Universidad San Jorge

Escuela de Arquitectura y Tecnología

Grado en Diseño y Desarrollo de Videojuegos

Proyecto Final

Estudio comparativo de algoritmos de búsqueda basados en inteligencia artificial para planificador de horarios y exámenes

Autor del proyecto: Esther Pérez Jerez

Director del proyecto: Antonio Iglesias Soria

Zaragoza, 11 de septiembre de 2020

Este trabajo constituye parte de mi candidatura para la obtención del título de Graduado en Ingeniería Informática por la Universidad San Jorge y no ha sido entregado previamente (o simultáneamente) para la obtención de cualquier otro título.

Este documento es el resultado de mi propio trabajo, excepto donde de otra manera esté indicado y referido.

Doy mi consentimiento para que se archive este trabajo en la biblioteca universitaria de Universidad San Jorge, donde se puede facilitar su consulta.

Firma

Fecha: **11 de septiembre de 2020**

Dedicatoria y agradecimientos

A mis padres por apoyarme en todas mis decisiones por muy descabelladas que sean, al final no fue ni Oxford ni Harvard, fue Zaragoza. Acabo mi carrera donde empezó todo, no podía ser de otra manera.

A Antonio Iglesias, tutor de este proyecto que ha sabido guiar mis pasos en la buena dirección, espero que nuestros caminos vuelvan a cruzarse en algún proyecto y poder empaparme un poco más del conocimiento que rebozas.

A mis amigas por sus incansables ánimos y por recordarme que soy más capaz de lo que creo cuando más lo necesito, incluso a kilómetros de distancia.

A Bryan Pérez por aportarme la calma que siempre me falta y por contagiarme de su dedicación y constancia que junto con su apoyo me han empujado a acabar este proyecto.

Tabla de contenido

Resumen **1**

Abstract **1**

1. Introducción **3**

2. Contexto **5**
2.1. Soluciones existentes **6**
2.1.1. GHC Peñalara 6
2.1.2. ascHorarios 7
2.1.3. FET Horarios 8
2.2. Comparativa **9**
2.3. Conclusión **9**

3. Objetivos **11**

4. Metodología **13**
4.1. Extreme Programming (XP) **13**
4.2. Adaptación de la metodología **14**
4.3. Seguimiento del desarrollo **14**
4.3.1. Trello 14
4.3.2. Diagrama de trabajo 15

5. Análisis **17**
5.1. Análisis del problema **17**
5.2. Algoritmos de búsqueda **18**
5.2.1. Algoritmo de búsqueda aleatoria 18
5.2.2. Algoritmos de búsqueda genéticos 18
5.2.3. Algoritmos de búsqueda dirigida 19
5.3. Análisis de la interfaz **19**

6. Diseño **21**
6.1. Diseño del editor **21**
6.1.1. Vistas generales 22
6.1.2. Vistas concretas 24
6.2. Diseño del planificador **26**

7. Implementación **29**
7.1. Iteración 1 – Estructura base **29**
7.1.1. Tareas 29
7.1.2. Desarrollo 29
7.1.3. Duración 31
7.2. Iteración 2 – Búsqueda aleatoria y evaluación de soluciones **31**
7.2.1. Tareas 31
7.2.2. Desarrollo 32
7.2.3. Duración 33
7.3. Iteración 3 – Búsqueda genética **34**
7.3.1. Tareas 34
7.3.2. Desarrollo 34
7.3.3. Duración 35

7.4. Iteración 4 e Iteración 5 – Interfaces gráficas 35
7.4.1. Tareas 35
7.4.2. Desarrollo 36
7.4.3. Duración 39
7.5. Iteración 6 – Búsqueda dirigida 39
7.5.1. Tareas 39
7.5.2. Desarrollo 40
7.5.3. Duración 40
7.6. Iteración 7 – Usabilidad y pruebas con experto 41
7.6.1. Pruebas con experto 41
7.6.2. Análisis 42
7.6.3. Tareas 42
7.6.4. Desarrollo 43
7.6.5. Duración 46
7.7. Iteración 8 – Estudio estadístico y optimización 46
7.7.1. Tareas 46
7.7.2. Desarrollo 46
7.7.3. Duración 48
7.8. Iteración 9 – Manual de usuario 48
7.8.1. Tareas 48
7.8.2. Desarrollo 48
7.8.3. Duración 48
8. Resultados de la aplicación 49
8.1. Algoritmo de búsqueda aleatorio 50
8.1.1. Puntuación media 50
8.1.2. Mejora por iteración 51
8.1.3. Tiempo de ejecución e iteraciones necesarias 52
8.1.4. Población total generada 53
8.2. Algoritmo de búsqueda genético 54
8.2.1. Puntuación media 54
8.2.2. Mejora por iteración 55
8.2.3. Tiempo de ejecución e iteraciones necesarias 56
8.2.4. Iteraciones / tiempo 57
8.2.5. Población total generada 57
8.3. Algoritmo de búsqueda dirigido 58
8.3.1. Puntuación media 58
8.3.2. Mejora por iteración 59
8.3.3. Tiempo de ejecución e iteraciones necesarias 60
8.3.4. Iteraciones / tiempo 61
8.3.5. Población total generada 61
8.4. Comparativa 62
8.4.1. Puntuación media 62
8.4.2. Mejora por iteración 63
8.4.3. Tiempo de ejecución e iteraciones necesarias 64
8.4.4. Iteraciones / tiempo 65
8.4.5. Población total generada 66
8.5. Conclusión 67
9. Estudio económico 69

9.1. Desglose de costes 69
9.1.1. Costes materiales 69
9.1.2. Costes humanos 69
9.1.3. Costes de infraestructura 70
9.1.4. Costes totales 70
10. Resultados 71
11. Conclusiones 73
11.1. Propuestas de mejora 73
12. Bibliografía 75
13. Anexos 77
13.1. Anexo 1: Propuesta 77
13.2. Anexo 2: Reuniones 78
13.3. Anexo 3: Historias de usuario 79
13.4. Anexo 4: Horas trabajadas 94

Tabla de ilustraciones

Ilustración 1 - Introducción de datos GHC Peñalara 6
Ilustración 2 - Vista de solución GHC Peñalara 7
Ilustración 3 - Vista de datos ascHorarios 7
Ilustración 4 - Introducción de datos FET Horarios 8
Ilustración 5 - Vista de datos FET Horarios 8
Ilustración 6 - Seguimiento con Trello 15
Ilustración 7 - Computo de horas 15
Ilustración 8 - Diagrama entidad relación 21
Ilustración 9 - Diseño inicial de la base de datos 22
Ilustración 10 - Diseño de la vista general de la aplicación web 22
Ilustración 11 - Diseño inicial para la muestra de soluciones 23
Ilustración 12 - Diseño de la introducción de horarios 24
Ilustración 13 - Diseño inicial para la creación de horarios 24
Ilustración 14 - Diseño inicial para la asignación de disponibilidad para profesores 25
Ilustración 15 – Diseño inicial asignación de horarios a grados 25
Ilustración 16 – Diseño inicial para la asignación de asignaturas a almunos 26
Ilustración 17 – Diseño inicial diagrama de clases 26
Ilustración 18 - Estructura de las soluciones 27
Ilustración 19 - Diseño final diagrama de clases 30
Ilustración 20 - Diagrama de clases de las soluciones 32
Ilustración 21 - Diagrama del funcionamiento del algoritmo de búsqueda aleatorio 33
Ilustración 22 - Diagrama del funcionamiento del algoritmo de búsqueda genético 35
Ilustración 23 - Pantalla inicial 36
Ilustración 24 - Pantalla creación de nuevo horario 36
Ilustración 25 - Pantalla de creación de un nuevo profesor 37
Ilustración 26 - Pantalla de creación de una nueva asignatura 37

Ilustración 27 – Pantalla de creación de un nuevo alumno 38
Ilustración 28 - Ventana modal para la elección de un grupo o curso existente 38
Ilustración 29 – Visualización de las soluciones 39
Ilustración 30 - Diagrama del funcionamiento del algoritmo de búsqueda dirigido 40
Ilustración 31 – Página de inicio 43
Ilustración 32 – Página para la administración de datos 43
Ilustración 33 – Formulario de edición de asignaturas 44
Ilustración 34 – Formulario de creación de plantillas horarias 44
Ilustración 35 – Formulario de creación de grados 45
Ilustración 36 – Formulario para la selección de disponibilidad/no disponibilidad de un docente 45
Ilustración 37 - Diagrama de clases para las estadísticas 47
Ilustración 38 - Manual de usuario 48
Ilustración 39 - Puntuación media población padre por iteración algoritmo de búsqueda aleatorio 50
Ilustración 40 - Media de puntuaciones medias población padre por iteración algoritmo de búsqueda aleatorio 50
Ilustración 41 - Porcentaje de mejora población padre por iteración algoritmo de búsqueda aleatorio 51
Ilustración 42 - Media de los porcentajes de mejora de la población padre algoritmo de búsqueda aleatorio 51
Ilustración 43 - Tiempo total de ejecución algoritmo de búsqueda aleatorio 52
Ilustración 44 - Iteraciones totales algoritmo de búsqueda aleatorio 52
Ilustración 45 - Tiempo medio por iteración algoritmo de búsqueda aleatorio 53
Ilustración 46 - Soluciones totales generadas algoritmo de búsqueda aleatorio 53
Ilustración 47 - Puntuación media población padre por iteración algoritmo de búsqueda genético 54
Ilustración 48 - Media de puntuaciones medias población padre por iteración algoritmo de búsqueda genético 54

Ilustración 49 - Porcentaje de mejora población padre por iteración algoritmo de búsqueda genético 55

Ilustración 50 – Media de porcentajes de mejora población padre por iteración algoritmo de búsqueda genético 55

Ilustración 51 - Tiempo total de ejecución algoritmo de búsqueda genético 56

Ilustración 52 - Iteraciones totales necesarias algoritmo de búsqueda genético 56

Ilustración 53 – Media de tiempo por iteración algoritmo de búsqueda genético 57

Ilustración 54 - Soluciones totales generadas algoritmo de búsqueda genético 57

Ilustración 55 - Puntuación media población padre por iteración algoritmo de búsqueda dirigido 58

Ilustración 56 - Media de puntuaciones medias población padre por iteración algoritmo de búsqueda dirigido 58

Ilustración 57 - Porcentaje de mejora población padre por iteración algoritmo de búsqueda aleatorio 59

Ilustración 58 – Media de porcentajes de mejora población padre por iteración algoritmo de búsqueda genético 59

Ilustración 59 - Tiempo total de ejecución algoritmo de búsqueda dirigido 60

Ilustración 60 - Iteraciones totales algoritmo de búsqueda dirigido 60

Ilustración 61 - Tiempo medio por iteración algoritmo de búsqueda dirigido 61

Ilustración 62 - Soluciones totales generadas algoritmo de búsqueda dirigido 61

Ilustración 63 - Comparativa puntuación media población padre algoritmos de búsqueda 62

Ilustración 64 - Comparativa porcentaje de mejora población padre algoritmos de búsqueda .. 63

Ilustración 65 - Comparativa tiempo total de ejecución algoritmos de búsqueda 64

Ilustración 66 - Comparativa iteraciones totales necesarias algoritmos de búsqueda 64

Ilustración 67 - Comparativa tiempo medio por iteración algoritmos de búsqueda 65

Ilustración 68 - Comparativa soluciones totales generadas algoritmos de búsqueda 66

Ilustración 69 - Planificación inicial y final 71

Ilustración 70 - Planificación total 72

Tabla de tablas

Tabla 1 - Comparativa de soluciones existentes 9

Tabla 2 - Valoración de las restricciones 30

Tabla 3 - Comparación datos relevantes de los algoritmos de búsqueda 67

Tabla 4 - Costes materiales 69

Tabla 5 - Costes humanos 69

Tabla 6 – Costes de infraestructura 70

Tabla 7 – Costes totales 70

Resumen

Este trabajo se ha elaborado como Proyecto Fin de Grado para la titulación "Grado en Diseño y Desarrollo de Videojuegos" de la Universidad San Jorge.

La idea de este proyecto nace de la necesidad de ayudar a un perfil concreto dentro de las instituciones educativas, principalmente, a aquellas personas que se encargan de la elaboración y coordinación de los horarios académicos, así como de los calendarios de exámenes.

El proceso de planificación de los horarios para las clases y los calendarios de exámenes es fundamental en las universidades y actualmente, es un proceso que se hace de forma manual y conlleva una gran inversión de tiempo. Sin embargo, este trabajo podría automatizarse con el uso de un programa informático que, con una interfaz amigable, aplicase un algoritmo para reducir drásticamente el tiempo invertido en esta tarea.

Para esto se desarrollará un estudio comparativo con tres algoritmos de búsqueda distintos con el fin de determinar cuál es el más adecuado para este tipo de problema. Además, se utilizarán datos reales de la universidad de cursos académicos anteriores para su estudio y poder sacar conclusiones basadas en un escenario real.

Abstract

This work has been prepared as a Final Degree Project for the degree "Degree in Video Game Design and Development" at Universidad San Jorge.

The idea of this project arises from the need to help a specific profile within educational institutions, mainly, those people who are responsible for the preparation and coordination of academic timetables, as well as exam schedules.

The process of planning class timetables and exam schedules is essential in universities and currently, it is a process that is done manually and involves a great investment of time. However, this work could be automated with the use of a software that, with a friendly interface, applied an algorithm to drastically reduce the time invested in this task.

For this, a comparative study with three different search algorithms will be developed in order to determine which is the most suitable for this type of problem. In addition, real data from the university from previous academic courses will be used to study it and to draw conclusions based on a real scenario.

1. Introducción

La elaboración de una planificación de horarios de clase y/o calendario de exámenes es un proceso complejo que sigue realizándose de forma manual, esto implica que la persona encargada de esta tarea debe generar una planificación preliminar tratando de tener en cuenta la mayor cantidad posible de condiciones o restricciones que pudiera haber. Sin embargo, esta primera aproximación, en su mayoría, presenta fallos, por lo que es sometida a comprobaciones, como mínimo, por parte del resto del equipo docente. Tras esta revisión, se aplicarán los cambios oportunos, una vez más tratando de respetar las restricciones, y se volverá a someter al proceso de validación, de esta forma y tras varias iteraciones, hasta no tener qué solucionar ningún error, se habrá llegado a la planificación final.

En el sector educativo, cómo en muchos otros, la presencia de la tecnología es cada vez mayor, la cantidad de tareas que son automatizadas o simplificadas por el uso de ella crece aceleradamente. Hoy en día, todos los datos necesarios para generar esta planificación están almacenados, toda la información relacionada con los cursos, los grados, las asignaturas, los profesores y los alumnos ya están digitalizados ¿por qué no usar esta información? De la mano de un algoritmo se podría encontrar una solución válida en un período de tiempo mucho menor.

Por este motivo es por lo que se decide desarrollar este Proyecto Fin de Grado, cuyo propósito es la creación de una aplicación web que facilite a aquellas personas encargadas de esta tarea, la elaboración de los horarios de clases y de los calendarios de exámenes.

2. Contexto

La planificación de clases es un problema de organización, extrapolable a cualquier otro ámbito o sector como podría ser la gestión de los turnos y recursos de un hospital. No obstante, para este proyecto se aplicará al ámbito educativo ya que se conoce más en profundidad y se ha participado de manera activa en esta planificación previamente.

Generar estos horarios de clases y exámenes desde cero es muy complejo, no solo a nivel de entendimiento, sino que, debido a la multitud de datos a gestionar, también es muy difícil de visualizar.

Supongamos que queremos llevar a cabo la planificación de los horarios para un único semestre de un curso concreto de un grado, con un horario formado por dos franjas horarias de lunes a viernes. Este semestre de un curso cuenta con 5 asignaturas esto implica que existen $5! = 120$ soluciones posibles de organizarlas entre sí, con tantas posibilidades parece fácil dar con una de ellas. Sin embargo, otro dato para tener en cuenta en este tipo de planificaciones son las restricciones.

En primer lugar, hay que tener en cuenta que este curso se está planificando de manera individual pero muy probablemente, los profesores de estas asignaturas impartan otras a otros cursos que compartirán las mismas franjas horarias, por lo que no podrá haber dos asignaturas con el mismo profesor a la vez, aunque sean de otro curso.

Por este mismo motivo, hay que tener en cuenta que no haya alumnos matriculados de dos asignaturas que compartan franja horaria, aunque sean de distinto curso, ya que en el ámbito universitario se da la peculiaridad de que los alumnos pueden matricularse de asignaturas de cursos distintos a la vez.

Además, la persona encargada de esta planificación debe evitar colocar más cantidad de asignaturas en la misma franja horaria que de aulas disponibles en el centro.

Otra restricción es la disponibilidad de los docentes, muchos profesores compaginan varios trabajos por lo que pueden tener ciertas restricciones de horario.

Si tenemos en cuenta todas estas restricciones, el abanico de 120 posibilidades que mencionábamos reduce cada vez más su índice de éxito, forzando así a la persona designada a tener en mente todas las restricciones de todas y cada una de las asignaturas y profesores. Recordemos que hablábamos de un único semestre, mientras que si intentáramos organizar las 10 asignaturas de las que suele constar un curso hablaríamos de $10! = 3.628.800$ posibilidades.

2.1. Soluciones existentes

Como hemos mencionado, este proceso de planificación es muy complejo de gestionar para una persona, por ello, se vio la necesidad de automatizar este proceso mediante la creación de aplicaciones destinadas a la generación de horarios.

En este apartado hablaremos de algunas de estas soluciones con el fin de analizar sus puntos fuertes y sus carencias.

2.1.1. GHC Peñalara

GHC [1] está estructurado en tres módulos distintos: el planificador, que consiste en la introducción de datos por parte del usuario; el motor, encargado de iterar hasta encontrar una solución válida y optimizarla; y por último el editor, que permite al usuario modificar la solución ofrecida. Además, cuenta con una gran cantidad de soporte en su página web, que incluye video tutoriales, atención al cliente, foros y blog.

En cuanto a la interfaz, como se puede apreciar en las siguientes ilustraciones, está saturada de opciones, resulta compleja y poco intuitiva, además de ser poco agradable a la vista.

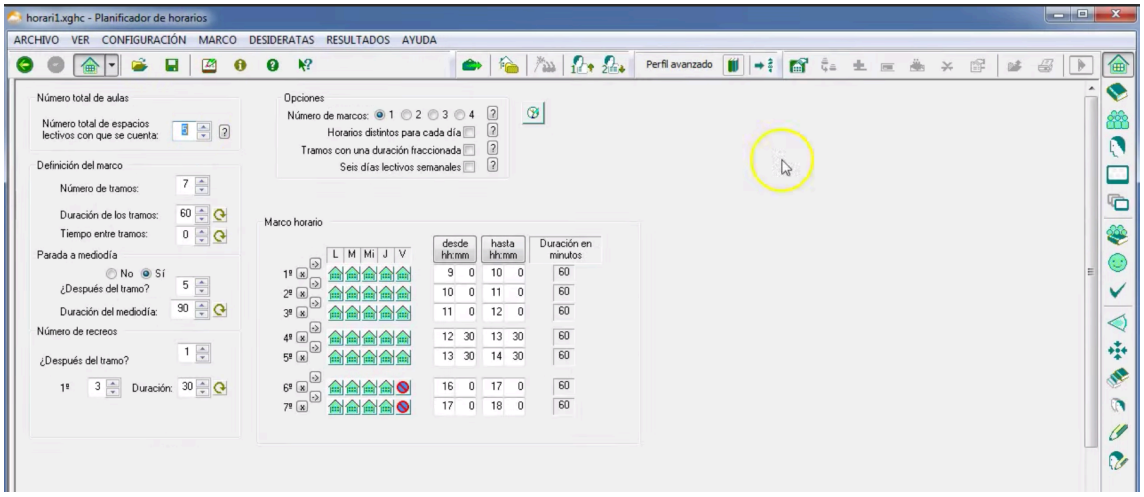


Ilustración 1 - Introducción de datos GHC Peñalara

En la *Ilustración 1*, se muestra una vista de la interfaz para la introducción de datos, dónde en la parte central se introducirán los datos relacionados con la plantilla horaria a generar y en la barra lateral derecha se encuentran el resto de los elementos que pueden ser añadidos como profesores o asignaturas, por ejemplo.

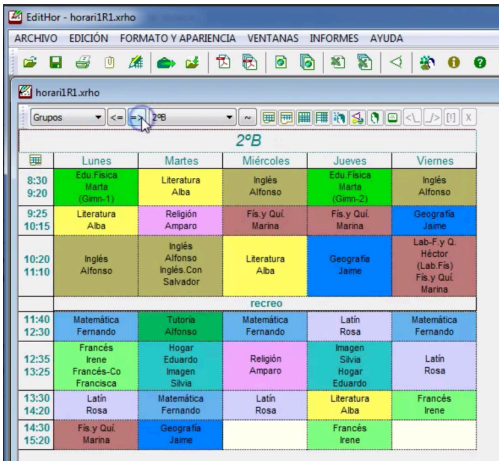


Ilustración 2 - Vista de solución GHC Peñalara

En la *Ilustración 2*, Se muestra un ejemplo de la vista utilizada para mostrar al usuario la solución, en este caso el horario de un grupo concreto de alumnos, Ofreciendo la posibilidad de cambiar de grupo en el menú superior donde podemos ver el ratón en la captura.

2.1.2. ascHorarios

ascHorarios [2] es un software que nace hace 20 años está presente en más de 173 países y se ha utilizado en 1.500 centros educativos. Ofrece distintos planes según las necesidades de los usuarios y además cuenta con un servicio de Atención al Cliente.

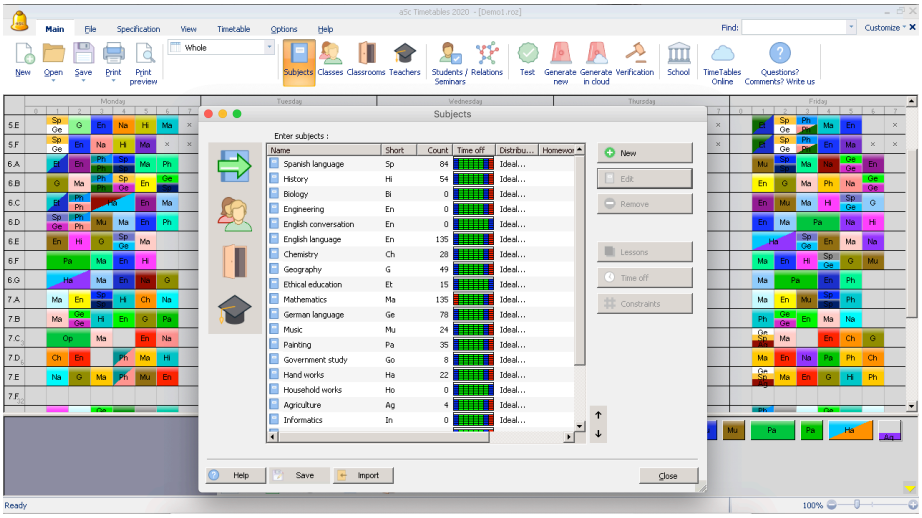


Ilustración 3 - Vista de datos ascHorarios

En la *Ilustración 3* podemos ver los distintos elementos clave de la interfaz de ascHorarios. En la sección 1, se encuentran resaltados las distintas opciones del menú de inserción de datos cómo pueden ser por ejemplo profesores o asignatura. En la sección dos, se puede ver cómo se mostraría al usuario la información, en este caso concreto, de las asignaturas ya introducidas. En la sección 3, podemos ver un ejemplo de cómo se muestra la solución a los usuarios.

2.1.3. FET Horarios

FET Horarios [3] es un software libre para la planificación de horarios escolares, que además, es multiplataforma.

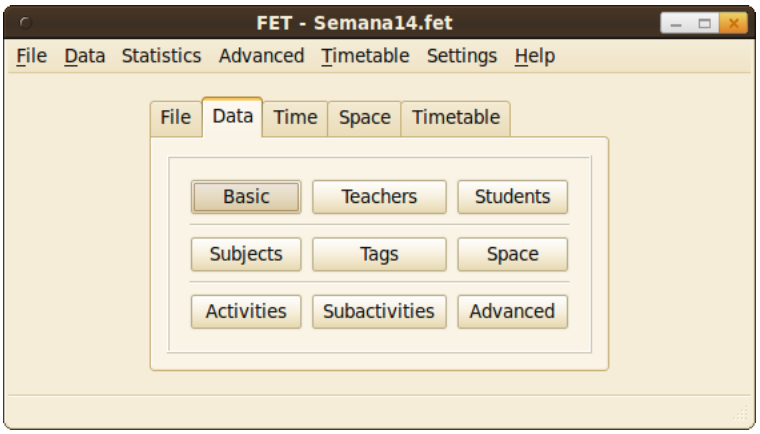


Ilustración 4 - Introducción de datos FET Horarios

En la *Ilustración 4,* se muestra la pantalla de inicio de la aplicación donde entre las diferentes opciones se encuentran la inserción de datos manual o mediante ficheros.

