Universidad ORT Uruguay

Facultad de Ingeniería

Sesionización de Logs Apache

Entregado como requisito para la obtención del título de
Licenciatura en Ingeniería de Software

Mauricio Pisabarro - 192095

Tutor: Ramiro Visca

2020

Declaraciones de autoría

Yo, Mauricio Pisabarro, declaro que el trabajo que se presenta en esta obra es de mi propia mano. Puedo asegurar que:

- La obra fue producida en su totalidad mientras realizaba el proyecto final de Licenciatura en Ingeniería de Software;
- Cuando he consultado el trabajo publicado por otros, lo he atribuido con claridad;
- Cuando he citado obras de otros, he indicado las fuentes. Con excepción de estas citas, la obra es enteramente mía;
- En la obra, he acusado recibo de las ayudas recibidas;
- Cuando la obra se basa en trabajo realizado conjuntamente con otros, he explicado claramente qué fue contribuido por otros, y qué fue contribuido por nosotros;
- Ninguna parte de este trabajo ha sido publicada previamente a su entrega, excepto donde se han realizado las aclaraciones correspondientes.

Mauricio Pisabarro
8 de agosto de 2020

Resumen

Este proyecto en la modalidad de trabajo de investigación se encuentra fuertemente vinculado a los proyectos de cátedra de Inteligencia Artificial y Big Data, en el marco de ANII, Fondo Sectorial de Investigación Basada en Datos y Fondo María Viñas. Estos tienen como objetivo la construcción de una herramienta anonimización de datos secuenciales para la detección de anomalías colectivas en ciberseguridad.

En el sentido de la detección de anomalías colectivas será de utilidad contar con sesiones de usuarios normales y de atacantes. En particular será necesario la reconstrucción de sesiones a partir de logs de auditoría de WAF (Web Application Firewall). Para ello se necesita una técnica de identificación y reconstrucción de sesiones.

Primero se hizo un relevamiento del estado del arte, seguido por un análisis de lo encontrado, discutiendo cual estudio de las soluciones se ajusta más a la problemática. Luego se desarrollaron objetivos sobre las mismas, y se hizo una experimentación sobre la cual se sacó conclusiones.

Palabras Clave

Sessionization; Sesionización; Time Heuristics; Referrer Heuristics; Time-Referrer Heuristics; Session; Web Session; Semantic Identification; Web Log Mining; Apache Logs Mining; Access Logs Mining;

Índice general

Capítulo 1

Introducción

El objetivo de este trabajo es la reconstrucción de sesiones de usuario a partir de logs de auditoría de un servidor apache que actúa de firewall. En particular esta idea se quiere aplicar a futuro a un conjunto de datos provenientes de AGESIC[1].

A su vez, es necesario definir el concepto de sesión con el que se tratará a lo largo de este trabajo. Si bien ninguno de los trabajos estudiados en la etapa de relevamiento coinciden exactamente en la definición de lo que es una sesión, todos comparten rasgos con la definición de sesión en el contexto de web analytics provista por Wikipedia[1]:

- *En Web Analytics, una* ***sesión*** *o visita es una unidad de medida sobre las acciones de un usuario tomadas dentro de un período de tiempo o con respecto a la finalización de una tarea.*

Es necesario poder reconstruir estas sesiones para luego poder detectar cuales de estas corresponden a amenazas o ataques a la ciberseguridad, y eventualmente publicar dichos datos. Este proceso de identificación de sesiones se lo conoce como **sesionización**.

[1] Agencia de Gobinerno y Sociedad de la Informacion y el Conocimiento

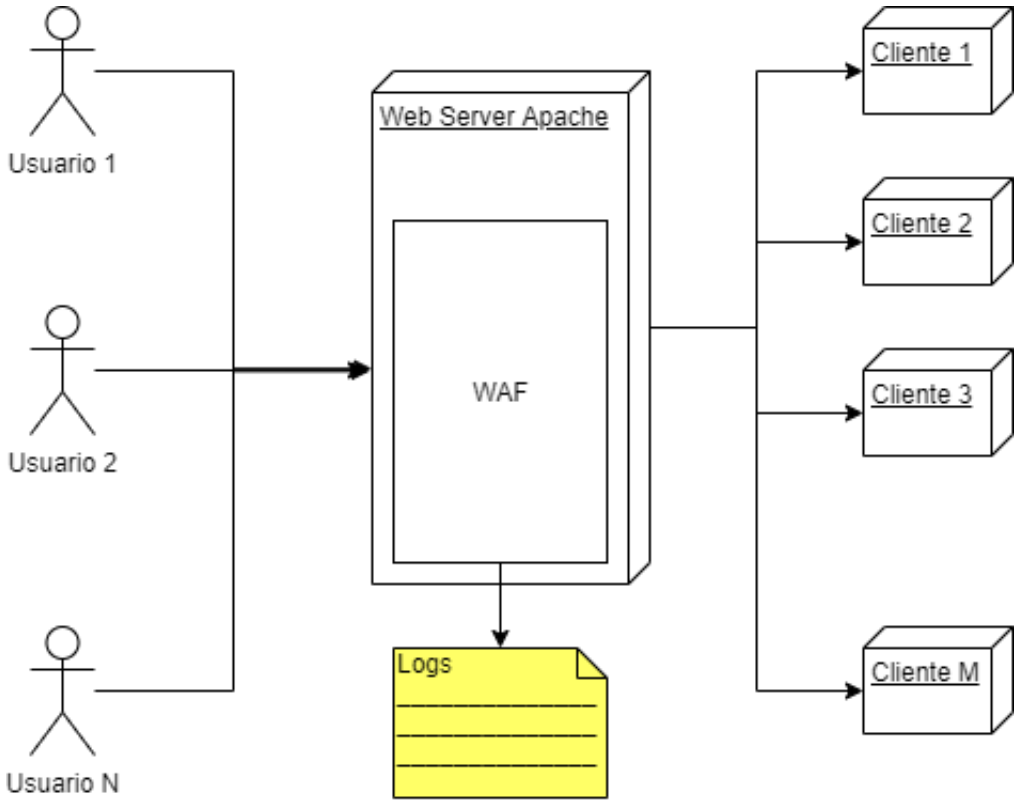


Figura 1.1: Sistema Generador de Logs

Como se puede ver en la figura anterior, se posee un WAF funcionando sobre un servidor Apache. El mismo tiene la capacidad de abrir y certificar paquetes HTTP o HTTPS, estudiarlos, y luego de verificar que el paquete no es una amenaza, redirigir el mismo al servidor web correspondiente.

Se espera poder detectar amenazas a la ciberseguridad estudiando las sesiones de usuarios implícitas en los logs del servidor WAF. Es por esta razón que se tiene la necesidad de investigar técnicas de sesionización capaces de reconstruir sesiones de usuario a partir de logs.

Con esto en mente, este trabajo se separa en otros 3 capítulos:

- **Relevamiento y Análisis del Estado del Arte**: se estudian ciertos trabajos de investigación en el área y se analiza que tan bien se ajustan a las necesidades del contexto previamente mencionado.
- **Solución Implementada**: se explica el enfoque tomado frente a la selección de Data Sets de prueba, y finalmente se habla de la implementación de algoritmos de sesionización.
- **Conclusiones**: se hace un breve resumen del trabajo y sus hallazgos, y luego se habla de otras posibles utilidades para los algoritmos desarrollados.

Capítulo 2

Relevamiento del Estado del Arte

Este capítulo hará un relevamiento del estado del arte en sesionización, y se le adjudicará una sección a cada trabajo estudiado. Con este dicho, primero se procede a explicar los conceptos previos comunes a todos los trabajos:

Las sesiones tal como se definen en el capítulo introductorio, existen en el contexto de web analytics. Web analytics, y más especificamente WUM (Web Usage Mining) es la motivación común a todas las publicaciones estudiadas, ya que todas buscan poder distinguir sesiones de usuario para luego poder extraer información significativa sobre los mismos y su comportamiento. Es importante entonces explicar el concepto de WUM, para lo cual se acude a la claridad y exhaustividad de la introducción del siguiente trabajo: *An Improved Session Identification Approach in Web Log Mining for Web Personalization*[2]

En el mismo, se define web mining como la transformación de técnicas de data mining para ser usadas con datos web. Luego distingue 3 fases dentro de WUM:

- **Contenido**, involucra extraer la información relevante del contenido web, por ejemplo dejando de lado imágenes.
- **Estructura** involucra identificar patrones a partir del resultado de la primera etapa.
- **Uso** es el análisis de los patrones descubiertos en la etapa anterior de minado. En esta etapa se busca entender la razón por la que se dan estos patrones.

Resumidamente WUM trata de identificar patrones de web browsing con la ayuda de información proveniente de web logs.

En particular, sesionización forma parte del pre-procesamiento de información para WUM (ver página siguiente):

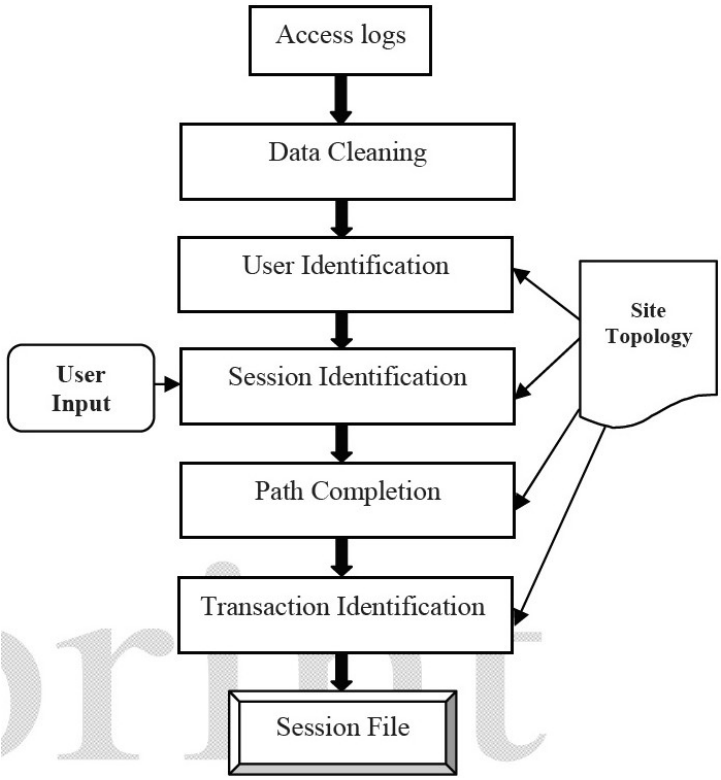


Figura 2.1: Proceso de WUM según: An Improved Session Identification Approach in Web Log Mining for Web Personalization[2]

- **Limpieza de la información**: proceso de eliminar información no relevante, como suelen ser imágenes, archivos css, html, entre otros.
- **Identificación de usuarios**: proceso de agrupación por usuarios. Usualmente se toma la decisión de agruparlos por IP, pero también pueden utilizarse técnicas más avanzadas de agrupamiento como K-means clustering considerando otros atributos de los logs.
- **Identificación de sesiones/sesionización**: proceso en el cual se toman los logs de cada usuario, y se agrupan en lo que se consideran sesiones. Puede considerarse o no la relación entre los mismos, como sería el caso del campo *referrer*, o la topología del sitio.
- **Finalización de camino tomado por el usuario**: proceso que determina el camino seguido por el usuario en el sitio web, a partir de los logs agrupados en la etapa anterior.
- **Identificación de la transacción**: proceso al que se someten los logs luego de la etapa anterior para determinar si los mismos, o un conjunto de ellos corresponden a una transacción. Por ejemplo: se agrupan aquellos logs que llevan a cabo a la realización de una compra online.

