Universidad Antonio Nariño
Doctorado en Ciencia Aplicada

Mario Augusto Pinto Serrano

Método y sistema para reducir tiempos de procesamiento de solicitudes en líneas de espera.

Método y sistema para reducir tiempos de procesamiento de solicitudes en líneas de espera.

Mario Augusto Pinto Serrano

Documento presentado como requisito para optar por el título de Doctor en Ciencia Aplicada

Directores: Diego Ferney Gómez Cajas, Rafael María Gutiérrez Salamanca
Codirector Metodológico: Andrés Ignacio Hernández Duarte.

Doctorado en Ciencia Aplicada
Universidad Antonio Nariño
Bogotá D.C.
2021

Prefacio

Las colas están presentes en múltiples servicios, donde los recursos de atención al cliente tienen una capacidad menor que la demanda de dichos servicios. Con el fin de mejorar la calidad de servicio prestado en presencia de colas, se propone un sistema de gestión de colas, buscando principalmente reducir el tiempo promedio de espera. De esta forma se parte de conceptos de sistemas de telecomunicaciones generalizándolos tanto a paquetes de datos como a personas. La producción científica, en forma de dos artículos, uno en conferencia internacional y el otro en revista indexada, fue realizada dentro del marco de cooperación internacional entre la Universidad Antonio Nariño en Colombia y la Universidad de Ghent en Bélgica.

Resumen

Dado que generalmente los recursos de atención al cliente no pueden atender a toda la demanda de servicios, todo el tiempo, se generan las colas, conocidas como líneas de espera. Estas colas están presentes en múltiples aspectos de la vida cotidiana. A partir de un fenómeno presente en los sistemas de comunicaciones ópticos conocido como contención (cuando dos paquetes o más se encuentran simultáneamente en un mismo punto), se han planteado algoritmos para manejar la contención. Estos algoritmos se clasifican en dos categorías: pre-reservación y post-reservación. En los algoritmos de pre-reservación, el canal de salida se reserva antes de que el paquete ingrese, en contraste con los de post-reservación, donde la reservación se pospone, hasta después de que el paquete ingrese al sistema. Con base en los fundamentos teóricos de los algoritmos de post-reservación, y haciendo abstracciones de los conceptos manejados, se propone un sistema de gestión de colas, enfocado a servicios de alta complejidad operativa, con altas demandas e interacciones entre los componentes. Para medir el desempeño en el sistema de gestión de colas se consideró la métrica de tiempo promedio de espera en cola, usada también cuando los recursos en los sistemas de comunicaciones, son ilimitados. En caso de restricciones en el número de bucles y/o recirculaciones, en sistemas ópticos de comunicaciones, se utilizan otras métricas como la probabilidad de pérdida (proporción de paquetes perdidos en comparación con el total de paquetes) y el tamaño de pérdida (LPSize), el cual compara el tamaño acumulado de los paquetes perdidos, en relación con el tamaño acumulado de todos los paquetes). Como aporte al conocimiento se propusieron los algoritmos WAS, XAS, T-WAS y T-XAS, donde el algoritmo WAS mostro mejor desempeño que el de referencia (VAS) en términos de la probabilidad de perdida y el XAS se desempeño mejor que el VAS, considerando el LPSize. Aunque en múltiples instancias las variantes con umbral (T-WAS, T-XAS) mostraron mejoras en términos de la probabilidad de perdida y LPSize, la mejoría no fue tan notable como con WAS y XAS. En ambientes sin restricciones los nuevos algoritmos, mostraron mejor desempeño en retardo de paquete.

Abstract

Given that generally the customer service's resources cannot give service to all service demands, all the time, the queues are generated, also known as waiting lines. These queues are present in several aspects of the daily life. From a phenomenon present in optical communication systems, known as contention (When two or more packets are simultaneously at the same point), are proposed algorithms in order to deal with the contention. These algorithms are classified in two categories: pre-reservation and post-reservation. In the pre-reservation algorithms, the output channel is reserved before the packet ingress, in contrast with the pos-reservation algorithms, where the reservation is postponed, until after the ingress of the packet to the system. Based on the theoretical foundations of the post-reservation algorithms, and making abstractions of the used concepts, we propose a queue management system, focused on high complexity operative services, with high demands and interactions between the components. In order to measure the performance of the queue management system, we consider the metric average waiting time, in queue, used also with unlimited resources in communication systems. In case of restrictions either in the loops number and/or recirculations in optical communication systems, are used other metrics as the loss probability (ratio of lost packets in comparison to the total number of packets) and LPSize (which compares the cumulative size of the lost packets, compared with the cumulative size of all packets). As the contribution to knowledge, are proposed the algorithms WAS, XAS, T-WAS and T-XAS, where the WAS algorithm gives a better performance than the reference (VAS) in terms of the Loss Probability and the XAS produces a better performance than VAS, in terms of LPSize. Even though in several instances the threshold variants (T-WAS, T-XAS) show improvements in terms of the Loss Probability and LPSize, the improvemens were not as remarkable as with WAS and XAS. In environments without constraints, the new algorithms gave better performance in the packet delay.

Agradecimientos

El autor agradece a los Doctores Rafael Gutiérrez, Diego Gómez, y Andrés Hernández por su asesoría y colaboración en la metodología para generar productos de innovación, así como en la preparación de documentos y presentaciones, durante todo el proceso del doctorado. Por otra parte, el autor agradece a los Doctores Wouter Rogiest y Herwig Bruneel, por la supervisión en la producción científica, y en la formación en los conceptos fundamentales y la metodología de investigación en el área de teoría de colas. Igualmente, el autor agradece a Kurt Van Hautegem por la excelente colaboración y formación en la producción científica, en particular en los artículos para el manejo de contención en buffers ópticos.

Finalmente, el autor manifiesta su gratitud a su madre y hermanas, quienes han sido su apoyo y fortaleza en el desarrollo del proyecto.

Lista de Abreviaturas y acrónimos

AMC: Adaptive Modulation and Coding

API: Application Program Interface

CAQT: Continuous Accusatory Queuing Theory

CDS: Channel and Delay Selection

CQ: Custom Queueing

D: Granularity

D2D: Device-to-Device

DG: Delay-Gap

FCFS: First-Come First Served

FDL: Fiber Delay Line

FPGA: Field-Programmable Gate Array

GD: Gap-Delay

IEEE: Institute of Electrical and Electronics Engineers

IP: Internet Protocol

LP: Loss Probability

LPSize: Size of Loss Probability

MANET: Mobile Ad-Hoc Network

MPTCP: Multi-Path TCP

OPNET: Optimized Network Engineering Tool (software)

PQ: Priority Queuing

QoS: Quality of Service

RFID: Radio Frequency Identification

SACK: Selective Acknowledgment

SLO: Service Level Object

SMACS: Stochastic Modelling and Analysis of Communication systems

TCP: Transmission Control Protocol

TELIN: Department of Telecommunication and Information Processing

T-WAS: Threshold WAS

T-XAS: Threshold XAS

VAS: Void Avoiding Schedule

VC: Videoconference

VF: Void Filling

VOIP: Voice over Internet Protocol

WAS: Minimizing the Wait for the Outgoing line to turn available after the departure of a couple of packets (Algorithm).

WC: Wavelength Converter

WFQ: Weighted Fair Queueing

WiFi: Name of a popular wireless networking technology that uses radio waves to provide wireless high-speed Internet and network connections.

WiMAX: Worldwide Interoperability for Microwave Access

WT: Waiting Time

XAS: eXtending the period during which the outgoing line is effectively used by the packets pair (Algorithm).

Tabla de Contenido

Lista de tablas

Lista de figuras

Capítulo 1. Introducción

Contenido

En este documento el primer capítulo contiene la introducción, explicando la importancia del trabajo realizado, los antecedentes, los objetivos, el alcance, la metodología usada y la aplicación en el área de conocimiento correspondiente. En el segundo capítulo se incluye información científica y tecnológica, con el fin de contextualizar la investigación presentada. El tercer capítulo contiene la metodología, mostrando el marco conceptual y las estrategias utilizados en el desarrollo del proyecto. En el cuarto capítulo se presentan los principales resultados obtenidos siguiendo la metodología y el análisis correspondiente de estos. productos de propiedad intelectual generados, con los detalles relevantes de cada uno de ellos. En el quinto capitulo se describe la producción tanto científica como tecnológica, generada a lo largo del proceso del doctorado. Finalmente, en el capítulo sexto se presentan las conclusiones obtenidas a partir del trabajo realizado. En los anexos se incluyen los productos generados.

Importancia del trabajo

En el caso en el que la demanda de servicios es superior a la capacidad de los entes prestadores de servicios, se generan colas, también conocidas como líneas de espera. Un factor crítico, que también funciona como métrica del desempeño, es el tiempo promedio de espera en cola. Asimismo, existen otras métricas como la proporción de clientes perdidos en comparación con el total de clientes que entran al sistema (conocido como Probabilidad de pérdida), y el tamaño acumulado de los paquetes perdidos con respecto al tamaño total de todos los paquetes (LPSize). Es de destacar que se pretendió reducir al mínimo los valores de las métricas mencionadas, dentro de las posibilidades y recursos disponibles.

Antecedentes

Dado que el trabajo de investigación se centró en algoritmos de contención en buffers ópticos, a continuación, se presentan los trabajos y conceptos esenciales con respecto a dichos algoritmos. En la siguiente figura se presenta un diagrama para ilustrar el manejo de la contención en los buffers ópticos.

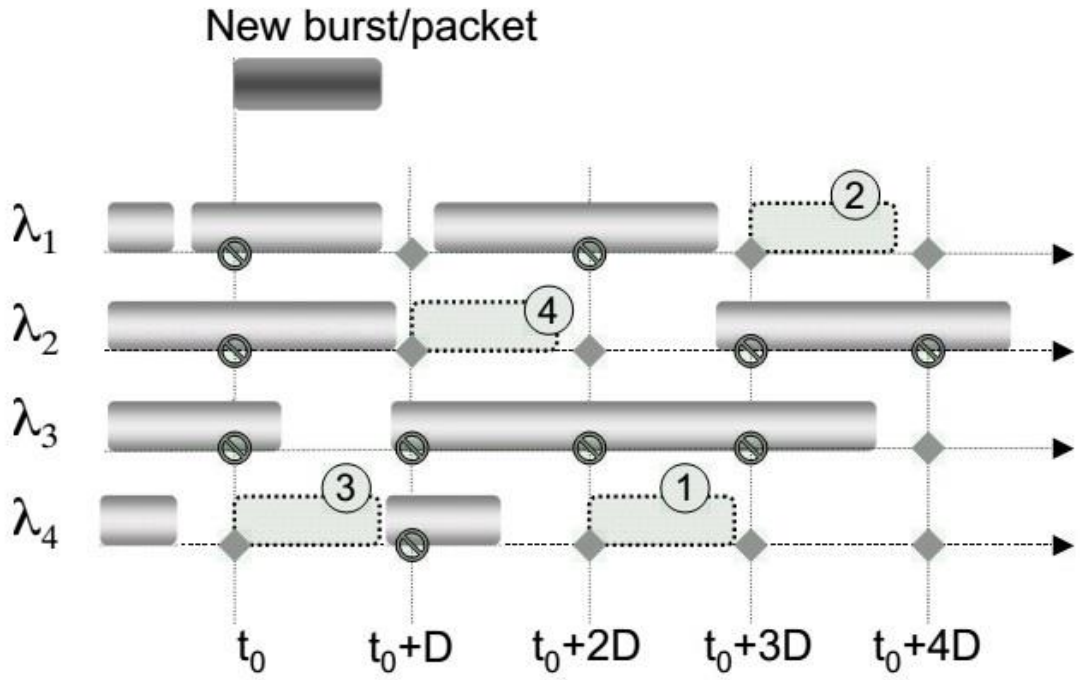

Figura 1: Diagrama de tiempo para diferentes algoritmos de programación de paquetes. Tomado de (F Callegati, Campi, & Cerroni, 2007)

En el eje vertical se tienen las longitudes de onda correspondientes, en el eje horizontal se encuentran los diferentes retardos tomando como punto de referencia un tiempo to. En este caso particular se trabajó con buffers degenerados en los cuales los retardos son múltiplos enteros de la granularidad, definida con D. Los rectángulos sólidos representan los paquetes que ya han sido programados. Los rectángulos punteados se corresponden con las posibles ubicaciones del nuevo paquete de acuerdo a ciertos algoritmos. Las posibles ubicaciones, representadas con los rombos, vienen dadas por las intersecciones de las longitudes de onda con los posibles retardos. Los círculos representan las ubicaciones que no son factibles por ya estar ocupadas con paquetes previamente programados. En general se consideran dos categorías, para este caso, con llenado de vacío (Void Filling) y sin llenado de vacío (No VF). Cuando no hay llenado de vacío, solo se tiene en cuenta el horizonte es decir el último paquete programado, que en el diagrama viene representado por el último paquete a la derecha. Teniendo en cuenta esto, hay dos opciones posibles: minimizar el delay o minimizar el gap. El delay es el mínimo retardo, cuando se trabaja sin Void Filling, la selección en particular es para el caso 1. Cuando se minimiza el gap, es decir la diferencia entre el extremo del paquete y el delay, la opción escogida viene dada por el caso 2. Para los casos en los que se consideran también los espacios entre paquetes (Voids), si se pretende minimizar el Delay se escogería la opción 3. Si el objetivo es minimizar el Gap (con llenado de vacío), la opción correspondiente seria la 4.

Para caracterizar el funcionamiento de un buffer óptico de bucles de fibra se deben tener en cuenta tanto el esquema de reservación como la disciplina de programación (Wouter Rogiest, Dorsman, & Fiems, 2014). Se tienen dos esquemas de reservación (Wouter Rogiest et al., 2014):

1. Pre-reservación: Se reserva el canal de salida en la llegada, previamente a la entrada al buffer.

2. Post-reservación: Se reserva el canal de salida al dejar el bucle de fibra. Se resuelve la contención en la salida, de forma que los paquetes pueden entrar libremente al buffer, dejando para un momento posterior la decisión del momento de salida del paquete.

Las disciplinas de programación usadas en los esquemas de pre-reservación, para un buffer de una sola longitud de onda, son (Wouter Rogiest et al., 2014):

1. Primero en llegar Primero en ser servido (FCFS, de First-Come First-Served): Los paquetes tienen un retardo que se corresponde con el múltiplo más pequeño de la duración de bucle, S, mayor que el tiempo requerido por todos los paquetes que han llegado anteriormente, término que se conoce como horizonte de programación (horizon scheduling)

2. Llenado de vacío (Void-Filling): Por defecto la programación es FCFS con la excepción de que exista un vacío (void) en el programa provisional, en el que se puedan programar paquetes entre otros, mejorando el desempeño con respecto al FCFS.

3. Creación de vacío (Void-Creating) Se mejora la programación de llenado de vacío al crear selectivamente vacíos mayores a lo estrictamente necesario, únicamente en los casos en los que exista una alta probabilidad de ser llenados más tarde.

Para el caso de post-reservación en (Wouter Rogiest et al., 2014) se propone una disciplina de programación conocida como VAS (Void-Avoiding Schedule), Programa de Evitado de Vacío, la cual busca minimizar los vacíos en el canal de salida. En esta disciplina los paquetes se empiezan a transmitir después de llegar cuando el canal está disponible, cuando no lo está, los paquetes entran a un bucle de fibra, del cual pueden salir en un Offset S. Si entonces el canal está disponible, el paquete sale de su bucle (de modo que se tiene una transmisión real). En el caso de no disponibilidad del canal, hay una recirculación del paquete en su bucle, de manera que se tiene de nuevo un retardo (delay) S (Wouter Rogiest et al., 2014).

Cuando dos o más paquetes tienen que acceder a un canal compartido, simultáneamente, es necesario un mecanismo para la resolución de la contención (Franco Callegati, Cerroni, & Pavani, 2007). La contención puede tratarse en los dominios del espacio, la longitud de onda y el tiempo, junto con una política óptima de programación (Franco Callegati et al., 2007). Para el dominio del espacio, es

necesario disponer de diferentes fibras, en el dominio de la longitud de onda, se requieren conversores para pasar de una longitud de onda a otra. En el dominio del tiempo, se usan retardos para retrasar la transmisión por cierto tiempo (Franco Callegati et al., 2007).

El problema de Selección de Canal y Retardo (CDS) requiere la consideración de varios parámetros, tales como el número de fibras y longitudes de onda, la cantidad disponible de retardos y la capacidad de conversión (Franco Callegati et al., 2007).

Alcance

Teniendo en cuenta que existen ciertos antecedentes teóricos en la gestión de colas, los cuales han evolucionado durante décadas, y han sido validados empíricamente, se propone que la investigación sea descriptiva, pero se pretende dar cierto nivel de explicación al manejo de variables, en particular el tiempo promedio de espera. Se busca de esta forma establecer las características del fenómeno de espera en cola, así como plantear estrategias para optimizar el desempeño de sistemas enfocados en el manejo de las líneas de espera.

Objetivos

Para reducir las métricas de desempeño, se han desarrollado algoritmos de post-reservación, en los que se analizan las combinaciones de pares de paquetes para optimizar la asignación de recursos a los paquetes de datos. Entre estos algoritmos se incluyen WAS, XAS, y las variantes respectivas con umbrales: T-WAS y T-XAS. Los algoritmos WAS y T-WAS se enfocaron en mejorar la probabilidad de pérdida, cuando hay restricciones en el número de bucles y/o recirculaciones, en contraste con los algoritmos XAS y T-XAS, donde el objetivo fue la optimización de la métrica LPSize. Cuando no hay límites en el número de bucles ni en el de recirculaciones, la métrica objetivo es el tiempo promedio de espera, puesto que no hay pérdidas, dados los recursos ilimitados.

Cabe destacar que la producción científica realizada estuvo dentro del marco de cooperación con la Universidad de Ghent en Bélgica, en particular con el grupo SMACS del departamento TELIN, de dicha universidad. En este marco se pretendió hacer investigación conjunta, generando como resultados artículos, los cuales se publicaron en Conferencias de renombre internacional y en revistas prestigiosas, aumentando la visibilidad de las instituciones involucradas. Uno de los artículos publicados en conferencia, fue el de la QTNA 2018, realizada en Japón en junio de 2018, considerando los algoritmos WAS y XAS, mencionados previamente. El otro artículo fue publicado en la revista Optical Switching and Networking (OSN), en el 2019, donde la revista OSN es Q2.

Aplicación

El trabajo de investigación es generado a partir el manejo de contención en buffers ópticos, y tiene aplicabilidad en la gestión de líneas de espera (o colas), considerando métricas de desempeño como el tiempo de espera o la probabilidad de pérdida. El trabajo maneja conceptos matemáticos (procesos estocásticos), de investigación de operaciones, de simulación, por lo que es interdisciplinar. En el desarrollo del proyecto colaboraron investigadores de Colombia y Bélgica, con producción intelectual en forma de Conferencia y artículo en revista indexada.

Capítulo 2. Estado del Arte

Aplicaciones en Telecomunicaciones

La teoría de colas tiene múltiples aplicaciones en los sistemas de telecomunicaciones, como los Switches o Conmutadores (W. Wang, Dong, & Wolf, 2002), o utilizando estrategias del tipo de la simulación de eventos discretos (Larocque & Lipoff, 1996), para evaluar el rendimiento de sistemas con tráfico mixto, o como en los buffers, los cuales pueden analizarse mediante teoría de colas (S. A. A. Shah, Shah, Rind, & Das Menghwar, 2009), o los sistemas digitales de comunicaciones, que se pueden analizar mediante métodos en tiempo discreto (W. Shah, Shah, Rind, & Das Menghwar, 2009), o en el estudio del tráfico del protocolo TCP (Raina & Wischik, 2005). Igualmente tienen cabida en el análisis de retardos en sistemas que funcionan bajo slots (Rubin & Ouaily, 1989), así como en el proceso de determinar cuáles servicios están disponibles en redes móviles ad hoc (MANET) (Wen, Jin, Li, Cheng, & Fang, 2012).

Del mismo modo se puede usar la teoría de colas junto con la modulación adaptativa y codificación (AMC), con el fin de contrarrestar propiedades variables en el tiempo del canal inalámbrico (Lin, Kwok, & Wang, 2007). Ciertos sistemas actuales de retardos basados en fibra óptica, pueden optimizarse mediante la teoría de colas (Kurt Van Hautegem, Rogiest, & Bruneel, 2014a). Es evidente la relevancia de los problemas inversos en teoría de colas, con los que se pretende encontrar los parámetros desconocidos del sistema a partir de trayectorias observadas parcialmente (Baccelli, Kauffmann, & Veitch, 2009). Basándose en una rama de la teoría de colas, conocida como teoría de colas acusatoria, en (Bonifaci, 2007) se desarrolló un modelo nuevo para el enrutamiento de servidor online. En cuanto a los servicios de telecomunicaciones, en particular la transferencia de archivos, el Video sobre demanda y la voz sobre IP, en (Nagy, Tombal, & Novotny, 2013) se ha implementado un modelo diseñado para simular dichos servicios, empleando teoría de colas. En (Mewara & Manghnani, 2015) se estudiaron los efectos y el desempeño de tres técnicas de colas (Primero en llegar primero en salir de cola, Cola de prioridad, cola de promedio justo) las cuales permiten el soporte de VOIP (Voz sobre Protocolo de Internet) en WiMAX (Interoperabilidad Global para acceso en microondas), utilizando la herramienta de simulación OPNET 14.5. Buscando obtener ganancias optimas en logística en un ambiente de nube, en (Jiang, Hsu, & Wang, 2017) fue usada la teoría de colas de fuente finita. Mediante teoría de colas en (Alqaydi, Salah, & Zemerly, 2015) fue posible sostener un objeto de nivel de servicio (SLO, Service Level Object) en aplicaciones almacenadas en la nube y en servicios, con un uso mínimo de recursos, que se usaron en su máxima capacidad. Existen diversos algoritmos enfocados en la optimización del ancho de banda en los casos en los que la Videoconferencia (VC) y la navegación en Internet son simultáneos (Hassan & Abdalla, 2015). En

(Hassan & Abdalla, 2015) se examinaron tres métodos: Ancho de banda garantizado o Cola típica (CQ, Custom Queueing), Cola ponderada justa (Weighted Fair Queueing, WFQ)y una combinación de los dos (Hassan & Abdalla, 2015). Estos métodos se aplicaron en la Red de Educación e Investigación Sudanesa (Hassan & Abdalla, 2015). En (Wigren, 2016) se diseñó un controlador en red para el control a nivel de cola de paquete de datos a través de dos nodos usando conexión a Internet. El vaciado de la cola de paquete se hizo por medio de una interfaz inalámbrica de radio (Wigren, 2016). Con el objetivo de tener un soporte de QoS extremo a extremo en servicios de aplicación a través de redes convergentes en 5G, los autores de (Al-Shaikhli & Esmailpour, 2015) han propuesto un marco de QoS común. Symbiot es un algoritmo distribuido y ligero usado para garantizar justicia multi-recursos entre usuarios de redes sensoriales multi-usuario (Tahir, Yang, Adeel, & McCann, 2015). Este algoritmo fue creado usando teoría de grafos, teoría de colas y el concepto de acciones de recurso dominante(Tahir et al., 2015).

En los sistemas de tráfico mixto se manejan al mismo tiempo voz y datos, con el fin de evitar interferencias se deben utilizar canales independientes, que no se traslapen entre sí (Larocque & Lipoff, 1996).

Análisis de sistemas basados en teoría de colas

Cuando se tienen sistemas usando tiempo discreto, este se divide en slots, y se habla de sincronía (Mowcheng Lee, 1995). No obstante, en ciertos casos se puede trabajar asíncronamente, con longitudes arbitrarias de paquetes y diferentes velocidades y retardos de propagación, esto se conoce como Teoría de Colas Acusatoria Continua (CAQT) (Blesa et al., 2009).

Otra de las variaciones que se pueden hacer en los comportamientos típicos en las colas, se describe en (Pazgal & Radas, 2008), donde los clientes pueden regresar a la cola, después de cierto tiempo.

Aunque generalmente se asume que los tiempos entre llegadas en teoría de colas siguen distribuciones exponenciales y por lo tanto son independientes, también existen casos en los que esto no se cumple, de manera que se tienen dependencias o correlaciones entre dichos procesos(Rodríguez, Lillo, & Ramírez-Cobo, 2016). Otro de los cambios en teoría de colas se da cuando se puede tener más de una entrada o salida, esto se conoce como lotes (Jiafu He & Sohraby, 2003), (Y. Wang et al., 2014), (Rodríguez et al., 2016).

Para el manejo de sistemas de transporte, de acuerdo a lo expuesto en (Y. Wang et al., 2014), se tienen medios de transporte alterno. Cuando el servicio principal de transporte, en este caso el que hacen los trenes, fallan se necesitan medios alternos para continuar con el servicio (Y. Wang et al., 2014).

Aparte de los cambios descritos anteriormente se tienen ciertas variantes como cuando no se cuenta con datos precisos en los modelos de colas, en ese caso se pueden hacer aproximaciones obteniendo valores fuzzy (Fuente & Pardo, 2009). Igualmente se puede usar la Teoría Fuzzy de conjuntos para realizar una descripción de los modelos de colas más aproximada a la realidad (Pardo & de la Fuente, 2007). Surge, la teoría Fuzzy de colas, con la que se pueden modelar sistemas de colas, teniendo en cuenta la imprecisión de dichos sistemas(Muñoz & Ruspini, 2014). Se ha relacionado la entropía con la incertidumbre, la cual es típica en los sistemas de colas, con el fin de determinar el desempeño operativo de dichos sistemas (Tiwari, Agrawal, Kapoor, & Chauhan, 2011).

Tan importantes como las variaciones en los sistemas de colas son los criterios de evaluación de las mismas, en general con ese fin, se utilizan medidas probabilísticas (Y. Wang et al., 2014). Uno de los comportamientos evaluados es la estabilidad de los sistemas (Gao, Chen, Zheng, & Zhu, 2014). Se ha encontrado que la precisión de los estimados de las medidas de desempeño depende de ciertas condiciones cambiantes (Rabta, Schodl, Reiner, & Fichtinger, 2013).

En teoría de colas son importantes, para determinar el desempeño de los sistemas mencionados, la longitud de cola y el tiempo de espera (Mowcheng Lee, 1995), y se trabaja con las distribuciones de los mismos (Mowcheng Lee, 1995). Cuando se requiere describir el desempeño o comportamiento de variables como la calidad de servicio se plantean modelos matemáticos(Pandit, Schmittt, & Steinmetz, 2004). Muchos de esos modelos se contrastan o se utilizan junto con la teoría de colas (Caixia Li, Anavatti, & Ray, 2013).

Una vez se ha descrito la evaluación de los sistemas de colas se hará una descripción de la aplicación de ellas. Entre estas aplicaciones se tiene la asignación de tráfico, para tal fin se usa junto con herramientas como la teoría de juegos (Caixia Li et al., 2013). Asimismo, la teoría de colas, se combina con la asignación de costos, para determinar el comportamiento de las colas (Ravner, 2014) . Otra de las aplicaciones de la teoría de colas se enfoca en aligerar el costo computacional de cálculos como los del análisis de punto de equilibrio (Barcelo & Ramon, 1997).

Se combinan diferentes conceptos matemáticos como las funciones generadoras multidimensionales con el análisis combinatorio, para entender el comportamiento de los sistemas de colas(Jiafu He & Sohraby, 2003). Se utilizan conceptos como la ley del logaritmo iterado para determinar métricas significativas en el sistema de colas (Minkevičius, Dolgopolovas, & Sakalauskas, 2016). También se ha relacionado la teoría de colas con la de fiabilidad (Tsitsiashvili & Osipova, 2008). Se ha encontrado que la dinámica particular de un sistema de colas se puede describir mediante conceptos como los factores de realización de perturbación(Xia & Cao, 2012).

Con el resultado de descomposición de los tiempos promedio de espera en una red de colas de prioridad relativa de dos estaciones fue posible tratar con problemas de control óptimo, muy comunes en el diseño de redes de comunicaciones D2D (Mayekar, Venkateswaran, Gupta, & Hemachandra, 2015). En (Kakubava, Prangishvili, & Sokhadze, 2016) se propusieron modelos matemáticos para la "interacción de la degradación y sus procesos de compensación" para "sistemas de reserva". Dichos sistemas estaban compuestos por componentes reparables y no confiables(Kakubava et al., 2016).

En (Tong, Lu, Haenggi, & Poellabauer, 2016) se usó la teoría de colas con geometría estocástica para formular un modelo matemático que describe el desempeño de IEEE 802.11p.

En (Gupta & Hemachandra, 2015) los investigadores estudiaron las leyes de conservación y la región alcanzable para las probabilidades de cola del tiempo promedio de espera en una cola M/G/1 de dos clases para un valor específico de cola.

Manejo de la contención en redes ópticas

En el caso en el que hay diferentes paquetes compitiendo por la misma salida al mismo tiempo, es necesario usar un mecanismo de resolución de la contención. Con este mecanismo los paquetes se retardan usando líneas de retardo de fibra (FDLs), las cuales actúan como buffers para los paquetes. Con el fin de minimizar la pérdida de paquetes, usando las FDLs, diferentes autores han propuesto varios algoritmos (Kurt Van Hautegem, Rogiest, & Bruneel, 2014b).

Con el propósito de satisfacer las demandas crecientes en redes de telecomunicaciones, la infraestructura de soporte ha crecido, usando dos alternativas a la conmutación óptica de circuitos: conmutación óptica de paquetes y conmutación óptica de ráfagas. De esta forma, el uso de la capacidad de la fibra óptica es mejor. Otra estrategia implementada y usada en las redes de telecomunicaciones es la contención de paquetes. Para tratar con la contención de paquetes, líneas de retardo de fibra se usan junto con conversores de longitud de onda, haciendo posible la reprogramación de paquetes. Los algoritmos existentes en conmutación óptica, asumen un número infinito de conversores. En contraste los algoritmos de programación propuestos en (Kurt Van Hautegem et al., 2014a), son paramétricos y basados en costo, teniendo en cuenta la escasez de FDLs y conversores. Además de la conmutación, es necesaria la programación, en este caso mediante las FDLs. Los dominios de programación, usados en (Kurt Van Hautegem et al., 2014a) son los de la longitud de onda y el tiempo. Otra estrategia adicional a la programación, es la conversión de longitudes de onda. Sin embargo, hay una restricción con el número limitado de conversores de longitud de onda, WCs (Kurt Van Hautegem et al., 2014a). El propósito de (W Rogiest & Bruneel, 2010) es proponer un método para optimizar los buffers basados en FDLs, y la longitud de las fibras.

En (Tatxé, 2010) se propuso un simulador de un programador óptico. El simulador trabajó con cuatro algoritmos (DG-VF, GD-VF, DG sin VF y GD sin VF). Con el simulador se pretendía encontrar las probabilidades de pérdida para cada uno de los cuatro casos mencionados, trabajando de forma modular.

El fin de (F Callegati et al., 2006) es generar una metodología de comparación para algoritmos de programación usados en la resolución de la congestión. El énfasis de (F Callegati et al., 2006) son las capacidades de conversión de longitud de onda.

Aunque el consumo de energía es menor en los conmutadores ópticos que en los electrónicos, dicho consumo es considerable (Kurt Van Hautegem, Rogiest, & Bruneel, 2015b). Considerando que el aporte de los conversores de longitud de onda al consumo global de energía es alto, se concluye que solo deben encenderse lo estrictamente necesario(Kurt Van Hautegem et al., 2015b).

Los algoritmos tradicionales generalmente buscan reducir la pérdida de paquetes utilizando un número infinito de conversores de longitudes de onda, y tienen la limitación de reducir su rendimiento con un reducido número de conversores (Kurt Van Hautegem, Rogiest, & Bruneel, 2015a) como generalmente sucede en la mayoría de arquitecturas de conmutadores. En (Kurt Van Hautegem et al., 2015a) se proponen diversos algoritmos de programación basados en costo utilizando un número limitado de conversores de longitud de onda.

Estado de la técnica

En (Rossiter Margaret, 2003) se presenta un sistema en el cual por lo menos un servidor debe atender varias colas y se hace una estimación del tiempo requerido para hacer el servicio, basándose en el conocimiento del servidor individual y en el tipo de servicio requerido.

En (Blake, 2008) se tiene un sistema de manejo de colas para tráfico descartable-elegible en un sistema de memoria compartida, con indicadores justos por clase.

En (Davies Neil James; Holyer, 2004) se asignan niveles de prioridad a paquetes de datos en función de las clases de perdida y de servicio del paquete de datos y de la carga actual del flujo de información.

En (Galbraith Simon David; Davidson, 2005) se evalúa el nivel de frustración del usuario de acuerdo al retardo experimentado. Con esos datos se puede generar una representación del sistema.

En (Cheng-Jia, 2010) se propuso un método para manejar una cola compartida que atiende un grupo de clústeres, para la que cada uno de los servidores mantiene una cola local, y el conjunto de colas conforman una cola unificada atendiendo todos los servidores en los clústeres. Las peticiones en cierta

cola sobrepoblada se reenvían a otras colas, de forma aleatoria y con un método de intercambio de mensajes (Cheng-Jia, 2010).

En (Corynen, 2004) se desarrolló un sistema para trabajar con problemas de decisión, realizar análisis de riesgos y optimizar la operación de ciertos sistemas y procesos. Se pretende que la aplicación de (Corynen, 2004) corra en un computador personal. La aplicación de (Corynen, 2004) consta de tres métodos. El primer método se orienta a proyectos y trabaja con procedimientos modularizados con el objetivo de resolver y analizar problemas de decisión secuenciales, probabilísticos y multi-objetivo (Corynen, 2004). El segundo método generaliza al primero y tiene el propósito de optimizar un proceso o sistema donde su operación implique decisiones probabilísticas y multiobjetivo que no se puedan tratar con métodos lineales convencionales de control, no lineales y estocásticos (Corynen, 2004). El tercer método se centra en una interfaz gráfica de usuario, multi-ventana, para facilitar el uso de los dos primeros métodos, por otra parte, contiene procedimientos para manejar y cambiar salidas de texto y gráficas (Corynen, 2004).

En (McKenney, 2015) para el caso de un sistema de cómputo con uno o más procesadores acoplados operativamente a uno o más dispositivos de memoria, se propone un método para baja sobrecarga de conmutación basado en contención, cambiando entre bloqueo de tiquetes y bloqueo de colas con el fin de acceder a datos compartidos en uno o más dispositivos de memoria.

En (Backer, 2017) se propuso un método implementado por computador para controlar una cola utilizada en servicios. El sistema está basado en Software y gestiona tanto una cola virtual como una cola real, con una correspondencia entre ellas. Para las comunicaciones entre el sistema de gestión de colas y el usuario hay alternativas como una red celular o una red WiFi. En los procesos de comunicación es importante determinar la ubicación para un servicio deseado (Backer, 2017).

En (Popkey Robert; Rzeznik, 2017) se propuso un Sistema de gestión de transacción automatizado para línea de espera. Para trabajar existen conexiones entre patrones, atracciones, IDs y servidor por medio de una red. En el proceso de ingreso se utilizaron tanto bandas para la muñeca como RFID. En particular la aplicación se enfocó para funcionar en un parque de diversiones. El Sistema registra variables tales como la espera actual. El método propuesto se centró en saltarse una cola de espera para una atracción, al pagar una cantidad específica de dinero. Entre las opciones de pago se consideró la tarjeta de crédito. Obviamente se requieren fondos suficientes para aceptar las transacciones. Una característica importante es la generación de estadísticas de los usuarios de línea (Popkey Robert; Rzeznik, 2017).

En (Sussman, 2017) se propone un Sistema y método para procesar eficientemente y gestionar datos almacenados en una cola. Se utilizaron protocolos para procesar y gestionar datos almacenados en la cola. Estos protocolos permiten un uso eficiente de los recursos. Al menos el primer protocolo es usado para trasferir datos a otra cola y otro servicio, el Segundo protocolo inhibe la trasferencia de datos a otra cola. La comunicación es hecha a través de una red. El sistema está compuesto por un dispositivo suministrador de eventos, un Sistema intermediario, un Sistema de sitio, un Sistema de gestión de acceso, un registro de cliente, un dispositivo de punto de cliente, y un dispositivo de agente cliente. Las capas consideradas en el diseño fueron: aplicación, transporte, IP y enlace. El Sistema está basado en la asignación de derechos de acceso (Sussman, 2017).

En (Le Thai Franck;Nahum, 2017) se propuso un Método de gestión de colas, Sistema y medio de grabación. Los componentes son: un dispositivo de extracción, un dispositivo examinador SACK (Reconocimiento Selectivo), un dispositivo examinador MPTCP (Protocolo de Control de Transmisión Multi-Trayectoria), un dispositivo examinador de cola, un dispositivo de caída, un dispositivo para establecer prioridades, y adicionalmente un procesador y una memoria. Una cola es usada para contener paquetes. También se requiere un expedidor basado en reconocimientos. Al Sistema pueden acceder computadores portátiles, computadores de escritorio, teléfonos inteligentes y automóviles, entre otros. Estos se comunican utilizando redes. Hay aspectos clave en el funcionamiento, tales como: Cargas de trabajo, gestión, virtualización, Hardware y Software. El objetivo es hacer una gestión de colas en dispositivos en red tales como puentes, enrutadores, y ciertos tipos de accesorios de función de red. La propuesta fue un método de gestión de colas, que comprendía: el examen de un flujo de cola reverso desde un expedidor para un paquete reconocido; y el descarte de un paquete en la cola de flujo de expedición que incluye el paquete reconocido en la cola del flujo de expedición (Le Thai Franck;Nahum, 2017).

En (Monk, 2015) se propone un esquema de gestión de colas por lo cual el estado del Sistema es considerado junto con un estado de cada cola particular o específica y el dispositivo asociado dentro del Sistema, con el fin de balancear eficiente y efectivamente las necesidades y deseos de una entidad que suministra servicios a los clientes.

En (Monk, 2015) se designó un método para monitorear, por un Sistema de cómputo, una cola de cliente para un dispositivo de venta de entradas de transito de servicio, en un Sistema de transito; determinando, por el Sistema de cómputo basado en el monitoreo, que un parámetro asociado con el tiempo de espera del cliente, para interactuar con el dispositivo de venta de entradas de transito de servicio, es mayor o igual que un valor particular de umbral; y en respuesta a la determinación, se controla el tipo del contenido de salida para presentación por el dispositivo de venta de entradas de

transito de servicio, para reducir el tiempo de espera del cliente, para interactuar con el dispositivo de venta de entradas de tránsito en servicio (Monk, 2015).

En (Sanchis Joseph; Muniz, 2016) se propuso un Sistema para suministrar servicios de gestión de colas virtuales. Estos servicios son personalizados a través de un sitio web y/o una API. Los servicios permiten la gestión de colas virtuales, reordenadas dinámicamente basándose en el comportamiento de los participantes en la cola. Los servicios les permiten a los participantes en la cola observar la posición relativa o absoluta de otros participantes en la cola virtual. Los servicios permiten la interacción en la cola virtual por medio de dispositivos de cómputo móviles o fijos, tanto en el sitio o remotamente, a partir de la ubicación de la cola. Existen componentes tales como: base de datos, servidor, dispositivo de usuario final (puede ser móvil), red, y componentes de terceros tales como: Servidor, dispositivo y base de datos. Con el fin de permitir al usuario iniciar sesión para usar la cola, se emplearon puntos q-tech.

En (Meiri Dror; Shazar, 2016) se propuso un Sistema y método para la gestión automática de colas de recursos. El método incluye la recepción de una selección de un recurso no disponible, a partir de la selección de una entidad de usuario, donde el recurso seleccionado está asociado con un orden de cola incluyendo al menos una entidad en cola; la actualización del orden en la cola del recurso seleccionado basada en la selección recibida; Se realiza un monitoreo de la información de disponibilidad de recursos con respecto al recurso seleccionado para identificar cambios en la disponibilidad del recurso seleccionado; y al detectar un cambio en la disponibilidad del recurso seleccionado, se asigna automáticamente el recurso disponible detectado a la siguiente entidad en cola en el orden de cola; para luego actualizar el orden de cola basado en la asignación. El diagrama de red contiene diversos elementos, tales como: Fuentes web, una base de datos, una red para conectar diferentes dispositivos, un servidor en el que se incluyen un Sistema de procesamiento y una memoria, y dispositivos de usuario. Los mapas ejemplares de recursos pueden contener diferentes estados: ocupado, disponible, en cola y asignado.

La patente en (X. Wang, 2016) propone mecanismos y un método para facilitar una gestión jerárquica y dinámica de recursos de cola en un ambiente de servicios sobre-demanda, en un ambiente de múltiples usuarios, de acuerdo con una realización. En una realización y por forma de ejemplo, un método incluye la asignación, en tiempo de ejecución, por el Sistema de base de datos, pesos para al menos uno de la pluralidad de usuarios, y uno o más de la pluralidad de tipos de mensaje, basados en uno o más de los pesos asignados. Los recursos asignados se pueden modificar dinámicamente en tiempo de ejecución, basándose en el escalamiento de los pesos asignados (X. Wang, 2016).

En la patente de (Weber Kurt;Cantonnet, 2015) se propuso un método implementado por computador, el cual reprograma citas médicas. El método comprende el almacenamiento de la información de programación de citas, asociada con un suministrador médico y la recepción de una indicación de que una vacante en las citas está disponible. Un paciente es seleccionado para llenar el espacio disponible basándose en un análisis de la información de programación de citas y en los datos del paciente o los datos del doctor. El método luego transmite una notificación del espacio disponible al paciente (Weber Kurt;Cantonnet, 2015).

En (Williams, 2010) se propuso un sistema de formación de cola virtual junto con un método con el fin de generar un control dinámico de datos en función de instrucciones de control de cola suministradas por una fuente de control de cola aparte. El sistema de formación de cola virtual está compuesto por una interfase dedicada a un sistema de formación de cola y por una interfase destinada a la fuente de control de cola separada (Williams, 2010). La interfase para el sistema de formación de cola se diseñó con el fin de obtener datos de la cola y controlarlos a partir de las instrucciones de cola suministradas por una fuente de control de cola separada. La interfase para la fuente de control de cola separada tiene como objetivos: i) entregar los datos de cola a la fuente de control de cola separada; así como ii) recibir las instrucciones de control de cola desde ella misma (Williams, 2010).

En (Holliday, 2010) se propone un sistema de gestión de colas, que puede ser utilizado en las cajas de supermercados, basándose en dispositivos de conteo ubicados en las cajas así como en las entradas y salidas, considerando simultáneamente información de los puntos de venta, con el fin de generar un estimativo de la cantidad requerida de cajas, de tal modo que se evite que las longitudes de cola superen ciertos límites preestablecidos (Holliday, 2010). El sistema cuenta con un sistema de aprendizaje dinámico para optimizar las programaciones obtenidas, de acuerdo con los registros de datos históricos (Holliday, 2010).

La propuesta de (Jensen, 2006) busca ubicar dinámicamente llamadas recibidas recientemente en una cola de colas establecidas, en un call center. La ubicación dentro de la cola se hace de acuerdo a objetivos de servicio preestablecidos, asignados a los tipos de colas soportados por el call center, así como a los periodos de tiempo de espera de las otras llamadas, ubicadas en la cola (Jensen, 2006). En el momento en que se recibe una llamada en el call center, se realiza un análisis considerando posiciones de cola individuales dentro de la cola, y este análisis finaliza cuando se localiza una posición en cola, la cual cumple con cierto criterio de selección preestablecido. Posteriormente, la nueva llamada es asignada a esta posición en cola, y las siguientes llamadas en la cola se mueven correspondientemente (Jensen, 2006).

En (Sahlin, 2003) se presenta un método para prestar uno o más tipos de servicios, en por lo menos una estación de trabajo, de forma ordenada, con un sistema de gestión de colas. Los clientes reciben un escaneo biométrico con el fin de encontrar correspondencia en una base de datos la cual contiene información de los clientes. Cuando hay correspondencia en la base de datos, a partir de la información en la base de datos se puede enviar por lo menos una identidad a una estación de trabajo apropiada, en caso de que el cliente haya sido convocado a esa estación de trabajo (Sahlin, 2003). Por otra parte, también es factible enlazar un escaneo biométrico, el cual no coincide con ninguna entrada en la base de datos, con una persona que hace negocios en la estación de trabajo, una vez que la persona se identifique, y de esa forma se genera una nueva entrada en la base de datos (Sahlin, 2003).

Por medio de un sistema electrónico, en (Salas Fehlandt, 2014) se puede hacer la administración de filas, así como la medición y la entrega periódica de información a usuarios finales de servicios. Entre la información entregada se tienen los tiempos requeridos de espera en fila, para recibir atención, el número correspondiente de turno. Dicha información es útil para las personas que se encuentren esperando en la fila respectiva (Salas Fehlandt, 2014).

En (Buschi Giovanni;Coulomb, 2006) se muestra un sistema y método para hacer una gestión inteligente de cola, empleando un localizador de destino virtual, con el fin de orientar a cualquier miembro de la cola por medio de una asignación de destino local preestablecida. La guía se encarga de organizar los miembros de la cola en una secuencia, y posteriormente los guía para que alcancen su destino final, moviendo el destino virtual de acuerdo con la secuencia (Buschi Giovanni;Coulomb, 2006).

Capítulo 3: Metodología

Con el fin de desarrollar y especificar un sistema de gestión de colas, es necesario conocer sus características particulares. Estas características son los modelos de colas y las políticas de programación. En los modelos de colas es necesario especificar las distribuciones entre llegadas, las distribuciones de los tiempos de servicios y el número de servidores. Las políticas de programación, como Primero en Entrar Primero en Ser Servido, especifica el orden de atención de los clientes. En algunos casos, para modelos como el M/M/1, hay fórmulas analíticas para el numero promedio de clientes en el sistema, la varianza del número de clientes en el sistema, el tiempo promedio en el sistema y el tiempo promedio en la cola (Harchol-Balter, 2012). Basados en estas fórmulas analíticas pueden determinarse la utilización y el rendimiento (número de trabajos por unidad de tiempo) (Harchol-Balter, 2012). En muchos casos los modelos pueden hacerse con cadenas de Markov, si se cumple la propiedad de falta de memoria (Harchol-Balter, 2012). No obstante, cuando existen cargas no Markovianas, frecuentemente estas cargas pueden ser aproximadas por mezclas de distribuciones exponenciales (Harchol-Balter, 2012). El conocimiento de estas características será usado como punto de referencia para las mediciones.

