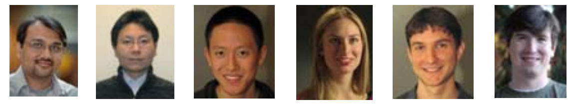
Planning to Control Crowd-Sourced Workflows

Daniel S. Weld
University of Washington

RL
Future
Challenge problems
PDDL
Overarching story – same techniques reappear (user model of uncertainty; POMDP) → efficiency!

Thanks



Crowdsourcing

- Obtaining ideas / content by **soliciting contributions** from a large group of people
- **Combine the efforts** of volunteers/part-time workers (each contributing a small portion) which adds to a relatively large or significant result

c2

How Motivate People to Contribute?

- Community
- Self-Interest

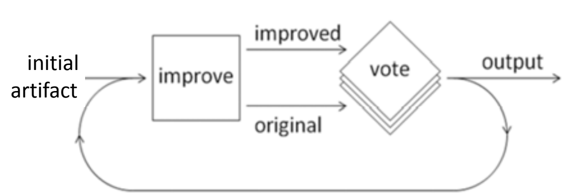
- Fun

- Money

Commonality: Large Tasks from Micro-Contributions

- Challenges
  - Small work units
  - Reliability & skill of individual workers vary
- Therefore
  - Use workflow to aggregate results & ensure quality
  - Manage workers with (unreliable) workers

Ex: Iterative Improvement



[Little et al, 2010]

Slide 4

---

c2 success stories, not motivation

cse, 6/7/2013

Iterative Improvement

[Little et al, 2010]

First version

A parial view of a pocket calculator together with some coins and a pen.

Version after 8 iterations

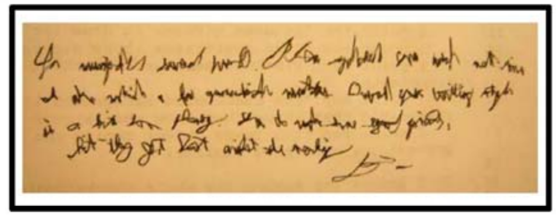
A CASIO multi-function, solar powered scientific calculator.

A blue ball point pen with a blue rubber grip and the tip extended.

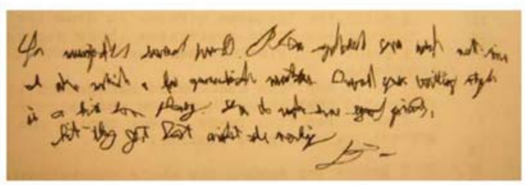
Six British coins; two of £1 value, three of 20p value and one of 1p value.

Seems to be a theme illustration for a brochure or document cover treating finance - probably personal finance.

7



[Little et al, 2010]

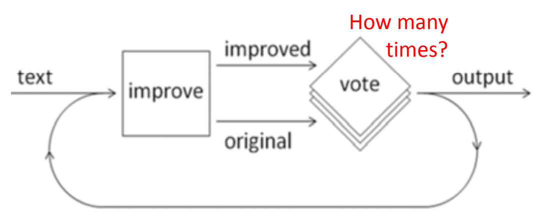


"You (misspelled) (several) (words). Please spellcheck your work next time. I also notice a few grammatical mistakes. Overall your writing style is a bit too phoney. You do make some good (points), but they got lost amidst the (writing). (signature)"

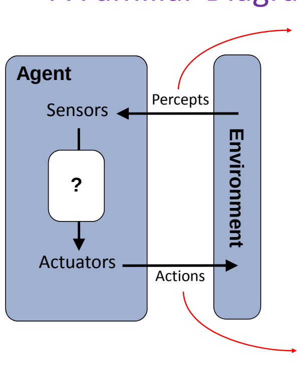
According to our ground truth, the highlighted words should be "flowery", "get", "verbiage" and "B-" respectively.

[Little et al, 2010]

Workflow Control Problem

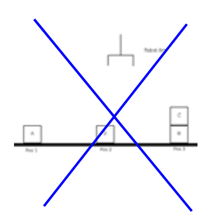


A Familiar Diagram



- Observe ballot
- Post **IMPROVE** job
- Post **BALLOT** job
- Return answer

World Representation



Artifact quality

Q ∈ [0, 1]

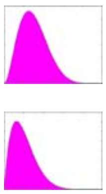
Problem difficulty

D ∈ [0, 1]

Slide 11

c1 say 'goal' seq process
cse, 6/7/2013

Belief States



Artifact quality
Q ∈ [0, 1]

Approximate with Beta distribution,
Truncated normal or
Discretized approximation

Partially-Observable MDP

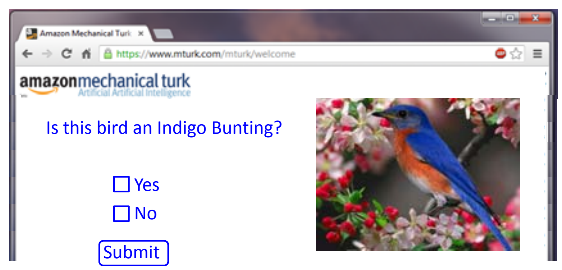
- World State: Quality of artifact(s), problem difficulty, …
- Belief State: Probability distribution over world states
- Actions: Submit jobs to labor mkt & observe results
  - Eg, improve job — prob distribution on new artifact
  - Eg, ballot job — Bayesian update on quality; EM update on difficulty, worker diligence
- Objective: Maximize **E**[R(w) –∑c]

