



UNIVERSIDAD
DE LA REPÚBLICA
URUGUAY



FACULTAD DE
INGENIERÍA

Un método de algoritmos genéticos para obtener el fixture del Carnaval de las Promesas

Informe de Proyecto de Grado presentado por

Nicolás Vassallo, Germán Costabel, Damián
Sancristóbal

en cumplimiento parcial de los requerimientos para la graduación de la carrera
de Ingeniería en Computación de Facultad de Ingeniería de la Universidad de
la República

Supervisores

Pedro Piñeyro
Héctor Cancela

Montevideo, 1 de agosto de 2025



Un método de algoritmos genéticos para obtener el fixture del Carnaval de las Promesas por Nicolás Vassallo, Germán Costabel, Damián Sancristóbal tiene licencia [CC Atribución 4.0](#).

Agradecimientos

En primer lugar, queremos agradecer a nuestros tutores Pedro Piñeyro y Héctor Cancela por habernos brindado la oportunidad de participar en este proyecto, que resultó ser una experiencia enriquecedora tanto a nivel académico como personal. Agradecemos especialmente su acompañamiento durante todo el proceso, así como las herramientas y el conocimiento que nos proporcionaron para alcanzar los objetivos propuestos.

A la Gerencia de Eventos de la Intendencia de Montevideo por recibirnos y facilitarnos el vínculo con la Asociación de Directores del Carnaval de las Promesas (ADICAPRO). Agradecemos también a ADICAPRO por su disposición y colaboración a lo largo de todo el desarrollo de este trabajo.

Finalmente, queremos agradecer a nuestras familias y amistades por su apoyo constante a lo largo de toda la carrera.

Resumen

Este proyecto aborda el desafío de planificar el calendario de presentaciones para el Carnaval de las Promesas 2024/2025, un evento artístico y cultural organizado en Uruguay que convoca a niños, niñas y adolescentes en espectáculos de carnaval. La elaboración de este calendario, también llamado *fixture*, representa un problema de programación de eventos con múltiples restricciones logísticas, sociales y culturales. La solución planteada se basa en dos líneas de trabajo: un modelo matemático resuelto mediante un solver comercial, y un algoritmo genético diseñado específicamente para abordar el problema. Se eligió utilizar un algoritmo genético como consecuencia de un análisis sistemático del estado del arte en cuanto a la aplicación de metaheurísticas en la programación de eventos. Ambos métodos representan las restricciones duras y blandas del evento, tales como la distribución equilibrada de agrupaciones populares, la separación mínima entre actuaciones y preferencias específicas, como agrupaciones que deben abrir o cerrar etapas. La herramienta desarrollada incluye una interfaz gráfica accesible para usuarios sin conocimientos técnicos, y permite generar diferentes calendarios a partir de configuraciones parametrizables. Se realizó un proceso de validación técnica que verifica que las soluciones cumplan con todas las restricciones impuestas. Los resultados fueron evaluados por la organización responsable del evento, quienes decidieron utilizar una de las soluciones generadas en el evento con cambios menores. Se llevaron a cabo pruebas comparativas con el solver comercial y con el algoritmo genético desarrollado. De los resultados obtenidos se concluye que, tomando como referencia el solver que logra soluciones adecuadas, el algoritmo genético ofrece una alternativa que genera soluciones viables y útiles. Se encontró viable el uso de esta metaheurística para enfrentar el problema de planificación planteado.

Palabras clave: Programación de eventos, Timetabling, Metaheurística, Algoritmos genéticos, Optimización

Índice general

1. Introducción	1
2. Problema de Estudio	3
2.1. Carnaval uruguayo	3
2.1.1. Cultura y carnaval	3
2.1.2. Concurso y reglamento	6
2.1.3. Programación del Carnaval de las promesas 2024/2025 . .	8
3. Marco teórico	13
3.1. Programación de eventos	13
3.1.1. Aplicación de metaheurísticas en la programación de eventos	13
3.2. El problema de timetabling	15
3.3. Revisión de antecedentes	17
3.3.1. Metaheurísticas aplicadas a timetabling en el ámbito cul- tural	17
3.3.2. Elección de metaheurística	18
4. Modelo matemático y método AG	21
4.1. Modelo Matemático	24
4.1.1. Formulación Matemática	26
4.1.2. Relevamiento, análisis y validación de datos	30
4.1.3. Resolución del problema y validación de los resultados . .	32
4.2. Método AG	36
4.2.1. Generación de cromosomas iniciales	37
4.2.2. Inicialización aleatoria	37
4.2.3. Inicialización por secciones	38
4.2.4. Cálculo de fitness	38
4.2.5. Selección de población	39
4.2.6. Recombinación	39
4.2.7. Mutación	44
4.2.8. Diversidad genética	44
4.2.9. Reinicio poblacional	45
4.3. Desarrollo de aplicación	47
4.3.1. Tecnologías utilizadas	47

4.3.2. Aplicación	48
5. Experimentación numérica	53
5.1. Análisis comparativo	53
5.1.1. Configuraciones de datos	54
5.2. Resultados	57
6. Conclusiones y Trabajo Futuro	63
6.1. Conclusiones	63
6.2. Trabajo Futuro	64
Referencias	67
Apéndice A. Metaheurísticas aplicadas a la programación de eventos Estado del Arte	71
Apéndice B. Resultados de pruebas	117
Apéndice C. Encuesta	133
Apéndice D. Resultados de la encuesta de convocatoria	135
Apéndice E. Reglamento del Carnaval de las promesas	137

Capítulo 1

Introducción

El Carnaval de las Promesas es un evento cultural uruguayo con una larga trayectoria, en el cual niñas, niños y adolescentes participan activamente en espectáculos artísticos que se desarrollan durante el verano montevideano. Este evento, organizado por la Intendencia de Montevideo y la Asociación de Directores del Carnaval de las Promesas (ADICAPRO), constituye un espacio de integración social, formación artística y celebración comunitaria. En cada edición, más de treinta agrupaciones compiten en distintas categorías, desplegando un calendario de presentaciones que se extiende por más de dos semanas. La planificación de este calendario representa un desafío considerable, dadas las múltiples restricciones y preferencias involucradas, tanto logísticas como culturales.

En la edición 2024/2025, el evento se desarrolla en dos rondas, y cada una se extiende a lo largo de nueve días, conocidos como etapas. Durante cada una de estas etapas, se realizan cuatro presentaciones, cada una en un horario distinto, lo que permite que diferentes agrupaciones actúen a lo largo de la jornada.

En este contexto, una agrupación es un colectivo integrado por niños, niñas y adolescentes que se organiza para participar del concurso con una propuesta artística completa. Estas agrupaciones pueden pertenecer a diversas categorías tradicionales del carnaval, que son: murgas, parodistas, revistas, humoristas o sociedades de negros y lubolos. Cada una prepara un espectáculo que combina música, interpretación, canto, vestuario y escenografía, ofreciendo una representación distintiva dentro del escenario. Más allá del aspecto competitivo, las agrupaciones constituyen espacios de aprendizaje, creatividad y construcción colectiva, promoviendo valores culturales y sociales entre los más jóvenes.

Este trabajo aborda el desafío de planificar este calendario como un problema de programación de eventos (*event scheduling*), caracterizado por su elevada complejidad computacional (Even, Itai, y Shamir, 1976). En particular, se lo modela como un problema de optimización con restricciones duras y blandas, siendo las blandas las que son deseables que se cumplan y las duras las que deben cumplirse, cuya resolución manual resulta lenta y propensa a errores. Frente a esta problemática, se propone un método basado en algoritmos genéticos, una

familia de técnicas metaheurísticas inspiradas en la evolución natural, capaces de encontrar soluciones viables en espacios de búsqueda complejos y de gran dimensión.

Desde el punto de vista académico, este proyecto presenta un caso de estudio interesante por diversas razones. En primer lugar, es un problema que cae dentro de los problemas de programación de eventos que en el caso general son NP-hard (Even y cols., 1976), lo que permite aplicar técnicas de modelado matemático y optimización combinatoria en un problema real del ámbito cultural. En segundo lugar, integra el desarrollo de un sistema software completo que incluye una interfaz gráfica amigable y un backend optimizado, diseñado para ser usado por personas sin formación técnica.

Durante el desarrollo de este trabajo se logró implementar un algoritmo que permite generar *fixtures* que respetan las restricciones impuestas por ADICA-PRO, con una calidad comparable a la de métodos exactos, pero con mayor usabilidad por parte de usuarios con poco conocimiento técnico. La solución desarrollada se entrega como una herramienta práctica para los organizadores del evento.

La estructura del presente documento se organiza de la siguiente forma: en el Capítulo 2 se describe el contexto del Carnaval de las Promesas y se detalla el problema de programación a resolver. El Capítulo 3 presenta el marco teórico, incluyendo una revisión del estado del arte en programación de eventos y metaheurísticas, con énfasis en el problema de *timetabling*. En el Capítulo 4 se describe el desarrollo del modelo matemático basado en el trabajo de un proyecto anterior y se detalla la implementación del método basado en algoritmos genéticos. El Capítulo 5 describe los resultados de la experimentación y el análisis comparativo de ambos métodos. Finalmente, el Capítulo 6 presenta las conclusiones del trabajo y plantea posibles líneas de trabajo a futuro.

Capítulo 2

Problema de Estudio

En este capítulo se presenta el contexto y el problema que da origen a este trabajo, la confección del *fixture* para el Carnaval de las Promesas 2024/2025. Se aborda en primer lugar el carnaval uruguayo como fenómeno cultural de gran relevancia nacional, dando particular énfasis en el Encuentro de Carnaval de las Promesas. Este evento, organizado por la Intendencia de Montevideo y ADICAPRO, se enfoca en la participación de niños, niñas y adolescentes en las diferentes expresiones artísticas del carnaval ([Portal Institucional de la Intendencia de Montevideo, 2025](#)).

A través de una descripción del origen, el funcionamiento y la importancia sociocultural del Carnaval de las Promesas, se establece el contexto de la necesidad de una herramienta que contribuya a la planificación eficiente de su calendario de presentaciones. Esta instancia, que involucra a decenas de agrupaciones y cientos de jóvenes artistas, requiere resolver un problema complejo de programación de eventos con múltiples restricciones.

Posteriormente, se define formalmente el problema de programación asociado a la elaboración del *fixture* del carnaval, caracterizándolo como un problema de optimización de tipo NP-hard. Se analizan sus similitudes y diferencias con problemas clásicos como el *timetabling* y el *tournament scheduling*, y se detallan las restricciones específicas que plantea el caso de estudio en cuestión.

2.1. Carnaval uruguayo

En esta sección se desarrolla información sobre el carnaval uruguayo y más concretamente sobre el “Encuentro de Carnaval de las Promesas” y su reglamento para la edición 2024/2025.

2.1.1. Cultura y carnaval

La cultura de Uruguay se ha visto influenciada por las costumbres de inmigrantes de diversas regiones del mundo. Dentro de esta rica mezcla, dos géneros



Figura 2.1: Foto del espectáculo en la primera etapa de la primera ronda. Fuente: ADICAPRO.

musicales puramente autóctonos destacan: el candombe y la murga. El candombe tiene raíces africanas y se caracteriza por el sonido de tambores como el chico, repique y piano. Es una manifestación de la cultura afrodescendiente en Uruguay, con su principal expresión durante el Carnaval en el Desfile de Llamadas en los barrios Sur y Palermo de Montevideo. Por otro lado, la murga, con orígenes en Cádiz, España, consiste en un coro que utiliza letras con contenido social de forma irónica y entretenida. Las murgas, junto con otras agrupaciones como Parodistas, Revistas, Humoristas y Lubolos, actúan en escenarios barriales llamados “tablados” y compiten en el Concurso Oficial del Carnaval ([Portal Intendencia de Montevideo, 2025](#)).

El Carnaval uruguayo es una fiesta popular, declarada de Interés Nacional y reconocida por ser el más largo del mundo, extendiéndose por más de cuarenta días, usualmente desde finales de enero hasta mediados de marzo ([Portal Intendencia de Montevideo, 2025](#)). Históricamente, sus raíces se encuentran en celebraciones europeas que funcionaban como espacios de libertad colectiva. El

primer concurso oficial de comparsas en Montevideo data de 1874. Las festividades comienzan en Montevideo con el Desfile Inaugural en la Avenida 18 de Julio. Una de las expresiones más emblemáticas es el Desfile de Llamadas, donde las comparsas de candombe recorren los barrios Sur y Palermo de Montevideo ([Cardozo, Guido, y Machin, 2020](#)). Durante las noches de verano, los conjuntos carnavaleros, incluyendo las murgas, se presentan en escenarios barriales conocidos como tablados, donde despliegan humor, sátira, elaborados vestuarios y maquillajes para ofrecer su visión del país y el mundo. Las categorías que participan tanto en los tablados montevideanos como en el Concurso Oficial del Carnaval son Parodistas, Revistas, Humoristas y Lubolos. El Concurso Oficial es organizado por DAECPU (Directores Asociados de Espectáculos Carnavalescos Populares del Uruguay), una asociación fundada en 1952 e integrada por las propias agrupaciones de carnaval ([Portal de DAECPU, 2025](#)). Si bien Montevideo es el epicentro, muchas ciudades del interior tienen sus propios desfiles, pudiendo adoptar elementos del carnaval brasileño en zonas fronterizas como Rivera, Artigas y Melo.

Insertado en este gran movimiento cultural, se encuentra el Carnaval de las Promesas. A diferencia del Concurso Oficial principal, que está orientado a mayores de edad, el Carnaval de las Promesas surge con el objetivo primordial de promover la inclusión social y la integración de niñas, niños y adolescentes. Es un concurso organizado por la Intendencia de Montevideo en conjunto con la Asociación de Directores del Carnaval de las Promesas (ADICAPRO). ADICAPRO es una institución sin fines de lucro, fundada el 4 de junio de 1988. Su meta principal es el desarrollo artístico, personal y cultural de los participantes, utilizando las diversas expresiones artísticas del carnaval como una excelente oportunidad para la generación de valores e identidad cultural. El evento se concibe como un espacio de aprendizaje y disfrute que fomenta la integración y la pertenencia. Anualmente, participan aproximadamente entre mil quinientos y dos mil jóvenes, cuyas edades oscilan entre los cinco y los dieciocho años. Estos participantes están regulados por INAU (Instituto del Niño y del Adolescente del Uruguay) y la Intendencia, con control sobre los textos de los espectáculos ([Portal Institucional de la Intendencia de Montevideo, 2025](#)). En las Figuras [2.1](#), [2.2](#) y [2.3](#) se pueden observar fotografías del Carnaval de las Promesas 2024/2025.

El concurso se estructura en dos rondas en las que participan todas las agrupaciones inscriptas. Idealmente, cada etapa se compone de cuatro horarios en los que en cada uno actúa una agrupación. Las categorías de las agrupaciones participantes son las mismas que en el carnaval de adultos, adaptadas para los jóvenes: Sociedades de Negros y Lubolos, Revistas, Parodistas, Murgas y Humoristas, de acuerdo al reglamento como se presenta en el Apéndice [B.5.2](#).

- Las Sociedades de Negros y Lubolos recrean manifestaciones afro uruguayas, evocando los orígenes del candombe con tambores, cantos, bailes y trajes.
- Las Revistas son una expresión artística integral, imaginativa y tendiente a la diversión, combinando lo artístico con lo divertido en un clima alegre y colorido, con cuerpo de baile.

- Los Parodistas parodian argumentos de obras, hechos o personas, utilizando coro y a veces pistas grabadas.
- Las Murgas cantan letras inéditas con contenido social, humor e ironía, utilizando instrumentos esenciales como platillo, bombo y redoblante.
- Los Humoristas desarrollan espectáculos de comedia, con música y vestuario característico. Además del concurso, ADICAPRO impulsa actividades culturales y talleres durante todo el año que abarcan diversas disciplinas como música, canto, coreografía, baile, actuación, maquillaje, vestuario y escenografía.

En las Figuras 2.4, 3.1, 3.2, 4.1, 4.8 y 4.9 se pueden observar algunas de las agrupaciones participantes del Carnaval de las Promesas 2024/2025.

La comunicación con ADICAPRO indica que la confección del calendario para el Carnaval de las Promesas es una tarea compleja, que tradicionalmente se realizaba de forma manual o mediante sorteos. Armar el calendario implica considerar múltiples aspectos y restricciones, algunas de estas por reglamento, buscando un equilibrio en la asistencia de público. Se deben determinar las categorías y agrupaciones que actúan en cada etapa y horario.

2.1.2. Concurso y reglamento

En esta sección se resumen algunos de los artículos del reglamento del Carnaval de las Promesas de la edición 2024/2025 que se consideran relevantes para el caso de estudio. Los artículos tomados en cuenta se encuentran en el Apéndice B.5.2.

El Carnaval de las Promesas es una celebración organizada por la Intendencia de Montevideo junto con ADICAPRO, pensada especialmente para niñas, niños y adolescentes. Su objetivo principal es promover la participación artística y cultural en la infancia y juventud, generando un espacio de aprendizaje, expresión y convivencia. Este evento cuenta con tres instancias: una etapa de monitoreo, un encuentro evaluatorio y el Encuentro de Agrupaciones Infantiles, donde cada grupo realiza dos presentaciones.

El proceso de evaluación está a cargo de un equipo de monitores y jurados que no solo califica, sino que también acompaña y orienta a los grupos. La primera instancia no tiene carácter competitivo; se trata de un momento de encuentro y devolución constructiva, donde lo más importante es el proceso y no el resultado final. En estas primeras presentaciones no se permite escenografía, maquillaje o disfraces elaborados, y las agrupaciones se presentan con ropa de ensayo, enfocándose en mostrar lo que vienen construyendo. Luego, el equipo visitará los ensayos de cada grupo para seguir fortaleciendo el trabajo colectivo, y también estará atento para prevenir o actuar ante cualquier situación de violencia o vulneración de derechos hacia los participantes.

Las agrupaciones se organizan en diferentes categorías artísticas, cada una con sus propias características. En todas las categorías debe haber integrantes de al menos siete edades distintas. Por ejemplo, la categoría de Negros y Lubolos



Figura 2.2: Foto del espectáculo en la primera etapa de la primera ronda. Fuente: ADICAPRO.

recrea la cultura afro a través de los tambores y personajes típicos del candombe. Las revistas se destacan por su alegría, música y coreografías coloridas. Los parodistas representan historias conocidas desde una mirada humorística. Las murgas combinan crítica social, humor y canto coral acompañado de percusión. Por último, los humoristas presentan escenas cómicas originales. Cada categoría tiene un tiempo máximo de actuación que va desde los 35 a los 45 minutos, según el caso.

El jurado está compuesto por personas con experiencia en carnaval y formación artística, que evalúan desde una mirada educativa, valorando tanto lo artístico como el proceso de aprendizaje. Además, una persona delegada de ADICAPRO integra el equipo con el fin de facilitar la comunicación entre las agrupaciones y el jurado.

En cuanto a la organización del evento, los grupos recibirán por escrito la fecha y hora de sus presentaciones. Si alguna actuación se suspende por causas como el mal tiempo, se pospondrá. Cada grupo participa en dos rondas y el

jurado suma los puntajes de ambas para obtener el resultado final, que será definitivo y no se podrá apelar.

2.1.3. Programación del Carnaval de las promesas 2024/2025



Figura 2.3: Foto del espectáculo en la primera etapa de la primera ronda. Fuente: ADICAPRO.

Durante la etapa de definición del problema, se mantuvo una comunicación fluida con ADICAPRO y la Gerencia de Festejos y Espectáculos de la Intendencia de Montevideo. En estas instancias, se nos proporcionó una explicación en la forma en la que tradicionalmente se define el *fixture* del concurso. El proceso actual de elaboración del calendario para el concurso se realiza de forma manual, lo que implica múltiples jornadas de trabajo. Este enfoque requiere considerar simultáneamente diversas restricciones, y cualquier ajuste puede generar conflictos con otras condiciones ya establecidas. En primer lugar, se define la cantidad de agrupaciones participantes y la categoría a la que pertenecen. Según el reglamento, las categorías contempladas son: murgas, sociedades de negros y

lubolos, humoristas, parodistas y revistas. Idealmente, cada una de las dos rondas se conforma con diez etapas, procurando que cada una cuente con cuatro agrupaciones de distintas categorías. En caso de no alcanzarse este número, se puede recurrir a agrupaciones invitadas para completar la etapa correspondiente o reducir la cantidad de etapas.

Una vez definidos los grupos participantes, se determina qué agrupaciones abrirán y cuáles cerrarán cada etapa. Para el cierre, se da prioridad a aquellas que generan mayor convocatoria, ya que esta instancia tiene una relevancia particular en el contexto del carnaval. Por este motivo, se seleccionan diez agrupaciones, en función de las mejores puntuaciones obtenidas en la edición anterior del concurso. Estas agrupaciones se agrupan en dos conjuntos o “bombos” que se utilizan para realizar los sorteos correspondientes. En una primera instancia, se realiza un sorteo seleccionando cuatro de las diez agrupaciones que obtuvieron las calificaciones más altas el año anterior, que luego vuelven a ser sorteadas para ser asignadas las últimas cuatro etapas para que actúen en el último turno horario. Esta asignación responde a una preferencia de las agrupaciones por participar en los tramos finales de la ronda. Luego, se sortean las etapas restantes.

Los sorteos por horario se organizan creando bombos específicos para cada etapa. Cada bombo agrupa las agrupaciones asignadas previamente a ese horario, y se determina aleatoriamente en qué etapa actuará cada una. La conformación de estos bombos se ajusta a la cantidad de agrupaciones registradas en cada categoría. Se procura mantener cierta regularidad en los horarios asignados a cada categoría, de modo que el público pueda anticipar el momento en que cada tipo de espectáculo se presenta y evitar que en una misma etapa se presenten más de una agrupación de una misma categoría. La presencia de los directores de agrupaciones durante el sorteo busca garantizar la transparencia del proceso y evitar posibles favoritismos.

Una de las mayores dificultades en la planificación del calendario es la programación de la segunda ronda, pues deben cumplirse restricciones adicionales con respecto a lo ocurrido en la primera. Por ejemplo, dos agrupaciones que compartieron etapa en la primera ronda no pueden volver a coincidir en la segunda. También se establece una cantidad mínima de días entre la actuación de una misma agrupación en ambas rondas, con el fin de asegurar condiciones de preparación equitativas. Finalmente, ninguna agrupación que haya abierto una etapa en la primera ronda podrá repetir ese rol en la segunda.

A raíz de estas conversaciones, se llegó a un acuerdo sobre las siguientes reglas para su generación. Los *fixtures* resultantes serían presentados a las autoridades competentes y, de ser satisfactorios, se utilizarían para la edición 2024/2025 del Carnaval de las Promesas, que tendrá lugar a fines de diciembre de 2024 y principios de enero de 2025. Las restricciones particulares acordadas con ADICAPRO para el *fixture* 2024, explicadas en lenguaje natural, son las siguientes:

- **Agrupaciones que deben cerrar:** En la tradición del carnaval es considerado un honor cerrar los espectáculos de una etapa. Se desea premiar a las agrupaciones con larga trayectoria o ganadoras de años anteriores. Por lo que desde ADICAPRO se elige una lista de estas y deben cerrar

alguno de los días.

- **Sólo agrupaciones habilitadas pueden cerrar:** Dependiendo de la cantidad de días también existen agrupaciones que podrían cerrar en caso de ser necesario, pero no es obligatorio que lo hagan.
- **Agrupaciones que deben abrir:** Abrir las actuaciones de una etapa, en general, no es deseable. Puede que el público aun no haya llegado completamente y algunas categorías, como lo humoristas, requieren que el público ya este “entrado en humor”. En general las agrupaciones que abren son aquellas mas nuevas o con bajo resultado en competencias anteriores.
- **Sólo agrupaciones habilitadas pueden abrir:** Dependiendo de la cantidad de días también existen agrupaciones que podrían abrir en caso de ser necesario, pero no es obligatorio que lo hagan.
- **No deben abrir en ambas rondas:** Siendo que abrir es algo poco deseable para las agrupaciones, es preferible distribuir la carga entre mas agrupaciones y no obligar a hacerlo mas de una vez.
- **Agrupaciones no deben actuar en los primeros cuatro días:** Los grupos que actúan en etapas posteriores de las rondas tienen mayor tiempo para preparar, ensayar y corregir sus espectáculos, por lo que es considerado “malo” tener que actuar en los primeros días del concurso. A modo de premio a las agrupaciones que quedaron primeros y segundos puestos en sus respectivas categorías se les otorga el beneficio de no participar en los primeros días de la primera ronda.
- **No repetir categoría en el mismo día:** Es deseable tener etapas variadas con distintos tipos de agrupaciones. Para el año 2024 en particular no es posible cumplir con este requisito para el caso de los parodistas, al ser el grupo mas numeroso, por lo que están excluidos de esta regla.
- **Evitar categorías continuas:** Es deseable que en un mismo día no actúen solamente agrupaciones de una misma categorías para presentar variedad de espectáculos. En combinación con la regla anterior, para el año 2024, esta regla solo tiene sentido para los parodistas.
- **Deben descansar al menos cuatro días entre rondas:** Las dos rondas se organizan de tal forma que en cuanto finaliza la primera comienza la segunda. La segunda ronda es una oportunidad para que las distintas agrupaciones realicen cambios y ajustes a sus espectáculos y para esto requieren tiempo. El objetivo de esta regla es que las agrupaciones que actuaron en los ultimas etapas de la primera ronda tengan un tiempo de descanso de al menos cuatro etapas antes de que deban actuar en la segunda ronda.
- **Agrupaciones no deben repetirse entre rondas:** Al consistir la competición de dos rondas no es deseable que las dos rondas sean iguales, o

solo un nuevo orden de los días con los mismos grupos cada día. Si un par de agrupaciones cualesquiera participaron el mismo día en la ronda uno, no deben participar el mismo día en la ronda dos.

- **No deben actuar en días determinados:** El concurso del Carnaval de las Promesas es una actividad social y de integración para jóvenes, pueden haber razones por las que algunos grupos no puedan participar días determinados. A modo de ejemplo: Un integrante de una agrupación realiza su fiesta de quince por las fechas de la primera ronda, y es razonable que invite a los integrantes de su agrupación a esta fiesta, por lo que no podrían realizar su espectáculo este día.



Figura 2.4: Foto de la murga Los Pepinitos. Fuente: ADICAPRO.

Capítulo 3

Marco teórico

En este capítulo se presenta el problema de programación de eventos, así como la aplicación de metaheurísticas para la resolución de este tipo de problemas. Se habla del problema de *timetabling*, un subconjunto del problema de programación de eventos. Posteriormente, se hace una revisión de antecedentes en el uso de metaheurísticas en eventos artísticos.

3.1. Programación de eventos

Un evento, definido por la Real Academia Española, es un “suceso importante y programado, de índole social, académica, artística o deportiva” (Real Academia Española, 2024a). Los problemas de programación de eventos refieren entonces a la coordinación y planificación de estos eventos asignando el horario de cada uno de ellos, considerando las restricciones de cada evento y otros factores. Esta clase de problemas es muy común tanto en el ámbito académico como en el deportivo y administrativo, por lo cual existe una gran cantidad de literatura al respecto (Rajwar, Deep, y Das, 2023). Según (de Werra, 1985, 1997), los problemas de programación de eventos presentan una alta complejidad, lo que puede hacer que encontrar una solución óptima requiera un tiempo excesivo de cómputo. En los últimos diez años ha habido un gran interés y desarrollo de múltiples métodos de resolución que utilizan metaheurísticas para resolver estos problemas de optimización (Rajwar y cols., 2023). El uso de metaheurísticas busca obtener resultados adecuados, cercanos al óptimo, en tiempos razonables.

3.1.1. Aplicación de metaheurísticas en la programación de eventos

El término metaheurística se deriva de la combinación de dos términos griegos. Heurística del griego εὕρισκειν que significa «hallar, inventar», y el prefijo *meta* que significa «más allá» o «a un nivel superior» (Real Academia Española,



Figura 3.1: Foto de la agrupación Zabrito. Fuente: ADICAPRO.

2024b).

Según (Rajwar y cols., 2023) una heurística se refiere a una estrategia que busca guiar las búsquedas dentro del espacio de soluciones para problemas complejos, logrando encontrar soluciones aceptables en un tiempo razonable. Dada la complejidad de muchos problemas, puede resultar poco práctico explorar todo el espacio de soluciones. En cambio, una heurística se enfoca en identificar soluciones aceptables y de calidad dentro de un plazo de tiempo razonable. Si bien usualmente no se garantiza que se obtendrán soluciones óptimas, la heurística busca un equilibrio entre eficiencia y tiempo. Se supone que este enfoque proporcione soluciones de buena calidad, algunas de las cuales podrían acercarse a ser óptimas, aunque sin la certeza de lograr la mejor solución posible.

Si bien no hay un consenso en la literatura sobre la diferencia entre los términos heurística y metaheurística y muchas veces se utilizan estos términos de manera intercambiable, en (Yang, 2014) se le llama a todos los algoritmos

estocásticos que utilizan aleatorización y que además utilizan algún método de búsqueda local como metaheurísticas. Sin embargo, otros autores definen a metaheurísticas en un sentido jerárquico respecto a heurísticas de menor nivel. En (Osman y Laporte, 1996) definen formalmente metaheurística como “... un proceso de generación iterativo que guía una heurística subordinada combinando inteligentemente diferentes conceptos para explorar y explotar el espacio de búsqueda; las estrategias de aprendizaje se utilizan para estructurar la información con el fin de encontrar soluciones eficientemente cercanas a las óptimas”. En (Fink y Voss, 1999) los autores definen metaheurística como “... un proceso maestro iterativo que guía y modifica las operaciones de heurísticas subordinadas para producir eficientemente soluciones de alta calidad. Puede manipular una única solución completa (o incompleta) o una colección de soluciones en cada iteración. Las heurísticas subordinadas pueden ser procedimientos de alto (o bajo) nivel, o una simple búsqueda local, o simplemente un método de construcción”.

Los algoritmos metaheurísticos se basan en dos pasos: la intensificación y la diversificación, también conocidas como explotación y exploración. La diversificación se enfoca en generar una variedad de soluciones para explorar el espacio de búsqueda de manera global, y en general es aquí donde entra el aspecto aleatorio del algoritmo. La intensificación se centra en profundizar en áreas locales, aprovechando la información obtenida de soluciones prometedoras en esas regiones, a través de un procedimiento de búsqueda local. Juntas, estas estrategias intentan acercarse al óptimo global. Mientras que la diversificación, mediante la aleatorización, ayuda a evitar que las soluciones se queden atrapadas en óptimos locales y fomenta la variedad, la intensificación asegura que se aprovechen las mejores soluciones encontradas (Yang, 2014). Igualmente, es importante encontrar un equilibrio adecuado para que un algoritmo funcione de manera óptima. Si se pone demasiado énfasis en la exploración, la búsqueda puede extenderse demasiado, lo que resultará en una convergencia lenta. En cambio, si se prioriza excesivamente la explotación en detrimento de la exploración, es posible que el algoritmo converja más rápido, pero con menos posibilidades de encontrar el verdadero óptimo global (Sadeeq y Abdulazeez, 2023). La combinación efectiva de ambos enfoques facilita la búsqueda de la mejor solución posible en un espacio de búsqueda extenso. (Yang, 2014)

3.2. El problema de timetabling

El problema de *timetabling*, o programación de horarios, es un tipo de problema de optimización combinatoria que consiste en asignar un conjunto de eventos (por ejemplo, clases o exámenes) a franjas horarias específicas y recursos disponibles (como salones y profesores), cumpliendo una serie de restricciones. Este problema es considerado NP-hard (Even y cols., 1976), lo que significa que no existe un algoritmo que lo resuelva eficientemente en todos los casos. Dado que explorar todas las soluciones posibles puede ser computacionalmente inviable, se

recurre a métodos heurísticos y metaheurísticos para encontrar soluciones aproximadas de buena calidad en tiempos razonables. La complejidad del problema de *timetabling* radica en la cantidad de elementos que deben ser coordinados simultáneamente, así como en las múltiples restricciones que deben respetarse (Babaei, Karimpour, y Hadidi, 2015). Para detallar mejor su estructura, a continuación se detallan ejemplos de entradas del problema y los tipos de restricciones que condicionan la validez y calidad de las soluciones generadas.

En términos de un problema programático, las entradas principales del problema incluyen, en primer lugar, el conjunto de eventos que deben programarse. Estos pueden ser clases, evaluaciones, reuniones o cualquier tipo de actividad que requiera una asignación específica de tiempo y recursos. En segundo lugar, se consideran los recursos disponibles, que comprenden a los profesores o responsables de cada evento, los salones o espacios físicos en los que estos deben llevarse a cabo, y las franjas horarias habilitadas para su realización. La disponibilidad de estos recursos varía según el contexto y representa una de las principales limitaciones a la hora de generar un calendario factible. Por último, se incorporan al modelo las restricciones y preferencias que condicionan la forma en que los eventos pueden ser asignados. Estas restricciones pueden ser de carácter obligatorio (restricciones duras) o deseables (restricciones blandas) (Babaei y cols., 2015).

Las restricciones duras son aquellas que deben cumplirse de forma obligatoria para que una solución sea considerada válida (Babaei y cols., 2015). Por ejemplo, un profesor no puede dictar más de una clase de manera simultánea, ya que físicamente no puede estar en dos lugares al mismo tiempo. De la misma forma, una sala no puede estar asignada a más de un evento a la vez, dado que solo puede albergar una actividad por turno. Además, los eventos deben programarse únicamente en horarios habilitados, es decir, dentro del rango de tiempo disponible y permitido por la institución organizadora. Por otro lado, las restricciones blandas representan preferencias que, aunque no invalidan una solución si se incumplen, sí afectan su calidad. Se busca, por tanto, minimizar la cantidad o gravedad de sus violaciones (Babaei y cols., 2015). Entre estas restricciones se encuentra la preferencia por reducir los huecos entre clases en los horarios de estudiantes o docentes, lo cual mejora la eficiencia del tiempo. Asimismo, se procura evitar programar clases a última hora del día, ya que suelen ser menos efectivas en términos de atención y asistencia. Otra práctica común es agrupar clases similares en días consecutivos, lo cual facilita la continuidad temática y mejora la experiencia de aprendizaje.

