UNIVERSITAT POLITÈCNICA DE VALÈNCIA

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Dpto. de Estadística e Investigación Operativa Aplicadas y Calidad

Diseño y análisis de la escalabilidad de modelos lineales enteros para el problema de la formación de equipos en el aula basado en características heterogéneas

Trabajo Fin de Máster

Máster Universitario en Ingeniería de Análisis de Datos, Mejora de Procesos y Toma de Decisiones

AUTOR/A: Tortosa Richart, Ana

Tutor/a: Sánchez Anguix, Víctor

Cotutor/a: Alberola Oltra, Juan Miguel

CURSO ACADÉMICO: 2023/2024

Agradecimientos

Quiero agradecer a mis amigos del Máster por haber convertido esta etapa en un recuerdo bonito y a ValgrAI (*Valencian Graduate School and Research Network for Artificial Intelligence*) y a la Generalitat Valenciana por su apoyo económico.

valgrAI
Valencian Graduate School
and Research Network of Artificial Intelligence

Resumen

El creciente interés por el trabajo colaborativo en contextos de aprendizaje está motivado por la efectividad que este puede tener frente al aprendizaje individual. A pesar de su importancia, hay una falta de apoyo para el proceso de la formación de los equipos. Una parte importante de este proceso es la elección de habilidades de los miembros que se considerarán para la formación de los equipos.

Si planteamos este problema, como es natural, como un problema de optimización, en estas habilidades se basará la heurística de evaluación del equipo, es decir, una aproximación a priori del desempeño real del equipo. No hay muchas evidencias de qué heurísticas de evaluación resultan en un buen desempeño real del equipo. Para poder comparar este desempeño entre distintas heurísticas de evaluación, se deben realizar pruebas A/B dentro del aula. Sin embargo, en estos estudios se debe reducir la variabilidad de los datos al mínimo, ya que, dentro de un aula, el número de equipos qué podemos formar no es muy elevado. Para ello, se deben utilizar métodos exactos que nos proporcionen la solución óptima del problema.

En este contexto, y con el objetivo de encontrar un modelo eficiente en tiempo de ejecución con el que podamos comparar el desempeño real de distintas heurísticas de evaluación, el presente estudio se enfoca en la propuesta, implementación y comparación, a nivel de tiempo de ejecución, de tres modelos de programación lineal entera destinados a abordar el desafío de la formación de equipos en entornos educativos. El primero de ellos, "modelo de elección de equipos", se basa en formar todos los equipos factibles y a partir de estos, escoger los que formarán parte en la partición de alumnos. El segundo, "modelo de asignación de tuplas a equipos", parte de un conjunto de equipos deseados que, en un principio, están vacíos y a los cuales vamos añadiendo tuplas de alumnos. Por último, el tercer modelo, "modelo de asignación de posiciones a equipos", también forma todos los equipos factibles como en el primer modelo y además de escoger los que formarán parte de la partición, les asigna la posición que ocupan en el conjunto de equipos deseados. Los resultados del estudio muestran que el "modelo de asignación de tuplas a equipos" es el más eficiente por varios órdenes de magnitud.

Palabras clave: Formación de equipos en el aula, modelos de optimización, roles de Belbin, heurística de evaluación del equipo

Abstract

The growing interest in collaborative work in learning contexts is motivated by its greater effectiveness compared to individual learning. Despite its importance, there is a lack of support for the team formation process. An important part of this process is the selection of the member skills that will be considered in the formation of the teams.

If we pose this problem as an optimization problem, the team's evaluation heuristic will be based on these skills. The team's evaluation heuristic is an a priori approximation of the team's real performance. There isn't much evidence of which evaluation heuristics result in actual good team performance. To compare this performance between different evaluation heuristics, A/B tests must be carried out within the classroom. However, in these studies the variability of the data must be reduced to a minimum, since, within a classroom, the number of teams we can form is not very high. To carry out these tests, we must use exact methods, that provide us with the optimal solution to the problem.

In this context, the present study focuses on the proposal, implementation and comparison at runtime level of three integer linear programming models intended to address the challenge of team building in educational environments. The purpose of this is finding an efficient model with which we can compare the real performance of different evaluation heuristics. The first one, "team selection model", is based on forming all feasible teams and from these, choosing those that will be part of the student division. The second, "assignment of tuples to teams model", starts from a set of desired teams that, at first, are empty and to which we add tuples of students. Finally, the third model, "assigning positions to teams model", also forms all the feasible teams as in the first model and in addition to choosing those that will be part of the partition, it assigns them the position they occupy in the set of desired teams. The results of the study show that the "assignment of tuples to teams model" is the most efficient by several orders of magnitude.

Keywords: Classroom team formation, optimization models, Belbin roles, team evaluation heuristics

Contenido

Resumen 1
Abstract 3
1. INTRODUCCIÓN 6
2. MARCO TEÓRICO 9
2.1. La formación de equipos 9
2.2. Los sistemas de apoyo a la formación de equipos 10
2.3. El problema de la formación de equipos (TFP) optimizada 13
2.4. El problema de la formación de equipos en el aula optimizada 14
3. METODOLOGÍA 23
4. MODELOS PROPUESTOS DE PROGRAMACIÓN LINEAL ENTERA 26
4.1. MODELO 1: modelo de elección de equipos 28
4.2. MODELO 2: modelo de asignación de tuplas a equipos 29
4.3. MODELO 3: modelo de asignación de posiciones a equipos 31
5. CASO DE ESTUDIO: TEORIA DE LOS ROLES DE BELBIN 34
5.1. Cuestionario belbin 35
5.2. Caso de estudio aplicado a los modelos 36
5.3. Datos del experimento 38
5.4. Generación de instancias 39
6. IMPLEMENTACIÓN DE LOS MODELOS 40
6.1. Herramientas utilizadas para la implementación 40
6.2. Organización del Código 41
6.3. Pseudo-código explicado del cálculo de parámetros 42
6.4. Pseudo-código explicado de los modelos 47
6.5. Pseudo-código explicado de la generación de instancias. 48
6.6. Pseudo-código explicado de los para la realización de los experimentos 49
7. EXPERIMENTOS 52
7.1. Resultados 52
7.2. Análisis del porcentaje de instancias resueltas por modelo 55
7.3. Análisis para aulas de 20 alumnos 56
7.4. Análisis para aulas de 30 alumnos 59
7.5. Análisis para aulas de 40 alumnos 63
7.6. Análisis para aulas de 50 alumnos 65
7.7. Discusión de los resultados 68
8. CONCLUSIONES 70
Bibliografía 72

ANEXO A: Tabla de resultados 75
ANEXO B: CÓDIGOS 78
GENERACIÓN DE INSTANCIAS SIN DEPENDENCIAS 78
GENERACIÓN DE INSTANCIAS CON DEPENDENCIAS (CON LOS PARÁMETROS ALPHA Y BETA) 79
SCRIPT EXPERIMENTO 82

1. INTRODUCCIÓN

En el ámbito educativo, la eficiente asignación de alumnos a equipos de trabajo es un desafío crucial. El aprendizaje colaborativo puede mejorar la calidad general y la eficiencia del aprendizaje individual de los alumnos dividiéndolos en diferentes equipos y alentándolos a ayudarse mutuamente (Ma, et al., 2022). Sin embargo, no todos los equipos facilitan el aprendizaje. Para que el aprendizaje en equipo sea efectivo, cada equipo compuesto en el aula debe ser heterogéneo, es decir, diverso en las características de los individuos. (Andrejczuk, Bistaffa, Blum, & Rodríguez-Aguilar, 2019). Estas características heterogéneas pueden ser tales como estilos de aprendizaje, habilidades sociales, conocimientos o roles, entre otras.

Esto se conoce como el problema de formación de equipos en el aula e implica encontrar las mejores combinaciones de miembros para formar equipos, es decir, encontrar las mejores particiones de los alumnos en equipos. La forma de particionar los alumnos en equipos puede considerarse de distintas formas. Es un desafío que se complica aún más cuando hay muchas combinaciones posibles. Supongamos que tenemos un total de *n* estudiantes y decidimos dividirlos en equipos de tamaño *k*. Vamos formando los equipos secuencialmente. Para escoger el primer equipo tendríamos una combinatoria similar a la selección de *k* elementos de un conjunto de *n* elementos, lo que se representa como $\binom{n}{k}$. A continuación, para escoger el segundo equipo tendríamos $\binom{n-k}{k}$ combinaciones distintas. Para formar el siguiente equipo tendríamos $\binom{n-2k}{k}$ combinaciones distintas. Sin embargo, cuando solo quedan 2 equipos por formar, escoger un conjunto de alumnos que formarán el siguiente equipo es equivalente a escoger el conjunto de alumnos complementarios ya que nos proporcionará la misma partición. De esta forma, la combinatoria total vendría dada por la siguiente fórmula:

$$\frac{\binom{n}{k} \cdot \binom{n-k}{k} \cdot \binom{n-2k}{k} \cdot \ldots \cdot \binom{n-n}{k}}{2}$$

Puedes tener restricciones adicionales, como asegurarte de que cada equipo tenga una combinación equilibrada de habilidades, personalidades, género, etc. Esto aumenta la complejidad del problema, ya que no solo estás considerando la combinación de estudiantes, sino también cómo cumplir con ciertos criterios o restricciones. Incluso una vez que hayas formado equipos, es posible que desees permutar los roles dentro de cada equipo (por ejemplo, asignar un líder, un secretario, un investigador, etc.). Esto agrega otra capa de combinatoria, ya que estás considerando diferentes configuraciones dentro de cada equipo.

Si bien una persona podría abordar manualmente el problema de la formación de equipos en un contexto educativo pequeño y simple, vemos que esto puede dificultarse enormemente por las razones ya comentadas. Para aulas grandes, como las universitarias, esta tarea se puede tornar complicada ya que haber desde 40 hasta 80 alumnos. Por lo tanto, se pueden utilizar técnicas de optimización para resolver este problema de manera eficiente.

Normalmente, el problema de la formación de equipos dentro del aula se basa en una función que estima el desempeño a priori del equipo basándose en diferentes criterios (estilos de aprendizaje, habilidades sociales, conocimientos, etc.), conocida como heurística de evaluación del equipo. Dentro de la optimización, la función que resulta de

sumar la heurística de evaluación de cada equipo desempeña el papel de la función objetivo, es decir, la función que debemos maximizar o minimizar. La heurística de evaluación es un proxy para la verdadera función, imposible de estimar, que es el desempeño real del equipo. De este modo, el buen funcionamiento de los equipos formados depende en gran parte de esta función/criterio empleado para formar los equipos. En el contexto de la formación de equipos en el aula, es muy común y existen diferentes heurísticas de evaluación de equipos que se basan en formar equipos con el máximo número de características posibles de un conjunto de características. Investigar qué heurísticas de evaluación llevan a buenos resultados es un área de investigación abierta.

Para poder determinar qué heurística de evaluación del equipo es la más apropiada en el contexto real del aula es necesario realizar experimentos reales o pruebas A/B en el aula, es decir, comparar el desempeño de varios equipos formados con distintas heurísticas de evaluación. Debido al pequeño número de equipos que se pueden formar dentro del aula, es crucial reducir la variabilidad en este tipo de estudio para aprovechar al máximo los datos disponibles. En la literatura, se han propuesto diversos enfoques, como algoritmos genéticos y heurísticas, para promover la homogeneidad entre los grupos y la diversidad dentro de ellos. Si bien estos enfoques son útiles para resolver problemas grandes, pueden no ser aconsejables para realizar experimentos para comparar la efectividad de varias heurísticas de evaluación en equipo en el aula ya que, debido a su naturaleza aproximada, pueden introducir ruido dentro de los experimentos realizados en el aula. Es decir, podemos estar comparando el desempeño de dos heurísticas de evaluación pero que este desempeño pueda estar sesgado porque en un caso el algoritmo ha encontrado una solución que se aproxima bastante a la solución óptima y en el otro caso esta solución se aleja bastante. Si esto ocurre, el desempeño puede verse afectado por este factor además de la heurística de evaluación usada. Por lo tanto, los métodos exactos, como la programación matemática, son preferibles en el contexto de estos experimentos. Idealmente, estos métodos exactos deben ser escalables para su uso en experimentos en aulas grandes. Los algoritmos de programación lineal entera (ILP) son adecuados para obtener soluciones óptimas en problemas complejos. Estos algoritmos nos permiten comparar varias heurísticas de evaluación de equipo de manera experimental. (Candel, Sánchez-Anguix, Alberola, Julián, & Botti, 2023)

A continuación, vemos los objetivos de este trabajo.

1. Proponer tres modelos exactos de programación lineal entera distintos capaces de encontrar la solución óptima del problema de la formación de equipos en el aula. Estos tres modelos, se basan en el criterio comentado anteriormente de buscar heterogeneidad de características. Estos nos permitirán formar grupos con diferentes aptitudes ya que procuraremos que las características estén bien distribuidas dentro de cada equipo y entre los equipos. Esto, además, evitará grandes diferencias entre los distintos grupos, que pueden llevar a una descompensación de equipos con desempeños muy distintos.

2. Implementar un algoritmo capaz de resolver el problema de la formación de equipos en el aula usando cada uno de los tres modelos propuestos. Este algoritmo tendrá como entrada el conjunto de alumnos, las dependencias entre alumnos y los tamaños de equipos permitidos.

3. Comparar la eficiencia a nivel computacional de estos tres modelos. Para evaluar y comparar estos modelos, vamos a realizar experimentos para analizar el tiempo de ejecución de los 3 modelos y compararlos usando métodos estadísticos. Como objetivo más concreto, buscaremos diferencias significativas en los tiempos de ejecución entre los 3 grupos definidos por el modelo usado para la resolución y discutiremos cuál de ellos es el más adecuado para realizar experimentos para comparar la efectividad de varias heurísticas de evaluación del equipo en el aula.

4. Buscar limitaciones de los modelos planteados a nivel de escalabilidad.

El trabajo está organizado como sigue. En la sección 2 veremos la revisión de la literatura. En la sección 3 detallaremos la metodología empleada. A continuación, en la sección 4, explicaremos los 3 modelos propuestos. En la sección 5 hablaremos de la teoría de roles de Belbin, que será el criterio que usaremos para la heurística de evaluación. En la sección 6 tenemos la implementación del modelo y como se han generado las instancias. En la sección 7 analizaremos los resultados obtenidos en el experimento. Por último, en la sección 8, comentaremos las conclusiones del estudio.

2. MARCO TEÓRICO

En esta sección sintetizaremos el conocimiento existente sobre el problema de la formación de equipos en el aula. Para ello, revisaremos estudios, investigaciones, teorías y enfoques relevantes que ya han sido publicados. Además, analizaremos estos estudios para identificar las lagunas existentes y poder proporcionar un sentido al objetivo de este trabajo. Empezaremos esta sección explicando qué es el problema de la formación de equipos y su importancia. A continuación, veremos las técnicas existentes de formación de equipos. Por último, introduciremos el problema de la formación de equipos optimizada y, específicamente, el problema de la formación de equipos en el aula optimizada.

2.1. La formación de equipos

Los equipos son un conjunto de dos o más individuos que interactúan de forma adaptativa, interdependiente y dinámica hacia un objetivo común y valorado. (Salas, Burke, & Cannon-Bowers, 2003). Aunque los equipos se consideran un subconjunto específico de grupos, los términos "grupo" y "equipos" se han usado indistintamente en la literatura (Osborne, et al., 2023).

La formación de un equipo es un proceso que considera la búsqueda, identificación y elección de los miembros de un equipo (Guimerà, Uzzi, Spiro, & Nunes Amaral, 2005) (Gómez-Zará, Dechurch, & Contractor, 2020). Este proceso es esencial para el ciclo de vida del desarrollo del grupo y ha sido una preocupación creciente para muchos investigadores. (Maqtary, Abdulqader, & Kamal, 2019)

Formar un equipo puede ser un desafío que se requiere en muchos ámbitos. Por ejemplo, en la formación de un equipo deportivo, de un grupo de estudiantes o de un grupo de trabajo en una empresa (Arrow, McGrath, & Berdahl, 2012). En un mundo donde el trabajo en equipo es cada vez más común y esencial, la capacidad de formar equipos efectivos se vuelve crucial. Sin embargo, este proceso no es trivial y enfrenta numerosos obstáculos. Entre ellos, la complejidad en la medición de atributos relevantes, la falta de conocimientos sobre los miembros del equipo, la diversidad de habilidades y personalidades, los conflictos interpersonales y la desigualdad en la participación (Arrow, McGrath, & Berdahl, 2012) (Katzenbach & Smith, 1993).

Los equipos disfuncionales pueden generar una serie de problemas y consecuencias negativas. En el ámbito deportivo, un equipo mal formado puede resultar en derrotas, disputas internas y un deterioro en el rendimiento general del equipo (Smith, Arthur, Hardy, & Callow, 2013). En el contexto educativo, los equipos mal equilibrados pueden conducir a conflictos entre estudiantes, falta de colaboración y un menor rendimiento académico (Johnson & Johnson, Cooperation and the use of technology, 1996). En el entorno empresarial, la formación de equipos ineficientes puede llevar a una disminución en la productividad, un clima laboral negativo y una pérdida de reputación para la empresa (Katzenbach & Smith, 2003). Generalizando, los equipos disfuncionales pueden enfrentar una serie de problemas que van desde la falta de comunicación y conflictos internos hasta la baja productividad y un clima laboral negativo. Contrariamente, cuando la formación de equipos es la adecuada, obtenemos resultados totalmente opuestos, es decir, mejora en el rendimiento y la satisfacción laboral y un mayor nivel de comunicación.

La literatura previa valora los aspectos que se deben considerar para la formación de un equipo. Algunos de ellos hacen referencia al equipo (normas, requisitos para formar parte, tamaño, formas de comunicación, tarea a realizar) y otras a los posibles miembros (diversidad y relaciones entre miembros, personalidad, experiencia). La forma de combinar y priorizar estos factores determinará el equipo final, por tanto, hay mucha diversidad a la hora de resolver el problema.

Como hemos comentado, una de las principales dificultades radica en la complejidad de medir los atributos relevantes de los posibles miembros del equipo. Estos atributos pueden incluir habilidades técnicas, competencias interpersonales, experiencia previa y capacidad de liderazgo, entre otros. La dificultad para evaluar adecuadamente estos atributos puede llevar a la formación de equipos disfuncionales.

Los atributos que hacen referencia a los posibles miembros, generalmente, se clasifican en tres grandes grupos. En primer lugar, las aptitudes técnicas, que se pueden representar con números precisos, números difusos o variables probabilísticas. En segundo lugar, las aptitudes para el trabajo en equipo. Estas son un conjunto de atributos que abarcan la coordinación, la adaptación, la comunicación, etc. Por último, los rasgos personales, que se miden habitualmente a partir de pruebas de personalidad. (Juárez, 2021)

Además, la importancia del trabajo en equipo ha experimentado un crecimiento en el entorno laboral moderno. Los equipos son esenciales en el mundo actual debido a su capacidad para fomentar la innovación, mejorar la eficiencia, desarrollar habilidades interpersonales, adaptarse al cambio y fomentar la diversidad e inclusión. La capacidad de colaborar efectivamente en equipos multidisciplinarios y diversos se ha convertido en una habilidad esencial para el éxito en muchas industrias. De este modo, el problema de la formación de equipos (TFP) se vuelve aún más relevante. El problema de la formación de equipos es un desafío que implica la selección y composición de grupos de individuos a partir de un conjunto de candidatos con ciertos atributos con el propósito de llevar a cabo una tarea o alcanzar un objetivo específico (Belbin R. , 2010).

Por ello, una variedad de enfoques y técnicas se han desarrollado para abordar este desafío, cada uno con sus propias características y aplicaciones. Las diferentes perspectivas desde las cuales se ha abordado este problema son la psicología, la gestión, la sociología, la educación y la matemática/informática (Robbins, Judge, & Campbell, 2017). Cada una de estas perspectivas contribuye con su visión. Este trabajo se va a enfocar desde este último punto de vista. En las últimas tres décadas, desde el ámbito de la informática y las matemáticas se ha intentado apoyar la formación de equipos desde la perspectiva de herramientas y/o sistemas que apoyan en las decisiones de la formación de equipos.

2.2. Los sistemas de apoyo a la formación de equipos

Un sistema de apoyo a la formación de equipos es una herramienta o conjunto de herramientas diseñadas para facilitar y optimizar el proceso de formación de equipos. Englobar el trabajo que vamos a realizar en el sistema correspondiente permitirá clasificarlo y relacionarlo con los estudios previos.

Podemos diferenciar 4 tipo de sistemas según la taxonomía de las dos dimensiones (Gómez-Zará, Dechurch, & Contractor, 2020). Esta proporciona una estructura para clasificar los sistemas de apoyo a la formación de equipos en función de dos aspectos

principales: la agencia y la participación de los usuarios. Al hacerlo, se busca entender mejor cómo estos sistemas operan y cómo afectan el proceso de formación de equipos.

La agencia de los usuarios hace referencia a la medida en qué los sistemas permiten a los usuarios ejercer su agencia en el proceso de la formación de equipos. Es decir, el control que tienen los usuarios sobre el proceso. Los usuarios pueden ejercer ciertos niveles de agencia y aun así ser apoyados por sistemas. Por ejemplo, un algoritmo que forma equipos basándose en las preferencias de los alumnos.

Por otra parte, la participación de los usuarios hace referencia a la cantidad de usuarios que el sistema permite participar en el proceso. Por un lado, la baja participación da lugar a equipos más heterogéneos, diversos y con más experiencia. Los resultados de estos sistemas son más predecibles, ya que dependen de unos pocos usuarios. Por otro lado, investigaciones anteriores muestran que los usuarios son más propensos a elegir trabajar con personas que ya conocen, han trabajado juntos en el pasado y/o son amigos y, por tanto, la alta participación da lugar a equipos más homogéneos.

Combinando estas dos dimensiones obtenemos cuatro tipos de sistemas de formación de equipos. Podemos visualizar la clasificación de estos sistemas según las dos dimensiones en la Figura 1. Clasificación de los sistemas según la taxonomía de las dos dimensionesFigura 1, obtenida de (Gómez-Zará, Dechurch, & Contractor, 2020).

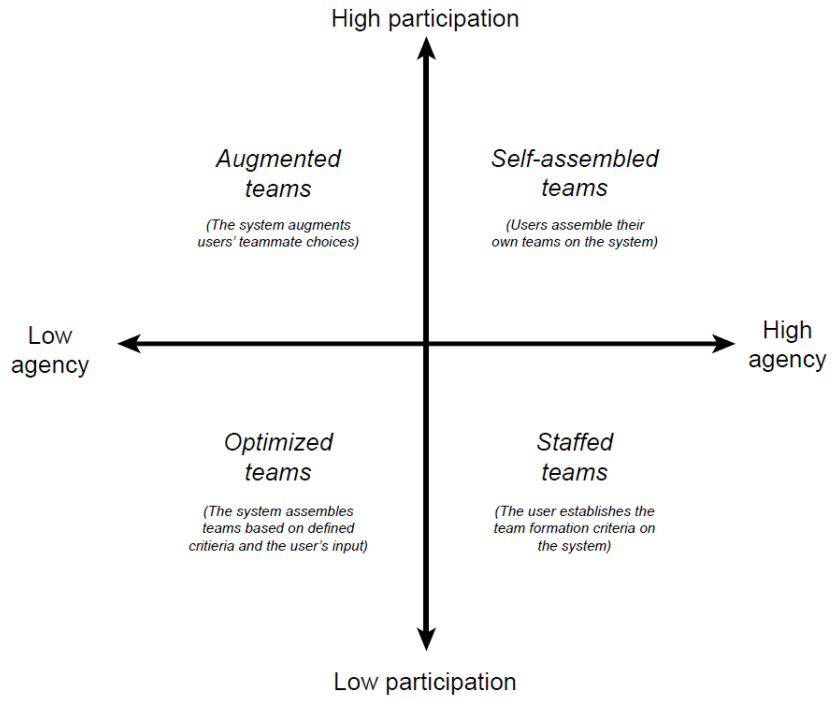

Figura 1. Clasificación de los sistemas según la taxonomía de las dos dimensiones

A continuación, vamos a describir las 4 categorías.

Equipos con personal (Staffed teams)

Un instructor que utiliza un sistema para formar equipos de estudiantes es un ejemplo. En este tipo de sistemas la agencia de los usuarios es alta y la participación es baja. Estos equipos pueden ser formados por una persona que no formará parte del equipo o por alguien que está buscando compañeros. Los sistemas se usan para apoyar a las decisiones de los usuarios, simular combinaciones de equipos o llegar a más miembros. En última

instancia, el usuario toma las decisiones finales en función de la salida de los sistemas. Los sistemas pueden proporcionar un equipo único como solución o varias recomendaciones del equipo al usuario, que tiene control sobre la entrada y los criterios de formación de equipos.

Las ventajas son que los criterios se pueden ajustar, se pueden obtener varias combinaciones de equipos y es más fácil obtener grupos heterogéneos, pero, en contraste, hay una falta de transparencia y solo un usuario participará.

Equipos autoasignados (Self-assembled teams)

Los jugadores que forman sus propios equipos en un juego virtual son un ejemplo. En este tipo de equipos la agencia y la participación de los usuarios es alta. Estos equipos habitualmente surgen de manera espontánea de los patrones de relaciones entre usuarios, tareas y sistemas. Para equipos más planificados, el sistema permite a los usuarios buscar, invitar y elegir a sus compañeros a medida que interactúan.

En los equipos autoasignados es más probable que los miembros del equipo se comprometan con las tareas ya que pueden elegir a sus compañeros y, además, las decisiones de los usuarios son transparentes. Sin embargo, se puede producir discriminación y probablemente los equipos serán homogéneos.

Equipos aumentados (Aumented teams)

Un sistema que ayuda a los usuarios a encontrar los compañeros de equipo más adecuados es un ejemplo. En este tipo de equipos la agencia de los usuarios es baja y la participación es alta. El sistema reduce las opciones de los miembros que pueden formar parte del equipo: muestra los candidatos potenciales y oculta los menos factibles.

Por un lado, es probable que se formen equipos heterogéneos y que los usuarios puedan participar en el proceso de formación. Pero, por otro lado, algunos usuarios pueden terminar sin grupo.

Equipos optimizados (Optimized teams)

Los equipos formados por los algoritmos de los sistemas son ejemplos. Este trabajo se enmarcará en este tipo de sistemas, puesto que nuestra propuesta son modelos matemáticos para la formación de equipos en el aula de acuerdo con uno o varios criterios.

En este tipo de equipos la agencia y la participación de los usuarios en el proceso de formación están limitados. Generalmente, un solo usuario, que no tiene control sobre el proceso de formación, proporciona los datos requeridos por el sistema. En nuestro caso, los datos serán los tamaños de equipo deseados, los alumnos que participarán y las dependencias entre alumnos. Como resultado, los miembros del equipo no pueden elegir la composición de su equipo ni establecer los criterios de montaje. El equipo final dependerá de la entrada que proporciona el usuario, de los criterios y del algoritmo usado para obtener la distribución de alumnos en los equipos.

Por un lado, las ventajas son que el equipo se forma de manera rápida, objetiva y reproducible, se pueden manejar grupos con muchos miembros y los criterios de montaje se pueden ajustar. Por otro lado, las desventajas son que las soluciones obtenidas son fijas y que los miembros del equipo están excluidos del proceso de formación.

2.3. El problema de la formación de equipos (TFP) optimizada

El Problema de Formación de Equipos consiste en la búsqueda de los miembros de un equipo, a partir de un conjunto de candidatos con ciertos atributos, con el objetivo de formar un equipo o varios equipos de máxima "eficacia". En este problema, se busca encontrar la combinación óptima de miembros del equipo que conduzca a resultados óptimos o cercanos a lo óptimo en términos de la tarea o meta específica que se pretende lograr. Estos equipos llevarán a cabo una tarea específica (Juárez, 2021). En este estudio nos centraremos en este tipo de problemas.

Para la resolución de este problema se necesita un algoritmo de formación de equipos, que generalmente incluye el algoritmo de optimización y la heurística de formación de equipos. Estos algoritmos de formación de equipos se engloban dentro del sistema de formación de equipos optimizados (*optimized teams*).

