Universidad de las Ciencias Informáticas
Facultad #3

Título: *Algoritmo para la estimación de información ausente en las trazas asociadas a subprocesos ordenados secuencialmente en la minería de proceso.*

Trabajo de Diploma para optar por el título de
Ingeniero en Ciencias Informáticas

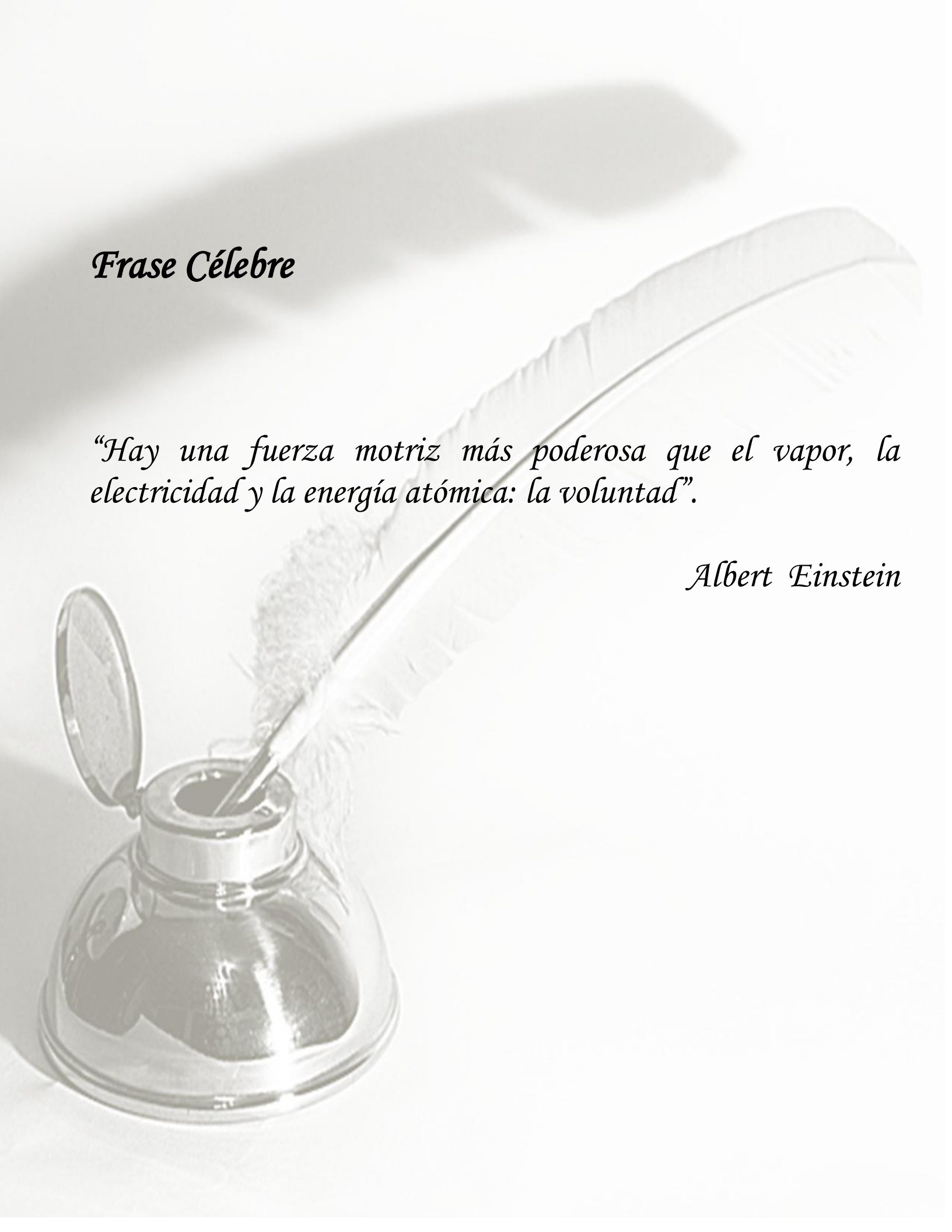
Autor: *Carlos Torres González*
Tutor: *MSc. Raykenler Yzquierdo Herrera*

Junio 2012

Frase Célebre

"Hay una fuerza motriz más poderosa que el vapor, la electricidad y la energía atómica: la voluntad".

Albert Einstein



DECLARACIÓN DE AUTORÍA

Declaro que soy el único autor de este trabajo y autorizo a la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

Para que así conste firmo la presente a los __ días del mes de _____ del año 2012.

_____ _____

Carlos Torres González MSc. Raykenler Yzquierdo Herrera

DATOS DE CONTACTO

Raykenler Yzquierdo Herrera: Graduado de Ingeniería en Ciencias Informáticas en la Universidad de las Ciencias Informáticas en el 2007. Se desempeñó durante el 2007 como analista principal del proyecto CCV y durante el 2008 como gerente general del proyecto. Durante el 2008 se incorpora al proyecto ERP Cuba, desempeñando el rol de jefe de Soporte hasta el 2009. Se desempeñó como arquitecto de la línea Planificación presupuestada y empresarial durante el 2010. En Septiembre del 2010 defiende su tesis de maestría. Actualmente trabaja en su doctorado. Ha participado en múltiples talleres, fórums y eventos en los cuales ha obtenido resultados relevantes. Ha impartido clases a pregrado generalmente de asignaturas de la disciplina de Programación y en posgrado ha impartido cursos de Soporte y Minería de proceso.

AGRADECIMIENTOS

Agradezco a todos los que han contribuido con la realización de este trabajo y con mi formación profesional, especialmente:

A mis padres, por guiarme y apoyarme siempre, por sus consejos y por todo su amor y dedicación.

A mi hermano que siempre me ha apoyado incondicionalmente y ha sabido ser hermano y amigo.

A Zerelda por su apoyo incondicional, por todo su amor y paciencia, por estar ahí cuando más lo he necesitado.

A Irma Zerelda por su preocupación y confianza.

A Leidy por su cariño de hermana y los momentos en familia.

A mis tíos Pedrito y Mercedes por toda su ayuda y apoyo.

A mi tía Odalys por su confianza y cariño.

A mis primas Ade y Mare por todo su amor de Hermanas.

A mis abuelos paternos por todo el cariño que me han dado.

A mi amiga Madelin por sus buenos consejos y su amistad.

A Rodney y familia por su amistad incondicional.

A Geisys y Viviana por haberlas conocido.

A mis amigos Boris, Dayan, Yordanis, Eyeris por su amistad y apoyo todos estos años.

A todas las buenas amistades y profesores que he conocido en estos años de universidad.

A mi tutor Raykenler por todo lo que he aprendido y sus buenas recomendaciones

A Alkaid y al grupo de trabajo por su colaboración.

A mis vecinos: Asela, Lazarito, Yeny, Gladis, Cecilia por tenerme siempre presente, por sus atenciones.

DEDICATORIA

*A mis **Padres**, porque sé que es un sueño hecho realidad para ellos, por ser mis guías en la vida, por su desvelo y sacrificio, por siempre confiar en mí.*

*A mis **Abuelos** maternos Adelfa y Pedro que ya no están.*

*A mi **Amigo** Rodney que por azahar de la vida no termino de graduarse, por su amistad y cariño de hermano.*

RESUMEN

La investigación de este trabajo va dirigida a realizar y proponer un algoritmo para la estimación de la información ausente en las trazas utilizadas en la Minería de Proceso. Para ello se realiza un estudio de diferentes técnicas pertenecientes a esta área. Sin embargo la mayoría de los algoritmos de minería de procesos parten del supuesto de que las trazas son completas y están libres de ruido, pero esto rara vez ocurre. La solución propuesta está basada en dos técnicas por separado, una utiliza la programación dinámica, y la otra está inspirada en una estrategia metaheurística. Para la estimación primeramente se realiza la alineación de las trazas, esto ocurre en la etapa de pre-procesado de las mismas, posibilitando detectar los diferentes casos de ausencia de información, a partir de este momento se estima la información ausente y se genera un nuevo registro de evento. En la etapa final se discuten los resultados obtenidos a partir de la aplicación de la propuesta.

TABLA DE CONTENIDOS

INTRODUCCIÓN

En la actualidad el avance de las tecnologías de la información y las comunicaciones TIC han propiciado que las grandes y medianas empresas por lo general gestionen sus recursos mediante la informatización de los mismos, permitiendo un mayor control y un mejor manejo de la empresa, así como el aprovechamiento eficiente del tiempo de sus procesos.

La mayoría de las empresas utilizan sistemas de información, con el objetivo de gestionar la ejecución de sus procesos de negocios. Para esto se utilizan softwares, que registran en forma de trazas toda la información que se generan al ocurrir instancias o casos de dichos procesos. El hecho de que a partir de la información contenida en las trazas se pueda identificar el proceso, se le denomina Minería de Proceso o minería de flujos de trabajo [1]. Con el modelo de procesos basado en trazas se pueden realizar comparaciones entre el modelo teórico y el modelo descubierto, además de que es posible apoyar el rediseño de los procesos en la empresa así como su utilidad como soporte a la operacionalización de los mismos. En los sistemas de hoy en día, mediante la extracción de la información de las trazas existentes, la minería de procesos nos permite descubrir, monitorear y mejorar los procesos reales [2, 3].

Un proceso puede ser analizado de tres perspectivas distintas, la perspectiva de flujo de control, la de los recursos u organizacional como también se conoce, y la de casos. Para seleccionar la o las perspectivas a analizar hay que tener en cuenta la información contenida en las trazas. La perspectiva de flujo de control que es la más utilizada, permite determinar cómo se organizan las tareas que dan lugar al proceso y deja bien definidas sus dependencias. Esta dimensión debe reflejar información relacionada con la identificación de la instancia ejecutada, la identificación de la tarea así como el tiempo de la misma. En la dimensión de los recursos, la información que debe contener es la referente al posible usuario que ejecutó la tarea, además es capaz de identificar la estructura organizacional con la que se ejecutan las tareas, permitiendo conocer la jerarquía entre roles y usuarios a lo largo del proceso. En la tercera perspectiva se debe tener información detallada de las tareas, tales como, el sexo y la edad del usuario. Además se pueden descubrir casos excepcionales, como la probabilidad que existe de que determinada persona demorará cierto tiempo en una tarea.

La mayoría de los algoritmos de minería de procesos parten del supuesto que las trazas son completas y están libres de ruido. En la realidad esto rara vez se cumple. [4] El hecho que una traza sea completa significa que se ha registrado de manera exacta el comportamiento de cada una de las instancias ejecutadas de un proceso determinado. Por lo tanto si las trazas reflejan parcialmente este comportamiento se considera que presentan ausencia de información.

La información ausente en las trazas se produce de dos formas diferentes, se puede manifestar en el hecho que determinadas instancias de un proceso no han ocurrido aún, denominado incompletitud de la información [5] o se puede manifestar en la ausencia de una o varias tareas del proceso en las trazas, debido a que las mismas no son registradas al no ser soportadas por los sistemas utilizados o porque las trazas fueron afectadas por el ruido. En el primer caso no afecta en nada el modelo descubierto debido a que este refleja lo que en realidad ocurre y no lo que puede ocurrir, sin embargo en el segundo caso puede provocar que los algoritmos obtengan modelos en los que se reflejan incorrectas relaciones entre las tareas, afectando al modelo ejecutado y a la utilidad del modelo descubierto.

No siempre se puede interpretar correctamente la información ausente en las trazas, debido a que los algoritmos no reflejan las tareas invisibles de manera explícita o no lo hacen de manera correcta en todas las circunstancias posibles [6]. Esto puede agravarse aún más cuando se manejan procesos poco estructurados. Para cubrir el comportamiento reflejado en las trazas, el modelo descubierto se apoya en un grupo de patrones de flujo de control, además para comprender el modelo descubierto puede ser útil reflejar las tareas ausentes.

Para evitar problemas de completitud, las trazas se pueden pre procesar con anterioridad. En este sentido, se realizan acciones de verificación, limpieza, agrupamiento y alineado de las mismas [7-9]. Sin embargo, esto no permite resolver el problema de la información ausente en todas sus manifestaciones. Otras soluciones se han centrado en algoritmos robustos ante el ruido y algunas situaciones de ausencia de información [10-14]. Existen algoritmos que reconocen los aspectos más significativos en las trazas y los reflejan en el modelo descubierto, permitiendo reducir el ruido y la información ausente en alguna medida [15].

La ausencia de información en las trazas asociadas a subprocesos puede provocar que los algoritmos utilizados para la estimación de la información obtengan modelos que reflejen relaciones incorrectas entre las tareas afectando esto la comprensión del proceso ejecutado y la utilidad del modelo descubierto. Cuando existe ausencia de información la mayoría de los algoritmos estudiados no reflejan con claridad las tareas invisibles o no lo hacen correctamente en todas las circunstancias posibles.

Teniendo en cuenta la anterior situación problemática queda como **problema a resolver:** ¿Cómo disminuir la afectación que provoca la ausencia de información en las trazas asociadas a subprocesos ordenados secuencialmente sobre la estructura y comprensión del modelo de procesos descubierto a partir de técnicas de minería de proceso?

Para dar solución al problema planteado se tiene como **objeto de estudio:** Minería de proceso, enfocado en el siguiente **campo de acción:** Estimación de información ausente en las trazas usadas en la minería de proceso.

Se define como **objetivo general:** Desarrollar un algoritmo para la estimación de la información ausente en las trazas asociadas a subprocesos ordenados secuencialmente, basado en un árbol de bloques de construcción y Meta-heurísticas para el descubrimiento de modelos de procesos más estructurados y comprensibles.

Como **Objetivos específicos:**

- Fundamentar la investigación mediante la evaluación del Marco Teórico.
- Definir el diseño de la propuesta de solución.
- Implementar el algoritmo diseñado.
- Validar la propuesta.

Para dar cumplimiento a los objetivos específicos planteados anteriormente, se proponen las siguientes **tareas investigativas:**

- Análisis de los principales conceptos y trabajos relacionados con la minería de proceso y la ausencia de información en las trazas.
- Estudio del mecanismo de alineación de las trazas propuesto para la minería de proceso.
- Estudio del mecanismo de descomposición del proceso analizado a partir de la alineación de las trazas.
- Desarrollo de un algoritmo que permita reconocer la ausencia de información en las trazas utilizadas a subprocesos ordenados secuencialmente.
- Implementación del algoritmo desarrollado como una biblioteca que permita la reutilización del código y el desarrollo rápido de aplicaciones para la minería de proceso.
- Diseño del experimento que permitirá la evaluación de la propuesta de solución.
- Evaluación de los algoritmos y herramientas desarrolladas en un entorno real.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

Este capítulo explora los principales conceptos de Minería de Procesos y técnicas relacionados a los procesos de negocio y la ausencia de información en las trazas. Se refieren algoritmos utilizados y mecanismos como alineación de las trazas y descomposición del proceso a partir de la alineación de las trazas. Además se demuestra como los algoritmos estudiados no son capaces de definir las tareas invisibles en una traza o log de eventos, y se propone el uso de técnicas de metaheurística y de programación dinámica para la identificación de la secuencia oculta que es una da las formas donde puede ocurrir ausencia de información.

1.1 Minería de proceso

Las técnicas de minería de proceso permiten extraer información útil de las trazas registradas por los sistemas de información. Un elemento fundamental en la minería de proceso es el descubrimiento del flujo de control, esto se refiere a la construcción automática del modelo de proceso en el que se describen las dependencias causales entre las actividades de procesos [16].

La minería de proceso permite la unión entre la minería de datos y el modelado y análisis de los procesos de negocio. Bajo el área de BI (siglas de Business Intelligence)[17-19] se ha dado lugar un grupo de términos que abarcan distintos tipos de análisis tales como, BAM (siglas de Business Activity Monitoring) referido a las tecnologías que hacen posible un análisis en tiempo real de los procesos de negocio[20, 21]. CEP (siglas de Complex Event Processing) referido a las tecnologías que permiten procesar grandes cantidades de eventos para monitorear, guiar y optimizar el negocio en tiempo real[22, 23]. CPM (siglas de Corporate Performance Management) referido a la medición del funcionamiento del proceso o la organización[24, 25]. Otros términos están vinculados con la gestión, como es el caso de CPI (siglas de Continuous Process Improvement)[26, 27], BPI (siglas de Business Process Improvement)[28, 29], TQM (siglas de Total Quality Management)[30, 31] y Six Sigma[32]. Las investigaciones en estas áreas tienen en común que los procesos son “empujados bajo el microscopio” con el objetivo de identificar posibles

mejoras. La minería de proceso puede considerarse una tecnología que contribuye a cada una de las áreas antes mencionadas [2, 33-35].

