UNIVERSIDAD ANDRES BELLO

Facultad de Ingeniería

Políticas de Admisión para Cache basada en Comportamiento de Usuario

Tesis de pregrado para optar al título de Ingeniero Civil Informático.

Autores:

Felipe Andrés Choque Henríquez.

Profesor tutor: Carlos Luis Gómez Pantoja.

Junio, 2016

.

CONTENIDO

.

.
Tabla de ilustraciones

.

.

.

.

.

RESUMEN

Actualmente el acceso a internet es cada vez más fácil. Con esta facilidad para acceder a internet la cantidad de personas que está aumentando cada vez más y lo seguirá haciendo en los siguientes años.

Este aumento a significado muchos retos para las aplicaciones y servicios que se ofrecen en internet. Esto debido a que tienen que idear nuevos métodos y estrategias para dar respuesta a este crecimiento explosivo de las personas que usan estos servicios.

Dentro de estas aplicaciones web están los buscadores como Google o Yahoo!, los cuales usan disantos métodos para seleccionar de mejor manera lo que guardan en sus servidores y así dar buenas respuestas a las consultas que hacen los distintos usuarios.

Para seleccionar las consultas y respuestas que se quedan en los servidores, se emplean distintas políticas. Estas políticas pueden ser de desalojo y admisión, sin embargo estas últimas están poco estudiadas.

Por esta razón este trabajo se centra en estas políticas. Para lo cual se diseña un método, con el cual se puede diseñar una política de admisión. Este método consiste en estudiar logs de consultas reales, para luego encontrar características que se repiten y permitan clasificar de mejor forma las consultas que se dejan entrar a los servidores. El análisis de los logs de consultas permitió encontrar una similitud de los resultados con una distribución gaussiana.

Basándose en esta distribución, se diseñan una política de admisión lo que busca mejorar lo que se deja ingresar a los servidores y con esto mejorar el los resultados que se registran sin usar estar política de admisión.

.

1. INTRODUCCIÓN

1.1 Motivación

En la actualidad el acceso a internet se ha vuelto cada vez más común lo que ha producido un explosivo aumento en la cantidad de personas que acceden a internet. Este aumento ha causado que la cantidad de datos que tienen que manejar los distintos servicios que se proveen en internet sea cada vez más altos. Con este aumento en el flujo de datos surgen nuevos requisitos que es necesario resolver para que los servicios que se ofrecen en internet puedan dar una buena respuesta a las expectativas de los usuarios. Los nuevos requisitos son de distinta índole y abarcan muchas áreas. Por esta razón, este trabajo se centra en el área de aplicaciones web. Más específicamente en el cache, que es lugar donde se almacenan las consultas que hacen los usuarios y las respuestas pre-computadas a estas consultas.

Debido al alto tráfico que existe actualmente en internet y a que éste seguirá aumentando en los próximos años, optimizar el espacio que destinan los servidores a almacenar las consultas y las respuestas pre-computadas se vuelve relevante. Optimizar este espacio evita que se gasten recursos computacionales en almacenar consultas y respuestas que no es necesario almacenar.

Para gestionar el cache de las aplicaciones web, se emplean distintas políticas que permiten definir que sale y que entra a cache. Estas políticas pueden ser de admisión y/o desalojo. Estas dos políticas son importantes para que los motores de búsqueda puedan dar respuestas de manera eficiente a las consultas. De estas dos, las que se utilizan mayormente son las de desalojo.

Dado esto, es que este trabajo se centra en las políticas de admisión, por lo que se propone una política de admisión basada en cómo los usuarios hacen sus consultas. En primer lugar, se hace un estudio del comportamiento de usuario

.
a través del análisis de log de consultas reales, donde se clasifican según la cantidad de términos de la consulta. Luego, se hace un estudio estadístico de estas clasificaciones para determinar patrones que permitan diseñar políticas que maximicen la tasa de hit en el cache. Para esto se utiliza un algoritmo que separa la información rescatada del log de consultas, como número de términos, tasa de hit, tasa de desalojo, número total de consultas, entre otras estadísticas.

La hipótesis a comprobar con este análisis es que los usuarios tienen un comportamiento que sigue un patrón, el cual permite diseñar mejores políticas de admisión. Al realizar este análisis se pudo comprobar que todas las consultas separadas por el número de términos siguen un comportamiento que se asemeja a una campana de gauss. Para hacer el estudio más preciso, se ajustan estas curvas a una distribución gaussiana. Luego, se procede a establecer un mecanismo que permita maximizar la tasa de hit.

Ya hecho este análisis se propone una admisión basado en percentiles. Al ajustar la curva obtenida, es posible filtrar las consultas que entran según su cantidad de términos, para así cubrir un porcentaje específico de consultas.

Para corroborar si existen mejoras, se modifica el algoritmo para filtrar por percentiles y verificar si aumenta la tasa de hit. Los percentiles van desde cien por ciento de admisión bajando en un cinco por ciento hasta llegar a un cincuenta por ciento. Con estos datos se hace una vez más el análisis estadístico y se extraen nuevas observaciones.

1.2 Contribución de la tesis

Como se menciona anteriormente, este trabajo está enfocado en el estudio de los distintos mecanismos para la gestión del cache. Principalmente este

.
trabajo se enfoca en las políticas de admisión. Las contribuciones principales de este trabajo son las siguientes:

1. Se hace un estudio del comportamiento de usuario basado en el análisis de log de consultas.
2. Se detallan características de las consultas hechas por los usuarios.
3. Se identifican las principales políticas utilizadas en la gestión del cache.
4. Se estudian los log de consultas y se identifican consultas que son poco relevantes almacenar en cache. Con esto se diseña un mecanismo que no deja ingresar a cache a estas consultas, para así optimizar el espacio del cache que es limitado y así privilegiar las consultas más solicitadas.
5. Un método replicable para establecer una política de admisión basada en percentiles.

1.3 Organización de la tesis

En el capítulo 1 se encuentra la introducción, donde se encuentra la sub sección motivación que explica porque se realiza este trabajo, la contribución realizada y también cómo se organiza este documento.

En el capítulo 2 se describe el problema que este trabajo aborda, el cual es la optimización del cache mediante la implementación de una política de admisión. También se mencionan las razones por la cual se producen los problemas en los sistemas de búsquedas y los servicios web. Se mencionan las metas generales de este trabajo, como también por qué una política de admisión y no una de desalojo.

En el capítulo 3 se detalla el objetivo general de este trabajo y luego se sigue con los objetivos específicos. Además se establece la hipótesis.

.

En el capítulo 4 se establece el marco teórico, y se comienza por explicar qué es el cache y los tipos de cache que hay. También se explican los tipos de consultas que se nombran en este trabajo, como las consultas en ráfagas, permanentes y periódicas; como también los tipos de políticas para la gestión del cache. Además se define el comportamiento de usuario.

En el capítulo 5 se expone el estado del arte donde se explican los distintos métodos encontrados en la literatura para la gestión del cache.

En el capítulo 6 se detallan y explican los experimentos realizados, para analizar el comportamiento de los usuarios y luego la forma que serán empleados para el desarrollo de las políticas de admisión. Se muestra los resultados de los experimentos realizados mediante la aplicación de la política de admisión a las distintas clases de consultas.

En el capítulo 7, se presenta las conclusiones de este trabajo.

.

2. DESCRIPCIÓN DEL PROBLEMA

Debido al gran aumento en el tráfico de internet, es que en los sistemas de búsquedas y los servicios web en general se producen problemas tales como:

- Sobrecarga de servidores por eventos masivos de los usuarios sobre un tópico en particular.
- Respuesta lenta a peticiones por gestión inadecuada de recursos.
- Gasto de recursos al procesar peticiones no relevantes.

Los tres puntos anteriores describen de manera general los problemas con los que tienen que lidiar constantemente tanto los sistemas de búsqueda como los servicios web. Es por esto que encontrar mecanismos para minimizar el impacto de estos problemas se hace relevante.

La sobrecarga en los servidores es debido a un evento de importancia local o mundial, que genera en las personas un interés importante, provocando que empiecen a buscar información en internet. Generalmente este interés es en un periodo de tiempo reducido. A este tipo de petición se le llama consulta en ráfaga. Este tipo de consulta puede generar severos problemas si no son detectados a tiempo. Enfocándose en este tipo de consultas como ejemplo, es que el cache también toma un papel importante no en la detección, sino en cómo puede ayudar a minimizar el impacto de estas ráfagas en los servicios web y sistemas de búsquedas, ya que el cache almacena las respuestas pre-computadas a estas consultas. Como el espacio del cache es limitado, ocupar este espacio en respuestas innecesarias es poco eficiente. Por este motivo se quiere evitar que tanto las consultas innecesarias o poco relevantes como también sus respuestas no se almacenen en cache. Para esto se propone una política de admisión que restringe la entrada a cache.

.

Este trabajo apunta a encontrar algún patrón en el comportamiento de usuario que permita diseñar una política de admisión al cache que maximice la tasa de hit y disminuir los fallos tales como:

- Aumento de miss (no está la respuesta en cache) en el cache.
- Almacenamiento de consultas no relevantes.

Permitir que cualquier consulta entre al cache, causa que el cache sea mal utilizado y por lo tanto sea poco eficiente.

El problema de los servicios de cache, es que, actualmente la mayoría de las consultas hechas por los usuarios son admitidas en cache, lo cual dado el gran número de usuarios que actualmente está en internet, es algo poco eficiente. Luego con algoritmos de desalojo como Least Recently Used (LRU) o Least Frequently Used (LFU) son desalojadas del cache las consultas poco relevantes. La idea que se tiene en este trabajo es evitar esto, diseñando una política de admisión que ignore consultas que no son muy preguntadas por los usuarios, ya que estas consultas quitan espacio a consultas que sí son relevantes tener en cache.

La dificultad de encontrar una política que ayude a maximizar la tasa de hit, es que el comportamiento de usuario no es uniforme y suele seguir patrones. Por ejemplo, el tráfico de internet durante el día y la noche tiene variaciones pero básicamente siempre sigue el mismo patrón (alto en el día y bajo en la noche). Otra característica es que las personas suelen preguntar mucho por un grupo reducido de términos, lo cual es un comportamiento zipfiano.

Los problemas de los motores de búsqueda y aplicaciones web de gran escala, principalmente son el desbalance de carga, inexistencia de políticas de admisión, políticas de desalojo o reemplazo ineficientes. Este trabajo se centra particularmente en las políticas de admisión.

.

A continuación en la ilustración 1, se muestra un diagrama de Ishikawa que ayuda a entender los problemas que causan que el cache sea poco eficiente.

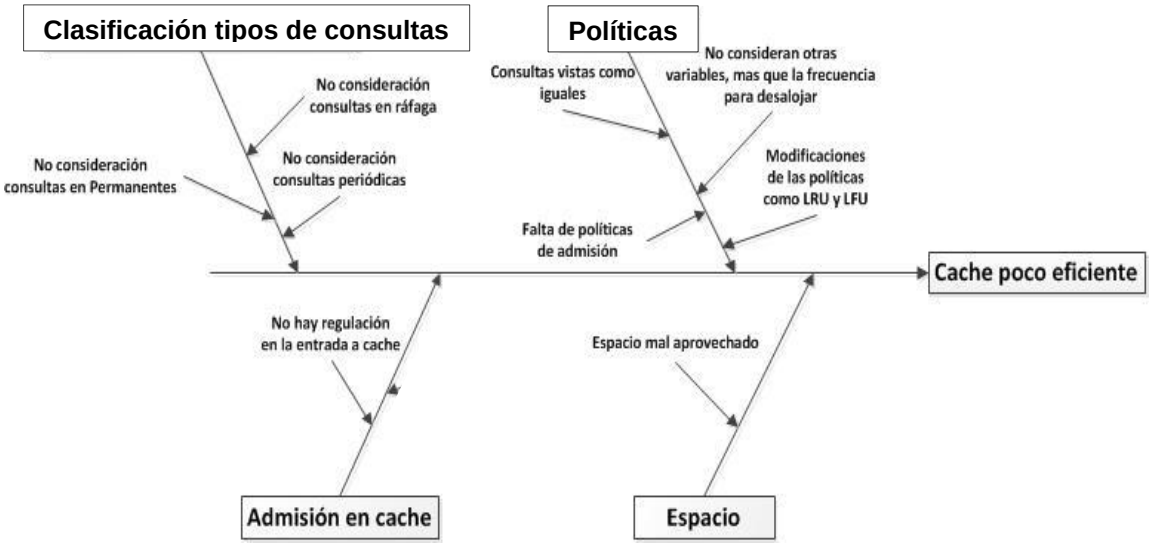


Ilustración 1. Diagrama Causa efecto – (Fuente: Elaboración propia)

Dentro de las causas detalladas en la ilustración 1 están, primero la clasificación del tipo de consultas que hay: consultas permanentes, periódicas y en ráfaga. Estas son necesarias de identificar debido a que cada una de este tipo de consultas debiera ser tratada de distinta forma en los servicios de cache, ya que cada una posee características diferentes. Luego, están los tipos de políticas empleadas en los servicios de cache. Las políticas usadas generalmente en los servicios de cache suelen omitir características que pueden ser relevantes para el diseño de mejores políticas: las consultas no son de tamaño homogéneo, falta de políticas de admisión o considerar solo la frecuencia a la hora de desalojar consultas del cache.

Otro punto importante son no solo diseñar políticas encargadas de desalojar del cache lo que no se ocupa, sino también restringir lo que entra para así aprovechar de mejor manera el espacio limitado.

.

3. OBJETIVOS

3.1 Objetivo general

Para mejorar la tasa de hit en el cache es necesario diseñar políticas de admisión que impidan el ingreso de ciertas consultas a cache. Es por esto que el presente trabajo tiene como objetivo general:

- Crear un método replicable para crear una política de admisión que mejore la eficiencia del cache.

3.2 Objetivos específicos

Para lograr el objetivo general de la tesis se requiere cumplir con los siguientes objetivos específicos.

- Separar las consultas en número de términos y cantidad de letras.
- Hallar un patrón en el comportamiento de los usuarios, al realizar consultas.
- Identificar las consultas que no debieran estar en cache.
- Encontrar un método matemático para filtrar las consultas que ingresan a cache.
- Determinar si la tasa de hit en el cache mejora con la política de admisión.

.

3.3 Hipótesis

En este trabajo se quiere determinar cómo impactan las políticas de admisión en la eficiencia del cache. Para esto se formula la siguiente hipótesis.

“Utilizar políticas de admisión junto con las políticas de reemplazo, mejoran la tasa de hit en el cache”.

Para comprobar esto se hace un estudio estadístico de los logs de consultas, con el objetivo de encontrar patrones que permitan diseñar una política de admisión. Una vez hecha la política de admisión en base a los patrones encontrados, se diseña un algoritmo que mezcla la política de admisión con una política de reemplazo. Luego se harán pruebas con log de consultas en los cuales se verifica cuanto impacta una política de admisión en la tasa de hit.

3.4 Justificación de la solución

Al hacer la revisión de la literatura y los distintos métodos que se usan para gestionar el cache, se evidencia que la mayoría de los trabajos de investigación se enfocan en políticas de reemplazo, dejando de lado las políticas de admisión. Sin embargo, ciertos trabajos indican lo importante que una política de admisión en conjunto con una política de reemplazo puede llegar a ser en la gestión del cache.

Por ejemplo en (Aggarwal, Wolf, & Yu, 1999), se menciona que se debe elegir mejor lo que entra en cache y para eso se debe diseñar una política de admisión. En (Baeza-Yate, Junqueira, Plachouras, & Witschel, 2007) también menciona que una política de admisión ayuda a mejorar la tasa de hit en el cache, y lo hacen mediante un método que intenta predecir si una consulta será frecuente más adelante. En (Long & Suel, 2006) se menciona que, según los

.
experimentos que ellos desarrollaron, las políticas de admisión en conjunto con políticas de desalojo son cruciales para que los motores de búsquedas tengan un mejor rendimiento.

En el estado del arte se detallan más trabajos que abordan la falta de políticas de admisión para complementar a las políticas de reemplazo ya conocidas como Least Recently Used (LRU) y Least Frequently Used (LFU).

.

4. MARCO TEÓRICO

4.1 Cache

En los servicios web de gran escala y motores de búsquedas existe un lugar de almacenamiento destinado a guardar las consultas y respuestas pre computadas que son frecuentemente preguntadas. Esto es el ideal, pero este ideal rara vez se logra. A este espacio de almacenamiento se le llama cache. El cache es una de las partes más importante dentro de los servicios web y motores de búsqueda (Aggarwal et al., 1999), (Altingovde, Ozcan, & Ulusoy, 2009), (Baeza-Yates et al., 2008), (Baeza-Yates et al., 2007). El cache básicamente es un lugar de almacenamiento de acceso rápido, en el cual se guardan las consultas más frecuentes hechas por los usuarios y sus respuestas. El cache es limitado, es por eso la importancia de encontrar políticas que maximicen su eficiencia. Estas políticas pueden ser de admisión y de reemplazo o desalojo. Estas dos políticas se explican más adelante.

En los motores de búsqueda, el cache es imprescindible, debido a la gran cantidad de consultas que realizan los usuarios hoy en día. El cache está encargado de almacenar consultas y sus respuestas, pero el cache no almacena todas sino que las consultas más frecuentes. Pero existe un problema y es que el cache es limitado, por lo que al llenarse se tiene que desalojar alguna consulta del cache. Para esto, generalmente se utiliza un algoritmo de reemplazo llamado LRU (Least Recently Used), el cual elige la menos usada recientemente para el desalojo. Esta política, al igual que casi todas las de reemplazo, tiene un problema: todas las consultas tienen derecho a entrar en cache al menos una vez, incluso si dicha consulta solo se utiliza una vez. Es entonces, que esta consulta no se debe aceptar en cache, ya que ocupa un lugar por un “X” tiempo en donde quizás puede haber entrado otra consulta que sí sea muy solicitada. Basándose en esto, es que no solo se deben considerar políticas de desalojo

.
sino que también de admisión. Estas políticas de admisión evitarían que entraran en cache consulta aisladas o que se consulten una sola vez o ninguna. La finalidad de prohibir que entren estas consultas es mantener en cache solo los datos relevantes y así mejorar la forma en que es ocupado el espacio en cache. Con esto, se aumenta la tasa de hits.

La importancia del cache radica en que los motores de búsqueda necesitan responder de forma rápida a las consultas hechas por los usuarios y, dado que la cantidad de usuarios ha crecido rápidamente, es que se necesita crear nuevas políticas de admisión para que el cache se utilice de una manera más eficiente.

4.2 Cache Distribuido

Es una tecnología que utiliza el concepto tradicional de cache, el cual está pensado para un solo nodo, y lo lleva a varios servidores los cuales están comunicados entre sí para un mejor trabajo y mayor almacenamiento. En este tipo de cache, cada nodo tiene una cantidad limitada de almacenamiento, en donde almacenan las consultas de los usuarios y sus respuestas. También puede ser visto como varios nodos los cuales tienen ciertas consultas y cuando estas son hechas por los usuarios son redirigidos a uno de estos nodos que pueda contener la petición hecha.

4.3 Cache Estático

En este cache se almacenan los resultados ya procesados por los motores de búsquedas. La forma en que se llena este cache es utilizando registros de las consultas hechas en los motores y se escogen las más frecuentes. Estas se guardan ya que es posible que esas consultas sean consultadas frecuentemente

.
y no es necesario sacarla de cache. En otras palabras, este cache almacena las peticiones del tipo permanente.

Este cache se puede traducir como el problema de la mochila, la cual tiene una capacidad predefinida en la que se deben almacenar las consultas más valiosas siguiendo algún criterio.

4.4 Cache Dinámico

El cache estático almacena las consultas que son constantemente solicitadas. Si bien esto es efectivo, puede dejar afuera consultas que son altamente frecuentes pero en un intervalo de tiempo reducido o peticiones del tipo periódico.

Para manejar esto está el cache dinámico, el cual se actualiza a medida que las consultas cambian. Este tipo de cache no requiere de registros anteriores de las consultas para almacenarlas en el cache. Este cache generalmente comienza vacío y se empieza a llenar a medida que se hacen consultas al motor de búsqueda. Si el cache está lleno y una consulta no está en cache, se emplean políticas de desalojo para dar espacio a la nueva consulta.

Este trabajo está interesado en este tipo de cache, para lo cual se propone una política de admisión que mejore la selección de consultas que se dejan ingresar a este cache.

4.5 Comportamiento de Usuario

El comportamiento de usuario se puede definir como, un área que intenta encontrar características, patrones o conductas de los usuarios en algún contexto específico, como por ejemplo en las redes sociales, en algunos sistemas de

.
compras, y otras. Esto es de importancia para las empresas en general, ya que pueden mejorar sus ofertas de productos o hacer sistemas más eficientes que consideren alguna conducta repetitiva de los usuarios. Esto está ampliamente estudiado en el área del comercio, debido a que las grandes empresas intentan identificar ciertos comportamientos o gusto de las personas para que así puedan diseñar ofertas más eficaces dirigidas a los usuarios. Esto también es de interés de las empresas para hacer sitios web dinámicos, los cuales puedan dar ofertas más personalizadas según los gustos o preferencias de las personas.

Encontrar patrones de comportamiento de usuarios es bastante difícil ya que las personas son cambiantes. A pesar de este comportamiento cambiante, se han encontrado ciertos patrones que se repiten, como por ejemplo la ley que formuló George Kingsley Zipf (Zipf, 1949), llamada la ley de Zipf. Esta ley dice que una pequeña cantidad de palabras son muy usadas y una gran cantidad de palabras son poco usadas por los usuarios. Esto ha sido altamente estudiado en el contexto de la web (Gómez Pantoja, 2013).

El comportamiento de usuario también está presente en el área de los motores de búsqueda y servicios web, como por ejemplo en las consultas hechas por los usuarios a estos servicios. Estas consultas siguen un comportamiento que se puede clasificar en: consultas permanentes, consultas en ráfaga y consultas periódicas (Gómez Pantoja, 2013).

4.6 Consultas en ráfaga

Una consulta en ráfaga se puede definir como un evento en el cual los usuarios se ven interesados por un tema en particular y comienzan a consultar de manera intensa sobre dicho tema. Esto ocurre en una ventana de tiempo acotada, por lo que suele producir problemas en los servidores que están encargados de esas consultas. Esto resulta relevante en los sistemas de cache,

.
ya que al ser un evento inesperado o repentino, puede ocurrir que estas consultas no se encuentren pre computadas. Al diseñar una política de admisión se tendría que tener en cuenta que cuando esto es detectado, directamente se tendría que enviar a cache sin pasar por esta política de admisión. Para representar mejor esto se toma el siguiente ejemplo del trabajo de (Subašić & Castillo, 2010):" En 18 de octubre 2008, después de haber sido parodiado varias veces en el programa de televisión Saturday Night Live, la político de EE.UU. Sarah Palin apareció en el show y se encontró con su imitador. Esto condujo a un aumento de x22 en la frecuencia de la consulta "snl sarah palin" en comparación con los dos días antes del evento". Esto es una consulta en ráfaga.

Un ejemplo de consulta en ráfaga conocida es "Osama Bin Laden", como muestra la siguiente imagen antes del año 2011. El interés sobre esta consulta es casi cero, sin embargo en el año 2011 aumenta bruscamente pero después de un tiempo el interés sobre esto vuelve a decaer volviendo a ser casi cero.

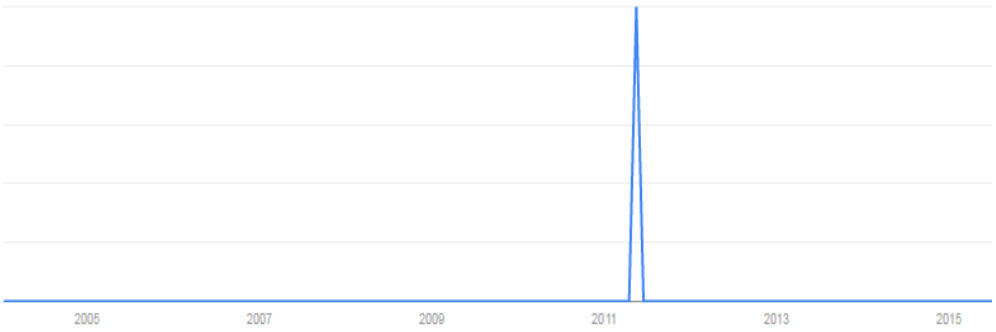


Ilustración 2: Ejemplo consulta en ráfaga "Osama Bin Laden" – Fuente: (www.google.com/trends, s.f.)

4.7 Consultas permanentes

Las consultas permanentes son aquellas que a diferencia de las consultas en ráfaga, la gran parte del tiempo tienen una alta frecuencia. Es decir, no ocurren dentro una ventana de tiempo acotada. Este tipo de consultas siempre debieran estar en cache, ya que se sabe de antemano que se harán muchas peticiones

.
sobre ellas, por lo que también deberían estar excluidas de la política de admisión. Por ejemplo, consultas permanentes actuales son “Facebook”, “YouTube”, etc.

Un ejemplo de consulta permanente es YouTube .La ilustración 3 muestra como YouTube es una consulta permanente desde el año 2007 en adelante.

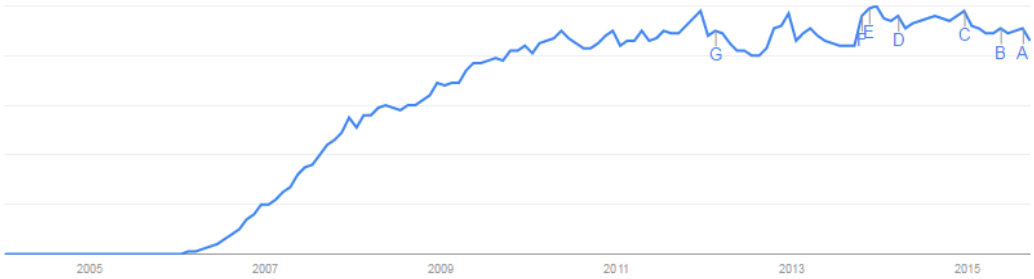


Ilustración 3: Ejemplo consulta permanente “YouTube” – Fuente: (www.google.com/trends, s.f.)

4.8 Consultas periódicas

Las consultas periódicas son aquellas que tienen un alza de actividad en forma periódica, por ejemplo cada fin de mes, cada mañana, etc. Esto quiere decir que se sabe que dicha consulta va a tener una gran cantidad de consultas en un mes o meses en particular y esto es repetitivo según el periodo de la consulta. Por ejemplo, la consulta navidad, se sabe que todos los años en diciembre comenzará a tener una alta frecuencia. Es importante identificar esto, porque se podría considerar que llegado un instante en particular, donde se sabe que hay un evento periódico, se podría aceptar inmediatamente estas consultas en cache sin necesidad que sean evaluadas por un algoritmo si son o no aceptadas.

Un ejemplo de consulta periódica es “US Open”. Durante el año hay dos eventos importantes relacionados con este nombre, el abierto de tenis y de golf.

.
A continuación en la ilustración 4, se observa como cada año en el mismo periodo la frecuencia de la consulta “Us Open” aumenta con respecto a los meses anteriores, repitiéndose esto todos los años.

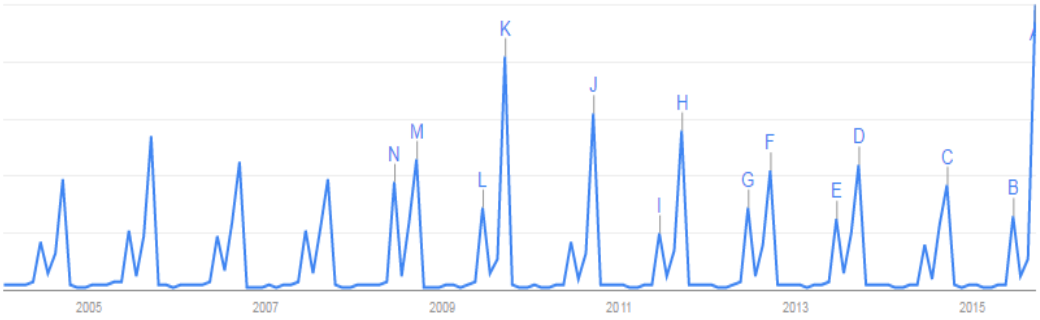


Ilustración 4: Ejemplo consulta periódica “US Open” – Fuente: (www.google.com/trends, s.f.)

4.9 Políticas de reemplazo o desalojo

El cache es un recurso limitado, por lo que cuando el cache se encuentra lleno es necesario liberar espacio para dar lugar a otras consultas entrantes. Para obtener ese espacio se utilizan políticas de reemplazo o desalojo. Estas políticas están encargadas de desalojar generalmente las consultas que menos se usan. Para esto se emplean principalmente dos algoritmos, LFU (Last frequently Used) y LRU (Last recently Used). Existen más algoritmos que buscan optimizar el cache, pero son modificaciones o mejoras que se hacen a estos dos algoritmos, por ejemplo TinyLFU (Einziger & Friedman, 2014) o WLFU (Karakostas & Serpanos, 2002). Otros algoritmos se fijan en el costo que tiene hacer el traspaso de disco a memoria, u establecen otros atributos medibles como el tamaño de una consulta, para así diseñar políticas de desalojo que consideren otros aspectos y no vean a todas las consultas como iguales.

.

4.10 Políticas de admisión

Las políticas de admisión buscan evitar que ciertas consultas entren a cache. Generalmente, todas las consultas tienen derecho a entrar al menos una vez a cache, pero esto produce que consultas que se preguntan poco o nada ocupen un lugar en el cache que puede ser ocupado por otra consulta. Básicamente, lo que se intenta es admitir solo lo que se sabe que va o puede ser consultado muchas veces. Este trabajo se centra en este tipo de políticas, tratando de encontrar algún comportamiento/patrón en los usuarios que permita diseñar una política de admisión para mejorar la eficiencia del cache.

Para lograr esto, se usa un algoritmo que permite rescatar datos de las consultas, como el largo, número de términos, cantidad de letras, etc. Posteriormente para realizar el análisis estadístico, con este algoritmo se simula el cache en el cual tiene implementada una política de desalojo. Con esto se obtienen datos tal como, la tasa de hit, la tasa de desalojo, número total de consultas, etc. Para esto, se separan en “clases de consultas”. A grandes rasgos, cada clase está definida según su cantidad de términos, y con esto se hacen los distintos análisis estadísticos para encontrar patrones en cómo los usuarios hacen sus consultas. Esto se explicará de manera más detallada en la evaluación experimental.

.

5. ESTADO DEL ARTE

Como se explicó, el cache es un elemento importante en los motores de búsqueda y en los servicios web. Trabajos como (Aggarwal et al., 1999; Altingovde et al., 2009; Baeza-Yates et al., 2008; Einziger & Friedman, 2014; Long & Suel, 2006; Ozcan, Altingovde, & Ulusoy, 2008, 2011; Zhang, Long, & Suel, 2008) se esfuerzan en optimizar el cache de los motores de búsqueda, aplicando distintas variaciones de algoritmos conocidos, principalmente LRU y LFU. Todos los trabajos referenciados intentan mejorar la eficiencia del cache, mediante la modificación de algoritmos existentes o proponiendo nuevas técnicas o atributos que los algoritmos base no consideran, como el tamaño de la consulta.

En (Long & Suel, 2006) se menciona que los motores de búsqueda tienen que responder miles de consultas por segundo. Debido a la gran cantidad de datos, una consulta puede implicar procesar miles de megabytes o más. Esto implica que una mejora en el cache, puede significar un ahorro importante de recursos en el procesamiento de consultas y sus respuestas.

Como se menciona anteriormente, los trabajos anteriores intentan optimizar el cache mediante la modificación de algoritmos base como LRU y LFU. Todas estas propuestas carecen de una arista importante a la hora de diseñar políticas para el cache, y es que la mayoría no se fija en cómo preguntan los usuarios. Existen trabajos relacionados que tratan de optimizar el cache restringiendo las consultas que entran a él, según su tamaño y lo que cuesta procesar una consulta de mayor tamaño. Otros trabajos lo ven desde el punto de vista de mejorar los algoritmos existentes. A continuación se mencionan trabajos similares que buscan optimizar el cache.

PDC (Lempel & Moran, 2003), Probability Driven Cache, es un modelo probabilístico para los motores de búsqueda. Este modelo, a diferencia de los

.
otros métodos de reemplazo, prioriza las páginas que están en cache en función del número de usuarios que actualmente está navegando en las páginas de resultado que dio la consulta. Este modelo se basa en que los resultados mostrados por los motores de búsqueda, suelen estar almacenados en cache. Sin embargo, esto no es eficiente ya que otros estudios explicados en este trabajo han demostrado que de la gran cantidad de páginas de resultados que dan los motores de búsqueda, solo la primera es visitada por la mayoría de los usuarios, por lo que almacenar todo el resto es innecesario. Entonces esta política desaloja estos resultados que no son relevantes o no tienen un flujo de visitas muy importante en ese momento y almacena las páginas de resultados que están siendo más visitadas actualmente en relación a la consulta hecha. Esta política de desalojo muestra aumentar la tasa de hit hasta en un 50% en caches grandes en comparación con políticas basadas en LRU.

TinyLFU (Einziger & Friedman, 2014) consta de dos partes, las cuales intentan aumentar la tasa de hit y además reducir la carga por el manejo de las consultas y respuestas manejadas en cache. Para esto, primero hace una versión aproximada de WLFU. WLFU utiliza una ventana de tiempo para decidir que desalojar del cache. Explicado de forma simple, TinyLFU hace una aproximación en el cálculo de los meta-datos que utiliza WLFU para decidir qué desalojar. Esto reduce el costo del manejo de meta-datos. Esto es, TinyLFU mantiene estadísticas aproximadas de la historia de las frecuencias, básicamente para reducir la cantidad de espacio requerido para mantener dicha información.

En segundo lugar TinyLFU es una política de admisión para cache. Esta propuesta viene dada por la observación de que la mayoría de los esquemas para el almacenamiento en cache se basan en políticas de desalojo, es decir qué elementos deben ser sacados de cache cuando este ya está lleno. Por lo que las políticas de admisión en general son dejadas de lado. Por lo tanto, la política de admisión de TinyLFU, decide qué consulta dejar entrar en cache. Esto lo hace de

.
la siguiente forma, por ejemplo, si una consulta es candidata a desalojo en cache, TinyLFU usa los datos aproximados almacenados de la consulta candidata a ser desalojado del cache y compara esta popularidad aproximada, con la popularidad de la consulta que quiere ingresar a cache. De ser efectivamente más popular la consulta a ingresar, se desaloja el candidato y se ingresa la nueva consulta. De no ser así se conserva el candidato a desalojo y la nueva consulta es descartada.