Para predecir el desempeño del sistema de gestión de colas, se utilizarán modelos de colas. La selección de los modelos debe basarse en la caracterización de los sistemas de colas, especificando las distribuciones entre llegadas, las distribuciones de los tiempos de servicio y el número de servidores. La caracterización se basará en mediciones de los tiempos de llegadas, los tiempos de servicio y el número de servidores. El objetivo de las mediciones es recolectar los datos reales que definen el sistema donde se aplicara la gestión. Las mediciones se harán en los lugares de clientes potenciales. Basados en las mediciones se calculará el tiempo de espera, el rendimiento y las probabilidades de perdida (proporción de clientes perdidos con respecto al total de clientes atendidos), en periodos de un mes, teniendo en cuenta días individuales y horas con el objetivo de estimar los costos y ganancias, con el sistema. Las mediciones se registrarán en software como Excel, organizadas en tablas, con el tiempo como variable independiente y las métricas (tiempo de espera, rendimiento y probabilidad de perdida) como variables dependientes. Estas mediciones se compararán con las opciones teóricas, obtenidas a partir de las fórmulas analíticas. Para la comparación se utilizará el promedio de los errores individuales, donde cada error individual viene definido por:

$$\text{Error} = \left| \frac{\text{Valor teorico} - \text{Valor medido}}{\text{Valor teorico}} \right| * 100\%$$

Estos puntos de datos reales, obtenidos a partir de las mediciones, se analizarán con el fin de determinar el modelo que mejor se ajusta a los datos y a la política de programación mas apropiada. Entre múltiples opciones, solo un modelo se escogerá, así como hay muchas políticas de programación disponibles.

El objetivo del análisis es encontrar la conexión entre los datos reales (obtenidos a partir de las mediciones) y los modelos teóricos, con el fin de escoger la política de programación mas apropiada, en el sistema de gestión de colas. Esta política de programación se adaptará a las condiciones específicas (tiempos entre llegadas y de servicio, numero de servidores). Estos modelos seleccionados se considerarán junto con los algoritmos, para la implementación de un desarrollo de Software.

El propósito de la implementación de Software es generar una respuesta automática basada en las condiciones (tiempos entre llegadas y de servicio, numero de servidores) que optimice las métricas de desempeño (tiempo de espera, probabilidad de perdida, rendimiento). Con esta implementación se asegurará un nivel deseado de desempeño en un sistema de servicio particular. En el desarrollo se considerarán los modelos seleccionados y las políticas de programación, junto con los algoritmos, para trasladar estos elementos a código. Cada módulo se implementará individualmente, y después de su desarrollo se depurará. A continuación, estos módulos se integrarán en un sistema funcional, y finalmente el sistema completo será depurado. El sistema de software se evaluará para alcanzar un nivel óptimo de desempeño, considerando las métricas correspondientes.

El objetivo de la evaluación es determinar si el sistema de gestión de colas, implementado en software, cumple con los requerimientos, y si estará adaptado a las condiciones de operación específicas y variables. El criterio de evaluación será el error promedio, calculado a partir de

$$\text{Error} = \left|\frac{\text{Valor simulado} - \text{Valor medido}}{\text{Valor simulado}}\right| * 100\%$$

Las simulaciones usaran millones de números aleatorios (conocidas como simulaciones de Monte Carlo) aplicadas a una estrategia de eventos discretos, donde las transiciones entre estados se hacen dependiendo de los valores de ciertas variables. Una vez se conoce la política de programación, se define una agenda la cual considera los tiempos entre llegadas, y los tamaños de paquetes (relacionados con el tiempo de servicio). Estos tamaños de paquetes pueden ser fijos o variables, y son generados a partir de una distribución (generalmente uniforme). En cada paso, se actualiza la agenda, teniendo en cuenta los tiempos en los que ocurren los eventos. Después de millones de iteraciones (típicamente diez millones) se calculan los promedios de los tiempos de espera y de las

probabilidades de perdida. En el caso de que tanto los recursos como las recirculaciones son infinitos, no hay paquetes perdidos, entonces todas las probabilidades de perdida son cero

La evaluación del tiempo de espera promedio, la probabilidad de perdida, y el rendimiento se hará en cuatro escenarios, para una carga de 25%, 50%, 75% y 100%, estas cargas son significativas, considerando que en particular un 50% representa una condición típica de operación. El 100% representa el peor escenario posible, por lo que es importante analizarlo. Las opciones del 25% y 75%, son valores intermedios, que también contribuyen a la caracterización del sistema. Dado que las evaluaciones (tiempo promedio de perdida, probabilidad de perdida y rendimiento) se harán con datos reales, y considerando diferentes números de servidores, capacidades de cada servidor y políticas de programación, la evaluación tendrá una exactitud de 100%. Estas evaluaciones tendrán en cuenta todas las condiciones relevantes (número de servidores, tiempos entre llegadas y de servicio, políticas de programación), incluyendo los escenarios peor y mas probable, de manera que sean representativas. Otra estrategia importante es el cálculo de los intervalos de confianza, con el fin de especificar el rango de variación de las variables. Con estos intervalos se puede evaluar la respuesta del sistema, y la tolerancia a variaciones, de una forma exacta y precisa.

Cuando se tienen recursos limitados para atender a los clientes en diferentes sistemas, y la demanda de servicios es mayor a la capacidad de atención al cliente, necesariamente se generan colas. La gran mayoría de las empresas prestadoras de servicio de atención al cliente, cuentan con recursos limitados, teniendo en cuenta que, para generar utilidades razonables, debe haber restricciones en los costos debidos al uso de dichos recursos. Para optimizar las utilidades, se deben optimizar diversas métricas de desempeño, principalmente el tiempo de espera en cola, y relacionada con él, la proporción de clientes perdidos con respecto al total de clientes que ingresan al recinto para recibir servicio. El tiempo de espera en cola es el principal inconveniente para los usuarios del servicio, especialmente cuando dicho tiempo es particularmente extenso, de tal forma que se afecta negativamente la empresa prestadora del servicio. Por esta razón, si se reduce el tiempo promedio de espera en cola, se obtiene una ventaja competitiva frente a otras empresas prestadora de servicio, en las que el tiempo de espera sea mayor. Teniendo esto en cuenta con el proyecto se busca reducir al mínimo posible el tiempo de espera redundando tanto en beneficio del cliente como los de la empresa. Los clientes o usuarios finales de servicios, necesitan aprovechar el tiempo disponible, mientras las empresas que prestan los servicios de atención al cliente necesitan recursos financieros, los cuales aumentan proporcionalmente con la satisfacción del cliente. Una situación clave, para poder resolver el problema mencionado, es la modificación y adaptación de los algoritmos para tratar con la contención en buffers ópticos, de tal forma que los nuevos procedimientos propuestos puedan ser usados en

situaciones de logística, evidentemente en un contexto diferente. En los algoritmos usados en el manejo de la contención, hay conceptos como los vacíos (voids), los cuales no tienen un equivalente obvio en los servicios de atención al público con presencia de colas. Un problema científico importante, en el diseño de los algoritmos, es la identificación del modelo apropiado para el sistema de gestión de colas, antes de ser modificado y optimizado para tal fin. De esta forma se parte de conceptos de ciencia básica haciendo ciertas abstracciones para llegar a una aplicación en un campo diferente, para optimizar la atención a clientes físicos, en lugar de paquetes de datos.

Para el servicio de gestión de colas se debe tener en cuenta un equilibrio entre la capacidad de servicio y el costo de cola (Chase & Aquilano, 1973). Inicialmente el costo de esperar en una línea es máximo cuando la organización se encuentra en una capacidad de servicio mínima (Chase & Aquilano, 1973). En la medida en que la capacidad de servicio se incrementa, se da una reducción en el número de clientes en la línea, así como en sus tiempos de espera, lo cual implica una reducción en el costo de colas. El costo total óptimo se encuentra en la intersección entre las curvas de capacidad de servicio y línea de espera, como se puede observar en la siguiente figura, tomada de (Chase & Aquilano, 1973). De esta manera se logra un balance entre los costos de optimizar el servicio y los recursos requeridos. Un sistema con muchos servidores es inviable económicamente, y uno supremamente económico genera una gran insatisfacción en los clientes.

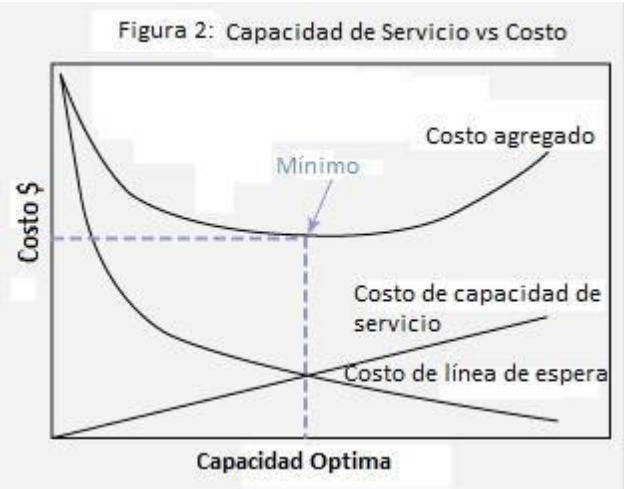

Figura 2: Capacidad de Servicio vs Costo. Tomado de (Chase & Aquilano, 1973).

Además del tiempo promedio de espera, con el sistema de gestión de colas se pueden mejorar otras métricas de desempeño tales como la probabilidad de pérdida (Relación entre los clientes perdidos y la totalidad de clientes que ingresan al sistema, que generalmente se presenta como un porcentaje) y el tamaño de pérdida (también conocido como LPSize, con el que se comparan el tamaño total de los

clientes perdidos con respecto al tamaño total de los clientes atendidos, el cual también se presenta en forma de porcentaje). De esta forma se puede tener una visión más completa de la calidad del servicio prestado, y se pueden diseñar e implementar estrategias para mejorar la atención al cliente.

Dependiendo de donde se utilice el sistema de gestión de colas, se pueden utilizar otras métricas de desempeño, adaptadas a los requerimientos de las empresas usuarias del servicio, y también se puede modificar la frecuencia con la que se almacenan dichas métricas en los registros respectivos.

Para lograr el mejoramiento, considerando la métrica de desempeño deseada, se usa una prioridad compuesta, la cual tiene en cuenta la importancia del cliente (con base en diversos criterios, como los ingresos generados por el cliente en particular) así como la urgencia de atención al mismo (clasificada en varios niveles, desde el más urgente al menos urgente). De esta forma se prioriza la atención a los clientes clave, y a aquellos que requieren atención inmediata. La prioridad compuesta es dinámica, de tal forma que puede cambiar en el tiempo, dado que la importancia en el tiempo, de los clientes, puede variar dependiendo de los ingresos generados (entre otros factores). El sistema de gestión de colas estará basado en módulos de software, cada uno con funciones únicas e independientes, y estos módulos interactuaran entre ellos para cumplir con las tareas requeridas. Los módulos se pueden adaptar a los requerimientos individuales de las empresas prestadoras de los servicios al cliente.

El sistema de gestión ha sido concebido para diversos tipos de colas (o líneas de espera), en diferentes ámbitos como en los bancos, supermercados, cines, aeropuertos, juzgados, oficinas del gobierno, entre otros. Todos estos tipos de colas, tienen características en común, no obstante, cada una de ellas tiene particularidades, por lo que se debe modificar el diseño propuesto, para optimizar el desempeño del sistema. Entre las particularidades se cuentan los tiempos entre llegadas (con sus distribuciones y parámetros específicos), los tiempos de servicio y el número de servidores. Estas tres variables se describen usando la notación de Kendall, y dependiendo de sus valores y distribuciones, existen modelos matemáticos para predecir y estudiar el comportamiento de las colas.

Con respecto al objeto de estudio, se tienen los algoritmos para el manejo de contención en buffers ópticos. La contención es un fenómeno que se presenta cuando dos o más paquetes, se encuentran simultáneamente en el mismo lugar, por lo que se requieren estrategias para minimizar la pérdida de paquetes. Estas estrategias han evolucionado desde FCFS (primero en llegar primero en ser servido), pasando por el llenado de vacíos (Void Filling), hasta la creación de vacíos. Con base en los conceptos y procedimientos, de los algoritmos de manejo de contención, se adaptan a colas de personas, considerando múltiples analogías, pero también abstracciones no evidentes, por ejemplo las recirculaciones (Si un paquete encuentra la salida ocupada, circula de nuevo por la línea de fibra

óptica, y posteriormente se revisa de nuevo la disponibilidad del medio, pueden existir límites en el número de recirculaciones, de tal forma que después de superado el límite, se descarta el paquete).

El contexto económico para la aplicación, inicialmente se centra en pequeñas y medianas empresas, donde se tenga el problema de la gestión ineficiente de colas, identificado por valores críticos en las métricas de desempeño, como el tiempo promedio de espera. La solución propuesta tendrá en cuenta el presupuesto de las empresas clientes, y con base en este presupuesto se definirá el nivel de sofisticación y complejidad del sistema de gestión de colas. Las empresas mencionadas, se dedican a la producción de bienes y/o servicios, en un sistema capitalista, con libre competencia. Sin embargo, sus recursos son limitados y se debe optimizar el manejo de estos recursos. Las empresas potenciales usuarias del servicio, en su gran mayoría se encuentran en ciudades, con gran densidad demográfica. En la población de estas ciudades, la mayor parte del capital se concentra en una minoría, por lo que los servicios prestados deben adaptarse a esta situación.

El sistema de gestión de colas afectara a la empresa usuaria del servicio de diversas formas. A continuación, se presenta una explicación de alto nivel de como la organización, herramientas, procesos, roles y responsabilidades serán afectadas como resultado de la implementación del proyecto.

Herramientas: El sistema de gestión de colas reemplazara al sistema tradicional, por lo que se requiere capacitación del personal en el manejo de la nueva herramienta, de modo que se garantice el manejo eficiente de la misma.

Procesos: Con la implementación del sistema de gestión de colas se agilizarán los procesos de atención al cliente en los servicios correspondientes. Los empleados encargados de atender al cliente cumplirán con sus responsabilidades de manera más eficiente y rápida.

Roles y responsabilidades: El proyecto propuesto redundará en mayor eficiencia por parte el personal de atención al cliente, y no requerirá nuevas funciones o roles dentro de la estructura organizacional de la empresa.

Hardware/Software: Además del licenciamiento del Software que va a gestionar los turnos en las colas, se requerirán computadores para automatizar las tareas, pantallas para visualizar los turnos actuales y en espera, así como impresoras para generar los turnos.

Capítulo 4. Resultados y Análisis

En la siguiente figura se presenta la diferencia en probabilidad de perdida (LP), obtenidas para los algoritmos WAS con respecto al VAS, para diferentes combinaciones de parámetros (bucles y recirculaciones).

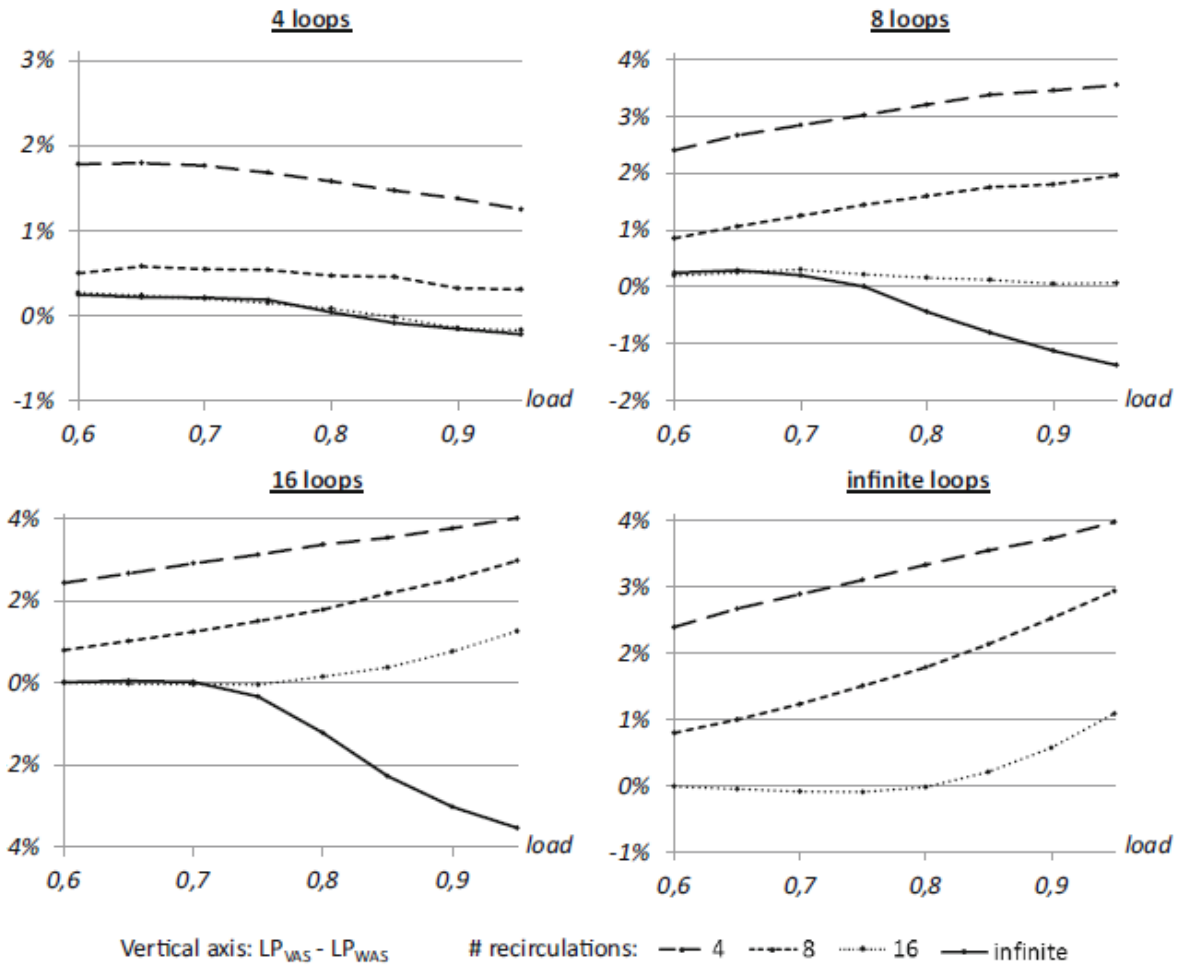

Figura 3: Diferencia en probabilidad de perdida entre VAS y WAS para diferentes combinaciones de parámetros. Tomada de (K. Van Hautegem, Pinto, Bruneel, & Rogiest, 2018)

Se puede observar que en la mayoría de los casos la diferencia entre las probabilidades de perdida del VAS y WAS es positiva, lo que evidencia la efectividad del algoritmo propuesto. También para la mayoría de los casos, al incrementar la carga, aumenta la diferencia entre las probabilidades de perdida, aunque no sucede lo mismo con el nivel de incremento en las probabilidades de perdida. Asimismo, se evidencia que, al aumentar las recirculaciones, aumenta la probabilidad de perdida, de tal forma que es deseable trabajar con menos recirculaciones, aunque parezca contraintuitivo.

La siguiente gráfica, tomada de (K. Van Hautegem et al., 2018) muestra las diferencias del tamaño de perdida (LPSize) para las diferentes combinaciones de los parámetros.

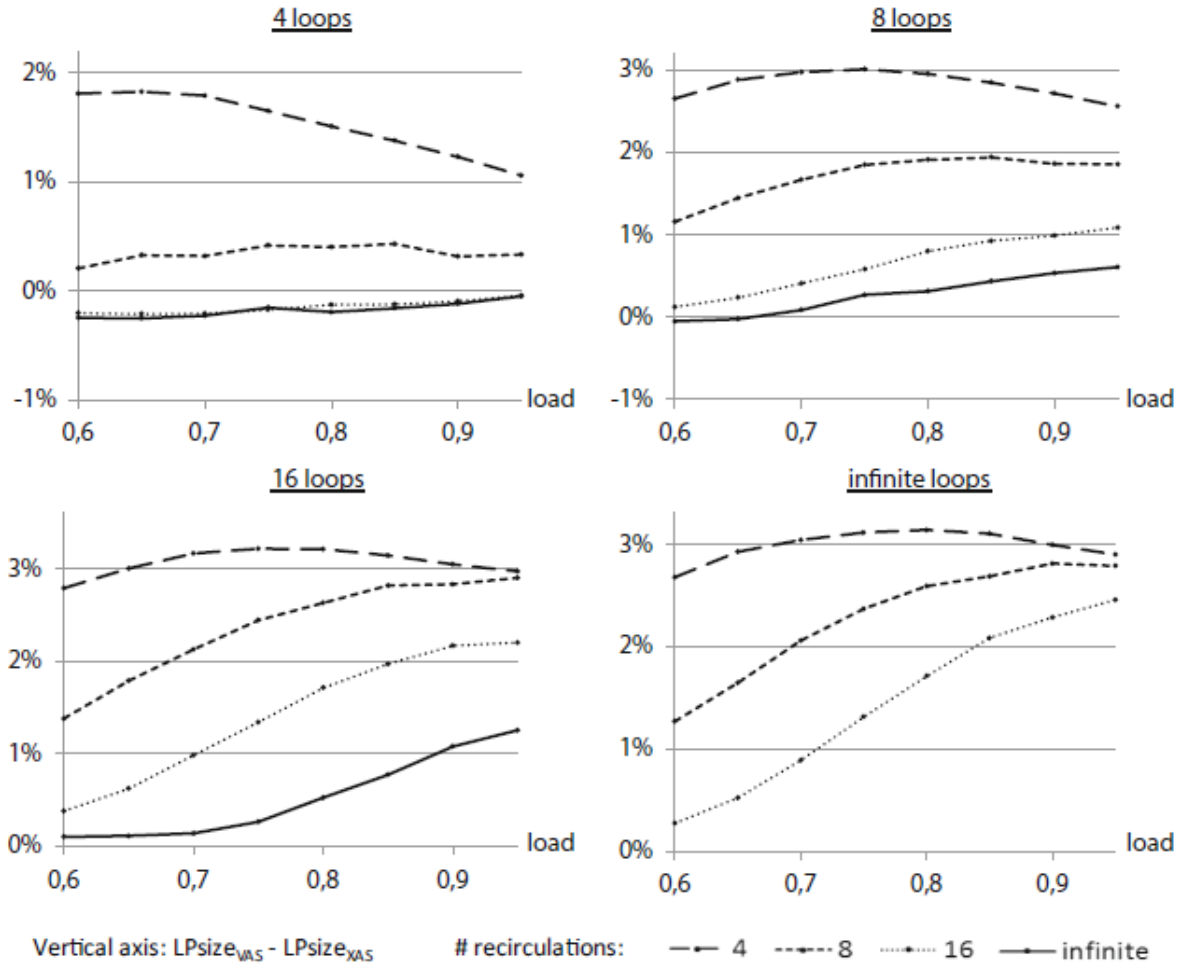

Figura 4: Diferencia en tamaño de perdida (LPSize) entre VAS y XAS para diferentes combinaciones de parámetros. Tomada de (K. Van Hautegem et al., 2018)

Se puede observar que para la gran mayoría de las curvas existe una mejoría con respecto a la referencia (VAS), y también, en general, al aumentar la carga se incrementa el porcentaje de diferencia inicialmente, y después se estabiliza o se reduce ligeramente dicha diferencia. Se puede apreciar que el incremento en las recirculaciones reduce el porcentaje de mejora, para un valor de carga dado, utilizando el mismo número de bucles, para hacer la comparación justa.

En el caso en que no haya restricciones ni en el número de recirculaciones ni en el numero de bucles, se tiene en cuenta el retardo del paquete, como lo muestra la siguiente figura, tomada de (K. Van Hautegem et al., 2018)

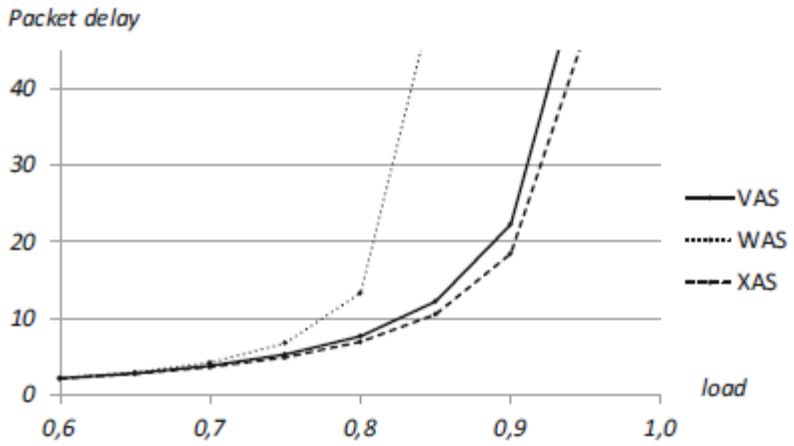

Figura 5: Retardo del paquete para los algoritmos VAS, WAS and XAS en un ambiente sin restricciones. Tomada de (K. Van Hautegem et al., 2018).

La siguiente tabla muestra la mejora en porcentaje de probabilidad de perdida (LP) del algoritmo T-WAS con respecto al VAS para diferentes valores de carga en diversas configuraciones de buffer.

T-WAS vs. VAS		# loops				# loops			
		4	8	16	∞	4	8	16	∞
		load = 0.60				load = 0.70			
max recirculations	4	17.3	27.3	27.7	27.4	11.3	21.4	22.0	21.9
	8	8.7	23.5	23.5	23.7	5.9	16.9	18.7	18.8
	16	6.5	18.9	8.5	7.4	4.2	10.4	8.3	6.3
	∞	6.1	25.3	64.0	NaN	4.2	10.6	27.2	NaN
		load = 0.80				load = 0.90			
max recirculations	4	7.8	17.7	18.7	18.6	5.5	15.0	16.6	16.5
	8	3.9	13.4	16.2	16.3	2.5	10.6	16.0	15.9
	16	2.5	5.0	9.7	8.4	1.4	2.9	11.0	11.3
	∞	2.4	2.0	4.9	NaN	1.3	0.2	0.3	NaN

Tabla 1: Mejora en porcentaje de la probabilidad de pérdida del algoritmo T-WAS con respecto al VAS para diferentes valores de la carga en diversas configuraciones limitadas del buffer. Tomada de (Kurt Van Hautegem, Pinto, Gomez, Bruneel, & Rogiest, 2020)

En la tabla se observa que se tienen mejoras para todas las combinaciones de recirculaciones con numero de bucles, en particular para bajas cargas y altos números de bucles, y bajo número de recirculaciones. Al aumentar la carga se reduce el porcentaje de mejora obtenido, dado que se está trabajando cada vez más cerca del límite de trabajo.

A continuación, se presentan las mejoras en porcentaje de tamaño de perdida, del algoritmo T-XAS con respecto a VAS, para diferentes valores de carga en varias configuraciones de buffer.

T-XAS vs. VAS		# loops				# loops			
		4	8	16	∞	4	8	16	∞
		load = 0.60				load = 0.70			
max recirculations	4	18.7	30.1	30.6	30.6	12.8	22.3	23.2	23.0
	8	7.4	32.6	37.9	37.6	6.7	23.1	29.9	30.1
	16	3.5	18.2	42.3	42.8	3.5	15.6	36.5	37.1
	∞	2.7	12.5	30.4	NaN	3.4	12.5	29.9	NaN
		load = 0.80				load = 0.90			
max recirculations	4	8.9	16.4	17.3	17.4	6.1	11.9	13.0	13.2
	8	5.4	16.7	22.8	23.0	3.9	11.5	16.9	17.4
	16	3.4	12.4	27.9	29.5	2.8	9.2	18.7	21.8
	∞	3.2	10.3	23.7	NaN	2.7	7.3	15.7	NaN

Tabla 2: Mejoras en porcentaje de tamaño de perdida, del algoritmo T-XAS con respecto a VAS, para diferentes valores de carga en varias configuraciones limitadas de buffer. Tomada de (Kurt Van Hautegem et al., 2020).

Se observa que las mayores mejoras se dan para una carga de 0.6, con bajo número de recirculaciones (8), y desempeño similar a partir de 8 bucles. Al aumentar la carga se reduce el porcentaje de mejora, dado que se reduce el margen de mejora posible. También en la gran mayoría de los casos al aumentar el numero de bucles para una carga y numero de recirculaciones fijos, se incrementa la mejoría posible, considerando una mejor utilización de los recursos posibles.

En el caso de un ambiente sin restricciones, la métrica utilizada es el retardo del paquete. En la siguiente tabla se muestra el retardo del paquete, el umbral optimo y el valor agregado del mecanismo de umbral para el algoritmo T-XAS considerando diferentes valores de carga en una configuración sin restricciones, basándose en (Kurt Van Hautegem et al., 2020)

Packet delay reduction	Load							
	0.60	0.65	0.70	0.75	0.80	0.85	0.90	0.95
XAS vs. VAS (%)	1.1	2.4	4.2	7.3	9.6	13.7	17.2	18.1
T-XAS vs. VAS (%)	6.3	8.1	10.1	12.5	14.1	17.1	18.6	18.1
optimal threshold	0.85	0.90	0.90	0.90	0.90	0.95	0.95	1
added value threshold (%)	83	70	58	42	32	20	8	0

Tabla 3: Retardo del paquete, el umbral optimo y el valor agregado del mecanismo de umbral para el algoritmo T-XAS considerando diferentes valores de carga en una configuración sin restricciones, tomado de (Kurt Van Hautegem et al., 2020).

Se puede apreciar que al aumentar la carga aumenta la reducción del retardo de paquete para el algoritmo T-XAS, así como se reduce la proporción entre las reducciones del T-XAS con respecto al XAS, haciéndose iguales para una carga de 0.95. El umbral optimo en la mayoría de los casos es menor a 1, variando entre 0.85 y 1. El umbral del valor agregado va decreciendo a medida que se incrementa la carga, hasta llegar a un valor de 0, para una carga de 0.95.

Capítulo 5. Producción

5.1 Productos

Teniendo en cuenta la naturaleza del Doctorado en Ciencia Aplicada de la Universidad Antonio Nariño, para la parte científica se proponen dos artículos, uno en Conferencia Internacional y uno en revista especializada. El artículo de la conferencia se enfocó en algoritmos para el manejo de contención en buffers ópticos, basándose en trabajo previo de algunos de los autores. En el artículo en revista especializada, se proponen extensiones de dos de los algoritmos presentados en el artículo de conferencia, los cuales también tienen aplicación en el manejo de la contención en buffers ópticos.

En cuanto a la parte tecnológica, el producto generado es una patente de un sistema de gestión de colas para reducir el tiempo promedio de espera en colas, aplicando de esta forma el conocimiento científico existente, en un sistema con potencial comercial.

Clase de Producto	Título	Medio de publicación	Clasificación	Fecha	Estado	Anexo Nro.
Artículo en Conferencia	Analysis of VAS, WAS and XAS Scheduling Algorithms for Fiber-Loop Optical Buffers	Libro de memorias del evento	-	Julio 25-27, 2018	Publicado	1
Artículo de Revista	T-WAS and T-XAS algorithms for Fiber-loop Optical buffers	Revista Optical Switching and Networking	Q2	Octubre 23, 2019	Publicado	2
Patente	Método y Sistema para reducir tiempos de procesamiento de solicitudes en líneas de espera	-	-	Junio 18, 2021	Radicado ante la SIC.	3

Tabla 4: Productos generados en el transcurso del Doctorado en Ciencia Aplicada.

Capítulo 6. Conclusiones

- Las aplicaciones de la teoría de colas cubren diversas áreas incluyendo los sistemas de comunicación, pasando por la genética y la investigación de operaciones. Con la teoría de colas se pueden diseñar y dimensionar múltiples sistemas. Las colas están presentes en diferentes aspectos de la vida cotidiana, en los que se requiere compartir recursos limitados entre varias entidades. Bajo ciertos principios generales es posible entender virtualmente todos los casos en los que se encuentran líneas de espera (colas).

- Con la teoría de colas es posible generar un diseño óptimo de diferentes sistemas, con la garantía de cierto nivel de desempeño, y mediante el uso sensato de los recursos disponibles. Entre estos sistemas se incluyen los buffers ópticos, por ejemplo. De esta forma se pueden aprovechar mejor los sistemas de comunicaciones basados en fibra óptica, alcanzando cada vez mayores niveles de desempeño.

- Las similitudes entre la teoría de colas y otras áreas se han explotado con el fin de comprender la operación de diversos sistemas, encontrando la relación entre parámetros y variables de dichos sistemas, las interacciones entre ellos, así como las dinámicas que las relacionan.

- La teoría de colas ha evolucionado durante mucho tiempo, adaptándose a las necesidades en diferentes épocas, dicha teoría intervino en las primeras comunicaciones telefónicas pasando por la televisión y el intercambio de información en redes de computadoras, como Internet.

- A la hora de diseñar sistemas basados en teoría de colas es necesario hacer un balance entre desempeño y complejidad de los sistemas, de modo que estos sistemas sean viables y eficientes.

- Existen diferentes métricas de desempeño de los sistemas de colas, tales como el tiempo de espera y la probabilidad de pérdida, los cuales permiten evaluar la eficiencia de estos sistemas. Cuando los recursos son ilimitados se utiliza el tiempo promedio de espera, para el caso de limitantes en los recursos, se emplean la probabilidad de pérdida o el LPSize (tamaño de pérdida). Adicionalmente se puede estimar la influencia de agentes o factores que afectan negativamente el funcionamiento de los sistemas de colas.

- La simulación mediante Software es una herramienta muy útil para analizar los sistemas de colas, y estudiar el comportamiento de estos sistemas y su respuesta a variaciones en las entradas y en los parámetros. También es una alternativa cuando una evaluación exhaustiva no es posible debido a la cantidad de recursos requeridos. Con el fin de verificar la validez de los resultados de simulación, estos se pueden comparar con los resultados analíticos.

- La complejidad de los modelos basados en colas se puede cambiar, con el fin de alcanzar cierto nivel de exactitud en el comportamiento de los sistemas modelados. Por ejemplo, hay diferentes variantes de modelos clásicos como el M/M/1, adaptados a los requerimientos actuales. Asimismo, los algoritmos empleados se pueden modificar para obtener una mejor aproximación.

- Cuando hay incertidumbre en los sistemas de colas, se pueden utilizar alternativas para manejar la incertidumbre, como las basadas en sistemas Fuzzy.

- El algoritmo WAS muestra una mejoría frente a alternativas como el VAS, usando la probabilidad de pérdida como métrica, en condiciones de tamaño de paquete distribuido uniformemente y tamaño limitado de buffer. Cuando la métrica es el tamaño de pérdida (LPSize), el algoritmo XAS supera al de referencia (VAS), en las condiciones mencionadas previamente.

- Bajo diversas condiciones de operación se encontró que el algoritmo T-WAS con respecto al VAS, mejora en términos de la probabilidad de perdida, en especial cuando las cargas son bajas y el número de bucles es alto, con bajos números de recirculaciones.

- Aunque hubo cierta mejora para el algoritmo T-XAS con respecto al VAS, el porcentaje de mejora, en términos de LPSize, se redujo al aumentar la carga, debido a que disminuye el margen de mejora posible. En la gran mayoría de los casos al incrementar el número de bucles con una carga y recirculaciones fijos, aumenta la mejoría posible, debido a un mejor uso de los recursos disponibles.
- Cuando no existen restricciones, en número de bucles y de recirculaciones, el retardo del paquete para el algoritmo T-XAS aumenta su reducción, al incrementar la carga. Asimismo, al incrementarse la carga se incrementa la reducción del retado del paquete, en el algoritmo

T-XAS, de la misma manera que disminuye con la proporción de las reducciones del T-XAS en comparación con el XAS.

- Aunque los algoritmos WAS y XAS son más complejos que los de referencia (VAS), su uso se justifica, considerando las mejorías obtenidas, y el aumento en la eficiencia del sistema.

- Los algoritmos de post-reservación, en los cuales no se reserva con anticipación el canal de salida, muestran mejor desempeño que los de pre-reservación, en los que se reserva el canal antes de utilizarlo. Los algoritmos WAS y XAS pertenecen a la categoría de post-reservación.

- La simulación de Monte Carlo permite evaluar con exactitud el comportamiento de los algoritmos propuestos, sin necesidad de utilizar sistemas físicos, en periodos de tiempo relativamente cortos. Sin embargo, para obtener simulaciones fiables se requieren abundantes recursos de Hardware, considerando los millones de casos estudiados, en cada simulación en particular.

- Como trabajo futuro, se tiene la consideración de más de dos paquetes simultáneamente, con el fin de escoger la mejor combinación, proponiendo de esta forma algoritmos modificados, y posiblemente mejorados.

- Las colas tienen un carácter universal, generado cuando la demanda de un servicio es mayor que la capacidad de atención al cliente, en un instante dado. Esta capacidad de servicio viene limitada por los recursos disponibles de atención al cliente, por lo que son necesarias estrategias para manejarla apropiadamente.

- Los algoritmos tradicionales de pre-reservación han avanzado a lo largo del tiempo pasando desde el primero en llegar, primero en ser servido, posteriormente los de llenado de vacío hasta los más recientes de creación de vacío. Aunque en general se mejoraron las métricas de desempeño, la complejidad de los nuevos algoritmos también es mayor.

- Para el diseño del sistema de gestión de colas, fue necesario abstraer conceptos de algoritmos usados en la contención óptica de paquetes, para aplicarse en sistemas de logística. Entre las abstracciones mencionadas se encuentran las de conceptos disimiles como los de recirculaciones, las cuales no tienen un equivalente obvio en sistemas de logística.

- Los sistemas de gestión de colas agilizan la atención a clientes en líneas de espera, incrementando su satisfacción, así como las utilidades y el prestigio de las empresas que adoptan dichos sistemas.

- Se desarrollo un documento de patente, el cual fue radicado ante la Superintendencia de Industria y Comercio. Con el método propuesto, en la patente, se pretende reducir los tiempos de procesamiento de solicitudes en líneas de espera, en servicios de alta complejidad operativa, donde los tiempos de procesamiento se refieren al tiempo transcurrido entre la llegada del cliente y el instante en que es atendido. El método y sistema se basa en el concepto de prioridad compuesta, donde se tiene en cuenta tanto la urgencia de atención como la importancia del cliente para la empresa prestadora del servicio.
- La prioridad, en el método y sistema, se asigna a partir de la identificación, empleando una serie de reglas, almacenadas en un dispositivo de memoria. Las reglas de prioridad pueden modificarse dependiendo de los requerimientos particulares, de la empresa prestadora del servicio. En la memoria también se tiene un conjunto de registros ordenados jerárquicamente, que se emplean en todo el proceso. Asimismo, además del conjunto de registros, existe un conjunto de variables de rendimiento, para evaluar el desempeño del método y sistema. Con base en los registros históricos, se puede optimizar la atención escogiendo los módulos de solicitudes más apropiados. Con respecto a la optimización, a partir de la modificación de las reglas de prioridad y otros procedimientos, se mejora la eficiencia en el uso de los recursos disponibles. Al mejorar la eficiencia mencionada se cumple con uno de los propósitos del método y sistema, el cual es el incremento de la satisfacción de los clientes, con la consecuencia de mayores ingresos y beneficios para las empresas prestadoras del servicio.

Bibliografía

Akar, N., & Gunalay, Y. (2013). Dimensioning shared-per-node recirculating fiber delay line buffers in an optical packet switch. *Performance Evaluation*, *70*(12), 1059-1071.

Al-Shaikhli, A., & Esmailpour, A. (2015). Quality of Service management in 5G broadband converged networks. *2015 36th IEEE Sarnoff Symposium*, 56–61. https://doi.org/10.1109/SARNOF.2015.7324643

Alqaydi, F., Salah, K., & Zemerly, J. (2015). *Queuing Theory Algorithm to find the minimal number of VMs to satisfy SLO response time*. https://doi.org/10.13140/RG.2.1.4476.7847

Arcese M.,Pichetti L., (2015), U.S, US9,009,717 B2, Managing scheduling of processes, New York, Mark and patents office.

Baccelli, F., Kauffmann, B., & Veitch, D. (2009). Inverse Problems in Queueing Theory and Internet Probing. *Queueing Syst. Theory Appl.*, *63*(1–4), 59–107. https://doi.org/10.1007/s11134-009-9150-9

Backer, A. M. T. R. (2017). *QUEUE AND RESERVATION MANAGEMENT SYSTEM*. Retrieved from http://

Bao, L (2014), U.S, US 8,886,899 B1, Managing memory requests based on priority, Westford, MA, Mark and patents office.

Barcelo, F., & Ramon, A. (1997). PRMA performance evaluation based on queueing theory. *1997 IEEE International Conference on Personal Wireless Communications (Cat. No.97TH8338)*, 455–459. https://doi.org/10.1109/ICPWC.1997.655561

Blake, S. (2008). *TRAFFIC MANAGER AND METHOD FOR PERFORMING ACTIVE QUEUE MANAGEMENT OF DISCARD-ELIGIBLE TRAFFIC*. Retrieved from http://

Blesa, M., Calzada, D., Fernández, A., López, L., Martínez, A. L., Santos, A., … Thraves, C. (2009). Adversarial Queueing Model for Continuous Network Dynamics. *Theory of Computing Systems*, *44*(3), 304–331. https://doi.org/10.1007/s00224-007-9046-1

Bonifaci, V. (2007). An Adversarial Queueing Model for Online Server Routing. *Theor. Comput. Sci.*, *381*(1–3), 280–287. https://doi.org/10.1016/j.tcs.2007.05.034

Burmeister, E. F., Blumenthal, D. J., & Bowers, J. E. (2008). A comparison of optical buffering technologies. *Optical Switching and Networking*, *5*(1), 10-18.

Buschi Giovanni;Coulomb, B. D. C. (2006). *METHOD AND SYSTEM FOR INTELLIGENT QUEUE MANAGEMENT*. Retrieved from http://

Caixia Li, Anavatti, S. G., & Ray, T. (2013). A game theory based traffic assignment using queueing networks. *2013 13th International Conference on ITS Telecommunications (ITST)*, 210–215. https://doi.org/10.1109/ITST.2013.6685547

Callegati, F, Campi, A., & Cerroni, W. (2007). A Cost-Effective Approach to Optical Packet/burst Scheduling. *2007 IEEE International Conference on Communications*, 2263–2268. https://doi.org/10.1109/ICC.2007.380

Callegati, F, Cerroni, W., Bonani, L. H., Barbosa, F. R., Moschim, E., & Pavani, G. S. (2006). OPN06-6:

Congestion Resolution in Optical Burst/Packet Switching with Limited Wavelength Conversion. *IEEE Globecom 2006*, 1–5. https://doi.org/10.1109/GLOCOM.2006.393

Callegati, Franco, Cerroni, W., & Pavani, G. S. (2007). Key parameters for contention resolution in multi-fiber Optical Burst/Packet Switching nodes. *Fourth International Conference on Broadband Communications, Networks and Systems, {(BROADNETS} 2007), 10-14 September 2007, Raleigh, North-Carolina, {USA}*, 217–223. https://doi.org/10.1109/BROADNETS.2007.4550428

Chase, R. B., & Aquilano, N. J. (1973). *Production and Operations Management*. Retrieved from https://

Chen, Y., Qiao, C., & Yu, X. (2004). *Optical burst switching: a new area in optical networking research. IEEE network*, *18*(3), 16-23.

Cheng-Jia, L. (2010). *Networked queuing system and method for distributed collaborative clusters of services*. Retrieved from http://

Chia, M. C., Kalavally, V., Chin, S. H., & Franzen, A. (2005, February). *Buffering and scheduling of asynchronous variable-length packets.* In *Network Architectures, Management, and Applications II* (Vol. 5626, pp. 448-456). International Society for Optics and Photonics.

Cisco Press Release: *The Zettabyte Era: Trends and Analysis* (2017). https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networkingindex-vni/vni-hyperconnectivity-wp.html

Corynen, G. C. (2004). *Computer method and user interface for decision analysis and for global system optimization*. Retrieved from http://

Davies Neil James; Holyer, J. Y. L. L. A. T. P. W. V. C. J. (2004). *Allocating priority levels in a data flow*. Retrieved from http://

Develder, C., Van Houdt, B., Blondia, C., Pickavet, M., & Demeester, P. (2004). *Analytical MMAP-based bounds for packet loss in optical packet switching with recirculating fdl buffers. Photonic Network Communications*, *8*(2), 149-161.

Dutta, M. K., & Chaubey, V. K. (2011, December). Modeling and performance analysis of optical packet switching network using fiber delay lines. In *2011 Annual IEEE India Conference* (pp. 1-4). IEEE.

El-Bawab, T. S., & Shin, J. D. (2002). *Optical packet switching in core networks: between vision and reality. IEEE Communications Magazine*, *40*(9), 60-65.

Fu, S., Shum, P., Ngo, N. Q., Wu, C., Li, Y., & Chan, C. C. (2008). An enhanced SOA-based double-loop optical buffer for storage of variable-length packet. *Journal of Lightwave Technology*, *26*(4), 425-431.

Fuente, D., & Pardo, M. (2009). Development of Queueing Models with Balking and Uncertain Data using Fuzzy Set Theory. *IEEM 2009 - IEEE International Conference on Industrial Engineering and Engineering Management*. https://doi.org/10.1109/IEEM.2009.5373330

Galbraith Simon David; Davidson, N. G. (2005). *Assessing user frustration with a load-tested system*. Retrieved from http://

Gao, C., Chen, X.-M., Zheng, F., & Zhu, G. (2014). Asymptotic stability of the M/G/1 queueing system with optional second service. *Applied Mathematical Modelling*, *38*, 4705–4716. https://doi.org/10.1016/j.apm.2014.03.032

Geldenhuys, R., Wang, Z., Chi, N., Tafur Monroy, I., Koonen, A. M. J., Dorren, H. J. S., ... & Yu, S. (2006). Time-slot interchanging using the crosspoint switch and a recirculating buffer. *Microwave and Optical Technology Letters*, *48*(5), 897-900.

Gupta, M. K., & Hemachandra, N. (2015). On a conservation law and the achievable region for waiting time tail probabilities in 2-class M/G/1 queueing systems. *2015 13th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*, 131–138. https://doi.org/10.1109/WIOPT.2015.7151064

Harchol-Balter, M. (2012). Performance Modeling and Design of Computer Systems. In *Performance Modeling and Design of Computer Systems*. https://doi.org/10.1017/cbo9781139226424

Hassan, Y. H. E., & Abdalla, G. M. T. (2015). Video conference traffic control for SudREN using QoS. *2015 International Conference on Computing, Control, Networking, Electronics and Embedded Systems Engineering (ICCNEEE)*, 303–307. https://doi.org/10.1109/ICCNEEE.2015.7381382

Holliday, S. (2010). *AUTOMATIC SELF-OPTIMIZING QUEUE MANAGEMENT SYSTEM*. Retrieved from http://

Jensen, R. A. (2006). *CALL MANAGEMENT SYSTEM USING DYNAMIC QUEUE POSITION*. Retrieved from http://

Jiafu He, & Sohraby, K. (2003). An extended combinatorial analysis framework for discrete-time queueing systems with general sources. *IEEE/ACM Transactions on Networking*, *11*(1), 95–110. https://doi.org/10.1109/TNET.2002.808404

Jiang, F., Hsu, C., & Wang, S. (2017). Logistic Support Architecture with Petri Net Design in Cloud Environment for Services and Profit Optimization. *IEEE Transactions on Services Computing*, *10*(6), 879–888. https://doi.org/10.1109/TSC.2016.2514506

Kakubava, R., Prangishvili, A., & Sokhadze, G. (2016). *Closed and Mixed Type Queuing Systems as Mathematical Models of Reliability and Survivability*. 23–28. https://doi.org/10.1109/SMRLO.2016.15

Lambert, J., Van Houdt, B., & Blondia, C. (2005). Single-wavelength optical buffers: non-equidistant structures and preventive drop mechanisms. *Proceedings of NAEC*, 545-555.

