

# Implementación de una biblioteca para resolución de juegos con información perfecta e imperfecta

Pedro Luis Soto Santos

Universidad de Sevilla  
*pepoluis712@gmail.com*

Presentación TFG  
22 de Junio de 2023

# Índice de la Presentación

- 1 Introducción
- 2 Marco Teórico
  - Juegos de Información Perfecta
  - Juegos de Información Imperfecta
- 3 Software Relacionado
- 4 Biblioteca *pyplAI*
- 5 Casos de Estudio
  - Minimax y MCTS
  - SO-ISMCTS
  - MO-ISMCTS
  - Algoritmo Genético
- 6 Experimentación y Pruebas
- 7 Resultados
- 8 Conclusiones y Trabajo Futuro

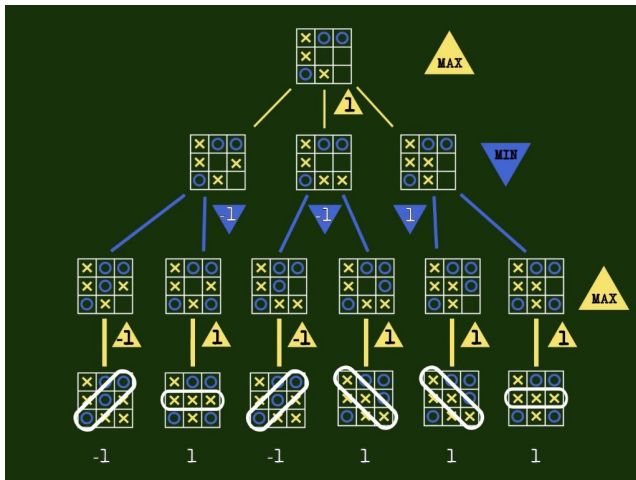
## Teoría de Juegos

- Juegos de Información Perfecta
- Juegos de Información Imperfecta
  - Estado
  - Movimiento

## Objetivos

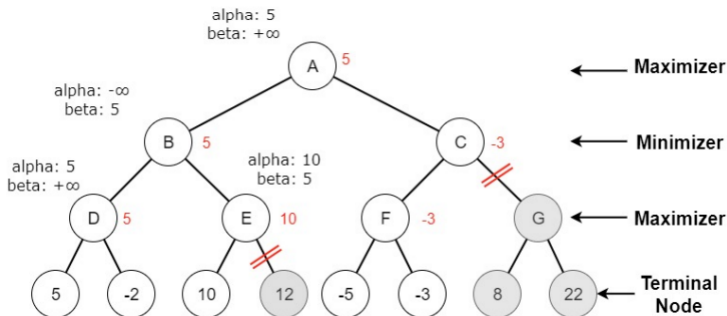
- Biblioteca pública para Python
- Manual de uso
- Desarrollo de juegos de mesa
- Análisis estadístico del rendimiento

