Toothpick Game

There are two players, A (Amanda) and B (Billy). They move alternately. In each move, A can pick either 1 or 4 toothpicks from the table. In each move, B can pick either 1, 3 or 5 toothpicks from the table. The winner of the game is the person who gets the last toothpick. (If there are 2 toothpicks, and for example its A’s move – then A must pick 1 toothpick, since it is not possible to pick 4 toothpicks.)

How do we find a winning strategy for A (if one exists)? We are interested in doing this for A since she is the smarter of the two, and knows dynamic programming!

We do this by defining two functions \( f \) and \( g \) (value of \( f(n) \) and \( g(n) \) is 0 or 1).

\[
f(n) = 1 \text{ iff there is a winning strategy for A given that it is A's move and there are } n \text{ toothpicks}
\]

\[
g(n) = 1 \text{ iff there is a winning strategy for A given that it is B's move and there are } n \text{ toothpicks}
\]

Assuming \( n > 5 \) (we can easily compute the function for small values of \( n \)) we can define \( f \) and \( g \) recursively as follows.

\[
f(n) = \max(g(n - 1), g(n - 4)).
\]

This is because A has a choice of picking either 1 or 4 toothpicks, clearly A will make the move that enables it to continue winning after its move (the function \( g \) gives us this information).

\[
g(n) = \min(f(n - 1), f(n - 3), f(n - 5)).
\]

We make the assumption that B will also be playing intelligently. If B makes an error (for example B picks 1 toothpick when \( f(n - 1) = 1 \) and \( f(n - 3) = 0 \), then from that point on A will win; clearly, picking 3 toothpicks was the right choice for B).

We can use the above equation to compute the \( f \) and \( g \) values (fill the rest of the table yourself).

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
</tr>
</thead>
<tbody>
<tr>
<td>( f )</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( g )</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A move</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B move</td>
<td>1</td>
<td>3</td>
<td>5</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 1: Example for small values of \( n \).

Observe that if \( f(n) = 1 \) then the corresponding entry in the 3rd row tells A what move to make. If \( f(n) = 0 \) then it does not matter what move A makes (since neither move can guarantee a win). If \( g(n) = 0 \) then A cannot guarantee a win. As a result, the corresponding entry in the 4th row tells B what move to make. If \( g(n) = 1 \) then it does not matter what move B makes (neither move can guarantee B a win).