Generador de trazas para algoritmos de encaminamiento oportunista

Sergi Camós Belmonte

Resumen– Las redes oportunistas están concebidas para complementar la comunicación tradicional basada en infraestructura al permitir que los distintos dispositivos se comuniquen directamente entre sí cuando se encuentran dentro del rango de comunicación en lugar de a través de una red. Las redes oportunistas se consideran un medio de comunicación apropiado tanto en escenarios urbanos donde la red celular está sobrecargada, como en escenarios donde la infraestructura no está disponible, como en áreas escasamente pobladas y durante desastres. Sin embargo, las redes oportunistas aún no se han implementado de forma contundente en nuestro día a día. En este artículo aprenderemos que son las redes oportunistas, los desafíos que trae implementarlas y que hay que tener en cuenta para poderlas llevar a cabo. Además, como parte práctica se realizara un generador de trazas para redes oportunistas donde la traza resultante tendrá una característica concreta.

Palabras clave– Redes oportunistas, trazas, algoritmos genéticos, the one

Abstract– Opportunistic networks are designed to complement traditional infrastructure-based communication by allowing multiple devices to communicate directly with each other when within communication range rather than across a network. Opportunistic networks are considered an appropriate means of communication in urban settings where the cellular network is overloaded, as well as in settings where the infrastructure is not available, such as in sparsely populated areas and during disasters. However, opportunistic networks have not yet been forcefully implemented in our day-to-day lives. In this article we will learn what opportunistic networks are, the challenges that implementing them brings and what must be taken into account in order to carry them out. In addition, as a practical part, a trace generator will be made for opportunistic networks where the resulting trace will have a specific characteristic.

Keywords– Opportunistic networks, traces, genetics algorithms, the one

1 Introducción - Contexto del trabajo

El rendimiento de las tecnologías de comunicación actuales pueden verse seriamente afectadas por problemas de congestión. Para este tipo de problema se usan las Redes Oportunistas como una buena alternativa a la falta de capacidad en la red.

Las Redes Oportunistas[1] se basan en la oportunidad de establecer contacto entre pares de nodos para propagar mensajes. La efectividad de este tipo de redes depende principalmente del número de contactos y de la movilidad de los nodos.

• E-mail de contacte: sergi.camos@e-campus.uab.cat
• Menció realitzada: Tecnologies de la Informació
• Treball tutoritzat per: Diego Mauricio Freire Bastidas (Àrea de Ciències de la Computació i Intel·ligència Artificial)
• Curs 2021/22

Una Red Oportunista consta de diferentes elementos[2] para poderse llevar a cabo. Por una parte, están los nodos, mencionados anteriormente. Los nodos en este tipo de red suelen ser en su mayoría móviles, ya que mediante el uso de conexiones bluetooth y Wifi Direct se van mandando mensajes unos con otros, de ahí la importancia de su movilidad y duración. Por otro lado, están los protocolos de enrutamiento, que se ocupan de gestionar como se reenvían los mensajes, y mecanismos para priorizar el reenvío y descarte de mensajes por parte del nodo.
Este tipo de redes se caracterizan[3] por no tener una infraestructura centralizada que gestione el envío de mensajes, motivo por el cual las conexiones entre los diferentes nodos pueden llevar o no el mensaje a su destinatario. Además, como los nodos están en constante movimiento requiere que estos dispongan de almacenamiento prolongado para guardar los distintos mensajes hasta poderlos reenviar.

Hay simuladores de redes oportunistas[4] que nos ayudan a verificar cualquier teoría sobre el funcionamiento de las

redes, o para simplemente probar el funcionamiento de una red particular bajo ciertas condiciones. Los Simuladores, como por ejemplo el the ONE[5], utilizan trazas para evaluar diferentes protocolos de difusión a partir de escenarios reales con condiciones específicas.

La cantidad de trazas[6] o su relevancia es insuficiente porque son un conjunto de nodos fijos limitados a funcionar de forma esperada en un escenario concreto, por lo que no se podría usar una misma traza para diferentes escenarios, ya que cada uno tiene sus peculiaridades.

Este trabajo busca resolver el problema anterior con la generación de trazas de los diferentes tipos de algoritmos en las Redes Oportunistas, a partir de la entrada de unos datos que servirán para la creación de una traza con las propiedades que queremos.

1.1 Objetivos

El objetivo principal es la generación de trazas para el análisis de los diferentes algoritmos de las Redes Oportunistas, a partir de unas trazas iniciales que tienen diferentes características.

Utilizar técnicas Heurísticas para la modificación de una traza extrayendo parte de su contenido y añadiéndole fragmentos de otras trazas.

Crear un set de trazas creadas a partir de dividir el contenido de las trazas ”originales”y aplicarles las técnicas heurísticas.

Establecer que valores vamos a analizar de las trazas para saber si tienen las características que queremos y en caso negativo crear nuevas trazas.

Decidir que algoritmos de las Redes Oportunistas vamos a usar para analizar el comportamiento de las trazas generadas mediante el uso del simulador the ONE.

Documentar los resultados obtenidos por la simulación de la traza con los diferentes algoritmos y decidir que algoritmo funciona mejor a partir de comparar algunos valores de los resultados.

1.2 Metodología

Para llevar a cabo este proyecto de forma organizada se utilizará la metodología de Kanban[7]. Esta metodología se considera ”ágil”y consiste en organizar los diferentes pasos que hay que realizar en tres estados: hecho, en proceso y sin empezar. Para llevar a cabo esta metodología se utilizará la plataforma ClickUp[8].

Además, para la ejecución de gran parte del proyecto se usará la herramienta de simulación The ONE. Con esta herramienta podremos hacer los diferentes estudios sobre las trazas a partir de ejecutarlas y posteriormente observar los resultados obtenidos en los reports.

1.3 Planificación

Para realizar la planificación del trabajo y tener un cierto orden y control de los tiempos, hemos usado Gantt[9] para tener una cronología. Gracias a su uso tenemos una mejor visión sobre las faenas a realizar para conseguir hacer los objetivos, las dependencias entre las tareas y los tiempos de estas, además del tiempo por objetivo y el global. Esto nos ayudará a ver que tareas son las más importantes y si estamos cumpliendo los tiempos.