[Dai et al, AAAI-2010]

Outline

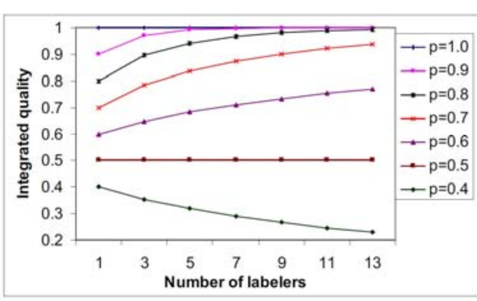
- Introduction
- Perception
- Controlling Iterative Improvement
- Controlling Taxonomy Generation
- Controlling Citizen Science
- POMDPs for the Masses

Interpreting Sensing Actions



Majority Voting

[Sheng et al, 2008; Snow et al, 2008]



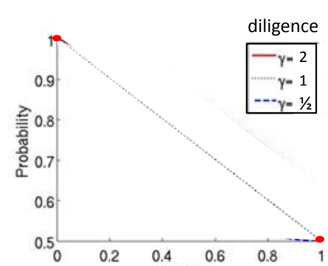
Majority vote of 8 Turkers better than expert labeling

17

Probability of a Correct Answer

$\text{accuracy}_w(d) = ½[1+(1-d)^{1/\gamma_w}]$

Assume: no malevolence

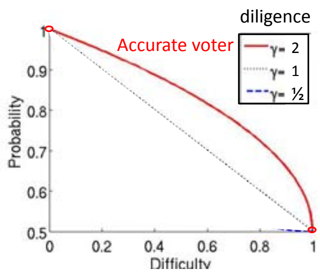


20

Probability of a Correct Answer

$accuracy_w(d) = ½[1+(1-d)^{1/\gamma_w}]$

Assume: no malevolence

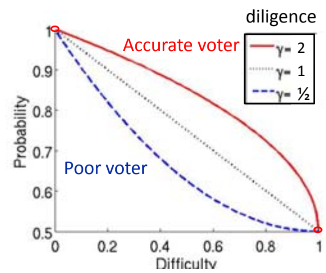


21

Probability of a Correct Answer

$accuracy_w(d) = ½[1+(1-d)^{1/\gamma_w}]$

Assume: no malevolence

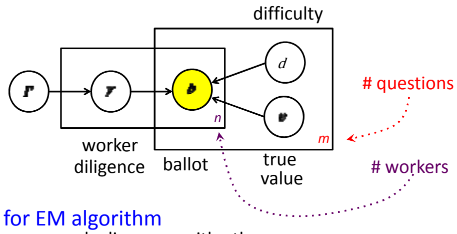


22

Unsupervised Learning

[Dawid and Sekine, 1979; Whitehill et al, 2009; Lin et al 2012; *etc*]

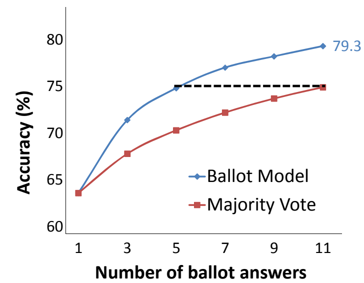
- No labeled data
- Joint estimation of all parameters: Expectation Max.



- Intuitions for EM algorithm
  – one who commonly disagrees with others: ~spammer
  – one who usually agrees with others: ~good worker
  – as we identify some good workers, we trust them more…

23

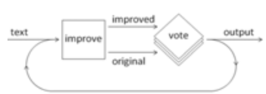
Effect of Probabilistic Model



Reduces cost by 50%

24

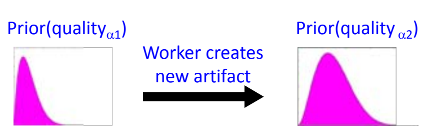
Iterative Improvement POMDP



- World State: Quality of artifact(s), problem difficulty
- Belief State: Probability distribution over world states
- Actions: Submit jobs to labor mkt & observe results
  – Eg, improve job — prob distribution on new artifact
  – Eg, ballot job — Bayesian update on quality; EM update on difficulty, worker diligence
- Objective: Maximize **E**[R(w) –∑c]

[Dai et al, AAAI-2010]

