Evaluación bibliométrica acerca de "Subprocesos con hilos de JAVA"

Bibliometric evaluation about "JAVA Threaded Subprocesses"

Wilian Delgado-Muentes

Universidad Laica Eloy Alfaro de Manabí. Manta, Ecuador
wilian.delgado@uleam.edu.ec

Armando Franco-Pico

Universidad Laica Eloy Alfaro de Manabí. Manta, Ecuador
armando.franco@uleam.edu.ec

Marco Ayoví-Ramírez

Universidad Laica Eloy Alfaro de Manabí. Manta, Ecuador
marco.ayovi@uleam.edu.ec

RESUMEN

JAVA es un lenguaje de programación de propósito general, basado en clases y orientado a objetos, diseñado para tener menores dependencias de implementación. Un hilo, en el contexto de JAVA, es el camino que se sigue al ejecutar un programa. Una aplicación de subproceso único tiene un solo subproceso y solo puede manejar una tarea a la vez. Para manejar varias tareas en paralelo, se utilizan subprocesos múltiples: se crean varios subprocesos, cada uno de los cuales realiza una tarea diferente. El propósito de la siguiente investigación fue desarrollar una evaluación bibliométrica de las publicaciones en torno a los subprocesos o hilos de JAVA, para lo cual se ejecutaron búsquedas de publicaciones relacionadas con JAVA mediante el uso de la base de datos ampliada Science Citation Index de Web of Science Core Collection (WoSCC). Un total de 965 documentos cumplieron con los criterios de búsqueda, que se aplicaron en este estudio. Entre los principales resultados derivados de este procedimiento se encontró que la producción anual de publicaciones se mantuvo en un nivel bastante bajo, pero aumentó drásticamente desde 2016 y alcanzó

un pico en 2020 con 550. Por su parte, el número de publicaciones anuales de China ha sido menor que el de los Estados Unidos en los primeros 6 años, pero la brecha se ha reducido gradualmente. Y en 2018, el número de publicaciones anuales de China superó al de Estados Unidos. Finalmente, La institución más productiva en investigación relevante fue la Universidad de Texas.

Palabras clave: JAVA, subprocesos, hilos, programación, evaluación bibliométrica, WoSCC.

ABSTRACT

JAVA is a general-purpose, class-based, object-oriented programming language designed to have fewer implementation dependencies. A thread, in the context of JAVA, is the path followed when executing a program. A single threaded application is single threaded and can only handle one task at a time. To handle multiple tasks in parallel, multithreading is used: multiple threads are created, each of which performs a different task. The purpose of the following investigation was to develop a bibliometric evaluation of publications around JAVA threads, for which JAVA-related publications were searched using the expanded Science Citation Index database of the Web of Science Core Collection (WoSCC). A total of 965 documents met the search criteria, which were applied in this study. Among the main results derived from this procedure, it was found that the annual production of publications remained at a fairly low level, but increased drastically since 2016 and reached a peak in 2020 with 550. For its part, the number of annual publications in China has been less than that of the United States in the first 6 years, but the gap has gradually narrowed. And in 2018, China's number of annual publications surpassed that of the United States. Finally, the most productive institution in relevant research was the University of Texas.

Keywords: JAVA, threads, bibliometric evaluation, programming, WoSCC.

1. INTRODUCCIÓN

JAVA fue desarrollado por James Gosling en Sun Microsystems Inc en el año 1991, luego adquirido por Oracle Corporation. Es un lenguaje de programación simple. Facilita la escritura, compilación y depuración de la programación. Ayuda a crear código reutilizable y programas modulares (Beilharz et al., 2016).

JAVA es un lenguaje de programación orientado a objetos y basado en clases y está diseñado para tener la menor cantidad posible de dependencias de implementación. Un lenguaje de programación de propósito general hecho para que los desarrolladores escriban una vez y se ejecuten en cualquier lugar que esté compilado. El código puede ejecutarse en todas las plataformas que admiten. Las aplicaciones se compilan en un código de bytes que se puede ejecutar en cualquier máquina virtual de JAVA. La sintaxis es similar a C/C++.

La historia de JAVA es muy interesante, es un lenguaje de programación creado en 1991. James Gosling, Mike Sheridan y Patrick Naughton, un equipo de ingenieros de Sun conocido como el equipo Green, inició el lenguaje JAVA en 1991. Sun Microsystems lanzó su primera implementación pública en 1996 como JAVA 1.0. Proporciona tiempos de ejecución sin costo en plataformas populares. El compilador JAVA 1.0 fue reescrito en Java por Arthur Van Hoff para cumplir estrictamente con sus especificaciones. Con la llegada de JAVA 2, las nuevas versiones tenían múltiples configuraciones construidas para diferentes tipos de plataformas (Bettini, 2015).

