IMPLEMENTACIÓN DE UN ALGORITMO INMUNE ARTIFICIAL APLICADO EN EL ÁREA DE PLANIFICACIÓN DE RECURSOS

NELSON EDUARDO DIAZ DIAZ

LEYDY JOHANA LUNA MARTÍNEZ

UNIVERSIDAD INDUSTRIAL DE SANTANDER

FACULTAD DE INGENIERÍA FÍSICO-MECÁNICAS

ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA

BUCARAMANGA

2012

IMPLEMENTACIÓN DE UN ALGORITMO INMUNE ARTIFICIAL APLICADO EN EL ÁREA DE PLANIFICACIÓN DE RECURSOS

NELSON EDUARDO DIAZ DIAZ

LEYDY JOHANA LUNA MARTÍNEZ

Trabajo de Grado para optar al título de

Ingeniero de Sistemas

Directora

Msc. LOLA XIOMARA BAUTISTA ROZO

Codirectores

Ing. WILFREDO ARIEL GOMEZ BUENO

Ing. MANUEL IGNACIO CUADRADO MORAD

UNIVERSIDAD INDUSTRIAL DE SANTANDER

FACULTAD DE INGENIERÍA FÍSICOMECÁNICAS

ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA

BUCARAMANGA

2012

ENTREGA DE TRABAJOS DE GRADO, TRABAJOS DE INVESTIGACIÓN O TESIS Y AUTORIZACIÓN DE SU USO A FAVOR DE LA UIS

Yo, Leydy Johana Luna Martínez, mayor de edad, vecino de Bucaramanga, identificado con la Cédula de Ciudadanía No. 1.098.642.570 de Bucaramanga, actuando en nombre propio, en mi calidad de autor del trabajo de grado, del trabajo de investigación, o de la tesis denominada(o): "Implementación de un algoritmo inmune artificial aplicado en el área de planificación de recursos" hago entrega del ejemplar respectivo y de sus anexos de ser el caso, en formato digital o electrónico (CD o DVD) y autorizo a LA UNIVERSIDAD INDUSTRIAL DE SANTANDER, para que en los términos establecidos en la Ley 23 de 1982, Ley 44 de 1993, decisión Andina 351 de 1993, Decreto 460 de 1995 y demás normas generales sobre la materia, utilice y use en todas sus formas, los derechos patrimoniales de reproducción, comunicación pública, transformación y distribución (alquiler, préstamo público e importación) que me corresponden como creador de la obra objeto del presente documento. PARÁGRAFO: La presente autorización se hace extensiva no sólo a las facultades y derechos de uso sobre la obra en formato o soporte material, sino también para formato virtual, electrónico, digital, óptico, uso en red, Internet, extranet, intranet, etc., y en general para cualquier formato conocido o por conocer.

EL AUTOR - ESTUDIANTE, manifiesta que la obra objeto de la presente autorización es original y la realizó sin violar o usurpar derechos de autor de terceros, por lo tanto la obra es de su exclusiva autoría y detenta la titularidad sobre la misma. PARÁGRAFO: En caso de presentarse cualquier reclamación o acción por parte de un tercero en cuanto a los derechos de autor sobre la obra en cuestión, EL AUTOR / ESTUDIANTE, asumirá toda la responsabilidad, y saldrá en defensa de los derechos aquí autorizados; para todos los efectos la Universidad actúa como un tercero de buena fe.

Para constancia se firma el presente documento en dos (02) ejemplares del mismo valor y tenor, en Bucaramanga, a los 19 días del mes de Noviembre de Dos Mil Doce (2012).

EL AUTOR / ESTUDIANTE:

Leydy Johana Luna Martínez
C.C: 1.098.642.570 de Bucaramanga

ENTREGA DE TRABAJOS DE GRADO, TRABAJOS DE INVESTIGACIÓN O TESIS Y AUTORIZACIÓN DE SU USO A FAVOR DE LA UIS

Yo, Nelson Eduardo Diaz Diaz, mayor de edad, vecino de Bucaramanga, identificado con la Cédula de Ciudadanía No. 1.101.320.518 de Simacota, actuando en nombre propio, en mi calidad de autor del trabajo de grado, del trabajo de investigación, o de la tesis denominada(o): "Implementación de un algoritmo inmune artificial aplicado en el área de planificación de recursos" hago entrega del ejemplar respectivo y de sus anexos de ser el caso, en formato digital o electrónico (CD o DVD) y autorizo a LA UNIVERSIDAD INDUSTRIAL DE SANTANDER, para que en los términos establecidos en la Ley 23 de 1982, Ley 44 de 1993, decisión Andina 351 de 1993, Decreto 460 de 1995 y demás normas generales sobre la materia, utilice y use en todas sus formas, los derechos patrimoniales de reproducción, comunicación pública, transformación y distribución (alquiler, préstamo público e importación) que me corresponden como creador de la obra objeto del presente documento. PARÁGRAFO: La presente autorización se hace extensiva no sólo a las facultades y derechos de uso sobre la obra en formato o soporte material, sino también para formato virtual, electrónico, digital, óptico, uso en red, Internet, extranet, intranet, etc., y en general para cualquier formato conocido o por conocer.

EL AUTOR - ESTUDIANTE, manifiesta que la obra objeto de la presente autorización es original y la realizó sin violar o usurpar derechos de autor de terceros, por lo tanto la obra es de su exclusiva autoría y detenta la titularidad sobre la misma. PARÁGRAFO: En caso de presentarse cualquier reclamación o acción por parte de un tercero en cuanto a los derechos de autor sobre la obra en cuestión, EL AUTOR / ESTUDIANTE, asumirá toda la responsabilidad, y saldrá en defensa de los derechos aquí autorizados; para todos los efectos la Universidad actúa como un tercero de buena fe.

Para constancia se firma el presente documento en dos (02) ejemplares del mismo valor y tenor, en Bucaramanga, a los 19 días del mes de Noviembre de Dos Mil Doce (2012).

EL AUTOR / ESTUDIANTE:

Nelson E. Diaz D.
Nelson Eduardo Diaz Diaz
C.C: 1.101.320.518 de Simacota

DEDICATORIA

A mis padres, por darme la vida, su apoyo incondicional y ser una voz que me alentaba a continuar en momentos de dificultades.

A mis hermanos, Néstor, Julio Cesar, Norberto por estar siempre a mi lado compartiendo mis triunfos y siendo un soporte en momentos donde mi espíritu empezaba a flaquear.

A mis abuelos, quienes a pesar de su partida siempre estarán presentes en mis recuerdos. A mi abuela Flor María por cada grato momento que pase a su lado.

A mis familiares, por creer en mí y enseñarme valiosas lecciones de vida.

A mi compañera de proyecto, quien con su valiosa colaboración este proyecto hoy es una realidad.

A mis amigos, en especial a Carlos, por sus oportunas sugerencias. A Jehiel, Rodolfo, David, Luz Dary, María Fernanda, Héctor y todos mis colegas sistémicos por el tiempo compartido durante mi vida Universitaria. Andrés por su apoyo incondicional a lo largo del proyecto. A Félix por su voz de aliento y por ser colega en la profesión.

Nelson Eduardo Diaz Diaz

A Dios.

Por cumplir este gran sueño, por su amor eterno y por hacer de este proceso una oportunidad para mostrarme una vez más su favor y gracia.

A mi abuela

Porque aunque no este a mi lado, siempre tendré presente que fue ella quien más soñó con este día, por amarme y cuidarme todos los días, por ser más que mi madre, mi mejor amiga.

A mi familia

A mi tío, por motivarme a buscar siempre la excelencia y ser mi gran ejemplo. A todos mis familiares por respaldarme en cada una de mis etapas, por mostrarme siempre su apoyo y amor incondicional.

A mi novio

Por ser instrumento del amor de Dios en mi vida. Por su consejo, y comprensión en momentos difíciles. Por motivarme e impulsarme a dar siempre más de mi.

A mi compañero de proyecto

Por su dedicación, por ser paciente, perseverante y estar siempre dispuesto a darlo todo. Creo que conocerle como persona, fue uno de los mejores tesoros que me dejó este proyecto.

"Y ahora, que toda la gloria sea para Dios, quien puede lograr mucho más de lo que pudiéramos pedir o incluso imaginar mediante su gran poder, que actúa en nosotros"

Efesios 3:20

Leydy Johana Luna Martínez

AGRADECIMIENTOS

A la UNIVERSIDAD INDUSTRIAL DE SANTANDER y a la Escuela de Ingeniería de Sistemas, por la formación profesional brindada.

Al Grupo de Investigación en Ingeniería Biomédica GIIB y a quienes lo integran, en especial a los ingenieros Wilfredo Ariel Gomez Bueno y Manuel Ignacio Cuadrado Morad, por dedicar parte de su tiempo y darnos la asesoría e instrucción en los momentos que las necesitamos.

A la PhD(c).Lola Xiomara Bautista Rozo, profesora de la escuela y directora del GIIB por su dedicación y por compartir con nosotros su amplio conocimiento.

CONTENIDO

Pag.

LISTA DE FIGURAS

LISTA DE TABLAS

LISTA DE ANEXOS

RESUMEN

TÍTULO: IMPLEMENTACIÓN DE UN ALGORITMO INMUNE ARTIFICIAL APLICADO EN EL ÁREA DE PLANIFICACIÓN DE RECURSOS*

AUTORES: Nelson Eduardo Diaz Diaz - Leydy Johana Luna Martínez**

Palabras Clave: JSP, Algoritmos Inmunes, Clonalg, GRASP.

Resumen: Los algoritmos inmunes constituyen un área de investigación en continuo desarrollo y han probado ser de utilidad en la resolución de problemas de calendarización de tareas. El presente trabajo caracteriza el problema de Job Shop Scheduling y muestra la implementación de dos metaheurísticas como respuesta al problema: Enfoque inmune y Enfoque estocástico. En primer lugar se implementó el algoritmo inmune Clonalg, el cual se basa en los mecanismos que generan anticuerpos para la defensa contra los antígenos. Por otra parte se implementó el algoritmo estocástico llamado GRASP que utiliza una fase de construcción de solución factible al que luego se le aplica una fase de búsqueda local con el fin de mejorar la solución. Para obtener los resultados se llevó acabo primero un análisis de sensibilidad, el cual permitió hallar los valores de entrada de cada uno de los algoritmos que permitiera tener un mejor desempeño con respecto al makespan, es decir, que el tiempo que tardaran en terminarse todas las operaciones del calendario estuviera cerca al BKS (Best Known Solution o Mejor Solución Conocida) hallados en la literatura. Adicionalmente, se compararon los resultados obtenidos de los algoritmos, y se concluyó que de dichos resultados, los obtenidos del algoritmo de Clonalg presentan una menor dispersión y a su vez los resultados del makespan se acercan más al BKS en Clonalg que en GRASP. Finalmente, se planteó un escenario que permite optimizar el proceso de prueba para instancias de Lawrence del algoritmo Clonalg.

---

* Proyecto de grado. Modalidad: Trabajo de investigación.

** Facultad de Ingenierías Físico-Mecánicas. Escuela de Ingeniería de Sistemas e Informática. Director: Lola Xiomara Bautista Rozo. Co-directores: Wilfredo Ariel Gómez Bueno, Manuel Ignacio Cuadrado Morad

ABSTRACT

TITLE: USING AN IMMUNE ALGORITHM FOR JOB SHOP SCHEDULING PROBLEM*

AUTHORS: Nelson Eduardo Diaz Diaz - Leydy Johana Luna Martínez.**

Index Terms: JSP, Immune Algorithms, Clonalg, GRASP.

Abstract: The immune algorithms constitute an area of research and ongoing development which have proven to be useful in solving problems of scheduling tasks. This paper characterizes the Job Shop Scheduling problem and shows the implementation of two metaheuristics in response to the problem: immune algorithm and stochastic approach. First, the immune algorithm Clonalg was implemented, and is based on the mechanisms that generate antibodies to defend the body from antigens. Then the stochastic algorithm GRASP was implemented and it uses a construction phase of a feasible solution which then goes through a local search phase in order to improve the solution. To obtain the results a sensitivity analysis was performed which allowed the identification of input values of each of the algorithms that would perform better in terms of the makespan, in other words, that the time it takes to complete all calendar operations were close to the BKS (Best Known Solution) found in the literature. In addition, we compared the results of the algorithms, and concluded that the results obtained by Clonalg algorithm have lower dispersion in Clonalg than in GRASP, and at the same time makespan results are closer to the BKS in Clonalg. Finally, it was proposed a process that allows testing the Lawrence instances more efficiently.

---

* Degree Thesis Mode: Research Work.

** Faculty of Physics and Mechanics Engineering. School of Computing and Engineering Systems. Director: Lola Xiomara Bautista Rozo Co-director: Wilfredo Ariel Gómez Bueno, Manuel Ignacio Cuadrado Morad

GLOSARIO

ANTICUERPO: proteína (inmunoglobulina) que consiste en dos cadenas pesadas idénticas y dos cadenas ligeras idénticas; reconoce un epítopo particular en un antígeno y facilita la eliminación de ese antígeno. [1]

ANTÍGENO: cualquier sustancia (casi siempre ajena) que se une de manera específica con un anticuerpo o un receptor en la célula T; a menudo se emplea como sinónimo de inmunógeno. [1]

APOPTOSIS: cambios morfológicos relacionados con la muerte celular programada; incluyen fragmentación nuclear, formación de vesículas y liberación de cuerpos apoptósicos, que se fagocitan. En contraste con la necrosis, no causa daño a las células circundantes. [1]

BAZO: órgano linfoide secundario en el que los eritrocitos viejos se destruyen y los antígenos transportados en la sangre se capturan para presentarlos a los linfocitos de la vaina linfoide periateriolar y la zona marginal [1]

CÉLULAS DENDRÍTICAS: fagocitos en los tejidos que están en contacto con el ambiente externo; por lo tanto están localizados principalmente en la piel, la nariz, los pulmones, el estómago y los intestinos. Las células dendríticas actúan como enlace entre los sistemas inmunitarios innato y adaptativo, pues presentan antígenos a las células T, uno de los tipos de célula clave del sistema inmunitario adaptativo. [2]

CITOCINAS: cualquier de las numerosas proteínas de bajo peso molecular secretadas que regulan la intensidad y la duración de la respuesta

inmunitaria mediante diversos efectos en los linfocitos y otras células inmunitarias. [1]

DETERMINANTE ANTIGÉNICO: el sitio de un antígeno que se reconoce y al que se une un anticuerpo particular llamado epítopo. [1]

EPÍTOPO: porción de antígeno a la que reconoce y se une un anticuerpo o combinación de determinante antigénico. [1]

HEURÍSTICA: técnica o procedimiento práctico para resolver problemas [3].

IDIOTIPO: es el epítopo propio de una molécula perteneciente a un clon en particular. Este elemento forma parte o está muy próximo al lugar de reconocimiento del antígeno, y está situado en la porción variable Fab. En otras palabras, es la región cercana de una inmunoglobulina puede ser reconocida como un epítopo por ciertos linfocitos [4].

LECUOCITOS: glóbulo blanco. [1]

LINFOCITOS: leucocito mononuclear que media la inmunidad humoral o celular. [1]

LINFOCITOS B:tipo de linfocito que madura en la médula ósea y presenta anticuerpos en su membrana [1]

LINFOCITOS T: tipo de linfocito que maduran en el timo y es responsable de la respuesta inmune mediada por células [1]

MAKESPAN: tiempo total en el que todos los trabajos de una máquina completan su ejecución. [5]

META HEURÍSTICAS: son estrategias inteligentes para diseñar o mejorar procedimientos heurísticos muy generales con un alto rendimiento. Los procedimientos heurísticos utilizan conocimiento acerca de un problema y las técnicas aplicables, tratado de aportar soluciones (o acercarse a ellas) usando una cantidad de recursos (generalmente tiempo) razonable [6].

NECROSIS: cambios morfológicos que acompañan a la muerte de células individuales o grupos de células y que liberan grandes cantidades de componentes intracelulares al ambiente, lo que causa deterioro y atrofia del tejido. [1]

OPSONIZACIÓN: depósito de opsoninas sobre un antígeno que favorece el contacto adhesivo estable con una célula fagocítica apropiada. [1]

PATÓGENO: organismo causante de enfermedad. [1]

RED INMUNE: red computacional que asume características del sistema inmune humano

SISTEMA INMUNOLÓGICO: conjunto de estructuras y procesos biológicos en el interior de un organismo que le protege contra enfermedades identificando y matando células patógenas y cancerosas. Detecta una amplia variedad de agentes, desde virus hasta parásitos intestinales y necesita distinguirlos de las propias células y tejidos sanos del organismo para funcionar correctamente [7].

TIMO: órgano linfoide primario localizado en la cavidad torácica, donde ocurre la maduración de las células T. [1]

TOLERANCIA INMUNOLÓGICA: característica fundamental del sistema inmune es la de no reaccionar frente a los componentes propios del

individuo, aun cuando posee la cualidad de responder frente a cualquier antígeno extraño al mismo. Esta capacidad de reconocimiento y aceptación de los componentes propios del organismo se debe al fenómeno de tolerancia inmunológica [8].

ACRÓNIMOS

AIS: Artificial Immune System

NIS: Natural Immune System

JSP: Job Shop Scheduling Problem

NKC: Natural Killer Cell

GRASP: Greedy Randomized Algorithm Search Procedure

BKS: Best Known Solution

INTRODUCCIÓN

Los problemas de calendarización[1] aparecen constantemente en la vida real en numerosos ambientes productivos y de servicios, se tratan de problemas en los cuales se requieren organizar durante el tiempo, la ejecución de trabajos que compiten por el uso de un conjunto finito de recursos y que están sujetos a un conjunto de restricciones impuestas por factores como las características físicas del entorno, relaciones temporales o la normativa laboral. Además se trata de optimizar uno o varios criterios que se representan mediante funciones objetivo y que están relacionados normalmente con el costo, el beneficio o el tiempo de ejecución. El objetivo puede consistir en maximizar la utilización de algunas máquinas que son cuello de botella o minimizar el tiempo de ejecución.

Dado que estos problemas son de naturaleza combinatoria, es decir que hay que elegir una entre un conjunto exponencialmente grande de combinaciones posibles, los problemas de calendarización precisan de algoritmos de búsqueda inteligentes para encontrar soluciones aceptables en un tiempo razonable. Por lo que en la literatura se pueden encontrar aproximaciones a los problemas de calendarización basadas en prácticamente todas las heurísticas conocidas y en particular en los algoritmos de búsqueda heurística propios de áreas como la Investigación Operativa y la Inteligencia Artificial.

Esta tesis se centra en el problema Job Shop Scheduling y en el Procedimiento de Búsqueda Miope Aleatorizado y Adaptativo GRASP y en el Algoritmo Inmune Artificial de selección clonal, cuyo objetivo es trazar estrategias que resulten eficaces y eficientes para una función objetivo[2], que permita obtener soluciones aproximadas para diferentes instancias. La función objetivo a la que

[1] Programación de tareas o asignación de trabajos.

[2] Función a optimizar (maximizar o minimizar) que esta sujeta a restricciones.

los investigadores en el mundo han prestado mayor atención es sin duda el *makespan*, o tiempo de finalización de la última tarea. En el presente trabajo se utilizó la representación de permutación con repetición con el fin de representar la asignación de los trabajos, y se pudo comprobar la viabilidad del algoritmo propuesto.

Este trabajo se encuentra organizado de la siguiente manera: en el primer capítulo se hace una breve descripción del proyecto, en el cual se plantea el problema a resolver, los objetivos a alcanzar, la viabilidad y el impacto que tiene el desarrollo de este proyecto.

En el segundo capítulo se presenta una revisión del marco teórico de los sistemas inmunológicos, una visión general de los sistemas inmunes artificiales y se presentan algunos de los algoritmos que se han desarrollado sobre los principios del sistema inmune natural. Por último se plantea el problema de Job Shop Scheduling y se hace una revisión de las diferentes técnicas empleadas para abordar estos problemas.

A continuación en el capítulo 3 se describe la metodología de desarrollo y se plantean las diferentes tareas necesarias para llevar a cabo el cumplimiento de los objetivos.

El capítulo 4 hace la descripción de la técnica y la representación del problema. En el capítulo 5 se muestran los resultados de la implementación de los algoritmos propuestos en las diferentes instancias de Lawrence para el problema de Job Shop Scheduling, también se comparan los resultados obtenidos por los algoritmos de Clonalg y GRASP con respecto a la mejor solución encontrada en la literatura.

Por último en el capítulo 6, las conclusiones de nuestro trabajo, así como recomendaciones para trabajos futuros.

1 DESCRIPCIÓN DEL PROYECTO

1.1 PLANTEAMIENTO DEL PROBLEMA

En las últimas décadas los científicos de la computación se han valido de modelos extraídos de sistemas biológicos para elaborar técnicas conocidas como meta heurísticas. Muchas de las meta heurísticas se preocupan por resolver problemas difíciles en términos computacionales. En este estudio se consideró un problema combinatorio, conocido como Job Shop Scheduling Problem (JSP) que es un problema común de calendarización en donde se cuenta con ciertos recursos (máquinas) que deben ser asignados a ciertos trabajos los cuales se componen de operaciones.

El problema consiste en encontrar un calendario que utilice de manera óptima los recursos disponibles en el menor tiempo posible. La razón por la cual este problema resulta de gran interés es su complejidad algorítmica (NP-duro) y su uso común en diferentes aplicaciones de la industria de hoy [9].

En la actualidad una de las áreas más dinámicas en la inteligencia artificial es la que se refiere a los sistemas inmunes artificiales. Esto se debe en gran parte a que este tipo de técnicas ofrecen ventajas en su implementación computacional. Por tanto, se escogió este tipo de técnica para comprobar qué mejoras pueden tener en instancias de problemas de calendarización. Se realizó un contraste frente a técnicas encontradas en la literatura actual a fin de verificar su desempeño [10]

1.2 JUSTIFICACIÓN

El presente trabajo de investigación hace parte del trabajo de maestría, "IMPLEMENTACIÓN DE ALGORITMOS BIOINSPIRADOS PARA LA SOLUCIÓN DEL PROBLEMA DE PLANIFICACIÓN DE TRABAJOS" que está siendo desarrollado por el Ingeniero Wilfredo Ariel Gómez.

El aporte específico de este trabajo de grado se sintetiza en el desarrollo de uno de los objetivos específicos del correspondiente trabajo de maestría. Este objetivo aborda específicamente las técnicas inmune inspiradas y plantea realizar una implementación a priori de la técnica. Este trabajo entregará un prototipo funcional con el que se partirá para realizar el proceso de pruebas de desempeño de la técnica inmune frente al resto de técnicas bioinspiradas contempladas en ese desarrollo.

Adicionalmente contempla hacer un análisis sobre las instancias del problema de Planificación para entender posibles escenarios de aplicación, y dejar pautas iniciales para una adecuada implementación en problemas reales relacionados con la computación y las comunicaciones.

Dado que existe un vasto panorama de técnicas convencionales para la solución de estos problemas [11], se pretende explorar nuevas alternativas que han tenido un uso creciente en el área de optimización con restricciones [5], como lo son los sistemas inmunes artificiales, con el fin de encontrar aproximaciones eficientes a las soluciones de dichos problemas.

1.3 OBJETIVOS

1.3.1 OBJETIVO GENERAL

Implementar un algoritmo inmune artificial para resolver instancias de un problema de planificación de recursos y compararlo frente a otras técnicas encontradas en la literatura.

1.3.2 OBJETIVOS ESPECÍFICOS

- Implementar un algoritmo inmune sobre la familia de instancias LA[3] del problema de JSP que presenta gran uso en la literatura.

- Evaluar el desempeño del algoritmo implementado frente a un algoritmo enumerativo GRASP sobre la familia de instancias LA del problema de Job Shop Scheduling.

- Caracterizar posibles escenarios de aplicación del problema JSP en el campo de la computación basados en la familia de instancias utilizadas.

[3] Instancias de Lawrence (instancias LA) : conjunto de funciones de prueba que se utilizarán para medir la eficiencia de nuestro algoritmo

1.4 VIABILIDAD DEL PROYECTO

El Grupo de investigación en Ingeniería Biomédica (GIIB) cuenta con personal idóneo en el área de algoritmos inmunes artificiales y problemas combinatorios. Así como estudios previos en estas dos áreas. Esto garantiza asesoría y acompañamiento necesario para la puesta en marcha, desarrollo e implementación del estudio planteado.

El desarrollo utilizará herramientas libres lo cual reduce la inversión total del proyecto debido al ahorro en gastos de licencias y permite tener acceso a lenguajes de programación, IDE´s, manejadores bases de datos, lenguajes Script, así como también frameworks. Estas herramientas permitirán implementar los algoritmos de tipo inmune aplicados a instancias de un problema combinatorio, lo cual es parte fundamental del presente estudio.

1.5 IMPACTO

El presente proyecto tendría un impacto positivo en el Grupo de investigación en Ingeniería Biomédica (GIIB) dado que se estudiará un tema actual en el campo de la inteligencia artificial como lo es la abstracción de algoritmos a partir del sistema inmune de los mamíferos. Esto permitirá a los estudiantes del grupo tener un estudio adicional que se encontrará a disposición y que servirá de soporte a la hora de abordar estas temáticas en futuros estudios.

En la actualidad este proyecto pertenece a un conjunto de proyectos que se están desarrollando de manera paralela sobre diferentes técnicas bioinspiradas para apoyar el trabajo de maestría del Ingeniero Wilfredo Ariel Gómez.

2 MARCO TEÓRICO

2.1 SISTEMA INMUNE NATURAL (NIS)

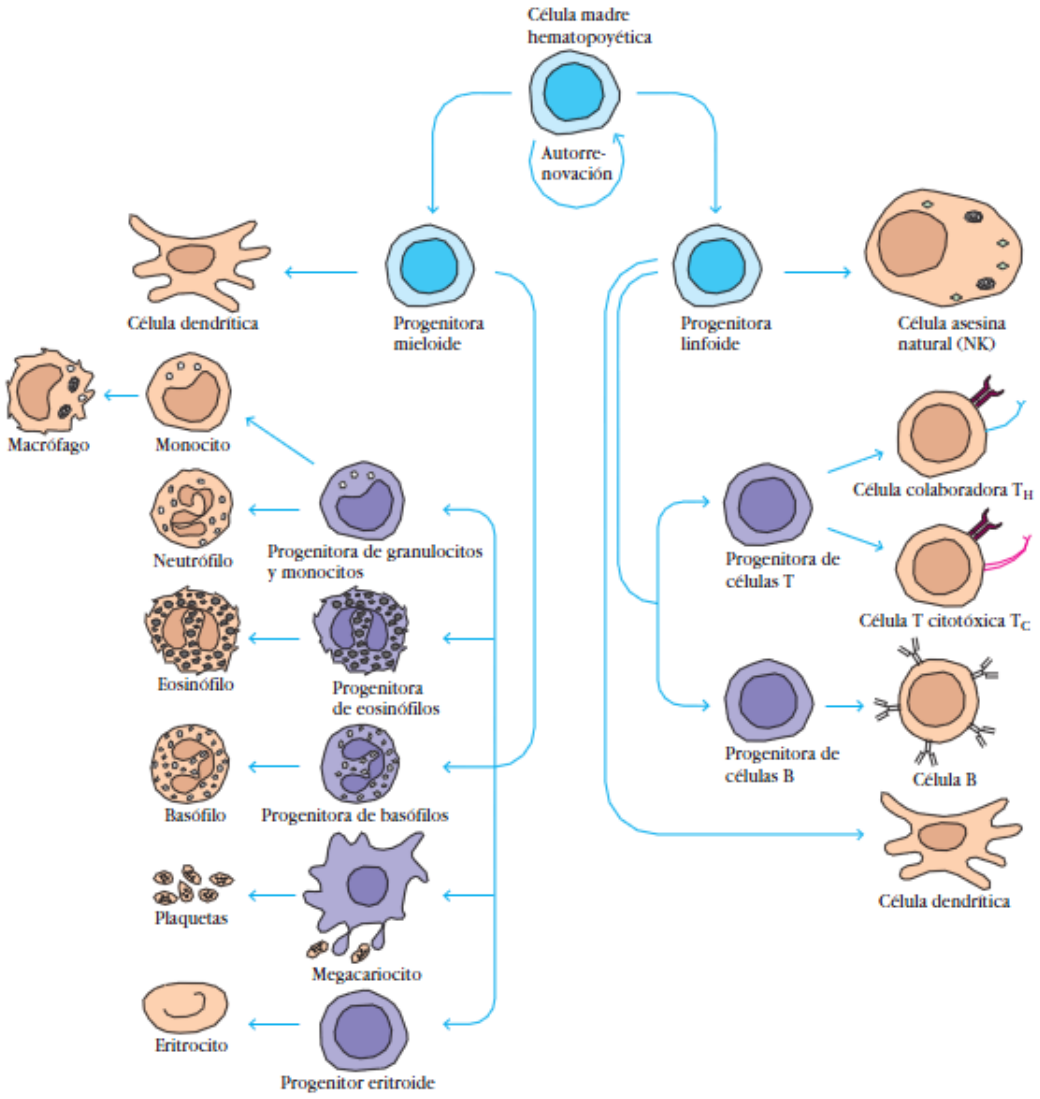
El Sistema Inmune Natural es un sistema complejo, robusto y adaptativo que defiende al organismo de patógenos extraños. Es capaz de clasificar todas las células (o moléculas) dentro del cuerpo como células propias o células no propias (ver Figura 1).

Lo hace con la ayuda de un grupo de trabajo distribuido que tiene la inteligencia para tomar acción de un local y también una perspectiva global utilizando su red de mensajeros químicos para la comunicación [12].

Hay dos ramas principales del sistema inmunológico: El sistema inmune innato el cual es un mecanismo inmutable que detecta y destruye ciertos organismos invasores y el sistema inmune adaptativo [13] que responde a las células extrañas desconocidas previamente y construye una respuesta a ellas la cual puede permanecer en el cuerpo durante un largo período de tiempo.

