Informe: Proyecto Final

Algoritmos Evolutivos - Curso 2019

Leticia Errandonea
Facultad de Ingeniería, Universidad de la República
Montevideo, Uruguay
errandonealeticia@gmail.com

Nicolás Tinte
Facultad de Ingeniería, Universidad de la República
Montevideo, Uruguay
nico.tinte@gmail.com

***Resumen*—En este artículo se presenta el problema del armado del calendario de la NBA, así como las bases para la implementación de un algoritmo evolutivo que permita encontrar una solución (o aproximación) para cualquier instancia del problema en un tiempo razonable.**

I. Descripción del problema

La *National Basketball Association* (NBA de aquí en adelante) es una liga profesional de baloncesto en Norte América, constituida por treinta equipos, de los cuales veintinueve se encuentran en Estados Unidos y el restante en Canadá. Estos equipos se dividen en las conferencia Este y Oeste y, a su vez, estas conferencias se subdividen en tres divisiones, con cinco equipos cada una.

La NBA arma su calendario con el fin de satisfacer dos objetivos: balance competitivo y reducción de costos. Hay ciertos factores que tienen un impacto para el armado de este calendario [1]:

- Cada equipo debe jugar un total de ochenta y dos partidos distribuidos de la siguiente forma:
  - Cuatro partidos contra los demás equipos de su división (dieciséis partidos en total).
  - Cuatro partidos[1] contra seis equipos de su conferencia que no sean de su división (veinticuatro partidos en total).
  - Tres partidos contra los cuatro equipos restantes de su conferencia (doce partidos en total).
  - Dos partidos contra los equipos de la otra conferencia (treinta partidos en total).
- Cada equipo debe proveer una lista con la disponibilidad del estadio que contenga:
  - Al menos cincuenta fechas en que el estadio está disponible.
  - Cuatro lunes.
  - Cuatro jueves.
- Días en lo que no se puede jugar un partido:
  - Nochebuena.
  - *All-star game*.
  - Final del campeonato NCAA.
- Conflictos, como por ejemplo, partidos de la *National Hockey League* (NHL) que coincidan en el mismo estadio deben ser resueltos.
- Los partidos pueden ser modificados para satisfacer las transmisiones televisivas.

[1]Una rotación cada cinco años define contra que equipos se juega solamente tres veces.

Entonces, el problema planteado es implementar un algoritmo evolutivo que logre armar un calendario de partidos de la NBA teniendo en cuenta algunos de los factores mencionados. Dado el alcance del proyecto, se optó por tomar en cuenta los siguientes:

- Minimizar la cantidad de millas recorridas por equipo, teniendo en cuenta que un equipo no esté más de cierto tiempo fuera de su cuidad.
- Jugar los ochenta y dos partidos distribuidos de la forma mencionada anteriormente.
- Y finalmente, respetar los días que no puede haber partidos.

A pesar que no está explícitamente indicado, se pudo observar que los equipos idealmente no pueden jugar más de dos días de forma consecutiva ni más de tres partidos en cuatro días, por lo cual, también se tomarán en cuenta estas restricciones.

II. Justificación de utilizar algoritmos evolutivos

El problema planteado es claramente un problema de optimización, ya que se busca minimizar la cantidad de millas recorridas por equipo. Este problema cae dentro de la clasificación de problemas NP difíciles, puesto que su solución puede verificarse en modo eficiente, pero no así hallarla.

En particular, este problema podría verse como una variante al problema *Travelling Salesman Problem (TSP)*, en la cuál las ciudades deben ser visitadas más de una vez y con una mayor cantidad de limitaciones y restricciones.

Sabiendo esto, se puede asegurar que un algoritmo común no encontrará una solución cercana al óptimo global de forma eficiente, y utilizar un algoritmo del tipo *Greedy* por ejemplo, podría sesgar el resultado y que este caiga siempre en un mismo óptimo local. Por esto, resulta útil utilizar un algoritmo evolutivo, para lograr encontrar una solución cercana al óptimo global y evitar el sesgo, aprovechando la característica de mutación y cruzamiento de los mismos, para extender la búsqueda de soluciones posibles.