Langenhorst, R., Eiselt, M., Pieper, W., Grosskopf, G., Ludwig, R., Kuller, L., ... & Weber, H. G. (1996). Fiber loop optical buffer. *Journal of Lightwave Technology*, *14*(3), 324-335.

Larocque, G. R., & Lipoff, S. J. (1996). Application of discrete event simulation to network protocol modeling. *Proceedings of ICUPC - 5th International Conference on Universal Personal Communications*, *2*, 508–512 vol.2. https://doi.org/10.1109/ICUPC.1996.562625

Le Thai Franck;Nahum, E. M. (2017). *SYSTEM, METHOD, AND RECORDING MEDIUM FOR QUEUE MANAGEMENT IN A FORWARDER*. Retrieved from http://

Lin, X., Kwok, Y., & Wang, H. (2007). On Improving the Energy Efficiency of Wireless Sensor Networks under Time-Varying Environment. *32nd IEEE Conference on Local Computer Networks (LCN 2007)*, 520–530. https://doi.org/10.1109/LCN.2007.87

Liu, A. M., Wu, C. Q., Lim, M. S., Gong, Y. D., & Shum, P. (2004). Optical buffer configuration based on a 3x3 collinear fibre coupler. *Electronics Letters*, *40*(16), 1017-1019.

Mayekar, P., Venkateswaran, J., Gupta, M. K., & Hemachandra, N. (2015). Performance analysis and decomposition results for some dynamic priority schemes in 2-class queues. *2015 13th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*, 115–122. https://doi.org/10.1109/WIOPT.2015.7151062

McKenney, P. E. (2015). *Low Overhead Contention-Based Switching Between Ticket Lock And Queued Lock*. Retrieved from http://

Meiri Dror; Shazar, J. (2016). *SYSTEM AND METHOD FOR QUEUE MANAGEMENT*. Retrieved from http://

Merchant, K. K., McGeehan, J. E., Willner, A. E., Ovadia, S., Kamath, P., Touch, J. D., & Bannister, J. A. (2005). Analysis of an optical burst switching router with tunable multiwavelength recirculating buffers. *Journal of lightwave technology*, *23*(10), 3302.

Mewara, H., & Manghnani, D. (2015). Comparative analysis of VOIP application with different queuing schemes in WiMAX using OPNET. *Proceedings - 2015 IEEE International Conference on Computational Intelligence and Communication Technology, CICT 2015*, 475–481. https://doi.org/10.1109/CICT.2015.169

Minkevičius, S., Dolgopolovas, V., & Sakalauskas, L. L. (2016). A Law of the Iterated Logarithm for the Sojourn Time Process in Queues in Series. *Methodology and Computing in Applied Probability*, *18*(1), 37–57. https://doi.org/10.1007/s11009-014-9402-y

Monk, P. (2015). *SYSTEMS AND METHODS FOR QUEUE MANAGEMENT*. Retrieved from http://

Mooney, C. Z. (1997). *Monte Carlo simulation*. Retrieved from https://books.google.com.co/books/about/Monte_Carlo_Simulation.html?id=xQRgh4z_5acC&redir_esc=y

Mowcheng Lee. (1995). Discrete-time limited 1-customer G/G/1 vacation queueing systems and asymmetric polling systems. *Proceedings of GLOBECOM 1995 Mini*, 92–96. https://doi.org/10.1109/CTMC.1995.502938

Muñoz, E., & Ruspini, E. H. (2014). Simulation of Fuzzy Queueing Systems With a Variable Number of Servers, Arrival Rate, and Service Rate. *IEEE Transactions on Fuzzy Systems*, *22*(4), 892–903. https://doi.org/10.1109/TFUZZ.2013.2278407

Mukherjee, B. (2007, August). Architecture, control, and management of optical switching networks. In *2007 Photonics in Switching* (pp. 43-44). IEEE.

Nagy, L., Tombal, J., & Novotny, V. (2013). *Proposal of a queueing model for simulation of advanced telecommunication services over IMS architecture*. 326–330. https://doi.org/10.1109/TSP.2013.6613945

Nippon Telegraph and Telephone Corporation: *One Petabit per Second Fiber Transmission Over a Record Distance of 200 km* (2017). https://www.ntt.co.jp/ news2017/1703e/pdf/170323a.pdf

Pandit, K., Schmittt, J., & Steinmetz, R. (2004). Network calculus meets queueing theory -a simulation based approach to bounded queues. *Twelfth IEEE International Workshop on Quality of Service, 2004. IWQOS 2004.*, 114–120. https://doi.org/10.1109/IWQOS.2004.1309365

Pardo, M. J., & de la Fuente, D. (2007). Optimizing a priority-discipline queueing model using fuzzy set theory. *Computers & Mathematics with Applications*, *54*(2), 267–281. https://doi.org/https://doi.org/10.1016/j.camwa.2007.01.019

Pazgal, A. I., & Radas, S. (2008). Comparison of customer balking and reneging behavior to queueing theory predictions: An experimental study. *Computers & Operations Research*, *35*(8), 2537–2548. https://doi.org/https://doi.org/10.1016/j.cor.2006.12.027

Popkey Robert; Rzeznik, M. (2017). *QUEUE MANAGEMENT SYSTEM AND METHOD*. Retrieved from http://

Rabta, B., Schodl, R., Reiner, G., & Fichtinger, J. (2013). A hybrid analysis method for multi-class queueing networks with multi-server nodes. *Decision Support Systems*, *54*, 1541–1547. https://doi.org/10.1016/j.dss.2012.05.056

Raina, G., & Wischik, D. (2005). Buffer sizes for large multiplexers: TCP queueing theory and instability analysis. *Next Generation Internet Networks, 2005*, 173–180. https://doi.org/10.1109/NGI.2005.1431663

Ravner, L. (2014). Equilibrium arrival times to a queue with order penalties. *European Journal of Operational Research*, *239*(2), 456–468. https://doi.org/https://doi.org/10.1016/j.ejor.2014.06.005

Rodríguez, J., Lillo, R. E., & Ramírez-Cobo, P. (2016). Nonidentifiability of the Two-State BMAP. *Methodology and Computing in Applied Probability*, *18*(1), 81–106. https://doi.org/10.1007/s11009-014-9401-z

Rogiest, W, & Bruneel, H. (2010). Exact Optimization Method for an FDL Buffer With Variable Packet Length. *IEEE Photonics Technology Letters*, *22*(4), 242–244. https://doi.org/10.1109/LPT.2009.2038237

Rogiest, Wouter, Dorsman, J.-P., & Fiems, D. (2014). Analysis of Fibre-loop Optical Buffers with a Void-avoiding Schedule. *Proceedings of the 8th International Conference on Performance Evaluation Methodologies and Tools*, 122–128. https://doi.org/10.4108/icst.valuetools.2014.258180

Rossiter Margaret, R. (2003). *State Based Management for Queue-Server Systems*. Retrieved from http://

Rostami, A., & Chakraborty, S. S. (2005). On performance of optical buffers with specific number of circulations. *IEEE Photonics Technology Letters*, *17*(7), 1570-1572.

Rubin, I., & Ouaily, M. (1989). Performance analysis of message delay limited synchronous

communication and queueing systems. *IEEE INFOCOM '89, Proceedings of the Eighth Annual Joint Conference of the IEEE Computer and Communications Societies*, *1*, 51–58 vol.1. https://doi.org/10.1109/INFCOM.1989.101433

Sahlin, M. (2003). *CUSTOMER QUEUE MANAGEMENT METHOD AND DEVICE THEREFORE*. Retrieved from http://

Salas Fehlandt, E. M. (2014). *SISTEMA DE INFORMACIÓN EN LÍNEA PARA TELÉFONOS INTELIGENTES, SMART PHONES, PC U OTRO DISPOSITIVO ELECTRÓNICO, DEL CONTEO DEL NÚMERO DE TURNO EN LAS FILAS, MEJORANDO LA ESPERA DE LOS USUARIOS*. Retrieved from http://

Sanchis Joseph; Muniz, J. C. (2016). *VIRTUAL QUEUE MANAGEMENT SYSTEM*. Retrieved from http://

Shah, S. A. A., Shah, W., Rind, U. A., & Das Menghwar, G. (2009). Analysis of networks buffer using discrete-time queueing models. *2009 IEEE Student Conference on Research and Development (SCOReD)*, 105–108. https://doi.org/10.1109/SCORED.2009.5443269

Shah, W., Shah, S. A. A., Rind, U. A., & Das Menghwar, G. (2009). Discrete-time queueing analysis of communication buffer with multiserver. *2009 IEEE Student Conference on Research and Development (SCOReD)*, 9–11. https://doi.org/10.1109/SCORED.2009.5443427

Sussman, A. B. R. D. D. (2017). *SYSTEM AND METHOD FOR DYNAMIC QUEUE MANAGEMENT USING QUEUE PROTOCOLS*. Retrieved from http://

Szczesniak, I. (2009). Overview of optical packet switching. *Theoretical and applied informatics*, *21*(3-4), 167-180.

Tahir, Y., Yang, S., Adeel, U., & McCann, J. (2015). Symbiot: Congestion-Driven Multi-resource Fairness for Multi-user Sensor Networks. *2015 IEEE 17th International Conference on High Performance Computing and Communications, 2015 IEEE 7th International Symposium on Cyberspace Safety and Security, and 2015 IEEE 12th International Conference on Embedded Software and Systems*, 654–659. https://doi.org/10.1109/HPCC-CSS-ICESS.2015.23

Tanemura, T., Soganci, I. M., Oyama, T., Ohyama, T., Mino, S., Williams, K. A., ... & Nakano, Y. (2010). Large-capacity compact optical buffer based on InP integrated phased-array switch and coiled fiber delay lines. *Journal of Lightwave Technology*, *29*(4), 396-402.

Tatxé, A. F. (2010). *Simulation of an Optical Packet Switch Scheduler*. Dipartimento di Elettronica, Informatica e Sistemistica, Univesitá di Bologna, Facoltá di Ingegneria.

Technical University of Munich Press Release, *New record data transfer speed in fiber optic network* (February 2019). URL https://www.technologist.eu/new-record-data-transfer-\ speed-in-fiber-optic-network/

Tian, C. Y., Wu, C. Q., Sun, G. N., Li, X., & Li, Z. Y. (2008). Quality improvement of the dual-wavelength signals in DLOB via power equalization. *Optoelectronics Letters*, *4*(5), 361-364.

Tiwari, L. M., Agrawal, S., Kapoor, S., & Chauhan, A. (2011). Entropy as a measure of uncertainty in queueing system. *2011 National Postgraduate Conference*, 1–4. https://doi.org/10.1109/NatPC.2011.6136265

Tong, Z., Lu, H., Haenggi, M., & Poellabauer, C. (2016). A Stochastic Geometry Approach to the Modeling of DSRC for Vehicular Safety Communication. *IEEE Transactions on Intelligent Transportation Systems*, *17*(5), 1448–1458. https://doi.org/10.1109/TITS.2015.2507939

Triki, A., Gravey, A., Gravey, P., & Morvan, M. (2017, May). Long-term CAPEX evolution for slotted optical packet switching in a metropolitan network. In *2017 International Conference on Optical Network Design and Modeling (ONDM)* (pp. 1-6). IEEE.

Tsitsiashvili, G. S., & Osipova, M. A. (2008). Limiting distributions in queueing networks with unreliable elements. *Problems of Information Transmission*, *44*(4), 385–394. https://doi.org/10.1134/S0032946008040091

Urquia M., A., & Martin V., C. (2013). *MODELADO Y SIMULACIÓN DE EVENTOS DISCRETOS*. Retrieved from https://books.google.com.co/books?id=BZBGAgAAQBAJ

Van Hautegem, K., Pinto, M., Bruneel, H., & Rogiest, W. (2018). Analysis of VAS, WAS and XAS scheduling algorithms for fiber-loop optical buffers. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. https://doi.org/10.1007/978-3-319-93736-6_16

Van Hautegem, Kurt, Pinto, M., Gomez, D., Bruneel, H., & Rogiest, W. (2020). T-WAS and T-XAS algorithms for Fiber-loop Optical Buffers. *Optical Switching and Networking*, *35*.

Van Hautegem, Kurt, Rogiest, W., & Bruneel, H. (2014a). Scheduling in optical switching: Deploying shared wavelength converters more effectively. *2014 IEEE International Conference on Communications, ICC 2014*, 3418–3424. https://doi.org/10.1109/ICC.2014.6883850

Van Hautegem, Kurt, Rogiest, W., & Bruneel, H. (2014b). Void-creating algorithm in OPS/OBS: mind the gap. In T. E. Simos & C. Tsitouras (Eds.), *AIP Conference Proceedings* (Vol. 1648, p. 4). The American Institute of Physics.

Van Hautegem, Kurt, Rogiest, W., & Bruneel, H. (2015a). Improving performance and energy consumption of shared wavelength converters in OPS/OBS. *Optical Switching and Networking*. https://doi.org/10.1016/j.osn.2014.10.003

Van Hautegem, Kurt, Rogiest, W., & Bruneel, H. (2015b). Improving the Energy Efficiency of Scheduling Algorithms for OPS/OBS Buffers. *Photonic Netw. Commun.*, *29*(2), 183–197. https://doi.org/10.1007/s11107-014-0481-z

Van Hautegem, K., Rogiest, W., & Bruneel, H. (2016, December). Optical switching for variable size packets: improved void filling through selective void creation. In *Proceedings of the 11th International Conference on Queueing Theory and Network Applications* (pp. 1-8).

Van Heddeghem, W., Lannoo, B., Colle, D., Pickavet, M., Musumeci, F., Pattavina, A., & Idzikowski, F. (2013, October). Power consumption evaluation of circuit-switched versus packet-switched optical backbone networks. In *2013 IEEE Online Conference on Green Communications (OnlineGreenComm)* (pp. 56-63). IEEE.

Verma, S., Chaskar, H., & Ravikanth, R. (2000). Optical burst switching: a viable solution for terabit IP backbone. *IEEE network*, *14*(6), 48-53.

Wang, J., McArdle, C., & Barry, L. P. (2014, July). Energy-efficient optical packet switch with recirculating fiber delay line buffers for data center interconnects. In *2014 16th International Conference on Transparent Optical Networks (ICTON)* (pp. 1-4). IEEE.

Wang, W., Dong, L., & Wolf, W. (2002). A Distributed Switch Architecture with Dynamic Load-balancing and Parallel Input-Queued Crossbars for Terabit Switch Fabrics. *Proceedings - IEEE INFOCOM*, *1*, 352–361. https://doi.org/10.1109/INFCOM.2002.1019277

Wang, X. (2016). *FACILITATING DYNAMIC HIERARCHICAL MANAGEMENT OF QUEUE RESOURCES IN AN ON-DEMAND SERVICES ENVIRONMENT*. Retrieved from http://

Wang, Y., Guo, J., Ceder, A., Currie, G., Dong, W., & Yuan, H. (2014). Waiting for public transport services: Queueing analysis with balking and reneging behaviors of impatient passengers. *Transportation Research Part B: Methodological*, *63*, 53–76. https://doi.org/10.1016/j.trb.2014.02.004

Wang, Y., Wu, C., Wang, Z., & Xin, X. (2009, March). A new large variable delay optical buffer based on cascaded double loop optical buffers (DLOBs). In *Optical Fiber Communication Conference* (p. OWA4). Optical Society of America.

Weber Kurt;Cantonnet, F. L. (2015). *AUTOMATED WAITING ROOM QUEUE AND MANAGEMENT SERVICE*. Retrieved from http://

Wen, Y., Jin, B., Li, K., Cheng, A. M. K., & Fang, W. (2012). A Queueing Theory Based Approach to QoS-Driven Adaptation for Service Discovery over MANETs. *2012 IEEE 15th International Conference on Computational Science and Engineering*, 594–601. https://doi.org/10.1109/ICCSE.2012.87

Wigren, T. (2016). Robust$\mathcal {L}_{2}$Stable Networked Control of Wireless Packet Queues in Delayed Internet Connections. *IEEE Transactions on Control Systems Technology*, *24*(2), 502–513. https://doi.org/10.1109/TCST.2015.2455933

Williams, M. (2010). *UN SISTEMA DE MANEJO DE COLA ACCESORIO Y METODO PARA INTERACTUAR CON EL SISTEMA DE FORMACION DE COLA*. Retrieved from http://

Xia, L., & Cao, X.-R. (2012). Performance optimization of queueing systems with perturbation realization. *European Journal of Operational Research*, *218*(2), 293–304. https://doi.org/https://doi.org/10.1016/j.ejor.2011.07.039

Xiong, Y., Vandenhoute, M., & Cankaya, H. C. (2000). Control architecture in optical burst-switched WDM networks. *IEEE journal on selected areas in communications*, *18*(10), 1838-1851.

Yao, S., Yoo, S. B., & Dixit, S. (2003). A unified study of contention-resolution schemes in optical packet-switched networks. *Journal of lightwave technology*, *21*(3), 672.

Yoo, M., Qiao, C., & Dixit, S. (2000, June). The effect of limited fiber delay lines on QoS performance of optical burst switched WDM networks. In *2000 IEEE International Conference on Communications. ICC 2000. Global Convergence Through Communications. Conference Record* (Vol. 2, pp. 974-979). IEEE.

Anexos

Analysis of VAS, WAS and XAS Scheduling Algorithms for Fiber-Loop Optical Buffers

Kurt Van Hautegem[1(B)], Mario Pinto[2], Herwig Bruneel[1], and Wouter Rogiest[1]

[1] SMACS Research Group, Department of Telecommunication and Information Processing (TELIN), Ghent University, Ghent, Belgium
{kurt.vanhautegem,herwig.bruneel,wouter.rogiest}@ugent.be

[2] Universidad Antonio Nariño, Bogotá, Colombia maupinto@uan.edu.co

Abstract. In optical packet/burst switched networks fiber loops provide a viable and compact means of contention resolution. For fixed size packets it is known that a basic void-avoiding schedule (VAS) can vastly outperform a more classical pre-reservation algorithm as FCFS. In this contribution we propose two novel forward-looking algorithms, WAS and XAS, that outperform VAS in the setting of a uniform distributed packet size and a restricted buffer size. This paper presents results obtained by Monte Carlo simulation, showing that improvements of more than 20% in packet loss in specific settings are obtainable. In other settings and for other performance measures similar improvements are within reach.

Keywords: Contention resolution · Scheduling · Optical buffering Fiber loops

Introduction

As video on demand (VoD) services increase in popularity and 4K video quality will become the new normal, global IP traffic is expected to grow at a compounding annual rate of 24% between 2016 and 2021 [1]. As wavelength and spatial multiplexing allows optical fiber technology to reach dazzling bandwidths of up to 1 Petabit/s [2], it seems that our unlimited demand for bandwidth can be met without a problem. Unfortunately, in existing optical networks capacity is not limited by the connections but in the nodes where slow electronic switching or inflexible optical circuit switching muffle the optical highway capacity.

Promising solutions to address the issues in optical backbones are optical burst switching (OBS) [3–5] and optical packet switching (OPS) [6–8]. In these packet based switching techniques, optical signals are, similar to optical circuit switching (OCS) [9,10], kept in the optical domain to avoid slow optical-electronic-optical (OEO) conversions but, similar to electronic switching, processed as packets to increase statistical multiplexing efficiency. Although RAM

buffering in the nodes is infeasible because it requires OEO conversions, at least a limited amount of buffering remains advisable to address the unavoidable contention that arises in the nodes [11,12].

One of the most compact implementations of optical buffering today is a fiber loop buffer. As opposed to feed-forward buffers where every line is traversed only once [13], fiber loop buffers allow contending packets to recirculate multiple times within the same coiled fiber loop [14,15]. Although alternative designs as dual-loop optical buffers exist [16–19], most use a set of single fiber loops in parallel which can all accommodate a single packet at once (shown in Fig.1). Because packets can only exit a loop after a round number of recirculations, fiber loops can only provide a discrete set of delays. As opposed to electronic memory (RAM) packets can thus not be retrieved at will, resulting in small time gaps or voids in between packets on the outgoing line. Moreover, as packets recirculate in the same loop, fiber loops can only accommodate packet sizes smaller than or equal to their loop length and packet length directly limits the resolution of possible delays. Since the footprint is preferably kept small, with a small number of fiber loops, and also the number of recirculations a packet can make in a loop is kept low to prevent signal degeneration, scheduling algorithms in which the resources are used as efficiently as possible are needed to achieve low packet loss and packet delay.

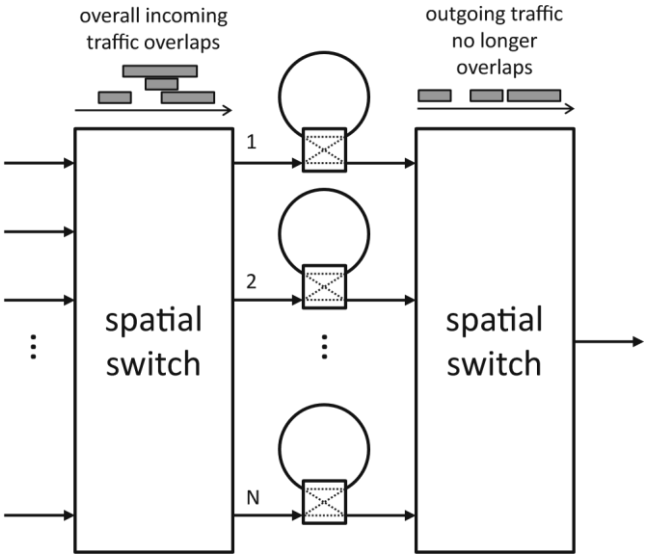

Fig.1. Parallel optical fiber loop buffer to resolve contention at the input.

In [20] an analytical model is used to evaluate performance of the void avoiding schedule (VAS) for fixed size packets equal to the loop length. The void-avoiding schedule is a post-reservation scheme [21], allowing the packets to enter the buffer freely, only deciding later when a packet has to exit its loop. In [20] it is shown that performance of the VAS is significantly better than that of algorithms with a pre-

reservation scheme, e.g. FCFS, in which the number of recirculations is decided upon arrival of a packet.
In this paper we evaluate the performance of the void-avoiding schedule (VAS) for uniform distributed packet size in both unlimited and constricted fiber loop buffer settings. We also propose two new algorithms, WAS and XAS, which, to the best of our knowledge, are the first known algorithms to outperform VAS. Particularly, both yield significant improvement for variable-length packet size, as discussed further below. The structure of this paper is organized as follows: System model and assumptions are discussed in Sect.2, the scheduling algorithms in Sect.3, performance measures and methodology in Sect.4, numerical results in Sect.5 and finally conclusions and future work in Sect.6.

System Model and Assumptions

Throughout the paper the same continuous-time setting as in [22–24] is supposed. The fiber loop buffer is assumed to be located at and dedicated to a single outgoing port of an optical switch. Wavelength convertors, if present within the switch, are assumed to perform conversion to a single outgoing wavelength associated with this single outgoing port. The analysis can thus be limited to a single wavelength. We assume the joint packet arrival at the output port on this single wavelength is a Poisson process, i.e. the inter-arrival times T are exponentially distributed with an average of $E[T]$. The length of arriving packets, B, is assumed to be uniform distributed on the interval $[0,S]$ with an average of $E[B] = S/2$. Related, the overall incoming traffic load at the output port is given by $\rho = E[B]/E[T] = S/(2 \cdot E[T])$.

Because of the nature of the arrival process it is possible that different arrivals overlap upon their arrival at which instant one of the contending packets has to be temporarily buffered in one of the fiber loops. We assume a set of parallel fiber loops of length S, that indepently of the packet's size, can accommodate a single packet. The assignable delays to a packet are thus multiples of S. The number of fiber loops and the maximum number of times a packet can recirculate are both varied in simulations. Combinations of both finite (4, 8, and 16) and infinite values for these buffer parameters are evaluated.

Scheduling Algorithms

As in a buffer loop setting the well-known FCFS algorithm is outperformed by the void avoiding schedule (VAS), we choose the latter as our benchmark algorithm. In the VAS packets that arrive are transmitted immediately if the outgoing line is available or stored in a fiber loop if not. After each loop recirculation, the availability of the outgoing line is checked. If the outgoing line is available upon such a check, the packet exits its fiber loop and is sent. If the outgoing line is not available, the packet is recirculated again and the procedure is repeated. The VAS does not preserve the arrival order and is a post-reservation algorithm.

In terms of resource usage, VAS is a greedy algorithm, sending a packet whenever a transmission opportunity, i.e. an available outgoing line and either an arrival or a finished recirculation, arises. The newly proposed WAS algorithm, where "W" may refer to the aim of minimizing the **W**ait for the outgoing line to turn available after departure of the two packets, is more considerate and well aware of the other packets present in the system. When a transmission opportunity arises, WAS will first calculate the time needed to send each combination of two packets (packets i and j, $i = j$) present in the system. When $N + 1$ packets are present in the system (i.e. packets which are either in the buffer or a new arrival), this gives an $N \times N$ two dimensional matrix with the time needed to send each combination. In this matrix rows are assumed to be the first packet sent and columns the second (rows before columns). Only when the packet is the first packet (i.e. the row, not the column) of the lowest combination in this matrix, the WAS algorithm will send this packet. Otherwise, depending on whether the transmission opportunity is a new arrival or a finished recirculation, this packet is buffered in a new loop or given another round in its loop. Note that in this situation the lowest combination will not necessarily be the next two packets to be sent as the matrix is re-evaluated at every transmission opportunity. Similarly when the packet that triggered the transmission opportunity is actually sent, the packet that was also part of the lowest combination is not guaranteed to be transmitted next.

Similar to the WAS algorithm, the XAS algorithm, where X may refer to the aim of e**X**tending the period during which the outgoing line is effectively used by the two packets, also calculates a combination matrix to decide upon transmission when a transmission opportunity arises. In this matrix the efficiency of the outgoing line is calculated for each combination by dividing the sum of both packet lengths by the total time needed to send each combination. Only when the packet that triggered the transmission opportunity is the first packet of the combination with the highest efficiency, the XAS algorithm will actually send this packet.

When either the number of fiber loops or the maximum number of recirculations is constricted, the WAS and XAS algorithms will transmit a packet upon a transmission opportunity if doing otherwise would result in an immediate and unnecessary loss. Suppose for example that upon arrival of a new packet the output line is available but all of the fiber loops (finite set) are occupied by other packets. In such a case both WAS and XAS will transmit the new arrival even though a more favorable combination may be present in the fiber loops. By doing so the new arrival need not be dropped and unnecessary loss is prevented. Likewise when the maximum number of recirculations is reached upon the end of a loop recirculation, WAS and XAS will always transmit the packet if the output line is available. In the terminology of [25] we could say that both WAS and XAS are tuned to avoid the use of preventive drop. A flowchart showing the decision process of WAS and XAS when both the number of fiber loops and the maximum number of recirculations is constricted is shown in Fig.2. When either the number of fiber loops and/or the maximum number of recirculations is not constricted, the flowchart slightly simplifies.

Note that in the case of fixed size packets both WAS and XAS schedule in exactly the same way as VAS. Indeed, as all packets have the same size, the combination that minimizes the time to transmit a pair of packets will always consist of the packet causing the transmission opportunity. Only when packet sizes are not equal to a fixed size, WAS and XAS schedule different, and thus possibly better, than VAS.

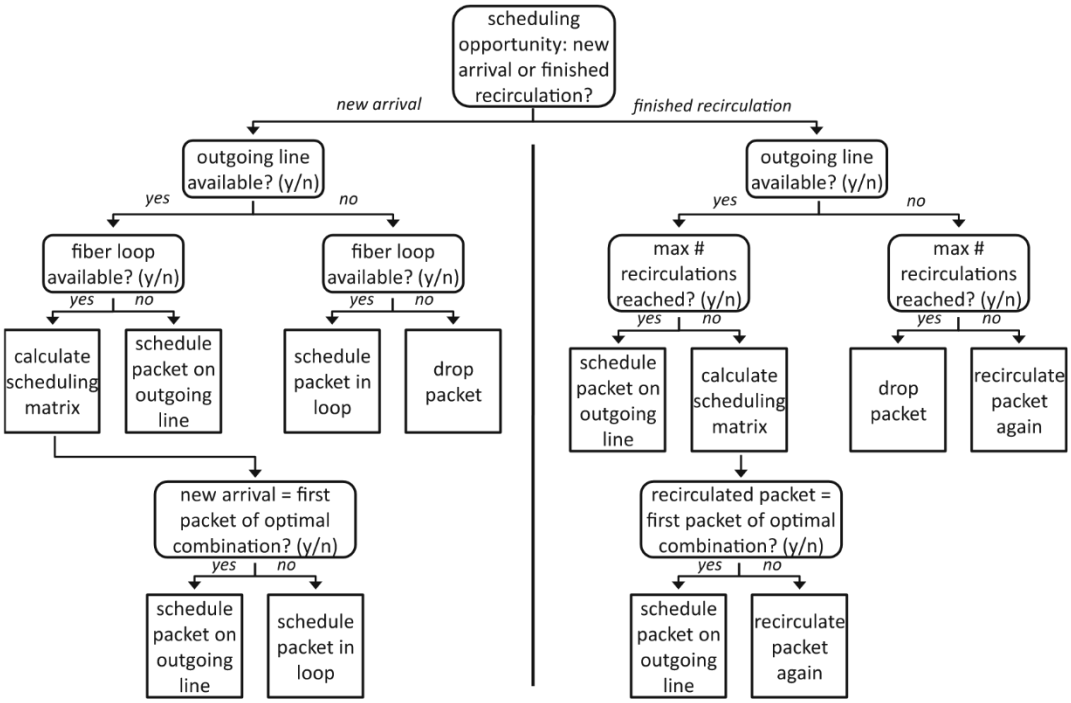

Fig.2. Flowchart for WAS and XAS both the number of fiber loops and the maximum number of recirculations is constricted.

Performance Measures and Methodology

To obtain a complete and representative image of the performance of the various algorithms we look at different performance measures for different settings. For the case with an unlimited number of fiber loops and no restriction on the number of recirculations, no packets are lost and we compare the algorithms on the average packet delay. In case either, or both, the number of fiber loops or the maximum number of recirculations is limited, the loss probability (LP), or equivalently, packet loss, is our main performance measure. In addition, we study a related performance measure which we refer to as LPsize, accounting for the relative total size of packets lost to the relative total size of packets, or, equivalently, the relative amount of data lost with respect to the total amount of data that arrived.

As extending the analytical method from [20] to different algorithms, settings and packet size distributions proved to be too challenging, the performance of the algorithms is evaluated by means of Monte Carlo Simulations. Specifically, all

algorithms are programmed in Matlab using a discrete event simulation (DES). In a DES, the system is modelled as a sequence of events marked by their particular instant in time, i.e. the simulation is event-based. The system state changes from one event to the next and does not change in-between events. This is as opposed to continuous simulation in which time is broken into small pieces called time slices. At each ending of a time slice the system state is (possibly) changed based on the events happened in the last time slice. Because DES simulations do not simulate every time slice, they are far more efficient in terms of computational resources.

Numerical Results

Fiber loops are assumed to have a length of one time unit and packets to be uniformly distributed on the interval [0,1]. Load is varied from 0.6 to 0.95 in steps of 0.05 by changing the average inter-arrival time of the Poisson arrival process. The arrival of 10^6 packets is simulated 10 times for each algorithm and parameter combination. In this way adequate average performance measures and accompanying confidence intervals are obtained.

Figure 3 shows the average packet delay for the setting with an unlimited number of fiber loops and no restriction on the maximum number of recirculations. As in this setting, for the load values investigated, the FCFS algorithm results in an unstable regime, it is not included in the graph. From Fig.3 it is clear that in the unrestricted case and with a uniform packet size distribution only XAS can outperform VAS. Table 1 shows the performance improvement (in percentage) XAS can obtain in waiting time relative to VAS. As the load increases, the obtainable improvement also goes up, reaching an improvement of almost 20% for a load of 0.95.

Table 1. Percentage-wise performance improvement in packet delay of XAS relative to VAS for different load values in an unrestricted buffer setting.

Packet delay reduction	Load							
	0.60	0.65	0.70	0.75	0.80	0.85	0.90	0.95
XAS	1.1%	2.4%	4.2%	7.3%	9.6%	13.7%	17.2%	18.1%

Table 2 shows the LP and LPsize of the VAS algorithm in different restricted buffer settings. Note that LP and LPsize have the same values for the VAS algorithm. This is because the length of a packet does not influence the way a packet is scheduled in VAS. In Tables 3 (WAS) and 4 (XAS) the performance improvements (in percentage) compared to VAS of LP and LPsize are shown. This is done for load values of 0.6, 0.7, 0.8 and 0.9, and all combinations of the number of loops and the maximum number of recirculations (both take on values of 4, 8, 16 and infinity). From these tables it is clear that not for all combinations of parameters a performance improvement is possible. In general, but not always, performance improvements increase for lower load values, a lower number of maximum recirculations, and a higher number of loops. Comparing

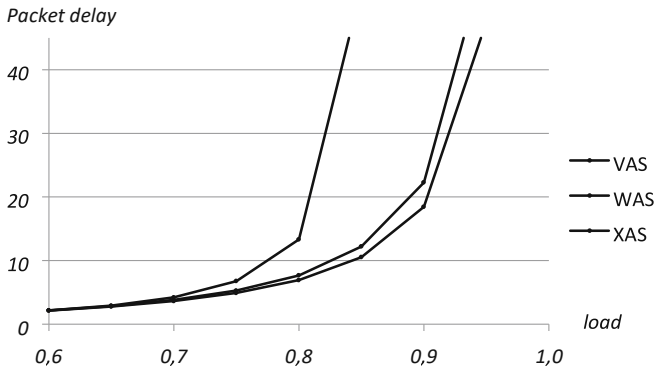

Fig.3. Packet delay for the VAS, WAS and XAS algorithms in an unrestricted buffer setting.

Table 2. LP and LPsize values for VAS for different load values in different restricted buffer settings.

VAS		Load															
		0.60				0.70				0.80				0.90			
		# loops				# loops				# loops				# loops			
		4	8	16	∞	4	8	16	∞	4	8	16	∞	4	8	16	∞
		LP = LPsize (%)															
Max recirculations	4	10.3	8.8	8.8	8.8	15.7	13.3	13.2	13.2	21.2	18.2	18.0	18.0	26.5	23.0	22.6	22.6
	8	7.2	3.6	3.4	3.4	12.6	7.6	6.8	6.8	18.7	12.7	11.2	11.3	24.6	18.4	16.2	16.1
	16	6.6	1.5	0.7	0.7	12.0	4.8	2.5	2.4	18.1	10.3	6.1	5.8	24.1	16.7	11.3	10.5
	∞	6.6	1.2	0.0	0.0	12.0	4.3	0.4	0.0	18.0	9.8	3.1	0.0	24.1	16.4	9.2	0.0

Table 3. Percentage-wise performance improvement in LP and LPsize of WAS relative to VAS for different load values in different restricted buffer settings.

WAS		Load															
		0.60				0.70				0.80				0.90			
		# loops				# loops				# loops				# loops			
		4	8	16	∞	4	8	16	∞	4	8	16	∞	4	8	16	∞
		LP reduction (%)															
Max Recirculations	4	17.3	27.3	27.7	27.4	11.3	21.4	22.0	21.9	7.5	17.7	18.7	18.6	5.2	15.0	16.6	16.5
	8	7.0	23.5	23.4	23.7	4.3	16.6	18.2	18.1	2.5	12.6	15.8	15.9	1.3	9.8	15.6	15.7
	16	4.1	13.7	−0.7	−1.3	1.7	6.4	−1.6	−3.6	0.5	1.6	2.3	−0.3	−0.6	0.3	6.7	5.5
	∞	3.8	21.2	53.6	NaN	1.8	4.8	3.4	NaN	0.2	−4.4	−39.2	NaN	−0.6	−6.9	−32.7	NaN
		LPsize reduction (%)															
Max Recirculations	4	8.8	8.8	8.9	8.4	3.4	1.3	1.2	0.8	0.5	−3.2	−3.7	−3.8	−1.2	−5.6	−6.8	−6.8
	8	3.4	−3.1	−8.5	−8.1	0.8	−10.1	−16.7	−16.6	−0.8	−11.7	−20.7	−20.6	−1.9	−11.9	−21.5	−21.5
	16	3.9	−5.0	−59.8	−61.3	1.2	−10.1	−59.7	−63.4	0.0	−12.2	−51.3	−58.6	−1.2	−11.3	−40.4	−49.5
	∞	3.6	21.5	53.0	NaN	1.8	4.9	3.3	NaN	0.3	4.4	39.0	NaN	0.6	6.8	32.6	NaN

Table 4. Percentage-wise performance improvement in LP and LPsize of XAS relative to VAS for different load values in different restricted buffer settings.

XAS		Load															
		0.60				0.70				0.80				0.90			
		# loops				# loops				# loops				# loops			
		4	8	16	∞	4	8	16	∞	4	8	16	∞	4	8	16	∞
		LP reduction (%)															
Max recirculations	4	9.9	15.9	16.2	16.3	4.6	8.3	8.7	8.7	1.2	3.2	3.6	3.6	−0.6	−0.1	0.1	0.2
	8	−0.9	10.1	12.4	12.1	−1.0	3.4	5.1	5.5	−1.2	−0.6	−0.2	0.2	−1.6	−2.8	−3.6	−3.4
	16	−3.3	−3.9	0.7	0.7	−2.0	−0.7	1.3	0.7	−1.1	0.2	−2.0	−2.5	−0.8	−0.1	−4.3	−6.2
	∞	−3.8	−4.3	−9.3	NaN	−1.8	2.0	7.6	NaN	−1.1	3.3	13.4	NaN	−0.5	3.2	10.5	NaN
		LPsize reduction (%)															
max recirculations	4	17.5	30.1	30.6	30.6	11.4	22.3	23.2	23.0	7.1	16.2	17.3	17.4	4.6	11.8	13.0	13.2
	8	2.9	31.6	37.9	37.6	2.5	22.0	29.7	30.1	2.1	15.0	22.5	23.0	1.3	10.1	16.9	17.4
	16	−3.1	8.3	41.0	40.9	−1.8	8.5	35.5	36.7	−0.7	7.8	26.3	29.4	−0.4	5.9	18.2	21.8
	∞	3.8	4.3	13.9	NaN	1.9	2.0	8.0	NaN	1.1	3.2	13.4	NaN	0.5	3.3	10.5	NaN

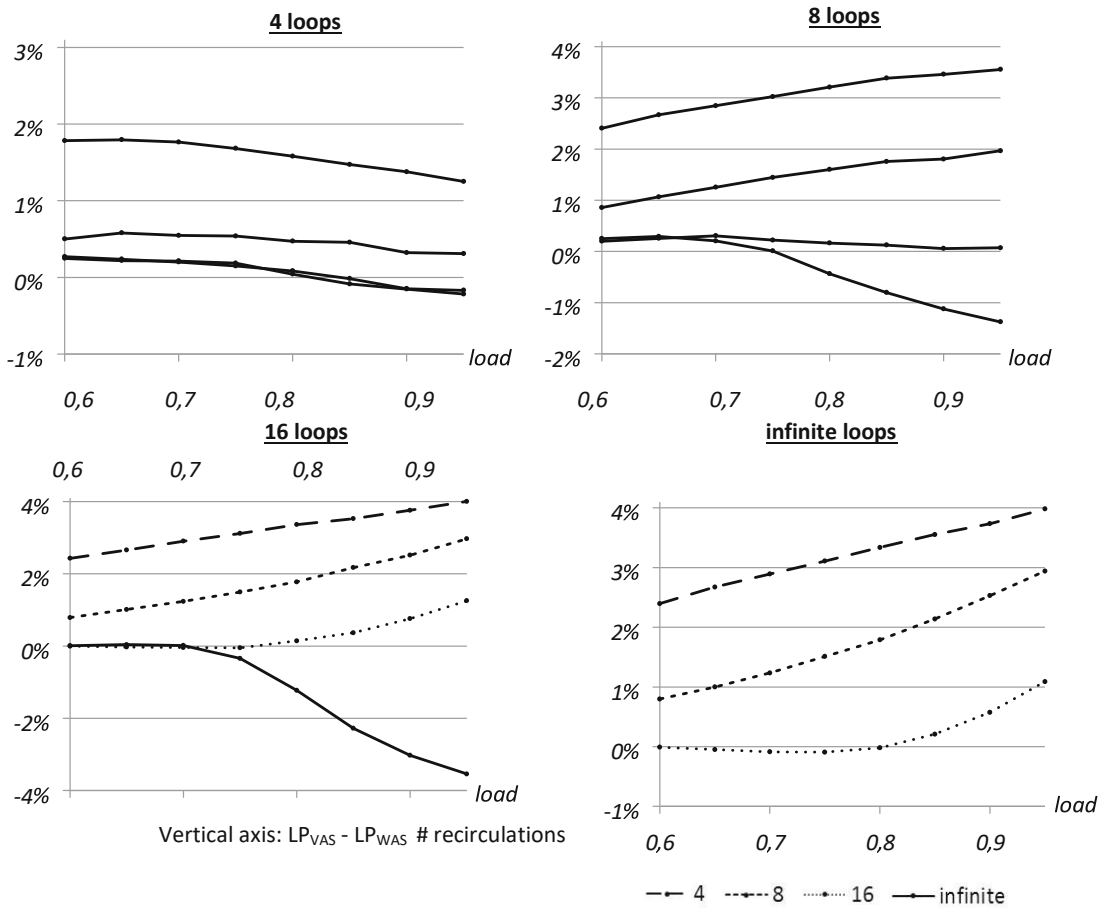

Fig.4. LP difference between VAS and WAS for different parameter combinations.

4 loops 8 loops

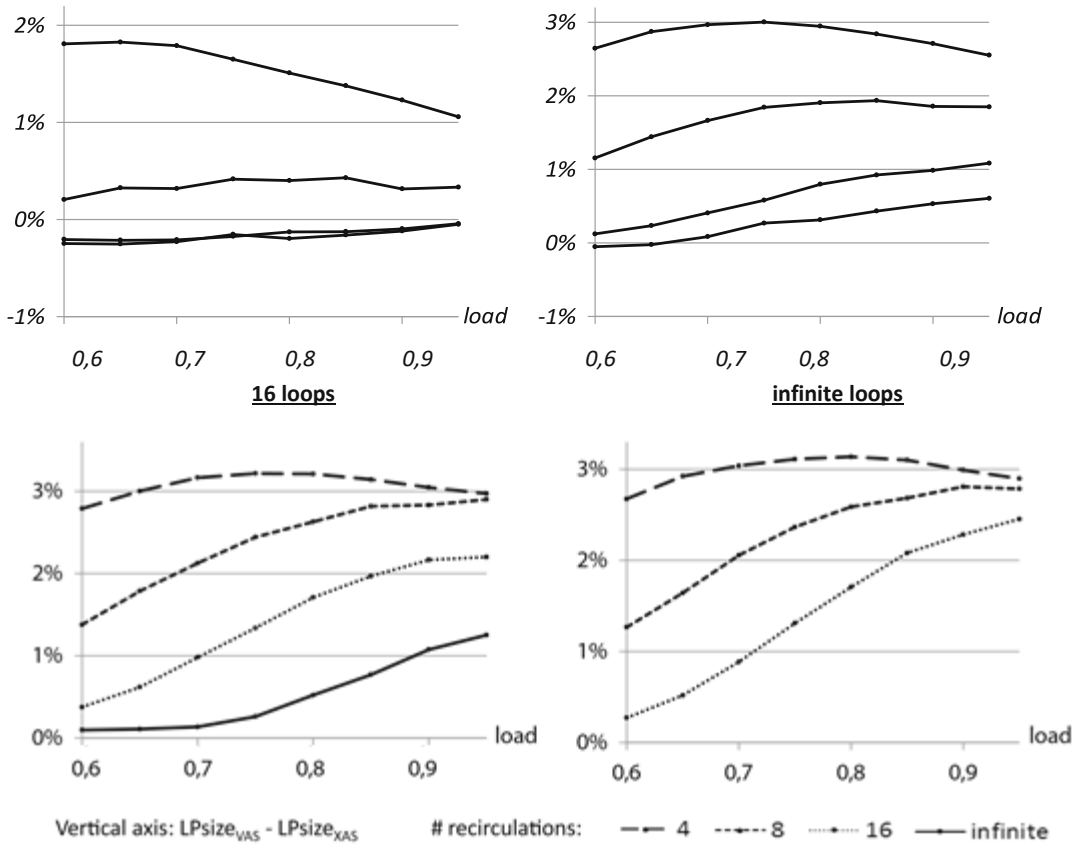

Fig.5. LPsize difference between VAS and XAS for different parameter combinations.

WAS and XAS, the WAS algorithm seems to be better suited to improve LP with improvements of up to 28% in a setting with a low load (0.6) and a large buffer size (16), but this only for small maximum number of recirculations (4). The XAS algorithm on the other hand seems to better at improving the LPsize, with improvements of up to 41% for certain parameter combinations, i.e. for a load value of 0.6, a maximum number of recirculations of 16 and a number of loops equal to 16 or infinite.

Note that the trends along certain parameter variations of Tables 3 and 4 are not necessarily visible in the actual performance differences of VAS, WAS and XAS. This is clear from Figs. 4 and 5 showing the difference in LP of VAS and WAS (Fig. 4) and the difference in LPsize of VAS and XAS (Fig. 5) for all parameter combinations. For example although the relative improvement tends to decrease with an increasing load, the actual differences are more steady or even increase. This is as both LP and LPsize increase with an increasing load and a smaller relative improvement can thus correspond with a larger actual performance difference. Similarly, as both LP and LPsize decrease with an increasing number of loops, the relative higher performance improvements for a higher number of loops do not necessarily translate in higher performance differences. This as opposed to the maximum number of recirculations, for which both the relative

improvement and the actual performance difference decrease with an increasing number.

Conclusions and Future Work

In this paper we proposed the WAS and XAS scheduling algorithms for optical fiber loop buffers in a variable sized packets setting. Performance was evaluated by means of Monte Carlo simulation and showed that they outperform the current state-of-the-art void-avoiding (VAS) schedule in both unlimited and restricted buffer settings. Both algorithms succeed in doing so by "looking ahead", i.e. by taking into account the schedule of other packets present in the system. In this way XAS is capable of improving packet delay with almost 20% for high loads in the unlimited buffer setting. In the restricted buffer setting WAS algorithm is better at improving loss probability (LP) while XAS is better at improving LPsize (related to LP, taking into account packet size). Both algorithms succeed to improve performance with dozens of percentages. To improve the performance of both WAS and XAS even further and for a wider parameter range, future work should focus on taking into account more packets simultaneously or using an optimizable threshold to decide among packet transmission orders.

