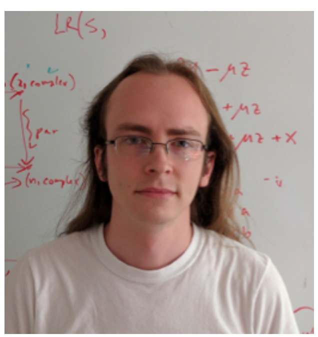
Certified Multiplicative Weights Update

Verified Learning Without Regret

Gordon Stewart

HCSS – May 10, 2017

Sam Merten
(PhD)



Alex Bagnall
(MSc)

Hard Problems in Assurance for AI

Specification

When is, e.g., a convolutional neural network for image classification "correct"?

- Performance on test set?
- Performance in real world?
- Proof of generalizability to some well-specified distribution over inputs?

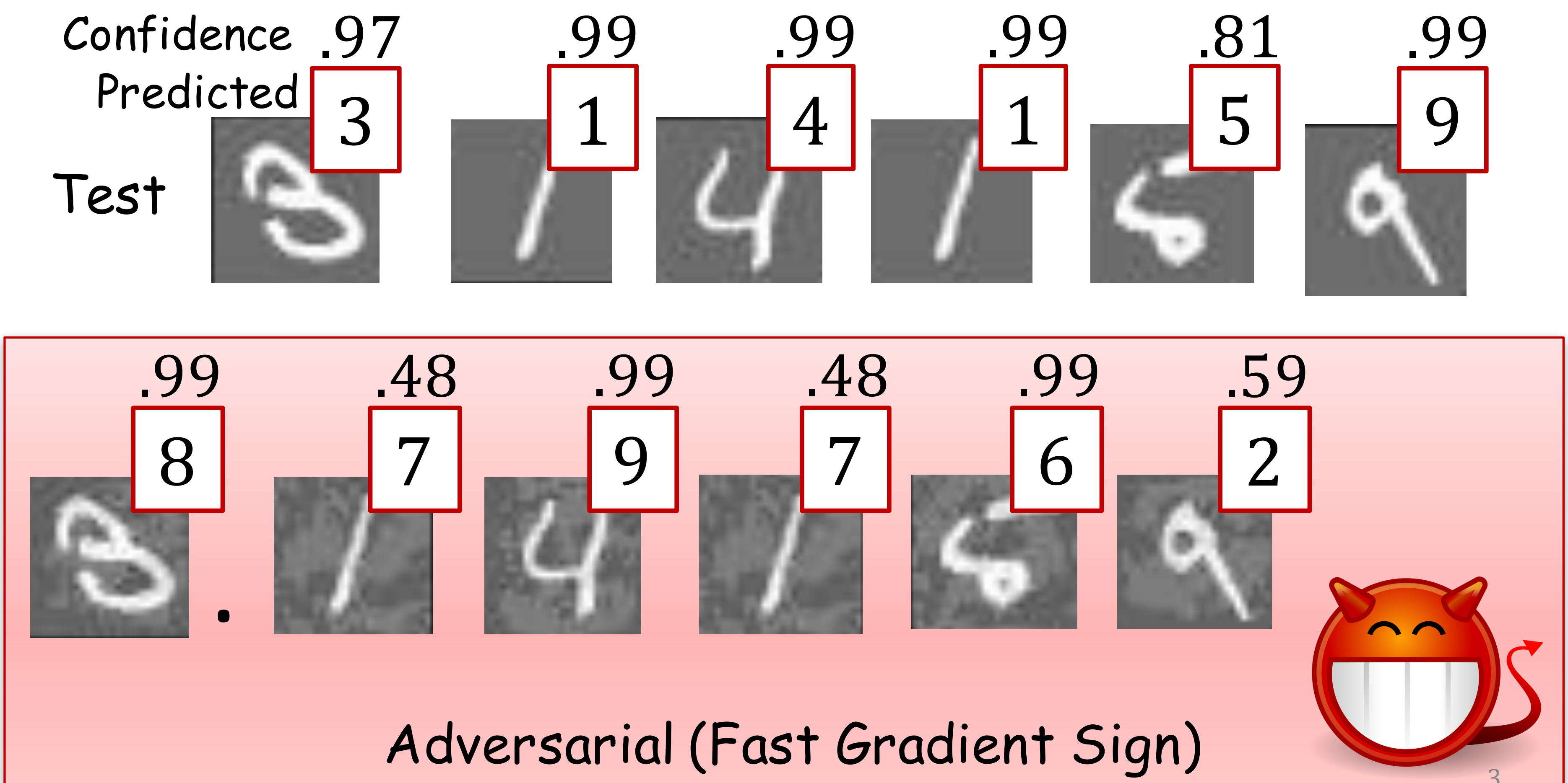
Resilience to Adversarial Input

Practitioners often (incorrectly) assume that **test set** accurately models inputs in the field.

- But quite easy to generate adversarial NN inputs that cause misclassification with high confidence [Goodfellow et al., '14]

This is not the number π...