El algoritmo de optimización es la parte que se encarga de encontrar la mejor combinación de estudiantes para formar equipos, de acuerdo con ciertos criterios o restricciones establecidos. Puede ser un algoritmo determinista o estocástico, y su objetivo es maximizar o minimizar una función objetivo que representa el rendimiento deseado de los equipos. Aunque este tipo de problemas se modelice como problemas de programación entera, por su elevado coste computacional, se suelen usar otros tipos de algoritmos para reducir el espacio de búsqueda y disminuir el tiempo de ejecución a expensas de perder optimalidad (Juárez, 2021). Algunos de los algoritmos utilizados en la literatura son heurísticos, algoritmos de aproximación y metaheurísticos (algoritmos genéticos, algoritmo de recocido simulado, optimización por enjambre de partículas, algoritmos culturales, optimización competitiva imperialista u optimización por fuga de cerebros). (Juárez, 2021) El problema de la formación de equipos es un problema de optimización y, por tanto, se han planteado varios métodos de programación lineal.

Por otra parte, la heurística de evaluación de equipos define los criterios o reglas que se utilizan para evaluar la idoneidad de una combinación particular de estudiantes en un equipo. Estas reglas pueden estar basadas en diversas consideraciones, como las habilidades individuales de los estudiantes, su experiencia previa, sus preferencias de trabajo, la diversidad del equipo, entre otros factores relevantes para el rendimiento del equipo. Algunos de los criterios utilizados para la heurística de formación de equipos en la literatura son las aptitudes técnicas, la comunicación, la coordinación y los rasgos personales (Juárez, 2021).

Recordemos que la heurística de evaluación es una aproximación del desempeño real del equipo y no el desempeño real del mismo. Por tanto, para poder determinar la heurística de evaluación del equipo es más apropiada para formar equipos, es necesario realizar experimentos reales o pruebas A/B comparando el desempeño real de equipos formados con distintas heurísticas de evaluación.

Según la taxonomía propuesta en el artículo (Juárez, 2021), las técnicas de la formación de equipos optimizada pueden clasificarse en dos grandes grupos en función de cómo se modele el problema. Por un lado, se encuentran los modelos basados en la asignación y, por otro lado, los modelos basados en la comunidad.

Modelos basados en la asignación

Este tipo de modelo implica un emparejamiento bipartito entre un conjunto de candidatos y un conjunto de equipos o posiciones de equipo. Cada emparejamiento potencial tiene

una puntuación que representa la adecuación de un candidato con respecto a un equipo o a una posición de equipo. De esta forma, el objetivo general de los TFP basados en la asignación es encontrar la combinación que maximice la puntuación. Dentro de este tipo de modelos, diferenciamos tres casos:

- Formación de un único equipo
- Formación de varios equipos
- Formación de equipos afines

En los tres casos podemos dividir los modelos en dos subcategorías dependiendo de si la posición dentro del equipo es importante o no. Sin embargo, la gran diferencia entre los dos primeros casos y el último es que en la formación de uno o varios equipos solo se tiene en cuenta las aptitudes técnicas de los candidatos como criterio para formar equipos y, en cambio, en los equipos afines se reconoce además el efecto social entre los individuos.

Modelos basados en la comunidad

Estos modelos consisten en encontrar un subconjunto de candidatos con interacciones estrechamente ligadas cuyos conjuntos de habilidades combinados cumplan los requisitos de la tarea. Este tipo de modelo se basa en un problema de grafos en el que cada nodo es cada uno de los individuos, que tendrá unos valores para un conjunto de actividades. Estas habilidades se pueden clasificar en tres grandes grupos:

- Habilidades binarias: El nivel de competencia de los candidatos se mide con un número binario.
- Habilidades ponderadas: El nivel de competencia de los candidatos se mide con un número entero o real.
- Habilidades probabilísticas: Los niveles de habilidad de los candidatos son probabilísticos.

Además, los nodos están conectados por unas aristas que pueden representar algún tipo de relación entre candidatos (coste de comunicación, nivel de colaboración, etc). Dada una tarea que requiere cierto nivel de habilidades, el objetivo será maximizar este nivel de habilidades (o simplemente que se cumpla) a la vez que minimizamos el coste global de comunicación, maximizamos el nivel global de colaboración u optimizamos alguna otra función equivalente que dependa de los valores de las aristas.

Nuestro trabajo se engloba dentro de los "Modelos basados en la asignación", específicamente en el caso de "Formación de varios equipos".

2.4. El problema de la formación de equipos en el aula optimizada

El problema de la formación de equipos se puede aplicar en muchos contextos diferentes. El contexto específico del problema, además, influirá en gran medida en la manera de formular el problema (Juárez, 2021).

El problema de la formación de equipos en el aula consiste en agrupar a los estudiantes de un aula en equipos o grupos de trabajo con el propósito de llevar a cabo actividades educativas, proyectos o tareas de aprendizaje en un entorno escolar. Las diferencias a nivel técnico con el TFP general es que en el aula se busca particionar el conjunto de alumnos en equipos disjuntos y no únicamente formar un equipo como en otros contextos.

Además, en el contexto del aula pueden aparecer restricciones y/o consideraciones adicionales como tamaños de equipo mínimos y máximos o restricciones impuestas por los profesores.

A nivel más general, la diferencia entre la formación de equipos en el aula y la formación de equipos en general radica principalmente en el contexto en el que se lleva a cabo. Mientras que la formación de equipos puede referirse al proceso de agrupar a personas en equipos en diversos ámbitos como el trabajo, el deporte, la investigación, entre otros, la formación de equipos en el aula se centra específicamente en el entorno educativo y en las dinámicas de aprendizaje entre los estudiantes.

Además, la formación de equipos en el aula suele estar dirigida por el docente, quien puede tomar decisiones sobre la composición de los equipos con el objetivo de fomentar la diversidad, la colaboración y el desarrollo de habilidades específicas. Los objetivos de la formación de equipos en el aula también pueden estar relacionados con el logro de metas académicas y la promoción del aprendizaje activo y significativo.

Este enfoque pedagógico promueve la colaboración, el trabajo en equipo y el aprendizaje cooperativo entre los estudiantes, lo que puede mejorar su participación, motivación, habilidades sociales y rendimiento académico. (Johnson, Johnson, & Smith, Cooperative Learning: Improving University Instruction by Basing Practice on Validated Theory, 2014)

La incorporación del trabajo colaborativo en todos los niveles del campo educativo crece día tras día. Uno de los problemas más recurrentes a los que se enfrentan los profesores que quieren emplear esta estrategia de aprendizaje es la buena formación de equipos de los estudiantes, ya que esta tarea puede ser compleja tanto conceptual como computacionalmente, especialmente si el objetivo es automatizarla (Revelo Sánchez, Collazos Ordóñez, Redondo Duque, & Santana Pinto, 2021). Además, numerosos estudios destacan los beneficios del Aprendizaje Basado en Equipos frente al Aprendizaje individual. La formación de equipos de trabajo es un aspecto clave para el éxito de las actividades grupales (Ugarte, Aranzabal, Arruarte, & Larrañaga, 2022, October).

Las dificultades y desafíos que deben enfrentar los equipos de estudiantes son muchos, especialmente si tienen poca experiencia en el trabajo en equipo y si no se proporciona orientación o apoyo. Algunos de los principales factores que afectan a un correcto desarrollo del trabajo en equipo son: composición grupal, diferente motivación, expectativas o compromiso dentro de los compañeros de equipo, choques de personalidad, miembros dominantes y pasivos, poca o ausencia de orientación, ambigüedad de tareas, ambigüedad de roles, disparidad académica, resistencia al trabajo en equipo, falta de habilidades interpersonales, falta de normas grupales, etc. Por lo tanto, la capacitación de los estudiantes en habilidades de trabajo en equipo es crucial (Aranzabal, Epelde, & Artetxe, 2022).

Sin embargo, la heurística de formación de equipos no es fácil de determinar debido a la variedad de factores que pueden influir en la efectividad de un equipo. Por un lado, los equipos se pueden evaluar en función de múltiples criterios, determinar cómo ponderar y combinar estos criterios de manera efectiva puede ser complicado. Además, aunque nos vayamos a limitar a la formación de equipos dentro del aula, la composición ideal de un equipo puede variar según el contexto y los objetivos específicos de la tarea. Por lo tanto, la heurística debe ser lo suficientemente flexible como para adaptarse a diferentes situaciones y necesidades. Por último, la heurística debe ser capaz de evaluar rápidamente

la idoneidad de diferentes combinaciones de equipos, especialmente en entornos con un gran número de estudiantes y equipos posibles.

En la taxonomía propuesta en (Maqtary, Abdulqader, & Kamal, 2019) se considera que, durante la formación de grupos, se deben tener en cuenta varios atributos para asegurar que los grupos alcancen sus metas. Así, los atributos pueden ser categorizados en dos clases: atributos de miembro y atributos de grupo. Introducir esta taxonomía nos proporciona una forma sistemática de organizar y clasificar los atributos que influyen en la composición y el desempeño de los equipos.

Atributos de los miembros

Estos son los que describen a la persona que se incluirá en el grupo. Todos estos atributos están relacionados con cómo se configura la heurística de evaluación de equipos. Hay cinco tipos de atributos de los miembros que se han usado comúnmente para formar equipos. Estos son el conocimiento, el estilo de aprendizaje, los rasgos de personalidad, la interacción social y el rol en el equipo. En nuestro trabajo usaremos el rol en el equipo como atributo de los miembros.

Atributos de grupo

Estos están relacionados con el objetivo y la tarea del grupo. Basándonos en el método de asignación, la formación de grupos se puede lograr de forma manual o automática. Asimismo, la formación manual puede ser de autoselección o de asignación del instructor y la automática puede ser aleatoria o basada en ciertos criterios. Si consideramos el tipo de tarea a realizar, un grupo puede necesitar características homogéneas o heterogéneas. La duración que se necesita para completar una tarea nos divide los grupos en grupos de corto y largo plazo. Por último, el proceso de asignación de miembros puede ser estático mientras dura la tarea o puede ser dinámico.

Dado que en este trabajo vamos a centrarnos en el proceso de formar los grupos y no en cómo se desarrollen los grupos en las tareas a realizar, podemos aplicar los atributos basados en el método de asignación. En este caso, la formación de grupos se logrará de manera automática y basada en ciertos criterios. Además, formaremos grupos heterogéneos.

La revisión de la literatura en (Maqtary, Abdulqader, & Kamal, 2019) muestra que el atributo más utilizado es el conocimiento. Además, el estilo de aprendizaje y los rasgos de personalidad se usaron de forma similar. Por un lado, el estilo de aprendizaje juega un papel vitar en la educación y, por otro lado, los rasgos de personalidad son los atributos recopilados más simples para experimentos y estudios.

Una de las pruebas de personalidad más conocidas es el Myers-Briggs Type Indicator (MBTI). Otros inventarios basados en la personalidad son Kirton's Adaption-Innovation Inventory (KAI), 16 Personality Factor Questionnaire (16PF), Occupational Personality Questionnaire (OPQ) y Big Five. Otras herramientas de formación de equipos ampliamente aceptadas se basan en la taxonomía de roles, entre ellos, la teoría del rol de Belbin, aceptada en todo el mundo (Aranzabal, Epelde, & Artetxe, 2022).

El enfoque reciente ha añadido nuevos atributos. En (Ugarte, Aranzabal, Arruarte, & Larrañaga, 2022, October) se añaden a los anteriores como criterios relevantes las relaciones sociales y la tendencia conductual en un ambiente de equipo. Aunque hay otros criterios que, con menos frecuencia, también se utilizan.

La literatura también muestra que hay más investigaciones sobre los grupos heterogéneos, es decir, equipos formados por miembros con características complementarias, que, sobre grupos homogéneos, que son los formados por miembros con características similares. En estos últimos, la similitud entre miembros puede llevar a un menor número de conflictos y una mejor comprensión, pero también puede causar dificultades para desempeñar el trabajo, ya que ciertas características no están reflejadas, lo cual no ocurre en los grupos heterogéneos. (Ugarte, Aranzabal, Arruarte, & Larrañaga, 2022, October). Algunos de los criterios heterogéneos usados en la literatura son la diversidad de personalidades, de roles y de género. En este trabajo nos centraremos en la formación de grupos heterogéneos ya que son los más adecuados en contextos de aprendizaje.

Siguiendo la línea de la heterogeneidad, un aspecto clave es el equilibrio entre géneros ya que, algunos estudios sugieren que esto se traduce en una mayor colaboración de los miembros, con contribuciones más equitativas. (Ugarte, Aranzabal, Arruarte, & Larrañaga, 2022, October). Sin embargo, la agrupación homogénea presenta ciertas ventajas para algunos tipos de actividades de aprendizaje, especialmente aquellas que involucran descubrimiento guiado, desarrollo de habilidades, revisión de material que ya ha sido aprendido, o en tareas altamente estructuradas de construcción de competencias, permitiendo a los estudiantes progresar a un ritmo similar, lo que es beneficioso para el logro de metas específicas. (Revelo Sánchez, Collazos Ordóñez, Redondo Duque, & Santana Pinto, 2021)

Un factor que también ha resultado importante a la hora de crear un grupo es el tamaño de este. Este varía según los objetivos o el tipo de trabajo a realizar. Los equipos que son demasiado pequeños pueden no tener recursos suficientes para lograr el trabajo perseguido, y los equipos que son demasiado grandes pueden tener problemas de comunicación o ser menos productivos (Ugarte, Aranzabal, Arruarte, & Larrañaga, 2022, October).

Por último, aunque se pueden encontrar diferentes experiencias sobre la formación de equipos en la literatura, no hay evidencia clara para apoyar el mejor método para formar equipos efectivos, en términos de mejorar el rendimiento general del equipo y el logro de los resultados del proyecto (Aranzabal, Epelde, & Artetxe, 2022). Veamos las propuestas específicas de diferentes artículos a este problema.

En el artículo (Revelo Sánchez, Collazos Ordóñez, Redondo Duque, & Santana Pinto, 2021) se presenta una técnica homogénea de formación grupal basada en los rasgos de personalidad con un cuestionario basado en el "Big Five Inventory (BFI)" que miden las dimensiones propuestas en el mismo (Extraversión, Amabilidad, Escrupulosidad, neuroticismo y apertura). Además, se plantea el problema como un problema de optimización que se resuelve mediante la búsqueda heurística ofrecida por algoritmos evolutivos. Se utiliza un algoritmo genético modificado tipo GGA. Las restricciones que se consideran son que el tamaño de los grupos ha de ser al menos dos; no puede exceder la mitad del número total de estudiantes si es par; no puede exceder la mitad más uno del número total de estudiantes si es impar; si el tamaño de los grupos no es un divisor del número total de estudiantes, entonces el tamaño de los grupos no puede diferir en más de 1 unidad. Este método se validó con cuatro grupos diferentes pertenecientes a la Facultad de Ingeniería de la Universidad de Nariño Campus Pasto. Como resultados se obtuvo una mejora significante en el funcionamiento y rendimiento de los grupos formados con este método respecto a los formados por los mismos estudiantes.

En el artículo (Ma, et al., 2022), se propone un nuevo enfoque de optimización colaborativa para la formación de equipos de aprendizaje con líder basado en un modelo de aprendizaje refinado y el modelo de entornos: clases, agentes, roles, grupos y objetos (E-CARGO). Además, se resuelve en base al paquete de optimización IBM CPLEX. Los equipos con líder garantizan de manera más efectiva la mejora de la eficiencia y la calidad del aprendizaje de todos los miembros, alentando a los líderes a fortalecer la organización y gestión de actividades de aprendizaje entre los miembros. En este enfoque, se modela a los alumnos combinando características 5D y tres tipos de restricciones. Las características son la capacidad cognitiva, el liderazgo, la sociabilidad, el estilo de aprendizaje y la personalidad (Esta última se basa también en la teoría de los Big Five). Para mejorar la eficiencia del aprendizaje colaborativo, los miembros del equipo de aprendizaje deben cumplir con muchas limitaciones que garantizan que los miembros puedan llevar a cabo sus actividades de aprendizaje de manera eficiente y armoniosa. Estas restricciones consisten en agrupar conflictos y en equilibrar el género y el número de miembros en los equipos. Además, los líderes se seleccionan en función del desempeño en seis características. La importancia de cada característica es diferente en varios escenarios, por tanto, se ajustan los pesos de estas características mediante el proceso de jerarquía analítica difusa (FAHP). El enfoque se ha validado con 51 estudiantes de la Universidad Normal de Hunan. Se han comparado los resultados del algoritmo escogido frente a otros 4, y se ha concluido que el rendimiento del método propuesto es excelente y puede calcular mejores soluciones que los algoritmos de comparación.

En (Andrejczuk, Bistaffa, Blum, & Rodríguez-Aguilar, 2019) se presenta un problema de composición de equipos, llamado problema de composición de equipos sinérgicos (STCP) que incorpora las competencias, la personalidad y el género a la hora de organizar los equipos. La personalidad se mide utilizando una variante reducida del Indicador de Tipo de Myers-Briggs (MBTI). Se buscan equipos diversos en personalidad y género y que cubran todas las competencias. Además, se requiere que todos los equipos estén equilibrados. Para resolver el problema se usan dos métodos, por una parte, la programación lineal exacta entera (STCPSolver) y, por otra parte, una heurística adaptable que generara soluciones de alta calidad en un tiempo de computación limitada (SynTeam). Los dos algoritmos se evaluaron con datos reales y se compararon entre ellos. No se observaron diferencias significativas en el rendimiento (relativo) de los algoritmos al comparar los distintos tipos de tareas. Esto es un claro indicio de la solidez de las técnicas desarrolladas. Esto demuestra que la heurística funciona bien y a un menor coste computacional.

En (Ugarte, Aranzabal, Arruarte, & Larrañaga, 2022, October) se implementa la plataforma TalSor, una herramienta que utiliza el Inventario de Autopercepción de Roles de Equipo Belbin para determinar la tendencia de comportamiento de los estudiantes y crear equipos heterogéneos usando un algoritmo genético. Aquí, no se consideran pruebas de personalidad, porque a veces los ítems que aparecen se refieren a situaciones que los estudiantes no han tenido la oportunidad de experimentar. Además, esta herramienta pretende buscar equipos equilibrados también en términos de género y permite determinar el número de miembros del equipo o el número de equipos que se generarán. Además del equilibrio de habilidades dentro de cada equipo, es decir, que todos los roles de Belbin estén cubiertos, también tiene en cuenta que todos los equipos deben tener capacidades similares. Esta herramienta se ha validado con alumnos de la Universidad del País Vasco UPV/EHU. Los resultados obtenidos muestran que los equipos de estudiantes creados tanto utilizando la tendencia conductual en un ambiente de equipo de los

estudiantes como único criterio y la combinación de este criterio junto con el género proporcionan resultados satisfactorios.

En (Aranzabal, Epelde, & Artetxe, 2022) se presenta un método para formar equipos equilibrados basados en los roles de Belbin, con el objetivo de impulsar la interdependencia positiva y la responsabilidad individual dentro de los equipos y mejorar su rendimiento en un entorno de aprendizaje basado en proyectos. El rendimiento de los estudiantes se ha medido a través de los puntajes obtenidos durante el proyecto, el examen individual y el Factor de Responsabilidad Individual (IAF). Se ha comparado la eficiencia de los equipos formados basándose en los roles de Belbin con equipos autoseleccionados por los alumnos en el Grado en Ingeniería Química de la Universidad del País Vasco (UPV/EHU), España, obteniendo resultados satisfactorios.

En (Chen & Lin, 2004) se trabaja con equipos destinados a tareas de ingeniería. Se utiliza el conocimiento multifuncional, la capacidad de trabajo y la colegialidad como características de los miembros de un equipo. Las dos primeras se capturan utilizando el proceso de jerarquía analítica (AHP). Por otro lado, se usa el perfil de personalidad utilizando el indicador tipo Myers-Briggs (MBTI) como base para evaluar las habilidades de cada miembro del equipo para trabajar con otros. Se resuelve el problema de optimización con un método exacto aplicado a un ejemplo.

En (Alberola, del Val, Sanchez-Anguix, Palomares, & Teruel, 2016) se presenta una herramienta computacional que intenta abordar los problemas que surgen al aplicar la taxonomía de roles de Belbin al aula. Esta herramienta hace uso del aprendizaje bayesiano para abordar la incertidumbre con respecto a los roles prominentes de los estudiantes, y el problema de encontrar equipos óptimos se trata como un problema de generación de estructuras de coalición, que se resuelve mediante métodos de programación lineal. Además, la herramienta propuesta es de naturaleza iterativa, de forma que en cada iteración tiene en cuenta la retroalimentación proporcionada por los compañeros después de trabajar en un equipo. Esta herramienta fue validada en un entorno experimental con estudiantes reales de Grado en Turismo de la Universitat Politècnica de València durante el curso 2013/2014. Los resultados muestran que los estudiantes percibieron un mayor grado de cooperación y coordinación en los equipos propuestos por la herramienta. Además, percibieron las actitudes de sus compañeros de equipo como positivas y tuvieron un mayor grado de satisfacción.

En (Sanchez-Anguix, Alberola, Del Val, Palomares, & Teruel, 2023) se presenta una herramienta de formación de equipos que implementa dos heurísticas de evaluación basadas en la taxonomía de roles de Belbin y el Indicador de Tipo Myer-Briggs con el objetivo de poder comparar las dos heurísticas. La herramienta de formación de equipos es una herramienta inteligente y extensible cuya optimización se basa en un modelo lineal entero que se puede ampliar para admitir diferentes heurísticas de evaluación de equipos. Esta herramienta se aplicó en un Programa de Licenciatura en Turismo y permitió comparar las dos heurísticas de evaluación. Se encontraron diferencias significativas tanto en la distribución de las personalidades como de los roles entre los géneros. También se concluyó que los equipos formados con la heurística basada en el MBTI quedaron ligeramente más satisfechos que los que se formaron con la heurística basada en la taxonomía de los roles de Belbin.

En (Flores-Parra, Castañón-Puga, Evans, Rosales-Cisneros, & Gaxiola-Pacheco, 2018, June) se implementa un método de agrupación usando el Análisis de redes Sociales (SNA)

basándose también en los roles de Belbin. Se usó el inventario de autopercepción de Belbin y una prueba de preferencia. El SNA es un método para estudiar las relaciones entre individuos o grupos de individuos, mientras se estudia simultáneamente el contexto social. El valor de SNA, como enfoque de investigación, es la capacidad de examinar a los individuos que están incrustados en una estructura social y cómo las estructuras sociales emergen de las relaciones entre los individuos. Una vez creado el sociograma en la que los nodos son los individuos y las aristas muestran preferencias entre individuos, la información se procesa utilizando algoritmos de análisis de red para identificar grupos potenciales y obtener valores métricos. Esta red se analiza con algoritmos de comunidades y tríadas. Por último, se identifican los grupos prospectivos y los valores de los parámetros. Finalmente, el profesor tiene en cuenta estas tríadas y comunidades para tomar la decisión final de qué equipos formar. Se realizó un estudio de caso en un grupo de estudiantes de la Universidad Autónoma de Baja California, México.

En (Yannibelli & Amandi, 2012) se propone un algoritmo evolutivo determinista de aglomeración para formar equipos bien equilibrados. El algoritmo diseña diferentes alternativas para dividir a los estudiantes en equipos y evalúa cada alternativa con respecto al criterio de agrupación mencionado anteriormente. Esta evaluación se lleva a cabo sobre la base del conocimiento de los roles de los estudiantes siguiendo los roles establecidos en el modelo de Belbin. Además de los roles, Belbin también tiene en cuenta los niveles de preferencia entre miembros. Como restricciones en la formación de grupos, se considera que, si el número de alumnos es un múltiplo del número de grupos, todos los grupos tendrán el mismo tamaño; si no, la diferencia de miembros entre dos grupos no puede exceder 1. Este método se validó sobre 10 conjuntos de datos diseñados con diferentes niveles de complejidad y se comprobó su capacidad de lograr soluciones de alta calidad en un tiempo de cálculo aceptable para cada uno de los conjuntos de datos empleados.

En la Tabla 1. Características de los estudios revisadosTabla 1 vemos un resumen de las características de los estudios revisados. Prácticamente en todos los artículos revisados se considera la personalidad como un criterio para formar equipos, en algunos de ellos se consideran adicionalmente otros atributos. Sin embargo, no hay ningún consenso sobre qué atributos se deben de tener en cuenta a la hora de formar grupos.

La elección de los atributos se evalúa posteriormente cuando se aplican los distintos métodos de resolución de problemas, es decir, si los equipos formados tienen un buen desempeño y eficacia. Por tanto, es importante hacer una buena elección de los atributos. Hemos comentado con anterioridad, que la única forma de poder realizar esta evaluación posterior sin añadir ruido a los resultados es usando métodos exactos.

Hemos visto también que la mayoría de los artículos trabajan con grupos heterogéneos ya que la similitud entre miembros puede llevar a un menor número de conflictos y una mejor comprensión.

En cuanto a los algoritmos utilizados en los artículos revisados, algunos de ellos utilizan métodos aproximados como son los algoritmos genéticos y algunas de sus versiones. El problema de este tipo de algoritmos, como ya hemos comentado, es que, al ser aproximados, no garantizan una solución óptima. Esto mete ruido en los estudios que comparan heurísticas de evaluación de la formación de grupos ya que el buen desempeño no dependerá únicamente de esta sino también de la calidad de la solución que ha encontrado el algoritmo. Además, en el contexto del aula, al poder formar pocos equipos,

debemos de tener la mínima variabilidad posible para que los resultados sean más fiables. De esta forma, podremos realizar pruebas A/B y poder comparar los desempeños de las dos particiones de estudio de forma segura. Por tanto, este tipo de algoritmos no nos sirve para comparar heurísticas de evaluación de grupos, la única forma de garantizar que esta comparación esté equilibrada es utilizando métodos exactos como ya hemos comentado.

Hemos visto también la importancia de determinar una buena heurística de evaluación de equipo ya que la mala elección de los atributos puede llevar a equipos disfuncionales. Los equipos disfuncionales pueden generar una serie de problemas y consecuencias negativas. En el ámbito del aprendizaje los equipos mal equilibrados pueden conducir a conflictos entre estudiantes, falta de colaboración y un menor rendimiento académico. Pese a que hay un consenso de la importancia que esto tiene, no hay una heurística de evaluación que se haya demostrado mejor que las demás.

Como conclusión de la revisión de la literatura, vemos que se han propuesto muchas metaheurísticas pero pocos modelos exactos, siendo estos últimos los más adecuados para hacer pruebas de heurísticas de evaluación dentro del aula. El objetivo de nuestro trabajo será cubrir esta brecha en el problema de la formación de equipos en el aula, aunque tendremos como limitación el elevado coste computacional del problema. Buscaremos un modelo de programación lineal exacta que pueda servir para la evaluación de estas heurísticas. Además, vemos también que en la mayoría de los trabajos se aboga por equipos heterogéneos. Los modelos que vamos a proponer en este trabajo intentarán dar soporte a este tipo de situaciones.

Estudio	Técnica de optimización	Criterio	Tamaño	Consideraciones	Objetivo
(Revelo Sánchez, Collazos Ordóñez, Redondo Duque, & Santana Pinto, 2021)	Algoritmo genético modificado tipo GGA	Rasgos de personalidad medidos con el Big Five Inventory	El algoritmo se probó con tamaños desde 250, hasta 5000. Los resultados del experimento se validaron con 132 alumnos	Se busca la homogeneidad entre grupos respetando la heterogeneidad de todos los estudiantes	Comparar el desempeño de grupos formados con este método frente a grupos formados por los mismos estudiantes.
(Ma, et al., 2022)	Optimización exacta	Características 5D: capacidad cognitiva, liderazgo, sociabilidad, estilo de aprendizaje y personalidad	51 alumnos	Agrupar conflictos Equilibrar el género y el número de miembros en los equipos Equipos con líder	Validar la efectividad del método propuesto
(Andrejczuk, Bistaffa, Blum, & Rodríguez-Aguilar, 2019)	Programación lineal exacta entera + heurística adaptable	Variante reducida del Indicador de Tipo Myers-Briggs	Tamaños que van desde 10 hasta 100 alumnos	Equipos sinérgicos: diversos en personalidad y género que cubran todas las competencias	Comparar los dos métodos propuestos
(Ugarte, Aranzabal, Arruarte, & Larrañaga, 2022, October)	Algoritmo genético	Roles de Belbin	32 alumnos	Heterogeneidad dentro del grupo en términos de género y de habilidades. Equipos homogéneos entre ellos.	Diseñar e implementar una herramienta (TalSor) que automatiza la creación de equipos eficientes.
(Aranzabal, Epelde, & Artetxe, 2022)		Roles de Belbin	Se aplicó a una clase de 37 alumnos y otra de 22	Equipos equilibrados en habilidades y género	Comparar la eficiencia de equipos formados según los roles de Belbin con equipos autoseleccionados
(Chen & Lin, 2004)	Método exacto	Conocimiento multifuncional, capacidad de trabajo y colegialidad	30 alumnos	Equipos destinados a tareas de ingeniería	Establecer un equipo multifuncional eficiente
(Alberola, del Val, Sanchez-Anguix, Palomares, & Teruel, 2016)	Métodos de programación lineal	Roles de Belbin	77 alumnos	Los equipos se van modificando después de cada actividad al obtener los miembros nuevas puntuaciones.	Probar el rendimiento de la herramienta propuesta.
(Sanchez-Anguix, Alberola, Del Val, Palomares, & Teruel, 2023)	Modelo de programación lineal entera	Roles de Belbin e Indicador de Tipo Myer-Briggs	Total de 162 alumnos. A la mitad se les aplicó roles de Belbin y a la otra mitad el Indicador de Tipo Myer-Briggs	Equipos con variedad de habilidades/roles	Comparar dos heurísticas de evaluación basados en 2 de las teorías más populares
(Flores-Parra, Castañón-Puga, Evans, Rosales-Cisneros, & Gaxiola-Pacheco, 2018, June)	Algoritmos de análisis de redes	Roles de Belbin y preferencias entre alumnos	7 alumnos		Proporcionar una herramienta de apoyo para la formación de equipos que tiene en cuenta preferencias de alumnos
(Yannibelli & Amandi, 2012)	Algoritmo evolutivo determinista de aglomeración	Roles de Belbin	Desde tamaño 18 hasta 3000	Equipos equilibrados en habilidades y en miembros	Comprobar la calidad del algoritmo a nivel de tiempos y de soluciones encontrada

Tabla 1. Características de los estudios revisados

3. METODOLOGÍA

En esta sección explicamos detalladamente la metodología que hemos seguido en este trabajo. Podemos ver esta metodología a grandes rasgos en la Figura 2.

