

PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

FACULTAD DE CIENCIAS E INGENIERÍA



PONTIFICIA
UNIVERSIDAD
CATÓLICA
DEL PERÚ

IMPLEMENTACIÓN DE UN ALGORITMO GENÉTICO PARA OPTIMIZAR LA DISTRIBUCIÓN DEL AGUA EN EL RIEGO DE CULTIVOS

Tesis para optar el Título de Ingeniero Informático, que presenta el bachiller:

Ángel Rolando Pizarro Rodríguez

ASESOR: Rony Cueva Moscoso

Lima, febrero de 2015

RESUMEN

En la actualidad, existen diversas razones que generan un enorme problema de escasez de agua, entre las cuales se encuentra el incremento de su uso en el riego de cultivos debido a una ineficiente distribución; siendo esta una de las más alarmantes. La gestión del agua es un aspecto importante a considerar en las diversas actividades en las que se utiliza este recurso, particularmente, en la agricultura en donde gran parte del recurso hídrico está destinado a la irrigación de cultivos y terrenos.

Debido a esto se requiere un uso eficiente del agua, que reduzca pérdidas o costos de producción, con una buena distribución del recurso de modo que reduzca los problemas de drenaje y salinidad, con adecuado requerimiento de agua para los cultivos y así se obtenga una calidad apropiada en los alimentos. De la misma manera, conseguir un buen uso del recurso hídrico con la finalidad de poseer suficiente agua para el riego de cultivos de modo que se eviten problemas de producción de cultivos o vedas por falta de un adecuado manejo de agua.

Además, que se manejen aspectos y factores naturales que son importantes en la actividad agrícola en cuanto a la irrigación de cultivos, minimizando el despilfarro del recurso, pérdidas de cultivos y disminución de cosechas. Este manejo de factores naturales inclusive de acuerdo a las diversas y cambiantes temporadas de producción de cultivos.

El presente proyecto presenta el desarrollo de un algoritmo genético que permita optimizar la distribución del recurso hídrico en el riego de cultivos en distintas áreas o terrenos tomando en cuenta los diferentes factores que condicionan la fase de crecimiento del sembrío hasta llegar a la etapa de cosecha. Esto de manera que se busque una mejor selección para el regado de plantaciones, logrando así organizar su desarrollo y el ahorro del consumo vital como es el agua. La importancia del desarrollo de este tema recae en el buen uso y manejo del recurso hídrico así como otras condiciones naturales que son importantes también.

FACULTAD DE
**CIENCIAS E
 INGENIERÍA**
 ESPECIALIDAD DE
 INGENIERÍA INFORMÁTICA

 PONTIFICIA
**UNIVERSIDAD
 CATÓLICA**
 DEL PERÚ

TEMA DE TESIS PARA OPTAR EL TÍTULO DE INGENIERO INFORMÁTICO
TÍTULO: IMPLEMENTACIÓN DE UN ALGORITMO GENÉTICO PARA OPTIMIZAR LA DISTRIBUCIÓN DEL AGUA EN EL RIEGO DE CULTIVOS

ÁREA: CIENCIAS DE LA COMPUTACIÓN.

PROPONENTE: Ing. Rony Cueva Moscoso.

ASESOR: Ing. Rony Cueva Moscoso.

ALUMNO: Ángel Pizarro Rodríguez

CÓDIGO: 20084106

TEMA N°: 583
FECHA: San Miguel, 14 de diciembre de 2014

DESCRIPCIÓN

En la actualidad, existen diversas razones que generan un enorme problema de escasez de agua, entre las cuales se encuentra el incremento de su uso en el riego de cultivos debido a una ineficiente distribución; siendo ésta una de la más alarmante. La gestión del agua es un aspecto importante a considerar en las diversas actividades en las que se utiliza este recurso, particularmente, en la agricultura en donde gran parte del recurso hídrico está destinado a la irrigación de cultivos y terrenos.

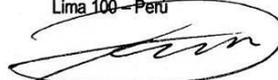
Debido a esto se requiere un uso eficiente del agua, que reduzca pérdidas o costos de producción, con una buena distribución del recurso de modo que reduzca los problemas de drenaje y salinidad, con adecuado requerimiento de agua para los cultivos de modo que se obtenga calidad en los alimentos. Además, que se manejen aspectos y factores naturales que son importantes en la actividad agrícola en cuanto a la irrigación de cultivos, minimizando el despilfarro del recurso, pérdidas de cultivos y disminución de cosechas.

El presente proyecto presenta el desarrollo de un algoritmo genético que permita optimizar la distribución del recurso hídrico en el riego de cultivos en distintas áreas o terrenos tomando en cuenta los diferentes factores que condicionan la fase de crecimiento del sembrío hasta llegar a la etapa de cosecha. Esto de manera que se busque una mejor selección para el regado de plantaciones, logrando así organizar su desarrollo y el ahorro del consumo vital como es el agua. La importancia del desarrollo de este tema recae en el buen uso y manejo del recurso hídrico así como otras condiciones naturales que son importantes también.

 Av. Universitaria 1801
 San Miguel, Lima – Perú

 Apartado Postal 1761
 Lima 100 – Perú

 Teléfono:
 (511) 626 2000 Anexo 4801


FACULTAD DE
**CIENCIAS E
 INGENIERÍA**
 ESPECIALIDAD DE
 INGENIERÍA INFORMÁTICA



PONTIFICIA
**UNIVERSIDAD
 CATÓLICA**
 DEL PERÚ

OBJETIVO GENERAL

Implementar un algoritmo genético para optimizar la gestión del agua en la irrigación de diversos cultivos.

OBJETIVOS ESPECÍFICOS

Los objetivos específicos del presente proyecto son:

- Desarrollar una herramienta que realice la generación de datos relacionados a los terrenos como el clima, el requerimiento de agua, número de hectáreas y otros de manera aleatoria, controlada y parametrizable utilizados en los algoritmos.
- Definir las estructuras de datos que permitan el manejo de los diversos factores asociados a los terrenos a regar como el clima, el requerimiento de agua, el beneficio de producción, entre otros.
- Definir una función objetivo (fitness) que utilice los diversos factores propios de los terrenos, además de considerar variables asociadas al problema como costo de producción, beneficio del cultivo, temperatura del cultivo, área del terreno y la distancia.
- Diseñar el pseudocódigo del algoritmo genético utilizado en la solución del problema de gestión de agua considerando diversos factores naturales y propios del problema.
- Implementar un algoritmo Grasp, en su fase construcción, que será utilizado tanto para la generación de una población inicial como la comparación de resultados.
- Realizar la experimentación numérica adecuada entre el algoritmo Grasp y el algoritmo genético de modo que se pueda comprobar que la ejecución del algoritmo genético, en el contexto del problema, genera una mejor solución.
- Implementar un prototipo funcional que soporte los algoritmos utilizados y efectúe la carga de datos generados respectivamente.

ALCANCE

El proyecto de investigación se basará en la creación de un algoritmo genético para resolver el problema de distribución de agua para el riego de cultivos. El proyecto se relaciona con las empresas de distribución comercial electrónica por la importancia y demanda que presentan en la actualidad.

Para el desarrollo del proyecto se implementarán dos algoritmos encargados de aportar a la solución del problema en general, un algoritmo meta-heurístico Grasp utilizado para la generación de la población inicial y la experimentación numérica y un algoritmo evolutivo Genético para resolver el problema planteado.

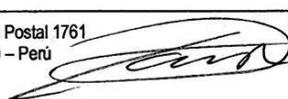
Tanto en el algoritmo de experimentación (Algoritmo Grasp) como en el de estudio (Algoritmo Genético) se considerarán diversos factores o variables naturales de modo que el proceso de riego de cultivos se ejecute bajo condiciones que realmente se manejan en la agricultura.

Máximo: 100 páginas

Av. Universitaria 1801
 San Miguel, Lima – Perú

Apartado Postal 1761
 Lima 100 – Perú

Teléfono:
 (511) 626 2000 Anexo 4801


DEDICATORIA

A mis padre Rolando quien siempre me aconsejó del duro y difícil camino que ofrece la vida en todos sus aspectos; A mi madre Noemí quien siempre llenaba de aliento y perseverancia los agotadores días de trabajo y A mis hermanos Rolando, Manuel, Rodrigo y Nova por llenar de alegría y tranquilidad ante la adversidad de las situaciones y problemas del día a día. Gracias por todo lo que significan para mí.

Ángel Pizarro

AGRADECIMIENTOS

Le agradezco a Dios por haberme guiado en todo este difícil camino en la universidad, por haberme brindado paz en los momentos adversos y sabiduría para tomar decisiones. Asimismo, agradezco a mi asesor Rony Cueva por la confianza y apoyo a lo largo del desarrollo del proyecto. Por último, a todos mis amigos, aprendí que ser honesto no puede ayudarte conseguir un montón de amigos, pero siempre te hacen obtener los correctos.

ÍNDICE GENERAL

RESUMEN	2
DEDICATORIA	5
AGRADECIMIENTOS	6
ÍNDICE GENERAL	7
ÍNDICE DE FIGURAS	10
ÍNDICE DE TABLAS	11
CAPÍTULO 1: GENERALIDADES	12
1 PROBLEMÁTICA	12
1.1 OBJETIVO GENERAL	14
1.2 OBJETIVOS ESPECÍFICOS	14
1.3 RESULTADOS ESPERADOS	15
2 HERRAMIENTAS, METODOLOGÍAS Y PROCEDIMIENTOS	16
2.1 MAPEO	16
2.2 HERRAMIENTAS	17
2.3 MÉTODOS, METODOLOGÍAS Y PROCEDIMIENTOS	17
2.3.1 RELACIONADO A LA GESTIÓN DE PROYECTO	17
2.3.2 RELACIONADO AL DESARROLLO DEL PROYECTO	18
3 ALCANCE	20
3.1 LIMITACIONES	22
3.2 RIESGOS	22
4 JUSTIFICACIÓN Y VIABILIDAD	23
4.1 JUSTIFICATIVA	23
4.2 VIABILIDAD	24
5 PLAN DE ACTIVIDADES	25
5.1 INICIACIÓN Y PLANIFICACIÓN	25
5.2 EJECUCIÓN Y CIERRE	26
CAPÍTULO 2: MARCO CONCEPTUAL	27
1 MARCO TEÓRICO	27
1.1 CONCEPTOS RELACIONADOS AL PROBLEMA	27

1.2	CONCEPTOS RELACIONADOS A LA SOLUCIÓN PROPUESTA	29
1.3	CONCLUSIÓN	33
1.4	MARCO REGULATORIO / LEGAL	33

2 ESTADO DEL ARTE **34**

2.1	MÉTODO USADO EN LA REVISIÓN DEL ESTADO DEL ARTE	34
2.2	FORMAS EXACTAS DE RESOLVER EL PROBLEMA	35
2.3	FORMAS APROXIMADAS DE RESOLVER EL PROBLEMA	36
2.4	PRODUCTOS COMERCIALES PARA RESOLVER EL PROBLEMA	38
2.5	PRODUCTOS NO COMERCIALES PARA RESOLVER EL PROBLEMA	40
2.6	CUADRO RESUMEN	41

CAPÍTULO 3: DISEÑO ESTRUCTURAL **42**

1 ESTRUCTURAS DE DATOS **42**

1.1	CROMOSOMA	42
1.2	ESTRUCTURA DE SELECCIÓN	45
1.3	ESTRUCTURA DE REQUERIMIENTOS	45
1.4	POBLACIÓN	48
1.5	ESTRUCTURA DE FITNESS	48
1.6	RESUMEN DE ESTRUCTURAS DE DATOS	49

2 FUNCIÓN OBJETIVO **50**

2.1	DEFINICIÓN	50
2.2	DETALLE DE VARIABLES	51

CAPÍTULO 4: MODELO COMPUTACIONAL GRASP **52**

1 ALGORITMO GRASP **52**

1.1	VARIABLES Y ESTRUCTURAS	52
1.2	PSEUDOCÓDIGO DEL ALGORITMO GRASP	53
1.3	EXPLICACIÓN DEL ALGORITMO GRASP	54
1.4	CALIBRACIÓN ALFA	57
1.5	CALIBRACIÓN NÚMERO DE ITERACIONES	59

CAPÍTULO 5: MODELO COMPUTACIONAL GENÉTICO **61**

1 ALGORITMO GENÉTICO **61**

1.1	VARIABLES Y ESTRUCTURAS	61
1.2	PSEUDOCÓDIGO DEL ALGORITMO GENÉTICO	62
1.3	EXPLICACIÓN DEL ALGORITMO GENÉTICO	63
1.4	PSEUDOCÓDIGO DEL PROCEDIMIENTO DE CONVERSIÓN	66
1.5	EXPLICACIÓN DEL PROCEDIMIENTO DE CONVERSIÓN	66
1.6	PSEUDOCÓDIGO DEL OPERADOR DE CASAMIENTO	67
1.7	EXPLICACIÓN DEL OPERADOR DE CASAMIENTO	68
1.8	PSEUDOCÓDIGO DEL OPERADOR DE MUTACIÓN	69
1.9	EXPLICACIÓN DEL OPERADOR DE MUTACIÓN	70
1.10	PSEUDOCÓDIGO DEL PROCEDIMIENTO DE LIMITACIÓN	72
1.11	EXPLICACIÓN DEL PROCEDIMIENTO DE LIMITACIÓN	72

1.12	PSEUDOCÓDIGO DE LA SELECCIÓN POR RULETA	73
1.13	EXPLICACIÓN DEL PROCEDIMIENTO DE SELECCIÓN	74
1.14	PSEUDOCÓDIGO DEL PROCEDIMIENTO DE REVISIÓN	75
1.15	EXPLICACIÓN DEL PROCEDIMIENTO DE REVISIÓN	76
1.16	PSEUDOCÓDIGO DE LA FUNCIÓN DE FITNESS	78
1.17	CALIBRACIÓN TASA DE CASAMIENTO	78
1.18	CALIBRACIÓN TASA DE MUTACIÓN	80
 CAPÍTULO 6: INTERFAZ GRÁFICA		82
1	PROGRAMA GENERADOR DE DATOS	82
1.1	FUNCIONAMIENTO DE GENERACIÓN	83
1.2	EXPORTACIÓN DE DATOS GENERADOS	85
2	VENTANAS DE LA APLICACIÓN	85
2.1	PANEL DE CARGA DE DATOS DE ENTRADA	86
2.2	PANEL OPTIMIZACIÓN DE SELECCIÓN DE TERRENOS	87
2.3	PANEL VISUALIZACIÓN DE RESULTADOS DE SIMULACIÓN	88
3	APLICACIÓN PRINCIPAL	89
3.1	FUNCIONAMIENTO INTERNO DE LA APLICACIÓN	89
 CAPÍTULO 7: MÉTODOS ESTADÍSTICOS Y RESULTADOS		90
1	EXPERIMENTACIÓN NUMÉRICA	90
1.1	INTRODUCCIÓN	90
1.2	RECOLECCIÓN DE RESULTADOS	91
1.3	PRUEBA K-S DE KOLMOGOROV-SMIRNOV	93
1.4	PRUEBA F DE FISHER	95
1.5	PRUEBA Z	96
 CAPÍTULO 8: CONCLUSIONES Y TRABAJOS FUTUROS		98
1	OBSERVACIONES	98
2	CONCLUSIONES	98
2.1	RECOMENDACIONES Y TRABAJOS FUTUROS	99
 REFERENCIAS BIBLIOGRÁFICAS		100

ÍNDICE DE FIGURAS

Figura 1: Utilización Mundial del agua por sectores	12
Figura 2: Diagrama de Gantt - Iniciación y Planificación	25
Figura 3: Diagrama de Gantt - Ejecución y Cierre	26
Figura 4: Interfaz del sistema WINQSB	38
Figura 5: Interfaz del software SAS/OR	38
Figura 6: Interfaz de software ALT-ERGO	39
Figura 7: Interfaz del software IMS	39
Figura 8: Ejemplo de conjunto de terrenos.....	42
Figura 9: Ejemplo de Gen	43
Figura 10: Ejemplo de Cromosoma.....	43
Figura 11: Ejemplo de una estructura selección.....	45
Figura 12: Ejemplo de Cultivo	46
Figura 13: Ejemplo de Terreno	46
Figura 14: Ejemplo de estructura fitness	49
Figura 15: Representación de Fitness Promedio por Iteraciones	60
Figura 16: Ejemplo Casamiento.....	69
Figura 17: Ejemplo Mutación	72
Figura 18: Ejemplo Selección por Ruleta	75
Figura 19: Representación de Mejora Promedio por Tasa Casamiento	79
Figura 20: Representación de Mejora Promedio por Tasa Mutación.....	81
Figura 21: Pantalla de Herramienta generadora de datos.....	84
Figura 22: Ventana de pestaña Carga Inicial	86
Figura 23: Ventana de pestaña Optimización	87
Figura 24: Ventana de pestaña de Visualización	88
Figura 25: Diagrama de flujo del funcionamiento de aplicación.....	89
Figura 26: Representación de Fitness Promedio por Muestra utilizada.....	92
Figura 27: Comparación algoritmo GRASP vs. Genético propuesto	99
Figura 28: Comparación Población Inicial Aleatoria vs GRASP del Genético	99

ÍNDICE DE TABLAS

Tabla 1: Mapeo de Herramientas.....	17
Tabla 2: Riesgos del Proyecto	23
Tabla 3: Estructura General de un Algoritmo GRASP	31
Tabla 4: Estructura General de la Fase Construcción de GRASP.....	31
Tabla 5: Estructura General de un Algoritmo Genético	32
Tabla 6: Cuadro Comparativo de soluciones para el problema	41
Tabla 7: Representación clase Gen.....	44
Tabla 8: Representación clase Individuo.....	44
Tabla 9: Representación clase Selección	45
Tabla 10: Representación clase Cultivo.....	47
Tabla 11: Representación clase Terreno	47
Tabla 12: Representación clase Zona.....	47
Tabla 13: Representación clase Población	48
Tabla 14: Representación clase Fitness	49
Tabla 15: Resumen de estructuras de datos.....	49
Tabla 16: Estructuras propias de Algoritmo GRASP	53
Tabla 17: Pseudocódigo del Algoritmo GRASP	53
Tabla 18: Pseudocódigo del procedimiento Construir Solución.....	54
Tabla 19: Calibración Alfa - Experimento 1	57
Tabla 20: Calibración Alfa - Experimento 2	58
Tabla 21: Calibración Alfa - Experimento 3	58
Tabla 22: Calibración Alfa - Experimento 4	58
Tabla 23: Calibración Número Iteraciones	59
Tabla 24: Estructuras propias de Algoritmo Genético	62
Tabla 25: Pseudocódigo del Algoritmo Genético.....	62
Tabla 26: Pseudocódigo del procedimiento Convertir Cromosomas	66
Tabla 27: Pseudocódigo del operador de Casamiento.....	67
Tabla 28: Pseudocódigo del operador de Mutación	70
Tabla 29: Pseudocódigo del procedimiento Limitar Población	72
Tabla 30: Pseudocódigo de la selección por Ruleta.....	73
Tabla 31: Pseudocódigo del procedimiento Revisar Fin.....	76
Tabla 32: Pseudocódigo de la función fitness	78
Tabla 33: Calibración Tasa de Casamiento	79
Tabla 34: Calibración Tasa de Mutación	80
Tabla 35: Ejemplo de generación de datos	84
Tabla 36: Ejemplo de Archivo de Entrada.....	85
Tabla 37: Hipótesis Prueba K-S GRASP.....	93
Tabla 38: Experimentación Prueba K-S GRASP.....	94
Tabla 39: Hipótesis Prueba K-S Genético.....	94
Tabla 40: Experimentación Prueba K-S Genético	94
Tabla 41: Hipótesis Prueba F.....	95
Tabla 42: Experimentación Prueba F.....	95
Tabla 43: Hipótesis Prueba Z – Experimento 1	96
Tabla 44: Experimentación Prueba Z – Experimento 1	96
Tabla 45: Hipótesis de prueba Z – Experimento 2	97
Tabla 46: Experimentación Prueba Z – Experimento 2	97

CAPÍTULO 1: GENERALIDADES

1 Problemática

Desde hace muchos años, los recursos naturales han sido utilizados en muchos aspectos para la contribución del bienestar de la sociedad humana y su desarrollo de manera directa. En particular, el agua es uno de los principales recursos naturales que ha sido empleado para el desarrollo de vida de las personas así como las actividades de producción que estas realizan. El recurso hídrico es utilizado a nivel mundial y su demanda varía básicamente de acuerdo al incremento del nivel poblacional y los sectores de producción que han sido establecidos en la sociedad. Aproximadamente, el 70% del agua es destinada a la actividad agrícola, el 20% al sector industrial y un 10% al consumo doméstico poblacional, tal como se muestra en la Figura 1 [FAO, 2013].

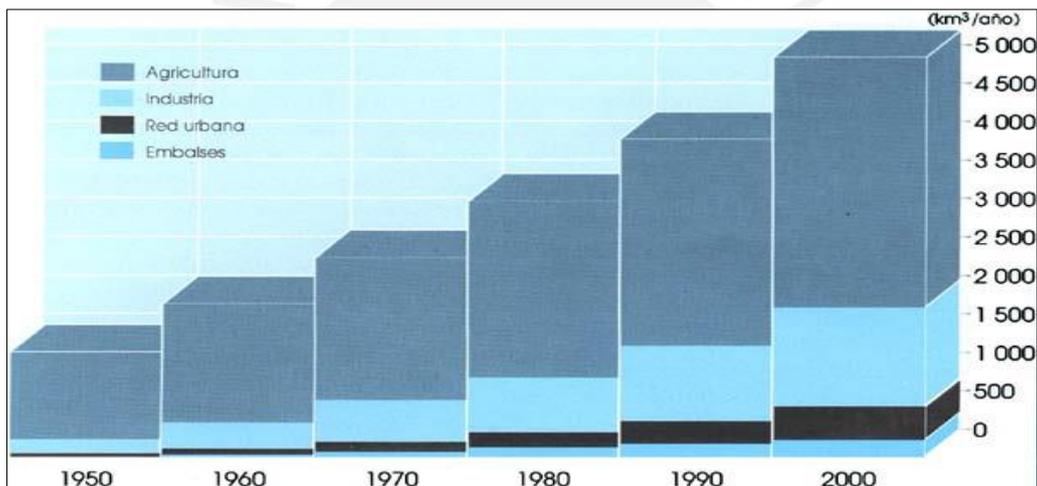


Figura 1: Utilización Mundial del agua por sectores

Por lo tanto, existe una notable necesidad e importancia del recurso hídrico en la sociedad. Sin embargo, este recurso se encuentra afectado por diversos problemas que si bien no han sido recientemente identificados, sus impactos vienen intensificándose en cada año. El principal problema es la Escasez del Agua debido a diversos cambios climatológicos y factores propios ocasionados por la sociedad. Según recientes estudios científicos, uno de cada cinco países tendría problemas de escasez de agua antes de finalizar el año 2030 [FAO, 2013]. Este problema tiene como origen, principalmente, la ausencia de lluvias y precipitaciones que viene sufriendo el mundo [MINAG, 2013]. Numerosas cuencas fluviales explotadas intensivamente, de las principales regiones productoras de alimentos, ya trabajan al límite del recurso hídrico. Inclusive el panorama es mucho más preocupante debido a

que se podría desatar una crisis de abastecimiento de no tomar las medidas adecuadas para enfrentar este problema [FAO, 2013].

Existen diversas razones que provocan la escasez de agua mencionada entre las que se pueden encontrar el crecimiento proyectado de las poblaciones urbanas, el cual incrementa las presiones localizadas sobre los recursos de agua para consumo urbano (la demanda por los recursos de agua ha excedido el crecimiento de la población por un factor de dos o más durante los últimos cien años) y el incremento del uso del recurso en las actividades de producción [ANA, 2013]. El Perú, un estado dotado de un gran número de recursos naturales, no es ajeno a este problema de escasez. La Autoridad Nacional del Agua (ANA) ha registrado un déficit del recurso hídrico del 25% [FAO, 2013]. Esto como consecuencia de la escasa presencia de lluvias en las temporadas anteriores. Si bien la escasez del agua se debe a problemas climatológicos, un aspecto que altera de sobremanera este problema es el uso inadecuado que se hace del recurso hídrico por parte de la población. Aproximadamente, el 50% del agua destinada al consumo poblacional es desperdiciada de manera alarmante mientras que el 65% del agua destinada al sector agrícola es distribuida inadecuadamente [ANA, 2013]. Como se puede notar, el mal uso del recurso hídrico afecta directamente al serio problema de la Escasez de Agua. Particularmente, el sector agrícola registra pérdidas exorbitantes del recurso debido a una inadecuada distribución del agua que se realiza en sus actividades.

Si bien la agricultura peruana, considerada una de las doce actividades más diversas en el mundo, viene sufriendo problemas de erosión, salinización, minifundio, asistencia técnica, crédito agrario, organizaciones, seguridad alimentaria, empleo, entre otros, es el agotamiento de fuentes de agua (ríos, lagunas, arroyos, acuíferos) el mayor problema que sufre para el desarrollo de sus actividades, particularmente, el riego de terrenos y cultivos [MINAG, 2013].

La forma de distribución del agua en el riego de cultivos es una de las principales razones que influye en el agotamiento del agua [ANA, 2013]. Si bien la distribución del recurso hídrico está regulada, existe un deficiente uso del agua en la actividad de riego de manera que se despilfarra mucha cantidad de este recurso. La problemática con el recurso hídrico genera desde bajas eficiencias de riego utilizadas por el riego por gravedad (produciendo problemas de drenaje y salinidad, cultivos de alta demanda de agua en la costa, erosión en los suelos de la sierra por prácticas agronómicas realizadas) hasta tarifas de agua muy bajas, que no cubren los costos de operación y

mantenimiento de los sistemas de riego e incentivos para mejorar la tecnología y capitalizar la agricultura.

Por otro lado, la falta de consideración de factores y condiciones naturales al momento del regado de terrenos es un tema de suma importancia que muchas veces pasa desapercibido [MINAG, 2013]. Entre los factores poco considerados al momento de realizar el riego de cultivos se encuentran el tipo de planta a cultivar, número de hectáreas, requerimiento hídrico, ciclo de vida de la planta, costo del cultivo, beneficio del cultivo, condición de la tierra, stress hídrico, temperatura del área de cultivo y distancia del terreno al reservorio de agua, los mismos que por lo general al ser obviados ocasionan un mal manejo y distribución del recurso [FAO, 2013].

En vista de la problemática introducida anteriormente y debido a las características que esta presenta, el problema puede ser traducido a un problema de satisfacibilidad (*Satisfiability Problem - SAT*) para lo cual el presente proyecto pretende brindar una solución que optimice la distribución del recurso hídrico en el riego de cultivos en distintos terrenos tomando en cuenta los principales factores que se manejan en la actividad agrícola. La adecuada distribución de agua se realizará mediante la búsqueda de una óptima selección de plantaciones para ser regadas, logrando así organizar su desarrollo, manejar la cantidad de agua disponible para una temporada y el ahorro del recurso vital. La importancia del desarrollo de este tema recae en el buen uso y manejo del recurso hídrico así como otras condiciones naturales que son también relevantes.

