UNIVERSIDAD AGRARIA DE LA HABANA
"FRUCTUOSO RODRÍGUEZ PÉREZ"

FACULTAD DE CIENCIAS TÉCNICAS

OPTIMIZACIÓN DE MODELOS DE PROCESOS DE NEGOCIO UTILIZANDO ALGORITMOS MULTIOBJETIVOS

Tesis presentada en opción al título académico de Máster en Biomatemática

Autora: Ing. Yaimi Barcenas Mompeller

Tutor: Dr. C. Alexander Sánchez Díaz

Mayabeque, 2019

AGRADECIMIENTOS

A todas aquellas personas que han apoyado y contribuido a la realización de este trabajo, reciban los más sinceros agradecimientos, especialmente:

***A mis padres**, por todo su apoyo, sus consejos, su sacrificio incansable, por lograr que llegase a la meta, por ser los mejores padres del mundo, por toda su confianza y todo su amor. Los quiero mucho.*

***A mi abuela**, por apoyarme y confiar siempre en mí, gracias por ser como una segunda madre para mí. Te quiero.*

***A mi tío**, por complacerme todos mis caprichos, gracias por estar siempre cuando te necesito.*

***A mi esposo**, por formar parte de mi vida, por su apoyo incondicional, por quererme tanto, por confiar siempre en mí. Gracias por todos los momentos bellos que me has regalado, a tu lado he conocido el significado del verdadero amor. Te amo.*

***A mi hijo,** porque aunque estés chiquito cuando te pedía que me dejaras estudiar me hacías caso como un hombrecito, por ser lo más bello que puede tener una mujer, que es ser mamá. Gracias por los momentos bellos que paso cada día junto a ti. Te amo.*

***A mis suegros**, por ser tan buenos, quererme y confiar en mí. Gracias por quererme como una hija.*

***A mi cuñada, su esposo y sus dos niñas,** por compartir su cariño conmigo.*

***A mi tutor**, que siempre me alentó con seguir mi superación, gracias por tu apoyo incondicional y todos los consejos que me distes como tutor y amigo que hoy eres para mí.*

***Al claustro de profesores**, que me formó de manera profesional durante el transcurso de estos dos años de maestría y me transmitieron sus conocimientos, experiencias y valores.*

***A mis amistades**, principalmente a mis compañeros del Departamento por siempre estar preocupados sobre el progreso de esta investigación. En especial a mi compañera de trabajo Adanay por sus consejos y a mi jefe de departamento Yusney por siempre apoyarme.*

***A Raydel y Piedra,** por ser mis estudiantes ya graduados y amigos que me apoyaron y dieron su granito de arena en el desarrollo de esta investigación, gracias por su dedicación y apoyo incondicional.*

A todos... ¡Muchas Gracias!

-Yaimi-

A mi familia...

RESUMEN

En el mundo empresarial competitivo de hoy, las organizaciones y empresas necesitan manipular sus procesos de negocio. La verdadera clave para tener éxito en estas organizaciones radica en el diseño, la gestión adecuada de sus procesos de negocio y la alineación TI (Tecnologías de Información) con los objetivos de la organización. Al concentrarse en la optimización y mejora de los procesos de negocio, las empresas pueden lograr costos reducidos, mayor calidad de los productos, mayor eficiencia de los productos, adaptación a los cambios en los requisitos y de esa manera ascender en este entorno competitivo. La Gestión de Procesos de Negocio y la Minería de Procesos, son alternativas utilizadas como solución a la propuesta de la mejora de los procesos de negocio en cada una de las empresas que necesitan gestionar sus procesos de una manera eficiente. El objetivo de la presente investigación es optimizar modelos de procesos de negocio utilizando los algoritmos NSGAII y SPEA2 teniendo en cuenta el costo y la completitud. Como resultado se obtiene una aplicación informática que le permite a los analistas del negocio de las organizaciones, la obtención de posibles mejoras del proceso en cuanto a los criterios analizados y así de esa manera lograr un mejor desempeño de estos en las organizaciones.

Palabras claves: procesos de negocio, gestión de procesos de negocio, minería de procesos, optimización

ABSTRACT

Today, in the competitive business world, organizations and companies need to manipulate their business processes. The real key to success in these organizations lies in the design, the proper management of the business processes and the alignment of IT (Information Technology) with the objectives of the organization. By focusing on the optimization and improvement of business processes, companies can achieve reduced costs, higher product quality, greater efficiency of products, requirements adaptation to changes and thus rise in this competitive environment. Business Process Management and Process Mining are alternatives used as a solution to the proposal of improving business processes in each of the companies that need to manage their processes in an efficient way. The objective of this research is to optimize business process models using the NSGAII and SPEA2 algorithms taking into account the cost and completeness. As a result, an app is obtained that allows the organization business analysts to again possible improvements of the process in terms of the analyzed criteria and thus in that way achieving a better performance of these in the organizations.

Keywords: business processes, business process management, process mining, optimization

.

ÍNDICE

INTRODUCCIÓN 1

CAPÍTULO 1. FUNDAMENTACIÓN TEÓRICA 5

1.1 GESTIÓN DE PROCESOS DE NEGOCIO Y MINERÍA DE PROCESOS 5
1.2 MODELADO DE PROCESOS DE NEGOCIO 8
1.2.1. BPMN (Notación de Modelado de Procesos de Negocio) 8
1.2.2. Redes de Petri 10
1.2.2.1 Formato PNML 13
1.2.2.2. Formato XES 15
1.3. OPTIMIZACIÓN MULTIOBJETIVO 17
1.3.1 Optimalidad de Pareto 18
1.4. ALGORITMOS EVOLUTIVOS EN LA OPTIMIZACIÓN MULTIOBJETIVO 21
1.4.1. Algoritmos Genéticos 22
1.4.1.1. Algoritmo NSGAII 24
1.4.1.2. Algoritmo SPEA2 28
1.5. CONCLUSIONES 28

CAPÍTULO 2. OPTIMIZACIÓN MULTIOBJETIVO DE PROCESOS DE NEGOCIO 29

2.1. MÉTODOS DE INVESTIGACIÓN 29
2.2. TECNOLOGÍAS Y HERRAMIENTAS UTILIZADAS PARA LA IMPLEMENTACIÓN 30
2.2.1. Lenguaje de programación Java 30
2.2.2. Entorno de desarrollo Netbeans 31
2.2.3. Framework JMetal 31
2.2.4. Bonita Studio 32
2.2.5. Yasper 32
2.3. FORMULACIÓN DEL PROBLEMA MATEMÁTICO 33
2.3.1. Función Objetivo: costo del proceso 33

2.3.2. Función Objetivo: completitud del proceso 33
2.4. CODIFICACIÓN PARA REPRESENTAR EL CONJUNTO DE SOLUCIONES 35
2.5. PASOS A REALIZAR PARA LA OPTIMIZACIÓN DE LOS PROCESOS DE NEGOCIO 37
2.5.1. Modelación del Proceso de Negocio 38
2.5.2. Aplicación de los algoritmos NSGAII o SPEA2 42
2.5.2.1. Generación de la población inicial 42
2.5.2.1.1. Transformación de Matrices Causales en BP-net 44
2.5.2.2. Evaluación de cada individuo 46
2.5.2.3. Operadores genéticos 47
2.5.2.3.1. Operador Genético: Selección 47
2.5.2.3.2. Operador Genético: Cruzamiento 49
2.5.2.3.3. Operador Genético: Mutación 50
2.6. CONCLUSIONES 51

CAPÍTULO 3. RESULTADOS Y DISCUSIÓN 52

3.1. APLICACIÓN INFORMÁTICA IMPLEMENTADA 52
3.2. Resultados experimentales 57
3.3. COMPARACIÓN DE LOS ALGORITMOS NSGAII Y SPEA2 63
3.3.1. Generación de vectores no dominados (GVND) 65
3.3.2. Razón de generación de vectores no dominados (RGVND) 65
3.3.3. Generación real de vectores no dominados (GRVND) 65
3.3.4. Distancia generacional (G) 66
3.3.5. Resultados de las métricas 66
3.4. CONCLUSIONES 68

CONCLUSIONES Y RECOMENDACIONES 69

REFERENCIAS BIBLIOGRÁFICAS 71

GLOSARIO DE TÉRMINOS 76

ANEXOS…. 78

ÍNDICE DE TABLAS

Tabla 1. Elementos de BPMN..... 9
Tabla 2. Elementos de una red de Petri..... 12
Tabla 3. Resultados obtenidos con el algoritmo NSGAII para 5 ejecuciones. 57
Tabla 4. Resultados obtenidos con el algoritmo SPEA2 para 5 ejecuciones. 60
Tabla 5. Conjunto de soluciones que conforma el conjunto $Ytrue$ 64
Tabla 6. Resultados de las métricas en las ejecuciones del algoritmo NSGAII. 66
Tabla 7. Resultados de las métricas en las ejecuciones del algoritmo SPEA2. 67
Tabla 8. Valor de las soluciones que dominan los algoritmos con respecto al otro. 68

ÍNDICE DE FIGURAS

Figura 1. Tipos básicos de la minería de procesos. 6
Figura 2. Ciclo de vida del BPM. 6
Figura 3. Etiquetas básicas de PNML. 14
Figura 4. Etiquetas de información gráfica de un archivo PNML.. 14
Figura 5. Modelo transaccional del estado de las actividades. 16
Figura 6. Diagrama esquemático del mecanismo de promoción de individuos del NSGAII. 26
Figura 7. Pasos para optimizar procesos de negocio. 29
Figura 8. Relación entre red de Petri y matriz causal. 37
Figura 9. Modelación del proceso: Producción de la Langosta Entera Precocinada en la notación BPMN. . 39
Figura 10. Modelación del proceso: Producción de la Langosta Entera Precocinada en una red de Petri. 40
Figura 11. Registro de eventos del proceso de negocio: Producción de la Langosta Entera Precocinada. 41
Figura 12. Matriz que representa el índice de dependencia para el proceso. 44
Figura 13. Mapeo en una red de Petri. 45
Figura 14. Mecanismo de disparo en el parseo de una traza. 46
Figura 15. Interfaz Principal de la herramienta Optimización de Procesos de Negocio. 53
Figura 16. Carga del proceso de negocio: Producción de la Langosta Entera Precocinada. 54
Figura 17. Configuración de los parámetros de entrada para el algoritmo NSGAII. 55
Figura 18. Configuración de los parámetros de entrada para el algoritmo SPEA2. 56
Figura 19. Resultado del proceso optimizado. 57
Figura 20. Diseño de clases utilizadas para el manejo de los algoritmos. 78

INTRODUCCIÓN

En el mundo moderno competitivo de los negocios son muchas las organizaciones que necesitan modificar los diseños de sus procesos de negocio, para volverse más competitivas en el mercado empresarial. Estas enfocan su principal atención en la optimización y la mejora continua de sus procesos. La automatización de las tareas en las organizaciones, es de gran importancia ya que esta es una de las disciplinas encargadas de descubrir, aportar y dar soluciones a diversos problemas que se presentan a diario en los procesos de negocio de las mismas.

La minería de procesos ha surgido como una disciplina que apoya las organizaciones en pos de enfrentar el desafío de gestionar mejor sus procesos. Esta permite realizar el descubrimiento de modelos, verificar la conformidad de los mismos con respecto a un modelo ideal y proponer mejoras para los procesos de negocio. Además, se enfoca en extraer conocimiento a partir de la información almacenada en los registros de eventos ubicados en los sistemas de información (Van der Aalst, et al., 2011).

Las organizaciones, en general, tratan día a día de revisar la manera de cómo están operando y analizan la forma de mejorar sus procesos. La Gestión de Procesos de Negocio (BPM[1]: *Business Process Management*, por sus siglas en inglés), fue creado para apoyar a las organizaciones que desean gestionar de una manera más eficiente sus procesos de negocio. Según Van der Aalst, et al. (2003), la administración de procesos de negocio incluye métodos, técnicas y herramientas creadas para apoyar el diseño, la mejora, la gestión y el análisis de los procesos de negocio operacionales. Es un reto de las organizaciones de hoy poder aprovechar la información histórica guardada en los registros de eventos de los sistemas, con el objetivo de realizar mejoras a sus procesos. El modelo de estos procesos de negocio puede estar representado mediante la notación BPMN[2] (*Business Proceses Management Notation*, por sus siglas en inglés) o una red de Petri, que en conjunto con su registro de eventos, es posible realizar una optimización a este mejorando su desempeño en las organizaciones que lo ejecutan.

[1] *https://www.bpm.com*
[2] *https://www.bpmn.org*

Las etapas que comprende el clico de vida de un proceso de negocio en BPM en conjunto con la disciplina de la minería de procesos se clasifican en diseño, modelación, ejecución, monitoreo y optimización. Esta última es una de las más significativas en el desempeño de un proceso, puesto que en ella es donde se toma la información de la etapa de modelación y los datos de desempeño de la etapa de monitoreo y se comparan, identificando así los cuellos de botellas en los procesos para determinar las alternativas o rutas más óptimas del proceso para su mejor comportamiento. Para la realización de la optimización se requiere de un conocimiento experto para que sea usada de una forma eficiente (Weske, 2010).

La optimización es el proceso de búsqueda de la mejor solución posible para un determinado problema. Puede verse como la búsqueda de los valores de las variables de decisión para los cuales cierta función objetivo (fo) logra su valor extremo (Chong y Zak, 2013). La optimización multiobjetivo aplicada a los procesos de negocio puede resultar una buena opción para mejorar estos ya que más de un criterio de optimización puede ser seleccionado y satisfecho simultáneamente.

Son muchos los criterios que pueden tenerse en cuenta para realizar la optimización y mejora de los procesos en las organizaciones. Vergidis, et al. (2007), consideran los criterios de tiempo y costo para llevar a cabo la optimización del diseño de los procesos de negocio. Mientras que Ahmadikatouli y Aboutalebi (2011), tienen en cuenta criterios como el tiempo, costo, calidad de productos y la longitud de la cola. Además, Medeiros (2006) propone que la completitud del proceso es significativa para lograr adecuados modelos de procesos teniendo en cuenta los registros de eventos de los procesos.

Analizando lo anterior se puede determinar que el costo es uno de los criterios más utilizados para mejorar el desempeño de las organizaciones. Por otra parte, al aplicar la minería de procesos a la gestión de los procesos de negocio se es necesario tomar en cuenta la representación (completitud) del proceso dado que esta permite que se puedan obtener los modelos de los procesos con una mayor representatividad de este en cuanto a su registro de eventos.

Han surgido diferentes técnicas de optimización como son: la programación no lineal, la búsqueda tabú y la optimización evolutiva. Diversos autores como Back (1996), Talavera (2005) y Puris (2009), plantean que los algoritmos evolutivos han demostrado ser eficientes en el manejo de los problemas multiobjetivo ya que estos pueden manejar cualquier criterio utilizando un método matemático, como por ejemplo las redes de

Petri. Dentro de los algoritmos evolutivos más utilizados en la optimización multiobjetivo se pueden mencionar los siguientes: el Algoritmo Genético de Clasificación No Dominada II (NSGAII, por sus siglas en inglés) propuesto por Deb, et al. (2002) y el Algoritmo Evolutivo Pareto de Fuerza 2 (SPEA2, por sus siglas en inglés) propuesto por Zitzler, et al. (2000).

En Cuba se han llevado a cabo investigaciones relacionadas con la aplicación de la minería de procesos como una alternativa útil para la gestión de los procesos de negocio. Algunas de estas investigaciones se han llevado a cabo en la Universidad de Ciencias Informáticas (UCI). Por otra parte, la Dirección de Informatización de la Universidad Agraria de La Habana (UNAH) junto con su grupo de investigación Proneg (Procesos de Negocio) tienen dentro de sus líneas de investigación el trabajo relacionado con la minería de procesos. En él se desarrollan algoritmos para el descubrimiento y optimización de modelos a partir de los registros de eventos.

Situación problemática

La Minería de Procesos es un área relativamente joven dentro de la Gestión de los Procesos de Negocio o Inteligencia Organizacional. Muchos autores como Alcover et al. (2007), plantean la necesidad de aportar con nuevas contribuciones; al análisis de las actividades que ocurren en los procesos de negocio que son gestionadas y monitoreadas por diferentes tipos de sistemas de información. Osuszek (2012), plantea la necesidad de incorporarle multiobjetividad a los procesos de negocio teniendo en cuenta varios objetivos a la misma vez. Por otra parte, Van der Aalst (2013) menciona que no es usual el uso de herramientas para el monitoreo y optimización de procesos. Giraldo (2016), sugiere que es necesario que se integren estas disciplinas y así poder contribuir con nuevas alternativas al óptimo desempeño y control de los procesos en las organizaciones.

Esto sustenta el interés de la investigación de cómo lograr una alineación entre la minería de procesos y la optimización multiobjetivo en la gestión de los procesos de negocio de las organizaciones de hoy. Así como la necesidad que existe actualmente en las empresas de disponer de herramientas informáticas que les permitan optimizar sus procesos de negocio teniendo en cuenta varios objetivos.

Por lo tanto, el **problema científico** de esta tesis se plantea mediante la siguiente interrogante: ¿Cómo optimizar modelos de procesos de negocio utilizando algoritmos multiobjetivos?

Para dar solución al mismo se propone la siguiente **hipótesis**: Mediante la utilización de los algoritmos NSGAII y SPEA2 se podrá optimizar modelos de procesos de negocio atendiendo al costo y completitud.

La investigación se realiza sobre el **objeto de estudio**: optimización multiobjetivo en la gestión de procesos de negocio y en la minería de procesos. Se enmarca como **campo de acción** los algoritmos multiobjetivos aplicados al modelado de los procesos de negocio.

Se establece como **objetivo general**: Optimizar modelos de procesos de negocio utilizando los algoritmos NSGAII y SPEA2 teniendo en cuenta el costo y la completitud.

Para la solución del problema científico y dar cumplimiento al objetivo general, se plantearon los siguientes **objetivos específicos**:

1. Sintetizar los referentes teóricos que sustentan la optimización multiobjetivo en la gestión de los procesos de negocio y en la minería de procesos.
2. Rediseñar los algoritmos NSGAII y el SPEA2 para la optimización multiobjetivo de los modelos de procesos de negocio en cuanto a costo y completitud.
3. Evaluar los algoritmos utilizando los mismos parámetros de entrada para un proceso determinado.

Entre **los aportes prácticos** esperados con este trabajo de tesis se encuentran: la obtención de una aplicación informática que permita optimizar modelos de procesos de negocio utilizando los algoritmos NSGAII y SPEA2 atendiendo al costo y completitud del mismo, ofreciéndole así a los analistas del negocio las soluciones más óptimas del proceso para su mejora continua.

CAPÍTULO 1. FUNDAMENTACIÓN TEÓRICA

En este capítulo se introducen los fundamentos teóricos relacionados con la minería de procesos y la gestión de procesos de negocio, destacando la etapa de optimización y mejora del proceso al aplicarse en la gestión de los mismos. Se detallan los modelos de representación de los procesos de negocio: BPMN y redes de Petri. Se describen los formatos PNML y XES para el empleo del modelo matemático redes de Petri. Seguidamente se introduce la optimización multiobjetivo y el criterio de optimalidad de Pareto a emplear en la investigación para el problema abordado. Además, se describen los algoritmos evolutivos multiobjetivos destacando así el funcionamiento de los algoritmos genéticos; y se explican detalladamente los algoritmos NSGAII y SPEA2 para llevar a cabo la optimización de los procesos de negocio.

1.1 Gestión de Procesos de Negocio y Minería de Procesos

La Gestión de Procesos de Negocio (BPM, por sus siglas en inglés), es una disciplina que permite a las organizaciones gestionar de una manera más eficiente sus procesos de negocio. El BPM incluye métodos, técnicas y herramientas creadas para apoyar el diseño, la mejora, la gestión y el análisis de los procesos de negocio (Van der Aalst, et al., 2003).

La minería de procesos es una disciplina que tiene como objetivo descubrir, monitorear y mejorar procesos a través de la extracción de conocimiento de los registros de eventos disponibles en los actuales sistemas de información (Van der Aalst, et al., 2011). Esta disciplina ha sido aplicada en varios ámbitos, por ejemplo, en el área de la salud (Rojas, et al., 2016) y en el área de educación (Trcka, et al., 2010), igualmente, se ha aprovechado en distintos tipos de sistemas, en sistemas legados de datos (Pérez, et al., 2011), en plataformas colaborativas como Sharepoint (Arias y Rojas, 2014) y en sistemas para asignación de recursos a actividades en procesos de negocio (Arias, et al., 2015).

Existen tres tipos fundamentales de técnicas de minería de procesos: el descubrimiento del modelo: toma un registro de eventos y produce un modelo; la verificación de conformidad: compara un modelo del proceso existente con un registro de eventos del mismo proceso y la técnica relacionada con el mejoramiento de los procesos de negocio: se extiende u optimiza un modelo del proceso existente utilizando la información sobre

el proceso real que reside en algún registro de eventos (Van der Aalst, et al., 2011). En la figura 1 se pueden observar los 3 tipos básicos de la minería de procesos explicados en términos de entradas y salidas.

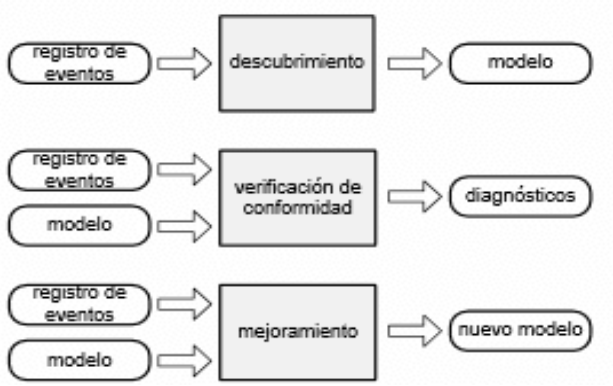


Figura 1. Tipos básicos de la minería de procesos. Fuente (Van der Aalst, et al., 2011).

El BPM y la minería de procesos permiten analizar e identificar las posibles mejoras a los procesos en las organizaciones, por medio del uso de un grupo de técnicas y herramientas. El BPM cubre todo el ciclo de vida de un proceso de negocio desde su diseño, modelamiento, ejecución, monitoreo y optimización, el cual se ve reflejado en la figura 2 propuesto por Weske (2010). Al igual, la minería de procesos se centra en el descubrimiento y el monitoreo de procesos reales, así como en proponer mejoras de acuerdo a los hallazgos encontrados (Van der Aalst, et al., 2011).

Figura 2. Ciclo de vida del BPM. Fuente (Weske, 2010).

Existen algunos trabajos relacionados con la minería de procesos como una alternativa para llevar a cabo la mejora de los procesos de negocio. Parka y Kangb (2016), realizaron un estudio que deriva un plan de mejoras del proceso y ofrece implicaciones académicas y prácticas a través del análisis de los datos municipales de *Netherlands*. Arias y Rojas (2016) proponen una serie de pasos que pueden servir como una guía útil para los responsables de gestionar los procesos de negocio en una organización apoyándose en la minería de procesos.

