

## Résolution de problèmes et Jeux

### L'IA moderne

Actuellement, depuis le tournant du millénaire environ, l'IA est de nouveau en plein essor.

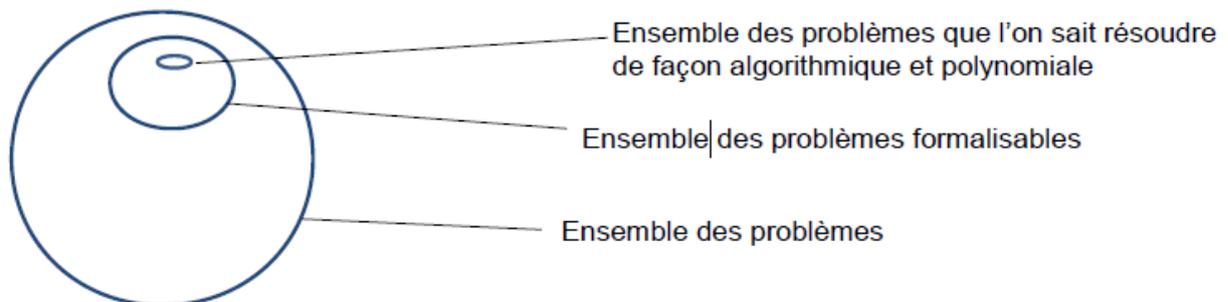
Les méthodes modernes en matière d'IA tendent à se concentrer sur la division d'un problème en un certain nombre de problèmes plus petits, isolés et bien définis, et à les résoudre l'un après l'autre.

L'IA moderne permet de contourner les grandes questions sur le sens de l'intelligence, de l'esprit et de la conscience, et de se concentrer sur la mise en place de solutions pratiques à des problèmes concrets.

Une autre caractéristique des méthodes d'IA moderne, étroitement liées au travail dans le monde réel, chaotique et complexe, est la capacité de gérer l'incertitude à l'aide des probabilités

Enfin, la phase ascendante que connaît actuellement l'IA est largement due au retour des réseaux de neurones et de techniques d'apprentissage profond capables de traiter mieux que jamais les images et autres données du monde réel.

### Classification des problèmes à résoudre

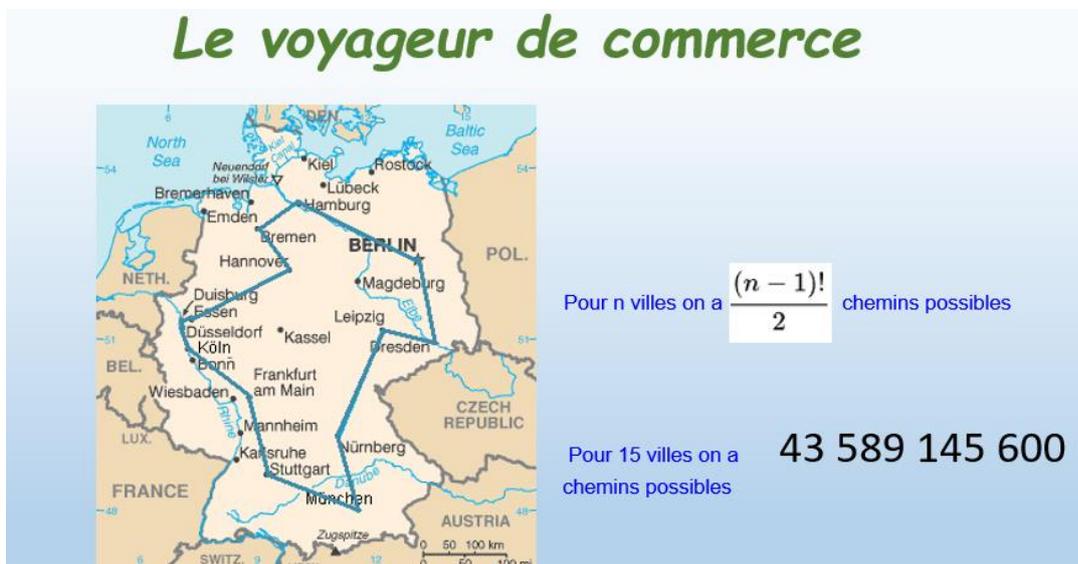


**Problèmes formels résolus par algorithmes polynomiaux** : résolution d'équations en variables entières, tester la connexité d'un graphe, ...

**Problèmes formalisables résolus par algorithmes non polynomiaux** : problème du voyageur de commerce, problème du sac à dos, diagnostics, ...

**Problèmes peu formalisables résolus par systèmes heuristiques**, réseaux de neurones, etc : gagner aux échecs, reconnaître un visage, ...

### Exemple du problème du voyageur de commerce



## Cas étudiés

Nous nous limiterons à **deux types** de problèmes :

- **La recherche et la planification en milieu statique, avec un seul «agent»**
- **Les jeux à deux joueurs («agents») en concurrence l'un avec l'autre**

*Ces catégories sont suffisamment génériques pour illustrer les principaux concepts et techniques*

### Casse tête du poulet

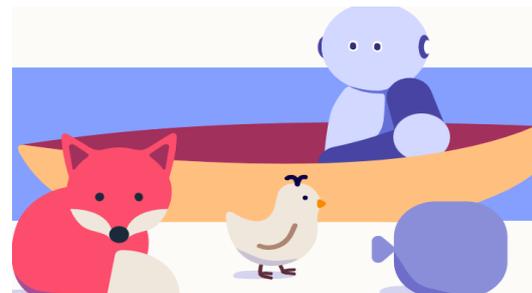
Source : Université Helsinki, MOOC Reaktor 2021

Un robot sur une barque doit faire traverser un cours d'eau à trois cargaisons : **un renard, un poulet et un sac d'aliments pour volailles**.

Le robot est capable d'empêcher les animaux de faire des bêtises s'il se trouve à proximité, mais s'il en a l'occasion, **le renard mangera le poulet et le poulet n'hésitera pas à picorer le sac d'aliments**.

Le robot est seul capable de faire avancer la barque et ne peut embarquer **qu'une ou deux** cargaisons avec lui.

**Comment le robot peut-il faire traverser le cours d'eau à toute sa cargaison ?**



#### Méthode : l'espace d'état, les transitions et les coûts

Pour formaliser un problème de planification, on utilise des concepts tels que l'espace d'état, les transitions et les coûts.

