IX Jornadas de Ciencia y Tecnología
de Facultades de Ingeniería del NOA
Santiago del Estero, 3 y 4 de Octubre de 2013

CluSim: Evolución del entorno de Simulación para Clusters

Nilda M. Pérez Otero[1], Adelina García[1], C. Marcelo Pérez Ibarra [1]& Abigaíl R. N. Verazay[1]

(1) Grupo de Ingeniería del Software, Facultad de Ingeniería, Universidad Nacional de Jujuy. {adelinagarcia15, nilperez, cmperezi, abigailrn}@gmail.com

RESUMEN: La demanda de poder computacional generada por las aplicaciones de alto rendimiento promovió el desarrollo de centros de cómputos de altas prestaciones. En el diseño y desarrollo de estos sistemas los simuladores resultan de gran utilidad para el desarrollo de técnicas y mecanismos de optimización de rendimiento. En este marco, el GIS desarrolló un simulador de cluster basado en OMNeT++ que permite evaluar y predecir el impacto en el rendimiento del sistema considerando distintas configuraciones de cluster. Desde 2010 y hasta el presente se realizaron a CluSim varias extensiones y modificaciones con el objeto de mejorar su modelo y ampliar el rango de aplicaciones que permite simular. En este artículo se presenta la evolución de CluSim desde el prototipo preliminar, el proceso de modelado seguido y la última propuesta de implementación de tolerancia a fallos al modelo.

1 INTRODUCCIÓN

En junio de 2013 se publicó en Top500.org la nueva lista de los 500 centros de supercomputación más importantes del mundo. La mayoría de las supercomputadoras incluidas en este ranking cuentan con cientos de miles de cores estando lideradas por Tianhe-2 (China) un cluster con más de 3 millones de cores y una memoria mayor a un millón de GB que, con el benchmark Linpack, alcanzó un rendimiento de 33,86 Pflops. En el diseño y desarrollo de estos sistemas, el modelado de rendimiento sustentado en la simulación es indispensable para evaluar las opciones de diseño y ayudar a optimizar el rendimiento de procesadores, red de interconexión y, finalmente, todo el sistema, incluyendo software y aplicaciones de High Performance Computing (HPC) (Denzel et al., 2007).

En la literatura se encuentran muchos trabajos que tratan la simulación de grandes redes y aplicaciones HPC. Núñez y colegas (2010) hacen una recopilación donde mencionan, entre otros:

- un simulador de Redes de Área de Almacenamiento que permite simula fallos en enlaces y switches, canales virtuales, diferentes algoritmos de ruteo, etc.,
- un entorno de simulación de eventos discretos, que simula entidades y constructores de mensajes de comunicación entre entidades.

También en (Nuñez et al., 2010) presentan SIMCAN, un entorno de simulación para grandes redes complejas de almacenamiento que permite simular estas redes y sus subsistemas subyacentes correspondientes (I/O, networking, etc.).

Además, es posible encontrar entornos de simulación de redes de propósito general que permiten crear diferentes configuraciones de redes, con diferentes tipos de nodos, switches, topologías, protocolos, etc. Ejemplos de éstos son OPNET Modeler (www.opnet.com) y OMNeT++ (www.omnetpp.org). Este es el caso de CluSim, un simulador de clusters desarrollado por el Grupo de Investigación de Ingeniería de Software (GIS) de la Facultad de Ingeniería (FI) de la Universidad Nacional de Jujuy (UNJu) que durante los últimos años, se ha centrado en el estudio de aplicaciones HPC, particularmente, en el análisis del rendimiento de aplicaciones paralelas que se ejecutan en clusters de computadoras.

CluSim es un framework basado en OMNeT++ que permite simular distintas configuraciones de un cluster para aplicaciones HPC, implementando los módulos de la arquitectura de forma parametrizable y configurable (Valdiviezo et al., 2010). De este modo, CluSim permite analizar el impacto de distintas configuraciones de un sistema paralelo en el rendimiento de aplicaciones HPC, haciendo posible determinar qué configuraciones resultan más adecuadas para cada tipo de aplicación.

El propósito de este artículo es presentar la evolución de CluSim desde la formulación del modelo inicial en el año 2010, pasando por sus distintas implementaciones, hasta el modelo de tolerancia a fallos propuesto en 2013.

El resto de este artículo se estructura de la siguiente manera, en el apartado 2 se presenta la caracterización de aplicaciones master/worker (M/W), en el apartado 3 se describe el prototipo inicial de CluSim, en el apartado 4 se detalla el proceso de modelado de aplicaciones y su utilización en la reformulación del modelo de CluSim, en el apartado 5 se presenta la extensión de Clu-

Sim a aplicaciones SPMD (Single Program Multiple Data), Pipeline y Divide/Conquer (D/C), en el apartado 6 se propone un modelo de tolerancia a fallos (TF) para CluSim. Finalmente se enuncian las conclusiones del trabajo.

