Inteligencia Artificial

Informe Final: Travelling Tournament Problem

Lukas Zamora

15 de diciembre de 2017

Evaluación

Evaluación

Mejoras 1ra Entrega (10 %):	_____
Código Fuente (10 %):	_____
Representación (15 %):	_____
Descripción del algoritmo (20 %):	_____
Experimentos (10 %):	_____
Resultados (10 %):	_____
Conclusiones (20 %):	_____
Bibliografía (5 %):	_____
Nota Final (100):	_____

Resumen

Los deportes profesionales han sido un gran negocio a lo largo de todos los tiempos tanto para los equipos involucrados como para la televisión, muchos investigadores se han abocado a resolver problemas relacionados a estos como por ejemplo sports scheduling problem. Travelling Tournament Problem (TTP) busca minimizar la distancia recorrida por todos los equipos en una competición. En este estudio se hará una revisión al estado del arte del problema definiendo algunos conceptos básicos para su entendimiento y proponiendo un modelo matemático para este. Para comparar cada una de las técnicas se utilizarán instancias NL compartidas en el sitio oficial del problema esto con el fin de en lo posible decir que técnica es la mejor.

1. Introducción

Travelling Tournament Problem es un problema inspirado en la MLB en dónde la principal tarea es crear el schedule de diferentes equipos satisfaciendo principalmente dos grandes restricciones: Disminuir la cantidad de viajes realizados por cada equipo y cumplir algún patrón de partidos de visita y local.

En el presente informe se analizará el estado del arte del problema *Travelling Torunament Problem* que desde ahora en adelante llamaremos TTP. Se comienza definiendo el problema de

una manera simple dando a conocer su origen, que lo hace interesante estudiar y por supuesto definiendo sus variables y restricciones.

Una vez terminado lo anterior se hará una revisión al estado del arte del problema. Analizando las aproximaciones más importantes hechas hasta la fecha, comparando las técnicas utilizados y en lo posible ver ventajas y desventajas de cada aproximación.

Luego de realizar el estado del arte se mostrará el modelo matemático que permite definir el problema. Finalmente se concluirá el trabajo mostrando las conclusiones extraídas del informe dando a conocer cuales son las mejores técnicas para la resolución del problema así como las mejores formas de modelarlo.

2. Definición del Problema

TTP es un problema relativamente moderno cuya primera descripción fue hecha por Easton, Nemhauser y Trick en [8]. El problema es inspirado en la Major League Baseball o más conocida como MLB en dónde equipos deben recorrer grandes distancias. El objetivo es buscar crear un fixture de campeonato cumpliendo dos grandes restricciones:

- Disminuir la cantidad de viajes para cada equipo.
- Cumplir algún patrón de la cantidad de partidos de local o visita que puede hacer un equipo de manera consecutiva.

Considerando lo anterior podemos decir que TTP es una combinación entre un problema de Scheduling en donde se busca satisfacer las restricciones sobre los patrones que un equipo tiene sobre sus juegos de local y visita en un campeonato. Además de TSP en donde se desea minimizar la distancia recorrida en un tour.

Antes de pasar a la definición formal del problema existen algunas definiciones que se deben conocer:

- **Round-Robin tournament**: Torneo en dónde se enfrentan los equipos todos contra todos sólo una vez. Este tipo de torneo se compone de n equipos y consta de $n-1$ fechas en dónde cada una posee $n/2$ partidos.
- **Double Round Robin Tournament**: Torneo en dónde se enfrentan los equipos todos contra todos dos veces en el mismo torneo, una en cada estadio. Este tipo de torneo se compone de n equipos y consta de $2(n-1)$ fechas en dónde cada una posee $n/2$ partidos.
- **Road Trip**: Se da cuando un equipo juega partidos consecutivos como visitante.
- **Home Stand**: Se da cuando un equipo juega partidos consecutivos como local.

Además de ciertas consideraciones:

- Dentro de un partido siempre habrá un equipo local y otro visita.
- El largo de un Road Trip y Home Stand se mide con el número de partidos jugados.
- La distancia se mide desde el estadio del local hasta el lugar de visita en un partido de visita para un equipo i. En un partido de local la distancia recorrida por un equipo es cero. En un Road Trip las distancias se cuentan desde el estadio de visita al siguiente.
- Cada equipo comienza la temporada en casa y la termina en el mismo lugar.
- Existe un intervalo para el largo de los Road Trip y Home Stand.

