Apprentissage par Renforcement: Plan du cours

Active Learning, position of the problem

Supervised learning, the setting

- Target hypothesis h^*
- Training set $\mathcal{E} = \{(x_i, y_i), i = 1 \dots n\}$
- Learn h_n from $\mathcal{E}$

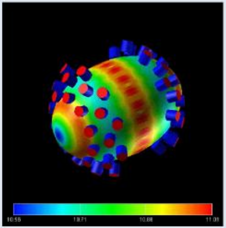
Criteria

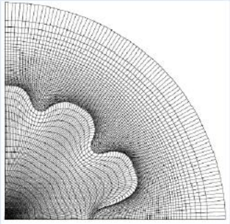
- Consistency: $h_n \rightarrow h^*$ when $n \rightarrow \infty$.
- Sample complexity: number of examples needed to reach the target with precision ϵ

$$\epsilon \rightarrow n_\epsilon \ s.t. \ ||h_n - h^*|| < \epsilon$$

Motivations

- Given *x*, obtaining *h*(x)* is costly
- Goal: reduce *sample complexity* while keeping *generalization error* low
- Motivating application: numerical engineering





=> Learn simplified models with only ~ 100 examples

Active Learning, definition

Passive learning

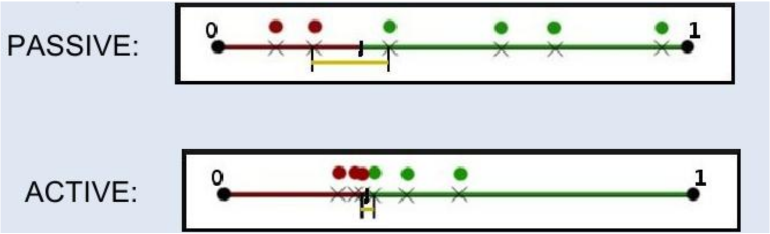
iid examples

$$\mathcal{E} = \{(x_i, y_i), i = 1 \ldots n\}$$

Active learning

x_{n+1} selected depending on $\{(x_i, y_i), i = 1 \ldots n\}$

In the best case, exponential improvement:



State of the art

Let H be the hypothesis space.
Realizable assumption: $h^* \in H$
Then, exponential improvements. Freund et al. 1997; Dasgupta 2005; Balcan et al. 2010.

Noisy case: improvement depends on noise model
Balcan et al. 2006; Hanneke 2007; Dasgupta et al. 2008.

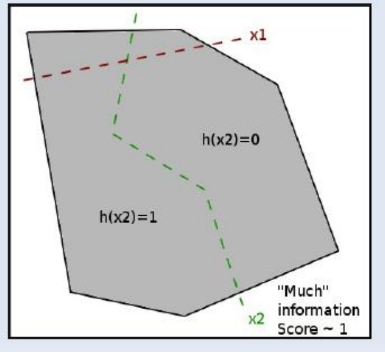
Realizable batch case
PhD Philippe Rolet, 23 dec. 2010.

How it works

Principle

- Design a measure of the information brought by an instance
- Iteratively select the best instance

Example: query by committee Seung et al. 92



Active Learning

Optimization problem

- T: time horizon (number of instances to select)
- States $s_t = \{(x_i, h^*(x_i)), i = 1 \ldots t\}$
- Action: select x_{t+1}
- $\mathcal{A}$: Machine Learning algorithm
- **Err**: Generalization error

Find Sampling strategy S minimizing $\mathbb{E}\mathbf{Err}(\mathcal{A}(S_T(h^*), h^*)$

Bottlenecks

- Combinatorial optimization problem - in a continuous space
- Generalization error unknown

Optimal Strategy for AL

- Learning algorithm $\mathcal{A}$
- Finite Horizon T
- Sampling strategy S_T
- Target concept h^*

Learner $\mathcal{A}$

selects x1 →

← *answers h*(x1)*

selects x2 →

← *answers h*(x2)*

⋮

T-size training set $S_T(h^*)$

$\{(x_1, h^*(x_1)), \ldots, (x_T, h^*(x_T))\}$

Target Concept h^*
(a.k.a. Oracle)

