UNIVERSIDAD NACIONAL DE CÓRDOBA

FACULTAD DE MATEMÁTICA, ASTRONOMÍA, FÍSICA Y COMPUTACIÓN

Enrutamiento Multiobjetivo en Redes Tolerantes a Demoras

Benjamin Maximilinano Martinez Picech

Trabajo Especial

Directores

Dr. Pedro R. D'Argenio

Dr. Juan A. Fraire

Abstract

Las redes tolerantes a demoras (DTN) han sido propuestas y estudiadas como un protocolo acorde al problema de la comunicación en redes que cambian a lo largo del tiempo. Un plan de contacto se construye a partir de la información empírica previamente obtenida de la topología de la red para proponer una tabla de enrutamiento acorde. En las constelaciones de satélites en la orbita terrestre baja (LEO) estos planes de contacto tienden a no ser fiables por causas como el conocimiento impreciso de las posiciones, velocidades y orientación (apuntamiento) de los satélites, nodos defectuosos, y cambios impredecibles en las condiciones ambientales que pueden llevar a interferencia y perdida de paquetes. Para resolver el enrutamiento en DTNs con planes de contacto inciertos los procesos de decisión de Markov (MDP) han sido utilizados para obtener planes de enrutamiento que garanticen una buena probabilidad de éxito en la transmisión de paquetes. Este trabajo profundiza este enfoque incorporando variables de decisión sobre el consumo de energía y latencias de los contactos en el cálculo de los planes de enrutamiento. Utilizando como base el algoritmo RUCoP (Routing under Uncertain Contact Plans) se desarrolla un algoritmo multi objetivo para intentar optimizar los objetivos de minimización de la probabilidad de fallo, latencia y consumo de energía.

Delay tolerant networks (DTN) have been proposed and studied as an architecture fit for the problem of comunication in networks changing in time. A contact plan is built from the empirical information previously obtained from the network topology to propose a corresponding routing table. Networks as the low-Earth orbit (LEO) constellations are of interest for the world, in these networks uncertainty in contact plans can arise from various sources; such as imprecise knowledge of the satellite positions, velocities and orientation (pointing), fault nodes and unpredictable changes in the atmospheric conditions. To solve routing in DTNs under uncertain contact plans Markov decision processes (MDP) have been studied and implemented to find schedulers with a good success delivery probability (SDP). This thesis analyzes further the inclusion of desition variables in the contacts in order to include energy consumption and delivery time in the routing calculus. Based on the RUCoP (Routing under Uncertain Contact Plans) algorithm a new development is introduced by applying multi-objective optimization looking after the minimization of the failure probability, delay and energy consumption.

Índice

I. Introducción

El desafío de desplegar redes como las satelitales en la orbita terrestre baja (LEO, low earth orbit), redes vehiculares, redes de sensores móviles, y otras. nos enfrenta a redes particularmente complicadas por su entorno. Las interferencias en la red, la escasez de opciones de enrutamiento y los tiempos excesivos de round-trip hacen que estas redes no puedan considerar protocolos de comunicación que asuman una conexión confiable y persistente en el tiempo como lo es TCP. Para subsanar este tipo de problemas, mega-constelaciones como Starlink (Foust 2019) pueden desplegar hasta 12.000 satélites para garantizar conectividad end-to-end de sus nodos, disminuir latencias y obtener así acceso en tiempo real a los datos de cada nodo participe de la red. Sin embargo, existen alternativas más sustentables que se pueden implementar cuando el acceso a los datos tiene tolerancia a una cierta demora. Las redes tolerantes a demoras (DTN, delay tolerant networks) (Fall 2003) fueron propuestas en 2003 como una alternativa que se vale de la posibilidad de almacenamiento persistente de los nodos para conservar los paquetes en espera de una opción de enrutamiento futura. Para proveer funcionalidades extra al protocolo de internet, se presento en (Scott & Burleigh 2007) el protocolo de paquete (BP, bundle protocol) con la posibilidad de agregar una capa por encima de capas especificas para cada familia de redes. Esto posibilita que los nodos tengan la custodia de los paquetes, opciones de retransmisión y la capacidad de utilizar las conexiones casuales o predichas por los planes de enrutamiento. Además, tanto la arquitectura DTN y el protocolo BP actualmente son estandares del IETF (Internet Engineering Task Force) y el CCSDS (Consultative Committee for Space Data Systems). A partir del protocolo de paquete se pueden abordar los problemas intrínsecos de las redes tolerantes a demora.

1. Problema

A partir de experimentación empírica o calculando los movimientos preestablecidos de los nodos que conforman la red, es posible obtener un plan de contacto etiquetado con las probabilidades de fallo de cada contacto. Estos planes describen la conectividad de la red a lo largo del tiempo, como en la figura 1 donde podemos ver en el eje vertical los nodos y en el horizontal el tiempo. Este formato de plan de contacto es el mas simple posible ya que asumimos que todos los contactos tienen la misma posibilidad de fallo o no se consideran dichas probabilidades. En la figura se puede notar además que el tiempo se considera como discreto y dividido en slots, dado que dependiendo del tipo de red a tratar los espacios temporales en los que existe un contacto entre dos nodos puede ser muy variable.

Lo que buscamos obtener es un plan de enrutamiento que ataque todos los objetivos que mencionamos anteriormente, pero particularmente que maximice la probabilidad de entregar un paquete (SDP). La mayoría de ideas que surgen como solución a estos problemas se enfocan únicamente en el SDP. El plan de enrutamiento debe marcar la decisión a tomar desde cada nodo en cada momento hacia cualquier nodo objetivo. La solución al enrutamiento en tales redes es una pregunta abierta de investigación (D'Argenio et al. 2020).

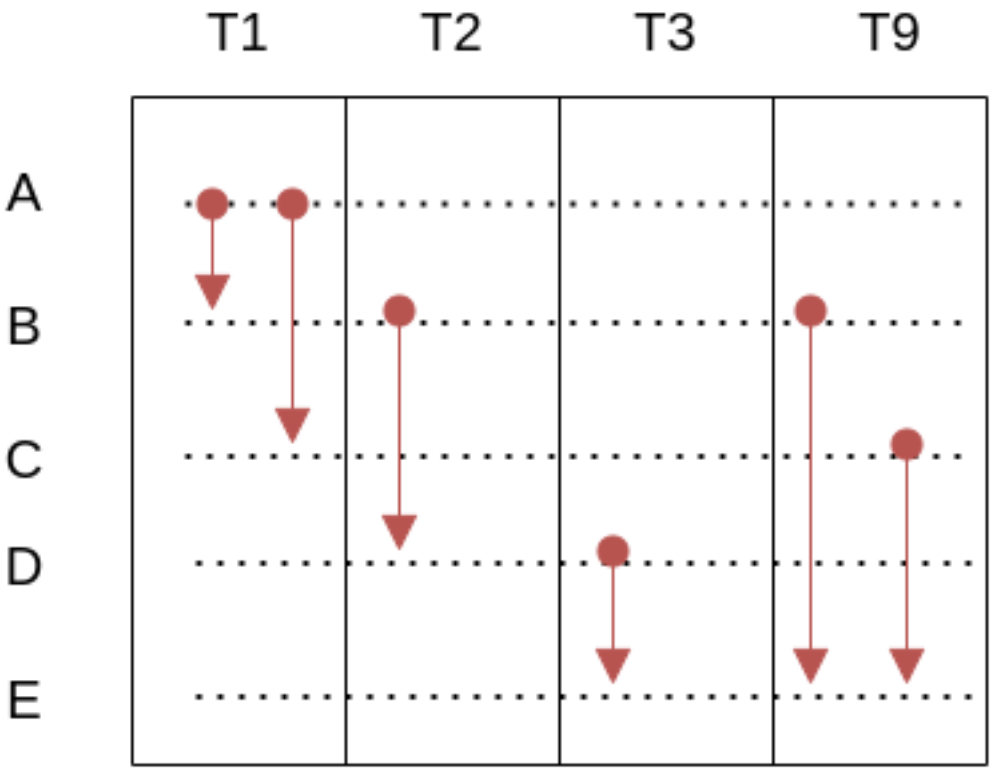


Figura 1: Plan de contacto Simple

2. Comparación de soluciones

El problema de enrutamiento en planes de contacto inciertos coincide muy bien con las capacidades de modelado de los procesos de decisión de Markov (MDP, Markov decision process) combinando transiciones probabilísticas discretas, que representan fallas en el contacto; con transiciones no deterministas, que representan opciones de enrutamiento. Esta idea se realizó de forma prototípica en (Raverta et al. 2018), se elaboró profundamente en (Raverta et al. 2021) y se estudió comparativamente en (D'Argenio et al. 2020). Utilizar MDPs considerando estados tiene una limitación en la búsqueda de las soluciones, se considera un

estado global del sistema probabilístico que resulta en un problema al buscar soluciones en las que la red admite mas de una copia del mismo paquete. Ya que en la practica, un nodo al momento de tomar una decisión de enrutamiento solo puede conocer la información local a el y no el estado general de la red. Por lo que la implementación de estas soluciones esta limitada para obtener planes de enrutamiento factibles en una red real que permitan mas de una copia por paquete enviado.

PMC

Dado un MDP podemos aplicar verificación de modelos probabilísticos (PMC, por sus siglas en ingles) para determinar la estrategia de enrutamiento con mayor probabilidad de entrega (SDP, success delivery probability). Las investigaciones mencionadas hasta ahora se enfocan en maximizar la probabilidad de éxito. Otros objetivos como el tiempo de entrega y el consumo de energía han sido puestos en consideración en (Torrella 2023). Como ya se mencionó, limitando a la red a contener solo una copia de cada paquete a enviar, pero logrando tomar en valor objetivos que pueden variar su importancia circunstancialmente. Así abrimos la puerta a el uso eficiente de redes de menor densidad de manera circunstancial, el tiempo de entrega resulta significativo para comunicación de control mientras que el consumo de energía es vital en estaciones espaciales donde los recursos energéticos no son abundantes. El análisis multi objetivo permite priorizar la elección de enrutamiento a partir de características múltiples de la red.

CGR

CGR (enrutamiento en grafos de contacto) (Araniti et al. 2015) es un protocolo que construye grafos acíclicos y dirigidos a partir del plan de contacto y lo sujeta a algoritmos de búsqueda del tipo Dijkstra, obteniendo como resultado un plan de enrutamiento estático que describe las rutas a tomar para cada intención de envío. Como no se considera la posibilidad de que un determinado contacto falle, la solución de esta técnica tiene complicaciones cuando los contactos difieren del plan original, ya sea por fallas o interrupciones en la transmisión, o por conocimiento incompleto/inexacto cuando se calculó el plan, las DTNs espaciales son propensas a estos problemas.

RUCoP

Algoritmo que se basa en procesos de decisión de Markov permitiendo la derivación de enrutamientos para múltiples copias de paquetes. Diseñado específicamente para tratar el problema en DTNs con planes de contacto inciertos. Además, dado el contexto de estas redes introduce la posibilidad de reruteo de los paquetes cuando uno de los contactos del plan falla, considerando el tiempo de detección del fallo. Pero también tiene la limitación para el caso de tratar con redes que aceptan mas de 1 copia de un mismo paquete para su envío, de que el estado de la red se lo considera como global, pudiendo resultar en soluciones que no son implementables en la realidad. Pero a pesar de esta limitación el algoritmo nos establece un limite teórico alcanzable por cualquier enrutamiento posible, es decir que no existirá un enrutamiento superior a RUCoP, si lo que buscamos es maximizar la probabilidad que un paquete llegue a destino.

3. Contribución

Con el desarrollo del algoritmo MORUCoP (Multiple Objective Routing Under Uncertain Contact Plans) se busca continuar con el enfoque introducido en (Torrella 2023). Si bien no se utilizaran modelos probabilísticos, partiendo de la base del algoritmo de RUCoP, le sumaremos una mayor versatilidad a la solución a partir de la consideración de las métricas de energía y delay en las transmisiones. Además, a diferencia del resto de soluciones, se agrega la posibilidad de obtener resultados implementables en redes que posibilitan múltiples copias para un mismo paquete. Es decir, que el algoritmo considera las falencias que se mencionaron anteriormente sobre las soluciones que consideran un estado global para derivar sus enrutamientos.
También se aporta la implementación de MORUCoP en un repositorio Python (`https://github.com/Benja272/dtn_routing`) con una integración para ejecutar simulaciones con Omnet++, utilizando la herramienta de simulación sobre una amplia gama de casos de estudio. Dentro del mismo repositorio se habilita una funcionalidad para la obtención de gráficos representativos de los resultados obtenidos en la simulación. También se agregaron utilidades para la obtención de las tablas de enrutamiento de los algoritmos MORUCoP y RUCoP.

4. Suposiciones

Para la implementación y desarrollo del algoritmo MORUCoP fue necesario tomar algunas suposiciones sobre el protocolo de comunicación y topología de las redes a considerar.

Como ya se mencionó arriba, consideramos el tiempo dividido en slots temporales y consideramos el tamaño de estos como el mínimo intervalo de tiempo que puede durar un contacto. Además, los envíos de paquetes utilizando un contacto necesariamente demoran como mínimo 1 slot de tiempo en completarse, evitando de esta forma obtener ciclos de conexión en un mismo slot. También se asumirá la existencia de ACK (acknowledgement) de confirmación de que un paquete llego al destino parcial de cada contacto, se asume que la latencia (delay) de enviar un paquete por un contacto es igual al tiempo de reconocimiento de perdida del paquete. Una vez enterado el nodo fuente de un contacto de la perdida, este paquete vuelve a estar disponible para ser enviado a otro nodo. De esta forma la decisión a tomar cuando un contacto falla y resulta en la perdida de un paquete, dependerá del plan de enrutamiento en el slot de reconocimiento de la perdida desde el nodo fuente del contacto hacia el nodo objetivo final.

5. Estructura de la tesis

Inicialmente introduciremos la teoría y conceptos fundamentales para el desarrollo del trabajo, algunos de ellos con ejemplos gráficos para facilitar el entendimiento del lector. Luego, se explicara el algoritmo MORUCoP, enfocándonos particularmente en el pseudocódigo y el cálculos de los costos que se definirán formalmente a través de ecuaciones que especifican cada uno de los costos en probabilidad de fallo, energía y delay. En la ultima parte de la sección del Algoritmo se ejemplificará la utilidad del análisis multiobjetivo con un ejemplo teórico. Lo siguiente será la experimentación, explicaremos los distintos escenarios de simulación elegidos así como los resultados obtenidos para cada uno de ellos con diferentes prioridades y capacidades de copias por paquete. Por último se analizarán las posibles mejoras al algoritmo así como las limitaciones del mismo.

II. Teoría

1. Planes de Contacto

Las DTNs fueron introducidas para designar a aquellas redes cuya tecnología y arquitectura las convierten en redes que evolucionan en el tiempo con conexiones que carecen de persistencia y seguridad. Es decir, redes en las que la conexiones entre los nodos que son necesarias para conectar cualquiera de ellos con otro no están garantizadas para un momento de tiempo en especifico (Torgerson et al. 2007). Como se mencionó, los problemas inherentes a dichas redes son atacados posicionando una capa de paquete (Scott & Burleigh 2007) por encima de otras capas superficiales al protocolo de comunicación a utilizar, como bien puede ser la capa de aplicación de internet. Permitiendo el almacenamiento persistente de los paquetes, esta capa decide a partir de un plan de enrutamiento si conservar un paquete para una oportunidad de transmisión futura, o utilizar ventanas de comunicación abiertas en el momento.

Un **contacto** esta definido como una ventana de comunicación que permite el envío de información de un nodo fuente a un nodo destino durante un determinado intervalo de tiempo, y es en base a una lista de contactos que esta conformado el plan de contacto. En trabajos previos se puede encontrar una clasificación de los distintos tipos de contactos que pueden conformar una DTN;

Programadas: cuando los contactos pueden ser previstos con exactitud. Este conocimiento se puede usar para estrategias de enrutamiento y de un mejor aprovechamiento de los recursos (Fraire & Finochietto 2015), son aplicables algoritmos como CGR (Fraire et al. 2021)

Probabilísticas: los contactos son inferidos dinámicamente según la evolución de la red. El enrutamiento se basa en las métricas probabilísticas de que exista un contacto en el futuro y se utilizan múltiples copias por distintos caminos para incrementar la probabilidad de entrega. (Feldmann & Walter 2017)

Oportunista: no existe ningún conocimiento de los contactos, se utilizan los contactos según exista la oportunidad. Estrategias de enrutamiento triviales como Spray-and-Wait (Spyropoulos et al. 2005) o por inundación han sido aplicadas en estos casos.