Para definir los pasos a seguir en el proyecto he dividido los objetivos en los siguientes apartados.

1. **División de las trazas originales.**
    - Obtención de las trazas originales
    - Decidir metodos de partición
    - Realizar un ejecutable dedicado a hacer la partición
    - Comprovar que se hace la partición correctamente
2. **Medir las características de las trazas**
    - Analizar que características queremos comparar
    - Ejecutar la traza y obtener los resultados
    - Comparar los resultados obtenidos con los esperados
3. **Generar nuevas trazas**
    - Analizar las diferentes heurísticas
    - Decidir que heurística usar
    - Unir fragmetos de trazas
    - Volver a medir
4. **Experimentación**
    - Comprovar que la traza tiene las medidas esperadas
    - Decidir que algoritmos de encaminamiento oportunista analizar
    - Decidir que datos comparar
    - Ejecutar diferentes algoritmos
    - Conclusión a partir de los datos

Los diagramas de Gantt que muestran la cronología del proyecto se encuentran en el apéndice A.1, que está compuesta por 4 figuras.

Antes de explicar los diferentes procesos de ejecución que se desarrollaran en el proyecto a partir de los diagramas de Gant quiero dejar constancia del trabajo previo que ha habido. Este trabajo previo ha constado de la cerca de información sobre que son las redes oportunistas, tutoriales de The ONE para entender como funciona el simulador, su instalación y la de los diferentes componentes necesarios para que se pueda ejecutar.

Una vez hecho el trabajo previo entraríamos a hacer el primer objetivo[Figura 1]. Este tendrá una duración de 12 días y empezaría con la obtención de las trazas originales, cercar sobre que son las trazas y el análisis del contenido de estas. Una vez obtenemos las trazas decidimos como vamos a partir las trazas en grupos más pequeños, sea teniendo en cuenta su contenido o dividiéndolo en un número concreto de fragmentos. Finalmente sabiendo que método de corte nos gustaría usar realizamos un ejecutable que haga dicha faena y posteriormente comprobamos si funciona correctamente.

Para el segundo objetivo [Figura 2] se necesitaría un tiempo estimado de 12 días. En este tramo del proyecto nos centraríamos en informarnos de los diferentes formas de medición de The ONE a partir de los reports y una vez se supiera sobre el tema decidir que métricas usar. Sabiendo que datos se quieren obtener ejecutaríamos las trazas con el simulador y posteriormente se compararían con los datos que se quieren obtener.

El tercer objetivo [Figura 3] trata de generar nuevas trazas a partir de los fragmentos generados en el primer proceso [Figura 1] en caso de que las métricas no sean las esperadas. Las dos primeras tareas tratan sobre buscar información sobre diferentes heurísticas y luego decidir cuál usar, en este caso simulated Annealing o algoritmos genéticos. Una vez se sabe cuál utilizar se realizaría la unión de diferentes trazas y se volvería a realizar mediciones sobre la traza generada para ver si cumple con lo esperado.

Finalmente, llegamos a la última parte del proyecto [Figura 4] donde una vez se obtiene la traza con las características deseadas se hace un estudio sobre las diferentes Redes oportunistas que ofrece The ONE y que métricas se pueden usar para compararlas. Una vez se realiza este estudio se decide que conjunto de redes queremos analizar, los datos a comparar y se ejecuta el simulador empleando la traza y guardando los datos obtenidos de las diferentes redes. Por último se hace una conclusión de los datos y se dictamina que red oportunista es mejor para la traza que hemos utilizado.

2 REDES OPORTUNISTAS

Las Redes Ooprtunistas[11] se basan en comunicaciones espontáneas entre distintos dispositivos inalámbricos, este nuevo tipo de redes difiere de sus predecesoras porque no utiliza un camino fijo para la comunicación entre estos nodos. A diferencia de otras redes el recorrida de un nodo A a un nodo B puede no existir o en caso que si exista no es un camino fijo, puede variar con el tiempo. Los dispositivos han de tener un espacio de almacenamiento reservado para guardar los mensajes y hacen la función de carry, que es llevar el mensaje consigo hasta entrar en contancto con un nodo y la función de forward que si el protocolo de enrutamiento lo permite entregar el mensaje al nodo con el que hemos entrado en contacto.

Su rendimiento depende principalmente de la movilidad de los nodos y de los algoritmos de enrutamiento, siendo estos los encargados de decidir como son intercambiados los mensajes cuando un contacto ocurre, intentando encontrar la mejor ruta para que un mensaje alcance su destino. Además, dado que el delivey depende de la movilidad y la conexión entre nodos sin una ruta fijada, no está garantizado que los mensajes lleguen a su destinatario.

Como se ha mencionado antes, estas redes no usan una conexión a Internet, ni ningún tipo de infraestructura fija para realizar el envío de información entre sus distintos componentes. Podemos considerar a este tipo de redes como una buena alternativa para su utilización en muchos escenarios, como áreas geográficamente remotas, en donde no es posible desplegar o no está disponible ningún tipo de sistema de telecomunicación.

2.1 Protocolos de enrutamiento

Para gestionar como se enviaran los mensajes en las Redes Oportunistas se usan los protocolos de Enrutamiento. Estos protocolos son encargados del proceso de intercambio de información cuando los nodos están en contacto y se pueden clasificar[12] según su uso o no de infraestructura, sus aplicaciones, la cantidad de información que manejan, etc.

En este caso las clasificaremos según el número de copias:

1. **Single Copy:** Aquellos que mantienen una sola copia del mensaje en la red hasta que sea entregado a su destino.
    - **First Contact:**[13] Los nodos transfieren el mensaje al primer nodo con el que se establezca contacto, repitiendo el proceso hasta que el mensaje llegue al nodo destino.
    - **Direct Deivery:**[14] El mensaje será enviado sí y solo si el nodo con el que se ha establecido contacto es el nodo destino
2. **Broadcast:** Se propagan el mensaje por toda la red.
    - **Epidemic:**[15] Se realiza una copia del mensaje para todos los nodos contactados, incrementando enormemente la posibilidad de entregar el mensaje. Los principales problema de este algoritmo son la enorme sobrecarga en la red y la alta demanda de almacenamiento en los nodos.
    - **Spray and Wait:**[16] Los nodos origen del mensaje asignan un número máximo de copias del mensaje que se pueden crear, por lo que solo pueden existir cierto número de copias del mensaje en la red. Primero se realiza el "Spray", que consiste en que los nodos origen dan una copia a sus contactos y luego la fase del "Wait", donde se espera a que el mensaje llegue al destinatario.
3. **Probabilísticos:** Se basan en la transmisión de copias del mensaje a los nodos que tienen mas posibilidades de contactar con el nodo destino.
    - **PRoPHET:**[17] Determina qué nodos son los mejores para el reenvío del mensaje para que llegue al destino final. Usando estos nodos para ha-

cer el reenvío podemos reducir el número de copias creadas, incrementando la probabilidad de que el mensaje llegue a su destino.