En los últimos diez años la tendencia común ha sido el uso de trazas o registros de eventos, propiciando esto que cada vez haya más sistemas de información utilizando técnicas de minería de procesos en sus soluciones. Algunos de los sistemas son: Comprehend (desarrollado por Open Connect) [36], Discovery Analyst (desarrollado por StereoLOGIC) [37], ARIS Process Performance Manager (desarrollado por Software AG) [38], Flow (desarrollado por Fourspark), Futura Reflect (desarrollado por Futura Process Intelligence), Process Discovery Focus (desarrollado por Iontas/Verint) [39].

Traza: Una traza de eventos σ es una secuencia de tareas originadas por una instancia ejecutada del proceso P en un Δt, es decir, $\sigma=<t_0, t_1,\ldots, t_n> \in T_{on}^{+}$. Las instancias del proceso P se inician con una tarea que denotaremos t_{ini} y termina con una tarea t_{fin} (t_{ini}, $t_{fin} \in T_{on}$). Se denota por $\sigma_i\ (k)$ a la k-ésima tarea de la traza σ_i. La representación de las trazas creada a partir de la alineación de las mismas constituye la base del algoritmo que se propone en este trabajo. En la figura que se muestra a continuación las trazas están constituidas por la descripción del evento, el tipo de evento, el usuario que llevó a cabo el evento y la fecha en que se realizó.

Caso 1

Descripción	Evento	Usuario	yyyy/mm/dd hh:mm
	inicio	ryzquierdo	10/06/2011 10:50
Registrarse	procesando	ryzquierdo	10/06/2011 10:50
Registrarse	terminado	ryzquierdo	10/06/2011 10:50
Enviar cuestionario	procesando	ryzquierdo	10/06/2011 10:50
Evaluar	procesando	ryzquierdo	10/06/2011 10:50
Enviar cuestionario	terminado	ryzquierdo	10/06/2011 10:52
Recibir cuestionario	procesando	ryzquierdo	10/06/2011 10:52
Recibir cuestionario	terminado	ryzquierdo	10/06/2011 10:52
Evaluar	terminado	ryzquierdo	10/06/2011 10:52
Archivar	procesando	ryzquierdo	10/06/2011 10:52
Archivar	terminado	ryzquierdo	10/06/2011 10:52
	final		10/06/2011 10:52

Caso 2

Descripción	Evento	Usuario	yyyy/mm/dd hh:mm
	inicio	ryzquierdo	10/06/2011 10:58
Registrarse	procesando	ryzquierdo	10/06/2011 10:58
Registrarse	terminado	ryzquierdo	10/06/2011 10:58
Enviar cuestionario	procesando	ryzquierdo	10/06/2011 10:58
Evaluar	procesando	ryzquierdo	10/06/2011 10:58
Enviar cuestionario	terminado	ryzquierdo	10/06/2011 10:58
Recibir cuestionario	procesando	ryzquierdo	10/06/2011 10:58
Recibir cuestionario	terminado	ryzquierdo	10/06/2011 10:58
Evaluar	terminado	ryzquierdo	10/06/2011 10:59
Proceso de queja	procesando	ryzquierdo	10/06/2011 10:59
Proceso de queja	terminado	ryzquierdo	10/06/2011 10:59
Proceso de chequeo	procesando	ryzquierdo	10/06/2011 11:00
Proceso de chequeo	terminado	ryzquierdo	10/06/2011 11:00
Archivar	procesando	ryzquierdo	10/06/2011 11:00
Archivar	terminado	ryzquierdo	10/06/2011 11:00
	final		10/06/2011 11:00

Figura 1. Log de evento

Existen tres tipos de técnicas de minería de proceso, estas son: Descubrimiento, Conformidad y Mejora. La primera técnica, permite descubrir un modelo del proceso ejecutado a partir de un log de eventos. Para ello existen gran variedad de investigaciones desarrolladas en este sentido, entre los algoritmos se puede mencionar: Alpha [40-42], Genetic Miner [11], Heuristic Miner [43, 44], Transition System Miner [45], Fuzzy Miner [46].

El segundo tipo de técnica es el chequeo de conformidad, donde se compara un modelo de procesos existente con un log de evento del proceso al que pertenece. Este chequeo permite saber hasta qué punto se corresponde el log de evento con el proceso y viceversa. Existen diferentes tipos de modelos que pueden ser utilizados en este tipo de técnica de la minería de procesos, entre los que se encuentran, los modelos normativos, organizacionales, reglas y políticas de negocios, leyes, etc.

El tercer y último tipo de técnica corresponde a la mejora. Esta permite ampliar el conocimiento que se tiene del proceso así como la mejora del mismo. A partir de un modelo de proceso existente y el log de evento correspondiente al mismo proceso se detectan aspectos como cuellos de botella, niveles de servicio, tiempos de espera y ejecución, frecuencia de algún evento, entre otros. Estos aspectos pueden reflejarse en un nuevo modelo de proceso. Los modelos descubiertos a partir de la aplicación de técnicas de minería de proceso pueden ser representados utilizando diferentes notaciones, entre las más empleadas se encuentran las Redes de flujo de trabajo (basadas en las redes de Petri) [47-49].

1.2 Ausencia de información

Descubrir un modelo de procesos requiere interpretar el comportamiento reflejado en las trazas y la correspondencia con dicho modelo en un grupo de patrones de flujo de control. No todos los algoritmos interpretan de la misma forma las trazas y no manejan todos los patrones de flujo de control [11].

En el uso de la minería de proceso, el modelo descubierto es verdaderamente útil cuando la información de las instancias de dichos procesos, se encuentra reflejada en las trazas de la manera más exacta posible, de lo contrario se estaría en presencia de un caso de ausencia de información. La ausencia de información se entiende como la falta de una o varias tareas ejecutadas en las trazas asociadas a las instancias de los procesos. También se le denominan a este tipo de tareas, tareas invisibles.
La existencia o presencia de tareas invisibles puede inferirse a partir de determinados patrones o situaciones de comportamiento presentes en las trazas [11]. Existen diferentes casos bases donde se evidencia este tipo de tareas. Los casos son:

Situación de salto. Cuando se produce un salto de una o varias tareas en una situación de selección, podría estar implícita una tarea invisible. Aquí se muestra el comportamiento de la siguiente secuencia de tareas ABD, ACD, AD, donde el recuadro gris representa la tarea invisible.

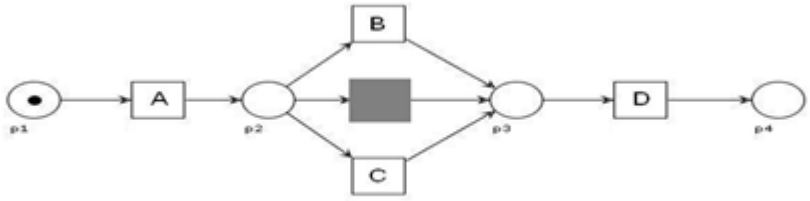


Figura 2. Red de workflow, muestra una situación de salto donde la tarea invisible se representa con el recuadro gris.

Situación de división/unión. Una tarea invisible también puede encontrarse en un caso donde sea necesario dividir o unir la ejecución del proceso, en este caso después de esta selección estamos en presencia de un paralelismo, que es la ejecución de dos o más tareas a la vez. Aquí se refleja el comportamiento registrado de la siguiente secuencia. Los recuadros grises representan las tareas invisibles.

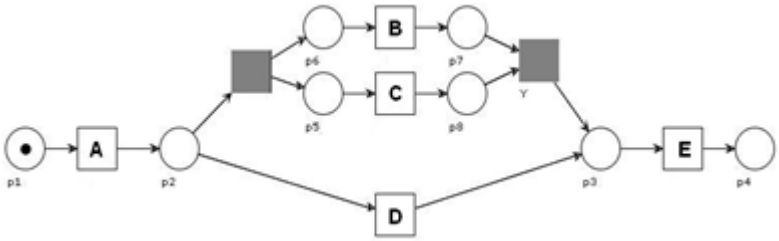


Figura 3. Red de workflow, muestra una situación de división/unión donde las tareas invisibles se representan con los recuadros grises.

Tareas invisibles contra tareas duplicadas. Una misma tarea puede aparecer más de una vez en una misma traza, la mayoría de los algoritmos desarrollados consideran que cada tarea solo aparece una sola vez en cada traza, ignorando las tareas duplicadas o las manejan a partir del constructor de lazos. Esta situación puede interpretarse también como ausencia de información (ver anexo 1).

Tareas invisibles contra lazos. Esta es una situación que se puede considerar como una unión de la situación de Tareas invisibles contra tareas duplicadas con la Situación de Salto (ver anexo 2).

Tareas invisibles contra sincronización. Esta situación puede considerarse como una generalización de la Situación de división/unión. Aquí se emplea el constructor de XOR-split/join y el constructor de tareas invisibles (ver anexo 3).

Lazos contra tareas invisibles junto a tareas duplicadas. Esta situación se puede interpretar como una unión de la Situación de salto con la Situación de división/unión. Los dos modelos pueden representar las mismas secuencias de tareas reflejadas en las trazas. Aquí se emplea el constructor de tareas invisibles junto al de tareas duplicadas y el constructor de lazos. El modelo descubierto usando el constructor de lazos es más expresivo, es decir, sobrepasa el comportamiento reflejado en la secuencia de tareas expuestas (ver anexo 4).

Sin embargo los algoritmos antes mencionados no son capaces de detectar las tareas invisibles en una traza y mostrarlas en el modelo descubierto. Por ejemplo, algoritmos como el Alpha no maneja el constructor de tareas invisibles, debido a que asume como precondición que las trazas están libres de ruido y completas. Obteniéndose así un resultado que no es real.

Secuencia oculta: Otro de los casos donde se manifiesta la ausencia de información es en la secuencia oculta. En este caso existe una secuencia entre las tareas de la traza, y existen tareas que no han sido registradas. Los siguientes ejemplos demuestran como la ausencia de información en los log de eventos afecta claramente el resultado de las aplicaciones de minería de procesos, incidiendo directamente en el modelo descubierto.

Al aplicar el algoritmo Alpha a un log de evento se puede demostrar claramente que no reconoce las tareas invisibles, pues en la Figura 4 a) la traza se encuentra supuestamente completa y el algoritmo lo asume como tal. Sin embargo por la estructura del log se puede suponer que hay ausencia de información, y al agregar tareas en cada uno de los casos se evidencia como varia el modelo descubierto Figura 4 b). Las tareas (X y Y) representan las tareas invisibles.

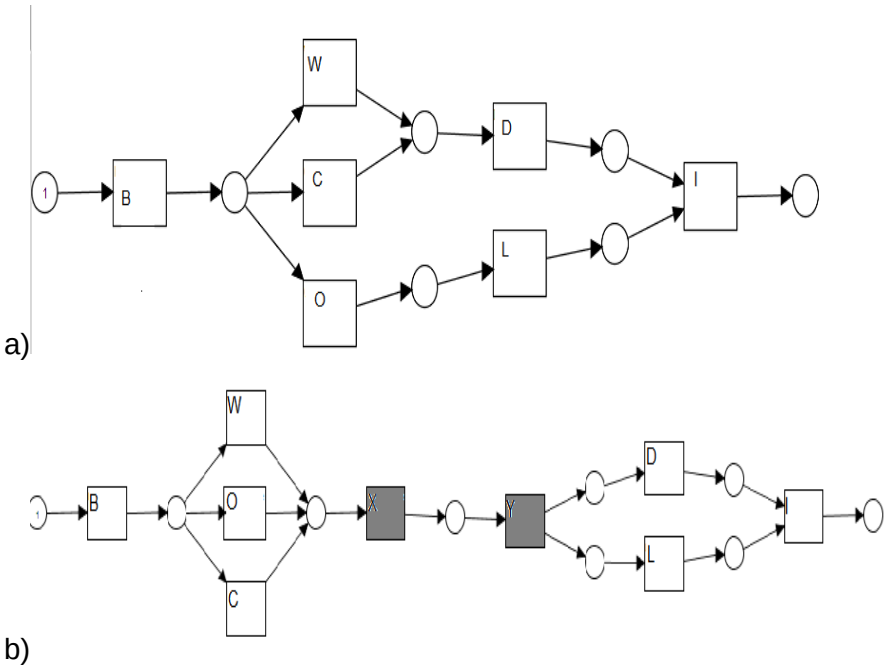
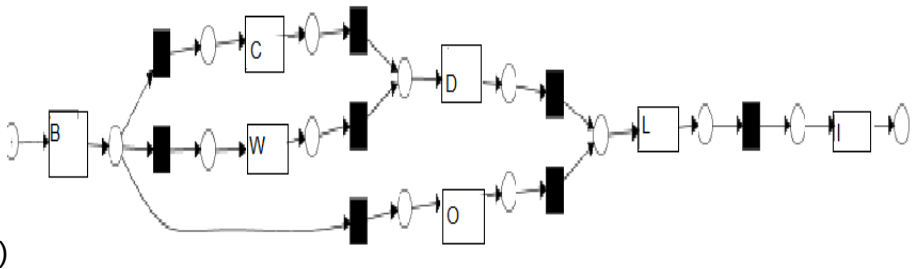
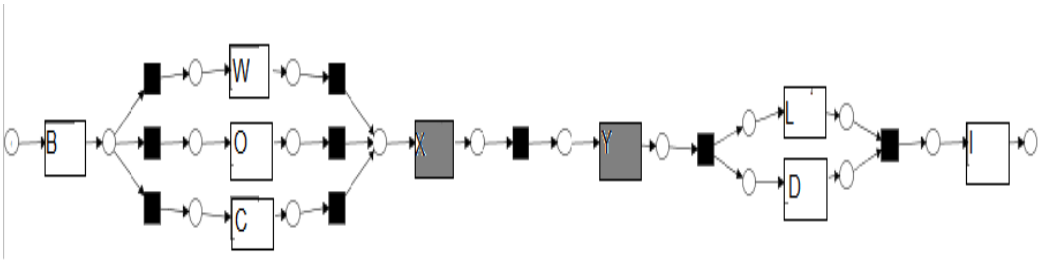


Figura 4. Red de workflow, en a) no se encuentra las tareas invisibles, y en b) los recuadros en gris representan las tareas invisibles. Algoritmo Alpha.

En el caso del Genetic Miner sucede algo similar. Al llevar a redes de Petri el resultado de este algoritmo, se puede obtener una representación gráfica del tratamiento de la traza. Evidenciando una vez más la importancia de la identificación correcta de las tareas invisibles. A la secuencia (BCODLI, BOCLDI, BODNI, BNDI, BCDNI) Figura 5 a) se le agregaron dos tareas (X y Y) (Tareas invisibles) en cada caso donde existe el supuesto que hay ausencia de información, demostrando la variabilidad de la información asociada a la traza y por ende la del modelo descubierto. Figura 5 b).

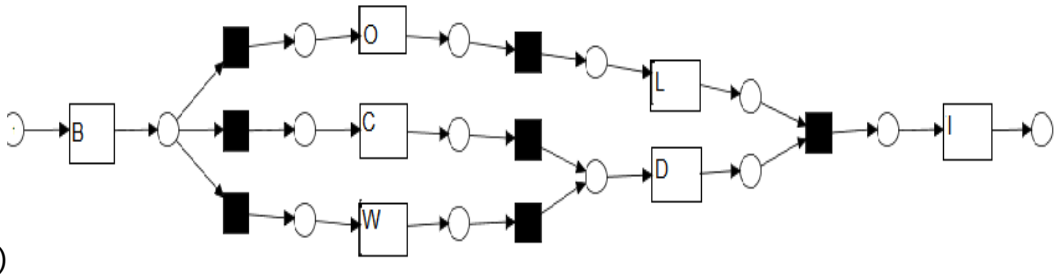




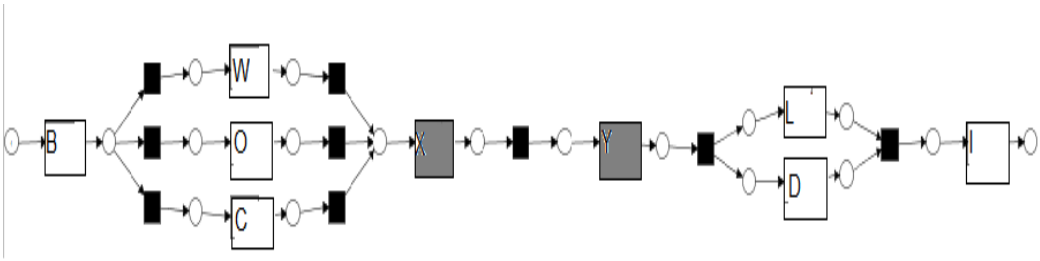
b)

Figura 5. Red de workflow, en a) no se encuentran las tareas invisibles, y en b) los recuadros en gris representan las tareas invisibles. Algoritmo Genetic Miner.

Los dos modelos de la Figura 6 representan la misma secuencia de tareas (BCODLI, BOCLDI, BODNI, BNDI, BCDNI) reflejadas en la traza. En la Figura 6 a) no se emplea el constructor de tareas invisibles, sin embargo al agregar las tareas (X y Y) en cada caso del log de evento, se puede apreciar que el modelo descubierto es mucho más expresivo y claro, Figura 6 b).