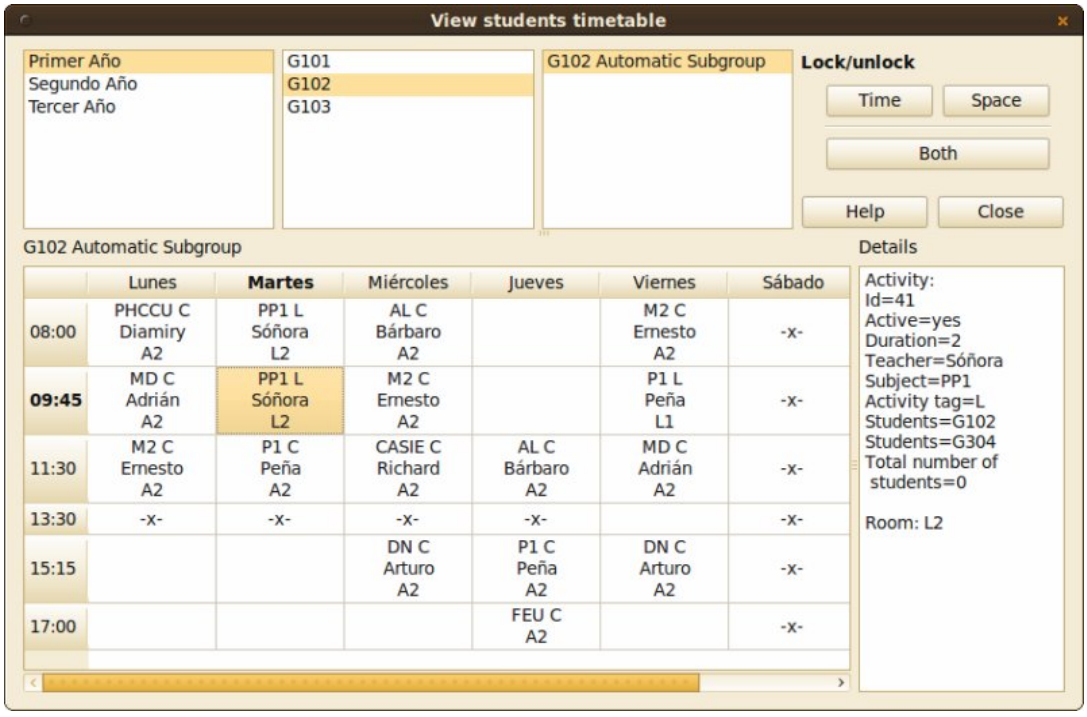


Ilustración 5 - Vista de datos FET Horarios

En la Ilustración 5, se puede ver el formato en el que se muestra a los usuarios el resultado final con la planificación de las clases.

2.2. Comparativa

	GHC Peñalara	*ascHorarios*	*FET Horarios*
Gratuidad	Versión de prueba 15 días con limite de 10 profesores	Versión de prueba, funcionalidad completa salvo exportar e incluye marca de agua	Totalmente gratuito
Código abierto	No	No	Sí
Plataformas soportadas	Windows	Windows	Windows, Linux, MacOS
Nivel escolar*	Colegio y universidad	Colegio	Colegio
Función de edición	Sí	Sí	No

Tabla 1 - Comparativa de soluciones existentes

*Esto indica si se contempla la posibilidad de que haya alumnos con asignaturas de otros cursos como ocurre en el ámbito universitario.

2.3. Conclusión

Después de esta comparación, podemos concluir que la mayoría de las soluciones del mercado actual están enfocadas a una única plataforma, a centros principalmente de educación obligatoria y que la interfaz es compleja y poco agradable.

Además, cómo se centran prioritariamente en centros no universitarios muchos de ellos no plantean la posibilidad de que varios alumnos puedan tener asignaturas de varios grupos por lo que no tiene en cuenta el posible solapamiento de asignatura en alumnos.

No obstante, de las soluciones estudiadas GHC Peñalara ha resultado ser, por experiencia y recorrido, la mejor de estas tres por lo que, servirá de guía a la hora de desarrollar este proyecto. Además, su interfaz es la más moderna y estética, y es la aplicación qué más escenarios tiene en cuenta.

3. Objetivos

Los objetivos principales que se definieron para este proyecto, que están recogidos en la propuesta de este [Anexo 1], son:

1. Creación de una aplicación web capaz de generar una organización de clases y/o exámenes.

2. Realización de un estudio comparativo con distintos algoritmos de búsqueda para encontrar la manera más eficiente de obtener soluciones viables, óptimas o subóptimas.

3. Creación de una interfaz que permita a cualquier usuario usar la aplicación anterior para generar sus propios calendarios de clases y/o exámenes.

Durante la primera fase del proyecto, se analizaron estos objetivos y se estableció el alcance del proyecto. Se concluyó qué dichos objetivos eran adecuados y válidos para el desarrollo del proyecto.

Tras el análisis de los antecedentes, concluimos que uno de los principales problemas de las soluciones disponibles en el mercado es que están enfocadas, en su mayoría, a una única plataforma. Por ello, el primer objetivo de este proyecto es que la aplicación a implementar sea multiplataforma, para que de este modo cualquier usuario pueda hacer uso de ella independientemente de su sistema operativo.

Otro de los problemas más patentes en las soluciones actuales es que la interfaz que utilizan es antigua, poco amigable y está saturada. Por ello, otro de los objetivos de este proyecto es que la interfaz para la introducción de datos sea lo más clara, limpia y minimalista posible para facilitar esta tarea al usuario.

Por último, este proyecto tiene como objetivo un estudio comparativo entre varios algoritmos de búsqueda para tratar de demostrar que un problema que resulta muy complejo para un humano, como puede ser gestionar toda la información de un centro educativo para la generación de sus horarios y/o calendario de exámenes, puede ser resuelto en una menor cantidad de tiempo gracias a la automatización mediante un algoritmo de búsqueda.

4. Metodología

Este proyecto se va a desarrollar como Trabajo Fin de Grado, asignatura estimada en 12 créditos ECTS [4] lo que supone, aproximadamente, 320 horas de trabajo a desarrollar por, en este caso, una sola persona. Existe una fecha de entrega para este proyecto y es el 25 de junio de 2020, por lo que la flexibilidad de las iteraciones tiene un límite.

Este proyecto es compaginado con otras asignaturas, el tiempo que puede dedicarse a su desarrollo no es total. La disponibilidad para trabajar en este proyecto cambiará de una semana a otra en función de la carga de trabajo externa que se tenga en el momento.

Teniendo en cuenta las características descritas, se elige Extreme Programming (XP) [5] como metodología para este proyecto.

4.1. Extreme Programming (XP)

La elección de XP se debe a que es una metodología ágil qué se basa en la refactorización del código y la retroalimentación, además de fomentar el uso de buenas prácticas y que conlleva la utilización de diversos elementos que se consideraron de gran interés para el desarrollo de este proyecto.

Dado que el *Objetivo 2* de este proyecto es realizar un estudio comparativo entre varios algoritmos de búsqueda creemos que el hecho de que XP promueva la refactorización y la retroalimentación durante todo el desarrollo puede ser muy beneficioso a la hora de optimizar y cuidar los distintos algoritmos.

El fomento en el uso de buenas prácticas es uno de los motivos principales por lo que se elige esta metodología ya que al tratarse del primer proyecto desarrollado de manera profesional opinamos que es una buena forma de adoptar buenas costumbres.

Dentro los elementos utilizados por esta metodología, los que se encontraron más útiles para este desarrollo son las historias de usuario, las pruebas de aceptación, las estimaciones, la planificación de iteraciones y el guardado de registros de las comunicaciones entre el cliente y los desarrolladores.

Por otro lado, XP se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico [6], como en este caso puede ser la falta de experiencia en el desarrollo de software.

4.2. Adaptación de la metodología

Como la mayoría de las metodologías ágiles XP está pensado para un equipo de desarrolladores, pequeño en este caso, por lo que el primer cambio qué debe plantearse en el uso de esta metodología es el hecho de que este equipo de desarrollo está formado por una persona. Esto conlleva la eliminación de las reuniones o *stand ups* diarias.

La figura del cliente es muy importante en XP, en este caso este papel será asumido por el director del proyecto, cuya valoración y *feedback* guiarán el desarrollo de este proyecto.

Otro cambio relevante es que las iteraciones no serán definidas por una cantidad de días, como ya hemos mencionado, el tiempo que puede emplearse a este proyecto es variable, por ello al principio de cada iteración se hará una estimación del tiempo que se podrá dedicar en función de la carga de trabajo externo que se tenga y en base a ella se definirá la duración de la próxima iteración.

4.3. Seguimiento del desarrollo

Aparte de los elementos empleados por XP para el desarrollo de software, se emplearán otras herramientas para facilitar el seguimiento del proyecto.

4.3.1. Trello

Esta herramienta consiste en un tablero que permite la creación de varias columnas en las que se pueden crear, mover y eliminar tarjetas, de esta forma permite de manera muy visual conocer cuál es el estado actual en el desarrollo de un proyecto.

En nuestro caso utilizaremos *Trello* [7] de la siguiente manera: al inicio de cada iteración las historias de usuario a realizar se dividirán en pequeñas tareas, cada tarea será una tarjeta y se le añadirá una estimación y una prioridad, a medida que se vaya desarrollando la iteración estas tarjetas o tareas irán cambiando de columnas o estados hasta ser completadas.

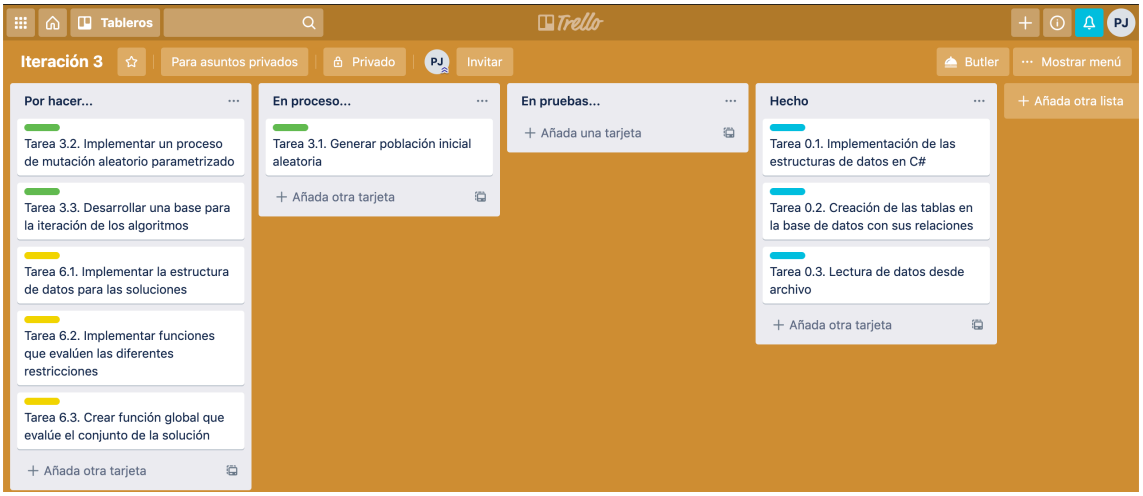


Ilustración 6 - Seguimiento con Trello

4.3.2. Diagrama de trabajo

Para hacer un seguimiento de la implicación en horas de trabajo a este proyecto se adaptó la plantilla [8] en *Excel* de un diagrama de Gantt. En ella se documenta el tiempo dedicado a cada una de las tareas de la iteración, con la finalidad de registrar y comparar el tiempo estimado con la carga de tiempo real y mejorar así las futuras estimaciones a lo largo del proyecto.

TFG			Semana 1							Semana 2							
		Estimación (h)	F	SA	SU	M	T	W	TH	F	SA	SU	M	T	W	TH	Tiempo real (h)
ID	**Iteración 2**	**35**	1	2	3	4	5	6	7	8	9	10	11	12	13	14	**43**
US 3	**Algoritmo de búsqueda aleatorio**	**20**															**28**
T3.1	Generar población inicial aleatoria		5			2											7
T3.2	Implementar un proceso de mutación aleatorio parametrizado			4				1		2			3				10
T3.3	Desarrollar una base para la iteración de los algoritmos				6			4				1					11
US 6	**Clase evaluadora**	**15**															**15**
T6.1	Implementar la estructura de datos para las soluciones											2	2				4
T6.2	Implementar funciones que evalúen las diferentes restricciones													3	4		7
T6.3	Crear función global que evalúe el conjunto de la solución															4	4
																	0

Ilustración 7 - Computo de horas

5. Análisis

5.1. Análisis del problema

La elaboración de los horarios de clases y/o calendarios de exámenes es un problema de planificación y optimización que conlleva el tratamiento de numerosos datos:

- Grados
- Cursos
- Asignaturas
- Profesores
- Alumnos
- Aulas
- Horarios

Y el resultado generado del tratamiento de estos datos debe cumplir con ciertas restricciones:

- R1: Un profesor no puede impartir dos clases a la vez
- R2: Una asignatura no puede impartirse a la vez que otra del mismo curso
- R3: Un profesor puede no tener una disponibilidad total
- R4: Un alumno no puede asistir a dos clases a la vez
- R5: Cada asignatura debe impartirse dentro del horario designado al curso al que pertenece

Como hemos mencionado se trata de un problema de planificación, en otras palabras, el problema que se nos plantea consiste en encontrar dentro de un abanico de posibilidades una solución que cumpla con los requisitos especificados en el menor tiempo posible. La comprobación manual de todas las opciones dentro de este abanico supondría un empleo enorme de tiempo para una persona, pero con la implementación del algoritmo de búsqueda correcto se reduciría el tiempo a emplear en esta tarea drásticamente.

5.2. Algoritmos de búsqueda

5.2.1. Algoritmo de búsqueda aleatoria

Los algoritmos de búsqueda aleatoria consisten en generar de manera aleatoria una gran cantidad de soluciones para un problema y conforme se va iterando estas soluciones presentan mutaciones entre iteración e iteración, generando así cada vez más soluciones de naturaleza aleatoria y seleccionando las mejores para la siguiente iteración.

De acuerdo con uno de los diversos documentos consultados [9] la búsqueda aleatoria de soluciones desemboca la mayoría de las veces en una solución óptima aplicable al problema a resolver. Sin embargo, el principal problema presente en este tipo de algoritmos es que son bastante lentos a la hora de encontrar dicha solución, problema que ha ido disminuyendo considerablemente a lo largo del tiempo debido a las mejoras de cómputo disponibles en la actualidad.

Dicho trabajo también menciona que este tipo de algoritmos suelen tener un resultado satisfactorio en problemas de gran complejidad y para los cual es no existe un procedimiento fijo a la hora de resolverlos, qué es precisamente el caso en el que nos encontramos en este proyecto, y por eso se ha decidido incluirlo como una posible opción para resolver el problema de la planificación de horarios.

5.2.2. Algoritmos de búsqueda genéticos

Los algoritmos de búsqueda genéticos se caracterizan por el cruce y las mutaciones a las que son sometidas sus soluciones, que emulan la evolución natural de los seres vivos. La mecánica de estos algoritmos consiste en que partiendo de una población inicial ésta es sometida a cruces entre sí a la vez qué parte de esa solucione son mutadas al azar como ocurría en el algoritmo de búsqueda aleatoria, después de esto se realiza un proceso de selección natural para mantener las mejores soluciones alcanzadas hasta el momento que pasaran a convertirse en la nueva población para repetir todo el proceso.

Se ha tomado la decisión de incorporar este tipo de algoritmo en el estudio de este proyecto para poder comparar con respecto al algoritmo de búsqueda aleatorio si estos pequeños cambios derivados de los cruces aportan y mejoran la calidad de las soluciones con mayor rapidez.

5.2.3. Algoritmos de búsqueda dirigida

Los algoritmos de búsqueda dirigida presentan la peculiaridad de incorporar un poco de inteligencia artificial ya que a la hora de mutar sus soluciones no lo hacen de manera aleatoria sino que lo hacen de manera controlada e informada, por ejemplo, en lugar de mutar parte de las soluciones al azar modifican genes negativos o no tan positivos con genes de mejor calidad o positivos de otras soluciones ya existentes con la intención de generar un hijo mejor.

La elección de este tipo de algoritmo para este proyecto se debe al interés de querer comprobar la mejoría con respecto a los otros dos algoritmos siendo que este añade un pequeño factor a la hora de decidir las mutaciones que podría acortar el camino hacia la solución óptima.

Se ha decidido implementar este tipo de algoritmos para resolver el problema de la planificación de horarios para comparar si la aplicación de mutaciones controladas o con una finalidad representa realmente una mejora drástica con respecto a los otros dos algoritmos, Aunque probablemente suponga un incremento en el tiempo de ejecución.

5.3. Análisis de la interfaz

Como se menciona en el Capítulo 3, uno de los objetivos principales de este proyecto es que sea una aplicación multiplataforma, ya que, como pudimos ver en el apartado de antecedentes, uno de los aspectos a mejorar de las soluciones existentes en el mercado es que están centradas en una única plataforma. Por este motivo se implementará (Nombre del proyecto) como una aplicación web de forma que se pueda usar indistintamente del sistema operativo del usuario.

Otro gran problema que se encuentra en las soluciones disponibles en el mercado es su anticuada y poco amigable interfaz. El problema de la planificación de horarios maneja una cantidad de datos compleja y por tanto tenemos como objetivo diseñar una interfaz agradable y que ofrezca al usuario una forma limpia y sencilla de introducirlos.

6. Diseño

Este proyecto se divide en dos partes bien diferenciadas, el editor y el planificador, donde el editor será la aplicación web accesible para los usuarios donde podrán introducir todos los datos relacionados con su centro educativo y el planificador será el encargado de ejecutar los algoritmos con el fin de encontrar una planificación de horarios.

Pero antes de comenzar con el diseño de cualquiera de estas partes, es necesario diseñar las estructuras de datos a utilizar que servirán de base para ambos.

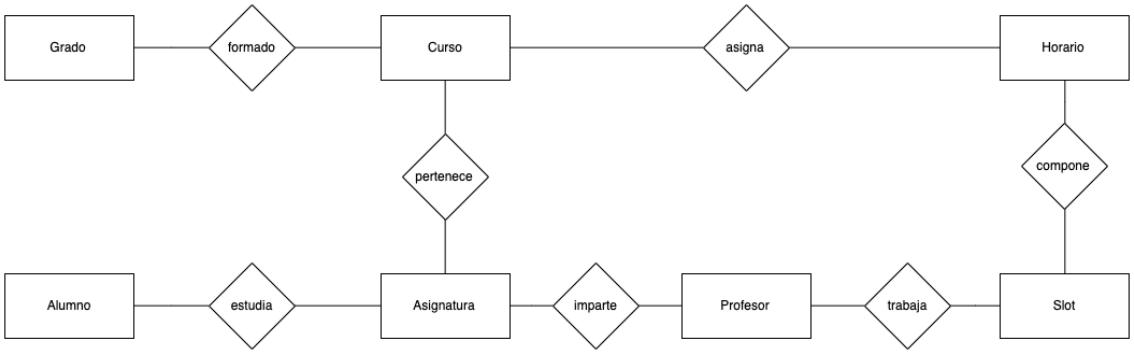


Ilustración 8 - Diagrama entidad relación

En la *Ilustración 8*, se muestra el diseño inicial de las entidades descritas en el apartado [5.1 Análisis del problema] ya enfocadas en la automatización del problema. En este proceso de diseño aparece una entidad nueva denominada *slot* qué será la encargada de representar cada uno de los períodos que componen un horario y que pueden ser asignados a una asignatura.

6.1. Diseño del editor

Después del análisis del problema y partiendo del diseño del diagrama entidad relación elaborado, se llevó a cabo el diseño de la base de datos a utilizar por el editor y que se muestra en la Ilustración 9.

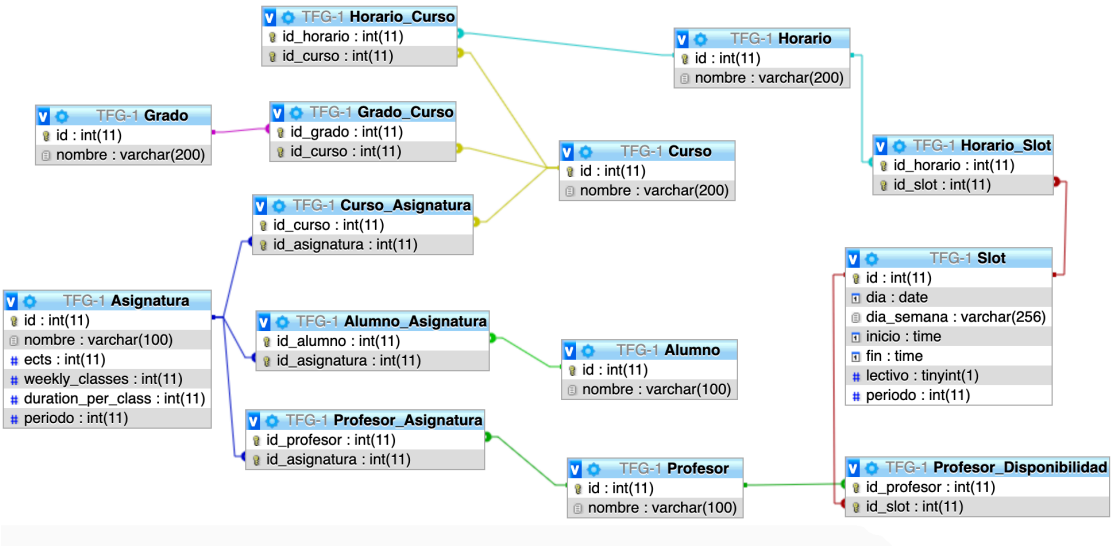


Ilustración 9 - Diseño inicial de la base de datos

A continuación, se comenzó con el diseño de las distintas vistas que conformarían la aplicación web. Separando entre vistas generales, que ofrecen una visión global del aspecto que tendrá la página, y vistas concretas, centradas en el diseño de los formularios encargados de recoger la información más compleja.

6.1.1. Vistas generales

Ilustración 10 - Diseño de la vista general de la aplicación web

En la Ilustración 10, se muestra la vista general de la página principal para la inserción de datos. Esta se compone de un *side bar* que recoge las distintas entidades del problema y una vista en forma de tabla para cada uno de ellos que muestra la lista de las instancias ya creadas de dicha entidad y cuenta con un botón, marcado en verde, para añadir una nueva y por cada fila de la tabla existen dos botones, uno para editar, marcado en azul y otro para borrar marcado en rojo. Además, como se prevé un gran volumen de datos contará con una barra de búsqueda en la parte superior para poder filtrar el contenido de la tabla.

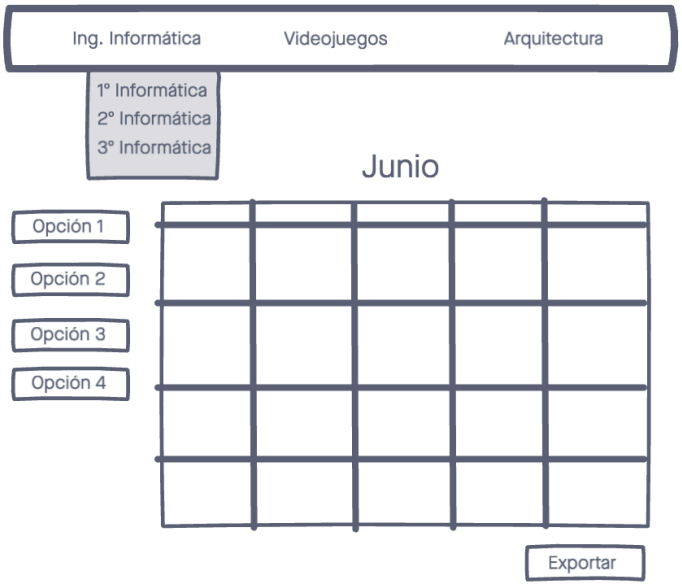


Ilustración 11 - Diseño inicial para la muestra de soluciones

En la Ilustración 11, se muestra la vista general de la página para la visualización de las soluciones.

En esta vista podemos encontrar un *top bar,* que contiene una lista de los grados para los que se ha generado un horario que, a su vez, es un *dropdown,* donde podemos encontrar los distintos cursos de dicho grado, de esta forma podemos elegir mostrar el horario generado para dicho curso.

La forma en la que se muestran las soluciones es un calendario que permite una mejor interpretación de la planificación.

Además, en la parte inferior se añadirá un botón para exportar las distintas soluciones.

Por último, en este diseño inicial se decidió añadir una serie de botones a la izquierda del calendario con la finalidad de permitir al usuario moverse entre varias soluciones posibles calculadas con los datos insertados, de forma que se pueda elegir entre ellas la más deseada.

6.1.2. Vistas concretas

Los diseños más destacables son los destinados a la introducción de la información relacionada con los horarios y la disponibilidad de los profesores. En ambos casos, la complejidad de la información es alta, por ello, después de haber hecho un análisis de las soluciones presentes en el mercado, se decidió que el formato elegido por GHC Peñalara resulta el más cómodo y sencillo, por lo que se decidió diseñar una adaptación que se muestra en la Ilustración 12.

Ilustración 12 - Diseño de la introducción de horarios

En la Ilustración 13 podemos apreciar la incorporación del diseño anterior en el formulario destinado a la creación y edición de los horarios.

Ilustración 13 - Diseño inicial para la creación de horarios

En la Ilustración 14, podemos apreciar la adaptación de este mismo diseño a la hora de añadir o editar la disponibilidad de los distintos docentes.

Ilustración 14 - Diseño inicial para la asignación de disponibilidad para profesores

Además, en este diseño se establece que a cada docente se le pueden asignar uno o más horarios existentes con la finalidad de indicar sobre los slots establecidos en los mismos la disponibilidad de los estos. De esta manera se intenta evitar la nueva entrada de franjas horarias de forma manual, ya que se entiende que solo es necesario conocer la disponibilidad de los docentes en aquellos periodos en los que sea posible impartir una asignatura.

Ilustración 15 – Diseño inicial asignación de horarios a grados

En la Ilustración 15, podemos ver el diseño elegido para la asignación de horarios a los distintos cursos dentro de un grado. El funcionamiento en este caso sería, elegir un horario del desplegable e indicar en los *checkboxes* a que cursos se desea asignar dicho horario. Además, se puede añadir otro horario al mismo grado con el botón [+] de la esquina superior, al igual que eliminar uno ya asignado seleccionando el botón [-].

