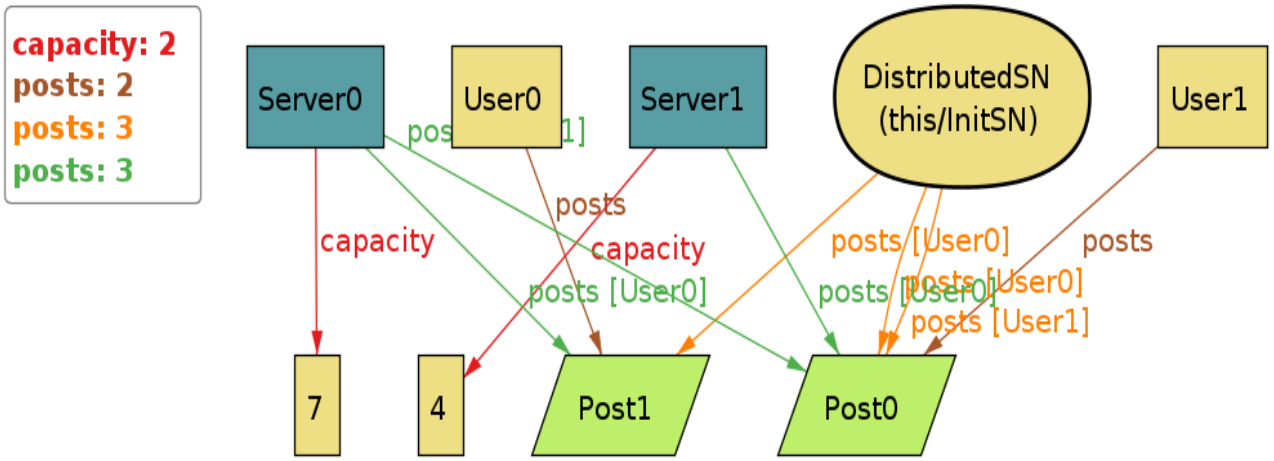
In alloy checks are made for consistency of the specification, invariants, algebraic properties of the operations, refinements and temporal properties.

Checking invariants on specific invariants

1. In asserting whether the addPost preserves the invariants in the Distributed Social Network we use the following code.

The code is

```
assert AddPostPreservesInv {
   // base case
   all sni : InitSN | invariant[sni]
   // inductive case
   all n1, n2 : DistributedSN, u : User, p : Post |
       invariant[n1] and addPost[n1, n2, u, p] implies
           invariant[n2]
}
check AddPostPreservesInv for 4 but exactly 1 DistributedSN
```

