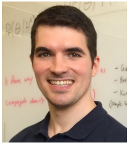
Reasoning about Hypothetical Agent Behaviours and their Parameters

Stefano Albrecht and **Peter Stone**



Learning Agents Research Group
The University of Texas at Austin

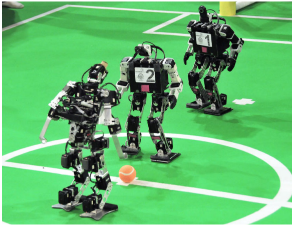
The University of Texas at Austin
Computer Science

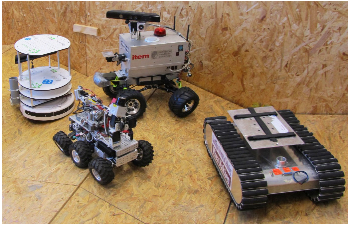
Introduction

Motivation: Ad Hoc Teamwork

Design individual agent which can collaborate effectively with other agents, without pre-coordination

- Flexibility – ability to collaborate with different teammates
- Efficiency – find effective policy quickly
- AAAI 2010 Challenge Paper (Stone et al.)





Motivation: Ad Hoc Teamwork

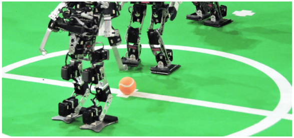
Design individual agent which can collaborate effectively with other agents, without pre-coordination

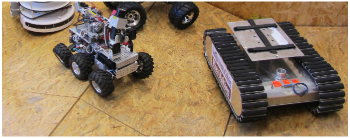
Multiagent Interaction without Prior Coordination

JAAMAS Special Issue on MIPC

AAMAS'17 Workshop on MIPC

mipc.inf.ed.ac.uk

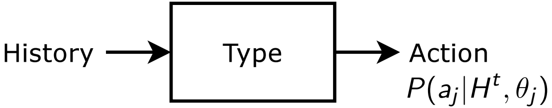




Type-Based Method

Hypothesise possible **types** of other agents:

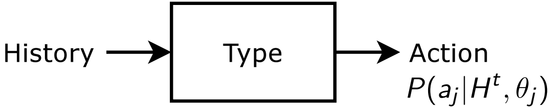
- Each type $\theta_j \in \Theta_j$ is blackbox behaviour specification



Type-Based Method

Hypothesise possible **types** of other agents:

- Each type $\theta_j \in \Theta_j$ is blackbox behaviour specification



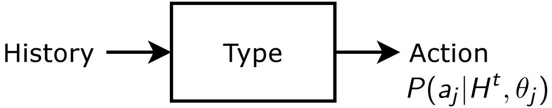
- Compute belief over types based on interaction history H^t

$$P(\theta_j|H^t) \propto P(H^t|\theta_j)P(\theta_j)$$

Type-Based Method

Hypothesise possible **types** of other agents:

