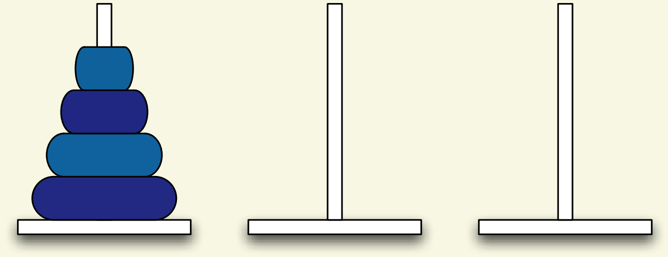
# Effective CNF Encodings for the Towers of Hanoi

Ruben Martins Inês Lynce

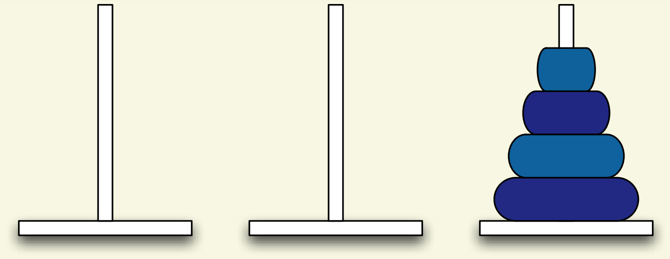
IST/INESC-ID
Technical University of Lisbon, Portugal

LPAR 2008, Qatar

# Towers of Hanoi (ToH)



# Towers of Hanoi (ToH)

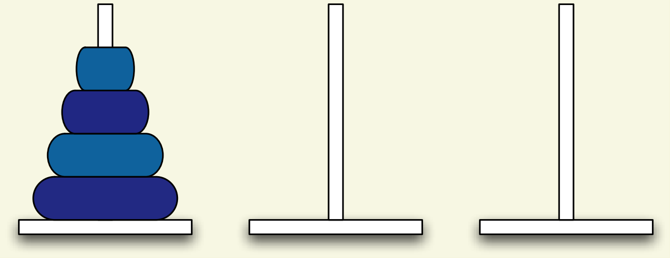


- Only one disk may be moved at a time;
- No disk may be placed on the top of a smaller disk;
- Each move consists in taking the upper disk from one of the towers and sliding it onto the top of another tower.

# Why ToH?

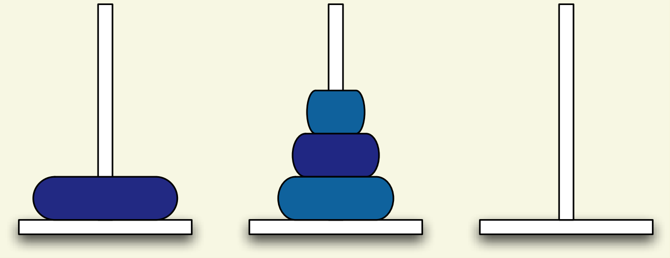
- Existing CNF encodings for ToH are hard to solve for SAT solvers;
- Although SAT technology has never advocated being the best approach for ToH this comes as a surprise;
- We propose a new encoding that incorporates a key number of properties of the ToH which seem to be essential for solving the problem;
- Show that even though SAT solvers are becoming more efficient the modelling still has an importante role for an effective solution.

# ToH Properties I



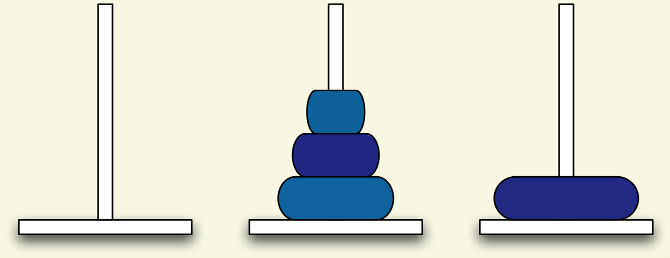
- Given a ToH of size $n$, one may easily find a solution taking into account the solution for a ToH of size $n-1$;

# ToH Properties I



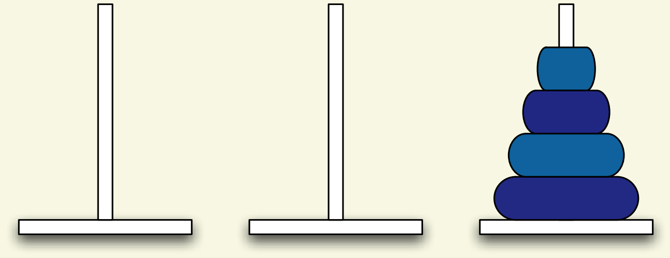
- Given a ToH of size $n$, one may easily find a solution taking into account the solution for a ToH of size $n - 1$;

