COURS SUR
L'APPRENTISSAGE ARTIFICIEL

COURS MASTER IFI 2010/2011

JEAN-DANIEL ZUCKER

DR À L'IRD UR GEODES
(MODÉLISATION MATHÉMATIQUES ET INFORMATIQUES DES SYSTÈMES COMPLEXES)
UMMISCO UMI 209

UNIVERSITE PIERRE & MARIE CURIE
SCIENCE A PARIS

IRD
Institut de recherche pour le développement

Administratif: 1/2 Module Apprentissage (18ECTS)

- **Séance 1: Jeudi 25 Novembre –** INTRO GÉNÉRALE
  - **Introduction, principe inductif, historique, formulation**
  - **Quelques mots sur l'apprentissage statistique**
  - **Espace des versions et algorithme**
- **Séance 2: Lundi 6 Décembre –** APPRENTISSAGE **SUPERVISÉ**
- **Séance 3: Mardi 7 Décembre –** APPRENTISSAGE **NON-SUPERVISÉ**
- **Séance 4: Mardi 11 Janvier 2011 –** ALGORITHMES **EVOLUTIONNAIRES**
- **Séance 5: Vendredi 14 Janvier 2011 –** ALGO. PAR **RENFORCEMENT**
- **Séance 6: Mardi 18 Janvier 2011 –** MINI-PROJET

- **Objectif : Implémentation d'un joueur artificiel pour une variante du jeu de *Alésia*.**

Chaque joueur reçoit, au début de la partie, un capital de 10 points. Chaque coup consiste en un combat de points; Les joueurs décident simultanément et secrètement du nombre de points (au moins un) qu'ils affectent à ce coup. Le joueur qui a choisi le nombre le plus élevé bouge le vers le centre. En cas de mises égales, les marqueurs ne bougent pas. Les nombres joués sont ensuite retranchés des capitaux respectifs des joueurs. Un joueur gagne s'il parvient à déplacer le marqueur jusqu'au centre. Sinon, il y a partie nulle.

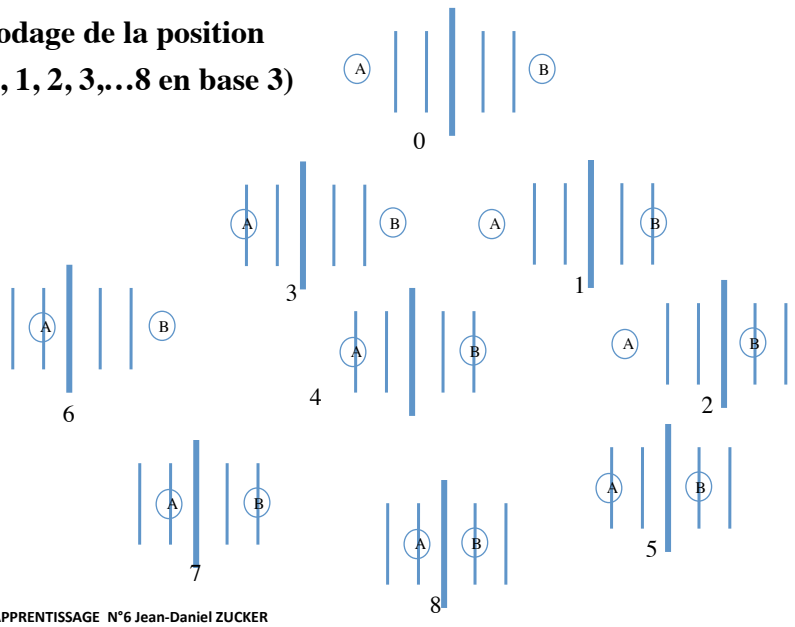
- **Championnat de Alésia**
  - **Joueurs du championnat**
    - **Joueurs programmés par les étudiants**
    - **Joueurs basiques**
  - **Chaque joueur affrontera l'ensemble des autres joueurs durant un match**
  - **Un match est composé de 1000 parties.**
    - **A chaque partie victorieuse : (reward) score = +1**
    - **A chaque partie perdante : (reward) score = - 1**
    - **A chaque partie nulle : (reward) score = 0**
  - **Classement du championnat : par score**