Admission policies for Cache of Search Engine result (Baeza-Yate et al., 2007). Si bien no se da un nombre explícito para esta política, este trabajo diseña una política la cual consiste en separar el cache en dos. La primera parte se le llama ***cache controlado***. Esta parte es gestionada por la política de admisión. La segunda parte se llama ***cache no controlado*** y acá entran las consultas que no fueron aceptadas por la política de admisión en el cache controlado.

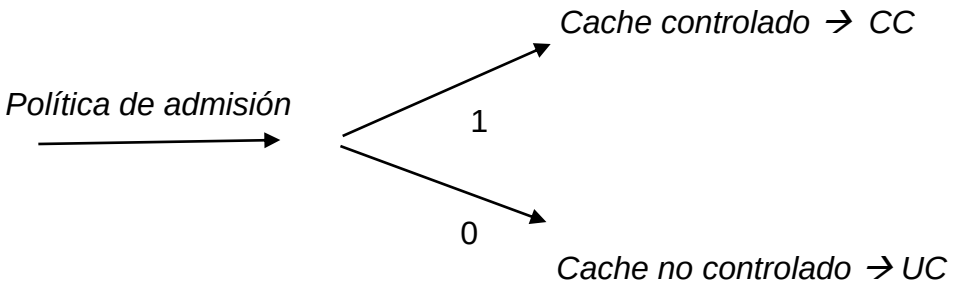


Ilustración 5: Separación cache – Fuente: (Baeza-Yate, 2007)

En la parte controlada, la política de admisión solo acepta las consultas que estima son posibles de ser preguntadas en un futuro, o que son o pueden ser populares. Las consultas que son rechazadas por la política de admisión son enviadas al cache no controlado. La finalidad de esto es tener almacenadas las consultas que pudieran ser solicitadas nuevamente por el mismo usuario en un corto periodo de tiempo, o también tener almacenadas las consultas que por alguna razón pueden convertirse en ráfaga en un corto período de tiempo.

.

Las políticas de admisión propuestas para el cache controlado son dos. La primera consiste en fijar un umbral ***K*** según el número de palabras, y solo dejar entrar al cache controlado las consultas que no superen dicho umbral. La segunda política de admisión también consiste fijar un umbral ***K.*** Pero este umbral está dado por la cantidad de caracteres no alfanuméricos encontrados en una consulta, esto debido a que es probable que una consulta con muchos caracteres no alfanuméricos nunca llegue a ser una consulta popular. A pesar de ser políticas de admisión simples, los investigadores hacen ver la importancia que es tener una política de admisión que controle lo que entre a cache.

SLRU (Aggarwal et al., 1999). Esta política de desalojo se basa en que la mayoría de las políticas de desalojo existente consideran a todas las consultas de un tamaño homogéneo, lo cual no es así. Basándose en la política LRU añaden una característica a considerar: el tamaño de la consulta. Desalojar una consulta de menor tamaño puede no ser suficiente para alojar una consulta de mayor tamaño, lo que provoca que se desalojen más de una. También pone énfasis en que si se desaloja una consulta que es recientemente usada de un mayor tamaño que la última de la lista, puede dar lugar a más consulta de menor tamaño. La forma de hacer esta selección es utilizando una heurística basada en el problema de la mochila, donde el tamaño de la consulta a entrar sería la capacidad, luego se calcula la relación costo – tamaño y se ordenan en forma creciente. Finalmente se seleccionan las consultas que estén en la parte superior de la lista hasta completar el tamaño de la consulta entrante.

PSS, Pyramidal Selection Scheme (Aggarwal et al., 1999). Es una variación del algoritmo SLRU, que también fue propuesto en el mismo trabajo. Esto lo hacen debido a que SLRU sería poco eficiente en casos reales según se indica en el mismo trabajo. Para esto se hace una pirámide donde se guardan las consultas según su tamaño. Cada nivel de esta pirámide está ordenada con la

.
política LRU. Para escoger qué consulta se debe desalojar, se selecciona en cada nivel la consulta menos recientemente usada. Entre todas las consultas seleccionadas con el algoritmo LRU de los niveles de la pirámide, se tiene que seleccionar el valor que más se adecue para desalojar, este valor se obtiene considerando tanto el tamaño de la consulta como su frecuencia. Para calcular el valor son seleccionadas las consultas menos frecuentemente usadas y luego de esas consultas se eligen las que tienen el menor valor de la multiplicación del tamaño y la frecuencia.

Hybrid Algorithms (Gan & Suel, 2009). Este algoritmo se sustenta en que la mayoría de los algoritmos de gestión del cache, está basado en políticas que intentan maximizar la tasa de hit en cache, pero ignoran el costo de procesar una consulta en cache. Ellos proponen usar una ponderación en las consultas. Otra característica importante que menciona el trabajo es que hay ciertas consultas que son ráfagas.

Para diseñar el método híbrido, propone dividir el cache en dos partes A y B de manera fija, y tener dos políticas A y B. Cada una de estas políticas asigna una puntuación a cada consulta. O sea la política “A” asigna una puntuación a su lado del cache y la “B” lo mismo. Cuando el cache tiene espacio, la consulta se inserta en cualquiera de los dos caches. Cuando se tiene que desalojar un elemento, se selecciona el elemento de menor puntuación, por ejemplo en “A”. Sin embargo antes de desalojar la consulta, se intenta insertar en la parte “B” del cache. Si la política le asigna una puntuación más alta que la mínima en la parte “B” del cache, entonces se desaloja la consulta de “B” y no la de “A”. Esto se podría repetir unas veces más, pero al final siempre un elemento es desalojado. Para probar estas políticas híbridas combinan políticas tanto tradicionales, con políticas que usan ponderaciones.

.

COST-AWARE STATIC AND DYNAMIC CACHING (Ozcan et al., 2011). En este trabajo se detallan varias políticas para la gestión del cache, pero se diferencian en que se considera el costo de procesar la consulta. Esto debido a que la mayoría de las políticas consideran a las consultas solo desde el punto de vista de los aciertos que se tenga en cache. Sin embargo en ocasiones traer una de estas consultas a cache puede resultar no beneficioso, si el costo de procesar la consulta tanto en tiempo como en términos de rendimiento es muy alto.

Considerando el costo de procesar las consultas, se proponen distintas técnicas para la gestión del cache estático y dinámico.

A continuación se presentan las políticas para cache estático.

- **Most Frequent (MostFreq).** Este es el método básico que se encarga de llenar el cache con las consultas con las frecuencias más altas.

- **Frequency Then Cost (FreqThenCost).** Esta política, es una modificación de la política **MostFreq**. *FreqThenCost* indica que las consultas siguen una distribución de ley de potencias. Esto quiere decir que pocas consultas tienen alta frecuencia mientras que muchas tienen una baja frecuencia. Considerar la frecuencia de la consulta no es efectivo ya que en tamaños grandes de cache, se tienen muchas consultas con baja frecuencia por lo que una política de desalojo tiene que decidir al azar cuál quitar. Por esto se define un atributo más, el costo de la consulta, lo que se traduce en que cada consulta tiene un par (Fq, Cq), que representa su frecuencia y costo. Para llenar el cache se seleccionan las consultas con mayor frecuencia y cuando dos consultas son de la misma frecuencia se saca el que tiene el costo más alto.

- **Stability Then Cost (StabThenCost).** Esta política se basa en el hecho de que una consulta puede cambiar su popularidad en un intervalo de

.

tiempo determinado. Es por esto que introduce un nuevo parámetro para definir la importancia de la consulta. Este parámetro es el costo de estabilidad de la consulta (*QFS*), la cual indica que tan estable es la consulta en su frecuencia. Esto se debe a que para ser almacenado en cache su frecuencia debe ser alta en un periodo de tiempo (su frecuencia debe tener pocas variaciones en esa ventana de tiempo), por lo tanto la consulta se define por el par (*QFS, Cq*), es decir, el cache se llena primero en base al parámetro *QFS* y luego por *Cq.*

- **Frequency and Cost (FC_K)**. Esta política calcula el valor esperado de una consulta como el producto de la frecuencia Fq y costo Cq de la consulta. Es decir, se espera que la consulta sea tan frecuente en el futuro como en los registros anteriores. En este trabajo hacen una modificación a esta fórmula debido a que es posible que consultas que fueron muy frecuentes en el pasado sigan apareciendo como frecuentes, incluso si actualmente su frecuencia es baja, o valores que tuvieron una frecuencia baja en el pasado puede que no aparezcan. Por esta razón sesgan esta fórmula para enfatizar los valores de frecuencias más altos y despreciar los inferiores. La fórmula sesgada es la siguiente.

$$Value(q) = C_q * F_q^k \ Donde\ K > 1$$

A continuación se detallan las políticas para la gestión del cache dinámico.

- **Least Recently Used (LRU).** Esta estrategia escoge la consulta menos recientemente referenciada como víctima de desalojo del cache.

.

- **Least Frequently Used (LFU).** Cada vez que una consulta entra a cache se le asocia un valor. Este valor es la frecuencia de veces que fue solicitada dicha consulta. Cuando el cache está lleno, esta política escoge aquella con frecuencia más baja para desalojar del cache.

- **Least Costly Used (LCU).** Esta política elige las consultas que tienen el costo más alto usado últimamente para el desalojo.

- **Least Frequently and Costly Used (LFCU_K).** Es la versión dinámica de la política detallada anteriormente **(FC_K)**. Usa exactamente la misma fórmula para calcular el costo de la consulta.

- **Greedy Dual Size (GDS).** Esta política agrega un valor H-valor para cada consulta en cache. El H-valor se calcula con la siguiente fórmula.

$$H - value(q) = \frac{C_q}{S_q} + L$$

Donde *L* es un factor de envejecimiento que se inicializa en cero en el comienzo, *Cq* es el costo de la consulta y *Sq* es el tamaño de la página de resultados asociados a la consulta. Esta política escoge a la consulta con el "*H – value"* más pequeño. Si esta consulta es nuevamente solicitada, *L* se inicializa nuevamente.

- **Greedy Dual Size Frequency (GDSF_K).** Esta política es una versión ligeramente modificada de **GDS.** Al igual que en GDS cada consulta cuenta con un valor *"H – value"* asociado, pero al cual se le agrega un ***k***. La nueva fórmula es la siguiente:

.

$$H - value(q) = \frac{F_q K \times C_q}{S_q} + L$$

Sigue la misma idea que GDS, pero se agrega un K (> 1) para privilegiar las consultas con frecuencias más altas.

.

6. EVALUACIÓN EXPERIMENTAL

6.1 Clase de consultas

Este trabajo busca diseñar una política de admisión. Para esto primero se toma un log de consultas reales el cual se usa para la experimentación.

Como primera parte de la experimentación, se usa el log de consultas para sacar la cantidad de términos que poseen las distintas consultas contenidas en él. Para determinar el número de términos que poseen las distintas consultas, se usa un algoritmo que va guardando en una estructura de datos, la cantidad de términos que va encontrando. A la cantidad de términos que posee una consulta se le llama “CLASE”. Así por ejemplo, una consulta que tenga un término, como *“Facebook”,* será una consulta de Clase 1 o abreviado C1.

Para representar las distintas clases de consultas, se consideran solo las consultas que tengan hasta diez términos, debido a que después la cantidad de consultas con una cantidad mayor a diez términos es insignificante y por lo tanto no es relevante.

El log de consultas usado para este experimento, corresponde a tres intervalos de tiempo distintos de un motor de búsqueda comercial, cuyos datos son públicos.

En la ilustración 6 se presenta un gráfico que muestra la cantidad de consultas de cada clase, hecho a partir de la tabla 1, la que contiene el porcentaje de consultas por clase del log usado para el experimento.

.

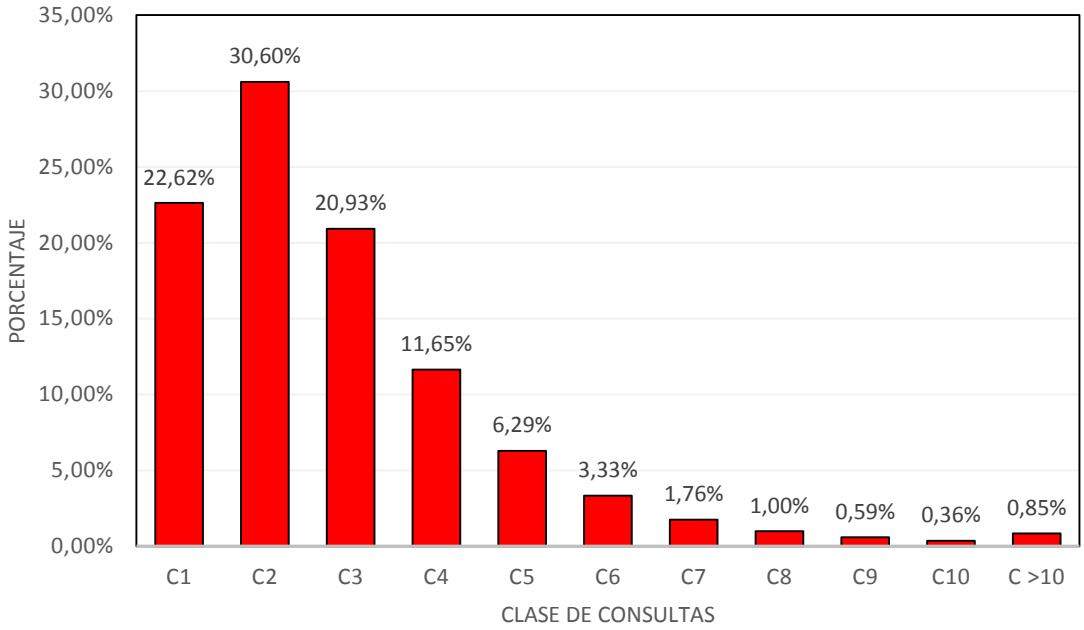


Ilustración 6: Gráfico clase de consultas – Fuente: Elaboración propia

En el gráfico de la ilustración 6 se observa que las consultas se concentran en las clases 1, 2, 3, 4 y 5. Luego de la clase 5, comienzan a disminuir considerablemente en relación a las 5 primeras clases.

Log 201104		
Clases de consultas	Porcentaje	Acumulado
C1	22,62%	22,62%
C2	30,60%	53,22%
C3	20,93%	74,15%
C4	11,65%	85,8%
C5	6,29%	92,09%
C6	3,33%	95,42%
C7	1,76%	97,18%
C8	1,00%	98,18%
C9	0,59%	98,77%
C10	0,36%	99,13%
C > 10	0,85%	100%

Tabla 1: Cantidad de consultas según la clase - Fuente: Elaboración propia

.

Las clases de la tabla 1 van desde C1 hasta C10 con su respectivo porcentaje de esa clase respecto al total de la muestra. También se incluye el porcentaje de las clases superior a C10, la cual no supera el 1% de la muestra. Como se puede ver en la tabla 1, las 3 primeras clases abarcan más del 74% de la muestra y considerando las 5 primeras se abarca más del 92% de la muestra lo que indica que las consultas más realizadas por los usuarios son las 5 primeras. Pero de estas 5 primeras clases las 3 primeras son las más importantes de mantener en cache.

Una vez hecha la primera parte de la evaluación experimental y ya clasificada las consultas según su cantidad de términos, se procede a determinar la cantidad de caracteres que posee cada una de estas clases de consultas para tener más detalles e información de las consultas. Para esto, se genera un gráfico representativo de cada clase de consultas. Se observa en el gráfico de la ilustración 6, que los usuarios tienen cierta tendencia a hacer consultas cortas, o sea de clases C1, C2 y C3, ya que entre las 3 clase abarcan el 70% de consultas hechas por los usuarios. Luego se ve una caída importante en C4 de casi un 50% con respecto a C3 lo cual se repite de C5 en adelante. Se observa que las clases con más de 10 términos no son algo relevante, ya que no alcanzan a ser el 1% de toda la muestra.

6.2 Cantidad Caracteres por Clase de Consultas

Para esta segunda parte, cada clase de consulta definida anteriormente en la sección 6.1, es decir, las clases de C1 hasta C10, se les determina la cantidad de caracteres que poseen a cada clase por separado. Cada consulta dependiendo de su clase puede tener términos con distinta cantidad de caracteres. Por ejemplo, para consultas de la clase uno (C1), estas pueden tener una letra solamente “q”, “s”, “a”, o también tener consultas de más caracteres, como puede ser “Casa”, “Perro”, etc. Todas estas consultas son de la misma clase C1. Entonces, como se puede observar cada clase de consulta puede tener

.
distintas consultas con distinta cantidad de caracteres. Por esto que se procede a graficar cada clase de consultas según la cantidad de caracteres que poseen las consultas de las clases y así determinar si existen un patrón entre las consultas y su cantidad de caracteres.

Con la cantidad de caracteres extraídos de cada clase de consulta se hace un gráfico. Cada gráfico tiene la cantidad de caracteres y la frecuencia con que sale la consulta con dicha cantidad de caracteres. Los gráficos mostrados a continuación corresponden al primer mes de datos de los 3 analizados. Este log se usa de acá en adelante debido a la similitud que existen con los otros 2. Los gráficos correspondientes a los otros dos meses se pueden ver en los anexos C, como también los gráficos de las consultas 6 hasta la 10 de este mes.

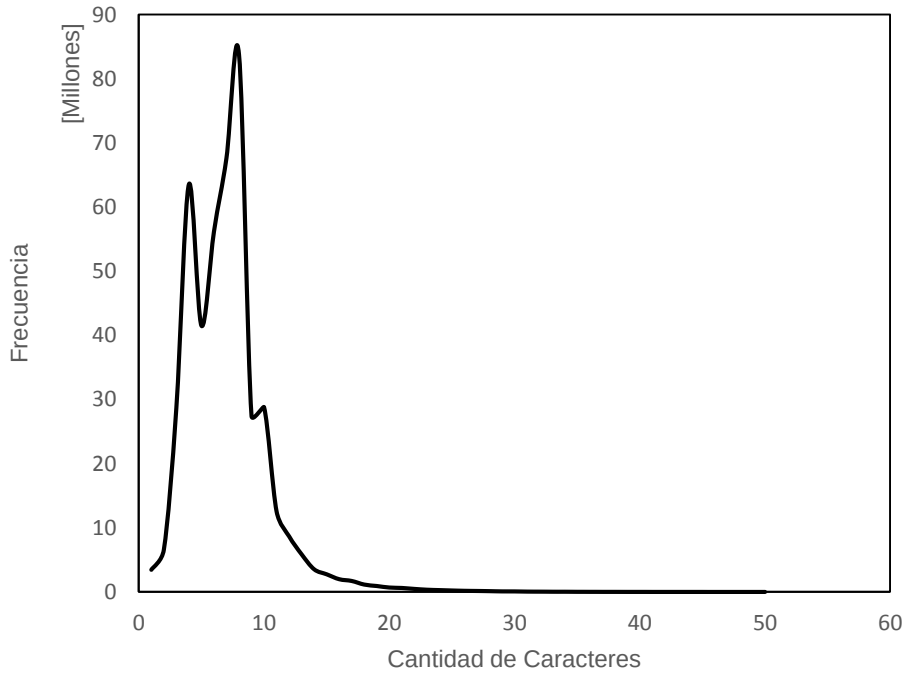


Ilustración 7: Clase 1 de consultas – Fuente: Elaboración propia

En la ilustración 7 se observa el gráfico correspondiente a la clase 1 de consultas. Este gráfico es el resultado de la cantidad de caracteres que poseen las distintas consultas con 1 término y su respectiva frecuencia. Es claro ver como

.
la cantidad de caracteres de la consulta comienza a subir hasta llegar a un pico y luego decae hasta llegar a cero. Se pueden observar ciertas variaciones. A pesar de esto, se puede extraer un dato interesante del gráfico de la ilustración 7. El promedio de caracteres de la consulta de 1 término es de aproximadamente siete caracteres. Esto es importante destacar, ya que quiere decir que de todas las consultas que se hacen con un término, la mayoría de estas tiene siete caracteres.

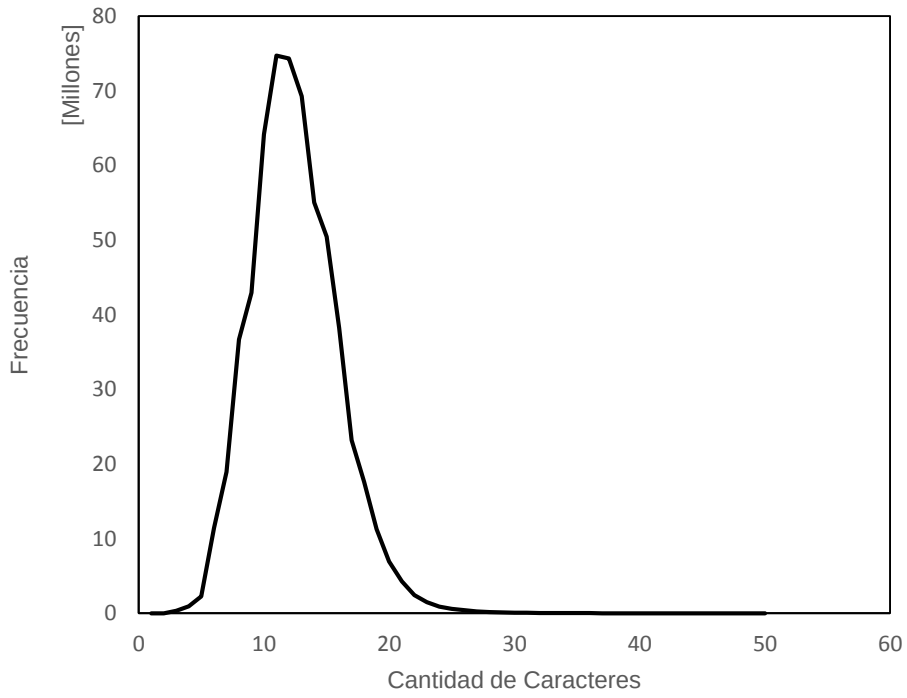


Ilustración 8: Clase 2 de consultas – Fuente: Elaboración propia

En la ilustración 8 se muestra la curva para la clase 2 de consultas. Se observa que a diferencia del gráfico de la clase 1 de consultas, el promedio se desplazó. Es decir, el número de caracteres promedio que posee una consulta de dos términos es de aproximadamente doce caracteres. Esto es importante porque nuevamente se ve un patrón, es decir de todas las consultas de dos

.
términos que las personas realizan a diario en un buscador, en promedio posee doce caracteres.

En la ilustración 7 se observa que la curva tiene variaciones, pero en cambio la ilustración 8 da una curva que se asemeja a una campana de gauss.

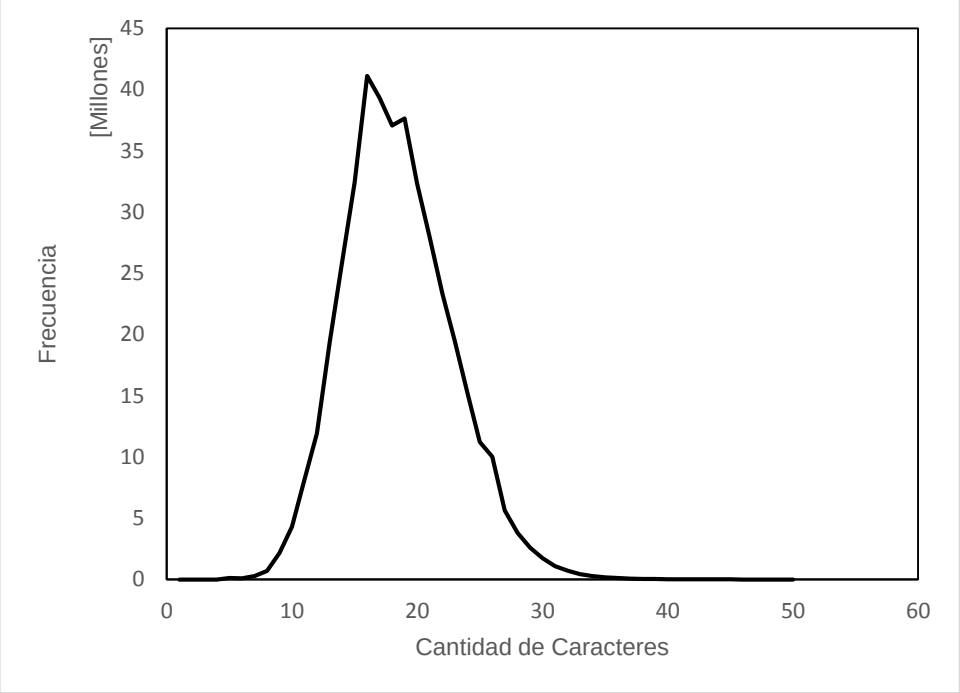


Ilustración 9: Clase 3 de consultas – Fuente: Elaboración propia

En la ilustración 9 correspondiente a la clase 3 de consultas se muestra como el promedio de caracteres por consultas aumenta nuevamente. Ahora se ubica entre los dieciocho y veinte caracteres, y la frecuencia comienza a bajar. La curva resultante nuevamente se asemeja a una campana de gauss.

.

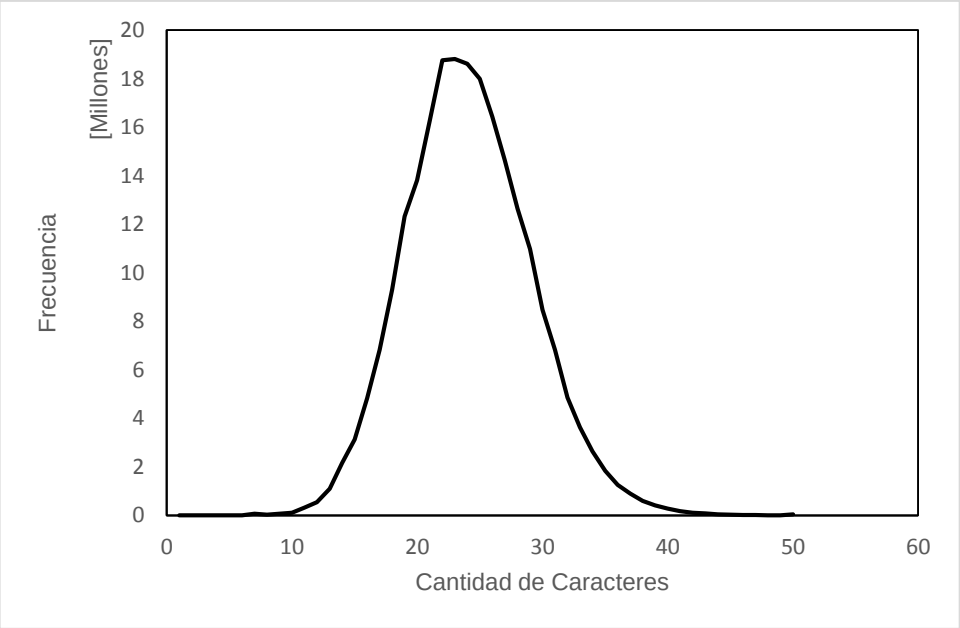


Ilustración 10: Clase 4 de consultas - Fuente: Elaboración propia

La ilustración 10 para la clase 4 de consultas muestra nuevamente como la curva resultante del gráfico se asemeja a una campana de gauss. El promedio vuelve a aumentar, está entre veintitrés y veinticinco caracteres.

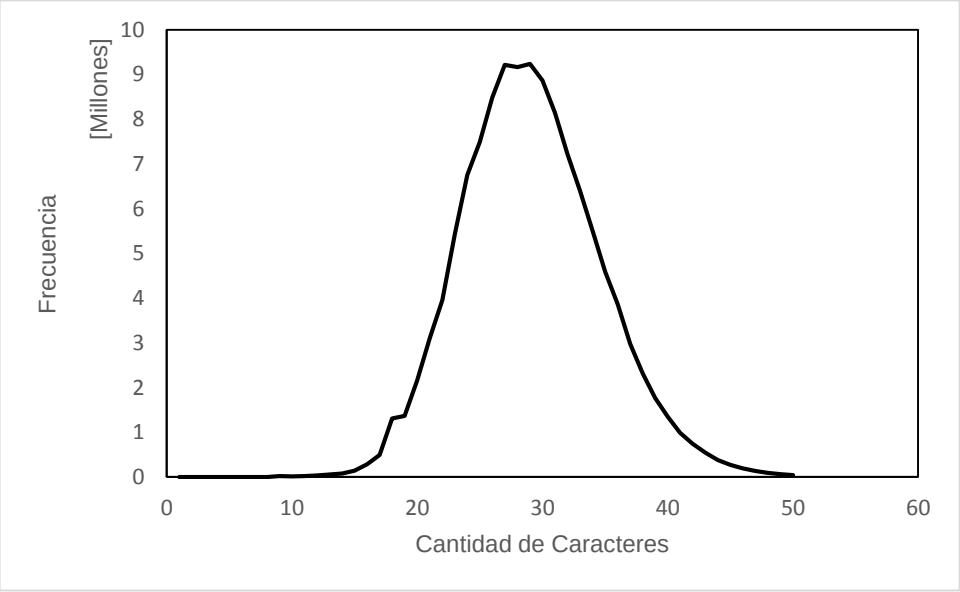


Ilustración 11: Clase 5 de consultas - Fuente: Elaboración propia

.

La ilustración 11 para la clase 5 de consultas vuelve a mostrar el patrón visto en los anteriores gráficos. La curva resultante es semejante a una campana de gauss. En el gráfico mostrado en la ilustración 7 no se ve claramente la campana de gauss, pero en los siguientes cuatro gráficos esto es una constante. Este patrón visto en las curvas de los gráficos, continúa repitiéndose en las consultas de seis hasta diez términos y en los tres meses de consultas usados para este trabajo.

Lo mismo ocurre para el promedio de caracteres de cada consulta, este continua subiendo linealmente, mientras que la frecuencia del promedio de caracteres de cada consulta también disminuye a medida que la cantidad de términos de la consulta aumenta.

Los gráficos de Clase 6 hasta la 10, están en el Anexo C.

De los gráficos anteriores es importante destacar lo siguiente:

- La mayoría de los gráficos muestran una curva similar a una campana de gauss lo cual es importante para la continuación del experimento.
- La frecuencia de la cantidad de caracteres de las consultas decae a medida que aumenta la cantidad de términos de la consulta.
- El promedio de la cantidad de caracteres aumenta a medida que crece la cantidad de términos de la consulta (se ve como la curva se va desplazando). Por ejemplo el gráfico de la clase 1 tiene un promedio aproximado de 6 caracteres, mientras que el de clase 2 tiene un promedio de 11 caracteres. Esto sigue el mismo patrón en las siguientes clases de consultas.

.

6.3 Ajuste de curvas

Ahora se procede a ajustar las curvas obtenidas en la segunda parte de la experimentación. Para esto se usa una herramienta llamada Scilab (Scilab Enterprises, s.f.), el cual sirve para hacer análisis numérico, visualización 2D y 3D, optimización, análisis estadístico, entre otras.

Como se vio en los gráficos de la sección 6.2, todos tenían una curva semejante a una campana de gauss, con la excepción de la ilustración 7.

Para seguir con la experimentación, se debe ajustar todas las curvas obtenidas a una campana de gauss. Con este ajuste se quieren obtener los parámetros necesarios para diseñar una política de admisión basada en percentiles, que es una propiedad de la distribución gaussiana.

El objetivo de este trabajo se observa en la ilustración 12. Es decir ajustar las curvas a una campana de gauss y así admitir cierto porcentaje de consultas en función de su promedio y desviación estándar.

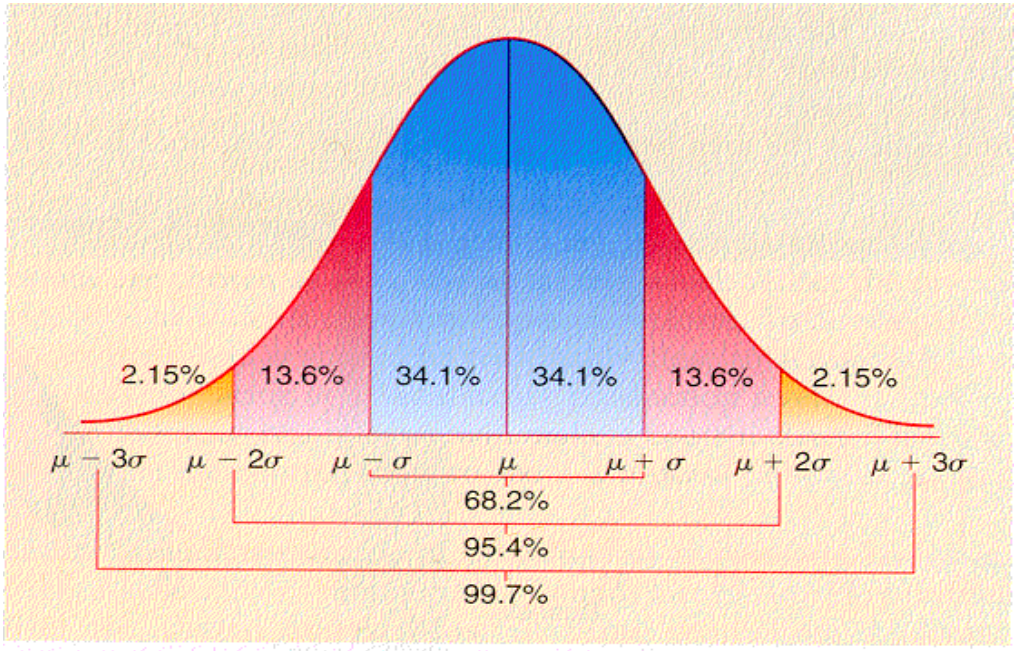


Ilustración 12: Distribución Gaussiana – Fuente: (CHEN, 2012)

.

Al ajustar las curvas a una campana gaussiana, es posible definir qué porcentaje de las consultas de cada clase se admite en cache, esto tomando en cuenta la cantidad de caracteres que posee cada consulta. Es posible definir una admisión de un 100% e ir reduciéndola esta hasta un intervalo que maximice la tasa de hit. Este intervalo es distinto según la cantidad de términos de una consulta, es decir no se puede usar los mismos rangos para consultas de uno, dos, tres términos que una que posea más de seis términos. Esto porque en la ilustración 6, se observa que la mayor cantidad de consultas hechas por los usuarios se concentra en las consultas de uno a cinco términos. Por lo tanto no es recomendable definir un rango pequeño para el intervalo de consultas con mayor frecuencia, ya que produciría un efecto contrario al esperado, es decir, se reduciría la tasa de hit.