References

1. Cisco Press Release: The Zettabyte Era: Trends and Analysis (2017). https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networkingindex-vni/vni-hyperconnectivity-wp.html
2. Nippon Telegraph and Telephone Corporation: One Petabit per Second Fiber Transmission Over a Record Distance of 200 km (2017). https://www.ntt.co.jp/news2017/1703e/pdf/170323a.pdf
3. Verma, S., Chaskar, H., Ravikanth, R.: Optical burst switching: a viable solution for terabit IP backbone. IEEE Netw. **14**(6), 48–53 (2000)
4. Chen, Y., Qiao, C., Yu, X.: Optical burst switching: a new area in optical networking research. IEEE Netw. **18**(3), 16–23 (2004)
5. Xiong, Y., Vandenhoute, M., Cankaya, H.C.: Control architecture in optical burst switched WDM networks. IEEE J. Sel. Areas Commun. **18**(10), 1838–1851 (2000)
6. El-Bawab, T.S., Shin, J.-D.: Optical packet switching in core networks: between vision and reality. IEEE Commun. Mag. **40**(9), 60–65 (2002)
7. Szczesniak, I.: Overview of optical packet switching. Theor. Appl. Inform. **21**(3–4), 167–180 (2009)
8. Triki, A., Gravey, A., Gravey, P., Morvan, M.: Long-term CAPEX evolution for slotted optical packet switching in a metropolitan network. In: Proceedings of International Conference on Optical Network Design and Modeling (ONDM), pp. 1–6, May 2017
9. Mukherjee, B.: Architecture, control, and management of optical switching networks. In: Proceedings of Photonics in Switching, pp. 43–44, August 2007
10. Heddeghem, W.V., Lannoo, B., Colle, D., Pickavet, M., Musumeci, F., Pattavina,A., Idzikowski, F.: Power consumption evaluation of circuit-switched versus packet switched optical backbone

networks. In: Proceedings of 2013 IEEE Online Conference on Green Communications, pp. 56–63, October 2013
11. Yao, S., Mukherjee, B., Yoo, S.J.B., Dixit, S.: A unified study of contention resolution schemes in optical packet-switched networks. J. Lightwave Technol. **21**(3), 672–683 (2003)
12. Yoo, M., Qiao, C., Dixit, S.: The effect of limited fiber delay lines on QoS performance of optical burst switched WDM networks. In: Proceedings of 2000 IEEE International Conference on Communications, vol. 2, pp. 974–979, June 2000
13. Tanemura, T., Soganci, I.M., Oyama, T., Ohyama, T., Mino, S., Williams, K.A.,Calabretta, N., Dorren, H.J.S., Nakano, Y.: Large-capacity compact optical buffer based on InP integrated phased-array switch and coiled fiber delay lines. J. Lightwave Technol. **29**(4), 396–402 (2011)
14. Burmeister, E., Blumenthal, D., Bowers, J.: A comparison of optical buffering technologies. Opt. Switch. Network. **5**(1), 10–18 (2008)
15. Langenhorst, R., Eiselt, M., Pieper, W., Grosskopf, G., Ludwig, R., Kuller, L.,Dietrich, E., Weber, H.G.: Fiber loop optical buffer. J. Lightwave Technol. **14**(3), 324–335 (1996)
16. Liu, A., Wu, C., Lim, M., Gong, Y., Shum, P.: Optical buffer configuration based on a 3 × 3 collinear fibre coupler. Electron. Lett. **40**, 1017–1019 (2004)
17. Fu, S., Shum, P., Ngo, N.Q., Wu, C., Li, Y., Chan, C.: An enhanced SOA-based double-loop optical buffer for storage of variable-length packet. J. Lightwave Technol. **26**(4), 425–431 (2008)
18. Tian, C.-Y., Wu, C.-Q., Sun, G.-N., Li, X., Li, Z.-Y.: Quality improvement of the dual-wavelength signals in DLOB via power equalization. Optoelectron. Lett. **4**(5), 361–364 (2008)
19. Wang, Y., Wu, C., Wang, Z., Xin, X.: A new large variable delay optical buffer based on cascaded double loop optical buffers (DLOBs). In: Proceedings of 2009 Conference on Optical Fiber Communication, pp. 1–3, March 2009
20. Rogiest, W., Fiems, D., Dorsman, J.-P.: Analysis of fibre-loop optical buffers with a void-avoiding schedule. In: Proceedings of Valuetools 2014, p. 7, December 2014
21. Rostami, A., Chakraborty, S.S.: On performance of optical buffers with specific number of circulations. IEEE Photon. Technol. Lett. **17**(7), 1570–1572 (2005)
22. Van Hautegem, K., Rogiest, W., Bruneel, H.: Scheduling in optical switching: deploying shared wavelength converters more effectively. In: Proceedings of 2014 IEEE International Conference on Communications (ICC), pp. 3418–3424, June 2014
23. Van Hautegem, K., Rogiest, W., Bruneel, H.: Optical switching for variable size packets: improved void filling through selective void creation. In: Proceedings of 11th International Conference on Queueing Theory and Network Applications (QTNA), pp. 1–8, December 2016
24. Van Hautegem, K., Rogiest, W., Bruneel, H.: Improving performance and energy consumption of shared wavelength converters in OPS/OBS. Opt. Switch. Netw. **17**, 38–51 (2015)
25. Lambert, J., Van Houdt, B., Blondia, C.: Single-wavelength optical buffers: non equidistant structures and preventive drop mechanisms. In: Proceedings of Networking and Electronic Commerce Research Conference (NAEC), pp. 545–555, October 2005

T-WAS and T-XAS algorithms for Fiber-loop Optical Buffers

Kurt Van Hautegem[a,*], Mario Pinto[b,**], Diego Gomez[b], Herwig Bruneel[a], Wouter Rogiest[a]

[a] *Ghent University, Department of Telecommunications and Information Processing (TELIN), SMACS Research Group, St.-Pietersnieuwstraat 41, B-9000 Ghent, Belgium*

[b] *Universidad Antonio Nariño, Kra 3 Este No. 47A-15, Bogota, Colombia*

Abstract

In optical packet/burst switched networks fiber loops provide a viable and compact means of contention resolution. For fixed size packets it is known that a basic void-avoiding schedule (VAS) can vastly outperform a more classical pre-reservation algorithm as FCFS. For the setting of a uniform distributed packet size and a restricted buffer size we proposed two novel forward-looking algorithms, WAS and XAS, that, in specific settings, outperform VAS up to 20% in terms of packet loss. This contribution extends the usage and improves the performance of the WAS and XAS algorithms by introducing an additional threshold variable. By optimizing this threshold, the process of selectively delaying packet longer than strictly necessary can be made more or less strict and as such be fitted to each setting. By Monte Carlo simulation it is shown that the resulting T-WAS and T-XAS algorithms are most effective for those instances where the algorithms without threshold can offer no or only limited performance improvement.

Keywords: Fiber Loops, OPS, OBS, scheduling.

* K. Van Hautegem and M. Pinto are joint first authors.

** Corresponding author

Email addresses: kurt.vanhautegem@ugent.be (Kurt Van Hautegem), maupinto@uan.edu.co (Mario Pinto), dfgomezc@uan.edu.co (Diego Gomez), herwig.bruneel@ugent.be (Herwig Bruneel), wouter.rogiest@ugent.be (Wouter Rogiest)

Preprint submitted to Optical Switching and Networking September 13, 2019

1. Introduction

As video on demand (VoD) services increase in popularity and 4K video quality will become the new normal, global IP traffic is expected to grow at a compounding annual rate of 24 % between 2016 and 2021 [1]. As wavelength 5 and spatial multiplexing allows optical fiber technology to reach dazzling bandwidths of up to 1 Petabit/s [2] over one fiber, while the bandwidth over a single wavelength was recently pushed to 500 Gbit/s [3]. It seems that our unlimited demand for bandwidth can be met without a problem. Unfortunately, capacity in existing optical networks is not limited by the connections but by the nodes in which slow electronic switching or inflexible optical circuit switching muffle the optical highway capacity.

Promising solutions to address the issues in optical backbones are optical burst switching (OBS) [4, 5, 6] and optical packet switching (OPS) [7, 8, 9]. In these packet based switching techniques, optical signals are, similar to optical circuit switching (OCS) [10, 11], kept in the optical domain to avoid slow optical-electronic-optical (OEO) conversions but, similar to electronic switching, processed as packets to increase statistical multiplexing efficiency. Although RAM buffering in the nodes is infeasible because it requires OEO conversions, at least a limited amount of buffering remains advisable to address the unavoidable contention that arises in the nodes [12, 13].

One of the most compact implementations of optical buffering today is a fiber loop buffer. As opposed to feed-forward buffers where every line is traversed only once [14], fiber loop buffers allow contending packets to recirculate multiple times within the same coiled fiber loop [15, 16]. Although alternative designs as dual-loop optical buffers exist [17, 18, 19, 20], most use a set of single fiber loops in parallel which can all accommodate a single packet at once (shown in Fig. 1). Because packets can only exit a loop after a round number of recirculations, fiber loops can only provide a discrete set of delays. As opposed to electronic memory

(RAM), packets can thus not be retrieved at will, resulting in small time gaps or voids in between packets on the outgoing line. Moreover, as packets recirculate in the same loop, fiber loops can only accommodate packet sizes smaller than or equal to their loop length and packet length directly limits the resolution of possible delays. Since the footprint is preferably kept small, with a small number of fiber loops, and also the number of recirculations a packet can make in a loop is kept low to prevent signal degeneration, scheduling algorithms in which the resources are used as efficiently as possible are needed to achieve low packet loss and/or packet delay.

In [21] the authors deal with the dimensioning of a router having N input data ports and M fiber delay lines (FDLs) over a OBS (Optical Burst Switching) network. The size of the tunable FDLs can change in order to match the buffered burst [21]. Contribution [22] presents the study of an optical packet switch having recirculating FDL buffers along with wavelength converters. A Markovian arrival process with marked transitions (MMAP) is considered. A lower and upper bound of the packet loss rate (PLR) of the particular switch is determined. Non-degenerate buffer depth may improve the packet loss performance in particular with bursty traffic [23]. In [23], a performance study is presented of a void filling algorithm using both a non-degenerate and degenerate (uniform) fiber delay lines [23]. The packet loss performance considering both non-degenerate and degenerate delay lines using a void filling algorithm is shown for an optical router having feedback delay lines using self similar traffic [23]. The authors in [24] developed a queueing model used in feedback-type shared-per-node recirculating FDL optical buffers for asynchronous optical switching nodes. The optical packets can recirculate through the FDLs provided that the number of recirculations does not exceed an established limit in order to meet signal loss requirements [24]. The packet arrival process in the optical switch is Poisson [24]. An MMPP-based queueing model is presented involving fixed-point iteration for studying the performance of feedback-type shared-per-node recirculating FDL buffers. The authors in [25] developed an optical packet switch architecture, by using Arrayed Waveguide Grating Router

(AWGR), needing considerably less tunable wavelength converters compared with wavelength channels, so the contention is managed by a low-power recirculating optical delay module. Contribution [26] is directed at reducing crosstalk at a CrossPoint switch with multiple recirculations [26], involving simplified time-slot interchange configurations with less fiber-delay lines. In [27], a simulation study is presented of optical node considering an n x m optical switch along with recirculating optical delay lines. Also they developed a mathematical model for the switch architecture through packet queueing control in order to find the blocking probability of the incoming traffic. The study includes assignment of priority to packets in view of a contention resolution algorithm.

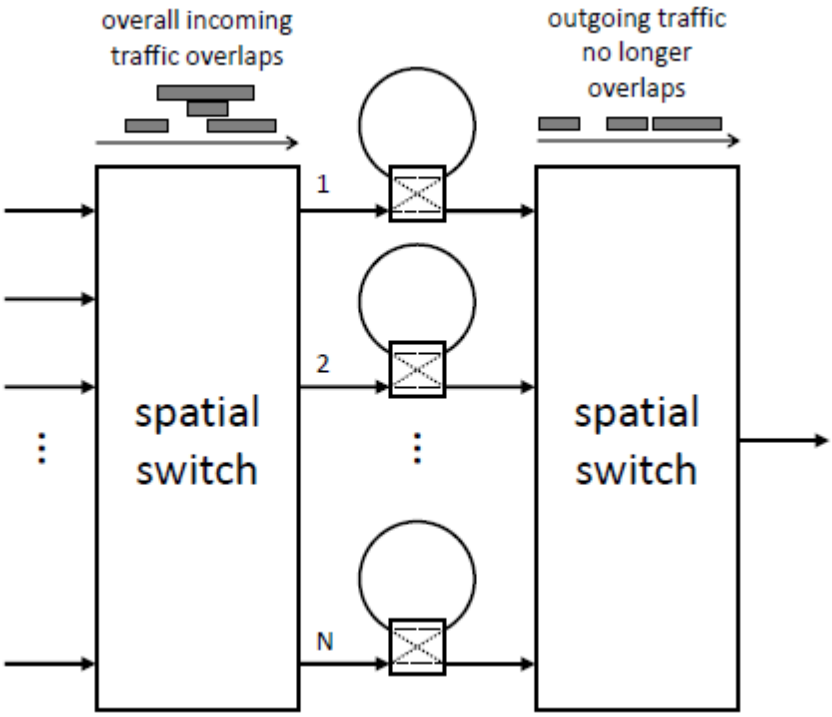

Figure 1: Parallel optical fiber loop buffer to resolve contention at the input.

In [28] an analytical model is used to evaluate performance of the void-avoiding schedule (VAS) for fixed size packets equal to the loop length. The 75 void-avoiding schedule is a post-reservation scheme [29], allowing the packets to

enter the buffer freely, only deciding later when a packet has to exit its loop. In [28] it is shown that performance of the VAS is significantly better than that of algorithms with a pre-reservation scheme, e.g. FCFS, in which the number of recirculations is decided upon arrival of a packet.

In [30] we evaluated the performance of the void-avoiding schedule (VAS) for uniform distributed packet size in both unlimited and constricted fiber loop buffer settings. We also proposed two new algorithms, WAS and XAS, which, to the best of our knowledge, are the first known algorithms to outperform VAS. Particularly, both yield significant improvement for variable-length packet size. Both algorithms succeed in doing so by "looking ahead", i.e. by taking into account the schedule of other packets present in the system. In this way XAS is capable of improving packet delay with almost 20% for high loads in the unlimited buffer setting. In the restricted buffer setting, the WAS algorithm showed to be better at improving loss probability (LP, number of lost packets/total number of packets) while XAS is better at improving LPsize (cumulative size of lost packets/total size of all packets). Both algorithms succeed to improve performance with dozens of percentages.

In this paper we further improve the performance and extend the usage of the WAS and XAS algorithms to a wider parameter range by introducing an additional threshold variable, with new algorithms T-WAS and T-XAS. By optimizing this threshold, the process of selectively delaying packet longer than strictly necessary can be made more or less strict and as such be fitted to each setting. This paper is organized as follows. The general system model and assumptions are discussed in Section 2. Section 3 presents the scheduling algorithms WAS and XAS, along with an example and the summarized results of [30]. The threshold-extended algorithms T-WAS and T-XAS and their performance evaluation are the subject matter of section 4. Conclusions are drawn in Section 5, along with a discussion of future work.

2. System Model and Assumptions

Throughout the paper the same continuous-time setting as in [31, 32, 33, 30] is supposed. The fiber loop buffer is assumed to be located at and dedicated to a single outgoing port of an optical switch. Wavelength convertors, if present within the switch, are assumed to perform conversion to a single outgoing wavelength associated with this single outgoing port. The analysis can thus be limited to a single wavelength. We assume the joint packet arrival at the output port on this single wavelength is a Poisson process, i.e. the inter-arrival times T are exponentially distributed with an average of E[T]. The length of arriving packets, B, is assumed to be uniform distributed on the interval [0; S] with an average of E[B] = S/2. Related, the overall incoming traffic load at the output port is given by ρ=E[B]/E[T] = S/(2· E[T]).

Because of the nature of the arrival process it is possible that different arrivals overlap upon their arrival at which instant one of the contending packets has to be temporarily buffered in one of the fiber loops. We assume a set of parallel fiber loops of length S that, indepently of the packet's size, can accommodate a single packet. The assignable delays to a packet are thus integer multiples of S. The number of fiber loops and the maximum number of times a packet can recirculate are both varied. Combinations of both finite (4, 8, and 16) and infinite values for these buffer parameters are evaluated. In the simulations, fiber loops are assumed to have a length of one time unit and packets to be uniformly distributed on the interval [0; 1]. Load is varied from 0:6 to 0:95 in steps of 0:05 by changing the average inter-arrival time of the Poisson arrival process. The arrival of 10^6 packets is simulated 10 times for each algorithm and parameter combination. In this way adequate average performance measures and accompanying confidence intervals are obtained.

As extending the analytical method from [28] to different algorithms, settings and packet size distributions proved to be too challenging, the performance

of the algorithms is evaluated by means of Monte Carlo Simulations. Specifi_-
cally, all algorithms are programmed in Matlab using a discrete event simulation
(DES). In a DES, the system is modelled as a sequence of events marked by
their particular instant in time, i.e. the simulation is event-based. The system
state changes from 140 one event to the next and does not change in-between events.
This is as opposed to continuous simulation in which time is broken into small
pieces called time slices. At each ending of a time slice, the system state is
(possibly) changed based on the events that happened in the last time slice.
Because DES simulations do not simulate every time slice, they are far more
efficient in terms of computational resources.

3. VAS, WAS and XAS Scheduling Algorithms

3.1. Approach and concept

As in a buffer loop setting the well-known FCFS algorithm is outperformed
by the void avoiding schedule (VAS), we chose the latter as our benchmark al-
gorithm in [30]. In the VAS, packets that arrive are transmitted immediately
if the outgoing line is available or stored in a fiber loop if not. After each loop
recirculation, the availability of the outgoing line is checked. If the outgoing
line is available upon such a check, the packet exits its fiber loop and is sent.
If the outgoing line is not available, the packet is recirculated again and the
procedure is repeated. The VAS does not preserve the arrival order and is a
post-reservation algorithm.

In terms of resource usage, VAS is a greedy algorithm, sending a packet
whenever a transmission opportunity, i.e. an available outgoing line and either
an arrival or a finished recirculation, arises. The newly proposed WAS algo-
rithm, where W may refer to the aim of minimizing the Wait for the outgoing
line to become available after departure of the two packets, is more considerate
and well aware of the other packets present in the system. When a transmission
opportunity arises, WAS will first calculate the time needed to send each com-

bination of two packets 165 (packets i and j, i 6= j) present in the system. When N + 1 packets are present in the system (i.e. packets which are either in the buffer or a new arrival), this gives an N _(N +1) two dimensional matrix with the time needed to send each combination. In this matrix, rows are assumed to be the first packet sent and columns the second (rows before columns). Only when the packet is the first packet (i.e. the row, not the column) of the lowest combination in this matrix, the WAS algorithm will send this packet. Otherwise, depending on whether the transmission opportunity is a new arrival or a finished recirculation, this packet is buffered in a new loop or given another round in its loop. Note that in this situation the lowest combination will not necessarily be the next two packets to be sent as the matrix is re-evaluated at every transmission opportunity. Similarly when the packet that triggered the transmission opportunity is actually sent, the packet that was also part of the lowest combination is not guaranteed to be transmitted next.

Similar to the WAS algorithm, the XAS algorithm, where X may refer to the aim of eXtending the period during which the outgoing line is effectively used by the two packets, also calculates a combination matrix to decide upon transmission when a transmission opportunity arises. In this matrix the efficiency of the outgoing line is calculated for each combination by dividing the sum of both packet lengths by the total time needed to send each combination. Only when the packet that triggered the transmission opportunity is the first packet of the combination with the highest efficiency, the XAS algorithm will actually send this packet.

The transmission opportunity depends on two facts: the availability of an outgoing line and either an arrival or a finished recirculation, that is currently scheduled in the agenda. The packet at present time which finished its recirculation or arrives (with an available outgoing line), causes the transmission opportunity. If the arriving packet finds free the outgoing line, then the matrix is calculated, and depending on the results, it is scheduled in the outgoing line,

or it is buffered (if there are available loops), but avoiding unnecessary losses (i.e an available outgoing line and no free loops).

When either the number of fiber loops or the maximum number of recirculations is constricted, the WAS and XAS algorithms will transmit a packet upon a transmission opportunity if doing otherwise would result in an immediate and unnecessary loss. Suppose for example that upon arrival of a new packet the output line is available but all of the fiber loops (finite set) are occupied by other packets. In such a case both WAS and XAS will transmit the new arrival, even though a more favorable combination may be present in the fiber loops. By doing so the new arrival need not be dropped and unnecessary loss is prevented. Likewise when the maximum number of recirculations is reached upon the end of a loop recirculation, WAS and XAS will always transmit the packet if the output line is available. In the terminology of [34] we could say that both WAS and XAS are tuned to avoid the use of preventive drop.

Note that in the case of fixed size packets both WAS and XAS schedule in exactly the same way as VAS. Indeed, as all packets have the same size, the combination that minimizes the time to transmit a pair of packets will always consist of the packet causing the transmission opportunity. Only when packet sizes are not equal to a fixed size, WAS and XAS schedule different, and thus possibly better, than VAS.

3.2. An example

In Fig. 2 and table 1 the difference between VAS, WAS and XAS is shown for an example with 5 fiber loops. Fig. 2 shows the occupation of the fiber loops upon a transmission opportunity (i.e., an arrival of a new packet and an available outgoing line in this case). The different fiber loops are shown horizontally and the packets they contain by means of an arrow on top of each line. In this visual representation the fiber loops are represented horizontally, i.e., as if they were laid at after being disconnected in the point where one can decide

upon recirculation of a packet. As time passes by, the arrows move to the right, possibly restarting their motion on the left side of the fiber when the arrowhead reaches the right end of the fiber loop (i.e., when the packet is recirculated). In the example of Fig. 2, fiber loops 1-4 are occupied while fiber loop 5 is still available. The newly arrived packet is shown in parallel to the already scheduled packets and aligns with the right side of the fiber loops. As a transmission opportunity is created by the new packet and the available outgoing line, one thus has to decide upon transmission or buffering of the packet.

As mentioned above, when either the number of fiber loops or the maximum number of recirculations is constricted, the WAS and XAS algorithms will transmit a packet upon a transmission opportunity if doing otherwise would result in an immediate and unnecessary loss. In the present example of Fig. 2 and table 1 both WAS and XAS will send the new packet first if the setting would contain 4 instead of 5 loops. This is because all 4 fiber loops would be occupied and thus not available to schedule a new packet.

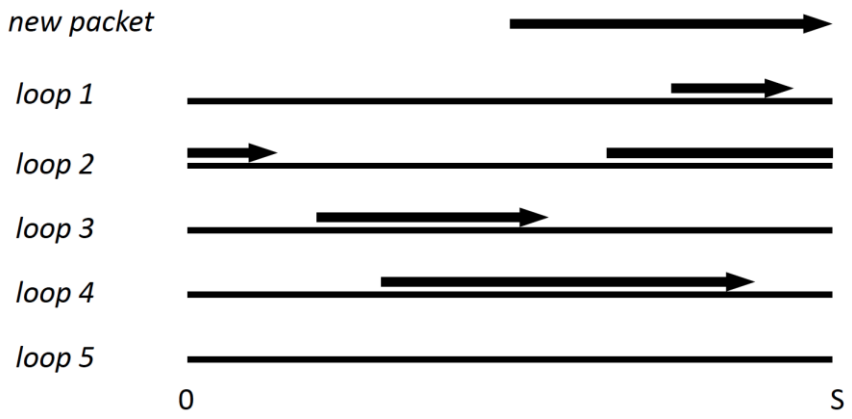

Figure 2: Example of a fiber loop occupancy with 5 fiber loops upon arrival of a new packet and an available outgoing line (i.e., a transmission opportunity).

3.3. Performance results

To obtain a complete and representative image of the performance of the various algorithms we looked at different performance measures for different

WAS (time to send (·S))		Second packet				
		new	*1*	*2*	*3*	*4*
first packet	*new*	/	1.25	1.35	1.80	1.70
	1	1.50	/	1.35	**0.80**	1.70
	2	2.50	2.25	/	1.80	2.70
	3	1.50	1.25	1.35	/	1.70
	4	1.50	1.25	1.35	1.80	/
XAS (efficiency)		*Second packet*				
		new	*1*	*2*	*3*	*4*
first packet	*new*	/	55.2 %	73.3 %	47.8 %	63.6 %
	1	46.0 %	/	50.4 %	68.8 %	45.3 %
	2	39.6 %	30.2 %	/	47.2 %	39.6 %
	3	57.3 %	44.0 %	63.0 %	/	55.3 %
	4	72.0 %	61.6 %	**79.3 %**	52.2 %	/

Table 1: Percentage-wise performance improvement in LP and LPsize of XAS relative to VAS for different load values in different restricted buffer settings.

settings in [30]. For the case with an unlimited number of fiber loops and no restriction on the number of recirculations, no packets are lost and we compare the algorithms on the average packet delay. In case either, or both, the number of fiber loops or the maximum number of recirculations is limited, the loss probability (LP), or equivalently, packet loss, is our main performance measure. In addition, we study a related performance measure which we refer to as LPsize, accounting for the relative total size of packets lost to the relative total size of packets, or, equivalently, the relative amount of data lost with respect to the total amount of data that arrived. In this section only the most important performance results of VAS and WAS are discussed. For a more detailed analysis we refer to [30].

Fig. 3 shows the average packet delay for the setting with an unlimited num-

ber of fiber loops and no restriction on the maximum number of recirculations. As in this setting, for the load values investigated, the FCFS algorithm results in an unstable regime, it is not included in the graph. From Fig. 3 it is clear that in the unrestricted case and with a uniform packet size distribution only XAS can outperform VAS. Table 2 shows the performance improvement (in percentage) XAS can obtain in waiting time relative to VAS. As the load increases, the obtainable improvement also goes up, reaching an improvement of almost 20 % for a load of 0:95.

Packet delay reduction	*load*							
	0.60	0.65	0.70	0.75	0.80	0.85	0.90	0.95
XAS	1.1 %	2.4 %	4.2 %	7.3 %	9.6 %	13.7 %	17.2 %	18.1 %

Table 2: Percentage-wise performance improvement in packet delay of XAS relative to VAS for different load values in an unrestricted buffer setting.

Table 3 shows the LP and LPsize of the VAS algorithm in different restricted buffer settings. Note that LP and LPsize have the same values for the VAS algorithm. This is because the length of a packet does not influence the way a packet is scheduled in VAS. In table 4 (WAS) and 5 (XAS) the performance improvements (in percentage) compared to VAS of LP and LPsize are shown. This is done for load values of 0:6, 0:7, 0:8 and 0:9, and all combinations of the number of loops and the maximum number of recirculations (both take on values of 4, 8, 16 and infinity). From these tables it is clear that not for all combinations of parameters a performance improvement is possible. In general, but not always, performance improvements increase for lower load values, a lower number of maximum recirculations, and a higher number of loops. Comparing WAS and XAS, the WAS algorithm seems to be better suited to improve LP with improvements of up to 28 % in a setting with a low load (0:6) and a large buffer size (16), but this only for small maximum number of recirculations (4). The XAS algorithm on the other hand seems to better at improving the LPsize,

with improvements of up to 41% for certain parameter combinations, i.e. for a load value of 0:6, a maximum number of recirculations of 16 and a number of

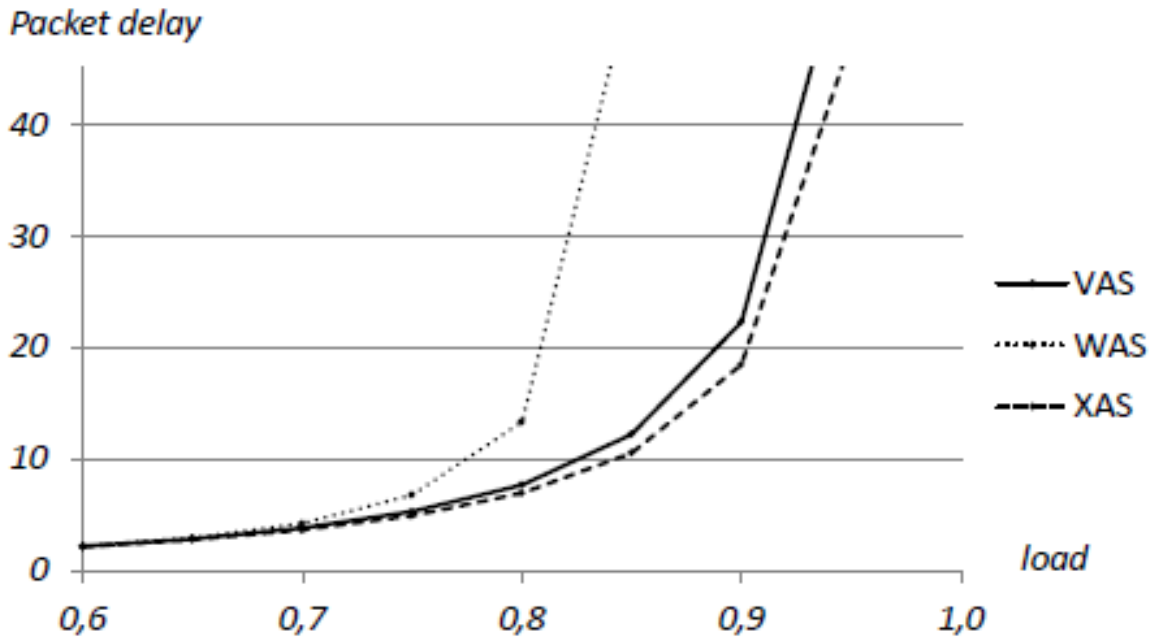

Figure 3: Packet delay for the VAS, WAS and XAS algorithms in an unrestricted buffer setting.

VAS		Load															
		0.60				0.70				0.80				0.90			
		# loops				# loops				# loops				# loops			
		4	8	16	∞	4	8	16	∞	4	8	16	∞	4	8	16	∞
		LP = LPsize (%)															
max recirculations	4	10.3	8.8	8.8	8.8	15.7	13.3	13.2	13.2	21.2	18.2	18.0	18.0	26.5	23.0	22.6	22.6
	8	7.2	3.6	3.4	3.4	12.6	7.6	6.8	6.8	18.7	12.7	11.2	11.3	24.6	18.4	16.2	16.1
	16	6.6	1.5	0.7	0.7	12.0	4.8	2.5	2.4	18.1	10.3	6.1	5.8	24.1	16.7	11.3	10.5
	∞	6.6	1.2	0.0	0.0	12.0	4.3	0.4	0.0	18.0	9.8	3.1	0.0	24.1	16.4	9.2	0.0

Table 3: LP and LPsize values for VAS for different load values in different restricted buffer settings.

4. WAS and XAS: threshold extension

In order to strive for a further improvement of the performance of the WAS and XAS algorithms, we introduce an additional threshold variable. This thresh-

WAS		Load															
		0.60				0.70				0.80				0.90			
		# loops				# loops				# loops				# loops			
		4	8	16	∞	4	8	16	∞	4	8	16	∞	4	8	16	∞
		LP reduction (%)															
max recirculations	4	17.3	27.3	27.7	27.4	11.3	21.4	22.0	21.9	7.5	17.7	18.7	18.6	5.2	15.0	16.6	16.5
	8	7.0	23.5	23.4	23.7	4.3	16.6	18.2	18.1	2.5	12.6	15.8	15.9	1.3	9.8	15.6	15.7
	16	4.1	13.7	-0.7	-1.3	1.7	6.4	-1.6	-3.6	0.5	1.6	2.3	-0.3	-0.6	0.3	6.7	5.5
	∞	3.8	21.2	53.6	NaN	1.8	4.8	3.4	NaN	0.2	-4.4	-39.2	NaN	-0.6	-6.9	-32.7	NaN
		LPsize reduction (%)															
max recirculations	4	8.8	8.8	8.9	8.4	3.4	1.3	1.2	0.8	0.5	-3.2	-3.7	-3.8	-1.2	-5.6	-6.8	-6.8
	8	3.4	-3.1	-8.5	-8.1	0.8	-10.1	-16.7	-16.6	-0.8	-11.7	-20.7	-20.6	-1.9	-11.9	-21.5	-21.5
	16	3.9	-5.0	-59.8	-61.3	1.2	-10.1	-59.7	-63.4	0.0	-12.2	-51.3	-58.6	-1.2	-11.3	-40.4	-49.5
	∞	3.6	21.5	53.0	NaN	1.8	4.9	3.3	NaN	0.3	-4.4	-39.0	NaN	-0.6	-6.8	-32.6	NaN

Table 4: Percentage-wise performance improvement in LP and LPsize of WAS relative to VAS for different load values in different restricted buffer settings.

XAS		Load															
		0.60				0.70				0.80				0.90			
		# loops				# loops				# loops				# loops			
		4	8	16	∞	4	8	16	∞	4	8	16	∞	4	8	16	∞
		LP reduction (%)															
max recirculations	4	9.9	15.9	16.2	16.3	4.6	8.3	8.7	8.7	1.2	3.2	3.6	3.6	-0.6	-0.1	0.1	0.2
	8	-0.9	10.1	12.4	12.1	-1.0	3.4	5.1	5.5	-1.2	-0.6	-0.2	0.2	-1.6	-2.8	-3.6	-3.4
	16	-3.3	-3.9	0.7	0.7	-2.0	-0.7	1.3	0.7	-1.1	0.2	-2.0	-2.5	-0.8	-0.1	-4.3	-6.2
	∞	-3.8	-4.3	-9.3	NaN	-1.8	2.0	7.6	NaN	-1.1	3.3	13.4	NaN	-0.5	3.2	10.5	NaN
		LPsize reduction (%)															
max recirculations	4	17.5	30.1	30.6	30.6	11.4	22.3	23.2	23.0	7.1	16.2	17.3	17.4	4.6	11.8	13.0	13.2
	8	2.9	31.6	37.9	37.6	2.5	22.0	29.7	30.1	2.1	15.0	22.5	23.0	1.3	10.1	16.9	17.4
	16	-3.1	8.3	41.0	40.9	-1.8	8.5	35.5	36.7	-0.7	7.8	26.3	29.4	-0.4	5.9	18.2	21.8
	∞	-3.8	-4.3	-13.9	NaN	-1.9	2.0	8.0	NaN	-1.1	3.2	13.4	NaN	-0.5	3.3	10.5	NaN

Table 5: Percentage-wise performance improvement in LP and LPsize of XAS relative to VAS for different load values in different restricted buffer settings.

old variable was varied to achieve optimal performance.

4.1. Approach and concept

The assumptions and the way in which the threshold extended algorithms schedule packets are similar to those discussed in section 3 for the WAS and XAS algorithms (without threshold extension). We will therefore only focus on those aspects that are specifically different for the T-WAS and T-XAS algorithms.

Similar to the WAS algorithm, the T-WAS algorithm (Threshold extension of WAS), when a transmission opportunity arises, first calculates the time needed to send each combination of two packets (packets i and j, $i \neq j$) present in the system. When N+1 packets are present in the system (i.e. packets in the buffer or a new arrival), this gives an N x (N + 1) two dimensional matrix with the time needed to send each combination. In this matrix, rows are assumed to be the first packet sent and columns the second (rows before columns). As opposed to WAS, T-WAS does not always discard the transmission opportunity when the packet causing the transmission opportunity is not the first packet (i.e. the row, not the column) of the lowest combination in this matrix. Instead, in an additional calculation, the T-WAS algorithm determines the ratio of the overall lowest combination of the matrix and the lowest combination containing the current packet as the first packet. When this ratio is lower than the threshold (varied and optimized in different simulation runs), the T-WAS algorithm, similarly to WAS algorithm, does not send the packet but rather, depending on whether the transmission opportunity is a new arrival or a finished recirculation, buffers it in a new loop or gives it another round in its loop. When, on the other hand, the ratio is higher than the chosen threshold, the difference between the optimal combination and the combination containing the packet that causes the transmission opportunity is considered small enough and the T-WAS algorithm will nevertheless send the packet causing the transmission opportunity. As the calculated ratio is the proportion of the time it takes to send the lowest combi-

nation to that of a combination that takes longer, it is always between 0 and 1. This is thus the range in which the threshold parameter is varied and optimized in simulation. Note that, similar to the WAS algorithm, when the packet that triggered the transmission opportunity is actually sent, the packet that was also part of the lowest local or overall combination is not guaranteed to be transmitted next. Similarly the overall lowest combination will not necessarily be the next two packets to be sent as the matrix is re-evaluated at every transmission opportunity.

The T-XAS algorithm (Threshold extension of XAS) calculates a similar combination matrix as the XAS algorithm to decide upon transmission when a transmission opportunity arises. In this matrix the efficiency of the outgoing line is calculated for each combination by dividing the sum of both packet lengths by the total time needed to send each combination. In a second calculation, the T-XAS algorithm calculates the ratio of the highest combination containing the packet causing the transmission opportunity as the first packet and the overall highest combination of the matrix. When this ratio is lower than the threshold (varied and optimized in different simulation runs), the T-XAS algorithm, similarly to the XAS algorithm, does not send the packet but rather, depending on whether the transmission opportunity is a new arrival or a finished recirculation, buffers it in a new loop or gives it another round in its loop. However, when the ratio is higher than the chosen threshold, the difference in efficiency between the optimal combination and the combination containing the packet that causes the transmission opportunity is considered small enough and the T-XAS algorithm will nevertheless send the packet causing the transmission opportunity. Although the ratio in the T-XAS algorithm is the inverse of the ratio of the T-WAS algorithm, it is, because of the way in which the elements of the matrix are calculated, also always between 0 and 1. Similarly to the T-WAS parameter, the T-XAS parameter is varied and optimized in this range. Both the T-WAS and T-XAS algorithm are illustrated by a single ow chart, shown in Fig. 4. Regular WAS and XAS correspond to the case where the threshold

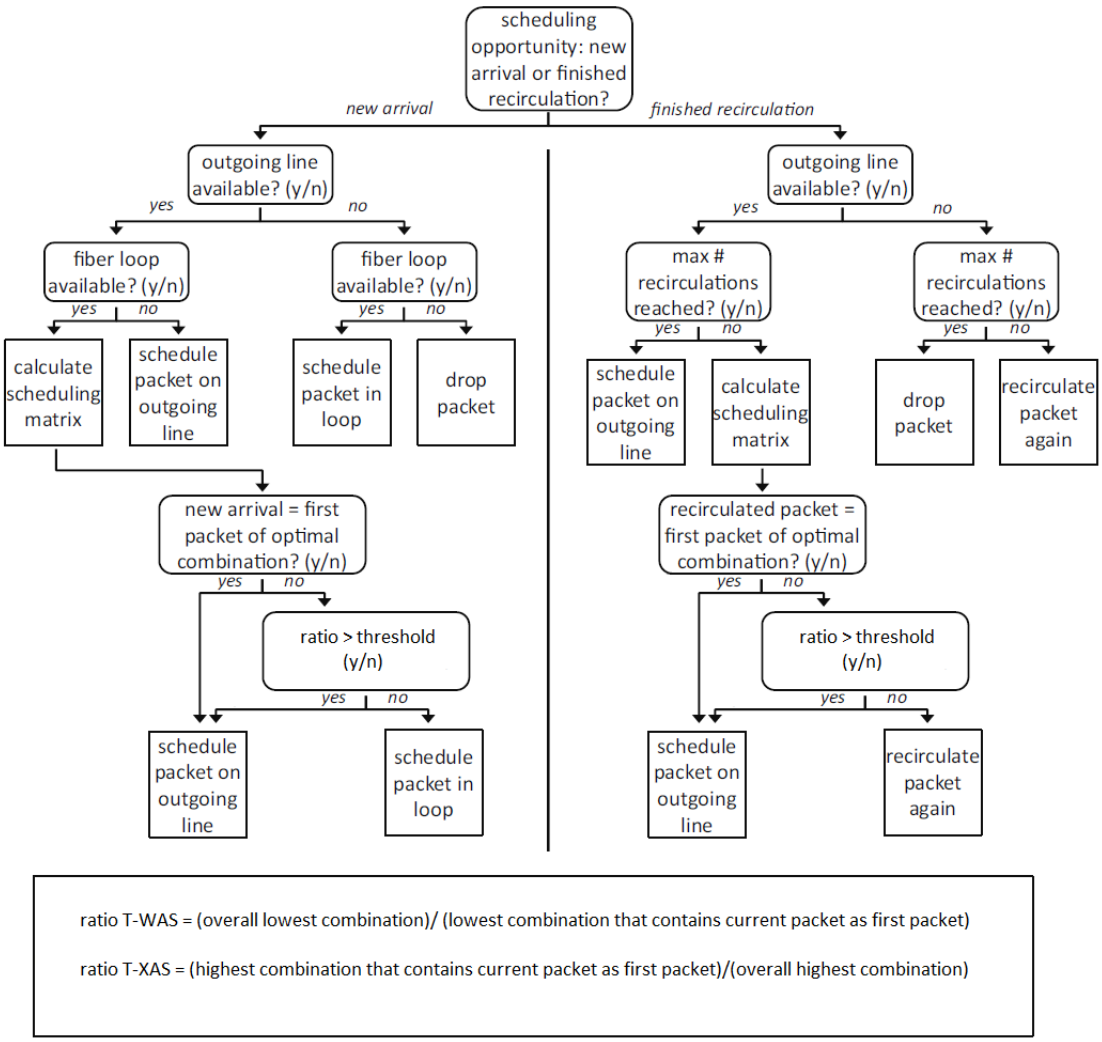

Figure 4: Flow chart illustrating the T-WAS and T-XAS algorithms. Regular WAS and XAS are obtained by setting the threshold to 1.

Similar to WAS and XAS, the T-WAS and T-XAS algorithms will prevent unnecessary loss. When either the number of fiber loops or the maximum number of recirculations is restricted, they will thus transmit a packet upon a transmission opportunity if doing otherwise would result in an immediate and unnecessary loss. This is the case when for example upon arrival of a new packet the output line is available but all of the fiber loops (finite set) are occupied by other packets. In such a case, both the T-WAS and T-XAS algorithms will transmit the new arrival even though the difference with the most favorable combination

in the matrix is large enough to argue against immediate transmission.

4.2. Performance results

To evaluate the performance of the T-WAS and T-XAS algorithms, we take the same assumptions as in section 2. More specifically fiber loops are assumed to have a length of one time unit and packets to be uniformly distributed on the interval [0; 1]. Load is varied from 0.6 to 0.95 in steps of 0.05 by changing the average inter-arrival time of the Poisson arrival process.

To obtain a complete and representative image of the performance of the various algorithms, we again look at different performance measures for different settings. For the case with an unlimited number of fiber loops and no restriction on the number of recirculations, no packets are lost and we compare the T-XAS algorithm on the average packet delay. The T-WAS algorithm is not evaluated for the unrestricted setting; as it was shown in section 3 that the WAS algorithm is unable to outperform VAS. We therefore choose to solely focus on the T-XAS algorithm in the unrestricted setting.

Table 6: Performance improvement in packet delay, optimal threshold and added value of the threshold mechanism of the T-XAS algorithm for different load values in an unrestricted buffer setting.

Packet delay reduction	*Load*							
	0.60	0.65	0.70	0.75	0.80	0.85	0.90	0.95
XAS vs. VAS (%)	1.1	2.4	4.2	7.3	9.6	13.7	17.2	18.1
T-XAS vs. VAS (%)	6.3	8.1	10.1	12.5	14.1	17.1	18.6	18.1
optimal threshold	0.85	0.90	0.90	0.90	0.90	0.95	0.95	1
added value threshold (%)	83	70	58	42	32	20	8	0

In case either, or both, the number of fiber loops or the maximum number of recirculations is limited, it was shown in section 3 that in general WAS outperforms XAS when it comes to improving LP compared to the reference algorithm VAS. The XAS algorithm on the other hand seemed to better at improving the LPsize. Taking into account these observations, we therefore choose to focus on improving LP for the T-WAS algorithm and LPsize for the T-XAS algorithm.

Table 6 extends the results of table 2 showing the performance improvement in waiting time (in percentage) the T-XAS algorithm can obtain when compared to the VAS algorithm (row 2). This performance improvement is calculated using the same reference base, i.e. the performance of the VAS algorithm. The corresponding optimal thresholds for which these performance improvements are obtained are shown in the line below (row 3). Hereby, the granularity of 0.05 in the selection of the threshold is a trade-off between sufficient accuracy and acceptable computation time. In the bottom line the added value of using the threshold is shown. This is the part (in percentage) of the performance improvement of the T-XAS algorithm that can be attributed to the incorporation of the threshold mechanism. The total improvement the T-XAS algorithm can obtain is indeed the sum of the improvement by using the XAS algorithm (row 1 in table 6) and the improvement by additionally using a threshold mechanism (row 2 - row 1 in table 6). The ratio of these 2 values is thus a measure of added value of the threshold mechanism.

It was already clear from table 2 that as the load increases, the obtainable improvement of XAS vs. VAS also goes up. Table 6 shows that as the obtainable improvement of XAS vs. VAS increases, the added value of using an additional threshold extension decreases. In the case of an unrestricted buffer setting extending the XAS algorithm with a threshold is thus only beneficial in those cases where the XAS algorithm (without a threshold) can only offer a limited performance improvement in packet delay. As the added value of using an threshold mechanism decreases with an increasing load, the optimal threshold

Table 7: Percentage-wise performance improvement in LP of the WAS and T-WAS algorithms relative to VAS for different load values in various restricted buffer settings.