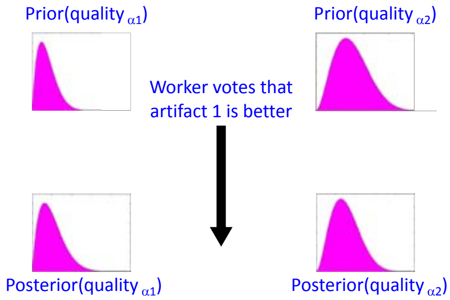
Transition Model of Improve Action



Slide 23

c4 cut plate model
cse, 6/7/2013

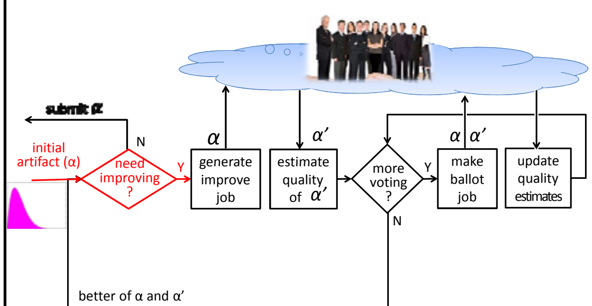
Transition Model of Ballot Action



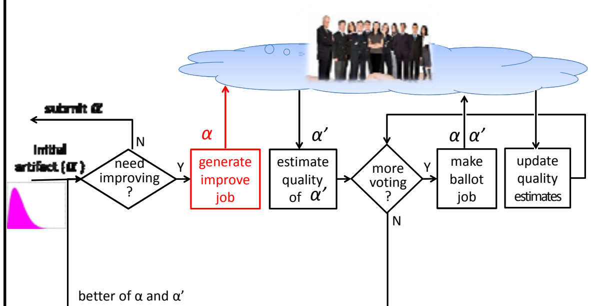
Solving the POMDP

- Beliefs over continuous world state
- Compare algorithms
  - Fixed lookahead search with beta distributions for belief states
  - ADBS – approximate belief state with fixed discretization & solve resulting MDP with VI
  - UCT on discretized space