En 1997, Sun Microsystems se acercó al organismo de estándares ISO y luego formalizó JAVA, pero pronto se retiró del proceso. En un momento, Sun hizo que la mayoría de estas implementaciones estuvieran disponibles sin cargo, a pesar de su estado de software propietario. Sun generó ingresos a través de la venta de licencias para productos especializados como JAVA Enterprise System.

El 13 de noviembre de 2006, Sun lanzó gran parte de su máquina virtual JAVA como software gratuito de código abierto. El 8 de mayo de 2007, Sun finalizó el proceso y puso a disposición todo el código central de su JVM bajo términos de distribución de fuente abierta (Bettini, L and De Nicola, R, 2015).

Los principios para crear JAVA fueron simple, robusto, seguro, de alto rendimiento, portátil, de subprocesos múltiples, interpretado, dinámico, etc. James Gosling en 1995 desarrolló Java, conocido como el padre de Java. Actualmente, se usa en dispositivos móviles, programación de Internet, juegos, comercio electrónico, etc.

Después del nombre OAK, el equipo decidió darle un nuevo nombre y las palabras sugeridas fueron Silk, Jolt, revolucionario, ADN, dinámico, etc. Todos estos nombres eran fáciles de deletrear y divertidos de decir, pero todos querían el nombre para reflejar la esencia de la tecnología. Según James Gosling, JAVA es uno de los nombres principales junto con Silk, y dado que era un nombre único, la mayoría de ellos lo prefería (Ciancarini, P; Rossi, D., 2019).

Entre las principales características de JAVA, a juicio de Ciatto, G; Mariani, S; Zambonelli, F (2020) destaca que es:

1. Independiente de la plataforma, es decir, el compilador convierte el código fuente en código de bytes y luego la JVM ejecuta el código de bytes generado por el compilador. Este código de bytes puede ejecutarse en cualquier plataforma, ya sea Windows, Linux, macOS, lo que significa que, si compilamos un programa en Windows, podemos ejecutarlo en Linux y viceversa. Cada sistema operativo tiene una JVM diferente, pero la salida producida por todos los sistemas operativos es la misma después de la ejecución del código de bytes. Es por eso por lo que llamamos a java un lenguaje independiente de la plataforma.

2. Lenguaje de programación orientado a objetos: Organizar el programa en términos de colección de objetos es una forma de programación orientada a objetos, cada uno de los cuales representa una instancia de la clase.

Los cuatro conceptos principales de la programación orientada a objetos son:

- Abstracción
- Encapsulación
- Herencia
- Polimorfismo

3. Simple: JAVA es uno de los lenguajes simples, ya que no tiene características complejas como punteros, sobrecarga de operadores, herencias múltiples, asignación de memoria explícita.

4. Robusto: El lenguaje JAVA es robusto, lo que significa confiable. Está desarrollado de tal manera que se esfuerza mucho en comprobar los errores lo antes posible, por eso el compilador JAVA es capaz de detectar incluso aquellos errores que no son fáciles de detectar por otro lenguaje de programación. Las características principales de Java que lo hacen robusto son la recolección de basura, el manejo de excepciones y la asignación de memoria.

5. Seguro: En JAVA no tenemos punteros, por lo que no podemos acceder a matrices fuera de límite, es decir, muestra ArrayIndexOutOfBound Exception si intentamos hacerlo. Es por eso por lo que varias fallas de seguridad como la corrupción de la pila o el desbordamiento del búfer son imposibles de explotar.

6. Distribuidas: Podemos crear aplicaciones distribuidas utilizando el lenguaje de programación JAVA. La invocación de métodos remotos y Enterprise Java Beans se utilizan para crear aplicaciones distribuidas. Los programas se pueden distribuir fácilmente en uno o más sistemas que están conectados entre sí a través de una conexión a Internet.

7. Subprocesos múltiples: JAVA admite subprocesos múltiples. Es una característica que permite la ejecución simultánea de dos o más partes de un programa para la máxima utilización de la CPU.

8. Portable: Como sabemos, el código JAVA escrito en una máquina se puede ejecutar en otra máquina. La característica independiente de la plataforma en la que su código de bytes independiente de la plataforma se puede llevar a cualquier plataforma para su ejecución hace que Java sea portátil.

9. Alto rendimiento: La arquitectura de JAVA se define de tal manera que reduce la sobrecarga durante el tiempo de ejecución y, en algún momento, usa el compilador Just In Time (JIT), donde el compilador compila los conceptos básicos de código bajo demanda donde solo compila los métodos que son llamado hacer que las aplicaciones se ejecuten más rápido.