2 CARACTERIZACIÓN DE APLICACIONES M/W

La configuración de recursos de un cluster y el tipo de aplicaciones que se ejecuten en él afectan directamente el rendimiento del sistema completo. El estudio de la arquitectura que soporta el cómputo paralelo y de las características de la aplicación (distribución de datos y modelo de comunicación) contribuye a establecer qué configuración de recursos es más adecuada para cada tipo de aplicación. En este apartado se presenta el estudio de la distribución de datos y el modelo de comunicación que permitieron caracterizar los patrones de cómputo y comunicación de aplicaciones M/W (Valdiviezo et al., 2010)

Este estudio se realizó en el cluster del Laboratorio Universitario de Tecnologías Informáticas (LUTI) de la FI de la UNJu, que contaba con 10 PC's con las siguientes características:

- Procesador Intel Core 2 Duo
- 2 GB de RAM
- 120 GB de disco duro
- Placa Ethernet 10/100 Mb
- Sistema Operativo: Linux Ubuntu 9.04
- Librería de Paso de Mensajes: MPICH2
- Librerías Adicionales: librería MPE y visualizador Jumpshot.

La librería MPE (MPI Parallel Environment) incluida en MPICH2 permite, entre otras utilidades, recopilar información acerca del comportamiento de programas MPI mediante la generación de trazas. La herramienta Jumpshot permite visualizar los archivos de traza generados con la librera MPE.

Para la caracterización se seleccionó una aplicación representativa del paradigma M/W: Producto de Matrices Cuadradas con Asignación Dinámica. El análisis gráfico de la traza obtenida de la ejecución de la aplicación permitió determinar los parámetros (tiempos) que se muestran en la Tabla 1. La traza analizada de dónde se obtuvieron estos tiempos se presenta en la Fig. 1.

Para determinar si el comportamiento de la aplicación presentaba alguna tendencia, se generaron las trazas de ejecución de la aplicación para diferentes tamaños de matriz y bloque. El análisis estadístico de los valores obtenidos de estas trazas permitió suponer que los tiempos estudiados podrían corresponder a dos tipos de distribuciones: uniforme o normal.

En el siguiente apartado se presenta la primera versión de CluSim y los resultados obtenidos de configurarlo con los parámetros identificados con el objetivo de determinar cuál de las dos distribuciones se asimilaba más al comportamiento real.

Tabla 1. Tiempos identificados a partir de la traza de la aplicación objetivo.

Parámetro	Tiempos Master/Worker
t_ini (1)	tiempo que transcurre entre el aviso de inicio de tarea y envío del bloque de la primera matriz.
t_matrices (2)	tiempo que transcurre entre el envío del bloque la primera matriz y el de la segunda matriz.
t_nodos (3)	tiempo que transcurre entre el envío de tareas a los workers.
t_computo (4)	tiempo de cómputo de la tarea.
t_aviso (5)	tiempo que transcurre entre el envio de aviso yel de resultados del worker(i) al master.
t_trabajo (6)	tiempo que transcurre entre la recepción de resultados del worker y el envío de una nueva tarea.

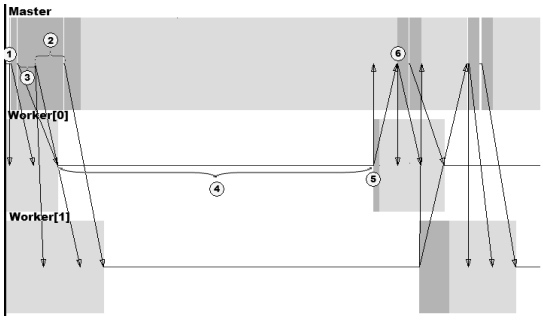


Figura 1. Segmento de una traza de la aplicación objetivo.

3 PROTOTIPO INICIAL DE CLUSIM

Una vez finalizada la caracterización de la aplicación M/W, se procedió a implementar CluSim en OMNeT++.

En primer lugar, se definieron los módulos simples y compuestos que forman la estructura del modelo de red y se especificaron los módulos que representan los nodos master y worker del cluster:

- El módulo simple master especifica los tiempos que caracterizan la comunicación de la aplicación (t_ini, t_matrices, t_nodos, t_trabajo, t_ini) y el número de bloques que corresponden al tamaño de la matriz.
- El módulo simple worker especifica los parámetros t_computo y t_aviso que caracterizan al tiempo de cómputo de la aplicación. En la sección gates de ambos módulos se indican las puertas de conexión hacia otros módulos.