## Minimax

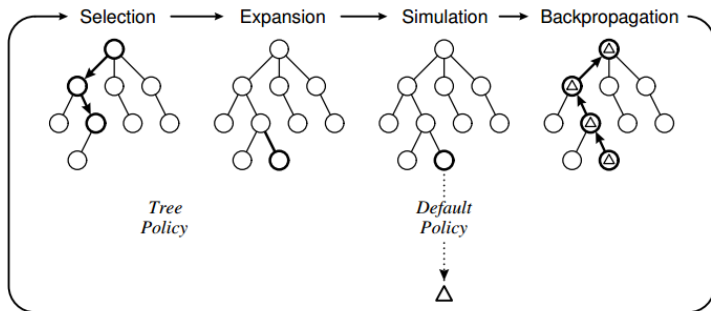


## Minimax

- Heurística
- Poda Alfa-Beta

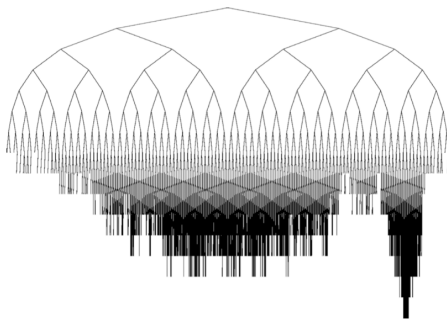


## MCTS



## MCTS

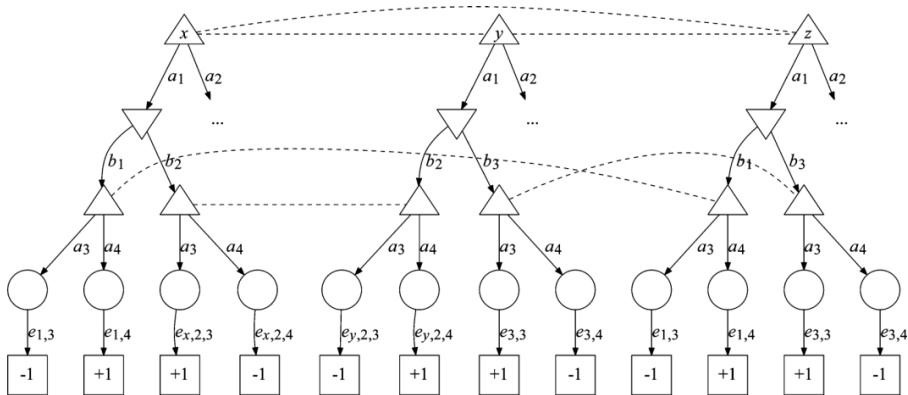
$$UCT(s) = \frac{r(s)}{n(s)} + K \sqrt{\frac{2 \ln n(s_0)}{n(s)}}$$





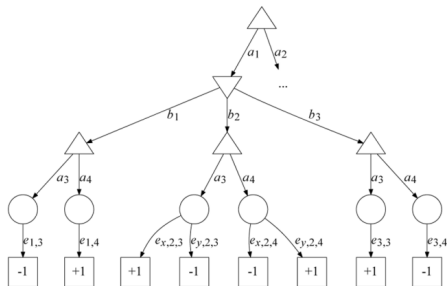
- Conjunto de Información
- Movimientos Parcialmente Observables

## PIMC

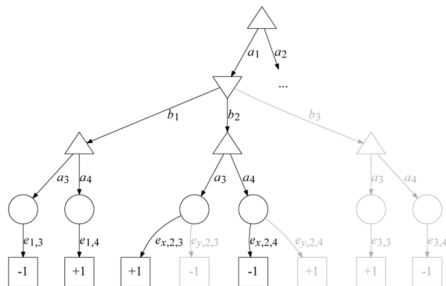


## SO-ISMCTS

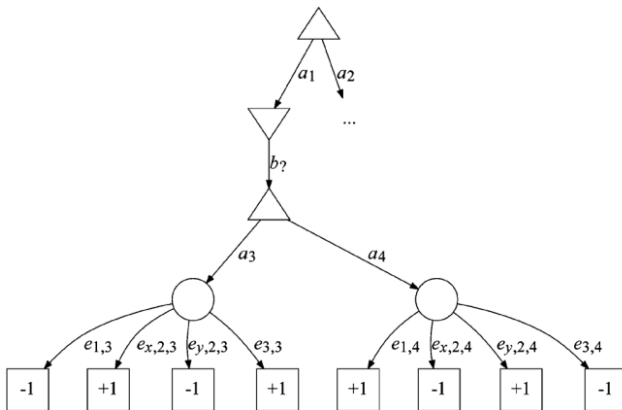
(a)



(b)

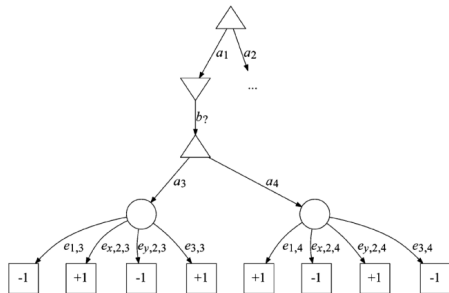


## SO-ISMCTS+POM

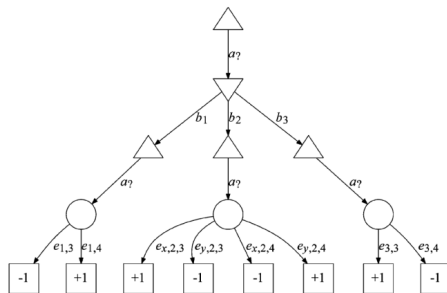


## MO-ISMCTS

(a)