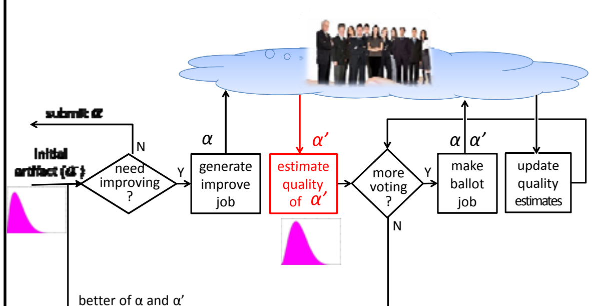
POMDP for Iterative Improvement



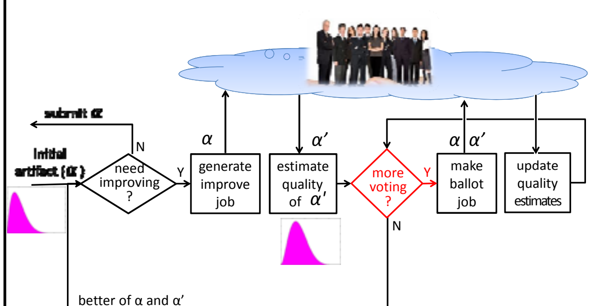
POMDP for Iterative Improvement



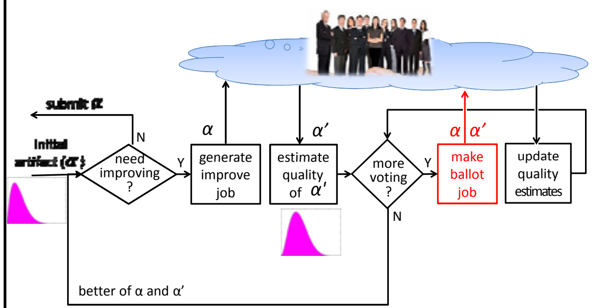
POMDP for Iterative Improvement



POMDP for Iterative Improvement

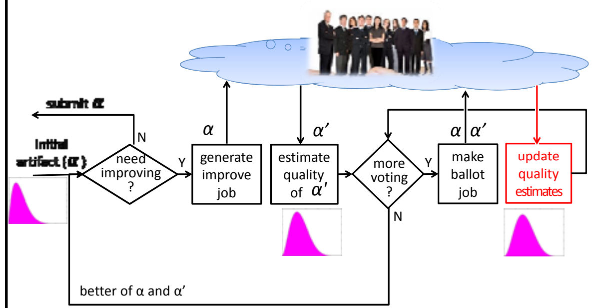


POMDP for Iterative Improvement



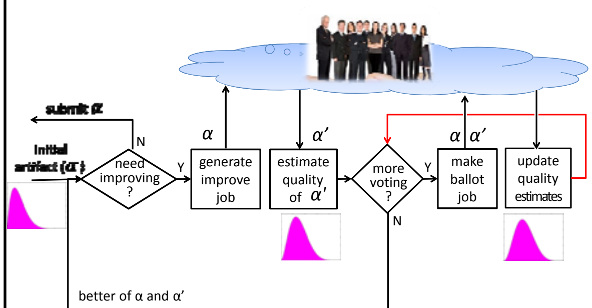
33

POMDP for Iterative Improvement



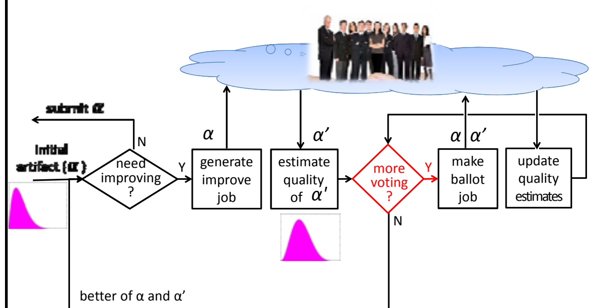
34

POMDP for Iterative Improvement



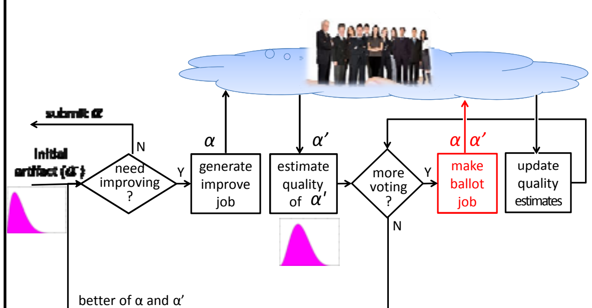
35

POMDP for Iterative Improvement



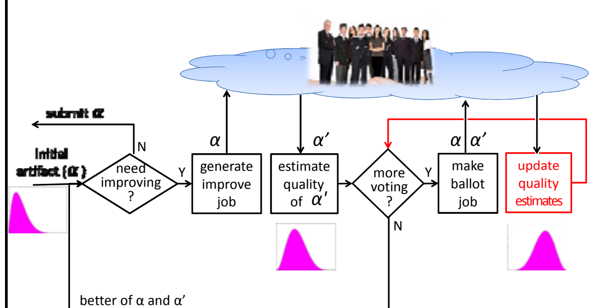
36

POMDP for Iterative Improvement



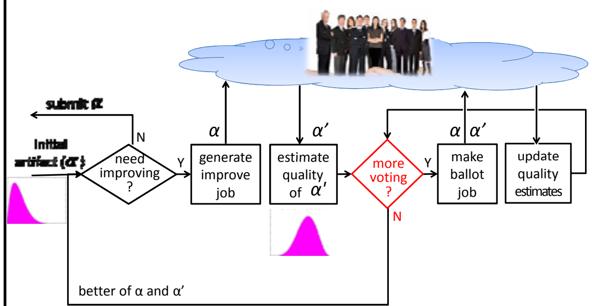
37

POMDP for Iterative Improvement



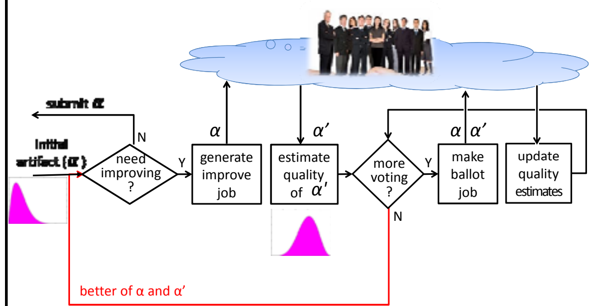
38

POMDP for Iterative Improvement



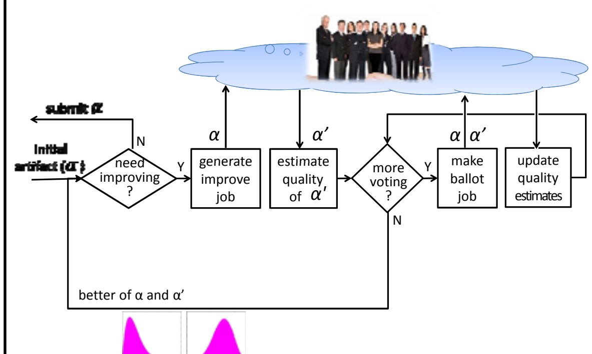
39

POMDP for Iterative Improvement



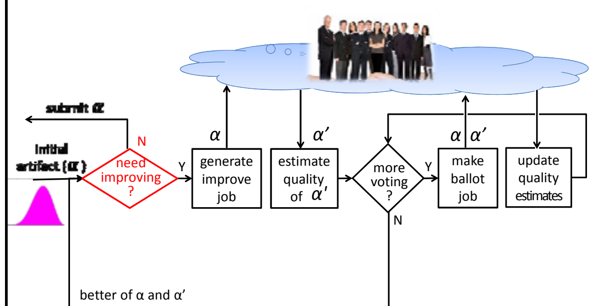
40

POMDP for Iterative Improvement



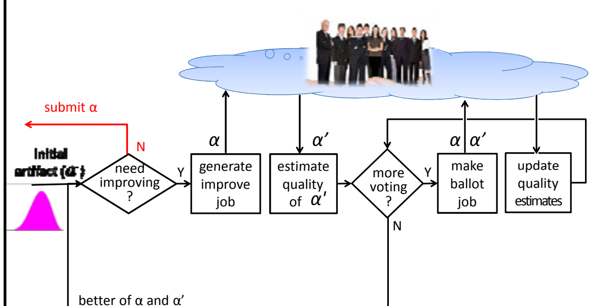
41

POMDP for Iterative Improvement



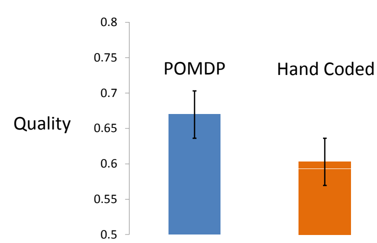
42

POMDP for Iterative Improvement



43

Comparison

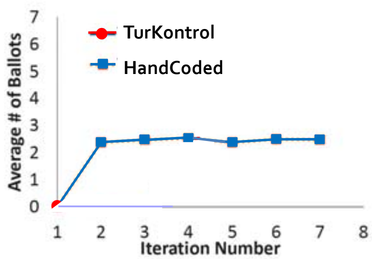


40 images, same average cost

Controlling quality: POMDP 30% cheaper

44

Observation: Ballot Use



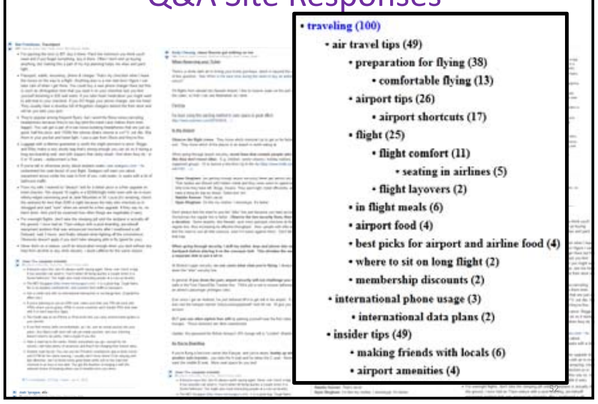
Outline

- Introduction
- Perception
- Controlling Iterative Improvement
- Controlling Taxonomy Generation
- Controlling Citizen Science
- POMDPs for the Masses