Especificados master y worker se define el modelo de red: numWorkers indica, al momento de iniciar la simulación, la cantidad de nodos workers del cluster, también se define un canal de comunicación llamado channels que es el responsable emular las características típicas del canal de comunicación (data rate, delay, entre otros) y por último, en la sección connections se define la topología de la red, que en este caso es estrella. La Fig. 2 muestra el modelo completo master-worker-red de interconexión.

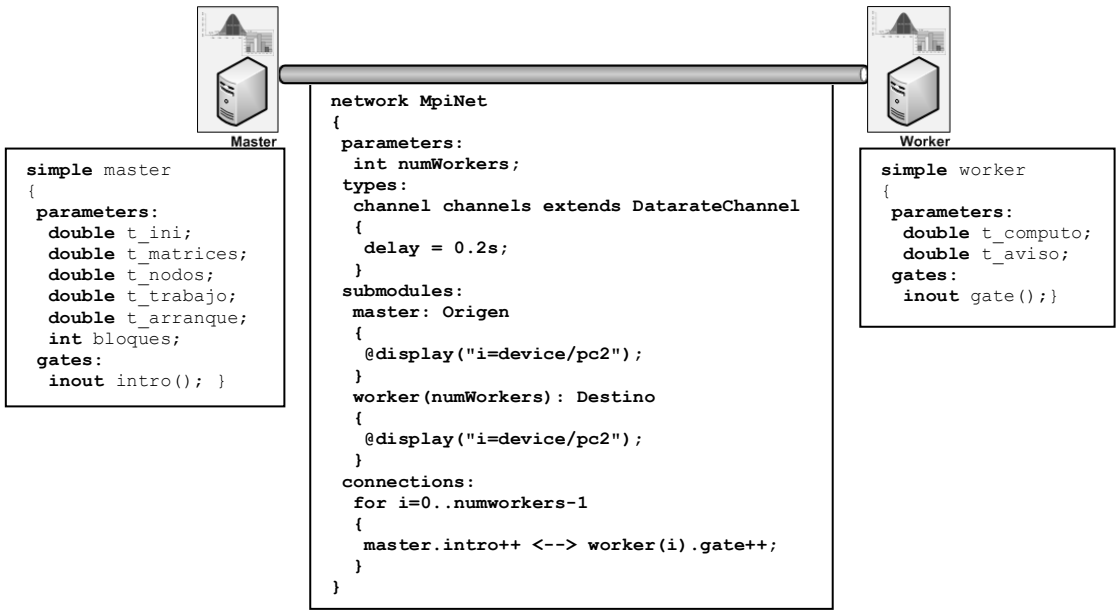


Figura 2. Modelo de simulación inicial de CluSim

Para determinar cuál de las distribuciones identificadas en la caracterización se acerca más al comportamiento de la aplicación real se ejecutaron simulaciones con ambas distribuciones para distintos tamaños de matriz.
Como se evidencia en la Fig. 3, la distribución normal representa con mayor precisión el comportamiento de la aplicación objetivo.

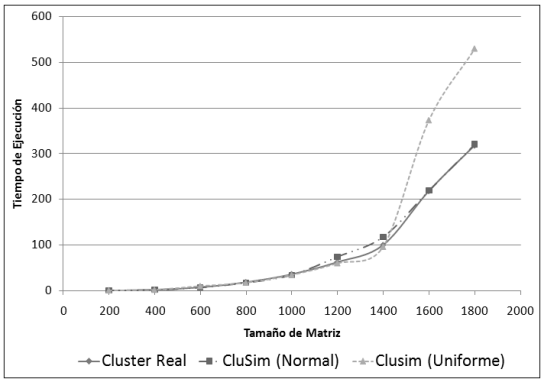


Figura 3. Comparativa de tiempos de ejecución en el cluster real y CluSim con distribución normal y uniforme.

Con estos resultados se ajustaron los parámetros del simulador y se ejecutaron las simulaciones correspondientes a cada uno de los escenarios planteados en el cluster real.
La Fig. 4 contrasta los tiempos de ejecución correspondientes al producto de matrices, con asignación dinámica, ejecutado en 2, 4 y 8 nodos del cluster, con los tiempos de ejecución generados por el simulador. Los tiempos de ejecución reales y simulados que se muestran en la figura son aproximadamente iguales, a pesar de las simplificaciones del modelo de simulación.

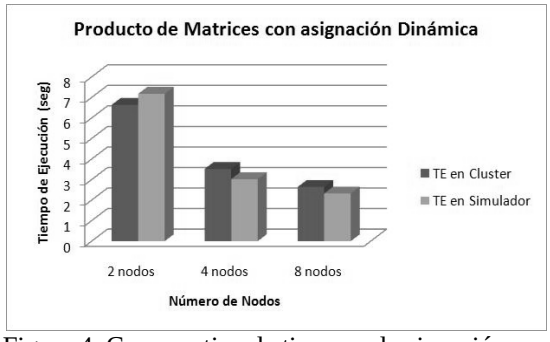


Figura 4. Comparativa de tiempos de ejecución.