El primer paso que hemos seguido es revisar la literatura existente sobre el tema tratado, el problema de la formación de equipos en el aula. Para ello, consultamos artículos para identificar estudios previos, investigaciones relevantes y enfoques metodológicos utilizados en problemas similares. La revisión de literatura nos permitió comprender el estado actual del conocimiento en el área, identificar teorías y modelos relevantes, buscar brechas y establecer los objetivos del trabajo.

Estos objetivos, a grandes rasgos, son proponer, implementar, comparar y buscar limitaciones de 3 modelos de programación lineal para la resolución del problema de la formación de equipos en el aula que nos permitan comparar heurísticas de evaluación de equipo.

Una vez que ya tenemos definidos los objetivos, definimos tres formas distintas de modelizar el problema de formación de equipos en el aula. Estos tres modelos son modelos de programación lineal entera. La definición de los modelos comprende la descripción las variables de decisión, las funciones objetivo a maximizar y las restricciones a cumplir de cada uno de ellos. La función objetivo, en todo caso, dependerá de la heurística de evaluación escogida. Sin embargo, estos modelos son independientes del criterio utilizado para medir los atributos de los miembros que decidimos a continuación.

Para poder realizar los futuros experimentos necesitamos decidir qué heurística de evaluación usaremos como caso de estudio. Para ello, exploramos la teoría de los roles de Belbin y su relevancia para el problema de formación de equipos en el aula. Nos quedamos con la teoría original que considera los ocho roles definidos por Belbin frente a los nueve roles que consideran las versiones más modernas. Además, detallamos la forma de medir estos atributos mediante el cuestionario de los roles de Belbin. Por último, vemos cómo incorporar los roles de Belbin en las modelizaciones definidas, con el fin de utilizarlos como parámetros en el problema de optimización.

Con todo esto ya definido y detallado, implementamos los tres modelos definidos en un entorno computacional de Python. La implementación de los modelos consta de dos partes, el cálculo de los parámetros y el solver. Para poder calcular los parámetros y resolver el problema, el docente, deberá introducir los tamaños permitidos para los alumnos (deberán ser como máximo dos tamaños diferentes que no difieran en más de una unidad), el conjunto de alumnos que queremos particionar y las dependencias entre los alumnos, es decir, si algún conjunto de alumnos debe ir obligatoriamente en el mismo equipo o si una pareja de alumnos no puede compartir equipo.

Para poder aplicar los modelos generamos instancias de distintos tamaños con datos de un experimento real en el que se realizó el cuestionario de Belbin a un conjunto de alumnos universitarios. Ejecutamos los modelos utilizando estas instancias, de forma que cada una de las instancias se ejecuta 10 veces y calculamos la media para obtener resultados más fiables. Como resultados obtenemos el tiempo de ejecución y el valor objetivo de cada instancia para cada modelo. Registramos y documentamos cuidadosamente los resultados obtenidos en cada ejecución del modelo para su posterior análisis y evaluación.

A continuación, realizamos un análisis estadístico de los resultados obtenidos utilizando técnicas apropiadas. En un primer lugar, se considera el análisis de varianza (ANOVA). El ANOVA es una herramienta poderosa para detectar diferencias significativas entre múltiples grupos, lo que permite a los investigadores determinar la influencia de una variable categórica en una variable continua. Permite realizar comparaciones simultáneas entre múltiples grupos, evitando la necesidad de realizar múltiples pruebas de comparación de pares. Proporciona una comprensión más completa de la relación entre las variables y ayuda a identificar patrones o tendencias que pueden no ser evidentes con otras técnicas estadísticas.

Sin embargo, el ANOVA parte de una serie de supuestos, los datos del experimento deben ser independientes, normales y homocedásticos. Nuestros datos sí que cumplen el supuesto de independencia, no obstante, no cumplen los supuestos de normalidad y homocedasticidad. Por ello, recurrimos a otro tipo de prueba, la prueba de Friedman. Esta es una prueba estadística no paramétrica utilizada para analizar datos de medidas repetidas. Se utiliza principalmente cuando no se cumplen los supuestos de normalidad y homogeneidad de varianzas, lo que la convierte en una alternativa sólida al ANOVA de medidas repetidas. El análisis de la varianza comprueba en qué medida difieren los valores medidos de la muestra dependiente. La prueba de Friedman, en cambio, utiliza rangos en lugar de los valores medidos reales. La gran ventaja de utilizar rangos es que, si no te fijas en la diferencia de medias, sino en la suma de rangos, no es necesario que los datos tengan una distribución normal. (Conover, 1999).

Dividimos nuestro conjunto de datos según el tamaño de la instancia resuelta y realizamos la prueba de Friedman a cada uno de estos subconjuntos de datos. Esto es porque no tenemos datos para todas las parejas de tamaño de instancia y modelo y, por tanto, no es un conjunto de datos equilibrado. Además, la prueba de Friedman es sustituto del ANOVA de una vía y, por tanto, no podemos considerar el factor tamaño.

Antes de realizar esta prueba, realizamos un análisis descriptivo de los datos y comprobamos que no hay datos anómalos que puedan estar invalidando nuestro análisis. A continuación, interpretamos los resultados obtenidos utilizando gráficos, tablas y otros recursos visuales para facilitar la comprensión y el análisis de los datos.

Por último, resumimos los principales hallazgos del estudio y discutimos las implicaciones prácticas de los resultados obtenidos como conclusiones. Además, reflexionamos sobre las limitaciones del estudio y sugerimos posibles áreas de investigación futura.

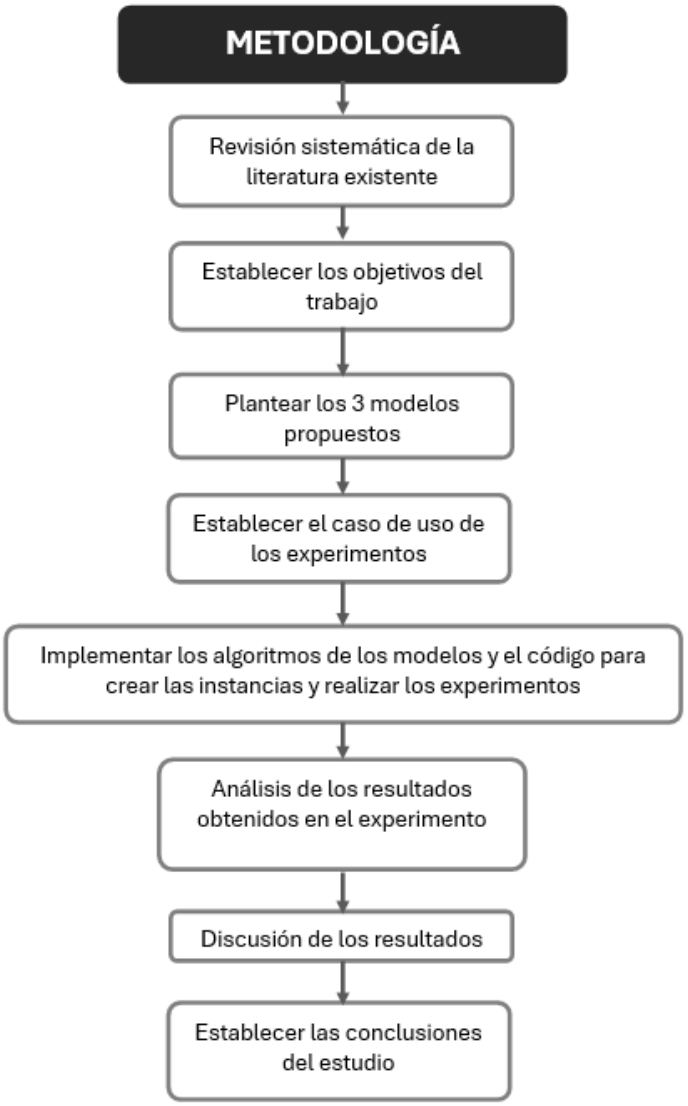

Figura 2. Metodología del trabajo. Elaboración propia.

4. MODELOS PROPUESTOS DE PROGRAMACIÓN LINEAL ENTERA

Para resolver el problema, se proponen 3 modelizaciones distintas. Estas 3 modelizaciones parten del supuesto de que tenemos un conjunto *n* de alumnos y unos tamaños de equipo deseados. Esto nos permitirá saber cuántos equipos debemos formar y de qué tamaño deberá ser cada uno de ellos. Por tanto, el objetivo de estos modelos será encontrar una determinada partición de los alumnos en los equipos deseados cumpliendo una serie de restricciones impuestas por los docentes. Esta partición de alumnos, además, deberá de ser tal que se maximice el número de características distintas que hay en los equipos. Veámoslo en un ejemplo pequeño. Supongamos que tenemos un conjunto de 6 alumnos (*n=6*) y queremos formar grupos de 3 personas. En este caso, la única forma de conseguirlo es formando 2 equipos de 3 personas. Para el ejemplo, supondremos que hay 5 características distintas representadas por las letras A, B, C, D y E. Estas son características generales que se pueden basar en diferentes criterios (personalidad, roles, habilidades sociales, etc.). En el ejemplo en el que estamos trabajando, las características que tiene presentes cada uno de los alumnos son las siguientes:

- Alumno 1: A y B
- Alumno 2: C, D y E
- Alumno 3: B y E
- Alumno 4: A y E
- Alumno 5: A, C y D
- Alumno 6: B y C

Además, el docente impone la restricción de que los alumnos 1 y 4 vayan juntos y que los alumnos 2 y 5 vayan en equipos separados. Una partición posible de los alumnos que cumpliría con las restricciones sería la siguiente:

- Equipo 1: Alumnos 1, 4 y 5
- Equipo 2: Alumnos 2, 3 y 6

Esta solución es factible ya que respeta los tamaños deseados y las restricciones impuestas por el docente. Además, tenemos en cada equipo un número de características presente:

- Equipo 1: Características A, B, C, D y E
- Equipo 2: Características B, C, D y E

La función objetivo (suma de la heurística de evaluación del equipo 1 con la heurística de evaluación del equipo 2) tiene un valor de 9, ya que hay 5 características distintas en el equipo 1 y 4 en el equipo 2.

Sin embargo, hay otra solución posible (en este caso, solo hay 2 soluciones posibles debido a las restricciones impuestas por el docente):

- Equipo 1: Alumnos 1, 2 y 4

- Equipo 2: Alumnos 3, 5 y 6

Esta solución es factible ya que respeta los tamaños deseados y las restricciones impuestas por el docente. Además, tenemos en cada equipo un número de características presente:

- Equipo 1: Características A, B, C, D y E
- Equipo 2: Características A, B, C, D y E

En este caso, la función objetivo tiene un valor de 10, ya que hay 5 características distintas en cada uno de los equipos. Por tanto, esta sería la solución óptima.

La forma en la que están definidos estos modelos engloba nuestro trabajo dentro de los "Modelos basados en la asignación", específicamente en el caso de "Formación de varios equipos". Estos 3 modelos, tienen una serie de definiciones comunes que vamos a relacionar con el ejemplo anterior:

- $\boldsymbol{\mathcal{S}} = \{\mathbf{1}, \dots, \boldsymbol{S}\}$ es el conjunto de estudiantes de una clase. **Ej.:** En nuestro ejemplo, tenemos 6 alumnos y, por tanto, $\mathcal{S} = \{1, 2, 3, 4, 5, 6\}$

- $\boldsymbol{\mathcal{C}}$ es un conjunto de tuplas de estudiantes que deben estar en el mismo equipo. Incluye al propio alumno si no está obligado a ir con ningún otro estudiante. **Ej.:** Hay una única tupla formada por 2 alumnos, los alumnos 1 y 4. Las demás tuplas están formadas por un único alumno y son las formadas por los alumnos restantes, es decir, los alumnos 2, 3, 5 y 6. Por tanto, $\mathcal{C} = \{(1,4), 2, 3, 5, 6\}$.

- $\boldsymbol{\mathcal{N}} \subset \boldsymbol{\mathcal{S}} \times \boldsymbol{\mathcal{S}}$ es el conjunto de parejas de estudiantes que no deben ir en el mismo equipo. Tanto este conjunto como el anterior vienen motivados por requisito de docentes con los que se ha trabajado y han pedido esta funcionalidad a lo largo del tiempo. **Ej.:** Hay una única pareja, $\mathcal{N} = \{(\mathbf{2}, \mathbf{5})\}$

- $\boldsymbol{\mathcal{L}} = \{\mathbf{1}, \dots, \boldsymbol{L}\}$ es el conjunto de tamaños de equipo permitidos. **Ej.:** En nuestro caso, solo podemos formar equipos de 3 alumnos, por lo que $\mathcal{L} = \{3\}$.

- $\boldsymbol{\mathcal{T}} = \{\mathbf{1}, \dots, \boldsymbol{T}\}$ es el conjunto de equipos factibles. Un equipo i es factible si y solo si $\forall (j,k) \in \mathcal{N}$, los estudiantes j y k no aparecen en el equipo i y el tamaño del equipo i es l con $l \in \mathcal{L}$. Es decir, un equipo es factible si y solo si no contiene ninguna pareja de alumnos que deban ir en equipos separados y tiene alguno de los tamaños deseados. **Ej.:** En nuestro caso:

$$\mathcal{T} = \{(1,2,3), (1,2,4), (1,2,6), (1,3,4), (1,3,5), (1,3,6), (1,4,5), (1,4,6), (1,5,6), (2,3,4), (2,3,6), (2,4,6), (3,4,5), (3,4,6), (3,5,6), (4,5,6)\}$$

  Existen 20 combinaciones distintas de agrupar a los alumnos de 3 en 3 $\left(\binom{6}{3} = 20\right)$. Además, existen 4 combinaciones de equipos de 3 alumnos en los que los alumnos 2 y 5 van juntos (Los dos primeros componentes del equipo están fijados ya que son los alumnos 2 y 5 y para el tercer componente tenemos 4 opciones, los 4 alumnos restantes). Por tanto, existen 16 combinaciones distintas de agrupar a los alumnos de 3 en 3 sin que los alumnos 2 y 5 vayan juntos. Estas combinaciones son las que formaran nuestro conjunto $\mathcal{T}$.

- $\boldsymbol{Q} = \{\mathbf{1}, \dots, \boldsymbol{Q}\}$ es el conjunto de características que queremos que estén presentes en un equipo. Ejemplos de características pueden ser el conocimiento, el estilo de aprendizaje o los rasgos de personalidad. **Ej.:** En nuestro caso $Q = \{\mathrm{A}, \mathrm{B}, \mathrm{C}, \mathrm{D}, \mathrm{E}\}$.

4.1. MODELO 1: modelo de elección de equipos

Empezamos definiendo un primer modelo que se basa en la elección de los equipos a partir del conjunto de todos los equipos factibles. A cada uno de estos equipos factibles le asocia una variable de decisión binaria que consiste en la elección o no del equipo para la partición final de la clase.

Antes de presentar la modelización del problema, veamos las definiciones de algunos parámetros particulares de este modelo:

- $\boldsymbol{u_t}$: Es una constante que representa la calidad del equipo t. Cuenta el número de características de Q presentes en el equipo t. Está relacionada con el número de características presentes en el equipo t. En este caso, representa el valor de la heurística de evaluación de equipos para el equipo t.

- $\boldsymbol{\gamma_{t,c}}$: Es una constante que vale 1 cuando los todos los estudiantes de la tupla c están en el equipo t, 0 en caso contrario.

- $\boldsymbol{\alpha_{t,l}}$: Es una constante que vale 1 cuando el equipo t tiene tamaño $l, 0$ en caso contrario.

- $\boldsymbol{n_l}$: Es una constante que representa el número de equipos deseados con tamaño l.

Algunos de estos parámetros, los usaremos también en los otros dos modelos. A continuación, vamos a modelizar el problema. Para ello, definiremos las variables de decisión, la función objetivo y las restricciones.

En este caso, las variables de decisión son variables asociadas a los equipos del conjunto $\mathcal{T}$:

- $\boldsymbol{x_t}$ es una constante que representa con un 1 si el equipo t es seleccionado, 0 en caso contrario. $\forall t \in \mathcal{T}$.

La función objetivo que deberemos maximizar es la siguiente:

$$\max \sum_{\mathrm{t} \in \mathcal{T}} \mathrm{u_t} \times \mathrm{x_t}$$

Por último, las restricciones son las siguientes:

$$\sum_{t \in \mathcal{T}} \gamma_{t,c} \times x_t = 1 \; \forall c \in \mathcal{C},$$

$$\sum_{t \in \mathcal{T}} \alpha_{t,l} \times x_t = n_l \;\; l \in \mathcal{L}$$

$$x_t \in \{0,1\}$$

Veamos una explicación más detallada de este modelo. En este modelo, cada uno de los equipos factibles (equipos que tienen uno de los tamaños deseados y no contienen parejas de alumnos que no pueden ir juntos) tiene asociada una variable binaria, que indicará si el equipo es seleccionado para formar la composición de grupos. Aunque definamos equipos factibles de esta manera, también hay que tener en cuenta que los alumnos que se han definido que deban de ir juntos, lo hagan. Veremos que es en la primera restricción donde se garantiza esta condición.

Una vez que tenemos definidas las variables del modelo, veamos la función objetivo. Esta consiste en la calidad total de la partición de alumnos formada, es decir, la suma de las heurísticas de evaluación de los equipos escogidos. La heurística de evaluación de un equipo representa el número de características distintas que están presentes en el equipo y viene dada por el parámetro $\mathrm{u_t}$. Se trata de un parámetro ya que se calcula para todos los equipos factibles, es decir, no depende de la partición final. En cambio, la partición final influye en qué heurísticas de evaluación se incluirán para el cálculo de la calidad total. Es decir, si un equipo forma parte de la partición final de los alumnos, su heurística de evaluación se considerará como un sumando para el cálculo del valor objetivo. Como es lógico, deberemos de maximizar esta función objetivo ya que queremos formar equipos diversos.

Por último, vemos las restricciones. Por un lado, la primera restricción procura que cada tupla de alumnos esté asignada exactamente a un equipo. Por otro lado, la segunda restricción procura que el número de equipos escogidos de cada tamaño sea el deseado.

A priori, la ventaja que podemos detectar en este modelo es su sencilla legibilidad e interpretación ya que resulta muy intuitivo. Además, tiene únicamente 2 restricciones y las variables son unidimensionales. Otra ventaja es que $\mathrm{u_t}$ podría ser cualquier función heurística de evaluación de equipos sin importar si estamos intentando maximizar la heterogeneidad o no.

4.2. MODELO 2: modelo de asignación de tuplas a equipos

Este segundo modelo parte de un conjunto de equipos deseados. En un primer momento, estos equipos están vacíos y se van asignando tuplas de alumnos a los equipos. Hay una variable asignada a cada tupla y cada equipo deseado que indicará si la tupla se ha asignado al equipo o no.

Para esta forma de modelizar el problema de la formación de equipos en el aula, también tenemos que definir algunos conjuntos y parámetros que usaremos:

- $\boldsymbol{\mathcal{P}} = \{\boldsymbol{1}, \dots, \boldsymbol{P}\}$ es el conjunto de equipos deseados. Ej.: En el ejemplo con el que estamos trabajando en esta sección $\mathcal{P} = \{1,2\}$ ya que queremos formar 2 grupos.

- $\boldsymbol{h_{q,c}}$: Es un parámetro que vale 1 cuando el conjunto de estudiantes c tiene la característica q, 0 en caso contrario.

- $\boldsymbol{l_p}$: Es un parámetro que mide el tamaño que ha de tener el equipo que ocupe la posición p.

Veamos como sería la modelización del problema en este caso. Para ello, definimos las variables de decisión, la función objetivo y las restricciones.

En este caso, tenemos dos tipos de variable de decisión. Por un lado, tenemos unas variables bidimensionales asociadas a cada tupla de alumnos de $\mathcal{C}$ y a cada posición del conjunto $\mathcal{P}$.

- $\boldsymbol{x_{c,p}}$: Es una variable binaria que representa con un 1 si los estudiantes c son asignados al equipo p-ésimo, 0 en caso contrario. $\forall(\mathrm{c},\mathrm{p}), \mathrm{c} \in \mathcal{C}, \mathrm{p} \in \mathcal{P}$

Por otro lado, tenemos unas variables auxiliares también bidimensionales. Estas están asociadas a cada característica del conjunto $\mathcal{Q}$ y a cada posición del conjunto $\mathcal{P}$. Son variables auxiliares ya que están completamente determinadas por las variables anteriores. Es decir, cuando las variables $x_{c,p}$ están fijadas, no hay margen de decisión en las siguientes variables:

- $\boldsymbol{y_{q,p}}$: Es una variable binaria que representa con un 1 si la característica q está presente en el equipo p-ésimo, 0 en caso contrario. $\forall(q,p), q \in \mathcal{Q}, p \in \mathcal{P}$

A continuación, vemos la función objetivo. Ésta trata de maximizar la suma de las últimas variables que hemos definido. De nuevo, la función objetivo es la suma de las heurísticas de evaluación de los equipos:

$$max \sum_{q \in \mathcal{Q}} \sum_{p \in \mathcal{P}} y_{q,p}$$

Por último, tenemos las restricciones:

$$\sum_{p \in \mathcal{P}} x_{c,p} = 1 \qquad c \in C,$$

$$\sum_{c \in \mathcal{C}} |c| \times x_{c,p} = l_p \qquad p \in \mathcal{P},$$

$$x_{s,p} + x_{s',p} \leq 1 \quad (s,s') \in \mathcal{N}, p \in \mathcal{P},$$

$$y_{q,p} \leq \sum_{c \in \mathcal{C}} h_{q,c} \times x_{c,p} \qquad q \in \mathcal{Q}, p \in \mathcal{P}$$

$$x_{c,p} \in \{0,1\}$$

$$y_{q,p} \in \{0,1\}$$

En este modelo, el problema se plantea de una forma ligeramente distinta a como se hacía en el anterior. En este caso, no usamos en ningún momento el conjunto de equipos factibles $\mathcal{T}$, sino que tenemos un conjunto $\mathcal{P}$ que enumera los equipos deseados. Cada uno de estos equipos deseados tiene asignado el tamaño l que va a tener. El modelo irá

asignando tuplas de alumnos de $\mathcal{C}$ a un equipo de $\mathcal{P}$. De esta forma, tenemos unas variables de decisión binarias que indican para cada tupla y para cada equipo escogido, si la tupla está incluida en el equipo. Esto ya nos garantiza que las tuplas incluidas en $\mathcal{C}$ no se van a separar ya que los trata como un único alumno al que asignará en algún equipo. Además, tenemos otras variables binarias que, para cada característica y para cada uno de los equipos de $\mathcal{P}$, indican si la característica está presente en el equipo.

En este modelo, aunque la función objetivo esté planteada de forma diferente a la del modelo 1, consiste también en evaluar la calidad total de la partición resultante de los alumnos. Esta calidad se mide del mismo modo que en el modelo 1, es decir, sumando las heurísticas de evaluación de los distintos equipos formados. La heurística de evaluación de un equipo se calcula de la siguiente forma:

$$\sum_{q \in Q} y_{q,p}$$

En este caso, la heurística de evaluación depende de las variables ya que no se calcula para todos los equipos factibles como se hacía en el modelo anterior, sino que se calcula para los equipos formados finales que dependen de las variables.

Por último, veamos las restricciones. En este caso, el número de restricciones es el doble que en el Modelo 1. La primera restricción, garantiza que cada tupla de alumnos esté incluida exactamente en un equipo. Con la segunda restricción, se procura que los todos los equipos que se formen tengan el tamaño deseado. Además, con la restricción 3, aseguramos que, si un miembro de una pareja de las que no pueden ir juntas va en un equipo, el otro miembro de la pareja no irá en ese equipo. Por último, la cuarta restricción procura que la variable binaria que indica si el equipo deseado tiene un rol determinado, sea menor o igual al número de tuplas del equipo donde esté presente el rol. De esta forma, si ninguna de las tuplas que forma el equipo tiene el rol, la variable valdrá 0 y si, de lo contrario, está presente en alguna de las tuplas, como esta variable está sumando en la función objetivo que tratamos de maximizar, adquirirá el máximo valor posible que, al ser binaria, será 1.

La ventaja de este modelo es que nos evitamos la combinatoria de equipos factibles ya que no partimos de ellos y vamos escogiendo, sino que, vamos formando los equipos deseados de cero asignando tuplas. Además, aunque tenemos 2 tipos de variables distintas bidimensionales seguimos teniendo un número menor de variables. Las posibles desventajas frente al primer modelo son que, en este caso, tenemos 4 restricciones y que el modelo es más difícil de interpretar.

4.3. MODELO 3: modelo de asignación de posiciones a equipos

Por último, este modelo mezcla aspectos de los dos modelos anteriores. Por un lado, volvemos a tener el conjunto de todos los equipos factibles y, por otro lado, tenemos el conjunto de equipos deseados. En este caso, se trata de escoger qué equipos se escogerán y que posición tomarán en el conjunto de equipos deseados. Hay una variable de decisión asociada a cada equipo y posición que indicará si el equipo toma la posición o no.

Definimos, una última vez, parámetros que servirán para este caso concreto:

- $\boldsymbol{u_t}$: Es una constante que representa la calidad del equipo t, basada en las características presentes en el equipo.

- $\boldsymbol{\gamma_{t,c}}$**:** es una constante que vale 1 si el equipo t contiene todos los estudiantes en c, 0 en caso contrario.

- $\boldsymbol{i_t}$: Es un orden lexicográfico para equipos del mismo tamaño.

Para esta modelización del problema, usaremos unas variables de decisión bidimensionales que están asociadas a cada equipo del conjunto $\mathcal{T}$ y cada posición del conjunto $\mathcal{P}$. Sin embargo, solamente existirán cuando el tamaño del equipo de $\mathcal{T}$ sea el deseado para el equipo de la posición del conjunto $\mathcal{P}$:

- $\boldsymbol{x_{t,p}}$: Es una variable binaria que representa con un 1 si el equipo t toma la posición p en la partición de la clase, 0 en caso contrario. $\forall(t,p), t \in \mathcal{T}, p \in \mathcal{P}, |t| = l_p$

La función objetivo es muy similar a la del primer modelo:

$$max \sum_{t \in T} \sum_{p \in \mathcal{P}} u_t \times x_{t,p}$$

Por último, estas son las restricciones que se consideran:

$$\sum_{p \in \mathcal{P}} \sum_{t \in \mathcal{T}} \gamma_{t,c} \times x_{t,p} = 1 \qquad c \in \mathcal{C}$$

$$\sum_{t \in \mathcal{T}} x_{t,p} = 1 \qquad p \in \mathcal{P}$$

$$\sum_{t \in \mathcal{T}} i_t \times x_{t,p} \leq \sum_{t \in \mathcal{T}} i_t \times x_{t,p'} \qquad p, p' \in \mathcal{P}, p < p', l_p = l'_p$$

$$x_{t,p} \in \{0,1\}$$

Veamos también una explicación detallada de este último modelo propuesto. Volvemos a una estructuración bastante cercana a la del Modelo 1 en ciertas cosas, pero también combina con algunos aspectos del modelo 2. Por un lado, volvemos a tener el conjunto $\mathcal{T}$ de equipos factibles con unas variables de decisión binarias asociadas. Sin embargo, estas variables, en lugar de indicar simplemente si escogemos el equipo, también van asociadas al conjunto $\mathcal{P}$ que hemos visto en el modelo 2 e indican si el equipo toma la posición p. Además, se añade un nuevo parámetro que no aparece en ninguno de los modelos previos, el parámetro i que, a cada equipo factible, le asigna una posición dentro de los equipos con el mismo tamaño que él.