#### Table des états

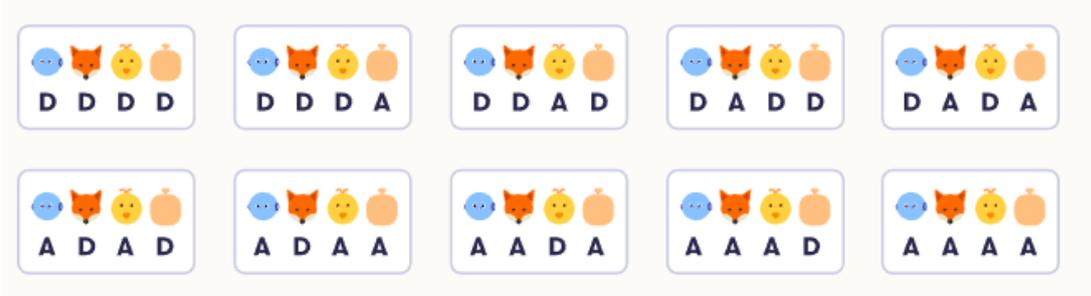
État	Robot	Renard	Poulet	Sac d'aliments
DDDD	Rive de départ	Rive de départ	Rive de départ	Rive de départ
DDDA	Rive de départ	Rive de départ	Rive de départ	Rive d'arrivée
DDAD	Rive de départ	Rive de départ	Rive d'arrivée	Rive de départ
DDAA	Rive de départ	Rive de départ	Rive d'arrivée	Rive d'arrivée
DADD	Rive de départ	Rive d'arrivée	Rive de départ	Rive de départ
DADA	Rive de départ	Rive d'arrivée	Rive de départ	Rive d'arrivée
DAAD	Rive de départ	Rive d'arrivée	Rive d'arrivée	Rive de départ
DAAA	Rive de départ	Rive d'arrivée	Rive d'arrivée	Rive d'arrivée
ADDD	Rive d'arrivée	Rive de départ	Rive de départ	Rive de départ
ADDA	Rive d'arrivée	Rive de départ	Rive de départ	Rive d'arrivée
ADAD	Rive d'arrivée	Rive de départ	Rive d'arrivée	Rive de départ
ADAA	Rive d'arrivée	Rive de départ	Rive d'arrivée	Rive d'arrivée
AADD	Rive d'arrivée	Rive d'arrivée	Rive de départ	Rive de départ
AADA	Rive d'arrivée	Rive d'arrivée	Rive de départ	Rive d'arrivée
AAAD	Rive d'arrivée	Rive d'arrivée	Rive d'arrivée	Rive de départ
AAAA	Rive d'arrivée	Rive d'arrivée	Rive d'arrivée	Rive d'arrivée

**Elimination des états interdits**

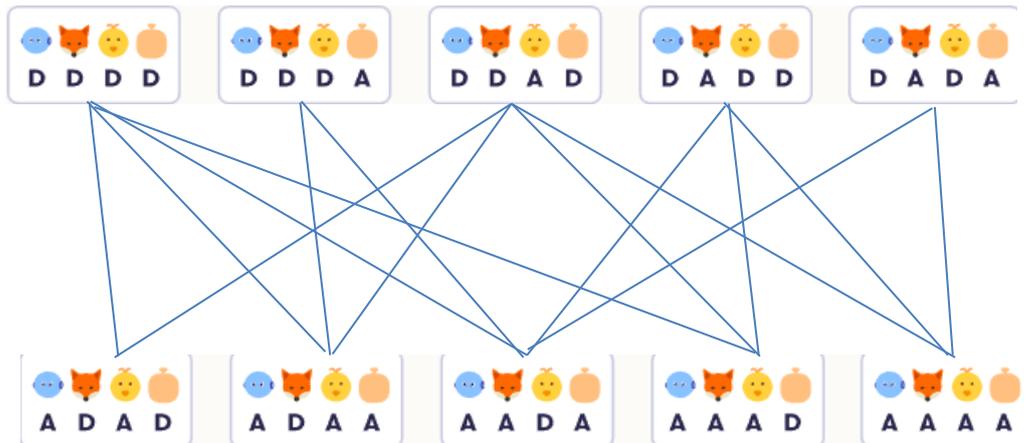
État	Robot	Renard	Poulet	Sac d'aliments
DDDD	Rive de départ	Rive de départ	Rive de départ	Rive de départ
DDDA	Rive de départ	Rive de départ	Rive de départ	Rive d'arrivée
DDAD	Rive de départ	Rive de départ	Rive d'arrivée	Rive de départ
<b>DDAA</b>	Rive de départ	Rive de départ	<b>Rive d'arrivée</b>	<b>Rive d'arrivée</b>
DADD	Rive de départ	Rive d'arrivée	Rive de départ	Rive de départ
DADA	Rive de départ	Rive d'arrivée	Rive de départ	Rive d'arrivée
<b>DAAD</b>	Rive de départ	<b>Rive d'arrivée</b>	<b>Rive d'arrivée</b>	Rive de départ
<b>DAAA</b>	Rive de départ	<b>Rive d'arrivée</b>	<b>Rive d'arrivée</b>	<b>Rive d'arrivée</b>
<b>ADDD</b>	Rive d'arrivée	<b>Rive de départ</b>	<b>Rive de départ</b>	<b>Rive de départ</b>
<b>ADDA</b>	Rive d'arrivée	<b>Rive de départ</b>	<b>Rive de départ</b>	Rive d'arrivée
ADAD	Rive d'arrivée	Rive de départ	Rive d'arrivée	Rive de départ
ADAA	Rive d'arrivée	Rive de départ	Rive d'arrivée	Rive d'arrivée
<b>AADD</b>	Rive d'arrivée	Rive d'arrivée	<b>Rive de départ</b>	<b>Rive de départ</b>
AADA	Rive d'arrivée	Rive d'arrivée	Rive de départ	Rive d'arrivée
AAAD	Rive d'arrivée	Rive d'arrivée	Rive d'arrivée	Rive de départ
AAAA	Rive d'arrivée	Rive d'arrivée	Rive d'arrivée	Rive d'arrivée

**Résultat**

État	Robot	Renard	Poulet	Sac d'aliments
DDDD	Rive de départ	Rive de départ	Rive de départ	Rive de départ
DDDA	Rive de départ	Rive de départ	Rive de départ	Rive d'arrivée
DDAD	Rive de départ	Rive de départ	Rive d'arrivée	Rive de départ
DADD	Rive de départ	Rive d'arrivée	Rive de départ	Rive de départ
DADA	Rive de départ	Rive d'arrivée	Rive de départ	Rive d'arrivée
ADAD	Rive d'arrivée	Rive de départ	Rive d'arrivée	Rive de départ
ADAA	Rive d'arrivée	Rive de départ	Rive d'arrivée	Rive d'arrivée
AADA	Rive d'arrivée	Rive d'arrivée	Rive de départ	Rive d'arrivée
AAAD	Rive d'arrivée	Rive d'arrivée	Rive d'arrivée	Rive de départ
AAAA	Rive d'arrivée	Rive d'arrivée	Rive d'arrivée	Rive d'arrivée