- 28*28 (784) input features
- 1 hidden layer with 256 neurons, rectified linear unit (ReLU) activation
- softmax output **97.97% accuracy on original test data (MNIST)**



https://www.mathworks.com/examples/matlab-computer-vision/mw/vision_product-DeepLearningRCNNObjectDetectionExample-object-detection-using-deep-learning#6

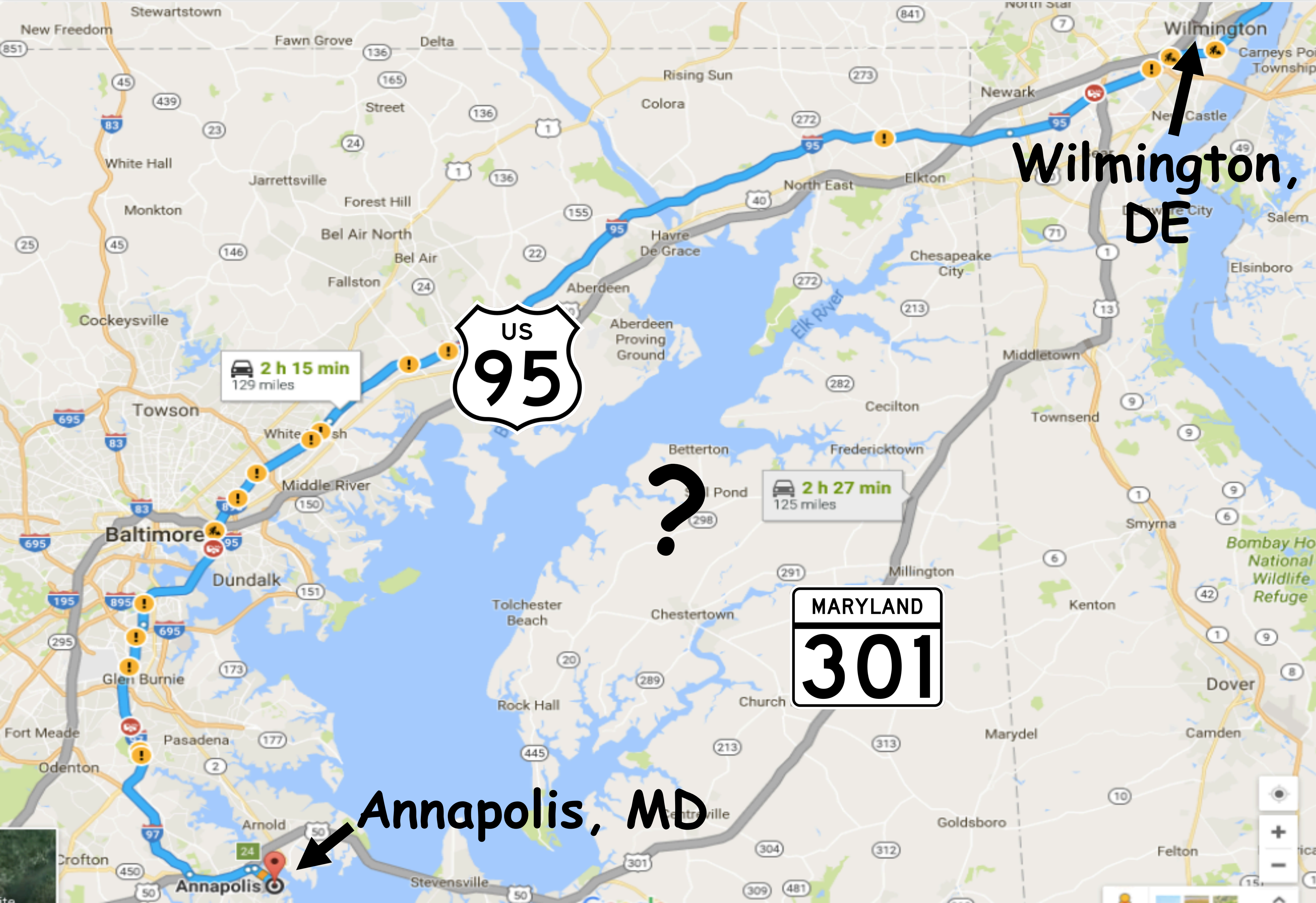
Daniel Kahneman > Quotes > Quotable Quote

"This is the essence of intuitive heuristics: when faced with a difficult question, we often answer an easier one instead, usually without noticing the substitution."