Programadas con incertidumbre: introducidas en (Raverta et al. 2021) como DTNs donde la materialización de los contactos puede diferir del plan original según una probabilidad conocida a partir de experimentación. Estas últimas son las que consideramos apropiadas para modelar redes espaciales en orbita baja, ya que estas sufren de fallas predecibles en sus contactos debido a causas como interferencias, conocimiento impreciso de las orbitas entre

otras circunstancias propias del entorno. Así una DTN incierta, a diferencia de una DTN programada, puede funcionar en casos de comunicaciones propensas a falla; y, a diferencia de una DTN probabilista, puede aprovechar información de probabilidades precalculadas. En el ejemplo particular de las redes satelitales, a partir de información orbital y de la tecnología que albergan los satélites podemos obtener previamente para cada contacto la probabilidad de fallo, el delay y la energía necesarios para utilizar determinada comunicación.

2. Proceso de Decisión de Markov (MDP)

Un proceso de decisión de Markov es una estructura matemática que permite el modelado de sistemas de tiempo discreto cuyo comportamiento es no determinístico y probabilístico (Filar & Vrieze 2012). Los MDP proveen un contexto teórico del modelado de sistemas con incertidumbre cuantificable a través de probabilidades e incertidumbre no cuantificable que se plasma a través de las posibles decisiones.

Un MDP $\mathbf{M}$ se define como una tupla (S, A, P, E, D, s_0) donde:

- S es un conjunto finito de estados.
- A es un conjunto finito de acciones.
- P: $S \times A \times S \to [0,1]$ es una función de probabilidad tal que $\sum_{s_i \in S} \mathbf{P}(s,a,s_i) \in \{0,1\}, \forall_{s,a} | s \in S \wedge a \in A$.
- E: $S \times A \times S \to \mathbb{Q}$ es una función de costos de energía.
- D: $S \times A \times S \to \mathbb{Q}$ es una función de costos de demora o delay.
- $s_0 \in S$ es el estado inicial del sistema.

Si $\sum_{s_i \in S} \mathbf{P}(s,a,s_i) = 1$ significa que la acción a esta habilitada en el estado s. La importancia de los distintos objetivos de optimización varia con las diferencias en las topologías de las redes. Se busca minimizar la probabilidad de fallo, la energía consumida y el tiempo transcurrido en la entrega de los paquetes. Tomaremos como ejemplo las prioridades $SDP > Energy > Delay$ para explicar como obtenemos la política correspondiente.

El comportamiento de un MDP $\mathbb{M}$ funciona como una ejecución que empieza desde el estado s_0. Luego consecutivamente para cada estado s_i una de las acciones habilitadas es elegida para romper con el no determinismo en dicho estado. Luego, al ser un modelo probabilístico, el siguiente estado es tomado aleatoriamente siguiendo la distribución de probabilidad de $\mathbf{P}(s,a,\cdot)$.

De esta forma, dependiendo del resultado deseado, lo que buscamos obtener es una política que minimice los costos obtenidos cumpliendo con la propiedad de que se llegue a uno de los estados objetivos. Una política es una función $\pi : S \to A$ que resuelve todo no determinismo posible en $\mathbb{M}$. Considerando los objetivos mencionados y un conjunto objetivo de estados $B \subseteq S$, buscamos obtener la política que minimice los costos siguiendo las prioridades predefinidas. Para conocer los costos y probabilidades de cada estado s definimos una variable x_s que representa dicha información en forma de tupla y también un conjunto $S_{=0}$ representando los estados con probabilidad 0 de llegar a un estado objetivo. Buscamos la solución vectorial $(x_s)_{s \in S}$ del siguiente sistema de ecuaciones, utilizando la sumatoria punto a punto de vectores y el mínimo considerando la comparación lexicográfica de vectores, es decir considerando a los vectores como palabras que se comparan punto a punto hasta obtener que uno es mayor que el otro o que son iguales.

$$x_s = (0,0,0) \qquad \text{si } s \in B$$
$$x_s = (1,0,0) \qquad \text{si } s \in S_{=0}$$

Luego si

$$s \in S \setminus (S_{=0} \cup B)$$
$$x_s = \min_{a \in A(s)} \sum_{t \in S} (Fp(s,a,t), Energy(s,a,t), Delay(s,a,t))$$

Con:

$$Fp(s,a,t) = \mathbf{P}(s,a,t) \cdot x_t^P$$
$$Energy(s,a,t) = \mathbf{P}(s,a,t) \cdot (\mathbf{E}(s,a,t) + x_t^E)$$
$$Delay(s,a,t) = \begin{cases} 0 & si \quad t \in S_{=0} \\ \frac{\mathbf{P}(s,a,t)}{\sum_{s' \in (S \setminus S_{=0})} \mathbf{P}(s,a,s')} \cdot (\mathbf{D}(s,a,t) + x_t^D) & si \quad t \notin S_{=0} \end{cases}$$

Obtenemos la política minimizadora de costos como:

$$\pi^{min}(s) = \underset{a \in A(s)}{argmin} \sum_{t \in S} (Fp(s,a,t), Energy(s,a,t), Delay(s,a,t))$$

Los estados que pertenecen al conjunto $S_{=0}$ por definición tienen probabilidad de fallar igual a 1 ya que el paquete nunca podrá ser enviado exitosamente desde dichos estados. Análogamente los estados en B no pueden fallar porque el objetivo ya fue alcanzado.

3. Costos

La noción de costos toma importancia al momento de tomar el mejor camino posible para un paquete. Dependiendo de la cantidad de copias disponibles en la red así como de el plan de contacto los costos de tomar una decisión u otra pueden variar. Buscamos minimizar el costo general del envío de los paquetes, pensando a estos como un vector con **(Probabilidad de fallo, Energía, Delay)**, dada una política predefinida que resuelva los no determinismos definimos.

Probabilidad de fallo: La probabilidad de que desde un estado fuente en un slot particular de tiempo se llegue a un estado de fallo que sea irrecuperable, a partir del cual no pueda alcanzarse un estado objetivo.

Energía: La energía total consumida desde el estado fuente hacia el objetivo considerando también la que se pierde en caso de llegar a un estado de fallo, depende de la función de costos energéticos antes mencionada.

Delay: El tiempo total necesario para alcanzar algún estado objetivo, para esta métrica los posibles estados de fallo que se pueden alcanzar son indiferentes.
En la figura 2 podemos ver un ejemplo del plan de contacto de una red en un formato diferente al que vimos en la introducción y además se calculan los costos de dos caminos diferentes que pueden tomar los paquetes enviados desde el nodo **A** al nodo **E** considerando un camino con posibilidad de rerouting y uno que no tiene esa posibilidad. Podemos notar en los cálculos de la figura que se calcula la probabilidad de éxito o SDP en lugar de la probabilidad de fallo ya que $SDP = 1 - Pf$.

4. Árbol de transiciones

Basándonos en los procesos de decisión de Markov podemos construir un árbol que contenga la información de todas las decisiones posibles a tomar y en que estado resultaría cada una, de esta forma podemos pensar en estrategias para obtener buenas políticas de enrutamiento dependiendo de cuales objetivos prioricemos. Para poder entender como se construye y como se codifica la información en el árbol introduciremos las definiciones pertinentes a nuestro caso de uso particular.

Estados

Cada estado contiene información de la cantidad de copias presentes en cada nodo de la red en un slot de tiempo determinado. Además a medida que nuevos estados se agregan al

árbol, se obtendrán los costos y probabilidades del mismo.

Transiciones

Estarán compuestas por dos tipos distintos de decisiones que se pueden tomar. La primera serán las *transiciones de transmisión* que implica una transmisión no deterministica utilizando un contacto disponible para alguno de los nodos custodios de paquetes. El segundo tipo serán las *transiciones de guardado* que mencionaremos también como transiciones *next* simbolizando la decisión de un nodo custodio de guardar el paquete en memoria para utilizar una ventana de transmisión posterior.

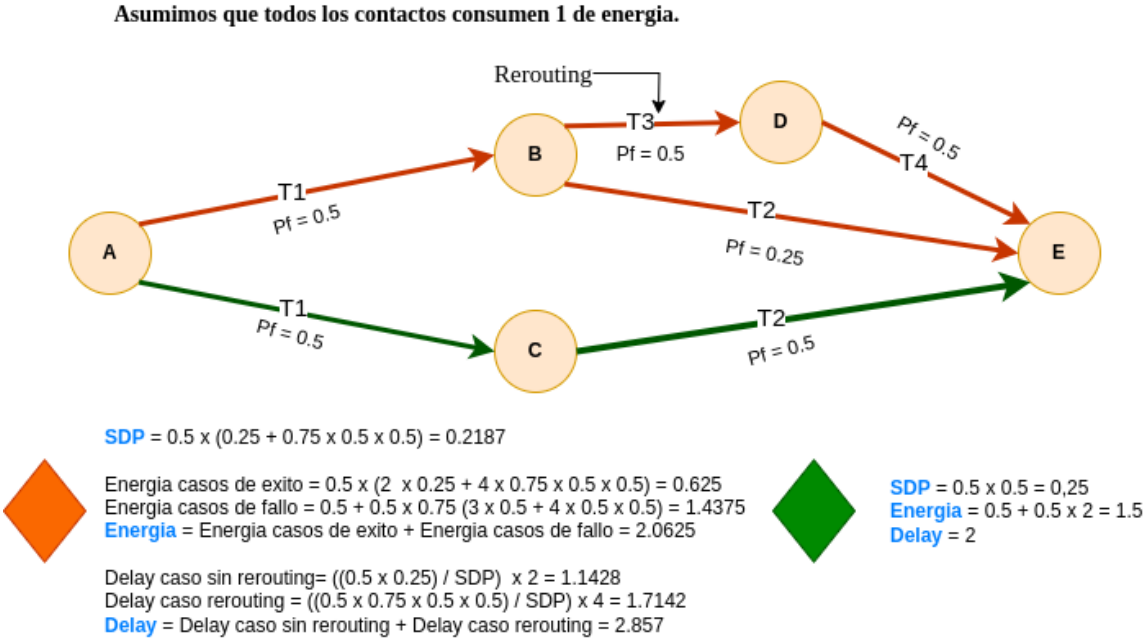


Figura 2: Grafo de Plan de contacto con cálculo de costos.

5. Soluciones Con Memoria

En este trabajo consideramos exclusivamente estrategias sin memoria o *memoryless*. Es decir que los nodos no acarrean absolutamente ninguna información sobre los paquetes o acciones previamente tomadas por el mismo nodo. Considerar esta información previa podría resultar en un incremento significativo de los resultados obtenidos intentando optimizar los distintos objetivos previamente planteados. Pero agregar esta característica al algoritmo también significaría que los estados deben considerar todas las posibilidades sobre las memorias de cada uno de los nodos. Se incrementa la complejidad de los estados con el objetivo de realizar la mejor acción posible con toda la información accesible sobre el camino tomado por los paquetes hacia su destino. Seria interesante considerar la memoria dentro de un mismo nodo sobre los paquetes previamente perdidos y también sobre los enviados, de esta forma los nodos podrían obtener información de como están los paquetes distribuidos sobre el sistema y poder tomar la decisión correspondiente con un estado que esta mas cerca de conocer el estado general del sistema.

III. Algoritmo

1. Idea

El algoritmo toma como entrada un plan de contacto de la forma de un Uncertain time varying graph Ω, concepto que será introducido en la siguiente sección, la cantidad de copias aceptadas y el nodo target de la red. A partir de estos datos el algoritmo construye el árbol de transiciones de la red empezando de atrás hacia adelante, primero se construye los estados objetivos. A partir de allí y navegando el plan de contacto hacia atrás en el tiempo, se calculan las transiciones que llegan a los estados conocidos del slot de tiempo actual y se calculan también los estados predecesores en el slot de tiempo previo. De esta forma completaremos el árbol considerando la información de costos codificada en el plan de contacto. Este orden en la búsqueda de los estados nuevos nos asegura la propiedad de que los nuevos estados construidos no se encuentran en $S_{=0}$, ya que existe un camino de transiciones en el árbol que alcanza algún estado objetivo.

A. Múltiples copias

El algoritmo permite obtener tablas de enrutamiento para múltiples copias de un mismo paquete de información. Como se mencionó existe un problema con la solución lograda por RUCoP dada por el manejo de un estado global de la red. Esto implica una limitación al considerar múltiples copias, ya que seria imposible en una red como las que estamos tratando que un nodo conozca con certeza el estado y ubicación de todas las copias de un mismo paquete. El problema se puede notar específicamente cuando la decisión de enrutamiento de un nodo varia según la información general del sistema que en realidad es inaccesible. Para abordar este problema luego de calcular el enrutamiento para una copia, se asume que cada nodo tomará siempre la misma decisión en caso de almacenar una copia. De esta forma limitamos algunas transiciones entre estados que no son implementables en la vida real. En este punto podemos notar la diferencia de la implementación de MORUCoP con el limite teórico perfecto en la probabilidad de que un paquete llegue a destino con éxito. Ejemplificaremos y ampliaremos sobre este punto en la sección III.3.

2. PseudoCodigo

Uncertain time varying graph

Se define como $\Omega = (\mathbb{G}, \Gamma, p_f, \delta, f_{dd}, \eta)$ compuesto por:

1. **Grafo Estático** $\mathbb{G} = (V, E)$ representando la conectividad de la red que permanece estable durante un slot de tiempo.

2. **Slots de tiempo** $\Gamma \subseteq \mathbb{T}$ donde $\mathbb{T}$ es el dominio temporal (normalmente es igual a $\mathbb{N}$). Y $\Gamma = t_1, t_2, ..., t_n$ es un conjunto finito donde n indica la cantidad de slots temporales que nos serán de interés. El tamaño de los slots es ajustable a las características topológicas de la red.

3. **Función de probabilidad de fallo** $p_f : E \times \Gamma \to [0,1]$ indicando la probabilidad de que una arista del grafo no ocurra como estaba planeado por el plan de contacto. Es decir la probabilidad de que el contacto en cuestión no pueda ser utilizado para el envío de paquetes.

4. **Función de delay de transición** $\delta : E \times \Gamma \to \Gamma$ modelando el costo en tiempo que utiliza una arista en enviar un paquete por el contacto que referencia, es decir que nos indica la cantidad de slots que deberán ocurrir para que el nodo objetivo pueda utilizar el paquete.

5. **Función de detección de fallo** $f_{dd} : E \times \Gamma \to \Gamma$ indica el tiempo en el que se detecta que el contacto no ocurrió como se esperaba, generalmente $f_{dd}(e,t) \geq \delta(e,t)$.

6. **Función de energía de transición** $\eta : E \times \Gamma \to \mathbb{Q}$ modelando el costo en energía de utilizar una arista de transición al enviar un paquete por el contacto de referencia.

