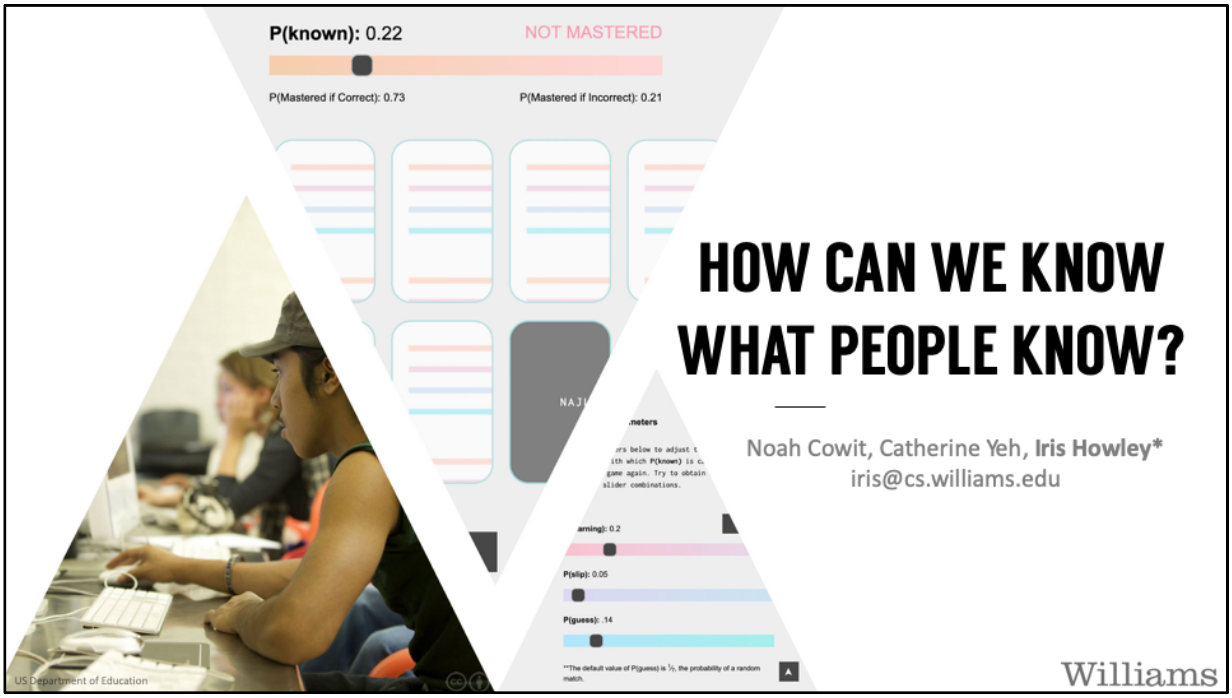


Hi everyone, my name is Iris Howley and I'm an assistant professor of computer science at Williams College (we're hiring!). I'm going to talk today about work I did with my undergraduate Ras, Noah and Catherine, who couldn't be here today as students apparently cannot miss a week of class, but professors can (the irony!). Our work happens at the overlap between HCI and the learning sciences, which means we spend our time thinking deeply about how to improve student and teacher learning with technology. Today I'm going to talk specifically, about how can we know what people know?

ALGORITHMS TO PREDICT IF STUDENTS KNOW A SKILL

- *Traditional*: Get above x percentage on the assignment
- *Khan Academy / WPI's Assistments*: Get n problems in a row correct
- *McGraw-Hill's ALEKS system*: Knowledge Space Theory
- *CMU's OLI / Stanford's OARS*: **Bayesian Knowledge Tracing (BKT)**

