Acercamiento a la Computación de Alta Productividad. Una experiencia de su aplicación en la Industria Cubana del Níquel

APPROACH TO HIGH THROUGHPUT COMPUTING. AN EXPERIENCE OF ITS APPLICATION IN CUBAN NICKEL INDUSTRY

Dannier Trinchet Almaguer
Universidad de las Ciencias Informáticas, Cuba, trinchet@uci.cu,
Carr. San Antonio de los Baños Km $2\ 1/2$ Torrens Ciudad de La Habana

RESUMEN

En el presente trabajo hace un acercamiento a la Computación Distribuida, en particular, a la Computación de Alta Productividad. Se presenta el panorama actual de esta rama de la Ciencia de la Computación que está ganando espacio en el mundo de las herramientas computacionales que persiguen dar solución a problemas de alto costo computacional; se hace un recorrido que muestra cómo combinando el tiempo ocioso de nodos computacionales no dedicados es posible resolver problemas de este tipo; se abordan además los sistemas distribuidos más representativos de hoy en día. Finalmente se presentan los resultados obtenidos con la aplicación de uno de estos sistemas en el modelado de yacimientos lateríticos a partir de un modelo de Markov, problema de suma importancia para la Industria Cubana del Níquel.

Palabras Clave: Computación de Alta Productividad, Modelos Markovianos, Industria Cubana del Níquel

ABSTRACT

This paper approaches Distributed Computing, in particular the High Throughput Computing. We present a overview of this Computer Science's area that currently is gaining ground in the world of computational tools that seek to solve problems of high computational cost, this panorama shows how combining the idle time of computing nodes not dedicates is possible to solve problems of this type, are discussed further distributed systems more representative of today. Finally, we present the results obtained by applying one of these systems in the modeling of lateritic deposits using a Markov model, a problem with a lot of importance for the Cuban nickel industry.

KeyWords: High Throughput Computing, Markovian models, Cuban nikel industry

1. Introducción

Muchas definiciones de Computación Distribuida pueden encontrarse en la literatura. Según [1] un Sistema Distribuido (SD) es una colección de computadoras independientes que se presenta ante el usuario del sistema como una simple computadora. Según [2] un SD es un sistema de procesamiento de información que está compuesto por varias computadoras independientes que cooperan entre ellas mediante una red de comunicaciones con el objetivo de alcanzar un objetivo específico.

Por su parte [3] plantea que un SD es un sistema que ante sus usuarios aparenta ser un sistema ordi-

nario, pero que corre sobre un conjunto de elementos autónomos de procesamiento (EP), cada uno de los cuales tiene su propia memoria física y la demora por la transmisión de mensajes entre ellos puede ser no despreciable. Existe una cooperación cercana entre cada par de EPs. El sistema debe soportar un número arbitrario de tareas y la extensión dinámica de EPs.

Lo cierto es que se coincide en que la computación distribuida brinda acceso transparente al poder de cómputo y almacenamiento, de muchas computadoras independientes, que un usuario necesita para realizar una determinada tarea, al tiempo que permite alcanzar altos índices de rendimiento y confiabilidad mediante el aprovechamiento de esos recursos computacionales.

2. Computación de Alta Productividad

La Computación de Alta Productividad (HTC, del inglés *High Throughput Computing*) estudia las posibilidades que ofrecen los entornos distribuidos, en particular los no dedicados, para la realización de cómputo paralelo. El término, y su diferencia respecto a la Computación de Altas Prestaciones (HPC, del inglés High Performance Computing), fue introducido por Miron Livny, padre del proyecto Condor (ver sección 2.1.2) en un seminario realizado en el *NASA Goddard Flight Center, US*.

En junio del 1997 la revista HPCWire publica una entrevista [4] donde Livny explica que la HTC está orientada a maximizar la cantidad de tareas a realizar por unidad de tiempo, o sea, dado un conjunto de trabajos a realizar minimizar el tiempo necesario para la resolución de todos ellos aprovechando la mayor cantidad posible de recursos.

Muchas instituciones tanto académicas como empresariales cuentan con redes de computadoras que aglutinan estaciones de trabajo (NOW/COW) cuyo poder de cómputo no es totalmente explotado.

Se han realizado estudios que determinan con precisión cual es el uso de las CPUs de estas estaciones de escritorio [5–8] y otros que muestran que el aprovechamiento de los ciclos ociosos de estas últimas es una alternativa importante para alcanzar elevadas prestaciones [9–11] y en consecuencia resolver problemas complejos.

En [6] se presenta un estudio realizado en una institución académica que arrojó que más de la mitad de las estaciones de trabajo de una red local se encuentran disponible más de la mitad del tiempo, como promedio de un 60 % a un 80 %; y una fracción importante de estas siempre se encuentran disponibles; muestra además que, a pesar de esa alta disponibilidad, en general los clusters no son estables pues hay una alta variación en la distribución del periodo de tiempo ocioso, se determinó que hasta la mitad de las estaciones son estables de 4 a 15 minutos y que una tercera parte lo son de 5 a 30 minutos, otro dato interesante fue el hecho de que como promedio las estaciones que estaban ociosas por 5 minutos tenían una alta probabilidad de seguir en ese estado en los siguientes 40 a 90 minutos.

En cuanto a las prestaciones obtenidas se señala que se lograron desempeños equivalentes a los alcanzados por un cluster de máquinas dedicadas de un tamaño que varió desde un tercio hasta tres cuartas partes de la cantidad de estaciones de trabajo abordadas por el estudio, esta equivalencia fue medida a partir de la métrica *Cluster Equivalence Metric (CEM)*[1] publicada por primera vez en este artículo. Se determinó además que las prestaciones no dependen solamente del poder de cómputo que cada máquina posee sino además de la flexibilidad de las tareas que pueden ejecutarse concurrentemente.

Para el monitoreo del desempeño de COWs y el pronóstico de las capacidades de estos últimos en [7] se propone *Network Weather Service*: un servicio que en las pruebas realizadas en dos universidades norteamericanas determinó los horarios en los cuales se podían obtener elevadas prestaciones.

La investigación presentada en [12] estuvo dirigida a determinar los horarios más adecuados para realizar cómputo en una universidad europea, los resultados obtenidos fueron que de lunes a viernes de 8 AM a 8 PM el por ciento de inactividad de las CPUs osciló de un 60 a un 80 % y durante las noches de un 95 a un 100 %, teniendo en cuenta las 24 horas del día la disponibilidad aproximada estuvo entre un 75 y un 80 % subiendo a 90-95 % durante los fines de semana.