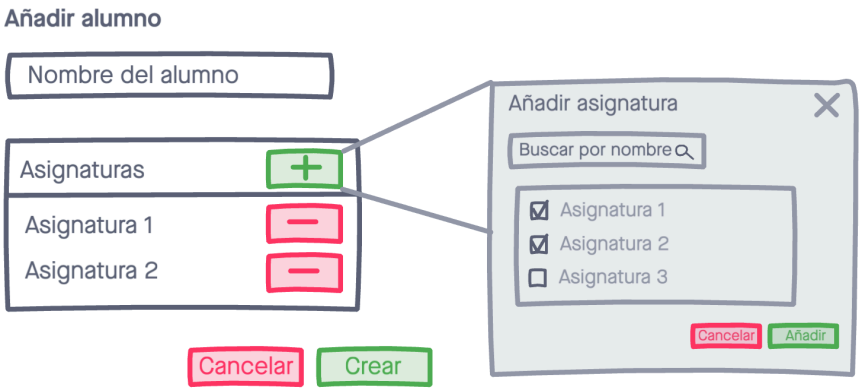


Ilustración 16 – Diseño inicial para la asignación de asignaturas a alumnos

Por último, en la Ilustración 16, se muestra el diseño ideado para la creación y edición de alumnos, en este caso, consta de un cuadro de texto para indicar un nombre y una tabla de asignaturas matriculadas, que permite añadir o eliminar elementos mediante los botones [+] y [-] respectivamente. En el caso de elegir añadir una asignatura se mostrará una ventana modal con la lista de las asignaturas ya existentes. Este modal consta de una barra de búsqueda que facilita la navegación por la lista de la cuál se pueden elegir las asignaturas a añadir a dicho alumno. Finalmente se pueden asignar mediante el botón verde o cancelar la operación mediante el botón rojo, ambos situados en la parte inferior derecha del modal.

Este mismo diseño se empleará para los profesores, formulario en el cual se podrán indicar las asignaturas a impartir por estos, y también será empleado para los formularios de asignaturas y grados, aunque en estos casos la tabla y el modal permitirán asignar los cursos a los que pertenece cada asignatura o que conforman cada grado.

6.2. Diseño del planificador

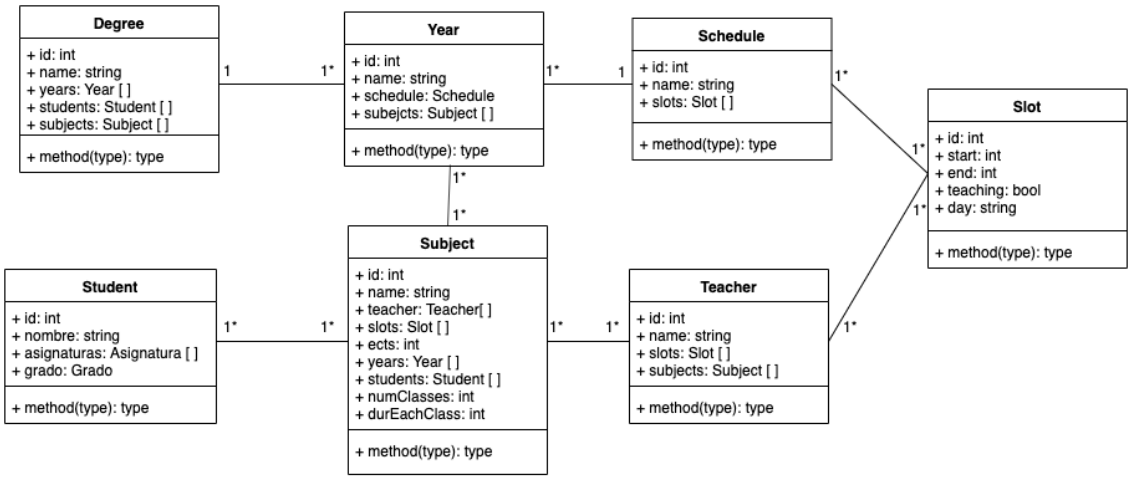


Ilustración 17 – Diseño inicial diagrama de clases

La Ilustración 17 muestra el diseño inicial del diagrama de clases que servirán de base a los algoritmos para generar las soluciones, en él se recogen los datos que en un principio se creen que serán necesarios para la correcta elaboración de los horarios. Además, podemos apreciar que este diseño es muy parecido al utilizado en la base de datos sustituyendo las tablas de relación por vectores con instancias de los elementos necesarios.

Además de esto, será necesario determinar una estructura que dé soporte a las soluciones obtenidas, que pueda ser fácilmente utilizada por los algoritmos en el proceso de generación. Por ello se plantea inicialmente que las soluciones sigan la siguiente estructura:

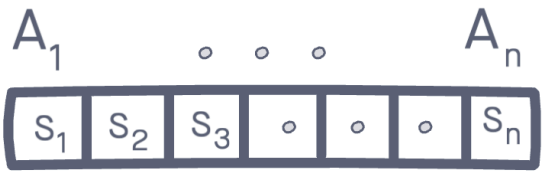


Ilustración 18 - Estructura de las soluciones

Como podemos ver en la Ilustración 18, las soluciones serán representadas mediante un vector donde cada posición del vector representará cada una de las asignaturas a organizar (A_1 – A_n) y el valor que corresponde a cada una de esas posiciones será el *Slot* o franja horaria asignada a cada asignatura.

7. Implementación

Después de analizar con detenimiento el problema y diseñar los distintos componentes necesarios para desarrollar una solución, se crearon las historias de usuario pertinentes [Anexo 3] y se procedió a comenzar con la implementación.

7.1. Iteración 1 – Estructura base

Como se ha mencionado previamente el volumen de datos a manejar en este proyecto es considerable, además de ser de suma importancia para la resolución del problema, es por eso por lo que durante las dos primeras iteraciones se llevó a cabo la historia de usuario US 0, creada para el desarrollo de una estructura de datos robusta y bien diseñada que dote de una buena base al proyecto.

7.1.1. Tareas

Las tareas en las que se dividió la US 0 son:

0.1. Implementación de las estructuras de datos en C#

0.2. Creación de las tablas en la base de datos con sus relaciones

0.3. Lectura de datos desde archivo

7.1.2. Desarrollo

Las estructuras de datos diseñadas sufrieron varias modificaciones, debido a que la duplicidad de los datos generó serios problemas a la hora de inicializar los datos.

Como podemos apreciar en el diagrama de clases mostrado en la *Ilustración 16,* se añadió un vector de enteros por cada vector de referencias a cada clase, de este modo se podrán crear las instancias iniciales indicando en este nuevo vector el *id* de la entidad con la que tiene relación para, posteriormente, una vez haya finalizado la carga de los datos, añadir al vector correspondiente la referencia con la entidad a la que pertenece dicho *id*. De este modo se facilitará y agilizará el acceso a todos estos datos durante la ejecución de los algoritmos.

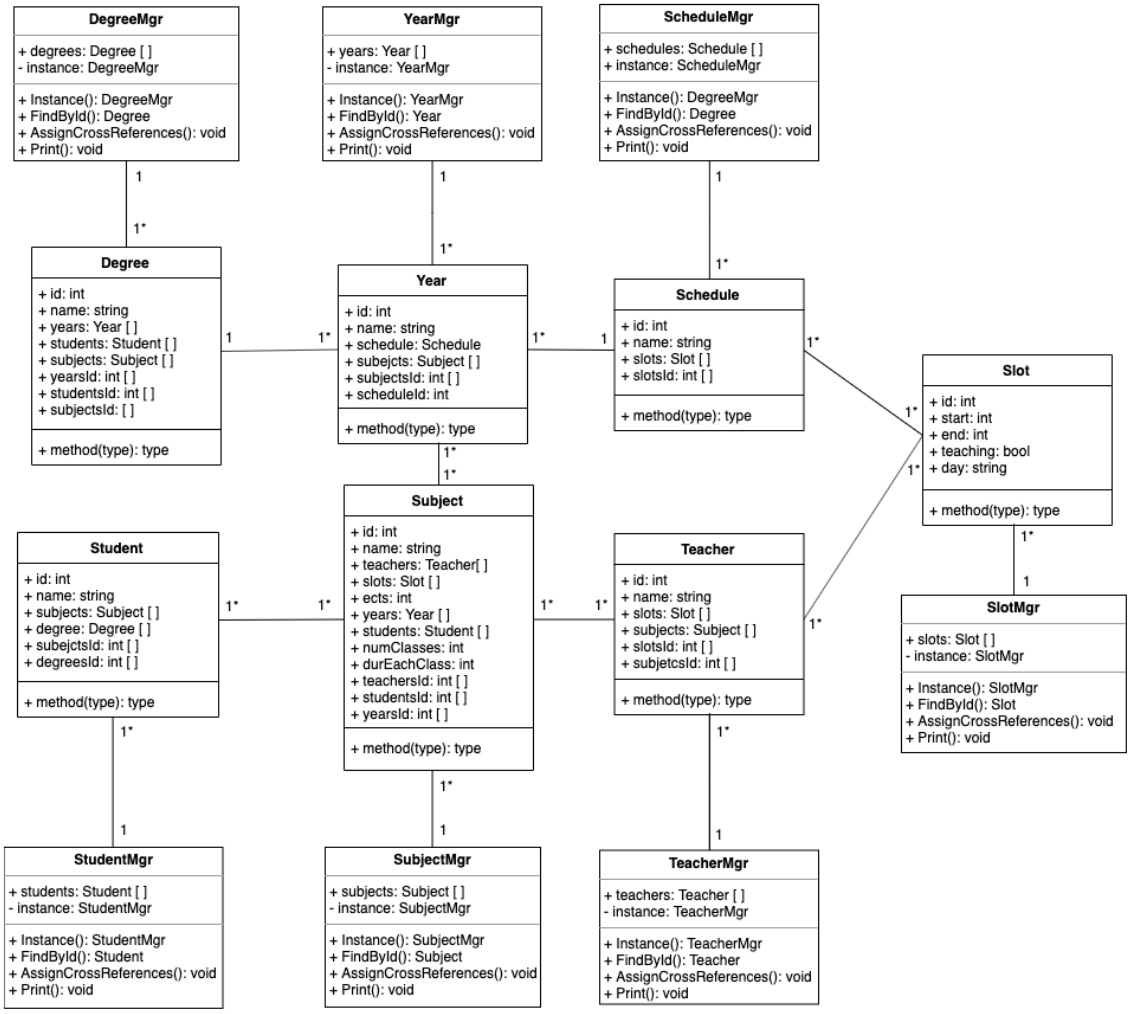


Ilustración 19 - Diseño final diagrama de clases

Además de esto, se llevó a cabo una revisión de las restricciones aplicables a este problema, se clasificaron por orden de importancia en un caso real y se les asignó una puntuación, toda esta información está recogida en la *Tabla 2.*

Restricción	Valoración
R5 (OUTOFSCHEDULE_COLLISION)	-100.000
R1 (TEACHER_COLLISION) R3 (TEACHER_SCHEDULE)	-10.000
R2 (SUBJECT_COLLISION)	-100
R4 (STUDENT_COLLISION)	-1

Tabla 2 - Valoración de las restricciones

Esta ordenación se debe al siguiente criterio de prioridades.

- R5: La asignatura debe cumplir con las limitaciones de la plantilla horaria asignada.
- R1 y R3: La asignatura debe tener a su profesor/es disponible para su impartición.
- R2: No debe haber dos asignaturas del mismo curso impartiéndose a la vez.
- R4: Los alumnos deben poder asistir a sus asignaturas matriculadas.

Las puntuaciones asignadas a cada una de estas restricciones, con distinto orden de magnitud, tienen como finalidad facilitar el entendimiento de las posibles colisiones que están teniendo lugar en una misma solución y poder así detectar fallos con mayor rapidez durante la implementación.

7.1.3. Duración

Las tareas de esta iteración estaban estimadas en 30 horas, pero se encontraron diversos problemas a la hora de implementar las estructuras de datos, causados principalmente por la duplicidad de los datos, por lo que el tiempo real empleado en esta iteración fue de 47 horas.

7.2. Iteración 2 – Búsqueda aleatoria y evaluación de soluciones

7.2.1. Tareas

US 03 – Algoritmo de búsqueda aleatorio

3.1. Generar población inicial aleatoria

3.2. Implementar un proceso de mutación aleatorio parametrizado

3.3. Desarrollar una base para la iteración de los algoritmos

US 06 – Clase evaluadora

6.1. Implementar la estructura de datos para las soluciones

6.2. Implementar funciones que evalúen las diferentes restricciones

6.3. Crear función global que evalúe el conjunto de la solución

7.2.2. Desarrollo

En primer lugar y antes de comenzar a desarrollar el primer algoritmo, es necesario implementar la estructura de datos que dará soporte a las soluciones.

Se mantuvo el diseño inicial de las soluciones, pero se vio la necesidad de crear otras estructuras para facilitar su manejo, como puede ser la creación de un controlador de soluciones (*SolutionMgr*), la estructura *Class*, que esta formada por una asignatura y un *slot,* y además se le asignó una variable *score* para almacenar la puntuación de esta combinación de asignatura y *slot*, generando así el diagrama de clases que podemos ver en la *Ilustración 17*.

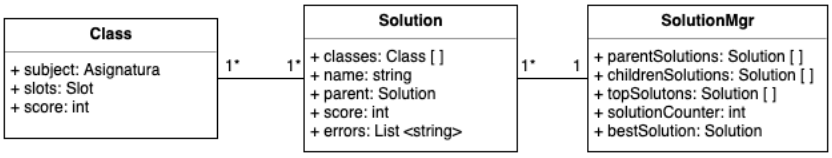


Ilustración 20 - Diagrama de clases de las soluciones

Una vez implementadas las estructuras necesarias para soportar las soluciones se procedió a la implementación de los métodos encargados de evaluar si estas soluciones cumplen con las restricciones o si por el contrario presentan colisiones. La fase de evaluación es crucial, va a establecer un criterio de discriminación de soluciones, para los algoritmos, en función del número de errores o colisiones que presenten. Para ello, se crearon los siguientes métodos:

- ***`evaluateYears:`*** método encargado de evaluar la restricción R2, comprueba todas las asignaturas dos a dos para asegurarse de que no hay ninguna colisión entre asignaturas del mismo curso.

- ***`evaluateStudents:`*** método encargado de evaluar la restricción R4, Una vez más comprueba todas las asignaturas dos a dos para comprobar que no hay ningún estudiante compartido entre asignaturas ubicadas en el mismo *slot*.

- ***`evaluateTeachers:`*** método encargado de evaluar las restricciones R1 y R3, comprueba todas las asignaturas dos para corroborar que las asignaturas colocadas en un mismo *slot* no comparten profesores.

- ***`evaluateSchedule:`*** método encargado de evaluar la restricción R5, comprueba todas las asignaturas para cerciorarse de que ninguna ha sido asignada a un *slot* fuera del horario asignado al curso al que pertenece.

Una vez se ha implementado la estructura de datos que da soporte a las soluciones y los métodos encargados de evaluarlas, se puede proceder con el desarrollo del primer algoritmo de búsqueda.

La forma en la que se ha aplicado el algoritmo de búsqueda aleatorio a este problema, mostrado en la *Ilustración 18,* es la siguiente:

En primer lugar, se genera una población de padres aleatoria que consiste en la asignación al azar de un *slot* para cada asignatura que conforma la solución.

A continuación, para cada padre se generan *n* hijos, donde *n* es el número de asignaturas de la solución y en cada una de ellas se muta al azar una asignatura.

Por último, todas estas soluciones, padres e hijos, son evaluadas y las mejor puntuadas pasarán a ser la población padre de la siguiente iteración.

Todo este proceso se repite hasta encontrar una solución óptima o que el número de iteraciones exceda el límite establecido.

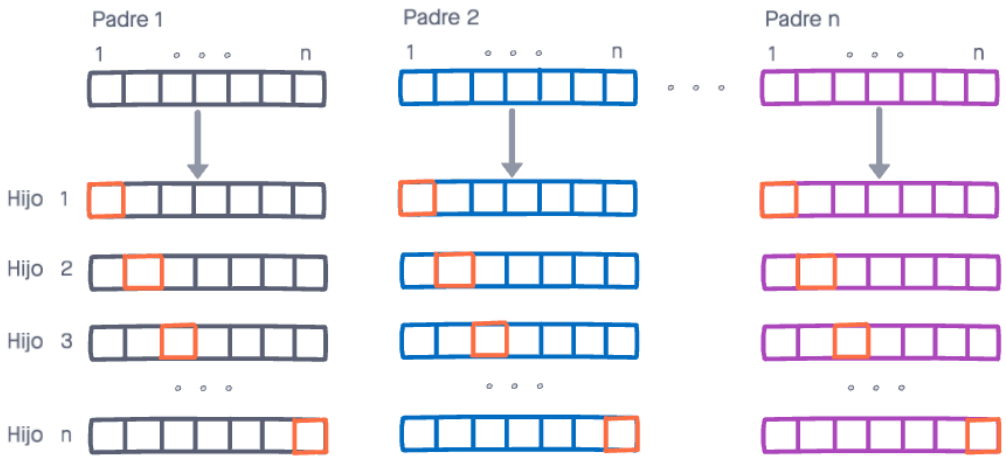


Ilustración 21 - Diagrama del funcionamiento del algoritmo de búsqueda aleatorio

7.2.3. Duración

Las tareas de esta iteración estaban estimadas en 35 horas y el tiempo real empleado en esta iteración fue de 43 horas, este pequeño retraso se debió a una falta de previsión, al ser el primer algoritmo desarrollado fue necesario implementar la estructura base que también usarían los demás algoritmos, esto llevó un poco más de tiempo de lo previsto.

7.3. Iteración 3 – Búsqueda genética

7.3.1. Tareas

US 04 – Algoritmo de búsqueda genético

4.1. Implementar varios tipos de mutaciones y/o combinaciones

7.3.2. Desarrollo

La manera en la que se ha aplicado este algoritmo a nuestro problema es la siguiente:

En primer lugar, se genera una población aleatoria de padres de igual manera que se hacía en el algoritmo de búsqueda aleatoria.

A continuación, se alternan cuatro tipos de mutaciones y/o combinaciones distintas:

- **Combinación de mitades**: en esta mutación se lleva a cabo una combinación entre 2 padres de forma que el hijo resultante se compone de la primera mitad del padre 1 y la segunda mitad del padre 2.

- **Combinación de posiciones pares:** en este caso La combinación entre los dos padres resulta en un hijo que reúne las soluciones distribuidas en las posiciones pares de ambos padres.

- **Combinación de posiciones impares:** al igual que en la mutación anterior, y como su propio nombre indica, esta combinación resulta en un hijo que reúne las soluciones distribuidas en las posiciones impares de ambos padres.

- **Mutación aleatoria:** no se trata de una combinación entre varios padres, sino de la misma mutación utilizada en el algoritmo de búsqueda aleatoria pero aplicado de manera diferente, en este caso se mutará un número *n* de clases en cada hijo.

En la *Ilustración 19*, se muestra un ejemplo visual de estos tipos de mutaciones y combinaciones:

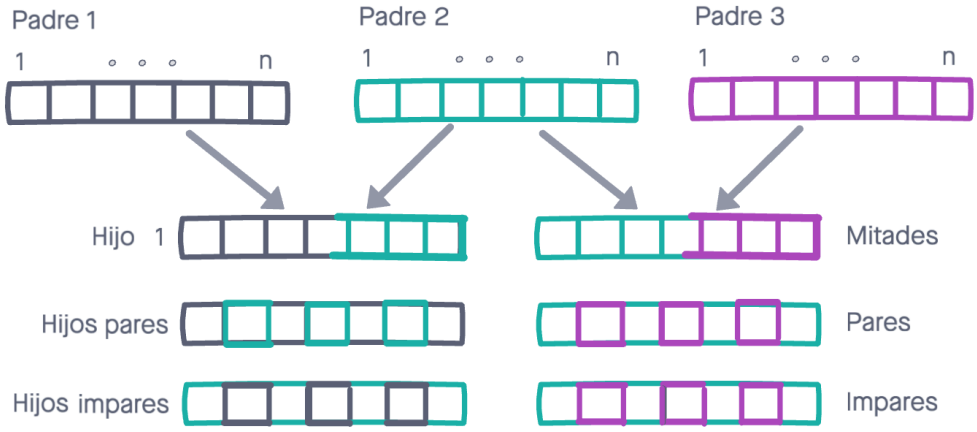


Ilustración 22 - Diagrama del funcionamiento del algoritmo de búsqueda genético

Y, por último, de igual manera que en el algoritmo anterior, todas estas soluciones son evaluadas y seleccionadas de manera iterativa hasta llegar a una solución óptima o que se agote el número de iteraciones límite.

7.3.3. Duración

Las tareas de esta iteración estaban estimadas en 30 horas, debido a que en la iteración anterior se implementó cierta base común para los algoritmos y que el código anterior fue fácilmente reutilizado el tiempo real empleado en esta iteración fue de 25 horas, menor al estimado.

7.4. Iteración 4 e Iteración 5 – Interfaces gráficas

7.4.1. Tareas

US 01 – Interfaz para la carga de datos

1.1. Página inicial

1.2. Vista general de cada elemento

1.3. Vista de edición y creación de cada elemento

US 02 – Interfaz para la muestra de soluciones

2.1. Búsqueda y adaptación de un calendario

2.2. Implementación de las soluciones en el calendario

7.4.2. Desarrollo

Durante estas iteraciones se implementaron las distintas vistas de la interfaz para la inserción de los datos y la muestra de las soluciones.

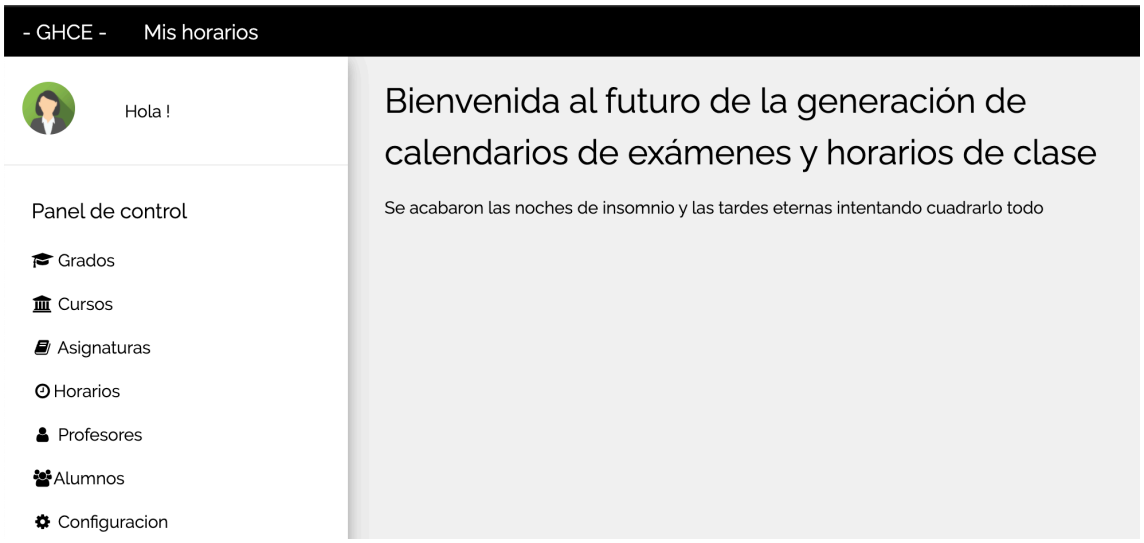


Ilustración 23 - Pantalla inicial

En la Ilustración 23 se muestra la implementación de la pantalla de inicio de la aplicación realizada durante estas iteraciones, que es bastante similar al boceto inicial que se desarrolló en la fase de diseño.

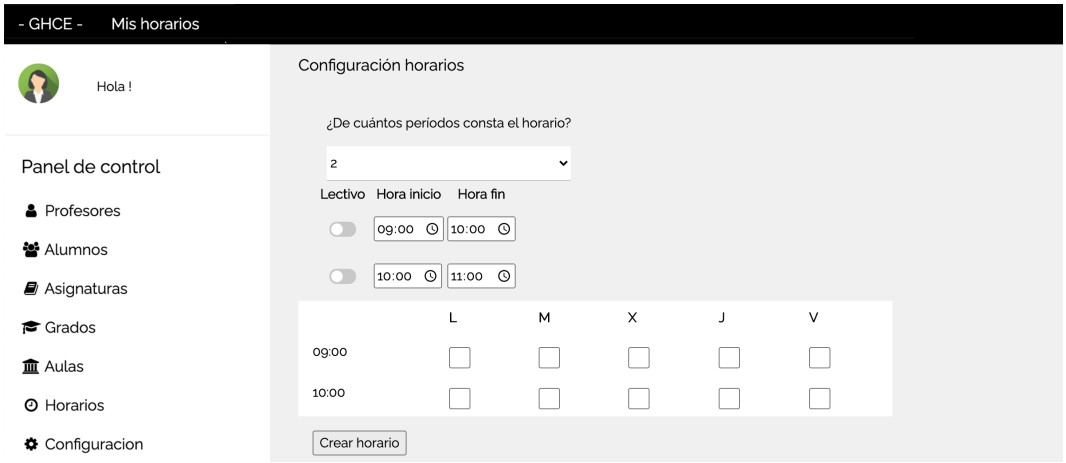


Ilustración 24 - Pantalla creación de nuevo horario

En la Ilustración 24 se muestra el resultado de la implementación de la pantalla destinada a la inserción De los datos de el nuevo horario a crear, Aunque sí que es cierto que presenta ciertas variaciones con respecto al boceto inicial ya que se ha movido el cuadro de texto destinado a la inserción del nombre de dicho horario de la parte inferior a la parte superior del formulario.

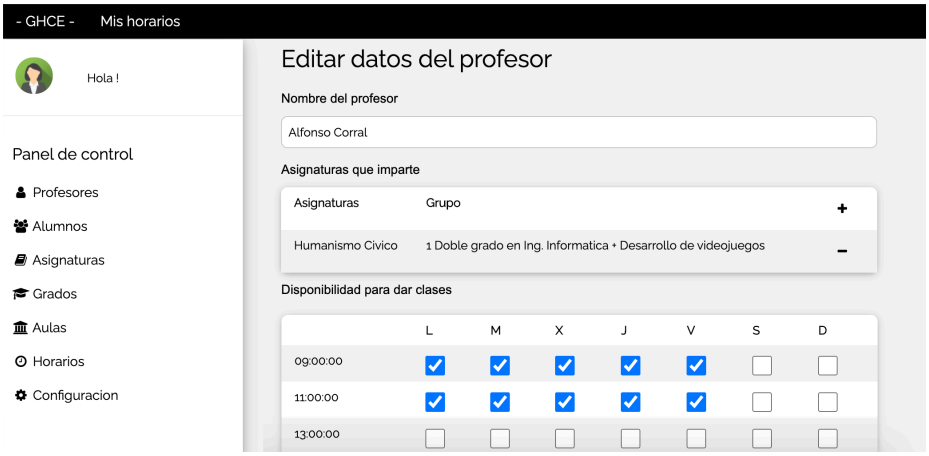


Ilustración 25 - Pantalla de creación de un nuevo profesor

En la Ilustración 25 se representa la pantalla destinada a la creación de un nuevo docente. En este caso sí que podemos apreciar varias diferencias con respecto al boceto inicial.