### Les transitions possibles



Christian Pasco

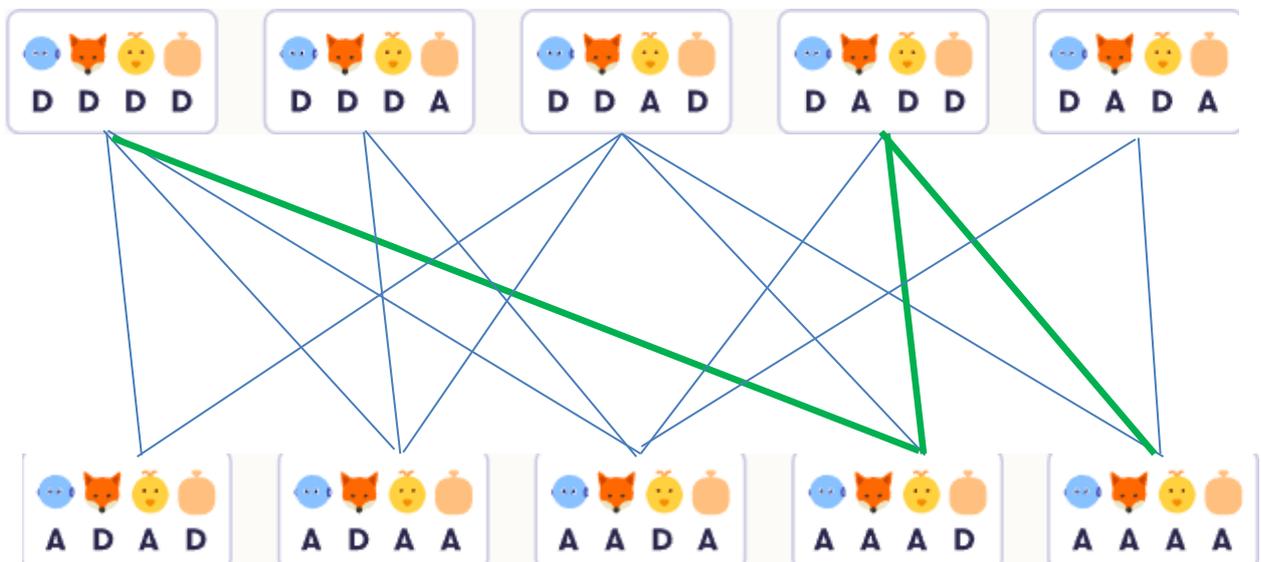
10



**Les différentes transitions ne sont pas toutes identiques. Elles peuvent varier d'une manière qui en rend certaines préférables ou moins « coûteuses » (pas nécessairement au sens monétaire), tandis que d'autres le sont plus.**

*Si l'objectif est de réduire au minimum la distance totale parcourue, la distance géographique entre les états est un coût naturel. D'un autre côté, l'objectif pourrait être de réduire au minimum le temps plutôt que la distance, auquel cas le coût naturel serait évidemment le temps. Si toutes les transitions sont égales, on peut alors ignorer les coûts*

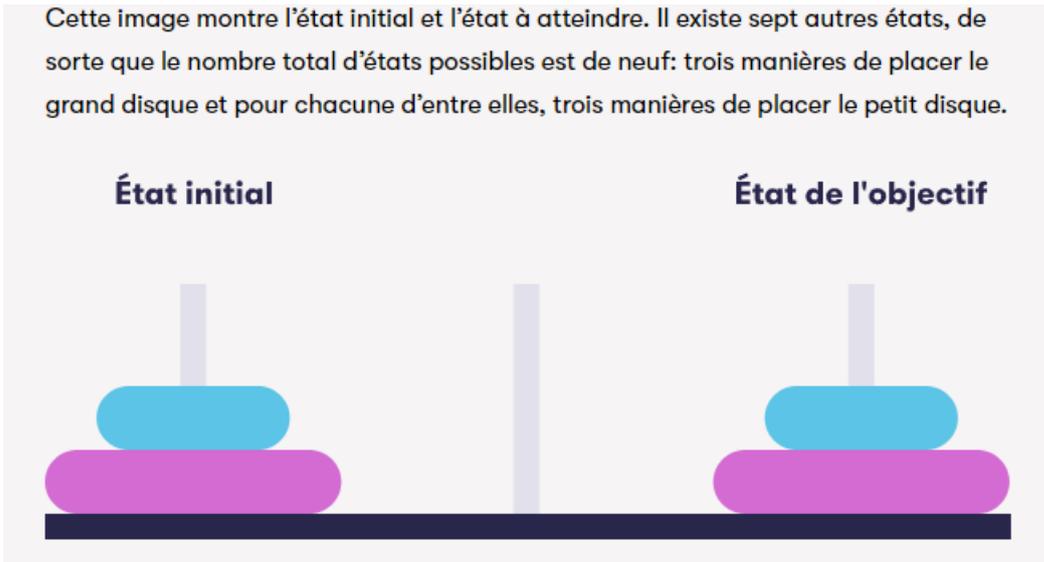
### La transition optimale



Il ne fait guère de doute que vous auriez pu le résoudre en utilisant simplement votre «intelligence naturelle». Mais face à des problèmes plus complexes, où le nombre de solutions possibles peut atteindre des milliers, voire des millions, notre approche systématique ou mécanique s'avérerait bien utile puisque, aussi difficile qu'elle nous paraisse, elle serait tout à fait adaptée aux capacités d'un simple ordinateur.

# Les tours de Hanoi

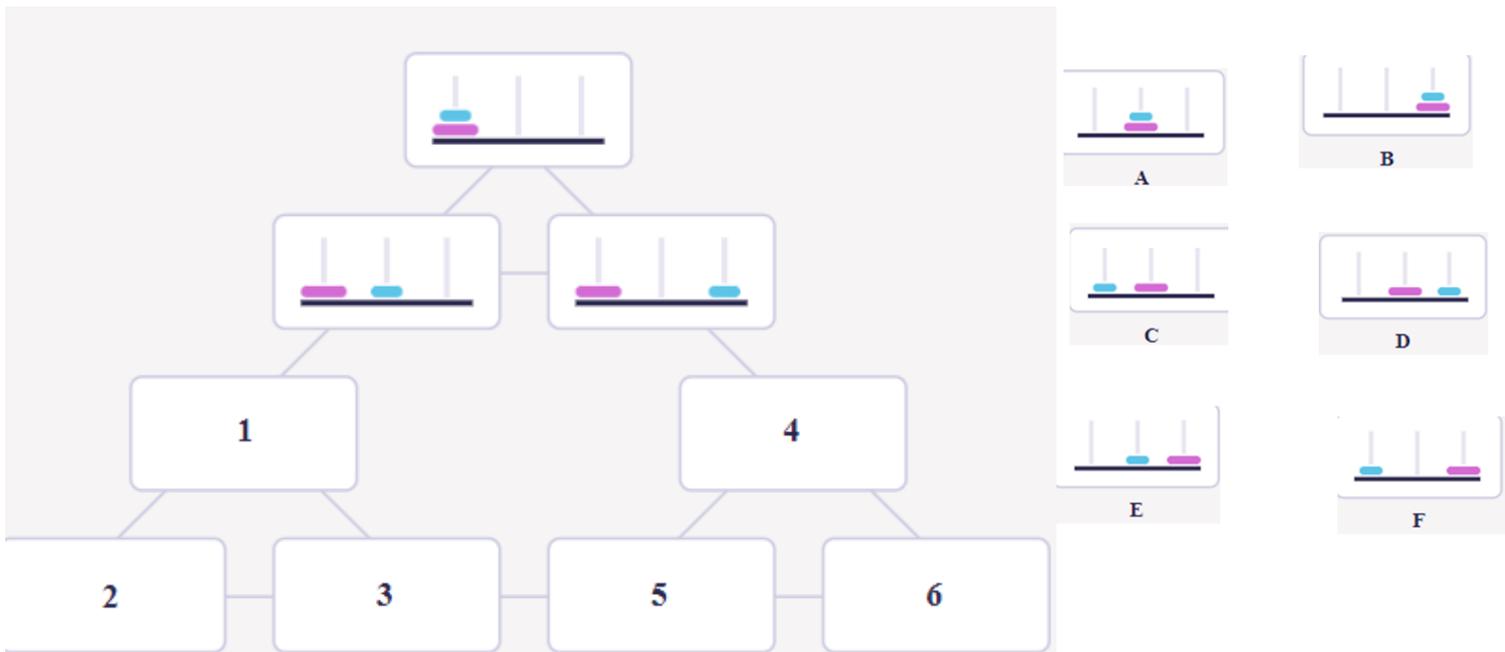
Source : Université Helsinki, MOOC Reaktor 2021



Dessiner le diagramme d'état.

