

**Desarrollo de un modelo de satisfacción de restricciones para la asignación de citas
terapéuticas en la Clínica de Rehabilitación del Valle**

Autores

**Harvin Jessid Rengifo Romero
Holmes Giovanni Salazar Osorio**



**Universidad del Valle
Escuela de Ingeniería de Sistemas y Computación
Ingeniería de Sistemas
Tuluá - Valle
2013**

**Desarrollo de un modelo de satisfacción de restricciones para la asignación de citas
terapéuticas en la Clínica de Rehabilitación del Valle**

Autores

Harvin Jessid Rengifo Romero*

Holmes Giovanni Salazar Osorio**

harssed@gmail.com*, holmes.salazar@correounivalle.edu.co**

Código. 200766976*, Código. 200766960**

Documento presentado como requisito parcial para la obtención de grado de Ingeniero de Sistemas

Director

Jesús Alexander Aranda Bueno, PhD



Universidad del Valle

Escuela de Ingeniería de Sistemas y Computación

Ingeniería de Sistemas

Tuluá - Valle

2013

Trabajo de grado presentado por:
Harvin Jessid Rengifo Romero
Holmes Giovanni Salazar Osorio
Como requisito parcial para la obtención del título de Ingeniero de Sistemas.

Jesús Alexander Aranda Bueno, Ph. D
Director

Jurado 1

Jurado 2

En el desarrollo de este trabajo de grado deseamos darle las gracias, en primer lugar a Dios por darnos la vida y la salud, por guiarnos en nuestro camino y en nuestro crecimiento como personas. De igual forma, extendemos un agradecimiento especial a todas las personas que fueron parte de este proceso:

A nuestros padres por su educación, confianza y apoyo incondicional a lo largo de nuestra carrera universitaria.

A nuestro director y mentor Jesús Aranda por su constante apoyo, comprensión y disposición, además de sus valiosos aportes a nuestro trabajo de grado.

Al profesor Juan Francisco Díaz y el grupo de investigación AVISPA, por ser uno de los primeros en darnos sus apreciaciones acerca de nuestro trabajo de grado, y por permitir que nuestra aplicación haga parte de uno de los proyectos del portal del grupo.

A los profesores David Przybilla y Oscar Ceballos quienes nos orientaron y prestaron su ayuda de manera desinteresada en el planteamiento de nuestro trabajo de grado.

Al ingeniero y compañero de clases Nestor Díaz, por proponer el problema y brindarnos sus conocimientos en el tema.

A la Clínica de Rehabilitación del Valle por la ayuda prestada en la definición del problema y otorgarnos los requerimientos necesarios para evaluar el problema.

(Holmes) A mi novia, por su paciencia, comprensión, amor y por brindarme ese apoyo incondicional durante los últimos años de mi carrera.

(Harvin) A mi novia, por su amor, compañía, preocupación, entendimiento y por ser mi fuente de inspiración de perseverancia y optimismo, durante el tiempo que llevamos juntos.

Por último, y no menos importante, a todas las personas y profesores que de alguna u otra manera contribuyeron en el desarrollo de nuestra carrera y en la culminación del presente trabajo.

Índice general

1. Contexto y Objetivos	1
1.1. Planteamiento del Problema	1
1.1.1. Descripción del Problema	1
1.1.2. Formulación del problema	2
1.2. Justificación	2
1.3. Objetivos	3
1.3.1. Objetivo General	3
1.3.2. Objetivos Específicos	3
1.4. Alcance	3
1.4.1. Definición y delimitación del problema	4
1.4.2. Especificación del problema	4
1.4.3. Solución del problema	6
2. Marco de Referencia	7
2.1. Estado del Arte	7
2.1.1. CREAM: Consejero para la Repartición de Artículos y Evaluadores en Eventos Académicos	7
2.1.2. Reingeniería de PATHOS	8
2.1.3. ChariTime	8
2.2. Marco Teórico	9
2.2.1. Programación por Restricciones	9
2.2.2. Problema de Satisfacción de Restricciones (CSP)	10
2.2.3. Gecode	10
2.2.4. Drupal	11
2.3. Marco Conceptual	12
2.3.1. Backtracking	12

2.3.2.	Heurística	12
2.3.3.	Propagación de Restricciones	13
2.3.4.	XML	13
2.3.5.	CGI	14
2.3.6.	GSL	14
3.	Modelo Matemático	15
3.1.	Variables y parámetros	15
3.1.1.	Parámetros de entrada	15
3.1.2.	Parámetros derivados	16
3.1.3.	Variables	16
3.1.4.	Dominios	17
3.2.	Restricciones	17
3.2.1.	Restricción Tiempo de Finalización	17
3.2.2.	Restricción Pre-asignación de Tiempo según Pacientes	17
3.2.3.	Restricción Pre-asignación de Tiempo según Profesionales	18
3.2.4.	Restricción No Solapamiento entre Citas	19
3.2.5.	Restricción Día Diferente	19
3.2.6.	Restricción de Capacidad	20
3.2.7.	Función Objetivo: Restricción de Preferencia	20
3.3.	Heurísticas	20
3.3.1.	Heurística para la Distribución Equilibrada de Pacientes	20
3.3.2.	Heurística para la Distribución Equilibrada de Trabajo	21
4.	Implementacion del Modelo	22
4.1.	Estructuras de Datos	22
4.2.	Declaración de Variables	24
4.3.	Implementación de las Restricciones	27
4.3.1.	Restricción Tiempo de Finalización	27
4.3.2.	Restricción Pre-asignación de Tiempo según Pacientes	28
4.3.3.	Restricción Pre-asignación de Tiempo según Profesionales	28
4.3.4.	Restricción No Solapamiento entre Citas	29
4.3.5.	Restricción Día Diferente	30
4.3.6.	Restricción de Capacidad	31
4.3.7.	Función Objetivo	32

4.4.	Implementación de las Heurísticas	33
4.4.1.	Heurística de Distribución Equilibrada de Pacientes	33
4.4.2.	Heurística para la Distribución Equilibrada de Trabajo	35
5.	Descripción General del Sistema	37
5.1.	Arquitectura	37
5.2.	Detalles de la Implementación	39
5.2.1.	Drupal	40
5.2.2.	PHP	40
5.2.3.	XML	40
5.2.4.	JQuery	40
5.2.5.	Datatables	40
5.2.6.	API Google Chart	41
5.2.7.	SyntaxHighlighter	42
5.2.8.	Fullcalendar	42
6.	Experimentación y Pruebas	43
6.1.	Descripción de las Pruebas	44
6.1.1.	Entradas de Prueba	44
6.1.1.1.	Pruebas de Menor Escala	44
6.1.1.2.	Pruebas de Mayor Escala	44
6.1.2.	Consideraciones para el inicio de las Pruebas	45
6.1.3.	Procedimiento de las Pruebas	47
6.2.	Resultados	48
6.2.1.	Pruebas de Menor Escala	48
6.2.1.1.	Resultados sin la Restricción de Preferencia	48
6.2.1.2.	Resultados con la Restricción de Preferencia	49
6.2.2.	Pruebas de Mayor Escala	51
6.2.2.1.	Resultados sobre la Heurística 1	51
6.2.2.2.	Resultados sobre la Heurística 2	53
6.2.2.3.	Resultados sobre la Heurística 3	56
6.3.	Análisis de Resultados	60
6.3.1.	Pruebas de Menor Escala	60
6.3.2.	Pruebas de Mayor Escala	61

7. Conclusiones y Trabajos Futuros	63
7.1. Conclusiones	63
7.2. Trabajos Futuros	65
Apéndices	70
A. Descripción del Formato del Archivo de Entrada	71
A.1. Archivo de ejemplo	71
A.2. Explicación del archivo de entrada	72
A.3. Explicación de la Disponibilidad	74
B. Descripción del Formato del Archivo de Salida	76
C. Especificaciones de las Especialidades	78
D. Especificaciones del Generador de Archivos de Prueba	79
E. Distribución Pacientes	80

Índice de figuras

2.1. Recorrido paso a paso de un árbol binario por medio de un esquema básico de Backtracking [14].	12
5.1. Arquitectura de la Aplicación.	38

Índice de tablas

6.1. Entradas de Prueba: Menor Escala	44
6.2. Entradas de Prueba: Mayor Escala	45
6.3. Pacientes:10-100. Heurísticas:1-3. Sin Preferencia	49
6.4. Pacientes:10-100. Heurísticas:1-3. Con Preferencia	50
6.5. Heurística:1. Sin Preferencia. Paciente:100-300.Distribuciones:D0-D4	52
6.6. Heurística:1. Con Preferencia. Paciente:100-300.Distribuciones:D0-D4	53
6.7. Heurística:2. Sin Preferencia. Paciente:100-300.Distribuciones:D0-D4	55
6.8. Heurística:2. Con Preferencia. Paciente:100-300.Distribuciones:D0-D4	56
6.9. Heurística:3. Sin Preferencia. Paciente:100-300.Distribuciones:D0-D4	57
6.10. Heurística:3. Con Preferencia. Paciente:100-300.Distribuciones:D0-D4	59
A.1. Tabla de Disponibilidad	75
C.1. Especificaciones de Especialidades	78
E.1. No. Pacientes atendidos	81

Resumen

La programación de restricciones es una tecnología de software utilizada para la descripción y posterior resolución efectiva de problemas grandes y complejos, particularmente aquellos de naturaleza combinatoria, los cuales comprenden muchas áreas de la vida real, especialmente las áreas de planificación y programación de tareas (scheduling). El problema de asignación de citas terapéuticas, hace parte del área de programación de tareas, ya que consiste en asignar las citas que componen el tratamiento de cada uno de los pacientes que necesitan ser atendidos, teniendo en cuenta la disponibilidad tanto de los pacientes como de los profesionales que los atenderán.

En este documento se proponen los mecanismos que se utilizaron para dar solución al problema de asignación de citas terapéuticas. Inicialmente se definió un modelo de satisfacción de restricciones (CSP) que incluye las variables presentes en el proceso de asignación de citas terapéuticas. Así mismo, se presenta la implementación de dicho modelo en el lenguaje de programación C++ a través de la librería para programación por restricciones Gecode. Finalmente, se desarrolló una aplicación web que permite visualizar los resultados obtenidos de la solución del problema.

Palabras Clave: Aplicación Web, Asignación de Citas Terapéuticas, C++, CSP, Gecode, Programación por restricciones, Scheduling.

Introducción

Muchas decisiones que se toman a la hora de resolver diversos problemas habituales están sujetas a restricciones. Decisiones tan cotidianas como fijar una cita, planificar un viaje, comprar un automóvil o preparar un plato de cocina puede depender de muchos aspectos interdependientes e incluso conflictivos, cada uno de los cuales está sujeto a un conjunto de restricciones que se deben satisfacer para que la decisión sea correcta.

El problema de la asignación de citas terapéuticas constituye un procedimiento de una naturaleza distinta a lo que se conoce comúnmente, ya que no se trata de la asignación de una única cita a un paciente determinado, sino de varias de ellas, es decir se involucra el concepto de tratamiento médico. Esto implica una mayor complejidad del modelo, puesto que se deben tener en cuenta consideraciones tan importantes, como por ejemplo, la disponibilidad de los profesionales, ya que debido a la naturaleza del problema ésta se ve reducida en la continuidad del tiempo.

El documento se encuentra organizado de la siguiente forma: en el capítulo 1, se contextualiza el problema y se describen los objetivos del presente proyecto de grado. En el capítulo 2, se presenta la conceptualización necesaria para facilitar el entendimiento del problema.

En el capítulo 3, se describe el modelo matemático que se ha definido para dar solución al problema de asignación de citas terapéuticas. En el capítulo 4, se definen distintos elementos, tales como: las estructuras de datos utilizadas, las variables, restricciones y heurísticas que se tuvieron en cuenta al momento de la implementación del modelo.

En el capítulo 5, se describen las especificaciones del sistema; en éste aparecen las estrategias que se utilizaron para implementar la solución como un servicio web. De igual forma, se describe el proceso que se realizó para la generación de las entradas de la aplicación. Finalmente, para evaluar la eficiencia del modelo, en el capítulo 6, se presentan las pruebas y el análisis correspondiente de las mismas.

Capítulo 1

Contexto y Objetivos

1.1. Planteamiento del Problema

1.1.1. Descripción del Problema

En un modelo de adjudicación de citas de una clínica de rehabilitación, por la condición de los pacientes que se atienden, se asignan varias sesiones o citas en un intervalo de tiempo definido para que los mismos sean tratados por uno o varios profesionales. Sin embargo, actualmente la Clínica de Rehabilitación del Valle cuenta con un sistema de asignación de citas (Sistema en Salud), el cual no se ajusta a este modelo, puesto que fue diseñado siguiendo un modelo de adjudicación de citas para una clínica u hospital común, donde el paciente solicita una única cita con un profesional determinado.

Debido a la situación descrita en el párrafo anterior, las secretarías de la clínica no pueden asignar automáticamente todas las citas o sesiones que componen el tratamiento de un paciente determinado. Para solucionar este problema, las secretarías han recurrido a procedimientos artesanales como suministrar un formato a los pacientes, con el objetivo de que éstos escriban los espacios de tiempo en los cuales están disponibles. Dicho formato es utilizado para asignar las sesiones de los pacientes, teniendo en cuenta los horarios de atención de los profesionales.

Otra falla que posee el Sistema en Salud, es que no tiene un procedimiento que asegure la distribución equilibrada de pacientes entre los diferentes profesionales que atienden en una misma especialidad, lo que obliga a las secretarías a realizar este proceso manualmente, con el fin de evitar la sobrecarga de trabajo entre los profesionales de la clínica.

1.1.2. Formulación del problema

¿Qué estrategia enfocada en el paradigma de programación por restricciones es necesaria implementar para dar solución a los inconvenientes presentados en el proceso de asignación de citas terapéuticas en la Clínica de Rehabilitación del Valle?

1.2. Justificación

En la Clínica de Rehabilitación del Valle las citas que hacen parte de un tratamiento médico de un paciente determinado se asignan de forma independiente, es decir la totalidad de las citas que corresponden al tratamiento no se asignan directamente desde el comienzo del mismo. El personal encargado de realizar el proceso de asignación de citas son las secretarias, las cuales requieren de un esfuerzo considerable y de un conocimiento previo y detallado de la disponibilidad, tanto de los horarios como las especialidades de los profesionales de la salud que laboran en la clínica.

La asignación no se realiza de la manera más eficiente y confiable, por tal motivo se pretende realizar la asignación de citas de una forma segura y correcta, donde no se dependa de una persona (solo se dependerá de ella para el ingreso de la información, el procedimiento como tal lo realizará la aplicación), ya que por distintas razones una empresa y más aún, una clínica, está sujeta a reestructuraciones. Es importante resaltar que en la clínica existe un flujo constante de pacientes, lo que se convierte en un aspecto representativo ya que esto significa que, a medida que pase el tiempo las condiciones iniciales van a variar constantemente de modo que se acrecentaría la dificultad de asignar las citas, debido a los cambios de disponibilidad de los profesionales.

Por lo descrito anteriormente, se logra evidenciar que la asignación de citas terapéuticas constituye un proceso complejo que realiza el personal encargado. Debido a la naturaleza del problema se hace indispensable utilizar alguna técnica apropiada para realizar dicho proceso de una manera mucho más eficiente. La programación por restricciones es uno de los enfoques que se pueden emplear para la consecución de este fin, puesto que ha sido utilizado resolver problemas combinatorios, lo que es en esencia el proceso de asignación de citas [7].

1.3. Objetivos

1.3.1. Objetivo General

Diseñar e implementar un modelo basado en el paradigma de programación por restricciones que permita resolver el problema de la asignación de citas terapéuticas en la Clínica de Rehabilitación del Valle.

1.3.2. Objetivos Específicos

1. Diseñar un modelo de satisfacción de restricciones que incluya las variables presentes en el proceso de asignación de citas terapéuticas de la Clínica de Rehabilitación del Valle.
2. Implementar el modelo de satisfacción de restricciones que describe el proceso de asignación de citas terapéuticas de la Clínica de Rehabilitación del Valle.
3. Implementar heurísticas que optimicen el tiempo que se requiere para encontrar una solución consistente¹ al problema de asignación de citas terapéuticas.
4. Desarrollar una aplicación web que permita el ingreso de los datos para ejecutar las restricciones del modelo, además de permitir visualizar las soluciones encontradas a través de la implementación de dicho modelo.

1.4. Alcance

El presente proyecto se limita exclusivamente al planteamiento e implementación del modelo de satisfacción por restricciones y al desarrollo de una aplicación web, que permite ingresar los datos para ejecutar las restricciones del modelo, y desplegar de la solución arrojada por el modelo.

Es importante tener claro que el proyecto no abarca temas relacionados al manejo de la información personal de los pacientes, secretarias y especialistas, manejo de historias clínicas, manejo de diagnósticos médicos y/o formulación de recetas médicas, ni gestión de facturación y cuentas.

En las secciones 1.4.1, 1.4.2 y 1.4.3, se especifican las principales características del problema de asignación de citas terapéuticas que se tuvieron en cuenta en el modelo de satisfacción por restricciones.

¹Es una asignación dónde se satisfacen todas las restricciones que existen sobre un conjunto de variables

1.4.1. Definición y delimitación del problema

En el problema de asignación de citas terapéuticas se tienen los siguientes elementos: pacientes, profesionales, tratamientos y citas, donde:

- Cada paciente requiere de un tratamiento con un profesional perteneciente a la especialidad que el paciente necesita.
- Cada tratamiento está compuesto por un número determinado de citas en una misma especialidad.
- Cada una de las citas de un tratamiento tiene un tiempo de inicio, un tiempo de finalización y una duración determinados, además todas las citas de un tratamiento tienen la misma duración, debido a que pertenecen a la misma especialidad.
- Un paciente puede elegir el profesional con el que desea ser tratado.
- Cada profesional dependiendo de su especialidad tiene un límite de pacientes simultáneos que puede atender.

El principal objetivo es encontrar los tiempos de inicio y de finalización adecuados de todas las citas, de manera que todos los pacientes puedan ser atendidos, además de maximizar el número de pacientes que desean ser atendidos por el profesional de su preferencia. Todo esto logrando una distribución equilibrada de pacientes entre los diferentes profesionales que trabajan en una misma especialidad de la clínica.

