Artificial intelligence, Neural Networks and Categorization

Pierre Andry - Alexandre Pitti
Université Cergy-Pontoise
pierre.andry@ensea.fr
ETIS UMR CNRS 8051

Overview

former «Classical» Artificial intelligence

(until the 80's)

- from advanced algorithmics...
- ...to space state search (A*)
- heuristics, and game basics (min-max)

Neural Networks and Categorization

(since the 80's)

- symbol grounding problem
- inspiration from biology
- formal model of a Neuron
- supervised learning
- unsupervised learning

Global introduction

Traditional opposition : *classical IA* / *connectionism*

Global introduction

Traditional opposition : *classical IA* / *connectionism*

- Logics (inference, deduction, etc...)
- Reasoning
- Symbol manipulation
- Solution discovery
- Rules (rules based system, expert systems)
- Advanced Algorithmics
- State space search (depth first, A*, ...)

Global introduction

Traditional opposition : *classical IA* / *connectionism*

- Logics (inference, deduction, etc...)
- Reasoning
  - LISP [MIT,McCarthy,58]
  - Prolog [Colmerauer,60]
- Symbol manipulation
- Solution discovery
- Rules (rules based system, expert systems)
- Advanced Algorithmics
- State space search (depth first, A*, ...)

Global introduction

Traditional opposition : *classical IA* / *connectionism*

- Logics (inference, deduction, etc...)
- Reasoning
  - "computers can be used to manipulate symbols" [McCarthy 60]
- Symbol manipulation
  - "dialog with a computer ELIZA [Weiezbaum, 60]
- Solution discovery
- Rules (rules based system, expert systems)
- Advanced Algorithmics
- State space search (depth first, A*, ...)

Global introduction

Traditional opposition : *classical IA* / *connectionism*

- Logics (inference, deduction, etc...)
- Reasoning
- Symbol manipulation
  - 80's
  - Scripts language
  - PLANNER : goal generation for program solving
- Solution discovery
  - GPS [Newel & Simon, 78]
- Rules (rules based system, expert systems)
- Advanced Algorithmics
- State space search (depth first, A*, ...)

Global introduction

One example : let's try to define an object : a chair

- Let's do it classical :
  - *object (legs, 4) and object(back,1) and object (seat,1)*

Global introduction

One example : let's try to define an object : a chair

- Let's do it classical :
  - *(object (legs, 4) or (object (legs,3)) and object(back,1) and object (seat,1)...*

Global introduction

One example : let's try to define an object : a chair

- Let's do it classical :
  - *(object (legs, 4) or (object (legs,3)) ~~and object(back,1)~~ and object (seat,1)...*

Global introduction

One example : let's try to define an object : a chair

- Let's do it classical :
  - *(object (legs, 4) or (object (legs,3)) ~~and object(back,1)~~ and object (seat,1)...*
  - *well...*

Global introduction

One example : let's try to define an object : a chair

- Let's do it classical :
  - *(object (legs, 4) or (object (legs,3)) ~~and object(back,1)~~ and object (seat,1)...*
  - *well...*
  - *..or (object, legs, 1) ?*

Global introduction

One example : let's try to define an object : a chair

(one of the world most known chair : Eames rocking chair)

- Let's do it classical :
  - *(object (legs, 4) or (object (legs,3)) ~~and object(back,1)~~ and object (seat,1)...*
  - *well...*
  - *..or (object, legs, 1) ?*
  - *..or 2 ?*

Global introduction

One example : let's try to define an object : a chair

- Let's do it classical :
  - *(object (legs, 4) or (object (legs,3)) ~~and object(back,1)~~ and object (seat,1)...*
  - *well...*
  - *..or (object, legs, 1) ?*
  - *..or 2 ? .....?*

Global introduction

One example : let's try to define an object : a chair

- Let's do it classical :
  - ~~*(object (legs, 4) or (object (legs,3)) and object(back,1) and object (seat,1)...*~~
  - ~~*well...*~~
  - ~~*..or (object, legs, 1) ?*~~
  - ~~*..or 2 ? .....?*~~
  - *....?*

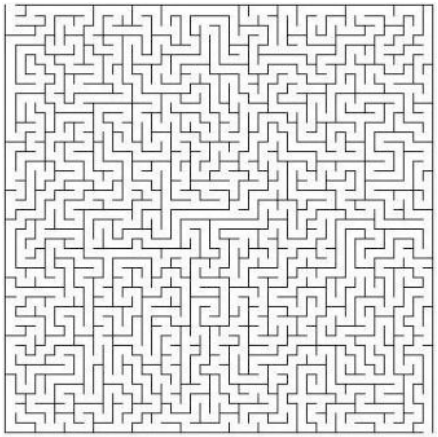
Global introduction

Traditional opposition : *classical IA* / *connectionism*

- Logics (inference, deduction, etc...)
- Reasoning
- Symbol manipulation
- Solution discovery
- Rules (rules based system, expert systems)
- Advanced Algorithmics
- State space search (depth first, A*, ...)

- A star [Hart & Al. 68]

State space search

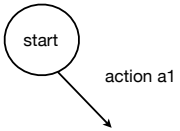


State space search

- let us consider space state
- Solving a problem in this space is :
    - starting from a particular state
    - applying operations (or actions)
    - find the solution state

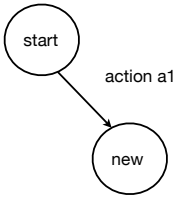
State space search

- let us consider space state
- Solving a problem in this space is :
    - starting from a particular state
    - applying operations (or actions)
    - find the solution state



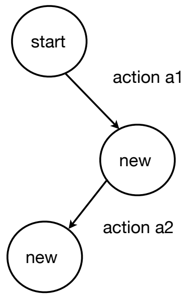
State space search

- let us consider space state
- Solving a problem in this space is :
    - starting from a particular state
    - applying operations (or actions)
    - find the solution state



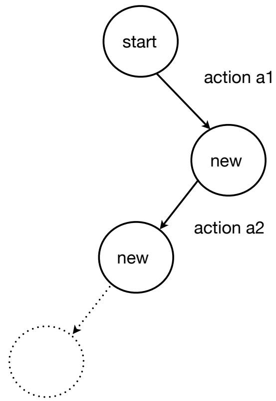
State space search

- let us consider space state
- Solving a problem in this space is :
    - starting from a particular state
    - applying operations (or actions)
    - find the solution state



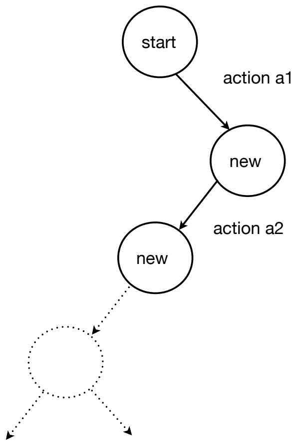
State space search

- let us consider space state
- Solving a problem in this space is :
    - starting from a particular state
    - applying operations (or actions)
    - find the solution state



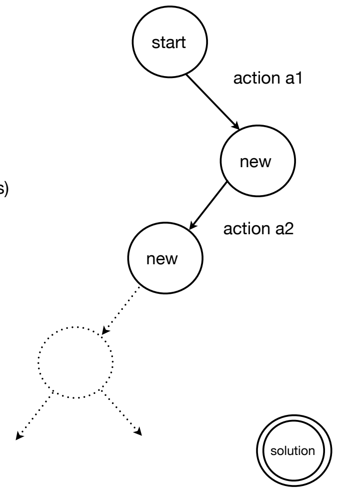
State space search

- let us consider space state
- Solving a problem in this space is :
    - starting from a particular state
    - applying operations (or actions)
    - find the solution state



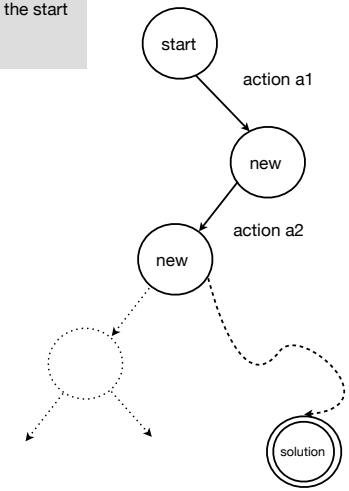
State space search

- let us consider space state
- Solving a problem in this space is :
    - starting from a particular state
    - applying operations (or actions)
    - find the solution state



State space search

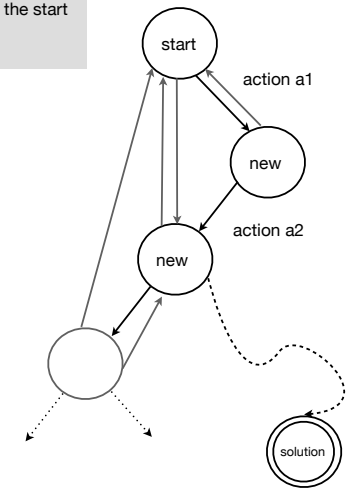
Finding a solution with an AI = find the path from the start state to the solution state



State space search

Finding a solution with an AI = find the path from the start state to the solution state

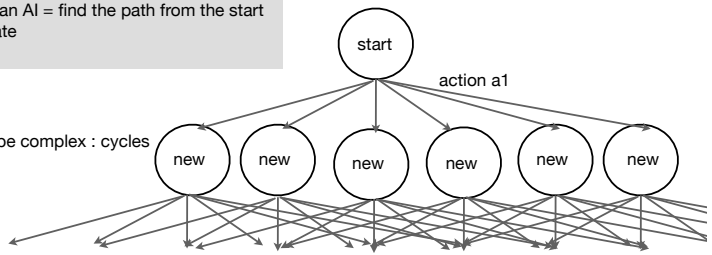
but....the problem can be complex : cycles



State space search

Finding a solution with an AI = find the path from the start state to the solution state

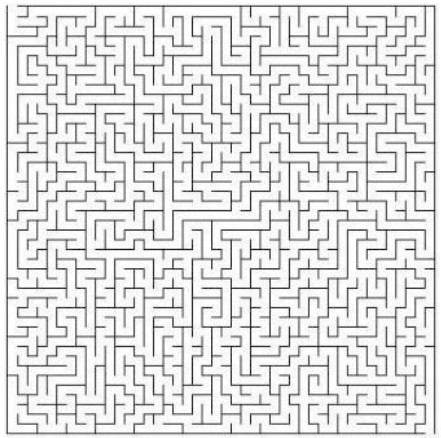
but....the problem can be complex : cycles



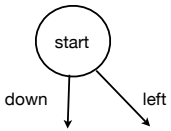
much more complex : combinatory explosion

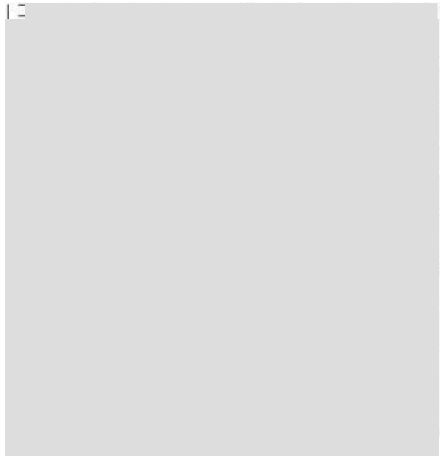
much more complex : combinatory explosion

State space search : your turn



What is your solution ?