- **Goal:** $\mathrm{argmin}\ E[\ Err(\mathcal{A}(S_T(h^*)), h^*)]$

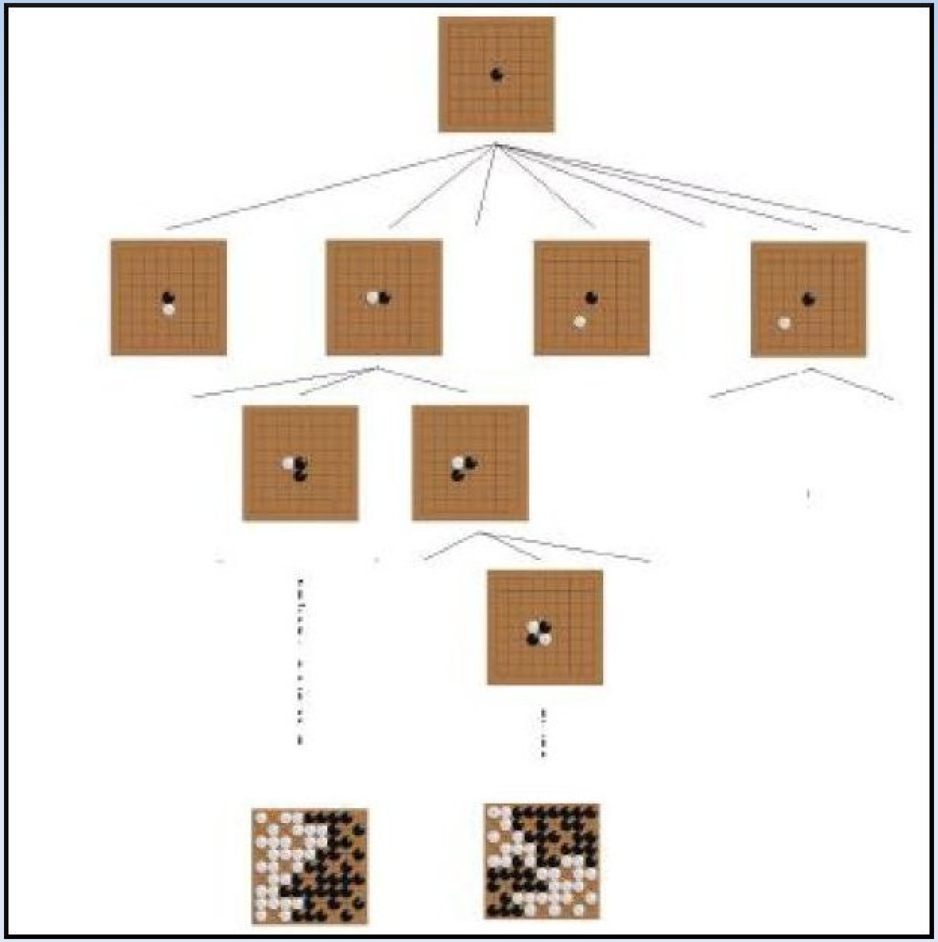
Optimal Strategy for AL

- AL modeled as a Markov decision process:
  - State space: all possible training sets of size $\leq T$
  - Action space: instances x available for query
  - Transition function: $P(s_{t+1} | s_t, x)$
  - Reward function: gen. err. $Err(\mathcal{A}(S_T(h)), h)$
- Optimal policy $\pi^* \rightarrow$ Optimal AL strategy

Active Learning: a 1-Player Game

- Bottlenecks:
  - Large state space
  - Large action space
  - Cannot use h* directly
- Approx. sol. inspired from Go: AL as a game

  Coulom 06, Chaslot et al. 06, Gelly&Sliver 07
- Browse game tree
- Estimate move values with *Monte-Carlo* simulations