La idea inicial, representada en la Ilustración 14 del capítulo anterior, era asignar al profesor los horarios existentes para seleccionar sobre ellos en qué franjas dicho docente está disponible. Sin embargo, durante su implementación se pensó qué sería mucho más cómodo si estas franjas horarias se generarán automáticamente en función de las asignaturas a impartir por dicho profesor. De este modo, solo sería necesario rellenar la información realmente relevante y no un horario entero con todas las franjas horarias posibles.

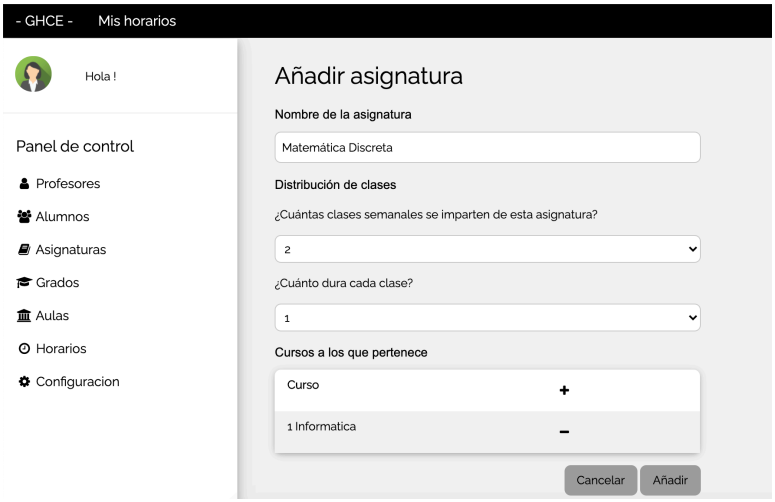


Ilustración 26 - Pantalla de creación de una nueva asignatura

En la Ilustración 26, se muestra el resultado de la implementación de la pantalla para la creación de una nueva asignatura. Salvo por la adición De un par de datos necesarios como son la duración y la repetición Semanal de la materia podemos apreciar que la implementación coincide con el diseño inicial.

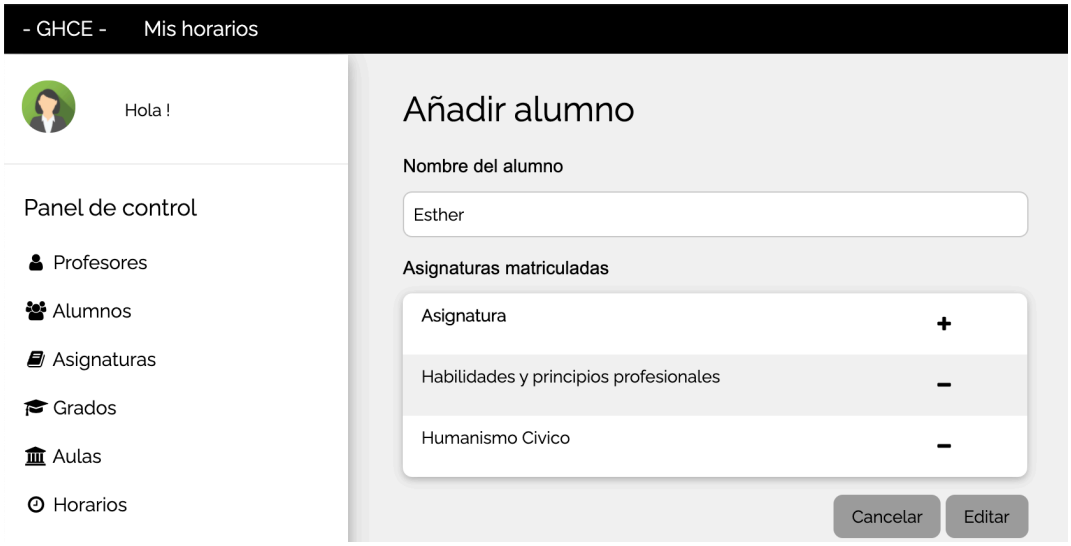


Ilustración 27 – Pantalla de creación de un nuevo alumno

En la Ilustración 27, se representa el resultado de la implementación de la pantalla destinada a la creación de un nuevo alumno, que cómo podemos apreciar no ha sufrido variaciones con respecto a su diseño inicial.

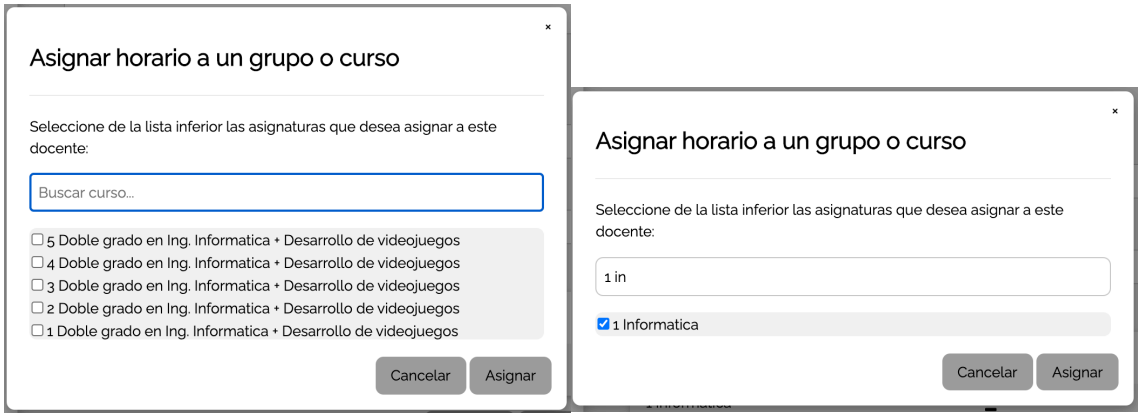


Ilustración 28 - Ventana modal para la elección de un grupo o curso existente

La Ilustración 28 el resultado de la implementación de la ventana modal destinaba a la asignación de un grupo o curso existente, este mismo diseño se repite con más elementos como pueden ser asignaturas o grados. Este tipo de ventanas modales que pudimos ver en diversos bocetos en el diseño inicial no han sufrido variaciones con respecto a su diseño y son utilizadas en la mayoría de los formularios para rellenar las tablas en las que generalmente podemos ver un botón [+].

Más concretamente a la izquierda se muestra el estado inicial derecha ventana modal mientras que la a la derecha muestra el resultado de ejecutar una búsqueda dentro de la misma.

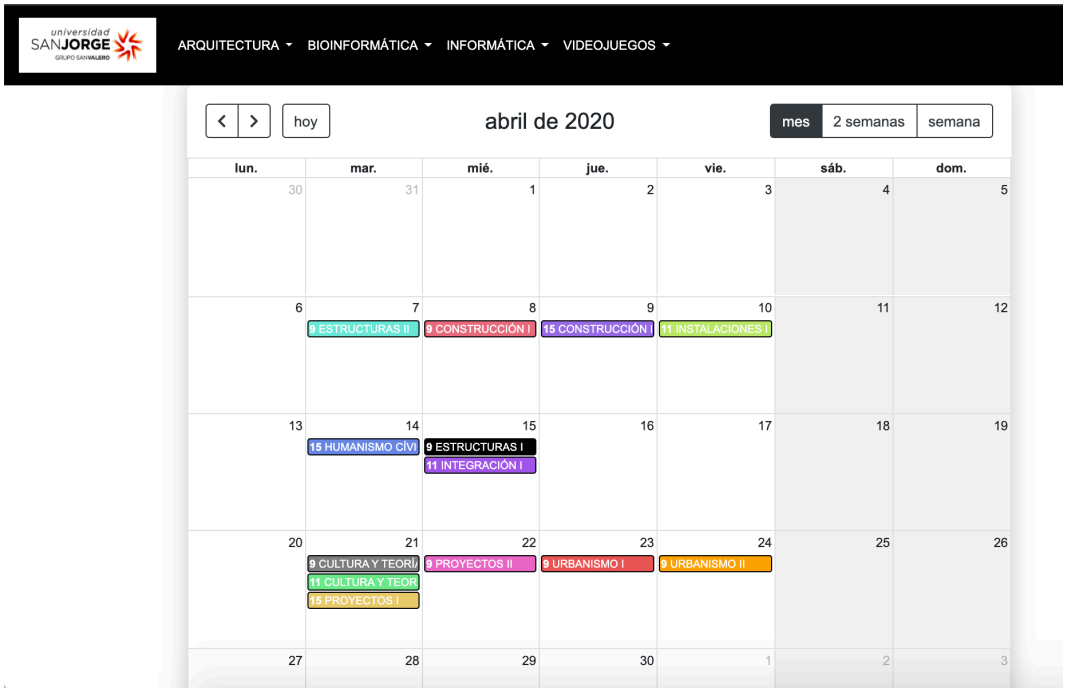


Ilustración 29 – Visualización de las soluciones

En la Ilustración 29 se muestra el resultado de la implementación de la vista destinada a las soluciones, que no ha sufrido variaciones con respecto al diseño inicial. Como se puede apreciar en la imagen, esta vista consta de un calendario que permite moverse entre distintas vistas, permite visualizar el mes completo, dos semanas o una única semana, y de una *navbar* que permite cambiar entre los distintos horarios generados, donde el grado es un *dropdown* que despliega cada curso que lo compone.

El calendario elegido para la muestra de las soluciones es *FullCalendar* [10], que es un plugin de JavaScript muy utilizado, con una gran comunidad y bastante documentación.

7.4.3. Duración

Las tareas de estas iteraciones estaban estimadas en 30 y 20 horas y el tiempo real empleado en fue de 34 horas y 22 horas respectivamente, una pequeña desviación con lo estimado en cada caso debido, a ciertos problemas con la integración del *plugin*.

7.5. Iteración 6 – Búsqueda dirigida

7.5.1. Tareas

US 05 – Algoritmo de búsqueda dirigido

5.1. Implementar mutación dirigida en combinación con otros padres

7.5.2. Desarrollo

La manera en la que se ha adaptado este algoritmo al problema de la planificación de horarios es la siguiente:

Una vez más comenzamos con la generación de una población padre aleatoria.

A continuación, se tienen en cuenta las asignaturas de esta solución que presentan algún tipo de colisión y se genera un hijo para cada una o en el caso de que sean demasiadas solo se generarán *n* hijos.

Una vez generados los hijos, se busca un padre que para esa misma asignatura (misma posición en el vector solución) no presente ninguna colisión y se sustituye en el hijo. De esta forma, se intentan solucionar algunas de las colisiones, aunque puede resultar en la generación de nuevas colisiones.

Además de esto, se sigue manteniendo la mutación aleatoria utilizada en los dos algoritmos anteriores con la finalidad de generar poblaciones más variadas.

Por último, estas nuevas soluciones se evalúan y seleccionan para seguir iterando hasta llegar a una solución válida.

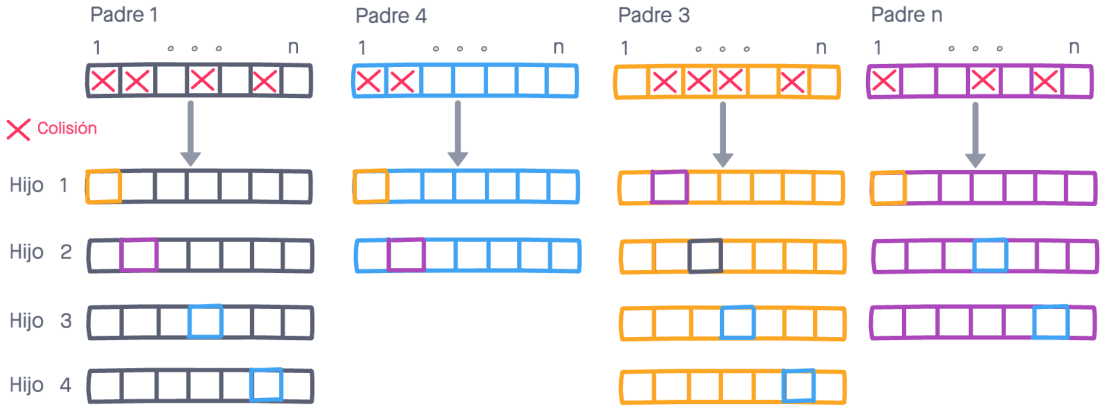


Ilustración 30 - Diagrama del funcionamiento del algoritmo de búsqueda dirigido

7.5.3. Duración

Las tareas de esta iteración estaban estimadas en 30 horas, debido a que en una iteración anterior se implementó cierta base común para los algoritmos y que el código anterior fue fácilmente reutilizado el tiempo real empleado en esta iteración fue de 28 horas, algo inferior al estimado.

7.6. Iteración 7 – Usabilidad y pruebas con experto

Debido a la complejidad del problema, el volumen de datos a procesar y la falta de experiencia, se decidió validar la implementación y el uso de los datos, hasta el momento, con una persona experta en la generación de horarios de manera manual.

7.6.1. Pruebas con experto

Durante las pruebas se recogieron los siguientes comentarios u opciones de mejora:

- C1 - Introducir todos los alumnos matriculados manualmente sería muy costoso, por ello el usuario propone modificar el concepto que se tiene de este elemento y transformarlo en "Alumno tipo" de esta forma solo es necesario crear los distintos itinerarios que pudieran darse dentro de un mismo curso.

- C2 - Las tablas de vista general recogen muy poca información acerca de los diversos elementos y sería mucho más cómodo y ayudaría a reconocer cada elemento si se mostrase acompañado de más información, como, por ejemplo, en lugar de enseñar únicamente una tabla que liste todas las asignaturas, acompañar esta información del curso o grado al que pertenece dicha materia.

- C3 - Existe un problema con el vocabulario utilizado, slots y horarios, son utilizados por este usuario como *pastilla* y *plantilla* respectivamente.

- C4 - La ejecución y la inserción de los datos deberían estar separados por cuatrimestres o periodos escolares.

- C5 - Crear las interfaces de generación de calendarios y de introducción masiva de datos de forma separada.

- C6 - La inserción de la información relativa a los grados y los cursos es un poco confusa.

- C7 - Tener en cuenta las aulas como elementos futuros con la finalidad de añadir distribución de espacios.

- C8 - El cuadrante generado para la inserción de la disponibilidad de los docentes fue muy bien recibido por parte del usuario siendo la sección con más éxito dentro de todas las pruebas realizadas. Como aportación el usuario sugiere que no sólo se pueda indicar la disponibilidad, sino que además se pueda indicar la no disponibilidad de forma que el usuario pueda elegir la manera más cómoda y rápida de rellenar dicho cuadrante.

- C9 - Con respecto a la interfaz de las soluciones el usuario sugiere que se añada la posibilidad de visualizar múltiples horarios a la vez y no sólo de manera individual.

7.6.2. Análisis

Después de recibir toda esta información por parte del usuario se analizaron las posibilidades y el tiempo restante para el proyecto y se decidió lo siguiente:

Se consideró oportuno realizar cambios en el vocabulario de la aplicación, por lo que el elemento "Alumnos" pasó a llamarse "Tipología de alumno" y el elemento "Horarios" pasó a llamarse "Plantillas horarias". Estos cambios, originados por los comentarios C1 y C3, se consideraron oportunos ya que durante las pruebas generaron bastante confusión y se cree que estos nuevos términos son más fieles a lo que estos elementos representan.

Se creará una nueva historia de usuario denominada [US 07 - Cambios de usabilidad], que recogerá las acciones necesarias para resolver los comentarios C1, C3, C4, C5, C6 y C8.

Además, derivado del comentario C3, también se decidió redactar una nueva historia de usuario denominada [US 08 – Manual de usuario], que consistirá en añadir a la web una pequeña guía que incluya una explicación de los distintos términos y elementos utilizados en la aplicación para que el usuario pueda conocer en más detalle cuales son estos elementos y como utilizarlos.

Los comentarios C7 y C9 se consideraron de gran interés para el proyecto y por ello se generaron las historias de usuario [US 09 – Gestor de espacios] y [US 10 – Vista múltiple] respectivamente.

7.6.3. Tareas

US 07 – Cambios en la usabilidad

7.1. Separación de la inserción de datos y la generación

7.2. Implementación de varios periodos en una misma ejecución

7.3. Cambiar el método de creación de cursos

7.4. Añadir la opción de disponibilidad o no disponibilidad para los docentes

7.5. Cambios en la terminología de la aplicación

7.6.4. Desarrollo

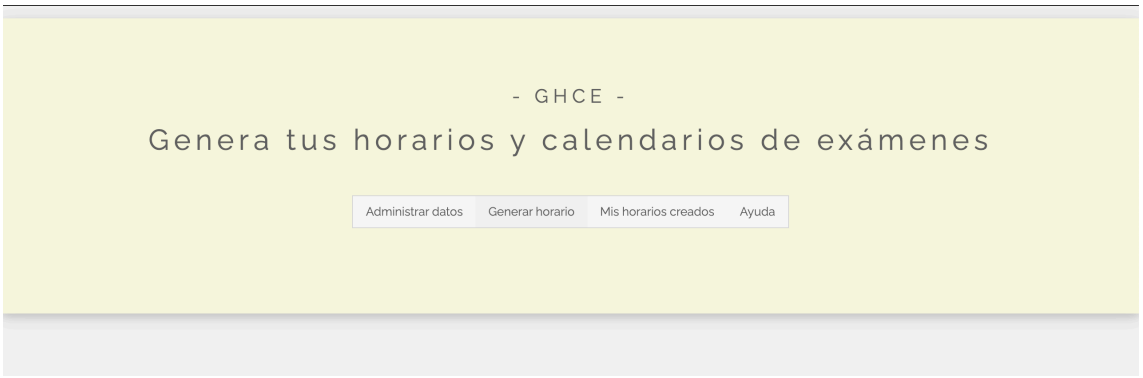


Ilustración 31 – Página de inicio

Con respecto a la tarea 7.1. se decidió añadir una página inicial que sirviese de lobby para los usuarios, desde aquí podrán elegir si acceder a la administración de los datos, la generación de horarios, sus horarios ya creados y, en el futuro, acceder al manual de usuario. En la Ilustración 31 se puede ver el aspecto final de esta nueva página inicial.

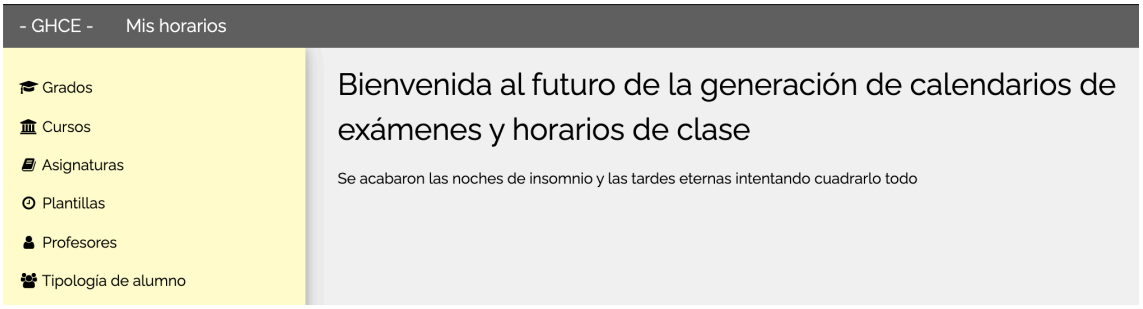


Ilustración 32 – Página para la administración de datos

En la Ilustración 32, podemos ver efectivos los cambios relacionados con la terminología, tarea 7.5., a partir de esto momento no hablaremos más de "horarios" sino de plantillas horarias y en lugar de "alumnos" nos referiremos a tipologías de alumnos.

El resultado del desarrollo de la tarea 7.2. se puede ver reflejada en la Ilustración 33 y la Ilustración 34 que muestran los cambios efectuados en los formularios de edición y creación de asignaturas y de edición y creación de plantillas horarias, respectivamente.

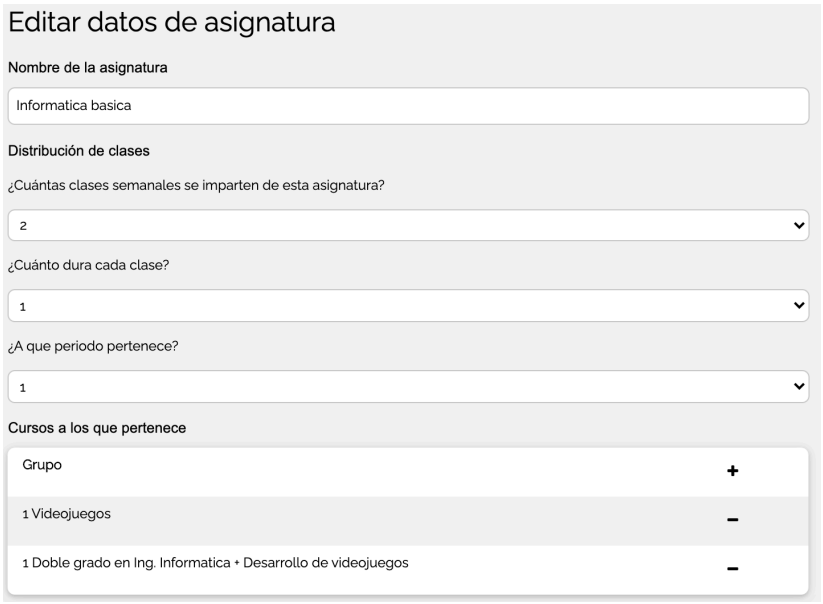


Ilustración 33 – Formulario de edición de asignaturas

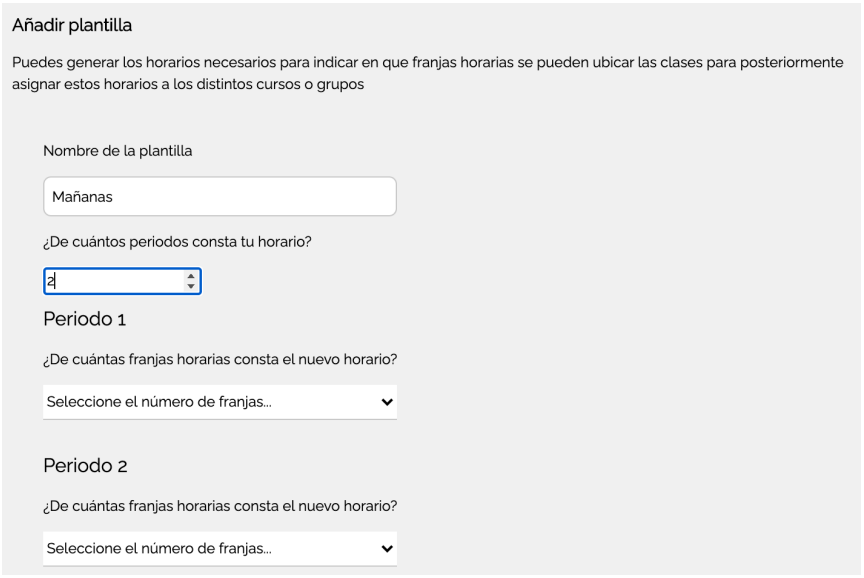


Ilustración 34 – Formulario de creación de plantillas horarias

En el primer caso solo fue necesario añadir un *dropdown* en el que el usuario puede indicar a que periodo corresponde la asignatura en cuestión.

En el segundo caso, también se añadió un *dropdown,* pero en este caso genera de forma dinámica el formulario de la plantilla *n* veces en función del número de periodos indicado. Pero finalmente acabó inhabilitado debido a la complejidad de los numerosos problemas surgidos por su implementación, se optó por no incluir esta funcionalidad por el momento y será retomada en versiones posteriores.

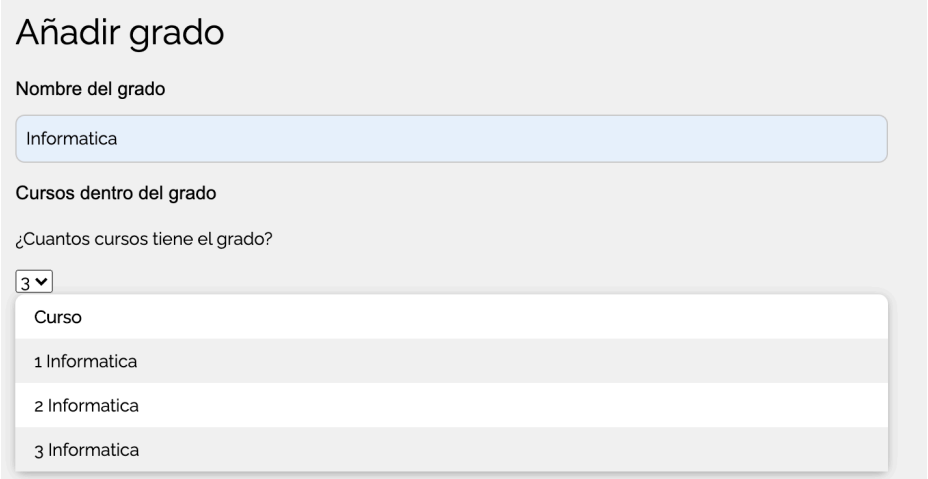


Ilustración 35 – Formulario de creación de grados

El resultado de la tarea 7.3. es el mostrado en la Ilustración 35. Esta nueva implementación permite al usuario generar automáticamente los distintos cursos de un grado, simplemente indicando el número de cursos que lo conforman. De este modo al seleccionar, por ejemplo, tres cursos se generarán estos formados por el número del curso y el nombre del grado, esto podrá ser modificado posteriormente en el apartado "Cursos".

