

Universidad Nacional de Asunción

Facultad Politécnica

Ingeniería Informática

Trabajo Final De Grado

PMTKU: A Parallel Modification algorithm for top-k high utility itemset

Abril 2017

1. **SÍNTESIS DEL PROYECTO**
   1. TÍTULO INICIAL DEL PROYECTO

|  |
| --- |
| PMTKU: A Parallel Modification algorithm for top-k high utility itemset |

SINTETIZAR EL PROYECTO EN TRES PALABRAS CLAVE

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | TKU |  | High Utility Itemset |  | Paralelo |

* 1. DURACIÓN DEL PROYECTO

|  |  |
| --- | --- |
| 12 | **MESES** |

* 1. DATOS DE LOS ESTUDIANTES Y DEL ASESOR DEL PROYECTO

|  |  |
| --- | --- |
| **NOMBRE DEL ASESOR PRINCIPAL:**  Ph.D. Diego Pinto | |
| **Email:** dppintoroa@gmail.com | **Teléfono:** 0984 401 410 |
| **CARGO:**  Docente | **FIRMA:** |

|  |  |
| --- | --- |
| **NOMBRE DEL ASESOR SECUNDARIO:**  Ing. Wilfrido Inchaustti | |
| **Email:** winchaus@gmail.com | **Teléfono:** 0971 599 500 |
| **CARGO:**  Docente | **FIRMA:** |

|  |  |
| --- | --- |
| **NOMBRE DEL ALUMNO:**  Federico Daniel Román Acosta | |
| **CI:** 3.550.387 | **PLAN DE ESTUDIOS:** 2008 |
| **Email:** fdra2323@gmail.com | **Teléfono:** 0981 207460 |
| **AÑO DE INGRESO: 2011** | **FIRMA:** |

|  |  |
| --- | --- |
| **NOMBRE DEL ALUMNO:**  Hernán David Medina Leiva | |
| **CI:** 3649882 | **PLAN DE ESTUDIOS:** 2008 |
| **Email:** hdmedina.py@gmail.com | **Teléfono:** 0961 459 653 |
| **AÑO DE INGRESO: 2011** | **FIRMA:** |

* 1. **RESUMEN DEL PROYECTO**

Actualmente uno de los problemas más explorados en el campo de la minería de datos es Frequent Pattern Mining [1], que consiste en encontrar patrones frecuentes en conjuntos de datos, pero que sean interesantes, inesperados y útiles. Existen algoritmos para descubrir varios tipos de patrones, como: tendencias, patrones periódicos, reglas secuenciales, patrones secuenciales, patrones de alta utilidad, etc.

Entre todos los algoritmos para descubrir distintos tipos de patrones uno de los más investigados en los últimos tiempos es el que busca patrones de alta utilidad, llamado High Utility Itemset Mining (HUIM) [2].

El problema de HUIM es dado un conjunto de datos, encontrar conjuntos de productos que generan una alta utilidad cuando son vendidos juntos. HUIM considera el caso donde los productos aparecen más de una vez en la misma transacción y también poseen un peso (por ej. la ganancia unitaria), al considerar esto se pueden encontrar conjuntos de productos con alta utilidad (por ej. alta ganancia). El usuario tiene que proveer un valor llamado mínima utilidad. El algoritmo HUIM devuelve todos los conjuntos de ítems con alta utilidad, o sea los que traen una ganancia de al menos el valor de la mínima utilidad.

El problema de HUIM es interesante por dos motivos:

1. Desde la perspectiva del negocio, porque descubre conjuntos de productos que generan una alta utilidad en las transacciones del negocio.
2. Desde la perspectiva de la investigación, porque dado un conjunto de productos, la utilidad de sus superconjuntos puede ser mayor, menor o igual. Esto finalmente repercute en la generación de posibles candidatos.

Identificar los elementos más frecuentes en un flujo de datos es un tema crítico pero desafiante en varias áreas. El desafío es aún mayor si este análisis se requiere a nivel individual, es decir, cuando se necesita una lista específica de elementos frecuentes ordenados por su utilidad de mayor a menor.

Encontrar los elementos más frecuentes es fundamental para cualquier análisis datos, ya que proporciona la información básica de las relaciones entre ítems.

En la investigación hecha se llegó al algoritmo llamado TKU que resuelve el problema de los últimos k (top-k) ítems con mayor utilidad en la lista de transacciones.

Se realizaron pruebas experimentales que resultaron en problemas de memoria comparado cualitativamente con el algoritmo EFIM (Efficient Frequent Itemset Mining) que calcula itemsets frecuentes. Las pruebas se realizaron con volúmenes de datos medianamente grandes.