El algoritmo MORUCoP 1

Inicialmente un conjunto de estados exitosos se genera en base al estado objetivo y la cantidad de copias permitidas. Existirá un estado exitoso por cada elemento en el conjunto $\{1, ..., num\_copies\}$ ya que se considera un estado exitoso si al menos una copia llega a destino. Luego en la línea 2 se agrega este conjunto al conjunto de estados explorados, y se inicia el loop sobre la cantidad de copias y slots de tiempo, destacando que el slot inicial es el anteúltimo del rango temporal a considerar.
En el conjunto de estados S_{t_i} del slot t_i encontraremos los posibles estados que pudieron derivar

en alguno de los pertenecientes a S_{t_i+1}. Para construir este conjunto determinamos el conjunto C_{t_i} de nodos que acarrean paquetes, por cada nodo c que acarree un paquete obtenemos los estados predecesores de c $pred^{+}_{C_{t_i}}$. Finalmente, como se ve en la línea 9, conseguimos los contactos que transmiten paquetes hacia c en el slot actual utilizando la función $contact_{C_{t_i}}$. A partir de este cálculo obtenemos todas las acciones posibles R_c que preservan la cantidad de paquetes acarreados por cada nodo $cp(c)$ y que podrían haber derivado en el estado s.
El condicional de la línea 12 se debe al problema del estado global que mencionamos anteriormente, es menester diferenciar las transiciones a considerar para el enrutamiento de una copia o de mas de una copia. Para el caso de solo tener disponibilidad de utilizar una copia el algoritmo simplemente considera todas las acciones posibles. En el otro caso debemos considerar solo aquellas transiciones que respeten las mejores acciones a tomar para estados con menor cantidad de copias totales precalculadas, es decir que solo consideramos las acciones que se condicen con las mejores acciones calculadas para estados con menor cantidad de copias. De esta forma sabemos que las acciones que consideramos para una mayor cantidad de copias no asumen que dos nodos diferentes pueden saber las copias acarreadas por el otro. Para cada transición que conforma la acción r calculamos su estado previo al estado s con la función $prev\_transition\_state(tr)$ y nos aseguramos que la acción minimizadora de costos $best\_action(s)$ este incluida en $\{\{tr\}, undefined\}$. Agregamos la posibilidad de que sea indefinido considerando el caso de acciones conformadas por solo una transición de $cp(c)$ copias, ya que en tal caso el algoritmo todavía no definió la acción a tomar en el estado previo. De esta forma podemos construir el conjunto de todas las acciones posibles que resultan en el estado s, como $T_r(s)$.
Recorriendo dicho conjunto obtenemos los nuevos estados que tienen una posibilidad de alcanzar algún estado objetivo mediante la función $get\_previus\_state(s,r)$ y los vamos almacenando en la variable acumulativa de los estados del slot particular. A partir del estado previo y la acción necesaria para llegar a s calculamos los costos relacionados con un nuevo algoritmo COSTS (Ver algoritmo 2). Por último, dentro del loop actualizamos el vector de costos $Cr(s')$ y también la mejor acción posible a tomar desde el estado s' en caso de que los costos sean menores a los almacenados previamente para el estado.

Cálculo de Costos

Una parte fundamental del algoritmo depende de un correcto cálculo de los costos relacionados a tomar una determinada acción desde un estado particular. Con este objetivo y además para facilitar el entendimiento del cálculo se define un algoritmo nuevo COSTS 2

Algorithm 1: The MORUCoP algorithm

Data: Uncertain time varying graph Ω, *num_copies*, Target

Result: Set of states S, Transitions Tr, Costs Cr

determine succesful states $S_{t_{end}}$ for *num-copies*;
$S \leftarrow S_{t_{end}}$
forall *c in [1,num_copies]* **do**
  forall t_i *in* Γ*, starting from* t_{end-1} **do**
    $S_{t_i} \leftarrow \emptyset$;
    forall $s \in S_{t_i+1}$ **do**
      determine carrier nodes C_{t_i}
      forall *node* $c \in C_{t_i}$ **do**
        $P_c \leftarrow c \cup \bigcup_{c' \in pred^+_{C_{t_i}}} contact_{C_{t_i}}(c', c)$
        $R_c \leftarrow \{r \subseteq \{0, ..., cp(c)\} \times P_c \mid \sum_{(k,p)\in r} k = cp(c)\}$
      end
      if $c > 1$ **then**
        $R_c \leftarrow \{r \in R_c \mid \forall_{tr \in r} best\_action(prev\_transition\_state(tr)) \in \{\{tr\}, undefined\}\}$
      end
      $Tr(s) \leftarrow \{\bigcup_{c \in C_{t_i}} r_c \mid \forall_{c \in C_{t_i}} : r_c \in R_c\}$
      forall $R \in Tr(s)$ **do**
        $s' \leftarrow get\_previus\_state(s, R)$
        $S_{t_i} \leftarrow S_{t_i} \cup \{s'\}$
        $costs^R \leftarrow COSTS(R, s', t_i)$
        if $Cr(s')$ *is undefined or* $Cr(s') > costs^R$ **then**
          $Cr(s') \leftarrow costs^R$
          $best\_action(s') \leftarrow R$
        end
      end
    end
    $S \leftarrow S \cup S_{t_i}$
  end
end
return S, Tr, Cr

que representa fielmente las definiciones de la sección de Costos 3. Para obtener el vector de costos debemos recorrer los diferentes casos de fallo posibles, por esto tomamos todos los conjuntos de contactos posibles y calculamos los costos para esa situación de fallo. En la línea 5 guardamos en la variable to_state el estado resultante de que fallen todos los contactos en fs a partir de la función $state\_after\_failures(R, s, fs)$. Luego obtenemos la probabilidad total de que ocurra determinado caso de fallo considerando también aquellos contactos que no fallaron pr_{fs}, es decir aquellos que pertenecen al conjunto $contacts(R) - fs$.
El estado resultante puede ser uno que no tenga probabilidad de éxito alguna para llegar al nodo objetivo, en dicho caso la función COSTS no estaría definida ya que el caso particular de fallo del conjunto de contactos en fs deriva en un estado de fallo irrecuperable. Dada la definición para el cálculo de la energía, debemos considerar incluso estos casos de fallo críticos para el promedio de energía consumida para cada acción. Por esto, en la línea 13, a partir de la función $energy\_of(R, s)$ y pr_{fs} obtenemos la energía que representa el caso de fallo actual.
Si el caso de fallo tiene una probabilidad de éxito no nula, implica que el vector de costos está definido y podemos utilizar las funciones de proyección del vector de costos $SDP(s)$, $ENERGY(s)$ y $DELAY(s)$. El SDP se acumula en la variable cr_{sdp} a partir de la probabilidad total de que fallen exactamente los contactos en fs multiplicando por la probabilidad de éxito una vez alcanzado el estado posterior a los fallos (línea 8). De igual manera, la energía es acumulada en la variable cr_{energy}, la energía del caso de fallo particular se obtiene multiplicando pr_{fs} por la suma entre la energía consumida en cada transición de R con el costo de energía esperada para el estado posterior a los fallos (línea 9). El cálculo del delay es particular porque, a diferencia de la energía, solo debemos considerar casos de fallo que tengan probabilidad de éxito mayor a 0. Como no sabemos a priori la probabilidad total de que el estado s con la acción R derive en un estado que no es de fallo, antes de calcular el delay para cada caso debemos conocer dicha información. Es por esto que en la línea 10 vamos guardando en el conjunto D_s las tuplas que contienen la probabilidad y el delay del caso particular.
Para el cálculo del delay, debemos obtener el promedio del tiempo de entrega obtenido para cada copia del paquete entregada. Por este motivo calculamos el promedio de los delays para todas las copias de la red, con la función $copies\_in\_target(s, target)$ incrementamos el delay solo para aquellas copias que aun no alcanzaron el nodo objetivo, notar que en caso de que ninguna copia se encuentre en el nodo target el incremento seria de una unidad temporal. Por último, para el caso en el que el estado posterior a los fallos resulta en un estado con posibilidad de éxito, acumulamos en la línea 11 la probabilidad total de que las combinaciones de fallos de contactos deriven en un estado con posible éxito. Para el cálculo del delay total

Algorithm 2: Costs Calculation (COSTS)

Data: Transition R, state s, current time slot t, target, num_copies

Result: Costs of current action

$cr_{sdp} \leftarrow 0$; $cr_{energy} \leftarrow 0$; $cr_{delay} \leftarrow 0$
$D_s \leftarrow \emptyset$
$pr_s \leftarrow 0$
forall $fs \in \mathscr{P}(contacts(R))$ **do**
  $to\_state \leftarrow state\_after\_failures(R, s, fs)$
  $pr_{fs} \leftarrow \left(\prod_{e \in contacts(R) - fs} 1 - p_f(e,t)\right) * \left(\prod_{e \in fs} p_f(e,t)\right)$
  if $COSTS(to\_state)$ *is defined* **then**
    $cr_{sdp} \leftarrow cr_{sdp} + pr_{fs} * SDP(to\_state)$
    $cr_{energy} \leftarrow cr_{energy} + pr_{fs} * \Big(energy\_of(R,s) + ENERGY(to\_state)\Big)$
    $D_s \leftarrow D_s \cup \left\{\left(pr_{fs}, \frac{num\_copies - copies\_in\_target(s,target)}{num\_copies} + DELAY(to\_state)\right)\right\}$
    $pr_s \leftarrow pr_s + pr_{fs}$
  else
    $cr_{energy} \leftarrow cr_{energy} + pr_{fs} * (energy\_of(R,s))$
  end
end
forall $(pr_{fs}, delay) \in D_s$ **do**
  $cr_{delay} \leftarrow cr_{delay} + \frac{pr_{fs}}{pr_s} * delay$
end
return $(1 - cr_{sdp}, cr_{energy}, cr_{delay})$

se toma cada una de las tuplas de probabilidad y delay precalculadas en el conjunto D_s para calcular la probabilidad de que luego de los fallos se llegue a estados con posibilidad de éxito, multiplicando dicha probabilidad por el delay estimado y acumulando los valores obtenemos el delay promedio de los casos exitosos para la acción preestablecida.

3. Un Ejemplo Simple

Para ejemplificar el funcionamiento y las optimizaciones de los objetivos que logra el algoritmo de MORUCoP, introducimos el plan de contacto de la figura 3. Una red simple bastante similar a la 2 pero con algunas diferencias esenciales que logran construir un escenario ideal para optimizaciones en el tiempo y energía. Ahora, algunos contactos particulares consumen menos energía que el resto. Esto es normal en redes cuyas topologías consideran

nodos con tecnologías muy diferentes, incluso considerando la posibilidad de que algunos nodos dispongan de una conexión constante a electricidad mientras que otros subsisten a base de baterías.

Como en el ejemplo anterior, consideramos 1 copia habilitada para el envío de paquetes y calculamos los costos para 2 caminos diferentes, el verde y el rojo. Los cálculos del camino rojo refieren a tomar la decisión de enviar desde $B \rightarrow D$ en T2, luego en caso de fallar en esta comunicación se opta por tomar el contacto hacia E, es decir que este camino a diferencia del verde tiene una posibilidad de rerouting. Lo que intuitivamente puede hacer pensar que la probabilidad de éxito debería ser mayor si tomamos el camino rojo. Sin embargo la probabilidad de éxito por ambos caminos resulta ser la misma con un 25 % de SDP. Esto se debe a que por el camino rojo una vez que la custodia del paquete se traslada al nodo D se pierde la posibilidad de reruteo porque no existe un contacto posterior con el nodo predecesor B. Por lo que el éxito en el envío del paquete termina dependiendo únicamente del contacto entre D y E en el cuarto slot temporal. Por otro lado la energía y delay promedio consumida por el camino rojo resulta ser sustancialmente menor al verde. Esto nos marca que bajo cualquier configuración de prioridades sobre los objetivos el algoritmo tomaría las acciones correspondientes con el camino rojo. Mientras que un algoritmo que solo considera como objetivo minimizar la probabilidad de fallo como lo es RUCoP podría elegir aleatoriamente cualquiera de los dos caminos dependiendo de la implementación.

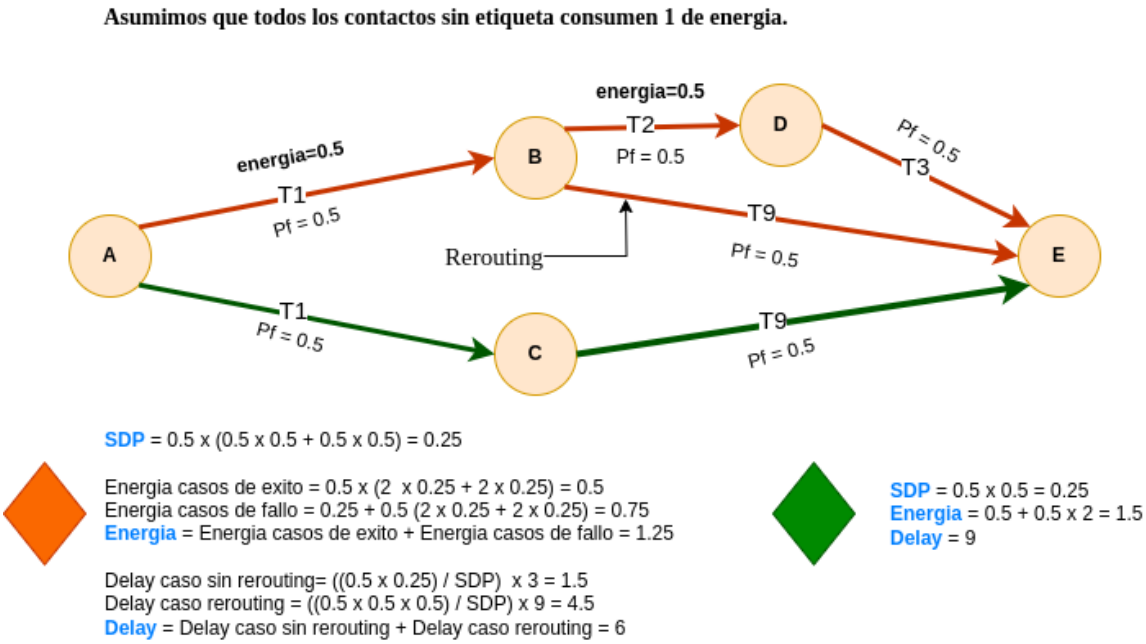


Figura 3: Plan de contacto del ejemplo simple.

IV. Experimentación

En esta sección reportamos los resultados de la experimentación llevada a cabo utilizando la herramienta omnet++, que es un framework que provee una libreria basada en el lenguaje de programación c++. Se utilizada para reproducir una amplia variedad de simulaciones sobre redes con distintas topologías y formatos. Gracias a trabajos previos en la implementación de varias simulaciones utilizando el algoritmo de RUCoP para el enrutamiento, podremos comparar la optimización de los objetivos para ambos algoritmos. Como consecuencia a la adaptabilidad del algoritmo de MORUCoP podremos comparar para distintos escenarios que introduciremos a continuación, con distintas configuraciones de prioridades en los objetivos. Debido a la limitación de configurar distintas probabilidades de fallo para los contactos involucrados en la red, en todas las simulaciones la probabilidad de fallo será la misma para todos los contactos en una simulación particular. Para cada probabilidad fija en el rango de [0, 0.1, 0.2 ..,1] se aleatoriza los fallos de los contactos con distintas semillas para cada simulación. Construyendo de esta forma un escenario de fallo diferente en cada ejecución. En el pseudocódigo se asume un único nodo objetivo mientras que en las simulaciones se toma

una tabla de enrutamiento con varios nodos como posibles targets finales.

1. Redes aleatorias

En este primer escenario se consideran 10 topologías de red diferentes con 8 nodos y una duración de 100 segundos. El tiempo se fragmenta en slots de 10 segundos. En cada instanciación de la simulaciones con estas topologías se decide la conectividad entre los nodos (i.e, presencia de contactos) en base a un parámetro de densidad de conectividad del 20 %, similar a (Madoery et al. 2018). Estas topologías suponen la posibilidad de comunicación de todos con todos entre los nodos que las componen.

2. Ring Road

Este caso de estudio presenta una constelación real de 16 satélites que fue propuesta y descrita en (Fraire et al. 2017). Compuesta por satélites que trabajan como mulas de información, es decir que reciben los datos de 22 terminales aisladas en la tierra y los almacenan hasta lograr transmitir hacia la estación base en la Argentina. A diferencia del escenario anterior, la topología y el objetivo de esta red conforma una estructura de todos a uno en el flujo de la información. Como podemos ver en la figura 4 los satélites están equipados con equipamiento que permite contactos ínter-satelitales (ISLs) por lo que las comunicaciones son posibles también en orbita. Los caminos tomados por los paquetes pueden involucrar tanto satélites como estaciones terrestres. El escenario es propagado por 24 horas, se dividió en 4 topologías diferentes separando 6 horas para cada una. Para cada topología generamos 1 plan de contacto fragmentando la red en 360 slots de 60 segundos cada uno. La condición necesaria para considerar que existe un contacto entre dos nodos a y b en un determinado slot de tiempo, es que exista una ventana de comunicación entre a y b durante al menos 30 segundos.