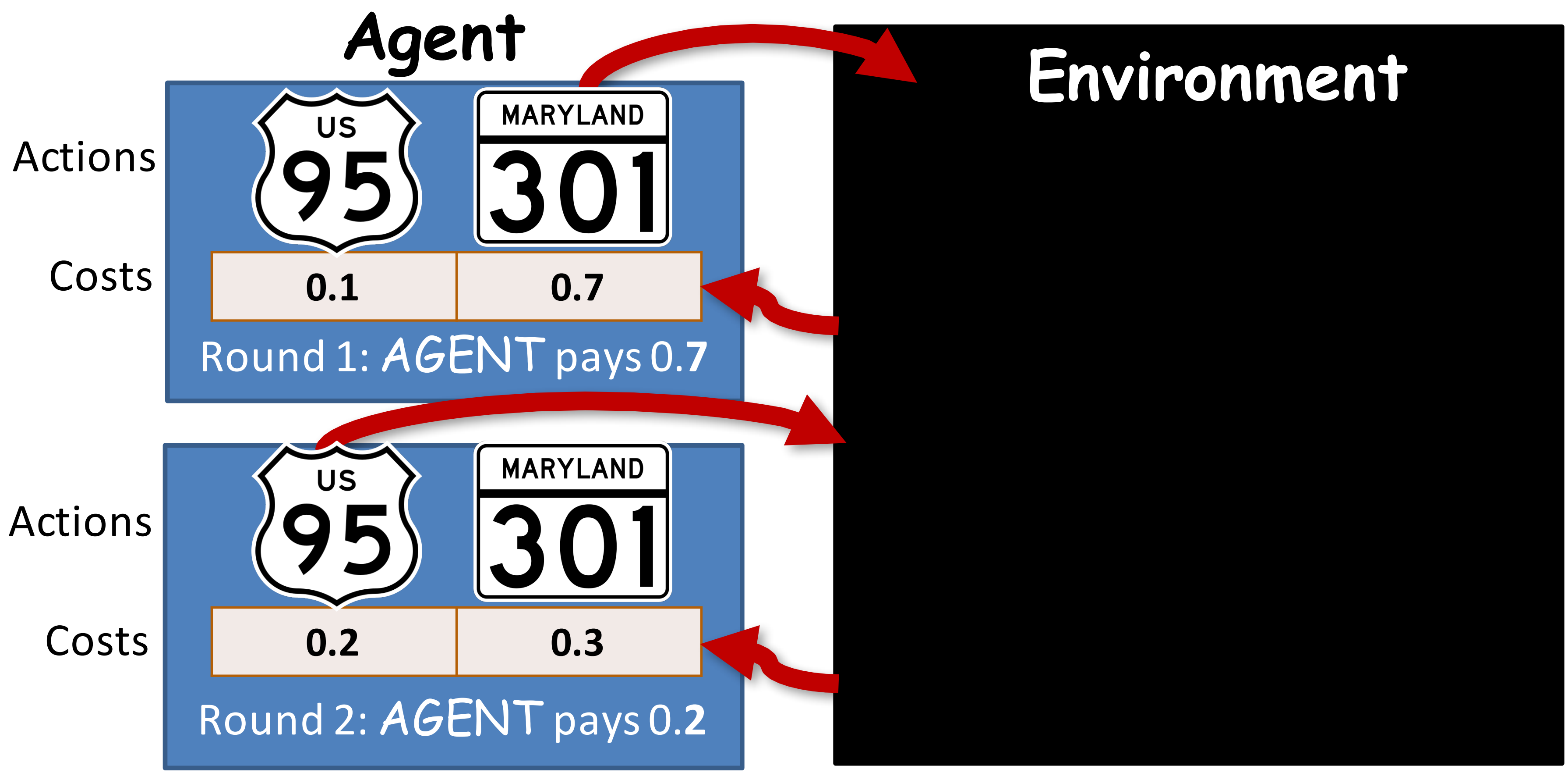
― Daniel Kahneman, Thinking, Fast and Slow

http://www.goodreads.com/quotes/754455-this-is-the-essence-of-intuitive-heuristics-when-faced-with