Tomando todo lo anterior se tiene como entrada del problema los siguientes parámetros:

- N: El número de equipos presentes en el torneo.
- D: Matríz de distancias.
- L, U: Parámetros enteros para acotar el largo de los Home Stand y Road Trip.

Una función objetivo que buscará minimizar la distancia total recorrida por cada equipo al finalizar el torneo. Consiguiendo como sálida la programación de un torneo sujeto a las siguientes restricciones duras:

- Cada equipo debe jugar contra cada otro equipo.
- En cada fecha, todos los equipos tienen que jugar.
- Cada equipo puede imponer sus propias restricciones.

Debido a la complejidad que ha resultado resolver el problema se han creado algunas variantes a este. A continuación pasamos a enumarar las más conocidas:

- Bao y Trick en [4] relajaron el problema permitendo la existencia de k fechas libres para todos los equipos dentro del torneo.
- Ribeiro y Urrutia presentaron la variante *Mirrored* en [16] en dónde se busca crear un Round Double Tournament con la condición que luego de terminar la primera mitad de enfrentamientos cada equipo se debe volver a enfrentar, en el mismo ordén, a los demás equipos intercambiando la condición de local y visita.

Otro elemento interesante del problema es que su condición de NP-Completo ya fue demostrada por Thielen y Westphal en [17]. Para ello se realizó una reducción del conocido problema $3-SAT$.

3. Estado del Arte

Antes de comenzar la revisión del estado del arte de TTP se partirá definiendo la metodología de trabajo para comparar las diversas técnicas utilizadas hasta la fecha. Debido a la diferencia de años en que cada artículo fue públicado resulta un poco díficil comparar los tiempos de ejecución de cada técnica por lo que para compararlas se considerará sólo el resultado de su función objetivo es decir la *distancia recorrida por todos los equipos*. Es sabido que para cada instancia esta distancia varia por lo que sólo analizaremos las instancias NL que pueden ser encontradas en [1].

3.1. Aproximaciones Tradicionales

Para comenzar la revisión al estado del arte de TTP se comenzará analizando las aproximaciones que utilizaron Easton, Nemhauser y Trick en [8] (artículo en dónde se definió el problema originalmente). En el artículo se describe TTP como un *Benchmark Problem* ya que el problema presenta componentes de optimización y satisfacción de restricciones por lo que podría ser interesante para ambos campos de estudio. Además se proponen dos aproximaciones:

- **Programación Lineal**: Utiliza una función objetivo que busca minimizar la cantidad de viajes realizados y las restricciones descritas en la sección anterior.
- **Problema de Satisfacción de Restricciones**: En el artículo se presenta un recetario para resolver el problema: Generar todos los conjuntos de patrones Local/Visita en ordén del número de viajes que ellos contienen, luego asignamos a cada equipo un patrón para finalmente minimizar la distancia viajada por cada uno.

Las dos aproximaciones descritas anteriormente fueron las primeras realizadas para resolver el problema las cuales pueden solamente encontrar soluciones óptimas para instancias de tan sólo seis equipos, instancias bastante alejadas de la realidad considerando que por ejemplo la MLB posee treinta equipos.

Sin embargo una conclusión interesante del trabajo realizado fue que existe una fuerte relación entre la dificultad de TTP y la del TSP asociado.

Otra aproximación interesante al problema fue realizada por Benoist, Laburthe y Rottembourg en [6]. En el artículo se realiza una aproximación mediante un algoritmo híbrido que combina dos técnicas: Relajación de Lagrange y Programación con restricciones. En el artículo los autores realizan dos relajaciones de lagrange al problema:

- **Compact Lagrange Relaxation**: Busca descomponer el problema global en un subproblema por equipo: *Construir el tour de distancia mínima para todo equipo i sin considerar ninguna restricción.* Esta relajación ayuda a encontrar una cota inferior U para el tamaño de los Road Trip sin embargo no es de gran cálidad.
- **Rich Lagrange Relaxation**:Busca asociar cada equipo a un partido específico en vez de hacerlo a una ubicación. Por lo tanto, los sub-problemas equivalen a computar, para cada equipo, la secuencia de partidos jugados en lugar de la secuencia de lugares visitados.