# ToH Properties I



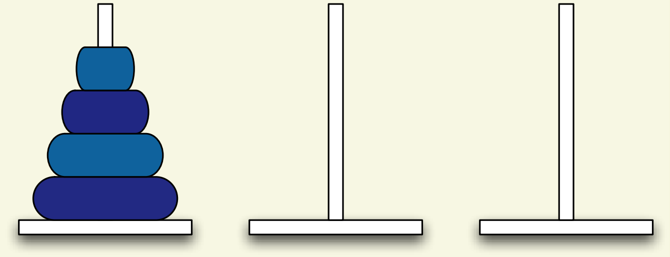
- Given a ToH of size $n$, one may easily find a solution taking into account the solution for a ToH of size $n-1$;
- The order of the disks to be moved after moving the largest disk is exactly the same as before;
- Notice that there is a relation between the towers involved in the same movement before and after the largest disk is moved that consists in shifting the towers from/to where the disks have to be moved ($T_1 \rightarrow T_2$, $T_2 \rightarrow T_3$, $T_3 \rightarrow T_1$).

# ToH Properties I



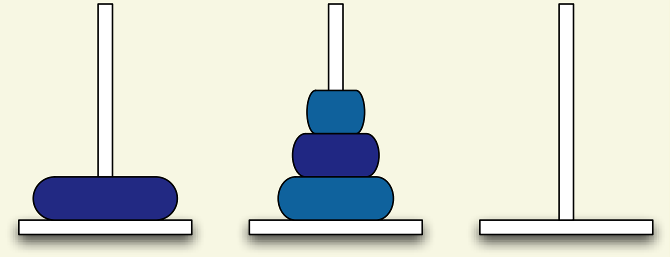
- Given a ToH of size $n$, one may easily find a solution taking into account the solution for a ToH of size $n-1$;
- The order of the disks to be moved after moving the largest disk is exactly the same as before;
- Notice that there is a relation between the towers involved in the same movement before and after the largest disk is moved that consists in shifting the towers from/to where the disks have to be moved ($T_1 \rightarrow T_2$, $T_2 \rightarrow T_3$, $T_3 \rightarrow T_1$).

# ToH Properties II



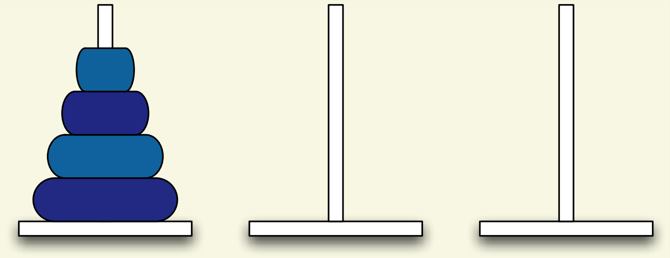
- ToH can be solved in $2^n - 1$ steps;
- Considering the relationship between the movement of the disks after/before moving the largest disk we only need to determine the first $2^{n-1} - 1$ steps

# ToH Properties II



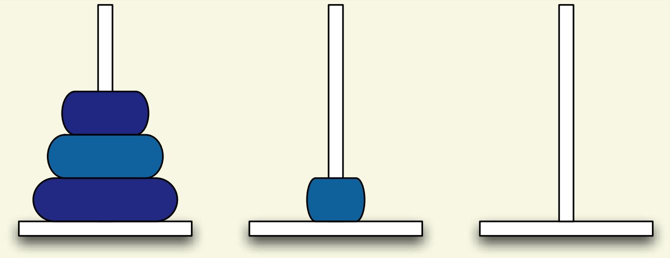
- ToH can be solved in $2^n - 1$ steps;
- Considering the relationship between the movement of the disks after/before moving the largest disk we only need to determine the first $2^{n-1} - 1$ steps

# ToH Properties III



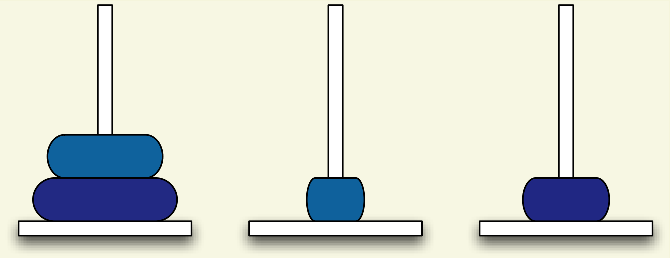
- When moving disks, no two odd/even disks can be moved next to each other.

