Ensemble Techniques for AIXI Approximation

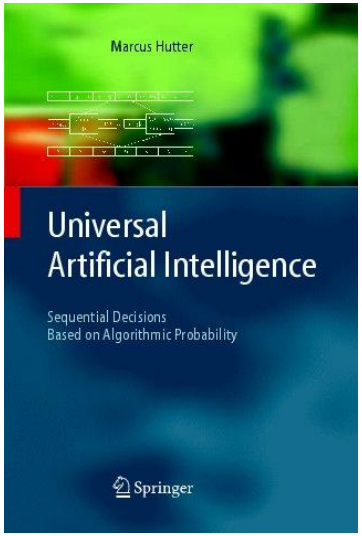
AGI @ Oxford 2012

Joel Veness
University of Alberta

Peter Sunehag, Marcus Hutter
Australian National University

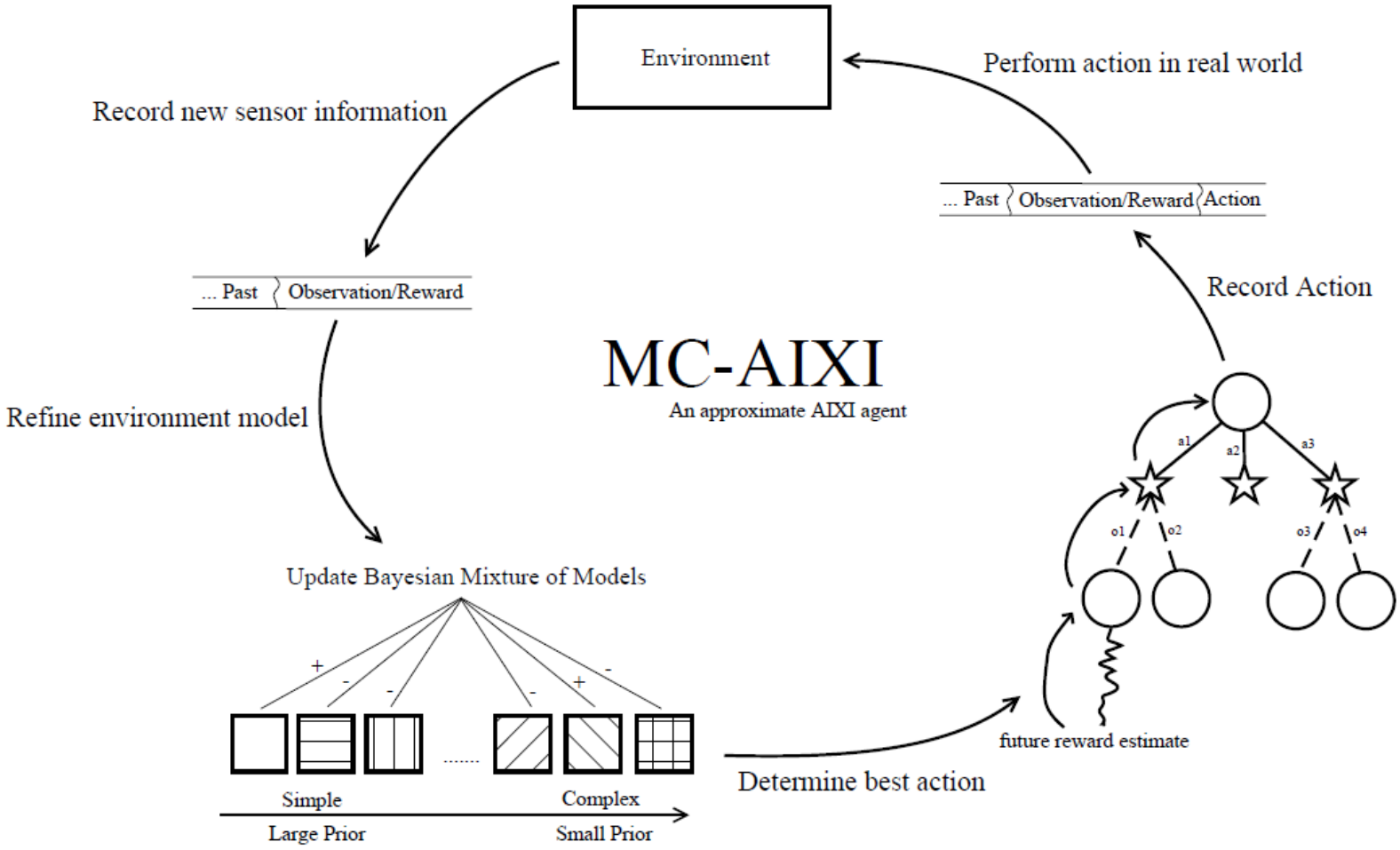
AIXI – An Optimality Notion for RL

$$\arg\max_{a_k} \sum_{o_k r_k} \ldots \max_{a_m} \sum_{o_m r_m} [r_k + \cdots + r_m] \sum_{q:U(q,a_1 \ldots a_m)=o_1 r_1 \ldots o_m r_m} 2^{-l(q)}$$