1.1 Objetivo general

Implementar un algoritmo genético para optimizar la distribución del agua en el riego de cultivos.

1.2 Objetivos específicos

Los objetivos específicos que se plantean en este proyecto son:

- Objetivo 1: Desarrollar un programa que permita construir de manera aleatoria, controlada y parametrizable información relativa de los terrenos como el área, la distancia, el rendimiento, el costo, el beneficio y la necesidad de agua.

- Objetivo 2: Definir las estructuras de datos que permitan el manejo de los diversos factores asociados a los terrenos así como el procesamiento de estos.
- Objetivo 3: Definir una función objetivo que permita medir la bondad de cada terreno y el aporte que ofrece a la solución del problema.
- Objetivo 4: Diseñar el pseudocódigo del algoritmo genético utilizado para resolver el problema de distribución de agua considerando los factores naturales.
- Objetivo 5: Implementar un algoritmo GRASP en su fase construcción utilizado para generar una población inicial y comparar sus resultados obtenidos.
- Objetivo 6: Realizar la experimentación numérica adecuada que permita comprobar que el algoritmo genético ofrece mejores soluciones comparado con los resultados del algoritmo GRASP.
- Objetivo 7: Implementar un prototipo funcional que efectúe la carga de datos, ejecute los algoritmos implementados y muestre una solución al problema.

1.3 Resultados esperados

Los resultados esperados que se plantean en este proyecto son:

- ✓ Resultado 1 para el objetivo 1: Aplicación que permita generar aleatoriamente datos de entrada necesarios para la ejecución del algoritmo GRASP y Genético.
- ✓ Resultado 2 para el objetivo 2: Documento que muestre el diseño de las estructuras de datos utilizadas para representar y manipular la información.
- ✓ Resultado 3 para el objetivo 3: Documento que muestre el diseño de la función objetivo que permita la evaluación de los terrenos a ser regados.
- ✓ Resultado 4 para objetivo 4: Documento que muestre el diseño del pseudocódigo del algoritmo genético así como la explicación de su funcionamiento, adaptación de los operadores genéticos y la función fitness.

- ✓ Resultado 5 para objetivo 5: Documento que muestre el algoritmo GRASP Construcción adaptado al problema para ser usado en la generación de una población inicial y la comparación de resultados.
- ✓ Resultado 6 para objetivo 6: Documento que muestre los métodos estadísticos utilizados en la experimentación numérica así como la interpretación de los resultados obtenidos.
- ✓ Resultado 7 para objetivo 7: Prototipo funcional que permita la carga de datos, ejecución de algoritmos y visualización los resultados obtenidos luego de la simulación realizada.

2 Herramientas, metodologías y procedimientos

En esta sección, se presentan los resultados esperados junto con las herramientas, y metodologías que se emplearon para su desarrollo. Asimismo, se procede a explicar estas metodologías tanto para la gestión del proyecto como la gestión del desarrollo.

2.1 Mapeo

En la Tabla 1, se muestran las metodologías empleadas para los resultados esperados definidos.

Resultados Esperados	Herramientas/ Metodologías a Utilizarse
RE1: Aplicación que permita generar aleatoriamente datos iniciales necesario para la ejecución de los algoritmos.	Extreme Programming es una metodología orientada al desarrollo y programación.
RE2: Documento que muestre el diseño de las estructuras de datos para que permitan el manejo de datos.	Extreme Programming es una metodología orientada al desarrollo y programación.
RE3: Documento que muestre el diseño de la función objetivo así como su funcionamiento al medir la bondad.	Metodología del algoritmo genético es una metodología que permite guiar la implementación de la función fitness.
RE4: Documento que muestre el diseño del pseudocódigo y la explicación del algoritmo genético.	Metodología del algoritmo genético es una metodología que permite guiar la implementación del algoritmo.

Resultados Esperados	Herramientas/ Metodologías a Utilizarse
RE5: Documento que muestre el diseño del pseudocódigo del algoritmo GRASP Construcción así su funcionamiento.	Metodología del algoritmo GRASP es una metodología que permite guiar la implementación del algoritmo.
RE6: Documento que muestre los métodos estadísticos y los resultados de la experimentación numérica realizada.	ANOVA es una prueba estadística usada para la comparación de medias entre dos tratamientos.
RE7: Prototipo funcional que permita la ejecución de los algoritmos y muestre los resultados obtenidos.	Extreme Programming es una metodología orientada al desarrollo y programación.

Tabla 1: Mapeo de Herramientas

2.2 Herramientas

a) Netbeans

El uso de esta herramienta se justifica ya que permite administrar los proyectos de programación de manera sencilla. Además, cuenta con funcionalidades simples de edición para la implementación de los algoritmos. Finalmente, está relacionado con el lenguaje de programación Java [NETBEANS, 2013].

b) Java

El uso de este lenguaje se justifica debido a que provee programación orientada a objetos, el cual se conoce. Este lenguaje es simple, distribuido, interpretado, sólido, seguro y portable. Además, es adecuado para el IDE Netbeans que es utilizado para este proyecto como ya se mencionó [JAVA, 2013].

2.3 Métodos, metodologías y procedimientos

Para cumplir con los objetivos del proyecto es necesario emplear algunas metodologías relacionadas a la gestión y desarrollo del proyecto.

2.3.1 Relacionado a la gestión de proyecto

a) PMBOK

El marco PMBOK es utilizado para la gestión de integración del proyecto, dado que esta metodología es ampliamente aceptada por el estándar de proyectos. PMBOK cuenta con nueve áreas de conocimiento (en su cuarta edición) distribuidos en cinco grupos de proceso.

Al momento de emplear PMBOK se debe tener en cuenta que, debido a la naturaleza del presente proyecto, no se usa todas las áreas del marco sino solo aquellas que se adaptan mejor a las necesidades [PMBOK, 2009].

Como se mencionó, en el presente proyecto solo se utilizó las siguientes áreas:

- Gestión del Tiempo del Proyecto
- Gestión de los Riesgos del Proyecto

2.3.2 Relacionado al desarrollo del proyecto

a) Extreme Programming (XP)

Extreme Programming (XP) ha sido seleccionado debido a que ofrece una manera ágil de realizar software sin descuidar la parte esencial del desarrollo. La particularidad de esta metodología es que se centra más en la programación que en la documentación del proyecto. [BECK, ANDRES 2004].

Esta metodología es usada tanto en la implementación de la herramienta generadora de datos como en la implementación de algoritmos, se debe tener en cuenta que no se usan todos los valores y etapas que posee la metodología sino solo aquellos que mejor se adapten.

Entre los valores y etapas de Extreme Programming que se tomaron en cuenta en este proyecto se encuentran [TUPIA, 2005]:

- Simplicidad
- Coraje
- Integración constante.
- Validación y pruebas.

b) Metodología del algoritmo genético

Esta metodología intenta representar un proceso de evolución natural, el cual comienza con una población inicial de soluciones potenciales (individuos). El proceso continúa con la selección de soluciones individuales de la población de acuerdo a un proceso estocástico que favorece aquellos individuos con mejor

desarrollo. Su información genética es recombinada y modificada siguiendo reglas de transición probabilística, tales como reproducción y mutación para formar así una nueva población [GOLDBERG, 1989]. La población actual va adoptando soluciones nuevas. El proceso se repite hasta lograr la convergencia o hasta que se alcanza un número máximo de evaluaciones [CALDERON, 2002].

La justificación del uso de esta metodología radica en que el presente proyecto de fin de carrera consiste, principalmente, en adaptar un algoritmo genético que resuelva el problema de distribución del agua por lo que su empleo es necesario.

c) Metodología del algoritmo GRASP

Para construir una solución en lugar de considerar apenas un candidato disponible para el seleccionado (criterio usado por el algoritmo goloso), la metodología GRASP crea un conjunto de candidatos de modo que selecciona uno de ellos aleatoriamente. Por tal motivo, la metodología permite construir varias soluciones diferentes. La calibración de la constante de relajación (alfa α) en la metodología es muy relevante para los resultados [GANOZA, SOLANO 2004].

La justificación del uso de esta metodología radica en que es necesario implementar un algoritmo para la generación de una población inicial así como la posterior comparación de resultados con el algoritmo de Genético.

d) Análisis de la varianza (ANOVA)

Anova es una prueba estadística usada comúnmente para la comparación de dos o más medias de alguna población Anova. Su gran utilidad es identificada tanto en el sector industrial como en el control de procesos y métodos analíticos.

- **Prueba K-S**

La prueba de Kolmogorov-Smirnov es un procedimiento de bondad de ajuste que permite medir el grado de concordancia existente entre la distribución de un conjunto de datos y una distribución teórica específica. Su objetivo es

señalar si los datos provienen de una población que tiene la distribución teórica especificada [UNIVERSIDAD DE VALENCIA, 2010].

La razón de usar esta prueba es asegurar que los resultados se comportan normalmente para así poder aplicar la prueba Z de comparación de dos muestras.

- **Prueba F**

La prueba F de Fisher se emplea para probar si dos muestras provienen de poblaciones que poseen varianzas iguales. Esta prueba es útil para determinar si una población normal tiene mayor variación que otra o cuando se trata de comparar varias medias poblacionales. La comparación simultánea de varias medias poblacionales se conoce como análisis de varianza [RETURETA, 2010].

La razón de usar esta prueba es garantizar la calidad de las soluciones ya que analiza la variación de las poblaciones. De este modo, se justifica su elección.

- **Prueba Z**

La prueba Z se basa en la diferencia de proporciones entre dos muestras. Bajo la hipótesis nula, se supone que las proporciones de dos poblaciones son iguales y son combinadas para así determinar si existe una diferencia en la proporción de éxitos entre los grupos (dos colas) o un grupo tiene mayor proporción (una cola) [BERENSON, LEVINE, KREHBIEL 2006].

La razón de usar esta prueba radica en que para comparar los algoritmos GRASP y Genético es necesario usar un número determinado de muestras. Debido a que son 40 las muestras a evaluar, es adecuada esta prueba estadística.

3 Alcance

El proyecto de investigación pretende optimizar la distribución del agua en áreas de cultivo. Esta óptima distribución se consigue mediante la selección de ciertos terrenos de modo que estos puedan ser regados según sus requerimientos de agua sin exceder la disponibilidad de agua con la que se cuenta en el reservorio. De esa

manera, se estaría optimizando la distribución del recurso hídrico y empleando un buen uso del mismo.

Asimismo, los terrenos que son seleccionados para ser regados dependerán de la disponibilidad total de agua que se posea. Es decir, no necesariamente todos los terrenos serán regados ya que cada uno posee sus propios requerimientos hídricos y son seleccionados de acuerdo a sus aportes sin sobrepasar la cantidad total del recurso. Para lograr este objetivo se toman en cuenta ciertas variables que pueden ser consideradas las más importantes para la selección. Los factores considerados son:

- Área del terreno: es el espacio del terreno en donde se pretende realizar el riego del cultivo.
- Costo de producción: es el costo que demanda sembrar un cultivo específico en una zona.
- Beneficio del cultivo: es el beneficio que se obtiene del cultivo después de llegar a la etapa de cosecha y realizar su posible venta.
- Distancia al reservorio de agua: es la longitud del trayecto entre la fuente de agua y el terreno a ser regado.
- Rendimiento del cultivo: es la productividad que genera el terreno para la siembra y riego del cultivo.
- Requerimiento de agua: es la cantidad de agua que requiere el cultivo para irrigar su área de terreno.

El proyecto se basa en la implementación de un algoritmo genético para resolver el problema de distribución del agua. Para lograr este objetivo, previamente, se realiza la implementación de un algoritmo meta-heurístico GRASP con el fin de obtener una población inicial y ser usado en la experimentación numérica. La generación de una población inicial es recomendable, según la metodología del algoritmo genético, de modo que el algoritmo principal comience su ejecución a partir de soluciones adecuadas reduciendo así el tiempo de convergencia de resultados.

La experimentación numérica entre ambos algoritmos es necesaria para comprobar la calidad de los resultados que se obtienen de un algoritmo respecto del otro. Si bien el proyecto está orientado a la implementación del algoritmo genético se realiza la experimentación correspondiente con el fin de verificar que los resultados ofrecidos por el algoritmo Genético son mejores que los ofrecidos por el algoritmo GRASP.

Tanto en el algoritmo de experimentación (GRASP) como en el de estudio (Genético) se consideran los diversos factores o variables mencionadas. Cabe resaltar que estas y solo estas variables son consideradas en el desarrollo del proyecto de fin de carrera.

3.1 Limitaciones

Entre las limitaciones del proyecto se encuentran los siguientes aspectos:

- El tiempo de ejecución se ve afectado por las características de software y hardware de la computadora en que se ejecuta (complejidad de los algoritmos y el número de ejecuciones requeridas para obtener resultados adecuados).
- El número de variables afecta directamente a la complejidad del problema. Por lo tanto, no se agregarán más variables a las ya establecidas en el alcance.
- El operador de inversión, establecido en la metodología del algoritmo genético, no será desarrollado para el presente proyecto.

3.2 Riesgos

En la Tabla 2, se muestran los riesgos identificados que pueden afectar el proyecto.

Id	Riesgo	Prob.	Imp.	Sev.	Amenaza	Plan de Mitigación
R1	Entregables revisados a destiempo.	40%	4	Alta	Mala planificación del proyecto.	Revisión de cronograma de plazos.
R2	Rechazo de entregables.	20%	2	Alta	Mala planificación del proyecto.	Revisión de cronograma de plazos.
R3	No realizar entregables.	20%	2	Alta	Mala planificación del proyecto.	Revisión de cronograma de plazos.
R4	No presentar entregables.	30%	4	Alta	Pérdida de información.	Respaldo de información.
R6	Desaprobar el curso.	10%	2	Baja	Pérdida de información.	Respaldo de información.
R7	Mal uso de herramienta.	25%	5	Alta	Desconocimiento de herramientas.	Aprender con anticipación

Id	Riesgo	Prob.	Imp.	Sev.	Amenaza	Plan de Mitigación
R8	Errores por mal manejo de software.	50%	4	Alta	Desconocimiento de herramientas nuevas.	Manejar programas similares.

Tabla 2: Riesgos del Proyecto

4 Justificación y viabilidad

El presente proyecto posee una justificación y viabilidad de desarrollo de acuerdo al beneficio y los recursos que este administra, respectivamente.

4.1 Justificativa

La herramienta informática basada en algoritmos genéticos que se desarrolla en este proyecto generaría beneficios en varios aspectos. Entre los beneficiarios se pueden encontrar, en primer lugar, a los agricultores ya que podrán manejar mejor la escasa cantidad de agua que posee para una determinada estación, de manera más adecuada y reduciendo las pérdidas del mismo. De esta manera, obtendrán mayores ingresos luego de haber reducido sus costos de producción e inclusive la mejora de su calidad de vida. En segundo lugar, la población consumidora de alimentos será beneficiada ya que se obtendrán cultivos de mejor calidad para consumo masivo reduciendo los problemas ocasionados por la falta de agua o sobrecarga de la misma en la distribución del recurso. Por último, las entidades reguladoras del recurso hídrico podrán realizar un mejor control del uso del agua. El presupuesto aproximado de agua para cada estación, procedente de lluvias entre otros aspectos, podrá ser manejada de manera responsable contribuyendo a enfrentar los períodos de escasez de agua y las temporadas de baja producción de alimentos o “vedas” (como se conoce en el ámbito agrícola) por la baja disponibilidad de agua.

Para el desarrollo del proyecto, el uso del algoritmo genético facilitará el tiempo de respuesta generando soluciones óptimas en un tiempo aceptable. Dada la naturaleza del problema, el proyecto servirá como base para el desarrollo de futuras herramientas que intenten optimizar la distribución del agua en otros contextos.

4.2 Viabilidad

Para la viabilidad del proyecto, se verifica que existe información referente al problema y su solución así como del proceso de producción agrícola y riego de cultivos.

A continuación, se presentan factores que influyen en la viabilidad del proyecto:

a) Viabilidad Técnica

En el aspecto técnico, es importante señalar que se poseen los conocimientos para el manejo de las herramientas Java y Netbeans ya que estas han sido desarrolladas en los últimos semestres de la carrera de Ingeniería Informática. Adicionalmente, se cuenta con la asesoría tanto del asesor como de profesores que tienen dominio del problema a resolver.

Finalmente, las herramientas a utilizar son de fácil acceso ya que son gratuitas, por lo que no será un impedimento para el desarrollo normal del proyecto.

b) Viabilidad Temporal

Para el proyecto a desarrollar se ha realizado un cronograma con una serie de actividades y fechas de entrega de los resultados de modo que se siga un orden de plazos para cada actividad del proyecto. Este cronograma es necesario para la realización de cada actividad e inclusive para el manejo de un tiempo extra de modo que se pueda aplicar mecanismos de control

Por lo tanto, la viabilidad temporal está asegurada para el desarrollo del proyecto en el período de tiempo establecido. La consideración del tiempo extra que se maneja es importante y será de gran utilidad para la culminación del trabajo.

c) Viabilidad económica

En este aspecto, el presente proyecto no tendrá gastos ya que las herramientas son recursos gratuitos. En cuanto a los equipos utilizados son de uso propio de modo que no se incurrirá en gastos de hardware. Las tecnologías manejadas son las adecuadas y suficientes para el desarrollo del proyecto debido al alto nivel de procesamiento en la ejecución de los algoritmos.

Por otro lado, no se contará con gastos por personal de desarrollo debido a que el trabajo es de responsabilidad propia y será realizado individualmente.

5 Plan de actividades

Para el presente proyecto se ha desarrollado un cronograma de actividades de modo que se realice un control adecuado del desarrollo del mismo.

5.1 Iniciación y planificación

A continuación en la Figura 2, se definen las actividades desarrolladas en la etapa de iniciación y planificación del proyecto a realizar.

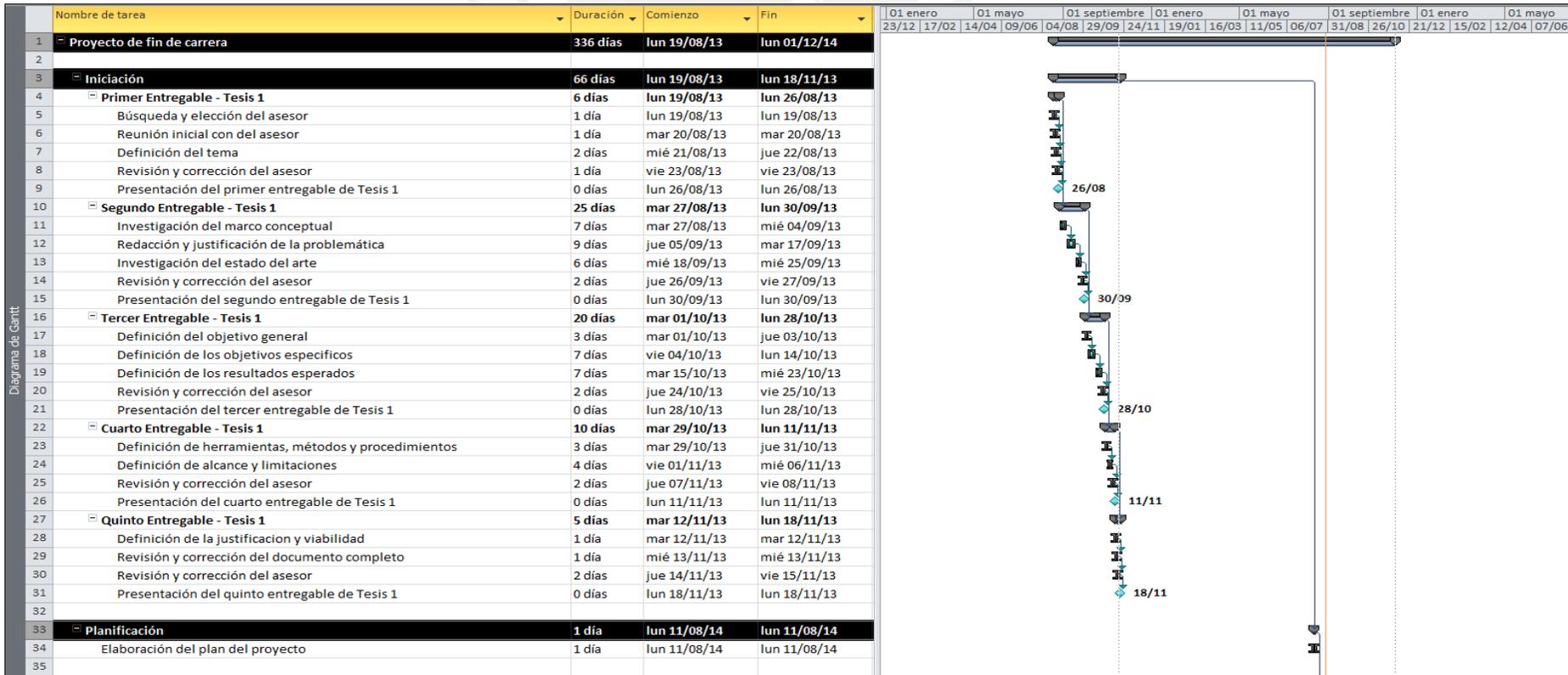


Figura 2: Diagrama de Gantt - Iniciación y Planificación

5.2 Ejecución y cierre

A continuación en la Figura 3, se definen las actividades desarrolladas en la etapa de ejecución y cierre del proyecto que se pretende realizar.

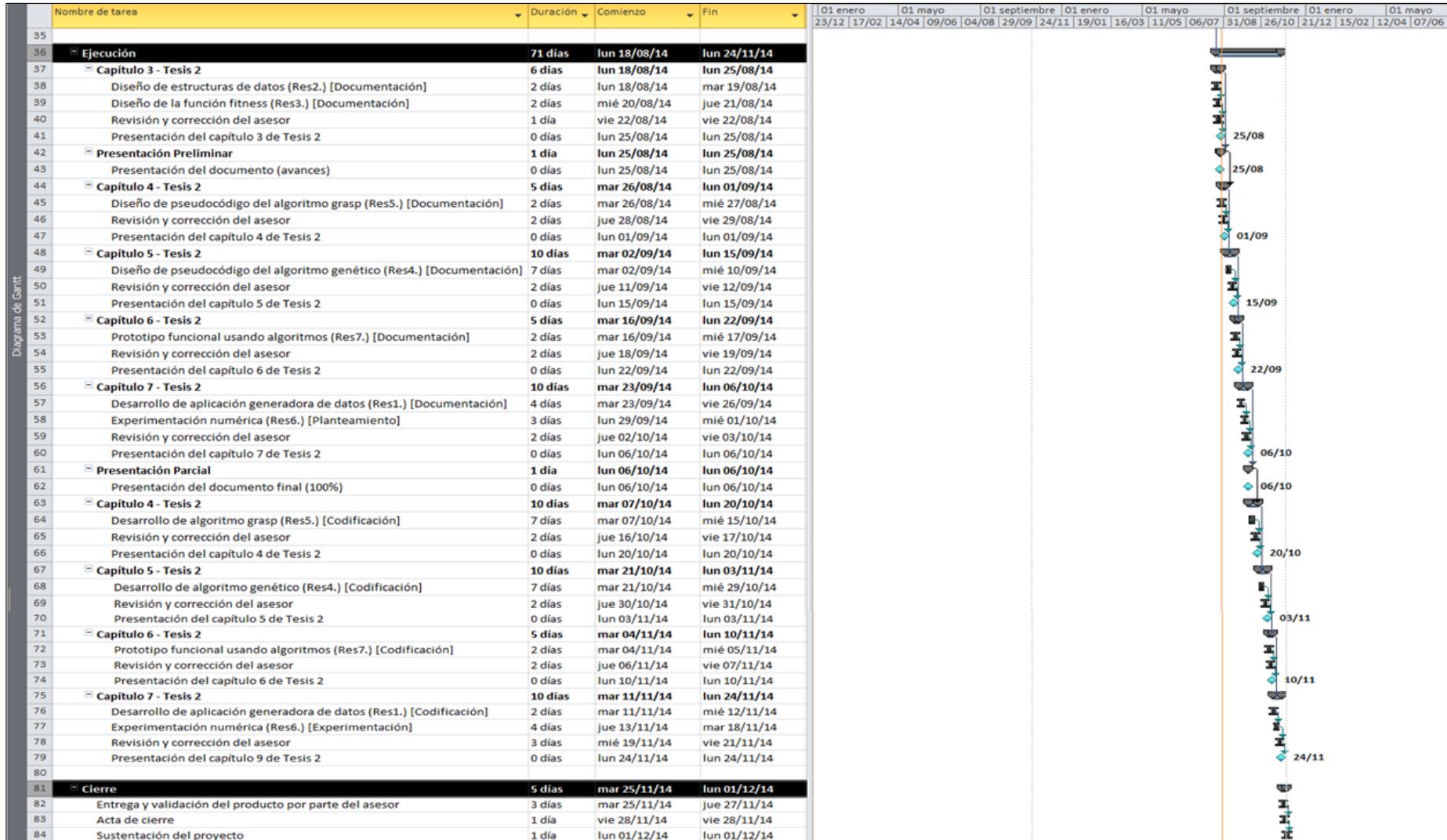


Figura 3: Diagrama de Gantt - Ejecución y Cierre

CAPÍTULO 2: MARCO CONCEPTUAL

1 Marco teórico

En el siguiente apartado, se procede a definir los conceptos relacionados a la problemática presentada, Estos conceptos contienen denominaciones claves que describen este contexto del problema. Asimismo, se explican aquellas nociones relacionadas a la propuesta de solución.

1.1 Conceptos relacionados al problema

▪ Definiciones según Ministerio de Agricultura y Riego del Perú

Se entiende por [MINAG, 2013]:

1. Recurso De Agua: El agua es un recurso natural, muy necesario pero vulnerable, que se renueva a través del ciclo hidrológico y que tiene un valor social, ambiental y estratégico para el desarrollo económico y social del país. Los aspectos climáticos y geográficos de nuestro país hacen del agua un recurso abundante en la región amazónica y con escasez en la costa, y por épocas en la sierra. Asimismo, la distribución del agua a través del año tiene una estacionalidad marcada, lo cual plantea un reto importante para el manejo del agua.
2. Suelo: El suelo se define como el material mineral y orgánico, no consolidado, de la capa superior de la tierra, el cual sirve como medio natural para el crecimiento de plantas terrestres. Este material ha sido expuesto a factores ambientales como el clima, los macro y micro organismos, que han actuado sobre la roca madre por cierto periodo de tiempo, para producir un suelo con características propias en su composición física, química, biológica y morfológica (*Soil Science Society of America*).
3. Recursos Naturales (RRNN): La temática de recursos naturales del Perú que aquí se presenta, describe los principales componentes de la naturaleza que son aprovechados por el humano en el territorio peruano.

▪ Definiciones en la terminología agrícola según FAO.