Este sistema de procesamiento de la información biológica notable ha llamado la atención de la informática en los últimos años, por lo que una nueva técnica de inteligencia computacional ha surgido inspirada en la inmunología, y se conoce como sistemas inmunes artificiales. Varios conceptos de la inmunología han sido extraídos y aplicados a la solución real del mundo de la ciencia y la ingeniería.

Figura 1: Clasificación de las células del sistema inmune



Fuente: Kindt, Goldsby y Osborne [1]

El sistema inmune biológico es un sistema de defensa elaborado que ha evolucionado durante millones de años. Aunque muchos detalles de los mecanismos de inmunidad (innata y adaptativa) y procesos (humoral y celular) son todavía desconocidos (incluso para los inmunólogos) [12], es sin embargo bien conocido que el sistema inmunológico utiliza defensa multinivel [14] (y la superposición) tanto en paralelo como de manera secuencial.

Dependiendo del tipo de patógeno, y la forma en que se introduce en el cuerpo, el sistema inmune utiliza diferentes mecanismos de respuesta (vías diferenciales) ya sea para neutralizar el efecto patógeno o para destruir las células infectadas.

Una descripción detallada del sistema inmune se puede encontrar en muchos libros de texto, tales como Kuby [1].

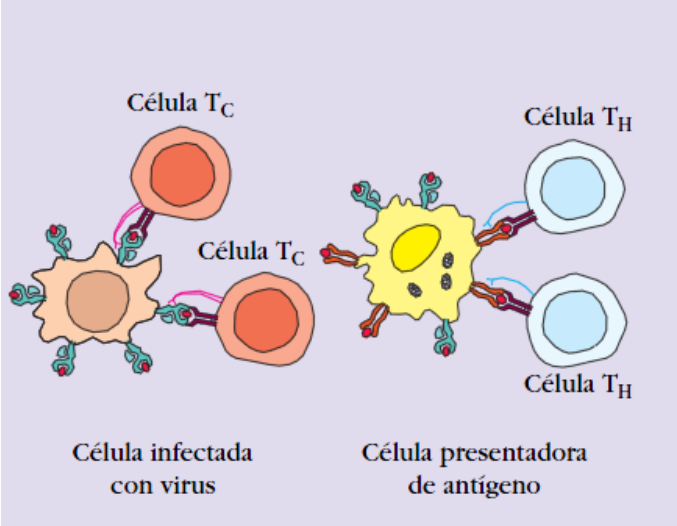
Las características inmunológicas que son particularmente relevantes en el sistema inmune son: el reconocimiento, la diversidad y el control distribuido.

El reconocimiento se refiere a la unión entre anticuerpos y antígenos. La diversidad se refiere al hecho de que, con el fin de lograr una cobertura óptima de espacios de antígenos, la diversidad de anticuerpos debe ser estimulada [15]. El control distribuido significa que no hay un controlador central; más bien, el sistema inmune se rige por las interacciones locales entre las células inmunes y los antígenos.

Dos de las células más importantes en este proceso son los glóbulos blancos, también llamadas células T y células B. Ambas se originan en la médula ósea, pero las células T se transmiten al timo para madurar antes de distribuirse en los vasos sanguíneos y linfáticos (Ver Tabla 1).

Las células T son de tres tipos: *células T auxiliares* que son esenciales para la activación de las células B, *células T asesinas* que se unen a los invasores extranjeros e inyectan productos químicos venenosos en ellos que causan su destrucción (Ver Figura 2), y las *células T supresoras* que inhiben la acción de otras células inmunes evitando así reacciones alérgicas y enfermedades autoinmunes.

Figura 2: Reconocimiento del antígeno por el linfocito T



Fuente: Kindt, Goldsby y Osborne [1]

Por otra parte las células B son responsables de la producción y secreción de anticuerpos, que son proteínas específicas que se unen al antígeno, el cual se encuentra en la superficie del organismo invasor; la unión de un anticuerpo al antígeno es una señal para destruir la célula invasora. Cada célula B sólo puede producir un anticuerpo particular.

A continuación se muestra una tabla comparativa (ver tabla 1) que permite evidenciar de una manera práctica las diferencias y semejanzas entre las células B y T.

Tabla 1: Comparación del reconocimiento de antígeno por células T y B

CARACTERÍSTICA	CÉLULAS B	CÉLULAS T
Interacción con antígeno	Incluye complejo binario de membrana Ig y Ag	Implica complejo temario de receptos de célula T, Ag y molécula MHC.
Unión de antígeno soluble	Si	No
Partición de moléculas MHC	No se requiere	Se requiere para exhibir antígeno procesado
Naturaleza química de los antígenos	Proteína, polisacárido, lípido	Sobre todo proteínas, pero también algunos lípido y glucolípidos presentados en moléculas parecidas a MHC
Propiedades del epítopo	Accesible, hidrófilo, péptidos móviles que contienen aminoácidos secuenciales o no secuenciales	Péptidos lineales internos producidos por el procesamiento de antígeno y unidos a moléculas MHC

Como se mencionó anteriormente, el cuerpo humano está protegido contra invasores extraños por un sistema de varias capas. El sistema inmune está compuesto de barreras físicas tales como la piel y el sistema respiratorio, las barreras fisiológicas, tales como enzimas destructivas y los ácidos del estómago.

El sistema inmune, que puede ser considerado de dos tipos: inmune innato (no específico) e inmune adaptativo (específico).Estos están vinculados entre sí y se influyen mutuamente (Ver Tabla 2).

Tabla 2: Comparación entre Inmunidad Innata y Adaptativa

INMUNIDAD INNATA	INMUNIDAD ADAPTATIVA
Respuesta de antígeno independiente	Respuesta de antígeno dependiente
Respuesta inmediata	Respuesta retardada (Fase de latencia)
Antígeno no específico	Antígeno específico
No tiene memoria inmunológica	Memoria inmunológica

2.1.1.1 RED INMUNITARIA

Aunque existe más de un mecanismo en el trabajo del sistema inmune, el proceso esencial es la adecuación del antígeno y el anticuerpo, que conduce a mayores concentraciones (proliferaciones) de más anticuerpos.

En particular, la teoría de redes idiotípicas, el mecanismo de selección negativa, y las teorías de "selección clonal" e "hipermutación somática" se utilizan principalmente en modelos artificiales del sistema inmunológico [1].

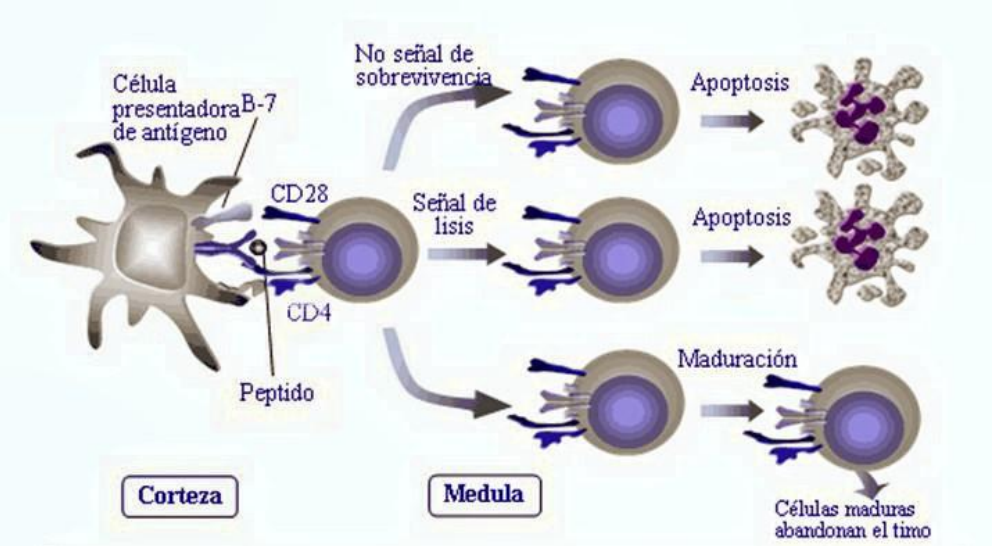
La teoría de la red inmunitaria fue propuesta por Jerne [16], la hipótesis era que el sistema inmune mantiene una red de idiotípos de células B interconectadas para el reconocimiento de antígenos. Estas células estimulan y suprimen a las otras en ciertas maneras que conducen a la estabilización de la red.

2.1.1.2 MECANISMO DE SELECCIÓN NEGATIVA

El propósito de la selección negativa es proporcionar tolerancia para células propias. Se trata de la capacidad del sistema inmunitario para detectar antígenos desconocidos mientras no reacciona o no destruya a sus propias células (Ver Figura 3). Durante la generación de células T, los receptores se hacen a través de un proceso de reordenamiento pseudo-aleatorio genético.

A continuación, se someten a un proceso de censura en el timo, el llamado selección negativa. Allí, las células T que reaccionan contra proteínas propias se destruyen, por lo que sólo aquellos que no se unen a proteínas propias se les permiten abandonar el timo. Estas células T maduras entonces circulan por todo el cuerpo para realizar las funciones inmunológicas y proteger el cuerpo contra los antígenos extraños.

Figura 3: Selección Negativa



Fuente: Galiani, Santamaría y Peña [17]

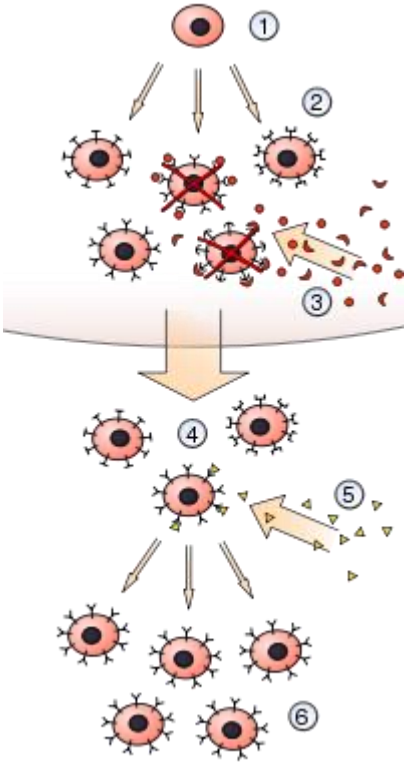
2.1.1.3 PRINCIPIO DE SELECCIÓN CLONAL

En el principio de selección clonal se describen las funciones básicas de una respuesta inmune a un estímulo antigénico. Se establece la idea de que sólo aquellas células que reconocen el antígeno proliferan, por lo tanto siendo

seleccionado en contra de aquellos que no lo hacen. Las características principales de la teoría de la selección clonal son:

a) Las nuevas células son copias de sus padres (clon) sometidos a un mecanismo de mutación (hipermutación somática)
b) Eliminación de linfocitos recién diferenciados que llevan receptores auto-reactivos
c) Proliferación y diferenciación de células maduras en contacto con antígenos

Figura 4: Proceso de Selección Clonal



Fuente: Karonen [18]

El proceso de selección clonal se lleva acabo de la siguiente manera: (1) la célula madre hematopoyética experimenta diferenciación y reorganización genética (2) para producir linfocitos inmaduros con muchos receptores de antígenos diferentes. (3) Aquellos que se unen antígenos de los propios tejidos del cuerpo son destruidos, (4) mientras que el resto se maduran en linfocitos inactivos. (5) La mayoría de ellos nunca se encontrará una coincidencia antígeno extraño, (6) pero los que lo hacen se activan y producen muchos clones de sí mismos.

La selección clonal proporciona un marco estructural para comprender la especificidad y el reconocimiento de lo propio y lo extraño que es característico de la inmunidad adaptativa. La especificidad se demuestra porque sólo linfocitos cuyos receptores son específicos para un epítopo determinado en un antígeno se expanden clonalmente y por tanto se movilizan para oponer una reacción inmunitaria.

La discriminación de lo propio y lo extraño se efectúa por la eliminación, durante el desarrollo, de linfocitos que llevan receptores auto reactivos o por la supresión funcional de estas células si alcanzan la madurez.

La memoria inmunitaria es otra consecuencia de la selección clonal. Durante ésta se amplifica en grado considerable el número de linfocitos específicos para un antígeno determinado.

2.2 SISTEMAS INMUNES ARTIFICIALES (AIS)

Los sistemas inmunes en la última década, han servido de inspiración para solucionar problemas computacionales complejos, puesto que cuentan con características como las de ser sistemas distributivos y adaptativos de naturaleza auto-organizativa junto a su memoria, aprendizaje, reconocimiento de patrones y capacidad de extracción.

El campo de los Sistemas Inmunológicos Artificiales son cada vez más populares [19], los comienzos de éste se dieron a mediados de los 80's con Farmer, Packard y Perelson's en [20] y Bersini y Varela en 1990 [21] tratando problemas de redes inmunes. En ese mismo año Ishida propone el primer algoritmo de redes inmunes [22], sin embargo fue solo a mediados de los 90's donde los Sistemas Artificiales Inmunes AIS se convirtieron en un tema importante.

En1994Forrest et al [23] propusieron un método llamado algoritmo de *selección negativa*, que está basado en la generación de células T en el sistema inmune. Este método fue aplicado a los problemas de detección de virus de computadores.

Durante 1995 Hunt y Cooke [24] continuaron trabajando sobre modelos de red inmune en el cual Timmis y Neal [25] hicieron algunas mejoras redefiniendo y re-implementado este algoritmo. Estas obras fueron llamadas formalmente AINE (Artificial Immune Network).

En el año 2000 Castro y Zuben [26] propusieron el algoritmo de selección clonal, más tarde conocido como Clonalg, que está inspirado en la selección clonal de los sistemas inmunes la cual explica cómo los linfocitos B y T mejoran su respuesta ante los antígenos. A este tiempo se le conoce como tiempo de maduración de la afinidad [27].

En 2008, Dasgupta y Niño [28] presentaron un panorama de los conceptos fundamentales de la inmunología y algunos modelos inmunológicos teóricos de los procesos inmunes. También se presenta un compendio de los trabajos relacionados hasta esa fecha basadosen técnicas de inmunología computacional y describe una amplia variedad de aplicaciones.

2.2.1 ALGORITMOS INSPIRADOS EN EL SISTEMA INMUNE

2.2.1.1 SELECCIÓN CLONAL

La teoría de la selección clonal ha sido usada como inspiración para el desarrollo de Sistemas Inmunes Artificiales que realizan optimización computacional y tareas de reconocimiento de parámetros.

En particular, la inspiración tomada de los antígenos que manejan el proceso de maduración de la afinidad de las células B, con su mecanismo asociado de hipermutación [29]. Estos Sistemas Inmunes Artificiales también utilizan a menudo la idea de células de memoria para mantener buenas soluciones al problema que está siendo resuelto.

En el libro de Castro y Timmis [25] ellos resaltan dos importantes características de maduración de afinidad en células B que pueden ser explotadas desde el punto de vista computacional. El primero de estos es que la proliferación de células B es proporcional a la afinidad de los antígenos que los unen, por tanto la alta afinidad produce más clones. Segundo, la mutación sufrida por los anticuerpos de una célula B es inversamente proporcional a la afinidad de los antígenos que los unen.

Utilizando estas características, Castro y Von Zuben [26] desarrollaron un algoritmo inspirado en Sistemas Inmunes Artificiales basados en selecciones clónales, el cuál ha sido utilizado para realizar tareas de emparejamiento de patrones y optimización de funciones multimodal [30].Este algoritmo se conoce como Clonalg y su pseudocódigo se muestra a continuación:

Algoritmo 1: Clonalg **[31]**

$$\textbf{\textit{Input}}: Population_{size}, Selection_{size}, Problem_{size}, RandomCells_{num}, Clone_{rate}, Mutation_{rate}$$

$$\textbf{\textit{Output}}: Population$$

$Population \leftarrow CreateRandomCells(Population_{size}, Problem_{size});$
$\boldsymbol{While}\ \neg StopCondition(.)\boldsymbol{do}$
$\boldsymbol{foreach} p_i \in Population\ \boldsymbol{do}$
$Affinity\ (p_i);$
$\boldsymbol{end}$
$Population_{select} \leftarrow Select(Population, Selection_{size});$
$Population_{clones} \leftarrow \emptyset;$
$\boldsymbol{foreach} p_i \in Population_{select} \boldsymbol{do}$
$Population_{clones} \leftarrow Clone(p_i, Clone_{rate});$
$\boldsymbol{end}$
$\boldsymbol{foreach}\ p_i \in Population_{clones}\ \boldsymbol{do}$
$Hypermutate(p_i, Mutation_{rate});$
$Affinity(p_i);$
$\boldsymbol{end}$
$Population \leftarrow Select(Population, Population_{clones}, Population_{size});$
$Population_{rand} \leftarrow CreateRandomCells(RendomCells_{num});$
$Replace(Population, Population_{rand});$
$\boldsymbol{end}$
$return\ Population;$

El algoritmo Clonalg se inspira en la teoría de selección clonal que establece la idea de que sólo aquellas células (anticuerpos) que reconocen los antígenos proliferan, a la generación de copias (clones).

Posteriormente, las mutaciones genéticas se producen solo en dichos clones. Solo la generación de clones (anticuerpos) que tengan mayor afinidad para el antígeno se convierten en células de memoria, lo que implica que se tendrán individuos mejor adaptados, más rápidos y eficientes para futuras respuestas ante antígenos similares.

La aplicación del algoritmo Clonalg involucra cuatro decisiones clave: codificación de anticuerpos y antígenos, la definición de la medida de afinidad entre anticuerpos y antígenos, y la configuración de los procesos de selección y mutación.

La aplicación se mueve a través de una población de anticuerpos $Ab_i\ i = 1, \dots, n$ cada uno representando una solución candidata (ejemplo: c centros de clúster), y un conjunto de elementos a ser agrupados, representados por la población de antígenos $Ag_j, j = 1, \dots, m.$ La población antígenos está representado por un arreglo Ag_{ml}, donde m es la cantidad de antígenos y l el número de características de cada uno. La población de anticuerpos también se representa mediante un arreglo Ab_{nkl}, donde n es la cantidad de anticuerpos, k el número de grupos y l el número de características.

La afinidad f entre anticuerpos y antígenos está dada por la siguiente ecuación:

$$f = 1 - \frac{makespan}{rango}$$

donde el rango está dado por:

$$rango = \max(makespan) - \min(makespan)$$

La cantidad n de clones para cada anticuerpo corresponde a un parámetro del algoritmo, el cual está dado por la ecuación:

$$N_c = (\beta * N)$$

donde β es un factor de clonación y Nes la cantidad de anticuerpos.

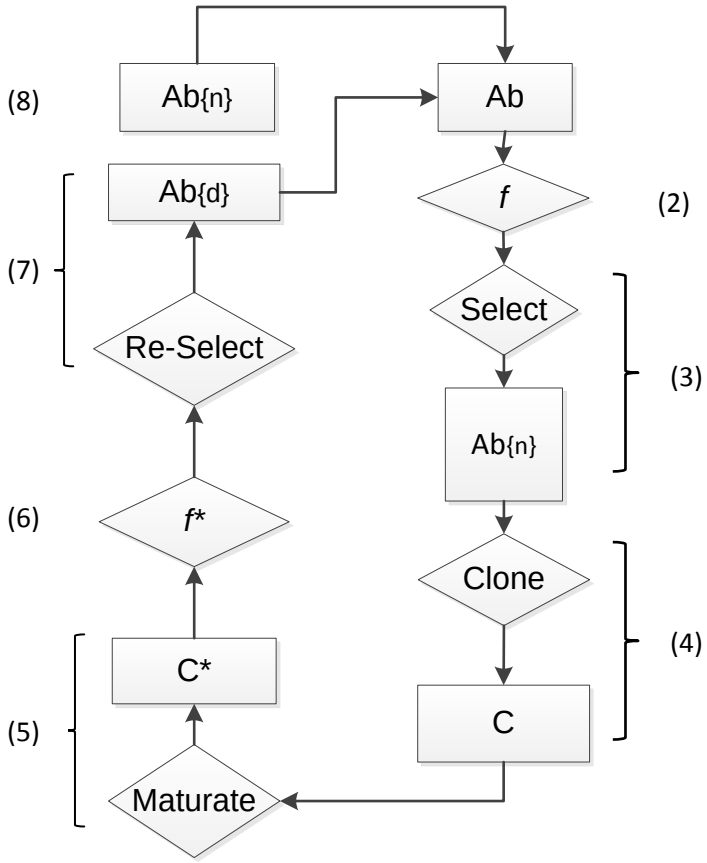
Los clones generados sufren un proceso de mutación la cual es inversamente proporcional a la afinidad del antígeno (entre mayor afinidad, menor es la tasa de mutación), dicha mutación esta dada por:

$$p = e^{(-\rho * f)}$$

donde, ρ es el factor de mutación y f la afinidad.

El algoritmo Clonalg presentado en esta tesis está basado en la versión propuesta por Castro [27] el cual se muestra en el diagrama de flujo de la Figura 5.

Figura 5: Diagrama de flujo de Clonalg



Fuente: Castro y Zuben [26]

El algoritmo empieza con la generación de la población. Cada individuo de la población corresponde a una solución candidata, que lleva acabo cada uno de los siguientes pasos:

1. Un conjunto de antígenos Ag es presentado a la población de anticuerpos Ab;

2. Se calcula la afinidad f de los anticuerpos en relación a los antígenos;
3. Los anticuerpos con afinidad más alta son seleccionados para ser clonados, generando el subconjunto de anticuerpos $Ab_{\{n\}}$;
4. Los anticuerpos seleccionados se clonarán de acuerdo a la afinidad con los antígenos (entre más alta la afinidad, se generarán más clones), produciendo un población C de clones.
5. La población Cde clones se somete a un proceso de maduración de la afinidad (entre más alta la afinidad, menor la tasa de mutación), produciendo entonces una nueva población de clones C^*.
6. La población de clones C^* es evaluada, y su afinidad f^* es calculada en relación a los antígenos.
7. Luego se seleccionan de los n anticuerpos maduros que tengan mayor afinidad que serán la siguiente población, siempre y cuando la afinidad sea mayor que la de los anticuerpos originales.

Los peores d anticuerpos son removidos y remplazados por una nueva población generada aleatoriamente.

2.3 JOB SHOP SCHEDULING

El Problema de Job Shop Scheduling[4] consiste en tener un conjunto m de trabajos y un conjunto n de máquinas en el cual cada trabajo tiene un número de operaciones y una secuencia en particular de máquinas que se tiene que seguir. Aquí cada operación sólo puede hacerse en una máquina es decir, cada máquina puede realizar una sola operación a la vez, lo que hace que este sea uno de los problemas más difíciles de optimización combinatoria.

[4] Se puede traducir como *el problema de asignación de trabajos, aquí se usan las siglas en inglés* **JSP**

Durante las últimas décadas muchos investigadores han trabajado el problema de Job Shop Scheduling derivando en muchas soluciones. A continuación se presentan diferentes trabajos donde se comparan los Sistemas Inmunes Artificiales ante otras heurísticas.

En el trabajo de Mahdi Mobini et al. [15] se aborda el uso de un sistema inmune artificial de Selección Clonal en el cual se tiene en cuenta la minimización de makespan como objetivo para la solución de Job Shop Scheduling. Los resultados obtenidos en el análisis computacional y la comparación con otros algoritmos genéticos tales como Búsqueda de dispersión, Algoritmo genético híbrido entre otros, revelan un rendimiento aceptable y competitivo del algoritmo propuesto en dicho estudio.

El trabajo de Castro et al. [16] presenta la adaptación de un modelo de red inmune diseñado especialmente para resolver problemas de optimización multimodales. Se comparó teóricamente con un algoritmo de selección clonal también aplicado para realizar la optimización multimodal, y estrategias de evolución.

En el trabajo de Coello et al. [5] se propone un algoritmo basado en un sistema inmune artificial (Selección Clonal) para solucionar Job Shop Scheduling. El enfoque utiliza otros 3 algoritmos para ser comparados con el Algoritmo de Sistema Inmune Artificial (AIS), los cuales son: Algoritmo Genético Hibrido (HGA), Algoritmo Genético Paralelo (PGA) y GRASP. Los resultados indican que el método propuesto es muy competitivo con respecto a los otros, ya que presenta una mejora del 0.23%, 0.74% y 0.28% del AIS con respecto a los algoritmos HGA, PGA y GRASP.

En el desarrollo de Quan Zuo y Shun Fan [18] se presenta un algoritmo inmune artificial (Selección Clonal) para resolver el Problema de Job Shop Scheduling JSP. El algoritmo inmune utiliza la tecnología de nicho para evitar óptimos locales y cuenta con sistema de caos para mejorar la eficiencia de búsqueda.

Los resultados experimentales demuestran la eficacia del método para resolver problemas de programación de tareas puesto a medida que se reduce el retardo también se reducen las iteraciones para encontrar la mejor solución

Cuando el retardo es de 200 el algoritmo necesita 39.5 iteraciones para encontrar la mejor solución, sin embargo si el retardo es de 10 el algoritmo se optimiza a 2,2 iteraciones.

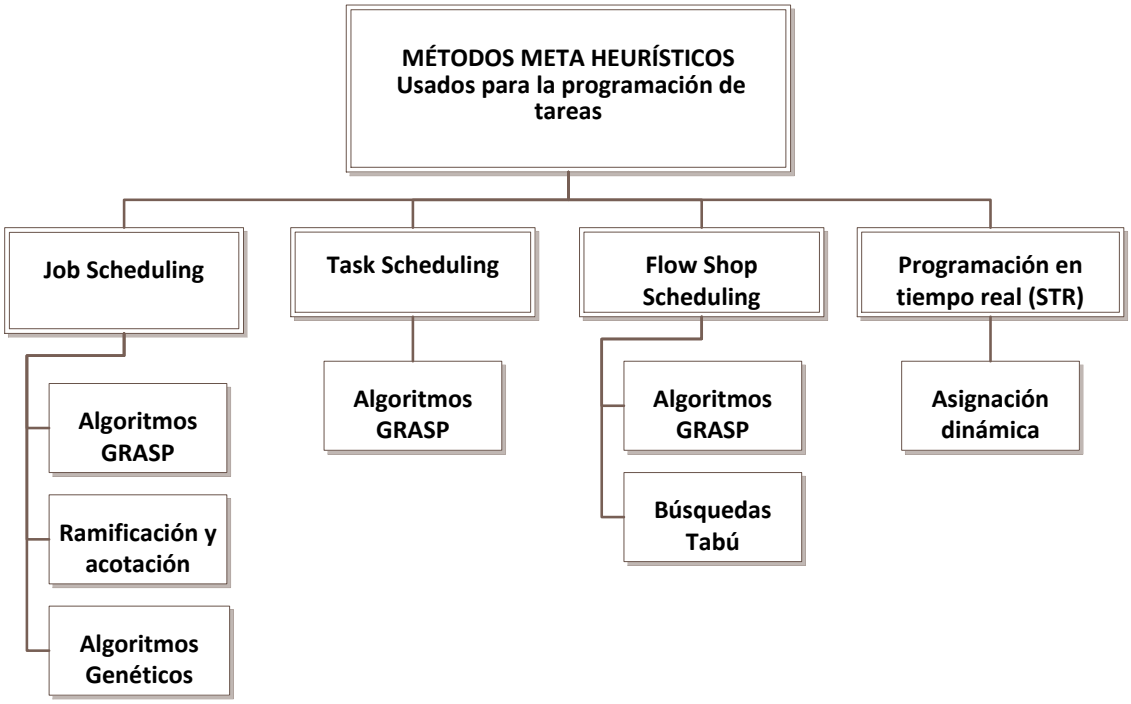
El trabajo de Chandrasekaran et al.[19], presenta un Algoritmo inmune de Selección Clonal, el cual se ha usado para resolver problemas de programación de tareas JSP con el objetivo de lograr la minimización del makespan. Los resultados son comparados con los algoritmos de Búsqueda Tabú y Procedimiento de Desplazamiento de Cuello de Botella (por sus siglas en inglés SBGLS1).

Al compararse dichos algoritmos se obtiene un porcentaje de error relativo, el cual muestra que el algoritmo inmune tiene el menor porcentaje con 1,865 mientras que Búsqueda Tabú tiene 2,56 y Cuello de Botella 3,68 mostrando así que la Selección clonal genera mayor cantidad de valores óptimos.

El trabajo de Cortés et al.[50], propone un algoritmo basado en un sistema inmune artificial de Selección Clonal para resolver el problema de Job Shop Scheduling, también dos Procedimientos de Búsqueda Ciega Aleatorizado y Adaptativo GRASP y GRASP+RT (GRASP que utiliza un re-enlace de trayectoria). Los resultados indican que: el AIS mostró tener un porcentaje de desviación menor que GRASP pero mayor que GRASP+RT esto respecto a la mejor solución conocida. Esto se debe que el algoritmo hibrido de GRASP está dotado con memoria el cual lo hace más poderoso.

Mientras la tecnología avanza a pasos agigantados es importante en estos días optimizar la programación de una mejor manera con la ayuda extra de la computación. El objetivo es obtener mejores resultados en la programación usando AIS (Artificial Immune System) en vez de usar métodos heurísticos tradicionales [32] (Ver Figura 6).

Figura 6: Métodos Meta Heurísticos



Fuente: Tupia [33]

JSP provee un conjunto de recursos para tareas a través del tiempo. El trabajo que se ha llevado en los últimos años se ha enfocado en encontrar maneras de darles tareas a la máquinas, tal que cumplan ciertos criterios y que la función objetivo sea optimizado.

Las restricciones para JSP son [32]:

1. Una tarea no puede visitar la misma máquina más de una vez
2. No existen prioridades para las restricciones en operaciones con distintas tareas
3. Las operaciones no pueden ser interrumpidas
4. Cada máquina procesa sólo un trabajo a la vez
5. El tiempo en que empieza la operación y en el que se vence no se especifican

6. Cada tarea debe atravesar una secuencia particular de operaciones que esta predefinida

El objetivo más usado en el JSP es el de encontrar un plan de trabajo válido (que no viole ninguna de las restricciones) que complete todos los trabajos en el menor tiempo posible. Este objetivo se conoce como *makespan* y es el que se minimiza.