III. Estrategia de resolución

Para la resolución se optó por aplicar un algoritmo genético (GA de aquí en adelante). Este algoritmo es una técnica de optimización aplicable a una gran variedad de problemas y esta inspirado en el principio de evolución. Es decir, se basa en el principio de la supervivencia del más apto y operadores basados en la selección natural y la herencia.

Cabe aclarar que tanto en la inicialización, como a la hora de mutar y cruzar soluciones, se deberá asegurar que las soluciones candidatas generadas sean factibles, pero estas serán penalizadas cuando un equipo juegue más de seis partidos seguidos de local, para así evitar que un equipo permanezca muchos días en su ciudad de origen (o fuera de la misma).

A. Representación de las soluciones

La representación elegida para las soluciones candidatas es una matriz NxM donde N es la cantidad de equipos y M es la cantidad de fechas en las que se pueden jugar partidos a partir del comienzo de temporada.

Cada celda ij de la matriz, puede tener tres valores:

- -2, si el equipo i no juega el día j.
- -1, si el equipo i juega de visitante el día j.
- n, si el equipo i juega contra el equipo n en condición de local el día j.

B. Función de fitness

La función de *fitness* a utilizar es simplemente minimizar la cantidad de millas promedio recorridas por los equipos, logrando así que la solución tienda a estar balanceada.

Como se mencionó anteriormente, se realiza una penalización de aquellas soluciones en las cuales un equipo juega cierta cantidad de partidos de local seguidos. Esta penalización se realizó de forma tal que, si una solución es penalizada, sea peor que cualquier otra solución factible no penalizada.

C. Condiciones de parada

Ya que no se sabe el valor final deseado, sino que se quiere llegar al menor valor posible, se decidió utilizar las siguientes condiciones de parada:

- Se alcanzó la cantidad máxima de generaciones. Esta condición se agrega para asegurar que la ejecución del algoritmo termine.
- Se mantuvo el mismo mejor valor de *fitness* durante un veinte por ciento de la cantidad máxima de generaciones. Esta condición permite detener la ejecución si el algoritmo se encuentra estancado.

D. Operadores evolutivos

1) Mutación: de forma aleatoria se intercambió la fecha de dos partidos en los que el mismo equipo fuera local.

2) Cruzamiento: de forma aleatoria se cruza una columna (un día entero del calendario) entre la solución candidata A y la B (la misma columna, es decir, el mismo día calendario).

IV. Propuesta de evaluación experimental

A. Generación de instancias

Las instancias del problema del calendario de la NBA dependen de:

- Cantidad de equipos.
- Ubicación geográfica de cada equipo.
- Fechas de *All-star game* y final del campeonato NCAA.

Dado que se observo qué la restricción respecto a en qué fechas no se pueden jugar partidos no afecta de manera relevante la solución alcanzada, la generación de instancias se centró en variar la cantidad de equipos y/o la cantidad de partidos que juega cada equipo. Esto último definió los equipos que jugaban, junto con sus ubicaciones.

Para la generación de instancias se varió la cantidad de equipos y partidos entre ellos, siempre manteniendo paridad entre conferencias y divisiones, de forma que la instancia siguiera siendo válida.

1) Solamente partidos de conferencia: En esta instancia, se utilizan los treinta cuadros existentes hoy en día en la NBA (ver Anexo A), modificando las reglas correspondientes de cuántos partidos se juegan contra los diferentes cuadros, logrando así:

- Un total de veinticuatro partidos por equipo.
- Dos partidos contra los rivales de su división (ocho en total).
- Dos partidos contra seis equipos de su conferencia (doce en total).

- Un partido contra los cuatro equipos restantes de su conferencia (cuatro en total)
- Ningún partido contra equipos de la otra conferencia.