Le diagramme doit inclure les neuf états possibles du jeu, reliés par des lignes qui montrent les transitions possibles. L'image ci-dessous montre la structure générale du diagramme d'état et les positions des trois premiers états. Elle indique qu'à partir de l'état de départ (en haut), vous pouvez déplacer le petit disque vers deux autres états. Notez que les transitions sont de nouveau symétriques et que vous pouvez aussi faire des déplacements sur les côtés (à gauche ou à droite) ou vers le haut dans le diagramme.

Après la résolution de la tâche à l'aide d'un stylo et d'un papier, saisissez votre solution en choisissant quel état appartient à quel nœud dans le diagramme (indice: chaque état appartient exactement à un nœud).



## Les Tours de Hanoi

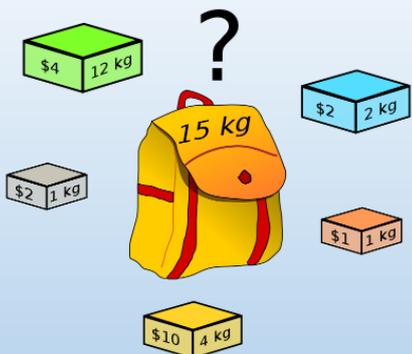


Source : Université Helsinki, MOOCReaktor2021

Christian Pasco

16

# Problème du sac à dos

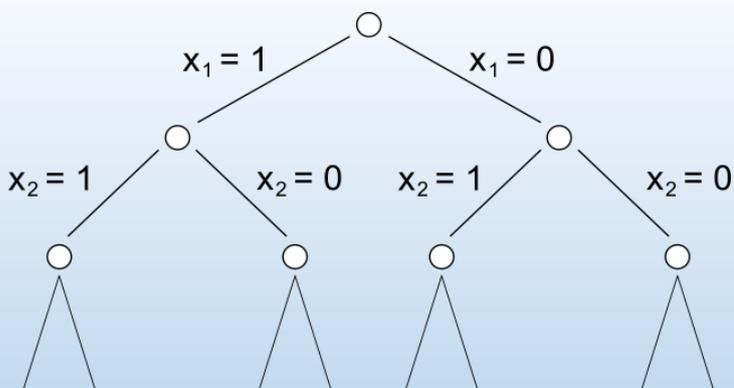
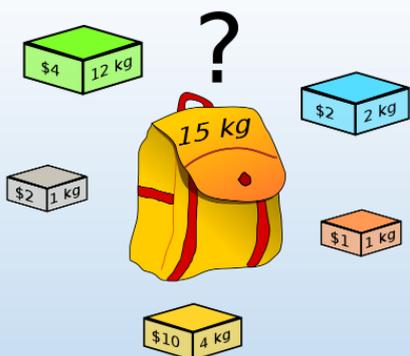


Le problème du sac à dos :  
 quelles boîtes choisir afin de  
 maximiser la somme emportée  
 tout en ne dépassant pas les  
 15 kg autorisés ?

Christian Pasco

18

# Problème du sac à dos



Christian Pasco

20

## Problème du sac à dos

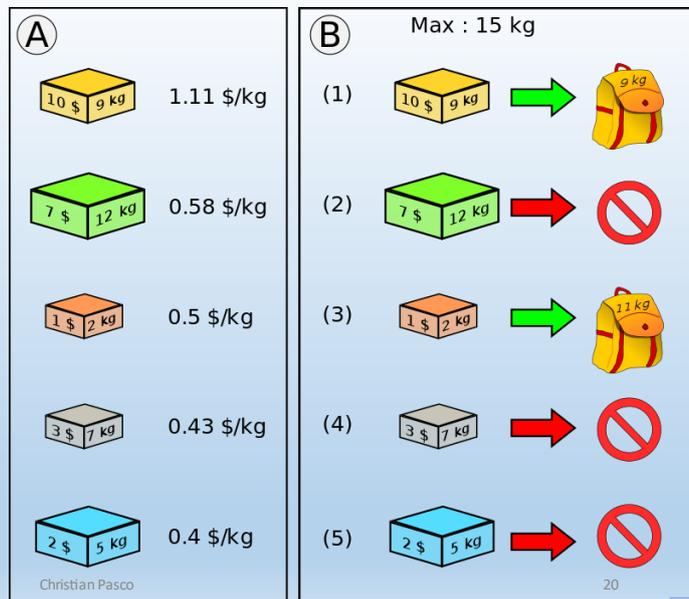
### Les deux phases de l'algorithme glouton.

À gauche : tri des boîtes par ordre d'intérêt (ici en dollars par kilogramme).

À droite : insertion dans l'ordre des boîtes, si cela est possible.

On obtient ici une solution de 11 \$ pour 11 kg

alors que la solution optimale est de 12 \$ et 14 kg.



## Applications

- *Gestion de portefeuille*
- *Chargement bateau ou avion*
- *Optimisation de la découpe de matériaux : minimisation chutes*

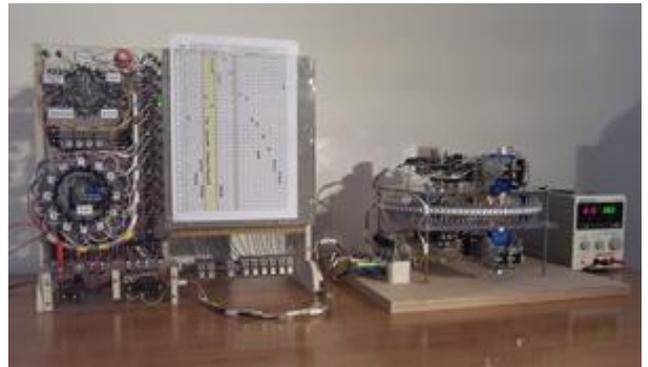
## Résolution de problèmes

### Déclaration de John McCarthy sur l'IA

*«L'étude doit se poursuivre sur la base de l'hypothèse selon laquelle tous les aspects de l'apprentissage ou toute autre caractéristique de l'intelligence peuvent, en principe, être décrits avec une telle précision qu'une machine puisse être conçue pour les simuler.»*

### Le rôle de Turing lors de la Seconde Guerre mondiale

*Turing a conçu un appareil très simple, capable de calculer tout ce qui est calculable. Son appareil est connu sous le nom de machine de Turing.*



### Ordinateurs programmables :

*Utilisés pour effectuer différentes tâches en fonction de ce pour quoi ils ont été programmés.*

*Au lieu d'avoir à construire un nouvel appareil pour chaque tâche, on utilise le même ordinateur pour de nombreuses tâches. C'est le concept de la programmation.*