En [10], realizada con 220 PCs y con objetivo fundamental estudiar el desempeño/utilidad real de la plataforma Grid Entropia [13] elemento este poco estudiado hasta entonces debido a las interacciones del comportamiento dinámico de los recursos y la compleja estructura de este tipo de aplicaciones, se cuantificó la utilidad de estos entornos para la realización de cómputo paralelo a partir de la estructura y distribución temporal de los ciclos ociosos y su impacto en aplicaciones Grid de escritorio. Se concluyó además que es posible en instituciones empresariales maximizar el rendimiento de las inversiones debido al bajo costo empleado para alcanzar elevada productividad, y sugieren su uso para resolver problemas cuya solución se base en tareas de grano

[1] Esta métrica expresa para un cluster COW, la fracción de un cluster dedicado que representa cada CPU no dedica enfrascada en la ejecución de una aplicación

fino debido a la volatilidad de los recursos.

Luego en [11] se presenta un estudio realizado a partir de 169 estaciones de trabajo en una universidad durante 77 días consecutivos. Este concluyó que como promedio el 97.9 % del tiempo las CPUs estaban ociosas y el 42.1 % de la memoria estaba desocupada. En cuanto al desempeño se determinó que con N recursos no dedicados se alcanzaba un poder de cómputo similar al de $N/2$ máquinas dedicadas, también medido a partir de la métrica CEM según las consideraciones hechas en [10].

Por otro lado se ha estudiado el uso que tienen los servidores. Por ejemplo, en [14] la IBM muestra que los servidores empresariales bajo Windows y Linux están ociosos aproximadamente el 95 % y 85 % del día respectivamente.

Estos estudios confirman las potencialidades que ofrece el uso de las estaciones de trabajos aglutinadas en una LAN para lograr no solo alta productividad sino elevados desempeños en la resolución de problemas complejos, hecho resaltado por Livny en su entrevista con la HPCWire [4]. No obstante, coinciden y de lo cual se puede concluir que los mayores desempeños se alcanzan en problemas donde se cumplan las siguientes condiciones:

- Es posible resolverlo mediante una descomposición de grano fino, o al menos su tamaño pueda ajustarse dinámicamente para garantizar una flexibilidad ante la volatilidad de los recursos de cómputo.
- Su resolución debe estar basada en el modelo SPMD.
- Es posible resolverlo sin información (ni suposiciones) a cerca del hardware con que se cuenta i.e la cantidad de máquinas disponibles, por ciento de uso de CPU de cada una, tiempo y longitud del ciclo ocioso, etc.

Por otro lado, es bastante tentador pensar en las ventajas en cuanto a costo-beneficio de los sistemas HTC sobre los HPC, nótese que en los primeros no es necesario hacer inversiones adicionales aunque las prestaciones no están al nivel de los segundos. En un estudio más reciente [15] sus autores muestran las altas bondades de cómputo que ofrecen los sistemas HTC a un bajo costo, pero también concluyen que para lograr altas prestaciones (equivalentes a supercomputadoras actuales) es necesario contar de un número importante de estaciones de trabajo, aunque señalan que la mayoría de los problemas que hoy en día requieren de mucho cómputo son solubles, a mejor costo, con la HTC; concluyen la investigación señalando que la decisión del uso de un tipo y otro queda más bien determinado por las posibilidades económicas y propósito u objetivo social de la institución cliente, así como la identificación de necesidades que conlleven a nuevas inversiones a corto, mediano y largo plazo.

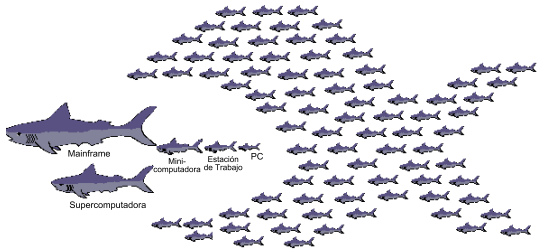

Fig. 1: **Potencialidades de la HTC según el proyecto Berkley NOW**

2.1 Sistemas HTC

En general, las plataformas HTC son sistemas que utilizan el tiempo ocioso de los procesadores, y que persiguen una efectiva administración y explotación de la mayor cantidad de recursos computacionales posible. En esta sección se presenta una revisión de los principales sistemas de este tipo reportados por la literatura especializada.

2.1.1 Berkeley Open Infrastructure for Network Computing (BOINC)

El proyecto *Search for Extraterrestrial Intelligence (SETI)* es un esfuerzo científico para la búsqueda de vida inteligente fuera de la Tierra, basado fundamentalmente en el análisis de señales electromagnéticas captadas del espacio exterior en el observatorio Arecibo [16], este procesamiento se llevaba a cabo en supercomputadoras dedicadas instaladas especialmente para ello.

En 1995 surge la idea de utilizar estaciones de trabajos para apoyar el proyecto dando lugar en 1999 a SETI@home, esfuerzo este que hasta el momento no ha encontrado señales de vida extraterrestre, pero conjuntamente a otros proyectos de computación y almacenamiento distribuido tales como GIMPS y Distributed.net demostraron la viabilidad de los recursos públicos para obtener elevados desempeños [17, 18].

Como parte de SETI@home surge BOINC, una iniciativa llevada a cabo por el *U.C. Berkeley Spaces Sciences Laboratory* para realizar computación distribuida a partir de recursos públicos [19]. Esta plataforma aunque inicialmente se concibió para apoyar a SETI@home, hoy en día[2] está siendo aplicada por 27,304 grupos de trabajo de 254 países, llegando

[2]Los datos tomados en abril-2010 de http://boinc.netsoft-online.com

a tener hasta 1,935,394 usuarios. Los 10 proyectos más significativos que hoy usan BOINC se muestran en la Tabla I ordenados por crédito[3], en la Tabla II se describe brevemente la aplicación de cada uno de estos.

