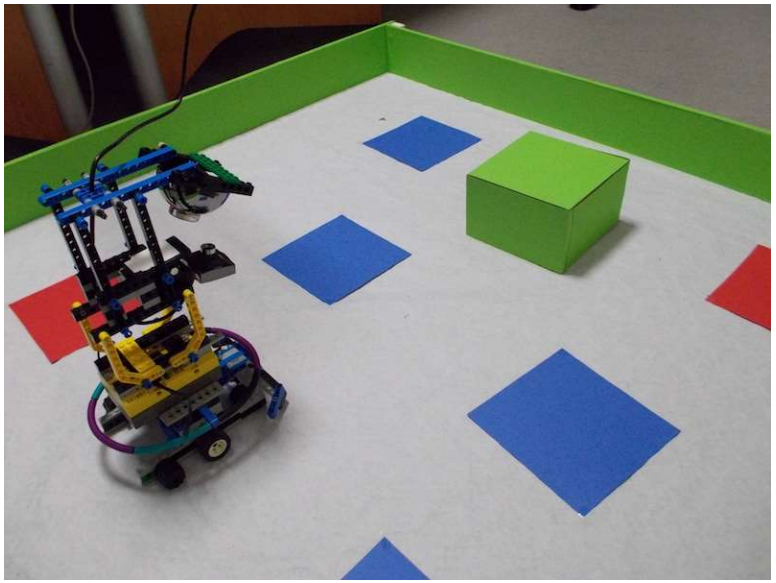
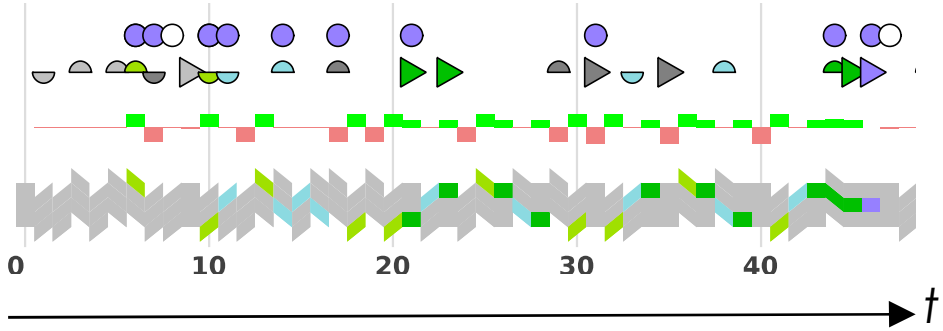
Intelligence Artificielle Développementale



26 Octobre 2022
Olivier.georgeon@gmail.com
http://www.oliviergeorgeon.com



Déroulement du cours

UE « IA et Cognition » (30h): Marie Lefevre

Module « IA Développementale » (12h) Olivier Georgeon

- Vendredi 14 octobre: 3h
- Vendredi 21 octobre: 3h
- **Mercredi 26 octobre: 3h**
- Vendredi 28 octobre: 3h

- Contrôle des connaissances:
  - TD par groupe de 2: 40% de le note de contrôle continu
  - Examen final: 7 points sur 20

Objectifs pédagogiques

Après ce cours, vous serez capables de:

- Cours
  - Expliquer ce qu'est l'IA développementale
  - Différencier IA en domaine modélisé / non modélisé
  - Nommer quelques auteurs de référence dans ce domaine
- TD
  - Implémenter un agent minimaliste dans lequel on ne code pas a priori une ontologie du "monde".

Séance 3: Plan

- **Séance précédente**
  - Cognition et déterminisme
- **Intelligence artificielle dans un domaine non modélisé a priori**
  - Auto-programmation
- **Travaux pratiques**
  - Agent 1
  - Agent 2
  - Agent 3
  - Agent 4

Déterminisme et cognition

L'humain est peut être déterministe mais néanmoins libre

- *Théorie évolutionniste de la liberté* (Dennett 2003).

Système déterministe peut être imprédictible.

- Inutile d'utiliser la Fonction Random()
  pour générer des comportements imprédictibles.
- Hervé Swirn

Un système déterministe peut « s'individualiser »

- En fonction des conditions initiales
- En fonction d'expériences individuelles
- *Autonomie constitutive* (Froese & Ziemke 2009).