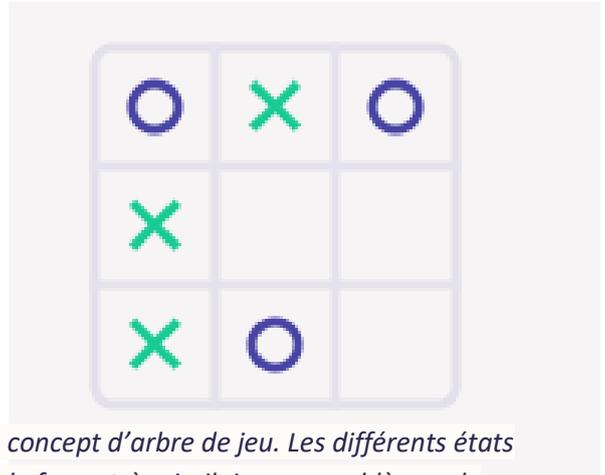
## Les Jeux

*Dans les années 1950, les problèmes d'IA les plus caractéristiques (outre le déchiffrement des codes nazis) concernaient les jeux.*

*Les jeux offraient un domaine restreint pratique, facile à formaliser.*

*Les jeux de plateau tels que les dames, les échecs et le jeu de Go ont inspiré, et inspirent toujours, d'innombrables chercheurs.*

### Le Jeu OXO



*Pour résoudre les jeux grâce à l'IA, nous allons introduire le concept d'arbre de jeu. Les différents états du jeu sont représentés par des nœuds dans l'arbre de jeu, de façon très similaire aux problèmes de planification vus précédemment. L'idée est juste légèrement différente. Dans l'arbre de jeu, les nœuds sont disposés en niveaux correspondant au tour de chaque joueur dans le jeu, de sorte que le nœud «racine» de l'arbre (généralement représenté en haut du diagramme) est la position de départ du jeu.*

*Dans le jeu OXO, il s'agit d'une grille vide, sans qu'aucun X ou O n'ait encore été joué. Sous la racine, au deuxième niveau, on dispose les états possibles qui peuvent résulter du premier tour de jeu, que ce soit un X ou un O. Nous appelons ces nœuds les «enfants» du nœud racine.*

*Chaque nœud du deuxième niveau aura à son tour ses nœuds enfants, soit les états découlant des mouvements de l'adversaire. Cela se poursuit ainsi, niveau après niveau, jusqu'à ce que soient atteints les états qui mettent fin au jeu. Pour OXO, cela signifie que l'un ou l'autre des joueurs obtient une ligne de trois et gagne, ou que la grille est pleine et que le jeu se termine ex-æquo.*

*Afin de pouvoir créer une intelligence artificielle de jeu susceptible de gagner une partie, on attribue une valeur numérique à chaque résultat final possible.*

*Aux positions sur la grille où les X forment une ligne de trois permettant à Max de gagner, on attribue la valeur +1 et, de la même manière, aux positions permettant à Min de l'emporter avec trois O sur une ligne, on attribue la valeur -1.*

*Aux positions où la grille est pleine et où aucun joueur ne peut jouer, on attribue la valeur neutre de 0 (en réalité, peu importent les valeurs du moment qu'elles sont dans cet ordre, de sorte que Max essaie d'augmenter la valeur, et Min tente de la réduire)*

## L'algorithme Minimax

On peut exploiter le concept de valeur du jeu ci-dessus pour obtenir un algorithme dénommé «algorithme Minimax». Cela garantit un jeu optimal en théorie pour tout jeu déterministe, à deux joueurs, à information parfaite et à somme nulle. À partir d'un état du jeu, l'algorithme calcule simplement les valeurs des enfants de l'état donné et choisit celui qui a la valeur la plus élevée si c'est à Max de jouer, ou celui qui a la valeur la plus faible si c'est au tour de Min.

L'algorithme peut être mis en œuvre en utilisant quelques lignes de code :

# Algorithme MiniMax

*À partir d'un état du jeu, l'algorithme calcule simplement les valeurs des enfants de l'état donné et choisit celui qui a la valeur la plus élevée si c'est à Max de jouer, ou celui qui a la valeur la plus faible si c'est au tour de Min.*

```
function minimax(node, depth, maximizingPlayer) is
  if depth = 0 or node is a terminal node then
    return the heuristic value of node
  if maximizingPlayer then
    value := -∞
    for each child of node do
      value := max(value, minimax(child, depth - 1, FALSE))
    return value
  else (* minimizing player *)
    value := +∞
    for each child of node do
      value := min(value, minimax(child, depth - 1, TRUE))
    return value
```

```
(* Initial call *)
minimax(origin, depth, TRUE)
```

**Récurtivité**

28

## Descente

function **minimax(1, 3, False)** is

if depth = 0 or node is a terminal node

then return the heuristic value of node

if maxPlayer then value = -infini

for each child of node do

value = max(value, minimax(child, depth-1, False)

return value

else (\* minimizingPlayer)

value = + infini \* node 1

for each child of node do

value = min(value, **minimax(2, 2, True)** idem pour 3 et 4

return value

function **minimax ( 2, 2, True)** is

if depth = 0 or node is a terminal node

then return the heuristic value of node

if maxPlayer then value = - infini

for each child of node do

value = max(value, **minimax(5, 1, False)** idem pour 6

return value

function **minimax(5, 1, False)** is

else (\* minimizingPlayer)

value = + infini

for each child of node do

value = min(+infini, **minimax(11, 0, True)**

return value

function **minimax(11, 0, True)** is

if depth = 0 or node is a terminal node

then return the heuristic value of

node **-1**

## Remontée

function **minimax(1, 3, False)** is

```

if depth = 0 or node is a terminal node
  then return the heuristic value of node
if maxPlayer then value = -infini
  for each child of node do
    value = max(-infini, minimax(child, depth-1, False))
  return value
else (* minimizingPlayer)
  value = +infini * node 3
  for each child of node do
    value = min(+infini, minimax(2, 2, True) idem pour 3 et 4
  return value -1
  
```

explorer 3 et 5 avant de conclure pour le noeud

function **minimax ( 2, 2, True)** is

```

if depth = 0 or node is a terminal node
  then return the heuristic value of node
if maxPlayer then value = -infini
  for each child of node do
    value = max(-infini, minimax(5, 1, False) idem pour 6
  return value -1
  
```

function **minimax(5, 1, False)** is

```

else (* minimizingPlayer)
  value = +infini
  for each child of node do
    value = min(+infini, minimax(11, 0, True) =-1
  return value -1
  