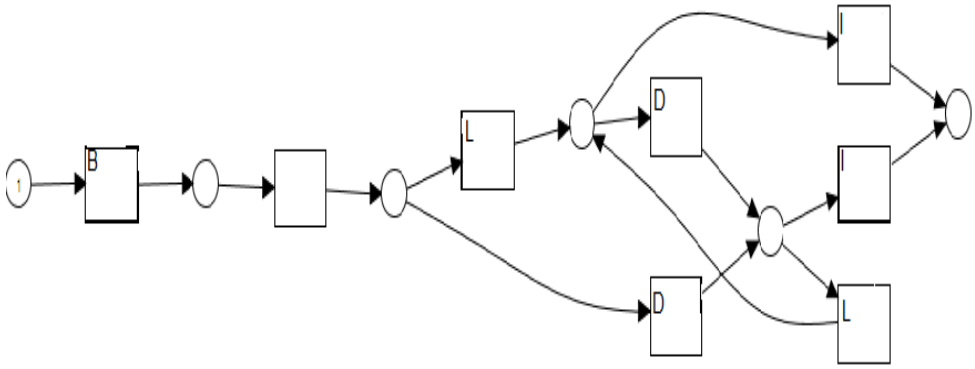
a)



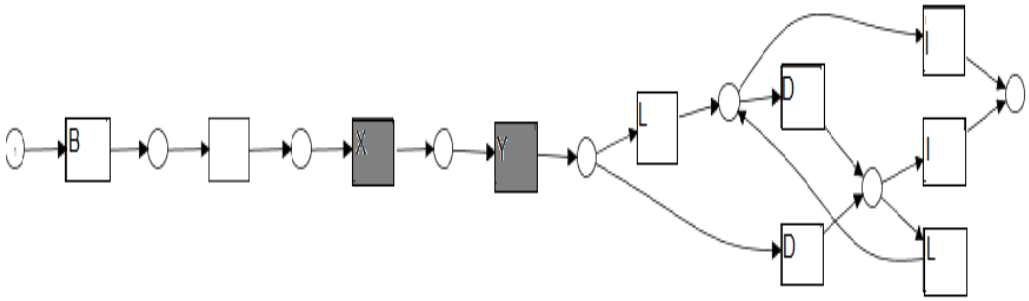
b)

Figura 6. Red de workflow, en a) no se encuentran las tareas invisibles, y en b) los recuadros en gris representan las tareas invisibles. Algoritmo Heuristic Miner.

En el algoritmo Transition System, se puede observar que al aplicarlo a la misma secuencia de tareas tampoco reconoce la ausencia de información en la traza. Al agregar a la secuencia de tareas de la traza la tarea invisible, se puede observar como el modelo descubierto varía, Figura 7 b), evidenciándose así la importancia que tiene poder identificar la ausencia de información en las trazas, para obtener resultados reales.



a)



b)

Figura 7. Red de workflow, en a) no se encuentran las tareas invisibles, y en b) los recuadros en gris representan las tareas invisibles. Algoritmo Transition System.

1.3 Bloques de construcción

Como punto de partida para la solución propuesta, se corresponden una serie de materiales que permiten entender la descomposición en bloques de construcción o subprocesos. Inicialmente se exponen un conjunto de definiciones que son necesarias para el correcto entendimiento de la propuesta.

Proceso de negocio: Un proceso de negocio es una colección de actividades que son realizadas coordinadamente en un ambiente técnico y organizacional. La conjunción de estas actividades logra un objetivo del negocio. Cada proceso de negocio es ejecutado por una simple organización, pero con él pueden interactuar procesos de negocios de otras organizaciones.

Subproceso: Un subproceso es una encapsulación de actividades del negocio que representan una compleja y lógica unidad de trabajo. Los subprocesos tienen sus propios atributos y metas, pero contribuyen a la meta del proceso que los contiene. Un subproceso es también un proceso y su mínima expresión es una actividad.

Un proceso puede descomponerse en varios subprocesos mediante los patrones de flujo de control siguientes:

• **Secuencia**: dos subprocesos se encuentran ordenados secuencialmente si inmediatamente después de que ocurra el primer subproceso ocurre el segundo.
• **Selección** (XOR u OR): dos subprocesos se encuentran ordenados como opciones de una selección si en cada caso o instancia del proceso solo ocurre uno de ellos (XOR) u ocurren los dos en cualquier orden (OR).
•**Paralelismo**: dos subprocesos se encuentran ordenados en paralelo si ocurren los dos simultáneamente.
• **Lazo**: un lazo se manifiesta cuando un subproceso se repite en múltiples ocasiones.
Los subprocesos pueden descomponerse en otros subprocesos hasta el nivel de actividad. Esto permite construir un árbol en el que cada nivel tiene menor grado de abstracción.

En la siguiente definición se formalizan los aspectos asociados a la descomposición de un proceso y su representación mediante bloques de construcción.

Bloque de construcción y descomposición en bloques de construcción: Sea S el conjunto de todos los subprocesos que componen a un proceso P, L el log de evento que representa a las instancias del

proceso P ejecutadas, A la matriz obtenida a partir de la alineación de las trazas contenidas en L y QA el conjunto de todas las sub-matrices de A.

La Figura 8 muestra un ejemplo de un árbol de bloques de construcción. Los bloques de construcción C2A y C3A representan subprocesos ordenados secuencialmente, C4A y C5A representan subprocesos ordenados como opciones de una selección. C6A y C7A representan subprocesos en paralelo.

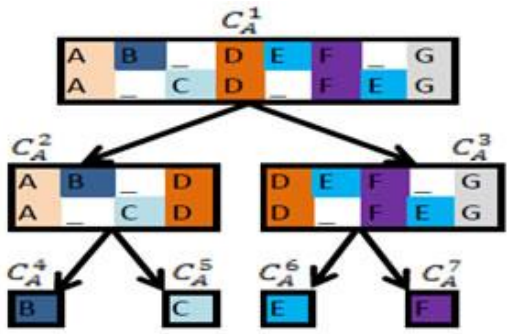


Figura. 8 Árbol de bloques de construcción.

Considerando los aspectos ya mencionados se presentan los pasos generales que conforman el algoritmo para determinar el árbol de bloques de construcción.

1 Alinear las trazas

El primer paso es el agrupamiento de las trazas y la alineación de las mismas según. Las trazas alineadas constituyen una representación de las actividades de acuerdo a un orden relativo y una estructuración de las mismas en casos. El orden establecido entre las actividades permite identificar los patrones de flujo de control que se manifiestan entre los subprocesos. Como resultado de la alineación de un grupo de trazas se obtiene una matriz A.

2 Pre-procesar las trazas alineadas

A partir de las trazas alineadas se pueden determinar los casos incompletos (casos que no terminan con las actividades finales identificadas para el proceso). Estos casos pueden ser tratados o eliminados según se considere, así se puede volver a alinear las trazas.

3 Determinar el árbol de bloques de construcción

En este paso se construye a partir de la matriz A un árbol de bloques de construcción que representan los subprocesos que componen al proceso analizado.

Los métodos que permiten determinar la descomposición mediante secuencia, lazo, XOR, paralelismo y secuencia oculta se describen a continuación.

Buscar secuencia: Los subprocesos ordenados secuencialmente pueden ser claramente identificados debido a que estos están separados por una o varias actividades (que aparecen de forma consecutiva) que aparecen ocupando una columna completa. Un ejemplo es la actividad D de la Figura 1, la cual permite distinguir la secuencia entre dos subprocesos representados por C2A y C3A. En ocasiones este tipo de actividades no se puede identificar debido a que pudieron no mapearse en el log de evento.

Buscar lazo: Para determinar un bloque de construcción que representa un subproceso que se repite en múltiples ocasiones es necesario identificar la actividad inicial de ese subproceso. Esta actividad inicial permitirá separar secuencias de actividades que posteriormente omitiendo las repeticiones conformarán las filas del nuevo bloque de construcción.

Buscar XOR: Para determinar los bloques de construcción que representan opciones de una selección se construyen conjuntos disjuntos con las actividades que componen el bloque de construcción Ci A. Inicialmente existe un conjunto que contiene las actividades que comparten una fila del bloque de construcción analizado, posteriormente se unen los conjuntos que se intersectan mediante alguna actividad. Si al finalizar este proceso queda más de un conjunto entonces se construyen los bloques de construcción que representan cada una de las opciones.

Buscar paralelismo: Para determinar los bloques de construcción que representan subprocesos en paralelo se construyen conjuntos disjuntos con las actividades que componen el bloque de construcción Ci A. Las actividades que pertenecen a conjuntos diferentes se encuentran en paralelo, mientras que las actividades que forman un mismo conjunto se relacionan mediante otro tipo de patrón de flujo de control. Si como resultado se obtiene más de un conjunto entonces los bloques de construcción que se forman a partir de estos representan subprocesos en paralelo.

Buscar secuencia oculta: Se determinan las posibles soluciones (variante de descomposición) considerando que existen columnas que no se mapearon en el log de evento y que permitían distinguir el fin de un subproceso y el inicio de otro que le seguía. Cada bloque de construcción que conforma una solución se puede descomponer mediante XOR, paralelismo o lazo. Las soluciones se evalúan y se selecciona la mejor, considerando en la evaluación que los bloques de construcción formados disminuyen la cantidad de lazos y paralelismos rotos (un ejemplo de un lazo roto se evidencia cuando una actividad aparece múltiples veces en una misma fila en el bloque de construcción analizado y en la solución propuesta no aparece como parte de un mismo bloque de construcción).

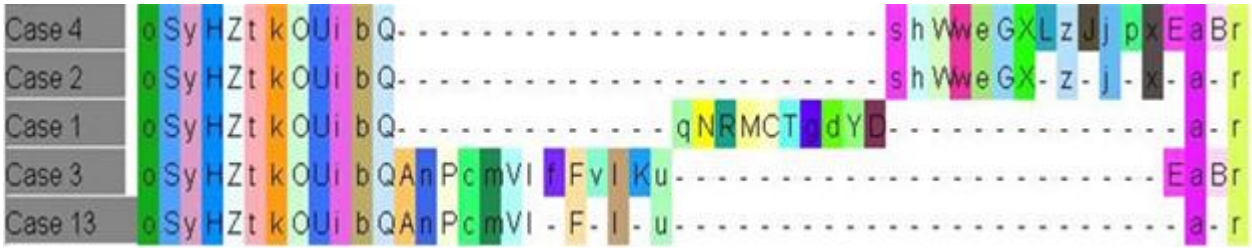


Figura 9. Alineación de trazas

A la matriz obtenida de la alineación (Figura 9) se le aplica el algoritmo 1 y se obtiene el árbol de bloques de construcción que se muestra en la Figura 10.

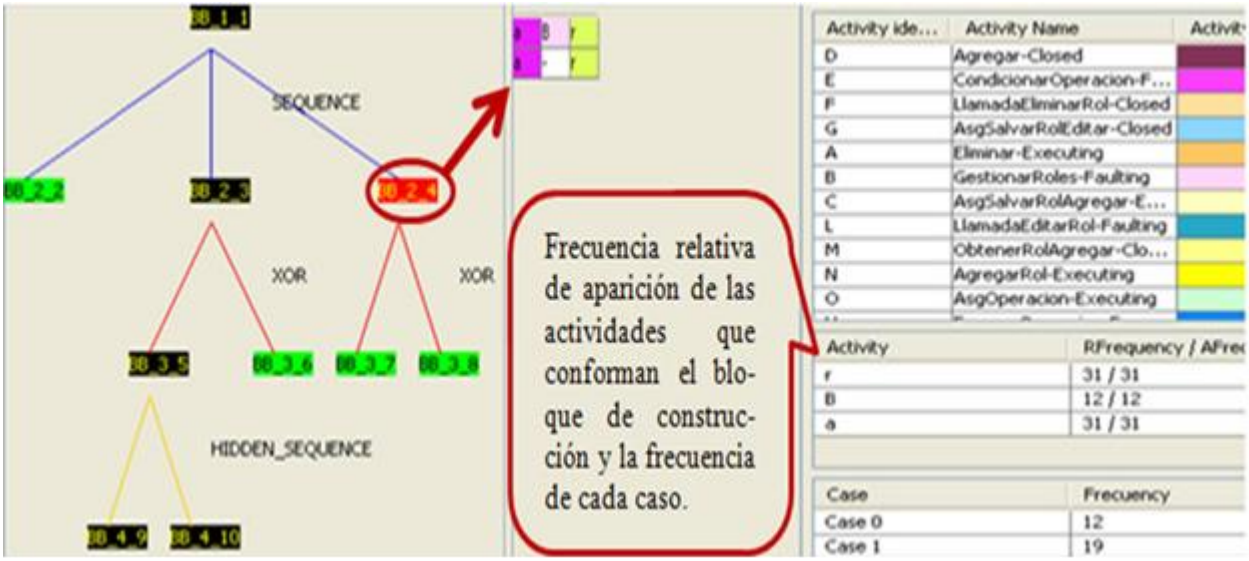


Figura 10. Árbol de bloques de construcción

1.4 Diferentes enfoques en el tratamiento de la ausencia de información en las trazas

En esta sección se hace un análisis del tratamiento de las tareas invisibles desde la perspectiva de los principales autores en el área de la minería de proceso. Algunas de las investigaciones analizadas se implementaron como algoritmos que forman parte de la herramienta ProM [50].

Cook et al. Los primeros trabajos en la minería de proceso fueron realizados por estos autores en el área del desarrollo de software. Cook participó en el desarrollo de tres algoritmos RNet, KTail y Markov, siendo este último el que mejor reflejaba las trazas analizadas. Markov es un algoritmo que tiene una base estadística y permite identificar los predecesores y sucesores de una tarea, para ello se apoya en una tabla de frecuencias en la que se refleja la probabilidad de que un evento ocurriese. Este algoritmo fue extendido posteriormente para manejar procesos concurrentes. Se agregaron cuatro métricas que permiten identificar puntos en los que aparecen relaciones XOR/AND-split/join. Este algoritmo es robusto ante el ruido debido a su basamento probabilístico [51-54]

Herbst et al. Uno de los aspectos más importantes del trabajo de estos autores es que sus algoritmos manejaban las tareas duplicadas. Herbst participó en el desarrollo de tres algoritmos: MergeSeq, SplitSeq y SplitPar. Los algoritmos ejecutan dos pasos, el primero permite capturar las dependencias entre las tareas apoyándose en el uso de la métrica LLH (siglas de log-likelihood) desarrollada. Esta métrica indicaba cuan bien el modelo expresado en SAG (siglas de Stochastic Activity Graph) expresaba el comportamiento recogido en las trazas. El segundo paso permite transformar el SAG a ADL (siglas de Adonis Defnition Language). El ADL es un lenguaje de estructuras en bloques que permite especificar modelos de flujos de trabajo. La conversión de SAG a ADL permite la creación de un modelo bien definido.

MergeSeq y SplitSeq son útiles para manejar procesos secuenciales, mientras que SplitPar puede manejar también procesos concurrentes. La implementación de los algoritmos desarrollados se realizó en la herramienta InWoLve[55-58].

Los algoritmos desarrollados pueden comportarse robustamente ante el ruido y manejan los constructores principales, secuencia, selección, paralelismo, lazos y tareas duplicadas. Sin embargo no se hace un tratamiento de las tareas invisibles de manera explícita. Se manejan las tareas invisibles en la Situación de salto y la Situación de selección[11].

Van der Aalst et al. Los autores fueron los desarrolladores del conocido algoritmo Alpha. El principal aporte fue que garantizaron una clase de modelo que permitía de manera efectiva el trabajo en el área. Esa clase de modelo se formalizó como SWF-nets (siglas de *Structured Workflow Nets*). El algoritmo Alpha se basa en cuatro tipos de relaciones binarias: *secuencial*, *causal*, *paralelismo* y *sin relación*. La relación de secuencia es la básica a partir de la cual se infieren las demás. Existe una secuencia cuando al menos en una instancia del proceso la tarea A es seguida directamente por B. La relación causal surge cuando A es seguida directamente por B, pero no siempre B es antecedida por A directamente. Si A es seguida por B y B es seguida por A entonces se establece una relación de paralelismo. Si A y B no están envueltos en una relación de secuencia entonces la relación es: sin relación.

El algoritmo Alpha no maneja tareas duplicadas y extensiones realizadas, permitiendo manejar correctamente lazos cortos, considerar tareas no atómicas, y de manera general mejoraron las relaciones binarias y las nociones de completitud de las trazas. Hay que resaltar que el algoritmo mina las trazas a partir de la concepción de que las trazas son completas y libres de ruido[40, 48, 49, 59, 60].