A partir de estos resultados se trabajó en extender CluSim a la simulación de clusters heterogéneos tanto cableados como con conexión inalámbrica, entornos a los que demostró adaptarse con relativa facilidad.
Sin embargo, CluSim sólo se había probado y validado con patrones de configuración obtenidos de las ejecuciones reales del producto de matrices con asignación dinámica. Por ello, en trabajos

posteriores (García et al., 2012) se desarrolló un proceso de modelado de aplicaciones y se lo utilizó para generalizar CluSim hacia cualquier aplicación M/W, e incluso otro tipo de paradigmas paralelos. La definición del proceso de modelado y su aplicación para la extensión de CluSim se describen en los siguientes apartados.

4 MODELADO DE APLICACIONES

Considerando el carácter específico de la versión original de CluSim (sólo simula el producto de matrices bajo el paradigma M/W) y en vista de alcanzar un modelo genérico que permita simular aplicaciones de diferentes paradigmas se inició un proceso generalización que comprendió:

- la definición de las fases del proceso de modelado y
- la reformulación del modelo de aplicaciones M/W.

4.1 *Proceso de modelado*

Para formular un modelo de aplicación lo ideal es medir las características del sistema real. Esto puede llevarse a cabo aplicando técnicas de análisis de rendimiento que registren y determinen los valores correspondientes a métricas o parámetros del sistema. Los parámetros identificados se utilizan para definir el modelo correspondiente. En la Figura 5 se presenta el proceso de modelado seguido por el GIS (García et al., 2012).
Las técnicas de análisis de rendimiento aplicables en este proceso pueden clasificarse como se indica en la Figura 6 (Rodríguez Martínez, 2011).
Por un lado, en función del instante en el que se realizan las medidas, estas técnicas se dividen en métodos de muestreo y métodos accionados por eventos. Los primeros utilizan un reloj o contador externo que registra el instante en el que se generan los valores de las métricas y de los parámetros de rendimiento. En tanto que los segundos realizan las medidas únicamente cuando se activan determinados eventos. Entre éstos últimos, el mecanismo más común es la instrumentación de aplicaciones, que consiste en insertar instrucciones en el código a fin de registrar el comportamiento dinámico durante la ejecución. Según como se aplique, la instrumentación puede ser de código fuente, de traductores código a código, de interposición de librerías o de forma dinámica.
Por otro lado, en función del formato en el que se almacenan los datos recopilados, las técnicas de análisis pueden clasificarse en perfiles (profiles) y trazas (traces). El perfil de una ejecución consiste en un resumen estadístico del comportamiento, durante la propia ejecución, de las diferentes métricas y parámetros de rendimiento considerados. Las trazas registran la secuencia temporal de los valores de rendimiento en cada instante de medida, así es sencillo reproducir o visualizar el comportamiento de cada métrica durante la ejecución de la aplicación.

Figura 5. Proceso de modelado de aplicaciones.

En particular, el proceso de modelado seguido por el GIS se realizó aplicando el método accionado por eventos mediante la instrumentación de código por interposición de librerías; eligiéndose para el registro de mediciones el método de trazas.

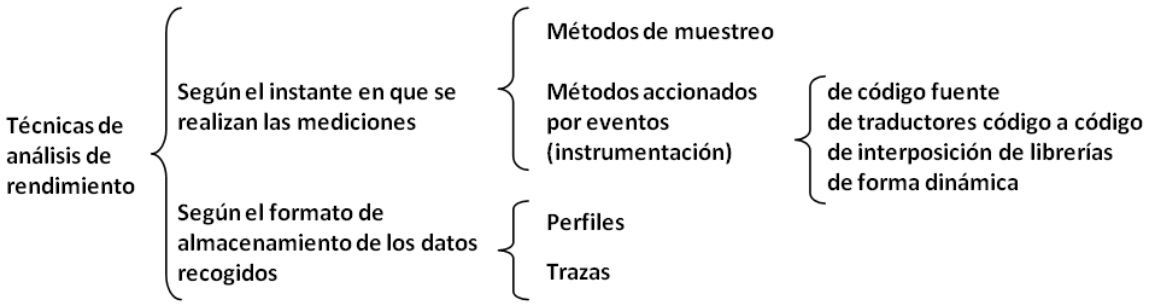


Figura 6. Clasificación de técnicas análisis de rendimiento.