- Each type $\theta_j \in \Theta_j$ is blackbox behaviour specification

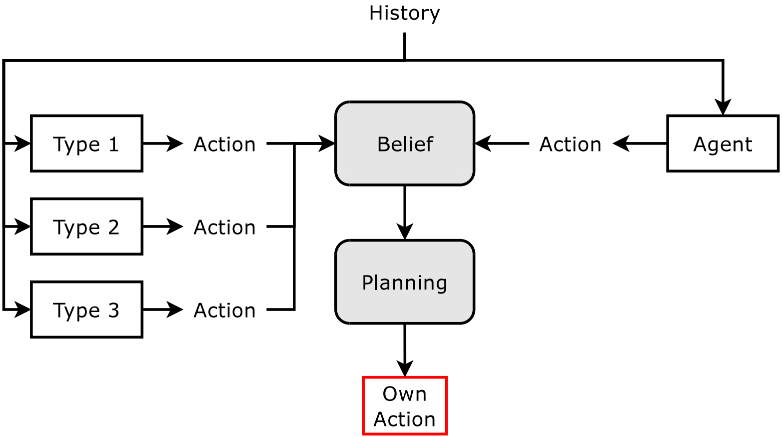


- Compute belief over types based on interaction history H^t

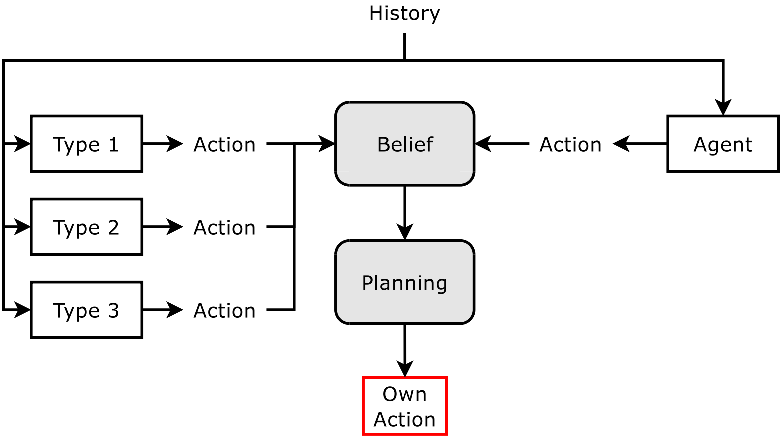
$$P(\theta_j|H^t) \propto P(H^t|\theta_j)P(\theta_j)$$

- Plan own action with respect to belief over types

Type-Based Method



Type-Based Method



- **HBA** (Albrecht & Ramamoorthy, AIJ'16)
- **PLASTIC** (Barrett & Stone, AIJ'16)

Type-Based Method and Parameters

Type-based method useful for ad hoc teamwork:

- Flexible – can hypothesise any types
- Efficient – can learn true type with few observations
- But...

Type-Based Method and Parameters

Type-based method useful for ad hoc teamwork:

- Flexible – can hypothesise any types
- Efficient – can learn true type with few observations
- But...

Limitation: method does not recognise parameters in types!

- Complex behaviours often have parameters
- If we want to reason about n parameter settings, have to store n copies of same type with different parameter settings

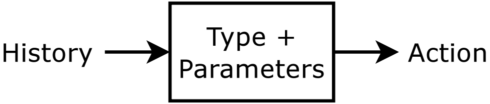
$\Rightarrow$ Inefficient, does not scale

Type-Based Method and Parameters

Goal in this work

Devise method which allows agent to reason about both:

- Relative likelihood of types *and*
- Values of **bounded continuous parameters** in types

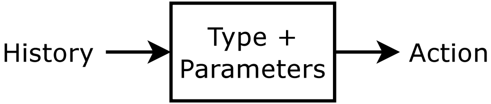


Type-Based Method and Parameters

Goal in this work

Devise method which allows agent to reason about both:

- Relative likelihood of types *and*
- Values of **bounded continuous parameters** in types



- Keep blackbox nature of types (can be any model)
- Work with any continuous parameters in types

Approach

Approach

For each $\theta_j \in \Theta_j$, maintain **parameter estimate** $p \in [p^{\min}, p^{\max}]^n$

Update estimates after new observations

Approach

For each $\theta_j \in \Theta_j$, maintain **parameter estimate** $p \in [p^{\min}, p^{\max}]^n$

Update estimates after new observations

Updating estimate incurs two computational costs:

Approach

For each $\theta_j \in \Theta_j$, maintain **parameter estimate** $p \in [p^{\text{min}}, p^{\text{max}}]^n$

Update estimates after new observations

Updating estimate incurs two computational costs:

- Computing new parameter estimate
  Types are blackboxes: must *sample* effects of parameters
  $\Rightarrow$ Need general, efficient estimation methods

Approach

For each $\theta_j \in \Theta_j$, maintain **parameter estimate** $p \in [p^{\min}, p^{\max}]^n$

Update estimates after new observations

Updating estimate incurs two computational costs:

- Computing new parameter estimate
  Types are blackboxes: must *sample* effects of parameters
  $\Rightarrow$ Need general, efficient estimation methods
- Adjusting *internal state* of type
  May depend on history of observations *and* parameter values
  $\Rightarrow$ New estimate may introduce model inconsistency