WAS vs. VAS		*# loops*				*# loops*			
		4	*8*	*16*	*∞*	*4*	*8*	*16*	*∞*
		load = 0.60				***load = 0.70***			
max recirculations	*4*	17.3	27.3	27.7	27.4	11.3	21.4	22.0	21.9
	8	7.0	23.5	23.4	23.7	4.3	16.6	18.2	18.1
	16	4.1	13.7	-0.7	-1.3	1.7	6.4	-1.6	-3.6
	∞	3.8	21.2	53.6	NaN	1.8	4.8	3.4	NaN
		load = 0.80				***load = 0.90***			
max recirculations	*4*	7.5	17.7	18.7	18.6	5.2	15.0	16.6	16.5
	8	2.5	12.6	15.8	15.9	1.3	9.8	15.6	15.7
	16	0.5	1.6	2.3	-0.3	-0.6	0.3	6.7	5.5
	∞	0.2	-4.4	-39.2	NaN	-0.6	-6.9	-32.7	NaN
T-WAS vs. VAS		*# loops*				*# loops*			
		4	*8*	*16*	*∞*	*4*	*8*	*16*	*∞*
		load = 0.60				***load = 0.70***			
max recirculations	*4*	17.3	27.3	27.7	27.4	11.3	21.4	22.0	21.9
	8	8.7	23.5	23.5	23.7	5.9	16.9	18.7	18.8
	16	6.5	18.9	8.5	7.4	4.2	10.4	8.3	6.3
	∞	6.1	25.3	64.0	NaN	4.2	10.6	27.2	NaN
		load = 0.80				***load = 0.90***			
max recirculations	*4*	7.8	17.7	18.7	18.6	5.5	15.0	16.6	16.5
	8	3.9	13.4	16.2	16.3	2.5	10.6	16.0	15.9
	16	2.5	5.0	9.7	8.4	1.4	2.9	11.0	11.3
	∞	2.4	2.0	4.9	NaN	1.3	0.2	0.3	NaN

Table 8: Percentage-wise performance improvement in LPsize of the XAS and T-XAS algorithms relative to VAS for different load values in various restricted buffer settings.

XAS vs. VAS		*# loops*				*# loops*			
		4	*8*	*16*	*∞*	*4*	*8*	*16*	*∞*
		load = 0.60				***load = 0.70***			
max recirculations	*4*	17.5	30.1	30.6	30.6	11.4	22.3	23.2	23.0
	8	2.9	31.6	37.9	37.6	2.5	22.0	29.7	30.1
	16	-3.1	8.3	41.0	40.9	-1.8	8.5	35.5	36.7
	∞	-3.8	-4.3	-13.9	NaN	-1.9	2.0	8.0	NaN
		load = 0.80				***load = 0.90***			
max recirculations	*4*	7.1	16.2	17.3	17.4	4.6	11.8	13.0	13.2
	8	2.1	15.0	22.5	23.0	1.3	10.1	16.9	17.4
	16	-0.7	7.8	26.3	29.4	-0.4	5.9	18.2	21.8
	∞	-1.1	3.2	13.4	NaN	-0.5	3.3	10.5	NaN
T-XAS vs. VAS		*# loops*				*# loops*			
		4	*8*	*16*	*∞*	*4*	*8*	*16*	*∞*
		load = 0.60				***load = 0.70***			
max recirculations	*4*	18.7	30.1	30.6	30.6	12.8	22.3	23.2	23.0
	8	7.4	32.6	37.9	37.6	6.7	23.1	29.9	30.1
	16	3.5	18.2	42.3	42.8	3.5	15.6	36.5	37.1
	∞	2.7	12.5	30.4	NaN	3.4	12.5	29.9	NaN
		load = 0.80				***load = 0.90***			
max recirculations	*4*	8.9	16.4	17.3	17.4	6.1	11.9	13.0	13.2
	8	5.4	16.7	22.8	23.0	3.9	11.5	16.9	17.4
	16	3.4	12.4	27.9	29.5	2.8	9.2	18.7	21.8
	∞	3.2	10.3	23.7	NaN	2.7	7.3	15.7	NaN

value increases. Indeed, as the threshold increases, the T-XAS algorithm increasingly approximates the XAS algorithm's scheduling behaviour and as such its performance.

Similar to table 6, table 7 and 8 extend the results of table 4 and table 5. In this, we focus on improving LP for the T-WAS algorithm and LPsize for the T-XAS algorithm. In table 7 the improvement in LP of the WAS and T-WAS algorithms is compared. It is clear that the T-WAS algorithm can always outperform the WAS algorithm. This comes naturally as the T-WAS algorithm performs equally to the WAS algorithm when the threshold parameter is set to 1. Optimizing this threshold thus automatically results in a performance at least as good as the WAS algorithm. The same observation can be made in table 8 showing the improvement in LPsize of the XAS and T-XAS algorithms.

From table 4 and table 5 it was already clear that in general, but not always, performance improvement increases for lower load values, a lower number of maximum recirculations, and a higher number of loops. Similar to the trend in table 6, table 7 show that in general as the obtainable improvement of WAS vs. VAS or XAS vs. VAS increases, the added value of using an additional threshold extension decreases.

Table 9 and 10 show the corresponding optimal thresholds for which the optimal performance improvements of the threshold algorithms in table 7 and 8 are obtained. The added value of using the threshold, i.e. the part (in percentage) of the performance improvement that can be attributed to the incorporation of the threshold mechanism, is also shown in these tables. As WAS and XAS are unable to outperform VAS for some parameter combinations, the added value of using the threshold is greater than 100% in these cases.

For the T-XAS algorithm the optimal threshold value stays fairly constant, i.e. in the range between 0.85 and 1. This is as opposed to the T-WAS algo-

Table 9: Optimal threshold and added value of the threshold mechanism of the T-WAS algorithm from table 7 for different load values in various restricted buffer settings.

optimal threshold T-WAS		# loops				# loops			
		4	8	16	∞	4	8	16	∞
		load = 0.60				load = 0.70			
max recirculations	4	1	1	1	1	0.95	1	1	1
	8	0.90	1	0.95	1	0.90	0.95	0.95	0.95
	16	0.90	0.90	0.75	0.75	0.85	0.90	0.55	0.60
	∞	0.90	0.90	0.90	NaN	0.90	0.90	0.80	NaN
		load = 0.80				load = 0.90			
max recirculations	4	0.95	1	1	1	0.90	1	1	1
	8	0.90	0.95	0.95	0.95	0.85	0.95	0.95	0.95
	16	0.90	0.90	0.55	0.55	0.90	0.90	0.75	0.60
	∞	0.90	0.75	0.45	NaN	0.90	0.30	0.25	NaN
addded value threshold (%)		# loops				# loops			
		4	8	16	∞	4	8	16	∞
		load = 0.60				load = 0.70			
max recirculations	4	0	0	0	0	1	0	0	0
	8	21	0	1	0	27	2	3	4
	16	37	28	109	119	60	38	119	157
	∞	39	16	16	NaN	57	56	88	NaN
		load = 0.80				load = 0.90			
max recirculations	4	4	0	0	0	5	0	0	0
	8	33	7	2	2	48	7	3	1
	16	84	68	76	105	143	86	39	51
	∞	92	320	898	NaN	146	3500	11033	NaN

Table 10: Optimal threshold and added value of the threshold mechanism of the T-XAS algorithm from table 8 for different load values in various restricted buffer settings.

optimal threshold T-XAS		# loops				# loops			
		4	*8*	*16*	∞	*4*	*8*	*16*	∞
		load = 0.60				***load = 0.70***			
max recirculations	*4*	0.95	1	1	1	0.90	1	1	1
	8	0.85	0.95	1	1	0.85	0.95	0.95	1
	16	0.85	0.90	0.95	0.95	0.85	0.90	0.95	0.95
	∞	0.8	0.85	0.85	NaN	0.85	0.85	0.90	NaN
		load = 0.80				***load = 0.90***			
max recirculations	*4*	0.90	0.95	1	1	0.90	0.95	1	1
	8	0.90	0.95	0.95	1	0.85	0.95	1	1
	16	0.85	0.90	0.95	0.95	0.85	0.90	0.95	1
	∞	0.85	0.90	0.90	NaN	0.85	0.90	0.95	NaN
addded value threshold (%)		# loops				# loops			
		4	*8*	*16*	∞	*4*	*8*	*16*	∞
		load = 0.60				***load = 0.70***			
max recirculations	*4*	6	0	0	0	11	0	0	0
	8	61	3	0	0	63	5	1	0
	16	189	54	3	4	151	46	3	1
	∞	241	134	146	NaN	156	84	73	NaN
		load = 0.80				***load = 0.90***			
max recirculations	*4*	20	1	0	0	25	1	0	0
	8	61	10	1	0	67	12	0	0
	16	121	37	6	0	114	36	3	0
	∞	134	69	43	NaN	119	55	33	NaN

rithm for which case the optimal thresholds can be much lower reaching values of 0.25. However, these low values are clearly outliers and are linked to those parameter combinations in which WAS greatly falls behind on VAS in terms of performance. In general, XAS underperforms VAS less often and mostly in a less severe way than WAS.

Despite some high numbers in added value, the overall improvement of the threshold algorithm compared to VAS can still be limited in some cases. For example, this is the case for the T-WAS algorithm for a load of 0.90, 16 loops and an infinite number of recirculations. In this case the added value reaches 11033% but the overall performance improvement of the T-WAS algorithm is merely 0.3% as the WAS algorithm falls behind the performance of the VAS algorithm by 32.7%. The corresponding optimal threshold is an outlier of 0.25, highlighting that for this parameter combinations the optimal algorithm greatly approximates the behaviour of the VAS algorithm. In fact, given the minimal overall performance improvement it is advisable in such cases to use the VAS algorithm and not the more complicated T-WAS algorithm.

To avoid a possible misinterpretation of tables 9 and 10 we therefore underlined those combinations in the added value parts for which the performance improvement of the threshold algorithm compared to VAS is at least 5% and the added value of the threshold is at least 20%. This means that for those parameter combinations, the performance improvement of the threshold algorithm is increased by at least 1% because of the threshold mechanism. Looking at those underlined values we can see that, in general, the best settings to apply a threshold mechanism are those with a high number of recirculations. In other settings either WAS or XAS (depending on the performance measure), or VAS are, while not necessarily better, preferred for their lower complexity.

5. Conclusions

In this paper we propose a threshold extension to the WAS and XAS scheduling algorithms for optical fiber loop buffers in a variable sized packets setting. This threshold is introduced to improve the performance of these algorithms even further and for a wider parameter range. By optimizing this threshold, the process of selectively delaying packet longer than strictly necessary can be made more or less strict and as such be fitted to each setting. Performance was evaluated by means of Monte Carlo simulation and showed that in the case of an unrestricted buffer setting extending the XAS algorithm with a threshold is only beneficial in those cases where the XAS algorithm (without a threshold) can only offer a limited performance improvement in packet delay. This corresponds with low values of the load and results in performance improvements that are up to 5 times as large as those without a threshold parameter. As the added value of using a threshold mechanism decreases with an increasing load, the optimal T-XAS algorithm increasingly approximates the XAS algorithms scheduling behaviour and as such its performance. For the restricted buffer setting a similar trend can be seen, i.e. the threshold extension adds most value for those instances where the algorithms without threshold can offer no or only limited performance improvement. While the algorithms without threshold underperform VAS strongly for some parameter combinations, the overall improvement of the threshold algorithms compared to VAS, although always positive, can still be severely limited. The settings for which the overall performance improvement of the threshold algorithms is at least 5% and the added value of the threshold is at least 20%, in general, correspond with those parameter combinations in which the maximum number of recirculations is high.

A future topic for performance evaluation is to compare algorithm throughputs at predetermined values of loss probability, on the one hand, and LPsize, on the other hand. An open research question remains what further performance improvement would be enabled by considering combinations with three

or four packets rather than only pairs, in order to further improve the key performance metrics (Loss probability, LPsize). Hereby, a key question is how such algorithms could be devised so as to be feasible in terms of computational and implementation complexity. Another topic for future work is to assess the impact of bursty arrivals on the proposed algorithms, either with specific assumptions on the correlation between arrival instants (with correlated arrivals, e.g., MMPP or BMAP) or an inhomogeneous arrival intensity profile (e.g., the inhomogeneous Poisson process, or a traffic trace).

References

[1] Cisco press release, The zettabyte era: trends and analysis (June 2017). URL https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/vni-hyperconnectivity-wp.html

[2] Nippon Telegraph and Telephone Corporation, One petabit per second fiber transmission over a record distance of 200 km (March 2017). URL https://www.ntt.co.jp/news2017/1703e/pdf/170323a.pdf

[3] Technical University of Munich Press Release, New record data transfer speed in fiber optic network (February 2019). URL https://www.technologist.eu/new-record-data-transfer-\speed-in-fiber-optic-network/

[4] S. Verma, H. Chaskar, R. Ravikanth, Optical burst switching: a viable solution for terabit ip backbone, IEEE Network 14 (6) (2000) 48{53. doi:10.1109/65.885670.

[5] Y. Chen, C. Qiao, X. Yu, Optical burst switching: a new area in optical networking research, IEEE Network 18 (3) (2004) 16{23. doi:10.1109/MNET.2004.1301018.

[6] Y. Xiong, M. Vandenhoute, H. C. Cankaya, Control architecture in optical burst-switched wdm networks, IEEE Journal on Selected Areas in Communications 18 (10) (2000) 1838{1851. doi:10.1109/49.887906.

[7] T. S. El-Bawab, J.-D. Shin, Optical packet switching in core networks: between vision and reality, IEEE Communications Magazine 40 (9) (2002) 60{65. doi:10.1109/MCOM.2002.1031830.

[8] I. Szczesniak, Overview of optical packet switching, Theoretical and Applied Informatics 21 (3-4) (2009) 167{180.

[9] A. Triki, A. Gravey, P. Gravey, M. Morvan, Long-term CAPEX evolution for slotted optical packet switching in a metropolitan network, in: Proceedings of International Conference on Optical Network Design and Modeling (ONDM), 2017, pp. 1{6.

[10] B. Mukherjee, Architecture, control, and management of optical switching networks, in: Proceedings of Photonics in Switching, 2007, pp. 43{44. doi:10.1109/PS.2007.4300735.

[11] W. V. Heddeghem, B. Lannoo, D. Colle, M. Pickavet, F. Musumeci, A. Pattavina, F. Idzikowski, Power consumption evaluation of circuit-switched versus packet-switched optical backbone networks, in: Proceedings of 2013 IEEE Online Conference on Green Communications, 2013, pp. 56{63. doi:10.1109/OnlineGreenCom.2013.6731029.

[12] S. Yao, B. Mukherjee, S. J. B. Yoo, S. Dixit, A unified study of contention-resolution schemes in optical packet-switched networks, Journal of Lightwave Technology 21 (3) (2003) 672{683. doi:10.1109/JLT.2003.809573.

[13] M. Yoo, C. Qiao, S. Dixit, The effect of limited fiber delay lines on QoS performance of optical burst switched WDM networks, in: Proceedings of 2000 IEEE International Conference on Communications, Vol. 2, 2000, pp. 974{979.

[14] T. Tanemura, I. M. Soganci, T. Oyama, T. Ohyama, S. Mino, K. A. Williams, N. Calabretta, H. J. S. Dorren, Y. Nakano, Large-capacity compact optical buffer based on InP integrated phased-array switch and coiled fiber delay lines, Journal of Lightwave Technology 29 (4) (2011) 396{402. doi:10.1109/JLT.2010.2102338.

[15] E. Burmeister, D. Blumenthal, J. Bowers, A comparison of optical buffering technologies, Optical Switching and Networking 5 (1) (2008) 10 { 18. doi: https://doi.org/10.1016/j.osn.2007.07.001.

[16] R. Langenhorst, M. Eiselt, W. Pieper, G. Grosskopf, R. Ludwig, L. Kuller, E. Dietrich, H. G. Weber, Fiber loop optical buffer, Journal of Lightwave Technology 14 (3) (1996) 324{335. doi:10.1109/50.485589.

[17] A. Liu, C. Wu, M. Lim, Y. Gong, P. Shum, Optical buffer configuration based on a 3x3 collinear fibre coupler, Electronics Letters 40 (2004) 1017 { 1019.

[18] S. Fu, P. Shum, N. Q. Ngo, C.Wu, Y. Li, C. Chan, An enhanced SOA-based double-loop optical buffer for storage of variable-length packet, Journal of Lightwave Technology 26 (4) (2008) 425{431.

[19] C.-y. Tian, C.-q. Wu, G.-n. Sun, X. Li, Z.-y. Li, Quality improvement of the dual-wavelength signals in DLOB via power equalization, Optoelectronics Letters 4 (5) (2008) 361{364.

[20] Y.Wang, C.Wu, Z.Wang, X. Xin, A new large variable delay optical buffer based on cascaded double loop optical buffers (DLOBs), in: Proceedings of 2009 Conference on Optical Fiber Communication, 2009, pp. 1{3.

[21] K. K. Merchant, J. Mcgeehan, A. E. Willner, S. Ovadia, P. Kamath, J. D. Touch, J. Bannister, Analysis of an optical burst switching router with tunable multiwavelength recirculating buffers, Lightwave Technology, Journal of 23 (2005) 3302 { 3312. doi:10.1109/JLT.2005.855687.

[22] C. Develder, B. Van Houdt, C. Blondia, M. Pickavet, P. Demeester, Analytical mmap-based bounds for packet loss in optical packet switching with recirculating fdl buffers, Photonic Network Communications 8 (2004) 149{161. doi:10.1023/B:PNET.0000033975.88406.28.

[23] S. H. C. A. F. Meow Chiow Chia, Vineetha Kalavally, Buffering and scheduling of asynchronous variable-length packets (2005). doi:10.1117/12.573437.

URL https://doi.org/10.1117/12.573437

[24] N. Akar, Y. Gunalay, Dimensioning shared-per-node recirculating fiber delay line buffers in an optical packet switch, Performance Evaluation 70 (2013) 10591071. doi:10.1016/j.peva.2013.09.003.

[25] J. Wang, C. McArdle, L. Barry, Energy-efficient optical packet switch with recirculating fiber delay line buffers for data center interconnects, 2014. doi:10.1109/ICTON.2014.6876546.

[26] R. Geldenhuys, Z. Wang, N. Chi, I. Tafur Monroy, A. M. J. Koonen, H. J. S. Dorren, F. W. Leuschner, G. D. Khoe, S. Yu, Timeslot interchanging using the crosspoint switch and a recirculating buffer 48 (2006) 897 { 900.

[27] M. Kumar Dutta, V. Chaubey, Modeling and performance analysis of optical packet switching network using fiber delay lines, in: Proceedings - 2011 Annual IEEE India Conference: Engineering Sustainable Solutions, INDICON-2011, 2011, pp. 1-4. doi:10.1109/INDCON.2011.6139451.

[28] W. Rogiest, D. Fiems, J.-P. Dorsman, Analysis of fibre-loop optical buffers with a void-avoiding schedule, in: Proceedings of Valuetools 2014, 2014, pp. 1-7.

[29] A. Rostami, S. S. Chakraborty, On performance of optical buffers with specific number of circulations, IEEE Photonics Technology Letters 17 (7) (2005) 1570-1572. doi:10.1109/LPT.2005.848548.

[30] K. Van Hautegem, M. Pinto, H. Bruneel, W. Rogiest, Analysis of VAS, WAS and XAS scheduling algorithms for fiber-loop optical buffers, in: Queueing Theory and Network Applications, Springer International Publishing, 2018, pp. 216{226.

[31] K. Van Hautegem, W. Rogiest, H. Bruneel, Scheduling in optical switching: Deploying shared wavelength converters more effectively, in: Proceedings of 2014 IEEE International Conference on Communications (ICC), 2014, pp. 3418{3424. doi:10.1109/ICC.2014.6883850.

[32] K. Van Hautegem, W. Rogiest, H. Bruneel, Optical switching for variable size packets: Improved void filling through selective void creation, in: Proceedings of 11th International Conference on Queueing Theory and Network Applications (QTNA), 2016, pp. 1-8.

[33] K. Van Hautegem, W. Rogiest, H. Bruneel, Improving performance and energy consumption of shared wavelength converters in OPS/OBS, Optical Switching and Networking 17 (2015) 38 - 51. doi:https://doi.org/10.1016/j.osn.2014.10.003.

[34] J. Lambert, B. Van Houdt, C. Blondia, Single-wavelength optical buffers: non-equidistant structures and preventive drop mechanisms, in: Proceedings of Networking and Electronic Commerce Research Conference (NAEC), 2005, pp. 545-555.

Industria y Comercio
SUPERINTENDENCIA

El futuro es de todos

BANDEJA DE ENTRADA >

NC2021/0008073 - Patente de Invención Nacional - MÉTODO Y SISTEMA PARA REDUCIR TIEMPOS DE PROCESAMIENTO DE SOLICITUDES EN LÍNEAS DE ESPERA

Salir

Datos de la solicitud

Referencia del solicitante	P2021/000124	Fecha de radicación	18 jun. 2021
Número de patente	NC2021/0008073		
Estado	Bajo Verificación de Requisitos Mínimos		

Contacto

Apoderado

Número de identificación	Nombre(s)	Apellido(s)	Dirección (es)
79782747	CARLOS REINALDO	OLARTE GARCIA	Dirección Física : Carrera 5 N° 34-03 BOGOTÁ D.C. (CO)

Solicitante(s)

Número de identificación	Nombre(s)	Apellido(s)	Dirección (es)
860056070		Universidad Antonio Nariño	Dirección Física : Carrera 3 Este 47A15 BOGOTÁ D.C. (CO)

Contacto de la solicitud

Número de identificación	Nombre	Dirección	Ciudad	Código postal	País	Tipo de dirección
79782747	CARLOS REINALDO OLARTE GARCIA	Carrera 5 N° 34-03	BOGOTÁ		CO	Dirección Física

Información de la Patente

Cerrar

Tipo de Patente	Patente de Invención Nacional
Solicitud vía PPH	☐

Inventor(a)

Número de identificación	Nombre(s)	Apellido(s)	Dirección (es)
19439502	RAFAEL MARÍA	GUTIÉRREZ SALAMANCA	Dirección Física : Carrera 3 Este No.47A-15, Bloque 4, Piso 3 BOGOTA CUNDINAMARCA (CO)
80417160	Andres Ignacio	Hernández Duarte	Dirección Física : Carrera 3 Este No. 47 A-15, BOGOTA CUNDINAMARCA (CO)
7714048	Mario Augusto	PINTO SERRANO	Dirección Física : Carrera 42 No 18 A - 05 NEIVA HUILA (CO)

Cesión	
Título	MÉTODO Y SISTEMA PARA REDUCIR TIEMPOS DE PROCESAMIENTO DE SOLICITUDES EN LÍNEAS DE ESPERA
Título inicial	MÉTODO Y SISTEMA PARA REDUCIR TIEMPOS DE PROCESAMIENTO DE SOLICITUDES EN LÍNEAS DE ESPERA
Tiempo para la publicación	18

Prioridad

Prioridad	

Documentos

Descripción

1 Documento (s)

Documento	Tipo	Confidencialidad
Descripción	Descripción	Públicos

Resumen

La presente divulgación se relaciona con métodos y sistemas para reducir tiempos de procesamiento de solicitudes en líneas de espera en servicios de alta complejidad. Por ejemplo, la presente divulgación describe un método que puede incluir una etapa de recibir en una unidad de cómputo un dato de objeto que puede incluir un dato de identificación y un dato de solicitud de servicio, y una etapa de obtener un dato de prioridad para el dato de objeto ejecutando un proceso de asignación de prioridad que puede tomar como entrada el dato de identificación y puede consultar un conjunto de reglas de prioridad. Adicionalmente, el método puede almacenar el dato de objeto en un primer registro de una estructura de datos de prioridad, seleccionar un primer módulo de procesamiento de solicitudes entre al menos dos módulos de solicitudes para procesar el dato de solicitud de se

Cerrar

ejecutar un proceso de selección de módulo en la unidad de cómputo. También, el método puede enviar desde la unidad de cómputo el dato de solicitud de servicio del dato de objeto que puede tener el mayor valor del dato de prioridad, al primer módulo de procesamiento de solicitudes. El método puede obtener mediante la unidad de cómputo un conjunto de variables de rendimiento en el módulo de memoria, generadas mientras el primer módulo de procesamiento de solicitudes procesa el dato de solicitud de servicio. También, el método puede incluir una etapa de obtener un dato de eficiencia del procesamiento de la solicitud de servicio, ejecutando un proceso de calificación. Adicionalmente, el método puede incluir una etapa de modificar mediante la unidad de cómputo el conjunto de reglas de prioridad que pueden estar almacenadas en el módulo de memoria a partir del dato de eficiencia.

Además, la presente divulgación también describe modalidades de un sistema que puede incluir un módulo de memoria y una unidad de cómputo que puede estar conectada al módulo de memoria y configurada para ejecutar uno o más de las modalidades de los métodos anteriormente descritos.

Reivindicaciones	
Número de Reivindicaciones	Recuerde que: -El número de Reivindicaciones diligenciado deberá estar coherente con el contenido del documento anexo -A partir de la undécima Reivindicación, deberá presentar pago por cada una de ellas para que éstas sean tenidas en cuenta 13
Número de folios	0
Declaraciones	Declaración sobre el uso de Recursos Genéticos o Biológicos Declaro que el objeto de la presente solicitud de patente fue obtenido a partir de recursos genéticos o biológicos de los que cualquiera de los países miembros de la Comunidad Andina es país de origen. ○ Si ◉ No Nota: En caso afirmativo deberá anexar copia del contrato de acceso de recursos genéticos o productos derivados, o certificado o número de registro, expedido por la Autoridad competente. Declaración sobre uso de Conocimientos Tradicionales Declaro que el objeto de la presente solicitud de patente fue obtenido a partir de conocimientos tradicionales de comunidades indígenas, afroamericanas o locales de países miembros de la Comunidad Andina. ○ Si ◉ No Nota: En caso afirmativo deberá anexar la licencia o autorización de uso de conocimiento tradicional, o certificado, o número de registro expedido por la Autoridad competente. Declaración de documentos presentados durante el año de gracia Declaro que la invención fue divulgada por parte del inventor o causahabiente durante el año anterior a la fecha de presentación y/o prioridad de la presente solicitud de acuerdo con el Art. 17 de la Decisión 486. ○ Si ◉ No Nota: En caso afirmativo deberá anexar el documento que divulga el objeto de la invención durante el año de gracia.

1 Histórico (s)

Tipo de actuación	Descripción	Fecha de creación	Gaceta	Fecha de publicación	
Entrada solicitud de patente.	Ha ingresado solicitud de	18 jun. 2021 12:00:17		19 jun. 2021	Cerrar

Ingresa	patente	a.m.

Obtener Reporte PDF

Cerrar

MÉTODO Y SISTEMA PARA REDUCIR TIEMPOS DE PROCESAMIENTO DE SOLICITUDES EN LÍNEAS DE ESPERA

CAMPO TÉCNICO

La presente invención se relaciona con un método y sistema para reducir tiempos de procesamiento de solicitudes de atención en líneas de espera en servicios de alta complejidad operativa.

DESCRIPCIÓN DEL ESTADO DE LA TÉCNICA

En la actualidad, existen algoritmos de programación de procesos adoptados por los sistemas modernos para asignar prioridades a los procesos. El propósito principal de tales algoritmos de programación de procesos es reducir la falta de recursos, garantizar la equidad entre las partes que utilizan los recursos y reducir los tiempos de ejecución de los procesos. Para lograr este objetivo, se realizan programaciones en los sistemas para modificar dinámicamente la prioridad real de los procesos en su cola de programación interna y reducir los tiempos de espera de los procesos.

Por lo tanto, en el estado de la técnica se identifican documentos relacionados a la reducción de tiempos de espera, por ejemplo, US 1,049,644 B2, US 8,886,899 B1 y US 9,009,717 B2.

El documento US 1,049,644 B2 divulga sistemas y métodos para el manejo de operaciones y asignaciones de recursos asociadas a dichas operaciones.

El método divulgado en US 1,049,644 B2 gestiona la asignación de recursos en un sistema de almacenamiento de datos que tiene recursos informáticos y comprende una etapa de recibir una primera solicitud y una etapa de recibir una segunda solicitud. La primera solicitud es una solicitud de asignación de una porción de los recursos informáticos del sistema de almacenamiento de datos a un primer trabajo de protección de datos. La segunda solicitud es una solicitud de asignación de una porción de los recursos informáticos del sistema de almacenamiento de datos a un segundo trabajo de protección de datos diferente del primer trabajo de protección de datos.

Además, el método de US 1,049,644 B2 incluye una etapa de bloquear uno o más objetos de recursos asociados a los recursos informáticos del sistema de almacenamiento de datos. Y posterior a esta, una etapa de asignar una porción de los recursos informáticos del sistema de almacenamiento de datos al primer trabajo de protección de datos y otra parte de los recursos informáticos del sistema de almacenamiento de datos al segundo trabajo de

protección de datos, si uno o más objetos de recursos están bloqueados. Posterior a la asignación, el método de US 1,049,644 B2 desbloquea el o los objetos de recursos asociados con los recursos informáticos del sistema de almacenamiento de datos.

Por su parte, el documento US 8,886,899 B1 divulga métodos para gestionar solicitudes de memoria según su prioridad y gestionar procesos, acceso a información y comunicación en un ambiente de procesamiento en paralelo. En particular, US 8,886,899 B1 divulga un método para gestionar el acceso a una memoria externa en un sistema informático que comprende uno o más núcleos.

El método divulgado en US 8,886,899 B1 gestiona el acceso a una memoria externa en un sistema informático que comprende uno o más núcleos mediante una etapa de recibir solicitudes de acceso a una memoria en un controlador de memoria, el cual está acoplado a uno o más núcleos de un procesador, una etapa de asignar mediante el controlador de memoria prioridades a solicitudes de memoria de la etapa anterior, también incluye una etapa de proporcionar acceso a la memoria mediante el controlador de memoria de acuerdo con las prioridades asignadas a las solicitudes de acceso, y finalmente una etapa de recibir mensajes en el controlador de memoria para modificar la información de configuración de prioridades.

Asimismo, US 8,886,899 B1 divulga que el controlador de memoria está conectado a al menos uno de los núcleos del procesador, que las prioridades están basadas en la información de configuración de prioridades, y que dicha información de configuración de prioridades está almacenada en un registro del controlador de memoria y pueden ser modificadas mediante la ejecución de instrucciones especiales de acceso a la memoria emitidas por cualquiera de los núcleos del sistema informático. La información de configuración de prioridades incluye una lista de control de prioridades con varias entradas, cada entrada con una expresión de un conjunto de atributos asociados con la solicitud de memoria que se utiliza para asignar un nivel de prioridad predeterminado.

Por otro lado, el documento US 9,009,717 B2 divulga un método para controlar y adaptar dinámicamente la prioridad de un árbol de procesos. El método divulgado por US 9,009,717 B2 gestiona procesos informáticos que se ejecutan en un sistema informático que comprende una etapa de seleccionar un conjunto de procesos y asignar al conjunto de procesos una clase de prioridad de proceso. El conjunto de procesos son procesos informáticos que se ejecutan en una unidad de cómputo. Las clases de prioridad de procesos están agrupadas según un tipo de consumo de recursos e incluyen unas reglas de programación de procesos, las cuales especifican cómo determinar la prioridad del proceso.

Asimismo, el método divulgado en US 9,009,717 B2 incluye una etapa de iniciar la monitorización del conjunto de procesos seleccionados; y una serie de subetapas que se ejecutan sobre cada proceso monitorizado e incluyen: recopilar un conjunto de métricas, calcular la media de las métricas, leer una regla de programación de procesos, determinar una nueva prioridad para el proceso y cambiar la prioridad actual del proceso supervisado. El conjunto de métricas incluye métricas de uso de CPU y métricas de tiempo de espera de E/S para el proceso monitorizado. La regla de programación de procesos está dada por la clase de prioridad asignada al proceso y especifica cómo determinar la prioridad del proceso en función de la media de la métrica de uso de la CPU y la media de la métrica de tiempo de espera de E/S. La nueva prioridad del proceso supervisado está determinada por la aplicación de la regla de programación de procesos a la media de las métricas.

Por lo tanto, el estado de la técnica divulga métodos y sistemas para la gestión de solicitudes, asignación de recursos asociados a dichas solicitudes, gestión de procesos, monitoreo de métricas, así como la determinación y asignación de prioridades asociadas a una solicitud.

BREVE DESCRIPCIÓN

La presente divulgación se relaciona con métodos y sistemas para reducir tiempos de procesamiento de solicitudes en líneas de espera en servicios de alta complejidad. Particularmente, cualquiera de las modalidades aquí descritas permite reducir tiempos de espera y agilizar el procesamiento de los datos en comparación con otros procesos convencionales de gestión de solicitudes.

Por ejemplo, la presente divulgación describe un método que permite a un sistema procesar datos de manera tal, que cuando la solicitud se refiere a una pluralidad de servicios solicitados por una pluralidad de humanos o uno o más entes computacionales, el procesamiento de los datos se realiza de manera más eficiente.

En una primera modalidad, el método puede incluir una etapa de recibir en una unidad de cómputo un dato de objeto que puede incluir un dato de identificación y un dato de solicitud de servicio. Además, el método puede incluir una etapa de obtener un dato de prioridad para el dato de objeto ejecutando con la unidad de cómputo un proceso de asignación de prioridad que puede tomar como entrada el dato de identificación y puede consultar un conjunto de reglas de prioridad, que la unidad de cómputo puede consultar en un módulo de memoria.

Adicionalmente, el método puede incluir una etapa de almacenar mediante la unidad de cómputo el dato de objeto en un primer registro de una estructura de datos de prioridad que

puede estar almacenada en el módulo de memoria. La estructura de datos de prioridad puede tener una pluralidad de registros ordenados de acuerdo con una jerarquía que puede determinarse con base en el dato de prioridad del dato de objeto. Cada registro de estructura de datos de prioridad puede estar configurado para almacenar un dato de objeto.

El método puede incluir una etapa de seleccionar un primer módulo de procesamiento de solicitudes entre al menos dos módulos de solicitudes para procesar el dato de solicitud de servicio al ejecutar un proceso de selección de módulo en la unidad de cómputo. También, el método puede incluir una etapa de enviar desde la unidad de cómputo el dato de solicitud de servicio del dato de objeto que puede tener el mayor valor del dato de prioridad, al primer módulo de procesamiento de solicitudes, que puede ser seleccionado en la etapa de seleccionar un primer módulo de procesamiento.

El método puede además incluir una etapa de obtener mediante la unidad de cómputo un conjunto de variables de rendimiento almacenadas en el módulo de memoria, generadas mientras el primer módulo de procesamiento de solicitudes procesa el dato de solicitud de servicio.

También, el método puede incluir una etapa de obtener un dato de eficiencia del procesamiento de la solicitud de servicio, ejecutando con la unidad de cómputo un proceso de calificación que puede tomar como entrada el conjunto de variables de rendimiento. Adicionalmente, el método puede incluir una etapa de modificar mediante la unidad de cómputo el conjunto de reglas de prioridad que pueden estar almacenadas en el módulo de memoria a partir del dato de eficiencia.

Particularmente, una de las ventajas del método divulgado es que las etapas de obtener un conjunto de variables de rendimiento y obtener un dato de eficiencia que pueden generarse mientras el primer módulo de procesamiento de solicitudes procesa el dato de solicitud de servicio permiten generar registros históricos que permitan una mejor selección de un módulo de solicitudes para atención de una solicitud de servicio y optimizar los tiempos netos de atención de la solicitud de servicio.

Adicionalmente, en el método divulgado la etapa de modificar el conjunto de reglas de prioridad permite estructurar las peticiones de manera que los tiempos promedios de procesamiento de solicitudes pueden reducirse en un 15 %

Otra ventaja de modificar el conjunto de reglas de prioridad es que permite al sistema una mejor utilización de los recursos, de tal manera que con la misma estructura de recursos se pueda realizar un procesamiento más eficiente de las solicitudes, aprovechando los recursos disponibles, de forma óptima.

Algunos ejemplos de lo anterior podrían ser la reducción de costos de procesamiento, establecer una función de costo o función objetivo que se puede optimizar (minimizar o maximizar de acuerdo con la función elegida), y que puede incluir variables como: costo operacional, beneficios obtenidos por procesamiento de solicitudes, percepción o valoración de un cliente de la calidad de un servicio, retención de clientes, evaluación continua del rendimiento del sistema, evaluación de las unidades de procesamiento, y otras variables de rendimiento similares o equivalentes. Adicionalmente, se podría realizar detección temprana de degradación o pérdida de eficiencia de las unidades de procesamiento e identificación de buenas prácticas en el procesamiento de los servicios entre otros ejemplos.

También, en el método divulgado la etapa de modificar el conjunto de reglas de prioridad podría realizarse de manera dinámica, ya sea de forma manual por un administrador del sistema o de forma automática por el sistema, con el fin de ajustar el método y mantener la tendencia a minimizar el tiempo de procesamiento de solicitudes.

En algunas modalidades del método, en las que las solicitudes procesadas involucran la atención de un humano, y la reducción de los tiempos de espera de una pluralidad de humanos, estas modalidades del método permiten incrementar la satisfacción de los clientes, y por ende las utilidades de las empresas prestadoras de servicio, teniendo en cuenta que, a mayor satisfacción, mayores ingresos, y la empresa se diferencia de la competencia, ganando nuevos clientes.

Además, la presente divulgación también describe modalidades de un sistema que puede incluir un módulo de memoria y una unidad de cómputo que puede estar conectada al módulo de memoria y configurada para ejecutar uno o más de las modalidades del método anteriormente descrito.

BREVE DESCRIPCIÓN DE LAS FIGURAS

La FIG. 1 muestra un diagrama de flujo de las etapas de una modalidad de un método para reducir tiempos de procesamiento de solicitudes en líneas de espera.

La FIG. 2 muestra un diagrama de bloques de una modalidad de una estructura de datos de prioridad la cual tiene una estructura vectorial.

La FIG. 3 muestra un diagrama de una modalidad de una estructura matricial con una pluralidad de vectores de procesamiento, donde cada vector de procesamiento esta asociado a un módulo de procesamiento de solicitudes.

La FIG. 4 muestra un diagrama de bloques de una modalidad del método para reducir tiempos de procesamiento de solicitudes en líneas de espera, donde el diagrama de bloques ilustra datos y procesos obtenidos y ejecutados en unas subetapas i1) a i4) de una etapa b) de dicha modalidad del método.

La FIG. 5 muestra un diagrama de bloques de una modalidad del método para reducir tiempos de procesamiento de solicitudes en líneas de espera, donde el diagrama de bloques ilustra datos y procesos obtenidos y ejecutados en unas subetapas j1) a j5) de una etapa d) de dicha modalidad del método.

La FIG. 6 muestra un diagrama de bloques de una modalidad del método para reducir tiempos de procesamiento de solicitudes en líneas de espera, donde el diagrama de bloques ilustra datos y procesos obtenidos y ejecutados en unos pasos k1) a k4) entre unos pasos j4) y j5) de dicha modalidad del método.

La FIG. 7 muestra un diagrama de bloques de una modalidad del método para reducir tiempos de procesamiento de solicitudes en líneas de espera, donde el diagrama de bloques ilustra datos y procesos obtenidos y ejecutados en unos subpasos v1) y v2) de un paso k5) de dicha modalidad del método.

La FIG. 8 muestra un diagrama de bloques de una modalidad del método para reducir tiempos de procesamiento de solicitudes en líneas de espera, donde el diagrama de bloques ilustra datos y procesos obtenidos y ejecutados en unos subpasos v3) y v4) de un paso k5) de dicha modalidad del método.

La FIG. 9 muestra un diagrama de bloques de una modalidad del método para reducir tiempos de procesamiento de solicitudes en líneas de espera, donde el diagrama de bloques ilustra datos y procesos obtenidos y ejecutados en un subpaso v5) de un paso k5) de dicha modalidad del método.

La FIG. 10 muestra un diagrama de bloques de una modalidad del método para reducir tiempos de procesamiento de solicitudes en líneas de espera, donde el diagrama de bloques ilustra datos y procesos obtenidos y ejecutados en unas subetapas g1 a g3 de la etapa g) de dicha modalidad del método.

La FIG. 11 muestra un diagrama de bloques de una modalidad del método para reducir tiempos de procesamiento de solicitudes en líneas de espera, donde el diagrama de bloques ilustra datos y procesos obtenidos y ejecutados en una etapa a1) previa a la etapa b) de dicha modalidad del método.

La FIG. 12 muestra un diagrama de bloques de una modalidad de un sistema configurado para ejecutar cualquiera de las modalidades del método aquí divulgado, donde el sistema incluye un módulo de memoria y una unidad de cómputo.

DESCRIPCIÓN DETALLADA

La presente divulgación se refiere a un método para reducir tiempos de procesamiento de solicitudes en líneas de espera en servicios de alta complejidad. Específicamente, la presente divulgación se relaciona con métodos implementados por computador para reducir tiempos de procesamiento de datos relacionados a conjunto de procesos o relacionados a múltiples solicitudes de servicios que pueden ser solicitadas a un conjunto de elementos o entidades que pueden encargarse de procesar dichas solicitudes.

Por ejemplo, haciendo referencia a la FIG. 1 y la FIG. 12 en la presente divulgación se describe un método para reducir el tiempo de procesamiento de solicitudes que comprende una etapa a) de recibir, por ejemplo, en una unidad de cómputo (103) un dato de objeto (100) que puede incluir un dato de identificación (101) y un dato de solicitud de servicio (102).

El dato de objeto (100) puede ser información relacionada al usuario del sistema, que lo caracteriza y permite diferenciarlo de otros usuarios. Particularmente, el dato de objeto (100) también puede ser información que caracteriza un elemento en un sistema de comunicación y lo hace único en dicho sistema.

También, el dato de identificación (101) puede relacionarse y no limitarse a un conjunto de símbolos, que permiten identificar a un usuario y que puede ser la combinación de caracteres numéricos, grafemas u otros símbolos, por ejemplo, número de cedula, número de pasaporte, licencia de conducir, entre otros. Por otro lado, el dato de solicitud de servicio (102) se relaciona a información de un tipo de servicio que requiere el usuario dentro de una gama de servicios posibles establecidos, por ejemplo, podría ser una transferencia bancaria en un banco, una solicitud de atención en un hospital o un procedimiento de envío de información en un sistema de comunicaciones.

El método puede incluir una etapa b) de obtener un dato de prioridad (104) para el dato de objeto (100) ejecutando con la unidad de cómputo (103) un proceso de asignación de prioridad (105) que puede tomar como entrada el dato de identificación (101) y puede consultar un conjunto de reglas de prioridad (106), que la unidad de cómputo (103) puede consultar en un módulo de memoria (107).

Debe entenderse en la presente divulgación que un proceso de asignación de prioridad (105) puede ser un procedimiento mediante el cual se puede establecer el orden de atención de un dato de solicitud de servicio (102), basándose en su importancia y en la urgencia con la que se requiere la atención. El orden de atención en el proceso de asignación de prioridad (105) puede ser dinámico, dado que, por ejemplo, un administrador del sistema puede modificar un conjunto de reglas de prioridad (106), dependiendo de las circunstancias. El resultado del proceso de asignación de prioridad (105) puede ser un dato de prioridad (104) que, además, puede ser utilizado en etapas posteriores como entrada para la ejecución de otro proceso, etapa o método de cualquiera de las modalidades del método acá divulgado.

Además, debe entenderse en la presente divulgación que el conjunto de reglas de prioridad (106) pueden ser y no limitarse a lineamientos que pueden establecer el orden de atención más apropiado de un dato de objeto (100) y para el cual puede ser asignado un dato de prioridad (104). El conjunto de reglas de prioridad (106) puede basarse en criterios que establezcan una jerarquía en la atención de un dato de objeto (100), por ejemplo, un dato de importancia (108) y/o un dato de urgencia (109), que pueden ser ponderados o determinados por un administrador del sistema. También, el conjunto de reglas de prioridad (106) puede basarse en registros históricos de desempeño de diferentes entes prestadores de servicio, por ejemplo, módulos de atención de solicitudes (114). Alternativamente, el conjunto de reglas de prioridad (106) pueden modificarse periódicamente o en tiempo real con base en dichos registros históricos de desempeño, o variables similares que permitan reducir el tiempo promedio que tarda en procesarse una solicitud.

El conjunto de reglas de prioridad (106) puede ser y no limitarse a expresiones matemáticas, funciones de costo o funciones objetivo, funciones de optimización, condicionales lógicos, funciones logísticas, y demás funciones, parámetros o conjuntos de datos que permitan determinar los niveles de prioridad en el sistema. También, el conjunto de reglas de prioridad (106) puede ser ingresado al sistema en forma de código, por ejemplo, un código numérico. Además, los códigos pueden ser modificados teniendo en cuenta, por ejemplo, registros históricos de los mismos que pueden estar contenidos en una de base de datos, o por ejemplo, los datos de solicitud de servicio (102) disponibles en el sistema, que además, pueden estar almacenados en el módulo de memoria (107).

Adicionalmente, el conjunto de reglas de prioridad (106) podrían modificarse de manera dinámica, ya sea de forma manual por un administrador del sistema o de forma automática por el sistema.

Por ejemplo, el conjunto de reglas de prioridad (106) podrían estar configuradas para permitir al sistema establecer una función de costo multicriterio que involucre los aspectos que un administrador del sistema quisiera optimizar, por ejemplo, la importancia de un cliente, en donde se podrían priorizar a clientes de una categoría VIP, en otro ejemplo, podría ser la urgencia del servicio que es solicitado, en donde un servicio crítico podría priorizarse con respecto a servicios de menor importancia (v.g., a partir de análisis tipo triage, como los usados en salas de urgencias de hospitales).

En otro ejemplo, también podría optimizarse el costo de operación, en cuyo caso podrían priorizarse servicios cuya espera suponga mayor aumento de costos operativos. Adicionalmente, cada uno de los aspectos a ser considerados dentro de la función de costo podrían ser ponderados de acuerdo a los criterios que establezca el administrador del sistema. También, la función de costo podría depender del orden de atención a las solicitudes y el orden de prioridad se podría asignar de modo que se minimice la función de costo.

En cualquiera de las modalidades del método en las que el conjunto de reglas de prioridad (106) usen o se obtengan mediante una función de costo o función objetivo, una o más reglas del conjunto de reglas de prioridad (106) puede determinarse a partir de procesos computacionales predictivos, aprendizaje automático y combinaciones de estos. Por ejemplo, las reglas del conjunto de reglas de prioridad (106) pueden ser datos de entradas en procesos de aprendizaje automático, o pueden ser hiperparámetros de los procesos de aprendizaje automático. Ejemplos de procesos de aprendizaje automático pueden ser procesos de regresión (v.g. lineal, logística, polinomial), procesos de árboles de decisión, procesos basados en redes neuronales, y otros procesos de aprendizaje automático supervisado y/o no-supervisados que sean conocidos por una persona medianamente versada en la materia.

Particularmente, cuando que el conjunto de reglas de prioridad (106) usen o se obtengan mediante una función de costo o función objetivo, dicha función de costo o función objetivo puede ser uno de los hiperparámetros del proceso de aprendizaje automático. Por ejemplo, la función de costo puede seleccionarse entre error medio absoluto, error cuadrático medio, entropía cruzada (v.g., categórica, binaria, "sparse-categorical").

En cualquiera de las modalidades del método y haciendo referencia a la FIG.4, el proceso de asignación de prioridad (105) de la etapa b) puede incluir una subetapa i1) de consultar mediante la unidad de cómputo (103) una regla de prioridad (122) que puede estar almacenada en el módulo de memoria (107).