Emergence de « macro-propriétés »

- Souvent non démontrable mais observable depuis un niveau d'observation supérieur

Auto-programmation

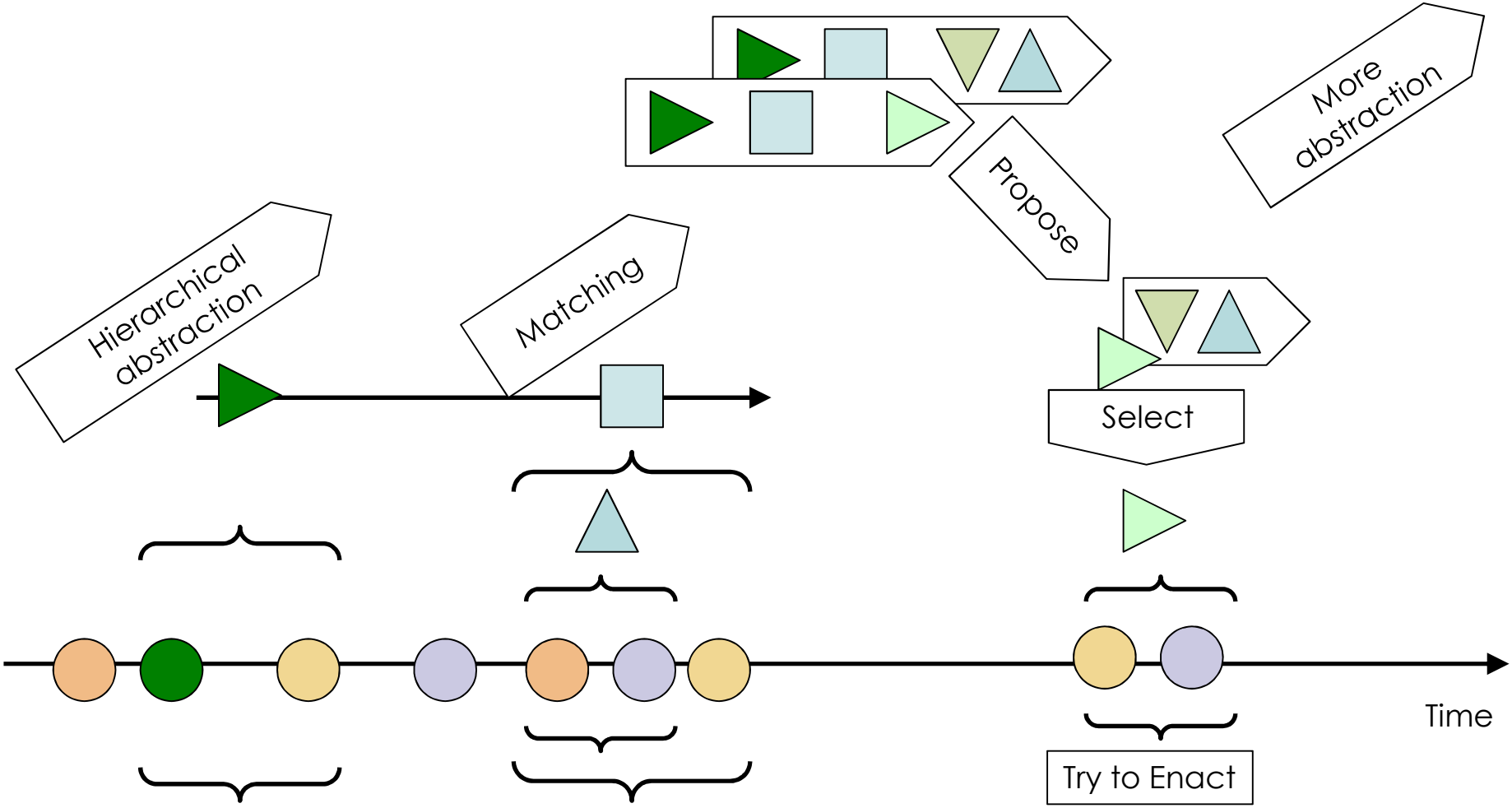
Auto programmation

Un système auto-programmant est un système capable d'apprendre du code qu'il peut ré-exécuter.