Una vez que tenemos definidas las variables del modelo, veamos la función objetivo. Esta es igual que la del modelo 1 con un ligero cambio ya que, al tener unas variables de decisión bidimensionales, tenemos un doble sumatorio.

Por último, debemos de entender las restricciones del modelo. La primera restricción es equivalente a la primera restricción del modelo 1, por tanto, procura que todas las tuplas de alumnos estén asignadas exactamente en un equipo. La segunda restricción garantiza que hay un único equipo asignado en cada posición de $\mathcal{P}$. La última restricción, aunque no restringe nada, sirve para limitar las combinaciones equivalentes ya que, si dos equipos del mismo tamaño se escogen, no importa cual se asigne antes.

A priori, este modelo no parece tener ventajas sobre el modelo 1 ya que tiene más variables y restricciones y se realiza el cálculo de todos los equipos factibles. Respecto al modelo 2, la ventaja que puede tener es que tiene una restricción menos y solamente un tipo de variables.

Aunque los 3 modelos plantean el problema de forma distinta, deberán de obtener el mismo valor objetivo para el mismo problema ya que la función objetivo es equivalente para los tres modelos. Por tanto, debemos de comparar la eficiencia de los modelos, es decir, el tiempo de computación. Sabemos que estos tiempos dependerán de las capacidades del equipo en el que se ejecute. Por tanto, debemos fijarnos no tanto en los tiempos de ejecución sino en qué modelos tardan más y qué modelos tardan menos.

5. CASO DE ESTUDIO: TEORIA DE LOS ROLES DE BELBIN

Nuestros modelos se centran en garantizar la heterogeneidad de las características en los equipos. Para poder probar estos modelos necesitamos un caso de estudio. Este caso de estudio se basará en la presencia o no de los diferentes roles de Belbin en el equipo. Sin embargo, es importante recalcar que los modelos descritos anteriormente son independientes de este criterio ya que también podríamos escoger, entre otros, el Myers-Briggs Type Indicator (Briggs Myers, 1998), el Big Five Inventory (John, Naumann, & Soto, 2008) o el 16 Personality Factor Questionnaire (Cattell, 1949).

Varios estudios demuestran que el ambiente del aula puede beneficiarse de la aplicación de la teoría de Belbin. Belbin define un rol de equipo como "una tendencia a comportarse, contribuir e interrelacionarse con otros de una manera particular" (Belbin, 1981) y afirma que los equipos necesitan un equilibrio adecuado entre los roles. Además, ha categorizado el comportamiento individual dentro del equipo en ocho roles, aunque los equipos no necesariamente deben estar compuestos por ocho miembros. Según la tarea a realizar, se pueden precisar grupos pequeños en los que no puede haber 8 miembros. Además, una persona puede tener más de un rol presente. (Alberola, del Val, Sanchez-Anguix, Palomares, & Teruel, 2016).

La teoría de Belbin afirma que existen 8 tipos de roles predominantes y que, cada individuo tendrá presentes un conjunto de estos roles (Alberola, del Val, Sanchez-Anguix, Palomares, & Teruel, 2016). En revisiones posteriores se introdujo un noveno rol, el especialista, para cuando se precisara experiencia técnica. Sin embargo, el rol de especialista no se considera en este estudio ya que, dentro del aula, se supone que no hay especialistas.

En el modelo de Belbin, un rol está definido por seis factores: personalidad, capacitad mental, valores actuales y motivación, limitaciones en el campo, experiencia y aprendizaje de roles (Aritzeta, Swailes, & Senior, 2007). Los roles de los que habla Belbin y que utilizaremos son los siguientes:

- Implementador (Implementer-CW): es práctico, confiable y eficiente. convierte ideas en acciones y organiza el trabajo que debe realizarse
- Coordinador (Coordinator-CH): Es maduro y confiado. Tiene una visión global del proyecto y delega eficazmente.
- Impulsor (Shaper-SH): es desafiante y dinámico. Tiene el empuje y el coraje para superar obstáculos.
- Cerebro/Creativo (Plant-PL): Es creativo e imaginativo. Genera ideas y resuelve problemas difíciles.
- Investigador de recursos (Resource Investigator-RI): Es extrovertido y comunicativo. Explora oportunidades e interactúa con personas ajenas al equipo.
- Monitor evaluador (Monitor Evaluator-ME): Es sobrio, estratégico y exigente. Ve todas las opciones y juzga con precisión.
- Cohesionador (Teamworker-TW): es cooperativo, perspicaz y diplomático. Es capaz de escuchar y evitar fricciones.
- Finalizador (Completer Finisher-CF): Es esmerado, concienzudo y ansioso. Busca errores, los pule y los perfecciona.

5.1. Cuestionario Belbin

La prueba Belbin Team Role Self-Perception Inventory (BTRSPI) es ahora una de las herramientas más utilizadas para identificar la fuerza relativa de las preferencias o afinidad de roles de equipo de un individuo, con el fin de formar y mantener equipos que sean fuertes en todas las áreas de rol de equipo. Esta prueba identifica los ocho roles de comportamiento (Ugarte, Aranzabal, Arruarte, & Larrañaga, 2022, October). El cuestionario (belbin team roles test pdf, 2024) consta de 7 secciones, en cada una de estas secciones se plantea una situación relacionada con el trabajo en equipo y se proporcionan 8 posibles respuestas. El alumno que esté realizando el cuestionario debe escoger qué respuestas se adaptan más a él y repartir 10 puntos entre las respuestas que ha escogido. En versiones más modernas del cuestionario hay 9 roles ya que se incluye el rol de especialista en la materia.

Una vez que el alumno ha completado todas las secciones, deberá sumar los puntos que les ha dado a las respuestas asociadas con cada rol y, por tanto, obtendrá unas puntuaciones para cada uno de los roles de Belbin.

Hay un umbral establecido para cada rol, de forma que, si la puntuación de un alumno obtenida en el cuestionario para ese rol es igual o mayor que ese umbral, se considerará que el rol está presente en el alumno. Los umbrales vienen dados por la Tabla 2 obtenida de (Sanchez-Anguix, Alberola, Del Val, Palomares, & Teruel, 2023).

ROL	CW	CH	SH	PL	RI	ME	TW	CF
Umbral	12	11	14	9	10	10	13	7

Tabla 2. Umbrales mínimos para la presencia de los roles de Belbin

Es decir, si por ejemplo un alumno ha obtenido la puntuación en el cuestionario de los roles de Belbin que viene dada por la Tabla 3, este alumno tiene presente los siguientes roles:

- Coordinador
- Implementador
- Shaper
- Teamworker
- Completer finisher.

ROL	CW	CH	SH	PL	RI	ME	TW	CF
Puntuación alumno	14	13	44	3	3	10	16	7

Tabla 3. Ejemplo de puntuación obtenida en el cuestionario de roles de Belbin para un alumno

Para poder seguir trabajando con el ejemplo anterior, supongamos que los 6 alumnos que debemos de dividir en 3 grupos han obtenido la siguiente puntuación en el cuestionario de roles de Belbin dada por la Tabla 4.

student_id	CW	CH	SH	PL	RI	ME	TW	CF
1	14	13	44	3	3	0	16	7
2	15	9	14	15	12	6	9	19
3	17	0	10	26	7	10	7	22
4	11	16	20	11	13	9	10	10
5	12	14	14	11	11	12	14	12
6	7	21	7	10	17	1	27	9

Tabla 4. Ejemplo de puntuación obtenida en el cuestionario de roles de Belbin para seis alumnos

Si transformamos la Tabla 4 en una tabla binaria en la que 1 indica que el alumno tiene presente el rol y 0 indica que no lo tiene presente, obtenemos la Tabla 5.

student_id	CW	CH	SH	PL	RI	ME	TW	CF
1	1	1	1	1	1	0	1	1
2	1	0	0	1	1	0	0	1
3	1	0	0	1	1	1	0	1
4	0	1	0	1	1	0	0	1
5	0	1	0	1	1	1	0	1
6	0	1	0	1	1	0	1	1

Tabla 5. Ejemplo de roles de Belbin presentes en 6 alumnos

Vamos a ver cómo cambia cada uno de los 3 modelos propuestos al trabajar con el caso concreto de Belbin.

5.2. Caso de estudio aplicado a los modelos

Veamos cómo se aplica este caso de estudio a los 3 modelos propuestos en la sección 4. Tanto en el primer modelo como en el último, se utiliza el parámetro u_t para medir la calidad de los equipos. Para poder implementar los resultados del cuestionario de Belbin, esta constante mide la calidad de cada equipo como la cantidad de roles diferentes que están presentes en ese equipo. De este modo $0 \leq u_t \leq 8$ para cada $t \in T$.
Siguiendo el ejemplo anterior, nuestro conjunto de equipos factibles es el siguiente:

$$\mathcal{T} = \{(1,2,3),(1,2,4),(1,2,6),(1,3,4),(1,3,5),(1,3,6),(1,4,5),(1,4,6),(1,5,6)$$
$$(2,3,4),(2,3,6),(2,4,6),(3,4,5),(3,4,6),(3,5,6),(4,5,6)\}$$

De este modo, el parámetro u viene dado por $u = (8,7,7,8,8,8,8,7,8,6,7,6,6,7,7,6)$.
Hay una diferencia entre estos dos modelos en cuanto a la manera de calcular la función objetivo. En el caso del primer modelo, se suman los elementos de u correspondientes a los equipos que se escogen. Por ejemplo, si escogemos los equipos $(1,2,4)$ y $(3,5,6)$, las variables x_t vendrían dadas por el siguiente vector:

$$x = (0,1,0,0,0,0,0,0,0,0,0,0,0,0,1,0)$$

La función objetivo sería el producto de los vectores u y x^T:

$$\sum_{t \in \mathcal{T}} u_t \times x_t = 1 \cdot 7 + 1 \cdot 7 = 14$$

En cambio, en el tercer modelo, aparte de escoger estos dos equipos, se les asigna un orden. Supongamos que el primer equipo es el $(1,2,4)$ y el segundo es el $(3,5,6)$. Entonces, las variables de decisión se representarían con la siguiente matriz:

$$x = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}^T$$

La función objetivo sería la suma de los elementos de la matriz resultante del producto de u y x:

$$\sum_{t \in T} \sum_{p \in \mathcal{P}} u_t \times x_{t,p} = \sum_{\mathrm{i}} (\mathrm{u} \cdot x)_i = \sum_{\mathrm{i}} (7 \quad 7)_i = 14$$

Si nos basamos ahora en el segundo modelo, no necesitamos el parámetro u_t, sino otro parámetro llamado $h_{q,c}$ que indica si el rol está presente en cada tupla de alumnos. En nuestro caso, $\mathcal{C} = \{(1,4), 2, 3, 5, 6\}$. Por tanto, los parámetros $h_{q,c}$ vienen dados por la siguiente matriz:

$$h = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

Las variables de decisión $\boldsymbol{x_{c,p}}$, si escogemos los equipos $(1,2,4)$ y $(3,5,6)$ en este mismo orden, vienen dadas por la matriz:

$$x = \begin{pmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \end{pmatrix}$$

No obstante, en este caso, para calcular la función objetivo en este modelo, necesitamos las variables auxiliares $\boldsymbol{y_{q,p}}$, cada una de ellas se calcula con la siguiente fórmula:

$$\boldsymbol{y_{q,p}} \leq \sum_{c \in \mathcal{C}} h_{c,q} \times x_{c,p}$$

Por ejemplo, para $\boldsymbol{q = p = 1}$,

$$\boldsymbol{y_{q,p}} \leq \sum_{c \in \mathcal{C}} h_{q,c} \times x_{c,p} = (1 \quad 1 \quad 1 \quad 0 \quad 0) \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} = 2$$

Para el cálculo de todas estas variables auxiliares, podemos considerar la siguiente fórmula:

$$\boldsymbol{y} \leq \boldsymbol{h} \cdot \boldsymbol{x} = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 2 & 1 \\ 1 & 2 \\ 1 & 0 \\ 2 & 3 \\ 2 & 3 \\ 0 & 1 \\ 1 & 1 \\ 2 & 3 \end{pmatrix}$$

Si consideramos que el número de veces que un rol está presente en un equipo es el número de tuplas que forman el equipo en las cuales dicho rol está presente, la matriz resultante de la derecha representa para cada posición y cada rol, el número de veces que el rol está presenta en el equipo que ocupa dicha posición. Por ejemplo, el primer elemento de la matriz indica que en el equipo que ocupa la primera posición, es decir, el equipo formado por los alumnos 1, 2 y 4, el rol de implementador está presente 2 veces y, por tanto, la variable auxiliar binaria que representa si el rol está presente en el equipo, al estar maximizando. Como los parámetros y_t son binarios y estamos maximizándolos, siempre que sea posible, se les asignará el valor 1. Por tanto, en nuestro ejemplo, tomarán los valores dados por la siguiente matriz:

$$\boldsymbol{y} = \begin{pmatrix} 1 & 1 \\ 1 & 1 \\ 1 & 0 \\ 1 & 1 \\ 1 & 1 \\ 0 & 1 \\ 1 & 1 \\ 1 & 1 \end{pmatrix}$$

Por tanto, el valor objetivo, será la suma de los elementos de la matriz y:

$$\sum_{q \in Q} \sum_{p \mathcal{P}} y_{q,p} = 14$$

En los 3 modelos hemos obtenido el mismo resultado ya que son modelos equivalentes, no obstante, hemos visto que hay diferencias en la forma en la que se plantea el problema. Estas diferencias podrán resultar en diferencias de rendimiento de los algoritmos diseñados para resolverlos. De este modo, este será el propósito general del estudio.

5.3. Datos del experimento

Contamos con una base de datos de 383 datos de alumnos del Grado de Turismo de la UPV, recogidos entre 2015 y 2019. Estos datos son anonimizados y son los resultados del cuestionario de los roles de Belbin y de la prueba de las 16 personalidades de Myer Briggs junto con el género del alumno. En nuestro caso, solamente usaremos los resultados del cuestionario de los roles de Belbin. No todos los alumnos realizaron la prueba de Belbin. De estos 383 alumnos, tenemos resultados del cuestionario de Belbin de 343.

5.4. Generación de instancias

Para poder evaluar los 3 modelos que vamos a proponer, necesitamos contar con instancias. Estas instancias vamos a generarlas a partir de los datos de los alumnos de los que disponemos. Para poder comparar los modelos en distintas condiciones, estas instancias irán desde grupos de 20 alumnos hasta grupos de 60 alumnos. Además, debido a la aleatoriedad a la hora de crear las instancias, para que los resultados sean más fiables, crearemos 10 instancias para cada uno de estos tamaños (los distintos tamaños que usaremos son 20, 30, 40, 50 y 60). Por tanto, tendremos 50 instancias con las que podremos validar nuestros resultados.

En este experimento, estas 50 instancias únicamente incluirán los alumnos que participarán. Es decir, no incluiremos ningún grupo de alumnos que necesariamente hayan de ir en un mismo equipo como tampoco habrá parejas de alumnos que no puedan ir en un mismo equipo. Para la obtención de estas instancias se elaboró un script en el que, para cada tamaño T de los propuestos y para cada instancia de 1 a 10, se escoge un conjunto formado por T alumnos al azar del conjunto de datos con los alumnos disponibles. A continuación, se guarda en un Excel nuevo la posición que ocupan los alumnos escogidos.

6. IMPLEMENTACIÓN DE LOS MODELOS

El algoritmo diseñado tiene como objetivo que el usuario pueda introducir una serie de datos relativos a los alumnos y la herramienta le devuelva una de las combinaciones óptimas de alumnos.

En esta sección detallamos como se han implementado los modelos previamente definidos. Empezaremos viendo las herramientas utilizadas para la implementación. A continuación, veremos cómo está organizado este código y, por último, explicaremos el pseudo-código de los modelos.

6.1. Herramientas utilizadas para la implementación

Hemos programado los modelos con Python (Python, 2024). Python es un lenguaje de programación de alto nivel, interpretado y de propósito general. Este es versátil y se utiliza en una amplia gama de aplicaciones, desde desarrollo web y científico hasta scripting y automatización de tareas. Es ampliamente utilizado en la industria tecnológica y en la academia debido a su gran cantidad de bibliotecas y su comunidad activa de desarrolladores. Además, juega un papel fundamental en la implementación y solución de problemas de optimización exacta, facilitando el desarrollo de soluciones eficientes y efectivas.

Usaremos el módulo gurobipy (Gurobi Optimization, LLC, 2024) que proporciona Gurobi Optimization. Es una interfaz de Python para el software de optimización Gurobi (Gurobi Optimization, LLC, 2024). El objetivo principal de este módulo es permitir a los usuarios formular y resolver problemas de optimización utilizando Gurobi directamente desde Python. Gurobi es un potente software de optimización matemática utilizado para resolver problemas de programación lineal, programación entera mixta, programación cuadrática, y otros tipos de problemas de optimización. Es una de las herramientas más utilizadas en el ámbito académico e industrial para abordar problemas complejos de toma de decisiones que se pueden modelar como problemas de optimización. Proporciona una interfaz amigable para varios lenguajes de programación, como Python, Java, MATLAB y C++. Es conocido por su eficiencia y rendimiento en la resolución de problemas de optimización de gran escala.

Gurobi Optimization, la empresa que desarrolla Gurobi, proporciona una interfaz Python llamada gurobipy que permite utilizar las capacidades de Gurobi directamente desde Python. Este módulo facilita la formulación y resolución de problemas de optimización mediante Gurobi.

Python cuenta con varios módulos de optimización que permiten resolver una amplia gama de problemas de optimización matemática. Algunos de estos son sciPy.optimize, PuLP, CVXPY, gurobipy o OR-Tools. Sin embargo, Gurobi es conocido por su eficiencia y alto rendimiento en la resolución de una amplia gama de problemas de optimización lineal, entera mixta y cuadrática. Algunas de las herramientas mencionadas, como ORTools, permiten interactuar con Gurobi. Aunque no se puede afirmar categóricamente que Gurobi sea siempre el más eficiente en todos los casos, tiene una reputación bien establecida por su capacidad para manejar problemas grandes y complejos de manera rápida y efectiva. Dado que la eficiencia puede facilitarnos el estudio de los modelos propuestos, hemos empleado Gurobi en nuestro estudio.

Este módulo nos permite definir las variables de decisión del problema de optimización, establecer restricciones definiendo relaciones lineales entre las variables, especificar la función que se debe maximizar o minimizar e invocar el solver de Gurobi para resolver el problema y obtener los resultados. Además, también podemos configurar los parámetros de optimización, aunque, en este caso, no lo hemos hecho.

6.2. Organización del Código

El código capaz de resolver un problema consta de dos partes. En primer lugar, se calculan los parámetros y conjuntos necesarios para los modelos, estos son todos los que hemos definido en la sección 3 y que recordamos a continuación:

- $\boldsymbol{S} = \{\mathbf{1}, \dots, \boldsymbol{S}\}$ es el conjunto de estudiantes de una clase.
- $\boldsymbol{\mathcal{C}}$ es un conjunto de tuplas de estudiantes que deben estar en el mismo equipo. Incluye al propio alumno si no está obligado a ir con ningún otro estudiante.
- $\boldsymbol{\mathcal{N}} \subset \boldsymbol{S} \times \boldsymbol{S}$ es el conjunto de parejas de estudiantes que no deben ir en el mismo equipo.
- $\boldsymbol{\mathcal{L}} = \{\mathbf{1}, \dots, \boldsymbol{L}\}$ es el conjunto de tamaños de equipo permitidos.
- $\boldsymbol{\mathcal{T}} = \{\mathbf{1}, \dots, \boldsymbol{T}\}$ es el conjunto de equipos factibles. Un equipo i es factible si y solo si $\forall (j,k) \in \mathcal{N}$, los estudiantes j y k no aparecen en el equipo i y el equipo i tiene tamaño l con $l \in L$.
- $\boldsymbol{\mathcal{Q}} = \{\mathbf{1}, \dots, \boldsymbol{Q}\}$: es el conjunto de roles que queremos que estén presentes en un equipo.
- $\boldsymbol{\mathcal{P}} = \{\mathbf{1}, \dots, \boldsymbol{P}\}$ es el conjunto de equipos deseados.
- $\boldsymbol{u_t}$: Es una constante que representa la calidad del equipo t. Cuenta el número de roles de Q presentes en el equipo t. En este caso, representa el valor de la heurística de evaluación de equipos para el equipo t.
- $\boldsymbol{\gamma_{t,c}}$: Es una constante que vale 1 cuando los todos los estudiantes de la tupla c están en el equipo t, 0 en caso contrario.
- $\boldsymbol{\alpha_{t,l}}$: Es una constante que vale 1 cuando el equipo t tiene tamaño l, 0 en caso contrario.
- $\boldsymbol{n_l}$: Es una constante que representa el número de equipos deseados con tamaño l.
- $\boldsymbol{h_{q,c}}$: Es una constante que vale 1 cuando el conjunto de estudiantes c tiene la característica q, 0 en caso contrario.
- $\boldsymbol{l_p}$: Es un parámetro que mide el tamaño que ha de tener el equipo que ocupe la posición p.
- $\boldsymbol{i_t}$: Es un orden lexicográfico para equipos del mismo tamaño.

Naturalmente, no todos los conjuntos y parámetros que se le pasan al modelo se deben de introducir a mano. Podemos distinguir tres tipos de parámetros en los modelos, según sean parámetros fijos, parámetros introducidos por el usuario (el profesor que va a formar los equipos) o parámetros calculados, a partir de los anteriores, por el algoritmo.

Como parámetros fijos tenemos el umbral mínimo de cada rol (explicado en la Tabla 2) y las puntuaciones de los alumnos en los distintos roles que se usarán para el experimento. Este se convierte en un diccionario de variables binarias asignando 1s a los elementos que superen el valor del umbral del rol correspondiente y 0 a los que no.

Los parámetros introducidos por el usuario se leen de un fichero y son el conjunto de alumnos $\mathcal{S}$, el listado $\mathcal{C}$ de tuplas de alumnos que deben de ir en el mismo equipo (aunque este lo completará el algoritmo), el listado de parejas $\mathcal{N}$ que no pueden ir en el mismo equipo y los tamaños de equipos permitidos $\mathcal{L}$. El conjunto de tamaños permitidos será como máximo de dos elementos y estos deben ser tamaños consecutivos. Es decir, un equipo de la partición final podrá tener como mucho un miembro más que otro equipo de la partición. De esta forma, estamos equilibrando también el número de miembros en los equipos. Solamente se podrán introducir dos tamaños de equipo si el número de alumnos no es divisible entre ninguno de los dos tamaños. Además, se introducirán de menor a mayor.

A partir de los parámetros mencionados, el algoritmo diseñado calculará los parámetros restantes.

- El algoritmo completa el conjunto $\mathcal{C}$ ya que el introducido por el usuario no debe contener al propio alumno si no está obligado a ir con ningún otro estudiante (en el caso de tener un tamaño considerable de alumnos, no sería muy práctico).

- El conjunto de equipos factibles $\mathcal{T}$ se calcula a partir de los conjuntos $\mathcal{S}$, $\mathcal{N}$ y $\mathcal{L}$. Esta parte del algoritmo se divide en dos fases. Primero calculamos las combinaciones de equipos que tienen alguno de los tamaños incluidos en el conjunto $\mathcal{L}$. A continuación, de estas combinaciones, escogemos únicamente las que no contienen a ninguna pareja de las incluidas en el conjunto $\mathcal{N}$.

- Los parámetros u, γ, α, η, h, l y i se calculan a partir de algunos de los parámetros que ya tenemos.

- El conjunto $\mathcal{P}$ es una lista que va del 1 al número de equipos que hay que formar.

Una vez que ya tenemos calculados todos los parámetros necesarios, el algoritmo los usa para resolver los modelos de optimización que explicamos en la subsección 6.4.

6.3. Pseudo-código explicado del cálculo de parámetros

Este cálculo de parámetros parte de los conjuntos $\mathcal{S}$, $\mathcal{C}$, $\mathcal{N}$ y $\mathcal{L}$.

- Calculamos el número de alumnos con el que trabajaremos:

  Asignar a $alumnos$ la longitud de la lista $\mathcal{S}$

- Empezamos con el cálculo del parámetro n, es decir, el número de equipos deseados para cada tamaño. Si podemos formar todos los equipos según el tamaño más grande lo hacemos, si no, si podemos formarlos todos según el tamaño más pequeño, lo hacemos. Si debemos de tener necesariamente equipos de los dos tamaños disponibles, vamos formando equipos del tamaño grande, hasta que podamos dividir los alumnos restantes en equipos del menor tamaño. Además, calculamos el número de equipos que vamos a formar (Figura 3).

```
Inicializar el parámetro n
Si tenemos dos tamaños de equipos permitidos entonces
      Asignar a alumnosrestantes el valor de alumnos
      Asignar a n_2 el valor 0
      Mientras que alumnosrestantes no sea divisible por ℒ[1] hacer
                  Restar ℒ[2] a alumnosrestantes
                  Sumar 1 a n_2
      Asignar a n_1 el cociente de alumnosrestantes dividido por ℒ[1]
      Asignar a equipos la suma de n_1 y n_2
Si no
      Asignar a equipos el cociente de alumnos dividido por ℒ[1]
      Asignar a n_1 el valor de equipos
```

Figura 3. Pseudo-código del cálculo del parámetro n

- A continuación, calculamos el conjunto de equipos deseados $\mathcal{P}$ que depende únicamente del número de equipos que debemos formar.

- Para el conjunto $\mathcal{Q}$ solo necesitamos la Tabla 2. A cada rol, le asignamos el umbral mínimo.