En el contexto del problema de *timetabling*, la función objetivo es un componente del modelo de optimización, ya que determina qué tan buena es una solución respecto a los criterios definidos (Babaei y cols., 2015). Esta función suele estar diseñada para minimizar la cantidad o severidad de las restricciones blandas no cumplidas, dado que las restricciones duras deben satisfacerse obligatoriamente para que la solución sea válida. En el ejemplo anterior, la función objetivo podría consistir en minimizar la cantidad de huecos en los horarios de los estudiantes o docentes. Finalmente, el objetivo de un algoritmo de *timetabling* es encontrar una asignación de eventos a horarios y recursos que satisfaga

todas las restricciones duras y minimice las violaciones de restricciones blandas, optimizando el valor de la función objetivo.

3.3. Revisión de antecedentes



Figura 3.2: Foto de la agrupación Magia Candombe. Fuente: ADICAPRO.

Para la revisión de antecedentes se realizó una búsqueda sistemática de literatura relevante en las colecciones *ScienceDirect*, *SpringerLink* y *Scopus*, la cual se encuentra documentada en el Apéndice 6.2. En esta sección se dará un breve resumen de las conclusiones de ese documento.

3.3.1. Metaheurísticas aplicadas a timetabling en el ámbito cultural

A pesar de que el ámbito cultural y artístico fue considerado explícitamente en la búsqueda sistemática de literatura, en el documento incluido en el Apéndice

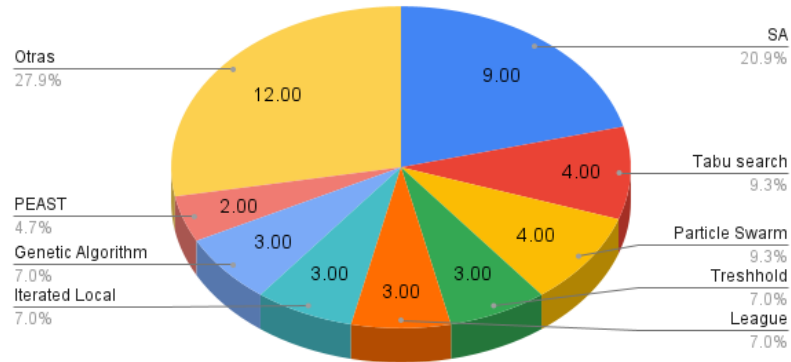


Figura 3.3: Metaheurísticas abordadas en los documentos relevados del Apéndice 6.2. Fuente: elaboración propia.

6.2 se señala que no se encontraron investigaciones específicas que apliquen metaheurísticas al problema de *timetabling* en este tipo de contextos.

La mayor parte de la literatura relevada se enfoca en problemas académicos o deportivos, principalmente utilizando instancias de competencias como la *International Timetabling Competition (ITC)*. Sin embargo, los resultados de la investigación indican que los eventos culturales también presentan características similares en cuanto a restricciones y necesidades logísticas, como la disponibilidad de salas, artistas, horarios y públicos.

Dentro de los documentos analizados se pudo apreciar que la tendencia general indica que el uso de metaheurísticas ofrece una mejora, o bien en el tiempo necesario para llegar a un resultado, o en la calidad del resultado, en varios casos en ambas dimensiones. En la Figura 3.3 se ven las metaheurísticas presentes en los documentos investigados.

3.3.2. Elección de metaheurística

Los problemas de *timetabling* en el caso general son NP-hard (Even y cols., 1976), como el problema de la generación del *fixture* para el Carnaval de las Promesas es un problema de este tipo, se puede suponer que también lo sea. Dada la cantidad de agrupaciones, rondas y etapas involucradas, el espacio de posibles soluciones es de una magnitud tal que su exploración completa resulta computacionalmente inviable. A esta complejidad se suma la necesidad de satisfacer un conjunto de restricciones duras, que definen la validez de un *fixture*, y a la vez optimizar una serie de restricciones blandas, que determinan su calidad. Este escenario, caracterizado por la posible existencia de múltiples

óptimos locales, conduce a la utilización de metaheurísticas como estrategia de resolución.

Para abordar este problema se optó por la implementación de un algoritmo genético. De acuerdo con el estado del arte incluido en el Apéndice 6.2, los algoritmos genéticos se destacan por su capacidad de explorar grandes espacios de búsqueda y evitar caer en óptimos locales gracias a su enfoque poblacional y mecanismos de diversidad. Su estructura flexible permite codificar soluciones como cromosomas, lo que facilita representar distintos tipos de problemas de programación de eventos. A diferencia de estrategias basadas en búsqueda local como *Simulated Annealing* o *Tabu Search*, que operan sobre una única solución candidata, los algoritmos genéticos trabajan con una población de soluciones. Este enfoque poblacional les permite mantener diversidad y explorar diferentes regiones del espacio de búsqueda de forma simultánea, lo cual puede reducir el riesgo de una convergencia prematura a una solución subóptima.

La eficacia de este enfoque se apoya en sus operadores, particularmente el operador de cruce (*crossover*). Este operador permite combinar segmentos de soluciones provenientes de dos individuos padres para generar un nuevo individuo. Se considera que este mecanismo de recombinación puede ser particularmente útil para este problema en comparación con los pequeños pasos incrementales de la búsqueda local. El vecindario de una solución, definido por un operador simple como el intercambio de dos actuaciones, es extenso, pues la cantidad de soluciones vecinas corresponde a combinaciones de treinta y tres tomadas de a dos por cada una de las dos rondas. Además, es un problema con un alto grado de restricciones, por lo que es probable que muchos de sus movimientos conduzcan a soluciones inválidas o no ofrezcan mejoras significativas. Esta característica del espacio de búsqueda puede dificultar el progreso de algoritmos que dependen exclusivamente de una exploración incremental.

Adicionalmente, la arquitectura de un algoritmo genético ofrece una gran flexibilidad. La representación de la solución mediante un cromosoma y la evaluación de su calidad a través de una función de *fitness* permiten modelar las particularidades del problema de manera directa, facilitando el manejo de las diversas restricciones. Considerando la estructura del problema, como una lista ordenada de horarios, y las características del método, el uso de un Algoritmo Genético se presenta como una estrategia metodológica bien fundamentada para el problema en estudio. Por estos motivos, se decidió implementar un algoritmo genético para afrontar el problema de *timetabling* en el Carnaval de las Promesas 2024/2025.

Capítulo 4

Modelo matemático y método AG

La confección de un *fixture* para la edición 2024/2025 del Carnaval de las Promesas requiere cumplir con requisitos exigidos por su reglamento y acordados entre los directores de las distintas agrupaciones. En la Sección 2.1.2 se hace un breve resumen de lo más relevante del reglamento. De forma general, es importante tomar en cuenta que el Carnaval de las Promesas es una celebración organizada por la Intendencia de Montevideo junto con ADICAPRO, pensada especialmente para jóvenes. Su objetivo principal es promover la participación artística y cultural de los jóvenes. Este carnaval cuenta con tres instancias: una etapa de monitoreo, un encuentro evaluatorio y el Encuentro de Agrupaciones Infantiles. El modelo matemático trabaja sobre el encuentro evaluatorio y sus reglas.

En esta edición se inscribieron treinta y tres agrupaciones, de las cuales son cinco murgas, doce parodistas, seis humoristas, tres lubolos y siete revistas. El concurso está compuesto de dos rondas, en las cuales deben participar una única vez en cada una de ellas todas las agrupaciones inscritas. Debido a que en esta edición están inscritas treinta y tres agrupaciones, se determinó que cada ronda consistirá de nueve etapas cada una, y actuarán cuatro agrupaciones por etapa en las etapas del día uno al seis, y tres agrupaciones por etapa en las etapas del día siete al nueve.

ADICAPRO presentó una lista con las agrupaciones que deben cerrar, es decir, actuar en el último horario de la etapa, compuesta por Bubys Bis, Quijotes, Celestinos, Dulcinea, Saphirus, Los Pepinitos, Mano a Mano, Los Chapitas. En la tradición del carnaval se considera un honor cerrar los espectáculos de una etapa. Se desea premiar a las agrupaciones con larga trayectoria o ganadoras de años anteriores, además de que poner estas al final de una etapa ayuda a motivar que el público se quede hasta el final del espectáculo. De forma similar, se indicó que las agrupaciones La Descocada, La Zafada, Yoguis, Gnomos podrían eventualmente cerrar alguna de las etapas, pero no es necesario que lo hagan.



Figura 4.1: Foto de la agrupacion Némesis. Fuente: ADICAPRO.

Además, por contraparte a como cerrar es algo deseable para las agrupaciones, abrir y ser la primera agrupación en actuar en una etapa no lo es. Puede que el público aún no haya llegado completamente y algunas categorías, como los humoristas, requieren que el público ya esté “entrado en humor”. En general, las agrupaciones que abren son aquellas más nuevas o con bajo resultado en competencias anteriores. En esta oportunidad las agrupaciones Principes, Zodiaco, Frenesi, Forever Kids, Némesis, La Resiliente, Magia Candombe, “Cosme y Damian” y Ohana deben abrir, mientras que Gnomos, Celestinos, Wizards, Klau’s, Wonkas, Zabritos, Indigos, Cherrys, Los Buscadores, Herederos, Deskara2 y Colibriquis pueden abrir, pero no es obligatorio que lo hagan. Además, siendo que abrir es algo poco deseable para las agrupaciones, es preferible distribuir la carga entre más agrupaciones y no obligar a hacerlo más de una vez.

Existen algunos requisitos con respecto a cómo se deben asignar los horarios en los que van a actuar las agrupaciones en una misma etapa. Debido a que parodistas es la única categoría en la cual hay más agrupaciones que etapas en la ronda, debe haber por lo menos una agrupación de tipo parodista en cada

una de las etapas, y a lo sumo dos. Por el caso contrario, como la cantidad de agrupaciones en las categorías murga, humoristas, lubolo y revistas es menor a la cantidad de etapas por ronda, se desea que estas estén esparcidas a lo largo de la ronda, por lo que se requiere que no se repitan agrupaciones de ninguna de estas categorías para una misma etapa. Además, existen requerimientos por parte de ADICAPRO que relacionan ambas rondas. Con el fin de presentar etapas variadas y mantener el atractivo a lo largo de ambas rondas, se requiere evitar que si un par de agrupaciones actúa en la ronda uno en la misma etapa, también lo hagan en la ronda dos. Gran parte del público del Carnaval de las Promesas corresponde a los padres de los integrantes de las agrupaciones, por lo que es normal que estos vayan a las etapas en las cuales actúan sus hijos y es deseable que no se repitan las actuaciones que ya vieron. Por otro lado, pensando en los jóvenes que componen las agrupaciones, las dos rondas se organizan de tal forma que en cuanto finaliza la primera comienza la segunda. La segunda ronda es una oportunidad para que las distintas agrupaciones realicen cambios y ajustes a sus espectáculos y para esto requieren tiempo. Es deseable que las agrupaciones que actuaron en las últimas etapas de la primera ronda tengan un tiempo de descanso de al menos cuatro etapas antes de que deban actuar en la segunda ronda. Se decidió en una asamblea de ADICAPRO que las agrupaciones que quedaron en los primeros puestos de la edición anterior para cada una de las diferentes categorías, a modo de premio, queden exentas de actuar en los primeros cuatro días de la primera ronda. Por último, el concurso del Carnaval de las Promesas es una actividad social y de integración para jóvenes y pueden haber razones por las que algunos grupos no puedan participar en días determinados. A modo de ejemplo, un integrante de una agrupación realiza su fiesta de quince por las fechas de la primera ronda, y es razonable que invite a los integrantes de su agrupación a esta fiesta, por lo que no podrían realizar su espectáculo este día.

Para generar un *fixture* que contemple todas estas reglas y restricciones se utilizaron dos métodos. El primero es la generación de un modelo matemático, el cual fue codificado en AMPL, un lenguaje de modelado algebraico diseñado para describir y resolver problemas de optimización matemática (Fourer, Gay, y Kernighan, 2003). Se utilizó CPLEX como software de optimización para generar el *fixture* de la edición 2024/2025 del evento. CPLEX es un software de optimización (*solver*) propiedad de IBM. Es una herramienta para la resolución de problemas de optimización de gran escala (Portal de IBM, 2025). Originalmente conocido por su implementación del método simplex para programación lineal (Karp, 2008), CPLEX ha evolucionado para abarcar un amplio rango de tipos de problemas, incluyendo programación entera, programación cuadrática y programación lineal con restricciones cuadráticas, así como la programación de enteros mixtos (Documentación de CPLEX, 2024). El otro método utilizado, llamado en este documento método AG, es la implementación de un algoritmo genético para tratar específicamente con el problema de estudio. La motivación para elegir algoritmos genéticos surge de las conclusiones obtenidas de realizar una revisión sistemática de la literatura relacionada con la aplicación de metaheurísticas a problemas de *timetabling* como se menciona en la Sección 3.3. Junto con la implementación del método AG, se desarrolló una interfaz para

que personas de carácter no técnico puedan utilizar el algoritmo.

Estos temas se desarrollan en este capítulo, además de explicar las validaciones realizadas para los métodos utilizados.

4.1. Modelo Matemático

En esta sección se presenta el modelo matemático propuesto para la obtención del *fixture* para la edición 2024/2025 del Carnaval de las Promesas. Este modelo está basado en el modelo presentado en (Cardozo y cols., 2020) para la edición 2019/2020 de la misma competencia, adaptado a las particularidades acordadas con ADICAPRO y explicadas en la Sección 2.1.3.

El objetivo de este modelo es lograr una programación del calendario del Carnaval de las Promesas de manera que las etapas sean equilibradas en relación con la asistencia del público. Para esto se estima la asistencia del público en base a una encuesta realizada a los distintos conjuntos carnavalescos, y se crea un *fixture* que minimiza la máxima desviación de la ponderación de popularidad de cada etapa con respecto al promedio de ponderaciones para cada una de las rondas, esto es la función objetivo que se busca minimizar, la misma será definida formalmente más adelante en esta sección en la formulación del modelo.

A continuación se definen los conjuntos utilizados en el modelo matemático. Estos representan colecciones de elementos discretos utilizados para estructurar las variables de decisión y las restricciones del problema de optimización. Para la edición 2024/2025, el calendario de carnaval contempla treinta y tres agrupaciones participantes que actúan durante dos rondas de nueve etapas. Esto implica que el conjunto de etapas D se define con $DN = 9$ y el conjunto de rondas C se define con $CN = 2$. El conjunto de horarios de las etapas T se define con $TN = 4$ correspondiente a que cada etapa tiene cuatro horarios. Dicho esto, hay etapas con un número reducido de horarios debido a la cantidad de agrupaciones participantes de esta edición. El conjunto de etapas con horario reducido D' , donde se presentan tres horarios en lugar de cuatro, corresponde a las últimas tres etapas, es decir, $D' = \{7, 8, 9\}$ y $DN' = 7$, es decir, la primera etapa con horario reducido.

Es importante destacar que, para completar las etapas con horarios reducidos, se define el conjunto U de agrupaciones ficticias, sobre las cuales se aplican restricciones específicas para modelar adecuadamente el problema.

- D : Conjunto de etapas $\{1, \dots, DN\}$
- D' : Conjunto de etapas con horario reducido $\{DN', \dots, DN\}$, $D' \subset D$ y $D' \neq D$.
- T : Conjunto de horarios $\{1, \dots, TN\}$
- C : Conjunto de rondas $\{1, \dots, CN\}$
- M : Conjunto de murgas.
- P : Conjunto de parodistas.
- H : Conjunto de humoristas.
- L : Conjunto de lubolos.

R : Conjunto de revistas.

U : Conjunto de agrupaciones ficticias.

A : Conjunto de todas las agrupaciones. $A = M \cup U \cup H \cup L \cup R \cup U$

$R1$: Conjunto de agrupaciones que pueden cerrar.

Q : Conjunto de agrupaciones que deben abrir.

$R2$: Conjunto de agrupaciones que pueden abrir.

O : Conjunto de agrupaciones que deben cerrar.

$K = \{1.,5\}$: Conjunto de tipo de categoría a la que pertenece la agrupación, con 1 = murga, 2 = lubolos, 3 = humoristas, 4 = parodistas, 5 = revistas. Notar que las agrupaciones ficticias no están en este conjunto.

UN : Conjunto de agrupaciones que no puede actuar en los cuatro primeros días.

Además, el modelo matemático se apoya en un conjunto de parámetros y variables que representan las decisiones o estados resultantes del proceso de optimización. Los parámetros se definen de la siguiente manera:

- V_i : Representa la ponderación asignada a cada agrupación $i \in A$, indicando su popularidad o capacidad de convocatoria. Este valor para cada agrupación fue obtenido mediante una encuesta realizada a las distintas agrupaciones en las que cada agrupación asignaba una calificación a todas las demás, y se tomo el promedio de sus calificaciones. La encuesta se puede ver en el Apéndice B.5.2 y los resultados se pueden ver en el Apéndice B.5.2. Para las agrupaciones de U se le signo el promedio de popularidad de todas las agrupaciones
- w : Se calcula como el promedio de ponderación de todas las agrupaciones, $w = \frac{\sum_{i \in A} V_i}{|A|}$, donde $|A|$ es el número total de agrupaciones.
- $F_i \in K$: Parámetro para agrupación i que define el tipo de categoría a la que pertenece la agrupación.

Las variables del modelo se dividen en dos categorías: variables de decisión y variables auxiliares. Las variables de decisión determinan las asignaciones dentro del cronograma:

- $x_{i,d,t,c} : \begin{cases} 1 & \text{si la agrupación } i \text{ actúa en la etapa } d, \text{ en el horario } t, \text{ en la ronda } c. \\ 0 & \text{si no.} \end{cases}$
- z_c : Esta variable auxiliar se emplea para medir la máxima desviación con respecto a la ponderación promedio en cada ronda c .
- $y_{d,c}$: Para cada etapa d y ronda c , esta variable representa la ponderación total de la etapa, calculada como la suma de las ponderaciones V_i de todas las agrupaciones que actúan en dicha etapa en esa ronda.

4.1.1. Formulación Matemática

La formulación matemática del problema se presenta a continuación, estructurada en una función objetivo y un conjunto de restricciones. Esta sección detalla cómo los conjuntos, parámetros y variables definidos previamente se relacionan para modelar las decisiones de programación. La función objetivo busca optimizar un criterio específico, mientras que las restricciones garantizan que la solución propuesta sea factible y cumpla con todos los requisitos operacionales y normativos.

El objetivo principal de este modelo es minimizar la variabilidad en la popularidad de las agrupaciones a lo largo de las etapas, buscando un equilibrio que evite que una etapa sea significativamente más o menos atractiva que el promedio general. Para lograr esto, la función objetivo (1) minimiza la suma de las máximas desviaciones (z_c) de la popularidad promedio en cada ronda.

$$\text{mín } \sum_{c \in C} z_c \quad (1)$$

sujeto a:

$$y_{d,c} = \frac{\sum_{i \in A} \sum_{t \in T} V_i x_{i,d,t,c}}{4}, \quad \forall d \in D, \forall c \in C \quad (2)$$

$$z_c \geq y_{d,c} - w, \quad \forall d \in D, \forall c \in C \quad (3)$$

$$z_c \geq w - y_{d,c}, \quad \forall d \in D, \forall c \in C \quad (4)$$

$$\sum_{t \in T} \sum_{d \in D} x_{i,d,t,c} = 1, \quad \forall i \in A, \forall c \in C \quad (5)$$

$$\sum_{i \in A} x_{i,d,t,c} = 1, \quad \forall t \in T, \forall d \in D, \forall c \in C \quad (6)$$

$$x_{i,d,TN,c} = 0, \quad \forall i \in A \setminus R1, \forall d \in D \setminus D', \forall c \in C \quad (7)$$

$$x_{i,d,TN-1,c} = 0, \quad \forall i \in A \setminus R1, \forall d \in D', \forall c \in C \quad (8)$$

$$\sum_{c \in C} (\sum_{d \in D \setminus D'} x_{i,d,TN,c} + \sum_{d \in D'} x_{i,d,TN-1,c}) \geq 1, \quad \forall i \in O \quad (9)$$

$$x_{i,d,1,c} = 0, \quad \forall i \in A \setminus R2, \forall d \in D, \forall c \in C \quad (10)$$

$$\sum_{c \in C} (\sum_{d \in D} x_{i,d,1,c}) \geq 1, \quad \forall i \in Q \quad (11)$$

$$\sum_{c \in C} (\sum_{d \in D} x_{i,d,1,c}) \leq 1, \quad \forall i \in A \quad (12)$$

$$x_{i,d,t,c} + x_{p,d,t+1,c} \leq 1, \quad \forall d \in D, \forall c \in C, \forall t \in \{1, (TN-1)\}, \forall i, p \in A, F_i = F_p \quad (13)$$

$$\sum_{i \in L} \sum_{t \in T} x_{i,d,t,c} \leq 1, \quad \forall d \in D, \forall c \in C \quad (14)$$

$$\sum_{i \in M} \sum_{t \in T} x_{i,d,t,c} \leq 1, \quad \forall d \in D, \forall c \in C \quad (15)$$

$$\sum_{i \in R} \sum_{t \in T} x_{i,d,t,c} \leq 1, \quad \forall d \in D, \forall c \in C \quad (16)$$

$$\sum_{i \in P} \sum_{t \in T} x_{i,d,t,c} \geq 1, \quad \forall d \in D, \forall c \in C \quad (17)$$

$$\sum_{i \in P} \sum_{t \in T} x_{i,d,t,c} \leq 2, \quad \forall d \in D, \forall c \in C \quad (18)$$

$$\sum_{i \in H} \sum_{t \in T} x_{i,d,t,c} \leq 1, \quad \forall d \in D, \forall c \in C \quad (19)$$

$$\sum_{t \in T} x_{i,d,t,1} + \sum_{t \in T} x_{p,d,t,1} + \sum_{t \in T} x_{i,l,t,2} + \sum_{t \in T} x_{p,l,t,2} \leq 3, \quad \forall d, l \in D, \forall i, p \in A \setminus U, i \neq p \quad (20)$$

$$\sum_{t \in T} x_{i,6,t,c} \leq \sum_{j=2}^9 \sum_{t \in T} x_{i,j,t,c+1}, \quad \forall i \in A, \forall c \in \{1, \dots, CN-1\} \quad (21)$$

$$\sum_{t \in T} x_{i,7,t,c} \leq \sum_{j=3}^9 \sum_{t \in T} x_{i,j,t,c+1}, \quad \forall i \in A, \forall c \in \{1, \dots, CN-1\} \quad (22)$$

$$\sum_{t \in T} x_{i,8,t,c} \leq \sum_{j=4}^9 \sum_{t \in T} x_{i,j,t,c+1}, \quad \forall i \in A, \forall c \in \{1, \dots, CN-1\} \quad (23)$$

$$\sum_{t \in T} x_{i,9,t,c} \leq \sum_{j=5}^9 \sum_{t \in T} x_{i,j,t,c+1}, \quad \forall i \in A, \forall c \in \{1, \dots, CN-1\} \quad (24)$$

$$x_{i,d,TN,c} = 0, \quad \forall i \in A \setminus U, \forall d \in D', \forall c \in C \quad (25)$$

$$\sum_{t \in T} x_{16,5,t,1} = 0 \quad (26)$$

$$\sum_{t \in T} x_{28,5,t,1} = 0 \quad (27)$$

$$x_{i,d,t,c} \in \{0, 1\}, \quad \forall i \in A, \forall d \in D, \forall t \in T, \forall c \in C \quad (28)$$

$$y_{d,c} \geq 0, \quad \forall d \in D, \forall c \in C \quad (29)$$

$$z_c \geq 0, \quad \forall d \in D, \forall c \in C \quad (30)$$

La función objetivo del modelo, representada por la restricción (1), busca minimizar la suma de la máxima desviación (z_c) de popularidad en cada ronda (c). El objetivo es que la popularidad de cada etapa dentro de una ronda sea lo más cercana posible al promedio de popularidad de todas las etapas, logrando así un equilibrio general del atractivo para el público a lo largo de todo el carnaval.

Las restricciones de (2) se encargan de calcular la ponderación promedio ($y_{d,c}$) para una etapa específica (d) en una ronda determinada (c). Para ello, suma las popularidades (V_i) de todas las agrupaciones (i) asignadas a esa etapa y ronda en todos los horarios (t), y divide el total por cuatro (cantidad de actuaciones por etapa). Esto cuantifica el nivel promedio de “atractivo” por actuación para esa combinación particular de etapa y ronda.

Las restricciones de (3) y (4) trabajan en conjunto para definir la desviación (z_c) para cada ronda. Las restricciones de (3) aseguran que z_c para una ronda c sea al menos la diferencia positiva entre la popularidad de cualquier etapa ($y_{d,c}$) y el promedio general (w), cubriendo los casos en que la etapa es más popular. Complementariamente, las restricciones de (4) garantizan que z_c sea al menos la diferencia positiva entre el promedio general (w) y la popularidad de la etapa ($y_{d,c}$), para casos donde la etapa es menos popular. Conjuntamente, estas restricciones aseguran que z_c capture la máxima desviación absoluta de popularidad de una etapa respecto al promedio dentro de cada ronda.

Las restricciones de (5) imponen que cada agrupación (i) debe actuar exactamente una vez en cada ronda (c). Esto se logra al sumar todas las posibles asignaciones de una agrupación específica (en cualquier etapa d y horario t) dentro de una ronda, con el total forzado a ser 1. Esta restricción garantiza la participación de todas las agrupaciones a lo largo del calendario.

Las restricciones de (6) aseguran que en cada horario (t), en cada etapa (d), y en cada ronda (c), debe actuar exactamente una agrupación (i). Esta restricción garantiza que cada espacio disponible en el calendario de actuaciones esté ocupado por una única agrupación, asegurando la coherencia de la programación.

Las restricciones de (7) y (8) limitan las posiciones de cierre de etapa. Específicamente, las restricciones de (7), $x_{i,d,TN,c} = 0 \quad \forall i \in A \setminus R1, \forall d \in D \setminus D', \forall c \in C$, prohíben que las agrupaciones no pertenecientes al conjunto $R1$ (aquellas que no pueden cerrar la etapa) sean asignadas al último horario (TN) de las etapas con cuatro horarios ($D \setminus D'$). De forma similar, las restricciones de (8), $x_{i,d,TN-1,c} = 0 \quad \forall i \in A \setminus R1, \forall d \in D', \forall c \in C$, impiden que las agrupaciones fuera del conjunto $R1$ sean asignadas al penúltimo horario ($TN - 1$) de las etapas con tres horarios (D'). Estas restricciones garantizan que solo las agrupaciones designadas para cerrar ocupen estos roles especiales. La restricción de (9), $\sum_{c \in C} (\sum_{d \in D \setminus D'} x_{i,d,TN,c} + \sum_{d \in D'} x_{i,d,TN-1,c}) \geq 1 \quad \forall i \in O$, es una condición imperativa que obliga a que las agrupaciones que deben cerrar (conjunto O) lo hagan al menos una vez en el transcurso de todas las rondas, considerando ambos tipos de horarios de cierre.

Respecto a la apertura, la restricción de (10), $x_{i,d,1,c} = 0 \quad \forall i \in A \setminus R2, \forall d \in D, \forall c \in C$, prohíben que las agrupaciones que no pertenecen al conjunto $R2$ (agrupaciones que pueden abrir) actúen en el primer horario (1) de cualquier etapa y ronda, reservando este espacio para las agrupaciones designadas. Las restricciones de (11), $\sum_{c \in C} (\sum_{d \in D} x_{i,d,1,c}) \geq 1 \quad \forall i \in Q$, obligan a que las agrupaciones que deben abrir (conjunto Q) actúen al menos una vez en el primer horario a lo largo de todas las rondas. Finalmente, la restricción de (12), $\sum_{c \in C} (\sum_{d \in D} x_{i,d,1,c}) \leq 1 \quad \forall i \in A$, limitan la cantidad de veces que cualquier agrupación puede abrir, estableciendo que ninguna agrupación puede hacerlo más de una vez en todas las rondas combinadas, previniendo una concentración excesiva en los roles de apertura.

Las restricciones de (13), $x_{i,d,t,c} + x_{p,d,t+1,c} \leq 1 \quad \forall d \in D, \forall c \in C, \forall t \in \{1, (TN-1)\}, \forall i, p \in A, F_i = F_p$, es una familia de restricciones de secuenciación que prohíben que dos agrupaciones (i y p) de la misma categoría (indicado por $F_i = F_p$) actúen consecutivamente en la misma etapa (d) y misma ronda (c). Es decir, si la agrupación i está asignada al horario t , entonces una agrupación p de la misma categoría no puede ser asignada al horario siguiente $t + 1$, promoviendo la variedad en el espectáculo. Notar que las reglas (14), (15), (16) y (19) hacen que esta regla solo tenga sentido para la categoría parodistas, pero generalizar su escritura representa mejor la idea de que no es deseable categorías contiguas.

Las restricciones de (14), (15), (16) y (19) establecen un límite de una agrupación por etapa y ronda para las categorías de Lubolos (L), Murgas (M), Revistas (R) y Humoristas (H), respectivamente. Esto se representa como $\sum_{i \in L} \sum_{t \in T} x_{i,d,t,c} \leq 1 \quad \forall d \in D, \forall c \in C$, $\sum_{i \in M} \sum_{t \in T} x_{i,d,t,c} \leq 1 \quad \forall d \in D, \forall c \in C$, $\sum_{i \in R} \sum_{t \in T} x_{i,d,t,c} \leq 1 \quad \forall d \in D, \forall c \in C$ y $\sum_{i \in H} \sum_{t \in T} x_{i,d,t,c} \leq 1 \quad \forall d \in D, \forall c \in C$. Es decir, en cada etapa (d) de cada ronda (c), puede actuar como máximo una agrupación de cada una de estas categorías. Por otro lado, las restricciones de (17) y (18) definen los límites para la categoría Parodistas (P) que,

por ser la categoría con mas agrupaciones, se desea siempre actúen por lo menos una de ellas en cada etapa, pero no mas de 2. Las restricciones de (17), $\sum_{i \in P} \sum_{t \in T} x_{i,d,t,c} \geq 1 \quad \forall d \in D, \forall c \in C$, garantizan que en cada etapa (d) de cada ronda (c) debe actuar al menos una agrupación de Parodistas, mientras que las restricciones de (18), $\sum_{i \in P} \sum_{t \in T} x_{i,d,t,c} \leq 2 \quad \forall d \in D, \forall c \in C$, complementan esto al establecer que pueden actuar como máximo dos agrupaciones de la categoría Parodistas en la misma etapa y ronda.

Las restricciones de (20), $\sum_{t \in T} x_{i,d,t,1} + \sum_{t \in T} x_{p,d,t,1} + \sum_{t \in T} x_{i,l,t,2} + \sum_{t \in T} x_{p,l,t,2} \leq 3 \quad \forall d, l \in D, \forall i, p \in A \setminus U, i \neq p$, es una familia de restricciones compleja que limita la co-ocurrencia de dos agrupaciones distintas (i y p) en las dos primeras rondas. Suma cuatro eventos: la actuación de la agrupación i en la etapa d (ronda uno), la de la agrupación p en la etapa d (ronda uno), la de la agrupación i en la etapa l (ronda dos), y la de la agrupación p en la etapa l (ronda 2). El total de estos cuatro eventos, para cualquier par de etapas d, l y cualquier par de agrupaciones distintas i, p , no puede exceder 3. Esto restringe la frecuencia con la que dos agrupaciones específicas pueden compartir escenario o aparecer en ciertas combinaciones de etapas durante las dos primeras rondas, forzando una mayor dispersión de las agrupaciones de alta popularidad y evitando tener dos etapas con agrupaciones muy similares en ambas rondas.

Las restricciones de (21), (22), (23) y (24) establecen una serie de dependencias entre las etapas de actuación de una agrupación en la ronda c y las posibles etapas en la ronda $c + 1$. Por ejemplo, las restricciones de (21), $\sum_{t \in T} x_{i,6,t,c} \leq \sum_{j=2}^9 \sum_{t \in T} x_{i,j,t,c+1} \quad \forall i \in A, \forall c \in \{1, \dots, CN-1\}$, indican que si una agrupación (i) actúa en la etapa seis durante la ronda c , entonces obligatoriamente debe actuar en alguna de las etapas dos a nueve durante la ronda $c + 1$. Esto garantiza al menos cuatro días de descanso entre las actuaciones de ambas rondas. De manera análoga, las restricciones de (22), $\sum_{t \in T} x_{i,7,t,c} \leq \sum_{j=3}^9 \sum_{t \in T} x_{i,j,t,c+1} \quad \forall i \in A, \forall c \in \{1, \dots, CN-1\}$, aplica esta lógica para la etapa siete en ronda c (requiriendo actuación en etapas tres a nueve en ronda $c + 1$); las restricciones de (23), $\sum_{t \in T} x_{i,8,t,c} \leq \sum_{j=4}^9 \sum_{t \in T} x_{i,j,t,c+1} \quad \forall i \in A, \forall c \in \{1, \dots, CN-1\}$, para la etapa ocho en ronda c (requiriendo actuación en etapas cuatro a nueve en ronda $c + 1$); y las restricciones de (24), $\sum_{t \in T} x_{i,9,t,c} \leq \sum_{j=5}^9 \sum_{t \in T} x_{i,j,t,c+1} \quad \forall i \in A, \forall c \in \{1, \dots, CN-1\}$, para la etapa nueve en ronda c (requiriendo actuación en etapas cinco a nueve en ronda $c + 1$). Estas restricciones son presentadas para el caso particular de nueve etapas por ronda y cuatro días de descanso, pero es posible generalizar para más un número diferente de etapas o días de descanso.