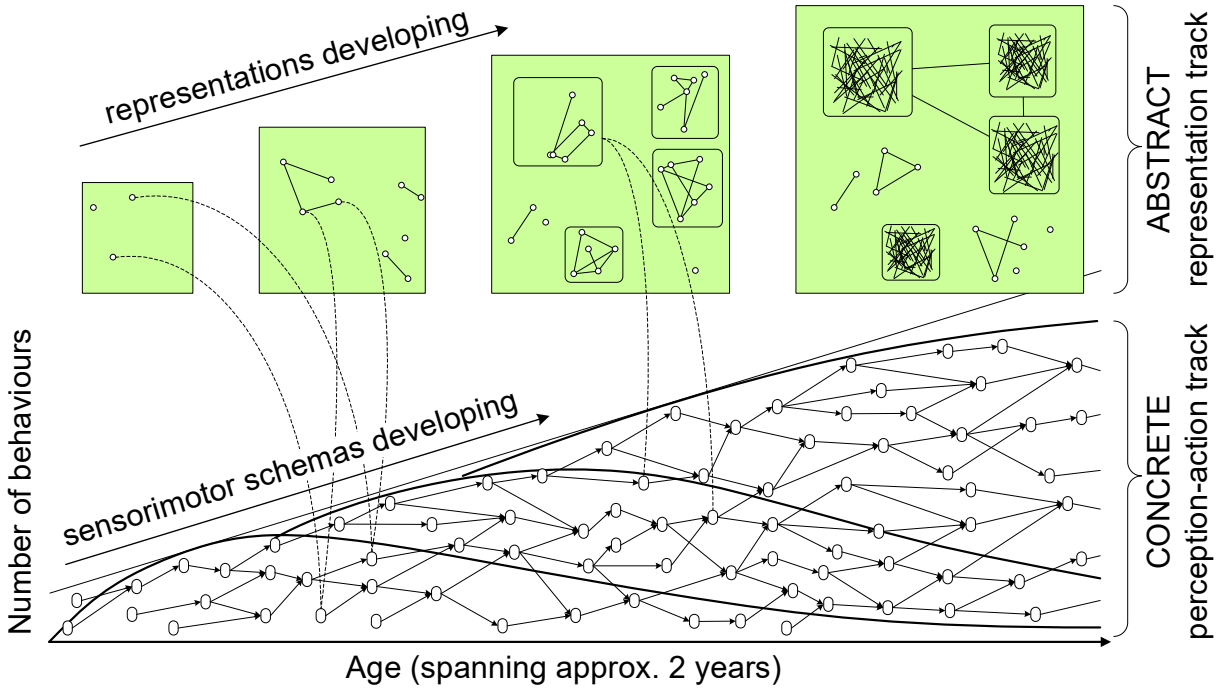
Cela pose de nombreuses questions:

- Quel jeu d'instructions?
- Quel moteur d'exécution?
- Quelle finalité?
  - Pourquoi apprendre un programme plutôt qu'un autre?

Apprentissage de régularités hierarchique



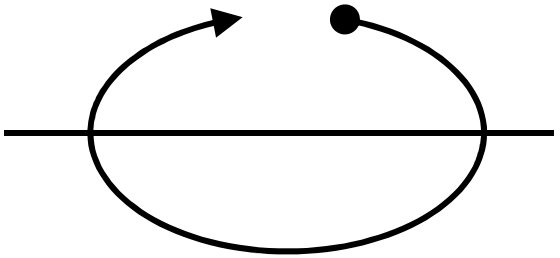
Construire des connaissances à partir de régularités d'interactions (schèmes sensorimteurs)



Conceptual diagram of developmental learning (Guerin, Krüger, and Kraft, 2013)

Couplage structurel

On parle de couplage structurel chaque fois que survient une histoire d'interactions récurrentes responsables d'une congruence structurelle entre deux systèmes ou plus.
(Maturana & Varela,1989, p. 78)