Luego de todos estos pasos es que se genera un archivo con información de sesiones para ser usado en la última etapa en el proceso de WUM.

2.1. Pre-procesamiento de datos

2.1.1. Identificación de usuarios

Esta etapa busca identificar cada usuario particular que accede a la web. Dicha etapa suele ser compleja dado que hay que tener en cuenta el cache local, y servicios como firewall o proxy. Tradicionalmente la identificación de usuario se basa simplemente tomando la IP asociada.

Un primer paso es identificar como usuarios individuales a todas aquellas direcciones IP distintas. Luego, para cada IP, identificar usuarios con diferentes sistemas operativos, o browsers. Y finalmente, para mismas IP, browsers, y sistemas operativos, identificar usuarios basándose en si la página pedida puede ser accedida de otras partes de la página [1]. Esto último es porque suelen existir porciones de una página que no pueden ser accedidas sin antes haber pasados por otras partes, y para saber esto es necesario contar información de la topología de la página.

2.1.2. Identificación de sesiones (Sessionization)

Esta etapa habla del proceso de segmentar la actividad de un usuario en sesiones[3], cada una representando una visita única al sitio. Al igual que la identificación de usuarios, esta etapa suele ser complicada dada la presencia de proxys y usuarios accediendo desde una misma máquina.

Sitios sin autenticación de usuario deberán confiar en el uso de heurísticas para sessionization. Estas ayudan a extraer la secuencia de acciones realizada por un usuario particular, en una visita particular al sitio.

La identificación de sesiones tradicional depende de timeouts uniformes y fijos. De lo contrario, una misma persona podría estar accediendo al mismo sitio pero con IP diferente (por haber hecho timeout), haciendo que el algoritmo lo identifique como una sesión completamente diferente. En otras palabras, si el tiempo entre una acción y otra es lo suficientemente grande, esto puede ser identificado como que ocurre en sesiones diferentes.

Es decir, las acciones que ocurran dentro de un lapso de tiempo fijo, se consideran parte de una misma sesión, de lo contrario, se tratan como sesiones diferentes. Experimentalmente se suele tomar este tiempo fijo como un intervalo de 10 minutos, con la contrapartida de que no se pueden identificar con exactitud sesiones más largas.

Para solucionar esto se pueden realizar estudios del tiempo que se toma una persona dentro de un sitio para cada una de las partes del mismo, para luego ajustar el intervalo de tiempo para cada una de las partes del sitio, y no tomar un sólo intervalo general. Si bien esto significa mejoras sobre la técnica anterior, según los

[1] Referrer based heuristic

autores de “An Improved Session Identification Approach in Web Log Mining for Web Personalization“[2], la exactitud aún no es lo suficientemente buena.

2.1.3. Finalización de camino tomado por el usuario

Luego de la identificación de sesiones, será necesario identificar camino tomado por el usuario buscando rearmar la secuencia de logs que ocurrió antes de un log particular, usualmente teniendo en cuenta la referencia de cada log, y a veces también considerando la topología de la red:

1. Para un pedido de página (log), se checkea si este se encuentra directamente relacionado con la página anterior, o no.

2. Si la página ya existe en el historial previo, entonces está claro que se utilizó el botón “back“.

3. Si muchas páginas están directamente relacionadas con la página pedida, entonces la más cercana (la que hizo el pedido) es la que se agrega a la sesión.

En caso de existir caches o proxys, las sesiones resultantes pueden contener “saltos“ entre una y otra por falta de información correspondiente a peticiones que fueron respondidas por los mismos, en lugar del servidor que generó los logs que se están estudiando.

2.2. Enfoques Tradicionales

En las etapas de pre-procesamiento de WUM se encuentran los dos enfoques tradicionales para identificación de sesiones basados en heurísticas de tiempo, y referencia, que se explicarán más adelante. Es importante definir heurística[4] en ciencias de computación, como una técnica para encontrar una solución a un problema más rápidamente en donde métodos clásicos son más lentos, o para encontrar una solución aproximada a problemas en los que métodos tradicionales fallan al encontrar una solución exacta.

La segunda definición es particularmente relevante para este trabajo, ya que **las sesiones encontradas no necesariamente reflejan sesiones reales exactas de usuarios**, y esto es debido a que los logs no contienen suficiente información como poder rearmar sesiones con exactitud.

2.3. Enfoques Tradicionales Basado en Heurísticas de Tiempo

2.3.1. Enfoque basado en largo de referencia

Este enfoque se basa en la idea de que el tiempo que un usuario se toma está directamente relacionado con si la página es principal o de contenido. Se espera que el

tiempo dedicado en la página principal sea menor que para las páginas de contenido.

Este tiempo de referencia se estima tomando la diferencia de tiempo entre el tiempo de una referencia y otra.

2.3.2. Enfoque de referencia hacia adelante máxima

Este enfoque considera como sesión a todos los logs ordenados cronológicamente que pertenezcan a una transacción. Una transacción se define para este enfoque como un conjunto ordenado de páginas que se termina cuando se visita una página ya visitada, es decir, cuando hay un ciclo.

2.3.3. Enfoque de ventana de tiempo

El contexto de una transacción dentro de una ventana de tiempo viene dado por (IP, id de usuario, largo límite de tiempo). Si la ventana es lo suficientemente grande, entonces cada transacción podría tener todas las referencias para cada usuario. Esto implica que se identificarían sesiones correctamente con baja probabilidad, ya que todos los usuarios tendrían una sola gran sesión.

2.4. Enfoque Tradicional Basado en la Heurísticas de Referencia/Navegación

Por otro lado, las metodologías basadas en *referencia* buscan tener en cuenta la topología del sitio estudiado para identificar sesiones. La idea detrás de estas metodologías es determinar si es posible visitar un conjunto de páginas dentro de una misma sesión. Dependiendo de la topología del sitio es posible determinar si una petición para una página **A** debería pertenecer a otra sesión, sabiendo que de la página actual **B** no se puede llegar directamente a **A**.

La topología del sitio se puede tener explícitamente en forma de grafo, o se puede ir descubriendo para cada sesión teniendo en cuenta sólo la referencia de los logs. Es decir, cuando la referencia de un log no fue antes vista, se considera como el comienzo de una nueva sesión, y al cerrar una sesión, se borra toda memoria de páginas visitadas.

2.5. An Improved Session Identification Approach in Web Log Mining for Web Personalization

Autores: P. Sengottuvelan, Lokeshkumar Ramasamy, Gopalakrishnan Thirumoorthy

2.5.1. Proceso de identificación de sesión mejorado

En este trabajo[2] se propone tomar los enfoques de largo de referencia y enfoque de ventana de tiempo, y hacerlos dinámicos con el fin de mejorar los enfoques basados en heurísticas de tiempo mencionadas en la sección 2.4: Enfoques Tradicionales Basado en Heurísticas de Tiempo. Esto significa que se trata de encontrar un largo de referencia y ventana de tiempo pero para cada una de las páginas, y no como un promedio genérico, o global a la página como proponen otros métodos.

Además se propone combinar estos métodos basados en heurísticas de tiempo con el método basado en la heurística de referencia previamente mencionado. Con esta adición se tiene en cuenta a la topología[2] del sitio para decidir si una petición pertenece o no a una misma sesión.

2.5.2. Conclusión

Los métodos mencionados no implican costos adicionales por sobre métodos más tradicionales[3], y en general su efectividad se encuentra 10 puntos porcentuales por arriba del promedio de efectividad conseguido por los métodos tradicionales. De todos los trabajos estudiados, este parece ser el más intuitivo.

Dada esta simpleza y efectividad es que se decide seguir adelante con este trabajo hacia la etapa de experimentación, y así poder seleccionar él o los métodos que mejor se adecuen a nuestra problemática.

[2]Información que se encuentra presente en los logs HTTP en forma del campo referrer".
[3]Como se definen en la introducción al capítulo 1.

2.6. User Session Identification Based on Strong Regularities in Inter-activity Time

Autores: Aaron Halfaker, Oliver Keyes, Daniel Kluver, Jacob Thebault-Spieker, Tien Nguyen, Kenneth Shores, Anuradha Uduwage, Morten Warncke-Wang

Este trabajo[5] explora la identificación de sesiones basándose en los tiempos que ocurren entre una acción y otra para un mismo usuario [4], dentro de una misma página. La razón por la que proponen esto deriva de un estudio realizado por los mismos en donde analizan logs de servidores web de distinto tipo, incluso enfocados a diferentes públicos. En este estudio se puede observar como acciones para un mismo usuario tienden a estar acumuladas en "períodos de actividad", y suelen estar separados por "períodos de inactividad" con diferentes grados de definición dependiendo del servidor.

Para reafirmar que existe una regularidad entre los tiempos de interacción los autores utilizan un Gaussian Mixture Model[5]. Con los tiempos ya identificados ellos identifican con mayor probabilidad sesiones utilizando métodos de sesionización basados en heurísticas de tiempo ya existentes.

2.6.1. Conclusión

Resumidamente este trabajo refuerza algunas ideas propuestas en métodos basados en heurísticas de tiempo. Según los autores utilizar este tipo de heurísticas no sólo es posible, si no que podrían brindarnos gran efectividad dada la regularidad aparente con la que ocurren estos períodos de actividad e inactividad.

Si bien no se habla de un método en particular, se valida la utilización de heurísticas de tiempo con las que se experimentará más adelante, y proveen un método para encontrar los intervalos de tiempo entre una sesión y otra. Además, si bien los autores identifican usuarios simplemente por su IP[6], se podría mejorar en este aspecto tomando métodos utilizados por otros trabajos como por ejemplo agrupando además por "User Agent", o características de la máquina que hace consultas.

[4] El trabajo no especifica los detalles de como identifica a estos usuarios más allá de considerar cada IP como usuario individual. Por esto se asume que este es el único criterio considerado.

[5] Un método de aprendizaje automático

[6] Lo cual no es del todo correcto porque muchos usuarios podrían tener la misma IP.

2.7. Reconstructing Sessions from Data Discovery and Access Logs to Build a Semantic Knowledge Base for Improving Data Discovery

Autores: Yongyao Jiang, Yun Li, Chaowei Yang, Edward M. Armstrong, Thomas Huang, David Moroni

En este trabajo[6] los autores proponen identificar sesiones basándose en una combinación de las heurísticas basadas en tiempo y referencia. Primero proponen identificar usuarios basándose en la dirección de IP de los logs, sin tener en cuenta el dato de User Agent de manera de poder acomodar logs web, y FTP[7]. Esto es debido a que los logs FTP no tienen esta información.

Luego, al igual que en otros trabajos estudiados previamente, se hace una limpieza de los logs, quitando todos aquellos que hacen peticiones no relevantes como son los archivos:

- favicon.ico
- robots.txt
- CSS
- Javascript
- Media (imágenes, video, etc)

La siguiente etapa involucra agrupar posibles sesiones utilizando las siguientes heurísticas de tiempo:

- La mencionada en *User Session Identification Based on Strong Regularities in Inter-activity Time*[5][8]
- Agrupando intervalos de tiempo utilizando el método de agrupación jerárquica[7] propuesto en este trabajo.
- Una combinación de la primera técnica mencionada, con el agregado de la heurística basada en referencia mencionada en *An Improved Session Identification Approachin Web Log Mining for Web Personalization*[2].