Las restricciones de (25), $x_{i,d,TN,c} = 0 \quad \forall i \in A \setminus U, \quad \forall d \in D', \quad \forall c \in C$, se encargan de gestionar la asignación de agrupaciones ficticias en los horarios de cierre. Esta restricción prohíbe específicamente que cualquier agrupación real (es decir, que no pertenezca al conjunto de agrupaciones ficticias U) actúe en el último horario (TN) de las etapas consideradas “cortas” (D'), en cualquier ronda (c). La razón subyacente a esta formulación se deriva de la estructura del calendario del Carnaval de las Promesas: aunque una ronda consta de nueve etapas de cuatro horarios cada una, hay solo treinta y tres agrupaciones, por lo que es insuficiente para llenar todos los horarios. Para compensar esto,

ADICAPRO decidió que, en los últimos tres días de cada ronda, solo actúen tres agrupaciones por día, dejando el último horario de cada uno de esos días vacío para las agrupaciones reales. Para mantener la coherencia del modelo matemático y permitir el cálculo del valor objetivo, se introdujeron agrupaciones ficticias, cuya popularidad se estima como el promedio de la popularidad de todas las agrupaciones reales. Estas agrupaciones ficticias son asignadas a esos horarios vacíos, garantizando que el modelo opere con un número constante de actuaciones por día sin afectar la popularidad real de las etapas.

Finalmente, la restricción (26) y (27) son restricciones de prohibición particulares. Estas limitaciones reflejan requisitos organizativos que debían ser incorporados en el calendario. Las restricciones de (28) definen la naturaleza binaria de la variable principal de decisión $x_{i,d,t,c}$, que toma el valor 1 si la agrupación i actúa en la etapa d , en el horario t , durante la ronda c , y 0 en caso contrario. Las restricciones de (29) y (30) restringen el dominio de estas variables auxiliares utilizadas para calcular el valor de la función objetivo a los reales positivos, incluyendo el cero.

4.1.2. Relevamiento, análisis y validación de datos

Los parámetros de entrada del modelo están compuestos por las agrupaciones a programar, su respectiva ponderación de popularidad, el promedio de esta ponderación, los conjuntos de agrupaciones que pueden cerrar y abrir una etapa y los conjuntos de agrupaciones que deben cerrar y abrir una etapa, los días que cuentan con menos horarios. Además, este año se solicitó que ciertos grupos no participen en los primeros cuatro días de la primera ronda, por lo que se agrega como parámetro.

La información de las agrupaciones que participan este año del concurso fue brindada por ADICAPRO, y para la edición 2024/2025 está compuesta de la siguiente manera: siete revistas, doce parodistas, seis humoristas, cinco murgas y tres lubolos, totalizando treinta y tres agrupaciones. Del total de agrupaciones que participan en esta edición.

- **Revistas:** Dulcinea, Saphirus, Mandala, Los Buscadores, Frenesi, Forever Kids, Némesis.
- **Parodistas:** Bubys Bis, Zero, Quijotes, Celestinos, Wizards, Klau's, Wonkas, Zabritos, Indigos, Principes, Zodiaco, Cherrys.
- **Humoristas:** Los Chapitas, Yoguis, Gnomos, Herederos, Deskara2, Colibriquis.
- **Murgas:** Los Pepinitos, La Descocada, La Zafada, Mano a Mano, La Resiliente.
- **Lubolos:** Magia Candombe, Cosme y Damian, Ohana.

- **Agrupaciones que pueden abrir:** Celestinos, Wizards, Klau's, Wonkas, Zabritos, Indigos, Cherrys, Los Buscadores, Gnomos, Herederos, Deskara2, Colibriquis, Principes, Zodiaco, Frenesi, Forever Kids, Némesis, La Resiliente, Magia Candombe, Cosme y Damian, Ohana.
- **Agrupaciones que deben abrir:** Principes, Zodiaco, Frenesi, Forever Kids, Némesis, La Resiliente, Magia Candombe, Cosme y Damian, Ohana.
- **Agrupaciones que pueden cerrar:** Celestinos, Bubys Bis, Quijotes, Dulcinea, Saphirus, Los Pepinitos, Mano a Mano, Los Chapitas, Gnomos, La Descocada, La Zafada, Yoguis.
- **Agrupaciones que deben cerrar:** Celestinos, Bubys Bis, Quijotes, Dulcinea, Saphirus, Los Pepinitos, Mano a Mano, Los Chapitas.
- **No pueden actuar en las primeras cuatro etapas:** Bubys Bis, Dulcinea, La Descocada, Los Chapitas, Magia Candombe.
- **Etapas cortas:** 7, 8, 9.
- **Ponderación promedio de cada agrupación:** 5.3.
- **Ponderación de popularidad:** 9.44 Bubys Bis, 5.44 Zero, 9.88 Quijotes, 7.63 Celestinos, 5.81 Wizards, 5.5 Klau's, 3.38 Wonkas, 1.31 Zabritos, 0.77 Indigos, 1.44 Principes, 2.39 Zodiaco, 2 Cherrys, 9.64 Dulcinea, 9.63 Saphirus, 4.73 Mandala, 2.25 Los Buscadores, 1.75 Frenesi, 2 Forever Kids, 2.36 Némesis, 9.13 Los Pepinitos, 8.56 La Descocada, 8.88 La Zafada, 9.5 Mano a Mano, 2.94 La Resiliente, 8 Los Chapitas, 6.63 Yoguis, 7.21 Gnomos, 4.36 Herederos, 2 Deskara2, 2.63 Colibriquis, 4.13 Magia Candombe, 6.38 Cosme y Damian, 7.25 Ohana.

Además de obtener un *fixture* válido según las restricciones impuestas, es de interés generar un cronograma equilibrado en términos de concurrencia del público. Para este propósito, se empleó la misma función objetivo utilizada en proyectos anteriores, la cual considera la distribución equitativa de agrupaciones de alta y baja convocatoria a lo largo del calendario.

Con el fin de estimar dicha convocatoria, se realizó una encuesta entre todas las agrupaciones participantes, en la cual debían evaluar la popularidad del resto de los grupos (exceptuando el propio), entendiendo por popularidad el nivel de atracción de público que cada agrupación genera. Los detalles metodológicos de esta encuesta y sus resultados se encuentran documentados en los Apéndices B.5.2 y B.5.2 respectivamente.

Es importante clarificar por qué no se recurrió a otros métodos de estimación. En este sentido, en (Cardozo y cols., 2020) se planteó el uso de la venta anticipada de entradas como indicador de popularidad. Sin embargo, este enfoque fue descartado por la organización, ya que existen agrupaciones con alta convocatoria histórica que no realizan ventas anticipadas, así como agrupaciones nuevas que carecen de datos históricos y quedarían clasificadas erróneamente.

4.1.3. Resolución del problema y validación de los resultados

Una vez implementado en AMPL el modelo de optimización de la Sección 4.1.1, se procedió a la generación de distintos calendarios de presentaciones utilizando los datos provistos por la organización. Para asegurar la validez de los resultados obtenidos, se implementó un script en JavaScript que replicaba las reglas y restricciones acordadas con ADICAPRO, verificando automáticamente que cada solución generada cumpliera con los criterios establecidos. Esta verificación, además de una validación manual, permitió confirmar que los resultados satisficieran las reglas.

ADICAPRO planteó que le interesaba contar con varios calendarios para elegir el que les resulte más conveniente. Para esto se ejecutó el modelo con diferentes límites de tiempo para asegurar variedad en las opciones generadas. Se ejecutó el modelo por cinco, quince y treinta minutos. Una vez enviados los resultados, ADICAPRO planteó cambios en los datos, por lo que se volvieron a generar los *fixtures* y son estos los que se presentan en las Figuras 4.2, 4.3 y 4.4.

RONDA 1									
	Martes 17/12	Miércoles 18/12	Jueves 19/12	Viernes 20/12	Sábado 21/12	Domingo 22/12	Lunes 23/12	Jueves 26/12	Viernes 27/12
Horario 1	Los Buscadores	Zabritos	Gnomos	Herederos	Cherrys	Frenesi	Cosme y Damian	Zodiaco	Principes
Horario 2	Colibriquis	La Zafada	Klau's	Nemesis	La Descocada	Yoguis	Indigos	Magia Candombe	Los Chapitas
Horario 3	Wizards	Forever Kids	La Resiliente	Ohana	Deskara2	Wonkas	Saphirus	Bubys Bis	Zero
Horario 4	Mano a Mano	Quijotes	Mandala	Celestinos	Dulcinea	Los Pepinitos			

RONDA 2									
	Dia 1	Dia 2	Dia 3	Dia 4	Dia 5	Dia 6	Dia 7	Dia 8	Dia 9
Horario 1	Forever Kids	Nemesis	Colibriquis	Magia Candombe	Ohana	Indigos	La Resiliente	Deskara2	Wizards
Horario 2	Klau's	Gnomos	Cosme y Damian	Zabritos	La Zafada	Dulcinea	Los Buscadores	Saphirus	Frenesi
Horario 3	Herederos	Wonkas	Cherrys	Yoguis	Principes	Zodiaco	Quijotes	Zero	Bubys Bis
Horario 4	Mano a Mano	La Descocada	Los Pepinitos	Celestinos	Mandala	Los Chapitas			

LEYENDA				
Parodistas	Revistas	Murgas	Humoristas	Lubolos

Figura 4.2: *Fixture* 1 enviado a ADICAPRO. Fuente: elaboración propia.

Enviados estos tres *fixtures*, ADICAPRO planteó interés en tener más variedad, por lo que se generaron dos *fixtures* adicionales con el tiempo de ejecución limitado a tres horas y a medio minuto, resultando en los *fixtures* de la Figura 4.5 y Figura 4.6 respectivamente.

Posteriormente ADICAPRO analizó y validó los *fixtures* provistos y seleccionó la *fixture* representada en la Figura 4.6, pero planteó realizar el cambio del grupo que actúa en el primer horario del primer día, por el que actúa tercero en el mismo día, concluyendo así en el *fixture* de la Figura 4.7. Este *fixture* se utilizó durante todo el evento sin problemas. Se dejó a disposición la posibilidad de recalcular la segunda ronda del evento en caso de ser necesario, pero no fue

RONDA 1									
	Martes 17/12	Miércoles 18/12	Jueves 19/12	Viernes 20/12	Sábado 21/12	Domingo 22/12	Lunes 23/12	Jueves 26/12	Viernes 27/12
Horario 1	Forever Kids	Wizards	Zabritos	Frenesi	La Resiliente	Indigos	Cosme y Damian	Zodiaco	Los Buscadores
Horario 2	La Zafada	Herederos	Gnomos	Ohana	Yoguis	Nemesis	Principes	Magia Candombe	La Descocada
Horario 3	Colibriquis	Klau's	Wonkas	Deskara2	Cherrys	Bubys Bis	Los Chapitas	Saphirus	Zero
Horario 4	Celestinos	Mandala	Mano a Mano	Quijotes	Dulcinea	Los Pepinitos			

RONDA 2									
	Dia 1	Dia 2	Dia 3	Dia 4	Dia 5	Dia 6	Dia 7	Dia 8	Dia 9
Horario 1	Ohana	Cherrys	Principes	Magia Candombe	Colibriquis	Deskara2	Gnomos	Nemesis	Herederos
Horario 2	Wonkas	Frenesi	La Zafada	Wizards	Los Buscadores	Saphirus	La Resiliente	Klau's	Zodiaco
Horario 3	Yoguis	Mano a Mano	Indigos	Forever Kids	Cosme y Damian	Zabritos	Zero	Los Chapitas	Los Pepinitos
Horario 4	Mandala	Celestinos	Dulcinea	Bubys Bis	Quijotes	La Descocada			

LEYENDA				
Parodistas	Revistas	Murgas	Humoristas	Lubolos

Figura 4.3: *Fixture 2* enviado a ADICAPRO. Fuente: Elaboración propia.

RONDA 1									
	Martes 17/12	Miércoles 18/12	Jueves 19/12	Viernes 20/12	Sábado 21/12	Domingo 22/12	Lunes 23/12	Jueves 26/12	Viernes 27/12
Horario 1	Colibriquis	Wizards	Principes	Forever Kids	La Resiliente	Indigos	Cosme y Damian	Zodiaco	Los Buscadores
Horario 2	Frenesi	Herederos	Gnomos	Ohana	Yoguis	Nemesis	Cherrys	Magia Candombe	La Descocada
Horario 3	La Zafada	Klau's	Wonkas	Deskara2	Zabritos	Bubys Bis	Los Chapitas	Saphirus	Zero
Horario 4	Celestinos	Mandala	Mano a Mano	Quijotes	Dulcinea	Los Pepinitos			

RONDA 2									
	Dia 1	Dia 2	Dia 3	Dia 4	Dia 5	Dia 6	Dia 7	Dia 8	Dia 9
Horario 1	Frenesi	Zabritos	Ohana	Herederos	Magia Candombe	Deskara2	Gnomos	Nemesis	Cherrys
Horario 2	Wonkas	La Zafada	Colibriquis	Zodiaco	Los Buscadores	Saphirus	La Resiliente	Klau's	Mandala
Horario 3	Yoguis	Forever Kids	Principes	Cosme y Damian	Wizards	Indigos	Zero	Los Chapitas	Mano a Mano
Horario 4	Quijotes	Bubys Bis	Dulcinea	Celestinos	Los Pepinitos	La Descocada			

LEYENDA				
Parodistas	Revistas	Murgas	Humoristas	Lubolos

Figura 4.4: *Fixture 3* enviado a ADICAPRO. Fuente: Elaboración propia.

solicitado por la organización.

Todos los *fixtures* fueron generados utilizando **AMPL Version 20240606 (MSVC 19.40.33811.0, 64-bit) CPLEX 22.1.1** y se corrieron utilizando una computadora con **12th Gen Intel(R) Core(TM) i5-12400F 4.40 GHz** y **32GB de RAM 2400mhz**.

En la Tabla 4.1 se detalla el valor de la función objetivo de las *fixtures* generadas, y también se detalla el valor objetivo por ronda.

	RONDA 1								
	Martes 17/12	Miércoles 18/12	Jueves 19/12	Viernes 20/12	Sábado 21/12	Domingo 22/12	Lunes 23/12	Jueves 26/12	Viernes 27/12
Horario 1	Frenesi	Ohana	Wizards	Cherrys	Principes	La Resiliente	Zabritos	Forever Kids	Magia Candombe
Horario 2	Klau's	Wonkas	Deskara2	Los Buscadores	Saphirus	Yoguis	Cosme y Damian	La Descocada	Zodiaco
Horario 3	Herederos	Colibriquis	La Zafada	Gnomos	Indigos	Nemesis	Los Chapitas	Zero	Dulcinea
Horario 4	Mano a Mano	Celestinos	Mandala	Quijotes	Los Pepinitos	Bubys Bis			

	RONDA 2								
	Dia 1	Dia 2	Dia 3	Dia 4	Dia 5	Dia 6	Dia 7	Dia 8	Dia 9
Horario 1	Indigos	Nemesis	Cosme y Damian	Colibriquis	Zodiaco	Gnomos	Los Buscadores	Herederos	Deskara2
Horario 2	La Zafada	Principes	Wonkas	Wizards	Forever Kids	Zabritos	Magia Candombe	Cherrys	Klau's
Horario 3	Frenesi	Mano a Mano	Yoguis	La Resiliente	Los Pepinitos	Ohana	Bubys Bis	Dulcinea	La Descocada
Horario 4	Quijotes	Celestinos	Mandala	Saphirus	Los Chapitas	Zero			

LEYENDA

Parodistas	Revistas	Murgas	Humoristas	Lubolos
------------	----------	--------	------------	---------

Figura 4.5: *Fixture 4* enviado a ADICAPRO. Fuente: Elaboración propia.

	RONDA 1								
	Martes 17/12	Miércoles 18/12	Jueves 19/12	Viernes 20/12	Sábado 21/12	Domingo 22/12	Lunes 23/12	Jueves 26/12	Viernes 27/12
Horario 1	Gnomos	Herederos	Wizards	Cherrys	Principes	Cosme y Damian	Indigos	Zabritos	Colibriquis
Horario 2	Wonkas	Klau's	Deskara2	Nemesis	Ohana	Zodiaco	Magia Candombe	Los Chapitas	Zero
Horario 3	Forever Kids	La Resiliente	Frenesi	La Zafada	Yoguis	Los Buscadores	Dulcinea	Los Pepinitos	Saphirus
Horario 4	Mano a Mano	Mandala	Quijotes	Celestinos	La Descocada	Bubys Bis			

	RONDA 2								
	Dia 1	Dia 2	Dia 3	Dia 4	Dia 5	Dia 6	Dia 7	Dia 8	Dia 9
Horario 1	Nemesis	Los Buscadores	Frenesi	La Resiliente	Forever Kids	Deskara2	Zodiaco	Ohana	Magia Candombe
Horario 2	Wonkas	Klau's	Principes	Dulcinea	Wizards	Indigos	Colibriquis	Mandala	Zabritos
Horario 3	Herederos	Yoguis	Gnomos	Cherrys	Cosme y Damian	La Zafada	Los Pepinitos	Zero	Mano a Mano
Horario 4	Quijotes	Celestinos	Bubys Bis	Los Chapitas	La Descocada	Saphirus			

LEYENDA

Parodistas	Revistas	Murgas	Humoristas	Lubolos
------------	----------	--------	------------	---------

Figura 4.6: *Fixture 5* enviado a ADICAPRO. Fuente: Elaboración propia.

RONDA 1									
	Martes 17/12	Miércoles 18/12	Jueves 19/12	Viernes 20/12	Sábado 21/12	Domingo 22/12	Lunes 23/12	Jueves 26/12	Viernes 27/12
Horario 1	Forever Kids	Herederos	Wizards	Cherrys	Principes	Cosme y Damian	Indigos	Zabritos	Colibriquis
Horario 2	Wonkas	Klau's	Deskara2	Nemesis	Ohana	Zodiaco	Magia Candombe	Los Chapitas	Zero
Horario 3	Gnomos	La Resiliente	Frenesi	La Zafada	Yoguis	Los Buscadores	Dulcinea	Los Pepinitos	Saphirus
Horario 4	Mano a Mano	Mandala	Quijotes	Celestinos	La Descocada	Bubys Bis			

RONDA 2									
	Dia 1	Dia 2	Dia 3	Dia 4	Dia 5	Dia 6	Dia 7	Dia 8	Dia 9
Horario 1	Nemesis	Los Buscadores	Frenesi	La Resiliente	Forever Kids	Deskara2	Zodiaco	Ohana	Magia Candombe
Horario 2	Wonkas	Klau's	Principes	Dulcinea	Wizards	Indigos	Colibriquis	Mandala	Zabritos
Horario 3	Herederos	Yoguis	Gnomos	Cherrys	Cosme y Damian	La Zafada	Los Pepinitos	Zero	Mano a Mano
Horario 4	Quijotes	Celestinos	Bubys Bis	Los Chapitas	La Descocada	Saphirus			

LEYENDA

Parodistas	Revistas	Murgas	Humoristas	Lubolos
------------	----------	--------	------------	---------

Figura 4.7: *Fixture* final utilizado por ADICAPRO. Fuente: Elaboración propia.

Fixture	Valor objetivo total	Valor objetivo ronda 1	Valor objetivo ronda 2	Tiempo de procesamiento (m)
1	2.52	1.02	1.50	5
2	1.59	0.80	0.79	15
3	1.24	0.80	0.44	30
4	0.63	0.31	0.32	180
5	5.42	3.67	1.75	0.5

Tabla 4.1: Detalle de la función objetivo para los *fixtures* provistos a ADICAPRO.

4.2. Método AG



Figura 4.8: Foto de la agrupación Dulcinea. Fuente: ADICAPRO.

Los algoritmos genéticos son un tipo de algoritmo no determinista caracterizados por estar inspirados en la evolución genética de diferentes poblaciones como forma de buscar soluciones. A grandes rasgos, según (Katoch, Chauhan, y Kumar, 2021), un algoritmo genético cuenta con las siguientes etapas, que se presentarán en más detalle a continuación.

Generación de cromosomas iniciales: En el contexto de algoritmos genéticos, cromosoma se refiere a una potencial solución. En el primer paso se busca generar múltiples cromosomas diferentes. A esto se llama población. Idealmente, los cromosomas se generan con algún tipo de criterio para facilitar llegar a una solución final.

Cálculo de fitness: Se le dice *fitness* de un cromosoma al resultado de una función que se utiliza para determinar qué tan bueno es un cromosoma como solución al problema que interesa solucionar. La función particular depende del problema. En esta etapa se calcula el *fitness* de cada cromosoma.

Selección de población: Una vez se ha evaluado el *fitness*, su principal función es servir como criterio de selección para decidir qué cromosomas se conservarán en la población. El objetivo es mantener aquellos con mayor aptitud para formar parte de la siguiente generación y, así, favorecer la mejora progresiva de la solución. Esto puede hacerse de diversas formas, como seleccionar directamente un porcentaje de los mejores individuos, o bien asignar a cada cromosoma una probabilidad de “supervivencia” proporcional a su *fitness*.

Recombinación: En la etapa de recombinación se utilizan los cromosomas seleccionados en la etapa anterior para generar nuevos cromosomas que agregar a la población con la intención de mejorar el *fitness*.

Una vez se pasa por estas etapas se vuelve a la etapa de cálculo de *fitness* y repite hasta alcanzar un *fitness* deseado o por cierta cantidad de iteraciones.

4.2.1. Generación de cromosomas iniciales

La generación de cromosomas iniciales constituye una etapa fundamental en los algoritmos genéticos, dado que afecta directamente la diversidad genética de la población y, por consiguiente, la eficacia del proceso evolutivo (Goldberg, 1989; Eiben y Smith, 2003). Una población inicial bien diseñada puede facilitar una exploración más efectiva del espacio de soluciones y contribuir a una convergencia más rápida hacia regiones prometedoras del óptimo global.

Existen múltiples enfoques para la generación de la población inicial. Las estrategias más simples recurren a la generación aleatoria de individuos, favoreciendo la diversidad pero sin garantizar la viabilidad de las soluciones respecto a las restricciones del problema. En cambio, los métodos guiados o heurísticos incorporan conocimiento específico del dominio para construir soluciones iniciales que respeten, total o parcialmente, dichas restricciones (Bäck, 1996). Este último enfoque puede reducir significativamente el tiempo necesario para alcanzar soluciones de alta calidad, a costa de una menor dispersión inicial en el espacio de búsqueda. Se experimentó con ambas formas de generar las soluciones iniciales, las cuales se presentan en esta sección.

4.2.2. Inicialización aleatoria

Cada ronda del evento puede representarse como una secuencia de agrupaciones, donde cada posición en la secuencia corresponde a un día y una franja horaria específicos. Una estrategia simple para generar cromosomas iniciales consiste en tomar una permutación aleatoria de todas las agrupaciones y asignarlas secuencialmente a los espacios disponibles en la ronda. Este enfoque fue útil como punto de partida para experimentar con versiones simplificadas del problema, ya que permite generar rápidamente poblaciones diversas sin incorporar conocimiento del dominio. Sin embargo, esta forma de generación presenta limitaciones cuando se incorporan las restricciones completas del problema. En particular, uno de los requerimientos centrales es que cada agrupación debe actuar exactamente una vez en cada ronda. Si el proceso de generación aleatoria no controla esta condición, pueden producirse soluciones inválidas que asignan una

misma agrupación en múltiples días o franjas horarias, o que dejan agrupaciones sin programar. Esto afecta la calidad de la población inicial y puede dificultar significativamente el proceso evolutivo, al requerir mecanismos de reparación o penalización para tratar estas violaciones. En consecuencia, aunque esta estrategia garantiza una alta diversidad, no resulta adecuada cuando se busca generar poblaciones iniciales viables en escenarios con restricciones estrictas.

4.2.3. Inicialización por secciones

Como alternativa a la generación aleatoria, en este trabajo se implementó una estrategia de inicialización guiada, basada en la división del cromosoma en secciones lógicas que reflejan la estructura del evento. Este enfoque tiene por objetivo construir cromosomas iniciales que respeten, desde su creación, algunas de las restricciones impuestas por el problema.

En primer lugar, se identifican agrupaciones que deben participar exclusivamente en ciertas posiciones específicas del cronograma, aquellas que deben abrir o cerrar una jornada. Estas agrupaciones se asignan directamente a los días y horarios correspondientes, asegurando el cumplimiento de estas restricciones estructurales.

A continuación, se consideran agrupaciones sujetas a restricciones de disponibilidad temporal (por ejemplo: Aquellas que no pueden presentarse en los primeros días del evento). Estas son ubicadas en jornadas posteriores mediante un reordenamiento controlado, que mantiene la distribución equilibrada de agrupaciones por día.

Finalmente, se completa la construcción del cromosoma distribuyendo aleatoriamente el resto de las agrupaciones entre los espacios restantes, respetando la capacidad diaria de presentaciones. Esta fase introduce un componente de variabilidad que favorece la diversidad de la población inicial, sin comprometer la viabilidad de las soluciones.

Este procedimiento logra un balance entre diversidad y factibilidad, generando cromosomas que pueden ser evaluados sin penalizaciones desde el inicio del proceso evolutivo. Además, al incorporar conocimiento del dominio, se mejora la eficiencia del algoritmo al reducir el número de generaciones necesarias para alcanzar soluciones de calidad.

4.2.4. Cálculo de fitness

En los algoritmos genéticos, la función de *fitness* cuantifica la calidad de una solución dentro del espacio de búsqueda. Su diseño guía el proceso evolutivo al influir en la selección y reproducción de individuos. Tradicionalmente, la función de *fitness* refleja directamente el valor de la función objetivo que se desea optimizar, pero en problemas con restricciones se acostumbra a incorporar penalizaciones para desincentivar soluciones inviables (Mitchell, 1998; Bäck, 1996).

En este trabajo, se implementó una función de *fitness* en la cual un menor valor indica una mejor solución, invirtiendo así el criterio clásico de maximización.

El cálculo se realiza en dos etapas conceptuales:

1. **Penalización por violación de restricciones:** se recorre la solución y se contabiliza cuántas veces se incumple cada una de las restricciones del problema. Estas violaciones se suman como penalizaciones aditivas al valor total del *fitness*. En particular, se presta atención a las restricciones duras que no están garantizadas por la estructura del cromosoma ni por el operador de cruce.
2. **Penalización por alejamiento del valor objetivo ideal:** se calcula el interés total, ponderación, acumulado de los conjuntos asignados, y se lo compara con un valor ideal. Este valor ideal corresponde al interés de los conjuntos de una ronda si todos los conjuntos aportan el mismo interés.

Esta estrategia busca priorizar la factibilidad de las soluciones por encima de la optimización estricta de la función objetivo. De este modo, se favorece la evolución hacia soluciones válidas antes de refinar la calidad de las mismas.

4.2.5. Selección de población

La selección de individuos para reproducción es la etapa del algoritmo genético donde se determina cuáles soluciones tienen mayor probabilidad de transmitir sus características a la siguiente generación. Una técnica comúnmente utilizada es la *selección por ruleta*, que asigna probabilidades de selección proporcionales al *fitness* de cada individuo (Goldberg, 1989).

Dado que en este trabajo se utiliza una función de *fitness* que debe minimizarse, se aplicó una variante conocida como *ruleta con fitness invertido*. En este esquema, a cada individuo se le asigna un valor proporcional a $1/(f + \varepsilon)$, donde f es su valor de *fitness* y ε es una constante positiva pequeña que evita divisiones por cero.

Los valores invertidos se normalizan dividiendo por su suma total, generando así una distribución de probabilidad sobre la población. La selección se realiza de forma aleatoria con reposición, utilizando esta distribución para determinar qué individuos serán progenitores en la próxima generación.

Este enfoque balancea la explotación de las mejores soluciones (que tienen mayor probabilidad de ser seleccionadas) con la exploración del espacio de búsqueda (ya que incluso los individuos menos aptos tienen una probabilidad no nula de ser elegidos), ayudando a evitar la convergencia prematura del algoritmo.

4.2.6. Recombinación

El operador de recombinación, también conocido como *crossover*, permite combinar información genética de dos cromosomas progenitores para generar un nuevo individuo. Su objetivo es explotar combinaciones prometedoras de soluciones parciales, promoviendo la transferencia de bloques genéticos de alto rendimiento entre generaciones (Goldberg, 1989). Durante el desarrollo del

método AG se experimentó con distintos mecanismos de recombinación, los cuales se describen a continuación.

Crossover uniforme

El *uniform crossover* es uno de los esquemas más simples de recombinación. Para cada gen, en este caso una agrupación asignada a un horario específico, se elige aleatoriamente cuál de los dos progenitores aporta el valor, con una probabilidad uniforme. De esta forma, el nuevo cromosoma contiene una mezcla de genes tomados de ambos padres (Umbarkar y Sheth, 2015).

Este operador fue elegido inicialmente por su facilidad de implementación. Sin embargo, presenta un problema en nuestro contexto: no garantiza la unicidad de las agrupaciones dentro de una ronda, lo que genera soluciones no viables. Se intentó mitigar este inconveniente penalizando aquellas soluciones que presentaran agrupaciones repetidas, pero esta aproximación resultó ineficiente y perjudicial para la calidad general del proceso genético.

Crossover uniforme con reparación de genes

Para subsanar el problema de unicidad en el *crossover* uniforme, se incorporó un mecanismo de reparación posterior a la recombinación. La reparación de genes según (Umbarkar y Sheth, 2015) consiste en generar un nuevo cromosoma y luego corregir los errores estructurales que puedan haberse generado durante el *crossover*. En este caso, luego de generar el cromosoma hijo, se detectaban y corregían las duplicaciones de agrupaciones dentro de cada ronda, asegurando así que cada grupo se asignara una única vez.

Si bien esta estrategia permitía generar soluciones válidas, el proceso de reparación resultaba costoso en términos computacionales y tendía a degradar la calidad de las soluciones a medida que aumentaba la cantidad de agrupaciones participantes.

Position crossover

El *position crossover*, también conocido como *two-point crossover* o, en general, *k-point crossover*, es un operador de cruce comúnmente utilizado en problemas donde el orden de los elementos es relevante, como el problema *Traveling Salesman Problem* (Umbarkar y Sheth, 2015), y como ocurre en el caso de estudio del *fixture* del Carnaval de las Promesas. Su funcionamiento consiste en seleccionar uno o más puntos de corte sobre el cromosoma, conservar los genes del primer progenitor en las posiciones delimitadas por esos cortes, y completar las posiciones restantes con los genes del segundo progenitor en el orden en que aparecen, omitiendo duplicados. Este procedimiento permite generar un descendiente que preserva parcialmente el orden y la estructura de ambos padres, respetando la restricción de unicidad de los elementos.

En el contexto del problema de estudio, donde cada individuo representa un *fixture* con dos rondas, se aplicó este operador de forma independiente para

cada una de ellas. En particular, existe una restricción entre la ronda uno y la ronda dos que impide que una misma pareja de agrupaciones actúe en la misma etapa en ambas rondas. Sin embargo, el operador de cruce no considera esta restricción al momento de generar los descendientes.

Inicialmente, se exploró una representación en la cual cada cronograma era modelado como una lista plana, ordenada etapa por etapa, de modo que los horarios de una misma etapa quedaran contiguos en el cromosoma. No obstante, esta codificación presentaba un inconveniente: al preservar un segmento del cromosoma de uno de los progenitores, se perdía la información estructural sobre qué agrupaciones abrían y cerraban cada etapa. Esto generaba una pérdida de calidad en las soluciones, ya que el operador de cruce comprometía propiedades relevantes para el cálculo del valor objetivo, provocando divergencia en lugar de convergencia.



Figura 4.9: Foto de la agrupación de parodistas 0s. Fuente: ADICAPRO.

Para mitigar este problema, se adoptó una codificación alternativa en la que los mismos horarios de diferentes etapas quedaban agrupados de forma contigua. Es decir, primero se listaban todos los primeros horarios de cada etapa,

luego todos los segundos, y así sucesivamente. Esta disposición permite que, al seleccionar un segmento para preservar durante el cruce, se conserve más adecuadamente el orden relativo de las agrupaciones que abren y cierran las jornadas, manteniendo estructuras temporales de la solución y favoreciendo la convergencia del algoritmo.

El Algoritmo 1 detalla la aplicación del operador de cruce para cada ronda por separado. Este se basa en el uso del *Order Crossover* (OX), definido en el Algoritmo 2, seguido de una mutación y una reconstrucción de la estructura matricial del cronograma.

Algorithm 1 Fixture creation

```

1: procedure FIXTUREPOSITIONCROSSOVER(parent1, parent2)
2:                                      $\triangleright$  Crossover for the first round schedule
3:   child1schedule  $\leftarrow$  OrderCrossover(parent1schedule1, parent2schedule1)
4:   mutatedChild1schedule  $\leftarrow$  Mutate(child1schedule)
5:   childschedule1  $\leftarrow$  UnflattenToMatrix(mutatedChild1schedule)
6:                                      $\triangleright$  Crossover for the second round schedule
7:   child2schedule  $\leftarrow$  OrderCrossover(parent1schedule2, parent2schedule2)
8:   mutatedChild2schedule  $\leftarrow$  Mutate(child2schedule)
9:   childschedule2  $\leftarrow$  UnflattenToMatrix(mutatedChild2schedule)
10:                                      $\triangleright$  Return the new child
11:   return new child with childschedule1 and childschedule2
12: end procedure

```

Algorithm 2 Round creation

```
1: procedure ORDERCROSSOVER(parent1, parent2)
2:    $n \leftarrow$  length of parent1
3:   child  $\leftarrow$  new array of size  $n$ , filled with nulls
4:    $\triangleright$  Step 1: Select two random crossover points
5:   start, end  $\leftarrow$  random distinct integers from 0 to  $n - 1$ 
6:   if start > end then
7:     swap(start, end)
8:   end if
9:    $\triangleright$  Step 2: Copy the segment from parent1 to the child
10:  child[start...end]  $\leftarrow$  parent1[start...end]
11:  used_elements  $\leftarrow$  a set containing elements from child[start...end]
12:   $\triangleright$  Step 3: Fill remaining slots from parent2
13:  child_idx  $\leftarrow$  0
14:  for each element in parent2 do
15:    if element is not in used_elements then
16:      while child[child_idx] is not null do
17:        child_idx  $\leftarrow$  child_idx + 1
18:      end while
19:      child[child_idx]  $\leftarrow$  element
20:      add element to used_elements
21:    end if
22:  end for
23:   $\triangleright$  Step 4: Fill any remaining gaps from parent1 (edge case)
24:  if any element in child is still null then
25:    for each element in parent1 do
26:      if element is not in used_elements then
27:        Find the first null position in child and place element there
28:      end if
29:    end for
30:  end if
31:  return child
32: end procedure
```

4.2.7. Mutación

En un algoritmo genético, el operador de mutación cumple la función de introducir variabilidad genética en la población, permitiendo así evitar la convergencia prematura hacia óptimos locales subóptimos (Banzhaf, Nordin, Keller, y Francone, 1998). De manera general, la mutación altera aleatoriamente uno o más genes de un individuo con una baja probabilidad, modificando su estructura con el objetivo de explorar nuevas regiones del espacio de búsqueda.

