Isolation Game Agent

by Miguel Ángel Martínez Fernández

www.github.com/miguelangel

The challenge

This project will cover the development of an adversarial search agent to play the game *Isolation*.

Isolation is a deterministic, two-player game of perfect information in which the players alternate turns moving a single piece from one cell to another on a board. Whenever either player occupies a cell, that cell becomes blocked for the remainder of the game. The first player with no remaining legal moves loses, and the opponent is declared the winner.

This project uses a version of Isolation where each agent is restricted to L-shaped movements (like a knight in chess) on a rectangular grid (like a chess or checkerboard). The agents can move to any open cell on the board that is 2-rows and 1-column or 2-columns and 1-row away from their current position on the board. Movements are blocked at the edges of the board (the board does not wrap around), however, the player can "jump" blocked or occupied spaces (just like a knight in chess).

Additionally, agents will have a fixed time limit for each turn to search for the best move and respond. If the time limit expires during a player's turn, that player forfeits the match, and the opponent wins.

The approach

The game will be solved by implementing both, the *MiniMax* and the *Alpha Beta Pruning* search algorithms.

Additionally, three different heuristic functions will be implemented. These algorithims will be used to score the different board states for any of the players.

The Solution

MiniMax and *Alpha Beta Pruning* algorithms are well-known by computer science community, being its implementation quite straightforward. Our implementation tries to balance performance and readibility.

Heuristic functions are a different thing. Creativity and genius are needed to make the differente between an average solution and a solution which stands out from the rest.

Our solution implements the following heuristic functions:

> **Easy score** evaluates de board position by calculating the difference of moves between the active player and the opponent's.
>
> **Offensive score** calculates the difference of moves between the active player and twice the opponent's. This function is a bit more aggresive than easy score. We will have to wait to the tests to see how it performs.
>
> **Chaser score** is a more creative solution. It is based on easy score but enriched by having into consideration the number of moves where both players overlap.

The Results

A few hundreds of hours later, the moment of truth is upon us. The solution is coded and it is time to test it.

In order to estimate the strength of our agent, it has been tested as follows:

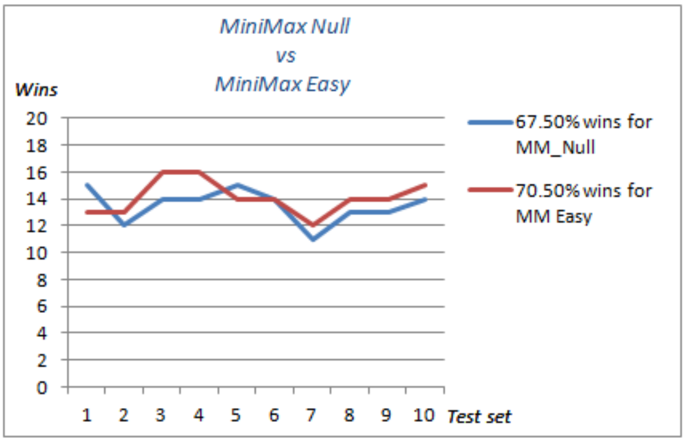
- The student agent plays a fixed number of *fair* matches against each test agent. The matches are fair because the board is initialized randomly for both players, and the players play each match twice -- switching the player order between games.
- Each heuristic evaluation function (*easy score*, *offensive score* and *chaser score*) are tested against *random play*, *fixed-depth minimax* and *alpha beta pruning* search agents by running a round-robin tournament for the student agent.

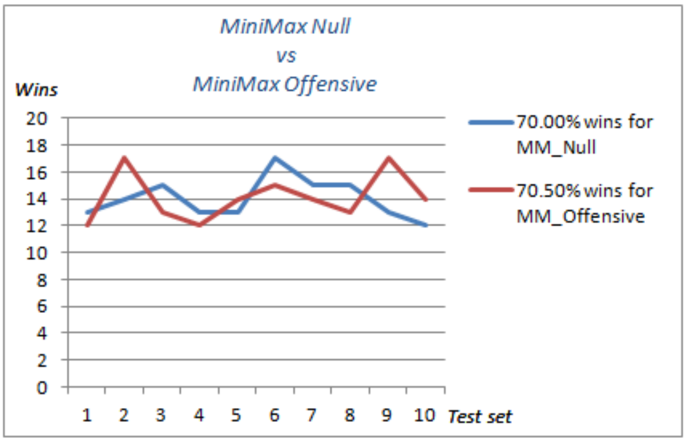
#4200 games later...