- **MaxProp:**[18] Difunde un mensaje sobre la red, haciendo una validación de todas las rutas posibles para cada mensaje, almacena el número de saltos y la probabilidad de que el mensaje llegué a su destino.

4. **Otros protocolos:** consideran aspectos como la sociabilidad para la selección de los mejores nodos para hacer reenvíos selectivos de mensajes.

    - **Randomized Rumor Spreading:**[19] cuando un nodo esta en contacto con otros nodos, se selecciona aleatoriamente un mensaje de los que tiene en cola o cache y lo envía a sus vecinos. Ninguno de los nodos que recibe o reenvía el mensaje sabe que mensajes tiene.
    - **BubbleRap:**[20] Cuando los mensajes son difundidos en la red, se prefieren los nodos con mayor centralidad y que sean de la misma comunidad, asumiendo que esos nodos se encontrarán antes con el destino final de la información.
    - **The Friendship and Selfishness Forwarding Algorithm:**[21] Se realiza una validación de los nodos contactados, teniendo en cuenta varios aspectos como: si los nodos trabajan juntos, viven cerca o coinciden en algunos lugares, antes de realizar el reenvío del mensaje.

2.2 Escenarios

Para poder llevar a cabo la simulación de las redes oportunistas es necesario el uso de escenarios[22]. Estos escenarios suelen ser recreaciones en 2D de lugares reales o parecidos donde se tiene interés en llevar a cabo el uso de una red oportunista concreta. En estos escenarios se pueden crear objetos como edificios, obstáculos fijos, obstáculos móviles, caminos, etc.También se puede gestionar el número de nodos que puede haber, si son fijos o se intercambian con nuevos nodos, decidir donde agrupar los nodos y la movilidad que estos pueden tener en el escenario. Para la simuladión de escenarios hay diferentes herramientas y un ejemplo de ellas es PedSim[23]. En este trabajo no analizaremos el uso de dichas herramientas,, pero es importante saber que también forman parte a la hora de ejecutar una simulación.

2.3 Trazas

Para hacer mas realista la simulación de una Red Oportunista se usan trazas. Estas trazas son un conjunto de instrucciones que generan las conexiones y desconexiones de diferentes pares de nodos en cietos momentos de tiempo, simulando el trafico de datos. En estas intrucciones se marca el tiempo cuando interactuan los dos nodos, si se conectan o desconectan i la ID de los dos nodos involucrados.

2.3.1 Ejecución de trazas

Al querer ejecutar las trazas en el simulador ONE, estas solo aportan información sobre las conexiones entre los nodos en diferentes instancias de tiempo, pero no proporcionan más información sobre los nodos, haciendo que no se cubran todas las necesidades a la hora de ejecutarlas. Las características que faltan se podrían catalogar en dos grupos: Eventos y nodos.

En el caso de las características relacionadas con los eventos, no se nos da información de la generación de los mensajes, como el intervalo de tiempo de los mensajes, el tamaño de los mensajes i el total de direcciones, ID totales para representar a los nodos, que son necesarias. Por el otro lado están las características de los nodos, que no nos indican la velocidad a la que se mueven, el rango de conexión que tienen, que tipo de conexión usan y que características tiene, el tamaño de memoria de los nodos, el movimiento que realizan los nodos y el TTL que le asignan a los mensajes.

Para solucionar toda esta falta de información y características que han de tener los nodos se usan los documentos de configuración. Estos documentos de configuración son archivos de texto donde se escriben toda la información que no aportan las trazas, además de llamar a las trazas, e información sobre el escenario, como el nombre, si genera las conexiones o no y el número de grupos de nodos que hay. Además de decidir si queremos recibir reportes de la simulación, que tipo de información queremos reportar y donde.

2.3.2 Problema de las trazas

Como hemos comentado en el apartado anterior, las trazas no nos proporcionan toda la información necesaria para ejecutarlas, pero tampoco nos proporcionan características propias que nos ayuden a poder clasificarlas o a seleccionarlas según los resultados que queramos obtener, como por ejemplo el ICT. Una característica común en las trazas que nos ayudará a seleccionarlas según el valor que tengan. Esta característica se obtiene con el uso de matrices de contactos.

3 USOS DE LAS REDES OPORTUNISTAS

Hay varias aplicaciones que pueden beneficiarse de la computación oportunista. Entre ellas podemos encontrar la computación oportunista[24] que utiliza recursos, servicios y aplicaciones compartidas para realizar operaciones distribuidas en relación a una tarea. También podemos encontrar su uso en aplicaciones cotidianas como sistemas de recomendación[25] que gracias a las redes oportunistas pueden rastrear las actividades de los usuarios y los patrones de movilidad y usar esta información para dar recomendaciones sobre varios elementos.
Como hemos explicado anteriormente las redes oportunistas ayudan a reducir la carga de trabajo de las redes y una ayuda que proporcionan es la descarga de datos móviles, que debido a un aumento en la cantidad de teléfonos inteligentes, las redes 3G y 4G están sobrecargadas. Entonces, la descarga de datos móviles en redes oportunistas se usa para reducir la carga en dichas redes.

También hay otros usos menos conocidos, pero que son muy importantes como la gestión de crisis[26] a partir de nodos, cuando las redes de comunicación tradicionales

no se pueden utilizar debido a imprevistos de distinto escalado y eventos disruptivos. Por lo tanto, las redes oportunistas se pueden usar para interconectar las partes de la red de telecomunicaciones y se puede utilizar para redes específicas.

