A Theory of Adaptive Scaffolding for LLM-Based Pedagogical Agents

Clayton Cohn, Surya Rayala, Namrata Srivastava, Joyce Horn Fonteles, Shruti Jain, Xinying Luo, Divya Mereddy, Naveeduddin Mohammed, Gautam Biswas

Institute for Software Integrated Systems
Vanderbilt University
Nashville, TN 37212 USA
clayton.a.cohn@vanderbilt.edu

A Technical Appendix

This document provides technical details and additional information to support both reproducibility and clarification, which could not be included in the main manuscript due to space constraints. It also includes and discusses the curriculum, formative assessments, rubrics, and knowledge graphs referenced in the manuscript.

A.1 Code and Data Appendix Information

We provide all Jupyter Notebooks (accessed via Google Colab) used for preprocessing, analysis, and evaluation in our Code and Data Appendix (CDA). Any code involving non-determinism (e.g., data splits and OpenAI API calls) was conducted using random seeds to ensure reproducibility, which can be found in the Jupyter Notebooks. The CDA is organized as follows:

- The *rq1_grading_eval* folder contains all materials used to address RQ1, including all scripts and prompts. The *scripts* subfolder includes the code used to calculate descriptive statistics, assess inter-rater reliability (IRR), split the data, generate LLM outputs, and evaluate the four prompting levels (I/O, ICL, CoT, and AL) for each formative assessment. The *prompts* subfolder contains the specific prompts used in each of the four LLM grading runs for every formative assessment.
- The *rq2_faithfulness_eval* folder contains all materials used to address RQ2, including scripts and prompts for all LLM-as-a-Judge evaluations. The root folder includes scripts used to compute descriptive statistics, split the data, and generate the case study figure. The six subfolders correspond to the theoretical and teacher constructs discussed in the main manuscript. Each construct's folder contains *scripts* and *prompts* for the LLM-as-a-Judge runs, as well as the script used for IRR validation. The ZPD *scripts* folder additionally contains code for generating each assessment's knowledge graph (as discussed in Section A.5), and the readability folder includes a script for calculating aggregated readability statistics across all assessments. Within each construct's *prompts* folder, we provide the prompts used in the LLM-as-a-Judge faithfulness evaluation (RQ2) for each formative assessment. Self-efficacy did not require formative assessment-specific prompts for evaluation, and IRR and LLM-as-a-Judge were not used for readability, which was computed deterministically.
- The *inquizzitor* folder contains the three prompts used by our agent during student interactions (one per formative assessment). Each prompt includes information about the curriculum, formative assessments and rubrics, theoretical constructs, teacher constructs, and other general guidelines informed by prior studies and insights from our participatory design sessions with teachers conducted over the past five years.

A.2 SPICE Curriculum

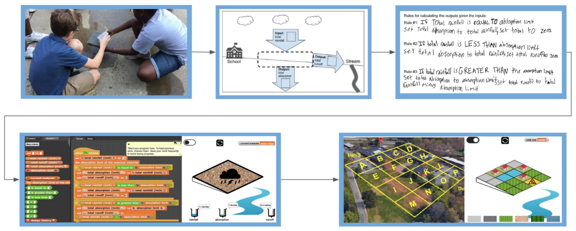


Figure 1: SPICE curriculum progression, including: physical experiments (top left), conceptual modeling (top middle), paper-based computational thinking (top right), computational modeling (bottom left), and engineering design (bottom right).

SPICE (Science Projects Integrating Computing and Engineering) is a three-week middle school curriculum unit targeting the science, computing, and engineering domains that challenges students to redesign their schoolyard using various surface materials to minimize water runoff after a rainstorm, while adhering to design constraints (such as including a parking lot, grassy field, and play area) and accounting for construction costs and accessibility needs. The problem-based learning curriculum consists of five core units, illustrated in Figure 1: (1) physical experiments, (2) conceptual modeling, (3) paper-based computational thinking, (4) computational modeling, and (5) engineering design. In the final unit, students use their computational models within the SPICE environment to redesign their schoolyard while addressing the specified constraints.