4.2 *Reformulación del modelo*

En la reformulación del modelo se consideraron 2 aplicaciones M/W: producto de matrices con asignación dinámica y suma de arreglos. La instrumentación del código se realizó utilizando la librería MPE de MPICH y los patrones de cómputo y comunicación se registraron mediante trazas de ejecución. El análisis gráfico de las trazas se realizó con el visor Jumpshot de la librería MPE. La Fig. 7 presenta un segmento de traza en el que se identifican los tiempos considerados en el modelo reformulado.

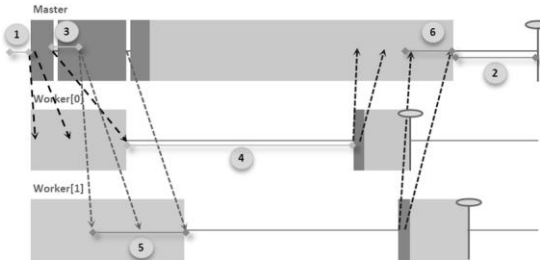


Figura 7 Segmento de una traza de la aplicación producto de matrices para el nuevo modelo.

En la tabla 2 se indican y describen los parámetros del modelo reformulado.

Tabla 2. Parámetros del modelo reformulado.

Parámetro	Tiempos Master/Worker
t_cómputo_inicial(1)	Preparación para enviar trabajo a los workers
t_computo_final (2)	Compilación de resultados
t_nodos (3)	tiempo entre el envío de tareas a los workers
t_computo_nodo (4)	Cómputo del worker
t_envío_trabajo (5)	Envío de trabajo al worker
t_envío_resultado (6)	Devolución de resultados al máster

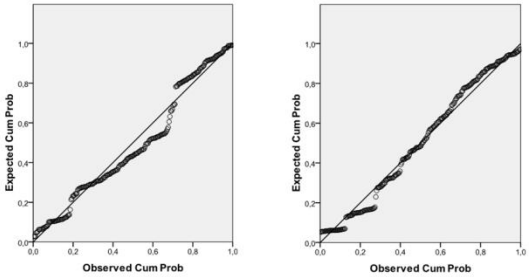
Para completar el modelo se identifican las distribuciones de probabilidad que corresponden a cada uno de los parámetros identificados utilizando el software SPSS 14 para el análisis estadístico de las trazas. La Tabla 3 presenta los parámetros del nuevo modelo y las distribuciones de probabilidad asociadas.

Tabla 3. Distribuciones del nuevo modelo.

Parámetro	D^N de probabilidad
t_computo_inicial	Uniforme
t_computo_final	Uniforme
t_nodos	Lognormal
t_computo_nodo	Lognormal
t_envio_trabajo	Weibull
t_envio_resultado	Lognormal

Finalmente, la validación del modelo se llevó a cabo contrastando los parámetros y distribuciones de probabilidad correspondientes con el análisis de rendimiento (aplicando las técnicas descriptas) de 2 aplicaciones M/W: búsqueda secuencial en arreglos y ordenación de arreglos por intercambio.

El análisis de las trazas de ejecución de las aplicaciones probadas permitió: identificar los parámetros del modelo en los resultados obtenidos y establecer el ajuste de los tiempos de las aplicaciones de búsqueda y ordenación a las distribuciones propuestas en el modelo (Fig. 8.a y 8.b) en base a su análisis estadístico.



(a) Distribución de t_computo_nodo (Lognormal) (b) distribución de t_envio_trabajo (Weibull)

Figura 8. Distribución de tiempos para el modelo

En virtud de los resultados positivos obtenidos para el modelado de aplicaciones M/W (Lasserre et al., 2012), el siguiente paso en el proceso de generalización del modelo de CluSim es el estudio y modelado de aplicaciones SPMD, pipeline y D/C. Esto se presenta en el siguiente apartado.

5 EXTENSIÓN DE CLUSIM A SPMD, PIPELINE Y D/C

Continuando con el desarrollo del modelo genérico, se aplicó el proceso descripto en el apartado 4.1 a aplicaciones tipo SPMD, pipeline y D/C. En la tabla 4 se presentan las aplicaciones utilizadas.

Los experimentos necesarios para el estudio se llevaron a cabo en el cluster del Laboratorio de Altas Prestaciones de la FI de la UNJu. Este cluster cuenta con 14 equipos con las siguientes características:

- procesador core I3
- 4 GB de RAM
- 500 GB de disco duro
- Placa Ethernet 10/100M
- Sistema operativo: Linux Ubuntu 12.04
- Librería de paso de mensajes: MPICH2
- Librerías adicionales: librería MPE y visualizador Jumpshot

A modo ilustrativo se muestran sólo dos trazas de ejecución por paradigma: Figuras 9 y 10 (SPMD), Figuras 11 y 12 (pipeline) y Figuras 13 y 14 (D/C).