Los datos generados para las pruebas fueron los siguientes:

* Utilidad máxima de un ítem (importancia en términos de ganancia en una transacción): 100.
* Cantidad máxima de columnas (cantidad de ítems diferentes en una transacción): 30.
* Las veces que puede repetirse un ítem en una transacción.
* Cantidad de transacciones: que fue variando durante las diferentes generaciones de datos.

Las pruebas y una comparativa entre los algoritmos TKU y EFIM son presentados en un gráfico (Figura 1). De las pruebas realizadas se puede concluir que, EFIM fue mucho más óptimo en tiempo de ejecución y uso de memoria durante las ejecuciones. Además se debe mencionar que para el algoritmo TKU se tiene un gran problema para conjunto de datos mayores a 10000 transacciones en términos de convergencia en un resultado. El problema que se tiene es la falta de memoria para terminar la ejecución.

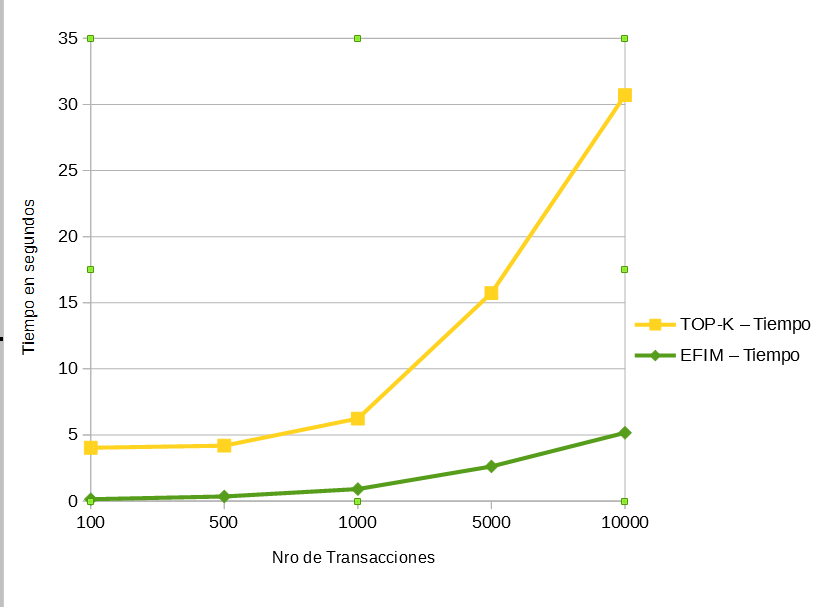


Figura 1: Comparación de tiempo de ejecución entre los algoritmos TKU y EFIM

Aunque en la pruebas se obtuvieron los mismos resultados con diferencias grandes en eficiencia ,no quiere decir que un algoritmo es mejor. Cada algoritmo tiene un enfoque distinto para resolver el mismo problema y según ese enfoque es que se debe elegir entre uno y otro.

* 1. **OBJETIVOS DEL PROYECTO.** 
     1. OBJETIVOS GENERALES
* Modificar y extender el algoritmo TKU para que pueda ser un algoritmo paralelo distribuido y comparable con otros algoritmos, permitiendo así la optimización del tiempo de descubrimiento de high utility itemsets en volúmenes de datos muy grandes.
  + 1. OBJETIVOS ESPECÍFICOS
* Implementar una modificación del algoritmo TKU para poder realizar comparaciones con otros algoritmos que asemejan su implementación en cuanto a entrada y salida del mismo.
* Extender el algoritmo TKU [3] mediante modelos como map reduce de manera que pueda ser paralelizable.
* Probar el algoritmo desarrollado y comparar los resultados con el algoritmo TKU.

1. **FORMULACIÓN GENERAL DEL PROYECTO**
   1. **PROBLEMA E HIPÓTESIS**

2.1.1 ¿CUÁL ES EL PROBLEMA U OPORTUNIDAD?

Existen muchos algoritmos para descubrir High Utility Itemsets, con distintos enfoques, estos algoritmos han ido mejorando con el tiempo enfocándose en la generación de candidatos, pero su rendimiento se ve afectado cuando se trabajan con volúmenes de datos muy grandes debido a la gran cantidad de memoria y tiempo que se necesita.

Uno de los algoritmos que presenta un buen desempeño es llamado EFIM [6] la ventaja que presenta el algoritmo de TKU sobre éste es que no requiere la entrada de un valor denominado utilidad mínima, en vez de está requiere la entrada de cuantos resultados es deseado obtener, siendo éste el valor de K.