Formative assessments are interspersed throughout the

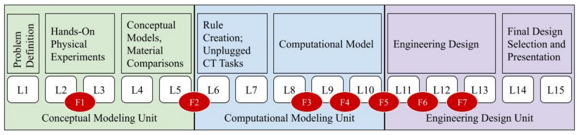

Problem Definition
Hands-On Physical Experiments
Conceptual Models, Material Comparisons
Rule Creation; Unplugged CT Tasks
Computational Model
Engineering Design
Final Design Selection and Presentation
L1
L2
L3
L4
L5
L6
L7
L8
L9
L10
L11
L12
L13
L14
L15
F1
F2
F3
F4
F5
F6
F7
Conceptual Modeling Unit
Computational Modeling Unit
Engineering Design Unit

Figure 2: SPICE curriculum (L items in white are lessons; F items in red are formative assessments.

units (illustrated in Figure 2) to help students self-evaluate their learning progress. These assessments, discussed in Section A.3, also enable teachers to monitor and support students' learning in science, computing, and engineering as they move through the curriculum. Students first build foundational science knowledge, then systematically advance to computational thinking (CT), computational modeling, and engineering problem-solving. This progression allows both teachers and students to assess how science and computing learning inform performance on engineering tasks. The associated grading rubrics, described in Section A.3, emphasize cross-domain connections, helping teachers track student progress and adjust instruction when challenges arise.

A.3 Assessments, Rubrics, Knowledge Graphs

We present all three formative assessments and rubrics used in our study (FAs 2, 3, and 4) and analysis below. In the rubrics, each point awarded to students targets one of three domains: science (SCI), computing (COM), or engineering (ENG). All formative assessments and rubrics were collaboratively developed using evidence-centered design principles (ECD) through participatory design sessions over the past five years between the research team and two teachers who implement this curriculum in their classrooms.

Additionally, we include each assessment's corresponding *knowledge graph*, presented alongside the assessment and rubric. Each knowledge graph is hierarchical, outlining the progression of knowledge students need to acquire—from "no knowledge" to "mastery." These knowledge graphs were used to teach Inquizzitor to identify each student's Zone of Proximal Development (ZPD), as discussed in Section A.5.

FA2 We refer to FA2 as the "Rules Task" because students are required to structure their understanding of the relationships between rainfall, absorption, absorption limit, and water runoff using conditional statements (i.e., "rules"). The rubric is multi-label, and students can earn up to 9 points: 1 point for identifying each of the three conditions where rainfall is less than, equal to, or greater than water runoff; 1 point for correctly specifying the absorption value within each statement; and 1 point for correctly specifying the runoff value within each statement.

FA2 requires students to first recognize the three conditional statements that correspond to the rules: when rainfall is less than, equal to, or greater than the surface absorption limit. Once these conditions are identified, students must correctly set the absorption and runoff values within each condition. Full credit for a rule is awarded only when the condition is correctly identified and both the absorption and runoff values are appropriately specified. For example, to receive full credit for the "less than" condition (i.e., three points), a student must provide the following statement: "If rainfall is less than absorption limit, set absorption to rainfall, and set runoff to 0." To achieve mastery—i.e., to earn the maximum score of 9 points—students must correctly identify all three rules. This progression is illustrated in Figure 4. All nodes are shown in blue to highlight the strong emphasis on the science domain; however, it should be noted that the conditional statements also require knowledge of computing—specifically, an understanding of *conditional structures*.

FA3 We refer to FA3 as the "Debugging Task" because students are asked to debug a fictitious student's computational model by identifying the five errors within it. Like FA2, FA3's rubric is multi-label, and students can earn up to 5 points—1 point for correctly identifying each of the five errors in the model (as shown in Figure 5).

We present FA3's knowledge graph in Figure 6. FA3 builds on the scientific concepts from FA2 (blue) and introduces new computing concepts (green), such as initializing and updating variables. Students must first understand that variable initialization must occur before the conditional statements, followed by the correct implementation of each FA2 rule in the computational model. Orange nodes represent cross-domain concepts requiring an integration of science and computing knowledge. Purple nodes correspond to the five specific errors in the computational model (see Figure 5).

FA4 We refer to FA4 as the "Engineering (or Fair Test) Task" because students must integrate their science and computing knowledge with the engineering concept of fair tests to determine whether two designs can be compared. The rubric is multi-class, and students can earn a maximum score of 4 if they correctly conclude that the two designs cannot be compared due to differing rainfall (i.e., input) values, which render the test unfair. FA4 and its rubric are presented in Figure 7.

