

Systemes multi-agents

Cours 5 – Agents logiques et hybrides

Cédric Buron

`cedric.buron@yahoo.fr` | `buron.cedric.free.fr`

Ingénieur de recherche décision

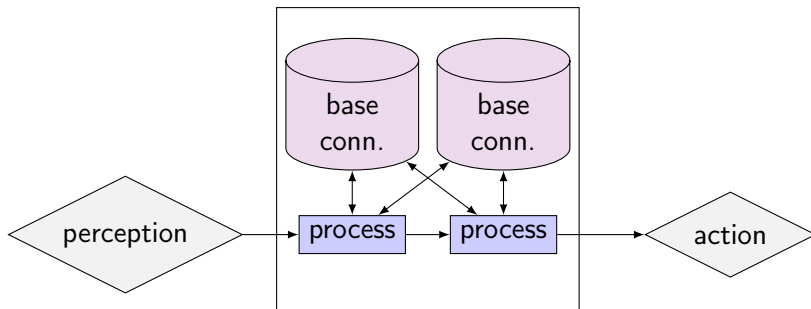
THALES



RAPPEL DES ÉPISODES PRÉCÉDENTS

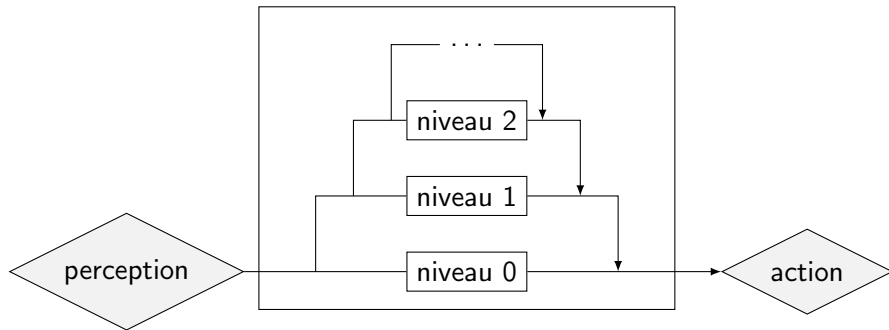
Agent cognitif

Agent cognitif



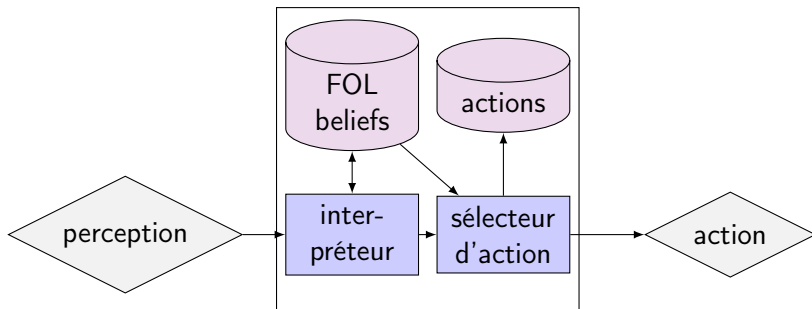
Agent réactif

Agent réactif



AGENTS LOGIQUES

Architecture générale



Exemple : Agent de diagnostique



AD



A₁



A₂



A₃



A₄

beliefs

$fievre(A) \wedge infl_gorge(A) \implies angine_blanche(A)$

$fievre(A) \wedge infl_oreilles(A) \implies otite_bact(A)$

$angine_blanche(A) \vee otite_bact(A) \iff bact(A)$

$\neg fievre(A) \wedge infl_gorge(A) \implies angine_rouge(A)$

$\neg fievre(A) \wedge infl_oreilles(A) \implies otite_vir(A)$

$angine_rouge(A) \vee otite_vir(A) \iff viral(A)$

$bact(A) \iff \neg viral(A)$

$bact(A) \implies Do(precrire(antibio, A))$

$viral(A) \implies Do(precrire(sympt, A))$

$\neg bact(A) \implies \neg Do(precrire(antibio, A))$

$prescrit(A) \implies Do(suivant())$

$\neg prescrit(A) \implies \neg Do(suivant())$

actions

examiner fievre(A)

examiner gorge(A)

examiner oreilles(A)

suivant()

prescrire antibio(A)

prescrire physio(A)