A faire (par binôme) 1/4

- **Implémentation en JAVA d'un joueur**
  - **Utilisation de ce que l'on a vu durant le cours d'apprentissage**
  - **Pour le nombre de jetons initial prendre 10**
  - **Chaque binôme doit choisir un nom de joueur en 6 lettres ASCII et « V » et un numéro de version de 0-99 (ex: JDZCKRV01)**
- **Rédaction d'un rapport**
  - **Sous la forme d'un article (introduction. Problème. Codage du jeu. Codage de l'apprentissage. Apprentissage. Expérimentation avant/après apprentissage. Score. Conclusion.**

Deadline : Lundi 7 Février envoi de joueur aux autres+Prof.
Vendredi 11 Février envoi du rapport et joueur au Prof.

- **Codage de la position (0, 1, 2, 3,…8 en base 3)**



A faire (par binôme) 2/4

- **Implémentation d'un joueur**
  - **Créer une classe qui implémente l'interface « Joueur »**
  - **Sauvegarde de votre joueur sous la forme d'un fichier. JDZCKRV01.txt et d'un jar JDZCKR.jar**
  - **Méthodes à implémenter (entre autre)… à discuter**
    - **public void NewMatch(int nbParties = 1000);**
    - **public int NextMove(int moneyA, int p, int moneyB);**
      - (p vaut 0, 1, 2, … 8 correspondant aux positions 00 01 02 10 11 12 20 21 22) et player qui possede moneyA
    - **public String getAuteur();**
    - **public void LoadPlayer(file);**
    - **public file SaveMe();**
  - **Remarques**
    - **Possibilité d'utiliser la lib Weka.jar**
    - **Eviter d'implémenter un joueur trop lent !**

A faire (par binôme) 3/4

- **Rédaction d'un rapport**
  - **Sous la forme d'un article**
  - **Utilisation du format « Cap 2010 »**
    - **Latex http://cap10.isima.fr/templates/template_latex.zip**
    - **Word http://cap10.isima.fr/templates/template_word.doc**
  - **Maximum 12 pages (tout inclus : résumé, algo, bouts de code, bibliographie, figures, etc.)**
  - **Plan type :**
    1. **Introduction**
    2. **Contexte**
    3. **Approche proposée: codage du jeu/codage des algos**
    4. **Expérimentations: apprentissage du joueur**
    5. **Conclusion**

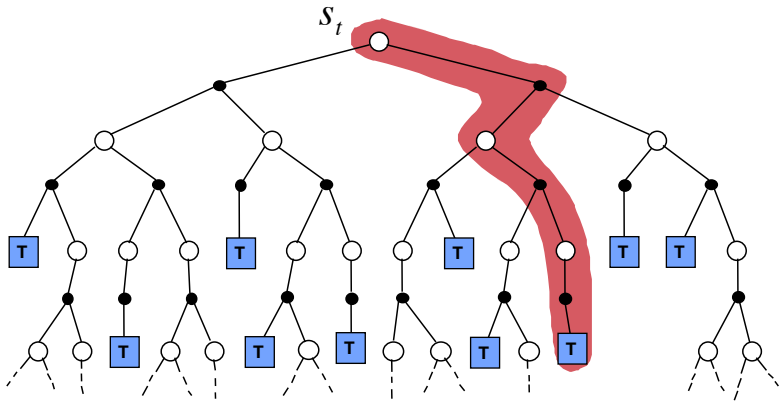
Evaluation 4/5

- **Joueur : 50 % de la note**
  - **Performance**
  - **Utilisation pertinente des techniques de l'apprentissage artificiel**
  - **Clarté du code**
- **Rapport : 50 % de la note**
  - **Originalité / Importance**
  - **Présentation / Rédaction**
  - **Qualité technique / Cohérence**
  - **Evaluation / Validation**
  - **Positionnement / Etat de l'art**