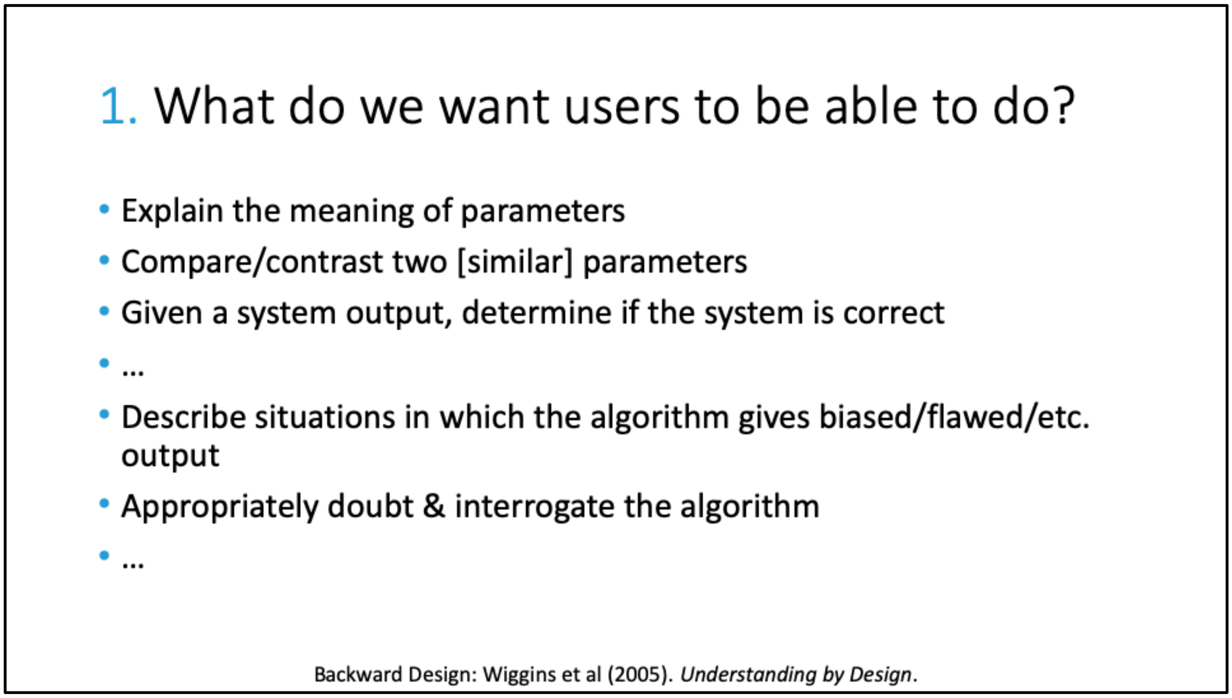
BKT: Corbett & Anderson (1994). Knowledge tracing: Modeling the acquisition of procedural knowledge.

Everyone already has some familiarity with this. <CLICK> such as, if you get a 95% on the exam, you've mastered the skill. Technology has introduced a few additional models such as <CLICK> get 4 in a row correct and you have mastery and <CLICK> the K-12 ALEKS system that has knowledge space theory. <CLICK> but today I'll be talking about Bayesian Knowledge Tracing , or BKT.

BKT is not particularly complex (3 formulas, ~4 parameters), [we built an explainable of it!], but it's complex enough that the users of these systems generally can't explain the algorithm, and if the students or instructors were to look up the research papers, it wouldn't necessarily help them develop any intuition for the algorithm.

How will algorithmic understanding impact user behavior with algorithmic systems?

This brings me to the topic I’m most excited about. Thinking about the end users of algorithmic systems and how better understanding of their algorithm can improve or change their use of the system. How might we do this?