Tabla I: Proyectos basados en BOINC más beneficiados

Proyecto	Crédito	PC Activas
SETI@home	70,946,426,717	284,606
MilkyWay@home	28,224,179,808	35,036
World Community Grid	23,936,476,897	204,253
Einstein@Home	18,949,904,863	171,612
Collatz Conjecture	15,556,683,096	10,925
Climateprediction.net	11,287,387,200	51,946
Rosetta@home	9,906,314,504	75,732
GPUGRID	6,914,270,163	4,526
AQUA@home	3,318,739,928	8,267
PrimeGrid	2,728,759,295	15,716

La distribución geográfica de BOINC según la potencia de cálculo de los recursos donados por voluntarios se muestra en la Fig. 2.

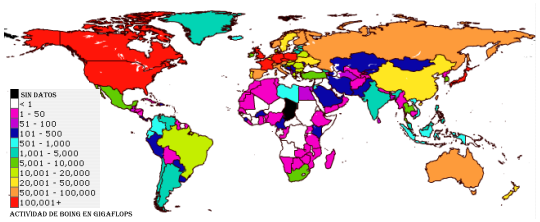

Fig. 2: **Actividad de BOINC en GigaFLOPS**

BOINC es un sistema de código abierto, programable y de propósito general, es libre y está disponible en Internet, pero tiene el inconveniente que no es multiplataforma lo que implica que solo puede usarse en aquellas para las cuales el software haya sido compilado. El mecanismo de seguridad permite la ejecución de cualquier programa (ejecutable) como tarea en las PC clientes, esto aunque brinda una mayor flexibilidad/generalidad en su uso le quita privilegios a los usuarios que brindan su PC pues no pueden limitar su ``donación´´a este nivel de seguridad, el sistema no es capaz de restringir al programador hasta donde puede llegar (*Inbuilt security mechanisms*); otro inconveniente, para algunos tipos de usuarios, es el hecho de que no permite la optimización de aplicaciones para arquitecturas específicas.

2.1.2 Condor

Condor [29] es uno de los primeros proyectos de Sistemas Distribuidos que se llevaron a cabo y uno de los que más tiempo lleva activo, ha estado operativo desde 1986. Fue desarrollado en la Universidad de Wisconsin por un grupo encabezado por Miron Livny.

Tabla II: Aplicación de algunos proyectos basados en BOINC

Proyecto	Aplicación
SETI@home	Búsqueda de Inteligencia Extraterrestre a partir del análisis de señales de radio [18].
MilkyWay@home	Modelado 3D de la Galaxia Milky Way a partir de datos del *Sloan Digital Sky Survey* [20].
Community Grid	Diseño de fármacos asistido por computadora para combatir enfermedades virales i.e Dengue, Hepatitis C, Fiebre Amarilla, etc [21].
Einstein@Home	Búsqueda de Estrellas de Neutrones Gigantes(*Pulsares*) a partir del análisis de ondas gravitacionales [22].
Collatz Conjecture	Estudios matemáticos, en específico experimentación a partir de la conjetura de Collatz [23].
Climateprediction.net	Predicción del clima de la Tierra hasta el año 2100 y chequeo de la exactitud de modelos climáticos [24].
Rosetta@home	Estudio 3D de proteínas para combatir enfermedades i.e HIV, la Malaria, el Cancer, y el Alzheimer [25].
GPUGRID	Investigaciones en la biomedicina, en específico Simulaciones Biomoleculares *all-atom* [26].
AQUA@home	Predicción del desempeño de computadoras cuánticas basado en la resolución de problemas complejos de la ciencia de los materiales y el aprendizaje automático [27].
PrimeGrid	Investigaciones a cerca de los números primos [28].

Actualmente[4] Condor cuenta con 317,925 CPUs disponibles, agrupadas en 2,331 *Pools*[5]. Su distribución geográfica se muestra en la Fig. 3.

Algunas de las principales características de Condor incluyen su capacidad de manipulación ante fallas arbitrarias en las estaciones clientes; soporte para programación sobre entornos paralelos como MPI y PVM; posibilidad para salvar el estado de la información (realización de *checkpointings*) del sistema en caso de fallas de energía o en las PC clientes; posibilidad de formar clusters virtuales (*flocking*) de unidades de cómputo para la resolución de determinados problemas; soporte para acoplarse u operar en entornos Grid.

[3]Es una medida numérica de cuanto contribuyen en CPU, almacenamiento y tráfico de red las PC clientes a un proyecto.

[4]Los datos fueron tomados de http://www.cs.wisc.edu/condor/ en abril-2010

[5]Una *Condor pool* es una colección de computadoras agrupadas virtualmente que ofrecen el poder de cómputo combinado a cualquier usuario del sistema.

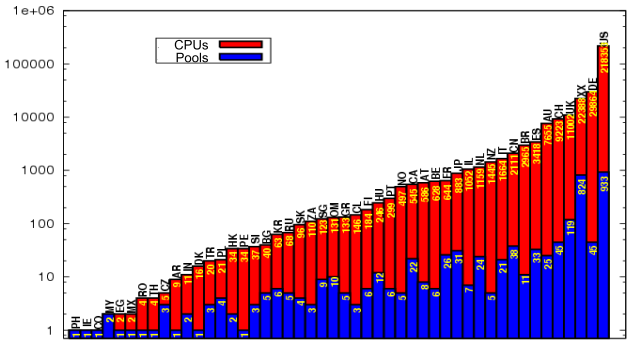

Fig. 3: ***Pools* y PC donadas a Condor por países**

A pesar de los significativos logros de Condor, este tiene algunas limitaciones. No es de código abierto; no es totalmente multiplataforma (cada solución subida a Condor debe especificar sus requerimientos), algunas de sus versiones son propias para plataformas específicas y no existen versiones para algunas arquitecturas, debido fundamentalmente a que su código no está disponible para su recompilación; la seguridad de las máquinas donadas no se consideran responsabilidad del software (*Inbuilt security mechanisms*), se limita a la que pueda establecer el usuario o administrador de la PC cuando se instala el sistema, si las políticas establecidas durante la instalación del cliente no son adecuadas la seguridad de la PC puede verse comprometida por una aplicación descargada via Condor; los trabajos pueden caer en inanición (*starvation*) sin que el usuario lo perciba, si el trabajo está programado para una Sistema Operativo específico y los *flocking* formados para este no están disponibles entonces el trabajo deberá esperar indefinidamente sin que el usuario tenga conocimiento del problema.