No solo son útiles en los ámbitos de comunicación y redes, sino también en el ámbito médico proporcionando atención médica generalizada[26] donde las redes oportunistas se pueden usar para crear un sistema de dispositivos inteligentes que pueden mantener un registro de los alrededores de los pacientes. Esto puede incluir redes de área corporal (BAN). Las redes oportunistas se puede utilizar para monitorear parámetros físicos y fisiológicos.

4 SEGURIDAD Y PRIVACIDAD

Las redes oportunistas no son perfectas y como toda red tiene sus problemas de seguridad y privacidad[27], pero antes de explicar cada uno de ellos hay que entender las diferentes partes del funcionamiento de estas redes que son vulnerables a estos problemas.

Para empezar hay que recordad que las redes oportunistas son descentralizadas y heterogéneas. Esto hace que este formada por varios tipos de dispositivos, como celulares, teléfonos, sensores, cámaras, etc. Estos dispositivos pueden depender de diferentes tecnologías, lo que da lugar a problema de interoperabilidad.
Como se trata una red de conectividad intermitente, al haber gran movilidad de los nodos de extremo a extremo, no se puede establecer la conexión y falta el conocimiento previo sobre la información de la red. Por lo tanto, los protocolos de enrutamiento ad-hoc tradicionales no se pueden utilizar en el caso de redes oportunistas. Este tipo de conectividad da la posibilidad de que un nodo entre en contacto con otro nodo en un momento imprevisto.
El almacenamiento es algo importante, ya que los nodos intermedios deben tener suficiente espacio de almacenamiento para almacenar los mensajes hasta que hacen un contacto oportunista con otro nodo. Si no hay suficiente espacio de almacenamiento, entonces los paquetes se pueden descartar y, por lo tanto, se puede perder información útil.

Una vez tenemos en mente las características básicas de las redes oportunistas, podemos hablar de los problemas de seguridad y privacidad. Uno de los desafíos a superar es asegurar el enrutamiento, para ello se debe mantener una lista de dispositivos confiables. Pueden ser diferentes tipos de nodos como comisarías, oficinas de gobierno, hospitales, universidades, etc. Se debe elegir la ruta que pase a través de dispositivos de máxima confianza, pero esto es muy difícil debido a la conectividad intermitente y para evitar fallos de seguridad se usan claves secretas y firmas.
Otro problema a tener en cuenta es la privacidad de los nodos y los datos. La privacidad de un nodo se puede garantizar mediante autenticación y autorización, prevención de intrusiones y detección de intrusiones. Esto es necesario, ya que nodos maliciosos pueden unirse a la red. En cuanto a los datos, el uso del cifrado es una forma de proporcionar privacidad de datos. La criptografía de clave pública se puede utilizar este caso. El controlador puede cifrar datos con clave pública y los dispositivos pueden descifrarlos con su clave privada. Se necesita un mecanismo seguro para la transmisión de la clave pública, de lo contrario, un dispositivo malicioso puede distribuir también su propia clave pública. En relación con los datos hay que mirar también su integridad, para ello se pueden usar firmas digitales.

Finalmente, hay que tener en cuenta los diferentes ataques que puede sufrir la red[28]:

1. **Man in the middle:** Una persona envía una solicitud de ayuda al controlador, entonces un nodo malicioso no envía la solicitud, pero le asegura a la persona que la ayuda está en camino y le envía un mensaje malicioso. Para resolver este problema, la persona puede enviar mensajes redundantes al controlador con la ayuda de varios vecinos.

2. **Packet dropping:** El nodo malicioso puede descartar los paquetes. Para resolver este ataque, se puede enviar mensajes redundantes a través de diferentes vecinos al controlador.

3. **Denial of Service:** Los nodos maliciosos pueden generar solicitudes falsas y debido a las cuales la red no está disponible para emergencias reales. Para resolver este ataque, se puede colocar un límite sobre el número total de solicitudes que los dispositivos pueden enviar.

4. **ID spoofing:** Un nodo malicioso puede generar solicitudes con muchas identificaciones. Para solucionar este ataque, los nodos necesitan controlar a sus vecinos en busca de suplantación de identidad.

5 GENERACIÓN DE TRAZAS

Como solución al problema de la creación de trazas con características concretes a partir d'un elemento común entre ellas, programaremos un generador de trazas. La idea principal de este generador[Figura 5] es que a partir de las trazas originales que tenemos nosotros introducimos un input. Este input será una característica relacionada con la información obtenida de los reports de nuestras trazas al ejecutarlas en el simulador ONE. Una vez introducido el input, el generador recorrerá una tabla con la información de nuestras trazas y se quedará con las que cumplan con la característica del input. En el caso que el input no exista en la tabla nos pedirá que introduzcamos uno nuevo.

Una vez recorrida la tabla y seleccionadas las trazas, estas serán las que usaremos como parents para la generación de nuevas trazas a partir de algoritmos genéticos[29]. Mediante el uso de estos algoritmos crearemos nuevas trazas que al ser ejecutadas, en el report se pueda observar que cumplen con el input introducido. Esto se realizará hasta obtener la traza esperada y el output del generador será la traza y el archivo de configuración que hemos utilizado para ejecutarla en el simulador ONE.

5.1 Algoritmos genéticos

Los algoritmos genéticos son algoritmos de búsqueda basados en la mecánica de selección natural y de la genética natural. Voy a explicar su funcionamiento usando como ejemplo las trazas que es de lo que consiste este trabajo.

Los algoritmos genéticos combinan conjuntos de trazas para formar nuevos con características parecidas o diferentes a sus trazas padres. Estos algoritmos nos ayudaran a crear nuevas trazas que se ajusten a nuestras necesidades. Para el paso de crear nuevas trazas se aplican una serie de operadores genéticos[30]:

1. **Selección:** Encargados de escoger qué trazas van a disponer de oportunidad de reproducirse y cuáles no.

    - **Selección por ruletat:** A cada traza se le asigna una parte proporcional a su ajuste de una ruleta, de tal forma que la suma de todos los porcentajes sea la unidad. Las mejores trazas recibirán una porción de la ruleta mayor que la recibida por los peores.
    - **Selección por torneo:** Consiste en escoger a las diferentes trazass en base a comparaciones directas entre sus características. Se selecciona al azar un número x de trazas y de entre las trazas seleccionadas se selecciona la más apta para pasarla a la siguiente generación.