Según la Organización de las Naciones Unidas para la Alimentación y la Agricultura se entiende por [FAO, 2013]:

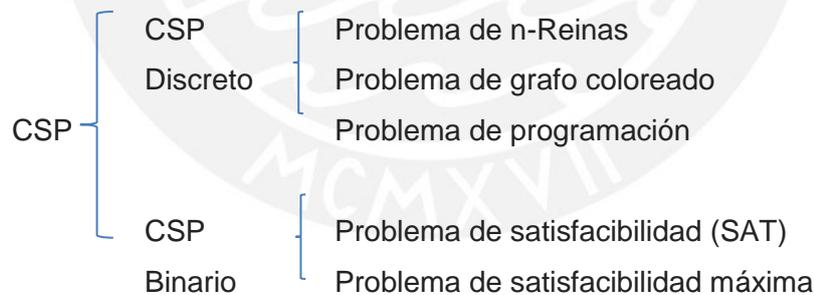
1. Agricultura tradicional: La agricultura tradicional son las prácticas agropecuarias indígenas, consecuencia de la evolución conjunta de los sistemas sociales y medioambientales autóctonos y que muestran un nivel alto de sentido ecológico expresado a través del uso intensivo de los conocimientos y recursos naturales autóctonos, que incluyen la gestión de la agrobiodiversidad mediante sistemas agropecuarios diversificados.
2. Escasez de Agua: La escasez de agua existe cuando los recursos hídricos son insuficientes para satisfacer las necesidades medias a largo plazo. La escasez se refiere a los desequilibrios hídricos que conjugan la poca disponibilidad de agua con una demanda que supera la capacidad de abastecimiento del sistema natural.
3. Seguridad del Agua: La expresión seguridad del abastecimiento de agua o seguridad del agua se refiere a la disponibilidad fiable de agua en cantidad y calidad aceptables para la producción, los medios de vida y la salud, junto con un nivel de riesgo socialmente aceptable de crisis imprevisibles relacionadas con el agua (por ejemplo, la variabilidad del clima).
4. Cultivo tolerante a la sequía: Cultivos que pueden resistir en condiciones de escasez de agua. Dada la intensificación de las sequías ocasionada por el cambio climático, la selección de cultivos tolerantes a la sequía es importante para la seguridad alimentaria.
5. Eficiencia en el uso del agua: La eficacia en el uso del agua de riego es la cantidad de biomasa o rendimiento de semillas producidas por unidad de agua de riego aplicada, que suele ser de aproximadamente 1 tonelada de materia seca por cada 100 mm de agua empleados.
6. Temporada de Cultivo: La temporada se entiende como el período de tiempo en el que se registra mayor demanda de cultivos de modo que la producción de estos se adecua más a las condiciones climatológicas. En la agricultura, está muy relacionado a las estaciones del año (verano, otoño, invierno, primavera).
7. Estabilidad del Suelo: La estabilidad del suelo depende de su resistencia al esfuerzo cortante, de su compresibilidad y de su tendencia a absorber agua. Las prácticas agrícolas que preservan la fertilidad del suelo y mantienen, o incluso aumentan, su contenido de materia orgánica pueden reducir los efectos negativos de las sequías y, además, aumentar la productividad primaria de los cultivos.

8. Agricultura responsable: La expresión “agricultura responsable” se refiere a un método integral de producción agrícola que combina las buenas prácticas agrícolas, la protección del medio ambiente, la seguridad y el bienestar de los trabajadores agropecuarios, el acceso al mercado a precios justos y vínculos más estrechos entre consumidores y productores.

1.2 Conceptos relacionados a la solución propuesta

▪ Problema de Satisfacibilidad

Los problemas NP de búsqueda tienen exponencialmente diversos estados posibles que verifican si un estado dado representa una solución. Los problemas de satisfacción restringida (CSP) son un ejemplo de este tipo. Un problema CSP consiste, básicamente, en el manejo de n variables $V_1 \dots V_n$ y el requerimiento para asignar un valor a cada variable de modo que se satisfaga las restricciones establecidas. Una asignación representa un valor para cada variable. Uno de los problemas CSP más importantes es el problema de satisfacibilidad (SAT), conocido en la literatura como Satisfiability Problem. El problema SAT consiste en una fórmula lógica proposicional de n variables y el requerimiento de encontrar un valor (verdadero o falso) para cada variable de modo que haga la fórmula verdadera [COOK, MITCHELL 1997].



Una instancia en particular del problema de satisfacibilidad es aquella que posee una fórmula booleana mediante tres componentes:

- ✓ Un conjunto de n variables: x_1, x_2, \dots, x_n
- ✓ Un conjunto de literales. Un literal es una variable ($Q = x$) o una negación de una variable ($Q = \sim x$).
- ✓ Un conjunto de m cláusulas distintas: C_1, C_2, \dots, C_m

El objetivo de un problema SAT es determinar si existe una asignación de valores verdaderos en las variables de modo que haga satisfacible una expresión CNF

(fórmula normal conjuntiva o negación de una DNF) [GU, PURDOM, FRANCO, WAH 1999]:

$$C_1 \wedge C_2 \wedge \dots \wedge C_m \quad \text{Donde: } \wedge \text{ es conector lógico.}$$

El problema satisfacibilidad es considerado el origen de una familia de problemas NP completo computacionalmente intratables. Tales problemas han sido identificados como el centro de un gran número de áreas en la teoría computacional e ingeniería. Debido a que SAT es un problema NP completo es poco probable que algún algoritmo que pretenda resolver este problema tenga un reducido tiempo del peor caso. Sin embargo, algoritmos inteligentes pueden resolver diversas fórmulas planteadas del problema SAT. Esto se refleja en el interés que se ha mostrado para el diseño de algoritmos eficientes que resuelvan la mayoría de fórmulas de este problema [SCHAEFER, 1978]

Por otro lado, existen diferentes tipos del problema de satisfacibilidad entre los que se pueden encontrar: el problema CSAT, el problema k-SAT, entre otros. Particularmente, en los problemas k-SAT cuando el valor de k es mayor o igual a 3 estos problemas se vuelven NP completos para esas situaciones.

▪ Algoritmos GRASP

Esta técnica meta heurística (Greedy Randomized Adaptative Search Procedure) fue desarrollada por T. Feo y M. Resende a finales de los años 80. Este algoritmo emplea una técnica iterativa de muestreo aleatorio donde cada una de sus iteraciones provee una solución y es la última la que se toma como resultado final [FEO, RESENDE 1995]. Los algoritmos GRASP relajan el criterio voraz (seleccionar un único elemento) de manera que trabajan con un conjunto de soluciones. Estos algoritmos funcionan en base a una lista de candidatos que viene dada por el mejor elemento, el peor elemento y una constante de relajación evitando así que la solución sea muy voraz o muy aleatoria. Diversos problemas de asignación han sido tratados con este algoritmo: el problema de asignación cuadrática, para el QAP, para el problema de asignación biquadrática, el problema de asignación de frecuencias y de asignación generalizada.

La Tabla 3 presenta la estructura general de un algoritmo GRASP que es utilizado en varios campos [TUPIA, 2009]:

Procedimiento GRASP (Instancia del Problema)

1. Leer (Instancia)
2. Mientras < no se cumpla condición de parada > hacer
 - 2.1.1 Procedimiento Construcción (S_k)
 - 2.1.2 Procedimiento Mejorar (S_k)
- Fin Mientras
3. Retornar (Mejor S_k)

Fin GRASP

Tabla 3: Estructura General de un Algoritmo GRASP

Asimismo, la Tabla 4 muestra la fase construcción de un algoritmo GRASP:

*Procedimiento GRASP Construcción (N, c, E, F, S, a)**Inicializar c, E, a* $S = \emptyset$ $N = E$

1. Mientras <no se cumpla condición de parada> hacer
 - 1.1 $RCL = \emptyset$
 - 1.2 $\beta = \text{Mejor } \{c(x) : x \in N\}$
 - 1.3 $\tau = \text{Peor } \{c(x) : x \in N\}$
 - 1.4 $RCL = \{ \forall x \in N : \beta \leq c(x) \leq \beta + \alpha(\tau - \beta) \}$
 - 1.5 $\alpha = \text{Aleatorio}(RCL)$
 - 1.6 Si $S \cup \{a\} \in F \rightarrow S \cup \{a\}$
 - 1.7 $N = N - \{a\}$
 - 1.8 Adaptar c

*Fin Mientras**Fin GRASP Construcción*

Tabla 4: Estructura General de la Fase Construcción de GRASP

▪ Algoritmos Genéticos

Los algoritmos genéticos (AG) fueron inventados por John Holland y desarrollado por el mismo Holland y colegas de la Universidad de Michigan entre 1960 y 1970. Según Holland, como lo presentó en su libro *Adaptation in Natural and Artificial Systems*, el Algoritmo Genético es una abstracción de la evolución biológica y ley de supervivencia. En este ejemplar muestra un marco del uso de este método [HOLLAND, 1975].

Básicamente, un algoritmo genético empieza su desarrollo utilizando una población inicial de individuos. A partir de la aplicación y evaluación, a la población inicial, de diferentes operaciones naturales se obtienen nuevas poblaciones que son otra vez soluciones al problema pero en diferentes generaciones.

Las ventajas más importantes de los algoritmos genéticos para los problemas de optimización son la búsqueda de soluciones sin considerar funcionamientos internos y específicos del problema, efectividad al momento de realizar búsquedas globales por medio de sus operadores genéticos y flexibilidad para tratar con el dominio.

Por otro lado, los componente principales que poseen los AG son el gen (mínima unidad de representar la instancia de un estado), cromosoma (conjunto de elementos del problema que representa una solución al mismo), población inicial (conformada por un conjunto de individuos, los cuales son posibles soluciones construidas por lo general de forma aleatoria), operadores genéticos (mecanismos que permiten relacionar los cromosomas y generar nuevos individuos en una nueva generación mediante su reproducción, mutación e inversión), aberración (solución inválida al problema que no cumple con los requerimientos principales que el problema impone a una solución y por lo tanto no se toma en cuenta como solución) y la función fitness (función aplicada al cromosoma de modo que se mida la calidad de la solución).

La forma básica de un algoritmo genético se muestra en la Tabla 5 [TUPIA, 2009]:

Procedimiento Algoritmo Genético

1. *Generar Población Inicial P_0 .*
2. *$P_{Anterior} = Población\ Inicial$.*
3. *Mientras < no se cumpla condición de parada > hacer*
 - 3.1 *Aplicar Operador Casamiento (tasa, $P_{Anterior}$, P_{Nueva}).*
 - 3.2 *Aplicar Operador Mutación (tasa, $P_{Anterior}$, P_{Nueva}).*
 - 3.3 *Aplicar Operador Inversión (tasa, $P_{Anterior}$, P_{Nueva}).*
 - 3.4 *Eliminar Aberraciones ($P_{Anterior}$, P_{Nueva}).*
 - 3.5 *Aplicar Mecanismo Selección Aleatoria (P_{Nueva} , $f_{fitness}$).*
 - 3.6 *Medir Convergencia (P_{Nueva} , mejor_individuo).*
 - 3.7 *$P_{Anterior} = P_{Nueva}$.*
 - 3.8 *$P_{Nueva} = \varnothing$.*

Fin Mientras 3.

Fin Algoritmo Genético.

Tabla 5: Estructura General de un Algoritmo Genético

1.3 Conclusión

Por lo tanto, para poder realizar una distribución adecuada del agua en el riego de cultivos es necesario tener conocimientos y familiarizarse con los términos usados en esta área, además de las metodologías algorítmicas que también son de importancia.

1.4 Marco regulatorio / legal

Para que la actividad agrícola y el uso del recurso hídrico en nuestro país puedan darse sin ningún contratiempo tienen que cumplir una serie de normas establecidas por la Autoridad Nacional del Agua (ANA) y que han sido previamente aprobadas por el Poder Legislativo Peruano. La ley N° 29338 de Recursos Hídricos entró en vigencia el primero de abril del 2009.

A continuación, se nombran los artículos de la ley, correspondiente al uso del agua en la actividad agrícola que se desarrolla en este proyecto.

- **Decreto Supremo N° 001-2010-AG, Ley de Recursos Hídricos. Artículo 4.-**

Toda administración del recurso hídrico y sus aspectos asociados son de manera exclusiva de la Autoridad Nacional del Agua. Además, de la gestión de los recursos hídricos en sus fuentes naturales y bienes asociados [ANA, 2013].

Este artículo hace referencia a la principal autoridad del recurso hídrico, además que faculta toda responsabilidad para administrar y gestionar el uso del mismo.

- **Decreto Supremo N° 001-2010-AG, Ley de Recursos Hídricos. Artículo 21.-**

Las acciones derivadas de funciones sobre la gestión multisectorial de los recursos hídricos son funciones de la Autoridad Nacional del Agua. Esta encargada de administrar las fuentes de agua en cantidad, calidad y oportunidad [ANA, 2103].

Las acciones, se puede notar, que se realicen sobre el recurso hídrico están normadas por esta entidad reguladora en aspectos de cantidad, calidad, entre otros.

- **Decreto Supremo N° 001-2010-AG, Ley de Recursos Hídricos. Artículo 55.-**

El uso del agua está condicionado bajo diversos aspectos como las necesidades reales del objeto y fluctuaciones de las disponibilidades del recurso. Además, su

uso debe ser eficiente sin afectar la calidad y destinada para el uso primario, poblacional y productivo, con el mismo orden de prioridad como fue mencionado [ANA, 2013].

La importancia de reconocer la eficiencia del manejo del agua así como las prioridades al destinar este recurso en el entorno de uso que se esté desarrollando.

De este modo, se pueden ver algunos de los artículos que contiene esta ley de regulación de recursos hídricos en el Perú. Además, se puede notar las formas de distribución del recurso que se realiza en la población así como las prioridades del consumo del mismo.

2 Estado del arte

La revisión de la literatura del problema de satisficibilidad adaptada a la distribución del recurso hídrico es necesario realizarla para poder conocer las diversas formas de resolución y soluciones aproximadas que se han desarrollado y se mantienen.

2.1 Método usado en la revisión del estado del arte

Para la búsqueda de toda la información relativa al problema que se centra este proyecto se ha realizado diferentes formas de evaluación en cuanto a las fuentes de dónde provienen. En primer lugar, se establecieron las fuentes electrónicas y digitales de búsqueda de carácter académico que proporciona la Internet. Para este caso se hizo uso de fuentes académicas como: ACM Digital Library, IEEE Xplore, Science Direct y Google Scholar.

Luego se procedió a realizar búsquedas con términos y frases claves que están relacionadas a nuestro problema a tratar. Algunos de los términos que fueron usados para la búsqueda son *Satisfiability Problem*, *Water Allocation*, *Agriculture and Water Resources*, *Assignment Water on Irrigation*.

Las fechas en que estas búsquedas fueron realizadas oscilan entre mediados del mes de Agosto (comienzo de clases PUCP 2013-2) y Septiembre del 2013. A partir de esto se realizó una rápida y superficial redacción a los recursos encontrados y así determinar si tal información es de utilidad y de beneficio para el tema del proyecto o

no. Además de considerar la confiabilidad de la fuente, se tomó importancia en el año de publicación y la relevancia del autor en el tema.

Además, es importante señalar que se han ido actualizando los diferentes contenidos que se han visto necesario modificar de modo que la presente revisión contenga información lo más actual posible así como datos que se consideraron luego como también importantes.

Finalmente, los documentos seleccionados como relevantes se organizaron de acuerdo al año de publicación y al tema tratado por el autor para luego poder acceder a ellos de manera rápida.

2.2 Formas exactas de resolver el problema

En esta sección, se muestran las formas exactas de solucionar el problema de satisfacibilidad definido anteriormente.

▪ Multi-Criteria Decision making

Criterios múltiples de toma de decisiones (MCDM) se pueden definir como el estudio de métodos y procedimientos mediante los cuales se consideran varios criterios que dificultan la distribución de recursos. MCDM se refiere a la toma de decisiones en presencia de múltiples criterios, por lo general, en conflicto. El rápido desarrollo de la tecnología informática en los últimos años ha hecho posible realizar un análisis sistemático de estos problemas, además de considerarlos más importantes [INTERNATIONAL SOCIETY ON MULTIPLE CRITERIA DECISION, 2013].

▪ Investigación Operativa

La investigación de operaciones (IO) es la disciplina que se centra en la aplicación de tecnologías de la información para la toma de decisiones. El objetivo de la IO es proporcionar bases racionales de decisión buscando comprender y estructurar situaciones complejas, predecir el comportamiento y mejorar el rendimiento de un sistema. Gran parte del trabajo se lleva a cabo mediante el uso de técnicas analíticas como por ejemplo la programación lineal (PL), la cual permite resolver problemas de optimización. El algoritmo Simplex es utilizado para tratar estos problemas PL. El modelo de programación lineal minimiza o maximiza una función objetivo sujeta a una serie de restricciones lineales [WINSTON, 2005].

▪ **Algoritmo de Davis-Putnam**

En este método, definido en 1960 por Davis y Putnam, las variables de una fórmula se eliminan una por una de manera que en cada paso se van generando posibles resolventes (basados en la elección de una variable) para luego eliminar todas las cláusulas que mencionan a aquella variable. En cada paso se genera un subproblema con unas pocas variables menos pero, posiblemente, con más cláusulas. Dos métodos heurísticos son utilizados durante este proceso con la finalidad de reducir el espacio de búsqueda: la regla del literal puro y la cláusula unitaria. Por otro lado, nuevas formas para solucionar los problemas SAT parten de este algoritmo DP con algunas modificaciones. Entre los más conocidos se pueden encontrar: DPL, DPLH, DPL_RN [VILLAMIZAR, 2004].

▪ **Algoritmo Greedy**

Los algoritmos voraces han sido utilizados para la resolución de problemas de satisfacibilidad, particularmente, en donde el número de cláusulas CNF no satisfacible es formulada como el valor de una función objetivo, transformando la fórmula SAT en un problema de minimización discreta y no restringida para la función objetivo. En este procedimiento, se parte de una solución aleatoria de modo que se compruebe si esta satisface la fórmula. De no ser así, se cambia la solución para que se maximice el número de cláusulas resueltas. El proceso es repetido hasta alcanzar un máximo de intentos fijados, inicialmente. Por otro lado, existen otros algoritmos que siguen el procedimiento mencionado: GSAT, GWSAT, WALKSAT [VILLAMIZAR, 2004].

2.3 Formas aproximadas de resolver el problema

En esta sección, se muestran algunas formas aproximadas de resolver el problema de satisfacibilidad.

▪ **GRASP un algoritmo de búsqueda para la satisfacibilidad proposicional**

Para la resolución de los problemas de satisfacibilidad que existen se presenta un algoritmo genérico GRASP (*Generic Search Algorithm for the Satisfiability Problem*), el cual es un marco integrado para la algorítmica de satisfacibilidad que unifica técnicas de búsqueda de poda. Se basa en la inevitabilidad de los conflictos durante la búsqueda y su característica principal es el aumento de la búsqueda de retroceso con un procedimiento de análisis de conflictos. Además, puede anticipar a la ocurrencia de

conflictos similares. Por último, la contabilidad de cadenas de causalidad permite identificar asignaciones necesarias para la solución [MARQUES, SAKALLAH, 1999].

- **GRASP con rutas de vinculación para el problema MAX-SAT ponderado**

Un algoritmo GRASP (*Greedy Randomized Adaptive Search Procedure*) con rutas de vinculación para la búsqueda de soluciones del problema SAT ha sido desarrollado. Los algoritmos GRASP muestran una eficacia considerable para la resolución de los casos MAX-SAT. La ruta de vinculación es un procedimiento utilizado para intensificar la búsqueda en torno a soluciones de buena calidad. La comparación experimental del GRASP puro (sin ruta de vinculación) y el GRASP con la ruta de vinculación ilustra la eficacia de la ruta a vincular en la disminución del tiempo medio necesario para encontrar una buena solución [FESTA, PARDALOS, PITSOULIS, RESENDE 2007].

- **Una red neuronal híbrida de Hopfield para resolver problemas de satisfacibilidad.**

Una nueva forma de resolver el problema de satisfacibilidad es realizada mediante dos componentes algorítmicos integrados. Por un lado, se propone un nuevo algoritmo imperialista competitivo (ICA). ICA expresa las fórmulas del problema SAT mediante multiplicadores LaGrange, los cuales representan pesos de las cláusulas de satisfacibilidad. Por otro lado, una red híbrida de Hopfield (HNN) integrado con un algoritmo ICA. La red neuronal HNN maneja las restricciones del problema mientras que el algoritmo ICA busca soluciones de alta calidad y mínimo costo. Estos algoritmos se utilizan para la solución de multiplicadores LaGrange [ABDECHIRI, REZA 2012].

- **Optimización de Colonia de Hormigas con una función fitness adaptativa para la revisión de satisfacibilidad.**

La Optimización de Colonia de Hormigas (ACO) es una meta heurística inspirada en el comportamiento del forrajeo de hormigas. De este modo, se propone una herramienta MaxMin-SAT, el cual es una alternativa de solución a problemas SAT. MaxMin-SAT es un algoritmo ACO que implementa una función adaptativa de fitness (técnica que se utiliza en los algoritmos genéticos para escapar de óptimos locales). Para demostrar la eficacia de esta técnica, tres funciones de la aptitud de adaptación diferentes son verificadas: la adaptación gradual de pesas, la función de refinación de soluciones y una mezcla de los dos anteriores funciones definidas [VILLAGRA, BARÁN 2007].

2.4 Productos comerciales para resolver el problema

Para resolver el problema de satisfacibilidad adaptada a la distribución del recurso hídrico existen algunos programas comerciales:

- **WINQSB**

WINQSB es un sistema interactivo de ayuda a la toma de decisiones que contiene herramientas muy útiles para resolver distintos tipos de problemas en el campo de la investigación operativa [WINQSB, 2013]. Entre estos se destacan la programación lineal (*LP*), programación entera (*IP*), modelado de redes (*NET*), entre otros. Además, se puede encontrar programas específicos para resolver el problema de asignación y satisfacibilidad [WINQSB, 2013]. La interfaz se puede apreciar en la Figura 4.

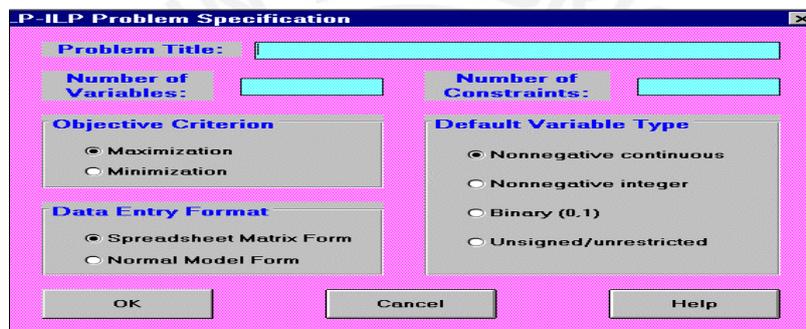


Figura 4: Interfaz del sistema WINQSB

- **SAS/OR**

Las soluciones desarrolladas por el software SAS/OR proveen herramientas que permiten modelar, resolver y comunicar las mejores solución a la planeación de problemas complejos con rapidez y eficacia. Este software resuelve problemas de planificación y desafíos de negocios tales como la asignación de recursos y administración [SAS/OR, 2013]. La interfaz de este sistema se puede apreciar en la Figura 5.

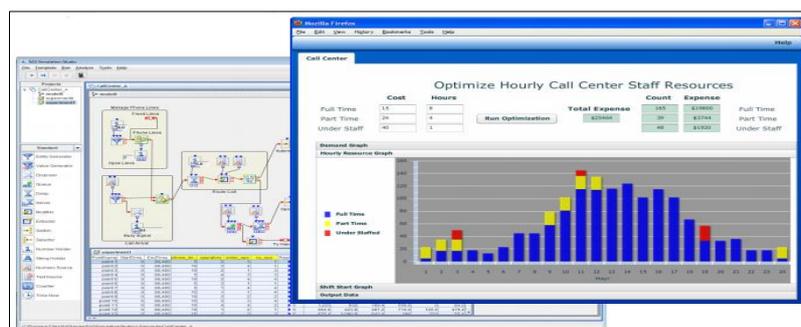


Figura 5: Interfaz del software SAS/OR

▪ SMT SOLVER

Los solucionadores SMT permiten resolver problemas de decisiones para fórmulas lógicas respecto a combinaciones posibles de valores. Una instancia SMT es una generalización de una instancia booleana del problema SAT donde las variables son reemplazadas por predicados. Entre los SMT más conocidos se encuentra Alt-Ergo, el cual permite resolver fórmulas lógicas de tamaño fijo, utiliza vectores de bits puros y con matrices para las expresiones de evaluación, emplea un procedimiento sofisticado para traducir expresiones en fórmulas booleanas [SMT, 2012]. La interfaz de este sistema se puede apreciar en la Figura 6.

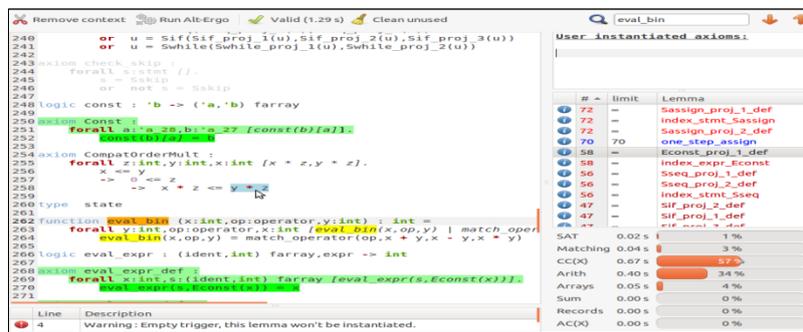


Figura 6: Interfaz de software ALT-ERGO

▪ IMS (Solución de gestión ICC - Motorola)

El software IMS es una herramienta de gestión de riego única y diseñada para ser usada con el software Motorola ICC. IMS proporciona funcionalidades de gestión sin precedentes desarrolladas para adaptarse a las exigentes aplicaciones de control de riego y ocuparse de los requisitos exactos de riego. El IMS permite cambios dinámicos, precisos y automáticos de cantidades de agua y fertilizantes para riego, en función de los parámetros profesionales y algoritmos sofisticados. La flexibilidad de este sistema permite realizar actividades simultáneas [IMS, 2013]. La interfaz de este sistema se puede apreciar en la Figura 7.

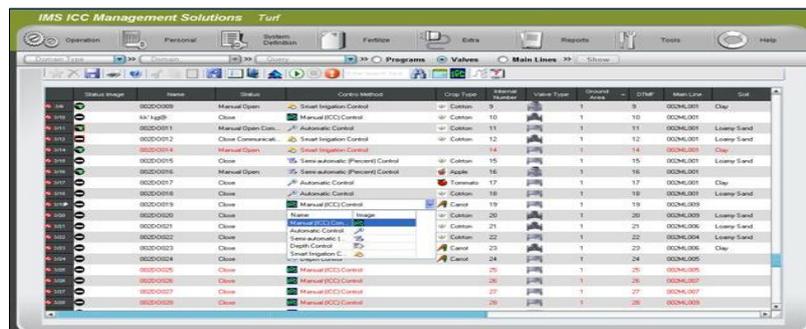


Figura 7: Interfaz del software IMS

2.5 Productos no comerciales para resolver el problema

En esta sección, se muestran algunos aportes de investigadores que tienen como objetivo solucionar el problema en cuestión para diferentes contextos.