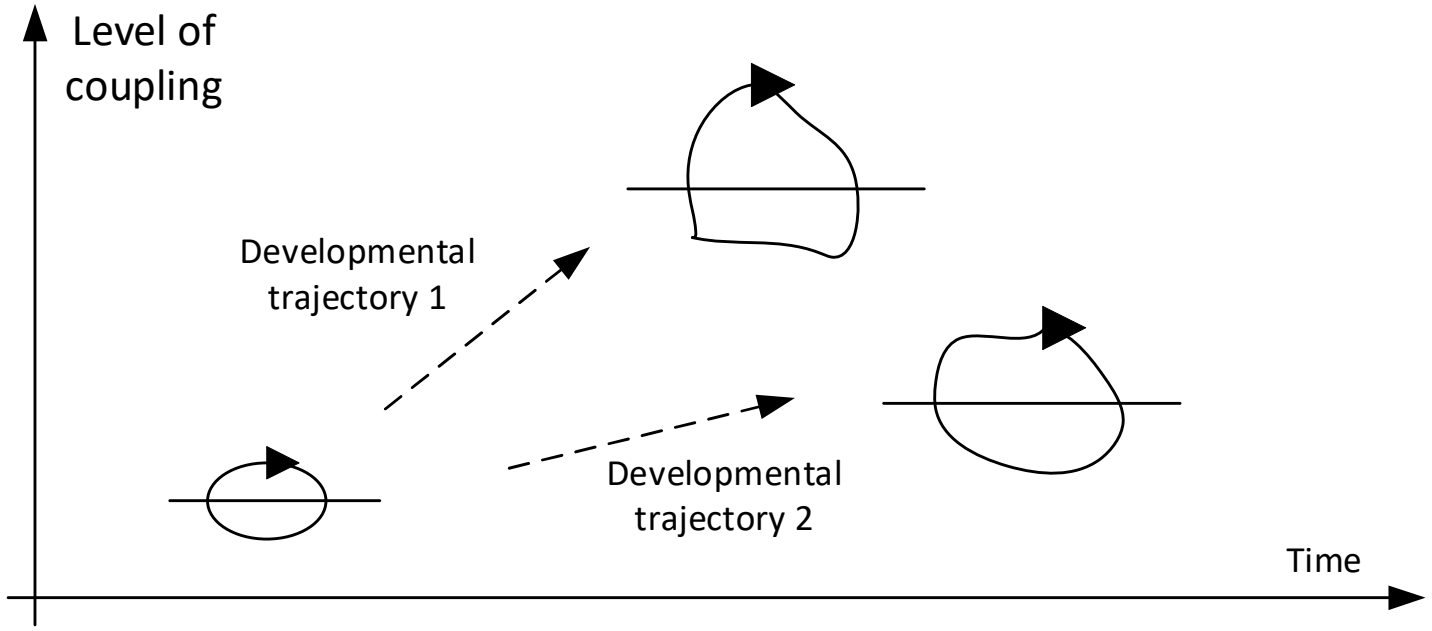


Couplage

Sujet
/
Objet

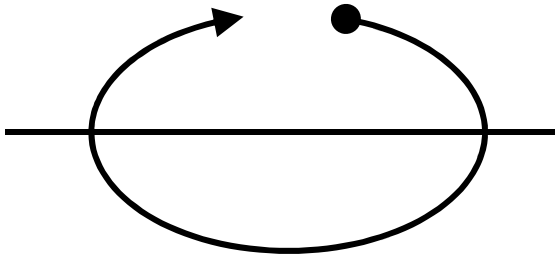
Apprentissage développemental

Capacité d'un sujet a développer individuellement sa structure interne et son couplage avec son environnement. (Théories de Piaget).



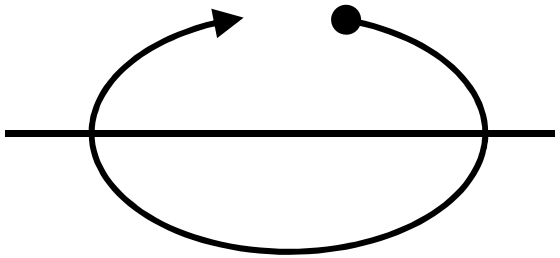
Constitutive autonomy

Dans un système artificiel: dissocier « couplage cognitif » vs « couplage physique »