- Aunque ya tenemos el conjunto $\mathcal{C}$, debemos completarlo con los alumnos mismos si no deben de ir con nadie obligatoriamente. Para cada alumno del conjunto $\mathcal{S}$, vemos si está en alguna tupla del conjunto $\mathcal{C}$ y, si no lo está, lo añadimos (Figura 4).

```
Para m = 0, alumnos − 1 hacer
    Asignar la longitud de 𝒞 a la variable s
    Asignar 0 a b
    Para cada tupla c en 𝒞 hacer
        Si la tupla c no está formada por un único alumno entonces
            Si el alumno 𝒮[m] está en la tupla c entonces
                Asignar 1 a b
    Si b es igual a 0 entonces
        Incrementar s en 1 unidad
        Asignar a 𝒞[s] el valor de 𝒮[m]
```

Figura 4. Pseudo-código del cálculo del conjunto $\mathcal{C}$

- En este momento, creamos el conjunto de equipos factibles $\mathcal{T}$. Definimos una función que obtiene todas las combinaciones de un conjunto con un tamaño dado. Calculamos el conjunto con todas las combinaciones de alumnos con alguno de los tamaños deseados. A continuación, para cada uno de estos equipos, compruebo si alguno de las parejas del conjunto $\mathcal{N}$ está presente y, si no es así, lo añado al conjunto $\mathcal{T}$ y al para ese mismo equipo, al parámetro i le asignamos la posición correspondiente (Figura 5).

```
Inicializar el objeto B
Si tenemos dos tamaños de equipo permitidos entonces
    Si ℒ[1] es igual a 1 entonces
            Asignar a B la concatenación de 𝒮 y las combinaciones
            de tamaño ℒ[2] de 𝒮
    Si no
            Asignar a B la concatenación de las combinaciones de
            tamaño ℒ[1] de 𝒮 y las combinaciones de tamaño ℒ[2] de 𝒮
Si no
    Asignar a B la concatenación de las combinaciones de tamaño ℒ[1]
    de 𝒮

Inicializar el objeto 𝒯
Inicializar el objeto i
Asignar 1 a j
Asignar 0 a uno
Asignar 0 a dos
Para cada combinación de alumnos k en B hacer
    Asignar 0 a a
    Si 𝒩 no está vacío entonces
        Para cada pareja de alumnos n en 𝒩 hacer
            Asignar 0 a b
            Para cada alumno t la pareja n hacer
               Si t está en el equipo actual k entonces
                     Incrementar b en 1
            Si b es igual a 2 entonces
                Asignar 1 a a
    Si a es igual a 0 entonces
        Asignar a 𝒯[j] la combinación k
        Si k es un entero o su longitud de es igual a ℒ[1] entonces
              Incrementar uno en 1
              Asignar a i_j el valor de uno
        Si no
            Incrementar dos en 1
            Asignar a i_j el valor de dos
        Incrementar j en 1
```

Figura 5. Pseudo-código del cálculo del conjunto $\mathcal{T}$

- A continuación, calculamos el parámetro u. Recordemos que este mide el número de características presentes en cada equipo. Para cada equipo del conjunto $\mathcal{T}$, vamos mirando si cada característica está presente, si lo está, el parámetro u aumenta en una unidad (Figura 6).

```
Inicializar el parámetro u
Para cada equipo t de T hacer
    Asignar a u_t el valor 0
    Para característica q de Q hacer
        Asignar a 0 a
        Para cada alumno m del equipo t hacer
            Si el alumno m tiene presente la característica q
            entonces
               Asignar a 1 a a
               Incrementar u_t en 1
               Salir del bucle
```

Figura 6. Pseudo-código del cálculo del parámetro u

- El parámetro γ es una constante que vale 1 si el equipo t contiene todos los estudiantes en c, 0 en caso contrario. Para su cálculo, para cada tupla del conjunto $\mathcal{C}$ y equipo del conjunto $\mathcal{T}$, compruebo si la tupla entera está presente en el equipo y, si es así, le doy el valor 1 al parámetro, si no están todos, le doy un 0 (Figura 7).

```
Inicializar el parámetro γ
      Para cada tupla c de C hacer
      Para cada equipo t de T hacer
              Asignar a 0 a b
              Si la tupla c es un solo alumno entonces
                  Si el alumno está en el equipo t entonces
                      Asignar a b el valor 1
              Si no
                  Asignar a b el valor 1
                  Para cada alumno j de la tupla c hacer
                      Si j no está en el equipo t entonces
                         Asignar a 0 a b
                         Salir del bucle
              Si b es igual a 1 entonces
                   Asignar a γ_{t,c} el valor 1
              Si no
                  Asignar a γ_{t,c} el valor 0
```

Figura 7. Pseudo-código del cálculo del parámetro g

- Seguimos con el cálculo del parámetro α, que vale 1 cuando el equipo t tiene tamaño l y 0 en caso contrario. Para cada equipo del conjunto $\mathcal{T}$, comprobamos si tiene alguno de los tamaños deseados (los incluidos en el conjunto $\mathcal{L}$). Si lo tiene, le asignamos el valor 1, si no, le asignamos un 0 (Figura 8).

```
Inicializar el parámetro α
Para cada equipo t de 𝒯 hacer
      Para cada longitud l de ℒ hacer
        Si la longitud del equipo t es igual a l entonces
            Asignar a α_{t,l}
```

Figura 8. Pseudo-código del cálculo del parámetro α

- El parámetro h es una constante que vale 1 cuando el conjunto de estudiantes c tiene el rol q, 0 en caso contrario. Para su cálculo, para cada rol del conjunto Q y para cada tupla del conjunto C, comprobamos si alguno de los alumnos de la tupla, tiene el rol, si es así, le asignamos un 1 al parámetro 1 y, si no, le asignamos un 0 (Figura 9).

```
Inicializar el parámetro h
Para cada característica q de 𝒬 hacer
    Para cada tupla c de 𝒞 hacer
        Si la tupla c está formada por un único alumno entonces
            Si el alumno tiene presente la característica q entonces
                Asignar h_{c,q} el valor 1
            Si no
                Asignar a h_{c,q} el valor 0
        Sino
            Para cada alumno j de la tupla c hacer
                Si el alumno tiene la característica q entonces
                    Asignar a h_{c,q} el valor 1
                    Salir del bucle
                        Si no
                            Asignar a h_{c,q} el valor 0
```

Figura 9. Pseudo-código del cálculo del parámetro h

- Por último, calculo el parámetro l, que representa el tamaño que tiene cada elemento del conjunto P. Para su cálculo, para cada elemento de P, si su índice es menor que el número de equipos que debo formar con el primer tamaño, le asignare el primer tamaño, en caso contrario, le asignaré el segundo tamaño (Figura 10).

```
Inicializar el parámetro l
Para j = 1, equipos hacer
    Si j ≤ n_1 entonces
        Asignar a l_j el valor de ℒ[1]
    Si no
        Asignar a l_j el valor de ℒ[2]
```

Figura 10. Pseudo-código del cálculo del parámetro l

6.4. Pseudo-código explicado de los modelos

Se ha creado una función para cada modelo. Las tres funciones están estructuradas de la misma forma. Para ir entendiendo los pasos de este código, nos basaremos en el modelo 1:

- Se crea un modelo llamado “Modelo 1”. Internamente, se reconocerá como “modelo”.

- Se crean las variables de decisión y se añaden al modelo como variables binarias. Para ello, Se crea un diccionario vacío llamado "variables" para almacenar las variables de decisión. Luego, para cada elemento en el conjunto de índices T, se crea una variable binaria con un nombre descriptivo y se agrega al modelo. Estas variables representan si se selecciona o no cada equipo (Figura 11).

```
Para cada equipo t de 𝒯 hacer
      Crear una variable binaria v
      Añadir la variable v al modelo
      Guardar v en la posición t de un vector llamado variables
```

Figura 11. Pseudo-código de creación las variables del modelo 1

- Se crean las restricciones y se añaden al modelo (Figura 12). Primero creamos la lista vacia constraints1, donde irán todas las subrestricciones correspondientes a la primera restricción del modelo, que aplica para cada tupla del conjunto $\mathcal{C}$. Esta restricción viene dada por la siguiente fórmula:

$$\sum_{t \in \mathcal{T}} \gamma_{t,c} \times x_t = 1 \; \forall c \in \mathcal{C},$$

Por tanto, para cada tupla, realizamos lo siguiente:

1. Creamos una expresión lineal a la cual vamos añadiendo, para cada elemento en el conjunto de índices T, el término correspondiente. Esta expresión representa el lado izquierdo de la ecuación.
2. Cuando ya hemos añadido todos los términos a la expresión lineal, creamos la restricción igualando esta expresión con la otra parte de la ecuación y añadimos la restricción al modelo.

```
Para cada tupla de estudiantes c de 𝒞 hacer
    Crear una expresión lineal vacía llamada restriccion
    Para cada equipo t de 𝒯 hacer
        Obtener el elemento del vector variables asociado al equipo t y
        asignarla a v
        Sumar v multiplicado por g_{t,c} a la expresión lineal restriccion
    Crear una restricción en el modelo donde la expresión lineal restriccion
    es igual a 1
```

Figura 12. Pseudocódigo de la creación de la primera restricción del modelo 1

De manera similar, creamos las subrestricciones correspondientes a la segunda restricción del modelo y las añadimos.

- Se crea una expresión lineal que representa la función objetivo. Para cada equipo en el conjunto T, se multiplica su variable correspondiente por el valor asociado y se suma a la expresión lineal. Luego, se establece esta expresión como la función objetivo del modelo con el objetivo de maximizar (Figura 13).

```
Crear una expresión lineal vacía llamada objetivo
Para cada equipo t de 𝒯 hacer
      Obtener el elemento del vector variables asociado al equipo t y
      asignarla a v
      Sumar v multiplicado por u_t a la expresión lineal objetivo
Establecer el objetivo del modelo como maximizar la expresión lineal
objetivo
```

Figura 13. Pseudocódigo de la creación de la función objetivo del modelo 1

- Se resuelve el modelo utilizando un solver.

- Dependiendo del resultado, se imprimen diferentes mensajes en la consola. Si se encuentra una solución óptima o factible, se imprimen los equipos seleccionados y el valor de la función objetivo.

6.5. Pseudo-código explicado de la generación de instancias.

Para la generación de instancias también se ha implementado un código. Vamos a explicar los pasos de este código.

- En primer lugar, creamos una lista vacía llamada belbin, leemos el fichero donde tenemos los resultados del cuestionario de Belbin y los añadimos a la lista vacía. De forma que cada elemento de la lista tendrá los resultados de cada alumno. A continuación, calculamos el tamaño de la lista, que será el número de alumnos disponibles (Figura 14).

```
Crear una lista vacía llamada belbin
Abrir el archivo CSV "student_data_belbin_mbti_anon.csv"
Para cada fila f del archivo hacer
    Agregar la fila f a la lista belbin
Asignar a la variable a la longitud de la lista belbin
```

Figura 14. Pseudo-código para importar los resultados del cuestionario

- Creamos una lista con los tamaños deseados para las instancias que vamos a generar.

- Por último, se generan las instancias. Para ello, utilizamos dos bucles anidados, para cada tamaño, realizo 10 veces el bucle principal. Cada una de las veces que se entre en el bucle, estaremos generando una instancia diferente. Para cada instancia, partimos de una lista que va desde 1 hasta la cantidad de alumnos que hay. Esta lista, la desordenamos y nos quedamos con los primeros elementos de la lista (tantos como el tamaño con el que está trabajando el bucle en ese momento). Para saber cuántos equipos necesito, como vamos a formar equipos de 5 alumnos, dividimos el tamaño entre 5. A continuación, se crea un archivo en el que se añade toda la información creada y se guarda en forma de archivo de Excel (Figura 15).

```
Para cada tamaño i en la lista tamaños hacer
    Para j = 1,10 hacer:

      Crear una lista números del 1 al a
      Mezclar aleatoriamente los elementos de la lista
      Seleccionar los primeros i elementos de la lista y asignarlos a
      la lista S
      Asignar a la variable tamañoequipos el valor 5

      Crear un nuevo archivo de Excel
      Añadir una hoja al archivo Excel y asignarle el nombre "Alumnos"
      Escribir "Tamaño equipos:" en la celda A1 y tamañoequipos en la
      celda B1
      Ordenar la lista S
      Insertar "Alumnos:" en la primera posición de la lista S
      Añadir la lista S como una fila en la hoja de Excel

      Guardar el archivo Excel
```

Figura 15. Pseudo-código del bucle principal de la generación de instancias

6.6. Pseudo-código explicado de los para la realización de los experimentos

Detallamos el pseudo-código usado para la obtención de los resultados de los experimentos planteados.

- Leemos el fichero con los resultados del cuestionario de Belbin y lo convertimos en un diccionario donde las claves son el alumno y el rol y el valor es 0 o 1 dependiendo de si el alumno tiene presente el rol (Figura 16).

```
Cargar el archivo CSV "student_data_belbin_mbti_anon.csv" y
guardarlo como un objeto llamado belbin
Definir una lista llamada umbrales que contiene los valores [12,
11, 14, 9, 10, 10, 13, 7]
Rellenar los valores faltantes del objeto belbin con 0

Dividir cada valor de belbin por su umbral correspondiente en la
lista umbrales
Convertir todos los valores en belbin a enteros
Convertir los valores no nulos en belbin a 1 y los valores nulos a 0
```

Figura 16. Pseudo-código para importar los resultados del cuestionario de belbin y adaptarlos

- Definimos los tres modelos. Omitimos los cuerpos porque los hemos explicado con anterioridad (Figura 17).

```
Definir la función modelo1:
    # Aquí va el pseudo-código para el modelo 1

Definir la función modelo2:
    # Aquí va el pseudo-código para el modelo 2

Definir la función modelo3:
    # Aquí va el pseudo-código para el modelo 3
```

Figura 17. Pseudo-código de la definición de los modelos

- A continuación, definimos una lista con los tamaños de las instancias que vamos a utilizar.

- Creamos un dataframe donde vamos a guardar los resultados de los experimentos (Figura 18).

```
Crear un DataFrame vacío llamado datos con las siguientes
columnas:
    - 'tamaño'
    - 'instancia'
    - 'modelo'
    - 'repeticion'
    - 'tiempo'
    - 'valor objetivo'
```

Figura 18. Pseudo-código de la creación de un objeto donde guardaremos los resultados

- En este momento, entramos en el bucle principal del código, este bucle va recorriendo cada instancia de las generadas y, de cada una de ellas, lee los conjuntos $\mathcal{S}$, $\mathcal{C}$, $\mathcal{N}$ y el número de equipos a formar. A continuación, se realizan los cálculos de los parámetros previamente descritos y que, por tanto, omitimos en esta subsección. Por último, entramos en un bucle ya que cada instancia vamos a resolverla 10 veces con cada modelo. En este último bucle, resolvemos la instancia con cada uno de los tres modelos y nos guardamos en el dataframe creado al principio el tamaño de la instancia, la instancia, el modelo, la repetición, el tiempo de ejecución y el valor objetivo (Figura 19).

```
Para instancia = 1,10 hacer
    Para cada tamaño alumnos de la lista tamaños hacer
        Cargar el archivo correspondiente a la instancia que vamos a usar
        Obtener el valor de ℒ[1] desde la hoja de cálculo

        Obtener los datos de la hoja de cálculo 'Alumnos'
        Crear una lista S con los datos obtenidos

        Inicializar el diccionario 𝒞 para los alumnos que deben ir juntos
        Si la hoja de cálculo 'Alumnos que deben ir juntos' existe entonces
            Obtener los datos de esa hoja y guardarlos en el diccionario 𝒞

        Inicializar el diccionario 𝒩 para los alumnos que no pueden ir juntos
        Si la hoja de cálculo 'Alumnos que no pueden ir juntos' existe entonces
        Obtener los datos de esa hoja y guardarlos en el diccionario 𝒩

        Calcular todos los demás parámetros (Subsección 6.3)

        Para repeticion = 1,10 hacer
            Calcular el tiempo de inicio
            Calcular el valor objetivo para el modelo 1
            Calcular el tiempo de fin
            Crear un DataFrame con los datos obtenidos y añadirlo a datos

            Calcular el tiempo de inicio
            Calcular el valor objetivo para el modelo 2
            Calcular el tiempo de fin
            Crear un DataFrame con los datos obtenidos y añadirlo a datos

            Calcular el tiempo de inicio
            Calcular el valor objetivo para el modelo 3
                    Calcular el tiempo de fin
                    Crear un DataFrame con los datos obtenidos y añadirlo
                    a datos
```

Figura 19. Pseudo-código del bucle principal para la resolución de las instancias

- Por último, guardamos el dataframe con todos los resultados en un fichero de Excel.

7. EXPERIMENTOS

En esta sección vamos a ver cuáles han sido los resultados obtenidos en los experimentos y su posterior análisis. Recordemos que el objetivo de estos experimentos es obtener los tiempos de ejecución de unas instancias determinadas de varios tamaños para cada uno de los modelos. La comparación de estos tiempos de ejecución nos permitirá encontrar el modelo más eficiente y, por tanto, este modelo servirá para, en un futuro, comparar heurísticas de evaluación en el aula.

Para el diseño de estos experimentos hemos creado un script que tiene integradas las 3 funciones correspondientes a los 3 modelos propuestos en este trabajo. A continuación, para cada una de las instancias, calcula los parámetros que necesitan los modelos y les aplica 10 veces las funciones que resuelven el problema de optimización guardándose el valor objetivo y el tiempo de ejecución.

Este script se ha ejecutado en una máquina remota 4xVCPU con 16GB de RAM. Para ello, creé una cuenta de github, que actuaba de repositorio para mi código y me permitía subir el código a la máquina remota. Después de intentar ejecutar el script con la máquina remota (aun aumentando la memoria RAM de 8 a 16 GB), esta no fue capaz de sacar resultados para el modelo 3 a partir del tamaño 40 ni de calcular algunos parámetros para instancias de tamaño 60.

7.1. Resultados

Como ya adelantábamos, para el modelo 3 solo hemos conseguido obtener resultados de los dos primeros tamaños. Además, los parámetros que no hemos podido calcular en tamaño 60 no nos han permitido obtener resultados para el modelo 1. En la Tabla 20 podemos ver los resultados que se obtienen sin ningún tipo de modificación. En esta tabla se puede comprobar que los 3 modelos, como ya sabíamos, son equivalentes, ya que obtienen siempre el mismo valor objetivo para una misma instancia.

En la Tabla 6, vemos los tiempos de ejecución de los distintos modelos para cada una de las 10 instancias y cada uno de los tamaños, estos tiempos son los tiempos medios al ejecutar el modelo 10 veces y, por tanto, esto nos da una mayor fiabilidad que si nos quedamos con una única repetición.

Sabemos que el valor objetivo no depende de los modelos ya que los tres modelos obtienen la solución óptima. No obstante, hemos podido ver que todos devolvían el mismo valor y por tanto se corroboraba el buen funcionamiento de los tres modelos. Por tanto, lo único que nos permitirá comparar modelos es el tiempo de ejecución. Ya hemos visto que este tiempo de ejecución en sí no es lo que nos importa ya que depende de la máquina en la que se ejecute, sino qué tiempos son mayores y cuáles son menores.

Una primera opción de análisis de los datos era el ANOVA. Sin embargo, este tipo de análisis tiene los siguientes supuestos:

- Independencia de observaciones.
- Normalidad de los datos.
- Homocedasticidad: Varianza constante de los residuos

		Modelo		
tamaño	instancia	1	2	3
20	1	0,831702	0,007285	10,70424
	2	0,726335	0,006812	7,371487
	3	0,822681	0,005913	6,67101
	4	0,806936	0,005782	7,161874
	5	0,833823	0,005928	8,461325
	6	0,79188	0,006896	8,956144
	7	0,809437	0,006104	7,745498
	8	0,834325	0,00582	9,757316
	9	0,852432	0,006118	10,28887
	10	0,785437	0,007888	9,891181
30	1	11,77857	0,009526	309,3148
	2	13,50771	0,014213	417,1819
	3	11,94228	0,012551	293,4672
	4	11,88005	0,014795	306,7775
	5	11,90559	0,012351	294,8911
	6	12,48521	0,011895	325,6146
	7	11,47452	0,010999	389,9454
	8	11,67077	0,017284	311,8722
	9	13,71835	0,00982	306,5684
	10	11,90816	0,01133	352,6092
40	1	74,29895	0,014185	
	2	85,5285	0,018865	
	3	71,61243	0,013389	
	4	71,81726	0,013695	
	5	74,17809	0,012673	
	6	91,69563	0,014058	
	7	91,4209	0,013428	
	8	72,68902	0,013531	
	9	70,29865	0,016033	
	10	88,23512	0,013398	
50	1	357,5372	0,025249	
	2	362,2441	0,024726	
	3	370,2269	0,034331	
	4	354,8606	0,027938	
	5	353,5939	0,022886	
	6	365,7661	0,023306	
	7	351,1236	0,020623	
	8	369,1176	0,025959	
	9	306,2276	0,028113	
	10	357,651	0,027802	
60	1		0,028702	
	2		0,02987	
	3		0,024801	
	4		0,033896	
	5		0,031474	
	6		0,030413	
	7		0,028307	
	8		0,025075	
	9		0,023973	
	10		0,032137	

Tabla 6. Resultados de los tiempos medios en segundos de las 10 repeticiones obtenidos del experimento

Nuestros datos siempre van a cumplir la independencia de observaciones ya que son el resultado de un diseño de experimentos balanceado en el que estamos controlando todas las variables que pueden influir en los resultados. Por otro lado, la normalidad de los datos y la homocedasticidad no se cumplen y, aunque podríamos intentar aplicar transformaciones a nuestros datos para lograr que se cumplan estos supuestos, puede que esto nos dificulte la interpretación de los resultados al modificar la naturaleza de la variable de estudio.

La prueba de Friedman (Friedman, 1937) es una prueba estadística no paramétrica utilizada para analizar datos de medidas repetidas. Se utiliza principalmente cuando no se cumplen los supuestos de normalidad y homogeneidad de varianzas, lo que la convierte en una alternativa sólida al ANOVA de medidas repetidas. El análisis de la varianza comprueba en qué medida difieren los valores medidos de la muestra dependiente. La prueba de Friedman, en cambio, utiliza rangos en lugar de los valores medidos reales. La gran ventaja de utilizar rangos es que, si no te fijas en la diferencia de medias, sino en la suma de rangos, no es necesario que los datos tengan una distribución normal. (Conover, 1999). Nuestra muestra es de medidas repetidas ya que, a la misma instancia, le aplicamos cada uno de los modelos, por tanto, es adecuado este tipo de análisis. Cuando tengamos más de 2 grupos para comparar, utilizaremos la prueba de Nemenyi para poder ver entre qué grupos las diferencias son significativas.

Debido a que para el modelo 3 solo tenemos resultados para los dos primeros tamaños, no podemos hacer un análisis directamente con el conjunto entero de datos. Ya que como veremos, el modelo 3 es el que más tiempo de ejecución tiene y si, en los tamaños más altos no está presente, podemos llegar a la falsa conclusión que el tiempo de ejecución no aumenta al aumentar el tamaño. Veamos una visión general de los datos. En el Gráfico 1, cada punto representa el tiempo medio para cada modelo y tamaño.

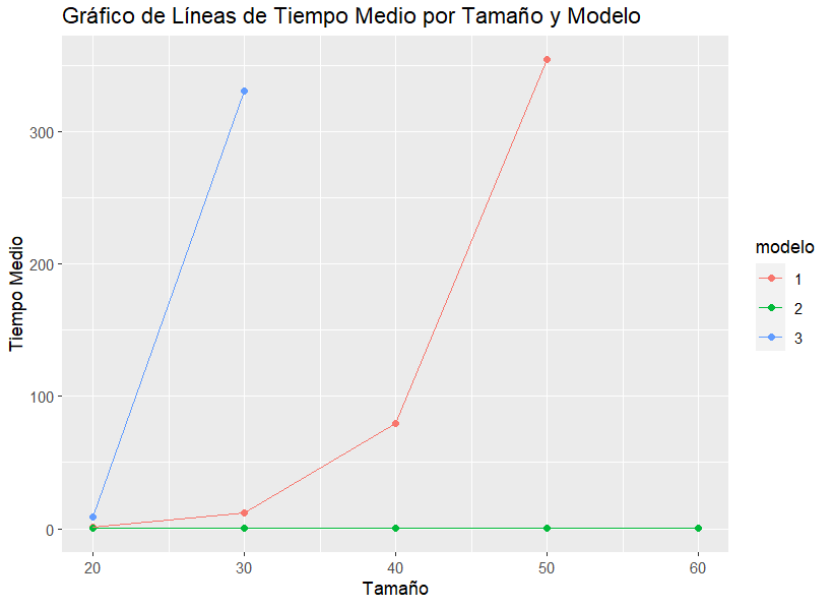

Gráfico 1. Tiempo medio de ejecución por tamaño y modelo

Además, la variable "valor objetivo" es una variable que no aporta nada a este análisis, ya que es una variable que dependerá del tamaño de la instancia y si la añadimos tendríamos correlación entre estas variables, lo cual no es bueno.

Veamos un primer análisis descriptivo del conjunto total de los datos. Nuestro conjunto de datos cuenta con 110 observaciones y 4 variables:

- **Tamaño**: Factor cuantitativo de 5 niveles. Tenemos 30 observaciones con tamaño 20 y con tamaño 30; 20 con tamaño 40 y 50; y únicamente 10 observaciones para el tamaño 60.
- **Instancia**: Variable que indica el individuo. Contamos con 50 individuos (instancias) diferentes.
- **Modelo**: Factor cualitativo de 3 niveles. Tenemos 40 observaciones para el modelo 1, 50 para el modelo 2 y 20 para el modelo 3
- **Tiempo**: Variable respuesta cuantitativa continua que va de 0,0058 a 417,1819. A priori, basándonos en el Gráfico 2 no parece mostrar una variable distribuida de forma normal ya que presenta una fuerte asimetría positiva.

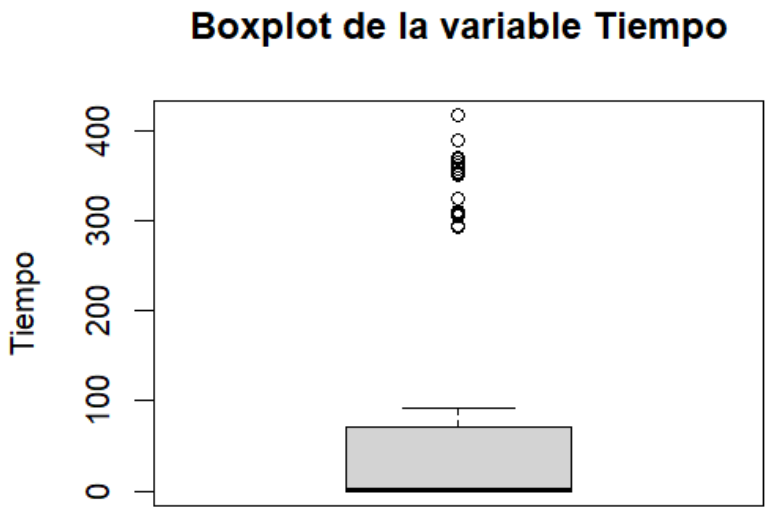

Gráfico 2. Diagrama de caja de la variable Tiempo

Vemos que la variable Tiempo no es normal y, por tanto, no podemos aplicar el ANOVA. Vamos a realizar una prueba de Friedman. No tenemos un diseño equilibrado, por lo tanto, no podemos realizar un análisis general con el conjunto global de los datos para poder comparar modelos, por tanto, realizaremos el estudio para cada tamaño por separado.

7.2. Análisis del porcentaje de instancias resueltas por modelo

Antes de proceder a estos análisis, vemos en la Tabla 7 qué porcentaje de las instancias han sido resueltas por cada uno de los 3 modelos, tanto a nivel general como a nivel de cada tamaño de aula.

		modelos		
		1	2	3
tamaños	20	100%	100%	100%
	30	100%	100%	100%
	40	100%	100%	0%
	50	100%	100%	0%
	60	0%	100%	0%
	TOTAL	80,00%	100,00%	40,00%

Tabla 7. Porcentaje de instancias resueltas por cada modelo

7.3. Análisis para aulas de 20 alumnos

En este caso, tenemos resultados disponibles para los 3 modelos. Nuestros datos constan de 3 variables:

- **Instancia**: Tenemos 10 individuos (instancias) diferentes
- **Modelo**: Tenemos 10 observaciones para cada uno de los 3 niveles modelos
- **Tiempo**: Variable respuesta cuantitativa continua que va de 0,0058 a 10,7042. Presenta asimetría positiva, no sigue una distribución normal (Gráfico 3).

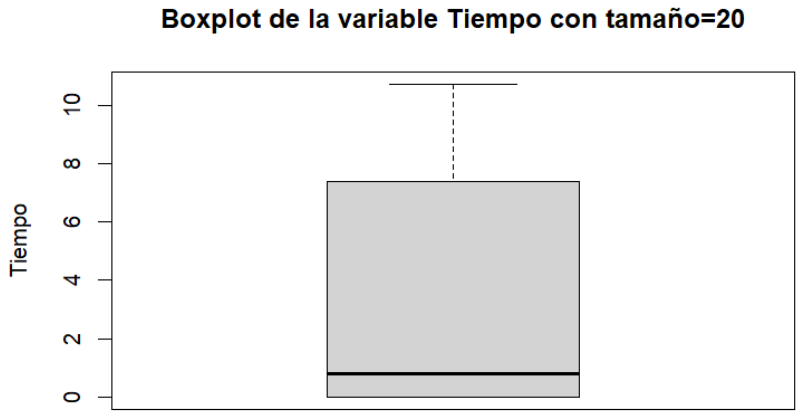

Gráfico 3. Diagrama de caja de la variable Tiempo

Lo que queremos comparar en este experimento es la eficiencia de los 3 modelos planteados. Estudiaremos las diferencias de Tiempos de ejecución entre grupos (los grupos son los diferenciados por los 3 modelos). Empezaremos haciendo un estudio únicamente sobre algunos gráficos y, más adelante, lo contrastaremos con una prueba de Friedman.

En el Gráfico 4 vemos el tiempo de ejecución medio según modelo en el gráfico. Vemos como cada el tiempo de ejecución del modelo 3 es el más elevado de los tres. Parece haber una diferencia entre los tiempos de ejecución cuando resolvemos el problema con el modelo 3 respecto a los otros dos modelos.