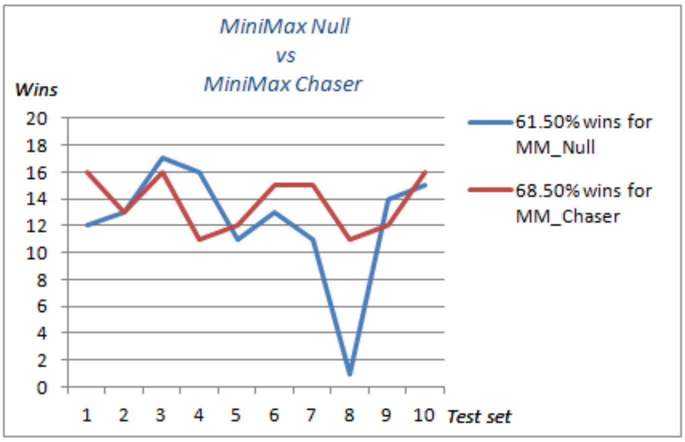
MiniMax Null Results

This test set is designed to evaluate our agent strength (in red) under the following conditions:

- Search algorithm: *MiniMax*
- Opponent heuristic evaluation function: *Null*
- Our heuristic evaluation functions:
    - *Easy score*
    - *Offensive score*
    - *Chaser score*







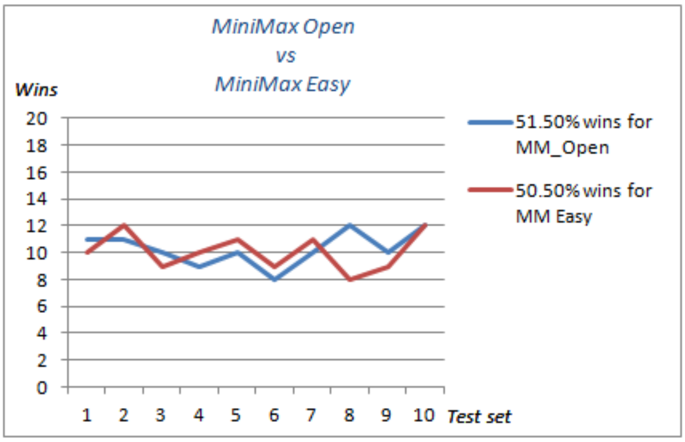
Based on the graphics above, it can be concluded that:

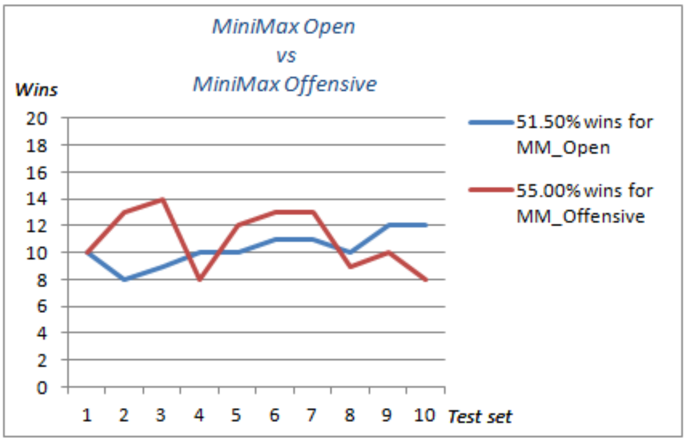
- All our heuristic functions outperform the *Null* heuristic.
- Our *chaser score* heuristic outperforms the others.