2.1.3 Java Distributed Computation Library (JDCL) y sus extensiones

El sistema original JDCL fue presentado en [30]. Diseñado para proveer una plataforma fácil de usar por los programadores que necesitaban implementar de forma rápida cómputo distribuido siguiendo el modelo SPMD. Esta plataforma resolvió los problemas asociados al código nativo generado para cada sistema operativo o arquitectura, al tiempo que brindaba una interfaz de programación fácil de usar. Sin embargo presentaba un inadecuado tratamiento de excepciones, carencia de una interfaz de usuario, ausencia de mecanismos de seguridad y permitía solo la ejecución de una sola tarea aunque existieran recursos disponibles para llevar a cabo varias simultáneamente.

Luego en [31] se corrigieron algunas deficiencias y se adicionaron otras características tales como la posibilidad de llevar un reporte completo de la ejecución tanto de los clientes como del servidor dando la posibilidad del seguimiento de fallos arbitrarios en ambos lados, implementación de mecanismos de seguridad en el cliente y el aumento del rango de aplicaciones posibles a desarrollar mediante la emulación de una arquitectura MIMD paralela. Sin embargo conservó algunas limitaciones: la política de planificación no permite la ejecución paralela de más de una computación: resolución de varios problemas o varias instancias de uno mismo; la granularidad de las tareas es estática y no puede ser ajustada durante la ejecución de una tarea según el comportamiento de las máquinas clientes; no brindaba una interfaz de usuario que permita administrar el servidor y no existe la posibilidad de actualizar el cliente de forma remota.

Este trabajo dio pie al desarrollo, presentado finalmente en [32], de un sistema con las siguientes características: en términos de seguridad se consideró al sistema responsable del uso, establecido por el programador de la tarea, de las PC clientes (*Inbuilt security mechanisms*); registro de información que posibilita, en dependencia de la disponibilidad y tipo de los recursos, ejecutar la tarea más adecuada; es multiplataforma e independiente del tipo de red de interconexión; brinda mecanismos que abstraen al programador de los fallos que puedan ocurrir en las máquinas donadas; el software instalado en los clientes pueden ser actualizados de forma remota; la granularidad de las tareas puede ser ajustada durante su ejecución, y se hizo posible la ejecución de varios trabajos de forma concurrente, incluso el establecimiento de prioridades entre ellos, sin embargo quedaron algunas limitaciones en cuanto a escalabilidad: la concurrencia que generaban los clientes en el servidor comprometía su rendimiento, integración con otros entornos paralelos o distribuidos y la imposibilidad del configurar el cliente para responder ante nuevos servidores.

En [33–35] se continua este trabajo y se erradican las limitaciones antes planteadas. Se logró un sistema, llamado T-arenal, con todas las ventajas de su predecesor y además escalable y con la posibilidad de integración con clusters dedicados y entornos Grid.

2.1.4 FightAIDS@Home

FightAIDS@Home [36] es el primer proyecto biomédico de computación distribuida. Se desarrolla en el Laboratorio Olson del *The Scripps Research Institute* en La Jolla, California. Su objetivo es la realización de investigaciones en el descubrimiento de

fármacos y la estructura biológica del virus la proteasa del VIH.

Ha permitido el estudio del mecanismo de resistencia a los tratamientos multidrogas que el VIH usa para evadir los actuales fármacos que intentan combatirlo. Algunos de los resultados obtenidos han sido publicados en [37, 38], el estado de los proyectos que actualmente están en progreso puede verse en http://fightaidsathome.scripps.edu/status.

En la literatura consultada refieren los éxitos alcanzados por este sistema pero no se aborda la posibilidad de emplearlo para la realización de cualquier tipo de cómputo, por lo que sugiere ser cauteloso al catalogarlo como de propósito general.

2.1.5 Otros

Otras plataformas de este tipo que pueden referenciarse, por ejemplo, propietarias y de código cerrado Entropia, GIMPS: The Great Internet Mersenne Prime Search, Distributed.net y United Devices de Intel [13, 39–41].

De código abierto pueden mencionarse JXTA, XtremWeb, Javelin, Charlotte, ATLAS, ParaWeb, JavaParty y JCluster [42–48].

3. Aplicación en la Industria Cubana del Níquel

La Industria Cubana del Níquel desde hace varias décadas ha impulsado la investigación en el modelado [49], exploración [50] y evaluación geólogo-económica de yacimientos lateríticos [51]. Todos estos esfuerzos han sido desarrollados con un objetivo común: optimizar el proceso metalúrgico en general que permita lograr una industria puntera a nivel mundial tanto en niveles de producción como en calidad y costos [52].

La complejidad de los yacimientos lateríticos, en cuanto a la composición química y mineralógica, es bien conocida [53] por lo que es un reto el modelado matemático de la estructura de estos tipos de yacimientos. Existen varios enfoques que persiguen lidiar con este inconveniente [54].

En [55] se presenta el modelo (3), con el que se persigue describir fielmente la realidad pretendiendo alcanzar una alta representatividad, es de tipo probabilístico, en específico markoviano. Se basa en la existencia de tipos o clases de materiales patrones en los yacimientos lateríticos, asumiendo que el conjunto de estas contiene todas las variables presentes en ellos y que representa el espacio muestral [56]. Cada clase se asume como uno de los estados en un proceso de Markov, teniendo como base una elemental Matriz de Probabilidades de Transición (MPT). Formalmente una matriz de transición M se define como:

$$M = \begin{pmatrix} p_{11} & p_{12} & \cdots & p_{1n} \\ p_{21} & p_{22} & \cdots & p_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ p_{n1} & p_{n2} & \cdots & p_{nn} \end{pmatrix}$$

donde n es la cantidad de estados en los cuales puede estar el proceso a modelar (en nuestro caso la cantidad de clases), y p_{ij} la probabilidad de que el sistema pase del estado i al j, M cumple que:

$$\forall i, j \in [1, n] : 0 \leq p_{ij} \leq 1 \tag{1}$$

$$\forall i \in [1, n] : \textstyle\sum_{j=1}^{n} p_{ij} = 1 \tag{2}$$

En la propuesta [57] esta MPT se asocia a una matriz de particularidades espaciales y se obtiene:

$$X(t, r) = \sum_{i=1}^{n} \pi(i)\rho(t, r) \tag{3}$$

Las matrices de transición $\pi(i)$ son matrices de probabilidades condicionales que en (3) están asociadas a características temporales[6] y geoespaciales $\rho(t, r)$, por lo que para toda combinación r de valores posibles de las variables que describen la dependencia entre las características observadas para cada par de puntos de muestreo en el momento t se tiene una MPT elemental de $k \times k$, donde k es la cantidad de clases que caracterizan a los yacimientos lateríticos según [56]. Las variables que se consideraron en r en la propuesta inicial de (3) son: dirección horizontal (α_H) y vertical (α_V), profundidad (H) y distancia entre los puntos de muestreo (P), las que unidas a la característica observada en cada uno de los dos puntos de muestreo determinan el hipercubo de probabilidades de transición que representa el modelo.