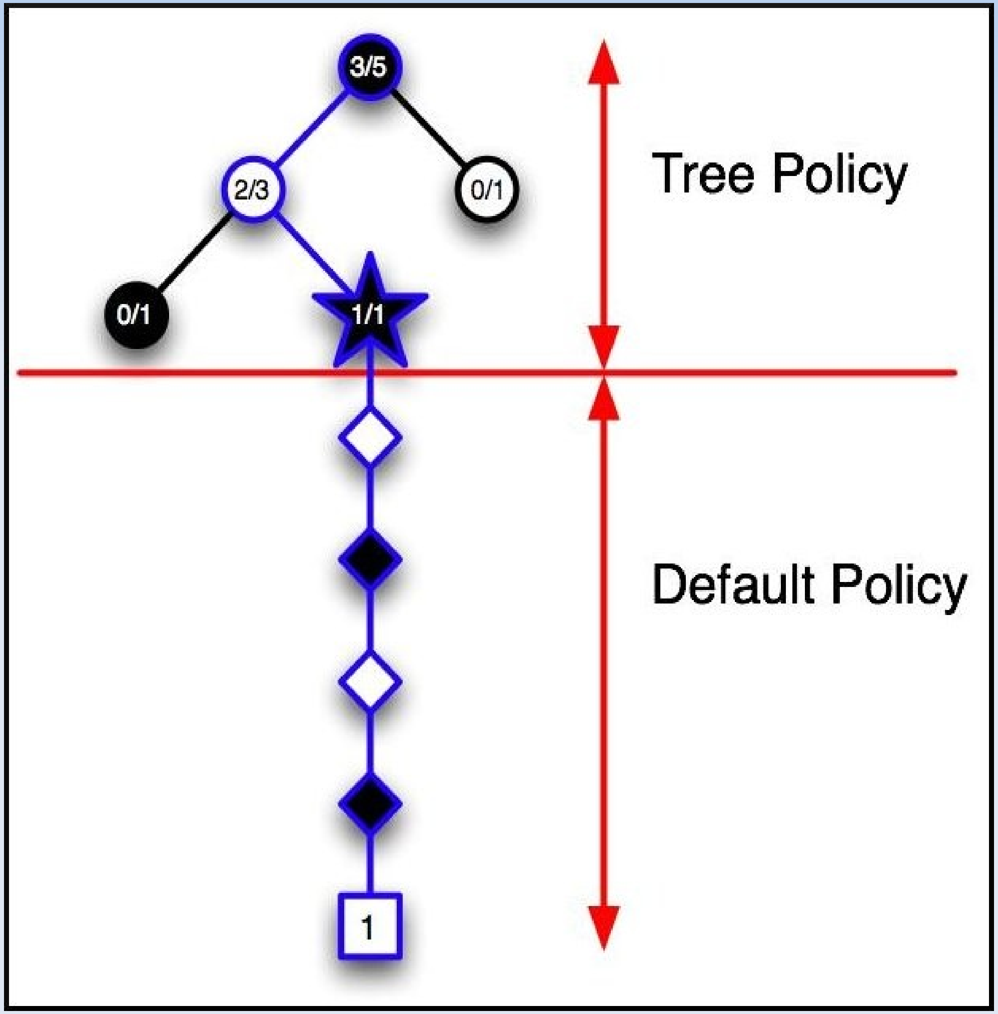
Apprentissage par Renforcement: Plan du cours