Weijters et al. La investigación de estos autores puede verse como una extensión del algoritmo Alpha. Sin embargo para establecer las relaciones de secuencia se emplea la frecuencia de aparición de dichas relaciones en las trazas analizadas. La implementación de dicho algoritmo se realizó en la herramienta Little Thumb y también en ProM como el plug-in Heuristics miner. El algoritmo es robusto ante el ruido[44, 61, 62].

Wen et al. El trabajo de estos autores estuvo dirigido a la extensión del algoritmo Alpha. Las primeras mejoras se realizaron en el algoritmo Beta, el cual fue implementado como el plug-in Tsinghua alpha en la herramienta ProM[50]. Lo distinguía el tratamiento de las tareas no atómicas y el uso del tiempo de ejecución de las tareas para determinar las relaciones de paralelismo y los lazos cortos. Una posterior mejora se implementó como el algoritmo Alpha++, este incluía otras mejoras como el tratamiento del constructor de no-libre-selección (en inglés non-free-choice). El algoritmo no es robusto ante el ruido[63, 64].

Medeiros et al. El principal aporte de estos autores estuvo en la aplicación de algoritmos genéticos en el área de la minería de proceso. El resultado de la aplicación de este tipo de técnica garantizó el manejo de todos los constructores estructurales en el caso del algoritmo GA, exceptuando las tareas duplicadas. El algoritmo DGA es una extensión de GA que permite manejar las tareas duplicadas. El principal inconveniente de estos algoritmos es el tiempo de ejecución. Los algoritmos son robustos ante el ruido [11].

Las pruebas realizadas utilizando la implementación realizada sobre ProM[50] arrojaron en cada uno de los ejemplos antes expuestos que el modelo obtenido puede interpretarse satisfactoriamente. Se puede inferir que existe ausencia de información, aunque esto no se hace en ningún caso de manera explícita aun cuando existe evidencia que la denota.

Rubin et al. Estos autores aplican técnicas de minería de proceso en el área de desarrollo de software. La fuente de información en este caso no fueron las trazas registradas por los sistemas, sino los documentos e información de manera general almacenadas durante el proceso de desarrollo de software en los repositorios de información. Los autores introducen como resultado de la minería de las trazas un modelo en TS (Transition System).

El algoritmo desarrollado se estructura en dos pasos. En el primer paso se genera el modelo en TS. Para obtener el modelo se transita por una fase de pre-procesamiento en la que se estructuran las trazas necesarias para el proceso de minería. Posteriormente se definen los posibles estados y transiciones derivados de los eventos almacenados en las trazas y se construye el modelo. A dicho modelo se le aplican un grupo de estrategias de modificación que permiten obtener un modelo que cubre el comportamiento expresado en las trazas y elimina los lazos. Un segundo paso permite generar una Red de Petri a partir del modelo en TS ya descubierto, este paso se basa en la teoría de la región. El algoritmo es capaz de manejar diferentes niveles de abstracción lo cual facilita la adaptabilidad del mismo. El algoritmo no es robusto ante el ruido [65].

Tabla1. Refleja los casos en los que los algoritmos analizados se distanciaron más de una correcta interpretación.

Algoritmo	Situación de salto	Situación de selección	Tareas invisibles contra tareas duplicadas	Tareas invisibles contra lazos	Tareas invisibles contra sincronización	Lazos contra tareas invisibles junto a tareas duplicadas	Secuencia oculta
Heuristics miner							*
(Weijters		*		*	*	*	

et al)							
Multi-phase							
(Dongen et al)					*	*	
Alpha							*
(Van der Aalst et al)	*	*	*		*	*	
Alpha ++							
(Wen et al)	*	*			*	*	
DWS mining							

1.5 Técnicas de metaheurística

Para la implementación de los algoritmos propuestos y la identificación de la secuencia oculta en las trazas asociadas, se propone utilizar técnicas de metaheurística en su solución. Los algoritmos metaheurísticos son utilizados frecuentemente para resolver problemas de optimización y búsqueda. Estos problemas, que son cada vez más complejos requieren de algoritmos cada vez más eficientes, con el objetivo obtener una solución aceptada y disminuir el esfuerzo computacional.

Los algoritmos exactos tienen la ventaja que siempre van a ofrecer como solución el óptimo global, pero su debilidad está, en que los requisitos de memoria y tiempo de ejecución crecen exponencialmente junto con el tamaño del problema. Sin embargo los algoritmos metaheurísticos tienen como características, que

ofrecen soluciones de buena calidad (el óptimo global en muchos casos) en un tiempo razonable sin importar la complejidad de dicho problema.

Existen una gran variedad de algoritmos metaheurísticos, estos han demostrado ser verdaderamente eficientes en temas de optimización. Entre los algoritmos más conocidos y utilizados se pueden mencionar los siguientes.

La Búsqueda Tabú: Es una de las metaheurísticas más utilizadas en problemas de optimización. Esta se basa fundamentalmente en la utilización de un historial de búsqueda, que permite ejecutar su estrategia de análisis y exploración de diferentes regiones del espacio de búsqueda. Este historial o memoria se implementa como una lista tabú. En cada iteración se elige la mejor solución entre las permitidas y se añade a la lista tabú, donde se mantienen las soluciones recientes que se excluyen de las siguientes iteraciones. [66]

La Búsqueda en Vecindario Variable: Este algoritmo es muy genérico, con muchos grados de libertad y permite variaciones y modificaciones particulares. Utiliza una estrategia de cambio entre diferentes estructuras del vecindario. Estas estructuras se definen en el comienzo del proceso algorítmico. [66]

El Recocido Simulado: Es una de las metaheurísticas más antiguas. En cada iteración se elige una solución S1, a partir de la solución actual S0. Si S1 es mejor que S0, S1 sustituye a S0 como solución actual. Si S1 es peor que S0, se continúa aceptando pero asignándole una determinada probabilidad. El algoritmo permite elegir soluciones peores a la actual para evitar caer en un óptimo local. [66]

Los Algoritmos Evolutivos: Este grupo de técnicas se inspiran en la capacidad de la evolución de seres o individuos para adaptarlos a los cambios de su entorno. Cada individuo representa una posible solución. El inconveniente de este algoritmo es que es muy costoso en cuanto a tiempo de ejecución, ya que ejecuta cada una de las fases (Selección, Reproducción, Mutación) a muchos individuos a la vez. [66]

La Búsqueda Dispersa: Se basa en mantener un conjunto relativamente pequeño de soluciones, conjunto de referencia, que contiene buenas soluciones y otras soluciones diversas. A los diferentes subconjuntos de soluciones que se forman se les aplican operaciones de recombinación y mejora. [66]

Los sistemas basados en Colonias de Hormigas: Se inspiran en el comportamiento de las hormigas cuando buscan comida. Inicialmente, las hormigas exploran el área cercana al hormiguero de forma aleatoria. Cuando una hormiga encuentra comida, la lleva al hormiguero. En el camino, la hormiga va depositando una sustancia que guía al resto de hormigas a encontrar la comida. Esta traza sirve a las hormigas para encontrar el camino más corto entre el hormiguero y la comida. Este rastro es simulado mediante un modelo probabilístico. Este algoritmo se vuelve altamente costoso ya que si el área de búsqueda es considerablemente grande, el tiempo de ejecución aumenta, haciendo al algoritmo ineficiente. [66]

Sin embargo para la propuesta se ha seleccionado el algoritmo de Recosido Simulado ya que esta metaheurística ha demostrado ser una herramienta verdaderamente exitosa para resolver una amplia gama de problemas de optimización combinatoria. El Recosido Simulado es una variante de la búsqueda local que permite movimientos ascendentes para evitar quedar atrapado permanentemente en un óptimo local. [67]. Además es un algoritmo que consume poco tiempo de ejecución ya que evalúa una propuesta a la vez en el espacio.

La mayoría de los algoritmos metaheurísticos son inspirados en situaciones o casos de la vida, incluso comportamientos naturales. El recocido simulado está basado a partir del proceso de enfriamiento de los metales, que recibe el nombre de “recocido”. Se hace una relación entre los elementos del algoritmo en cuestión y su inspiración en la termodinámica (ver anexo 5).

1.6 Técnica de programación dinámica

Para la implementación de los algoritmos propuestos en la estimación de la información ausente de las trazas utilizadas en la minería de proceso, se ha hecho un estudio de la técnica de programación dinámica.

Esta técnica es utilizada fundamentalmente en problemas de optimización, ya que puede mejorar sustancialmente el tiempo de ejecución. En este tipo de problemas se pueden presentar distintas soluciones y lo que se desea es encontrar la solución con el valor óptimo. La solución de problemas mediante esta técnica se basa en el llamado principio de óptimo, enunciado por Bellman en 1957. Para que un problema pueda ser resuelto con la técnica de programación dinámica, debe cumplir con ciertas características:

- Naturaleza secuencial de las decisiones: El problema puede ser dividido en etapas. Cada etapa tiene un número de estados asociados a ella.
- La decisión óptima de cada etapa depende solo del estado actual y no de las decisiones anteriores.
- La decisión tomada en una etapa determina cual será el estado de la etapa siguiente.

En síntesis, la política óptima es de un estado *S* de la etapa *k* a la etapa final está constituida por una decisión que transforma S en un estado S_0 de la etapa *k +1* y por la política óptima desde el estado S_0 hasta la etapa final.[68]

1.7 Conclusiones del capitulo

En lo expuesto en este capítulo se puede apreciar la importancia que tiene para la minería de proceso la identificación de la información ausente en las trazas. Aquí se abordaron diversos algoritmos que no son capaces de reconocer las tareas invisibles de un log de evento, ya que parten del supuesto de que estos logs están completos.

Es importante destacar que la presencia de tareas invisibles en las trazas afecta directamente al modelo descubierto, provocando relaciones erróneas entre las actividades del proceso, trayendo como consecuencia que se obtengan resultados distintos de la realidad.

Esto ha sido demostrado mediante el análisis de los resultados de cada uno de los algoritmos antes expuestos. Debido a este estudio se puede llegar a la conclusión de que el modelo descubierto es sensible a cualquier variación en las trazas asociadas a los procesos.

CAPÍTULO 2: PROPUESTA DE SOLUCIÓN

En este capítulo se proponen nuevos algoritmos para la estimación de la información ausente en las trazas asociadas a subprocesos ordenados secuencialmente utilizados en la minería de procesos. Aquí se describe teóricamente la funcionalidad de cada uno de ellos, específicamente el caso de la secuencia oculta.

2.1 Algoritmos para la Estimación de la información ausente

Dada la propuesta es necesario definir ¿qué se entiende en este trabajo por Estimación de la información ausente?

Estimación de información ausente: Es el proceso mediante el cual se transforma el conjunto de trazas $T = \{T_1, T_2, \ldots, T_n\}$ en otro conjunto de trazas $\check{\mathbf{T}} = \{\check{T}_1, \check{T}_2, \ldots, \check{T}_n\}$ donde, $\check{T}_i \in (\Sigma \cup \Lambda)^+$ para $1 \leq i \leq n$ y $\Lambda = \{\lambda_1, \lambda_2, \ldots, \lambda_w\}$ es el conjunto de actividades invisibles estimadas, Σ es el conjunto de todas las actividades. $|T_i| \leq |\check{T}_i|$.

El grupo de trabajo de minería de proceso de la universidad, desarrolló un modelo para la estimación de la información ausente con el objetivo de contribuir a la mejora de la estructura y comprensión de los modelos de procesos descubiertos. Dicho modelo está compuesto por cinco módulos o componentes: Componente para la alineación de las trazas, Componente para pre-procesar la alineación, Componente para determinar el árbol de bloques de construcción, Componente para la aplicación de los operadores de ausencia de información y Componente para construir el registro de eventos con información estimada.

El modelo tiene como entrada un registro de evento y tiene como salida un registro de evento que incluye la información estimada, ambos registros de eventos con un formato XES o MXML. Los componentes se encuentran en el mismo orden en que fueron mencionados, en la figura 11 se refleja el esquema de dicho modelo. Los componentes más importantes son el Componente para determinar el árbol de bloques de construcción y el Componente para la aplicación de los operadores de ausencia de información, ya que ellos determinan:

- La descomposición del proceso analizado
- La identificación de las situaciones de ausencia de información
- La estimación de las actividades invisibles

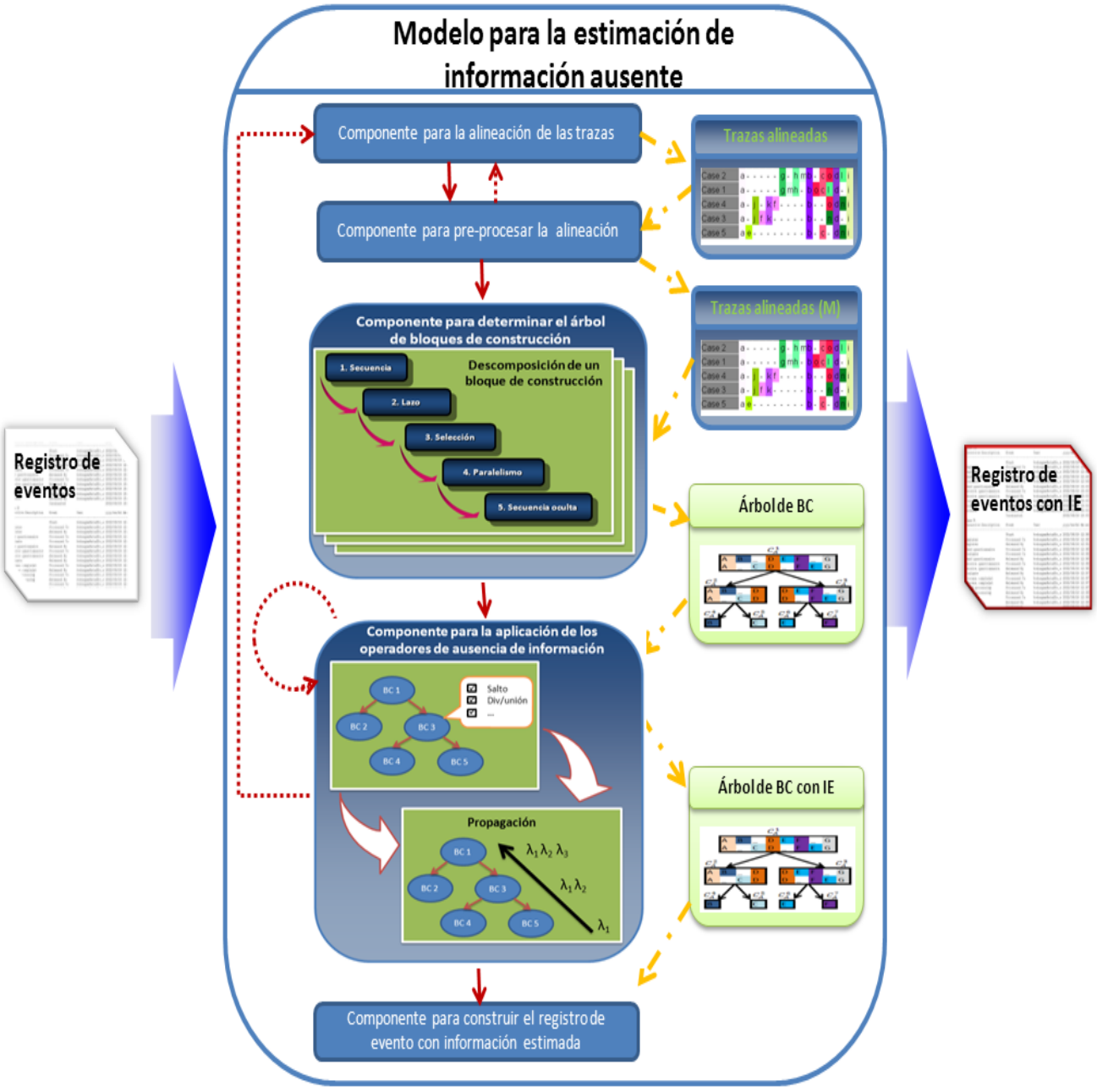


Figura 11 Esquema representativo del modelo.

En el desarrollo de la propuesta en cuestión, son muy importantes los componentes antes mencionados y abordados en la sección (1.3), debido a que las trazas deben ser alineadas y pre-procesadas con anterioridad, así como el Componente para determinar el árbol de bloques de construcción, que es la línea de partida del algoritmo para estimar la información ausente en las trazas utilizadas en la minería de proceso. A continuación se describen brevemente cada uno de estos componentes.

Componente para la alineación de las trazas: Este componente se encarga del agrupamiento de las trazas y su alineación. El agrupamiento de las trazas se puede realizar utilizando alguna de las técnicas desarrolladas en este sentido [23, 123, 133]. Como resultado de la alineación de un grupo de trazas se obtiene una matriz A.

Componente para pre-procesar la alineación: Este componente permite corregir algunos aspectos de la alineación de las trazas que pueden constituir un problema para los componentes que se aplican posteriormente. Aquí se separan las actividades en columnas independientes y continuas, ademas permite detectar los casos incompletos.