Además de la relajación y división del problema, este es previamente modelado como un problema de Programación con restricciones normal. Sumando la relajación de Lagrange y el modelo se crea una *Arquitectura Colaborativa* que resuelve el problema en su totalidad. Con este método se logró encontrar el óptimo a sólo instancias de con seis equipos. Sin embargo también se encontró soluciones para instancias de tamaño 8, 10, 12 y 14 las cuales no pudieron ser verificadas como óptimas.

Otra aproximación interesante la hicieron nuevamente Easton, Nemhauser y Trick en [9] utilizando Branch and Price Algorithm para combinar programación entera y programación con restricciones. La primera técnica se preocupa de resolver el problema maestro mientras que la segunda de resolver el pricing problem. Esta técnica fue la primera en resolver una instancia con ocho equipos, lo cual la hace sumamente valiosa.

3.2. Técnicas Incompletas

Técnicas incompletas también han sido utilizadas para intentar resolver TTP. Lim, Rodriguez y Xhang en [14] resolvieron el problema utilizando Hill-Climbing y Simulated Annealing. Para hacerlo dividieron el espacío de búsqueda en dos partes:

- **Timetable Space**: Espacio explorado por SA con el fin de encontrar buenas programaciones.
- **Team assignment space**: Una vez encontrada una buena programación Hill-Climbing busca la mejor asignación posible a la programación encontrada por SA.

El proceso anterior continúa hasta converger a una solución en lo posible óptima. Además para la generación de soluciones iniciales de schedules se utiliza beam search. Este conjunto de técnincas logró encontrar soluciones factibles para instancias de tamaño 16.

La técnica anterior es una mejora al artículo escrito por Xhang en [18] con la diferencia que este no generaba soluciones iniciales con *beam search.*

Otra técnica incompleta para resolver TTP fue Tabu Search. Implementada por Di Gaspero y Schaerf en [10]. Ellos seleccionaron como espacio de búsqueda el conjunto de todas las posibles programaciones de torneo, incluyendo las que no satisfacen las restricciones duras del problema. Su función de costo se definió como la suma de las violaciones hechas por cada torneo más la distancia total recorrida por todos los equipos. Además definieron cinco movimientos distintos

para cambiar de solución.La aproximación resulto ser bastante competitiva frente a las otras aunque no logró encontrar las mejores soluciones hasta la fecha de publicación.

Otro trabajo interesante es el que realizaron Goerigk y Westphal en [12] combinando Tabu Search con programación entera. La aproximación busca mejorar la solución encontrada por Tabu Search optimizando los patrones de local y visita con programación entera. En este artículo Tabu Search es implementado al igual que lo hicieron Di Gaspero y Schaerf en [10] mientras que el modelo de programación entera fue inspirado en la aproximación de Ribeiro en [15]. Este técnica resultó obtener los mejores resultados conocidos hasta la fecha de publicación para la instancia *Galaxy*. En nuestro análisis sin embargo no posee los mejores ya que no obtiene tan buenos resultados en la instancia *NL-X*.

Una aproximación distinta utilizarón Crauwels y Van Oudheusden en [7]. Para resolver TTP utilizaron la técnica de Ant Colony Optimization (ACO) y técnicas de alguna mejoras locales. Su implementación se basa en n hormigas las cuales realizan un número definido de iteraciones. Durante cada iteración t, cada hormiga k construye una programación para el campeonato. Para la determinación de un oponente para cierto equipo i una lista de candidatos es construida. Esta es explorada utilizando dos criterios: La información de la heurística y las feromonas.

ACO en [7] obtuvo malos resultados que serán evidenciados en la siguiente sección. Sus autores creen que esto se debe principalmente a que ACO se ocupa más de la parte de optimización y deja algo de lado la satisfacción de restricciones.