Este operador actúa típicamente después de la selección y el *crossover*, y su implementación depende del tipo de codificación de los cromosomas. Por ejemplo, en representaciones binarias, la mutación puede consistir en un simple cambio de bit (de cero a uno o viceversa), mientras que en codificaciones reales puede implicar la adición de una perturbación estocástica (Mitchell, 1998). En la mayoría de los casos, la probabilidad de mutación se mantiene baja (por ejemplo, del orden del 1%), ya que tasas más altas tienden a desestabilizar el proceso evolutivo, asemejándolo a una búsqueda aleatoria (Eiben y Smith, 2003).

En el contexto de esta implementación, se definió como mutación el intercambio aleatorio de agrupaciones que actúan en dos horarios distintos dentro de una misma ronda. Esta operación se aplica de forma independiente en cada una de las rondas del cronograma. Se adoptó una estrategia de mutación con tasa decreciente: se comienza con una probabilidad inicial elevada, que disminuye progresivamente hasta alcanzar un valor mínimo también configurable. Esta estrategia busca favorecer la exploración en las primeras etapas del algoritmo y priorizar la explotación en las etapas finales (Bäck, 1993).

Si bien esta técnica produjo buenos resultados en la mayoría de los casos, se detectaron situaciones en las que el algoritmo se estancaba en soluciones no factibles. En estos casos, la baja probabilidad de mutación dificultaba escapar de óptimos locales. Para enfrentar este problema, se implementó un esquema adaptativo en el que la tasa de mutación se incrementa nuevamente cuando se detecta un número determinado de generaciones sin mejoras significativas. Una vez registrada una mejora en la función objetivo, la tasa de mutación vuelve a descender gradualmente. Este enfoque se alinea con estrategias de control adaptativo propuestas en la literatura, que buscan mantener el equilibrio entre exploración y explotación en función del progreso evolutivo (Eiben, Hinterding, y Michalewicz, 1999).

4.2.8. Diversidad genética

Una vez implementado el algoritmo con los pasos y técnicas descritos previamente, se observó que, al ejecutarlo sobre los datos correspondientes a la edición 2024/2025 del concurso, algunas ejecuciones se estancaban en soluciones no factibles. Este estancamiento persistía incluso al aumentar la tasa de mutación o aplicar mutaciones más agresivas. Al analizar el valor de *fitness* de los individuos en estas poblaciones, se detectó una fuerte homogeneidad: muchos individuos presentaban valores idénticos o extremadamente similares. Un

análisis más profundo reveló que, en algunos casos, los operadores de cruce generaban exactamente los mismos individuos, lo que causaba una pérdida severa de diversidad genética.

Para mitigar esta situación, se incorporó un mecanismo basado en funciones *hash*: se calcula un identificador único para cada solución, y solo se insertan en la siguiente población aquellos individuos cuya representación no se haya generado previamente. Esta técnica permitió eliminar duplicados exactos y evitó que se propagaran clones en la población.

Sin embargo, este control no fue suficiente para mantener una diversidad genética adecuada en todos los casos. A pesar de no repetir individuos idénticos, aún se observaba una convergencia prematura hacia conjuntos de soluciones estructuralmente similares, que además no cumplían con los requisitos de factibilidad. Aunque aumentar el tamaño de la población podría contrarrestar este efecto, se comprobó que ello reducía la eficiencia del algoritmo, ya que incrementaba el tiempo de búsqueda incluso en situaciones donde la diversidad genética era satisfactoria.

Esto motivó la incorporación de una estrategia de reinicio poblacional condicionada al estancamiento del proceso, explicada a continuación.

4.2.9. Reinicio poblacional

En los algoritmos genéticos, es común emplear mecanismos de reinicio poblacional para contrarrestar la convergencia prematura hacia óptimos locales. Estos mecanismos se activan cuando no se observan mejoras significativas en la aptitud durante un número determinado de generaciones, lo cual indica estancamiento en la búsqueda. Al reiniciar total o parcialmente la población, se incrementa la diversidad genética y se promueve la exploración de nuevas regiones del espacio de soluciones (Kurniawan, Nurmala, Abdullah, y Ramli, 2023).

El algoritmo desarrollado en este trabajo incorpora un mecanismo de reinicio condicionado al estancamiento del progreso evolutivo. Específicamente, si la población no presenta mejoras sensibles en su mejor valor de aptitud tras cierto número de generaciones consecutivas, se desencadena un proceso de reinicio controlado. Aprovechando la estructura del problema que se presenta en dos rondas, durante este proceso se eligen un pequeño número de soluciones de la población anterior, seleccionadas aleatoriamente, y se generan nuevas soluciones desde cero, aunque con una pequeña probabilidad de heredar parcialmente una de las rondas de alguna de las soluciones elegidas. Este enfoque híbrido permite mantener cierta información útil del proceso evolutivo anterior, al tiempo que se favorece la introducción de diversidad mediante la creación de individuos completamente nuevos. Además, se controla la diversidad mediante el uso de funciones *hash* para evitar duplicados, como se explicó previamente. La nueva población se reordena en función de la factibilidad y calidad de las soluciones, asegurando una base sólida para continuar el proceso evolutivo con renovada variabilidad.

Algorithm 3 Algoritmo Genético

```
1:  $Poblacion\_Actual \leftarrow Generar\_Poblacion\_Inicial()$ 
2: Bucle Principal del Proceso Evolutivo
3: while no se cumpla el Criterio.de.Terminacion do
4:   for all  $cromosoma \in Poblacion\_Actual$  do
5:      $cromosoma.Fitness \leftarrow Calcular\_Fitness(cromosoma)$ 
6:   end for
7:   Detección de Estancamiento y Reinicio Poblacional
8:   if  $Hay\_Estancamiento(Poblacion\_Actual)$  then
9:      $Poblacion\_Actual \leftarrow Aplicar\_Reinicio\_Poblacional(Poblacion\_Actual)$ 
10:  end if
11:  Selección de Progenitores
12:   $Progenitores \leftarrow Seleccionar\_Progenitores(Poblacion\_Actual)$ 
13:  Creación de la nueva generación
14:   $Poblacion\_Nueva \leftarrow Lista\_Vacía$ 
15:  while tamaño de  $Poblacion\_Nueva <$  tamaño configurado do
16:     $Progenitor1, Progenitor2 \leftarrow Elegir\_Progenitores(Progenitores)$ 
17:     $Cromosoma\_Hijo \leftarrow Aplicar\_Recombinacion(Progenitor1, Progenitor2)$ 
18:     $Aplicar\_Mutacion(Cromosoma\_Hijo)$ 
19:    if  $Cromosoma\_Hijo \notin Poblacion\_Nueva$  then
20:      Añadir  $Cromosoma\_Hijo$  a  $Poblacion\_Nueva$ 
21:    end if
22:  end while
23:  Actualización de la Población
24:   $Poblacion\_Actual \leftarrow Poblacion\_Nueva$ 
25:  Detección de Estancamiento y Reinicio Poblacional
26:  if  $Hay\_Estancamiento(Poblacion\_Actual)$  then
27:     $Poblacion\_Actual \leftarrow Aplicar\_Reinicio\_Poblacional(Poblacion\_Actual)$ 
28:  end if
29: end while
30: return  $Mejor\_Cromosoma(Poblacion\_Actual)$ 
```

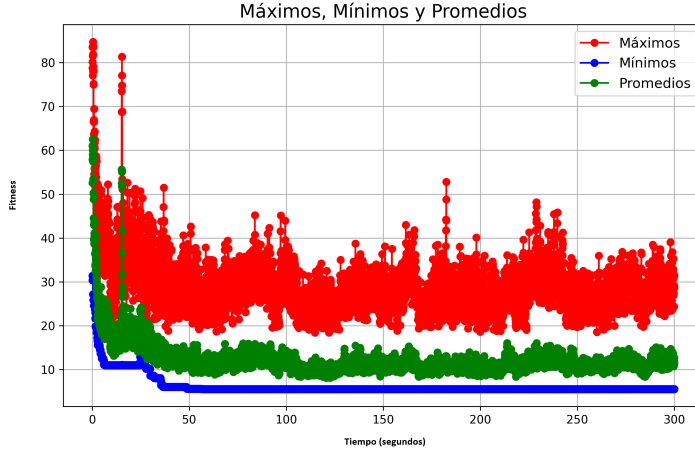


Figura 4.10: Gráfica de *fitness* en función del tiempo de la ejecución 3, instancia 1, configuración 1

Se puede observar en la Figura 4.10 correspondiente a una de las ejecuciones del AG cómo el *fitness* queda estancado, y el momento del reinicio poblacional parcial cuando empeoran la *fitness* de la peor solución de la población y el *fitness* promedio de toda la población. Posterior a este reinicio se observa cómo estos valores y el *fitness* de la mejor solución mejoran al acercarse a cero luego de encontrar soluciones factibles.

4.3. Desarrollo de aplicación

Parte del proyecto consistió en generar una aplicación que sirva como herramienta para generar *fixtures* a partir del algoritmo genético generado. Bajo esta propuesta, se desarrolló una aplicación de escritorio sencilla que permita a ADICAPRO generar *fixtures* sin necesidad de tener conocimiento técnico.

Se planteó el desarrollo de una aplicación simple que permita ejecutar el algoritmo genético sobre datos proporcionados por el usuario, y que el usuario pueda modificar la *fixture* y verificar el estado de las reglas conforme se hagan modificaciones.

4.3.1. Tecnologías utilizadas

Para el desarrollo de la aplicación se empleó el framework Electron, una herramienta que permite crear aplicaciones de escritorio multiplataforma (Windows, macOS y Linux) utilizando tecnologías de desarrollo web como HTML, CSS y JavaScript ([OpenJS Foundation, 2024](#)). Electron combina el motor de renderizado de Chromium con el entorno de ejecución Node.js, lo que posibilita

acceder tanto a la interfaz gráfica como a funcionalidades del sistema operativo desde un mismo entorno de desarrollo. Esta elección tecnológica permite empaquetar la aplicación como un ejecutable nativo, facilitando su distribución e instalación sin necesidad de navegadores externos.

La decisión de utilizar Electron estuvo motivada por la intención de integrar React como biblioteca principal para el desarrollo del frontend. React es una biblioteca declarativa y basada en componentes para construir interfaces de usuario interactivas, desarrollada por Meta ([Meta Platforms, Inc., 2024](#)). Su arquitectura modular facilita la reutilización de código, el mantenimiento de la aplicación y la implementación de vistas altamente dinámicas. Además, React cuenta con un ecosistema maduro que incluye herramientas como React Router y Redux, lo que permitió al equipo trabajar con mayor agilidad y aprovechar conocimientos previos.

En cuanto a la lógica de generación de *fixtures*, como se presenta en el Capítulo 4, se desarrolló un algoritmo genético implementado en el lenguaje Python ([Python Software Foundation, 2024](#)), elegido por su sintaxis clara y su amplio soporte para bibliotecas científicas y de optimización. Este componente se encarga de procesar los datos proporcionados por el usuario en formato de planilla y generar automáticamente una propuesta de programación que cumple con las restricciones definidas.

Para facilitar la comunicación entre el algoritmo y la interfaz gráfica, se desarrolló un microservidor con Flask, un microframework web de Python que permite construir servicios web ligeros de manera rápida y flexible ([Pallets Projects, 2024](#)). Flask se encarga de exponer el algoritmo como una API HTTP que puede ser consumida por el frontend de React desde la aplicación de escritorio. Esta arquitectura basada en servicios favorece la separación de responsabilidades y permite que el módulo de generación de *fixtures* funcione de forma independiente.

Finalmente, con el objetivo de simplificar la ejecución del microservidor en entornos donde Python no está instalado, se utilizó la herramienta PyInstaller, la cual permite empaquetar scripts de Python como ejecutables autónomos ([PyInstaller Development Team, 2024](#)). De esta manera, el usuario final puede utilizar la aplicación sin preocuparse por instalar dependencias adicionales o configurar un entorno de ejecución, lo que mejora la portabilidad y la experiencia de uso.

4.3.2. Aplicación

La aplicación fue diseñada para facilitar su uso sin formación técnica, por lo que se definió un mecanismo simple y accesible para la carga de datos. El usuario ingresa la información necesaria a través de una planilla en formato de hoja de cálculo, en la cual debe completar detalles como el nombre de las agrupaciones, su categoría, su ponderación, posibles restricciones particulares y otras características relevantes para la generación del calendario. Para evitar errores y garantizar una estructura coherente, se proporcionará al usuario una planilla modelo o *template*, que funcionará como guía. Esta plantilla incluye los

encabezados y formatos requeridos, así como ejemplos que ayudan a comprender cómo completar cada campo correctamente. Una vez completada la planilla, el usuario la puede ingresar al sistema como se muestra en la Figura 4.11.

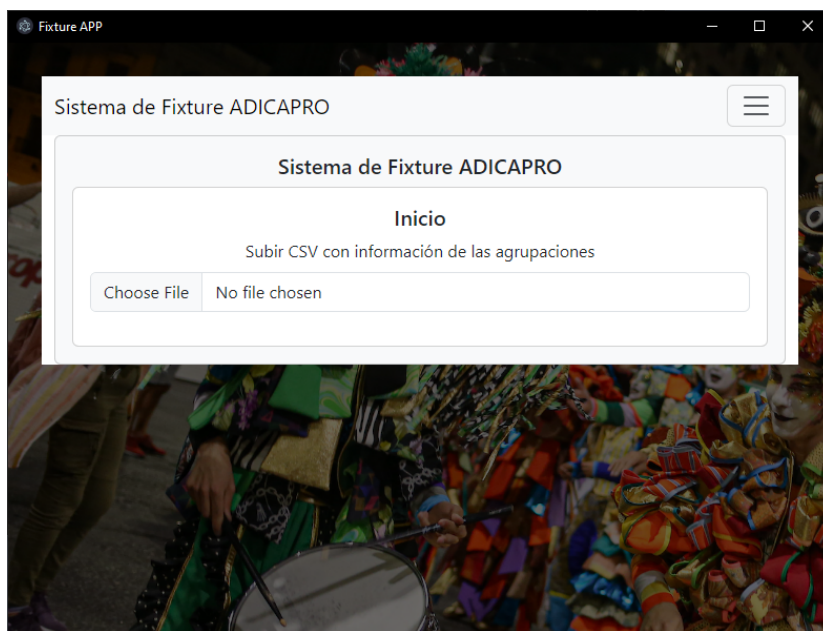


Figura 4.11: Captura de la pantalla de inicio. Fuente: Elaboración propia.

Luego de cargar la planilla, el usuario accede a una interfaz donde puede seleccionar qué restricciones desea aplicar en la generación del *fixture* como se muestra en la Figura 4.12. Estas restricciones van a estar activadas por defecto, pero se pueden desactivar según las características específicas del evento. Este enfoque permite adaptar el modelo a distintas situaciones y otorga flexibilidad para personalizar los criterios según las necesidades de cada edición del concurso. La posibilidad de activar o desactivar restricciones específicas también facilita la experimentación con distintos escenarios, sin necesidad de modificar manualmente los datos o el código del sistema.

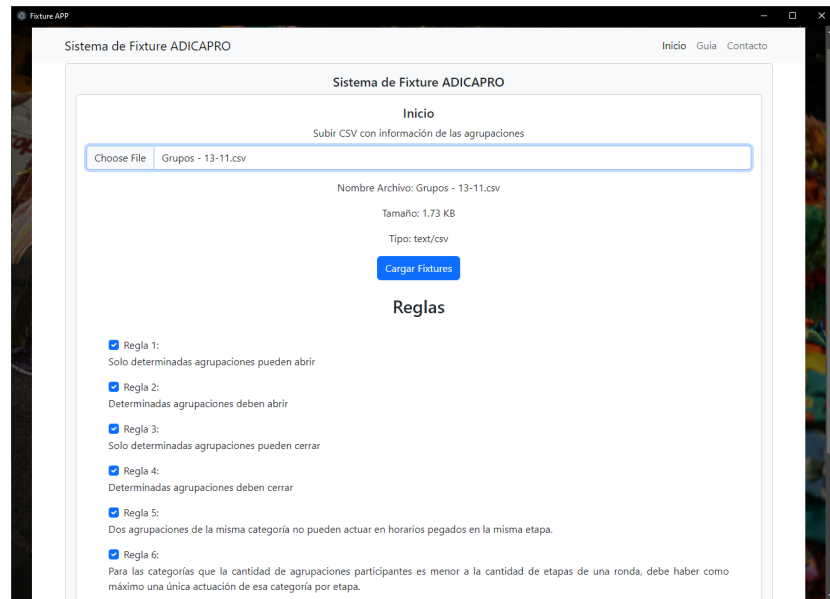


Figura 4.12: Captura de la pantalla de selector de reglas. Fuente: Elaboración propia.

Una vez generado el *fixture*, la aplicación permite al usuario realizar modificaciones manuales sobre la propuesta inicial. La interfaz está diseñada para reflejar en tiempo real cómo afectan estos cambios al cumplimiento de las reglas previamente seleccionadas. A medida que el usuario mueve agrupaciones a distintos horarios o fechas, la pantalla le indica claramente qué restricciones se mantienen satisfechas y cuáles dejan de cumplirse, como se muestra en la Figura 4.13.

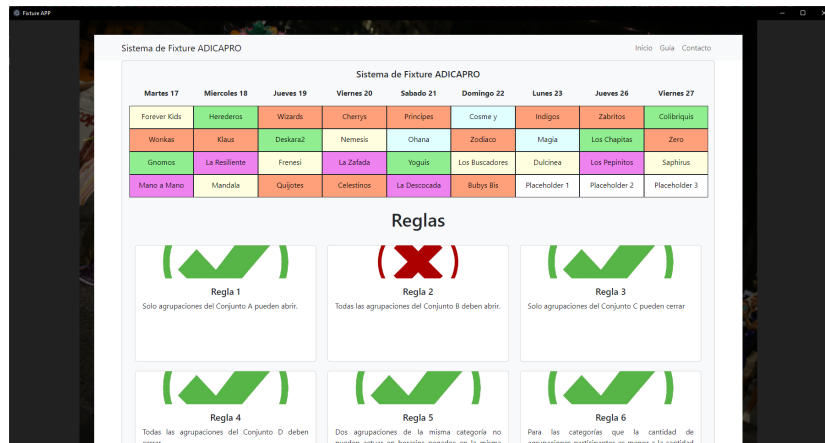


Figura 4.13: Captura de la pantalla de *fixture*. Fuente: Elaboración propia.

Es importante señalar que la aplicación no pretende imponer una solución única y definitiva, sino que actúa como una herramienta de apoyo a la toma de decisiones. Ofrece un *fixture* base que cumple los requerimientos técnicos desde la cual los organizadores pueden explorar alternativas, comparar escenarios y construir, de forma asistida, un calendario final que refleje tanto las restricciones técnicas como las prioridades institucionales o culturales del evento.

Capítulo 5

Experimentación numérica

En este capítulo se presenta la experimentación realizada para evaluar el método AG utilizando el modelo en CPLEX como referencia. Se detallan las distintas configuraciones utilizadas en las pruebas y se realiza un análisis de los resultados.

5.1. Análisis comparativo

Para evaluar el método AG se realizó una comparación con CPLEX. Se generaron casos de prueba sintéticos para examinar distintas condiciones de forma tal de poder comparar el rendimiento de ambos métodos.

Es de interés medir para cada ejecución el tiempo a la primera solución factible y a la solución final, *fitness* y valor de la función objetivo de la primera solución factible y de la solución final, cantidad de generaciones y tiempo a primera solución factible y a la solución final. La medida de generaciones y *fitness* solo aplican al algoritmo genético.

La función objetivo corresponde a la suma de la peor desviación en valor absoluto del promedio de ponderación de popularidad de una etapa para cada una de las rondas. A modo de ejemplo, tomemos el caso base donde el promedio de la ponderación de popularidad de cada agrupación es de 5,3, esto hace que la popularidad de una etapa de cuatro horarios sea de 21,2. Si la popularidad total de la etapa con máxima desviación a este promedio en la ronda uno es de 23 y en la ronda dos es de 20, esto da un valor objetivo de 3.

A ambos métodos se les asigna un límite de tiempo y se evalúan. En el caso del método AG, como es un algoritmo no determinista, se corre diez veces con diferentes semillas para cada juego de datos para contar con cierta certeza de que los resultados obtenidos no son solo azar.

Las pruebas son realizadas utilizando **AMPL Version 20240606 (MSVC 19.40.33811.0, 64-bit)** **CPLEX 22.1.1** y se corrieron en una computadora con **12th Gen Intel(R) Core(TM) i5-12400F 4.40 GHz** y **32GB de RAM 2400mhz**.

5.1.1. Configuraciones de datos

Para tener una comparación útil, se generaron diferentes configuraciones de datos usadas para generar instancias de prueba con las que probar ambos métodos de resolución del problema. Estas se presentan en esta sección.

Para todas las configuraciones se parte del caso real del Carnaval de las Promesas año 2024/2025. Lo que interesa es ver dentro de casos cercanos al real cómo se comportan ambos métodos de solución del problema. El caso real consta de dos rondas de nueve etapas cada una, y cada etapa se compone de cuatro horarios, salvo las últimas tres etapas, que se componen de tres horarios cada una. En cada horario debe actuar una única agrupación, y cada agrupación debe actuar exactamente una vez por ronda. De estas agrupaciones, siete de ellas son revistas, doce parodistas, seis humoristas, cinco murgas y tres lubolos, totalizando treinta y tres agrupaciones. De estas agrupaciones, ocho deben cerrar y diez deben abrir. Es importante destacar que no se deben repetir categorías para una misma etapa, salvo para el caso particular de la categoría de parodistas, donde pueden participar hasta dos parodistas en una misma etapa. La distribución de las ponderaciones de popularidad de las agrupaciones se observan en la Figura 5.1.

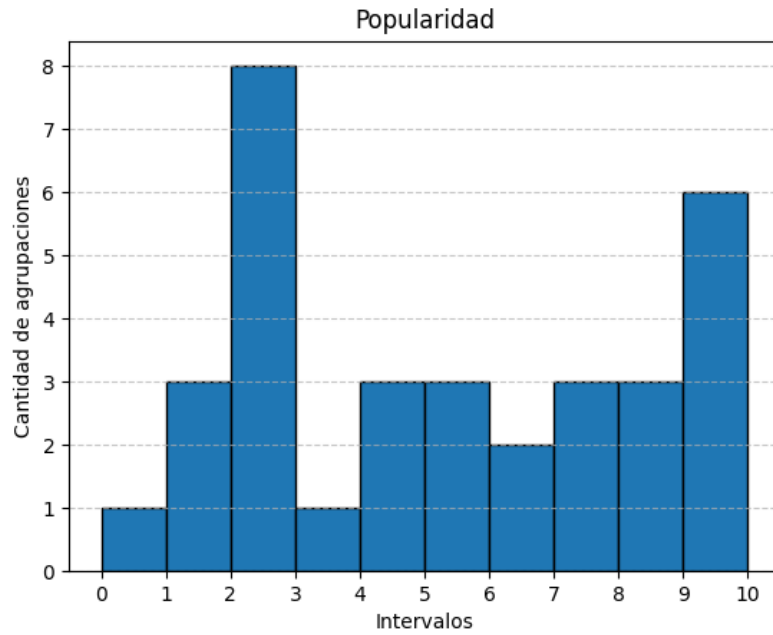


Figura 5.1: Distribución de ponderación de popularidad de las agrupaciones.

Configuración 1

Para la primera configuración de datos se generan instancias a partir del caso real, pero se modifican cuáles agrupaciones deben abrir y cerrar. Se busca evaluar qué efecto genera llevar al límite de la viabilidad las reglas que restringen qué grupos pueden cerrar y abrir. En estas instancias la cantidad de grupos que deben abrir y cerrar son exactamente dieciocho, la misma cantidad que la cantidad de etapas entre las dos rondas, por lo que cada una de ellas debe abrir o cerrar exactamente una vez. Para la generación de cada una de las instancias de esta configuración se toman las treinta y tres agrupaciones originales y se le asigna a cada una una nueva categoría con la misma distribución que el caso base, se le asigna una nueva ponderación de popularidad siguiendo una distribución uniforme entre uno y diez, se eligen al azar cuatro que no pueden actuar los primeros cuatro días y dos que no pueden algún día particular, también al azar. Para las restricciones de quienes abren y cierran se eligen dieciocho agrupaciones que deben abrir, posteriormente se obliga a las restantes quince agrupaciones que no fueron asignadas para abrir a cerrar, y se agregan tres de las agrupaciones que deben abrir a que además deban cerrar.

Configuración 2

Para la configuración dos se generan nuevamente las instancias de forma aleatoria con las mismas cantidades de agrupaciones en cada categoría y agrupaciones que pueden cerrar y abrir, pero se asigna la ponderación de popularidad a los grupos de forma desbalanceada, teniendo pocos grupos muy populares y el resto poco populares, como se puede observar en la Figura 5.2 correspondiente a la distribución de ponderaciones de uno de los casos de la configuración de datos dos. Esta distribución de la ponderación corresponde a una distribución Beta con los parámetros $a = 0,1$ y $b = 0,4$ que se genera en el rango $[0, 1]$, que luego se multiplica para que se corresponda en el intervalo $[0,10)$.

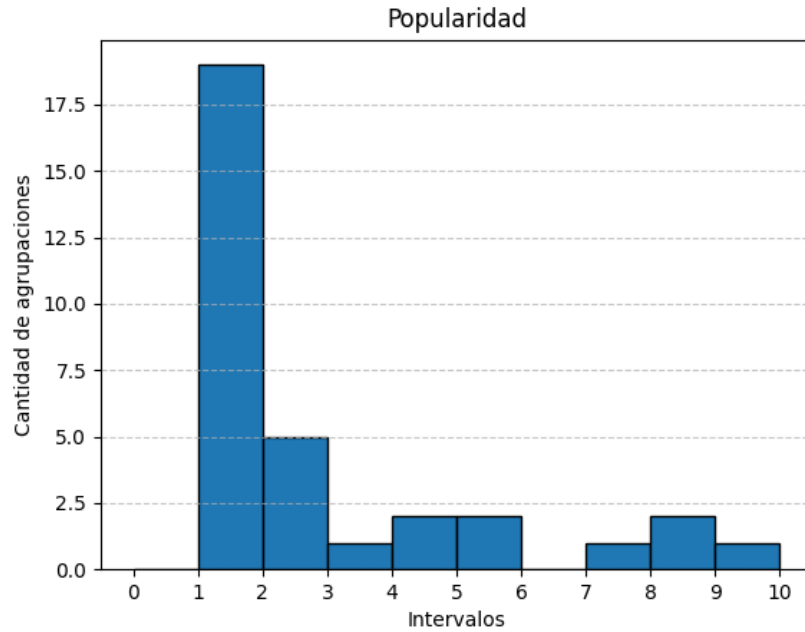


Figura 5.2: Distribución de ponderación de la instancia 1 de la configuración 2.

Configuración 3

Es de interés ver cómo se comportan los métodos para solucionar el problema cuando crece la entrada. Para esto se generan instancias a partir del caso real, pero se agregan más agrupaciones, subiendo a cuarenta y dos agrupaciones totales. Las categorías y ponderaciones de popularidad de estas nuevas agrupaciones siguen una distribución uniforme. Para que el problema siga siendo consistente, es necesario que esas agrupaciones puedan participar, para esto se puede o bien agregar más horarios por etapa o más etapas.

En este caso se eligió agregar más horarios por etapa, haciendo que cada etapa tenga cinco horarios en los que puedan actuar las agrupaciones. Nuevamente, para el caso de los últimos días, actuarán menos agrupaciones, quedando en estas etapas cuatro horarios.

Configuración 4

Para esta configuración el objetivo es el mismo que la configuración anterior, pero en este caso se eligió agregar más etapas a las rondas, esto permite evaluar nuevamente cómo se comportan los métodos ante un aumento del tamaño de entrada. En este caso se agregaron dos etapas y se busca evaluar si alguno de

los dos casos es más favorable o si alguno de los métodos es mejor que el otro ante los cambios.

Configuración 5

En esta configuración se busca probar qué tan afectados se ven los métodos cuando se restringe la factibilidad de la solución. En cada instancia de esta configuración se aumenta gradualmente la cantidad de agrupaciones parodistas hasta llegar al límite de la factibilidad, dieciocho. Para esto se modifica de forma aleatoria el tipo de agrupación de alguna de las otras agrupaciones. Para la instancia uno hay trece agrupaciones parodistas, para la instancia dos hay catorce agrupaciones parodistas y así sucesivamente hasta la instancia cinco con dieciocho agrupaciones parodistas.

En la última instancia de esta configuración, al contar con dieciocho parodistas, obliga a que cada etapa del concurso deba tener dos horarios con parodistas. Esto fuerza a que la única forma de poder cumplir con las restricciones es que se alterne una agrupación de parodistas con otra de otro tipo para cada etapa, lo cual restringe el espacio de soluciones.

5.2. Resultados

A continuación se muestran los resultados globales de todas las pruebas realizadas como promedio de los resultados obtenidos para todas las configuraciones. Los resultados completos se encuentran en el Apéndice 6.2. En la Tabla 5.1 se encuentra para cada método el promedio obtenido de correr todas las pruebas para cada una de las métricas que nos interesa evaluar. Se registró el tiempo que tomó en cada instancia de cada configuración llegar a la primera solución y el valor de la función objetivo que se busca minimizar. Además, se registró para ambos métodos el tiempo que tomó llegar a la solución final dentro del tiempo determinado y la función objetivo de dicha solución.

	CPLEX	Método AG
Tiempo a primera solución (s)	22.2	49.6
Valor objetivo de primera solución	20.1	20.6
Tiempo a solución final(s)	218.1	178.7
Valor objetivo de solución final	7.2	5.0

Tabla 5.1: Comparación de resultados de pruebas de ambos métodos.

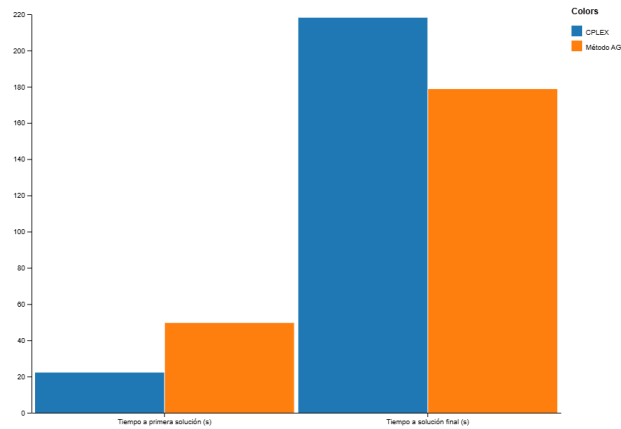


Figura 5.3: Representación de los datos de tiempo de la Tabla 5.1.

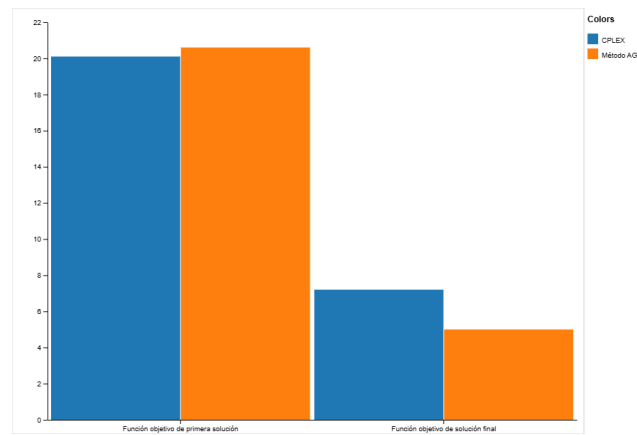


Figura 5.4: Representación de los datos de la función objetivo de la Tabla 5.1.

En la Tabla 5.2 están los resultados promedios para cada configuración ejecutada en CPLEX junto con el promedio total. Estos resultados también se pueden visualizar en la Figura 5.3 y en la Figura 5.4. En el Apéndice 6.2 se encuentran los resultados para cada instancia.

Configuración	Tiempo a primera solución (s)	Función objetivo de primera solución	Tiempo a solución final (s)	Función objetivo de solución final
1	9.1	18.1	271.6	3.3
2	9.2	30.5	213.8	3.4
3	59.1	22.8	204.3	12.9
4	25.7	16.8	112.7	13.3
5	7.8	12.4	288.1	3.0
Promedio	22.2	20.1	218.1	7.2

Tabla 5.2: Resultados de pruebas para cada configuración con CPLEX.

En la Tabla 5.3 se muestran los resultados promedios para cada configuración ejecutada con el método AG junto con el promedio total. En el Apéndice 6.2 se encuentran los resultados para cada instancia.

Configuración	Tiempo a primera solución (s)	Función objetivo de primera solución	Tiempo a solución final (s)	Función objetivo de solución final (s)
1	47.8	20.1	159.9	4.5
2	14.6	21.3	126.0	6.8
3	108.3	23.5	237.5	6.9
4	31.7	20.1	213.9	2.9
5	45.6	18.2	156.2	3.6
Promedio	49.6	20.6	178.7	5.0

Tabla 5.3: Resultados de pruebas para cada configuración del método AG.

Viendo los resultados generales presentados en la Tabla 5.1 y la Figura 5.3, se puede observar que para obtener una primera solución el método AG es más lento en promedio, requiriendo más del doble del tiempo necesario en comparación con CPLEX para generar una solución que cumpla con los requerimientos mínimos de validez. Además, el valor de la función objetivo para esta primera solución también es mejor para el caso de CPLEX. Sin embargo, cuando se comparan los datos para la solución final, el método AG resulta ser, en promedio, alrededor de un veinte por ciento más rápido que CPLEX en las pruebas realizadas, y la función objetivo es aproximadamente un treinta por ciento mejor. Debemos mencionar también que, como el método AG no es determinista, se

ejecutó cada instancia diez veces y en algunas de estas ejecuciones fue capaz de superar a CPLEX.