Se procede a explicar las últimas dos tecnicas:

[7]Esto es un punto sobre el cual se puede mejorar, ya que la información disponible para la cátedra contiene al campo del User Agent, lo cual brindaría mayor granularidad a la hora de identificar usuarios porque permite diferenciar peticiones de usuarios distintos con una misma IP.

[8]Se busca con un Gaussian Mixture Model los tiempos de interactividad, y con ello se agrupan logs que difieran en una cantidad de tiempo igual o menor a la encontrada.

En la técnica de agrupación jerárquica toma los logs ordenados por fecha, y los va agrupando recursivamente en diferentes grupos. El criterio de agrupación depende de intervalo de tiempo T que se determina con el método de optimización de "natural breaks"de Jenks[8], e indica si una agrupación debe subdividirse si logs consecutivos superan este intervalo T. El resultado final de esta técnica de agrupación son las sesiones buscadas.

Por otro lado, la combinación de las heurísticas de tiempo y referencia consiste en un proceso de dos etapas. Primero se toma un enfoque basado en heurísticas de tiempo, y al resultado de esto se lo subdivide o agrupan con alguna heurística de referencia.

Si bien este trabajo no hace menciones de la precisión, exactitud, ni eficiencia de los métodos propuestos, se cree que es uno de los trabajos estudiados más completos ya que combina varias metodologías. Es un trabajo que vale la pena analizar.

2.7.1. Conclusión

Este trabajo resumidamente combina heurísticas de tiempo y referencia ya existentes. El mismo compara y contrasta dos técnicas propuestas basadas en heurísticas de tiempo, una utilizando un Gaussian Mixture Model, y la otra utilizando agrupación jerárquica.

Entre estas dos técnicas no parece haber grandes diferencias en cuanto a resultados, pero lo que si resulta interesante es la combinación de las mismas con alguna técnica basada en la heurística de referencia. Se considera intuitivo considerar los tiempos de interacción entre una acción y otra en una sesión, y la topología de la página.

2.8. Semantic Identification of Web Broswing Sessions

Autor: Neel Guha.

Este trabajo[9] busca poder identificar sesiones en casos que los métodos tradicionales de identificación de sesiones (basados en tiempo, y/o basados en referencia). En particular el paper trata de identificar sesiones sin el presupuesto de que hay un sólo usuario por cada dispositivo. Este tipo de situaciones suele darse en países poco desarrollados, o paises como China, en donde la presencia de cyber cafes es fuerte, y se comparte el mismo dispositivo entre muchos usuarios. Esta situación también suele darse cuando existe la presencia de un proxy, o un VPN, que representan a muchos usuarios.

2.8.1. Identificación Semántica

Para lograr una identificación de una sesión sin información de la proveniencia de los logs, el paper propone entender el contenido semántico de los mismos. Es decir, se busca relacionar logs que llevan a lograr un fin particular (como puede ser aprender de programación), de manera que se pueda entender el comportamiento del usuario. Entendiendo el comportamiento del mismo es que podremos identificarlo.

2.8.2. Supuestos

Unicidad

Se supone que el usuario a identificar tiene una huella característica a el mismo. De lo contrario, si todos los usuarios se comportaran de la misma manera, sería imposible identificar a alguno basado tan sólo en comportamiento, todos se verían iguales.

Consistencia

Se asume que logs de una sesión de un mismo usuario son aproximadamente consistentes entre si. Es prácticamente imposible que dos sesiones del mismo usuario se vean exactamente igual, pero se espera que tengan un alto grado de similitud en promedio.

2.8.3. Información relevante a la semántica (señales)

- Las palabras contenidas en una página visitada.
- Las clasificaciones categóricas de una página (noticias, red social, comida, etc).
- Los títulos de las páginas visitadas.
- Las URL de las páginas visitadas

- Los links dentro de las páginas visitadas

Se considera algunas de estas "señales", como dice el autor, más valiosas que otras.

2.8.4. Métodos de Sessionization

Se mencionan dos métodos de identificación semántica:

1. Cercanía ponderada
2. Aprendizaje supervisado

La cercanía ponderada busca determinar que tan similar (y disimilar) es una página de otras. En base a este puntaje de similitud y disimilitud es que se decide si una petición pertenece a una sesión particular o no.

Este puntaje se calcula utilizando el método de similitud de coseno[9].

Por otro lado, el método de aprendizaje supervisado se basa en los mismos principios, con la diferencia de que el puntaje de similitud es calculado por un agente inteligente. Este agente está modelado por una red neuronal conocida como perceptrón multicapa[10] capaz de aprender mecanismos para determinar este puntaje.

2.8.5. Conclusión

Si bien los métodos mencionados en este trabajo son una mejora sobre un enfoque base[10], no se considera una mejora suficiente. Con el método más efectivo siendo el de cercanía ponderada de los componentes semánticos de los logs (tags, texto, e información sobre el dominio de la página) con alrededor de 50 % de efectividad, este no se compara con métodos basados en heurísticas de tiempo o referencia que en promedio suelen superar el 80 % de efectividad.

Por esta razón es que los métodos mencionados en este trabajo no se consideraron para la etapa de experimentación. Quizás los mismos sean explorados en conjunto con los métodos basados en heurísticas de tiempo o referencia, pero ello queda por fuera del acance de este trabajo.

[9] Cosine similarity

[10] Según los autores, un enfoque base es aquel donde se toma a un agente que adivina aleatoriamente las sesiones en un conjunto de web logs.

2.9. Otros Trabajos Estudiados

Se estudian otros trabajos, pero deciden dejarse de lado debido a su gran complejidad de implementación y/o por repetición de ideas de otros trabajos previos. Ellos son:

- Identifying User Sessions from Web Server Logs With Integer Programming[11]
- Identifying the User Access Pattern in Web Log Data[12]
- Predicting user behavior through Sessions using the Web log mining[13]

Capítulo 3

Solución Implementada

3.1. Algoritmos Implementados

Luego del análisis del estado del arte se procede a implementar dos algoritmos de identificación de sesiones, uno basado exclusivamente en heurísticas de tiempo (que no fue probado con datos reales ya que identifica sesiones con baja probabilidad), y otro que combina heurísticas de tiempo y referencia.

Si bien en los trabajos estudiados se hacen descripciones de alto nivel, es importante aclarar que no se logró encontrar implementaciones a nivel de código ni pseudocódigo para los algoritmos. Por esta razón es que se decide implementar los algoritmos desde cero incluyendo la ideación del concepto de sesión como estructura de datos.

3.1.1. Paso en Común de Pre-Procesamiento de Datos: Identificación de Usuarios

Los algoritmos implementados tienen un paso base común en donde se agrupan las sesiones por IP y User Agent. Se considera que el par <IP, User Agent>permite diferenciar la gran mayoría de los usuario presentes en los logs con la única excepción siendo cuando dos usuarios distintos usan el mismo dispositivo bajo la misma red (e IP) como suele ser el caso de los cyber-café. En este caso dos usuarios distintos tendrían el mismo par <IP, User Agent>.

De todos los trabajos investigados sólo *Semantic Identification of Web Broswing Sessions*[9] busca resolver este problema utilizando la semántica de las páginas contenidas en los logs para diferenciar usuarios y sesiones. Sin embargo, no se considera este trabajo para la implementación de los algoritmos.

3.1.2. Algoritmo Basado en Heurística de Tiempo

Como se menciona en el Capítulo 2 del Relevamiento del Estado del Arte, los algoritmos basados en heurísticas de tiempo son aquellos que buscan identificar los logs que pertenezcan a una misma ventana de tiempo, y luego el momento de "corte" en donde comienza una nueva ventana en caso de existir. Es decir, únicamente se

IP	Date	Request	Status	Referrer	User Agent
31.56.96.51	[23/01/2019:02:56:16]	GET /G	200	/K	Chrome
31.56.96.51	[22/01/2019:23:00:43]	GET /Z	200	/A	Chrome
31.56.96.51	[22/01/2019:02:56:16]	GET /G	500	/A	Chrome
31.56.96.51	[24/01/2019:02:56:16]	GET /K	200	/Z	Chrome
31.56.96.51	[23/01/2019:02:56:16]	GET /C	400	/M	Chrome
31.56.96.51	[25/01/2019:02:56:16]	GET /M	300	/C	Chrome

Tabla 3.1: Tabla de logs de una sola IP y User Agent

considera el timestamp de un log para medir su diferencia de tiempo con el inmediatamente anterior, y en base a esto decidir a que ventana de tiempo (sesión) pertenece.

A la hora de implementar un algoritmo basado en heurísticas de tiempo se siguen los siguientes pasos luego de la agrupación de logs por <IP, User Agent>:

- **Se ordenan los logs en órden cronológico** según su timestamp.
- **Se define un delta de tiempo** que actuará como diferencia máxima entre los timestamps de los logs de una misma sesión o ventana de tiempo. [1]
- **Se recorren los logs en órden y se asignan a sesiones**. Se comienza con una sesión[2] vacía y se van agregando logs siempre y cuando el log que está siendo considerado se encuentre dentro del delta de tiempo respecto del último log en la sesión. De lo contrario se da por terminada la sesión anterior, y se inserta el log previamente mencionado en una nueva sesión.

3.1.3. Algoritmo Basado en Heurísticas de Tiempo y Referencia

El algoritmo considera los logs sin algún orden particular e identifica como sesión al conjunto de logs que pertenecen al mismo grafo dirigido débilmente conexo[3]. Esto significa que las sesiones identificadas se pueden visualizar como grafos "de una sola pieza", en donde los vértices son las URI de los logs, y las aristas son las relaciones formadas por las referencias presentes en los logs.

En la página siguiente se puede ver un ejemplo de logs de una misma IP y User Agent agrupados en sesiones según una heurística de referencia:

[1] Se toma este delta de tiempo como 10 minutos, acorde con lo que los trabajos estudiados toman como base.

[2] Para la implementación del algoritmo basado en heurística de tiempo se considera una sesión como una lista de logs ordenada.

[3] Un grafo no dirigido es conexo si existe un camino entre cada par de vértices. Si se considera el grafo subyacente (sin sentido en los arcos, i.e: como no dirigido), y este es conexo, entonces el grafo es débilmente conexo.

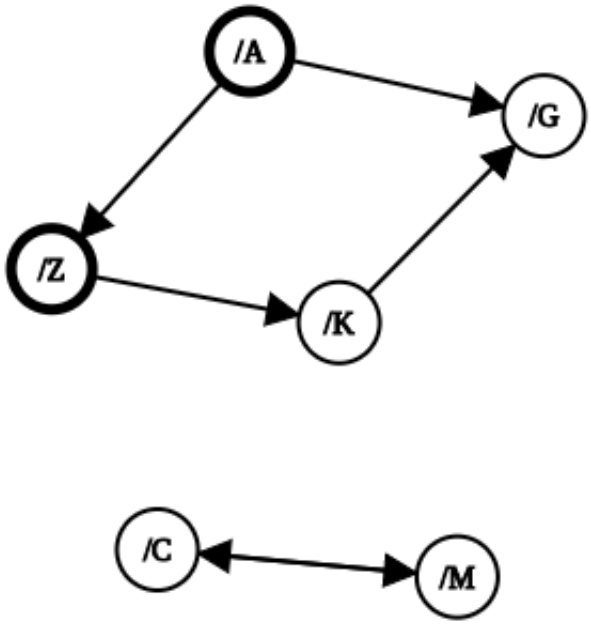


Figura 3.1: Sesiones resultantes de la tabla de logs.

