Proyecto 2 - Búsquedas Locales

[Yeray Martínez Martínez](mailto:yeray.martinez@opendeusto.es)

[Adrián Martínez Tejerina](mailto:adrian.martinez.t@opendeusto.es)

## Análisis de resultados

Análisis de resultados después de ejecutar cada algoritmo 10 veces.

### Hill Climbing Search

El algoritmo Hill Climbing básico. Se puede usar este como referencia para el análisis de sus variantes.

|  | **Evaluation** | **Runtime (secs)** |
| --- | --- | --- |
| **Promedio** | 54,5 | 0,455 |
| **Mínimo** | 52 | 0,43 |
| **Desviación estándar** | 1,269295518 | 0,04552166761 |

### Stochastic Hill Climbing

Los resultados del Hill Climbing estocástico son peores posiblemente debido a la propia implementación del problema Bin-Packing, ya que la inicialización del problema no genera un estado inicial aleatorio y por lo tanto se ve más afectado ya que el cambio de consistencia frente a posibilidad de llegar a un máximo global no es rentable en este caso.

|  | **Evaluation** | **Runtime (secs)** |
| --- | --- | --- |
| **Promedio** | 76,3 | 0,509 |
| **Mínimo** | 71 | 0,45 |
| **Desviación estándar** | 3,267686916 | 0,04909175083 |

### Random Restart Hill Climbing

Los resultados del Hill Climbing con reinicio aleatorio son muy parecidos al básico ya que, en este caso, el algoritmo se basa en el propio Hill Climbing pero ejecutándose varias veces. Como hemos dicho antes, la ejecución no es aleatoria por lo que genera nodos sucesores parecidos.

#### 5 reinicios

|  | **Evaluation** | **Runtime (secs)** |
| --- | --- | --- |
| **Promedio** | 54,7 | 0,512 |
| **Mínimo** | 53 | 0,45 |
| **Desviación estándar** | 1,251665557 | 0,07728734265 |

#### 15 reinicios

|  | **Evaluation** | **Runtime (secs)** |
| --- | --- | --- |
| **Promedio** | 55,1 | 0,512 |
| **Mínimo** | 53 | 0,45 |
| **Desviación estándar** | 1,197219 | 0,05692099788 |

#### 30 reinicios

|  | **Evaluation** | **Runtime (secs)** |
| --- | --- | --- |
| **Promedio** | 54,3 | 0,501 |
| **Mínimo** | 52 | 0,45 |
| **Desviación estándar** | 1,251665557 | 0,06190494505 |

### Stochastic Random Restart

Este algoritmo es casi sin duda el que peor resultados ha dado, ya que se basa tanto en el estocástico como en el de Random Restart, heredando así los problemas de ambos en este caso. Debido a la implementación del problema, la parte de reinicio aleatorio de este algoritmo básicamente ejecuta lo mismo pero un número dado de iteraciones, y como hemos visto anteriormente, hacer la variante estocástica no da buenos resultados.

#### 5 reinicios

|  | **Evaluation** | **Runtime (secs)** |
| --- | --- | --- |
| **Promedio** | 76,7 | 0,495 |
| **Mínimo** | 75 | 0,43 |
| **Desviación estándar** | 1,251665557 | 0,05380004131 |

#### 15 reinicios

|  | **Evaluation** | **Runtime (secs)** |
| --- | --- | --- |
| **Promedio** | 77,8 | 0,506 |
| **Mínimo** | 75 | 0,44 |
| **Desviación estándar** | 1,813529401 | 0,0900863783 |

#### 30 reinicios

|  | **Evaluation** | **Runtime (secs)** |
| --- | --- | --- |
| **Promedio** | 77,5 | 0,471 |
| **Mínimo** | 75 | 0,45 |
| **Desviación estándar** | 1,58113883 | 0,03348299734 |

### Local Beam Search

Aunque es mejor que el Random Restart al hacer la ejecución con distintos nodos actuales en vez de eligiendo el mejor nodo en cada expansión, la ejecución es más costosa. Esto es porque metemos tanto los nodos actuales como los sucesores en una lista y seleccionamos los mejores entre ellos, llevando a un tiempo de ejecución mayor pero con una mejor y más consistente evaluación. Podría dar incluso mejores resultados si la inicialización fuera diferente en cada beam.