Un análisis más detallado por configuración, utilizando los datos de las Tablas 5.2 y 5.3, revela dinámicas particularmente interesantes sobre el comportamiento de ambos enfoques. Específicamente, en las configuraciones 3 y 4, diseñadas para incrementar el tamaño del problema combinatorio (cuarenta y dos agrupaciones con más horarios por etapa en la configuración 3, y cuarenta y dos agrupaciones con más etapas por ronda en la configuración 4), el método AG demostró una capacidad superior para encontrar soluciones finales de mayor calidad. Para la configuración 3, el método AG alcanzó un valor promedio de la función objetivo final de 6,9 (Tabla 5.3), mientras que CPLEX obtuvo 12,9 (Tabla 5.2). De manera aún más pronunciada, en la configuración 4, el método AG logró un promedio de 2,9 en la función objetivo final (Tabla 5.3), mejor que el 13,3 obtenido por CPLEX (Tabla 5.2). Estos resultados sugieren que, ante un aumento en la complejidad combinatoria, ya sea por un mayor número de actuaciones dentro de las etapas o por un incremento en la cantidad de etapas, la naturaleza exploratoria y basada en poblaciones de los algoritmos genéticos podría ofrecer ventajas para converger hacia soluciones de mejor calidad dentro del límite de tiempo establecido en las pruebas, en comparación con métodos exactos que pueden enfrentar mayores dificultades para explorar exhaustivamente espacios de búsqueda de gran magnitud.

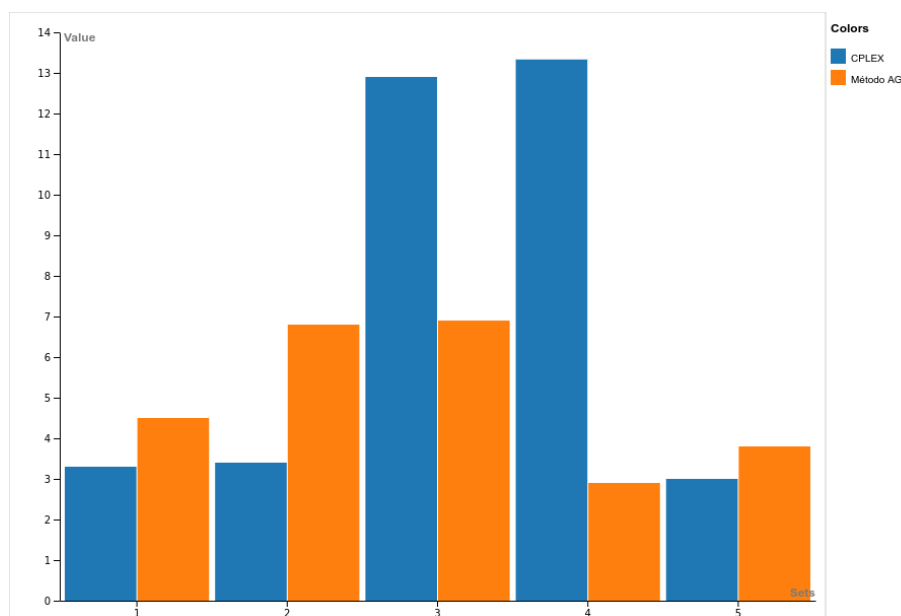


Figura 5.5: Representación de los datos comparando CPLEX y el Método AG para cada configuración.

Por otro lado, la configuración 5 introdujo un desafío diferente, al probar el comportamiento de los métodos bajo una restricción de factibilidad progresivamente más difícil, mediante el aumento de la cantidad de agrupaciones de la categoría parodistas hasta 18, lo que implicaba la necesidad de incluir dos agrupaciones de esta categoría por etapa. En este escenario, se observó un comportamiento divergente entre los dos métodos. Mientras CPLEX mantuvo una calidad de solución relativamente estable, con un promedio de función objetivo final de 3,0 (Tabla 5.2), el método AG mostró una disminución en su capacidad para encontrar soluciones de calidad e incluso para hallar soluciones factibles a medida que aumentaba el número de parodistas en las sucesivas instancias de esta configuración. El promedio de la función objetivo final para el método AG en esta configuración fue de 3,8 para los casos en los que logró llegar a una solución factible (Tabla 5.3).

Es importante notar, en línea con lo observado en la configuración 5, que el método AG no siempre obtuvo solución en todas las ejecuciones. Veintidós de las doscientas cincuenta ejecuciones totales del método AG no llegaron a obtener una solución factible. La quinta instancia de la configuración cinco, en particular, que representaba el escenario más restrictivo en cuanto a la cantidad de parodistas en el concurso, resultó ser especialmente problemática para el método AG, ya que no se encontraron resultados para ninguna de las ejecuciones, constituyendo estas casi la mitad de todas las ejecuciones en las que no hubo resultados. Las ejecuciones en las que no se obtuvo solución no se utilizaron para los promedios para no distorsionar los datos.

Capítulo 6

Conclusiones y Trabajo Futuro

6.1. Conclusiones

En este proyecto se abordó el problema de la elaboración del *fixture* para el Carnaval de las Promesas 2024/2025, aplicando algoritmos genéticos como metaheurística para resolver el problema de timetabling asociado. Se desarrolló y evaluó un modelo de optimización para el problema, incluyendo las diversas restricciones planteadas por ADICAPRO, tales como la distribución de agrupaciones por día, evitar la repetición de parejas de agrupaciones en las mismas etapas entre rondas, la asignación de agrupaciones que deben abrir y cerrar etapas, y restricciones de disponibilidad específicas.

Como punto de partida, se realizó un estudio del estado del arte de la literatura sobre metaheurísticas aplicadas a la programación de eventos artísticos y deportivos en los últimos 10 años. En los documentos analizados se pudo apreciar que la tendencia general indica que el uso de metaheurísticas ofrece una mejora, ya sea en el tiempo necesario para llegar a un resultado, en la calidad del mismo, o en varios casos, en ambas dimensiones.

A partir de estos antecedentes, se implementó el método AG para resolver el problema y se comparó con la resolución del modelo matemático en CPLEX, un solver comercial de optimización, mostrando resultados prometedores para el enfoque basado en algoritmos genéticos. Aunque CPLEX tendió a encontrar en promedio una primera solución factible más rápidamente en la mayoría de configuraciones de prueba, el método AG logró obtener soluciones finales con mejores valores de función objetivo (menor desviación promedio de la popularidad por etapa) en algunos de los casos evaluados. Esto sugiere que, si bien la complejidad del problema dificulta encontrar un óptimo global, los algoritmos genéticos son una alternativa viable para encontrar soluciones comparables con CPLEX en tiempos computacionales razonables para instancias realistas.

El desarrollo de la aplicación de escritorio fue un componente del proyecto

que permitirá a ADICAPRO generar *fixtures* sin requerir conocimiento técnico avanzado. La aplicación facilita la visualización y modificación de los *fixtures*, así como la verificación del cumplimiento de las restricciones. La interacción con ADICAPRO durante el proceso de validación y ajuste del *fixture* final para la edición 2024/2025 demostró la utilidad práctica de las herramientas y el modelo desarrollado, siendo uno de los *fixtures* generados el utilizado durante el evento.

Se identificaron desafíos durante la implementación del algoritmo genético, particularmente el estancamiento en soluciones no factibles posiblemente debido a la pérdida de diversidad genética. La incorporación de mecanismos como el reinicio poblacional y el control de duplicados mediante funciones *hash* demostró utilidad para mitigar estos problemas y mejorar la robustez del algoritmo.

El proyecto cumplió con el objetivo de aplicar una metaheurística, en este caso algoritmos genéticos, para generar *fixtures* equilibrados para el Carnaval de las Promesas, desarrollando una herramienta útil y validada por los organizadores del evento. Se encontró viable el uso de metaheurísticas para problemas de programación artística con múltiples restricciones.

6.2. Trabajo Futuro

A partir del trabajo realizado en este proyecto, se identifican diversas oportunidades de mejora tanto a nivel del modelo como de la aplicación desarrollada.

Una línea podría consistir en la exploración de otras metaheurísticas o enfoques híbridos, combinando algoritmos genéticos con técnicas como *Simulated Annealing* o *Tabu Search*. La revisión bibliográfica realizada sugiere que estos métodos pueden ofrecer mejoras respecto a la calidad de las soluciones o a los tiempos de procesamiento, por lo que su implementación y evaluación futura podría ser de interés.

En el mismo sentido, la función objetivo podría ser objeto de revisión y mejora. Explorar diferentes formulaciones que capten de forma más precisa el concepto de “equilibrio” deseado entre etapas o que consideren otros criterios relevantes para organizadores y participantes, más allá de la popularidad estimada.

Por otro lado, el rendimiento del algoritmo genético creemos que podría mejorarse. Esto incluye el ajuste fino de los operadores evolutivos como selección, cruce y mutación, así como la implementación de mecanismos de control de diversidad poblacional, como el reinicio adaptativo. Estas mejoras podrían mejorar aspectos relevantes como evitar estancamientos prematuros y mejorar el desempeño del algoritmo, especialmente en escenarios con un número creciente de agrupaciones o restricciones más complejas.

En relación al rendimiento computacional, sería deseable considerar la re-implementación del algoritmo, o de sus partes más críticas, en un lenguaje compilado como C++. Al tratarse de un algoritmo actualmente implementado en Python, un lenguaje interpretado, esta modificación podría traducirse en mejoras en cuanto a eficiencia y tiempos de ejecución.

También se visualiza como una posible mejora la paralelización de la ejecución del algoritmo genético. Esto podría lograrse, por ejemplo, aplicando operadores genéticos a diferentes individuos de la población de forma concurrente, aprovechando los núcleos múltiples de los procesadores actuales. En esta misma línea, otra estrategia sería permitir la ejecución simultánea de múltiples instancias independientes del algoritmo, cada una con diferentes semillas aleatorias. Finalizada la ejecución, la aplicación podría seleccionar la mejor solución encontrada entre todas las instancias lanzadas, incrementando así la probabilidad de obtener soluciones de calidad en tiempos razonables.

En conjunto, estas posibles líneas de trabajo abren la puerta a una mejora continua del método propuesto, tanto en términos de eficiencia computacional como en la calidad de las *fixtures* generadas para la planificación de este evento artístico.

Referencias

- Babaei, H., Karimpour, J., y Hadidi, A. (2015). A survey of approaches for university course timetabling problem. *Computers & Industrial Engineering*, 86, 43–59. doi: <https://doi.org/10.1016/j.cie.2014.11.010>
- Banzhaf, W., Nordin, P., Keller, R. E., y Francone, F. D. (1998). *Genetic Programming: An Introduction: On the Automatic Evolution of Computer Programs and Its Applications*. San Francisco: Morgan Kaufmann. (ISBN: 978-1-55860-510-7)
- Bäck, T. (1993). Optimal mutation rates in genetic search. En S. Forrest (Ed.), *Proceedings of the fifth international conference on genetic algorithms* (pp. 2–8). San Francisco: Morgan Kaufmann. (ISBN: 978-1-55860-299-1)
- Bäck, T. (1996). *Evolutionary Algorithms in Theory and Practice*. Oxford University Press. (ISBN: 978-0-19-756092-1) doi: <https://doi.org/10.1093/oso/9780195099713.001.0001>
- Cardozo, A., Guido, C., y Machin, J. C. (2020). *Métodos cuantitativos para la programación de espectáculos artísticos: Una aplicación en el Carnaval Uruguayo* (Proyecto de grado, Facultad de Ingeniería, Universidad de la República, Uruguay). doi: <https://doi.org/10.36561/ING.20.3>
- de Werra, D. (1985). An introduction to timetabling. *European Journal of Operational Research*, 19(2), 151–162. doi: [https://doi.org/10.1016/0377-2217\(85\)90167-5](https://doi.org/10.1016/0377-2217(85)90167-5)
- de Werra, D. (1997). The combinatorics of timetabling. *European Journal of Operational Research*, 96(3), 504–513. doi: [https://doi.org/10.1016/S0377-2217\(96\)00111-7](https://doi.org/10.1016/S0377-2217(96)00111-7)
- Documentación de CPLEX. (2024). *Tipos de problemas que puede resolver CPLEX*. <https://www.ibm.com/docs/es/icos/22.1.2?topic=cplex-types-problems-solved>. (Último acceso: 26/05/2025)
- Eiben, A., Hinterding, R., y Michalewicz, Z. (1999). Parameter control in evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 3(2), 124–141. doi: <https://doi.org/10.1109/4235.771166>
- Eiben, A., y Smith, J. (2003). *Introduction to Evolutionary Computing*. Springer.
- Even, S., Itai, A., y Shamir, A. (1976). On the complexity of time table and multi-commodity flow problems. *SIAM Journal on Computing*, 184–193. doi: <https://doi.org/10.1109/SFCS.1975.21>

- Fink, A., y Voss, S. (1999). Generic metaheuristics application to industrial engineering problems. *Computers & Industrial Engineering*, 37(1–2), 281–284. doi: [https://doi.org/10.1016/S0360-8352\(99\)00074-1](https://doi.org/10.1016/S0360-8352(99)00074-1)
- Fourer, R., Gay, D. M., y Kernighan, B. W. (2003). Ampl: A modeling language for mathematical programming (2da ed.) [Manual de software informático]. Duxbury Press / Brooks/Cole–Thomson Learning.
- Goldberg, D. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley. doi: <https://doi.org/10.1023/A:1022602019183>
- Karp, R. M. (2008, mayo). George Dantzig's impact on the theory of computation. *Discrete Optimization*, 5(2), 174–185.
- Katoch, S., Chauhan, S. S., y Kumar, V. (2021). A review on genetic algorithm: past, present, and future. *Multimedia Tools and Applications*, 80(5), 8091–8126. doi: <https://doi.org/10.1007/s11042-020-10139-6>
- Kurniawan, T., Nurmala, F., Abdullah, A. H., y Ramli, K. (2023). MRDPGA: A multiple restart dynamic population genetic algorithm for scheduling road traffic. *Journal of Electrical Systems and Information Technology*, 10(1), 12. doi: <https://doi.org/10.1186/s43067-023-00099-w>
- Meta Platforms, Inc. (2024). *React (versión 18)* [Software]. <https://reactjs.org>. (Último acceso: 26/05/2025)
- Mitchell, M. (1998). *An Introduction to Genetic Algorithms*. MIT Press.
- OpenJS Foundation. (2024). *Electron (versión 33)* [Software]. <https://www.electronjs.org>. (Último acceso: 26/05/2025)
- Osman, I. H., y Laporte, G. (1996). Metaheuristics: A bibliography. *Annals of Operations Research*, 63(5), 511–623. doi: <https://doi.org/10.1007/BF02125421>
- Pallets Projects. (2024). *Flask (versión 2.3)* [Software]. <https://flask.palletsprojects.com/>. (Último acceso: 26/05/2025)
- Portal de DAECPU. (2025). *DAECPU*. <https://archive.ph/20160629170118/https://www.daecpu.org.uy/>. (Último acceso: 26/05/2025)
- Portal de IBM. (2025). *IBM ILOG CPLEX Optimizer*. <https://www.ibm.com/products/ilog-cplex-optimization-studio/cplex-optimizer>. (Último acceso: 26/05/2025)
- Portal Institucional de la Intendencia de Montevideo. (2025). *Carnaval de las Promesas*. <https://montevideo.gub.uy/carnaval-de-las-promesas>. (Último acceso: 26/05/2025)
- Portal Intendencia de Montevideo. (2025). *Carnaval y llamadas*. <http://montevideo.gub.uy/carnaval-y-llamadas>. (Último acceso: 26/05/2025)
- PyInstaller Development Team. (2024). *PyInstaller (versión 6)* [Software]. <https://www.pyinstaller.org>. (Último acceso: 26/05/2025)
- Python Software Foundation. (2024). *Python (versión 3.7.0)* [Software]. <https://www.python.org>. (Último acceso: 26/05/2025)
- Rajwar, K., Deep, K., y Das, S. (2023). An exhaustive review of the metaheuristic algorithms for search and optimization: taxonomy, applica-

- tions, and open challenges. *Artificial Intelligence Review*, 1–71. doi: <https://doi.org/10.1007/s10462-023-10470-y>
- Real Academia Española. (2024a). *Evento*. <https://dle.rae.es/evento>. (En *Diccionario de la lengua española* (23.^a ed.). Último acceso: 26/05/2025)
- Real Academia Española. (2024b). *Heurístico, ca*. <https://dle.rae.es/heurstico>. (En *Diccionario de la lengua española* (23.^a ed.). Último acceso: 26/05/2025)
- Sadeeq, H. T., y Abdulazeez, A. M. (2023). Metaheuristics: A Review of Algorithms. *International Journal of Online and Biomedical Engineering*, 19(09), 142–164. doi: <https://doi.org/10.3991/ijoe.v19i09.39683>
- Umbarkar, A. J., y Sheth, P. D. (2015, octubre). Crossover operators in genetic algorithms: A review. *ICTACT Journal on Soft Computing*, 06(01), 1083–1092. doi: <http://dx.doi.org/10.21917/ijsc.2015.0150>
- Yang, X.-S. (2014). Introduction to Algorithms. En *Nature-Inspired Optimization Algorithms* (pp. 1–21). Elsevier. doi: <https://doi.org/10.1016/B978-0-12-416743-8.00001-4>

Metaheurísticas aplicadas a la programación de eventos Estado del Arte



UNIVERSIDAD
DE LA REPÚBLICA
URUGUAY



FACULTAD DE
INGENIERÍA

Metaheurísticas aplicadas a la programación de eventos

Estado del Arte

Nicolás Vassallo, Germán Costabel, Damián
Sancristóbal

Supervisores

Pedro Piñeyro
Héctor Cancela

Montevideo, 24 de mayo de 2025



Metaheurísticas aplicadas a la programación de eventos

Estado del Arte por Nicolás

Vassallo, Germán Costabel, Damián Sancristóbal tiene licencia

[CC Atribución 4.0.](#)

Índice general

1. Introducción	3
2. Conceptos claves	5
2.1. Metaheurísticas	5
2.2. Simulated Annealing	7
2.3. Threshold Acceptance	7
2.4. Tabu Search	8
2.5. Particle Swarm Optimization	8
2.6. Genetic Algorithm	9
2.7. Timetabling	10
2.8. Competencia Internacional de Timetabling	10
3. Revisión sistemática	13
3.1. Proceso	13
3.1.1. Procedimiento de búsqueda y selección	14
3.2. Clasificación de la Literatura	14
4. Análisis	17
4.1. Académicos	17
4.1.1. Examination Timetabling Problem	17
4.1.2. University Course Timetabling	19
4.2. Deportivos	22
4.2.1. Round Robin	22
4.2.2. Carry over effect	23
4.2.3. Traveling tournament	24
4.3. Otros	25
4.3.1. Médicos	25
4.3.2. Cross Subsidization	26
4.3.3. Workforce scheduling	26
4.3.4. Sin aplicación	26
5. Conclusiones	29
Referencias	31

Capítulo 1

Introducción

El propósito de este documento es relevar y analizar la literatura sobre el uso de metaheurísticas para la resolución de problemas de programación de eventos. Un evento, definido por la Real Academia Española, es un "suceso importante y programado, de índole social, académica, artística o deportiva" (*Diccionario de la lengua española*, 23.^a ed., 2014). Los problemas de programación de evento refieren entonces a la coordinación y planificación de estos eventos asignando el horario de cada uno de ellos considerando las restricciones de cada evento y otros factores.

Esta clase de problemas es muy común tanto en el ámbito académico como en el deportivo y administrativo, por lo cual existe una gran cantidad de literatura al respecto (Rajwar, Deep, y Das, 2023). Sin embargo, las restricciones de cada problema particular condiciona el tipo de resolución que se puede utilizar (Siddiqui y Arshad Raza, 2021). Se buscó clasificar los problemas abordados en documentos de programación de eventos y encontrar que tipo de metaheurísticas son utilizadas. Según (Rajwar y cols., 2023) una heurística se refiere a una forma de guiar la búsqueda de soluciones dentro del espacio soluciones. Según (de Werra, 1985, 1997), los problemas de programación de eventos presentan una alta complejidad computacional, lo que puede hacer que encontrar una solución óptima requiera un tiempo excesivo de cómputo. En los últimos diez años ha habido un gran interés y desarrollo de múltiples métodos de resolución que utilizan metaheurísticas para resolver estos problemas de optimización (Rajwar y cols., 2023). El uso de metaheurísticas busca obtener resultados adecuados, cercanos al óptimo, en tiempos razonables.

El resto del documento está organizado de la siguiente forma: En el Capítulo 2 se abordan ciertos conceptos claves que se entienden fundamentales para comprender el contenido del documento. En el Capítulo 3 se describe en detalle el proceso de búsqueda llevado a cabo, sobre la aplicación de técnicas metaheurísticas a los problemas de programación de eventos. En el Capítulo 4 se analizan los documentos encontrados, habiendo sido previamente clasificado de

acuerdo al problema que abordan. Finalmente, en Capítulo 5 se presentan las conclusiones de este trabajo.

Capítulo 2

Conceptos claves

El objetivo de este capítulo es presentar y explicar algunos conceptos y términos básicos que son necesarios para facilitar la comprensión el resto del documento. Estos conceptos son fundamentales, ya que proporcionan el marco teórico y práctico sobre el cual se basan los análisis que se discutirán más adelante. Al presentar estos términos se busca establecer una base común sobre la cual se apoyará la explicación de las técnicas y métodos analizados más adelante. Este capítulo también introducirá algunas de las metaheurísticas que aparecieron mas veces en los documentos relevados y que se utilizarán posteriormente en el análisis, detallando sus principales características y cómo se aplican en la resolución de problemas.

También se busca explicar el problema de la programación de eventos, normalmente conocido como Timetabling Problem, las distintas facetas del problema y de porqué es un problema computacionalmente desafiante.

2.1. Metaheurísticas

El término metaheurística se deriva de la combinación de dos términos griegos. Heurística del griego εὕρισκειν que significa «hallar, inventar», y el prefijo *meta* que significa «mas allá» o «a un nivel superior».

Según (Rajwar y cols., 2023) una heurística se refiere a una estrategia que busca guiar las búsquedas dentro del espacio de soluciones para problemas complejos, logrando encontrar soluciones aceptables en un tiempo razonable. Dada la complejidad de muchos problemas, puede resultar poco practico explorar todo el espacio de soluciones. En cambio, una heurística se enfoca en identificar soluciones aceptables y de calidad dentro de un plazo de tiempo razonable. Si bien usualmente no se garantiza que se obtendrán soluciones óptimas la heurística busca un equilibrio entre eficiencia y tiempo. Se supone que este enfoque proporcione soluciones de buena calidad, algunas de las cuales podrían acercarse a

ser óptimas, aunque sin la certeza de lograr la mejor solución posible.

Si bien no hay un consenso en la literatura sobre la diferencia entre los términos heurística y metaheurística y muchas veces se utilizan estos términos de manera intercambiable, en (X.-S. Yang, 2014) se le llama a todos los algoritmos estocásticos que utilizan aleatorización y que además utilizan algún método de búsqueda local como metaheurísticas.

Otros autores definen a metaheurísticas en un sentido jerárquico respecto a heurísticas de menor nivel. En (Osman y Laporte, 1996) definen formalmente metaheurística como “... un proceso de generación iterativo que guía una heurística subordinada combinando inteligentemente diferentes conceptos para explorar y explotar el espacio de búsqueda; las estrategias de aprendizaje se utilizan para estructurar la información con el fin de encontrar soluciones eficientemente cercanas a las óptimas”. En (Fink y Voss, 1999) los autores definen metaheurística como “... un proceso maestro iterativo que guía y modifica las operaciones de heurísticas subordinadas para producir eficientemente soluciones de alta calidad. Puede manipular una única solución completa (o incompleta) o una colección de soluciones en cada iteración. Las heurísticas subordinadas pueden ser procedimientos de alto (o bajo) nivel, o una simple búsqueda local, o simplemente un método de construcción”.

Los algoritmos metaheurísticos se basan en dos pasos esenciales: la intensificación y la diversificación, también conocidas como explotación y exploración. La diversificación se enfoca en generar una variedad de soluciones para explorar el espacio de búsqueda de manera global, y en general es aquí donde entra el aspecto aleatorio del algoritmo. La intensificación se centra en profundizar en áreas locales, aprovechando la información obtenida de soluciones prometedoras en esas regiones, a través de un procedimiento de búsqueda local. Juntas, estas estrategias intentan acercarse al óptimo global. Mientras la diversificación, mediante la aleatorización, ayuda a evitar que las soluciones se queden atrapadas en óptimos locales y fomenta la variedad, la intensificación asegura que se aprovechen las mejores soluciones encontradas (X.-S. Yang, 2014). Igualmente es importante encontrar un equilibrio adecuado para que un algoritmo funcione de manera óptima. Si se pone demasiado énfasis en la exploración, la búsqueda puede extenderse demasiado, lo que resultará en una convergencia lenta. En cambio, si se prioriza excesivamente la explotación en detrimento de la exploración, es posible que el algoritmo converja más rápido, pero con menos posibilidades de encontrar el verdadero óptimo global (Sadeeq y Abdulazeez, 2023). La combinación efectiva de ambos enfoques facilita la búsqueda de la mejor solución posible en un espacio de búsqueda extenso. (X.-S. Yang, 2014)

2.2. Simulated Annealing

Simulated Annealing (SA) es una metaheurística inspirada en el proceso físico del recocido, donde un material, como el metal o el vidrio, se calienta a altas temperaturas para luego enfriarse lentamente. Durante este proceso físico, los átomos del material ganan libertad de movimiento cuando está caliente y pueden reorganizarse en un estado de menor energía. A medida que se enfría, los átomos se asientan en una estructura más estable, que representa un estado de mínima energía. La idea central de la metaheurística es comenzar con una solución inicial y mejorarla iterativamente, permitiendo en ciertos momentos aceptar soluciones que empeoran el objetivo (Kirkpatrick, Gelatt, y Vecchi, 1983).

Inicialmente, se asigna una temperatura alta al sistema, lo que permite aceptar soluciones peores con una probabilidad determinada. Esta probabilidad depende de la diferencia entre la nueva solución y la actual, y disminuye conforme baja la temperatura. Esto permite que inicialmente se exploren mas soluciones, incluso algunas que no mejoran el objetivo, para evitar quedarse atrapado en óptimos locales.

El esquema de enfriamiento controla cómo disminuye la temperatura. Generalmente se busca que la temperatura disminuya mas lentamente al inicio, para permitir una mayor exploración en esta etapa. A medida que la temperatura disminuye, la probabilidad de aceptar una solución peor se reduce, volviendo al algoritmo más selectivo. Esto hace que el algoritmo explore libremente al principio y se concentre en refinar las mejores soluciones al final, dependiendo de cómo se controle el enfriamiento (Leite, Melício, y Rosa, 2019).

2.3. Threshold Acceptance

Threshold Acceptance (TA) es una metaheurística de optimización similar a SA, pero con una diferencia clave: en lugar de aceptar soluciones peores basándose en una probabilidad que depende de una "temperatura" una función exponencial, TA utiliza un umbral explícito que define qué tan peor puede ser una nueva solución para ser aceptada. Este umbral disminuye a lo largo de las iteraciones, permitiendo inicialmente una mayor flexibilidad en la aceptación de soluciones peores, pero volviéndose más estricto con el tiempo.

El algoritmo comienza con una solución inicial x_0 y un umbral inicial T_0 . En cada iteración, una nueva solución x_{new} se genera a partir de x_{current} aplicando una perturbación o cambio pequeño. La diferencia en la calidad entre ambas soluciones se calcula como:

$$\Delta f = f(x_{\text{new}}) - f(x_{\text{current}})$$

Donde $f(x)$ es la función objetivo a minimizar. La solución x_{new} se acepta si:

$$\Delta f \leq T_k$$

Es decir, si el deterioro de la solución no supera el umbral actual T_k . A lo largo de las iteraciones, el valor de T_k disminuye, restringiendo progresivamente la aceptación de soluciones peores, hasta que finalmente el algoritmo se comporta como una búsqueda local estricta (Dueck y Scheuer, 1990).

2.4. Tabu Search

Tabu Search (TS) es una metaheurística que utiliza memoria para evitar que el algoritmo se quede atrapado en mínimos locales o ciclos. Esta memoria se implementa a través de la llamada "lista tabú", que restringe ciertos movimientos para que no se vuelvan a considerar durante un número determinado de iteraciones, impidiendo que el algoritmo regrese a soluciones previas.

El proceso comienza con una solución inicial a partir de la cual se generan soluciones vecinas aplicando pequeños cambios. El objetivo en cada iteración es seleccionar la mejor solución vecina posible, incluso si empeora el valor de la función objetivo. Sin embargo, para evitar que el algoritmo se quede atascado en un ciclo de soluciones ya visitadas, la lista tabú almacena movimientos recientes o características de las soluciones previas, prohibiendo que sean repetidos durante un número determinado de iteraciones (Glover, 1989).

2.5. Particle Swarm Optimization

Particle Swarm Optimization (PSO) es una metaheurística basada en el comportamiento colectivo de sistemas naturales como bandadas de aves o cardúmenes de peces. Cada partícula en el enjambre representa una posible solución al problema y tiene una posición y una velocidad en el espacio de búsqueda. El movimiento de las partículas está guiado por dos factores: la mejor posición individual que cada partícula ha encontrado (pbest) y la mejor posición global que cualquier partícula del enjambre ha alcanzado (gbest) (Kennedy y Eberhart, 1995).

La velocidad de cada partícula se actualiza en función de estos dos valores, así como de un componente inercial que evita cambios rápidos en la trayectoria. La actualización de la velocidad de una partícula i en el instante $t+1$ se realiza de acuerdo a la fórmula:

$$v_i(t+1) = w \cdot v_i(t) + c_1 \cdot r_1 \cdot (pbest_i - x_i(t)) + c_2 \cdot r_2 \cdot (gbest - x_i(t))$$

donde $v_i(t)$ es la velocidad de la partícula i en el instante t , w es el peso inercial, c_1 y c_2 son los coeficientes de aceleración personal y social, respectivamente, r_1

y r_2 son factores aleatorios entre 0 y 1, $pbest_i$ es la mejor posición personal de la partícula i , y $gbest$ es la mejor posición global alcanzada por cualquier partícula.

La actualización de la posición de la partícula está dada por:

$$x_i(t+1) = x_i(t) + v_i(t+1)$$

El balance entre exploración y explotación es crucial. Un valor alto del parámetro inercial w favorece la exploración del espacio de búsqueda, mientras que un valor bajo promueve la explotación local alrededor de las mejores soluciones encontradas. A lo largo de las iteraciones, las partículas tienden a agruparse cerca de la mejor solución global, con sus velocidades reduciéndose a medida que el enjambre converge (Poli, 2008).

2.6. Genetic Algorithm

Genetic Algorithm (GA) es una metaheurística inspirada en los principios de la evolución natural y la genética. Esta metaheurística opera sobre una población de soluciones candidatas, llamadas individuos. Cada individuo representa una posible solución al problema, codificada como un cromosoma. El algoritmo busca mejorar la calidad de las soluciones a través de las siguientes operaciones: selección, cruzamiento, mutación y reemplazo.

- La selección elige a los individuos que pasarán a la siguiente generación basándose en su calidad, que mide qué tan buena es una solución en comparación con otras. Esto imita el proceso de "supervivencia del más apto".
- El cruzamiento ocurre cuando dos individuos de la población se seleccionan para intercambiar partes de sus cromosomas y generar nuevos individuos, combinando características de las soluciones padres para crear nuevas soluciones hijas.
- La mutación introduce pequeñas modificaciones aleatorias en los cromosomas de algunos individuos para mantener la diversidad genética en la población y evitar la convergencia prematura a óptimos locales.
- Los nuevos individuos generados por cruzamiento y mutación reemplazan a los individuos más viejos o menos aptos de la población, creando una nueva generación de soluciones.

El proceso de selección, cruzamiento, mutación y reemplazo se repite hasta que se cumple un criterio de parada, como alcanzar un número máximo de generaciones o encontrar una solución suficientemente buena (Reeves, 2010).

2.7. Timetabling

Los problemas de Timetabling son un tipo específico de problemas de planificación que se enfocan en asignar ciertos eventos a franjas horarias dadas ciertas restricciones. Estos problemas son NP-hard(Perzina, 2007) y, por lo tanto, no existe un algoritmo conocido para resolverlo en tiempo polinómico en una máquina determinista. Esta asignación de ciertos eventos a franjas horarias está sujeta a ciertas restricciones, las cuales se suelen dividir en restricciones duras (hard) y restricciones blandas (soft). Las restricciones duras son aquellas que son necesarias se cumplan en una solución para que sea válida. Mientras que las restricciones blandas son aquellas que son deseables pero si no se cumplen la solución aun puede ser válida y, en general, se intenta minimizar la cantidad de restricciones blandas no cumplidas(Imran Hossain, Akhand, Shuvo, Siddique, y Adeli, 2019a). Por ejemplo, en un problema de asignación de clases a profesores un restricción dura sería que ningún profesor tenga mas de una clase en el mismo horario. En el mismo problema una restricción blanda podría ser que no haya horarios libres mayores a media hora entre las clases asignadas al mismo profesor en el mismo día.

2.8. Competencia Internacional de Timetabling

La Competencia Internacional de Timetabling (ITC por sus siglas en inglés) es una competencia internacional patrocinada por la Conferencia Internacional de Prácticas y Teoría del Timetabling automatizado (PATAT por sus siglas en ingles) (EURO Working Group on Automated Timetabling, 2024). En esta competencia se busca motivar la investigación en problemas de Timetabling desde un aspecto teórico y práctico. Además provee un gran conjunto de pruebas de referencia con las cuales medir y comparar algoritmos.

Si bien la temática de esta competencia orbita el ámbito académico y no al ámbito artístico o deportivo, resulta relevante ya que gran parte de la investigación académica utiliza los conjuntos de pruebas de estas competencias en la comparación de metaheurísticas.