# ToH Properties III



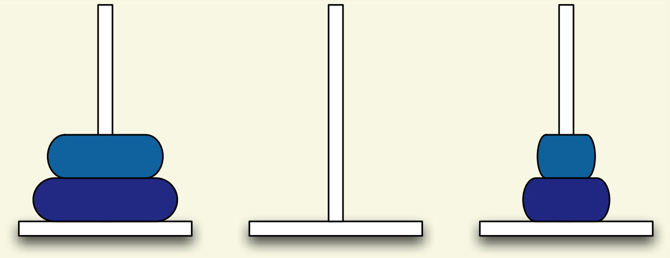
- When moving disks, no two odd/even disks can be moved next to each other.

# ToH Properties III



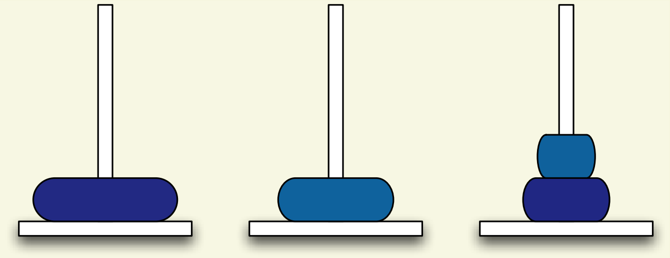
- When moving disks, no two odd/even disks can be moved next to each other.

# ToH Properties III



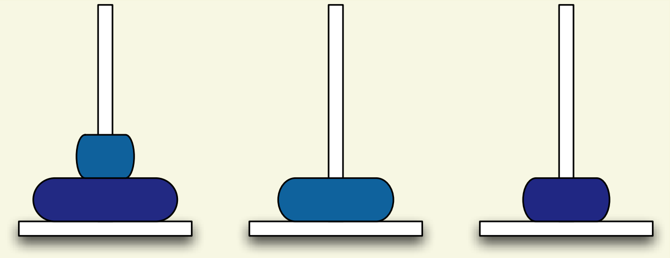
- When moving disks, no two odd/even disks can be moved next to each other.

# ToH Properties III



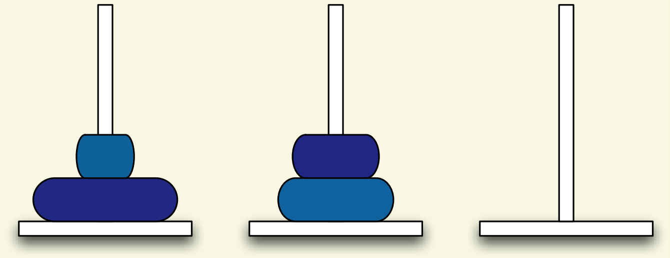
- When moving disks, no two odd/even disks can be moved next to each other.

# ToH Properties III



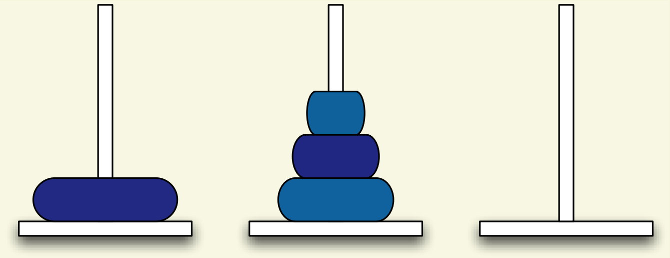
- When moving disks, no two odd/even disks can be moved next to each other.

# ToH Properties III



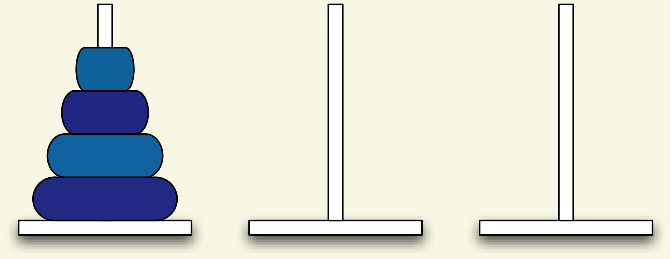
- When moving disks, no two odd/even disks can be moved next to each other.