Tabla 4. Aplicaciones objetivo y sus parámetros

	Aplicación	Parámetros Requeridos	Aplicación en C
SPMD	Estimación de Pi por método Montecarlo		`Montecarlo`
	Resolución de Sistemas de ecuaciones por Gauss-Seidel		`Gaussseidel`
Pipeline	Eliminación Gauss-Jordan		`ge_pipe`
	Factorización de Cholesky	Tamaño del vector: 1000 Tamaño del bloque: 100	`Cholesky`
	Problema de n cuerpos	n: 1000	`n-body`
Divide and Conquer	N° Primos		`Primes`
	Merge-Sort	1024 datos	`Mergesort`
	Alineamiento Múltiple de Secuencias		`Clustalw`
	Fractal de Mandelbrot	x real: -2, 2 x imag.: -2, 2	`Fractal`
	Cálculo de Pi por el método de la integral	32 intervalos	`pi_calc`

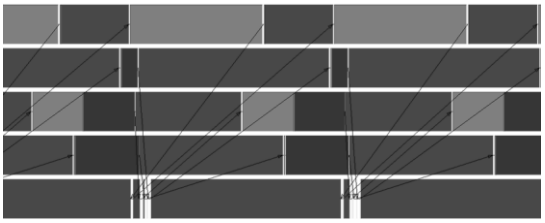


Figura 9: Segmento de traza de montecarlo_mpi.c

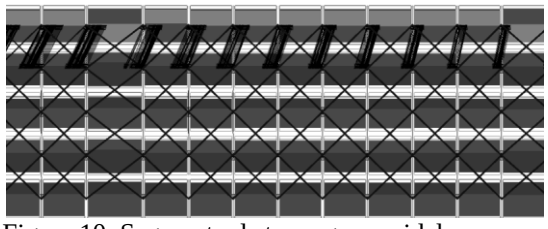


Figura 10: Segmento de traza gaus-seidel.c

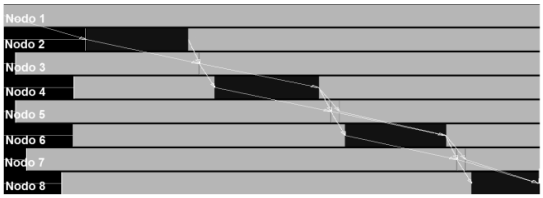


Figura 11. Segmento de una traza de ge_pipe.c.

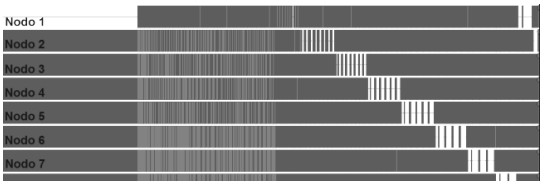


Figura 12. Segmento de una traza de cholesky.c.

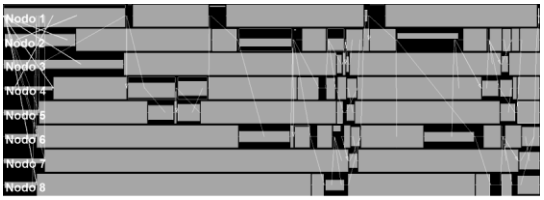


Figura 13. Segmento de una traza de clustalw.c.

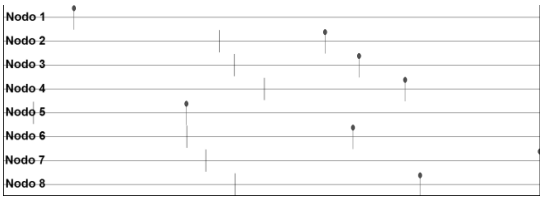


Figura 14. Segmento de una traza de primes.c.

Del análisis de las trazas de ejecución se puede observar:

- aplicaciones SPMD: en las trazas de ejecución es posible identificar seis parámetros de la aplicación que tienen correspondencia con los parámetros definidos para el modelo genérico de CluSim. Esta correspondencia se muestra en la Tabla 5 (García et al., 2012).
- aplicaciones pipeline: como en el caso anterior, en las trazas de ejecución pueden identificarse seis parámetros que se corresponden con los definidos para el modelo genérico CluSim. Esta correspondencia se muestra en la Tabla 5.
- aplicaciones D/C: las aplicaciones que siguen este paradigma no presentan patrones similares de comportamiento (como ejemplo obsérvese las figuras 13 y 14). Por lo tanto, siguiendo el modelo planteado, resulta difícil la identificación de parámetros que caractericen a las aplicaciones que siguen este paradigma.