We present the FA4 knowledge graph in Figure 8. As with FA2 and FA3, blue nodes represent scientific concepts, while yellow nodes indicate knowledge integration across multiple domains. Red nodes correspond to engineering concepts such as input/output relationships and project specifications. To achieve mastery, students must understand both the concept of fair tests and the relevant design constraints. However, because this assessment specifically targets understanding of fair tests, students could earn a perfect score by recognizing that equal input values are necessary for a valid comparison. Responses that focused only on design constraints were awarded fewer points.

A.4 Active Learning (RQ1)

As discussed in the manuscript, our validation set results for RQ1 revealed several mis-scoring trends in LLM grading for FA4. To address these deficiencies, we refined the rubric and prompt, adding a new few-shot example to support in-context learning. We describe this procedure below.

To conduct each active learning run on the validation set, we first analyzed the results quantitatively to determine whether the LLM exhibited a tendency toward false positives (FPs; i.e., overscoring) or false negatives (FNs; i.e., underscoring), in order to identify specific error trends for each rubric item. We then examined the LLM's chains-of-thought to understand the underlying reasons for these trends.

During the first active learning run on the validation set, we identified several deficiencies in the LLM's scoring behavior. First, the LLM frequently confused the zero- and one-point rubric items, failing to award a point to students who correctly stated that the two tests could not be compared but did not sufficiently explain that the rainfall values differed. To address this, we clarified the one-point criteria in the rubric. Second, some three-point responses were incorrectly scored as full credit (i.e., four points) due to ambiguity in rubric interpretation. We resolved this by revising the four-point criteria to explicitly require that students use language synonymous with "unfair" (e.g., "biased" or "invalid"). Additionally, we added a guideline to the prompt stating that full credit should only be awarded if one of these fairness-related terms was explicitly included. Finally, we restructured the rubric to reflect a clear hierarchical progression, aligning it with how human raters naturally interpreted and applied the scoring criteria.

During the second round of active learning, we observed that the LLM continued to struggle with distinguishing between three- and four-point student responses. To address this, we expanded the list of fairness-related synonyms in the four-point criteria (e.g., "unequal," "bad," and "unreliable"). Similarly, we revised the three-point criteria to focus solely on the mention of rainfall differences, removing any references to fairness-related language to create a clear decision boundary between the two levels.

During our third round of active learning, we added an additional few-shot example illustrating how a student might invoke fairness-related language without recognizing that the differing rainfall inputs were the source of the unfairness. This distinction was important, as a response that mentioned fairness but failed to identify the rainfall discrepancy qualified for only one point, whereas a full-score (four-point) response required both elements. We also revised the two-point criteria to align with human scorer interpretations by broadening it to include any meaningful discussion of engineering constraints, even if no specific constraint was explicitly named—clarifying a nuance that was previously unclear to the LLM.

At this point, no discernible mis-scoring trends remained in the validation set, so the prompt was finalized and deployed for testing (see Code and Data Appendix).

A.5 ZPD (RQ2)

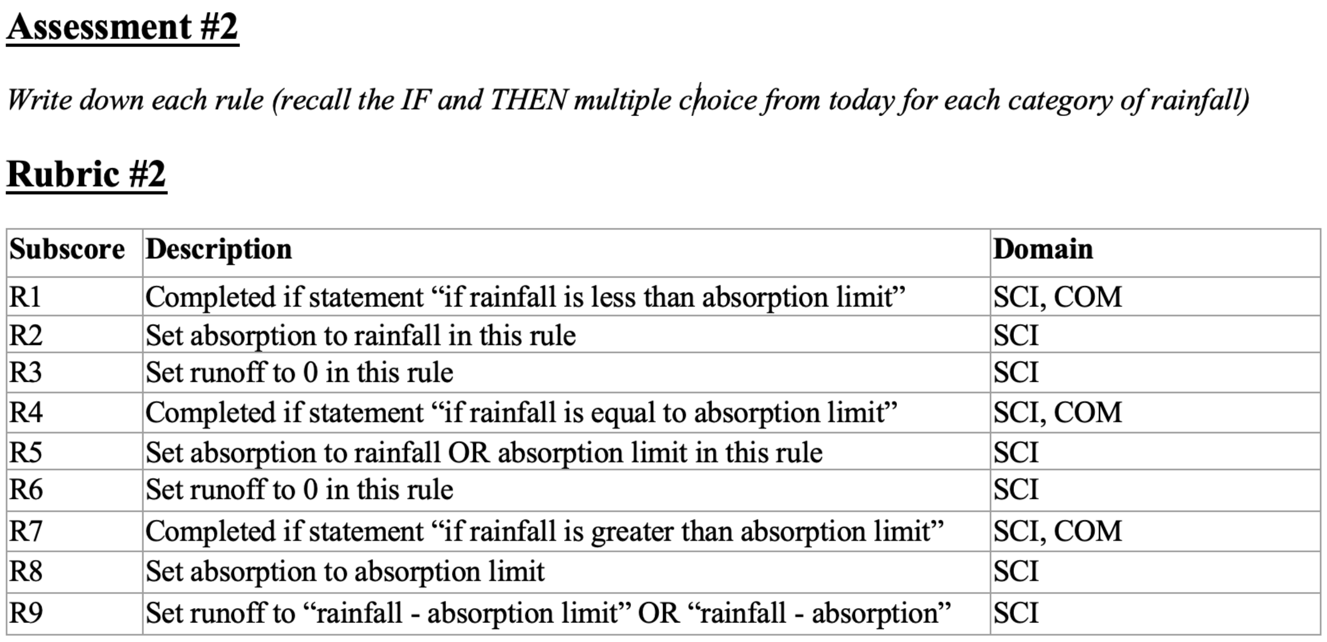
As noted in the manuscript, ZPD required a "specialized" LLM-as-a-Judge evaluation procedure, which we detail here. Our initial attempts to align the GPT-o3 reasoning judge with human consensus revealed significant human-LLM disagreement. While human scorers had a clear and shared understanding of what constituted a "ZPD hit" (i.e., the agent provided just enough information to allow the student to progress based on their assessment evidence, without removing the need for critical thinking to reach mastery), the subjectivity of the task proved too difficult to convey effectively through prompt engineering without representing the task using a formal structure.