Exemple : Agent de diagnostique



AD



A₁



A₂



A₃



A₄

beliefs

$fievre(A) \wedge infl_gorge(A) \implies angine_blanche(A)$

$fievre(A) \wedge infl_oreilles(A) \implies otite_bact(A)$

$angine_blanche(A) \vee otite_bact(A) \iff bact(A)$

$\neg fievre(A) \wedge infl_gorge(A) \implies angine_rouge(A)$

$\neg fievre(A) \wedge infl_oreilles(A) \implies otite_vir(A)$

$angine_rouge(A) \vee otite_vir(A) \iff viral(A)$

$bact(A) \iff \neg viral(A)$

$bact(A) \implies Do(precrire(antibio, A))$

$viral(A) \implies Do(precrire(sympt, A))$

$\neg bact(A) \implies \neg Do(precrire(antibio, A))$

$prescrit(A) \implies Do(suivant())$

$\neg prescrit(A) \implies \neg Do(suivant())$

actions

examiner fievre(A)

examiner gorge(A)

examiner oreilles(A)

suivant()

prescrire antibio(A)

prescrire physio(A)

$\neg infl_gorge(A_1)$

Exemple : Agent de diagnostique



AD



A₁



A₂



A₃



A₄

beliefs

$fievre(A) \wedge infl_gorge(A) \implies angine_blanche(A)$

$fievre(A) \wedge infl_oreilles(A) \implies otite_bact(A)$

$angine_blanche(A) \vee otite_bact(A) \iff bact(A)$

$\neg fievre(A) \wedge infl_gorge(A) \implies angine_rouge(A)$

$\neg fievre(A) \wedge infl_oreilles(A) \implies otite_vir(A)$

$angine_rouge(A) \vee otite_vir(A) \iff viral(A)$

$bact(A) \iff \neg viral(A)$

$bact(A) \implies Do(precire(antibio, A))$

$viral(A) \implies Do(precire(sympt, A))$

$\neg bact(A) \implies \neg Do(precire(antibio, A))$

$prescrit(A) \implies Do(suivant())$

$\neg prescrit(A) \implies \neg Do(suivant())$

actions

examiner fievre(A)

examiner gorge(A)

examiner oreilles(A)

suivant()

prescrire antibio(A)

prescrire physio(A)

$\neg infl_gorge(A_1)$

$infl_oreille(A_1)$

Exemple : Agent de diagnostique



AD



A₁



A₂



A₃



A₄

beliefs

$fievre(A) \wedge infl_gorge(A) \implies angine_blanche(A)$

$fievre(A) \wedge infl_oreilles(A) \implies otite_bact(A)$

$angine_blanche(A) \vee otite_bact(A) \iff bact(A)$

$\neg fievre(A) \wedge infl_gorge(A) \implies angine_rouge(A)$

$\neg fievre(A) \wedge infl_oreilles(A) \implies otite_vir(A)$

$angine_rouge(A) \vee otite_vir(A) \iff viral(A)$

$bact(A) \iff \neg viral(A)$

$bact(A) \implies Do(precire(antibio, A))$

$viral(A) \implies Do(precire(sympt, A))$

$\neg bact(A) \implies \neg Do(precire(antibio, A))$

$prescrit(A) \implies Do(suivant())$

$\neg prescrit(A) \implies \neg Do(suivant())$

actions

examiner **fievre(A)**

examiner **gorge(A)**

examiner **oreilles(A)**

suivant()