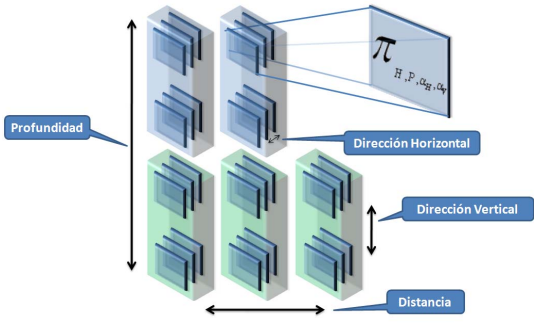

Fig. 4: **Abstracción del modelo**

[6] El tiempo es considerado discreto, cada iteración completa sobre el ciclo ORENI representa una unidad de tiempo

3.1 Resultados

En [58] se presenta un algoritmo paralelo escalable y óptimo en cuanto ganancia de velocidad y eficiencia para el cálculo MTPs (ver **Algoritmo 1**) y se prueban sus prestaciones empleándolo para la aplicación de (3) sobre un cluster Beowulf. El método mostró sus bondades para explotar, de manera exhaustiva, los recursos computacionales disponibles. El presente trabajo emplea ese algoritmo para modelar yacimientos, intentando medir y explorar las potencialidades de los entornos no dedicados. Se utilizó el sistema distribuido **T-arenal** para la implementación de las dos variantes del algoritmo determinadas por la estructura de datos empleada para representar (3) y propuestas en [58]; la primera de ellas (Variante Clásica) emplea una estructura de datos estándar y la segunda (Variante AVL) un árbol AVL para la representación del hipercubo de probabilidades condicionales. En ese trabajo se muestra que la Variante Clásica es aproximadamente 2 veces más rápida que la Variante AVL.

Para desarrollar el presente estudio se usaron hasta 30 estaciones de trabajo conectadas mediante una red local de 100 Mbps, todas con un procesador Intel (R) Core (TM) 2 Duo CPU E4500 2.20GHz, memoria principal de 1 (2 x 512) GB de RAM, y sistema operativo Windows XP.

Veamos cuales fueron los resultados arrojados. Puede apreciarse en la Fig. 5 y Fig. 6, las que muestran respectivamente las mayores ganancias de velocidad y eficiencia alcanzadas por el sistema durante las pruebas[7], que los tiempos fueron reducidos hasta 11.3 veces (de 21 horas y 5 minutos a 112 minutos) y la eficiencia llegó hasta un 51 %.

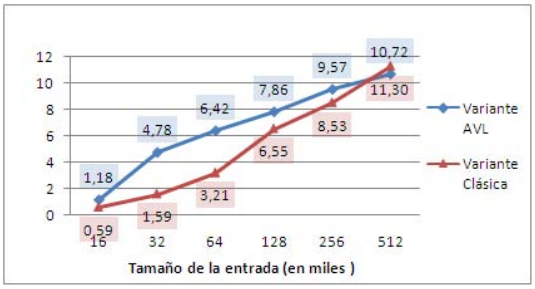

Fig. 5: **Ganancia máxima**

En las Fig. 7 y Fig. 8 se muestra el comportamiento medio del algoritmo durante las pruebas.

Como puede apreciarse tanto la ganancia de velocidad como la eficiencia crecen al tiempo en que lo hace el tamaño de la entrada, la misma tendencia

[7] Se realizaron 360 ejecuciones (180 por variante), 60 por cada tamaño de la entrada

Algoritmo 1 Algoritmo paralelo

$$c_i \in \begin{cases} B_{iDIV(n/2p)} & \text{si } i < n/2 \\ B_{(n-i-1)DIV(n/2p)} & \text{si } i \geq n/2 \end{cases}$$

Entrada: Registro O de n observaciones
Salida: Matriz P de probabilidad de transición

EN PARALELO:
for $pr = 1, 2, \cdots, p-1$ **do**
  EN pr :
  //FASE I: Cada procesador pr calcula su matriz P^0_{pr}
  for $i := pr\ n/2p$ to $(pr+1)n/2p$ **do**
    for $j := i+1$ to n **do**
      $e_O \leftarrow estado\_del\_proceso(O_i)$
      $e_D \leftarrow estado\_del\_proceso(O_j)$
      if $existe\_transicion(e_O, e_D)$ **then**
        $ajustar\_probabilidad\_en\_P(e_O, e_D)$
      end if
      if $existe\_transicion(e_D, e_O)$ **then**
        $ajustar\_probabilidad\_en\_P(e_D, e_O)$
      end if
    end for
  end for
  for $i := n(1-(pr+1)/2p)$ to $n(1-pr/2p)-1$ **do**
    for $j := i+1$ to n **do**
      $e_O \leftarrow estado\_del\_proceso(O_i)$
      $e_D \leftarrow estado\_del\_proceso(O_j)$
      if $existe\_transicion(e_O, e_D)$ **then**
        $ajustar\_probabilidad\_en\_P(e_O, e_D)$
      end if
      if $existe\_transicion(e_D, e_O)$ **then**
        $ajustar\_probabilidad\_en\_P(e_D, e_O)$
      end if
    end for
  end for
  //FASE II: Cálculo de $P = \sum_{i=0}^{p-1} P^0_i = P^{\log p}_0$
  for $i := 1$ to $\log n$ **do**
    if $pr\ MOD\ 2^i = 0$ **then**
      Recibir $Matriz$
      $P^i_{pr} = P^{i-1}_{pr} + Matriz$
    else
      Enviar P^i_{pr} a p_k donde $k = (pr + 2^{(i-1)}) MOD\ p$
      Terminar
    end if
  end for
  if $pr = 0$ **then**
    return $P^{\log p}_{pr}$
    Terminar
  end if
end for