▪ **Decomposing SAT problems into connected components**

Esta investigación pretende establecer que muchas instancias del problema de satisfacibilidad pueden ser descompuestos en componentes conectados ya sean inicialmente después del procesamiento o durante la fase de solución cuando las nuevas unidades de cláusulas de conflicto son aprendidas. Este punto permite que los componentes puedan ser resueltos individualmente. La investigación presenta una técnica que permite manejar estos componentes definidos mediante un GRASP como si fuese un solucionador de satisfacibilidad (SAT solver) sin requerir muchos cambios en su adaptación respectiva [BIERE, SINZ 2006].

▪ **Research on Allocation of Water Resources Sustainable Based on methodology GO**

Un método de optimización global para resolver un problema de asignación de recursos hídricos y utilización sostenible de fuentes de agua es presentado en esta sección. El objetivo es determinar el diseño óptimo del proceso y funcionamiento de una producción de agua determinada. Aunque el modelo se derivó de una hipótesis simplificada se consideran los aspectos más importantes de un proceso natural. El modelo tiene en cuenta las limitaciones de demanda de recursos y suministro del agua así como sus restricciones de equilibrio del elemento hídrico [LONGZHU, SHIZHANG, SHENGMIN 2009].

▪ **Optifert Project**

Optifert es un proyecto de irrigación inteligente que permitirá a las personas dedicadas a la agricultura realizar un mejor manejo de recursos naturales de modo que se ahorre agua y fertilizantes necesarios para la actividad agrícola. Los investigadores de este proyecto pretenden mezclar fertilizantes y agua, para luego inyectarlo en el prototipo de un sistema de fertirrigación que riega los campos con un monitoreo constante y en tiempo real de las condiciones atmosféricas. Además, conoce las necesidades del suelo y plantas en las diferentes etapas de los cultivos de modo que la distribución se realice de la forma más real y cercana a la agricultura en sí [OPTIFERT, 2013].

2.6 Cuadro resumen

A continuación en la Tabla 6, se muestra un breve resumen de la información recopilada de productos comerciales e investigaciones.

	Características	Prod.I: WINQSB	Prod II:SAS/OR	Prod.III: SMT SOLVER	Prod. IV: IMS	T E S I S
PRODUCTOS COMERCIALES	Manejo de Múltiples Criterios	X	X	X	X	
	Manejo de Problema SAT	X	-	X	-	
	Manejo de Algoritmos Integrados	-	X	-	X	
	Características	Invest I: DESC-SAT	Invest.II: GO	Invest. III: QuBE	Invest.IV: OPTIFERT	
INVESTIGACIONES ACADÉMICAS	Orientado a Problema SAT	X	-	X	-	X
	Orientado a Distribución Agua	-	X	-	X	X
	Orientado a Formulas CNF	X	-	X	-	-

Tabla 6: Cuadro Comparativo de soluciones para el problema

CAPÍTULO 3: DISEÑO ESTRUCTURAL

1 Estructuras de datos

Las estructuras utilizadas en el proyecto se pueden definir de acuerdo a su importancia y uso en el desarrollo. Por un lado, se posee la estructura principal del algoritmo genético, el cromosoma, mientras que por otro lado, se encuentran estructuras auxiliares, las cuales son empleadas para realizar cálculos, almacenamientos y la simulación requerida. Si bien estas estructuras han sido definidas principalmente para el soporte del algoritmo genético también son de utilidad para el desarrollo del algoritmo GRASP. Asimismo, cada método algorítmico maneja sus propias estructuras internas, las cuales son definidas en su diseño respectivo.

1.1 Cromosoma

▪ Definición de la estructura

Según los conceptos mencionados sobre algoritmos genéticos, es importante recordar que cada individuo de la población representa una solución al problema que se intenta resolver. Además, cada individuo a través de su cromosoma está compuesto por varios componentes denominados genes.

En este proyecto, el problema que se trata de resolver es la distribución de agua mediante la selección de terrenos. Adicionalmente, es importante mencionar que cada terreno posee características propias. Por lo tanto, cada terreno estará representado por un gen del cromosoma de modo que indique si el terreno ha sido seleccionado o no para ser regado. De esta manera, la estructura principal está representada mediante cadenas binarias de modo que indiquen la selección de los terrenos. Esta representación posee como longitud el mismo valor de cantidad total de terrenos.

A continuación en la Figura 8, se muestra un ejemplo de los terrenos con los que se cuentan y requieren ser regados:



Figura 8: Ejemplo de conjunto de terrenos

En el ejemplo presentado se requiere regar cierta cantidad de terrenos sin alterar o exceder la cantidad del recurso disponible, en este caso la cantidad de agua que se ofrece. El primer gen podría estar definido como se muestra en la Figura 9:



Figura 9: Ejemplo de Gen

Este gen representa que el terreno A ha sido seleccionado para ser regado debido a que cumple ciertas condiciones de beneficio y no altera o sobrepasa la disponibilidad del recurso hídrico, que es lo más importante.

En el momento que se haya utilizado por completo la disponibilidad del recurso hídrico, se agrupan los genes correspondientes a cada terreno formando de esta manera el cromosoma que representa los estados de selección de los terrenos manejados. Ahora los terrenos pueden ser vistos como se muestra en la Figura 10:

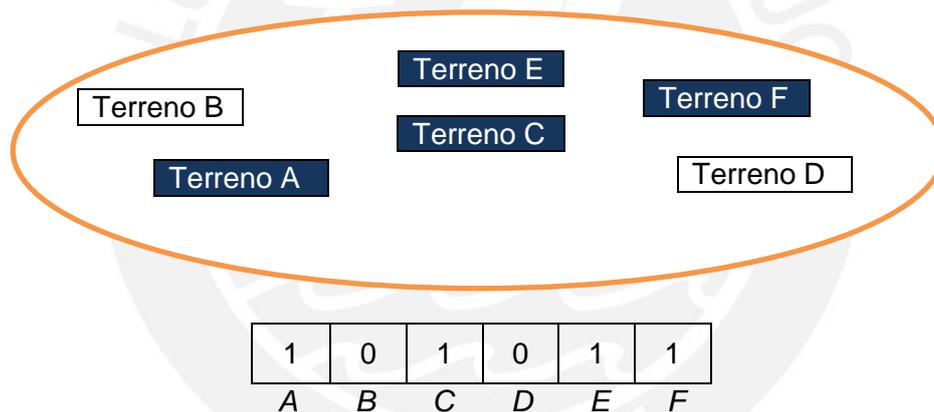


Figura 10: Ejemplo de Cromosoma

La imagen anterior muestra un conjunto de genes que conforman un cromosoma. Este cromosoma representa el estado de selección de los terrenos. A nivel de estructura de datos, este cromosoma es representado a través de un vector de números enteros. En términos del cromosoma, se puede interpretar de la siguiente manera: cada gen representa el estado de un terreno, si este es seleccionado (valor 1) o no (valor 0).

Ahora, para entender un poco más cómo es empleada la estructura cromosómica mencionada se procede a contextualizar su uso. Debido a que se requiere regar una cierta cantidad de terrenos utilizando una cantidad limitada de agua se comienza buscando los terrenos que aporten una utilidad considerable, verificando que sus requerimientos hídricos no superen la cantidad de agua disponible, para luego ser seleccionados.

▪ Representación

Luego de definir la estructura que utiliza el cromosoma es necesario señalar la representación de esta estructura en la herramienta informática que se desarrolla en este proyecto. Para esto se ha realizado la creación de una clase Gen y una clase Individuo.

A continuación en la Tabla 7, se muestra como se representa la clase Gen para este componente:

Gen		
Campo	Descripción	Estructura de Datos
<i>Id</i>	Identificador para representar cada gen que se encuentra en la estructura.	No es utilizado en el Proceso.
<i>Estado</i>	Contiene el estado de selección del terreno asociado al Gen, si ha sido seleccionado o no.	Byte
<i>Fracción</i>	Indica la parte total o parcial que ha sido seleccionado del terreno respecto al requerimiento de agua.	Double

Tabla 7: Representación clase Gen

Luego en la Tabla 8 se muestra la clase Individuo, el cual contiene el cromosoma respectivo:

Individuo		
Campo	Descripción	Estructura de Datos
<i>Id</i>	Identificador para representar cada individuo que se encuentra en la estructura.	No es utilizado en el Proceso.
<i>Cromosoma</i>	Contiene, a través de sus genes, los terrenos en total que se requieren regar en el problema.	Lista de Genes

Tabla 8: Representación clase Individuo

1.2 Estructura de selección

▪ Definición de la estructura

Una de las estructuras auxiliares más importante que se utiliza es la estructura de selección. Esta estructura sirve para poder almacenar solo los terrenos que han sido seleccionados. Consiste en una lista conteniendo todos los terrenos seleccionados sin considerar los que no han sido seleccionados, como se muestra en la Figura 11.

A	C	E	F
1	2	3	4

Figura 11: Ejemplo de una estructura selección

En esta estructura se puede ver la secuencia de terrenos que solo han sido seleccionados. La estructura almacena los índices de dichos terrenos para su rápido acceso en alguna consulta futura o para la adaptación de la solución brindada por el algoritmo GRASP en la estructura cromosómica del algoritmo Genético.

▪ Representación

Para la representación de esta estructura auxiliar, se utiliza una clase que pueda almacenar los índices requeridos. Para esto se ha realizado la creación de una clase Selección como se muestra en la Tabla 9:

Selección		
Campo	Descripción	Estructura de Datos
<i>Índices</i>	Contiene los índices de los terrenos que han sido seleccionados luego de la ejecución respectiva.	Lista de Byte

Tabla 9: Representación clase Selección

1.3 Estructura de requerimientos

▪ Definición de la estructura

Para poder manejar la información de terrenos se crea una serie de clases que soporten estos datos de modo que se obtenga un rápido acceso. Es así que, en primer lugar, se posee la información de cada cultivo (costo, beneficio y requerimiento hídrico) que se siembra en un terreno específico, como se muestra en la Figura 12:

10
12
15

Cultivo "A"

Figura 12: Ejemplo de Cultivo

En segundo lugar, se maneja la información propia del terreno (área, distancia y rendimiento) en un nivel superior a la información de su cultivo, como se muestra en la Figura 13:

30
15
0.3
10
12
15

Terreno "A"

Figura 13: Ejemplo de Terreno

De este modo, cada terreno posee información relacionada a sus características (Clase Terreno) mientras que la información relacionada a los requerimientos del cultivo se maneja de manera interna (Clase Cultivo dentro de Clase Terreno).

▪ **Representación**

Luego de definir la estructura que maneja la información de los terrenos es necesario señalar su representación en la herramienta informática. El manejo de la información del cultivo y terreno se realiza mediante la creación de una clase Cultivo y clase Terreno, respectivamente.

A continuación en la Tabla 10, se muestra la representación de la clase Cultivo:

Cultivo		
Campo	Descripción	Estructura de Datos
<i>Id</i>	Identificador para cada cultivo asociado a un terreno respectivo que se encuentra en la estructura.	No es utilizado en el Proceso.
<i>Costo</i>	Contiene el costo de producción que demanda sembrar el cultivo en el terreno específico en soles por hectárea como unidad.	Double

Cultivo		
Campo	Descripción	Estructura de Datos
<i>Beneficio</i>	Contiene la rentabilidad que ofrece el cultivo en cosecha en soles por hectárea como unidad.	Double
<i>Agua</i>	Contiene el requerimiento de agua que necesita el cultivo en metros cúbicos por hectárea.	Integer

Tabla 10: Representación clase Cultivo

Ahora, recién se puede consultar la información del terreno. Para esto, se utiliza una clase Terreno que maneja esta información, tal como se muestra en la Tabla 11:

Terreno		
Campo	Descripción	Estructura de Datos
<i>Id</i>	Identificador para cada terreno generado en la zona generada.	No es utilizado en el Proceso.
<i>Área</i>	Contiene el área del terreno en hectáreas como unidad.	Integer
<i>Rendimiento</i>	Contiene el rendimiento del terreno. No posee unidades.	Double
<i>Cultivo</i>	Contiene la información del cultivo asociado al terreno accedido.	Cultivo

Tabla 11: Representación clase Terreno

Sin embargo, si bien de esta manera se puede manejar los datos iniciales es posible que se necesite una nueva estructura que permita almacenar las diferentes generaciones de los datos de terrenos. Es así que se crea la clase Zona, tal como se muestra en la Tabla 12:

Zona		
Campo	Descripción	Estructura de Datos
<i>Id</i>	Identificador para cada generación de datos obtenidos	No es utilizado en el Proceso.
<i>Terrenos</i>	Contiene la información de los terrenos así como el cultivo propio.	Lista de Terrenos

Tabla 12: Representación clase Zona

De este modo, se maneja la clase Zona, Terreno y Cultivo. La descripción de los campos utilizada en cada clase ha sido explicada en los puntos anteriores. Debido a esto, su información es accedida de la siguiente manera: $R[i]$ representa el terreno i , $R[i].T$ el área, $R[i].B$ el beneficio, $R[i].C$ el costo, $R[i].K$ el rendimiento, $R[i].D$ la distancia y $R[i].A$ el requerimiento hídrico del terreno i .

1.4 Población

- **Definición de la estructura**

Luego de haber creado la clase Gen e Individuo es necesaria la creación de una nueva clase que contenga estos elementos. Para que el algoritmo genético pueda manejar los diversos individuos se crea una clase Población. Esta clase evoluciona en cada iteración.

- **Representación**

Después de definir la estructura que utiliza esta población es necesario señalar la representación de esta estructura. Para esto, se ha realizado la creación de una clase población como se muestra en la Tabla 13:

Población		
Campo	Descripción	Estructura de Datos
<i>id</i>	Identificador para representar cada población que ha sido generada en la etapa de evolución.	No es utilizado en el Proceso.
<i>individuos</i>	Contiene la información de los individuos o soluciones para una población.	Lista de Individuos

Tabla 13: Representación clase Población

1.5 Estructura de Fitness

- **Definición de la estructura**

Esta estructura contendrá los valores de bondad de cada individuo. Es decir, se evalúa la función fitness de cada individuo y se almacena para su uso posterior. Por ello, se utiliza una lista de números reales que representa el valor fitness obtenido por cada

individuo e irá variando según la evolución de la población, como se muestra en la Figura 14:

100.20	150.22	130.10	189.21
1	2	3	4

Figura 14: Ejemplo de estructura fitness

En esta estructura se puede ver la secuencia de los valores fitness de los terrenos.

▪ Representación

Para la representación de esta estructura auxiliar, se utiliza una clase que almacene los valores de la función objetivo, la clase *Fitness*, tal como se muestra en la Tabla 14:

Fitness		
Campo	Descripción	Estructura de Datos
<i>Fitness</i>	Contiene los valores calculados de la función fitness para los terrenos seleccionados luego de la ejecución.	Lista de Double

Tabla 14: Representación clase *Fitness*

1.6 Resumen de estructuras de datos

A continuación en la Tabla 15, se presenta un cuadro resumen de las estructuras junto con su representación simplificada que es utilizada en el diseño de pseudocódigos:

Estructura	Descripción	Símbolo
Cromosoma	Permitirá almacenar los estados de selección de los terrenos además de su importancia ya mencionado.	C
Estructura de Selección	Almacena la selección de terrenos elegidos y sirve como soporte para el algoritmo genético y GRASP.	S
Estructura de Requerimientos	Permite conocer los requerimientos de cada terreno, lo cual es esencial para la ejecución de ambos algoritmos.	R
Población	Almacena el conjunto de individuos de una instancia del problema para el algoritmo Genético.	P
Estructura de Fitness	Funciona a la par con <i>P</i> pues almacena el fitness de cada individuo que pertenece a la población.	F

Tabla 15: Resumen de estructuras de datos

2 Función objetivo

Para el desarrollo del algoritmo genético propuesto es necesario evaluar la suficiencia de los resultados obtenidos de modo que se establezca la solución más óptima.

2.1 Definición

El presente proyecto pretende ofrecer un valor representativo de la utilidad que generaría el riego de los terrenos sin exceder la disponibilidad de agua. Debido a que los algoritmos pretenden producir el mejor resultado posible dentro un conjunto de candidatos, se requiere una función objetivo para esto. Esta función establece un criterio de selección considerando que cada terreno posee factores naturales propios.

De este modo, se establece una función objetivo que permita seleccionar los terrenos que sean más rentables para ser regados. Cabe resaltar que el objetivo es maximizar la expresión de la función objetivo de modo que se obtenga el mayor valor posible. La expresión de esta función es definida a continuación:

$$Max \sum_{i=1}^N \frac{T_i * B_i * K_i}{C_i * D_i}$$

De la expresión anterior, se especifican diversas variables siendo T_i el área del terreno asociado al cultivo i , B_i el beneficio del cultivo i , C_i el costo de producción del cultivo i , K_i el rendimiento asociado al cultivo i y D_i la distancia del terreno i al reservorio de agua. La información representativa de estas variables se encuentra disponible en una estructura auxiliar de requerimientos (R) definida anteriormente.

Como se ha podido observar, la expresión definida pretende ser maximizada. Esto se debe a que se requiere:

- Aumentar el beneficio B_i
- Reducir el costo C_i
- Aumentar el rendimiento K_i
- Reducir la distancia D_i
- Aumentar el área del terreno T_i

Luego de ver que se pretende maximizar la expresión anterior es importante explicar la necesidad de aumentar y reducir los valores de la función objetivo para cada terreno establecido al momento de aplicar el criterio de selección.

Debido a que se pretende seleccionar los terrenos más rentables posibles, en primer lugar, es adecuado tratar de elegir los terrenos con alto beneficio y bajo costo, es por eso que se intenta aumentar el beneficio B_i y reducir el costo C_i , respectivamente.

Luego, es necesario elegir los terrenos con mayor rendimiento posible. De este modo, se intenta aumentar el rendimiento K_i ofrecido por el terreno i . Por otro lado, es adecuado seleccionar los terrenos que se encuentren más cerca del reservorio, es decir, con menor distancia. Así, se requiere reducir la distancia D_i del cultivo i .

Por último, debido a que los terrenos poseen diferentes dimensiones no basta con evaluar su costo y beneficio. Aparte de esto, se debe elegir los terrenos con mayor área debido a que podrían generar mayor utilidad y aparte su razón de costo-beneficio.

Al momento de la elección de los terrenos se tiene en cuenta la restricción de la disponibilidad de agua (DA) que se maneja y que está expresada de la siguiente forma:

$$\text{Sujeto a } \sum_{i=1}^N A_i * T_i \leq DA$$

2.2 Detalle de variables

Luego de definir la función objetivo es importante explicar las variables que han sido consideradas de modo que se obtenga una comprensión adecuada de los mismos.

T_i : Indica la cantidad de tierra asignada o que se dispone en el terreno i asociado a su cultivo respectivo.

B_i : Indica el beneficio de producción que ofrece el terreno i mediante la siembra de su cultivo asociado. Particularmente y a diferencia de las otras variables, el valor del beneficio es obtenido a partir de una serie de cálculos realizados (Ver capítulo 6).

C_i : Indica el costo de producción que demanda realizar el sembrado del cultivo i en el terreno específico.

K_i : Indica el rendimiento del cultivo i en un terreno respecto a la producción que se genera por la adecuada cantidad de agua aplicada.

D_i : Indica la distancia hacia el reservorio de agua del terreno i mediante la siembra de su cultivo.

A_i : Indica el requerimiento de agua que se necesita para el sembrado y riego del cultivo i en el terreno.

CAPÍTULO 4: MODELO COMPUTACIONAL GRASP

1 Algoritmo GRASP

En la siguiente sección, se realiza la presentación del algoritmo GRASP en su fase construcción. Como se mencionó en el alcance, el presente proyecto está orientado a la implementación de un algoritmo genético, sin embargo, se requiere la implementación adicional de un algoritmo que permita generar una población inicial y ser usado en una experimentación numérica. Por lo tanto, se elige el algoritmo GRASP debido a que, según la literatura sobre la resolución del problema, es uno de los algoritmos más empleados en este campo además de su fácil comprensión e implementación.

1.1 Variables y estructuras

Para el diseño del algoritmo GRASP es importante señalar las variables y estructuras que son utilizadas. Si bien anteriormente se definió las estructuras principales que usan, existen otras estructuras adicionales de uso interno propias del algoritmo.

A continuación en la Tabla 16, se presentan las estructuras propias del algoritmo GRASP:

Símbolo	Descripción
B	Esta variable contiene la bondad calculada que ofrece cada uno de los terrenos (y su cultivo asociado) a evaluar.
DA	Esta variable indica la disponibilidad de agua inicial con la que cuenta el reservorio establecido.
CJ	Esta variable almacena las diferentes soluciones que se obtienen de la ejecución del algoritmo GRASP. La cantidad de soluciones en este conjunto es igual al tamaño de la población del algoritmo Genético.
NI	Esta variable hace referencia al número de iteraciones que se requiere ejecutar el algoritmo GRASP.
SL	Esta variable indica una solución obtenida por el algoritmo. En términos del problema, esta variable indica los estados de selección de los terrenos.
LC	Esta variable contiene la estructura utilizada para representar la lista de posibles candidatos.

Símbolo	Descripción
LCC	Esta variable hace referencia a una copia de la lista de candidatos LC pero su contenido es modificado cada vez por los parámetros del algoritmo.
E	Esta variable representa un terreno seleccionado aleatoriamente de la estructura anterior (LCC). Este elemento es un componente de la solución que se va formando para una determinada iteración del algoritmo.
Alfa	Esta variable indica una constante de relajación propia del algoritmo GRASP. Mediante esta variable se logra una adecuada ejecución del algoritmo, ni tan voraz ni tan aleatorio.

Tabla 16: Estructuras propias de Algoritmo GRASP

1.2 Pseudocódigo del algoritmo GRASP

En esta sección, se presenta el pseudocódigo del algoritmo de experimentación GRASP utilizado para crear la población inicial (PI). Adicionalmente, como ya se mencionó este algoritmo también es utilizado en la experimentación numérica, como se muestra en la Tabla 17:

Inicio Algoritmo GRASP (B, DA, NI, alfa)	
1. :	LC = Ordenar_Lista(B)
2. :	CJ = []
3. :	contador = 0
4. :	Mientras (NI > contador) hacer
5. :	LCC = LC
6. :	SL = Construir_Solución (LCC, DA, alfa)
7. :	Agregar_Solución (CJ, SL)
8. :	contador = contador + 1
9. :	Fin Mientras
Fin Algoritmo GRASP	

Tabla 17: Pseudocódigo del Algoritmo GRASP

De la misma manera, se muestra el pseudocódigo del procedimiento que permite construir la solución como se indica en la Tabla 18:

Inicio Construir Solución (LCC, DA, alfa)

1. : SL = []
2. : **Mientras** (continua) **hacer**
3. : beta = Obtener_Mayor (LCC)
4. : tao = Obtener_Menor (LCC)
5. : límite = Restringir_Lista_Candidatos (LCC, alfa, beta, tao)
6. : E = Elemento_Aleatorio (LCC, límite)
7. : Agregar_Elemento (SL, E)
8. : continua = Evaluar_Disponibilidad (DA, E)
9. : **Fin Mientras**
- 10.: Construir_Solución = SL

Fin Construir Solución

Tabla 18: Pseudocódigo del procedimiento Construir Solución

1.3 Explicación del algoritmo GRASP

Para poder obtener un entendimiento adecuado del funcionamiento que posee el algoritmo GRASP es recomendable realizar una breve explicación de cada una de las partes que posee el diseño del pseudocódigo del algoritmo mencionado.

Los parámetros que maneja el procedimiento algoritmo GRASP para iniciar su ejecución, según el pseudocódigo mostrado, son *B* (estructura de bondad), *DA* (disponibilidad de agua), *NI* (número de iteraciones) y *alfa* (constante de relación). La variable *B* contiene la bondad calculada que ofrece cada terreno, cálculo que ha utilizado las características asociadas a los terrenos y aplicado la función objetivo establecido. La variable *DA* indica la cantidad de agua con la que se dispone para el riego de cultivos. La variable *NI* representa el número de iteraciones que se realiza la ejecución del algoritmo GRASP. La variable *alfa* hace referencia al parámetro de relajación necesaria para la restricción de elementos en la lista de candidatos restringidos. Tanto el valor *NI* como de *alfa* son establecidos en la calibración de parámetros del algoritmo GRASP.

A continuación, se explica la ejecución del algoritmo GRASP mediante el detalle del funcionamiento de cada línea presentada en el pseudocódigo anterior.

- Línea 1: Ordenamiento descendente de la lista *B* y almacenada en la lista de candidatos *LC* con el fin de conseguir accesos y búsquedas más eficientes.

- Línea 2: Inicialización de la estructura *CJ* en vacío. Esta estructura es creada para almacenar el conjunto de soluciones para luego generar una población inicial.
- Línea 3: Inicialización en cero de una variable “contador” utilizada para controlar la cantidad de iteraciones que se requiere ejecutar el algoritmo.
- Línea 4: Comienzo de una iteración controlada por el número de iteraciones. Esto se realiza mediante el uso de la variable *NI* y el contador definido de modo que el algoritmo se ejecute la cantidad de iteraciones indicada por la variable.
- Línea 5: Copia de datos de la estructura original *LC* en una estructura auxiliar *LCC*. Esta transcripción se realiza debido a que la longitud de lista de candidatos se reduce conforme se retiran valores. Es por eso que, para no volver a generar esta lista, se trabaja con una copia de esta estructura.
- Línea 6: Construcción de una solución a través de la llamada al procedimiento *Construir Solución*, el cual maneja los parámetros *LCC*, *DA* y *alfa*. Como resultado se obtiene una solución al problema, el cual indica el estado de selección de los terrenos para luego almacenarla en la estructura *SL*. Más adelante se explica el funcionamiento de este procedimiento.
- Línea 7: Almacenamiento de la solución obtenida *SL* en la estructura *CJ*, esta última agrega soluciones verificando su aporte a la población. La estructura *CJ* almacena soluciones (estados de selección de terrenos) obtenidas en cada iteración.
- Línea 8: Actualización de la variable contador de modo que vaya aumentando su valor y así conseguir el control de la ejecución basado en el número de iteraciones.

Tal como se mencionó, a continuación se explica el funcionamiento del procedimiento interno *Construir Solución*:

- Línea 1: Inicialización de la estructura *SL* en vacío. Esta estructura representa una solución al problema (estado de selección de los terrenos) para una iteración.
- Línea 2: Comienzo de una iteración controlada por una variable de estado “continua”. Esta variable controla que el proceso se ejecute mientras haya disponibilidad del recurso hídrico y/o terrenos disponibles para ser evaluados.

- Línea 3: Elección del terreno con mayor utilidad representativa en la lista de candidatos *LCC*. Esta bondad es representada por la variable “beta” como el mejor elemento de la lista. Debido a contar con la lista *LCC* ordenada descendientemente basta con acceder al primer elemento de la lista para obtener este valor mayor.
- Línea 4: Elección del terreno con menor utilidad representativa en la lista de candidatos *LCC*. Esta bondad es representada por la variable “tao” como el peor elemento. Para obtener el menor beneficio basta con acceder al último elemento de la lista.
- Línea 5: Actualización de la lista *LCC* de modo que su longitud se limite por un valor calculado a partir de la variable beta, tao y *alfa*, definidos anteriormente.