En la ilustración 12 se observa que el promedio más una desviación estándar y el promedio menos una desviación estándar ($\bar{X} - \sigma$ y $\bar{X} + \sigma$) se cubre aproximadamente un 68% del total. Si aumenta esto a dos distribuciones estándar, se cubre aproximadamente un 95% del total. Para hacer esto más preciso, se utiliza una tabla de distribución la cual da los valores para ir reduciendo el rango de cinco en cinco por ciento y de esta forma definir los rangos con mayor precisión en cada clase.

Para realizar este ajuste en las curvas, es necesario identificar tres diferentes datos de las curvas sin ajuste: primero la amplitud de la curva, el promedio y finalmente la desviación estándar. Estos tres datos son requeridos por Scilab para hacer un ajuste de mínimos cuadrados. Este ajuste sirve para corregir estos datos a unos que permitan graficar una campana de gauss “ajustada”.

La tabla 1 muestra los datos usados en un principio para realizar el ajuste de las curvas en Scilab. Estos corresponden a los datos aproximados extraídos de cada clase de consulta. Estos datos son determinados manualmente ya que no se tiene conocimiento de estos de manera exacta. Esto es posible porque el

.
algoritmo luego los corrige usando la misma fuente de datos, que son los archivos con la etiqueta “1.dat”, “2.dat”, etc. El número indica la cantidad de términos de la consulta, es decir, el archivo “1.dat” contiene los datos de las consultas de 1 término, la “2.dat” la de dos términos y así sucesivamente hasta el diez.

Datos Aproximados			
Archivos	*a*	*b*	*c*
1.dat	83814501	7	1
2.dat	74713540	12	1
3.dat	41123237	19	1
4.dat	18818504	23	1
5.dat	9237017	29	1
6.dat	4604718	33	1
7.dat	2163725	37	1
8.dat	1148552	41	1
9.dat	633921	45	1
10.dat	356714	49	1

Tabla 2: Datos Aproximados - Ajuste de curva - Fuente: Elaboración propia

En la tabla 2 el primer dato “a” corresponde a la amplitud del gráfico, o sea es la frecuencia máxima encontrada en los datos. El dato “b” es el promedio de caracteres de cada clase de consulta. El archivo “1.dat” muestra un promedio de siete para la clase 1, mientras que “2.dat” el promedio es doce para la clase 2 y de igual forma los siguientes. “c” indica la desviación estándar de los datos. Este dato es desconocido en un principio y solo es puesto ya que el algoritmo de mínimos cuadrado lo requiere. Estos datos aproximados no alteran en nada el ajuste de curva ya que son nuevamente calculados y corregidos por el mismo algoritmo.

A partir de los datos del cuadro anterior se lanza el experimento en Scilab, dando como resultado los datos corregidos en la tabla 3.

.

Datos de Scilab			
Archivos	a	b	c
1.dat	68364420	6,583274	2,589116
2.dat	74713540	12,093928	3,239223
3.dat	39354193	17,944688	4,195118
4.dat	18818504	23,805156	4,896706
5.dat	9257812	28,681942	5,36112
6.dat	4525192,4	33,216527	5,776489
7.dat	2163725	37,660879	6,374314
8.dat	1143053,2	42,000843	6,863871
9.dat	624980,31	46,439178	7,391278
10.dat	353675,88	51,094065	7,935539

Tabla 3: Datos ajustados – Fuente: Elaboración propia

Observando las dos tablas, es posible ver que tanto la amplitud como el promedio son similares a pesar de que los primeros corresponden a datos aproximados usados para el experimento. Lo que cambia es la desviación estándar, ya que en un comienzo no se tiene información de este dato en particular. Este dato es corregido en los datos de la segunda tabla.

.

6.4 Curvas ajustadas

En esta parte de la experimentación se procede a graficar nuevamente, pero ahora haciendo uso de los datos obtenidos mediante el ajuste hecho con Scilab. Estos datos corresponden a la Tabla 3 en la sección 6.3.

A continuación se presentan los gráficos obtenidos. Cada gráfico muestra dos “líneas”. Primero las cruces representan la curva original obtenida sin el ajuste, mientras que la línea muestra la curva corregida.

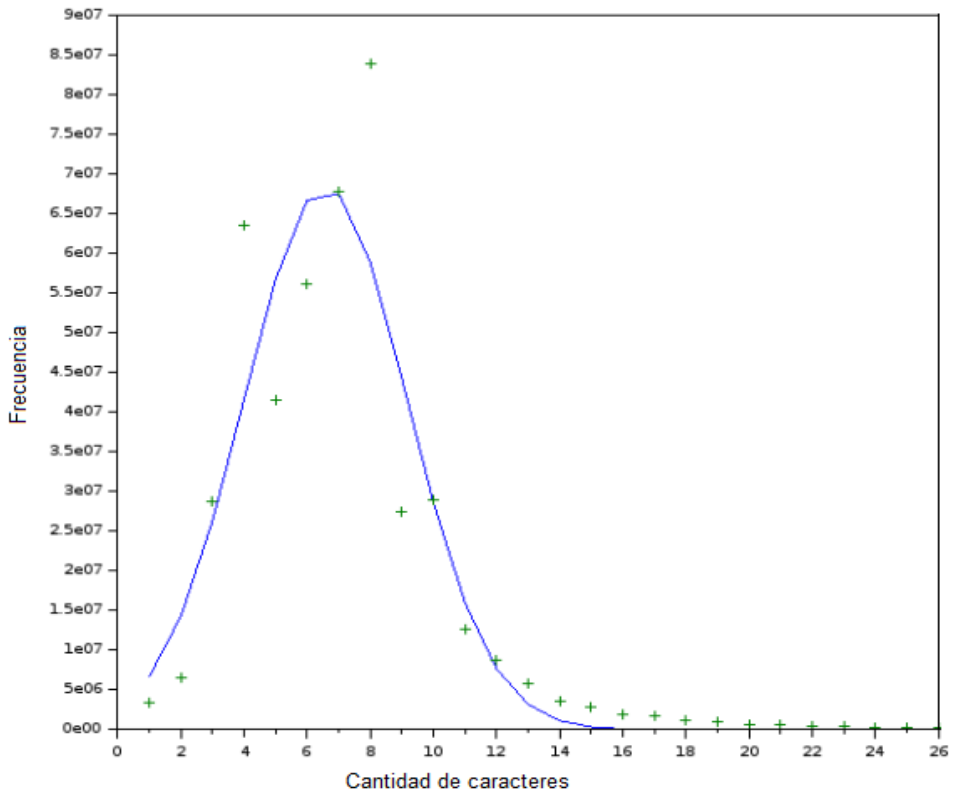


Ilustración 13: Clase 1 de consultas – corregida - Fuente: Elaboración propia

En la ilustración 13 se observa como el ajuste de la curva hizo que el gráfico de la clase 1 de consultas se ajuste a una distribución gaussiana, pese a

.
ser el gráfico que menos se asemeja a una distribución gaussiana. En la ilustración 13 se observan las cruces que representan la curva original (ilustración 7), también se observa como al corregir el gráfico, el algoritmo deja fuera a los datos que se escapan de la media.

La desviación estándar es corregida en todos los gráficos. En los datos originales no se tiene el valor de la desviación estándar por lo que se asigna un valor inicial**,** ya que este dato es requerido. En los datos resultantes, la desviación estándar es corregida y varía en todos los datos. La desviación estándar aumenta a medida que aumenta la cantidad de términos en la consultas.

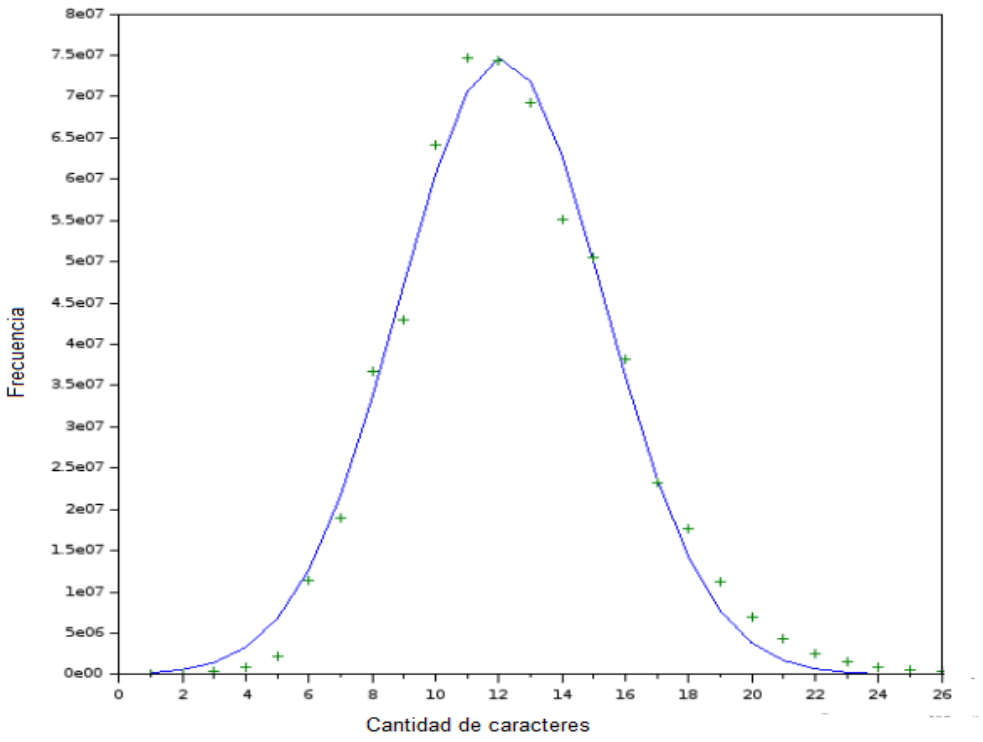


Ilustración 14: Clase 2 de consultas – corregida - Fuente: Elaboración propia

En el gráfico de la clase 2 de consulta (ilustración 14), la curva original es semejante a una distribución gaussiana sin ningún tipo de corrección de datos.

.
Sin embargo, a pesar de ser ya semejante a una distribución gaussiana se tiene que aplicar el mismo método para la obtención de los datos. Esta semejanza se puede ver claramente ya que al aplicar la corrección con Scilab los datos varían poco. Esto se puede ver en las tabas 2 y 3 de la sección 6.3, en el archivo “2.dat”.

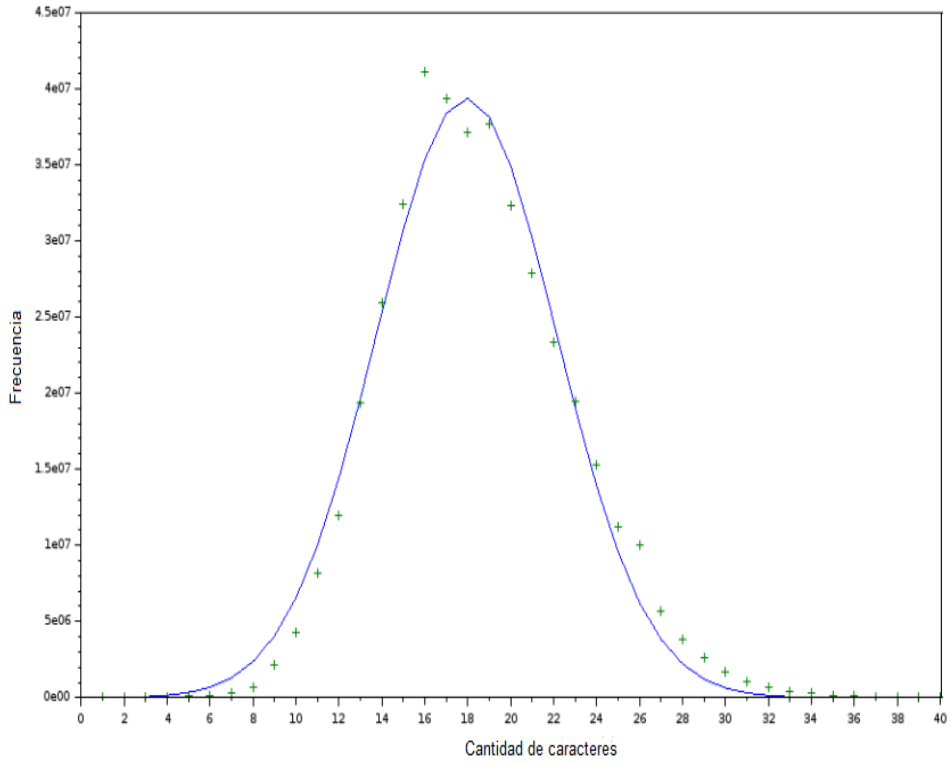


Ilustración 15: Clase 3 de consultas – corregida - Fuente: Elaboración propia

En la ilustración 15 se observa como el promedio de caracteres aumenta al igual que en los gráficos anteriores. Otra observación importante es que nuevamente se ve como el algoritmo, para el ajuste de la curva, deja fuera los datos que se escapan. Esto se ve reflejado en las cruces verdes en el pico del gráfico.

.

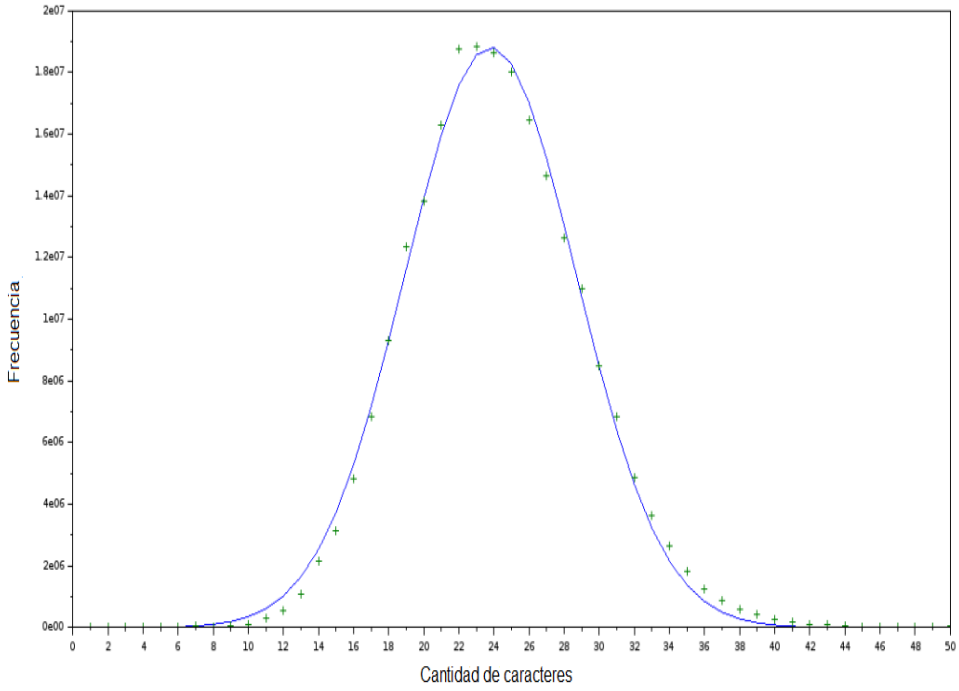


Ilustración 16: Clase 4 de consultas – corregida - Fuente: Elaboración propia

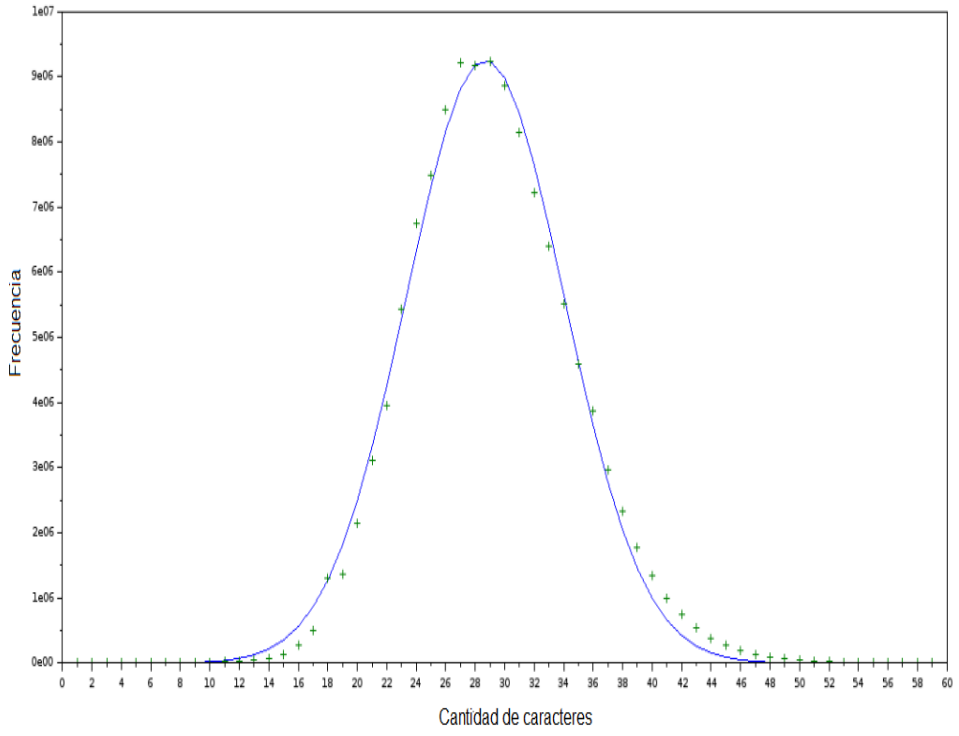


Ilustración 17: Clase 5 de consultas – corregida - Fuente: Elaboración propia

.

La ilustración 16 y 17 se observa nuevamente el mismo patrón observado en los gráficos anteriores. El promedio de caracteres sube tanto en la ilustración 16 y 17. En la ilustración 16 el promedio es de aproximadamente **24** caracteres, mientras que en la ilustración 17 el promedio de caracteres aumenta a **28** caracteres aproximadamente. Estos promedios no son distintos a lo que se ve en la tabla 2 que contiene los datos originales y esto es debido a que la curva original ya es semejante a una campana gaussiana.

Los gráficos corregidos desde la clase 6 a las 10, como también los correspondientes a los otros 2 meses están en el Anexo D.

.

6.5 Análisis de datos corregidos

Una vez finalizado el experimento del ajuste de curvas, se sigue con el análisis de los datos obtenidos. Luego de ver que el ajuste hecho a los gráficos originales de la sección 6.2, logra corregir todos los gráficos a una distribución gaussiana. Con este ajuste se obtienen los datos corregidos de estos gráficos. Los datos son los de la tabla 3 de la sección 6.3.

Para realizar el análisis de los datos, se hace uso de métricas, las cuales ayudan a entender de mejor manera cómo impactan las políticas de admisión usadas en este trabajo.

- **Tasa de Hit:** La tasa de hit es la media utilizada para saber cuándo una consulta es encontrada en cache. Esta métrica es usada en este trabajo para ver cómo impactan las políticas de admisión al cache.
- **Tasa de Hit total o global:** Es la misma tasa de hit, pero esta para medir el hit de todas las clases en conjunto.
- **Tasa de hit por clase de consulta:** Es la misma tasa de hit, pero mide cuánto es la tasa de hit por cada clase de consulta. Esto para ver en qué clase de consulta impacta más la política de admisión.
- **Accesos por clase de consultas:** Los accesos por clase de consultas. Es una medida para saber cuántas consultas ingresan de cada tipo de consultas de 1 a 10, como fue explicado en la sección 6.1.
- **Acceso por rango de admisión:** Esto mide cuántas consultas acceden a cache, según el porcentaje de restricción puesto a la clase de consulta. Tiene que ver con la política de admisión puesta a la clase de consulta. Esto se explica en la sección 6.3.
- **Desalojo total:** Esta indica cuál es la cantidad de consultas que el algoritmo de reemplazo saca de cache cuando está lleno.

.

- **Desalojo por clase de consultas:** Se usa para medir cuántas consultas según su clase se desalojan del cache cuando este está lleno. Esto es para comprobar qué clases de consultas son más desalojadas del cache.

Utilizando las métricas anteriormente definidas, se lanza el experimento con los datos corregidos. La idea de este experimento es ver cómo afecta la política de admisión a cada clase de consultas. Para esto se aplican percentiles de admisión a cada clase, los percentiles van desde un 100% hasta llegar a un 50% de admisión, con saltos de 5%. Estos saltos son para ver cómo impacta cada vez que se reduce la admisión de consultas, el llegar al 50% es porque como se explica en la sección 6.3, con ese porcentaje de admisión se está dejando ingresar las consultas que están dentro del promedio de caracteres que se consultan con mayor frecuencia, es decir, las consultas que se encuentran en la parte de la campana de gauss que se tiene una mayor frecuencia. El experimento es para cada clase de consulta individualmente, es decir, no se combinan porcentaje de restricción. Son evaluadas todas con 100%, luego todas con 95%, así hasta llegar al 50%. Es por esto que se ve una disminución del hit en la clase de consultas cuando se reduce la admisión. El tamaño de cache usado para este experimento es de 10.000 entradas y 100.000 entradas.

A continuación se presenta los rangos de admisión con sus respectivas cantidades de caracteres.

.

Rango de caracteres											
Clase de consultas	Porcentaje de admisión										
	100%	95%	90%	85%	80%	75%	70%	65%	60%	55%	50%
C1	na	2 - 11	2 - 11	3 - 10	3 - 10	4 - 9	4 - 9	4 - 9	4 - 9	5 -8	5 - 8
C2	na	6 - 18	7 - 17	7 - 17	8 - 16	8 - 16	9 - 15	9 - 15	9 - 15	10 - 14	10 - 14
C3	na	10 - 26	11 - 25	12 - 24	13 - 23	13 - 23	14 - 22	14 - 22	14 - 22	15 - 21	15 - 21
C4	na	14 - 33	16 - 32	17 - 31	18 - 30	18 - 29	19 - 29	19 - 28	20 - 28	20 - 27	21 - 27
C5	na	18 - 39	20 - 37	21 - 36	22 - 35	23 - 35	23 - 34	24 - 34	24 - 33	25 - 33	25 - 32
C6	na	22 - 44	24 - 43	25 - 41	26 - 40	27 - 40	27 - 39	28 - 39	28 - 38	29 - 37	29 - 37
C7	na	25 - 50	27 - 48	29 - 47	30 - 46	30 - 45	31 - 44	32 - 43	32 - 43	33 - 42	33 - 42
C8	na	29 - 55	31 - 53	32 - 52	33 - 51	34 - 50	35 - 49	36 - 48	36 - 48	37 - 47	37 - 47
C9	na	32 - 61	34 - 58	36 - 57	37 - 56	38 - 55	39 - 54	40 - 53	40 - 53	41 - 52	41 - 51
C10	na	36 - 66	38 - 64	40 - 62	41 - 61	42 - 60	43 - 59	44 - 58	44 - 58	45 - 57	46 - 56

Tabla 4: Rango de caracteres - Fuente: Elaboración propia

La tabla 4 muestra la cantidad de caracteres aplicado en cada porcentaje de admisión para cada clase. En ciertos casos se repiten los rangos, lo cual se explica porque al calcular estos dos valores dan resultados en decimal y como una consulta no puede tener un tamaño decimal este se aproxima.

La siguiente tabla se muestra la frecuencia de las consultas que se deja entrar en cada rango de caracteres mostrados en la tabla 4.

.

Frecuencia por rango											
Clase de consultas	Porcentaje de admisión										
	100%	95%	90%	85%	80%	75%	70%	65%	60%	55%	50%
C1	450714842	416865310	416865310	397717014	397717014	340185021	340185021	340185021	340185021	249187524	249187524
C2	609724807	576914471	547893463	547893463	505798266	505798266	430890865	430890865	430890865	337498024	337498024
C3	417063692	396828620	382495338	363131274	335918220	335918220	297077066	297077066	297077066	247778102	247778102
C4	232100920	221368937	212433027	202746305	189102742	180625412	171317950	160339002	148014001	135377932	121558026
C5	125363280	119319377	112555765	107430838	100454185	96495627	91912722	86483335	80969237	74214626	67811338
C6	66366265	62285240	60053786	56140354	52728908	50670820	48435651	45848804	43223573	37144586	37144586
C7	35060279	32977914	31558884	29877152	28279771	27350034	25211446	22644821	22644821	19666235	19666235
C8	20019006	18642103	17802846	17184096	16408606	15448345	14299572	12935691	12935691	11377718	11377718
C9	11830131	11048344	10468612	10004228	9556285	9012249	8366275	7615201	7615201	6761874	6326619
C10	7225037	6642557	6385044	5993156	5733432	5421613	5053581	4631525	4631525	4150249	3613795

Tabla 5: Frecuencia por rango de admisión - Fuente: Elaboración propia

.

A continuación se presentan las tablas que resumen el conjunto de resultados con la tasa de hit por clase de consulta. Esto para ver las clases que concentran el hit.

Tasa de hit por clase de consulta					
Clases de consultas	Porcentaje de admisión				
	100%	95%	90%	85%	80%
C1	60,60%	59,01%	59,01%	57,46%	57,46%
C2	30,25%	29,91%	28,62%	28,62%	26,78%
C3	11,42%	11,24%	10,63%	10,25%	9,60%
C4	3,47%	3,40%	3,34%	3,18%	2,88%
C5	1,56%	1,54%	1,40%	1,35%	1,31%
C6	0,93%	0,90%	0,88%	0,86%	0,84%
C7	0,69%	0,48%	0,46%	0,44%	0,43%
C8	0,44%	0,41%	0,39%	0,38%	0,36%
C9	0,47%	0,43%	0,41%	0,39%	0,38%
C10	0,49%	0,45%	0,43%	0,41%	0,39%

Tabla 6: Tasa de hit por clase de consulta 10.000 entradas – Parte 1 - Fuente: Elaboración propia

Tasa de hit por clase de consulta						
Clases de consultas	Porcentaje de admisión					
	75%	70%	65%	60%	55%	50%
C1	50,22%	50,22%	50,22%	50,22%	38,44%	38,44%
C2	26,78%	22,87%	22,87%	22,87%	18,50%	18,50%
C3	9,60%	8,65%	8,65%	8,65%	7,35%	7,35%
C4	2,84%	2,55%	2,20%	1,77%	1,72%	1,62%
C5	1,28%	1,27%	1,12%	1,10%	0,73%	0,71%
C6	0,82%	0,81%	0,78%	0,77%	0,73%	0,73%
C7	0,42%	0,39%	0,37%	0,37%	0,33%	0,33%
C8	0,34%	0,32%	0,29%	0,29%	0,26%	0,26%
C9	0,36%	0,34%	0,31%	0,31%	0,29%	0,27%
C10	0,38%	0,36%	0,33%	0,33%	0,31%	0,28%

Tabla 7: Tasa de hit por clase de consulta 10.000 entradas – Parte 2 - Fuente: Elaboración propia

.

La tablas 6 y 7 muestran cómo la tasa de hit decae a medida que la cantidad de términos que tiene la consulta aumenta. Esto tanto para el cache de 10.000 entradas como para el de 100.000. Los resultados de la tabla 6 corresponden a la admisión de 100% a 80%. Los cuales continúan en la tabla 7 hasta llegar al 50% de admisión. Estas tablas son el resultado de dividir el hit de la clase con las entradas totales para cada clase individualmente.

Se observa que la mayor cantidad de hit está en las dos primeras clases de consulta C1 y C2, luego en C3 ronda el 10% pero en las clases C4 y C5 es cercana a un 5%. Desde la clase C6 a C10 la tasa de hit es aproximadamente de un 1%. Lo importante de estas tablas es ver como el hit siempre se concentra en las tres primeras clases. Luego, en las demás clases el hit decae considerablemente. Esta concentración en el hit en las 3 primeras clases se sigue manteniendo a pesar de ir bajando la tasa de admisión. Las tablas 6 y 7 son los resultados para el cache de 10.000 entradas, los resultados para el cache de 100.000 entradas está en el Anexo B.

Ahora se analiza el impacto de la reducción de la admisión de consultas en el rango de la clase, es decir, cómo afecta la política de admisión a las distintas clases de consultas por separado. Esto permite analizar el impacto que tiene en el hit reducir la admisión de consultas. Para entender cómo afecta esto, se presenta la ilustración 18 la cual tiene las distintas clases de consultas con su respectivo porcentaje de admisión y el porcentaje de hit en eje y.

.

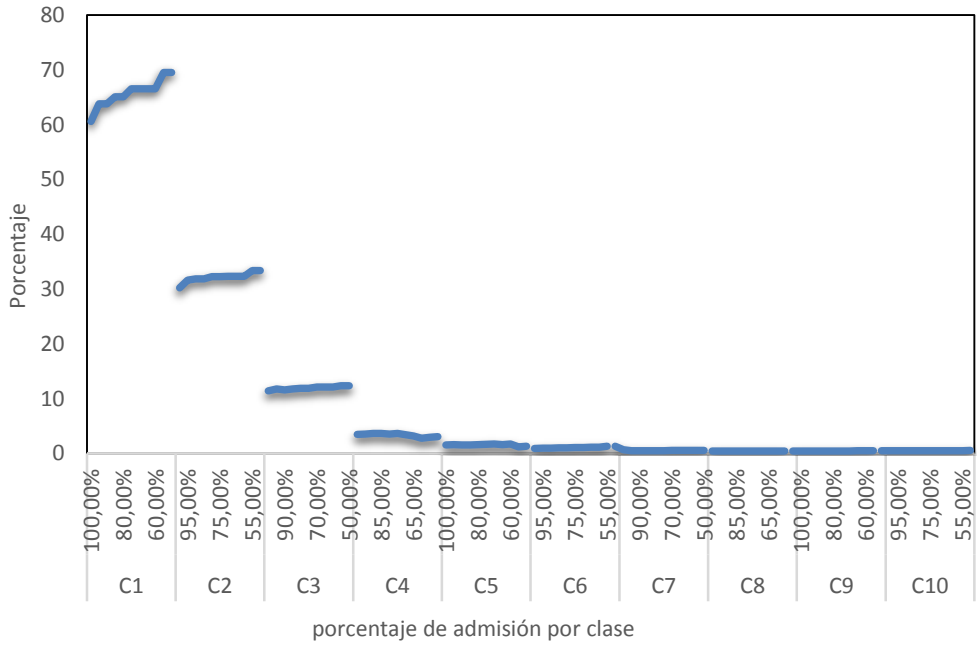


Ilustración 18: Tasa de Hit según rango de admisión por clase – 10.000 entradas - Fuente: Elaboración propia.

En la ilustración 18 correspondiente a la tasa de hit de la clases según su rango de admisión. Se observa cómo para cada clase de consultas se restringe la admisión desde un 100% hasta un 50%. Esto para ver cómo varía el hit de cada clase según el rango de admisión, el hit en el rango de admisión se calcula dividiendo el hit de la clase, pero entre el total de consultas que ingresaron en cada rango, es decir, la cantidad de consultas de que ingresaron con una admisión de 100%, luego en 95%, así hasta llegar al 50%. A continuación en la ilustración 19 se presenta el mismo gráfico pero correspondiente al cache con 100.000 entradas.

.

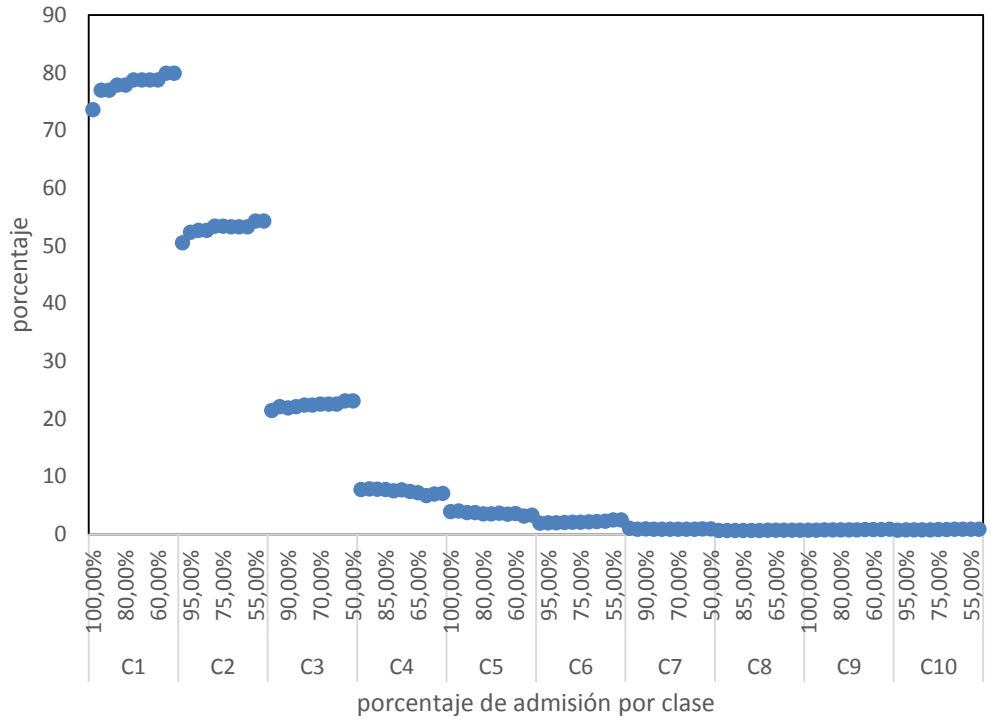


Ilustración 19: Tasa de Hit según rango de admisión por clase – 100.000 entradas - Fuente: Elaboración propia

En la ilustración 19 se observa un gráfico similar al de la ilustración 18, con variaciones que se explican al mayor espacio disponible en un cache con 100.000 entradas.

De las diez clases de consultas estudiadas se observa que solo C1, C2 y C3 presentan un comportamiento creciente a medida que se restringe la tasa de admisión tanto para un cache de tamaño de 10.000 entradas y uno de 100.000 entradas. Por ejemplo la clase C1 de consultas aumenta el hit desde un 60% aproximadamente hasta un 70% prácticamente en un cache de 10.000 entradas mientras que para el cache de 100.000 entradas, aumenta el hit desde 73% hasta un 79%. Lo mismo ocurre con la clase C2, que aumenta el hit de la clase desde 30% hasta un 32% para un cache de 10.000 entradas y desde un 51% hasta un 54% para un cache de 100.000 entradas. Finalmente con C3 se tiene un aumento desde 11% hasta un 12% aproximadamente para un cache de 10.000 entradas

.
y para un cache de 100.000 entradas tenemos un aumento del hit desde 22% a un 23%.