2. **Cruce:** Las trazas seleccionadas son recombinadas para producir la descendencia que se insertará en la siguiente generación.

    - **Cruce de 1 punto:** Una vez seleccionadas dos trazas se cortan sus instrucciones por un punto seleccionado aleatoriamente para generar dos segmentos diferenciados en cada uno de ellos: la cabeza y la cola. Se intercambian las colas entre las dos trazas para generar los nuevos descendientes.
    - **Cruce de 2 puntos:** En vez de cortar por un único punto como en el caso anterior, se realizan dos cortes.Para generar la descendencia se escoge el segmento central de uno de las trazas y los segmentos laterales de la otra.

3. **Algoritmos de reemplazo:** Si se trabaja con un conjunto de trazas fijas, sobre la que se realizan las selecciones e inserciones, deberá tenerse en cuenta que para insertar un nuevo individuo deberá de eliminarse previamente otro de la población.

    - **Aleatorio:** La traza substituye otra de manera aleatoria dentro del conjunto.
    - **Reemplazo de padres:** Se añade la nueva traza al conjunto, pero se elimina a los padres.
    - **Reemplazo de similares::** Se selecciona un grupo de trazas con características similares y se reemplazan aleatoriamente las que sean necesarias.
    - **Reemplazo de los peores:** Se eliminan las trazas que por porcentaje son peores a la hora de conseguir el resultado esperado para añadir las nuevas trazas.

4. **Copia:** Consiste en la copia de una traza en la nueva generación. Esto se suele utilizar han quedado trazas sin pasar por el proceso de cruce.

5. **Elitismo:** Parecido al operador genético de copia, pero en este caso solo se seleccionan las trazas con mayor porcentaje. De esta manera se garantiza que el proceso de búsqueda no dará resultados negativos y se podra reconducir a encontrar mejores resultados.

6. **Mutación:** La mutación de una traza provoca que alguno de sus características, generalmente una sola, varíe su valor de forma aleatoria.

6 PARTE PRÁCTICA

En este momento, dentro de la planificación que había estipulado al principio de este proyecto, nos encontramos en la fase de Experimentación, pero habrá que volver a la fase de anterior, en este caso generar nuevas trazas, para rehacer la parte de código encargada de la heurística, ya que el resultado no es el esperado.
En cuanto a los cambios que se han producido en la práctica hay que destacar que se reduce la base de datos que obtuvimos a la mitad, ya que solo trabajaremos con trazas de ocho horas y descartaremos las de cuatro horas por problemas de rendimiento del ordenador, porque tarda demasiado tiempo en ejecutar la gran cantidad de trazas que tenemos.

6.1 Metodología

El primer paso que he hecho ha sido tratar la base de datos que se me ha proporcionado al principio separando las trazas originales en pequeños fragmentos que representan trazas de cuatro y ocho horas. Para hacer esto posible utilizo un script que he llamado separar-traces.py que recorre una traza original que yo paso como parámetro y guarda el contenido en una lista.
Mientras recorre la traza original va abriendo documentos de texto donde introduce un conjunto de nodos que van desde el tiempo 0 hasta el segundo 14.400 o 28.800 según el tiempo que queramos y estos documentos se les asigna un nombre compuesto por el nombre de la traza original más el tiempo querido y el índice que va del uno hasta finalizar los fragmentos[Figura 6].

Una vez tenemos las trazas divididas en cuatro y ocho horas trabajaremos con las de ocho horas por los motivos explicados anteriormente. Lo siguiente que hay que realizar es ejecutar estas trazas usando la propagación epidemic y spary and wait. Para hacer esto hice el script ejecutar-simulacion.py, este ejecutable espera que le pasemos por parámetro el nombre de la traza original y el método de propagación.
Una vez le entran los parámetros va al directorio de trazas-separadas y accede a la carpeta con el nombre de la traza original que hemos pasado por parámetro y recorre las trazas que hay dentro del directorio. Mientras recorre

las diferentes trazas, según el método de propagación que hemos decidido modifica el contenido del documento epidemic-settings.txt o snw-settings.txt que es el que utilizara the one para ejecutar las trazas[Figura 7].
En el caso del epidemic lo ejecutará una vez mientras el spray and wait tres, ya que queremos que lo realice con dos, ocho y veinticuatro copias. Los resultados obtenidos se guardaran en una carpeta dentro de log llamada reports, donde en su interior estará el nombre de la traza original y en guardar carpetas con el nombre de los fragmentos que contendrán sus respectivos reportes separados en epidemic y spray and wait.

Una vez tenemos los reports necesitamos adjuntar la información de estos para facilitar su comparación y a su vez la creación de trazas que tengan datos que no se encuentran en el espectro proporcionado por las trazas fragmentadas. Para ello uso un script llamado generador-reporte.py que espera por parámetro el nombre de la traza origen.
Al recibir el parámetro de entrada entra en el directorio de reportes y accede a la carpeta con el nombre de la traza origen. Una vez dentro, mientras recorre las trazas, creo un documento de texto donde se escribirá los resultados en orden. Es decir, primero ira el nombre de la traza y seguido cogerá la primera variable que ofrecen los reportes y pondrá el resultado en orden, primero epidemic y luego el spary and wait de dos, ocho y veinticuatro copias una vez acabado con una traza, escribe abajo los resultados de la siguiente y así hasta finalizar el recorrido de la carpeta[Figura 8].
Esto se puede hacer porque los reportes del the one aportan la misma información independientemente del método de propagación que se utilice. Además, estos reportes generales se guardan en una nueva carpeta dentro de log llamada final-report.

Ahora que tenemos los report finales se pueden crear las nuevas trazas. Para la creación de las nuevas trazas he utilizado el algoritmo genético cruce de un punto y lo he implementado en el script nueva-traza.py. Este script recorre los reportes finales y busca los nombres de las trazas originales que se usaran como para realizar el algoritmo.
Una vez encuentra las trazas, guarda en una lista sus nombres y selecciona un par de forma aleatoria generando una traza nueva a partir del contenido de las elegidas. Para realizar el cruce lo que sugiere el algoritmo es que se escoja de forma aleatoria un porcentaje del contenido de la primera traza y el contenido faltante se rellene con la segunda traza, en mi caso he decidido que de la primera traza se pueda escoger de forma aleatoria entre un 20 y un 80 por ciento para que independientemente de que extremo pueda darse haya suficiente contenido de una de las partes como para observar cambios.