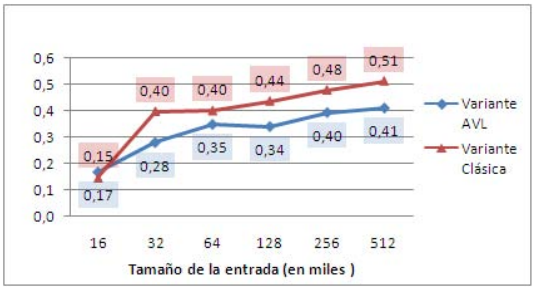

Fig. 6: **Eficiencia máxima**

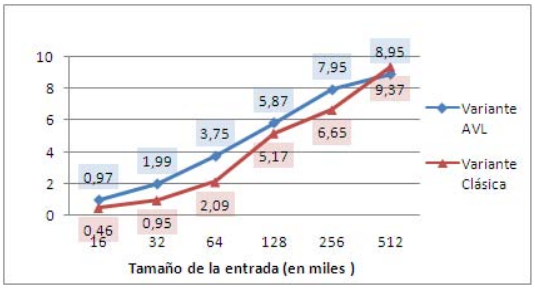

Fig. 7: **Ganancia media**

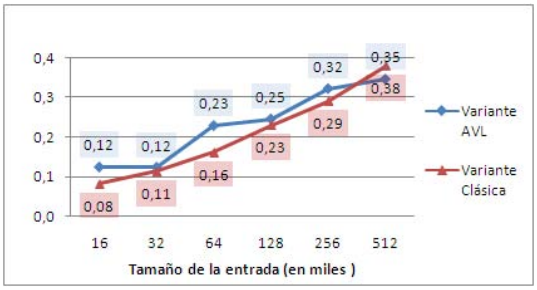

Fig. 8: **Eficiencia media**

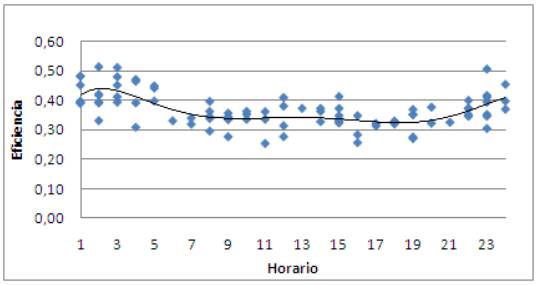

Fig. 9: **Eficiencia en función del horario**

que muestra el algoritmo cuando se ejecuta en un entorno dedicado, lo que significa que la implementación realizada siguiendo la HTC también es más ventajosa aplicarla, y usa los procesadores disponibles de forma más efectiva, para problemas de mayor dimensión. Los índices de eficiencia alcanzados permiten concluir que, a pesar de ser no dedicados los elementos computacionales, es considerablemente alto el tiempo que están ociosos, una eficiencia máxima de $0{,}51$ en un algoritmo óptimo en cuanto a este parámetro, indica que aproximadamente la mitad del tiempo las estaciones de trabajo estuvieron disponibles y en función de la ejecución del algoritmo.

Otro parámetro interesante de analizar para sistemas de este tipo, debido a que las unidades de cómputo son no dedicadas, es cómo influye el horario en la ejecución del algoritmo, específicamente en la eficiencia del sistema para realizar el cómputo.

En la Fig. 9, la que muestra la eficiencia alcanzada para $n = 512$ mil, puede comprobarse que los horarios más provechosos para la realización de los cálculos fueron la madrugada, las horas finales de la noche y en cierto grado el horario del medio día.

De forma semejante, veamos cómo influye el tamaño del problema junto a la disponibilidad de las estaciones de trabajo en la ejecución del algoritmo. Para ello analicemos la desviación estándar del tiempo de respuesta observado durante las pruebas, como puede notarse en la Fig. 11 su valor aumentó a medida que lo hacía el tamaño del problema, esto es debido a que la probabilidad de perder estaciones de trabajo (Fig. 10) durante la ejecución del algoritmo aumenta a medida en que lo hace el tiempo necesario para resolver el problema.

Fig. 10: **Número de PC utilizadas (DesvEst)**

En general se obtuvieron mejores resultados al usar un árbol AVL para representar el hipercubo de probabilidades condicionales, lo que pudiera sugerir una contradicción con lo planteado en [58], sin embargo, en esa investigación se hizo la salvedad que a pesar de que el tiempo total del algoritmo empleando el árbol sería mayor, el comportamiento de ese parámetro podía disminuir bajo ciertas condiciones. La razón principal de estos resultados, aparejado al cumplimiento de estas premisas, está en la arquitectura del sistema HTC empleado unido al grado de dispersión del modelo. **T-arenal** tiene una arquitectura cliente-servidor que permitió, sirviéndose de los altos grados de dispersión del modelo (siempre por encima del 87 %), realizar las operaciones de costo

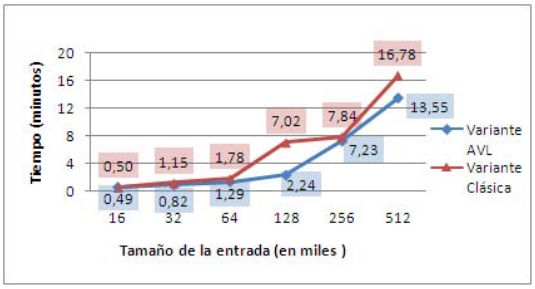

Fig. 11: **Tiempos de respuesta (DesvEst)**

constante en el propio servidor logrando liberar al algoritmo original de operaciones que influían, según la estructura de datos empleada, en su costo (ver Fase II del **Algoritmo 1**).

4. Conclusiones

Se presentó una revisión del estado del arte de la computación de alta productividad, los principales sistemas y algunos estudios que se han realizado sobre sus potencialidades para resolver problemas de alto costo computacional. Se realizó una implementación siguiendo este paradigma de un algoritmo paralelo que evidenció que las redes de estaciones de trabajo no dedicadas son una opción viable para la Industria Cubana del Níquel a la hora de modelar yacimientos lateríticos. Los resultados muestran que aproximadamente la mitad del tiempo ocioso de estas máquinas son aprovechados por estos sistemas cuando se emplea un algoritmo óptimo en cuanto a ganancia de velocidad y eficiencia.

REFERENCIAS BIBLIOGRÁFICAS

[1] A. Tanenbaum, *Distributed Operating Systems*. NJ: Prentice Hall, 1995.

[2] A. Puder, K. RÖMER, y F. PILHOFER, *Distributed Systems Architecture. A Middleware Approach*. San Francisco: Morgan Kaufmann publishers is an imprint of Elsevier, 2006.