El funcionamiento del procedimiento interno *Restringir Lista Candidatos* consiste en conocer un índice que limite el acceso a la lista. Esto se puede realizar debido a que la lista se encuentra ordenada descendientemente de modo que si bien los valores de la lista no se modifican lo que se trata de identificar es el elemento hasta donde se considera retirar un elemento. Este índice es almacenado en la variable “límite”.

- Línea 6: Elección aleatoria de un elemento de la lista *LCC* considerando los elementos permitidos mediante índice límite obtenido. Este elemento elegido es almacenado en la estructura *E* y representa el beneficio que ofrece el terreno seleccionado para ser regado. El elemento *E* es removido de la lista *LCC*.
- Línea 7: Almacenamiento del elemento *E* en la estructura *SL*, esta última agrega elementos que aporten a la función objetivo establecida. La estructura *SL* posee los terrenos que van siendo elegidos mediante la evaluación de beneficios de cada elemento formándose así los terrenos seleccionados.
- Línea 8: Análisis de la disponibilidad del recurso hídrico que se maneja en ese momento de modo que se verifique si se posee suficiente cantidad de agua para satisfacer el requerimiento hídrico del terreno seleccionado y continuar la ejecución.

El funcionamiento del procedimiento interno *Evaluar Disponibilidad* consiste en verificar que el requerimiento hídrico que demanda el terreno seleccionado no exceda a la cantidad de agua disponible *DA*. Si bien el elemento seleccionado de todas maneras forma parte de la solución se valida si el terreno seleccionado es

regado de forma parcial o completa. Si hubiera recurso hídrico suficiente entonces el terreno seleccionado es regado por completo y se reduce la disponibilidad de agua en la cantidad de agua requerida por el terreno seleccionado. En caso contrario, el terreno seleccionado es regado parcialmente y la disponibilidad del recurso hídrico se reduce por completo. Este valor del análisis es representado por la variable de estado “continua”. De no contar con disponibilidad del recurso la iteración termina.

- Línea 10: Envío de la solución obtenida SL por la iteración en el procedimiento Construir Solución.

1.4 Calibración alfa

Para obtener una ejecución adecuada del algoritmo GRASP es necesario la calibración de sus parámetros. Debido a esto, se realiza un procedimiento de calibración del parámetro alfa mediante diversos experimentos, modificando su valor en distintos intervalos. En primer lugar, se experimentó haciendo variar valores alfa desde 0.2 hasta 0.65 en un intervalo de 0.05 en 40 pruebas como se muestra en la Tabla 19. Para más detalle se puede revisar el Anexo 1.

	0.20	0.25	0.30	0.35	0.40	0.45	0.50	0.55	0.60	0.65
Promedio	10.66	10.82	10.86	10.92	11.30	11.34	11.28	11.19	11.27	10.97
Mínimo	7.75	7.75	7.69	8.22	9.19	9.19	9.00	8.21	8.22	8.22
Frec Mínima	1	1	1	1	1	1	1	1	1	1
Máximo	13.72	13.69	13.72	13.72	13.69	13.72	13.69	13.58	13.69	13.72
Frec Máximo	1	1	1	1	1	1	1	1	1	1
Desviación	1.238	1.220	1.240	1.208	0.9831	0.951	1.0463	1.075	1.164	1.168
	5940	7710	6056	7849	71217	3170	31687	0809	4635	7775
	90	63	99	11		36		15	61	50

Tabla 19: Calibración Alfa - Experimento 1

Debido al objetivo de maximizar la función de fitness, se puede apreciar que entre los valores alfa de 0.40 y 0.50 se obtienen los valores más óptimos en promedio de esta función. Por ello, se decide realizar una segunda prueba con valores alfa que oscilen entre 0.35 y 0.53 en un intervalo de 0.02 al igual que en el caso anterior para 40 pruebas como se muestra en la Tabla 20. Para mayor detalle se puede revisar el Anexo 2.

	0.35	0.37	0.39	0.41	0.43	0.45	0.47	0.49	0.51	0.53
Promedio	10.92	11.03	11.11	11.29	11.28	11.34	11.37	11.25	11.17	11.19
Mínimo	8.22	8.29	8.22	9.19	9.19	9.19	9.19	8.22	8.29	8.22
Frec Mínima	1	1	1	1	1	1	1	1	1	1
Máximo	13.72	13.72	13.72	13.69	13.72	13.72	13.69	13.69	13.72	13.58
Frec Máximo	1	1	1	1	1	1	1	1	1	1
Desviación	1.215 6230 86	1.118 5563 04	1.142 7687 64	0.977 0121 91	0.954 5851 23	0.945 5379 69	0.962 9174 42	1.061 1550 31	1.086 7556 69	1.100 2097 41

Tabla 20: Calibración Alfa - Experimento 2

Ahora se ha obtenido un rango dese 0.43 a 0.49 donde el valor de la función fitness es mayor. Por esta razón, se repite el procedimiento de calibración entre 0.42 y 0.52 con un intervalo de 0.01 para las 40 pruebas como se muestra en la Tabla 21. Para más detalle se puede revisar el Anexo 3.

	0.42	0.43	0.44	0.45	0.46	0.47	0.48	0.49	0.50	0.51	0.52
Promedio	11.23	11.26	11.27	11.34	11.37	11.35	11.37	11.25	11.27	11.17	11.22
Mínimo	9.19	9.19	9.19	9.19	9.19	9.19	9.52	8.22	9.00	8.29	9.19
Frec Mínima	1	1	1	1	1	1	1	1	1	1	1
Máximo	13.69	13.69	13.69	13.72	13.53	13.69	13.72	13.69	13.69	13.72	13.69
Frec Máximo	1	1	1	1	1	1	1	1	1	1	1
Desviación	0.967 89088 0	0.967 18677 4	0.992 55898 2	0.951 31703 6	0.898 96749 9	0.961 02426 0	0.969 37460 0	1.061 15503 1	1.028 0257 09	1.086 75566 9	0.989 6386 00

Tabla 21: Calibración Alfa - Experimento 3

Luego del experimento anterior se puede observar que se han obtenido dos valores iguales de función fitness para los valores alfa 0.46 y 0.48. Es así que se procede a realizar una última comparación entre ambos valores para un muestreo de 100 pruebas como se muestra en la Tabla 22. Para mayor detalle se puede revisar el Anexo 4.

	0.46	0.48
Promedio	11.36	11.29
Mínimo	9.50	9.52
Frec Mínima	1	1
Máximo	13.53	13.42
Frec Máximo	1	1
Desviación	0.77138136	0.786317745

Tabla 22: Calibración Alfa - Experimento 4

Por último, se puede observar que la ejecución del algoritmo GRASP con un valor alfa de 0.46 ofrece resultados más óptimos para el problema planteado, por lo tanto se finaliza el procedimiento de calibración de la variable alfa y se establece su valor.

1.5 Calibración número de iteraciones

Luego de realizar la calibración del parámetro alfa se procede a calibrar, como segundo parámetro, el número de iteraciones que será ejecutado el algoritmo GRASP. Para esto, se procede a realizar distintas pruebas modificando el valor del número de iteraciones de 1000 hasta 10500 en un intervalo de 500. Para cada valor de número de iteraciones, se utiliza un muestreo de 40 pruebas de modo que se pueda observar el comportamiento del fitness promedio y tiempo de ejecución consumido de manera que se consiga maximizar la función fitness en un tiempo de ejecución aceptable mediante un balance óptimo como se muestra en la Tabla 23. Para mayor detalle se puede revisar el Anexo 5.

Fitness Promedio	Iteraciones	Tiempo Promedio
11.27	1000	930.00
11.29	1500	970.00
11.32	2000	1037.50
11.33	2500	1212.50
11.34	3000	1472.50
11.35	3500	1525.00
11.37	4000	1670.00
11.37	4500	1950.00
11.38	5000	2200.00
11.38	5500	2222.50
11.38	6000	2435.00
11.38	6500	2522.50
11.38	7000	2802.50
11.38	7500	2995.00
11.39	8000	3130.00
11.38	8500	3262.50
11.38	9000	3612.50
11.38	9500	3660.00
11.39	10000	3800.00
11.38	10500	3850.00

Tabla 23: Calibración Número Iteraciones

Adicionalmente en la Figura 15, se puede visualizar un gráfico con los valores de la función fitness y el número iteraciones ejecutadas por el algoritmo.

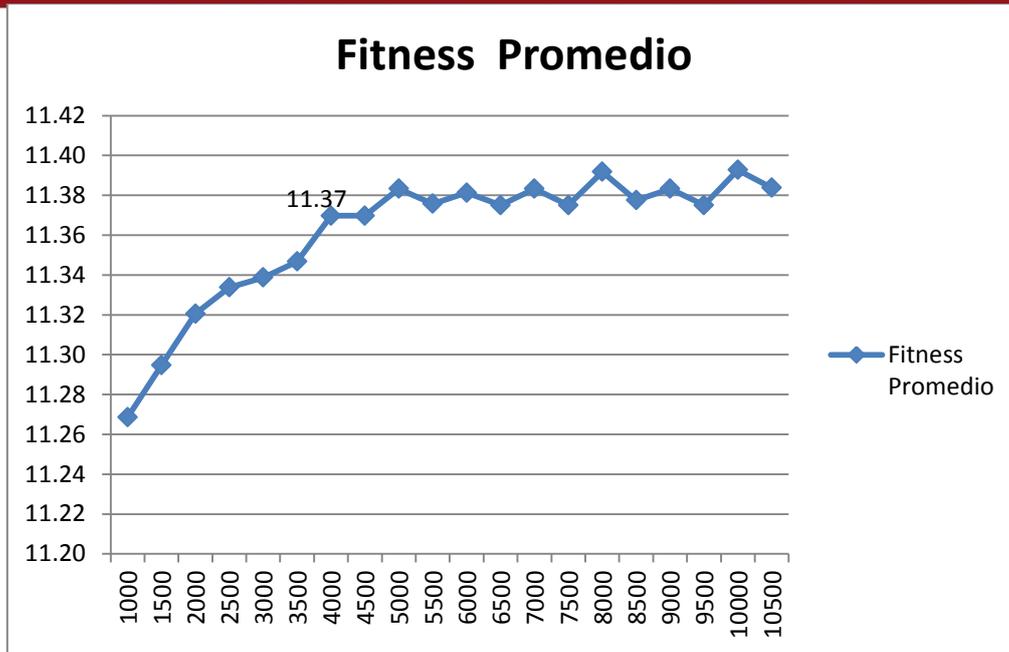


Figura 15: Representación de Fitness Promedio por Iteraciones

Después de visualizar los resultados, se puede notar que la ejecución del algoritmo GRASP con un valor de número de iteraciones de 4000 ofrece resultados más óptimos para el problema planteado. Si bien se observa que existen valores fitness un poco más superiores, se elige este valor de número de iteraciones (4000) debido a que brinda, en promedio, una función fitness bastante buena en un tiempo de ejecución aceptable considerando los recursos computacionales con los que se dispone y otros aspectos de ejecución. Es decir, el valor fitness mejora considerablemente hasta la iteración 40000, sin embargo luego de eso la variación de la misma no es lo suficientemente abrupta como para tomarla en cuenta, considerando el constante incremento del tiempo de ejecución.

En síntesis, se puede observar que la ejecución del algoritmo GRASP con un valor de número de iteraciones de 4000 ofrece resultados óptimos para el problema que se pretende resolver en este proyecto, por lo tanto se finaliza el procedimiento de calibración de la variable de número de iteraciones y se establece su valor para los siguientes procedimientos.

CAPÍTULO 5: MODELO COMPUTACIONAL GENÉTICO

1 Algoritmo Genético

En la siguiente sección, se realiza la presentación del algoritmo Genético. Este algoritmo es la herramienta principal al que está orientado este proyecto. En la sección anterior, se definió el algoritmo de experimentación GRASP, el cual es necesario para la creación de una población inicial del algoritmo genético así como la comparación de resultados de ambos algoritmos en la experimentación numérica. Ahora se realiza la definición del algoritmo principal además del uso del algoritmo auxiliar implementado.

1.1 Variables y estructuras

Para el diseño del pseudocódigo del algoritmo Genético si bien anteriormente se definió las estructuras principales que usa el algoritmo, existen otras estructuras adicionales de uso interno, propio del algoritmo.

A continuación en la Tabla 24, se presentan las estructuras propias del algoritmo:

Símbolo	Descripción
R	Esta variable contiene los requerimientos y características de cada terreno, lo cual es necesario para el cálculo de beneficios de cada uno.
B	Esta variable posee los beneficios calculados que ofrecen cada uno de los terrenos (y su cultivo asociado) a evaluar.
DA	Esta variable indica la disponibilidad de agua inicial con la que cuenta el reservorio establecido.
N	Esta variable indica el número de total de terrenos que se requiere regar en el problema.
CJ	Esta variable contiene el conjunto de las soluciones resultante de la ejecución del algoritmo GRASP.
PI	Esta variable almacena la población inicial adaptada a partir del conjunto de soluciones que generó el algoritmo GRASP.
PN	Esta variable hace referencia a una población nueva con contenido inicial similar a PI pero modificado a través de los operadores genéticos.
NI	Esta variable hace referencia al número de iteraciones que se requiere ejecutar el algoritmo Genético.
NCC	Esta variable indica el número de ciclos que se acepta sin optimizar el valor resultante debido a que la ejecución puede estar en un óptimo.

Símbolo	Descripción
tasaCas	Esta variable indica el porcentaje de la población para el casamiento.
tasaMut	Esta variable indica el porcentaje de la población para la mutación.

Tabla 24: Estructuras propias de Algoritmo Genético

1.2 Pseudocódigo del algoritmo Genético

A continuación, se muestra el pseudocódigo del algoritmo de estudio en la Tabla 25. Sus operadores genéticos y mecanismos de control se muestran en secciones posteriores:

Inicio Algoritmo Genético (R, DA, N, NI, NCC, alfa, tasaCas, tasaMut)	
1. :	Cargar_Datos (R, DA, N, NI, NCC, alfa, tasaCas, tasaMut)
2. :	Para <i>i</i> : 1..N hacer
3. :	$B[i] = R[i].T * (R[i].B / R[i].C) * (R[i].K / R[i].D)$
4. :	Fin Para
5. :	CJ = Algoritmo_GRASP (B, DA, NI, alfa)
6. :	PI = Convertir_Cromosomas (CJ)
7. :	mejor_antiguo = Obtener_MejorFitness (PI)
8. :	PN = PI
9. :	contador = 0
10. :	cambios = 0
11. :	Mientras (seguir) hacer
12. :	Casamiento (tasaCas, PN)
13. :	Mutación (tasaMut, PN)
14. :	Limitar_Población (PN, MP)
15. :	Ruleta (PN)
16. :	mejor_actual = Obtener_MejorFitness (PN)
17. :	seguir = Revisar_Fin (mejor_antiguo, mejor_actual, NI, contador, NCC, cambio, NA, error)
18. :	Fin Mientras
Fin Algoritmo Genético	

Tabla 25: Pseudocódigo del Algoritmo Genético

1.3 Explicación del algoritmo Genético

Luego de la definición del pseudocódigo anterior es necesario obtener un entendimiento adecuado del funcionamiento que posee el algoritmo Genético. Debido a esto, se realiza una explicación de cada una de las partes que posee el pseudocódigo.

Los parámetros que utiliza el algoritmo Genético, según el pseudocódigo mostrado, son R (estructura de requerimientos), DA (disponibilidad de agua), N (número de terrenos), NI (número de iteraciones), $alfa$ (constante de relación), $tasaCas$ (tasa de casamiento) y $tasaMut$ (tasa de mutación). La variable R contiene los requerimientos y características de todos los terrenos. La variable DA indica la cantidad de agua que se dispone para el riego de cultivos. La variable N representa la cantidad total de terrenos. La variable NI representa el número de iteraciones que se realiza la ejecución del algoritmo GRASP. La variable NCC representa el número de ciclos que es aceptable la repetición o no mejora de valores de cada población. La variable $alfa$ hace referencia al parámetro de relajación usada para restringir los elementos de la lista de candidatos. Las variables $tasaCas$ y $tasaMut$ hacen referencia a los porcentajes de individuos de la población que son sometidos a los operadores genéticos de casamiento y mutación, respectivamente.

A continuación, se explica la ejecución del algoritmo Genético mediante el detalle del funcionamiento de cada línea presentada en el pseudocódigo del algoritmo.

- Línea 1: Inicialización y carga de datos de iniciales de los terrenos, además de los parámetros de los algoritmos GRASP y Genético para realizar su ejecución. Más adelante se explica el funcionamiento del procedimiento *Cargar Datos*.
- Línea 2: Inicialización de la lista de utilidad representativo de cada cultivo controlado por el número de terrenos posibles N .
- Línea 3: Generación de la lista B donde los valores indican una utilidad representativa de cada terreno. Esto se realiza a través de los valores que contiene la estructura de requerimientos R . Para cada terreno se evalúa su área T , multiplicado por una razón (beneficio B entre el costo C) y multiplicado por otra razón (rendimiento K entre la distancia D).
- Línea 5: Ejecución del algoritmo GRASP implementado. Más adelante se menciona, brevemente, el funcionamiento del procedimiento *Algoritmo GRASP*.

- Línea 6: Adaptación de la estructura obtenida por la ejecución del algoritmo GRASP a la estructura cromosomática que se utiliza en el algoritmo Genético. En secciones posteriores se realiza la definición del procedimiento *Convertir Cromosomas*.
- Línea 7: Cálculo de la utilidad total del mejor individuo en la población *PI* mediante la función fitness establecida. Más adelante, se explica el funcionamiento del procedimiento *Obtener MejorFitness*.
- Línea 8: Inicialización de la estructura *PN* utilizando la información de *PI* con la finalidad de manejar esta información en la creación de una nueva población.
- Línea 9: Inicialización en cero de una variable “contador” utilizada para controlar la cantidad de iteraciones que se requiere ejecutar el algoritmo. Por lo tanto, es necesario el incremento constante de esta variable.
- Línea 10: Inicialización en cero de una variable “cambios” utilizado para controlar la cantidad de ciclos en que la población actual no ofrece mejores resultados que la anterior (meseta). Por lo tanto, es necesario el incremento constante de esta variable.
- Línea 11: Comienzo de una iteración controlada por una variable de estado “seguir”. Esta variable controla que el proceso se ejecute mientras el número de iteraciones no se haya completado y la cantidad de soluciones repetidas sea aceptable.
- Línea 12: Aplicación del operador genético de casamiento. Como resultado se obtienen nuevos individuos que indican el estado de selección de los terrenos. En secciones posteriores, se realiza la definición del procedimiento *Casamiento*.
- Línea 13: Aplicación del operador genético de mutación. Como resultado se obtienen individuos con genes alterados que representan terrenos. En secciones posteriores, se explica la definición del procedimiento *Mutación*.
- Línea 14: Restricción de la cantidad de individuos dentro de la nueva población que está siendo creada. Si se excede el número de individuos, la cantidad es reducida. En secciones posteriores, se explica la definición del procedimiento *Limitar Población*.

- Línea 15: Aplicación del análisis de individuos mediante la función de ruleta. Esto permite calcular la probabilidad de supervivencia de los individuos generados. En secciones posteriores, se explica la definición del procedimiento *Ruleta*.
- Línea 16: Cálculo de la utilidad total del mejor individuo en la población *PN* mediante la función fitness establecida. Más adelante, se explica el funcionamiento del procedimiento *Obtener MejorFitness*.
- Línea 17: Análisis de la ejecución del algoritmo respecto al número de iteraciones requeridas, la cantidad aceptable de soluciones repetidas y aberraciones generadas. En secciones posteriores, se explica la definición del procedimiento *Revisar Fin*.

Ahora se explica el funcionamiento del procedimiento interno *Cargar Datos*:

- El funcionamiento de este procedimiento consiste en colocar los datos de los terrenos (área, beneficio, costo, rendimiento, distancia y requerimiento hídrico) en memoria de la aplicación a través de la estructura de requerimientos *R*. Además, se establece la configuración de variables y parámetros que necesitan ambos algoritmos para iniciar su ejecución como son las variables *DA*, *N*, *NI*. Por otro lado, *NI* y *alfa* solo para el algoritmo GRASP así como *tasaCas* y *tasaMut* solo para el algoritmo genético. Los valores de estos parámetros son obtenidos luego de realizar el proceso de calibración respectivo.

Explicación del funcionamiento del procedimiento interno *Algoritmo GRASP*:

- El funcionamiento de este procedimiento consiste en la ejecución del algoritmo de experimentación GRASP. Este algoritmo necesita la estructura de utilidad *B*, la variable de disponibilidad de agua *DA* y la variable de relajación *alfa*, las cuales fueron cargadas anteriormente. De este modo, el conjunto de soluciones, mediante la estructura *CJ*, es obtenida. La explicación a detalle del funcionamiento de este procedimiento es realizado en el capítulo 4 de este documento.

Explicación del funcionamiento del procedimiento interno *Obtener MejorFitness*:

- El funcionamiento de este procedimiento consiste en obtener la utilidad representativa del mejor individuo de la población *PI*. En términos del problema que se intenta resolver, se refiere al mayor beneficio total que ofrece un individuo mediante sus terrenos seleccionados. Este valor resultante es representado por la

variable "mejor_antiguo". Este procedimiento realiza el cálculo de la función *Fitness* internamente luego de ubicar el mejor elemento en la población respectiva. El pseudocódigo de la función fitness es mostrada en secciones posteriores.

1.4 Pseudocódigo del procedimiento de conversión

A continuación en la Tabla 26, se muestra el pseudocódigo del procedimiento interno que permite adaptar las soluciones ofrecidas por el algoritmo GRASP a estructuras cromosómicas:

Inicio Convertir Cromosomas (CJ)	
1. :	P = []
2. :	Mientras (CJ <> []) hacer
3. :	S = Obtener_Selección (CJ)
4. :	C = Convertir_Selección (S)
5. :	Agregar_Cromosoma (P, C)
6. :	Fin Mientras
7. :	Convertir_Cromosomas = P
Fin Convertir Cromosomas	

Tabla 26: Pseudocódigo del procedimiento Convertir Cromosomas

1.5 Explicación del procedimiento de conversión

Este procedimiento permite la conversión del conjunto de soluciones a la estructura de un cromosoma. Esta conversión es necesaria debido a que las soluciones no poseen la misma estructura que manejan los cromosomas.

A continuación, se explica el funcionamiento que sigue el procedimiento interno de *Convertir Cromosomas*:

- Línea 1: Inicialización de la estructura P en vacío. Esta estructura es utilizada para almacenar la población inicial luego de la conversión de las estructuras selectivas.
- Línea 2: Comienzo de una iteración controlada por la cantidad de valores en el conjunto *CJ*. La iteración se ejecuta mientras la lista *CJ* contenga elementos.
- Línea 3: Obtención de la estructura de selección de una solución de la lista *CJ*. Este valor es almacenado en la estructura S.

- Línea 4: Conversión de la información de la estructura obtenida S en una estructura cromosómica C. Esta conversión se realiza inicializando los genes del cromosoma en cero y modificando en 1 solo los genes cuyos índices se encuentren en la estructura de selección S (terrenos seleccionados).
- Línea 5: Almacenamiento del cromosoma C en la población P. El almacenamiento del cromosoma se realiza con la creación de un individuo para luego agregarlo.
- Línea 7: Envío de la estructura creada o población P luego de realizar las conversiones y almacenamientos correspondientes.

1.6 Pseudocódigo del operador de casamiento

En esta sección, se presenta el pseudocódigo del operador de casamiento que permite la reproducción de los individuos en la población.

A continuación en la Tabla 27, se muestra el pseudocódigo que posee el operador genético *Casamiento* así como los principales procedimientos internos que maneja:

Inicio Casamiento (tasaCas, PN)

```

1. :   PT = PN
2. :   PH = [ ]
3. :   TP = Obtener_Tamaño (PT)
4. :   índice = 1
5. :   Mientras (TP*tasaCas >= índice) hacer
6. :       padre1 = Obtener_Individuo (PT)
7. :       padre2 = Obtener_Individuo (PT)
8. :       posición = Aleatorio( )
9. :       longitud = Aleatorio( )
10. :      hijo = Obtener_Hijo (padre1, padre2, posición, longitud)
11. :      Si (Revisar_Aberración (hijo)) entonces
12. :          Agregar_Individuo (PH, hijo)
13. :      Fin Si
14. :      índice = índice + 1
15. :   Fin Mientras
16. :   Actualizar_Población (PN, PH)
  
```

Fin Casamiento

Tabla 27: Pseudocódigo del operador de Casamiento

1.7 Explicación del operador de casamiento

Este operador permite el casamiento de individuos con la finalidad de generar nuevos individuos que mejoren la población. Los individuos que son seleccionados para realizar la reproducción son elegidos mediante un procedimiento de selección *Ruleta*.

A continuación, se explica el funcionamiento que sigue el operador *Casamiento*:

- Línea 1: Inicialización de la información de la estructura temporal PT a través de la estructura PN. Esta inicialización se realiza con el fin de no modificar directamente la información de la población original PN y manejar una estructura auxiliar PT.
- Línea 2: Inicialización de la estructura PH en vacío. Esta estructura almacena los nuevos individuos hijos que son generados en este operador.
- Línea 3: Cálculo de la cantidad de individuos que posee la población PT. Este valor permite calcular la cantidad de reproducciones o casamientos que se debe realizar.
- Línea 4: Inicialización en uno de una variable “índice” utilizada para controlar la cantidad de veces que se requiere aplicar el operador de casamiento.
- Línea 5: Comienzo de una iteración controlada por el número de reproducciones que se debe aplicar el operador. Esto se realiza utilizando la variable TP, *tasaCas* e índice de modo que el operador de casamiento se ejecute a una cantidad de veces.
- Línea 6: Elección aleatoria de un individuo de la población PT. Este individuo es almacenado en la variable “padre1” y retirado de la población temporal PT.
- Línea 7: Nuevamente, la elección aleatoria de un individuo de la población PT. Este individuo es almacenado en la variable “padre2” y retirado de la población PT.
- Línea 8: Generación aleatoria de un número que indique la posición de un gen en un cromosoma. En términos del problema, esta posición representa el índice de un terreno dentro del cromosoma. Este valor es representado en la variable “posición”.
- Línea 9: Generación aleatoria de un número que indique una longitud de genes en un cromosoma de modo que se conozca la longitud de corte para el casamiento. Este valor es representado en la variable “longitud”.

- Línea 10: Creación de un nuevo individuo mediante el casamiento de los padres seleccionados. Los padres padre1 y padre2 mezclan sus genes entre los bits posición y longitud. Este nuevo individuo (hijo) es almacenado en la variable “hijo”.
- Línea 11: Verificación de la información del individuo hijo creado. Revisión de posibles aberraciones e información duplicada en el individuo hijo.
- Línea 12: Almacenamiento del nuevo individuo hijo en la estructura PH. Esta estructura contiene los nuevos individuos generados luego del casamiento de padres.
- Línea 14: Actualización de la variable índice mediante el aumento constante de su valor en una unidad de modo que se pueda controlar el número de casamientos.
- Línea 16: Adición de los nuevos individuos creados y contenidos en la estructura PH en la nueva población PN.