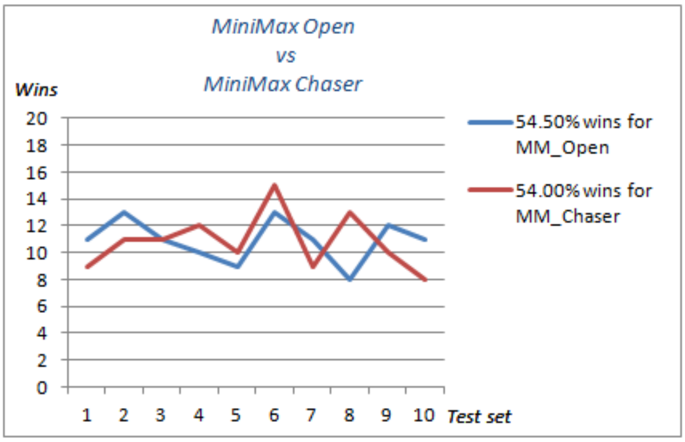
MiniMax Open Results

This test set is designed to evaluate our agent strength (in red) under the following conditions:

- Search algorithm: *MiniMax*
- Opponent heuristic evaluation function: *Open*
- Our heuristic evaluation functions:
    - *Easy score*
    - *Offensive score*
    - *Chaser score*





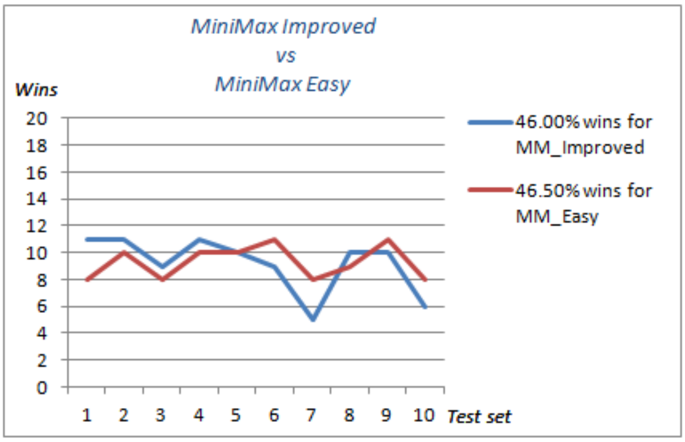


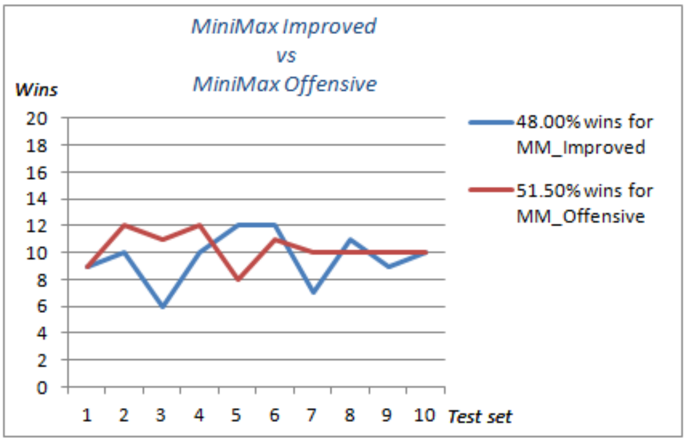
Based on the graphics above, it can be concluded that our offensive score heuristic outperforms the others.

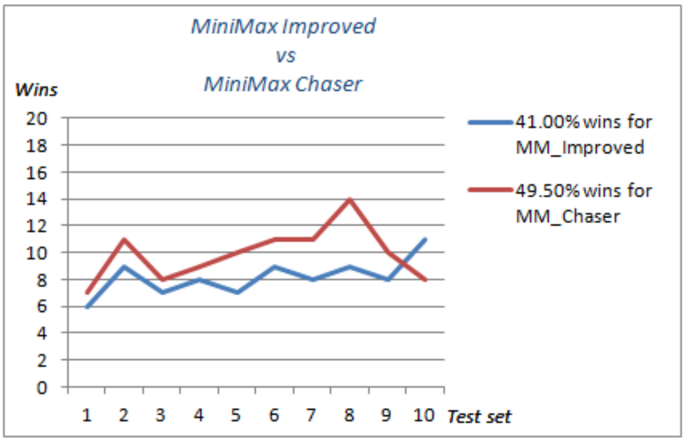
MiniMax Improved Results

This test set is designed to evaluate our agent strength (in red) under the following conditions:

- Search algorithm: *MiniMax*
- Opponent heuristic evaluation function: *Improved*
- Our heuristic evaluation functions:
    - *Easy score*
    - *Offensive score*
    - *Chaser score*