Conociendo ambos algoritmos se puede proceder a explicar la heurística que combina a las previamente mencionadas.

- **Se considera el conjunto de logs ordenados cronológicamente** y se los recorre en ese órden.
- **Se define un delta de tiempo** que actuará como diferencia máxima entre los timestamps de los logs de una misma sesión o ventana de tiempo.
- **Las sesiones se consideran como grafos dirigidos** al igual que en el caso de la heurística de referencia, pero con una diferencia clave: todos los nodos tienen un "super padre" análogo a la raíz de un árbol, y corresponde al nodo más joven cronológicamente en la estructura de la sesión.
- Al encontrar un nuevo log se considera la referencia del mismo, y si la misma es desconocida o su última visita tiene una antigüedad mayor a la ventana de tiempo, entonces se considera como el comienzo de una nueva sesión. De lo contrario el log actual pasa a formar parte de los nodos hijo de su referencia.

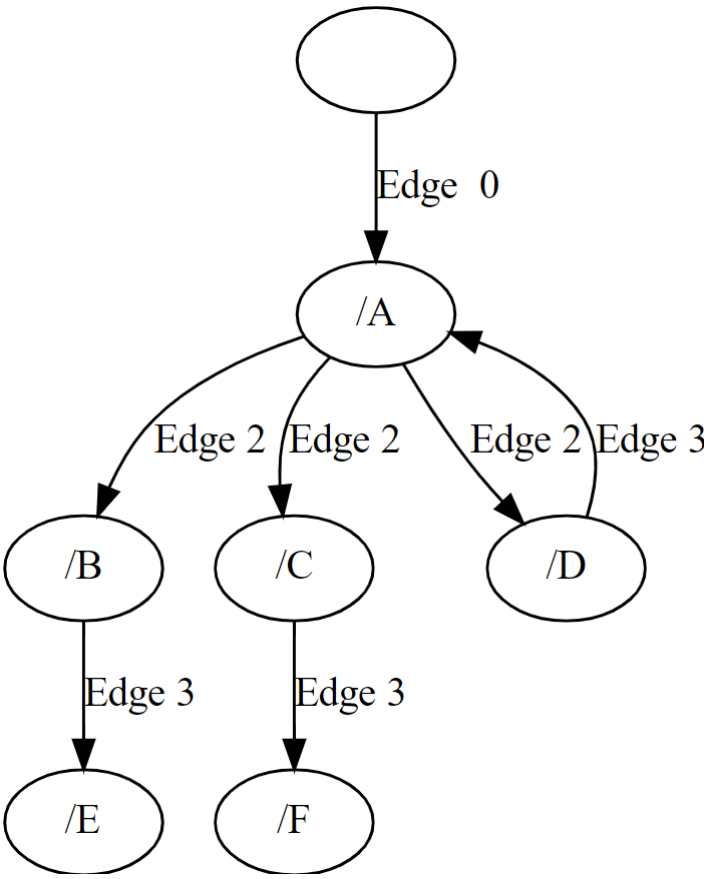


Figura 3.2: Visualización de una sesión del algoritmo basado en tiempo y referencia.

3.1.4. Estructuras de datos relevantes

Se identifican 2 estructuras de datos más relevantes:

- **Sesión**: tal como se menciona anteriormente, se consideran las sesiones como grafos dirigidos donde cada nodo se identifica por su URI y timestamp asociados. Todos los nodos poseen un nodo que se encuentra por sobre todos en la jerarquía (como la raíz de un árbol), que es el comienzo cronológico de la sesión identificada. Además estos nodos pueden a su vez tener nodos hijo.
- **Map de URIs a nodos de una sesión**: es una estructura que dada una URI retorna una referencia al último nodo (cronológicamente) con dicha URI. Gracias a esta estructura se tiene rápido acceso por medio de hashing al camino previo correspondiente a la referencia del log que se está considerando, lo cual permite concatenarlo con el nuevo nodo generado. Es importante mencionar que esta estructura no considera el timestamp de los logs, y siempre se queda con la referencia al último nodo correspondiente con la URI.

3.2. Casos Base del Algoritmo

3.2.1. Casos Base Identificados

Al no tener las sesiones de usuario previamente identificadas se tuvo que crear varios DataSets de ejemplo que actúan de casos base para la identificación de sesiones. Se espera que los mismos son lo suficientemente exhaustivos como para que cualquier sesión identificada por nuestro algoritmo esté compuesta por alguna combinación de estos casos.

Existe una relación directa entre los casos base y los criterios que el algoritmo utiliza para identificar sesiones. Para cada uno de los casos se hará una explicación al respecto.[4]

Sesiones Secuenciales con Partición por Tiempo

Este caso busca identificar la partición de sesiones existente más simple, en donde se identifica una separación entre una sesión y otra causada por una diferencia de tiempo entre dos logs de más de 10 minutos.

IP	Time	Request	Referrer
192.168.1.1	22-01-19 03:56	GET /A	-
192.168.1.1	22-01-19 03:56	GET /B	https://www.zanbil.com/A
192.168.1.1	22-01-19 03:56	GET /C	https://www.zanbil.com/B
192.168.1.1	22-01-19 04:00	GET /D	https://www.zanbil.com/C
192.168.1.1	22-01-19 04:03	GET /E	https://www.zanbil.com/D
192.168.1.1	22-01-19 04:15	GET /F	https://www.zanbil.com/E
192.168.1.1	22-01-19 04:15	GET /G	https://www.zanbil.com/F
192.168.1.1	22-01-19 04:20	GET /H	https://www.zanbil.com/G

[4]Para simplificar se dejarán de lado algunos atributos del log: User Agent, status, size.

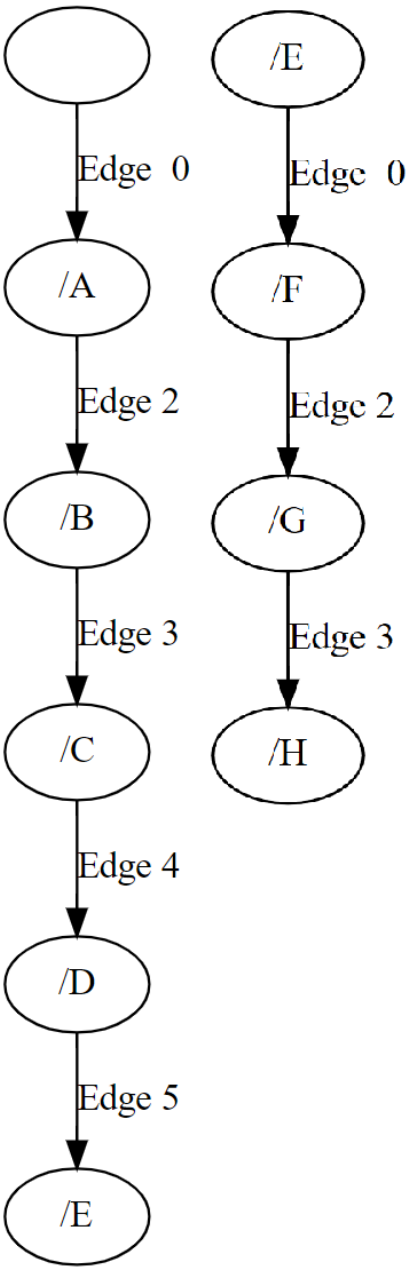


Figura 3.3: Sesiones Secuenciales con Partici on por Tiempo

Sesiones Secuenciales con Partición por Referencia

Este caso base considera la partición dos sesiones basado en la no existencia o no visita previa del referrer.

IP	Time	Request	Referrer
192.168.1.1	2019-01-22 03:56:17	GET /A	-
192.168.1.1	2019-01-22 03:56:17	GET /B	https://www.zanbil.com/A
192.168.1.1	2019-01-22 03:56:17	GET /C	https://www.zanbil.com/B
192.168.1.1	2019-01-22 04:00:17	GET /D	https://www.zanbil.com/C
192.168.1.1	2019-01-22 04:03:47	GET /E	-
192.168.1.1	2019-01-22 04:05:00	GET /F	https://www.zanbil.com/E
192.168.1.1	2019-01-22 04:05:20	GET /G	https://www.zanbil.com/F
192.168.1.1	2019-01-22 04:08:20	GET /H	https://www.zanbil.com/G

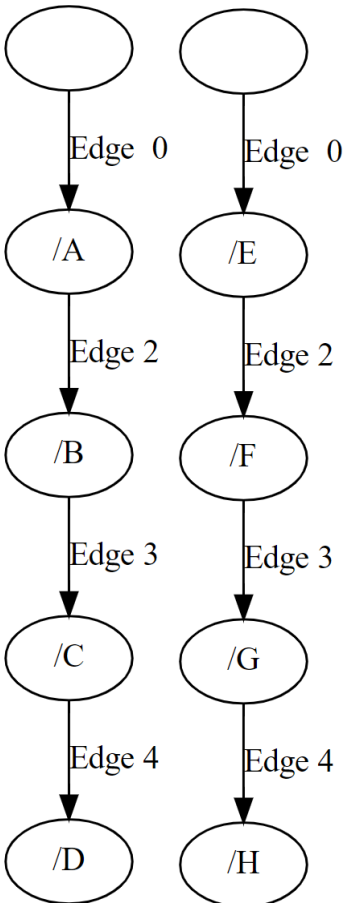


Figura 3.4: Sesiones Secuenciales con Particion por Referencia

Sesiones Paralelas

Este caso considera la existencia de sesiones paralelas con orígenes distintos y que no tienen conflictos (no se cruzan).

IP	Time	Request	Referrer
192.168.1.1	2019-01-22 03:56:17	GET /A	-
192.168.1.1	2019-01-22 03:56:17	GET /B	-
192.168.1.1	2019-01-22 03:56:17	GET /C	-
192.168.1.1	2019-01-22 04:00:17	GET /D	https://www.zanbil.com/A
192.168.1.1	2019-01-22 04:03:47	GET /E	https://www.zanbil.com/D
192.168.1.1	2019-01-22 04:05:00	GET /F	https://www.zanbil.com/B
192.168.1.1	2019-01-22 04:05:20	GET /A	https://www.zanbil.com/C

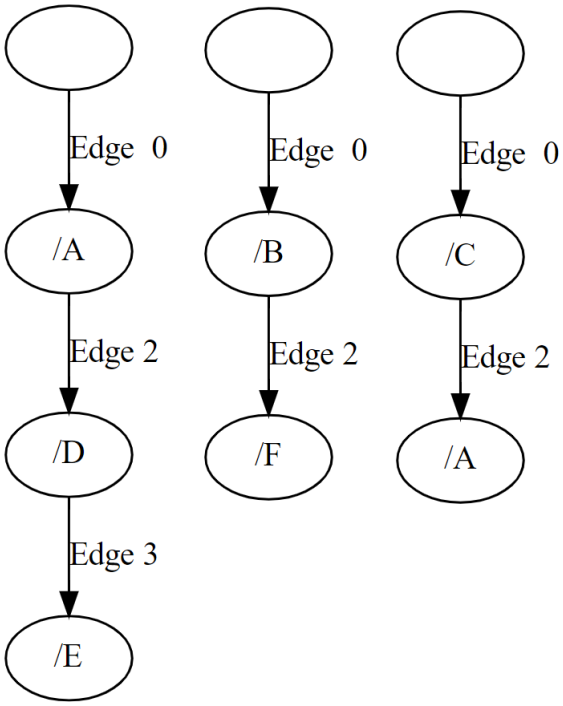


Figura 3.5: Sesiones Paralelas

Sesiones Paralelas con Partición por Tiempo

Este caso considera la existencia de sesiones paralelas con orígenes distintos, que no tienen conflictos (no se cruzan), y a su vez una de las mismas se separa en otra sesión por poseer una referencia que fue visitada por fuera de la ventana de tiempo límite (10 minutos): **G**.