Igualmente existen algunas investigaciones relacionadas con la optimización de procesos de negocio de las cuales se pueden mencionar la de Zho y Chen (2003), donde propusieron que la optimización de los procesos de negocios conduce a una reducción en el tiempo de integridad del proceso y a los costos de funcionamiento, así como a un aumento de la calidad de los productos y la satisfacción del cliente. Con esta perspectiva, casi literalmente una organización puede adquirir la ventaja competitiva que estaba buscando. Hofacker y Vetschera (2001), han hecho algunos esfuerzos para optimizar un proceso de negocio con el empleo del algoritmo genético, pero como su método depende principalmente de varias fórmulas matemáticas y ha requerido muchas restricciones, apenas se produjeron soluciones factibles. Sin embargo, Tiwari et al. (2006) y Vergidis et al. (2006) expandieron este modelo matemático y propusieron un algoritmo de optimización multiobjetivo que se ha reportado como satisfactorio. Por otra parte, Ahmadikatouli y Aboutalebi (2011) proponen un nuevo enfoque multiobjetivo para llevar acabo la optimización de los procesos de negocio mediante el empleo del algoritmo genético.

La autora de la actual investigación se enfoca en la última etapa de las dos disciplinas referentes a la optimización y mejora continua de los procesos de negocio ya que estas ofrecen nuevas alternativas para el óptimo desempeño y control de los mismos que hoy se presentan a diario en las organizaciones. Como apoyo a estas disciplinas se han creado distintos estándares que incluyen notaciones simbólicas para la definición de los procesos de negocio. Dentro de estos, los más importantes y utilizados por la industria son *Business Process Management Notation* (BPMN) por la OMG (2010), *Yet Another Workflow Language* (YAWL[3]) por

[3] *https://yawlfoundation.org*

Ter Hofstede y Van der Aalst (2005), redes de Petri por Murata (1989) y *Workflow Language* por Fischer (2003).

1.2 Modelado de Procesos de Negocio

Los sistemas de gestión de procesos de negocio están guiados por modelos de procesos y modelos de organización. Se han propuesto diversas técnicas para modelar la perspectiva de procesos, en esta investigación se hace uso del BPMN y las redes de Petri.

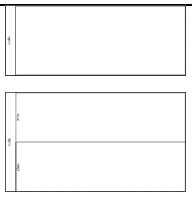
1.2.1. BPMN (Notación de Modelado de Procesos de Negocio)

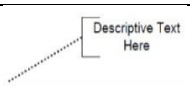
El Grupo de Gestión de Objetos (OMG[4]: *Object Management Group*, por sus siglas en inglés), ha desarrollado un modelo de procesos de negocio estándar y su notación BPMN. Este estándar tiene como principal objetivo proporcionar una notación que sea fácilmente comprensible por todos los usuarios del negocio, desde los analistas que crean los borradores iniciales de los procesos, a los desarrolladores técnicos responsables de la aplicación de la tecnología que va a llevar a cabo estos procesos, y por último, a los empresarios que gestionarán y controlarán dichos procesos. Por lo tanto, BPMN crea un puente estandarizado para la brecha entre el diseño de procesos de negocio y su implementación (OMG, 2010).

BPMN es una notación gráfica que describe la lógica detallada de los pasos de un proceso de negocio. Esta notación ha sido especialmente diseñada para coordinar la secuencia de los procesos y los mensajes que fluyen entre los participantes de las diferentes actividades. En la Tabla 1 se muestran los principales elementos del BPMN.

[4] *https://www.omg.org*

Tabla 1. Elementos de BPMN (Elaboración Propia).

Elemento	Descripción	Notación
Evento de Inicio	Indica el inicio de un proceso particular.	
Evento Intermedio	Ocurren entre un evento de inicio y uno de fin; afectan el flujo del proceso, pero no iniciarán o (directamente) terminarán el proceso.	
Evento de Fin	Indica el fin de un proceso particular.	
Actividad	Indica el trabajo que se ejecuta en el proceso. Puede ser atómica (tarea) o compuesta (sub-proceso).	
Tarea	Se utiliza cuando el trabajo no se puede romper en un nivel más fino de detalle.	
Subproceso	Indica que la actividad es un sub-proceso y por ende tiene un nivel inferior de detalle.	
Flujo de Secuencia	Un flujo de secuencia es utilizado para mostrar el orden de actividades que serán realizadas en el proceso.	
Flujo de Mensaje	Un Flujo de Mensaje es utilizado para mostrar la interacción entre dos participantes. En BPMN, dos Pools separados en el diagrama representará dos participantes.	
Asociaciones	Son utilizados para asociar información con Objetos de Datos. Los textos también pueden ser asociados con esta primitiva (constructo).	
Compuerta	Se usan para controlar la divergencia o convergencia de los flujos de secuencia en un proceso ya sean condicionadas o no. Para indicar lo que condiciona el comportamiento de la compuerta, éstas contienen diferentes figuras en su interior.	
Swimlane	**Pool:** Un Pool representa un participante en un Proceso. También actúa como un contenedor gráfico para separar un conjunto de actividades de otros Pools. **Lane:** Un Lane es una sub-división dentro de un Pool y son utilizados para organizar y categorizar actividades.	
Objetos de Datos	Son considerados artefactos porque no tienen un efecto directo en el flujo de la secuencia o de mensajes dentro del proceso, pero proveen información sobre las actividades que necesitan o producen éstos (entradas y salidas).	

Grupo	Una agrupación de actividades que no afectan la secuencia del flujo. También pueden ser utilizados para identificar las actividades de una transacción distribuida que debe ser mostrada a lo largo de los Pools.	
Anotaciones de textos	Es un mecanismo para modelar información adicional para los lectores del modelo.	

La tabla 1 muestra los elementos de la notación BPMN que son utilizados para la modelación del proceso que será realizado en los próximos capítulos mediante la herramienta Bonita Studio[5]. Posteriormente se procede a convertir el proceso BPMN hacia el modelo matemático redes de Petri, el cual se aborda a continuación.

1.2.2. Redes de Petri

Las redes de Petri son una herramienta muy adecuada para el modelado y análisis de los procesos de negocio de las organizaciones. Por una parte, se pueden utilizar como lenguaje de diseño para la especificación de complejos flujos de trabajo, y por otra, la teoría de redes de Petri proporciona una potente herramienta de análisis para verificar la corrección de los procedimientos de *workflow* (Van der Aalst, 1998). Las redes de Petri aparecieron por primera vez en 1962, en la tesis doctoral de Carl Adam Petri (Petri, 1962).

Las redes de Petri ofrecen una semántica matemática formal y notación gráfica para representar modelos de procesos. Gráficamente se visualizan mediante la unión de círculos denominados lugares y rectángulos conocidos como transiciones, relacionados por arcos dirigidos que determinan un grafo bipartido. De este modo, nunca dos lugares se relacionan mediante un arco, al igual que ocurre con las transiciones. A continuación, se ofrece la definición formal de red de Petri tomada de Medeiros (2006).

Una **red de Petri** es una terna (P, T, F) donde: P es un conjunto finito de lugares, T es un conjunto finito de transiciones tal que $P \cap T = \emptyset$, $F \subseteq (P \times T) \cup (T \times P)$ es un conjunto de arcos dirigidos. Para cada $\chi \in (P \cup T)$, $\bullet \chi = \{y | y, \chi) \in F\}$ denota al conjunto de nodos de entrada y $\chi\bullet = \{y | \chi, y) \in F\}$ denota al conjunto de nodos de salida.

[5] *https://es.bonitasoft.com*

A cada lugar se le pueden asociar marcas, que son empleadas para guiar y controlar ejecuciones de trazas sobre el modelo. Cada distribución de marcas sobre los lugares, determina un estado de la red de Petri conocido como marcaje M, en términos matemáticos $M: P \rightarrow N$. Una transición $t \in F$ se dice habilitada en un marcaje M, si cada uno de sus lugares de entrada $\bullet t$ contiene al menos una marca.

Dado un marcaje M sobre una red de Petri R, si una transición t se encuentra habilitada, entonces puede ser disparada, dando lugar a un nuevo marcaje M^* denotado por $M[t] \rightarrow M^*$. El disparo de t consume una marca en cada uno de sus lugares de entrada $\bullet t$ y produce una nueva marca en cada uno de sus lugares de salida $t\bullet$.

De este modo, un marcaje M^* es alcanzable a partir de M, si existe una secuencia de transiciones disparadas: $\sigma = t_1, t_2 \ldots, t_n$, que transforman M en M^* , denotado por $M[\sigma] \rightarrow M^*$. En cuyo caso, existe una secuencia de marcajes $M_0, M_1 \ldots, M_n$, tal que, para $M = M_0$ y $M_n = M^*$, se cumple que $M_i[t_i + 1] \rightarrow M_i + 1$, para $0 \leq i < n$.

Según Van der Aalst (2002b), las redes de procesos de negocio: BP-net (*Business Process Net*, por sus siglas en inglés), son una subclase de redes de Petri empleadas para modelar procesos y que permite verificar la correctitud del modelo.

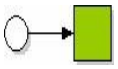
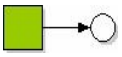
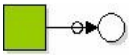
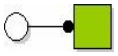
Este tipo de red de Petri son usadas en Murata (1989), Reisig y Rozenberg (1998), Van der Aalst (2002a) y Medeiros et al. (2007) y por ende en esta investigación como modelo matemático para la representación de los procesos de negocio. Medeiros et al. (2007), definen las BP-net de la manera siguiente:

Una **red de Petri** $R = (P; T; F)$ es una **BP-net** si y solo si: existe un lugar de origen s tal que $\bullet s = \emptyset$, existe un lugar final d tal que $d\bullet = \emptyset$. Cada nodo $x \in (P \cup T)$ se encuentra en un camino de s a d.

El modelado de procesos con BP-net, donde cada transición identifica a una actividad, permite la ejecución de trazas para ver si se encuentran correctamente representadas. Para ejecutar una traza $\sigma = \{a_1, a_2, \ldots, a_n\}$, se parte de un estado inicial M_s en la red con una única marca en el lugar de origen s. Se tiene que cumplir que $M_s[\sigma] \rightarrow M_d$, donde M_d es el estado en que solamente existe una marca ubicada en el lugar final d. En caso contrario, se considera que la traza no está representada (Medeiros, 2006).

La tabla 2 muestra los elementos que conforman una red de Petri con la descripción y representación de cada uno.

Tabla 2. Elementos de una red de Petri (Elaboración propia).

Elemento	Descripción	Notación
Transición	Una transición es un modelado de unidad que representa un paso del proceso.	
Lugar	Un lugar puede contener símbolos (tokens) que definen una parte del estado de la red. Una distribución de tokens en los lugares es llamado un marcado, representando el estado de la red.	Lugar  Lugar con marcado 
Arco de entrada	Un arco de entrada es la flecha (entrante) desde un lugar a una transición. Si todos los lugares conectados a una transición tienen suficientes fichas según las reglas de disparo, la transición está habilitado y puede disparar.	
Arco de salida	Un arco de salida es la flecha de una transición a un lugar (saliente). Si una transición de salida, consume tokens desde sus lugares, entrantes y salientes produce tokens en sus lugares de acuerdo a las normas de salida.	
Reset Arc	Cuando un lugar está conectado a una transición con un reset arc, todos los tokens se extraen de ella cuando la transición se dispara.	
Arco inhibidor	Cuando un lugar conectado a una transición a través de un arco inhibidor contiene un token, la transición no puede ser activada.	

Para interpretar el modelo de las redes de Petri y así llevar a cabo una optimización de estas es necesario abordar sobre el tema de los archivos PNML y el XES, los cuales son detallados en las siguientes secciones.

1.2.2.1 Formato PNML

PNML[6] (*Petri Net Markup Language*, por sus siglas en inglés), es un formato de intercambio de red de Petri diseñado para ser independiente de herramientas y plataformas específicas. Un archivo que cumple los requerimientos del formato PNML es llamado archivo de red de Petri. Cada red de Petri consiste en objetos que básicamente representa su estructura de grafo. Cada objeto dentro de un archivo de red de Petri tiene un identificador único, el cual puede usarse para referirse a este objeto. En un PNML básico, un objeto es una plaza, transición o arco. Por conveniencia un lugar o una transición son nombrados nodo (Billington, et al., 2003).

Para darle más significado a un objeto, cada uno puede tener etiquetas. Una etiqueta representa el nombre de un nodo, la marca inicial de una plaza, la guarda de una transición, o la inscripción de un arco. Existen dos tipos de etiquetas: las anotaciones que comprenden la información que es mostrada como texto cerca del objeto correspondiente y los atributos que en este caso son los que especifican una propiedad gráfica de un objeto. La etiquetas que se emplean en un archivo de red de Petri dependen del tipo de red de Petri que se esté modelando (Billington, et al., 2003). La figura 3 muestra las etiquetas básicas que se pueden encontrar en un archivo PNML de una red de Petri.

En un archivo PNML de una red de Petri se pueden encontrar etiquetas que muestran información geográfica de los objetos que lo conforman. En la figura 4 se puede observar las etiquetas de información gráfica que conforman este archivo PNML. Cada objeto y cada notación están equipados por información gráfica. Para un nodo se incluye la posición; para una anotación se incluye su posición relativa con respecto al objeto correspondiente. Además, puede haber información adicional correspondiente al tamaño, color, y formas de los nodos o arcos, o concerniente al color, fuente y tamaño de fuente de las etiquetas (Billington, et al., 2003).

[6] *https://www.pnml.org*

Class	XML element	XML Attributes
PetriNetFile	<pnml>	
PetriNet	<net>	id: ID
		type: anyURI
Place	<place>	id: ID
Transition	<transition>	id: ID
Arc	<arc>	id: ID
		source: IDRef (Node)
		target: IDRef (Node)
Page	<page>	id: ID
RefPlace	<referencePlace>	id: ID
		ref: IDRef (Place or RefPlace)
RefTrans	<referenceTransition>	id: ID
		ref: IDRef (Transition or RefTrans)
ToolInfo	<toolspecific>	tool: string
		version: string
Graphics	<graphics>	

Figura 3. Etiquetas básicas de PNML. Fuente (Billington, et al., 2003).

XML element	Attribute	Domain
<position>	x	decimal
	y	decimal
<offset>	x	decimal
	y	decimal
<dimension>	x	nonNegativeDecimal
	y	nonNegativeDecimal
<fill>	color	CSS2-color
	image	anyURI
	gradient-color	CSS2-color
	gradient-rotation	{vertical, horizontal, diagonal}
<line>	shape	{line, curve}
	color	CSS2-color
	width	nonNegativeDecimal
	style	{solid, dash, dot}
<font>	family	CSS2-font-family
	style	CSS2-font-style
	weight	CSS2-font-weight
	size	CSS2-font-size
	decoration	{underline, overline, line-through}
	align	{left, center, right}
	rotation	decimal

Figura 4. Etiquetas de información gráfica de un archivo PNML. Fuente (Billington, et al., 2003).

1.2.2.2. Formato XES

Han existido ya varios formatos para almacenar la información de ejecuciones de un proceso de negocio junto a sus características y propiedades. En el 2009 se define un nuevo estándar: XES[7] (*Extensible Event Stream*, por sus siglas en inglés) por Günther (2009), el cual ha tenido gran aceptación internacionalmente. En septiembre del 2010 el estándar XES fue adoptado por la IEEE[8] y es soportado en la actualidad por diversas herramientas.

El XES está basado en la estructura XML[9] (*eXtensible Markup Language*, por sus siglas en inglés) y consiste en un estándar para el intercambio de los datos de un registro de eventos entre herramientas o aplicaciones. Este contiene una jerarquía definida con restricciones para determinadas etiquetas XML, dando cierto orden a la información. La jerarquía de las etiquetas está compuesta por el *log* en lo más alto, seguido por la *trace* y luego el *event*, que en lo adelante son llamados registro, traza y evento. Existen además *extension*, *global* y *classifier*, que en lo adelante serán llamados extensión, global y clasificador por ese orden. El estándar XES posee cinco tipos de atributos: *string, int, date, float y boolean*, en los cuales se encuentra toda la información del registro; estos están representados por las etiquetas XML *<string>, <int>, <date>, <float> y <boolean>* respectivamente (Günther, 2009).

El registro contiene información relacionada con un proceso específico. Representado por la etiqueta XML ***<log>,*** cada registro contiene un número finito de extensiones, globales, clasificadores, atributos y trazas en ese orden. A grandes rasgos se puede decir que las extensiones definen conjuntos de atributos en cualquier nivel de la jerarquía del registro XES. De esta forma provee puntos de referencia para la interpretación de estos atributos. Por otra parte las etiquetas global son usadas al definir atributos globales que han sido diseñados para estar disponibles y apropiadamente definidos, en todo el documento, para cada elemento en su respectivo nivel. Así, por ejemplo, un atributo definido para el nivel de evento va a estar disponible para

[7] *https://www.xes-standard.org*
[8] *https://www.ieee.org*
[9] *https://www.w3.org/XML/*

cada evento en cada traza. Los clasificadores son propiedades obligatorias que le asignan una identidad a cada evento, haciéndolo comparable con otros eventos (Günther, 2009).

Un evento representa el nivel atómico de una actividad que ha sido observada durante la ejecución del proceso. En el modelo de datos del XES se propone un modelo transaccional que define el estado en el cual se puede encontrar una actividad, este se ve reflejado en la figura 5.

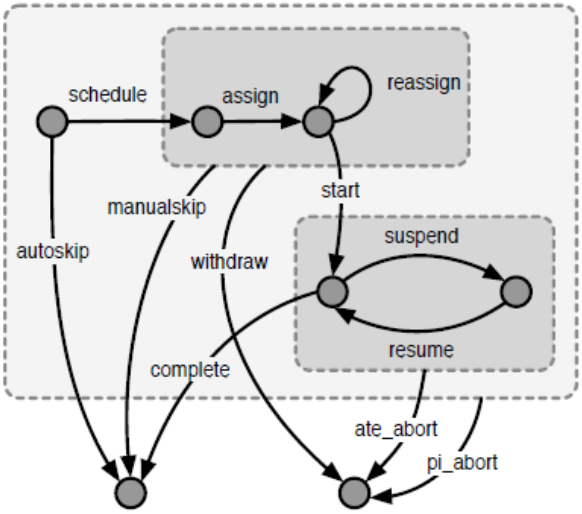


Figura 5. Modelo transaccional del estado de las actividades. Fuente (Günther y Verbeek, 2014).

Los estados de las actividades están representados por el atributo transición y tiene los siguientes valores: *schedule:* la actividad está programada para su ejecución, *assign:* la actividad es asignada a un recurso para su ejecución, *withdraw:* la asignación ha sido revocada, *reassign:* la asignación tras una revocación previa, *start:* comienza la ejecución de la actividad, *suspend:* la ejecución se está suspendiendo, *resume:* la ejecución se reinicia, *pi_abort:* la ejecución completa del proceso se cancela para este caso, *ate_abort:* la ejecución de la actividad es abortada, *complete:* se completa la ejecución de la actividad, *autoskip:* la actividad ha sido omitida por el sistema, *manualskip:* la actividad ha sido omitida a propósito y *unknown:* cualquier transición no capturada por las categorías anteriores (Günther y Verbeek, 2014).

Previamente conocida las notaciones de modelado de los procesos de negocio, se procede a la modelación del proceso en notación BPMN utilizando la herramienta Bonita Studio. Posteriormente se convierte este modelo

BPMN al modelo matemático, en este caso una red de Petri. Finalmente, con el modelo del proceso en una red de Petri y su registro de eventos se puede efectuar la optimización de este. Para ello en la siguiente sección se detallan aspectos generales sobre la optimización multiobjetivo reflejando las pautas necesarias para la investigación.

1.3. Optimización Multiobjetivo

Los problemas de optimización multiobjetivo (MOP: *Multiobjective Optimization Problem*, por sus siglas en inglés*)* se separan de la optimización convencional de un único objetivo, ya que, en la primera generalmente no se entrega una única solución. En vez de esto, el MOP genera un conjunto de posibles soluciones sobre las cuales los decisores deben seleccionar cual adoptar, basado en una evaluación del desempeño de la misma en todos los objetivos (Miettinen, 2008).

Un MOP general incluye un conjunto de n parámetros (variables de decisión), un conjunto de k funciones objetivo, y un conjunto de m restricciones. Las funciones objetivo y las restricciones son funciones de las variables de decisión. Para la siguiente investigación se plantea el MOP de la siguiente manera:

$$\textbf{\textit{Optimizar (Maximizar/Minimizar)}}\ \ y = f_{(x)} = f_1(x), f_2(x), \dots, f_k(x))$$
$$\textbf{\textit{sujeto a}}\ \ e_{(x)} = e_1(x), e_2(x), \dots, e_m(x)) \geq 0$$
$$\textbf{\textit{donde}}\ \ \boldsymbol{x} = (\boldsymbol{x_1}, \boldsymbol{x_2}, \dots \boldsymbol{x_n}) \in \boldsymbol{X}$$
$$\boldsymbol{y} = (\boldsymbol{y_1}, \boldsymbol{y_2}, \dots \boldsymbol{y_k}) \in \boldsymbol{Y}$$

Siendo $\boldsymbol{y}$ el vector objetivo y $\boldsymbol{x}$ el vector de decisión. El espacio de decisión se denota por $\boldsymbol{X}$ y al espacio objetivo por $\boldsymbol{Y}$. Optimizar, dependiendo del problema puede significar igualmente maximizar o minimizar. El conjunto de restricciones $e(x) \geq 0$ determina el conjunto de soluciones factibles $\boldsymbol{X_f}$ y su correspondiente conjunto de vectores objetivos factibles $\boldsymbol{Y_f}$.

El **conjunto de soluciones factibles** $\boldsymbol{X_f}$ Se define como el conjunto de vectores de decisión x que satisface los requerimientos $e(x)$:

$$\boldsymbol{X_f = \{x \in X |\, e(x) \geq 0\}}$$

La imagen de X_f es decir, la región factible del espacio objetivo, se denota por:

$$\boldsymbol{Y_f = f(X_f) = \cup_{x \in X_f} \{y = f(x)\}}$$

De estas definiciones se tiene que cada solución del MOP en cuestión consiste de una n tupla $x = (x_1, x_2, \dots, x_n)$, que conduce a un vector objetivo $y = (f_1(x), f_2(x), \dots, f_k(x))$, donde cada x debe cumplir con el conjunto de restricciones $e(x) \geq 0$.