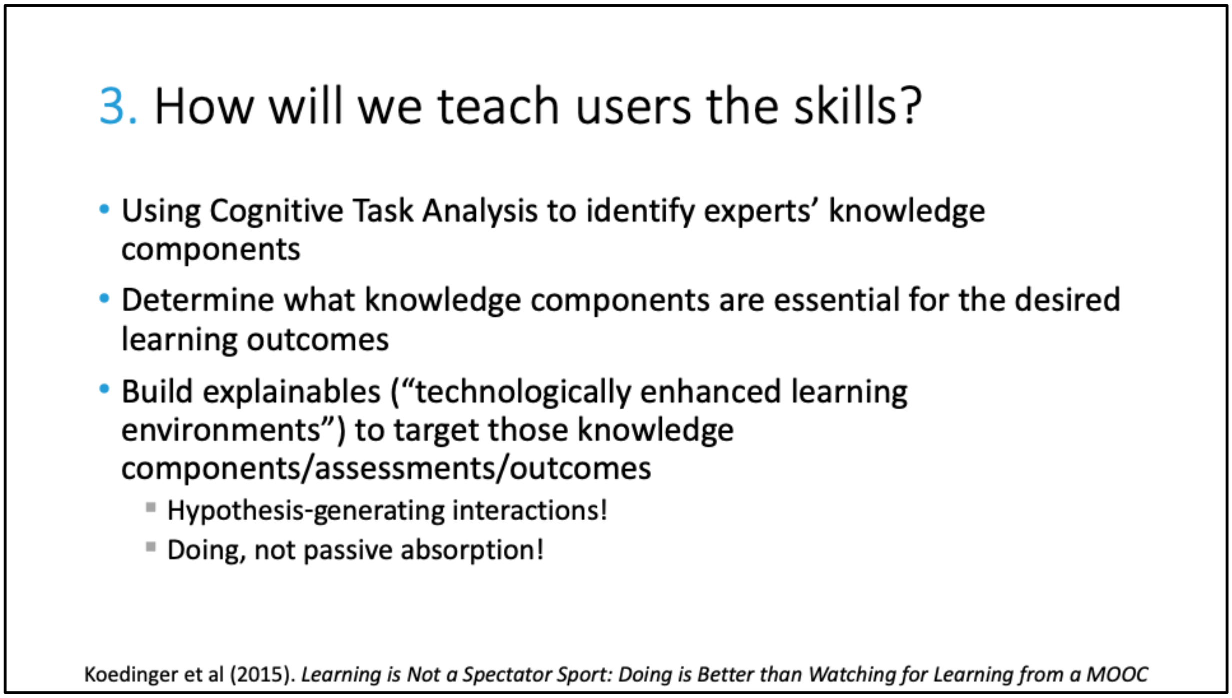
We use something from education research called backward design, although there's some similar parallels to the design of classic info viz tasks here. Basically, we start with what we want learners to be able to do. Maybe that's basic comprehension like defining the parameters…but then we build up to more complex concepts like verifying the output of the system…or even more complex concepts like identifying the edge cases for the algorithm, or doubting and interrogating the algorithm at an appropriate time (we don't want complete distrust, because then you might as well not have the system at all!).