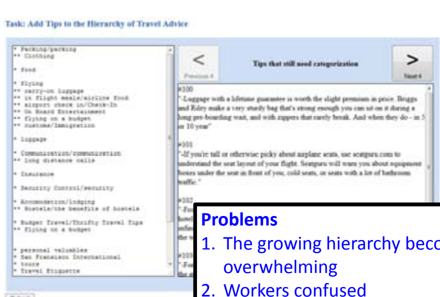
Image Data Sets

51

Q&A Site Responses



Iterative Improvement

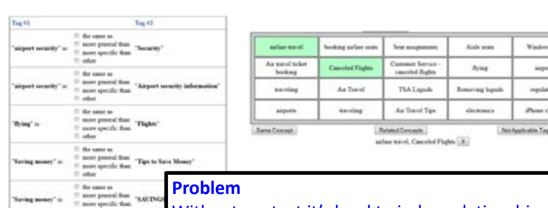


Problems
1. The growing hierarchy becomes overwhelming
2. Workers confused

Lesson: Decompose the task into smaller steps

53

Initial Approach 2: Category Comparison



Problem
Without context it's hard to judge relationships:
- **flying** *vs.* **flights**
- **TSA liquids** *vs.* **removing liquids**
- **Packing** *vs.* **what to bring**

Lesson: Don't compare abstractions to abstractions

54

Slide 47

---

c3 summarize lessons
introduce different kinds of tasks
itI seems like only kind of workflow in talkd

design patterns
binary choice, multi-label
cse, 6/7/2013

Cascade Overview

Use the ***crowd*** to:
1. Generate category names
2. Select the best categories
3. Place the data into the best categories

Use ***machines*** to:

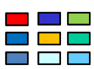
4. Infer global structure of categories

Input: 100 Random Colors

Step 0. Sample Data

Step 1. Generate Categories

For each color

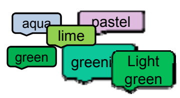


Task

What category do you suggest for this color?

greenish

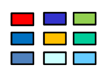
Crowd responses



This generates an initial set of category names.

Step 2. Select Best Categories

For each color



Task

What is the best category for this color?

Category	Best?
Aqua	☐
Greenish	☑
Lime	☐
Pastel	☐

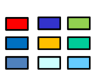
Crowd responses

Category	Votes
Aqua	1/5
Greenish	4/5
Lime	0/5
Pastel	0/5

An early filter for spam and vague categories

Step 3. Label Data

For each color and category



Categories
Green
Greenish
Yellow
Pink

Task

What categories does this belong to?

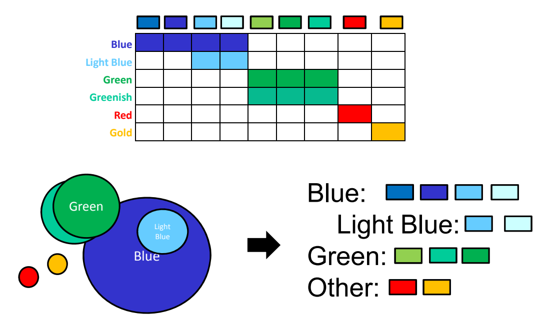
Category	Fits	Doesn't Fit
Green	☑	☐
Greenish	☑	☐
Yellow	☐	☑
Pink	☐	☑

Crowd responses

Category	Votes
Green	4/5
Greenish	5/5
Yellow	1/5
Pink	0/5

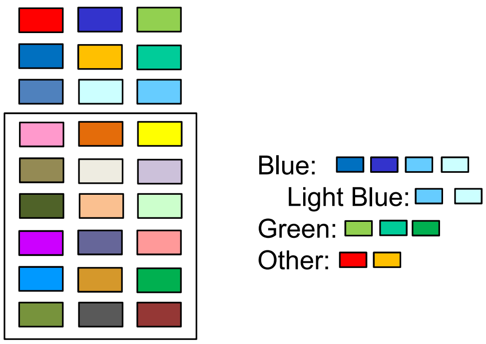
This determines category membership.

Step 4. Global Structure Inference



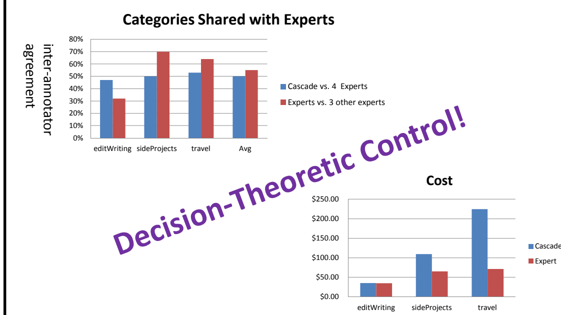
Determine parent/child relations; eliminate duplicates.