prescrire **antibio(A)**

prescrire **physio(A)**

$\neg infl_gorge(A_1)$

$infl_oreille(A_1)$

$\neg fievre(A_1)$

$otite_vir(A_1)$

$\neg bact(A_1)$

$\neg Do(precire(antibio, A_1))$

$Do(precire(sympt, A))$

Exemple : Agent de diagnostique



AD



A₁



A₂



A₃



A₄

beliefs

$fievre(A) \wedge infl_gorge(A) \implies angine_blanche(A)$

$fievre(A) \wedge infl_oreilles(A) \implies otite_bact(A)$

$angine_blanche(A) \vee otite_bact(A) \iff bact(A)$

$\neg fievre(A) \wedge infl_gorge(A) \implies angine_rouge(A)$

$\neg fievre(A) \wedge infl_oreilles(A) \implies otite_vir(A)$

$angine_rouge(A) \vee otite_vir(A) \iff viral(A)$

$bact(A) \iff \neg viral(A)$

$bact(A) \implies Do(precire(antibio, A))$

$viral(A) \implies Do(precire(sympt, A))$

$\neg bact(A) \implies \neg Do(precire(antibio, A))$

$prescrit(A) \implies Do(suivant())$

$\neg prescrit(A) \implies \neg Do(suivant())$

actions

examiner fievre(A)

examiner gorge(A)

examiner oreilles(A)

suivant()

prescrire antibiotio(A)

prescrire physio(A)

$\neg infl_gorge(A_1)$

$infl_oreille(A_1)$

$\neg fievre(A_1)$

$otite_vir(A_1)$

$\neg bact(A_1)$

$\neg Do(precire(antibio, A_1))$

$Do(precire(sympt, A))$

Implémentations

Agent Oriented Programming & AGENT0 (Shoham, 1993)

Implémentation d'un agent logique accessible

- Composants :
 - ▶ ensemble de croyances,
 - ▶ capacités (*i.e.* actions),
 - ▶ engagements,
 - ▶ règles d'engagement.
- Comportement de l'agent basé sur les règles d'engagement :

$regle_engagement : (message, croyance) \rightarrow engagement$

- Messages :
 - ▶ *request/unrequest* : changement d'engagement,
 - ▶ *inform* : changement de croyances.
- Actions privées ou communicatives (messages).

Implémentations

Concurrent MetateM (Fisher *et al.*, 1991)

Logique temporelle (FOL + opérateurs temporels) :

- $\Box\phi$: ϕ sera toujours vrai
- $\blacksquare\phi$: ϕ a toujours été vrai
- $\bigcirc\phi$: ϕ sera vrai au prochain pas de temps
- $\bigcirc\phi$: ϕ était vrai au dernier pas de temps
- $\Diamond\phi$: ϕ sera vrai à un moment dans le futur
- $\Diamond\phi$: ϕ a été vrai à un moment dans le *passé
- $\phi U \psi$: ϕ est vrai jusqu'à ce que ψ soit vrai
- $\phi W \psi$: ϕ est vrai à moins que ψ soit vrai
- $\phi S \psi$: ϕ est vrai depuis que ψ est vrai
- $\phi Z \psi$: ϕ est ou sera vrai à partir du moment où ψ est ou sera vrai

Implémentations

Concurrent MetateM (Fisher *et al.*, 1991)

Exemples :

- « Si on vient de me demander la ressource r , je donne la ressource r au pas de temps suivant »
- « Si on vient de me demander la ressource r , je fournis la ressource r jusqu'à ce qu'on me demande d'arrêter »
- « Si j'ai fourni la ressource r , r ne sera plus jamais disponible »

Implémentations

Concurrent MetateM (Fisher *et al.*, 1991)