2.3.1 DEFINICIÓN MATEMÁTICA DE JSP

En el Job Shop Scheduling Problem (JSP), un conjunto finito de trabajos es procesado sobre un conjunto finito de maquinas. Cada trabajo se caracteriza por un orden fijo de las operaciones, cada una de las cuales será procesada en una maquina específica por una duración especificada. Cada maquina puede procesar a lo mas un trabajo al mismo tiempo y una vez que un trabajo ha iniciado sobre una maquina se debe completar su procesamiento sobre esa maquina por un tiempo ininterrumpido. Un Calendario es una asignación de operaciones en intervalos de tiempo sobre las maquinas. El makespan es el máximo tiempo en completar los trabajos. El objetivo de JSP es encontrar un calendario que minimice el makespan.

Formalmente, el JSP puede ser definido como se muestra a continuación. Dado un conjunto M de maquinas ($|M|$ denota el tamaño de M) y un conjunto J de trabajos ($|J|$ denota el tamaño de J), sean $\sigma_1^j \prec \sigma_2^j \prec \cdots \prec \sigma_{|M|}^j$ sea el orden de un conjunto $|M|$operaciones del trabajo j, donde $\sigma_k^j \prec \sigma_{k+1}^j$ indica que la operación σ_{k+1}^j solo puede empezar el procesamiento después de completar la operación σ_k^j. Sea O el conjunto de operaciones. Cada operación es definida por dos parámetros: M_k^j es la maquina sobre la cual σ_k^j es procesada y $p_k^j = p(\sigma_k^j)$ es el tiempo de procesamiento de la operación σ_k^j. Definiendo

$t(\sigma_k^j)$ como el tiempo de inicio de la $k-esima$ operación $\sigma_k^j \in O$, una formulación de programación disyuntiva para el JSP se muestra a continuación:

$\min C_{max}$

$sujeto\ a:$

$C_{max} \geq t(\sigma_k^j) + p(\sigma_k^j),\ \ para\ toda\ \sigma_k^j \in O,$

$(1a).\ \ t(\sigma_k^j) \geq t(\sigma_l^j) + p(\sigma_l^j), \qquad para\ toda\ \ \sigma_l^j \prec \sigma_k^j,$

$(1b).\ \ t(\sigma_k^j) \geq t(\sigma_l^i) + p(\sigma_l^i) \vee t(\sigma_l^i) \geq t(\sigma_k^j) + p(\sigma_k^j)\ \ para\ todo\ i,j\ \in J$
$\exists\ M_{\sigma_l^i} = M_{\sigma_k^j},$

$t(\sigma_k^j) \geq 0, para\ toda\ \ \sigma_k^j \in O$

C_{max} es el makespan a ser minimizado.

Una solución factible puede ser construida de una permutación de J sobre cada una de las maquinas M, observando las restricciones de precedencia, la restricción de que cada maquina puede procesar solo una operación a la vez y requiriendo una vez iniciada, el procesamiento de una operación debe ser ininterrumpido hasta ser completada. Cada conjunto de permutaciones tiene un correspondiente Calendario.

Por tanto, el objetivo del JSP es encontrar un conjunto de permutaciones con el makespan más pequeño.

2.4 PROCEDIMIENTO DE BÚSQUEDA MIOPE ALEATORIA Y ADAPTATIVA (GRASP)

Fase de construcción conforma una solución factible, un elemento a la vez, En el caso del problema de JSP, se considera una sola operación como bloques

de construcción. Se crea un calendario factible asignando operaciones individuales, una a la vez hasta que todas las operaciones son asignadas.

Sea O_k^j denota k-th operación del trabajo j y está definido por (M_k^j, p_k^j), donde M_k^j es la maquina sobre la cual la operación O_k^j es realizada y p_k^j es el tiempo de procesamiento de la operación O_k^j.

Mientras se construye un calendario factible, no todas las operaciones pueden ser seleccionadas en una fase de construcción dada. Una operación O_k^j puede solo ser asignada a un calendario si todas las operaciones a priori de los trabajos j han sido asignadas. Por tanto, en cada iteración de la fase de construcción, a lo mucho $|J|$ operaciones son candidatas a ser asignadas. Este conjunto de operaciones candidatas se denota como O_c.

Más de un algoritmo codicioso puede ser propuesto para el problema de JSP. Uno de los algoritmos selecciona las operaciones O_k^jque se traduce en el menor incremento en el makespan de los trabajos ya programados para siguiente calendario. Sea la función de codicia $h(O)$ que denote el makespan como resultado de la adición de operaciones O a las operaciones ya programadas.

La elección codiciosa es la siguiente operación de asignación de tareas:

$$\underline{O} = argmin(h(O)|O \in O_c)$$

Definiendo

$$\overline{O} = argmax(h(O)|O \in O_c)$$

$\underline{h} = h(\underline{O})$,Y $\overline{h} = h(\overline{O})$, la lista de candidatos restringidos (RCL) se define como:

$RCL = \{(O \in O_c | \underline{h} \leq h(O) \leq \underline{h} + \alpha(\overline{h} - \underline{h}))\}$.

Donde α es un parámetro tal que $0 \leq \alpha \leq 1$.

La búsqueda local empleada en este GRASP para el problema de JSP es búsqueda local de dos intercambios basado en el grafo disyuntivo [34]

$G = (V, A, E)$es definido tal que:

$$V = \{O \cup \{0, |J|.|M| + 1\}\}$$

Es el conjunto de nodos, donde $\{0\}$ y $|J|.|M| + 1$ son fuentes y sumideros artificiales.

$$A = \{((v,w)|v,w \in O, v \prec w)\} \cup \{((0,w)|w \in O, \nexists\, v \in O \ni v \prec w)\} \cup \{((v,|J|.|M| + 1)|v \in O, \nexists\, w \in O \ni w \prec v)\}$$

Es el de arcos dirigidos, conectando consecutivamente operaciones del mismo trabajo, y

$$E = \{((v,w)|M_v = M_w)\}$$

Es el conjunto de operaciones conectadas sobre la misma maquina. Vértices en el modelo de grafo disyuntivo ponderado. Los vértices 0 y $|J|.|M| + 1$ y tienen peso cero, mientras que el peso de los vértices $i \in \{1, \dots, |J|.|M|\}$ es el tiempo de procesamiento de las operaciones correspondientes al vértice i.

La entrada para GRASP incluye parámetros para ajustar el tamaño de la lista de candidatos y el número máximo de iteraciones de GRASP, y la semilla para el generador de números aleatorios. A continuación se presenta el seudocódigo de GRASP [35]:

Algoritmo 2: GRASP

$$\textbf{procedure } GRASP(Max_{Iterations}, Seed)$$

$$Read_{input}(.);$$

$$\textbf{for } k = 1, \dots, Max_{Iterations} \textbf{ do}$$

$$solution \leftarrow GreedyRandomised_{Construction}(Seed);$$

$$solution \leftarrow LocalSearch(Solution);$$

$$Update_{Solution}(Solution, Best_{Solution});$$

$$\textbf{end};$$

$$\textbf{return } Best_{solution};$$

$$\textbf{end } GRASP$$

3 METODOLOGÍA DE DESARROLLO

Según lo enunciado en el objetivo general de la propuesta de investigación se extrajeron tres tareas básicas:

1. Implementación del algoritmo Clonalg para instancias de Lawrence

2. Implementación del algoritmo GRASP para instancias de Lawrence

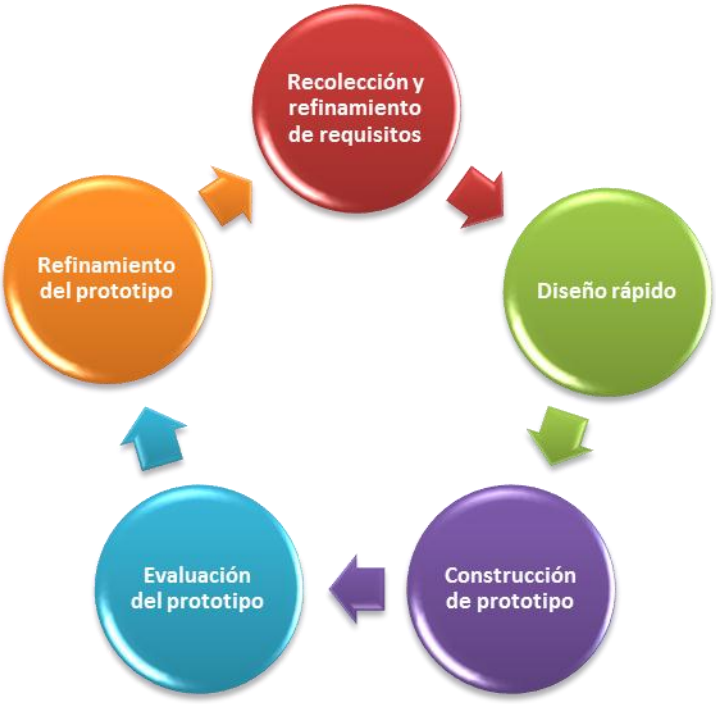
3. Caracterización de posibles escenarios de aplicación del problema JSP en el campo de la computación basados en la familia de instancias utilizadas.

Se planteó tres pasos para alcanzar los objetivos. Luego de realizar el estudio de las metodologías de desarrollo, se pudo concluir que la metodología que mejor se ajustaba a la realización de las anteriores tareas era la metodología de trabajo Prototipado Evolutivo.

El objetivo principal de este tipo de Prototipado es la construcción de un prototipo robusto en la medida que el usuario final es quien provee la realimentación, obteniendo un mejor producto con cada iteración de la metodología. Esta técnica permitió que el equipo de desarrollo pudiera agregar funciones, y hacer cambios que no pudieron ser definidos en la elaboración de requerimientos y en la fase de diseño.

El prototipado evolutivo tiene una ventaja sobre los prototipos desechables, ya que son sistemas funcionales. Aunque puede que no tengan todas las características que los usuarios han planeado, se pueden utilizar de forma provisional hasta que el sistema final sea entregado.

Esta metodología constó de las siguientes etapas:



Dado la investigación propuesta y las tres tareas extraídas del objetivo general, se propuso elaborar tres prototipos funcionales.

Prototipo 1: El primer prototipo implementó, un algoritmo inmune tipo Clonalg y fue probado con las instancias de Lawrence.

Prototipo 2: En el segundo prototipo se implementó el algoritmo de GRASP y fue probado con las instancias de prueba utilizadas en el primer prototipo.

Prototipo 3: El tercer prototipo relacionó los datos obtenidos con el fin de contrastar el algoritmo inmune y el algoritmo de GRASP con cada una de las instancias. En este informe se pudo apreciar los resultados para los valores de makespan (tiempo en que termina el último trabajo) para las meta heurísticas del primer y segundo prototipo.

Adicionalmente, se caracterizó un escenario para realizar las pruebas de cada instancia sobre los dos algoritmos propuestos, diseñando para tal fin dos instancias, una por cada algoritmo.

3.1 FASE DE RECOLECCIÓN Y REFINAMIENTO DE REQUISITOS

Aquí se analizaron los problemas detalladamente y las necesidades del sistema, para lo cual se llevaron acabo las siguientes actividades:

- Recolección de la documentación requerida para definir y estructurar el problema de Job Shop.

- Caracterización de los algoritmos inmunes, en especial Clonalg. Estudio del Algoritmo GRASP. Definición de análisis necesarios para contrastar los resultados, tablas, gráficos, etc.

- Se definió como necesario presentar los resultados del estudio adelantado en una interfaz que permita interactuar con los algoritmos implementados.

3.2 FASE DE DISEÑO RÁPIDO

Se usó toda la información recopilada anteriormente y se elaboró un diseño lógico del sistema. Se hicieron las especificaciones formales, lo cual implicó diseñar procedimientos precisos de captura de datos, accesos efectivos al sistema, interfaz de usuario, etc., todo esto se llevo acabo por medio de:

- Elaboración de diagramas de flujo de los algoritmos usaría el sistema.

- Diseño de la interfaz de entrada y elaboración de formatos de pantalla para la entrada y salida de datos en el sistema.

3.3 FASE DE CONSTRUCCIÓN DEL PROTOTIPO

Esta fase, que tuvo como objetivo traducir las especificaciones del diseño en un código de programación, se llevó a cabo las siguientes actividades:

- Desarrollo de la interfaz de entrada y salida de datos del sistema con los formatos definidos.

- ·Módulo de selección de instancias de la familia LA (Ver Anexos1, 2 y 3) las cuales se usaron en el prototipo 1 y 2.

- Implementación de algoritmo inmune Clonalg y el algoritmo de GRASP con el fin de ajustarlos al problema de Job Shop, luego de esto se evaluó los resultados y la visualización del calendario que estaba ya optimizado. Creación del modulo de análisis de resultados y graficación.

3.4 FASE DE EVALUACIÓN DEL PROTOTIPO

En esta etapa se efectuaron los análisis de sensibilidad que permitieron deducir los parámetros con los que se alcanza el mejor desempeño para los algoritmos implementados, ya que en la literatura no existe un criterio estándar para la elección de los mismos.

Posteriormente se realizó un cuadro compartido en donde se contrastan los makespan de los dos algoritmos implementados frente al BKS[5].

3.5 FASE DE REFINAMIENTO DEL PROTOTIPO

Una vez realizados los análisis de sensibilidad se probaron los algoritmos con los parámetros establecidos y se realizó un cuadro comparativo con los

[5]BKS (Best Known Solution): mejor solución conocida en la literatura

makespan de los algoritmos y el BKS. Este estudió permitió verificar cual de los dos enfoques, el inmune o el estocástico, produce mejores soluciones en términos de makespan.

4 DESCRIPCIÓN DE LA TECNICA

En este capítulo se hará una breve descripción de las técnicas con las cuales se optimizará el Problema de Job Shop Scheduling. Las técnicas están basadas en la selección clonal del sistema inmune natural, más concretamente en el algoritmo Clonalg y la técnica estocástica GRASP.

En este trabajo se propusieron dos algoritmos uno de tipo Inmune y otro de tipo estocástico para resolver el problema de asignación de tareas. El tipo de representación para la asignación de los trabajos es permutación con repetición y la codificación es de tipo decimal.

En el caso del algoritmo Clonalg el anticuerpo corresponde al orden en que se asignan los trabajos; mientras que el antígeno es representado por el valor con el que se evalúa el anticuerpo en la función objetivo, es decir, el makespan. Los anticuerpos con el paso de las generaciones se ajustan según la función objetivo, la cual depende de la afinidad de ellos mismos con respecto del antígeno.

El algoritmo GRASP tiene dos etapas: Construcción y búsqueda local, las cuales fueron implementadas.

En la primera etapa se construye un lista restringida de candidatos (RCL) y en la segunda, se implementa la búsqueda local la cual tiene como entrada uno de los candidatos de la lista restringida con el fin de mejorar el makespan del candidato a través de las iteraciones y así llegar a una solución optima.

4.1 REPRESENTACIÓN DEL PROBLEMA

Como se ha mencionado antes los problemas de Job Shop Scheduling son problemas aplicados a la vida real, y son más frecuentes en entornos productivos y deservicios, donde se requiere organizar en el tiempo la ejecución de tareas que comparten un conjunto finito de recursos. Esta planificación se muestra por medio de instancias.

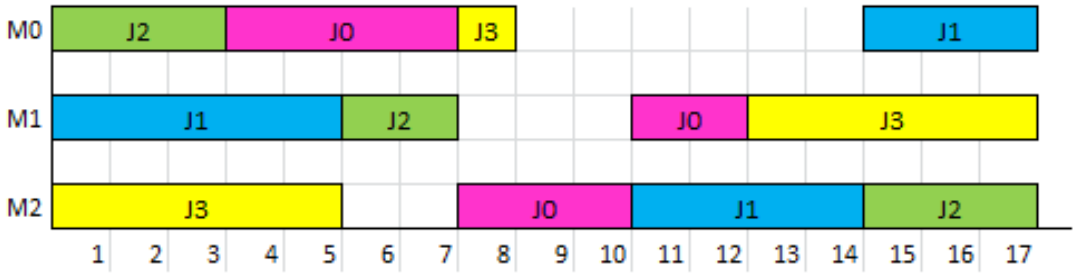
Una instancia del JSP se define mediante una matriz, la cual contiene cada uno de los trabajos, el orden dentro de las maquinas y el tiempo de procesamiento correspondiente a cada trabajo en cada máquina. (Véase Tabla 3)

Tabla 3: Flujo y tiempos de los trabajos en el JSP

Trabajo	Máquina	Tiempo	Máquina	Tiempo	Máquina	Tiempo
J0	0	4	2	3	1	2
J1	1	5	2	4	0	3
J2	0	3	1	2	2	3
J3	2	5	0	1	1	5

Para facilitar la visualización de la programación de los trabajos, se hace uso de planes de trabajos donde se acomodan los trabajos en las máquinas de tal forma que cumplan con las restricciones del problema, dicha visualización se conoce como *diagrama de Gantt* (ver figura7).

Figura 7: Diagrama de Gantt



Los diagramas de Gantt representan cada máquina en un renglón diferente y cada cuadro representa una operación. Los cuadros están marcados con el número de trabajo al que corresponde, es decir se identifica por el color y el número especificado, por último en el eje x se encuentran las unidades de tiempo que los trabajos gastan en completar cada una de las operaciones.

En los problemas de Job Shop Scheduling se encuentran dos tipos de planes de trabajos: activo o semi-activo. Los primeros son planes de trabajo donde ninguna operación se inicia sin que se retarde cualquier otra operación o sin que se viole alguna de las restricciones. Por el contrario los planes de trabajos semi-activos son planes en los cuales ninguna operación puede iniciarse sin cambiar el orden de procesamiento o sin que se viole alguna de las restricciones [5].

Figura 8 Paso 1: Optimización de un plan de trabajo

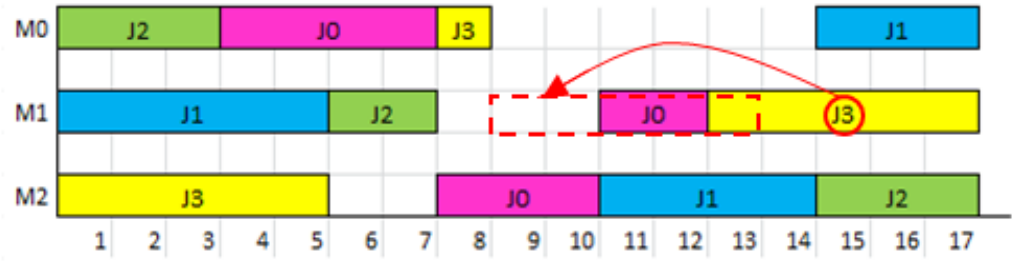


Figura 9 Paso 2: Optimización de un plan de trabajo

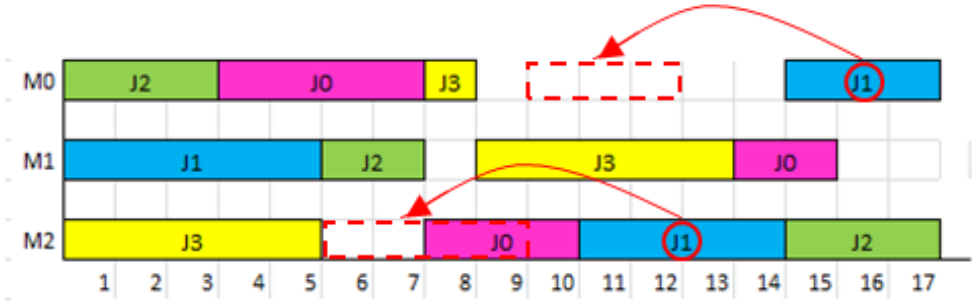


Figura 10 Paso 3: Optimización de un plan de trabajo

Paso 1: En la Figura 8, se muestra un calendario semi-activo, en el cual se puede optimizar al cambiar el orden del trabajo J0 por el trabajo J3 en la máquina M1, dado que si se inicia antes no afecta el tiempo de la operación del trabajo.

Paso 2: En la Figura 9 se puede observar que al cambiar el orden del trabajo J1 en la maquina M2 permitirá reducir el tiempo de retardo y a su vez este mismo trabajo en la máquina M0 podrá llevarse acabo antes.

Paso 3: Finalmente, en la Figura 10 se observa un calendario no retardado, es decir, que no existe ningún movimiento que haga que no se retrase ninguna operación.

Un calendario donde ninguna de las máquinas está ociosa es decir, que está lista para procesar los trabajos, se conoce como calendario o plan de trabajo *sin retardo.*

Se han detallado las características del Problema de Job Shop Scheduling, ahora solo queda presentar las diferentes técnicas para resolverlo. En esta tesis se abordarán el algoritmo de selección clonal (Clonalg), y la meta heurística GRASP.

5 PROBLEMAS DE PRUEBA Y ANÁLISIS DE RESULTADOS

5.1 MÉTRICA Y CONTENDIENTE

La calidad de nuestros algoritmos (Clonalg, GRASP) fue medida por medio del valor de la solución obtenida luego de realizar la optimización con cada una de las instancias de Lawrence, estos resultados fueron comparados entre ellos mismos y con respecto al mejor resultado conocido (BKS).

De igual manera se llevó acabo mediciones estadísticas como el promedio, la varianza y la desviación estándar de la mejor solución obtenida de las 20 ejecuciones de cada una de las instancias en los dos algoritmos. Estos resultados se presentaron de manera gráfica para permitir un mejor análisis de los resultados.

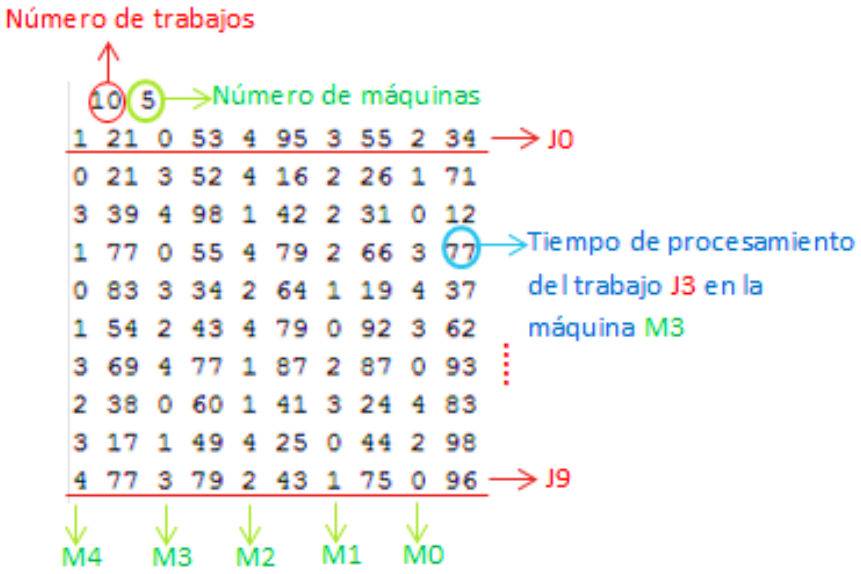
5.2 DESCRIPCIÓN DE LOS PROBLEMAS

Las instancias de Lawrence LA tiene 40 problemas de 8 diferentes tamaños propuestos: 10 x 5, 15 x 5, 20 x 5, 10 x 10, 15 x 10, 20 x 10, 30 x 10 y 15 x 15. Lawrence llamó Fl-5, Gl-5, Hl-5, Al-5, Bl-5, Cl-5, Dl-5, y Il-5 a las instancias respectivamente. Sin embargo el nombre LA fue dado por Applegate y Cook [36] y es uno de los más comúnmente utilizados. Los tiempos de procesamiento fueron generados en el de 5 a 99 unidades de tiempo.

Cada ejemplo se compone de una línea de descripción; cada fila contiene el número de puestos de trabajo y el número de máquinas, y luego una línea para cada puesto de trabajo, indicando el número de la máquina y el tiempo de procesamiento de cada paso del trabajo. Las máquinas se numeran

empezando por 0. A continuación (Ver figura 11), un ejemplo de la instancia de Lawrence con cada uno de sus componentes.

Figura 11: Representación de la instancia La01



5.2.1 DETALLES DE LOS PROBLEMAS

A continuación se muestran cada una de las instancias de Lawrence que fueron tomadas de [37] , junto con los mejores resultados encontrados en la literatura de múltiples trabajos de investigación basados en la optimización de JSP y comparados con los dos algoritmos Clonalg y GRASP.

Las comparaciones se baso en las principales características de cada uno de los problemas tales como: nombre del Problema (Problema), tamaño de la instancia (n, m), el mejor makespan encontrado, varianza y desviación estándar.

5.3 RESULTADOS

En esta sección se muestran los resultados obtenidos con las instancias previamente descritas. La prueba realizada con cada una de las instancias que consta de 20 ejecuciones del algoritmo. Entre los resultados obtenidos, para este caso se tiene el makespan, la evaluación en la que se encontró, y las estadísticas de todas las ejecuciones del algoritmo.

5.3.1 ENTORNO COMPUTACIONAL

Las pruebas y ejecuciones del programa fueron ejecutados en un computador con procesador Intel Core 2 Duo a 2.20 GHz con 2 GB de RAM. El software se compilo para el sistema operativo Windows y el código fue desarrollado en el lenguaje de programación Java.

5.3.2 ANÁLISIS DE SENSIBILIDAD

El análisis de sensibilidad consistió en buscar empíricamente los criterios de optimización, es decir, los parámetros de entrada de los algoritmos que generen soluciones óptimas. Este proceso se da a través de contrastar esta observación empírica de los algoritmos implementados, con la teoría desarrollada en trabajos de los autores revisados [5, 27, 29].

Con el fin de tener una muestra variada se tomó una instancia por cada tamaño diferente de todas las instancias de Lawrence. Estas fueron las instancias con sus respectivos tamaños: LA 01 (10 x 5), LA 09 (15 x 5), LA 15 (20 x 5), LA 20 (10 x 10), LA 25 (15 x 10), LA 30 (20 x 10), LA 35 (30 x 10), LA 40 (15 x 15).

En los distintos autores revisados en este trabajo, se halló que la forma en que se seleccionaron los parámetros es por prueba y error; ya que en la literatura

no hay una forma exacta para determinar las soluciones óptimas en la implementación de los algoritmos.

Por ejemplo, el número de generaciones requeridas para que la función objetivo obtuviera resultados aproximados a la BKS (Best Known Solution), se definió por medio de un análisis de sensibilidad que se desarrolla a continuación. En cuanto al tamaño de la población, se definió arbitrariamente una población de tamaño 100 pues es un valor típico usado en la literatura [28].

5.3.2.1 CLONALG: SENSIBILIDAD DE GENERACIONES

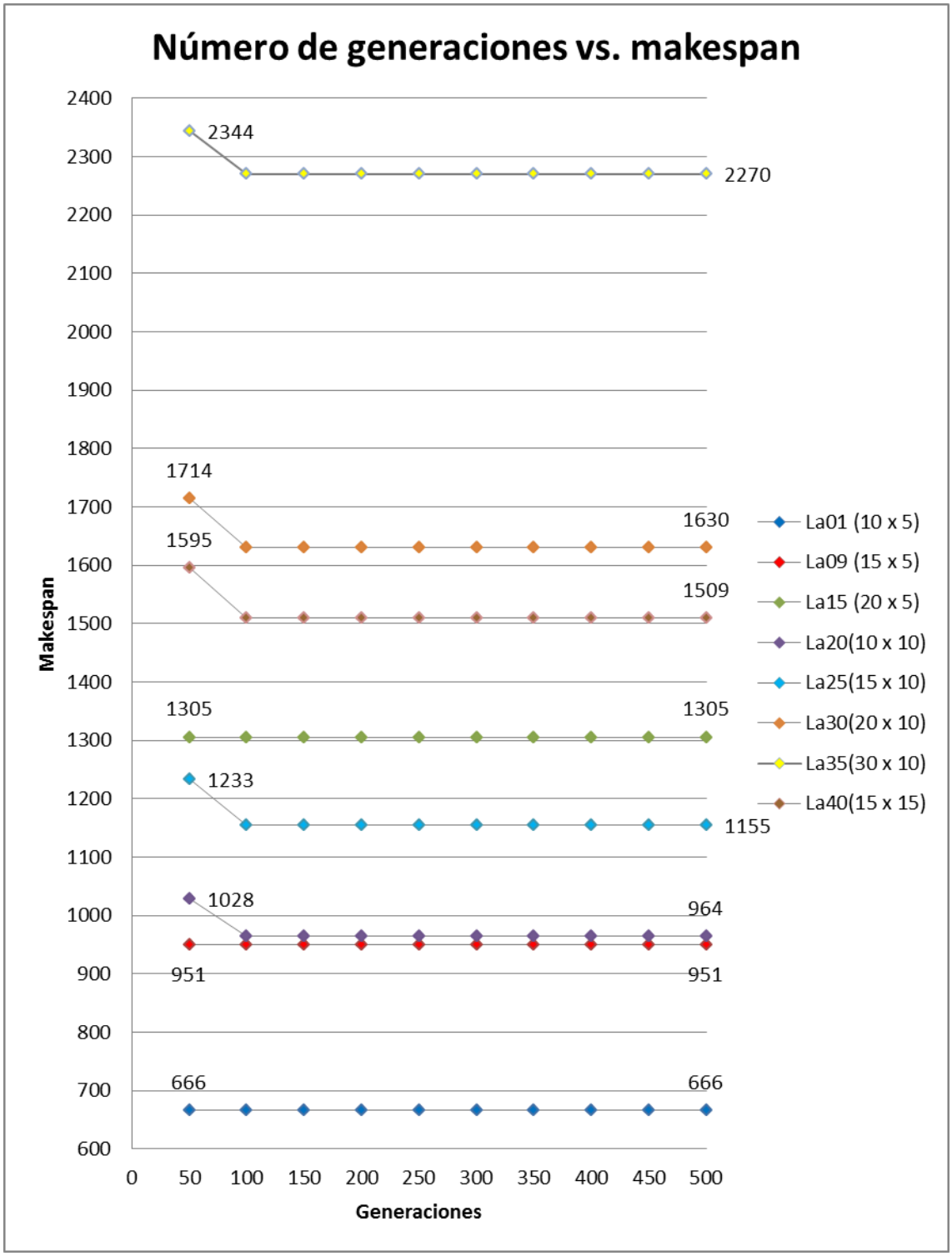
Dadas las herramientas computacionales mencionadas en lasección5.3.1, se hicieron 500 generaciones divididas en intervalos de 50 generaciones. Se tomó como referencia la variación del makespan respecto al número de generaciones; los resultados se pueden ver en la tabla 4.