Online Learning in Adversarial Environments



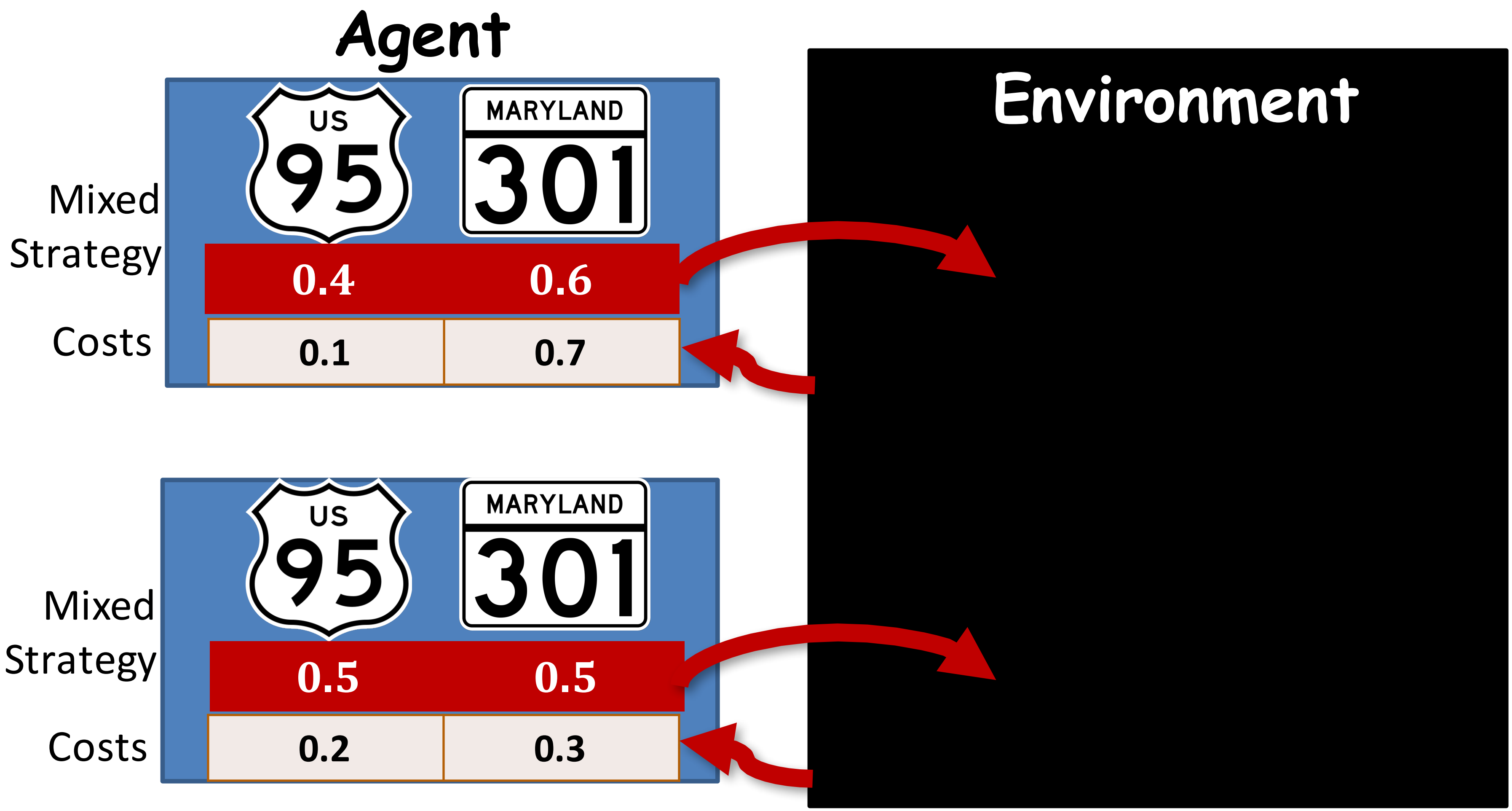
Online Learning in Adversarial Environments



Agent pays 0.9 total

Online Learning in Adversarial Environments

Agents may randomize over set of possible actions
[mixed strategies]



Good Learners Have No Regret

$$Regret^*(\boldsymbol{A}) := \left(\sum_{t=1}^{T} \boldsymbol{E}[C_t(\boldsymbol{A})] - \min_a \sum_{t=1}^{T} C_t(a)\right) / T$$

$\boldsymbol{E}[C_{tot}(A)]$

How well adaptive alg. ***A*** performs (in expectation)

against fixed action a with lowest total cost

$\min_a C_{tot}(a)$

A is ***No Regret*** if *Regret(**A**)* approaches 0 as $T \to \infty$.