```

function **minimax(11, 0, True)** is

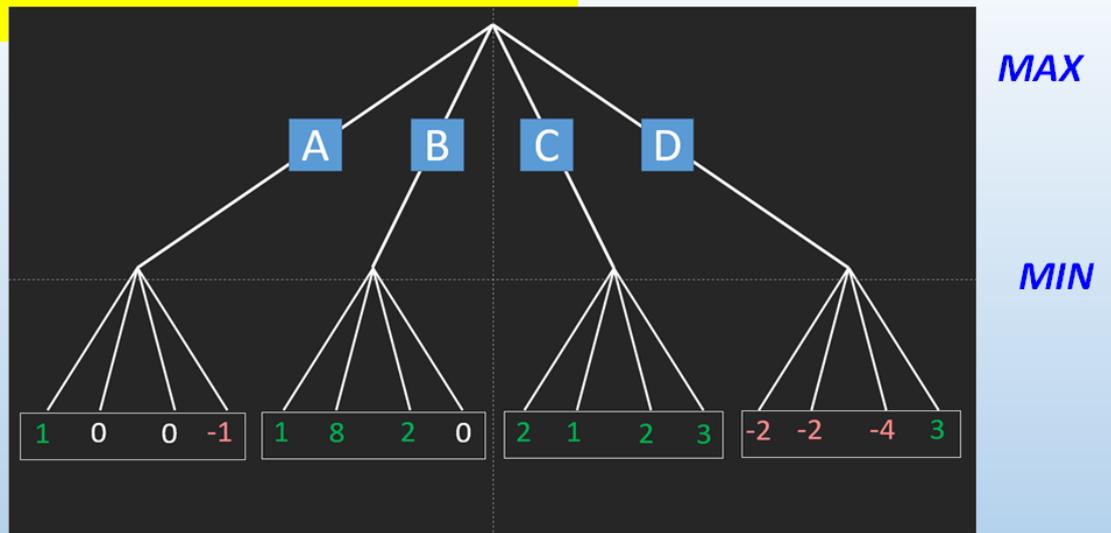
```

if depth = 0 or node is a terminal node
  then return the heuristic value of
  node -1
  
```

## Exercice

C'est à Max de jouer  
A B C ou D ?

JEUX



Video Minimax et Elagage

[https://www.youtube.com/watch?v=f30Ry1WOe\\_Q](https://www.youtube.com/watch?v=f30Ry1WOe_Q)

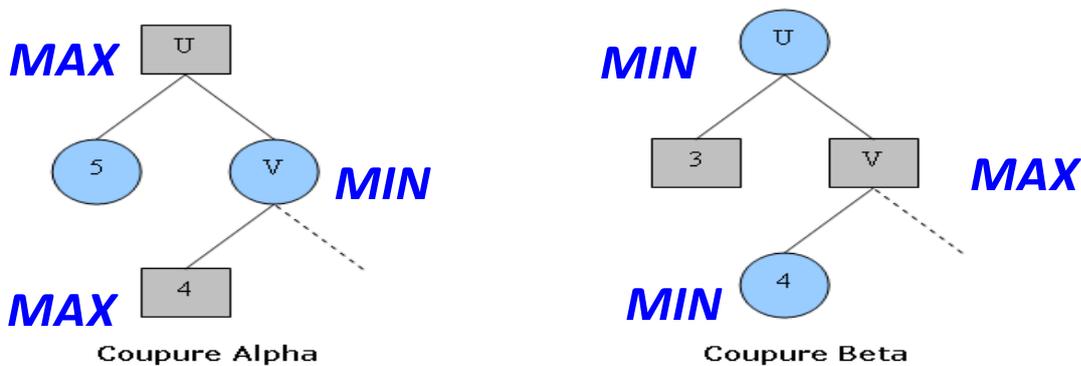
# Gros arbres de jeux

Pour n'explorer qu'une petite partie de l'arbre de jeu, il faut diminuer ou stopper la réursion de minimax.

Plusieurs solutions

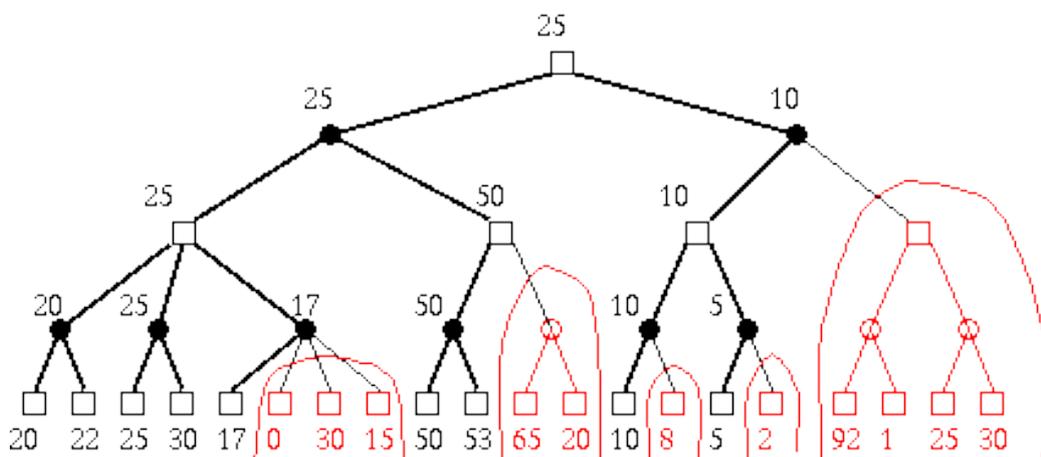
- Algorithme d'élagage Alpha Beta
- Utiliser une fonction d'évaluation heuristique :
- 

Algorithme alpha beta



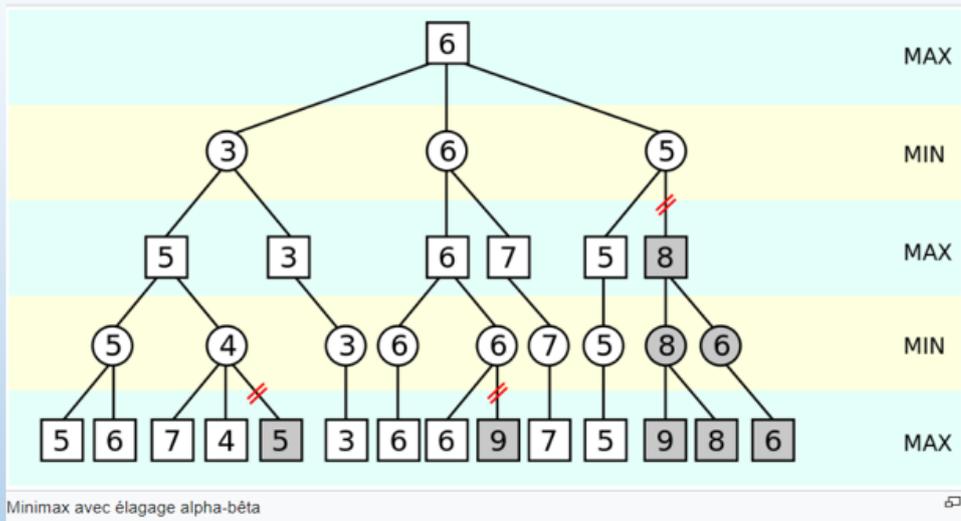
Le Noeud "V" ayant un enfant de valeur 4 < la valeur de son frère (5), le noeud "U" aura la valeur 5

Vidéo : [https://www.youtube.com/watch?v=f30Ry1WOe\\_Q](https://www.youtube.com/watch?v=f30Ry1WOe_Q)



Vidéo : [https://www.youtube.com/watch?v=f30Ry1WOe\\_Q](https://www.youtube.com/watch?v=f30Ry1WOe_Q)

## l'algorithme d'élagage alpha-bêta



Dans de nombreux jeux, l'arbre de jeu est tout simplement trop grand pour être parcouru en entier.