# ToH Properties III



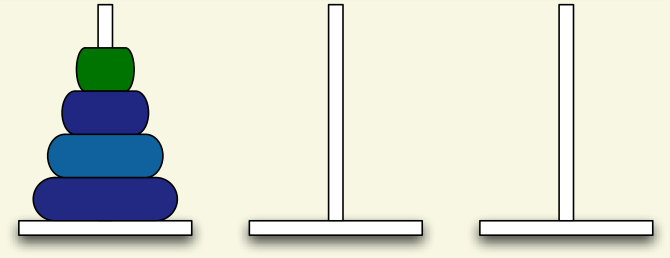
- When moving disks, no two odd/even disks can be moved next to each other.

# ToH Properties IV



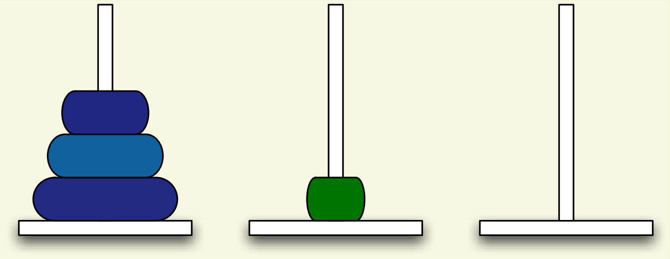
- All disks cycle in a given order between the towers:
  - If $n$ is even the odd disks will cycle clockwise ($T_1 \to T_2 \to T_3 \to T_1$) while the even disks will cycle counterclockwise ($T_1 \to T_3 \to T_2 \to T_1$);
  - If $n$ is odd the odd disks will cycle counterclockwise while the even disks will cycle clockwise.

# ToH Properties IV



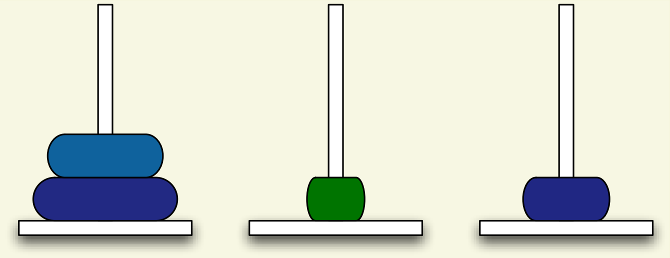
- All disks cycle in a given order between the towers:
    - If $n$ is even the odd disks will cycle clockwise ($T_1 \rightarrow T_2 \rightarrow T_3 \rightarrow T_1$) while the even disks will cycle counterclockwise ($T_1 \rightarrow T_3 \rightarrow T_2 \rightarrow T_1$);
    - If $n$ is odd the odd disks will cycle counterclockwise while the even disks will cycle clockwise.

# ToH Properties IV



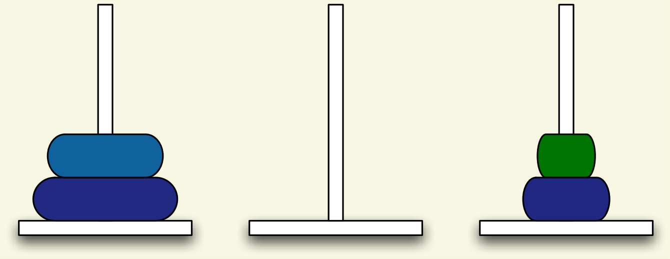
- All disks cycle in a given order between the towers:
  - If $n$ is even the odd disks will cycle clockwise ($T_1 \rightarrow T_2 \rightarrow T_3 \rightarrow T_1$) while the even disks will cycle counterclockwise ($T_1 \rightarrow T_3 \rightarrow T_2 \rightarrow T_1$);
  - If $n$ is odd the odd disks will cycle counterclockwise while the even disks will cycle clockwise.

# ToH Properties IV



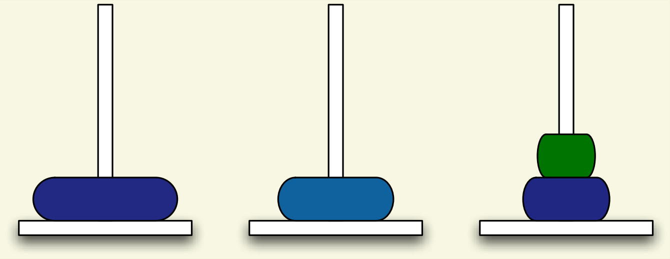
- All disks cycle in a given order between the towers:
  - If $n$ is even the odd disks will cycle clockwise ($T_1 \rightarrow T_2 \rightarrow T_3 \rightarrow T_1$) while the even disks will cycle counterclockwise ($T_1 \rightarrow T_3 \rightarrow T_2 \rightarrow T_1$);
  - If $n$ is odd the odd disks will cycle counterclockwise while the even disks will cycle clockwise.