[3] J. Wu, *Distributed System Design*. CRC-Press, 1998.

[4] A. Beck, "High throughput computing: An interview with miron livny," 1997. [Online]. Disponible en: http://www.cs.wisc.edu/condor/HPCwire.1

[5] R. Arpaci, A. Dusseau, A. Vahdat, L. Liu, T. Anderson, y D. Patterson, "The interaction of parallel and sequential workloads on a network of workstations," en *Proceedings of the 1995 ACM SIGMETRICS joint international conference on Measurement and modeling of computer systems*. ACM, 1995, pp. 267–278.

[6] A. Acharya, G. Edjlali, y J. Saltz, "The utility of exploiting idle workstations for parallel computation," *ACM SIGMETRICS Performance Evaluation Review*, vol. 25, no. 1, pp. 225–234, 1997.

[7] R. Wolski, N. Spring, y J. Hayes, "The network weather service: A distributed resource performance forecasting service for metacomputing," *Future Generation Computer Systems*, vol. 15, no. 5-6, pp. 757–768, 1999.

[8] S. Smallen, H. Casanova, y F. Berman, "Tunable on-line parallel tomography," en *Proceedings of SuperComputing 01, Denver, Colorado*, 2001.

[9] H. Casanova, D. Zagorodnov, F. Berman, y A. Legrand, "Heuristics for scheduling parameter sweep applications in grid environments," en *Proceedings of the 9th Heterogeneous Computing Workshop (HCW 00)*. IEEE Computer Society, 2000, p. 349.

[10] D. Kondo, M. Taufer, C. Brooks, H. Casanova, y A. Chien, "Characterizing and evaluating desktop grids: An empirical study," en *Parallel and Distributed Processing Symposium, 2004. Proceedings. 18th International*, 2004.

[11] P. Domingues, P. Marques, y L. Silva, "Resource usage of windows computer laboratories," en *Parallel Processing, 2005. ICPP. International Conference Workshops on Parallel Processing*, Oslo, Norway, 2005, pp. 469–476.

[12] M. Vlădoiu, Z. Constantinescu, y C. NegoiŃă, "Availability of computational resources for desktop grid computing," *Bulletin of PG University of Ploiesti, BMIF - Mathematics, Informatics, Physics Series*, 2009.

[13] A. Chien, B. Calder, S. Elbert, y K. Bhatia, "Entropia: architecture and performance of an enterprise desktop grid system," *Journal of Parallel and Distributed Computing*, vol. 63, no. 5, pp. 597–610, 2003.

[14] D. Heap, "Taurus-a taxonomy of actual utilization of real unix and windows servers," *IBM White Paper GM12-0191*, 2003.

[15] D. Kondo, B. Javadi, P. Malecot, F. Cappello, y D. P. Anderson, "Cost-benefit analysis of cloud computing versus desktop grids," *Parallel and Distributed Processing Symposium, International*, vol. 0, pp. 1–12, 2009. [Online]. Disponible en: http://doi.ieeecomputersociety.org/10.1109/IPDPS.2009.5160911

[16] S. Shostak, "Sharing the universe- Perspectives on extraterrestrial life," *Berkeley, CA: Berkeley Hills Books*, 1998.

[17] E. Korpela, D. Werthimer, D. Anderson, J. Cobb, y M. Leboisky, "Seti@ home-massively distributed computing for seti," *Computing in Science and Engineering*, vol. 3, no. 1, pp. 78–83, 2001.

[18] D. Anderson, J. Cobb, E. Korpela, M. Lebofsky, y D. Werthimer, "SETI@ home: an experiment in public-resource computing," *Communications of the ACM*, vol. 45, no. 11, p. 61, 2002.

[19] D. Anderson, "BOINC: A system for public-resource computing and storage," en *proceedings of the 5th IEEE/ACM International Workshop on Grid Computing*. IEEE Computer Society, 2004, pp. 4–10.

[20] N. Cole, T. Desell, D. Lombraña González, F. Fernández de Vega, M. Magdon-Ismail, H. Newberg, B. Szymanski, y C. Varela, "Evolutionary Algorithms on Volunteer Computing Platforms: The MilkyWay@ Home Project," *Parallel and Distributed Computational Intelligence*, pp. 63–90, 2010.

[21] W. C. Grid. (2010) World community grid and the discovering dengue drugs. [Online]. Disponible en: http://www.worldcommunitygrid.org/

[22] B. Abbott, R. Abbott, R. Adhikari, P. Ajith, B. Allen, G. Allen, R. Amin, D. Anderson, S. Anderson, W. Anderson *et al.*, "Einstein@ Home search for periodic gravitational waves in LIGO S4 data," *Physical Review D*, vol. 79, no. 2, p. 22001, 2009.

[23] C. Conjecture. (2010) Research project that uses internet-connected computers to do research in mathematics. [Online]. Disponible en: http://boinc.thesonntags.com/collatz/

[24] D. Stainforth, J. Kettleborough, A. Martin, A. Simpson, R. Gillis, A. Akkas, R. Gault, M. Collins, D. Gavaghan, y M. Allen, "Climateprediction. net: Design principles for public-resource modeling research," en *Proceedings of the 14th IASTED International Conference on Parallel and Distributed Computing Systems*. Citeseer, 2002.

[25] R. Das, B. Qian, S. Raman, R. Vernon, J. Thompson, P. Bradley, S. Khare, M. Tyka, D. Bhat, D. Chivian *et al.*, "Structure prediction for CASP7 targets using extensive all-atom refinement with Rosetta@ home," *PROTEINS-NEW YORK*, vol. 69, p. 118, 2007.

[26] I. Buch, M. Harvey, T. Giorgino, D. Anderson, y G. De Fabritiis, "High-Throughput All-Atom Molecular Dynamics Simulations Using Distributed Computing," *Journal of Chemical Information and Modeling*, vol. 50, no. 3, pp. 397–403, 2010.

[27] D.-W. S. Inc. (2010) D-wave the quantum computing company:aqua@home. [Online]. Disponible en: http://aqua.dwavesys.com/

[28] R. Slatkevic y P. community, "Primegrid: bring the excitement of prime finding to the ``everyday´´computer user," 2010. [Online]. Disponible en: http://www.primegrid.com/

[29] D. Thain, T. Tannenbaum, y M. Livny, "Distributed computing in practice: The condor experience." *Concurrency and Computation Practice and Experience*, vol. 17, no. 2-4, pp. 323–356, 2005.