La clase C1 tiene un aumento considerable de la tasa de hit de aproximadamente un 10% para un cache de 10.000 entradas y un 6% para un cache de 100.000 entradas. Para C2 el aumento es de un 2% para un cache de 10.000 entradas y un 4% para uno de 100.000 entradas. En C3 ya se nota una disminución de ganancia de hit, en un cache de 10.000 entradas es de 1% mientras que para el de 100.000 entradas es de un 2%. En las clases C4 y C5 se observa un comportamiento casi constante en el hit. En la clase C4 más en detalle se ve una pérdida de hit, cuando la admisión es de 65%, y un aumento de un 0,5% cuando la admisión es de 50% para un cache de 10.000 entradas, para el cache de 100.000 ocurre lo mismo. Para la clase C5 la pérdida es de un 0,4% aproximadamente para el cache de 10.000 entradas, mientras para el de 100.000 hay una pérdida de hit de un 0,7% aproximadamente.

Para las clases C6 a C10, las tasas de hit son menores en comparación a las 5 primeras clases, teniendo cada una un hit de aproximadamente 1% a 2%. Esto indica que no son relevantes y se puede excluir la mayoría de entrar a cache. Es decir, no se deben dejar ingresar el 100% de las consultas, sino que es posible ser estrictos en la admisión, en comparación a las 5 primeras clases.

La ilustración 20 y 21, muestran el desalojo de las clases de consultas y cómo afecta a este desalojo la reducción de la admisión de las distintas clases de consultas.

.

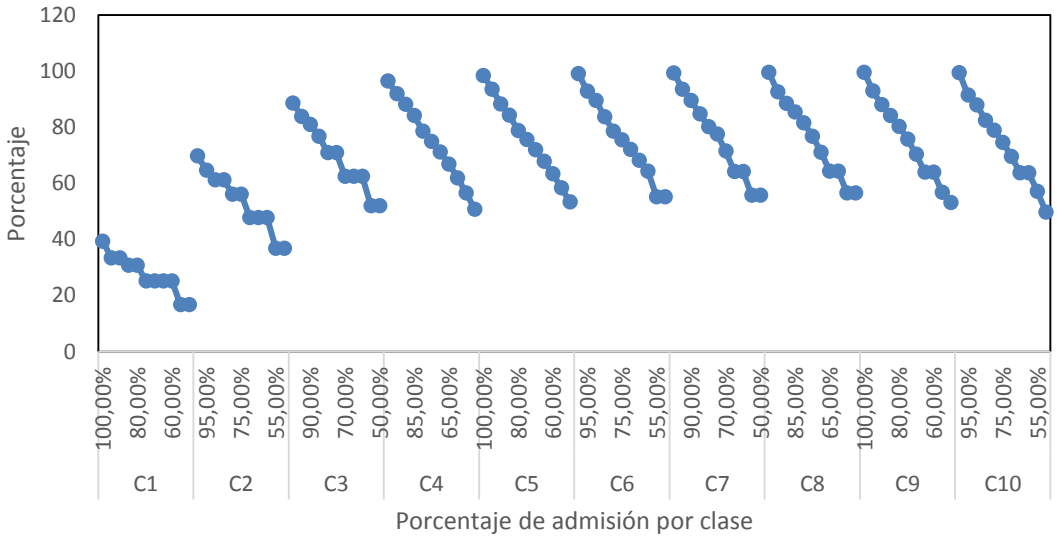


Ilustración 20: Desalojo de consultas por clase-10.000 entradas - Fuente: Elaboración propia.

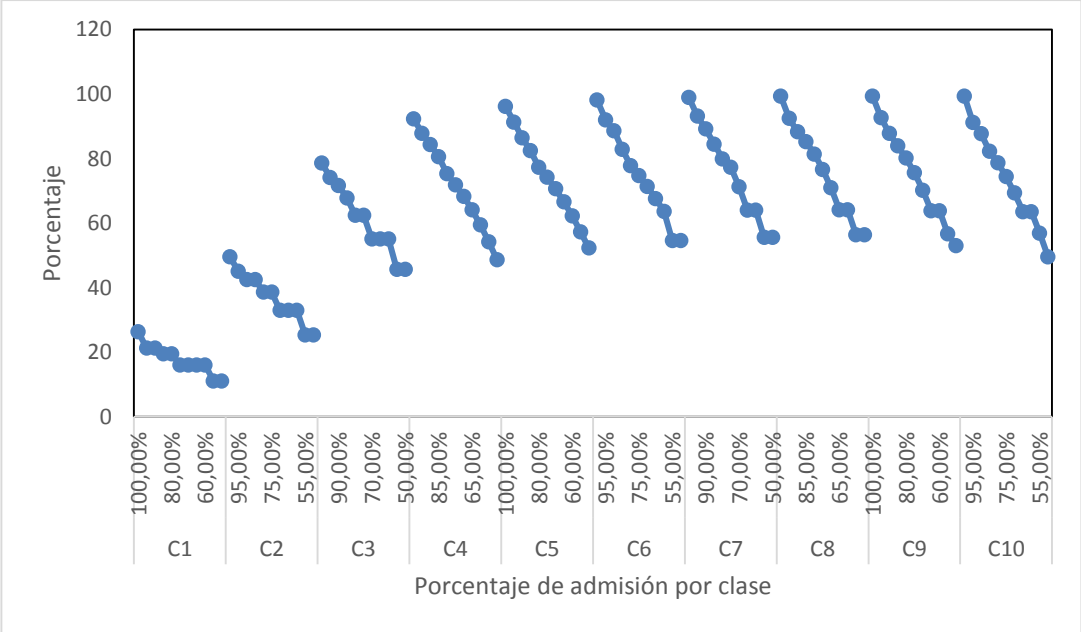


Ilustración 21: Desalojo de consultas por clase-100.000 entradas - Fuente: Elaboración propia.

En el gráfico de la ilustración 20 y 21, las clases de consultas C1, C2 y C3 son las que más bajo desalojo presentan, teniendo cada una un 26%, 45% y un 78% respectivamente de desalojo cuando se admite el 100% de las consultas. Desde la clase C4 hasta C10 el desalojo al admitir el 100% de las consultas es cercano al 99%, lo cual indica que al dejar entrar el 100% de las consultas a

.
cache y pertenezcan a una de estas 7 clases es prácticamente un hecho que saldrá de cache. Esto es importante ya que confirma lo observado en los dos gráficos anteriores en esta sección: las 3 clases que más hit abarcan son las clases C1, C2 y C3, mientras que las 7 restantes son poco relevante para el cache siendo más notorio las 5 clases finales (C6, C7, C8, C9, C10).

Esto es algo común tanto para el cache de 10.000 entradas y el de 100.000 entradas, variando mínimamente. Esto es importante ya que el cache de 100.000 entradas es 10 veces más grande, sin embargo las variaciones no son significativos. Por lo tanto las políticas de admisión tienen un mayor impacto en un cache más pequeño como es el de 10.000 entradas.

Un aspecto importante es que a medida que se comienza a restringir la admisión de las clases de consultas, el desalojo empieza a bajar. Esto es normal ya que hay menos consultas que desalojar de cada clase. Esto es un punto común en las 10 clases.

A continuación se presenta la tabla 8, donde se observa la tasa de hit **total** obtenido cuando se filtra solo 1 clases de consultas a la vez. Esta tabla, a diferencia de las tablas 6 y 7, es que estas últimas tienen el hit por clase de consulta.

Porcentaje de hit Total											
Clase de consultas que varía	Porcentaje de admisión										
	100%	95%	90%	85%	80%	75%	70%	65%	60%	55%	50%
c1	25,99	25,72	25,72	25,41	25,41	23,99	23,99	23,99	23,99	21,57	21,57
c2	25,99	25,87	25,53	25,53	25,00	25,00	23,92	23,92	23,92	22,72	22,72
c3	25,99	25,97	25,85	25,80	25,74	25,74	25,63	25,63	25,63	25,45	25,45
c4	25,99	25,99	26,00	25,99	25,97	25,98	25,97	25,94	25,91	25,92	25,94
c5	25,99	26,00	25,99	26,00	26,00	26,00	26,01	26,00	25,99	25,98	25,97
c6	25,99	25,99	26,00	26,00	26,00	26,01	26,01	26,01	26,01	26,02	26,02
c7	25,99	25,99	25,99	25,99	26,00	25,99	26,00	25,98	25,98	25,99	25,99
c8	25,99	25,99	25,99	25,99	25,99	25,99	26,00	26,00	26,00	26,00	26,00
c9	25,99	25,99	25,99	25,99	25,99	25,99	25,99	25,99	25,99	25,99	25,99
c10	25,99	25,99	25,99	25,99	25,99	25,99	25,99	25,99	25,99	25,99	25,99

Tabla 8: Porcentaje de hit total 10.000 entradas -100% a 50%- Fuente: Elaboración propia

.

En la tabla 8 se observa el porcentaje de hit total cuando se filtró una clase de consulta solamente y las demás clases se mantienen con una admisión del 100%. Este primer experimento busca demostrar cómo puede mejorar el hit global si se reduce la admisión de las clases de consultas. Por ejemplo, cuando se varió la clase C1, el hit global más alto se obtiene cuando se deja el 100% de consultas con 1 término. Lo mismo ocurre para la clase C2 y C3. Esto cambia cuando se filtra la clase C4 y las demás se mantienen sin variar. Entonces el hit global más alto se obtiene cuando se deja entrar 90% de las consultas de 4 términos, para la clase C5 el hit más alto se obtiene cuando se deja entrar solamente el 70% de las consultas con 5 términos.

Siguiendo esto, se puede hacer una configuración que mescle las mejores tasas de hit obtenidas cuando se varió cada clase, la cual sería donde se admite el mejor porcentaje de hit de cada una de las clases. Es decir, la configuración seria: 100% para la clase C1, C2, C3, 90% para C4, 70% para C5, 55% para C6, 70 % para C7, 50% para C8, 70% para C9 y 50% para C10.

En la tabla 8 se puede ver también que en ciertos casos el hit global más alto se obtiene en el mínimo rango de admisión establecido en un principio, es decir, cuando se deja entrar el 50%. Otra observación es que en las 3 primeras clases al variar el porcentaje de admisión el hit global decae considerablemente por lo que estas 3 primeras clases no se variaran en la continuación de este experimento, el cual consiste en bajar la admisión de 45% a 0% para ver si el hit global aumenta.

En la tabla 9 se muestra el porcentaje de hit global obtenido cuando se restringen las clases c4 a la c10 con una admisión de 45% a 0%. Este experimento solo considera las estas 7 clases ya que como se ve en la tabla 7 el hit en de las 3 primeras clases siempre decae cuando se restringe la admisión.

.

Porcentaje de hit global										
Clase de consultas que varía	Porcentaje de admisión									
	45%	40%	35%	30%	25%	20%	15%	10%	5%	0%
c4	25,94	25,97	25,97	25,97	25,98	25,94	25,94	25,95	25,95	25,99
c5	25,97	25,97	25,97	25,98	25,98	25,98	25,98	25,99	25,99	26,03
c6	26,01	26,02	26,02	26,02	26,02	26,01	26,01	26,01	26,02	26,00
c7	25,99	25,99	25,97	25,97	25,96	25,96	25,96	25,97	25,97	25,97
c8	26,00	26,00	26,00	26,00	25,99	25,99	25,99	25,99	25,99	25,99
c9	25,99	25,99	25,99	26,00	26,00	26,00	26,00	26,00	26,00	26,00
c10	25,99	25,99	25,99	25,99	25,99	25,99	25,99	25,99	25,99	25,99

Tabla 9: Porcentaje de hit total 10.000 entradas -45% a 0%- Fuente: Elaboración propia

Se puede observar en la tabla 9 que de las 7 clases el mayor hit global se obtuvo cuando no se dejó ingresar la clase 5 completamente, es decir, su admisión fue 0%. Este hit es el mayor obtenido tanto para el primer experimento donde la admisión iba desde 100% hasta 50% (tabla 8) y el segundo que va desde 45 hasta 0% (tabla 9). El segundo hit más alto de este segundo experimento se obtuvo cuando se dejó ingresar el 30% de la clase 6 el cual es 26,02%.

Combinando los hit más altos obtenidos en las tablas 8 y 9, se puede hacer una mezcla y crear una política combinada para evaluar si el hit mejora. Se privilegia el cache de 10.000 entradas para el porcentaje de tasa de hit total, ya que es más restrictivo que el de 100.000 entradas.

Las principales conclusiones de este análisis son:

- Las clases de consultas relevantes son C1, C2 y C3, ya que son las que más hit poseen.
- Las 7 clases C4, C5, C6, C7, C8, C9 y C10 son las que menos impacto tienen en el cache, ya que el desalojo es aproximadamente de un 98%, cuando la admisión es de un 100%.

.

- Los análisis apuntan a que mientras más restrictivo es el tamaño del cache mayor es el impacto de las políticas de admisión.
- Cuando se reduce la admisión bajo 50% (tabla 9) el hit mejora en algunos casos y en otros se alcanza el mismo hit que con una admisión entre 100% y 50%. Sin embargo en otros baja el hit como es el caso de la clase C4.
- Variar una sola clase a la vez no ayuda a mejorar el hit. Es necesario encontrar una configuración donde se combinen distintos porcentajes de admisión para lograr un mejor resultado.
- Al restringir la admisión de las 3 primeras clases se ve como el hit cae de manera brusca (tabla 8). Por esta razón es mejor mantener la admisión de estas 3 clases en 100%.

6.6 Políticas de admisión combinadas

Realizado el análisis para las clases de consultas, se continúa con la experimentación combinando porcentajes de admisión para cada clase de consulta de manera individual y así llegar a una configuración de porcentajes de admisión, que permita obtener una ganancia en el hit global.

Como primer paso, para determinar cuánto es el hit global normal sin los porcentajes de admisión, se hacen dos experimentos, esto para establecer un punto de comparación y ver como el método mejora el hit.

El primer experimento consiste en ir dejando entrar de una clase en una sin ningún tipo de restricción más que el número de términos de la consultas. Es decir, se comienza en primer lugar dejando entrar solo a la clase C1 a cache y se calcula el hit total obtenido con solo dejar entrar a la clase C1. Luego se agrega la clase C2, teniendo ahora C1 y C2 en cache. Luego se deja ingresar C3 junto con C1 y C2, teniendo ahora C1, C2 y C3. Esto se repite hasta dejar entrar a las

.
diez clases (C1 hasta C10). Este experimento se hace para un cache de 10.000 entradas y para 100.000 entradas.

A continuación, se presenta la Tabla 10 como resumen con los hits obtenidos.

Nro.	Clases de consultas admitidas en cache	Hit global 10 mil entradas	Hit global 100 mil entradas
N1	C1	15,78%	18,83%
N2	C1, C2	24,72%	34,59%
N3	C1, C2, C3	26,09%	37,91%
N4	C1, C2, C3, C4	26,09%	38,16%
N5	C1, C2, C3, C4, C5	26,06%	38,11%
N6	C1, C2, C3, C4, C5, C6	25,98%	38,03%
N7	C1, C2, C3, C4, C5, C6, C7	25,98%	37,98%
N8	C1, C2, C3, C4, C5, C6, C7, C8	26,01%	37,94%
N9	C1, C2, C3, C4, C5, C6, C7, C8, C9	25,99%	37,93%
N10	C1, C2, C3, C4, C5, C6, C7, C8, C9, C10	25,99%	37,92%

Tabla 10: Experimento Hit global restringiendo clases de consultas, cache 10 y 100 mil entradas - Fuente: Elaboración propia

La tabla 10 muestra como el hit global varía dependiendo de qué clases se admiten completamente en cache. Se observa que el hit global para el cache de 10.000 entradas el máximo hit es obtenido en el experimento N3 y N4. En donde ambos tienen un 26,09% de hit, solo dejando entrar a las 3 y 4 primeras clases correspondientemente. Esto es concordante con el análisis hecho en la sección 6.5, donde se ve que las 3 primeras clases abarcan la mayoría del hit. Es por esto que a medida que se dejan ingresar las otras clases el hit va disminuyendo levemente. Cuando se dejan entrar las 10 clases a cache se llega a un hit global de 25,99%. Esto es debido también a que como se vio en el análisis de la sección 6.5, las clases desde C6 a C10 son en su gran mayoría de las veces desalojados. Es por esto que al dejar entrar a estas clases a cache el hit decae. Esta baja es leve, debido también a que la cantidad de consultas de esas clases

.
son pocas en comparación a las 5 primeras. Esto se puede ver en el gráfico de la ilustración 6 de la sección 6.1.

Para el cache de 100.000 entradas el hit máximo obtenido está en el experimento N4 con un 38,16% de hit global, donde solo se dejan entrar las 4 primeras clases a cache. Acá existe una diferencia con el cache de 10.000 entradas, ya que al ser un espacio más restrictivo, la posibilidad de entrar a cache es menor. Esto no ocurre tan estrictamente en un cache 10 veces mayor, por lo que es posible dejar entrar más consultas de una clase superior a cache sin necesidad de sacar consultas de las clases con más hit, que son las 3 primeras clases.

En el experimento representado en la tabla 10 se usa una política de admisión simple, la cual solo deja ingresar a las clases de una en una partiendo por la C1 hasta llegar a la C10. Este tipo de política es muy restrictiva, ya que ese está dejando entrar el 100% de las consultas de una clase y 0% de las demás. Es por esta razón que se proponen los rangos de admisión para tener una mayor flexibilidad en lo que se permite ingresar a cache.

El segundo experimento es con política de admisión por rango. Para esto se definen rangos de admisión para cada clase de consultas. Estos rangos son escogido tomando en cuenta los resultados de los análisis hechos hasta acá y el juicio experto. El primer paso de este experimento consiste en dejar entrar las 10 clases estudiadas con un 100% de admisión y ver el hit global obtenido. Una vez claro este el hit global obtenido dejando entrar las 10 clases sin restricciones de admisión, este caso se considera el caso base y con el cual se deben comparar los resultados, para ver si al aplicar rangos de admisión se mejora este caso base.

Para los siguientes experimentos se debe hacer lo mismo pero ahora estableciendo rangos de admisión para las clases, obtener hit global y comparar este hit obtenido con el hit global obtenido del caso base.

.

Con esto se espera obtener una configuración que permita aumentar la tasa de hit global obtenida en comparación a cuando se dejan entrar sin restricción las distintas clases a cache.

La elección de los rangos de admisión está basado en el análisis hecho en la sección 6.5. Es decir se priorizan las 3 primeras clases de consultas, ya que abarcan mayormente el hit y como se pudo ver en los experimentos anteriores, al reducir las 3 primeras clases se observa una caída en el hit global. Por lo tanto las clases que más varían en sus rangos de admisión son las 7 clases restantes.

A continuación en la tabla 11 se muestra los rangos de admisión que se escogieron para realizar este experimento.

	Clases de consultas									
	C1	**C2**	**C3**	**C4**	**C5**	**C6**	**C7**	**C8**	**C9**	**C10**
Test Base	100	100	100	100	100	100	100	100	100	100
Test 1	95	95	95	90	90	50	50	50	50	50
Test 2	100	100	100	90	90	50	50	50	50	50
Test 3	100	100	90	80	80	60	60	60	60	60
Test 4	90	90	90	90	90	50	50	50	50	50
Test 5	90	90	90	95	95	70	70	70	70	70
Test 6	90	90	90	100	100	50	50	50	50	50
Test 7	100	100	100	50	50	50	50	50	50	50
Test 8	100	100	100	40	40	40	40	40	40	40
Test 9	100	100	100	40	40	30	30	30	30	30
Test 10	100	100	100	30	30	30	30	30	30	30
Test 11	100	100	100	40	40	25	25	25	25	25
Test 12	95	95	95	50	50	30	30	30	30	30
Test 13	100	100	100	30	30	20	20	20	20	20
Test 14	100	100	100	20	20	20	20	20	20	20

Tabla 11: Pruebas con los rangos de admisión por clase de consulta - Fuente: Elaboración propia

.

Ahora se presenta la tabla 12 con un resumen del hit obtenido para cada prueba realizada.

	Hit global 10 mil entradas	Hit global 100 mil entradas
Test Base	25,99%	37,92%
Test 1	25,72%	37,21%
Test 2	26,04%	38,00%
Test 3	25,94%	37,69%
Test 4	25,26%	36,43%
Test 5	25,22%	36,43%
Test 6	25,23%	36,45%
Test 7	26,01%	37,83%
Test 8	26,04%	37,86%
Test 9	26,07%	37,88%
Test 10	26,08%	37,87%
Test 11	26,07%	37,89%
Test 12	25,75%	37,11%
Test 13	26,13%	37,89%
Test 14	26,12%	37,85%

Tabla 12: Hit global pruebas rangos de admisión por clase de consultas - Fuente: Elaboración propia

Las 14 pruebas realizadas con distintos rangos de admisión permiten ver cómo afecta al hit estas variaciones en la admisión de las consultas. De las distintas pruebas hechas, las que mejoran el hit global mayoritariamente son en las cuales se restringe el rango de admisión de las 7 clases C4, C5, C6, C7, C8, C9 y C10. Se observa que en la prueba base el hit es 25,99% para el cache de 10.000 entradas y 37,92% para el de 100.000 entradas. Luego las pruebas donde se mejora este hit global es donde se deja un rango de 100% para las 3 primeras clases, y se varían las 7 clases restante. Esto para un cache de 10.000 entradas.

.
Mientras que para un cache de 100.000 entradas la mejor prueba fue cuando se dejaron las 3 primeras clases con el 100% de admisión, las dos siguientes con una restricción de 90% y las siguientes 5 clases con un 50% de admisión. Esto ratifica que al restringir estrictamente las 7 clases finales es cuando se registran los mejores hits.

Al impedir que entre el porcentaje de consultas menos consultadas de su clase hace que el hit aumente, ya que solo se deja ingresar a cache el porcentaje de consultas más probable de ser consultado. Es decir las clases que suelen ser mayormente desalojadas (C4, C5, C6, C7, C8, C9, C10) solo se está dejando entrar lo más probablemente consultado.

En las pruebas 7 hasta la 15 excluyendo la prueba 12, superan el hit de la prueba base para el cache de 10.000 entradas. Estas 7 pruebas tienen en común que se deja entrar el 100% de C1, C2 y C3 y se varían las admisiones de las otras 7 clases.

Para el cache de 100.000 entradas, la prueba 2 es la única que supera a la prueba base. Esto se debe a que hay más espacio en cache, lo cual permite que ingresen más consultas de las clases siguientes a las que más se consultan, es decir la clase C4 y C5. Es importante destacar que en la prueba 2 no se deja entrar completamente las consultas de la clase C4 y C5, lo cual confirma que la política de admisión es efectiva, esto es visto mejor en los resultados de las prueba hechas para el cache de 10.000 entradas, ya que las pruebas superan la prueba base tanto para el cache de 10.000 entradas como para el de 100.000.

En un cache menos restrictivos como el de 100.000 entradas, dejar entrar las 5 primeras clases implica dejar entrar a las clases que más hit tienen en el cache, lo cual, al tener 10 veces el espacio del cache de 10.000 entradas provoca que no exista demasiada competencia por entrar a cache, lo que hace que el hit aumente ya que las consultas con más hit siempre están en cache. En un ambiente real no es así, debido a que la cantidad de consultas serán muchas más que las usadas en este trabajo, y el cache tampoco será de tanta capacidad.

.

Siguiendo con los experimentos de políticas de admisión combinadas, ahora se realiza una prueba donde se varían en conjunto las 7 clases que mayor impacto tuvieron en el hit cuando se modificó su admisión.

Este experimento consiste en variar conjuntamente las clases, la forma en que se varían las clases es partiendo por variar las 2 últimas clases, es decir, C9 con C10. Luego de esto se agrega una clase más, en este caso C8, y así sucesivamente, hasta variar conjuntamente C4, C5, C6, C7, C8, C9, C10.

Este experimento tiene como objetivo ver que impacto tiene variar las clases que han tenido mejores resultados en el hit al variar su admisión.

La tabla 13 muestra la variación de las clases conjuntamente para un cache de 10.000 entradas.

Porcentaje de hit Total											
Clase de consultas que varían conjuntamente	Porcentaje de admisión										
	100%	95%	90%	85%	80%	75%	70%	65%	60%	55%	50%
c9, c10	25,99	25,99	25,99	25,99	25,99	25,99	26,00	26,00	26,00	26,00	26,00
c8, c9, c10	25,99	25,99	26,00	26,00	26,00	26,00	26,00	26,01	26,01	25,99	25,99
c7, c8, c9, c10	25,99	25,99	26,00	26,00	26,00	26,00	26,00	26,01	26,01	25,99	25,99
c6, c7, c8, c9, c10	25,99	25,99	26,00	26,00	26,00	25,99	25,99	25,99	25,99	25,98	25,98
c5, c6, c7, c8, c9, c10	25,99	26,00	26,00	26,00	26,00	26,00	25,99	25,99	26,00	26,01	26,01
c4, c5, c6, c7, c8, c9, c10	25,99	25,99	25,99	26,00	25,99	26,00	26,01	26,01	26,02	26,02	26,01

Tabla 13: Variación de clases conjuntamente – 10.000 entradas - Fuente: Elaboración propia

En la tabla 13 se observa los resultados obtenidos, de los 6 experimentos hechos todos superaron la tasa de hit obtenido cuando se dejan entrar todas las clases sin restricción. De las 6 pruebas hechas la tasa de hit más alta se registró cuando se variaron las 7 clases conjuntamente, la cual registró una tasa de hit de 26,02%. En las demás pruebas también se superó la prueba donde se dejan entrar las 10 clases sin restricciones.

.

En los experimentos anteriores se observó que los mejores resultados se presentaban cuando se restringen las 7 clases finales, es decir, cuando a todas estas clases se le asignaba una admisión distinta de 100%. Esto se sigue repitiendo en estos experimentos lo que demuestra que la política de admisión funciona.

Ahora el mismo experimento se realiza con el cache de 100.000 entradas

Porcentaje de hit Total											
Clase de consultas que varían conjuntamente	Porcentaje de admisión										
	100%	95%	90%	85%	80%	75%	70%	65%	60%	55%	50%
c9, c10	37,92	37,92	37,93	37,93	37,93	37,93	37,93	37,93	37,93	37,94	37,94
c8, c9, c10	37,92	37,93	37,93	37,93	37,94	37,94	37,94	37,94	37,94	37,95	37,95
c7, c8, c9, c10	37,92	37,93	37,93	37,93	37,94	37,94	37,94	37,94	37,94	37,95	37,95
c6, c7, c8, c9, c10	37,92	37,93	37,93	37,94	37,94	37,94	37,95	37,95	37,95	37,96	37,97
c5, c6, c7, c8, c9, c10	37,92	37,93	37,94	37,95	37,96	37,97	37,98	37,99	37,99	38,01	38,01
c4, c5, c6, c7, c8, c9, c10	37,92	37,94	37,93	37,95	37,95	37,96	37,97	37,98	37,99	37,99	37,99

Tabla 14: Variación de clases conjuntamente – 100.000 entradas - Fuente: Elaboración propia

La tabla 14 muestra los mismos 6 experimentos realizados anteriormente pero ahora con un cache de 100.000 entradas, nuevamente se ve que al variar las clases conjuntamente se supera el hit obtenido cuando se dejan entrar todas la clases con un 100%.

Al variar las clases conjuntamente se observa que el hit aumenta en cada porcentaje de admisión, como se espera que ocurra al variar las últimas 7 clases. Pero también se observa que el hit mejora cuando más clases son variadas conjuntamente. Es decir, el mayor hit obtenido al variar solo la clase C9 y C10 es menor que el obtenido al variar C8, C9 y C10.

Como se explicó anteriormente los mejores resultados en cuanto al aumento de hit, se han obtenido al variar las 7 últimas clases. Esto se ha visto en

.
todos los experimentos hechos durante este trabajo. El aumento de hit al variar estas clases se debe a que estas 7 clases son las que menos son consultadas por las personas y al evitar que entren a cache se está dando espacio a que entren las consultas de las 3 primeras clases que si son muy consultadas por las personas. Además al reducir la admisión de las clases poco consultadas se está dejando entrar el porcentaje de esas consultas más hechas por las personas.

Continuando con este experimento se observa que los resultados con mejor hit se obtienen cuando se restringe su admisión y en ciertos casos este mejor hit es alcanzado con una admisión de 50% o cercana. Teniendo en cuenta estos resultados se puede pensar que si se sigue restringiendo la admisión el hit continúe aumentando.

Como se ve en las tabla 13 los mejores resultados para un cache con 10.000 entradas no siempre se alcanza cuando se deja entrar el 50% de la consultas, de hecho solo se alcanza en un caso. Este caso es cuando se restringe las clases C5 a la C10. Pero los resultados son distintos para un cache de 100.000 entradas donde el hit más alto si se obtiene siempre en la admisión de 50%.

Teniendo estos resultados en cuenta se vuelve a repetir el experimento pero ahora se baja la restricción de admisión de 45% a 0%. Tanto para el experimento con 10.000 entradas como con 100.000. Esto para ver si se mejora el hit obtenido en el primer experimento.

Al igual que antes se comienza con el experimento para el cache de 10.000 entradas. En la tabla 15 se presentan los resultados obtenidos para el mismo experimento variando las 7 últimas clases conjuntamente. Ahora esta variación en es con una admisión que va desde 45% a 0%.

.

Porcentaje de hit Total										
Clase de consultas que varían conjuntamente	Porcentaje de admisión									
	45%	40%	35%	30%	25%	20%	15%	10%	5%	0%
c9, c10	26,00	26,00	26,00	26,00	26,00	26,00	26,00	26,00	26,01	26,01
c8, c9, c10	25,99	25,99	25,98	25,99	25,99	25,99	25,97	25,97	25,97	25,98
c7, c8, c9, c10	25,98	25,98	25,99	25,99	26,00	25,96	25,96	25,96	25,97	25,98
c6, c7, c8, c9, c10	26,03	26,03	26,00	26,01	26,01	26,01	26,01	26,04	26,05	26,06
c5, c6, c7, c8, c9, c10	26,02	26,04	26,04	26,05	26,09	26,09	26,10	26,13	26,13	26,09
c4, c5, c6, c7, c8, c9, c10	26,02	26,04	26,07	26,08	26,14	26,12	26,05	26,09	26,09	26,09

Tabla 15: Variación de clases conjuntamente – 10.000 entradas - Fuente: Elaboración propia

Se observa en la tabla 15 que al restringir la admisión de las 2 últimas clases la ganancia de hit no presenta mayor variación, lo cual también ocurre con las los siguientes experimentos hasta llegar al experimento donde se varían las conjuntamente las clases C6, C7, C8, C9, C10.

En el experimento donde se varían las clases C6, C7, C8, C9, C10 la ganancia de hit se comienza a notar de mejor manera respecto a los 3 primeros experimentos de la tabla 15. Esto continúa con los dos últimos experimentos llegando al mejor hit cuando se varían las clases C4, C5, C6, C7, C8, C9, C10 conjuntamente.

El mejor hit obtenido en este experimento se registra cuando se varían las 7 clases las cuales tenían una admisión de 25% y el hit es de 26,14%.

Ahora el mismo experimento se realiza con un cache de 100.000 entradas. El cual como se pudo ver en el mismo experimento pero con una admisión de 100% a 50% todos los hit altos se registraron con una admisión del 50%.

En la tabla 16 se muestran los resultados obtenidos al realizar el experimento de la variación de las clases conjuntamente con una admisión que va desde 45% a 0% para un cache de 100.000 entradas.

.

Porcentaje de hit Total										
Clase de consultas que varían conjuntamente	Porcentaje de admisión									
	45%	40%	35%	30%	25%	20%	15%	10%	5%	0%
c9, c10	37,94	37,94	37,94	37,93	37,93	37,94	37,94	37,93	37,94	37,94%
c8, c9, c10	37,94	37,95	37,95	37,95	37,96	37,96	37,96	37,97	37,97	37,98%
c7, c8, c9, c10	37,98	37,98	37,99	37,99	38,00	38,00	37,99	38,00	38,01	38,03%
c6, c7, c8, c9, c10	38,02	38,03	38,03	38,04	38,05	38,05	38,06	38,07	38,08	38,11%
c5, c6, c7, c8, c9, c10	38,02	38,04	38,05	38,06	38,07	38,08	38,10	38,11	38,12	38,16%
c4, c5, c6, c7, c8, c9, c10	37,85	37,86	37,85	37,87	37,89	37,85	37,85	37,87	37,88	37,91%

Tabla 16: Variación de clases conjuntamente – 100.000 entradas - Fuente: Elaboración propia

En la tabla 16 se observan como los resultados obtenidos al ejecutar el experimento con una admisión desde 45% a 0% mejora los resultados obtenidos previamente en el mismo experimento con una admisión de 100% a 50%.

Al igual que en el experimento con el cache de 10.000 entradas se observa una similitud. Esta similitud se da en los 3 primeros experimentos, donde no se observa una variación en el hit. Lo cual cambia al llegar al experimento donde se varían las clases C6, C7, C8, C9, C10. Acá la variación registrada al variar la admisión es mayor que en los 3 experimentos anteriores.

El aumento del hit mejora en el siguiente experimento donde se varían las clases C5, C6, C7, C8, C9, C10, en donde el mejor hit se alcanza cuando todas estas clases tienen una admisión de 0%. Esto quiere decir que dejar entrar las 4 primeras clases completamente mejora el hit en un cache con mayor espacio.

Luego al incluir la clase C4 en esta variación conjunta de clases para un cache de 100.000 entradas se observa una caída en la tasa de hit. Esto indica que la clase C4 tiene una importancia en el resultado del hit, el cual impacta de manera negativa al reducir su admisión.

.

Este impacto negativo en el hit al reducir la admisión de la clase C4 en un cache de 100.000 entradas se explica debido al mayor espacio que existe para almacenar las consultas que en un cache restrictivo como el de 10.000 entradas no hay.

Recapitulando lo explicado con los dos experimentos al variar las clases de manera conjunta. En primer lugar se observaron los resultados con la variación de las clases con una admisión de 100% a 50% como se ha hecho hasta ahora en todas las otras pruebas. En este experimento se observó una leve variación en el hit, tanto para el cache de 10.000 entradas como para el de 100.000 entradas.

Para esta primera parte del experimento el hit más alto registrado para un cache de 10.000 entradas fue en la prueba donde se varían las clases C4, C5, C6, C7, C8, C9 y C10 con una admisión de 55% y la cual registro un hit de 26,02%.