3.3. Perspectivas Alternativas

Una perspectiva totalmente distinta fue la realizada por Goerigk, Hoshino, Kawarabayashi y Westphal en [11] quienes abordaron TTP desde el punto de vista de la teoría de grafos. La aproximación posee tres fases que pasaremos a definir:

- **Fase 1**: Se utiliza $P_3 packing$ para producir una solución factible.
- **Fase 2**: La solución encontrada en el paso previo es mejorada utilizando *"Pairwise-swapping"*.
- **Fase 3**: Finalmente a la solución encontrada le aplicamos el algoritmo hibrído descrito por Goerigk y Westphal en [12].

El proceso anterior encontró los mejores resultados para las intancias *Galaxy* y NFL-28 por lo que su implementación fue en éxito y mejoró claramente el proceso realizado en [12].

Por otra parte Anagnostopoulos, Michel, Van Hentenryck y Vergados utilizaron Simulated Annealing en [3]. Su implemententación se basa en cuatro características que ayudan a encontrar mejores soluciones:

- Separar las restricciones del torneo y las de patrones de local y visita en restricciones duras y blandas respectivamente
- El tamaño del vecindario en dónde SA actúa es de $O(n^3)$ en dónde n es el número de equipos.
- Se utiliza una estrategia oscilatoria con el fin de balancear el tiempo invertido de búsqueda en regiones factibles e infactibles.
- Se incorpora el concepto de calentamiento para escapar de mínimos locales.

3.4. Análisis de resultados

A continuación mostraremos un resumen de todas las técnicas utilizadas y sus resultados con las instancias más utilizadas:

Técnica	Solución
Programación entera combinada con restricciones	39,479
Relajación de Lagrange y programación con restricciones	40,868
Simulated Annealing	39,721
Ant Colony Optimization	47,161

Cuadro 1: NL-8 [8], [6], [3], [7]

Técnica	Solución
Tabú Search	111,483
Tabú Search y programación entera	114,355
Simulated Annealing y Hill-Climbing Mejorado	115,089
Simulated Annealing	111,248
Ant Colony Optimization	144,373

Cuadro 2: NL-12 [10], [14], [3], [7]

De las tablas anteriores podríamos asumir que hay técnicas mejores que otras. Sin embargo esto no es verdad ya que dependiendo de la instancia evaluada alguna técnica puede obtener mejores resultados que otras. Ejemplos de lo anterior se encuentran en la literatura referenciada como por ejemplo las implementaciones de Simulated Annealing y Hill-Climbing con o sin Beam-Search en [14] y [18] respectivamente. La tabla anterior da como mejor algoritmo la implementación sin la generación de soluciones iniciales, sin embargo en el artículo muestrán otras instancias en dónde existen mejores sustanciales para la implementación con beam search. En palabras sencillas el rendimiento de una técninca tiene dos variables:

- La misma técnica en sí posee un límite de rendimiento.
- La instancia utilizada para medir el rendimiento de la misma.

4. Modelo Matemático

A continuación mostraremos un modelo matemático descrito por Jin Ho Lee, Young Hoon Lee, y Yun Ho Lee en [13]. El modelo fue hecho usando programación entera y es una versión extendida del modelo presentado por James y John en [5]. Cabe destacar que el modelo es válido para un Double Round Robin Tournament.

Notación

$$
\begin{aligned}
i &: \text{Indice del estadio de un equipo} (i = 1, 2, ..., n) \\
j &: \text{Indice del estadio de un equipo} (j = 1, 2, ..., n) \\
k &: \text{Indice del estadio un equipo} (k = 1, 2, ..., n) \\
t &: \text{Indice que indica la fecha} (t = 1, 2, ..., 2n - 2) \\
d_{ij} &: \text{Distancia entre las estadios de los equipos i y j}
\end{aligned}
$$

Variables de decisión

$\mathbf{Y_{i,j,k,t}}$: Si el equipo i viaja desde el estadio del equipo j al estadio del equipo k en la fecha t.