Approach: Selective Parameter Updating

Observe action a_j^{t-1} of agent j

Approach: Selective Parameter Updating

Observe action a_j^{t-1} of agent j

$\downarrow$

Select types $\Phi \subset \Theta_j$ for updating

Approach: Selective Parameter Updating

Observe action a_j^{t-1} of agent j

↓

Select types $\Phi \subset \Theta_j$ for updating

↓

For each $\theta_j \in \Phi$, update estimate $p^{t-1} \rightarrow p^t$

Approach: Selective Parameter Updating

Observe action a_j^{t-1} of agent j

$\downarrow$

Select types $\Phi \subset \Theta_j$ for updating

$\downarrow$

For each $\theta_j \in \Phi$, update estimate $p^{t-1} \rightarrow p^t$

$\downarrow$

Update beliefs:

$$P(\theta_j | H^t) \propto P(a_j^{t-1} | H^{t-1}, \theta_j, p^t) \, P(\theta_j | H^{t-1})$$

Approach: Selective Parameter Updating

Observe action a_j^{t-1} of agent j

Select types $\Phi \subset \Theta_j$ for updating

For each $\theta_j \in \Phi$, update estimate $p^{t-1} \rightarrow p^t$

Update beliefs:

$$P(\theta_j | H^t) \propto P(a_j^{t-1} | H^{t-1}, \theta_j, p^t)\, P(\theta_j | H^{t-1})$$

Plan own action

Approach: Selective Parameter Updating

Observe action a_j^{t-1} of agent j

↓

Select types $\Phi \subset \Theta_j$ for updating

↓

For each $\theta_j \in \Phi$, update estimate $p^{t-1} \rightarrow p^t$

↓

Update beliefs:

$$P(\theta_j|H^t) \propto P(a_j^{t-1}|H^{t-1}, \theta_j, p^t)\, P(\theta_j|H^{t-1})$$

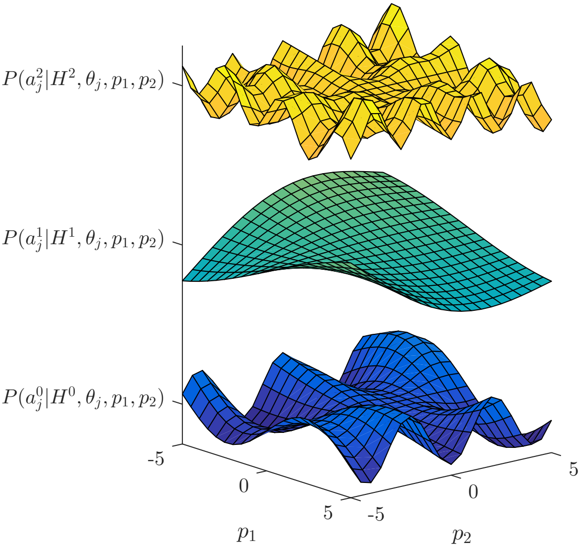
Plan own action

Updating Parameter Estimates

Given type θ_j, update parameter estimate $p^{t-1} \rightarrow p^t$

Type defines action likelihoods

$$P(a_j^{t-1} \mid H^{t-1}, \theta_j, p)$$



Approximate Bayesian Updating (ABU)

Idea: construct Bayesian update using polynomials

- Maintain prior $P(p|H^{t-1}, \theta_j)$, represented as polynomial

Approximate Bayesian Updating (ABU)

Idea: construct Bayesian update using polynomials

- Maintain prior $P(p|H^{t-1}, \theta_j)$, represented as polynomial
- Approximate likelihood $f(p) = P(a_j^{t-1}|H^{t-1}, \theta_j, p)$ as polynomial by sampling over p

Approximate Bayesian Updating (ABU)

Idea: construct Bayesian update using polynomials

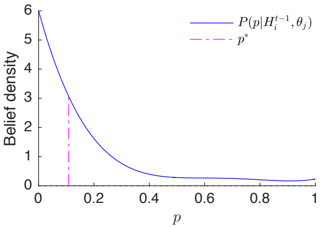
- Maintain prior $P(p|H^{t-1}, \theta_j)$, represented as polynomial
- Approximate likelihood $f(p) = P(a_j^{t-1}|H^{t-1}, \theta_j, p)$ as polynomial by sampling over p
- Take convolution of prior and likelihood, refit to original degree, normalise to get posterior $P(p|H^t, \theta_j)$

Approximate Bayesian Updating (ABU)