Mientras que para un cache de 100.000 entras el hit más alto registrado fue en la prueba donde se variaron las clases C5, C6, C7, C8, C9 y C10 con una admisión de 50% y la cual dio un hit de 38,01%.

Esta primera parte del experimento dio paso a la segunda parte debido principalmente a que los resultados obtenidos en el cache de 100.000 entradas mostraba a todos los resultados altos en el porcentaje de admisión más bajo definido para esta primera parte, es decir 50%. Lo que llevo a comprobar si bajar más el porcentaje de admisión mejoraría el hit.

Para la segunda parte del experimento el hit más alto registrado para un cache de 10.000 entradas fue donde se varían las clases C4, C5, C6, C7, C8, C9 y C10 con una admisión de 25% para cada clase, la cual registro un hit de 26,14%.

.

Mientras que para un cache de 100.000 entradas el hit más alto registrado en esta segunda parte, es cuando se varían las clases C5, C6, C7, C8, C9 y C10 con una admisión de 0% para cada clase, la cual registro un hit de 38,16%.

.

7. CONCLUSIONES

Este trabajo da evidencia de cómo las políticas de admisión son relevantes para la gestión eficiente del cache. Esto es posible verificar en literatura estudiada y presentada en el capítulo 5, trabajos como (Aggarwal, Wolf, & Yu, 1999), (Baeza-Yate, Junqueira, Plachouras, & Witschel, 2007), (Long & Suel, 2006) especifican lo importante que es tener políticas de admisión en conjunto con las políticas de desalojo para mejorar la eficiencia del cache.

Los log de consultas estudiados muestran como la proporción de consultas hechas por los usuarios es muy dispar, siendo las consultas de 2 términos las más realizadas por los usuarios, y las 5 primera las con mayor porcentaje por ende las más relevante. Sin embargo considerando las 10 clases estudiadas, las 3 primeras son las que poseen mayor frecuencia en comparación a las 7 restantes y por lo tanto las más importantes dentro del cache. Al estar concentrado la frecuencia de las consultas en las 3 primeras clases, es posible ser más estrictos con la admisión de las 7 clases restante, dejando ingresar un bajo número de estas o excluyéndolas completamente.

En la parte experimental se pude observar que todas las clases de consultas se asemejan a una campana gaussiana, por lo que se pueden usar las propiedades de esta distribución para poder diseñar políticas de admisión.

Finalmente se comprueba que las 3 primeras clases son las más importantes de conservar en cache. Esto se ratifica tanto por la cantidad de consultas que se hacen para estas 3 clases donde en conjunto abarcan aproximadamente el 74% de las peticiones. Como también cuando se hacen las pruebas de admisión, en donde los resultados obtenidos son solo bajas en el hit.

En las pruebas de admisión combinadas, se observa que las tasas de hit más altas es donde se deja entrar a las 3 primeras clases de forma completa y se restringe las restante 7.

.

Reducir la admisión bajo un 50% para las 7 últimas clases contribuye a la mejora del hit en experimentos con los ambos tamaños de cache escogidos para este trabajo.

En todos los experimentos donde se redujo la admisión de las 3 primeras clases solo se registraron bajas en el hit. Por lo que en los experimentos donde la admisión se baja aún más (45% a 0%) solo se consideraron las 7 últimas clases. Ya que estas son las que mejores resultados presentaron al disminuir su admisión.

Los resultados obtenidos aplicando la política de admisión no fueron los esperados. Ya que al disminuir la admisión de las 3 clases más importantes solo se registraron bajas en el hit, sin embargo la política de admisión muestra que funciona, ya que al aplicarlo en las últimas 7 clases estudiadas el hit aumenta. Lo que indica que admitir solo el porcentaje de las consultas más hechas por los usuarios contribuye a la mejora en el hit.

.

7.1 Trabajo futuro

Durante el desarrollo de este trabajo se encontraron ciertas líneas de investigación en las que se puede seguir profundizando para mejorar los resultados obtenidos.

Al comenzar la experimentación del método basado en rangos de admisión por consultas se comenzó a idear cuales podrían ser las combinaciones que lograran maximizar el hit global. Se encontraron limitaciones propias de la experimentación con grandes volúmenes de datos, es decir el tiempo requerido para realizar estos experimentos.

Para encontrar las mejores combinaciones de rangos se requiere hacer una combinatoria de todos los rangos de admisión y las 10 clases de consultas. Esta combinatoria es impracticable debido a que este método requiere la utilización de logs de consultas reales los cuales son muy grandes y provoca que el tiempo requerido para hacer dicha combinatoria sea imposible.

Es por esto que se pensó que una buena posibilidad de resolver esto es mediante una heurística. Esta heurística podría ser similar al problema de la mochila debido a las características del problema que acá se estudia. Donde la capacidad de la mochila está dada por el tamaño del cache y los objetos a introducir en la mochila son los rangos de admisión de las consultas.

La utilización de una heurística podría determinar que rangos de consultas son los mejores para cada consulta y con esto se podrían lograr mejores resultados. Si bien esto es una primera aproximación se podría usar otro tipo de estrategia que permita determinar que rangos son los mejores, se ejemplifica con la mochila debido a la similitud del problema estudiado con este clásico problema.

Otra mejora que se puede hacer consiste en realizar un nuevo estudio de logs de consultas reales y encontrar otro patrón que permitan idear una nueva política de admisión. Luego con esta nueva política de admisión creada a partir

.
de este nuevo análisis se tendría que combinar esta nueva política con la propuesta en este trabajo, para así ver si los resultados mejoran con respecto a los obtenidos en este trabajo.

7.2 Lecciones aprendidas

Durante la realización de este trabajo se pueden mencionar lecciones aprendidas respecto a la ingeniería civil informática.

Mientras se realizó este trabajo uno se puede percatar de que toda la formación que se recibe durante la carrera puede servir en un momento determinado del proyecto, lo cual puede ayudar a encontrar una solución al problema, como también encontrar distintos métodos.

Estas respuestas o métodos no necesariamente tienen que ser las relacionadas con la carrera que se estudia, sino que todas las materias que uno tiene en la malla pueden servir para esto.

Por ejemplo durante este trabajo no solo se utilizaron los conocimientos relacionados con el fuerte de la carrera, sino que el método empleado tiene que ver directamente con estadística. Si bien durante la carrera no se recibe mayor profundización acerca de ciertas materias. Estas materias tienen que tenerse presente a la hora de desarrollar un trabajo ya que siempre pueden ser de utilidad.

Otra lección aprendida tiene relación con lo utilizado durante los cursos. Es decir, durante la carrera se emplean ciertas herramientas para enseñar a los alumnos, lo cual no quiere decir que estas sean las mejores ni las más indicadas para todo lo que se desarrolle durante la vida profesional. Ya que estas herramientas, metodologías, son usadas con fines académicos por lo que no siempre son lo mejor.

.

Por lo tanto, lo aprendido con esas herramientas, metodologías, tienen que servir de base para después encontrar soluciones adecuadas y no siempre usar lo mismo- Ya que al utilizar lo mismo se cae en los mismo errores y no se mejora, por lo que se tiene que seguir en constante aprendizaje de las nuevas tecnologías. Ya que este constante aprendizaje es intrínseco a la carrera.

Durante toda la carrera se hace énfasis en la adopción de las buenas prácticas a la hora de realizar trabajos de todo tipo. Estas buenas prácticas tendrían que ser empleadas siempre y no solo cuando se exigen. Esto porque se suelen privilegiar los resultados dejando de lado las buenas practicas. Lo que a la larga puede ocasionar problemas que se pueden evitar haciendo uso de estas.

7.3 Implicaciones prácticas

Actualmente las aplicaciones de gran escala están siendo usadas por gran cantidad de personas. Aplicaciones como Facebook, Twitter, YouTube, Google son usadas día a día por millones de personas. Lo cual implica crear nuevas estrategias para hacer estas aplicaciones mejores y eficientes.

Este trabajo da evidencia de que esta gran cantidad de personas que utiliza estas aplicaciones tienen comportamientos repetitivos. Este comportamiento repetitivo o patrones de comportamientos se pueden usar para diseñar métodos que mejoren la eficiencia de estas aplicaciones.

Principalmente este trabajo da evidencia sobre la manera de cómo las personas realizan las consultas. Las cuales siguen un patrón característico. Este patrón es utilizado para diseñar una política de admisión la cual tiene como objetivo mejorar la eficiencia del cache.

Este trabajo se puede usar como base para diseñar una política de admisión robusta que ayude a mejorar los sistemas de cache utilizados por las

.
aplicaciones de gran escala. La forma en que se puede utilizar este trabajo, es en conjunto con otras nuevas políticas que se diseñen. Como también mejorar la política presentada en este trabajo agregando características nuevas que permitan mejorar los resultados obtenidos.

La principal área donde este trabajo impacta es donde se manejan grandes volúmenes de datos relacionados con usuarios. De los cuales se pueda hacer un análisis y extraer patrones y características relevantes para el posterior diseño de una política de admisión.

Por lo tanto innovar en las estrategias que se utilizan hoy en día es fundamental para ir a la par con este crecimiento en el uso de estas aplicaciones. Por lo que incluir políticas de admisión basadas en patrones encontrados en el comportamiento de usuario genera un impacto positivo en los sistemas de cache utilizado por estas grandes aplicaciones.

.

8. REFERENCIAS

1. Einziger, G., & Friedman, R. (2014, February). Tinylfu: A highly efficient cache admission policy. In *Parallel, Distributed and Network-Based Processing (PDP), 2014 22nd Euromicro International Conference on* (pp. 146-153). IEEE.

2. Palpanas, T., Larson, P. Å., & Goldstein, J. (2001). *Cache management policies for semantic caching*. Technical Report CSRG-439, Dept. of Computer Science, University of Toronto.

3. Baeza-Yate, R., Junqueira, F., Plachouras, V., & Witschel, H. F. (2007, January). Admission policies for caches of search engine results. In *String Processing and Information Retrieval* (pp. 74-85). Springer Berlin Heidelberg.

4. Aggarwal, C., Wolf, J. L., & Yu, P. S. (1999). Caching on the world wide web.*Knowledge and Data Engineering, IEEE Transactions on*, *11*(1), 94-107.

5. Altingovde, I. S., Ozcan, R., & Ulusoy, Ö. (2009). A cost-aware strategy for query result caching in web search engines. In *Advances in Information Retrieval* (pp. 628-636). Springer Berlin Heidelberg.

6. Cambazoglu, B. B., Altingovde, I. S., Ozcan, R., & Ulusoy, Ö. (2012). Cache-based query processing for search engines. *ACM Transactions on the Web (TWEB)*, *6*(4), 14.

7. Baeza-Yates, R., Gionis, A., Junqueira, F. P., Murdock, V., Plachouras, V., & Silvestri, F. (2008). Design trade-offs for search engine caching. *ACM Transactions on the Web (TWEB)*, *2*(4), 20.

8. Subašić, I., & Castillo, C. (2010, August). The effects of query bursts on web search. In *Web Intelligence and Intelligent Agent Technology (WI-IAT), 2010 IEEE/WIC/ACM International Conference on* (Vol. 1, pp. 374-381). IEEE.

.
9. Karakostas, G., & Serpanos, D. N. (2002). Exploitation of different types of locality for Web caches. In *Computers and Communications, 2002. Proceedings. ISCC 2002. Seventh International Symposium on* (pp. 207-212). IEEE.

10. Baeza-Yates, R., Gionis, A., Junqueira, F., Murdock, V., Plachouras, V., & Silvestri, F. (2007, July). The impact of caching on search engines. In*Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 183-190). ACM.

11. Zipf, G. K. (1949). Human behavior and the principle of least effort.

12. Ozcan, R., Altingovde, I. S., & Ulusoy, Ö. (2011). Cost-aware strategies for query result caching in web search engines. *ACM Transactions on the Web (TWEB)*, *5*(2), 9.

13. Zhang, J., Long, X., & Suel, T. (2008, April). Performance of compressed inverted list caching in search engines. In *Proceedings of the 17th international conference on World Wide Web* (pp. 387-396). ACM.

14. Ozcan, R., Altingovde, I. S., & Ulusoy, Ö. (2008, April). Static query result caching revisited. In *Proceedings of the 17th international conference on World Wide Web* (pp. 1169-1170). ACM.

15. Long, X., & Suel, T. (2006). Three-level caching for efficient query processing in large web search engines. *World Wide Web*, *9*(4), 369-395.

16. Lempel, R., & Moran, S. (2003, May). Predictive caching and prefetching of query results in search engines. In *Proceedings of the 12th international conference on World Wide Web* (pp. 19-28). ACM.

17. Gan, Q., & Suel, T. (2009, April). Improved techniques for result caching in web search engines. In *Proceedings of the 18th international conference on World Wide Web* (pp. 431-440). ACM.

.

18. Fagni, T., Perego, R., Silvestri, F., & Orlando, S. (2006). Boosting the performance of web search engines: Caching and prefetching query results by exploiting historical usage data. *ACM Transactions on Information Systems (TOIS)*, *24*(1), 51-78.

19. Otoo, E., Rotem, D., & Shoshani, A. (2005). Impact of admission and cache replacement policies on response times of jobs on data grids. *Cluster Computing*, *8*(4), 293-303.

20. González-Cañete, F. J., Casilari, E., & Trivino-Cabrera, A. (2006, May). Two new metrics to evaluate the performance of a web cache with admission control. In *Electrotechnical Conference, 2006. MELECON 2006. IEEE Mediterranean* (pp. 696-699). IEEE.

21. Scilab Enterprises. (s.f.). *Scilab*. Obtenido de Scilab: http://www.scilab.org/

22. McNeese, D. B. (2009). *spcforexcel*. Obtenido de spcforexcel: https://www.spcforexcel.com/knowledge/basic-statistics/normal-distribution.

23. Gómez Pantoja, C. L. (2013). Servicios de Cache Distribuidos para Motores de Búsqueda Web. 180.

24. Subašic, I., & Castillob, C. (2013). Investigating query bursts in a web search engine.

.

9. ANEXOS

9.1 Anexo A

Log 201105		Log 201106
Clases de consultas	Porcentaje	Porcentaje
C1	22,67%	22,14%
C2	29,97%	30,49%
C3	21,25%	21,08%
C4	11,82%	11,89%
C5	6,34%	6,44%
C6	3,35%	3,37%
C7	1,78%	1,80%
C8	1,01%	1,01%
C9	0,60%	0,60%
C10	0,36%	0,36%
C > 10	0,84%	0,81%

Anexo 1: Cantidad de consultas según clase- Fuente: Elaboración propia

9.2 Anexo B

Tasa de hit por clase de consulta					
Clases de consultas	Porcentaje de admisión				
	100%	95%	90%	85%	80%
C1	73,61%	71,21%	71,21%	68,71%	68,71%
C2	50,53%	49,53%	47,34%	47,34%	44,33%
C3	21,47%	21,03%	20,07%	19,24%	18,05%
C4	7,75%	7,50%	7,16%	6,75%	6,14%
C5	3,91%	3,84%	3,36%	3,21%	2,83%
C6	1,93%	1,86%	1,80%	1,70%	1,66%
C7	1,05%	0,84%	0,81%	0,74%	0,71%
C8	0,65%	0,61%	0,59%	0,56%	0,54%
C9	0,72%	0,67%	0,65%	0,63%	0,61%
C10	0,73%	0,68%	0,66%	0,63%	0,61%

Anexo 2: Tasa de hit por clase de consulta 100.000 entradas – Parte 1 - Fuente: Elaboración propia

.

Tasa de hit por clase de consulta						
Clases de consultas	Porcentaje de admisión					
	75%	70%	65%	60%	55%	50%
C1	59,45%	59,45%	59,45%	59,45%	44,20%	44,20%
C2	44,33%	37,68%	37,68%	37,68%	30,05%	30,05%
C3	18,05%	16,07%	16,07%	16,07%	13,72%	13,72%
C4	5,97%	5,47%	4,97%	4,26%	4,06%	3,70%
C5	2,72%	2,67%	2,40%	2,33%	1,86%	1,79%
C6	1,59%	1,58%	1,50%	1,46%	1,38%	1,38%
C7	0,70%	0,63%	0,57%	0,57%	0,52%	0,52%
C8	0,52%	0,48%	0,44%	0,44%	0,41%	0,41%
C9	0,58%	0,55%	0,52%	0,52%	0,47%	0,45%
C10	0,59%	0,57%	0,54%	0,54%	0,51%	0,42%

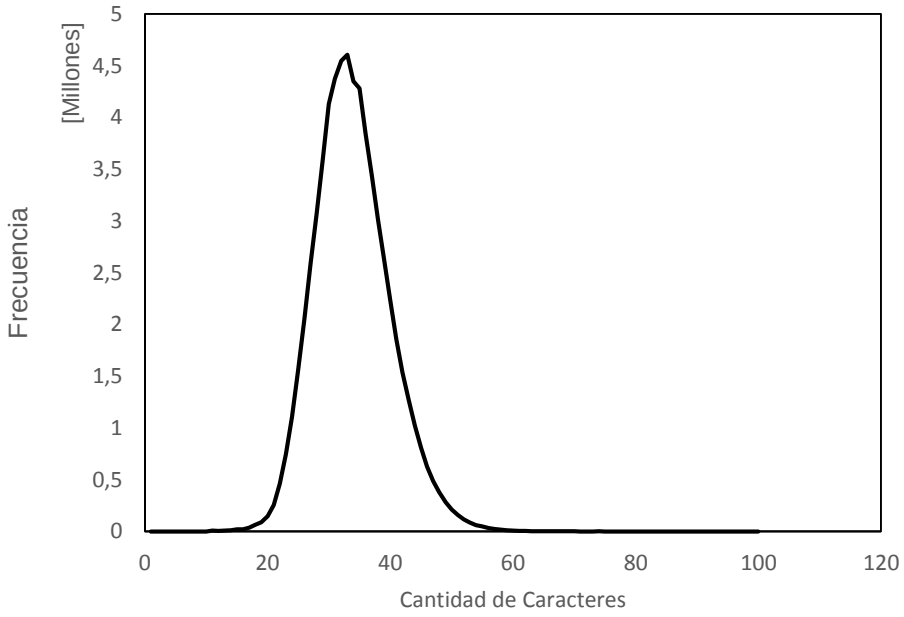
Anexo 3: Tasa de hit por clase de consulta 100.000 entradas – Parte 1 - Fuente: Elaboración propia

Porcentaje de hit total											
Clases de consultas	Porcentaje de admisión										
	100%	95%	90%	85%	80%	75%	70%	65%	60%	55%	50%
C1	37,90	37,47	37,47	36,96	36,96	35,00	35,00	35,00	35,00	31,71	31,71
C2	37,90	37,65	37,03	37,03	36,21	36,21	34,27	34,27	34,27	32,07	32,07
C3	37,90	37,86	37,70	37,57	37,38	37,38	37,07	37,07	37,07	36,68	36,68
C4	37,90	37,91	37,89	37,88	37,85	37,86	37,82	37,79	37,73	37,73	37,73
C5	37,90	37,91	37,91	37,91	37,91	37,91	37,92	37,92	37,92	37,91	37,92
C6	37,90	37,91	37,91	37,93	37,93	37,94	37,94	37,95	37,96	37,97	37,97
C7	37,90	37,90	37,91	37,91	37,92	37,92	37,93	37,93	37,93	37,94	37,94
C8	37,90	37,91	37,91	37,91	37,91	37,91	37,91	37,92	37,92	37,93	37,93
C9	37,90	37,91	37,91	37,91	37,91	37,91	37,91	37,91	37,91	37,91	37,92
C10	37,90	37,90	37,90	37,91	37,91	37,91	37,91	37,91	37,91	37,91	37,91

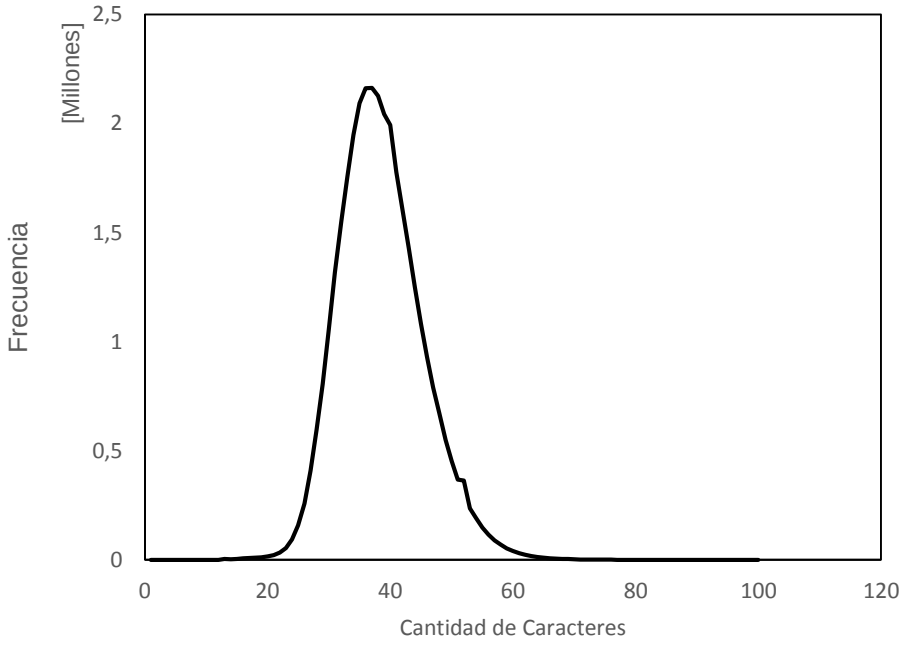
Anexo 4: Porcentaje de hit total 100.000 entradas - Fuente: Elaboración propia

.

9.3 Anexo C

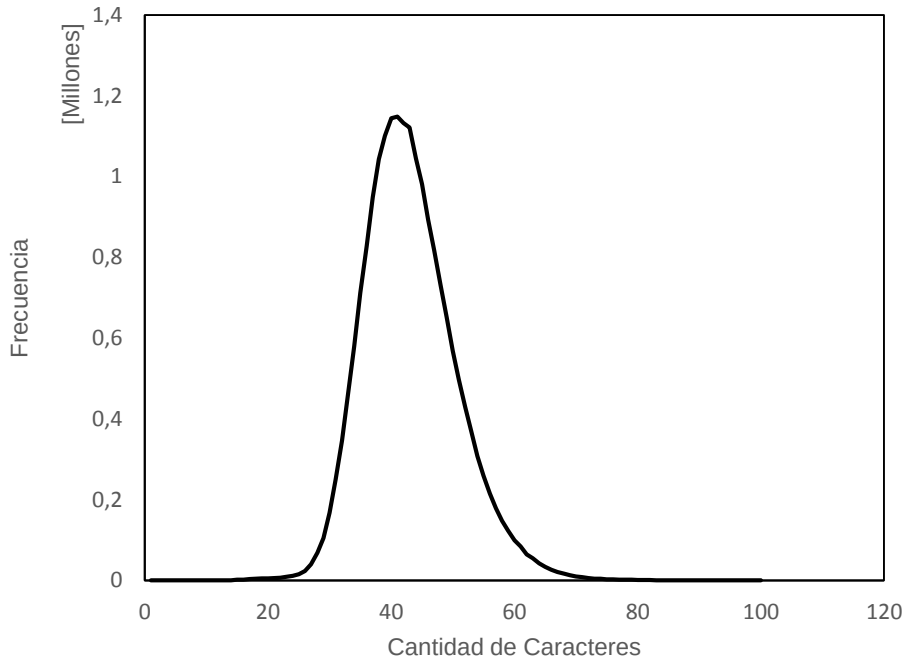


Anexo 5: Clase 6 de consultas – Fuente: Elaboración propia

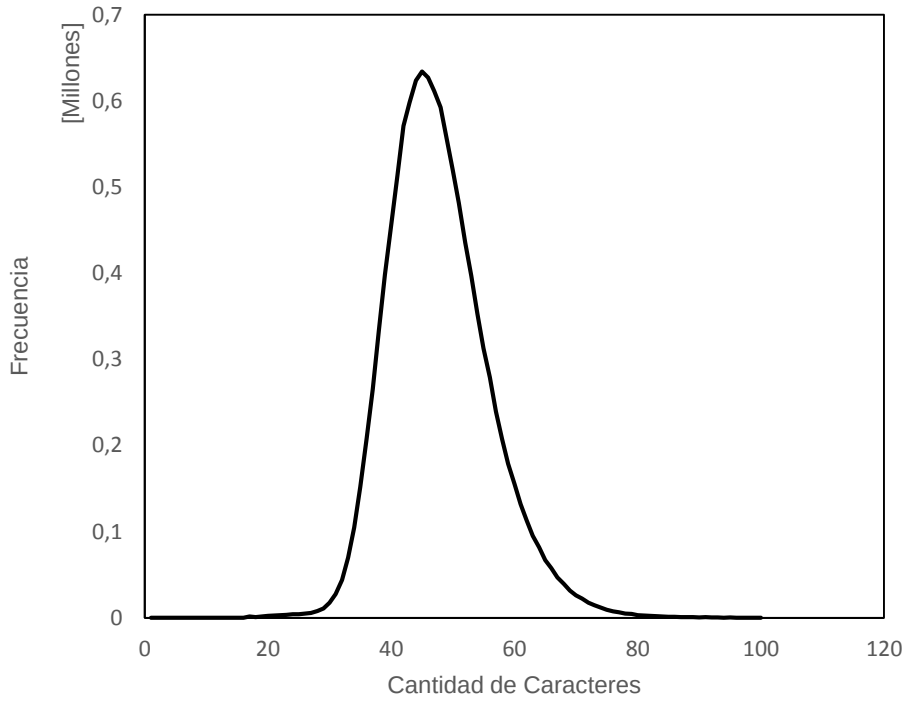


Anexo 6: Clase 7 de consultas – Fuente: Elaboración propia

.

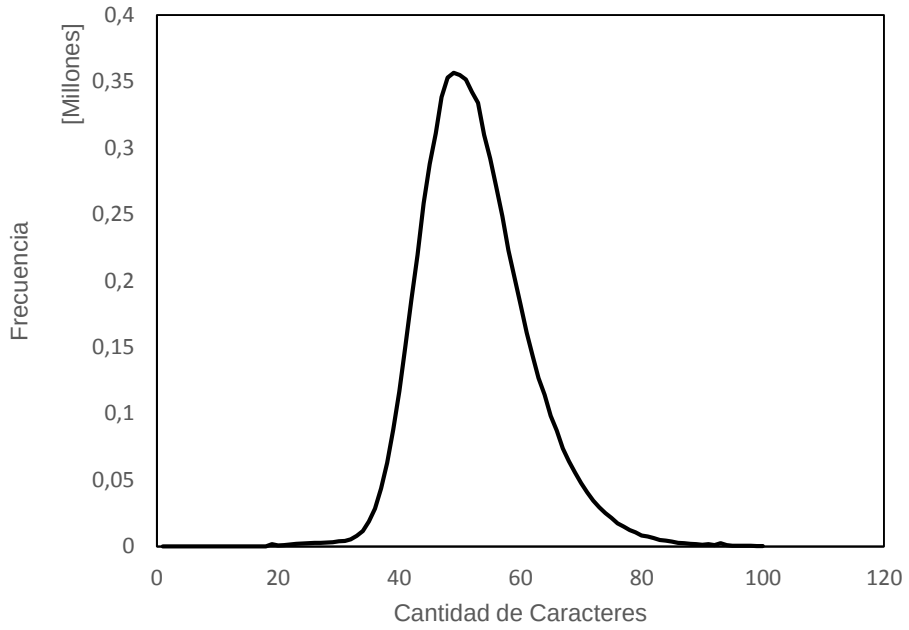


Anexo 7: Clase 8 de consultas – Fuente: Elaboración propia



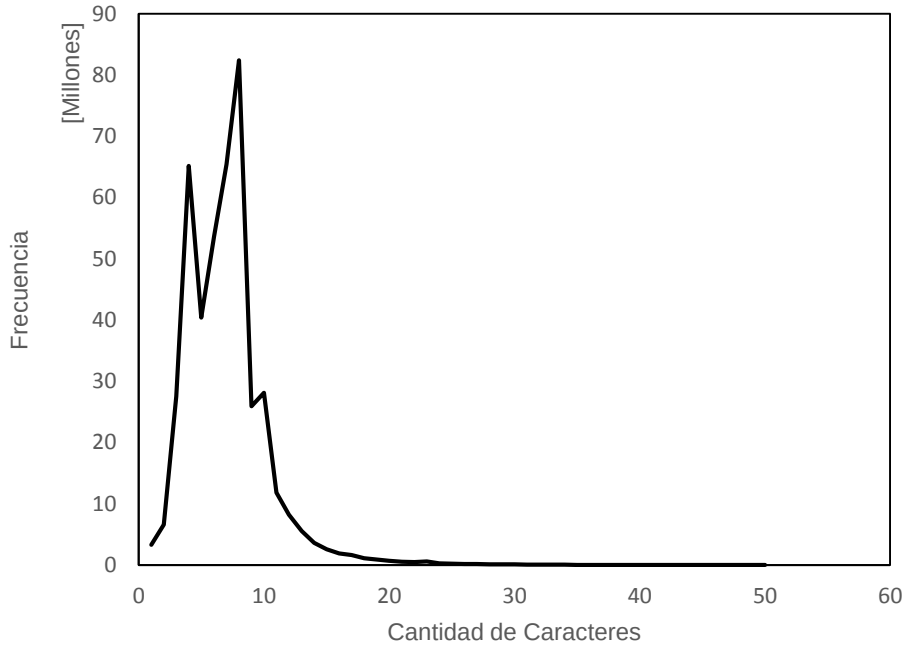
Anexo 8: Clase 9 de consultas – Fuente: Elaboración propia

.



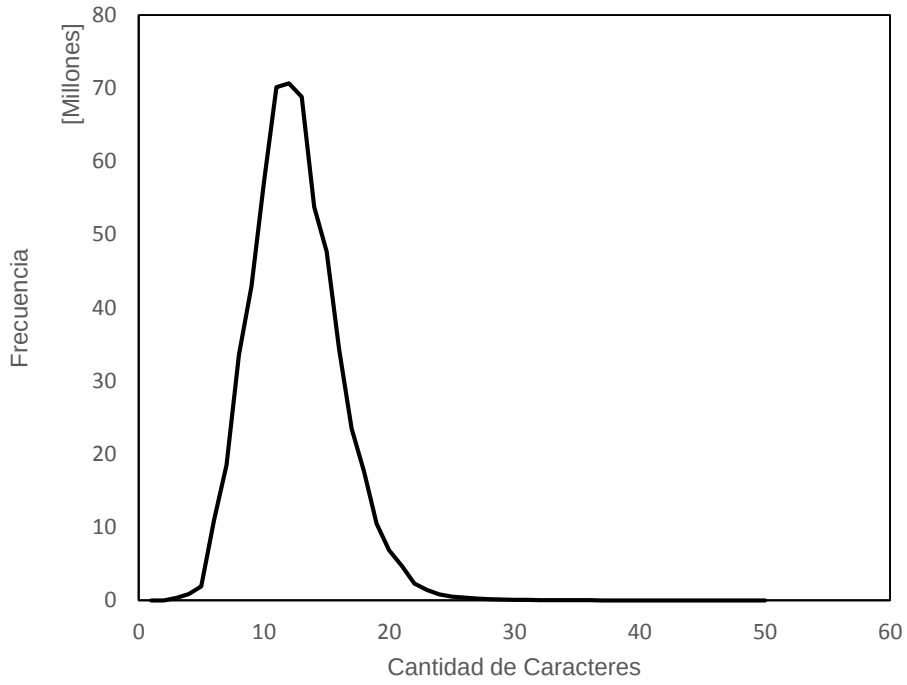
Anexo 9: Clase 10 de consultas – Fuente: Elaboración propia

- Log 201105

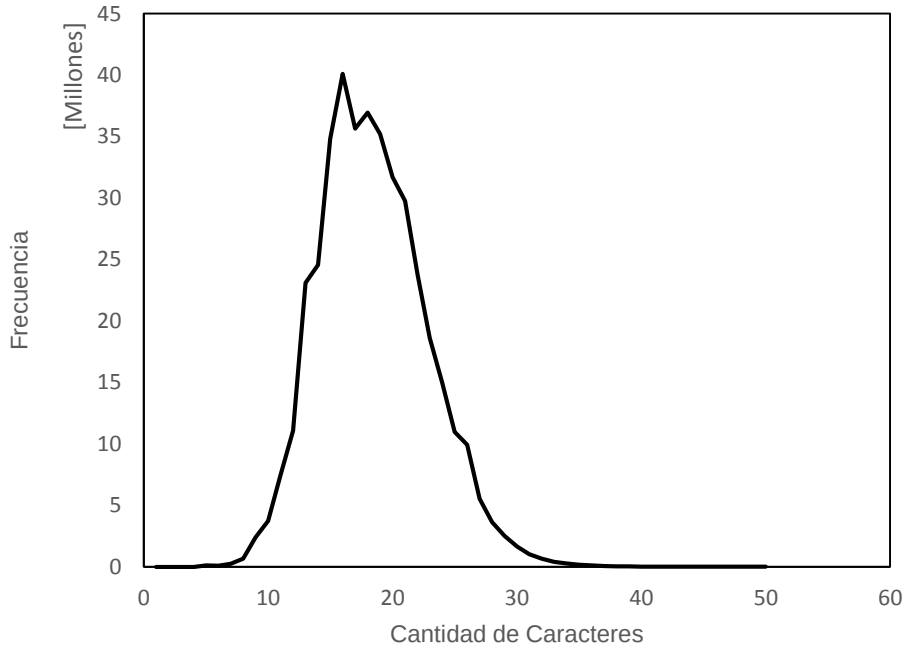


Anexo 10: Clase 1 de consultas – Log 201105 – Fuente: Elaboración propia

.

