Soporte para el análisis de workloads en el proyecto eNANOS

Ivan Rodero, Julita Corbalán, Alejandro Duran, Jesús Labarta
Dept. Arquitectura de Computadors
Universitat Politècnica de Catalunya
Campus Nord. Jordi Girona 1-3
08034 Barcelona
{irodero, juli, aduran, jesus}@ac.upc.edu

Resumen

El proyecto eNANOS plantea la planificación coordinada de trabajos entre varios niveles, desde el entorno heterogéneo y dinámico de un Grid hasta la ejecución de procesos y threads en las CPU's de un computador o un cluster. Para estudiar las políticas de planificación se necesita algún mecanismo de análisis de workloads. En este artículo presentamos un mecanismo de monitorización de trabajos integrado en el planificador eNANOS Scheduler. También presentamos una herramienta que a partir de la información obtenida en la monitorización es capaz de generar trazas que pueden ser visualizadas y analizadas fácilmente con Paraver. Para mostrar las ventajas del sistema planteado se presenta la evaluación de un workload y se compara con posibles alternativas.

1. Introducción

La planificación de trabajos en Grid es todavía una cuestión que hay que resolver y en la cual se están dedicando muchos esfuerzos. Normalmente un Grid está formado por diversos recursos que pertenecen a diferentes propietarios. Estos recursos no tienen porqué estar dedicados exclusivamente para ser utilizados por el Grid. También hay que tener en cuenta que los recursos Grid por definición son heterogéneos y tienen un comportamiento muy dinámico. Por lo tanto, lo más habitual es que los recursos de altas prestaciones (HPC) de un Grid dispongan de sistemas de planificación locales. Estos sistemas de encargan de gestionar la planificación de los trabajos a nivel local de un cluster mediante sistemas de colas.

Esto nos sugiere que necesitaremos un modelo de planificación global diferente al que se utiliza habitualmente en un entorno local o centralizado. Será necesario añadir un nuevo nivel de planificación que se coordine con los diversos gestores locales que se encuentran en los recursos computacionales. Por lo tanto las decisiones de planificación a nivel Grid deben ser tomadas siendo conscientes de la información obtenida en los niveles inferiores. Así, se pretende realizar una planificación eficaz haciendo un uso óptimo de los recursos del Grid. También hay que tener en cuenta que un planificador o broker de Grid no tiene control directo sobre los recursos computacionales, por lo tanto necesita un planificador local que los gestione. Aunque se está trabajando en definir una arquitectura en la planificación en Grid [21] todavía no se ha decidido y no hay disponible ningún estándar. De todos modos parece razonable pensar que estas capas de planificación necesitan interactuar entre ellas. Tampoco está claro el tipo de jerarquía que debe formarse entre los distintos niveles de planificación.

En nuestra propuesta planteamos la planificador coordinada de tres niveles, desde el Grid hasta la gestión de procesadores. De momento hemos implementado la coordinación entre los dos niveles inferiores de la arquitectura.

Para analizar el comportamiento del sistema aplicando la coordinación entre eNANOS Scheduler y NANOS-RM se necesita algún mecanismo para analizar los workloads. En este artículo presentamos una herramienta que nos permite analizar y visualizar los workloads de forma sencilla. A partir de la información de monitorización en formato XML proporcionada por el eNANOS Scheduler se genera una traza que puede ser analizada con la herramienta de análisis y visualización Paraver [2]. Este mecanismo nos permitirá incluir la información de todos los niveles de planificación en una única traza. La información obtenida mediante estas trazas nos permitirá analizar y mejorar las distintas políticas de planificación, así como comprender el comportamiento de nuestro entorno de ejecución.

El resto del artículo está organizado como sigue a continuación. En el apartado 2 se presenta la propuesta del proyecto eNANOS. En el apartado 3 se hace un resumen de los principales trabajos relacionados. Los apartados 4 y 5 presentan las principales características del eNANOS Scheduler y del NANOS-RM respectivamente. En el apartado 6 se expone el mecanismo de monitorización y la herramienta para generar las trazas. Mediante ejemplos reales, en el apartado 7 se evalúan varios mecanismos para analizar workloads y se muestra la utilidad de la herramienta desarrollada. Finalmente en el apartado 8 se presentan las conclusiones y las posibles líneas de trabajo futuro.