IP	Time	Request	Referrer
192.168.1.1	2019-01-22 03:56:17	GET /A	-
192.168.1.1	2019-01-22 03:56:17	GET /B	-
192.168.1.1	2019-01-22 03:56:17	GET /C	-
192.168.1.1	2019-01-22 04:00:17	GET /D	https://www.zanbil.com/A
192.168.1.1	2019-01-22 04:03:47	GET /E	https://www.zanbil.com/D
192.168.1.1	2019-01-22 04:05:00	GET /F	https://www.zanbil.com/B
192.168.1.1	2019-01-22 04:07:02	GET /G	https://www.zanbil.com/F
192.168.1.1	2019-01-22 04:18:02	GET /H	https://www.zanbil.com/G
192.168.1.1	2019-01-22 04:05:20	GET /A	https://www.zanbil.com/C

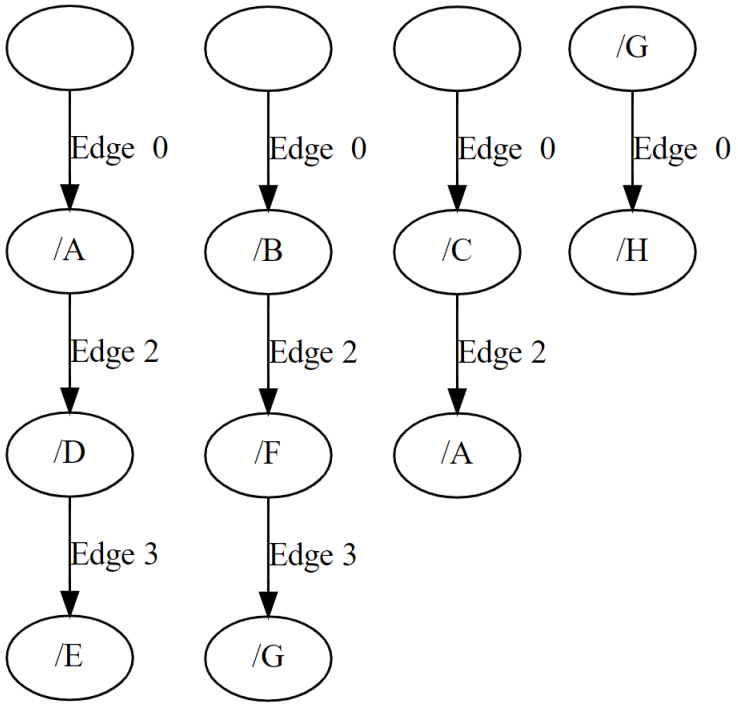


Figura 3.6: Sesiones Paralelas con Partición por Referencia

Sesiones Paralelas con Partición por Referencia

Este caso considera la existencia de sesiones paralelas con orígenes distintos, que no tienen conflictos (no se cruzan), y a su vez una de las mismas se separa en otra sesión por poseer una referencia nunca antes visitada: **G**.[5]

IP	Time	Request	Referrer
192.168.1.1	2019-01-22 03:56:17	GET /A	-
192.168.1.1	2019-01-22 03:56:17	GET /B	-
192.168.1.1	2019-01-22 03:56:17	GET /C	-
192.168.1.1	2019-01-22 04:00:17	GET /D	https://www.zanbil.com/A
192.168.1.1	2019-01-22 04:03:47	GET /E	https://www.zanbil.com/D
192.168.1.1	2019-01-22 04:05:00	GET /F	https://www.zanbil.com/B
192.168.1.1	2019-01-22 04:07:02	GET /G	-
192.168.1.1	2019-01-22 04:12:02	GET /H	https://www.zanbil.com/G
192.168.1.1	2019-01-22 04:05:20	GET /A	https://www.zanbil.com/C

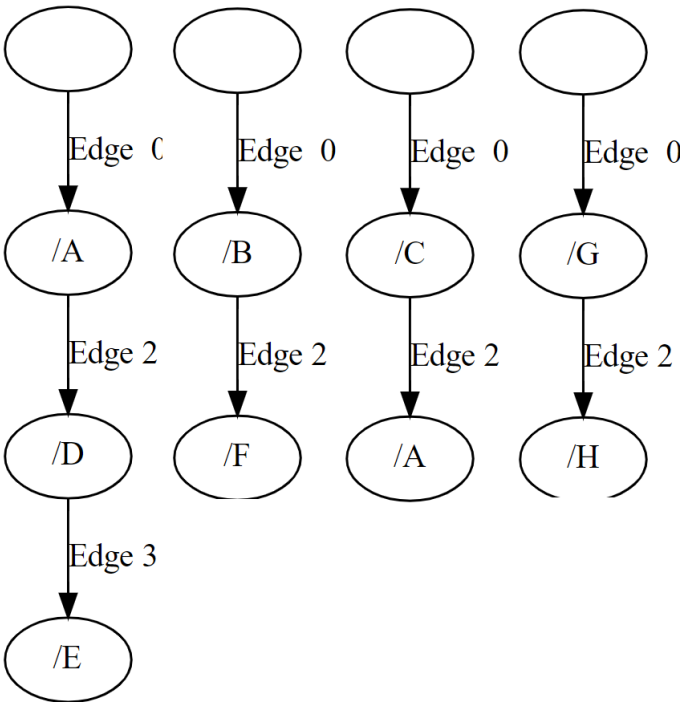


Figura 3.7: Sesiones Paralelas con Partición por Referencia

[5]Puede ser considerada como un caso particular de sesiones paralelas

Sesiones Paralelas con Partición por Tiempo y Referencia

Este caso considera la existencia de sesiones paralelas con orígenes distintos, que no tienen conflictos (no se cruzan), y a su vez una de las mismas se separa en otra sesión por poseer una referencia nunca antes visitada: **google.com**, y otras por tener una referencia que fue visitada por fuera de la ventana de tiempo límite: **K, J, I**.[6]

IP	Time	Request	Referrer
192.168.1.1	2019-01-22 03:56:17	GET /A	-
192.168.1.1	2019-01-22 03:56:17	GET /B	-
192.168.1.1	2019-01-22 03:56:17	GET /C	-
192.168.1.1	2019-01-22 04:00:17	GET /D	https://www.zanbil.com/A
192.168.1.1	2019-01-22 04:03:47	GET /E	https://www.zanbil.com/D
192.168.1.1	2019-01-22 04:05:00	GET /F	https://www.zanbil.com/B
192.168.1.1	2019-01-22 04:07:02	GET /G	https://google.com?=qasdfasdf
192.168.1.1	2019-01-22 04:12:02	GET /H	https://www.zanbil.com/G
192.168.1.1	2019-01-22 04:05:20	GET /A	https://www.zanbil.com/I
192.168.1.1	2019-01-22 04:05:40	GET /I	https://www.zanbil.com/J
192.168.1.1	2019-01-22 04:15:45	GET /J	https://www.zanbil.com/K

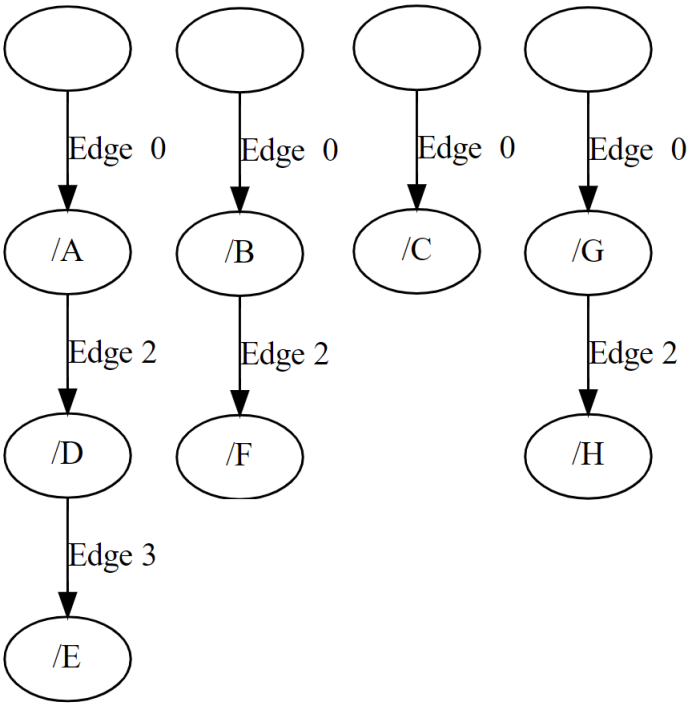


Figura 3.8: Sesiones Paralelas con Partición por Tiempo y Referencia 1/2

[6]Puede ser considerada como una combinación de los dos casos anteriores.

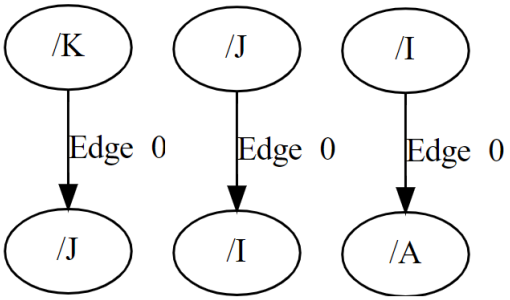


Figura 3.9: Sesiones Paralelas con Partición por Tiempo y Referencia 2/2

Sesiones Paralelas con Mismo Orígen

Este caso considera la existencia de sesiones con estructura reminiscente de un árbol (aunque más correctamente definida por un grafo débilmente conexo), donde todos los nodos se encuentran bajo una raiz y cada uno puede tener uno más hijos.