(b)



Se buscaba:


- Biblioteca genérica
- Bien documentada

Se encontró:

- Bibliotecas poco documentadas (*mcts* y *mctspy*)
- Implementaciones aisladas

- *pyplAI*
- Única
- Python
- Eficiencia y simplicidad

- Manual de uso



Buscar proyectos

Ayuda Patrocinadores Acceder Registrarse

## pyplAI 1.0.3

`pip install pyplAI`

✓ Versión más reciente

Publicación: 27 may 2023

Biblioteca para Python con algoritmos de resolución de juegos de mesa (minimax, MCTS, SO-ISMCTS y MO-ISMCTS)

## Navegación

Descripción de proyecto

Histórico de versiones

Archivos de descarga

## Enlaces del proyecto

Homepage

## Estadísticas

Estadísticas de GitHub:

★ Estrellas: 0

🔗 Bifurcaciones: 0

🔓 Open issues: 0

## Descripción de proyecto

### pyplAI

#### Introducción


Esta biblioteca para *Python* es el resultado final de un Trabajo de Fin de Grado centrado en la implementación de algoritmos de *Monte Carlo Tree Search (MCTS)* y *minimax* para diferentes tipos de juegos de mesa, entre ellos, según clasifica la teoría de juegos, los juegos de información perfecta, como lo son el *ajedrez* o las *damas*, en los que en todo momento cualquier jugador puede ver toda la información del juego, como los posibles movimientos del rival, y juegos de información imperfecta, como lo son la mayoría de juegos de cartas como por ejemplo el *Uno* o la *brisca*, en el que los jugadores solo conocen información propia, como sus cartas, o información general del juego, como el estado de la mesa, pero desconocen la información del rival, como las cartas de los demás jugadores.


#### Juegos de Información Perfecta

Para este tipo de juegos se han implementado los algoritmos de *MCTS* con *UCT* [1] y *minimax* con la técnica de *poda de alfa-beta* [2]. Además, como ejemplos para ver el correcto uso de la biblioteca, se han creado diferentes juegos de mesa, entre ellos, el *Tic-Tac-Toe* (3 en raya), el *Ultimate Tic-Tac-Toe* [3] y las *damas*.




- Publicada en pypi y GitHub



Buscar proyectos 

[Ayuda](#) [Patrocinadores](#) [Acceder](#) [Registrarse](#)

## pyplAI 1.0.3

`pip install pyplAI` 

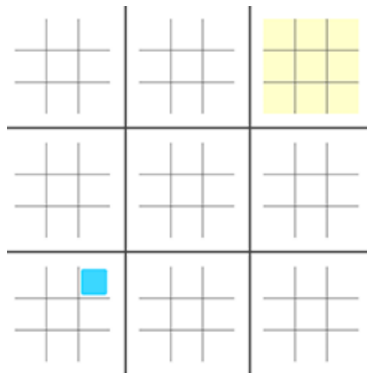
✓ [Versión más reciente](#)

Publicación: 27 may 2023

Biblioteca para Python con algoritmos de resolución de juegos de mesa (minimax, MCTS, SO-ISMCTS y MO-ISMCTS)