La regla de prioridad (122) puede ser un criterio usado para determinar la prioridad de atención de un dato de objeto (100), y, además, puede permitir establecer el orden en una jerarquía (113). Dicha regla de prioridad (122) puede ser establecida por un administrador del sistema, y también, puede estar basada en el resultado de realizar una operación matemática de multiplicar el valor del dato de importancia (108) por el valor del dato de urgencia (109) relacionados al dato de solicitud de servicio (102), donde previamente se han asignado los valores de importancia y urgencia a cada solicitud de servicio (102) entre un conjunto de solicitudes de servicio (102) disponibles en el sistema.

Por ejemplo, la regla de prioridad (122) podría generarse por la minimización de una función de costo multicriterio que podría incluir aspectos que un administrador del sistema considere, por ejemplo, el dato de importancia (108) podría derivarse de la ponderación de criterios de importancia tales como la importancia de un cliente, costos de operación, optimización de recursos informáticos, entre otros. Adicionalmente, el dato de urgencia (109) podría obtenerse de la ponderación de criterios de urgencia tales como una ruta crítica dentro de un proceso, colapso o posibles fallas del sistema o solicitudes de servicios cuya no ejecución inmediata pudieran comprometer la integridad del sistema o de un cliente.

En cualquiera de las modalidades del método, el proceso de asignación de prioridad (105) de la etapa b) puede incluir también una subetapa i2) de consultar mediante la unidad de cómputo (103) un dato de importancia (108) asociado al dato de identificación (101) y un dato de urgencia (109) asociado al dato de solicitud de servicio (102) almacenados en el módulo de memoria (107).

El dato de importancia (108) puede incluir información asociada con el dato de identificación (101) asociado a un dato de objeto (100) y que puede estar relacionada con el valor que puede tener el dato de objeto (100) en el sistema, de manera que se pueda contar con un criterio para diseñar una estrategia para la atención y prestación de servicio (o proceso computacional) relacionado con el dato de objeto (100) al cual está asociado el dato de identificación (101).

Por ejemplo, para facilitar las operaciones que son ejecutadas mediante la unidad de cómputo (103), el dato de importancia (108) puede ser un número entero o también puede ser un número real, sin embargo, no se limita a dichos tipos de datos. En ejemplos particulares, el dato de importancia (108) podría ser un número entre 1 y 6 que, además, podría relacionarse a un número de una tarjeta de crédito de un banco categorizada como VIP, un número de membresía, un número de un programa de fidelidad o también, podría relacionarse a un número de compras acumuladas en un establecimiento en un periodo de tiempo definido y asociadas al dato de identificación (101). Alternativamente, el dato de

importancia (108) puede tener valores de variables categóricas que clasifiquen a un usuario o dispositivo cliente.

Además, en la presente divulgación un dato de urgencia (109) puede entenderse por la información asociada al dato de solicitud de servicio (102) que permite determinar la rapidez con la cual debe atenderse el dato de objeto (100) al cual está asociado el dato de solicitud de servicio (102), con el objetivo de evitar consecuencias negativas, riesgos o procesos innecesarios, por ejemplo, pueden ser criterios de urgencia tales como una ruta crítica de un proceso relacionado al dato de solicitud de servicio (102), pueden deberse a condiciones puntuales del sistema, como un colapso o compromiso del sistema, pueden estar relacionados a solicitudes de servicio (102) cuya no ejecución inmediata comprometan la integridad del sistema o del dato de objeto (100).

El dato de urgencia (109) puede predefinirse y modificarse en el sistema con base en la totalidad de solicitudes de servicio (102) disponibles en el sistema y dicho sistema podría estar en capacidad de brindar atención inmediata al dato de solicitud de servicio (102) en caso que el dato de urgencia (109) sea el de mayor valor o mayor prioridad, sin considerar el dato de identificación (101). En situaciones críticas el dato de urgencia (109) podría provocar saltos en el protocolo normal de operaciones del sistema. Adicionalmente, el dato de urgencia (109) puede ser un número entero o también puede ser un número real, sin embargo, no se limita a dichos tipos de datos.

En un ejemplo particular, en una línea de espera para realizar una transferencia bancaria, un primer usuario con un primer dato de identificación (101) que tenga una edad de 65 años podría tener un primer dato de urgencia (109) con mayor valor que un segundo dato de urgencia (109) de un segundo usuario con un segundo dato de identificación (101) que tenga una edad de 20 años.

En cualquiera de las modalidades del método, la regla de prioridad (122) puede ser, aunque no limitarse a la siguiente relación:

valor del dato de prioridad (104) = valor del dato de importancia (108) * valor del dato de urgencia (109).

Opcionalmente, el dato de importancia (108) puede ser un dato privado que no se revela a un usuario al cual está relacionado un dato de objeto (100). Por otro lado, el dato de urgencia (109) puede contener información pública relacionada un dato de objeto (100).

También, en cualquiera de las modalidades del método, el proceso de asignación de prioridad (105) de la etapa b) puede incluir una subetapa i3) de calcular el dato de

prioridad (104) con base en la regla de prioridad (122) que puede tomar como entrada el dato de identificación (101) y el dato de solicitud de servicio (102).

Una de las ventajas de calcular el dato de prioridad (104) con base en la regla de prioridad (122) es asegurar que el dato de objeto (100) con mayor jerarquía asignada por el sistema sea el primer dato de objeto (100) en ser asignado a un primer módulo de procesamiento de solicitudes (115). Además, el orden de atención de otros datos de objeto (100) podría realizarse de acuerdo al orden jerárquico establecido por el sistema y podría optimizarse la atención de solicitudes de los datos de objeto (100).

Otra ventaja de calcular el dato de prioridad (104) con base en la regla de prioridad (122), es que un administrador del sistema podría modificar los criterios para la asignación del dato de prioridad (104) de forma manual y a su vez podría modificar también de forma manual la regla de prioridad (122) en función de cualquier cambio o circunstancia particular del sistema.

Por ejemplo, la regla de prioridad (122) podría basarse en la optimización de una función de costo parametrizable, lo que podría permitir a un administrador de sistema modificar los criterios de la función de costo según sus necesidades o requerimientos particulares, lo que podría modificar automáticamente la regla de prioridad (122) ajustando el sistema a las nuevas necesidades o requerimientos del administrador del sistema sin que dicho administrador tenga que realizar una actualización manual del sistema. Alternativamente, la función de costo parametrizable podría ser ajustada automáticamente, por ejemplo, mediante procesos de aprendizaje automático no-supervisado.

También, el cálculo del dato de prioridad (104) con base en la regla de prioridad (122) podría realizarse de forma automática sin la intervención de un administrador del sistema y podrían utilizarse algoritmos que utilicen técnicas inteligencia artificial, machine learning o aplicaciones de control automático para realizar el cálculo, lo cual podría permitir mayor flexibilidad en la forma en que el sistema podría asignar el dato de prioridad (104) a cada dato de objeto (100), así como también, podría modificarse la regla de prioridad (122) de forma automática, dinámica y adaptativa utilizando las técnicas mencionadas anteriormente, de manera tal, que el sistema podría adaptarse a cualquier cambio o circunstancia particular en momentos específicos y optimizar la atención de las solicitudes asociadas a los datos de objeto (100).

Además, en cualquiera de las modalidades del método, el proceso de asignación de prioridad (105) de la etapa b) puede incluir una subetapa i4) de asociar el dato de prioridad (104) al dato de objeto (100).

Asociar el dato de prioridad (104) al dato de objeto (100) puede entenderse como asignar al dato de objeto (100) un nivel de prioridad para que el dato de solicitud de servicio (102) asociado al dato de objeto (100) pueda tener asociado un orden de atención o procesamiento en la línea de espera del sistema, de forma tal, que permita al sistema atender el dato de solicitud de servicio (102) de forma eficiente y en consecuencia de analizar el dato de importancia (108) y el dato de urgencia (109) asociados al dato de solicitud de servicio (102).

En cualquiera de las modalidades del método y haciendo referencia a la FIG. 11, el conjunto de reglas de prioridad (106) de la etapa b) puede obtenerse en una etapa a1) previa a la etapa b), que puede incluir ejecutar con la unidad de cómputo (103) un proceso de clasificación (139) que puede tomar como entrada una pluralidad de datos de prioridad (104) y unos datos de eficiencia (118) asociados a datos de objeto (100) previamente procesados por módulos de solicitudes (114). El proceso de clasificación (139) puede tener un dato de función objetivo (138) configurado para minimizar el tiempo que tarde un dato de objeto (100) en ser procesado.

El proceso de clasificación (139) podría ser la acción de escoger y asignar una categoría a un dato de objeto (100) entre una pluralidad de categorías preestablecidas en el sistema, de tal forma que se pueda diferenciar a un primer dato de objeto (100) con una categoría asociada, de otros datos de objeto (100) con categorías asociadas y diferentes a la categoría del primer dato de objeto (100). El proceso de clasificación (139) podría basarse en características y propiedades únicas de cada dato de objeto (100) y cada categoría podría ser diferente una de la otra. Adicionalmente, cada categoría del sistema podría estar asociado al tipo de dato de solicitud de servicio (102) asociado a cada dato de objeto (100).

El dato de función objetivo (138) puede ser información utilizada para minimizar el tiempo empleado para procesar un dato de objeto (100). El dato de función objetivo (138) puede definir el valor que se pretende alcanzar en la atención de un dato de solicitud de servicio (102) asociado a un dato de objeto (100), teniendo en cuenta las condiciones de operación y las limitaciones, dentro de las que se encuentra el sistema.

También, el método puede incluir una etapa c) de almacenar mediante la unidad de cómputo (103) el dato de objeto (100) en un primer registro de una estructura de datos de prioridad (111) que puede ser almacenada en el módulo de memoria (107). La estructura de datos de prioridad (110) puede tener una pluralidad de registros ordenados (112) de acuerdo con una jerarquía (113) que puede ser determinada con base en el dato de prioridad (104) del dato de objeto (100), además, cada registro de estructura de datos de prioridad (110) puede estar configurado para almacenar un dato de objeto (100).

Se entenderá en la presente divulgación por una estructura de datos de prioridad (110) a cualquier arreglo de registros que pueden estar ordenados de forma jerárquica, es decir, de acuerdo con una jerarquía (113) y que permiten almacenar datos, por ejemplo, datos de objeto (100). El primer registro de dicha estructura, el cual en la presente divulgación corresponde a un primer registro de una estructura de datos de prioridad (111) puede almacenar el elemento con mayor prioridad, y el orden en el que pueden estar ordenados los registros puede corresponder con el nivel de prioridad de cada uno de los elementos, por lo tanto, al recorrer el arreglo en un sentido determinado, por ejemplo, de izquierda a derecha, pueden localizarse una pluralidad de registros ordenados (112) desde el elemento con mayor prioridad hasta el elemento con menor prioridad.

Haciendo referencia a la FIG. 2, en las modalidades del método, la estructura de datos de prioridad (110) puede ser una estructura vectorial (120) que contiene una pluralidad de registros de priorización (121) que pueden tener una jerarquía (113) en la que el registro de priorización (121) con una posición i puede almacenar un dato de objeto (100) con mayor valor del dato de prioridad (104) que un dato de objeto (100) que puede estar almacenado en una posición i+1.

La estructura vectorial (120) puede ser un arreglo que puede estar conformado por registros de priorización (121), en la que puede almacenarse información concerniente a datos de prioridad (104), que puede basarse en una matriz de una sola dimensión (conocida matemáticamente como vector).

Los registros de priorización (121) pueden ser estructuras de almacenamiento de información, en las que las posiciones dentro de los registros de priorización (121) pueden estar asociadas con niveles de prioridad, donde al aumentar el índice de posición "i" puede decrecer el valor de prioridad correspondiente. Considerando que los niveles de prioridad pueden cambiar en el tiempo, también podría actualizarse la información que puede estar almacenada en los registros de priorización (121).

Entiéndase en la presente divulgación que la pluralidad de registros de priorización (121) pueden estar ordenados de forma horizontal según la jerarquía (113), sin embargo, no se limita a dicha forma de organización, ya que la organización mostrada en la FIG. 2 puede corresponder a una forma de mostrar los registros ordenados (112) de manera que sean particularmente visibles en la estructura de datos de prioridad (110).

Una ventaja de que la estructura de datos de prioridad (110) pueda ser una estructura vectorial (120), radica en podría simplificarse el acceso a los registros de priorización (121) de dicha estructura. También, la estructura vectorial (120) podría permitir acceder a los registros de priorización (121) de dicha estructura utilizando solamente un índice asociado

a cada uno de los elementos de la estructura y donde cada uno de los registros de priorización (121) podrían estar ordenados apropiadamente. Por otra parte, se podrían comparar los registros de priorización (121), tan solo con conocer el valor del índice de cada registro de priorización (121), facilitando y optimizando los procesos basados en la jerarquía (113).

Adicionalmente, el método puede incluir una etapa d) de seleccionar un primer módulo de procesamiento de solicitudes (115) entre al menos dos módulos de solicitudes (114) para procesar el dato de solicitud de servicio (102) al ejecutar un proceso de selección de módulo (116) en la unidad de cómputo (103);

Se entenderá en la presente divulgación por un módulo de solicitudes (114) a la unidad que puede realizar el procesamiento o atención del dato de solicitud de servicio (102) de forma separada e independiente de otros módulos de solicitudes (114), por ejemplo, en un banco, un módulo de atención (114) puede ser un módulo de caja o un módulo de atención de servicio al cliente.

Por otro lado, en un sistema informático, un módulo de solicitudes (114) podría ser un elemento de hardware, por ejemplo, un núcleo de un procesador multinúcleo.

Entiéndase en la presente invención que un procesador multinúcleo se refiere a cualquier unidad central de procesamiento (conocida por las siglas CPU, del inglés Central Processing Unit) que contiene dos o más núcleos de procesamiento y cuyos núcleos operan por separado. Ejemplos de procesadores multinúcleo pueden incluir y no limitarse a procesadores dual-core, qual-core, hexa-core, octa-core, deca-core, entre otros. Cada núcleo podría realizar un proceso específico relacionado a un dato de solicitud de servicio (102) de manera independiente.

En otro ejemplo, un módulo de solicitudes (114) podría ser un servidor que puede ser utilizado a nivel local o a través de una red para realizar un proceso específico y que podrían operar de forma simultánea. Por servidor se entiende un equipo informático que forma parte de una red informática o de telecomunicaciones y provee servicios a otros equipos, por ejemplo, servidor de impresiones, servidor de correo, servidor de fax, servidor de telefonía, servidor proxy, servidor de acceso remoto (RAS), servidor web, servidor de base de datos, servidor de seguridad, entre otros.

Además, el primer módulo de procesamiento de solicitudes (115) se entenderá en la presente divulgación como el módulo de atención que ha sido seleccionado entre al menos dos módulos de solicitudes (114) para la atención del dato de solicitud de servicio (102).

También, la selección del primer módulo de procesamiento de solicitudes (115) puede ser el resultado de un proceso de selección de módulo (116) que puede tomar como datos de entrada el dato de prioridad (104) de la etapa b) y unos registros históricos (131) de cada módulo de solicitudes (114).

Una de las ventajas de realizar una buena selección del primer módulo de procesamiento de solicitudes (115) es que la atención del dato de solicitud de servicio (102) se realiza de forma más eficiente y puede reducirse el tiempo de procesamiento de dicha solicitud en el sistema.

Otra ventaja de la selección del primer módulo de procesamiento de solicitudes (115), es que se asigna el módulo de solicitudes (114) con mayor compatibilidad para atender un dato de solicitud de servicio (102). También, podría considerarse el desempeño de los diferentes módulos de solicitudes (114) según registros históricos (131) almacenados en el módulo de memoria (107), lo cual podría reducir la atención de un dato de solicitud (102) asociada a un dato de objeto (100) dado que podría asignarse al módulo de solicitudes (114) más eficiente o que puede estar disponible.

La eficiencia relacionada a cada módulo de solicitudes (114) podría calcularse realizando una evaluación utilizando una ventana de tiempo deslizante, es decir, podría establecerse un periodo de evaluación, por ejemplo, de un mes, dos semanas, una semana, un día, un minuto, un segundo, o inclusive milisegundos, para el cual el sistema podría evaluar la evolución del rendimiento de cada módulo de solicitudes (114). La eficiencia relacionada a cada módulo de solicitudes (114) también podría ser utilizada para determinar si un módulo de solicitudes (114) presenta sobrecarga o degradación, es decir, a dicho módulo de solicitudes (114) podría tomarle más tiempo del esperado para realizar el procesamiento de solicitudes. También, dicha eficiencia podría permitir al sistema determinar cuál podría ser el módulo de solicitudes (114) con menor carga, lo cual podría contribuir en la selección del primer módulo de procesamiento de solicitudes (115) más indicado y podría permitir al sistema realizar dicha selección de forma automática y adaptativa.

En algunos ejemplos particulares una sobrecarga o degradación podría corresponder a la degradación de un módulo de solicitudes (114) que realiza procesamientos menos rápido y el sistema puede adaptarse para no sobrecargar dicho módulo de solicitudes (114), o un módulo de solicitudes (114) mejora su eficiencia, en donde el módulo de solicitudes (114) podría ser un operario más eficiente o con mejor entrenamiento, un núcleo de procesamiento, dispositivo de cómputo o elemento de hardware (unidades de memoria, procesadores, BUS de comunicaciones, redes de comunicaciones, módulos de comunicaciones) que ha sido renovado, actualizado, o agregado, entre otros, y

ventajosamente el sistema se podría adaptar para asignar tareas de acuerdo al nuevo comportamiento del módulo de solicitudes (114).

Para este ejemplo la función de costo para la evaluación de la priorización de solicitudes podría no sufrir modificaciones, y podría cambiar uno o algunos de los datos de entrada para dicha función. En este ejemplo, el desempeño de los módulos de solicitudes (114) son los que presentan una variación y de esta manera la asignación de un proceso relacionado a un dato de solicitud de servicio (102) que puede ser asignado a un módulo de solicitudes (114) podría ser modificada de forma automática y adaptativa.

Haciendo referencia a la FIG. 3, en cualquiera de las modalidades del método, en el proceso de selección de la etapa d) la unidad de cómputo (103) puede generar una estructura matricial (123) con una pluralidad de vectores de procesamiento (124). Cada vector de procesamiento (129) puede estar asociado a un módulo de procesamiento de solicitudes (114, 115) y puede tener una pluralidad de registros de atención (125). Cada registro de atención (125) puede almacenar un dato de objeto (100).

La ventaja de generar una estructura matricial (123) es que se podría organizar la información asociada a los datos de objeto (100) de forma compacta, además, por medio de dos índices se podría acceder a cualquier elemento de la matriz de forma rápida y eficiente. Por otra parte, se podrían agregar nuevos entes prestadores de servicio, por ejemplo, más módulos de solicitudes (114) que podrían corresponden a nuevas filas o vectores de procesamiento (129) en la estructura matricial (123), esto mediante una operación simple y sin afectar los vectores de procesamiento (129) que previamente podrían formar parte de la estructura matricial (123).

Entiéndase en la presente divulgación por una estructura matricial (123) al arreglo que puede estar formado por una pluralidad de vectores de procesamiento (124), donde para acceder a los elementos individuales se necesitan dos índices, uno para una fila y otro para una columna. Cada uno de los vectores de procesamiento (129) que pueden representar las filas en la estructura matricial (123) pueden estar asociados con un módulo de solicitudes (114). Las columnas de la estructura matricial (123) pueden estar asociados a registros de atención (125), adicionalmente, la estructura matricial (123) podría actualizarse periódicamente para saber cuando los módulos de solicitudes (114) se encuentran en estado de "ocupados" o "indisponibles". También, la actualización de la estructura matricial (123) y tener en cuenta el estado de los módulos de solicitudes (114) podría generar la atención de datos de solicitud de servicio (102) de forma más rápida, con la posibilidad de asignar un primer módulo de procesamiento de solicitudes (115) a un dato de solicitud de servicio (102) con el dato de prioridad (104) más alto en un momento específico.

También, debe entenderse en la presente divulgación por un registro de atención (125) a una estructura en la que puede almacenarse un dato de objeto (100), con toda la información asociada a dicho dato de objeto (100), de tal forma que se pueda acceder a la información y utilizarla en los diferentes procedimientos y funciones, empleados en el método de la presente solicitud. Los registros de atención (125) se actualizan periódicamente, dado que el dato de objeto (100) puede cambiar en las diferentes variables asociadas con el mismo, por ejemplo, un dato de identificación (101) y un dato de solicitud de servicio (102) a medida que se realiza la actualización de la estructura matricial (123).

En cualquiera de las modalidades del método y haciendo referencia a la FIG. 5, el proceso de selección de un primer módulo de procesamiento de solicitudes (115) de la etapa d) puede incluir una subetapa j1) de obtener un dato de duración de servicio (126) asociado al dato de solicitud de servicio (102).

Entiéndase en la presente divulgación por dato de duración de servicio (126) a un dato con un valor de tiempo empleado en la atención de un dato de objeto (100) al cual está asociado un dato de solicitud de servicio (102) dentro de una gama de solicitudes de servicio disponibles en el sistema. El dato de duración de servicio (126) también puede estar asociado al tiempo promedio en la atención de un dato de solicitud de servicio (102), de manera que el dato de duración de servicio (126) puede ser utilizado para realizar la selección eficiente de un primer módulo de procesamiento de solicitudes (115).

El dato de duración de servicio (126) también podría servir como un indicador del desempeño del sistema, lo cual podría permitir encontrar estrategias para mejorar la atención de solicitudes de los datos de objeto (100) al analizar las condiciones en las que el valor de dato de duración de servicio (126) se reduce, así como también al analizar aquellas condiciones en las que el valor de dato de duración de servicio (126) se incrementa, lo cual podría permitir al sistema determinar las condiciones óptimas de operación, también podría permitir evitar un colapso del sistema.

Por ejemplo, si las solicitudes asignadas a un primer módulo de procesamiento de solicitudes (115) son ejecutadas por un humano y el valor del dato de duración de servicio (126) incrementa con respecto a promedios históricos, podría indicar que el humano se encuentra afectado por uno o varios factores que afectan su desempeño en la atención de solicitudes, de manera que podría buscarse reducir la cantidad de solicitudes asignadas a ese primer módulo de procesamiento de solicitudes (115) durante un periodo de tiempo definido o buscar otra alternativa para mejorar el valor del dato de duración de servicio (126) asociado al primer módulo de procesamiento de solicitudes (115) de manera tal, que se normalicen los tiempos de atención de solicitudes de dicho módulo.

Por otro lado, si las solicitudes asignadas a un primer módulo de procesamiento de solicitudes (115) son ejecutadas por un dispositivo de cómputo y/o elementos de hardware, y el valor del dato de duración de servicio (126) incrementa con respecto a promedios históricos, esto puede indicar que es necesario hacer un mantenimiento, reemplazo o actualización de los mismos, y/o de los componentes de software que puedan utilizar el dispositivo de cómputo y/o elementos de hardware (v.g., drivers, sistema operativo, programas dedicados al procesamiento de datos).

Por ejemplo, el dato de duración de servicio (126) podría permitir adaptar la asignación de clientes a módulos de solicitudes (114), sin alterar la función de costo y la asignación de prioridades, lo que podría incrementar la flexibilidad del sistema y optimizar la asignación de prioridades. Una mayor flexibilidad del sistema podría permitir mejorar las estrategias de optimización en la asignación de solicitudes, y alcanzar niveles óptimos de operación en comparación con un grado de flexibilidad menor. Desde el punto de vista matemático, un nivel de operación óptimo se podría encontrar en un valor máximo o mínimo de una superficie de solución. También, desde el punto de vista matemático, sería posible que al aumentar la dimensión de un espacio de solución pudieran aparecer valores mínimos o valores máximos globales del sistema que pudieran ser mejores que los valores máximos o valores mínimos locales que podrían encontrarse en una dimensión inferior, como, por ejemplo, relacionados a módulos de solicitudes (114).

En cualquiera de las modalidades del método, el proceso de selección de un primer módulo de procesamiento de solicitudes (115) de la etapa d) puede incluir una subetapa j2) de obtener un dato de duración de registro (128) asociado a cada registro de atención (125).

Entiéndase en la presente divulgación por dato de duración de registro (128) a la información asociada con cada registro de atención (125) y cuyo valor podría ser mayor o igual al valor del dato de duración de servicio (126) y que al cumplirse dicha condición, permite que los registro de atención (125) puedan ser tomados en cuenta para realizar la selección eficiente de un primer módulo de procesamiento de solicitudes (115) entre el total de módulos de solicitudes (114) disponibles en el sistema y en donde dicha selección podría permitir optimizar el proceso general de atención de solicitudes asociadas a datos de objeto (100), considerando las circunstancias y condiciones de operación del sistema en diferentes instantes de tiempo.

En cualquiera de las modalidades del método, el proceso de selección de un primer módulo de procesamiento de solicitudes (115) de la etapa d) puede incluir una subetapa j3) de identificar registros de atención (125) disponibles y ordenados de forma consecutiva en un vector de procesamiento (129) cuyo dato de duración de registro (128) puede tener un valor igual o mayor al valor del dato de duración de servicio (126), esta subetapa también puede

contribuir a realizar la selección eficiente de un primer módulo de procesamiento de solicitudes (115) en la medida que se consideran solamente los registros de atención (125) con capacidad suficiente para atender el dato de solicitud de servicio (102) al cual está asociado el dato de duración de servicio (126).

También, en cualquiera de las modalidades del método, el proceso de selección de un primer módulo de procesamiento de solicitudes (115) de la etapa d) puede incluir una subetapa j4) de identificar un vector de procesamiento (129) asociado con los registros de atención (125) que pueden ser identificados en la subetapa j3), lo cual es ventajoso ya que, entre todos los módulos de solicitudes (114) son considerados solamente los módulos de solicitudes (114) con capacidad suficiente para atender el dato de solicitud de servicio (102) al cual está asociado el dato de duración de servicio (126) y podría realizarse la selección de un primer módulo de procesamiento de solicitudes (115) de forma más rápida, contribuyendo a optimizar el proceso de selección.

También, la subetapa j4) de identificar un vector de procesamiento (129) asociado con los registros de atención (125) que pueden ser identificados en la subetapa j3) podría asegurar la atención apropiada de las solicitudes de los datos de objeto (100), puesto que solamente se podrían tener en cuenta aquellos módulos de solicitudes (114) aptos para prestar el nivel de servicio requerido para la atención de cada solicitud y, de acuerdo a registros de operación del sistema en periodos pasados, en donde dichos registros podrían almacenarse en el módulo de memoria (107), y que al analizar dichos registros, el sistema podría determinar cuáles podrían ser los módulos de solicitudes (114) más apropiados para atender datos de solicitud de servicio (102) específicos entre una pluralidad de datos de solicitud de servicio (102) disponibles en el sistema.

Adicionalmente, en cualquiera de las modalidades del método, el proceso de selección de un primer módulo de procesamiento de solicitudes (115) de la etapa d) puede incluir una subetapa j5) de seleccionar el primer módulo de procesamiento de solicitudes (115). El primer módulo de procesamiento de solicitudes (115) puede estar asociado al vector de procesamiento (129) que puede ser identificado en la etapa j4) y asociado a un módulo de solicitudes (114) específico.

Haciendo referencia a la FIG. 6, en cualquiera de las modalidades del método además puede incluir entre los pasos j4) y j5) un subpaso k1) de obtener un dato de duración de servicio mínimo (130) que puede estar asociado a cada vector de procesamiento (129) y puede estar asociado al dato de solicitud de servicio (102). El dato de duración de servicio mínimo (130) puede determinarse con base en unos registros históricos (131) que pueden estar almacenados en el módulo de memoria (107).

Entiéndase en la presente divulgación por dato de duración de servicio mínimo (130) al valor que especifica cuál de los módulos de solicitudes (114) es el más apropiado para atender un dato de solicitud de servicio (102), dado que es el módulo de solicitudes (114) con el dato de duración de servicio mínimo (130) de menor valor con respecto al resto de módulos de solicitudes (114) del sistema y que además, está relacionado a la atención de un dato de solicitud de servicio (102) entre un conjunto de solicitudes de servicio disponibles en el sistema. Para establecer el dato de duración de servicio mínimo (130) pueden utilizarse registros históricos (131) de cada módulo de solicitudes (114) y pueden compararse los registros históricos (131) asociados a un dato de solicitud de servicio (102) y además, obtener un promedio del dato de duración de servicio mínimo (130), de tal manera que se puede seleccionar un primer módulo de procesamiento de solicitudes (115) que puede corresponder al módulo de solicitudes (114) con un menor valor del dato de duración de servicio mínimo (130) con respecto al resto de módulos de solicitudes (114).

También, debe entenderse en la presente solicitud por registros históricos (131) al conjunto de datos, por ejemplo, datos estadísticos, que permiten seguir la evolución de parámetros de un dato de objeto (100), como el tiempo de espera y/o el tiempo de atención de un dato de objeto (100) en un módulo de solicitudes (114), entre otros, de manera que se cuente con un criterio para escoger el módulo de solicitudes (114) más eficiente, utilizando la información de los registros históricos (131).

En cualquiera de las modalidades del método, se puede incluir entre los pasos j4) y j5) un subpaso k2) de comparar el dato de duración de servicio mínimo (130) que puede estar asociado a cada vector de procesamiento (129), lo anterior es ventajoso debido a que cada vector de procesamiento (129) está asociado a un módulo de solicitudes (114) y comparar el dato de duración de servicio mínimo (130) de cada módulo de solicitudes (114) permite al sistema seleccionar el módulo de solicitudes (114) más indicado para la atención de un dato de solicitud de servicio (102).

En cualquiera de las modalidades del método, se puede incluir entre los pasos j4) y j5) un subpaso k3) de identificar el vector de procesamiento (129) que puede tener el dato de duración de servicio mínimo (130) con menor valor con base a la comparación que puede ser realizada en la etapa k2) y por lo tanto identificar el módulo de solicitudes (114) con el dato de duración de servicio mínimo (130) entre la pluralidad de módulos de solicitudes (114) del sistema.

También, en cualquiera de las modalidades del método, se puede incluir entre los pasos j4) y j5) un subpaso k4) de seleccionar el vector de procesamiento (129) que puede estar asociado a un módulo de solicitudes (114) para procesar el dato de solicitud de servicio (102) con base en la identificación que puede ser realizada en la etapa k3). En la etapa j5)

se puede seleccionar como primer módulo de procesamiento de solicitudes (115) el módulo de solicitudes (114) que puede tener el dato de duración de servicio mínimo (130) con menor valor.

Asimismo, se podría contar con la ventaja de que el vector de procesamiento (129) podría permitir la optimización de la atención de solicitudes asociadas a datos de objeto (100), ya que podría contener solamente aquellos elementos necesarios para la toma de decisiones por parte del sistema y para ejecutar los procesos permanentemente. Esta sencillez podría simplificar los procedimientos que ejecuta el sistema, y podría, además, permitir incrementar la velocidad de procesamiento y operación del sistema en general.

Haciendo referencia a la FIG. 7, en cualquiera de las modalidades del método, además, se puede incluir un paso k5) de actualizar con la unidad de cómputo (103) la estructura matricial (123) de forma periódica que puede incluir un subpaso v1) de obtener un dato de actualización (132) que puede estar asociado a la cantidad de módulos de solicitudes (114) habilitados.

El dato de actualización (132) puede incluir la información asociada con el número de módulos de solicitudes (114) que se encuentran habilitados y que puede determinar la capacidad del sistema para atender un número limitado de datos de objeto (100), de manera tal, que no se afecte la calidad de la atención de los datos de solicitudes (102).

En cualquiera de las modalidades del método acá divulgado, opcionalmente, módulos de solicitudes (114) adicionales podrían habilitarse en función del dato de actualización (132).

En cualquiera de las modalidades del método, el paso k5) también puede incluir un subpaso v2) de modificar la pluralidad de vectores de procesamiento (124) de la estructura matricial (123) con base al número de módulos de solicitudes (114) habilitados. Lo anterior podría evitar que el sistema tuviera que realizar reprocesos innecesarios o erróneos, como por ejemplo un proceso de selección de un primer módulo de procesamiento de solicitudes (115) en el cual se consideren módulos de solicitudes (114) que podrían provocar fallos o retrasos en la atención de datos de solicitud de servicio (102).

Adicionalmente, el subpaso v2) de modificar la pluralidad de vectores de procesamiento (124) de la estructura matricial (123) con base al número de módulos de solicitudes (114) habilitados podría incrementar la eficiencia del sistema en general, debido a que podrían procesarse exclusivamente solicitudes que no tienen errores y asignarlas solamente a los módulos de solicitudes (114) habilitados y de esta forma, se podría evitar sobrecargar innecesariamente el sistema. También, el sistema podría omitir realizar el procesamiento de datos relacionados a solicitudes que podrían contener errores y que podrían ser asignados a

módulos de solicitudes (114) habilitados, mientras que las solicitudes sin errores podrían ser procesadas inmediatamente por los módulos de solicitudes (114).

Por otro lado, haciendo referencia a la FIG. 8, en cualquiera de las modalidades del método, el paso k5) también puede incluir después del paso v2 un subpaso v3) de generar con la unidad de cómputo (103) un dato de notificación de sobrecarga (133) que puede estar asociado a un módulo de solicitudes (114) que puede tener un dato de rendimiento (134) con un valor que puede caer dentro de un intervalo de criticidad (135), de lo contrario, puede pasar a la etapa e).

El dato de rendimiento (134) puede incluir la información asociada a un módulo de solicitudes (114) como consecuencia de la atención de al menos un dato de solicitud de servicio (102). El valor del dato de rendimiento (134) podría estar dentro de un rango preestablecido y también podría estar asociado a un periodo específico en el cual el módulo de solicitudes (114) realizó la atención de al menos un dato de solicitud de servicio (102).

Un ejemplo de un dato de rendimiento (134) podría ser la duración de atención de un dato de solicitud de servicio (102) por parte de un primer módulo de procesamiento de solicitudes (115) con una duración de atención de 2 minutos y que podría estar dentro de un rango establecido por el sistema que podría variar de 1 a 5 minutos, por lo cual, el dato de rendimiento (134) podría considerarse dentro del rango de eficiencia.

En una modalidad de la invención el dato de rendimiento (134) podría estar relacionado a la optimización energética del sistema, en donde podría asociarse a la cantidad de energía necesaria para atender un número de solicitudes. De acuerdo a una ponderación en función del costo energético, el conjunto de reglas de prioridad (106) podría asignar mayor número de tareas a aquellas unidades de procesamiento o módulos de solicitudes (114) que consuman menor cantidad de energía, de modo que en su conjunto el sistema podría dejar una menor huella de carbono y podría provocar una reducción en los costos de operación.

En otra modalidad de la invención el dato de rendimiento (134) podría estar relacionado a unidades de proceso, en donde el dato de rendimiento (134) podría ser un indicador de la eficiencia de cada una de las unidades de proceso. Considerando que cada tarea puede dividirse en unidades básicas de proceso, estas unidades podrían usarse como elemento normalizador para la evaluación comparativa del desempeño de cada una de las unidades de procesamiento. En caso de que el rendimiento de una unidad decaiga, podrían realizarse ajustes, mantenimiento preventivo o correctivo que hagan falta, entre otros. En el caso de unidades de procesamiento humano, el dato de rendimiento (134) podría servir para generar estímulos y recompensas al trabajo bien ejecutado.

El intervalo de criticidad (135) puede incluir un rango de valores preestablecidos asociados al dato de rendimiento (134) de modo que, cuando el dato de rendimiento (134) se encuentre en el rango de valores antes mencionado, se puede producir un dato de notificación de sobrecarga (133) que puede ser interpretado en el sistema como una notificación de alerta o alarma. El intervalo de criticidad (135) podría modificarse como consecuencia de un cambio en el número de módulos de solicitudes (114) del sistema o en caso de realizar mejoras en las características de los módulos de solicitudes (114) y donde las mejoras podrían ser mejoras de hardware en los módulos de solicitudes (114) para incrementar la eficiencia de atención de datos de solicitud de servicio (102).

En cualquiera de las modalidades del método acá divulgado, opcionalmente, algunos módulos de solicitudes (114) podrían deshabilitarse periódicamente para evitar la sobrecarga de estos y evitar afectar al sistema en la atención de datos de solicitudes (102), por ejemplo, evitando seleccionar un primer módulo de procesamiento de solicitudes (115) entre módulos de solicitudes (114) que presentan sobrecarga.

También, en cualquiera de las modalidades del método, el paso k3) además puede incluir después del paso v2 un subpaso v4) de generar un dato de inhabilitación temporal (136) con la unidad de cómputo (103) cuando se genera el dato de notificación de sobrecarga (133). El dato de inhabilitación temporal (136) puede impedir a dicho módulo de solicitudes (114) procesar un dato de solicitud de servicio (102) durante un intervalo de tiempo predefinido.

El dato de inhabilitación temporal (136) en la presente divulgación puede entenderse como la información obtenida al activarse el dato de notificación de sobrecarga (133) asociado a un módulo de solicitudes (114), de tal forma que el dato de inhabilitación temporal (136) podría evitar que el módulo de solicitudes (114) al cual está asociado el dato de notificación de sobrecarga (133) procese un dato de solicitud de servicio (102) dentro de un periodo de tiempo establecido y se eviten fallos o retrasos en la atención del dato de solicitud de servicio (102).

En las modalidades del método en las que el módulo de solicitudes (114) tenga un procesador, unidad de cómputo o elemento de hardware similar, el tener obtener el dato de inhabilitación temporal (136) y evitar que el módulo de solicitudes (114) al cual está asociado el dato de notificación de sobrecarga (133) procese un dato de solicitud de servicio (102) dentro de un periodo de tiempo establecido, permite prevenir, y hasta evitar daños en elementos de hardware debido a sobrecalentamiento de los procesadores y/o unidades de cómputo del módulo de solicitudes (114) que puedan a su vez sobrecalentar otros elementos de hardware y circuitos (v.g., memorias, BUS de comunicaciones, módulos de comunicaciones, etc).

El dato de inhabilitación temporal (136) también podría aumentar la eficiencia global del sistema en la medida que un módulo de solicitudes (114) sobrecargado podría provocar disminuir la eficiencia de todo el sistema en una ventana de tiempo, en cuyo caso, podría utilizarse el dato de inhabilitación temporal (136) para lograr que uno o varios módulos de solicitudes (114) sobrecargados pudieran inhabilitarse temporalmente, recuperarse y nuevamente alcanzar un punto óptimo de funcionamiento.

En un ejemplo particular, una limitación del uso de elementos sobrecargados podría evitar una degradación de la eficiencia energética del sistema, reduciendo su huella de carbono. En otro ejemplo en el cual un ser humano procesa la solicitud, como en el caso de un cajero bancario, inhibir al cajero durante un tiempo le podría permitir a dicho cajero recuperarse y retomar su trabajo en máxima eficiencia. Continuar realizando tareas en sobrecarga, puede significar el aumento de errores de proceso que pueden poner en riesgo al sistema, a un cliente o causar costos innecesarios al sistema.

El dato de inhabilitación temporal (136) también podría asegurar el uso eficiente de los recursos del sistema, de tal forma que dichos recursos podrían emplearse solamente cuando el sistema considera que es absolutamente necesario utilizar dichos recursos.

En cualquiera de las modalidades del método y haciendo referencia a la FIG. 9, el paso k5) de actualizar la estructura matricial (123) además puede incluir un subpaso v5) de habilitar uno o más módulos de solicitudes (114) adicionales por medio de la unidad de cómputo (103) con base al número de módulos de solicitudes (114) que pueden encontrarse inhabilitados y con base a la notificación de sobrecarga (133). Habilitar uno o más módulos de solicitudes (114) podría reducir el dato de duración de servicio (126) asociado al dato de solicitud de servicio (102) en la medida que se podría seleccionar un primer módulo de procesamiento de solicitudes (115) con un valor del dato de duración de servicio mínimo (130) para hacer más eficiente la atención del dato de solicitud de servicio (102).

En cualquiera de las modalidades del método el subpaso v5 es ejecutado posterior a ejecutarse el subpaso v4, sin embargo, en una modalidad particular del método los subpasos v4) y v5) podrían implementarse simultáneamente y podrían permitir una mejora en la eficiencia del sistema, en la medida que se utilizan o se inhabilitan los módulos de solicitudes (114) considerando el desempeño particular de cada módulo de solicitudes (114).

El método puede además incluir una etapa e) de enviar desde la unidad de cómputo (103) el dato de solicitud de servicio (102) del dato de objeto (100) que tenga el mayor valor del dato de prioridad (104) al primer módulo de procesamiento de solicitudes (115) que puede ser seleccionado en la etapa d). Preferiblemente, el primer módulo de procesamiento de solicitudes (115) al cual es enviado el dato de solicitud de servicio (102) puede garantizar

que la atención del dato de solicitud de servicio (102) se realizará de la forma más rápida puesto que corresponde al módulo de solicitudes (114) con mejor desempeño.

Haciendo referencia a la FIG. 1 y la FIG.12, el método puede incluir una etapa f) de obtener mediante la unidad de cómputo (103) un conjunto de variables de rendimiento (117) en el módulo de memoria (107) que pueden ser generadas mientras el primer módulo de procesamiento de solicitudes (115) procesa el dato de solicitud de servicio (102) enviado en la etapa e).

El conjunto de variables de rendimiento (117) puede ser un grupo de valores o conjunto de variables utilizadas para determinar el nivel de funcionamiento de los módulos de solicitudes (114), con el propósito de asignar en futuros procesos los módulos de solicitudes (114) con mayor nivel de servicio a datos de objeto (100) con un dato de prioridad (104) mayor. Ejemplos de variables de rendimiento (117) pueden ser y no se limitan a datos estadísticos que pueden ser resultado de un análisis utilizando estadística descriptiva, estadística diferencial, estadística aplicada o estadística matemática; ejemplos de variables de rendimiento (117) pueden ser tiempo total de procesamiento de solicitudes por cada módulo de solicitudes (114), número de solicitudes procesadas por unidad de tiempo por cada módulo de solicitudes (114), un valor de proporción entre la cantidad de solicitudes que ingresan al sistema y la cantidad de solicitudes atendidas o procesadas por cada módulo de solicitudes (114), cantidad de recursos energéticos consumidos para realizar un proceso en el caso que un módulo de solicitudes (114) pudiera ser el núcleo de un procesador, entre otros. Este conjunto de variables puede ser establecido por un administrador del sistema, y puede variar en el tiempo de forma flexible, adaptándose a los diferentes tipos de solicitudes y también pueden ser utilizadas en procesos de selección de módulo (116) de la etapa d) o en un proceso de calificación (119) de la etapa g).

También, el método puede incluir una etapa g) de obtener un dato de eficiencia (118) del procesamiento de la solicitud de servicio (102) que puede ser ejecutando con la unidad de cómputo (103) un proceso de calificación (119) que puede tomar como entrada el conjunto de variables de rendimiento (117).

Entiéndase en la presente solicitud que el dato de eficiencia (118) puede incluir o representar información que permite evaluar el desempeño de los módulos de solicitudes (114), por ejemplo, un valor de un número entero, fracción, porcentaje, un numero racional, una proporción, entre otros, de tal forma que aquellos con mejor desempeño, es decir, los más eficientes, puedan ser asignados en la atención de datos de objeto (100) con un dato de prioridad (104) mayor, según la clasificación. Alternativamente, el dato de eficiencia (118) puede incluir valores de variables categóricas, por ejemplo, escalas de valoración de satisfacción que diligencia o informa un usuario asociado a un dato de objeto (100).

También, el dato de eficiencia (118) puede ser un indicador multicriterio de un conjunto de indicadores de desempeño.

Un dato de eficiencia (118) también puede ser un dato de una sola dimensión, por ejemplo, un número real obtenido del computo de un polinomio de cualquier grado, en donde las variables pueden ser las variables de desempeño y los coeficientes de dicho polinomio podrían ser proporciones de ponderación o podrían ser un vector de "n" dimensiones, en donde cada dimensión podría representar un criterio y donde cada uno de los criterios podría representar una dimensión ortogonal del espacio de desempeño y de esta forma, podrían definirse diferentes tipos de desempeño. En esta forma podrían definirse diferentes tipos de desempeño que, a pesar de tener misma magnitud, por ejemplo, módulo de solicitudes (114) asociado a un vector, podría representan situaciones diferentes, por ejemplo, diferencia entre eficiencia energética y la velocidad de atención de clientes u otros indicadores de desempeño.

Particularmente, la eficiencia del sistema podría ser balanceada al evitar que los módulos de solicitudes (114) no se sobrecarguen y pueda distribuirse la atención de los datos de objeto (110) entre todos o la mayoría de los módulos de solicitudes (114), las variables de rendimiento (117) de la etapa f) pueden ser utilizadas para identificar módulos de solicitudes (114) con posibles sobrecargas.

En cualquiera de las modalidades del método y haciendo referencia a la FIG. 10, en la etapa g) en la cual la unidad de cómputo (103) puede ejecutar un proceso de calificación (119) puede incluir una subetapa g1) de registrar un conjunto de variables de rendimiento (117) relacionadas con la atención del dato de solicitud de servicio (102) del dato de objeto (100) mediante la unidad de cómputo (103) en el módulo de memoria (107). Opcionalmente, el conjunto de variables de rendimiento (117) pueden ser utilizadas como datos de entrada para futuras selecciones del primer módulo de procesamiento de solicitudes (115) de la etapa b).

En cualquiera de las modalidades del método, la etapa g) puede incluir una subetapa g2) de evaluar el conjunto de variables de rendimiento (117) con base en un tiempo de procesamiento (137) del dato de solicitud de servicio (102). El tiempo de procesamiento (137) del dato de solicitud de servicio (102) puede ser el periodo comprendido desde la recepción del dato de solicitud de servicio (102) en la unidad de cómputo (103) hasta que el primer módulo de procesamiento de solicitudes (115) finaliza el procesamiento del dato de solicitud de servicio (102).

En cualquiera de las modalidades del método, la etapa g) puede incluir una subetapa g3) de registrar un dato eficiencia (118) del procesamiento de la solicitud de servicio (102) en el módulo de memoria (107) con base en la evaluación del conjunto de variables de rendimiento (117). Opcionalmente, el dato eficiencia (118) puede ser utilizado como dato de entrada para futuras selecciones del primer módulo de procesamiento de solicitudes (115) de la etapa b).

El uso del conjunto de variables de rendimiento (117) podría posibilitar conocer en un instante dado la evolución de cada uno de los procesos y rutinas asociados a la atención de datos de solicitud de servicio (102) desde diferentes puntos de vista, y podría facilitar la comparación entre diferentes alternativas con las que podría contar el sistema para la atención de las solicitudes de servicio (102).

Un dato eficiencia (118) podría permitir al sistema evaluar el desempeño en tiempo real de los módulos de solicitudes (114) o de todo el sistema en general, de tal forma que se podría contar con un valor de referencia para determinar el funcionamiento del sistema y si dicho sistema cumple con los niveles de operación y atención esperados, o si es necesario realizar acciones de mejora al sistema con base en un conjunto de estrategias definidas. Adicionalmente, el dato eficiencia (118) podría permitir la adaptación dinámica del sistema ante cambios progresivos o repentinos que podría sufrir el sistema, tales como degradación de elementos del sistema, fallas súbitas del sistema o fallas puntuales en elementos del sistema, colapso de un operador que podría ser un ser humano en un ejemplo de un cajero bancario, entre otros.