Algorithm 1 : choose random action

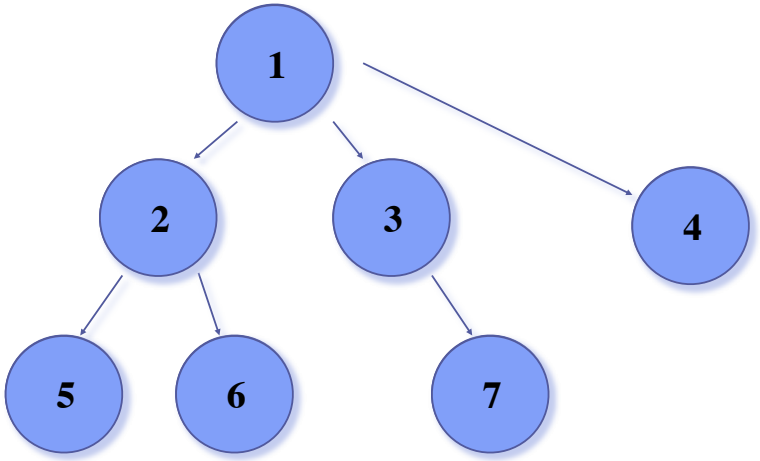
- simple implementation
- need no *a priori* information
- safe :
    - we know that we will find the solution...(large numbers : sampling randomly = you will sample every state)
    - ...but yo don't know when..
- problem : convergence time

we are sure to find the solution state in a finite time, but this time may tend asymptotically toward infinite.

Algorithm 2 and 3 : systematic exploration

- depth first search G(S,A)
- breadth first search G(S,A)

breadth-first search

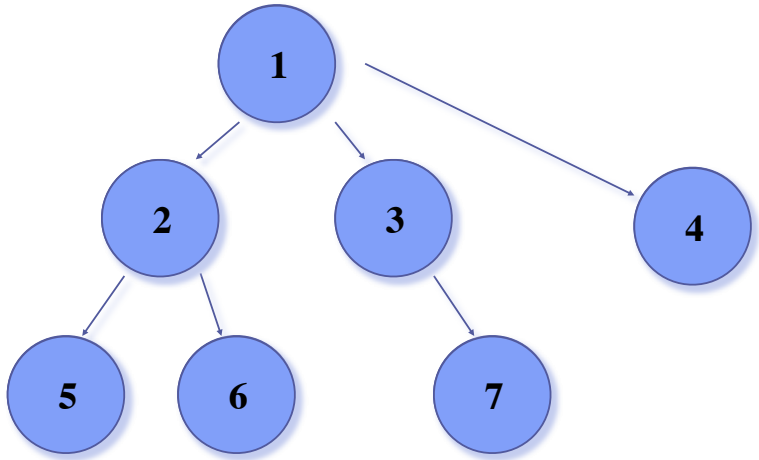


breadth-first search

- Let us suppose G(S,A) a Graph (or tree) with S a set of vertices and A a set of edges
- OPEN is a queue (FIFO), CLOSED is a set/list/container

1. Put (enqueue) the Start state s0 in OPEN
2. while OPEN is not empty
    n is the first element of OPEN
    enqueue the sons of n in OPEN
    process n
    put n in CLOSED

depth-first search



depth-first search

- Let us suppose G(S,A) a Graph (or tree) with S a set of vertices and A a set of edges
- OPEN is a stack (LIFO), CLOSED is a set/list/container

1. Put (push) the Start state s0 in OPEN
2. While OPEN is not empty and depth < depth_max
    n is the first element of OPEN
    pop n
    push the sons of n in OPEN
    process n
    put n in CLOSED

Algorithm 1 : choose random action

- breadth-first, depth-first : this is not AI.
- advanced algorithms applied to trees and graphs
- problem : convergence time

we are sure to find the solution state in a finite time, but the convergence highly depends on the position of the solution state in the graph. complexity is O (S+A), worst case.

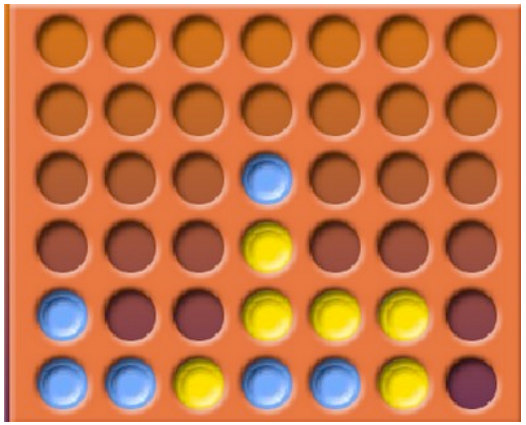
State space search : A*

- To speed-up the search : find a strategy
- a heuristic : a non optimal function nevertheless sufficient to solve the problem
- i.e : an approximation of the real theoretical (but unknown) optimal function leading to the solution
    - often : external information, different modality,
    - notion of «distance» to the solution
    - specific or different metrics

All the «intelligence» of the algorithm is linked to the heuritic : convergence time

What is your solution ?

connect 4 : yellow plays, who wins ?



State space search : A*

- A* : depends on f : cost function
- our algorithm has to minimize the cost function f(n) calculated at node n
- f is composite : f(n) = g(n) + h(n)
- g(n) is the **calculated** cost of the path from start to n
- h(n) is the **estimated** cost of the path from n to the solution state

State space search : A*

- A* : depends on f : cost function
- our algorithm has to minimize the cost function f(n) calculated at node n
- f is composite : f(n) = g(n) + h(n)
- g(n) is the **calculated** cost of the path from start to n
- h(n) is the **estimated** cost of the path from n to the solution state

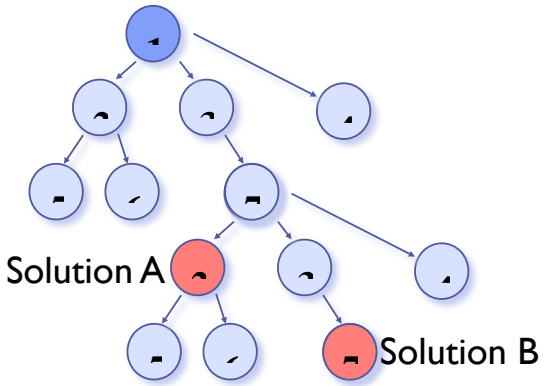
not the same metrics, not the same rules !

The minimization of f(n) will ensure to find a solution (if it exists) and the shortest «path» according to g and h.

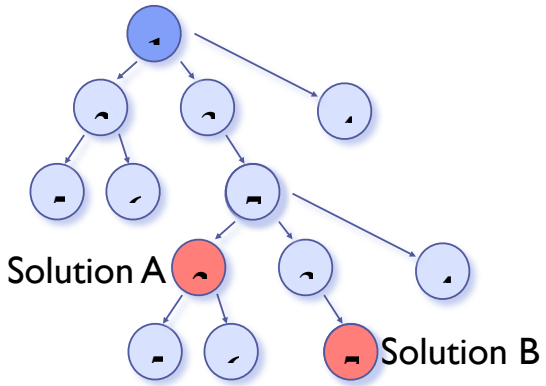
State space search : A*

- f(n) = g(n) + h(n)
- g(n) must be the smallest as possible
- h must be the best approximation as possible ; if h(n) is small then a good solution will be found

State space search : A*

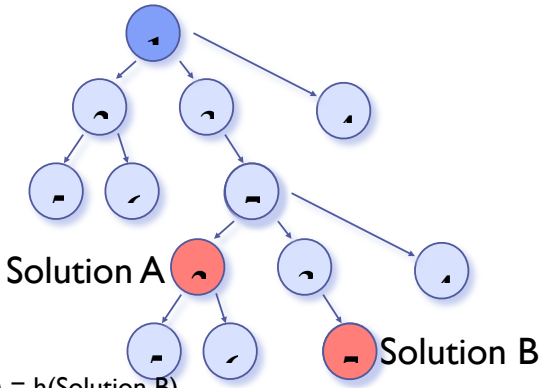


State space search : A*



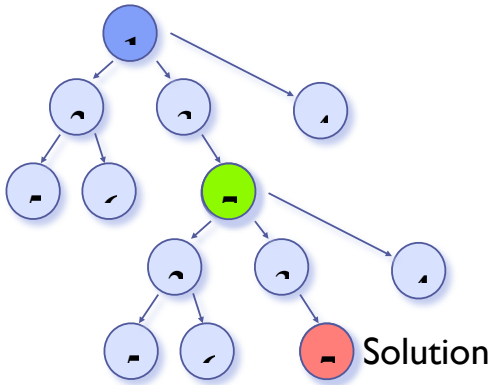
h(Solution A) = h(Solution B)

State space search : A*

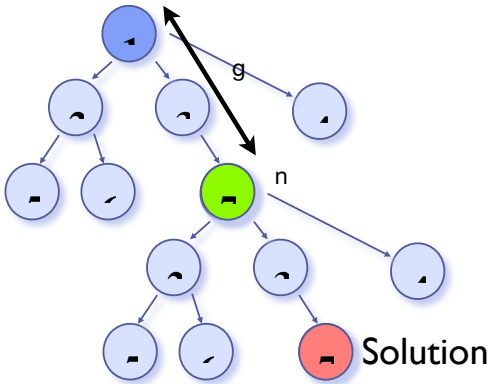


h(Solution A) = h(Solution B)
g(Solution A) < g(SolutionB)
donc fA<fB : A est une meilleure solution que B