Aux échecs :

- pour un nœud, le nombre moyen d'enfants (coups possibles) est d'environ 35,
- pour deux coups d'avance  $35 \times 35 = 1\,225$  nœuds,
- trois coups : 42 875 nœuds,
- dix coups : 2 758 547 353 515 625
- -

Pour le jeu de Go, le facteur de ramification moyen est estimé à environ 250. Le jeu de Go est hors de portée pour MiniMax.

<https://www.youtube.com/watch?v=UFqaCkTZ65w>

### Nombre de parties possibles :

**Echecs :** Ce nombre est le nombre de Shannon  $10^{120}$

**Go:** Ce nombre est évalué à  $10^{600}$

Il y a  $10^{480}$  fois plus de parties au Go qu'aux échecs !

# Gros arbres de jeux

## Utilisation d'une fonction d'évaluation heuristique :

- **Entrées** : une position sur le plateau, informations relatives au joueur dont le tour de jouer est à suivre
- **Sortie** : score représentant une estimation du résultat probable du jeu

### Exemple pour les échecs

La bonne heuristique compte typiquement la quantité de matériaux (pièces), pondérée en fonction de leur type :

- la reine est généralement considérée comme **valant** près de deux fois une tour, trois fois un cavalier ou un fou et neuf fois un pion.

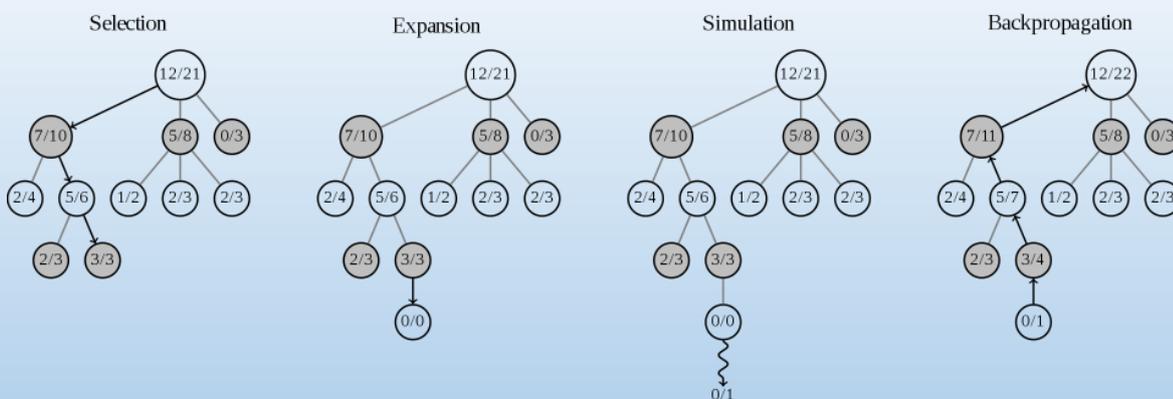
- le roi vaut plus que toutes les autres pièces réunies.

En outre, l'occupation des positions stratégiquement importantes, à proximité du centre du jeu, est considérée comme un avantage et l'heuristique leur attribue une **valeur plus élevée**.

## Monte Carlo Tree Search

L'algorithme MCTS est un algorithme qui explore l'arbre des possibles.

MCTS conserve en mémoire un arbre qui correspond aux nœuds déjà explorés de l'arbre des possibles



<https://www.youtube.com/watch?v=UFqCkTZ65w>

Christian Pasco

40

<https://www.youtube.com/watch?v=UFqCkTZ65w>

## Google AlphaGo et Deep Mind

Google AlphaGO (2016)



### Google AlphaGo

L'IA a battu le meilleur joueur mondial du jeu de go.

L'intelligence artificielle **DeepMind** de Google était derrière le programme **AlphaGo**, qui a gagné 4 parties sur 5 face au champion humain, le joueur coréen Lee Sedol qui ne savait pas qu'il allait jouer contre une machine

**AlphaGo** a été implémentés en utilisant **Monte Carlo** et un **réseau de neurones profond**

**AlphaGo** a été entraîné pour « imiter » les joueurs humains, en retrouvant les coups enregistrés lors de dizaines de milliers de parties menées par des joueurs experts.

Dans une version suivante, **AlphaGoZero**, il a été entraîné à jouer des millions de parties contre d'autres instances de lui-même, utilisant **l'apprentissage par renforcement** pour s'améliorer

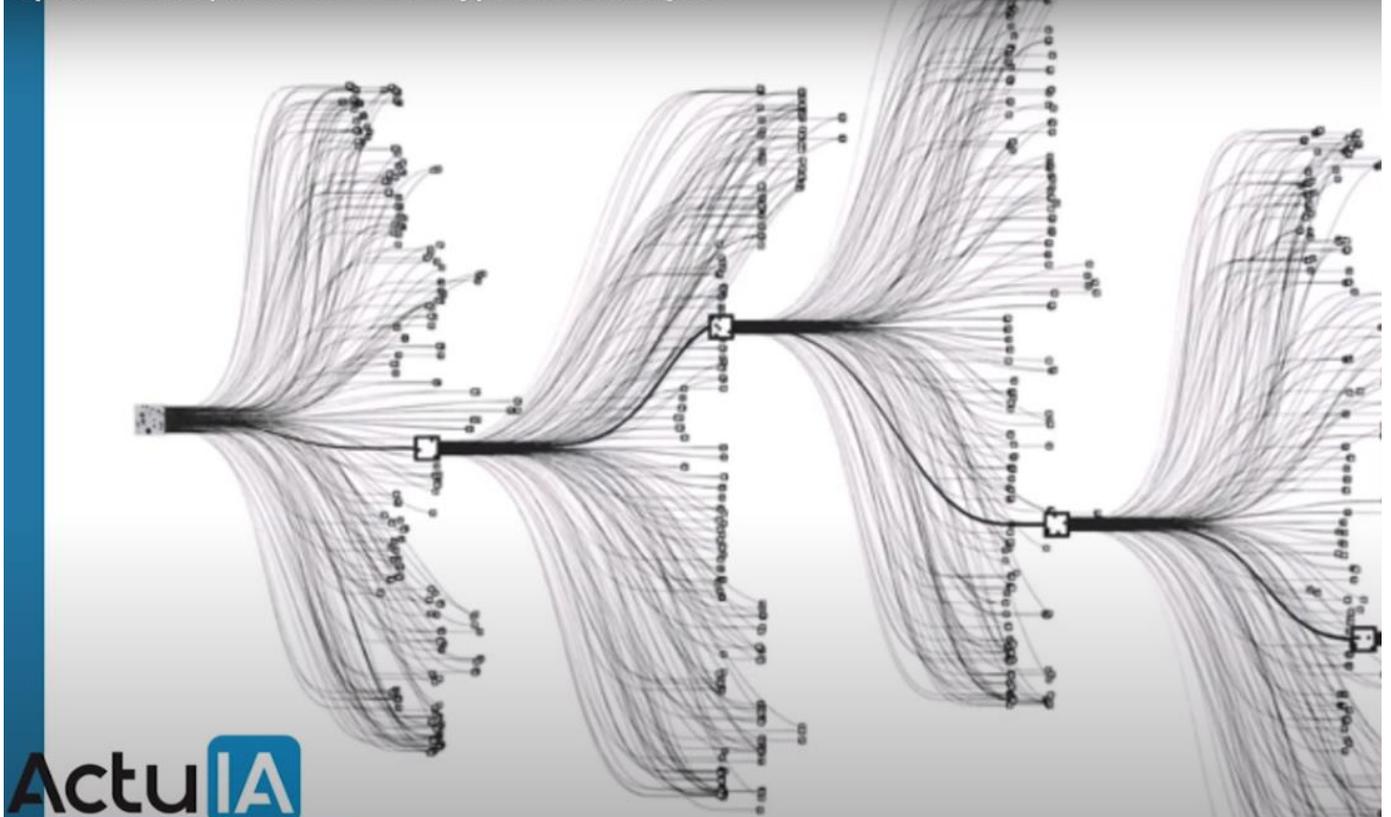
À partir de 2018, un projet collaboratif et open source **AlphaZero**, a obtenu en un an des résultats analogues, portables sur des ordinateurs individuels, et même sur des **smartphones**.

