Comments on MacLellan, Harpstead, Alevan, and Koedinger

- Multiple attribute and relation types (nominal, numeric, component, relational)

  - Can components include components as well as base attributes?

- The incremental (iterative) algorithm (again)

- Predicting missing attributes (again)

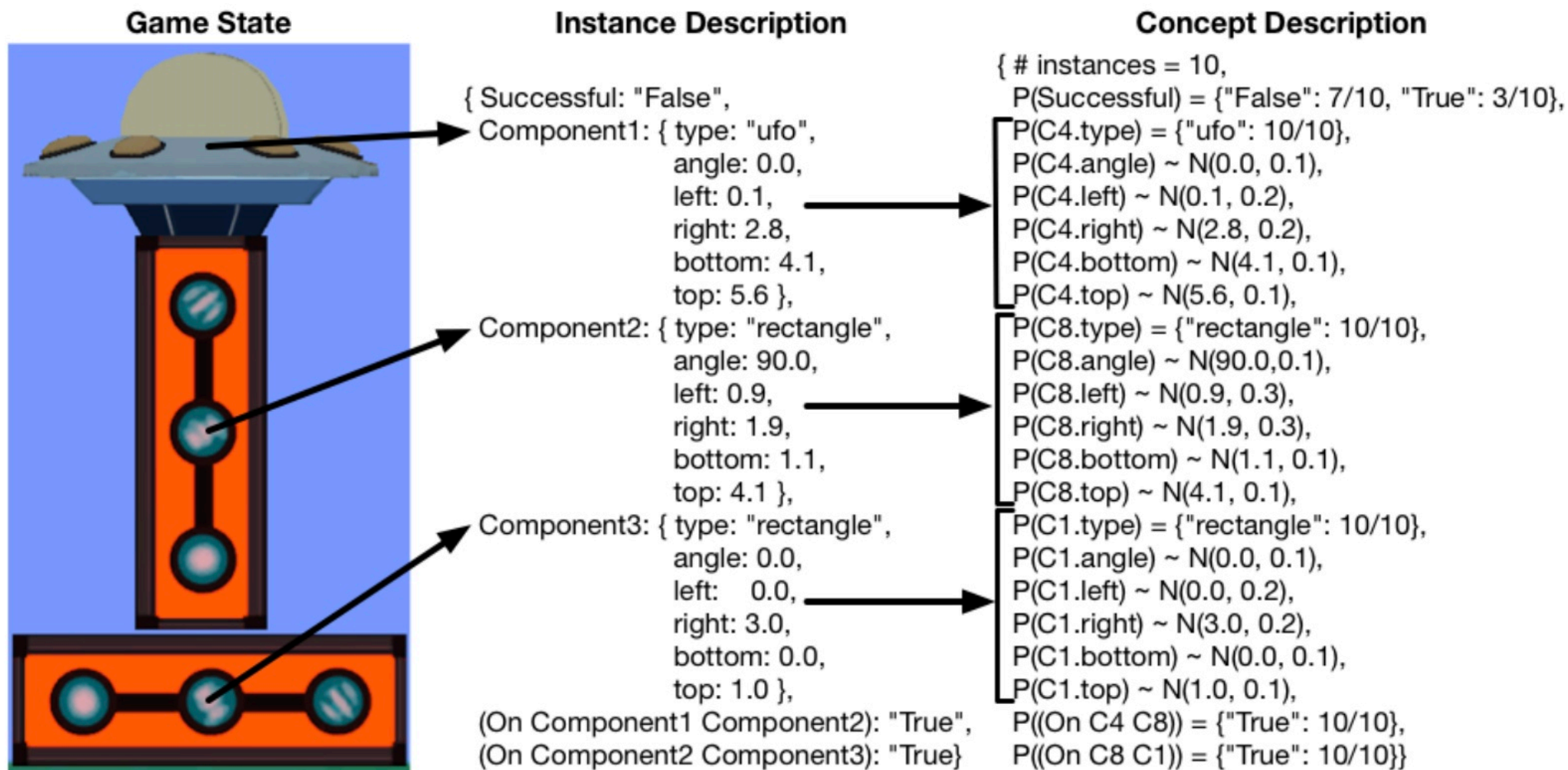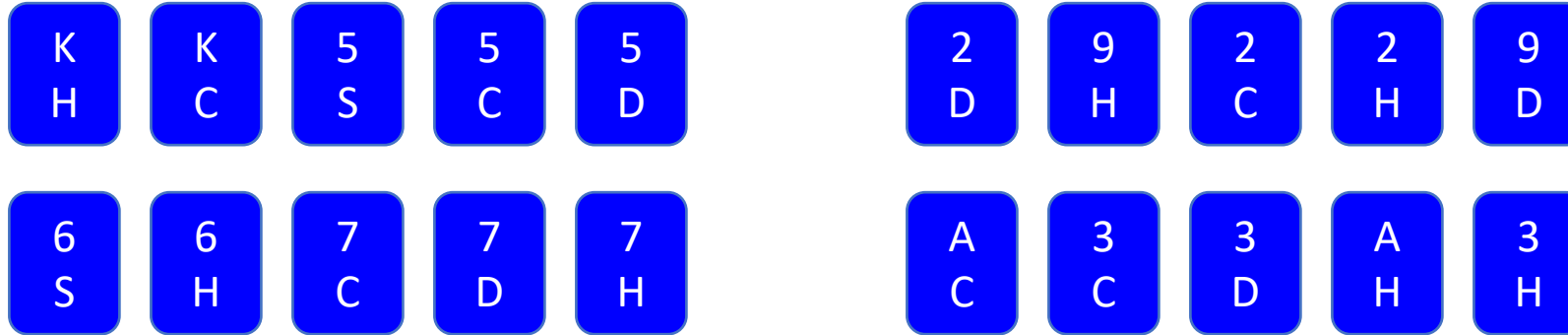- The matching problem (next slides)