To remedy this, we developed knowledge graphs for each assessment, which the LLM could use to identify the node that best represented a student's current knowledge state. While students could demonstrate knowledge corresponding to multiple nodes, we determined that a reasonable way to quantify a ZPD hit was to identify whether the agent response pushed the student one node forward (i.e., toward mastery) beyond what the student had already demonstrated. However, even with the structured knowledge graphs, the search space remained too large to reliably constrain the LLM's reasoning.

Our solution was to strike a balance: use the knowledge graphs to provide structure, but derive generalized decision trees from them to reduce the search space. For example, in the FA2 knowledge graph (see Figure 4), there are three nodes corresponding to the "less than," "equal to," and "greater than" conditional statements. Within each, students must correctly set both the absorption and runoff values. Rather than treating this subgraph as nine nodes with six edges, we collapsed the space by representing it as three nodes with two edges. In this case, we created a decision tree branch that defined a ZPD hit as follows: if any conditional statement is incomplete, the next step should be to set either the rainfall or runoff value within that condition to complete the "rule." Once we provided the decision trees (one per formative assessment) to each reasoning judge, the LLM's evaluations aligned closely with human consensus (κ_w = 93.15).

A.6 Reproducibility Statement

All data analyzed in this paper were collected from an in situ middle school classroom study conducted in 2025, as described in Section 3 of the main manuscript. Due to privacy considerations and IRB restrictions on data from minor participants, the formative assessment and student-agent interaction datasets cannot be made publicly available, despite being fully anonymized. Researchers or reviewers who wish to access these data for reproducibility may contact us; we will coordinate with our IRB to provide the anonymized datasets upon request in accordance with Vanderbilt University IRB guidelines.



Assessment #2

Write down each rule (recall the IF and THEN multiple choice from today for each category of rainfall)

Rubric #2

Subscore	**Description**	**Domain**
R1	Completed if statement “if rainfall is less than absorption limit”	SCI, COM
R2	Set absorption to rainfall in this rule	SCI
R3	Set runoff to 0 in this rule	SCI
R4	Completed if statement “if rainfall is equal to absorption limit”	SCI, COM
R5	Set absorption to rainfall OR absorption limit in this rule	SCI
R6	Set runoff to 0 in this rule	SCI
R7	Completed if statement “if rainfall is greater than absorption limit”	SCI, COM
R8	Set absorption to absorption limit	SCI
R9	Set runoff to “rainfall - absorption limit” OR “rainfall - absorption”	SCI

Figure 3: Formative assessment question and rubric for FA2.