La ventaja de requerir un valor K frente a una la utilidad mínima es que si el valor de la utilidad mínima es muy chica la cantidad de resultados será escasa y si es muy grande los resultados no serán de gran utilidad, a diferencia de si se tiene como entrada una cantidad K el resultado será más especificó proveyendo así más utilidad de resultados.

2.1.2 ¿CUÁL ES LA SOLUCIÓN PROPUESTA POR EL PROYECTO DE INVESTIGACIÓN?

Realizar la modificación del algoritmo TKU para que su comportamiento permite la comparación con otros algoritmos, en cuanto a la entrada y salida del mismo, y extender el algoritmo para buscar un mejor rendimiento mediante la paralelización y ejecución distribuida del mismo permitiendo agilizar la búsqueda de high utility itemsets mediante el framework Apache Spark?.

2.1.3 ¿CUÁL ES LA HIPÓTESIS DEL PROYECTO?

Se espera que el algoritmo PMTKU que se pretende desarrollar optimice el tiempo de búsqueda de high utility itemsets comparado con el TKU base y otros algoritmos que sigan el mismo patrón de entrada a la hora de tratar con volúmenes de datos muy grandes

* 1. **ANTECEDENTES.**

Frequent pattern mining ha sido un tema importante desde que el concepto de ‘conjunto de elementos frecuentes’ fue introducido por primera vez por Agrawal et al. [6]. Dado las transacciones de un conjunto de datos, frequent pattern mining encuentra los conjuntos de elementos cuyo soporte (es decir, el porcentaje de transacciones que contienen el conjunto de elementos) no es menor que un umbral de mínimo soporte dado. Sin embargo, ni el número de ocurrencias de un elemento en una transacción, ni la importancia de un elemento, se considera en frequent pattern mining. Los conjuntos de elementos con más ocurrencias o importancia pueden ser más interesantes para los usuarios, ya que pueden traer más beneficios.

A la luz de esto, se ha estudiado high utility itemset mining [4 - 5]. En high utility itemset mining, el término utilidad se refiere a la importancia de un conjunto de elementos; Por ejemplo, la ganancia total que el conjunto de elementos trae. Un conjunto de elementos es un conjunto de elementos de alta utilidad (HUI) si la utilidad del conjunto de elementos no es inferior a un umbral mínimo determinado.

High utility itemset mining se centra más en los valores de utilidad del conjunto de datos, que suelen estar relacionados con los beneficios del negocio. Estas utilidades son de interés para los propietarios del negocio, que podrían obtener más beneficios de ellos. Por ejemplo, los supermercados usan frequent itemset mining para encontrar productos que los clientes generalmente compran, para hacer recomendaciones a los clientes. Sin embargo, con high utility itemset mining, los supermercados podrán recomendar no sólo los productos que la gente suele comprar juntos, sino también los productos que generan más ganancias para la tienda esto sería de acuerdo a la ganancia o categoría del producto es decir si el cliente lleva una bebida de alta calidad está debería ser acompañada con algún producto comestible del mismo nivel de calidad, mientras mayor sea la calidad del producto mayor ganancia generará para el supermercado también podría ser visto desde un punto de vista educativo si bien existen materias complicadas dentro de una institución académica no todas las materias requieren el mismo esfuerzo, si se aplica la ejecución de un algoritmo que realice el cálculo de frequent itemsets se podría encontrar patrones que indiquen qué combinación de materias que requieran el mismo esfuerzo generan mayor cantidad de alumnos que deben volver a tomar la materia por segunda o inclusive tercera vez para poder avanzar en su plan curricular, teniendo estos datos la institución académica es capaz de revisar y analizar el contenido de las materias para realizar un ajuste en éste o poder estructurar una mejor planificación de las materias.

La mayoría de los algoritmos de frequent pattern mining eliminan los conjuntos de elementos en una etapa temprana, basados ​​en la popular propiedad Apriori [11]: cada sub-patrón de un patrón frecuente debe ser frecuente (también llamado propiedad de cierre descendente). Sin embargo, esta propiedad no se sostiene en high utility itemset mining, lo que hace que encontrar los conjuntos de elementos de alta utilidad sean más difíciles. Los enfoques del estado del arte alcanzan un buen rendimiento cuando el conjunto de datos es relativamente pequeño. Sin embargo, el volumen de datos puede crecer más rápido de lo esperado, que una sola máquina puede no llegar a ser suficiente para manejar grandes volúmenes de datos.