El problema de optimización consiste en hallar la $\boldsymbol{x}$ que tenga el "mejor valor" de $\boldsymbol{f(x)}$. En general y según ya se ha introducido, no existe un único "mejor valor", sino un conjunto de soluciones. Entre estas, ninguna se puede considerar mejor a las demás si se tienen en cuenta todos los objetivos al mismo tiempo. Este hecho deriva de que puede existir, y generalmente existe, un conflicto entre los diferentes objetivos que componen el problema. Por ende, al tratar con MOPs se precisa de un nuevo concepto de **"óptimo"**. El concepto predominante en la definición de un punto óptimo es el de la **Optimalidad de Pareto**, donde se define que un punto, $x^* \in X$ es un **óptimo de Pareto** si y sólo si no existe otro punto $x^* \in X$, tal que $F(x) \leq F(x^*)$, y $F_i(x) < F_i(x^*)$ para al menos una función (Chong y Zak, 2013).

1.3.1 Optimalidad de Pareto

En los problemas de optimización de un solo objetivo, el algoritmo se concentra en encontrar el óptimo global (o su vecindad). Sin embargo, en los problemas multiobjetivos se trata con un valor de aptitud por cada función objetivo. Esto hace que la noción de óptimo se torne más compleja en este contexto. En los MOP, con objetivos en conflicto, se busca un conjunto de soluciones óptimas, en lugar de una sola solución. La noción de optimalidad que se acepta comúnmente es aquella que propuso originalmente Francis Ysidro Edgeworth (Edgeworth, 1881) y que más tarde generalizara Wilfredo Pareto (Pareto, 1896).

La optimalidad de Pareto es el concepto predominante para determinar las soluciones que pertenecen a un conjunto de puntos óptimos. La autora de esta investigación asume el criterio de optimalidad de Pareto definida en Zitzler, et al. (2008) de la siguiente manera:

Optimalidad Pareto: Dado un vector de decisión $\boldsymbol{x} \in \boldsymbol{X}_f$ y su correspondiente vector objetivo $\boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x}) \in \boldsymbol{Y}_f$, se dice que x es no dominado respecto a un conjunto $\boldsymbol{A} \subseteq \boldsymbol{X}f$ si y solo si:

$$\forall\, a \in A: (x > a \vee x \sim a)$$

En caso que x sea no dominado respecto a todo el conjunto X_f, y solo en ese caso, se dice que x es una **solución Pareto óptima** ($\boldsymbol{x} \in \boldsymbol{X}_{true}$, el conjunto Pareto óptimo real). Mientras que la y correspondiente es parte del **frente Pareto óptimo real** $\boldsymbol{Y}_{true}$. Estos se definen a continuación:

Conjunto Pareto óptimo y frente Pareto óptimo: Dado el conjunto de vectores de decisión factibles $\boldsymbol{X}f$. Se denomina $\boldsymbol{X}_{true}$ al conjunto de vectores de decisión no dominados que pertenecen a $\boldsymbol{X}f$, es decir:

$$\boldsymbol{X}_{true} = \{\boldsymbol{x} \in \boldsymbol{X}_f \mid \boldsymbol{x} \text{ es no dominado con respecto a } \boldsymbol{X}f\}$$

El conjunto $\boldsymbol{X}_{true}$ también es conocido como el conjunto Pareto óptimo. Mientras que el conjunto correspondiente de vectores objetivo $\boldsymbol{Y}_{true} = \boldsymbol{f}(\boldsymbol{X}_{true})$ constituye el frente Pareto óptimo.

A continuación, se exponen las **Dominancias de Pareto** en dependencia del tipo de optimización a realizar (minimización o maximización), al punto de que un objetivo puede ser maximizado mientras que otro puede ser minimizado tomado en Zitzler, et al. (2008).

Dominancia Pareto en un contexto de Maximización. Para dos vectores objetivos a y b,

$\boldsymbol{a} \prec \boldsymbol{b}$	(a domina a b)	si y solo si	$a > b$
$\boldsymbol{b} \prec \boldsymbol{a}$	(b domina a a)	si y solo si	$b > a$
$\boldsymbol{a} \sim \boldsymbol{b}$	(a y b no son comparables)	si y solo si	$a \not\geq b \wedge b \not\geq a$

Dominancia Pareto en un contexto de Minimización. Para dos vectores objetivos a y b,

$$\begin{array}{lll} \boldsymbol{a \prec b} \quad (a \text{ domina a } b) & \text{si y solo si} & a < b \\ \boldsymbol{b \prec a} \quad (b \text{ domina a } a) & \text{si y solo si} & b < a \\ \boldsymbol{a \sim b} \quad (a \text{ y } b \text{ no son comparables}) & \text{si y solo si} & a \not\leq b \wedge b \not\leq a \end{array}$$

En esta investigación se pretende optimizar procesos de negocio atendiendo a dos funciones objetivos (costo y completitud), dado que se quiere minimizar el costo y maximizar la completitud. En este caso se hace oportuno plantear la dominancia de Pareto para un contexto de minimización/maximización de la siguiente manera:

Dominancia Pareto en un contexto de Minimización-Maximización. Para dos vectores objetivos a y b,

$\boldsymbol{a \prec b}$ (a domina a b) **si y solo si** $\forall i = 1 \ldots n, a_i \leq b_i$ para los objetivos i que se minimizan

o $a_i \geq b_i$ para los objetivos que se maximizan

$\boldsymbol{a \sim b}$ (a y b no son comparables) **si y solo si** $a_i > b_i$ para al menos un objetivo i que se minimiza

o $a_i < b_i$ para al menos un objetivo i que se minimiza

siendo n el número de objetivos.

En la mayoría de los problemas multiobjetivo no es fácil obtener una descripción exacta del conjunto de Pareto debido a que puede abarcar un número muy grande o infinito de puntos. Aunque en teoría es posible encontrar estos puntos exactamente, es computacionalmente difícil y costoso. Se puede destacar que, una descripción aproximada de este conjunto de Pareto puede ser obtenida mediante un algoritmo de aproximación lo que puede resultar conveniente, dado que requiere menos esfuerzo y muchas veces puede ser lo suficientemente preciso para desempeñar el papel del conjunto de soluciones y apoyar con eficacia la toma de decisiones. Los enfoques de aproximación emplean un método iterativo para producir puntos que se aproximen al conjunto de Pareto. Algunos enfoques son exactos y se basan en algoritmos que aseguran la obtención de óptimos de Pareto, mientras que otros enfoques se basan en heurísticas que no necesariamente aseguran que los puntos encontrados sean óptimos de Pareto (Ehrgott y Wiecek, 2005).

Existe una gran variedad de investigaciones relacionadas con el desarrollo de algoritmos que facilitan aplicar la optimización multiobjetivo a diversos problemas que a diario se presentan. Diversos autores como Back

(1996), Deb (1999), Talavera (2005) y Puris (2009) han demostrado que los algoritmos evolutivos son eficientes en múltiples problemas de optimización. Estos plantean que los algoritmos evolutivos hacen posible que se manejen espacios de búsqueda de casi cualquier tamaño y características; y debido a su paralelismo inherente, se puedan generar múltiples soluciones en una sola corrida. Con la afirmación previa, la autora de esta investigación asume el empleo de los algoritmos evolutivos para lidiar con el problema multiobjetivo.

1.4. Algoritmos Evolutivos en la Optimización Multiobjetivo

Los Algoritmos Evolutivos (*EA: Evolutionary Algorithm*, por sus siglas en inglés) se refiere a técnicas de búsqueda y optimización inspiradas en el modelo de la evolución propuesto por Charles Darwin (Darwin, 1959). Según Back (1996), son métodos de optimización y búsqueda de soluciones basados en los postulados de la evolución biológica. En ellos se mantiene un conjunto llamado población, cuyos elementos representan posibles soluciones, las cuales se mezclan, y compiten entre sí, de tal manera que las más aptas son capaces de prevalecer a lo largo del tiempo, evolucionando hacia mejores soluciones. La población, en el contexto de la computación evolutiva de forma general se refiere a un conjunto de posibles soluciones (soluciones factibles) del problema que se desea resolver (Back, 1996).

En la investigación se asume el criterio de Puris (2009), el cual plantea que los EA se caracterizan por operar sobre una población inicial (usualmente generada aleatoriamente) que sufre un proceso iterativo que conduce a la evolución de dicha población (a semejanza de la natural). A continuación, se muestran los pasos a seguir en un algoritmo evolutivo:

- Se aplica un operador de selección que determina la probabilidad de cada individuo de perdurar en la generación siguiente, construyéndose así una población temporal.
- Se aplican operadores evolutivos a la población temporal, con lo que se produce un conjunto de nuevas soluciones.
- Se calcula el valor de la función a optimizar en las nuevas soluciones generadas.
- Se obtiene una nueva población a partir de la población temporal y por lo tanto los nuevos individuos generados.

Dichos pasos son realizados hasta que se cumple un criterio de terminación (por ejemplo, se alcanza un límite máximo temporal o un número máximo de evaluaciones de la función objetivo). El procedimiento de crear

nuevos individuos a partir de los existentes en la población, y en el que los mejores sustituyen a los anteriores, es lo que se denomina evolución de la población. Construir los nuevos individuos combinando características de buenas soluciones encontradas hasta el momento (almacenadas en la población) es lo que permite orientar la búsqueda hacia las regiones más prometedoras (Puris, 2009). De forma general, los EA comparten los mismos conceptos básicos, aunque difieren en el mecanismo empleado para codificar las soluciones y los operadores que emplean para producir la siguiente generación.

Aproximadamente desde mediados de la década del 70, en el ámbito de los EAs, se han introducido numerosos algoritmos que se pueden clasificar principalmente como Algoritmos Genéticos, Programación Evolutiva y Estrategias de Evolución, Sistemas Clasificadores y Programación Genética. A pesar de que el principio que los soporta es en apariencia muy simple, estos algoritmos han probado ser herramientas generales, robustas y potentes. En esta investigación se utilizan los algoritmos genéticos para la optimización multiobjetivo ya estos son menos susceptibles a la forma y continuidad de la frontera de Pareto, requieren de poca información del dominio y son relativamente fáciles de usar e implementar. En la siguiente sección se brinda una introducción a estos y por ende se mencionan los algoritmos genéticos utilizados en la investigación.

1.4.1. Algoritmos Genéticos

Los Algoritmos Genéticos (*GA: Genetic Algorithm*, por sus siglas en inglés) propuesto por Goldberg (1998), surgen como herramientas para la solución de complejos problemas de búsqueda y optimización; producto del análisis de los sistemas adaptativos en la naturaleza y como resultado de abstraer la esencia de su funcionamiento. El término Algoritmo Genético se usa por el hecho de que estos simulan los procesos de la evolución a través del uso de operadores genéticos que operan sobre una población de individuos que "evoluciona" de una generación a otra.

Según González (2014), los GA trabajan a partir de una población inicial de estructuras artificiales que se van modificando iterativamente durante el proceso de búsqueda, a través de la aplicación de los siguientes operadores genéticos:

- Operador de Selección
- Operador de Cruzamiento
- Operador de Mutación

González (2014), en su investigación plantea que un algoritmo genético puede presentar diversas variaciones, dependiendo de cómo se aplican los operadores genéticos (selección, cruzamiento, mutación), de cómo se realiza la selección y de cómo se decide el reemplazo de los individuos para formar la nueva población. A continuación, se describen los pasos del funcionamiento del algoritmo genético:

- **Inicialización:** Se genera aleatoriamente la población inicial, que está constituida por un conjunto de cromosomas los cuales representan las posibles soluciones del problema. En caso de no hacerlo aleatoriamente, es importante garantizar que, dentro de la población inicial, se tenga la diversidad estructural de estas soluciones para tener una representación "uniforme" del espacio de soluciones.
- **Evaluación:** A cada uno de los cromosomas de esta población se aplicará la función de aptitud para saber cómo de "buena" es la solución que se está codificando.
- **Condición de término:** El algoritmo debe detenerse cuando se alcance la solución óptima, pero ésta generalmente se desconoce, por lo que se deben utilizar otros criterios de parada. Frecuentemente se utilizan dos criterios: número máximo de generaciones o número máximo de evaluaciones de la función objetivo. Mientras no se cumpla alguna de estas condiciones de parada se hace lo siguiente:

  - **Selección:** Después de saber la aptitud de cada cromosoma se procede a elegir los cromosomas que serán cruzados en la siguiente generación. Los cromosomas con mejor aptitud tienen mayor probabilidad de ser seleccionados.
  - **Cruzamiento:** El cruzamiento es el principal operador genético, representa la reproducción sexual, opera sobre dos cromosomas a la vez para generar dos descendientes donde se combinan las características de ambos cromosomas padres.
  - **Mutación**: Modifica al azar parte del cromosoma de los individuos y permite alcanzar zonas del espacio de búsqueda que no estaban cubiertas por los individuos de la población actual.

- **Reemplazo:** Una vez aplicados los operadores genéticos, se seleccionan los mejores individuos para conformar la población de la generación siguiente.

En el transcurso de los años diferentes autores como Srinivas y Deb (1994), Zitzler y Thiele (1999), Zitzler, et al. (2000), Deb y Goel (2001), Deb, et al. (2002), han propuesto diversos algoritmos genéticos para problemas multiobjetivos buscando siempre obtener un frente de Pareto mejor delineado y con mejores soluciones. En la actualidad, se ha sumado a este objetivo la eficiencia del algoritmo, es decir que la solución que arroje sea en el menor tiempo posible.

Dado el auge que han tenido estos algoritmos genéticos en la solución de problemas multiobjetivos se han desarrollado diferentes metodologías que apuntan a obtener muy buenas soluciones en estos tipos de problemas. Los algoritmos genéticos multiobjetivos considerados para esta investigación son el NSGAII propuesto por Deb, et al. (2002) y el SPEA2 por Zitzler, et al. (2000), ya que estos son los algoritmos más representativos para resolver problemas de optimización multiobjetivo. Estos son considerados como algoritmos de segunda generación por su estrategia elitista, la cual permite que no se desaprovechen buenas soluciones cuando hay un cambio generacional y a su vez son mejoras de sus versiones anteriores como es el caso del SPEA y NSGA.

1.4.1.1. Algoritmo NSGAII

En el año 2000 los investigadores Deb, Pratap, Agarwal, Meyarivan y sus estudiantes desarrollaron un algoritmo llamado Algoritmo Genético de Clasificación No Dominada II (NSGAII, por sus siglas en inglés) descrito en Deb, et al. (2002). El NSGAII usa un procedimiento rápido para organizar la población por no dominancia, un enfoque para preservar el elitismo y un operador diferente al nicho, para dispersar los individuos en la frontera de Pareto.

En este algoritmo, la población descendiente Q_t (tamaño N) es creada en primera instancia usando la población de padres P_t (tamaño N). Después de esto, las dos poblaciones son combinadas para formar R_t de tamaño $2N$. Después de lo anterior, mediante un ordenamiento no dominado se clasifica la población R_t en diferentes frentes de Pareto. Aunque esto requiere un mayor esfuerzo, se justifica por el hecho de permitir una verificación global de dominancia entre la población de padres y descendientes. Una vez el proceso de ordenamiento no dominado ha finalizado, la nueva población es generada a partir de las configuraciones de los frentes no dominados. Esta nueva población empieza a ser construida con el mejor frente no dominado

(F_1), continúa con las soluciones del segundo frente (F_2), tercero (F_3) y así sucesivamente. Como la población R_t es de tamaño $2N$, y solamente existen N configuraciones que conforman la población descendiente, no todas las configuraciones de los frentes pertenecientes a la población R_t podrán ser acomodadas en la nueva población. Aquellos frentes que no pueden ser acomodados desaparecen (Deb, et al., 2002).

Cuando se está considerando el último frente, la soluciones que hacen parte de este pueden exceder las restantes por acomodar en la población descendiente, dicha situación se ilustra en la figura 6. En este caso resulta útil emplear alguna estrategia que permita seleccionar las configuraciones situadas en un área poco poblada (alejada de otras soluciones) para llenar las posiciones restantes de la población descendiente a cambio de optar por escoger configuraciones de forma aleatoria.

Lo anterior es poco relevante en los primeros ciclos generacionales del algoritmo, ya que en esta etapa existen muchos frentes que sobreviven hacia la siguiente generación, pero a medida que el proceso avanza, muchas configuraciones pasan a ser parte del primer frente inclusive haciendo que dicho frente tenga más de N individuos, por lo que se hace importante que las configuraciones no rechazadas sean de buena calidad y escogidas mediante una metodología que garantice la diversidad. La idea es que siempre se promuevan las configuraciones que aseguren diversidad dentro del mismo frente de Pareto. Cuando la población en su totalidad converge al frente de Pareto óptimo, el algoritmo asegura que las soluciones estén distanciadas una de la otra.

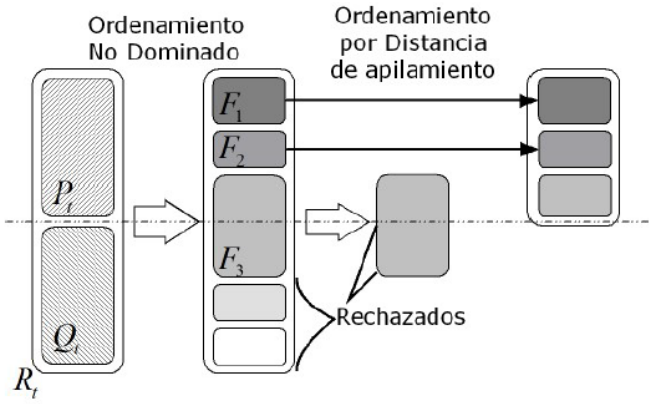


Figura 6. Diagrama esquemático del mecanismo de promoción de individuos del NSGAII. Fuente (Deb, et al., 2002).

En el algoritmo NSGAII inicialmente se crea una población (aleatoria o mediante una técnica de inicialización) de padres P_0. La población se ordena de acuerdo a los niveles de no dominancia (ordenamiento de los frentes de Pareto, F_1 , F_2,…). A cada solución se le asigna una función *fitness* de acuerdo a su nivel de no dominancia (1 es el mejor nivel) y se entiende que durante el proceso debe disminuir dicha función. La selección por torneo (empleando un operador de torneo para apilamiento, descrito posteriormente), el cruzamiento y la mutación son utilizadas para crear la población de descendientes Q_0 de tamaño N. Los principales pasos del algoritmo NSGAII son descritos a continuación:

1. Combinar las poblaciones de padres y descendientes para crear $R_t = P_t \cup Q_t$. Realizar el ordenamiento no dominado a R_t e identificar los frentes F_i, *i =1,2,…, etc.*
2. Hacer $P_{t+1} = \emptyset, e\ i = 1$. Mientras $|P_{t+1}| + |F_i| < N$ hacer $|P_{t+1}| = |P_{t+1}| \cup |F_i|$ $e\ i = i + 1$.
3. Realizar ordenamiento por apilamiento $F_i' < C$ Descrito posteriormente e incluir en P_i las $N - |P_{t+1}|$ soluciones más esparcidas usando los valores de distancia de apilamiento asociadas al frente F_i.
4. Crear la población descendiente Q_{i+1} a partir de P_{i+1} usando la selección por torneo para apilamiento, cruzamiento y mutación.

Selección por torneo para apilamiento:

Este operador $F_i' < C$ compara dos soluciones y elige un ganador del torneo. Se considera que una solución tiene un rango de no dominancia asociado (r_i) y una distancia de apilamiento (d_i). La distancia de apilamiento d_i de una solución i es la medida del espacio de búsqueda alrededor de i que no está ocupado por otra solución en la población. Por medio de estos dos atributos se puede encontrar la mejor configuración a través del operador de selección por torneo para apilamiento como aquella que tiene mejor rango de no dominancia. En el caso de estar situadas en el mismo frente de Pareto la ganadora del torneo es aquella que tiene mejor (más alta) distancia de apilamiento. La distancia de apilamiento $d_{I_j^m}$ para cada solución j según un índice I, es determinada algorítmicamente, haciendo uso de la ecuación siguiente propuesta en Deb, et al.(2002):

$$d_{I_j^m} = d_{I_j^m} + \frac{f_m^{(I_{j+1}^m)} - f_m^{(I_{j-1}^m)}}{f_m^{max} - f_m^{min}}$$

Donde, f_m^{max} , f_m^{min} son el valor máximo y mínimo de la función objetivo m, $f_m^{(I_{j+1}^m)}$, $f_m^{(I_{j-1}^m)}$ son las soluciones vecinas a la configuración j para cada una de las funciones objetivo m. Las distancias consideran todas las funciones objetivo y se asigna el valor de infinito a las soluciones extremas del frente de Pareto considerado. Por ser estas las que cuentan con el mejor valor de una de las funciones objetivo del frente, la distancia resultante es la suma de las distancias en cada una de las direcciones de las funciones objetivo del problema. De esta manera el algoritmo utiliza los operadores genéticos básicos y promueve hacia el siguiente ciclo generacional las configuraciones que ocupen los mejores frentes, y las más diversas, a través de las distancias de apilamiento (Deb, et al., 2002).

En la actualidad el NSGAII y el SPEA2 son los algoritmos que mejores encuentran las soluciones muy cercanas a la frontera real de Pareto, por tanto, son tomados como algoritmos de referencia para la comunidad científica y por ende para esta investigación.

1.4.1.2. Algoritmo SPEA2

En el año 2001 Zitzler, Laumanns y Thiele hacen unas correcciones al algoritmo SPEA, al que llaman Algoritmo Evolutivo Pareto de Fuerza 2 (SPEA2, por sus siglas en inglés) propuesto por Zitzler, et al. (2000) y éste último se diferencia del primero por la asignación de la aptitud, la selección de los padres, el operador de truncamiento y en fijar el tamaño del archivo externo para todas sus generaciones.

En este, la función de asignación de aptitud se mejora teniendo en cuenta para cada individuo el número de individuos a los que domina y el número de individuos por los que es dominado. Este esquema también añade una estimación de densidad poblacional. El tamaño N_{Emax} de la población externa P_E (utilizada para elitismo) es fijo, a diferencia del SPEA, en el cual el tamaño de P_E es variable pero acotado. P_E está conformada sólo por individuos no dominados siempre y cuando el número de estos sea mayor o igual que N_{Emax}. En el caso en que el número de individuos no dominados sea menor que N_{Emax} se incluyen individuos dominados dentro de P_E hasta que el tamaño de P_E sea igual a N_{Emax}. La técnica de agrupamiento *(clustering)*, encargada de mantener la diversidad de la población en SPEA, es sustituida por un método de truncamiento, el cual evita eliminar las soluciones extremas del conjunto de soluciones no dominadas. La selección se realiza mediante torneo binario, tomando como criterio de comparación el *fitness* de cada uno de los individuos. SPEA2 asume minimización de *fitness*, por lo tanto, gana el torneo aquel individuo que tenga un menor valor de *fitness*.

