IA générale ?**