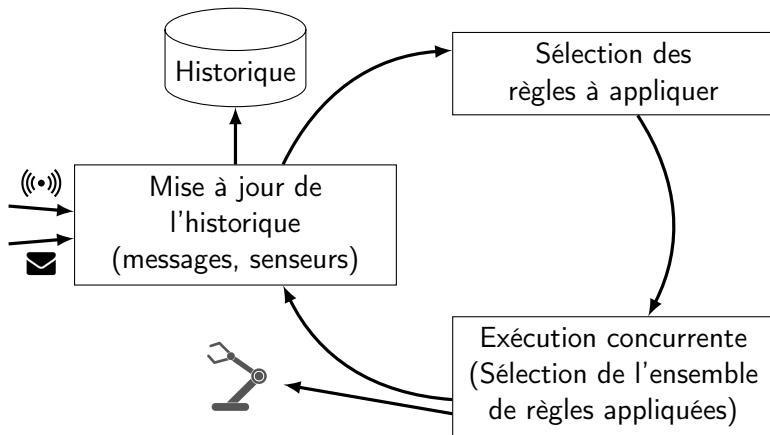
Exemples :

- « Si on vient de me demander la ressource r , je donne la ressource r au pas de temps suivant »
- $\mathcal{O}demande(r) \implies \mathcal{O}fournit(r)$
- « Si on vient de me demander la ressource r , je fournis la ressource r jusqu'à ce qu'on me demande d'arrêter »
- $\mathcal{O}demande(r) \implies fournit(r) \text{ } U \text{ } stop(r)$
- « Si j'ai fourni la ressource r , r ne sera plus jamais disponible »
- $\diamondsuit fournit(r) \implies \Box[\neg disponible(r)]$

Implémentations

Concurrent MetateM (Fisher *et al.*, 1991)

Boucle d'exécution



Agents logiques : Bilan

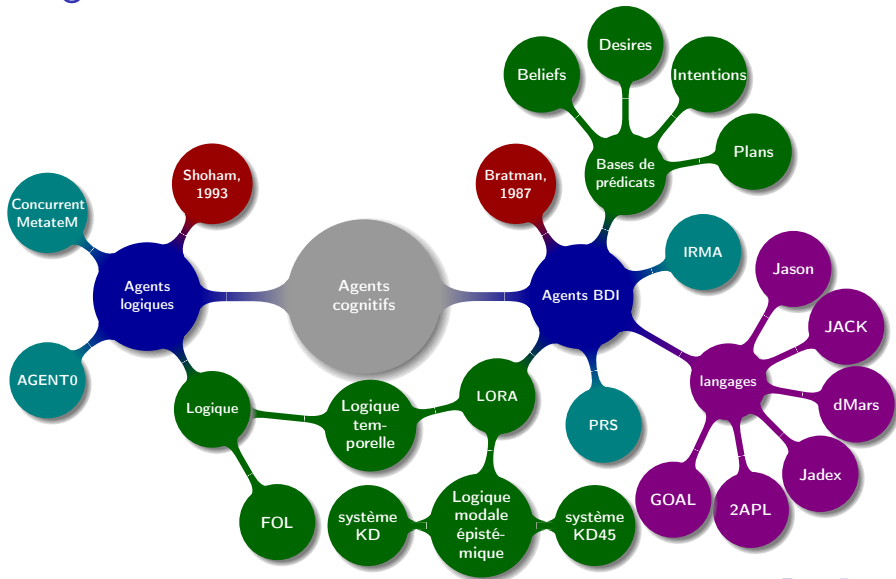
Résumé :

- agents dotés d'une base de propositions logiques (premier ordre/temporelle)
- règles permettant de déclencher des actions
- pas de différenciation entre désirs, croyances...

Implémentations :

- Agent0 : logique du premier ordre
- Concurrent MetateM : logique temporelle

Agents cognitifs : carte des idées



AGENTS HYBRIDES

Principe

Agents cognitifs :

- capables de résoudre des problèmes complexes,
- capables de proactivité,
- processus long.