Componente para determinar el árbol de bloques de construcción: Este componente permite obtener un árbol de bloques de construcción que representa la descomposición jerárquica en subprocesos del proceso analizado. La matriz A' constituye la entrada de este componente. Este componente presenta un algoritmo del mismo nombre, donde se van a ejecutar cada uno de los operadores para buscar los diferentes casos de ausencia de información. Estos son: *Buscar lazo, Buscar paralelismo, Buscar XOR-OR* y *Buscar secuencia oculta.*

Componente para la aplicación de los operadores de ausencia de información: En este componente se aplican un conjunto de operadores definidos para el manejo de la información ausente. En dependencia de las características del bloque de construcción puede no aplicarse la totalidad de los operadores

propuestos. Estos son: *Operador de salto*, *Operador de lazo*, *Operador de división/unión*, *Operador de secuencia oculta* y *Operador probabilístico*.

Componente para construir el registro de evento con información estimada: Este componente tiene como objetivo construir el registro de evento que contiene la información estimada. El registro de evento modificado constituye la salida del modelo propuesto.

2.1.1 Algoritmo para determinar el árbol de bloques de construcción

En Algoritmo 1 se describen los pasos para determinar el árbol de bloques de construcción. Se emplea un árbol para la representación porque muestra la forma en la que el proceso se descompone en otros subprocesos en cada nivel.

Algoritmo 1 Algoritmo para determinar el árbol de bloques de construcción

Entrada: Matriz A'

Salida: Árbol de bloques de construcción

1: Crear un árbol vacío.

2: Crear un bloque de construcción C^1_A y asociarlo al nodo raíz del árbol la matriz A', tal que C^1_A = A'.

3: **Si** C^i_A no es una matriz con una sola fila **Entonces**

4: *LH* = *Buscar secuencia*(C^1_A). *LH* es la lista que almacena los bloques de construcción obtenidos producto a la descomposición realizada.

5: **Si** |*LH*| < 1 **Entonces**

6: *LH* = Buscar lazo(C^i_A)

7: **Si** |*LH*| < 1 **Entonces**

8: LH = Buscar XOR-OR(C^i_A)

9: **Si** |*LH*| < 1 **Entonces**

10: *LH* = Buscar paralelismo(C^i_A)

11: **Si** |*LH*| < 1 **Entonces**

12: *LH* = Buscar secuencia oculta(C^i_A)

FinSi

FinSi

FinSi

FinSi

13: **Para** cada bloque de construcción *i* contenido en la lista *LH* **Hacer**

14: Modificar *i*. Eliminándose de ser necesario, las filas repetidas y las columnas que solo contienen símbolos vacíos (“–”)

15: Adicionar *i* como hijo del nodo que contiene a C^1_A

16: Aplicar el Algoritmo para determinación del árbol de bloques de construcción al bloque de

construcción modificado comenzando por el paso 3

17: **Si** el árbol obtenido en el paso anterior ≠ Ø **Entonces**

18: Adicionar los nodos hijos de la raíz del árbol obtenido como hijos del nodo que contiene el bloque de construcción *i*

FinSi

FinPara

Sino

19: Devolver un árbol vacío

FinSi

20: Devolver el árbol de bloques de construcción construido

En la figura 12 muestra un ejemplo de un árbol de bloques de construccion obtenido al descomponer un proceso usando la implementacion del algoritmo antes enunciado.

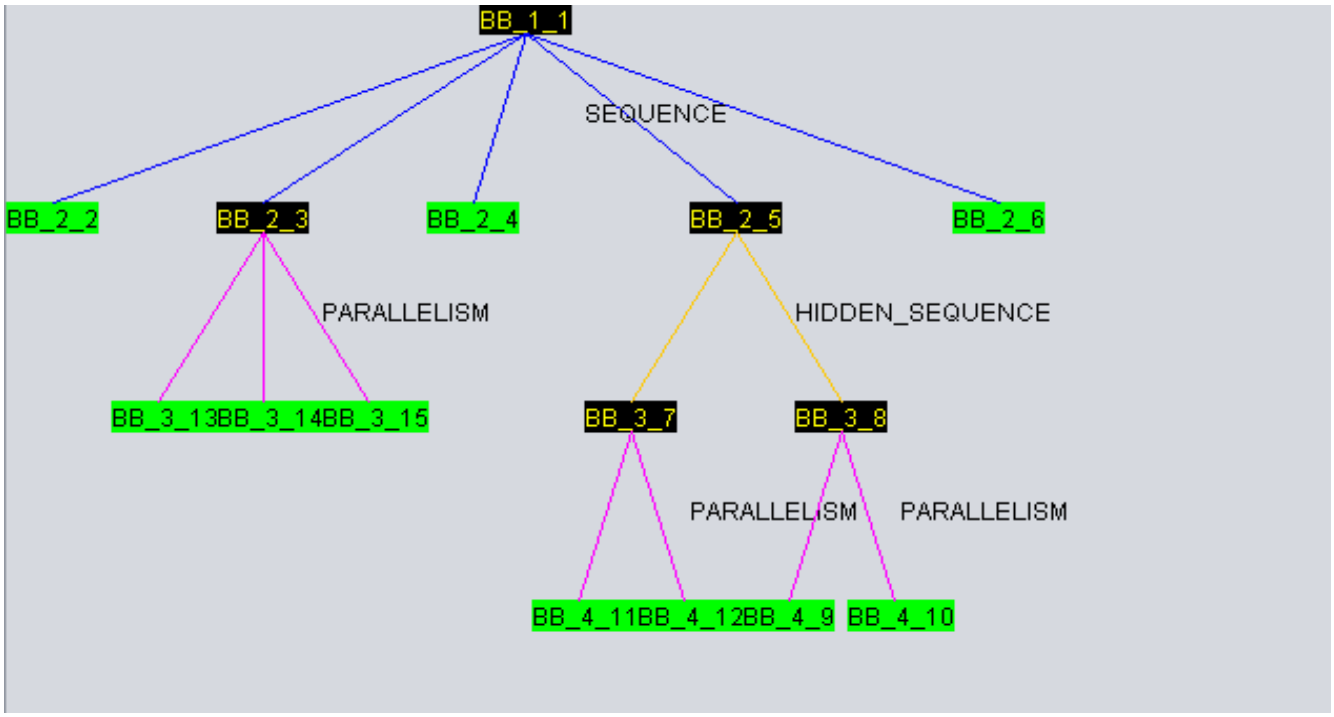


Figura 12 Árbol de bloques de construcción

2.1.2 Algoritmo para buscar Secuencia Oculta

El objetivo del *Algoritmo 2 Buscar secuencia oculta* es determinar si el proceso analizado se puede descomponer en otros subprocesos ordenados secuencialmente. En este algoritmo no existe una columna que no contenga símbolos vacíos (“–”) y permite delimitar el final de un bloque de construcción y el inicio del siguiente. Como resultado, se determinan las posibles soluciones (variantes de descomposición que representan subprocesos ordenados secuencialmente). Teniendo en cuenta los parámetros mencionados a continuación.

- Cada bloque de construcción que conforma una solución se puede descomponer mediante XOR, OR, paralelismo o lazo.

- Las soluciones se evalúan y se seleccionan las mejores, teniendo en cuenta la evaluación que los bloques de construcción formados disminuyen la cantidad de lazos y paralelismos rotos. Un lazo roto es cuando una actividad aparece más de una vez en una misma fila en el bloque de construcción analizado y en la solución propuesta no aparece como parte de un mismo bloque de construcción. De manera análoga se determina la cantidad de paralelismos rotos.

Algoritmo 2 Buscar secuencia oculta

Entrada: Bloque de construcción C^{i}_{A}

Salida: Lista de bloques de construcción

1: Crear una lista vacía *ListaHijos*. *ListaHijos* almacena los bloques de construcción obtenidos en una posible descomposición

3: Determinar el orden de precedencia entre las actividades que conforman el bloque de construcción. Una actividad *a* precede a otra actividad *b,* si en todas las filas, *a* está antes que *b*

4: Determinar grupos de actividades, donde las actividades que conforman un grupo no tienen orden de precedencia

5: Determinar el orden de precedencia entre los grupos. Un grupo *A* precede a otro grupo *B* si todas las actividades de *A* preceden a todas las actividades de *B*

6: Ajustar el bloque de construcción considerando los grupos y la precedencia establecida. Este paso garantiza que existe una coherencia entre la precedencia establecida entre las actividades y el orden de las columnas en el bloque de construcción

7: **Para** *i* = 1 **Hasta** *m* **Con Paso** 1 **Hacer**

8: B_i = β(1, i, C^i_A). Donde B_i es el valor asociado a la evaluación de los bloques de construcción que forman una secuencia oculta desde la primera columna hasta la columna i. β(x, y, BC) es un procedimiento que determina el valor de μ para el bloque de construcción formado por las columnas desde la x hasta la y del bloque de construcción BC.

9: **Para** j=2 Hasta i Con Paso 1 Hacer

10: **Si** $B_i > B_{j-1}$ + β(j, i, C^i_A) **Entonces**

11: $B_i = B_{j-1}$ + β(j, i, C^i_A)

12: $S_i = j$

FinSi

FinPara

FinPara

13: **Si** $B_m \neq \infty$ **Entonces**

14: $p = m$

15: **Mientras** $p > 1$ **Hacer**

16: Adicionar la sub-matriz que se obtiene desde las columnas S_p hasta la p a *ListaHijos*

17: $p = S_p - 1$

FinMientras

FinSi

18: Devolver *ListaHijos*

En la figura 13 se muestra un ejemplo de un proceso descompuesto en el que se evidencia la detección de una secuencia oculta. La descomposición se realizó mediante la implementacion del algoritmo 2.

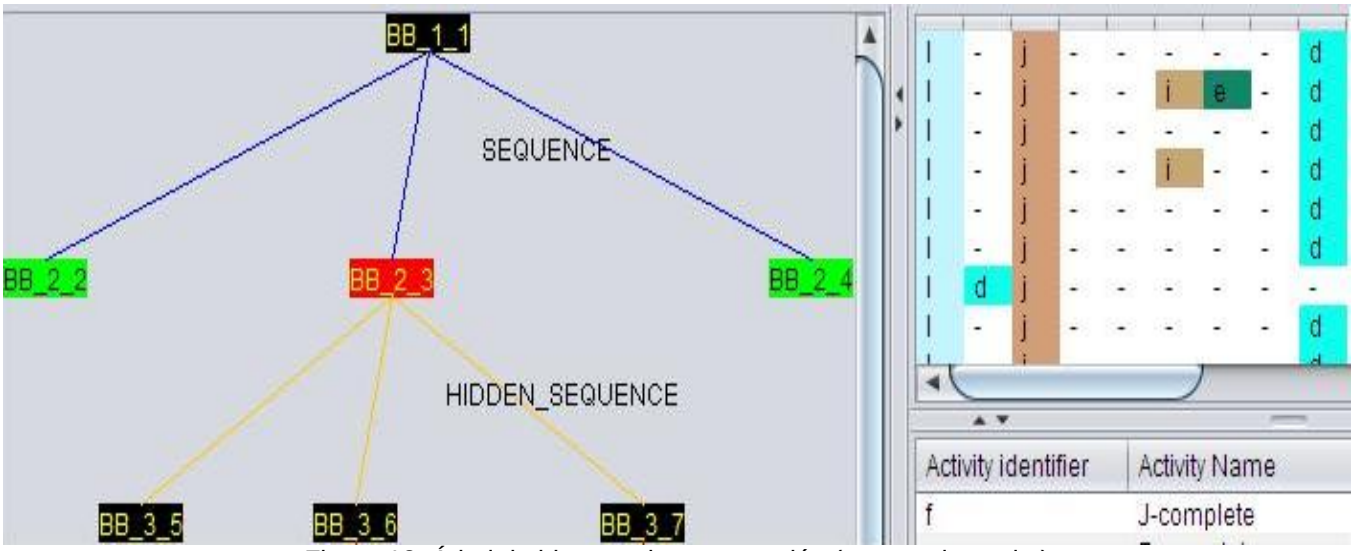


Figura 13. Árbol de bloques de construcción (secuencia oculta)

Algoritmo 3 Buscar secuencia oculta con metaheurística

Entrada: Bloque de construcción C^{i}_{A}

Salida: Lista de bloques de construcción

1: Crear una lista vacía ListaHijos. ListaHijos almacena los bloques de construcción obtenidos en una posible descomposición

3: Determinar el orden de precedencia entre las actividades que conforman el bloque de construcción. Una actividad *a* precede a otra actividad *b* si en todas las filas, a está antes que *b.*

4: Determinar grupos de actividades, donde las actividades que conforman un grupo no tienen orden de precedencia

5: Determinar el orden de precedencia entre los grupos. Un grupo A precede a otro grupo B si todas las actividades de A preceden a todas las actividades de B

6: Ajustar el bloque de construcción considerando los grupos y la precedencia establecida. Este paso garantiza que exista una coherencia entre la precedencia establecida entre las actividades y el orden de las columnas en el bloque de construcción.

7: Se selecciona una solución inicial S0
8: Se selecciona una temperatura inicial T0 tal que T0>0
9: Se selecciona una función de reducción de la temperatura t
10: Se selecciona un número de iteraciones Nrep
11: Se selecciona un criterio de parada
12: Sea f(S) el coste de las soluciones
13: **Para** i = 1Hasta m Con Paso 1 Hacer
14: **Para** j = i Hasta m Con Paso 1 Hacer
15: Se selecciona una solución S1
16: **Si** f (S1) < f (S0) **Entonces**
17: S0= S1
18: **SiNo**
19: Generar aleatoriamente una probabilidad u
20: **Si** u< Exp ((S1- S0)/T) **Entonces**
21: S0= S1
22: **FinSiNo**
23: **Si** cant_iteraciones = Nrep
24: **FinPara**
24: T=u (T)
25: **Si** Crit_Parada == Verdadero
26: **FinPara**
27: Devolver ListaHijos

2.2 Módulo para la aplicación de los operadores de ausencia de información

El operador aplicado modifica el bloque de construcción solo si detecta alguna actividad invisible. Para reflejar la información estimada se crea un nuevo árbol de bloques de construcción estimados Ĕ. En el caso específico de la secuencia oculta se aplica el operador para identificar la *Situación de secuencia oculta* y estimar la información ausente. A continuación se describe el operador de estimación de información ausente propuesto para este caso.