1.4.2. Especificación del problema

1. Datos de entrada:

- Número de Especialidades.
- Número de Pacientes.
- Número de Profesionales.
- Horarios de atención de los profesionales.
- Conjunto de especialidades en las que atiende un profesional.
- Máximo número de pacientes simultáneos que puede atender un profesional.
- Disponibilidad de los pacientes.

- Especialidades en las que necesitan ser atendidos los pacientes.
- Número de citas que componen el tratamiento de un paciente.
- Conjunto de profesionales de preferencia de un paciente en una determinada especialidad.
- Duración en slots de tiempo de las citas de cada especialidad. Un slot es una cantidad unitaria de tiempo.
- Número de intervalos en los que se divide un día (Generalmente cada intervalo representa una hora).
- Número de slots de tiempo en los que se divide un intervalo.
- Número de slots de tiempo en los que se divide un día.
- Número total de semanas disponibles para asignar la totalidad de las citas.
- Número total de intervalos en los que se divide una semana.
- Número total de slots de tiempo disponibles para asignar la totalidad de las citas.

2. Variables de salida:

- Número de pacientes atendidos por especialidad.
- Número de pacientes atendidos por profesional.
- Tiempos de inicio y finalización de cada una de las citas que componen el tratamiento de un paciente.
- Profesional que atenderá todas las citas del tratamiento de un paciente.

3. Restricciones:

- El tiempo de finalización de una cita debe ser igual al tiempo de inicio más la duración estipulada para todas las citas que pertenecen a una misma especialidad.
- Cada asignación de una cita debe estar acorde con los horarios establecidos por el paciente para recibir su tratamiento, es decir, al asignar una cita se debe tener en cuenta la disponibilidad de un paciente.
- Cada asignación de una cita debe estar acorde con los horarios de atención del especialista al que le corresponde atenderla, es decir, al asignar una cita se debe tener en cuenta la disponibilidad del profesional.

- En caso que un paciente necesite tratamiento en varias especialidades, los horarios de las citas a asignar a este paciente para una especialidad determinada no se deben cruzar con los horarios de las demás citas asignadas en otras especialidades.
- No se pueden asignar dos citas en un mismo día para un mismo paciente en la misma especialidad.
- Los pacientes asignados a un profesional en ningún momento deben superar su límite de pacientes.
- Un paciente puede solicitar que sea atendido por un profesional de su preferencia.

1.4.3. Solución del problema

Al tratar de solucionar el problema de asignación de citas terapéuticas, se busca encontrar un horario de atención a todos los tratamientos de los pacientes de manera que se logren alcanzar en un grado aceptable los siguientes objetivos:

- Asignar automáticamente todas las citas que componen el tratamiento de un paciente, teniendo tanto la disponibilidad del paciente como la disponibilidad del profesional por el que va a ser atendido.
- Atender la totalidad de los pacientes que necesitan de un tratamiento dentro del tiempo límite estipulado por el usuario.
- Tener un mayor control sobre el cumplimiento de los tratamientos asignados a los pacientes.
- Maximizar el número de pacientes que son atendidos por el profesional de su preferencia, con el fin de generar mayor confianza a los pacientes en el tratamiento que recibirán.
- Distribuir equitativamente los pacientes entre los profesionales de una misma especialidad, evitando así la sobrecarga o escasez de pacientes en algunos profesionales.

Capítulo 2

Marco de Referencia

2.1. Estado del Arte

A continuación presentaremos tres proyectos, en los cuales se han solucionado problemas similares al proceso de asignación de citas terapéuticas utilizando el paradigma de programación por restricciones. Sin embargo, pese a que éstos parten de una misma base para resolver sus problemas, utilizan distintos enfoques, como lo son CSPs que involucran tanto soluciones parciales como totales, CSPs distribuidos, uso de heurísticas, entre otros.

2.1.1. CREAM: Consejero para la Repartición de Artículos y Evaluadores en Eventos Académicos

CREAM, es una aplicación computacional desarrollada para realizar eficientemente el proceso de distribución de los artículos candidatos para participar en eventos académicos. Para su desarrollo se diseñó un modelo computacional del problema, utilizando el Paradigma de Programación por Restricciones y se implementó utilizando MOzArt¹, un lenguaje de programación apropiado para este paradigma. La aplicación posee una arquitectura de tres capas, la primera capa incluye todas las funcionalidades que permiten la interacción entre el programa y el usuario, la segunda capa se encuentran el control del programa y las principales funcionalidades de la aplicación, y en la tercera capa se incluyen los archivos de entrada y salida y las funcionalidades que permiten su comunicación con el programa [8].

CREAM es un proyecto realizado por Jesús Alexander Aranda, Juan Francisco Díaz y James Ortiz,

¹<http://www.mozart-oz.org/>

integrantes del grupo de investigación AVISPA².

2.1.2. Reingeniería de PATHOS

Los estudiantes de la Pontificia Universidad Javeriana, Gustavo Adolfo Pabón Sanchez y Carlos Eduardo Rodriguez de Silvestre, presentaron el proyecto de grado denominado “Reingeniería de PATHOS” para optar por el título de Ingenieros de Sistemas y Computación de esta universidad. PATHOS es un proyecto del grupo de investigación AVISPA³ que da solución al problema de la planificación de horarios de clase por medio del paradigma de programación por restricciones. PATHOS mejora el estado del arte en estrategias de programación de horarios automático, ya que maneja los problemas de un tamaño mucho más grande de lo que se ha considerado hasta el momento. Las estrategias utilizadas en PATHOS son también aplicables a otros tipos de planificación y programación de problemas [12], como lo es el problema de asignación de citas.

Este proyecto de grado tenía como objetivo hacer un proceso de reingeniería a PATHOS, realizando una nueva versión de este. En ese proceso se logró mejorar el tiempo de ejecución y consumo de memoria, y disminuyeron el tiempo dedicado a la puesta en marcha de la aplicación en ámbitos reales. Para el desarrollo de este proyecto, se utilizaron las siguientes tecnologías: Mozart, PHP, MYSQL, Apache, JavaScript, HTML y XML.

2.1.3. ChariTime

Es un proyecto desarrollado para la programación de citas medicas en la Clínica Cardiológica Charité de Berlín, ChariTime fue diseñado por varios investigadores del instituto GDM FIRST, de la Universidad Humboldt de Berlín y del Instituto Técnico de Berlín.

Para el desarrollo de ChariTime se utilizó el paradigma de programación por restricciones, específicamente se modeló el problema a través de CSPs Distribuidos, conocidos como DCSPs (Los cuales son CSPs en los que las variables y restricciones se distribuyen entre agentes automatizados. Éstos agentes se comunican mediante el envío de mensajes, los cuales los agentes reciben en el orden en que les fueron enviados[13]). Además de programación por restricciones, también se hizo uso de un Modelo de Componentes Objetos Distribuidos de Microsoft(DCOM)⁴ y un Servidor de Cola de Mensajes(MSMQ, un servicio de correo electrónico totalmente transparente para los programas)⁵

²Grupo de investigación conformado por la Pontificia Universidad Javeriana seccional Cali y la Universidad del Valle, el cual tiene como principal objetivo promover la investigación en ciencias de la computación y específicamente en programación aplicativa. URL:<http://avispa.univalle.edu.co>

³Ibíd

⁴<http://www.opengroup.org/comsource/>

⁵[http://msdn.microsoft.com/en-us/library/ms711472\(v=vs.85\).aspx](http://msdn.microsoft.com/en-us/library/ms711472(v=vs.85).aspx)

como bases de comunicación entre los agentes de los DCSPs [10].

2.2. Marco Teórico

2.2.1. Programación por Restricciones

La programación por restricciones es un poderoso paradigma que busca resolver problemas mediante la declaración de variables y sus dominios y restricciones que condicionan las posibles soluciones para ese conjunto de variables. Además se deben definir estrategias de búsqueda, que se basan en métodos estándares como el backtracking, la propagación de restricciones y la utilización de heurísticas [7].

Los inicios de la programación por restricciones se remontan a los años sesenta y setenta, con investigaciones en las áreas de la Inteligencia Artificial y la Computación Gráfica, pero sólo hasta los años noventa surgió una creciente conciencia de que esas investigaciones constituyen la base de un enfoque poderoso a la programación, modelización y resolución de problemas, y que los diferentes esfuerzos para aprovechar los resultados de las investigaciones podían ser unificados bajo un marco conceptual común y práctico, la programación por restricciones [15]. El ámbito de la programación por restricciones ha ido avanzando, y no sólo se basa en técnicas de la inteligencia artificial, sino que también toma un amplio rango de técnicas de las áreas de las ciencias de la computación y la investigación operativa [7].

La programación por restricciones puede dividirse en dos ramas claramente diferenciadas: la satisfacción de restricciones y la resolución de restricciones. Ambas comparten la misma terminología pero sus orígenes y técnicas de resolución son diferentes. La satisfacción de restricciones trata con problemas de dominios finitos, mientras que la resolución de restricciones está orientada principalmente a problemas sobre dominios infinitos o dominios más complejos [9, 15].

Problemas que pueden ser mejor resueltos por medio de la programación por restricciones suelen ser aquellos que se pueden formular en términos de requerimientos, propiedades generales o leyes, y para los que los métodos de dominio específico conducen a formalizaciones excesivamente complejas [1]. Algunos dominios para los cuales se ha aplicado programación por restricciones se encuentran en la sección 1.2 de [1].

2.2.2. Problema de Satisfacción de Restricciones (CSP)

En programación por restricciones, una parte muy importante para la resolución de problemas de la vida real es el modelado del problema en términos de un CSP (Del inglés, Constraint Satisfaction Problems)[9], es decir, definir:

- El conjunto de variables.
- Para cada variable, un conjunto de posibles valores que puede tomar.
- Por último, un conjunto de restricciones que limitan los valores que las variables pueden tomar [1, 9].

Formalmente un CSP se define como una terna (X, D, C) donde:

- X es un conjunto de n variables $\{x_1, \dots, x_n\}$.
- $D = \langle D_1, \dots, D_n \rangle$ es un vector de dominios. La i -ésima componente D_i es el dominio que contiene todos los posibles valores que se le pueden asignar a la variable x_i .
- C es un conjunto finito de restricciones. Cada restricción n -aria (c_n) está definida sobre un conjunto de variables $\{x_1, \dots, x_n\}$ restringiendo los valores que éstas pueden tomar simultáneamente [9].

Una solución a un CSP es una asignación de valores a todas las variables de forma que se satisfagan todas las restricciones. Una asignación de variables es un conjunto $\{(x_1, a_1), \dots, (x_n, a_n)\}$, donde cada componente representa una tupla de pares ordenados, (x, a) donde a la variable x , se le asigna el valor a . Una tupla es localmente consistente si satisface todas las restricciones que existen sobre las variables de la tupla [9].

2.2.3. Gecode

Gecode es una librería multiplataforma, desarrollada en C++ y distribuida bajo la licencia MIT; es utilizada para desarrollar sistemas y aplicaciones basados en restricciones. Esta librería proporciona un ambiente de desarrollo genérico y flexible, característica que le permite acoplarse fácilmente con otros sistemas [18].

Además, Gecode puede ser adaptada a cualquier tipo de problema, debido a que proporciona las herramientas necesarias para implementar nuevas variables, propagadores, estrategias de distribución y motores de búsqueda; con el mismo nivel de eficiencia de los componentes predefinidos en su Kernel principal [18].

2.2.4. Drupal

Drupal [19] es un CMS (Sistema de Gestión de Contenidos, del inglés Content Management System). Un CMS básicamente es un programa que se encuentra instalado en un servidor web y que provee un sitio donde el diseño está separado del contenido.

Drupal nació en el año 2000 cuando el estudiante belga, Dries Butayert [20], creó un sistema de boletines para su universidad, aunque su lanzamiento oficial fue a principios del año siguiente. Drupal está escrito en el lenguaje de programación PHP [21], todo el contenido de este CMS y casi todas las configuraciones del mismo se guardan en una base de datos, que por lo general es del tipo MySQL [22], aunque puede ser de otros tipos. Su instalación puede realizarse sobre cualquier servidor web aunque preferiblemente se recomienda que sea sobre el servidor Apache [23]. El código fuente de Drupal está disponible bajo los términos de las Licencias de GNU GPLv2/GPLv3 (Licencia Pública General, en inglés: General Public License) [24], esto significa que todo el código puede ser usado libremente, modificado según las necesidades y distribuido. Su uso es gratuito.

Drupal se compone de un sistema base (núcleo de drupal, en inglés, drupal core) sobre el que se instalan y configuran los módulos, los cuales son pequeños programas que añaden funcionalidades al CMS. El sistema básico trae varios módulos y sólo basta con activarlos para hacer uso de ellos. Todo en Drupal puede configurarse, y seguramente existirá el módulo (o combinación de ellos) que hará que las cosas funcionen como se desea. Hacer un sitio en Drupal es como armar un rompecabezas de módulos. Con la misma base se puede crear un portal de noticias, un blog, un sitio de avisos clasificados o una web institucional. Se puede también, agregar bloques con publicidades, comentarios, foros de discusión, formularios de contacto, galerías de fotos, elementos multimedia, entre otros; la lista es tan larga como lo que uno imagine que se puede ver y hacer en un sitio web. Además cuenta con un sistema de permisos de usuarios sumamente flexible. Se pueden crear varios tipos de usuarios y darles diferentes permisos a cada uno de estos.

En la actualidad cientos de programadores dedican su tiempo a Drupal. La comunidad de desarrolladores trabaja con base al consenso, su objetivo es el desarrollo del sistema y no el lucro personal. El código es permanentemente revisado y corregido, tanto el sistema base como los módulos externos más usados, se corrigen y actualizan frecuentemente. La comunidad de usuarios es también muy activa y su propósito es la difusión del producto y la ayuda mutua. Si bien la implementación de las soluciones es compleja para los desarrolladores, las interfaces para el usuario final son claras y eficientes.

2.3. Marco Conceptual

2.3.1. Backtracking

Es un algoritmo general para encontrar todas (o algunas) soluciones a un problema computacional, que construye incrementalmente los candidatos a las soluciones, y abandona cada candidato parcial tan pronto como se determina que no es adecuado para completar una solución válida [3, 4].

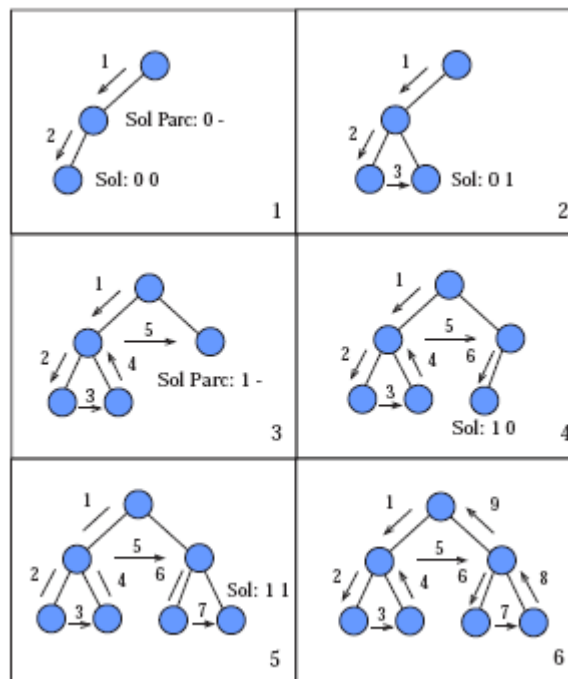


Figura 2.1: Recorrido paso a paso de un árbol binario por medio de un esquema básico de Backtracking [14].

En la figura 2.1, se muestra a manera de ejemplo como se comportaría un algoritmo de backtracking para resolver un problema donde el número de decisiones a tomar es dos y solo podría darse los casos de elegir o no elegir. Para éste caso los descendientes de un nodo se generan de izquierda a derecha [14].

2.3.2. Heurística

Una heurística es una técnica diseñada para resolver un problema ignorando si la solución es la mejor, pero que por lo general produce una buena solución o soluciona un problema más simple

que contenga la solución del problema más complejo. Las heurísticas están concebidas para conseguir el rendimiento de cómputo o simplicidad conceptual, potencialmente a costa de la exactitud y la precisión [17].

Un método heurístico puede cumplir su tarea mediante el uso de árboles de búsqueda. Sin embargo, en lugar de generar todas las ramas de posibles soluciones, una heurística selecciona las ramas con más probabilidades de producir resultados [11, 17].

Un enfoque dónde se pueden aplicar heurísticas es para encontrar el camino mas corto en un árbol de búsqueda. En este caso una heurística es una función matemática, $h(n)$ definida cada uno de los nodos del árbol, que sirve como una estimación del coste del camino más económico de un nodo dado hasta el nodo objetivo. El algoritmo de búsqueda escogerá el nodo que tiene el valor más bajo en la función heurística y expandirá los nodos que tienen el valor más bajo para $g(n) + h(n)$, donde $g(n)$ es el coste (exacto) del camino desde el estado inicial al nodo actual [16].

2.3.3. Propagación de Restricciones

En general, este procedimiento reemplaza un CSP dado, por uno mas simple y equivalente. La idea es que esa sustitución, si es eficiente, resulta beneficiosa, ya que la búsqueda posterior resultante de los llamados reiterados a los procedimientos de búsqueda se realizan sobre un espacio (por lo general) mas pequeño. El hecho de reemplazar un CSP por uno mas simple, se refiere a que los dominios y/o restricciones del CSP se hacen más pequeñas. La propagación de restricciones se realiza repetidamente, reduciendo los dominios y/o restricciones y manteniendo al mismo tiempo la equivalencia [1]

2.3.4. XML

XML es un lenguaje basado en el almacenamiento de datos. Utiliza una serie de etiquetas para definir, almacenar y "marcar" los datos que van a ser intercambiados entre aplicaciones. El marcado o etiquetado de los datos puede ser reconocido y utilizado por diferentes aplicaciones sin ningún problema. XML proporciona un tipo de lenguaje que se utiliza entre diferentes plataformas y lenguajes de programación para expresar mensajes complejos y funciones [2].

2.3.5. CGI

CGI (Interfaz de entrada común, en inglés Common Gateway Interface) es un programa de interfaz que permite al servidor de Internet utilizar programas externos para realizar una función específica. Este tipo de programas consisten generalmente en una serie de instrucciones escritas en un lenguaje de programación como C o PERL que procesan la petición de un navegador, ejecutan un programa y formatean los resultados en HTML de manera que puedan ser presentados en el navegador [25].