Rappel 1/2: Algo1: Simple Monte Carlo

$$V(s_t) \leftarrow V(s_t) + \alpha\left[R_t - V(s_t)\right]$$

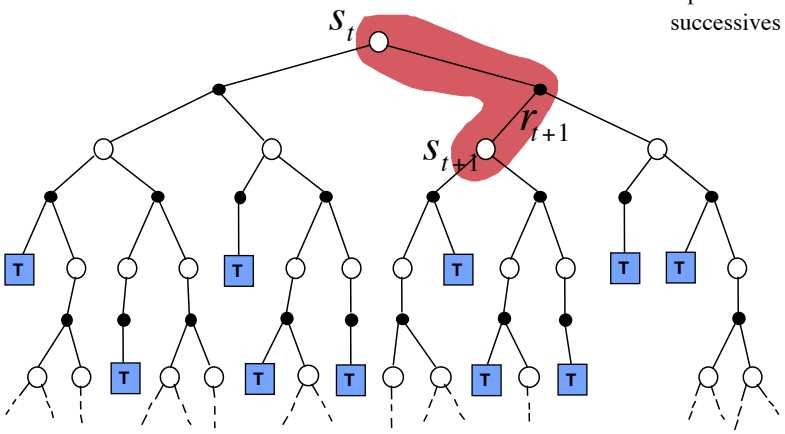
where R_t is the actual return following state s_t.



Rappel 2/2: Algo2: Simplest TD Method

$$V(s_t) \leftarrow V(s_t) + \alpha\left[r_{t+1} + \gamma\, V(s_{t+1}) - V(s_t)\right]$$

On met à jour incrémentalement par ré-estimations successives et locales



Bilan : trois idées principales

1. **La passage par des fonctions d'utilité (V(s) ou Q(a,s)**
2. **La rétro-propagation de ces valeurs le long de trajectoires réelles ou simulées**
3. **Itération généralisée de politique : (i) calculer continuellement une estimation de la fonction d'utilité optimale et (ii) chercher une politique optimale grâce à cette estimation, qui, en retour, s'adapte en conséquence**

Algorithme de mise-à-jour de la fonction V

Initialisation :

$\pi \leftarrow$ politique à évaluer

$V \leftarrow$ une fonction arbitraire d'évaluation

Répéter (pour chaque pas de l'épisode) :

$a \leftarrow$ action préconisée par π pour s

Faire a; recevoir r; voir état suivant s'

$V(s) \leftarrow V(s) + \alpha\,[r + \gamma V(s') - V(s)]$

$s \leftarrow s'$

jusqu'à s terminal

Sélection d'action ε-gloutonne

- **Sélection d'action gloutonne :**

$$a_t = a_t^* = \arg\max_a Q_t(a)$$ **= Arg max Vt(∂(s),a)**

- **ε-gloutonne :**

$$a_t = \begin{cases} a_t^* & \text{with probability } 1-\varepsilon \\ \text{random action} & \text{with probability } \varepsilon \end{cases}$$

. . . La manière la plus simple de pondérer l'exploration et l'exploitation

Des applications dans des problèmes réels

- TD-Gammon**: Tesauro**
  - **world's best backgammon program**
- Elevator Control**: Crites & Barto**
  - **high performance down-peak elevator controller**
- Inventory Management**: Van Roy, Bertsekas, Lee&Tsitsiklis**
  - **10–15% improvement over industry standard methods**
- Dynamic Channel Assignment**: Singh & Bertsekas, Nie & Haykin**
  - **high performance assignment of radio channels to mobile telephone calls**

Sources documentaires

- **Ouvrages / articles**
  - **Sutton & Barto (98) : *Reinforcement Learning : an introduction*. MIT Press, 1998.**
  - **Kaelbling L.P. (93) : *Learning in embedded systems*. MIT Press, 1993.**
  - **Kaelbling, Littman & Moore (96) : *Reinforcement learning : A survey*. Journal of Artificial Intelligence Research, 4:237-285.**
- **Sites web**
  - http://webdocs.cs.ualberta.ca/~sutton/RL-FAQ.html
    (FAQ maintenue par Rich Sutton et point d'entrée pour de nombreux sites)