Idea: construct Bayesian update using polynomials

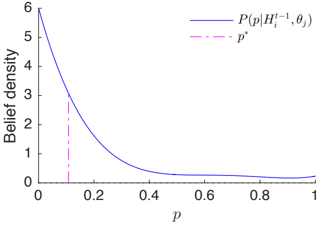
- Maintain prior $P(p|H^{t-1}, \theta_j)$, represented as polynomial
- Approximate likelihood $f(p) = P(a_j^{t-1}|H^{t-1}, \theta_j, p)$ as polynomial by sampling over p
- Take convolution of prior and likelihood, refit to original degree, normalise to get posterior $P(p|H^t, \theta_j)$
- Get parameter estimate p^t by taking maximum or sampling from posterior

Approximate Bayesian Updating (ABU) – Example



Prior

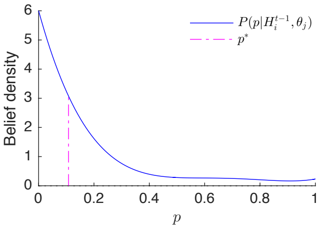
Approximate Bayesian Updating (ABU) – Example



Prior

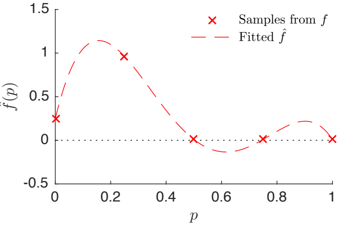
Past action a_j^{t-1}

Approximate Bayesian Updating (ABU) – Example



Prior

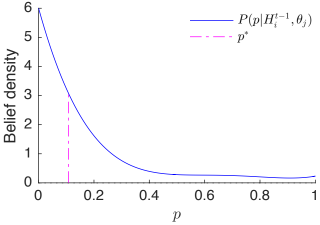
Past action a_j^{t-1}



Likelihood of a_j^{t-1} given type θ_j

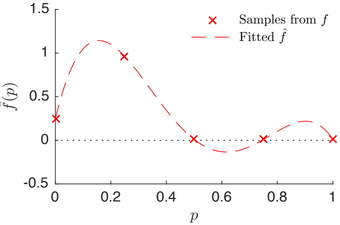
$$f(p) = P(a_j^{t-1}|H^{t-1}, \theta_j, p)$$

Approximate Bayesian Updating (ABU) – Example

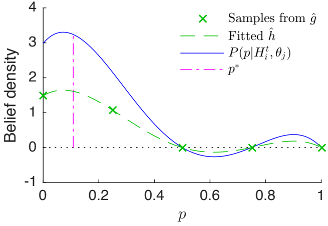


Prior

Past action a_j^{t-1}

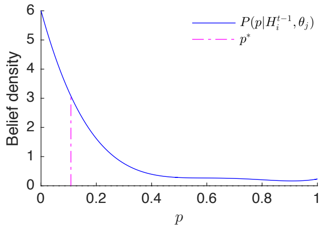


Likelihood of a_j^{t-1} given type θ_j
$f(p) = P(a_j^{t-1}|H^{t-1}, \theta_j, p)$



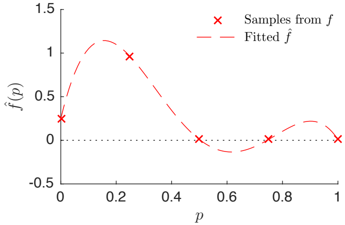
Posterior (blue)

Approximate Bayesian Updating (ABU) – Example



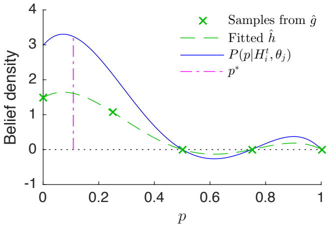
Prior

Past action a_j^{t-1}



Likelihood of a_j^{t-1} given type θ_j

$f(p) = P(a_j^{t-1}|H^{t-1}, \theta_j, p)$



Posterior (blue)

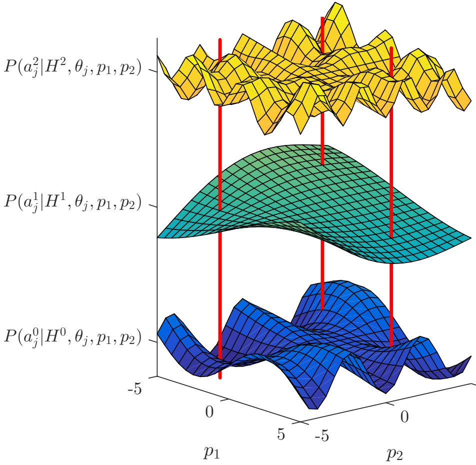
$\leftarrow$ Generate estimate p^t from posterior