2.3.6. GSL

GSL (Biblioteca Científica de GNU, en inglés, GNU Scientific Library) es una librería escrita en C, destinada a cálculos numéricos, generalmente es utilizada en las áreas de las matemáticas y la ciencia. Es un software libre distribuido bajo la licencia GNU GPL. La librería ofrece una amplia gama de rutinas matemáticas tales como generadores de números aleatorios, funciones especiales, números complejos, combinatoria, álgebra lineal, integración y derivación numéricas, transformadas, estadística, entre otras. Esta librería posee más de 1000 funciones en total, con un extenso conjunto de pruebas [26].

Capítulo 3

Modelo Matemático

En el presente capítulo se definirá el problema de asignación de citas terapéuticas (ver sección 1.4.1) como un modelo de satisfacción por restricciones que cubra las características y propiedades presentadas en la sección 1.4.2:

3.1. Variables y parámetros

3.1.1. Parámetros de entrada

- E : Número total de especialidades del problema.
- N : Número total de pacientes.
- P : Número total de profesionales.
- $disP_i$: Conjunto de horarios en los que está disponible el paciente i . $1 \leq i \leq N$
- $EP_i = \{e_k : 1 \leq k \leq E\}$: Conjunto de especialidades en las que el paciente i necesita ser atendido. $1 \leq i \leq N$.
- $numC_{i,e}$: Número de citas que componen el tratamiento del paciente i para la especialidad e . $1 \leq i \leq N; e \in EP_i$.
- $pref_{i,e}$: Profesional de preferencia del paciente i , para la especialidad e . $1 \leq i \leq N; e \in EP_i$.
- $EProf_p = \{e_k : 1 \leq k \leq E\}$: Conjunto de especialidades en las que atiende el profesional p . $1 \leq p \leq P$.

- $hrrioProf_{p,e}$: Conjunto de horarios en los que el profesional p , atiende en la especialidad e . $1 \leq p \leq P; e \in EProf_p$.
- $maxP_e$: Número máximo de pacientes simultáneos que pueden atender los profesionales pertenecientes a la especialidad e . $1 \leq e \leq E$.
- D_e : Duración en slots de tiempo de las citas de la especialidad e . $1 \leq e \leq E$.
- $intervalosDia$: Número de intervalos de tiempo en los que se divide el día. (Generalmente cada intervalo representa una hora).
- $slotsIntervalo$: Número de slots en los que se divide un intervalo. (Generalmente cada intervalo esta compuesto por 12 slots)
- $semanas$: Número total de semanas disponibles para asignar la totalidad de las citas. (Generalmente se toma de Lunes a Sábado).

3.1.2. Parámetros derivados

En esta sección se especificarán aquellos valores que son calculados a partir de algunos de los parámetros de entrada:

- $slotsDia = intervalosDia * slotsIntervalo$: Número de slots de tiempo en que se divide el día.
- $makespan = semanas * intervalosSemana * slotsIntervalo$: Número total de slots de tiempo disponibles para asignar la totalidad de las citas.

3.1.3. Variables

- $inicio_{j,i,e}$: Tiempo de inicio de la j -ésima cita del paciente i , en el tratamiento perteneciente a la especialidad e . $1 \leq j \leq numC_{i,e}, 1 \leq i \leq N, e \in EP_i$.
- $fin_{j,i,e}$: Tiempo de finalización de la j -ésima cita del paciente i , en el tratamiento perteneciente a la especialidad e . $1 \leq j \leq numC_{i,e}, 1 \leq i \leq N, e \in EP_i$.
- $profP_{i,e}$: Profesional que atenderá todas las citas del paciente i , del tratamiento perteneciente a la especialidad e . $1 \leq i \leq N, e \in EP_i$.

- $pacProf_{p,s,e}$: Pacientes atendidos por el especialista p , en slot de tiempo s , en la especialidad e . $1 \leq p \leq P, 1 \leq s \leq makespan, e \in EProf_p$.
- $b_{i,e}$: Variable que indica si al paciente i le fueron asignadas las citas de su tratamiento en la especialidad e con su profesional de preferencia. $1 \leq i \leq N, e \in EP_i$.

3.1.4. Dominios

- $inicio_{j,i,e} \in \{0, \dots, makespan - 1\}$
- $fin_{j,i,e} \in \{1, \dots, makespan\}$
- $profP_{i,e} \in \{1, \dots, P\}$
- $pacProf_{p,s,e} \in \{1, \dots, N\}$
- $b_{i,e} \in \{0, 1\}$

3.2. Restricciones

3.2.1. Restricción Tiempo de Finalización

El tiempo de finalización de una cita debe ser igual al tiempo de inicio más la duración estipulada para todas las citas que pertenecen a una misma especialidad.

$$fin_{j,i,e} = inicio_{j,i,e} + D_e;$$

$$\forall j \in \{1, \dots, numC_{i,e}\}; \forall i \in \{1, \dots, N\}; e \in EP_i$$

El tiempo de finalización de la j -ésima cita del tratamiento del paciente i en la especialidad e , debe ser igual al tiempo de inicio de la misma cita del mismo paciente en la misma especialidad, más la duración estipulada para todas las citas de la especialidad e .

3.2.2. Restricción Pre-asignación de Tiempo según Pacientes

Cada asignación de una cita debe estar acorde con los horarios establecidos por el paciente para recibir su tratamiento, es decir, al asignar una cita se debe tener en cuenta la disponibilidad de un paciente.

Sea $c_{j,i,e} = \{inicio_{j,i,e}, \dots, inicio_{j,i,e} + a, \dots, fin_{j,i,e}\}; 0 < a < D_e$

$$\forall s \in c_{j,i,e} : s \in disP_i$$

$$\forall j \in \{1, \dots, numC_{i,e}\}; \forall i \in \{1, \dots, N\}; e \in EP_i$$

Sea $c_{j,i,e}$ el conjunto de slots que conforman la j -ésima cita del tratamiento del paciente i en la especialidad e .

Cada uno de los elementos del conjunto $c_{j,i,e}$ debe pertenecer al conjunto de horarios en los que está disponible el paciente i .

3.2.3. Restricción Pre-asignación de Tiempo según Profesionales

Cada asignación de una cita debe estar acorde con los horarios de atención del profesional al que le corresponde atenderla, es decir, al asignar una cita se debe tener en cuenta la disponibilidad del profesional.

Sean:

- $profP_{i,e} = p$
- $c_{j,i,e} = \{inicio_{j,i,e}, \dots, inicio_{j,i,e} + a, \dots, fin_{j,i,e}\}; 0 < a < D_e$

$$\forall s \in c_{j,i,e} : s \in hrioProf_{p,e}$$

$$\forall j \in \{1, \dots, numC_{i,e}\}; \forall i \in \{1, \dots, N\}; e \in EP_i \cap EProf_p$$

Sea p , el profesional que atenderá todas las citas del paciente i , del tratamiento perteneciente a la especialidad e . Además, $c_{j,i,e}$ el conjunto de slots que conforman la j -ésima cita del tratamiento del paciente i en la especialidad e .

Cada uno de los elementos del conjunto $c_{j,i,e}$ debe pertenecer al conjunto de horarios en los que está disponible el profesional p en la especialidad e .

3.2.4. Restricción No Solapamiento entre Citas

En caso que un paciente necesite tratamiento en varias especialidades, los horarios de las citas a asignar a este paciente para una especialidad determinada no se deben cruzar con los horarios de las demás citas asignadas en otras especialidades.

Sean:

- $c_{j,i,e} = \{inicio_{j,i,e}, \dots, inicio_{j,i,e} + a, \dots, fin_{j,i,e}\}; 0 < a < D_e$
- $Citas_{i,e} = c_{1,i,e} \cup \dots \cup c_{t,i,e}; t = numC_{i,e}$

Ademas, $\{\forall e \wedge \forall e' \in EP_i, e \neq e'\};$

$$Citas_{i,e} \cap Citas_{i,e'} = \emptyset$$

Sea $c_{j,i,e}$ el conjunto de slots que conforman la j -ésima cita del tratamiento del paciente i en la especialidad e . Además, sea $Citas_{i,e}$ el conjunto de citas que conforman la totalidad del tratamiento del paciente i en la especialidad e . Se debe cumplir que:

La intersección entre el conjunto de citas de un mismo paciente, en diferentes especialidades debe tener como resultado el conjunto vacío. El objetivo de esta restricción es evitar que dos citas pertenecientes a un mismo paciente en distintas especialidades sean al mismo tiempo.

3.2.5. Restricción Día Diferente

No se pueden asignar dos citas en un mismo día para un mismo paciente en la misma especialidad.

$$\left\lfloor \frac{fin_{j,i,e}}{slotsDia} \right\rfloor < \left\lfloor \frac{inicio_{j+1,i,e}}{slotsDia} \right\rfloor;$$

$$\forall j \in \{1, \dots, numC_{i,e}\}; \forall i \in \{1, \dots, N\}; e \in EP_i$$

El día al que pertenece el tiempo de finalización de la j -ésima del paciente i en la especialidad e , debe ser menor al día al que pertenece el tiempo de inicio de la siguiente cita del mismo paciente, en la misma especialidad.

3.2.6. Restricción de Capacidad

Los pacientes asignados a un profesional en ningún momento deben superar su límite de pacientes.

$$pacProf_{p,s,e} \leq maxP_e;$$

$$e \in EProf_p; \forall p \in \{1, \dots, P\}; \forall s \in \{1, \dots, slotsDia\}$$

El número de pacientes que atiende el profesional p , en el slot de tiempo s , debe ser menor o igual que el número máximo de pacientes simultáneos que pueden atender los profesionales pertenecientes a la especialidad e .

3.2.7. Función Objetivo: Restricción de Preferencia

Un paciente puede solicitar que sea atendido por un profesional de su preferencia.

$$b_{i,e} \Leftrightarrow profP_{i,e} = pref_{i,e}$$

$$\forall i \in \{1, \dots, N\}; e \in EP_i$$

$$max(\sum_{i=0}^N b_{i,e})$$

El objetivo para que la restricción de preferencia fuese planteada como una función de maximización, es establecer que el cumplimiento de ésta sea opcional. En otras palabras, la restricción de preferencia se establece como débil.

3.3. Heurísticas

3.3.1. Heurística para la Distribución Equilibrada de Pacientes

Para lograr una distribución equilibrada de pacientes entre los profesionales de una misma especialidad se implementó una heurística de selección de valor sobre la variable $profP_{i,e}$ que determina el profesional que atiende al paciente i en la especialidad e .

Para determinar el valor que debe tomar la variable $profP_{i,e}$, la heurística se basa en el número de pacientes asignados que tienen los profesionales; con base en este número, se determina la prioridad de asignación de pacientes para ellos. El profesional con menor número de pacientes será seleccionado como valor para la variable. A continuación se define la función que representa

el número de pacientes atendidos por el profesional p :

Sea C el instante de tiempo en el que se decide asignar un paciente a un profesional:

$$numPac_p = \sum_{s=0}^C pacProf_{p,s,e};$$

$$e \in EProf_p; \forall p \in \{1, \dots, P\}$$

3.3.2. Heurística para la Distribución Equilibrada de Trabajo

En la sección anterior, se presentó una heurística que busca una distribución equilibrada de pacientes entre los profesionales, el objetivo principal de esta heurística es lograr que el número de pacientes asignado a los profesionales de una misma especialidad sea el más equitativo posible. Sin embargo, ésta no asegura que el porcentaje de ocupación de los profesionales sea equilibrado, pues aunque los profesionales de una misma especialidad tengan el mismo número de pacientes asignados, puede suceder que algunos pacientes tengan más citas que otros, lo que causaría que algunos profesionales estén ocupados más tiempo que otros.

Para determinar el valor que debe tomar la variable $profP_{i,e}$, la heurística se basa en el número de citas atendidas que tienen los profesionales, con base en este número, se determina la prioridad de asignación de pacientes para ellos. El especialista con menor número de citas será seleccionado como valor para la variable. A continuación definimos la función que representa el número de citas atendidas por el profesional p :

Sea C el instante de tiempo en el que se decide asignar un paciente a un profesional:

$$nCitas_p = \sum_{s=0}^C \sum_{i=1}^{pacProf_{p,s,e}} numC_{i,e};$$

$$e \in EProf_p \cap EP_i; \forall p \in \{1, \dots, P\}$$

Capítulo 4

Implementacion del Modelo

En este capítulo se presentará la implementación del modelo de satisfacción de restricciones del capítulo 3. Para implementarlo, se utilizó el lenguaje de programación C++ junto con la versión 3.7.1 de Gecode. En las siguientes secciones se explicarán las estructuras de datos utilizadas, la manera en que son declaradas las variables, la imposición de las restricciones y las estrategias de distribución implementadas.

4.1. Estructuras de Datos

Para poder encontrar una solución al problema de asignación de citas terapéuticas, el motor de búsqueda necesita tener definidos algunos datos antes de aplicar las estrategias de distribución y propagación. Estos datos, son especificados por medio de un archivo de entrada (el formato de este archivo es especificado en el Apéndice A)

Para lograr una mejor organización de los datos del archivo de entrada se crearon 3 clases: Especialidad, Paciente y Profesional, cada una de estas clases contiene a su vez estructuras de datos propias de C++. A continuación, se detallarán las estructuras de datos más importantes utilizadas por cada clase:

1. Clase Paciente:

- `int id`: Representa el identificador de cada paciente.
- `int nTratamientos`: Indica el número de tratamientos que necesita el paciente.

- `vector<vector<int>>` `infoTratamientos`: Matriz que contiene toda la información correspondiente a los tratamientos en los que necesita ser atendido el paciente. Cada subvector representa un tratamiento, en el cual las posiciones están distribuidas de la siguiente manera:
 - Pos 0: ID de la especialidad a la que pertenece el tratamiento.
 - Pos 1: Número de citas que componen el tratamiento de dicho paciente.
 - Pos 2: Duración en minutos de cada una de las citas.
 - Pos 3: ID del profesional de preferencia en la especialidad a la que pertenece el tratamiento.
- `vector<int>` `disponibilidad`: Conjunto de horarios en los que esta disponible el paciente. Cada posición del vector representa un intervalo de tiempo de una hora, si el paciente esta disponible en ese intervalo de tiempo, el valor en la posición de dicho intervalo es 1, de lo contrario el valor es 0. Una explicación más amplia del formato de las disponibilidades se encuentra en el Apéndice A.3.

2. Clase Profesional:

- `int id`: Representa el identificador de cada profesional.
- `int nEspecialidades`: Indica el número de especialidades en las que atiende el profesional.
- `vector<int>` `idEspecialidades`: Estructura que almacena todos los identificadores de las especialidades en las que atiende el profesional.
- `vector<vector<int>>` `horariosAtencion`: Matriz que almacena los diferentes horarios de atención en las especialidades que atiende el profesional. Cada subvector representa el horario de atención en una especialidad, y al igual que el vector de disponibilidades del paciente, cada posición de dicho subvector representa un intervalo de tiempo. Una explicación más amplia del formato de los horarios de atención de los profesionales se encuentra en el Apéndice A.3.

3. Clase Especialidad:

- `int id`: Representa el identificador de cada especialidad. `int totalCitas`: Número total de citas que se deben asignar para la especialidad.
- `int duracionCitas`: Duración en minutos de las citas de la especialidad.

- `int nEspecialistas`: Total de profesionales que atienden en la especialidad.
- `int nPacientes`: Total de pacientes que necesitan tratamiento en la especialidad.
- `int capacidad`: Limite de pacientes que pueden atender los profesionales de la especialidad.
- `vector<Profesional> profesionales`: Lista de profesionales que atienden en la especialidad.
- `vector<Paciente> pacientes`: Lista de pacientes que necesitan ser atendidos en la especialidad.
- `vector<int> idProfesionales`: Lista con todos los identificadores de los profesionales que atienden en la especialidad.
- `vector<int> idPacientes`: Lista con todos los identificadores de los pacientes que requieren de un tratamiento en la especialidad.

Adicionalmente a los parámetros incluidos en las 3 clases anteriores, existen otros parámetros que ayudan a determinar la equivalencia entre las unidades de tiempo utilizadas por el modelo y las unidades de tiempo utilizadas en el mundo real:

- `int intervalosDia`: Número de intervalos de tiempo en los que se divide el día. Generalmente cada intervalo de tiempo representa 60 minutos.
- `int slotsIntervalo`: Número de slots de tiempo en los que se divide un intervalo.
- `int slotsDia`: Número de slots de tiempo en que se divide el día.
- `int semanas`: Número total de semanas disponibles para asignar la totalidad de las citas.
- `int intervalosSemana`: Número total de intervalos en los que se divide una semana.
- `int makespan`: Número total de slots de tiempo disponibles para asignar la totalidad de las citas. Es calculado a partir de los parámetros: `semanas`, `intervalosSemana`, `intervalosDia` y `slotsDia`.

4.2. Declaración de Variables

Las estructuras especificadas en la sección anterior, sirvieron como base para la construcción de las estructuras que contienen las variables de dominio finito, sobre las cuales se quiere hacer el

proceso de propagación y distribución. Las estructuras proporcionadas por Gecode usadas en esta implementación fueron:

- **IntVar:** Representa una variable entera. Su dominio debe estar dentro de un rango de números enteros. Un ejemplo de dominio puede ser: $\{-4, \dots, 20\}$.
- **IntVarArray:** Es un arreglo o vector de variables enteras.
- **IntArgs:** Representa un vector de argumentos enteros.
- **IntSet:** Representa un conjunto de variables enteras. A diferencia de las variables enteras su dominio no es un rango de números enteros, sino que es un conjunto de conjuntos de números enteros. Un ejemplo de dominio puede ser: $\{\{\}, \{2\}, \{1, 2\}, \{1, 2, 3\}\}$.

Por medio de estas estructuras suministradas por Gecode, se implementaron las variables del modelo que fueron planteadas en la sección 3.1.3. A continuación, se listan las estructuras de datos utilizadas para la declaración de las variables y sus correspondientes dominios:

- **IntVarArray inicio:** Vector de variables enteras que representa los tiempos de inicio de todas las citas de la totalidad de pacientes. Está declarado de la siguiente manera:

$$IntVarArray inicio(totalCitas, 0, makespan - 1)$$

El primer parámetro indica el tamaño del vector, es decir, se creará una variable por cada cita que se tenga que asignar.