Finally, … Recurse



61

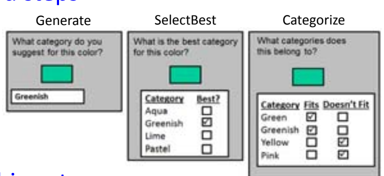
Evaluation



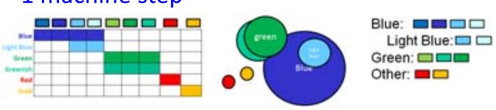
63

Why is Cascade Expensive?

3 crowd steps

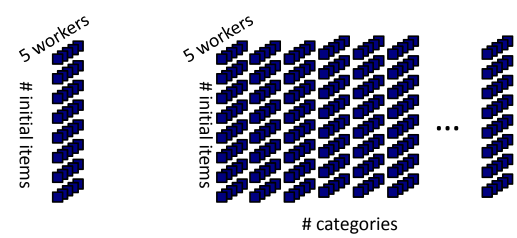


1 machine step



65

Why is Cascade Expensive?



But do we really need all these votes?
What's the best order to ask them?

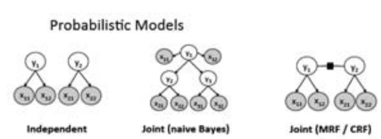
66

Agent Belief State

- Probability distribution for which labels apply
  - If a worker says X is "professional athlete"
  - Then increase confidence that X is "person"
  - Prioritize asking if X is "football player"
  - Downgrade asking if X is "vehicle"
- Must learn label co-occurrence model during plan execution

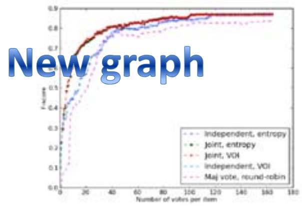
Cycle

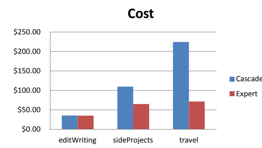
- Given new item
  - Do until confident:
    - Ask worker about most interesting label (VOI)
    - Probabilistic inference to update posteriors
  - Expectation maximization
    - Determine which labels apply
    - Update accuracy models for workers



Savings

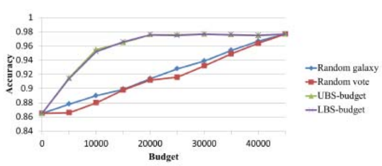
- 60/165 votes to reach same F score
- 74% savings



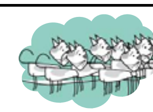


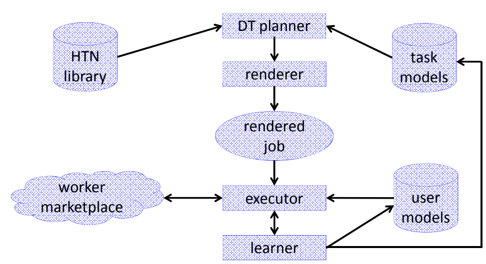
Outline

- Citizen Science

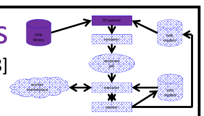


CLOWDER



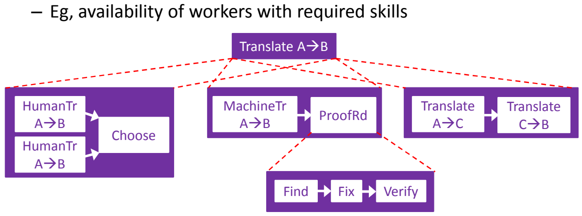


Hierarchical Task Networks



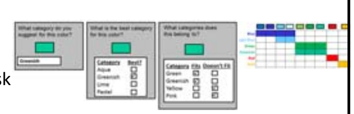
Compiles into HAM [Parr&Russell'98]

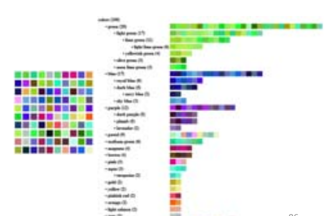
- Partially-ordered set of tasks
  - → Parallel execution
- Dynamic workflow switching [Lin AAAI12, UAI12]
- Recursive expansion
  - Preconditions & resources
  - Eg, availability of workers with required skills



Other Future Work

1. Use Cascade to organize:
   1. jobs available on oDesk
   2. eGov't suggestions
   3. Product reviews
   4. Free-form survey comments
2. Standardize "human task primitives" for future workflows





Related Work

Collaborative Taxonomies
- CardSorting
- Wikipedia

Crowdsourcing Workflows:
- TurKit [Little, UIST'10]
- Soylent [Bernstein, UIST'10]
- Mobi [Zhang, CHI'12]
- CrowdForge[Kittur, CHI'12]

Optimization of Workflows
- [Shahaf&Horvitz AAAI'10]
- [Kamar et al. AAMAS'12]

87

Conclusion

- Decision-theoretic methods for scaling crowdsourcing
  - POMDP model, EM & reinforcement learning
  - Objective: DT compiler to make techniques accessible
- Novel workflow for decentralized taxonomy construction
  - Global view from locally informed microwork

88

Barbados

World Domination?

Cool, huh? Why don't you use it too?

