Instituto Tecnológico de Costa Rica

Tarea #1

Curso: Lenguajes

Profesor: Marco Rivera Meneses

Estudiantes:
Greivin Carrillo Rodríguez
Noemí Vargas Soto
Andrés Vivallo Hurtado

Primer Semestre 2023

Link de GitHub: https://github.com/Noemi2002/The-Knight-s-Tour.git

Descripción detallada de los algoritmos:

- **Backtracking:** Es una técnica algorítmica que utiliza la recursividad, funciona enumerando las alternativas que existen en ese momento, prueba una y guarda en memoria el resto, en caso no encontrar la solución da marcha atrás y prueba otra alternativa. (Ciberaula, 2021)
- **Warnsdorff:** Es un algoritmo heurístico para resolver el problema del caballo. Las son que se puede iniciar desde cualquier posición y se mueve "a una casilla adyacente no visitada con un grado mínimo (número mínimo de adyacentes no visitados)". (Barcelona Geeks, 2022)
- **DFS:** "Es un algoritmo de búsqueda que recorre los nodos de un grafo. Consiste en ir expandiendo cada uno de los nodos que va localizando, de forma recurrente. Cuando ya no quedan más nodos que visitar en dicho camino, regresa al nodo predecesor, y repite el proceso con los vecinos". (encora, 2020)

Descripción de las funciones implementadas:

- **Función PDC-Sol:** Esta función recibe una posición inicial en forma de par ordenado '(0 . 0) o cualquier otra posición y el tamaño del tablero, un número del 5 al 16.
  Dentro de ella se definen funciones como:
  - valid-position? que recibe una posición como *x y* y un camino en forma de lista con pares ordenados:
    (valid-position? 3 3 '((0 . 0) (2 . 1) (4 . 2) (3 . 4) (1 . 5))), se encarga de validar si el caballo puede moverse en esa dirección.
  - next-moves? que indica cuáles son los siguientes movimientos del caballo dependiendo de la posición dada, recibe los mismos parámetros que valid-position?.
  - warnsdorff que se encarga de encontrar el recorrido del caballo y recibe los mismos parámetros que las funciones anteriores.
- **Función PDC-Todas:** Esta función devuelve 5 soluciones para una misma posición y un tamaño de tablero dado, recibe los mismos parámetros que PDC-Sol, dentro de esta función se definen otras funciones como:
  - PDC-Sol-Wrapper que obtiene una solución individual, recibe el tamaño del tablero y una posición inicial.
  - generate-solutions esta función de encarga de buscar todas las soluciones para la misma posición, recibe como parámetros un contador para obtener las soluciones necesarias, la posición inicial y una lista.
  - unique-solutions esta función se encarga de eliminar las soluciones repetidas y de devolver solamente 5 soluciones únicas, recibe una lista con las soluciones encontradas.

Ambas funciones devuelven una estructura de lista con pares ordenados, PDC-Sol devuelve solo 1 y PDC-Todas devuelve 5 listas con el siguiente estilo:

'((0 . 0) (2 . 1) (0 . 2) (1 . 0) (2 . 2) (0 . 1) (2 . 0) (1 . 2))

- **Función PDC-Test:** Se encarga de verificar si una matriz es una solución al problema del caballo, recibe un número que es el tamaño del tablero y una lista-matriz de pares ordenados como la siguiente:
  ((0 . 0) (2 . 1) (1 . 3)
  (2 . 3) (0 . 2) (1 . 0)
  (0 . 1) (2 . 0) (1 . 2))

Problemas sin solución:

Para todos los problemas se encontró una solución.

Problemas encontrados:

A veces cuando se modificaban las funciones, estas no devolvían el tipo de dato que debían.

Conclusiones:

En resumen, este trabajo se implementó el problema del caballo en Racket utilizando algoritmos conocidos como backtracking, DFS y heurísticas como Warnsdorff que es un algoritmo utilizado para este problema. Además de implementar funciones auxiliares para validar los movimientos y conseguir soluciones completas.

En el proceso se presentaron complicaciones con el código, más que todo con las modificaciones ya que a veces las funciones no terminaban cumpliendo con los requisitos, pero en general para todos los problemas encontró su solución.

Recomendaciones:

- Antes de ejecutar el programa, asegúrese de que su versión de Racket esté actualizada.

- Si encuentra algún problema con el programa, consulte la sección "Issues" del repositorio de GitHub para posibles soluciones o informe el problema al equipo de soporte técnico.

- Para obtener más información sobre el problema del recorrido del caballo y sus soluciones, recomendamos leer más en los recursos enumerados en la sección "Recursos adicionales" de este manual.