Los siguientes parámetros indican el dominio que tendrá cada variable, en este caso será desde el slot 0 hasta el penúltimo slot de tiempo.

- **IntVarArray fin:** Vector de variables enteras que representa los tiempos de finalización de todas las citas de la totalidad de pacientes. Está declarado de la siguiente manera:

$$IntVarArray fin(totalCitas, 1, makespan)$$

El primer parámetro indica el tamaño del vector, es decir, se creará una variable por cada cita que se tenga que asignar.

Los siguientes parámetros indican el dominio que tendrá cada variable, en este caso será desde el slot 1 hasta el último slot de tiempo.

- `IntVarArray` profesionales: Vector de variables enteras que representa el especialista que le corresponde atender cada cita.

*IntVarArray*profesionales(*totalCitas*, *listaCodigosEspecialistas*)

El primer parámetro indica el tamaño del vector, es decir, se creará una variable por cada cita que se tenga que asignar.

El segundo parámetro representa un *IntSet*, cada elemento dentro de este conjunto representa un identificador perteneciente a un especialista determinado. Este conjunto indica que el dominio de cada variable entera es el conjunto de identificadores de los especialistas.

Como se puede observar, las variables anteriores tienen el compendio del total de citas de todas las especialidades y sobre éstas se aplicarán las estrategias de distribución (heurísticas). Sin embargo, aplicar las restricciones a dichas variables sería un proceso dispendioso, debido a que cada especialidad tiene parámetros propios y condiciones específicas para poder aplicar las restricciones. Por esta razón, se crearon variables auxiliares utilizando estructuras de datos similares a las utilizadas en las variables anteriores. A continuación, se listan las estructuras de datos utilizadas para la declaración de las variables auxiliares:

- `vector<Especialidad>` `lstEspecialidades`: Aunque no es una variable propia de Gecode, esta estructura de datos almacena todas las especialidades provenientes del archivo de entrada. Esta estructura de datos es fundamental para la inicialización de algunos parámetros auxiliares.
- `vector<IntVarArray>` `inicioEsp`: Lista de vectores de variables enteras que representa los tiempos de inicio de todas las citas de la totalidad de pacientes de una especialidad determinada. Cada vector dentro de la lista representa una especialidad distinta.
- `vector<IntVarArray>` `finEsp`: Lista de vectores de variables enteras que representa los tiempos de finalización de todas las citas de la totalidad de pacientes de una especialidad determinada. Cada vector dentro de la lista representa una especialidad distinta.
- `vector<IntVarArray>` `profesionalesEsp`: Lista de vectores de variables enteras que representa el especialista que le corresponde atender cada cita perteneciente a una especialidad determinada. Cada vector dentro de la lista representa una especialidad distinta.
- `vector<IntArgs>` `durEsp`: Lista de vectores de argumentos enteros. En este caso, cada argumento dentro de un vector representa la duración de una cita en determinada especialidad.

Cada vector es inicializado de la siguiente manera:

```
IntArgs(lstEspecialidades[i].totalCitas(), toSlots(lstEspecialidades[i].duracionCitas()))
```

El primer parámetro indica el tamaño del vector, es decir, se creará un argumento por cada cita que se tenga que asignar en la especialidad *i*.

El siguiente parámetro indica el valor que tendrá cada argumento, en este caso será la duración de las citas para la especialidad *i*, dada en slots de tiempo.

4.3. Implementación de las Restricciones

Para realizar la implementación de cada una de las restricciones establecidas en la sección 3.2, se utilizaron los propagadores predefinidos por *Gecode*. A medida que se presente la implementación de cada restricción, se irá explicando brevemente la función de cada propagador. Es importante mencionar, que en las siguientes secciones se omitieron porciones de código, especialmente de aquellas que están relacionadas con el preprocesamiento que se debe realizar antes de aplicar cada propagador, con el objetivo de dar mayor claridad y entendimiento sobre la implementación de cada restricción y su funcionamiento.

4.3.1. Restricción Tiempo de Finalización

La función de esta restricción es establecer el tiempo de finalización de todas las citas de cada especialidad por medio del propagador **rel**[5], el cual es utilizado para establecer un tipo de relación entre variables y valores enteros. La implementación de esta restricción tiene la siguiente forma:

```
1 rel(home, finEsp[i][j] == inicioEsp[i][j] + durEsp[i][j], ICL_DOM)
```

- El primer parámetro indica el espacio de cómputo en el que se almacenarán las variables, sus dominios y la restricción.
- El segundo parámetro corresponde a una expresión entera [5] que establece una relación de igualdad entre la variable que representa el tiempo de finalización de la cita *j* de la especialidad *i*, y la suma entre la variable que representa el tiempo de inicio y la duración de la misma cita para la misma especialidad.
- El tercer parámetro indica el nivel de consistencia [6] que deben cumplir los dominios de las variables involucradas en la restricción. Para este caso se utilizó dominio consistencia [5].

4.3.2. Restricción Pre-asignación de Tiempo según Pacientes

Esta restricción se encarga de establecer los slots de tiempo en los cuales, las citas para cada paciente, no pueden ser asignadas debido a que éste no se encuentra disponible.

Del mismo modo que la restricción anterior, la implementación de la restricción se hizo utilizando el propagador **rel**[5]:

```

1  for (int slot = 0; slot < disponibilidad.size(); slot++) {
2      if ( disponibilidad[slot] == 0) {
3          rel(home, inicioEsp[i][n], IRT_NQ, slot, ICL_DOM)
4      }
5  }
```

- El primer parámetro indica el espacio de cómputo en el que se almacenarán las variables, sus dominios y la restricción.
- El segundo parámetro corresponde al tiempo de inicio de la cita n de un paciente determinado para la especialidad i .
- El tercer parámetro establece el tipo de relación entre el segundo parámetro y el cuarto parámetro; en esta ocasión el valor *IRT_NQ* establece que el valor que tome la variable *inicioEsp*[i][n] no puede ser igual a el valor del argumento slot.
- El cuarto parámetro toma el valor de todos los slots de tiempo en los que no está disponible el paciente al que le corresponde la cita n .
- El quinto parámetro indica el nivel de consistencia[6] que deben cumplir los dominios de las variables involucradas en la restricción. Para este caso se utilizó dominio consistencia[5].

4.3.3. Restricción Pre-asignación de Tiempo según Profesionales

Esta restricción se encarga de establecer los slots de tiempo en los cuales las citas asignadas a cada profesional no pueden ser asignadas, debido a que dichos slots no pertenecen al conjunto de horarios en los que el profesional atiende en la especialidad.

Para la implementación de esta restricción, se utilizaron dos propagadores de Gecode: **wait**[5] y **rel**:

```

1  wait(home, profesionales, &recorteDisponibilidad, ICL_DOM)
```

La principal función del propagador **wait** es indicar que sólo se aplicará el propagador existente en la función *recorteDisponibilidad* cuando todas las variables dentro del arreglo especialistas estén asignadas. Una vez se conoce el especialista asignado a una cita determinada, se puede proceder a recortar el dominio de la variable *inicioEsp* por medio del propagador **rel**:

```

1 //funcion recorteDisponibilidad:
2
3 Profesional prof = lstEspecialidades[i].buscarProfesional(profesionales
4     [j]);
5 for (int slot = 0; slot < prof.horariosAtencion[i].size(); slot++) {
6     if ( horariosAtencion[i][slot] == 0) {
7         rel(home, inicio[j], IRT_NQ, slot, ICL_DOM)
8     }
9 }

```

como se puede observar, la función *recorteDisponibilidad*, establece el profesional asignado a la cita *j* en la especialidad *i* y el horario de atención de dicho profesional, posteriormente se realiza un recorte similar al realizado en la implementación de la restricción anterior.

4.3.4. Restricción No Solapamiento entre Citas

Esta restricción únicamente es aplicada a aquellos pacientes que necesitan ser atendidos en más de una especialidad, su objetivo es asegurar que no se crucen los horarios de las citas que pertenecen a especialidades distintas.

```

1 infoT = paciente.infoTratamientos();
2 // infoT[espAct][2] Duracion en minutos de la citas de la especialidad
3   actual.
4 for ( int k = 0; k < toSlots(infoT[espAct][2]); k++) {
5     rel(home, inicioEsp[espSig][j] != inicioEsp[espAct][i] + k,
6         ICL_DOM)
7 }

```

Una vez establecidos los pacientes que necesitan tratamiento en más de una especialidad, se procede a aplicar la restricción por medio del propagador **rel**. Los parámetros que se utilizan en el propagador son:

- El primer parámetro indica el espacio de cómputo en el que se almacenarán las variables, sus

dominios y la restricción.

- En el segundo parámetro tanto el iterador *espAct* como el iterador *espSig* indican especialidades pertenecientes a un mismo paciente, el iterador *espAct* comienza en la primera de las especialidades en las que el paciente necesita tratamiento y termina en la penúltima. Por su parte, el iterador *espSig* comienza en la segunda especialidad y termina en la última. Los iteradores *i* y *j* recorren cada una de las citas de ambas especialidades. Además, el iterador *k* se encarga de recorrer cada uno de los slots que se encuentran entre el tiempo de inicio y el tiempo de finalización de cada una de las citas de la especialidad representada por el iterador *espAct*.

La función de la expresión entera[5] que representa el segundo parámetro, es la de establecer una relación de desigualdad entre la duración de la cita *i* en la especialidad *espAct* y el tiempo de inicio de la cita *j* en la especialidad *espSig*.

- El tercer parámetro indica el nivel de consistencia[6] que deben cumplir los dominios de las variables involucradas en la restricción. Para este caso se utilizó dominio consistencia[5].

4.3.5. Restricción Día Diferente

La función de esta restricción es evitar que dos o más citas pertenecientes a uno de los tratamientos de un paciente determinado, sean asignadas en un mismo día. En la implementación de la restricción día diferente, se utilizaron 2 tipos de propagadores, el propagador **div**[5], que determina el día de cada cita y el propagador **rel**, establece que el resultado del propagador **div** para 2 citas sea diferente:

```
1 div(home, inicioEsp[i][n], intervalosDia, diaC1, ICL_DOM)
2 div(home, inicioEsp[i][n+1], intervalosDia, diaC2, ICL_DOM)
```

El propagador **div**, establece una división entera entre el segundo y el tercer parámetro de entrada, a la vez que almacena el resultado en el cuarto parámetro. Para este caso, el valor que se almacene en el cuarto parámetro, corresponderá al día en que es asignada cada cita.

Una vez obtenido el día en que es asignada cada cita, se debe asegurar que todas las citas de un paciente que pertenecen a un mismo tratamiento sean asignadas en días distintos, para cumplir con este propósito es utilizado el propagador **rel**:

```
1 rel(home, diaC1, IRT_LE, diaC2, ICL_DOM)
```

Como se puede observar, en lugar de establecer una relación de diferencia (por medio del parámetro *IRT_NQ*) entre las variables *diaC1* y *diaC2*, se establece una relación de precedencia, (por medio del parámetro *IRT_LE*) de manera que la cita n siempre será asignada al menos un día antes que la cita $n + 1$. La finalidad de esta relación de precedencia es evitar simetrías.

4.3.6. Restricción de Capacidad

Esta restricción se encarga de garantizar que cada uno de los especialistas no sobrepase el límite de pacientes simultáneos que puede atender. Para implementar esta restricción se utilizó el propagador **cumulatives**[5].

Cumulatives está diseñado como un propagador genérico para cualquier problema de *scheduling*, en el cual se considera que existe un conjunto de tareas que necesitan ser procesadas en un conjunto de máquinas. Donde cada tarea puede consumir uno o más recursos de la máquina en la cual se procesará. Para el problema de asignación de citas terapéuticas, se considera que el conjunto de tareas, es la totalidad de citas a asignar por especialidad, donde un paciente se ve representado en cada cita. Además, se considera que el conjunto de máquinas es representado por los profesionales que atienden en la especialidad. El propagador *cumulatives* se utilizó de la siguiente manera:

```

1  cumulatives(home,
2      profesionalesEsp[i],
3      inicioEsp[i],
4      durEsp[i],
5      finEsp[i],
6      capacidadEsp[i],
7      ICL_DOM)

```

Los parámetros que se utilizan en el propagador son:

- *profesionalesEsp*: Representa al conjunto de profesionales que atienden en dicha especialidad.
- *inicioEsp*: Conjunto de tiempos de inicio. Cada elemento de este conjunto indica el tiempo de inicio de cada cita.
- *durEsp*: Conjunto de tiempos de duración. Cada elemento de este conjunto indica la duración establecida para cada cita.

- *finEsp*: Conjunto de tiempos de finalización. Cada elemento de este conjunto indica el tiempo de finalización de cada cita.
- *capacidadEsp*: Corresponde a una lista de argumentos enteros, donde cada argumento representa el límite de pacientes simultáneos que puede atender un profesional.
- *ICL_DOM*: Indica el nivel de consistencia[6] que deben cumplir los dominios de las variables involucradas en la restricción. Para este caso se utilizó dominio consistencia[5].

4.3.7. Función Objetivo

La implementación de la función objetivo se divide en dos partes. La primera parte se implementa por medio de una restricción reificada, dicha restricción establece que, de ser posible, cada paciente debe ser atendido por el profesional de su preferencia y a su vez, asocia el cumplimiento de dicha restricción a una variable booleana. La segunda parte de la implementación maximiza el número de veces que es satisfecha la restricción reificada, en otras palabras, busca maximizar el número de pacientes a los cuales les es asignado su profesional de preferencia.

```

1  if (paciente.tienePreferencia()) {
2      infoT = paciente.infoTratamientos();
3      // infoT[i][3] id del profesional de preferencia en la especialidad
        i
4      rel(home,
5          (profesionalesEsp[i][j] == infoT[i][3]) == preferencias[p],
6          ICL_DOM);
7  }
8  rel(home, maximo == sum(preferencias), ICL_DOM);

```

Como se aprecia, la restricción reificada es activada sólo si el paciente tiene algún profesional de preferencia, además, es implementada a través del propagador **rel**, por medio del cual se establece que el profesional a asignar que atenderá en la cita j de la especialidad i al paciente, sea el que éste ha seleccionado como su profesional de preferencia. Posteriormente se iguala la restricción a una variable booleana.

Para finalizar se almacena la suma de todas las variables booleanas que almacenan el cumplimiento de la restricción de preferencia en una variable entera *maximo*. Esta variable será maximizada por Gecode a través del motor de búsqueda **BAB** (Branch and Bound) y el Script de Maximización **MaximizeScript**.

4.4. Implementación de las Heurísticas

En esta sección se presenta la implementación de cada una de las heurísticas definidas en la sección 3.3. Al igual que en la implementación de las restricciones, para las heurísticas se utilizaron funciones predefinidas por *Gecode*. A medida que se presente la implementación de cada heurística, se irá explicando brevemente la función de cada función.

4.4.1. Heurística de Distribución Equilibrada de Pacientes

Como su nombre lo dice, la finalidad de esta heurística es lograr una distribución equilibrada de pacientes entre los especialistas pertenecientes a una misma especialidad. La heurística se implementó por medio del siguiente algoritmo:

```
1  DistribucionEquilibradaPacientes(IntVarArray x) {
2      /*
3       * Almacena el numero de pacientes que tiene cada especialista en
4       * cada
5       * especialidad
6       */
7      vector<map<int, int> > numPacientes;
8      int * codigos; //arreglo que almacena los codigos de los
9                      especialistas
10     int id; //Indica el identificador del especialista
11     int esp; //Indica la especialidad a la que pertenece cada variable
12             de x
13     for (int i = 0; i < x.size(); i++) {
14         if (!x[i].assigned()) {
15             esp = determinarEspecialidad(i);
16             codigos = lstEspecialidades[esp].idProfesionales();
17             id = determinarProfesional(numPacientes[esp], codigos);
18             numPacientes[esp][id] += 1;
19             return new PosVal(home, i, id);
20         }
21     }
22 }
```

El algoritmo *DistribucionEquilibradaPacientes* recibe un arreglo de variables enteras que representa a los profesionales que atenderán en la totalidad de las citas a asignar. En primer lugar, el algoritmo busca aquellas citas pertenecientes a pacientes que no tienen asignado un profesional para su tratamiento. En segundo lugar, se determina la especialidad a la que pertenece dicha cita y se obtienen los identificadores de los profesionales pertenecientes a esa especialidad. En tercer lugar, se procede a determinar el profesional que tiene menos pacientes asignados. Por último, se suma una unidad al contador de pacientes del especialista y se asigna el profesional a la variable que representa la cita.

La asignación de cada variable, se hace por medio de la clase *PosVal*, la cual es una implementación de una clase propia de Gecode llamada *Choice*, su principal función es asignar a la variable ubicada en la posición que se indique en el segundo parámetro, el valor indicado en el tercer parámetro. Para este caso, se asigna el especialista que representa la variable *id*, a la variable ubicada en la posición *i* del vector *x*.

La función auxiliar de mayor importancia que utiliza el algoritmo de distribución equilibrada de pacientes, es *determinarProfesional*, la cual determina el profesional a asignar a la variable que representa una de las citas pertenecientes al tratamiento de alguno de los pacientes. La función se implementó a través del siguiente algoritmo:

```
1  determinarProfesional(map<int, int> contador, vector<int> codigos) {
2      //Indica el identificador con el menor valor dentro del hash map
3      int idProf;
4      //Almacena el menor valor dentro de la variable contador
5      int menor = MAX_INTEGER;
6      for (int i = 0; i < numPacientes.size(); i++) {
7          if (contador[codigos[i]] < menor) {
8              contador[codigos[i]] = menor;
9              idProf = codigos[i];
10         }
11     }
12     return idProf;
13 }
```

Antes de especificar la manera en que actúa la función *determinarProfesional*, es importante mencionar que la variable *contador*, es de tipo *hash map*, el cual es un objeto que asocia un código

o llave con un valor determinado, en este caso el atributo que actúa como llave en el *hash map* es el id de cada profesional, y el atributo que actúa como valor es el número de pacientes que le corresponden a dicho profesional.

El algoritmo *determinarProfesional*, busca en la variable *contador* la llave asociada al menor valor, en consecuencia, esta función encuentra el identificador del especialista que tenga el menor número de pacientes. Esta función auxiliar es invocada cada vez que se va a asignar un profesional a un paciente determinado, de esta manera, se cumple con el objetivo de lograr una distribución equilibrada de pacientes.