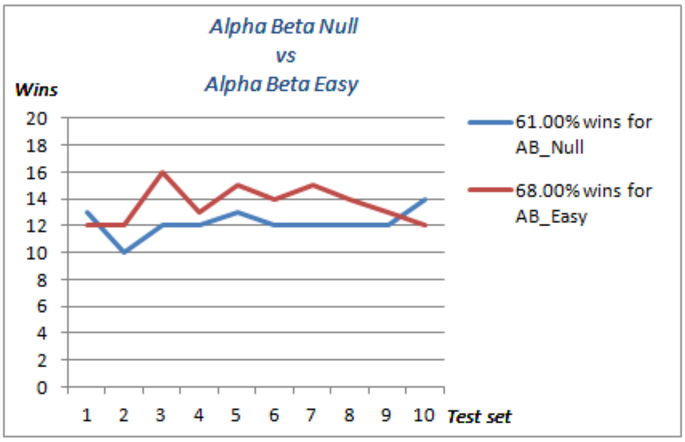


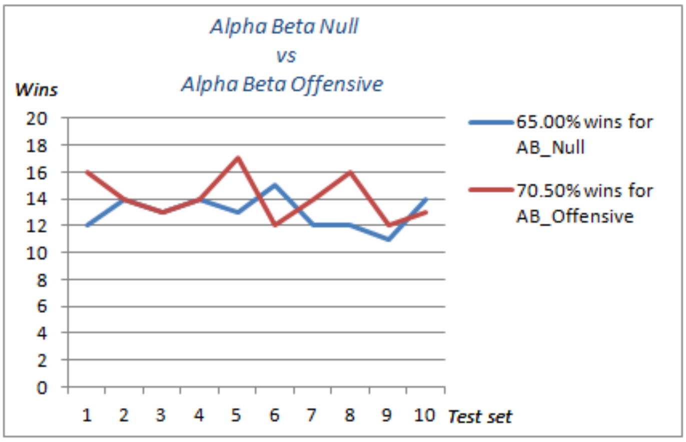
Based on the graphics above, it can be concluded that our chaser score heuristic outperforms the others.

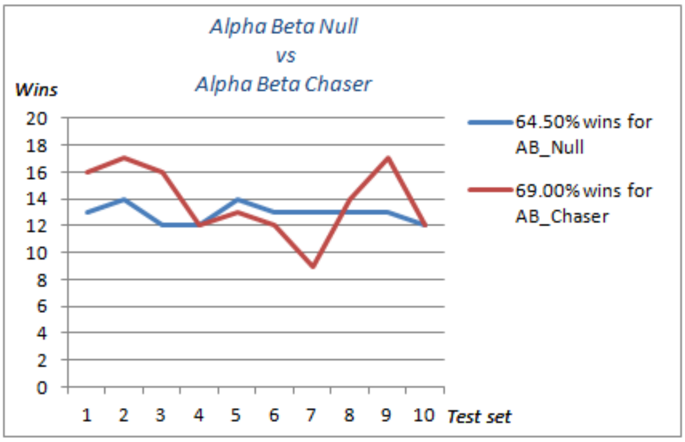
Alpha Beta Null Results

This test set is designed to evaluate our agent strength (in red) under the following conditions:

- Search algorithm: *Alpha Beta*
- Opponent heuristic evaluation function: *Null*
- Our heuristic evaluation functions:
  - *Easy score*
  - *Offensive score*
  - *Chaser score*







Based on the graphics above, it can be concluded that:

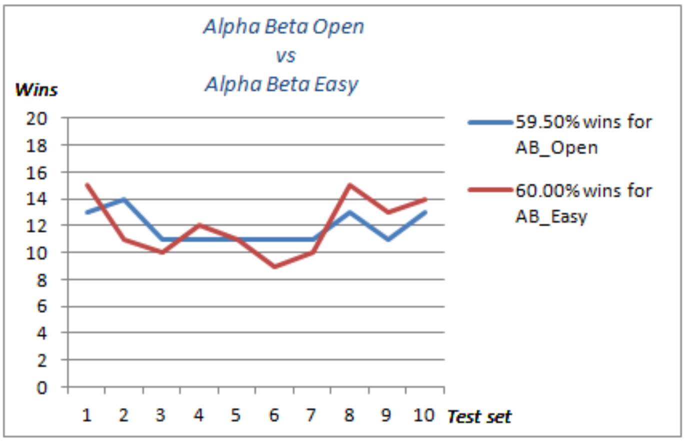
- All our heuristic functions outperform the *Null* heuristic.
- None of our heuristic functions stand out more than others.