10. Flexibilidad dinámica: JAVA, al estar completamente orientado a objetos, nos brinda la flexibilidad de agregar clases, nuevos métodos a las clases existentes e incluso crear nuevas clases a través de subclases, incluso admite funciones escritas en otros lenguajes como C, C++, que se conocen como métodos nativos.

11. Ejecución de Sandbox: Los programas JAVA se ejecutan en un espacio separado que permite al usuario ejecutar sus aplicaciones sin afectar el sistema subyacente con la ayuda de un verificador de bytecode. El verificador de código de bytes también proporciona seguridad adicional, ya que su función es verificar el código en busca de cualquier violación de acceso.

12. Escribir una vez y ejecutar en cualquier lugar: Como se discutió anteriormente, la aplicación JAVA genera un archivo '.class' que corresponde a nuestras aplicaciones (programa) pero contiene código en formato binario. Proporciona facilidad de arquitectura neutral, ya que el código de bytes no depende de ninguna arquitectura de máquina. Es la razón principal por que se usa en la industria de TI emprendedora a nivel mundial.

13. Poder de compilación e interpretación: La mayoría de los lenguajes están diseñados con un propósito, ya sea que sean lenguajes compilados o que sean lenguajes interpretados. Pero JAVA integra un enorme poder que surge a medida que el compilador de JAVA compila el código fuente en un código de bytes y JVM ejecuta este código de bytes en el código ejecutable dependiente del sistema operativo de la máquina.

Los subprocesos o hilos, por su parte, permiten que un programa funcione de manera más eficiente al hacer varias cosas al mismo tiempo. Los subprocesos se pueden utilizar para realizar tareas complicadas en segundo plano sin interrumpir el programa principal (Custodio, JF and Cunha, JC, 2014).

Conceptualmente, la noción de hilo o subproceso no es difícil de comprender: es una ruta de ejecución independiente a través del código del programa. Cuando se ejecutan varios subprocesos, la ruta de un subproceso a través del mismo código generalmente difiere de los demás. Por ejemplo, suponga que un subproceso ejecuta el código de bytes equivalente a la parte if de una instrucción if-else, mientras que otro subproceso ejecuta el código de bytes equivalente a la parte else. ¿Cómo realiza la JVM un

seguimiento de la ejecución de cada hilo? La JVM le da a cada hilo su propia pila de llamadas a métodos. Además de rastrear la instrucción de código de bytes actual, la pila de llamadas al método rastrea las variables locales, los parámetros que la JVM pasa a un método y el valor de retorno del método.

Cuando varios subprocesos ejecutan secuencias de instrucciones de código de bytes en el mismo programa, esa acción se conoce como hilos múltiples. El hilo múltiple beneficia a un programa de varias maneras (De Nicola, R and Loreti, M, 2018):

- Los programas basados en GUI (interfaz gráfica de usuario) multiproceso siguen respondiendo a los usuarios mientras realizan otras tareas, como repaginar o imprimir un documento.
- Los programas con subprocesos normalmente terminan más rápido que sus homólogos sin subprocesos. Esto es especialmente cierto en el caso de los subprocesos que se ejecutan en una máquina multiprocesador, donde cada subproceso tiene su propio procesador.

A diferencia de muchos otros lenguajes informáticos, JAVA proporciona soporte integrado para subprocesos múltiples. El subproceso múltiple en JAVA contiene dos o más partes que pueden ejecutarse al mismo tiempo. Se plantea el siguiente ejemplo:

Imagine una aplicación de corredor de bolsa con muchas capacidades complejas. Estas son algunas de sus funciones (Drejhammar, F; Schulte, C; Brand, P; Haridi, S, 2013):

Para descargar los últimos precios de opciones sobre acciones

Para comprobar los precios de las advertencias

Analizar datos históricos de la empresa XYZ

Estas son funciones que requieren mucho tiempo. En un entorno de ejecución de un solo subproceso, estas acciones se ejecutan una tras otra. La siguiente acción puede suceder solo cuando finalice la anterior.

Ahora bien, si un análisis histórico lleva media hora y el usuario selecciona realizar una descarga y una verificación después, la advertencia puede llegar demasiado tarde

para comprar o vender acciones como resultado. Simplemente imaginamos el tipo de aplicación que pide a gritos el multihilo. Idealmente, la descarga debería ocurrir en segundo plano (es decir, en otro hilo). De esa forma, podrían ocurrir otros procesos al mismo tiempo para que, por ejemplo, se pudiera comunicar una advertencia al instante. Mientras tanto, el usuario interactúa con otras partes de la aplicación. El análisis también podría ocurrir en un hilo separado, por lo que el usuario puede trabajar con el resto de la aplicación mientras se calculan los resultados. Aquí es donde ayuda el hilo de Java (Eugster, P, 2017).