4.4.2. Heurística para la Distribución Equilibrada de Trabajo

Como se mencionó en la sección 3.3.2, la finalidad de heurística es lograr una distribución equilibrada de citas entre los especialistas pertenecientes a una misma especialidad. La implementación de esta heurística es muy similar a la implementación de la anterior:

```

1  DistribucionEquilibradaTrabajo(IntVarArray x) {
2      /*
3       * Almacena el numero de citas que tiene cada especialista en cada
4       * especialidad
5       */
6      vector<map<int, int> > numCitas;
7      int * codigos; //arreglo que almacena los codigos de los
          especialistas
8      int id; //Indica el identificador del especialista
9      Paciente actual; //Indica el paciente actual
10     int esp; //Indica la especialidad a la que pertenece cada variable
          de x
11     for (int i = 0; i < x.size(); i++) {
12         if (!x[i].assigned()) {
13             esp = determinarEspecialidad(i);
14             p = determinarPaciente(i);
15             infoT = p.infoTratamientos();
16             codigos = lstEspecialidades[esp].idProfesionales();
17             id = determinarProfesional(numCitas[esp], codigos);
18             // infoT[esp][1] numero de citas en la especialidad

```

```
19         numero esp
20         numCitas[esp][id] += infoT[esp][1];
21         return new PosVal(home, i, id);
22     }
23 }
```

El algoritmo *DistribucionEquilibradaTrabajo* recibe un arreglo de variables enteras que representa a los profesionales que atenderán en la totalidad de las citas a asignar. En primer lugar, el algoritmo busca aquellas citas pertenecientes a pacientes que no tienen asignado un profesional para su tratamiento. En segundo lugar, se determina la especialidad y el paciente a la que pertenece dicha cita y se obtienen los identificadores de los profesionales pertenecientes a la especialidad. En tercer lugar, se procede a determinar el profesional que tiene menos citas asignadas. Por último, se suma una unidad al contador de pacientes de citas del especialista y se asigna el id del profesional a la variable que representa la cita.

Del mismo modo que la implementación de la heurística anterior, este algoritmo también hace uso de la clase *PosVal* para asignar el profesional representado por la variable *id* a la cita representada por la variable ubicada en la posición *i* del vector *x*. Además, el algoritmo de distribución equilibrada de trabajo, también utiliza la función auxiliar *determinarProfesional*, esta función no fue modificada para este algoritmo, sigue el mismo funcionamiento descrito en la anterior sección.

Capítulo 5

Descripción General del Sistema

El sistema desarrollado denominado ACiT(Asignador de Citas Terapéuticas), básicamente es un sistema CGI [25] que consta de dos aplicaciones interdependientes. La primera, es una aplicación web cuya finalidad es la de ofrecer la interfaz gráfica al usuario facilitando de esta manera, la entrada de los datos y posterior salida de los resultados del proceso de asignación de las citas. La segunda parte, es una aplicación de escritorio, un programa ejecutable que resulta de la compilación de los archivos fuente programados en el lenguaje C++ [27], utilizando el motor de restricciones Gecode [18].

El proceso de asignación de las citas inicia a partir del ingreso de un archivo de entrada a través de la aplicación web o de la selección de uno de los archivos de prueba que proporciona la misma. (El formato del archivo de entrada está descrito en el Apéndice A)

Dicho archivo es procesado posteriormente por la aplicación de escritorio que genera, en caso de existir solución, un archivo de salida en formato xml con los resultados del proceso de asignación. (El formato del archivo de salida está descrito en el Apéndice B). Esta información es transformada en distintas vistas en la aplicación web para facilitar el análisis y comprensión de la solución, y para lograr este objetivo se utilizan mecanismos como: gráficas, tablas, y calendarios.

5.1. Arquitectura

La Figura 5.1, presenta el esquema con la relación de los componentes de la aplicación. Al lado del cliente se puede apreciar la interacción que realiza el cliente con el navegador web; en éste, el usuario digita la dirección del portal del grupo de investigación AVISPA y selecciona la opción

Aplicación del Bloque de ACiT. Una vez realizado esto, al lado del servidor web se procesan las instrucciones necesarias para mostrar un formulario al usuario, que despliega los parámetros de entrada que requiere la aplicación para su correcta ejecución.

Posteriormente, se ingresa o selecciona un archivo de entrada; luego, se elige el número máximo de tiempo en el que se desea asignar las citas (este es el número de semanas); y seguidamente, si se quiere, se habilita la Restricción de Preferencia, cuya finalidad es la de establecer cierto nivel de prioridad a aquellos pacientes que han decidido preferir un profesional para la atención de su tratamiento. Cuando se realiza todo este proceso, la aplicación alojada en el CMS Drupal, gracias a la función `exec` de PHP, efectúa el llamado al programa ejecutable que resulta de la compilación de los archivos fuente implementados en el lenguaje de programación C++.

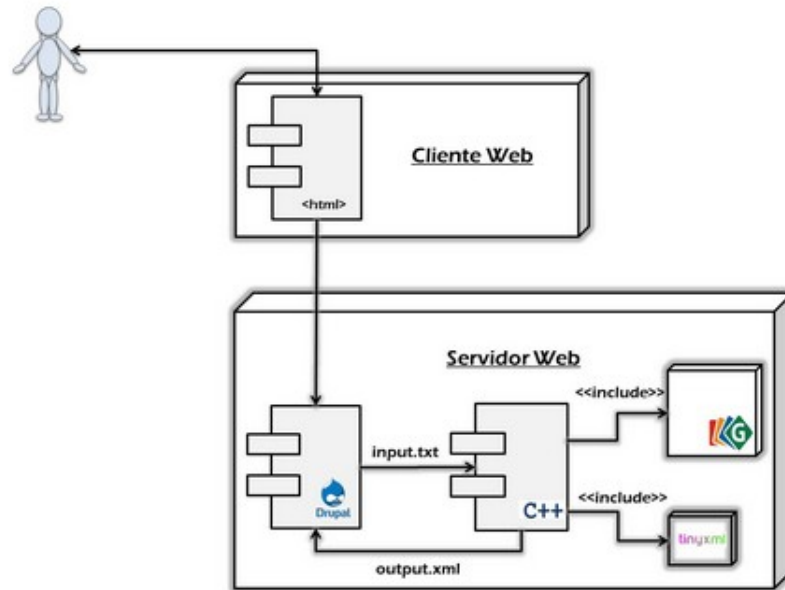


Figura 5.1: Arquitectura de la Aplicación.

Los archivos fuente que se relacionan directamente con la lógica del proceso de asignación de las citas, hacen uso de las librerías que provee el motor de restricciones Gecode [18]. Específicamente dichas librerías son las siguientes: `gecodedriver`, `gecodesupport`, `gecodeint`, `gecodekernel`, `gecodeset`, `gecodeminimodel`, `gecodesearch` y `gecodegist`.

Luego de que se ejecuta la aplicación de escritorio, gracias a las bondades del parser TinyXML [37], se genera un archivo de salida en formato xml, el cual es almacenado en una ubicación es-

pecífica en el servidor y posteriormente es procesado por la aplicación web, facilitando de esta manera, la visualización de los resultados por parte del usuario.

5.2. Detalles de la Implementación

El desarrollo de ACiT comenzó con la elaboración de un CSP [9] en el cual se definen las variables, dominios y restricciones que se encontraron en el problema de Asignación de Citas Terapéuticas (ver capítulo 3). Al evidenciar dichos elementos se procede a diseñar el formato del archivo de entrada de la aplicación, cuya especificación se detalla en el Apéndice A. En vista de que se hacía dispendioso crear distintos escenarios de prueba de forma manual, se decidió implementar un generador de archivos de prueba para facilitar dicho proceso.

El generador cuenta con una base de datos con aproximadamente cinco mil (5.000) registros, cada uno con un identificador y un nombre. Según la información suministrada por la Clínica de Rehabilitación del Valle, actualmente cuenta con cerca de diez (10) especialidades cada una con un número de profesionales de la salud determinado. La información detallada de las especialidades se encuentran en el Apéndice C. Es importante resaltar que según las especificaciones, existen profesionales que están en la facultad de prestar su servicio de atención como máximo en tres (3) especialidades diferentes y que los horarios de atención son distintos en cada una de ellas.

Debido a que no se tuvo acceso a información de vital importancia como el número de especialidades de cada profesional y horarios de atención; estas y otras consideraciones se realizaron de forma aleatoria. (Ver Apéndice D. Especificaciones del Generador). El resultado de la ejecución del generador es un archivo de texto que define los datos necesarios para la implementación y posterior ejecución del modelo.

Como se mencionó al inicio de este capítulo, el sistema esta dividido en dos partes: la implementación de la segunda está descrita en el capítulo 4, mientras que para la parte relacionada con la aplicación web, se optó por implementar la metodología del grupo de investigación AVISPA [28–30] ya que facilita en gran medida al desarrollo y acceso a la aplicación. A continuación, se especificarán las tecnologías de dicha metodología que fueron utilizadas y la forma en que fueron implementadas.

Las tecnologías adoptadas de la metodología fueron las siguientes:

5.2.1. Drupal

Como se mencionó en el marco teórico, Drupal[19] es un gestor de contenido que permite fácilmente la integración de Frameworks y código PHP dentro de sus páginas. En éste se despliega la aplicación del Asignador de Citas Terapéuticas, al igual que aparecen las distintas aplicaciones que ya hacen parte del portal del Grupo de Investigación AVISPA.

5.2.2. PHP

PHP[21] es un lenguaje de programación interpretado que facilita la creación de páginas web con contenido dinámico. Sus principales objetivos dentro de la aplicación son el de obtener la información suministrada por el usuario, procesarla, posteriormente hacer el llamado de la aplicación de escritorio que se encuentra al lado del servidor, y finalmente servir como mecanismo para mostrar los resultados.

5.2.3. XML

XML[31] es un lenguaje que garantiza el intercambio de información entre distintas aplicaciones. El archivo de salida de la aplicación de escritorio es definido en este formato, y es procesado para mostrar los resultados.

5.2.4. JQuery

JQuery[32] es una librería gratuita de código abierto que cuenta con una gran cantidad de complementos que permiten realizar diversas funciones interactivas con gran facilidad. Se utilizó principalmente, con la finalidad de proporcionar efectos agradables para el usuario y para implementar la función de autocompletado en el calendario de los pacientes.

5.2.5. Datatables

Datatables[33] son un complemento de jQuery que permite dibujar tablas con distintos elementos como paginado, opciones de búsqueda, ordenación por columnas, entre otros. ACiT cuenta básicamente con tres tablas que serán descritas a continuación:

- Tabla de datos de las Especialidades: Contiene la información básica de cada especialidad: un identificador, que detalla el valor único que relaciona cada especialidad; el nombre de la

especialidad, el número de profesionales que atienden en dicha especialidad, el número de pacientes atendidos y el total de citas que se registraron en la especialidad determinada.

- **Tabla de datos de los Profesionales:** Contiene la información detallada de cada profesional médico: un identificador único, la especialidad a la cual pertenece, el número de pacientes atendidos y total de citas que fueron atendidas por el profesional determinado.
- **Tabla de datos de los Pacientes:** Contiene información importante de cada paciente: un identificador único, la especialidad en la cuál requiere atención de su tratamiento médico, el especialista que está a cargo de su atención, el número de citas y, la fecha y hora correspondientes de las citas que fueron asignadas al paciente determinado.

5.2.6. API Google Chart

API Google Chart[34] es una tecnología que permite crear distintos tipos de gráficas que son visualizadas en una página web. Se utiliza para la generación de las gráficas permitiendo una mejor comprensión de las salidas de la aplicación ACiT. Las gráficas implementadas con esta tecnología fueron las siguientes:

- **Pacientes atendidos por Especialidad:** Presenta en forma de gráfico circular (más conocido como Torta o Pastel, en inglés Pie) el número y el porcentaje de pacientes que fueron atendidos en cada especialidad.
- **Pacientes atendidos por Profesional:** Presenta mediante un gráfico circular el número y el porcentaje de pacientes que fueron atendidos por cada profesional médico.
- **Demanda diaria de Pacientes por Especialidad:** Presenta mediante un gráfico lineal el número total de pacientes atendidos en los distintos días de la semana.
- **Porcentaje de Ocupación de los Profesionales:** Presenta mediante un gráfico de barras, el porcentaje del tiempo en que cada profesional de una especialidad seleccionada, está libre versus el porcentaje en que está ocupado. El tiempo libre de un profesional se define como el tiempo en que está disponible, pero que no se encuentra atendiendo a ningún paciente, mientras que el tiempo en que está ocupado se define como el tiempo en que está atendiendo a uno o más pacientes en la misma sesión.

5.2.7. SyntaxHighlighter

SyntaxHighlighter[35] es una librería JavaScript desarrollada para dar formato al código fuente de lenguajes de programación, permitiendo de esta manera resaltar su sintaxis dentro de la página web. Se implementó en ACiT para visualizar el contenido del archivo de salida.

5.2.8. Fullcalendar

Fullcalendar[36] es un complemento de JQuery que permite la fácil creación de calendarios. Esta tecnología no se encuentra en la metodología del grupo de investigación AVISPA pero se decidió utilizarla para proporcionar mayor claridad de los horarios de las citas asignadas. Los calendarios implementados son los siguientes:

- Calendario del Profesional Médico: Mediante la selección de un profesional y una o todas las especialidades del mismo, este calendario permite visualizar el horario de atención de dicho profesional, al igual que la fecha y la hora en que fueron asignadas las citas de cada paciente que requiere atención médica en la especialidad que inicialmente fue especificada en el archivo de entrada.
- Calendario de Pacientes: Mediante la escritura del identificador de un paciente, este calendario permite visualizar la fecha y la hora en que le fueron asignadas las citas al paciente en las diferentes especialidades que requiere atención.

Capítulo 6

Experimentación y Pruebas

Una vez concluida la implementación del modelo, se hace necesario analizar su comportamiento y verificar que sus resultados sean correctos. Por lo tanto, en este capítulo se presenta el proceso de pruebas de la aplicación desarrollada en este trabajo de grado, con el fin de evaluar las distintas heurísticas implementadas conforme al número de pacientes y la distribución de la disponibilidad de los mismos. Una vez realizadas las pruebas, se presenta un análisis que permite diferenciar cada una de las características de las soluciones encontradas.

6.1. Descripción de las Pruebas

Los distintos casos de prueba se han dividido en dos tipos: los de menor y los de mayor escala. En los primeros, lo que se busca es analizar exhaustivamente el comportamiento de las heurísticas implementadas en el modelo. Mientras que, en los del segundo tipo, principalmente se pretende evaluar las métricas en las distintas distribuciones de la disponibilidad de los pacientes.

6.1.1. Entradas de Prueba

6.1.1.1. Pruebas de Menor Escala

Puesto que, el propósito principal de este tipo de pruebas es el de evaluar las heurísticas del modelo desarrollado, se decidió generar exclusivamente la disponibilidad de los pacientes con una distribución normal. Esto con el fin, de orientar las pruebas específicamente al comportamiento de dichas heurísticas. La elección de esta distribución, se debe a que como se describe en el Apéndice E, ésta refleja el comportamiento real de la disponibilidad de los pacientes.

A continuación, se describen los casos de prueba:

Código	Descripción
C1	Número Pacientes: 10
C2	Número Pacientes: 20
C3	Número Pacientes: 30
C4	Número Pacientes: 40
C5	Número Pacientes: 50
C6	Número Pacientes: 60
C7	Número Pacientes: 70
C8	Número Pacientes: 80
C9	Número Pacientes: 90
C10	Número Pacientes: 100

Tabla 6.1: Entradas de Prueba: Menor Escala

6.1.1.2. Pruebas de Mayor Escala

Para probar diferentes escenarios en el proceso de asignación de citas terapéuticas, se establecieron las entradas de la aplicación que aparecen en la Tabla 6.2.

Código	Descripción
P1	No. Pacientes: 100, Distribución Pacientes: Ninguna
P2	No. Pacientes: 150, Distribución Pacientes: Ninguna
P3	No. Pacientes: 200, Distribución Pacientes: Ninguna
P4	No. Pacientes: 250, Distribución Pacientes: Ninguna
P5	No. Pacientes: 300, Distribución Pacientes: Ninguna
P6	No. Pacientes: 100, Distribución Pacientes: Uniforme
P7	No. Pacientes: 150, Distribución Pacientes: Uniforme
P8	No. Pacientes: 200, Distribución Pacientes: Uniforme
P9	No. Pacientes: 250, Distribución Pacientes: Uniforme
P10	No. Pacientes: 300, Distribución Pacientes: Uniforme
P11	No. Pacientes: 100, Distribución Pacientes: Normal
P12	No. Pacientes: 150, Distribución Pacientes: Normal
P13	No. Pacientes: 200, Distribución Pacientes: Normal
P14	No. Pacientes: 250, Distribución Pacientes: Normal
P15	No. Pacientes: 300, Distribución Pacientes: Normal
P16	No. Pacientes: 100, Distribución Pacientes: Exponencial
P17	No. Pacientes: 150, Distribución Pacientes: Exponencial
P18	No. Pacientes: 200, Distribución Pacientes: Exponencial
P19	No. Pacientes: 250, Distribución Pacientes: Exponencial
P20	No. Pacientes: 300, Distribución Pacientes: Exponencial
P21	No. Pacientes: 100, Distribución Pacientes: Logarítmica
P22	No. Pacientes: 150, Distribución Pacientes: Logarítmica
P23	No. Pacientes: 200, Distribución Pacientes: Logarítmica
P24	No. Pacientes: 250, Distribución Pacientes: Logarítmica
P25	No. Pacientes: 300, Distribución Pacientes: Logarítmica

Tabla 6.2: Entradas de Prueba: Mayor Escala

6.1.2. Consideraciones para el inicio de las Pruebas

Para iniciar el procedimiento de pruebas se hizo necesario tener en cuenta las siguientes consideraciones para recopilar los datos y realizar el posterior análisis de los mismos:

- **Métricas:** Son los elementos de mayor importancia para realizar las comparativas, puesto que, son los parámetros que definen el desempeño de cada uno de los casos de prueba. Las métricas que se definieron para evaluar el modelo se describen a continuación:
 - **Nsm:** Número de semanas mínimo para encontrar una solución.
 - **Ps:** Porcentaje de satisfacción (%)

- **Te:** Tiempo de ejecución (Sg)
 - **Ns:** Número de soluciones
 - **Us:** Uso de memoria (KB)
 - **Np:** Número de propagaciones
 - **F:** Número de fallas
-
- **Distribuciones de los Pacientes:** Las distribuciones definen el comportamiento de la disponibilidad de todos los pacientes. Debido al análisis de los datos suministrados por la clínica se logró evidenciar que dicha distribución posee un comportamiento de tipo Gaussiano o Normal (Apéndice E). Se decidió evaluar otras distribuciones distintas ya que, como podría suponerse, las distribuciones pueden variar de acuerdo al tiempo o temporada del año. Debido a lo anterior, se presenta a continuación las siguientes distribuciones de la disponibilidad de los pacientes:
 - D0:** Sin distribución o Ninguna
 - D1:** Distribución Uniforme
 - D2:** Distribución Normal
 - D3:** Distribución Exponencial
 - D4:** Distribución Logarítmica
 - **Estrategias de Distribución:** Puesto que en muchos casos, la mayor parte del tiempo de ejecución de un programa ocurre durante la ejecución de la exploración, se optó por considerar distintas estrategias de distribución para encontrar la más eficiente. Esto con el fin de tener varias opciones y así lograr evaluar cuál de ellas permite reducir el espacio de búsqueda de la mejor manera y llega a una solución más rápida.