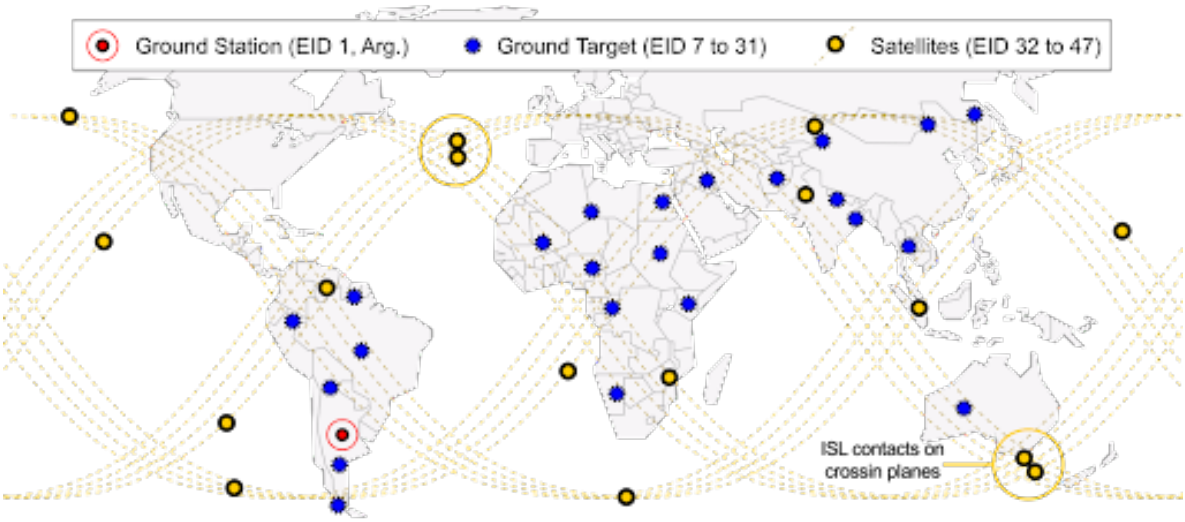


Figura 4: Ring Road Network.

3. Comparación con RUCoP

Se obtuvieron gráficos con la información pertinente a cada escenario de simulación, se calcularon diferentes metricas con la intención de mostrar las ventajas en versatilidad del algoritmo de MORUCoP contra RUCoP. Las métricas obtenidas fueron las siguientes:

1. **Delivery Ratio**: Porcentaje de los paquetes enviados por los nodos fuente hacia los objetivos finales que efectivamente llegaron a destino.

2. **Energy Efficiency**: Representa cuanto se aprovecho cada una de las transiciones y como medida del ahorro o desgaste energético, obtenida de dividir la cantidad de paquetes recibidos por los nodos objetivos sobre la cantidad de transiciones utilizadas particularmente para el envío de los paquetes entregados.

3. **Delay**: Promedio de los tiempos de demora en la entrega de todos los paquetes que arribaron exitosamente al nodo objetivo.

Teniendo en cuenta estas métricas que representan cada uno de los objetivos que tomamos como prioridad al momento del enrutamiento, introduciremos algunos gráficos que son pertinentes y que muestran diferencias sustanciales entre los algoritmos. Para que las soluciones difieran, en particular cuando la primera prioridad para el algoritmo de MORUCoP es el SDP, deberán existir al menos 2 caminos posibles diferentes para los paquetes que tengan igual probabilidad de éxito pero que difieran en los costos de energía y de delay.

4. Resultados

Los gráficos que veremos a continuación presentan los resultados de 100 simulaciones ejecutadas para cada uno de los escenarios y para las diferentes cantidades de copias admitidas. Para cada una de las probabilidades de fallo simuladas se muestran velas correspondientes con los resultados de cada algoritmo. Cada vela se compone por una linea horizontal interior a una caja en vertical de la cual se desprenden dos lineas negras, una hacia arriba y otra hacia abajo. La linea interior marca el promedio de los valores obtenidos en todas las simulaciones, la longitud de la caja muestra la variabilidad de los resultados ya que se calcula a partir de la desviación estándar de los mismos, por ultimo las lineas con tope representan los valores máximos y mínimos obtenidos. No se consideran los mínimos y máximos para las simulaciones con el escenario Ring Road para las métricas de Energy Efficiency y Delivery Ratio. Ya que dichas simulaciones consisten en el envío de un único paquete hacia un nodo particular, por lo que el máximo de cada vela resultaba ser recurrentemente uno y análogamente para los mínimos resultaba ser cero.

Prioridades: *SDP > Energy > Delay*

Podemos Notar en los gráficos 6 y 5 que las diferencias entre los dos algoritmos para la métrica de Delivery Ratio es casi indistinguible, incluso los resultados son idénticos si utilizamos solo una copia (Ver tambien 12). Dado que ambos algoritmos toman como objetivo prioritario maximizar la probabilidad de entrega de los paquetes, es normal que ocurra esto ya que los caminos elegidos por cada algoritmo desde cada estado recorrido tienen necesariamente la misma probabilidad. Diferenciándose únicamente en que el algoritmo de MORUCoP tiene la posibilidad de elegir un camino con menores costos de energía y delay. Las pequeñas variaciones se deben a esta causa y a la aleatorización de los fallos en los contactos. Priorizando la probabilidad de éxito de los envíos incluso puede ocurrir que ambos algoritmos generen las mismas tablas de enrutamiento. Es interesante remarcar que mientras menor es la probabilidad de fallo, mayor es la libertad para que MORUCoP tome caminos con una mayor eficiencia energética. Por otro lado, para las probabilidades mas grandes, las simulaciones tienden a tener una variabilidad mayor en las métricas de Eficiencia Energética y Delay.