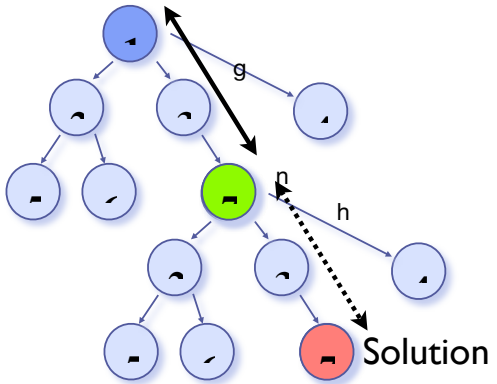
State space search : A*



State space search : A*



State space search : A*



State space search : A*

1. Put the initial state n0 in OPEN; f(n0)=0; g(n0)=0

2. If OPEN is empty => there is no solution

3.**select the best n (according f(n)) in OPEN and put it in CLOSED**

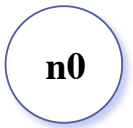
4. for each successors of n; if none is the solution :

   - apply g(successor) = g(n)+1

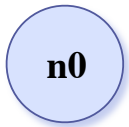
   - if the successor is in CLOSED do nothing

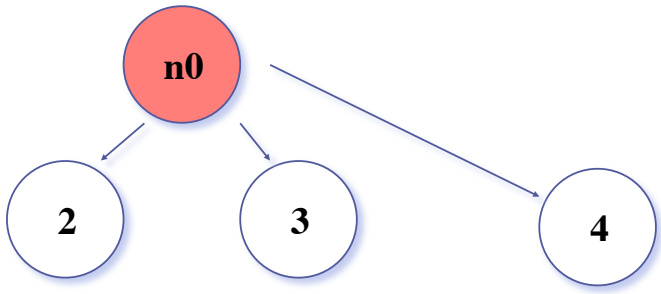
   - if the successor is in OPEN enqueue the best instance in OPEN
     else enqueue the successor in OPEN

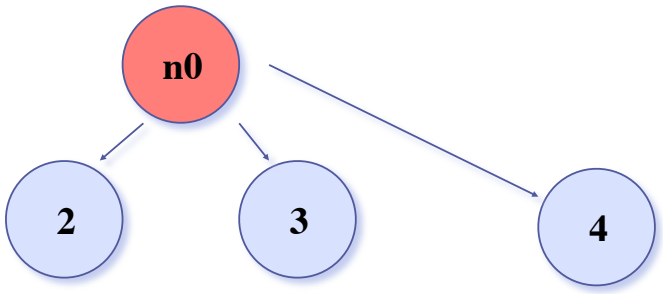
5. goto 2

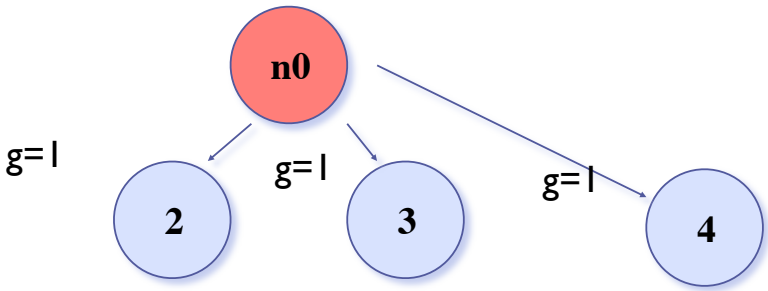


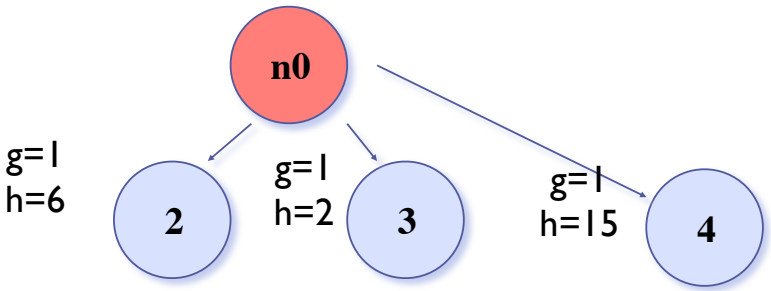
White : discovered
Blue : in OPEN
Red: in CLOSED

