Traces and Ghosts

Forge 4 discussion:

- Prims' and Kruskal's will always produce a spanning tree with the same weights
- Something regarding Conway's game of Life

A bit about Integers:

- Forge has two types of integers: **atoms** and **values**
- In fields and relations, we can only use **integer atoms**
- To apply operations (e.g. add, subtract, >) on integers, we need to apply them to **values**
    - Can't compare e.g. an atom and a value
    - This is why sometimes you might need to do sum/min/etc on only 1 value
- Make sure to check out the Forge docs! This link is also on the Assignments page.

Back to our linked list modeling:

- When we check well-formedness of a linked list, we are limited by our bounds
    - Forge will only guarantee you correctness up to the bounds
    - This is reassuring in the sense that there's no small counterexample
- The small-scope hypothesis
    - Engineering principle
    - If there is a bug, there's usually a small demonstration of the bug

Projection:

- Projecting over Linked List means: every relation with a linked list on the left side, we will let you visualize those and go through them
- Useful when working on Stateful problems!

We've proven some invariants about linkedlists!

- Invariant 1:
    - 'Null' doesn't exit in Forge, so use 'no' instead
- Invariant 2ab:
    - s.ghost.univ is the set of all integers used as indices in this state
        - They are Int Atoms, so we can use min[] on them.
        - However, the indexing uses Int Values, so we use sing[] in order to convert it to the correct type.

Note about Forge 3 stack problem:

- What is 'Event'?
    - A way to impose structure to the transitions
    - Gives you better visibility into what the arguments are
    - Otherwise, you would just see the sequence of states; this tricks the visualizer into giving you more information.

Note about Goats & Wolves:

- No need for ‘next’ relation in the State sig