2. How will we assess if users can do that?

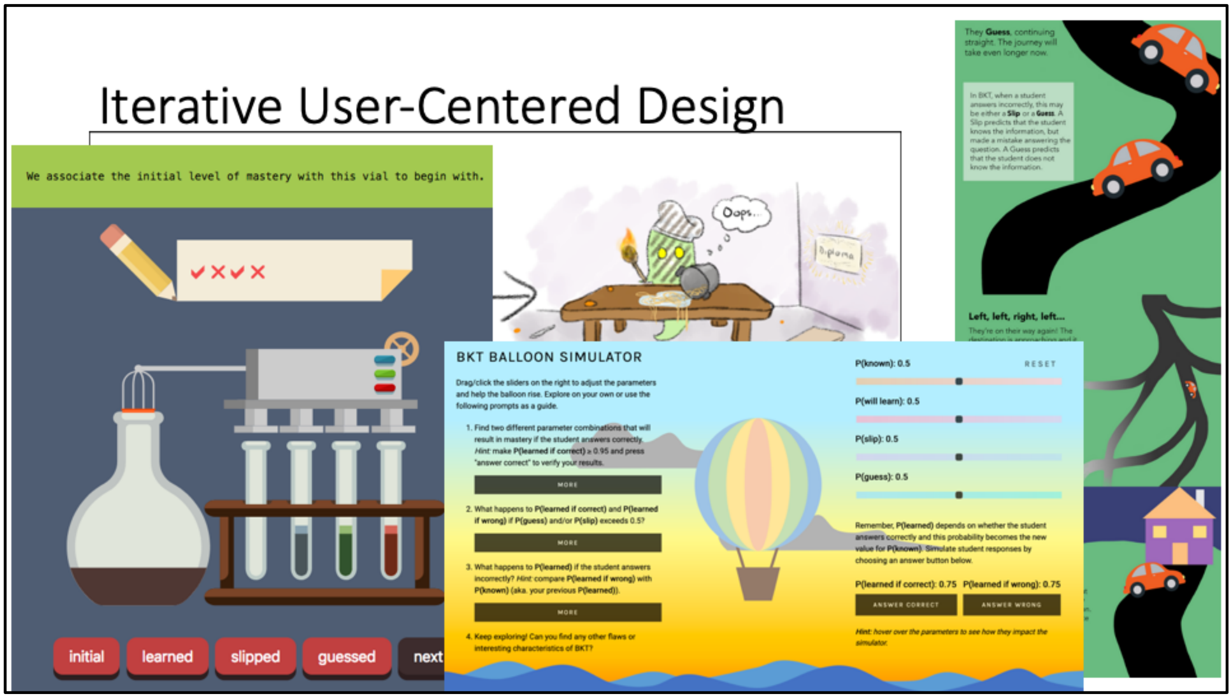
- Pretest/Posttest items
  - Hypothetical situations
- Interviews
- Self-report
- Behavioral outcome measures

Backward Design: Wiggins et al (2005). *Understanding by Design*.

Then you need to determine how to assess if users can do those things. This gets stickier. Pre/posttest is pretty classic, interviews, self-report of trust or fairness, and then behavioral outcome measures. We've been working with hypothetical situations or vignette surveys to help evaluate in the lab, but going forward we're still designing ways to look at this in the field.



Then we start thinking about how to teach the skills we're assessing. We're using CTA to help us define the skills and then targeting the development of explainables at components of those skills to determine which are essential for our desired outcomes. I do want to point out (as was mentioned in a earlier talk in this session) that we really need to focus on hypothesis-generating interactions, not passive absorption. Scrolling through a website will not create as robust learning as an interaction where the user pulls knowledge to make a prediction. So the more of that we can do, the better.