Función Objetivo

Se busca minimizar la distancia total viajada por los equipos.

$$\text{Min} \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{n} \sum_{t=1}^{2n-2} d_{jk} \cdot Y_{i,j,k,t} \tag{1}$$

Restricciones

- Para todo equipo, $n-1$ partidos deben ser jugados en su estadio.

$$\sum_{j=1}^{n} \sum_{t=1}^{2n-2} Y_{i,j,i,t} = n-1 \qquad \forall i \tag{2}$$

- Cada equipo debe jugar contra todos los demás exactamente una vez como visita y local.

$$\sum_{j=1}^{n} \sum_{t=1}^{2n-2} Y_{i,j,k,t} = 1 \qquad \forall i, k(k \neq i) \tag{3}$$

- No más de dos equipos pueden estar en un estadio simultáneamente.

$$\sum_{i=1}^{n} \sum_{j=1}^{n} Y_{i,j,k,t} \leq 2 \qquad \forall i, j, t(i \neq j) \tag{4}$$

- Si un equipo esta de visita, el equipo local debe estar presente en su estadio.

$$\sum_{j=1}^{n} Y_{i,j,k,t} \leq \sum_{j=1}^{n} Y_{k,j,k,t} \qquad \forall i, k, t \tag{5}$$

- Sí un equipo i está en el estadio del equipo j en la fecha t, el equipo i debe partir desde el estadio del equipo j en la fecha $t+1$

$$\sum_{j=1}^{n} Y_{i,j,k,t} = \sum_{j=1}^{n} Y_{i,k,j,t+1} \qquad \forall i, k, t \tag{6}$$

- Todos los equipos deben comenzar el torneo en sus propios estadios.

$$\sum_{j=1}^{n} Y_{i,j,i,1} = 1 \qquad \forall i \tag{7}$$

- Todos los equipos deben terminar el torneo en sus propios estadios.

$$\sum_{j=1}^{n} Y_{i,j,i,2n-1} = 1 \qquad \forall i \tag{8}$$

- Es la restricción *no-repeat*.

$$Y_{i,k,i,t} + Y_{k,k,i,t} \leq 1 \qquad \forall i, k, t(i \neq k) \tag{9}$$

- No pueden existir *Home Stands* de largo mayor a 3.

$$\sum_{t}^{t+3} Y_{i,i,i,t} \leq 3 \qquad \forall i, t = 1, 2, ..., 2n-5 \tag{10}$$

- No pueden existir *Road Trips* de largo mayor a 3.

$$\sum_{\substack{j=1 \\ \neq i}}^{n} \sum_{\substack{k=1 \\ \neq i}} \sum_{t}^{t+3} Y_{i,j,k,t} \leq 3 \qquad \forall i, t = 1, 2, ..., 2n-5 \tag{11}$$

5. Representación

Para representar la solución se utiliza una matriz de n columnas y $2(n-1)$ filas en donde cada uno de los equipos es representado por un número desde 1 a n. Además cada una de las columnas reperesenta el schedule para el equipo i con números positivos para los partidos de local y negativos en caso contrario. A continuación se presentará una solución factible para 4 equipos:

1	**2**	**3**	**4**
3	-4	-1	2
2	-1	4	-3
4	-3	2	-1
-3	4	1	-2
-2	1	-4	3
-4	3	-2	1

Cuadro 3: Representación de Solución

En el ejemplo anterior notamos que la columna 1 es el schedule para el equipo 1 quien juega sus primeros tres partidos en casa con $3, 2$ y4 respectivamente para finalizar en un Road Trip en los estadios de $3, 2$ y 4 respectivamente. Esta representación fue utilizada en [10] y [3]. Su principal ventaja es que se pueden realizar diversos movimientos de manera sencilla ya que sólo son operaciones dentro de una matriz normal. Además mapear los equipos a su respectivo número tiene complejidad $O(n)$ por lo que no se pierde mucho tiempo en decodificar la solución.

Para representar TTPR simplemente se agrega una columna dummies de ceros al final de la matriz con la esperanza que los movimientos efecutados a la solución inicial redistribuyan los ceros por el resto de la matriz.

Representación matemática y estructura de datos que se usa (arreglos, matrices, etc.), por qué se usa, la relación entre la representación matemática y la estructura.