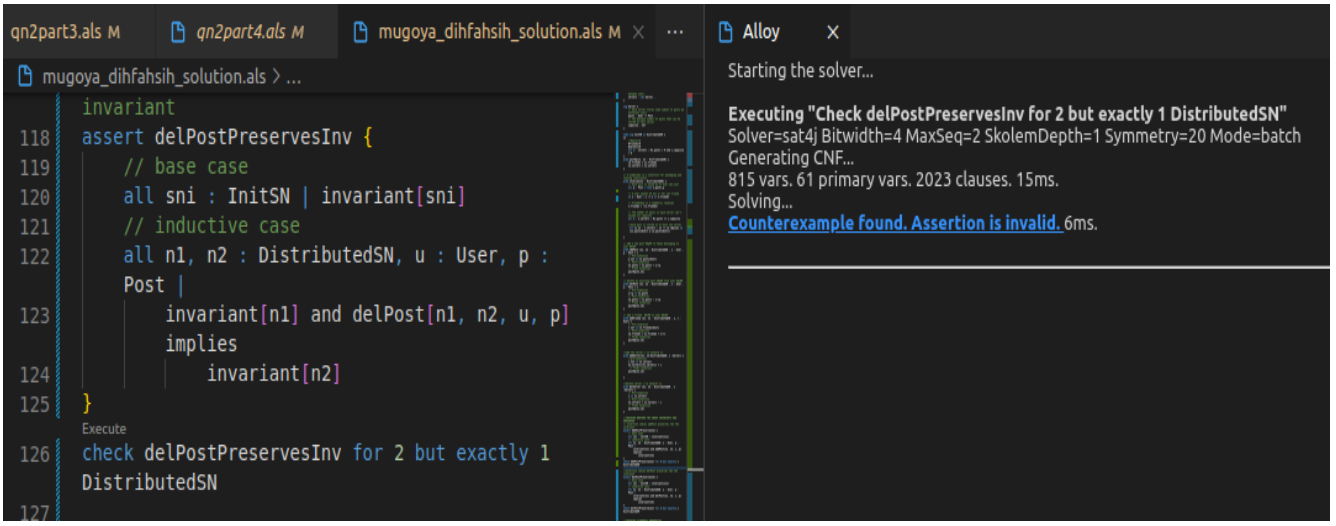


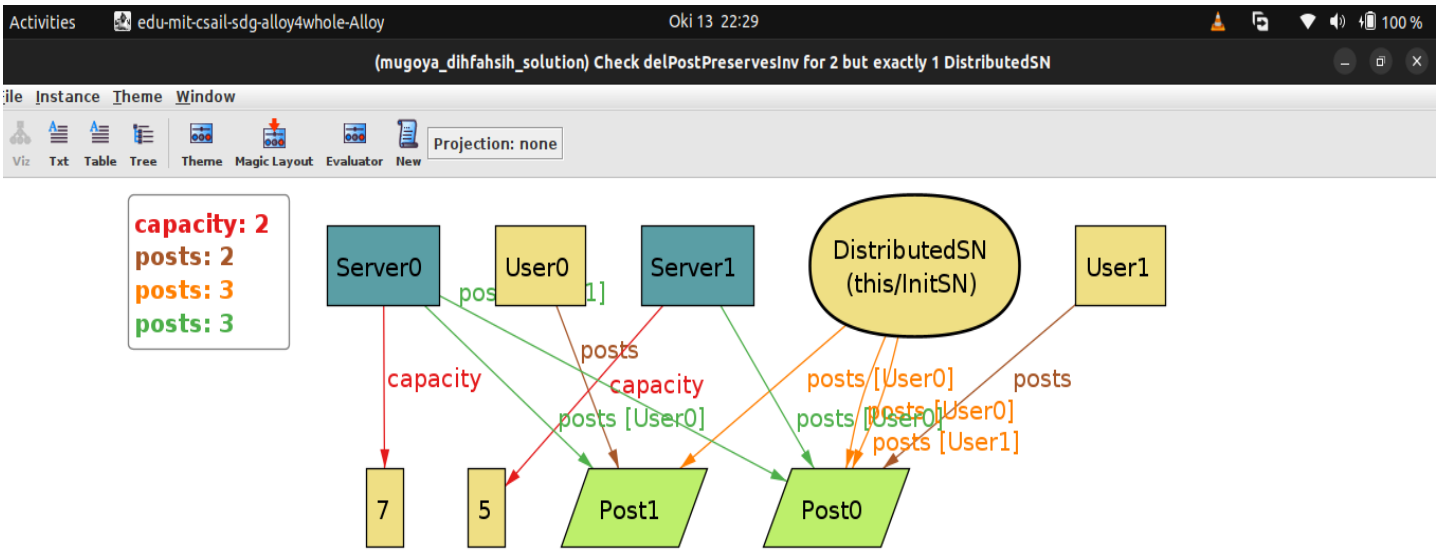
When we execute the above check code for 4 instances but on only one DistributedSN, alloy provides counter example instances that produce the above visualisation when the posts are added to the social network, the servers do not exceed the capacity specified which is always a positive number as defined in the operations, The posts are not shared among the users and they are stored in either two or more servers hence the invariants hold for this Distributed Social Network

The counterexample that alloy produces simply means that we need to refine our code and fix the loopholes discovered so as to have a valid specification.