2) Tres equipos por división: En esta instancia se utilizan tres equipos por división, totalizando nueve equipos por conferencia y dieciocho en total. De esta forma, juegan:

- Un total de cincuenta partidos por equipo.
- Cuatro partidos contra los rivales de su división (ocho en total).
- Cuatro partidos contra los demás equipos de su conferencia (veinticuatro en total). Cabe aclarar que en esta instancia no rige la rotación que causa que se jueguen menos partidos contra ciertos equipos de la misma conferencia (pero de otra división).
- Dos partidos contra los equipos de la otra conferencia (dieciocho en total)

3) Cuatro equipos por división: En esta instancia se utilizan cuatro equipos por división, totalizando doce equipos por conferencia y veinticuatro en total. De esta forma, juegan:

- Un total de sesenta y ocho partidos por equipo.
- Cuatro partidos contra los rivales de su división (doce en total).
- Cuatro partidos contra los demás equipos de su conferencia (treinta y dos en total). Cabe aclarar que en esta instancia no rige la rotación que causa que se jueguen menos partidos contra ciertos equipos de la misma conferencia (pero de otra división).
- Dos partidos contra los equipos de la otra conferencia (veinticuatro en total).

4) Temporada 2011-2012 ("Lockout season"): Esta es una instancia real de una temporada peculiar, por lo que se utilizan tres divisiones por conferencia, y en cada una de ellas, cinco equipos. Ya que la temporada comenzó a fines de diciembre y se jugaron un total de sesenta y seis partidos, para identificar qué partidos no se jugaron se buscó el calendario de dicha temporada y en base a las reglas actuales, se sacaron los partidos correspondientes.

5) Temporada NBA: Esta es una instancia real de una temporada habitual de la NBA, utilizando la rotación actual para elegir contra qué equipos de su misma conferencia pero distinta división juega tres veces.

B. Comparación con otras técnicas o soluciones

Tanto para comparar como para inicializar el GA se buscó soluciones utilizando un algoritmo *Greedy*.

En el algoritmo *Greedy* se usó como estrategia completar el calendario de cada equipo secuencialmente. Es importante destacar que, esta estrategia provoca que los primeros equipos cuyo calendario se completa, salgan beneficiados con respecto a los demás, causando que las distancias recorridas por los equipos no quede balanceada. Para esto, dado un equipo i y un rival j, se buscó la mejor fecha para jugar el partido. Para definir la mejor fecha se tuvo en cuenta:

- Millas que recorrería cada equipo desde su último partido.
- Cantidad de partidos consecutivos como local y visitante.

Cuando el algoritmo *Greedy* se utilizó para inicializar a la población en el *GA*, se empezó por un equipo i aleatorio, de forma que no todos los individuos fueran el mismo. Por otro lado, cuando se utilizó para comparar, se hicieron tantas corridas como equipos, siempre empezando por uno distinto y se eligió la mejor solución (usando la misma función de *fitness* que el *GA*).

C. Calidad de soluciones

Primeramente, se determinó la calidad de las soluciones halladas verificando que sea factible. Luego se comparó con los valores hallados por el algoritmo *Greedy*.

D. Eficiencia computacional

Se utilizarán máquinas con las especificaciones que se presentan en la siguiente tabla:

Sistema operativo	Linux Fedora 30
Procesador	quad-core 3.07GHz Intel Core i3
Memoria RAM	3GB DDR3

Para la implementación del algoritmo se optó por usar la biblioteca Malva [2]. La principal motivación para decidir el uso de dicha biblioteca es que está ya había sido utilizada por el equipo anteriormente, y esta experiencia fue positiva.

Por otro lado, se presentarán datos respecto a los valores de *fitness* y tiempo de ejecución de las diferentes configuraciones paramétricas.

Para que dichas comparaciones tengan sentido, se garantiza que las ejecuciones se realizaron en el mismo entorno, con la misma instancia del problema.

E. Métricas

Para métricas se utilizó el cálculo de *fitness* de la solución obtenida usando otras técnicas. Se comparó el *finess* promedio y desviación estándar obtenido por el algoritmo evolutivo con el de dicha solución.

Al igual que en el caso de *eficiencia computacional*, se realizaron comparaciones entre ejecuciones en un mismo entorno y con la misma instancia.

V. Resultados

A. Configuración paramétrica

Para realizar la configuración paramétrica del GA, se analizaron distintas configuraciones variando tres parámetros:

- Tamaño de la población.
- Probabilidad de cruzamiento.
- Probabilidad de mutación.