Un hilo es en realidad un proceso ligero. A diferencia de muchos otros lenguajes informáticos, JAVA proporciona soporte integrado para programación multiproceso. Un programa multiproceso contiene dos o más partes que pueden ejecutarse al mismo tiempo. Cada parte de dicho programa se llama subproceso y cada subproceso define una ruta de ejecución separada. Por lo tanto, el subproceso múltiple es una forma especializada de multitarea (Fahringer, T and Jugravu, A., 2015).

El sistema de ejecución de JAVA depende de hilos para muchas cosas. Los hilos o subprocesos reducen la ineficiencia al evitar el desperdicio de ciclos de CPU. Existen hilos en varios estados. Los siguientes son esos estados:

- Nuevo: Cuando creamos una instancia de la clase Thread, un hilo está en un nuevo estado.
- Ejecutable: El subproceso de JAVA está en estado de ejecución.
- Suspendido: Un hilo en ejecución se puede suspender, lo que suspende temporalmente su actividad. Luego, se puede reanudar un hilo suspendido, lo que le permite continuar donde lo dejó.
- Bloqueado: Un subproceso de JAVA se puede bloquear cuando se espera un recurso.
- Terminado: Un hilo se puede terminar, lo que detiene su ejecución inmediatamente en cualquier momento. Una vez que se termina un hilo, no se puede reanudar (Hijma, P; van Nieuwpoort, RV; Bal, HE, 2012).

El sistema de subprocesos múltiples de JAVA se basa en la clase Thread, sus métodos y su interfaz complementaria, Runnable. Para crear un nuevo hilo, su programa extenderá el hilo o implementará la interfaz Runnable (Linden, I; Jacquet, JM; (...); Brogi, A, 2016).

La investigación en torno a JAVA en general y los subprocesos en específico, pareciera no ser tan prolífica como la de otros campos de la ciencia. Este artículo ahonda en esa circunstancia.

2. MATERIALES Y MÈTODOS

Estrategias de búsqueda

Se realizaron búsquedas en publicaciones sobre la investigación de JAVA en enero de 2021 mediante el uso de la base de datos ampliada Science Citation Index de Web of Science Core Collection (WoSCC). Los términos de búsqueda que se utilizaron fueron los siguientes: JAVA AND (Thread OR hilos OR subprocesos). Solo se incluyeron publicaciones anteriores a 2021. Un total de 965 documentos cumplieron con los criterios de búsqueda, que se aplicaron en este estudio.

Recogida y análisis de datos

Se exportó el registro completo de datos de WoSCC, incluida la cantidad de publicaciones anuales; productos de países/regiones, revistas, autores y citas totales; factor de impacto (IF) en 2019, Journal Citation Reports (JCR) 2019 e índice de Hirsch (H-index). JCR es una base de datos que sirve como base para Journal Impact Factor (JIF), que es una herramienta de evaluación de revistas que puede reflejar el impacto y la calidad de las revistas. Y el índice H significa que H artículos publicados por un autor o un país/región han sido citados al menos H veces, y cada uno de los artículos restantes ha sido citado menos o igual a H veces. Se puede utilizar para evaluar la cantidad y el nivel de producción académica de investigadores o países/regiones.

Luego, se importaron los datos a Microsoft Excel 2019, VOSviewer y CiteSpace 5.7. R2 para un análisis más detallado. Se utilizó Microsoft Excel 2019 para crear un gráfico de tendencias de cambios en el volumen de documentos publicados cada año, citas por artículo, número total de citas e índice H de los diez países/regiones principales y proporcionar una tendencia de desarrollo intuitiva en este campo.

Se determinó la importancia de un artículo con base en el supuesto de que los artículos muy citados pueden tener un mayor impacto en la literatura científica. Co-

citación significa que dos artículos (o autores) relacionados son citados por el tercer artículo (o autor) al mismo tiempo. Los artículos con un tema común tienden a formar grupos alrededor del mismo par de artículos cocitados, destacando así su importancia y relevancia (Luc Moreau, 2017).