Exact Global Optimisation (EGO)

Estimation as **Global Optimisation**:

$$\arg\max_{p} \prod_{\tau=1}^{t} P(a_j^{\tau-1} | H^{\tau-1}, \theta_j, p)$$

Solve with Bayesian Optimisation



Approach: Selective Parameter Updating

Observe action a_j^{t-1} of agent j

↓

Select types $\Phi \subset \Theta_j$ for updating

↓

For each $\theta_j \in \Phi$, update estimate $p^{t-1} \rightarrow p^t$

↓

Update beliefs:

$$P(\theta_j|H^t) \propto P(a_j^{t-1}|H^{t-1}, \theta_j, p^t)\, P(\theta_j|H^{t-1})$$

Plan own action

Selecting Types for Parameter Updates

Expensive to update all types after each observation...

Idea: let agent decide which types to update

- Focus on types which are "most useful" to update

Two selection methods:

- Posterior selection
- Bandit selection

Posterior Selection

Focus on types which are believed to be most likely

- Don't waste time on unlikely types

But: can lead to premature convergence of belief to wrong type...

- Occasionally update types which are less likely

Posterior Selection

Focus on types which are believed to be most likely

- Don't waste time on unlikely types

But: can lead to premature convergence of belief to wrong type...

- Occasionally update types which are less likely

$\boxed{\text{Tradeoff: } \textit{sample } \Phi \text{ from belief } P(\theta_j | H^{t-1})}$

Bandit Selection

Assumption: parameter estimates converge

- Focus on types which are expected to make largest leap toward convergence
- Don't waste time on estimates that wouldn't change much

Bandit Selection

Assumption: parameter estimates converge

- Focus on types which are expected to make largest leap toward convergence
- Don't waste time on estimates that wouldn't change much

Frame as multi-armed bandit problem:

- Each type θ_j is an arm
- Pulling arm (= updating type) θ_j gives reward

$$r^t = \eta^{-1} \sum_{k=1}^{n} |p_k^t - p_k^{t-1}|, \quad \eta = \sum_{k=1}^{n} p_k^{\max} - p_k^{\min}$$

- Can solve efficiently using bandit algorithm (e.g. UCB1)

Experiments

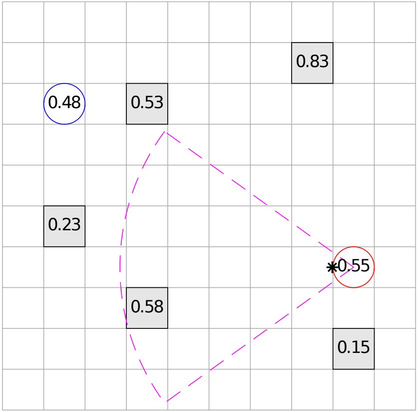
Level-Based Foraging

Blue = our agent, red = other agent

Goal: collect all items in minimal time

Agents and items have *skill levels* $\in [0, 1]$

$\Rightarrow$ Have to coordinate skills



Level-Based Foraging

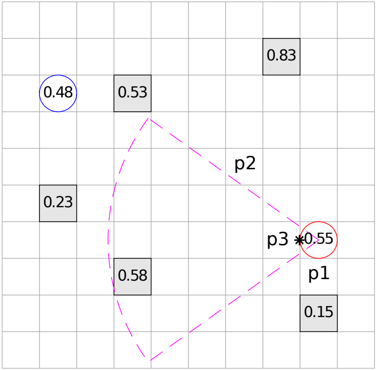
Red has one of 4 types:

θ_j^{L1}: Search for item, try to load

θ_j^{L2}: Search for *feasible* item, try to load

θ_j^{F1}: Search for agent, load item closest to agent

θ_j^{F2}: Search for agent, load closest *feasible* item



Level-Based Foraging

Red has one of 4 types:

θ_j^{L1}: Search for item, try to load

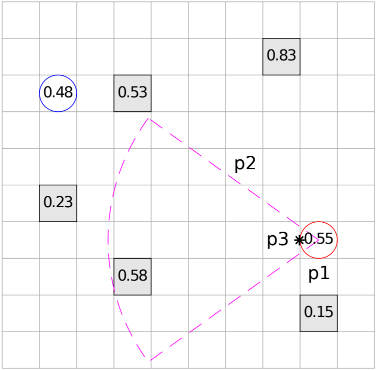
θ_j^{L2}: Search for *feasible* item, try to load