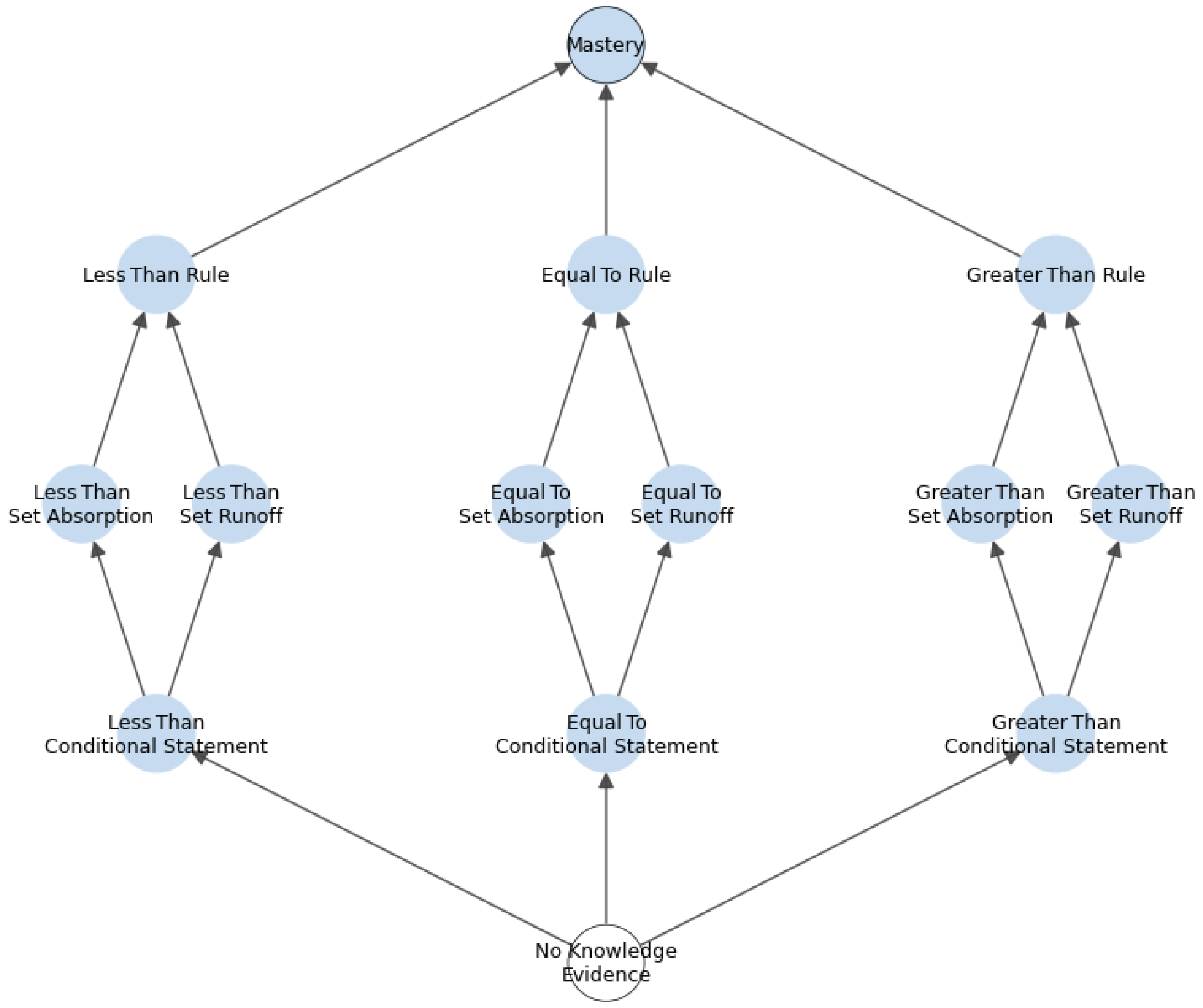
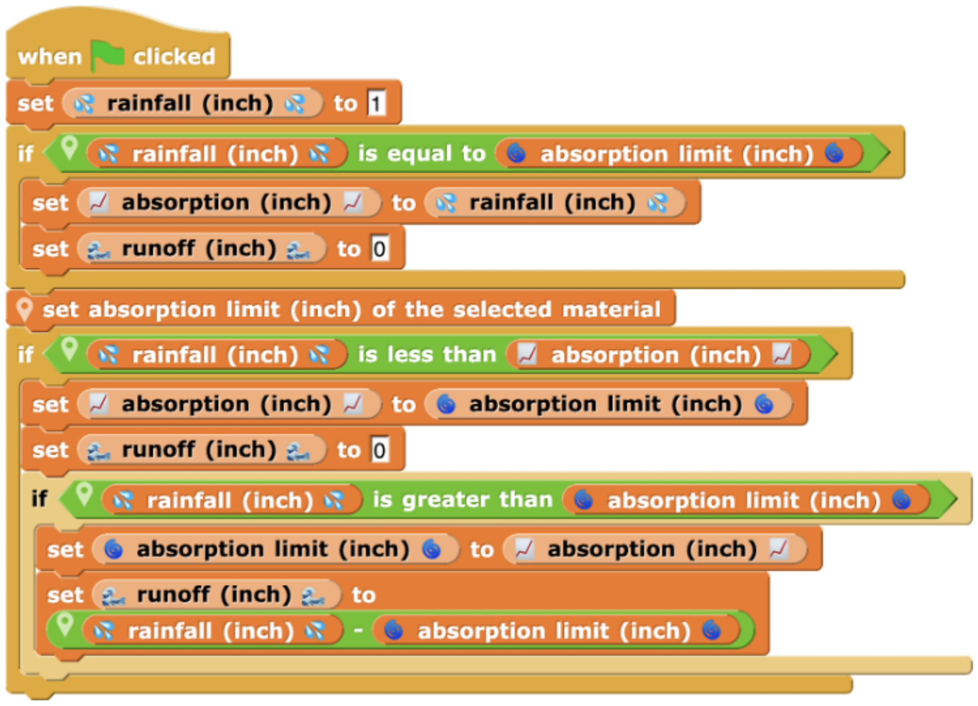