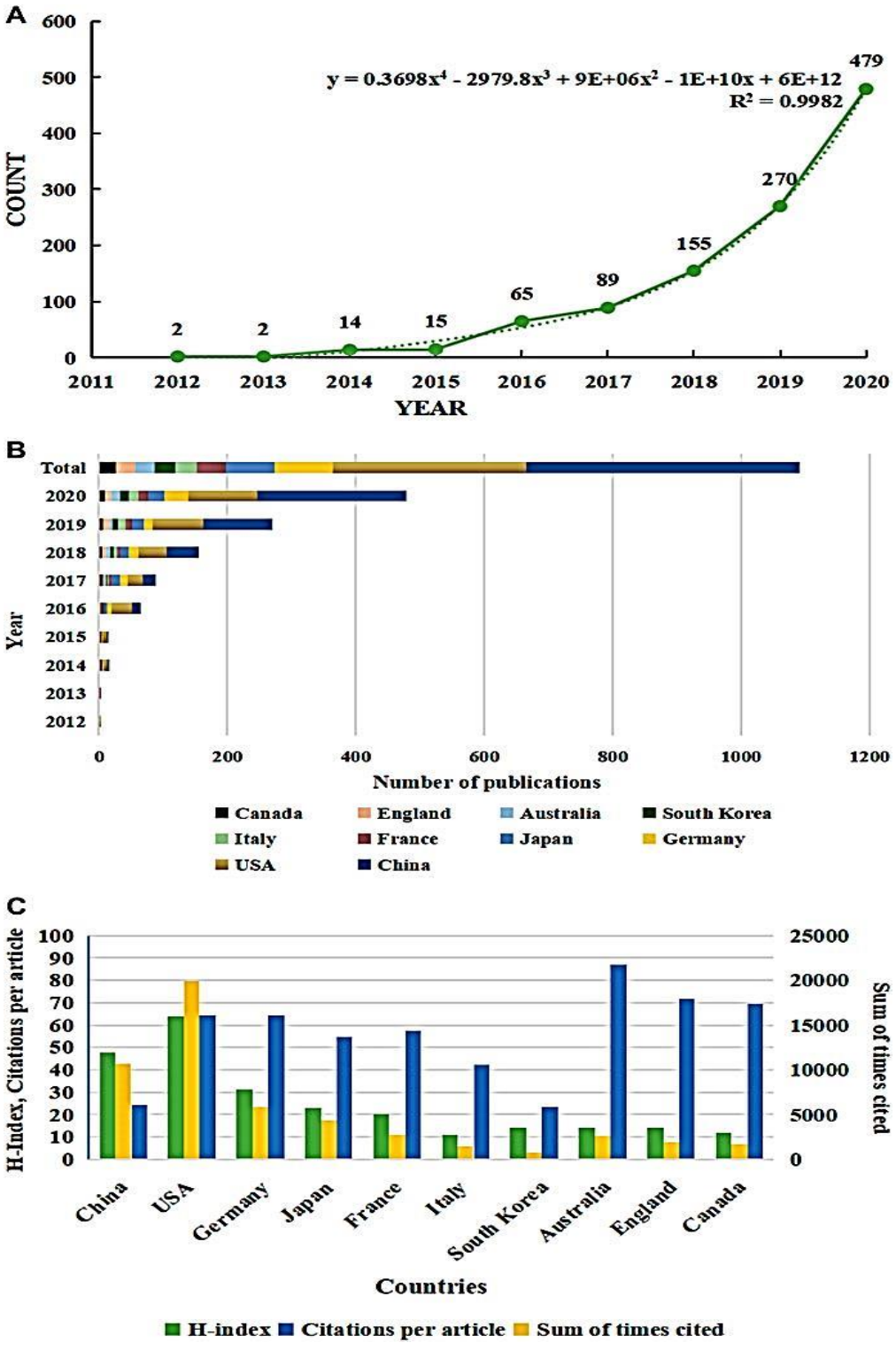
Se aplicó VOSviewer para analizar la cocitación de revistas y autores. VOSviewer puede proporcionar datos de revistas cocitadas, autores, referencias y sus citas, de modo que reflejen la relación cooperativa y las relaciones académicas junto con el mapa de visualización creado por CiteSpace.

Se empleó CiteSpace para visualizar colaboraciones entre países/regiones, organizaciones y autores y realizar análisis de cocitación de autores, revistas y referencias. Además, se exploró los cambios en las direcciones y tendencias de la investigación mediante la creación de una vista de línea de tiempo de la referencia cocitada. Para explorar mejor los puntos críticos de investigación, se usó CiteSpace para capturar palabras clave con fuertes ráfagas de citas y se construyó un mapa de visualización.

3. RESULTADOS

En este artículo, se utilizó visualización de información para analizar artículos originales sobre threads o hilos, de la base de datos Web of Science (WOS) de 2012 a 2020. Un total de 965 publicaciones cumplieron con los criterios de búsqueda. En los primeros 4 años (2012-2015), la producción anual de publicaciones se mantuvo en un nivel bastante bajo, pero aumentó drásticamente desde 2016 y alcanzó un pico en 2020 con 550 (Figura 1A). Mediante ajuste de modelos hemos obtenido una curva de crecimiento del volumen de publicaciones cada año. Sobre la base de esta curva, se concluyó que los subprocesos están a punto de convertirse en un tema cada vez más importante.