[30] K. Fritsche, J. Power, y J. Waldron, "A Java distributed computation library," en *Proc. 2nd International Conference on Parallel and Distributed Computing, Applications and Technologies*. Citeseer, 2001, pp. 236–243.

[31] T. Keane, R. Allen, T. Naughton, J. McInerney, y J. Waldron, "Distributed Java platform with programmable MIMD capabilities," *Scientific Engineering for Distributed Java Applications*, pp. 122–131, 2003.

[32] T. Keane, "A general purpose heterogeneous distributed computing system," Tesis de maestría, National University of Ireland, 2004.

[33] A. Page, T. Keane, R. Allen, T. J. Naughton, y J. Waldron, "Multitiered distributed computing platform." en *Second International Conference on the Principles and Practice of Programming in Java*, J. F. P. editions y J. T. Waldron, Eds., Kilkenny City, Ireland, 2003, p. 191.

[34] L. Aguilera, "Sistema de cómputo distribuido aplicado a la bioinformática," Tesis de Maestría, Universidad de las Ciencias Informática, Ciudad de La Habana, Cuba, 2008.

[35] C. R. Jacas, L. Aguilera, y D. Mariño, "Platform of distributed task v2.0," Laboratorio Nacional de Computación, Petrópolis, Rio de Janeiro, Brasil, 2010.

[36] A. Olson *et al.*, "FightAIDS@ Home: Accelerate AIDS Research by Deploying Global Grid of Distributed Computing Power."

[37] M. W. Chang, W. Lindstrom, A. J. Olson, y R. K. Belew, "Analysis of HIV Wild-Type and Mutant Structures via in Silico Docking against Diverse Ligand Libraries," *Journal of Chemical Information and Modeling*, vol. 47, no. 3, pp. 1258–1262, 2007.

[38] G. Morris, R. Huey, W. Lindstrom, M. Sanner, R. Belew, D. Goodsell, y A. Olson, "AutoDock4 and AutoDockTools4: Automated docking with selective receptor flexibility," *Journal of Computational Chemistry*, vol. 30, no. 16, pp. 2785–91, 2009.

[39] G. Woltman y S. Kurowski, "The great internet mersenne prime search," 2004. [Online]. Disponible en: http://www.mersenne.org.prime.htm

[40] Distributed.net, "The fastest computer on earth," 1997. [Online]. Disponible en: http://www.distributed.net/

[41] Intel®, "Intel-united devices cancer research project," 2010. [Online]. Disponible en: http://www.ud.com

[42] L. Gong, "JXTA: A network programming environment," *IEEE Internet Computing*, vol. 5, no. 3, pp. 88–95, 2001.

[43] G. Fedak, C. Germain, V. Neri, y F. Cappello, "Xtremweb: A generic global computing system," en *First IEEE/ACM International Symposium on Cluster Computing and the Grid, 2001. Proceedings*, 2001, pp. 582–587.

[44] B. Christiansen, P. Cappello, M. Ionescu, M. Neary, K. Schauser, y D. Wu, "Javelin: Internet-based parallel computing using Java," *Concurrency Practice and Experience*, vol. 9, no. 11, pp. 1139–1160, 1997.

[45] M. Neary, A. Phipps, S. Richman, y P. Cappello, "Javelin 2.0: Java-based parallel computing on the Internet," en *Euro-Par 2000 Parallel Processing*. Springer, 2000, pp. 1231–1238.

[46] M. Philippsen y M. Zenger, "JavaParty- transparent remote objects in Java," *Concurrency Practice and Experience*, vol. 9, no. 11, pp. 1225–1242, 1997.

[47] B. Haumacher, T. Moschny, J. Reuter, y W. Tichy, "Transparent distributed threads for Java," en *Parallel and Distributed Processing Symposium, 2003. Proceedings. International*, 2003, p. 7.

[48] B. Zhang, G. Yang, y W. Zheng, "Jcluster: an efficient Java parallel environment on a large-scale heterogeneous cluster," *Concurrency and Computation*, vol. 18, no. 12, p. 1541, 2006.

[49] O. Gómez, E. Estévez, y J. Q. Cuador, "Modelaje geológico y de recursos del yacimiento "pastelilloütilizando el krigeaje de indicadores," en *I Convención Cubana de Ciencias de la Tierra, VI Congreso de Geología (GEO 14-20)*, 2005.

[50] A. A. Legrá, "Metodología para el pronóstico, planificación y control integral de la minería en yacimientos lateríticos," Ph D, Instituto Minero-Metalúrgico de Moa, 1999.

[51] J. Arias, M. Pérez, y M. Campo, "Determinacion de la continuidad de la mineralizacion del horizonte de serpentinitas duras (sd) en el yacimiento yamaniguey," en *I Convención Cubana de Ciencias de la Tierra, I Congreso de Minera (MIN3-6)*, 2005.

[52] Y. García, "Balance anual del grupo empresarial cubaníquel," Moa, Holguín, 2010.

[53] A. Vera Yeste, *Introducción a los yacimientos de níquel cubanos*. Ciudad de la Habana: ORBE, 1979.

[54] L. O. Vera Sardinas, "Procedimiento para la determinación de las redes racionales de exploración de los yacimientos lateríticos de níquel y cobalto en la región de moa," Tesis doctoral, ISMM, 2001.

[55] R. E. Peña, "Modelo matemático para la optimización de las redes de exploración y explotación en yacimientos lateríticos," en *II Convención Cubana de Ciencias de la Tierra*, La Habana, Cuba, 2007.

[56] R. E. Peña, L. Matos, E. Ortiz, y V. Robles, "Propuesta de clases patrones en yacimientos lateríticos ferro-niquelíferos," en *II Convención Cubana de Ciencias de la Tierra*, La Habana, Cuba, 2007.

[57] R. E. Peña, "Algoritmo de conteo para modelos markovianos en yacimientos lateríticos," en *COMPUMAT*, La Habana, Cuba, 2007.

[58] D. Trinchet y A. Guirola, "Algoritmo paralelo para el cálculo de matrices de transicion. Aplicación en el modelado de yacimientos lateríticos." en *Proceedings of Latin American Conference on High Performance Computing (CLCAR 2010)*, Rio Grande do Sur, Brasil, 2010.