6. Descripción del algoritmo

Para explicar el desarrollo del algoritmo Hill-Climbing con mejor mejora que fue desarrollado para la resolución de TTP se procederá a hacerlo por partes:

6.1. Generación de soluciones iniciales

Uno de las principales dificultades al intentar resolver TTP con técnicas incompletas es la generación de soluciones iniciales. Esto se debe principalmente a que TTP es una combinación entre un problema de optimalidad y satisfacción de restricciones. Para poder generar soluciones iniciales que al menos cumplan con ser un double round robin se utilizó el método del polígono descrito en [2].

6.2. Movimientos

Muchos movimientos pueden ser aplicados a las soluciones iniciales en TTP. En este trabajo sin embargo se utilizaron los movimientos descritos en [3] y [10] los cuales han demostrado tener una buena performance a pesar de lo simple que son. A continuación se pasara a revisar los tres utilizados en este trabajo: Sea $\mathcal{T}$ el conjunto de equipos y $\mathcal{R}$ el conjunto de rounds se tienen los siguientes movimientos:

1. **Swap Homes**
   - Atributos : (t_1, t_2), donde $t_1, t_2 \in \mathcal{T}$.
   - Requisito : $t_1 \not= t_2$
   - Efectos: Se realiza un cambio de localidad para los dos partidos entre t_1 y t_2.
2. **Swap Teams**
   - Atributos : (t_1, t_2), donde $t_1, t_2 \in \mathcal{T}$.
   - Requisito : $t_1 \not= t_2$
   - Efectos: Se intercambia el schedule entre t_1 y t_2 excepto cuando ellos juegan.
3. **Swap Rounds**
   - Atributos : (r_1, r_2), donde $r_1, r_2 \in \mathcal{R}$.
   - Requisito : $r_1 \not= r_2$
   - Efectos: Todos los partidos asignados a r_1 son movidos a r_2 y viceversa.

Todos los movimientos descritos fueron seleccionados debido a que el vecindario generado por cada uno de ellos no es tan grande como el que podría generar otros movimientos más complejos. Además cada uno de ellos posee una complejidad de $O(n^2)$.

6.3. Función de Evaluación

La elección de una buena función de evaluación es vital para intentar entregar buenas soluciones cuando se trabaja con movimientos que tienen una alta probabilidad de generar soluciones infactibles. En [3] se presentó la siguiente función de evaluación:

$$\text{F\_evaluación}(S) = \begin{cases} costo(S) & \text{si S es factible} \\ costo(S) + w \cdot f(nbv) & \text{e.t.o.c.} \end{cases}$$

En dónde :

- costo(S): Distancia recorrida por todos los equipos.
- nbv: Número de restricciones violadas
- f(v): Función castigo. Descrita por $f(v) = 1 + (\sqrt{v} \cdot \ln v)/2$

6.4. Hill-Climbing con mejor mejora

Como ya hemos descrito todos los pasos de nuestro algoritmo pasaremos a describir mediante un pequeño pseudocódigo nuestra implementación de HC con restarts:

```
t = 0
m =  movimiento aleatorio de los 3 disponibles
inicializar s_{best}
f_{best} = INF
repeat
    local = False
    s_c = generador_solucion(equipos)
    f_c = funcion_evaluacion(s_c)
    repeat
        Asignar a s_n la mejor soluci n del vecindario generado con el movimiento
        if f(s_n) es mejor que f(s_c) then
            s_c = s_n
        else
            local = True
        end if}
    until local
    t = t+1
    if f_c < f_{best} then
        s_best = s_c
        f_c = f_{best}
    end if
until t = M ximo de iteraciones
```

7. Experimentos

Como se esta utilizando HC con mejor mejora se sintonizará sólo el parámetros w de la función objetivo. Por tiempos de computación muy grandes sólo se experimentará con las siguientes instancias: $NL4$, $NL6$, $NL8$. Para sintonizar los parámetros se probó HC con cada uno de los movimiento mencionados arriba por separado haciendo un total de 10 iteraciones para cada uno. Los resultados expuestos en la siguiente sección dejan un par de conclusiones relevantes.

8. Resultados

A continuación se presentarán los resultados encontrados para los tres movimientos mencionados en la sección anterior. Para probar el valor del parámetro w se utilizaron valores utilizados por los creadores de la función de evaluación en [3].