Agents réactifs :

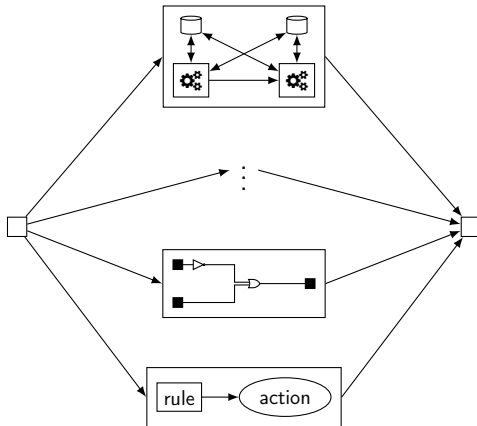
- rapides,
- capables d'émergence,
- limitations.

Objectif : concilier les avantages des deux approches

Deux possibilités :

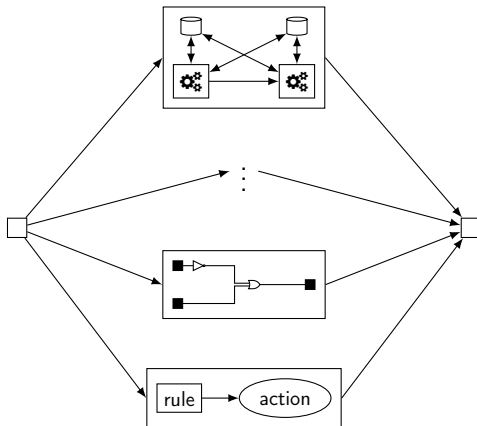
- architectures cognitives/réactives dans une architecture réactive (couches horizontales)
- architectures cognitives/réactives dans une architecture cognitive (couches verticales)

Couches horizontales



Problématiques :

Couches horizontales

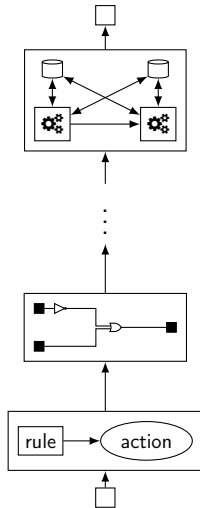


Problématiques :

- Quels modules ?
- Comment choisir quel module décide ?

Couches verticales

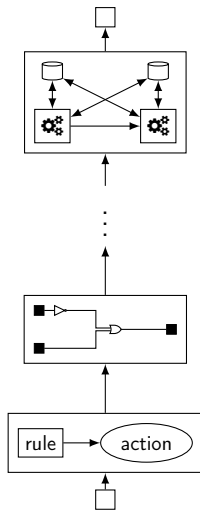
Architecture à une passe



Problématiques :

Couches verticales

Architecture à une passe

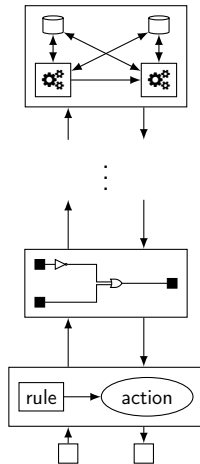


Problématiques :

- Quelles couches ?
- Que se passe-t-il en cas de problème dans une couche ?

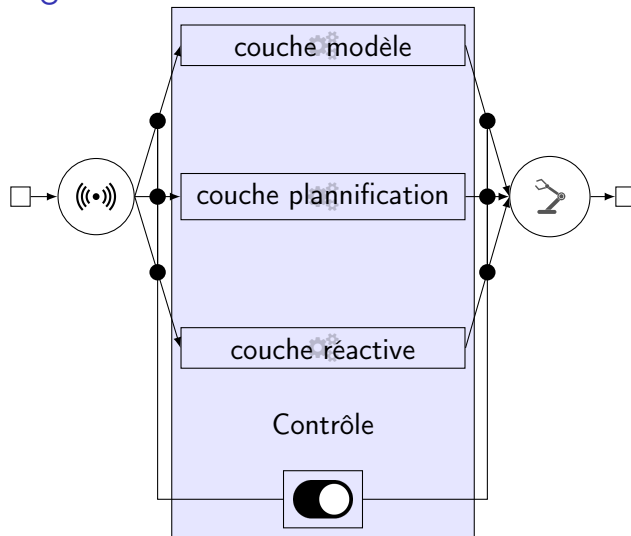
Couches verticales

Architecture à deux passes



Application

Machines de Turing



Application

Machines de Turing

couche réactive système de règles :