2. Checking for invariant preservation in deletePost

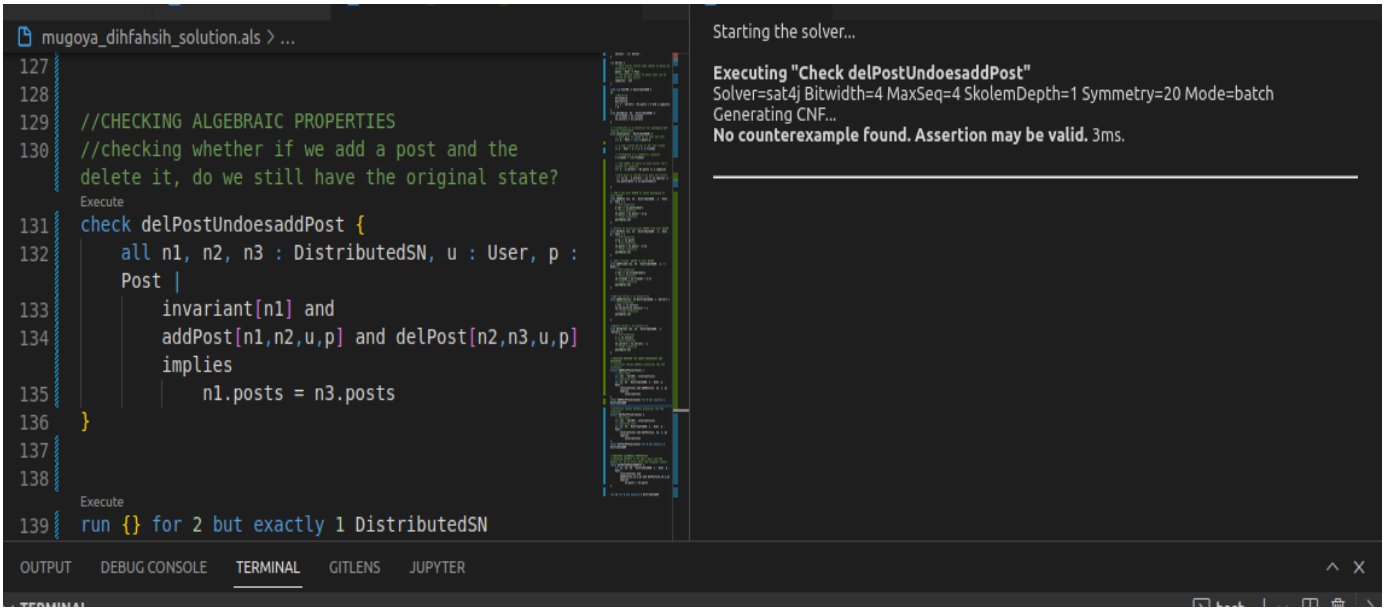
When the code is run, alloy produces a counterexample which means the invariant was not preserved and we need to fact check this assertion and fix the identified bugs as identified in the alloy visualizer.





The above alloy visualised code simply identifies that server capacity is not changed by deleting a post, positivity of the capacity holds but the user can only delete one post in the Network which makes the operation not hold for the specified invariants thus need for refinement of the code to meet the specified design of the Distributed Social Network