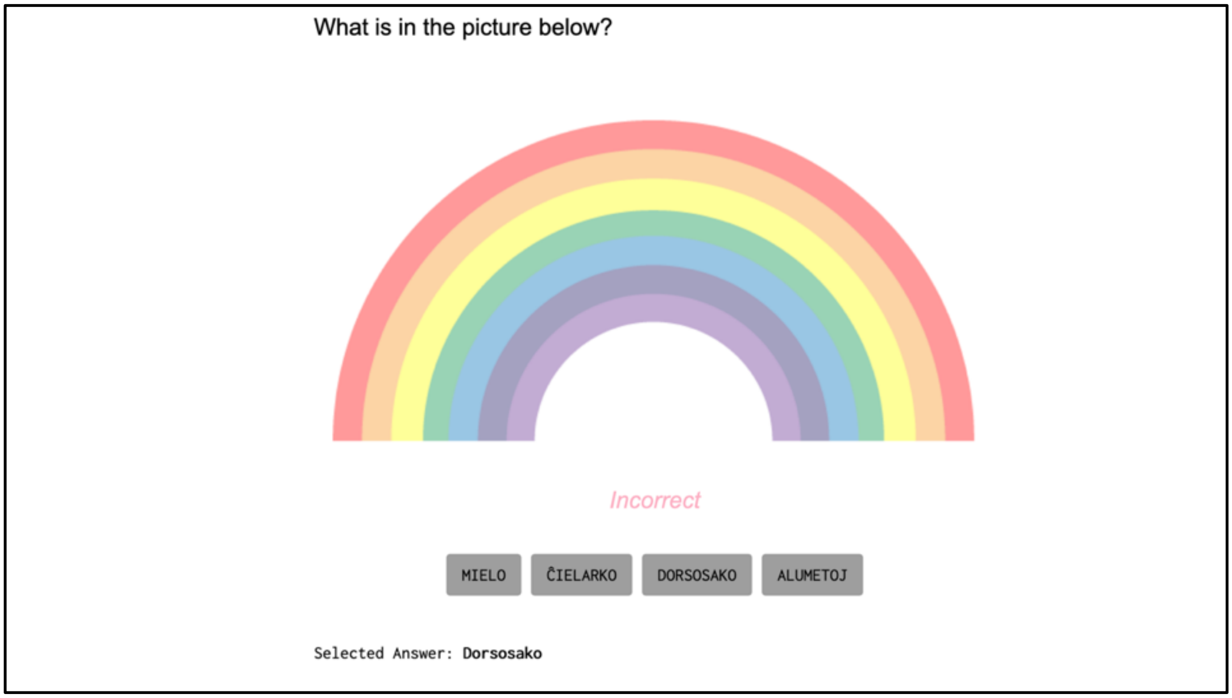


We also need to add in a little iterative user-centered design. <CLICK> <CLICK> <CLICK> <CLICK>. We're on our 5th or 6th design, but it's necessary to go through this process to determine how to maximize learning while minimizing usability issues. There's a really lovely discussion of "desirable difficulties" in the learning science literature that I won't go into now, but it's an interesting conversation that can run in opposition to some HCI conversations.

Do you want to learn about BKT while also learning a little Esperanto?

bit.ly/visxaibkt

And that brings us to our current explainable. We teach you a teensy bit of Esperanto, I turned it into a bitly link but you can also access it from the workshop website. I'm going to do a little introduction of it, but the explainable will do a much better job of explaining BKT than I can in the remaining time.



We start basic, explaining what the probability of guessing a question correctly is. Not all our users are statisticians and many are students, so considering our end users' prior knowledge here is very necessary.

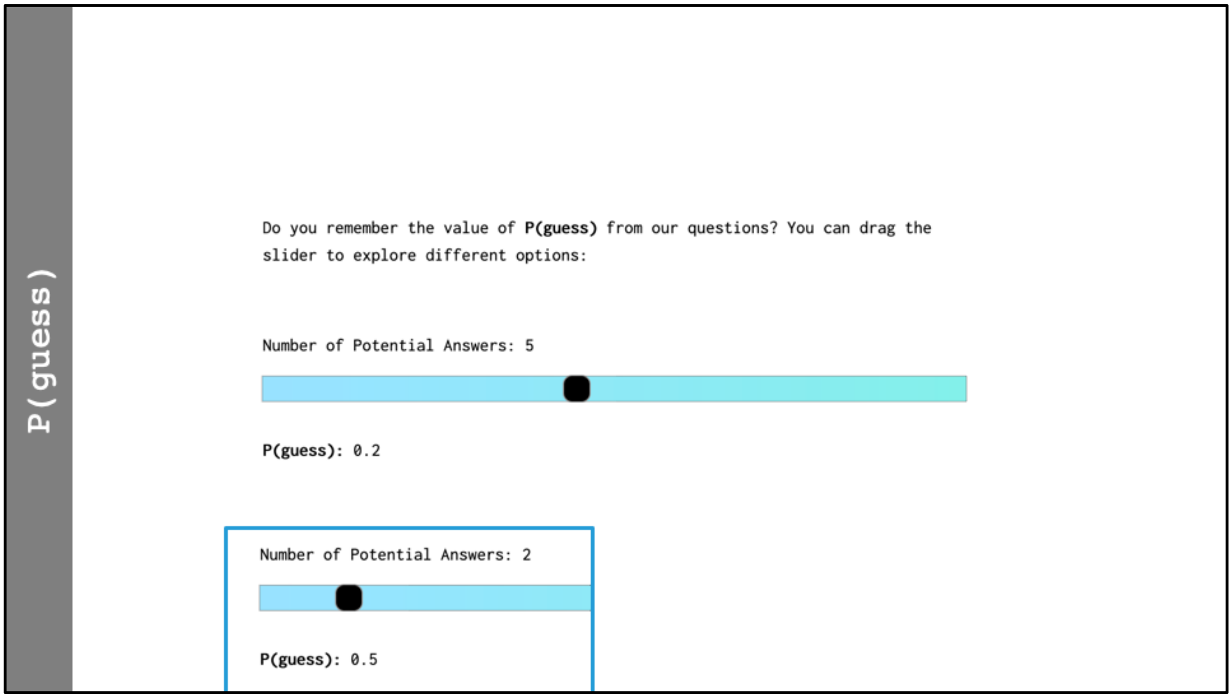
What is in the picture below?