Estos parámetros fueron variados de forma tal que pudiera apreciarse su impacto en el GA, pero no demasiado como para perder posibles buenos valores. Entonces, se utilizaron los siguiente conjuntos para dichos parámetros:

- Tamaño de población: $\{50, 100, 150\}$
- Probabilidad de cruzamiento: $\{0.6, 0.8, 1.0\}$
- Probabilidad de mutación: $\{0.001, 0.01, 0.1\}$

Por lo tanto, se cuenta con un total de veintisiete configuraciones posibles. Para cada una de estas configuraciones se tomaron cincuenta muestras (es decir, se realizaron cincuenta ejecuciones del algoritmo) de forma tal de obtener información estadísticamente aceptable. Cabe aclarar que de cada una de estas ejecuciones se reporta el valor del *fitness* y el tiempo que tomó la ejecución.

A lo largo de la ejecución de la configuración paramétrica se dejaron fijos los siguientes parámetros:

- Número de corridas independientes: 10.
- Cantidad de generaciones: 1000.
- Cantidad de hijos: 100.
- Selección por torneo.

Luego de realizar las ejecuciones correspondientes a todas las permutaciones de los parámetros antes mencionados (ver Anexo B), se tomaron los valores del *fitness* obtenidos (cincuenta por permutación) y se realizaron tests de normalidad para evaluar si las muestras obtenidas siguen una distribución normal.

Dichos tests de normalidad fueron realizados utilizando tanto el test de Shapiro–Wilk [3] como el test de Anderson-Darling [4] para distribuciones normales. En ambos casos, se utilizaron funciones existentes en la biblioteca *scipy* [5] de *python* [6]. Se pueden apreciar los resultados de estos tests de normalidad en las tablas I y II.

Observando los resultados de los tests de normalidad, se puede argumentar que las distribuciones no son normales, ya que los tests fallan para una gran cantidad de conjuntos de muestras.

Dado que las muestras no siguen una distribución normal, se utilizaron las medianas de las muestras obtenidas en cada configuración para decidir cuál utilizar. Se toma como métrica de decisión la mediana puesto que los valores atípicos (es decir, lejanos a la media) tienen un efecto menor en la misma y como no se está ante una distribución normal, no se puede asegurar la ausencia de estos valores (podrían utilizarse también tests no paramétricos, tales como el test de Friedman [7] o Kurskal-Wallis [8]).

Las figuras 1 y 2 contienen un gráfico de barras de las medianas obtenidas en cada conjunto de muestras, donde cada barra es la mediana de la configuración i, siendo i el índice en las tablas I y II respectivamente.

Se observó que los valores más bajos para la instancia *Solamente por conferencia* sucedían cuando la probabilidad de mutación era 0.01, sin notorias diferencias ante cambios en los otros parámetros. Por otro lado, los menores valores para la instancia *Tres por división* también se dieron cuando la probabilidad de mutación era 0.01. Pero, en esta instancia, es posible notar una diferencia en cuanto a los otros parámetros. En particular, se puede observar que las configuraciones correspondientes a una probabilidad de cruzamiento de 0.6 o 1 son mejores que las correspondientes a esta probabilidad con valor 0.8. Entre estos dos posibles valores, se destaca como mejor la configuración con probabilidad de cruzamiento 0.6 y tamaño de población 100.

Entonces, la configuración que se utilizará para evaluar las siguientes instancias será:

- Número de corridas independientes: 10.
- Cantidad de generaciones: 1000.
- Cantidad de hijos: 100.
- Selección por torneo.
- Probabilidad de mutación: 0.01.
- Probabilidad de cruzamiento: 0.6.
- Tamaño de población: 100.