****External*** **Regret**

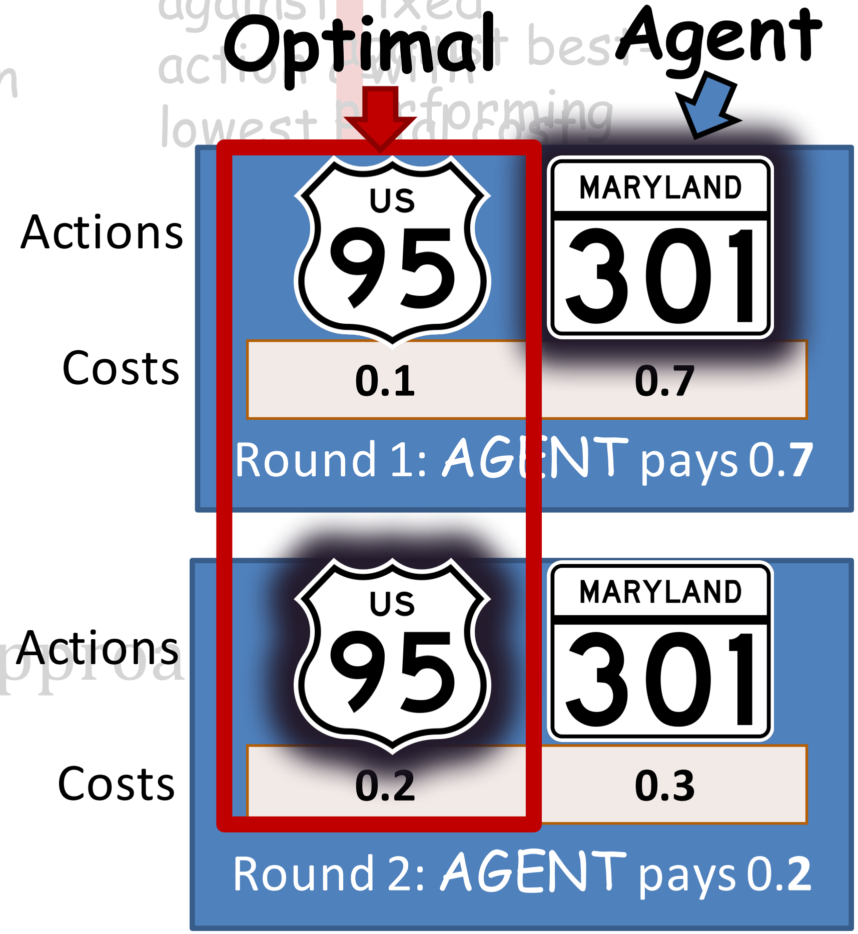
Good Learners Have No Regret

$$Regret^*(A) := \left(\sum_{t=1}^{T} E[C_t(A)] - \min_a \sum_{t=1}^{T} C_t(a)\right) / T$$

$E[C_{tot}(A)]$

How well adaptive alg. *A* performs (in expectation)

against fixed action with lowest total costs / best performing



Regret = 0.9-0.3

A is *No Regret* if *Regret(A)* approa…

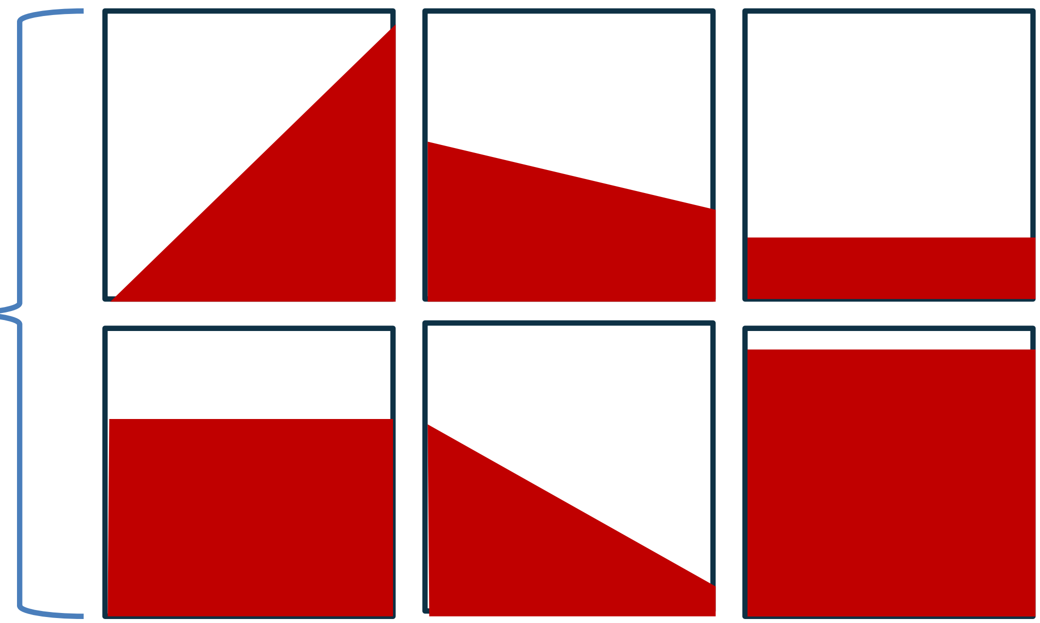
* *External* Regret

No-Regret Online Learning

Inputs:

- A set of fixed decision rules / classifiers / "experts"
- Sequence of points with unknown labels {**red**, white}

Decision Rules

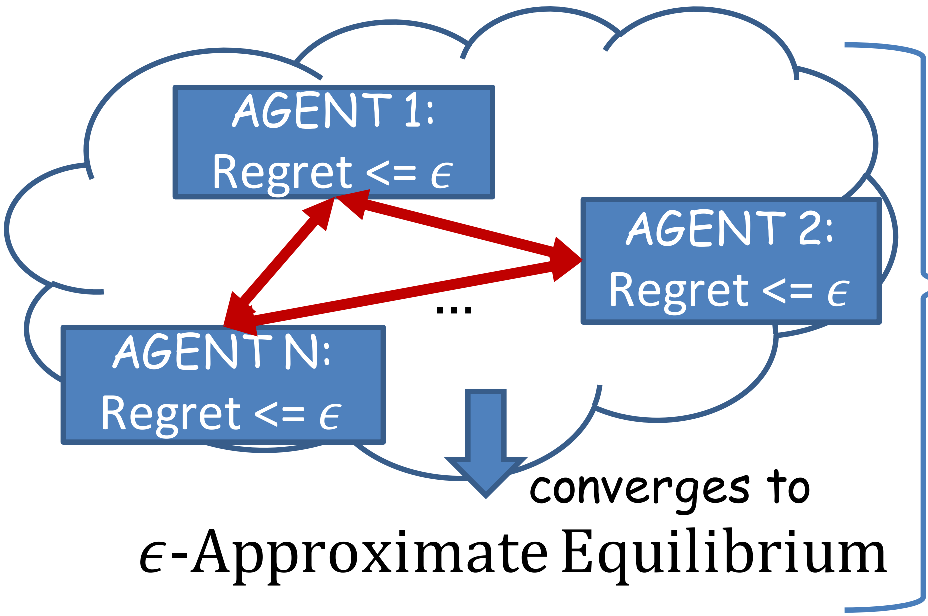


No-Regret Algorithm Outputs:

- ***online*** classification performance on input sequence nearly as good as best fixed decision rule.

