SIA
Trabajo Práctico Especial 1: 0hn0

Francis Gilly 54053
Lucas Kania 54257
Emilio Tylson 54022

Reglas del juego

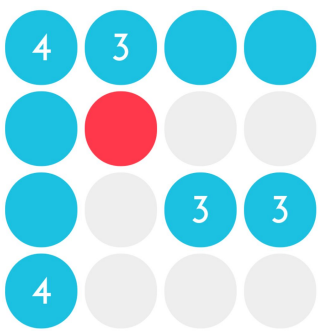
- Tokens:

Token floor

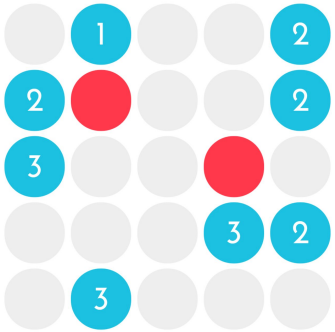
Token label

Token wall

- Disposición de los Tokens:

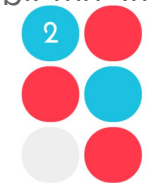


- Tablero Inicial:



Descripción de la búsqueda

- Se dividió el problema en dos subproblemas:
  - Primer subproblema:
    - Estado inicial, casilleros vacíos llenos de token floor.
    - Encontrar satisfacción de todos los tokens label.
    - Las reglas consisten en aplicar un token wall solo en los espacios alcanzables por los token label.
    - Las reglas dejan espacios ciegos.
  - Segundo subproblema:
    - Estado inicial es el estado *goal* del subproblema anterior.
    - Reemplazar token floor aislados por token wall.
    - En algoritmos informados cambia la heurística.
    - Las reglas que tiene es un conjunto más acotado y tiene en cuenta los espacios ciegos ignorados en el primer subproblema. Implica un branching factor menor.



Algoritmo no informados

BFS

- Se toman los nodos ordenados de menor a mayor profundidad.

DFS

- Se toman los nodos ordenados de mayor a menor profundidad

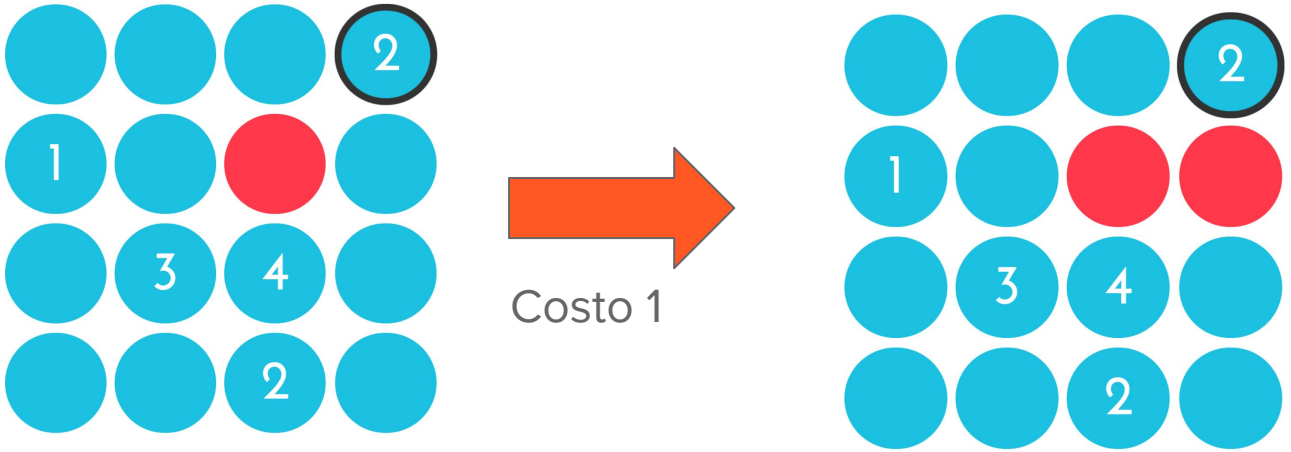
IDDFS

- Se realiza DFS con límite de profundidad, y se repite incrementando el límite hasta alcanzar la solución.