El proceso de calificación (119) puede ser un procedimiento de evaluación del conjunto de variables de rendimiento (117), almacenadas en el módulo de memoria (107), donde dicho conjunto de variables de rendimiento (117) son evaluadas de acuerdo al tiempo de procesamiento del dato de solicitud de servicio (102). El resultado de la evaluación del conjunto de variables de rendimiento (117) se puede almacenarse en el módulo de memoria (107), en forma de registros, de tal forma que se pueda analizar la evolución en el tiempo, y las tendencias de las diferentes variables de rendimiento, para los diferentes procesos de toma de decisiones. Ejemplos de un proceso de calificación (119) puede ser determinar cuáles son los tres módulos de solicitudes (114) más rápidos, determinar cual es el módulo de solicitudes (114) que procesa la menor cantidad de solicitudes asociadas a datos de objeto (100), cual es el módulo de solicitudes (114) que ha procesado el mayor número de solicitudes asociadas a datos de objeto (100), cual es el módulo de solicitudes (114) con los mejores registros históricos para una métrica en particular, entre otros.

Adicionalmente, haciendo referencia a la FIG. 1 y la FIG.12, el método puede incluir una etapa h) de modificar mediante la unidad de cómputo (103) el conjunto de reglas de

prioridad (106) que pueden estar almacenadas en el módulo de memoria (107) a partir del dato de eficiencia (118).

Particularmente, la etapa h) permite que el proceso de asignación de prioridad (105) y el proceso de selección de módulo (116) para la atención de los datos de objeto (100) y sus respectivos datos de solicitud de servicio (102) se realicen de manera más eficiente, ya que permite al sistema adaptarse a la cantidad de módulos de solicitudes (114) disponibles o a posibles sobrecargas de uno o varios de los módulos de procesamiento (114), así como también asignar la atención de un dato de objeto (100) al módulo de solicitudes (114) más adecuado en función de su dato de eficiencia (118).

Una ventaja de la modificación del conjunto de reglas de prioridad (106), es que el sistema podría adaptarse a las condiciones particulares de operación, de tal forma que el sistema podría permanecer funcionando de forma óptima en caso de que cambiaran dichas condiciones e independiente de dichas condiciones, además, el sistema podría actualizarse en cualquier instante en caso de ser necesario.

La modificación del conjunto de reglas de prioridad (106) también podría permitir a un administrador del sistema adaptar fácilmente la estrategia de asignación de prioridades para de acuerdo a cambios en la función de costos del sistema para la atención de solicitudes de datos de objeto (100), por ejemplo, en caso que el administrador del sistema cambie ponderaciones de los criterios de eficiencia, podría incluir nuevos criterios o eliminar algunos de los criterios ya presentes en el sistema, de esta forma, el sistema podría adaptarse a los nuevos cambios sin que se necesite una actualización masiva en la estrategia de asignación de prioridades. El tiempo necesario para la realización de cambios podría reducirse y la complejidad de la tarea de actualización en el sistema también podría reducirse.

Opcionalmente, en cualquiera de las modalidades del método que incluyen la etapa h), el conjunto de reglas de prioridad (106) pueden modificarse de manera automática al ejecutar con una unidad de cómputo (v.g., la unidad de cómputo (103)) un proceso de aprendizaje automático, el cual puede ser supervisado o puede ser no-supervisado.

En el caso de que el proceso de aprendizaje automático sea supervisado, el proceso de aprendizaje automático toma como entrada los datos de eficiencia (118) y un dato de modificación de regla que ingresa un usuario en la unidad de cómputo. El dato de modificación de regla puede ser un dato que ajusta parámetros, reglas, hiperparámetros (profundidad máxima, profundidad mínima, límite de iteraciones, función objetivo, pesos, sesgos, entre otros) del proceso de aprendizaje automático.

Por ejemplo, el proceso de aprendizaje automático puede tomar como entrada un arreglo de datos históricos que incluye valores de datos de objeto (100) previamente procesador y sus correspondientes datos de eficiencia (118) asociados a los módulos de solicitudes (114) que procesaron dichos datos de objeto (100).

Similarmente, el proceso de aprendizaje automático puede incluir pasos de normalización, reducción dimensional de variables, estandarización, y demás pasos de procesamiento de datos que permitan facilitar la identificación de correlaciones entre las variables que se pueden obtener de los datos del arreglo de datos históricos.

Cuando el proceso de aprendizaje automático es no supervisado, la unidad de cómputo puede entrenarse con el arreglo de datos históricos, el cual puede dividirse en un arreglo de datos de entrenamiento y en un arreglo de datos de validación, por ejemplo, en una proporción 70%-30%, 75%-25%, 80%-20%, o cualquier otra proporción viable. La proporción seleccionada puede opcionalmente modificarse de manera manual por un usuario con el fin de evitar sobreajuste (overfitting, en inglés). Opcionalmente, el proceso de aprendizaje automático es no supervisado se reentrena constantemente con base al añadirse en tiempo real, o periódicamente nuevos registros al arreglo de datos históricos de acuerdo a como se van procesando los datos de objeto (100).

De acuerdo con lo anterior, el proceso de aprendizaje automático puede seleccionarse entre procesos de procesos de regresión (v.g., regresión, lineal, regresión polinómica, regresión exponencial, regresión logarítmica, regresión logística), procesos de clasificación lineal (v.g. regresión logística, clasificación de Naive Bayes, discriminante lineal de Fisher), árboles de decisión, aprendizaje de reglas de asociación, aprendizaje profundo, algoritmos de aprendizaje genético, programación de lógica inductiva, máquinas de vectores de soporte (SVM), redes bayesianas, aprendizaje de refuerzo, aprendizaje de representación, aprendizaje automático basado en reglas, escaso aprendizaje de diccionarios, similitud y aprendizaje métrico, sistemas de clasificación de aprendizaje (LCS), (agrupamiento) clustering (v.gr., k-means, k-nearest neighborg, fuzzy k-means) redes neuronales artificiales, (v.gr., Artificial Neural Networks (ANN), Convolutional Neural Networks (CNN), Radial Basis Networks (RBN), Feed-Forward Neural Networks (FF), Deep Feed-forward Neural Networks (DFF), Hopfield Neural Networks (HN), Deep Convolutional Networks (DCN), Deconvolutional Neural Networks (DN), Deep Residual Networks (DRN)), máquinas de soporte vectorial (support vector machines, en inglés), procesos de clasificación basados en árboles de decisión (v.g., de árbol, XGBoost, Random Forest) y demás procesos de aprendizaje automático similares conocidos por una persona medianamente versada en la materia.

Una de las ventajas de incluir en cualquiera de las modalidades del método acá divulgadas los procesos de aprendizaje automatico supervisado es que un administrador del sistema podria modificar el conjunto de reglas de prioridad (106) y podria tener mayor control de estas.

Similarmente, una de las ventajas de incluir en cualquiera de las modalidades del método acá divulgadas los procesos de aprendizaje automático no-supervisado es el sistema podria disminuir la necesidad que un administrador del sistema intervenga para modificar el conjunto de reglas de prioridad (106) y de esta forma el sistema podria tener mayor autonomia.

Por otra parte, haciendo a la FIG.12, la presente divulgación también se relaciona con un sistema que puede incluir un módulo de memoria (107) y una unidad de cómputo (103) que puede estar conectada al módulo de memoria (107) y configurada para ejecutar uno o más de las modalidades de los métodos anteriormente descritos.

Se entenderá en la presente divulgación por módulo de memoria (107), un elemento de hardware que incluye, pero no se limita a, memorias RAM (memoria caché, SRAM, DRAM, DDR), memoria ROM (Flash, Caché, discos duros, SSD, EPROM, EEPROM, memorias ROM extraíbles (v.g. SD (miniSD, microSD, etc), MMC (MultiMedia Card), Compact Flash, SMC (Smart Media Card), SDC (Secure Digital Card), MS (Memory Stick), entre otras)), CD-ROM, discos versátiles digitales (DVD por las siglas en inglés de Digital Versatile Disc) u otro almacenamiento óptico, casetes magnéticos, cintas magnéticas, almacenamiento o cualquier otro medio que pueda usarse para almacenar información y a la que se puede acceder por una unidad de cómputo, unidad de procesamiento, o módulo de procesamiento. En los módulos de memoria generalmente se incorporan instrucciones, estructuras de datos, módulos de programas informáticos. Algunos ejemplos de estructura de datos son: una hoja de texto o una hoja de cálculo, o una base de datos.

Particularmente, en una o varias de las modalidades de la presente divulgación puede utilizarse una base de datos para almacenar, por ejemplo, un dato de identificación (101) y un dato de solicitud de servicio (102). También, se entenderá en la presente divulgación que una base de datos es un conjunto de datos almacenados en un registro de memoria sistemáticamente para su posterior uso.

La "base de datos" se selecciona entre otras de bases de datos jerárquicas, base de datos de red, bases de datos transaccionales, bases de datos relacionales, bases de datos multidimensionales, bases de datos orientadas a objetos, bases de datos documentales,

bases de datos deductivas y otras bases de datos conocidas por una persona medianamente versada en la materia.

Por otra parte, se entenderá en la presente divulgación que la unidad de cómputo (103) es cualquier unidad de procesamiento, módulo de procesamiento, o cualquier dispositivo que procesa datos, por ejemplo, microcontroladores, micro procesadores, DSCs (Digital Signal Controller por sus siglas en inglés), FPGAs (Field Programmable Gate Array por sus siglas en inglés), CPLDs (Complex Programmable Logic Device por sus siglas en inglés), ASICs (Application Specific Integrated Circuit por sus siglas en inglés), SoCs (System on Chip por sus siglas en inglés), PSoCs (Programmable System on Chip por sus siglas en inglés), computadores, servidores, tabletas, celulares, celulares inteligentes, generadores de señales y unidades de cómputo, unidades de procesamiento o módulos de procesamiento conocidas por una persona medianamente versada en la materia y combinaciones de estas.

Una unidad de cómputo puede incluir, además, un dispositivo de visualización y/o un Dispositivo de Interfaz Humana (HID, por las siglas en inglés de Human Interface Device), puede ser o incluir una unidad de computación de propósito especial programada para ejecutar el método de esta divulgación.

Un dispositivo de visualización incluye, sin limitación, este corresponde a cualquier dispositivo que pueda conectarse a una unidad de cómputo y mostrar su salida, se selecciona entre otros de monitor CRT (por las siglas en inglés de Cathode Ray Tube), pantalla plana, pantalla de cristal líquido LCD (por las siglas en inglés de Liquid Crystal Display), pantalla LCD de matriz activa, pantalla LCD de matriz pasiva, pantallas LED, proyectores de pantallas, TV (4KTV, HDTV, TV de plasma, Smart TV), pantallas OLED (por las siglas en inglés de Organic Light Emitting Diode), pantallas AMOLED (por las siglas en inglés de Active Matrix Organic Light Emitting Diode), Pantallas de puntos cuánticos QD (por las siglas en inglés de Quantic Display), pantallas de segmentos, entre otros dispositivos capaces de mostrar datos a un usuario, conocidos por los expertos en la técnica, y combinaciones de estos.

Un dispositivo HID (por las siglas en inglés de Human Interface Device) incluye, sin limitación, teclado, mouse, trackball, touchpad, dispositivo apuntador, joystick, pantalla táctil, entre otros dispositivos capaces de permitir que un usuario ingrese datos en la unidad de cómputo del dispositivo y combinaciones de estos.

Adicionalmente, la presente divulgación se relaciona con un programa de computador que comprende instrucciones, las cuales, cuando se ejecutan en un sistema de acuerdo con cualquiera de las modalidades aquí divulgadas, causa que el sistema ejecute los pasos de cualquiera de las modalidades de los métodos acá divulgados. El programa de computador

que puede estar escrito en Java, Javascript, Perl, PHP y C++, #C, Python, SQL, Swift, Ruby, Delphi, Visual Basic, D, HTML, HTML5, CSS, y otros lenguajes de programación conocidos por una persona medianamente versada en la materia.

Además, la presente divulgación se relaciona con un medio legible por computador que comprende instrucciones, las cuales, cuando se ejecutan en un sistema de acuerdo con cualquiera de las modalidades aquí divulgadas, causa que el sistema ejecute los pasos de cualquiera de las modalidades de los métodos acá divulgados.

El medio legible por computado puede seleccionarse entre archivos ejecutables, archivos instalables, discos compactos, memorias RAM (memoria cache, SRAM, DRAM, DDR), memoria ROM (Flash, Cache, discos duros, SSD, EPROM, EEPROM, memorias ROM extraíbles (v.g. SD (miniSD, microSD, etc), MMC (MultiMedia Card), Compact Flash, SMC (Smart Media Card), SDC (Secure Digital Card), MS (Memory Stick), entre otras)), CD-ROM, discos versátiles digitales (DVD por las siglas en inglés de Digital Versatile Disc) u otro almacenamiento óptico, casetes magnéticos, cintas magnéticas, almacenamiento o cualquier otro medio que pueda usarse para almacenar información y a la que se puede acceder por una unidad de procesamiento.

El medio legible por computador puede ser un conjunto de elementos legibles por computador en los que se dividen o fraccionan instrucciones que al ser ejecutadas por uno o más servidores, o dispositivos computacionales permite llevar a cabo los pasos, etapas, subetapas de un método de acuerdo con cualquiera de las modalidades de los métodos anteriormente descritos en esta divulgación. La división o fraccionamiento de los pasos, etapas, subetapas permite que el servidor o dispositivo computacional que lea las instrucciones el medio legible por computador pueda ejecutar específicamente los pasos, etapas, subetapas que le corresponden dentro de un método de acuerdo con cualquiera de las modalidades de los métodos anteriormente descritos en esta divulgación.

EJEMPLOS

Ejemplo 1: método para reducir tiempos de procesamiento de solicitudes en líneas de espera de un banco.

Haciendo referencia a las FIG.1 y FIG. 6 se tiene un método para reducir tiempos de procesamiento de solicitudes en líneas de espera en un banco, en donde un usuario se dispone a realizar una transferencia bancaria. Como primer paso el usuario debe solicitar un digiturno de atención que corresponde a un dato de objeto (100), para ello, el usuario ingresa en un dispositivo de interfaz humana (HID) un dato de identificación (101) que en

este caso es el número de identificación personal del usuario, por ejemplo, número de cédula o número de pasaporte, además, el usuario ingresa un dato de solicitud de servicio (102), que, para nuestro ejemplo, es el tipo de transferencia que el usuario desea realizar.

El dato de objeto (100) es enviado desde el dispositivo de interfaz humana (HID) hacia un gestor de turnos, que para nuestro ejemplo particular corresponde a una unidad de cómputo (103) y el dispositivo de interfaz humana (HID) en este caso es un quiosco digital, una Tablet, o un teléfono inteligente, entre otros dispositivos electrónicos con los que puede interactuar el usuario.

Posteriormente, el gestor de turnos asigna un dato de prioridad (104) que corresponde a una prioridad de atención al usuario y está asociado al digiturno de atención. La prioridad es asignada por el gestor de turnos después de realizar un proceso de asignación de prioridad (105) que toma como entrada el número de identificación del usuario y se consulta un conjunto de reglas de prioridad (106) establecidas por el banco y que puede incluir evaluar datos del usuario, como, por ejemplo, edad del usuario, categoría del usuario según clasificación del banco, condición física o condición especial del usuario, etc. El conjunto de reglas de prioridad (106) es consultado por el gestor de turnos, es decir, por la unidad de cómputo (103) en un módulo de memoria (107) que almacena tanto las reglas de prioridad (106) como la información del usuario.

El módulo de memoria (107) también almacena una estructura de datos de prioridad (111), donde cada digiturno es asignado a un registro (112) de dicha estructura ordenada de forma jerárquica de acuerdo al dato de prioridad (104) asignado a cada digiturno, es decir, de acuerdo a la importancia establecida.

Para la atención del digiturno el banco cuenta con una pluralidad de módulos de atención de solicitudes (114) que, para nuestro ejemplo, corresponde a las cajas o los módulos de atención al usuario. El gestor de turno realiza un proceso de selección de módulo (116) para atender el digiturno, por ejemplo, selecciona el cajero del módulo más rápido o el módulo que se encuentre libre para atender el digiturno. En este caso, el módulo seleccionado corresponde a un primer módulo de procesamiento de solicitudes (115) al que luego, el gestor de turnos asigna la atención del digiturno.

El gestor de turnos realiza un monitoreo de la atención del digiturno y realiza una evaluación de dicha atención considerando un conjunto de variables de rendimiento (117), por ejemplo, el tiempo transcurrido desde que se realiza la asignación del módulo de atención (115) y es atendido el digiturno o el tiempo que se demora el cajero del módulo de atención (115) en procesar la solicitud.

Posteriormente, se realiza un proceso de calificación (119) de la atención del digiturno, en el que se obtiene un dato de eficiencia (118) con base en el conjunto de variables (117) que, como se mencionó anteriormente, están relacionadas con el tiempo de atención del digiturno. Finalmente, el gestor de turnos puede modificar las reglas de prioridad (106) almacenadas en el módulo de memoria teniendo en cuenta el dato de eficiencia (118), es decir, el cajero del módulo de atención (115) en una próxima asignación es asignado a la atención de un digiturno con un menor dato de prioridad (104) por tener un dato de eficiencia (118) menor que los cajeros de otros módulos de atención de solicitudes (114) o continuar siendo el módulo con el cajero asignado a los digiturnos con mayor prioridad por ser el módulo con mayor dato de eficiencia (118), es decir, el módulo más rápido.

Ejemplo 2: método para reducir tiempos de espera en la asignación de turnos de atención en un hospital

En un segundo ejemplo y nuevamente haciendo referencia a las FIG.1 y FIG. 6 se tiene un método para reducir tiempos de espera en la asignación de turnos de atención en un hospital, en donde un usuario requiere atención médica. Inicialmente, el usuario debe solicitar un turno de atención en un módulo de atención en el que se encuentra un operador, en el presente ejemplo, el turno de atención corresponde a un dato de objeto (100); en el módulo de atención se solicita al usuario un dato de identificación (101), que corresponde al número de identificación personal del usuario, por ejemplo, número de cédula u otro número de documento que identifique al usuario.

La información es ingresada por el operador en una unidad de cómputo (103), y, además, el operador solicita al usuario información del tipo de atención que necesita y que corresponde a un dato de solicitud de servicio (102), que, para el presente ejemplo, corresponde a una consulta general, una consulta de emergencia, una solicitud de agendamiento de una cita médica, entre otros servicios de atención. El dato de solicitud de servicio (102) es ingresado por el operador a la unidad de cómputo (103), que almacena los datos en un módulo de memoria (107).

La unidad de cómputo (103) realiza un proceso de asignación de prioridad (105) con base en el dato de identificación (101) del usuario y el dato de solicitud de servicio (102) y genera un dato de prioridad (104) asociado al turno de atención que corresponde al dato de objeto (100).

El proceso de asignación de prioridad (105) utiliza un conjunto de reglas de prioridad (106) que se encuentra almacenado en el módulo de memoria (107) y dichas reglas de prioridad (106) son definidas por el hospital, que toma como datos de entrada información como, edad del usuario, tipo de atención solicitada (consulta general, consulta de

emergencia) o información de un diagnóstico preliminar hecho por el operador del módulo de atención (v.g., triage).

En el hospital del presente ejemplo, cada cubículo de atención en el que se encuentra un médico asignado a la atención de usuarios (pacientes) corresponde a un módulo de solicitudes (114) y la unidad de cómputo (103) debe ejecutar un proceso de asignación de un médico para atender el turno de atención, este proceso de asignación de un médico corresponde a un proceso de selección de módulo (116) y una vez se realiza dicho proceso, el turno de atención es asignado al cubículo del médico seleccionado, que, corresponde a un primer módulo de procesamiento de solicitudes (115).

De forma similar al ejemplo 1, la unidad de cómputo (103) realiza una evaluación de la atención del paciente, considerando variables de rendimiento (117), por ejemplo, el tiempo transcurrido desde que se realiza la asignación del cubículo del médico seleccionado y el tiempo que transcurre desde la asignación hasta que el médico realiza la atención y diagnóstico del caso. También, se realiza un proceso de calificación (119) de la atención del paciente, que consiste en obtener un dato de eficiencia (118) con base en el conjunto de variables (117) relacionadas con la atención de una solicitud. Adicionalmente, la unidad de cómputo (103) puede modificar las reglas de prioridad (106) almacenadas en el módulo de memoria (107) teniendo como entrada el dato de eficiencia (118) relacionado a la atención brindada por el médico del cubículo que fue seleccionado para atender la solicitud y que puede considerarse en futuras asignaciones en el transcurso del día.

Ejemplo 3: método para reducir tiempos de procesamiento de un procesador informático.

En este ejemplo en particular es posible considerar el un dato de objeto (100) como un procedimiento que debe ser llevado a cabo por un procesador que cuenta con al menos dos núcleos de procesamiento de información, los cuales pueden corresponder cada uno a un módulo de solicitudes (114) y debe realizarse la selección del núcleo de procesamiento que debe ejecutar el procesamiento del dato de objeto (100), en donde el núcleo seleccionado puede corresponder a un primer módulo de procesamiento de solicitudes (115) y de forma similar a los ejemplos 1 y 2, se realizan las etapas del método descrito en la presente divulgación para realizar un proceso de selección del núcleo de procesamiento más indicado considerando datos de prioridad (104) que pueden ser establecidos según la importancia del procesamiento que debe realizarse y niveles de jerarquía de un conjunto de procedimientos definidos en el sistema informático.

En este ejemplo, un teléfono celular tiene un procesador con un núcleo con frecuencia de procesamiento nominal de 2GHz y un núcleo con frecuencia de procesamiento nominal de

2,97GHz. Cada núcleo del procesador esta configurado para desempeñarse como un módulo de solicitudes (114). Un tercer núcleo del procesador del teléfono se configura como unidad de cómputo (103) y ejecuta el método para reducir tiempos de procesamiento de solicitudes en líneas de espera cuando lee en un módulo de memoria (107) del teléfono celular (v.g., memoria ROM, memoria SD, MicroSD) un programa de computador.

En el ejemplo, cuando el tercer núcleo configurado como unidad de cómputo (103) recibe un dato de objeto (100), dicho núcleo procesa el dato de objeto (100) de acuerdo con un dato de identificación (101), que en este caso es un "header" o un "tail" del dato de objeto (100), y de acuerdo con el dato de solicitud de servicio (102).

El dato de prioridad (104) incrementa su valor cuando corresponde a un dato de objeto (100) de una aplicación de software ejecutada por el teléfono celular que tiene una pantalla de interacción desplegada y activa, o que corresponde a un dato de objeto (100) que requiere ser procesado para dar inicio a otros procesos en cola. Esta asignación del valor del dato de prioridad (104) la hace el tercer núcleo configurado como unidad de cómputo (103) mediante el proceso de asignación de prioridad (105) que toma como entrada el dato de identificación (101) y consulta un conjunto de reglas de prioridad (106) en un módulo de memoria (107) de dicho teléfono celular (v.g., memoria ROM, memoria SD, MicroSD).

En este caso, el tercer núcleo configurado como unidad de cómputo (103) modifica la estructura de datos de prioridad (111) de acuerdo con los datos de objeto (100) que se recibe, y asigna el procesamiento de cada dato de objeto (100) a los núcleos del procesador configurados como módulos de solicitudes (114).

La unidad de cómputo (103) monitorea los núcleos configurados como módulos de solicitudes (114) calculando los datos de eficiencia (118) y modifica el conjunto de reglas de prioridad (106) para optimizar el tiempo de procesamiento promedio para cada dato de objeto (100). En este caso del dato de eficiencia (118) se mide como el tiempo total que le toma a cada módulo de solicitudes (114), es decir, cada uno de los núcleos en realizar el procesamiento del dato de objeto (100) y que debe estar dentro de un rango de eficiencia definido por un conjunto de variables de rendimiento (117), que, en este ejemplo, podría ser un promedio de segundos o un rango de segundos totales y específicos que idealmente podría requerirle a cada núcleo realizar el procesamiento del dato de objeto (100).

Posteriormente, el tercer núcleo configurado como unidad de cómputo (103) podría realizar un proceso de calificación de todos los núcleos configurados como módulos de solicitudes (114) y de esta forma podría determinarse cuál de todos los núcleos es el más rápido. Finalmente, el tercer núcleo configurado como unidad de cómputo (103) podría modificar el conjunto de reglas de prioridad (106) almacenadas en el módulo de memoria (107) para

que en las próximas solicitudes se pueda asignar al módulo de solicitudes (114) más rápido el procesamiento de un dato de objeto (100) que tenga el mayor valor del dato de prioridad (104) y de esta forma incrementar la eficiencia en la atención de las solicitudes.

Ejemplo 4: método para reducir tiempos de procesamiento de un procesador informático reduciendo la posibilidad de daños por sobrecalentamiento.

Se modificó el método del ejemplo 1, de manera que ahora la unidad de cómputo (103) monitorea los núcleos configurados como módulos de solicitudes (114) calculando los datos de eficiencia (118) y tomando en consideración un parámetro indicador de sobrecarga, por ejemplo, un valor de temperatura de cada núcleo. En este caso, la unidad de cómputo (103) suspende la asignación de datos de objeto (100) al núcleo del procesador configurados como módulos de solicitudes (114) que supere una temperatura de operación crítica (v.g., de 50°C, 60°C), y reanuda la asignación de datos de objeto (100) a dicho núcleo cuando su temperatura disminuye (v.g., al 90% de la temperatura de operación crítica).

De acuerdo con lo anterior, este ejemplo permite optimizar el uso de los núcleos del procesador para agilizar el procesamiento de datos de objeto (100) y evitar deteriorar los elementos de hardware, como los mismos núcleos del procesador, módulos de memoria (107) y demás circuitos y dispositivos que puedan dañarse por sobrecalentamiento.

GLOSARIO

Unidad de cómputo (103): Una unidad de cómputo, unidad de procesamiento, o módulo de procesamiento es un dispositivo que procesa datos, por ejemplo, microcontroladores, micro procesadores, DSCs (Digital Signal Controller por sus siglas en inglés), FPGAs (Field Programmable Gate Array por sus siglas en inglés), CPLDs (Complex Programmable Logic Device por sus siglas en inglés), ASICs (Application Specific Integrated Circuit por sus siglas en inglés), SoCs (System on Chip por sus siglas en inglés), PSoCs (Programmable System on Chip por sus siglas en inglés), computadores, servidores, tabletas, celulares, celulares inteligentes, generadores de señales y unidades de cómputo, unidades de procesamiento o módulos de procesamiento conocidas por una persona medianamente versada en la materia y combinaciones de estas.

Dispositivo de Interfaz Humana (HID): Se entenderá en la presente divulgación que un Dispositivo de Interfaz Humana (HID) puede ser cualquier dispositivo capaz de permitir que un usuario ingrese datos en la unidad de cómputo del dispositivo computacional (300). Ejemplos de Dispositivos de Interfaz Humana (HID) incluyen, sin limitación, teclado, mouse, trackball, touchpad, dispositivo apuntador, joystick, pantalla táctil, micrófonos

acoplados a módulos de reconocimiento y generación de datos por voz, cámaras y otros dispositivos de captura de imagen acoplados a módulos de reconocimiento y generación por gestos, entre otros dispositivos capaces de permitir que un usuario ingrese datos en la unidad de cómputo del dispositivo y combinaciones de estos.

Módulo de memoria (107): Se entenderá en la presente divulgación por módulo de memoria (103) o registro de memoria, un elemento de hardware que incluye, pero no se limita a, memorias RAM (memoria caché, SRAM, DRAM, DDR), memoria ROM (Flash, Cache, discos duros, SSD, EPROM, EEPROM, memorias ROM extraíbles (v.g. SD (miniSD, microSD, etc), MMC (MultiMedia Card), Compact Flash, SMC (Smart Media Card), SDC (Secure Digital Card), MS (Memory Stick), entre otras)), CD-ROM, discos versátiles digitales (DVD por las siglas en inglés de Digital Versatile Disc) u otro almacenamiento óptico, casetes magnéticos, cintas magnéticas, almacenamiento o cualquier otro medio que pueda usarse para almacenar información y a la que se puede acceder por una unidad de cómputo, unidad de procesamiento, o módulo de procesamiento. En los módulos de memoria generalmente se incorporan instrucciones, estructuras de datos, módulos de programas informáticos. Algunos ejemplos de estructura de datos son: una hoja de texto o una hoja de cálculo, o una base de datos.

Dato: Se entenderá en la presente divulgación por dato a una representación simbólica que puede ser numérica, alfabética, algorítmica, lógica, y/o vectorial que codifica información.

Un dato puede tener una estructura o trama compuesta de bloques de caracteres o de bits que representan diferentes tipos de información. Cada bloque se conforma de cadenas de caracteres, números, símbolos lógicos, entre otros.

También un dato puede formase solo de bits (cadenas en lenguaje binario), formarse de caracteres formados uno a uno por una combinación de bits, formarse a partir de campos, registros o de tablas formadas de campos y registros, o formarse de archivos de intercambio de datos (formatos como csv, json, xls, entre otros). Además, un dato puede ser una matriz de n filas por m columnas. A su vez un dato puede contener varios datos.

Por ejemplo, cuando el dato tiene estructura de trama, la trama puede tener un bloque de caracteres de identificación, conocida generalmente como encabezado o *"header"*, la cual contiene información relacionada con un dispositivo de cómputo o procesador que envía el dato, y puede contener información relacionada con un dispositivo de cómputo o procesador que recibe el dato. Preferiblemente, si dato tiene un formato de trama, la trama contiene bloques relacionados con capas de acuerdo con el modelo de referencia OSI.

Asimismo, la trama puede tener un bloque de caracteres de cola (o simplemente cola), o *"tail"* en inglés, que permite identificar a una unidad de computo o servidor que es el fin del dato, es decir, que después de ese bloque ya no se encuentra información contenida en el dato identificado previamente por la unidad de computo o servidor con el *"header"*. Además, el dato tiene entre el bloque *"header"* y el bloque *"tail"* uno o más bloques de caracteres que representan estadísticas, números, descriptores, palabras, letras, valores lógicos (e.g. booleanos) y combinaciones de estos.

Base de datos: Una base de datos se define como una serie de datos organizados y relacionados entre sí, y un conjunto de programas que permitan a los usuarios acceder y modificar esos datos. Además, una base de datos puede ser una recopilación de información configurada para facilitar la recuperación, modificación, reorganización y eliminación de datos.

Servidor: Se entenderá en la presente divulgación por servidor un dispositivo que tiene una unidad de procesamiento configurada para ejecutar una serie de instrucciones correspondientes a etapas o pasos de métodos, rutinas o procesos. El servidor puede instalar y/o ejecutar un programa de computador que puede estar escrito en Java, Javascript, Perl, PHP y C++, #C, Python, SQL, Swift, Ruby, Delphi, Visual Basic, D, HTML, HTML5, CSS, y otros lenguajes de programación conocidos por una persona medianamente versada en la materia.

Además, el servidor tiene un módulo de comunicaciones que permite establecer conexión con otros servidores o dispositivos computacionales.

Adicionalmente, los servidores pueden conectarse entre sí, y conectarse con otros dispositivos computacionales a través de arquitecturas de servicios web y comunicarse por protocolos de comunicaciones como SOAP, REST, HTTP/HTML/TEXT, HMAC, HTTP/S, RPC, SP y otros protocolos de comunicaciones conocidos por una persona medianamente versada en la materia.

Similarmente, los servidores mencionados en el Capítulo Descriptivo de la presente divulgación pueden ser interconectarse a través de redes como la internet, redes VPN, redes LAN, WAN, otras redes equivalentes o similares conocidas por una persona medianamente versada en la materia y combinaciones de las mismas. Estas mismas redes pueden conectar uno o más dispositivos computacionales o terminales a uno o más servidores.

Algunos de los servidores mencionados en el Capítulo Descriptivo de la presente divulgación pueden ser servidores virtuales o servidores web.

Cualquiera de los servidores de la presente divulgación puede incluir un módulo de memoria configurado para almacenar instrucciones que al ser ejecutadas por el servidor ejecuten una parte, o la totalidad de una o más etapas de cualquiera de los métodos aquí divulgados.

En algunas modalidades de la presente divulgación, uno o más de los servidores pueden ser servidores físicos o servidores virtuales con una arquitectura de respaldo o arquitectura en clúster en la cual se tienen uno o más servidores de reemplazo configurados para para garantizar alta disponibilidad.

Usuario o cliente: Se entenderá en la presente divulgación por usuario o cliente a una persona natural o jurídica, sociedad, entidad o empresa. Particularmente, el usuario o cliente solicita un servicio de atención.

Dato de objeto (100): Se entenderá en la presente divulgación por dato de objeto (100) la información asociada por ejemplo con un cliente o usuario de un sistema, donde el dato de objeto está compuesto por un dato de identificación (101) y un dato de solicitud de servicio (102). El dato de objeto (100) permite, por ejemplo, caracterizar a un usuario y diferenciarlo completamente de otros usuarios, para brindarle la atención personalizada, dependiendo de sus necesidades y de sus características particulares, que lo hacen único en comparación con otros usuarios.

Dato de identificación (101): Se entenderá en la presente divulgación por dato de identificación (101) a un conjunto de símbolos, que permiten identificar a un dato de objeto (100) que puede solicitar el procesamiento de un dato de solicitud de servicio (102) en un sistema. Cada sistema puede establecer las reglas para asignar los datos de identificación (101) a los datos de objeto (100), por ejemplo, una entidad bancaria podría basarse en información personal de sus usuarios para asignar un dato de identificación (101) a cada usuario, que puede incluir al menos un dato seleccionado entre el nombre del usuario, el número de identificación personal (v.g. cédula de ciudadanía, ID), seguro social, la dirección de residencia actual o cualquier otro dato distintivo para garantizar que los datos de identificación (101) de cada usuario estén asociados exclusivamente a dicho usuario.

Dato de solicitud de servicio (102): Se entenderá en la presente divulgación por dato de solicitud de servicio (102) a la información concerniente al tipo de servicio requerido, dentro de una gama de servicios posibles, donde se considera la duración estimada de cada uno de los servicios. Un dato de solicitud de servicio (102) puede tener un tiempo de duración determinado, aunque dicho tiempo de duración también podría cambiar en la medida que en un sistema se realiza un procesamiento del dato de solicitud de servicio (102) de forma más eficiente.

Dato de prioridad (104): Se entenderá en la presente divulgación por dato de prioridad (104) la información que define el orden de jerarquía para la atención de un grupo de datos de objeto (100), donde se especifica el orden en que el primer dato de objeto (100) puede ser atendido, hasta el último, basándose en un conjunto de reglas de prioridad (106), las cuales pueden ser establecidas por el administrador del sistema.

Proceso de asignación de prioridad (105): Se entenderá en la presente divulgación por proceso de asignación de prioridad (105) al procedimiento mediante el cual se establece el orden de atención a un grupo de datos de objeto (100), basándose en su importancia en un sistema y en la urgencia de atención. La prioridad asignada es dinámica, dado que un administrador puede modificar los factores de asignación, dependiendo de las circunstancias.

Conjunto de reglas de prioridad (106): Se entenderá en la presente divulgación por conjunto de reglas de prioridad (106) a los lineamientos que establecen el orden de atención de un grupo de usuarios de servicios o datos de objeto (100), basándose en los criterios determinados por un administrador del sistema. Los criterios pueden ser asignados de forma manual o de forma automática y pueden tomar en cuenta registros históricos del desempeño de los diferentes prestadores de servicio, por ejemplo, módulos de solicitudes (114). Se busca optimizar determinando el orden de atención del grupo de usuarios o datos de objeto (100) más apropiado. Un ejemplo para determinar el orden de atención de un dato de objeto (100) puede ser utilizando un cálculo de prioridad compuesta que puede utilizar un valor del dato de importancia (108) y un valor del dato de urgencia (109) asociados al dato de objeto (100) para realizar, por ejemplo, una multiplicación de ambos valores y obtener un valor de prioridad compuesta asociado al dato de objeto (100). Ejemplos de criterios que pueden incluirse para establecer el orden de atención de un usuario o un dato de objeto (100) pueden ser, edad de un usuario, condición física de un usuario, grado de importancia de un usuario para una organización, pertenencia a un tipo de plan, clase, o membresía de trato preferencial y/o prioritario, urgencia del tipo de solicitud requerida por un usuario, cantidad de solicitudes requeridas por un usuario, prioridad de tipo de aplicación de software que requiere el procesamiento del dato de objeto (100), entre otros.

Dato de importancia (108): Se entenderá en la presente divulgación por dato de importancia (108) a la información, asociada con un dato de identificación (101) asociado a un dato de objeto (100) y que puede estar relacionado con el valor relativo que puede tener un dato de objeto (100) en un sistema, de manera que se pueda contar con un criterio para diseñar la estrategia de atención y prestación de servicio a dicho dato de objeto (100). Puede ser un número entero en cierta escala, o también un número real. Por simplicidad, se

recomienda el uso de números enteros, para facilitarle las operaciones a la unidad de cómputo (103).

Dato de urgencia (109): Se entenderá en la presente divulgación por dato de urgencia (109) a la información asociada a un dato de solicitud de servicio (102) que puede permitir saber que tan rápido podría atenderse un dato de objeto (100), por ejemplo, pueden ser criterios de urgencia tales como ruta crítica de un proceso relacionado a la solicitud de servicio (102), pueden deberse a un colapso o compromiso del sistema, pueden estar relacionados a solicitudes de servicio (102) cuya no ejecución comprometan la integridad del sistema o del dato de objeto (100). El sistema puede estar en capacidad de brindar atención inmediata a un dato de objeto (100) cuyo dato de urgencia (109) presente un valor máximo, sin necesidad de considerar el dato de importancia (108) asociado al dato de objeto (100), saltándose el protocolo normal de operaciones, cuando no se presenten situaciones críticas.

Estructura de datos de prioridad (110): Se entenderá en la presente divulgación por estructura de datos de prioridad (110) al arreglo de registros ordenados jerárquicamente, donde el primer registro almacena el elemento con mayor prioridad, y el orden de los registros se corresponde con el nivel de prioridad de cada uno de los elementos. Se cuenta de esta forma, con un arreglo, donde al recorrerlo se pueden localizar los elementos, de mayor a menor prioridad.

Registro de una estructura de datos de prioridad (111): Se entenderá en la presente divulgación por registro de una estructura de datos de prioridad (111) a la unidad de almacenamiento en un módulo de memoria (103) del dato de objeto (100), mediante la que se puede acceder a toda la información relevante del dato de objeto (100) para aplicar procedimientos y obtener resultados requeridos.

Pluralidad de registros ordenados (112): Se entenderá en la presente divulgación por pluralidad de registros ordenados (112) al conjunto de registros organizados según una jerarquía (113), que a su vez se establece de acuerdo al dato de prioridad (104) del dato de objeto (100), empleados en el almacenamiento de información y en los procedimientos diversos, manejados por el sistema.

Jerarquía (113): Se entenderá en la presente divulgación por jerarquía (113) al orden de importancia aplicado a un grupo de usuarios, determinando cuales van a ser atendidos en primer lugar, en segundo lugar y así sucesivamente. La jerarquía (113) viene determinada por el conjunto de reglas de prioridad (106) establecidas por el administrador del sistema.

Módulo de solicitudes (114): Se entenderá en la presente divulgación por módulo de solicitudes (114) a una unidad física, un algoritmo ejecutado en una unidad de cómputo o

módulo de cómputo, unidad de cómputo, estación de trabajo en la que se puede realizar el procesamiento de datos de solicitud de servicio (102). También se utiliza para verificar si el dato de rendimiento (134) está dentro del intervalo de criticidad. Cuando se da una notificación de sobrecarga, esta unidad podría inhabilitarse para realizar el procesamiento de un dato de solicitud de servicio (102) dentro de un periodo de tiempo preestablecido. Entre mayor sea el número de módulos de solicitudes (114), podría agilizarse el proceso de atención de un dato de solicitud de servicio (102). Un módulo de solicitudes (114) puede ser y no limitarse a un punto de atención operado por un humano en una sucursal bancaria o en un hospital, una estación de trabajo de un humano donde el humano ejecuta una o más tareas que contribuyen a procesar el dato de objeto (100). También puede ser un medio de transporte o un elemento de hardware, por ejemplo, uno de los núcleos de un procesador, un servidor informático, entre otros.

Primer módulo de procesamiento de solicitudes (115): Se entenderá en la presente divulgación por primer módulo de procesamiento de solicitudes (115) a la unidad física, algoritmo ejecutado , unidad de cómputo, estación de trabajo en la que se intervendrá un dato de solicitud de servicio (102), y puede corresponder al módulo de solicitudes (114) más conveniente para atender el dato de solicitud de servicio (102) y además, puede corresponder al módulo de solicitudes (114) al cual está asociado el dato de duración de servicio mínimo (130) con menor valor.

Proceso de selección de módulo (116): Se entenderá en la presente divulgación por proceso de selección de módulo (116) al procedimiento llevado a cabo en la unidad de cómputo (103), realizado con el fin de seleccionar un módulo de procesamiento de solicitudes (115) del dato de solicitud de servicio (102). La selección puede realizarse entre por lo menos dos módulos de solicitudes (114). Además, la selección se podría basar en criterios establecidos y acordados para mejorar la optimización de resultados y procesos.

Conjunto de variables de rendimiento (117): Se entenderá en la presente divulgación por conjunto de variables de rendimiento (117) al grupo de variables utilizados para medir el rendimiento, eficiencia, precisión, y/o eficacia del funcionamiento de los módulos de solicitudes (114), con el propósito de asignar en el futuro módulos de solicitudes (114) con mayor nivel de servicio, a usuarios o dato de objeto (100) con mayor prioridad. Este conjunto de variables puede ser establecido y puede variar en el tiempo, adaptándose a las circunstancias particulares del sistema. Una variable utilizada podría ser, por ejemplo, la cantidad de datos de solicitud de servicio (102) que son atendidos por un módulo de solicitudes (114) en un periodo de tiempo específico.

Dato de eficiencia (118): Se entenderá en la presente divulgación por dato de eficiencia (118) a la información que podría permitir evaluar el desempeño de uno o varios módulos

de solicitudes (114), de tal forma que aquellos con mejor desempeño (los más eficientes) se puedan asignar a los usuarios o datos de objeto (100) con mayor dato de prioridad (104), según la clasificación. Opcionalmente, el dato de eficiencia (118) de cada módulo de solicitudes (114) podría evaluarse para realizar un balance en la carga de los módulos de solicitudes (114), a modo de evitar que se sobrecarguen los módulos de solicitudes (114) y por ejemplo, se puedan repartir los datos de solicitud de servicio (102) entre todos o la mayoría de módulos de solicitudes (114). Ejemplos de dato de eficiencia (118) pueden ser número total de solicitudes procesadas, número de solicitudes procesadas por unidad de tiempo, calificación promedio asignada por un usuario, calificación de satisfacción de asignada por un usuario, meta de cumplimiento predefinida, cantidad de fallos de atención, porcentajes de error, entre otras. Un dato de eficiencia (118) también puede ser un dato de una sola dimensión, por ejemplo, un número real obtenido del cómputo de un polinomio de cualquier grado, en donde las variables pueden ser las variables de desempeño y los coeficientes de dicho polinomio podrían ser proporciones de ponderación o podrían ser un vector de "n" dimensiones, en donde cada dimensión podría representar un criterio y donde cada uno de los criterios podría representar una dimensión ortogonal del espacio de desempeño y de esta forma, podrían definirse diferentes tipos de desempeño.

Proceso de calificación (119): Se entenderá en la presente divulgación por proceso de calificación (119) al procedimiento de evaluación del conjunto de variables de rendimiento (117), almacenadas en el módulo de memoria (107), donde dicho conjunto de variables de rendimiento (117) son evaluadas de acuerdo al tiempo de procesamiento del dato de solicitud de servicio (102). El resultado de la evaluación del conjunto de variables de rendimiento (117) se almacena en el módulo de memoria (107), en forma de registros, de tal forma que se pueda analizar la evolución en el tiempo, y las tendencias de las diferentes variables de rendimiento, para los diferentes procesos de toma de decisiones. Ejemplos de un proceso de calificación (119) puede ser determinar cuáles son los tres módulos de solicitudes (114) más rápidos, determinar cuál es el módulo de solicitudes (114) que procesa la menor cantidad de solicitudes asociadas a datos de objeto (100), cuál es el módulo de solicitudes (114) que ha procesado el mayor número de solicitudes asociadas a datos de objeto (100), cuál es el módulo de solicitudes (114) con los mejores registros históricos para una métrica en particular, entre otros.

Estructura vectorial (120): Se entenderá en la presente divulgación por estructura vectorial (120) al arreglo que puede estar formado por registros de priorización (121), en la que puede almacenarse información concerniente a los datos de prioridad (104), por ejemplo, basándose en una matriz de una sola dimensión (conocida matemáticamente como vector).

Registro de priorización (121): Se entenderá en la presente divulgación por registro de priorización (121) a las estructuras de almacenamiento de información, en los que las posiciones dentro de los registros están asociadas con niveles de prioridad, donde, opcionalmente, al aumentar el índice de posición decrece el valor de prioridad correspondiente. Debe considerarse que los niveles de prioridad pueden cambiar en el tiempo, y cuando eso sucede se debe actualizar la organización de los elementos dentro de los registros de priorización (121).

Regla de prioridad (122): Se entenderá en la presente divulgación por regla de prioridad (122) al criterio usado para determinar la prioridad de atención de un usuario, permite establecer el orden en la jerarquía (113), para atender al grupo usuarios de un servicio. Dicha regla puede ser establecida por el administrador del sistema, y puede estar basada en la multiplicación de la prioridad y de la urgencia, donde previamente se han asignado los pesos de la prioridad y la urgencia, de acuerdo a las condiciones de operación en la prestación del servicio.

Estructura matricial (123): Se entenderá en la presente divulgación por estructura matricial (123) al arreglo constituido por uno o más vectores, donde para acceder. Cada uno de los vectores puede estar asociado con uno de los módulos de solicitudes (114), encargados de suministrar los servicios a un grupo de usuarios, y se actualiza periódicamente, para saber cuándo algunos de los módulos de solicitudes (114) se encuentran ocupados. También, la actualización, tiene en cuenta el estado (disponible o no), al realizarse servicios más rápidos de lo esperado, con lo que existe la posibilidad de atender a un nuevo usuario, con la prioridad más alta, en ese momento.