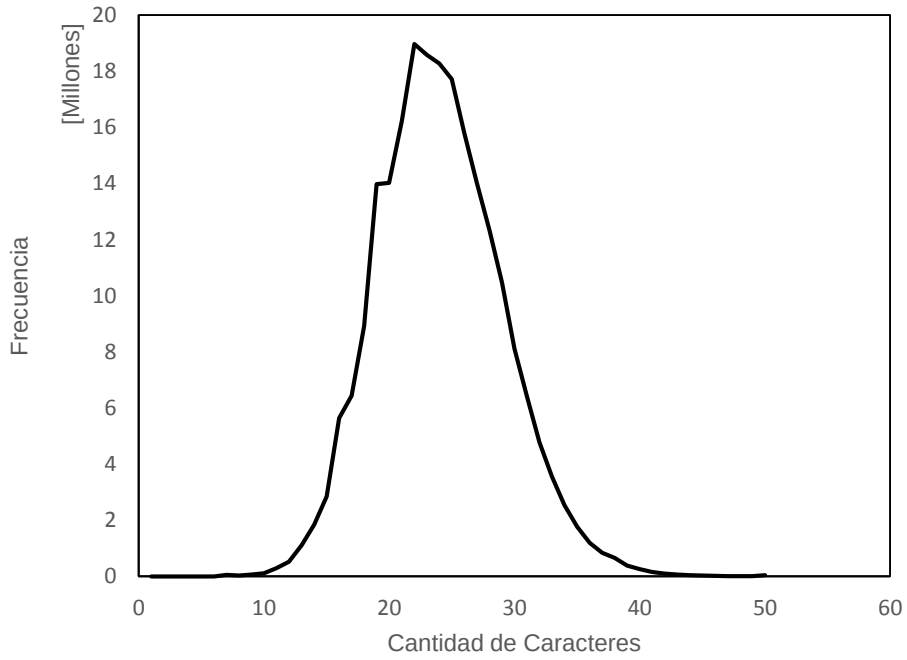


Anexo 11: Clase 2 de consultas – Log 201105 – Fuente: Elaboración propia

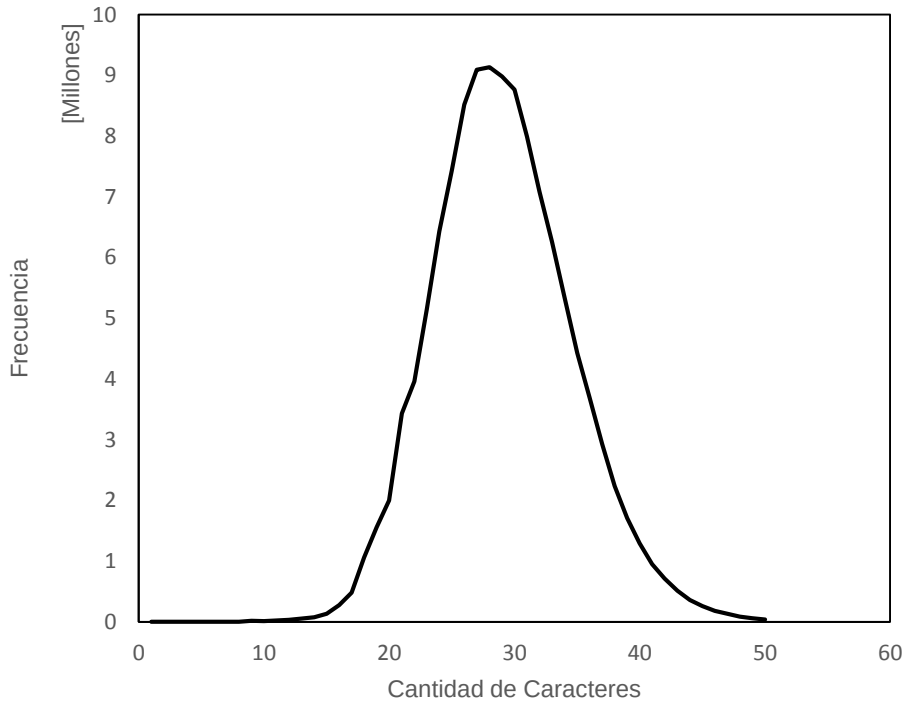


Anexo 12: Clase 3 de consultas – Log 201105 – Fuente: Elaboración propia

.

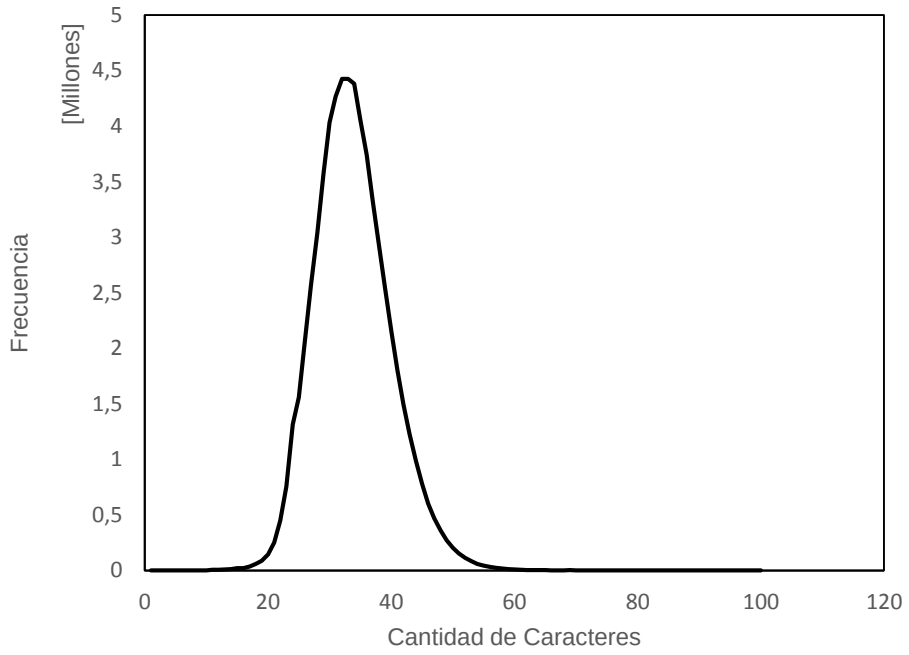


Anexo 13: Clase 4 de consultas – Log 201105 – Fuente: Elaboración propia

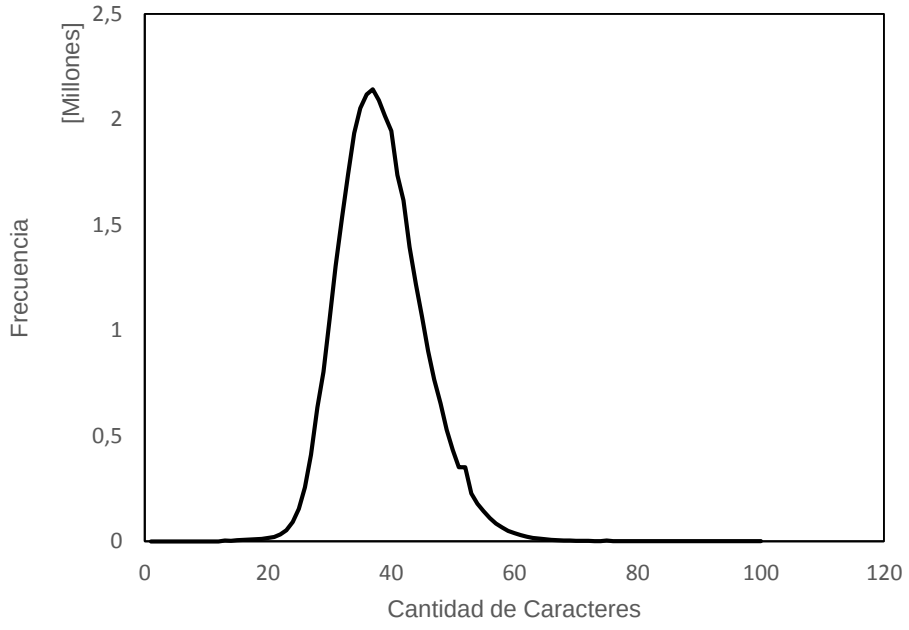


Anexo 14: Clase 5 de consultas – Log 201105 – Fuente: Elaboración propia

.

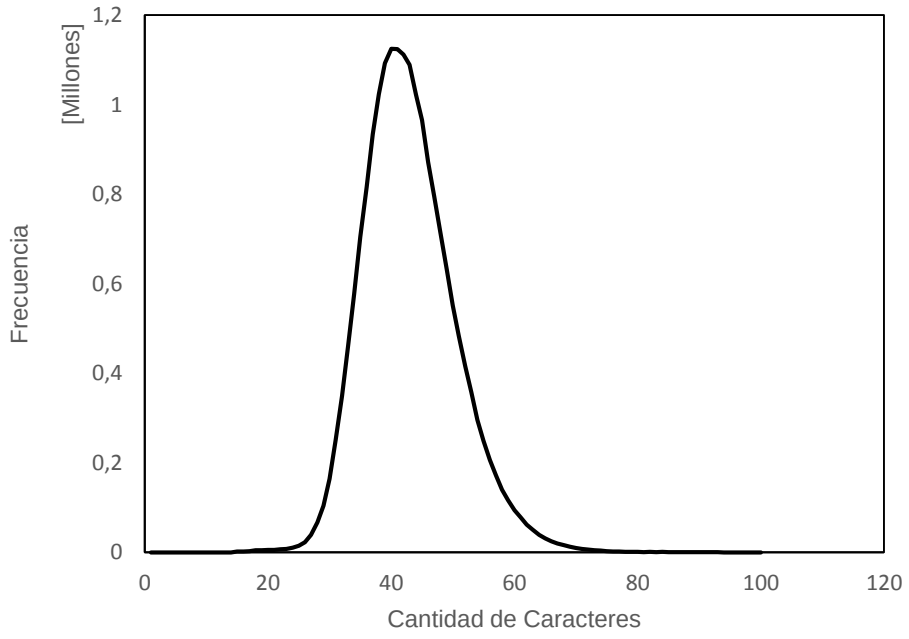


Anexo 15: Clase 6 de consultas – Log 201105 – Fuente: Elaboración propia

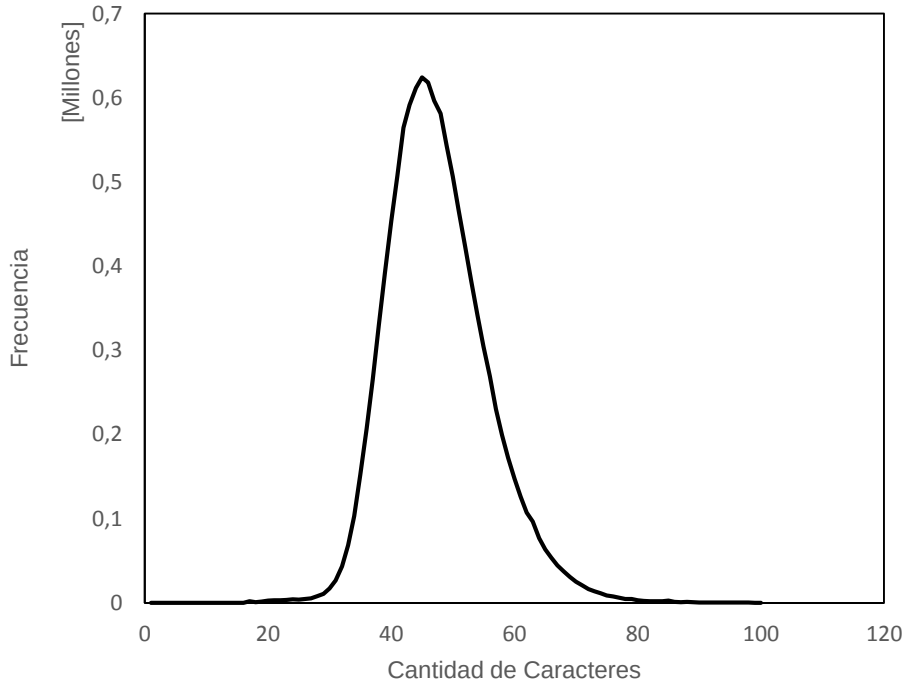


Anexo 16: Clase 7 de consultas – Log 201105 – Fuente: Elaboración propia

.

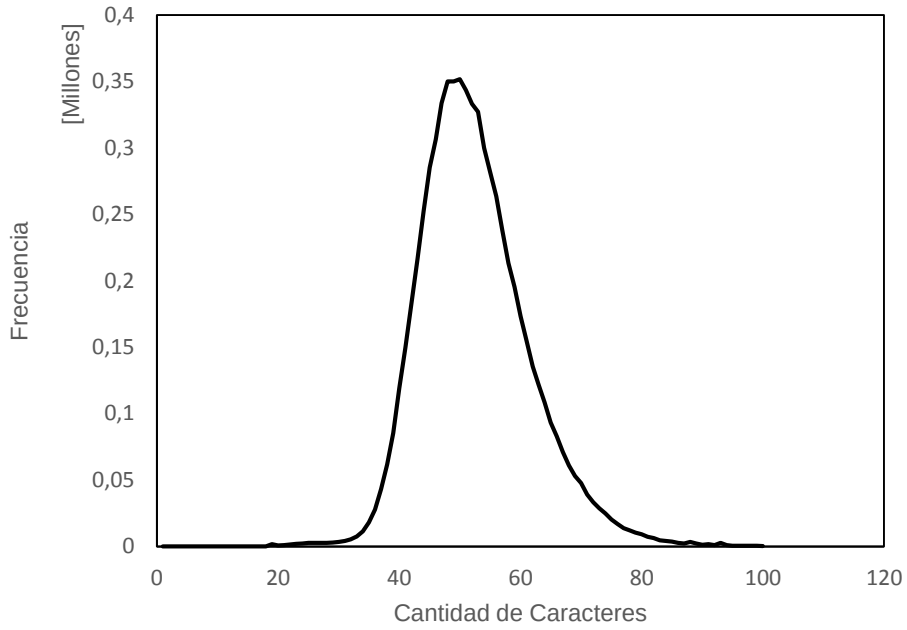


Anexo 17: Clase 8 de consultas – Log 201105 – Fuente: Elaboración propia



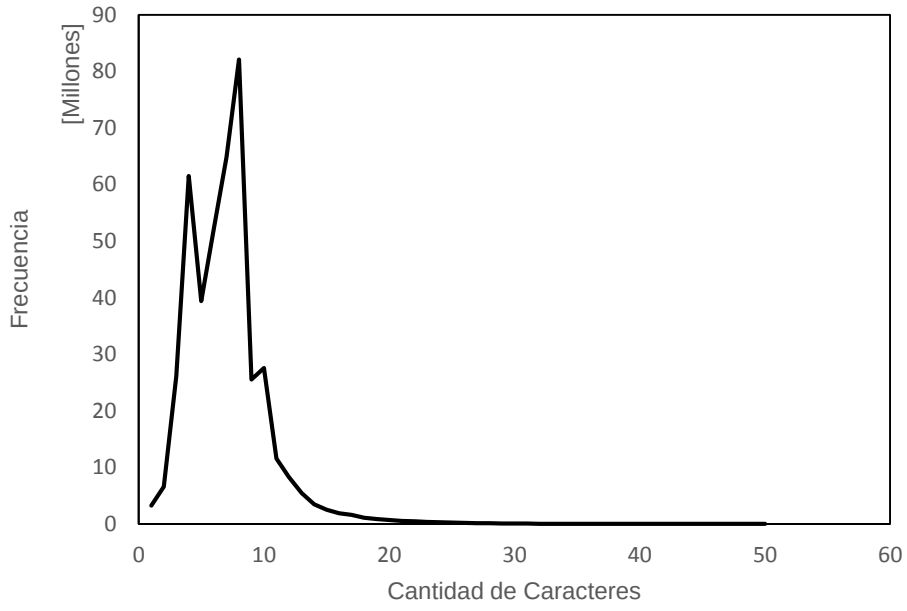
Anexo 18: Clase 9 de consultas – Log 201105 – Fuente: Elaboración propia

.



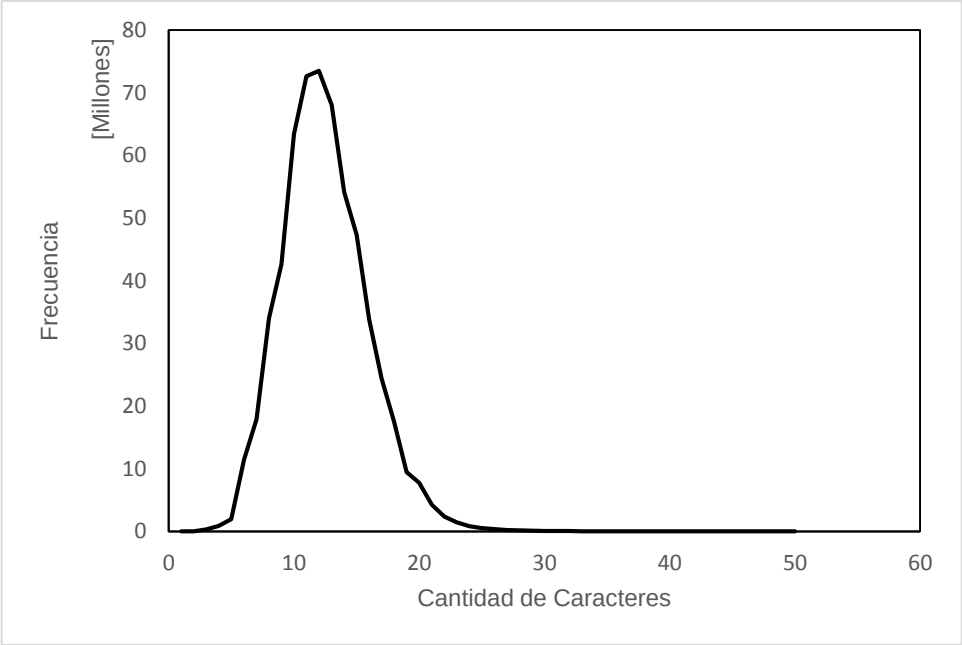
Anexo 19: Clase 10 de consultas – Log 201105 – Fuente: Elaboración propia

- Log 201106

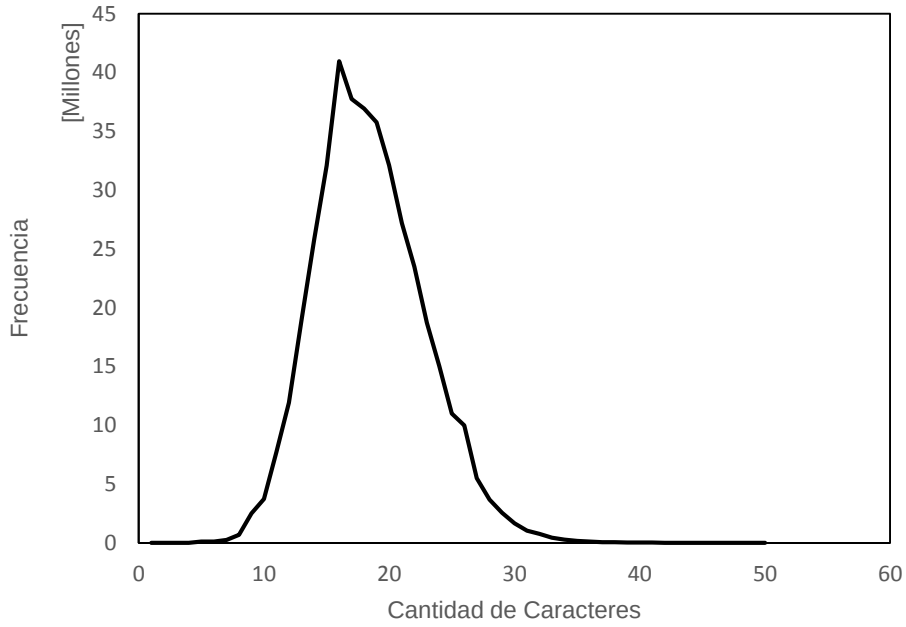


Anexo 20: Clase 1 de consultas – Log 201106 – Fuente: Elaboración propia

.

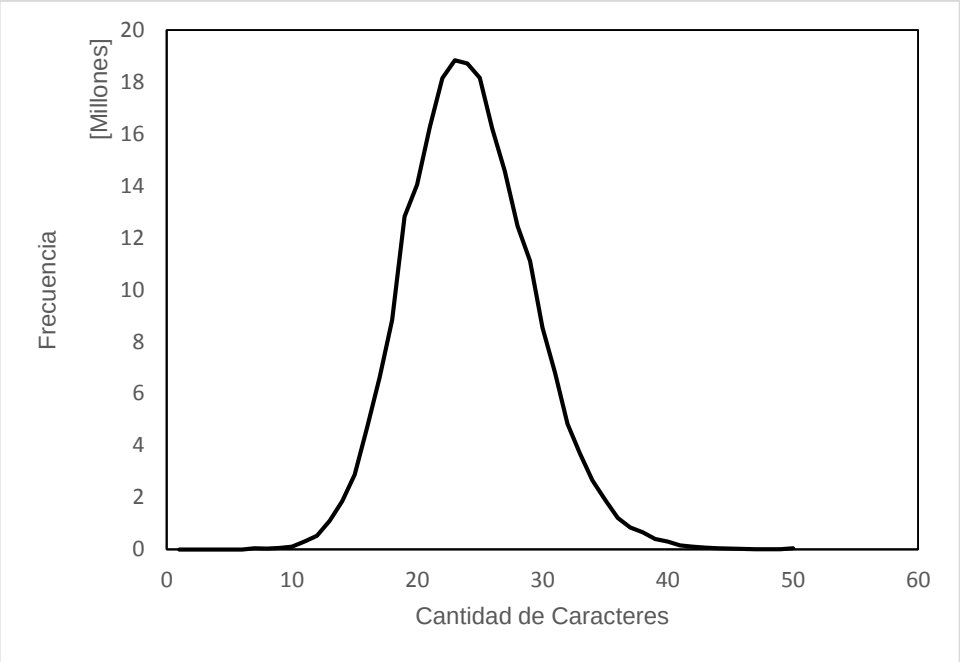


Anexo 21: Clase 2 de consultas – Log 201106 – Fuente: Elaboración propia

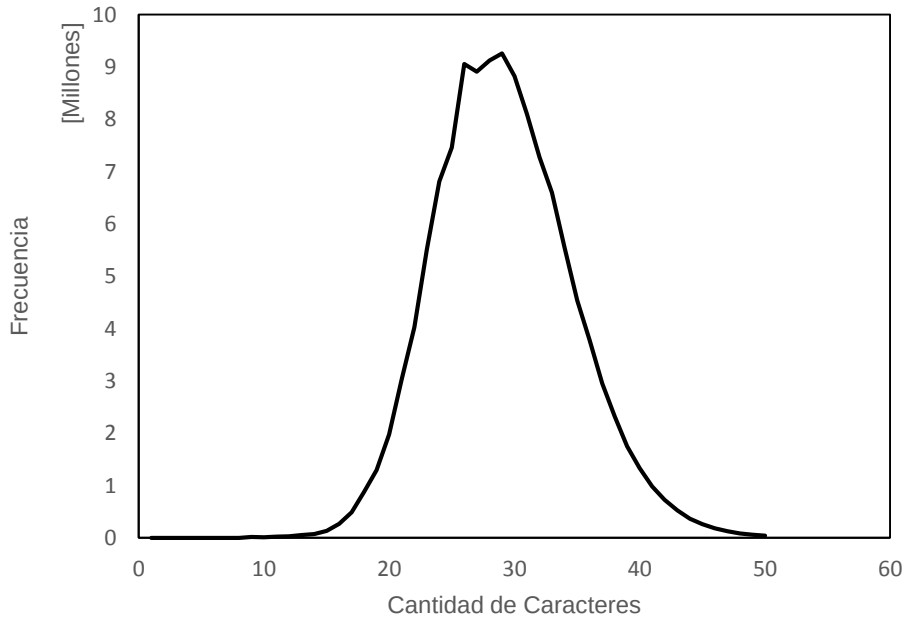


Anexo 22: Clase 3 de consultas – Log 201106 – Fuente: Elaboración propia

.

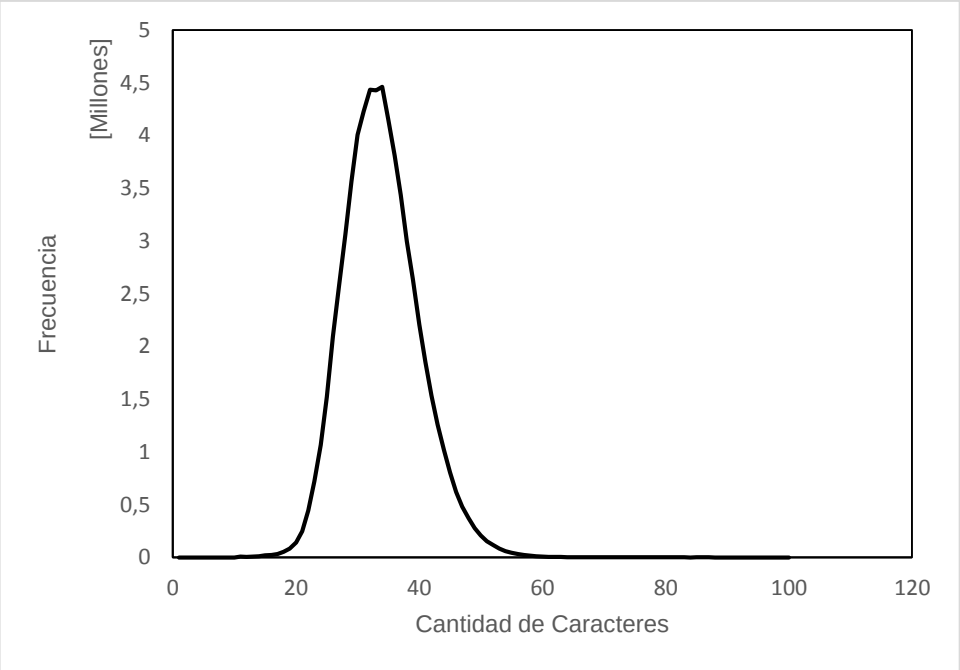


Anexo 23: Clase 4 de consultas – Log 201106 – Fuente: Elaboración propia

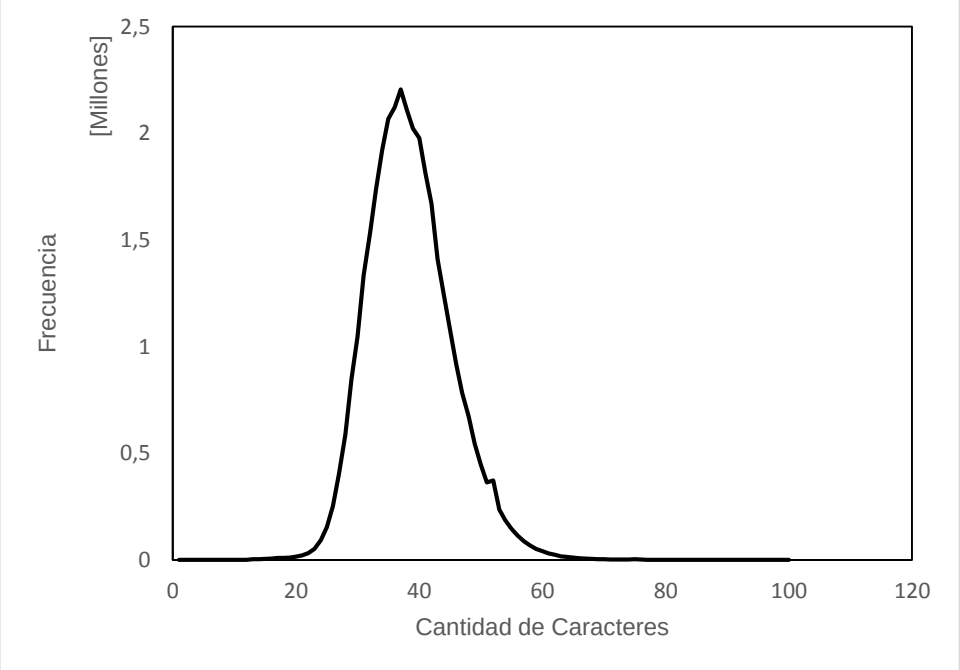


Anexo 24: Clase 5 de consultas – Log 201106 – Fuente: Elaboración propia

.

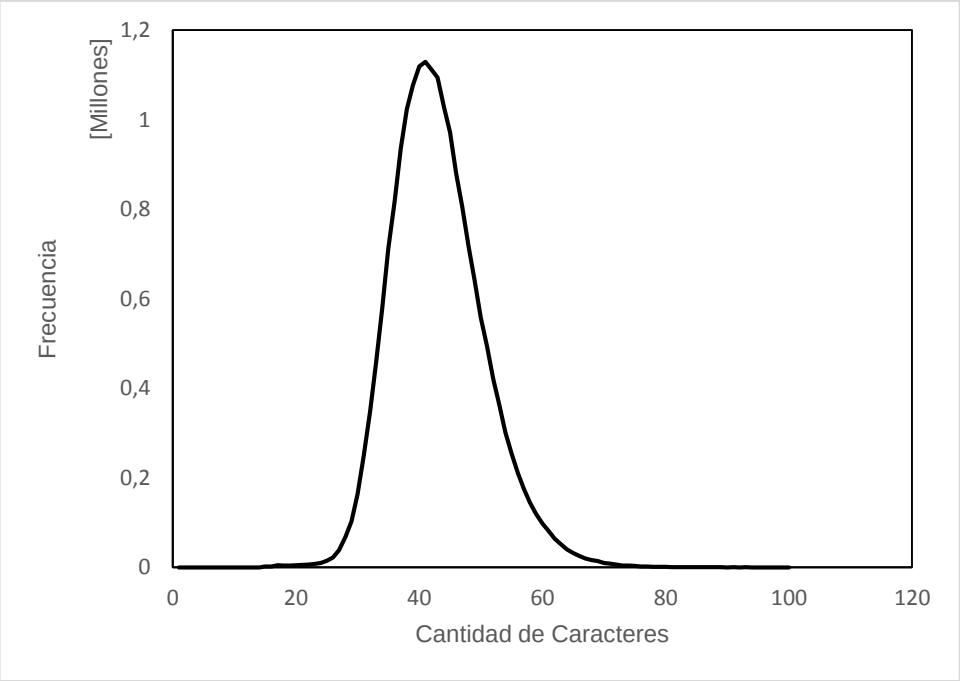


Anexo 25: Clase 6 de consultas – Log 201106 – Fuente: Elaboración propia

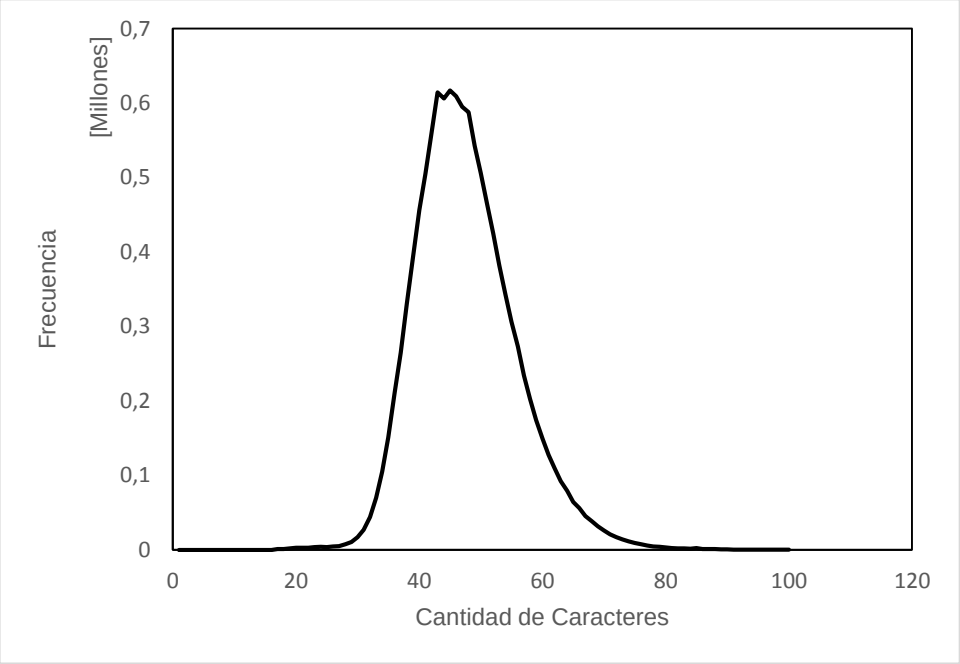


Anexo 26: Clase 7 de consultas – Log 201106 – Fuente: Elaboración propia

.

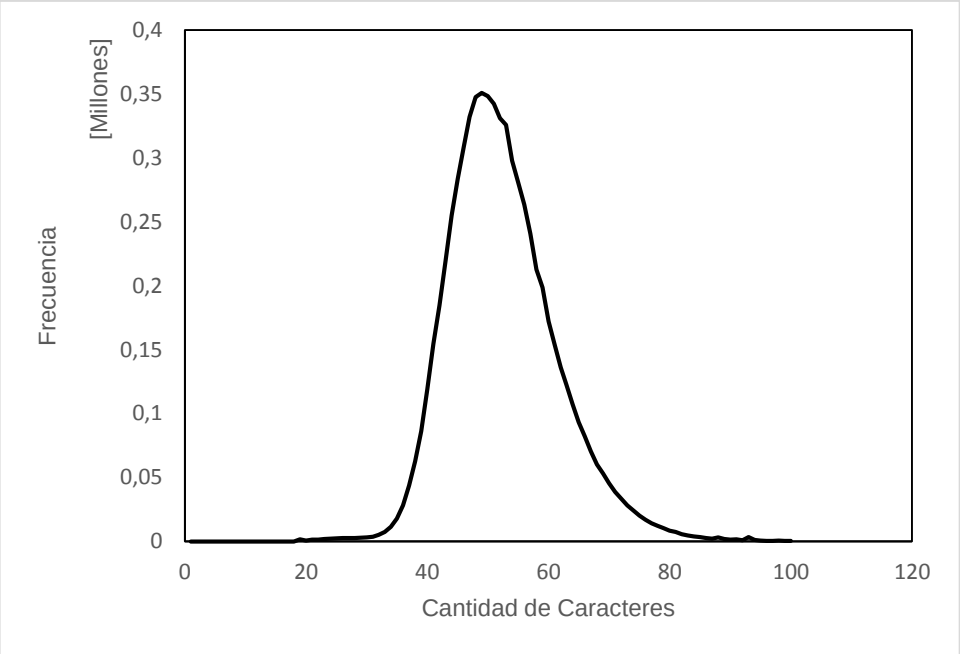


Anexo 27: Clase 8 de consultas – Log 201106 – Fuente: Elaboración propia



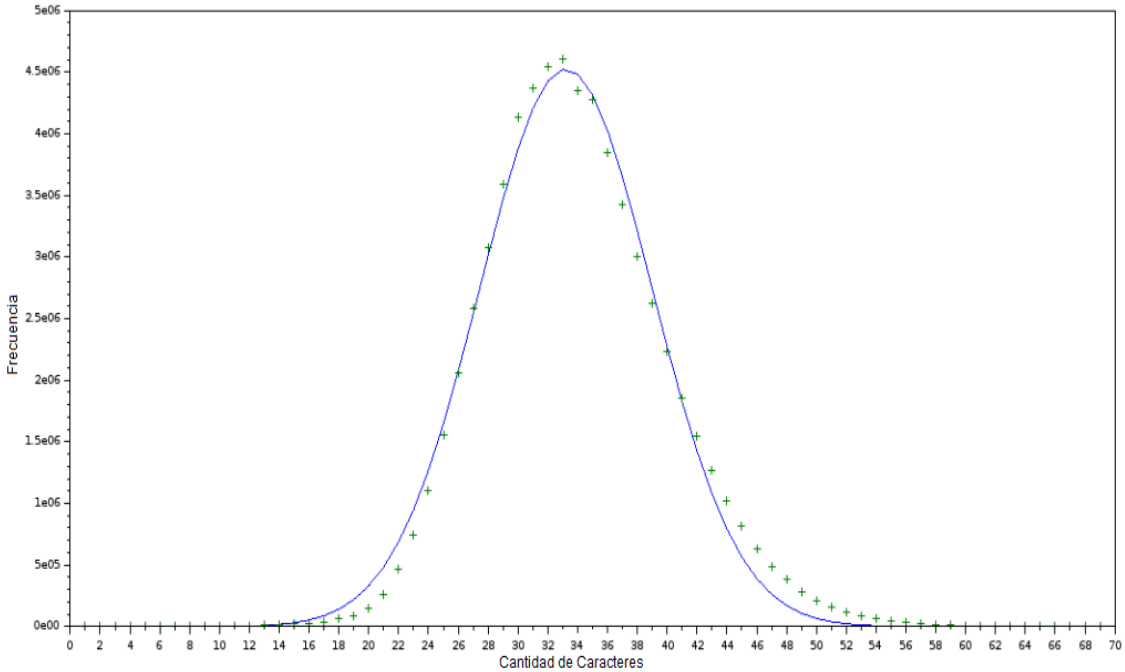
Anexo 28: Clase 9 de consultas – Log 201106 – Fuente: Elaboración propia

.