Swap Rounds

Instancia	w	Distancia Total Recorrida	Factible?
NL4	8000	8669	Si
	10000	7492	No
	15000	22492	Si
NL6	8000	54775	Si
	10000	63019	Si
	15000	62656	Si
NL8	8000	127407	Si
	10000	170400	Si
	15000	130400	Si

Cuadro 4: Swap Rounds

Swap Teams

Instancia	w	Distancia Total Recorrida	Factible?
NL4	8000	8741	Si
	10000	9331	Si
	15000	8741	Si
NL6	8000	23130	No
	10000	23130	No
	15000	23130	No
NL8	8000	46943	Si
	10000	46943	Si
	15000	46943	Si

Cuadro 5: Swap Teams

A continuación se analizará el desempeño de la función de evaluación en cada uno de los movimientos por seperado:

8.1. Swap Rounds

Se puede apreciar que el movimiento encuentra soluciones bastante cercanas al óptimo en $NL4$ sin embargo esto no ocurre en $NL6$ y tampoco en $NL8$ en dónde las soluciones a pesar de ser factibles están bastante alejadas del óptimo encontrado por otros autores. Lo anterior se debe principalmente a una mala elección en la función de evaluación y a la poca cantidad de restart realizados en el experimento. Como complemento a lo anterior se aprecia que a medida que aumenta el valor del parámetro w los valores tienden a aumentar más que disminuir. Por lo que es probable que con este movimiento el castigo haya tenido que ser menor al realizado en la experimentación.

8.2. Swap Teams

Se puede apreciar que el movimiento encontró sólo soluciones factibles y todas bastante acercadas al óptimo en $NL4$ por lo que los parámetros seleccionados fueron suficientes. Para

Swap Homes

Instancia	w	Distancia Total Recorrida	Factible?
NL4	8000	7572	No
	10000	7492	No
	15000	8108	No
NL6	8000	26068	Si
	10000	25886	Si
	15000	25886	No
NL8	8000	47591	Si
	10000	46529	Si
	15000	47194	Si

Cuadro 6: Swap Homes

el caso de $NL6$ y $NL8$ se aprecia que el valor encontrado por el algoritmo es constante para los tres valores de w esto probablemente se pudo haber solucionado aumentando la cantidad de Restart en HC ya que es probable que el algoritmo se haya quedado atrapado en un óptimo local. Cabe destacar que para $NL6$ se hayó una solución infactible pero bastante cercana al óptimo mientras que para $NL8$ a pesar de ser una solución factible esta estaba bastante alejada del óptimo global.

8.3. Swap Homes

Se puede apreciar que el movimiento encontró sólo soluciones infactibles para $NL4$ independiente del valor que se le asignará al parámetro w. Para $NL6$ el algoritmo encontró sólo soluciones factibles y todas bastante cercanas al óptimo global . Por oro lado para $NL8$ se encontrarón sólo soluciones factibles pero todas bastantes alejadas del óptimo global.

8.4. Comentarios

Basado en los resultados encontrados se puede afirmar que el desempeño de cada uno de los movimientos posee una alta relación con el tipo de instancia y el tamaño de la está. Además el mal desempeño de Swap Rounds se podría explicar principalmente al bajo números de iteraciones con que se realizó el algoritmo. Por otro lado los movimientos Swap Homes y Swap Teams poseen buenos resultados en $NL6$ y $NL8$ con la diferencia que el primero entrega sólo soluciones factibles para $NL6$ mientras que el segundo sólo infactibles para $NL6$ a pesar de que ambas son bastante cercanas al óptimo.

9. Conclusiones

TTP es un problema sumamente complejo y que debe tener un tratamiento especial en cada una de sus aproximaciones. Esto se debe a la combinación entre Optimalidad y Factibilidad que las soluciones al problema deben tener. Como bien sabemos muchos algoritmos se encargan de alguna de las dos tareas anteriores pero conseguir una aproximación que pueda tratar los dos problemas es algo díficil como se ha demostrado en el artículo. Además al ser un problema NP-Completo no se puede verificar que las soluciones encontradas son óptimas por lo que hasta ahora se hace una comparación de cada una de las nuevas propuestas hechas por la comunidad científica. Si bien al comienzo se lograban resolver instancias de tamaño 8 hoy en día ya podemos ver técnicas no tradicionales que han logrado encontrar soluciones a instancias de tamaño 28. Actualmente los investigadores han apostado por resolver el problema mediante técnicas

incompletas utilizando heurísticas y metaheurísticas e incluso teoría de otras áreas como al de grafoscon el fin de poder resolver instancias mayores.

