Aplicación de algoritmos metaheurísticos a un taxi Autónomo

Pau York Brinkhaus Tort
Universidad de La Laguna
La Laguna, España
alu0101525600@ull.edu.es

Karim Martin Sohan Bauer
Universidad de La Laguna
La Laguna, España
alu0101530833@ull.edu.es

Wojciech Stachyra
Universidad de La Laguna
La Laguna, España
wojciechstachyra.wow@gmail.com

***Resumen*—Este Informe mostrará la aplicación de Algoritmos Metaheurísticos al problema del taxi autónomo. Aquí se implementan los Algoritmos más comunes conocidos como (μ,λ) y (μ+λ) desde cero**

Keywords— Metaheuristics, AI, (μ,λ), (μ+λ)

I INTRODUCCIÓN

El problema al que nos enfrentamos en este proyecto es la navegación de un taxi a través de una cuadrícula llena de obstáculos desde un punto inicial hasta un punto final. Al resolver esta tarea nos centramos en utilizar algoritmos metaheurísticos.

II FORMULACIÓN DEL PROBLEMA

En este apartado se formulará el problema que queremos resolver desde el punto de vista de Espacios de Estados.

III ENTORNO DE SIMULACIÓN

Utilizamos phyton como lenguaje de programación porque es fácil de usar y accesible. no dedicamos mucho tiempo a crear una bonita presentación, sino que nos conformamos con una simple salida de línea de comandos. Esto nos permitió dedicar más tiempo a la aplicación del algoritmo. este apartado se describirá la interface y su implementación. Se incluirá capturas de pantallas que muestren las funcionalidades de la interface. También se incluirá la justificación sobre el entorno de programación escogido. En caso de que fuera necesario añadir código se añadirá como un apéndice.

IV METODOLOGÍA DE TRABAJO

En primer lugar, realizamos la elección del algoritmo.

A continuación, creamos juntos un diagrama de clases UML para dejar clara la estructura que tendrá nuestro programa. Después, Karim creó el marco con nuestro apoyo e implementó la clase de tablero con todas sus funciones. A continuación, Wojtek y York aplicaron los algoritmos en cooperación. Todos contribuyeron a la evaluación final de los algoritmos. A lo largo del proyecto, todo el mundo estuvo siempre dispuesto a ofrecer consejos.

V ALGORITMO DE BÚSQUEDA

La elección de los algoritmos recayó en el algoritmo (μ,λ) conocido por Estrategias de Evolución y el algoritmo (μ+λ)|. Ambos son algoritmos basados en la población que funcionan mediante mutación y selección. (Reference 1)

Algorithm 18 *The (μ, λ) Evolution Strategy*

$\mu \leftarrow$ number of parents selected
$\lambda \leftarrow$ number of children generated by the parents

$P \leftarrow \{\}$
for λ times **do** ▷ Build Initial Population
  $P \leftarrow P \cup \{\text{new random individual}\}$
$Best \leftarrow \square$
repeat
  for each individual $P_i \in P$ **do**
    AssessFitness(P_i)
    if $Best = \square$ or Fitness(P_i) > Fitness($Best$) **then**
      $Best \leftarrow P_i$
  $Q \leftarrow$ the μ individuals in P whose Fitness( ) are greatest ▷ Truncation Selection
  $P \leftarrow \{\}$ ▷ Join is done by just replacing P with the children
  for each individual $Q_j \in Q$ **do**
    for λ/μ times **do**
      $P \leftarrow P \cup \{\text{Mutate}(\text{Copy}(Q_j))\}$
until $Best$ is the ideal solution or we have run out of time
return $Best$

Algorithm 19 *The $(\mu + \lambda)$ Evolution Strategy*

$\mu \leftarrow$ number of parents selected
$\lambda \leftarrow$ number of children generated by the parents

$P \leftarrow \{\}$
for λ times **do** ▷ Or perhaps $\lambda + \mu$. See Footnote 18, page 34
  $P \leftarrow P \cup \{\text{new random individual}\}$
$Best \leftarrow \square$
repeat
  for each individual $P_i \in P$ **do**
    AssessFitness(P_i)
    if $Best = \square$ or Fitness(P_i) > Fitness($Best$) **then**
      $Best \leftarrow P_i$
  $Q \leftarrow$ the μ individuals in P whose Fitness( ) are greatest
  $P \leftarrow Q$ ▷ The Join operation is the only difference with (μ, λ)
  for each individual $Q_j \in Q$ **do**
    for λ/μ times **do**
      $P \leftarrow P \cup \{\text{Mutate}(\text{Copy}(Q_j))\}$
until $Best$ is the ideal solution or we have run out of time
return $Best$

VI EVALUACIÓN EXPERIMENTAL

Durante la evaluación hicimos una en un campo pequeño y otra en un campo grande. Los algoritmos tenían un tiempo determinado por carrera hasta que tenían que entregar. En el campo pequeño, ambos tuvieron un rendimiento similar, lo que probablemente se deba a la similitud de los algoritmos. Se nota que uno acaba mucho más cerca del objetivo que el otro cuando no lo encuentra.

En el campo grande, ninguno de los dos encontró un objetivo en 5 recorridos. Es interesante observar que el algoritmo (μ,λ) se comportó mejor aquí. Probablemente porque con un campo tan grande, el enfoque exploratorio del algoritmo (μ+λ) es más elitista que el enfoque de explotación del algoritmo (μ+λ). Sin embargo, cabe suponer que con más tiempo de ejecución también se encontraría una solución en un campo de juego amplio.

```
Small Evaluation

start the evolutionary Algorithms with following board and algorithm parameters:
board size: 20 20 with obstacle ratio: 0.2 and time to run 5
Algorithm muCommaLambda
population size is 500 and the parent size is 100
Algorithm muPlusLambda
population size is 500 and the parent size is 10
Parameters for the creation of both Populations: mean = 50 ,std = 10 ,time = 1
mcl ratio for comma 0.65 , average solution length: 53.53846153846154
and a average distance missing to the solution of  161.85714285714286
mcl ratio for  plus 0.75 , average solution length: 54.93333333333333
and a average distance missing to the solution of  58.0
the small took : 200.53594970703125 seconds


big evaluation

start the evolutionary Algorithms with following board and algorithm parameters:
board size: 50 50 with obstacle ratio: 0.2 and time to run 10
Algorithm muCommaLambda
population size is 500 and the parent size is 100
Algorithm muPlusLambda
population size is 500 and the parent size is 10
Parameters for the creation of both Populations: mean = 60 ,std = 30 ,time = 1
mcl ratio for comma 0.0 , average solution length: 0
and a average distance missing to the solution of  258.4
mcl ratio for  plus 0.0 , average solution length: 0
and a average distance missing to the solution of  318.8
the small took : 600.1127190589905 seconds
```

VII CONCLUSIONES

En conclusión, se puede decir que los algoritmos metaheurísticos pueden ser útiles en la búsqueda de caminos y no pocas veces encuentran el camino. Pero, por supuesto, no se pueden comparar con los algoritmos desarrollados específicamente para este problema.

REFERENCES

1 *Sean Luke, Essentials of Metaheuristics, second, 2013, Lulu, Available for free at http://cs.gmu.edu/$\ sim$sean/book/metaheuristics/*