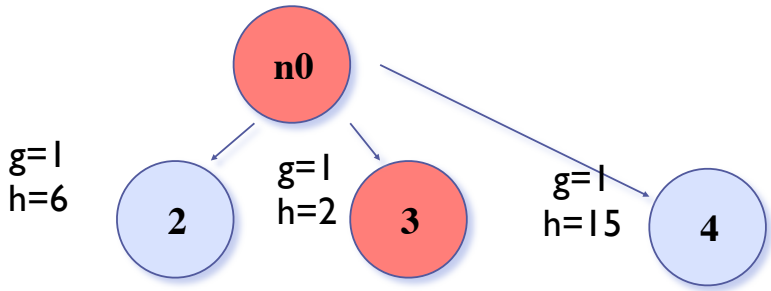


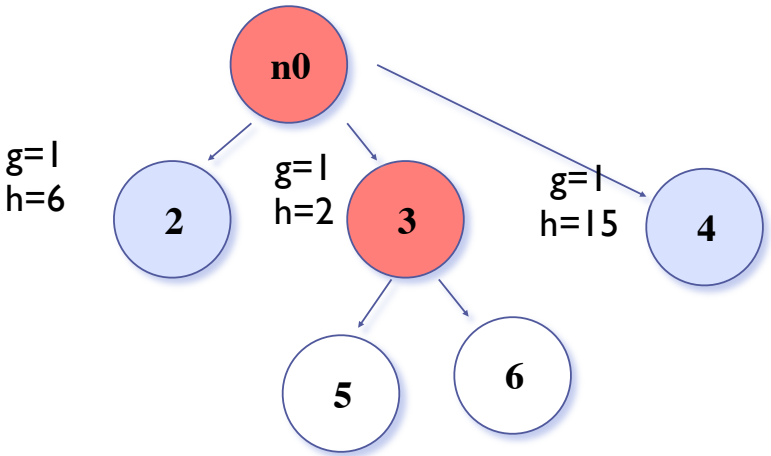

n0
2
3
4

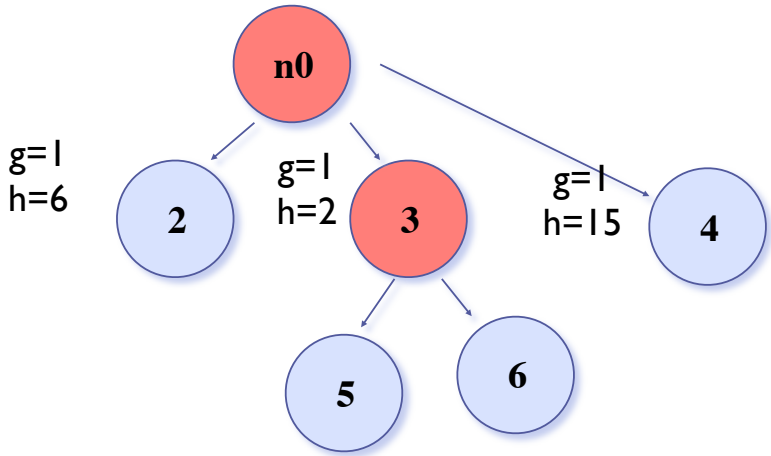

n0
2
3
4

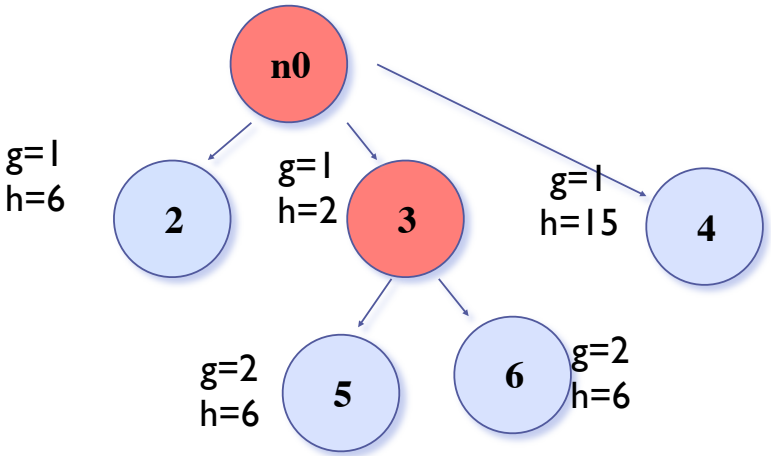

g=1
n0
g=1
2
g=1
3
4

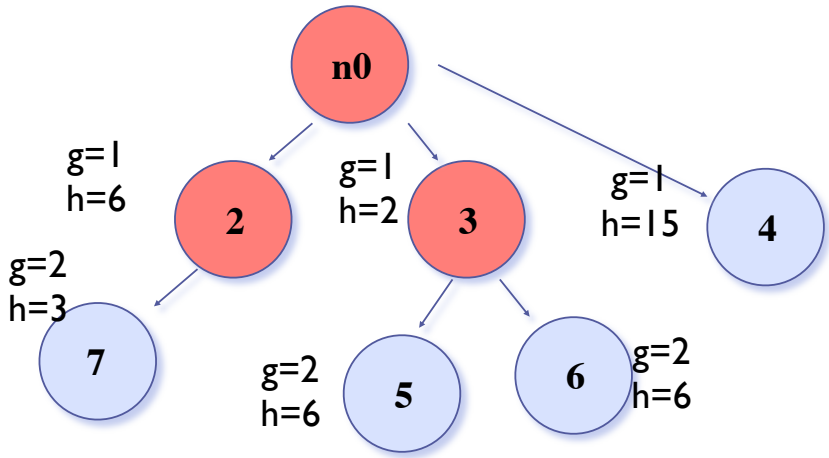

g=1
h=6
n0
g=1
h=2
2
g=1
h=15
3
4

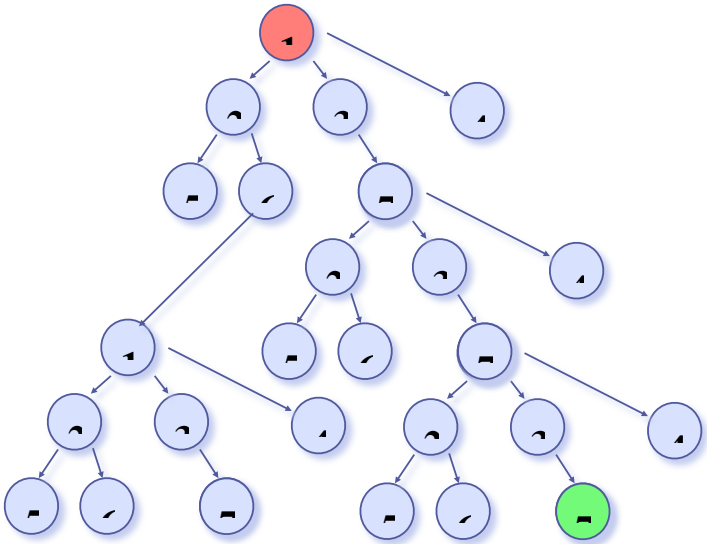

n0
g=1
h=6
2
g=1
h=2
3
g=1
h=15
4

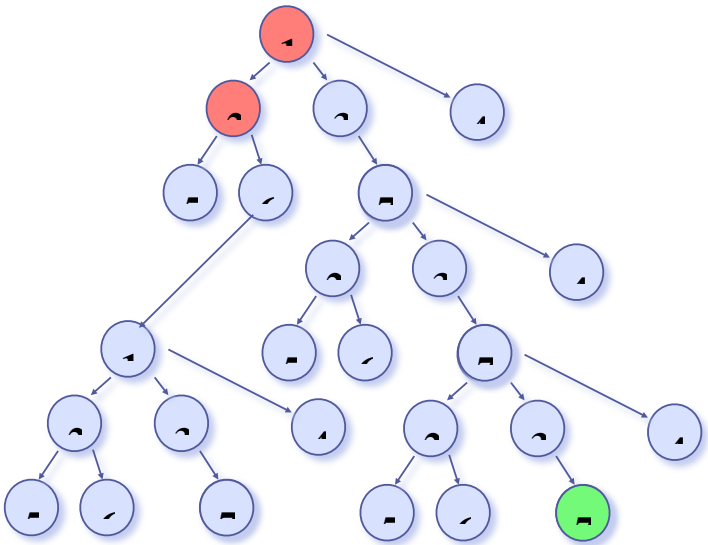

n0
g=1
h=6
2
g=1
h=2
3
g=1
h=15
4
5
6

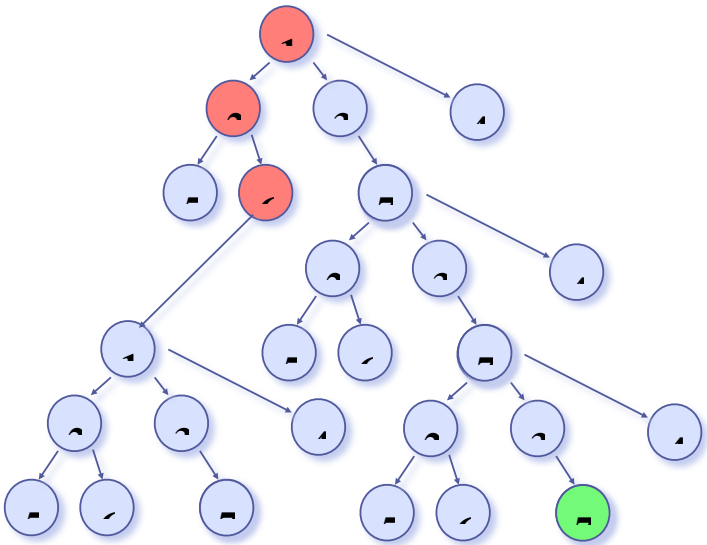

n0
g=1
h=6
2
g=1
h=2
3
g=1
h=15
4
5
6

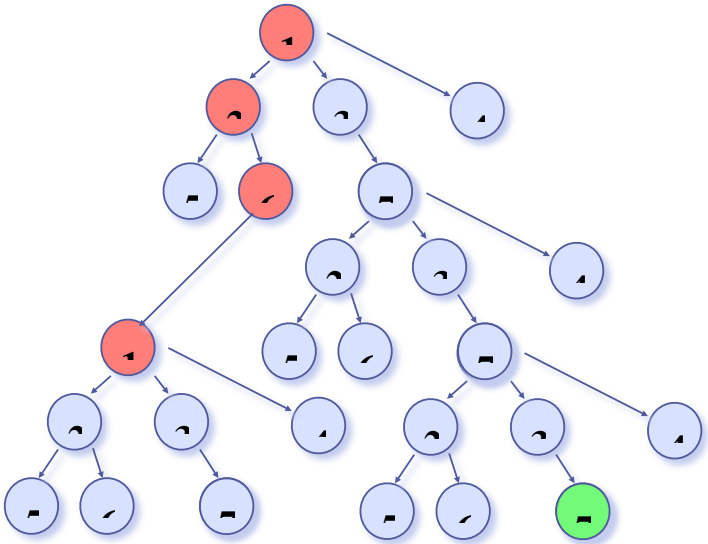

n0
g=1
h=6
2
g=1
h=2
3
g=1
h=15
4
g=2
h=6
5
6
g=2
h=6

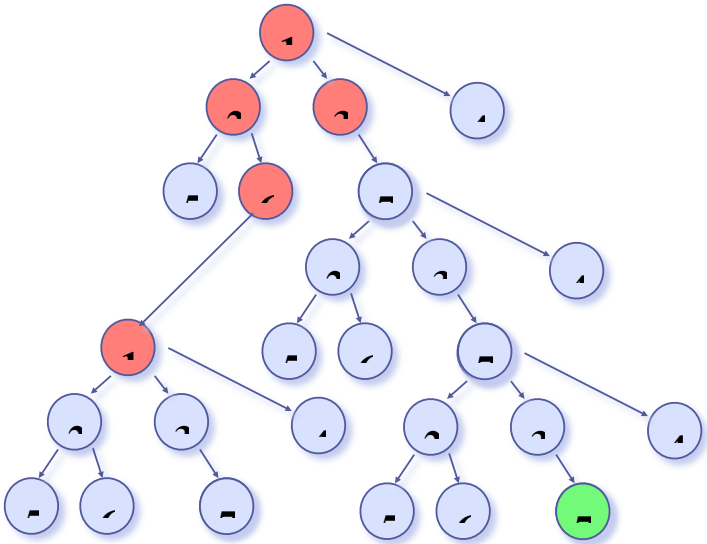

n0
g=1
h=6
2
g=1
h=2
3
g=1
h=15
4
g=2
h=3
7
g=2
h=6
5
6
g=2
h=6

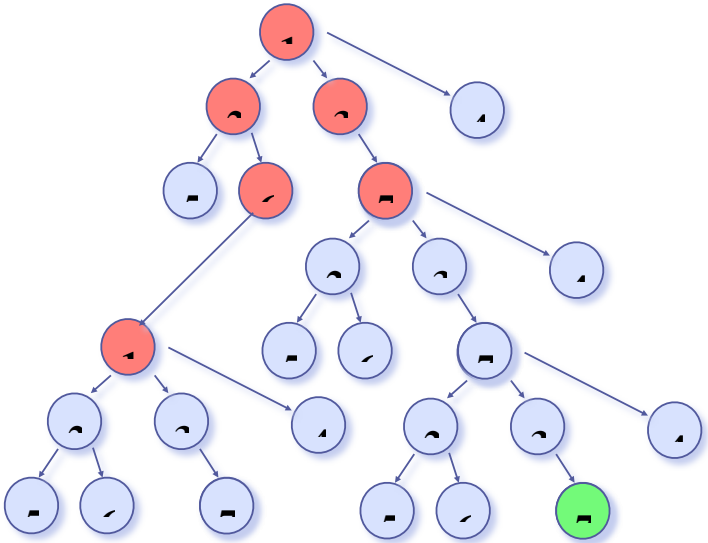


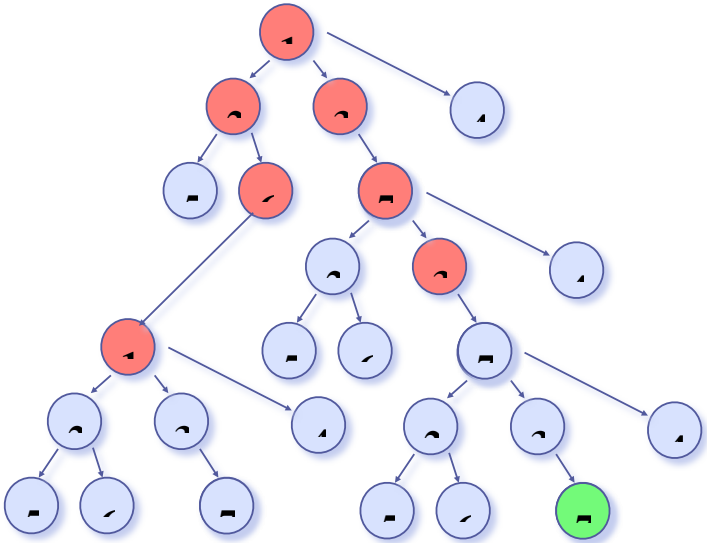


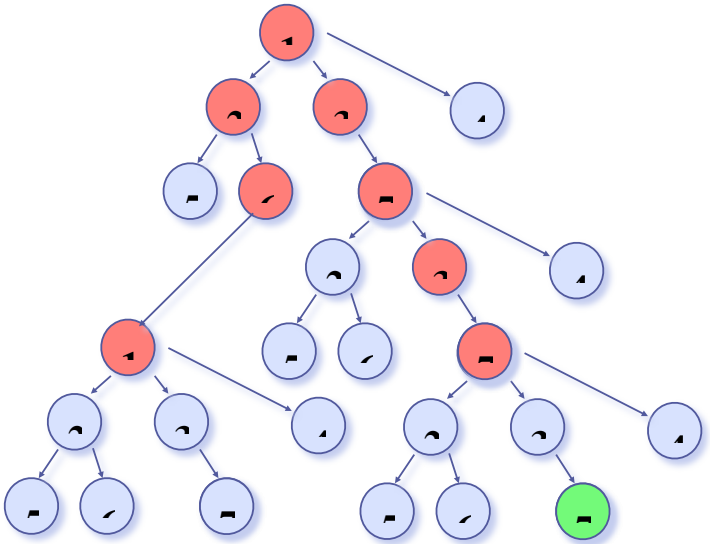


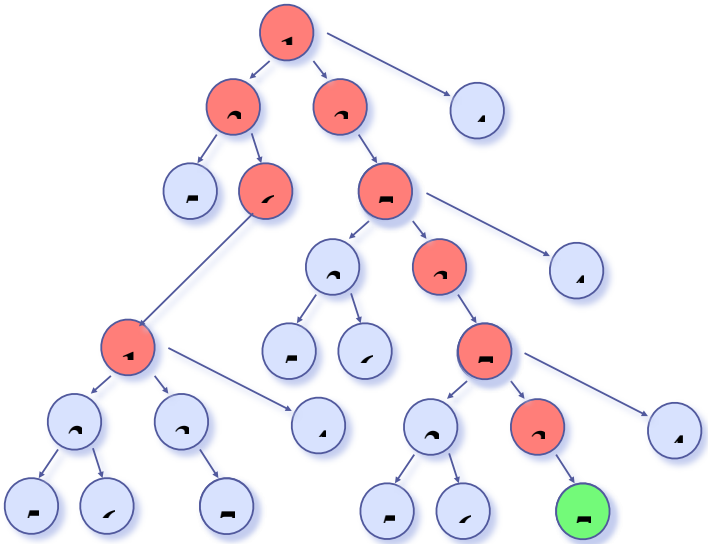


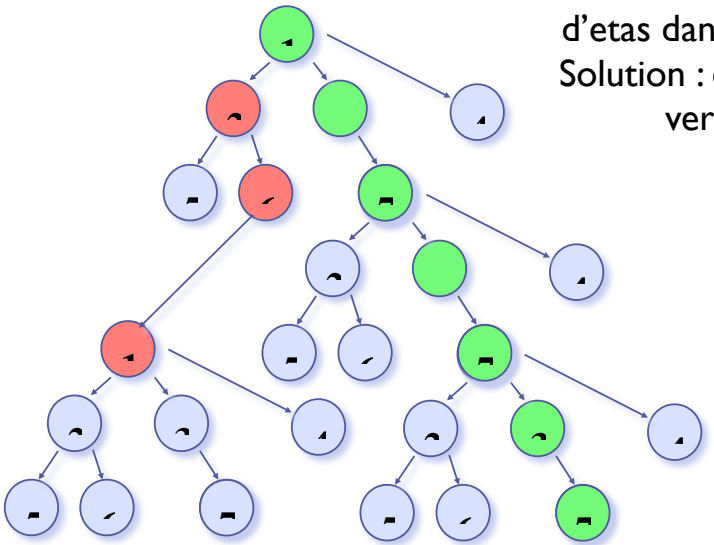










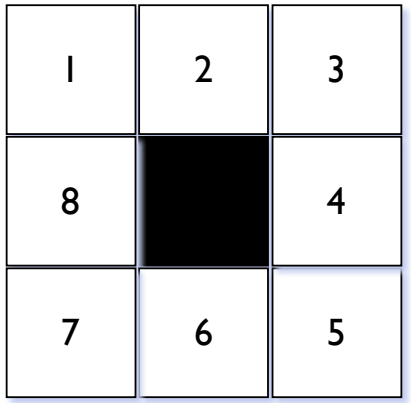


Etats visités : nb d'etas dans fermé
Solution : etats en vert

2	8	3
1	6	4
7		5

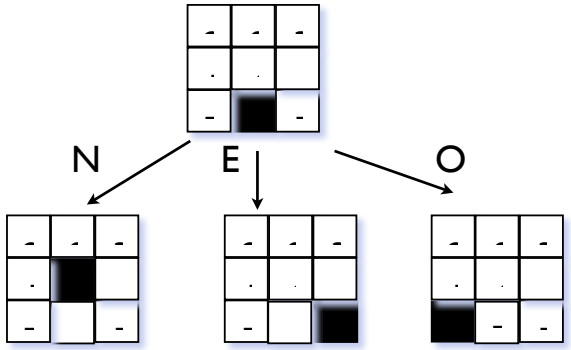
Etat de départ

objet etat_taquin
tableau de 9 valeurs
le 0 désigne le trou