θ_j^{F1}: Search for agent, load item closest to agent

θ_j^{F2}: Search for agent, load closest *feasible* item

Each type has 3 parameters:

- level p_1
- view radius p_2
- view angle p_3



Level-Based Foraging

Red has one of 4 types:

θ_j^{L1}: Search for item, try to load

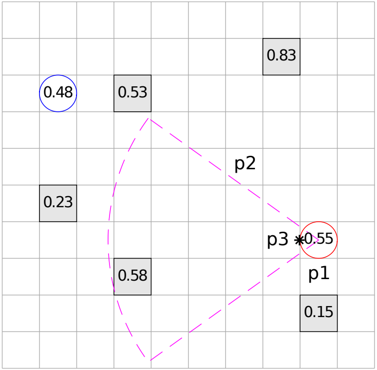
θ_j^{L2}: Search for *feasible* item, try to load

θ_j^{F1}: Search for agent, load item closest to agent

θ_j^{F2}: Search for agent, load closest *feasible* item

Each type has 3 parameters:

- level p_1
- view radius p_2
- view angle p_3



Blue does not know true type, parameter values, or meaning of parameters
Uses MCTS to plan own actions

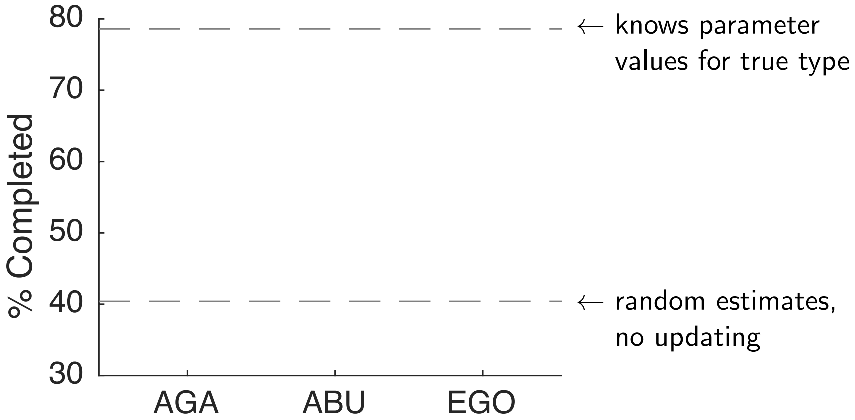
Videos

2 agents, 5 items, 10x10 world
Starting with random parameter estimates
First video without updating
Second video with updating, using bandit selection and EGO

3 agents, 10 items, 15x15 world
Starting with random parameter estimates
First video without updating
Second video with updating, using bandit selection and EGO