?!?!? My students don't really get this POMDP, RL stuff

Specifying a POMDP

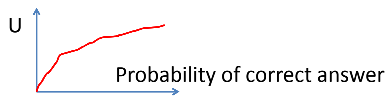
- Actions
  - ☺ Worker tasks – user has to specify anyway
- Transition & Observation Probabilities
  - ☺ Learned from experience (*vs* PDDL)
- World State
  - ☹ POMDP specific, unintuitive …→ … templates
- Utility Function
  - ☹ Implicit, F(world state) … → utility elicitation
- Control Guidance
  - ☹ Options? HAMs? MAX-Q? Basis functions? Constraints?

RELATED WORK

- PPDDL, RDDL
- Alisp [Andre *et al*, 2002]
- A^2BL [Simpkins *et al*, 2008]
- Adaptive Programs [Pinto *et al*, 2010]

Utility / Control Advice

- Users specify utility *procedurally!*
  - Utility is implicit in their control program
  - Eg, "majority vote of three people"
  - *Vs* "83% probability threshold"
  - Let alone utility curve

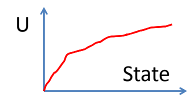


- Procedural behavior also good for roll-out policy

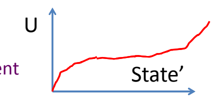
How Compose Utility?

- Want to encapsulate sub-behaviors

(define-behavior X …
    (do A …
    (choose-between B or C…
    (do iterative-improvement …)



Transformed utility fn
For iterative improvement



POAPS Primitives

A *primitive* is a ten-tuple $\langle \mathcal{D}, \mathcal{R}, \Omega, \mathcal{T}, \mathcal{O}, \mathcal{I}, \mathcal{C}, \mathcal{D}_U, \mathcal{R}_U, \mathcal{F} \rangle$, where:

- $\mathcal{D} = \mathcal{D}^1 \times \ldots \times \mathcal{D}^n$ is a set of *domain states.*
- $\mathcal{R}$ is a set of *range states.*
- Ω is a set of *observations.*
- $\mathcal{T} : \mathcal{D} \times \mathcal{R} \to [0, 1]$ is a *transition function.*
- $\mathcal{O} : \mathcal{R} \times \Omega \to [0, 1]$ is an *observation function.*
- $\mathcal{I}$ is an n-dimensional indicator vector indicating which of the $\mathcal{D}^i$ are observable.
- $\mathcal{C} : \mathcal{D} \to \mathbb{R}^+$ is a *cost function.*

Model of the function

- $\mathcal{D}_U = \mathcal{D}_U^1 \times \ldots \times \mathcal{D}_U^n$ is a set of *user domain states.*
- $\mathcal{R}_U$ is a set of *user range states.*
- $\mathcal{F} : \mathcal{D}_U \to \mathcal{R}_U$ is a *user function.*

Some Function

Primitive: *c-imp*

- $\mathcal{F}(\alpha \in \mathcal{D}_U) = \text{calltoAPI}(\alpha)$
- $\mathcal{D}_U, \mathcal{R}_U$ is the set of all artifacts α.

- $\mathcal{D} = \mathcal{R} = [0, 1]$
- $\mathcal{T}(q \in \mathcal{D}, q' \in \mathcal{R}) = P(q'|q)$
- $\mathcal{C} = \$0.05$
- No observations/observation function
- $\mathcal{I} = (0)$

POAPS LANGUAGE

```
(define (improve text)
    (choose
        (improve (c-imp text))
        text)))
```

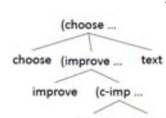
Call by *Poaps Value* Semantics

Compile into HAM

Step 1: Define a set of states *S(p)*

```
(define (improve text)
    (choose
        (improve (c-imp text))
        text)))
```

A state variable for every argument

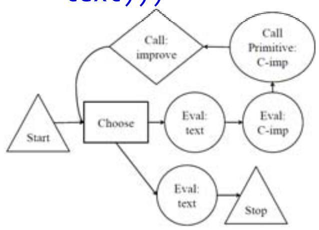


A state variable for every sub-expression

Not Shown: State variables for called functions (Recursive Definition)

Step 2: Construct a HAM

```
(define (improve text)
    (choose
        (improve (c-imp text))
        text)))
```



Step 3: Merge

- Final State Space: *S(p)* + States of HAM
- Actions – Given by HAM
- Transitions – Ensure "Call by *Poaps value* semantics"
- Observations – given by primitives
- Costs – given by primitives

Conclusion

- How bring RL to the masses?
- Compose dynamical systems?
- Specify & compose utility functions?
- RL algorithms?

Cascade

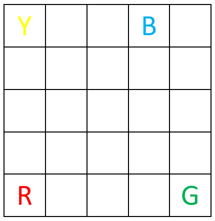
```
def iterativeCascade(tips):
   catsAndMembers = []
   tipsToRun = [tips]
   while( len(tipsToRun) > 20):
      tipSubset = createTipSubset(tipsToRun)
      catsAndMembersForSubset = cascade(tipSubset)
      catsAndMembersForSubset = pruneTaxonomy(catsAndMembersForSubset)
      catsAndMembers.add(catsAndMembersForSubset)
      newCategories = catsAndMembersForSubset.getCategories()
      catsAndMembersForNewCatsAndOldMembers = categorize(newCategories, tipsAlreadyRun)
      catsAndMembers.add(catsAndMembersForNewCatsAndOldMembers)
      catsAndMembers = pruneTaxonomy(catsAndMembers)
      tipsToRun.remove(tipSubset)
      tipsAlreadyRun.add(tipSubset)
   return generateTaxonomy(catsAndMembers)

def cascade(tips):
   suggestedCateogries= generateCategories(tips)
   bestSuggestedCategories = getBestSuggestedCategories(tips, suggestedCategories)
   allTipsWithAllCategories = createDictionaryTipsToCategories(tips, bestSuggestedCategories, k = 7)
   catsAndMembersFirstPass = categorize(allTipsWithAllCategories)
   catsAndMembersSecondPass = categorize(catsAndMembersFirstPass)
   return catsAndMembersSecondPass
```