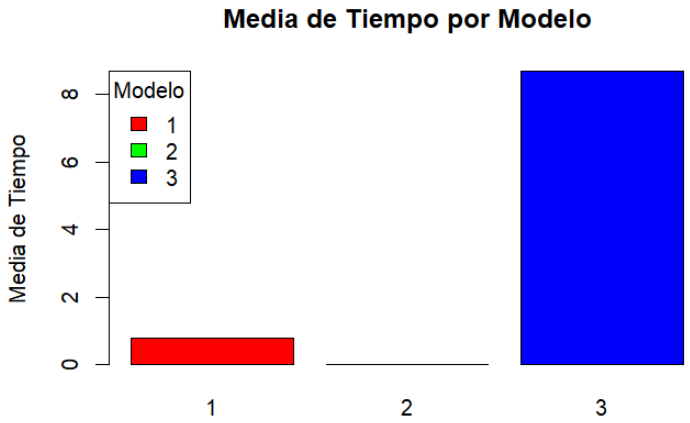

Gráfico 4. Tiempo medio de ejecución por modelo

Los diagramas "Box-plot" (Gráfico 5) nos ayudan a identificar posibles diferencias significativas, asimetrías, valores atípicos y homogeneidad de varianza entre los distintos niveles. Se puede acompañar a los gráficos con las medias y varianza de cada grupo (Tabla 8).

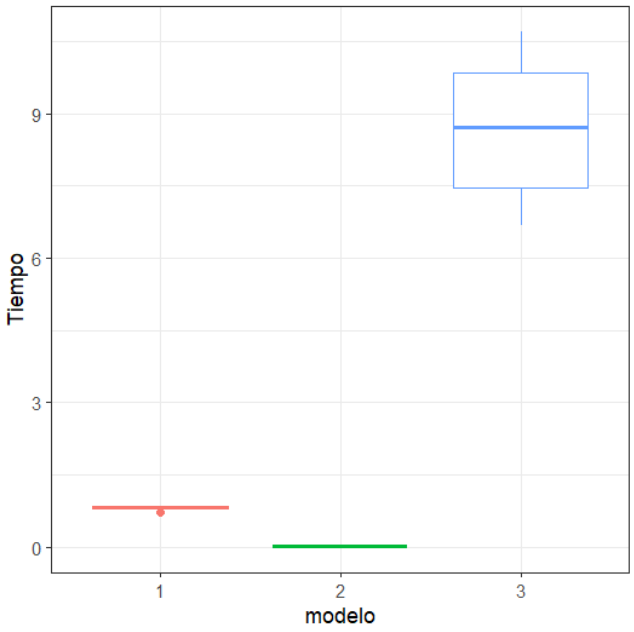

Gráfico 5. Diagrama de caja de la variable Tiempo por modelo

modelo	Media	Desviación
1	0,80949881	0,03579295
2	0,00645452	0,00072496
3	8,700895	1,428268

Tabla 8. Media y desviación típica por modelo

Debido a la diferencia de escala entre grupos, es difícil observar algunas de estas características. Si quisiésemos estudiar la asimetría por grupos, sería más adecuado hacerlo por separado. Sin embargo, podemos afirmar que no hay homogeneidad de

varianza entre los distintos niveles. Parece haber un valor extremo en el modelo 1. También se pueden intuir diferencias de medias entre grupos. El modelo 3 parece ser el modelo con mayor tiempo medio seguido del modelo 2. Más adelante comprobaremos si estas diferencias entre rangos son significativas.

Antes de realizar la prueba de Friedman, vamos a ver que no hay ningún dato anómalo que pueda estar invalidando el análisis. La prueba de Grubbs parte de la hipótesis nula de que no hay datos anómalos en el conjunto de datos. Cuando aplicamos esta prueba a nuestros datos, obtenemos un $p - valor = 0{,}8735$ y, por tanto, podemos aceptar que no hay datos anómalos.

Cuando aplicamos la prueba de Friedman, obtenemos un $p - valor = 0{,}0000454$ (Tabla 9). Por tanto, rechazamos la hipótesis nula de que los tres grupos tienen la misma media.

Estadístico (Chi-cuadrada de Friedman)	df	p-valor
20	2	0,0000454

Tabla 9. Resultados prueba de Friedman

La prueba de Nemenyi, también conocido como prueba post hoc de Nemenyi, es una prueba no paramétrica utilizada para comparar múltiples grupos después de haber realizado un análisis de varianza (ANOVA) o una prueba de Friedman. Esta prueba determina si hay diferencias significativas entre pares de grupos en un conjunto de datos. Los resultados de la prueba se muestran en la Tabla 10.

modelo	1	2
2	0,065	-
3	0,065	0,000023

Tabla 10. Resultados prueba de Nemenyi

Vemos que hay diferencias de rangos entre los niveles 2 y 3, es decir, entre el segundo y el tercer modelo. Por tanto, la diferencia que intuíamos entre el grupo modelo 3 y los demás no es del todo significativa ya que solo es significativa entre los modelos 2 y 3 con un nivel de 0,05. Sin embargo, las demás parejas de modelos tienen un p-valor muy cercano al 0,05. No podemos afirmar con un p-valor de 0,05 que el modelo 1 y el modelo 3 sean diferentes, pero esto se debe a que tenemos una muestra muy pequeña de únicamente 10 instancias. En el Gráfico 6, que representa el Tiempo por cada observación agrupadas por modelo, podemos visualizar esta diferencia de rangos.

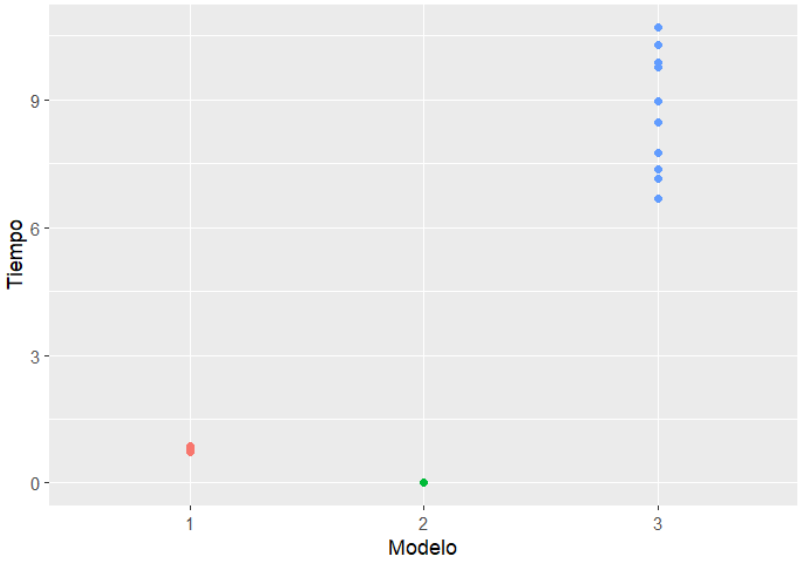

Gráfico 6. Tiempos obtenidos para cada observación agrupados por modelo

Si realizamos la prueba de Friedman con los modelos 1 y 2 estamos eliminando grados de libertad y esto permite que los resultados muestren diferencias significativas (Tabla 11).

Estadístico (Chi-cuadrada de Friedman)	df	p-valor
10	1	0,001565

Tabla 11. Resultados prueba de Friedman sin el modelo 3

Volvemos a visualizar el Tiempo por cada observación agrupadas por modelo, pero, en este caso, únicamente con los modelos 1 y 2 (Gráfico 7).

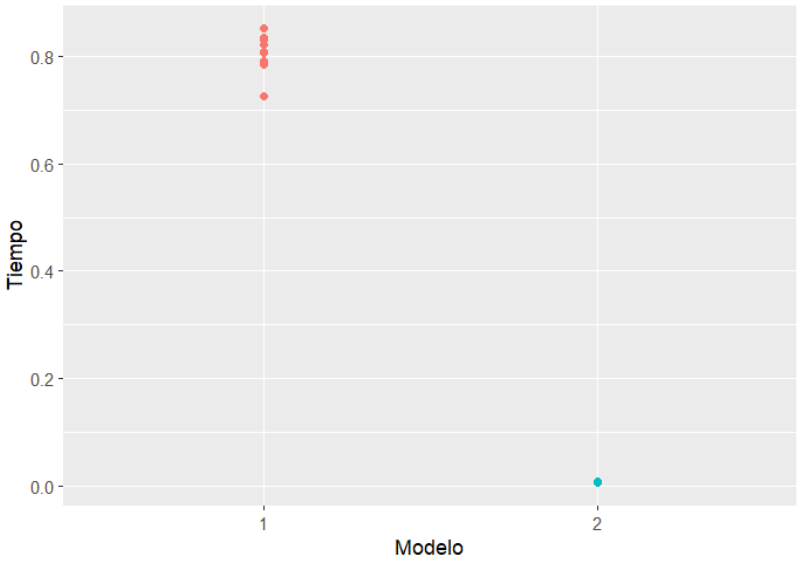

Gráfico 7. Tiempos obtenidos para cada observación agrupados por modelo

7.4. Análisis para aulas de 30 alumnos

En este caso, también tenemos resultados disponibles para los 3 modelos. Nuestros datos constan de 3 variables:

- **Instancia**: Contamos también con 10 individuos (instancias) diferentes

- **Modelo**: Tenemos 10 observaciones para cada modelo
- **Tiempo**: Variable respuesta cuantitativa continua que va de 0,0095 a 417,1819. A priori, el Gráfico 8 no muestra una variable distribuida de forma normal ya que también presenta una fuerte asimetría positiva.

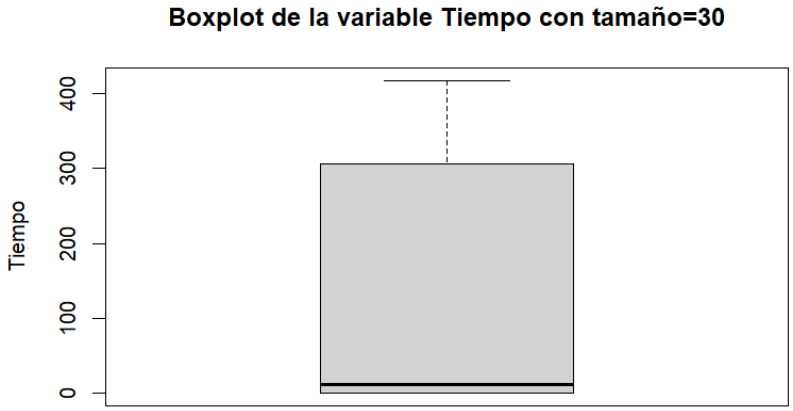

Gráfico 8. Diagrama de caja de la variable Tiempo

Nuevamente, en el Gráfico 9 vemos como cada el tiempo de ejecución del modelo 3 es el más elevado de los tres. Parece haber una diferencia entre los tiempos de ejecución cuando resolvemos el problema con el modelo 3 respecto a los otros dos modelos.

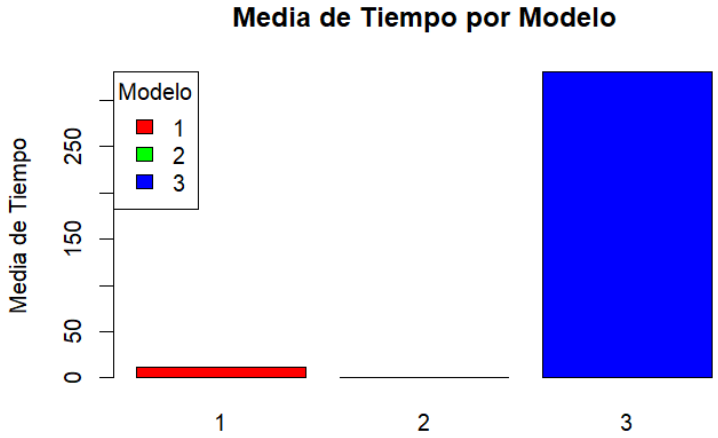

Gráfico 9. Tiempo medio de ejecución por modelo

Representamos de nuevo un diagrama "Box-plot" (Gráfico 10) que nos ayuda a identificar posibles diferencias significativas, asimetrías, valores atípicos y homogeneidad de varianza entre los distintos niveles. Lo acompañamos con las medias y varianza de cada grupo (Tabla 12).

Podemos afirmar que no hay homogeneidad de varianza entre los distintos niveles. Parece haber un valor extremo en el modelo 3. También se pueden intuir diferencias de rangos entre grupos. El modelo 3 parece ser el modelo con mayor tiempo medio seguido del modelo 2. Más adelante comprobaremos si estas diferencias entre rangos son significativas.

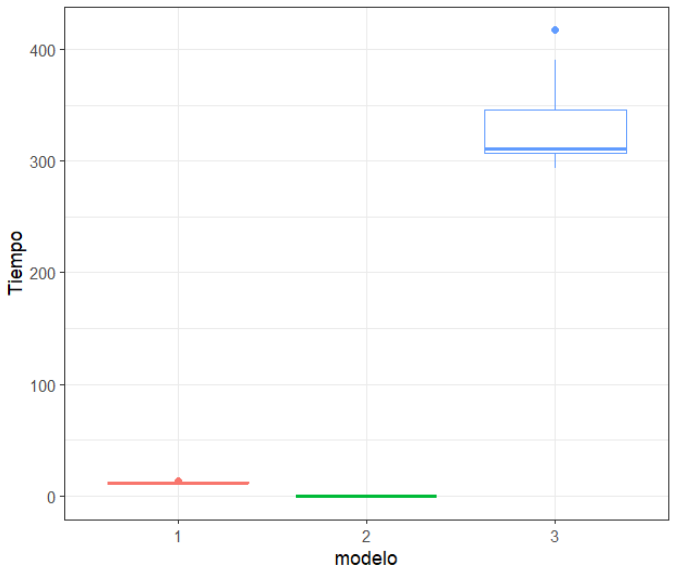

Gráfico 10. Diagrama de caja de la variable Tiempo por modelo

modelo	Media	Desviación
1	12,2271201	0,7756107
2	0,012476330	0,002383403
3	330,82423	42,33445

Tabla 12. Media y desviación típica por modelo

De nuevo, antes de aplicar la prueba de Friedman, comprobamos mediante la prueba de Grubbs que no hay datos anómalos. Obtenemos un $p - valor = 0{,}7294$ y, por tanto, podemos aceptar la hipótesis nula de que no hay datos anómalos.

Aplicamos la prueba de Friedman y de nuevo obtenemos diferencias significativas entre grupos con el mismo estadístico y p-valor que antes (Tabla 13).

Estadístico (Chi-cuadrada de Friedman)	df	p-valor
20	2	0,0000454

Tabla 13. Resultado prueba de Friedman

Al aplicar la prueba de Nemenyi para ver entre que grupos existen diferencias también obtenemos los mismos resultados que antes (Tabla 14).

modelo	1	2
2	0,065	-
3	0,065	0,000023

Tabla 14. Resultado prueba de Nemenyi

Tiene sentido que hayamos obtenido los mismos resultados que en el análisis anterior ya que en esta prueba no importa qué valor tenga la media para cada grupo, sino en qué lugar está cada grupo si ordenamos las medias obtenidas de mayor a menor.

Vemos que, con un nivel de significatividad de 0,05, hay diferencias entre los niveles 2 y 3, es decir, entre el segundo y el tercer modelo. Sin embargo, las demás parejas de modelos tienen un p-valor muy cercano al 0,05 y, pero que no llega a ser más pequeño por tener pocas muestras. En el Gráfico 11 podemos visualizar estas diferencias.

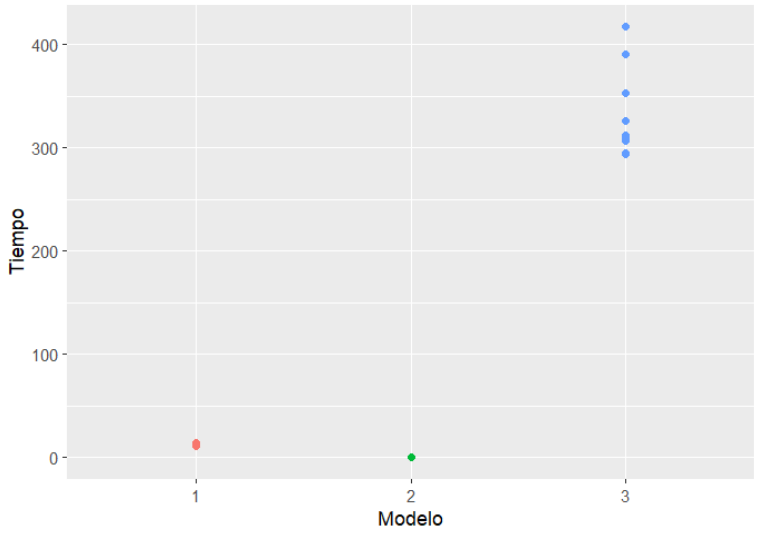

Gráfico 11. Tiempos obtenidos para cada observación agrupados por modelo

Igual que en el caso anterior, si realizamos la prueba de Friedman con los modelos 1 y 2 estamos eliminando grados de libertad y ya podemos ver diferencias significativas (Tabla 15).

Estadístico (Chi-cuadrada de Friedman)	df	p-valor
10	1	0,001565

Tabla 15. Resultado prueba de Friedman

En el Gráfico 12 vemos la diferencia entre rangos de los modelos 1 y 2.

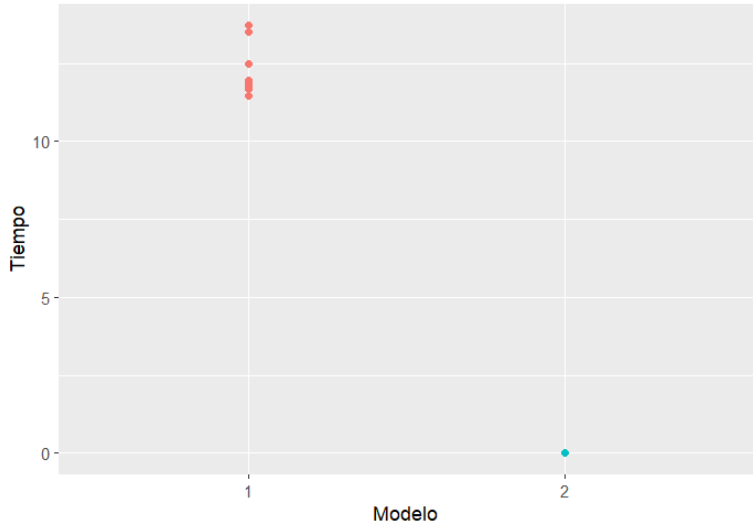

Gráfico 12. Tiempos obtenidos para cada observación agrupados por modelo

7.5. Análisis para aulas de 40 alumnos

A partir de aquí, ya no tenemos resultados para el modelo 3. De nuevo, nuestros datos constan de 3 variables:

- **Instancia:** Tenemos 10 individuos (instancias) diferentes.
- **Modelo**: Tenemos 10 observaciones para cada uno de los dos primeros modelos.
- **Tiempo**: Variable respuesta cuantitativa continua que va de 0,001267 a 91,669563. Vemos una ligera asimetría positiva (Gráfico 13).

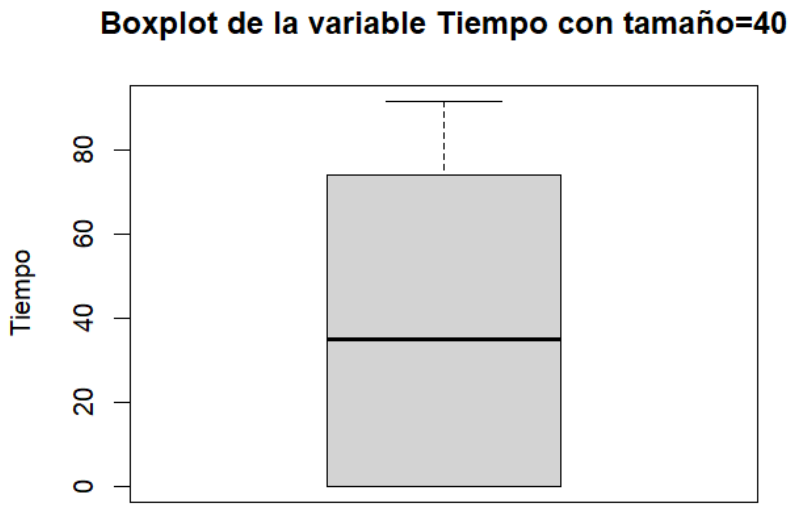

Gráfico 13. Diagrama de caja de la variable Tiempo

Del mismo modo que en los análisis previos, si calculamos el tiempo medio por modelo (Gráfico 14) vemos como parece haber un tiempo de ejecución mayor cuando resolvemos las instancias con el modelo 1.

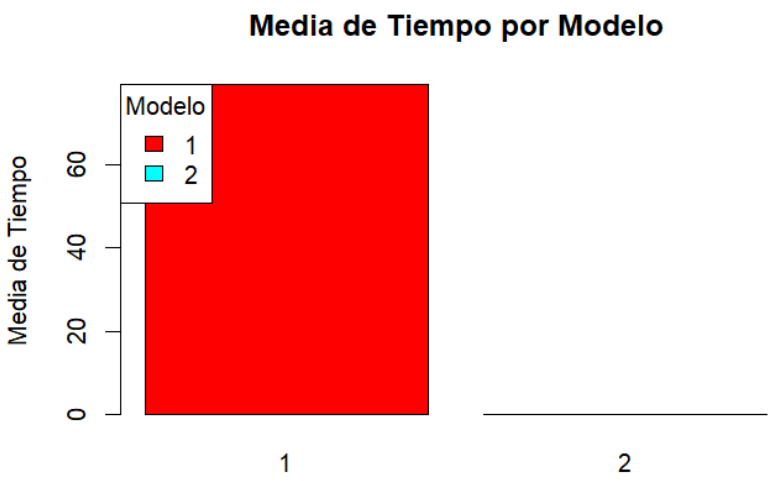

Gráfico 14. Tiempo medio de ejecución por modelo

De nuevo, el diagrama "Box-plot" (Gráfico 15) nos ayudan a identificar posibles diferencias significativas, asimetrías, valores atípicos y homogeneidad de varianza entre los distintos niveles. Lo acompañamos con las medias y varianza de cada grupo (Tabla 16).

De nuevo, podemos ver que no hay homogeneidad de varianza entre los distintos niveles. También se pueden intuir diferencias de medias entre grupos. El modelo 1 parece ser el modelo con mayor tiempo medio. Más adelante comprobaremos si las diferencias entre rangos son significativas.

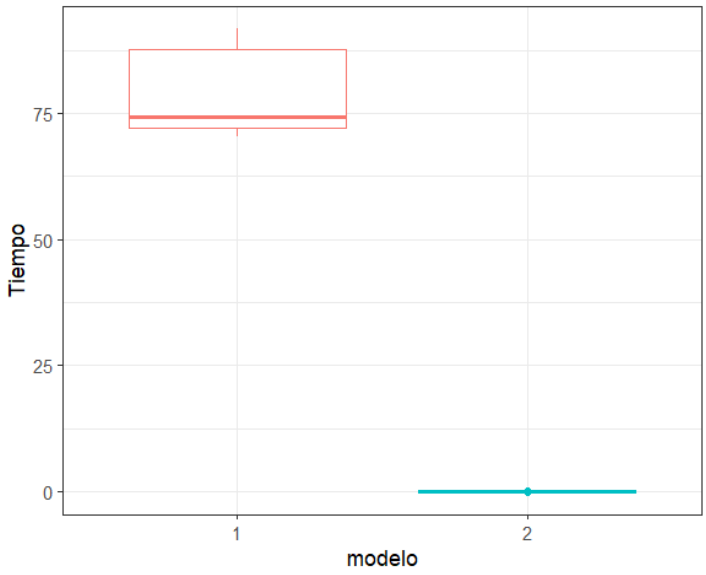

Gráfico 15. Diagrama de caja de la variable Tiempo por modelo

modelo	Media	Desviación
1	79,177454	8,882644
2	0,014325581	0,001823195

Tabla 16. Media y desviación típica por modelo

Antes de aplicar la prueba, vemos que no tenemos datos anómalos. Obtenemos un $p - valor = 1$ y, por tanto, podemos aceptar la hipótesis nula de que no hay datos anómalos.

En este caso, solamente con la prueba de Friedman (Tabla 17) tenemos suficiente para afirmar que hay diferencias significativas entre el modelo 1 y 2.

Estadístico (Chi-cuadrada de Friedman)	df	p-valor
10	1	0,001565

Tabla 17. Resultado prueba de Friedman

De forma gráfica (Gráfico 16), comprobamos que el modelo 1 tiene un rango de tiempo de ejecución mayor que el modelo 2.

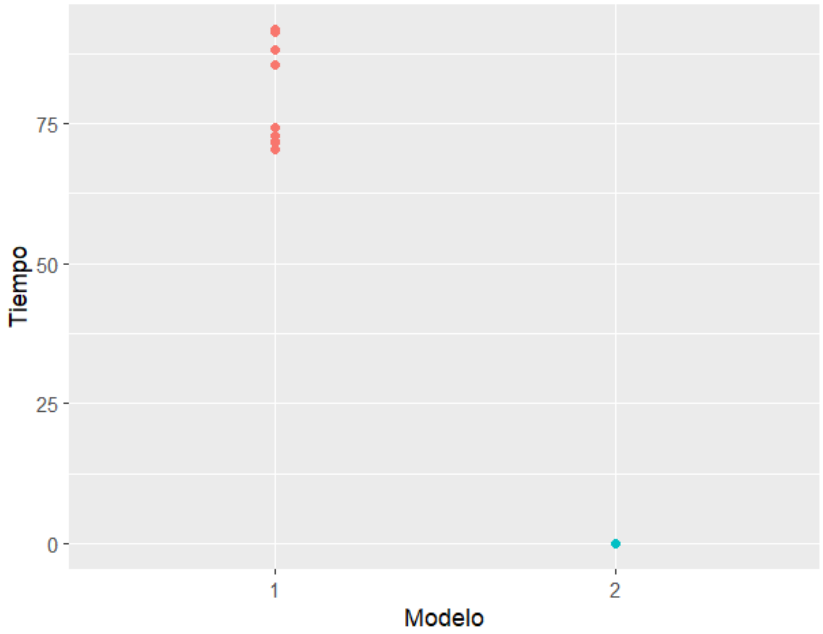

Gráfico 16. Tiempos obtenidos para cada observación agrupados por modelo

7.6. Análisis para aulas de 50 alumnos

Por último, vamos a analizar si hay diferencias entre grupos para el tamaño 50. En este caso, también tenemos resultados disponibles para los 2 modelos. Nuestros datos constan de las mismas 3 variables:

- **Instancia**: Tenemos 10 individuos (instancias)
- **Modelo**: Contamos con 10 observaciones para cada modelo
- **Tiempo**: Variable respuesta cuantitativa continua que va de 0,0206 a 370,2269. De nuevo, no parece que sigan una distribución normal (Gráfico 17).

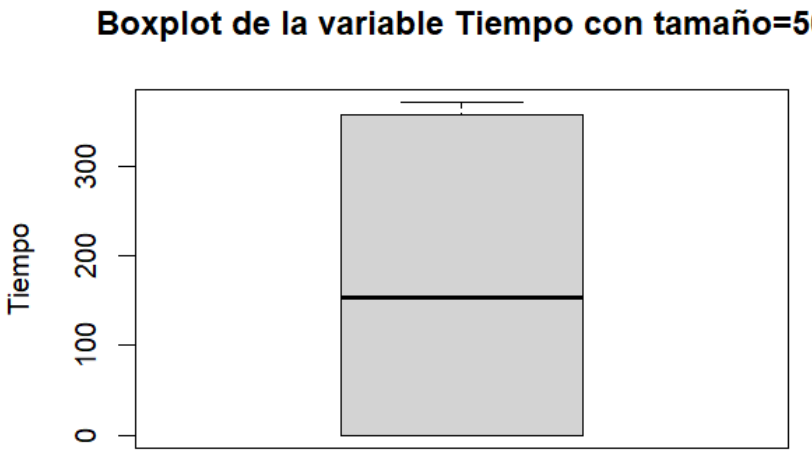

Gráfico 17. Diagrama de caja de la variable Tiempo

Si calculamos, una última vez, el tiempo medio por modelo (Gráfico 18) como parece haber un tiempo de ejecución mayor cuando resolvemos las instancias con el modelo 1.