Figura 5: Random Net 1 with 3 copies

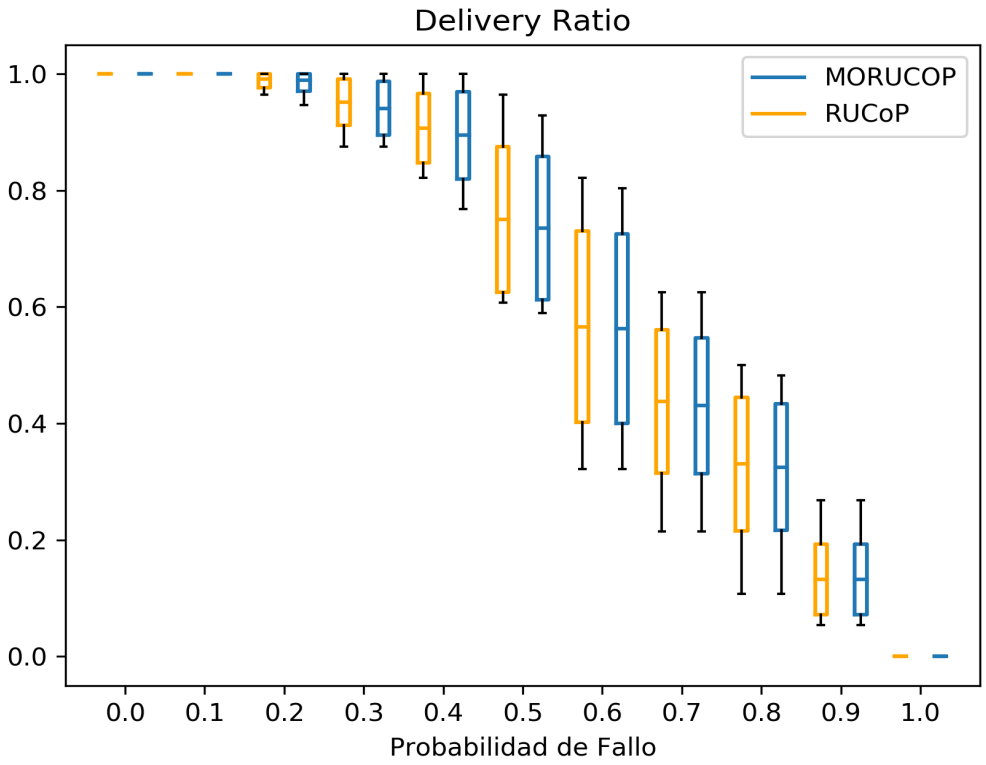


Figura 6: Random Net 1 with 1 copy

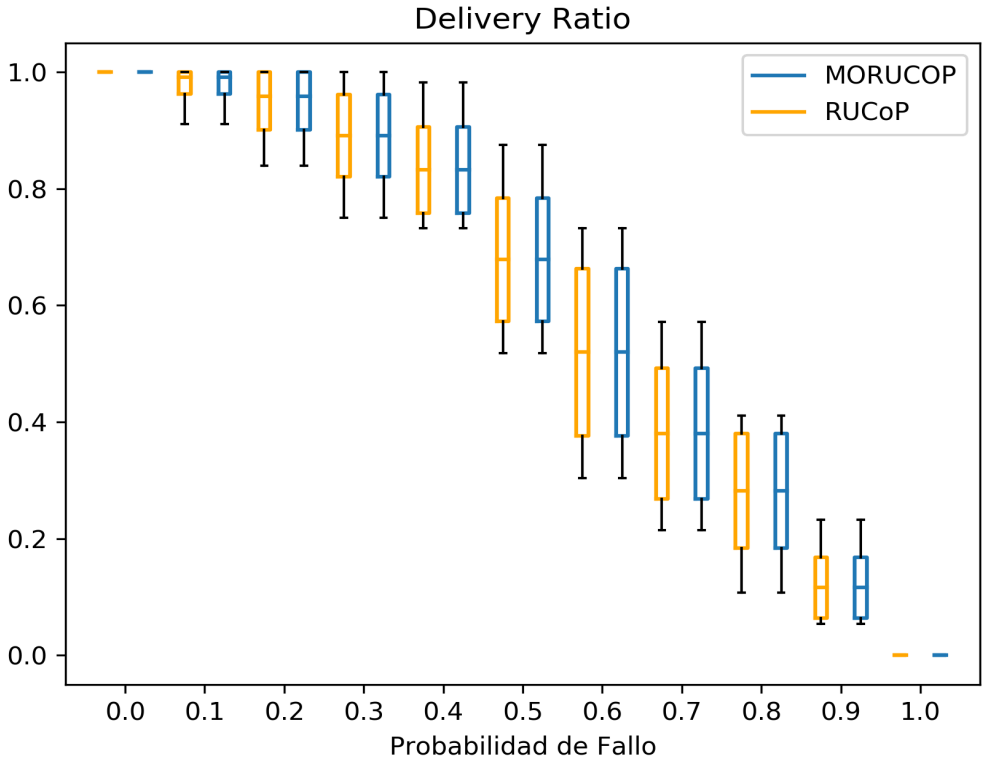


Figura 7: Random Net 1 with 3 copies

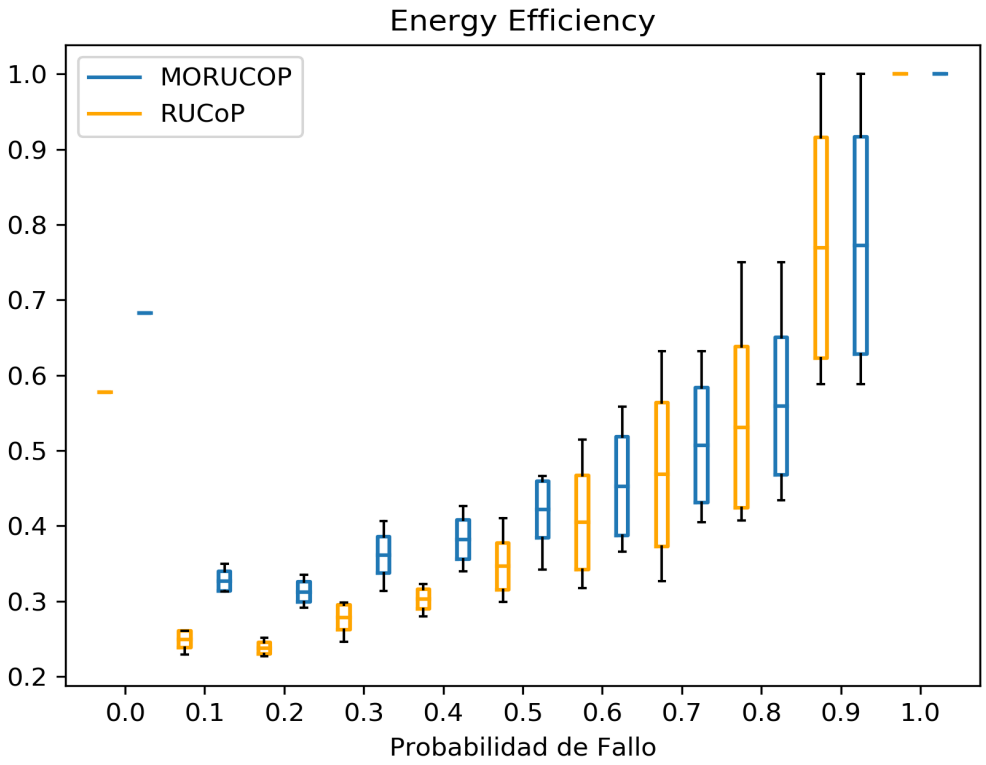


Figura 8: Random Net 1 with 1 copy

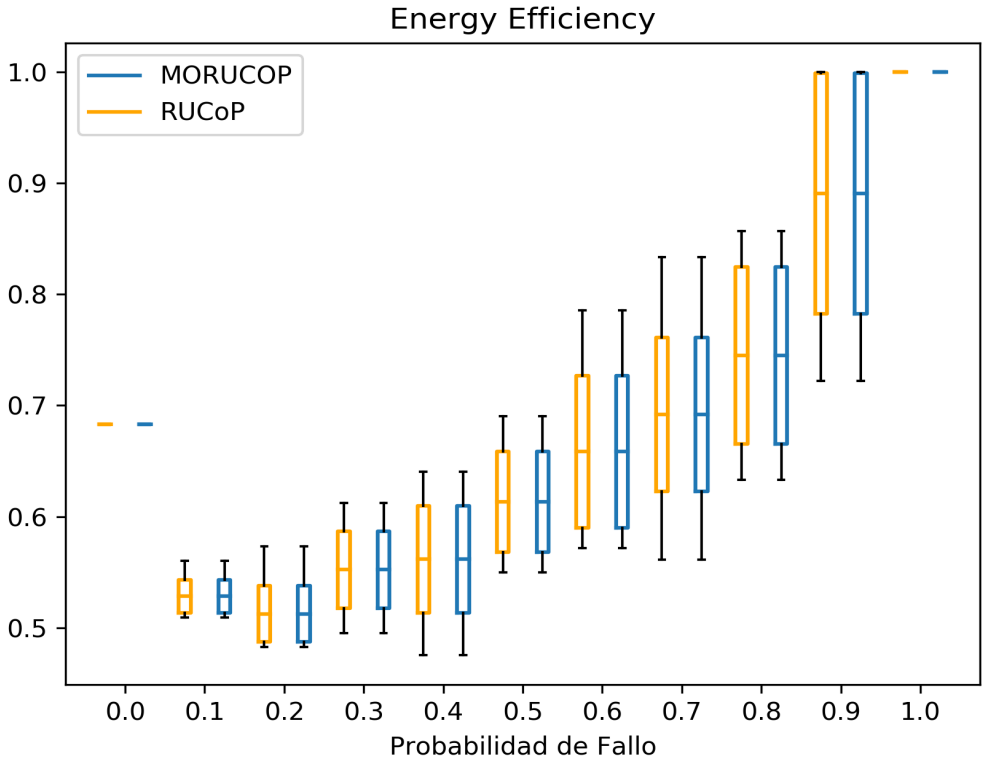


Figura 9: Random Net 1 with 3 copies

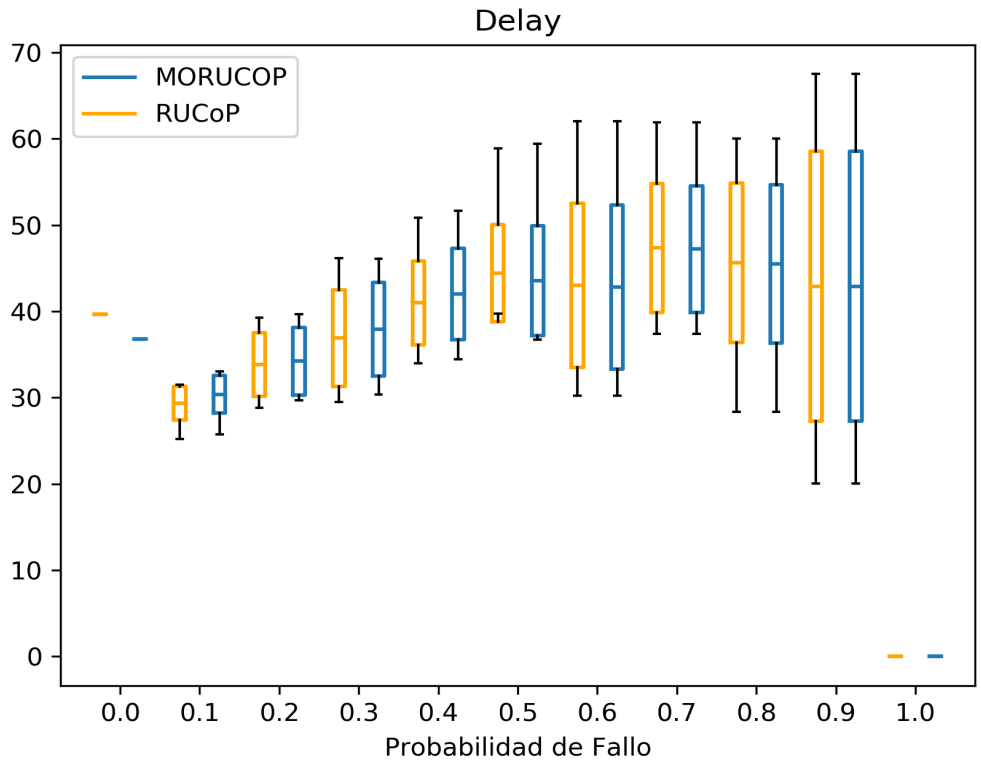


Figura 10: Random Net 1 with 1 copy

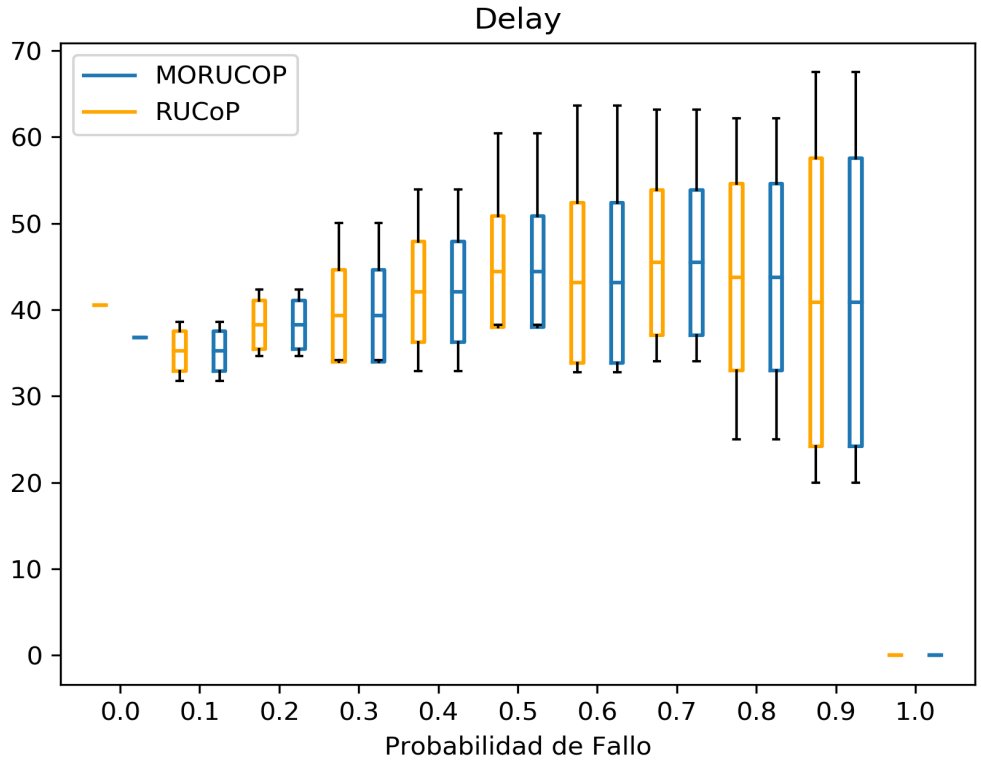


Figura 11: Ring Road Net 1 with 3 copies

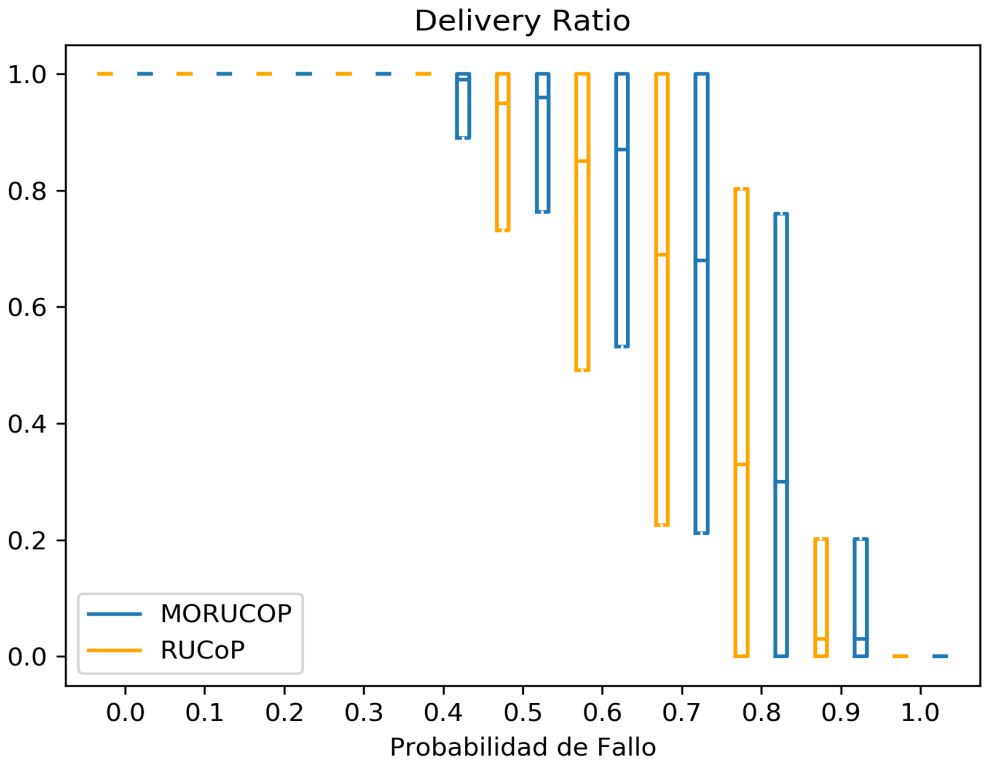


Figura 12: Ring Road Net 1 with 1 copy

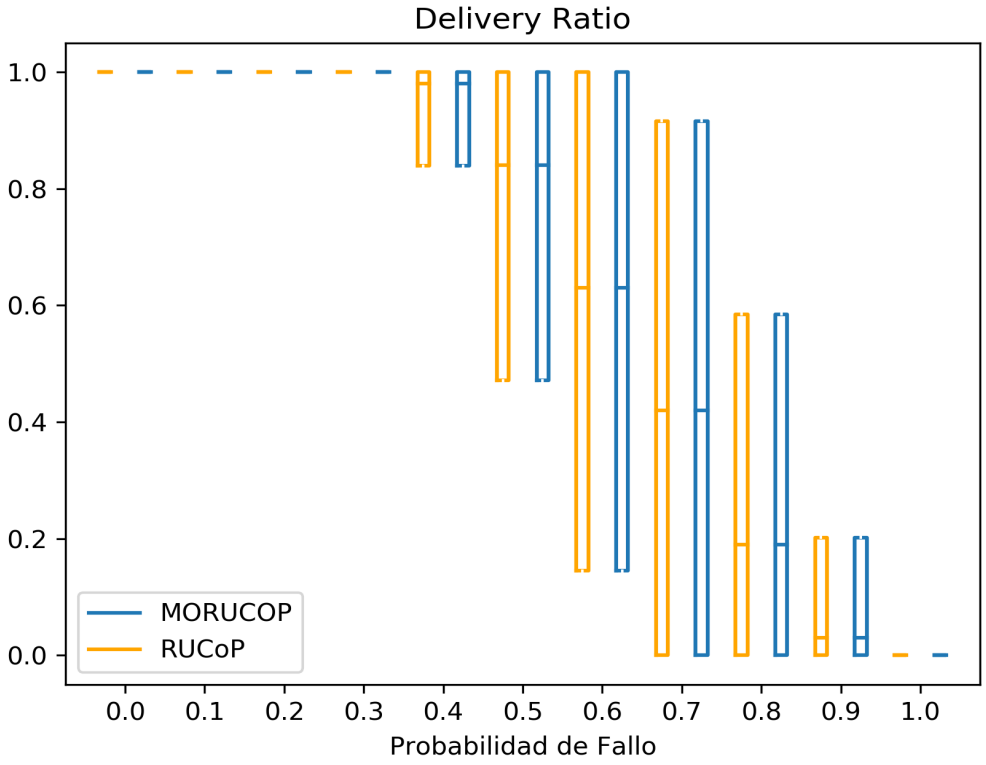


Figura 13: Ring Road Net 1 with 3 copies

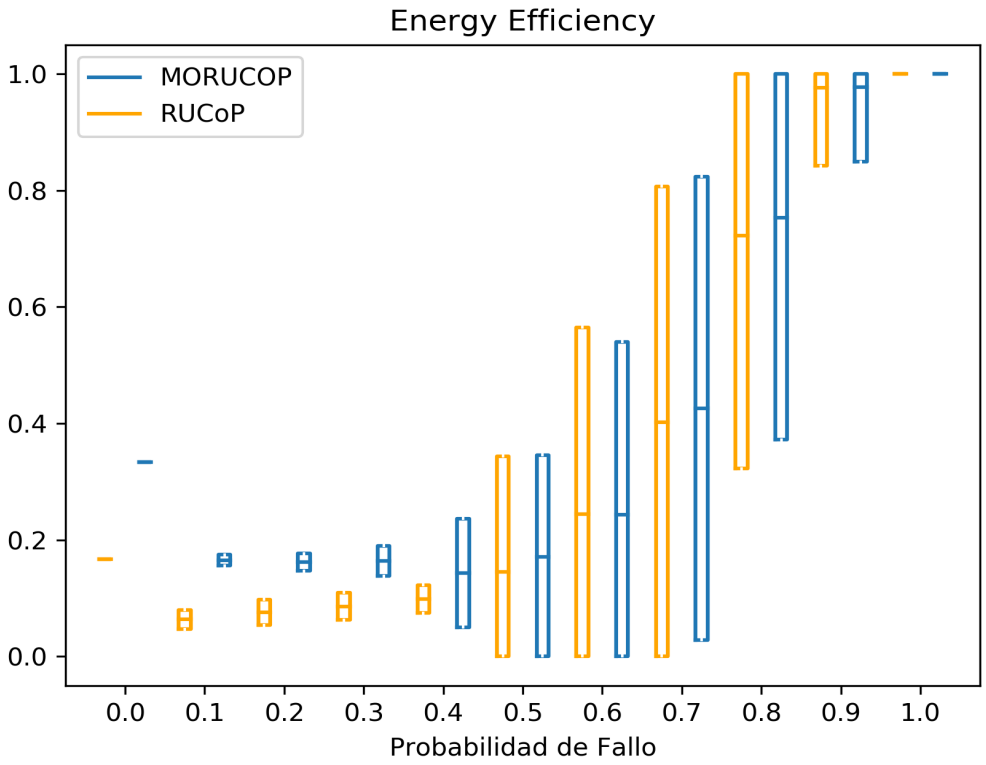


Figura 14: Ring Road Net 1 with 1 copy

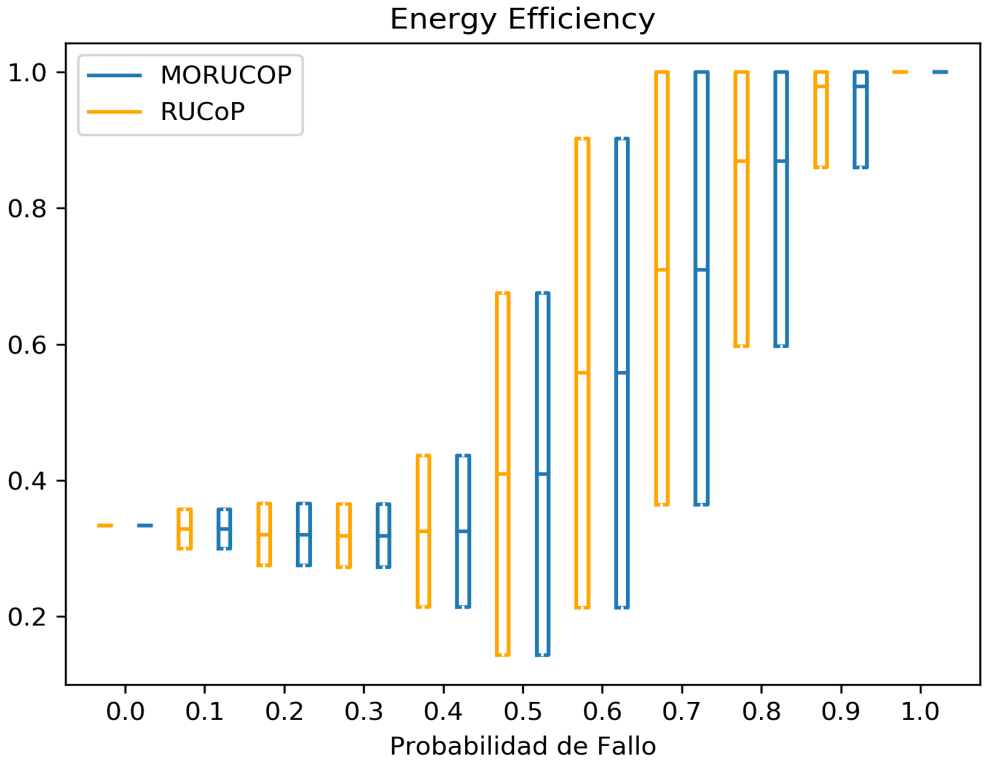


Figura 15: Ring Road Net 1 with 3 copies

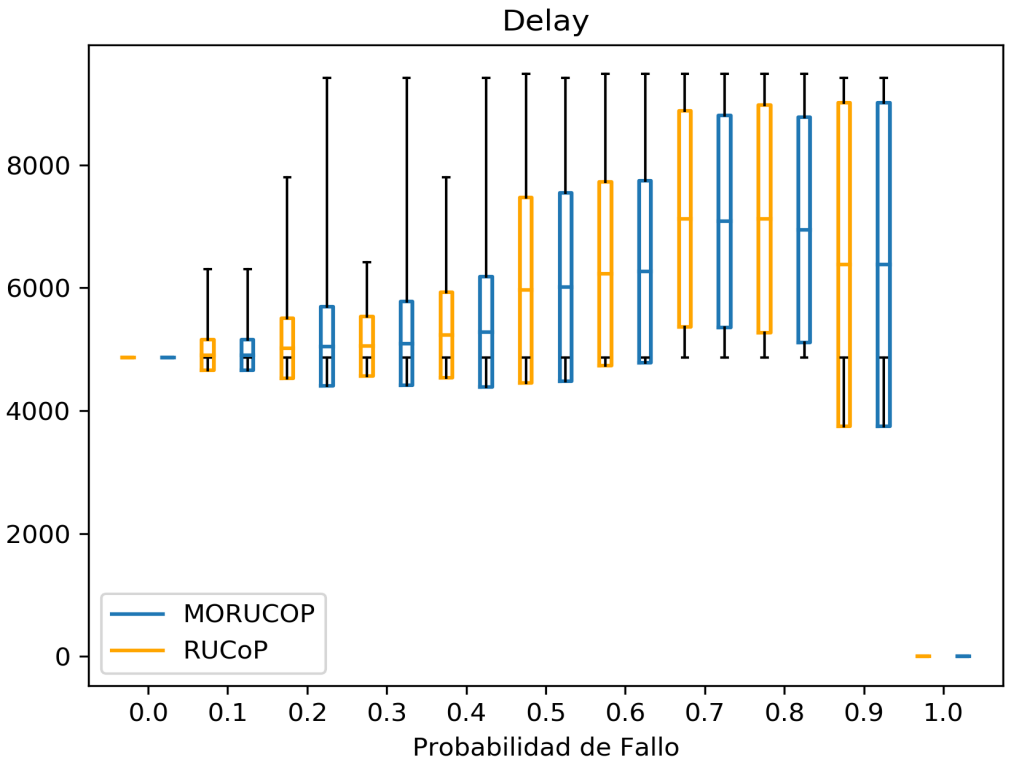
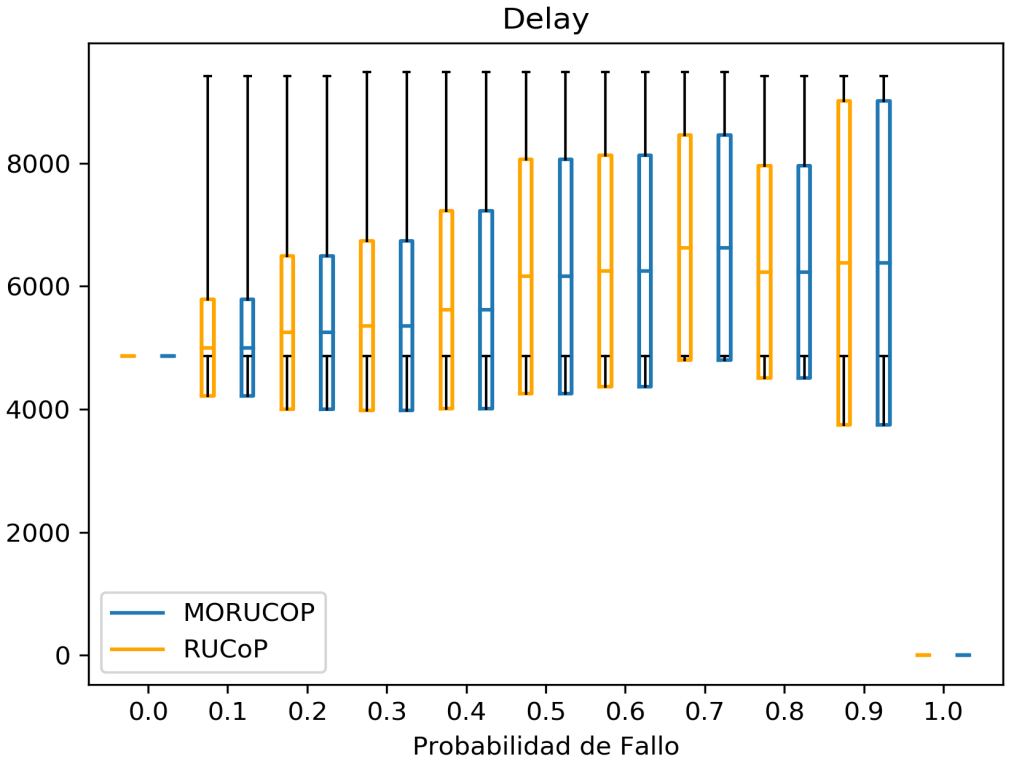
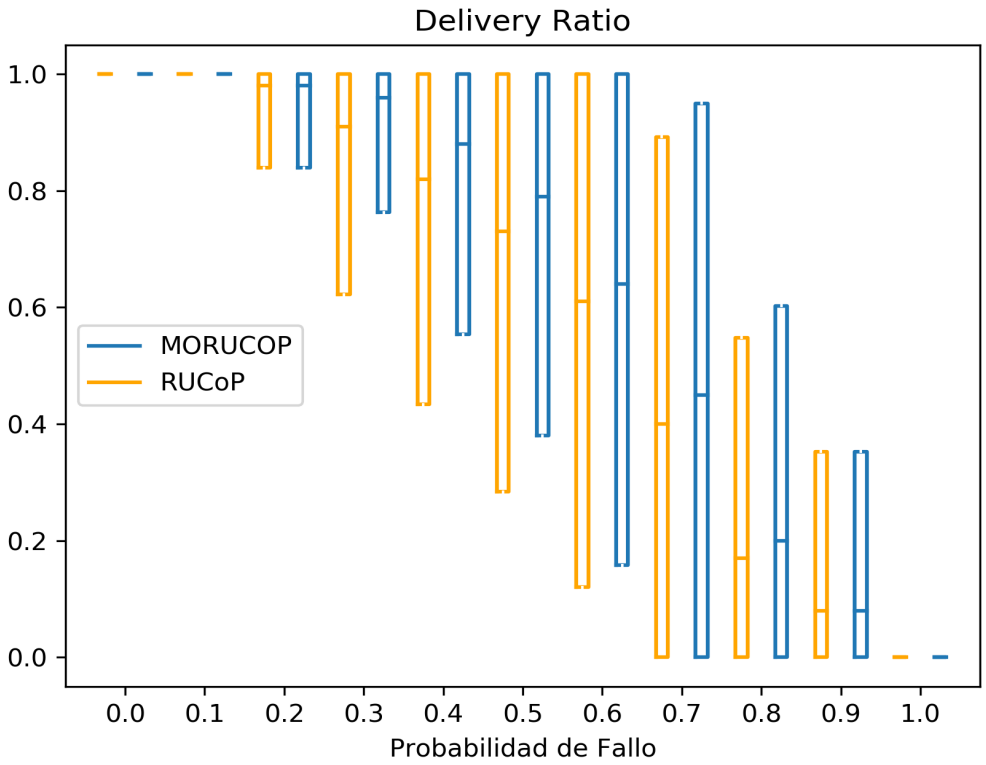


Figura 16: Ring Road Net 1 with 1 copy



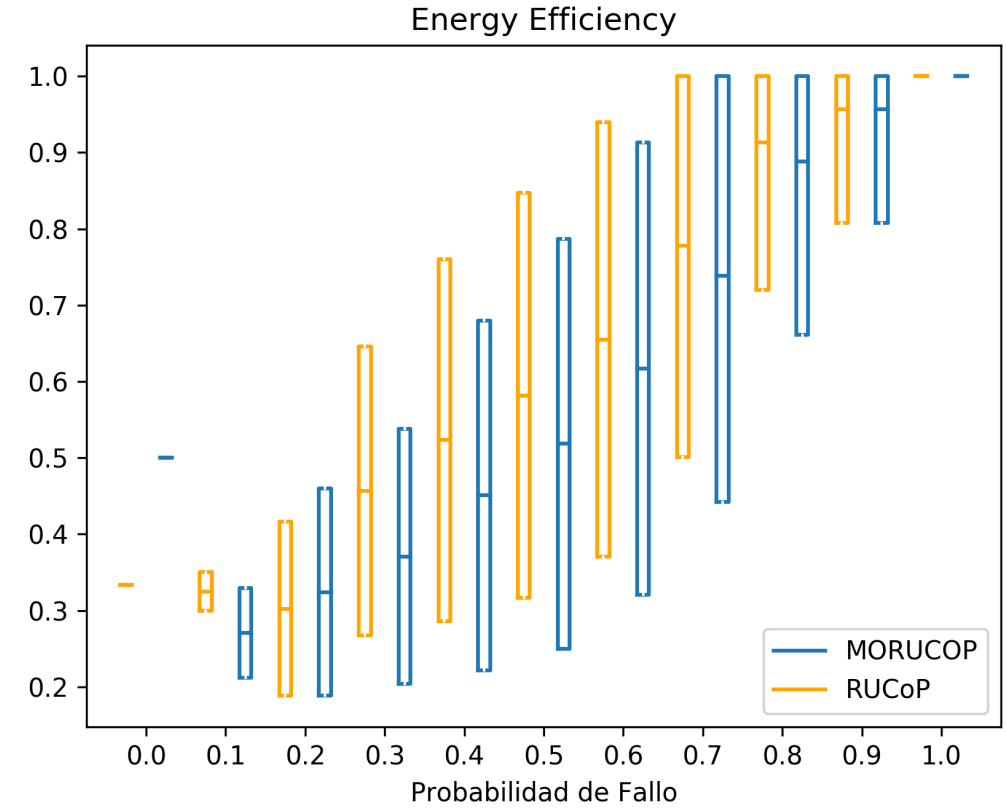
Resultados similares se obtuvieron de los escenarios graficados en 7 y 13, si comparamos los gráficos de Delay Promedio y Delivery Ratio son muy parecidos. En cambio, en términos de Eficiencia Energética, obtenemos una mejora sustancial para el algoritmo MORUCoP. Como era de esperarse las prioridades del algoritmo surten efecto en los resultados de las simulaciones para 2 escenarios diferentes, uno con una red aleatorizada y el otro que considera un rango temporal de la red ring road. También es interesante notar que la diferencia entre los algoritmos se incrementa con una mayor cantidad de copias permitidas a la red, esto le da mayor libertad al algoritmo de MORUCOP.