Decisional mechanism
/
Software Interface

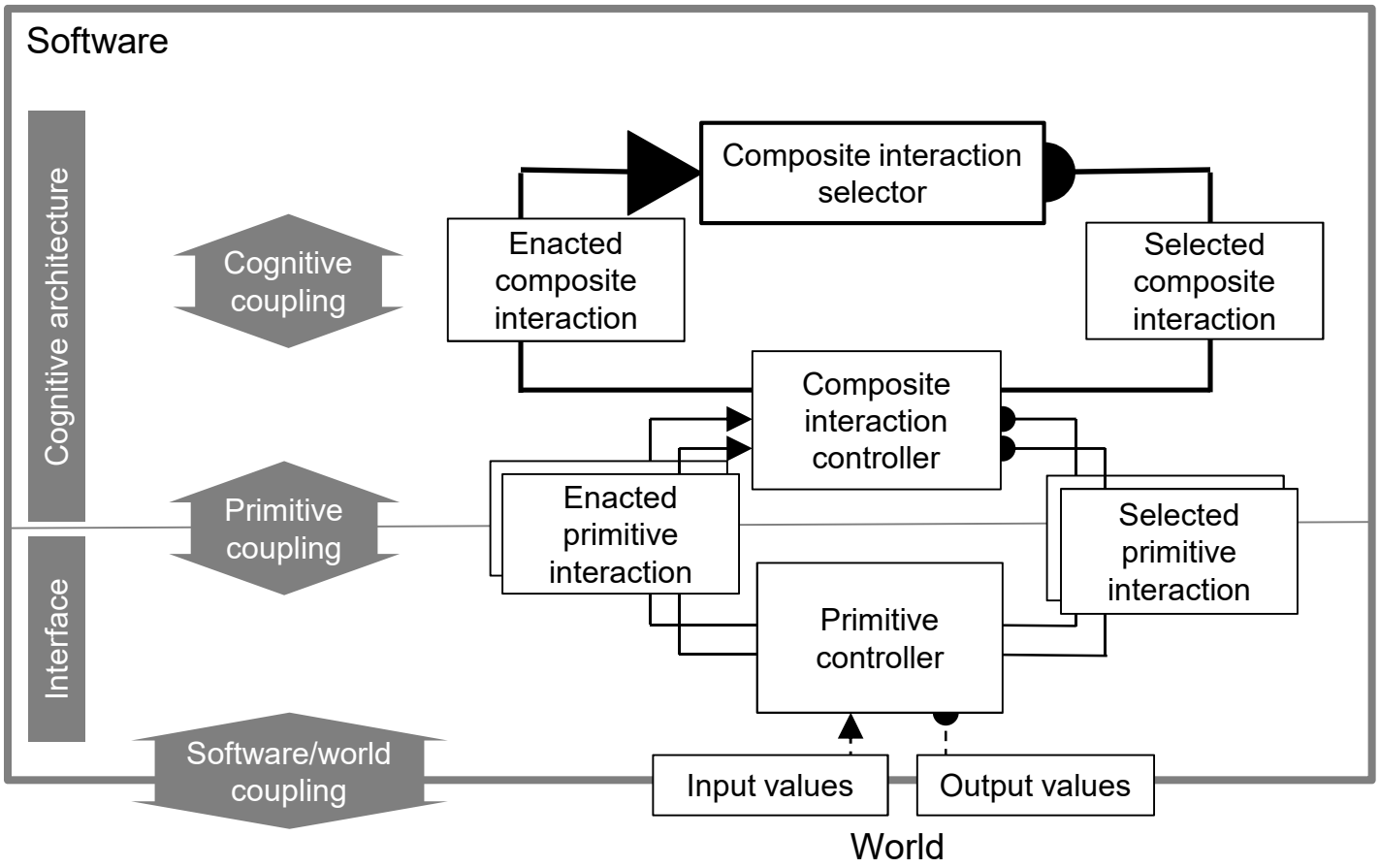
Evolue



Software
/
World

Fixe

Evolution du couplage cognitif



Georgeon O., Riegler A. (2019). Constitutive Autonomy through Motorsensory Self-Programming. Cognitive Systems Research, 58: 366-374.

Travaux dirigés

Séance 3

Setup

Suivre la procédure écrite
ici https://github.com/OlivierGeorgeon/TestROS/wiki/Implementer-un-agent-rudimentaire

Créer un nouveau projet python dans votre environnement de développement Python favori (par exemple Pycharm) contenant le fichier world.py. Vous avez deux méthode possibles :

- Cloner le repository https://github.com/OlivierGeorgeon/TestROS
- Créer un nouveau projet et copier le fichiers world.py

Exécuter world.py et vérifiez que vous obtenez la trace d'interaction montrée en Figure 1
sur https://github.com/OlivierGeorgeon/TestROS/wiki/Implementer-un-agent-rudimentaire)

Agent 1

Dans le fichier world.py, modifier la class Agent pour créer l'Agent 1 en suivant les instructions :

https://github.com/OlivierGeorgeon/TestROS/wiki/Agent-1

Tester votre agent dans Environment1 puis dans Environment2 en commentant et décommentant les lignes appropriés (lignes 70 et 71 dans le fichier world.py initial)

Agent 2

Créér l'Agent2 en suivant les instructions :

https://github.com/OlivierGeorgeon/TestROS/wiki/Agent-2

Tester votre agent dans Environment1 puis dans Environment2 en commentant et décommentant les lignes appropriés (lignes 70 et 71 dans le fichier world.py initial)

Modifier la table des valences d'interaction.

Agent 3

Créer l’Agent3 en suivant les instructions :

https://github.com/OlivierGeorgeon/TestROS/wiki/Agent-3

Tester votre agent dans l’environnement TurtlePy

Agent 4

Créer l'Agent4 en suivant les instructions :

https://github.com/OlivierGeorgeon/TestROS/wiki/Agent-4

Tester votre agent dans les environnements 1, 2 et 3 et montrer qu'il sais s'adapter à chacun de ces trois environnements.