Tower Puzzle Game

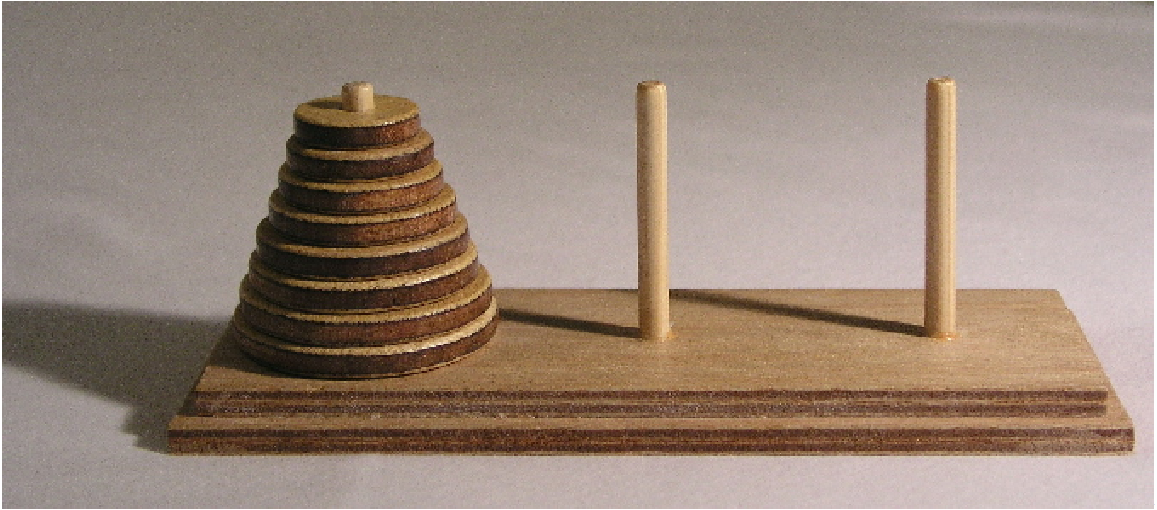
KF6009 – Model Based Design and Verification

Dr Alun Moon
alun.moon@northumbria.ac.uk

Northumbria University

Lecture 2.a

Tower puzzle
aka Tower of Hanoi, Lucas' Tower



Rules

The objective of the puzzle is to move the entire stack to another rod, obeying the following simple rules:

1. Only one disk can be moved at a time.
2. Each move consists of taking the upper disk from one of the stacks and placing it on top of another stack or on an empty rod.
3. No larger disk may be placed on top of a smaller disk.

With 3 disks, the puzzle can be solved in 7 moves. The minimal number of moves required to solve a Tower of Hanoi puzzle is $2^n - 1$, where n is the number of disks.

Model in TLA+

State

The *state* of the system is described by the sets of disks on the three posts, we'll need three variables A,B, and C

Model in TLA+

State

The *state* of the system is described by the sets of disks on the three posts, we'll need three variables A,B, and C

Represent state as?

The state of each post is the sequence of disks on the post. We need:

- preserve the order of disks
- represent the disk size

Model in TLA+

State

The *state* of the system is described by the sets of disks on the three posts, we'll need three variables A,B, and C

Represent state as?

The state of each post is the sequence of disks on the post. We need:

- preserve the order of disks
- represent the disk size

Sequences

We'll represent the state as a *Sequence* of integers, the integer being the number/size of the disk.

TLA

MODULE *TowerHanoi*

EXTENDS *Naturals*, *Sequences*

CONSTANT *Disks* — Set of disks in the problem

VARIABLE *A*, *B*, *C*

$$
\begin{aligned}
TypeInv \triangleq\ & \\
& \land A \in Seq(Disks) \quad \text{A is some sequence of Disks} \\
& \land B \in Seq(Disks) \quad \text{B is some sequence of Disks} \\
& \land C \in Seq(Disks) \quad \text{C is some sequence of Disks}
\end{aligned}
$$

- Naturals gives us numbers and ways to manipulate them
- Sequences gives us a sequence object and means to manipulate (Head, Tail, Concatenate)
- the operator $Seq(\mathcal{S})$ (from *Sequences*) is the set of all possible sequences drawn from the set $\mathcal{S}$

- Naturals gives us numbers and ways to manipulate them
- Sequences gives us a sequence object and means to manipulate (Head, Tail, Concatenate)
- the operator $Seq(\mathcal{S})$ (from *Sequences*) is the set of all possible sequences drawn from the set $\mathcal{S}$

Ordering

- We have a requirement that the sequence is ordered
- Smaller disks must sit on top of larger disks
- The values in the sequence *must* be in ascending order

Ordering a sequence

- A *Sequence* is a function that maps the values $1 \dots n$ to it's values
- The operator *Len* returns the number of values n in the sequence
- For ordering values $v_1, v_2, \dots, v_{n-1}, v_n$ we require, $v_1 \leq v_2 \leq \dots \leq v_{n-a} \leq v_n$

$Ordered(s) \triangleq$	define ordering property for sequence s
$\quad \forall\, n \in 2 \,..\, Len(s)$	for every element after the first
$\qquad : s[n-1] < s[n]$	it is greater than its predecessor

TLA with Ordering

MODULE *TowerHanoi*

EXTENDS *Naturals*, *Sequences*

CONSTANT *Disks* — Set of disks in the problem

VARIABLE A, B, C

$Ordered(s) \triangleq$ — define ordering property for sequence s

$\quad \forall\, n \in 2 \,..\, Len(s)$ — for every element after the first

$\quad\quad : s[n-1] < s[n]$ — it is greater than its predecessor

$TypeInv \triangleq$

$\quad \land\, A \in Seq(Disks) \land Ordered(A)$ — A is some ordered sequence of Disks

$\quad \land\, B \in Seq(Disks) \land Ordered(B)$ — B is some ordered sequence of Disks

$\quad \land\, C \in Seq(Disks) \land Ordered(C)$ — C is some ordered sequence of Disks

Initial state

The initial state is that all the disks are on one pole, and that they are in order.