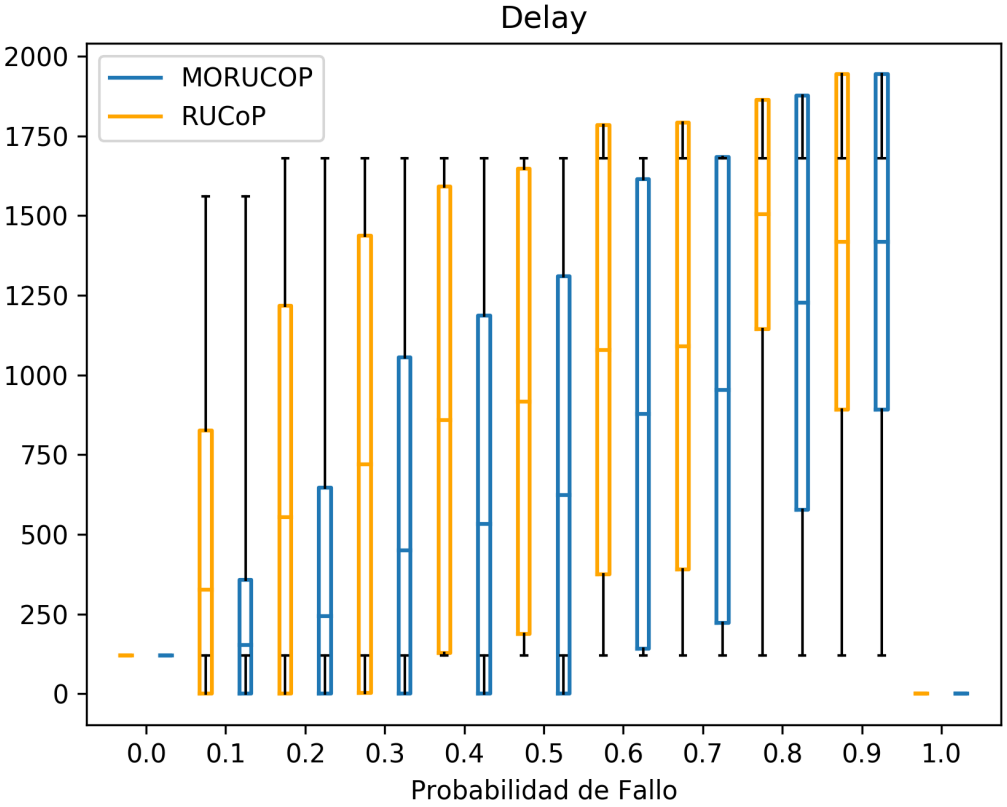
Figura 17: Ring Road Net 2 Delivery Ratio



En este segundo caso de la red ring road podemos notar una leve diferencia en la cantidad de paquetes entregados y en el delay obtenido que resultó ser mejor para MORUCoP (Ver 17 y 18). Por otro lado, en el gráfico 18 de eficiencia energética vemos una leve superioridad de RUCoP para probabilidades de fallo mayores al 30 % . Mientras que, como se puede ver en todos los gráficos de esta sección sobre la eficiencia energética con tres copias, si la probabilidad de fallo es nula MORUCoP se comporta mucho mejor.

Figura 18: Ring Road Net 2





Prioridades: *Delay > SDP > Energy*

En los siguientes gráficos las prioridades de los objetivos fueron cambiadas intentando minimizar el tiempo de demora en el envío de paquetes. Por eso es que obtuvimos resultados muy favorables en la métrica de Delay Promedio, particularmente mejores en la red ring road (Ver figura 23). Esta mejora implica una mayor exposición a riesgos de perdida de paquetes y de alcanzar un estado de fallo total. Esto lo podemos ver en los gráficos de Delivery Ratio (En las figuras 20, 24 y 21). En el escenario particular de la segunda red aleatorizada se obtuvieron diferentes resultados en la comparación sobre el consumo de energía. MORUCoP resultó superior para las simulaciones con tres copias (Las figuras 19 y 24) mientras que RUCOP lo fue para aquellas con una sola copia admitida (Ver 22). Nuevamente vemos como la optimizacion en tiempos de entrega y energia surte un mayor efecto para las simulaciones que admiten una mayor cantidad de copias.