Pluralidad de vectores de procesamiento (124): Se entenderá en la presente divulgación por pluralidad de vectores de procesamiento (124) a un conjunto de arreglos asociados con un módulo de solicitudes (114), los cuales pueden contener los registros de atención de forma ordenada y se pueden asociar también a datos de duración de servicio mínimo (130). Se encuentra relacionado con el procesamiento de los datos y su evolución en el tiempo.

Pluralidad de registros de atención (125): Se entenderá en la presente divulgación por pluralidad de registros de atención (125) a las estructuras en las que se almacena el dato de objeto (100), con toda la información asociada a dicho objeto, de tal forma que se pueda acceder a ella y utilizarla en los diferentes procedimientos y funciones, empleados en el método en su totalidad. Los registros de atención (125) se podrían actualizan periódicamente, dado que el dato de objeto (100) puede cambiar en las diferentes variables asociadas con el mismo.

Dato de duración de servicio (126): Se entenderá en la presente divulgación por dato de duración de servicio (126) a la información relacionada con el tiempo empleado en la atención a un usuario, para un servicio específico dentro de la gama ofrecida. Este dato se encuentra asociado al dato de solicitud de servicio (102). En general se tiene en cuenta el promedio de la duración de servicio, pues si se consideran valores individuales, aumenta la complejidad de las operaciones requeridas.

Dato de duración de registro (128): Se entenderá en la presente divulgación por dato de duración de registro (128) a la información que puede estar asociada con cada uno de los registros de atención, su valor podría ser mayor o igual al valor del dato de duración de servicio (126), para ser tomado en cuenta.

Vector de procesamiento (129): Se entenderá en la presente divulgación por vector de procesamiento (129) al arreglo asociado con un módulo de solicitudes (114), el cual puede contener los registros de atención de forma ordenada y se puede asociar también al dato de duración de servicio mínimo (130). Se puede relacionar con el procesamiento de los datos y su evolución en el tiempo.

Dato de duración de servicio mínimo (130): Se entenderá en la presente divulgación por dato de duración de servicio mínimo (130) al valor que especifica cuál de los módulos de solicitudes (114) es el más apropiado, dado que es la opción con el menor valor de todos, con respecto al conjunto de módulos de solicitudes (114), para cierto tipo de servicio. Se puede asociar a los vectores de procesamiento y a las solicitudes de servicio. Para determinar el dato de duración de servicio mínimo (130) se pueden utilizar unos registros históricos (131) para cada módulo de solicitudes (114), comparando los resultados de cada módulo, en la atención de cierto tipo de servicio, y obteniendo un promedio de los registros históricos, de tal manera que se pueda escoger el menor valor entre los promedios de cada uno de los módulos de solicitudes (114).

Registros históricos (131): Se entenderá en la presente divulgación por registros históricos (131) al conjunto de datos que permiten seguir la evolución de parámetros de atención de un dato de solicitud de servicio (102) asociado a un dato de objeto (100), como por ejemplo, el tiempo de espera para la atención de un dato de solicitud de servicio (102) o el tiempo de atención total del dato de solicitud de servicio (102), entre otros, de manera que se cuente con un criterio para escoger el módulo de solicitudes (114) más eficiente, utilizando la información de unos registros históricos.

Dato de actualización (132): Se entenderá en la presente divulgación por dato de actualización (132) a la información asociada con el número de módulos de solicitudes (114) que se encuentran habilitados, y que puede determinar la capacidad para atender

cierto número de datos de objeto (100) en el sistema, sin afectar la calidad del servicio. Periódicamente, pueden inhabilitarse los módulos de solicitudes (114), para evitar la sobrecarga, y evitar consecuencias negativas, a mediano y largo plazo.

Dato de notificación de sobrecarga (133): Se entenderá en la presente divulgación por dato de notificación de sobrecarga (133) a la información que alerta acerca de un nivel de carga que excede los límites para un funcionamiento sin riesgo, ingresando a un rango o intervalo de criticidad, y que requiere atención inmediata para no afectar negativamente a los módulos de solicitudes (114).

Dato de rendimiento (134): Se entenderá en la presente divulgación por dato de rendimiento (134) a la información de eficiencia, relacionada con la atención de al menos un dato de solicitud de servicio (102) por parte de un primer módulo de procesamiento de solicitudes (115) previamente seleccionado. El valor puede estar en un rango preestablecido que permita realizar una comparación del rendimiento de cada módulo de solicitudes (114). Ejemplo de un dato de rendimiento (134) podría ser la duración de atención de un dato de solicitud de servicio (102) por parte de un primer módulo de procesamiento de solicitudes (115) con una duración de atención de 2 minutos y que podría estar dentro de un rango establecido por el sistema que podría variar de 1 a 5 minutos, por lo cual, el dato de rendimiento (134) podría considerarse dentro del rango de eficiencia.

Otro ejemplo de dato de rendimiento (134) y para el caso de una atención de solicitud en una sucursal bancaria podría ser el nivel de satisfacción de un cliente, donde se tiene una calificación de 4, en un rango entre 1 y 5, siendo 1 el peor nivel posible, y 5 la mejor calificación posible, de tal forma que se puede considerar el nivel de satisfacción del cliente dentro del rango permisible o aceptable.

Intervalo de criticidad (135): Se entenderá en la presente divulgación por intervalo de criticidad (135) al rango de valores del dato de rendimiento (134), de modo que cuando el dato de rendimiento (134) se encuentre en el rango, se produce un dato de notificación de sobrecarga, en forma de alerta al sistema de administración. Este rango se puede modificar como consecuencia de un cambio de módulos de solicitudes (114), o en caso de hardware con una mejora de las características de los dispositivos.

Dato de inhabilitación temporal (136): Se entenderá en la presente divulgación por dato de inhabilitación temporal (136) a la información obtenida cuando se activa una notificación de sobrecarga, de tal forma que el dato de inhabilitación temporal (136) evita que un módulo de solicitudes (114) procese un dato de solicitud de servicio (102), dentro de un periodo de tiempo establecido.

Tiempo de procesamiento (137): Se entenderá en la presente divulgación por tiempo de procesamiento (137) al lapso de tiempo transcurrido entre la llegada del dato de solicitud de servicio (102) a la unidad de cómputo (103) y la conclusión del procesamiento del dato de solicitud de servicio (102), en el primer módulo de procesamiento de solicitudes (115). Se pretende que el tiempo de procesamiento sea lo más reducido posible, agilizando de esta forma los procesos de atención de un usuario, en general.

Dato de función objetivo (138): Se entenderá en la presente divulgación por dato de función objetivo (138) a la información utilizada para minimizar el tiempo empleado para procesar un dato de objeto (100) o un conjunto de datos de objeto (100). Dicho dato de función objetivo (138) puede definir el valor que se pretende alcanzar, teniendo en cuenta las condiciones de operación y las limitaciones, dentro de las que se encuentra el sistema. Entre los ejemplos de dato de función objetivo (138) podrían incluirse tiempos de espera y de procesamiento de solicitudes de servicio, considerando las condiciones de operación y las restricciones del sistema, costos, consumo energético, satisfacción de un cliente, entre otros.

Proceso de clasificación (139): Se entenderá en la presente divulgación por proceso de clasificación (139), la acción de escoger y asignar una categoría a un dato de objeto (100) entre una pluralidad de categorías preestablecidas en el sistema, de tal forma que se pueda diferenciar a un primer dato de objeto (100) con una categoría asociada, de otros datos de objeto (100) con categorías asociadas y diferentes a la categoría del primer dato de objeto (100). El proceso de clasificación (139) podría basarse en características y propiedades únicas de cada dato de objeto (100) y cada categoría podría ser diferente una de la otra. Adicionalmente, cada categoría del sistema podría estar asociado al tipo de dato de solicitud de servicio (102) asociado a cada dato de objeto (100).

Sistema (200): Se entenderá en la presente divulgación por sistema (200) a la unidad de hardware que incluye al menos un módulo de memoria (107) y una unidad de cómputo (103), con el que se pretende disminuir los tiempos de procesamiento de solicitudes en líneas de espera, teniendo en cuenta los recursos disponibles y las restricciones de operación. El sistema (200) puede estar configurado para ejecutar aplicaciones de software para realizar múltiples operaciones y procedimientos, sobre los datos de entrada y sobre variables temporales, así como para tomar decisiones, y hacer cálculos.

Se debe entender que la presente divulgación no se halla limitada a las modalidades descritas e ilustradas, pues como será evidente para una persona versada en el arte, existen variaciones y modificaciones posibles que no se apartan del espíritu de la invención, el cual solo se encuentra definido por las siguientes reivindicaciones.

REIVINDICACIONES

1. Un método para reducir tiempos de procesamiento de solicitudes en líneas de espera, que comprende:

a) recibir en una unidad de cómputo (103) un dato de objeto (100) que incluye un dato de identificación (101) y un dato de solicitud de servicio (102);

b) obtener un dato de prioridad (104) para el dato de objeto (100) ejecutando con la unidad de cómputo (103) un proceso de asignación de prioridad (105) que toma como entrada el dato de identificación (101) y consulta un conjunto de reglas de prioridad (106), que consulta la unidad de cómputo (103) en un módulo de memoria (107);

c) almacenar mediante la unidad de cómputo (103) el dato de objeto (100) en un primer registro de una estructura de datos de prioridad (111) almacenada en el módulo de memoria (107),

donde la estructura de datos de prioridad (110) tiene una pluralidad de registros ordenados (112) de acuerdo con una jerarquía (113) determinada con base en el dato de prioridad (104) del dato de objeto (100), y

donde cada registro de estructura de datos de prioridad (110) está configurado para almacenar un dato de objeto (100);

d) seleccionar un primer módulo de procesamiento de solicitudes (115) entre al menos dos módulos de solicitudes (114) para procesar el dato de solicitud de servicio (102) al ejecutar un proceso de selección de módulo (116) en la unidad de cómputo (103);

e) enviar desde la unidad de cómputo (103) el dato de solicitud de servicio (102) del dato de objeto (100) que tenga el mayor valor del dato de prioridad (104) al primer módulo de procesamiento de solicitudes (115) seleccionado en la etapa d);

f) obtener mediante la unidad de cómputo (103) un conjunto de variables de rendimiento (117) en el módulo de memoria (107) generadas mientras el primer módulo de procesamiento de solicitudes (115) procesa el dato de solicitud de servicio (102) enviado en la etapa e);

g) obtener un dato de eficiencia (118) del procesamiento de la solicitud de servicio (102) ejecutando con la unidad de cómputo (103) un proceso de calificación (119) que toma como entrada el conjunto de variables de rendimiento (117); y

h) modificar mediante la unidad de cómputo (103) el conjunto de reglas de prioridad (106) almacenadas en el módulo de memoria (107) a partir del dato de eficiencia (118).

2. El método de la Reivindicación 1, donde la estructura de datos de prioridad (110) es una estructura vectorial (120) que contiene una pluralidad de registros de priorización (121) que tienen una jerarquía (113) en la que el registro de priorización (121) con una posición i almacena un dato de objeto (100) con mayor valor de dato de prioridad (104) que un dato de objeto (100) almacenado en una posición i+1.

3. El método de acuerdo con cualquiera de las Reivindicaciones anteriores, donde el proceso de asignación de prioridad (105) de la etapa b) comprende las subetapas:

i1) consultar mediante la unidad de cómputo (103) una de regla de prioridad (122) almacenada en el módulo de memoria (107);

i2) consultar mediante la unidad de cómputo (103) un dato de importancia (108) asociado al dato de identificación (101) y un dato de urgencia (109) asociado al dato de solicitud de servicio (102) almacenados en el módulo de memoria (107);

i3) calcular el dato de prioridad (104) con base en la regla de prioridad (122) tomando como entrada el dato de identificación (101) y el dato de solicitud de servicio (102); y

i4) asociar el dato de prioridad (104) al dato de objeto (100).

4. El método de la Reivindicación 3, en donde la regla de prioridad (122) es la siguiente relación:

valor del dato de prioridad (104) = valor del dato de importancia (108) *valor del dato de urgencia (109).

5. El método de la Reivindicación 1, donde en el proceso de selección de la etapa d) la unidad de cómputo (103) genera una estructura matricial (123) con una pluralidad de vectores de procesamiento (124),

donde cada vector de procesamiento (129) esta asociado a un módulo de procesamiento de solicitudes (114, 115) y tiene una pluralidad de registros de atención (125),

donde en cada registro de atención (125) se almacena un dato de objeto (100).

6. El método de la Reivindicación 5, donde el proceso de selección de un primer módulo de procesamiento de solicitudes (115) de la etapa d) comprende las subetapas:

j1) obtener un dato de duración de servicio (126) asociado al dato de solicitud de servicio (102);

j2) obtener un dato de duración de registro (128) asociado a cada registro de atención (125);

j3) identificar registros de atención (125) disponibles y ordenados de forma consecutiva en un vector de procesamiento (129) cuyo dato de duración de registro (128) tenga un valor igual o mayor al valor del dato de duración de servicio (126);

j4) identificar un vector de procesamiento (129) asociado con los registros de atención (125) identificados en la subetapa j3); y

j5) seleccionar el primer módulo de procesamiento de solicitudes (115), donde el primer módulo de procesamiento de solicitudes (115) está asociado al vector de procesamiento (129) identificado en la etapa j4).

7. El método de la Reivindicación 6, que además comprende entre los pasos j4) y j5) los pasos de:

k1) obtener un dato de duración de servicio mínimo (130) asociado a cada vector de procesamiento (129) y asociado al dato de solicitud de servicio (102);

en donde el dato de duración de servicio mínimo (130) se determina con base en unos registros históricos (131) almacenados en el módulo de memoria (107);

k2) comparar el dato de duración de servicio mínimo (130) asociado a cada vector de procesamiento (129);

k3) identificar el vector de procesamiento (129) que tiene el dato de duración de servicio mínimo (130) con menor valor con base a la comparación realizada en la etapa k2);

k4) seleccionar el vector de procesamiento (129) asociado a un módulo de solicitudes (114) para procesar el dato de solicitud de servicio (102) con base en la identificación realizada en la etapa k3),

donde, en la etapa j5) se selecciona como primer módulo de procesamiento de solicitudes (115) el módulo de solicitudes (114) que tiene el dato de duración de servicio mínimo (130) con menor valor.

8. El método de acuerdo con cualquiera de las reivindicaciones 6 y 7, que además comprende un paso k5) de actualizar con la unidad de cómputo (103) la estructura matricial (123) de forma periódica que comprende los subpasos de:

v1) obtener un dato de actualización (132) asociado a la cantidad de módulos de solicitudes (114) habilitados; y

v2) modificar la pluralidad de vectores de procesamiento (124) de la estructura matricial (123) con base al número de módulos de solicitudes (114) habilitados.

9. El método de la Reivindicación 8, donde el paso k5) de actualizar la estructura matricial (123) además comprende después del paso v2 los siguientes subpasos:

v3) generar con la unidad de cómputo (103) un dato de notificación de sobrecarga (133) asociado a un módulo de solicitudes (114) que tiene un dato de

rendimiento (134) con un valor que cae dentro de un intervalo de criticidad (135), de lo contrario pasar a la etapa e);

v4) generar un dato de inhabilitación temporal (136) con la unidad de cómputo (103) cuando se genera el dato de notificación de sobrecarga (133), donde el dato de inhabilitación temporal (136) impide a dicho módulo de solicitudes (114) procesar un dato de solicitud de servicio (102) durante un intervalo de tiempo predefinido.

10. El método de la Reivindicación 8, donde el paso k5) de actualizar la estructura matricial (123) además incluye un subpaso v5) de habilitar uno o más módulos de solicitudes (114) adicionales por medio de la unidad de cómputo (103) con base al número de módulos de solicitudes (114) que se encuentran inhabilitados y con base a la notificación de sobrecarga,

en donde habilitar uno o más módulos de solicitudes (114) reduce el dato de duración de servicio (126) asociado al dato de solicitud de servicio (102);

11. El método de acuerdo con cualquiera de las Reivindicaciones anteriores, donde en la etapa g) la unidad de cómputo (103) ejecuta un proceso de calificación (119) que comprende las siguientes subetapas:

g1) registrar un conjunto de variables de rendimiento (117) relacionadas con la atención del dato de solicitud de servicio (102) del dato de objeto (100) mediante la unidad de cómputo (103) en el módulo de memoria (107);

g2) evaluar el conjunto de variables de rendimiento (117) con base en un tiempo de procesamiento (137) del dato de solicitud de servicio (102),

donde el tiempo de procesamiento (137) del dato de solicitud de servicio (102) es el periodo comprendido desde la recepción del dato de solicitud de servicio (102) en la unidad de cómputo (103) hasta que el primer módulo de procesamiento de solicitudes (115) finaliza el procesamiento del dato de solicitud de servicio (102); y

g3) registrar un dato eficiencia (118) del procesamiento de la solicitud de servicio (102) en el módulo de memoria (107) con base en la evaluación del conjunto de variables de rendimiento (117).

12. El método de acuerdo con cualquiera de las Reivindicaciones anteriores, el conjunto de reglas de prioridad (106) de la etapa b) se obtiene en una etapa a1) previa a la etapa b), que incluye ejecutar con la unidad de cómputo (103) un proceso de clasificación (139) que toma como entrada una pluralidad de datos de prioridad (104) y datos de eficiencia (118) asociados a datos de objeto (100) previamente procesados por módulos de solicitudes (114), donde el proceso de clasificación (139) tiene un dato de función objetivo (138) configurado para minimizar el tiempo que tarde un dato de objeto (100) en ser procesado.

13. Un sistema (200) para reducir tiempos de procesamiento de solicitudes en líneas de espera, que comprende:

un módulo de memoria (107); y

una unidad de cómputo (103) conectada al módulo de memoria (107) y configurada para:

- recibir un dato de objeto (100) que incluye un dato de identificación (101) y un dato de solicitud de servicio (102);
- obtener un dato de prioridad (104) para el dato de objeto (100) ejecutando un proceso de asignación de prioridad (105) que toma como entrada el dato de identificación (101) y consultando un conjunto de reglas de prioridad (106) en un módulo de memoria (107);
- almacenar el dato de objeto (100) en un primer registro de una estructura de datos de prioridad (111) almacenada en el módulo de memoria (107),

donde la estructura de datos de prioridad (110) tiene una pluralidad de registros ordenados (112) de acuerdo con una jerarquía (113) determinada con base en el dato de prioridad (104) del dato de objeto (100), y

donde cada registro de estructura de datos de prioridad (110) está configurado para almacenar un dato de objeto (100);

- seleccionar un primer módulo de procesamiento de solicitudes (115) entre al menos dos módulos de solicitudes (114) para procesar el dato de solicitud de servicio (102) al ejecutar un proceso de selección de módulo (116) en la unidad de cómputo (103);
- enviar desde la unidad de cómputo (103) el dato de solicitud de servicio (102) del dato de objeto (100) que tenga el mayor valor del dato de prioridad (104) al primer módulo de procesamiento de solicitudes (115) seleccionado en la etapa d);
- obtener un conjunto de variables de rendimiento (117) en el módulo de memoria (107) generadas mientras el primer módulo de procesamiento de solicitudes (115) procesa el dato de solicitud de servicio (102) enviado en la etapa e);
- obtener un dato de eficiencia (118) del procesamiento de la solicitud de servicio (102) ejecutando un proceso de calificación (119) que toma como entrada el conjunto de variables de rendimiento (117); y
- modificar el conjunto de reglas de prioridad (106) almacenadas en el módulo de memoria (107) a partir del dato de eficiencia (118).

****

RESUMEN

La presente divulgación se relaciona con métodos y sistemas para reducir tiempos de procesamiento de solicitudes en líneas de espera en servicios de alta complejidad. Por ejemplo, la presente divulgación describe un método que puede incluir una etapa de recibir en una unidad de cómputo un dato de objeto que puede incluir un dato de identificación y un dato de solicitud de servicio, y una etapa de obtener un dato de prioridad para el dato de objeto ejecutando un proceso de asignación de prioridad que puede tomar como entrada el dato de identificación y puede consultar un conjunto de reglas de prioridad. Adicionalmente, el método puede almacenar el dato de objeto en un primer registro de una estructura de datos de prioridad, seleccionar un primer módulo de procesamiento de solicitudes entre al menos dos módulos de solicitudes para procesar el dato de solicitud de servicio al ejecutar un proceso de selección de módulo en la unidad de cómputo. También, el método puede enviar desde la unidad de cómputo el dato de solicitud de servicio del dato de objeto que puede tener el mayor valor del dato de prioridad, al primer módulo de procesamiento de solicitudes. El método puede obtener mediante la unidad de cómputo un conjunto de variables de rendimiento en el módulo de memoria, generadas mientras el primer módulo de procesamiento de solicitudes procesa el dato de solicitud de servicio. También, el método puede incluir una etapa de obtener un dato de eficiencia del procesamiento de la solicitud de servicio, ejecutando un proceso de calificación. Adicionalmente, el método puede incluir una etapa de modificar mediante la unidad de cómputo el conjunto de reglas de prioridad que pueden estar almacenadas en el módulo de memoria a partir del dato de eficiencia.

Además, la presente divulgación también describe modalidades de un sistema que puede incluir un módulo de memoria y una unidad de cómputo que puede estar conectada al módulo de memoria y configurada para ejecutar uno o más de las modalidades de los métodos anteriormente descritos.

FIGURAS

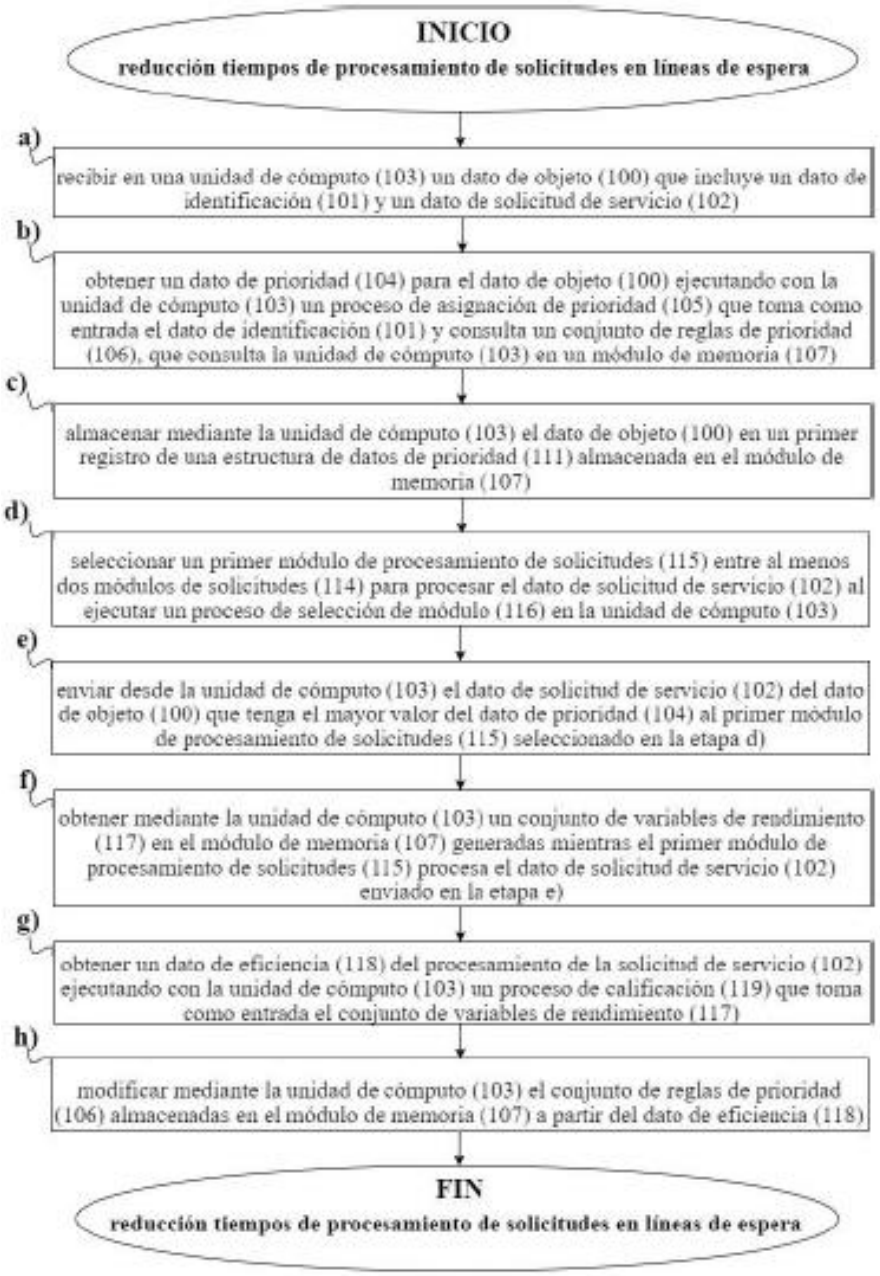

FIG. 1

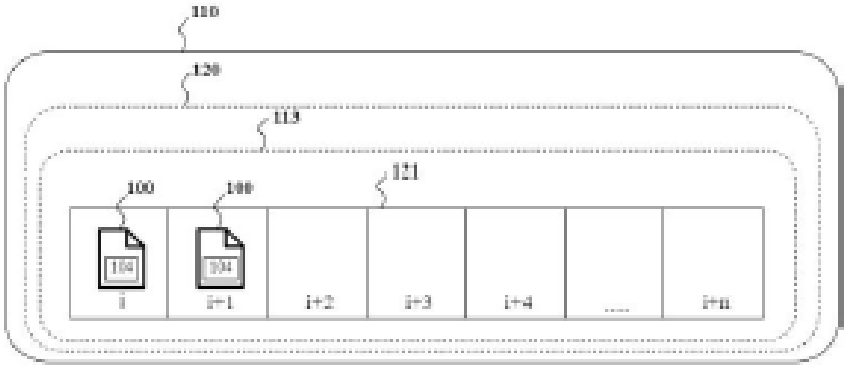

FIG. 2

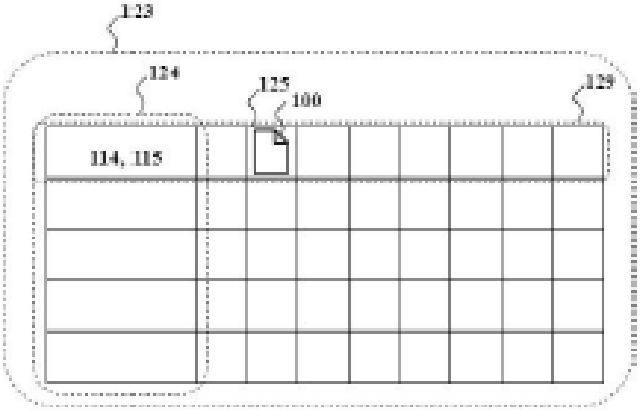

FIG. 3

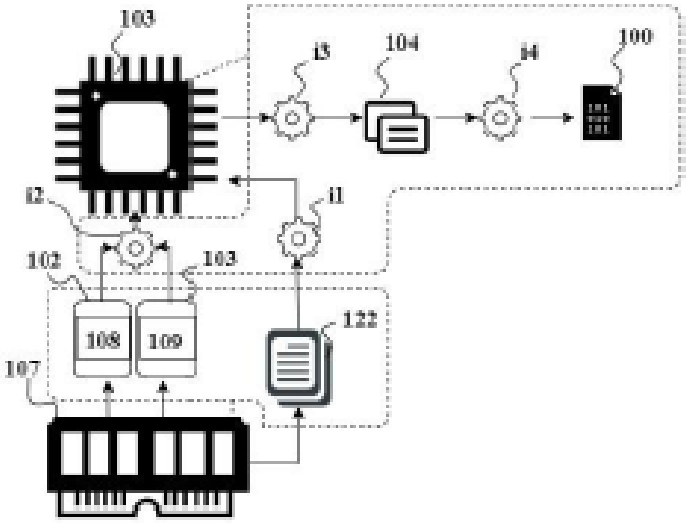

FIG. 4

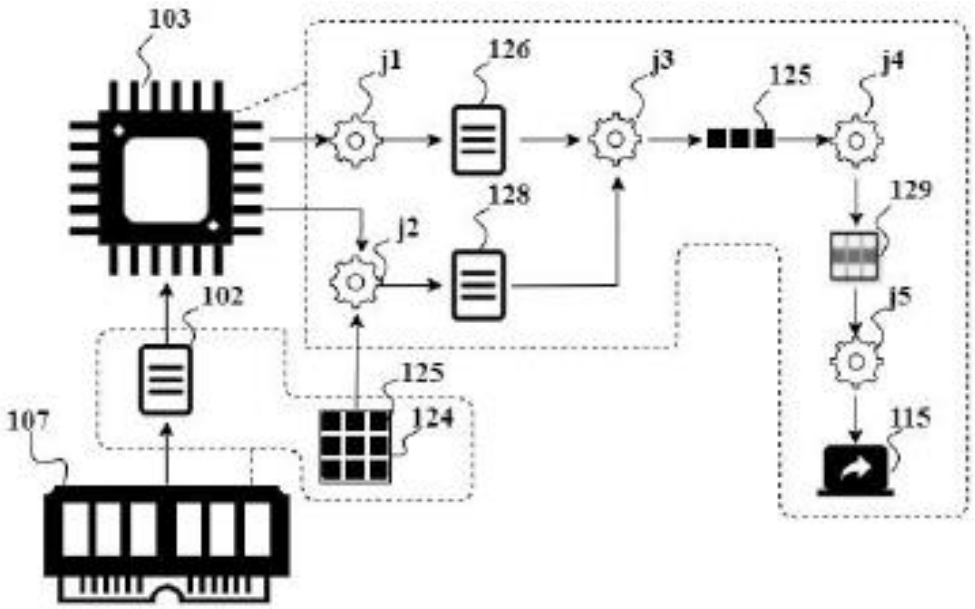
103
j1
126
j3
125
j4
129
128
j2
102
j5
125
124
107
115

FIG. 5

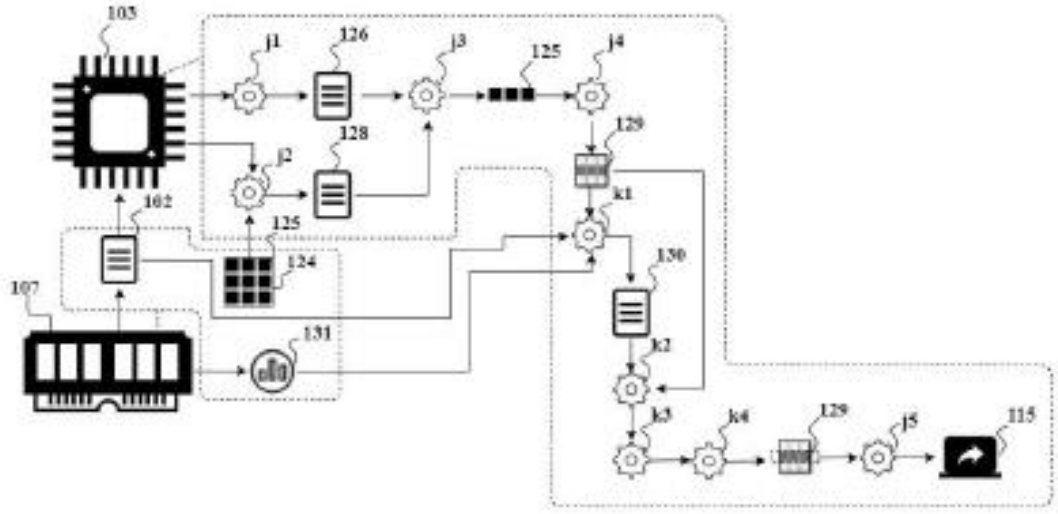
103
j1
126
j3
125
j4
129
128
j2
102
k1
125
124
130
107
131
k2
k3
k4
129
j5
115

FIG. 6

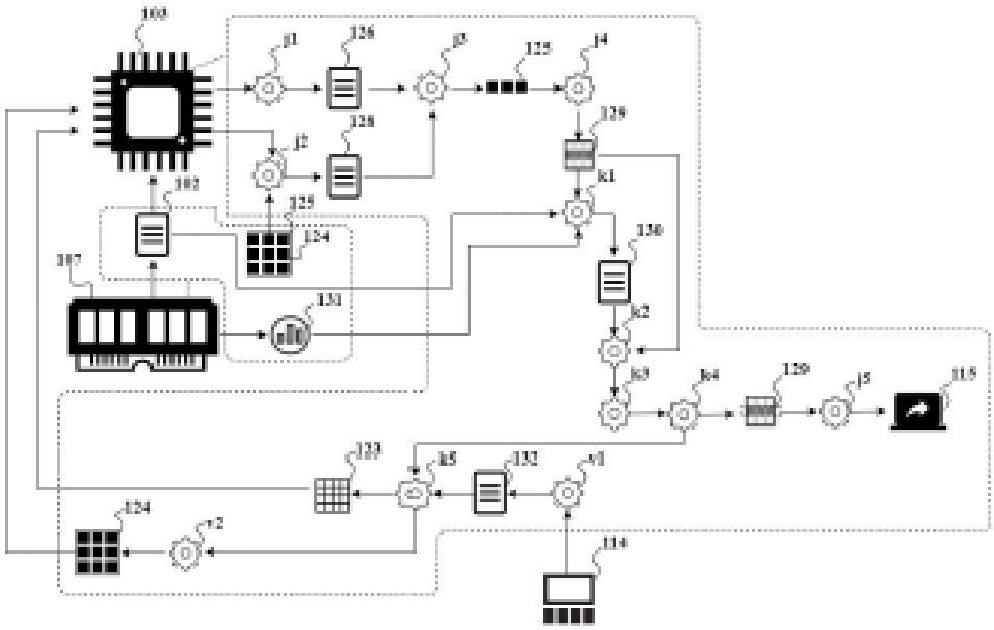

FIG. 7

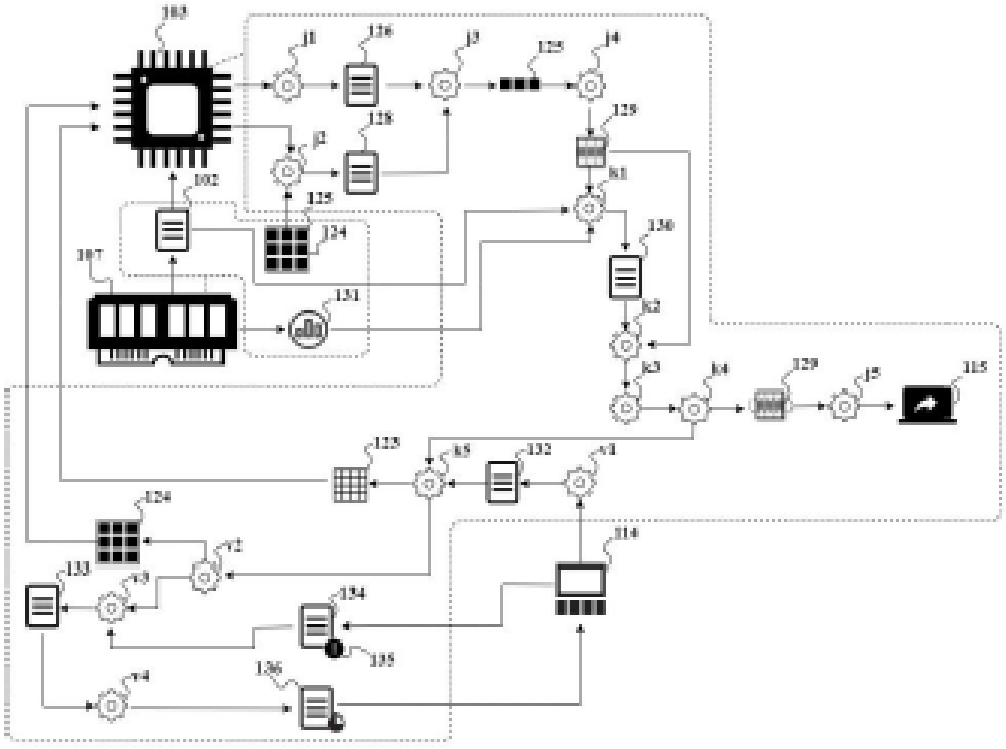

FIG. 8

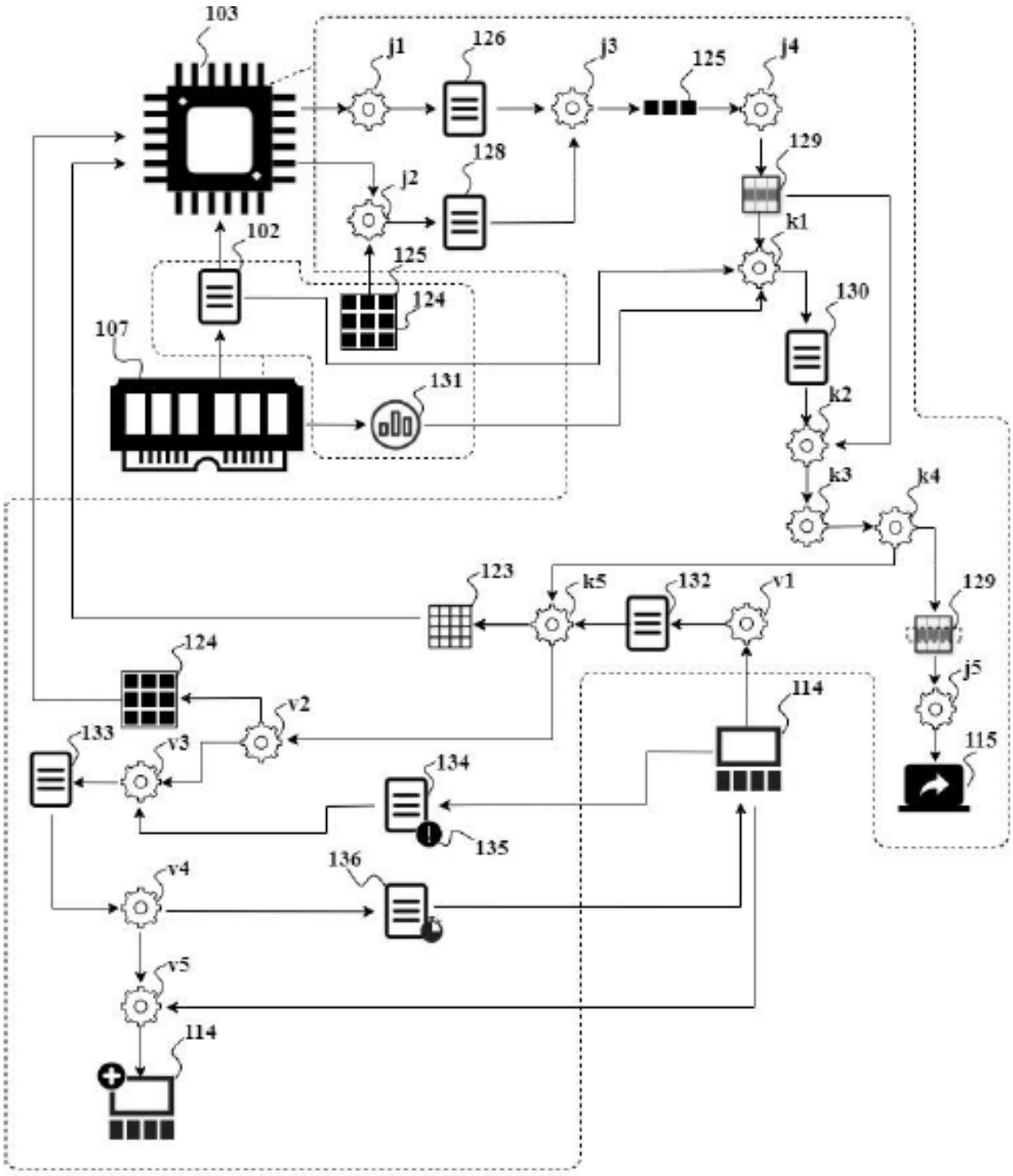
103
j1
126
j3
125
j4
129
128
j2
k1
102
125
124
130
107
131
k2
k3
k4
123
k5
132
v1
129
124
j5
v2
114
133
v3
115
134
135
136
v4
v5
114

FIG. 9

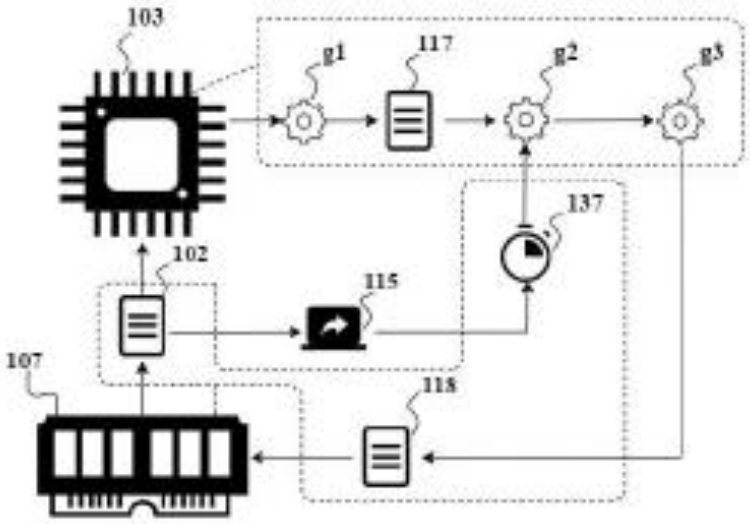
103
g1
117
g2
g3
137
102
115
107
118

FIG. 10

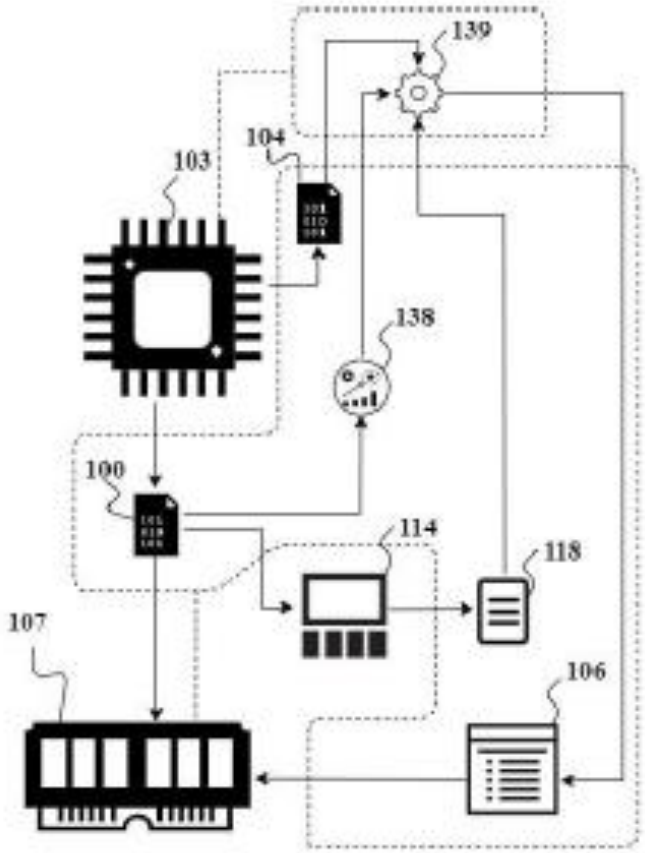
139
104
103
138
100
114
118
107
106

FIG. 11

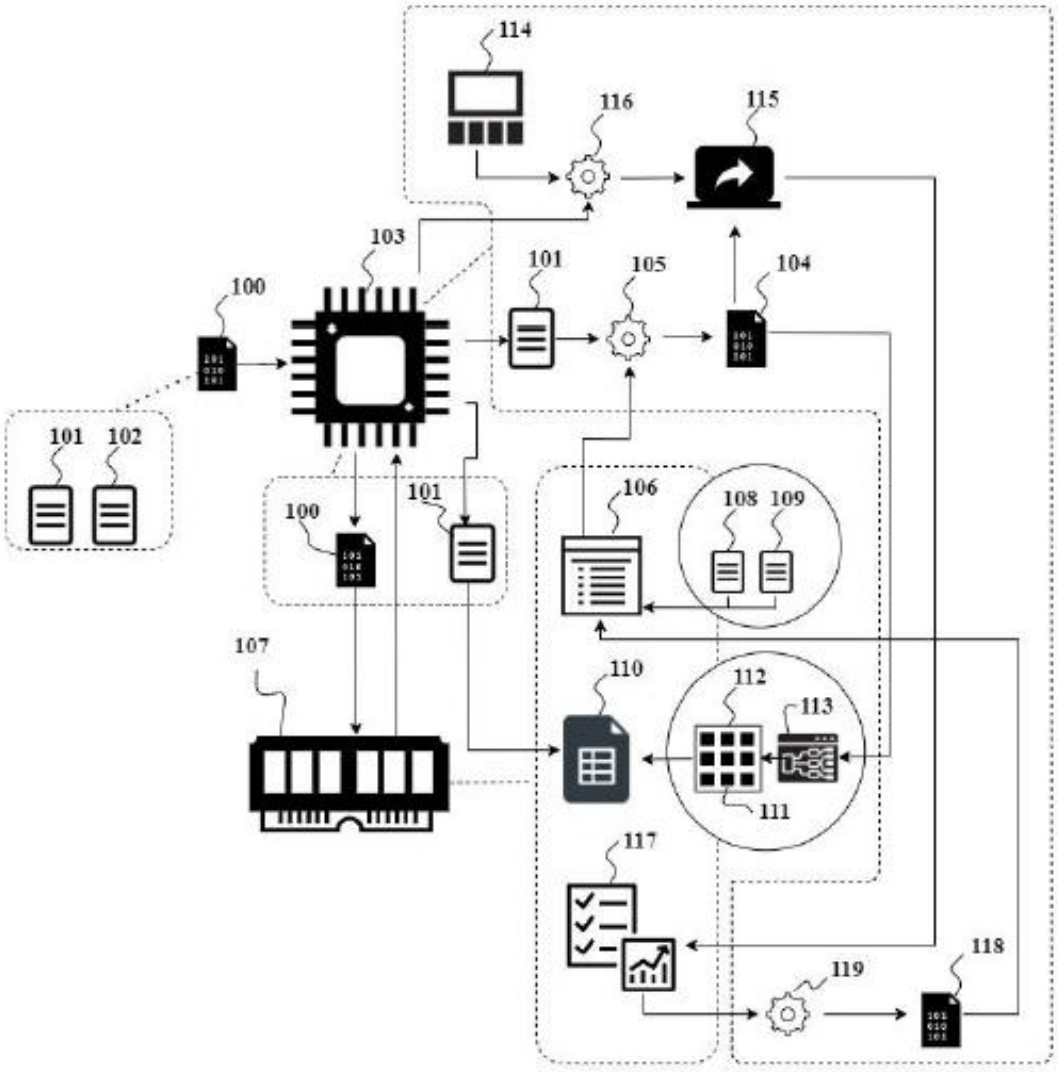
114
116
115
103
101
105
104
100
101
102
100
101
106
108
109
107
110
112
113
111
117
118
119

FIG. 12