En 2002-2003 se realizó la Primer Competencia Internacional de Timetabling (ITC2002) con la consigna de planificar cursos universitarios (University Course Timetabling Problem o UCTP, por sus siglas en ingles), generando un fixture semanal que permita que todos los estudiantes puedan asistir a todos los cursos. El conjunto de datos para la evaluación de los algoritmos se generó de forma sintética (Kostuch, 2005).

La segunda competencia se realizó en 2007 (ITC2007), basándose en el éxito de la competencia anterior y con el foco en atraer el interés académico a la resolución de problemas del "mundo real". Con el objetivo de reducir la

distancia entre la investigación teórica y la aplicación práctica. Esta competencia tuvo tres categorías, planificación de exámenes (Examination Timetabling), planificación de cursos post-inscripción (Post-enrolment Course Timetabling) y planificación de cursos basada en currículum (Curriculum-based Course Timetabling) (Mccollum, Mcmullan, Burke, Parkes, y Qu, 2007).

La tercer competencia realizada en 2011 (ITC2011) se enfocó en la planificación de exámenes de escuelas secundarias. El conjunto de datos pruebas utilizado corresponde a datos de 50 escuelas secundarias reales, recompiladas en un formato estandarizado XML (Post y cols., 2011).

La cuarta instancia se celebró en 2019 (ITC2019), regresando a la temática de cursos universitarios. Introdujo el uso de un complejo modelado del problema permitiendo utilizar problemas de varias universidades utilizando el software no comercial UniTime. En el sitio de dicho software se pueden encontrar los datos de prueba utilizados y sirven como una gran referencia a la hora de evaluar algoritmos de Timetabling (Müller, Rudová, y Müllerová, 2024).

La quinta competencia se realizó en 2021 (ITC2021), cambiando el foco a motivar investigación en la programación de eventos deportivos. La consigna de esta competencia consistió en la programación de torneos Round-Robin dobles con 16 a 20 equipos y restricciones duras y blandas (Van Bulck y Goossens, 2023).

A lo largo de este documento se mencionarán estas competencias, en particular haciendo referencia a los datos de prueba utilizados comúnmente para poner a prueba algoritmos de Timetabling.

Capítulo 3

Revisión sistemática

En este capítulo se presentará el proceso de búsqueda y clasificación de la literatura sobre metaheurísticas aplicadas a programación de eventos.

3.1. Proceso

La búsqueda se realizó en las colecciones SienceDirect, SpringerLink y Scopus. Para ello se confeccionó una consulta que nos permitiera obtener un numero razonable de artículos relacionados a nuestro sujeto de estudio.

Se filtró y registró en una hoja de calculo los resultados de la búsqueda, para cada documento encontrado, su titulo, problema abordado, resumen, autores, año, revista, y editorial.

Se utilizó la siguiente consulta que busca utilizar los términos por los que se conoce a este tipo de problemas y escenarios en los que se suelen presentar y porque se llegó a un numero adecuado de documentos:

```
(fixture OR schedule OR timetabling) AND (metaheuristic)
AND (sports OR artistic) AND NOT (production)
```

En particular, se excluyeron documentos relacionados al Scheduling de cadenas de producción. Este tipo de problema se centra en la optimización de cadenas de producción con el objetivo reducción de costos y maximizaron de ganancias(Aminzadegan, Tamannaei, y Fazeli, 2021).

Se descartaron los problemas mencionados ya que el objeto de estudio corresponde a la programación de eventos sociales y deportivos que son diferentes a los de producción.

3.1.1. Procedimiento de búsqueda y selección

En los buscadores de las colecciones SpringerLink, SienceDirect y Scopus se recopilaron todos los documentos obtenidos con la consulta mencionada, limitando la búsqueda a los últimos 10 años.

- Se obtuvieron la siguiente cantidad de documentos
 - SpringerLink: 57
 - SienceDirect: 151
 - Scopus: 15
- Se removieron los duplicados, llegando a 221 documentos.
- Se clasificaron los documentos según que tipo de problema solucionan.
- Se removieron documentos que no interesan al objeto de estudio, esto fueron problemas de big data, reconocimiento de imagen, logística, cadena de producción, teoría de juegos y médicos no relacionados a programación de eventos. A su vez se descartan documentos que no apliquen metaheurísticas. Este filtro llevó la cantidad de documentos a 181.
- Se descartaron los problemas menos relacionados y con menor cantidad de documentos. Se mantienen los documentos de Timetabling Problem (25), Sport Scheduling (19) y Resource Managment (9) para un resultante de 53 documentos.
- Se leyeron y analizaron en mayor profundidad todos los documentos que quedaron tras los filtros anteriores.

Un vez se completa este proceso el total de documentos analizados fue de 53. Dentro de los documentos recopilados se descartaron algunos por que a pesar de ser resultados de la consulta trabajaban sobre problemas que no era de interés abordar en este documento, en particular se descartaron 3 documentos que trabajaban sobre problemas de Job Shop, de líneas de producción y programación lineal. También fue necesario descartar 10 documentos por que a pesar que parecían trabajar sobre los temas correctos no se pudo acceder a su versión completa.

Los 40 documentos resultantes abordan diversas metaheurísticas, como se ve en la Figura 3.1

3.2. Clasificación de la Literatura

Se clasificaron los documentos obtenidos de acuerdo al tipo de problema abordado y no a la metaheurística utilizada para resolverlo como lo mostramos en la Figura 3.2. Basándonos en la clasificación de los problemas deportivos encontrada en (C. Ribeiro, Urrutia, y Werra, 2023) y académicos en (Al-Ashhab y Alghamdi, 2018). Se consideró mas importante la similitud del problema que la metaheurística particular aplicada en su resolución.

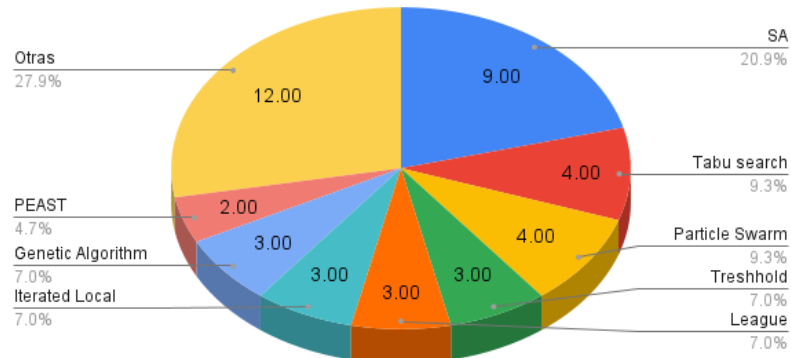


Figura 3.1: Metaheurísticas abordadas en los documentos relevados.
Fuente: elaboración propia.

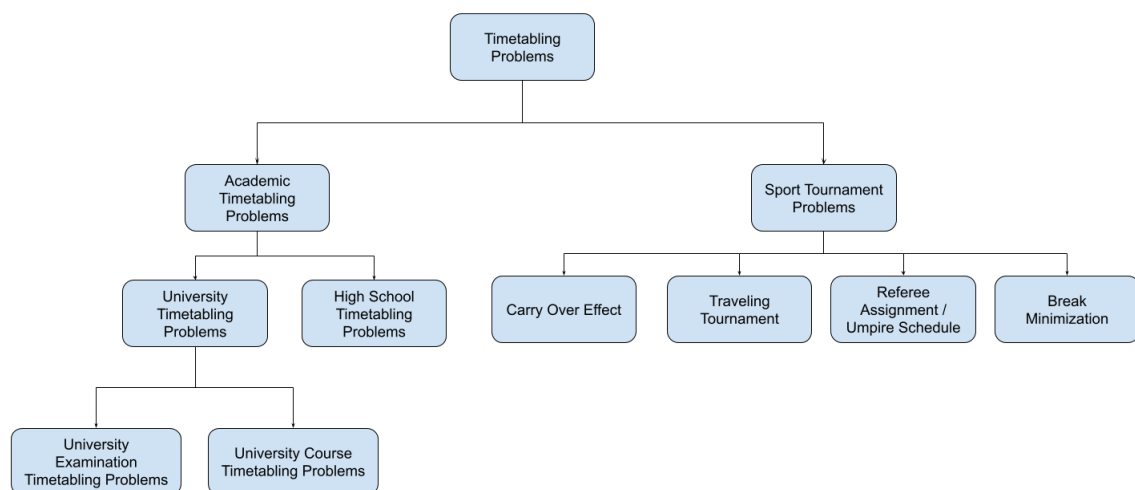


Figura 3.2: Clasificación de documentos utilizada.
Fuente: elaboración propia.

Capítulo 4

Análisis

En este capítulo se examinan los documentos recopilados durante la revisión sistemática, organizados según los diferentes tipos de problemas de programación de eventos abordados en la literatura. Se analizan y comparan las metaheurísticas empleadas en estos estudios, así como los resultados obtenidos, con el objetivo de comprender su aplicación en distintos contextos y bajo diversas restricciones. El análisis se estructura en función de las principales categorías de problemas tratados, como se definió previamente en la Sección 3.2, poniendo el énfasis en las metodologías empleadas y los resultados obtenidos en cada caso.

4.1. Académicos

Una de las categorías con mayor ocurrencias en los documentos observados corresponden a problemas del ámbito académico, principalmente para la administración y programación de actividades en centros educativos. Esto puede deberse a que la ITC dedicó sus primeras cuatro ediciones exclusivamente a problemas de índole académico (EURO Working Group on Automated Timetabling, 2024). Buena parte de los documentos encontrados corresponde a participaciones en estas competencias o estudios posteriores de dichas participaciones. Además, hay documentos que utilizan los datos de pruebas para evaluar el rendimiento de sus propuestas.

4.1.1. Examination Timetabling Problem

La organización de exámenes fue uno de los problemas más encontrados (ETTP, The Examination Timetabling Problem) dentro de esta categoría. Las restricciones del problema corresponden a la capacidad de salones, la disponibilidad docente y restricciones estructurales como que un estudiante solo puede dar un examen en un tiempo dado. A esto se le suman restricciones blandas, como evitar que estudiantes tengan que dar más de un examen un mismo día. La solución al problema debe no violar ninguna restricción dura y, en general,

minimizar la cantidad de restricciones blandas violadas. Este tipo de problemas suele ser muy heterogéneo unos de otros pues cada institución y periodo de exámenes presenta sus propias particularidades, lo cual hace cada problema real muy específico(Ceschia, Di Gaspero, y Schaerf, 2023a).

El documento (Larabi Marie-Sainte, 2015) es un estudio de literatura sobre el uso de la metaheurística "Particle Swarm Optimization" para resolver el problema ETP. Concluyen que el estudio de esta heurística es relativamente nuevo para resolver el este problema concreto y es escasa la literatura comparado con otras técnicas como los algoritmos genéticos.

En (Leite, Fernandes, Melício, y Rosa, 2018) se propone utilizar un algoritmo híbrido que combina dos metaheurísticas, el uso de un algoritmo evolutivo celular y el algoritmo de búsqueda Local Threshold Acceptance. Este algoritmo híbrido funciona en dos fases, una primera fase donde se buscan soluciones factibles y una segunda fase donde estas soluciones son mejoradas. Este enfoque es evaluado en los conjuntos de datos de la Universidad de Toronto y en el conjunto de datos de prueba de la International Timetabling Competition del año 2007(ITC2007) para Examination Timetabling Problem donde encuentran cuatro y tres nuevas mejores soluciones para cada conjunto respectivamente. Este enfoque híbrido es comparado además con resultados de ambas metaheurísticas de manera individual, demostrando resultados similares o mejores en comparación.

En (Leite y cols., 2019) se presenta una variante de Simulated Annealing, a la que llaman Fast Simulated Annealing(FastSA), y estudian los efectos de la temperatura de recocido en el problema de planificación de exámenes. Observaron que es esencial que los exámenes que tienen asignaciones posibles más restringidas sean asignadas de forma óptima o cercana a la óptima antes de poder asignar de forma óptima los exámenes que tienen más libertad de posibles asignaciones validas. Además observaron que los exámenes "difíciles"de asignar solo cambiaban su asignación bajo una alta temperatura.

La velocidad de esta variante de SA se basa en no volver a evaluar posibles movimientos de exámenes que, a un mayor rango de temperatura, no presentaron movimientos validos posibles, lo cual por contrapartida limita el espacio de búsqueda y podría llevar a peores resultados si algún examen "difícil"podía ser asignado a menor temperatura. Utilizaron los datos de pruebas de la ITC2007, donde en promedio FastSA utiliza 17 % menos evaluaciones que SA, aunque este valor varía entre 1 % y 57 % dependiendo de la instancia. FastSA presenta mejores tiempos promedios en 4 de 12 instancias comparado con SA y mejora 1 de 12 instancias comparado con los mejores algoritmos ejecutados en esos conjuntos de datos.

En un estudio posterior (Leite, Melício, y Rosa, 2021) de los mismos autores utilizan una estrategia similar, donde no se vuelven a revisar los exámenes con

una cantidad determinada pequeña o nula de movimientos validos en iteraciones previas, para desarrollar una variante de Threshold Acceptance presentando resultados similares sobre los mismos datos de prueba.

En (Ceschia, Di Gaspero, y Schaerf, 2023b) se realiza un survey de varios métodos de Timetabling en el ambiente académico. Además se presenta una formulación estándar y una suerte de zoológico de ejemplos de problemas, soluciones y datasets en <https://opthub.uniud.it/>. El documento se centra en comparar soluciones a varios problemas de Timetabling académicos. Se concluye que muchos de los resultados de los documentos relacionados a Timetabling no son reproducibles ya que no se encuentra de forma pública el dataset o la solución. Además se muestran problemas en la formulación de problemas que son demasiado específicos sin tomar en cuenta posibles generalizaciones. En algunos casos la formulación no es suficientemente específica para poder reproducir el problema. También se plantea que el código fuente rara vez se hace público lo que dificulta verificar los resultados de los documentos.

En (Muklason, Pratama, y Premananda, 2024) se utiliza un sistema de hiperheurísticas para intentar solucionar un problema de Timetabling de exámenes. Es importante notar que el problema cuenta con una sola restricción dura y maneja 20 slots de tiempo. Se utilizan datos reales de año académico 2021/2022 del Department of Information Systems. Aunque no se encuentran los datos utilizados de forma pública en las conclusiones se expresa que se logró solucionar el problema de forma satisfactoria con una buena velocidad y que la calidad de la solución mejora a más iteraciones se toman.

4.1.2. University Course Timetabling

La organización de horarios de los cursos en una universidad es un problema NP-hard (Babaei, Karimpour, y Hadidi, 2015) que implica la asignación óptima de recursos, tales como alumnos, profesores, y salones, a horarios específicos, cumpliendo una serie de restricciones (Imran Hossain y cols., 2019a). Este problema es complejo (Badoni, Gupta, y Mishra, 2014) debido a la gran cantidad de posibles asignaciones y la diversidad de restricciones que deben respetarse. Las restricciones pueden ser tanto duras como blandas, variando según los objetivos específicos de cada institución y la naturaleza de sus programas académicos.

En este tipo de problemas, las restricciones duras son (Tuga, Berretta, y Mendes, 2007; Brélaz, 1979):

- H1: Todo evento debe ser asignado en un espacio de tiempo determinado en un salón adecuado.
- H2: Ningún estudiante puede participar en más de un evento a la vez.
- H3: Solo un evento puede ser asignado a un salón en un tiempo determinado.

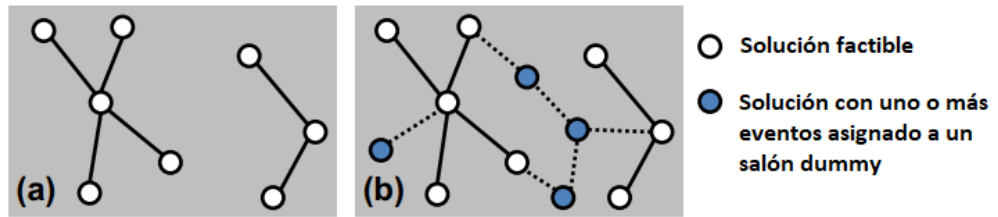


Figura 4.1: Extraído de (Lewis y Thompson, 2015)

En (Badoni y cols., 2014) proponen un enfoque híbrido combinando un algoritmo genético con un algoritmo de búsqueda local (GALS), pero la parte más significativa es la agrupación de estudiantes, pasando el problema de asignar cada estudiante individualmente a un evento a asignar una agrupación de estudiantes a un evento. Comparan este enfoque de agrupaciones con GALS sin agrupar en los datos de prueba de (Rossi-Doria y cols., 2003) e ITC2002 donde la agrupación por eventos mejora en todas las pruebas comparando GALS sin agrupar y encontraron soluciones factibles en instancias donde GALS no pudo. Comparado con otros algoritmos de referencia alcanzan resultados competitivos.

En (Lewis y Thompson, 2015) estudian como una mayor conectividad entre las soluciones del espacio de soluciones mejora la exploración de soluciones válidas. Para ello introducen de forma novedosa el uso de salones dummy, que están disponibles en cualquier momento y son aptos para cualquier evento. Utilizan un enfoque en dos pasos donde primero se busca una solución factible para las restricciones duras sin considerar las restricciones blandas y luego utilizan SA para optimizar la solución, como lo indica la Figura 4.1. Presentan resultados competitivos en las pruebas ITC2007. En general mejorar el espacio de soluciones mejora la solución obtenida, pero esto no es cierto en todas las instancias.

En (Kampke, Segatto, Boeres, Rangel, y Mauri, 2017) se utiliza Neighborhood Search combinado con otras heurísticas y neighborhoods específicos para intentar conseguir soluciones sobre el dataseta de ITC-2007. En general al disminuir el movimiento entre neighborhoods y enfocándose mas sobre Steepest Descent se consiguió los mejores resultados.

En (Song, Liu, Tang, Peng, y Chen, 2018) solo se centran en encontrar soluciones factibles que no violen restricciones duras del problema utilizando un algoritmo en 3 fases: inicialización, intensificación y diversificación. Una vez que se construye una solución parcialmente factible, se realizan de manera iterativa una búsqueda local basada en Simulated Annealing y un procedimiento de diversificación que perturba la solución, hasta que se cumple una condición de parada. Lograron encontrar soluciones factibles para 58 de 60 instancias de pruebas propuestas en (Lewis y Paechter, 2007), 3 instancias más que el algo-

ritmo que más instancias había resuelto previamente.

Posteriormente en (Chen y cols., 2020) utiliza una estrategia de obtener por construcción una solución inicial parcialmente factible para las restricciones duras H1 y H3 sin considerar H2. Luego utilizan Tabu Search para minimizar las violaciones de este ultimo tipo, obteniendo 60 soluciones factibles en las instancias de pruebas propuestas en (Lewis y Paechter, 2007).

En (Song y cols., 2021) proponen un algoritmo de búsqueda local que llaman Competition-Guided Multi-Neighborhood Local Search (CMLS). Este algoritmo selecciona solo un vecindario en cada iteración, utiliza dos reglas heurísticas para determinar las probabilidades de selección del vecindario e implementa dos procedimientos de búsqueda local multi-vecindario basados en SA durante el proceso de búsqueda. Luego, se elige el mejor de los dos resultados obtenidos como la solución inicial para la siguiente etapa de búsqueda. Utilizan un grupo reducido de pruebas de la ITC2007, donde alcanzan mínimos en 17 de 21 instancias comparado con otros 6 algoritmos conocidos.

En (Tan, Goh, Sura, Kendall, y Sabar, 2021) se utiliza PSO(Particle Swarm Optimization) con HC(Hill Climbing) para intentar obtener de múltiples valores iniciales una solución aceptable. El documento concluye que a pesar de que en la práctica HC sin PSO dio mejores resultados esto es causado solo por limitante de recursos de hardware y que con mejor hardware PSO con HC funcionaria mejor. El documento explica que la diferencia se da por que HC solo puede realizar muchas más iteraciones con menos recursos.

El documento (Siddiqui y Arshad Raza, 2021) si bien utiliza Simulated Annealing para refinar soluciones, el uso o la aplicación de la metaheurística no es el centro del estudio. En este documento proponen una ontología que sirve como una base común para el modelado de problemas y que esto permita aprovechar mejor su estructura, permitiendo encontrar mejores soluciones iniciales que luego son explotadas con una metaheurística. Esto es algo que pudimos observar de varios de los documentos analizados, donde las ganancias más grandes del valor de la función objetivo se dieron por la explotación de alguna particularidad en el modelado del problema concreto y no por la metaheurística elegida.

En (Tayarani-N., 2020) se explora el uso de Algoritmos Evolutivos con Inspiración Cuántica (QEA por sus siglas en ingles) para la resolución del problema de la programación de cursos universitarios. Estos son algoritmos basados en la población, que utilizan principios de la computación cuántica con algoritmos evolutivos, en particular el estado de los individuos se codifica a partir de estados superpuestos de forma probabilística. El documento menciona que este enfoque se ha utilizado con éxito en varios problemas combinatorios, pero muy poco en problemas de Timetabling. Los autores proponen mejoras sobre QEA anteriores y su propuesta fue puesta a prueba junto a otros algoritmos evolutivos y Simulated Annealing en varias pruebas sintéticas y dio resultados superadores

en algunas pruebas pero peores en otras.

En (Imran Hossain, Akhand, Shuvo, Siddique, y Adeli, 2019b) se busca atacar el problema de programación de cursos universitarios con mejoras al algoritmo evolutivo de Particle Swarm Optimization (PSO). PSO es un algoritmo basado en el comportamiento social de enjambres. Trabaja con una población de partículas, donde cada partícula indica una solución candidata al problema de optimización en un espacio de búsqueda multidimensional. En cada iteración la posición de las partículas se ajusta calculando una velocidad considerando su posición actual, su mejor posición personal y la mejor posición global de la población. En el documento se propone un PSO con búsqueda selectiva (PSOSS) que busca optimizar el proceso de búsqueda de soluciones para un problema con muchas restricciones como lo es el de la programación de cursos universitarios.

4.2. Deportivos

En esta categoría se incluyen los problemas de índole deportiva, sobre los cuales se encontró mucha literatura. Estos problemas corresponden a la organización de distintos tipos de torneos deportivos.

4.2.1. Round Robin

Los torneos Round Robin consisten en el enfrentamiento de cada participante contra cada participante. Se encuentran incluso generalizaciones de Round Robin donde se estudia algoritmos para planificar torneos Round Robin x donde cada participante se enfrenta x veces a cada otro participante (Cao, Wu, y Yagiura, 2022). El objetivo a optimizar en este problema puede variar entre minimizar la distancia recorrida por los participantes, minimizar la cantidad de enfrentamientos consecutivos como local o visitante, y otros criterios que ayuden a obtener un torneo equilibrado o cumplir objetivos comerciales. En los resultados obtenidos de la consulta uno de los problemas que se repite es el de scheduling de torneos Round-Robin.

En el estudio (Fonseca y Toffolo, 2022) analizan y comparan su participación en la ITC2021. Utilizan una estrategia donde ejecutan el mismo algoritmo de optimización 2 veces. En la primera ejecución, se obtiene una solución inicial utilizando el Método del Polígono propuesto por (C. C. Ribeiro y Urrutia, 2007)). En la segunda ejecución, se utiliza como entrada la solución factible obtenida en la primera ejecución. El algoritmo perturba la solución junto con una búsqueda local para mejorar el resultado.

En (Jaradat, 2022) Se utilizan dos algoritmos basados en Ant Colony Optimization aplicado al caso de torneos Round Robin. Buscando satisfacer las condiciones "hardz minimizar las violaciones a las condiciones "soft". En el documento no se trabaja sobre el tiempo que toma llegar a soluciones si no que

se centra en la calidad de soluciones. De acuerdo a lo presentado se logran mejoras. En sus pruebas encontraron soluciones factibles para 37 de las 45 instancias. Además encontraron la mejor solución general para 4 instancias, 2 de ellas óptimas.

En (Rosati, Petris, Di Gaspero, y Schaerf, 2022) se utiliza Simulated Annealing con CPLEX sobre el dataset del ITC2021, específicamente sobre torneos Double Round-Robin. Utilizando un límite de tiempo de una hora, solo se consiguió solución para un caso de 45. Aumentando el límite a 24 horas se consiguió solo para 6. Reduciendo la dificultad del problema se logran mejores resultados.

En (Dong y cols., 2023) se utiliza Dinamic Scheduling para resolver el problema de scheduling asociado a torneos. Es interesante marcar que es uno de los pocos documentos analizados en el que se intenta optimizar los espectadores de los eventos. Se toma como mayor indicador de interés que tan difíciles de predecir son los resultados de una partida dada. Se utiliza CPLEX 12.9 para intentar solucionar los casos de Round-Robin, el sistema suizo y eliminatorias simple.

4.2.2. Carry over effect

En muchos torneos de deportes competitivos en los que se participa en múltiples encuentros, enfrentarse a un oponente fuerte o débil afecta el rendimiento general posterior de los equipos. Dado que las habilidades de los equipos no son iguales, es posible que algunos equipos se beneficien de la forma en que se programa un torneo. Por ejemplo, si un equipo juega contra un oponente difícil, es probable que esté cansado para el próximo encuentro debido al gran esfuerzo realizado. Supongamos que el equipo A juega contra el equipo B, y el equipo B es un equipo fuerte. Luego, cuando el equipo A juegue contra el equipo C, el equipo C tendrá ventaja sobre el equipo A porque es probable que el equipo A esté agotado del encuentro anterior. Esto se conoce como Carry Over Effect, donde el equipo C se ve beneficiado por el cansancio causado por el equipo B, mientras que el equipo A sale perjudicado. Este es uno de los posibles parámetros a querer minimizar al planificar un torneo, para que este sea más justo. (Günneç y Demir, 2019)

En (Guedes y Ribeiro, 2011) estudian el efecto Carry Over y como minimizar este puede no ser suficiente para garantizar un torneo justo. Proponen un utilizar una ponderación del efecto en base a la diferencia de “fuerza” de un encuentro particular. Utilizan un enfoque híbrido en el cual primero construyen 100 soluciones factible eligiendo con misma probabilidad agregar como siguiente enfrentamiento el que minimiza el efecto Weighted-Carry-Over o un enfrentamiento al azar. Luego utilizan ITL para cada uno de los 100 soluciones candidatos refinando la solución minimizando Weighted-Carry-Over. Posteriormente seleccionan la mejor de las 100 soluciones y vuelven a repetirlo utilizando ITL pero perturbando la solución hasta llegar al criterio de parada.

Este algoritmo es probado utilizando datos de pruebas creado por los mismos autores con cuatro características diferentes:

- Instancias aleatorias: Se asigna una fuerza aleatoria $[1, 2n]$ a cada equipo
 - Instancias lineales: Se asigna una fuerza en el intervalo $[1, n]$ a cada equipo. Por simplicidad, la fuerza del equipo i se asigna igual a i
 - Instancias lineales perturbadas: Se perturba cada entrada de la instancia lineal con un valor entre $[-n/2, n/2]$ y se toma valor absoluto
- Instancias inspiradas en la vida real: Inspiradas en seis ediciones del campeonato de fútbol brasileño. La fuerza de cada equipo se da por el número de puntos que obtuvo en el año anterior

En (Cao y cols., 2022) formulan el problema como uno de grafos donde una programación de un torneo Round Robin puede ser considerada como una descomposición en emparejamientos perfectos de un grafo completo K_n como se ve en el ejemplo de la Figura 4.2 para K_8 y cada ronda del torneo es un emparejamiento perfecto en K_n como se muestra en la Figura 4.3.

En general, calcular el efecto Carry Over de una programación particular es $O(n^2)$, pero calcularlo de una programación generada a partir de estos emparejamientos perfectos en $O(n)$ utilizando un algoritmo propuesto en (Anderson, 1997). Utilizan un algoritmo de búsqueda local en el que una solución se representa mediante un emparejamiento perfecto base como se muestra en la Figura 4.3. El objetivo es encontrar un emparejamiento perfecto base que genere un calendario con un valor del efecto Carry Over lo más pequeño posible. Además utilizan Tabu Search para evitar volver a revisar soluciones previas. Logran resultados competitivos en 15 de 17 instancias, mejorando 2 de los mínimos valores conocidos.

4.2.3. Traveling tournament

El problema del Traveling Tournament consiste en minimizar la distancia de viaje para un torneo donde cada participante debe enfrentarse a cada oponente dos veces, una vez como local y otra como visitante, además dos oponentes no se enfrentan en dos encuentros consecutivos y existe un número máximo de encuentros consecutivos que un equipo puede jugar como visitante y como local. Cada equipo comienza el torneo en su ubicación de local y se asume que siempre viajan desde la ubicación de su último encuentro al siguiente (Frohner, Neumann, y Raidl, 2020).

En (F.-C. Yang, 2017) se trabaja sobre una versión más compleja del Traveling Tournament con los requerimientos específicos de la NBA y un algoritmo genético. Se consiguió una mejora notable sobre el calendario oficial usado, tanto en la distancia de viaje como en las restricciones suaves. Las pruebas realizadas en ese documento fueron hechas contra los datos reales de la NBA de 2014-2015.

En (Frohner y cols., 2020) modelan el problema como un grafo de estados donde cada nodo representa una solución parcial al problema y cada transición una programación de un partido y utilizan Beam Search, que es una heurística “greedy” de búsqueda local en la cual solo una parte de las mejores soluciones parciales son conservadas como soluciones candidatas. Además perturban la solución para introducir diversidad al proceso de búsqueda y paralelizan la búsqueda en estas soluciones perturbadas. Utilizan los datos de pruebas “National League Circular Benchmark” propuestos en (Easton, Nemhauser, y Trick, 2001) donde su enfoque en general se compara o supera las soluciones obtenidas mediante Ant Colony Optimization y Tabu Search. Además, para uno de los conjuntos de prueba encontraron 2 nuevas mejores soluciones que no habían sido mejoradas en 12 años.

4.3. Otros

En esta categoría se incluyen los problemas sobre los que se encontró escasa literatura, debido a la especificidad sobre el problema que resuelven.

4.3.1. Médicos

En la literatura se presentan mas de una vez problemas relacionados al ámbito de la salud. Es difícil describir los problemas que se presentan de forma genérica ya que no caen todos dentro del mismo tipo de problemas.

En (Kling, Kraul, y Brunner, 2024) se utiliza IP(Integer programming) y GRASP(Greedy Randomized Adaptive Search Procedure) para solucionar el problema de agendar pacientes en un hospital con varios terapeutas, tomando en cuenta varios problemas del ambiente. GRASP es la diferencia en llegar a soluciones en el tiempo esperado y no llegar. Para este problema se consiguieron soluciones en menos de dos horas.

En (Niessner, Rauner, y Gutjahr, 2018) estudian el problema de manejar incidentes con múltiples víctimas (MCIs por sus siglas en ingles) en Austria, donde los servicios médicos de emergencia en países de Europa Central suelen establecer pequeños hospitales de campaña llamados puestos médicos avanzados (AMPs por sus siglas en ingles). Estos centros estabilizan y tratan a los pacientes de emergencia antes de su traslado a los hospitales. Investigan este problema utilizando tres métodos de simulación-optimización, uno de ellos es el uso de el motor de optimización OptQuest que utiliza metaheurísticas y que, por razones comerciales, funciona como una caja negra, donde solo se observa la entrada y salida. El uso de este motor dio resultados comparables a otro de los métodos estudiados.

4.3.2. Cross Subsidization

Un problema encontrado, particularmente complejo, fue el de el subsidio cruzada, donde servicios con alto margen de ganancia mantienen servicios con poca o ninguna ganancia, como por ejemplo una cadena hotelera utilizando las ganancias de las habitaciones normales para subsidiar la oferta de habitaciones de lujo que promocionan la marca. En (Lu, Nakao, Shen, y Zhao, 2021) se consideran modelos genéricos para la planificación y administración de recursos de actividades sin fines de lucro con subsidio cruzado.

4.3.3. Workforce scheduling

El problema de la programación de horarios laborales (Workforce Scheduling) se presenta en compañías o instituciones con empleados que trabajan en turnos o con jornadas laborales irregulares. Esto puede incluir hospitales, call centers, atención domiciliaria, limpieza, entre otros. Normalmente el principal objetivo de la optimización de la programación horarios laborales es la eficiencia financiera, pero otros objetivos pueden incluir cargas de trabajo más justas y la satisfacción de los empleados. En (Nurmi, Kyngäs, y Kyngäs, 2023) se explora el uso de la metaheurística PEAST para resolver el problema de la programación de horarios laborales. Esta metaheurística combina características de otras metaheurísticas bien conocidas: Genetic Algorithms, Tabu Search, Simulated Annealing y otras.

4.3.4. Sin aplicación

Algunos documentos encontrados no trabajaban en problemas de Timetabling particulares, sino sobre otros problemas, pero aun así trabajan con metaheurísticas que resultan de interés para el estudio llevado a cabo.

En (Husseinzadeh Kashan, 2014) se trabaja sobre LCA(The League Championship Algorithm), no se aplica específicamente sobre problemas de Timetabling si no que se trabaja de forma genérica como metaheurística. Se investiga sobre cambios a LCA y como lo afectan, además de realizar comparaciones con PSO. Contra la mayoría de los datasets el documento concluye que LCA tiene un buen rendimiento comparándolo con PSO, y que dependiendo del problema en particular se le puede aplicar diferentes optimizaciones que mejoran el rendimiento.

En (Kashan, 2009) se compara LCA contra PSO. Se comparan los dos algoritmos sobre funciones tradicionales de optimización, siendo la función de Rosenbrock la más notable. Se concluye que el algoritmo de LCA es suficientemente útil para poder utilizarse y converge mas rápido que PSO en los casos evaluados.