Figura 19: Random Net 2 with 3 copies

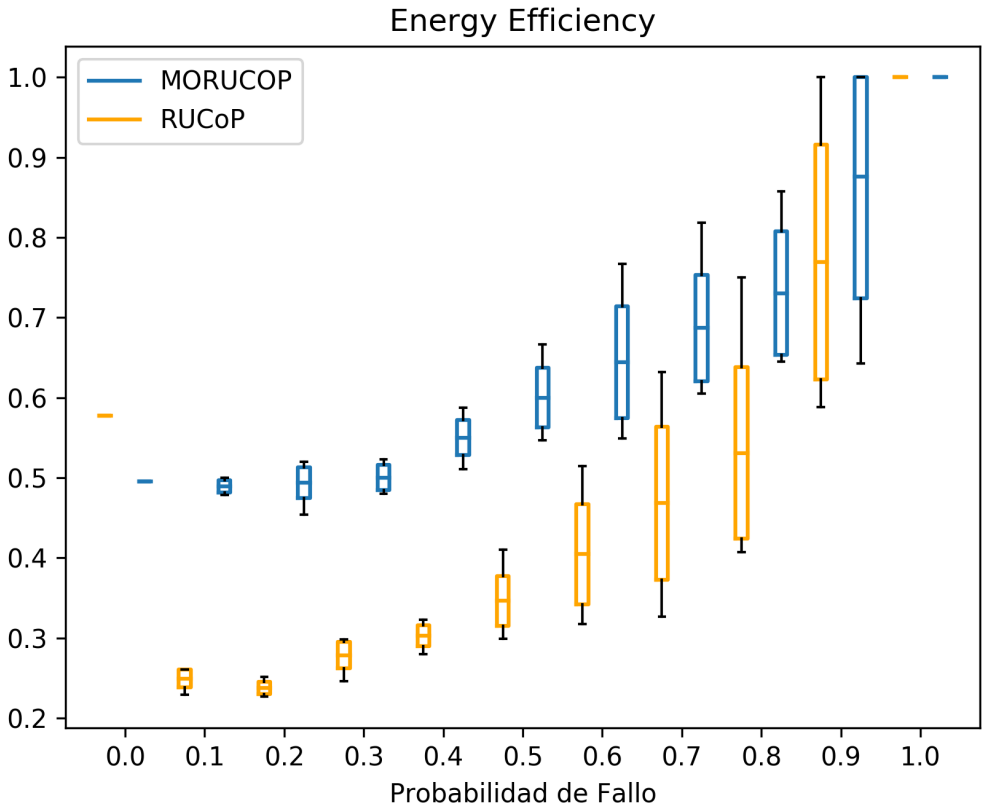
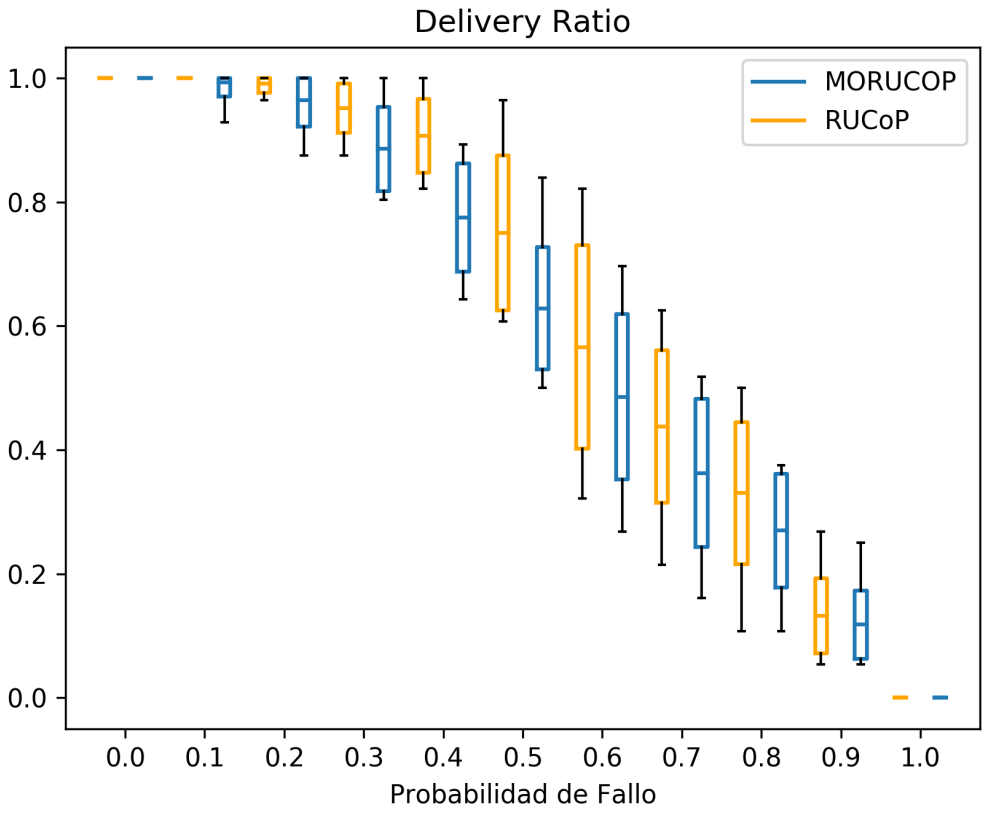


Figura 20: Random Net 2 with 3 copies



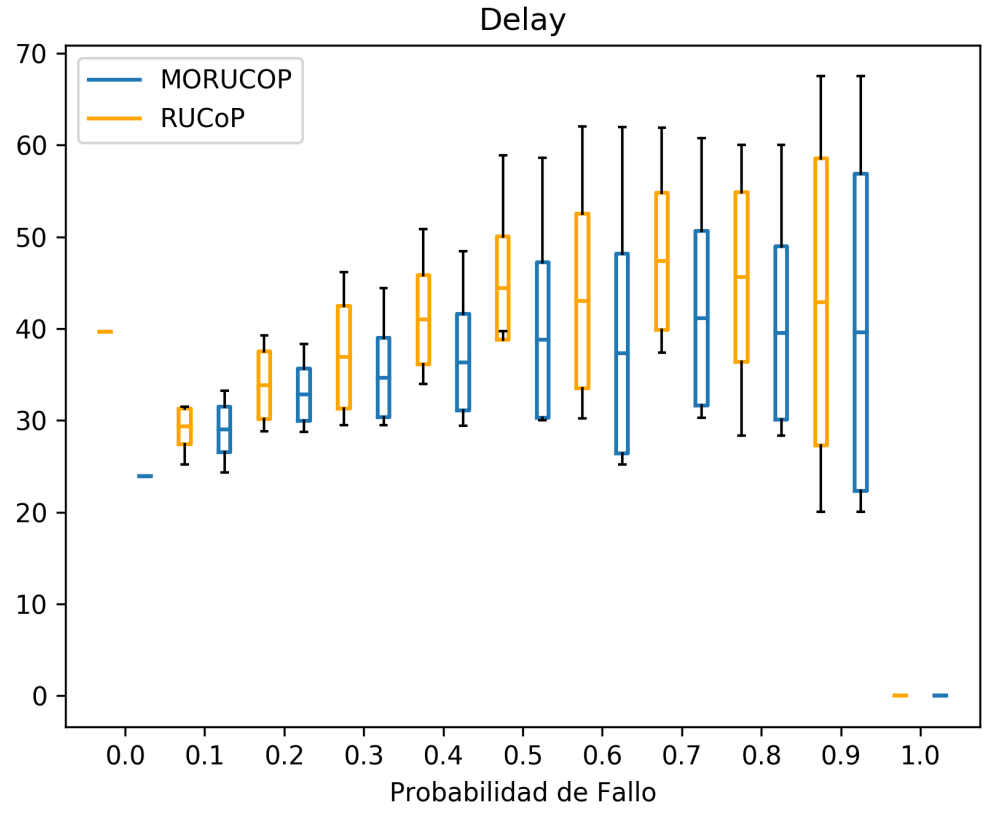
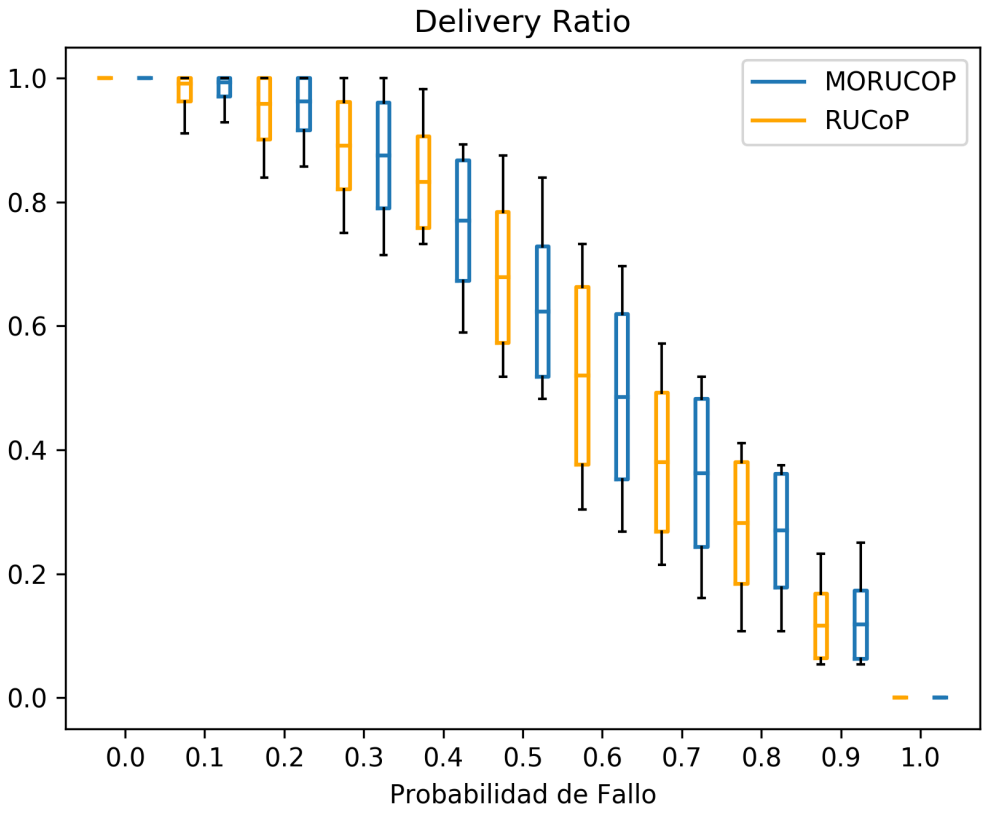