# ToH Properties IV



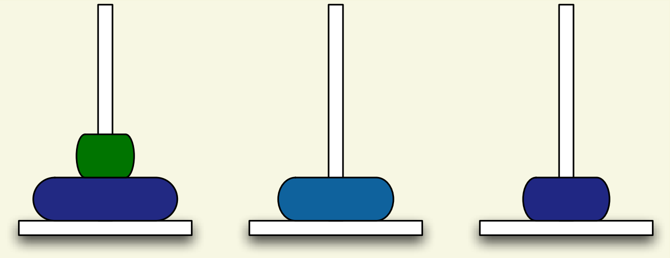
- All disks cycle in a given order between the towers:
  - If $n$ is even the odd disks will cycle clockwise ($T_1 \rightarrow T_2 \rightarrow T_3 \rightarrow T_1$) while the even disks will cycle counterclockwise ($T_1 \rightarrow T_3 \rightarrow T_2 \rightarrow T_1$);
  - If $n$ is odd the odd disks will cycle counterclockwise while the even disks will cycle clockwise.

# ToH Properties IV



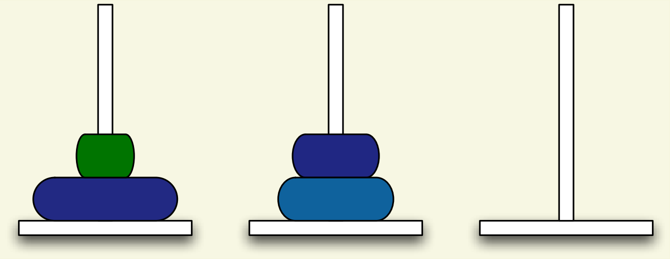
- All disks cycle in a given order between the towers:
  - If $n$ is even the odd disks will cycle clockwise ($T_1 \rightarrow T_2 \rightarrow T_3 \rightarrow T_1$) while the even disks will cycle counterclockwise ($T_1 \rightarrow T_3 \rightarrow T_2 \rightarrow T_1$);
  - If $n$ is odd the odd disks will cycle counterclockwise while the even disks will cycle clockwise.

# ToH Properties IV



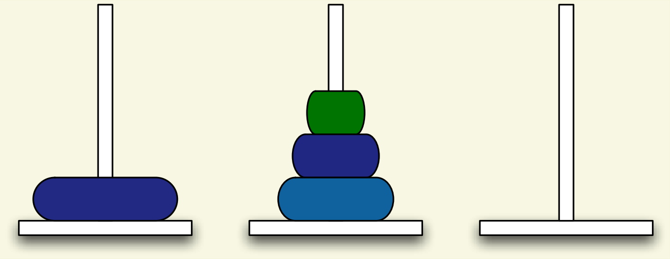
- All disks cycle in a given order between the towers:
  - If $n$ is even the odd disks will cycle clockwise ($T_1 \rightarrow T_2 \rightarrow T_3 \rightarrow T_1$) while the even disks will cycle counterclockwise ($T_1 \rightarrow T_3 \rightarrow T_2 \rightarrow T_1$);
  - If $n$ is odd the odd disks will cycle counterclockwise while the even disks will cycle clockwise.

# ToH Properties IV



- All disks cycle in a given order between the towers:
  - If $n$ is even the odd disks will cycle clockwise ($T_1 \rightarrow T_2 \rightarrow T_3 \rightarrow T_1$) while the even disks will cycle counterclockwise ($T_1 \rightarrow T_3 \rightarrow T_2 \rightarrow T_1$);
  - If $n$ is odd the odd disks will cycle counterclockwise while the even disks will cycle clockwise.

# ToH Properties IV



- All disks cycle in a given order between the towers:
  - If $n$ is even the odd disks will cycle clockwise ($T_1 \rightarrow T_2 \rightarrow T_3 \rightarrow T_1$) while the even disks will cycle counterclockwise ($T_1 \rightarrow T_3 \rightarrow T_2 \rightarrow T_1$);
  - If $n$ is odd the odd disks will cycle counterclockwise while the even disks will cycle clockwise.