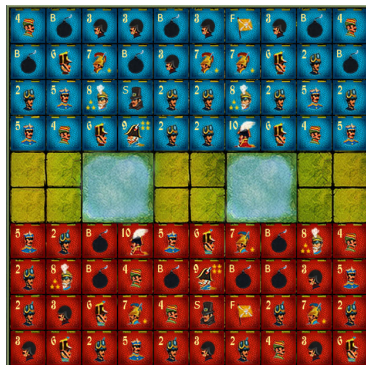
## Minimax y MCTS

- Tic-Tac-Toe
- Ultimate Tic-Tac-Toe
- Damas



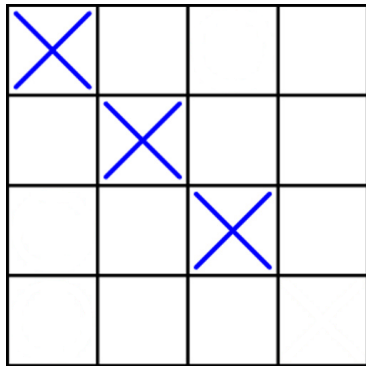
## SO-ISMCTS

- Blackjack
- Escoba
- Stratego



## MO-ISMCTS

- Phantom (4, 4, 4)
- Holjjak



## Algoritmo Genético Configuración Inicial Stratego

1	2	2	2	2	2	3	3	3	3
4	4	5	5	6	6	6	7	7	7
8	8	9	10	B	B	B	B	B	F



Cromosoma

[ 1, 2, 2, 2, 2, 2, 3, 3, 3, 3, 4, 4, 5, 5, 6, 6, 6, 7, 7, 7, 8, 8, 9, 10, 12, 12, 12, 12, 12, 11 ]

## Algoritmo Genético Configuración Inicial Stratego

- Evaluación por simulaciones
- Selección por ruleta
- Mutación por intercambio

## Algoritmo Genético Configuración Inicial Stratego

- Cruce basado en orden con tabla de frecuencia

### Paso 1

2	1	2	3	3	5	1	4	4	5
---	---	---	---	---	---	---	---	---	---

4	3	5	5	2	1	3	2	1	4
---	---	---	---	---	---	---	---	---	---



			3	3	5	1			
--	--	--	---	---	---	---	--	--	--

Tabla de Frecuencias	
Número	Frecuencia
1	2
2	2
3	2
4	2
5	2

### Paso 2

2	1	2	3	3	5	1	4	4	5
---	---	---	---	---	---	---	---	---	---

4	3	5	5	2	1	3	2	1	4
---	---	---	---	---	---	---	---	---	---

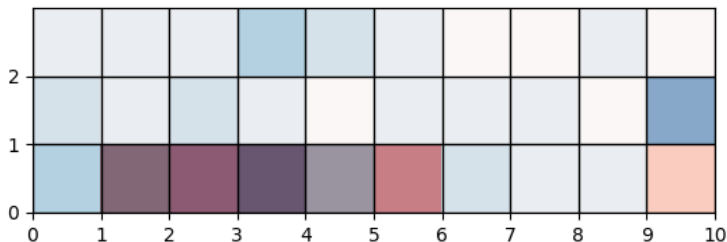


4	5	2	3	3	5	1	2	1	4
---	---	---	---	---	---	---	---	---	---

- Minimax vs MCTS
- Tiempo medio computación minimax
- SO-ISMCTS y MO-ISMCTS vs Agente Aleatorio
- SO-ISMCTS vs Croupier (Blackjack)
- MO-ISMCTS vs SO-ISMCTS



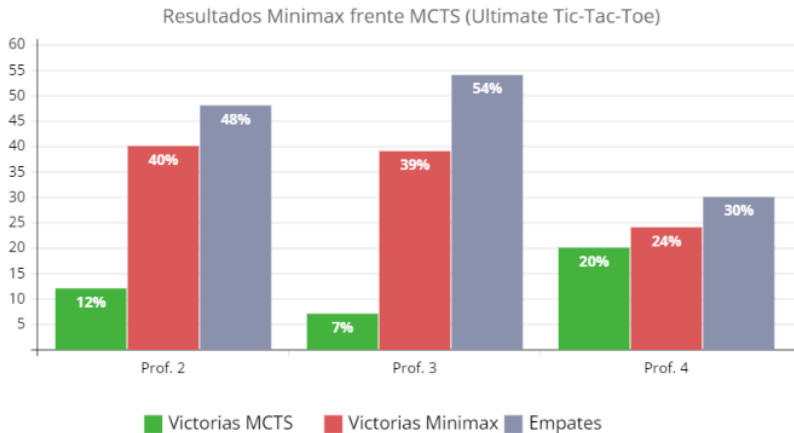
- Mapa de calor configuraciones iniciales AG