Por tal razón, se implementaron las siguientes heurísticas cuya diferencia radica en el orden en que las variables y los valores de las mismas son explorados.

- H1:** Asigna aleatoriamente el profesional a un paciente
- H2:** Asigna un paciente al profesional con menor número de pacientes
- H3:** Asigna un paciente al profesional que haya atendido un menor número de citas

- **Restricción de Preferencia:** Constituye un aspecto importante ya que, esta busca optimizar el porcentaje de satisfacción de los pacientes. Lo que quiere decir que, gracias a esta restricción se pretende maximizar el número de pacientes que son atendidos por el profesional de su preferencia. Por esta razón, la restricción de preferencia se tiene en cuenta al momento de crear los casos de prueba para evaluar el comportamiento que tiene este proceso de optimización.

6.1.3. Procedimiento de las Pruebas

El desarrollo de las pruebas de menor escala, consiste en modificar las distintas heurísticas teniendo en cuenta la restricción de preferencia.

El desarrollo de las pruebas de mayor escala, consiste en ensayar los distintos valores o indicadores de la aplicación, modificando cada heurística manteniendo el número de pacientes estáticos por cada caso de prueba, con el fin de evaluar el efecto que tiene el número de pacientes y la distribución de la disponibilidad de los mismos con respecto a la heurística.

Nota: En los distintos casos de prueba, van reseñadas las métricas que se definieron inicialmente en las consideraciones. Adicionalmente, se suman las siguientes columnas:

- **DST (Distribución):** Indica la distribución de los pacientes (ver Apéndice E)
- **E (Entrada):** Indica el código de la prueba

6.2. Resultados

6.2.1. Pruebas de Menor Escala

Las pruebas de menor escala presentaron los siguientes resultados:

6.2.1.1. Resultados sin la Restricción de Preferencia

En la tabla 6.3 se muestran los distintos resultados que se obtuvieron en las distintas heurísticas cuando la función de optimización es deshabilitada. Es decir, cuando el proceso de asignación de las citas se realiza de tal modo, que no se tiene en cuenta la preferencia de un profesional por parte de un paciente determinado.

Heurística H1						
No. Pacientes	Nsm	Ps (%)	Te (Sg)	Ns	Us (KB)	Np
10	1	60,00	0,009	1	198	1.344
20	2	60,00	0,017	1	1.928	4.240
30	2	61,29	0,020	1	2.155	5.596
40	2	60,46	0,030	1	5.006	7.549
50	2	61,82	0,039	1	7.122	9.144
60	2	61,19	0,043	1	7.060	10.937
70	2	46,30	0,048	1	8.852	11.584
80	2	59,46	0,066	1	12.569	13.360
90	2	52,44	0,079	1	15.617	14.365
100	2	54,46	0,090	1	17.565	16.390
Heurística H2						
No. Pacientes	Nsm	Ps (%)	Te (Sg)	Ns	Us (KB)	Np
10	1	50,00	0,013	1	198	1.352
20	2	55,00	0,018	1	1.321	4.167
30	2	61,29	0,032	1	2.155	5.642
40	2	60,46	0,047	1	3.982	7.159
50	2	60,00	0,082	1	5.585	9.007
60	2	58,21	0,120	1	7.060	11.059
70	2	50,00	0,178	1	8.277	11.573
80	2	58,11	0,209	1	11.096	13.296

Continúa en la página siguiente.

90	2	48,78	0,410	1	14.364	15.490
100	2	57,42	0,447	1	15.581	16.283
Heurística H3						
No. Pacientes	Nsm	Ps (%)	Te (Sg)	Ns	Us (KB)	Np
10	1	50,00	0,006	1	198	1.352
20	2	55,00	0,019	1	1.321	4.151
30	2	61,29	0,028	1	2.155	5.503
40	2	67,44	0,051	1	3.928	7.184
50	2	54,55	0,082	1	5.585	8.945
60	2	58,21	0,118	1	7.060	10.944
70	2	44,44	0,167	1	8.277	11.510
80	2	55,40	0,211	1	11.096	13.146
90	2	48,78	0,406	1	14.364	15.523
100	2	56,44	0,447	1	15.517	16.251

Tabla 6.3: Pacientes:10-100. Heurísticas:1-3. Sin Preferencia

6.2.1.2. Resultados con la Restricción de Preferencia

En la tabla 6.4 se muestran los distintos resultados que se obtuvieron en las distintas heurísticas cuando la función de optimización se encuentra habilitada. Es decir, cuando se busca la mayor satisfacción de los pacientes al complacerle con la asignacion de sus citas con su profesional médico de preferencia.

Heurística H1						
No. Pacientes	Nsm	Ps (%)	Te (Sg)	Ns	Us (KB)	Np
10	1	90,00	0,046	5	185	7.914
20	2	100,00	0,041	9	1.541	22.398
30	2	100,00	0,067	13	3.514	42.225
40	2	100,00	0,107	17	7.177	70.433
50	2	100,00	0,171	21	10.970	109.181
60	2	100,00	0,265	28	18.761	183.140
70	2	100,00	0,384	32	25.964	202.544

Continúa en la página siguiente.

80	2	100,00	0,514	34	35.273	254.428
90	2	100,00	0,970	45	60.522	400.959
100	2	100,00	1,029	45	65.208	408.984
Heurística H2						
No. Pacientes	Nsm	Ps (%)	Te (Sg)	Ns	Us (KB)	Np
10	1	90,00	0,069	5	185	10.201
20	2	100,00	0,051	10	1.717	24.931
30	2	100,00	0,086	13	3.482	43.340
40	2	100,00	0,185	18	7.514	73.432
50	2	100,00	0,291	23	11.933	122.006
60	2	100,00	0,395	29	19.474	189.678
70	2	100,00	0,699	28	22.822	181.268
80	2	100,00	0,982	32	33.237	240.429
90	2	100,00	2,069	43	57.589	380.277
100	2	100,00	1,940	44	63.573	400.301
Heurística H3						
No. Pacientes	Nsm	Ps (%)	Te (Sg)	Ns	Us (KB)	Np
10	1	90,00	0,071	5	209	10.506
20	2	100,00	0,051	10	1.717	24.926
30	2	100,00	0,088	13	3.498	42.895
40	2	100,00	0,164	15	6.287	62.019
50	2	100,00	0,322	26	13.440	136.462
60	2	100,00	0,387	29	19.490	189.491
70	2	100,00	0,758	31	25.187	199.480
80	2	100,00	1,016	34	35.233	253.511
90	2	100,00	2,098	43	57.629	381.917
100	2	100,00	1,996	45	64.935	408.318

Tabla 6.4: Pacientes:10-100. Heurísticas:1-3. Con Preferencia

6.2.2. Pruebas de Mayor Escala

Para facilitar la comprensión de los resultados en este tipo de pruebas, se optó por agrupar la información obtenida por heurísticas, que a su vez se subdividen por número de pacientes. De esta forma, se podrá comparar el funcionamiento del modelo con respecto a las distintas distribuciones que pueden tener las disponibilidades de los pacientes.

A continuación, se presentan los resultados de las pruebas de mayor:

6.2.2.1. Resultados sobre la Heurística 1

Resultados de la selección de la heurística 1 como estrategia de distribución con la función de optimización deshabilitada.

100 Pacientes								
DST	E	Nsm	Ps (%)	Te (Sg)	Ns	Us (KB)	Np	F
D0	P1	2	54,55	0,119	1	24.416	20.514	13
D1	P6	2	49,44	0,099	1	14.685	17.470	4
D2	P11	2	54,08	0,111	1	20.703	16.340	14
D3	P16	4	54,26	0,111	1	20.062	17.267	14
D4	P21	2	51,58	0,103	1	16.990	17.906	4
150 Pacientes								
DST	E	Nsm	Ps (%)	Te (Sg)	Ns	Us (KB)	Np	F
D0	P2	2	51,02	0,208	1	41.261	26.428	5
D1	P7	2	53,55	0,224	1	39.406	30.055	12
D2	P12	2	56,52	0,175	1	31.211	23.077	0
D3	P17	10	-	0,630	0	25.803	8.264.197	24.726
D4	P22	2	57,75	0,335	1	37.356	27.482	7
200 Pacientes								
DST	E	Nsm	Ps (%)	Te (Sg)	Ns	Us (KB)	Np	F
D0	P3	2	59,15	0,354	1	68.350	38.143	0
D1	P8	2	54,73	0,343	1	56.573	37.489	0
D2	P13	2	54,59	0,369	1	76.734	38.904	12
D3	P18	10	-	-	-	-	-	-

Continúa en la página siguiente.

D4	P23	2	56,28	0,326	1	65.852	33.176	16
250 Pacientes								
DST	E	Nsm	Ps (%)	Te (Sg)	Ns	Us (KB)	Np	F
D0	P4	2	58,06	0,526	1	107.592	43.228	18
D1	P9	2	53,06	0,549	1	91.912	46.145	11
D2	P14	2	58,66	0,577	1	113.738	47.015	11
D3	P19	4	50,23	0,597	1	123.464	45.597	23
D4	P24	2	57,38	0,561	1	102.408	43.308	24
300 Pacientes								
DST	E	Nsm	Ps (%)	Te (Sg)	Ns	Us (KB)	Np	F
D0	P5	2	54,76	0,833	1	148.697	54.094	17
D1	P10	2	54,20	0,826	1	133.403	58.067	7
D2	P15	2	54,78	0,907	1	173.342	54.114	33
D3	P20	4	57,05	0,951	1	183.452	60.910	17
D4	P25	2	52,28	0,786	1	139.290	53.579	11

Tabla 6.5: Heurística:1. Sin Preferencia. Paciente:100-300.Distribuciones:D0-D4

Resultados de la selección de la heurística 1 como estrategia de distribución con la función de optimización habilitada.

100 Pacientes								
DST	E	Nsm	Ps (%)	Te (Sg)	Ns	Us (KB)	Np	F
D0	P1	2	100,00	1,228	41	625.561	482.797	2.076
D1	P6	2	100,00	0,992	43	454.516	421.552	1.971
D2	P11	2	100,00	1,195	47	576.645	408.813	2.161
D3	P16	4	100,00	1,110	42	498.861	411.040	1.831
D4	P21	2	100,00	1,095	41	519.468	403.136	1.969
150 Pacientes								
DST	E	Nsm	Ps (%)	Te (Sg)	Ns	Us (KB)	Np	F
D0	P2	2	100,00	4,591	70	2.016.729	1.011.220	4.960
D1	P7	2	100,00	5,276	75	1.948.881	1.226.514	5.391

Continúa en la página siguiente.

D2	P12	2	100,00	3,605	65	1.520.544	843.796	4.170
D3	P17	-	-	-	-	-	-	-
D4	P22	2	100,00	4,134	60	1.592.629	928.118	4.249
200 Pacientes								
DST	E	Nsm	Ps (%)	Te (Sg)	Ns	Us (KB)	Np	F
D0	P3	2	100,00	13,864	89	4.194.293	1.930.202	8.504
D1	P8	2	100,00	13,542	95	4.098.709	1.913.754	8.831
D2	P13	2	100,00	14,939	94	4.193.889	2.108.026	8.981
D3	P18	10	-	-	-	-	-	-
D4	P23	2	100,00	9,674	83	3.645.035	1.475.484	7.547
250 Pacientes								
DST	E	Nsm	Ps (%)	Te (Sg)	Ns	Us (KB)	Np	F
D0	P4	2	100,00	24,046	112	4.193.335	2.632.035	12.652
D1	P9	2	100,00	24,43	108	4.193.773	2.577.378	12.753
D2	P14	2	100,00	26,267	110	4.194.175	2.916.820	13.294
D3	P19	4	100,00	25,614	106	4.194.008	2.730.236	12.299
D4	P24	2	100,00	23,508	111	4.194.149	2.623.514	12.632
300 Pacientes								
DST	E	Nsm	Ps (%)	Te (Sg)	Ns	Us (KB)	Np	F
D0	P5	2	100,00	52,120	139	4.194.104	4.279.779	18.794
D1	P10	2	100,00	52,548	132	4.194.179	4.138.094	18.949
D2	P15	2	100,00	52,747	135	4.194.032	4.082.752	18.799
D3	P20	4	100,00	54,771	130	4.194.170	4.659.511	19.108
D4	P25	2	100,00	47,121	137	4.193.856	4.049.434	19.149

Tabla 6.6: Heurística:1. Con Preferencia. Paciente:100-300.Distribuciones:D0-D4

6.2.2.2. Resultados sobre la Heurística 2

Resultados de la selección de la heurística 2 como estrategia de distribución con la función de optimización deshabilitada.

100 Pacientes								
DST	E	Nsm	Ps (%)	Te (Sg)	Ns	Us (KB)	Np	F
D0	P1	2	55,68	0,566	1	19.744	20.123	0
D1	P6	2	53,93	0,453	1	13.918	17.364	0
D2	P11	2	58,16	0,476	1	16.350	16.280	0
D3	P16	4	52,13	0,439	1	15.965	16.824	0
D4	P21	2	55,90	0,469	1	16.351	17.765	0
150 Pacientes								
DST	E	Nsm	Ps (%)	Te (Sg)	Ns	Us (KB)	Np	F
D0	P2	2	53.06	1,624	1	38.637	26.151	0
D1	P7	2	54.84	1,699	1	34.541	29.630	0
D2	P12	2	54.35	1,280	1	31.210	23.144	0
D3	P17	10	-	-	-	-	-	-
D4	P22	2	59.15	1,314	1	34.540	27.106	0
200 Pacientes								
DST	E	Nsm	Ps (%)	Te (Sg)	Ns	Us (KB)	Np	F
D0	P3	2	57.28	3,504	1	68.349	37.957	0
D1	P8	2	53.73	3,373	1	56.508	37.184	0
D2	P13	2	53.62	3,464	1	69.437	38.643	0
D3	P18	10	-	-	-	-	-	-
D4	P23	2	56.28	2,637	1	57.018	32.774	0
250 Pacientes								
DST	E	Nsm	Ps (%)	Te (Sg)	Ns	Us (KB)	Np	F
D0	P4	2	57.66	5,528	1	92.613	42.436	0
D1	P9	2	57.14	6,407	1	88.583	45.630	0
D2	P14	2	55.90	6,450	1	106.633	46.332	0
D3	P19	4	47.89	6,284	1	105.091	44.496	1
D4	P24	2	56.12	5,830	1	87.045	42.321	0
300 Pacientes								
DST	E	Nsm	Ps (%)	Te (Sg)	Ns	Us (KB)	Np	F
D0	P5	2	54,08	10,980	1	134.424	53.503	0
D1	P10	2	53,84	10,720	1	127.964	58.096	0

Continúa en la página siguiente.

D2	P15	2	54,78	11,370	1	138.458	53.508	0
D3	P20	4	57,38	10,870	1	166.168	60.239	2
D4	P25	2	54,04	9,840	1	132.058	53.354	0

Tabla 6.7: Heurística:2. Sin Preferencia. Paciente:100-300.Distribuciones:D0-D4

Resultados de la selección de la heurística 2 como estrategia de distribución con la función de optimización habilitada.

100 Pacientes								
DST	E	Nsm	Ps (%)	Te (Sg)	Ns	Us (KB)	Np	F
D0	P1	2	100,00	4,057	40	609.217	473.005	1.979
D1	P6	2	100,00	2,923	42	442.141	404.920	1.901
D2	P11	2	100,00	1,854	42	512.869	369.968	1.919
D3	P16	4	100,00	1,780	46	546.100	461.306	2.013
D4	P21	2	100,00	3,108	43	538.445	419.456	2.016
150 Pacientes								
DST	E	Nsm	Ps (%)	Te (Sg)	Ns	Us (KB)	Np	F
D0	P2	2	100,00	6,950	70	2.016.153	1.005.274	4.931
D1	P7	2	100,00	8,940	71	1.842.648	1.158.124	5.073
D2	P12	2	100,00	9,437	64	1.498.948	818.624	4.114
D3	P17	10	-	-	-	-	-	-
D4	P22	2	100,00	12,113	59	1.568.791	905.057	4.156
200 Pacientes								
DST	E	Nsm	Ps (%)	Te (Sg)	Ns	Us (KB)	Np	F
D0	P3	2	100,00	38,659	92	4.194.165	1.999.810	8.779
D1	P8	2	100,00	26,195	94	4.058.476	1.879.744	8.739
D2	P13	2	100,00	19,517	97	4.194.146	2.161.563	9.259
D3	P18	10	-	-	-	-	-	-
D4	P23	2	100,00	24,381	88	3.857.446	1.565.160	7.956
250 Pacientes								
DST	E	Nsm	Ps (%)	Te (Sg)	Ns	Us (KB)	Np	F

Continúa en la página siguiente.