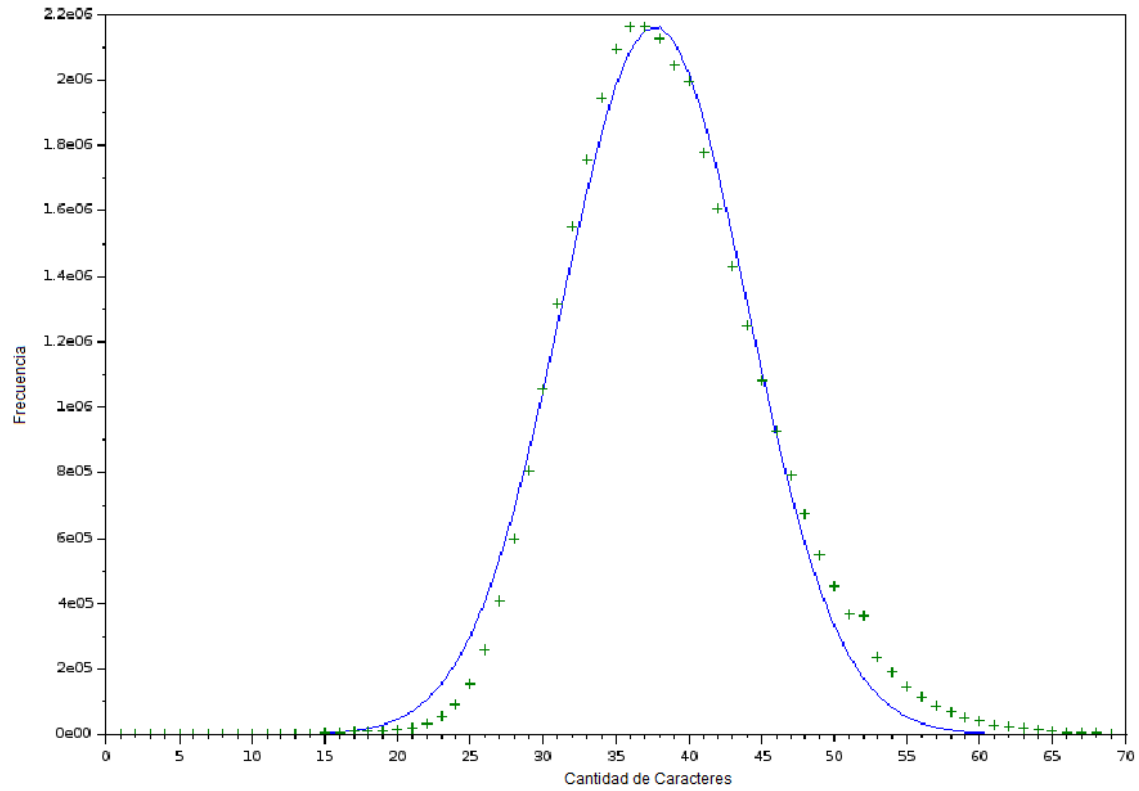


Anexo 29: Clase 10 de consultas – Log 201106 – Fuente: Elaboración propia

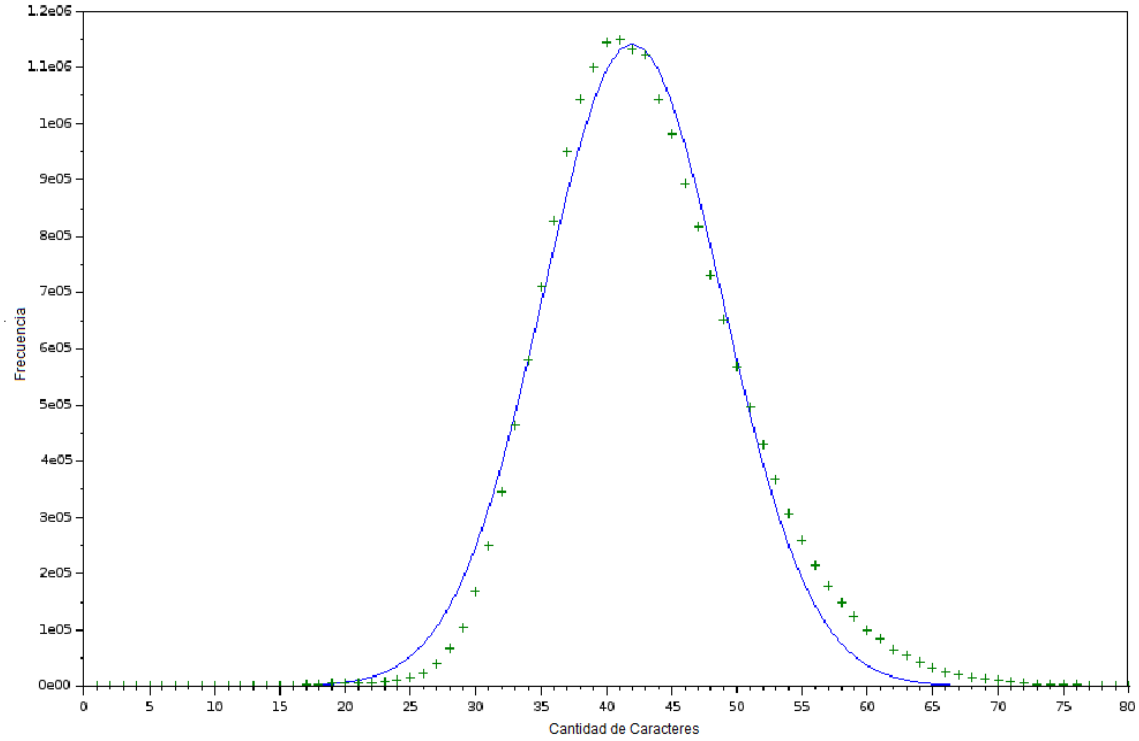
9.4 Anexo D



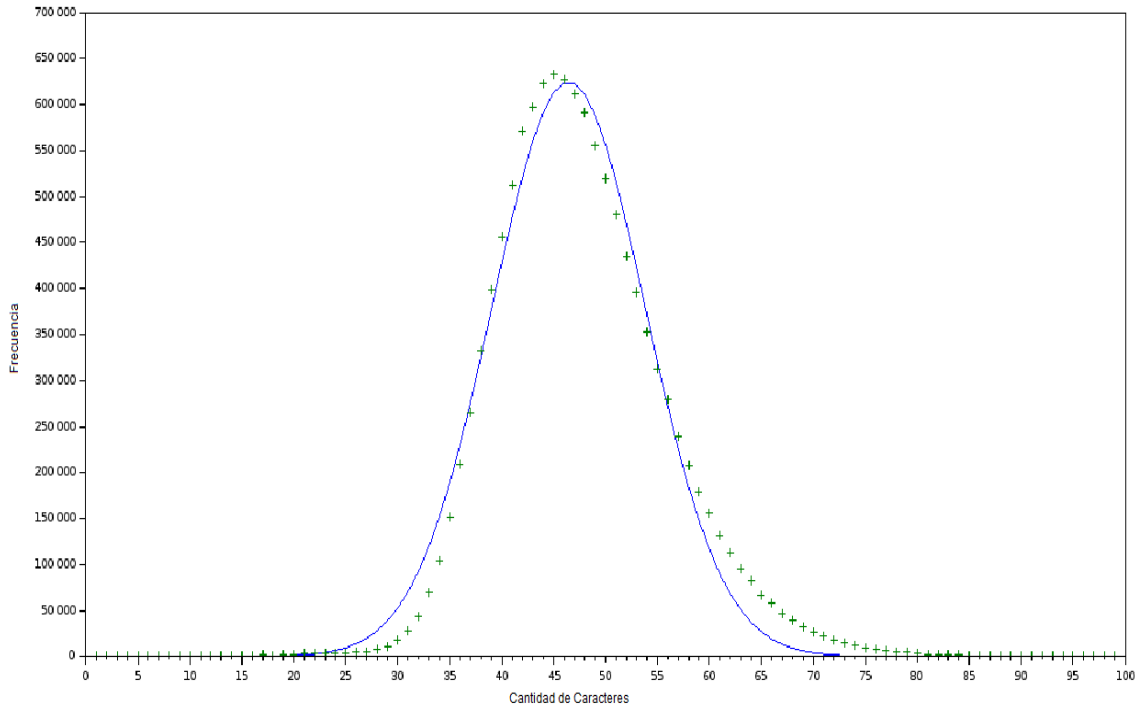
Anexo 30: Clase 6 de consultas – corregida - Fuente: Elaboración propia



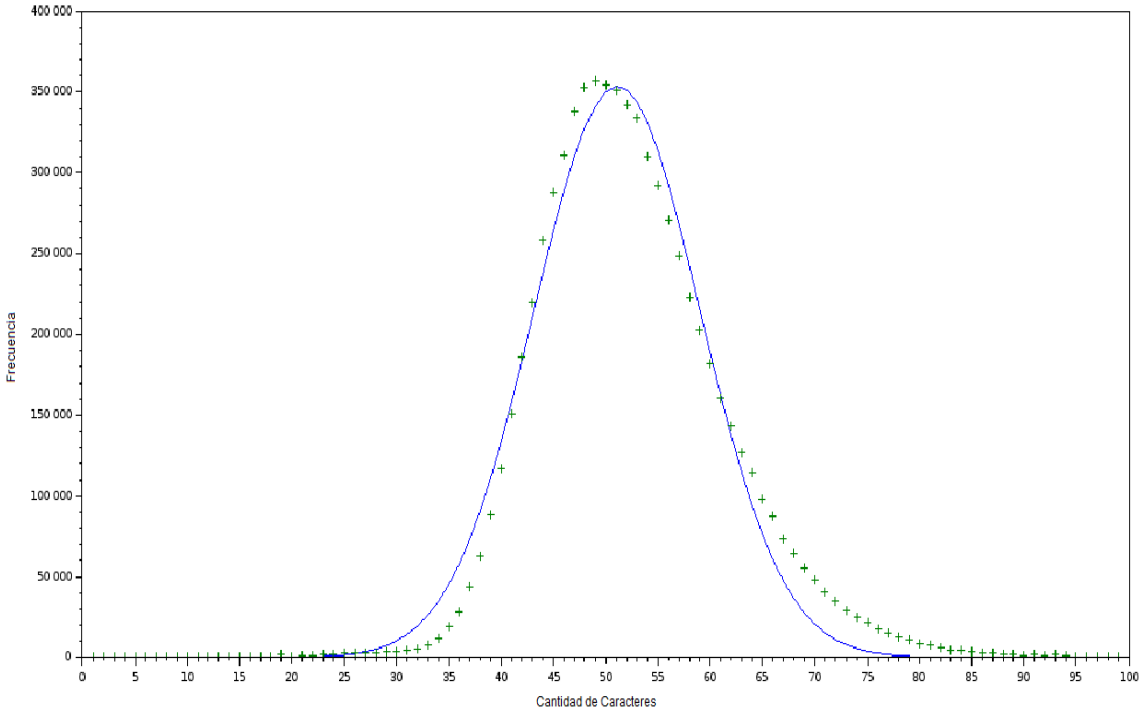
Anexo 31: Clase 7 de consultas – corregida - Fuente: Elaboración propia



Anexo 32: Clase 8 de consultas – corregida - Fuente: Elaboración propia



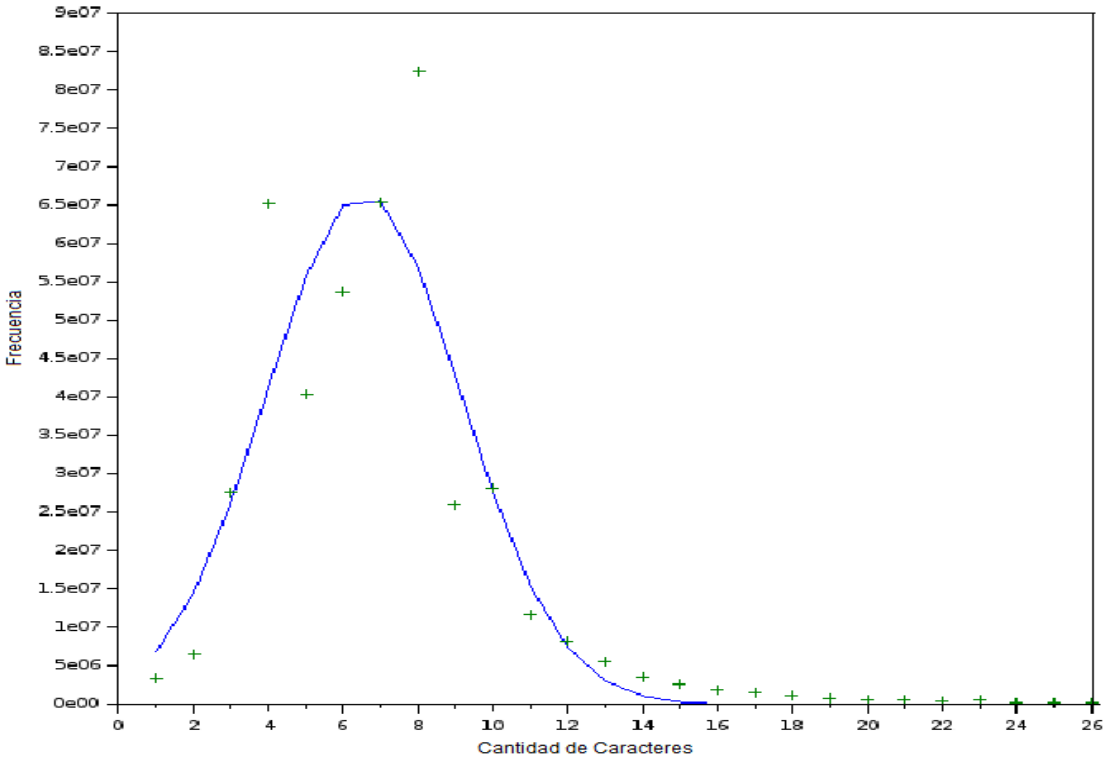
Anexo 33: Clase 9 de consultas – corregida - Fuente: Elaboración propia



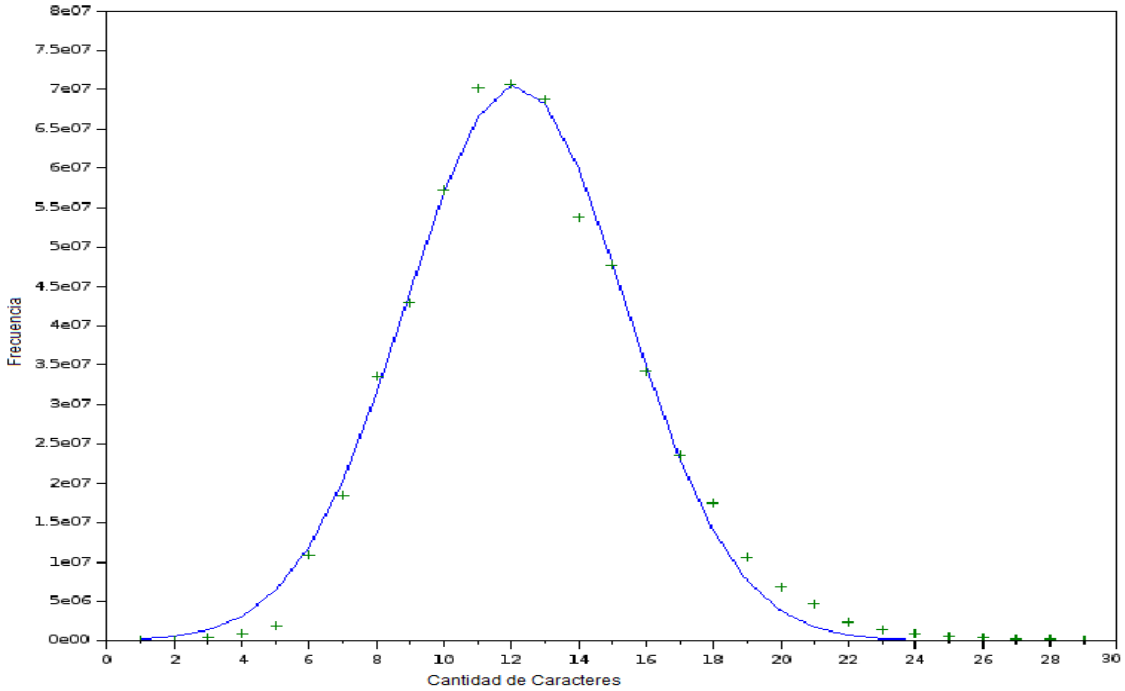
Anexo 34: Clase 10 de consultas – corregida - Fuente: Elaboración propia

.

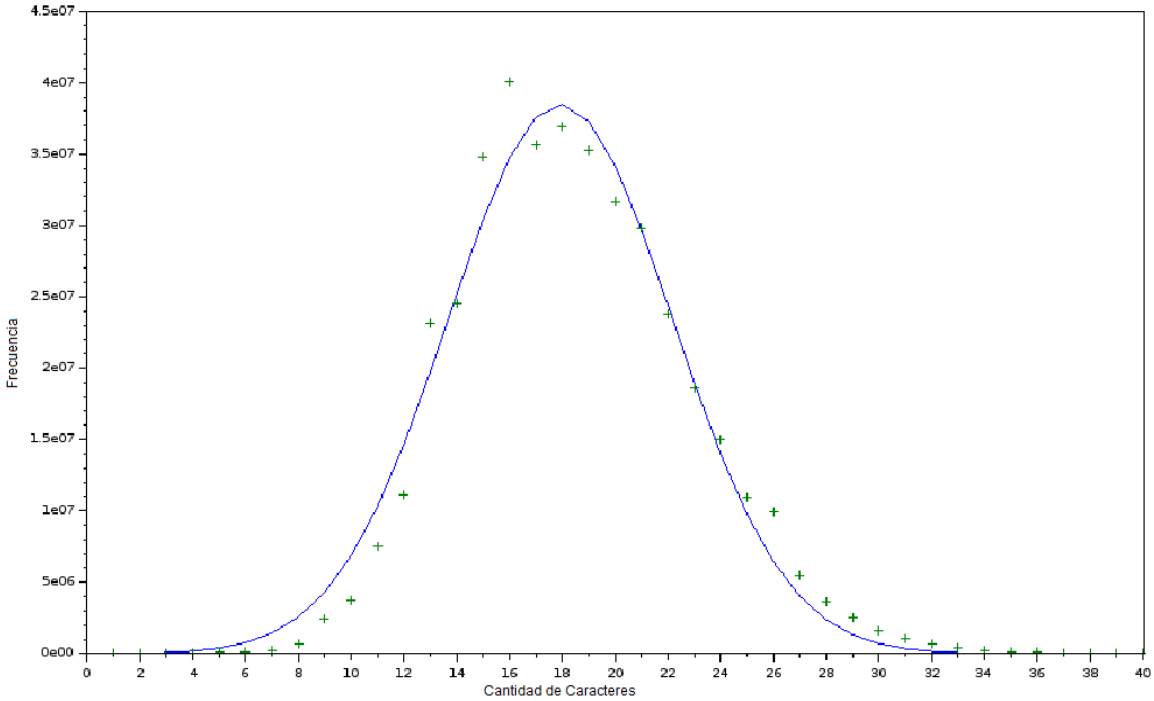
- Log 201105



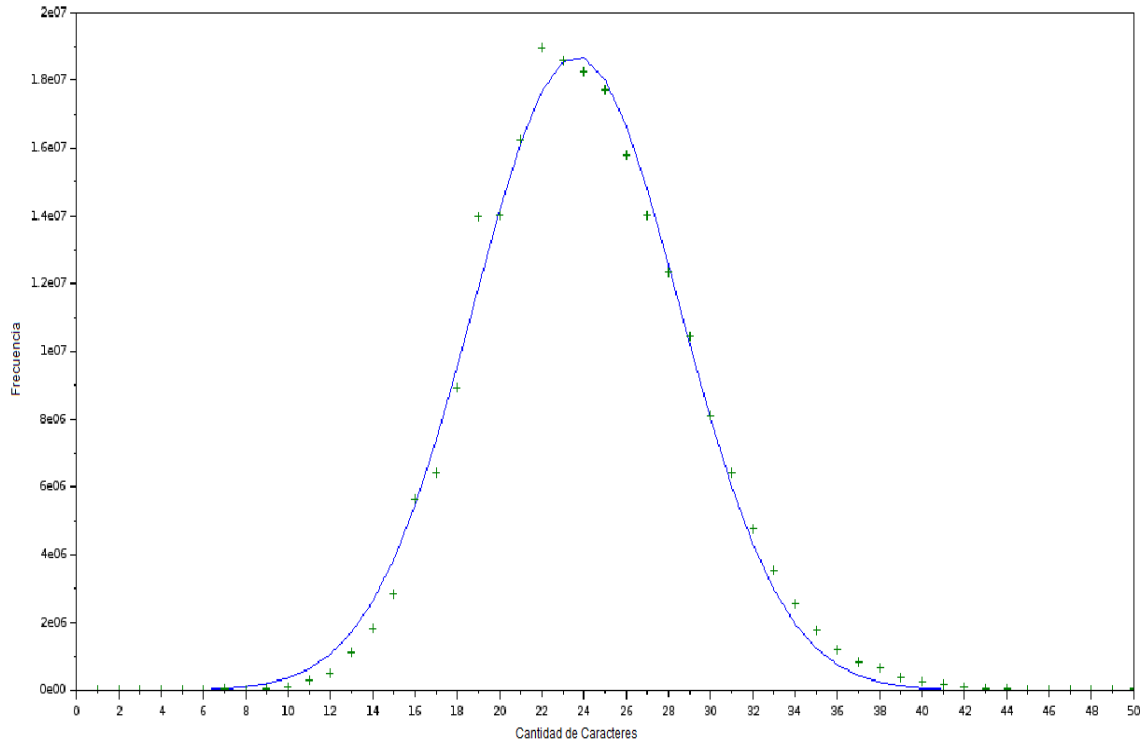
Anexo 35: Clase 1 de consultas – Log 201105 – corregida - Fuente: Elaboración propia



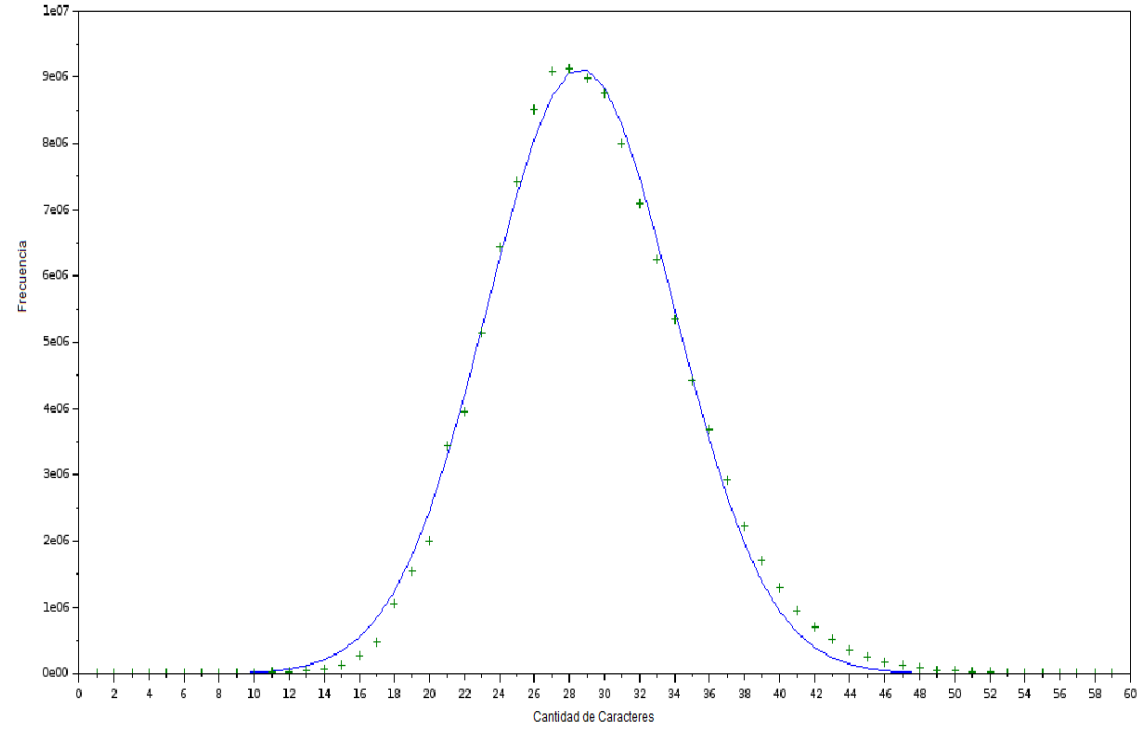
Anexo 36: Clase 2 de consultas – Log 201105 – corregida - Fuente: Elaboración propia



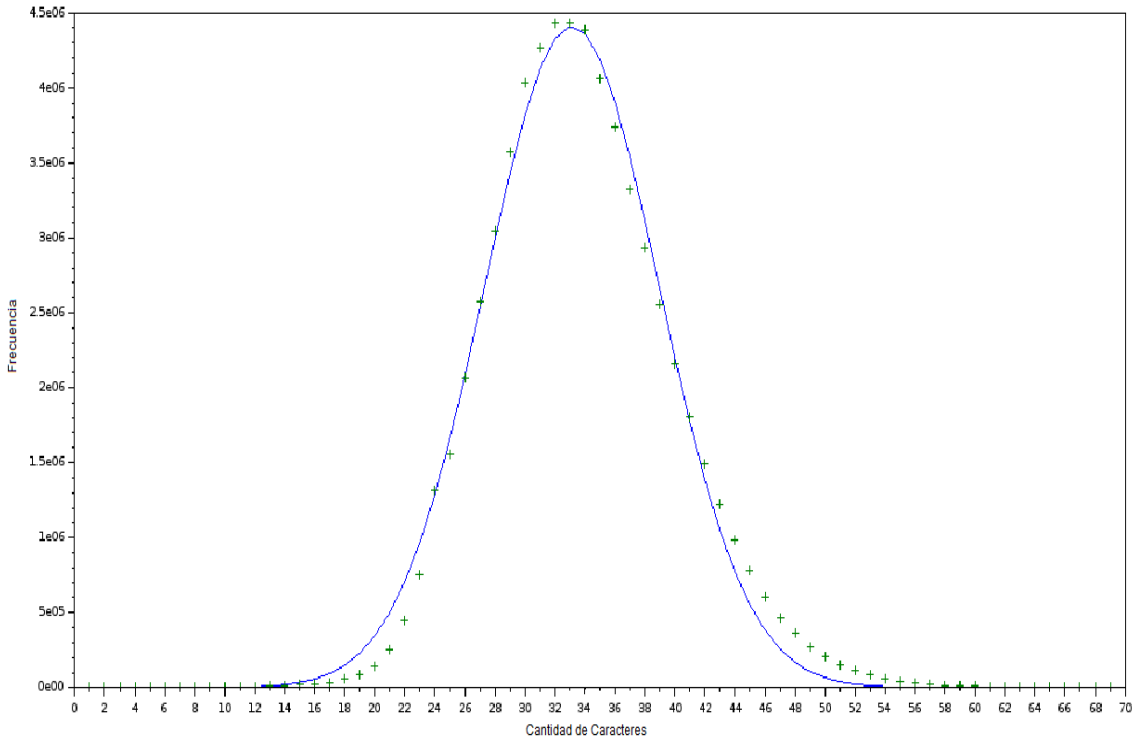
Anexo 37: Clase 3 de consultas – Log 201105 – corregida - Fuente: Elaboración propia



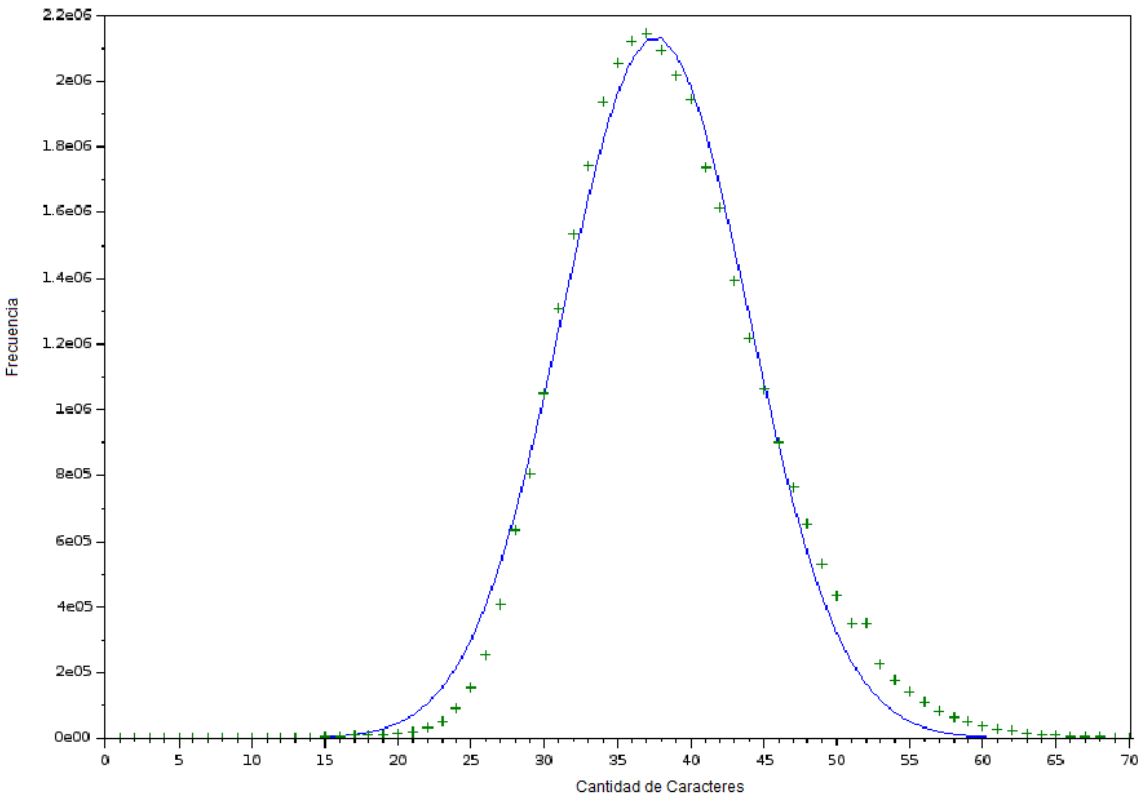
Anexo 38: Clase 4 de consultas – Log 201105 – corregida - Fuente: Elaboración propia



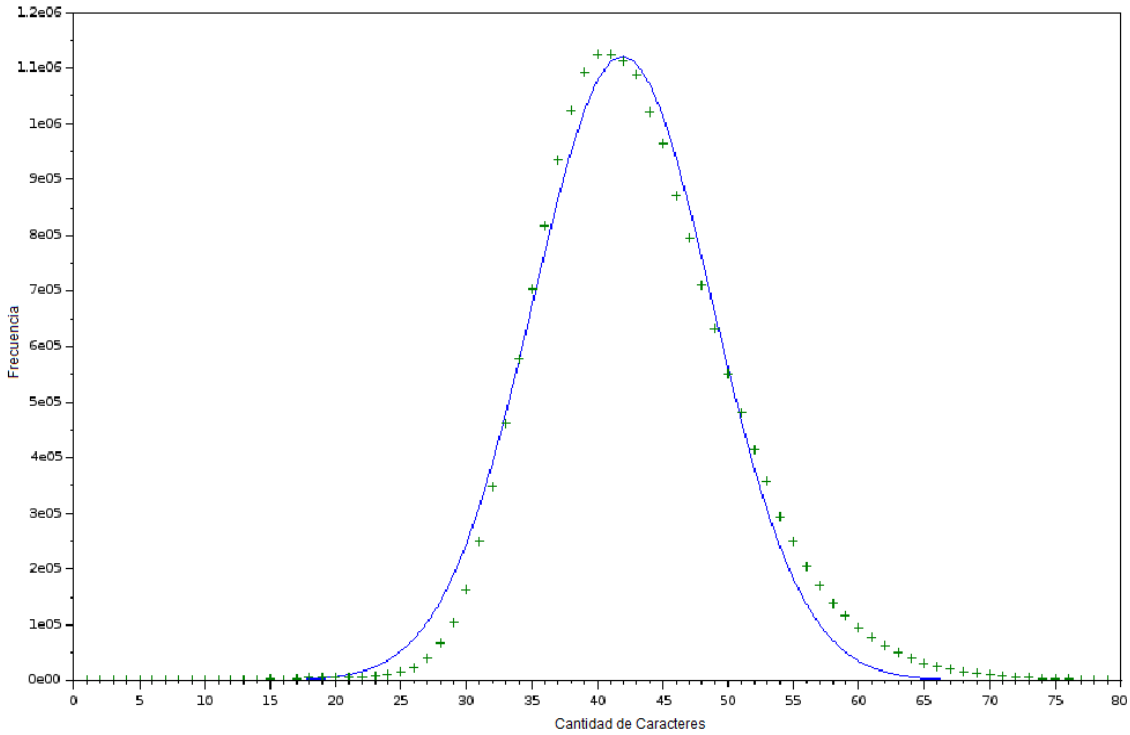
Anexo 39: Clase 5 de consultas – Log 201105 – corregida - Fuente: Elaboración propia