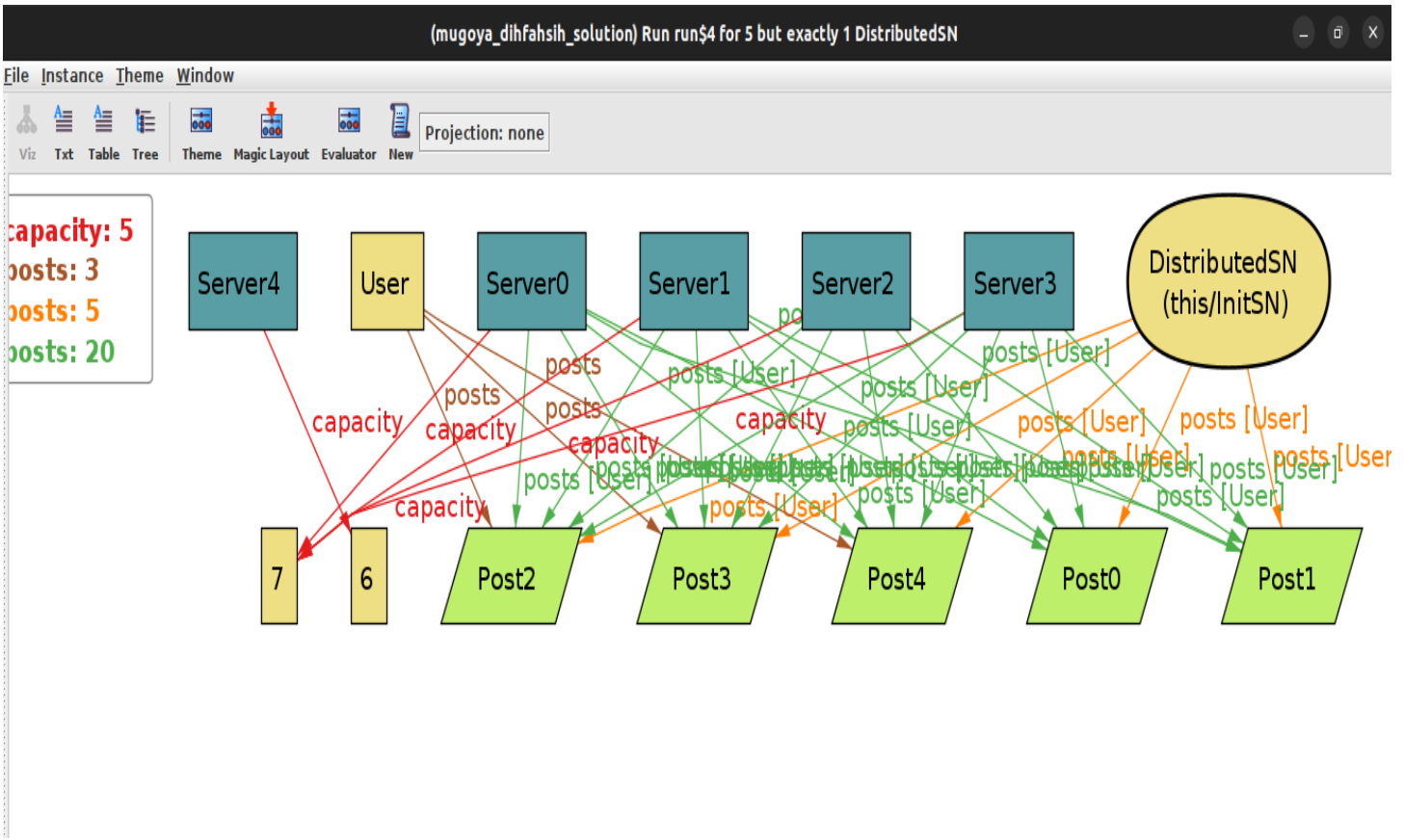
3. Checking Algebraic Operations



The above check produced no counterexample because the alloy analyzer did not find any loopholes in our adding a post and then deleting it. This means that the original state of our system is maintained and thus the assertion is valid making our specification probably correct to be used in the system.

Writing invariants for the specific properties on our system enables us to use alloy to explore different behaviours of our system as early as possible.

The Distributed Social Network after checking the properties are valid or are preserved, the alloy tool produces a more complex relational model as seen below.



This makes it hard to be analysed and thus making alloy tool cumbersome for large instances