Una particularidad que posee el algoritmo de TKU o top-k high utility itemsets es que sigue un patrón de búsqueda diferente al resto de los algoritmos que realizan la búsqueda de high utility itemsets, debido a que éste realiza la búsqueda de las k mejores opciones que existen en los datos analizados sin embargo los otros algoritmos devuelven sus resultados a partir de un valor de utilidad mínima.

Una opción para resolver el problema de grandes volúmenes de datos es utilizar técnicas de computación paralela distribuida. El framework MapReduce [12] (por ejemplo, Hadoop) ha sido una solución popular recientemente, que permite el procesamiento distribuido de grandes volúmenes de datos en clusters de manera escalable y tolerante a fallos. Las aplicaciones que utilizan el framework de MapReduce tienen que conformar los protocolos de mapeo y reducción en un paradigma basado en disco, que restringe la flexibilidad así como el funcionamiento del algoritmo. Spark es también un framework de computación distribuida, basado en la memoria, que proporciona un rendimiento hasta 100 veces más rápido que Hadoop para ciertas aplicaciones [13]. Spark utiliza Resilient Distributed Dataset (RDD), que es una abstracción de memoria distribuida, para el cálculo de datos en memoria, lo que permite una reutilización eficiente de los datos.

En este trabajo, abordamos el problema de high utility itemset mining al proponer PMTKU (Parallel Modification algorithm for top-k high utility itemset). PMTKU es un algoritmo paralelo distribuido, que paraleliza a TKU y modifica la entrada del algoritmo para descubrir high utility itemset.

* 1. **METODOLOGÍAS**
* Investigación del estado del arte de:
  + High utility itemset mining
  + Parallel high utility itemset mining
* Selección del algoritmo a hacer paralelizable.
* Generación de los conjuntos de datos para las pruebas.
* Pruebas con los conjuntos de datos generados.
* Modelado del algoritmo PMTKU
* Implementación y depuración del algoritmo PMTKU
* Comparación con el algoritmo EFIM
* Redacción del documento científico
  1. **RESULTADOS ESPERADOS DEL PROYECTO**

Se espera que el algoritmo PMTKU propuesto para descubrir high utility itemsets en grandes conjuntos de datos mejore en términos de tiempo al algoritmo TKU y permita la comparación con otros algoritmos que sigan el mismo patrón de entrada.

1. **PLANEAMIENTO DEL PROYECTO**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Dic | Ene | Feb | Mar | Abr | May | Jun | Jul | Ago | Set | Oct | Nov |
| Investigación del Estado del Arte |  |  |  |  |  |  |  |  |  |  |  |  |
| Diseño del modelo propuesta. |  |  |  |  |  |  |  |  |  |  |  |  |
| Implementación. |  |  |  |  |  |  |  |  |  |  |  |  |
| Pruebas y Ajustes |  |  |  |  |  |  |  |  |  |  |  |  |
| Escritura del Libro |  |  |  |  |  |  |  |  |  |  |  |  |
| Evaluación de Resultados y elaboración de las conclusiones |  |  |  |  |  |  |  |  |  |  |  |  |
| Defensa del trabajo |  |  |  |  |  |  |  |  |  |  |  |  |

1. **TEMARIO TENTATIVO DEL REPORTE TÉCNICO FINAL**
2. Introducción
3. Descripción del problema
4. High utility itemset mining
5. Algoritmos paralelos
6. Parallel high utility itemset mining
7. Algoritmo propuesto basado en el TKU
8. Resultados Experimentales
9. Conclusiones y trabajos futuros
10. Referencias

**REFERENCIAS BIBLIOGRÁFICAS**

[1] An introduction to frequent pattern mining. <http://data-mining.philippe-fournier-viger.com/introduction-frequent-pattern-mining/>.

[2] An introduction to High-Utility Itemset Mining. <http://data-mining.philippe-fournier-viger.com/introduction-high-utility-itemset-mining/>.

[3] Cheng Wei Wu , Bai-En Shie , Philip S. Yu , Vincent S. Tseng Mining Top-K High Utility Itemsets .

[4] Quang-Huy Duonga,, Bo Liaoa, Philippe Fournier-Viger b, Thu-Lan Dama, An efficient algorithm for mining the top-k high utility itemsets, using novel threshold raising and pruning strategies

[5] Jerry Chun-Wei Lin a, Wensheng Gan a , Philippe Fournier-Viger b , Tzung-Pei Hong c,d, Vincent S. Tseng Fast algorithms for mining high-utility itemsets with various discount strategies

[6] Souleymane Zida, Philippe Fournier-Viger, Jerry Chun-Wei Lin,Cheng-Wei Wu, Vincent S. Tseng, EFIM: A Highly Efficient Algorithm for High-Utility Itemset Mining