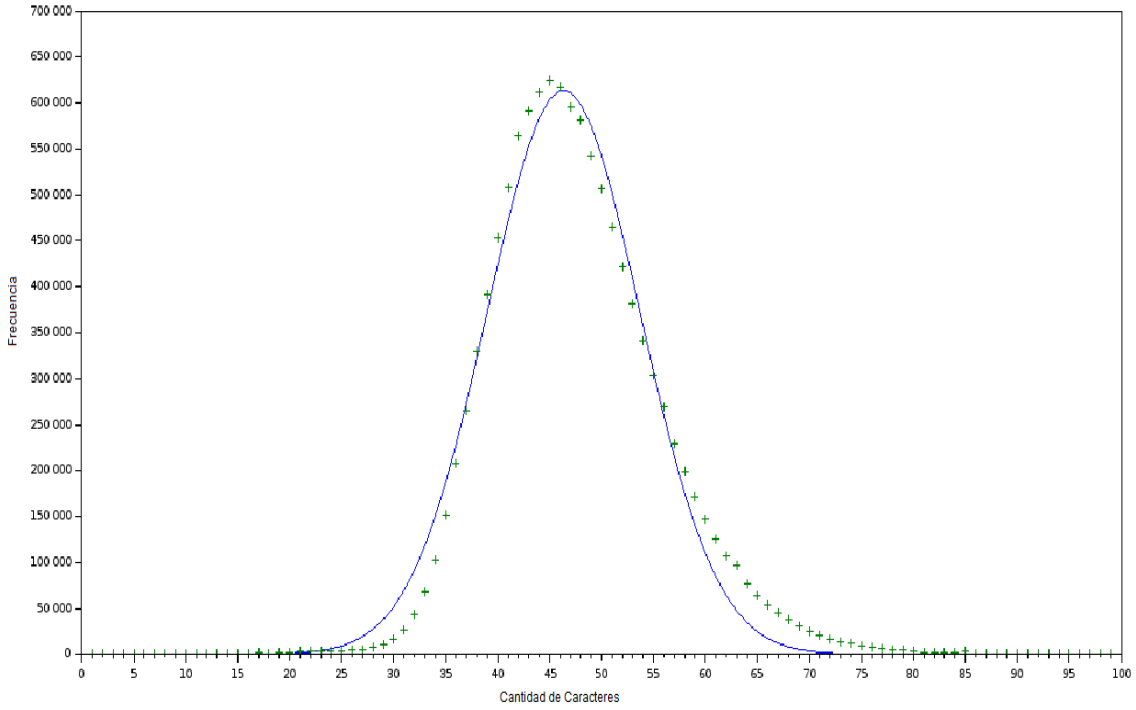
Anexo 40: Clase 6 de consultas – Log 201105 – corregida - Fuente: Elaboración propia



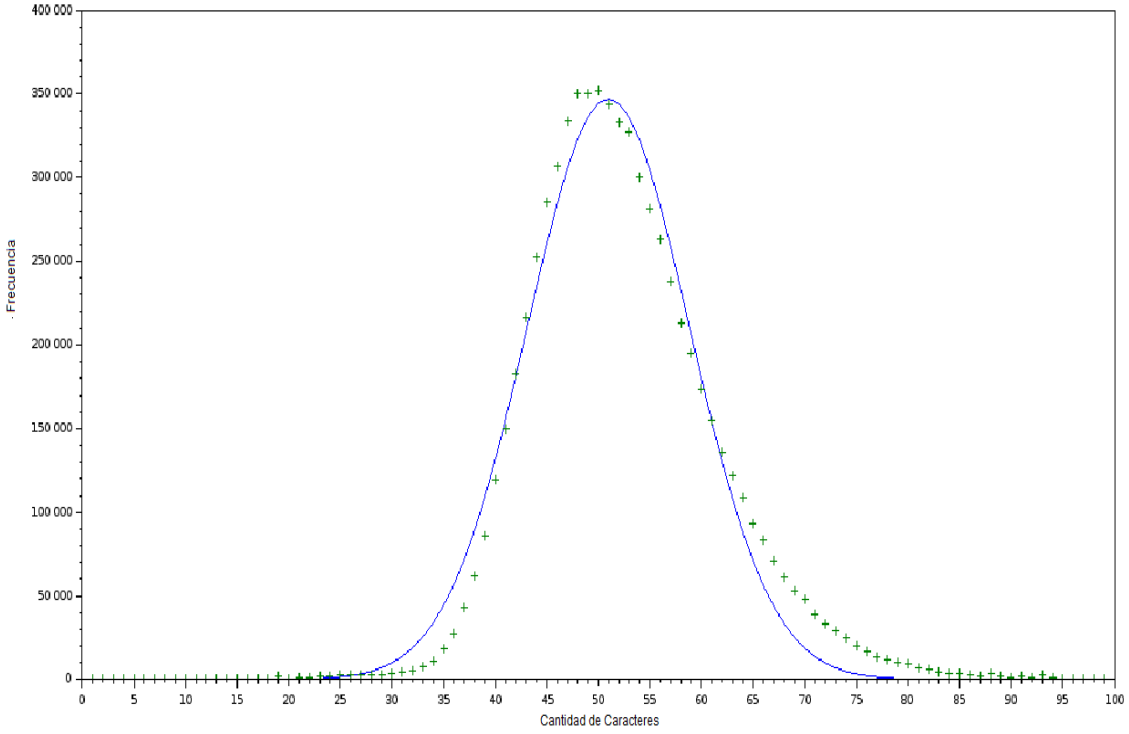
Anexo 41: Clase 7 de consultas – Log 201105 – corregida - Fuente: Elaboración propia



Anexo 42: Clase 8 de consultas – Log 201105 – corregida - Fuente: Elaboración propia



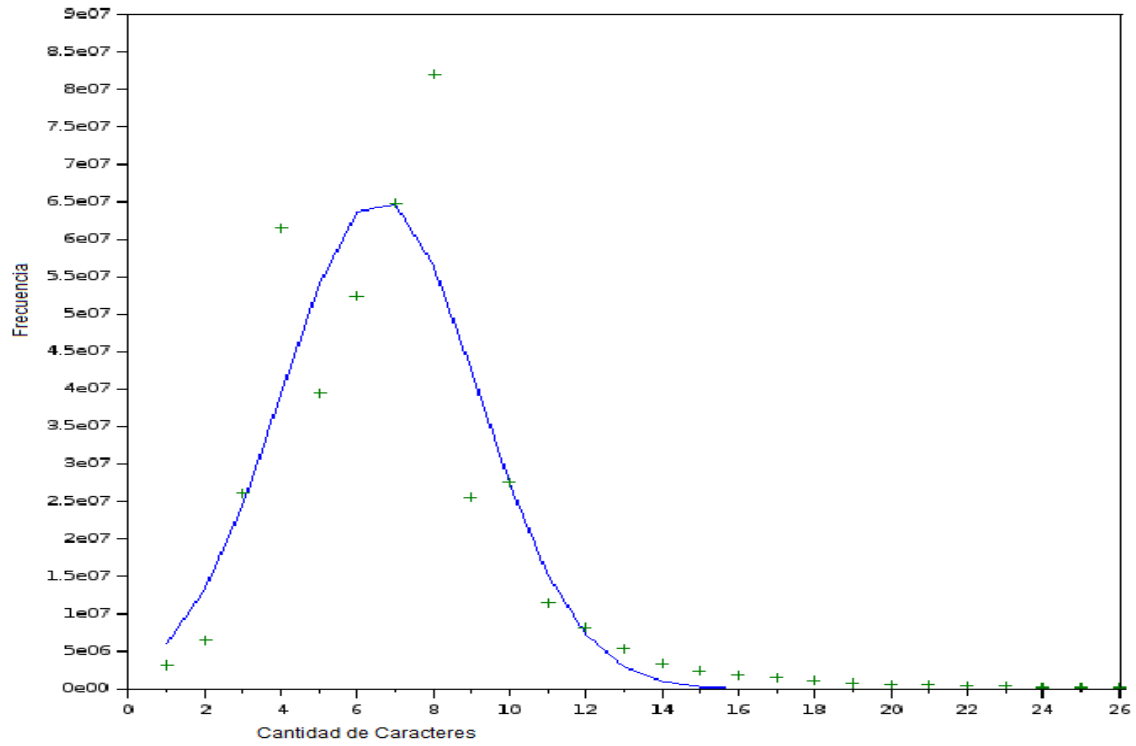
Anexo 43: Clase 9 de consultas – Log 201105 – corregida - Fuente: Elaboración propia



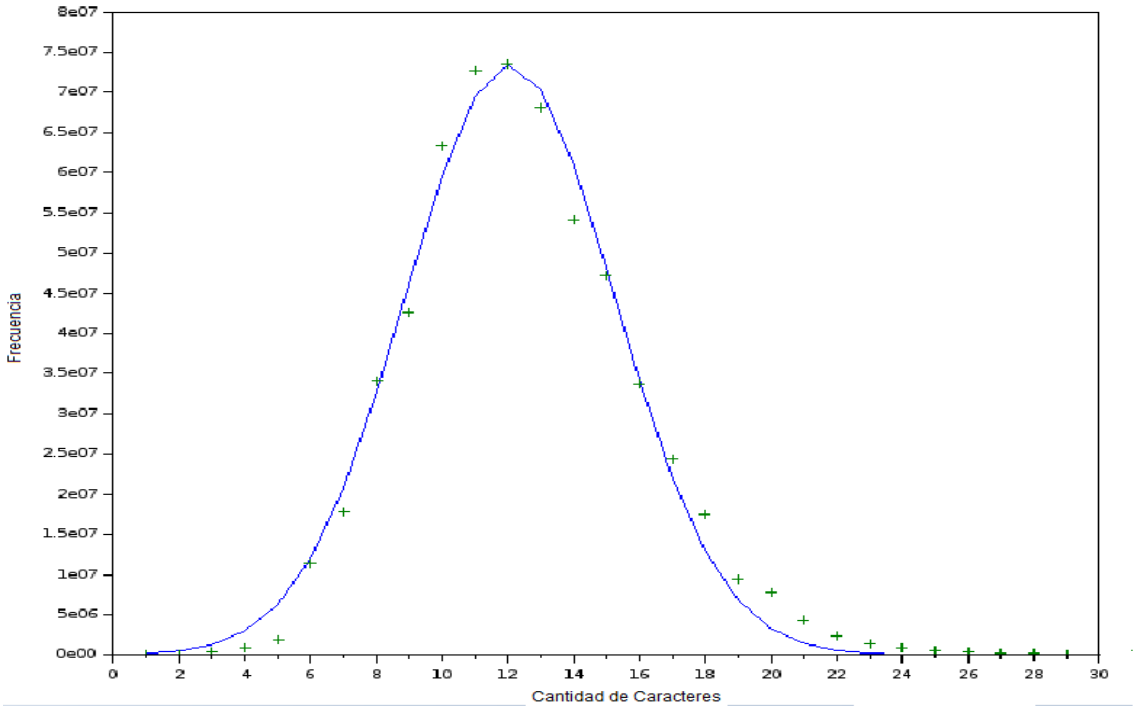
Anexo 44: Clase 10 de consultas – Log 201105 – corregida - Fuente: Elaboración propia

.

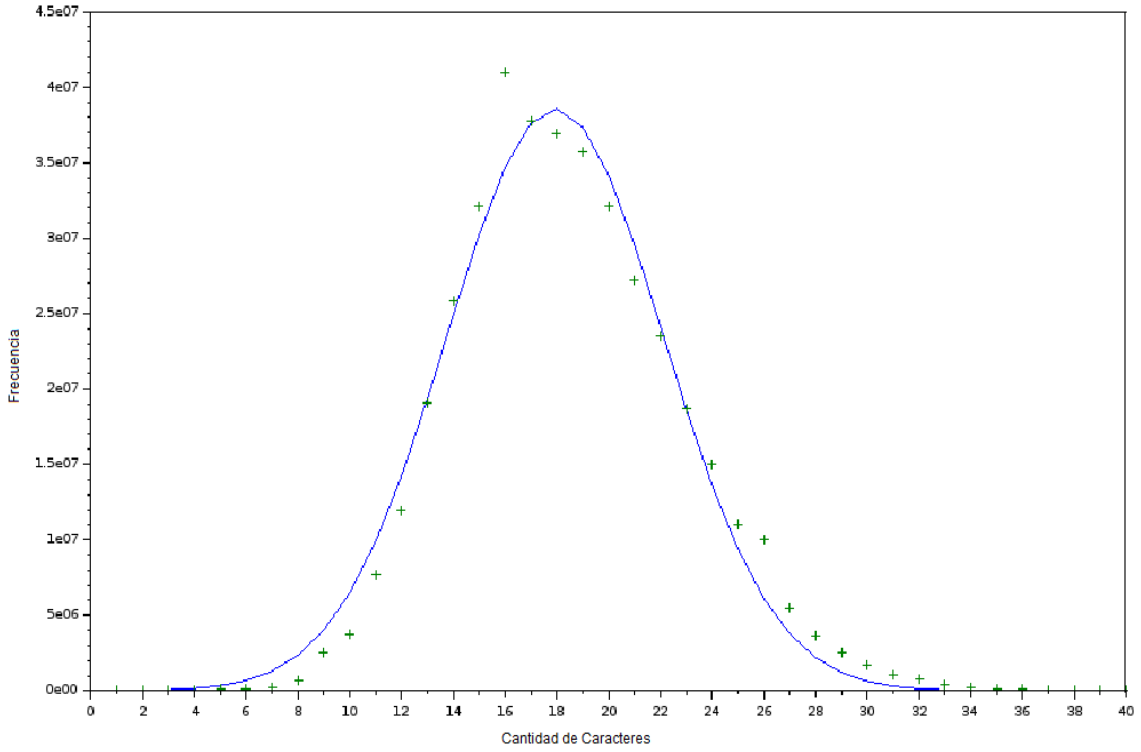
- Log 201106



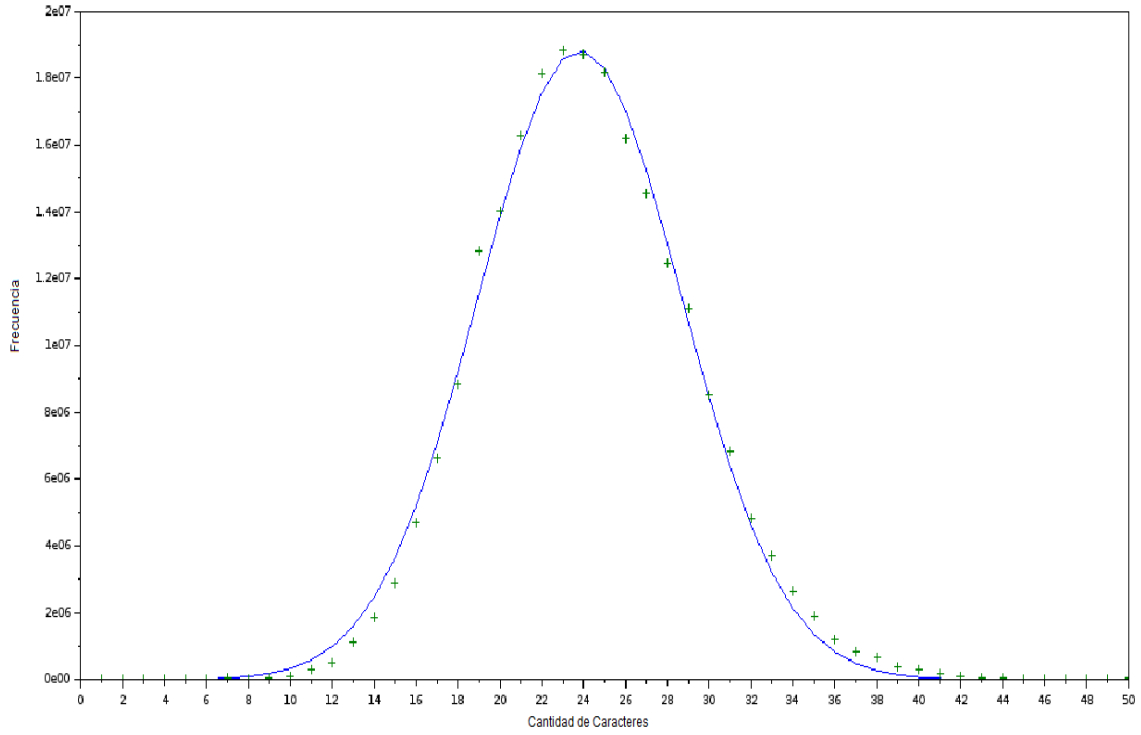
Anexo 45: Clase 1 de consultas – Log 201106 – corregida - Fuente: Elaboración propia



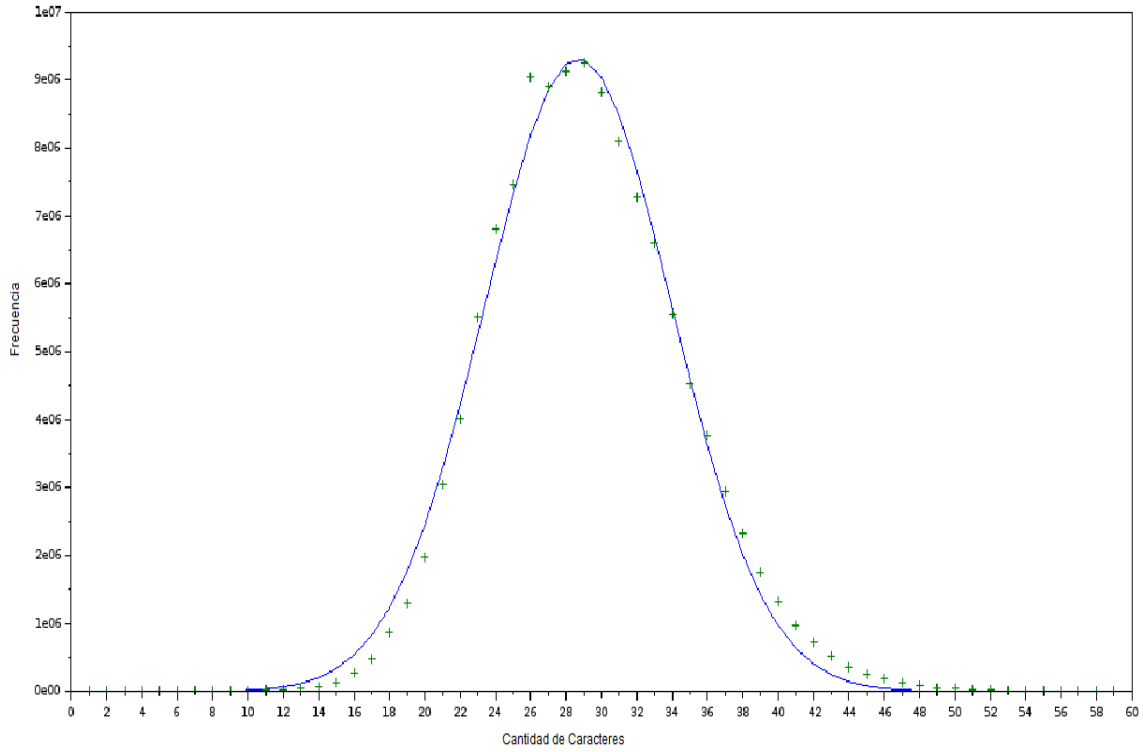
Anexo 46: Clase 2 de consultas – Log 201106 – corregida - Fuente: Elaboración propia



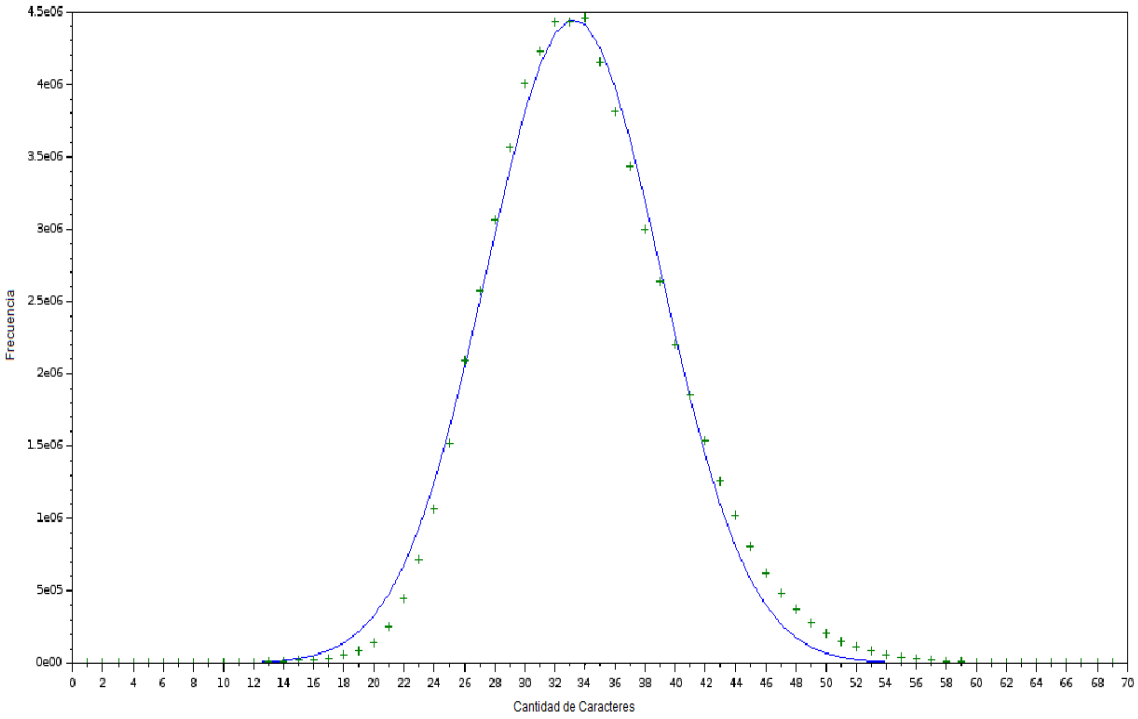
Anexo 47: Clase 3 de consultas – Log 201106 – corregida - Fuente: Elaboración propia



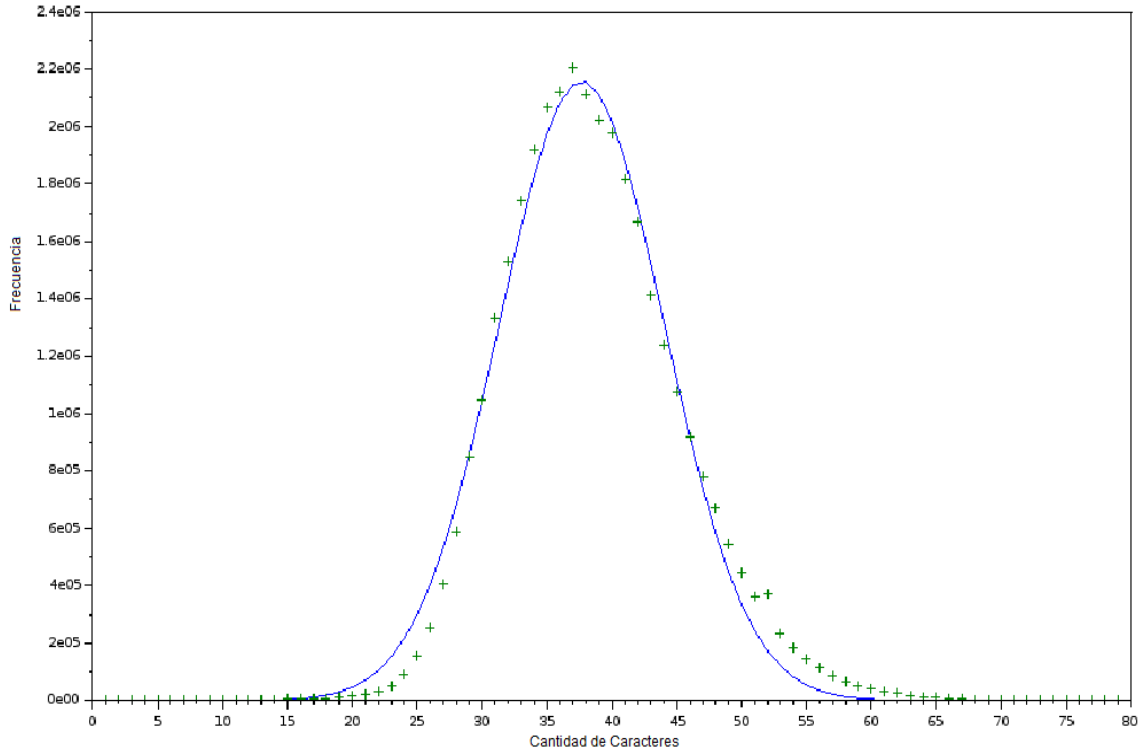
Anexo 48: Clase 4 de consultas – Log 201106 – corregida - Fuente: Elaboración propia



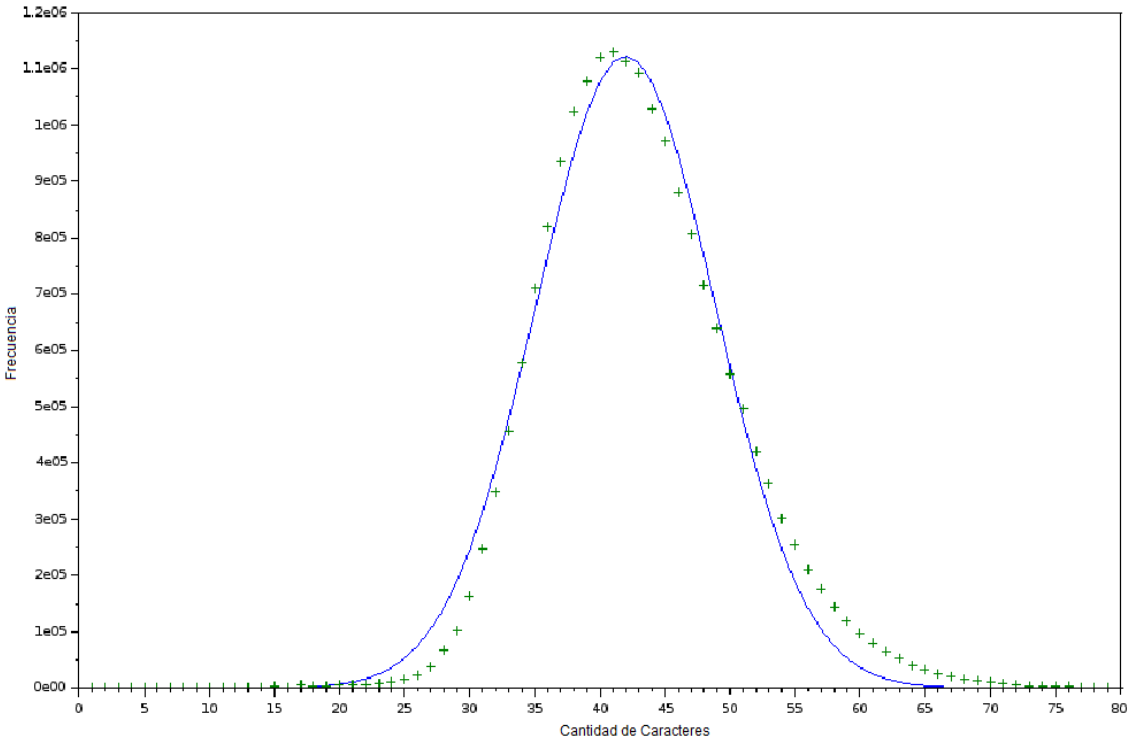
Anexo 49: Clase 5 de consultas – Log 201106 – corregida - Fuente: Elaboración propia



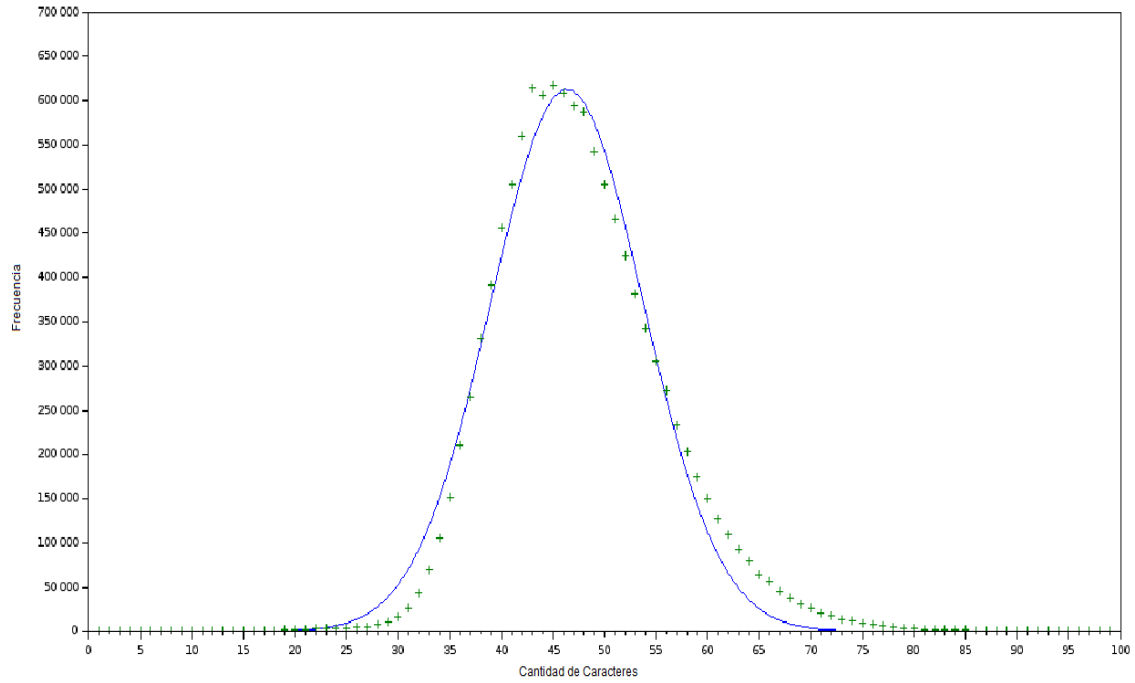
Anexo 50: Clase 6 de consultas – Log 201106 – corregida - Fuente: Elaboración propia



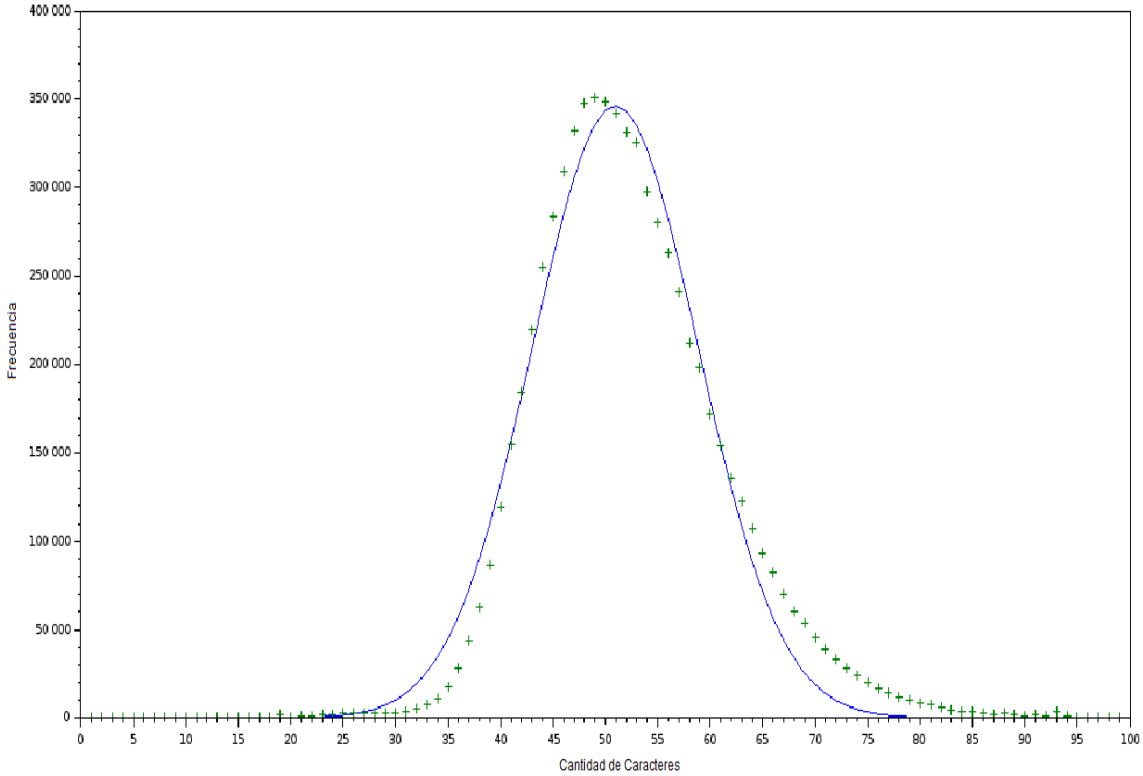
Anexo 51: Clase 7 de consultas – Log 201106 – corregida - Fuente: Elaboración propia



Anexo 52: Clase 8 de consultas – Log 201106 – corregida - Fuente: Elaboración propia



Anexo 53: Clase 9 de consultas – Log 201106 – corregida - Fuente: Elaboración propia



Anexo 54: Clase 10 de consultas – Log 201106 – corregida - Fuente: Elaboración propia