Figura 21: Random Net 2 with 1 copies



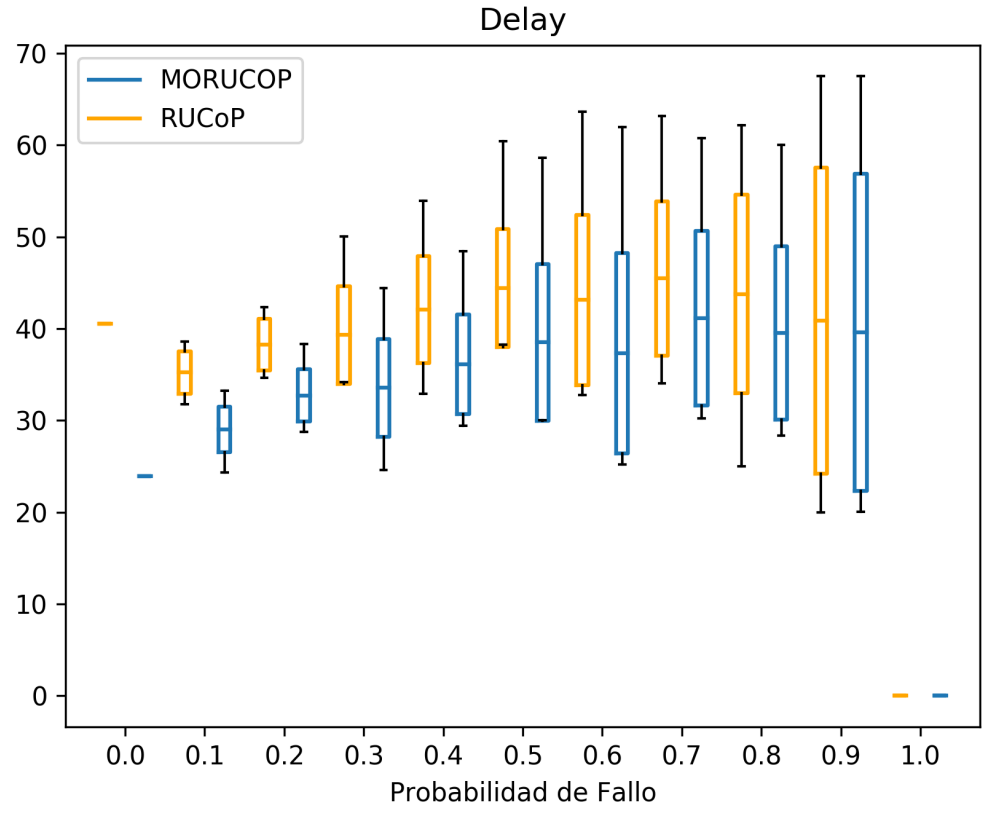


Figura 22: Random Net 2 with 1 copies

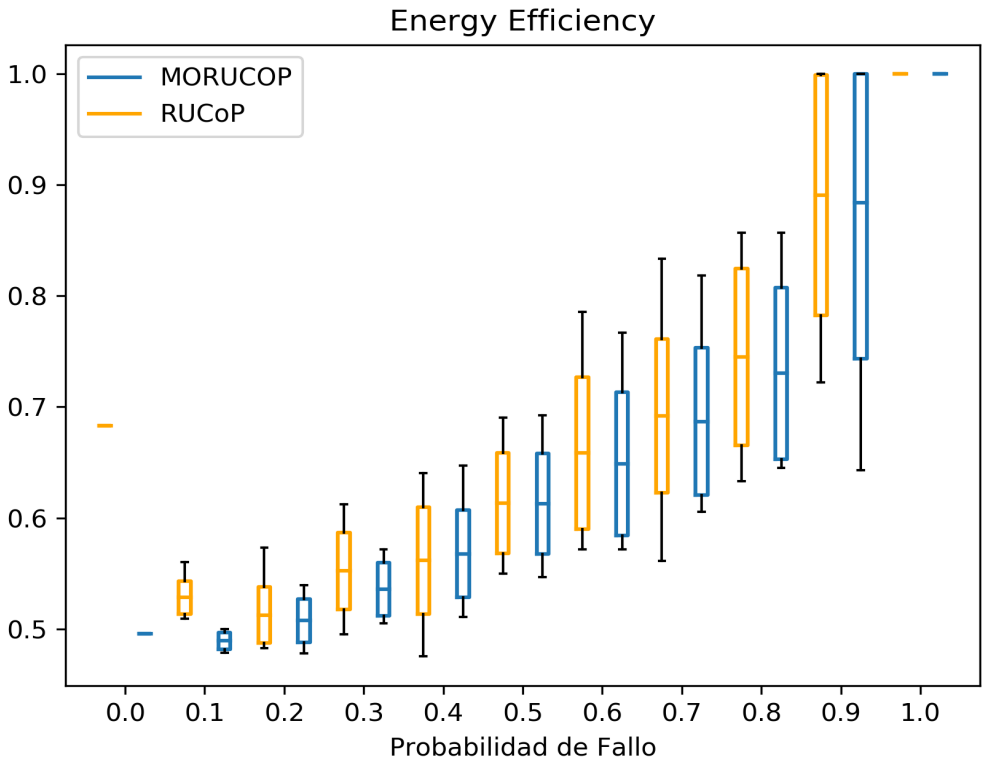


Figura 23: Ring Road Net 3 with 3 copies

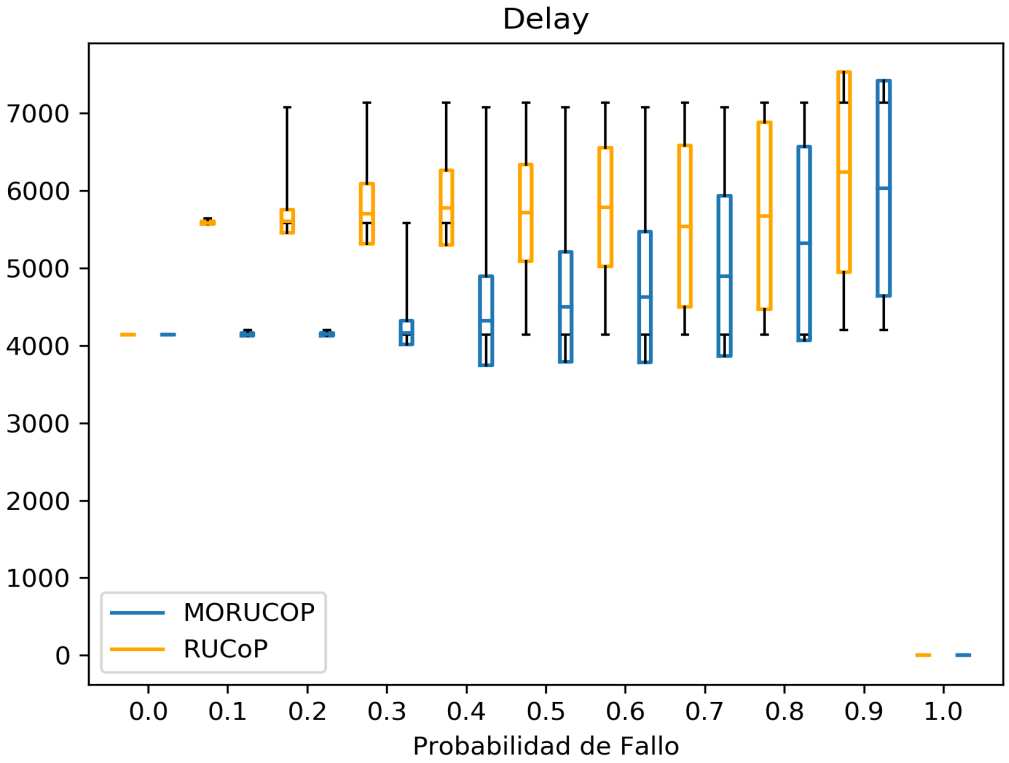
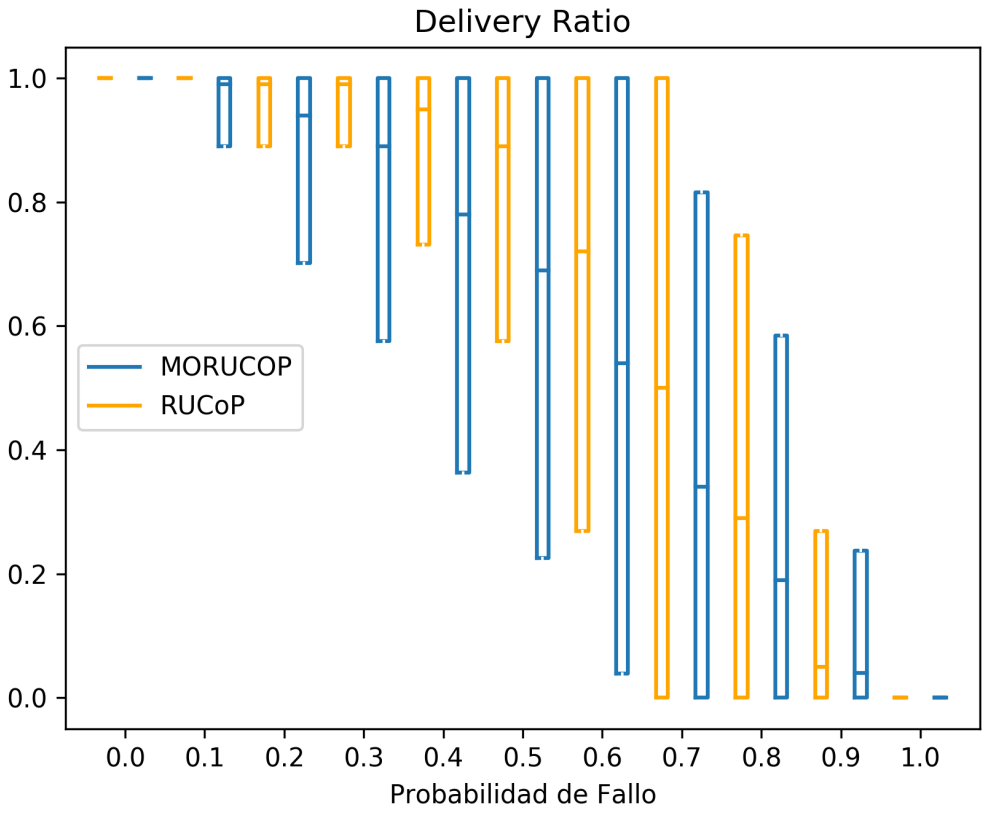
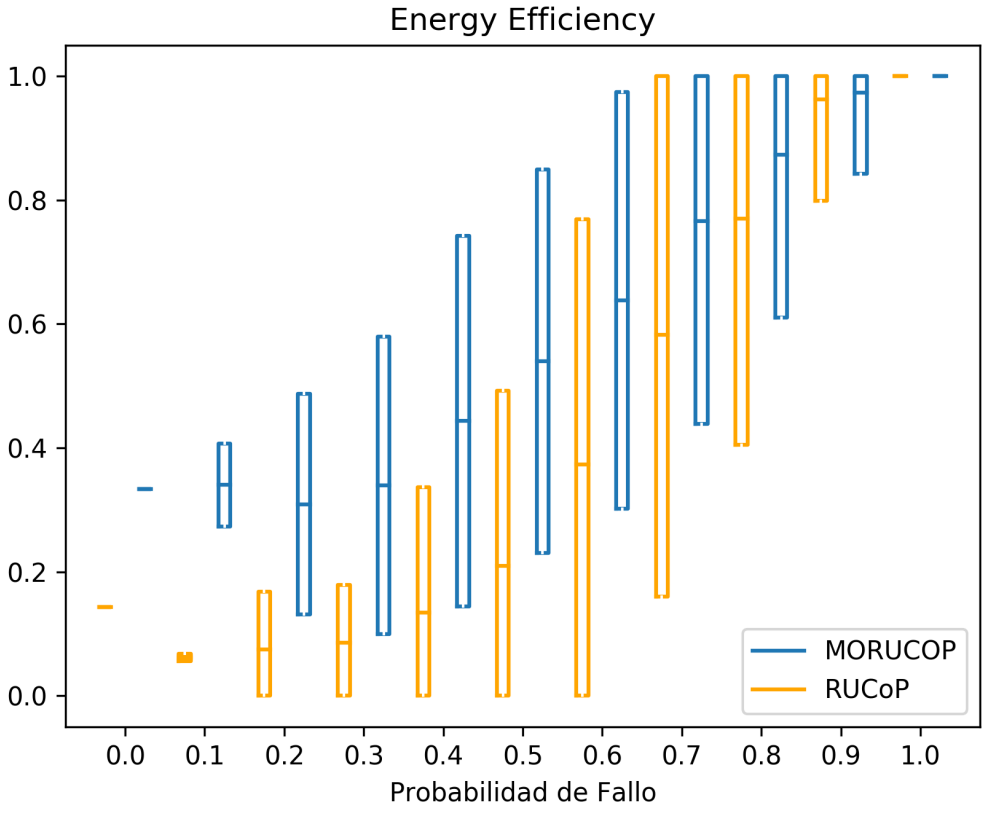


Figura 24: Ring Road Net 3 with 3 copies





V. Consideraciones Finales

1. Posibles Mejoras

El flujo de control que siguen las instrucciones del algoritmo MORUCoP (Algoritmo 1) empieza obteniendo las respuestas a los enrutamientos de estados con menor cantidad de copias. Estas decisiones son las que terminan moldeando el enrutamiento general del sistema que normalmente admiten mas de una copia de cada paquete. Si bien el algoritmo de RUCoP teórico es un algoritmo que garantiza obtener siempre la mejor decisión para cada estado, al no ser implementable en una red real, su implementación también termina dependiendo de estas acciones tomadas localmente por los nodos con una menor cantidad de copias de las que existen en la totalidad del sistema. Por esto es que seria interesante considerar una solución que tenga en cuenta para dichos estados con menor cantidad de copias las mejores decisiones considerando la cantidad de copias admitida por la red, es decir que el loop base del algoritmo tendría que iniciar en el sentido contrario al que se definió, empezando a conseguir las mejores decisiones para los estados con el total de copias disponibles y luego definir en base a estas decisiones el resto. En ese sentido, surgiría una nueva complejidad ya que por ejemplo las transiciones elegidas por un nodo N que solo tengan una copia y sin conocimiento del resto de nodos, dependerá de los costos de todos aquellos estados con mas de una copia que coincidan en que un paquete esta alojado en el nodo N.

2. A Futuro

Se propone como trabajo futuro a este proceso que se considere tanto en el algoritmo como en la implementación del mismo la posibilidad de la ocurrencia de ciclos comunicacionales entre los nodos dentro de un mismo slot temporal, de esta forma lograríamos agregarle mucho mas realismo y utilidad al algoritmo. También hay espacio de mejora en términos de configurabilidad en la implementación, ya que no se provee una forma de configurar la energía especifica consumida por cada contacto del plan de contactos. Otro aspecto a tener en cuenta seria sobre la suposición que se tomó sobre el ACK de una comunicación ya que en la realidad este mensaje también contendría una probabilidad de fallo que no fue considerada en las simulaciones.

Sobre la forma de elegir la optimización de los costos con prioridades podemos decir que seria ideal poder tener mayor posibilidad de elección que simplemente ordenar los objetivos de mayor a menor prioridad. De esta forma se pierden muchas posibles elecciones de enrutamiento que son intermedias a las conseguidas por MORUCoP. Al igual que la mejora propuesta

en la sección anterior, si nos imaginamos el árbol de transiciones del algoritmo, sobre un estado E, podríamos obtener la Probabilidad de fallo, Energía y Delay máximos de los estados consecuentes al mismo, es decir aquellos que están en el tiempo t + 1. Para luego normalizar los costos de todas las decisiones posibles a tomar desde el estado E, esto se lograría dividiendo correspondientemente los costos por los máximos y así obtener valores en el rango $[0, 1]$. Luego, en lugar de tomar las prioridades como entrada, tomamos un peso a considerar para cada objetivo incrementando el peso para aquel objetivo que menos importancia tenga, de esta forma al sopesar y sumar el resultado para los costos de cada estado comparamos el numero de costo para cada estado y elegimos aquel que tenga el menor numero para elegir la mejor transición disponible.

3. Conclusión

En este trabajo, se propuso un algoritmo que aborda los problemas del análisis multiobjetivo del enrutamiento en redes tolerantes a demora. Se mostró con ejemplos teóricos y reales con distintos tipos de topologías que el análisis multiobjetivo puede ser de gran valor al momento de optimizar en tiempo y energía los costos de la red. Se implemento el algoritmo de enrutamiento y se lo adapto para obtener resultados de distintas simulaciones para comparar con RUCoP. En conclusión, si bien el algoritmo se presta a algunas mejoras como se menciona en la sección anterior, resulta ser de gran configurabilidad y adaptabilidad a las distintas redes, logrando obtener además soluciones realistas a los problemas admitiendo grandes cantidades de copias en las comunicaciones. Si bien se asumen características que simplifican la implementación, este trabajo sienta las bases para futuros trabajos que busquen acercar las condiciones iniciales de las redes a tratar, lo mas cerca de la realidad posible.

Notación del Algoritmo

Algoritmo COSTS Definiciones utilizadas en el pseudocódigo.

$DELAY(s)$ Proyección de la tercera posición del vector de costos del estado s. 16

D_s Conjunto de tuplas representando los casos de fallo exitosos, de la forma (probabilidad, delay). 16, 17

$ENERGY(s)$ Proyección de la segunda posición del vector de costos del estado s. 16

$SDP(s)$ Definido el vector de costos de s, es 1 menos la proyección de la primera posición del vector. 16

$copies\_in\_target(s, target)$ Cantidad de copias del estado s que están alojadas en el nodo target. 16

$energy\_of(R, s)$ Energía necesaria para tomar la acción R desde el estado s. 16

pr_{fs} Probabilidad de que todos los contactos en fs fallen. 16

$state\_after\_failures(R, s, fs)$ Estado obtenido partiendo de s tomando la acción R y considerando fs como conjunto de contactos que fallaron. 16

Algoritmo MORUCoP Definiciones utilizadas en el pseudocódigo.

C_{t_i} Conjunto de nodos que acarrean copias del paquete en el slot t_i. 14

$Cr(s')$ Costos de tomar la acción minimizadora de costos desde el estado s. 14

R_c Conjunto de transiciones hacia el nodo c, cumpliendo que $cp(c)$ copias llegan a c. 14

S_{t_i} Conjunto de estados en el slot t_i. 13

$T_r(s)$ Conjunto de acciones que llevan al estado s, cumpliendo que sus transiciones consideran *num_copies* cantidad de copias. 14

$best\_action(s)$ Acción minimizadora de costos para el estado s. 14

$contact_{C_{t_i}}$ Conjunto de contactos que comunican c' con c. 14

$cp(c)$ Numero de copias en el nodo c. 14

$get\_previus\_state(s, r)$ Estado anterior a tomar la accion r. 14

$pred^{+}_{C_{t_i}}$ Conjunto de nodos que pueden transmitir a c en el slot t_i. 14

$prev\_transition\_state(tr)$ Estado anterior considerando una única transición tr. 14

Bibliografía

ARANITI, GIUSEPPE; NIKOLAOS BEZIRGIANNIDIS; EDWARD BIRRANE; IGOR BISIO; SCOTT BURLEIGH; CARLO CAINI; MARIUS FELDMANN; MARIO MARCHESE; JOHN SEGUI; und KIYOHISA SUZUKI. 2015. Contact graph routing in DTN space networks: overview, enhancements and performance. *IEEE Communications Magazine* 53.38–46.

D'ARGENIO, PEDRO R.; JUAN A. FRAIRE; und ARND HARTMANNS. 2020. Sampling distributed schedulers for resilient space communication. *Nasa formal methods*, hrsg. von Ritchie Lee, Susmit Jha, Anastasia Mavridou, und Dimitra Giannakopoulou, 291–310. Cham: Springer International Publishing.

FALL, KEVIN. 2003. A delay-tolerant network architecture for challenged internets. *Proceedings of the 2003 conference on applications, technologies, architectures, and protocols for computer communications*, SIGCOMM '03, 27–34. New York, NY, USA: Association for Computing Machinery. URL `https://doi.org/10.1145/863955.863960`.

FELDMANN, MARIUS, und FELIX WALTER. 2017. Routing in ring road networks with limited topological knowledge. *2017 ieee international conference on wireless for space and extreme environments (wisee)*, 116–121.

FILAR, JERZY, und KOOS VRIEZE. 2012. *Competitive Markov decision processes*. Springer Science & Business Media.

FOUST, JEFF. 2019. Spacex's space-internet woes: Despite technical glitches, the company plans to launch the first of nearly 12,000 satellites in 2019. *IEEE Spectrum* 56.50–51.

FRAIRE, J., und J.M. FINOCHIETTO. 2015. Routing-aware fair contact plan design for predictable delay tolerant networks. *Ad Hoc Networks* 25.303–313, New Research Challenges in Mobile, Opportunistic and Delay-Tolerant Networks Energy-Aware Data Centers: Architecture, Infrastructure, and Communication. URL `https://www.sciencedirect.com/science/article/pii/S1570870514001371`.

FRAIRE, JUAN A.; OLIVIER DE JONCKÈRE; und SCOTT C. BURLEIGH. 2021. Routing in the space internet: A contact graph routing tutorial. *Journal of Network and Computer Applications* 174.102884. URL `https://www.sciencedirect.com/science/article/pii/S1084804520303489`.

FRAIRE, JUAN ANDRES; PABLO GUSTAVO MADOERY; M. BURLEIGH, S.; FELDMANN; und JORGE MANUEL FINOCHIETTO. 2017. Assessing contact graph routing performance and reliability in distributed satellite constellations. *Journal of Computer Networks and Communications*.

MADOERY, PABLO G.; FERNANDO D. RAVERTA; JUAN A. FRAIRE; und JORGE M. FINOCHIETTO. 2018. Routing in space delay tolerant networks under uncertain contact plans. *2018 ieee international conference on communications (icc)*, 1–6.

RAVERTA, FERNANDO D.; RAMIRO DEMASI; PABLO G. MADOERY; JUAN A. FRAIRE; JORGE M. FINOCHIETTO; und PEDRO R. D'ARGENIO. 2018. A Markov Decision Process for Routing in Space DTNs with Uncertain Contact Plans. *2018 6th ieee international conference on wireless for space and extreme environments (wisee)*, 189–194.

RAVERTA, FERNANDO D.; JUAN A. FRAIRE; PABLO G. MADOERY; RAMIRO A. DEMASI; JORGE M. FINOCHIETTO; und PEDRO R. D'ARGENIO. 2021. Routing in delay-tolerant networks under uncertain contact plans. *Ad Hoc Networks* 123.102663. URL `https://www.sciencedirect.com/science/article/pii/S1570870521001761`.

SCOTT, KEITH, und SCOTT C. BURLEIGH. 2007. Bundle Protocol Specification. RFC 5050. URL `https://www.rfc-editor.org/info/rfc5050`.

SPYROPOULOS, THRASYVOULOS; KONSTANTINOS PSOUNIS; und CAULIGI S. RAGHAVENDRA. 2005. Spray and wait: An efficient routing scheme for intermittently connected mobile networks. *Proceedings of the 2005 acm sigcomm workshop on delay-tolerant networking*, WDTN '05, 252–259. New York, NY, USA: Association for Computing Machinery. URL `https://doi.org/10.1145/1080139.1080143`.

TORGERSON, LEIGH; SCOTT C. BURLEIGH; HOWARD WEISS; ADRIAN J. HOOKE; KEVIN FALL; DR. VINTON G. CERF; KEITH SCOTT; und ROBERT C. DURST. 2007. Delay-Tolerant Networking Architecture. RFC 4838. URL `https://www.rfc-editor.org/info/rfc4838`.

TORRELLA, ULISES NICOLAS. 2023. Trabajo de grado Universidad Nacional de Córdoba, Análisis multiobjetivo sobre DTNs. URL `https://rdu.unc.edu.ar/handle/11086/548449`.