Una vez creada la traza, elimina a los padres de la lista para evitar que sean usados al crear la nueva generación de trazas y mientras se van creando se ejecutan según el protocolo de propagación pasado por parámetro y se revisan los reports. En el caso de que la traza generada tenga el valor esperado se detiene el script y en la carpeta output, dentro de log estará la traza y la configuración para ejecutarla.

Cabe destacar que cada generación de trazas se crea a partir de la anterior y en el cado de llegar a la última generación donde solo existe una traza, se borran las generaciones anteriores y se vuelve a empezar desde las trazas originales, originando un nuevo conjunto de trazas que quizás nos dan el resultado esperado. [Figura 9]

7 EXPERIMENTACIÓN

Tras haber ejecutado el script de nueva-traza.py varias veces me he dado cuenta de tres factores. El primero es que el proyecto realiza el algoritmo correctamente, desde la selección de las trazas, pasando por la creación de las nuevas generaciones y finalizando con la aportación de resultados. El segundo es que a diferencia de la idea anterior, al usar padres de forma aleatoria obtengo valores fuera del rango ya obtenido por parte de las trazas originales. Esto nos hace ver que el algoritmo funciona, ya que la idea es obtener trazas con valores fuera del rango conocido. El tercer punto es que depende del valor que queramos conseguir se ejecutara más o menos recorrido. Un ejemplo práctico, seria que si buscamos una traza con un valor determinado de delivery quizás no llega a necesitar la creación de trazas de segunda generación. Mientras que si buscamos probabilidades si es necesario más generaciones de trazas. Esto me hace ver que es más fácil obtener resultados de enteros que probabilidades y flotantes.

8 RESULTADOS

Los resultados que obtenemos al finalizar la ejecución del script es dentro de la carpeta log, en el directorio output, dos carpetas. En una de ellas estará la traza obtenida a partir del valor característico que buscábamos y en la otra carpeta las configuraciones de la propagación de epidemic y spray and wait. De esta forma podremos observar si da los resultados esperados ejecutándolo de forma individual en el the one con la propagación esperada y a su vez ver que resultados nos da al usar la otra configuración. Tras ver los resultados obtenidos podemos decir que hemos cumplido con la idea del proyecto, ya que realiza lo que teníamos pensado a partir de pseudocódigo. [Figura5]

9 CONCLUSIÓN

Como conclusión cabe destacar la importancia que tiene el uso de algoritmos genéricos para crear trazas con nuevos valores. Porque incluso tras fragmentar las trazas originales, que de cada unos se obtenían 500 trazas nuevas, aún hay huecos en el espectro. Es por este motivo la importancia que tienen los algoritmos genéricos, porque amplían el espectro y así según que condiciones queramos tendremos una traza con los valores específicos para esos resultados.

REFERÈNCIES

[1] L. Pelusi, A. Passarella y M. Conti. “Opportunistic networking: data forwarding in disconnected mobile ad hoc networks.”, 2006.

[2] Chancay Garcia, Leonardo Javier. "Evaluación y modelado de redes oportunistas."Diss. Universitat Politècnica de València, 2019.

[3] García Giménez, Miguel. Ïmplementación y evaluación de soluciones de anonimato en redes oportunistas.", 2019.

[4] Garcia Robles, David. "Evaluación de algoritmos de propagación de mensajes en redes oportunistas". BS thesis. Universitat Politècnica de Catalunya, 2013

[5] Vinent Mercadal, Irene, and Ramon Martí Escalé. "Agents mòbils i DTN: disseny i implementació d'un protocol d'encaminament en l'entorn PROSES.", 2010.

[6] Chancay-Garcıa, Leonardo, et al. "Evaluación de Protocolos de Encaminamiento para Redes Oportunistas en Escenarios con Alta Renovación de personas.", 2018.

[7] Kniberg, Henrik, et al. "Kanban y Scrum–obteniendo lo mejor de ambos."Prólogo de Mary Poppendieck and David Anderson. ESTADOS UNIDOS DE AMÉRICA: C4Media Inc, 2010.

[8] ClickUP. Retrieved February 28, 2022, from ClickUP.com website: https://clickup.com/

[9] Hinojosa, María Alejandra. "Diagrama de Gantt."Producción, procesos y operaciones, 2003.

[10] Huang, Chung-Ming, Kun-chan Lan, and Chang-Zhou Tsai. "A survey of opportunistic networks."22nd International conference on advanced information networking and applications-workshops (aina workshops 2008). IEEE, 2008.

[11] hancay Garcia, Leonardo Javier. Evaluación y modelado de redes oportunistas. Diss. Universitat Politècnica de València, 2019.

[12] . Liu, B. Zhang, H. T. Mouftah, X. Shen and J. Ma, "Opportunistic routing for wireless ad hoc and sensor networks: Present and future directions,ïn IEEE Communications Magazine, vol. 47, no. 12, pp. 103-109, Dec. 2009, doi: 10.1109/MCOM.2009.5350376.

[13] rea Hueso, Carlos. "Modelización de la difusión y persistencia de datos en redes oportunistas de comunicación móvil."(2019).

[14] rossglauser, Matthias, and David NC Tse. "Mobility increases the capacity of ad hoc wireless networks.ÏEEE/ACM transactions on networking 10.4 (2002): 477-486.

[15] ain, Sushant, Kevin Fall, and Rabin Patra. "Routing in a delay tolerant network."Proceedings of the 2004 conference on Applications, technologies, architectures, and protocols for computer communications. 2004.

[16] amanathan, Ram, et al. "Prioritized epidemic routing for opportunistic networks."Proceedings of the 1st international MobiSys workshop on Mobile opportunistic networking. 2007.

[17] indgren, Anders, Avri Doria, and Olov Schelén. "Probabilistic routing in intermittently connected networks."ACM SIGMOBILE mobile computing and communications review 7.3 (2003): 19-20.

[18] indgren, Anders, et al. "Probabilistic routing protocol for intermittently connected networks."RFC Series (2012).