D0	P4	2	100,00	64,187	106	4.193.976	2.507.346	11.881
D1	P9	2	100,00	54,800	106	4.193.838	2.537.808	12.458
D2	P14	2	100,00	48,600	113	4.194.247	2.952.920	13.598
D3	P19	4	100,00	112,320	112	4.194.154	2.851.081	12.637
D4	P24	2	100,00	61,930	105	4.194.163	2.483.620	11.813
300 Pacientes								
DST	E	Nsm	Ps (%)	Te (Sg)	Ns	Us (KB)	Np	F
D0	P5	2	100,00	118,170	136	4.194.155	4.194.559	18.247
D1	P10	2	100,00	138,840	133	4.193.596	4.172.168	19.010
D2	P15	2	100,00	109,080	138	4.194.197	4.142.321	19.142
D3	P20	4	100,00	164,000	131	4.194.193	4.691.649	19.123
D4	P25	2	100,00	117,000	132	4.194.268	3.901.528	18.384

Tabla 6.8: Heurística:2. Con Preferencia. Paciente:100-300.Distribuciones:D0-D4

6.2.2.3. Resultados sobre la Heurística 3

Resultados de la selección de la heurística 3 como estrategia de distribución con la función de optimización deshabilitada.

100 Pacientes								
DST	E	Nsm	Ps (%)	Te (Sg)	Ns	Us (KB)	Np	F
D0	P1	2	53,40	0,522	1	19.744	20.258	0
D1	P6	2	56,18	0,455	1	13.918	17.373	0
D2	P11	2	54,08	0,479	1	16.350	16.193	0
D3	P16	4	53,19	0,436	1	16.347	16.882	2
D4	P21	2	58,95	0,476	1	16.351	17.707	0
150 Pacientes								
DST	E	Nsm	Ps (%)	Te (Sg)	Ns	Us (KB)	Np	F
D0	P2	2	51,70	1,640	1	38.637	26.142	0
D1	P7	2	53,55	1,670	1	34.541	29.566	0
D2	P12	2	52,17	1,270	1	31.210	23.013	0

Continúa en la página siguiente.

D3	P17	10						
D4	P22	2	55,64	1,290	1	34.540	27.063	0
200 Pacientes								
DST	E	Nsm	Ps (%)	Te (Sg)	Ns	Us (KB)	Np	F
D0	P3	2	57,28	3,540	1	68.285	38.084	0
D1	P8	2	54,72	3,400	1	56.508	37.168	0
D2	P13	2	53,62	3,480	1	69.437	38.504	0
D3	P18	10	-	-	-	-	-	-
D4	P23	2	55,78	2,650	1	56.826	32.877	0
250 Pacientes								
DST	E	Nsm	Ps (%)	Te (Sg)	Ns	Us (KB)	Np	F
D0	P4	2	54,03	5,490	1	92.613	42.506	0
D1	P9	2	55,10	6,360	1	88.647	45.496	0
D2	P14	2	58,66	6,420	1	106.633	46.440	0
D3	P19	4	51,17	6,340	1	104.709	44.338	0
D4	P24	2	56,12	5,790	1	86.981	42.320	0
300 Pacientes								
DST	E	Nsm	Ps (%)	Te (Sg)	Ns	Us (KB)	Np	F
D0	P5	2	53,40	11,02	1	134.424	53.328	0
D1	P10	2	56,99	10,550	1	127.964	57.959	0
D2	P15	2	55,44	11,220	1	138.394	53.432	0
D3	P20	4	58,03	10,890	1	166.168	60.383	2
D4	P25	2	51,58	9,810	1	132.058	53.215	0

Tabla 6.9: Heurística:3. Sin Preferencia. Paciente:100-300.Distribuciones:D0-D4

Resultados de la selección de la heurística 3 como estrategia de distribución con la función de optimización habilitada.

100 Pacientes								
DST	E	Nsm	Ps (%)	Te (Sg)	Ns	Us (KB)	Np	F
D0	P1	2	100,00	4,310	42	638.891	498.367	2.079

Continúa en la página siguiente.

D1	P6	2	100,00	2,830	40	421.303	286.957	1.808
D2	P11	2	100,00	1,980	46	562.072	403.068	2.099
D3	P16	4	100,00	1,760	45	542.950	457.681	2.053
D4	P21	2	100,00	2,890	40	501.523	389.085	1.877
150 Pacientes								
DST	E	Nsm	Ps (%)	Te (Sg)	Ns	Us (KB)	Np	F
D0	P2	2	100,00	7,200	72	2.075.733	1.033.332	5.067
D1	P7	2	100,00	8,670	73	1.894.100	1.187.668	5.217
D2	P12	2	100,00	9,820	67	1.568.725	853.213	4.303
D3	P17	10	-	-	-	-	-	-
D4	P22	2	100,00	13,300	64	1.699.691	981.359	4.507
200 Pacientes								
DST	E	Nsm	Ps (%)	Te (Sg)	Ns	Us (KB)	Np	F
D0	P3	2	100,00	38,040	92	4.194.168	2.002.959	8.777
D1	P8	2	100,00	26,060	92	3.972.829	1.839.145	8.553
D2	P13	2	100,00	18,440	97	4.194.210	2.151.298	9.256
D3	P18	10	-	-	-	-	-	-
D4	P23	2	100,00	24,210	89	3.900.684	1.586.754	8.045
250 Pacientes								
DST	E	Nsm	Ps (%)	Te (Sg)	Ns	Us (KB)	Np	F
D0	P4	2	100,00	69,370	115	4.193.528	2.721.529	12.888
D1	P9	2	100,00	57,660	111	4.193.902	2.647.970	13.045
D2	P14	2	100,00	46,650	106	4.194.220	2.778.079	12.762
D3	P19	4	100,00	106,500	105	4.194.143	2.668.810	11.829
D4	P24	2	100,00	62,500	105	4.193.650	2.483.932	11.814
300 Pacientes								
DST	E	Nsm	Ps (%)	Te (Sg)	Ns	Us (KB)	Np	F
D0	P5	2	100,00	119,970	138	4.194.287	4.253.822	18.513
D1	P10	2	100,00	126,650	124	4.193.340	3.883.698	17.730
D2	P15	2	100,00	107,360	136	4.194.164	4.071.817	18.870
D3	P20	4	100,00	160,260	129	4.194.197	4.628.763	18.829
D4	P25	2	100,00	120,010	139	4.194.286	4.091.065	19.356

Continúa en la página siguiente.

Tabla 6.10: Heurística:3. Con Preferencia. Paciente:100-
300.Distribuciones:D0-D4

6.3. Análisis de Resultados

6.3.1. Pruebas de Menor Escala

De las pruebas de menor escala se puede inferir lo siguiente:

1. El tiempo de ejecución y el uso de memoria aumentan conforme aumenta el número de pacientes. Cuando la función de optimización se encuentra habilitada el tiempo y el uso de memoria son mucho mayor que cuando no lo está.
2. Sin la función de optimización el porcentaje de satisfacción se comporta de manera variable siendo éste superior en la mayoría de los casos al 50 %.
3. Para el caso en el que la función de optimización está habilitada, el porcentaje de satisfacción cumple en gran parte con las expectativas de asignar las citas a los pacientes con su profesional de preferencia, siendo éste igual al 100 % en la mayoría de los casos.
4. El porcentaje de satisfacción no parece tener relación directa con el número de pacientes.
5. El tiempo mínimo para la asignación de las citas son 2 semanas en la mayoría de los casos, exceptuando el caso que el número de pacientes sea 10 cuyo número de semanas es 1.
6. Entre mayor sea el número de pacientes en la heurística H1, mayor es el uso de memoria en comparación con las distribuciones H2 y H3. Es decir, que en la heurística H1 se consume mayor memoria que en las otras dos.
7. Sin la función de optimización, el porcentaje de ocupación promedio es mayor en H1, pero este no es equilibrado, lo que quiere decir que existen profesionales con mayor carga laboral que otros.
8. Cuando la función de optimización se encuentra deshabilitada, existe una diferencia notable en la carga equilibrada de pacientes tanto en la heurística H2 como H3, con respecto a H1. De igual forma, la carga equilibrada de pacientes tanto en H2 como H3, se adapta perfectamente con un mayor número de estos.
9. Cuando la función de optimización se encuentra deshabilitada, existe una diferencia notable en la carga equilibrada de citas tanto en la heurística H3 como H2, con respecto a H1. De igual forma, la carga equilibrada de citas tanto en H3 como H2, se adapta perfectamente con un mayor número de estos.

10. Cuando se habilita la función de optimización, el porcentaje de ocupación promedio de las 3 heurísticas son medianamente distintos y no parece tener relación directa con el número de pacientes.
11. Cuando se habilita la función de optimización, no se logra evidenciar que exista una carga equilibrada de pacientes ni de citas en las 3 heurísticas, puesto que, la dispersión de los datos se encuentra muy alta.

6.3.2. Pruebas de Mayor Escala

De las pruebas de mayor escala se puede inferir lo siguiente:

1. Tanto el consumo de memoria como el tiempo de ejecución se ven visiblemente afectados por la función de optimización, que busca que todos los pacientes sean atendidos por el profesional de su preferencia.
2. A pesar de que la función de optimización afecta el rendimiento del modelo y la distribución equilibrada de pacientes y citas entre los profesionales, se ve un cambio positivo en cuanto al índice de satisfacción de pacientes. Debido a que se logra el 100 % de satisfacción, en lugar de un 50 % en promedio, cuando no se activa la función de optimización.
3. El modelo no tiene un buen comportamiento cuando la disponibilidad de los pacientes sigue una distribución exponencial. Pues, en comparación con los otros tipos de distribuciones, se necesita de más semanas para encontrar una solución y en algunos casos, el problema es tan complejo, que el modelo no encuentra solución.
4. Se puede evidenciar que las distribuciones que tienen las disponibilidades de los pacientes no es un factor influyente en el rendimiento o consumo de memoria del modelo. Por su parte, el tamaño de la entrada (o número de pacientes) si constituye un factor determinante para el rendimiento y el consumo de memoria del modelo.
5. La distribución normal es la segunda distribución (después de la distribución uniforme) en la que el porcentaje de ocupación de los distintos profesionales médicos es equitativo. No se evidencian grandes diferencias relacionadas con las heurísticas y el número de pacientes.
6. El modelo presenta un mejor comportamiento en términos de manejo de memoria, tiempo de ejecución y porcentajes de ocupación de los profesionales cuando la disponibilidad de los pacientes sigue una distribución uniforme.

7. Las heurísticas que buscan una distribución equilibrada de pacientes o citas entre los profesionales, logran que el modelo tenga un mejor consumo de memoria, en comparación a la heurística proporcionada por Gecode. Sin embargo, el tiempo de ejecución del modelo se ve afectado en gran medida.
8. El número de fallas presentadas, antes de encontrar una solución (óptima o no) es menor cuando se utilizan las heurísticas que logran una distribución equilibrada de pacientes o citas entre los profesionales.

Capítulo 7

Conclusiones y Trabajos Futuros

7.1. Conclusiones

1. El modelo para la asignación de citas terapéuticas, fue diseñado con base al problema de Job-Shop Scheduling [39] y en las condiciones especiales presentadas en la Clínica de Rehabilitación del Valle, se logró abstraer el problema de tal manera que se pudiera expresar en un modelo matemático, que facilitó la representación de las restricciones más importantes que se deben tomar en cuenta en este proceso.
2. El uso de un modelo matemático para la representación del problema, permite realizar un proceso de implementación transparente ya que no se requirieron funciones propias de un lenguaje de programación, lo que facilita la representación del modelo en cualquier motor de satisfacción por restricciones. En este caso, se decidió utilizar GECODE, debido a las ventajas que ofrece aspectos como: La cantidad y calidad de propagadores implementados, la optimización de recursos computacionales que ofrece un lenguaje de programación como C++, así como el nivel de abstracción que ofrece la paradigma de programación orientado a objetos para representar los actores propios del problema de asignación de citas terapéuticas.
3. La metodología del grupo de investigación AVISPA relacionada con las aplicaciones por restricciones, fue de gran ayuda para que la implementación del modelo matemático realizada en GECODE y el lenguaje de programación C++, fuera fácilmente acoplada a un entorno Web y finalmente accesible desde el mismo.
4. Las heurísticas relacionadas con la carga equilibrada del número de pacientes y de citas, superan las expectativas cuando la función de optimización no es tomada en cuenta. No obstante, cuando es el caso contrario, éstas se ven considerablemente afectadas; esto se debe

esencialmente a que el factor determinante de dichas heurísticas son los pacientes y las citas, mientras que la optimización busca maximizar a como dé lugar la satisfacción del paciente por la escogencia de su profesional médico de preferencia.

5. El modelo demostró ser eficiente cuando la disponibilidad de los pacientes permite que se puedan asignar las citas en diferentes espacios de tiempo un mismo día y a lo largo de la semana. Sin embargo, el modelo no es eficiente cuando las disponibilidades de los pacientes se encuentran concentradas en un mismo espacio de tiempo a lo largo de la semana. Es decir, cuando los pacientes poseen una disponibilidad que sigue una distribución exponencial, el modelo no presenta un comportamiento aceptable, en términos de recursos computacionales consumidos y número de semanas necesitadas para encontrar una solución.
6. Se pudo observar que, mientras en el modelo sin preferencia se enfoca en encontrar una solución lo más rápido posible, el modelo con preferencia se enfoca en encontrar una solución que satisfaga al máximo posible las peticiones de los clientes, esto conlleva a que el motor de búsqueda tenga que buscar varias soluciones hasta encontrar la más óptima, lo que lleva un tiempo considerable, además, en este proceso, GECODE debe crear copias del espacio de cómputo cada cierto número de nodos, lo que produce un mayor gasto de memoria.
7. La solución presentada en este proyecto queda sujeta a algunas limitaciones computacionales, como lo es el tamaño de la entrada que principalmente viene determinado por el número de pacientes. Si este número es muy grande, el rendimiento se ve afectado considerablemente.
8. Finalmente, se puede concluir que la propuesta presentada en este proyecto, es un buen acercamiento inicial al problema de asignación de citas terapéuticas. Sin embargo, en la vida real no se adapta a la naturaleza actual de la Clínica, ya que el sistema implementado no asigna las citas inmediatamente conforme la solicitan los pacientes, sino que previamente requiere de la acumulación de la información para realizar el proceso de asignación de todos los pacientes. No obstante, esta propuesta fácilmente se adaptaría a las necesidades de la Clínica si se evaluara la viabilidad de cambiar la forma en que se realiza el proceso actual para la asignación de las citas.

7.2. Trabajos Futuros

1. En vista de que, se hace dispendioso crear los archivos de entrada de la aplicación Web de forma manual, una buena medida sería la de acoplar el generador de archivos de prueba al entorno Web. Esto evitaría posibles complicaciones por parte del usuario final, ya que permitiría que la aplicación pueda ser probada sin necesidad de que éste tenga que crear un archivo de prueba propio.
2. Para reducir el gasto computacional del proceso de lectura de archivos, se propone modificar la forma en que se ingresan los datos de entrada de la aplicación, de manera que sea mucho más eficiente y entendible para el usuario común; una alternativa puede ser mediante el uso de formularios u otro tipo de mecanismos a través de la aplicación web.
3. Se debe considerar la idea de implementar el modelo matemático, utilizando el concepto de Dynamic Scheduling [38] con el fin de dar inmediatez en el proceso de asignación de citas, es decir, conforme se generen las solicitudes de atención de un tratamiento por parte de los pacientes, se realice la asignación de la cita inmediatamente.
4. Convertir la aplicación web, en un sistema de información, de tal manera que se puedan proveer funcionalidades como: manejo de la información personal de los pacientes, secretarias y especialistas, manejo de historias clínicas, manejo de diagnósticos médicos y/o formulación de recetas médicas y gestión de facturación de las citas.

Referencias

- [1] Apt K. Principles of constraint programming. USA: Cambridge University Press; 2003.
- [2] Bai Y. Practical Database Programming With Visual C#.NET. Wiley-IEEE Press; 2010.
- [3] Cormen T. Introduction to algorithms. 2da Edición. USA: MIT Press; 2001.
- [4] Knuth D. The Art of Computer Programming: Fundamental Algorithms. 3ra Edición. USA: Addison-Wesley; 1968.
- [5] C. Schulte, G. Tack, and M. Z. Lagerkvist, “Modeling,” in Modeling and Programming with Gecode, C. Schulte, G. Tack, and M. Z. Lagerkvist, Eds. 2012.
- [6] P. VanHentenryck, Consistency techniques in logic programming. 1987.
- [7] Walsh, Beek y Rossi F. Handbook of Constraint Programming. Italia: Elsevier; 2006.
- [8] Aranda et al. The Problem of Assigning Evaluators to the Articles Submitted in an Academic Event: A Practical Solution Incorporating Constraint Programming and Heuristics. Multiparadigm Programming in Mozart/OZ 2005. p 305-316.
- [9] Barber y Salido. Introducción a la Programación de Restricciones. Inteligencia Artificial, Revista Iberoamericana de Inteligencia Artificial 2003; Volumen: 20. p 13-30.
- [10] Hannenbauer y Geske. Coordinating distributed CLP-solvers in medical appointment scheduling. Proceedings of the Twelfth International Conference on Applications of Prolog 1999. p 117–125.
- [11] Newell, A. y Simon, H. A. (1976). Computer science as empirical inquiry: symbols and search. Comm. Of the ACM. 19, p 113-126.
- [12] Rueda et al. Pathos: Object-oriented concurrent constraint timetabling for real world cases. Proceedings of XXVIII Latin American Conference on Informatics 2002.

- [13] Yokoo et al. The distributed constraint satisfaction problem: formalization and algorithms. Knowledge and Data Engineering, IEEE Transactions 1998; Volumen: 10. p 673-685.
- [14] Martínez M., Cánovas D. Técnicas de autooptimización en recorridos de árboles por medio de backtracking [Trabajo de Grado]. Murcia: Universidad de Murcia. Facultad de Informática; 2009. 108p.
- [15] Barták R. Online guide to constraint programming. [Artículo en Internet]. URL: <http://ktiml.mff.cuni.cz/~bartak/constraints/intro.html>. [Consultado: 2011-05-23].
- [16] Comunidad Wikipedia en Español. Heurística (Informática). URL: [http://es.wikipedia.org/wiki/Heur%C3%ADstica_\(inform%C3%A1tica\)](http://es.wikipedia.org/wiki/Heur%C3%ADstica_(inform%C3%A1tica)). [Consultado: 2011-06-16].
- [17] Crocker L. Heuristic. [Artículo en Internet]. URL: <http://en.wikipedia.org/wiki/Heuristic>. [Consultado: 2011-05-23].
- [18] «GECODE - An open, free, efficient constraint solving toolkit». [Online]. Disponible en: <http://www.gecode.org/>.
- [19] «Drupal - Open Source CMS — drupal.org». [Online]. Disponible en: <http://drupal.org/>.
- [20] «Dries Buytaert — Personal website of Dries Buytaert». [Online]. Disponible en: <http://buytaert.net/>.
- [21] «PHP: Hypertext Preprocessor». [Online]. Disponible en: <http://php.net/>.
- [22] «MySQL: The world's most popular open source database». [Online]. Disponible en: <http://www.mysql.com/>.
- [23] «Welcome to The Apache Software Foundation!» [Online]. Disponible en: <http://www.apache.org/>.
- [24] «Licencias - Proyecto GNU - Free Software Foundation (FSF)». [Online]. Disponible en: <http://www.gnu.org/licenses/>.
- [25] «Definicion de CGI». [Online]. Disponible en: <http://www.pergaminovirtual.com.ar/definicion/CGI.html>.