Figura 1. Producción de publicaciones anuales y por países



Vale la pena señalar que la producción anual de publicaciones incluso ha aumentado exponencialmente en los últimos 3 años. Revela que la investigación de los hilos se

está desarrollando rápidamente y que la atención ha ido en aumento en el campo de la computación, que se espera que se convierta en un tema candente en el futuro. El artículo publicado por Martin, D; Wutke, D and Leymann, F (2015), que confirmaba que los subprocesos representan quizás lo más significativo de JAVA. Este artículo se considera la pieza más importante y fundamental, que es pionera en la investigación en este campo (Dixon et al., 2012). Además, Pittarello, F (2014) publicó una revisión en las revisiones de CompScience, con el IF alto (34.1060), el 29 de enero de 2021. Esta revisión presenta las estructuras de los subprocesos aplicadas a diversos campos.

Análisis de país/región e institución

La publicación relacionada con los subprocesos fue publicada por primera vez por académicos estadounidenses en 2012, que fue el único artículo de investigación en ese momento. Su investigación está por delante de los investigadores en todas las regiones del mundo. Todas las publicaciones en el campo se distribuyeron entre 1.169 instituciones de 59 países/regiones, de las cuales la República Popular China ocupó el primer lugar con 437 (45,28 %) documentos con diferencia, seguida por Estados Unidos (310, 32,12 %). Alemania (90, 9,33%), Japón (80, 8,29%) y Francia (47, 4,87%) (Tabla 1). A través del análisis de cambios en el volumen anual de documentos emitidos de los 10 países/regiones más productivas (Figura 1B), se puede concluir que la tendencia general en el número anual de documentos publicados por esos países mostró un aumento similar a partir de 2012.

En una comparación horizontal, el número de publicaciones anuales de China ha sido menor que el de los Estados Unidos en los primeros 6 años, pero la brecha se ha reducido gradualmente. Y en 2018, el número de publicaciones anuales de China superó al de Estados Unidos y se convirtió en el primero. Aunque la producción total de investigación de países como Alemania, Japón, Francia e Italia es inferior a la de China y Estados Unidos, también ha habido un crecimiento significativo en los últimos años. Obviamente, una de las razones más importantes es la estrecha y amistosa cooperación entre los diferentes países. El análisis del mapa de visualización de la red CiteSpace de países/regiones (Figura 2) mostró que existe una cooperación muy fuerte entre los Estados Unidos y China, lo que ha impactado significativamente la tendencia de investigación de los subprocesos.

Tabla 1. Principales países por artículos y citas

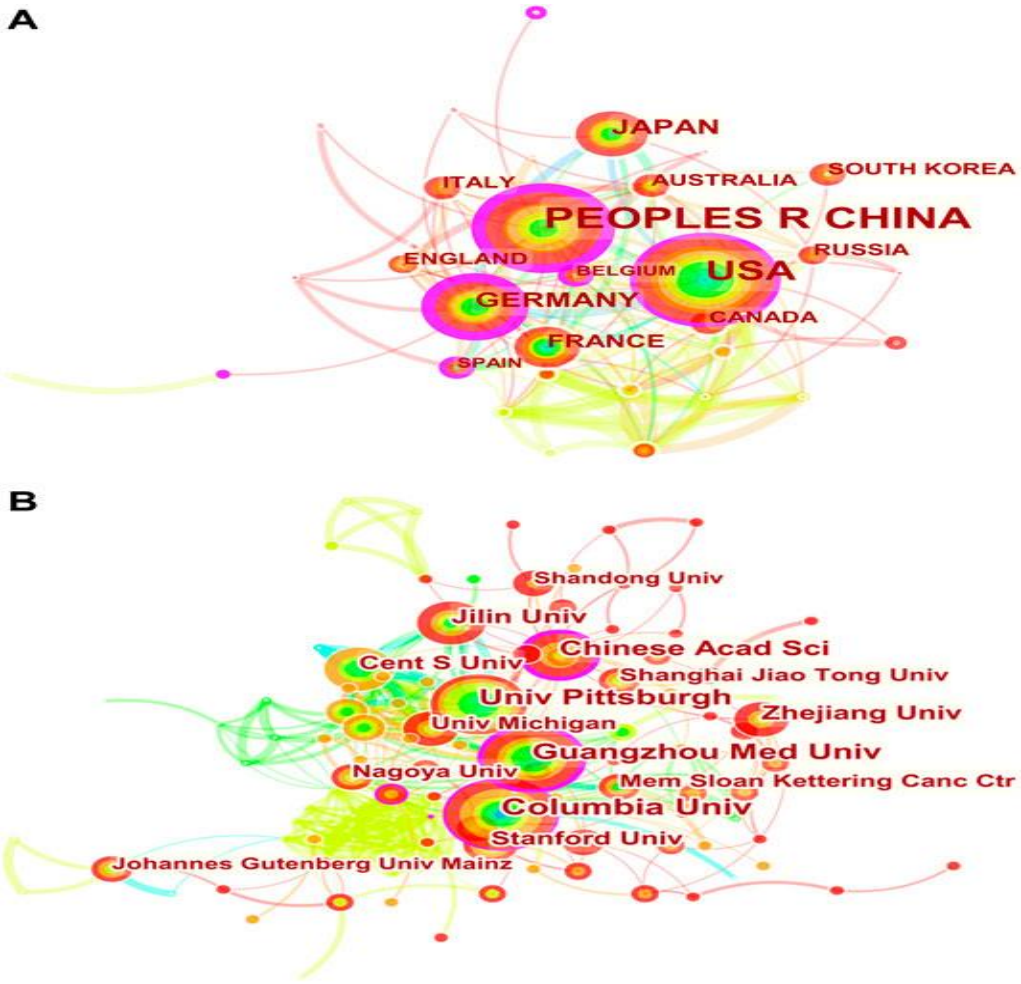
Rank	Countries/regions	Articles (N)	Percentage (N/965	H-index	Citations per article	Sum of times cited
1	China	437	45.28	48	24.51	10,712
2	United States	310	32.12	64	64.33	19,942
3	Germany	90	9.33	31	64.38	5,794
4	Japan	80	8.29	23	54.81	4,385
5	France	47	4.87	20	57.68	2,711
6	Italy	34	3.52	11	42.44	1,443
7	South Korea	32	3.32	14	23.19	742
8	Australia	30	3.11	14	87.00	2,610
9	Canada	26	2.69	14	71.96	1871
10	England	25	2.59	12	69.40	1735