The BAAL Algorithm

=> Bandit-based Active Learner

- Simulation planning with Multi-armed bandits
- Asymetric tree growth

  More exploration for promising moves



BAAL: Exploration v. Exploitation

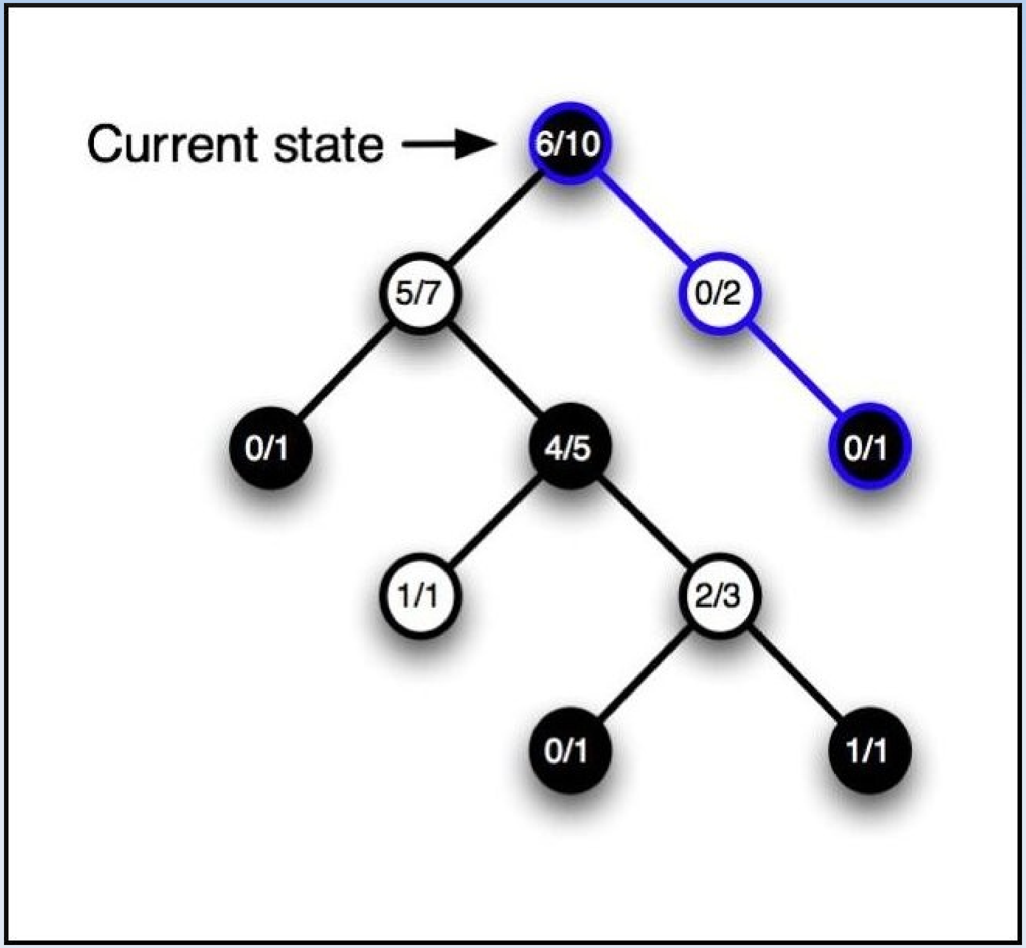