Respecto a las distribuciones de probabilidad de los parámetros para cada tipo de aplicación, actualmente se trabaja en el procesamiento de los datos numéricos de estas trazas.

6 UN MODELO DE TF

6.1 *Necesidad de la TF*

En la introducción se destacaba el crecimiento de los centros de supercomputación para satisfacer las demandas de poder de cómputo de las actuales

aplicaciones de altas prestaciones. Sin embargo, la confiabilidad de los sistemas disminuye a medida que el sistema crece, debido a que se reduce el tiempo medio entre fallos. Esta situación provoca que muchas aplicaciones científicas complejas corran el riesgo de no completar su ejecución perdiéndose el trabajo realizado. Por ello, la administración de los fallos en estos sistemas de cómputo constituye uno de los desafíos más grandes en esta área; habiéndose desarrollado muchas técnicas para agregar confiabilidad y alta disponibilidad a los sistemas distribuidos (Hursey, 2010).

Tabla 5. Correspondencia entre parámetros y tiempos de aplicación Master/Worker, SPMD y Pipeline.

Parámetro	Tiempos Master/Worker	Tiempos SPMD	Tiempos Pipeline
t_cómputo_inicial	Preparación para enviar trabajo a los workers	Preparación para difusión inicial	Preparación para distribución inicial
t_computo_final	Compilación de resultados	Reducción final	Reducción de resultados
t_nodos	tiempo entre el envío de tareas a los workers	Igual a 0	Igual a 0
t_computo_nodo	Cómputo del worker	Cómputo del nodo	Cómputo del nodo
t_envío_trabajo	Envío de trabajo al worker	Difusión	Envío de trabajo / difusión
t_envío_resultado	Devolución de resultados al máster	Reducción / Difusión de resultados parciales	Envío de resultados / diffusion

Las técnicas de TF incluyen transacciones, comunicaciones de grupo y rollback recovery en sus diferentes variantes y enfoques (Elnozahy, 2002).
Así, el objetivo de la TF es permitir que las aplicaciones completen correctamente su ejecución, a pesar de la falla de alguno de los componentes del sistema de cómputo, reduciendo al mínimo la pérdida del trabajo y el overhead inherente a la introducción de los mecanismos de protección (Rodrigues de Souza, 2006).
Por ello, una de las líneas de investigación del GIS pretende extender CluSim para que simule sistemas HPC con distintas configuraciones de TF.
En este apartado se presenta la propuesta inicial del modelo de TF desarrollada por el GIS.

6.2 *Modelo de simulación propuesto*

En esta propuesta se tomaron las cuatro fases de TF descriptas por Jalote (1994):

- detección del fallo,
- confinamiento del daño,
- recuperación del error,
- tratamiento del fallo y reconfiguración del sistema.

Continuando con la filosofía de CluSim, las fases de TF y los mecanismos de protección asociados a cada una de ellas se implementarán de forma configurable y parametrizable, de este modo, el simulador también permitirá evaluar el impacto de diferentes configuraciones de TF para un cluster de computadoras.
La fase de detección del error se implementará mediante el chequeo de tiempo, una de las técnicas más comunes en la detección de fallas en nodos (Jalote, 1994). El chequeo de tiempo podrá configurarse como centralizado con un componente dedicado o descentralizado utilizando un proceso de chequeo implementado en cada nodo (Fig. 15). En ambos casos, cada nodo implementará el mecanismo de respuesta (heartbeat).

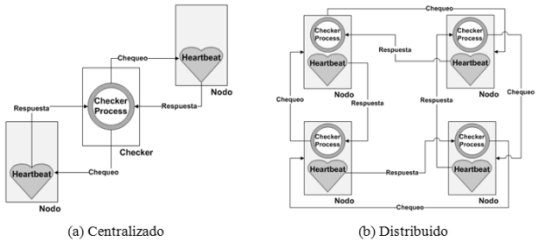


Figura 15. Módulos asociados al mecanismo de detección de fallos.

En la fase de confinamiento del daño, la determinación de los límites de la propagación del error se implementará mediante los mecanismos de log pesimista de mensajes y checkpoint. Según indican Elnozahy y colegas (2002), el log pesimista de mensajes garantiza la recuperación del estado observable de cada proceso. En cuanto al checkpointing, éste podrá configurarse como centralizado o distribuido y también se podrá optar coordinado (inhabilitándose el log de mensajes) o no coordinado (Fig. 16).
La recuperación de error se realizará en base a la configuración de los mecanismos de la fase anterior. Si el checkpoint se realizó de manera coordinada se aplicará rollback-recovery basado en checkpoint y en el caso no coordinado se aplicará el rollback-recovery basado en log de mensajes.
Para el tratamiento del fallo y reconfiguración del sistema se mantendrá un registro de los recursos

que fallan y la reconfiguración del sistema se implementará mediante la sobrecarga de alguno de los recursos disponibles o a través de la utilización de nodos redundantes, permitiendo la selección de una de las alternativas y la cantidad de nodos spare.