- The disks are sized $1, 2, \ldots, n$
- we want the sequence to be, $[1 \mapsto 1, 2 \mapsto 2, \ldots, (n-1) \mapsto (n-1), n \mapsto n]$

Initial state for the polls

$Init \triangleq$

$\land\ A = [n \in Disks \mapsto n]$ A has disks in order

$\land\ B = \langle\rangle$ B empty

$\land\ C = \langle\rangle$ C empty

Completeness

We can define a predicate to show that the puzzle is complete.

- The puzzle is complete when the pile of disks is on either of the other two poles.

Initial conditions

$Complete \triangleq$	Puzzle is complete when
$\quad \lor\ B = [n \in Disks \mapsto n]$	B has disks in order, or
$\quad \lor\ C = [n \in Disks \mapsto n]$	C has disks in order

Moving disks

- We can split the action of moving disks into two parts
  1. An action that moves the top disk from one pole to another.
  2. An action that makes valid moves

Taking a disk from a pole leaves the rest of the disks on the pole

Putting a disk on a pole makes it the first disk in the sequence of disks on the pole

Valid moves are when

- Move a disk *from* an occupied pole
- to and empty pole
- to an occupied pin, if the disk is smaller that the disk on the top of the pile

Moving disks

From one pile to another

Taking a disk from a pole leaves the rest of the disks on the pole

Putting a disk on a pole makes it the first disk in the sequence of disks on the pole

$$
\begin{array}{ll}
\mathit{MoveDisk}(p,\ q) \triangleq & \text{Move a disk from the top of p to the top of q} \\
\quad \land\ p' = \mathit{Tail}(p) & \text{p' is everything but the first disk} \\
\quad \land\ q' = \langle \mathit{Head}(p) \rangle \circ q & \text{add the top disk on p to the pile on q}
\end{array}
$$

Moving disks

Valid moves

Valid moves are when

- Move a disk *from* an occupied pole
- to and empty pole
- to an occupied pin, if the disk is smaller that the disk on the top of the pile

$$
\begin{array}{ll}
\mathit{Move}(p,\ q) \ \triangleq & \text{Move disk from one pole to another} \\
\quad \land\ p \neq \langle\rangle & \text{From an occupied (not empty) pin} \\
\quad \land\ (& \\
\qquad \lor\ \ q = \langle\rangle & \text{to empty pin} \\
\qquad \lor\ \ q \neq \langle\rangle \land \mathit{Head}(p) < \mathit{Head}(q) & \text{to occupied pin} \\
\quad\ \ ) & \\
\quad \land\ \mathit{MoveDisk}(p,\ q) &
\end{array}
$$

Next Action

We need to define the *Next* action as the set of possible moves.

$$
\begin{aligned}
Next \triangleq \quad & \\
& \lor\ Move(A, B) \land \text{UNCHANGED } C \\
& \lor\ Move(A, C) \land \text{UNCHANGED } B \\
& \lor\ Move(B, A) \land \text{UNCHANGED } C \\
& \lor\ Move(B, C) \land \text{UNCHANGED } A \\
& \lor\ Move(C, A) \land \text{UNCHANGED } B \\
& \lor\ Move(C, B) \land \text{UNCHANGED } A
\end{aligned}
$$

Verifying the Model

We have enough to verify the model:

- We can set the *Initial* and *Next* predicates
- We can set an invariant to check

```
\lnot Complete
```

$\neg Complete$

- We can set the set of disks to $\{1, 2, 3\}$)

```
Disks <- 1 .. 3
```

$Disks \leftarrow 1 \mathrel{..} 3$

Results from TLC

- TLC reports that the invariant we set $\neg Complete$ has been violated,
  - *i.e.* it is complete.
- The error trace shows the moves
- The *Complete* predicate is false after 9 states
  - 16 moves
  - theory suggests best moves for n disks is 2^n

TLA/TLC solves logic puzzles.

Results from TLC

State	A	B	C	
1	$\langle 1, 2, 3 \rangle$	$\langle \rangle$	$\langle \rangle$	Initial predicate
2	$\langle 2, 3 \rangle$	$\langle 1 \rangle$	$\langle \rangle$	
3	$\langle 3 \rangle$	$\langle 1 \rangle$	$\langle 2 \rangle$	
4	$\langle 1, 3 \rangle$	$\langle \rangle$	$\langle 2 \rangle$	
5	$\langle 3 \rangle$	$\langle \rangle$	$\langle 1, 2 \rangle$	
6	$\langle \rangle$	$\langle 3 \rangle$	$\langle 1, 2 \rangle$	
7	$\langle 1 \rangle$	$\langle 3 \rangle$	$\langle 2 \rangle$	
8	$\langle 1 \rangle$	$\langle 2, 3 \rangle$	$\langle \rangle$	
9	$\langle \rangle$	$\langle 1, 2, 3 \rangle$	$\langle \rangle$	$\neg \mathit{Complete}$ violated