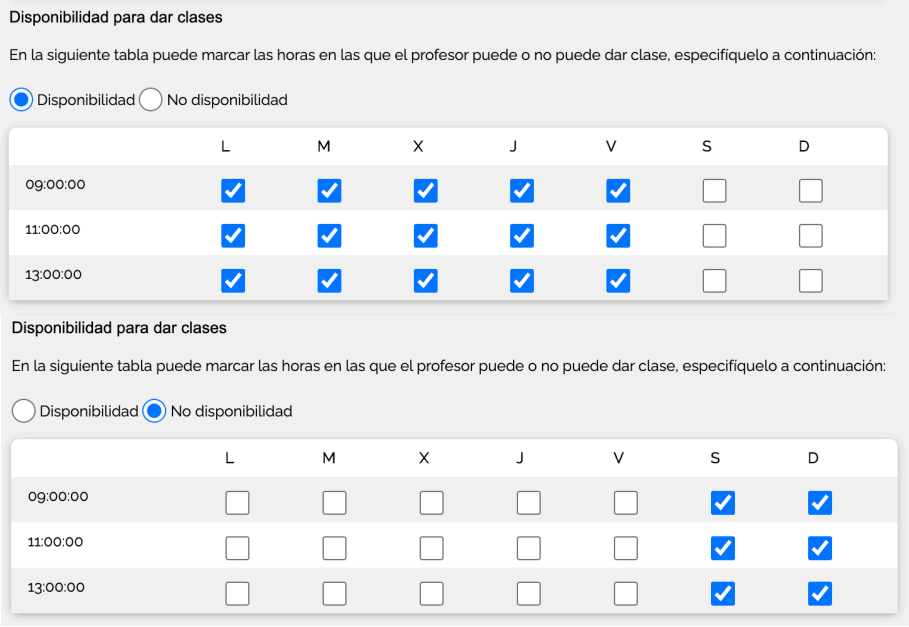


Ilustración 36 – Formulario para la selección de disponibilidad/no disponibilidad de un docente

El resultado de la tarea 7.4. se ve reflejado en la Ilustración 36 que muestra la nueva funcionalidad. En este caso el usuario puede elegir entre ingresar la disponibilidad de un docente o simplemente indicar en que periodos de tiempo no estará disponible en función de lo que sea más cómodo en cada caso.

7.6.5. Duración

Las tareas de esta iteración, junto con las pruebas con experto, estaban estimadas en 15 horas, pero el tiempo real empleado en esta iteración fue de 26 horas, debido a ciertos problemas con la implementación de los cambios, algunos de ellos requerían modificaciones en más elementos de los contemplados en un primer momento.

7.7. Iteración 8 – Estudio estadístico y optimización

7.7.1. Tareas

US 6.1 – Clase estadística

6.1.1. Estructura de datos para dar soporte a las estadísticas

6.1.2. Guardado en archivo de las estadísticas

6.1.3. Elaboración de un estudio comparativo entre los algoritmos

Refactorización y optimización del código

- *Profiling* de los algoritmos
- Refactorización necesaria

7.7.2. Desarrollo

Una vez terminado el desarrollo de los distintos algoritmos y comprobar que su funcionamiento era el esperado, se decidió añadir la "duplicidad" de las asignaturas. Las asignaturas, en su mayoría, constan de dos clases semanales que hasta el momento no se habían tenido en cuenta, por lo que la refactorización de los algoritmos se aprovechó también para incluir un indicador del número de clases que es necesario generar para cada asignatura.

Añadir esto supuso la inclusión de una nueva restricción y un nuevo método de evaluación a los ya mencionados en apartados anteriores:

R6 (CLASS_SUBJECT_COLLISION)	-1.000

- evaluateClassesSlot: método encargado de evaluar la restricción R6

Este método y esta restricción se encargan de evaluar la separación entre dos clases de la misma asignatura de forma que, al menos, no sea colocadas el mismo día de la semana, y con la intención de separarlas en la distribución semanal.

Además, durante esta iteración se llevó a cabo el desarrollo de la clase *Statistics* encargada de recoger los datos necesarios para efectuar un análisis del tiempo y los recursos empleados por cada algoritmo hasta encontrar una solución válida.

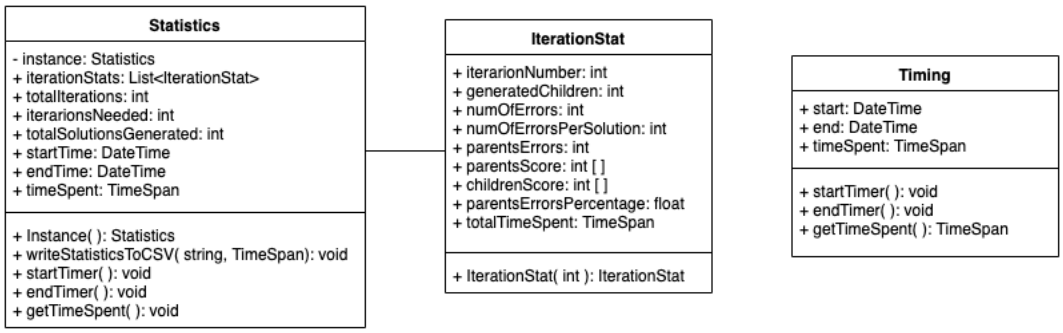


Ilustración 37 - Diagrama de clases para las estadísticas

Con el fin de poder llevar a cabo un estudio válido sobre que algoritmo es el más adecuado para resolver este problema se recogió la información que podemos observar en la Ilustración 37 por lo que los datos a comparar son:

- Número de iteraciones totales necesarias.
- Número total de soluciones generadas.
- Número de soluciones generadas en cada iteración.
- Tiempo empleado en cada iteración.
- Tiempo total empleado hasta encontrar la solución óptima.
- Puntuación media de la población padre en cada iteración.
- Puntuación media de la población hija en cada iteración.

Además de las estructuras de datos generadas para almacenar la información relevante con la ejecución de los algoritmos, se creó la clase *Timing* para poder crear diversos "cronómetros" con el fin de realizar un *profiling* y obtener información para poder llevar a cabo una mejor optimización de estos.

7.7.3. Duración

Las tareas de esta iteración, junto con la refactorización del código, estaban estimadas en 20 horas, en este caso el estudio comparativo entre los algoritmos tuvo que ser repetido en más de una ocasión debido a cambios en el uso de la CPU en cada momento por lo que el tiempo real empleado en esta iteración fue de 37 horas, más del doble del estimado.

7.8. Iteración 9 – Manual de usuario

7.8.1. Tareas

US08 – Manual de usuario

7.8.2. Desarrollo

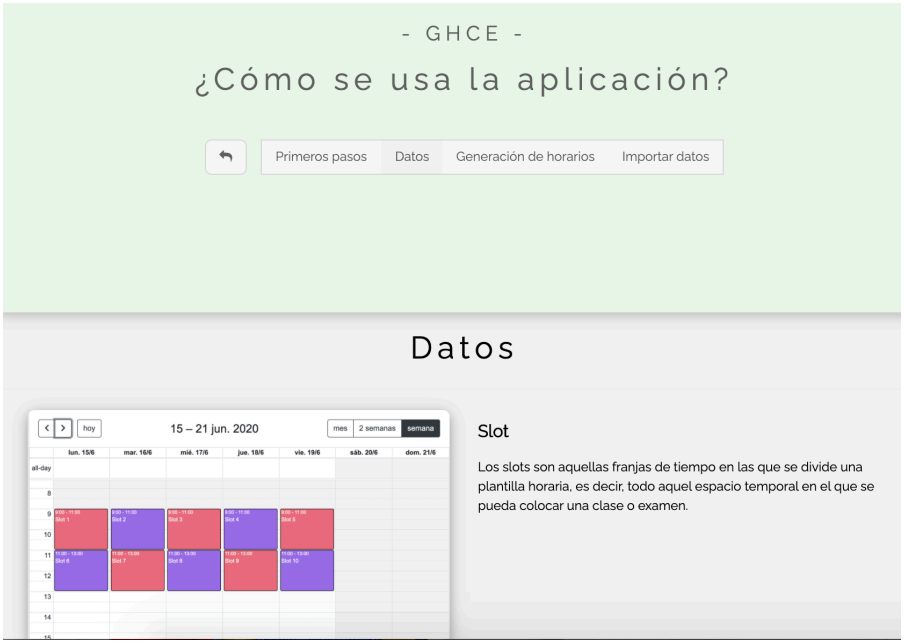


Ilustración 38 - Manual de usuario

Durante esta iteración se llevó a cabo la redacción e integración en la web principal de un manual de usuario, dividido en varias secciones. La finalidad de este manual es permitir al usuario un mayor entendimiento de los elementos y el uso que el algoritmo les da, para explotar al máximo esta herramienta y ajustarla correctamente a sus necesidades.

7.8.3. Duración

Las tareas de esta iteración estaban estimadas en 10 horas y el tiempo real empleado en esta iteración fue de 11 horas, una pequeña desviación con el tiempo estimado.

8. Resultados de la aplicación

A continuación, los tres algoritmos implementados en este proyecto se ejecutaron 20 veces con el mismo conjunto de datos y con los mismos parámetros para poder así realizar un estudio de cuál es más efectivo.

Para este estudio de resultados se utiliza un conjunto de datos mediano, qué trata dentro de lo posible de simular un escenario real con los siguientes datos:

- 3 grados universitarios, donde uno de ellos es un doble grado formado por una combinación de los otros dos.
- 13 cursos, distribuidos en 4, 4, y 5 para cada grado.
- 34 asignaturas, algunas de ellas compartidas entre 2 o todos los grados.
- 13 tipologías de alumnos, uno para cada curso (sin alumnos repetidores).
- 24 profesores, repartidos entre los 3 grados.
- 2 plantillas horarias, una de mañana para los primeros cursos y otra de tardes para los últimos cursos de cada grado.

Los parámetros comunes que se establecieron fueron:

- Tamaño de la población padre: 10
- Número de mutaciones aleatorias por solución: 1
- Número de iteraciones limite: 1.000
- Número máximo de errores a arreglar por solución: 10 (Solo afecta al algoritmo de búsqueda dirigido)

Además, cada uno de los algoritmos fue ejecutado, 20 veces, en el mismo ordenador y con la misma capacidad de CPU en cada una de ellas.

8.1. Algoritmo de búsqueda aleatorio

8.1.1. Puntuación media

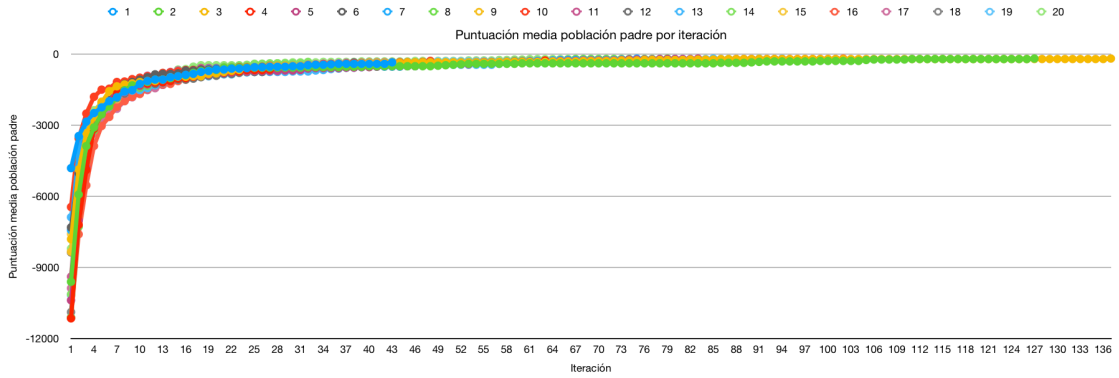


Ilustración 39 - Puntuación media población padre por iteración algoritmo de búsqueda aleatorio

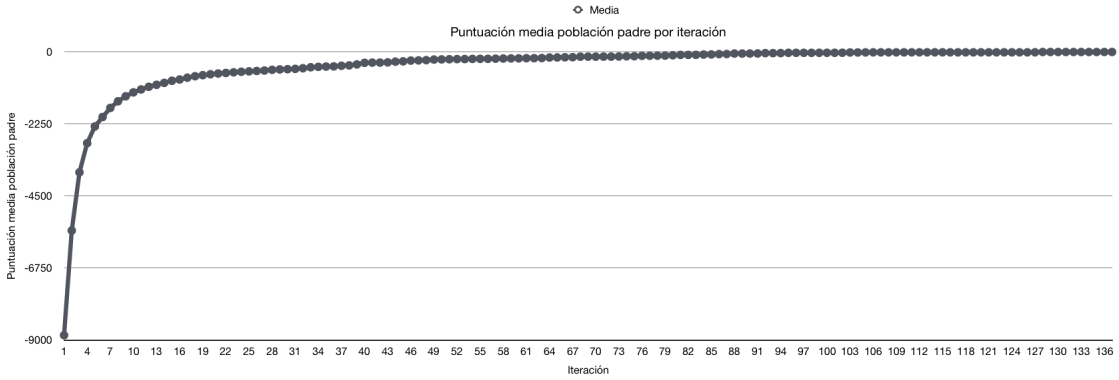


Ilustración 40 - Media de puntuaciones medias población padre por iteración algoritmo de búsqueda aleatorio

En la Ilustración 39, se muestra un gráfico que indica el valor medio de la puntuación de la población padre en cada una de las iteraciones de las 20 ejecuciones del algoritmo. En todos los gráficos las puntuaciones han sido expresadas en valor absoluto para facilitar su representación, siendo 0 la puntuación óptima.

En la Ilustración 40, se muestra la media aritmética resultante de las 20 ejecuciones mostradas en la Ilustración 39. De esta forma se puede apreciar mejor que la puntuación media de la población se va aproximando a 0 de forma exponencial. Esto concuerda con el funcionamiento del algoritmo ya que en cada iteración se eligen las 10 mejores soluciones y por tanto en cada iteración la puntuación de la población padre debería mejorar o al menos mantenerse. En este caso se puede ver que durante las primeras iteraciones la puntuación se acerca al valor óptimo aceleradamente, pero luego se ralentiza.

8.1.2. Mejora por iteración

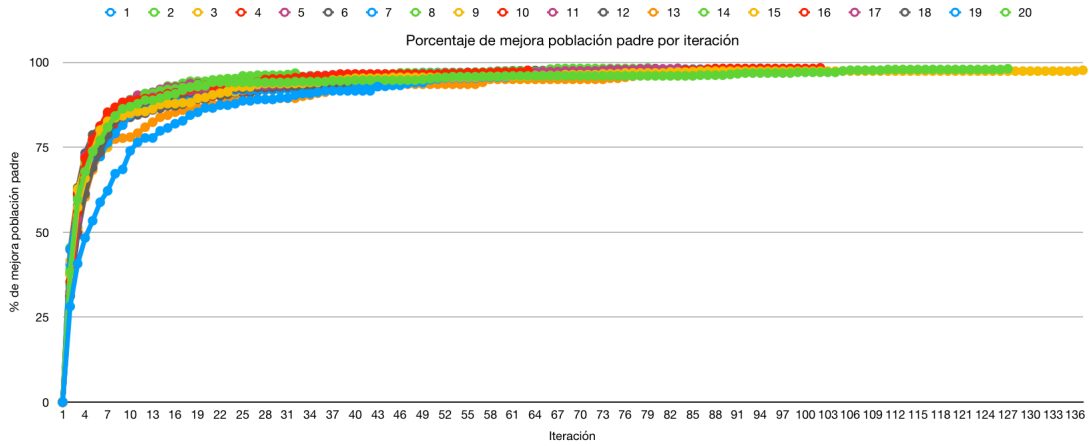


Ilustración 41 - Porcentaje de mejora población padre por iteración algoritmo de búsqueda aleatorio

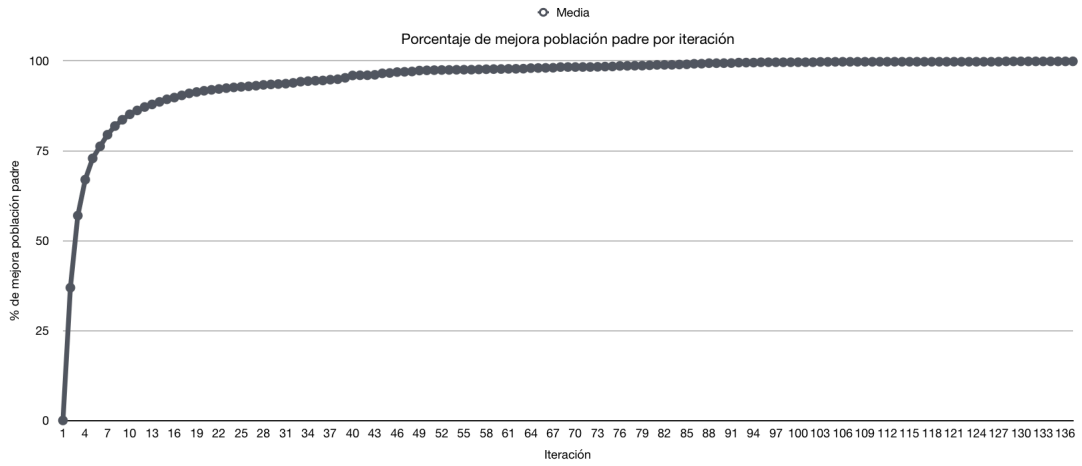


Ilustración 42 - Media de los porcentajes de mejora de la población padre algoritmo de búsqueda aleatorio

En la Ilustración 41, se muestra un gráfico indica el porcentaje de mejora de la puntuación media de los padres en cada iteración, en cada una de las 20 ejecuciones del algoritmo. este gráfico está estrechamente relacionado con el mencionado en la subsección 8.1.1 donde pudimos ver que por la naturaleza del algoritmo la puntuación mejoraba o se mantenía en cada iteración de esta misma manera podemos apreciar que el porcentaje de mejora sigue este mismo patrón, aunque con la forma de una función logarítmica debido a que en este caso cada vez nos aproximamos más al 100% en lugar de al cero.

En la Ilustración 42, se muestra la curva resultante de calcular la media aritmética de las 20 ejecuciones anteriormente descritas. De esta forma se puede apreciar con mayor detalle la evolución a lo largo de las iteraciones.

8.1.3. Tiempo de ejecución e iteraciones necesarias

Ilustración 43 - Tiempo total de ejecución algoritmo de búsqueda aleatorio

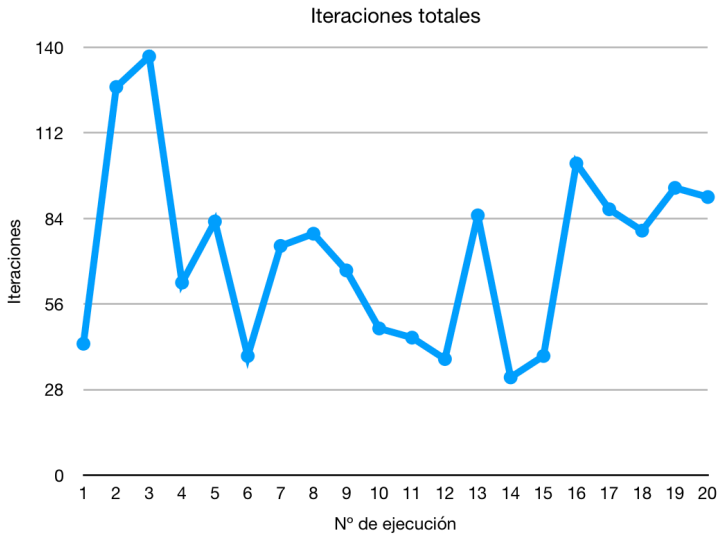


Ilustración 44 - Iteraciones totales algoritmo de búsqueda aleatorio

En la Ilustración 43 y la Ilustración 44, se representan el gráfico que recoge el tiempo total empleado y el gráfico que muestra el número total de iteraciones necesarias, respectivamente, en cada una de las ejecuciones del algoritmo.

Ambos gráficos están estrechamente relacionados debido a que a mayor número de iteraciones mayor será el tiempo de ejecución necesario, por eso podemos apreciar que ambos gráficos son prácticamente idénticos. Además, debido a la naturaleza "aleatoria" de este algoritmo la representación del número de iteraciones y, por ende, el tiempo de ejecución es tan dispersa.

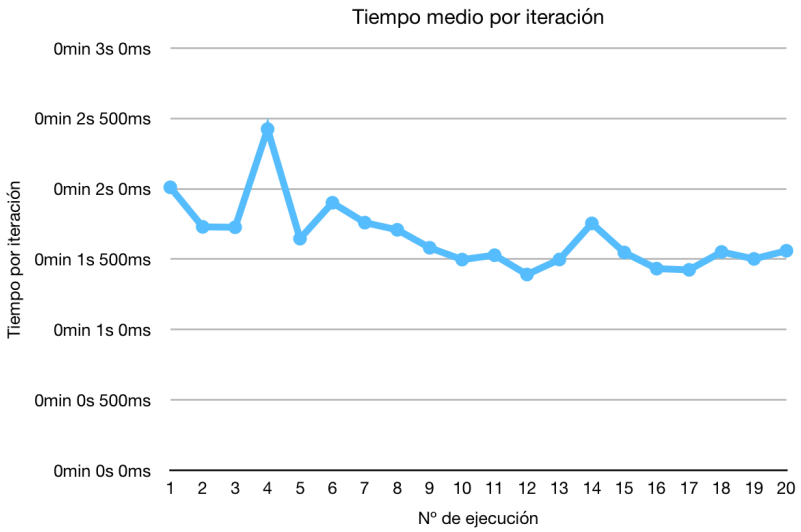


Ilustración 45 - Tiempo medio por iteración algoritmo de búsqueda aleatorio

En el gráfico mostrado en la Ilustración 45 se muestra la media de tiempo necesaria por cada iteración en cada una de las 20 ejecuciones del algoritmo. cómo podemos apreciar el tiempo oscila entre el segundo y medio y los dos segundos y medio, por lo que la diferencia es mínima.

8.1.4. Población total generada

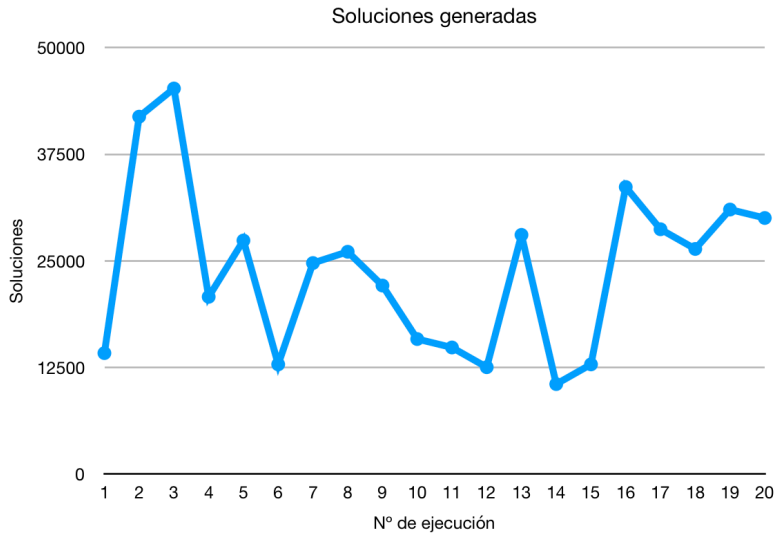


Ilustración 46 - Soluciones totales generadas algoritmo de búsqueda aleatorio

En el gráfico representado en la Ilustración 46, se muestran los datos recogidos durante las 20 ejecuciones del algoritmo con relación al número de soluciones generadas en cada una de ellas. Como ya se ha expresado con anterioridad la naturaleza aleatoria de este algoritmo provoca una gran dispersión, este valor oscila entre las 11.000 y las 45.000 soluciones durante las ejecuciones realizadas.

8.2. Algoritmo de búsqueda genético

8.2.1. Puntuación media

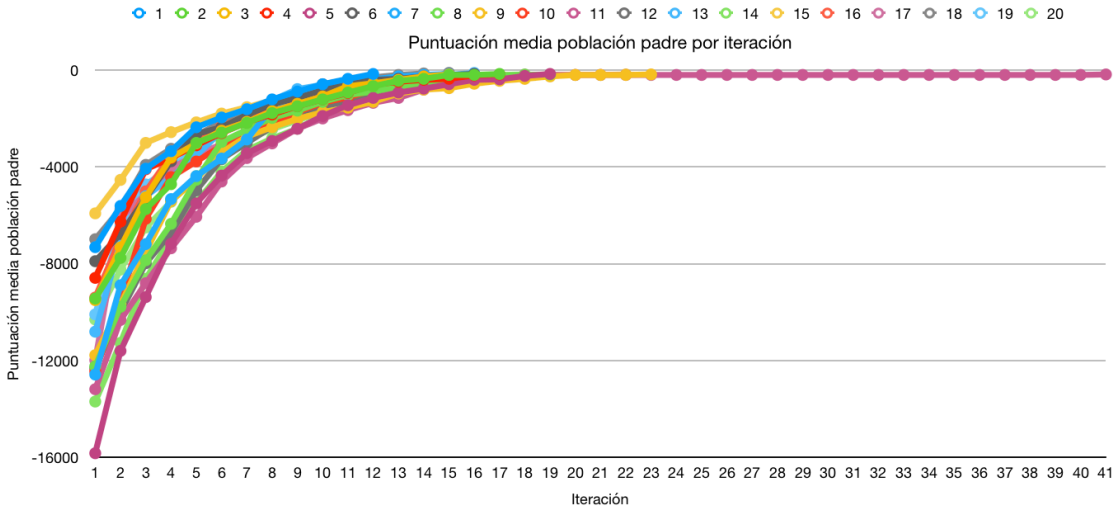


Ilustración 47 - Puntuación media población padre por iteración algoritmo de búsqueda genético

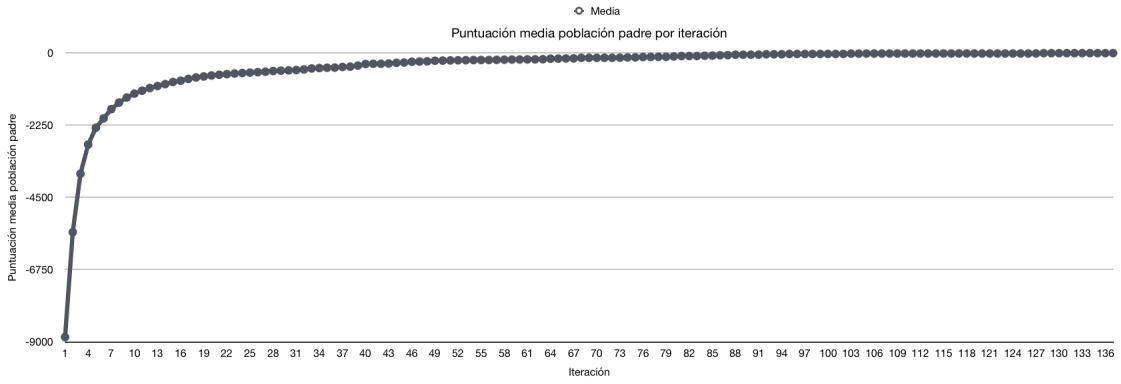


Ilustración 48 - Media de puntuaciones medias población padre por iteración algoritmo de búsqueda genético

En la Ilustración 47 se muestra el gráfico que recoge el valor medio de la puntuación de la población padre en cada una de las iteraciones de las 20 ejecuciones del algoritmo.