Tabla 4: Sensibilidad del número de generaciones

	La01	La09	La15	La20	La25	La30	La35	La40
Número de Generaciones	m x n **10x5**	m x n **15x5**	m x n **20x5**	m x n **10x10**	m x n **15x10**	m x n **20x10**	m x n **30x10**	m x n **15x15**
50	666	951	1305	1028	1233	1714	2344	1595
100	666	951	1305	964	1155	1630	2270	1509
150	666	951	1305	964	1155	1630	2270	1509
200	666	951	1305	964	1155	1630	2270	1509
250	666	951	1305	964	1155	1630	2270	1509
300	666	951	1305	964	1155	1630	2270	1509
350	666	951	1305	964	1155	1630	2270	1509
400	666	951	1305	964	1155	1630	2270	1509
450	666	951	1305	964	1155	1630	2270	1509
500	666	951	1305	964	1155	1630	2270	1509
Generación en que se halla la mejor solución	16	15	34	95	93	90	75	84

En la figura 12 se observa que luego de 100 generaciones el algoritmo de Clonalg no encuentra nuevas mejoras en el makespan. De acuerdo a este análisis, se definió que el máximo número de generaciones necesarias para hallar un valor óptimo es de 100.

Figura 12: Makespan vs. generaciones de Clonalg



5.3.2.2 CLONALG: SENSIBILIDAD DEL FACTOR DE MUTACIÓN

Dado que en la literatura los autores toman valores al azar para el análisis de sensibilidad del factor de mutación, se tomaron valores entre 0 y 5, puesto que se probó con valores superiores a 5 y no se observaron mejorías en el makespan, por el contrario generaba valores que se alejaban al BKS conforme se aumentaba el valor del factor de mutación.

De acuerdo con dicho análisis, se estableció el rango del factor de mutación entre 0 y 1, ya que es ahí donde se obtuvieron los valores menores de makespan. Esto se ve reflejado en la tabla 5:

Tabla 5: Sensibilidad del Factor de Mutación

	La01	**La09**	**La15**	**La20**	**La25**	**La30**	**La35**	**La40**
Factor de mutación	m x n	m x n	m x n	m x n	m x n	m x n	m x n	m x n
	10x5	**15x5**	**20x5**	**10x10**	**15x10**	**20x10**	**30x10**	**15x15**
0,01	683	961	1402	1025	1290	1729	2436	1672
0,19	666	951	1328	1035	1236	1709	2349	1666
0,26	666	951	1305	965	1183	1636	2289	1586
0,5	689	954	1368	1059	1249	1727	2354	1708
1	685	968	1378	1052	1309	1756	2403	1706
3	696	971	1410	1069	1298	1763	2406	1743
5	694	972	1381	1058	1297	1769	2409	1730
BKS	666	951	1207	902	977	1355	1888	1222

- Luego de fijar el rango del factor de mutación, se establecieron nuevamente valores aleatorios entre dicho rango y se realizaron de nuevo las pruebas.
- A medida que se observaba una mejora en el makespan, la distancia entre un valor y otro se ajustaba.
- Luego del análisis el valor específico para el Factor de mutación (porcentaje de mutación), se fijó en 0,26.

Al fijar este parámetro se confirmó la noción planteada por Cortés [5] donde *"El impacto que tiene la mutación en un anticuerpo para generar un nuevo individuo es mínima, ya que se aplica de tal forma que sólo se realiza un cambio en la cadena".*

Un cambio mínimo en la cadena produjo mejoras del 11% en los valores del makespan. Esto se ve en las gráficas13, 14, 15, 16, 17, 18, 19, 20en donde para el valor fijado de 0,26 en el factor de mutación optimizó la mayoría de las instancias, y las que no se optimizaron hasta conseguir el valor del BKS, tienen una diferencia pequeña, la cuál se encuentra entre el rango del 8% y 21%.

Figura 13: Factor de mutación vs. makespan (La01)

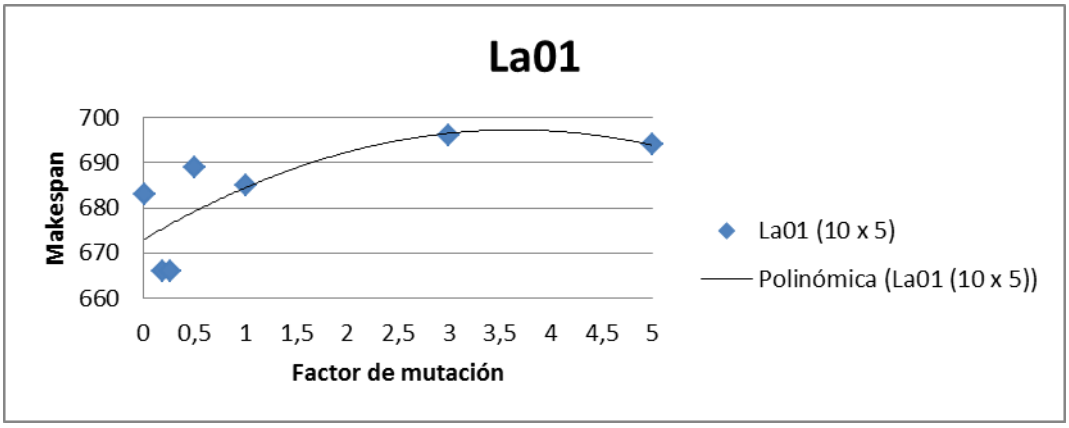


Figura 14: Factor de mutación vs. makespan (La09)

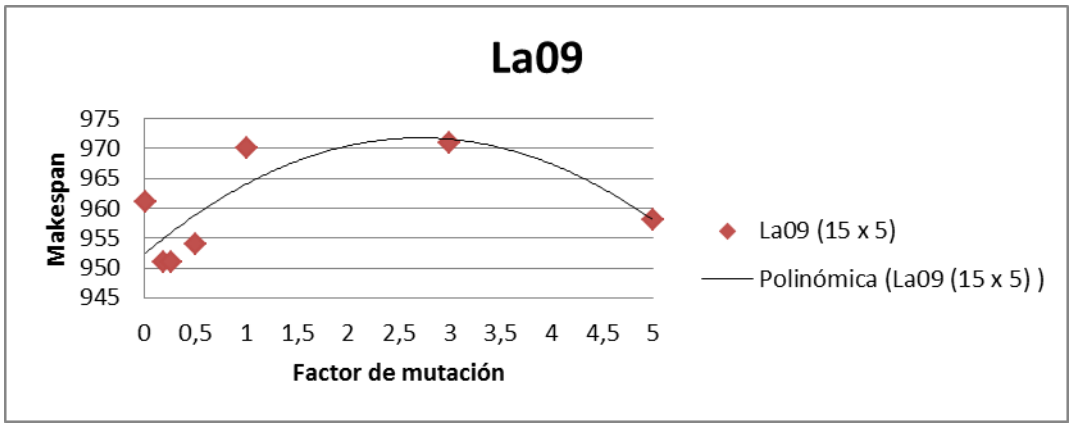


Figura 15: Factor de mutación vs. makespan (La35)

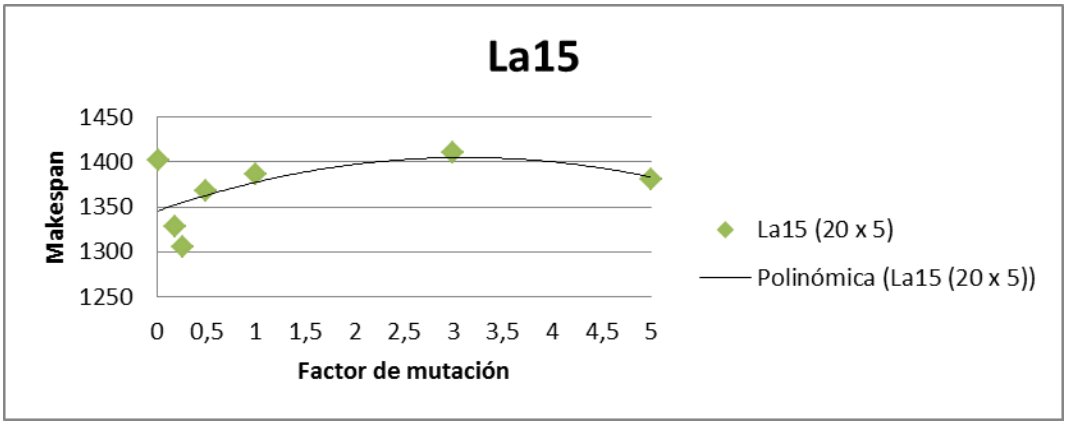


Figura 16: Factor de mutación vs. makespan (La20)

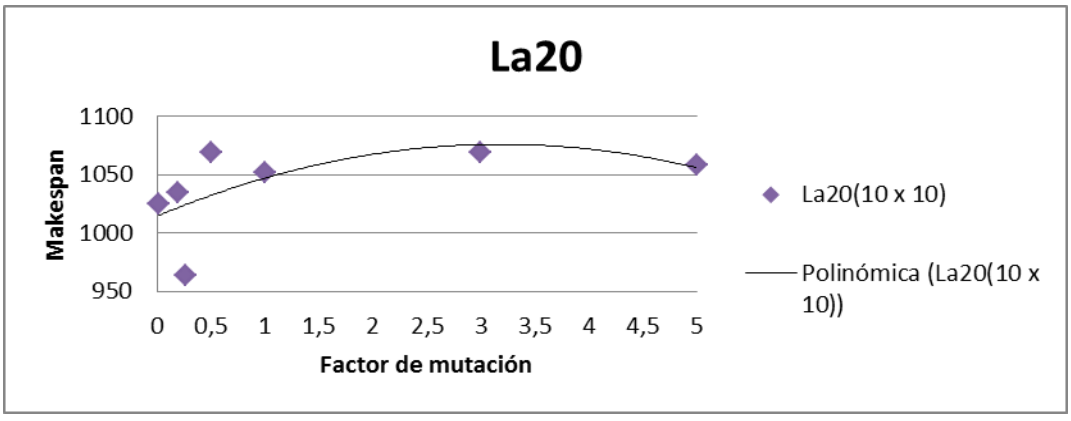


Figura 17: Factor de mutación vs. makespan (La25)

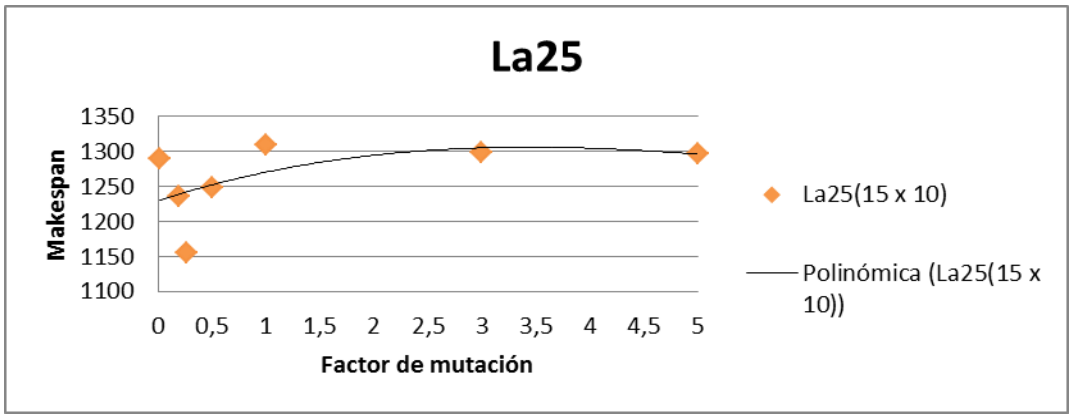


Figura 18: Factor de mutación vs. makespan (La30)

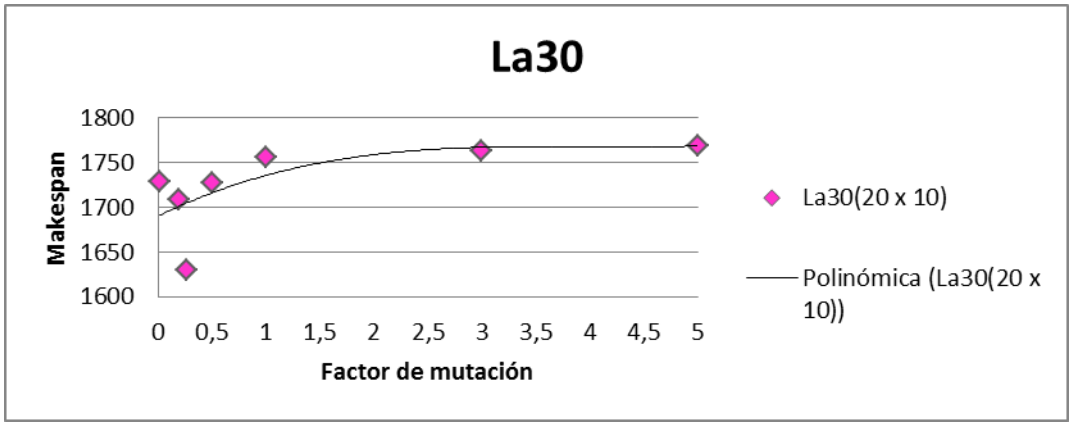


Figura 19: Factor de mutación vs. makespan (La35)

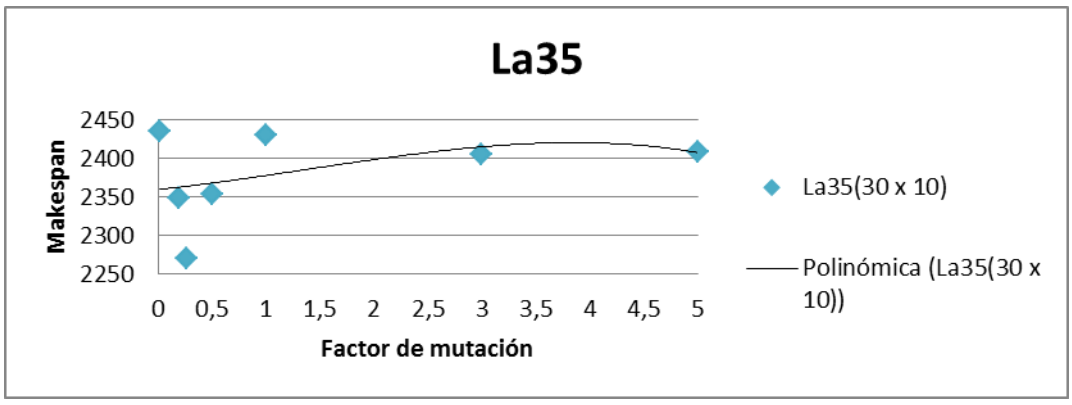
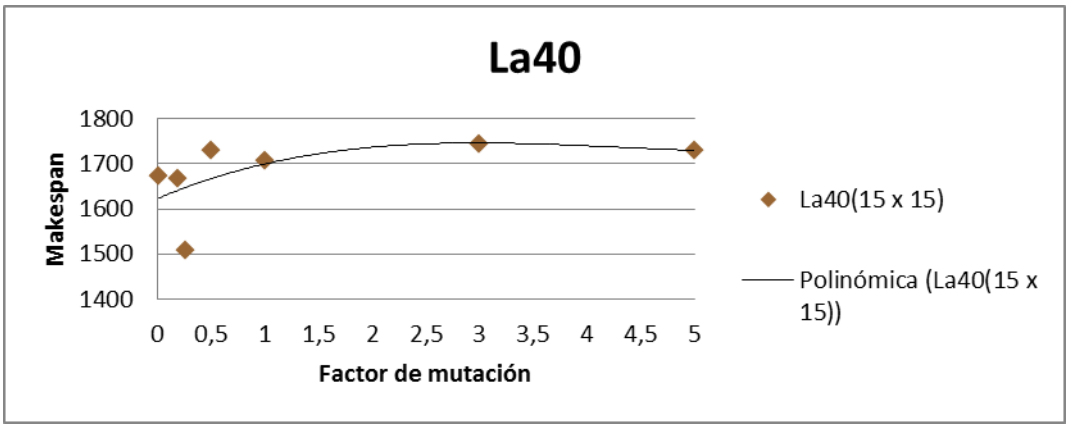


Figura 20: Factor de mutación vs. makespan (La40)



5.3.2.3 CLONALG: SENSIBILIDAD DEL FACTOR DE CLONACIÓN

El factor de clonación tiene un rango entre 0,1 y 1(ver tabla 6). Con el fin de abarcar el rango completo se empezó en 0.1, y se fue aumentando el factor hasta alcanzar su límite superior con un paso de 0.1 para cada ejecución.

Se concluye entonces partir de la tabla que el mejor candidato para el factor de clonación es de 0.3, pues este minimizó el makespan en 7 de las 8 instancias experimentadas tal como se observa en las gráficas de dispersión 21 al 24.

Tabla 6: Sensibilidad del factor de clonación

Factor de Clonación	La01	La09	La15	La20	La25	La30	La35	La40
	m xn	m xn	m xn	m xn	m xn	m xn	m xn	m xn
	10x5	15x5	20x5	10x10	15x10	20x10	30x10	15x15
0,1	666	951	1342	978	1246	1697	2264	1670
0,2	666	952	1308	987	1211	1679	2197	1653
0,3	666	964	1305	964	1155	1614	2189	1509
0,4	666	954	1334	996	1228	1647	2308	1546
0,5	675	955	1317	1004	1191	1628	2240	1616
0,6	666	951	1377	1012	1236	1621	2195	1650
0,7	675	951	1352	968	1200	1630	2240	1556
0,8	666	951	1344	979	1241	1637	2315	1620
0,9	666	962	1348	978	1247	1614	2205	1618
1	666	951	1364	992	1177	1614	2206	1623

Figura 21: Factor de clonación vs. makespan(La01 – La09)

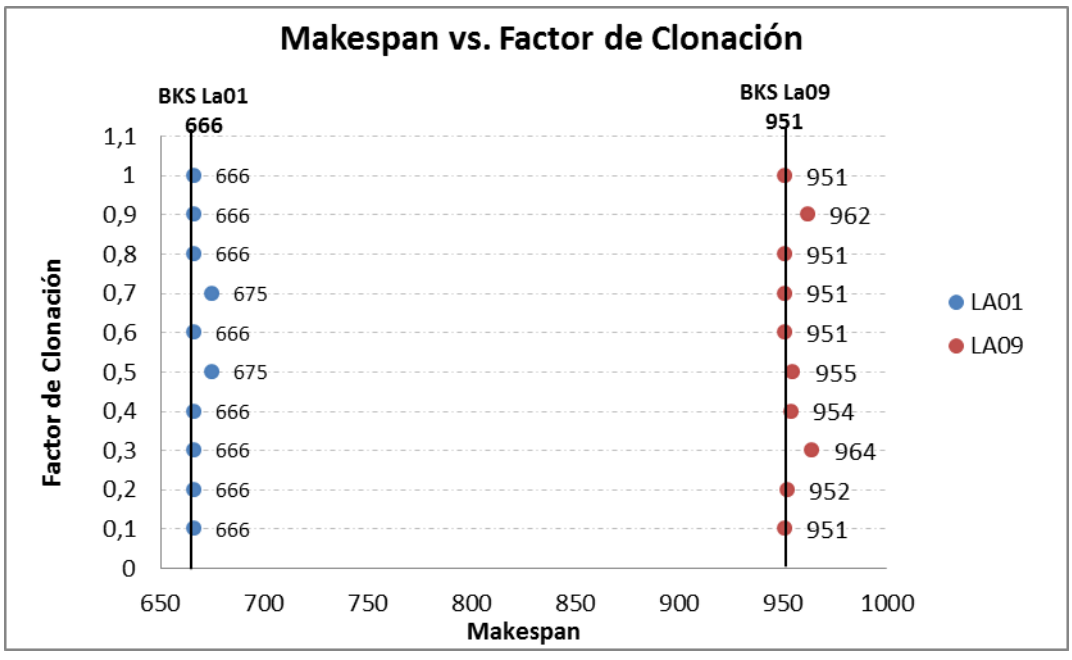


Figura 22: Factor de clonación vs. makespan (La15 - La20)

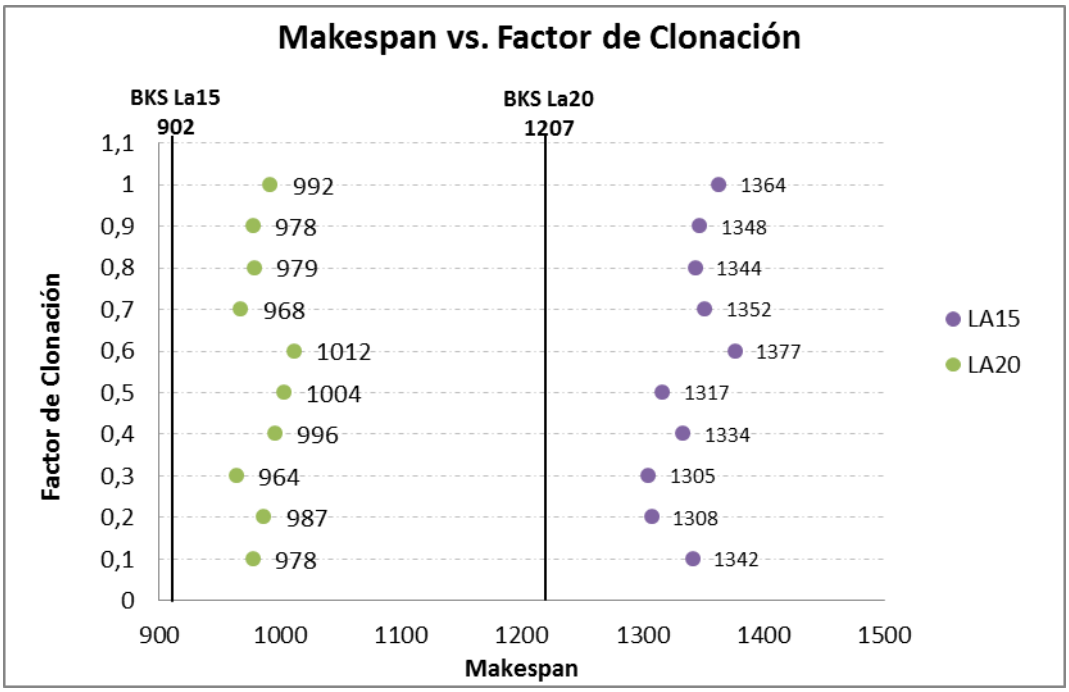


Figura 23: Factor de clonación vs. makespan (La25 – La30)

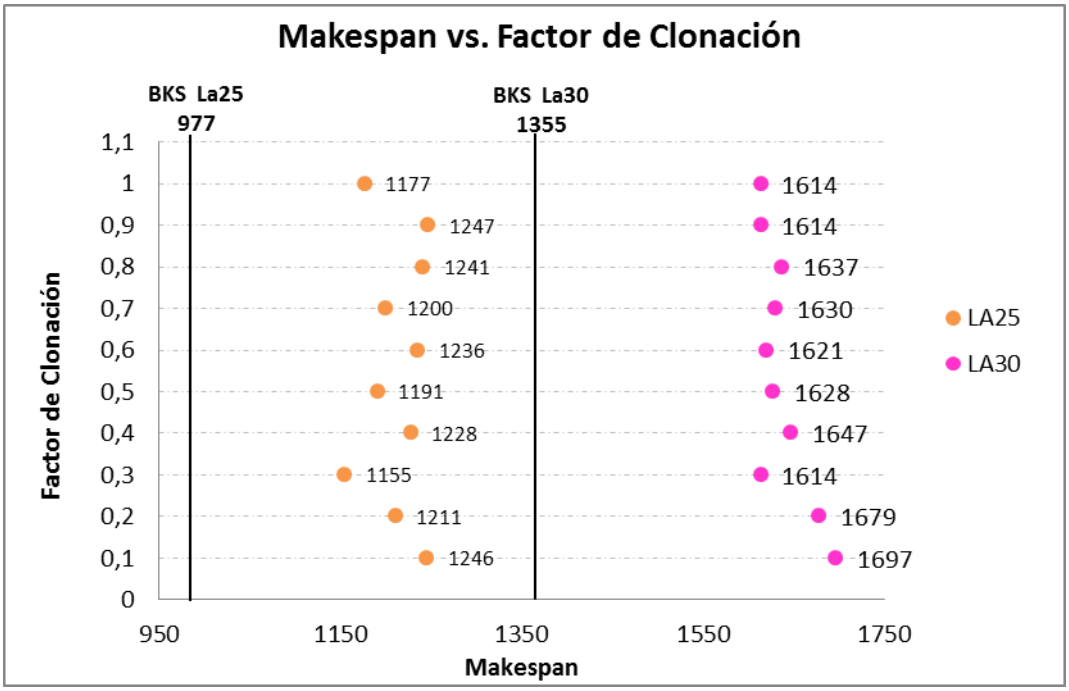
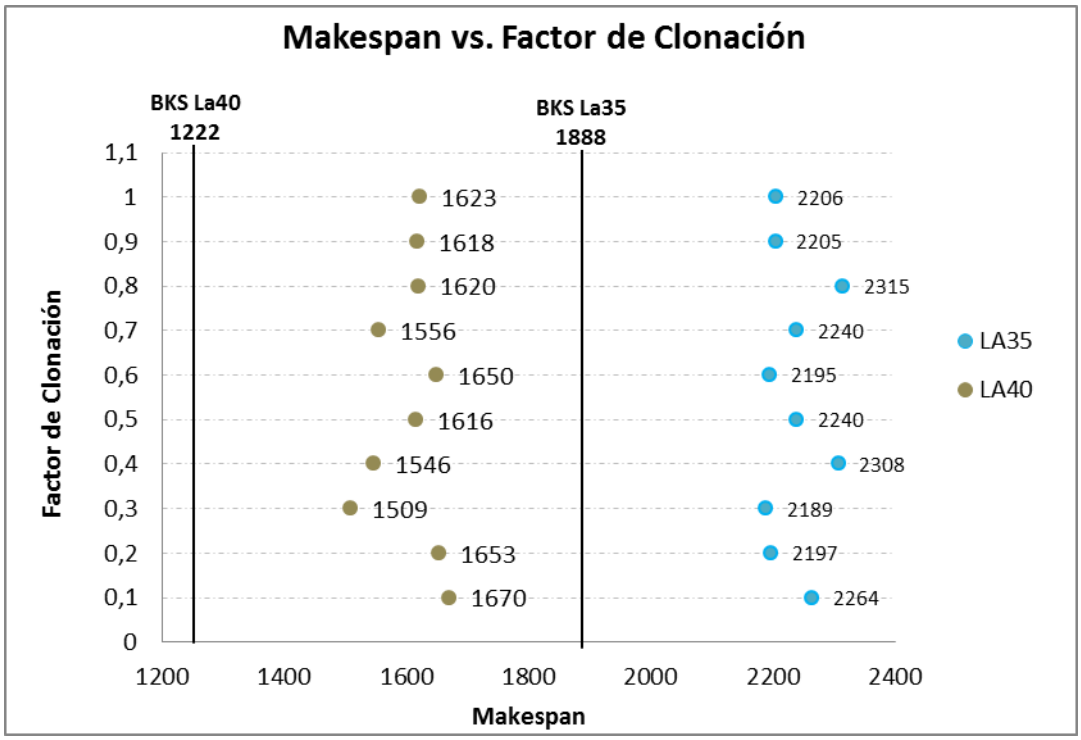


Figura 24: Factor de clonación vs. makespan (La35 – La40)



5.3.2.4 CLONALG: SENSIBILIDAD DEL NÚMERO DE RANDÓMICO DE CÉLULAS

El valor del número randómico de células corresponde al porcentaje de individuos respecto a la población que serían generados aleatoriamente con el fin de mantener la diversidad en la población.