# Existing CNF Encodings for ToH

- The first CNF encoding uses the following set of variables:
  - $\{on(d, dt, t), clear(dt, t), move(d, dt, dt', t)\}$, where the variable $move(d, dt, dt', t)$ is replaced by conjunction $(obj(d, t) \wedge from(dt, t) \wedge to(dt', t))$.

# Existing CNF Encodings for ToH

- The first CNF encoding uses the following set of variables:
  - $\{on(d, dt, t), clear(dt, t), move(d, dt, dt', t)\}$, where the variable $move(d, dt, dt', t)$ is replaced by conjunction $(obj(d, t) \wedge from(dt, t) \wedge to(dt', t))$.
- A recent encoding proposed by Prestwich produces a more compact formula since it restricts the action $move(d, tw, tw', t)$ to consider only movements of disks between towers.
  - For example, for 5 towers the previous encoding requires 1,931 variables and 14,468 clauses, whereas this new encoding requires 821 variables and 6,457 clauses.

# Effective CNF Encodings for ToH

- The Disk Parity encoding:
  - We improved Prestwich encoding by incorporating properties 2 and 3:
    - Property 2 is incorporated in the encoding by reducing the search space to the first $2^{n-1} - 1$ steps and modifying the goal state so that we have the larger disk on $T_1$ and the remaining disks on $T_2$;
    - Property 3 is encoded through additional clauses.

# Effective CNF Encodings for ToH

- The Disk Parity encoding:
  - We improved Prestwich encoding by incorporating properties 2 and 3:
    - Property 2 is incorporated in the encoding by reducing the search space to the first $2^{n-1} - 1$ steps and modifying the goal state so that we have the larger disk on $T_1$ and the remaining disks on $T_2$;
    - Property 3 is encoded through additional clauses.
- The Disk Cycle encoding:
  - This encoding extends the previous one by addionally incorporating property 4.

# Effective CNF Encodings for ToH

- The Disk Sequence encoding:
  - Property 1 recursively determines the disks to be moved at each step;
  - Taking into consideration this we can keep only the variables *on* and drop all the others;
  - Our set of variables is the following: $on(d, tw, t)$ with $1 \leq d \leq n$, $1 \leq tw \leq 3$ and $0 \leq t \leq 2^{n-1}$;
  - By addionally incorporating properties 2 and 3 we are able to produce an encoding that is solved by only using unit propagation.

# Experimental Results I

Table: Results for the encodings with a time limit of 10,000 seconds.

| Size | Prestwich | Disk Parity | Disk Cycle | Disk Sequence |
|---|---|---|---|---|
| 4 | 0.01 | 0 | 0 | 0 |
| 5 | 0.08 | 0.01 | 0.02 | 0 |
| 6 | 0.47 | 0.03 | 0.05 | 0 |
| 7 | 3.65 | 0.70 | 0.20 | 0.01 |
| 8 | 109.7 | 5.19 | 5.18 | 0.03 |
| 9 | 7126.57 | 79.11 | 7.65 | 0.09 |
| 10 | - | 1997.19 | 973.95 | 0.23 |
| 11 | - | - | 1206.37 | 0.56 |
| 12 | - | - | - | 1.32 |

# Experimental Results II

Table: Results for the new encoding for the ToH.

| Size | #Vars | #Cls | Mem | GenTime | SolveTime |
|---|---|---|---|---|---|
| 13 | 159,705 | 827,007 | 20.9 | 0.74 | 3.14 |
| 14 | 344,022 | 1,859,150 | 50.9 | 1.73 | 7.33 |
| 15 | 737,235 | 4,177,431 | 121.6 | 4.07 | 17.16 |
| 16 | 1,572,816 | 9,272,800 | 294.1 | 9.44 | 38.94 |
| 17 | 3,342,285 | 20,577,699 | 708.6 | 12.31 | 90.15 |
| 18 | 7,077,834 | 45,219,174 | 1,637.4 | 50.48 | 203.05 |

# Conclusions

- Making use of properties of the ToH we were able to improve the previous CNF encodings;
- Taking advantage of those properties we were also able to produce a new encoding that is more compact and can solve ToH by only using unit propagation;
- This is an example that modelling as an important role for an effective solution since a problem that was intractable at first for SAT solvers can now be solved by only using unit propagation.