1.5. Conclusiones

Sobre la base de los fundamentos teóricos relacionados a la investigación, se pudo ratificar que la aplicación de los algoritmos multiobejtivos en la optimización de los procesos de negocio resulta una buena opción para el desempeño de estos en sus organizaciones, ya que más de un criterio de optimización puede ser tomado y satisfecho simultáneamente. La minería de procesos y la adecuada gestión de procesos de negocio son alternativas que pueden asumir las organizaciones de hoy para enfrentar e instaurar nuevas estrategias que impacten de manera positiva la competitividad y la calidad de sus procesos. En este capítulo se logró definir el problema multiobjetivo y el criterio de dominancia de Pareto para un contexto de maximización/minimización de dos funciones objetivos (costo y completitud). Además, se pudo apreciar que el empleo de los algoritmos evolutivos es una buena opción para este tipo de problemas por las prestaciones que ellos ofrecen al brindar sus soluciones óptimas a estos.

CAPÍTULO 2. OPTIMIZACIÓN MULTIOBJETIVO DE PROCESOS DE NEGOCIO

Una vez estudiado los fundamentos teóricos que sustentan la investigación se hace imprescindible caracterizar las tecnologías, métodos y herramientas necesarias para su desarrollo. Se procede a formular el problema matemático a emplear detallando las funciones objetivos (costo y completitud). La figura 7 muestra un esquema con los pasos a seguir en el desarrollo de la investigación.

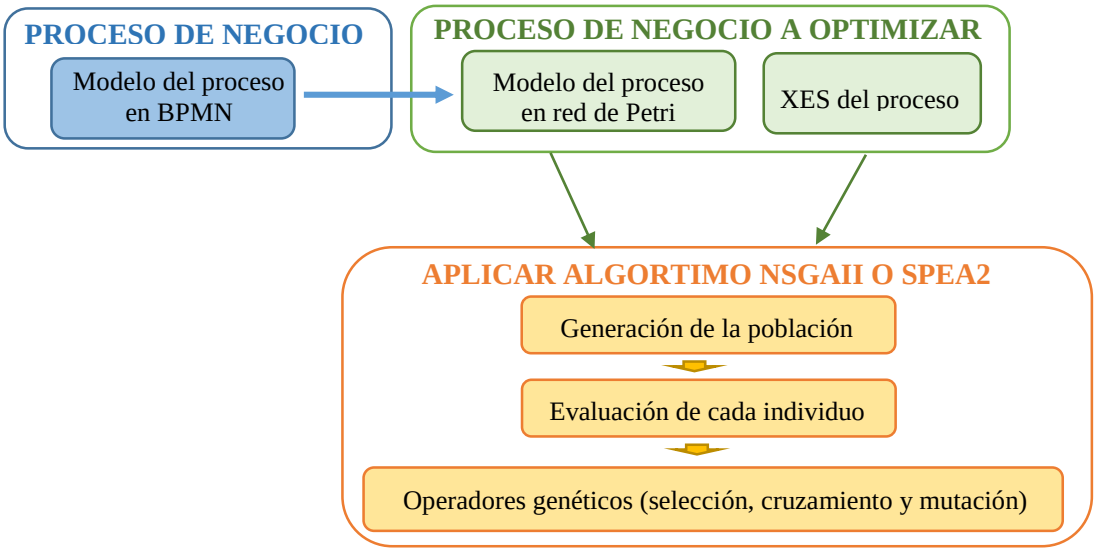


Figura 7. Pasos para optimizar procesos de negocio. Fuente (Elaboración propia).

2.1. Métodos de investigación

Para el desarrollo de la investigación se tuvieron en cuenta los siguientes métodos:

Histórico-Lógico: se utilizó con el objetivo de analizar los problemas de optimización multiobjetivo y el criterio de dominancia de Pareto para contextos de maximización o minimización en más de un objetivo. Esto permitió en la investigación poder definir el problema multiobjetivo y la dominancia de Pareto en un contexto de minimización/maximización de dos objetivos. Para esto se tuvo en cuenta diferentes trabajos relacionados con la optimización multiobjetivo en diferentes contextos.

Inductivo-deductivo: se tuvo en cuenta para el análisis de las diversas aplicaciones de la gestión de procesos de negocio, la minería de procesos y la optimización multiobjetivo en la optimización de los procesos de negocio de las organizaciones, es decir, en el estudio de diferentes investigaciones donde se integran estas disciplinas ofreciendo resultados beneficiosos.

Sistémico: se aplicó para explicar cada una de los pasos para realizar la optimización de los procesos de negocio, partiendo de la representación de sus modelos en BPMN y en redes de Petri. Además de la aplicación de los algoritmos multiobjetivos a estos modelos de procesos de negocio.

Para la evaluación de los algoritmos se utilizaron las métricas de generación de vectores no dominados, la razón de generación de vectores no dominados, la generación real de vectores no dominados y la distancia generacional permitiendo verificar cual algoritmo es el que ofrece las mejores soluciones óptimas del proceso que se está analizando.

2.2. Tecnologías y herramientas utilizadas para la implementación

En esta sección se caracterizan las tecnologías que se utilizaron en la implementación de la aplicación. Se describen las herramientas y *framework* que permitieron el trabajo con los algoritmos multiobjetivos.

2.2.1. Lenguaje de programación Java

La implementación de la aplicación se realizó utilizando el lenguaje de programación Java[10]. Dentro de las principales características se pueden mencionar que es un lenguaje orientado a objetos de propósito general mediante el cual se puede construir cualquier tipo de proyecto, es estable, portable, sencillo y menos complejo que otros lenguajes de alto nivel. De forma general es un lenguaje seguro, robusto, multiprocesos, independiente de arquitectura, compilado y ampliable. Por todas estas características antes mencionadas es que se decide utilizar este lenguaje de programación para el desarrollo de este trabajo, además de que para el

[10] https://www.java.com

manejo con los algoritmos multiobjetivos se utiliza el *framework* JMetal que está desarrollado bajo este mismo lenguaje (Peñarrieta, 2017).

2.2.2. Entorno de desarrollo Netbeans

NetBeans[11] es un entorno de desarrollo integrado (IDE) libre, orientado principalmente al desarrollo de aplicaciones Java. Es una plataforma gratis y *open source* que permite construir aplicaciones completas para el cliente, ya sea de escritorio, móvil, web o de empresa. Dentro de sus principales características se puede mencionar que presenta mejoras en el editor de código, la instalación y actualización es simple, tiene soporte para varios lenguajes como PHP, Java Script, HTML y CSS. Es un producto sin restricciones de uso. Por las facilidades antes mencionadas es que se escoge como entorno de desarrollo Netbeans para el desarrollo de la aplicación (Peñarrieta, 2017).

2.2.3. Framework JMetal

JMetal[12] es un *framework open source* basado en Java para la optimización multiobjetivo con metaheurísticas propuesto por Nebro y Durillo (2014). Es un software libre, cuenta con una arquitectura lo suficientemente genérica para permitir la flexibilidad necesaria de implementar cualquier metaheurística. Jmetal5 ofrece una serie de plantillas que incluyen el comportamiento de las subfamilias de los algoritmos evolutivos. El uso de estas plantillas facilita la reusabilidad de código, es decir, para implementar una variante de un algoritmo determinado solo hay que redefinir los métodos que difieren del algoritmo original. En este trabajo se utilizó el *framework* JMetal para el rediseño de los algoritmos multiobjetivos NSGAII y SPEA2 en función del problema que se aborda. En el anexo 1, en la figura 20 se puede observar las clases diseñadas para el manejo de estos algoritmos en la optimización de los procesos de negocio.

[11] https://netbeans.org
[12] https://jmetal.sourceforge.net

2.2.4. Bonita Studio

Bonita Studio es un software libre que permite a los usuarios diseñar gráficamente procesos de negocio utilizando el estándar BPMN, cuenta con una interfaz muy intuitiva y con soporte para demás acciones dentro del programa como por ejemplo la programación, ejemplos ilustrativos. Es compatible con muchos sistemas operativos, es multidioma. Permite conectar procesos a otras piezas del sistema de información (tales como: mensajería, ERP, ECM, bases de datos...) para generar una aplicación de negocios autónoma accesible como formulario web. Bonita Studio permite también al usuario diseñar gráficamente el formulario web que será mostrado al usuario final para interactuar con el proceso. Además, le permite al usuario comenzar con procesos diseñados con otros estándares y tecnologías tales como XPDL o jBPM. En la presente investigación se utiliza Bonita Studio en su versión 7.5.4 para la modelación del proceso de negocio mediante la notación BPMN (Bonitasoft, 2018).

2.2.5. Yasper

Se utiliza la herramienta Yasper[13] para la importación de los procesos en formato BPMN y convertirlos en una red de Petri, exportando así el proceso en el archivo PNML que es usado para realizar la optimización mediante la aplicación informática. Yasper es una herramienta para especificar y simular modelos de procesos de paso discreto. Fue desarrollado en colaboración entre *TU Eindhoven* y *Deloitte*. Las características que admite son bien conocidas, bien analizadas y existe soporte para ellas en otras herramientas. Además, las herramientas discretas de modelado de procesos en general tienden a soportar técnicas como máquinas de estado, diagramas de flujo y diagramas de actividad, que están estrechamente relacionados con Yasper por modelos (Kees, et al., 2006).

[13] https://yasper.org

2.3. Formulación del problema matemático

En un problema multiobjetivo se pueden escoger tantos objetivos como se desee, pero en esta investigación surge la necesidad de limitar el problema que se va a tratar en cuestión a la optimización de $k = 2$ objetivos; costo y completitud del proceso que se quiere optimizar. El problema queda formulado de la siguiente manera:

$$\textbf{\textit{Optimizar (Maximizar/Minimizar)}} \quad y = f_{(x)} = (f_1(x), f_2(x))$$

$$\textbf{\textit{Sujeto a}} \quad e_1(x) > 0$$

Donde:

$\boldsymbol{f_1}$ $(\boldsymbol{x})$: es la función objetivo de costo del proceso.

$\boldsymbol{f_2}$ $(\boldsymbol{x})$: es la función objetivo de completitud del proceso.

La variable de decisión x en el problema de optimización de un proceso de negocio sería la representación artificial del individuo (es lo llamado codificación de los individuos). En el problema multiobjetivo de esta investigación, se quiere optimizar los modelos de procesos de negocio atendiendo a que la función del costo del proceso se minimice y la función de la completitud se maximice, es decir obtener las posibles mejoras del desempeño del proceso con un menor costo y una mejor completitud. Para una mejor comprensión de las funciones objetivos se detallan cada una de ellas a continuación.

2.3.1. Función Objetivo: costo del proceso

La función objetivo costo del proceso consiste en sumar el costo de cada transición, es decir, la actividad en el proceso; disparada durante el parseo de las trazas. Solo que si una transición disparada por esta vía no tiene los tokens necesarios para su disparo entonces su costo no se añadirá al costo total. De esta forma se asegura que mientras más mala sea una solución (según el cálculo de la completitud) para el registro de eventos, su costo puede ser menor.

2.3.2. Función Objetivo: completitud del proceso

Un proceso de negocio está representado mediante su modelo, por lo que, un modelo será más completo en la medida que sea capaz de procesar un mayor número de trazas en el registro de eventos. Una de las formas

básicas de obtener la completitud sería dividir la cantidad de trazas ejecutadas correctamente entre el total de trazas existentes en el registro de eventos. No obstante, esta forma es bastante limitada pues no contempla la semántica de los conectores *AND/XOR-SPLIT/JOIN*, ni los errores locales a las actividades ocurridos durante la ejecución de las trazas. Medeiros en su investigación propone que la completitud de un modelo, es el *fitness* (criterio para seleccionar a las mejores soluciones) utilizado en el algoritmo *Genetic Miner* y en la herramienta académica ProM [14], donde es posible simular este algoritmo, a lo cual denominan *Continuous Semantics* (Medeiros, 2006).

En la aplicación informática para la optimización de los procesos de negocio en desarrollo, se tiene en cuenta la fórmula que se describe a continuación, para formalizar el cálculo de la completitud propuesta por Medeiros (2006), la cual contempla las actividades ausentes y abandonadas a la hora de evaluar la completitud empleando el algoritmo *Genetic Miner.*

$$PF_{Comp} = \frac{act.Eject(AP, CM) - penalidad}{total\ Evt(Q)}$$

$$penalidad = \frac{totMarcAusent(AP, CM)}{totalTrazas(AP) - trazasMarcAusent(AP, CM) + 1} + \frac{totMarcAband(AP, CM)}{totalTrazas\ (AP) - trazasMarcAband(AP, CM) + 1}$$

Donde **AP** representa las trazas, **CM** es la matriz causal (codificación seleccionada para representar a cada individuo o solución), $\boldsymbol{act.Eject(AP, CM)}$ a las actividades ejecutadas correctamente, $\boldsymbol{total\ Evt(Q)}$ al total de eventos en el registro de eventos (XES del proceso), $\boldsymbol{totMarcAusent(AP, CM)}$ al total de tokens necesarios para disparar una actividad (transición en la red de Petri) pero que no se encontraban en la plaza de entrada de dicha actividad, $\boldsymbol{totalTrazas(AP)}$ al total de trazas en el registro de eventos,

[14] *https://www.promtools.org*

$\boldsymbol{trazasMarcAusent}(\boldsymbol{AP}, \boldsymbol{CM})$ al número de trazas donde se perdieron tokens, $\boldsymbol{totMarcAband}(\boldsymbol{AP}, \boldsymbol{CM})$ al número de tokens que quedaron en la red cuando terminó la ejecución de cada traza, $\boldsymbol{totalTrazas}$ $(\boldsymbol{AP})$ al total de trazas en el registro de eventos, $\boldsymbol{trazasMarcAband}(\boldsymbol{AP}, \boldsymbol{CM})$ al número de trazas donde quedaron tokens durante su ejecución.

Durante el parseo de las trazas sucede con frecuencia que un evento (actividad) correspondiente a la transición real que se representa en la red de Petri (no oculta) no esté habilitada. En este caso se incrementa la variable $\boldsymbol{totMarcAusent}(\boldsymbol{AP}, \boldsymbol{CM})$ y se asume que dicha actividad estaba habilitada para continuar el parseo.

Una vez ya enunciadas las funciones objetivos a utilizar para la optimización de los procesos de negocio, es necesario establecer la codificación a tener en cuenta para representar el conjunto de soluciones de las posibles mejoras del proceso.

2.4. Codificación para representar el conjunto de soluciones

Un algoritmo evolutivo manipula una población de individuos la cual cambiará generacionalmente. Su codificación resulta vital para luego aplicar los operadores genéticos (cruzamiento y mutación). En este trabajo se utilizará la codificación propuesta por Medeiros (2006), la cual consiste en una matriz causal. Descubrir redes de Petri a partir de un registro de eventos puede resultar muy complejo. Mientras que los eventos fácilmente se asocian con las transiciones, la semántica de los lugares no puede ser inferida directamente de un registro de eventos. Por otro lado, en un procedimiento iterativo donde constantemente se modifican los conectores AND/XOR-SPLIT/JOIN, la transformación de una red de Petri resulta engorrosa. En este sentido, las matrices causales están centradas únicamente en las actividades y son tan expresivas como las redes de Petri.

La matriz causal muestra cuáles actividades habilitan la ejecución de otras a partir de las funciones de entrada y salida (*Input Function* y *Output Function*) que almacenan respectivamente las actividades que preceden y suceden a una actividad dada. Las funciones de entrada (I) y salida (O) son dos conjuntos de subconjuntos que contienen a las actividades del modelo. A continuación, se ofrece la definición de matriz causal tomada de Medeiros (2006) y utilizada por ende en la presente investigación.

Matriz Causal: Sea LS un conjunto de etiquetas. Una Matriz Causal es una tupla $CM = (A, C, I, O, Label)$, donde:

$\boldsymbol{A}$: es un conjunto finito de actividades,

$\boldsymbol{C \subseteq A \times A}$**:** es la relación de causalidad,

$\boldsymbol{I: A \rightarrow P(P(A))}$**:** es la función que establece las condiciones de entrada,

$\boldsymbol{O: A \rightarrow P(P(A))}$**:** es la función que establece las condiciones de salida,

$\boldsymbol{label \in A \rightarrow LS}$: es la función que asocia a cada actividad A en una etiqueta en LS.

De modo que:

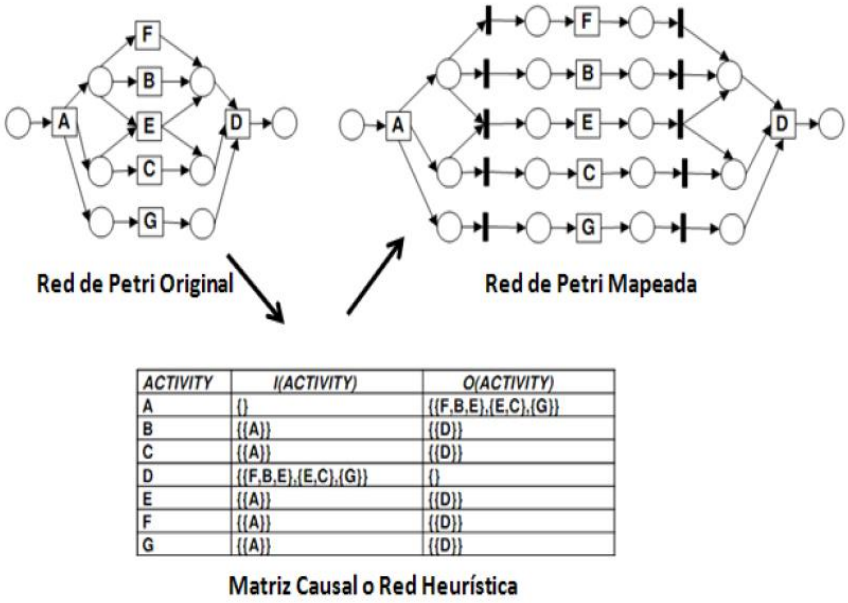
$C = \{(a_1, a_2) \in A \times A | a_1 \in \cup I(a_2)\}$,

$C = \{(a_1, a_2) \in A \times A | a_2 \in \cup O(a_1)\}$,

$C = \{(a_o, a_i)^1 \in A \times A | I(a_2) = \emptyset \wedge O(a_i) = \emptyset\}$, es un grafo fuertemente conexo,

$label \in A \rightarrow LS$ es una función inyectiva, lo que quiere decir que si a_1 , $a_2 \in A$ y $label\ (a_1) = label\ (a_2)$ entonces $a_1 = a_2$.

Las actividades contenidas en un mismo subconjunto, están relacionadas por una disyunción (OR) mientras que los diferentes subconjuntos establecen una relación de conjunción (ADN). De esta manera, cada función $I(Actividad)$, $O\ (Actividad)$representa una conjunción de disyunciones exclusivas. Una actividad puede aparecer en uno o más subconjuntos. Toda matriz causal debe cumplir además, que dadas dos actividades $x, y, x \in I(y)$ ssi $y \in O(x)$. En la figura 8 se ilustra cómo la matriz causal representa las mismas dependencias entre las tareas que la red de Petri original.



ACTIVITY	I(ACTIVITY)	O(ACTIVITY)
A	{}	{{F,B,E},{E,C},{G}}
B	{{A}}	{{D}}
C	{{A}}	{{D}}
D	{{F,B,E},{E,C},{G}}	{}
E	{{A}}	{{D}}
F	{{A}}	{{D}}
G	{{A}}	{{D}}

Matriz Causal o Red Heurística

Figura 8. Relación entre red de Petri y matriz causal. Fuente (Medeiros, 2006).

Por ejemplo, considérese la actividad en la red de Petri original de la figura 8. La matriz causal establece que $I(D) = \{\{F, B, E\}\{E, C\}\{G\}\}$ porque (D) está habilitada por una construcción *AND/JOIN* que tiene tres lugares. Nótese que $I(A) = \{\}$ y $O(D) = \{\}$ porque A comienza el proceso, al no estar precedida por ninguna actividad, mientras que D no está sucedida por actividad alguna pues termina el proceso.

La aplicación de los operadores genéticos sobre matrices causales, resulta un proceso mucho más simple, con relación a su modelo equivalente visualizado como red de Petri, pues solamente hay que adicionar o eliminar elementos de conjuntos. Por otra parte, reproducir la ejecución de un proceso según las secuencias de eventos que ocurren en las trazas, aspecto imprescindible para calcular la adaptabilidad de un individuo se torna compleja. Una vez que se haya mapeado la red de Petri que representa al proceso de negocio en una matriz causal se está apto para aplicar los algoritmos NSGAII y SPEA2.

2.5. Pasos a realizar para la optimización de los procesos de negocio

Como se abordó al inicio del capítulo en la figura 7 se muestra un esquema a seguir para llevar a cabo la optimización de un proceso de negocio determinado. En esta investigación se escoge como caso de estudio la

Empresa Pesquera Industrial del combinado pesquero de Batabanó creada en febrero de 1969. Esta tiene como objetivo la captura, industrialización y comercialización de forma mayorista, de especies de la plataforma frescas y congeladas; siendo la langosta su principal renglón. Es una necesidad actual para la empresa poder lograr una optimización de los procesos de producción de la langosta como clave para reducir sus costos y expandirse de una manera más rápida en su entorno empresarial.

2.5.1. Modelación del Proceso de Negocio

El proceso de negocio seleccionado para optimizar utilizando la aplicación informática, es el proceso relacionado con la Producción de la Langosta Entera Precocinada perteneciente a la empresa de Batabanó. Este describe un conjunto de actividades a tener en cuenta para lograr la producción de este tipo de especie que es la langosta. El proceso consiste en que una vez que se realice la recepción y pesaje de este tipo de especie, esta se someta a un proceso de selección y preclasificación y si cumple con las características, transita por una serie de acciones donde finalmente se procede al envasado y almacenamiento de esta para su posterior distribución y comercialización. En la figura 9 se muestra la modelación del proceso mediante la notación BPMN utilizando la herramienta Bonita Studio.

Dado que este modelo solo presenta una representación organizacional de las actividades implícitas dentro del proceso de Producción de la Langosta Entera Precocinada, al cual no se le puede realizar un procesamiento cuantitativo, surge la necesidad de trasformar este modelo BPMN del proceso hacia un modelo matemático, en este caso una red de Petri. Para ello se utiliza la herramienta Yasper que permite la importación de modelos BPMN. En esta herramienta se procede a añadir los datos cuantitativos a tener en cuenta para la optimización. En este caso se agregaron los datos que representan los costos de cada una de las actividades por medio de sus transiciones del proceso, posteriormente se exporta este en el modelo del proceso en el formato PNML, el cual es utilizado por la aplicación informática que se propone. La figura 10 muestra la modelación matemática del proceso Producción de la Langosta Entera Precocinada mediante una red de Petri.