Figure 4: FA2 knowledge graph.

Assessment #3

Identify and describe 5 errors in this student's code.



```
when clicked
set rainfall (inch) to 1
if rainfall (inch) is equal to absorption limit (inch)
    set absorption (inch) to rainfall (inch)
    set runoff (inch) to 0
set absorption limit (inch) of the selected material
if rainfall (inch) is less than absorption (inch)
    set absorption (inch) to absorption limit (inch)
    set runoff (inch) to 0
    if rainfall (inch) is greater than absorption limit (inch)
        set absorption limit (inch) to absorption (inch)
        set runoff (inch) to
        rainfall (inch) - absorption limit (inch)
```

Rubric #3

Subscore	Description	Domain
D1	"Set absorption limit" should be before the first conditional statement.	COM
D2	In the less than condition, rainfall should be compared to the absorption limit.	SCI, COM
D3	In the less than condition, absorption should be set to rainfall.	SCI
D4	The greater than condition should not be embedded in the less than condition but connected to it.	COM
D5	In the greater than condition, you should set absorption to the absorption limit not the other way around.	SCI, COM

Figure 5: Formative assessment question and rubric for FA3, including the fictitious student's erroneous computational model.

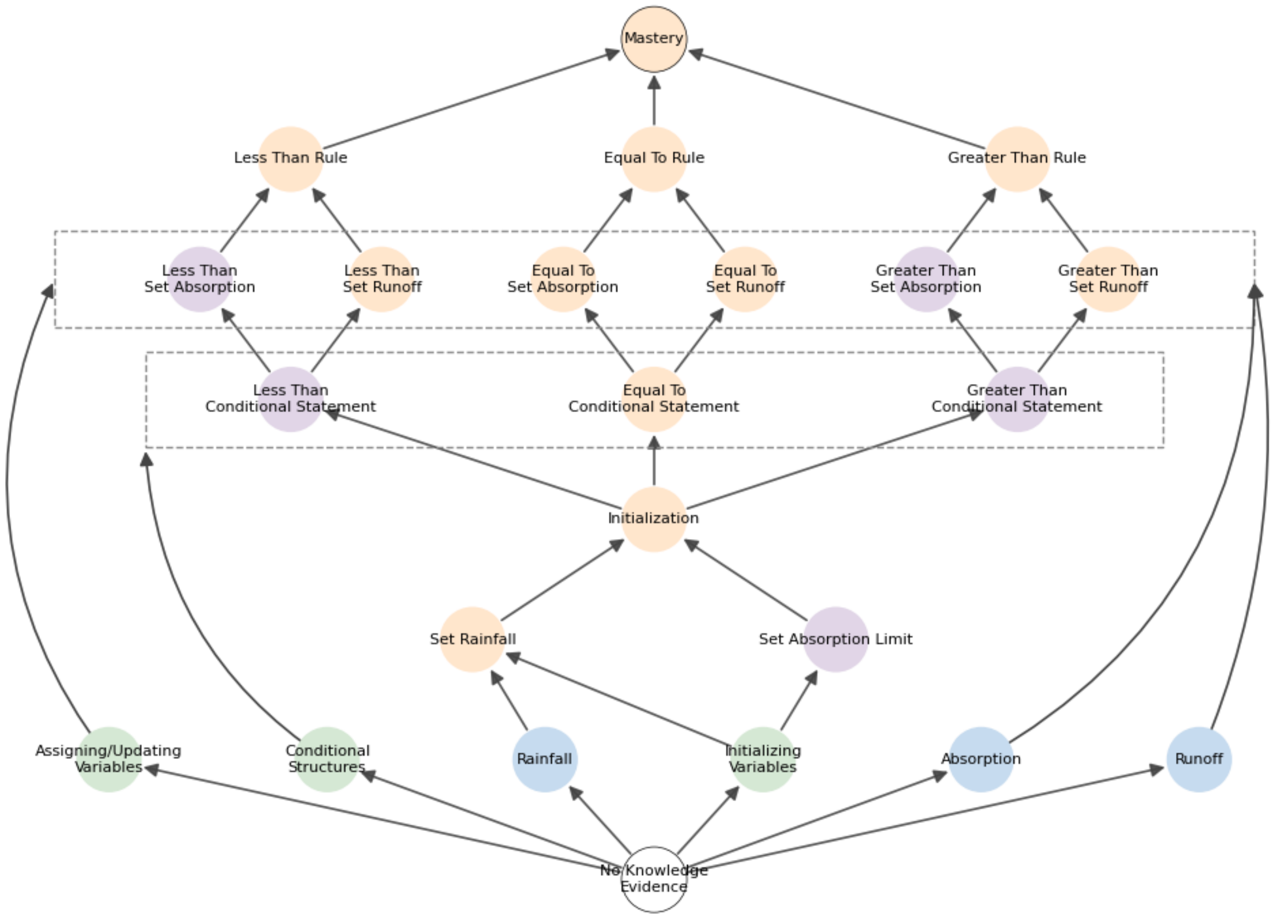
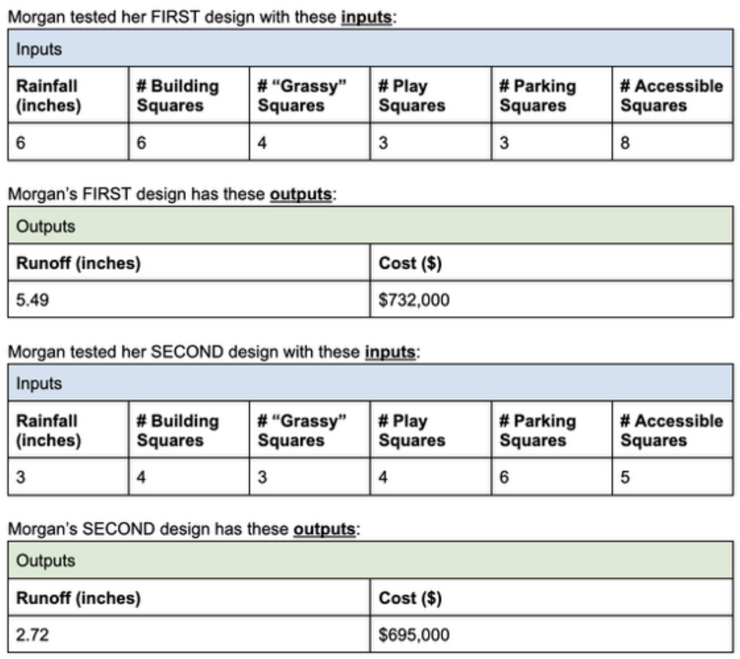


Figure 6: FA3 knowledge graph.

Assessment #4

Can Morgan tell which design is better based on these tests? Explain why or why not.