El rango se encuentra entre 0 y 100% respecto a la población. El valor calculado que optimiza el makespan para la mayoría de instancias fue del 10%. Este valor cobra sentido pues valores pequeños en este parámetro no alteran negativamente la diversidad en la población, lo cual haría que el algoritmo se estancara en un espacio de búsqueda. Las siguientes gráficas muestran estos resultados:

Tabla 7: Sensibilidad del número randómico de células

Número Randómico de Células	La01 mxn **10x5**	La09 mxn **15x5**	La15 mxn **20x5**	La20 mxn **10x10**	La25 mxn **15x10**	La30 mxn **20x10**	La35 mxn **30x10**	La40 mxn **15x15**
0 %	666	951	1347	984	1223	1637	2182	1631
10 %	666	951	1289	936	1174	1623	2173	1585
20 %	666	971	1331	1021	1203	1653	2296	1629
30 %	666	956	1389	1022	1223	1644	2256	1611
40 %	699	953	1392	975	1210	1633	2223	1609
50 %	669	959	1396	981	1193	1633	2237	1600
60 %	671	975	1329	1001	1204	1621	2262	1615
70 %	678	951	1369	1002	1204	1634	2269	1622
80 %	677	954	1360	997	1236	1643	2274	1592
90 %	686	973	1340	995	1222	1622	2240	1635
100 %	693	960	1393	1000	1219	1647	2295	1617

Figura 25: Número randómico de células vs. makespan (La01 – La09)

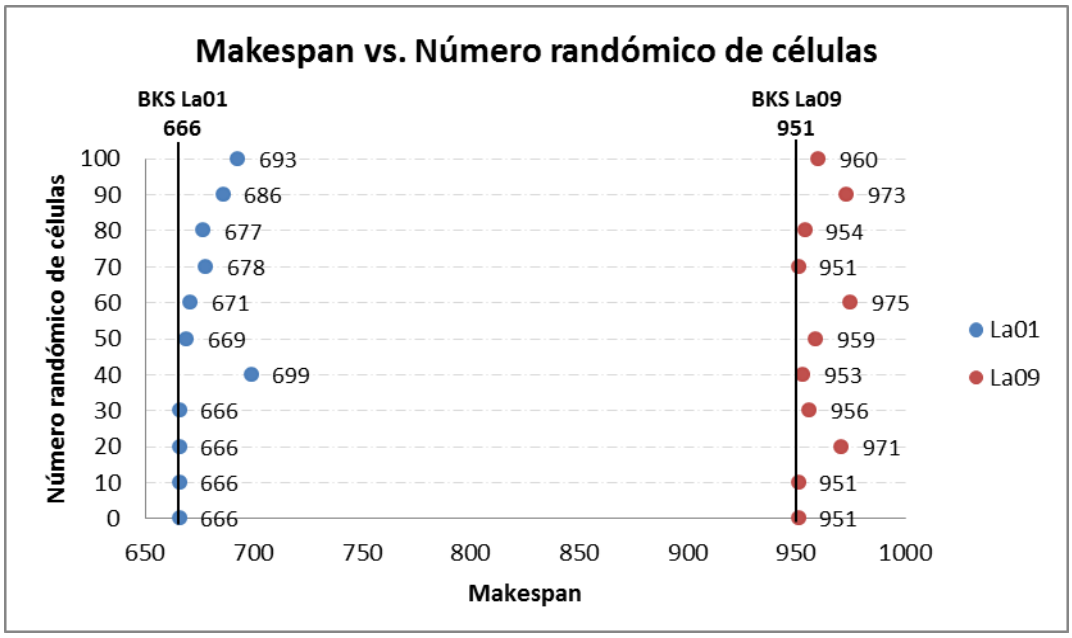


Figura 26: Número randómico de células vs. makespan (La15 – La20)

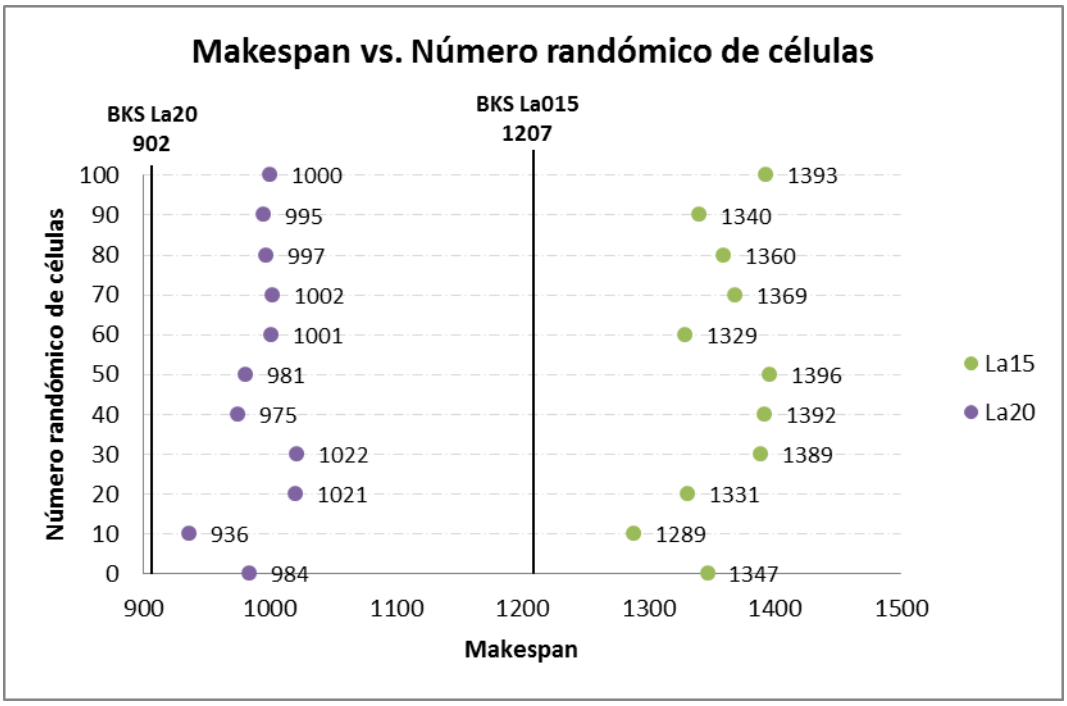


Figura 27: **Número randómico de células vs. makespan (La25- La30)**

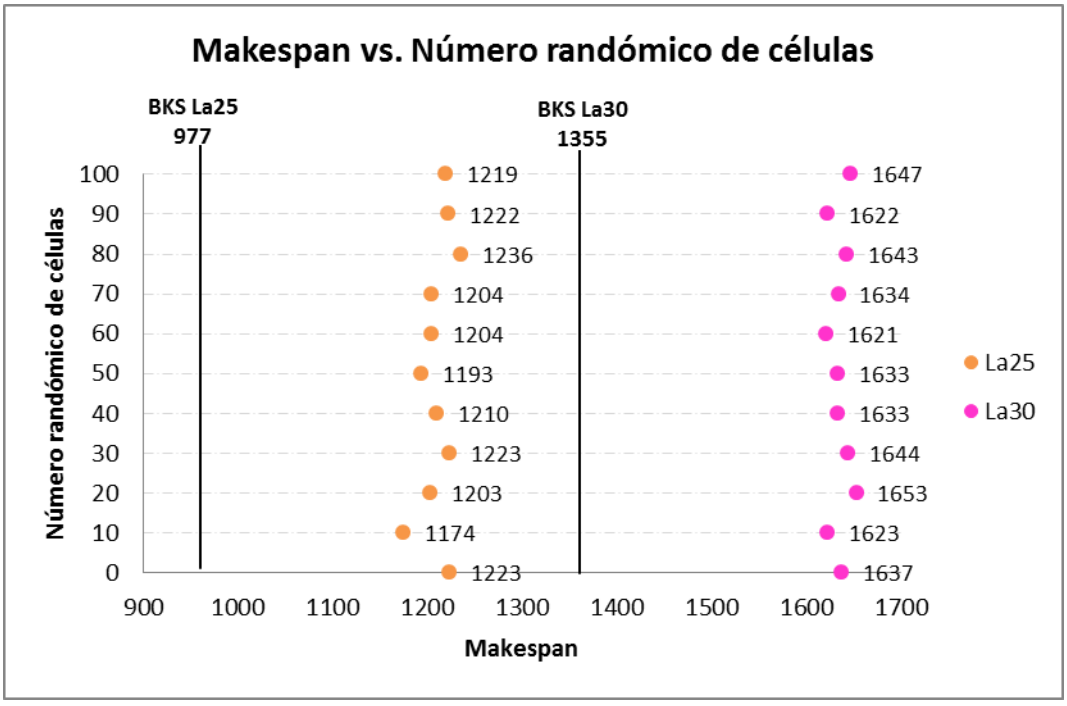
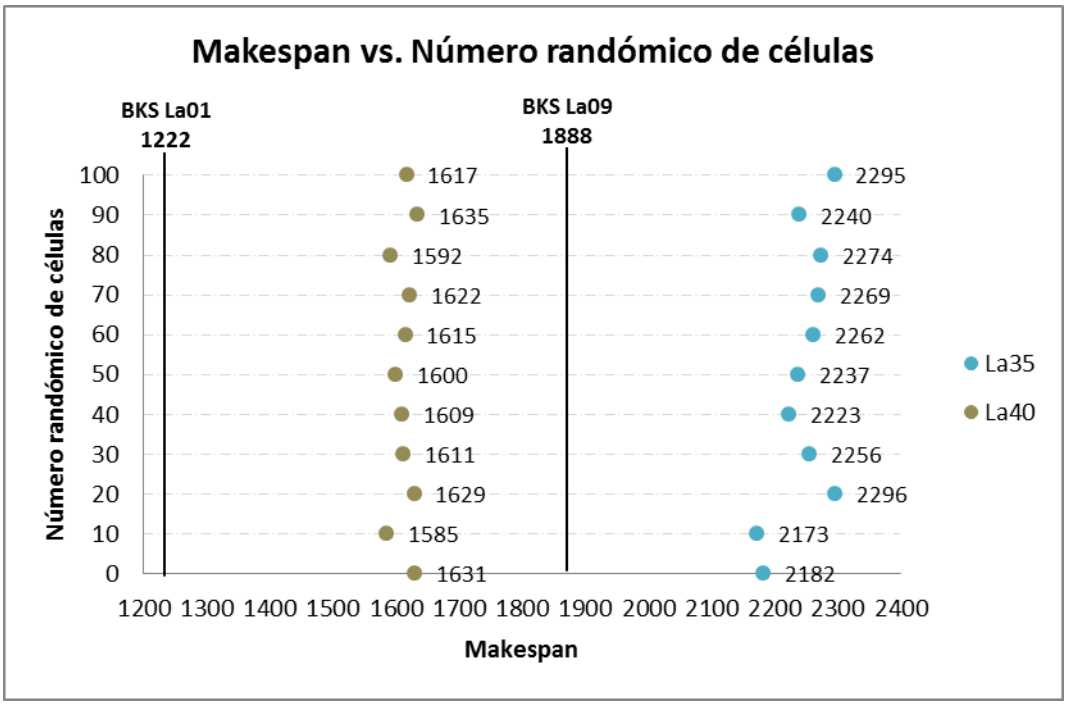


Figura 28: Número randómico de células vs. makespan (La35 – La40)



5.3.2.5 GRASP: SENSIBILIDAD MÁXIMO NÚMERO DE "NO MEJORAMIENTOS"

Este parámetro determina el número de veces que se realiza la búsqueda local. "El algoritmo utiliza un modelo estocástico de procedimiento de búsqueda local con un número fijo de iteraciones de "no mejoramientos" como la condición de parada" [31].

Luego del análisis de sensibilidad el factor se fijó en 1900, con un rango entre 0 a 2000.Las siguientes gráficas ilustran las pruebas:

Figura 29: Máximo número de no mejoras vs. makespan (La01)

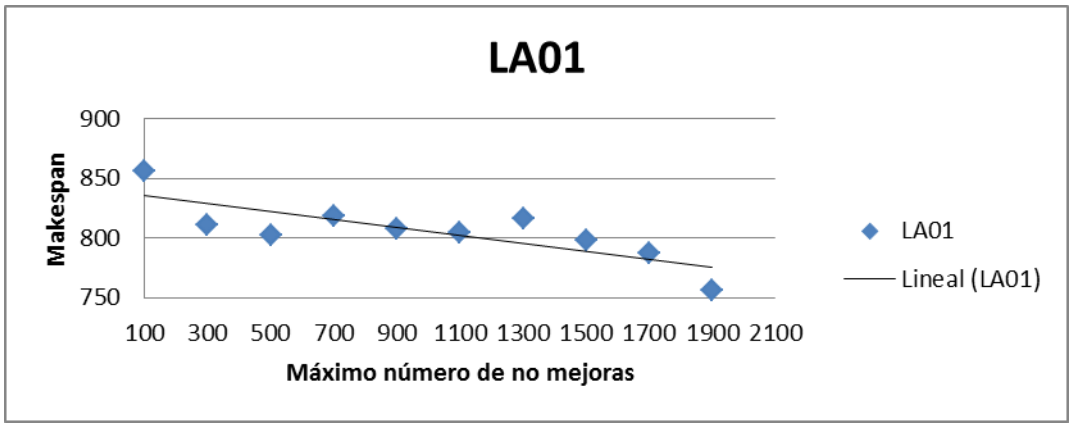


Figura 30: Máximo número de no mejoras vs. makespan (La09)

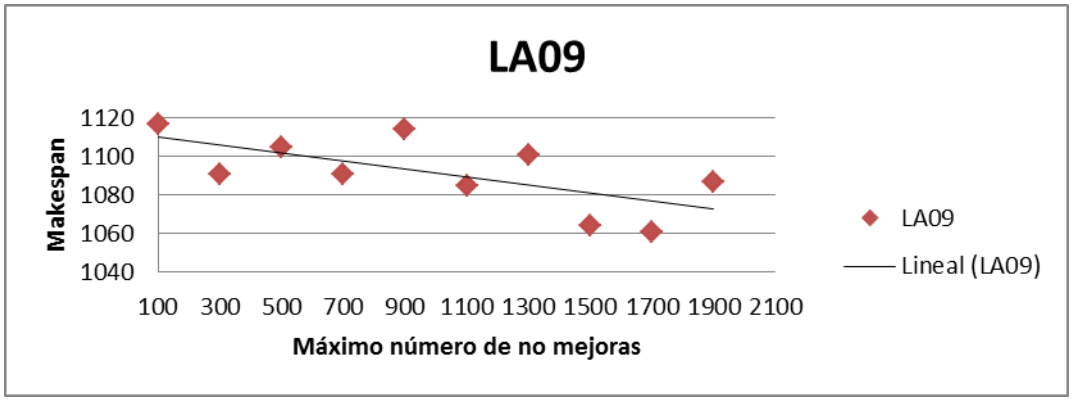


Figura 31: Máximo número de no mejoras vs. makespan (La15)

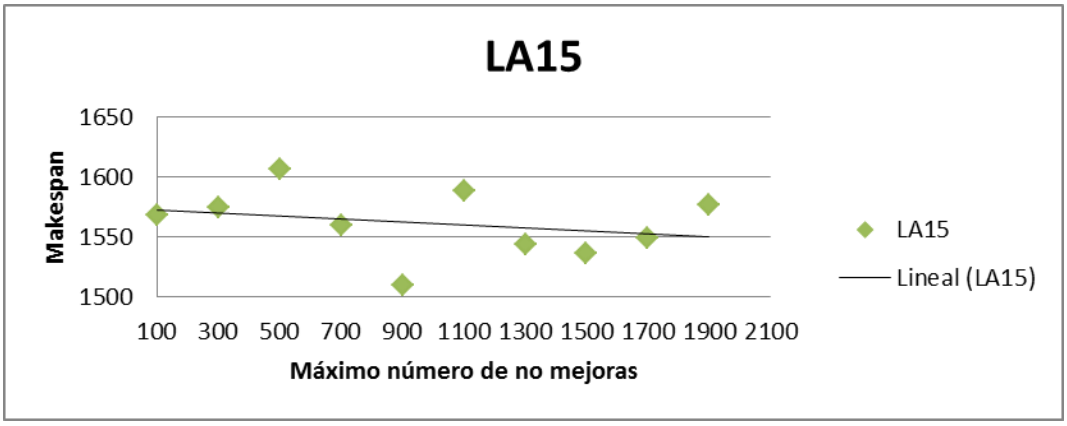


Figura 32: Máximo número de no mejoras vs. makespan (La20)

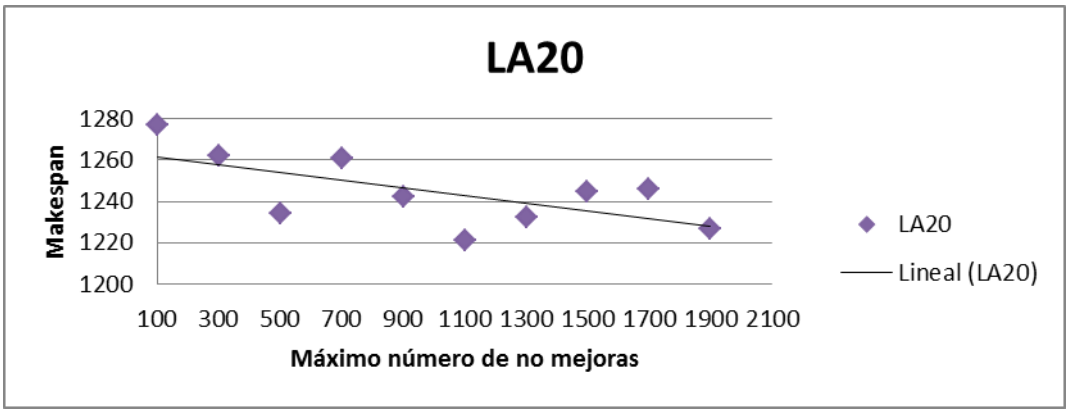


Figura 33: Máximo número de no mejoras. makespan (La25)

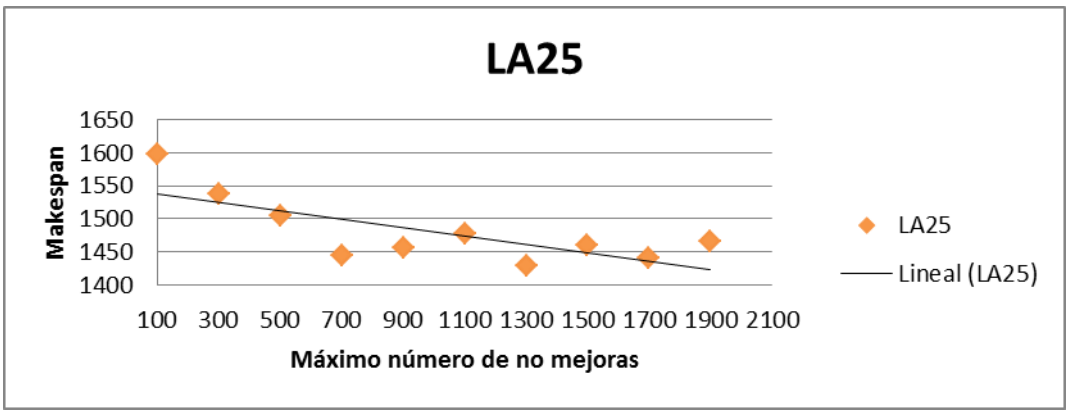


Figura 34: Máximo número de no mejoras vs. makespan (La30)

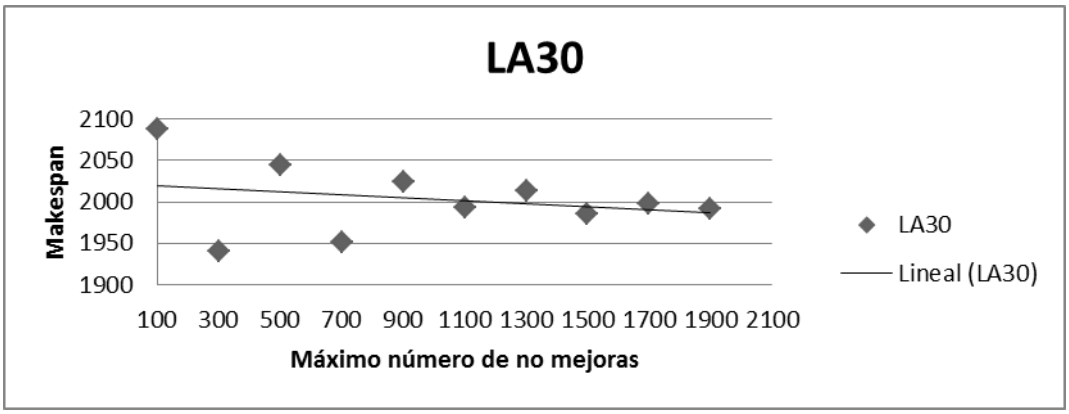


Figura 35: Máximo número de no mejoras vs. makespan (La35)

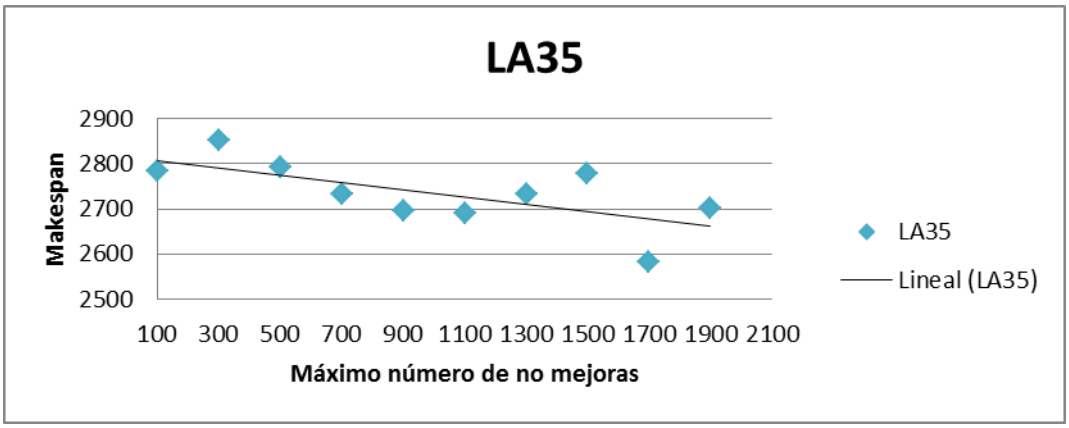
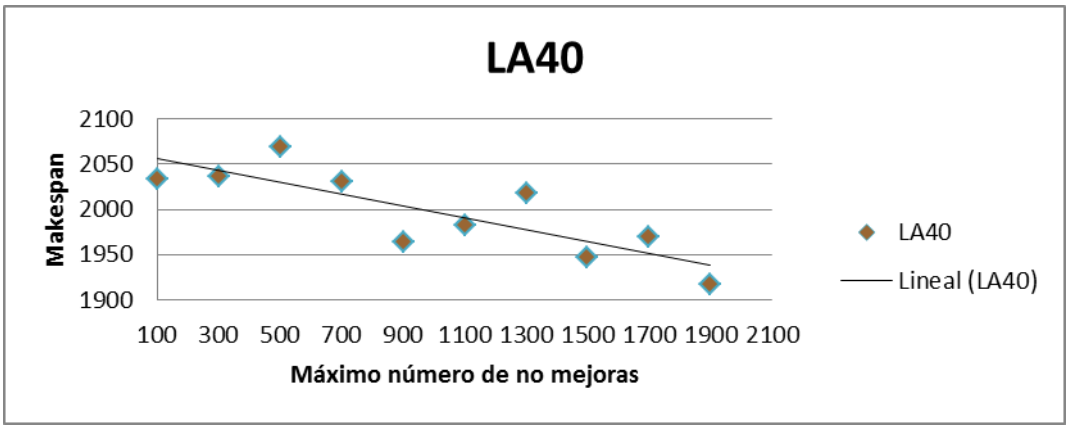


Figura 36: Máximo número de no mejoras vs. makespan (La40)

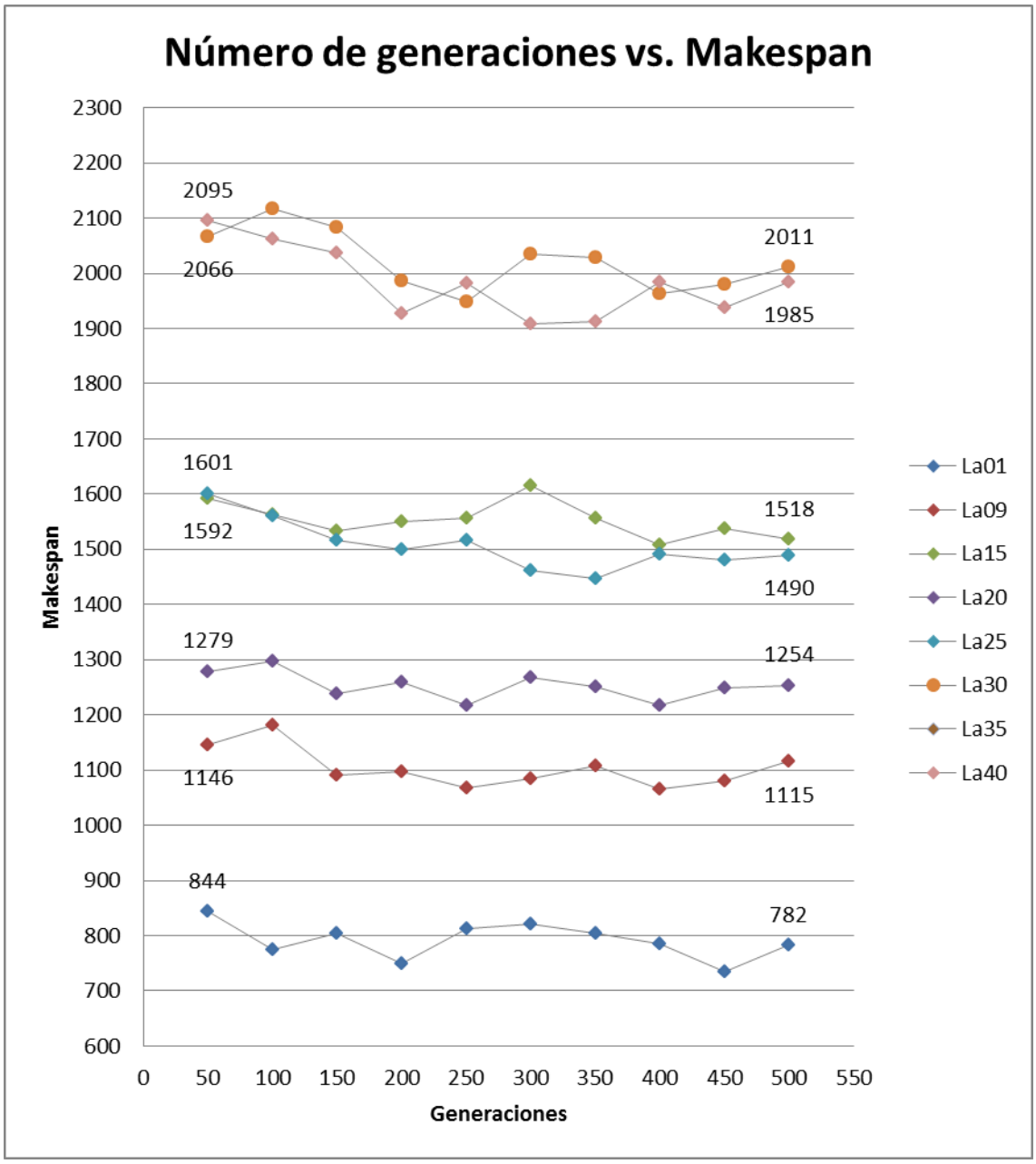


5.3.2.6 GRASP: SENSIBILIDAD NÚMERO DE GENERACIONES

El número de iteraciones representa la cantidad de veces que se repiten las dos fases del algoritmo GRASP, es decir, la fase de construcción y la fase de búsqueda local.

Este parámetro se estableció en400después de encontrar que el makespan se minimizaba con este valor para la mayoría de las instancias como se ve en la siguiente gráfica:

Figura 37: Generaciones vs. makespan de GRASP



5.3.2.7 GRASP: SENSIBILIDAD FACTOR ALPHA

Es el factor que determina la función de la lista restringida de candidatos (RCL o RestrictedCandidateList). El umbral α define la cantidad de codicia (Greediness) del mecanismo de construcción. En valores cercanos a 0 es codicioso, y valores cercanos a 1 es muy generalizado.