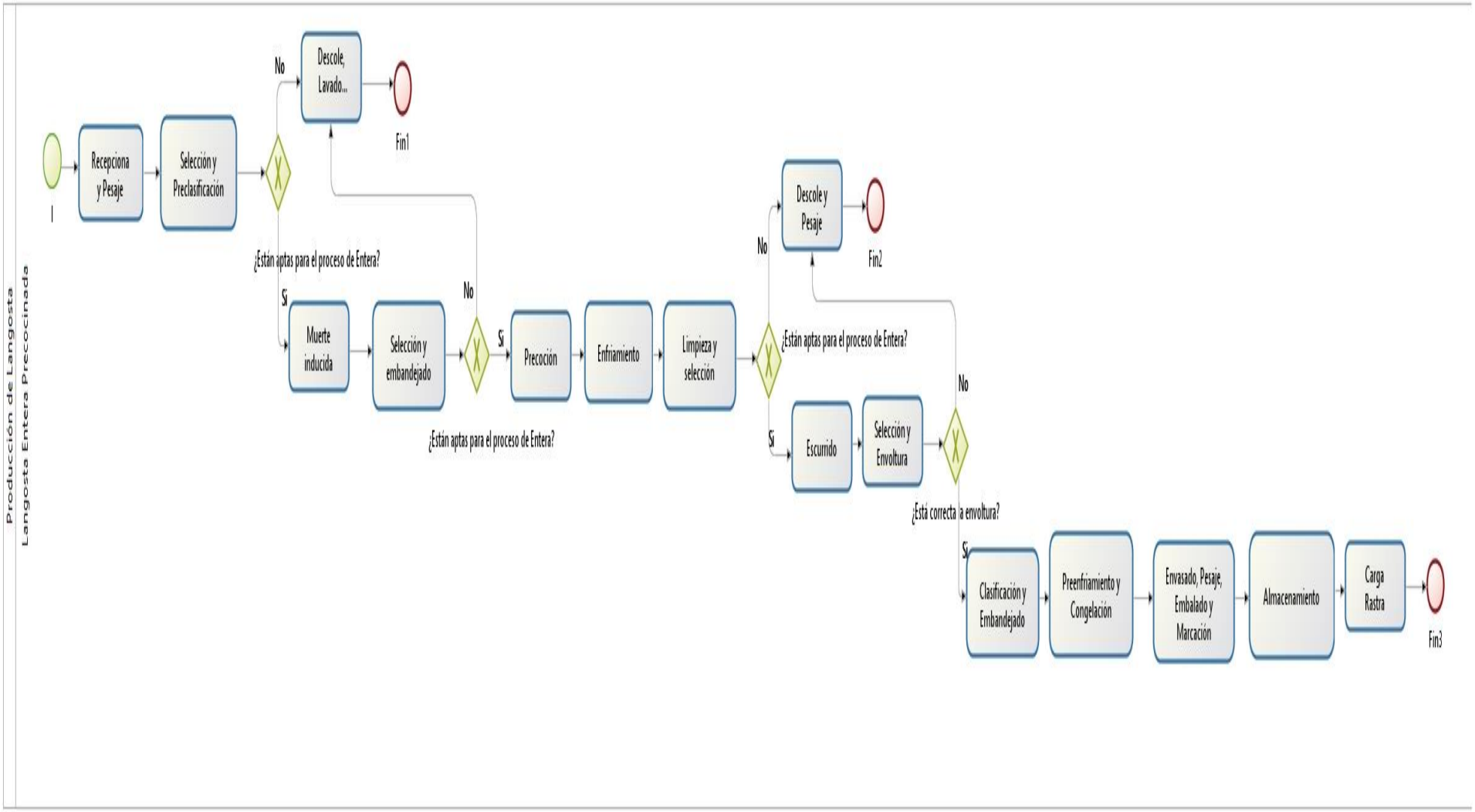


Figura 9. Modelación del proceso: Producción de la Langosta Entera Precocinada en la notación BPMN. (Fuente. Elaboración Propia).

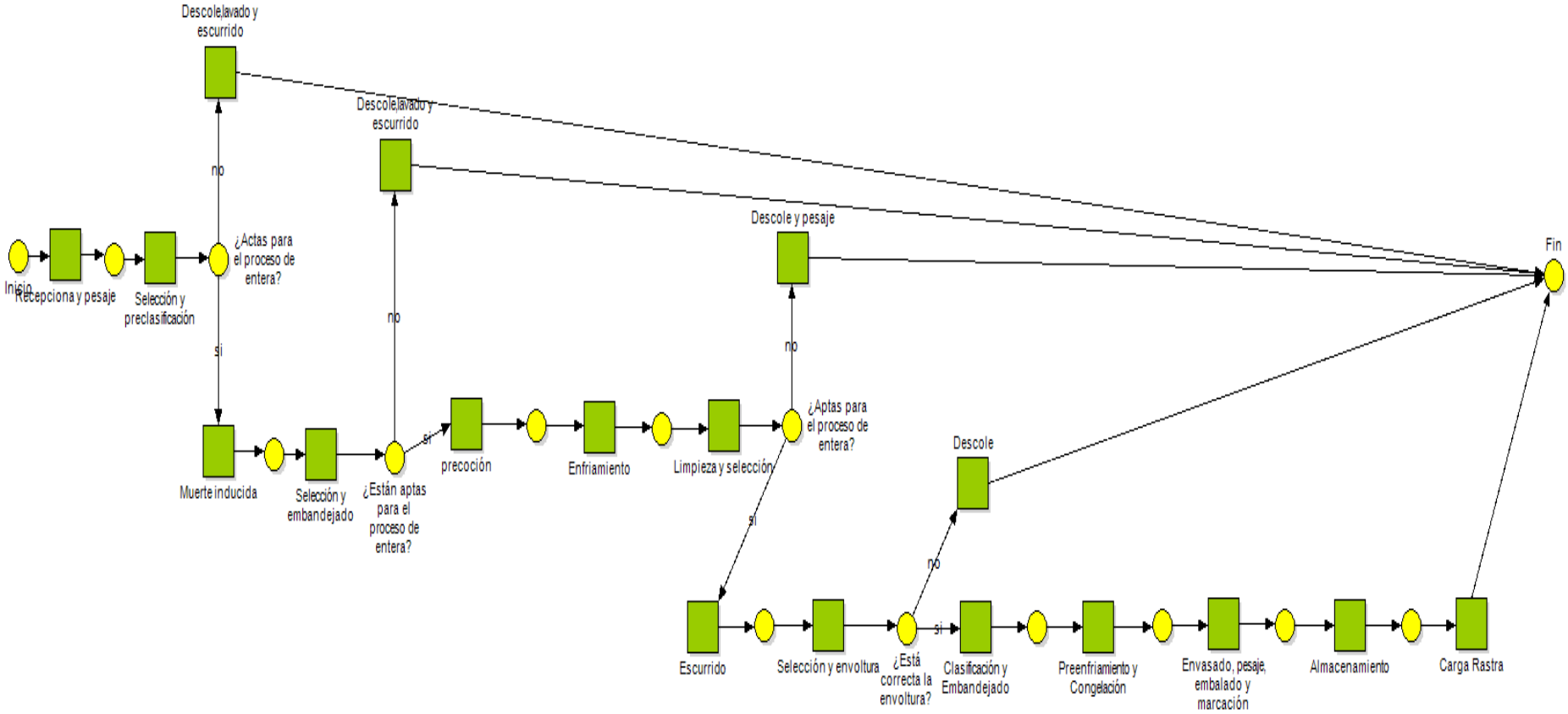


Figura 10. Modelación del proceso: Producción de la Langosta Entera Precocinada en una red de Petri.

(Fuente. Elaboración Propia).

Una vez obtenido el modelo del proceso de negocio en una red de Petri (en su archivo PNML), se necesita su correspondiente registro de eventos en su formato XES para ser procesado por la aplicación informática propuesta. Para ello se procede a generar el registro de eventos del proceso mediante la herramienta GeneradorXes utilizando el registro de eventos que posee la empresa donde se encuentran los datos relacionados con este proceso. El archivo XES generado posee las siguientes características: 1000 trazas, con 5135 eventos para un total de 17 actividades que representan al proceso. En la figura 11 se muestra una parte del registro de evento del proceso a optimizar.

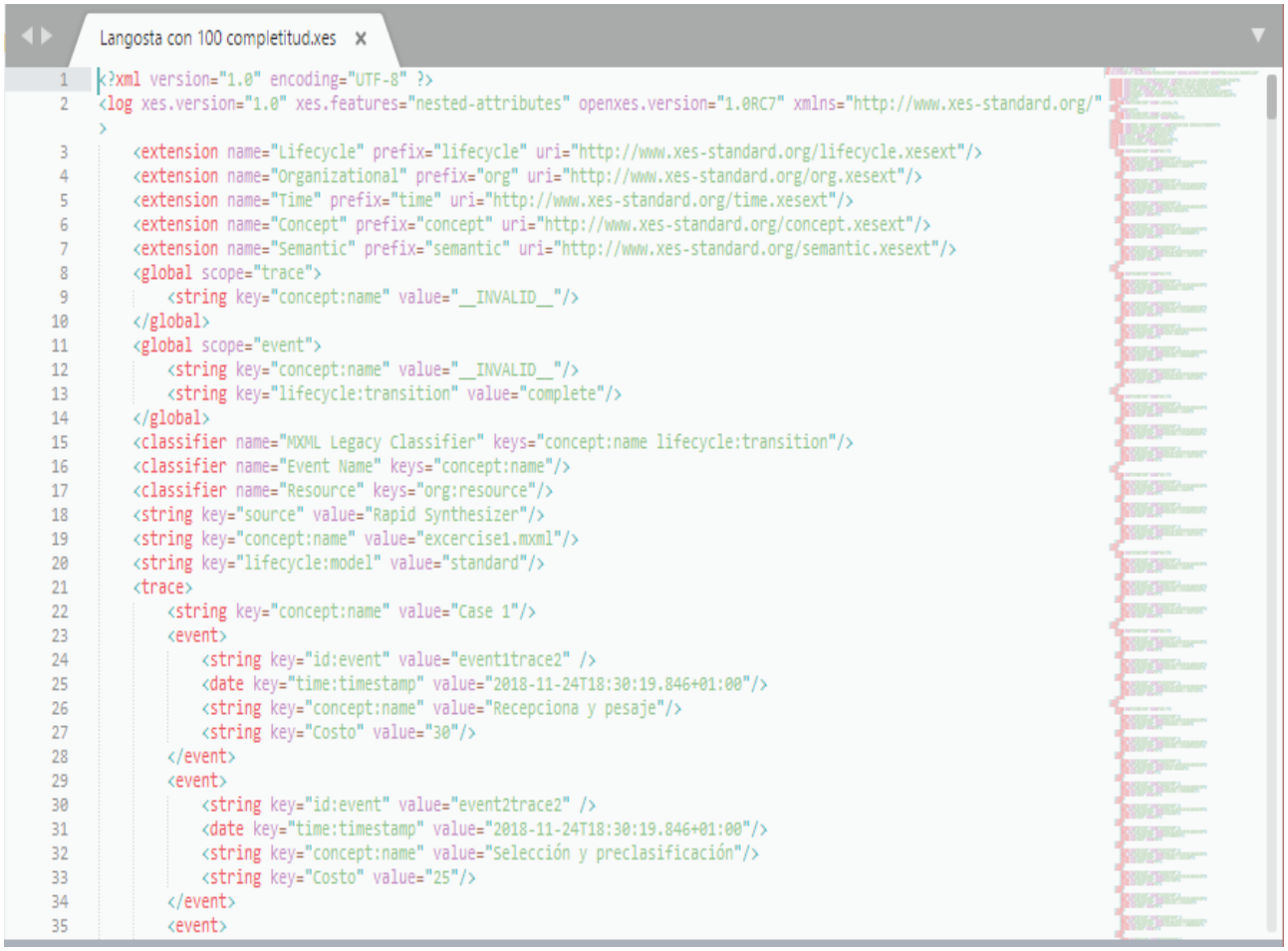


Figura 11. Registro de eventos del proceso de negocio: Producción de la Langosta Entera Precocinada. (Fuente. Elaboración Propia).

2.5.2. Aplicación de los algoritmos NSGAII o SPEA2

Una vez que se cuente con la representación del modelo del proceso en una red de Petri en formato PNML y su registro de eventos en formato XES se procede a aplicar los algoritmos NSGAII o el SPEA2. En los algoritmos genéticos y en este caso para el NSGAII y SPEA2, el primer paso es crear la población de individuos la cual comenzará a evolucionar por medio de operadores genéticos (selección, cruzamiento y mutación) que se explicarán más adelante. Esto es equivalente a decir que, mediante estos operadores, un conjunto de soluciones va a ir cambiando iterativamente hacia mejores soluciones.

2.5.2.1. Generación de la población inicial

Para la generación de la población inicial se tiene en cuenta que la primera solución en esta población inicial seria el proceso original, posteriormente para la generación de las restantes soluciones se utiliza una heurística propuesta por Medeiros (2006) y utilizada por Díaz (2014) en su investigación. Esta establece la siguiente relación: mientras más frecuente ocurra que una actividad esté directamente sucedida por otra actividad (la subtraza que aparece en el registro de eventos), es más probable que exista la relación causal. Una vez que las relaciones causales son determinadas, las funciones $\boldsymbol{I}$ y $\boldsymbol{O}$ se construyen aleatoriamente. Para ello se establece un tamaño máximo para los conjuntos $\boldsymbol{I}$ y $\boldsymbol{O}$ de cada actividad que no excederá el número de actividades presentes en el proceso. Cada actividad a_i que sucede causalmente a una actividad a_2 es insertada aleatoriamente en uno o más subconjuntos de $\boldsymbol{O}$ (a_2).

La heurística construye la población inicial basada en un índice de dependencia que toma en consideración información local almacenada en el registro de eventos del proceso. Este índice de dependencia indica cuan fuertemente una tarea depende de otra, este se diseñó sobre una matriz (D) de tamaño nxn donde n es la cantidad de actividades del proceso modelado. Así, el valor correspondiente a la posición $D(i,j)$ indica cuan probable es que exista la subtraza "$a_i a_j$" en el registro de eventos. Para la construcción del índice de dependencia se recorren cada una de las trazas presentes en el registro de eventos y se construyen las funciones:

1. $\boldsymbol{bucle2}: \boldsymbol{TxTxL} \rightarrow \mathbb{N}$ **:** detecta cuántas veces una tarea está involucrada en un ciclo de longitud 2. Es decir, $bucle2(t_1, t_2, L)$ devuelve la cantidad de veces que la subcadena "$t_1 t_2 t_1$" aparece en el registro de eventos.

2. $\boldsymbol{follow}: \boldsymbol{TxTxL} \rightarrow \mathbb{N}$ **:** detecta cuántas veces una tarea está sucedida directamente por otra. Es decir $follow(t_1, t_2, L)$ devuelve cuántas veces la subcadena "$t_1 t_2$" aparece en el registro de eventos.

El índice de dependencia es una función $D: TxTxL \rightarrow \mathbb{R}$ donde T es el conjunto de tareas y L el registro de eventos. Esta función se calcula de la siguiente manera:

$$D(t_1, t_2, L) = \begin{cases} \dfrac{bucle2(t_1, t_2, L) + bucle(t_2, t_1, L)}{bucle2(t_1, t_2, L) + bucle(t_2, t_1, L) + 1} & Si\ t_1 \neq t_2\ y\ bucle2(t_1, t_2, L) > 0 \\ \dfrac{follow(t_1, t_2, L) - follow(t_2, t_1, L)}{follow(t_1, t_2, L) + follow(t_2, t_1, L) + 1} & Si\ t_1 \neq t_2\ y\ bucle2(t_1, t_2, L) = 0 \\ \dfrac{follow(t_1, t_2, L)}{follow(t_1, t_2, L) + 1} & Si\ t_1 = t_2 \end{cases}$$

Esta función fue implementada en la aplicación propuesta como una matriz (un arreglo bidimensional) la cual queda reflejada en la figura 12, para que la acción de pedir $D(t_1, t_2, L)$ tenga un costo de $O(1)$. Posteriormente se puede generar la población inicial cuyo tamaño depende del tamaño que defina el usuario. Para ello se comienza construyendo la función de entrada o de salida de la matriz causal, se buscan todas las combinaciones posibles de causalidad entre las actividades y por cada una se establece un valor aleatorio r: si $r \leq D(t_1, t_2, L)$ entonces se procede a poner en la función de salida O de t_1 (función elegida en la herramienta propuesta) en cualquier subconjunto la actividad t_2. Solamente restaría arreglar la función de entrada I, ya que si t_2 se enlaza por la salida de t_1 entonces hay que añadir que t_1 se enlaza por la entrada de t_2.

```
0.0 0.999000999000999 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
-0.999000999000999 0.0 0.9981132075471698 0.9978813559322034 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 -0.9981132075471698 0.0 0.0 -0.9956709956709957 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 -0.9978813559322034 0.0 0.0 0.9978813559322034 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.9956709956709957 -0.9978813559322034 0.0 0.9958677685950413 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 -0.9958677685950413 0.0 0.9958677685950413 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 -0.9958677685950413 0.0 0.9958677685950413 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 -0.9958677685950413 0.0 0.9921259842519685 0.9913793103448276 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 -0.9921259842519685 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 -0.9913793103448276 0.0 0.0 0.9913793103448276 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 -0.9913793103448276 0.0 0.9836065573770492 0.0 0.0 0.0 0.0 0.0 0.9821428571428571
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 -0.9836065573770492 0.0 0.9836065573770492 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 -0.9836065573770492 0.0 0.9836065573770492 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 -0.9836065573770492 0.0 0.9836065573770492 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 -0.9836065573770492 0.0 0.9836065573770492 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 -0.9836065573770492 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 -0.9821428571428571 0.0 0.0 0.0 0.0 0.0 0.0 0.0
```

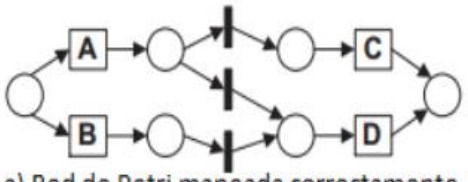
Figura 12. Matriz que representa el índice de dependencia para el proceso. Fuente (Elaboración Propia).

Por cada individuo que está representado en una matriz causal hay que calcular el valor de la funciones objetivos para esa solución mediante el parseo de trazas, por lo que se hace necesario transformar las matrices causales en redes de Petri, en este caso en BP-net.

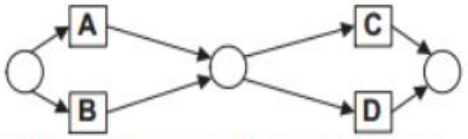
2.5.2.1.1. Transformación de Matrices Causales en BP-net

Para el proceso de transformar una matriz causal a una red de Petri se necesita inferir lugares por cada actividad. Por lo que es necesario extender la red de Petri, de no hacerlo se puede incurrir en errores de causalidad, en la figura 13 se muestra el mapeo de una matriz causal a una red de Petri. Debido a que las matrices causales son más expresivas que la propia red de Petri, Medeiros (2006) propone que se resuelva añadiendo transiciones ocultas, que al final no se cuentan a la hora de evaluar sobre las trazas, pero mantiene la causalidad, tanto de entrada como de salida entre las actividades del proceso.

Actividad	Entrada	Salida
A	$\{\}$	$\{\{C, D\}\}$
B	$\{\}$	$\{\{D\}\}$
C	$\{\{A\}\}$	$\{\}$
D	$\{\{A, B\}\}$	$\{\}$



a) Red de Petri mapeada correctamente



b) Mapeo incorrecto de la red de Petri

Figura 13. Mapeo en una red de Petri. Fuente (Medeiros, 2006).

A continuación, se describen brevemente los pasos a seguir para obtener una red de Petri mapeada partiendo de una matriz causal propuesta por Medeiros (2006):

- Se crean tantas transiciones como tareas existan en la matriz causal (una transición por cada actividad).
- Se crea un lugar inicial i y un lugar final o.
- Por cada transición (actividad) cuya $I(t) \neq \emptyset$ se crea un lugar de entrada i_t por cada subconjunto en I.
- Por cada transición (actividad) cuya $O(t) \neq \emptyset$ se crea un lugar de salida o_t por cada subconjunto en O.
- Se agrega un arco entre i y todas las transiciones cuya $I(t) \neq \emptyset$.
- Se agrega un arco entre todas las transiciones cuya $O(t) \neq \emptyset$ y O.
- Se agrega un arco entre cada transición y su lugar de salida en caso de existir.
- Se agrega un arco entre cada lugar de entrada (en caso de existir) y su respectiva transición.
- Por cada relación causal presente en la matriz causal se crea una transición invisible $m_{t_1 t_2}$ (las relaciones causales se detectan recorriendo la columna correspondiente a I u O).
- Si entre dos tareas t_1, t_2 existe una relación causal $(t_2 \in O(t_1)\ o\ t_1 \in I(t_2))$ se agregan dos arcos, el primero de o_{t_1} a $m_{t_1 t_2}$ y el segundo de $m_{t_1 t_2}$ a i_{t_2} donde $m_{t_1 t_2}$ es la transición invisible creada para una relación causal entre t_1 y t_2.

Una vez transformada la matriz causal en red de Petri se procede a realizar la evaluación de la misma mediante el parseo de las trazas, aspecto que se detalla a continuación.

2.5.2.2. Evaluación de cada individuo

Una vez que se transforma la matriz causal que representa la solución obtenida después de haber generado la población inicial en una BP-net se procede a evaluarla mediante el parseo de las trazas sobre la BP-net. Según Díaz (2014), una traza es parseada correctamente por un individuo, si para una marca inicial con un solo token ubicado en el lugar inicial de la BP-net después de ser ejecutadas las tareas visibles en el orden en que aparecen en la traza, el lugar final es el único marcado y con un solo token. En la figura 14 se ilustra el mecanismo de disparo en una traza perteneciente a un registro de eventos. Este se corresponderá con una traza formada por las actividades A, B, C y D almacenadas en ese orden.

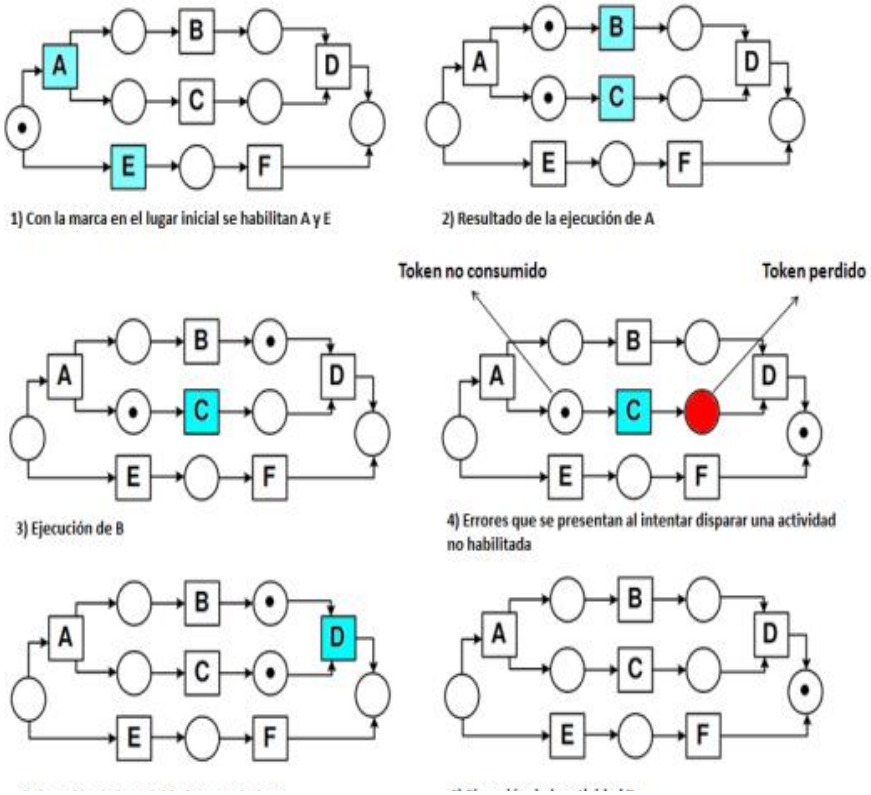


Figura 14. Mecanismo de disparo en el parseo de una traza. Fuente (Díaz, 2014).