Por ejemplo, en la Figura 16 se observa el funcionamiento del casamiento respectivo:

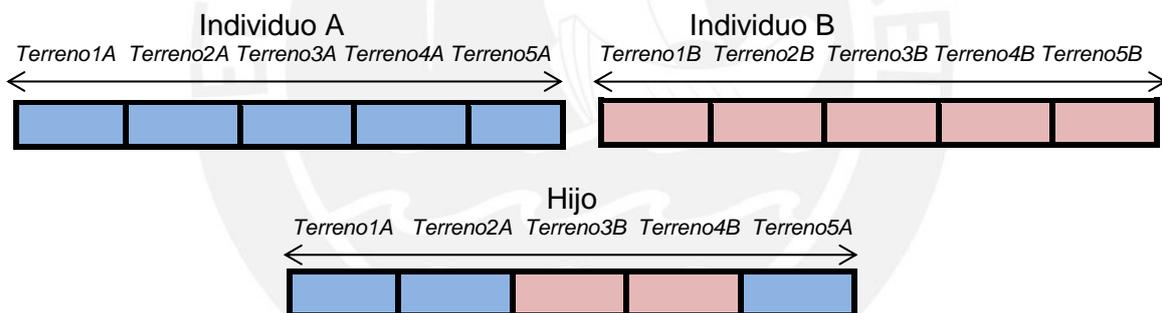


Figura 16: Ejemplo Casamiento

1.8 Pseudocódigo del operador de mutación

Este procedimiento permite la aplicación del operador genético de mutación sobre los individuos que han sido generados por el anterior operador de reproducción.

A continuación en la Tabla 28, se muestra el pseudocódigo del procedimiento interno *Mutación*:

```

Inicio Mutación (tasaMut, PN)
1. :   PT = PN
2. :   PH = [ ]
3. :   TP = Obtener_Tamaño (PT)
4. :   índice = 1
5. :   Mientras (TP*tasaMut >= índice) hacer
6. :       individuo = Obtener_Individuo (PT)
7. :       posición = Aleatorio( )
8. :       individuo_nuevo = Modificar_Individuo (individuo, posición)
9. :       Si (Revisar_Aberración (individuoNuevo)) entonces
10.:           Agregar_Individuo (PH, individuo_nuevo)
11.:       Fin Si
12.:       índice = índice + 1
13.:   Fin Mientras
14.:   Actualizar_Población (PN, PH)
Fin Mutación
  
```

Tabla 28: Pseudocódigo del operador de Mutación

1.9 Explicación del operador de mutación

Este operador permite la mutación de los individuos que han sido creados anteriormente con la finalidad de obtener diversidad en los genes y así mejorar la nueva población. Los individuos que son seleccionados para realizar esta operación son elegidos de manera aleatoria.

A continuación, se explica el funcionamiento que sigue el operador *Mutación*:

- Línea 1: Inicialización de la información de la estructura temporal PT a través de la estructura PN. Esta inicialización se realiza con el fin de no modificar directamente la información de la población original PN y manejar una estructura auxiliar PT.
- Línea 2: Inicialización de la estructura PH en vacío. Esta estructura almacena los individuos con genes modificados que son generados por este operador.
- Línea 3: Cálculo de la cantidad de individuos que posee la población PT. Este valor permite calcular la cantidad de mutaciones que se debe realizar.

- Línea 4: Inicialización en uno de una variable “índice” utilizada para controlar la cantidad de veces que se requiere aplicar el operador de mutación.
- Línea 5: Comienzo de una iteración controlada por el número de mutaciones que se debe aplicar el operador. Esto se realiza utilizando la variable TP, *tasaMut* e índice de modo que el operador de mutación se ejecute a una cantidad de veces.
- Línea 6: Elección aleatoria de un individuo de la población PT. Este elemento es almacenado en la variable “individuo” y retirado de la población temporal PT.
- Línea 7: Generación aleatoria de un número que indique la posición de un gen en un cromosoma. En términos del problema, esta posición representa el índice de un terreno dentro del cromosoma. Este valor es representado en la variable “posición”.
- Línea 8: Modificación genética del individuo mediante la mutación de sus genes. El gen en la posición generada es cambiado. Si es cero se cambia por uno y si es uno, por cero. Este individuo modificado se almacena en la variable “individuo_nuevo”.
- Línea 9: Verificación de la información del individuo nuevo. Revisión de posibles aberraciones e información duplicada en el nuevo individuo.
- Línea 10: Almacenamiento del nuevo individuo en la estructura PH. Esta estructura contiene los nuevos individuos modificados luego de la mutación de genes.
- Línea 12: Actualización de la variable índice mediante el aumento constante de su valor en una unidad de modo que se pueda controlar el número de mutaciones.
- Línea 14: Adición de los nuevos individuos modificados y contenidos en la estructura PH en la nueva población PN.

La mutación de un individuo se realiza de cultivo a cultivo pero del mismo individuo y no respecto de otro como en el casamiento. Esto quiere decir que el intercambio del gen, que representa la selección de un terreno, se realizará con la de otro cultivo en un mismo conjunto de terrenos expresados en el cromosoma asociado al Individuo en cuestión.

Por ejemplo, en la Figura 17 se observa el funcionamiento de la mutación:

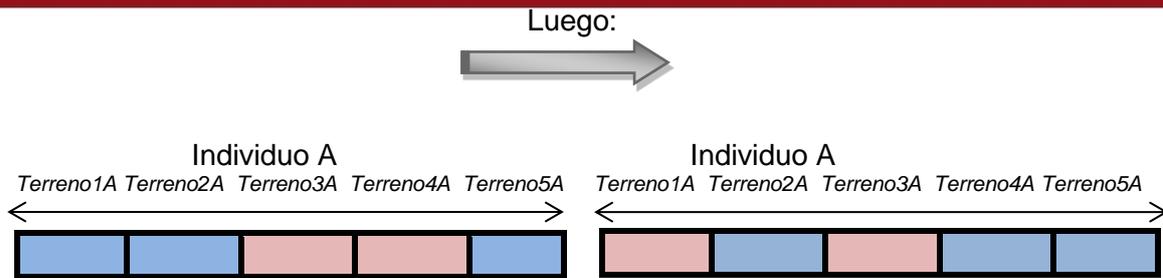


Figura 17: Ejemplo Mutación

1.10 Pseudocódigo del procedimiento de limitación

Este procedimiento permite limitar el tamaño de la población o número de individuos que se posee la nueva población creada.

A continuación en la Tabla 29, se muestra el pseudocódigo que sigue el procedimiento interno de *Limitar población*:

Inicio Limitar Población (PN, MP)	
1. :	TP = Obtener_Tamaño (PN)
2. :	Si (TP > MP) entonces
3. :	FO = Ordenar_ÍndiceFitness (PN)
4. :	Para i: MP..TP hacer
5. :	Remove_Individuo (PN(FO(i)))
6. :	Fin Para
7. :	Fin Si
Fin Limitar Población	

Tabla 29: Pseudocódigo del procedimiento Limitar Población

1.11 Explicación del procedimiento de limitación

Mediante este procedimiento se limita la cantidad de individuos de la población (luego de aplicar los operadores de casamiento y mutación) debido a que este tamaño debe ser una cantidad determinada y no debe cambiar de población en población.

A continuación, se explica el funcionamiento que sigue el procedimiento interno de *Limitar Población*:

- Línea 1: Cálculo de la cantidad de individuos que posee la población PN. Es decir, se requiere el tamaño de la población con la finalidad de conocer, más adelante, si la cantidad de individuos existente excede el tamaño máximo de una población.
- Línea 2: Verificación del tamaño de la población. Esto se realiza mediante el uso de la variable TP y MP de modo que se compruebe si el número de individuos de la población ha excedido el valor máximo del tamaño de una población (MP).
- Línea 3: Ordenamiento descendente de los individuos de la población a través de sus índices o posiciones. Los valores fitness calculados de los individuos son comparados para luego almacenar sus respectivos índices en la estructura FO.
- Línea 4: Inicialización del recorrido en la población mediante la lista de índices generada de modo que los individuos que exceden el tamaño de la población (desde la posición MP hasta TP) sean retirados.
- Línea 5: Retiro de los individuos con menor valor fitness y que exceden el tamaño máximo de la población a través de la lista de índices FO.

1.12 Pseudocódigo de la selección por ruleta

Este procedimiento permite calcular la probabilidad de supervivencia de cada individuo de modo que pueda ser utilizado en los operadores genéticos y otros procedimientos que necesitan seleccionar individuos.

A continuación en la Tabla 30, se muestra el pseudocódigo que sigue el procedimiento interno de *Ruleta*:

Inicio Ruleta (PN)	
1. :	TP = Obtener_Tamaño (PN)
2. :	fitness_total = Calcular_FitnessTotal (PN)
3. :	acumulado = 0
4. :	Para <i>i</i> : 1..TP hacer
5. :	fitness_individuo = Calcular_Fitness (PN(<i>i</i>))
6. :	acumulado = acumulado + fitness_individuo/fitness_total
7. :	Aplicar_Puntuacion (PN(<i>i</i>), acumulado)
8. :	Fin Para
Fin Ruleta	

Tabla 30: Pseudocódigo de la selección por Ruleta

1.13 Explicación del procedimiento de selección

Mediante este procedimiento se realiza el cálculo correspondiente de la probabilidad de supervivencia de cada individuo en la población. Esto se realiza mediante la asignación de un peso o puntuación a cada individuo correspondiente a la bondad del mismo. Asimismo, es importante resaltar que este procedimiento calcula la puntuación de cada individuo mientras que la selección se realiza en el procedimiento de casamiento.

A continuación, se explica el funcionamiento que sigue el procedimiento interno de *Ruleta*:

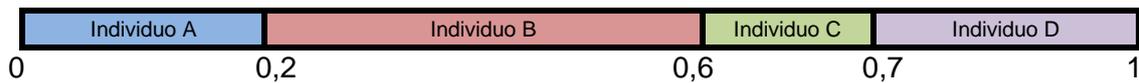
- Línea 1: Obtención de la cantidad de individuos que posee la población PN. Es decir, se requiere el tamaño de la población con la finalidad de realizar, más adelante, el recorrido de cada individuo en la población y obtener su bondad.
- Línea 2: Cálculo del valor fitness total de los individuos en la población PN. Este valor se almacena en la variable “fitness_total” y es utilizado para realizar una normalización de valores acumulados.
- Línea 3: Inicialización en cero de una variable “acumulado” utilizada para almacenar los valores fitness de cada individuo. Debido a esto, esta variable es actualizada constantemente
- Línea 4: Inicialización del recorrido de individuos en la población de tamaño TP con la finalidad de obtener el valor fitness de cada individuo y estimar una puntuación para cada uno mismo respecto al total.
- Línea 5: Cálculo del valor fitness de un individuo específico en la población. Este valor se almacena en la variable “fitness_individuo” y es utilizada posteriormente.
- Línea 6: Acumulación de los valores fitness de cada individuo respecto del valor fitness total. Es decir, se realiza una normalización de valores acumulados dividiendo el valor fitness de cada individuo entre el total y sumándole la puntuación acumulada de los anteriores individuos. De esta forma, se obtiene la puntuación para un individuo, el cual se almacena en la variable “acumulado”.
- Línea 7: Asignación de la puntuación calculada para un individuo de la población. Esta puntuación se encuentra en la variable acumulado.

El procedimiento explicado realiza este funcionamiento obteniendo puntuaciones para cada individuo como se muestra en el siguiente ejemplo:

Antes del cálculo:



Después del cálculo:



De esta manera, se puede observar que ahora cada individuo de la población posee una puntuación relativa al valor fitness total de modo que se pueda realizar una selección adecuada de individuos favoreciendo a aquellos que aportan más a la población.

Como se mencionó al comienzo de este procedimiento, la selección de los individuos se realiza utilizando estas puntuaciones o pesos calculados para cada individuo de modo que aquel individuo con mayor fitness, posee mayor puntuación y mayores probabilidades de ser elegido.

En el operador de casamiento se seleccionan aleatoriamente los individuos utilizando las puntuaciones calculadas de modo que se explore toda la población hasta que se encuentre un valor de puntuación acumulada superior a un valor aleatorio (entre 0 y 1), tal como se muestra en la Figura 18.

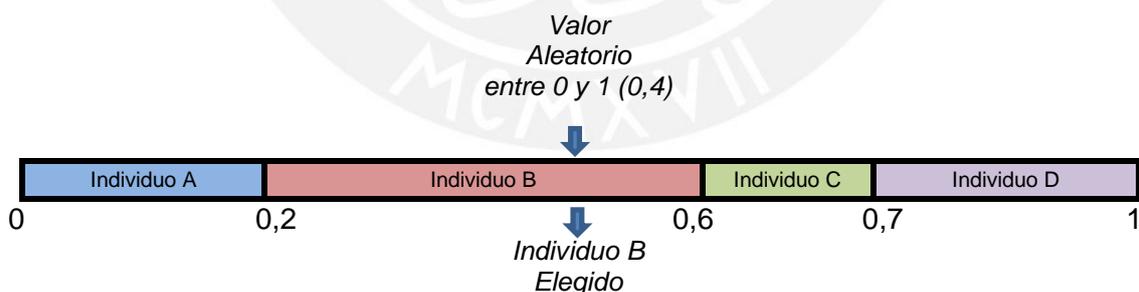


Figura 18: Ejemplo Selección por Ruleta

1.14 Pseudocódigo del procedimiento de revisión

Este procedimiento permite revisar si la ejecución del algoritmo genético ha llegado a su fin o aún falta completar algunas iteraciones más. Básicamente, realiza la comparación de 3 aspectos: la cantidad de valores repetidos generados en la ejecución, el número de aberraciones o soluciones erradas y por último el número de

iteraciones que está permitido ejecutar el algoritmo en su búsquedas de soluciones adecuadas. La explicación de estos aspectos serán explicados a más detalle en el siguiente punto.

A continuación en la Tabla 31, se muestra el pseudocódigo que sigue el procedimiento interno de *Revisar Fin*:

Inicio Revisar Fin (mejor_antiguo, mejor_actual, NI, contador, NCC, cambio, NA, error)	
1. :	contador = contador + 1
2. :	Si (mejor_antiguo >= mejor_actual) entonces
3. :	cambio = cambio + 1
4. :	Caso contrario
5. :	cambio = 0
6. :	Fin Si
7. :	error = error + consultarAberraciones()
8. :	Si (cambio >= NCC error >= NA contador >= NI) entonces
9. :	Revisar_Fin = false
10.:	Caso contrario
11.:	Revisar_Fin = true
12.:	Fin Si
Fin Revisar Fin	

Tabla 31: Pseudocódigo del procedimiento Revisar Fin

1.15 Explicación del procedimiento de revisión

Mediante este procedimiento se realiza un mecanismo de control para detener la ejecución del algoritmo. Este análisis es muy importante debido a que se debe verificar cuándo es necesario y adecuado salir de la ejecución. Para este análisis se verifican tres aspectos. Por un lado, la cantidad de valores repetidos que se obtienen de población en población (conocido como un aspecto tipo meseta), el número de aberraciones obtenida en cada generación poblacional y el número de iteraciones que se va ejecutando el algoritmo.

A continuación, se explica el funcionamiento que sigue el procedimiento interno de *Revisar Fin*:

- Línea 1: Actualización de la variable “contador” de modo que vaya aumentando su valor en una unidad y así conseguir el control de la ejecución basado en el número de iteraciones.
- Línea 2: Comparación entre el mejor valor fitness de la población anterior y la población actual. Esta verificación es posible mediante el uso de la variable mejor_anterior y mejor_actual con la finalidad de comprobar la obtención de valores repetidos o no mejorados en la nueva población.
- Línea 3: Actualización de la variable cambios de modo que vaya aumentando su valor en una unidad de no encontrar mejora en la población. De esta manera, se consigue un control de la ejecución basado en el número de ciclos sin cambiar.
- Línea 5: Reinicialización en cero de la variable cambios debido a la obtención de mejora en los valores fitness de la población. De esta manera, se administra un control de la ejecución basado en el número de ciclos sin cambiar.
- Línea 7: Actualización de la variable “error” de modo que vaya aumentando su valor en una nueva cantidad de aberraciones ocasionadas en la evolución poblacional y así controlar la ejecución basada en el número de aberraciones.
- Línea 8: Verificación de tres criterios de control en la ejecución del algoritmo. El primer control verifica la optimización de la nueva población. Esto es posible a través de las variables mejor_anterior y mejor_actual de modo que se vea si los valores están mejorando, empeorando o inclusive se repiten (meseta). El segundo control verifica la cantidad de aberraciones que se pueden aceptar. Esto es posible a través de las variables NA y error de modo que se compruebe que la ejecución se realice sin ofrecer errores en exceso. El tercer control verifica la cantidad de ejecuciones permitida. Esto es posible a través de las variables NI y contador de modo que se compruebe que la ejecución se realice una cantidad fija de veces.
- Línea 9: Envío del valor resultante obtenido por la verificación de controles en el procedimiento Revisar Fin. Para este valor se indica que algún control o ambos ya no se están cumpliendo y se debe detener la ejecución.
- Línea 11: Envío del valor resultante obtenido por la verificación de controles en el procedimiento Revisar Fin. Para este valor se indica que ambos aún se cumplen y se debe continuar con la ejecución.

1.16 Pseudocódigo de la función de fitness

Este procedimiento permite calcular el valor de la función fitness para un individuo, es decir, la utilidad total que ofrece un individuo a partir de ciertos terrenos seleccionados.

A continuación en la Tabla 32, se muestra el pseudocódigo de la función de *Fitness*:

Inicio Fitness (I)	
1. :	suma = 0
2. :	Para $i: 1..N$ hacer
3. :	Si ($I.S[i] = 1$) entonces
4. :	suma = suma + $B[i]$
5. :	Fin Si
6. :	Fin Para
7. :	Fitness = suma
Fin Fitness	

Tabla 32: Pseudocódigo de la función fitness

1.17 Calibración tasa de casamiento

Para obtener una ejecución adecuada del algoritmo Genético es necesario realizar la calibración de sus parámetros de configuración. Debido a esto, se realiza un procedimiento de calibración del parámetro tasa de casamiento, el cual permitirá conocer el porcentaje de la población que será sometida al operador de casamiento. Esta calibración se realiza mediante un experimento haciendo variar valores tasa de casamiento desde 5% hasta 95% en intervalo de 5% en cada iteración realizada. Para cada valor de tasa de casamiento se emplea un muestreo de 40 pruebas con la finalidad de conocer el comportamiento de la función fitness en promedio ofrecida por el algoritmo GRASP y el algoritmo Genético, además del porcentaje de casamientos realizados con éxito para cada prueba, tal como se muestra en la Tabla 33. Para más detalle se puede revisar el Anexo 6.

	Promedio GRASP	Promedio Genético	Porcentaje Mejora	Promedio Casamiento Exitoso	Mínimo Casamiento Exitoso	Máximo Casamiento Exitoso	Desviación Estándar
5%	11.36	12.28	8.04%	38.50%	30.52%	50.00%	0.907152632
10%	11.36	12.29	8.20%	32.05%	17.27%	50.00%	0.957933138
15%	11.34	12.45	9.77%	37.55%	23.65%	50.00%	1.061092537
20%	11.38	12.41	9.09%	41.06%	25.90%	50.00%	1.046601827
25%	11.37	12.36	8.70%	39.56%	27.63%	48.48%	1.074280200

	Promedio GRASP	Promedio Genético	Porcentaje Mejora	Promedio Casamiento Exitoso	Mínimo Casamiento Exitoso	Máximo Casamiento Exitoso	Desviación Estándar
30%	11.38	12.52	10.04%	39.46%	29.25%	47.40%	1.103937911
35%	11.33	12.47	10.08%	36.72%	27.60%	47.61%	0.979873749
40%	11.34	12.58	10.87%	33.91%	27.50%	43.46%	1.046285126
45%	11.33	12.53	10.61%	31.63%	23.50%	41.27%	1.133138482
50%	11.36	12.69	11.65%	47.04%	40.84%	50.00%	1.094728160
55%	11.34	12.68	11.87%	44.63%	37.44%	49.58%	1.074216202
60%	11.34	12.68	11.78%	42.26%	33.20%	50.00%	1.040198422
65%	11.36	12.72	11.92%	39.50%	30.99%	49.62%	0.986804607
70%	11.37	12.75	12.09%	47.99%	46.21%	50.00%	0.996228786
75%	11.35	12.76	12.48%	48.37%	46.84%	50.00%	0.983225461
80%	11.36	12.80	12.71%	48.54%	44.76%	50.00%	0.995824616
85%	11.34	12.84	13.24%	48.54%	44.76%	50.00%	1.027622068
90%	11.37	12.82	12.68%	48.11%	43.44%	50.00%	1.036678621
95%	11.33	12.79	12.86%	47.43%	40.47%	49.86%	1.023274501

Tabla 33: Calibración Tasa de Casamiento

La información mostrada permite observar el comportamiento del algoritmo Genético para los distintos valores de tasa de casamiento. En este aspecto, la información más importante viene a ser el porcentaje de mejora entre los algoritmos GRASP y genético.

Asimismo, es importante mencionar que el porcentaje de mejora obtenido posee una relación directa con el juego de datos de pruebas con el que se realiza la calibración, la misma que ofrece una idea de cómo se comporta la mejora, sin ser exclusiva. Adicionalmente, se puede visualizar en la Figura 19 los valores del porcentaje de mejora y los valores de tasa de casamiento empleados por el algoritmo.

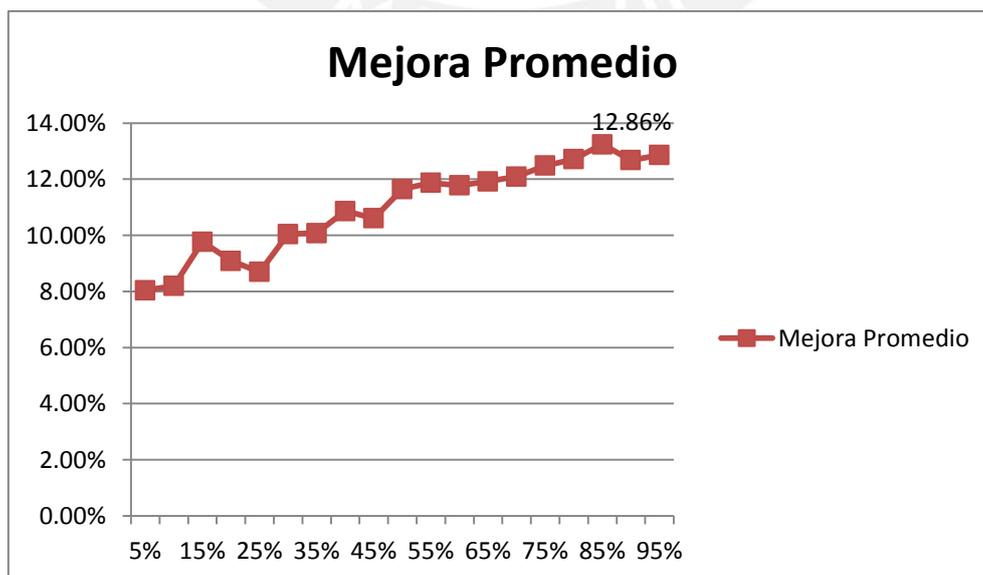


Figura 19: Representación de Mejora Promedio por Tasa Casamiento

De este modo, se puede observar que la tasa de casamiento que brinda mejores resultados es de 85%. Además, este valor posee un porcentaje alto de casamientos exitosos (48.54%), lo cual reafirma la elección de este valor. Por lo tanto, se finaliza el procedimiento de calibración de la variable tasa casamiento y se establece su valor.

1.18 Calibración tasa de mutación

Ahora, se procede a calibrar la tasa de mutación, el cual permitirá conocer el porcentaje de la población que será sometida al operador de mutación. Esta calibración se realiza mediante un experimento haciendo variar valores tasa de mutación desde 4% hasta 13% en intervalo de 1% en cada iteración. Para cada valor de tasa de mutación se emplea un muestreo de 40 pruebas con la finalidad de conocer el comportamiento de la función fitness en promedio del algoritmo GRASP y Genético, además del porcentaje de mutaciones realizadas con éxito como se muestra en la Tabla 34. Para más detalle se puede revisar el Anexo 7.

	Promedio GRASP	Promedio Genético	Porcentaje Mejora	Promedio Mutación Exitoso	Mínimo Mutación Exitoso	Máximo Mutación Exitoso	Desviación Estándar
4%	11.37	12.65	11.25%	23.16%	17.84%	29.22%	1.076538107
5%	11.36	12.66	11.49%	39.97%	30.13%	50.26%	1.027967375
6%	11.35	12.68	11.76%	40.08%	31.97%	53.41%	1.050520750
7%	11.35	12.67	11.60%	38.81%	28.51%	53.39%	1.068240346
8%	11.39	12.75	11.93%	59.22%	42.92%	77.30%	1.040707846
9%	11.37	12.71	11.86%	57.59%	42.60%	78.22%	1.056789981
10%	11.37	12.71	11.80%	74.60%	59.93%	85.90%	1.013585885
11%	11.36	12.72	11.98%	71.63%	57.50%	85.24%	1.069703260
12%	11.37	12.76	12.16%	70.27%	55.83%	84.40%	1.016599535
13%	11.35	12.67	11.63%	76.52%	60.85%	85.20%	1.033941798

Tabla 34: Calibración Tasa de Mutación

La información mostrada permite observar el comportamiento que posee el algoritmo Genético para los distintos valores de tasa de mutación seleccionados. En este aspecto, la información más importante viene a ser el porcentaje de mejora, el cual indica el nivel de mejora del algoritmo genético versus el algoritmo GRASP.

Asimismo, es importante mencionar que el porcentaje de mejora obtenido posee una relación directa con el juego de datos de pruebas con el que se realiza la calibración, la misma que ofrece una idea de cómo se comporta la mejora, sin ser exclusiva. Adicionalmente, se puede visualizar en la Figura 20 los valores del porcentaje de mejora y los valores de tasa de mutación que emplea el algoritmo, respectivamente.