Etat Solution

on cherche donc la
suite des coups a jouer
pour aller de l'état
de départ vers l'etat
solution



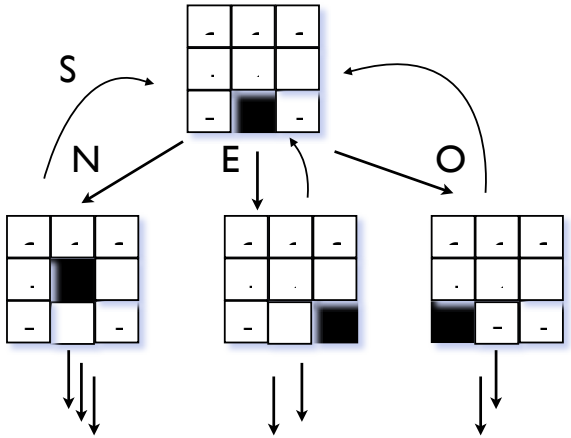
Etat de départ

generer_coup_suivant (etat_taquin etat_depart)
va nous retourner la liste des coups possible a partir de l'etat de départ
(attention au clonage d'un objet en java)

OUVERT et FERME vont donc être des liste d'éléments etat_taquin

chaque état contiendra un champ h, dédié au calcul de l'heuristique

ATTENTION, nous ne sommes plus dans un arbre, mais dans un graphe !



Etat de départ

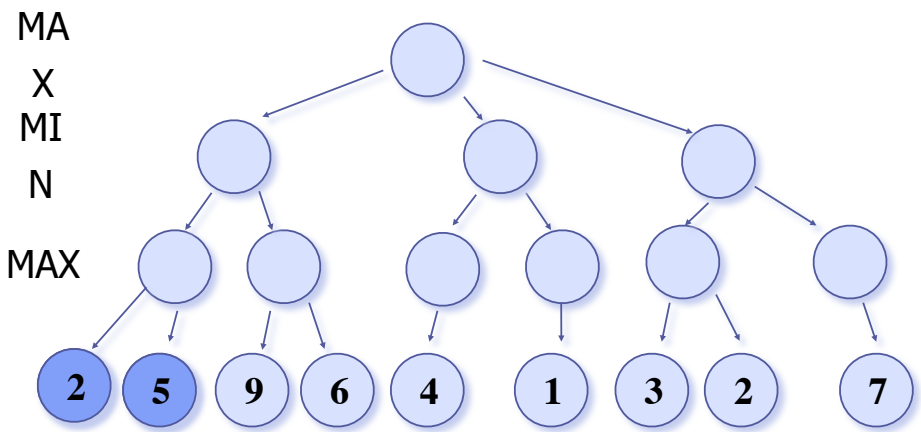
Application de A* aux graphes : cf algo du TP

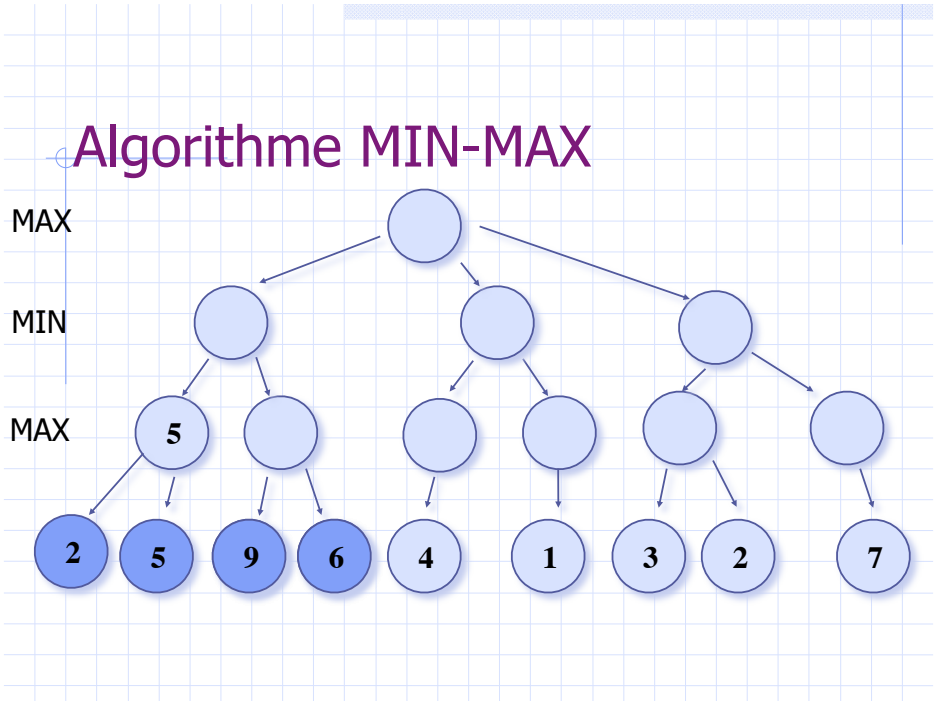
Deux cas nouveaux :

- Un nouvel état généré est déjà dans fermé :"on tourne en rond" ; on doit théoriquement réévaluer ce nouvel état (cas de jeux/problèmes ou l'on doit repasser
plusieurs fois par le même état pour trouver la solution)
Pour nous, la mesure g nous évitera de considérer le nouvel état

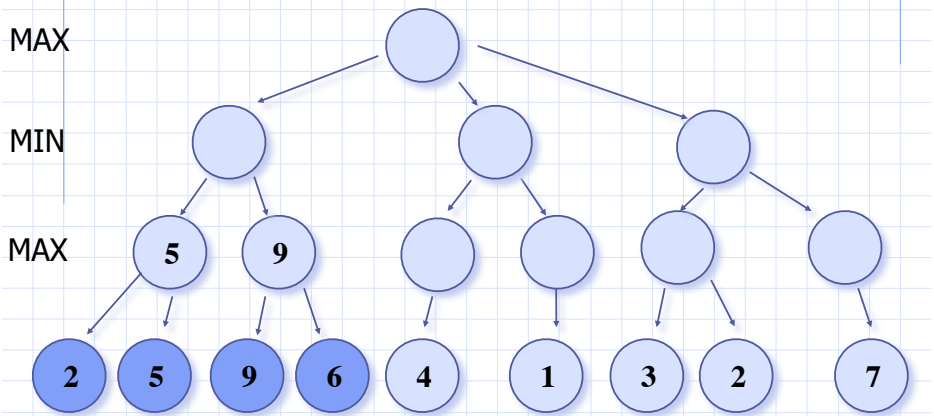
- un nouvel état généré est déja dans ouvert : seul le meilleur des deux reste dans ouvert.
Pour nous, la mesure g, nous evitera de considérer le nouvel état