El factor alpha que se determinó fue 0,4 al optimizar el makespan y se ve en las siguientes gráficas:

Figura 38: Gráfica alpha vs. makespan (La01)

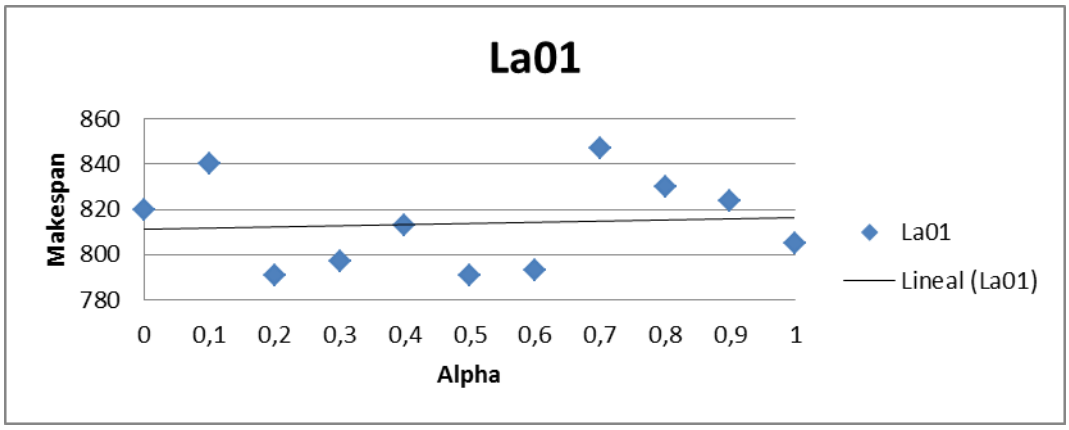


Figura 39: Gráfica alpha vs. makespan (LA09)

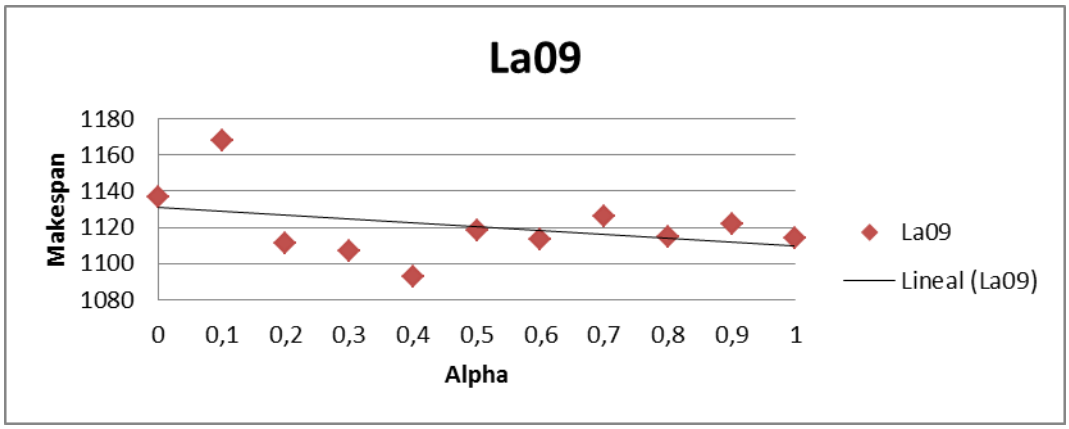


Figura 40: Gráfica alpha vs. makespan (La15)

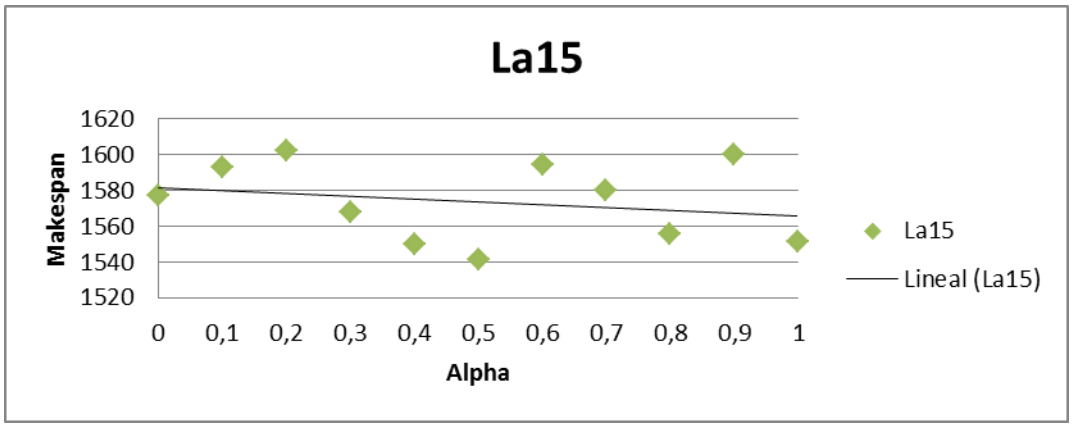


Figura 41: Gráfica alpha vs. makespan (La20)

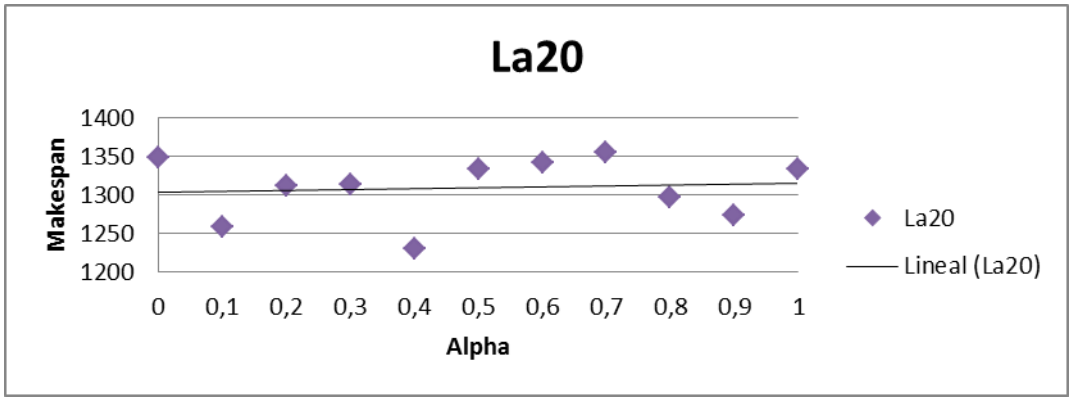


Figura 42: Gráfica alpha vs. makespan (La25)

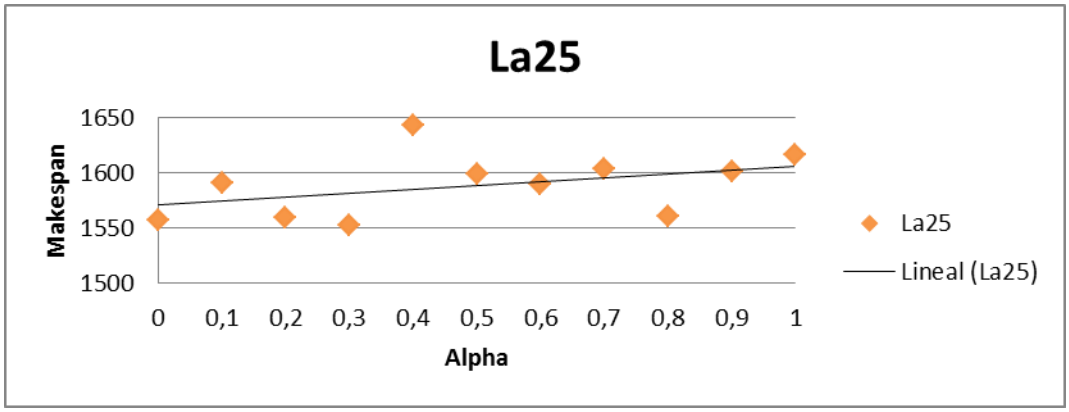


Figura 43: Gráfica alpha vs. makespan (La30)

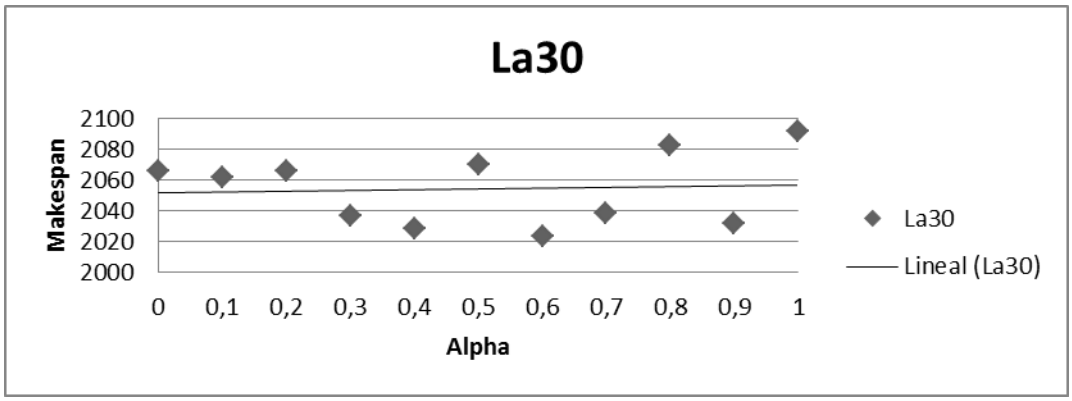


Figura 44: Gráfica alpha vs. makespan (La35)

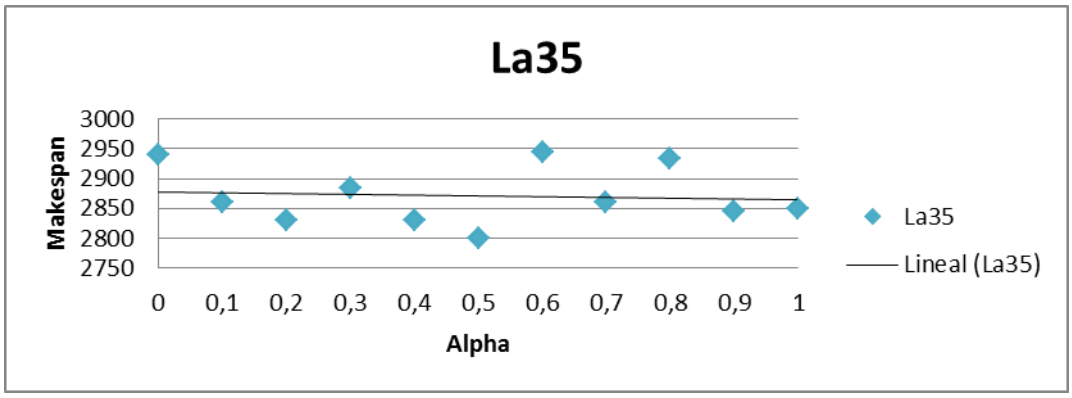
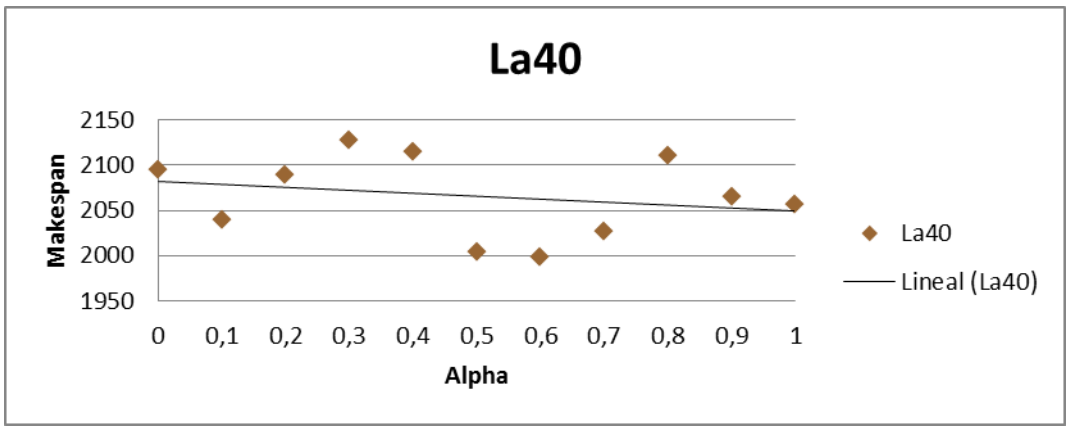


Figura 45: Gráfica alpha vs. Makespan (La40)



5.3.3 RESULTADOS A LOS PROBLEMAS DE LAS INSTANCIAS DE LAWRENCE (la)

Los resultados de las instancias se muestran en el orden que se describieron previamente (Ver tablas 8, 10, 12, 14). Las medidas estadísticas que se reportan son el promedio $\bar{x}$, la varianza S^2y la desviación estándar S. Todas las medidas son realizadas con el mejor resultado (C_{max})en cada una de las 20 ejecuciones de los algoritmos de Clonalg y GRASP (Ver tablas 9, 11, 13, 15).

Adicional a esto se graficó la variación del makespan de cada uno de los algoritmos durante las 20 ejecuciones, esto con el objetivo de observar el comportamiento de los contendientes y tener una idea más clara de cual algoritmo tenía un comportamiento más estable, el análisis de estos resultados se encuentran en la sección 5.5 (Ver figuras 46 - 53).

Tabla 8: Cuadro comparativo de makespan entre Clonalg y GRASP

	Instancia LA01	
	Clonalg	GRASP
Ejecuciones	C_{max}	C_{max}
0	687	818
1	666	852
2	675	785
3	666	867
4	667	827
5	685	817
6	666	823
7	670	825
8	693	868
9	692	815
10	666	767
11	682	826
12	674	791
13	677	808
14	680	793
15	687	848
16	671	787
17	688	804
18	666	814
19	677	819

	Instancia LA09	
	Clonalg	GRASP
Ejecuciones	C_{max}	C_{max}
0	951	1211
1	988	1143
2	951	1100
3	956	1146
4	956	1163
5	960	1135
6	980	1115
7	967	1145
8	965	1164
9	971	1101
10	951	1167
11	970	1180
12	951	1093
13	970	1179
14	951	1105
15	951	1143
16	979	1149
17	952	1165
18	952	1181
19	951	1134

Figura 46: Gráfica de makespan entre Clonalg y GRASP (La01)

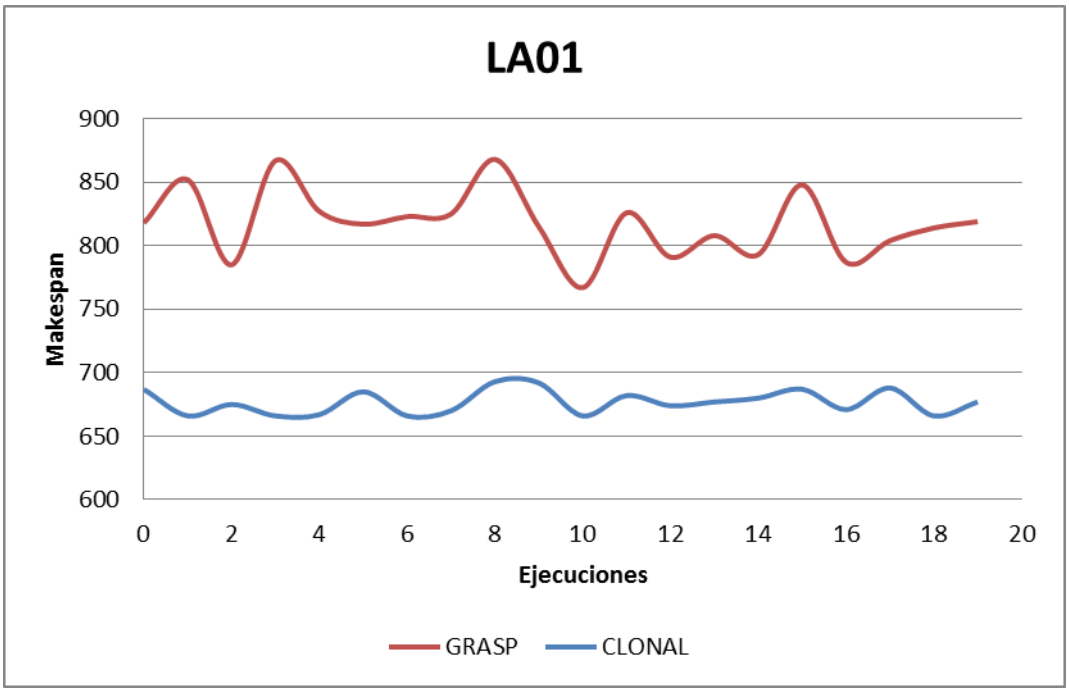


Figura 47: Gráfica de makespan entre Clonalg y GRASP (La09)

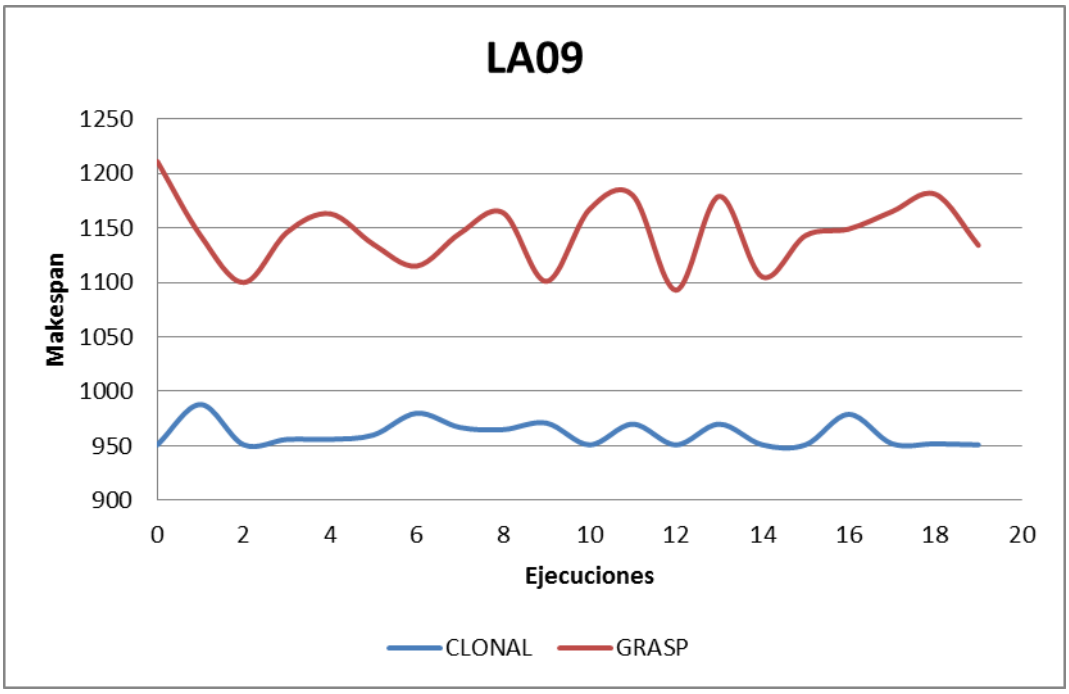


Tabla 9: Comparación estadística

	Instancia LA01	
	Clonalg	**GRASP**
$\overline{x}$	676,75	817,7
S	9,2	25,99
S^2	85,09	675,11

	Instancia LA09	
	Clonalg	**GRASP**
$\overline{x}$	961,15	1145,95
S	11,49	30,83
S^2	132,03	949,75

Tabla 10: Cuadro comparativo de makespan entre Clonalg y GRASP

	Instancia LA15	
	Clonalg	**GRASP**
Ejecuciones	C_{max}	C_{max}
0	1332	1564
1	1333	1568
2	1348	1605
3	1347	1629
4	1320	1566
5	1345	1638
6	1347	1564
7	1335	1585
8	1324	1589
9	1318	1582
10	1336	1615
11	1298	1565
12	1347	1551
13	1291	1665
14	1357	1547
15	1339	1552

	Instancia LA20	
	Clonalg	**GRASP**
Ejecuciones	C_{max}	C_{max}
0	1037	1310
1	1046	1309
2	985	1273
3	1013	1251
4	955	1276
5	1003	1262
6	1001	1293
7	1017	1263
8	1005	1243
9	1034	1280
10	994	1264
11	987	1269
12	1005	1234
13	1029	1243
14	1000	1296
15	1032	1266

16	1279	1592	16	1032	1218
17	1339	1533	17	981	1328
18	1330	1535	18	1052	1234
19	1323	1624	19	982	1301

Figura 48: Gráfica del makespan entre Clonalg y GRASP (La15)

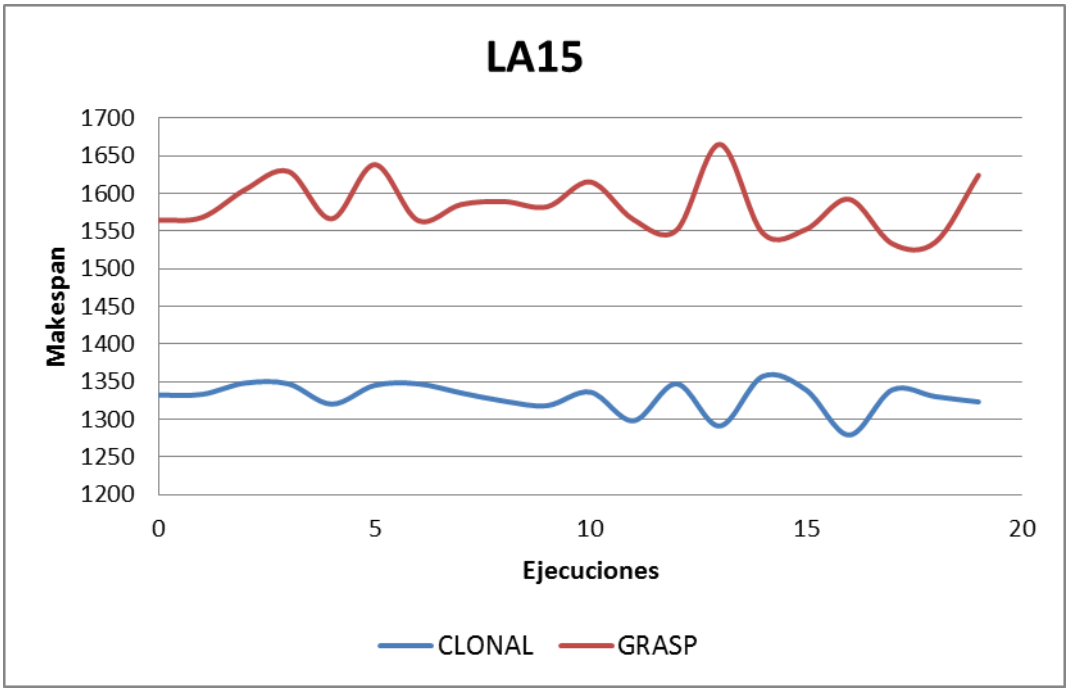


Figura 49: Gráfica de makespan entre Clonalg y GRASP (La20)

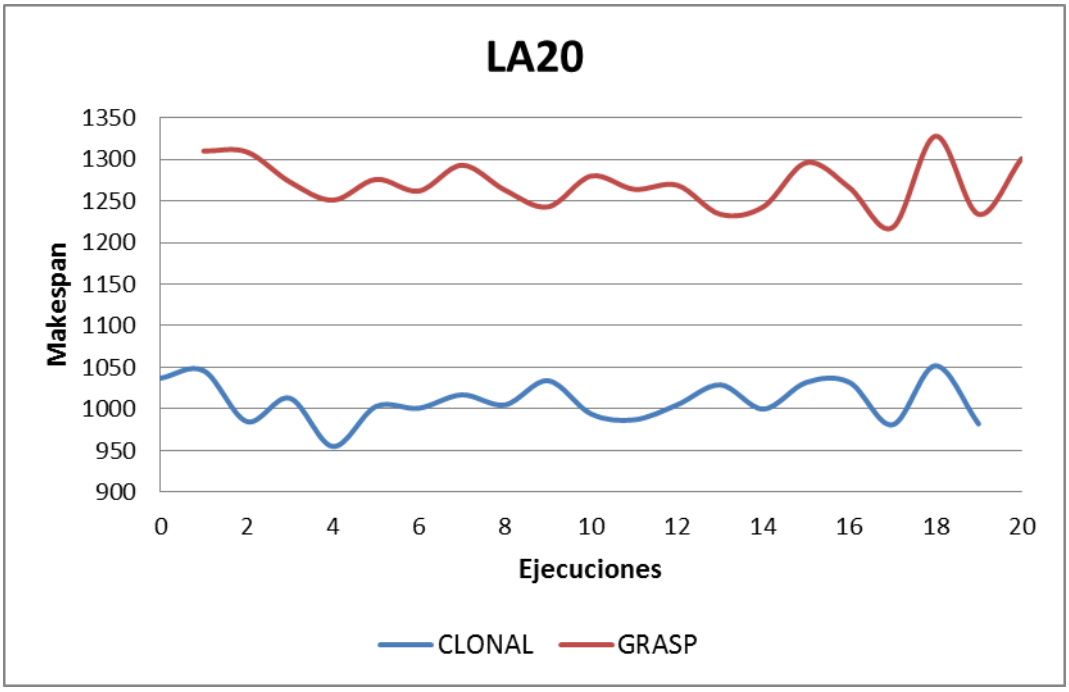


Tabla 11: Comparación estadística

	Instancia LA15	
	Clonalg	**GRASP**
$\bar{x}$	1329,4	1583,45
S	19,81	35,22
S^2	392,44	1240,84

	Instancia 20	
	Clonalg	**GRASP**
$\bar{x}$	1009,5	1270,65
S	24,59	28,345
S^2	604,35	803,43

Tabla 12: Cuadro comparativo de makespan entre Clonalg y GRASP

	Instancia LA25	
	Clonalg	GRASP
Ejecuciones	C_{max}	C_{max}
0	1222	1489
1	1196	1502
2	1258	1479
3	1194	1557
4	1219	1506
5	1170	1551
6	1220	1523
7	1210	1522
8	1207	1508
9	1245	1529
10	1159	1476
11	1214	1530
12	1171	1540
13	1197	1458
14	1209	1575
15	1256	1534
16	1213	1528
17	1209	1527
18	1157	1553
19	1155	1507

	Instancia LA30	
	Clonalg	GRASP
Ejecuciones	C_{max}	C_{max}
0	1597	2138
1	1618	2019
2	1663	2099
3	1663	1976
4	1621	2075
5	1714	2112
6	1657	1996
7	1677	2127
8	1648	2013
9	1737	2034
10	1649	2027
11	1703	2064
12	1581	2093
13	1671	2039
14	1648	2080
15	1625	2029
16	1642	2117
17	1620	2113
18	1705	2033
19	1647	2060

Figura 50: Gráfica de makespan entre Clonalg y GRASP (La25)

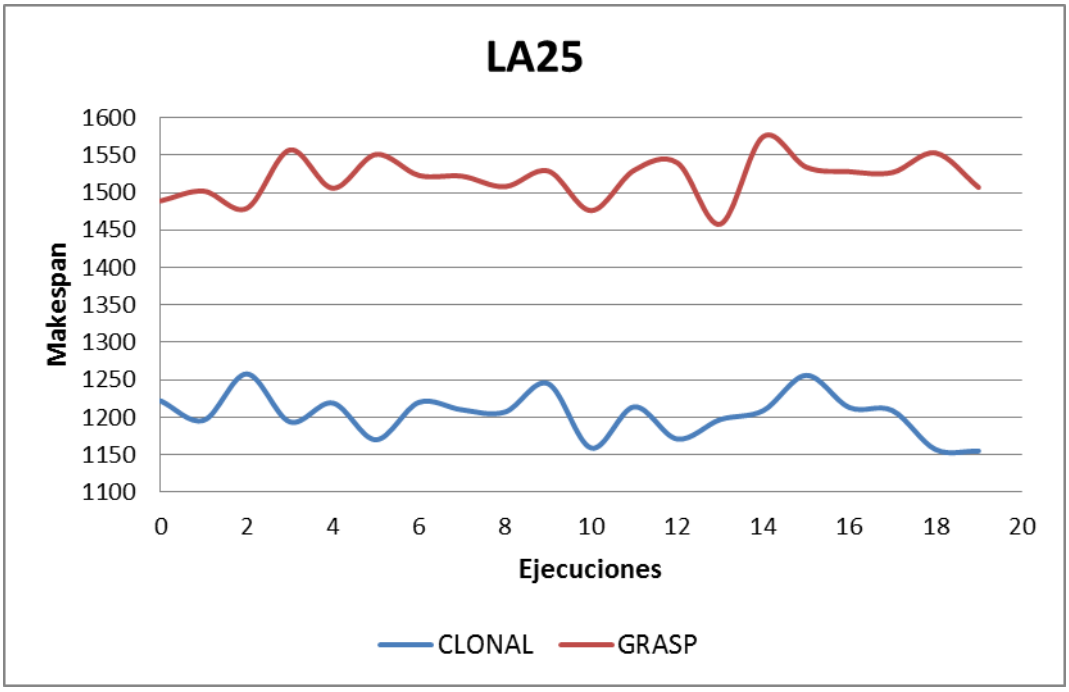


Figura 51: Gráfica de Makespan entre Clonalg y GRASP (La30)

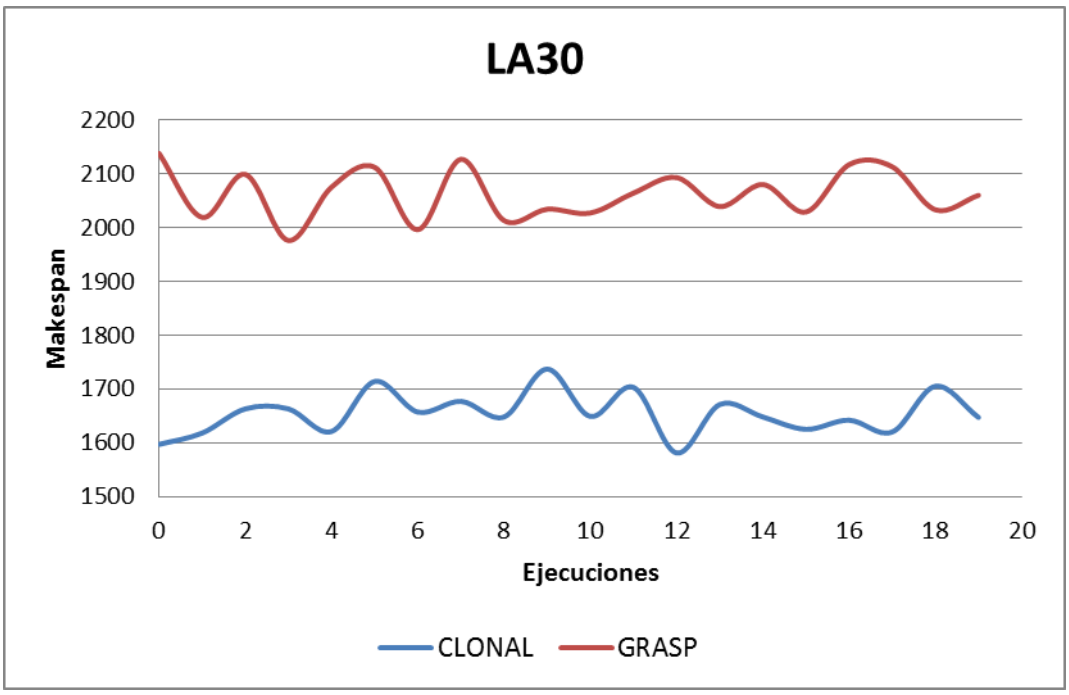


Tabla 13: Comparación estadística

	LA25	
	Clonalg	**GRASP**
$\overline{x}$	1204,05	1519,7
S	29,56	28,81
S^2	873,75	830,01

	LA30	
	Clonalg	**GRASP**
$\overline{x}$	1654,3	2062,2
S	38,55	45,52
S^2	1486,41	2071,36

Tabla 14: Cuadro comparativo de makespan entre Clonalg y GRASP

	Instancia LA35	
	CLONAL	**GRASP**
Ejecuciones	C_{max}	C_{max}
0	2265	2625
1	2282	2791
2	2311	2797
3	2276	2831
4	2247	2702
5	2243	2911
6	2302	2710
7	2182	2705
8	2278	2735
9	2244	2820
10	2286	2833
11	2203	2712
12	2359	2810
13	2265	2792
14	2227	2730
15	2305	2750

	Instancia LA40	
	CLONAL	**GRASP**
Ejecuciones	C_{max}	C_{max}
0	1566	1960
1	1627	1991
2	1610	1960
3	1601	1897
4	1579	2095
5	1540	2070
6	1615	1953
7	1615	1982
8	1622	2011
9	1576	2032
10	1662	1990
11	1616	2002
12	1599	1949
13	1631	1922
14	1635	1998
15	1598	2003

16	2211	2705	16	1645	1982
17	2225	2833	17	1586	2016
18	2179	2887	18	1552	2060
19	2269	2770	19	1582	2004