[19] ouza, Camilo, et al. "Fsf: Friendship and selfishness forwarding for delay tolerant networks."2016 IEEE Symposium on Computers and Communication (ISCC). IEEE, 2016.

[20] urgess, John, et al. "MaxProp: Routing for Vehicle-Based Disruption-Tolerant Networks.Ïnfocom. Vol. 6. 2006.

[21] ui, Pan, Jon Crowcroft, and Eiko Yoneki. "Bubble rap: Social-based forwarding in delay-tolerant networks.ÏEEE transactions on mobile computing 10.11 (2010): 1576-1589.

[22] hancay-García, Leonardo, et al. Ïnformation dissemination using opportunistic networks in scenarios with people renewal."2018 14th International Wireless Communications and Mobile Computing Conference (IWCMC). IEEE, 2018.

[23] es Bachelor-Grades, Bachelorarbeit zur Erlangung. "Der Einfluss der Geometrie auf Crowd-Desaster-Szenarien: Untersuchungen basierend auf Erweiterung der Open-Source Crowd-Simulation Software PEDSIM um eine neuartige Dichtefunktion."

[24] onti, Marco, and Mohan Kumar. "Opportunities in opportunistic computing.Çomputer 43.01 (2010): 42-50.

[25] as, Debashis, Laxman Sahoo, and Sujoy Datta. "A survey on recommendation system.İnternational Journal of Computer Applications 160.7 (2017).

[26] aur, Navneet, and Gauri Mathur. "Opportunistic networks: A review.ÏOSR Journal of Computer Engineering (IOSR-ICE) 18.2 (2016): 20-26.

[27] ilien, Leszek, et al. "The concept of opportunistic networks and their research challenges in privacy and security."Mobile and wireless network security and privacy (2007): 85-117.

[28] u, Yue, et al. "Security and trust management in opportunistic networks: a survey."Security and Communication Networks 8.9 (2015): 1812-1827.

[29] uri, Ángel, and José Galaviz. Algoritmos genéticos. No. Sirsi) i9789681663834. IPN, 2002.

[30] arcos; Rivero Gestal (Daniel; Rabuñal, Juan Ramón; Dorado, Julián; Pazos, Alejandro), and Marcos Gestal. Introducción a los algoritmos genéticos y la programación genética. Coruña: Universidade da Coruña, 2010.