Algoritmo 3 Operador de secuencia oculta

Entrada: Bloque de construcción: *Ce*, Lista de bloques de construcción obtenidos al descomponer a *Ce*: *List*

Salida: Lista de bloques de construcción

1: **Si** |*List*| > 0 y *List*[1] se descompuso según la relación *Secuencia oculta* **Entonces**

2: Crear k actividades invisibles. k = (cantidad bloques de construcción hijos Ce) -1

3: Para *i* = 1 Hasta *k* Con Paso 1 Hacer

4: Insertar una nueva columna en la última posición del bloque de construcción *List*[*i*] y en cada fila de la columna insertada se pone la misma actividad invisible λ*i*

5: Insertar una nueva columna en la primera posición del bloque de construcción *List*[*i*+1] y en cada fila de la columna insertada se pone la misma actividad invisible λ*i*

FinPara

6: Devolver *List* modificada

FinSi

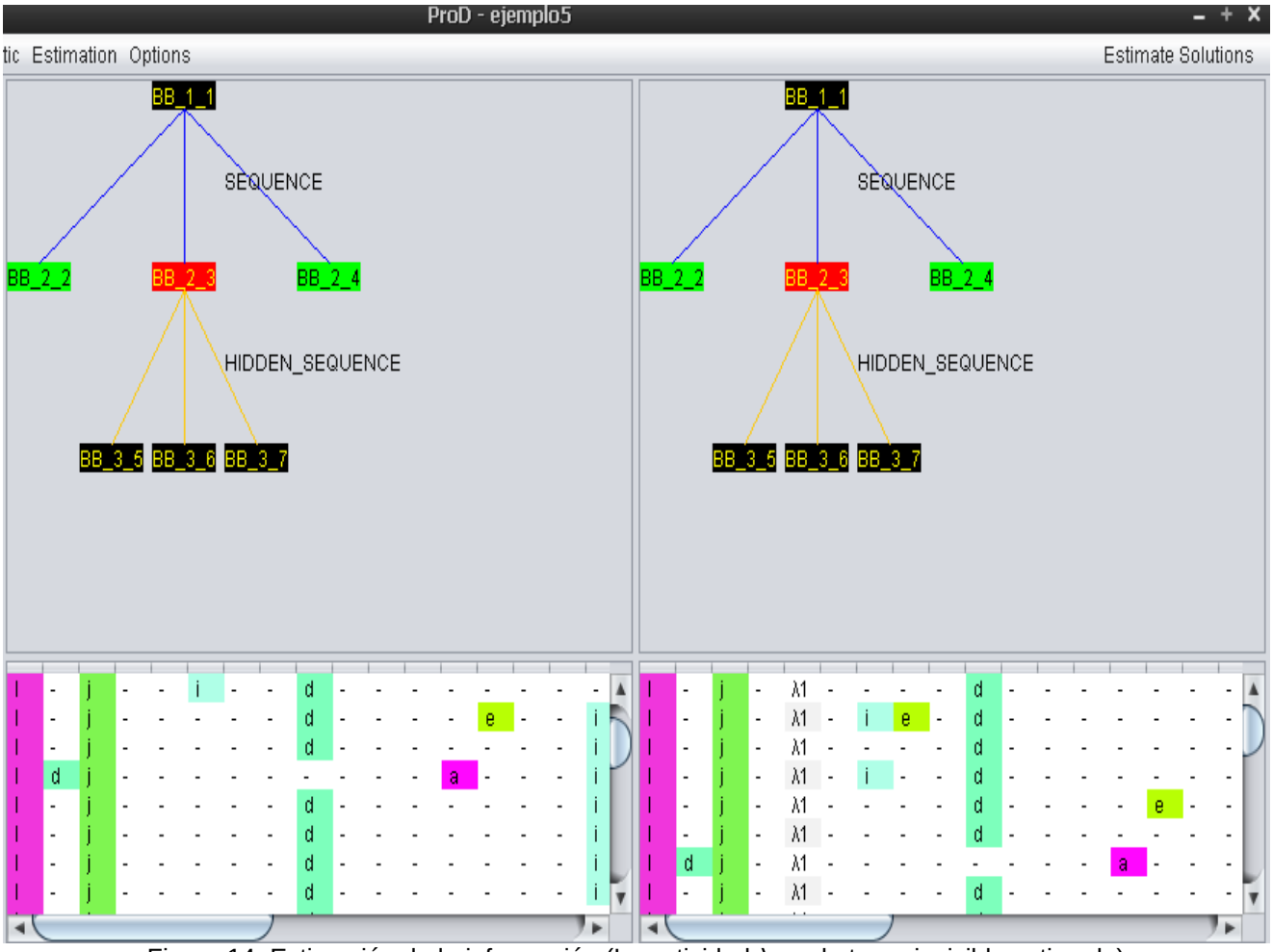


Figura 14. Estimación de la información (La actividad λ_1 es la tarea invisible estimada)

Al estimar la información ausente en el caso de la secuencia oculta, se genera un nuevo árbol de bloques de construcción y la matriz que representa a dicho bloque muestra la actividad invisible estimada, denotada por λ_1 como se puede apreciar en la figura 14.

2.3 Conclusiones del capítulo

En este capítulo se mencionan los módulos por los que está compuesto el modelo para la estimación de la información ausente, principalmente el módulo para determinar el árbol de bloques de construcción así como la descripción del algoritmo que lo compone. De aquí se derivan los operadores para buscar los diferentes casos de ausencia de información mencionados anteriormente en este trabajo, describiendo principalmente los algoritmos asociados a la estimación de la información ausente en el caso de la secuencia oculta.

CAPÍTULO 3: VALIDACIÓN DE LA SOLUCIÓN

En el presente capítulo se registran cada uno de los resultados arrojados por el proceso de validación. Para esto se efectúan pruebas a cuatro procesos diferentes y se aplican los algoritmos antes propuestos para estimar la información ausente en los mismos.

3.1 Aplicación informática para la estimación de información ausente

La herramienta informática desarrollada ProD, permite estimar la información ausente a partir de los log de eventos utilizados en la minería de proceso. El sistema hace uso de los registros de eventos en formato MXML o XES como entrada. Como salida se obtiene un registro con la información estimada en el formato MXML o XES en correspondencia con el formato del registro de entrada.

La aplicación desarrollada recorre tres momentos importantes en la estimación de información ausente en las trazas utilizadas en la minería de proceso.

- En un primer momento la aplicación permite leer un registro de eventos y configurar los parámetros necesarios para la alineación de las trazas.
- Posterior a la alineación de las trazas se realiza la descomposición del proceso. Aquí es donde se conforma el árbol de bloques de construcción.
- En un último momento se realiza la estimación de información ausente a partir del árbol de bloques de construcción obtenido anteriormente.

Para la estimación de la información ausente en el sistema ProD se muestra una ventana con tres paneles, en la parte superior se encuentra un menú que permite salvar la solución obtenida de un determinado registro de eventos.

En el panel de la izquierda se muestra el identificador del árbol en la parte superior y cada uno de los operadores que se pueden seleccionar para la estimación de la información en la parte inferior. En el

panel del centro se muestra el árbol de bloques de construcción referente al registro cargado, además de la leyenda y los detalles del mismo que se encuentran en la parte inferior.

En el panel de la derecha se refleja el árbol de bloques de construcción estimado y la información de cada una de las actividades invisibles que fueron agregadas producto de la estimación, así como la causa por la que fueron insertadas.

Figura 15 Estimación de información ausente.

La figura 15 muestra lo explicado anteriormente para el caso específico de Secuencia Oculta. Las actividades invisibles estimadas son denotadas en este caso por λ_1 y λ_2.

3.2 Resultados experimentales

Se definió un experimento para probar la certeza de los algoritmos desarrollados y para ello se emplea la aplicación informática antes expuesta.

Se debe demostrar que la aplicación propuesta es capaz de obtener modelos de procesos mejor estructurados y más comprensibles. Además es necesario definir la forma en que se debe medir la estructura y comprensión de los modelos descubiertos.

La ausencia de información en el registro de evento trae como consecuencia que en el modelo descubierto se establezcan incorrectas relaciones entre las actividades. Por eso es necesario medir el grado en el que el modelo descubierto representa el comportamiento observado en el registro de eventos. Este tipo de verificaciones se enmarcan en el área de Chequeo de conformidad [17, 19, 69, 70].

3.2.1 Diseño experimental

Para el experimento se definen cuatro grupos de procesos y tres momentos. Cada grupo está asociado a un proceso de negocio. Los momentos están relacionados a las etapas por las que transita un proceso, es decir, la evaluación a partir de un registro de eventos en su estado original, con ausencia de información y con información estimada. La tabla 2 muestra el diseño experimental realizado y la simbología utilizada es la siguiente:

- G: Grupo de participantes. Asociados a cada momento (Original, Ausencia de Información e Información estimada).
- R: Asignación al alzar.

- X: Tratamiento o estímulo. En este caso X_1 se corresponde con la extracción de información del registro de eventos analizado en cada grupo durante el primer momento. X_2 se corresponde con la aplicación del modelo propuesto para la estimación de información ausente.
- O: Observación.

Tabla 2 Diseño experimental propuesto

	Original		Ausencia de información		Información estimada
R G_1	O_1	X_1	O_2	X_2	O_3
R G_2	O_4	X_1	O_5	X_2	O_6
R G_3	O_7	X_1	O_8	X_2	O_9
R G_4	O_{10}	X_1	O_{11}	X_2	O_{12}

En el primer momento (Original) las observaciones O_1, O_4, O_7 y O_{10} representan la evaluación del registro de eventos en su estado original. En el segundo momento (Ausencia de información) las observaciones O_2, O_5, O_8 y O_{11} están asociadas a la evaluación después de extraer información del registro de eventos original (X_1).

En el último momento (Información estimada) las observaciones O_3, O_6 y O_9, O_{12} están asociadas a la evaluación de aplicar el modelo propuesto (X_2). Cada registro de eventos con ausencia de información se transformó usando la aplicación informática desarrollada y se obtuvo un nuevo registro de eventos con la información estimada.

3.2.2 Características de los procesos analizados

Se utilizaron para realizar las pruebas cuatro procesos que fueron generados con la aplicación informática Process Log Generator en la versión 1.4 beta [75]. Se decidió utilizar una herramienta que generara de manera aleatoria un registro de eventos artificial, debido a que, en el área de minería de proceso no se detectó una base de datos que contenga procesos que puedan ser utilizados para la validación de las técnicas desarrolladas. Existen algunos registros de eventos asociados a procesos reales pero estos no cuentan con las características adecuadas para realizar una correcta validación de la propuesta. Los principales problemas están relacionados con el reflejo parcial de los patrones de flujo de trabajo y la cantidad de casos. Además no se conoce el proceso original de estos registros de eventos, debido a esto no es posible saber qué actividades pueden estar faltando o cuánto ruido pudo manifestarse.

Un registro de eventos generado artificialmente puede reflejar los patrones de flujo de control conocidos, tener diferentes niveles de patrones anidados y la cantidad de casos puede variar en correspondencia con los elementos enunciados. En la Tabla 3 se muestran las características de cada uno de los procesos generados aleatoriamente, para obtener estos datos fue utilizada la herranienta ProM.

Tabla 3 Descripción de los registros de eventos

	Proceso 1	Proceso 2	Proceso 3	Proceso 4
Cantidad de casos	500	1000	500	500
Eventos	11000	17304	9668	13240
Clases de eventos	22	26	36	36
Cantidad de patrones anidados	2	2	3	3
Refleja patrones de flujo de trabajo: Secuencia, AND Split/join, XOR Split/join	Si	Si	Si	Si

Refleja el patrón de flujo de trabajo Lazo	Si	Si	Si	No

Para la conformación de los registros de eventos se fueron incrementando las variables, cantidad de casos, cantidad de patrones anidados y la probabilidad de reflejar los patrones de flujo de trabajo. La complejidad del descubrimiento aumenta en la medida en la que se manifiestan, en el registro de eventos.

3.2.3 Análisis de los resultados

Para realizar el experimento, en primer lugar se generaron de manera aleatoria cada uno de los procesos mencionados anteriormente. Con la aplicación propuesta ProD, se efectúa la alineación de las trazas, este paso corresponde al primer momento de los antes mencionados. En este instante se decide cuales actividades extraer para provocar la ausencia de información, en cada proceso individualmente.

A continuación, utilizando la herramienta ProM se sustraen las actividades aleatoriamente y se genera un fichero con las características del registro en cada uno de los momentos, en este caso el log de eventos original y el que presenta la ausencia de información. Este paso está asociado al segundo momento antes mencionado.

El próximo paso corresponde a estimar la información ausente con el sistema propuesto, por último se pasa a evaluar los resultados de la estimación de cada uno de los procesos, comparándolos con el registro de eventos original al que pertenecen. Esta parte está asociada al tercer momento antes explicado.

Luego de haber transitado por los tres momentos mencionados, se utilizó para el análisis la siguiente fórmula: (*1- I/T*) donde *I* es la cantidad de tareas incorrectas y *T* es el total de tareas estimadas. De esta manera es posible saber la efectividad de descubrimiento y de estimación de actividades invisibles de los algoritmos propuestos para el caso de secuencia oculta.

Las tareas incorrectas son aquellas tareas que fueron estimadas de más, no fueron estimas o aquellas que han sido colocadas en una posición incorrecta en el registro de eventos estimado, en comparación con el registro original.

En la Tabla 4 se muestra la relación de estas variables para cada uno de los registros de eventos evaluados con los dos algoritmos propuestos, para la estimación de información ausente en el caso de secuencia oculta.

Tabla 4 Resultados del experimento

Tabla 4 Resultados del experimento

Secuencia Oculta (Programación Dinámica)				Secuencia Oculta (Metaheurística)		
Procesos	**Tareas Incorrectas**	**Total de Tareas**	**AVG (1-I/T)**	**Tareas Incorrectas**	**Total de Tareas**	**AVG (1-I/T)**
Proceso 1	0	1	1	0	1	1
Proceso 2	0	1	1	0	1	1
Proceso 3	0	1	1	0	1	1
Proceso 4	0	1	1	0	1	1

La tabla 4 muestra los resultados obtenidos luego de aplicar los algoritmos propuestos a los procesos generados anteriormente, se puede observar que ambos algoritmos identificaron correctamente cada uno de los casos de ausencia de información generados aleatoriamente. Tanto el algoritmo inspirado en la técnica metaheurística de Recocido Simulado, como el basado en programación dinámica son eficaces en el descubrimiento de la situación de secuencia oculta ya que no se estimaron tareas incorrectas en los procesos analizados.

Al realizar el experimento se pudo observar, que el algoritmo inspirado en la técnica metaheurística utiliza más tiempo de ejecución que el algoritmo basado en la técnica de programación dinámica, debido a que su complejidad algorítmica depende del número de iteraciones que realice en la búsqueda. Sin embargo ambos son aplicables al descubrimiento de la ausencia de información en las trazas utilizadas en la

minería de proceso para el caso de secuencia oculta. La figura 16 muestra gráficamente el comportamiento para cada uno de los algoritmos, aplicados a los procesos antes generados.

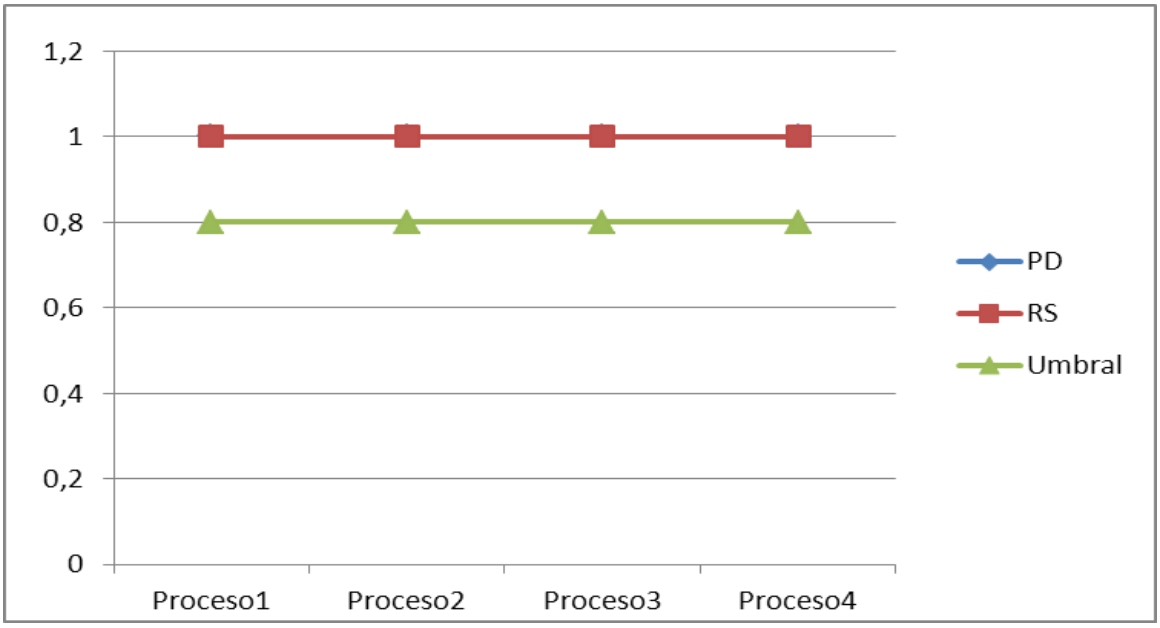


Figura 16 Resultados del experimento

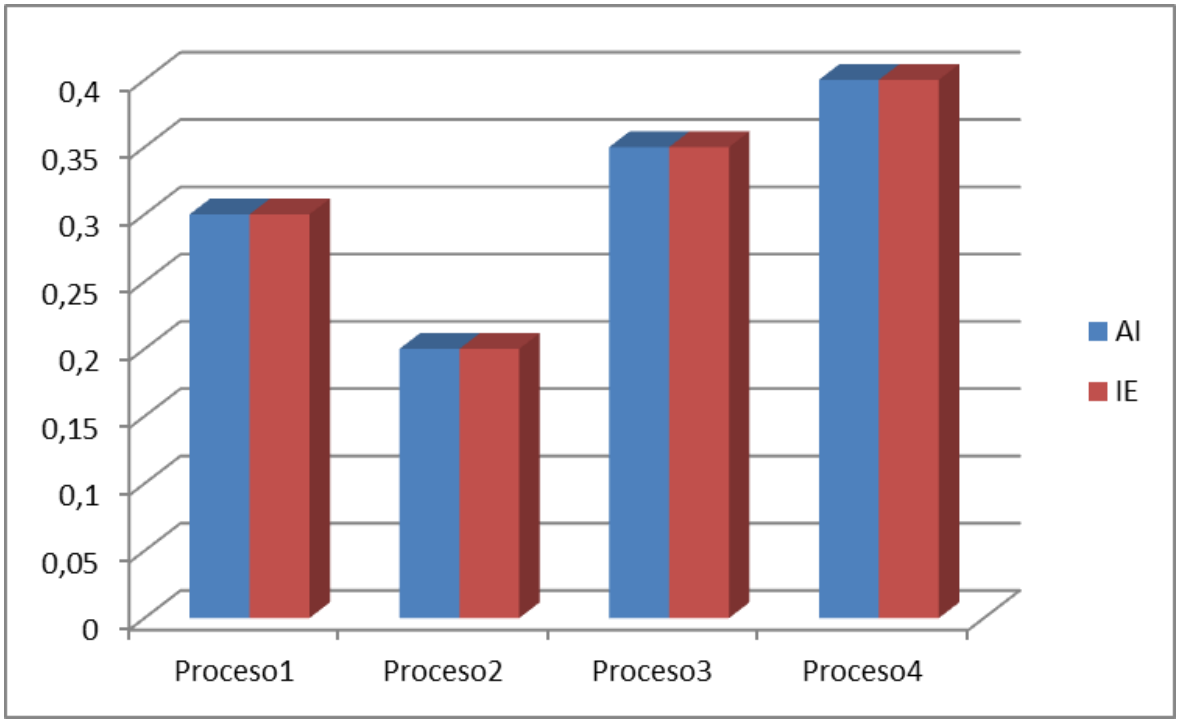


Figura 17 Relación entre ausencia de información (AI) e información estimada (IE)

La figura 17 representa el porciento de ausencia de información (AI) y de información estimada (IE) para cada proceso. En el proceso 1 la ausencia de información (AI) representa un 30%, en el proceso 2 la ausencia de información representa el 20%, la ausencia de información generada en el proceso 3 representa el 35% y por último en el proceso 4 se generó un 40% de ausencia de información. Luego de aplicar los algoritmos propuestos, tanto el basado en programación dinámica como el inspirado en la técnica metaheurística Recocido Simulado, se logró estimar el total de la ausencia de información generada aleatoriamente para cada proceso.