Asimismo, en la Ilustración 48, se muestra la media aritmética de los datos mostrados en la Ilustración 47. En estos gráficos podemos apreciar que la puntuación media de la población se va aproximando a 0 de forma exponencial, concordando con la implementación del algoritmo, donde en cada iteración la puntuación de la población padre debe mejorar o al menos mantenerse.

8.2.2. Mejora por iteración

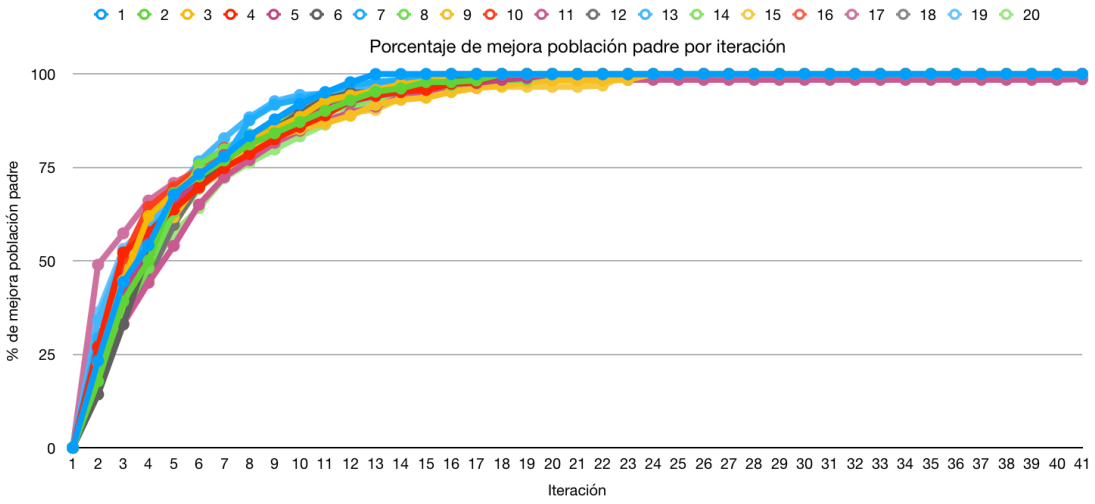


Ilustración 49 - Porcentaje de mejora población padre por iteración algoritmo de búsqueda genético

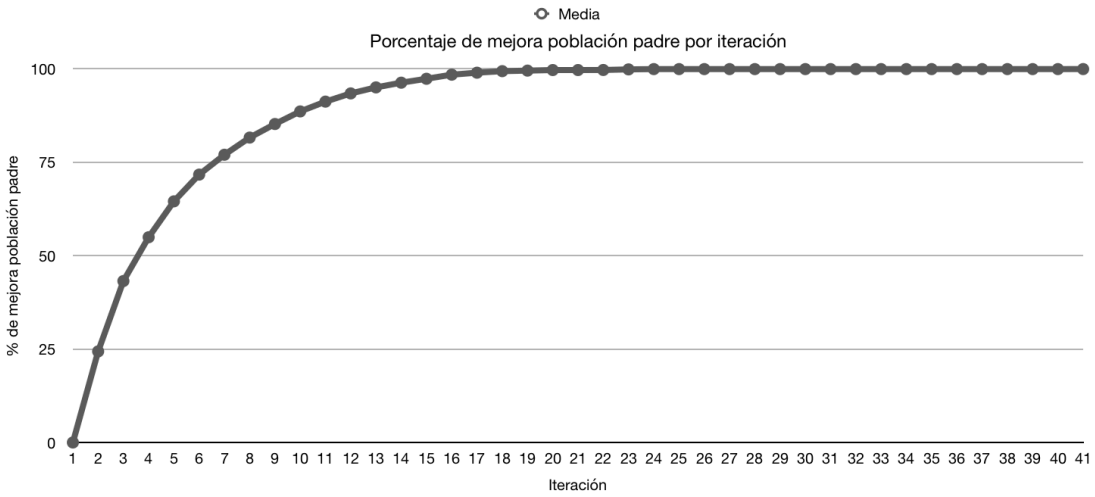


Ilustración 50 – Media de porcentajes de mejora población padre por iteración algoritmo de búsqueda genético

En el gráfico mostrado en la Ilustración 49 se representa el porcentaje de mejora de la puntuación de la población padre en cada una de las iteraciones de las 20 ejecuciones realizadas del algoritmo.

Para facilitar la interpretación de estos datos, en la Ilustración 50 se muestra la media aritmética de los datos representados en la ilustración anterior. Estos gráficos están estrechamente relacionados con los mencionados en la subsección 8.2.1.

8.2.3. Tiempo de ejecución e iteraciones necesarias

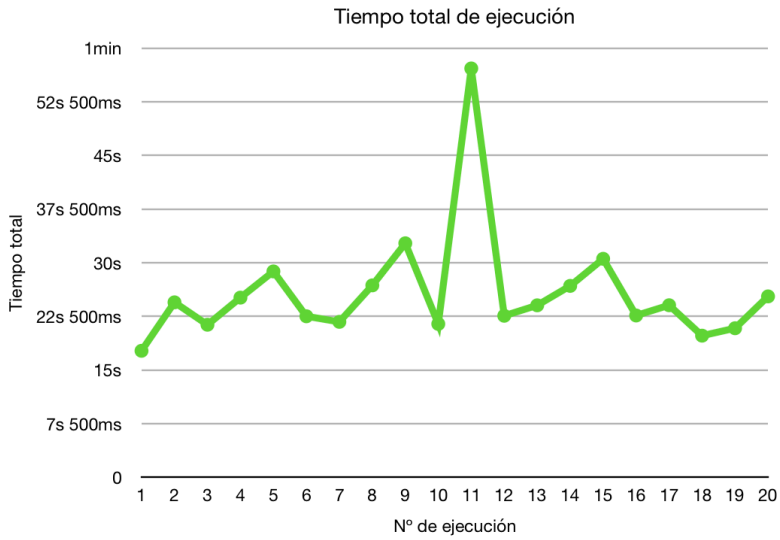


Ilustración 51 - Tiempo total de ejecución algoritmo de búsqueda genético

Ilustración 52 - Iteraciones totales necesarias algoritmo de búsqueda genético

En la Ilustración 51 y la Ilustración 52, se representan el gráfico que recoge el tiempo total empleado y el gráfico que muestra el número total de iteraciones necesarias, respectivamente, en cada una de las ejecuciones del algoritmo.

Ambos gráficos están estrechamente relacionados debido a que a mayor número de iteraciones mayor será el tiempo de ejecución necesario, por eso podemos apreciar que ambos gráficos son muy similares. Por lo general los datos se encuentran bastante agrupados entre las 10 o 20 iteraciones, a excepción de la ejecución número 11 que necesitó por encima de 40.

8.2.4. Iteraciones / tiempo

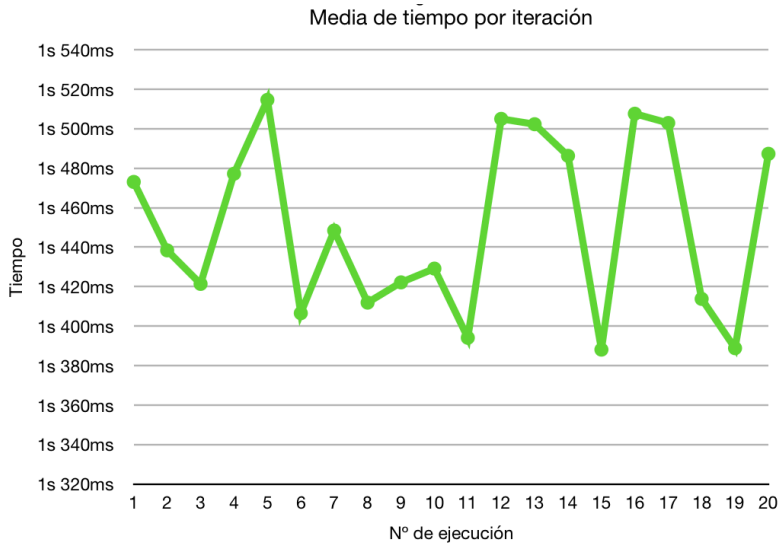


Ilustración 53 – Media de tiempo por iteración algoritmo de búsqueda genético

En este caso, aunque debido a la escala del gráfico no es tan perceptible visualmente, los valores son bastante estables siendo la diferencia entre ellos de tan solo 200 milisegundos.

8.2.5. Población total generada

Ilustración 54 - Soluciones totales generadas algoritmo de búsqueda genético

En el gráfico representado en la Ilustración 54, se muestran el número de soluciones generadas en cada una de las 20 ejecuciones del algoritmo. Cómo podemos apreciar y con relación a lo anteriormente descrito en la subsección 8.2.3 estos valores son medianamente estable comprendidos entre las 3.500 y las 7.000 soluciones a excepción de la ejecución número 11 que como ya explicamos tardo más de lo habitual y por ende género más soluciones de lo normal.

8.3. Algoritmo de búsqueda dirigido

8.3.1. Puntuación media

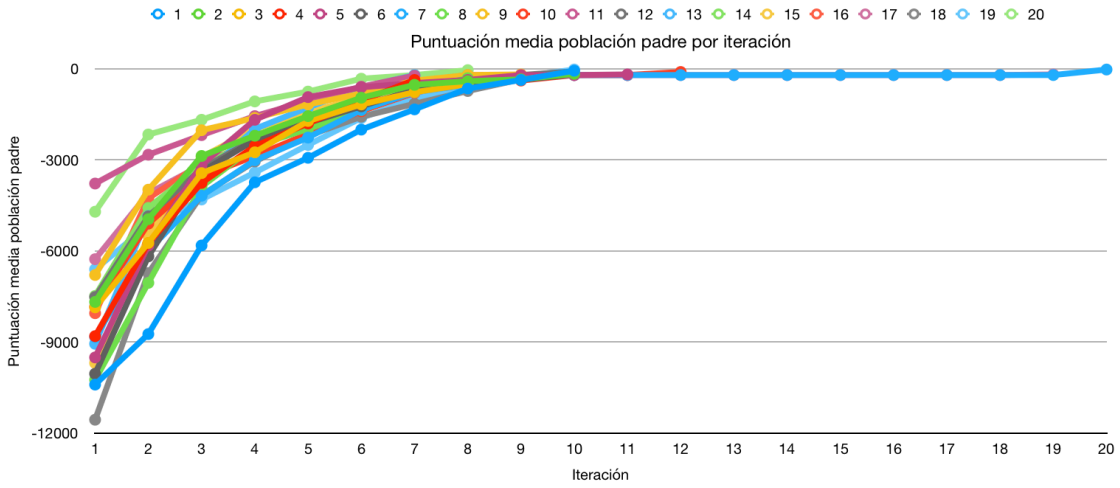


Ilustración 55 - Puntuación media población padre por iteración algoritmo de búsqueda dirigido

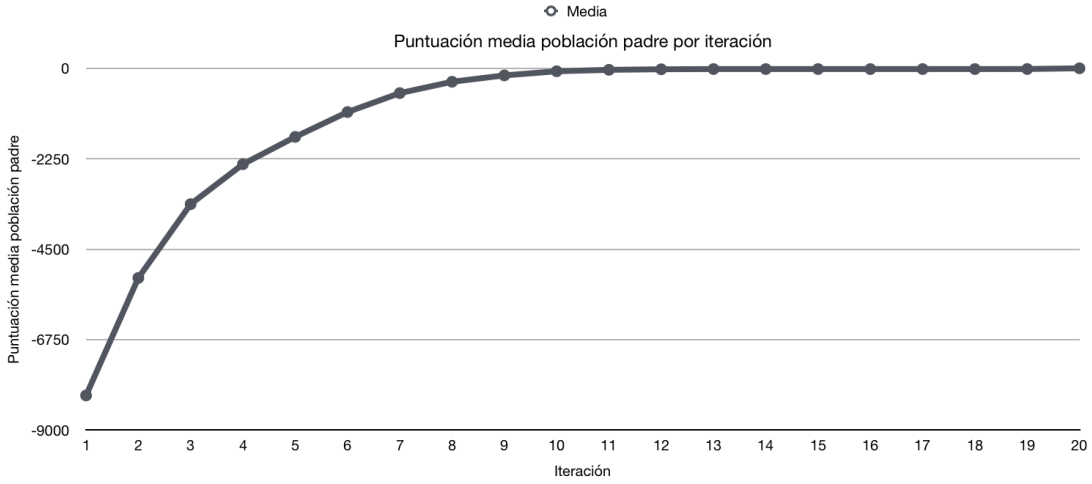


Ilustración 56 - Media de puntuaciones medias población padre por iteración algoritmo de búsqueda dirigido

En la Ilustración 55 se muestra el gráfico que recoge el valor medio de la puntuación de la población padre en cada una de las iteraciones de las ejecuciones del algoritmo.

Asimismo, en la Ilustración 56, se muestra la media aritmética de los datos mostrados en la Ilustración 55. En estos gráficos podemos apreciar que la puntuación media de la población se va aproximando a 0 de forma exponencial, concordando con la implementación del algoritmo, donde en cada iteración la puntuación de la población padre debe mejorar o al menos mantenerse.

8.3.2. Mejora por iteración

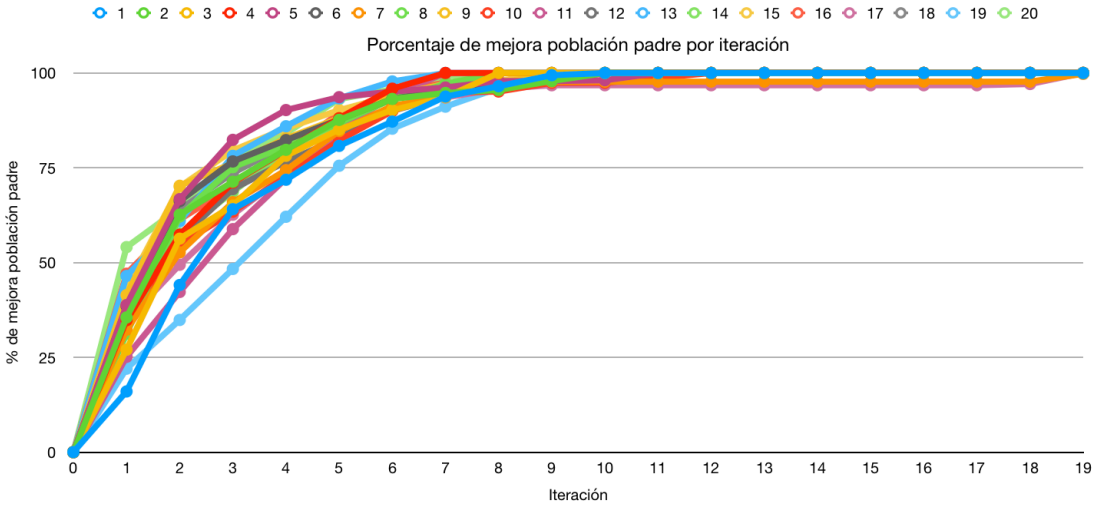


Ilustración 57 - Porcentaje de mejora población padre por iteración algoritmo de búsqueda aleatorio

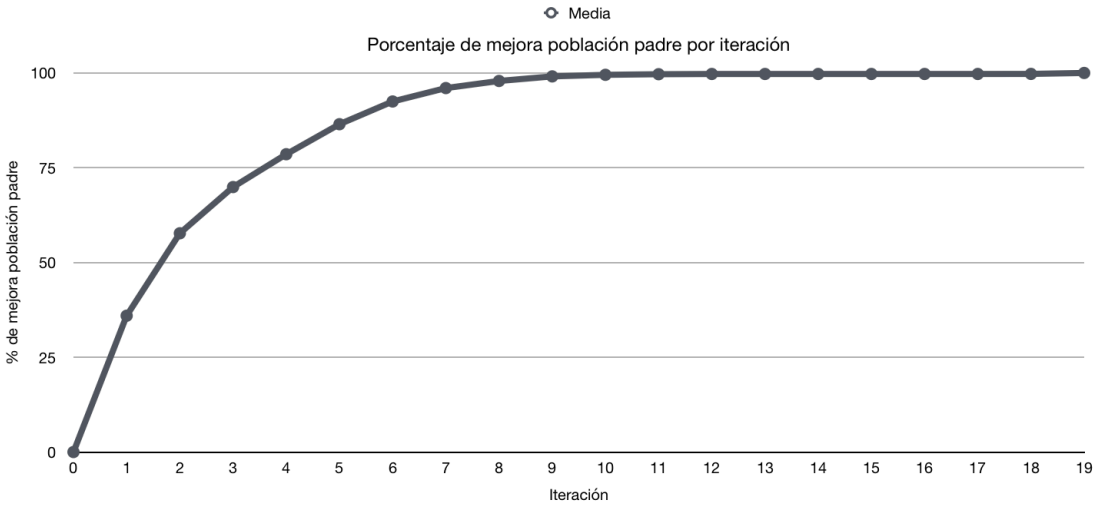


Ilustración 58 – Media de porcentajes de mejora población padre por iteración algoritmo de búsqueda genético

En el gráfico mostrado en la Ilustración 57 se representa el porcentaje de mejora de la puntuación de la población padre en cada una de las iteraciones de las ejecuciones realizadas del algoritmo.

Para facilitar la interpretación de estos datos, en la Ilustración 58 se muestra la media aritmética de los datos representados en la ilustración anterior. Estos gráficos están estrechamente relacionados con los mencionados en la subsección 8.3.1.

8.3.3. Tiempo de ejecución e iteraciones necesarias

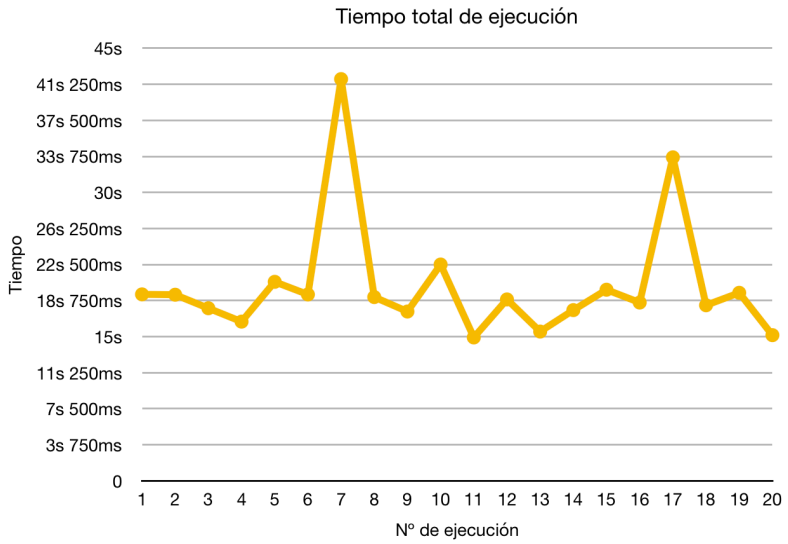


Ilustración 59 - Tiempo total de ejecución algoritmo de búsqueda dirigido

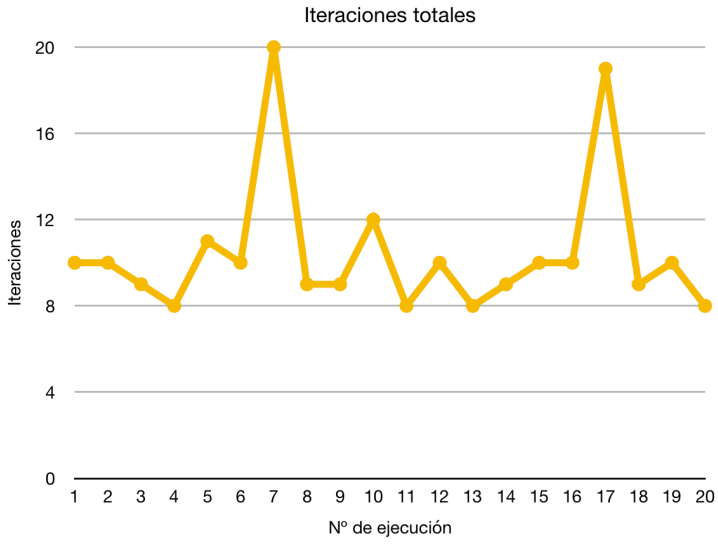


Ilustración 60 - Iteraciones totales algoritmo de búsqueda dirigido

En la Ilustración 59 y la Ilustración 60, se representan el gráfico que recoge el tiempo total empleado y el gráfico que muestra el número total de iteraciones necesarias, respectivamente, en cada una de las ejecuciones del algoritmo.

Ambos gráficos están relacionados debido a que el número de iteraciones y el tiempo de ejecución necesario están intrínsecamente relacionados, de ahí su forma tan idéntica. Por lo general los datos se encuentran bastante agrupados entre las 8 o 12 iteraciones, a excepción de las ejecuciones número 7 y 17 que necesitaron casi 20, que aún así no es un valor alejado.

8.3.4. Iteraciones / tiempo

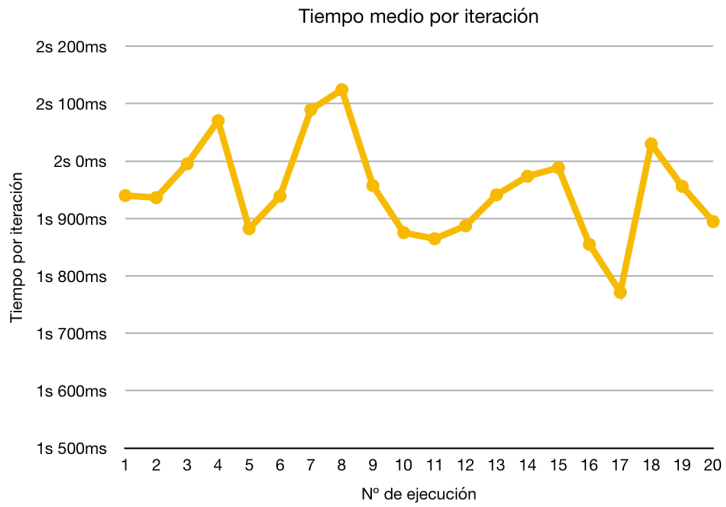


Ilustración 61 - Tiempo medio por iteración algoritmo de búsqueda dirigido

El tiempo medio de ejecución de cada iteración se muestra en la Ilustración 61, donde es visible que el conjunto de datos se encuentra bastante agrupado con valores que oscilan menos de 200 milisegundos entre sí.

8.3.5. Población total generada

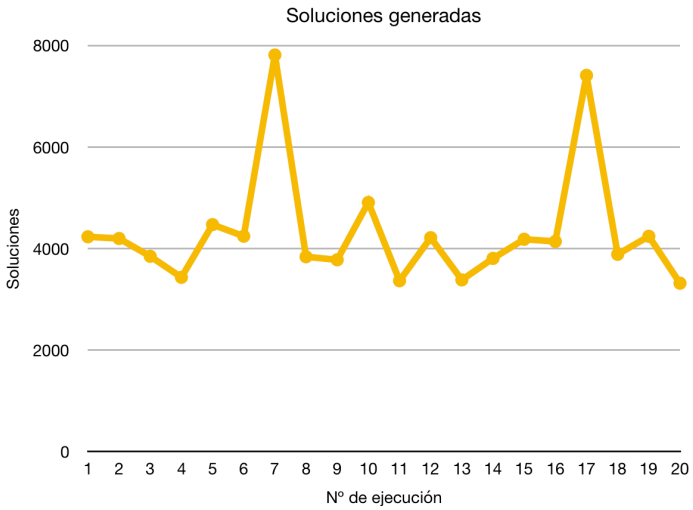


Ilustración 62 - Soluciones totales generadas algoritmo de búsqueda dirigido

En el gráfico representado en la Ilustración 62, se muestran el número de soluciones generadas en cada una de las ejecuciones del algoritmo. Cómo podemos apreciar y en relación con lo anteriormente descrito en la subsección 8.3.3 estos valores son medianamente estable comprendidos entre 3.500 y 5.000 soluciones, a excepción de las ejecuciones 7 y 17 que como ya se mencionó tuvieron un mayor tiempo de ejecución y, por ende, generaron más soluciones.