Cuando se trata de la suma de las veces citadas y las citas por artículo, las cosas son totalmente diferentes (Figura 1C y Tabla 1). EE. UU. tuvo 19 942 citas y un índice H de 64, los cuales ocuparon el primer lugar entre todos los países/regiones incluidos, pero su proporción de citas/artículo (64,33) fue menor que la de Canadá (71,96), Inglaterra (69,4) y Alemania. (64.38). Como el país o las regiones más productivas en relación con los subprocesos en la investigación sobre JAVA, China tuvo una proporción relativamente baja de citas por artículo (24,51) en comparación con los demás países de la tabla.

Para explorar la cooperación internacional e interinstitucional, construimos un mapa de visualización de redes para publicaciones sobre la investigación de los subprocesos por CiteSpace. La Figura 2A muestra colaboraciones entre países/regiones que cumplen el criterio de búsqueda (58). Los países marcados con círculos morados, incluidos Alemania (0,37), Estados Unidos (0,30) y la República Popular China (0,28) tienen la centralidad de intermediación más fuerte, lo que significa que jugaron un papel clave en la colaboración internacional. La conexión entre nodos representa la relación de cooperación entre países, y el ancho de la conexión representa la fuerza de la cooperación. Alemania tuvo la centralidad de intermediación más alta, y los países/regiones que más colaboraron con Alemania

fueron Camerún, Japón y Australia. Además, la República Popular de China colaboró más con EE. UU., Suiza y Francia.

Figura 2. Colaboraciones de instituciones entre países y regiones



Las 10 instituciones más productivas en investigación relevante se muestran en la Tabla 2. Las instituciones líderes fueron University of Texas System (51,5,28%), Columbia University en la ciudad de Nueva York (44,4,56%), Central South University (42,4,35%), Guangzhou University (42,4,35%) y Asociación Helmholtz (39, 4,04%). Las colaboraciones entre instituciones se muestran claramente en la Figura 2B. La Universidad de Columbia en la ciudad de Nueva York, la Academia de Ciencias de China, la Universidad Central del Sur y la Universidad de Pittsburgh tienen una fuerte centralidad de intermediación. Además, se puede ver claramente que la cooperación entre las instituciones americanas es extremadamente activa y cercana, con cinco instituciones entre las 10 instituciones con mayor producción de publicaciones. Las

ventajas regionales se han aprovechado superlativamente en gran medida, fortaleciendo la influencia académica de los Estados Unidos en este campo.

Tabla 2. Diez importantes instituciones de investigación

Rank	Institutions	Articles (N)	Percentage (N/965)	Location
1	Univ of Texas System	51	5.28	United States
2	Columbia Univ	44	4.56	United States
3	Cent S Univ	42	4.35	China
4	Guangzhou Med Univ	42	4.35	China
5	Helmholtz association	39	4.04	Germany
6	Inserm	39	4.04	France
7	Pcshe	38	3.94	United States
8	Chinese Acad Sci	35	3.63	China
9	Univ of Pittsburgh	35	3.63	United States
10	Ut SouthWestern	28	2.90	United States

Análisis de palabras clave

Se usó VOSviewer para analizar las palabras clave y se formó tres grupos. Al mismo tiempo, para analizar la cercanía de la relación entre diferentes temas, se usó CiteSpace para crear un mapa de co-ocurrencia de palabras clave. El tamaño de los nodos refleja la frecuencia de las palabras clave, y el ancho de las líneas representa la frecuencia de la coexistencia de dos palabras clave. Las palabras clave, como JAVA, subprocesos y programación, tienen una fuerte centralidad de intermediación. Es obvio que el mecanismo de los subprocesos siempre ha sido el punto caliente de la dirección de investigación.