3.3 Aplicación de la propuesta en un entorno real

Para la aplicación de la propuesta en un entorno real se examinó un registro de eventos almacenado por el Sistema Único de Identificación Nacional (SUIN), específicamente del módulo Gestionar Recursos. El

SUIN fue desarrollado por el Ministerio del Interior de Cuba en conjunto con la Universidad de las Ciencias Informáticas.

El proceso Gestionar Recursos a partir del cual se generó el registro de eventos, tiene como objetivo gestionar las operaciones que se realizan sobre cualquier recurso del entorno informatizado. Un recurso puede ser desde un dispositivo de hardware hasta una interface del SUIN.

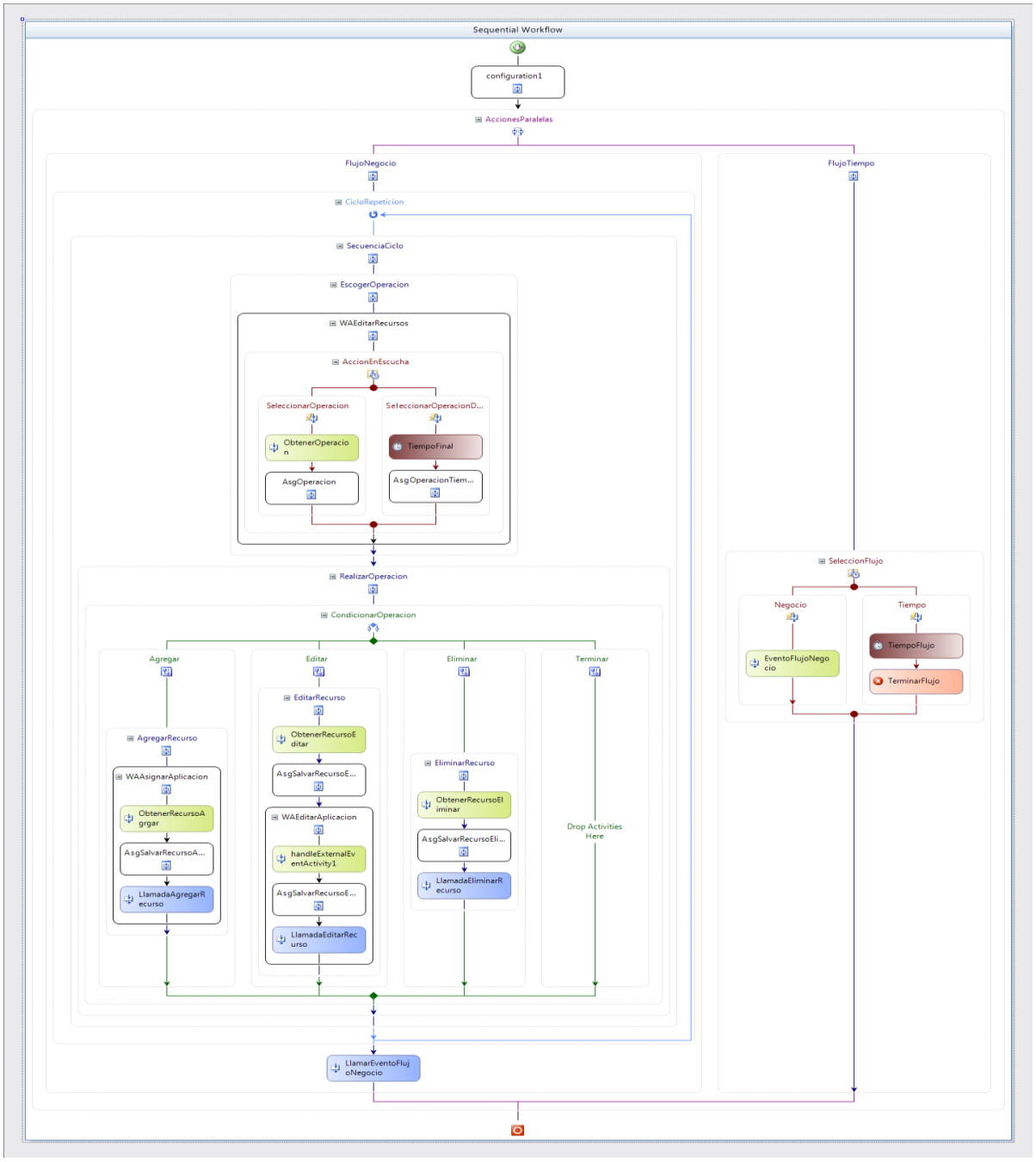


Figura 18 Modelo representativo del proceso Gestionar Recursos

El SUIN fue desarrollado utilizando la tecnología .Net, específicamente Windows Workflow Foundation. El registro de eventos utilizado presenta 51 casos, 11011 eventos, 90 clases de eventos, 3 tipos de eventos (Execute, Closed y Faulting) y 47 participantes.

Primeramente haciendo uso de la herramienta informática desarrollada se alinean las trazas en un solo grupo. Luego se realiza un análisis de las trazas alineadas y se comprueba que solo siete casos son completos. A pesar de esto, se decide trabajar con los 51 casos ya que de extraer los casos incompletos el registro de evento representaría solo una parte del comportamiento registrado del proceso. Si se dejan solo los siete casos completos no se aprecian eventos relacionados con el subproceso Agregar recurso. Después de este análisis se descompone el proceso analizado.

A partir del árbol de bloques de construcción generado se realiza la estimación de información ausente y se obtienen ocho actividades invisibles.

La Figura 19 muestra en el panel izquierdo, los operadores seleccionados para la estimación (Operador de salto, Operador de lazo, Operador de división/unión y Operador de secuencia oculta). En el panel derecho, en la sección donde se describen las actividades invisibles, se muestran las actividades estimadas λ_1, λ_{11}, λ_{16}, λ_{17}, λ_{18}, λ_{19}, λ_{20} y λ_{21}. Siete de las ocho actividades estimadas se originaron por el Operador de salto, mientras que la restante se creó por el Operador de secuencia oculta.

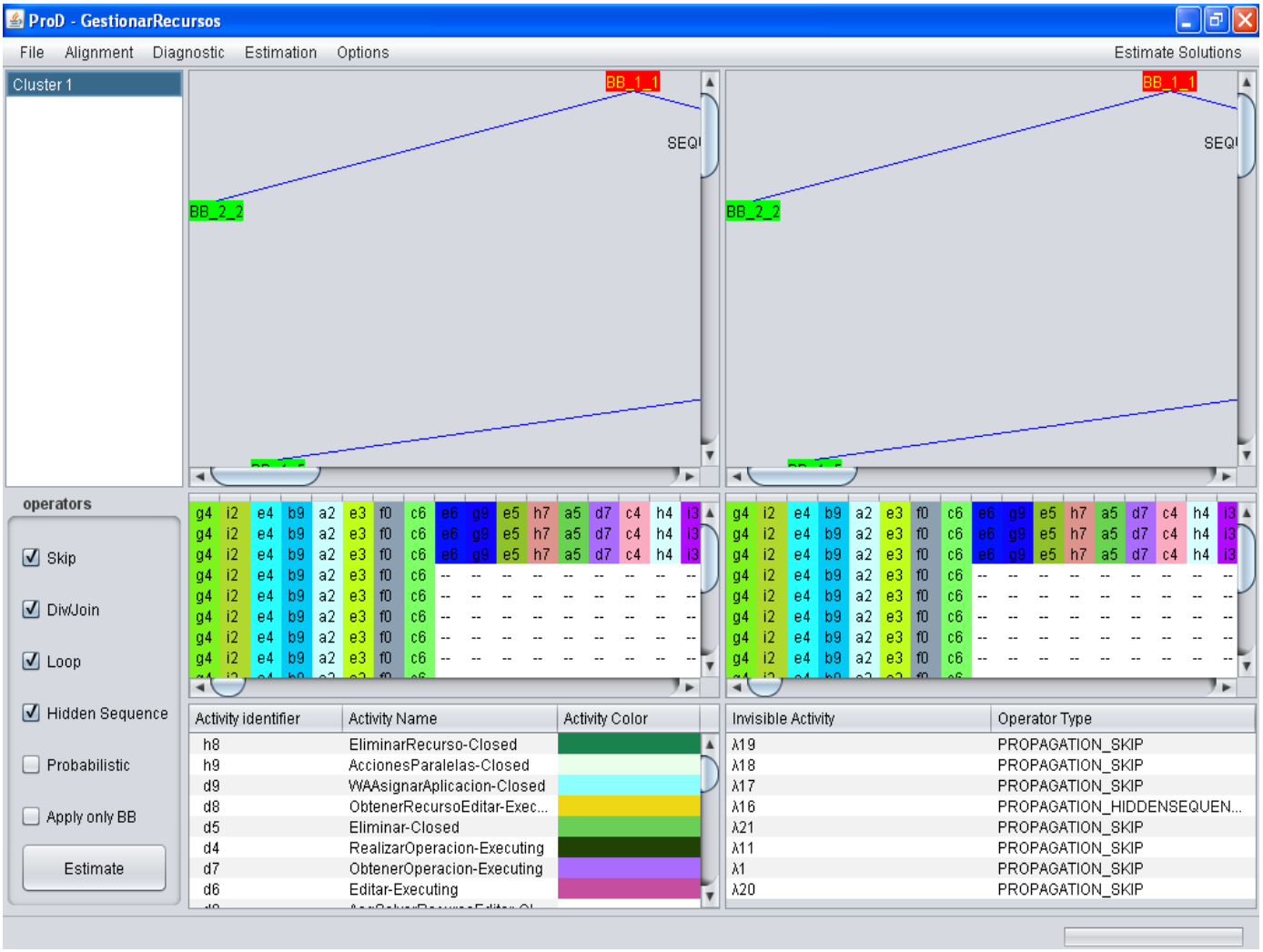


Figura 19 Estimación de información ausente para el proceso Gestionar Recursos

Las actividades invisibles no aparecen numeradas de manera consecutiva porque en el proceso de estimación se originan 22 actividades invisibles, pero se eliminan del resultado final las actividades invisibles que solo tienen sentido localmente.

A partir de la estimación realizada se hace un análisis en conjunto con un especialista del negocio para garantizar que las actividades estimadas se manifiestan en el proceso ejecutado. De este análisis se pudo concluir lo siguiente:

- Las actividades λ_{11} y λ_{21} indican casos incompletos (ausencia de actividades finales del proceso). Es decir, durante el pre-procesamiento de la alineación no se eliminaron del registro de eventos los posibles casos incompletos, para así evaluar la respuesta del sistema desarrollado en estos casos. Cada una de las actividades estimadas se corresponde con la ausencia de información en casos diferentes.
- Las actividades λ_{1}, λ_{17}, λ_{18}, λ_{19}, y λ_{20} indican el éxito en los cinco subprocesos Agregar, Editar, Eliminar recursos, Condicionar Operación y Flujo de Negocio. Las actividades estimadas permitieron identificar fácilmente los casos en los que el proceso funcionó correctamente.
- La actividad λ_{16} permite delimitar el subproceso Condicionar Operación. Después del evento Condicionar Operación en uno de los casos no se registró ningún otro evento (caso incompleto), por lo cual, no es posible en el registro de eventos analizado identificar adecuadamente el subproceso Condicionar Operación. El subproceso Condicionar Operación es el que incluye los subprocesos Agregar, Editar y Eliminar recursos. La actividad invisible en este caso permitió delimitar acertadamente el subproceso Condicionar Operación lo cual facilita el análisis de la información contenida en el registro de eventos.

Cada una las actividades estimadas se corresponden con actividades existentes en el proceso de negocio ejecutado, aun cuando existen casos incompletos que pudiesen dificultar el análisis.

3.4 Conclusiones del capítulo

En este capítulo se describió la manera por la que fue comprobada la propuesta. Para ello se generaron con la aplicación informática Process Log Generator un total de cuatro procesos. Con los cuales se logró obtener los resultados esperados en la validación. El experimento demostró la variación de la información

en los diferentes momentos antes descritos, donde los procesos con ausencia de información pertenecientes el segundo momento, recuperan la misma con la estimación de las tareas invisibles en el tercer momento.

Es importante destacar que el experimento realizado logró demostrar también que al ocurrir la estimación de la información ausente, posibilita una mejor comprensión del modelo descubierto, así como la correcta relación entre las actividades del registro de eventos.

Además se aplicó la propuesta en un entorno real, obteniendo resultados satisfactorios para la misma. El entorno empleado fue el Sistema Único de Identificación Nacional.

CONCLUSIONES

Durante la investigación se fueron cumpliendo los objetivos propuestos en la medida del avance de la misma. Esto conllevó a arribar a las siguientes conclusiones.

En el área de la minería de proceso la ausencia de información, afecta directamente al modelo descubierto y dificulta la comprensión del mismo. Ninguna de las técnicas analizadas en el presente trabajo, permite el tratamiento de todas las situaciones asociadas a la ausencia de información que fueron mencionadas.

Para la estimación de la información ausente en las trazas utilizadas en la minería de proceso, específicamente para el caso de la secuencia oculta, se desarrollaron dos algoritmos que responden a diferentes técnicas en su implementación. Ambos algoritmos demuestran tener una complejidad aceptable.

Las pruebas realizadas haciendo uso del sistema propuesto y con la utilización de las herramientas antes mencionadas, permitieron establecer una valoración de los algoritmos desarrollados, así como la certeza de cada uno por separado. Los resultados obtenidos demuestran que ambos algoritmos representan dos aristas fundamentales para la estimación de la información ausente para el caso de secuencia oculta. Debido a esto el sistema propuesto logra una disminución de las afectaciones provocadas por la ausencia de información sobre la comprensión de los modelos descubiertos en el área de la minería de proceso.

RECOMENDACIONES

- Implementar un algoritmo para la estimación de la información ausente, utilizando otras técnicas de metaheurística.
- Validar los resultados aplicando otros algoritmos utilizados en la minería de proceso.
- Perfeccionar la herramienta.

REFERENCIAS BIBLIOGRÁFICAS

1. Agrawal, 1998.
2. Aalst, W.M.P.v.d., *Process Mining. Discovery, Conformance and Enhancement of Business Processes*. 2011: Springer Heidelberg Dordrecht London New York.
3. Aalst, W.M.P.v.d. and A.J.M.M. Weijters, *Process Mining: A Research Agenda.* Special Issue of Computers in Industry, Elsevier Science Publishers, Amsterdam, 2004. **53**(3).
4. Gunter, A.a., 2007.
5. Aalst, 2011.
6. Medeiros, 2006.
7. Bose, R.P.J.C. and W.M.P.v.d. Aalst. *Context Aware Trace Clustering: Towards Improving Process Mining Results*. in *International Conference on Data Mining*. 2009: is.ieis.tue.nl.
8. Song, M., C.W. Günther, and W.M.P. Aalst. *Trace Clustering in Process Mining*. in *Business Process Management Workshops (2009)*. 2009. Milano, Italy Lecture Notes.
9. Bose, R.P.J.C. and W.M.P.v.d. Aalst. *Trace Alignment in Process Mining: Opportunities for Process Diagnostics*. in *International Conference on Business Process Management (BPM'2010)*. 2010: Springer-Verlag Berlin, Heidelberg.
10. Agrawal, R., D. Gunopulos, and F. Leymann. *Mining Process Models from Workflow Logs*. in *EDBT '98 Proceedings of the 6th International Conference on Extending Database Technology: Advances in Database Technology*. 1998: Springer-Verlag London, UK.
11. Medeiros, A.K.A.d., *Genetic Process Mining*. 2006, Technische Universiteit Eindhoven.
12. Schimm, G. *Mining Most Specic Workflow Models from Event-Based Data.* in *International Conference on Business Process Management (BPM 2003)*. 2003: Lecture Notes in Computer Science.
13. Schimm, G., *Mining Exact Models of Concurrent Workflows.* Computers in Industry, 2004. **53**(3): p. 265-281.
14. Bergenthum, R., et al., *Process Mining Based on Regions of Languages.* Lecture Notes in Computer Science 2007. **4714**: p. 375-383.