La evaluación se realiza para ambos algoritmos de igual manera, después de haber parseado las trazas correspondientes al proceso con respecto a la BP-net se obtienen un conjunto de datos, en este caso, el número de actividades ejecutadas correctamente, el número de tokens perdidos, el número de tokens que quedaron en la red cuando finalizó la ejecución de una traza, el total de trazas en el registro de eventos, el número de trazas

donde se perdieron tokens, el número de trazas donde quedaron tokens y el total de actividades en el registro de eventos. Con estos datos se procede a calcular el valor de la función del costo y la completitud utilizando las fórmulas anteriormente mencionadas. Finalmente, sobre esta población se le aplican los operadores genéticos (selección, cruzamiento y mutación) los cuales son detallados a continuación.

2.5.2.3. Operadores genéticos

En estos algoritmos evolutivos se debe elegir un criterio de parada para controlar que el ciclo generacional finalice y se obtenga el conjunto de soluciones finales. Existen varios criterios de parada de los algoritmos que se tienen en cuenta para la finalización de estos, dentro de ellos se pueden mencionar la cantidad de evaluaciones, el número de generaciones y la cantidad de generaciones máxima sin que cambien los individuos del frente de Pareto. En la presente investigación se toma como criterio de parada de los algoritmos el número de evaluaciones, que a su vez es el que propone el *framework* JMetal por González, et al. (2011) y el mismo corresponde con el problema abordado.

2.5.2.3.1. Operador Genético: Selección

El operador de selección consiste en elegir a los individuos que serán cruzados y mutados en la siguiente generación. Este operador funciona de diferentes formas en ambos algoritmos. La diferencia entre SPEA2 y NSGAII en este operador radica en cómo cada uno de los algoritmos preservan el elitismo y como seleccionan los individuos más aptos.

En el caso del NSGAII el operador de selección se aplica en dos momentos:

Primeramente, en una nueva generación para seleccionar los individuos necesarios para cruzar y luego mutar. Sobre la población son seleccionadas N parejas de soluciones escogidas aleatoriamente. Cada pareja compite en un torneo en este caso mediante el torneo binario, donde gana la alternativa que pertenezca al rango de mejor calidad. Si las alternativas en competencia pertenecen al mismo frente, entonces gana la que introduzca un mayor grado de diversidad al conjunto en construcción. Para ello es necesario tener en cuenta que el rango se calcula según el criterio de no dominancia explicado en el algoritmo en el capítulo anterior, mientras menor sea el frente donde está una solución, mejor será el rango.

En un segundo momento, se procede a aplicar este operador una vez que se haya aplicado ya el cruzamiento y la mutación (operadores que se detallan en las siguientes secciones) a las soluciones antes obtenidas. Posteriormente se obtiene una nueva población que consiste en la unión de padres y nuevos descendientes. De esta nueva población se extraen los diferentes frentes (agrupados según el número de soluciones que los dominan). El frente F_1 coincide con el actual frente Pareto. Pero la siguiente generación tiene que ser de tamaño N (según los parámetros de entrada del algoritmo) por lo que se seleccionan los frentes de mejor a peor, hasta alcanzar el tamaño máximo. Si es necesario, el último frente se trunca atendiendo al orden basado en el *crowding* (medida de concentración), que permite seleccionar las soluciones más dispersas. Cuanto mayor sea la distancia de *crowding* de una solución al resto de su frente, mejor, ya que hay menos concentración en esa zona.

Al igual que en el NSGAII en el SPEA2 el operador de selección se aplica en dos momentos:

Primeramente, para seleccionar las parejas de soluciones que se recombinarán (cruzarán) en una generación determinada del archivo. Luego de asignar el *fitness* a cada individuo (solución) se seleccionan las parejas por torneo binario, como en NSGAII pero por este valor de *fitness* $F(i)$.

Y en un segundo momento al determinar las soluciones que se pondrán en el archivo de la siguiente generación. En el archivo van las soluciones no dominadas por ninguna otra solución perteneciente al archivo y a la población, las cuales tienen menor *fitness*. Si al determinar el siguiente archivo el tamaño queda por debajo de su tamaño real (pasado como parámetro del algoritmo) entonces se seleccionan para agregar al nuevo archivo las mejores soluciones dominadas en el archivo previo y la población. Pero si el nuevo archivo es más grande que su tamaño real, se aplica un criterio de truncamiento que consiste en ordenar el archivo ascendentemente por valor de distancia $D(i)$.

Para el cálculo del *fitness* se tiene en cuenta que a cada individuo i en el archivo A_t y en la población P_t se le asigna un valor de fuerza $S(i)$ representando el número de soluciones que el domina:

$$S(i) = |\{j | j \in P_t + A_t \wedge i \prec j\}|$$

Donde || denota la cardinalidad del conjunto, + como la unión de dos conjuntos y $\prec$ corresponde a la relación de dominancia Pareto. Basándose en el valor de fuerza $S(i)$ se calcula el *fitness* crudo $R(i)$ de un individuo i:

$$R(i) = \sum_{j \in P_t + A_t, j \prec i} S(j)$$

La densidad $D(i)$ correspondiente a i está definida por

$$D(i) = \frac{1}{\sigma_i^k + 2}$$

Donde σ_i^k es la distancia (en el espacio objetivo) del elemento i con respecto a un elemento k en P_t y en A_t después de ordenar la lista en orden creciente. Finalmente, el cálculo de *fitness* quedaría:

$$F(i) = R(i) + D(i)$$

2.5.2.3.2. Operador Genético: Cruzamiento

El operador genético de cruzamiento consiste en que en cada generación se seleccionen por torneo binario los individuos más capaces y se sometan a la recombinación o cruzamiento para producir dos nuevos individuos con características de ambos padres. A continuación, se muestra pseudocódigo del operador cruzamiento:

Entrada: Dos padres (*parent1* y *parent2*), un índice de cruzamiento.

Salida: Dos posibles descendientes recombinados (*offspring1* y *offspring2*).

Con probabilidad establecida por el índice de cruzamiento hacer:

a) Seleccionar aleatoriamente una actividad a como punto de cruzamiento de los descendientes.

b) Seleccionar aleatoriamente un punto de intercambio $sp1$ para $I_1(a)$(para el *offspring1*). El punto de intercambio está entre 0 y n − 1, donde n es el número de subconjuntos en la función $I_1(a)$.

c) Seleccionar aleatoriamente un punto de intercambio $sp2$ para $I_2(a)$ para el *offspring2*).

d) Construir el subconjunto $remainingSet1(a)$ formado por los subconjuntos en $I_1(a)$ que se encuentran entre las posiciones $[0\ y\ sp_1)$.

e) Construir el conjunto $swapSet_1(a)$formado por los subconjuntos en $I_1(a)$ cuyas posiciones sean mayores o iguales que $sp1$.

f) Repetir los pasos d), e) para construir los conjuntos $remainingSet2(a)$ y $swapSet_2(a)$ a partir de sp_2 y $I_2(a)$.

g) Para cada subconjunto S_2 en $swapSet_2(a)$ adicionar S_2 como nuevo subconjunto en $remainingSet1(a)$.

h) Repetir el paso g) pero utilizando S_1, $swapSet_2(a)$ y $remainingSet1(a)$.

i) Repetir los pasos b) hasta h) usando ahora $O(a)$.

j) Retornar *offspring1* y *offspring2*.

2.5.2.3.3. Operador Genético: Mutación

Luego de aplicar el operador de cruzamiento y según el valor aleatorio del índice de mutación, puede que se muten ambos descendientes nuevos: *offspring1* y *offspring2*. El pseudocódigo de este operador es el siguiente:

Entrada: Un individuo, un índice de mutación

Salida: Un individuo posiblemente mutado

1. Por cada actividad en el individuo hacer:

a) Según la probabilidad dada por el índice de mutación, realizar alguna delas siguientes operaciones para la función $I(a)$:

i. Seleccionar un subconjunto *X* en $I(a)$ y adicionar una actividad *a'* en *X*, donde *a'* pertenece al conjunto de actividades del individuo.

ii. Seleccionar un subconjunto *X* en $I(a)$ y eliminar una actividad *a'* de *X*, donde $a' \epsilon x$. Si después de extraer *a'*, *X* se queda vacío entonces se eliminará *X* de $I(a)$.

iii. Redistribuir los elementos en $I(a)$:

A. Obtener una lista I con todos los elementos en los subconjuntos de $I(a)$:

B. Eliminar todos los subconjuntos de $I(a)$.

C. Crear n conjuntos $(1 \leq n \leq |I|)$ y distribuir aleatoriamente los elementos de *I* en los *n* conjuntos.

D. Insertar los *n* conjuntos en $I(a)$.

b) Repetir el paso a) para la función $0(a)$.

Una vez aplicado los operadores genéticos y definido el criterio de parada de los algoritmos se procede a obtener el conjunto de soluciones óptimas del proceso para cada uno de los algoritmos previstos en el desarrollo de la aplicación informática. Posteriormente es el analista del negocio (usuario) el decisor final de cuál de las soluciones obtenidas desea emplear para la mejora del desempeño del proceso que se esté analizando.

2.6. Conclusiones

En el presente capítulo se especificaron las tecnologías y herramientas a utilizar para el desarrollo de la aplicación informática. Java como lenguaje de programación, Netbeans como entorno de desarrollo, JMetal como *framework* para el trabajo con los algoritmos NSGAII y SPEA2 y las herramientas Bonita Studio y Yasper para el modelado del proceso seleccionado. Se formuló el problema multiobjetivo, identificando dos funciones objetivos (costo y completitud) a tener en cuenta para la optimización. Se especificó el caso de estudio referente al proceso de la Producción de la Langosta Entere Precocinada de la empresa pesquera de Batabanó. Se realizó la modelación del proceso en la notación BPMN y red de Petri; y se generó el XES asociado al proceso para su posterior optimización mediante los algoritmos utilizados.

Se explicaron cada uno de los pasos a tener en cuenta por los algoritmos en dicha optimización, profundizándose en cada una de las características propias de ellos y por tanto esto permite evaluar los conceptos claves en el diseño de los algoritmos evolutivos como son el elitismo, la diversidad genética y la posibilidad de distribuir las soluciones halladas en todo el frente de Pareto. A continuación, en el capítulo 3 se presenta la aplicación informática desarrollada y se muestran los resultados arrojados por los algoritmos NSGAII y SPEA2 en la optimización del proceso del caso de estudio seleccionado.

CAPÍTULO 3. RESULTADOS Y DISCUSIÓN

Utilizando las herramientas y tecnologías mencionadas en el capítulo anterior, se desarrolló una aplicación informática de optimización de procesos de negocio atendiendo a dos funciones objetivos: el costo y completitud de proceso, para ello se utilizan dos algoritmos multiobjetivos el NSGAII y SPEA2. La aplicación desarrollada permite optimizar un modelo de un proceso de negocio, en este caso, el de Producción de la Langosta Entera Precocinada descrito en el capítulo anterior. Esta aplicación tiene como entrada la representación del proceso en una red de Petri (en su formato PNML) y el registro de eventos (en formato XES), como resultado se obtienen varias propuestas del diseño optimizado del proceso seleccionado.

Además, en este capítulo se muestran los resultados obtenidos al aplicarle los algoritmos de optimización NSGAII o el SPEA2 al proceso de Producción de la Langosta Entera Precocinada, con el objetivo de obtener las posibles soluciones óptimas para la mejora del proceso en su desempeño. En aras de igualar las condiciones de ejecución de ambos algoritmos se tuvieron en cuenta los mismos parámetros de entrada como son: el tamaño de la población inicial (10), el número de evaluaciones (1000), el factor de cruzamiento (90%) y el factor de mutación (50%). La única diferencia entre estos algoritmos es que el algoritmo SPEA2 utiliza un tamaño de archivo para guardar las soluciones más prometedoras, en este caso se le dio valor de (10). Para la validación de los resultados se utilizaron las métricas de generación de vectores no dominados, la razón de generación de vectores no dominados, la generación real de vectores no dominados y la distancia generacional que medirán el desempeño de estos y así identificar cual algoritmo es el más eficiente en proponer sus soluciones óptimas del proceso que se está analizando.

3.1. Aplicación informática implementada

En esta sección se presenta la aplicación informática implementada la cual consiste en optimizar procesos de negocio en cuanto al costo y completitud del proceso. Estos procesos tienen que estar representados en una red de Petri y a partir de su representación en el archivo PNML del mismo y su registro de eventos en su archivo XES, se procede a cargar el proceso y a configurar los parámetros de entrada a considerar para cada uno de los algoritmos a emplear para efectuar la optimización.

En la figura 15 se muestra la ventana principal de la herramienta propuesta, en la cual se da la opción de cargar el proceso en su archivo PNML y XES.

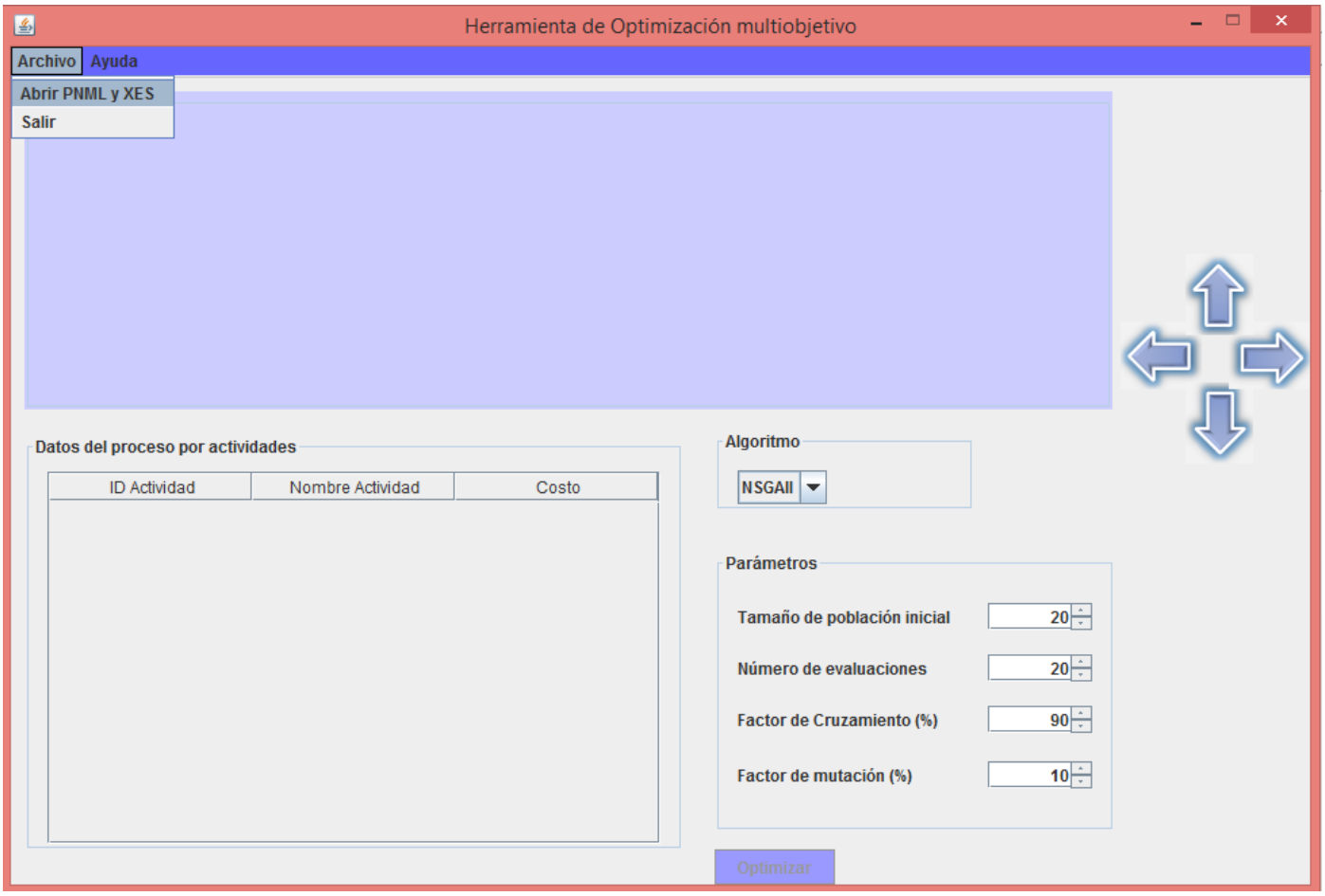


Figura 15. Interfaz Principal de la herramienta Optimización de Procesos de Negocio.

Posteriormente en la figura 16 se muestra la carga el modelo del proceso de negocio: Producción de la Langosta Entera Precocinada en sus archivos PNML y XES para su procesamiento.

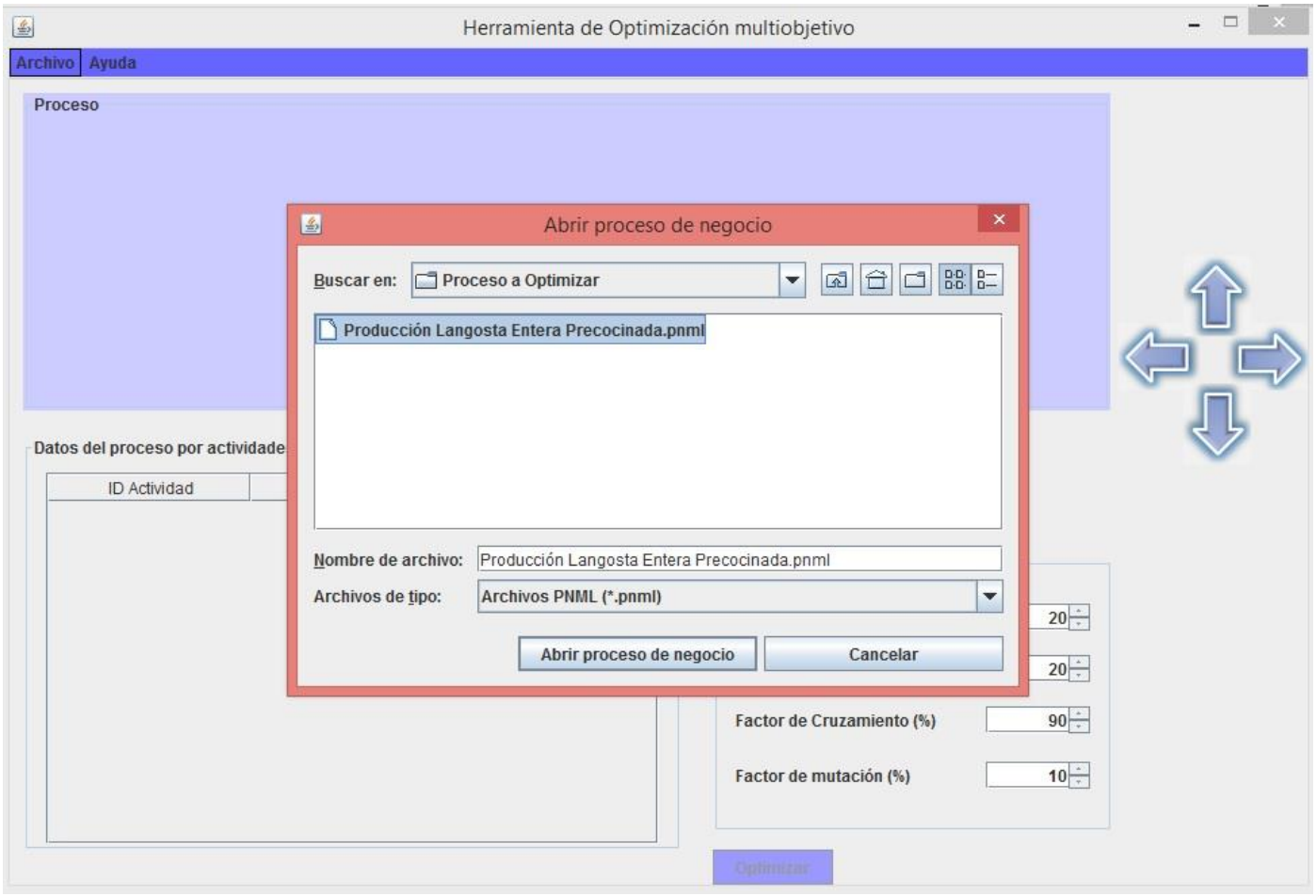


Figura 16**. Carga del proceso de negocio: Producción de la Langosta Entera Precocinada.**

Una vez importado el proceso en la herramienta propuesta, se procede a configurar los parámetros de entrada de los algoritmos a emplear en este caso el NSGAII y el SPEA2. Los parámetros de entrada para ambos algoritmos son: el tamaño de la población inicial (10), el número de evaluaciones (1000), el factor de cruzamiento (90%), el factor de mutación (50%) y en el caso del algoritmo SPEA2 se requiere de un nuevo parámetro que es el tamaño del archivo (10). Esto se ve reflejado en la figura 17 para el algoritmo NSGAII y en la figura 18 para el algoritmo SPEA2.