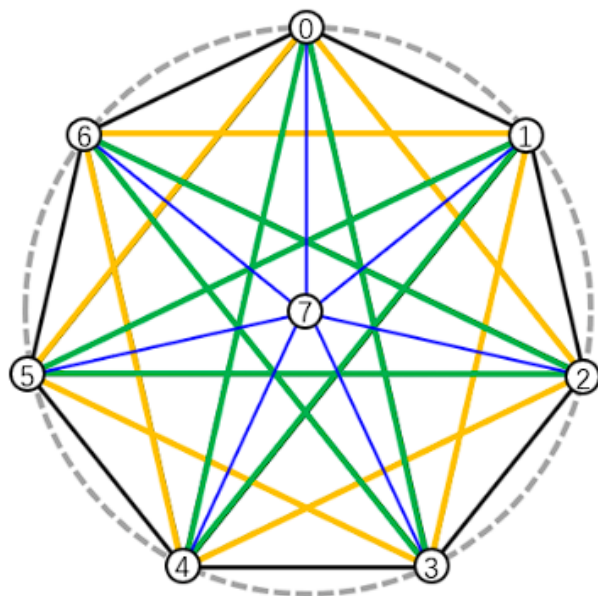


Figura 4.2: Grafo completo K_8 .
Fuente: Extraído de (Cao y cols., 2022)

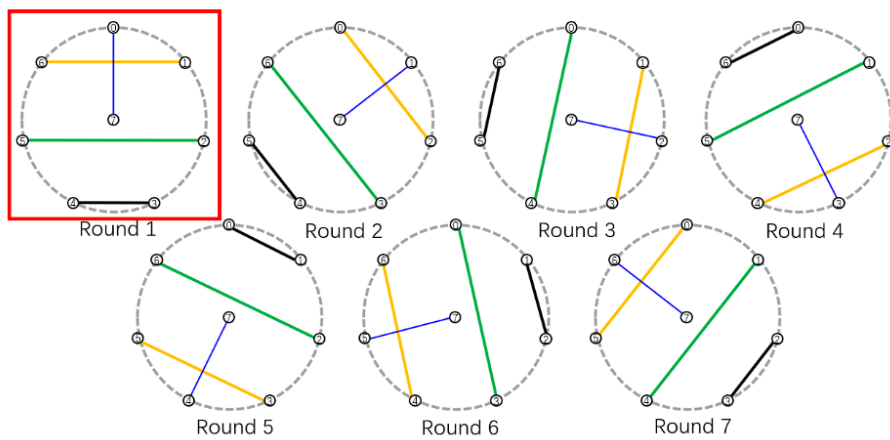


Figura 4.3: Programación de un torneo generada a partir de un emparejamiento perfecto de un grafo completo.
Fuente: Extraído de (Cao y cols., 2022)

Capítulo 5

Conclusiones

En este documento se presenta la revisión de la literatura sobre metaheurísticas aplicadas a la programación de eventos. En esta sección se presentan y comentan las conclusiones más relevantes.

A pesar de incluirlo explícitamente en el término de búsqueda, no se encontró ningún documento que trabaje sobre eventos de carácter artístico. La mayor parte de los documentos analizados se centran en eventos deportivos o académicos.

Los casos de prueba utilizados en la mayoría de los documentos son casos sintéticos, en general los utilizados en las ITC, siendo muy pocos los documentos que utilizaron casos de pruebas basados en datos reales.

Uno de los problemas más notorios que se encontró al momento de intentar comparar resultados es la disparidad de criterios de evaluación y medida dentro de los diferentes documentos. Muchos no tienen registros de cuanto tiempo tomó llegar a las soluciones y solo se centran la calidad del resultado. Dentro de los que si registran el tiempo hay diferencias entre registrar el tiempo en cantidad de iteración o en tiempo real, y a su vez usualmente no hay información completa sobre que hardware se utilizó para las pruebas. Esto se ve agravado por la falta de acceso a código fuente y los conjuntos de datos utilizados que dificulta reproducir los resultados para poder compararlos.

Dentro de los documentos analizados se pudo apreciar que la tendencia general indica que el uso de metaheurísticas ofrece una mejora o bien en el tiempo necesario para llegar a un resultado, o en la calidad del resultado, en varios casos en ambas dimensiones.

Uno de los aspectos más importantes que se observó a lo largo del análisis es el papel que juega el modelado del problema en la efectividad de las metaheurísticas aplicadas. En particular, un diseño de la representación de los datos utilizados que facilite operar sobre la misma a la vez que restrinja opciones no deseadas y que faciliten buenas soluciones iniciales. Buenas soluciones iniciales suelen mejorar la velocidad de convergencia de las metaheurísticas y la calidad de las soluciones finales encontradas por las metaheurísticas.

Referencias

- Al-Ashhab, y Alghamdi, A. (2018). Two-stage multi-objective university courses timetabling using genetic algorithms. *International Journal of Engineering and Technology*, 10(4), 1102–1111. doi: <http://dx.doi.org/10.21817/ijet/2018/v10i4/181004030>
- Aminzadegan, S., Tamannaie, M., y Fazeli, M. (2021). An integrated production and transportation scheduling problem with order acceptance and resource allocation decisions. *Applied Soft Computing*, 112(107770), 107770. doi: <https://doi.org/10.1016/j.asoc.2021.107770>
- Anderson, I. (1997). *Combinatorial designs and tournaments*. Oxford, England: Clarendon Press.
- Babaei, H., Karimpour, J., y Hadidi, A. (2015). A survey of approaches for university course timetabling problem. *Computers & Industrial Engineering*, 86, 43–59. doi: <http://10.1016/j.cie.2014.11.010>
- Badoni, R. P., Gupta, D. K., y Mishra, P. (2014). A new hybrid algorithm for university course timetabling problem using events based on groupings of students. *Computers & Industrial Engineering*, 78, 12–25. doi: <https://doi.org/10.1016/j.cie.2014.09.020>
- Brélaz, D. (1979). New methods to color the vertices of a graph. *Communications of the ACM*, 22(4), 251–256. doi: <https://doi.org/10.1145/359094.359101>
- Cao, Y., Wu, W., y Yagiura, M. (2022). A metaheuristic algorithm for the carry-over effect value minimization problem in round-robin tournaments. *Journal of Advanced Mechanical Design, Systems, and Manufacturing*, 16(4). doi: <http://dx.doi.org/10.1299/jamdsm.2022jamdsm0042>
- Ceschia, S., Di Gaspero, L., y Schaerf, A. (2023a). Educational timetabling: Problems, benchmarks, and state-of-the-art results. *European Journal of Operational Research*, 308(1), 1–18. doi: <https://doi.org/10.1016/j.ejor.2022.07.011>
- Ceschia, S., Di Gaspero, L., y Schaerf, A. (2023b). Educational timetabling: Problems, benchmarks, and state-of-the-art results. *European Journal of Operational Research*, 308(1), 1–18. doi: <https://doi.org/10.1016/j.ejor.2022.07.011>
- Chen, M., Tang, X., Song, T., Wu, C., Liu, S., y Peng, X. (2020). A tabu search algorithm with controlled randomization for constructing feasible univer-

- sity course timetables. *Computers & Operations Research*, 123(105007), 105007. doi: <https://doi.org/10.1016/j.cor.2020.105007>
- de Werra, D. (1985). An introduction to timetabling. *European Journal of Operational Research*, 19(2), 151–162. doi: [https://doi.org/10.1016/0377-2217\(85\)90167-5](https://doi.org/10.1016/0377-2217(85)90167-5)
- de Werra, D. (1997). The combinatorics of timetabling. *European Journal of Operational Research*, 96(3), 504–513. doi: [https://doi.org/10.1016/S0377-2217\(96\)00111-7](https://doi.org/10.1016/S0377-2217(96)00111-7)
- Diccionario de la lengua española*, 23.^a ed. (2014). REAL ACADEMIA ESPAÑOLA.
- Dong, Z.-L., Ribeiro, C. C., Xu, F., Zamora, A., Ma, Y., y Jing, K. (2023). Dynamic scheduling of e-sports tournaments. *Transportation Research Part E: Logistics and Transportation Review*, 169(102988), 102988. doi: <https://doi.org/10.1016/j.tre.2022.102988>
- Dueck, G., y Scheuer, T. (1990). Threshold accepting: A general purpose optimization algorithm appearing superior to simulated annealing. *Journal of Computational Physics*, 90(1), 161–175. doi: [http://dx.doi.org/10.1016/0021-9991\(90\)90201-B](http://dx.doi.org/10.1016/0021-9991(90)90201-B)
- Easton, K., Nemhauser, G., y Trick, M. (2001). The traveling tournament problem description and benchmarks. En *Principles and practice of constraint programming — CP 2001* (p. 580–584). Springer Berlin Heidelberg. doi: http://dx.doi.org/10.1007/3-540-45578-7_43
- EURO Working Group on Automated Timetabling. (2024). *PATAT Community Support*. Descargado de <https://patatconference.org/communityService.html> (ultimo acceso: 14.08.2024)
- Fink, A., y Voss, S. (1999). Generic metaheuristics application to industrial engineering problems. *Computers & Industrial Engineering*, 37(1–2), 281–284. doi: [10.1016/S0360-8352\(99\)00074-1](https://doi.org/10.1016/S0360-8352(99)00074-1)
- Fonseca, G. H. G., y Toffolo, T. A. M. (2022). A fix-and-optimize heuristic for the ITC2021 sports timetabling problem. *Journal of Scheduling*, 25(3), 273–286. doi: <https://doi.org/10.1007/s10951-022-00738-6>
- Frohner, N., Neumann, B., y Raidl, G. R. (2020). A beam search approach to the traveling tournament problem. En *Evolutionary computation in combinatorial optimization* (pp. 67–82). Springer International Publishing. doi: https://doi.org/10.1007/978-3-030-43680-3_5
- Glover, F. (1989). Tabu search—Part I. *ORSA Journal on Computing*, 1(3), 190–206. doi: <http://dx.doi.org/10.1287/ijoc.1.3.190>
- Guedes, A. C. B., y Ribeiro, C. C. (2011). A heuristic for minimizing weighted carry-over effects in round robin tournaments. *Journal of Scheduling*, 14(6), 655–667. doi: <https://doi.org/10.1007/s10951-011-0244-y>
- Günneç, D., y Demir, E. (2019). Fair-fixtue: Minimizing carry-over effects in football leagues. *Journal of Industrial & Management Optimization*, 15(4), 1565–1577. doi: <http://dx.doi.org/10.3934/jimo.2018110>
- Husseinzadeh Kashan, A. (2014). League championship algorithm (LCA): An algorithm for global optimization inspired by sport championships.

- Applied Soft Computing*, 16, 171–200. doi: <https://doi.org/10.1016/j.asoc.2013.12.005>
- Imran Hossain, S., Akhand, M., Shuvo, M., Siddique, N., y Adeli, H. (2019b). Optimization of university course scheduling problem using particle swarm optimization with selective search. *Expert Systems with Applications*, 127, 9–24. doi: <http://dx.doi.org/10.1016/j.eswa.2019.02.026>
- Imran Hossain, S., Akhand, M. A. H., Shuvo, M. I. R., Siddique, N., y Adeli, H. (2019a). Optimization of university course scheduling problem using particle swarm optimization with selective search. *Expert Systems with Applications*, 127, 9–24. doi: <https://doi.org/10.1016/j.eswa.2019.02.026>
- Jaradat, G. M. (2022). Elitist-ant system metaheuristic for ITC 2021—sports timetabling. En *Studies in computational intelligence* (pp. 51–83). Springer International Publishing. doi: https://doi.org/10.1007/978-3-030-99079-4_3
- Kampke, E. H., Segatto, E. A., Boeres, M. C. S., Rangel, M. C., y Mauri, G. R. (2017). Neighborhood analysis on the university timetabling problem. En *Computational science and its applications – ICCSA 2017* (pp. 148–164). Springer International Publishing. doi: https://doi.org/10.1007/978-3-319-62398-6_11
- Kashan, A. H. (2009). League championship algorithm: A new algorithm for numerical function optimization. En *2009 International Conference of Soft Computing and Pattern Recognition*. IEEE. doi: <https://doi.org/10.1109/SoCPaR.2009.21>
- Kennedy, J., y Eberhart, R. (1995). Particle swarm optimization. En *Proceedings of ICNN'95 - International Conference on Neural Networks* (Vol. 4, p. 1942–1948). IEEE. doi: <http://dx.doi.org/10.1109/ICNN.1995.488968>
- Kirkpatrick, S., Gelatt, C. D., y Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, 220(4598), 671–680. doi: <http://dx.doi.org/10.1126/science.220.4598.671>
- Kling, S., Kraul, S., y Brunner, J. O. (2024). Customized GRASP for rehabilitation therapy scheduling with appointment priorities and accounting for therapist satisfaction. *OR Spectrum*. doi: <https://doi.org/10.1007/s00291-023-00742-y>
- Kostuch, P. (2005). The university course timetabling problem with a three-phase approach. En *Practice and theory of automated timetabling V* (pp. 109–125). Springer Berlin Heidelberg. doi: https://doi.org/10.1007/11593577_7
- Larabi Marie-Sainte, S. (2015). A survey of Particle Swarm Optimization techniques for solving university Examination Timetabling Problem. *Artificial Intelligence Review*, 44(4), 537–546. doi: <https://doi.org/10.1007/s10462-015-9437-7>
- Leite, N., Fernandes, C. M., Melício, F., y Rosa, A. C. (2018). A cellular memetic algorithm for the examination timetabling problem. *Computers & Operations Research*, 94, 118–138. doi: <https://doi.org/10.1016/j.cor.2018.02.009>
- Leite, N., Melício, F., y Rosa, A. C. (2019). A fast simulated annealing algorithm

- for the examination timetabling problem. *Expert Systems with Applications*, 122, 137–151. doi: <https://doi.org/10.1016/j.eswa.2018.12.048>
- Leite, N., Melício, F., y Rosa, A. C. (2021). A fast threshold acceptance algorithm for the examination timetabling problem. En *International series in operations research & management science* (pp. 323–363). Springer International Publishing. doi: https://doi.org/10.1007/978-3-030-74051-1_11
- Lewis, R., y Paechter, B. (2007). Finding feasible timetables using group-based operators. *IEEE Transactions on Evolutionary Computation*, 11(3), 397–413. doi: <http://dx.doi.org/10.1109/TEVC.2006.885162>
- Lewis, R., y Thompson, J. (2015). Analysing the effects of solution space connectivity with an effective metaheuristic for the course timetabling problem. *European Journal of Operational Research*, 240(3), 637–648. doi: <https://doi.org/10.1016/j.ejor.2014.07.041>
- Lu, M., Nakao, H., Shen, S., y Zhao, L. (2021). Non-profit resource allocation and service scheduling with cross-subsidization and uncertain resource consumptions. *Omega*, 99(102191), 102191. doi: <https://doi.org/10.1016/j.omega.2019.102191>
- Mccollum, B., McMullan, P., Burke, E., Parkes, A., y Qu, R. (2007). The second international timetabling competition: Examination timetabling track.. Descargado de <https://www.researchgate.net/publication/215777351-The-Second-International-Timetabling-Competition-ITC-2007-Curriculum-based-Course-Timetabling-Track-3> (accedido: 14.08.2024)
- Muklason, A., Pratama, E. J., y Premananda, I. G. A. (2024). Generic university examination timetabling system with steepest-ascent hill climbing hyper-heuristic algorithm. *Procedia Computer Science*, 234, 584–591. doi: <https://doi.org/10.1016/j.procs.2024.03.043>
- Müller, T., Rudová, H., y Müllerová, Z. (2024). Real-world university course timetabling at the International Timetabling Competition 2019. *Journal of Scheduling*. doi: <https://doi.org/10.1007/s10951-023-00801-w>
- Niessner, H., Rauner, M. S., y Gutjahr, W. J. (2018). A dynamic simulation–optimization approach for managing mass casualty incidents. *Operations Research for Health Care*, 17, 82–100. doi: <https://doi.org/10.1016/j.orhc.2017.07.001>
- Nurmi, K., Kyngäs, J., y Kyngäs, N. (2023). Practical recommendations for staff rostering justified by real-world optimization. *Procedia Computer Science*, 219, 1518–1526. doi: <http://dx.doi.org/10.1016/j.procs.2023.01.443>
- Osman, I. H., y Laporte, G. (1996). Metaheuristics: A bibliography. *Annals of Operations Research*, 63(5), 511–623. doi: [10.1007/bf02125421](https://doi.org/10.1007/bf02125421)
- Perzina, R. (2007). Solving the university timetabling problem with optimized enrollment of students by a self-adaptive genetic algorithm. En *Practice and theory of automated timetabling vi* (p. 248–263). Springer Berlin Heidelberg. doi: http://dx.doi.org/10.1007/978-3-540-77345-0_16
- Poli, R. (2008). Analysis of the publications on the applications of particle swarm optimisation. *Journal of Artificial Evolution and Applications*, 2008(1), 1–10. doi: <http://dx.doi.org/10.1155/2008/685175>

- Post, G., Kingston, J., Ahmadi, S., Daskalaki, S., Gogos, C., Kyngas, J., ... Santos, H. (2011). Xhstt: An xml archive for high school timetabling problems in different countries. *Annals of Operations Research*, 218, 295–301. doi: <https://doi.org/10.1007/s10479-011-1012-2>
- Rajwar, K., Deep, K., y Das, S. (2023). An exhaustive review of the metaheuristic algorithms for search and optimization: taxonomy, applications, and open challenges. *Artificial Intelligence Review*, 1–71. doi: <https://doi.org/10.1007/s10462-023-10470-y>
- Reeves, C. R. (2010). Genetic algorithms. En *Handbook of metaheuristics* (p. 109–139). Springer US. doi: http://dx.doi.org/10.1007/978-1-4419-1665-5_5
- Ribeiro, C., Urrutia, S., y Werra, D. (2023). *Combinatorial models for scheduling sports tournaments*. doi: <http://dx.doi.org/10.1007/978-3-031-37283-4>
- Ribeiro, C. C., y Urrutia, S. (2007). Heuristics for the mirrored traveling tournament problem. *European Journal of Operational Research*, 179(3), 775–787. doi: <http://dx.doi.org/10.1016/j.ejor.2005.03.061>
- Rosati, R. M., Petris, M., Di Gaspero, L., y Schaerf, A. (2022). Multi-neighborhood simulated annealing for the sports timetabling competition ITC2021. *Journal of Scheduling*, 25(3), 301–319. doi: <https://doi.org/10.1007/s10951-022-00740-y>
- Rossi-Doria, O., Sampels, M., Birattari, M., Chiarandini, M., Dorigo, M., Gambardella, L. M., ... Stützle, T. (2003). A comparison of the performance of different metaheuristics on the timetabling problem. En *Practice and theory of automated timetabling IV* (pp. 329–351). Springer Berlin Heidelberg. doi: https://doi.org/10.1007/978-3-540-45157-0_22
- Sadeeq, H. T., y Abdulazeez, A. M. (2023). Metaheuristics: A review of algorithms. *International Journal of Online and Biomedical Engineering (iJOE)*, 19(09), 142–164. doi: 10.3991/ijoe.v19i09.39683
- Siddiqui, A. W., y Arshad Raza, S. (2021). A general ontological timetabling-model driven metaheuristics approach based on elite solutions. *Expert Systems with Applications*, 170(114268), 114268. doi: <https://doi.org/10.1016/j.eswa.2020.114268>
- Song, T., Chen, M., Xu, Y., Wang, D., Song, X., y Tang, X. (2021). Competition-guided multi-neighborhood local search algorithm for the university course timetabling problem. *Applied Soft Computing*, 110(107624), 107624. doi: <https://doi.org/10.1016/j.asoc.2021.107624>
- Song, T., Liu, S., Tang, X., Peng, X., y Chen, M. (2018). An iterated local search algorithm for the University Course Timetabling Problem. *Applied Soft Computing*, 68, 597–608. doi: <https://doi.org/10.1016/j.asoc.2018.04.034>
- Tan, J. S., Goh, S. L., Sura, S., Kendall, G., y Sabar, N. R. (2021). Hybrid particle swarm optimization with particle elimination for the high school timetabling problem. *Evolutionary Intelligence*, 14(4), 1915–1930. doi: <https://doi.org/10.1007/s12065-020-00473-x>
- Tayarani-N., M.-H. (2020). Novel operators for quantum evolutionary algorithm in solving timetabling problem. *Evolutionary Intelligence*, 14(4), 1869–1893. doi: <https://doi.org/10.1007/s12065-020-00438-0>

- Tuga, M., Berretta, R., y Mendes, A. (2007). A hybrid simulated annealing with Kempe chain neighborhood for the university timetabling problem. En *6th IEEE/ACIS international conference on computer and information science (ICIS 2007)*. IEEE. doi: <https://doi.org/10.1109/ICIS.2007.25>
- Van Bulck, D., y Goossens, D. (2023). The international timetabling competition on sports timetabling (ITC2021). *European Journal of Operational Research*, 308(3), 1249–1267. doi: <https://doi.org/10.1016/j.ejor.2022.11.046>
- Yang, F.-C. (2017). NBA sports game scheduling problem and GA-based solver. En *2017 international conference on industrial engineering, management science and application (ICIMSA)*. IEEE. doi: <https://doi.org/10.1109/ICIMSA.2017.7985598>
- Yang, X.-S. (2014). Introduction to algorithms. En *Nature-Inspired Optimization Algorithms* (p. 1–21). Elsevier. doi: <https://doi.org/10.1016/B978-0-12-416743-8.00001-4>

Resultados de pruebas

Para comparar el método AG con CPLEX, se crearon varias instancias de datos, cada cual correspondiente a una configuración con diferentes particularidades, y se corrieron con ambos métodos para tener información sobre cómo se comportan. Cada instancia de prueba consiste en correr el método con un tope de tiempo de cinco minutos y registrar algunos datos relevantes para poder comparar los métodos.

Para las pruebas en el caso del método AG, por ser un algoritmo no determinista, se realizaron para cada instancia de cada configuración 10 corridas del algoritmo con diferentes semillas. Esto es para asegurar que los resultados sean significativos. Luego se toma el promedio de los resultados obtenidos. En los casos en los que en alguna semilla no se llegó a una solución factible, no se toma en cuenta para el promedio.

Para CPLEX se registró en cada corrida el tiempo que le tomó llegar a la primera solución factible en segundos y el valor de la función objetivo en esa solución. Además se registró el tiempo que le tomó alcanzar la solución final en segundos y el valor de la función objetivo en esa solución.

Para el método AG se registran los mismos datos, y además de los tiempos, se registra en qué generación se alcanzó la primera solución factible y la solución final.

B.1. Configuración 1

B.1.1. Resultados CPLEX

Instancia	Tiempo a primera solución (s)	Función objetivo de primera solución	Tiempo a solución final (s)	Función objetivo de solución final
1	6.5	16.6	232.0	2.7
2	8.3	21.8	265.6	3.7
3	13.4	17.2	245.3	4.4
4	7.0	21.4	293.8	3.8
5	10.2	13.5	271.6	2.1
Promedio	9.1	18.1	261.7	3.3

Tabla B.1: Resultados de la configuración uno de pruebas para CPLEX.

B.1.2. Resultado Método AG

Instancia 1

Corrida	Tiempo a primera solución (s)	Generaciones a primera solución	Fitness de primera solución	Tiempo a solución final (s)	Generaciones a solución final	Fitness de solución final
0	91.4	2094	27.9	198.2	4334	4.9
1	99.2	2293	18.9	158.8	3555	3.7
2	15.9	358	18.6	87.5	1846	5.6
3	23.5	530	26.4	56.5	1198	5.6
4	254.2	5791	20.9	264.8	6034	4.9
5	17.4	393	22.8	278.6	5410	5.7
6	26.6	600	22.6	234.9	4898	3.5
7	20.0	452	24.9	78.9	1717	4.7
8	-	-	-	-	-	-
9	68.4	1519	12.2	284.8	5719	3.6
Promedios	68.5	1558.9	21.7	182.6	3856.8	4.7

Tabla B.2: Resultados de las diez corridas del método AG con diferentes semillas para la instancia uno de la configuración uno.

Instancia 2

Corrida	Tiempo a primera solución (s)	Generaciones a primera solución	Fitness de primera solución	Tiempo a solución final (s)	Generaciones a solución final	Fitness de solución final
0	11.5	265	19.5	120.3	2494	2.0
1	42.7	1030	15.8	223.1	4894	4.5
2	24.9	594	16.6	188.3	4013	2.4
3	5.6	138	15.5	168.2	3554	1.9
4	22.2	531	15.5	81.6	1806	4.6
5	56.7	1347	26.6	289.7	6542	5.1
6	22.6	538	19.7	88.7	1940	3.0
7	85.9	2028	18.1	121.5	2811	5.1
8	76.9	1736	18.1	90.6	2035	6.7
9	26.1	593	25.8	60.7	1329	6.0
Promedios	37.51	880	19.12	143.27	3141.8	4.13

Tabla B.3: Resultados de las diez corridas del método AG con diferentes semillas para la instancia dos de la configuración uno.

Instancia 3

Corrida	Tiempo a primera solución (s)	Generaciones a primera solución	Fitness de primera solución	Tiempo a solución final (s)	Generaciones a solución final	Fitness de solución final
0	10.7	263	19.7	293.8	5884	8.2
1	4.6	118	16.9	22.3	541	9.8
2	201.2	4885	22.7	242.8	5816	5.5
3	65.2	1596	25.3	198.1	4373	2.1
4	257.7	6228	17.2	291.5	7009	6.8
5	27.0	657	21.8	271.3	5718	3.1
6	6.2	157	18.9	293.4	6358	3.1
7	16.3	401	19.4	29.1	705	8.6
8	69.0	1712	28	127.8	2978	4.8
9	6.9	174	21.7	150.7	3260	5.9
Promedios	66.48	1619.1	21.16	192.08	4264.2	5.79

Tabla B.4: Resultados de las diez corridas del método AG con diferentes semillas para la instancia tres de la configuración uno.

Instancia 4

Corrida	Tiempo a primera solución (s)	Generaciones a primera solución	Fitness de primera solución (s)	Tiempo a solución final (s)	Generaciones a solución final	Fitness de solución final
0	32.6	771	18.2	59.2	1358	4.8
1	111.3	2630	23.7	159.1	3655	5.9
2	8.0	196	20.0	240.6	5104	3.9
3	22.7	552	21.2	89.7	2076	6.1
4	19.0	466	13.3	108.9	2348	4.3
5	8.4	206	21.5	42.0	960	3.4
6	9.0	215	15.8	164.1	3534	4.9
7	38.3	916	16.2	82.2	1915	3.8
8	57.5	1408	21.3	239.3	5292	3.3
9	9.9	239	18.9	183.1	3923	6.3
Promedios	31.7	759.9	19.0	136.8	3016.5	4.7

Tabla B.5: Resultados de las diez corridas del método AG con diferentes semillas para la instancia cuatro de la configuración uno.

Instancia 5

Corrida	Tiempo a primera solución (s)	Generaciones a primera solución	Fitness de primera solución	Tiempo a solución final (s)	Generaciones a solución final	Fitness de solución final
0	4.0	98	16.3	95.8	2085	3.4
1	38.0	921	22.6	69.9	1642	4.5
2	20.5	495	28.7	171.4	3523	2.3
3	28.9	694	20.6	132.8	2913	5.6
4	60.7	1461	16.5	191.4	4100	2.3
5	74.0	1762	16.1	219.6	4951	4.5
6	25.9	629	19.2	86.9	2047	4.4
7	40.9	986	19.5	109.0	2449	2.6
8	45.3	1084	20.1	177.1	3940	2.1
9	11.4	275	18.3	195.9	4160	2.7
Promedios	34.9	840.5	19.8	145.0	3181.0	3.4

Tabla B.6: Resultados de las diez corridas del método AG con diferentes semillas para la instancia cinco de la configuración uno.

B.2. Configuración 2

B.2.1. Resultados CPLEX

Instancia	Tiempo a primera solución (s)	Función objetivo de primera solución	Tiempo a solución final (s)	Función objetivo de solución final
1	14.2	21.7	145.4	3.9
2	1.8	12.2	244.8	7.0
3	9.2	30.0	243.3	1.8
4	5.8	44.4	240.2	2.2
5	15.3	44.4	195.5	2.2
Promedio	9.2	30.5	213.8	3.4

Tabla B.7: Resultados de la configuración dos de pruebas para CPLEX.

B.2.2. Resultados metodo AG

Instancia 1

Corrida	Tiempo a primera solución (s)	Generaciones a primera solución	Fitness de primera solución	Tiempo a solución final (s)	Generaciones a solución final	Fitness de solución final
0	14.0	359	20.7	82.1	1995	4.7
1	44.3	1136	34.5	285.3	6728	6.1
2	37.6	976	22.3	143.6	3415	5.9
3	13.3	343	21.6	186.3	4350	4.8
4	8.9	228	21.5	53.9	1267	5.9
5	15.5	394	17.2	46.2	1125	6.4
6	8.5	215	25.3	89.6	2063	7.5
7	11.7	293	19.3	289.4	6738	4.9
8	8.5	222	19.0	19.3	493	10.0
9	7.2	189	24.0	64.0	1542	6.4
Promedios	16.9	435.5	22.5	126.0	2971.6	6.3

Tabla B.8: Resultados de las diez corridas del método AG con diferentes semillas para la instancia uno de la configuración dos.

Instancia 2

Corrida	Tiempo a primera solución (s)	Generaciones a primera solución	Fitness de primera solución	Tiempo a solución final (s)	Generaciones a solución final	Fitness de solución final
0	3.5	90	27.2	129.9	3135	7.2
1	9.1	237	24.0	188.5	4629	8.5
2	10.7	281	18.8	271.3	6680	8.2
3	8.8	230	23.2	209.6	5035	6.3
4	9.3	242	14.2	90.2	2258	8.3
5	19.1	480	17.9	210.3	5135	7.6
6	8.9	233	27.3	67.6	1725	7.7
7	37.9	973	20.0	205.8	4900	7.5
8	7.3	190	27.2	99.1	2394	8.2
9	26.6	670	21.0	193.3	4619	7.2
Promedios	14.1	362.6	22.1	166.6	4051.0	7.7

Tabla B.9: Resultados de las diez corridas del método AG con diferentes semillas para la instancia dos de la configuración dos.

Instancia 3

Corrida	Tiempo a primera solución (s)	Generaciones a primera solución	Fitness de primera solución	Tiempo a solución final (s)	Generaciones a solución final	Fitness de solución final
0	6.2	161	29.3	60.3	1417	5.9
1	13.5	350	31.6	138.4	3281	5.0
2	6.7	177	29.2	257.5	5968	5.3
3	7.9	205	21.6	126.0	3028	3.7
4	5.8	152	16.9	36.6	910	3.7
5	36.6	951	21.5	61.8	1557	5.7
6	6.4	169	20.8	39.0	943	4.3
7	6.5	171	26.6	299.3	6534	4.8
8	10.6	276	19.8	191.8	4402	4.0
9	5.8	152	17.7	149.4	3553	7.3
Promedios	10.6	276.4	23.5	136.0	3159.3	5.0

Tabla B.10: Resultados de las diez corridas del método AG con diferentes semillas para la instancia tres de la configuración dos.

Instancia 4

Corrida	Tiempo a primera solución (s)	Generaciones a primera solución	Fitness de primera solución	Tiempo a solución final (s)	Generaciones a solución final	Fitness de solución final
0	9.2	239	27.2	124.8	2964	3.5
1	16.6	430	25.0	290.0	6076	3.5
2	5.3	137	20.5	81.4	1916	5.3
3	4.5	117	21.5	100.5	2429	5.7
4	15.9	411	21.6	39.3	975	2.3
5	19.6	506	17.7	60.9	1497	4.1
6	5.8	152	22.7	88.2	2124	4.2
7	106.3	2736	21.3	124.0	3182	10.6
8	8.2	215	20.6	73.4	1852	7.2
9	18.2	469	29.4	290.4	6858	5.4
Promedios	21.0	541.2	22.8	127.3	2987.3	5.2

Tabla B.11: Resultados de las diez corridas del método AG con diferentes semillas para la instancia cuatro de la configuración dos.

Instancia 5

Corrida	Tiempo a primera solución (s)	Generaciones a primera solución	Fitness de primera solución	Tiempo a solución final (s)	Generaciones a solución final	Fitness de solución final
0	10.7	275	15.2	22.5	566	10.1
1	4.6	122	18.3	52.5	1335	10.1
2	10.7	274	11.2	46.3	1164	10.1
3	11.5	303	15.8	14.5	380	10.1
4	11.7	306	15.8	46.8	1187	10.1
5	11.2	292	16.5	21.3	541	10.1
6	16.6	435	16.9	132.1	3307	10.1
7	4.8	130	14.0	57.1	1419	10.1
8	12.5	331	17.0	183.1	4594	10.1
9	12.2	319	13.7	31.9	809	10.1
Promedios	10.6	278.7	15.4	60.8	1530.2	10.1

Tabla B.12: Resultados de las diez corridas del método AG con diferentes semillas para la instancia cinco de la configuración dos.

B.3. Configuración 3

B.3.1. Resultados CPLEX

Instancia	Tiempo a primera solución (s)	Función objetivo de primera solución	Tiempo a solución final (s)	Función objetivo de solución final
1	75.5	16.7	300.0	5.9
2	40.4	24.9	40.4	24.9
3	35.0	22.7	300.0	22.7
4	65.2	17.7	300.0	3.0
5	79.2	31.9	81.2	8.1
Promedio	59.1	22.8	204.3	12.9

Tabla B.13: Resultados de la configuración tres de pruebas para CPLEX.

B.3.2. Resultados método AG

Instancia 1

Corrida	Tiempo a primera solución (s)	Generaciones a primera solución	Fitness de primera solución	Tiempo a solución final (s)	Generaciones a solución final	Fitness de solución final
0	267.7	5754	19.0	296.7	6351	11.9
1	47.3	1025	23.5	227.2	4476	9.4
2	-	-	-	-	-	-
3	133.8	2876	27.9	236.7	4850	4.8
4	57.8	1251	25.9	207.5	4370	6.1
5	206.1	4450	19.2	244.0	5251	10.8
6	152.4	3260	27.2	163.6	3501	11.0
7	28.6	613	30.5	225.5	4340	6.0
8	291.4	6257	23.2	298.4	6406	11.4
9	-	-	-	-	-	-
Promedios	148.2	3185.8	24.6	237.5	4943.1	8.9

Tabla B.14: Resultados de las diez corridas del método AG con diferentes semillas para la instancia uno de la configuración tres.