- JE joue contre LUI
  - JE doit maximiser mes gains
  - LUI doit minimiser mes gains
- Représentation MIN-MAX

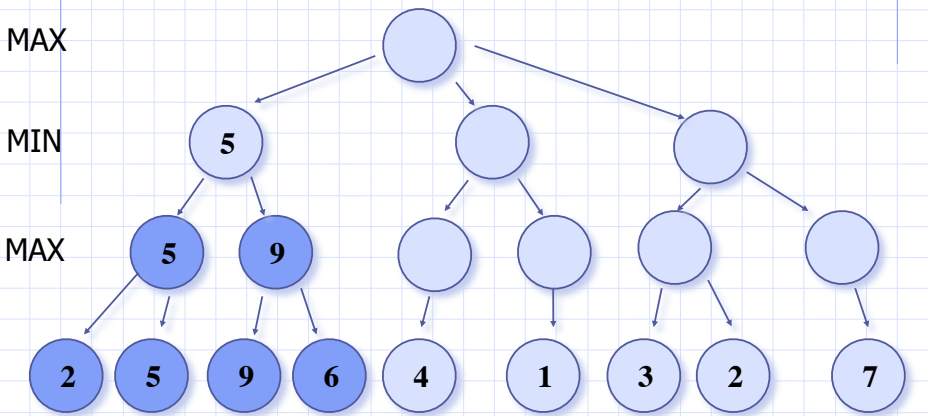




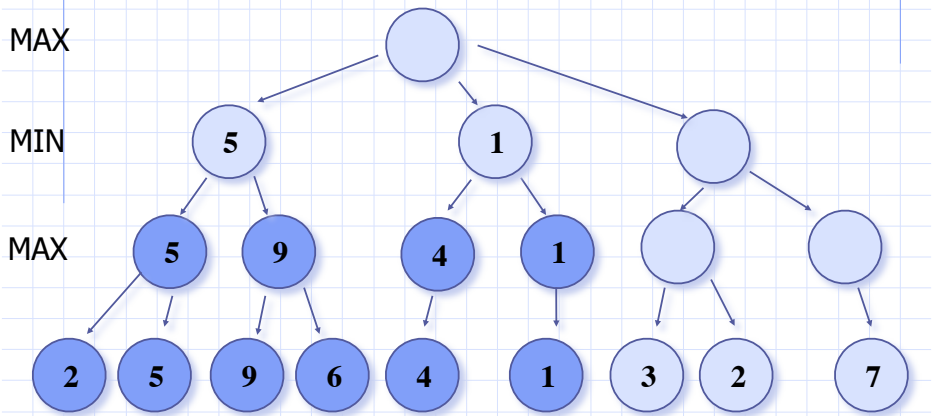
Algorithme MIN-MAX



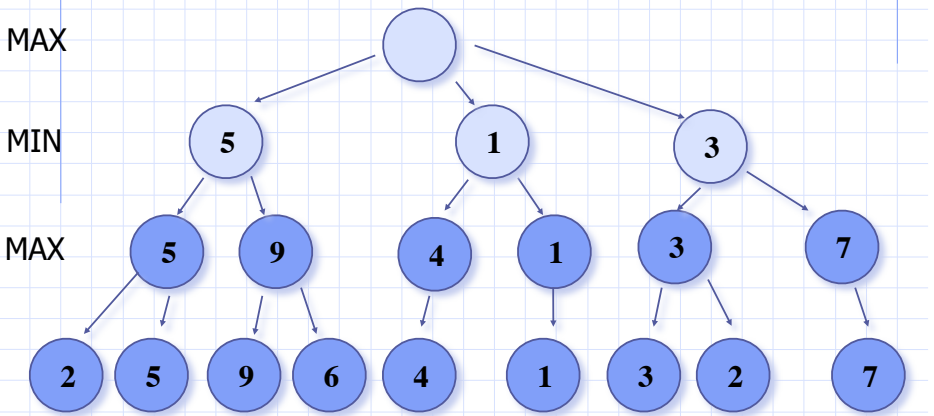
Algorithme MIN-MAX

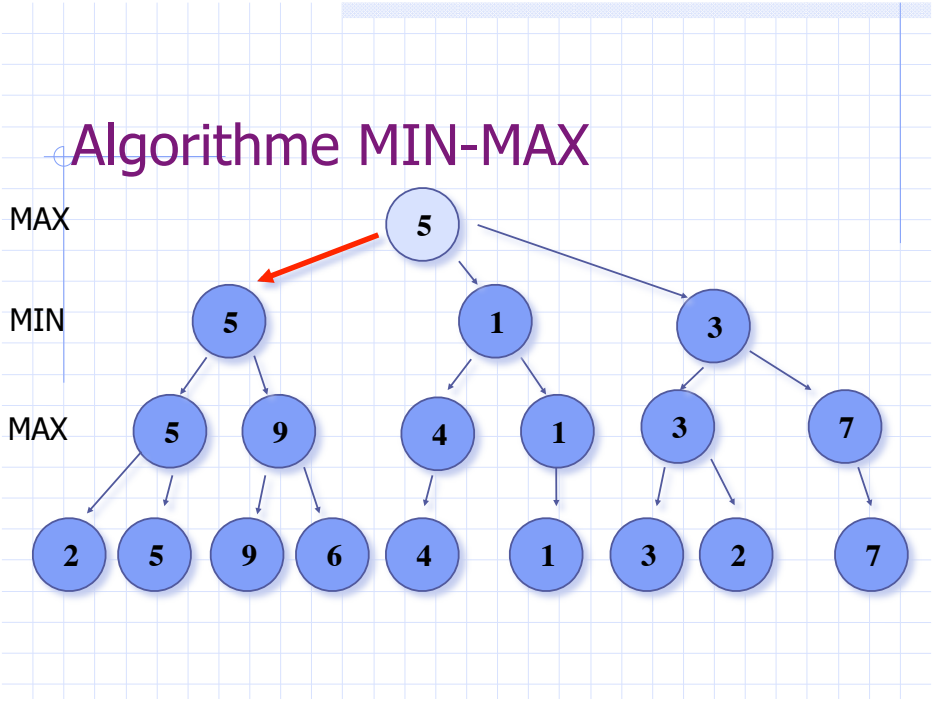


Algorithme MIN-MAX



Algorithme MIN-MAX





Global introduction

Traditional opposition : *classical IA* / *connectionism*

Achievements ?

Global introduction

Traditional opposition : *classical IA* / *connectionism*

Achievement :

In 1997, Deep Blue super computer (IBM) beat world chess champion Gary Kasparov

... and Chess is *one* expression of intelligence (memory, symbol manipulation, adaptation...)

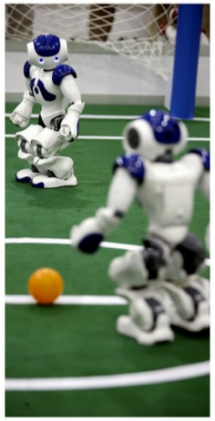
Global introduction

Traditional opposition : *classical IA* / *connectionism*

- ...but

Global introduction

Traditional opposition : *classical IA* / *connectionism*

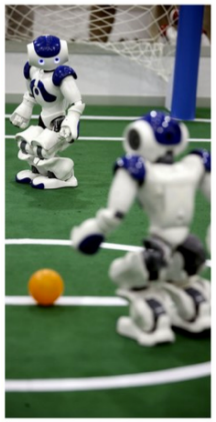


- ...but

Nao humanoid robot is still not able to compete at soccer with a 3 years-old child...

Global introduction

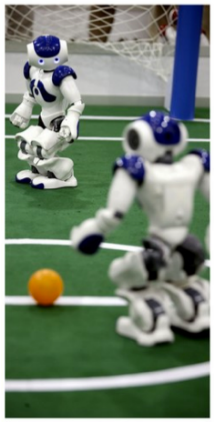
Traditional opposition : *classical IA* / *connectionism*



- we are able to beat the chess world champion....
- ...but not to play elementary ball game such as football
- Why ?

Global introduction

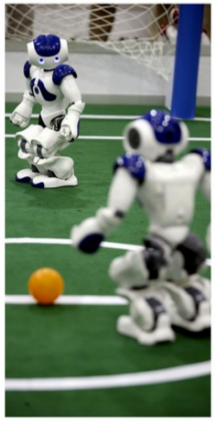
Traditional opposition : *classical IA* / *connectionism*



- perception
- action
- sensorimotor coordination
- memory
- fast decision making
- adaptation (“on line”)
- prediction / anticipation
- theory of mind ? strategy ?
- collective ?

Global introduction

Traditional opposition : *classical IA* / *connectionism*

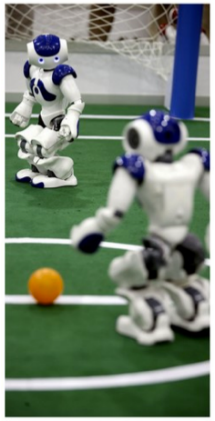


- perception
- action
- sensorimotor coordination
- memory
- fast decision making
- adaptation (“on line”)
- prediction / anticipation
- theory of mind ? strategy ?
- collective ?

All simultaneously...

Global introduction

Traditional opposition : *classical IA* / *connectionism*



- perception
- action
- sensorimotor coordination
- memory
- fast decision making
- adaptation (“on line”)
- prediction / anticipation
- theory of mind ? strategy ?
- collective ?