Existing Tools Inadequate

- No support for
  - Workflow planning & optimization
  - Worker modeling & parameter learning
  - Adaptive workflow execution
- Managing Turkers is like … herding cats…

Clowder

ALisp




```
(if (not (have-pass))
    (get)) (put))

(define (get)
  (choose
    (action 'load)
    (navigate (pickup))))
```

MDP + Partial Program

PPDDL

```
(define (domain bomb-and-toilet)
        (:requirements :conditional-effects :probabilistic-effects)
        (:predicates (bomb-in-package ?pkg) (toilet-clogged)
        (bomb-defused))
        (:action dunk-package
                :parameters (?pkg)
                :effect (and (when (bomb-in-package ?pkg) (bomb-
                                defused))
                                (probabilistic 0.05 (toilet-clogged)))))

(define (problem bomb-and-toilet)
        (:domain bomb-and-toilet)
        (:requirements :negative-preconditions)
        (:objects package1 package2)
        (:init (probabilistic 0.5 (bomb-in-package package1)
                0.5 (bomb-in-package package2)))
        (:goal (and (bomb-defused) (not (toilet-clogged)))))
```

Adaptive Programs

State

```
for (i = 1; i < N; i++) {
  c = randomContext();
  m = move.suggest(c);
  reward(payoff(c,m));
}
```

POAPS Primitives

A *primitive* is a ten-tuple $\langle \mathcal{D}, \mathcal{R}, \Omega, \mathcal{T}, \mathcal{O}, \mathcal{I}, \mathcal{C}, \mathcal{D}_U, \mathcal{R}_U, \mathcal{F} \rangle$, where:

- $\mathcal{D} = \mathcal{D}^1 \times \ldots \times \mathcal{D}^n$ is a set of *domain states.*
- $\mathcal{R}$ is a set of *range states.*
- Ω is a set of *observations.*
- $\mathcal{T} : \mathcal{D} \times \mathcal{R} \to [0, 1]$ is a *transition function.*
- $\mathcal{O} : \mathcal{R} \times \Omega \to [0, 1]$ is an *observation function.*
- $\mathcal{I}$ is an n-dimensional indicator vector indicating which of the $\mathcal{D}^i$ are observable.
- $\mathcal{C} : \mathcal{D} \to \mathbb{R}^+$ is a *cost function.*
- $\mathcal{D}_U = \mathcal{D}_U^1 \times \ldots \times \mathcal{D}_U^n$ is a set of *user domain states.*
- $\mathcal{R}_U$ is a set of *user range states.*
- $\mathcal{F} : \mathcal{D}_U \to \mathcal{R}_U$ is a *user function.*

Model of the function

Some Function

Primitive: *c-imp*

- $\mathcal{F}(\alpha \in \mathcal{D}_U) = \text{calltoAPI}(\alpha)$
- $\mathcal{D}_U, \mathcal{R}_U$ is the set of all artifacts α.

- $\mathcal{D} = \mathcal{R} = [0, 1]$
- $\mathcal{T}(q \in \mathcal{D}, q' \in \mathcal{R}) = P(q'|q)$
- $\mathcal{C} = \$0.05$
- No observations/observation function
- $\mathcal{I} = (0)$

POAPS LANGUAGE

```
(define (improve text)
    (choose
        (improve (c-imp text))
        text)))
```

Call by *Poaps Value* Semantics

Compilation

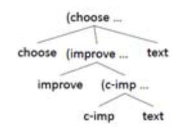
- Input: program *p*

```
(define (improve text)
    (choose
        (improve (c-imp text))
        text)))
```

Step 1: Define a set of states *S(p)*

```
(define (improve text)
    (choose
        (improve (c-imp text))
        text)))
```

A state variable for every argument

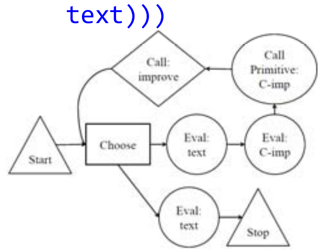


A state variable for every sub-expression

Not Shown: State variables for called functions (Recursive Definition)

Step 2: Construct a HAM

```
(define (improve text)
    (choose
        (improve (c-imp text))
        text)))
```



Step 3: Merge

- Final State Space: *S(p)* + States of HAM
- Actions – Given by HAM
- Transitions – Ensure "Call by *Poaps value* semantics"
- Observations – given by primitives
- Costs – given by primitives

```
(define (vote q a0 a1 c0 c1)
  (choose
    (if (ask-crowd q a0 a1)
      (vote q a0 a1 (+ c0 1) c1)
      (vote q a0 a1 (+ c1 1))
    (if (> c0 c1)
        True
        False)))
```

```
(define (it-i image worse-text better-text)
     (choose
          (it-i image better-text
                    (c-imp better-text))
          (if (vote image better-text
               worse-text 0 0)
              (it-i image worse-text better-
                                   text)
              (it-i image better-text worse-
                                    text))
          better-text))
```