No-Regret Game Dynamics

No-regret algorithms: natural ***distributed*** execution model for games, converging to ***approximate equilibria****



At time T, each AGENT has regret at most ϵ.

Intuition:

Unilateral deviation from ϵ-regret algorithm A to any fixed action a

$$\boldsymbol{E}[C_i(\boldsymbol{A}, \dots)] \leq \boldsymbol{E}[C_i(a, \dots)] + \epsilon$$

allows agent to gain at most ϵ.

*Approximate Coarse Correlated Equilibria

Multiplicative Weights (MW)

- Associate to each action $\boldsymbol{a} \in \boldsymbol{ACT}$ weight $\boldsymbol{w(a)}$ (=1)
- Choose actions by drawing from the distribution

$$p(a) = \frac{w(a)}{\sum_b w(b)}$$

- Adversary sends cost vector

$$c : A \to [-1,1]$$

- Update weights according to the following rule

$$w^{i+1}(a) = w^i(a) * (1 - \epsilon * c^i(a))$$

PARAMETER $\epsilon \in (0, ½]$

Exploration vs. Exploitation

MW Is No Regret

Theorem: MW is no regret.

$$\left(\mathbf{E}[C_{tot}(MW)] - \min_a C_{tot}(a)\right) / T \leq \epsilon + \frac{\ln |A|}{\epsilon T}$$

cumulative expected cost of MW

cost of best fixed action

number of steps

size of action space

Proof: Potential function $\Gamma^i = \sum_a w^i(a)$

Corollary: $\left(\mathbf{E}[C_{tot}(MW)] - \min_a C_{tot}(a)\right) / T \leq 2\sqrt{\frac{\ln |A|}{T}}$

Letting $0 < \epsilon = \sqrt{\frac{\ln |A|}{T}} \leq \frac{1}{2}$

A Rose By Any Other Name…

- **"Combining Expert Advice"**
- **Winnow**
  - an algorithm for learning linear classifiers
  - [Littlestone '88]

- **Weighted Majority Hedging**
  - Exponential update rule:

$$w^{i+1}(a) = w^{i}(a) * (1 - \epsilon^{c^{i}(a)})$$

- **AdaBoost / Boosting**
  - [Freund and Schapire '97]

PART I

- Assurance for AI
- No-Regret Learning & Why
- Multiplicative Weights (MW)

PART II

- Formalizing MW
- Verifying Regret

VERIFIED MW

MW Formalized

The Coq Proof Assistant

Core Files

spec	proof	comments	
390	939	35	weights.v
842	1073	80	weightslang.v
322	892	68	weightsextract.v
1554	2904	183	total

Auxiliary Files

spec	proof	comments	
300	1168	20	numerics.v
217	1015	31	dyadic.v
144	9	1	strings.v
117	87	3	dist.v
60	109	11	extrema.v
77	111	3	bigops.v
915	2499	69	total

TOTAL:

7862 LOC

Theorem: MW Is Bounded Regret

Formal:

```
Notation astar := (best_action a0 cs).
Notation OPT := (\sum_(c <- cs) c astar).
Notation OPTR := (rat_to_R OPT).
```

... more definitions and notations ...

```
Lemma perstep_weights_noregret :
((expCostsR - OPTR) / T <= epsR + ln size_A / (epsR * T))%R.
```

Informal:

$$\left(\mathbf{E}[C_{tot}(MW)] - \min_a C_{tot}(a)\right) / T \leq \epsilon + \frac{\ln|A|}{\epsilon T}$$

- $\mathbf{E}[C_{tot}(MW)]$: cumulative expected cost of MW
- $\min_a C_{tot}(a)$: cost of best fixed action
- T: number of steps
- $|A|$: size of action space

A Hierarchy of Refinements

High-Level Functional Specification

```
Definition update_weights (w:weights) (c:costs) : weights :=
  finfun (fun a : A => w a * (1 - eps * c a)).
```

REFINES

MW DSL

```
Binary Arith. Operations
  b ::= + | - | *
Expressions
  e ::= q
      | eps
      | e b e | …
Commands
  c ::= skip
      | update f | …
```

Operational Semantics

$$\vdash c, \sigma \Rightarrow c', \sigma'$$

REFINES

```
Fixpoint interp (c:com A.t) (s:cstate)
: option cstate := match c with … end.
```