All simultaneously

...with one brain

Global introduction

Traditional opposition : *classical IA* / *connectionism*

- We are successful when the system (a computer) manipulates symbols...
- when the states are well *defined*
- *well identified*
- when the concepts are correctly *framed*
- when the world is easily *segmented*

Global introduction

Traditional opposition : *classical IA* / *connectionism*

- We are successful when the system (a computer) manipulates symbols...
- when the states are well *defined*
- *well identified*
- when the concepts are correctly *framed, formalised*
- when the world is easily *segmented*
- *Chess is perfect for that*

Global introduction

Traditional opposition : *classical IA* / *connectionism*

A. de Rengervé, J. Hirel, P Andry, M. Quoy, P. Gaussier -ETIS-Cergy
[BioRob2011]

- We fail when the system is immersed in the real world...
- always changing
- unpredictable
- things, peoples, objects, concepts are undefined
- noisy
- *A nightmare for Intelligent robotics*

Global introduction

Traditional opposition : *classical IA* / *connectionism*

- To bypass that these issues, we need :
  - *to let the system adapt*
  - *to let the system learn*
  - *build its own categories*
  - *to exploit the perception-action dynamics*

Global introduction

Traditional opposition : *classical IA* / *connectionism*

- To bypass that these issues, we need :
  - *to let the system adapt*
  - *to let the system learn*
  - *build its own categories*
  - *to exploit the perception-action dynamics*

The brain does it well : connectionism

Global introduction

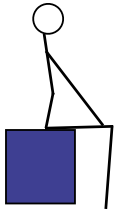
One more example : let's try to define an object : a chair

- Let's do it classical :
  - *object (legs, 4) and object(back,1) and object (seat,1)*

Global introduction

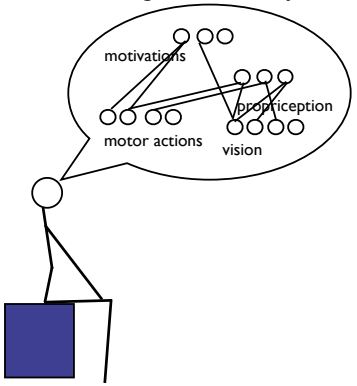
If you want to recognize a chair, you need to :

- **Build** the experience of seating
- to **categorize** it
  - *you need to have legs*
  - *you need to need to sit*



Global introduction

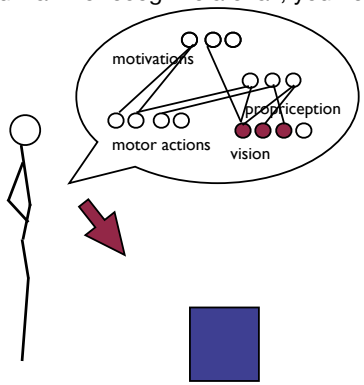
If you want to recognize a chair, you need to :



- **Build** the experience of seating
- to **categorize** it
  - *you need to have legs*
  - *you need to need to seat*
  - *merge the perceptions, the motor actions*
  - *to associate the objects vision with the whole*

Global introduction

If you want to recognize a chair, you need to :



- **Build** the experience of seating
- to **categorize** it
  - *you need to have legs*
  - *you need to need to seat*
  - *merge the perceptions, the motor actions*
  - *to associate the objects vision with the whole*
  - *in order to generalize and recognize*

Global introduction

Connectionism is a part of this perception-action philosophy...

and neural networks is a tool to achieve such adaptive categorization

Overview

Introduction : from biology to the formal neuron

Part I : supervised learning

- perceptron
  - simple rule
  - Widrow Hoff rule
  - limitations
- associative memories
- multi-layer perceptron
- backpropagation

Part II : unsupervised learning

- brain mechanisms
  - competition and cooperation
  - WTA
- Self Organizing Maps
  - Kohonen maps
  - K-means (analogy)
- Let's put it all together
  - ART

From biology...

The brain is the main unit for information processing.

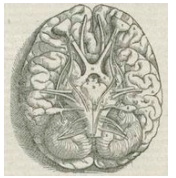
It is composed of 100 billions of nervous cells: the neurons

The neurons are connected in networks in order to :

- monitor
- regulate
- modulate

all the function of the organism.

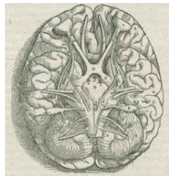
Moreover : the "organ" of intelligence



From biology...

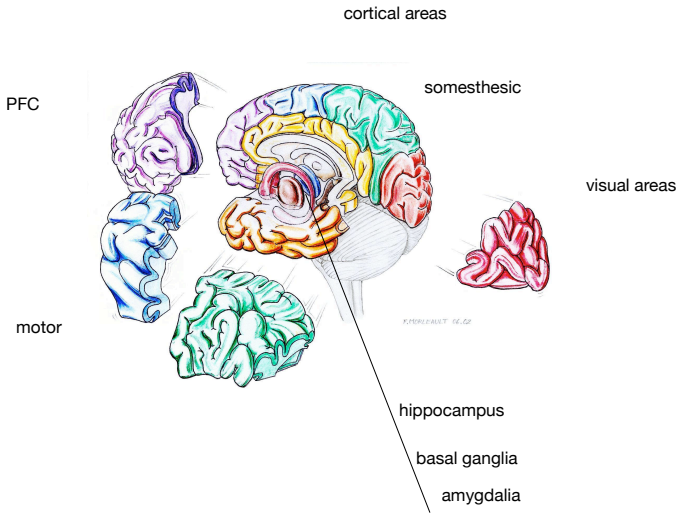
Cognition : all the mental processes involved in the scaffolding of *our* reality, et the basis of reasoning and all the high-level functions.

- perception
- memories
- building categories
- learning
- inhibition
- action selection
- *representations*



Information processing : input -> evaluation->decision->action

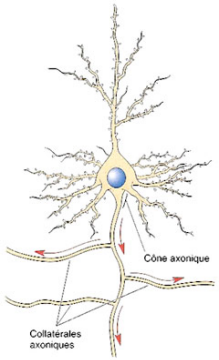
From biology...



From biology...

Neuron : the main unit for information processing

- Dendrit
- Axon
- Soma
- Synapses

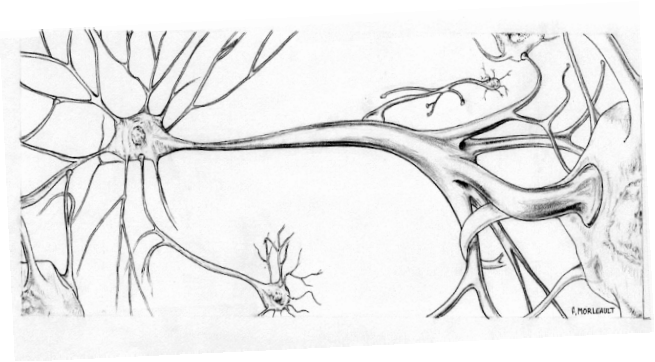


Electric information propagation

uni-directional

from one unit to the other

From biology...



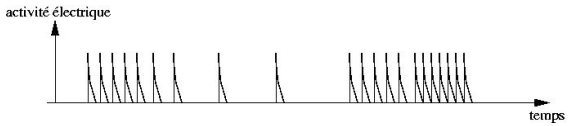
A single neuron can be 1m long

A single neuron can have up to 10 000 connections with other ones

(average of 1000 connections per neurons)

From biology...

But very slow

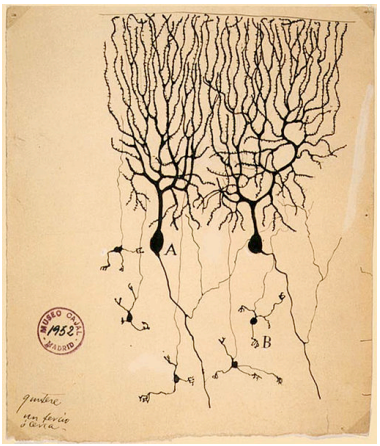


- If you measure the speed of information transmission, you get a rate of 300HZ
- 300HZ = frequency of the action potential : Spike train
- very slow, when compared to modern computer buses (circa 1GHZ)

From biology...

But...

- Multiple connections
- massive parallelism : continuous flow of information tat can be processed by multiples areas at the same time (asynchronous calculation - no main clock-)
- Adaptation : the synapse can adapt to modify the information intensity transmitted by a neuron to the other: learning



From biology...

comparison brain/ modern computer

Von Neumann Architecture	Brain
Computation and memory: separated and centralized	Computation and memory integrated and distributed
program = sequence of instructions	calculus = multiple constraints satisfaction
execution of one process at a time	permanent combination of multiples different information sources
1 to 8 very fast processors t	hundreds of billions of slow connected units

...to information processing...

...to information processing...

At the basis any behavior, is information processing

2 different kinds :

- Chemical, long term (LT) "message sending" **: hormones.** chemical and not restricted to a given receptor, diffusion in all the body, transported by the blood. Slow process.

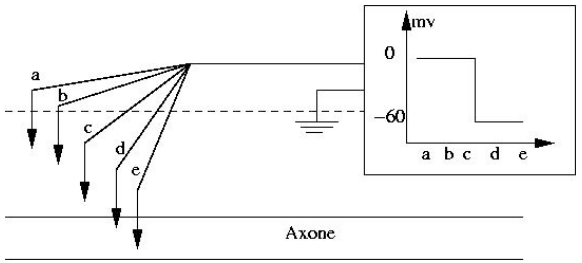
- Electro-chemical, short term, **potential trains** sent by neurons. "Fast"

  - *One neuron produces electrical potential changes, and the changes (the potential variation) are transmitted all along the membranes : emission of electrical signals*

  - *The communication int the inter-neural space (between the synapse and the dendrite) is made by production of chemical substances, the neurotransmitters*

...to information processing...

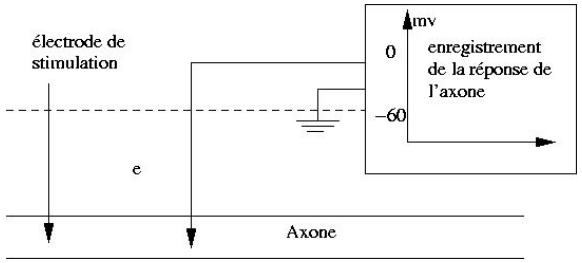
- suppose you have a multimeter to measure the value of the electrical potential inside and outside the axon membrane



- default resting potential (of the neuron): -66 mv : the inside of the axone is negatively charged (difference between a,b,c, and d, e)

...to information processing...

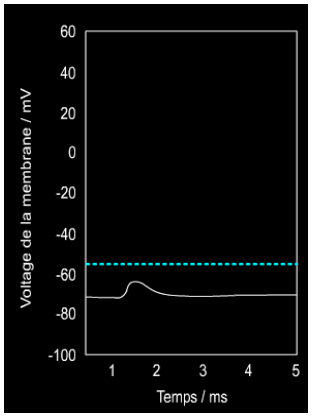
- Suppose you plug an electrode and send a given voltage in the membrane of the axone...



- ...and you record the axon's membrane reaction at a given point

...to information processing...

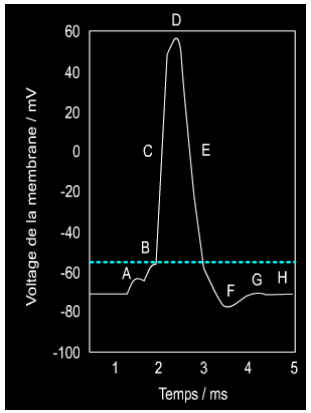
- Stimuli under 10 or 15mv : no significative response from the membrane



...to information processing...