IP	Time	Request	Referrer
192.168.1.1	2019-01-22 03:56:17	GET /A	-
192.168.1.1	2019-01-22 03:56:20	GET /B	https://www.zanbil.com/A
192.168.1.1	2019-01-22 03:56:20	GET /C	https://www.zanbil.com/A
192.168.1.1	2019-01-22 03:56:20	GET /D	https://www.zanbil.com/A
192.168.1.1	2019-01-22 04:03:47	GET /E	https://www.zanbil.com/B
192.168.1.1	2019-01-22 04:05:00	GET /F	https://www.zanbil.com/C
192.168.1.1	2019-01-22 04:05:20	GET /A	https://www.zanbil.com/D

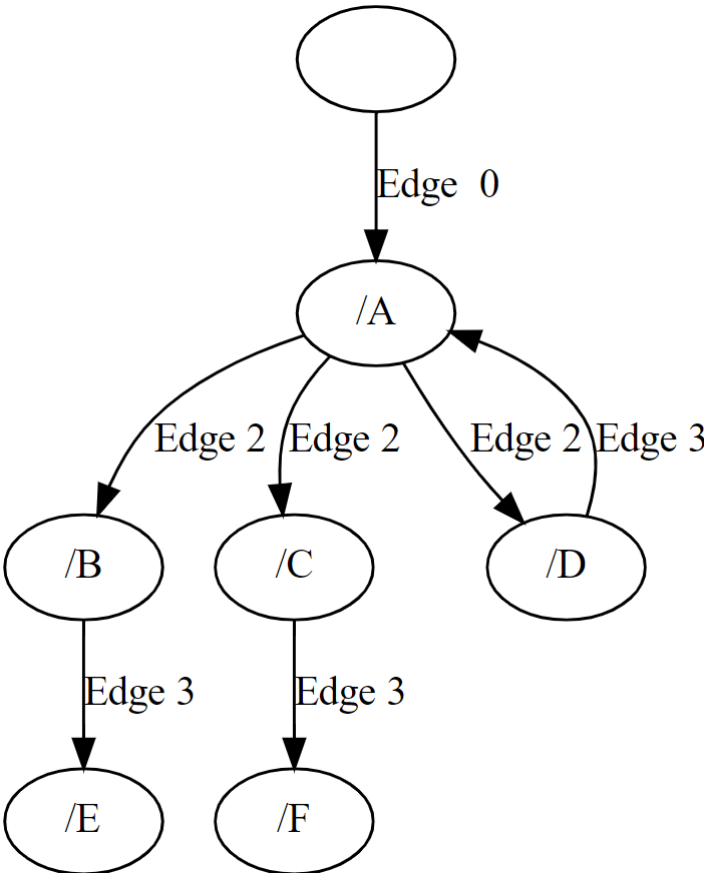


Figura 3.10: Sesiones Paralelas con Mismo Orígen

Sesiones Paralelas con Partición por Tiempo

Este caso considera la existencia de sesiones con estructura reminiscente de un árbol (aunque más correctamente definida por un grafo débilmente conexo), donde todos los nodos se encuentran bajo una raiz y cada uno puede tener uno más hijos. A su vez, una de sus ramas se ve particionada debido a que **C** tiene como referencia a **A**, y esta fue visitada por fuera de la ventana de tiempo limite.

IP	Time	Request	Referrer
192.168.1.1	2019-01-22 03:56:17	GET /A	-
192.168.1.1	2019-01-22 03:56:20	GET /B	https://www.zanbil.com/A
192.168.1.1	2019-01-22 04:56:20	GET /C	https://www.zanbil.com/A
192.168.1.1	2019-01-22 03:56:20	GET /D	https://www.zanbil.com/A
192.168.1.1	2019-01-22 04:03:47	GET /E	https://www.zanbil.com/B
192.168.1.1	2019-01-22 05:00:00	GET /F	https://www.zanbil.com/C
192.168.1.1	2019-01-22 04:05:20	GET /A	https://www.zanbil.com/D

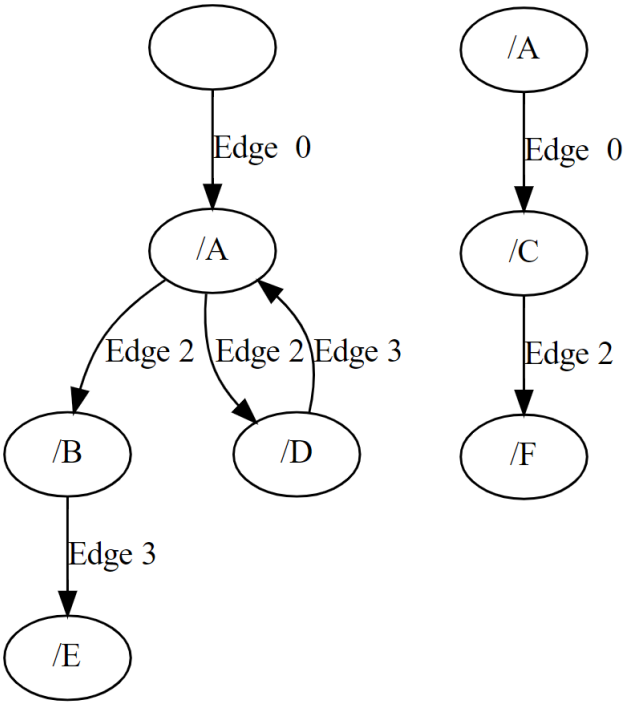


Figura 3.11: Sesiones Paralelas con Partición por Tiempo

3.2.2. Casos no cubiertos por implementación

Se considera a la implementación como la versión más robusta de los algoritmos de sesionización implementados, i.e.: basado en heurísticas de tiempo y referencia. Dicho esto, existen un par de casos que la implementación no logra distinguir:

- **Usuarios distintos usando el mismo dispositivo**: si bien es posible determinar las sesiones de un dispositivo particular bajo una misma IP, la implementación no logra asignar sesiones a un usuario particular. Ejemplo: un cibercafé.
- **Cuando se abren muchas instancias de una misma página dentro de la ventana de tiempo límite y luego se visitan otras páginas**: la implementación no logra distinguir el camino recorrido por la sesión, ya que la referencia siempre se adjudica a la última instancia abierta de la página mencionada. Por ejemplo: cuando se abren muchas pestañas con la misma página, y luego se sigue navegando a lugares distintos en cada una de las pestañas abiertas.
- **Cuando existe la presencia de proxys o caches:** estos servidores intermedios son capaces de responder a ciertas peticiones en lugar del servidor que genera los logs que se están estudiando. Esto significa que al armar sesiones podrían experimentarse saltos en las mismas (huecos de información), correspondientes a las peticiones que fueron atendidas por los proxy/caches y no fueron logeadas por el servidor estudiado.

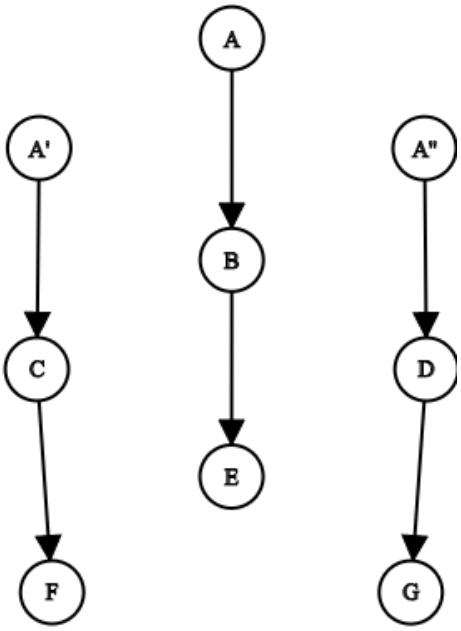


Figura 3.12: Visualización del 2do caso considerando A, A' y A" como instancias de una misma página.

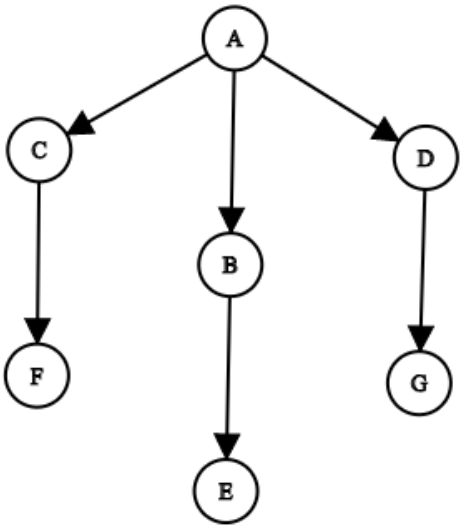


Figura 3.13: Visualización del 2do caso como resultado esperado del algoritmo.

3.3. Prueba con Datos Reales

Cuando se trabaja analizando grandes cantidades de datos, encontrar una fuente confiable de los mismos suele ser un gran desafío. Este caso no es la excepción. La falta de DataSets ya etiquetados (con sesiones reconocidas) que permitan la verificación de los datos procesados, obligó a tomar la decisión de tener que elegir un DataSet sin etiquetar [14].

El DataSet con logs de apache seleccionado forma parte de la librería de datos de Harvard, y corresponde a un sitio de compras online: https://www.zanbil.com.
Se toma el mismo dataset previamente mencionado[14], y se lo reduce con el fin de filtrar datos irrelevantes (como son las peticiones de imágenes, archivos css y javascript), y reducir el tamaño de la muestra para que sea menos costoso de procesar.

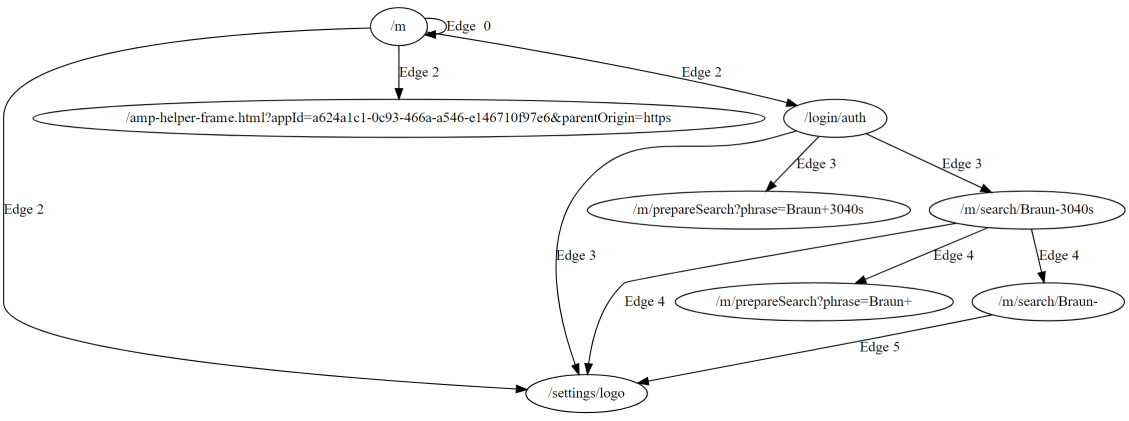
La muestra reducida consiste de logs que se expanden a lo largo de un período de entre 5 y 6 horas, y si bien los mismos fueron filtrados, algunas peticiones indeseadas permanecen en los logs. Con esta reducción se pasa de un archivo original de cerca de 3GB a uno de unos pocos cientos de MB, lo cual permitió su procesamiento completo en menos de 5 minutos.

3.3.1. Resultados

A los efectos de la visualización de casos encontrados, los logs contenidos en los mismos contendrán únicamente logs correspondientes a las sesiones visualizadas.