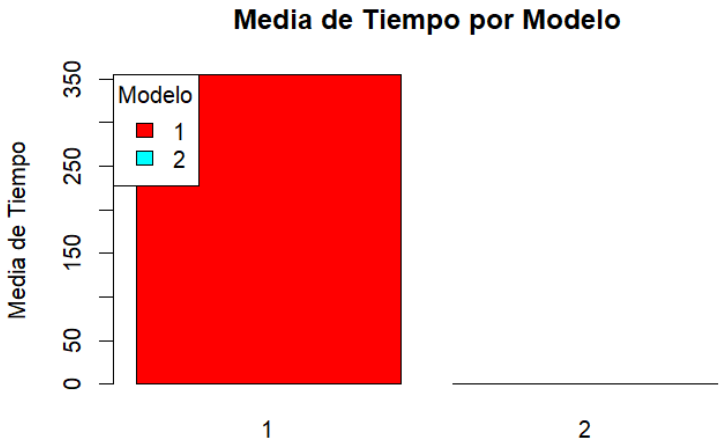

Gráfico 18. Tiempo medio de ejecución por modelo

El diagrama "Box-plot" (Gráfico 19) nos permite ver una marcada diferencia de medias entre grupos. Aunque la diferencia de varianzas no es muy clara en este gráfico, la varianza de cada grupo (Tabla 18) parece indicar que hay heterocedasticidad. Además, en el modelo 1 parece haber una observación extrema.

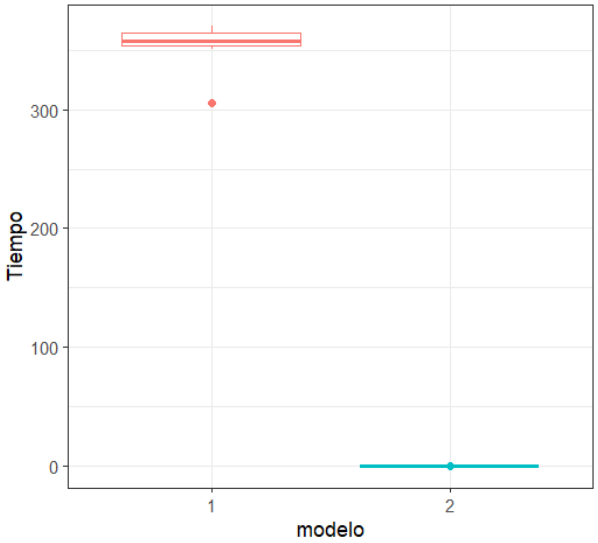

Gráfico 19. Diagrama de caja de la variable Tiempo por modelo

modelo	Media	Desviación
1	354,83486	18,28331
2	0,026093335	0,003783611

Tabla 18. Media y desviación por modelo

De nuevo, comprobamos que no tenemos datos anómalos con la prueba de Grubbs. Obtenemos un $p - valor = 1$ y, por tanto, podemos aceptar la hipótesis nula de que no hay datos anómalos.

Del mismo modo que en el caso anterior, nos basta con la prueba de Friedman (Tabla 19) para comprobar que hay diferencia significativa entre los dos modelos analizados.

Estadístico (Chi-cuadrada de Friedman)	df	p-valor
10	1	0,001565

Tabla 19. Resultado prueba de Friedman

Vemos gráficamente que el modelo 1 tiene un rango de tiempo de ejecución mayor que el modelo 2 (Gráfico 20).

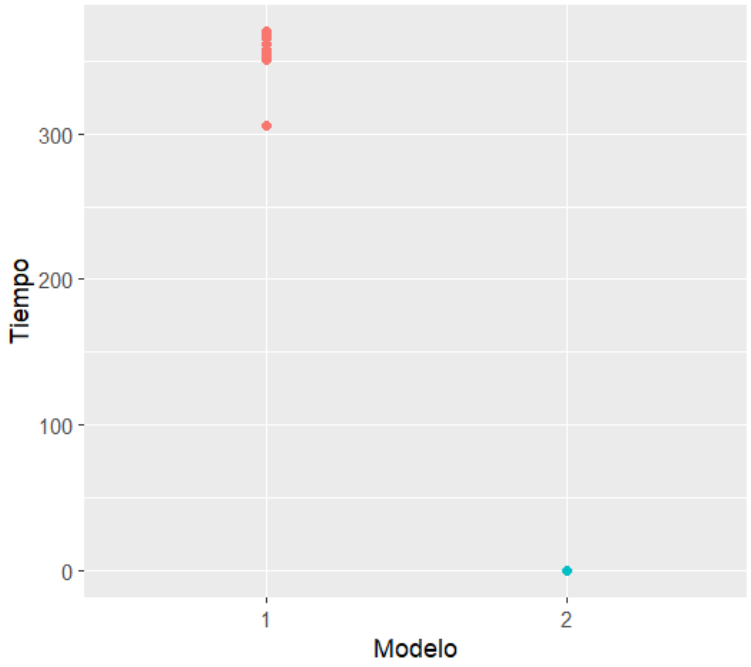

Gráfico 20. Tiempos obtenidos para cada observación agrupados por modelo

7.7. Discusión de los resultados

En esta sección vamos a analizar en profundidad los resultados obtenidos en el experimento planteado en este estudio.

En este estudio hemos establecido el objetivo de proponer, implementar y comparar tres modelos capaces de resolver el problema de la formación de equipos en el aula. La sección de los experimentos se relaciona con la última parte de los objetivos, es decir, se han realizado estos experimentos con el fin de poder comparar los tiempos de ejecución de los distintos modelos.

Para ello, hemos realizado un estudio en profundidad de los resultados obtenidos en el experimento. Hemos visto una visión general del desempeño de los modelos con la Tabla 7, que nos mostraba el porcentaje de las instancias resueltas por cada modelo a nivel de tamaño y a nivel global. Además, hemos comparado los diferentes rangos del tiempo de ejecución entre los modelos. Esto lo hemos podido hacer para cada uno de los tamaños planteados.

Por un lado, el desempeño global de los modelos nos permite concluir una diferencia entre el modelo 2, "modelo de asignación de tuplas a equipos", y los otros dos. Por un lado, el modelo 3, "modelo de asignación de posiciones a equipos" ha mostrado una limitación muy clara ya que ha podido resolver el 40% de las instancias. Por otro lado, vemos un ratio mucho mayor para el modelo 1, "modelo de elección de equipos", que ha conseguido resolver doble de las instancias. Sin embargo, no ha sido capaz de conseguir el 100% de instancias resueltas que ha conseguido el segundo modelo. El primer modelo está limitado por el cálculo de los parámetros, ya que al llegar al tamaño 60 nos ha resultado imposible calcularlos y esto no nos ha permitido evaluar si el modelo es capaz de resolver estas últimas instancias. Por tanto, tampoco lo hemos podido comparar con los demás modelos. Teniendo esto en cuenta, el modelo 2 ya parte de con ventaja.

Por otro lado, veamos las conclusiones que podemos sacar de las pruebas de rangos que hemos realizado. Estas pruebas las hemos aplicado a todos los tamaños para los cuales hemos obtenido resultados para más de un modelo (20, 30, 40 y 50) y hemos analizado si hay diferencias entre grupos (modelos) en el rango de la variable Tiempo. En todos los tamaños en los que estaba presente el modelo 3, hemos concluido que era el modelo con un tiempo de ejecución mayor. Además, estas diferencias de tiempos eran significativas en todos los casos. Por tanto, si consideramos esto junto con el hecho de haber podido resolver únicamente el 40% de las instancias, el modelo 3 es el peor de todos en términos de eficiencia. Esta diferencia abismal entre el modelo 3 y los otros 2 puede deberse a que es el modelo con más combinatoria. Por un lado, el modelo forma todos los equipos factibles, lo cual ya tiene una combinatoria grande, que también tiene el modelo 1. Sin embargo, en el modelo 3, se multiplica esta combinatoria, ya que, a los equipos escogidos para formar parte de la partición final de los alumnos, se les asigna una posición dentro del conjunto de equipos deseados. Por tanto, los parámetros que se precisan para el modelo son de tamaños elevados al estar asociados con estos equipos.

Hemos obtenido unos resultados mucho mejores para el modelo 1. Sin embargo, hemos visto un tiempo de ejecución mayor que en el modelo 2, esta diferencia podemos considerarla significativa para todos los tamaños. Además, también llegamos a la limitación causada por no poder calcular los parámetros de este modelo en el tamaño 60. Esto lo convierte en un modelo mejor que el 3 pero peor que el 2. La mejora de este modelo respecto al modelo 3 puede deberse, como ya hemos comentado, a que la combinatoria del modelo 1 es menor al no considerar la posición que ocupan los equipos escogidos.

Por último, en el modelo 2 hemos visto los menores tiempos de ejecución por varios ordenes de magnitud para todos los tamaños excepto para el tamaño 60 ya que no hemos tenido ningún otro modelo con el que compararlo. Además, los parámetros son menos costosos de calcular que en los demás modelos. Esto lo convierte en el mejor modelo. La diferencia a nivel de modelización entre este modelo y los anteriores que puede tener un gran impacto en el tiempo de ejecución de las instancias es la disminución de cálculos a realizar, ya que, en lugar de ir escogiendo sobre el total de equipos factibles, se van formando los equipos añadiendo las tuplas. De este modo, los parámetros necesarios para resolver el problema son mucho menores en este modelo.

Vemos, además, que las observaciones se distribuyen de forma muy similar en todos los tamaños, siendo el tiempo de ejecución cada vez mayor al aumentar el tamaño.

8. CONCLUSIONES

El proceso de la formación de equipos en el aula está experimentando un interés creciente debido a que el aprendizaje colaborativo ha demostrado tener numerosos beneficios frente al aprendizaje individual. De este modo, el problema de la formación de equipos trata de buscar las mejores combinaciones de alumnos en el aula para particionar la clase. El problema de optimización de equipos en el aula, trata de particionar los miembros de una clase buscando el valor óptimo de una función objetivo que depende de una heurística de evaluación. Esta heurística depende de atributos de los miembros. Sin embargo, es difícil de establecer qué criterios resultan en un buen desempeño del equipo en la tarea a la que se les asigna. Para poder realizar comparaciones entre distintas heurísticas de evaluación en el aula, se precisan algoritmos capaces de encontrar la solución óptima del problema.

En este trabajo, hemos revisado la literatura previa relacionada con el problema de la formación de equipos en el aula. De este modo, hemos podido comprobar que la falta de modelos exactos que puedan servir para comparar heurísticas de evaluación puede estar limitando el problema. Por ello, en este trabajo, hemos propuesto tres modelos capaces de encontrar la solución óptima. Estos modelos se han implementado en un entorno de Python. A continuación, para poder realizar experimentos que nos permitan evaluar la eficiencia a nivel de tiempos de ejecución de los modelos, necesitamos establecer un caso de uso. Para ello, se han establecido los roles de Belbin como atributos de los miembros de los que dependerá la heurística de evaluación de los equipos.

Los experimentos que hemos realizado consisten en la resolución de 50 instancias que hemos generado a partir de unos resultados reales obtenidos del cuestionario de Belbin realizado por alumnos universitarios. Estas instancias van desde tamaños de 20 alumnos hasta 50 alumnos. La resolución para cada una de estas instancias la hemos realizado con cada uno de los 3 modelos que hemos implementado. A partir de esto, hemos obtenido los tiempos de ejecución de cada modelo para cada instancia.

El análisis de los resultados obtenidos en el experimento ha concluido en una gran diferencia en tiempos de ejecución entre el modelo 2 y el resto. Siendo este, el "modelo de asignación de tuplas a equipos" el modelo con un tiempo de ejecución menor. En un grado menor en eficiencia se encuentra el modelo 1, "modelo de elección de equipos" aunque sigue siendo mucho mejor que el modelo 3, "modelo de asignación de posiciones a equipos".

Esto convierte al modelo 2, "modelo de asignación de tuplas a equipos", en la mejor opción para realizar experimentos que puedan comparar distintas heurísticas de evaluación en el aula. Las heurísticas con mejor desempeño podrán aplicarse a algoritmos no exactos, capaces de encontrar una buena solución en un tiempo de ejecución aceptable.

Con esto, hemos podido completar el objetivo de proponer, implementar y comparar distintos modelos capaces de resolver el problema de la formación de equipos en el aula. Además, hemos podido ver cuales son las limitaciones de estos modelos. El eterno problema de la optimización exacta es la limitación informática ya que, resolver problemas con tamaños muy grande resulta muy costoso computacionalmente y todavía no existen las herramientas necesarias para abordar en tiempos aceptables estos tipos de problemas. Sin embargo, el objetivo de este trabajo no era encontrar un modelo en el que podamos

resolver instancias muy grandes sino obtener el modelo más eficiente con el que podamos validar los criterios utilizados para la formación de equipos. Aun así, hemos tenido problemas a la hora de resolver algunas instancias con algunos modelos. Estos modelos están limitados por una parte por el cálculo de los parámetros, que en algunos casos pueden ser muy costosos de calcular. Por otro lado, aun pudiendo calcular estos parámetros, la combinatoria posible de formación de equipos supone también un límite para los modelos. Sin embargo, el modelo 2, "modelo de asignación de tuplas a equipos", es el que menos afectado está por estas limitaciones.

Como trabajo futuro podemos plantear realizar el mismo experimento añadiendo dependencias entre alumnos. En el planteamiento de los modelos hay dos parámetros que son un conjunto de alumnos que deben de ir juntos y otro conjunto de alumnos que no pueden ir juntos. Para ello, se ha llegado a crear un código (GENERACIÓN DE INSTANCIAS CON DEPENDENCIAS (CON LOS PARÁMETROS ALPHA Y BETA)) para la generación de las instancias con estas dependencias. En ellas tenemos los dos parámetros alpha y beta para calibrar el porcentaje de alumnos que debe haber en cada uno de estos conjuntos. Se podría plantear un nuevo experimento en el que cada uno de estos parámetros es un factor con tres niveles (0, 0.05 y 0,1). El experimento que se realizado en este trabajo correspondería a alpha=0 y beta=0. Adicionalmente, proponemos realizar otro experimento añadiendo estos dos factores (tendríamos que hacer las 9 combinaciones posibles para Alpha y beta) y tratarlos como factores con dos niveles.

Además, también podríamos intentar acelerar el tiempo de ejecución de alguno de los modelos. Sabiendo qué modelos funcionan mejor y basándonos en las posibles razones, podríamos realizar cambios en los modelos con la intención de aumentar su efectividad. Estos cambios podrían ser, por ejemplo, formulando alguna restricción de forma diferente. También podríamos añadir cotas a la mejor solución.

Bibliografía

Alberola, J. M., del Val, E., Sanchez-Anguix, V., Palomares, A., & Teruel, M. D. (2016). An artificial intelligence tool for heterogeneous team formation in the classroom. *Knowledge-Based Systems, 101*, 1-14.

Andrejczuk, E., Bistaffa, F., Blum, C., & Rodríguez-Aguilar, J. A. (2019). Synergistic Team Composition: A Computational Approach to Foster Diversity in Teams. *Knowledge-Based Systems, 182*.

Aranzabal, A., Epelde, E., & Artetxe, M. (2022). Team formation on the basis of Belbin's roles to enhance students' performance in project based learning. *Education for Chemical Engineers, 38*, 22-37.

Aritzeta, A., Swailes, S., & Senior, B. (2007). Belbin's Team Role Model: Development, Validity and Applications for Team Buiñding. *Journal of Management Studies*, 44(1):96-118.

Arrow, H., McGrath, J. E., & Berdahl, J. L. (2012). *Small Groups as Complex Systems: Formation, Coordination, Development, and Adaptation.* SAGE Publications, Inc.

belbin team roles test pdf. (2024). Obtenido de https://www.pdffiller.com/jsfiller-desk15/?flat_pdf_quality=low&requestHash=e292303ad4d6beedfd8402eb81ed223a811bd7cec312f6e916ab30e0694f8688&lang=en&projectId=1465547963&loader=tips&PAGE_REARRANGE_V2_MVP=true&richTextFormatting=true&isPageRearrangeV2MVP=t

Belbin, M. R. (1981). *Management Teams: Why They Succeed or Fail.*

Briggs Myers, I. (1998). *Introduction to Type: A Guide to Understanding Your results on the Myers-Briggs Type Indicator.* CPP, Inc.

Candel, G., Sánchez-Anguix, V., Alberola, J. M., Julián, V., & Botti, V. (2023). An Integer Linear Programming Model for Team Formation in the Classroom with Constraints. *Hybrid Artificial Intelligent Systems* (págs. 397-408). Salamanca: Springer, Cham.

Cattell, R. B. (1949). The Sixteen Personality Factor Questionnaire (16PF).

Chen, S. J., & Lin, L. (2004). Modeling Team Member Characteristics for the Formation of a Multifunctional Team in Concurrent Engineering. *IEEE transactions on Engineering Management, 51(2)*, 111-124.

Christodoulopoulos, C. E. (2007). A group formation tool in an e-learning context. *19th IEEE international conference on tools with artificial intelligence* (págs. 117-123). IEEE.

Conover, W. (1999). *DATAtab.* John Wiley & Sons.

Flores-Parra, J. M., Castañón-Puga, M., Evans, R. D., Rosales-Cisneros, R., & Gaxiola-Pacheco, C. (2018, June). Towards Team Formation Using Belbin Role Types and a Social Networks Analysis Approach. *2018 IEEE Technology and Engineering Management Conference (TEMSCON)* (págs. 1-6). IEEE.

Gómez-Zará, D., Dechurch, L. A., & Contractor, N. S. (2020). A Taxonomy of Team-Assembly Systems: Understanding How People Use Technologies to Form Teams. *Proc. ACM Hum.-Comput. Interact. 4, CSCW2*, 1-36.

Guimerà, R., Uzzi, B., Spiro, J., & Nunes Amaral, L. A. (2005). Team assembly mechanisms determine collaboration network structure and team performance. *Science*, 697-702.

John, O. P., Naumann, L. N., & Soto, C. J. (2008). The Big Five Inventory--Versions 4a and 54. *University of California, Berkeley, Institute of Personality and Social research*.

Johnson, D. W., & Johnson, R. T. (1996). Cooperation and the use of technology. *Hnadbook of Research for Educational Communications and Technology, 2*, 1017-1044.

Johnson, D. W., Johnson, R. T., & Smith, K. A. (2014). Cooperative Learning: Improving University Instruction by Basing Practice on Validated Theory. *Journal on Excellence in College Teaching, 25(3&4)*, 85-118.

Juárez, J. S. (2021). A comprehensive review and a taxonomy proposal of team formation problems. *ACM Computing Surveys (CSUR), 54 (7)*, 1-33.

Katzenbach, J. R., & Smith, D. K. (March-April de 1993). The Discipline of Teams. Harvard Business Review.

Katzenbach, J., & Smith, D. (2003). The wisdom of teams: Creating the high-performance organization. Harvard Business Press.

Ma, H., Li, J., Zhu, H., Tang, W., Huang, Z., & Tang, Y. (2022). Collaborative Optimization of Learning Team Formation Based on Multidimensional Characteristics and Constraints Modeling: A Team Leader-Centered Approach via E-CARGO. *IEEE Transactions on Computational Social Systems*.

Maqtary, N., Abdulqader, M., & Kamal, B. (2019). Group Formation Techniques in Computer-Supported Collaborative Learning: A Systematic Literature Review. *Technology, Knowledge and Learning, 24*, 169-190.

Odo, C. M. (June 25-29, 2019). Group formation for collaborative learning: A systematic literature review. *Artificial Intelligence in Education: 20th International Conference, AIED* (págs. 206-212). Chicago, IL, USA: Springer. International publishing.

Osborne, A., Fielder, S., Mcveigh-Schultz, J., Lang, T., Kreminsky, M., Butler, G., . . . Isbister, K. (2023). Being social in VR Meetings: A landscape analysis of Current Tools. *ACM Digital Library*, 1789-1809.

Revelo Sánchez, O., Collazos Ordóñez, C. A., Redondo Duque, M. Á., & Santana Pinto, I. I. (2021). Homogeneous Group Formation in Collaborative Learning Scenarios: An Approach Based on Personality Traits and Genetic Algorithms. *IEEE Transactions on Learning Technologies, 14(4)*, 486-499.

Salas, E., Burke, C. S., & Cannon-Bowers, J. A. (2003). Teamwork: Emerging principles. *International Journal of management reviews, 2, Issue 4*, 339-356.

Sanchez-Anguix, V., Alberola, J. M., Del Val, E., Palomares, A., & Teruel, M. D. (2023). Comparing computational algorithms for team formation in the classroom experience. *53*, 23883-23904.

Smith, M., Arthur, C., Hardy, J., & Callow, N. (2013). Transformational leadership and task cohesion in sport: The mediating role of intrateam communication. *Psychology of Sport and Exercise, 14(2)*, 249-257.

Ugarte, N., Aranzabal, A., Arruarte, A., & Larrañaga, M. (2022, October). Using the Behavioural Tendency of Students in a Team Environment for Team Formation. *2022 IEEE Frontiers in Education Conference (FIE)* (págs. 1-9). IEEE.

Yannibelli, V., & Amandi, A. (2012). A deterministic crowding evolutionary algorithm to form learning teams in a collaborative learning context. *Expert systems with applications, 39(10)*, 8584-8592.

Gurobi Optimization, LLC. (2024). GurobiPy. Recuperado de https://www.gurobi.com/documentation/11.0/quickstart_mac/the_gurobi_python_interfac.html

Gurobi Optimization, LLC. (2024). Gurobi Optimizer. Recuperado de https://www.gurobi.com/

Python. (2024). Python Programming Language. Python Software Foundation. Recuperado de https://www.python.org/

ANEXO A: Tabla de resultados

				Repetición									
Modelo	Instancia	Tamaño	Valor objetivo	1	2	3	4	5	6	7	8	9	10
1	1	20	31	0,852575	0,854419	0,826337	0,819831	0,843022	0,825739	0,81427	0,831597	0,829107	0,820127
		30	46	12,0792	11,77784	11,7432	11,63703	11,69127	11,7495	11,62448	11,85431	11,82709	11,80176
		40	63	75,99527	74,10717	74,255	74,02794	74,14886	74,21059	74,03433	73,89878	74,42357	73,88796
		50	77	375,7437	370,365	358,1165	355,8888	352,093	357,7861	351,4434	351,1761	351,7664	350,9924
	2	20	29	0,747069	0,714274	0,709252	0,746301	0,718974	0,713898	0,757777	0,716075	0,717633	0,722093
		30	45	13,62777	13,51085	13,62069	13,62934	13,38269	13,42327	13,53246	13,43047	13,48937	13,4302
		40	61	87,83281	85,45516	85,36848	84,63544	85,41324	85,81461	85,15106	84,95917	85,17145	85,48361
		50	79	403,0317	393,5163	354,8079	352,3715	355,7957	353,59	352,0924	352,4527	352,4806	352,3021
	3	20	29	0,826703	0,833435	0,830619	0,811185	0,837025	0,815783	0,805077	0,830795	0,802079	0,834106
		30	42	12,16156	11,87616	11,86345	12,00131	11,98163	12,00141	11,95297	11,89101	11,94371	11,74955
		40	64	72,42423	71,39321	71,57654	71,41122	71,31849	71,69662	71,61333	71,5023	71,55985	71,62846
		50	76	417,7591	375,8347	365,6242	363,0423	360,2876	361,9288	359,7482	368,2834	371,2922	358,4689
	4	20	32	0,827175	0,807238	0,804881	0,821111	0,797826	0,803908	0,799917	0,828106	0,786734	0,792465
		30	47	11,85932	11,92801	11,8339	11,84528	11,93618	11,81269	11,90332	11,91742	11,85305	11,91134
		40	60	72,3426	71,80361	71,62109	71,3816	71,53356	71,89061	71,84875	72,14728	71,77017	71,8333
		50	78	401,0804	354,5075	348,6757	348,3961	349,3618	348,5847	349,1757	348,361	348,9861	351,4773
	5	20	32	0,823645	0,845768	0,824646	0,820147	0,848622	0,826427	0,836434	0,85794	0,827085	0,827515
		30	48	12,09818	11,93012	11,91137	11,89396	11,85984	11,78983	11,84059	11,90472	11,9044	11,92291
		40	60	75,10828	74,2675	74,33226	73,97032	73,9614	73,94731	74,09464	73,9182	73,91221	74,26876
		50	73	397,4292	355,0922	347,3808	350,9932	347,0453	345,4317	349,3683	347,4311	347,8966	347,8706
	6	20	31	0,822993	0,778125	0,804055	0,801657	0,80422	0,781325	0,770258	0,803215	0,779756	0,773196
		30	44	12,51629	12,39847	12,36593	12,46317	12,41766	12,31859	12,54371	12,57059	12,28938	12,96827
		40	61	92,81849	91,88803	91,29874	91,30598	91,74955	91,54335	91,70844	91,49731	91,30412	91,84234
		50	75	400,6435	367,0672	360,8674	361,8716	361,6107	360,9338	361,6882	361,3607	361,1859	360,4321
	7	20	32	0,790257	0,834235	0,80478	0,804979	0,832147	0,802871	0,804616	0,825371	0,797369	0,797744
		30	47	11,64195	11,51294	11,38656	11,32167	11,38984	11,50599	11,43268	11,52363	11,544	11,4859
		40	64	92,20084	91,34786	91,31098	91,83472	91,36583	91,05376	91,16027	91,29983	90,9611	91,67377
		50	76	388,1753	352,3221	348,2633	344,2119	346,446	347,0398	346,0425	346,761	346,6659	345,3083
	8	20	31	0,857788	0,828529	0,834701	0,846691	0,820746	0,808596	0,827783	0,842505	0,823067	0,852845
		30	48	11,78458	11,67056	11,75438	11,63425	11,6426	11,60768	11,65414	11,57748	11,69902	11,68297
		40	60	73,17016	72,33273	72,45985	72,84854	72,80532	72,84867	72,69376	72,78358	72,38854	72,55903
		50	79	420,2025	383,2187	361,2098	359,5488	361,0311	363,0289	360,537	361,3352	360,093	360,9705
	9	20	28	0,835197	0,849674	0,829446	0,839862	0,867843	0,869814	0,839255	0,88097	0,858925	0,853329
		30	46	13,99717	13,71061	13,75351	13,57075	13,75311	13,68012	13,71489	13,70523	13,63019	13,66796
		40	61	70,97989	69,92044	70,3904	70,33898	70,28641	70,14196	70,01848	70,13164	70,40837	70,36999
		50	78	354,4662	336,8497	301,9284	294,6636	295,9065	295,8508	296,2482	295,5572	295,6496	295,1558
	10	20	30	0,802876	0,770119	0,77529	0,804524	0,782559	0,800711	0,797304	0,775636	0,771827	0,773526
		30	46	12,03181	11,86779	11,84657	11,9351	11,82169	11,95791	11,88628	11,92339	11,85025	11,96083
		40	64	89,00685	88,33358	88,02438	88,42848	87,99976	88,10726	88,23017	88,13681	87,91575	88,16811
		50	77	403,5734	366,3854	350,692	350,4562	350,4897	351,6916	351,088	350,9943	350,5499	350,5891
2	1	20	31	0,007106	0,00599	0,005747	0,005933	0,010063	0,006028	0,006003	0,014057	0,006027	0,005897

