Group 24: Confirming Tents and Trees

Cooper Harasyn (15cjh7)

Felix Miao (15fm23)

Course Modelling Project

CMPE 204 - Logic for Computing Science

November 2nd, 2020

# Project Summary

Tents and Trees is a logic game that involves placing "tents" within a square board containing a random assortment of "trees".

The number of tents in each row or column is specified, and each tent must be next (or assigned) to one tree.

Tents cannot touch vertically, horizontally, or diagonally (i.e. there can only be one tent within any 2x2 square).

Our project aims to assess if a 5x5 Tents and Trees board has been solved.

A model will correspond to the tent and tree positions as well as the row and column tent counts which describe a solved configuration.

The images below illustrate a puzzle before and after solving.

A picture containing graphical user interface

Description automatically generatedGraphical user interface, application

Description automatically generated

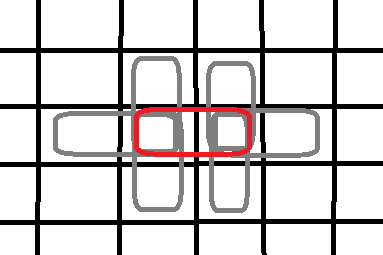
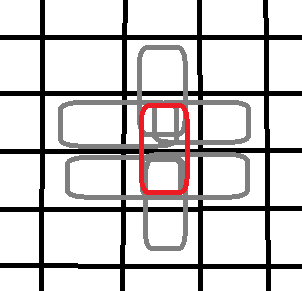
# Propositions

In our problem, there are two main types of propositions which we will consider.

* “Primary” propositions – correspond directly to puzzle state
  + ti,j: This is true if there is a tree at the location (i, j)
  + xi,j: This is true if there is a tent at the location (i, j)
  + ri;n: This is true when the row i should have n tents in the row
  + cj;n: This is true when the column j should have n tents in the column
* “Auxiliary” propositions – describing puzzle state in less direct terms
  + Ri: This is true when the row i has the correct number of tents
  + Cj: This is true when the column j has the correct number of tents
  + Pi,j: This is true when the tree at (i, j) has a paired tent
  + Ai,j: This is true when the tent at (i, j) does not have an adjacent tent
  + Bi,j;d: This is true when there is an association (tent/tree pair) between the cell at (i, j) and the adjacent cell in the direction d, where d=0 indicates the cell at (i+1, j) and d=1 indicates the cell at (i, j+1)
  + S: This is true when the board is solved

# Constraints

The constraints on our problem are as follows.

* Each row or column is defined to contain 0, 1, 2, or 3 tents
  + Maximum of three tents per row/column since tents can’t be adjacent on the 5x5 grid
  + Ex. For our particular board, row 4 has one tent, so there is the constraint (¬r4;0 ∧ r4;1 ∧ ¬r4;2 ∧ ¬r4;3)
* The board grid is modeled with the equation:  
  (¬t1,1 ∧ ¬x1,1) ∧ (¬t1,2 ∧ ¬x1,2) ∧ … ∧ (t1,5 ∧ ¬x1,5) ∧ (¬t2,1 ∧ x2,1) ∧ …   
  [similar for each grid square of the puzzle]
* Each row or column must have the correct number of tents
  + If ri;n is true then exactly n of xi,j (in the i-th row) must be true
  + Assuming the above condition holds, then Ri is true
* A tent cannot occupy the same spot as a tree, i.e. there is either a tree, tent, or nothing in each square
  + Each square (i,j) is defined by: (¬xi,j ∧ ¬ti,j) ∨ (xi,j ∧ ¬ti,j) ∨ (¬xi,j ∧ ti,j)
* A tent must not be adjacent to another tent, i.e. every 2x2 square can have at most one tent
  + Each 2x2 square is defined by:
    - (¬xi,j ∧ ¬ xi+1,j ∧ ¬xi,j+1 ∧ ¬xi+1,j+1) ∨
    - (xi,j ∧ ¬ xi+1,j ∧ ¬xi,j+1 ∧ ¬xi+1,j+1) ∨
    - (¬xi,j ∧ xi+1,j ∧ ¬xi,j+1 ∧ ¬xi+1,j+1) ∨
    - (¬xi,j ∧ ¬xi+1,j ∧ xi,j+1 ∧ ¬xi+1,j+1) ∨
    - (¬xi,j ∧ ¬xi+1,j ∧ ¬xi,j+1 ∧ xi+1,j+1)
* A tree must be paired with exactly one tent
  + This constraint will be imposed on the B\* propositions.
  + If one of the red pairings exist, then none of the grey pairings can exist.
  + 
  + 

# Model Exploration

Possible model exploration steps that are being considered include:

* Removing the constraining of the tent variables and allowing the SAT solver to solve the puzzle
  + This could also include examining each cell individually to determine if the cell either must be a tent, must not be a tent, or if it may or may not be a tent depending on the particular solution
  + We can also determine if a puzzle has more than one unique solution by determining if multiple models solve the same set of constraints
* Examining our constraints to see if we can “pare down” the constraints we have into a smaller set of rules for the game

# First-Order Extension

Our Tents and Trees problem extends well to first-order logic. To extend it to first-order logic, we could define the following functions over the set of positions in the grid:

* T(p): Tree function. It is T if the grid at position p contains a tree, and F otherwise.
* X(p): Tent function. It is T if the grid at position p contains a tent, and F otherwise.
* P(p1,p2): Partial adjacency function. This function is commutative. It is T if positions p1 and p2 are adjacent horizontally, or vertically, but not diagonally. It is F otherwise.
* F(p1,p2): Full adjacency function. This function is commutative. It is T if positions p1 and p2 are adjacent horizontally, vertically, or diagonally. It is F otherwise.
* A(p1,p2): Association function. This function is commutative. It is T if the tent/tree at p1 is associated (paired) with the tree/tent at p2, and F otherwise.
* E(p1,p2): Equality function. It is T if p1 is the same position as p2, and F otherwise.

This would allow us to represent our constraints more effectively. Instead of having one constraint per cell in the grid, we could do as follows: (note that two-letter variables are used as shorthand for subscript, meaning pt -> p­t)

* A tent cannot occupy the same spot as a tree
  + ∀p. ¬(T(p) ∧ X(p))
* A tent associated with a tree must be partially adjacent to it
  + ∀pt. (T(pt) → ∀px. (X(px) ∧ A(pt,px) → P(pt,px)))
* A tree must be associated (paired) with a tent
  + ∀pt. (T(pt) → ∃px1. (X(px1) ∧ A(pt,px1)))
* A position cannot be associated with multiple other positions
  + ∀p1. (∀p2. (A(p1,p2) → (∀p3. (A(p1,p3) → E(p2,p3)))))
  + If any p1 is associated with any p2, then any p3 which is also associated with p1 must be equal to p2

# Requested Feedback

1. We got stuck on D3 (Jape proofs) regarding what to prove. We considered looking at the cell constraint (a tent cannot occupy the same spot as a tree) and concluding xi,j NAND ti,j but we have not covered this in class.
2. The constraining of association between one tent and one tree is not immediately apparent. We’ve managed to solve it in the first-order extension, and we believe that we can do something similar in the propositional logic section, but we are open to any feedback on better ways to do this.
3. The constraint for checking that the number of tents placed is the same as the corresponding hint can be implemented with an adder network, but we are not sure which type of adder to use. Here, we are adding five 1-bit values together to check against the hint number. From our understanding, adder networks typically add multi-bit values together. We are open to suggestions on implementations for this.