8.4. Comparativa

8.4.1. Puntuación media

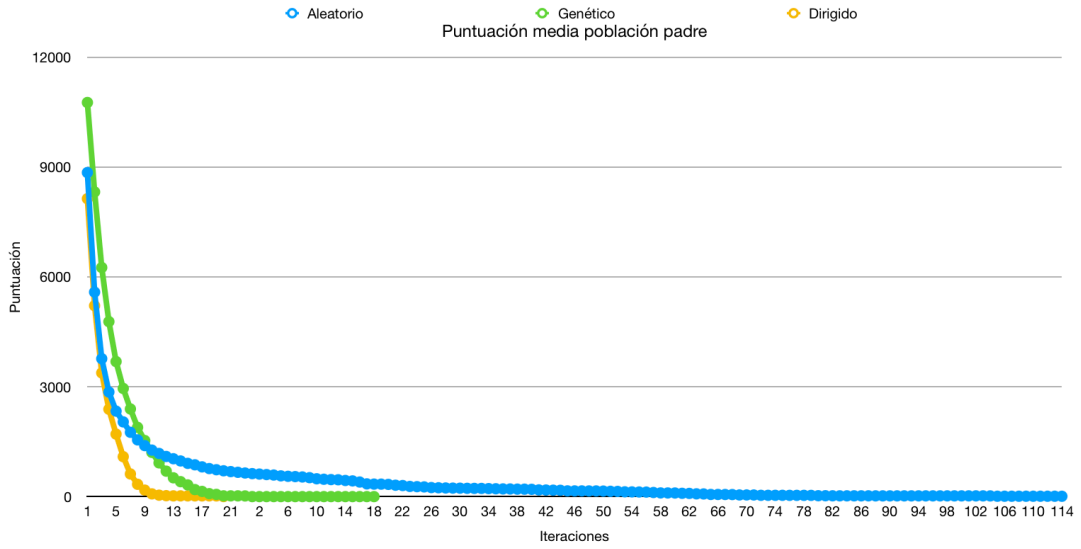


Ilustración 63 - Comparativa puntuación media población padre algoritmos de búsqueda

En la Ilustración 63 podemos ver las curvas de los distintos algoritmos y comparar así la pendiente de cada uno. Como se mencionó en los apartados concretos de cada algoritmo, la forma que posee la curva de la puntuación media de la población padre es exponencial, con un gran índice de mejora en las primeras iteraciones y ralentizándose al final hasta encontrar la solución óptima.

En este caso, podemos apreciar que el algoritmo de búsqueda aleatorio (azul) cuenta con una pendiente bastante pronunciada, muy similar a la del algoritmo de búsqueda dirigido, aunque en iteraciones aún bastante tempranas la pendiente disminuye su inclinación provocando una ralentización en la búsqueda de soluciones con mejor puntuación, lo que provoca que sea la curva más larga del gráfico.

Por otro lado, la curva del algoritmo de búsqueda genético (verde) es la que menos pendiente presenta de las tres, en las primeras iteraciones, pero se mantiene durante más tiempo que la curva del algoritmo de búsqueda aleatorio y finalmente, alcanza la curva del algoritmo de búsqueda dirigido.

Por último, la curva del algoritmo de búsqueda dirigido (amarillo) es la que mayor pendiente presenta y también es la que más se mantiene durante las distintas iteraciones, resultando en la curva más corta del gráfico.

8.4.2. Mejora por iteración

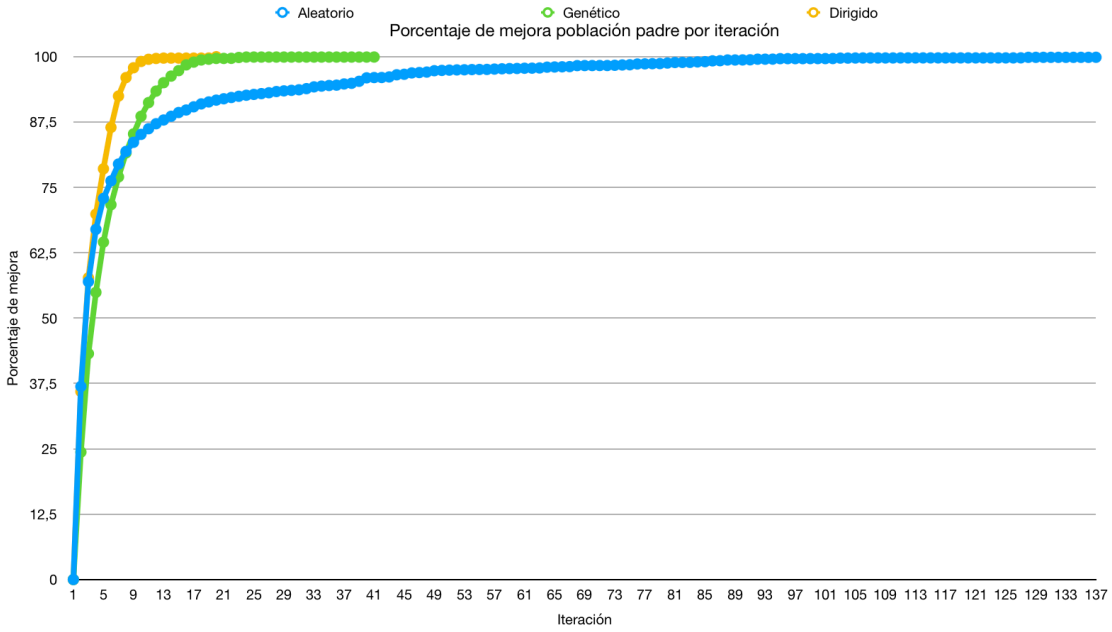


Ilustración 64 - Comparativa porcentaje de mejora población padre algoritmos de búsqueda

La Ilustración 64 muestra las curvas de los distintos algoritmos con relación al porcentaje de mejora de sus respectivas poblaciones padres, que como hemos mencionado en apartados anteriores está estrechamente relacionada con la gráfica de la Ilustración 63.

Podemos apreciar que estas tres curvas representadas cuentan con las mismas características que las curvas del aparatado 8.4.1. De esta forma, podemos apreciar que el algoritmo de búsqueda aleatorio (azul) cuenta con una pendiente bastante pronunciada, muy similar a la del algoritmo de búsqueda dirigido, aunque en iteraciones aún bastante tempranas la pendiente disminuye su inclinación provocando una ralentización en la búsqueda de soluciones con mejor puntuación, lo que provoca que sea la curva más larga del gráfico.

Al igual que en el apartado anterior podemos apreciar que la curva del algoritmo de búsqueda genético (verde) es la que menos pendiente presenta, al principio, manteniéndose nuevamente más tiempo que la curva del algoritmo de búsqueda aleatorio (azul) y finalmente, alcanzando la curva del algoritmo de búsqueda dirigido (amarillo). También se repite la tendencia con la curva del algoritmo de búsqueda dirigido (amarillo) teniendo así, una vez más, la mayor pendiente y la menor duración de todas.

8.4.3. Tiempo de ejecución e iteraciones necesarias

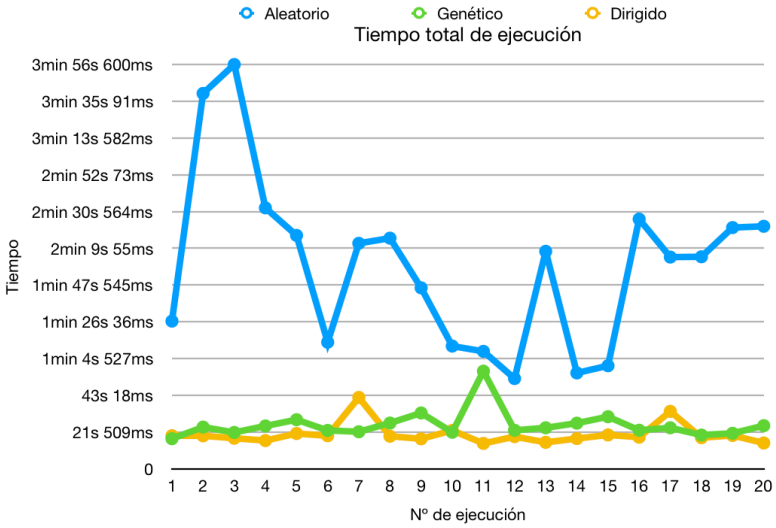


Ilustración 65 - Comparativa tiempo total de ejecución algoritmos de búsqueda

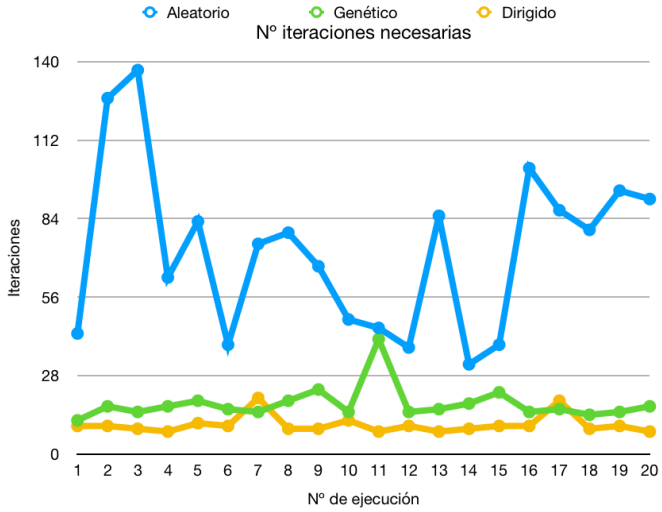


Ilustración 66 - Comparativa iteraciones totales necesarias algoritmos de búsqueda

En la Ilustración 65 y la Ilustración 66, se representan el gráfico que recoge el tiempo total empleado y el gráfico que muestra el número total de iteraciones necesarias, respectivamente, en cada una de las ejecuciones de los algoritmos.

En este caso, podemos apreciar que el algoritmo de búsqueda aleatorio presenta un conjunto de datos muy disperso y en su mayoría bastante superiores al del resto de algoritmos.

Por otro lado, el conjunto de datos de los algoritmos de búsqueda genético y dirigido es más estable, siendo este último el que emplea una menor cantidad de tiempo e iteraciones.

8.4.4. Iteraciones / tiempo

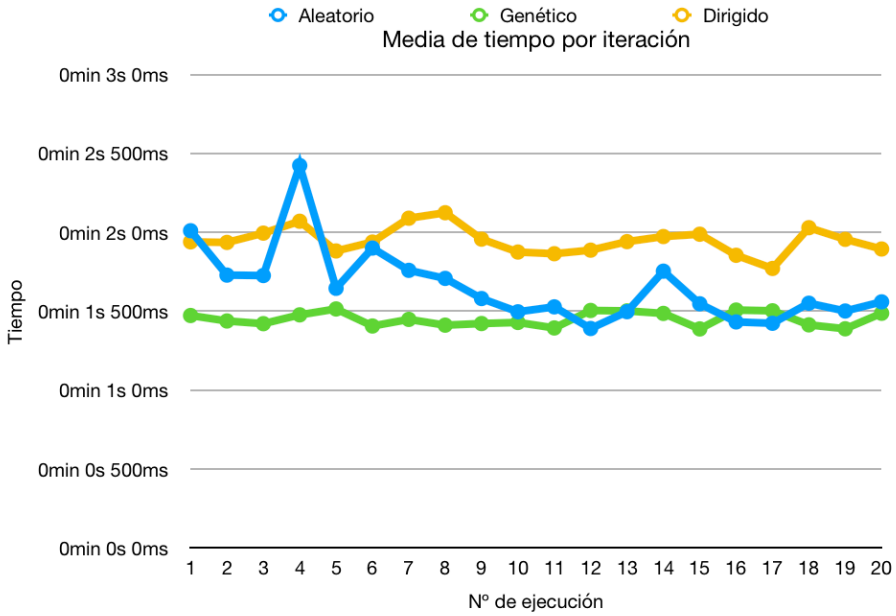


Ilustración 67 - Comparativa tiempo medio por iteración algoritmos de búsqueda

En el gráfico mostrado en la Ilustración 67, se recogen los valores medios del tiempo empleado por iteración de cada uno de los algoritmos.

En contraste con los gráficos anteriores, el algoritmo de búsqueda dirigido (amarillo) es, en este caso, el que presenta un mayor tiempo de ejecución por iteración, debido a que por construcción realiza más cálculos a la hora de generar su población para de esta forma mutar de forma controlada.

Además, podemos apreciar que el algoritmo de búsqueda genético es el que presenta un menor tiempo de ejecución por iteración y que el algoritmo de búsqueda aleatorio, recoge valores mixtos entre los dos algoritmos.

Por último, a pesar de todo esto las diferencias entre los tres algoritmos son de medio segundo lo cual, dependiendo del número de iteraciones totales, es prácticamente imperceptible.

8.4.5. Población total generada

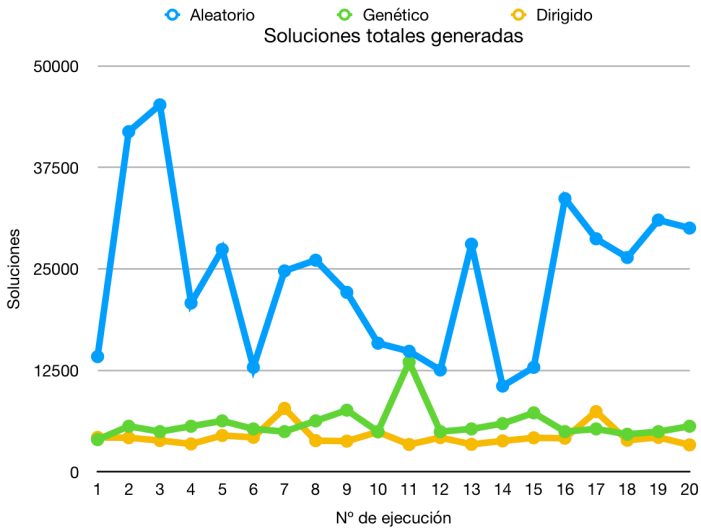


Ilustración 68 - Comparativa soluciones totales generadas algoritmos de búsqueda

En última instancia, el gráfico mostrado en la *Ilustración 68,* representa los valores recogidos con relación al tamaño de la población total generada por cada uno de estos algoritmos hasta encontrar una solución óptima.

Una vez más debido a su aleatoriedad el algoritmo de búsqueda aleatorio (azul) presenta un conjunto de valores muy dispersos comprendidos entre 12.500 y 45.000 soluciones generadas, siendo el algoritmo que más soluciones genera, con diferencia.

Por otro lado, los algoritmos, genético (verde) y dirigido (amarillo), presentan valores muy similares entre sí, en general menores a 12.500 soluciones generadas. En este caso, el algoritmo de búsqueda dirigido es el que presenta, en general, los valores más bajos, tratándose así del que menos soluciones genera hasta encontrar una solución óptima.

8.5. Conclusión

Para poder establecer qué algoritmo de búsqueda es el más adecuado para la resolución de este tipo de problema se van a tener en cuenta los datos recogidos en la *Tabla 3* qué sirve de resumen para todos los datos analizados en el apartado de resultados.

	Algoritmo de búsqueda aleatorio	**Algoritmo de búsqueda genético**	**Algoritmo de búsqueda dirigido**
Media soluciones totales generada	23.991	5.890,5	4.334,5
Tiempo medio de ejecución	1min 59s 651ms	25s 809ms	20s 327ms
Media de iteraciones necesarias	72,7	17,85	10,45

Tabla 3 - Comparación datos relevantes de los algoritmos de búsqueda

Según estos datos el algoritmo de búsqueda dirigido resultaría ser el óptimo para resolver este problema ya que se trata del más rápido y el que menos recursos necesita, pues genera menor cantidad de soluciones e itera menos veces que los otros algoritmos.

Estos datos son el resultado de la ejecución de los algoritmos sobre un conjunto de datos relativamente pequeño, por lo que todas las ventajas del algoritmo de búsqueda dirigido aumentarían exponencialmente con conjuntos de datos más voluminosos y cuya diferencia, en este caso pequeña con los demás, puede ser crítica para encontrar una solución en un tiempo razonable.

9. Estudio económico

9.1. Desglose de costes

9.1.1. Costes materiales

Dentro de los materiales utilizados para el desarrollo de este Proyecto Fin de Grado los costes se distribuyen de la siguiente manera:

	Precio	**Vida útil**	**Tiempo de uso**	**Coste real**
Macbook pro 13"	1.200€	5 años [11]	2 meses	60€
Periféricos	60€	6 años [12]	2 meses	1,67 €
Monitor 25"	109€	30.000 horas [13]	348 horas	1,27€
			Total	**62,94€**

Tabla 4 - Costes materiales

9.1.2. Costes humanos

En la Tabla 5, se recogen los costes derivados del personal necesario para la implementación del proyecto, en función de los roles desempeñados y el tiempo en horas invertido en cada fase del desarrollo:

	Precio / Hora	**Horas empleadas**	**Coste total**
Diseñador	15€/hora	22	330€
Desarrollador	20€/hora	326	6.520€
	Total	**348**	**6.850€**

Tabla 5 - Costes humanos

9.1.3. Costes de infraestructura

Durante los 2 meses en los que se desarrolla el proyecto se necesita un espacio dónde trabajar y consecuentemente esto deriva en unos costes de infraestructura que se recogen en la siguiente tabla:

	Precio / mes	**Tiempo de uso**	**Coste**
Alquiler	200€/mes	2 meses	400€
Agua	20€/mes	2 meses	40€
Luz	50€/mes	2 meses	100€
Internet	50€/mes	2 meses	100€
		Total	**640 €**

Tabla 6 – Costes de infraestructura

9.1.4. Costes totales

Finalmente podemos calcular los costes totales del desarrollo de este proyecto mediante la suma de los apartados anteriores y que podemos ver reflejado en la siguiente tabla:

Costes materiales	**Costes humanos**	**Costes infraestructura**	**Costes totales**
62,94€	6.850€	640€	7.552,94€

Tabla 7 – Costes totales

Hay que destacar qué resultado de este proyecto es sólo un prototipo y que para llegar al producto final acabado sería necesario emplear aún más tiempo, se estima, en función de las historias de usuario restantes, que se necesitarían entre 140 y 200 horas más de trabajo, que constituyen entre un 40% y un 57% más del tiempo ya empleado, por lo que el coste total real de la aplicación acabada sería aproximadamente de entre 10.574,12€ y 11.858,12€.

10. Resultados

En primer lugar, los objetivos planteados para este proyecto se han completado satisfactoriamente resultando así en una aplicación web que permite generar horarios de clases y calendarios de exámenes de forma automatizada y rápida. Siendo capaz de resolver un caso real formado por tres grados universitarios en una media de 2 minutos, en el caso del algoritmo más lento y 20 segundos, en el caso del algoritmo más rápido.

Las fases de análisis y diseño desempeñadas previamente a la implementación de la aplicación fueron de gran ayuda para el desarrollo de este proyecto, ya que gracias a ellas se construyó una base sólida sobre la que levantar el generador con cierta consistencia y coherencia.

La implementación se desarrolló con normalidad, se produjeron algunos cambios en la planificación inicial, que se muestran en la *Ilustración 69,* pero no se encontró ningún obstáculo insuperable que paralizara el transcurso del proyecto.

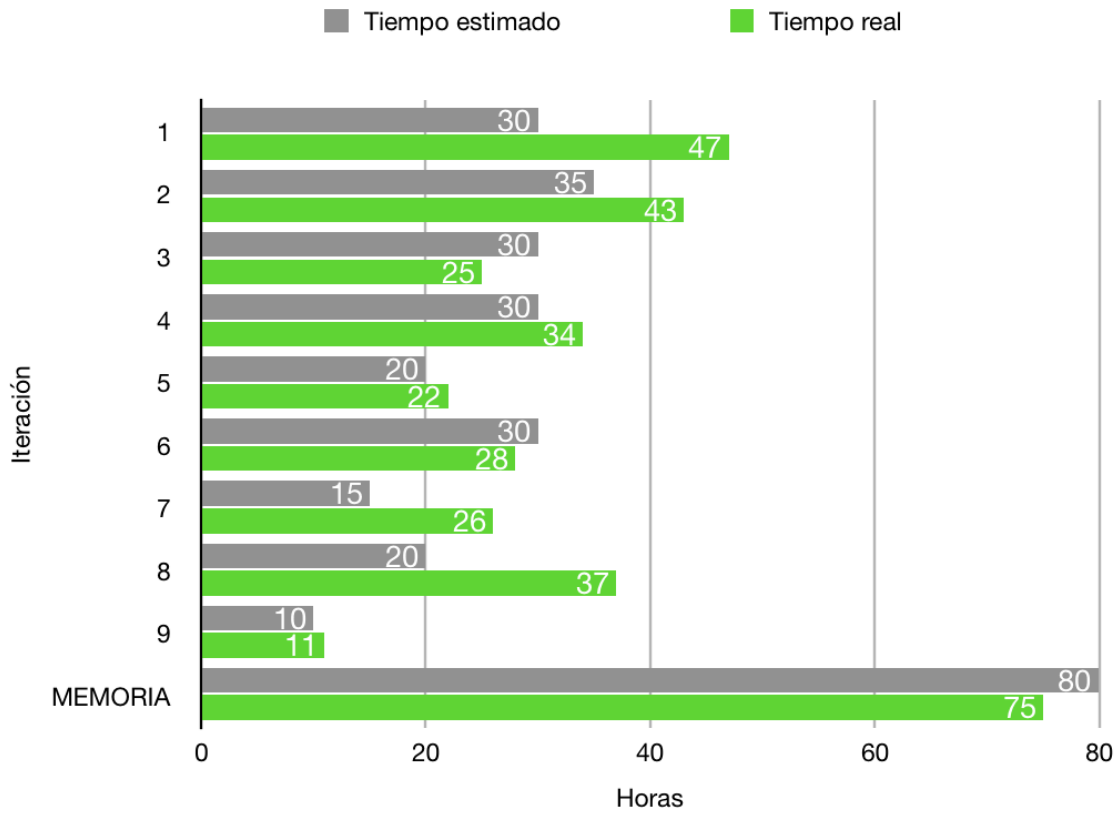


Ilustración 69 - Planificación inicial y final

Resultando el tiempo total dedicado a este proyecto de la siguiente manera, *Ilustración 70:*

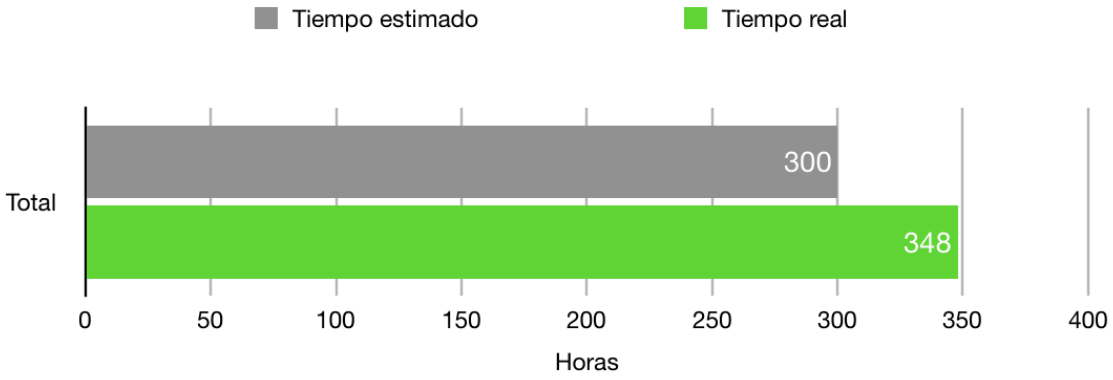


Ilustración 70 - Planificación total

Por otro lado, los resultados obtenidos en el estudio comparativo de los distintos algoritmos son los esperados, independientemente del algoritmo el problema se resuelve en menor tiempo que de forma manual, aunque cuanta más "inteligencia" se aporte al algoritmo se obtienen mejores resultados.

Además, tras la aplicación de las mejoras resultantes de las pruebas con la persona experta se obtuvieron resultados muy positivos, ya que la interfaz cumple con las expectativas y las necesidades propias de este problema.

Por último, cada uno de los algoritmos fueron optimizados para no exceder el uso de los recursos del sistema, además de esto, se añadieron algunas condiciones para que en aquellos casos en los que no sea posible encontrar una solución sin colisiones el algoritmo devuelva la mejor solución encontrada tras ciertas iteraciones sin variaciones en los resultados.

11. Conclusiones

Una vez analizados los resultados de este proyecto y siendo tan satisfactorios, concluyo que la aplicación resultante del mismo podría utilizarse en entornos educativos reales para mejorar la generación de horarios de clases, calendarios de exámenes o incluso calendarios de reuniones académicas.

Esta aplicación reduciría considerablemente el tiempo y los recursos empleados hoy en día por los distintos centros educativos en esta tarea, no obstante, este proyecto es sólo un prototipo que presenta un gran potencial y muchas opciones de mejora.

Además, es fácilmente adaptable y podría usarse en otros sectores, como podría ser el sanitario para organizar y planificar los turnos de los empleados o los horarios de los quirófanos.

Por último, este proyecto ha sido validado por una persona experta en la generación de horarios de clases, confirmando así que tanto la interfaz diseñada como los algoritmos implementados son adecuados y válidos, resultando así en una aplicación completa qué podría utilizarse perfectamente en entornos reales para la resolución de problemas reales de manera satisfactoria.

11.1. Propuestas de mejora

El prototipo desarrollado durante este proyecto mejoraría mucho la calidad del proceso de generación y planificación de horarios, pero aún podrían añadirse algunos cambios para que los resultados obtenidos tengan una calidad mayor o incluso permitir un mayor grado de personalización a la hora de configurar las restricciones y valoraciones utilizadas por el algoritmo.

Durante este proyecto se han recogido las siguientes posibles mejoras o propuestas a futuro que podrían resultar interesantes para esta aplicación:

- US 09 – Gestor de espacios, esta historia de usuario recoge la idea de incluir los distintos espacios disponibles en el centro con la finalidad de generar horarios más completos y realistas con el entorno disponible.
- US 10 – Vista múltiple, esta historia recoge la posibilidad de mostrar al usuario varias soluciones óptimas, en el caso de que sea posible, en lugar de un único resultado válido, para que sea éste quién elija el que más se ajuste a sus necesidades.

- US 11 – Editor de soluciones, en el apartado de antecedentes pudimos comparar entre las distintas soluciones del mercado y varias presentaban la opción de editar las soluciones obtenidas por parte del usuario.

- US 12 – Control en las planificaciones, esta historia de usuario recoge la idea de añadir un formulario que permita al usuario personalizar la planificación que se va a realizar, en lugar de generar una planificación completa con todos los datos introducidos, generar solo el calendario de una carrera, por ejemplo.