		30	46	0,009108	0,009055	0,008894	0,009156	0,013609	0,009417	0,008824	0,008897	0,009228	0,009069
		40	63	0,014399	0,013734	0,014248	0,014088	0,014608	0,013951	0,013875	0,014327	0,014442	0,01418
		50	77	0,06721	0,034446	0,018817	0,018172	0,019463	0,018533	0,018831	0,019535	0,018272	0,019208
		60	92	0,035271	0,029704	0,02784	0,02758	0,027728	0,027256	0,027356	0,027539	0,028726	0,028023
	2	20	29	0,009241	0,011062	0,005936	0,005864	0,006002	0,005922	0,005928	0,005999	0,006083	0,006078
		30	45	0,00929	0,014442	0,014034	0,019567	0,014444	0,017513	0,011539	0,011657	0,009606	0,020035
		40	61	0,019872	0,018877	0,017796	0,018085	0,022818	0,018191	0,018364	0,018219	0,018345	0,018084
		50	79	0,060571	0,018118	0,017503	0,018049	0,041869	0,01787	0,018377	0,018092	0,018128	0,018684
		60	94	0,029684	0,029951	0,030305	0,029705	0,029734	0,029781	0,029876	0,030234	0,029615	0,029812
	3	20	29	0,006092	0,005846	0,00594	0,005916	0,006115	0,005875	0,005885	0,006035	0,005748	0,005674
		30	42	0,01451	0,011964	0,012195	0,016842	0,012889	0,009602	0,009127	0,014203	0,009514	0,014661
		40	64	0,013147	0,013328	0,013311	0,013478	0,013336	0,013468	0,013371	0,013864	0,013329	0,013259
		50	76	0,110556	0,019356	0,033599	0,019412	0,019609	0,019725	0,037273	0,036712	0,021051	0,026019
		60	91	0,024604	0,024556	0,024891	0,024931	0,025017	0,025012	0,024629	0,024905	0,024384	0,025076
	4	20	32	0,005743	0,005684	0,005753	0,005786	0,005901	0,005908	0,005639	0,005892	0,00584	0,005676
		30	47	0,014872	0,014983	0,014612	0,015593	0,014815	0,01475	0,014549	0,014604	0,014494	0,01468
		40	60	0,01365	0,013692	0,013794	0,013867	0,013369	0,013988	0,013846	0,01367	0,013885	0,013185
		50	78	0,045128	0,057406	0,021759	0,021636	0,021832	0,021457	0,022404	0,022987	0,022494	0,02228
		60	92	0,031239	0,03199	0,033226	0,032944	0,031246	0,031057	0,053689	0,031106	0,030814	0,031651
	5	20	32	0,005735	0,00594	0,005715	0,00595	0,005716	0,00593	0,006238	0,006129	0,006043	0,005882
		30	48	0,014943	0,009474	0,009252	0,010215	0,01474	0,009547	0,009578	0,015065	0,014981	0,015716
		40	60	0,012316	0,012629	0,012415	0,01278	0,012653	0,012969	0,013202	0,012678	0,012841	0,012244
		50	73	0,043767	0,018791	0,018639	0,038787	0,018345	0,01807	0,018326	0,018022	0,018284	0,017833
		60	94	0,031475	0,031468	0,031395	0,031217	0,03138	0,031489	0,031632	0,031662	0,031549	0,031468
	6	20	31	0,010427	0,006137	0,006257	0,005907	0,006099	0,00601	0,010156	0,006139	0,005896	0,005933
		30	44	0,01436	0,008807	0,014519	0,008866	0,014909	0,009574	0,014623	0,009056	0,015375	0,008857
		40	61	0,014213	0,014303	0,013658	0,013854	0,014261	0,013969	0,014279	0,014191	0,014278	0,013575
		50	75	0,0532	0,019011	0,027027	0,017745	0,018926	0,023995	0,018303	0,018433	0,018344	0,018074
		60	94	0,03004	0,030324	0,030613	0,030214	0,030629	0,030843	0,030199	0,030366	0,030691	0,03021
	7	20	32	0,005984	0,006169	0,006057	0,006042	0,006124	0,006011	0,00622	0,006185	0,006031	0,006215
		30	47	0,031507	0,00884	0,00845	0,008309	0,010756	0,008135	0,008555	0,008124	0,008748	0,008571
		40	64	0,013097	0,013237	0,013499	0,01309	0,013349	0,013606	0,01338	0,013864	0,013779	0,01338
		50	76	0,03882	0,019565	0,018725	0,019066	0,018931	0,017855	0,018042	0,020215	0,017559	0,017453
		60	94	0,028715	0,028198	0,028612	0,02839	0,028201	0,028208	0,028145	0,028448	0,028087	0,028061
	8	20	31	0,005982	0,005858	0,005767	0,005696	0,005659	0,005616	0,005619	0,00588	0,005884	0,006236
		30	48	0,015501	0,020253	0,015361	0,017819	0,017647	0,015513	0,018236	0,018449	0,017856	0,016207
		40	60	0,013614	0,01366	0,013514	0,013785	0,013558	0,013483	0,013342	0,013626	0,01298	0,013752
		50	79	0,083423	0,019658	0,019758	0,01963	0,019179	0,019188	0,019812	0,019688	0,019611	0,019642
		60	94	0,024824	0,025276	0,025087	0,024412	0,02517	0,025043	0,025347	0,025054	0,025339	0,0252
	9	20	28	0,006194	0,006057	0,00624	0,005895	0,005839	0,006114	0,006075	0,006429	0,006055	0,006284
		30	46	0,00831	0,014152	0,008322	0,014045	0,008193	0,008278	0,008657	0,008326	0,011379	0,008534
		40	61	0,014311	0,014184	0,013858	0,014373	0,013973	0,035877	0,013545	0,013349	0,013467	0,013397
		50	78	0,091094	0,037497	0,023543	0,018028	0,018198	0,018067	0,01856	0,018601	0,019515	0,018028
		60	92	0,023973	0,023945	0,023744	0,023664	0,023533	0,023609	0,024391	0,025427	0,023783	0,023667
	10	20	30	0,006201	0,006825	0,006246	0,01388	0,006073	0,00638	0,012995	0,006391	0,007648	0,006242
		30	46	0,013622	0,011605	0,013738	0,008101	0,013716	0,00805	0,014228	0,007936	0,014033	0,008273
		40	64	0,013076	0,013284	0,013265	0,013645	0,01379	0,013098	0,013831	0,013829	0,013321	0,012844

		50	77	0,092933	0,023744	0,020362	0,019724	0,019898	0,020189	0,020939	0,020257	0,020278	0,019694
		60	94	0,032123	0,032126	0,032713	0,031759	0,031454	0,032026	0,031845	0,032313	0,032336	0,032679
3	1	20	31	10,941	10,727	10,64256	10,53156	10,73642	10,61834	10,65113	10,83166	10,67336	10,68937
		30	46	319,8292	305,0151	305,5204	315,9168	304,4581	308,3977	317,5384	304,661	306,8016	305,0096
	2	20	29	7,362699	7,373881	7,307029	7,440883	7,408147	7,393358	7,38214	7,356869	7,34674	7,343127
		30	45	414,5975	416,8742	416,0145	424,4361	418,5232	416,9463	415,9395	414,5854	416,69	417,2125
	3	20	29	6,67626	6,74535	6,683759	6,625553	6,681405	6,717405	6,651873	6,667827	6,632275	6,628394
		30	42	292,2569	295,1378	293,4052	293,1259	292,9084	292,9786	294,4494	291,8333	293,9826	294,5939
	4	20	32	7,205991	7,127556	7,150214	7,103557	7,202576	7,210714	7,110768	7,162321	7,18037	7,164676
		30	47	308,1475	305,7988	305,2846	304,4637	310,3121	304,9572	310,6157	307,0938	305,1728	305,9288
	5	20	32	8,514188	8,504035	8,444084	8,455568	8,50499	8,473307	8,490389	8,411565	8,431678	8,383449
		30	48	293,2393	293,5635	294,7156	296,0748	294,1751	295,1396	295,1103	293,6239	298,0863	295,1827
	6	20	31	8,965183	8,970184	8,90682	8,912297	8,952884	8,919307	8,918171	8,986866	9,039052	8,990677
		30	44	324,3993	320,7388	332,0385	325,7483	326,6824	324,0996	326,7394	325,6223	325,0209	325,056
	7	20	32	7,775803	7,805946	7,700268	7,737578	7,705621	7,683118	7,761112	7,748434	7,74105	7,796048
		30	47	388,5701	388,4994	389,4146	389,7828	391,4189	390,2416	390,7516	391,0889	390,0251	389,661
	8	20	31	9,645953	9,825347	9,70123	9,753314	9,679476	9,731206	9,764293	9,704749	9,773141	9,994452
		30	48	311,3144	310,9854	311,0583	320,1465	316,8677	312,0142	311,926	306,8423	307,2028	310,3643
	9	20	28	10,2401	10,30839	10,26792	10,20593	10,40964	10,31328	10,29734	10,29472	10,28194	10,26944
		30	46	306,7684	305,8147	306,6055	308,1513	306,066	303,9744	307,2403	307,3021	306,5309	307,2308
	10	20	30	9,988881	9,912456	9,946386	10,10675	9,941272	9,831378	9,789135	9,827816	9,804465	9,763275
		30	46	362,594	355,7319	349,5597	352,0268	351,5065	349,775	351,3211	350,4592	352,6406	350,4772

Tabla 20. Resultados completos de los experimentos

ANEXO B: CÓDIGOS

GENERACIÓN DE INSTANCIAS SIN DEPENDENCIAS

```
# import csv file
import csv
belbin = []
with open("C:/Users/annat/Downloads/student_data_belbin_mbti_anon.csv",
newline='') as File:
    reader = csv.DictReader(File,delimiter=';', restval='0')
    for row in reader:
        belbin.append(row)

from operator import length_hint
a = length_hint(belbin)

tamaños = [20,30,40,50,60]

import numpy as np
import random as rd
import openpyxl as opx

for i in tamaños:
    for j in range(1,11):
        lista=list(range(1,a+1))
        rd.shuffle(lista)
        S = lista[0:i]
        TamañoEquipos=5

        wb = opx.Workbook()
        hoja = wb.active
        hoja.title = "Alumnos"
        hoja['A1']="Tamaño equipos:"
        hoja['B1']= TamañoEquipos
        S.sort()
        S.insert(0,"Alumnos:")
        hoja.append(S)
        wb.save("instancia"+str(i)+"_"+str(j)+".xlsx")
```

GENERACIÓN DE INSTANCIAS CON DEPENDENCIAS (CON LOS PARÁMETROS ALPHA Y BETA)

```
# import csv file
import csv
belbin = []
with open("C:/Users/annat/Downloads/student_data_belbin_mbti_anon.csv",
newline='') as File:
    reader = csv.DictReader(File,delimiter=';', restval='0')
    for row in reader:
        belbin.append(row)

from operator import length_hint
a = length_hint(belbin)

tamaños = [20,30,40,50,60]
alpha = 0.3
beta = 0.2

import numpy as np
import random as rd
import openpyxl as opx
import math
for i in tamaños:
    for j in range(1,11):
        print(j)
        lista=list(range(1,a+1))
        rd.shuffle(lista)
        S = lista[0:i]
        TamañoEquipos=5

        wb = opx.Workbook()
        hoja = wb.active
        hoja.title = "Alumnos"
        hoja['A1']="Tamaño equipos:"
        hoja['B1']= TamañoEquipos
        hoja2 = wb.create_sheet("Alumnos que deben ir juntos")
        C={}
        J = S[0:math.ceil(alpha*i)]
        if len(J)%2==0:
            for k in range(int(len(J)/2)):
                B = [J[2*k],J[2*k+1]]
                C[k]=(J[2*k],J[2*k+1])
                B.insert(0,"Grupo"+str(k+1))
                hoja2.append(B)
        else:
            for k in range(int(len(J)/2)):
                if k < int(len(J)/2)-1:
                    B = [J[2*k],J[2*k+1]]
                    C[k]=(J[2*k],J[2*k+1])
                    B.insert(0,"Grupo"+str(k+1))
                    hoja2.append(B)
```

```
        else:
            B = [J[2*k],J[2*k+1],J[2*k+2]]
            C[k]=(J[2*k],J[2*k+1],J[2*k+2])
            B.insert(0,"Grupo"+str(k+1))
            hoja2.append(B)

hoja3 = wb.create_sheet("Alumnos que no pueden ir juntos")
cantidad = math.ceil(beta*i)
N={}
D=[]
if int(cantidad/2)%2==0:
    for k in range(int(cantidad/2)+1):
        B=[]

        while True:
            l = rd.choice(list(set(S)-set(D)))
            n = rd.choice(list(set(S)-set(D)-{l}))

            if all(l not in C[m] or n not in C[m] for m in C):
                if all(l not in N[m] or n not in N[m] for m in N):
                    N[k]=(l,n)
                    B = ["Grupo" + str(k+1), l, n]
                    hoja3.append(B)
                    D.append(l)
                    D.append(n)
                break
else:
    for k in range(int(cantidad/2)):
        B=[]

        while True:
            l = rd.choice(list(set(S)-set(D)))
            n = rd.choice(list(set(S)-set(D)-{l}))

            if all(l not in C[m] or n not in C[m] for m in C):
                if all(l not in N[m] or n not in N[m] for m in N):
                    N[k]=(l,n)
                    B = ["Grupo" + str(k+1), l, n]
                    hoja3.append(B)
                    D.append(l)
                    D.append(n)
                break
    while True:
            l = rd.choice(D)
            n = rd.choice(list(set(S)-set(D)))

            if all(l not in C[m] or n not in C[m] for m in C):
                if all(l not in N[m] or n not in N[m] for m in N):
                    N[k+1]=(l,n)
                    B = ["Grupo" + str(k+2), l, n]
                    hoja3.append(B)
                break
S.sort()
```

```
        S.insert(0,"Alumnos:")
        hoja.append(S)
        wb.save("C:/Users/annat/OneDrive/Documents/MASTER/TFM/Modelos/instancia_C
_N"+str(i)+"_"+str(j)+"_"+str(alpha)+"_"+str(beta)+".xlsx")
```

SCRIPT EXPERIMENTO

```
from itertools import combinations
import time
import copy
import pandas as pd

# leemos el fichero con los resultados del cuestionario de Belbin

belbin =
pd.read_csv(r"C:\Users\annat\OneDrive\Documents\MASTER\TFM\Modelos\student_da
ta_belbin_mbti_anon.csv",delimiter=";", index_col="student_id",
usecols=['student_id','CW','CH','SH','PL','RI','ME','TW','CF'])
umbrales=[12,11,14,9,10,10,13,7]
belbin = belbin.fillna(0)
belbin = belbin.div(umbrales,axis=1)
belbin=belbin.astype(int)
belbin = belbin.applymap(lambda x: 1 if x != 0 else 0)
r=belbin.stack().to_dict()

def obtener_combinaciones(conjunto, n):
        return list(combinations(conjunto, n))

#conjunto de roles y los baremos mínimos para poder considerarlo
Q = {'CW':12,'CH':11,'SH':14,'PL':9,'RI':10,'ME':10,'TW':13,'CF':7}


from gurobipy import GRB, Model, LinExpr
def modelo1():

    modelo= Model("Modelo 1")
    variables = {}
    for i in T:
      v = modelo.addVar(vtype=GRB.BINARY, name= "a binary variable that
represents with a 1 whether team " +str(i) +" is selected, 0 otherwise")
      variables[i] = v

    constraints1 = []

    for ce in C:
      restriccion =LinExpr()
      for t in T:
        v = variables[(t)]
        restriccion += v*g[(t,ce)]
      c = modelo.addConstr(restriccion == 1)
      constraints1.append(c)

    constraints2 = []
    for el in L:
        restriccion = LinExpr()
        for t in T:
```

```
            v = variables[t]
            restriccion += v * a[(t, el)]
        c = modelo.addConstr(restriccion == n[el])
        constraints2.append(c)

    objetivo = LinExpr()
    for t in T:
        v = variables[t]
        objetivo += v * u[t]

    modelo.setObjective(objetivo, GRB.MAXIMIZE)

    result = modelo.optimize()
    resultado = modelo.Status

    if resultado == GRB.INFEASIBLE:
        print("There is no feasible solution for the problem")
    elif resultado == GRB.INF_OR_UNBD:
        print("The solution is infeasible or unbounded")
    elif resultado == GRB.UNBOUNDED:
        print("The solution is unbounded")
    elif resultado == GRB.OPTIMAL:
        print("In the specified time limit the solver has found the optimal
solution")
        for t in T:
            v = variables[t]
            if v.x > 0:  # Usar el atributo 'x' para obtener el valor de la
solución
                print(v, v.x)
                print(T[t])
        print("The optimal value for the objective function is",
modelo.objVal)
        return modelo.objVal
    elif resultado == GRB.FEASIBLE:
        print("In the specified time limit the solver has found a feasible
solution")
        for t in T:
            v = variables[t]
            if v.x > 0:  # Usar el atributo 'x' para obtener el valor de la
solución
                print(v, v.x)
                print(T[t])
        print("The value for the objective function is", modelo.objVal)
    else:
        print("Unknown error code")

def modelo2():
    modelo = Model("Modelo2")

    xvariables = {}
    yvariables = {}
    for ce in C:
      for p in P:
```

```
        x = modelo.addVar(vtype=GRB.BINARY,name="a binary variable that
represents with a 1 whether students "+str(C[ce])+" are assigned to
"+str(p)+"-th team, 0 otherwise")
        xvariables[(ce,p)] = x

    for q in Q:
      for p in P:
        y = modelo.addVar(vtype=GRB.BINARY,name="a binary variable that
represents with a 1 whether trait " +str(q) +" is present in the " +str(p)+"-
th team, 0 otherwise")
        yvariables[(q,p)] = y

    constraints1 = []

    for ce in C:
      restriccion =LinExpr()
      for p in P:
        x = xvariables[(ce,p)]
        restriccion += x
      c = modelo.addConstr(restriccion == 1)
      constraints1.append(c)

    constraints2 = []

    for p in P:
      restriccion = LinExpr()
      for ce in C:
        x = xvariables[(ce,p)]
        if isinstance(C[ce],int):
          size = 1
        else:
          size = len(C[ce])
        restriccion += x*size
      c = modelo.addConstr(restriccion==l[p])
      constraints2.append(c)

    if N:
      constraints3 = []

      for p in P:
        for s1 in S:
          for s2 in S:
            for i in N:
              if N[i]==(s1,s2):
                for ce in C:
                  if isinstance(C[ce],int):
                    if s1 == C[ce]:
                      c1=ce
                    if s2 == C[ce]:
                      c2=ce
                  else:
                    if s1 in C[ce]:
                      c1=ce
```

```
                    if s2 in C[ce]:
                      c2=ce
                restriccion=LinExpr()
                x1 = xvariables[(c1,p)]
                x2 = xvariables[(c2,p)]
                restriccion += x1 + x2
                c=modelo.addConstr(restriccion<=1)
                constraints3.append(c)

    constraints4 = []

    for q in Q:
      for p in P:
        restriccion = LinExpr()
        for ce in C:
          x = xvariables[(ce,p)]
          restriccion += x*h[(ce,q)]
        y = yvariables[(q,p)]
        restriccion += -y
        c = modelo.addConstr(restriccion>=0)
        constraints4.append(c)

    objective = LinExpr()

    for q in Q:
      for p in P:
        y = yvariables[(q,p)]
        objective += y

    modelo.setObjective(objective, GRB.MAXIMIZE)
    result = modelo.optimize()
    resultado = modelo.Status

    if resultado == GRB.INFEASIBLE:
        print("There is no feasible solution for the problem")
    elif resultado == GRB.INF_OR_UNBD:
        print("The solution is infeasible or unbounded")
    elif resultado == GRB.UNBOUNDED:
        print("The solution is unbounded")
    elif resultado == GRB.OPTIMAL:
        print("In the specified time limit the solver has found the optimal
solution")
        for ce in C:
          for p in P:
            xvar = xvariables[ce,p]
            if xvar.x>0:
              print(xvar, xvar.x)
        print("The optimal value for the objective function is",
modelo.objVal)
        return(modelo.objVal)
    elif resultado == GRB.FEASIBLE:
        print("In the specified time limit the solver has found a feasible
solution")
```

```
        for ce in C:
          for p in P:
            xvar = xvariables[ce,p]
            if xvar.x>0:
              print(xvar, xvar.x)
        print("The value for the objective function is", modelo.objVal)
    else:
        print("Unknown error code")

def modelo3():

  modelo= Model("Modelo 3")

  variables= {}
  for t in T:
    for p in P:
      if len(T[t])==l[p]:
          v = modelo.addVar(vtype=GRB.BINARY, name="a binary variable that
represents with a 1 whether team "+str(t)+" takes position "+str(p)+" in the
class partition, 0 otherwise")
          variables[(t,p)] = v


  constraints1 = []

  for ce in C:
    restriccion = LinExpr()
    for p in P:
      for t in T:
          if len(T[t])==l[p]:
              v = variables[(t,p)]
              restriccion +=v* g[(t,ce)]
    c = modelo.addConstr(restriccion == 1)
    constraints1.append(c)

  constraints2 = []

  for p in P:
      restriccion =LinExpr()
      for t in T:
          if len(T[t])==l[p]:
              v = variables[(t,p)]
              restriccion +=v
      c = modelo.addConstr(restriccion == 1)
      constraints2.append(c)

  constraints3 = []

  for p1 in P:
    for p2 in P:
      if p1 < p2 and l[p1]==l[p2]:
          restriccion =LinExpr()
          for t in T:
```

```
            if (t,p1) in variables:
              v1 = variables[(t,p1)]
              restriccion -=v1*i[t]
            if (t,p1) in variables:
              v2 = variables[(t,p2)]
              restriccion +=v2*i[t]
          c = modelo.addConstr(restriccion >= 0)
          constraints3.append(c)

  objetivo = LinExpr()
  for t in T:
    for p in P:
      if (t,p) in variables:
          v = variables[(t,p)]
          objetivo += v * u[t]
  modelo.setObjective(objetivo, GRB.MAXIMIZE)

  result = modelo.optimize()
  resultado = modelo.Status

  if resultado == GRB.INFEASIBLE:
      print("There is no feasible solution for the problem")
  elif resultado == GRB.INF_OR_UNBD:
      print("The solution is infeasible or unbounded")
  elif resultado == GRB.UNBOUNDED:
      print("The solution is unbounded")
  elif resultado == GRB.OPTIMAL:
      print("In the specified time limit the solver has found the optimal
solution")
      for t in T:
        for p in P:
          if (t,p) in variables:
            v = variables[t,p]
            if v.x>0:
              print(v, v.x)
              print(T[t])
      print("The optimal value for the objective function is", modelo.objVal)
      return(modelo.objVal)
  elif resultado == GRB.FEASIBLE:
      print("In the specified time limit the solver has found a feasible
solution")
      for t in T:
          v = variables[t]
          if v.x > 0:  # Usar el atributo 'x' para obtener el valor de la
solución
              print(v, v.x)
              print(T[t])
      for t in T:
        for p in P:
          if (t,p) in variables:
            v = variables[t,p]
            if v.x>0:
              print(v, v.x)
```

```
                print(T[t])
        print("The value for the objective function is", modelo.objVal)
    else:
        print("Unknown error code")


tamaños = [20, 30, 40, 50, 60]

datos= pd.DataFrame(columns=['tamaño', 'instancia','modelo', 'repeticion',
'tiempo','valor objetivo'])

for instancia in range(1,11):
    print("instancia "+ str(instancia))
    for alumnos in tamaños:
        file =
"C:/Users/annat/OneDrive/Documents/MASTER/TFM/Modelos/instancia"+str(alumnos)
+"_"+str(instancia)+".xlsx"
        import openpyxl as opx
        archivo = opx.load_workbook(file)
        Hoja = archivo['Alumnos']
        L={}
        cell_obj=Hoja.cell(row = 1, column = 2)
        L[1] = cell_obj.value
        cell_obj=Hoja.cell(row = 1, column = 3)
        if cell_obj.value:
            L[2] = cell_obj.value


        max_col = Hoja.max_column
        S=[]
        for i in range(2, max_col + 1):
            cell_obj = Hoja.cell(row = 2, column = i)
            S.append(cell_obj.value)

        C={}
        if "Alumnos que deben ir juntos" in archivo.sheetnames:
          Hoja2 = archivo['Alumnos que deben ir juntos']
          max_col = Hoja2.max_column
          max_row = Hoja2.max_row
          D=[]
          for i in range(1, max_row + 1):
              for j in range(2, max_col + 1):
                cell_obj = Hoja2.cell(row = i, column = j)
                if cell_obj.value!=None:
                  D.append(cell_obj.value)
              C[i] = tuple(D)
              D = []

        N={}
        if "Alumnos que no pueden ir juntos" in archivo.sheetnames:
          Hoja3 = archivo['Alumnos que no pueden ir juntos']
          max_col = Hoja3.max_column
          max_row = Hoja3.max_row
```

```
            D=[]
            for i in range(1, max_row + 1):
                for j in range(2, max_col + 1):
                  cell_obj = Hoja3.cell(row = i, column = j)
                  D.append(cell_obj.value)
                N[i] = tuple(D)
                D = []

        #Crear el parámetro n
        n={}
        if 2 in L:
            alumnos_remaining = alumnos
            n[2]=0
            while alumnos_remaining>=L[2] and
alumnos_remaining/L[1]!=alumnos_remaining//L[1]:
              alumnos_remaining=alumnos_remaining-L[2]
              n[2]+=1
            n[1]=alumnos_remaining//L[1]
            equipos=n[1]+n[2]
        else:
            equipos = alumnos//L[1]
            n[1]=alumnos//L[1]

        # crear el conjunto P
        P = []
        for i in range(equipos):
          P.append(i+1)

        #crear el conjunto entero C incluyendo a los propios estudiantes si
no están obligados a ir con nadie más
        for m in range(alumnos):
            s = len(C)
            b = 0
            for c in C:
                if not(isinstance(C[c], int)):
                    if S[m] in C[c]:
                        b=1
            if b == 0:
                s+=1
                C[s]=S[m]

        # crear todos los equipos factibles (T)

        if 2 in L:
          if L[1]==1:
            B= S+obtener_combinaciones(S,L[2])
          else:
            B=obtener_combinaciones(S,L[1])+obtener_combinaciones(S,L[2])
        else:
          B=obtener_combinaciones(S,L[1])
        long = len(B)
        B=dict(zip(range(1,long+1), B))
```

```
        T={}
        i={}
        j=1
        uno=0
        dos=0
        for k in B:
          a = 0
          if N:
            for n in N: #escojo una pareja de alumnos no que puedan ir juntos
              b = 0
              for t in N[n]:
                if t in B[k]:
                  b += 1 #si los dos están en el equipo actual, b será 2
              if b == 2:
                a=1
          if a==0: #solo consideramos el equipo si a es 0
            T[j]=B[k]
            if isinstance(B[k], int) or len(B[k])==L[1]:
                uno += 1
                i[j]=uno
            else:
                dos += 1
                i[j]=dos
            j+=1


        # calcular el parámetro u
        u={}
        for t in T:
            u[t]=0
            for q in Q:
                a = 0
                for m in T[t]:
                    if r[(m,q)]==1:
                        a = 1
                        u[t]+=1
                        break

        #creación parámetro gamma
        g = {}
        for c in C: #recorro conjuntos de alumnos que deben ir juntos
            for t in T: #recorro equipos
                b = 0
                if isinstance(C[c], int): # si solo hay un alumno en el
conjunto de alumnos
                    if C[c] in T[t]: #compruebo si ese alumno está en el
equipo
                        b = 1
                else: # si hay más de un alumno en el conjunto
                    b = 1
                    #compruebo si los dos alumnos están en el equipo (si lo
están b=1, si no b=0)
                    for j in C[c]:
```

```
                        if j not in T[t]:
                            b = 0
                            break
                if b == 1:# si todos los alumnos del conjunto c están en el
equipo t, entonces gamma(t,c)=1
                    g[(t,c)] = 1
                else:
                    g[(t,c)] = 0

        #crear el parámetro alpha
        a={}
        for t in T:
          for el in L:
            if len(T[t])==L[el]:
              a[(t,el)]=1
            else:
              a[(t,el)]=0

        h={}
        # crear el parametro h
        for q in Q:
          for ce in C:
            if isinstance(C[ce], int):
              if r[(C[ce],q)]== 1:
                h[(ce,q)] = 1
              else:
                h[(ce,q)] = 0
            else:
                for j in range(len(C[ce])):
                  if r[(C[ce][j],q)]== 1:
                    h[(ce,q)]=1
                    break
                  else:
                    h[(ce,q)]=0

      # crear el parametro l
        l={}
        for j in range(1,equipos+1):
          if j<=n[1]:
            l[j]=L[1]
          else:
            l[j]=L[2]

        for repeticion in range(1,11):
            inicio =time.time()
            fo = modelo1()
            fin=time.time()
            datos2 = pd.DataFrame({'tamaño':[alumnos], 'instancia':
[instancia], 'modelo':[1], 'repeticion':repeticion, "tiempo":[fin-inicio],
"valor objetivo":[fo]})
            datos=pd.concat([datos,datos2])

            inicio =time.time()
```

```
            fo = modelo2()
            fin=time.time()
            datos2 = pd.DataFrame({'tamaño':[alumnos], 'instancia':
[instancia], 'modelo':[2], 'repeticion':repeticion, "tiempo":[fin-inicio],
"valor objetivo":[fo]})
            datos=pd.concat([datos,datos2])

            inicio =time.time()
            fo = modelo3(equipos, alumnos, S, copy.copy(C), copy.copy(N))
            fin=time.time()
            datos2 = pd.DataFrame({'tamaño':[alumnos], 'instancia':
[instancia], 'modelo':[3], 'repeticion':repeticion, "tiempo":[fin-inicio],
"valor objetivo":[fo]})
            datos=pd.concat([datos,datos2])

datos.to_excel("Resultados_equipos_sin_restricciones.xlsx",
index=False)
```