- Flattening

Game State

Instance Description

{ Successful: "False",
 Component1: { type: "ufo",
                angle: 0.0,
                left: 0.1,
                right: 2.8,
                bottom: 4.1,
                top: 5.6 },
 Component2: { type: "rectangle",
                angle: 90.0,
                left: 0.9,
                right: 1.9,
                bottom: 1.1,
                top: 4.1 },
 Component3: { type: "rectangle",
                angle: 0.0,
                left: 0.0,
                right: 3.0,
                bottom: 0.0,
                top: 1.0 },
 (On Component1 Component2): "True",
 (On Component2 Component3): "True}

Concept Description

{ # instances = 10,
  P(Successful) = {"False": 7/10, "True": 3/10},
  P(C4.type) = {"ufo": 10/10},
  P(C4.angle) ~ N(0.0, 0.1),
  P(C4.left) ~ N(0.1, 0.2),
  P(C4.right) ~ N(2.8, 0.2),
  P(C4.bottom) ~ N(4.1, 0.1),
  P(C4.top) ~ N(5.6, 0.1),
  P(C8.type) = {"rectangle": 10/10},
  P(C8.angle) ~ N(90.0, 0.1),
  P(C8.left) ~ N(0.9, 0.3),
  P(C8.right) ~ N(1.9, 0.3),
  P(C8.bottom) ~ N(1.1, 0.1),
  P(C8.top) ~ N(4.1, 0.1),
  P(C1.type) = {"rectangle": 10/10},
  P(C1.angle) ~ N(0.0, 0.1),
  P(C1.left) ~ N(0.0, 0.2),
  P(C1.right) ~ N(3.0, 0.2),
  P(C1.bottom) ~ N(0.0, 0.1),
  P(C1.top) ~ N(1.0, 0.1),
  P((On C4 C8)) = {"True": 10/10},
  P((On C8 C1)) = {"True": 10/10}}

Figure 2. A tower in *RumbleBlocks*, its representation as an instance in TRESTLE using the four attribute-

Relational (e.g., first-order) representations, such as:

IF R(?c1, ?r1) ⋀ R(?c2, ?r1) ⋀ R(?c3, ?r2) ⋀ R(?c4, ?r2) ⋀ R(?c5, ?r2)
  ⋀ ≠(?c1, ?c2) ⋀ ≠(?c3, ?c4) ⋀ ≠(?c3, ?c5) ⋀ ≠(?c4, ?c5)

THEN FullHouse(?c1, ?c2, ?c3, ?c4, ?c5)

The matching problem (on sets of feature vectors)

Back to Trestle

- Is Trestle limited to a one-to-one matching of components?

- How would Trestle handle towers of different heights (i.e., a different number of composite blocks)?

  - Are one or more components (e.g., blocks, cards) left unmatched?

  - Are one or more components double (multiply) matched (i.e., a many to one matching)

- How does Trestle's matching strategy compare to human matching strategies?

Fit to human behavior

- Compared Trestle end-results to human end-results

  - No comparisons on HOW these end-results results were obtained

    - e.g., no comparison of Trestle matching with human matching

  - Supervised: Just end-result accuracies

  - Supervised: No response time predictions

  - Unsupervised: only (two →) one human clustering for comparison

    - A lab colleague at that

- In addition to comparing human and Trestle behavior without functional knowledge,

  - Could functional knowledge be encoded into Trestle too?

  - Trestle would have to be extended to use background knowledge

Interesting observations

- Categorization during problem solving (p. 131)

- Can ordering effects be exploited – "how best to order practice problems" (p. 132)

- Supervised and unsupervised together (p. 132 …)

- Aside: CFE converts parse trees to feature vectors (p. 134). Why not cluster parse trees directly?

  - Fisher and Yoo: "Categorization, Concept Learning, and Problem Solving: A Unifying View"

Difficulties in understanding

- I (Doug) doesn't understand how CART was used (pp. 140-141)

    - and why it went unmentioned in discussion of the supervised learning results

    - Why was CART needed at all since CFE can do prediction (p. 134)?

- CFE insufficiently described