- UCB: balance exploration and exploitation

  Auer, 2002

- UCT = UCB for trees

  Kocsis&Szepesvari, 2006

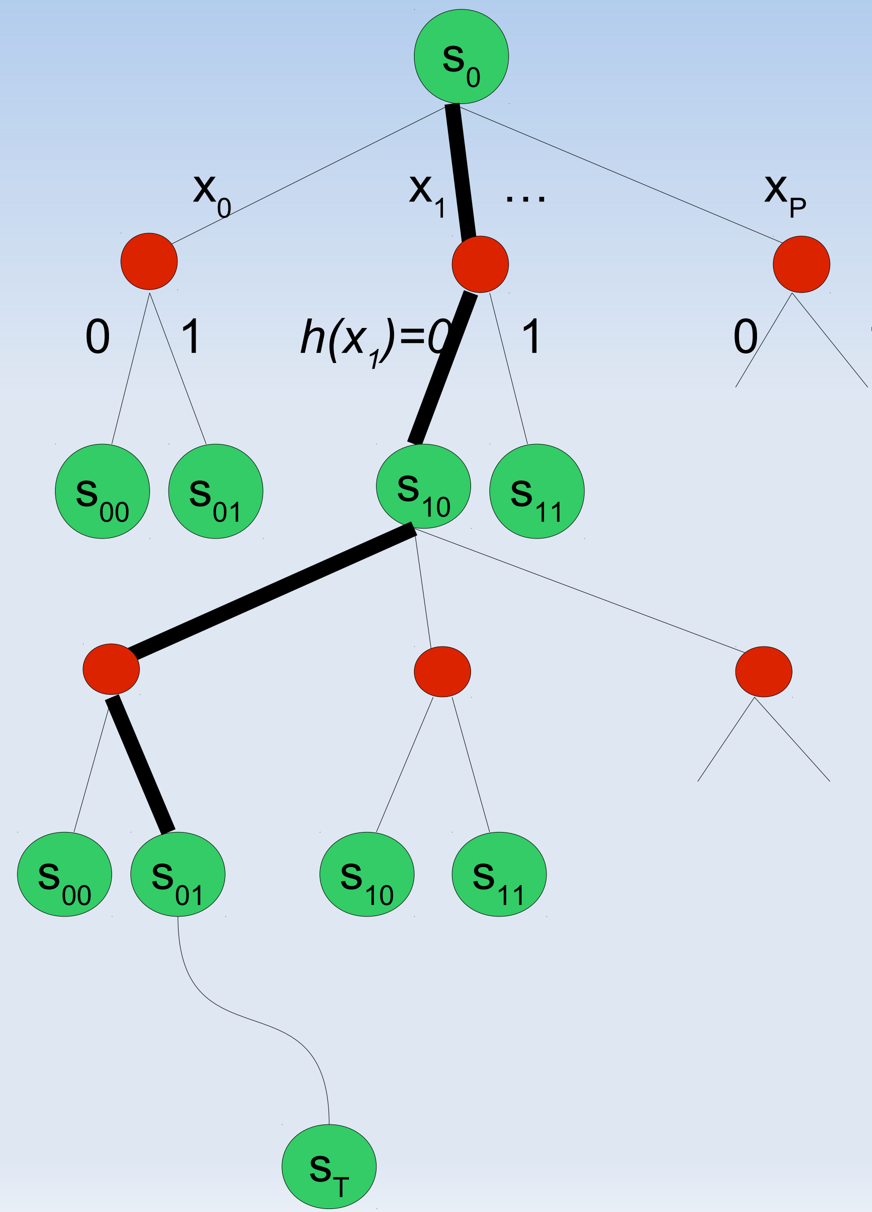
$$\hat{\mu}_i + C\sqrt{\frac{\log(\sum_j n_j)}{n_i}}$$