As there are 4 possible answers, we have a ¼ chance of a correct guess, or a probability of 0.25.

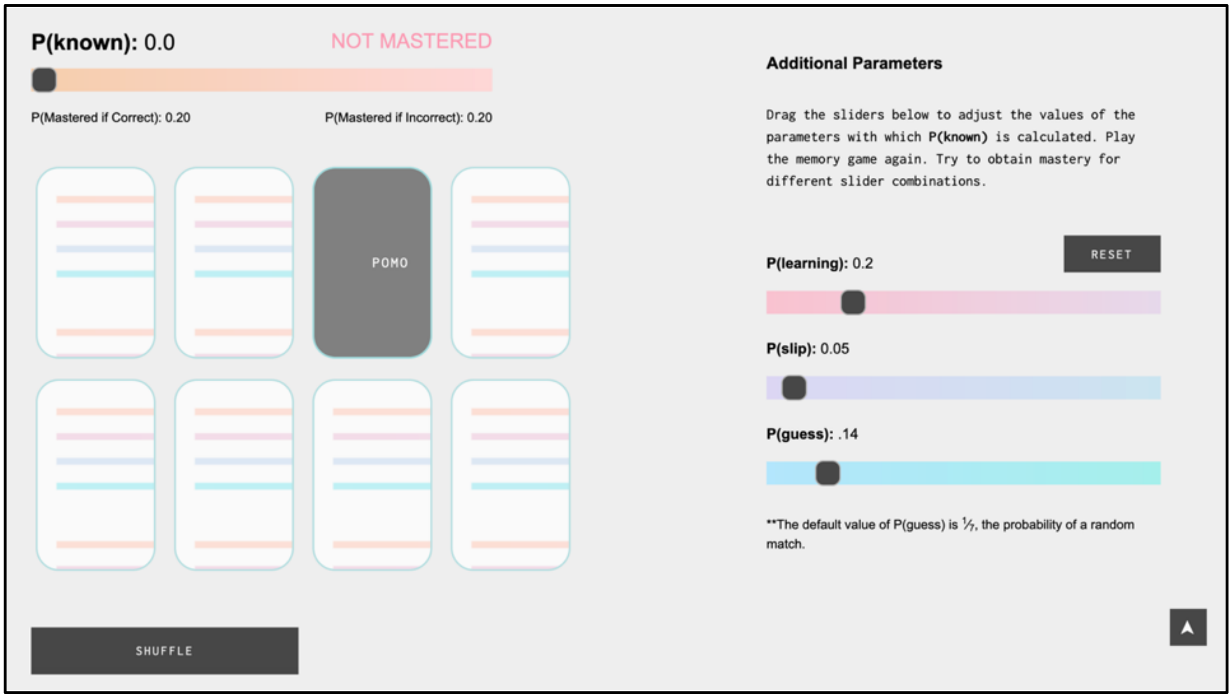
Correct

MIELO | ĈIELARKO | DORSOSAKO | ALUMETOJ

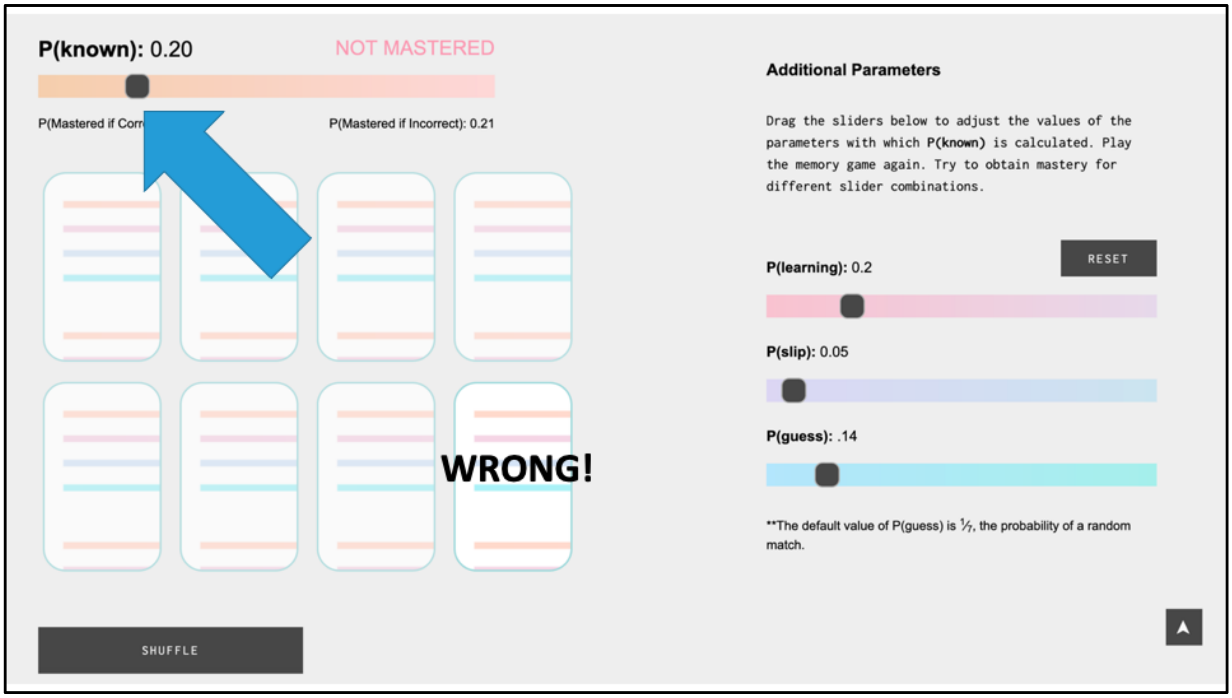
Selected Answer: **Ĉielarko**



We also provide some sliders to illustrate that as the number of possibly answers goes up, the probability of guessing correctly goes down. We have other interactions like this for the remaining 3 parameters, and these all build up to …<CLICK>



…the final activity in which you play a game of memory which illustrates the system's prediction of your mastery. This is a little stretch for BKT, but illustrates the concepts. And it also lets you modify the parameters on the side to see how that impacts game play.



What this explainable does particularly well is it illustrates that even when you answer incorrectly, the prediction of mastery still goes up a little bit. So being wrong, but getting feedback on that produces some small chance that you've mastered the concept. In the game of memory, it means you've learned a location for another word and so this makes sense here. But in our user studies, participants are often quite surprised by this and this explainable illustrates this idea particularly well.

How will algorithmic understanding impact user behavior with algorithmic systems?

Noah Cowit, Catherine Yeh, **Iris Howley***
*** iris@cs.williams.edu**

THANK YOU!

Funding for this research was provided by the IIS:Cyber-Human Systems program: NSF CISE award number 1849984.

And with that, I'd like to leave you with encouraging everyone to consider who their users are and what they'd like them to be able to do, and what changes you expect that to have on not only their workflow, but their attitudes and future behavior with the system as well. Thank you!