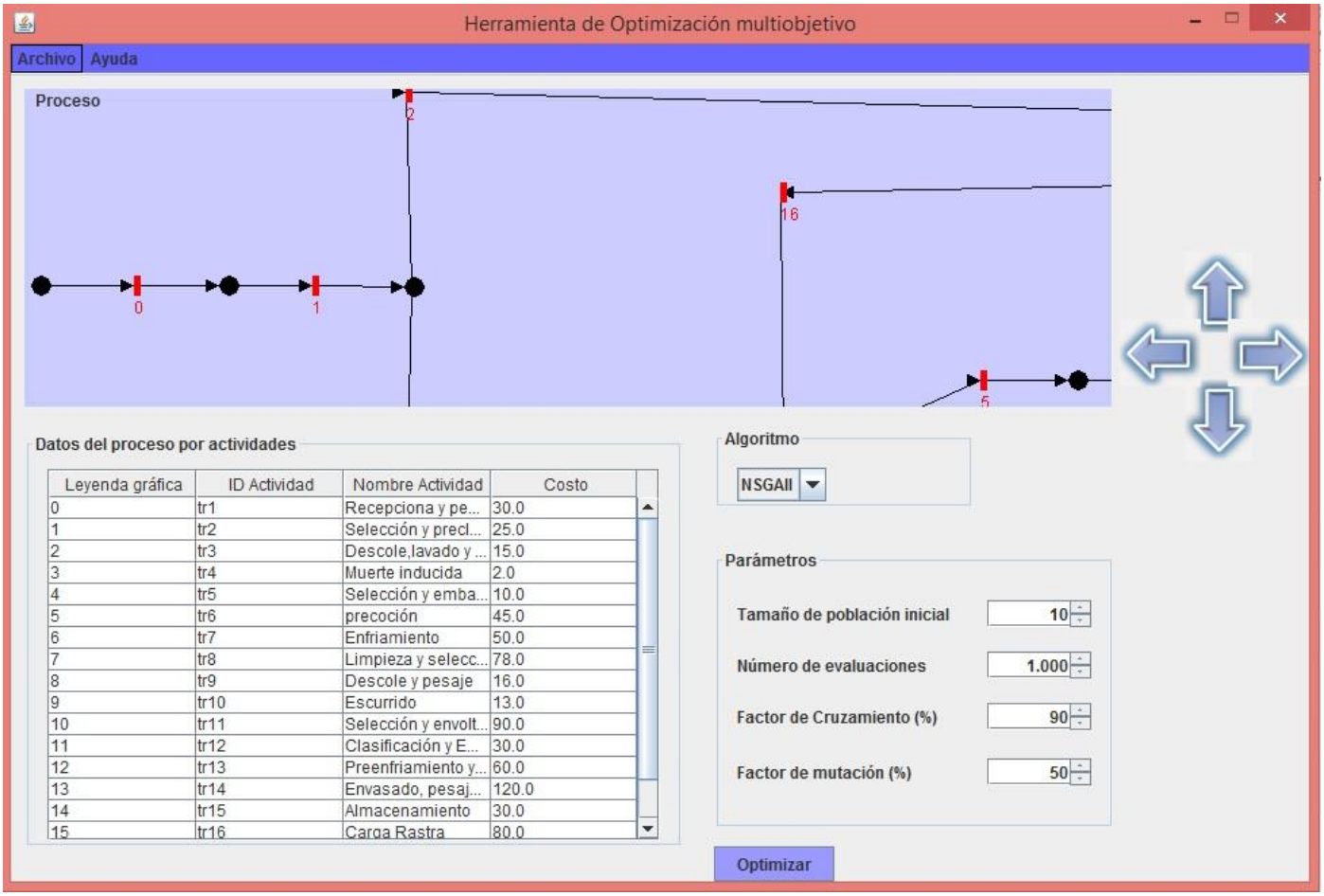


Figura 17. Configuración de los parámetros de entrada para el algoritmo NSGAII.

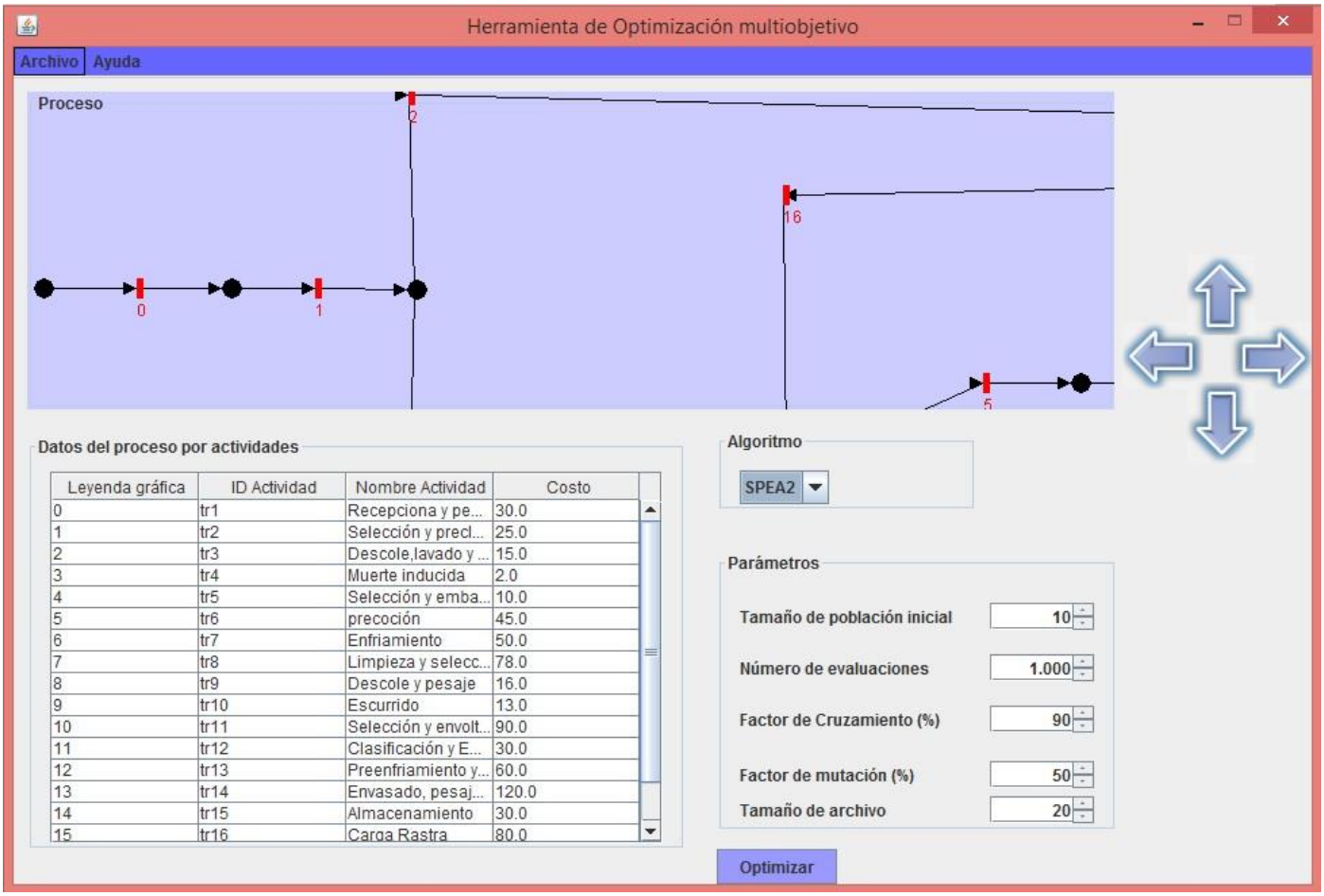


Figura 18. Configuración de los parámetros de entrada para el algoritmo SPEA2.

Una vez configurado los parámetros de entrada se procede a realizar la opción de Optimizar y posteriormente en la figura 19 se muestra el conjunto de soluciones óptimas del proceso optimizado según el algoritmo seleccionado. Del conjunto de las soluciones óptimas propuestas se puede seleccionar la que desea visualizar y se muestra el diseño de proceso optimizado según la solución seleccionada y se le muestran al usuario la leyenda de las actividades eliminadas para ese diseño. Luego el analista del negocio (usuario) es el encargado de escoger cual es la mejor para aplicar en el desempeño del proceso de negocio.

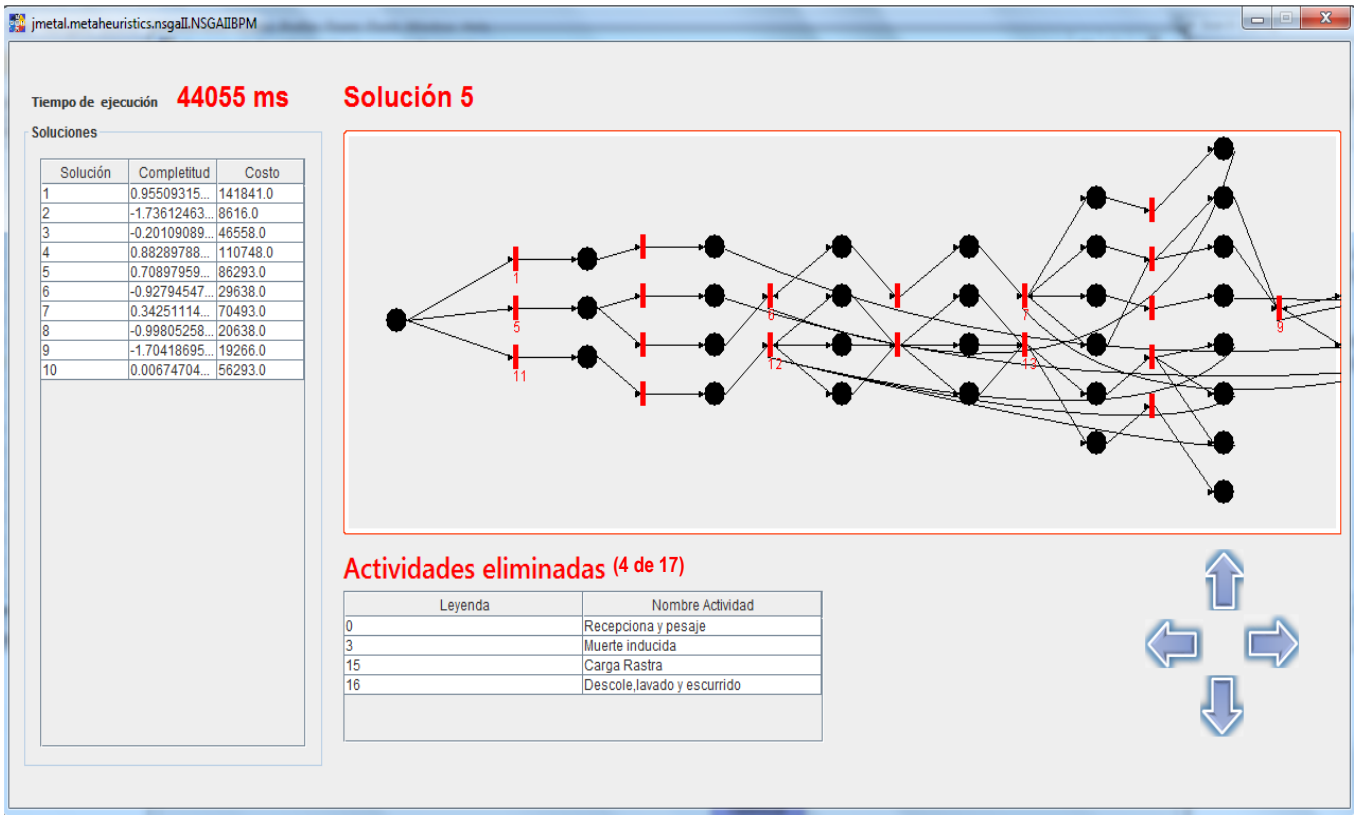


Figura 19. Resultado del proceso optimizado.

3.2. Resultados experimentales

Luego de ejecutar ambos algoritmos en 5 ejecuciones cada uno, se obtuvieron los resultados que se exponen en las tablas 3 y 4. Estos resultados se emplearan para establecer la comparación entre estos algoritmos y así poder identificar el más eficiente para el problema que se está abordando en el trabajo.

Tabla 3. Resultados obtenidos con el algoritmo NSGAII para 5 ejecuciones.

Número de corrida	Soluciones (Completitud, Costo)	Tiempo (en milisegundos)
1	0.8512761945482271, 135791.0 -1.679649464459591, 942.0 0.7312277929856148, 98373.0	

	0.21383439584060557, 82656.0 -0.7072871931532825, 10372.0 0.13136626864141854, 59733.0 -0.4300222299792398, 21791.0 -1.4788704965920156, 7687.0 -0.2000689284067032, 37773.0 -0.18835366166145645, 44973.0	25767
2	0.9184456516168708, 116228.0 -4.0017526777020445, 7266.0 -2.751898734177215, 12451.0 -1.2886075949367088, 12918.0 -0.5004699941408802, 19528.0 -0.3696647131230365, 48876.0 0.7076838948011626, 80223.0 0.12273886451101643, 67273.0 0.7192935096111596, 90188.0 0.7665895286148451, 111147.0	38939
3	0.8512761945482271, 135791.0 -0.7362966147776274, 8143.0 0.4443708190543634, 107242.0 0.2820130169630503, 82317.0 0.10055117101419571, 56138.0	

	-0.20080931726501344, 47188.0 0.14335507339567527, 69888.0 -0.44200187997656354, 32053.0 -0.23576170664778257, 39988.0 -0.4877080725181991, 19528.0	24997
4	0.89581397826946, 139241.0 -1.8519961051606622, 942.0 0.8149900163316376, 102067.0 0.6973645242794112, 70772.0 0.11953531700367145, 61072.0 -0.9933787731256085, 30917.0 -0.46488597345359506, 34745.0 -1.069328140214216, 13972.0 -1.688218111002921, 12327.0 -0.13514586679143642, 44312.0	17115
5	**0.8052580331061344, 147176.0** -1.963777994157741, 8208.0 0.7918340651466995, 90238.0 0.6413306271871002, 71678.0 0.03407653726736599, 64038.0 -1.4852969814995132, 8545.0 -1.0629016553067185, 17403.0	25116

	-0.5879293049916409,	37645.0	
	-0.6674079109331079,	18478.0	
	-0.3486472782054344,	51430.0	

Tabla 4. Resultados obtenidos con el algoritmo SPEA2 para 5 ejecuciones.

Número de corrida	Soluciones (Completitud, Costo)		Tiempo (en milisegundos)
1	0.04439189502480637,	52603.0	
	-0.229007279640191,	37712.0	
	-0.9139063468177392,	1327.0	
	-0.4770832542984442,	26327.0	30389
	0.755947897337914,	108106.0	
	-0.6184912134279222,	16480.0	
	0.04974519151734339,	67822.0	
	0.6971005493193216,	84125.0	
	0.7342749536109937,	97756.0	
	0.8052580331061344,	**147176.0**	
2	-0.7709834469328141,	24607.0	
	-1.6229795520934762,	2437.0	
	0.6281008452033429,	67557.0	
	-0.8648490749756572,	12557.0	
	0.8203978144884208,	94367.0	

	0.03849568153365621, 48837.0 0.7507762013691395, 76022.0 -0.12759055290700855, 34398.0 0.1886484823193684, 58133.0 0.89581397826946, 139241.0	26650
3	-0.5425680315686978, 35095.0 -1.6311587147030184, 22588.0 -1.8558909444985394, 11287.0 -2.79396299902629, 1880.0 0.05481944216121435, 60143.0 -0.22375851996105162, 47267.0 0.7092396940984675, 73682.0 0.8123305653028605, 130175.0 0.9430551801722363, 140126.0 0.8029979239771453, 97583.0	36227
4	-1.421811100292113, 10893.0 0.570395851241891, 69680.0 -0.599463173559185, 34800.0 -0.016790344898272948, 55391.0 -0.355126306700226, 44188.0 -0.8122687439143136, 24388.0 0.6887325237456596, 83728.0	19238

	0.7673515721014244, 95138.0 0.7912972387058845, 106117.0 **<u>0.8052580331061344, 147176.0</u>**	
5	-0.7573820065771343, 15065.0 0.9313340191653112, 131726.0 -0.301129135970311, 22283.0 -1.5037974683544304, 385.0 -0.187837813010968, 50827.0 0.8010605750012811, 91838.0 0.15552324091631223, 78203.0 -0.2251436327385695, 38188.0 0.8275553543845511, 123791.0 0.043264316290349256, 64541.0	39574

En las tablas 3 y 4 se muestran los datos recopilados en 5 ejecuciones de cada uno de los algoritmos, la solución que está subrayada significa que esta coincide con el proceso de entrada (0.8052580331061344, 147176.0), es decir, que corresponde al proceso original cuyos valores se obtuvieron en segundo plano en la aplicación propuesta. Aunque el algoritmo SPEA2 mostró que en dos de sus ejecuciones se obtuvo como solución óptima el proceso original y el algoritmo NSGAII solo en una de sus ejecuciones, no se puede arribar a las conclusiones, que uno encuentra mejor que el otro el conjunto de soluciones óptimas puesto que ambos algoritmos manejan mucha la aleatoriedad de sus operadores y además solamente se realizaron 5 ejecuciones de estos para una población inicia de 10.

Existen varias investigaciones acerca de cómo comparar algoritmos multiobjetivo dentro de estas se encuentra la investigación de Duarte, (2001), la cual fue elegida como referencia para seleccionar las métricas a emplear en la comparación de ambos algoritmos, ya que las métricas que la autora menciona en su investigación son un compendio de varios autores.

3.3. Comparación de los algoritmos NSGAII y SPEA2

En esta sección se establece una comparación entre los algoritmos empleados en el desarrollo de la investigación atendiendo a 4 métricas que medirán el desempeño de estos y así poder identificar cual algoritmo es el más eficiente en proponer sus soluciones óptimas del proceso que se está analizando. Las métricas a tener en cuenta para establecer la comparación son: la generación de vectores no dominados, la razón de generación de vectores no dominados, la generación real de vectores no dominados y la distancia generacional utilizadas por Duarte (2001).

Para el uso de estas métricas se define un conjunto de variables que serán utilizadas posteriormente por estas. Para ello es necesario definir a $\boldsymbol{Y_{true}}$ como el frente Pareto óptimo real y a $\boldsymbol{Y_{known}}$ como el frente de Pareto computado (los resultados obtenidos en cada una de las ejecuciones de ambos algoritmos). Y como $\boldsymbol{Y_{true}}$ es la incógnita, en Duarte (2001) citado de Schott (1995) lo que se hace es reunir todas las soluciones no dominadas halladas en la unión de todos los conjuntos solución propuestos por ambos algoritmos en un conjunto que se denota como $\boldsymbol{Y_{true}}$.

Para los resultados antes descritos en las tablas 3 y 4 correspondientes a los algoritmos utilizados en este trabajo, se elaboró aplicando el concepto de no dominancia el conjunto $\boldsymbol{Y_{true}}$. Este conjunto está constituido por los elementos descritos en la tabla 5 con una cardinalidad de 23 soluciones.

Tabla 5. Conjunto de soluciones que conforma el conjunto Y_{true} .

Número de solución	Soluciones (Completitud, Costo)
1	0.04439189502480637 , 52603.0
2	-0.9139063468177392 , 1327.0
3	-0.7072871931532825 , 10372.0
4	-0.6184912134279222 , 16480.0
5	-0.4300222299792398 , 21791.0
6	0.9184456516168708 , 116228.0
7	0.6281008452033429 , 67557.0
8	0.8203978144884208 , 94367.0
9	0.03849568153365621 , 48837.0
10	0.7507762013691395 , 76022.0
11	-0.12759055290700855 , 34398.0
12	0.1886484823193684 , 58133.0
13	-0.7362966147776274 , 8143.0
14	0.10055117101419571 , 56138.0
15	0.7092396940984675 , 73682.0
16	0.9430551801722363 , 140126.0
17	-0.4877080725181991 , 19528.0
18	0.6973645242794112 , 70772.0
19	0.9313340191653112 , 131726.0
20	0.7918340651466995 , 90238.0
21	-0.301129135970311 , 22283.0
22	-1.5037974683544304 , 385.0
23	0.8010605750012811 , 91838.0

Con estos datos se puede pasar a establecer una comparación de desempeño entre ambos algoritmos, para ello se utilizan las métricas de los siguientes subepígrafes.

3.3.1. Generación de vectores no dominados (GVND)

Esta métrica cuenta el número de soluciones en el frente Pareto propuesto $\boldsymbol{Y_{known}}$. O sea, el número de soluciones obtenidas en cada ejecución de cada uno de los algoritmos. Se puede definir mediante la siguiente ecuación:

$$GVND \triangleq |\boldsymbol{Y_{known}}|_c$$

De acuerdo con Duarte (2001), este en su investigación hace una observación respecto a esta métrica ya que a pesar de contar el número de soluciones no dominadas obtenidas, la métrica no refleja que tan "lejos" se encuentran los vectores de $\boldsymbol{Y_{known}}$ de los de $\boldsymbol{Y_{true}}$. La autora de esta investigación asume el criterio citado anteriormente.

3.3.2. Razón de generación de vectores no dominados (RGVND)

Esta métrica calcula la razón entre la cantidad de vectores no dominados hallados en (GVND para cada ejecución de cada algoritmo) y la cantidad de vectores no dominados existentes. Esta métrica se define como:

$$RGVND \triangleq \frac{\text{GVND}}{|\boldsymbol{Y_{true}}|_c}$$

Como el objetivo es obtener un frente Pareto $\boldsymbol{Y_{known}}$ tan similar al verdadero frente Pareto como sea posible, un valor cercano a 1 sería los más deseable.

3.3.3. Generación real de vectores no dominados (GRVND)

Esta métrica calcula la cantidad de elementos en el frente Pareto hallado $\boldsymbol{Y_{known}}$, que en efecto pertenecen al frente Pareto óptimo real por cada ejecución de ambos algoritmos. Se define de la siguiente manera:

$$GRVND \triangleq |\{y | y \in \boldsymbol{Y_{known}} \wedge y \in \boldsymbol{Y_{true}}\}|_c$$

3.3.4. Distancia generacional (G)

La distancia generacional indica que tan lejos están los elementos del frente de Pareto actual respecto al frente de Pareto verdadero, es decir su valor representa que tan lejos está $\boldsymbol{Y_{known}}$ de $\boldsymbol{Y_{true}}$. Se define como:

$$G \triangleq \frac{(\sum_{i=0}^{N} d_i^2)^{\frac{1}{2}}}{GVND}$$

Donde d_i es la distancia euclidiana (en el espacio objetivo) entre cada vector objetivo de f en $\boldsymbol{Y_{known}}$ y su miembro correspondiente más cercano en $\boldsymbol{Y_{true}}$.

$$d_i = min_j\{\sqrt{(f_1^i - f_1^j)^2 + (f_2^i - f_2^i)^2}\}$$

Un valor grande de G indica que $\boldsymbol{Y_{known}}$ está alejado de $\boldsymbol{Y_{true}}$; siendo $G = 0$ la situación ideal.

3.3.5. Resultados de las métricas

Al aplicar estas métricas a los resultados de las ejecuciones de los algoritmos empleados en esta investigación se obtuvieron los resultados mostrados en las tablas 6 y 7.

Tabla 6. Resultados de las métricas en las ejecuciones del algoritmo NSGAII.

$\boldsymbol{N^o}$	GVND	RGVND	GRVND	G	Tiempo (ms)
1	10	0.4347	2	1028.686057553229	25767
2	10	0.4347	1	742.4495985176429	38939
3	10	0.4347	3	1330.966674735487	24997
4	10	0.4347	1	1056.8096230922495	17115
5	10	0.4347	1	877.1880986361423	25116
Promedio				1007.212	26386.8

Tabla 7. Resultados de las métricas en las ejecuciones del algoritmo SPEA2.