- US 13 – Integración con los sistemas, esta historia de usuario recoge la idea de exportar los resultados en el formato específico de las aplicaciones utilizadas por el centro educativo para el tratamiento de los horarios.

- US 14 – Algoritmo de búsqueda propio, esta historia de usuario representa la idea de añadir más parámetros con el fin de dirigir de manera más concreta la búsqueda con el fin de obtener soluciones más personalizadas.

- US 15 – Múltiples periodos, esta historia de usuario recoge el cambio C4 sugerido por la experta que no se pudo terminar de implementar por una subestimación de su complejidad.

12. Bibliografía

[1] «GHC Peñalara - Software generador de horarios para centros educativos,» Peñalara Software S.L., [En línea]. Available: https://www.penalara.com/es/ES. [Último acceso: Junio 2020].

[2] «aSc Horarios de clases,» aSc Applied Software Consultants, [En línea]. Available: https://www.asctimetables.com/timetables_es.html. [Último acceso: Junio 2020].

[3] «FET Free Timetabling Software,» Liviu Lalescu y Volker Dirr, [En línea]. Available: https://lalescu.ro/liviu/fet/. [Último acceso: Junio 2020].

[4] E. C. E. Y. E. S. E. ESPAÑOL, «Espacio Europeo de Educación Superior,» [En línea]. Available: http://www.eees.es/pdf/credito-europeo.pdf. [Último acceso: Junio 2020].

[5] «Extreme Programming - A Gentle introduction,» [En línea]. Available: www.extremeprogramming.org. [Último acceso: Junio 2020].

[6] «Extreme Programming Ingeniería de Software,» [En línea]. Available: http://ingenieriadesoftware.mex.tl/52753_xp---extreme-programing.html. [Último acceso: Junio 2020].

[7] «Trello,» [En línea]. Available: https://trello.com/. [Último acceso: Junio 2020].

[8] «Diagrama De Gantt, Microsoft Excel, Plantilla PNG,» [En línea]. Available: https://www.freepng.es/png-4tx15x/. [Último acceso: Octubre 2019].

[9] M. C. Velez Gallego, C. A. Castro Zuluaga y J. Maya Toro, «Algoritmo de búsqueda aleatoria para la programación de la producción en un taller de fabricación,» REVISTA Universidad EAFIT, 2003.

[10] [En línea]. Available: https://fullcalendar.io/. [Último acceso: Abril 2020].

[11] [En línea]. Available: https://www.ticpymes.es/tecnologia/noticias/1111246049504/vida-util-de-tecnologia-apple-puede-llegar-12-anos.1.html#:~:text=La%20vida%20%C3%BAtil%20de%20la,a%C3%B1os%20%7C%20Noticias%20%7C%20Tecnolog%C3%ADa%20%7C%20TicPymes&text=Si%20hay%20algo%20que%20ca. [Último acceso: julio 2020].

[12] [En línea]. Available: https://www.wikiversus.com/informatica/cuanto-tiempo-equivalen-millones-de-pulsaciones/. [Último acceso: julio 2020].

[13] [En línea]. Available: https://hardzone.es/2019/01/13/vida-util-monitor-antes-rompa/. [Último acceso: julio 2020].

13. Anexos

13.1. Anexo 1: Propuesta

Nombre alumno:	Esther Pérez Jerez
Titulación:	Grado en Diseño y Desarrollo de Videojuegos
Curso académico:	2019 - 2020

1. ESTUDIO COMPARATIVO DE ALGORITMOS DE BÚSQUEDA BASADOS EN INTELIGENCIA ARTIFICIAL PARA PLANIFICADOR DE HORARIOS Y EXÁMENES

Planificador de horarios y exámenes.

2. DESCRIPCIÓN Y JUSTIFICACIÓN DEL TEMA A TRATAR

El proyecto consiste en el diseño e implementación de un sistema web de planificación de horarios y/o exámenes. El proceso de desarrollo se centrará en la utilización de diversos algoritmos con el fin encontrar el más adecuado para la planificación de horarios

3. OBJETIVOS DEL PROYECTO

Creación de una aplicación web capaz de generar una organización de clases y/o exámenes.
Realización de un estudio comparativo con distintos algoritmos de búsqueda para encontrar la manera más eficiente de obtener soluciones viables, óptimas o subóptimas.
Creación de una interfaz que permita a cualquier usuario usar la aplicación anterior para generar sus propios calendarios de clases y/o exámenes.

4. METODOLOGÍA

La metodología se establecerá en las primeras fases del proyecto.

5. PLANIFICACIÓN DE TAREAS

Las tareas se definirán de acuerdo a los objetivos. Serán fijadas de forma concreta durante el desarrollo del proyecto.

6. OBSERVACIONES ADICIONALES

El tutor del proyecto será Antonio Iglesias Soria.

13.2. Anexo 2: Reuniones

REUNIÓN: 01

Fecha:		
Hora comienzo:		**Hora finalización:**
Lugar:	Universidad San Jorge	
Elabora acta:	Esther Pérez Jerez	
Convocados:	Antonio Iglesias Soria, Esther Pérez Jerez	

Orden del día / Acta

No.	**Asunto**	**Acuerdo**
1	**Disculpas por ausencia**	
2	**Aprobar última acta**	
3	**Asuntos pendientes última acta**	
4	Se tratan las historias de usuario generadas para la implementación del proyecto y se muestran los diseños iniciales derivados del análisis.	
5	Se eligen las historias de usuario para la siguiente iteración.	001
6		
7		
8		
9	**Otros asuntos**	
10	**Próxima reunión** A determinar	

Resumen de acuerdos

Número	**Acuerdo**	**Plazo**	**Responsable**
001	Implementación de la historia de usuario US0	No aplica	Esther Pérez
002			
003			
004			
005			
006			

REUNIÓN: 02

Fecha:		
Hora comienzo:		**Hora finalización:**
Lugar:	Universidad San Jorge	
Elabora acta:	Esther Pérez Jerez	
Convocados:	Antonio Iglesias Soria, Esther Pérez Jerez	

Orden del día / Acta

No.	**Asunto**	**Acuerdo**
1	**Disculpas por ausencia**	
2	**Aprobar última acta**	
3	**Asuntos pendientes última acta**	
4	Se realiza una demostración de lo desarrollado en la iteración 1.	
5	Se eligen las historias de usuario para la siguiente iteración.	001
6		
7		
8		
9	**Otros asuntos**	
10	**Próxima reunión** A determinar	

Resumen de acuerdos

Número	**Acuerdo**	**Plazo**	**Responsable**
001	Implementación de las historias de usuarios US3 y US6	No aplica	Esther Pérez
002			
003			
004			
005			
006			

REUNIÓN: 03

Fecha:		
Hora comienzo:		**Hora finalización:**
Lugar:	Universidad San Jorge	
Elabora acta:	Esther Pérez Jerez	
Convocados:	Antonio Iglesias Soria, Esther Pérez Jerez	

Orden del día / Acta

No.	**Asunto**	**Acuerdo**
1	**Disculpas por ausencia**	
2	**Aprobar última acta**	
3	**Asuntos pendientes última acta**	
4	Se realiza una demostración de lo desarrollado en la iteración 2.	
5	Se eligen las historias de usuario para la siguiente iteración.	001
6		
7		
8		
9	**Otros asuntos**	
10	**Próxima reunión** A determinar	

Resumen de acuerdos

Número	**Acuerdo**	**Plazo**	**Responsable**
001	Implementación de la historia de usuario US4	No aplica	Esther Pérez
002			
003			
004			
005			
006			

REUNIÓN: 04

Fecha:		
Hora comienzo:		**Hora finalización:**
Lugar:	Universidad San Jorge	
Elabora acta:	Esther Pérez Jerez	
Convocados:	Antonio Iglesias Soria, Esther Pérez Jerez	

Orden del día / Acta

No.	**Asunto**	**Acuerdo**
1	**Disculpas por ausencia**	
2	**Aprobar última acta**	
3	**Asuntos pendientes última acta**	
4	Se realiza una demostración de lo desarrollado en la iteración 3.	
5	Se eligen las historias de usuario para la siguiente iteración.	001
6		
7		
8		
9	**Otros asuntos**	
10	**Próxima reunión** A determinar	

Resumen de acuerdos

Número	**Acuerdo**	**Plazo**	**Responsable**
001	Implementación de las historias de usuarios US1 y US2	No aplica	Esther Pérez
002			
003			
004			
005			
006			

REUNIÓN: 05

Fecha:		
Hora comienzo:		**Hora finalización:**
Lugar:	Universidad San Jorge	
Elabora acta:	Esther Pérez Jerez	
Convocados:	Antonio Iglesias Soria, Esther Pérez Jerez	

Orden del día / Acta

No.	**Asunto**	**Acuerdo**
1	**Disculpas por ausencia**	
2	**Aprobar última acta**	
3	**Asuntos pendientes última acta**	
4	Se realiza una demostración de lo desarrollado en la iteración 4.	
5	Se eligen las historias de usuario para la siguiente iteración.	001
6		
7		
8		
9	**Otros asuntos**	
10	**Próxima reunión** A determinar	

Resumen de acuerdos

Número	**Acuerdo**	**Plazo**	**Responsable**
001	Implementación de la historia de usuario US5	No aplica	Esther Pérez
002			
003			
004			
005			
006			

REUNIÓN: 06

Fecha:		
Hora comienzo:		**Hora finalización:**
Lugar:	Universidad San Jorge	
Elabora acta:	Esther Pérez Jerez	
Convocados:	Antonio Iglesias Soria, Esther Pérez Jerez	

Orden del día / Acta

No.	**Asunto**	**Acuerdo**
1	**Disculpas por ausencia**	
2	**Aprobar última acta**	
3	**Asuntos pendientes última acta**	
4	Se realiza una demostración de lo desarrollado en la iteración 5.	
5	Se eligen las historias de usuario para la siguiente iteración.	001
6		
7		
8		
9	**Otros asuntos**	
10	**Próxima reunión** A determinar	

Resumen de acuerdos

Número	**Acuerdo**	**Plazo**	**Responsable**
001	Implementación de la historia de usuario US7 y realización de pruebas con una persona experta en la generación de horarios manualmente	No aplica	Esther Pérez
002			
003			
004			
005			
006			

REUNIÓN: 07

Fecha:		
Hora comienzo:		**Hora finalización:**
Lugar:	Universidad San Jorge	
Elabora acta:	Esther Pérez Jerez	
Convocados:	Antonio Iglesias Soria, Esther Pérez Jerez	

Orden del día / Acta

No.	**Asunto**	**Acuerdo**
1	**Disculpas por ausencia**	
2	**Aprobar última acta**	
3	**Asuntos pendientes última acta**	
4	Se realiza una demostración de lo desarrollado en la iteración 6.	
5	Se eligen las historias de usuario para la siguiente iteración.	001
6		
7		
8		
9	**Otros asuntos**	
10	**Próxima reunión** A determinar	

Resumen de acuerdos

Número	**Acuerdo**	**Plazo**	**Responsable**
001	Implementación de la historia de usuario US6.1 y tareas de refactorización y optimización del código.	No aplica	Esther Pérez
002			
003			
004			
005			
006			

REUNIÓN: 08

Fecha:		
Hora comienzo:		**Hora finalización:**
Lugar:	Universidad San Jorge	
Elabora acta:	Esther Pérez Jerez	
Convocados:	Antonio Iglesias Soria, Esther Pérez Jerez	

Orden del día / Acta

No.	**Asunto**	**Acuerdo**
1	**Disculpas por ausencia**	
2	**Aprobar última acta**	
3	**Asuntos pendientes última acta**	
4	Se realiza una demostración de lo desarrollado en la iteración 7.	
5		001
6		
7		
8		
9	**Otros asuntos**	
10	**Próxima reunión** A determinar	

Resumen de acuerdos

Número	**Acuerdo**	**Plazo**	**Responsable**
001	Implementación de la historia de usuario US08.	No aplica	Esther Pérez
002			
003			
004			
005			
006			

REUNIÓN: 10

Fecha:		
Hora comienzo:		**Hora finalización:**
Lugar:	Universidad San Jorge	
Elabora acta:	Esther Pérez Jerez	
Convocados:	Antonio Iglesias Soria, Esther Pérez Jerez	

Orden del día / Acta

No.	**Asunto**	**Acuerdo**
1	**Disculpas por ausencia**	
2	**Aprobar última acta**	
3	**Asuntos pendientes última acta**	
4	Se realiza una demostración de lo desarrollado en la iteración 9.	
5	Se evalúa el estado final del proyecto.	
6		
7		
8		
9	**Otros asuntos**	
10	**Próxima reunión** A determinar	

Resumen de acuerdos

Número	**Acuerdo**	**Plazo**	**Responsable**
001		No aplica	Esther Pérez
002			
003			
004			
005			
006			

13.3. Anexo 3: Historias de usuario

ID: 0	Título: Preparación	
Descripción: Es necesario crear las estructuras de datos, las bases de datos y todo lo necesario previamente a la implementación del proyecto, además de comprobar que funciona y que el diseño realizado cumple con los requerimientos del proyecto y no necesita modificaciones.		
Prioridad: Alta	Riesgo: Bajo	Esfuerzo: 30 horas

ID: 01	Título: Interfaz para la carga de datos	
Descripción: Es necesario la generación de un interfaz sencillo y agradable que permita a los usuarios de la aplicación la inserción de los datos necesarios para la generación de sus calendarios y/o horarios. Esto incluye la creación y diseño de la base de datos que dará soporte a dicha información.		
Prioridad: Alta	Riesgo: Medio	Esfuerzo: 30 horas

ID: 02	Título: Interfaz para la muestra de las soluciones	
Descripción: Es necesario la generación de un interfaz sencillo y agradable que permita a los usuarios visualizar y entender los resultados de la generación de sus calendarios y/o horarios.		
Prioridad: Alta	Riesgo: Medio	Esfuerzo: 20 horas

ID: 03	Título: Algoritmo de búsqueda aleatorio	
Descripción: El planificador debe ser capaz de encontrar soluciones válidas siguiendo el modelo de un algoritmo de búsqueda aleatoria por lo que será necesario implementar una adaptación que se ajuste a las necesidades del proyecto.		
Prioridad: Alta	Riesgo: Medio	Esfuerzo: 20 horas

ID: 04	Título: Algoritmo de búsqueda genético	
Descripción: El planificador debe ser capaz de encontrar soluciones válidas siguiendo el modelo de un algoritmo de búsqueda genético por lo que será necesario implementar una adaptación que se ajuste a las necesidades del proyecto.		
Prioridad: Alta	Riesgo: Alto	Esfuerzo: 30 horas

ID: 05	Título: Algoritmo de búsqueda dirigido	
Descripción: El planificador debe ser capaz de encontrar soluciones válidas siguiendo el modelo de un algoritmo de búsqueda dirigida, partiendo del algoritmo de búsqueda genético, por lo que será necesario implementar una adaptación que se ajuste a las necesidades del proyecto.		
Prioridad: Alta	Riesgo: Alto	Esfuerzo: 30 horas

ID: 06	Título: Clase evaluadora / soluciones	
Descripción: Para poder establecer qué soluciones son mejores será necesario la implementación de un modelo de evaluación que permita mediante un sistema de puntuación discriminar entre las distintas soluciones generadas.		
Prioridad: Alta	Riesgo: Medio	Esfuerzo: 15 horas

ID: 6.1	Título: Clase estadística	
Descripción: Para poder llevar a cabo un estudio comparativo de los distintos algoritmos de búsqueda a implementar, es necesario crear una estructura de datos que permita recolectar y almacenar la información relevante (soluciones generadas, puntuaciones, tiempo, etc) para posteriormente poder comparar los resultados obtenidos por los diversos algoritmos.		
Prioridad: Media	Riesgo: Bajo	Esfuerzo: 10 horas

ID: 7	Título: Mejoras usabilidad

Descripción:

Después de llevar a cabo las pruebas con el experto se recibieron diversas sugerencias de mejora que se decidieron tener en cuenta:

- C1 - Introducir todos los alumnos matriculados manualmente sería muy costoso, por ello el usuario propone modificar el concepto que se tiene de este elemento y transformarlo en "Alumno tipo" de esta forma solo es necesario crear los distintos itinerarios que pudieran darse dentro de un mismo curso.
- C3 - Existe un problema con el vocabulario utilizado, slots y horarios, son utilizados por este usuario como pastilla y plantilla respectivamente.
- C4 - La ejecución y la inserción de los datos deberían estar separados por cuatrimestres o periodos escolares.
- C5 - Crear las interfaces de generación de calendarios y de introducción masiva de datos de forma separada.
- C6 - La inserción de la información relativa a los grados y los cursos es un poco confusa.
- C8 - El cuadrante generado para la inserción de la disponibilidad de los docentes fue muy bien recibido por parte del usuario siendo la sección con más éxito dentro de todas las pruebas realizadas. Como aportación el usuario sugiere que no sólo se pueda indicar la disponibilidad, sino que además se pueda indicar la no disponibilidad de forma que el usuario pueda elegir la manera más cómoda y rápida de rellenar dicho cuadrante.

Prioridad: Baja	Riesgo: Bajo	Esfuerzo: 10 horas

ID: 8	Título: Manual de usuario	
Descripción: Dada la complejidad del problema y de los datos a utilizar, se decidió redactar una guía que explique los distintos elementos que emplea el algoritmo y como se utilizan, así como una explicación paso a paso de como generar un horario por primera vez.		
Prioridad: Media	Riesgo: Bajo	Esfuerzo: 10 horas

ID: 9	Título: Gestor de espacios	
Descripción: Incluir las aulas como un nuevo elemento en la generación de algoritmos que incluya nuevas restricciones: ○ R7 (NO_MORE_SPACES): esta restricción será la encargada de comprobar que no se ha excedido el máximo de aulas disponibles en una misma franja horaria. ○ R8 (OUT_OF_CAPACITY): esta restricción será la encargada de comprobar que el número de alumnos matriculado en la asignatura a impartir no supera la capacidad del aula asignada.		
Prioridad: -	Riesgo: Medio	Esfuerzo: 20 horas

ID: 10	Título: Vista múltiple	
Descripción: Añadir al interfaz para la muestra de las soluciones la posibilidad de mostrar más de un calendario a la vez, permitiendo así que el usuario pueda elegir entre los distintos cursos todos los que quiera mostrar a la vez.		
Prioridad: -	Riesgo: Bajo	Esfuerzo: 10 horas

ID: 11	Título: Editor de soluciones	
Descripción: Permitir al usuario mediante la técnica de "drag and drop" mover las clases que desee cambiar de franja horaria. Esto implica también añadir a esta vista una visualización de las colisiones que se generen a partir de estos cambios.		
Prioridad: -	Riesgo: Alto	Esfuerzo: 40 horas

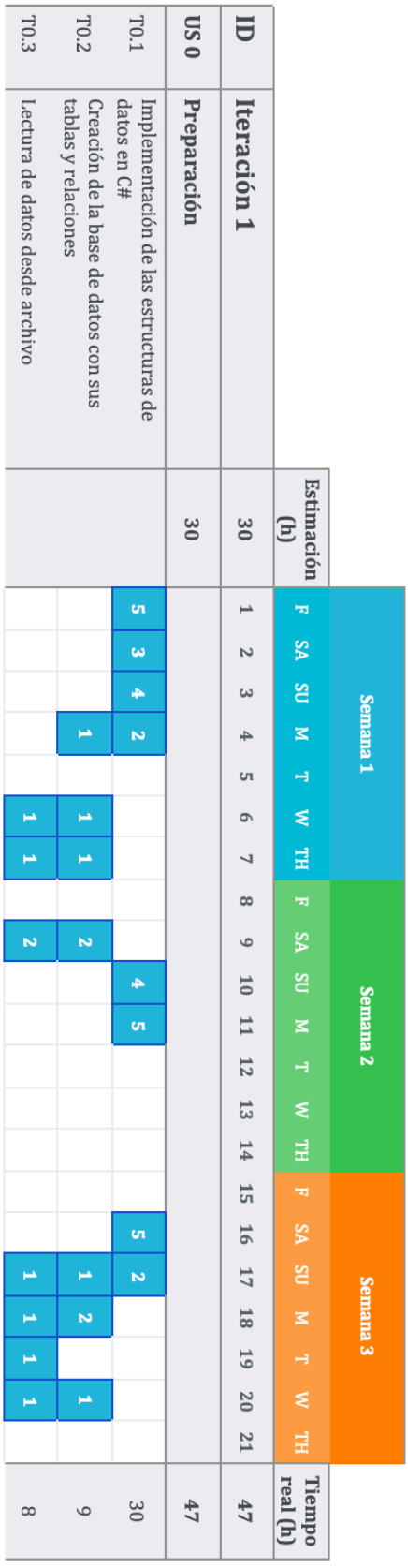
ID: 12	Título: Control en las planificaciones	
Descripción: Implementar la posibilidad de generar solo una parte del horario o generar solo el horario de unos grupos en concreto. Esto implicaría la elaboración de un formulario que permita al usuario elegir las asignaturas, cursos y/o alumnos entre otros para generar parcialmente un horario.		
Prioridad: -	Riesgo:	Esfuerzo: 20 horas

ID: 13	Título: Integración con los sistemas	
Descripción: Permitir al usuario importar y exportar los datos y las soluciones en el formato que utilice su entidad para facilitar el proceso y maximizar la compatibilidad.		
Prioridad: -	Riesgo: Medio	Esfuerzo: 30 horas

ID: 14	Título: Algoritmo de búsqueda propio	
Descripción: Creación de un algoritmo propio enfocado en este problema con el fin de optimizar el proceso y comparar con los elaborados previamente si resulta en alguna mejora la creación de un algoritmo concreto.		
Prioridad: -	Riesgo: Alto	Esfuerzo: 40 horas

ID: 15	Título: Múltiples periodos	
Descripción: Implementación de la posibilidad de generar horarios o calendarios de exámenes divididos en varios periodos lectivos o cuatrimestres.		
Prioridad: -	Riesgo: Medio	Esfuerzo: 20 horas

13.4. Anexo 4: Horas trabajadas



ID	Iteración 2	Estimación (h)	Semana 1							Semana 2							Tiempo real (h)
			F	SA	SU	M	T	W	TH	F	SA	SU	M	T	W	TH	
		35	1	2	3	4	5	6	7	8	9	10	11	12	13	14	43
US 3	**Algoritmo de búsqueda aleatorio**	20															28
T3.1	Generar población inicial aleatoria		5			2											7
T3.2	Implementar un proceso de mutación aleatorio parametrizado			4				1		2			3				10
T3.3	Desarrollar una base para la iteración de los algoritmos				6			4				1					11
US 6	**Clase evaluadora**	15															15
T6.1	Implementar la estructura de datos para las soluciones										2	2					4
T6.2	Implementar funciones que evalúen las diferentes restricciones												3	4			7
T6.3	Crear función global que evalúe el conjunto de la solución									4							4

ID	Iteración 3	Estimación (h)	Semana 1							Semana 2							Tiempo real (h)
			F	SA	SU	M	T	W	TH	F	SA	SU	M	T	W	TH	
		30	1	2	3	4	5	6	7	8	9	10	11	12	13	14	25
US 4	**Algoritmo de búsqueda genético**	30															25
T4.1	Implementación de varias mutaciones y/o combinaciones		4	4	2		1			4	3	3		2		2	25

ID		Estimación (h)	Semana 1							Semana 2							Tiempo real (h)
			F	SA	SU	M	T	W	TH	F	SA	SU	M	T	W	TH	
			1	2	3	4	5	6	7	8	9	10	11	12	13	14	
Iteración 4		30															34
US 1	**Interfaz para la carga de datos**	30															34
T1.1	Página inicial		3	4													7
T1.2	Vista general de cada elemento			2	4					3			2				11
T1.3	Vista de edición y creación de cada elemento						3			4	4			2		3	16

ID		Estimación (h)	Semana 1							Tiempo real (h)
			F	SA	SU	M	T	W	TH	
			1	2	3	4	5	6	7	
Iteración 5		20								22
US 2	**Interfaz para la muestra de soluciones**	20								22
T2.1	Búsqueda y adaptación de un calendario		5	4	1					10
T2.2	Implementación de las soluciones en el calendario		4				4		4	12

ID	Iteración 6	Estimación (h)	Semana 1							Semana 2							Tiempo real (h)
			F	SA	SU	M	T	W	TH	F	SA	SU	M	T	W	TH	
		30	1	2	3	4	5	6	7	8	9	10	11	12	13	14	28
US 5	**Algoritmo de búsqueda dirigido**	30															28
T5.1	Implementar mutación dirigida en combinación con otros padres		4	5	3		2			1	3	4	4		2		28

ID	Iteración 7	Estimación (h)	Semana 1							Semana 2							Tiempo real (h)
			F	SA	SU	M	T	W	TH	F	SA	SU	M	T	W	TH	
		15	1	2	3	4	5	6	7	8	9	10	11	12	13	14	26
US 7	**Cambios en la usabilidad**	10															18
T7.1	Separación de la inserción de datos y la generación				5												5
T7.2	Implementación de varios periodos en una misma ejecución										3	2					5
T7.3	Cambiar el método de creación de cursos					2											2
T7.4	Añadir la opción de disponibilidad o no disponibilidad para los docentes						2										5
T7.5	Cambios en la terminología de la aplicación							3									1
-	**Pruebas con experto**	5	4	4													8

ID	Iteración 8	Estimación (h)	Semana 1							Semana 2							Semana 3							Tiempo real (h)
			F	SA	SU	M	T	W	TH	F	SA	SU	M	T	W	TH	F	SA	SU	M	T	W	TH	
		20	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	37
US 6.1	Clase estadística	10																						24
T6.1.1	Estructura de datos para dar soporte a las estadísticas				2	3																		5
T6.1.2	Guardado en archivo de las estadísticas					1	3				2														6
T6.1.3	Elaboración de un estudio comparativo entre los algoritmos									4	2	4		3										13
	Refactorización y optimización del código	10															5	4	4					13

ID	Iteración 9	Estimación (h)	Semana 1							Tiempo real (h)
			F	SA	SU	M	T	W	TH	
		10	1	2	3	4	5	6	7	11
US 8	Manual de usuario	10								11
T8.1	Redacción e implementación de una guía para usuarios		3	4	2		2			11