Figura 52: Gráfica de makespan entre Clonalg y GRASP (La35)

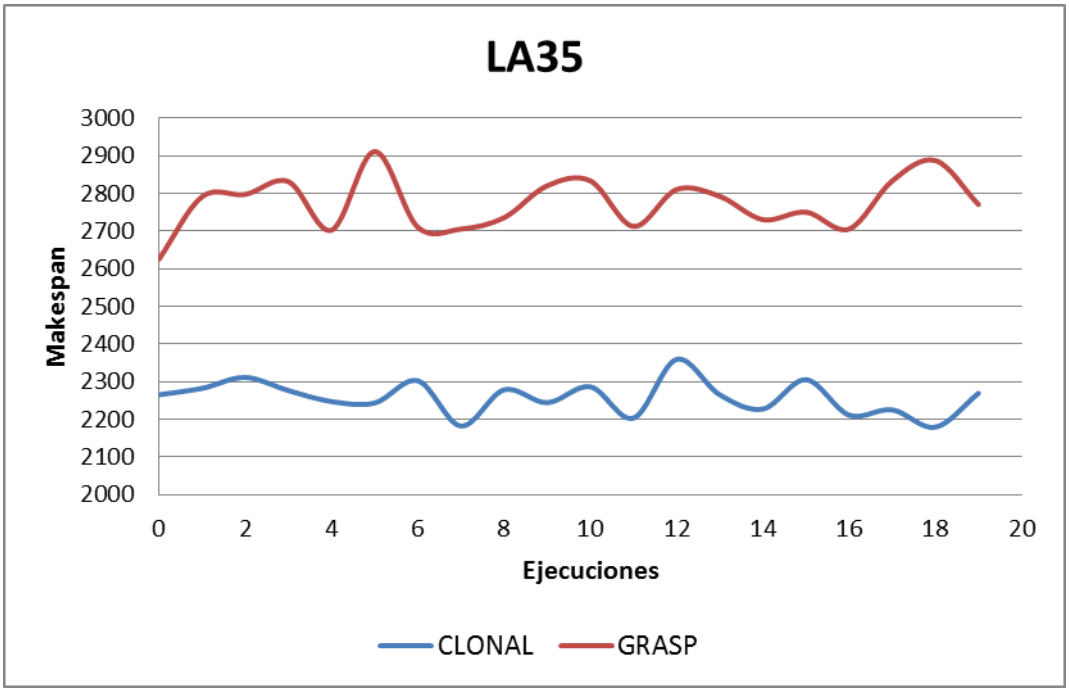


Figura 53: Gráfica de makespan entre Clonalg y GRASP (La40)

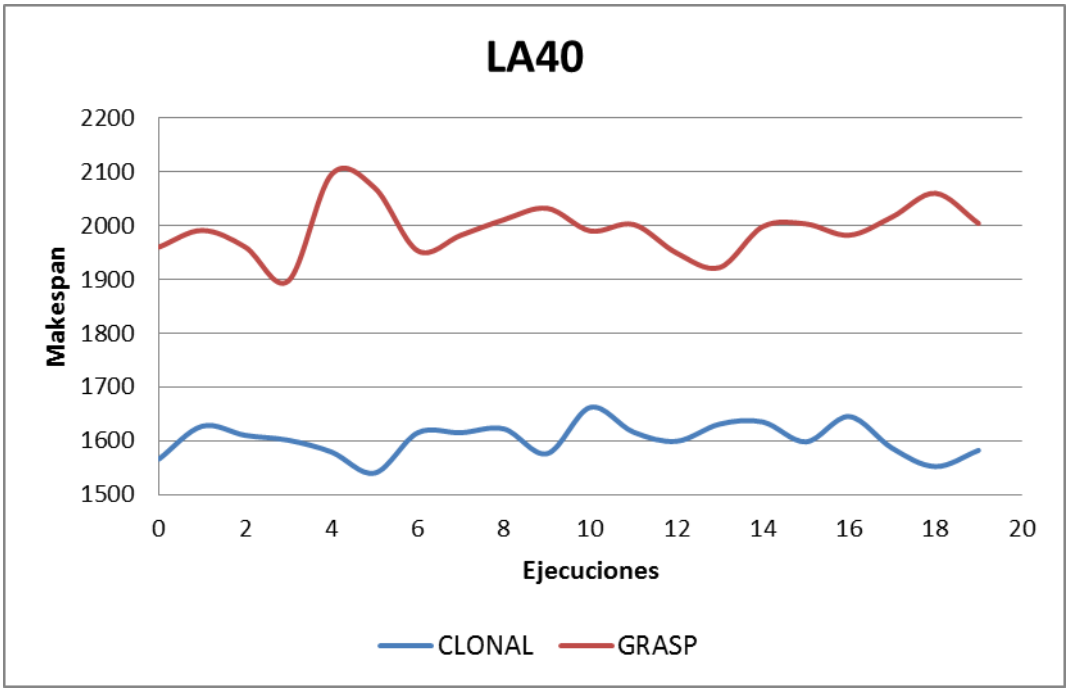


Tabla 15: Comparación estadística

	LA 35	
	Clonalg	**GRASP**
$\bar{x}$	2257,95	2772,45
S	44,33	68,96
S^2	1965,05	4757,58

	LA40	
	Clonalg	**GRASP**
$\bar{x}$	1602,85	1993,85
S	30,43	46,763
S^2	925,73	2186,73

5.4 CONTENDIENTE (GRASP)

En esta sección se describe el algoritmo contra el contendiente GRASP, el cual se hace una comparación más extensa. La comparación se lleva acabo en términos del mejor resultado obtenido y su desviación.

El contendiente contra el cual se comparó nuestro Sistema Inmune Artificial (SIA) es GRASP (Procedimiento de Búsqueda Miope Aleatoria y Adaptativa) [38]. En la tabla 16 se pueden ver valores bajos en la desviación para las primeras 14 instancias con las que fue probado el algoritmo Clonalg.

En contraste, GRASP presentó valores altos para el mismo número de instancias, cabe destacar que la desviación en este caso se da respecto al BKS de la instancia 15 a la 40 el algoritmo de Clonalg

Tabla 16: Comparación entre Clonalg y GRASP con respecto al BKS.

Problemas	m	n	**BKS**	**Clonalg**	**Desviación**	**GRASP**	**Desviación**
la01	10	5	666	666	0,00%	767	15,17%
la02	10	5	655	678	3,51%	799	17,85%
la03	10	5	597	638	6,87%	739	15,83%
la04	10	5	590	614	4,07%	733	19,38%
la05	10	5	593	593	0,00%	593	0,00%
la06	15	5	926	926	0,00%	1025	10,69%
la07	15	5	890	917	3,03%	1068	16,47%
la08	15	5	863	863	0,00%	1022	18,42%
la09	15	5	951	951	0,00%	1093	14,93%
la10	15	5	958	958	0,00%	1012	5,64%
la11	20	5	1222	1222	0,00%	1343	9,90%

la12	20	5	1039	1039	0,00%	1190	14,53%
la13	20	5	1150	1150	0,00%	1304	13,39%
la14	20	5	1292	1292	0,00%	1408	8,98%
la15	20	5	1207	1279	5,97%	1533	19,86%
la16	10	10	945	1029	8,89%	1204	17,01%
la17	10	10	784	842	7,40%	1074	27,55%
la18	10	10	848	928	9,43%	1100	18,53%
la19	10	10	842	941	11,76%	1163	23,59%
la20	10	10	902	955	5,88%	1218	27,54%
la21	15	10	1046	1251	19,60%	1555	24,30%
la22	15	10	927	1093	17,91%	1406	28,64%
la23	15	10	1032	1182	14,53%	1462	23,69%
la24	15	10	935	1148	22,78%	1424	24,04%
la25	15	10	977	1155	18,22%	1458	26,23%
la26	20	10	1218	1518	24,63%	1860	22,53%
la27	20	10	1235	1509	22,19%	1979	31,15%
la28	20	10	1216	1481	21,79%	1887	27,41%
la29	20	10	1157	1463	26,45%	1858	27,00%
la30	20	10	1355	1581	16,68%	1976	24,98%
la31	30	10	1784	2001	12,16%	2538	26,84%
la32	30	10	1850	2122	14,70%	2576	21,39%
la33	30	10	1719	1983	15,36%	2373	19,67%
la34	30	10	1721	2089	21,38%	2505	19,91%
la35	30	10	1888	2179	15,41%	2625	20,47%
la36	15	15	1268	1556	22,71%	1892	21,59%
la37	15	15	1397	1709	22,33%	2073	21,30%
la38	15	15	1196	1528	27,76%	1905	24,67%
la39	15	15	1233	1516	22,95%	1910	25,99%
la40	15	15	1222	1540	26,02%	1897	23,18%

Tabla 17: Comparación de la cantidad de mejoras de las instancias

	GRASP		
	Ganados	Empatados	Perdidos
CLONALG	39	1	0

Figura 54: Desviación de los algoritmos con respecto al BKS

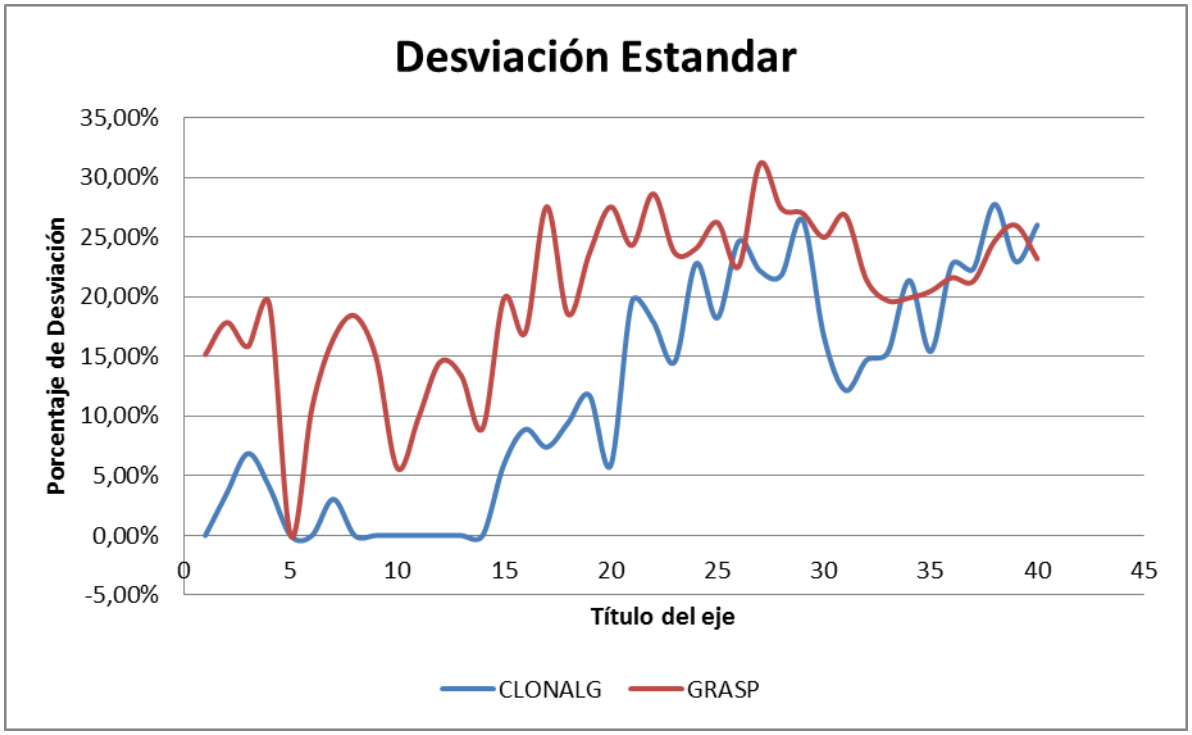


Figura 55: Resultados de makespan de cada algoritmo con respecto al BKS

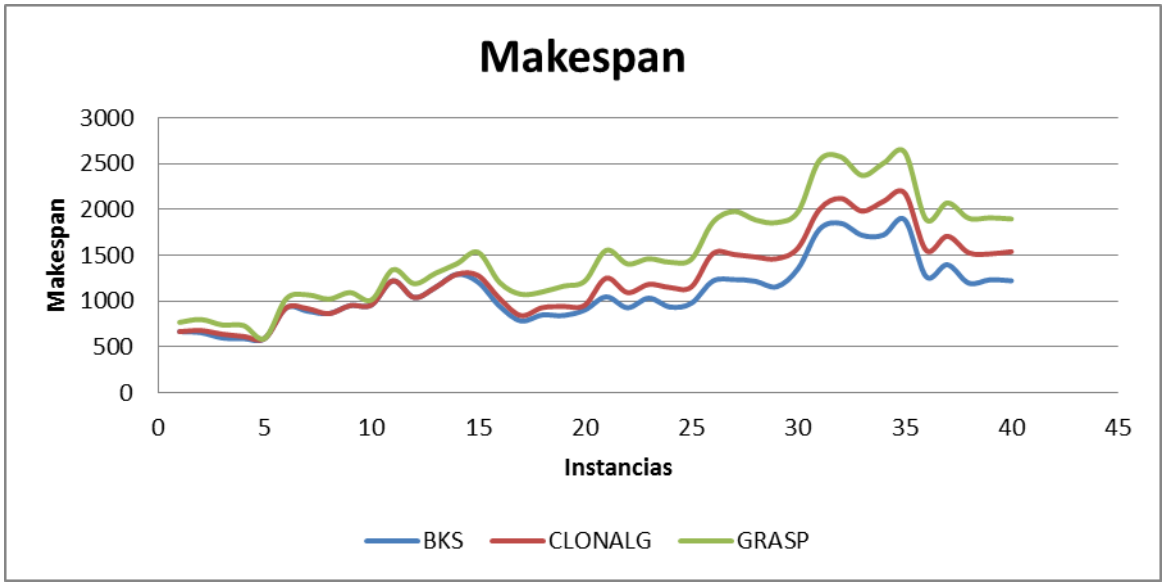


Tabla 18: Desviación Estándar Clonalg y GRASP.

DESVIACIÓN ESTÁNDAR	
Clonalg	11.81%
GRASP	20.01%

Figura 56: Error relativo de cada algoritmo

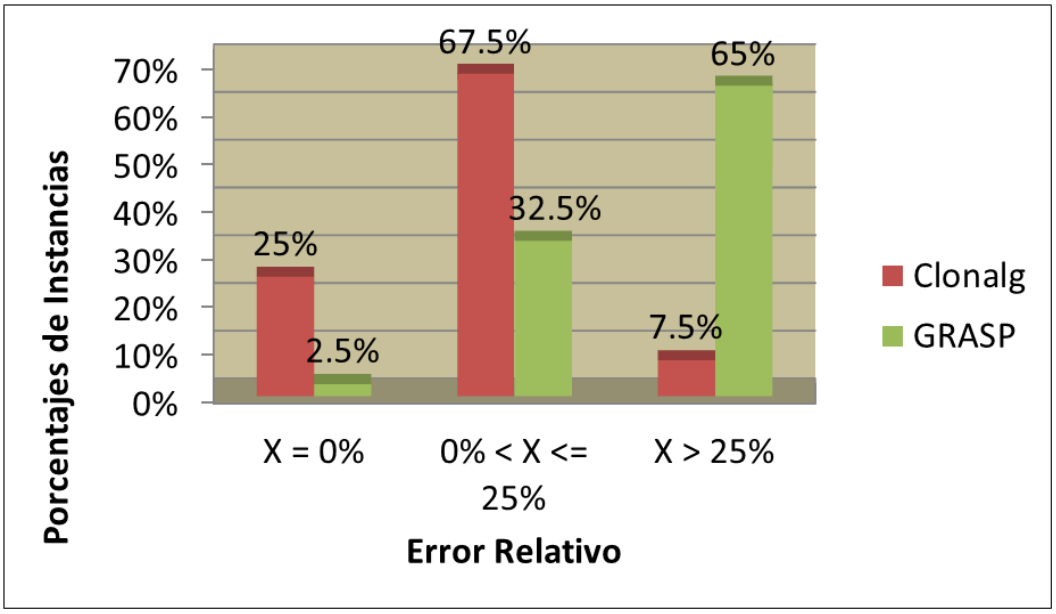
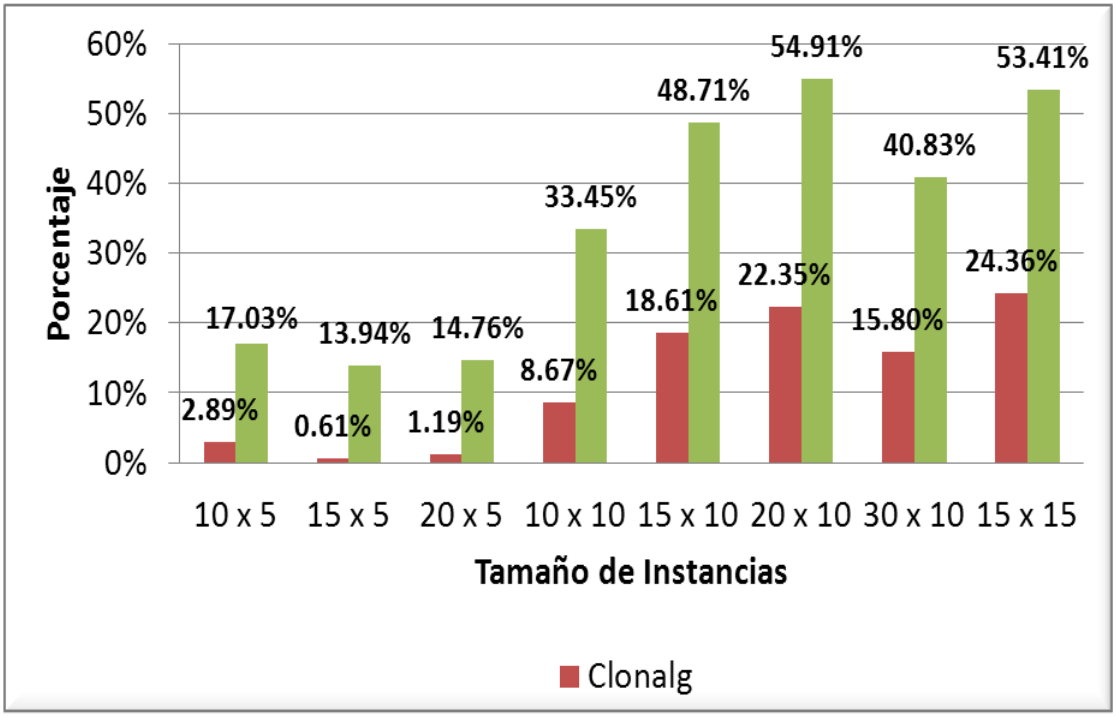


Figura 57: Error relativo medio por tamaño de instancia.



5.5 ANÁLISIS DE RESULTADOS

En la sección 5.3 se mostraron los resultados a 8 problemas de Lawrence en los cuales se les aplicaron los algoritmos de Clonalg y GRASP.

En las tablas de la sección 5.3.3 se realizó un paralelo entre ambos algoritmos y sus resultados para cada una de las 20 ejecuciones realizadas en cada una de las 8 instancias seleccionadas. A continuación se graficaron dichos resultados y se observó:

- Clonalg permite resolver Job Shop Scheduling y ofrece mejores resultados en cuanto a la minimización del makespan; comparado GRASP superándolo en 39 de las 40 instancias (Ver tabla 17). Además, Clonalg presenta una menor dispersión que el algoritmo GRASP. Esto se muestra con la desviación estándar que en el algoritmo Clonalg fue 11,81% y en GRASP fue 20,01% (Ver Tabla 18).

- El algoritmo Clonalg igualó la mejor solución conocida en el 25% de las instancias de Lawrence, mientras que el algoritmo GRASP lo logró para el 2,5% (Ver Figura 56). El error relativo medio crece conforme se incrementa el tamaño de las instancias (Ver Figura 57).

Con el fin de comparar no solo los resultados del makespan sino también el compartimiento de ambas muestras, se calculó el promedio, la varianza y la desviación estándar, con lo que se evidenció que el algoritmo de Clonalg tiene una distribución más homogénea que la del contendiente.

Por último en la sección 5.4 se compararon los resultados de los dos algoritmos implementados con la mejor solución conocida es decir el BKS, para los cuales se calculó únicamente la desviación estándar, puesto que esta medida estadística permite mostrar el grado de dispersión de los datos con respecto al valor promedio.

Con lo anterior se demostró nuevamente que el algoritmo inmune artificial Clonalg tuvo un mejor comportamiento, dado que su dispersión fue menor lo que indica que los resultados obtenidos se encontraban cerca de la mejor solución conocida.

6 ESCENARIO DE APLICACIÓN DEL PROBLEMA DE JOB SHOP

Las instancias usadas en este trabajo, fueron propuestas por Stephen R. Lawrence en un artículo no publicado [39], las cuales luego fueron difundidas ampliamente por J. E Beasley [40].

En la librería de Beasley que se encuentra en la página web de Brunel University [41] se hallaron varios problemas de "Operational Research"[6] (OR), entre estos el Problema de Job Shop Scheduling.

En esta librería el autor recopiló instancias planteadas por múltiples autores para desarrollar el Job Shop Scheduling, entre otros problemas de OR. Esto con la intención de incentivar la investigación y la publicación de resultados con instancias de acceso generalizado que permitieran el avance en la materia.

Estas instancias se convirtieron en el punto de partida para muchas de las pruebas que realizan los autores en la literatura de este campo. Como se mencionó en la sección 5.2, estas instancias son 40 problemas de 8 diferentes tamaños propuestos por Lawrence, que Applegate y Cook [36]nombraron "LA".

Estas instancias de Lawrence fueron propuestas para cualquier aplicación del problema de Job Shop que cumpliera con las especificaciones donde hay $n$el número de trabajos que deben recorrer $m$número de máquinas.

Particularmente, uno de estos campos de aplicación es el de procesamiento computacional, el cual ha tomado mucha relevancia. Un ejemplo de esto se encuentra en el trabajo de Streit [42] , donde se buscan estrategias para abordar el problema del Job Shop en un clúster HPC. En este trabajo se implementan y evalúan tres estrategias: El principio de primero en llegar,

[6]Operations Research.Siglas en ingles para referirse a Investigación de Operaciones

primero en ser atendido (FCFS: First Come FirstServe), Principio del trabajo más corto, primero en ser atendido (SJF: Shortest Job First), y Principio trabajo mas largo, primer trabajo en ser atendido (LJF: Longest Job First). Éstas básicamente determinan el orden de procesamiento de las tareas. También realizan pruebas con la carga de trabajo de las máquinas, como estrategia de optimización.

Otros estudios están enfocados directamente en el problema del Job shop Scheduling en paralelo, que es una rama reciente que aborda el procesamiento en paralelo de la información en máquinas “paralelas y distribuidas” (Grids), como lo plantea Frachtemberg y Schwiegelshohm [43]. Este es todavía un campo muy amplio sin explorar pues hay muchas maneras de abordarlo y en su trabajo, estos autores mencionan estrategias para enfocar las soluciones. Entre estas se encuentra la optimización de la carga de trabajo, la evaluación de la heterogeneidad de las máquinas, las métricas y evaluaciones de los procesos de optimización hechos por las máquinas, etc.

De forma simultánea se han desarrollado los sistemas inmunes artificiales. Estos tienen su propio campo de aplicación al JSP que ha generado una rama de estudio particularmente interesante. Este avance se ha dado muy recientemente con estudios como los Cruz-Chávez, Et. Al [44] donde se presenta un algoritmo híbrido paralelo evolutivo ejecutado en un entorno de red.

El algoritmo realiza búsquedas locales utilizando Simmulated Annealing (recocido simulador) dentro de un algoritmo genético para resolver el problema de Job Shop Scheduling. Se muestran los resultados experimentales que se obtuvieron del algoritmo que se aplicó en un mini-Grid. Este fue implementado mediante la vinculación de dos clusters en diferentes ubicaciones geográficas de México (Morelos y Veracruz).

En el presente capítulo se busca caracterizar un posible escenario de aplicación de los sistemas inmunes artificiales al problema del Job Shop Scheduling. Para lo cual se toma como base la evaluación de cada instancia de Lawrence[7] en un algoritmo tipo Clonalg. Mientras se realizaban los análisis de sensibilidad y de recolección de datos para la prueba del contendiente[8]se observó que dicho proceso se comporta como un problema de Job Shop Scheduling (JSP). Para lo cual las instancias son equivalentes a los trabajos del JSP, las máquinas son los computadores donde se corrieron cada una de las etapas del algoritmo, y las operaciones representan las fases de cada algoritmo (Ver figura 58).

Este problema aparece de forma constante en varios campos de la investigación dado que cualquier implementación de un proceso que tiene recursos limitados necesita optimizar su ejecución para aprovechar al máximo estos recursos. Bajo esta perspectiva el campo de aplicación de este problema es muy amplio, por ejemplo:

En el trabajo de Hauser y Manner [45] se implementó un algoritmo genético estándar en una serie de multiprocesadores asignados a un servidor. Este caso se puede caracterizar como un problema de calendarización dado que las fases del algoritmo genético se pueden ver como los trabajos, las máquinas como los recursos computacionales disponibles y las operaciones como el orden de realización de los trabajos.

[7] Instancias de Lawrence: Son datos para la prueba del problema de JSP, usados ampliamente en la literatura especializada.

[8] Contendiente: Prueba en la que se comparó dos algoritmos. El primero un algoritmo llamado Clonalg y el segundo un algoritmo denominado GRASP.

6.1 PLANTEAMIENTO DEL PROBLEMA.

Con base en el análisis de sensibilidad realizado en la sección 5.3.2 se pudo observar que conforme algunos parámetros varían se puede conseguir una mejora más significativa en la optimización ya sea utilizando la implementación inmune o la estocástica. En contraste, esta mejora acarrea un aumento en el tiempo de cómputo del algoritmo.

Además, en la literatura consultada [5] se observa que los algoritmos propuestos son probados utilizando un número elevado de iteraciones respecto al tomado en el presente estudio.

6.2 SOLUCIÓN PROPUESTA.

Fue necesario realizar pruebas con valores superiores en las iteraciones para los parámetros seleccionados con el fin de obtener valores más cercanos al BKS, lo cual conduciría al mejoramiento en el makespan, siempre y cuando se realicen modificaciones en el los algoritmos que permitan explorar otros espacios de búsqueda. Respecto al problema en el aumento del tiempo de cómputo se propone realizar dichas pruebas en un clúster de altas prestaciones a fin de reducir el tiempo procesamiento.

A continuación se describe un escenario donde se propone un modelo de calendarización tipo flow shop que asigna cada una de las fases del algoritmo de Clonalg como operaciones al respectivo nodo del clúster.

6.3 ESCENARIO

En el escenario propuesto se tiene una instancia:

- Asignación de tareas para cada una de las fases del algoritmo de Clonalg.

Máquinas: Los nodos del Clúster (4).

Trabajos: Cada una de las 40 instancias de Lawrence (La01 – La40)

Operaciones: Cada una de las fases del respectivo algoritmo.

- Fase1: Generar Población
- Fase 2: Evaluar Población
- Fase 3: Clonación e Hipermutación
- Fase 4: Evaluar Clones

Tiempo de procesamiento: Corresponde al tiempo de procesamiento de cada de las fases del algoritmo de Clonalg en cada una de los 4 nodos del clúster; éste esta medido en milisegundos.

Tamaño de la instancia: 40 x 4, es decir 40 trabajos (La01 - La40) y 4 Nodos o estaciones de trabajo (Por cada una de las fases del algoritmo, un nodo). (Ver figura 56)

Figura 58: Instancia de Clonal

Número de trabajos

40	4 → Número de máquinas						
0	20	1	3	2	90	3	11 → J0 (La01)
0	19	1	2	2	86	3	7
0	19	1	2	2	79	3	8
0	20	1	2	2	93	3	8 → **Tiempo de procesamiento** del trabajo **J3** en la máquina **M3** es decir, la instancia **La04** demora **8 milisegundos** en la máquina 3 la cual lleva acabo **la fase de evaluación de los clones**
0	21	1	3	2	75	3	7
0	25	1	13	2	105	3	14
0	30	1	2	2	102	3	14
0	22	1	7	2	103	3	10
0	17	1	14	2	107	3	11
0	21	1	13	2	110	3	9
0	33	1	13	2	134	3	11
0	21	1	13	2	133	3	12
0	22	1	14	2	139	3	11
0	24	1	13	2	165	3	18
0	49	1	13	2	139	3	12
0	26	1	2	2	144	3	15
0	23	1	14	2	141	3	12
0	23	1	2	2	150	3	13
0	34	1	2	2	147	3	13
0	23	1	13	2	135	3	16
0	43	1	3	2	205	3	11
0	54	1	2	2	188	3	11
0	64	1	3	2	185	3	12
0	41	1	3	2	204	3	12
0	57	1	3	2	187	3	12
0	47	1	2	2	248	3	15
0	65	1	3	2	247	3	16
0	2	1	3	2	247	3	15
0	61	1	3	2	257	3	15
0	48	1	3	2	244	3	15
0	53	1	2	2	358	3	18
0	82	1	2	2	370	3	16
0	58	1	2	2	356	3	17
0	64	1	2	2	365	3	26
0	92	1	4	2	372	3	21
0	59	1	6	2	282	3	16
0	65	1	2	2	281	3	17
0	62	1	2	2	269	3	17
0	57	1	3	2	280	3	23
0	60	1	2	2	273	3	17 → J39 (La40)

M0 M1 M2 M3

6.4 RESULTADOS

Como se pudo ver en la figura 58 el tamaño de la instancia es bastante grande. Por lo que mostrar la calendarización en diagramas de Gantt se hace una tarea complicada y sin ningún resultado.

Por esto se decidió llevar acabo una comparación en tablas de los resultados luego de ejecutar ambos algoritmos en la instancia 20 veces, y así comparar no sólo los valores obtenidos del makespan (Ver tabla 19)(Ver figura 59) sino analizar la dispersión de los resultados(Ver tabla 20).