Ejemplo de Sesión en un Dispositivo Móvil

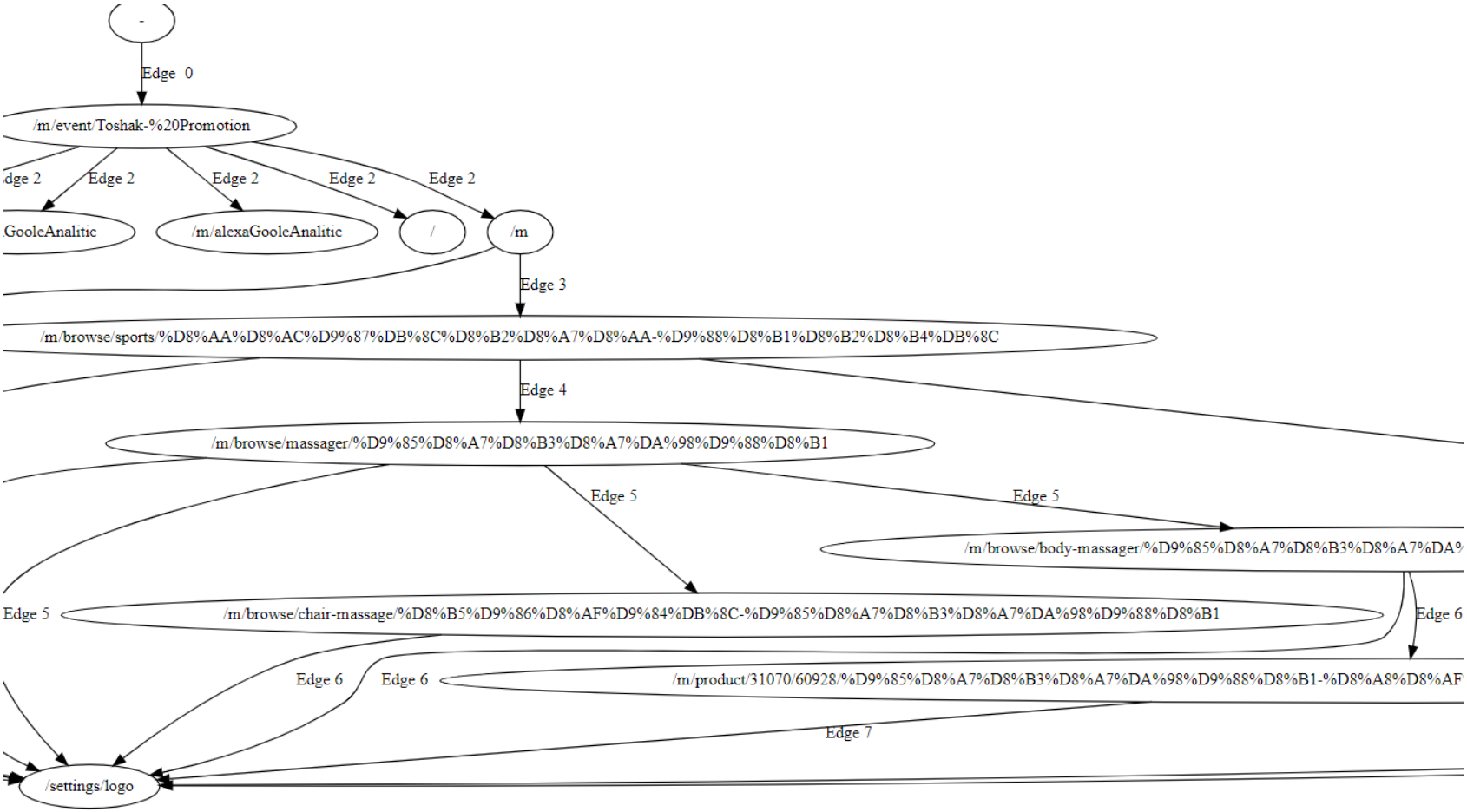
A partir del grafo de sesión se puede observar información sobre el tipo de usuario. En este caso, un usuario utilizando un dispositivo móvil, ya las URI visitadas contienen el prefijo /m correspondiente a “mobile“[7].



[7]Esta información también se encuentra disponible en el campo User Agent disponible en los logs subyacentes en los vértices del grafo.

Ejemplo de Sesión Representando Búsqueda de Artículos

Si bien la imágen se ve cortada, se puede visualizar como el usuario hace búsquedas relacionadas a deportes. En la categoría deportes busca por un masajeador, y luego su búsqueda se bifurca (lo cual se puede interpretar como que abre dos pestañas) en donde busca un masajeador para el cuerpo, y una silla masajeadora. Su búsqueda continúa luego de entrar a un producto particular, pero fue necesario recortar la sesión para que entre en la página (ver siguiente).


-
Edge 0
/m/event/Toshak-%20Promotion
dge 2
Edge 2
Edge 2
Edge 2
Edge 2
GooleAnalitic
/m/alexaGooleAnalitic
/
/m
Edge 3
/m/browse/sports/%D8%AA%D8%AC%D9%87%DB%8C%D8%B2%D8%A7%D8%AA-%D9%88%D8%B1%D8%B2%D8%B4%DB%8C
Edge 4
/m/browse/massager/%D9%85%D8%A7%D8%B3%D8%A7%DA%98%D9%88%D8%B1
Edge 5
Edge 5
/m/browse/body-massager/%D9%85%D8%A7%D8%B3%D8%A7%DA%
Edge 5
/m/browse/chair-massage/%D8%B5%D9%86%D8%AF%D9%84%DB%8C-%D9%85%D8%A7%D8%B3%D8%A7%DA%98%D9%88%D8%B1
Edge 6
Edge 6
Edge 6
/m/product/31070/60928/%D9%85%D8%A7%D8%B3%D8%A7%DA%98%D9%88%D8%B1-%D8%A8%D8%AF
Edge 7
/settings/logo

Estadísticas

A continuación se muestran algunas estadísticas sobre las sesiones generadas con los datos extraídos del data set de Harvard, seleccionando los primeros 72359 logs representando un lapso de 6 horas aproximadamente:

Usuarios identificados: 5929

10 minutos Como Delta de Tiempo

Mínimo de logs por usuario: 1
Máximo de logs por usuario: 2949
Mediana de logs por usuario: 4.0
Promedio de logs por usuario: 12.21

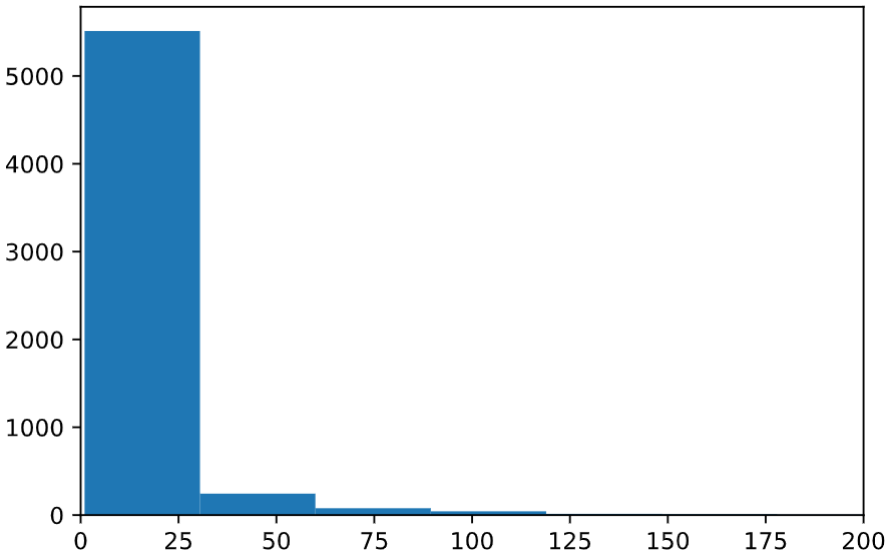


Figura 3.14: Usuario con Cantidad de Logs

Mínimo de sesiones por usuario: 1
Máximo de sesiones por usuario: 1111
Mediana de sesiones por usuario: 1.0
Promedio de sesiones por usuario: 4.16

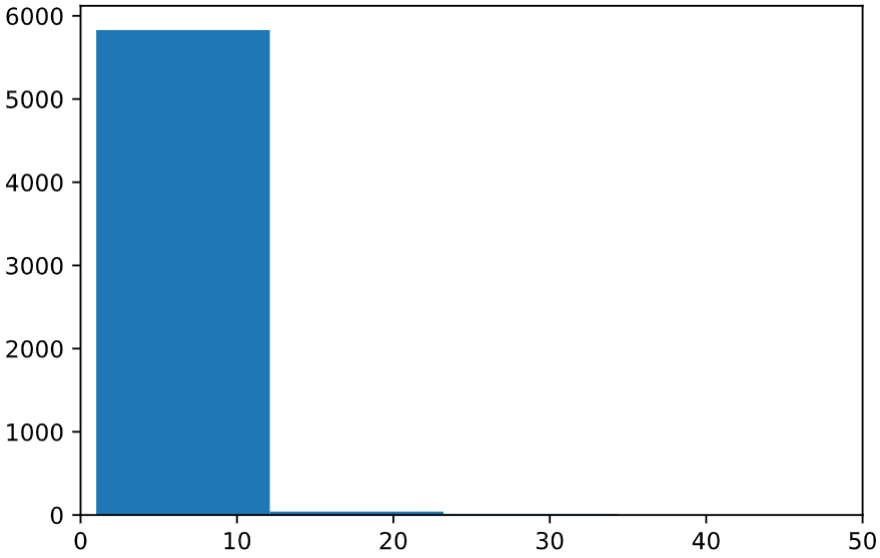


Figura 3.15: Usuarios con cantidad de Sesiones

Mínimo de logs por sesión: 1
Máximo de logs por sesión: 373
Mediana de logs por sesión: 1.0
Promedio de logs por sesión: 2.93

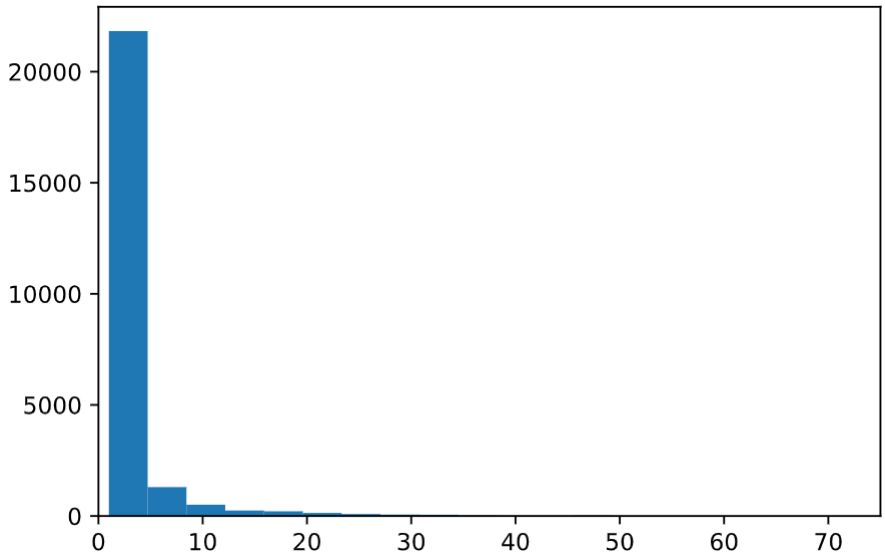


Figura 3.16: Sesiones con Cantidad de Logs

Estos resultados se ven afectados según la elección de un delta de tiempo (10 minutos en este caso) que actúa como ventana límite de tiempo entre un log y otro de una sesión. A continuación se puede ver las mismas estadísticas mostradas anteriormente, pero luego de haber modificado dicho parámetro.