Executable Interpreter

Even moderate-size proof developments (just like moderate-size software developments!) benefit from abstraction

Update Weights

```
Definition update_weights (w:weights) (c:costs) : weights :=
  finfun (fun a : A => w a * (1 - eps * c a)).
```

REFINES

```
Definition update_weights (f : A.t -> expr A.t) (s : cstate)
  : option (M.t D) :=
  M.fold
  (fun a _ acc =>
     match acc with
      | None => None
      | Some acc' =>
        match evalc (f a) s with
         | None => None
         | Some q =>
           match 0 ?= q with
            | Lt => Some (M.add a (Qred q) acc')
            | _ => None
           end end end)
  (SWeights s)
  (Some (M.empty Q)).
```

Data Refinement

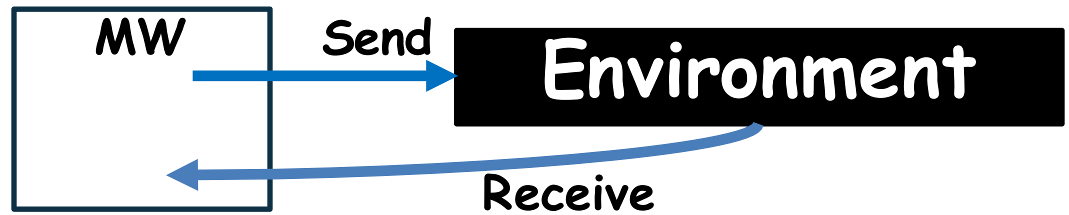
```
weights = {ffun A.t -> rat}
```

REFINES

```
Sweights s : M.t D
```

Efficient AVLTree over dyadic rational weights

Specifying the Environment

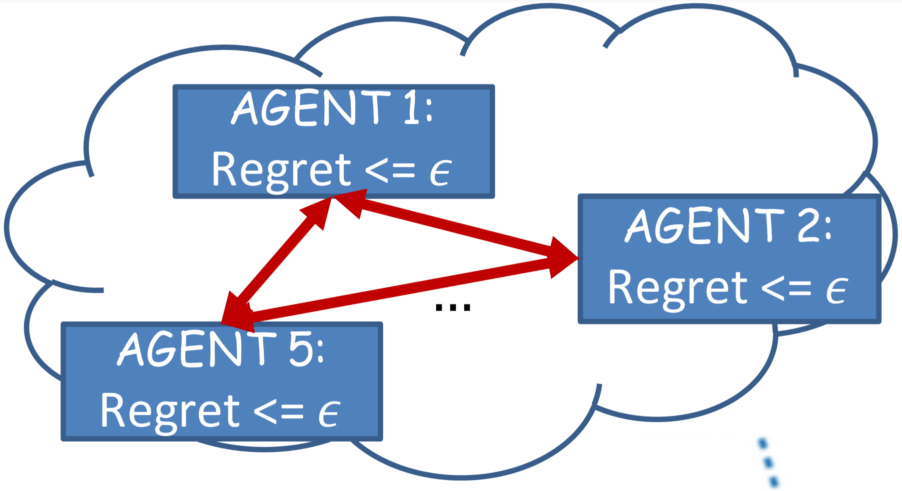


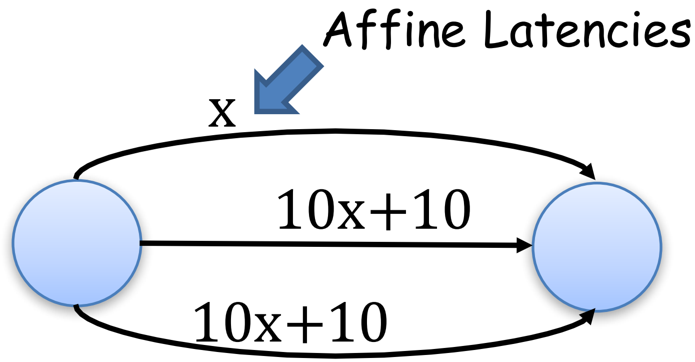

```
Class ClientOracle {A} :=
  mkOracle { T : Type (* oracle private state *)
           ; oracle_init_state : T
           ; oracle_chanty : Type
           ; oracle_bogus_chan : oracle_chanty
           ; oracle_recv : T -> oracle_chanty -> (list (A*D) * T)
           ; oracle_send : T -> list (A*D) -> (oracle_chanty * T)
           ; oracle_recv_ok : forall st ch a,
               exists d,
                 [/\ In (a,d) (oracle_recv st ch).1
                   , Dle (-D1) d & Dle d D1]
           ; oracle_recv_nodup : forall st ch,
               NoDupA (fun p q => p.1 = q.1) (oracle_recv st ch).1
           }.
```