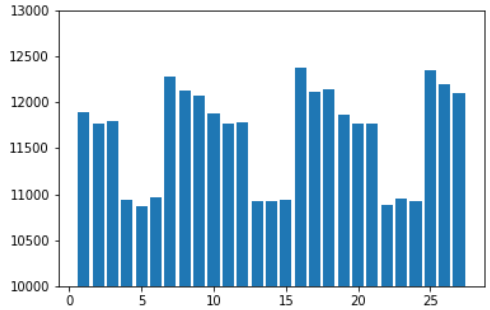

Fig. 1. Medianas para la instancia: “solamente por conferencia”

B. Evaluación experimental

Para la evaluación del algoritmo implementado, se realizaron 50 ejecuciones con los parámetros elegidos luego

P_{cruz}	P_{mut}	tam_pob	Shapiro-Wilk	Anderson-Darling
0,6	0,001	50	Sí	Sí
0,6	0,001	100	Sí	Sí
0,6	0,001	150	Sí	Sí
0,6	0,01	50	Sí	Sí
0,6	0,01	100	Sí	Sí
0,6	0,01	150	Sí	No
0,6	0,1	50	Sí	Sí
0,6	0,1	100	Sí	Sí
0,6	0,1	150	Sí	Sí
0,8	0,001	50	Sí	Sí
0,8	0,001	100	Sí	Sí
0,8	0,001	150	Sí	Sí
0,8	0,01	50	Sí	Sí
0,8	0,01	100	Sí	Sí
0,8	0,01	150	Sí	Sí
0,8	0,1	50	Sí	Sí
0,8	0,1	100	Sí	Sí
0,8	0,1	150	No	No
1	0,001	50	Sí	Sí
1	0,001	100	Sí	Sí
1	0,001	150	Sí	Sí
1	0,01	50	Sí	Sí
1	0,01	100	Sí	Sí
1	0,01	150	Sí	Sí
1	0,1	50	Sí	Sí
1	0,1	100	Sí	Sí
1	0,1	150	Sí	Sí

TABLA I
TESTS DE NORMALIDAD PARA LA INSTANCIA: "SOLAMENTE POR CONFERENCIA"

P_{cruz}	P_{mut}	tam_pob	Shapiro-Wilk	Anderson-Darling
0,6	0,001	50	No	No
0,6	0,001	100	No	No
0,6	0,001	150	No	No
0,6	0,01	50	Sí	No
0,6	0,01	100	No	No
0,6	0,01	150	Sí	Sí
0,6	0,1	50	Sí	Sí
0,6	0,1	100	Sí	Sí
0,6	0,1	150	Sí	Sí
0,8	0,001	50	No	No
0,8	0,001	100	No	No
0,8	0,001	150	No	No
0,8	0,01	50	No	No
0,8	0,01	100	No	No
0,8	0,01	150	No	No
0,8	0,1	50	Sí	Sí
0,8	0,1	100	Sí	Sí
0,8	0,1	150	Sí	Sí
1	0,001	50	No	No
1	0,001	100	No	No
1	0,001	150	No	No
1	0,01	50	Sí	Sí
1	0,01	100	No	No
1	0,01	150	No	No
1	0,1	50	Sí	Sí
1	0,1	100	Sí	Sí
1	0,1	150	Sí	Sí

TABLA II
TESTS DE NORMALIDAD PARA LA INSTANCIA: "TRES POR DIVISIÓN"

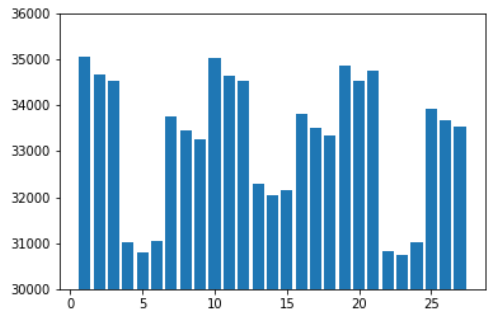

Fig. 2. Medianas para la instancia: "tres por división"

Instancia	$\overline{f}$	$\hat{f}$	$min(f)$	s	$\overline{t}$ (s)
4 por división	44.625,01	44.999,90	42.273,20	1.155,68	434,30
Temporada Lockout	40.057,55	40.054,95	38.925,60	496,37	735,88
Temporada Real	51.279,99	51.243,10	50.222,60	675,32	485,28

TABLA III
RESULTADOS DEL GA

Instancia	Fitness	$\overline{t}$ (s)
4 por división	53.139,60	1,3015
Temporada Lockout	47.065,60	1,736
Temporada Real	59.772,70	2,403

TABLA IV
RESULTADOS DEL ALGORITMO *greedy*

de la etapa de configuración paramétrica para las instancias restantes de las antes mencionadas. Estos resultados se compararon con los obtenidos con el algoritmo *Greedy* implementado, tanto en términos del valor de *fitness* obtenido como el tiempo de ejecución.