- [26] «GSL - GNU Scientific Library - GNU Project - Free Software Foundation (FSF)». [Online]. Disponible en: <http://www.gnu.org/software/gsl/>.
- [27] «A brief description - C++ Information». [Online]. Disponible en: <http://www.cplusplus.com/info/description/>.
- [28] «Portal CCP Avispa — Conozca los proyectos en programación por restricciones del Grupo». [Online]. Disponible en: <http://avispa.univalle.edu.co/site/>.
- [29] J. F. Díaz, C. Martínez, C. A. Delgado, M. A. Cruz, L. F. Vargas, “Metodología para crear aplicaciones de programación por restricciones como servicio web. Definición e implementación con diversos casos”. Colombia: Universidad del Valle, 2012. [Online]. Disponible en <http://avispa.univalle.edu.co/~cardel/proyInv/informeFinal.pdf>.
- [30] C. A. Delgado, “Manual portal Web de aplicaciones de programación por restricciones”. Colombia: Universidad del Valle, 2012. [Online]. Disponible en: <http://avispa.univalle.edu.co/~cardel/proyInv/ManualPortal.pdf>.
- [31] «Extensible Markup Language (XML)». [Online]. Disponible en: <http://www.w3.org/XML/>.
- [32] «jQuery». [Online]. Disponible en: <http://jquery.com/>.
- [33] «DataTables (table plug-in for jQuery)». [Online]. Disponible en: <http://datatables.net/>.
- [34] «Google Chart Tools — Google Developers». [Online]. Disponible en: <https://developers.google.com/chart/>.
- [35] «Overview - syntaxhighlighter - Overview of SyntaxHighlighter. - Free syntax highlighter written in Java Script - Google Project Hosting». [Online]. Disponible en: <http://code.google.com/p/syntaxhighlighter/wiki/Overview>.
- [36] «FullCalendar - Full-sized Calendar jQuery Plugin». [Online]. Disponible en: <http://arshaw.com/fullcalendar/>.
- [37] «TinyXml Main Page». [Online]. Disponible en: <http://www.grinninglizard.com/tinyxml/>.
- [38] ELKHYARI, Abdallah, et al. Constraint programming for dynamic scheduling problems. Hiroshi Kise, editor, 2004, no 04, p. 84-89.

-
- [39] Yamada, T., Nakano, R. (1997). Job shop scheduling. Genetic Algorithms in Engineering Systems, 134.

APÉNDICES

Apéndice A

Descripción del Formato del Archivo de Entrada

Para brindar mayor claridad acerca del formato del archivo de entrada, se realizará la explicación del mismo a partir del siguiente archivo de ejemplo:

A.1. Archivo de ejemplo

```
1 3
2 4
3 5
4 1 'Rehab Fisica' 5 30
5 2 'T Lenguaje' 6 20
6 3 'Espirometrias' 5 15
7 4 'Psicologia' 5 15
8 1 'Profesional01' 1
9 1 - 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
   1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1
10 2 'Profesional02' 2
11 2 - 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 0 0 0 0 0 0 1 1 1 1 1 0 0 0 0 0 0 1
   1 1 1 1 0 0 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1
12 3 - 1 1 1 1 1 0 0 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 0 0 1 1 1 1 0
   0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 0 0 1 1 1 1 0 0 0 0 0
13 3 'Profesional03' 1
```

```

14 2 - 1 1 1 1 1 0 0 1 1 1 1 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 1 1 1 1 0
    0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0
15 1 'Paciente01' 1 [2 4 0]
16 1 1 1 1 1 0 0 1 1 1 1 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 1 1 1 1 0 0 0
    0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0
17 2 'Paciente02' 2 [1 3 1][3 4 0]
18 1 1 1 1 1 0 0 1 1 1 1 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 1 1 1 1 0 0 0
    0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0
19 3 'Paciente03' 1 [2 6 0]
20 1 1 1 1 1 0 0 1 1 1 1 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 1 1 1 1 0 0 0
    0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0
21 4 'Paciente04' 1 [4 4 0]
22 1 1 1 1 1 0 0 1 1 1 1 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 1 1 1 1 0 0 0
    0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0
23 5 'Paciente05' 2 [1 3 1]
24 1 1 1 1 1 0 0 1 1 1 1 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 1 1 1 1 0 0 0
    0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0

```

A.2. Explicación del archivo de entrada

A continuación, mediante el archivo de entrada de ejemplo, se especificará detalladamente la información que debe considerar el usuario al momento de crear dicho archivo.

El archivo de entrada de ejemplo se encuentra organizado de la siguiente manera:

- En la primera línea se especifica el número total de profesionales
En el archivo de ejemplo: Tres profesionales
Línea 01: 3
- En la segunda línea se especifica el número total de especialidades
En el archivo de ejemplo: Cuatro especialidades
Línea 02: 4
- En la tercer línea se especifica el número total de pacientes
En el archivo de ejemplo: Cinco pacientes
Línea 03: 5

- Las E líneas siguientes contienen la información de las Especialidades y se encuentran organizadas de la siguiente manera. Por cada línea habrán 4 columnas:

- La primera indicará el id de la especialidad
- La segunda (entre comillas simples) indicará el nombre de la especialidad
- La tercera indicará el límite de pacientes a atender por slot de tiempo
- La cuarta indicará la duración en minutos de las citas para esa especialidad

En el archivo de ejemplo:

Especialidad con id: 1, nombre: Rehab. Física, limite: 5 pacientes y duración: 30 minutos

Línea 04: 1 'Rehab Física' 5 30

- Las P líneas siguientes contienen la información de los Profesionales y se encuentran organizadas de la siguiente manera:

Por cada línea existe una serie de columnas:

- La primera indica el identificador del profesional
- La segunda indica, entre comillas simples, el nombre del profesional
- La tercera indica el número de especialidades que está capacitado para atender

En la siguiente línea se indican, primero el código de especialidad en la que el profesional puede atender, seguido de un guión y a continuación los horarios de atención para dicha especialidad (la aclaración de los horarios se realizará en la Explicación de la Disponibilidad).

En el archivo de ejemplo:

Profesional con Identificador: 1, nombre: Profesional01, número de especialidades: 1

Línea 08: 1 'Profesional01' 1

Identificador de la Especialidad del Profesional: 1

Disponible: Lunes: tarde, Martes: mañana, Jueves: mañana, Viernes: tarde, Sábado: mañana

Línea 09: 1 - 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 0
0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1

- Las N líneas siguientes contienen la información de los Pacientes y se encuentran organizadas de manera similar a las anteriores P líneas:

Por cada línea existe una serie de columnas:

Cada campo de la Tabla A.1 se define como un slot, el cual es un espacio de tiempo determinado. Así por ejemplo, el primer slot (el slot 0) corresponde al intervalo de tiempo del día lunes desde las 07:00 a.m hasta las 08:00 a.m del mismo día. Para el caso del slot 25, por ejemplo, se refiere al intervalo de tiempo desde las 08:00 a.m hasta las 09:00 a.m del día miércoles. Lo mismo sucede para el resto de los slots de la Tabla A.1

Horario	Dias de la Semana						
	L	M	M	J	V	S	D
07:00 - 08:00	0	12	24	36	48	60	X
08:00 - 09:00	1	13	25	37	49	61	X
09:00 - 10:00	2	14	26	38	50	62	X
10:00 - 11:00	3	15	27	39	51	63	X
11:00 - 12:00	4	16	28	40	52	64	X
12:00 - 13:00	5	17	29	41	53	X	X
13:00 - 14:00	6	18	30	42	54	X	X
14:00 - 15:00	7	19	31	43	55	X	X
15:00 - 16:00	8	20	32	44	56	X	X
16:00 - 17:00	9	21	33	45	57	X	X
17:00 - 18:00	10	22	34	46	58	X	X
18:00 - 19:00	11	23	35	47	59	X	X

Tabla A.1: Tabla de Disponibilidad

Para el caso de los profesionales, la disponibilidad se gestiona a partir de jornadas: La jornada de la mañana va desde las 07:00 a.m hasta el mediodía, en cambio la jornada de la tarde va desde el mediodía hasta las 07:00 p.m (19:00 hh). De esta manera, si un profesional está disponible el día lunes en la jornada de la mañana deberá contener el valor 1 en los slots 0, 1, 2, 3 y 4.

Para el caso de los pacientes, se sabe que la disponibilidad de los mismos viene dada por factores externos a la clínica, puesto que en gran medida depende de las ocupaciones de los distintos pacientes y por esta razón su disponibilidad puede variar dependiendo de la hora y el día. En vista de esto, como se explica en el Apéndice D (Consideraciones del Generador) se optó por generar la disponibilidad de los pacientes de acuerdo a una distribución.

En conclusión, se puede deducir fácilmente que un profesional o un paciente está disponible en un slot determinado si el valor en dicho slot es uno (1), o en caso de no estarlo si su valor es cero (0).

Apéndice B

Descripción del Formato del Archivo de Salida

El formato del archivo de salida tiene la siguiente estructura básica:

```
1 <?xml version="1.0" encoding="utf-8" ?>
2 <acit>
3   <num_semanas>S</num_semanas>
4   <especialidad id="idE" nombre="nomE" duracion="durE">
5     <profesional id="idPr" nombre="nomPr">
6       <disponible dia="d" hora_ini="HH:mm" hora_fin="HH:mm">
7         </disponible>
8       <paciente id="idPa" nombre="nomPa" num_citas="nCPa">
9         <cita fecha="AAAA-MM-DD" hora_ini="HH:mm" hora_fin="HH:mm">
10          </cita>
11        </paciente>
12      </profesional>
13    </especialidad>
14  </acit>
```

Inicialmente se define la codificación del archivo, seguidamente se encuentran las siguientes etiquetas:

- **Codificación:** `<?xml version = "1,0" encoding = "utf - 8"? >`
- **<acit>:** Es la etiqueta general del archivo xml.

- **<num_semanas>**: Esta etiqueta se encuentra en el interior de la etiqueta general, cuyo valor de S identifica el número de semanas que han sido programadas para asignar las citas.
- **<especialidad>**: Esta etiqueta también se encuentra en el interior de la etiqueta general (<acit>), y se crea una de estas etiquetas por cada una de las especialidades que se han definido en el archivo de entrada. Contiene la información referente a cada especialidad, cuyos parámetros son *id*, *nombre* y *duracion*, en los cuales, respectivamente se definen el identificador, el nombre y la duración de las citas en la especialidad determinada.
- **<profesional>**: Esta etiqueta se encuentra en el interior de la etiqueta <especialidad>. Hace referencia a los profesionales que están en la capacidad de atender en una especialidad dada. Se crea una de estas por cada profesional perteneciente a la especialidad determinada. Sus parámetros son *id* y *nombre*, en los cuales, respectivamente se definen el identificador y el nombre del profesional médico.
- **<disponible>**: Esta etiqueta se encuentra en el interior de la etiqueta <profesional>. Hace referencia a los distintos días y horarios en que un profesional dado está disponible para atender las citas. Se crea una de estas por cada combinación diferente del día y horas de inicio y fin en que está disponible el profesional. Sus parámetros son *dia*, *hora_ini* y *hora_fin*, los cuales definen el día, hora de inicio y hora de fin en que está disponible el profesional dado. El día se especifica con un número entero entre 1 y 7, siendo el número 1 el día lunes, el 2 el día martes, y así sucesivamente hasta el 7 que especifica el día domingo. La hora de inicio y de fin se definen en formato de hora militar.
- **<paciente>**: Esta etiqueta se encuentra en el interior de la etiqueta <profesional>. Hace referencia a los pacientes que son atendidos por un profesional específico. Se crea una de estas por cada paciente que es atendido por el profesional determinado. Sus parámetros son *id*, *nombre*, *num_citas*, en los cuales se definen respectivamente el identificador, el nombre y el número de citas que solicitó el paciente.
- **<cita>**: Esta etiqueta se encuentra en el interior de la etiqueta <paciente>. Hace referencia a la fecha y hora de atención de las citas que fueron asignadas al paciente dado. Se crea una de estas por cada cita que se le asignó al paciente. Sus parámetros son *fecha*, *hora_ini* y *hora_fin*, en los cuales respectivamente se definen la fecha con el formato (AAAA-MM-DD, es decir, el año con 4 dígitos, el mes y el día con 2), la hora de inicio y hora de fin de la cita en formato militar.

Apéndice C

Especificaciones de las Especialidades

La siguiente tabla presenta la información que fue suministrada por el personal de la clínica, la cual muestra las diferentes especialidades en las que la clínica ofrece atención médica, el número de profesionales encargados de realizar la atención en dichas especialidades, al igual que el número máximo de pacientes que pueden ser atendidos en las mismas, y finalmente aparece la duración de la cita en cada una de estas:

No.	Nombre Especialidad	No. Profesionales	Límite Pacientes	Duración Cita
1	Rehabilitación Física	10	5 a 6 pacientes	1 hora
2	Rehabilitación Cardíaca	1	5 a 6 pacientes	1 hora
3	Terapia de Lenguaje	1	4 pacientes	30 minutos
4	Terapia Ocupacional	1	4 a 5 pacientes	30 minutos
5	Terapia Respiratoria	1	3 a 4 pacientes	20 minutos
6	Nutrición	1	1 paciente	20 minutos
7	Psicología	1	1 paciente	20 minutos
8	Audiología	1	1 paciente	20 minutos
9	Piso Pélvico	1	1 paciente	1 hora
10	Espirometrías	1	1 paciente	15 minutos

Tabla C.1: Especificaciones de Especialidades

Apéndice D

Especificaciones del Generador de Archivos de Prueba

Al momento de implementar el generador de archivos de prueba se tienen en cuenta las siguientes consideraciones:

- **Número de especialidades:** Es definido a partir de la información suministrada por la Clínica.
- **Número de Profesionales:** En vista de que esta información no fue suministrada, el número de profesionales se calcula de forma aleatoria teniendo en cuenta que un profesional puede atender un máximo de tres especialidades.
- **Número de pacientes:** Es definido de forma manual por el usuario.
- **Disponibilidad de los Profesionales:** Se realizó de forma aleatoria, se implementó mediante la concepción de jornadas (mañana, tarde, mañana y tarde).
- **Disponibilidad de los Pacientes:** Se implementó haciendo un análisis de la muestra que fue suministrada por la clínica, la cual arrojó que la distribución de asignación de las citas fue de tipo Gaussiana, cuya particularidad fue que posee dos medias. (Ver Apéndice E: Distribución Pacientes). Con el propósito de evaluar el comportamiento de ACiT, se decidió implementar cuatro distribuciones adicionales: la primera sin ningún tipo de distribución, la segunda con distribución uniforme, la tercera con distribución exponencial y la cuarta con distribución logarítmica.

Apéndice E

Distribución Pacientes

En vista de que la disponibilidad de los pacientes puede depender de distintos factores, como por ejemplo, la temporada del año y la ocupación de los mismos, se decidió contemplar varios escenarios que permitieran abarcar de manera global el comportamiento que puede tener la disponibilidad de estos. La implementación de dichos escenarios se realizó por medio de la librería GSL[26], ya que ésta proporciona gran variedad de métodos para la generación de distribuciones de números pseudoaleatorios.

A continuación, se detallan las distribuciones que finalmente se desarrollaron en este proyecto:

1. Ninguna Distribución

Este escenario no guarda ninguna relación entre la disponibilidad de los diferentes pacientes, puesto que no se puede negar que la naturaleza de esta variable disponibilidad puede resultar impredecible. Por tal motivo, se generaron números pseudoaleatorios sin ningún tipo de comportamiento en particular.

2. Distribución Uniforme

Esta distribución parte del principio de que existe un número equilibrado de pacientes que están disponibles a lo largo de toda la jornada de atención de la entidad de salud.

3. Disribución Normal

Un análisis de la información suministrada por la clínica, logró evidenciar la distribución con que eran atendidos los pacientes. Dicha información se encuentra tabulada a continuación:

No. Rango	Horario	Pacientes Atendidos
1	07:00 - 08:00	130
2	08:00 - 09:00	363
3	09:00 - 10:00	358
4	10:00 - 11:00	295
5	11:00 - 12:00	128
6	12:00 - 13:00	19
7	13:00 - 14:00	174
8	14:00 - 15:00	294
9	15:00 - 16:00	305
10	16:00 - 17:00	317
11	17:00 - 18:00	196
12	18:00 - 19:00	52

Tabla E.1: No. Pacientes atendidos

Esta información corresponde al número de pacientes atendidos en el mes de noviembre del año 2011. Estos datos, se utilizaron para simular la distribución de la disponibilidad de los pacientes. Dicha distribución posee un comportamiento muy similar al de la distribución Normal o Gaussiana, cuya mayor concentración de los datos se encuentra en los rangos: 2, 3, 4, 8, 9 y 10. De igual forma, se observa que tal distribución posee dos medias (media aritmética o promedio), una en horas de la mañana y otra en horas de la tarde; específicamente presenta un número elevado de pacientes entre las 08:00 a.m. y las 11:00 a.m., en horas de la mañana, y entre las 02:00 p.m. y las 06:00 p.m., en horas de la tarde.

4. Distribución Exponencial

Esta distribución propone una mayor concentración de los datos al inicio de la jornada de atención de la entidad de salud, es decir, de que un mayor número de pacientes esté disponible en horas de la mañana.

5. Distribución Logarítmica

Contrario a la distribución exponencial, ésta presenta una mayor concentración de los datos al finalizar la jornada de atención de la entidad de salud, es decir que supone que un mayor número de pacientes está disponible en horas de la tarde.