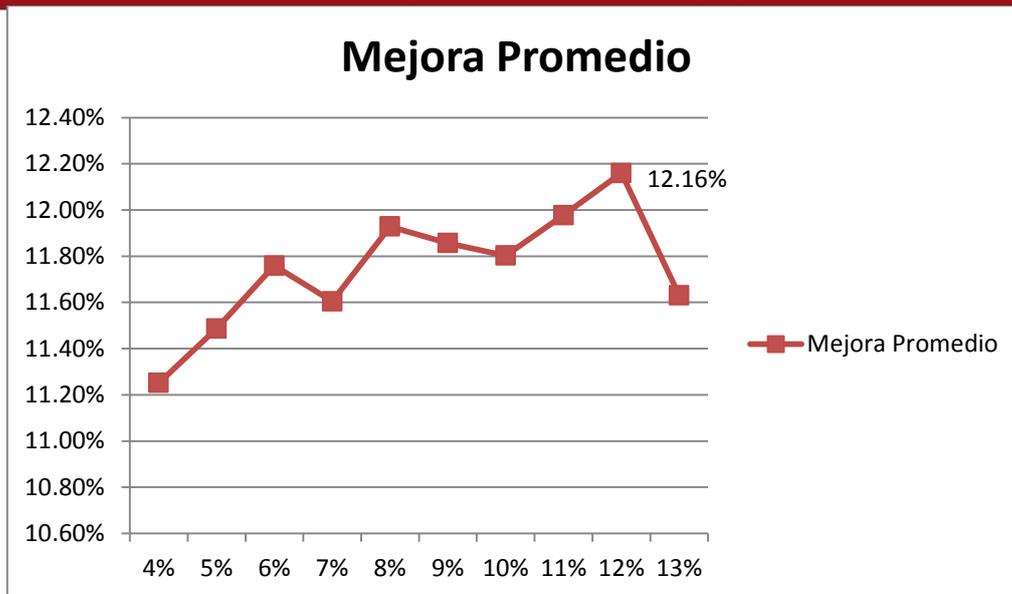


Figura 20: Representación de Mejora Promedio por Tasa Mutación

Análogamente al gráfico obtenido para la calibración de la tasa de casamiento, se posee un gráfico sobre el comportamiento del algoritmo Genético respecto a la mejora que ofrece al ser ejecutado por diversos valores de tasa mutación. En este gráfico se nota claramente un porcentaje de mejora para el valor de tasa de mutación de 0.12 lo que podría indicar que cierta cantidad de aberraciones se están creando en la ejecución haciendo necesario así realizar procesos de mutaciones necesarias en los individuos de la población.

Este último aspecto mencionado podría ser tomado en cuenta más adelante para establecer ciertos mecanismos de control para el número de aberraciones permitido, entre otros.

De este modo, se puede observar que la tasa de mutación de acuerdo al número total de población que brinda mejores resultados es de 12%. Además, este valor posee un porcentaje alto de mutaciones exitosas (70.27%), lo cual reafirma la elección de este valor. Por lo tanto, se finaliza el procedimiento de calibración de la variable tasa mutación y se establece su valor.

CAPÍTULO 6: INTERFAZ GRÁFICA

1 Programa generador de datos

La aplicación principal que se desarrolla en este proyecto requiere de datos iniciales adecuados ya sean por información histórica o generada aleatoriamente. Esta última opción es la elegida de manera que los datos iniciales son soportados por un programa externo, el cual funciona de manera aleatoria, controlada y parametrizable. La generación de datos en el proyecto tiene la particularidad de que si bien se usa un programa externo para tal fin, este se basa en parámetros reales de modo que brinde necesidades, requerimientos y características que se manejan realmente en la actividad agrícola. En primer lugar, se cuenta con características generales como pueden ser:

- Número de terrenos.
- Disponibilidad de agua.

Luego, se poseen otras variables restringidas a un rango de valores máximo y mínimo:

- Área de terreno.
- Costo de producción.
- Rendimiento del cultivo.
- Requerimiento de agua.
- Distancia al reservorio.

Adicionalmente, se cuenta con una variable Beneficio, la cual sigue un procedimiento especial y simplificado para su cálculo debido a la importancia que representa esta variable respecto a las otras que se manejan. Para obtener un adecuado valor del beneficio se adapta previamente una ecuación que permite calcular el margen bruto que genera un cultivo. El uso de esta ecuación es utilizada con la finalidad de simular de la manera más real posible la obtención del beneficio para un cultivo.

$$MB = (Y_p P_p + Y_{sp} P_{sp} + P_c) - (C_p + C_a L_b)$$

siendo MB el margen bruto (s./ha), Y_p el rendimiento agrícola real (kg/ha), P_p el precio de venta del producto (s./kg), Y_{sp} el rendimiento del subproducto comercial (kg/ha), P_{sp} el precio de venta del subproducto comercial (s./kg), P_c los pagos procedente de ayuda compensatoria para algunos cultivos (s./ha), C_p los costos globales de producción (s./ha), C_a los costos globales de aplicación de riego (s./m³) y L_b la lámina bruta de riego (m³/ha) [LEITE, 2013].

De la ecuación definida se puede notar que los ingresos que se manejan son respecto a la venta del producto, subproducto y pagos procedentes de ayuda mientras que los egresos contienen el costo de producción y de aplicación de agua de riego. De este modo, el cálculo del beneficio para este proyecto ha sido simplificado al ingreso obtenido por venta del producto (no del subproducto) luego que el cultivo ha llegado a la etapa de cosecha para su venta comercial.

$$B = Y_p P_p$$

De la expresión anterior, se puede notar que el beneficio (s./ha) es obtenido a partir del rendimiento (kg/ha) y el precio de venta (s./kg). Por un lado, el rendimiento es generado aleatoriamente pero controlado mientras que el precio de venta es calculado.

1.1 Funcionamiento de generación

El procedimiento de generación de datos se realiza de la siguiente manera:

- Primero, se seleccionan el número de terrenos y la cantidad de agua del reservorio que se manejarán en la simulación.
- Luego, se seleccionan valores máximos y mínimos para el área, distancia, costo, rendimiento y requerimiento de hídrico del cultivo que se piensa regar.
- Después, se realiza la generación de datos a través de las variables y rangos seleccionados de máximos y mínimos valores.
- Adicionalmente, se utilizan los valores de requerimiento de agua, rendimiento y costo de cada cultivo para el cálculo de su respectivo beneficio. De este modo, se asegura que los datos que se manejan se encuentren relacionados y simulen la forma como son tratados realmente en la actividad agrícola.
- De esta manera, se obtiene un conjunto de datos asociados a terrenos y cultivos. Estos datos generados son adecuados debido a que están condicionados por valores reales que se manejan en la agricultura.
- Más adelante, luego de la generación, estos datos son exportados para su posterior uso. La exportación de estos datos se realiza en archivos de texto para su mejor visualización y carga posterior.

A continuación en la Figura 21, se muestra la pantalla que posee el programa:



Figura 21: Pantalla de Herramienta generadora de datos

Si bien este programa posee una interfaz gráfica, no se incluye en la interfaz de la aplicación principal. Para la generación de datos se considera que:

- El beneficio es mayor que el costo de producción de cada terreno.
- El rendimiento porcentual de cada cultivo podrá ser a lo más de 100%.

Un breve ejemplo de la generación de datos se muestra en la Tabla 35:

▪ Generación General aleatoriamente	
Número Terrenos = 10 unid	DispAgua = 70,000 m ³
▪ Generación para cada Terreno	
<u>Aleatoriamente</u>	
Área = Random [1 a 3]	→ 2 ha
Distancia = Random [1 a 4]	→ 1km
Req. Agua = Random [6,000 a 11,000]	→ 8,000 m ³ /ha
Costo = Random [3,500 a 6,200]	→ 4,500 s./ha
<u>Calculado</u>	
factorAgua = (8,000 – 6,000) / (11,000 – 6,000)	→ 0.40
Rendimiento = Random [12, 000 a 24,000] * 0.40 + 12,000	→ 16,800 kg/ha
precioCompra = 4,500 * (1/16,800)	→ 0.21 s./ha
precioVenta = Random [0.21 a 1.00]	→ 0.48 s./ha
Beneficio = 16,800 * 0.48	→ 8,350 s./ha

Tabla 35: Ejemplo de generación de datos

1.2 Exportación de datos generados

Como se mencionó, luego de la generación de datos se realiza el almacenamiento de estos en un archivo de texto con el fin de cargar posteriormente esta información.

▪ Definición del archivo

Un archivo es exportado automáticamente luego de la generación de datos. Su estructura cuenta con la información de los terrenos, como se muestra en la Tabla 36:

7	5000					
3	3	0.4	2800	4320	8797	
4	4	0.5	2580	3671	7997	
5	4	0.8	3200	5462	9973	
1	3	1.0	3500	4120	11990	
5	1	0.3	2900	6442	8605	
3	2	0.2	2800	4627	6414	
7	3	0.3	3650	5150	8708	

Tabla 36: Ejemplo de Archivo de Entrada

En el archivo de texto generado se puede visualizar que contiene el número de terrenos y la cantidad de agua en la primera línea. Luego en las siguientes, se repite para cada línea un conjunto de datos conformado por el área, la distancia, el rendimiento, el requerimiento de agua, el costo y el beneficio del cultivo.

2 Ventanas de la aplicación

El propósito del diseño de la interfaz gráfica es permitir a los usuarios cargar la información relacionada a los terrenos para que la aplicación comience a ejecutar sus componentes. El adecuado diseño de la interfaz brinda las siguientes ventajas:

- Los usuarios pueden seleccionar el archivo de datos (definido anteriormente) que les parezca más adecuado para iniciar la ejecución de la aplicación.
- Los usuarios tienen la posibilidad de generar una solución al problema planteado a través de la ejecución del algoritmo genético.
- Los usuarios pueden visualizar los resultados obtenidos por la simulación ejecutada basada en los datos iniciales seleccionados.

2.1 Panel de carga de datos de entrada

La pestaña *carga inicial* permite realizar la búsqueda y carga del archivo que contiene los datos relacionados a los terrenos. Una vez ubicado el archivo se realiza la carga de esta información en las estructuras definidas para el funcionamiento de la aplicación.

Para comenzar con la búsqueda del archivo se presiona el botón “Examinar” de modo que nos muestre una ventana de abrir archivo. Luego de ubicar el archivo y abrirlo se puede ver la ruta del archivo seleccionado en un campo de la ventana. Si el archivo seleccionado no es el indicado se puede borrar esta ruta al presionar el botón “Cancelar” y se vuelve a realizar la búsqueda. Después de abrir el archivo correcto se realiza la carga del contenido al presionar el botón “Cargar” y automáticamente se puede ver que el contenido principal del archivo (nombre del archivo, número de terrenos y disponibilidad de agua) se muestra en una tabla de la ventana. Además, se comprueba la carga exitosa de datos mediante ventanas de confirmación adicionales que se generan.

Por lo tanto en la Figura 22, se cuenta con los datos de los terrenos de manera que se pueda realizar la ejecución del algoritmo genético en otra pestaña de la misma ventana.

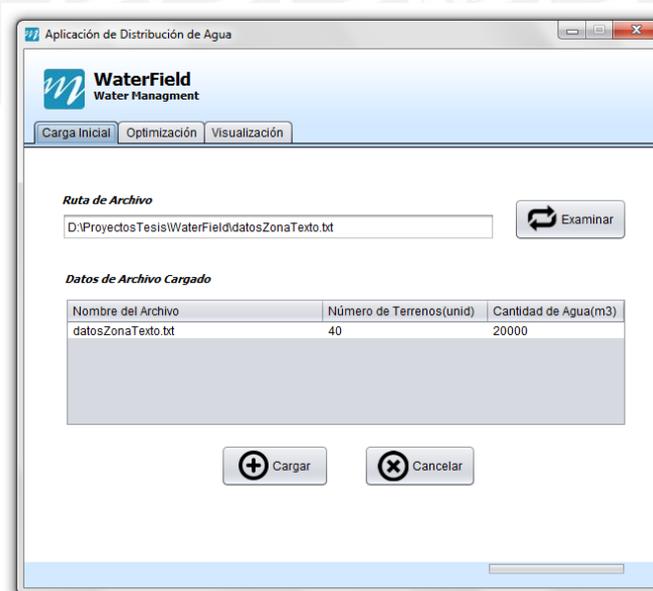


Figura 22: Ventana de pestaña Carga Inicial

2.2 Panel optimización de selección de terrenos

La pestaña *optimización* permite realizar la ejecución del algoritmo genético utilizando los datos de terrenos que ya fueron cargados en las estructuras de la aplicación. Con estos datos ya cargados recién se puede realizar la ejecución del algoritmo principal.

En esta pestaña, antes de realizar la ejecución del algoritmo, se puede ver los principales datos que fueron cargados del archivo tales como el número de terrenos y la disponibilidad de agua en el reservorio. Si no se desea empezar la ejecución utilizando esos valores mostrados del archivo cargado se puede presionar el botón “Cancelar”. Adicionalmente, se puede observar los parámetros de configuración con los cuales trabajarán los algoritmos como son el número de iteraciones y la constante de relajación para el algoritmo GRASP mientras que el número de soluciones, la tasa de casamiento, la tasa de mutación, el valor de elitismo y la opción de seleccionar una población inicial, generada aleatoriamente o construida por GRASP, para el algoritmo Genético. Para comenzar con la ejecución del algoritmo principal se presiona el botón “Generar”. De esta manera, se ejecuta el algoritmo genético el cual brindará una solución al problema de distribución de agua mediante la selección de terrenos. Luego de un tiempo se muestra una ventana indicando que la ejecución se ha realizado correctamente de modo que ya se generó una solución al problema tratado. Para poder ver estos resultados obtenidos se presiona el botón “Visualizar” de modo que la solución generada es mostrada en una nueva pestaña.

Finalmente, se ha realizado la ejecución del algoritmo principal en la aplicación como se muestra en la Figura 23 de modo que se visualizan los resultados en otra pestaña.



Figura 23: Ventana de pestaña Optimización

2.3 Panel visualización de resultados de simulación

La pestaña *visualización* permite observar los resultados obtenidos de la ejecución del algoritmo Genético y GRASP. Además, se cuenta con la información de todos los terrenos que se manejaron para la actual ejecución.

La aplicación o herramienta que permite la adecuada distribución de agua ha utilizado el algoritmo genético implementado y explicado en secciones anteriores. Entre la información más importante que se visualiza en este pestaña se encuentra el identificador del terreno (permite saber si todos los terrenos han sido considerados en la selección), la utilidad del terreno (permite conocer la utilidad que ofrece el terreno al ser elegido o no para ser regado), el área del terreno (permite conocer cuántas hectáreas poseen los terrenos evaluados), el requerimiento de agua y, por último y más importante aún, el estado de selección del terreno (permite verificar si el terreno evaluado ha sido elegido para ser regado o no) con su porcentaje de riego. Adicionalmente, se puede conocer la cantidad total de terrenos seleccionados, la cantidad de hectáreas regadas así como la utilidad representativa y monetaria que ofrece el conjunto de terrenos seleccionados.

De este modo, se puede visualizar los resultados que ha ofrecido la herramienta informática a través de la ejecución del algoritmo Genético y GRASP. Por otro lado, es posible realizar diversas ejecuciones del algoritmo utilizando el mismo archivo de datos o algún otro generando así nuevos resultados que posiblemente no sean los mismos para cada generación, tal como se muestra en la Figura 24.

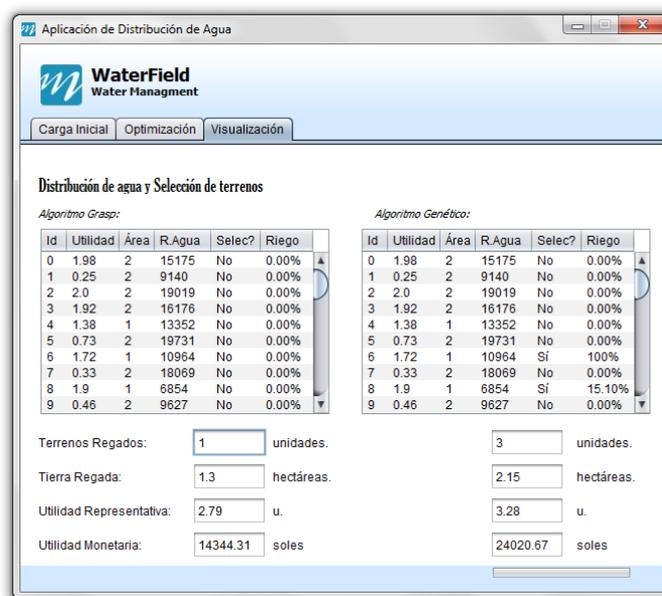


Figura 24: Ventana de pestaña de Visualización

3 Aplicación principal

3.1 Funcionamiento interno de la aplicación

La finalidad de esta sección es explicar cómo internamente la aplicación funciona y hace uso de sus componentes definidos:

- En primer lugar, es necesario la carga de los datos iniciales relacionados a los terrenos para luego cargarlas en las estructuras de la aplicación.
- Luego, se pretende realizar la ejecución del algoritmo Genético. Debido a que el algoritmo genético necesita de una población inicial se ejecuta a priori el algoritmo GRASP usando las estructuras cargadas. Si bien este último proceso no es realizado directamente por el usuario, ya que es ejecutado internamente por el algoritmo genético, es importante mencionarlo.
- Ahora, recién se realiza la ejecución como tal del algoritmo genético. Cada terreno posee sus respectivas características, adicionalmente un estado que representa si fue seleccionado o no, establecido luego de la ejecución.
- Los resultados brindados por el algoritmo principal son mostrados en una tabla que contiene los datos de cada terreno, su estado de selección y beneficio que ofrece. Adicionalmente, se muestra la cantidad de terrenos seleccionados, el beneficio generado y el total de tierra utilizada en la distribución de agua.

A continuación en la Figura 25, se muestra el diagrama de flujo del funcionamiento:

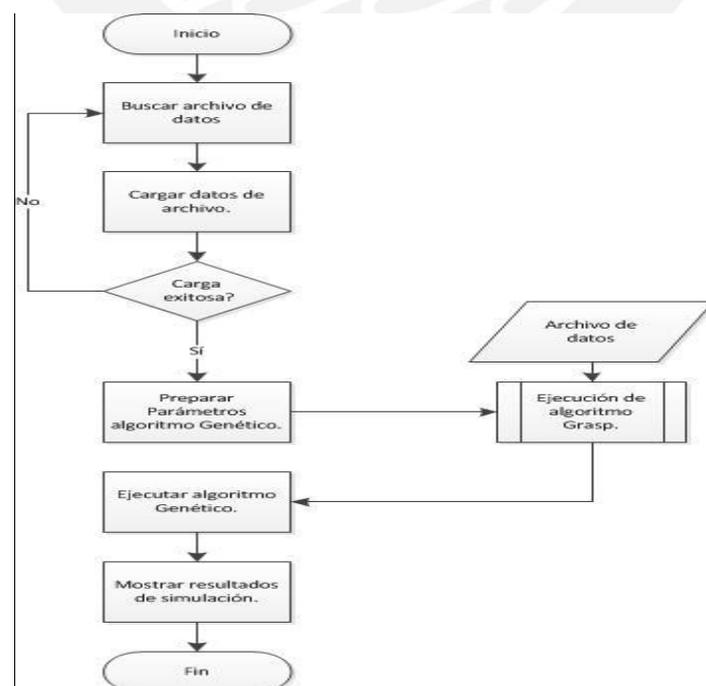


Figura 25: Diagrama de flujo del funcionamiento de aplicación

CAPÍTULO 7: MÉTODOS ESTADÍSTICOS Y RESULTADOS

1 Experimentación numérica

1.1 Introducción

En esta sección, se presenta la experimentación numérica de los algoritmos que tratan de resolver el problema de distribución de agua mediante la selección de terrenos. El objetivo principal de este proceso es determinar qué algoritmo ofrece la mejor solución ante el problema mencionado, siendo por lo tanto necesario e importante su desarrollo.

▪ Justificación del uso de pruebas estadísticas

Para poder justificar la optimalidad de los resultados obtenidos por los algoritmos desarrollados es necesario realizar una comparación de estos resultados. Para realizar la comparación de algoritmos meta heurísticos existen tres aspectos importantes que se deben considerar. Por un lado, se compara el tiempo de ejecución ofrecido por los algoritmos, sin embargo, este aspecto no es necesario en este proyecto debido a que esto se limita en el desarrollo de los algoritmos tratados al considerar un tiempo límite de ejecución. Por otro lado, se compara la complejidad de los algoritmos, aunque este aspecto ya es conocido previamente al momento de seleccionar las metodologías algorítmicas. Como un tercer aspecto, se considera la comparación de la calidad de los resultados, el cual aún no es conocido por lo que se hace necesario su análisis [SILBERHOLZ, GOLDEN, 2010].

La calidad de los resultados es medido en cuánto se desvían los resultados obtenidos de los algoritmos respecto a la solución óptima del problema. Sin embargo, aún no se conoce la solución óptima que ofrece cada algoritmo aunque sí se sabe que el mayor valor de la función objetivo representará la solución óptima. Debido a esto, se puede asumir que a mayor función objetivo, la solución se acerca más al óptimo. Para realizar este procedimiento no basta utilizar una muestra estadística a menos que se seleccione una muestra representativa o aleatoria. De esta manera, se requiere conocer qué algoritmo es el que obtiene la solución óptima con más frecuencia. Es decir, el objetivo es conocer qué algoritmo en promedio obtiene resultados con mayor función objetivo. Este aspecto es posible realizar mediante la comparación de medias de resultados y una prueba que ofrece esta comparación es la prueba ANOVA, la cual permite comparar diversas muestras con varianzas no conocidas aunque partiendo de

que las muestras evaluadas siguen una distribución normal y las varianzas sean homogéneas [SILBERHOLZ, GOLDEN, 2010].

Por lo tanto, queda justificado la comparación de medias de los resultados de los algoritmos además de la elección de la prueba que permita realizar esta comparación: la prueba ANOVA de comparación de medias. Debido a los requisitos de normalidad y homogeneidad de varianzas que exige ANOVA, se realiza la elección de otras pruebas estadísticas adicionales que son detalladas en adelante.

Como ya se mencionó, la experimentación numérica a desarrollar tiene como objetivo determinar qué algoritmo ofrece mejores resultados. Esto se realizará mediante la prueba de comparación de medias que ofrece cada algoritmo desarrollado. Con esta finalidad, se elige la prueba estadística Z. Si bien existen otras pruebas estadísticas como la prueba t-Student, se escoge la prueba Z debido a que el número de elementos con las que se trabaja es superior a 30, además que la varianza de la población es desconocida. Tal como lo recomiendan Berenson y Levine, el uso de la prueba Z para una situación como la mencionada sería lo adecuado. Sin embargo, para poder realizar la prueba Z es necesario cumplir algunos requisitos previamente: las varianzas que poseen las muestras analizadas deben ser significativamente homogéneas y además que estas muestras deben poseer un comportamiento normal.

Debido a esto, se aplica la prueba F de Fisher. Mediante esta prueba se verifica la homogeneidad significativa de la varianza de las muestras. De esta manera, se reconoce la importancia del uso de esta prueba. Del mismo modo con el objetivo de verificar que las muestras tengan un comportamiento normal, se aplica la prueba K-S de Kolmogorov-Smirnov. Si bien existen otras pruebas que pudieron haber sido seleccionados, la razón de elección de esta prueba radica en la experiencia previa que se posee sobre su ejecución además de la confiabilidad de su uso respecto a otras pruebas.

1.2 Recolección de resultados

Para poder iniciar la experimentación numérica, entre los algoritmos desarrollados, es necesario realizar una serie de ejecuciones de ambos algoritmos utilizando un número establecido de juegos de datos de entrada. Debido a que estos juegos de datos son una muestra representativa del comportamiento en general de los algoritmos, al finalizar la experimentación numérica, los resultados de este proceso pueden ser

generalizados a partir de pruebas estadísticas de modo que se determine qué algoritmo es más eficiente.

Por esta razón, se establece la realización de 40 pruebas, en donde cada prueba posee un juego diferente de datos. Los juegos de datos mencionados se refieren a diferentes datos iniciales que se utilizan para cada ejecución de los algoritmos (requerimientos de terrenos en el contexto del problema) de modo que haya diversidad para la generación de resultados. Si bien cada juego de datos es distinto entre sí, estos aún poseen cierta similitud de modo que se maneje una muestra representativa con varianza aceptable entre ellos.

Además, debido a que los algoritmos utilizados poseen un aspecto aleatorio, se realiza una serie de ejecuciones para cada juego de datos de modo que se reduzca el posible margen de error que se genera en el proceso. Adicionalmente, se establece como dato para la experimentación numérica el valor promedio de los resultados de las ejecuciones realizadas. De este modo, se realiza una serie de 10 ejecuciones para cada juego obtenido resultando así un total de 400 ejecuciones para cada uno de los algoritmos.

Luego de promediar los resultados de cada juego de datos recién es posible aplicar las pruebas estadísticas elegidas para esta experimentación. A continuación, se muestra en la Figura 26 el comportamiento del promedio obtenido de función fitness para cada uno de los juegos de datos. Para mayor detalle se puede revisar el Anexo 8.

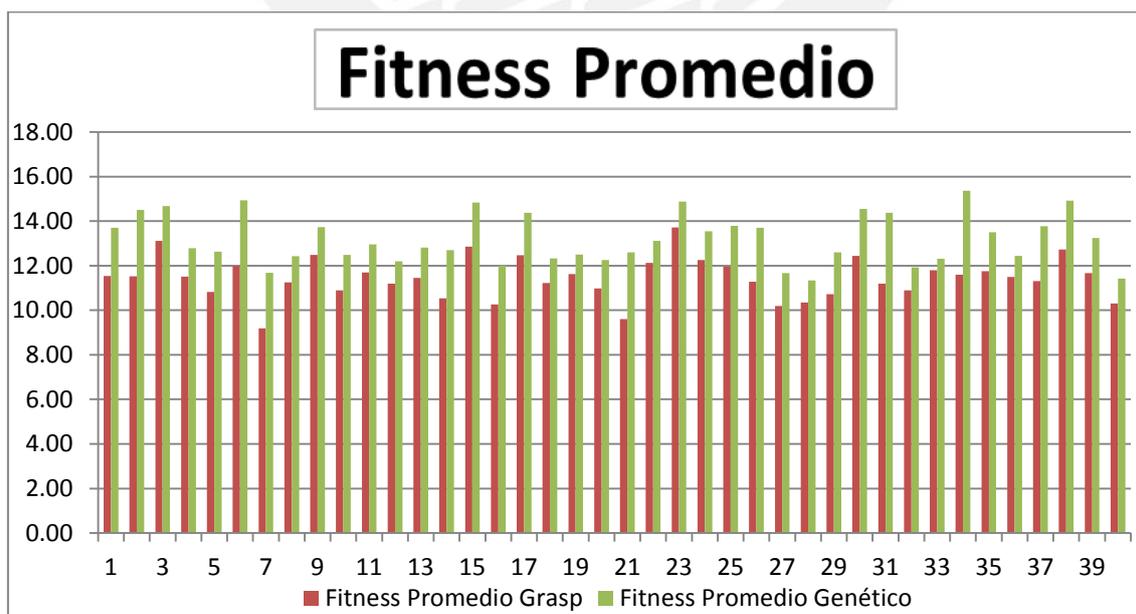


Figura 26: Representación de Fitness Promedio por Muestra utilizada

Luego de culminar la recolección de datos adecuada es posible realizar la experimentación numérica. En esta experimentación se emplean las tres pruebas estadísticas mencionadas en la justificación anterior: prueba K-S (Kolmogorov-Smirnov), prueba F (Fisher) y prueba Z.