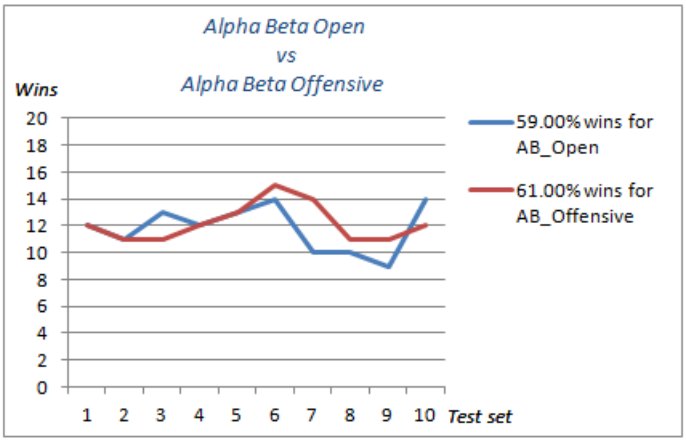
.

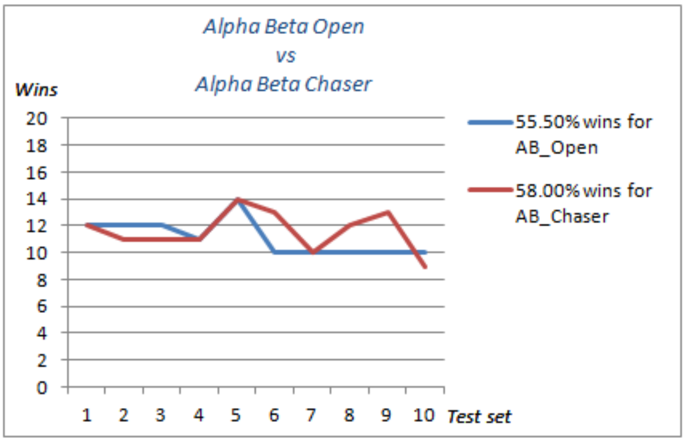
Alpha Beta Open Results

This test set is designed to evaluate our agent strength (in red) under the following conditions:

- Search algorithm: *Alpha Beta*
- Opponent heuristic evaluation function: *Open*
- Our heuristic evaluation functions:
    - *Easy score*
    - *Offensive score*
    - *Chaser score*







Based on the graphics above, it can be concluded that:

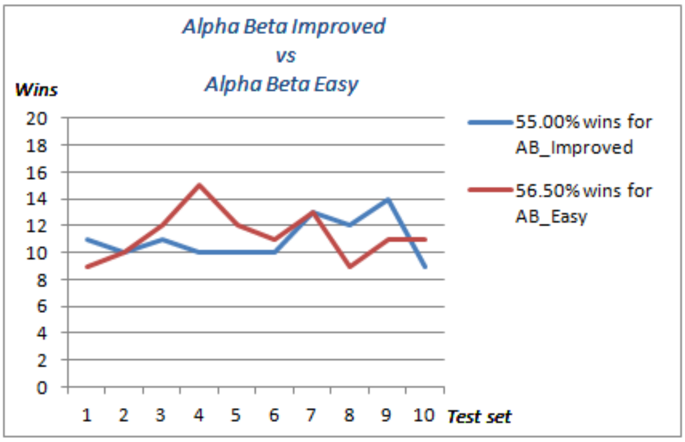
- All our heuristic functions outperform the *Open* heuristic.
- *Offensive score* slightly improves *easy score*, and both are improved by *chaser score*.

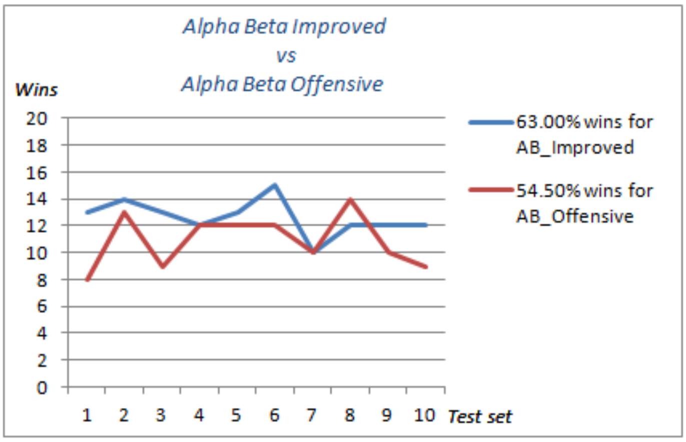
.