N^o	GVND	RGVND	GRVND	G	Tiempo (ms)
1	10	0.4347	3	1385.3888675658022	30389
2	10	0.4347	6	349.1493484256466	26650
3	10	0.4347	2	457.98615586326014	36227
4	10	0.4347	0	1493.7774507198146	19238
5	10	0.4347	4	951.6957906146208	39574
Promedio				927.592	30415.6

Analizando los datos de las tablas 6 y 7, se puede observar que para ambos algoritmos el valor de la métrica de generación de vectores no dominados es 10 para cada una de las ejecuciones, esto está dado por el tamaño de la población inicial en ambos algoritmos y en el caso del algoritmo SPEA2 por el tamaño del archivo utilizado. En la métrica razón de generación de vectores no dominados ambos algoritmos presentan los mismos valores: 0.4347 porque dependen de la métrica anterior. Ambas métricas no muestran resultados factibles para comparar ambos algoritmos porque no existen diferencias entre ellos en cuanto a estas. Sin embargo, con la métrica generación real de vectores no dominados es diferente. Se ve claramente como el algoritmo SPEA2 logra encontrar más soluciones en el frente óptimo de Pareto. Y esta afirmación la demuestra la distancia generacional ($\boldsymbol{G}$) entre $\boldsymbol{Y_{known}}$ y $\boldsymbol{Y_{true}}$. Para las ejecuciones con el algoritmo SPEA2, la distancia generacional es menor en promedio que para las ejecuciones con el algoritmo NSGAII. Aunque, se puede destacar que el algoritmo SPEA2 consume en promedio un poco más de tiempo para hallar las soluciones que el algoritmo NSGAII.

Sin embargo, sería oportuno calcular cuántas soluciones de cada uno de estos dos algoritmos dominan al otro. Este resultado se puede ver en la tabla 8.Y sigue demostrando que el SPEA2 encuentra mejores soluciones que el algoritmo NSGAII, es decir posee 41 soluciones dominadas de las 50 soluciones que ofrecen cada uno de los algoritmos. Teniendo en cuenta estos planteamientos se puede concluir que el algoritmo SPEA2 obtiene mejores soluciones con respecto al algoritmo NSGAII, aunque este primero consume un poco más de tiempo.

El SPEA2 ofrece las soluciones más óptimas para llevar a cabo un mejor desempeño del proceso en la organización que lo ejecuta. No obstante, se deja a criterio de los analistas del negocio la selección de las posibles soluciones más óptimas que ofrecen los algoritmos pues es el propio analista el que conoce qué alternativa pudiera mejorar el proceso en la organización.

Tabla 8. Valor de las soluciones que dominan los algoritmos con respecto al otro.

Número de soluciones del NSGAII que dominan sobre las soluciones del SPEA2	33
Número de soluciones del SPEA2 que dominan sobre las soluciones del NSGAII	**41**

3.4. Conclusiones

En el presente capítulo se ha presentado la aplicación informática desarrollada la cual permite la optimización de los procesos de negocio en función del costo y la completitud mediante dos algoritmos multiobjetivos: NSGAII y SPEA2.

Se utilizó un caso de estudio para probar la aplicación, en este caso el proceso de negocio relacionado con la Producción de la Langosta Entera Precocinada perteneciente a la empresa de Batabanó, para el cual se obtuvo diseños optimizados del proceso teniendo en cuenta un menor costo y una mayor completitud de las actividades que conforman al proceso.

Se expusieron los resultados obtenidos por cada uno de los algoritmos y se validaron estos mediante 4 métricas, mostrando los siguientes resultados: el algoritmo SPEA2 logra encontrar más soluciones en el frente óptimo de Pareto según la generación real de vectores no dominados (GRVND); para las corridas con SPEA2, la distancia generacional es menor en promedio que para las corridas con NSGAII, lo que significa que sus soluciones tienden a acercarse más al óptimo real que con el NSGAII; el algoritmo SPEA2 consume en promedio un poco más de tiempo para hallar las soluciones, que el algoritmo NSGAII. Por tanto, se puede corroborar que el algoritmo SPEA2 ofrece mejores soluciones óptimas del diseño del proceso optimizado con respecto al NSGAII, aunque consume un poco más de tiempo en ofrecer estos resultados.

CONCLUSIONES Y RECOMENDACIONES

El desarrollo de esta investigación permitió lograr un nivel determinado de integración entre la minería de procesos y la optimización multiobjetivo facilitando así un compendio entre estas para la aplicación de la optimización multiobjetivo a los procesos de negocio. Se definió el problema multiobjetivo y la dominancia de Pareto para un problema de minimización y maximización de dos objetivos: costo y completitud. Se escogieron a los algoritmos evolutivos, en este caso, el NSGAII y SPEA2 como una alternativa para lograr nuevos diseños de procesos optimizados y así obtener un mejor desempeño de estos en las organizaciones.

Por otra parte, se definieron las funciones objetivos costo y completitud para la optimización de los modelos de los procesos de negocio. Se diseñó la codificación de las soluciones utilizando la matriz causal facilitando así el trabajo con los operados genéticos (selección, cruzamiento y mutación) de los algoritmos utilizados. Se rediseñaron los algoritmos NSGAII y SPEA2 mediante el empleo del *framework* JMetal hacia un problema de optimización aplicado a procesos de negocio.

Se implementó una aplicación que permite la optimización de los modelos de procesos de negocio basándose en la representación del mismo en una red de Petri y su archivo XES. Se probó la aplicación para el proceso de la Producción de la Langosta Entera Precocinada facilitándole a los analistas del negocio la obtención de modelos del proceso optimizado y así lograr un mejor desempeño de este en la empresa de Batabanó. Se evaluaron los algoritmos teniendo en cuenta las métricas de generación de vectores no dominados, la razón de generación de vectores no dominados, la generación real de vectores no dominados y la distancia generacional. Para ello se tuvieron en cuenta los mismos parámetros de entrada como son: el tamaño de la población inicial (10), el número de evaluaciones (1000), el factor de cruzamiento (90%) y el factor de mutación (50%) y el caso de SPEA2, el tamaño del archivo (10). Estas arrojaron que el algoritmo SPEA2 logra encontrar mejores soluciones en el frente óptimo de Pareto, pero sacrificando un poco más de tiempo de ejecución comparado con el algoritmo NSGAII.

Se recomienda para próximas investigaciones:

- Extender la herramienta de optimización de procesos de negocio a otros algoritmos evolutivos multiobjetivos de manera tal que se puedan realizar comparaciones entre los mismos y definir cuales presentan un mejor comportamiento en ofrecer un mejor diseño del proceso de negocio que se quiera optimizar.
- Brindarle al analista del negocio la posibilidad de escoger los objetivos de optimización que estime pertinente para la realización de la optimización del proceso que se quiera mejorar.
- Evaluar nuevos criterios de optimización.

Referencias Bibliográficas

Ahmadikatouli, A. y Aboutalebi, M., 2011. New Evolutionary Approach to Business Process Model Optimization. Proceedings of the International MutiConference of Engineers and Computer Scientists, Volumen II.

Alcover, R., Benlloch, J., Blesa, P., Calduch, M.A., Celma, M. y Ferri, C., 2007. Análisis del rendimiento académico en los estudios de informática de la universidad politécnica de valencia aplicando técnicas de minería de datos.

Arias, M. y Rojas, E., 2014. Deciphering event logs in SharePoint Server: A methodology based on process mining. In Computing Conference, pp. 1-12.

Arias, M. y Rojas, E., 2016. A guide to manage business processes through process mining. SCielo.

Arias, M., Rojas, E., Muñoz, J. y Sepúlveda, M., 2015. A framework for Recommending Resource Allocation based on Process Mining. In Business Process Managament Workshops, 13 th Interational, pp. 458-470.

Back, T., 1996. Evolutionary Algorithms in Theory and Practice: Evolution Strategies Evolutionary Programmig Genetic Algorithms. Oxtford University Press. New York.

Billington, J. y otros, 2003. The Petri Net Markup Language: Concepts, Technology and Tools.

Bonitasoft, 2018. BonitaSoft. [En línea] Available at: http://www.bonitasoft.com [Último acceso: 4 6 2018].

Chong, E. y Zak, S., 2013. An introduction to optimization. En: An introduction to optimization 4th ed. Segunda ed. John Wiley y Sons, p. 15.

Darwin, C. R., 1959. The Origin of Species by Means of Natural Selection, or the preservation of favoured races in the struggle for life.

Deb, K., 1999. Solving Goal Programming Problems Using Multi-Objective Genetic Algorithms. Congress on Evolutionary Computation. IEEE Service Center.

Deb, K. y Goel, T., 2001. Controlled elitist non-dominated sorting genetic algorithm for better convergence. Springer-Verlag Berlin Heidelberg, Volumen 1993, pp. 67-81.

Deb, K., Patrap, A., Agarwal, S. y Meyarivan, T., 2002. A Fast and Elitist Multi-Objective Genetic Algorithm: NSGAII. IEEE Transactions on Evolutionary Computation, 6(2), pp. 182-197.

Díaz, A. V., 2014. Algoritmo genético distribuido para el descubrimiento de procesos.

Duarte, S. F., 2001. OPTIMIZACIÓN MULTIOBJETIVO. Asunción, Paraguay.

Edgeworth, F. Y., 1881. Mathematical Physics. P. Keagan.

Ehrgott, M. y Wiecek, M., 2005. Multiobjetive programming: Multiple criteria decision analysis state of the art surveys. Springer, pp. 667-722.

Fischer, L., 2003. The Workflow Handbook. Future Strategies Inc.

Giraldo, J. M., 2016. Modelo de integración de BPM y Minería de Procesos con un Enfoque Dimensional para la Optimizacion de Indicadores KPI. Facultad de Minas, Departamento de Ciencias de la Computación y de la Desición, Medellín, Colombia.

Goldberg, D., 1998. Genetic Algorithms in Search. University of Alabama: Addison-Wesley Publishing Company.

González, B. C. y otros, 2011. Un Framework para Big Data Optimization Basado en jMetal y Spark.

González, P. D., 2014. Optimización por algoritmos genéticos y redes de Petri del tiempo de proceso de un sistema de manufactura flexible.

Günther, C. W., 2009. XES: Extensible Event Stream. Standard definition.

Günther, W. y Verbeek, E., 2014. XES Standar Definition.

Hofacker, I. y Vetschera, R., 2001. Algorithmical approaches to business. Computers y Operations Research , Issue 28, pp. 1253-1275.

Kees, O. O. V. H., Post, R. L. S. y Martijn, J. v. d. W., 2006. Yasper: a tool for workflow modeling and analysis. Researchgate.

Medeiros, A., 2006. Genetic Process Mining, Ph.D. thesis, Technische Universiteit Eindhoven, The Netherlands.

Medeiros, A., Weijters, A. y Van der Aalts, W., 2007. Genetic process mining: an experimental evaluation. Springer Science+Business Media.

Miettinen, K., 2008. Intorduction to Multiobjetive Optimization: Noninteractive Approaches. Springer berlin Heidelberg, Volumen 5252, pp. 1-26.

Murata, T., 1989. Petri nets: Properties, analysis and applications. Proceedings of the IEEE, 77(4), pp. 541-580.

Nebro, A. y Durillo, J., 2014. jMetal 4.5 User Manual.

OMG, 2010. Business Process Model and Notation (BPMN). Object Management Group. [En línea] Available at: http://www.omg.org/spec/BPMN/2.0/PDF/ [Último acceso: 24 4 2018].

Osuszek, L., 2012. BPM optimization. Workflow map optimization by using multiobjective algorithms.

Pareto, V., 1896. Cours D'Economie Politique. F. Rouge.

Parka, S. y Kangb, Y., 2016. A Study of Process Mining-based Business Process Innovation. Procedia Computer Science, Issue 91, pp. 734-743.

Peñarrieta, R., 2017. Programacion Java y Netbeans.

Pérez, R. y otros, 2011. Generating events logs from non-process-aware systems enabling business process mining. Enterprise Information Systems, 5(3), pp. 301-335.

Petri, C. A., 1962. Kommunikation mit Automaten. Phd thesis, Institut fur Instrumentelle Mathematik, Bonn.

Puris, A., 2009. Desarrollo de meta-heurísticas poblacionales para la solución de problemas complejos. Universidad Central Marta Abreu de las Villas.

Reisig, W. y Rozenberg, G., 1998. Lectures on Petri nets I: basic models. Lectures notes in computer science. Springer, Berlin, Volumen 1491.

Rojas, E., Muñoz, J., Sepúlveda, M. y Capurro, D., 2016. Process mining in healthcare: A literature review. Jorunal of biomedical informatics, Issue 61, pp. 224-236.

Schott, J., 1995. Fault tolerant design using single and multicriteria genetic algorithm optimization. Tesis Magistral, Departamento de Aeronáutica y Astronáutica, Instituto de Tecnología de Massachussets.

Srinivas, N. y Deb, K., 1994. Multiobjective Optimization Using Non-dominated Sorting Genetic Algorithm. MIT Evolutionary Computation, 2(3), pp. 221-248.

Talavera, F. S., 2005. Comparación de Algoritmos Evolutivos Multi-Objetivos en un ambiente Multicast. Facultad de Ciencias y Tecnología. Universidad Católica "Nuestra Señora de la Asunción". Paraguay.

Ter Hofstede, A. y Van der Aalst, W., 2005. YAWL: Yet Another Workflow Language. Information Systems, 30(4), pp. 245-275.

Tiwari, A., Vergidis, A. y Majeed, B., 2006. Evolutionary Multi-Objective Optimization of Business Process. Proceeding of IEEE Congress on Evolutionary Computing, pp. 3091-3097.

Trcka, N., Pechenizkiy, M. y Van der Aalst, W., 2010. Process mining from educational data. Chapman Hall/CRC, pp. 123-142.

Van der Aalst, W., 1998. The aplication of Petri Nets to workflow management. The Journal of Circuits, Systems and Computers, 8(1), pp. 21-66.

Van der Aalst, W., 2002a. A class of Petri nets for modeling and analysing business processes. Department of Mathematics and Computing Science, Eindhoven University of Technology.

Van der Aalst, W., 2002b. Making work flow: on the application of Petri Nets to business process management. Springer-Verlag, Issue 2360, pp. 1-22.

Van der Aalst, W., 2013. Using Process Mining to Bridge the Gap between BI and BPM. IEEE Computer Society.

Van der Aalst, W., Adriansyah, A. y Medeiros, A. F., 2011. Process Mining Manifiesto. Business Process Management Workshops. Springer-Verlag, Issue 99, pp. 5-20.

Van der Aalst, W., Hofstede, A. y Weske, M., 2003. Business Process Management : A Survey. In Business Process Management, pp. 1-12.

Vergidis, K., Tiwari, A. y Majeed, B., 2006. Business process improvement using multi-objetive optimization. BT Technology Journal, 2(24), pp. 229-235.

Vergidis, K., Tiwari, A., Majeed, B. y R, R., 2007. Optimisation of business process designs: An Algorithmic approach with multiple objetives. Int. Journal Production Economics, Issue 109, pp. 105-121.

Weske, M., 2010. Business Process Management. Concepts, Languages, Architectures.. Berlin: Springer.

Zhou, Y. y Chen, Y., 2003. Project-oriented business process perfomance optimization. Proceedings of IEEE International Conference on System, Man and Cybernetics.

Zitzler, E., Knowles, J. y Thiele, L., 2008. Quality Assessment of Pareto Set Approximations. Springer Berlin Heidelberg, Volumen 5252, pp. 373-404.

Zitzler, E., Laumanns, M. y Thiele, L., 2000. SPEA2: Improving the Strengh Pareto Evolutionar Algorithm. Computer Engineering and Networks Laboratory (TIK).

Zitzler, E. y Thiele, L., 1999. “Multiobjective Evolutionary Algorithms: A comparative Case Study and the Strength Pareto Approach”. EEE Trans. Evolutionary, 3(4), pp. 257-271.

Glosario de términos

Función objetivo (FO): El término es utilizado en la optimización multiobjetivo. Hace alusión a una función matemática que dentro de un proceso de negocio ha de ser optimizada, o sea, el conjunto de resultados al evaluarla debe minimizarse o maximizarse según el caso.

Proceso de negocio: Es una serie de actividades que las personas y sistemas realizan de forma ordenada para lograr un objetivo dentro de un negocio.

Minería de procesos: Es una técnica de administración de procesos que permite analizar los procesos de negocios de acuerdo con un registro de eventos. A través de esta actividad se desea extraer conocimiento desde los registros de evento de los procesos almacenados por los sistemas.

BPMN: Es una notación gráfica que plasma la lógica de las actividades, los mensajes entre los diferentes participantes y toda la información necesaria para que un proceso sea analizado, simulado y ejecutado.

OMG: *Object Management Group* (Grupo de Gestión de Objetos). Es un consorcio, formado en 1989, dedicado al cuidado y el establecimiento de diversos estándares de tecnologías orientadas a objetos, tales como *UML, XMI, CORBA* y *PNML*. Es una organización sin fines de lucro que promueve el uso de tecnología orientada a objetos mediante guías especificaciones.

BPM: *Business Process Management* (Gestión de Procesos de Negocio). Es la disciplina de gestión empresarial enfocada a los procesos de negocio.

Optimización multiobjetivo. Cuando en un problema de optimización se tienen varias funciones objetivo. Tiene la tarea de encontrar una o más soluciones óptimas. También se puede nombrar toma de decisiones multicriterio.

NSGAII: *Non dominated Sorting Genetic Algorithm II* (Algoritmo Genético de Clasificación No Dominada II). Algoritmo genético elitista (elige los mejores P individuos de la unión de las poblaciones padre e hijo) computacionalmente más eficiente que su versión anterior (NSGA). La variable P se refiere al tamaño de la población con que trabaja.

SPEA2: *Strength Pareto Evolutionary Algorithm 2* (Algoritmo Evolutivo de Pareto de Fuerza 2). Algoritmo genético que dirige mejor la búsqueda de soluciones que la versión anterior (SPEA). Incorpora una estrategia fina de asignación de *fitness* que considera, para cada individuo, el número de los individuos que lo domina y el número de los individuos por los cuales es dominado.

Red de Petri: Es un grafo bipartito ponderado y dirigido cuyos nodos son de tipo plaza (lugar) o de tipo transición.

Lugares: Se representan mediante círculos y representan esa fase "estable" por la que atraviesa el sistema entre dos "sucesos" consecutivos que acontecen en una red de Petri.

Transiciones: Se representan por segmentos de recta o un rectángulo, los cuales llevan asociados los eventos, cuya activación, provoca el disparo de la transición y por ende el marcado de uno o más lugares siguientes.

Arcos: Son segmentos de recta orientados que unen lugares y transiciones de forma alternativa. Cada arco, lleva asociado una función de peso w, que deberá ser un entero positivo {0, 1, 2,…}.

Tokens: Se representan por puntos en los lugares, y determinan la disponibilidad de los recursos o la ejecución de las actividades.

PNML: Es un archivo que cumple los requerimientos del formato de una red de Petri.

MOEA: *Multiobjetive Evolutionary Algorithm* (Algoritmo Evolutivo Multiobjetivo). Se refiere al algoritmo evolutivo que puede ser empleado en la optimización multiobjetivo.

Anexos

Anexo #1.

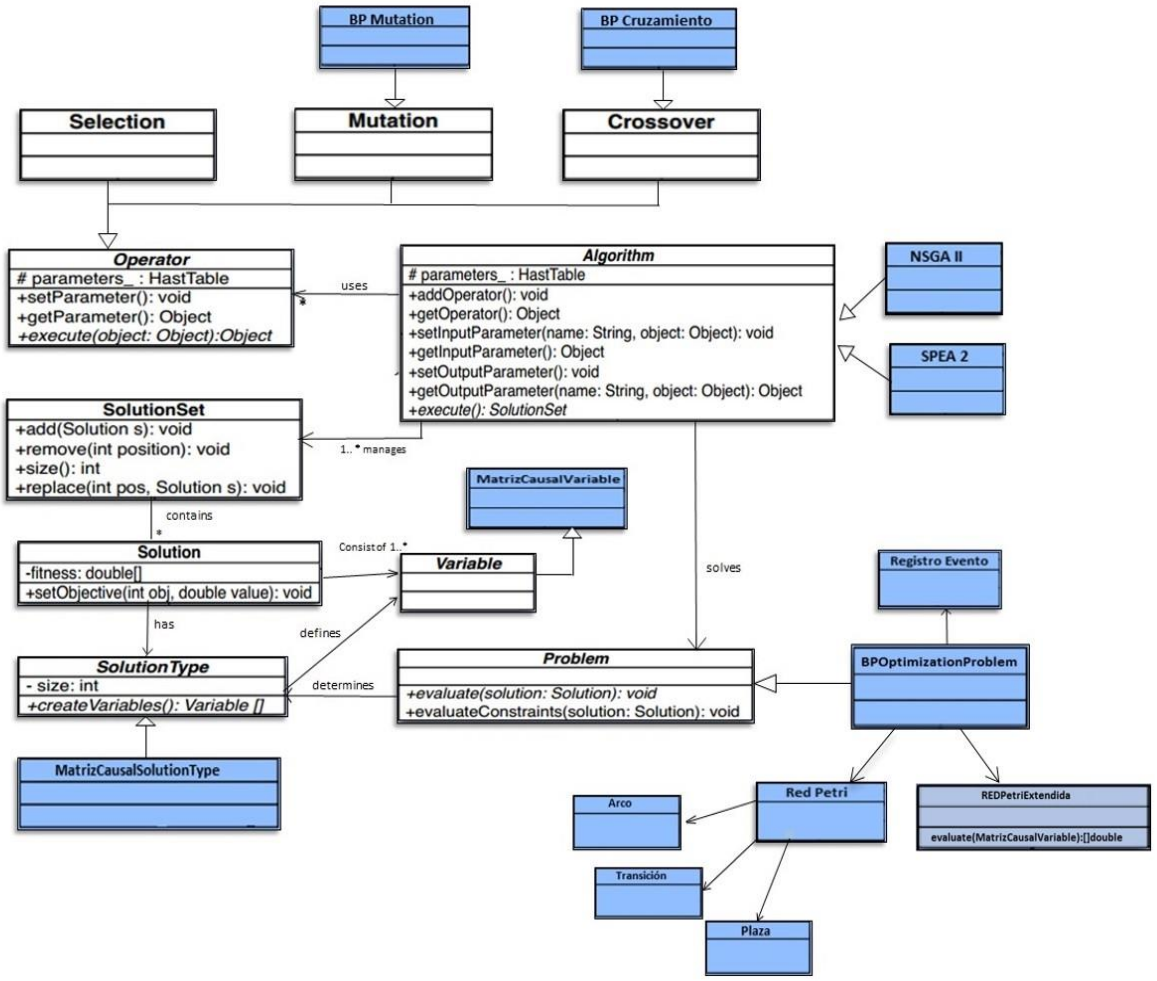


Figura 20. Diseño de clases utilizadas para el manejo de los algoritmos.