1.3 Prueba K-S de Kolmogorov-Smirnov

Al iniciar el análisis estadístico de una serie de datos, un primer paso consiste en describir la distribución de las variables estudiadas y, en particular, de los datos numéricos para poder determinar si estos siguen una distribución normal o no. Con el objetivo de poder demostrar si los datos recolectados de los algoritmos desarrollados siguen una distribución normal, son evaluados mediante la prueba K-S de una muestra.

El test de Kolmogorov-Smirnov (K-S) de una muestra se basa en la idea de comparar la función de distribución acumulada de los datos observados con la de una distribución normal, midiendo la máxima distancia entre ambas curvas. Como en cualquier prueba de hipótesis, la hipótesis nula se rechaza cuando el valor del estadístico supera un cierto valor crítico que se obtiene de una tabla de probabilidad. A través de esta prueba se verifica que las muestras, obtenidas de la ejecución del algoritmo GRASP y Genético, poseen una distribución normal. Debido a que se analizan las muestras de ambos algoritmos de forma independiente, se define un juego de hipótesis para cada algoritmo.

El planteamiento de hipótesis de la prueba K-S para el algoritmo GRASP se muestra en la Tabla 37:

H_0 : <i>La muestra sigue una distribución normal</i> H_1 : <i>La muestra no sigue una distribución normal</i>

Tabla 37: Hipótesis Prueba K-S GRASP

Las hipótesis que han sido definidas indican por un lado, la hipótesis nula H_0 adopta el caso de que la muestra sigue una distribución normal mientras que la hipótesis alterna H_1 adopta el caso de que la muestra no sigue una distribución normal. A partir de estas hipótesis se realiza el experimento respectivo de la prueba K-S para el algoritmo GRASP utilizando las muestras mencionadas obteniendo así los resultados que se muestran en la Tabla 38. Para mayor detalle se puede revisar el Anexo 9.

Resultados GRASP

Valor del estimador	0.078
Grados de libertad	40
Significancia	0.05
Valor crítico	0.21012
Resultado	Estimador fuera de la zona crítica. Se acepta la hipótesis nula.

Tabla 38: Experimentación Prueba K-S GRASP

De la tabla anterior, se puede observar que el valor del estimador obtenido (0.078) es menor al valor crítico obtenido para la prueba de una cola (0.21012) por lo que el valor estimado de la muestra se encuentra fuera del rango de la zona crítica. De este modo, se acepta la hipótesis nula y se concluye que la muestra del algoritmo GRASP sigue una distribución normal.

Para el caso de la muestra obtenida del algoritmo genético, se plantean las mismas hipótesis según la Tabla 39:

H_0 : La muestra sigue una distribución normal H_1 : La muestra no sigue una distribución normal

Tabla 39: Hipótesis Prueba K-S Genético

Análogamente, se realiza el experimento K-S para el algoritmo Genético utilizando las muestras como se indica en la Tabla 40. Para mayor detalle se revisa el Anexo 10.

Resultados Genético

Valor del estimador	0.134
Grados de libertad	40
Significancia	0.05
Valor crítico	0.21012
Resultado	Estimador fuera de la zona crítica. Se acepta la hipótesis nula.

Tabla 40: Experimentación Prueba K-S Genético

Nuevamente, de la tabla anterior, se puede observar que el valor del estimador obtenido (0.134) es menor al valor crítico obtenido para la prueba de una cola (0.21012) por lo que se determina que el estimado de la muestra se encuentra fuera la región crítica. Por lo tanto, se acepta la hipótesis nula y se concluye que la muestra del algoritmo Genético sigue también una distribución normal. Con base en los resultados que se han obtenido por la prueba K-S, la siguiente prueba a realizar es la prueba F de Fisher.

1.4 Prueba F de Fisher

Partiendo de la base de que algunos contrastes de hipótesis dependen del supuesto de normalidad, muchos de estos contrastes siguen siendo aproximadamente válidos cuando se aplican a muestras muy grandes. Tales contrastes se denominan paramétricos.

La prueba de Fisher (F) es una prueba paramétrica que compara la varianza de dos muestras relacionadas y determina si existen diferencias entre ellas. A través de esta prueba se verifica si las varianzas de ambas muestras son significativamente homogéneas o diferentes. Es importante señalar que para poder realizar esta prueba se ha tenido que verificar que la muestra que a evaluar sigue una distribución normal. El planteamiento único de hipótesis para la prueba F se muestra en la Tabla 41:

H_0 : Las varianzas son significativamente homogéneas
 H_1 : Las varianzas son significativamente diferentes

Tabla 41: Hipótesis Prueba F

Las hipótesis que han sido planteadas indican que por un lado la hipótesis nula H_0 adopta el caso en que ambas muestras son significativamente homogéneas mientras que la hipótesis alterna H_1 adopta el caso en que son significativamente distintas. A partir de estas hipótesis se realiza el experimento F para ambos algoritmos utilizando las muestras mencionadas obteniendo los resultados que se muestran en la Tabla 42:

	GRASP	Genético
Media	11.4505	13.18875
Varianza	0.8626	1.23
Observaciones	40	40
Grados de libertad	39	39
F	1.425125786	
P(F<=f) una cola (derecha)	0.136479393	
Valor crítico para F (una cola)	1.704465067	

Tabla 42: Experimentación Prueba F

De la tabla anterior, se puede observar que el valor F para ambas muestras (1.4251) es menor al valor crítico obtenido para la prueba de una cola (1.7044). De este modo, se acepta la hipótesis nula y se concluye que los modelos algorítmicos poseen varianzas significativamente homogéneas. Con el objetivo de corroborar los datos obtenidos mediante esta prueba, se procede a realizar el análisis Z para determinar si la media de una población presentada es mayor o no que la de otra población.

1.5 Prueba Z

Uno de los métodos estadísticos más utilizados para comparar muestras es la prueba Z. La prueba (Z) compara las medias de dos muestras relacionadas y determina si una es mayor o no que la otra. A través de esta prueba se verifica cuál de los dos modelos algorítmicos, posee la media más alta. La realización de esta prueba es de mucha importancia debido a que permite demostrar que los resultados obtenidos por algoritmo Genético son más óptimos que los ofrecidos por el algoritmo GRASP.

Para poder determinar lo planteado es recomendable realizar dos pruebas de hipótesis. La primera prueba consiste en diferenciar si las medias de ambos modelos son iguales mientras que la segunda, si son distintas. Para ello se presentan las siguientes hipótesis tal como se muestra en la Tabla 43:

H_0 : La media del algoritmo Genético es igual a la media del algoritmo Grasp
H_1 : La media del algoritmo Genético es diferente a la media del algoritmo Grasp

Tabla 43: Hipótesis Prueba Z – Experimento 1

Las hipótesis que han sido planteadas indican que por un lado, la hipótesis nula H_0 adopta el caso en que la media del algoritmo Genético es igual a la ofrecida por el algoritmo GRASP mientras que en la hipótesis alterna H_1 adopta el caso en que la media del algoritmo Genético es diferente a la ofrecida por el algoritmo GRASP. A partir de estas hipótesis se realiza el primer experimento respectivo de la prueba Z para ambos algoritmos utilizando las muestras mencionadas obteniendo así los resultados que se muestran en la Tabla 44:

	GRASP	Genético
Media	11.4505	13.18875
Varianza	0.862625385	1.229349679
Observaciones	40	40
Diferencia Hipotética de medias	0	
Z	7.600881853	
P(Z ≤ z) dos colas	6.66134E-15	
Valor crítico de z (dos colas)	1.959963985	

Tabla 44: Experimentación Prueba Z – Experimento 1

De la tabla anterior, se puede observar que el valor Z para ambas muestras (7.60088) es mayor al valor crítico obtenido para la prueba de dos colas (1.959963) por lo que el valor F de la muestra se encuentra dentro del rango de la zona crítica. De este modo, se rechaza la hipótesis nula y se concluye que la media obtenida por el algoritmo Genético es distinta a la media obtenida por el algoritmo GRASP.

Luego de conocer y confirmar que las medias de las muestras dadas son distintas, es necesario determinar si una de ellas posee mayor o menor media que la otra. Para ello, se plantean las siguientes hipótesis, según la Tabla 45:

H_0 : La media del algoritmo Genético es menor a la media del algoritmo Grasp

H_1 : La media del algoritmo Genético es mayor a la media del algoritmo Grasp

Tabla 45: Hipótesis de prueba Z – Experimento 2

Análogamente, las hipótesis que han sido planteadas indican que por un lado, la hipótesis nula H_0 adopta el caso en que la media del algoritmo Genético es inferior a la ofrecida por el algoritmo GRASP mientras que en la hipótesis alterna H_1 adopta el caso en que la media del algoritmo Genético es superior a la ofrecida por el algoritmo GRASP. A partir de estas hipótesis se realiza el segundo experimento de la prueba Z obteniendo así los resultados que se muestran en la Tabla 46:

	GRASP	Genético
Media	11.4505	13.18875
Varianza	0.862625385	1.229349679
Observaciones	40	40
Diferencia Hipotética de medias		0
Z		7.600881853
P(Z ≤ z) una cola		3.33067E-15
Valor crítico de z (una cola)		1.645

Tabla 46: Experimentación Prueba Z – Experimento 2

De la tabla anterior, se puede observar que el valor Z para ambas muestras (7.60088) es mayor al valor crítico obtenido para la prueba de una cola (1.645) por lo que el valor F de la muestra se encuentra dentro del rango de la zona crítica. De este modo, se rechaza la hipótesis nula, se acepta la hipótesis alterna y se concluye que la media obtenida por el algoritmo Genético es mayor a la media del algoritmo GRASP.

A partir de los resultados obtenidos, se puede notar que el algoritmo Genético ofrece, en promedio, un mejor valor de función fitness que el algoritmo GRASP, el cual era el resultado que se esperaba pero necesitaba ser comprobado mediante diversas pruebas estadísticas u alguna otra herramienta de validación.

Por último, debido a que en el presente proyecto se pretende maximizar la función objetivo para cada muestra que utilizan los algoritmos se puede concluir que el algoritmo Genético ofrece resultados más óptimos que el algoritmo GRASP para el problema planteado en este proyecto.

CAPÍTULO 8: CONCLUSIONES Y TRABAJOS FUTUROS

1 Observaciones

Cabe señalar que el presente proyecto ha restringido ciertos aspectos de la aplicación desarrollada como ha sido mencionado en el alcance debido a que el proyecto está orientado al desarrollo de un algoritmo genético más que a un sistema en particular. El manejo de la información de los terrenos se está simplificando en la carga del archivo de modo que su ingreso sea más eficiente y consistente al momento de manejar esta.

Por otro lado, es importante mencionar que solo se está optimizando la distribución de agua mediante algoritmos genéticos ya que para realizar una completa gestión del recurso hídrico se utilizaría, además del algoritmo, nuevas herramientas de soporte. Debido a la importancia de encontrar un balance entre la eficiencia del algoritmo y los recursos utilizados, se ha tenido un cuidado especial en el diseño de las estructuras.

2 Conclusiones

A partir de la investigación realizada se puede concluir, en primer lugar, que el problema de satisficibilidad es un problema complejo que posee diversas restricciones y consideraciones. Para el caso de estudio, algunas de las restricciones más importantes son el costo y beneficio, así como el rendimiento del cultivo.

Por otro lado, luego de haber realizado la implementación de los algoritmos GRASP y Genético para resolver el problema específico de distribución del agua en el riego de cultivos así como la comparación de resultados en la experimentación numérica, se pueden establecer las siguientes conclusiones:

- En primer lugar, los algoritmos desarrollados efectivamente resuelven el problema planteado en el presente trabajo de modo que permiten obtener una distribución del recurso hídrico con una alta utilidad de riego generada.
- En segundo lugar, los resultados del algoritmo Genético superan los resultados del algoritmo GRASP para las mismas instancias (cantidad de terrenos y de agua).

Para haber establecido esta última conclusión se realizó la experimentación numérica mencionada anteriormente. Previamente a esto, para obtener resultados muchos más precisos y cercanos al óptimo por parte de cada algoritmo, se realizó una calibración de los distintos parámetros que maneja cada metodología. De este modo, para el caso en particular del problema que se pretende resolver, se obtuvo un valor de alfa de 0.46

y número de iteraciones de 4000 para el algoritmo GRASP, mientras que un valor de tasa de casamiento de 0.85 y tasa de mutación de 0.12 para el algoritmo Genético. De este modo, se obtuvieron los resultados por cada algoritmo desarrollado en la experimentación numérica percibiendo una mejora de resultados del 14% de la función fitness del algoritmo Genético respecto al GRASP, como se muestra en la Figura 27:

Algoritmo GRASP		Algoritmo Genético propuesto	
($\alpha = 0.46$, iteraciones = 4000)	11.46	(tCasamiento = 0.85, tMutación = 0.12)	13.14

Figura 27: Comparación algoritmo GRASP vs. Genético propuesto

Las ejecuciones fueron realizadas para instancias de 40 terrenos y 80000 metros cúbicos de agua. Por lo tanto, la mejora señalada confirma la optimización de resultados esperados en este proyecto. Adicionalmente, se realizaron ejecuciones del algoritmo Genético partiendo de una población inicial aleatoria. Se percibe una mejora de resultados del 12% como se muestra en la Figura 28 (revisar el Anexo 11).

Promedio Genético Población Inicial Aleatoria		Promedio Genético Población Inicial GRASP	
(tCasamiento = 0.85, tMutación = 0.12)	12.88	(tCasamiento = 0.85, tMutación = 0.12)	13.14

Figura 28: Comparación Población Inicial Aleatoria vs GRASP del Genético

2.1 Recomendaciones y trabajos futuros

En el presente trabajo, se desarrolló un algoritmo genético para poder obtener la óptima distribución del agua mediante la selección de terrenos a ser regados, de modo que se obtenga una solución en un tiempo adecuado.

Entre los trabajos futuros que se recomiendan realizar para el presente algoritmo implementado se encuentran:

- Confrontar el algoritmo desarrollado con otras meta heurísticas que han tenido éxito en tareas de optimización combinatoria y en particularmente para este problema como pueden ser Colonia de Hormigas, Búsqueda Tabú, entre otras.
- Realizar la fase mejoría del algoritmo GRASP para afinar la distribución de agua que realiza previamente el algoritmo de experimentación.
- Integrar el algoritmo Genético desarrollado como parte de un sistema de gestión de riego modificando ciertos componentes de entrada y salida de modo que se pueda establecer comunicaciones con otras aplicaciones del sistema.

Referencias Bibliográficas

Libros

BECK, Kent and ANDRES, Cynthia

2008 *“Extreme Programming Explained: Embrace Change”. Second Edition*
Pearson Education, USA.

BERENSON, Mark, LEVINE, David and KREHBIEL, Timothy

2006 *“Estadística para Administración”*.
Pearson Educación. USA.

COOK, Stephen and MITCHELL, David

1997 *“Finding hard instances of the Satisfiability problem”*. American
Mathematical Society DIMACS, USA.

DU, Ding-Zhu and PARDALOS, Panos*

1998 *“Handbook of combinatorial optimization”*.
Kluwer, NETHERLANDS.

GOLDBERG, David

1989 *“Genetic Algorithms in Search, Optimization and machine learning”*.
Addison-Wesley, USA.

GU, Jun, PURDOM Paul, FRANCO John and WAH Benjamin

1999 *“Algorithms for the Satisfiability (SAT) problem”*. Kluwer Academic
Publisher, USA.

HOLLAND, John

1975 *“Adaptation in natural and artificial systems”*.
MIT press, USA.

MARTELLO, Silvano and TOTH, Paolo*

1990 *“Knapsack Problem: Algorithms and Computer Implementations”*
John Wiley & Sons, USA.

PAPADIMITRIOU, Christos and STEIGLITZ, Kenneth*

1982 *“Combinatorial Optimization, Algorithms and Complexity”*.
Prentice-Hall, USA.

PMBOK

2009 *“A guide to The Project Management Body of Knowledge (Pmbok
Guide)”*. Fourth Edition.
PMI Publications, USA.

TUPIA, Manuel

2009 *“Fundamentos de Inteligencia Artificial”*.
Tupia Consultores y Auditores S.A.C, PERU.

WINSTON, Wayne

2005 *“Investigación de Operaciones: Aplicaciones y algoritmos 4.Ed”*.
Thomson, USA.

Artículos**ABDECHIRI, Marjan and REZA, Mohammad**

2012 *"A Hybrid Hopfield Network-Imperialist Competitive Algorithm for Solving the Satisfiability Problems"*.
 International Journal of Computer and Electrical Engineering
 Consulta: 31 de Octubre del 2014.
 Disponible en:
 < <http://www.ijcee.org/papers/594-P20011.pdf> >

BIERE, Armin and SINZ, Carsten

2006 *"Decomposing SAT problems into connected components"*.
 Journal of Satisfiability, Boolean Modeling and Computation
 Consulta: 31 de Octubre del 2014.
 Disponible en:
 < http://jsat.ewi.tudelft.nl/content/volume2/JSAT2_9_Biere.pdf >

CARRION, Pedro, LOPEZ, Eulogio, ORTEGA, José and DE JUAN, Arturo*

2003 *"Optimización mediante algoritmos genéticos de la gestión del agua en el regadío"*. Revista Internacional de Métodos Numéricos para Cálculo y Diseño en Ingeniería.
 Consulta: 05 de Septiembre del 2013.
 Disponible en:
 < <http://upcommons.upc.edu/revistes/bitstream/2099/2975/1/RR194D.pdf> >

FEO, Thomas and RESENDE, Mauricio

1995 *"Greedy Randomized Adaptive Search Procedures"*.
 Journal of Global Optimization.
 Consulta: 15 de Septiembre del 2013.
 Disponible en:
 < <http://link.springer.com/article/10.1007/BF01096763#page-1> >

FESTA, Paola, PARDALOS, Panos, PITSOULIS, Leonidas and RESENDE, Mauricio

2007 *"GRASP with path relinking for the weighted MAXSAT problem"*.
 Journal of experimental Algorithmics (JEA)
 Consulta: 31 de Octubre del 2014.
 Disponible en:
 < <http://dl.acm.org/citation.cfm?id=1216581> >

GIUNCHIGLIA, Enrico, NARIZZANO, Massimo and TACCHELLA, Armando

2001 *"QuBE: A system for deciding quantified Boolean formulas Satisfiability"*
 Proceedings of the First International Joint Conference on Automated Reasoning
 Consulta: 31 de Octubre del 2014.
 Disponible en:
 < <http://dl.acm.org/citation.cfm?id=753955> >

LONGZHU, Guo, SHIZHANG, Peng and SHENGMIN, Peng

2009 *"Research on Allocation Schemes of Water Resources Sustainable Utilization Based on Global Optimization"*.

IEE 2009 International Conference on Information Science and Engineering
 Computer Society.

Consulta: 17 de Septiembre del 2013.

Disponible en:

< <http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5303385> >

MARQUES, Joao and SAKALLAH, Karem

1999 “GRASP – a new search algorithm for Satisfiability”.

IEEE/ACM International conference on computer aided design.

Consulta: 31 de Octubre del 2014.

Disponible en:

< <http://dl.acm.org/citation.cfm?id=244560>>

RETURETA, Elsa

2010 “Estadística Inferencial: Planteamiento de Hipótesis en más de dos poblaciones”. Facultad de Administración, Universidad Veracruzana.

Consulta: 04 de Octubre del 2013.

Disponible en: <<http://www.slideshare.net/lizgc/fplanteamiento-de-hipotesis-en-mas-de-dos-poblaciones>>

SCHAEFER, Thomas

1978 “The complexity of Satisfiability problems”.

Department of Mathematics. University of California, Berkeley, California.

Consulta: 31 de Octubre del 2014.

Disponible en:

< <http://dl.acm.org/citation.cfm?id=804350> >

SILBERHOLZ, John and GOLDEN, Bruce

2010 “Comparison of Metaheuristics”. Center for Scientific Computing and Modeling, University of Maryland.

Consulta: 17 de Septiembre del 2013.

Disponible en:

< http://www.cscamm.umd.edu/publications/BookChapter_CS-09-13.pdf >

VILLAGRA, Marcos and BARÁN, Benjamín

2007 “Ant colony optimization with adaptive fitness function for Satisfiability testing”.

Proceedings of the 14th international conference on Logic, language, information and computation

Consulta: 31 de Octubre del 2014.

Disponible en:

< <http://dl.acm.org/citation.cfm?id=1770202f>>

XIAOYA, Li and JINCHUAN, Cui*

2009 “Real-Time water resources allocation: methodology and mechanism”

IEE 2009 International Conference on Information Science and Engineering
 Computer Society.

Consulta: 17 de Septiembre del 2013.

Disponible en:

< <http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5373121>>

Tesis**BEJARANO, Gissella**

2010 *“Planificación de horarios del personal de cirugía de un hospital del estado aplicando algoritmos genéticos (time tabling problem)”*. Tesis para optar por el Título de Ingeniero Informático. Especialidad de Ingeniería Informática. Pontificia Universidad Católica del Perú, PERÚ.
 Consulta: 03 de Septiembre del 2013.

CALDERON, Imanaida

2002 *“Desarrollo y Prueba de una metodología de Simulación Basada en Algoritmos Genéticos”*. Tesis para optar por el Título de Ingeniero de Petróleo. Universidad central de Venezuela, VENEZUELA.
 Consulta: 04 de Octubre del 2013.

GANOZA, Dante and SOLANO, Úrsula

2004 *“Un algoritmo de búsqueda adaptativa aleatoria y golosa para la resolución del problema de cortes”*. Tesis para optar por el Título Profesional de Ingeniero de Sistemas e Informática. Lima: Universidad Nacional Mayor de San Marcos, PÉRU.
 Consulta: 04 de Octubre del 2013.

LEITE, Kelly

2013 *“Mejora de la eficiencia del uso del agua de riego de cultivos hortícolas, mediante el uso del modelo Mopeco”*. Página 54. Tesis Doctoral. Departamento de Producción Vegetal y Tecnología Agraria. Universidad de Castilla La Mancha, ESPAÑA.
 Consulta: 04 de Noviembre del 2014.

PACHECO, Rossana

2008 *“Conflictos en torno al Manejo de Agua para Riego en la Microcuenca del Rio Payac”*. Tesis para optar por el título de Magister en Sociología. Escuela de Graduados. Pontificia Universidad Católica del Perú, PERÚ.
 Consulta: 12 de Septiembre del 2013

ORTEGA, José Fernando

2000 *“Desarrollo de una Metodología para la Ayuda a la Toma de Decisiones en el Marco de una Agricultura de Regadío Sostenible. Aplicación del modelo en el Sistema Hidrogeológico 08.29”*. Tesis Doctoral. Departamento de Producción Vegetal y Tecnología Agraria. Instituto de Desarrollo Regional, ESPAÑA.
 Consulta: 04 de Noviembre del 2014.

PORTOCARRERO, Daniel

2010 *“Manejo Sostenible del Agua subterránea con fines de riego en el valle de Acari”*. Tesis para optar por el título de Magister Scientiae. Especialidad de Recursos Hídricos. Universidad Nacional Agraria La Molina, PERÚ.
 Consulta: 12 de Septiembre del 2013.

ROJAS, Vicente

2011 *“Software para determinar costos de producción en cultivos transitorios de la costa central del Perú”*. Tesis para optar por el título de Magister Scientiae. Especialidad de Producción agrícola. Universidad Nacional Agraria La Molina, PERÚ.

Consulta: 12 de Septiembre del 2013.

TUPIA, Manuel

2005 *“Un algoritmo GRASP para resolver el problema de la programación de tareas dependientes en máquinas diferentes”*. Tesis para optar por el título de Magister en Ingeniería de Sistemas con la mención en Ingeniería de Software. Lima: Universidad Nacional Mayor de San Marcos.
Consulta: 04 de Octubre del 2013.

Revistas

REEAP*

2008 *“Diseños alternativos para un índice de sostenibilidad agrícola: el caso de la agricultura de regadío del Duero (*)”*.
Consulta: 21 de Septiembre del 2013.
Disponible en:
<http://www.magrama.gob.es/ministerio/pags/Biblioteca/Revistas/pdf_REEAP%2Fr219_135_180.pdf >

VILLAMIZAR, Juan Carlos

2004 *“Técnicas para resolver el problema Satisfacibilidad y sus aplicaciones”*.
Revista de Tecnología.
Consulta: 31 de Octubre del 2014.
Disponible en: <
http://www.uelbosque.edu.co/sites/default/files/publicaciones/revistas/revista_tecnologia/volumen3_numero2/tecnicas_resolver_problema_satisfacibilidad3-2.pdf >

Referencias Web

ANA

2013 *“Estadísticas del recurso hídrico”*.
Consulta: 03 de Septiembre del 2013.
Disponible en: <<http://www.ana.gob.pe/>>

2013 *“Reglamento de la Ley de Recursos Hídricos. Ley N° 29338”*.
Consulta: 03 de Septiembre del 2013.
Disponible en: <<http://www.ana.gob.pe/>>

FAO

2013 *“Glossary on Organic Agriculture”*.
Consulta: 03 de Septiembre del 2013.
Disponible en:
<http://www.fao.org/fileadmin/templates/organicag/files/Glossary_on_Organic_Agriculture.pdf >

IMS

2013 *“Mottech Water Management”*.
Consulta: 16 de Septiembre de 2013.
Disponible en:

<http://www.mottech.com/index.php?option=com_k2&view=item&layout=item&id=24&Itemid=97>

JAVA

2013

“Características y ventajas de Java”.

Consulta: 29 de Agosto del 2013.

Disponible en:

< <http://www.oracle.com/es/technologies/java/features/index.html> >

MINAG

2013

“Problemas en la agricultura”.

Consulta en: 29 de Agosto del 2013.

Disponible en: < <http://www.minag.gob.pe/portal/sector-agrario/agricola>>

NETBEANS

2013

“Netbeans IDE - The Smarter and Faster Way to Code”.

Consulta: 04 de Octubre del 2013.

Disponible en: < <https://netbeans.org/features/index.html#> >

OPTIFERT

2013

“Optifert Survey”

Consulta: 04 de Septiembre de 2013.

Disponible en: < <http://www.optifert.eu/> >

2013

“Riego Inteligente para un Crecimiento Verde”

Consulta: 04 de Septiembre del 2013.

Disponible en: < <http://es.euronews.com/2013/07/15/riego-inteligente-para-un-crecimiento-verde/> >

SAS/OR

2013

“Optimize business processes and address management science challenges with enhanced operations research methods”.

Consulta: 16 de Septiembre del 2013.

Disponible en: < http://www.sas.com/resources/factsheet/factsheet_or.pdf >

SH LEVI, Mark

2013

“Combinatorial Optimization: some main problems”.

Consulta: 03 Septiembre del 2013.

Disponible en: < <http://www.mslevin.iitp.ru/CO.HTM> >

SMT

2012

“The 2012 SMT Competition”.

Consulta: 31 de Octubre del 2014.

Disponible en: < <http://smt2012.loria.fr/SMT-COMP2012.pdf> >

2014

“The international SAT Competitions.”

Consulta: 31 de Octubre del 2014.

Disponible en: < <http://www.satcompetition.org/> >

WINQSB

2013

“Introducción al programa WINQSB”.

Consulta: 16 de Septiembre del 2013.

Disponible en: < <http://www.uv.es/martinek/material/WinQSB2.0.pdf> >