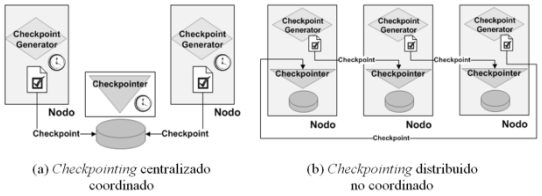


Fig. 16. Módulos asociados a los mecanismos de confinación del daño. Se muestran sólo dos de las posibles combinaciones, con los mismos módulos podrá configurarse un checkpointing centralizado no coordinado o distribuido coordinado.

7 CONCLUSIONES

En este artículo se presentó la evolución de CluSim, el simulador de cluster desarrollado por el GIS, desde su prototipo inicial hasta el proceso de generalización de su modelo de simulación y una propuesta de extensión para TF. El proceso de generalización permitió formular un modelo genérico para aplicaciones M/W, SPMD y pipeline. No obstante, en el caso de las aplicaciones D/C los resultados obtenidos no presentan un patrón común de comportamiento que permita asimilar este tipo de aplicación a los parámetros definidos para los otros paradigmas. Respecto a la extensión para TF, se propone simular las 4 fases de TF implementando los mecanismos de protección más utilizados de forma parametrizable y configurable, de modo que el usuario pueda evaluar el impacto del overhead en su aplicación y así optar por la configuración más conveniente para un sistema determinado.

Los siguientes trabajos futuros se desprenden de lo realizado: a) determinación de las distribuciones de probabilidad de los parámetros SPMD y pipeline, b) implementación de la propuesta de TF y c) validación de CluSim extendido.

8 REFERENCIAS

Denzel, W.E.; Li, J.; Walker, P. and Y. Jin. A framework for end-to-end simulation of high-performance computing systems. *In Simutools '08*. 1-10, ICST, Brussels. 2008

Elnozahy, E. N. (Mootaz), Alvisi, L., Yi-Min Wang, and D.B. Johnson. A survey of rollback-recovery protocols in message-passing systems. *ACM Comput. Surv. 34, 3*, 375-408. 2002

García, A.; Lasserre, C.M.; Verazay, A.R.N.; Pérez Otero, N.M.; Pérez Ibarra, M.; Martínez, J.G. y S.A. Nolasco. Modelado de Aplicaciones SPMD. *VIII Jornadas de Ciencia y Tecnología de Facultades de Ingeniería del NOA*. Ed. Grupo Loza Impresiones S.R.L. ISSN 1853-7871 San Miguel de Tucumán. 2012

Hursey, J. *Coordinated Checkpoint/Restart Process Fault Tolerance for MPI Applications on HPC Systems*. Ph.D. Dissertation. Indiana University, Indianapolis, IN, USA. Advisor(s) Andrew Lumsdaine. AAI3423687. 2010

Jalote, P. *Fault Tolerance in Distributed Systems*. Prentice Hall, Universidad de Michigan. ISBN: 0133013677. 1994

Lasserre, C.M.; García, A.; Pérez Otero, N.M.; Verazay, A.R.N.; Pérez Ibarra, M.; Nolasco S.A. y J.G. Martínez. Hacia un Modelo Genérico de Aplicaciones Paralelas. XVIII *Congreso Argentino de Ciencias de la Computación* 296-305. ISBN 978-987-1648-34-4. UNS. Bahía Blanca. 2012

Núñez, A.; Fernández, J.; García, J.D.; García, F. and J. Carretero. New techniques for simulating high performance MPI applications on large storage networks. *J. Supercomput. 51, 1*. 40-57. 2010

Rodrigues de Souza, J. *FTDR: Tolerancia a fallos, en clusters de computadoras geográficamente distribuidos, basada en Replicación de Datos*. Tesis doctoral de la Universidad Autónoma de Barcelona. 2006

Rodríguez Martínez, D. *Modelado Analítico del Rendimiento de Aplicaciones en Sistemas Paralelos.* Tesis Doctoral. Departamento de Electrónica e Computación. Universidade de Santiago de Compostela. 2011

Valdivezo, L.M.; Pérez Otero, N.M.; Pérez Ibarra, C.M. y C.M. Lasserre. Caracterización de una aplicación paralela con distintas configuraciones en CluSim. *Investigaciones en Facultades de Ingeniería del NOA*. ISSN 3367-5072. Ed. EdiUNJu. S.S. de Jujuy. 499-504. 2010