<https://www.youtube.com/watch?v=MRBLKLxjfxM>

Christian Pasco 41

<https://www.youtube.com/watch?v=MRBLKLxjfxM>

AlphaGo et AlphaGo Zero decryptés en Français



**Alphabet** annonce la création d'une nouvelle filiale, **Isomorphic Labs**, qui sera un laboratoire de recherche pharmaceutique s'appuyant sur l'IA de **DeepMind**, société du groupe.

Ce laboratoire construira des modèles capables de prédire les interactions entre les futurs médicaments et nos organismes

La société a également récemment **publié** les formes prédites de plus de 350 000 protéines et ainsi établi une carte des protéines humaines très précise grâce au logiciel **Alphafold** (qui pourrait jouer un rôle important dans la découverte de médicaments).

Des chercheurs indépendants ont utilisé ces données pour accélérer la recherche biologique, y compris pour comprendre le coronavirus.

**7 janvier 2022**

**DeepMind**, filiale du groupe **Alphabet** spécialisée en Intelligence Artificielle, a annoncé en décembre dernier avoir conçu un nouveau modèle de langage basé sur Transformer : **GOPHER**, qui, avec plus de **280 milliards de paramètres**, va lui permettre de tester les limites des grands modèles de traitement naturel.

La modélisation du langage permet de concevoir des systèmes de communication intelligents grâce à de grands référentiels de connaissances humaines écrites. Les chercheurs de DeepMind ont réalisé une analyse des performances du modèle de langage basé sur **Transformer** à travers une large gamme d'échelles de modèles – des modèles avec des dizaines de millions de paramètres jusqu'à un modèle de **280 milliards de paramètres appelé Gopher**. Les transformeurs ont été créés pour faire de la traduction, de la classification ou de la génération de textes mais ont pu très vite être exploités dans de nombreuses tâches de NLP (Natural Language Processing). Ils sont capables d'adapter les mots en fonction du contexte.

Les très grands modèles de langage sont très énergivores : ils consomment des quantités massives de calcul et génèrent des quantités croissantes de dioxyde de carbone.

D'autre part, la reproduction de biais dans la génération des chaînes de texte produites par les modèles pose un réel problème d'éthique.

## AlphaGo et la recherche pharmaceutique

**Novembre 2021**

**Alphabet** annonce la création d'une nouvelle filiale, **Isomorphic Labs**, qui sera un laboratoire de recherche pharmaceutique s'appuyant sur l'IA de **DeepMind**, société du groupe.

Ce laboratoire **construira des modèles capables de prédire les interactions entre les futurs médicaments et nos organismes**

La société a également récemment **publié** les **formes prédites de plus de 350 000 protéines** et ainsi établi une carte des protéines humaines très précise grâce au logiciel **Alphafold** (**qui pourrait jouer un rôle important dans la découverte de médicaments**).

Des chercheurs indépendants ont utilisé ces données pour accélérer la recherche biologique, y compris pour **comprendre le coronavirus**.

## Google AlphaFold DeepMind AlphaBet

Prédit la structure 3D d'une protéine uniquement à partir de sa séquence d'acides aminés

En 2022 :

1. Il trouve des corrélations entre la séquence et la structure en **étant entraîné sur les quelques 170 000 structures connues**
2. Il prédit les structures des protéines à partir de leurs séquences d'acides aminés : actuellement **2 000 000 de protéines**

Christian Pasco

48

## ESMFold Meta

**Octobre 2022 :**

Meta révèle son intelligence artificielle

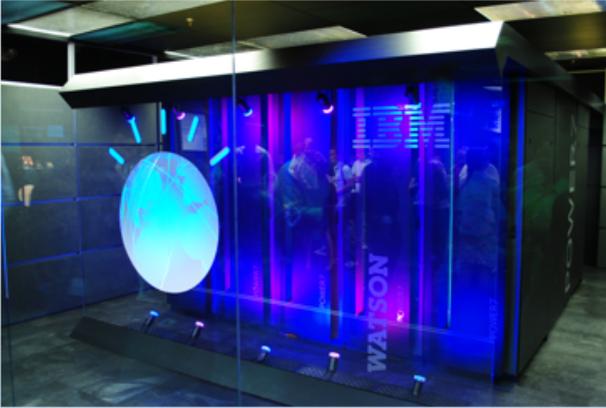
Peut prédire la structure de **200 millions de protéines** après avoir été formée sur des **millions de séquences de protéines**.



Christian Pasco

49

## Watson gagne au Jeopardy



Destiné aux diagnostics médicaux, **Watson** s'est distingué en gagnant **en 2011**, au jeu télévisé **Jeopardy**.

La machine est capable de répondre à des questions posées **dans un anglais naturel** au lieu d'un code utilisant la syntaxe d'un langage de programmation.

**Watson avait 4 To de mémoire vive** à sa disposition, le texte de Wikipedia et l'équivalent **de 200 millions de pages de contenu**.

La machine n'était pas connectée à Internet durant la partie, mais pouvait traiter l'équivalent **d'un million de livres à la seconde**.

Le système qui a joué au jeu télévisé regroupait **90 serveurs IBM PowerPC 750**.

La puissance de calcul atteinte par Watson est de **80 téraflops** (8 000 milliards d'opérations par seconde)

**En 2016**, un centre de recherche de la faculté de médecine de l'université de Tokyo a utilisé **Watson** pour le **diagnostic d'un cas rare de leucémie**. **En dix minutes**, le supercalculateur a trouvé la véritable cause ; les médecins estiment qu'il leur aurait fallu **deux semaines** pour faire la même tâche, rendant le traitement beaucoup plus incertain.

Après 2012, IBM se focalise sur les façons de monétiser Watson en business analytics (analyse commerciale) : « L'objectif est de finaliser trois projets pilotes avec des clients cette année et de lancer la phase de commercialisation de nouvelles offres avant la fin de 2012<sup>12</sup> ».

Du fait de sa capacité à comprendre le langage naturel, Watson serait utile dans le domaine du diagnostic médical<sup>13</sup> ou pour assister les conseillers financiers<sup>14</sup>.

Citigroup explore l'utilisation potentielle de cette technologie pour gérer la masse de données produite en finance, par exemple « les 9 000 pages de données produites quotidiennement par l'agence Reuters ».<sup>1</sup> D'autres cas d'usage de Watson sont envisagés notamment dans des centres d'appels.

## Questions à ce stade

*Test de Turing ?*

*La machine a-t-elle remplacé  
l'homme ?*