BAAL: Outline

$\boldsymbol{BAAL}(P_H, \mathbf{s}_0, T, N)$
for i=1 to N **do**
  h = `DrawSurrogateHypothesis`($\mathbf{s}_0$)
  Tree-Walk($\mathbf{s}_0, T, h$)
end for
Return $x = \arg\max_{x' \in \mathcal{X}} \{n(s \bigcup \{x'\})\}$

Tree-Walk($\mathbf{s}, t, h$)
Increment $n(\mathbf{s})$
if t>0 **then**
  $\mathcal{X}(\mathbf{s})$ = `ArmSet`($\mathbf{s}, n(\mathbf{s})$)
  Select $x^* = UCB(\mathbf{s}, \mathcal{X}(\mathbf{s}))$
  Get label $h(x^*)$ from surrogate
  r = Tree-Walk($\mathbf{s} \bigcup \{(x^*, h(x^*))\}, t-1, h$)
else
  Compute $r = Err(\mathcal{A}(\mathbf{s}), h)$
end if
$r(\mathbf{s}) \leftarrow (1 - \frac{1}{n(\mathbf{s})}) r(\mathbf{s}) + \frac{1}{n(\mathbf{s})}\ r$
Return r



Baal: Continuous action space

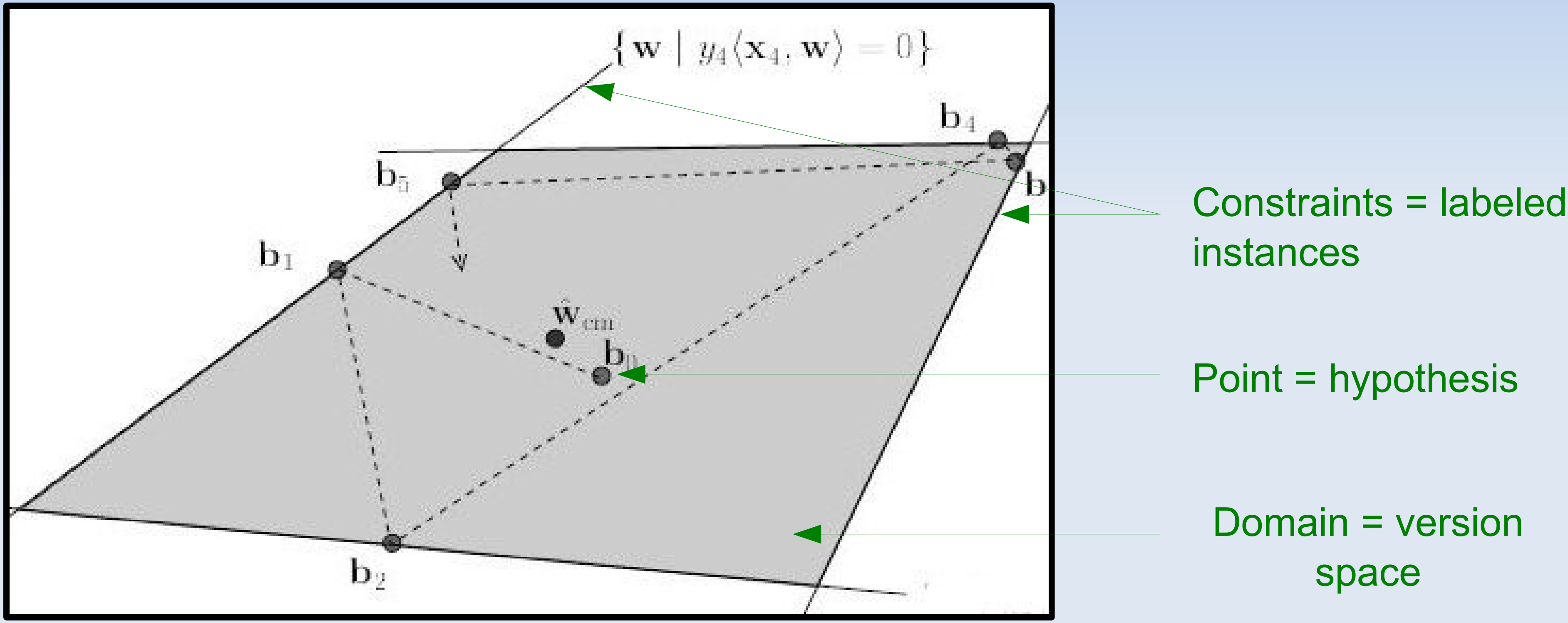
- UCB is designed for finite action spaces
- AL: action space = R^D
- Control the number of arms: *progressive widening* *# instances ~ (# visits)$^{1/4}$*

  Coulom, 2007
  Wang, Audibert, Munos, 2008
- Select new instances
  - In a random order
  - Following a given heuristic (e.g. QbC heuristic)

Baal: draw surrogate hypotheses

- *Billiard* algorithms

Rujan 97,
Comets et. al. 09, ...



- **Sound:** provably converges to uniform draw
- **Scalable** w.r.t. dimension, # constraints