Resolver TTP con HC mejor mejora no es una tarea sencilla para instancias grandes. Los tiempos de computo en general llegan a ordenes de magnitud de horas y encontrar una solución óptima es algo sumamente costoso. El desempeño de cada uno de los movimientos implementados en este informe varía con respecto al tamaño de cada instancia. Como trabajo futuro queda probar si es que esta variación tambien se puede generalizar a la forma de cada una de las instancias.

Referencias

[1] Challenge traveling tournament problems. `http://mat.gsia.cmu.edu/TOURN/`. (Accessed on 12/15/2017).

[2] Tournament scheduling : nrich.maths.org. `https://nrich.maths.org/1443`. (Visitado el 12/15/2017).

[3] Aris Anagnostopoulos, Laurent Michel, Pascal Van Hentenryck, and Yannis Vergados. A simulated annealing approach to the traveling tournament problem. *Journal of Scheduling*, 9(2):177–193, 2006.

[4] R Bao and M.A. Trick. The relaxed traveling tournament problem. pages 472–476, 01 2010.

[5] James C Bean and John R Birge. Reducing travelling costs and player fatigue in the national basketball association. *Interfaces*, 10(3):98–102, 1980.

[6] Thierry Benoist, François Laburthe, and Benoit Rottembourg. Lagrange relaxation and constraint programming collaborative schemes for travelling tournament problems. 01 2001.

[7] H Crauwels and D Van Oudheusden. Ant colony optimization and local improvement. In *Workshop of Real-Life Applications of Metaheuristics, Antwerp, Belgium*, 2003.

[8] Kelly Easton, George Nemhauser, and Michael Trick. The traveling tournament problem description and benchmarks. In *International Conference on Principles and Practice of Constraint Programming*, pages 580–584. Springer, 2001.

[9] Kelly Easton, George Nemhauser, and Michael Trick. Solving the travelling tournament problem: A combined integer programming and constraint programming approach. In *International Conference on the Practice and Theory of Automated Timetabling*, pages 100–109. Springer, 2002.

[10] Luca Di Gaspero and Andrea Schaerf. A composite-neighborhood tabu search approach to the traveling tournament problem. *Journal of Heuristics*, 13(2):189–207, Apr 2007.

[11] Marc Goerigk, Richard Hoshino, Ken-ichi Kawarabayashi, and Stephan Westphal. Solving the traveling tournament problem by packing three-vertex paths. In *AAAI*, pages 2271–2277, 2014.

[12] Marc Goerigk and Stephan Westphal. A combined local search and integer programming approach to the traveling tournament problem. *Annals of Operations Research*, 239(1):343–354, 2016.

[13] Jin Ho Lee, Young Hoon Lee, and Yun Ho Lee. Mathematical modeling and tabu search heuristic for the traveling tournament problem. In *International Conference on Computational Science and Its Applications*, pages 875–884. Springer, 2006.

[14] Andrew Lim, Brian Rodrigues, and Xingwen Zhang. A simulated annealing and hill-climbing algorithm for the traveling tournament problem. *European Journal of Operational Research*, 174(3):1459–1478, 2006.

[15] Celso C Ribeiro. Sports scheduling: Problems and applications. *International Transactions in Operational Research*, 19(1-2):201–226, 2012.

[16] Celso C Ribeiro and Sebastián Urrutia. Heuristics for the mirrored traveling tournament problem. *European Journal of Operational Research*, 179(3):775–787, 2007.

[17] Clemens Thielen and Stephan Westphal. Complexity of the traveling tournament problem. *Theoretical Computer Science*, 412(4-5):345–351, 2011.

[18] X Zhang. Constraint programming, simulated annealing and hill-climbing algorithm for traveling tournament problems. *Bachelor thesis, School of Computing, National University of Singapore*, 2002.