Morgan tested her FIRST design with these **inputs**:

Inputs					
Rainfall (inches)	**# Building Squares**	**# “Grassy” Squares**	**# Play Squares**	**# Parking Squares**	**# Accessible Squares**
6	6	4	3	3	8

Morgan’s FIRST design has these **outputs**:

Outputs	
Runoff (inches)	**Cost ($)**
5.49	$732,000

Morgan tested her SECOND design with these **inputs**:

Inputs					
Rainfall (inches)	**# Building Squares**	**# “Grassy” Squares**	**# Play Squares**	**# Parking Squares**	**# Accessible Squares**
3	4	3	4	6	5

Morgan’s SECOND design has these **outputs**:

Outputs	
Runoff (inches)	**Cost ($)**
2.72	$695,000

Rubric #4

Score	Description	Domain
4	Answer describes that the student cannot compare the designs because the student used different rainfall values to test each design AND student explains that the runoff comparisons will not be fair (or the comparison is therefore not a fair test).	ENG, SCI, COM
3	Answer describes that the student cannot compare the designs because the student used different rainfall values to test each design.	ENG, SCI
2	Answer describes that the student cannot compare the two designs because all contain issues meeting the engineering constraints (potentially including that runoff is too high, cost needs to be below $750,000, and there needs to be more than 6 accessible squares). This answer demonstrates an understanding of constraint satisfaction but does not identify the need to create fair tests.	ENG
1	Answer identifies that the student cannot compare the designs but does not provide reasoning or gives an ambiguous explanation.	ENG
0	Any other “error” identified. In this case, if the student answers “yes” on the first part they should get 0 here.	SCI, COM

Figure 7: Formative assessment question and rubric for FA4, including the inputs and outputs for the two tests students were asked to evaluate for fairness.

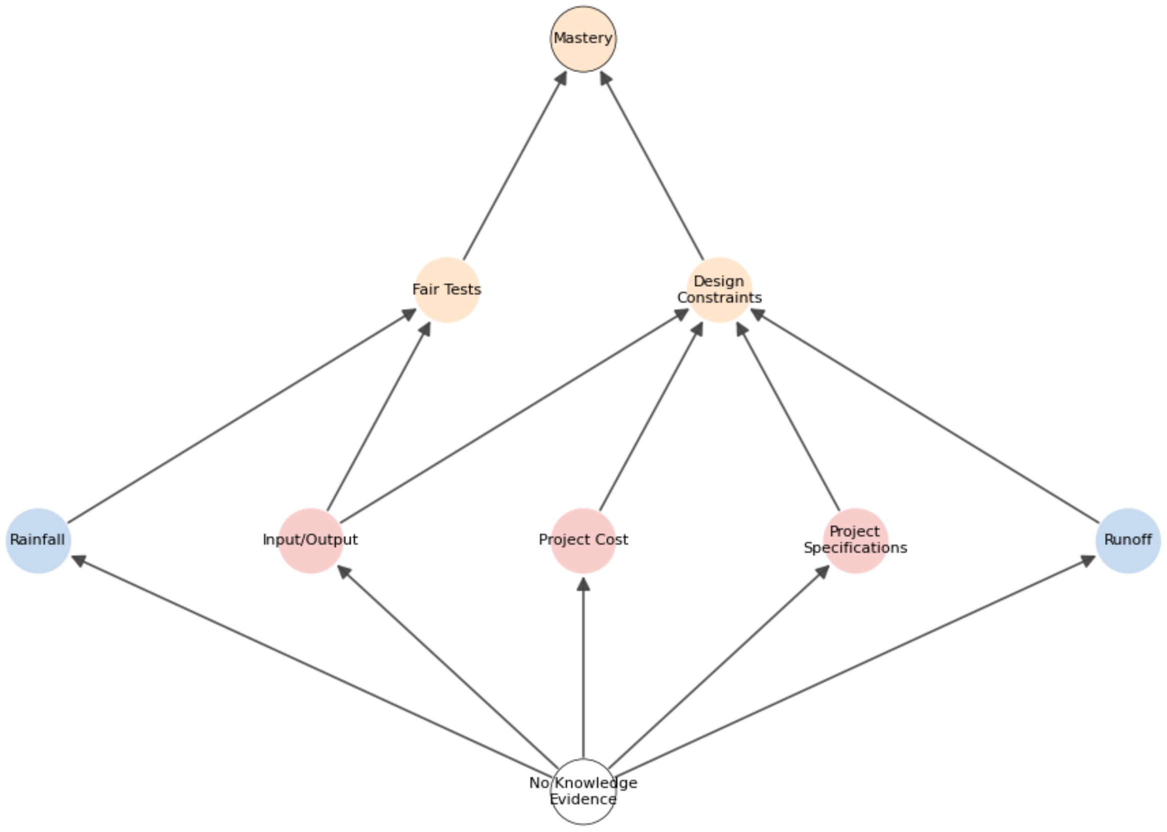


Figure 8: FA4 knowledge graph.