En las tablas III y IV se puede observar los resultados obtenidos de dichas ejecuciones para el GA y algoritmo *Greedy* implementados respectivamente.

Es importante destacar que cada ejecución del algoritmo *Greedy* implicaba correrlo tantas veces como equipos hubiera en la instancia, garantizando que siempre se comenzara a completar el calendario de un equipo diferente. Posteriormente, se tomó como mejor solución aquella que tuviera el menor valor de *fitness*, siendo este calculado con la misma función que para el GA. Es debido a esto que, para realizar esta comparación, el resultado final de ejecutar el algoritmo *Greedy* siempre fue el mismo, por lo que no se necesita hallar la media, mediana ni mínimo.

Observando las tablas, lo primero que se aprecia es que los resultados de *fitness* obtenidos por el GA fueron aproximadamente un 15% menores a los obtenidos por el algoritmo *Greedy*. Por otro lado, los tiempos de ejecución para el GA fueron mucho mayores (como era de esperar) a los del algoritmo *Greedy*, pero aún así en un tiempo razonable (todos menores a 10 minutos).

VI. Conclusiones

Durante la etapa de generación de instancias, se hicieron pruebas usando distinta cantidad de equipos y partidos, buscando aquellas instancias que fueran útiles para decidir valores de configuración paramétrica y, a su vez, fueran más pequeñas que las instancias que luego se usarían para la etapa experimental. Para generar instancias válidas era suficiente con mantener el criterio de que hubiera dos conferencias con la misma cantidad de divisiones de N equipos. Así fue que se probó con instancias que tenían solo dos equipos por división. Luego de realizar el análisis de resultados obtenidos para las distintas configuraciones paramétricas, se decidió usar una configuración que, finalmente, no condujo a buenos resultados en el GA, dado que se llegaban a soluciones con peor *fitness* que el algoritmo *Greedy*.

Comparando las instancias usadas originalmente con las que finalmente fueron útiles, se podría sospechar que la cantidad de partidos disputados por los equipos tiene un alto impacto en la elección de la mejor configuración.

Con respecto a los resultados obtenidos por el *GA*, lo primero interesante a destacar es que, para la instancia real, se obtiene un promedio de millas recorridas por equipo aproximadamente un 25% mayor que aquel calculado a partir del calendario confeccionado por la NBA.

A los efectos de verificar la calidad de la solución planteada, se hizo pruebas aumentando la cantidad de generaciones usadas por el *GA*. De esta forma, se llegó a pruebas que dieron por debajo de la media del calendario de la NBA. Esta solución resulta mejor (a nivel de resultados) que la propuesta en este trabajo, aunque su tiempo de ejecución es 10 veces mayor. Una posible solución al hecho de requerir un mayor números de generaciones para llegar a mejores resultados, podría ser cambiar la forma de inicialización de la población, de forma tal que los individuos iniciales se encuentren más cercanos a una solución óptima.

Analizando las millas recorridas por equipo y el tiempo de ejecución, la solución propuesta parecería ser lo suficientemente buena. De cualquier manera, si el objetivo fuera realmente armar el calendario de la NBA, la mejor opción sería aquella que insume más tiempo pero pareciera obtener a mejores resultados.

Referencias

[1] "How the NBA Schedule is Made". [Online]. Disponible: https://www.nbastuffer.com/analytics101/how-the-nba-schedule-is-made/. [Accedido: 02-Oct-2019].

[2] "The Malva Project: A framework for comutational intelligence in C++,". [Online]. Disponible: https://github.com/themalvaproject. [Accedido: 12-Oct-2019].

[3] "Shapiro-Wilk Test". Disponible: https://www.itl.nist.gov/div898/software/dataplot/refman1/auxillar/wilkshap.htm. [Accedido: 30-Nov-2019].

[4] "Anderson-Darling Test". Disponible: https://www.itl.nist.gov/div898/handbook/eda/section3/eda3661.htm. [Accedido: 30-Nov-2019].