APÉNDICE

A.1 Cronología de los objetivos

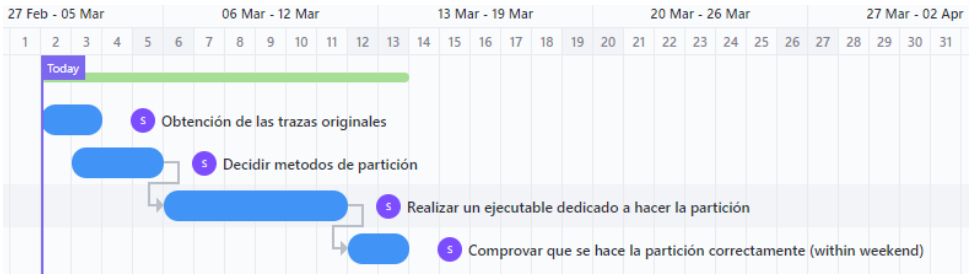

Fig. 1: Gantt división de las trazas originales

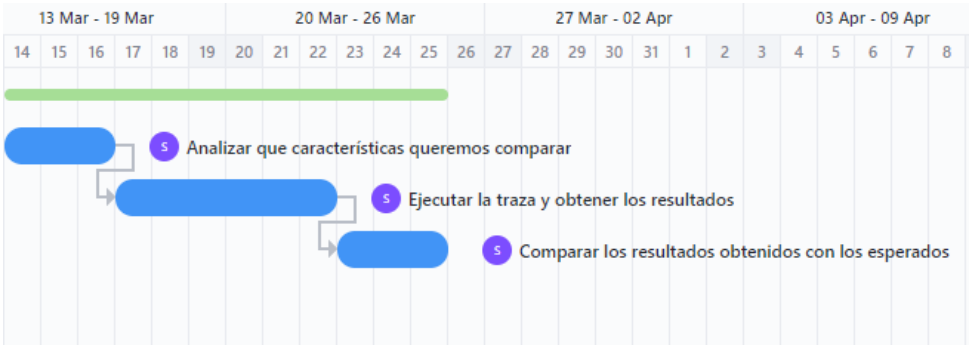

Fig. 2: Gantt medir las características de las trazas

Fig. 3: Gantt generar nuevas trazas

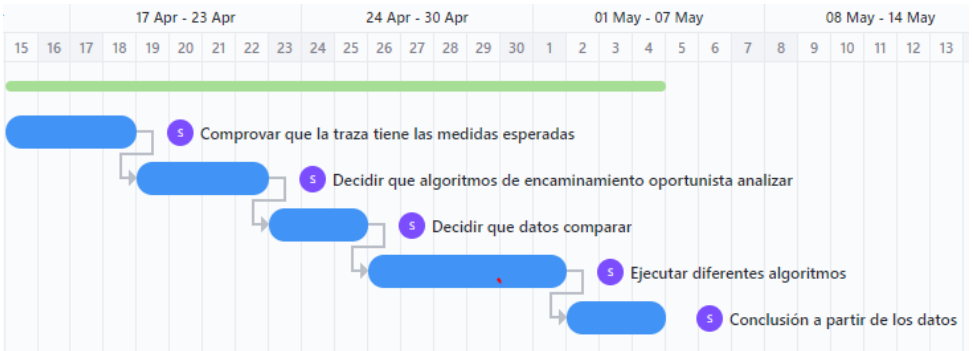

Fig. 4: Gantt experimentación

B.2 Imágenes generador de trazas

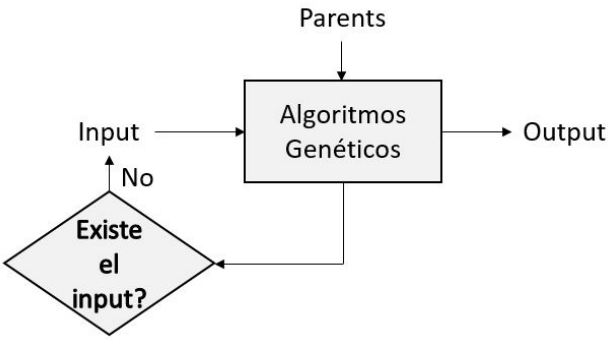

Fig. 5: Modelo de generación de trazas

C.3 Imágenes parte práctica

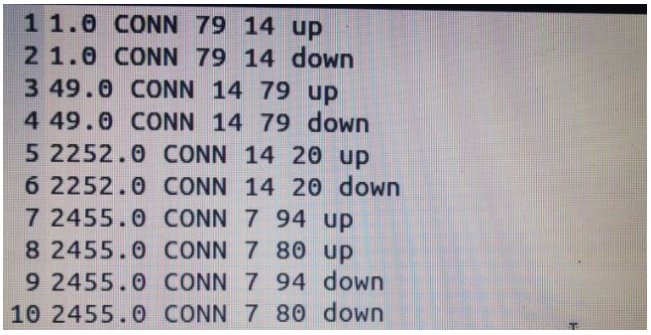

Fig. 6: Fragmento de una traza

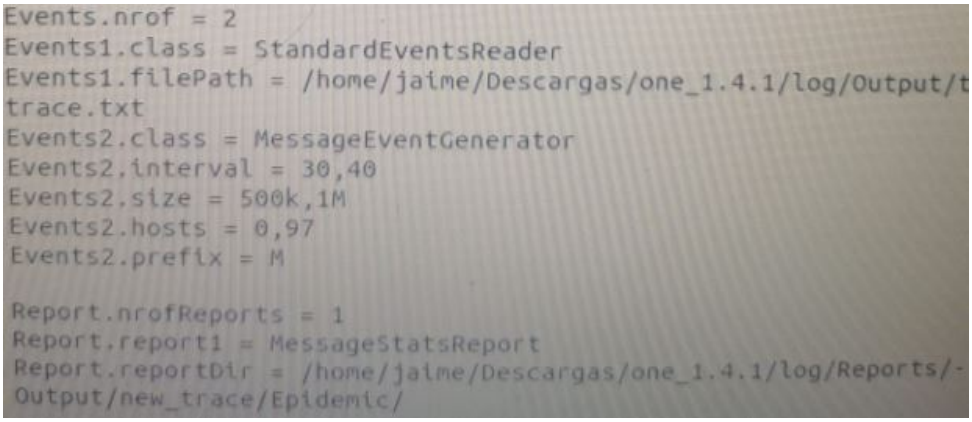

Fig. 7: Parte modificable Epidemic settings

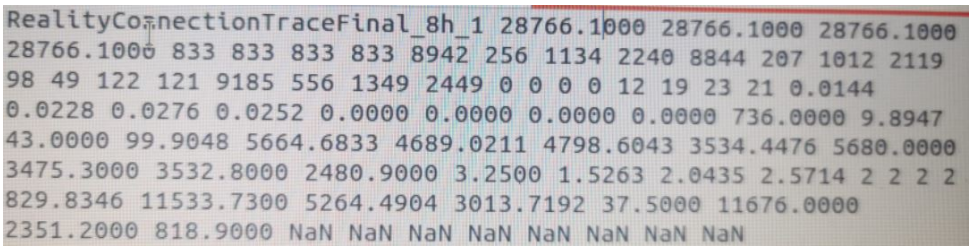

Fig. 8: Reporte general de una traza

D.4 Imágenes algoritmo genético

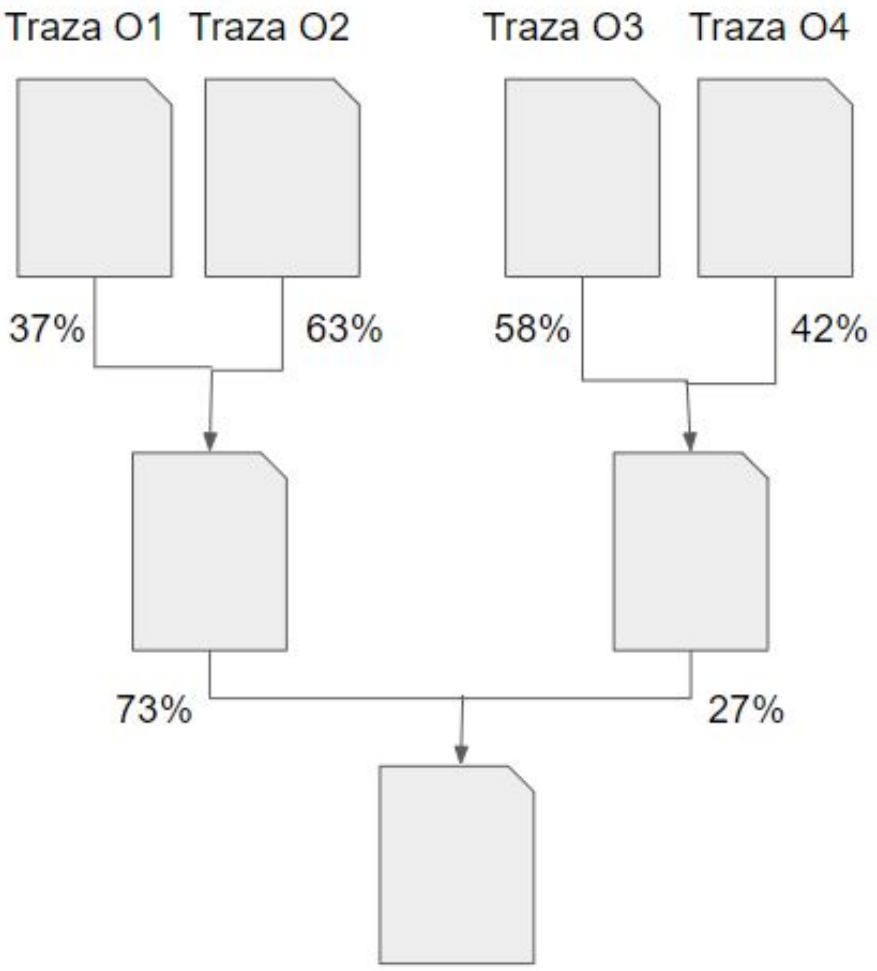

Fig. 9: Funcionamiento del algoritmo genético de cruce