Hutter [2000,2005]

AIXI Approximation



Veness et al [2010,2011]

Model Combination Problem

- Assume have access to a set M of probabilistic models in the form:

Definition 1 *An environment ρ is a sequence of parametrized probability mass functions $\{\rho_0, \rho_1, \rho_2, \dots\}$, where $\rho_n : \mathcal{A}^n \to \mathit{Density}\,(\mathcal{X}^n)$, that satisfies*

$$\forall a_{1:n} \forall x_{<n} : \rho_{n-1}(x_{<n} \,|\, a_{<n}) = \sum_{x_n \in \mathcal{X}} \rho_n(x_{1:n} \,|\, a_{1:n}).$$

In the base case, we have $\rho_0(\epsilon \,|\, \epsilon) = 1$.

- Want a principled way to combine the models in M into some kind of new, more powerful model.

Why Ensemble Techniques?

- Lots of examples in Machine Learning, e.g. Boosting, Bagging, Voting, Model Averaging, Prediction with Expert Advice.
- Empirically quite successful for various kinds of prediction tasks. E.g. Netflix Competition, KDD-Cup, PAQ.
- Would be good to have a suite of such techniques for AIXI approximation. This talk describes 3 such methods: Weighting, Switching, Convex Mixing.

Weighting

Definition 2 *Given a finite set of environment models* $\mathcal{M} := \{\rho_1, \rho_2, \dots\}$ *and a prior weight* $w_0^\rho > 0$ *for each* $\rho \in \mathcal{M}$ *such that* $\sum_{\rho\in\mathcal{M}} w_0^\rho = 1$*, the mixture environment model is* $\xi(x_{1:n} \,|\, a_{1:n}) := \sum_{\rho\in\mathcal{M}} w_0^\rho \rho(x_{1:n} \,|\, a_{1:n})$.

- Just simple Bayesian Model Averaging over a class of probabilistic models of the environment.
- Asymptotically guarantees predictive performance (in KL sense) not much worse than the best model in the class of environments.

Weighting

- Makes a lot of sense if the model class is guaranteed to contain a good model of the environment, e.g. For AIXI or Solomonoff Induction.

- If the model class is impoverished (real world case), a better goal might be to use methods that provide competitive guarantees with respect to a larger, *induced* model class.

Switching / Tracking

- Consider a method that guarantees good performance with respect to the best *sequence* of individual model predictions.
- E.g. Maybe model 1 is good initially, but later Model 2 (after a lot of training) starts predicting really well, and so on.
- Perhaps surprisingly, one can simply use Bayesian model averaging over this larger space for free (compared with normal weighting).

Switching: Definition

Definition 3 *Given a finite set $\mathcal{M} = \{\rho_1, \ldots, \rho_N\}$, $N > 1$, of environment models and a switching sequence $\alpha = \alpha_2\alpha_3 \ldots \in [0,1]^\infty$, for all $n \in \mathbb{N}$, for all $x_{1:n} \in \mathcal{X}^n$, the switching environment model with respect to $\mathcal{M}$ and α is defined as*

$$\tau_\alpha(x_{1:n} \,|\, a_{1:n}) := \sum_{i_{1:n} \in \mathcal{I}_n(\mathcal{M})} w_\alpha(i_{1:n}) \prod_{k=1}^{n} \rho_{i_k}(x_k \,|\, a x_{<k} a_k)$$

where $\mathcal{I}_n(\mathcal{M}) := \{1, 2, \ldots, N\}^n$ and the prior over model sequences is recursively defined by

$$w_\alpha(i_{1:n}) := \begin{cases} 1 & \text{if} \quad i_{1:n} = \epsilon \\ \frac{1}{N} & \text{if} \quad n = 1 \\ w_\alpha(i_{<n}) \times \left((1-\alpha_n)\mathbb{I}[i_n = i_{n-1}] + \frac{\alpha_n}{N-1}\mathbb{I}[i_n \neq i_{n-1}] \right) & \text{otherwise,} \end{cases}$$

(Minor) generalisation of FixedShare [Herbster 1998] using logarithmic loss to probabilistic agent setting.