[5] "SciPy". Disponible: https://www.scipy.org/. [Accedido: 30-Nov-2019].

[6] "Python". Disponible: https://www.python.org/. [Accedido: 30-Nov-2019].

[7] "Firedman Test". Disponible: https://www.itl.nist.gov/div898/software/dataplot/refman1/auxillar/friedman.htm. [Accedido: 30-Nov-2019].

[8] "Kruskal-Wallis Test". Disponible: https://www.itl.nist.gov/div898/software/dataplot/refman1/auxillar/kruskwal.htm. [Accedido: 30-Nov-2019].

[9] "Repositorio del proyecto". Disponible: https://github.com/Tintef/algoritmos-evolutivos (solicitar acceso). [Accedido: 02-Dic-2019].

ANEXO A
INFORMACIÓN ACTUAL SOBRE LA NBA

Equipos	Conferencia	División
Atlanta Hawks	Eastern	Southeast
Boston Celtics	Eastern	Atlantic
Brooklyn Nets	Eastern	Atlantic
Charlotte Hornets	Eastern	Southeast
Chicago Bulls	Eastern	Central
Cleveland Cavaliers	Eastern	Central
Dallas Mavericks	Western	Southwest
Denver Nuggets	Western	Northwest
Detroit Pistons	Eastern	Central
Golden State Warriors	Western	Pacific
Houston Rockets	Western	Southwest
Indiana Pacers	Eastern	Central
Los Angeles Clippers	Western	Pacific
Los Angeles Lakers	Western	Pacific
Memphis Grizzlies	Western	Southwest
Miami Heat	Eastern	Southeast
Milwaukee Bucks	Eastern	Central
Minnesota Timberwolves	Western	Northwest
New Orleans Pelicans	Western	Southwest
New York Knicks	Eastern	Atlantic
Oklahoma City Thunder	Western	Northwest
Orlando Magic	Eastern	Southeast
Philadelphia 76ers	Eastern	Atlantic
Phoenix Suns	Western	Pacific
Portland Trail Blazers	Western	Northwest
Sacramento Kings	Western	Pacific
San Antonio Spurs	Western	Southwest
Toronto Raptors	Eastern	Atlantic
Utah Jazz	Western	Northwest
Washington Wizards	Eastern	Southeast

TABLA V
CUADROS ACTUALES EN LA NBA, SUS CONFERENCIAS Y DIVISIONES

ANEXO B
RESULTADOS OBTENIDOS DE LA CONFIGURACIÓN PARAMÉTRICA

P_{cruz}	P_{mut}	tam_pob	$\overline{f}$	$\hat{f}$	$min(f)$	s	$\overline{t}$ (s)
0,6	0,001	50	11.908,21	11.884,30	11.612,00	170,01	232,51
0,6	0,001	100	11.781,71	11.769,70	11.553,40	134,45	223,24
0,6	0,001	150	11.794,60	11.798,00	11.450,70	128,64	228,93
0,6	0,01	50	10.926,81	10.937,90	10.522,60	175,12	240,19
0,6	0,01	100	10.893,03	10.874,75	10.605,10	172,75	230,54
0,6	0,01	150	10.976,11	10.971,25	10.687,10	141,43	232,64
0,6	0,1	50	12.245,74	12.279,65	11.870,80	178,17	282.00
0,6	0,1	100	12.117,97	12.131,05	11.749,40	182,47	278,14
0,6	0,1	150	12.073,85	12.072,90	11.638,20	189,37	287,82
0,8	0,001	50	11.877,10	11.873,45	11.602,90	143,19	253,85
0,8	0,001	100	11.755,84	11.760,70	11.455,50	134,87	243,04
0,8	0,001	150	11.782,21	11.777,45	11.426,00	196,57	248,09
0,8	0,01	50	10.945,04	10.925,75	10.572,90	151,70	264,12
0,8	0,01	100	10.914,51	10.918,75	10.603,90	165,18	252,26
0,8	0,01	150	10.950,74	10.942,00	10.651,60	139,14	253,76
0,8	0,1	50	12.365,77	12.369,70	12.005,70	207,43	303,33
0,8	0,1	100	12.127,57	12.111,85	11.874,70	148,89	296,47
0,8	0,1	150	12.111,97	12.135,05	11.704,80	157,48	304,03
1	0,001	50	11.875,27	11.864,75	11.586,30	149,52	329,68
1	0,001	100	11.759,26	11.764,50	11.480,80	139,52	319,44
1	0,001	150	11.755,95	11.760,75	11.316,60	153,70	324,83
1	0,01	50	10.916,77	10.888,45	10.611,10	161,80	337,75
1	0,01	100	10.944,77	10.947,80	10.546,20	140,32	328,77
1	0,01	150	10.939,76	10.929,15	10.719,10	117,22	334,47
1	0,1	50	12.320,82	12.344,85	11.787,50	211,85	386,63
1	0,1	100	12.163,79	12.192,20	11.836,50	157,37	387,93
1	0,1	150	12.102,74	12.101,60	11.735,90	169,82	405,79