Tabla 19: Comparación de resultados entre Clonalg y GRASP

Ejecuciones	Clonalg	GRASP
0	7811	8048
1	7821	7983
2	7842	8091
3	7811	7918
4	7832	8069
5	7834	8029
6	7812	7958
7	7812	8110
8	7811	8098
9	7824	7903
10	7813	8035
11	7825	7946
12	7828	7964
13	7824	8004
14	7811	7958
15	7817	8019
16	7808	7963
17	7811	7983
18	7829	8131
19	7811	8042

Figura 59: Gráfica de resultados de Clonalg y GRASP en la instancia clonal

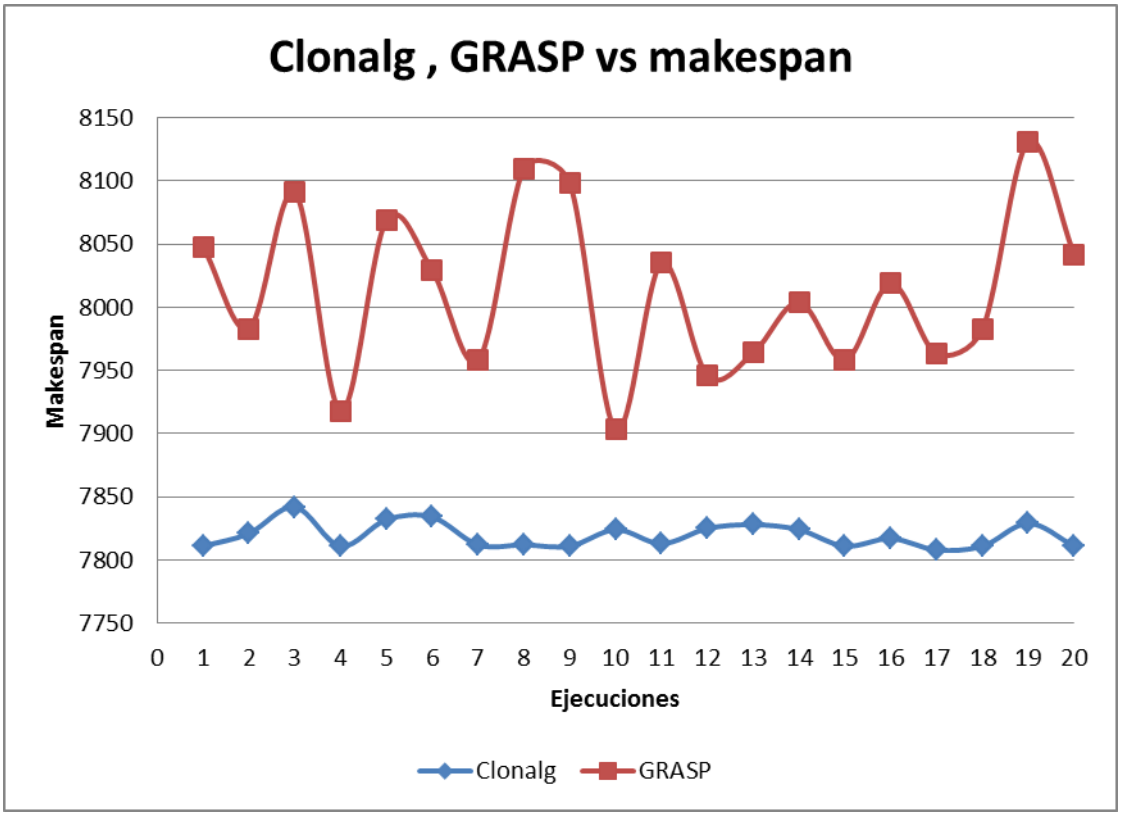


Tabla 20: Comparación estadística.

	Clonalg	GRASP
$\bar{x}$	7819,35	8012,6
S^2	91,73	4053,14
S	9,58	63,66

6.5 ANÁLISIS

En la tabla 18 se realizó un paralelo entre ambos algoritmos y sus resultados para cada una de las 20 ejecuciones realizadas en cada una de las 40 instancias seleccionadas. A continuación se graficaron dichos resultados (Ver figura 59), con los cuales se pudo concluir que el algoritmo de Clonalg genera resultado homogéneos (Ver desviación en tabla 20) y mejores en las instancias que los presentados por el algoritmo de GRASP.

Dado que el makespan inicial presentado en la instancia fue de 9386 milisegundos y el algoritmo Clonalg alcanzó un promedio en el makespan de 7819 y en GRASP 8012 milisegundos (Ver tabla 20), se observó que Clonalg tiene una mejora del 16,70% en comparación con GRASP que obtuvo un 14,14%. Esto puede verse también en la tabla 19 y la gráfica 59.

Así mismo se observó que Clonal optimiza la instancia disminuyendo es tiempo en 1566 milisegundo es decir 1,57 segundos.

Esto cobra relevancia cuando se lleva a proporciones más grandes, dado que nuestro análisis solo se efectuó para realizar 1 generación. Al analizar el caso de las tablas 8, 10, 12 y 14 donde se realizaron 100 generaciones de cada instancia y adicional a esto que se ejecuto 20 veces el algoritmo de Clonalg, se logra una disminución de 52 minutos en total lo que es un avance significativo.

7 CONCLUSIONES

- De las pruebas realizadas al algoritmo Inmune Artificial, Clonalg permite resolver Job Shop Scheduling y ofrece mejores resultados en cuanto a la minimización del makespan; comparado con el algoritmo estocástico GRASP superándolo en 39 de las 40 instancias.

- Se evidencia que el comportamiento del algoritmo Clonalg presenta una menor dispersión que el comportamiento del algoritmo GRASP. Esto se evidencia con la desviación estándar que en el algoritmo Clonalg fue 11,81% y en GRASP fue 20,01%

- El algoritmo Clonalg igualó la mejor solución conocida en el 25% de las instancias de Lawrence, mientras que el algoritmo GRASP lo logró para el 2,5%.

- Se evidenció que el error relativo medio crece conforme se incrementa el tamaño de las instancias.

- En el escenario computacional propuesto el algoritmo inmune superó con un porcentaje de mejoramiento del makespan del 16,7% frente al 14,14% del algoritmo GRASP.

8 RECOMENDACIONES

- Dado que el costo computacional aumenta en la medida en la que parámetros como: el valor de la población, el factor de clonación y el número de generaciones aumentan(todos estos para el algoritmo Clonal), y aumenta en el número de iteraciones y la cantidad de no mejoramientos (para el algoritmo GRASP), se sugiere realizar las pruebas con este tipo de parámetros en una infraestructura tipo clúster, ya que este tipo de tecnología permite reducir el tiempo de cómputo, que la provista por un solo computador .

- La implementación presente responde a un problema mono objetivo en el que se optimiza el makespan, se sugiere que a partir de esta aproximación preliminar al problema de Job Shop Scheduling realizar implementaciones que optimicen casos multi-objetivo en los que se tenga en cuenta el tiempo consumido por todos los trabajos (total flow time), la suma de todos los retrasos de los trabajos (Total latenees) entre otras medidas de desempeño.

- Con el fin de alcanzar valores más cercanos al BKS se recomienda para el caso del algoritmo Clonalg realizar una construcción de los individuos a partir de librerías de anticuerpos, dado que esto permite arrancar un individuo para el que su costo inicial este cercano a la mejor solución conocida, facilitando así que el algoritmo pueda mejorar mucho más rápido el problema.

9 BIBLIOGRAFÍA

[1] T. J. Kindt, R. A. Goldsby y B. A. Osborne, Inmunología de Kuby, Sexta ed., Mc Graw Hill.

[2] P. Guermonprez, J. Valladeau, L. Zitvogel, C. Théry y S. Amigonera, «Annu Rev Immunol,» pp. 621,67.

[3] «Glosario.net,» 09 Noviembre 2006. [En línea]. Available: http://lengua-y-literatura.glosario.net/terminos-filosoficos/heur%EDstica-5819.html. [Último acceso: 2012 Octubre 2012].

[4] C. Janeway y V. Staff, Inmunobiología: El sistema inmunitario en condiciones de salud y enfermedad, España: Elsevier, 2003.

[5] C. A. C. a. N. C. Cortés, Solving Multiobjective Optimization Problems Using an Artificial Immune System, Genetic Programming and Evolvable Machines 6, 2 (June 2005), 163-190., 2005.

[6] B. Melián, J. Moreno y M. Moreno, «Metaheuristics: A global view,» *Revista Iberoamericana de Inteligencia Artificial,* pp. 7-28, 2003.

[7] B. Roberts, Manual Merck, 2 ed., Oceano, 2008, p. 837.

[8] J. Miller y M. L., «Apostosis Science,» de *Apostosis Science*, 1998, p. 281.

[9] P. Brucker, «The job-shop problem: old and new challenges,» de *Proceedings of the MISTA Conference*, 2007.

[10] E. Hart y J. Timmis, «Applications areas of AIS: The Past, The Present an the Future,» *Applied Soft Computing,* vol. 8, nº 1, pp. 191-201, 2008.

[11] C. B. a. A. Roli, Metaheuristics in combinatorial optimization: Overview and conceptual comparison, ACM Comput. Surv. 35, 3 (September 2003), 268-308, 2003.

[12] D. D. F. G. U. Aickelin, «Artificial Immune Systems,» de *Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques*, 2 ed., Springer, 2010.

[13] M. D. S. F. A. P. Dennis L. Chao, «Stochastic stage-structured,» de *Bioinformatics Conference*, 2003.

[14] S. Y. a. N. S. M. D. Dasgupta, «MILA - Multilevel Immune Learning Algorithm,» de *Genetic and Evolutionary Computation Conference (GECCO)*, Chicago, 2003.

[15] R. R. F. S. a. P. A. S. Hightower, «The evolution of emergent organization in immune system gene libraries,» de *6th. Conference on Genetic Algorithms*, 1995.

[16] N. Jerne, «Towards a network of the immune system.,» *Ann Immunol,* pp. 73-89, 1974.

[17] M. Galiani, M. Santamaría y J. Peña, «Inmunología en línea,» [En línea]. Available: http://www.inmunologiaenlinea.es/index.php?option=com_content&view=article&id=97:tolerancia-y-regulacion-sistema-inmune&catid=47:tolerancia&Itemid=126. [Último acceso: 20 Octubre 2012].

[18] I. Karonen, «Wikimedia Commons,» 16 Agosto 2006. [En línea]. Available: http://commons.wikimedia.org/w/index.php?title=File:Clonal_selection.svg&page=1. [Último acceso: 20 Octubre 2012].

[19] D. Dasgupta, Artificial Immune Systems: A Bibliography, MEMPHIS: COMPUTER SCIENCE DEPARTMENT THE UNIVERSITY OF MEMPHIS, 2007.

[20] F. J.D, P. N. y P. A., The immune system, adaptation and machine learning, vol. 2, 1986, pp. 187-204.

[21] B. H y V. F.J, «Hints for adaptive problem solving gleaned from immune networks. Parallel Problem Solving from Nature,» de *First Workshop PPSW 1*, Dortmund, 1990.

[22] I. Y, «Fully distributed diagnosis by PDP learning algorithm: towards immune network PDP model,» de *IEEE International Joint Conference on Neural Networks*, San Diego, USA, 1990.

[23] F. S, P. A.S, A. L y C. R, «Self–nonself discrimination in a computer,» de *IEEE Symposiumon Research in Security and Privacy*, Los Alamitos, 1994.

[24] J. E. Hunt y D. E. Cooke, «An Adaptative, Distributed Learning System, based on the Immune System,» de *he IEEE International Conference on Systems Man and Cybernetics*, 1995.

[25] T. J, N. M y H. J, «An artificial immune system for data analysis,» de *Biosystems 55*, 2000, pp. 143-150.

[26] C. L.N.D. y Z. F.J.V., «The clonal selection algorithm with engineering,» de *Genetic and Evolutionary Computation Conference (GECCO'00) - Workshop Proceedings*, Las Vegas,

Nevada, USA, 2000.

[27] d. C. Leandro N. y Z. Fernando J. Von, «Learning and Optimization Using the Clonal Selection Principle,» de *IEEE Transactions on Evolutionary Computation, Special Issue on Artificial Immune Systems*, vol. 6, 2002, pp. 239-251.

[28] D. Dasgupta y L. F. Niño, Immunological Computation: Theory and Applications, Aurbach Publications, 2008.

[29] A.-S. Khaled, A.-K. H. y I. Nabil, «Artificial Immune Clonal Selection Algorithms: A Comparative Study of CLONALG, opt-IA, and BCA with Numerical Optimization Problems,» de *IJCSNS International Journal of Computer Science and Network Security*, 2010.

[30] «Artificial Immune Systems Web,» [En línea]. Available: http://www.artificial-immune-systems.org/algorithms.shtml. [Último acceso: 23 Mayo 2012].

[31] J. Brownlee, Clever Algorithms. Nature-Inspired Programming Recipes, Primera ed., 2011, pp. 1-436.

[32] K. . M. DEEPAK , JOB SHOP SCHEDULING USING ARTIFICIAL IMMUNE SYSTEM, ROURKELA: DEPARTMENT OF MECHANICAL ENGINEERING NATIONAL INSTITUTE OF TECHNOLOGY, 2012.

[33] M. F. Tupia Anticona, Un algoritmo GRASP para resolver el problema de la programación de tareas dependientes en máquinas diferentes (task scheduling), Lima - Perú: UNIVERSIDAD NACIONAL MAYOR DE SAN MARCOS , 2005.

[34] B. Roy y B. Sussmann, «Les problemes d'ordonnancement avec constraints disjonctives,,» SEMA, Paris, France, 1964.

[35] M. Resende y C. Ribeiro, «Greedy randomized adaptive search procedures,» de *State of the Art Handbook in Metaheuristics*, 2002.

[36] D. Applegate y W. Cook, «A computational study of the job-shop scheduling instance,» *ORSA Journal on Computing,* nº 3, pp. 149-156, 1991.

[37] J. E. Beasley, «OR-Library: Distributing Test Problems by Electronic Mail,» *Journal of the Operations Research Society,* pp. 1069-1072, 1990.

[38] S. Binato, W. J. Hery, D. M. Loewenstern y M. G. C. Resende, «A GRASP for Job Shop Scheduling.,» *Essays and Surveys in Metaheuristics,* pp. 59-80, 2001.

[39] S. R. Lawrence, Resource-Constrained Project Scheduling: An Experimental Investigation of Heuristic Scheduling Techniques, Pittsburgh PA: Graduate School of Industrial Administration, Carnegie-Mellon University, 1984.

[40] J. E. Beasley, «OR - Library: Distributing Test Problems by Electronic Mail,» *Journal of Operations Research Society,* nº 41, pp. 1069-1072, 1990.

[41] J. E. Beasley, «Brunel University West London,» 7 Septiembre 2004. [En línea]. Available: http://people.brunel.ac.uk/~mastjjb/jeb/orlib/jobshopinfo.html. [Último acceso: 11 Octubre 2012].

[42] A. Streit, «On Job Scheduling for HPC-Clusters and the dynP Scheduler,» de *HiPC '01 Proceedings of the 8th International Conference on High Performance Computing*, Springer-Verlag London, 2001.

[43] E. Frachtemberg y U. Schwiegelshohm, «Job scheduling strategies for parallel processing,» de *Proceedings of The 13th international conference*, Springer-Verlag Berlin, Heidelberg, 2008.

[44] M. A. Cruz Chavez, «Gridification of Genetic Algorithm with Reduced Communication for the Job Shop Scheduling Problem.,» de *International Journal of Grid and Distributed Computing*, 2010.

[45] R. Hauser y R. Manner, «Implementation of Standard Genetic Algorithm on MIMD machines,» de *Parallel Processing Symposium, 1995. Proceedings., 9th International*, Lehrstuhl fur Inf. V, Mannheim Univ., 1995.

[46] M. a. A.-F. M. Ben-Daya, «A tabu search approach for the flow shop scheduling problem,» de *European Journal of Operational Research*, vol. 109, 1998, pp. 88-95.

[47] J. E. Biegal y J. J. Davem, «Genetic algorithms and job shop scheduling,» de *Computers and Industrial Engineering*, vol. 19, 1990, pp. 81-91.

[48] V. A. y. Z. L. Peña, Cadena de Suministros: sus niveles e importancia. Modelado de Procesos de Negocios., 2006.

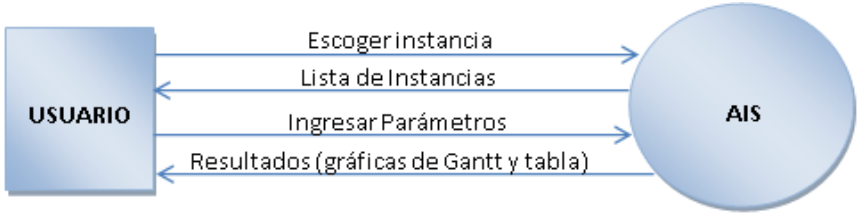
[49] M. Á. González Fernández, Soluciones Metaheurísticas al "Job-Shop Scheduling Problem with Sequence-Dependent Setup Times", Oviedo, España: Universidad de Oviedo, 2011.

[50] Daniel Cortés Rivera and Carlos A. Coello Coello. Uso de un Sistema Inmune Artificial para Problemas de Calendarización. In F. J. Martínez D. Ortiz y S. Ventura (editores) C.

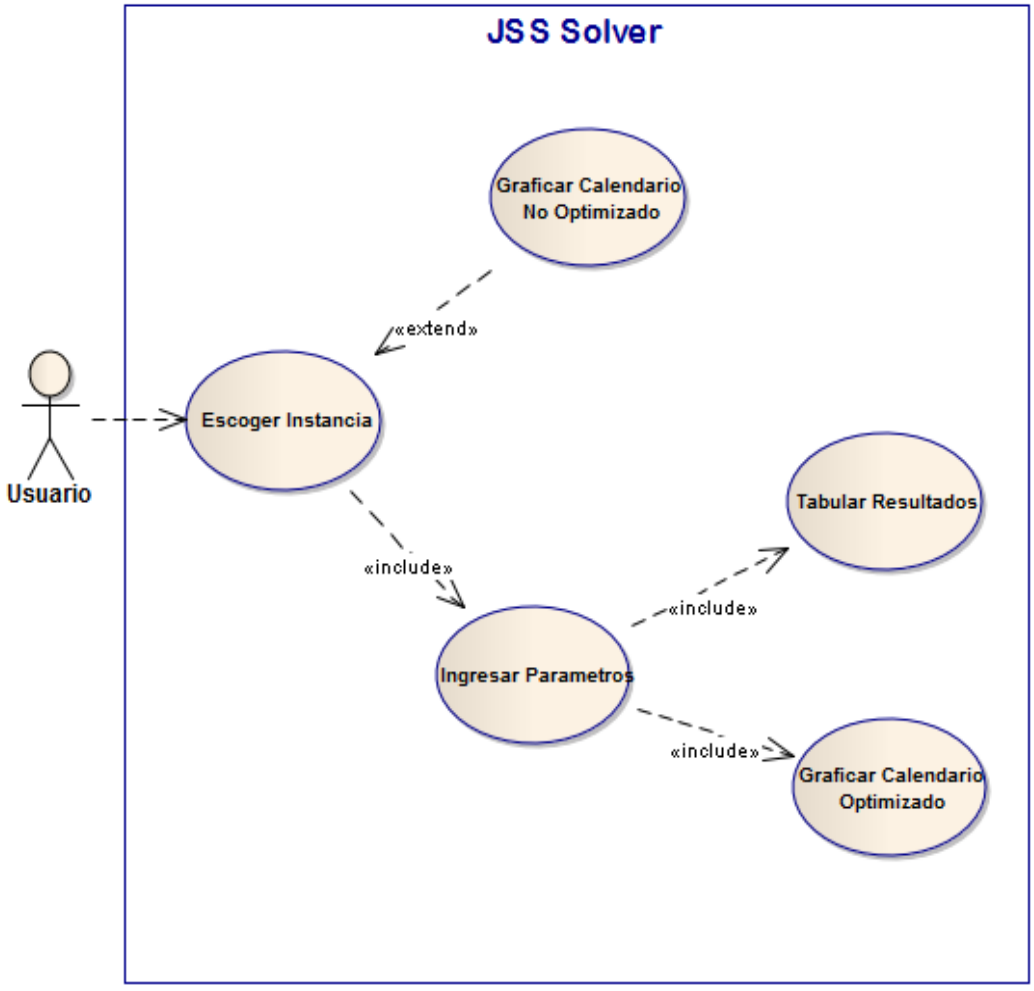
Hervás, N. García, editor, *Actas del III Congreso Español sobre Metaheurísticas, Algoritmos Evolutivos y Bioinspirados (MAEB 04)*, pages 507 { 514, Córdoba, España, Febrero 2004. Universidad de Córdoba. ISBN 84-688-4224-9.

10 ANEXOS

Anexo A. DIAGRAMA DE CONTEXTO

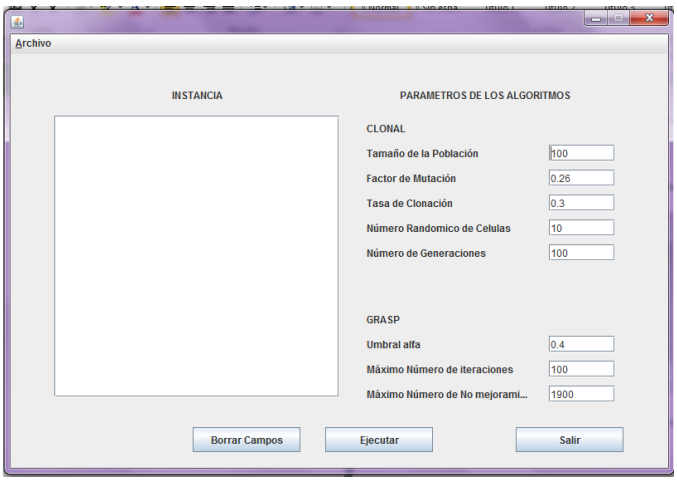


Anexo B. DIAGRAMA DE CASOS DE USO

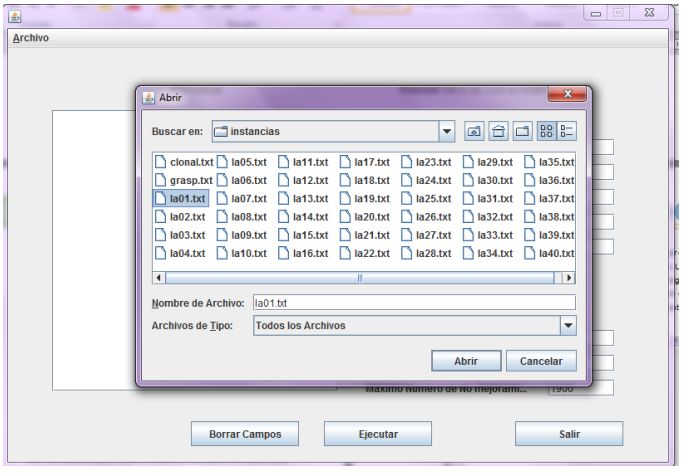


Anexo C. FUCIONAMIENTO DEL ALGORITMO

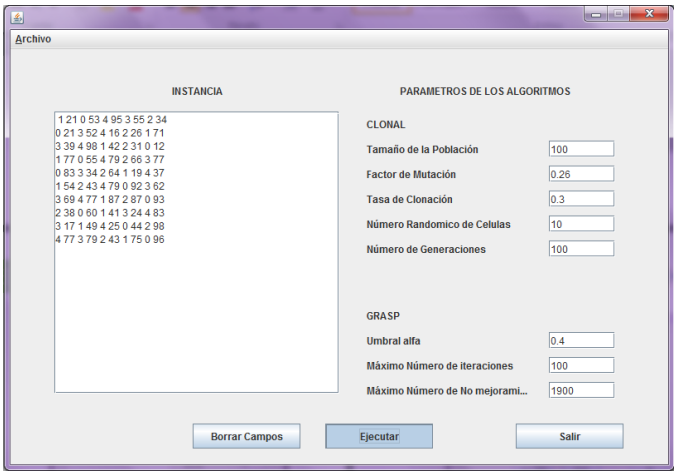
Ejecución: para iniciar el algoritmo basta con ubicarse en el directorio donde se encuentre la aplicación. Luego se carga el formulario donde se están los parámetros de los algoritmos con valores predeterminados.



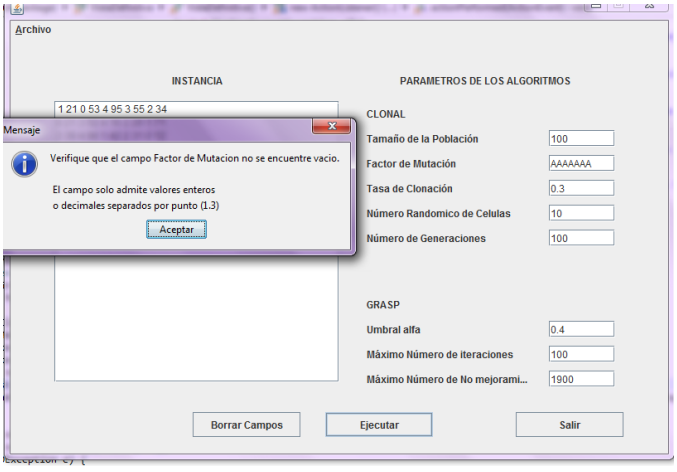
Cargar instancia: en la Barra de Menú el usuario selecciona la instancia que desea usar para trabajar el algoritmo



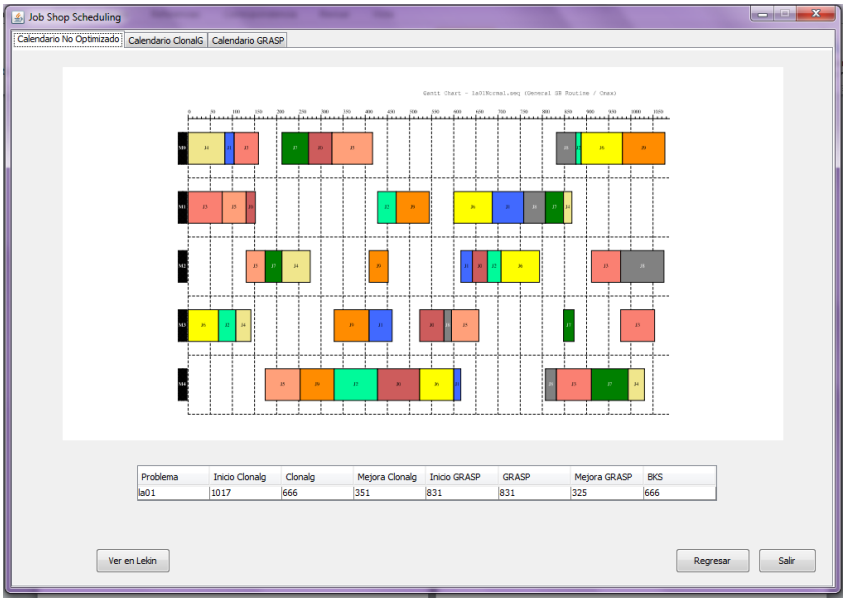
Una vez seleccionada la instancia, ésta será mostrada en el área de texto



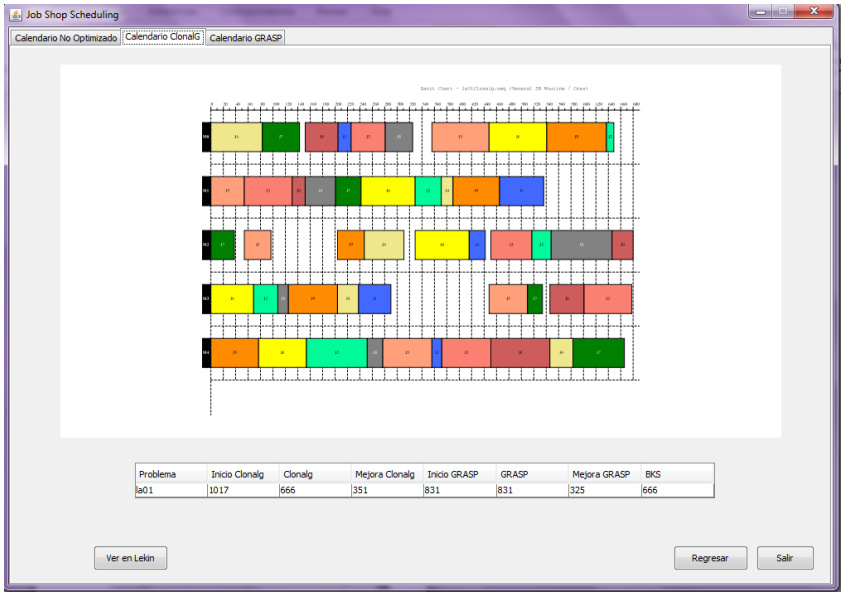
Validar parámetros y ejecutar: El usuario puede dejar los valores por defecto que se encuentran el los campos de texto o puede introducir otros. Una vez que se tengan los campos de los parámetros llenos y la instancia cargada se procede a dar clic en el botón ejecutar. Dicho botón entonces validará que los campos se encuentren correctos de lo contrario, mostrará una ventana de alerta.



Resultados: una vez se ejecute se mostrara el formulario de resultados con 3 pestañas. La primera mostrará el Calendario sin optimizar por medio de un diagrama de Gantt.



En la segunda pestaña, el calendario optimizado usando el algoritmo de selección Clonalg.



La tercera muestra el calendario optimizado utilizando el método de GRASP.

Cada una de estas pestañas contiene una tabla donde se muestra: el nombre de la instancia, el makespan con el que inicia el algoritmo de Clonalg, el makespan final de Clonalg, el mejoramiento (la diferencia entre el makespan inicial y el final), el makespan inicial del algoritmo de GRASP, el makespan final de GRASP, el mejoramiento (la diferencia entre le makespan inicial y el final), y el BKS (la mejor solución conocida en la literatura).