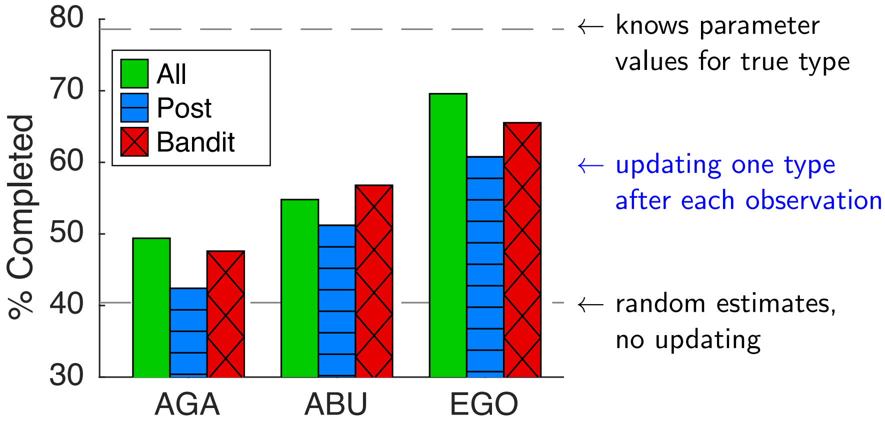
Results

15x15 world, 10 items, 3 agents
Averaged over 500 random instances

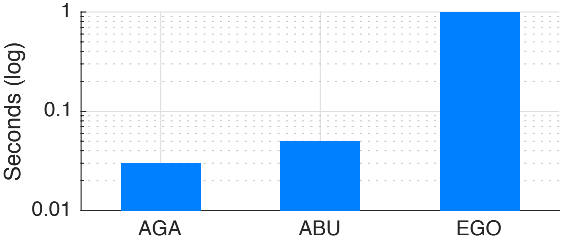


Results

15x15 world, 10 items, 3 agents
Averaged over 500 random instances

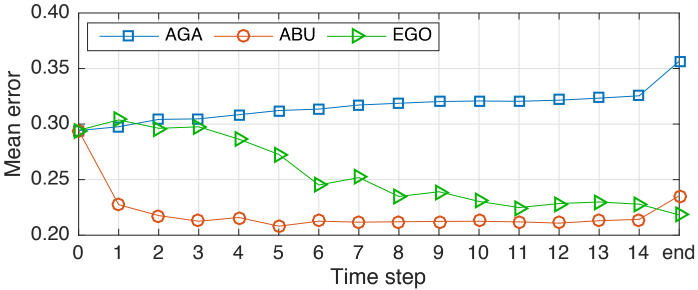


Results



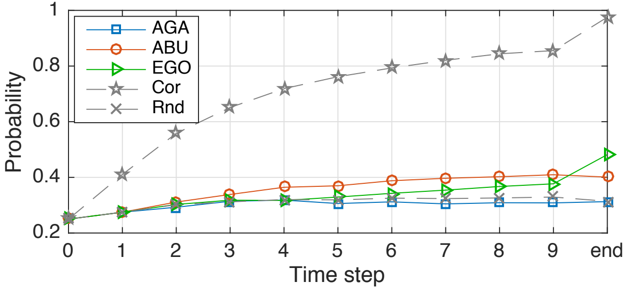
Average seconds (log-scale) needed per parameter update for single type

Results



Mean error in estimates of view radius p_2 for true type in 15x15 world (updating all types in each time step)

Results



Average belief $P(\theta_j^*|H^t)$ for true type θ_j^* in 10x10 world (updating all types in each time step)

Conclusion

- Updating single type after each observation already achieves substantial improvements over random estimates

Conclusion

- Updating single type after each observation already achieves substantial improvements over random estimates
- Posterior selection tends to select more greedily than Bandit selection, premature convergence of beliefs

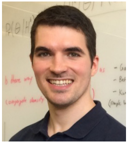
Conclusion

- Updating single type after each observation already achieves substantial improvements over random estimates
- Posterior selection tends to select more greedily than Bandit selection, premature convergence of beliefs
- EGO best estimation, can detect parameter correlation, but also most expensive

Conclusion

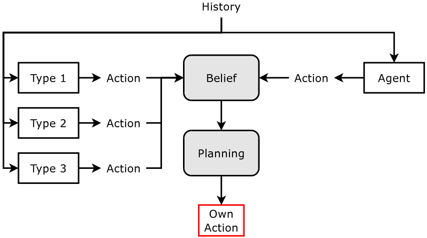
- Updating single type after each observation already achieves substantial improvements over random estimates
- Posterior selection tends to select more greedily than Bandit selection, premature convergence of beliefs
- EGO best estimation, can detect parameter correlation, but also most expensive
- **Future work:** improved methods for type selection; theoretical understanding of interaction between parameter estimates and belief evolution

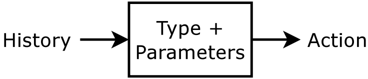
Thank you



Alexander von Humboldt
Stiftung/Foundation

Raytheon





Algorithm: Selective Parameter Updating

Given: type space Θ_j, initial belief $P(\theta_j | H^0)$ and parameter estimate p^0 for each $\theta_j \in \Theta_j$

Repeat for each $t > 0$:

1: Observe action a_j^{t-1} of agent j

2: Select a subset $\Phi \subset \Theta_j$ for parameter updates

3: For each $\theta_j \in \Phi$:

4: Obtain new parameter estimate p^t for θ_j

5: Adjust internal state of θ_j wrt p^t

6: Set $p^t = p^{t-1}$ for all $\theta_j \notin \Phi$

7: For each $\theta_j \in \Theta_j$, update belief:

$$P(\theta_j | H^t) \propto P(a_j^{t-1} | H^{t-1}, \theta_j, p^t)\, P(\theta_j | H^{t-1})$$