5 Minutos Como Delta de Tiempo

Mínimo de sesiones por usuario: 1
Máximo de sesiones por usuario: 1111
Mediana de sesiones por usuario: 1.0
Promedio de sesiones por usuario: 4.38

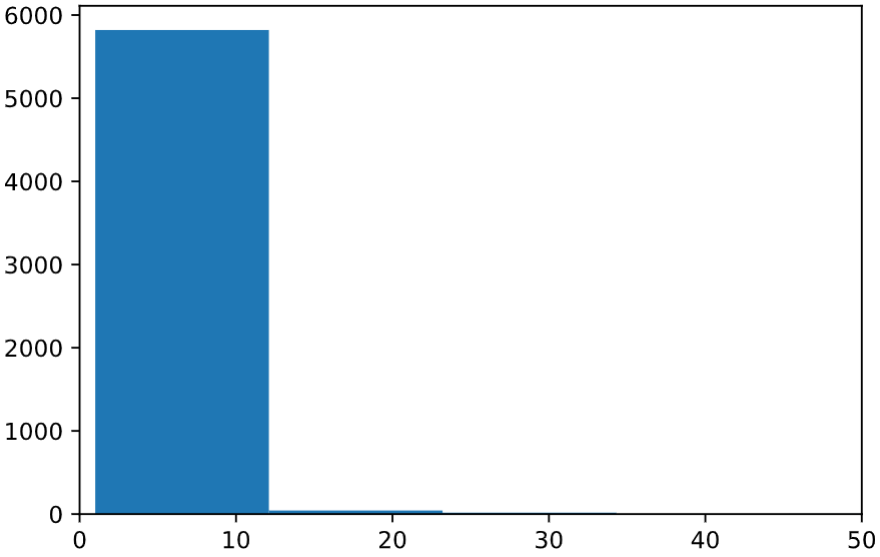


Figura 3.17: Usuarios con cantidad de Sesiones (Delta = 5 minutos)

Mínimo de logs por sesión: 1
Máximo de logs por sesión: 324
Mediana de logs por sesión: 1.0
Promedio de logs por sesión: 2.78

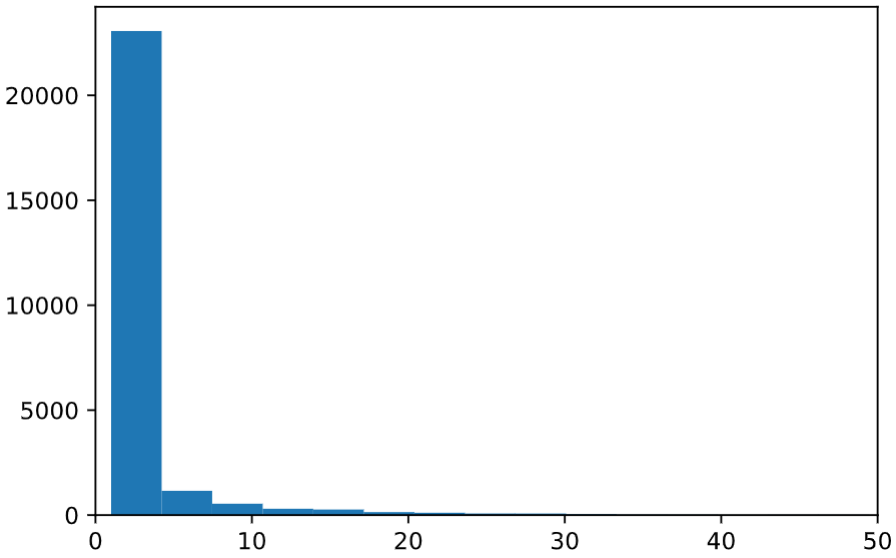


Figura 3.18: Sesiones con Cantidad de Logs (Delta = 5 minutos)

15 Minutos Como Delta de Tiempo

Mínimo de sesiones por usuario: 1
Máximo de sesiones por usuario: 1111
Mediana de sesiones por usuario: 1.0
Promedio de sesiones por usuario: 4.04

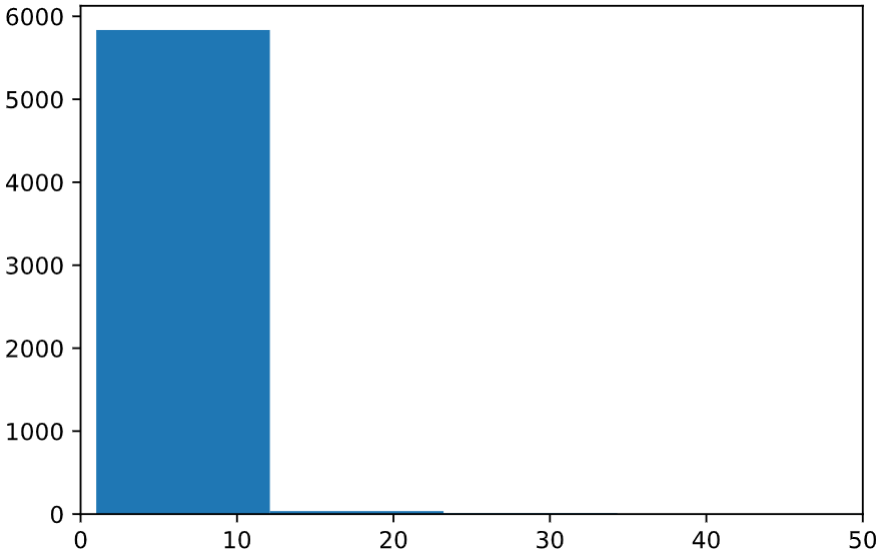


Figura 3.19: Usuarios con cantidad de Sesiones (Delta = 15 minutos)

Mínimo de logs por sesión: 1
Máximo de logs por sesión: 472
Mediana de logs por sesión: 1.0
Promedio de logs por sesión: 3.02

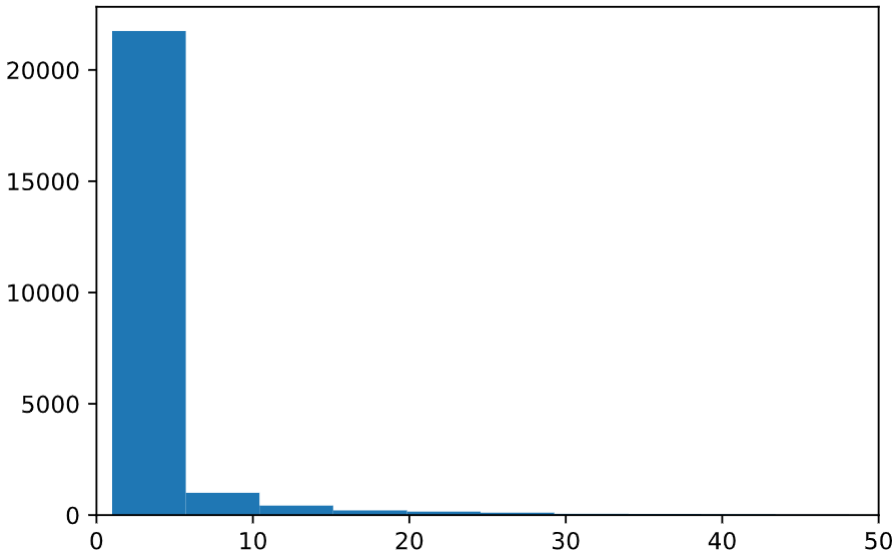


Figura 3.20: Sesiones con Cantidad de Logs (Delta = 15 minutos)

Comparación de Resultados

Si bien las variaciones en los resultados estadísticos son pequeñas, los resultados son esperados. Para un delta de tiempo de 5 minutos, se espera que las sesiones identificadas sean más cortas que para una ventana de tiempo más grande ya que entran menos logs en la misma. Al identificarse sesiones más cortas también aumenta la cantidad de sesiones identificadas, ya que los logs restantes deben ser adjudicados a alguna sesión.

Para las ventanas de 10 minutos y 15 minutos se ve un aumento en cuanto a la cantidad de logs agrupados en una sesión, y una disminución en la cantidad de sesiones identificadas. Esto parece ser cierto siempre que se aumenta el delta de tiempo utilizado por el algoritmo.

Capítulo 4

Conclusiones

A lo largo de este trabajo se realizó un relevamiento de información respecto a técnicas de sesionización. La información recabada fue analizada, y se seleccionaron las técnicas que fueran más capaces de resolver la problemática plateada.

Las mismas contenían descripciones a alto nivel sobre el concepto de sesión, y sobre los algoritmos de sesionización. Por esta razón fue necesario establecer una definición de sesión, para luego poder diseñar e implementar algoritmos de sesionización.

Se implementaron dos algoritmos, uno basado en heurísticas de tiempo, y otro basado en heurísticas de tiempo y referencia. El primero sirvió para familiarizarse con el concepto de sesión a nivel de código, y con las tecnologías utilizadas. Dada su relativa simpleza y baja capacidad para armar sesiones correctamente, es que se decide no probar con datos reales. En cambio, el segundo algoritmo si fue expuesto a experimentación con datos reales.

No se pudo encontrar datos ya procesados, es decir, un conjunto de logs que ya tuviera identificadas sesiones correctamente. Por esta razón es difícil evaluar los algoritmos de sesionización implementados, y para ello se diseñan casos base inspirados en los algoritmos de manera que cada sesión pueda ser expresada como cualquier combinación de dichos casos.

Aún así, al inspeccionar a mano parte de las sesiones identificadas, se puede decir que el segundo algoritmo identifica y arma sesiones relevantes para su posterior estudio que es un caso de uso científico.

4.1. Mejoras a Futuro

Se identifica a futuro la posibilidad de elegir el delta de tiempo utilizado por la heurística de tiempo dinámicamente según el data set elegido. Como se menciona en User Session Identification Based on Strong Regularities in Inter-activity Time[5], es posible utilizar técnicas de aprendizaje automático (como la utilización de un Gaussian Mixture Model), para poder determinar dicho delta de tiempo para que optimice la precisión y exactitud de los algoritmos implementados frente al data set, o tipos de datos utilizados.

4.2. Utilidad de los Resultados

A partir de las sesiones identificadas es posible crear y entrenar modelos de aprendizaje automático capaces de armar sesiones rápidamente. No sólo ello, si no que también sería posible identificar distintos tipos de transacciones, por ejemplo: determinar que una sesión corresponde a una compra online.

Esto habilita un abanico de posibilidades en el contexto de la ciencia de datos sobre el uso de páginas web, permitiendo reconocer comportamientos de usuarios y actuar sobre los mismos.

Bibliografía

[1] Wikipedia. "session (web analitycs)", agosto 2020. [Online]. Available: https://en.wikipedia.org/wiki/Session_(web_analytics)

[2] P. Sengottuvelan et al., *An Improved Session Identification Approachin Web Log Mining for Web Personalization.* Journal of Internet Technology, 2015.

[3] Wikipedia. "session reconstruction", agosto 2020. [Online]. Available: https://en.wikipedia.org/wiki/Session_(web_analytics)#Session_reconstruction

[4] Wikipedia. "heuristic", agosto 2020. [Online]. Available: https://bit.ly/3f0Houb

[5] Aaron Halfaker et al., *User Session Identification Based on StrongRegularities in Inter-activity Time.* arXiv.org, 2017.

[6] Yongyao Jiang et al., *Reconstructing Sessions from Data Discovery and Access Logs to Build a Semantic Knowledge Base for Improving Data Discovery.* International Journal of Geo-Information, 2016.

[7] Wikipedia. "hierarchical clustering", agosto 2020. [Online]. Available: https://en.wikipedia.org/wiki/Hierarchical_clustering

[8] wikipedia. "jenks' natural break optimization", agosto 2020. [Online]. Available: https://en.wikipedia.org/wiki/Jenks_natural_breaks_optimization

[9] Neel Guha, *Semantic Identification of Web Broswing Sessions.* arXiv.org, 2017.

[10] Wikipedia. "multilayer perceptron", agosto 2020. [Online]. Available: https://en.wikipedia.org/wiki/Multilayer_perceptron

[11] Pablo Enrique Roman et al., *Identifying User Sessions from Web ServerLogs With Integer Programming.* Inteligent Data Analysis, 2014.

[12] Neetu Anand et al., *Identifying the User Access Pattern in Web Log Data.* Internation Journal of Computer Science & Information Technologies, 2012.

[13] G. Neelima et al., *Predicting user behavior through Sessions using the Web log mining.* International Conference on Advances in Human Machine Interaction, 2016.

[14] Harvard. "harvard's example dataset from an online shop", agosto 2020. [Online]. Available: https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/3QBYB5