- règle \rightarrow action (pas de prédicat)
- pas de représentation du monde,
- pas d'accès à l'historique

couche planification exécution de plans :

- utilise une librairie de plans
- assemble des plans de différents niveaux pour obtenir un plan global
- détermine des plans à partir de buts

Application

Machines de Turing

couche modèle modélisation du monde

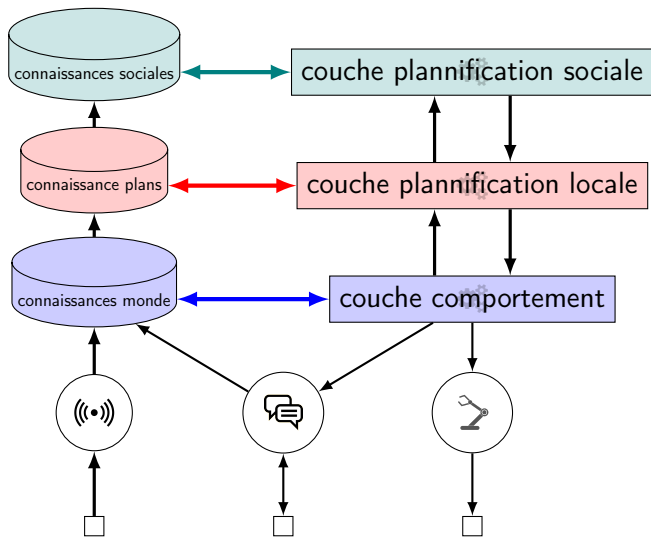
- représentation de l'agent, des autres agents etc.
- détecte les conflits
- génère de nouveaux buts et les soumet à la couche planif.

système de contrôle gère les entrées/sorties

- décide quelle couche agit,
- système à base de règles
- capable de supprimer une entrée ou censurer une sortie

Applications

InteRRaP



Applications

InterRaP

- chaque couche reliée à une base de connaissance pertinente
- couche composée de 2 fonctions :
 - ▶ reconnaissance de situation et activation de but : en fonction des bases de connaissances
 - ▶ planification : en utilisant les plans à sa disposition
- déroulement :
 - ▶ couche activée : utilisation des fonctionnalités fournies par la couche inférieure pour accomplir le plan
 - ▶ couche non activée : envoi de l'information à la couche supérieure
- responsabilité des couches :
 - couche comportementale comportement réactif vis à vis de l'environnement
 - couche planification locale comportement cognitif vis à vis de l'environnement
 - couche planification sociale comportement cognitif vis à vis des autres agents

- Comparaison des architectures hybrides

	architecture horizontale	architecture verticale
complexité	m^n	$m^2 \cdot (n - 1)$
tolérance aux fautes	✓	✗
concept	simple	complexe
exemple	InteRRaP	machines de Turing

Bilan



Carte des idées



Téléchargeable/clonable depuis

<https://gitlab.data-ensta.fr/buron/2020-2021-ia310-cours-5>

Références I

-  Yoav Shoham. “Agent-oriented programming”. *Artificial intelligence*, 60.1, pp.51-92, 1993.
-  Michael Fisher et Howard Barringer. “Concurrent METATEM processes—A language for distributed AI.” *In Proceedings of the European Simulation Multiconference*. 1991.
-  Innes A Ferguson. “Touring machines : Autonomous agents with attitudes”. *Computer*, 25.5, pp.51-55, 1992.
-  Jörg P Müller et Markus Pischel. “The agent architecture inteRRaP : Concept and application.” rapport de l’université de la Sarre, 1993.
-  R. Peter Bonasso, David Kortenkamp, and Troy Whitney. “Using a robot control architecture to automate space shuttle operations.” *In AAAI/IAAI*, pp. 949-956. 1997.