Bibliografía:

Barcelona Geeks. (5 de julio de 2022). *Algoritmo de Warnsdorff para el problema del recorrido de Knight*. Obtenido de https://barcelonageeks.com/algoritmo-de-warnsdorff-para-el-problema-de-la-gira-de-knight/

Ciberaula. (2021). *Backtraking*. Obtenido de https://www.ciberaula.com/cursos/java/backtracking.php

encora. (25 de mayo de 2020). *DFS vs BFS*. Obtenido de https://www.encora.com/es/blog/dfs-vs-bfs

Racket. (s.f.). *Racket Documentation*. Obtenido de https://docs.racket-lang.org/

Bitácora:

ANDRÉS

- 07 – 05 – 2023
  - Reunión con el grupo para leer la descripción de la tarea y repartir la tarea individualmente. Acordamos en trabajar en la solución del algoritmo hasta el siguiente martes.
  - Investigué acerca del algoritmo que soluciona el problema, chat GPT me lo resolvió, lo copié y pequé en el archivo solucion.rkt
- 14 – 05 – 2023
  - Reunión para ver los avances de nuestro trabajo, además de hacer un archivo de gui base en src/gui/board.rkt. Organizamos la estructura del proyecto base y separamos las tareas de las cosas faltantes de acuerdo con los requerimientos del proyecto
- 17 – 05- 2023
  - Trabajé en la interfaz dibujando un tablero de ajedrez utilizando la librería Racket chess.
- 18 – 05- 2023
  - Trabajé en la interfaz, ahora haciendo que cambiara de tamaño y en la conexión con las funciones que solucionan el problema del caballo.
- 19 – 05- 2023
  - Trabajé en aspectos generales del proyecto que hacían falta.

NOEMÍ

- 07 – 05 – 2023
  - Se hizo una reunión con el grupo de trabajo para leer el documento y dar ideas de como trabajar.
  - Se acordó montar un algoritmo de solución para el martes 9/05.
  - Busqué un seudocódigo de una solución del problema.
- 08 – 05 – 2023
  - Investigué el cómo se ve una solución del problema gráficamente en un video de YouTube.
  - Empecé a montar la solución en código de Racket.
- 09 – 05 – 2023
  - La solución estaba completa pero el código mandaba errores que no pude solucionar.
- 10 – 05 – 2023
  - Le pregunté al profe al respecto de los errores que me mandaba, quedé en pasarle el código comentado para que me ayudara. (Por cosas de la vida no se lo pude mandar).

- 11 – 05 – 2023 / 12 – 05 – 2023
    - El código tenía una función principal y dentro de esta todas las funciones, lo reestructuré para que cada función esté por fuera (que sean independientes entre sí y se llamen con los parámetros necesarios). El código ya funciona, recibe el tamaño del tablero y un par ordenado con la posición inicial, devuelve todos los pares ordenados del camino que recorrió. Se implementó Backtracking. Se vuelve ineficiente para tableros mayores o iguales a 12x12.
- 14 – 05 – 2023
    - Realizamos una reunión para ver los avances de cada uno con respecto al algoritmo y demás información que habíamos encontrado, decidimos dividir lo que nos hace falta, yo decidí enforcarme en mejorar el algoritmo que ya tenemos y trabajar en paralelo en la documentación.
- 15 – 05 – 2023
    - Arreglé la función PDC-Sol para términos del curso y las comenté, empecé a investigar cómo obtener todas las soluciones para un mismo tablero, encontré código útil que devolvía solamente 2 soluciones para un tablero 3x3, estuve tratando de poder visualizar más soluciones.
- 17 – 05 – 2023
    - Estuve trabajando en la función que verifica si una matriz es solución del tablero, pero aún manda errores.
- 18 – 05 – 2023 / 19 – 05 – 2023
    - Trabajé en PDC-Todas y PDC-Test, monté unas funciones, pero no funcionaban al 100%. Me enfoqué solo en PDC-todas y al final quedó funcionando bien.
    - PDC-test ya devuelve un resultado, pero no hace las validaciones adecuadas.
    - Trabajé en aspectos generales del proyecto y la documentación.

GREIVIN

- 07 – 05 – 2023
    - Se hizo una reunión para leer lo solicitado, dar ideas de como trabajar.
    - Se acordó tener la respuesta a la función PDC-Sol para el martes 09.
- 09 – 05 – 2023
    - Investigué acerca del algoritmo hamiltoniano para la solución del problema.
    - Codifique una solución, pero utilizando todas las funciones disponibles de Racket para entender mejor el problema.
- 11 – 05 – 2023
    - Investigue un poco sobre la librería Racket Graphical Interface.
    - Intente montar un código de prueba para la interfaz gráfica.
- 13 – 05 – 2023
    - Monté una interfaz y comencé a investigar cómo hacer la función de pintar las casillas.

- 14 – 05- 2023
  - Monté una función para pintar las casillas de la interfaz.
- 16 – 05- 2023
  - Trabajé en el manual de usuario.
- 17 – 05- 2023
  - Trabajé en la documentación.
- 18 – 05- 2023 / 19 – 05- 2023
  - Arreglé la función de PDC-Test.