4. CONCLUSIONES

Los subprocesos con hilos de JAVA son importantes en procesos interactivos y en segundo plano con elementos asincrónicos y de estructura modular que aceleran la

ejecución en hilos a nivel de usuarios y del Kernel en diversas aplicaciones de áreas comerciales y científicas.

El conocimiento y comprensión de los subprocesos ha mejorado significativamente a través de un estudio serio de los artículos anteriores de alta calidad. Según la búsqueda, este artículo es el primer análisis bibliométrico basado en los resultados de CiteSpace y VOSviewer para estudiar las tendencias de investigación y los puntos calientes de los subprocesos en la programación con JAVA. Cada vez más estudios han demostrado que los subprocesos resultan fundamentales en la programación actual.

Además, los subprocesos también están relacionados con la optimización de los procesos en diversidad de áreas del conocimiento (Rossi, G; Panegai, E and Poleo, E, 2015). A partir del estallido de citas detectado, se pudo encontrar que la aplicación complementaria de los subprocesos en la programación es una tendencia emergente. Con la ayuda de la visualización de información, es posible identificar las prioridades de investigación y las tendencias generales en el campo, y poner la información recopilada a disposición de los investigadores en este campo. A través de una mayor exploración del mecanismo de los subprocesos, se espera que se proporcionen nuevas ideas de análisis y organización de la información.

REFERENCIAS

Beilharz, J; Feinbube, F; (...); Polze, A (2016). Claud: Coordination, Locality and Universal Distribution. Parallel computing: on the road to exascale 27, pp.605-614.

Bettini, L. (2015). Data Privacy in Tuple Space Based Mobile Agent Systems. Electronic notes in theoretical computer science 128 (5), pp.3-16.

Bettini, L and De Nicola, R (2015). Mobile distributed programming in X-KLAIM. Formal methods for mobile computing 3465, pp.29-68.

Ciancarini, P; Rossi, D. (2019). Java: Coordination and communication for Java agents. Lecture Notes in Computer Science.

Ciatto, G; Mariani, S; (...); Zambonelli, F (2020). Twenty years of coordination technologies: Coordination contribution to the state of art. journal of logical and algebraic methods in programming 113.

Custodio, JF and Cunha, JC (2014). JGroupSpace: Combining Shared Spaces and Groups. Proceedings of the 2009 international symposium on collaborative technologies and systems, pp.284-291.

De Nicola, R and Loreti, M (2018). Modelling global computations with KLAIM. Philosophical transactions of the royal society a-mathematical physical and engineering sciences 366 (1881), pp.3737-3745.

Drejhammar, F; Schulte, C; Brand, P; Haridi, S (2013). Flow Java: Declarative concurrency for Java. Cconcurrency and computation-practice & experience 27 (17), pp.4716-4740.

Eugster, P (2017). Type-based publish/subscribe: Concepts and experiences. ACM transactions on programming languages and systems 29 (1).

Fahringer, T and Jugravu, A. (2015). JavaSymphony: a new programming paradigm to control and synchronize locality, parallelism and load balancing for parallel and distributed computing. Concurrency and computation-practice & experience 17 (7-8), pp.1005-1025.

Hijma, P; van Nieuwpoort, RV; Bal, HE (2012). Generating synchronization statements in divide-and-conquer programs. Parallel computing 38 (1-2), pp.75-89.

Linden, I; Jacquet, JM; (...); Brogi, A (2016). On the expressiveness of timed coordination models. Science of computer programming 61 (2), pp.152-187.

Luc Moreau (2017). Formalizing the safety of Java, the Java virtual machine, and Java card. ACM Computing Surveys.

Martin, D; Wutke, D and Leymann, F (2015). Tuplespace middleware for Petri net-based workflow execution. International journal of web and grid services 6 (1), pp.35-57.

Pittarello, F (2014). The time-pillar world - A 3D paradigm for the New Enlarged TV information domain. Personalized digital television: targeting programs to individual viewers, pp.287-320.

Rossi, G; Panegai, E and Poleo, E (2015). JSetL: a Java library for supporting declarative programming in Java. Software-practice & experience 37 (2), pp.115-149.