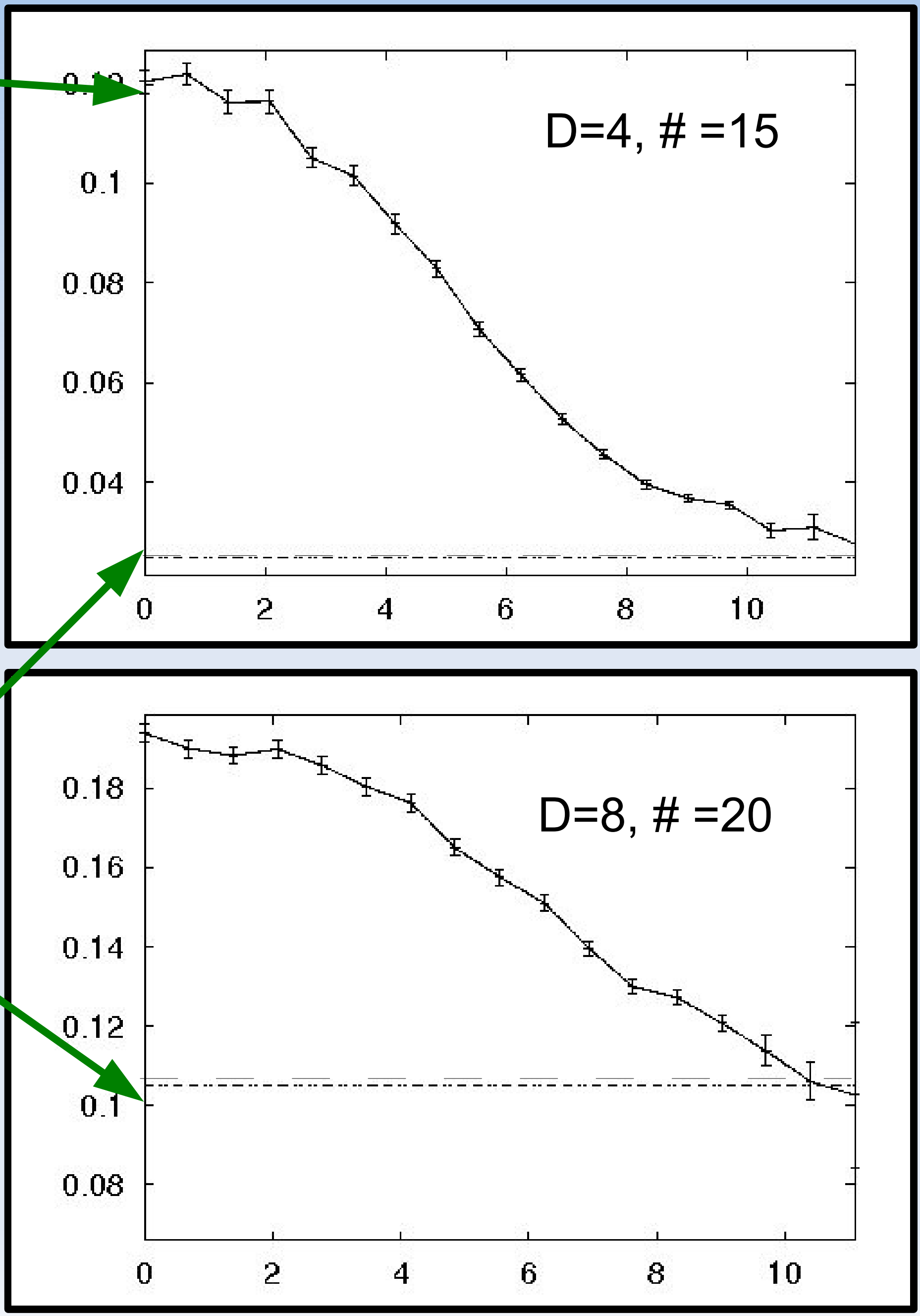
Apprentissage par Renforcement: Plan du cours

Some results

- Setting:
  - Linear sep. of R^D
  - Dimension : 4, 8
  - # queries: 15, 20
- X-axis: log(# sims)
  Y-axis: Gen. Error

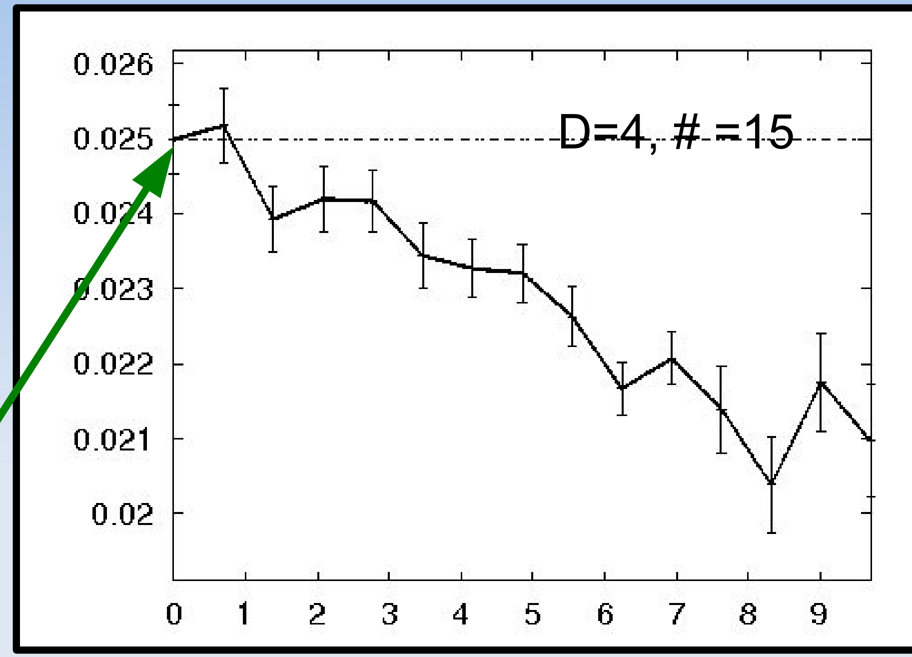
Passive learning

Almost optimal AL
(QbC-based)