TABLA VI
RESULTADOS CONFIGURACIÓN PARAMÉTRICA: INSTANCIA "SOLAMENTE PARTIDOS DE CONFERENCIA"

P_{cruz}	P_{mut}	tam_pob	$\overline{f}$	$\hat{f}$	$min(f)$	s	$\overline{t}$ (s)
0,6	0,001	50	35,139,76	35.046,20	33.925,50	771,73	293,26
0,6	0,001	100	34,771,80	34.663,60	33.516,90	750,81	284,93
0,6	0,001	150	34,516,78	34.536,20	33.726,10	527,81	297,10
0,6	0,01	50	31,016,97	31.028,70	29.918,10	545,61	279,00
0,6	0,01	100	30,856,26	30.790,10	29.739,50	737,15	289,05
0,6	0,01	150	31,082,20	31.040,55	29.801,20	658,95	279,14
0,6	0,1	50	33,754,96	33.758,15	32.158,60	639,25	295,28
0,6	0,1	100	33,492,48	33.458,10	32.254,60	516,82	301,95
0,6	0,1	150	33,215,57	33.259,75	31.914,70	479,85	295,94
0,8	0,001	50	35,155,70	35.034,05	33.787,10	862,50	297,37
0,8	0,001	100	34,754,72	34.632,60	33.866,20	696,78	288,80
0,8	0,001	150	34,690,83	34.533,00	33.491,70	902,64	302,20
0,8	0,01	50	32,304,18	32.283,20	30.061,70	1.526,44	313,85
0,8	0,01	100	32,109,04	32.055,00	29.638,10	1.522,17	288,79
0,8	0,01	150	32,242,97	32.161,25	30.188,40	1.230,08	283,57
0,8	0,1	50	33,735,64	33.808,75	32.177,80	634,94	312,48
0,8	0,1	100	33,544,54	33.502,55	32.393,20	484,68	308,88
0,8	0,1	150	33,368,53	33.347,90	32.129,60	619,80	300,58
1	0,001	50	35,210,90	34.853,75	34.121,20	1.150,11	298,02
1	0,001	100	34,665,96	34.518,90	33.386,10	1.048,50	264,93
1	0,001	150	34,850,16	34.758,40	33.713,80	988,27	301,96
1	0,01	50	30,847,39	30.820,45	29.510,70	525,54	299,58
1	0,01	100	30,863,98	30.736,75	29.998,70	595,99	288,10
1	0,01	150	31,153,10	31.025,75	29.979,00	756,69	280,75
1	0,1	50	33,878,38	33.918,75	32.304,20	572,15	314,06
1	0,1	100	33,654,33	33.677,10	31.885,60	513,81	320,68
1	0,1	150	33,489,43	33.524,50	32.397,00	447,88	297,97

TABLA VII
RESULTADOS CONFIGURACIÓN PARAMÉTRICA: INSTANCIA "TRES EQUIPOS POR DIVISIÓN"

ANEXO C
CÓDIGO FUENTE DEL PROYECTO

El código fuente, tanto del GA como del algoritmo *greedy*, así como la salida de cada ejecución del GA realizada para la configuración paramétrica y la evaluación experimental puede ser encontrada en el repositorio del proyecto [9].