#### 3 beams

|  | **Evaluation** | **Runtime (secs)** |
| --- | --- | --- |
| **Promedio** | 54,5 | 1,296 |
| **Mínimo** | 53 | 1,2 |
| **Desviación estándar** | 0,7071067812 | 0,07183313998 |

#### 5 beams

|  | **Evaluation** | **Runtime (secs)** |
| --- | --- | --- |
| **Promedio** | 54,1 | 2,093 |
| **Mínimo** | 53 | 2 |
| **Desviación estándar** | 0,994428926 | 0,1240116482 |

#### 10 beams

|  | **Evaluation** | **Runtime (secs)** |
| --- | --- | --- |
| **Promedio** | 54,2 | 4,441 |
| **Mínimo** | 53 | 4,23 |
| **Desviación estándar** | 0,91893658 | 0,1884704516 |

## 

## Conclusiones

| **Algoritmo** | **Métrica** | **Diferencia evaluación** | **Evaluation** | **Runtime (secs)** |
| --- | --- | --- | --- | --- |
| **Hill Climbing Search** | **Promedio** | 0,4 | 54,5 | 0,455 |
| **Desviación estándar** | 0,5621887364 | 1,269295518 | 0,04552166761 |
| **Stochastic Hill Climbing** | **Promedio** | 22,2 | 76,3 | 0,509 |
| **Desviación estándar** | 2,560580134 | 3,267686916 | 0,04909175083 |
| **Random Restart Hill Climbing**  **(5 reinicios)** | **Promedio** | 0,6 | 54,7 | 0,512 |
| **Desviación estándar** | 0,5445587758 | 1,251665557 | 0,05380004131 |
| **Random Restart Hill Climbing**  **(15 reinicios)** | **Promedio** | 1,0 | 55,1 | 0,512 |
| **Desviación estándar** | 0,4901122185 | 1,197219 | 0,05692099788 |
| **Random Restart Hill Climbing**  **(30 reinicios)** | **Promedio** | 0,2 | 54,3 | 0,501 |
| **Desviación estándar** | 0,5445587758 | 1,251665557 | 0,06190494505 |
| **Stochastic Random Restart**  **(5 reinicios)** | **Promedio** | 22,6 | 76,7 | 0,495 |
| **Desviación estándar** | 0,5445587758 | 1,251665557 | 0,05380004131 |
| **Stochastic Random Restart**  **(15 reinicios)** | **Promedio** | 23,7 | 77,8 | 0,506 |
| **Desviación estándar** | 1,1064226198 | 1,813529401 | 0,0900863783 |
| **Stochastic Random Restart**  **(30 reinicios)** | **Promedio** | 23,4 | 77,5 | 0,471 |
| **Desviación estándar** | 0,8740320488 | 1,58113883 | 0,03348299734 |
| **Local Beam Search**  **(3 beams)** | **Promedio** | 0,4 | 54,5 | 1,296 |
| **Desviación estándar** | 0,0 | 0,7071067812 | 0,07183313998 |
| **Local Beam Search**  **(5 beams)** | **Promedio** | 0,0 | 54,1 | 2,093 |
| **Desviación estándar** | 0,2873221448 | 0,994428926 | 0,1240116482 |
| **Local Beam Search**  **(10 beams)** | **Promedio** | 0,1 | 54,2 | 4,441 |
| **Desviación estándar** | 0,2118297988 | 0,91893658 | 0,1884704516 |

Podemos observar que el Local Beam Search produce los mejores resultados en cuanto al promedio y a la desviación estándar, sacrificando en tiempo de ejecución. En segundo lugar, el Random Restart produce resultados un poco peores y menos consistentes, pero tiene una gran mejora en tiempo de ejecución.

Por último, las variaciones estocásticas tanto del Hill Climbing como del Random Restart, son notoriamente peores en cuanto a evaluación, dando diferencias de hasta más de 20 contenedores usados. El hecho de que este tipo de algoritmos de unos resultados peores se puede atribuir al tipo de problema con el que estamos trabajando.

# Bibliografía

[1] Documentación con una gran variedad de recursos y ejemplos en forma de tutorial para el aprendizaje del lenguaje de programación de R. Disponible en: [https://www.geeksforg  
eeks.org/r-programming-language-introduction/](https://www.geeksforgeeks.org/r-programming-language-introduction/).

Hemos utilizado la extensión de GitHub Copilot integrada en R como IA generativa para ayudarnos y darnos sugerencias a la hora de hacer el código. Nos ha sido de gran ayuda frente a otros recursos ya que utiliza la información que posee del resto del código para brindarnos información más precisa y personalizada.