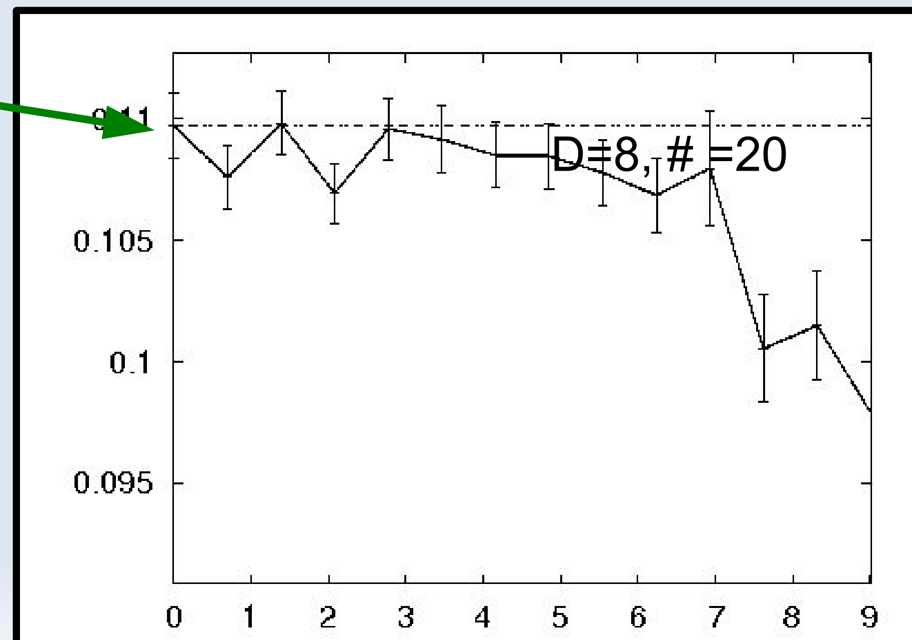


Some results

- Combining with AL criteria (inspired from QbC)
- Best of both worlds!



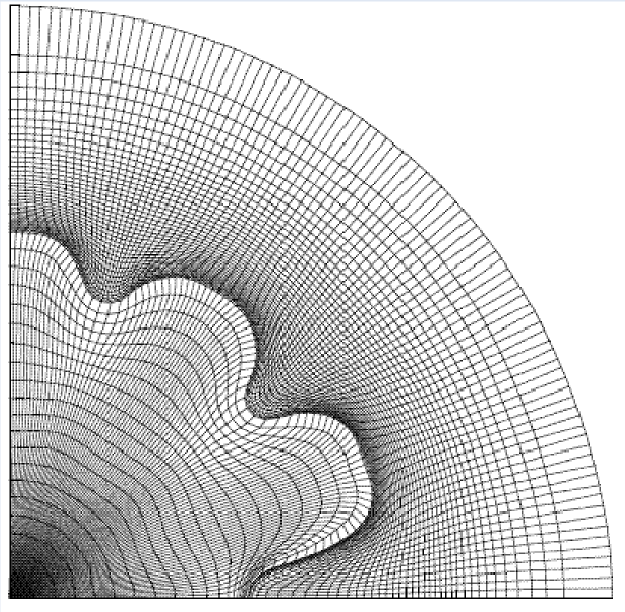
Almost optimal AL (QbC-based)



Partial Conclusion on BAAL

- A new approach to AL: *AL as a Game*
- Boosts heuristic to optimal strategy *(provably)*
- *Anytime* algorithm
- Straightforward extension to Optimization

Rolet, Sebag, Teytaud, 2009b



- **Perspectives:**
  - Kernelized Baal
  - Numerical engineering application

Apprentissage par Renforcement: Plan du cours

KDD 2009 – Orange

Targets

1. Churn
2. Appetency
3. Up-selling

Core Techniques

1. Feature Selection
2. Bounded Resources
3. Parameterless methods