Map: Let the size of map be mxn

Variables:

Variables Xij to represent the state of each tile on the grid, where i denotes the row number and j denotes the column number. [i in range(m), j in range(n)]

Xij = False: represents a (revealed) tile with a gem.

Xij = True: represents a (revealed) trap.

Xij = k: represents a **beginning tile** with k is the number of trap within its surroundings

Xij = None: represents an unreavel tile (can be True or False only after being revealed)

Rules and infered subrules:

1. Beginning tile is not appled by the rule of reavel or unreavel.
2. Each unrevealed tile can either be a trap or gem.
3. The revealed tiles with traps should not be safe.
4. The revealed tiles with gems are safe.
5. Beginning tiles at corner have k in range(3)
6. Beginning tiles beside boulder but not corner have k in range(5)
7. Beginning tiles not beside any boulder have k in range(8)
8. Each beginning tile has a number represents the number of traps (k) surrounding it
   1. If number of unrevealed tiles surrounding is equal to k then k unrevealed tiles surrounding are traps (and are revealed).
   2. If k is equal to 0 then all surrounding tiles are not trap (which mean it is revealed as gem if it is an unrevealed tile)
   3. If there are enough k surrounding traps being revealed then all remaining surrounding unrevealed tiles are gems.
   4. If there are not enough k surrounding traps being revealed (n trap tiles, n < k) then the remaining surrounding unreavel tiles can be either traps or gems. And the probability of trap each tile is (k – n) / (number of unrevealed tiles)
9. A unrevealed tiles with 100% be traps or gems must be revealed first

Advanced rules

1. Conflicted probability with no 100% options (after reveal all 100% unrevealed tile) might lead to multiple possible outcomes.
2. An area of x unreavel tiles can be determined to have y (x < y) trapped tiles but no surefire option yet.
3. If an area of x is within surronding tiles of a beginning tile, (k – y) is the number of remain traps.

Advanced rules 2 and 3 examples:

A screenshot of a game

Description automatically generated

Convert gameplay-related rules to CNF:

Definition

* Predicates:
  + Revealed(i, j): Represents that the tile at position (i, j) is revealed. (opposite is unrevealed)
  + Trap(x, y): Represents that the tile at position (i, j) is a trap. (opposite is gem)
  + Surrounding(i1, j1, i2, j2): Represents that the tile at position (i2, j2) is a neighbor of the tile at position (i1, j1).
  + Begin(i, j, k): Represents that tile at position (i, j) is a beginning tile that has k unrevealed trap surrounding it.

Converting to First-order-logic

Each beginning tile has a number represents the number of traps (k) surrounding it

* ∀i, j, k Begin(i, j, k)
  1. If number of unrevealed tiles surrounding is equal to k then k unrevealed tiles surrounding are traps (and are revealed).
* ∀i, j, k (Begin(i, j, k) ∧ (∃i1, j1, i2, j2…ik, jk, ~∃ik+1, jk+1 ((i, j) != (i1, j1) != (i2, j2)… != (ik, jk) != (ik+1, jk+1)) ∧ Surrounding(i, j, i1, j1) ∧ Surrounding(i, j, i2, j2) … ∧ Surrounding(i, j, ik, jk) ∧ ~Revealed(i1, j1) ∧ ~Revealed(i2, j2) … ∧ ~Revealed(ik, jk) ∧ (~Revealed(ik+1, jk+1) ∧ Surrounding(i, j, ik+1, jk+1)) ⇒ (Trap(i1, j1) ∧ Trap(i2, j2) … ∧ Trap(ik, jk) ∧ Revealed(i1, j1) ∧ Revealed(i2, j2) … ∧ Reveal(ik, jk))
  1. If k is equal to 0 then all surrounding tiles are not trap (which mean it is revealed as gem if it is an unrevealed tile)
* ∀i1, j1 (Begin(i1, j1, 0) ∧ (∀i2, j2 ((i1, j1) != (i2, j2)) ∧ ~Revealed(i2, j2) ∧ Surrounding(i1, j1, i2, j2)) ⇒ (~Trap(i2, j2) ∧ Revealed(i2, j2))
  1. If there are enough k surrounding traps being revealed then all remaining surrounding unrevealed tiles are gems.
* ∀i, j, k (Begin(i, j, k) ∧ (∃i1, j1, i2, j2…ik, jk ((i, j) != (i1, j1) != (i2, j2)… != (ik, jk)) ∧ Surrounding(i, j, i1, j1) ∧ Surrounding(i, j, i2, j2) … ∧ Surrounding(i, j, ik, jk) ∧ Trap(i1, j1) ∧ Trap(i2, j2) … ∧ Trap(ik, jk)) ⇒ [(∀ik+1, jk+1 ((i, j) != (i1, j1) != (i2, j2)… != (ik, jk) != (ik+1, jk+1)) ∧ Surrounding(i, j, ik+1, jk+1) ∧ ~Revealed(ik+1, jk+1)) ⇒ (~Trap(ik+1, jk+1) ∧ Revealed(ik+1, jk+1))]