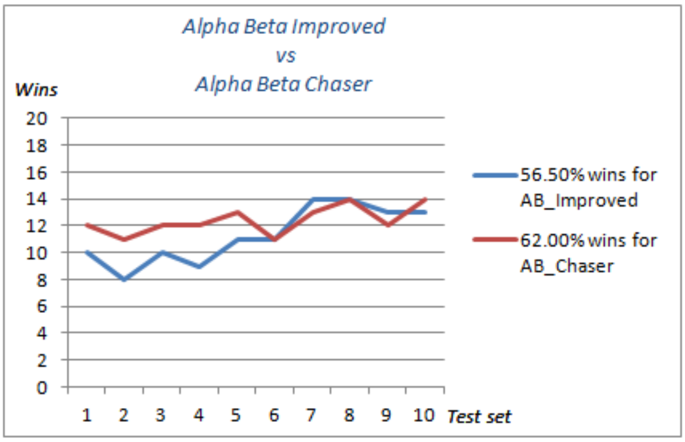
Alpha Beta Improved Results

This test set is designed to evaluate our agent strength (in red) under the following conditions:

- Search algorithm: *Alpha Beta*
- Opponent heuristic evaluation function: *Improved*
- Our heuristic evaluation functions:
    - *Easy score*
    - *Offensive score*
    - *Chaser score*







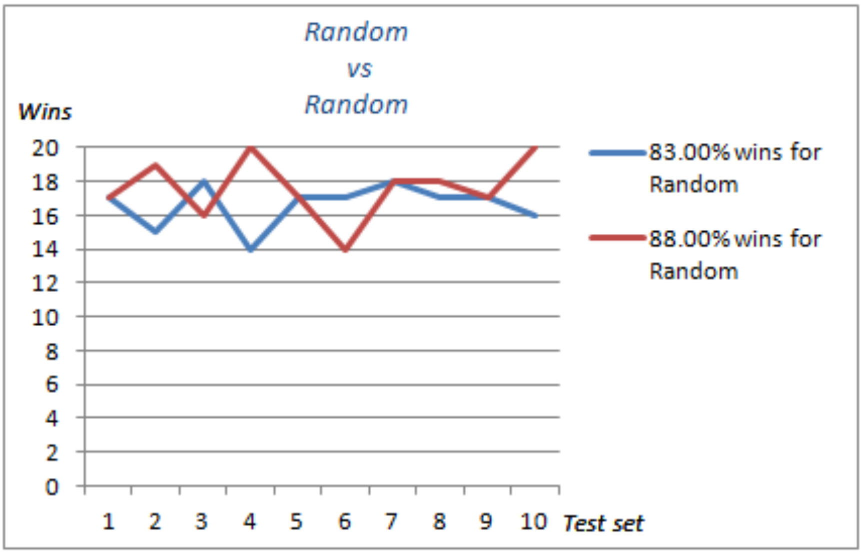
Based on the graphics above, it can be concluded that:

- *Improved* heuristic outperforms our *Offensive score* heuristic. :-(
- *Chaser score* heuristic outperforms the *Improved* heuristic. :-)

.

Random Play Results

This test set is designed to evaluate our agent strength (in red) when both agents play random moves.



It is worth telling the high percentage of success for both agents.

The Conclusion

Testing suggests that the *chaser score* heuristic is the most solid strategy. In fact, it can be observed that it outperforms its rivals in any scenario but *MiniMax Open* (54.50% vs 54.00%).

Offensive score also performs quite well in most scenarios but *Alpha Beta Improved*. Unfortunately for it, *Alpha Beta Improved* is the rival to beat...

In terms of search algorithms, *Alpha Beta with Pruning* is our choice due to the fact that it not only provides the same results than *MiniMax* does, but also performs much better (by reducing the number of nodes to be explored).

Therefore, we may conclude with...

…and the winner is Alpha Beta Pruning with Chaser Score!!!

I sincerely hope you enjoyed this reading as much as I did writing it for you.

Miguel Ángel