- Depolarization > 10 or 15 mv : strong reaction called action potential .

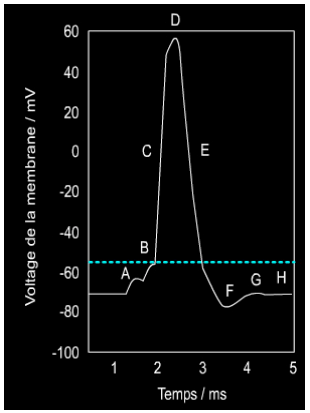
- spike reaching +60mv (C-D)
- overshoot
- whatever the stimulation is (>15mn), the spike is the same
- stereotyped
- followed by a short undershoot (F-G) during no new stimulation is possible: explain the 300Hz limit



...to information processing...

- Depolarization > 10 or 15 mv : strong reaction called action potential .

- The spike will propagate all along the axon with conservation of shape and amplitude
- traveling electrical wave
- 30 m/s



...to information processing...

- If the intensity of the trigger has no effect on the shape of the peak, it will nevertheless influence the amount of peaks generated.

- the stronger the stimulus is, the greater is the number of peaks (frequency max : 300hz)

- The axon is providing a frequency code for the intensity of the stimulus

- potential trains

- naturally, the trigger of the pikes potential will depend on the activity of the neuron's nucleus : the soma

...to information processing...

- The neuron's nucleus will act as an *adder*, summing the incoming potentials (arriving potentials from the dendrites).

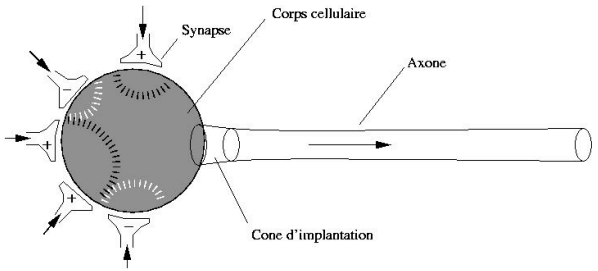
- it is called the soma (soma = sum in latin)

- temporal en spatial sum according to :
  - the synapses positions on the dendrite
  - the frequency of the peaks

- The soma will trigger the axon if a given threshold is overshoot

- non-linearity : thresholds + saturation

...to information processing...



...to a simple formal neuron

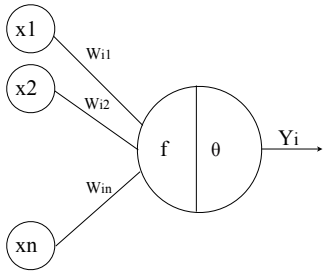
Formal neuron i

- main processing unit
- combined in networks
- calculate the incoming potentials, the Xi [-1,1]:
- summation in an internal potential Poti [-1,1]
- Threshold θ [-1,1]
- deliver one output Yi [-1,1]
- For our introduction we won't use the frequency coding : just a value expressing the mean number of emitted potential
- McCulloc & Pitts model [MacCullocs & Pitts48]

See the works of Thorpe & Al. for models of neurons coding the frequency of the action potential : spiking neurons

...to a simple formal neuron

Formal neuron i



...to a simple formal neuron

Formal neuron i

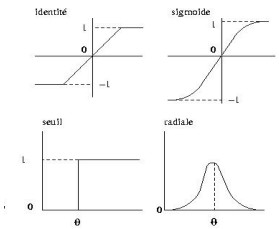
- calculate the incoming potentials, the Xi [-1,1]:
- summation in an internal potential Poti [-1,1]
- Threshold [-1,1]
- deliver one output Yi [-1,1]
- we have seen that the information transmission between two neurons is made by emission of neurotransmitters.
- Coefficient W [-1,1] of the incoming potential

...to a simple formal neuron

Formal neuron i

- Internal potential of neuron i : poti = $\sum$ wij.xj
- activation of output : Yi = f(poti)
- with f, the transfer function according to the model

For example :



identity function

$$f(x) = \begin{cases} 0 \text{ if } x \leq \alpha \\ 1 \text{ if } x > \beta \\ x \text{ if } x \in [\alpha,\beta] \end{cases}$$

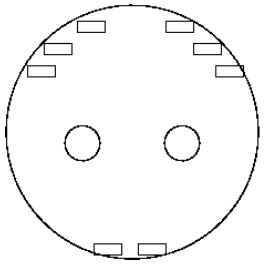
with $\theta = \beta - \alpha$

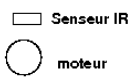
...to a simple formal neuron

Topology of the network ?

example : vehicles

- Braitenberg's vehicles [Braitenberg80]
- You have to build a roomba robot that avoid obstacles.





- Sensors [0,1024] saturation when contact to obstacle
- motors : speed control [-1,1]

example : vehicles

- Classical analysis:

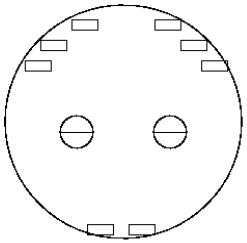
```
if obstacle_detected == left
        turn (angle_right)
if obstacle_detected == right
       turn (angle_left)
 if obstacle_detected == right&left
       ...
else ...
```

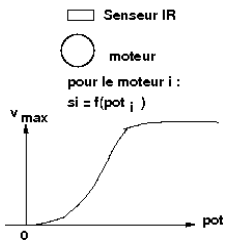
obstacle is an important symbol

Difficult to recognize, frame, (remember the chair...)

example : vehicles

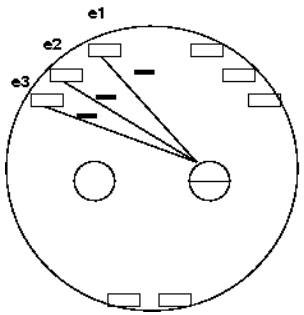
- normalize sensor activities [0,1024] -> [0,1]

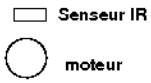




example : vehicles

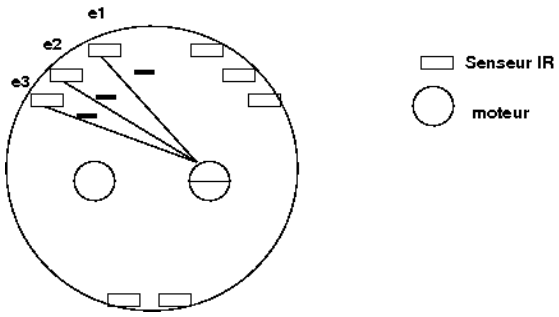
- build connection (weight 1/3 to normalize) to the opposite motor.
- suppose the motor is activated by a neuron
- use identity transfer function





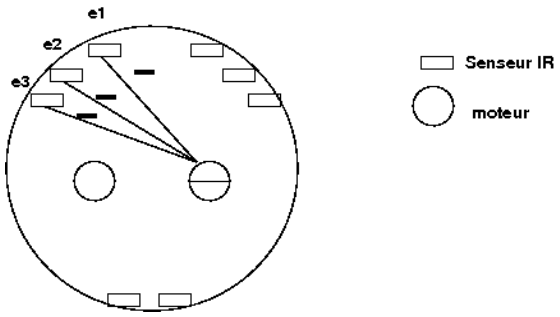
example : vehicles

- sensor saturation induce a diminution of the neuron potential : the opposite motor runs slower, the robot turn



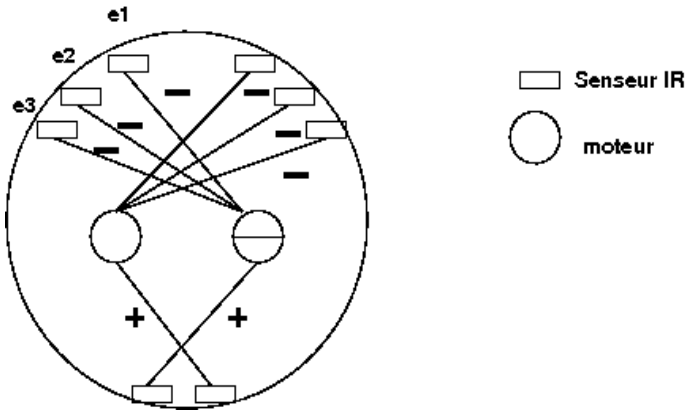
example : vehicles

- Code : only simple operations: pot(i) = e1*w1+e2*w2+e3*w3



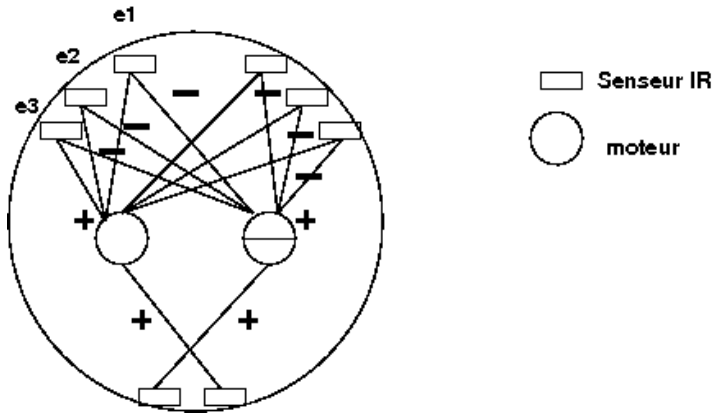
example : vehicles

- Complete architecture:



example : vehicles

- enhancement :

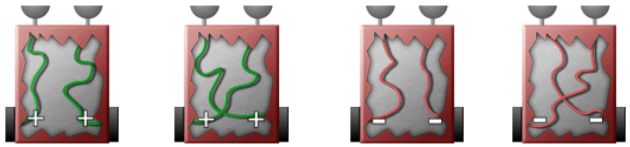


example : vehicles

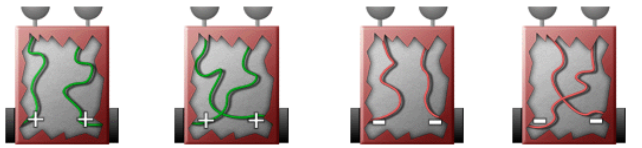
- only 2 neurons
- topology matters
- No “if”, just a vector product : light calculation.
- No notion of obstacle

- behaviors can be stacked with more sensors :
  - obstacle avoidance and phototaxis =
  - stacking circuits

example : vehicles



example : vehicles



fear angry love exploration