Algoritmos informados

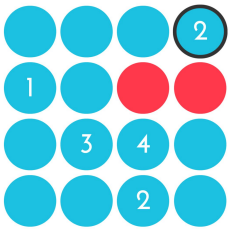
Función de costo

- Consiste en determinar la cantidad de wall que se aplica al expandir un nodo. Como se inserta de a un token wall, el costo de un estado a otro es siempre 1.



Heurística no admisible: H1

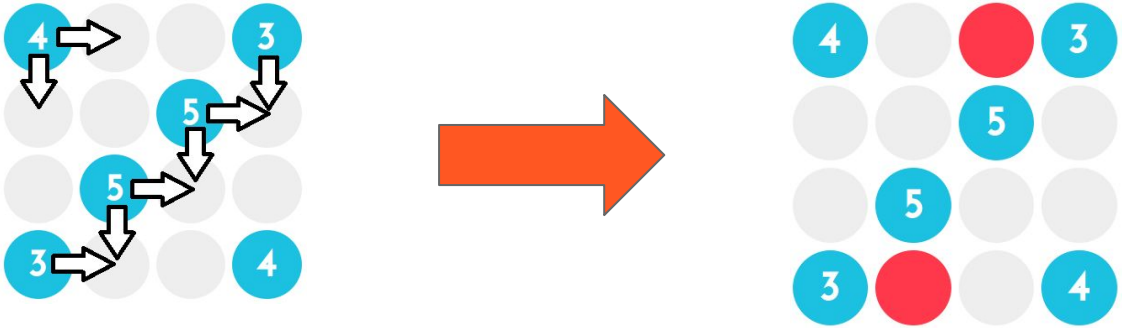
- Determinar cuán bueno es un tablero en función a los *tokens* que poseen más vecinos de lo que dicta su *label*.
- Función weight aplicada a cada token label::
  - weight(token) = vecinos - label
- H1 consiste en la sumatoria de todos los weight.
  - Excepcion si un weight da negativo, no tiene sentido seguir expandiendo. H1 emite un valor grande para ser colocado a lo último en los nodos frontera. A partir de la heurística evita que se analicen tableros que no tienen sentido seguir expandiendo.



H1(tablero) = 1 + 3 + 3 + 0 + 2 = 9

Heurística admisible: H2

- Objetivo: Estimar cantidad mínima de *walls* que faltan para lograr la solución.
- Generar un conjunto de pares *tokens* con *label* - dirección.
- A partir de cada *token* del conjunto, eliminar los *tokens* que pueda ver en la dirección en que lo ve.
- Tomar el tamaño del conjunto y dividirlo por dos (la cantidad de direcciones)
- Dividir el resultado por dos porque con un *wall* se pueden satisfacer dos *tokens* con *label*, aún cuando no se ven entre sí.



Heurística para el segundo subproblema

- La heurística consiste en contar token floor aislados.
- Heuristica admisible. Particularmente es H* con respecto a la función de costo.

A*

- La función f consta de la suma entre el costo y H2, heurística admisible.
- Encuentra solución óptima en cuanto costo, a pesar de que no es el objetivo del juego.
- Se obtuvo solución en casi todo los tableros, a excepción de que algunos le falta memoria para que termine.

Greedy

- Se utiliza H1 no admisible, pues define qué tan bueno es cada estado.
- Se ajusta al problema, ya que el objetivo es encontrar una solución sin tener en cuenta el camino a ésta.
- Hay tableros donde la solución está a gran profundidad y no alcanza la memoria.

Resultados

BFS - DFS - IDDFS

- BFS y DFS obtienen los mismos resultados. Resuelven tableros de hasta 5x5 y la máxima profundidad de la solución es 5.
- La performance de BFS y DFS depende del tablero pero no del tamaño ni de la profundidad de la solución
- IDDFS logra resolver más tableros que BFS y DFS, hasta tamaño 6x6 y la máxima profundidad a la que llega es 7.

A* vs Greedy

- Greedy más eficiente en tiempo y nodos expandidos, debido que busca solución sin importar el camino a ésta.
- A* expande más nodos debido a que la función limita la profundidad a analizar en cada iteración