Instancia 2

Corrida	Tiempo a primera solución (s)	Generaciones a primera solución	Fitness de primera solución	Tiempo a solución final	Generaciones a solución final (s)	Fitness de solución final
0	97.8	2115	11.8	295.2	5868	5.0
1	281.1	6005	21.2	297.2	6345	11.1
2	114.4	2453	26.1	295.0	5957	5.3
3	62.8	1359	25.0	258.6	5297	6.0
4	206.4	4415	23.0	230.3	4922	8.7
5	35.9	775	20.6	281.7	5475	5.0
6	-	-	-	-	-	-
7	182.7	3901	16.9	283.5	5837	6.3
8	36.0	764	21.4	149.1	2990	5.3
9	67.3	1455	20.9	130.4	2705	5.0
Promedios	120.5	2582.4	20.8	246.8	5044.0	6.4

Tabla B.15: Resultados de las diez corridas del método AG con diferentes semillas para la instancia dos de la configuración tres.

Instancia 3

Corrida	Tiempo a primera solución (s)	Generaciones a primera solución	Fitness de primera solución	Tiempo a solución final (s)	Generaciones a solución final	Fitness de solución final
0	43.2	926	20.7	226.5	4765	8.1
1	131.0	2797	32.7	289.2	5930	4.9
2	125.6	2690	23.2	278.7	5600	6.8
3	58.2	1256	23.2	118.1	2447	6.5
4	184.2	3951	22.3	271.6	5679	8.6
5	107.6	2305	15.8	251.1	5148	5.4
6	41.7	883	27.4	248.0	4836	6.8
7	46.9	1001	24.6	127.3	2574	6.2
8	48.4	1031	24.1	286.6	5753	6.6
9	157.1	3357	25.2	244.8	5057	7.3
Promedios	94.4	2019.7	23.9	234.2	4778.9	6.7

Tabla B.16: Resultados de las diez corridas del método AG con diferentes semillas para la instancia tres de la configuración tres.

Instancia 4

Corrida	Tiempo a primera solución (s)	Generaciones a primera solución	Fitness de primera solución	Tiempo a solución final (s)	Generaciones a solución final	Fitness de solución final
0	36.4	781	19.7	237.4	4760	4.4
1	24.4	529	32.5	57.2	1214	10.3
2	32.0	700	19.7	233.2	4602	6.2
3	117.9	2548	21.5	188.2	3970	7.9
4	123.2	2653	23.3	280.0	5740	3.5
5	88.3	1898	19.4	135.6	2848	6.7
6	24.7	539	23.7	111.4	2276	4.6
7	61.0	1312	31.3	122.9	2549	10.4
8	51.7	1117	22.6	163.9	3324	4.9
9	150.5	3267	13.0	159.0	3451	9.0
Promedios	71.0	1534.4	22.7	168.9	3473.4	6.8

Tabla B.17: Resultados de las diez corridas del método AG con diferentes semillas para la instancia cuatro de la configuración tres.

Instancia 5

Corrida	Tiempo a primera solución (s)	Generaciones a primera solución	Fitness de primera solución	Tiempo a solución final (s)	Generaciones a solución final	Fitness de solución final
0	-	-	-	-	-	-
1	96.0	2022	28.2	271.8	5306	2.6
2	135.6	2905	24.9	253.3	5187	4.5
3	22.3	480	27.5	237.6	4660	5.7
4	-	-	-	-	-	-
5	-	-	-	-	-	-
6	143.0	3064	25.3	298.8	6152	8.9
7	55.9	1197	22.3	172.2	3457	2.0
8	167.4	3555	24.4	283.1	5847	10.7
9	132.7	2838	25.5	261.3	5284	4.7
Promedios	107.6	2294.4	25.4	254.0	5127.6	5.6

Tabla B.18: Resultados de las diez corridas del método AG con diferentes semillas para la instancia cinco de la configuración tres.

B.4. Configuración 4

B.4.1. Resultados CPLEX

Instancia	Tiempo a primera solución (s)	Función objetivo de primera solución	Tiempo a solución final (s)	Función objetivo de solución final
1	5.5	17.1	300.0	6.8
2	29.7	14.2	30.3	11.6
3	37.9	23.1	36.5	23.1
4	26.7	15.0	168.0	12.8
5	28.6	14.7	28.8	12.2
Promedio	25.7	16.8	112.7	13.3

Tabla B.19: Resultados de la configuración cuatro de pruebas para CPLEX.

B.4.2. Resultados método AG

Instancia 1

Corrida	Tiempo a primera solución (s)	Generaciones a primera solución	Fitness de primera solución	Tiempo a solución final (s)	Generaciones a solución final	Fitness de solución final
0	-	-	-	-	-	-
1	17.4	377	15.6	275.7	5456	4.1
2	8.2	181	11.7	182.4	3415	2.5
3	13.0	283	19.8	173.5	3480	3.1
4	8.3	183	19.1	107.7	2099	1.9
5	13.3	290	16.5	233.9	4416	1.9
6	12.1	264	20.6	249.3	4833	3.9
7	118.0	2570	17.9	299.2	5966	2.7
8	11.7	257	23.0	144.5	2862	2.6
9	10.2	226	17.2	259.3	4957	1.7
Promedios	23.6	514.6	17.9	213.9	4164.9	2.7

Tabla B.20: Resultados de las diez corridas del método AG con diferentes semillas para la instancia uno de la configuración cuatro.

Instancia 2

Corrida	Tiempo a primera solución (s)	Generaciones a primera solución	Fitness de primera solución	Tiempo a solución final (s)	Generaciones a solución final	Fitness de solución final
0	30.3	666	20.2	75.7	1576	3.8
1	4.2	93	26.8	173.1	3445	2.8
2	12.8	283	17.8	189.6	3729	2.0
3	15.9	348	18.6	163.2	3262	3.6
4	12.7	284	18.4	175.6	3408	3.1
5	152.7	3287	21.6	258.5	5260	3.1
6	95.7	2076	27.9	188.9	3971	3.6
7	16.2	357	19.6	166.4	3337	3.3
8	4.9	108	20.4	220.4	4317	4.0
9	5.9	130	26.5	107.0	2167	2.9
Promedios	35.1	763.2	21.8	171.8	3447.2	3.2

Tabla B.21: Resultados de las diez corridas del método AG con diferentes semillas para la instancia dos de la configuración cuatro.

Instancia 3

Corrida	Tiempo a primera solución (s)	Generaciones a primera solución	Fitness de primera solución	Tiempo a solución final (s)	Generaciones a solución final	Fitness de solución final
0	36.6	799	21.0	286.8	5338	4.3
1	35.1	763	22.2	71.8	1511	5.1
2	13.3	293	18.1	105.9	2175	2.6
3	19.7	434	21.5	236.7	4746	2.3
4	12.5	277	22.9	100.1	1983	3.7
5	39.9	876	28.5	139.8	2904	4.2
6	30.5	664	26.2	60.0	1281	4.1
7	48.6	1050	29.6	283.9	5498	3.2
8	195.0	4205	25.0	245.6	5250	6.7
9	7.7	170	13.5	195.2	3712	1.9
Promedios	43.9	953.1	22.8	172.6	3439.8	3.8

Tabla B.22: Resultados de las diez corridas del método AG con diferentes semillas para la instancia tres de la configuración cuatro.

Instancia 4

Corrida	Tiempo a primera solución (s)	Generaciones a primera solución	Fitness de primera solución	Tiempo a solución final (s)	Generaciones a solución final	Fitness de solución final
0	11.6	247	22.7	85.1	1706	2.9
1	80.2	1730	15.6	203.5	4069	2.3
2	92.2	1987	19.7	219.9	4282	2.3
3	17.5	378	17.9	66.9	1356	2.8
4	11.3	246	13.4	283.8	5530	2.0
5	9.4	208	25.3	124.8	2377	3.1
6	9.4	208	24.5	250.7	4780	1.2
7	18.7	405	24.6	115.4	2169	2.6
8	21.0	457	16.8	212.9	4069	3.1
9	17.2	375	18.6	173.8	3240	3.1
Promedios	28.9	624.1	19.9	173.7	3357.8	2.5

Tabla B.23: Resultados de las diez corridas del método AG con diferentes semillas para la instancia cuatro de la configuración cuatro.

Instancia 5

Corrida	Tiempo a primera solución (s)	Generaciones a primera solución	Fitness de primera solución	Tiempo a solución final (s)	Generaciones a solución final	Fitness de solución final
0	22.5	494	16.0	223.5	4325	2.9
1	9.8	217	17.2	111.9	2239	1.9
2	13.0	287	15.5	276.1	5282	1.9
3	18.5	412	22.0	103.6	2116	3.2
4	61.2	1341	19.9	142.3	2976	2.4
5	44.9	986	12.7	93.8	1962	2.8
6	10.5	233	19.0	75.5	1534	2.9
7	68.1	1489	15.4	276.0	5231	2.7
8	13.8	306	24.8	143.7	2807	1.8
9	8.7	193	15.7	294.8	5369	2.0
Promedios	27.1	595.8	17.8	174.1	3384.1	2.4

Tabla B.24: Resultados de las diez corridas del método AG con diferentes semillas para la instancia cinco de la configuración cuatro.

B.5. Configuración 5

B.5.1. Resultados CPLEX

Instancia	Tiempo a primera solución (s)	Función objetivo de primera solución	Tiempo a solución final (s)	Función objetivo de solución final
1	5.1	13.5	299.0	2.9
2	6.1	10.3	272.5	3.2
3	6.2	11.7	300.0	2.9
4	6.1	13.1	293.5	2.8
5	15.4	13.6	275.7	3.0
Promedio	7.8	12.4	288.1	3.0

Tabla B.25: Resultados de la configuración cinco de pruebas para CPLEX.

B.5.2. Resultados método AG

Instancia 1

Corrida	Tiempo a primera solución (s)	Generaciones a primera solución	Fitness de primera solución	Tiempo a solución final (s)	Generaciones a solución final	Fitness de solución final
0	5.1	132	18.7	47.6	1136	2.9
1	48.2	1254	14.9	72.7	1830	2.5
2	14.3	376	10.6	219.3	5028	1.2
3	13.1	345	16.8	164.4	3854	1.5
4	49.1	1267	19.3	108.0	2690	3.5
5	7.7	204	17.6	241.6	5425	3.0
6	10.1	264	17.5	22.0	556	5.1
7	48.6	1255	15.7	128.4	3109	2.2
8	7.2	192	24.9	182.8	4284	1.8
9	23.4	611	14.5	67.1	1620	4.3
Promedios	22.7	590.0	17.0	125.4	2953.2	2.8

Tabla B.26: Resultados de las diez corridas del método AG con diferentes semillas para la instancia uno de la configuración cinco.

Instancia 2

Corrida	Tiempo a primera solución	Generaciones a primera solución	Fitness de primera solución	Tiempo a solución final	Generaciones a solución final	Fitness de solución final
0	23.3	606	26.9	153.5	3416	4.2
1	14.2	376	18.1	162.2	3825	4.1
2	13.4	352	18.0	110.4	2598	3.4
3	9.0	239	18.1	174.6	4083	2.8
4	4.7	126	19.3	282.5	6628	2.5
5	53.3	1391	21.1	99.7	2512	3.2
6	13.2	351	20.0	78.1	1874	3.7
7	7.1	190	11.4	132.7	2982	3.7
8	-	-	-	-	-	-
9	5.9	157	16.8	123.6	2767	2.0
Promedios	16.0	420.9	18.9	146.4	3409.4	3.3

Tabla B.27: Resultados de las diez corridas del método AG con diferentes semillas para la instancia dos de la configuración cinco.

Instancia 3

Corrida	Tiempo a primera solución (s)	Generaciones a primera solución	Fitness de primera solución	Tiempo a solución final (s)	Generaciones a solución final	Fitness de solución final
0	40.7	1068	11.0	71.6	1786	1.8
1	32.3	845	16.0	265.3	6136	2.3
2	98.4	2562	20.0	256.4	6428	5.5
3	27.4	719	11.0	47.1	1184	4.2
4	13.3	351	24.2	181.0	4155	3.0
5	11.3	299	16.4	238.4	5572	2.8
6	10.9	290	19.9	137.8	3350	4.2
7	22.9	605	14.9	77.2	1881	3.3
8	95.9	2489	18.6	148.0	3702	3.5
9	27.7	728	11.8	60.5	1561	10.8
Promedios	38.1	995.6	16.4	148.3	3575.5	4.1

Tabla B.28: Resultados de las diez corridas del método AG con diferentes semillas para la instancia tres de la configuración cinco.

Instancia 4

Corrida	Tiempo a primera solución (s)	Generaciones a primera solución	Fitness de primera solución	Tiempo a solución final (s)	Generaciones a solución final	Fitness de solución final
0	-	-	-	-	-	-
1	54.3	1395	21.2	292.0	6240	3.1
2	-	-	-	-	-	-
3	242.5	6231	17.1	263.9	6746	5.5
4	56.2	1454	14.2	85.0	2128	3.3
5	11.5	302	18.7	103.5	2447	5.5
6	116.7	3008	22.6	206.5	5113	2.2
7	159.7	4111	22.3	197.5	5066	6.6
8	-	-	-	-	-	-
9	97.7	2513	28.9	285.8	6947	3.3
Promedios	105.5	2716.3	20.7	204.9	4955.3	4.2

Tabla B.29: Resultados de las diez corridas del método AG con diferentes semillas para la instancia cuatro de la configuración cinco.

Instancia 5

Para la instancia 5 el algoritmo genético no llegó a ninguna solución.

Encuesta

El objetivo de la encuesta es estimar la convocatoria de publico esperada para cada agrupación. Es importante notar que no se refiere al desempeño, calidad o preferencia de las agrupaciones si no a la asistencia a la actuación.

Como forma de realizar la encuesta se presento a través de Google Docs donde cada grupo califica a los grupos que no son de su categoría. De la siguiente manera:

Por favor, califique a todas las agrupaciones a las que usted NO PERTENECE

Por favor, califique a las agrupaciones de Parodistas de acuerdo a la cantidad de público que espera que convoquen

	Alta	Media	Baja	Soy integrante de este conjunto
Bubys Bis				
Zero				
Quijotes				
Celestinos				
Wizards				
Klau's				
Wonkas				
Zabritos				
Indigos				
Principes				
Zodiaco				
Cherrys				

Por favor, califique a las agrupaciones de Revistas de acuerdo a la cantidad de público que espera que convoquen

	Alta	Media	Baja	Soy integrante de este conjunto
Dulcinea				
Saphirus				
Mandala				
Los Buscadores				
Frenesi				
Forever Kids				
Némesis				

Por favor, califique a las agrupaciones de Murgas de acuerdo a la cantidad de público que espera que convoquen

	Alta	Media	Baja	Soy integrante de este conjunto
Los Pepinitos				
La Descocada				
La Zafada				
Mano a Mano				
La Resiliente				

Por favor, califique a las agrupaciones de Humoristas de acuerdo a la cantidad de público que espera que convoquen

	Alta	Media	Baja	Soy integrante de este conjunto
Los Chapitas				
Yoguis				
Gnomos				
Herederos				
Deskara2				
Colibriquis				

Por favor, califique a las agrupaciones de Lubolos de acuerdo a la cantidad de público que espera que convoquen

	Alta	Media	Baja	Soy integrante de este conjunto
Magia Candombe				
Cosme y Damian				
Ohana				

Resultados de la encuesta de convocatoria

Agrupacion	Alta	Media	Baja	Ponderacion
Bubys Bis	29	2	1	9.44
Zero	3	24	5	5.44
Quijotes	31	1	0	9.88
Celestinos	16	14	2	7.63
Wizards	3	26	3	5.81
Klau's	5	21	6	5.50
Wonkas	0	18	14	3.38
Zabritos	0	7	25	1.31
Indigos	0	4	27	0.77
Principes	1	6	25	1.44
Zodiaco	2	9	20	2.39
Cherrys	1	9	22	2.00
Dulcinea	30	3	0	9.64
Saphirus	29	3	0	9.63
Mandala	6	16	11	4.73
Los Buscadores	0	12	20	2.25
Frenesi	2	6	24	1.75
Forever Kids	0	11	22	2.00
Némesis	0	13	20	2.36
Los Pepinitos	25	7	0	9.13
La Descocada	22	9	1	8.56
La Zafada	23	9	0	8.88
Mano a Mano	28	4	0	9.50
La Resiliente	1	14	17	2.94
Los Chapitas	19	11	2	8.00
Yoguis	11	17	4	6.63
Gnomos	13	18	2	7.21
Herederos	6	14	13	4.36
Deskara2	1	9	22	2.00
Colibriquis	0	14	18	2.63
Magia Candombe	3	17	12	4.13
Cosme y Damian	6	24	2	6.38
Ohana	13	17	2	7.25

Reglamento del Carnaval de las promesas

En esta sección se mencionan algunos de los artículos del reglamento del Carnaval de las Promesas de la edición 2024/2025 que se considera mas relevantes para el caso de estudio.

Artículo 1o- Misión y Visión: El Departamento de Cultura de la Intendencia de Montevideo, a través de su Gerencia de Festejos y Espectáculos, organiza el evento “Carnaval de las Promesas” en coordinación con la Asociación de Directores del Carnaval de las Promesas (A.DI.CA.PRO), con el objetivo de promover la inclusión social y la integración de niñas, niños y adolescentes de todo el territorio de Montevideo y área metropolitana. El objetivo de la integración social se busca plasmar a través del carnaval y su diversidad de expresiones artísticas, como excelente oportunidad para la generación de valores e identidad cultural. De este modo, promoviendo la formación artística a través del Carnaval, también se buscará fomentar y promover políticas específicas destinadas a infancia y juventud, como segmentos etarios principales para las políticas culturales del Departamento de Cultura. Dichas políticas contemplarán la formación en valores democráticos de participación, diversidad, tolerancia y perspectiva de género. En este último punto resulta fundamental el cuidado de nuestras niñas y adolescentes tendiendo a la resignificación del lugar de las mismas en el Carnaval, deconstruyendo roles y estereotipos femeninos heredados, para construir nuevos roles femeninos desde una perspectiva de igualdad de género. A través de la etapa de Monitoreo, del Encuentro evaluatorio inicial, y del Encuentro de Agrupaciones Infantiles, se pretende combinar una serie de actividades de carácter participativo, formativo y de socialización que deberán encontrar en el Encuentro simplemente una excusa para juntarse a aprender y a disfrutar del carnaval como ámbito de integración y pertenencia.

Artículo 2o - Los eventos que constituyen el Carnaval de las Promesas 2024 serán: Monitoreo, Encuentro Evaluatorio y el Encuentro de Agrupaciones Infantiles, a desarrollarse en el Teatro de Verano “Ramón Collazo” en fecha a definir, rigiéndose en un todo por la presente reglamentación. El Encuentro de Agrupaciones Infantiles constará de dos ruedas en las que participarán todas las agrupaciones inscriptas. La administración de estos eventos, será res-

ponsabilidad de la Gerencia de Festejos y Espectáculos en conjunto con ADI-CAPRO y se regirán por las normas que están encuadradas dentro de las disposiciones de la Constitución de la República, de las Leyes Nacionales y Decretos u Ordenanzas Departamentales y Municipales, especialmente en lo referente a la protección de niños, niñas y adolescentes. En particular, por el Protocolo de prevención y protección ante la violencia hacia las niñas, niños y adolescentes en el carnaval de las Promesas aprobado por Resolución N° 5083/21 de fecha 22/12/21.

Artículo 18o- La Gerencia de Festejos y Espectáculos designará al equipo de jurados/monitores que actuarán en el Encuentro Evaluatorio, en el Encuentro en el Teatro de Verano, así como en la etapa de Monitoreo. En fecha y lugar a determinar por la Gerencia de Festejos y Espectáculos, se desarrollará el encuentro inicial, en el que participarán todos los conjuntos inscriptos para participar del Carnaval de las Promesas 2024. Dicho encuentro tiene por objetivos: - Un primer acercamiento al trabajo desarrollado hasta entonces por cada agrupación; - El encuentro de todas y todos los integrantes de las diversas agrupaciones en una instancia en la que todos participan sin carácter competitivo; - Una evaluación del trabajo de cada agrupación por parte del equipo del Jurado con la finalidad de hacer una devolución a cada agrupación (participantes y adultos que trabajan junto a los y las participantes) para potenciar las fortalezas y realizar sugerencias para mejorar las debilidades encontradas en cada propuesta. El objetivo de estas devoluciones es de carácter formativo y apuntarán a que las propuestas se adecúen a la Misión que el Carnaval de las Promesas tiene según el Artículo 1o del presente reglamento.

Artículo 19o- Finalizadas las presentaciones del Encuentro Evaluatorio, el jurado realizará las respectivas devoluciones a las agrupaciones.

Artículo 20o- La Gerencia de Festejos y Espectáculos acordará con A.DI.CA.PRO. y con los y las integrantes del Jurado actuante, los días, horarios y lugares en los cuales se desarrollará la devolución a las agrupaciones.

Artículo 21o- Las etapas del Encuentro Evaluatorio serán conformadas por la Gerencia de Festejos y Espectáculos en acuerdo con A.DI.CA.PRO. Las actuaciones de los conjuntos tendrán las siguientes duraciones en todas las categorías: máximo 20 minutos.

Artículo 22o- Las actuaciones tendrán el carácter de muestra de proceso de trabajo, por lo cual los conjuntos no podrán actuar con vestuario, ni escenografía, ni maquillaje artístico, ni recurrir a objetos de mediano y gran porte en sus escenas o coreografías. Estará permitido el uso de objetos de utilería que sean imprescindibles para el desarrollo de la actuación, así como el uso de maquillaje de tipo social o vestimenta de ensayo (equipos deportivos o remeras identificatorias de la agrupación).

Artículo 23o- Luego del primer acercamiento en el Encuentro Evaluatorio, el equipo de jurados/monitores realizará una recorrida por los ensayos de todos los conjuntos inscriptos para participar del Carnaval de las Promesas 2024. El equipo visitará los ensayos para evaluar las condiciones de trabajo de cada conjunto y realizar aportes y sugerencias para la mejora de los mismos de acuerdo a lo observado en el Encuentro Evaluatorio. En caso de considerarlo necesario, el equipo de monitoreo podrá realizar más de una visita a los conjuntos que encuentren con mayores dificultades en el proceso de trabajo. Al finalizar la recorrida por los ensayos de los conjuntos, el equipo de monitores deberá realizar un informe escrito sobre lo evaluado acerca del proceso de cada conjunto.

En paralelo a este trabajo, la IdeM establecerá un servicio de supervisión para el cumplimiento de todo lo dispuesto en el Protocolo de prevención de situaciones de violencia en el Carnaval de las Promesas. Este servicio cumplirá el rol de vinculación con los equipos institucionales que abordarán el seguimiento de situaciones de violencia detectadas o denunciadas ante el mismo, o ante cualquier parte involucrada en la organización del Carnaval de las Promesas.

Artículo 24o - Sociedades de Negros y Lubolos: Constituyen una recreación libre de las manifestaciones afrouuguayas con sus trajes, cantos y bailes típicos, evocando los orígenes afro y su devenir natural y acompasado a la actualidad en vestimenta, coreografía y temática, sin perder la esencia conceptual del candombe y su evolución, no limitando la creación artística y su fusión argumental y musical. El espectáculo se desarrollará bajo el signo predominante del tambor, aunque se admitirá la utilización de otros instrumentos musicales, fuera de los de percusión. Las letras y la música de los temas serán inéditas, debiendo regirse por la definición “inédita” contenida en el artículo 46o de este reglamento excepto en la despedida final del conjunto, que podrá utilizar el canto y la música tradicional del conjunto. Las propuestas podrán transmitir la evolución natural de la tradición del candombe. Los tambores podrán utilizar tensores, pero queda terminantemente prohibido la utilización de parches de “nylon” o similares, pudiendo solamente utilizar lonja. Deberán contar como mínimo con una (1) bailarina principal; un (1) escobero/escobera; un (1) gramillero; una (1) mama vieja, seis (6) bailarinas, quienes danzarán acompañando los cuadros que se representen; dos (2) portabanderas, un (1) portaestandarte; una (1) estrella; una (1) medialuna; un (1) cantante y nueve (9) tamborileros/tamborileras. Se pueden contar con un (1) bailarín, pero no es obligatorio. Integrantes: mínimo de veinticinco (25) y un máximo de sesenta y cinco (65). Estos/as integrantes deberán representar como mínimo siete (7) edades. Sus actuaciones tendrán una duración máxima de cuarenta (40) minutos.

Artículo 25o - Revistas: La categoría Revistas debe constituir una expresión artística integral de libre creación, conceptualmente imaginativa, tendiente a la diversión. Sin perjuicio de su característica de libre creación, se entiende que una Revista prioriza la alegría, la música y el baile. Los textos deberán ser inéditos, debiendo regirse por la definición “inédita”, contenida en el artículo 46o de este reglamento. Podrán utilizarse músicas grabadas. Armonizará coreografías,

vestimenta, bailes, música, orquestaciones, canciones y parlamentos con marcada alegría, en una sucesión de cuadros enlazados que eviten la interrupción del espectáculo, dotándolo de continuidad y dinamismo. Estos cuadros alternarán lo artístico con lo divertido dentro de un clima alegre y colorido. Las letras serán inéditas. Las melodías musicales podrán ser inéditas. Integrantes: mínimo dieciocho (18) máximo treinta y cinco (35). Estos integrantes deberán representar como mínimo siete (7) edades. Sus actuaciones tendrán una duración máxima de cuarenta y cinco (45) minutos.

Artículo 26o - Parodistas: Esta categoría deberá parodiar el argumento de obras, historias de hechos y/o personas de público y notorio conocimiento, en una imitación burlesca, realizada en tono jocoso, pudiendo, en determinados pasajes del espectáculo, tener matices dramáticos según la propuesta de cada conjunto. Para desarrollar su espectáculo cada conjunto de parodistas deberá: tener textos inéditos, debiendo regirse por la definición “inédita” contenida en el artículo 46o de este reglamento; cantar, realizar bailes y coreografías, usar vestuario, utilizar maquillaje, hacer una o varias parodias dentro del mismo. La música podrá ser interpretada por los propios integrantes o utilizando pistas grabadas. Los cuadros de presentación y despedida serán optativos de cada conjunto y, en caso de realizarse, podrán ser o no parodias. Integrantes: un mínimo de dieciocho (18), y un máximo de treinta y dos (32). Estos integrantes deberán representar como mínimo siete (7) edades. Sus actuaciones tendrán una duración máxima de cuarenta y cinco (45) minutos.

Artículo 27o - Murgas: Es una manifestación artística que refleja y representa diversos aspectos de la identidad uruguaya. El canto, la crítica, la sátira, la jocosidad, la picardía y la ironía son los pilares en que se sustenta el desarrollo artístico del género murguero, apostando fundamentalmente a la risa. El vestuario será alegre y colorido y el rostro deberá lucir maquillaje. Distingue a la Murga la mímica, la pantomima, la vivacidad, el movimiento, el contraste, la informalidad escénica y lo grotesco. La sincronización de movimientos se considerará válida si ésta diera mayor vuelo creativo al espectáculo y no fuera en contra de la informalidad característica de la Murga. Los textos deberán ser inéditos, debiendo regirse por la definición “inédita” contenida en el artículo 46o de este reglamento. Los instrumentos esenciales serán el platillo, bombo y redoblante, pudiendo utilizarse libremente otros instrumentos en carácter de apoyo. Integrantes: mínimo diecisiete (17) máximo veintinueve (29). Estos integrantes deberán representar como mínimo siete (7) edades. Sus actuaciones tendrán una duración máxima de cuarenta (40) minutos.

Artículo 28o- Humoristas: Esta categoría se basará en la libre comicidad de escenas, situaciones o personajes no pudiendo basarse en argumentos de una obra literaria, hecho o suceso real. Podrá utilizarse una creación jocosa única o plantearse varios cuadros con pequeños intervalos muy ágiles, enmarcados siempre en una faz cómica. El planteo del espectáculo debe tender, con ingenio y creatividad, a descubrir y presentar cuadros, personajes, situaciones, sucesos y actos con características incongruentes y absurdas en donde predomi-

na el humor, la sátira, la picardía y la jocosidad. Los textos deberán ser inéditos, debiendo regirse por la definición “inédita”, contenida en el artículo 46o de este reglamento. Deberán utilizar canciones y música, ésta podrá ser interpretada por los propios integrantes, o en su defecto utilizando pistas. Integrantes: mínimo de dieciséis (16) y un máximo de veintinueve (29). Estos integrantes deberán representar como mínimo siete (7) edades. Sus actuaciones tendrán una duración máxima de treinta y cinco (35) minutos.

Artículo 29o - Del Jurado del Encuentro: El Jurado del Encuentro de Agrupaciones Infantiles, estará integrado por una persona que oficiará de Coordinador/a del equipo, tres (3) integrantes titulares y un/una (1) funcionario/a administrativo/a que oficiará de Secretario/a del cuerpo, designados por la Gerencia de Festejos y Espectáculos de la Intendencia de Montevideo.

La designación de los integrantes del Jurado deberá contemplar el conocimiento de los mismos en materia de carnaval en general y de Carnaval de las Promesas en particular. También se ponderará su experiencia y formación artística vinculada al carnaval y a otros ámbitos de las artes y la cultura. Por último, resulta fundamental el perfil docente y pedagógico de quienes conformen el jurado, ya que el mismo deberá participar en instancias de evaluación con sus respectivas devoluciones, en las que se pondrá énfasis en el aspecto formativo de niños, niñas y adolescentes a través de las herramientas artísticas que cada categoría emplea para la creación de sus espectáculos.

Los y las integrantes del jurado no podrán tener vinculación directa con los conjuntos participantes del Carnaval de las Promesas durante su designación para la tarea. A.DI.CA.PRO designará un o una representante que oficiará de delegado/a entre los Directores y las Directoras Responsables de los conjuntos participantes y el Jurado, con el único fin de establecer los contactos que sean necesarios entre las partes.

Serán deberes de quien ejerza la Coordinación del Jurado junto con la Secretaría administrativa: a) Hacer cumplir el reglamento del Encuentro de Agrupaciones Infantiles en todos sus términos. b) Ordenar y fiscalizar el trabajo del jurado, así como realizar lo necesario para la unificación de criterios en el trabajo del mismo previo al inicio del encuentro. c) Ser responsable de la comunicación del Jurado con los diferentes conjuntos, así como con la prensa. d) Asegurar el correcto cumplimiento de los horarios establecidos para la participación de las agrupaciones y establecer las sanciones correspondientes en caso de incumplimiento. Será responsabilidad del Delegado o Delegada de ADICAPRO: a) Hacer conocer a el/la Coordinador/a del Jurado cualquier inquietud que algún o algunos Directores Responsables deseen evacuar. b) Controlar el cumplimiento del horario preestablecido para que el conjunto que termina de actuar retire todos sus elementos y entregue el escenario vacío al siguiente concursante e informar de ello a el/la Coordinador/a del Jurado. c) Convocar al o a los Directores o Directoras Responsables toda vez que el jurado lo solicite. d) En caso de ser requerido por el jurado, entregar las pistas utilizadas por los concursantes a la secretaría administrativa del Jurado, inmediatamente después de culminar la actuación. e) Recibir dos (2) copias de las notificaciones (sanciones, solicitudes,

respuestas, etc.) que cupieran durante el desarrollo del concurso, y entregarlas, una al conjunto involucrado y otra a ADICAPRO. f) Recibir y trasladar al equipo del Jurado cualquier denuncia hecha por algún concursante, oralmente en forma inmediata y por escrito 24 horas después. g) Estar presente en la comunicación de los resultados en nombre de ADICAPRO.

Artículo 33o - En cada rueda se hará un listado de la programación para conocimiento de los Directores y las Directoras Responsables de los conjuntos, indicándose día y hora en que deberán estar presentes en el local donde se efectúe el encuentro. Además, se procederá por parte de A.DI.CA.PRO. a la notificación de la convocatoria a cada uno, en forma individual y por escrito. Un funcionario o funcionaria designado/a por la Gerencia de Festejos y Espectáculos fiscalizará la hora de llegada de los distintos conjuntos con la totalidad de sus integrantes y el Director o Directora Responsable firmará un registro que acreditará la misma, aplicándose las sanciones correspondientes si se presentara fuera de los plazos establecidos. En caso que el/la Director/a no se presente el día de la actuación, y no haya comunicado a la Gerencia de Festejos y Espectáculos las razones justificadas en forma previa a través de ADICAPRO, se establecerán las sanciones establecidas en el artículo 40º.

Artículo 34o - En caso de que en una etapa queden conjuntos por concursar, debido a la suspensión por lluvia u otro motivo, pasarán a etapas restantes. La actuación de dichos conjuntos en las restantes etapas, se realizará el día que establezcan ADICAPRO y la Gerencia de Festejos y Espectáculos de mutuo acuerdo.

Artículo 35o- El Encuentro de Agrupaciones Infantiles Carnaval de las Promesas 2024, se habrá de realizar de la siguiente manera:

DOS RUEDAS, donde participarán todos los conjuntos habilitados por este reglamento.

RESULTADO FINAL Luego de concluida la actuación del último conjunto de la última etapa de la Segunda Rueda, el Jurado, conjuntamente con los funcionarios y las funcionarias que la Gerencia de Festejos y Espectáculos designe a tal efecto, el delegado o delegada de A.DI.CA.PRO., un/a Escribano/a de la IdEM, que labrará el acta respectiva y en presencia de la prensa oral, escrita y televisiva debidamente acreditada, se reunirán a efectos de emitir el resultado final. De inmediato se procederá a la apertura de la urna, se contarán los sobres, clasificándolos por categoría y por nombre de conjunto.

Acto seguido se procederá a la apertura de los sobres, contabilizando lo dictaminado por cada uno de los Jurados, en planillado de clasificación que a tales efectos proporcionará la Gerencia de Festejos y Espectáculos.

Las boletas de clasificación, fiscalizadas y contabilizadas, serán guardadas sin sufrir alteración de ninguna índole, a efectos de un eventual contralor posterior.

EL RESULTADO FINAL, SERÁ EL PRODUCTO DE LA SUMATORIA DE LOS PUNTOS DE LA PRIMERA Y SEGUNDA RUEDA.

EL FALLO DEL JURADO SERA INAPELABLE