Switching: Algorithm

Algorithm 1 SwitchMixture - $\tau_\alpha(x_{1:n} \,|\, a_{1:n})$

Require: A finite model class $\mathcal{M} = \{\rho_1, \ldots, \rho_N\}$ such that $N > 1$
Require: A weight vector $(w_1, \ldots, w_N) \in \mathbb{R}^N$, with $w_i = \frac{1}{N}$ for $1 \leq i \leq N$
Require: A switching sequence $\alpha_2, \alpha_3, \ldots, \alpha_n$

$r \leftarrow 1$
for $i = 1$ to n **do**
  $r \leftarrow \sum_{j=1}^{N} w_j \rho_j(x_i \,|\, a x_{<i} a_i)$
  $k \leftarrow (1 - \alpha_{i+1})N - 1$
  for $j = 1$ to N **do**
    $w_j \leftarrow \frac{1}{N-1} \left[\alpha_{i+1} r + k w_j \rho_j(x_i \,|\, a x_{<i} a_i)\right]$
  end for
end for
return r

Switching: Justification

Theorem 1 *Given a base model class $\mathcal{M}$ and switch rate $\alpha_t := \frac{1}{t}$ for $t \in \mathbb{N}$, for all $n \in \mathbb{N}$, for all $i_{1:n} \in \mathcal{I}_n(\mathcal{M})$,*

$$-\log_2 \tau_\alpha(x_{1:n} \,|\, a_{1:n}) \leq (m(i_{1:n}) + 1)\,[\log_2 |\mathcal{M}| + \log_2 n] - \log_2 \rho_{i_{1:n}}(x_{1:n} \,|\, a_{1:n}).$$

- Can compete with best *rarely changing* sequence of models.
- Weighting is a special case of rarely changing, so the method is more general, but comes with the cost of O(log n) redundancy instead of O(1).

Convex Mixtures

- Consider a convex combination of model predictions. Clearly more powerful than weighting.

Definition 4 *Given a finite set of ϵ-positive environment models $\mathcal{M}$ and a sequence of weights $\lambda := \{\lambda_1, \lambda_2, \ldots\}$, where each $\lambda_i := \{ \lambda_i^\rho \}_{\rho \in \mathcal{M}}$ such that $\lambda_i^\rho \in \mathbb{R}$, $\lambda_i^\rho \geq 0$ and $\sum_{\rho \in \mathcal{M}} \lambda_i^\rho = 1$ for $i \in \mathbb{N}$, the convex environment model with respect to λ is defined as*

$$\nu_\lambda(x_{1:n} \,|\, a_{1:n}) := \prod_{i=1}^{n} \sum_{\rho \in \mathcal{M}} \lambda_i^\rho \; \rho(x_i \,|\, a x_{<i} a_i).$$

Convex Mixtures

- Using the code length (logarithmic) loss, one can recast the problem of finding a good set of weights to performing well with respect to an (unknown) sequence of convex loss functions.

$$\ell_n(\lambda_n\,;ax_{1:n}) := -\log_2 \sum_{\rho\in\mathcal{M}} \lambda_n^\rho\ \rho(x_n \,|\, ax_{<n}a_n)$$

- This is an instance of online convex programming [Zinkevich 2003], for which good techniques (e.g. OGD, ONS) are known.

Convex Mixing: Algorithm

Algorithm 2 ConvexMixture - $\nu_\lambda(x_{1:n}|a_{1:n})$

Require: A history $ax_{1:n} \in (\mathcal{A} \times \mathcal{X})^n$, $n \in \mathbb{N}$
Require: An initial weight vector $\boldsymbol{\lambda}_1 \in \Delta^{|\mathcal{M}|-1}$
Require: A sequence $\eta_1, \eta_2, \ldots, \eta_n$, of positive, real-valued step sizes

1: $r \leftarrow 1$
2: **for** $i = 1$ to n **do**
3: $\quad r \leftarrow r \times \sum_{\rho \in \mathcal{M}} \lambda_i^\rho \; \rho(x_i \,|\, ax_{<i}a_i)$
4: $\quad \boldsymbol{\lambda}_{i+1} = \text{SimplexProject}(\boldsymbol{\lambda}_i - \eta_i \nabla \ell_i(\boldsymbol{\lambda}_i \,;\, ax_{1:i}))$
5: **end for**
6: **return** r

Convex Mixing: Justification

- Competitive guarantee with respect to the best fixed set of weights.
- Extra generality comes at the price of $O(n^{0.5})$ redundancy.

Theorem 2 *Using Algorithm 2 with a step size of* $\eta_i = \frac{\epsilon \ln 2}{\sqrt{i}}$ *for* $1 \leq i \leq n$,

$$\max_{\lambda_* \in \Delta^{|\mathcal{M}|-1}} \left\{ \log_2 \prod_{i=1}^{n} \sum_{\rho \in \mathcal{M}} \lambda_*^\rho \rho(x_i \mid a x_{<i} a_i) \right\} - \log_2 \nu_{\hat{\lambda}}(x_{1:n} \mid a_{1:n}) \leq \frac{3|\mathcal{M}|\sqrt{n}}{\epsilon \ln 2}$$

Summary

- Discussed 3 efficient ways to combine arbitrary probabilistic agent models, building on previous work from related areas in machine learning.
- Useful building blocks for building larger scale AIXI approximations (but you'll have to wait a little longer to see them in action).
- **Marketing**: Also see "Partition Tree Weighting" at http://jveness.info for another ensemble technique currently under review.