15. Aalst, W.M.P.v.d. and C.W. Günther. *Finding Structure in Unstructured Processes: The Case for Process Mining*. in *ACSD '07 Proceedings of the Seventh International Conference on Application of Concurrency to System Design*. 2007: IEEE Computer Society.
16. Springer Heidelberg Dordrecht London New York, *W. M. P. v. d. Aalst, Process Mining. Discovery, Conformance and Enhancement of Business Processes:.* 2011.
17. AALST, W.M.P.V.D., 2011.
18. al., A.e., 2011.
19. AALST, R.a., 2008.
20. al, A.e., 2011.
21. Goomas, D.T., S.M. Smith, and T.D. Ludwig, *Business Activity Monitoring: Real-Time Group Goals and Feedback Using an Overhead Scoreboard in a Distribution Center.* Journal of Organizational Behavior Management 2011. **31**(3): p. 196-209.
22. Wang, D., et al. *Active Complex Event Processing infrastructure: Monitoring and reacting to event streams* in *Data Engineering Workshops (ICDEW), 2011 IEEE 27th International Conference*. 2011. Hannover, USA.
23. Roth, M. and S. Donath. *Applying Complex Event Processing towards Monitoring of Multi-party Contracts and Services for Logistics – A Discussion*. in *Business Process Management Workshops*. 2012: Lecture Notes in Business Information Processing.
24. Sheer, A.-W., et al., *Corporate performance management: ARIS in practice*. 2006: Springer.
25. Paladino, B., *Innovative corporate performance management: Five key principles to accelerate results*. 2010: John Wiley & Sons.
26. Wang, L., D.B.G. Herve, and Y. Shen, *Continuous Process Improvement in Banking Sector and a Model Design for Performance Enhancement.* International Journal of Business and Management, 2012. **7**(2).
27. Chen, L., T. Xue, and A. Yang. *Business Process Continuous Improvement System Based on Workflow Mining Technology*. in *Computer Science and Information Engineering, 2009 WRI World Congress*. 2009. Los Angeles, CA.
28. Abou Moghdeb, F., M. Indulska, and P. Green. *Business process improvement and organizational theory: The missing link*. in *Information Resources Management Assocation (IRMA) International Conference*. 2007. Vancouver, British Columbia IGI Publishing

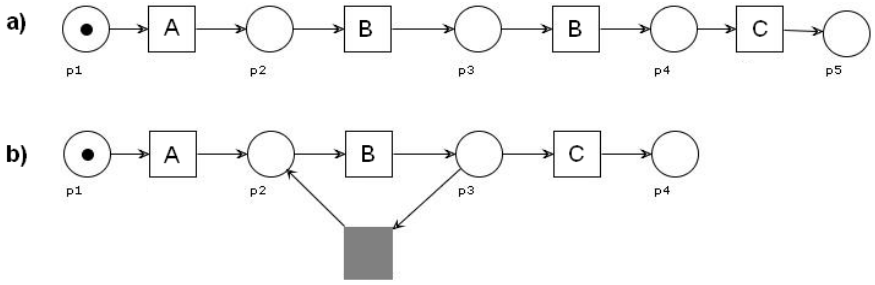
29. Zellner, G., *A structured evaluation of business process improvement approaches.* Business Process Management Journal, 2011. **17**(2): p. 203 - 237.
30. Soltani, E., et al., *A review of the theory and practice of managing TQM: An integrative framework.* Total Quality Management & Business Excellence 2008. **19**(5): p. 461-479.
31. Gorji, A.M.H. and J.A. Farooquie, *A comparative study of total quality management of health care system in India and Iran.* BMC Research Notes, 2011. **4**(566).
32. Schroeder, R.G., et al., *Six Sigma: Definition and underlying theory.* Journal of Operations Management, 2008. **26**(4): p. 536-554.
33. Aalst, W.M.P.v.d., et al., *Business process mining: An industrial application.* Information Systems, 2007. **32**(5).
34. Grigori, D., et al., *Business Process Intelligence.* Computers in Industry, 2004. **53**(3): p. 321-343.
35. Aalst, W.M.P.V.d., et al., *Process Mining Manifesto.* Business Process Management Workshops 2011, Lecture Notes in Business Information Processing.Springer-Verlag, 2011. **99**.
36. PYMNTS. (2011). OpenConnect Comprehend Business Process Intelligence and Analytics Software Increases Productivity, P.J.C.A., 2011: p. http://pymnts.com/news/businesswire-feed/2011/march/21/openconnect-comprehend-business-process-intelligence-and-analytics-software-increases-productivity-promotes-job-creation-20110321005124.
37. Stereologic-Ltd. (2012). Discover, A.h.w.s.c., *Visualize and Analyze Business Processes in Real Time.* 2012.
38. http://www.softwareag.com/corporate/products/aris_platform/aris_controlling/aris_process_performance/overview/default.asp, S.-A.A., *ARIS Process Performance Manager.* 2011.
39. Lontas., *Process Discovery Focus. Available:.* 2012: p. http://www.iontas.com/pages/products/pdf.php.
40. Aalst, W.M.P.v.d., A.J.M.M. Weijters, and L. Maruster, *Workflow Mining: Discovering process models from event logs.* IEEE Transactions on Knowledge and Data Engineering, 2004. **16**(9): p. 1128-1142.
41. Medeiros, A.K.A.d., W.M.P.v.d. Aalst, and A.J.M.M. Weijters. *Workflow Mining: Current Status and Future Directions*. in *The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE*. 2003: Lecture Notes in Computer Science. Springer, Berlin.

42. Rozinat, A., et al. *The Need for a Process Mining Evaluation Framework in Research and Practice. .* in *BPM 2007 International Workshops (BPI, BPD, CBP, ProHealth, RefMod, Semantics4ws)*. 2008: Lecture Notes in Computer Science. Springer, Berlin.
43. Weijters, A.J.M.M. and J.T.S. Ribeiro, *Flexible Heuristics Miner (FHM). .* BETA Working Paper Series, WP 334, Eindhoven University of Technology, Eindhoven., 2010.
44. Weijters, A.J.M.M. and W.M.P.v.d. Aalst, *Rediscovering Workflow Models from Event-Based Data using Little Thumb.* Integrated Computer-Aided Engineering, 2003: p. 151-162.
45. Aalst, W.M.P.v.d., et al., *ProcessMining: A Two-Step Approach to Balance Between Underfitting and Overfitting.* Software and Systems Modeling, 2009. **9**(1): p. 87-111.
46. Günther, C.W. and W.M.P.v.d. Aalst. *Fuzzy Mining: Adaptive Process Simplification Based on Multi-Perspective Metrics*. in *International Conference on Business Process Management (BPM 2007)*. 2007: Lecture Notes in Computer Science. Springer, Berlin.
47. Aalst, W.M.P.v.d. *Business Process Simulation Revisited*. in *Enterprise and Organizational Modeling and Simulation*. 2010: Lecture Notes in Business Information Processing. Springer, Berlin.
48. Aalst, W.M.P.V.d. *Three Good Reasons for Using a Petri-net-based Workflow Management System*. in *The International Working Conference on Information and Process Integration in Enterprises (IPIC'96)*. 1996.
49. Aalst, W.M.P.v.d., *Structural Characterizations of Sound Workflow Nets.* Computing Science Reports 96/23, 1996.
50. Aalst, W.M.P.v.d., et al. *ProM: The Process Mining Toolkit*. in *Proceedings of BPM (Demos)'2009*. 2009. Ulm, Germany.
51. Cook, J.E., *Process Discovery and Validation Through Event-Data Analysis*. 1996.
52. Cook, J.E. and A.L. Wolf. *Automating Process Discovery Through Event-Data Analysis*. in *ICSE '95: Proceedings of the 17th international conference on Software engineering*. 1995. New York, USA: ACM Press.
53. Cook, J.E., et al., *Discovering Models of Behavior for Concurrent Workflows.* Computers in Industry, 2004: p. 297-319.

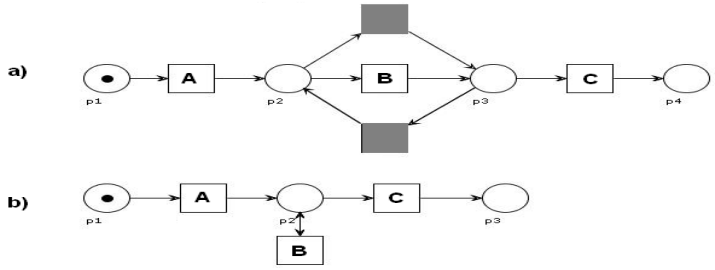
54. Cook, J.E. and A.L. Wolf. *Event-Based Detection of Concurrency*. in *Proceedings of the Sixth International Symposium on the Foundations of Software Engineering (FSE-6)*. 1998. New York, NY, USA.
55. Herbst, J. *A Machine Learning Approach to Workflow Management*. in *11th European Conference on Machine Learning*.
56. Herbst, J., *Ein induktiver Ansatz zur Akquisition und Adaption von Workflow-Modellen*. 2001, University Ulm.
57. Herbst, J. and D. Karagiannis, *Integrating Machine Learning and Work-flow Management to Support Acquisition and Adaptation of Workflow Models.* International Journal of Intelligent Systems in Accounting, Finance and Management, 2000: p. 67-92.
58. Herbst, J. and D. Karagiannis, *Workflow Mining with InWoLvE.* Computers in Industry, 2004. **53**(3): p. 245-264.
59. Aalst, W.M.P.v.d. and K.M.v. Hee, *Workflow Management: Models, Methods, and Systems*. 2004, USA.
60. Aalst, W.M.P.v.d., A.H.M.t. Hofstede, and M. Weske, *Business Process Management: A Survey* Lecture Notes in Computer Science, 2003. **2678**.
61. Rozinat, A. and W.M.P.v.d. Aalst. *Conformance Testing: Measuring the Fit and Appropriateness of Event Logs and Process Models*. in *Third International Conference on Business Process Management(BPM 2005)*. 2005. France.
62. Aalst, W.M.P.v.d. and A.J.M.M. Weijters, *Process Mining.* Special Issue of Computers in Industry. Elsevier Science Publishers, Amsterdam, 2004. **53**.
63. Wen, L., et al., *A Novel Approach for Process Mining Based on Event Types.* BETA Working Paper Series, WP 118, Eindhoven University of Technology, Eindhoven, 2004.
64. Wen, L., J. Wang, and J. Sun, *Detecting Implicit Dependencies Between Tasks from Event Logs.* APWeb, Lecture Notes in Computer Science. Springer, 2006. **3841**: p. 591-603.
65. Rubin, V., *A Workflow Mining Approach for Deriving Software Process Models*. 2007, University of Paderborn.
66. *TÉCNICAS METAHEURÍSTICAS. ALGORITMOS BASADOS EN NUBES DE PARTÍCULAS.*
67. Kathyn A. Dowsland, B.A.D., *Diseño de Heuristicas y Fundamentos de Recocido Simulado.* 2003.
68. Goic, M., *Programación Dinámica.*

69. Adriansyah, A., B.F.v. Dongen, and W.M.P.v.d. Aalst. *Towards Robust Conformance Checking*. in *BPM 2010 Workshops, Proceedings of the 6th Workshop on Business Process Intelligence (BPI2010)*. 2011. Lecture Notes in Business Information Processing. Springer, Berlin.
70. Rozinat, A. and W.M.P.v.d. Aalst, *Conformance checking of processes based on monitoring real behavior.* Inf. Syst., 2008. **33**(1): p. 64-95.
71. Arendonk, R.P.J.M.v., *A Benchmark Set for Process Discovery Algorithms*, in *Mathematics & Computer Science*. 2011, Eindhoven University of Technology: Eindhoven.
72. Rozinat, A., et al., *Towards an Evaluation Framework for Process Mining Algorithms.* BPM Center Report, 2007.
73. Weerdt, J.D., et al. *A critical evaluation study of model-log metrics in process discovery*. in *6th International Workshop on Business Process Intelligence*. 2010: Springer.
74. Muñoz-Gama, J. and J. Carmona. *A fresh look at precision in process conformance*. in *8th international conference on Business process management*. 2010. Hoboken, NJ, USA: Springer-Verlag.
75. Process-Mining-group, *Process Log Generator*, in *Process Mining group*. 2011: Padua, Italy.
76. Werf, J.M., et al., *Process Discovery Using Integer Linear Programming*, in *Proceedings of the 29th international conference on Applications and Theory of Petri Nets*. 2008, Springer-Verlag: Xi'an, China. p. 368-387.
77. Wiel, T.v.d., *Process Mining using Integer Linear Programming*, in *Department of Mathematics and Computer Science*. 2010, Eindhoven University of Technology: Eindhoven.

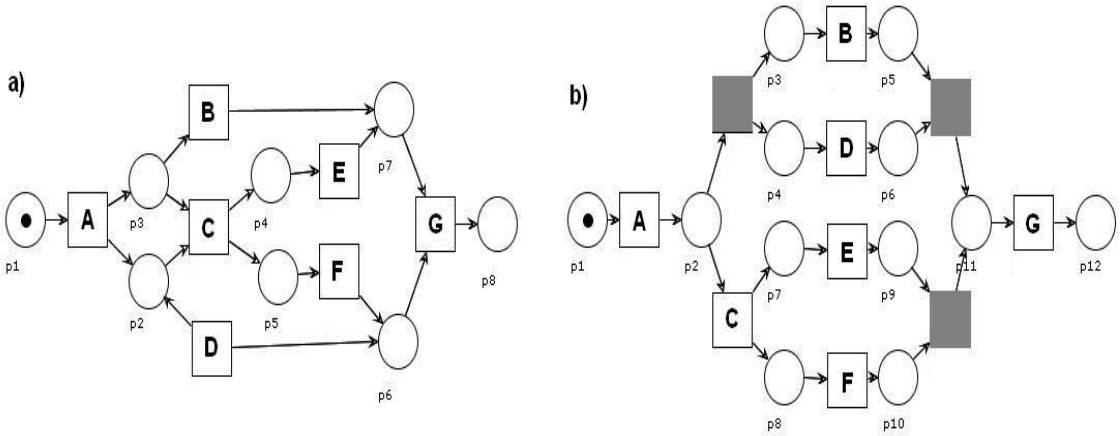
ANEXOS



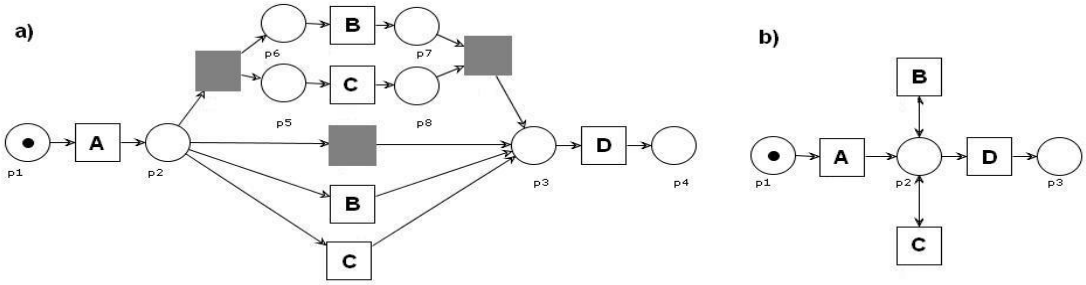
1. Esta imagen representa una Red de flujo de trabajo, muestra en a) el uso del constructor de actividades duplicadas y en b) el constructor de actividades invisibles.



2. Red de flujo de trabajo que, muestra en a) el uso del constructor de actividades invisibles y en b) el constructor de lazos.



3. Red de flujo de trabajo, muestra en a) el uso del constructor de XOR-split/join y en b) el constructor de actividades invisibles



4. Red de flujo de trabajo, muestra en a) el uso del constructor de actividades invisibles unido al de actividades duplicadas y en b) el constructor de lazos

Simulación Termodinámica	Optimización Combinatoria
Estado del sistema	Soluciones factibles
Energía	Coste
Cambio de estado	Solución en el entorno
Temperatura	Parámetro de control
Estado congelado	Solución heurística

5. Relación establecida entre los elementos de Simulación termodinámica y Optimización Combinatoria.