Para resolver el problema del Sudoku, se evaluaron tres enfoques algorítmicos avanzados: programación dinámica, divide y vencerás y algoritmos voraces con backtracking. Se seleccionó el enfoque de algoritmos voraces con backtracking optimizado, por las siguientes razones:

- Naturaleza del problema: El Sudoku es un problema de búsqueda combinatoria con restricciones. Requiere explorar configuraciones válidas siguiendo reglas específicas. El backtracking con heurísticas voraces permite priorizar las decisiones más prometedoras, reduciendo el espacio de búsqueda.

Técnicas descartadas:

- Programación dinámica: No es adecuada porque el Sudoku no tiene subproblemas solapados que puedan aprovecharse para almacenar resultados parciales.
- Divide y vencerás: No funciona eficientemente porque las celdas del Sudoku tienen interdependencias complejas (filas, columnas y bloques 3x3).

Tiempo de ejecución:

El tiempo medido en un entorno Python fue de 0.008937 segundos, lo que confirma que el algoritmo es eficiente para resolver tableros estándar de Sudoku.

Análisis de Complejidad Computacional

- **Complejidad Temporal (Big-O):** $O(9n)$, donde n es el número de celdas vacías.
    - En el peor caso, se prueban todas las combinaciones posibles en nnn celdas, pero las heurísticas reducen drásticamente el espacio de búsqueda en Sudokus reales.
- **Complejidad Espacial:** $O(81)$, ya que el tablero es fijo(9×9) y la profundidad del stack de recursión es limitada.

Conclusión

El enfoque de **algoritmos voraces con backtracking** fue seleccionado por su capacidad de resolver eficientemente problemas de búsqueda combinatoria con restricciones como el Sudoku. Gracias a la implementación de heurísticas (como elegir la celda más restringida), se logró reducir significativamente el espacio de búsqueda, obteniendo soluciones rápidas y precisas.

Este método demostró ser práctico y eficiente, resolviendo el Sudoku en milisegundos, con una complejidad computacional manejable. A diferencia de otras técnicas como programación dinámica o divide y vencerás, el enfoque seleccionado se adapta mejor a las interdependencias complejas entre las celdas, maximizando el rendimiento y la simplicidad de implementación.