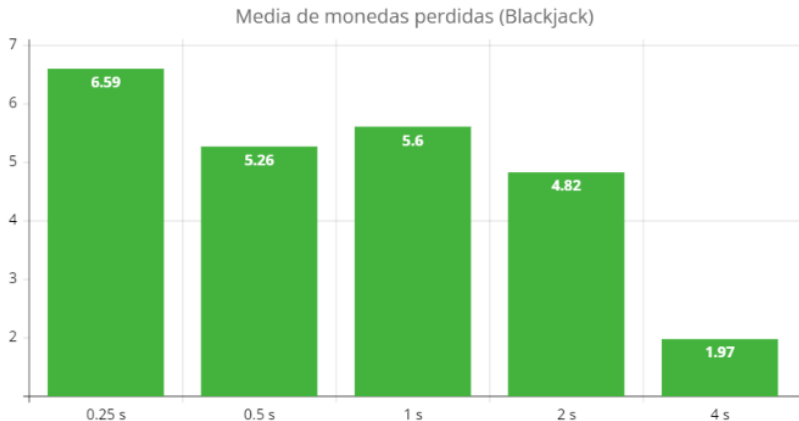
- SO-ISMCTS+AG vs SO-ISMCTS+Configuraciones Aleatorias

# Resultados

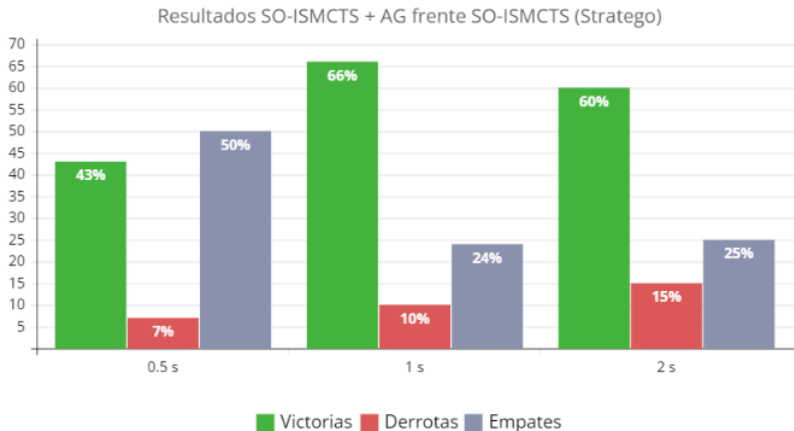
- Importancia de la función heurística en minimax



- Buen rendimiento del SO-ISMCTS

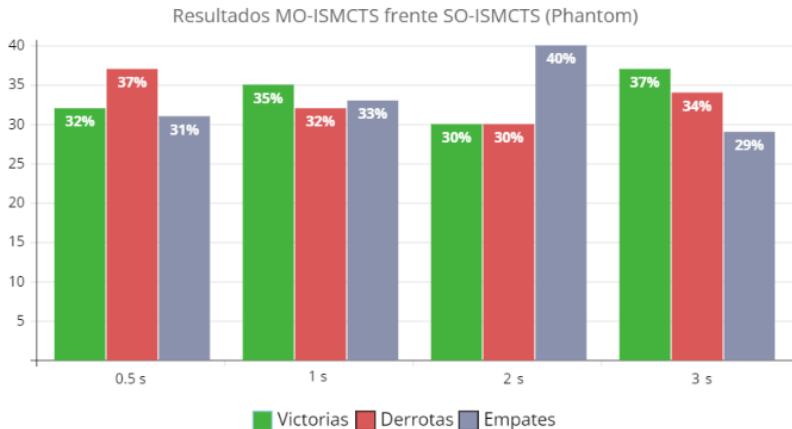


- Calidad de las configuraciones iniciales del algoritmo genético



# Resultados

- Ligera superioridad de MO-ISMCTS sobre SO-ISMCTS



# Conclusiones y Trabajo Futuro

- Biblioteca genérica, bien documentada y accesible
  - Completa batería de juegos
  - Extenso análisis de rendimiento
- 
- Ampliar variedad de algoritmos
  - Dar visibilidad a la biblioteca
  - Adaptar la biblioteca a la resolución de otros problemas de la Teoría de Juegos o incluso robótica