Receive cost vector FROM environment

Send (mixed) action TO environment

Experiment: Multi-Agent Affine Routing



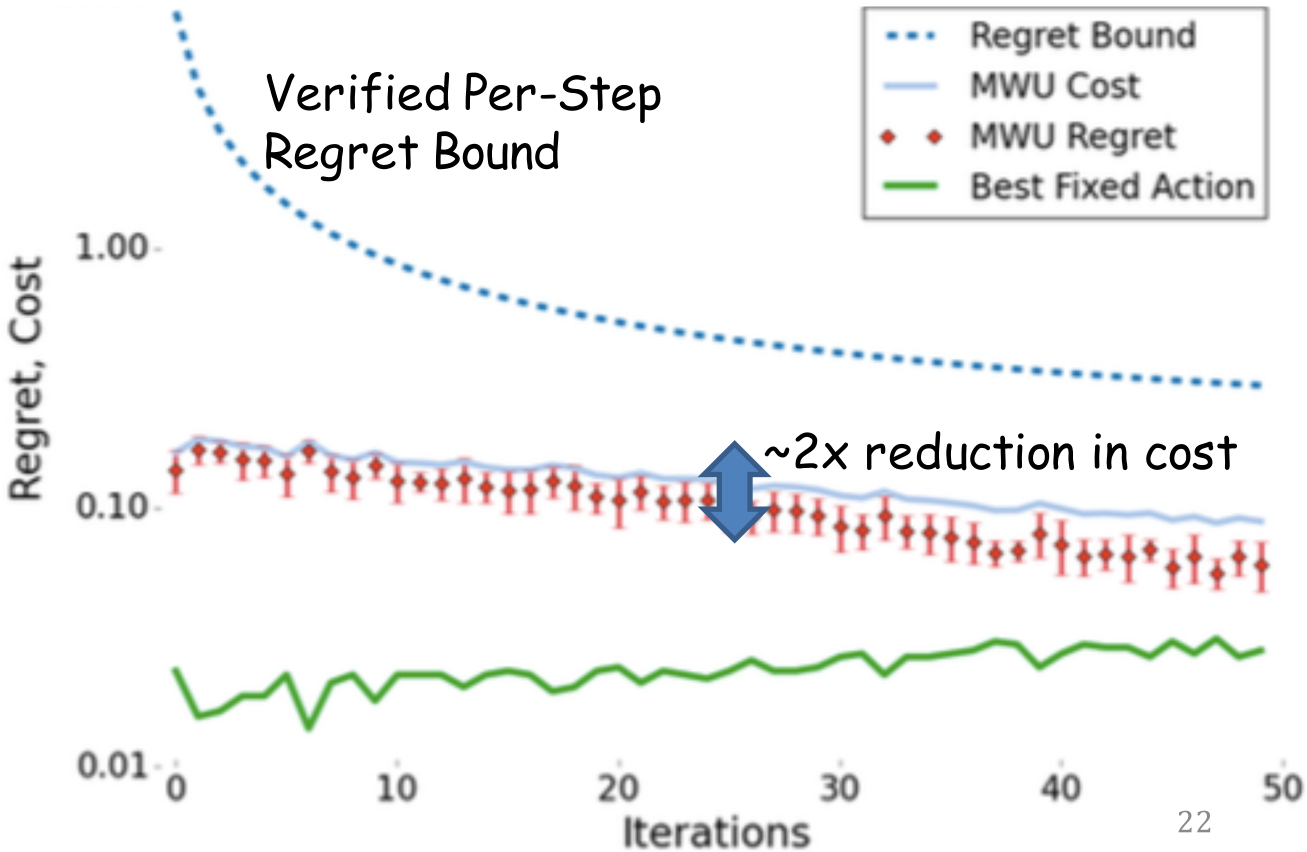


Multi-Agent Distributed Routing

- 5 players
- 50 iterations
- 10 trials
- $\epsilon = 0.1325$

Environment oracle:

Coq server + extraction to OCaml network primitives



Extensions, Connections

- **Linear Programming**
  - Verified MW as a verified LP solver
- **AdaBoost** **[Freund & Schapire '97]**
  - From weak to strong learners
- **Bandit Model**
  - revealing cost of all actions at each step imposes high communication overhead
  - assume, instead, only chosen action's cost is revealed
  - slightly more complex algorithms, slightly worse bounds, but perhaps faster in practice?
- **[Arora et al., '12]**
  - a treasure trove of additional connections!

Certified Multiplicative Weights Update

Machine-verified implementation of a simple yet powerful algorithm for online learning in adversarial environments

Proof strategy: layered program refinements, from high-level specification to executable MW

Freely available online: https://github.com/gstew5/cage

The Coq Proof Assistant

Thank You!

References

[Arora et al., '12]: The Multiplicative Weights Update Method: A Meta-Algorithm and Applications. Theory of Computing, Volume 8 (2012), pp. 121–164.

[Freund & Schapire '97]: A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. Journal of Comp. and System Sci. 55, 119-139 (1997).

[Goodfellow et al., '14]: Goodfellow, Ian J., Jonathon Shlens, and Christian Szegedy. "Explaining and harnessing adversarial examples." *arXiv preprint arXiv:1412.6572* (2014).

[Littlestone, '88]: Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning* 2.4 (1988): pp. 285-318.