2. Visión general del proyecto eNANOS

Nuestro proyecto se centra en la planificación coordinada y de forma integral de todos los componentes involucrados en la ejecución de aplicaciones HPC en Grid, desde nivel Grid hasta la planificación de procesadores. No sólo queremos coordinar el planificador Grid con los planificadores de los distintos recursos, sino que queremos realizar la planificación consciente de todos los niveles involucrados. Consideramos tres niveles de planificación básicos, desde el entorno heterogéneo y dinámico de un Grid hasta la ejecución eficaz de procesos y threads en una o varias CPU de un computador o un cluster. En la Figura 1 se muestra un esquema de los principales componentes involucrados. También se puede apreciar el flujo de información, tanto la existente (en negrita) como la que tenemos prevista añadir más adelante (más claro). Esta información es parte de la que se necesita para realizar la planificación de forma coordinada

Este trabajo se enmarca dentro del proyecto eNANOS [3] cuyo objetivo principal es la gestión autónoma de recursos, una de las tareas necesarias del autonomic computing [11]. Teniendo en cuenta que la tecnología Grid es muy extensa, este proyecto se centra en la ejecución de aplicaciones paralelas de altas prestaciones en Grids compuestos de computadores paralelos, con arquitecturas de memoria compartida (SMP i CC-NUMA) con un número medio-alto de procesadores por nodo (16-128 CPU's).

En aplicaciones HPC para Grid se acostumbra a utilizar modelos de programación paralela como es el caso de MPI o PVM. Estos modelos tienen ciertas carencias de rendimiento en las arquitecturas basadas en SMP. Nosotros planteamos la utilización del modelo híbrido MPI+OpenMP.

También hay que tener en cuenta que para realizar una planificación eficiente es muy importante la monitorización tanto de los recursos como de las aplicaciones. Normalmente los sistemas de monitorización se encargan de proporcionar información sobre los recursos e información muy general sobre la carga de las CPU's de los nodos. Nosotros queremos proporcionar información más concreta sobre el comportamiento de las aplicaciones, por ejemplo el speedup que está obteniendo. Este tipo de información dinámica nos podrá servir para tomar decisiones de planificación, como puede ser la ejecución de nuevas aplicaciones o la migración de ciertos procesos o threads.

El primer paso realizado fue desarrollar un resource broker, el eNANOS broker [18]. Este broker está desarrollado sobre Globus Toolkit 3 como Grid Service y es compatible tanto con GT2 como con GT3. Para gestionar la planificación a nivel local hemos implementado un planificador, eNANOS Scheduler, que interactúa con el sistema de colas LoadLeveler mediante su API. La planificación a nivel de CPU's se realiza a través del NANOS-RM, sus principales características se exponen en el apartado 5. Hemos adaptado el NANOS-RM para proporcionar al eNANOS Scheduler información sobre el comportamiento de las aplicaciones de forma dinámica.

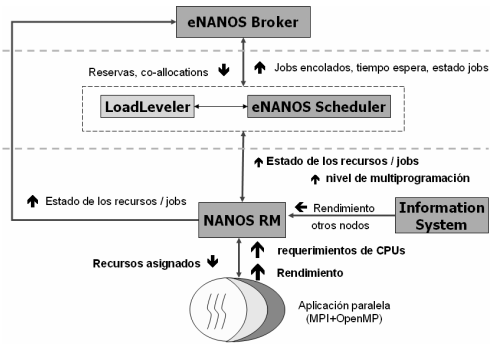

Figura 1. Coordinación entre los componentes del proyecto eNANOS

3. Trabajo relacionado

Existen diversas iniciativas que atacan de forma individual la problemática de planificación en los tres niveles que proponemos en nuestro proyecto. Por un lado podemos encontrar diversos proyectos de meta-schedulers o resource brokers para Grid como son AppLes [1], Condor-G [6], EZ-Grid[5], GridBus [14], GridLab Resource Management System (GRMS) [7], o GridWay [8]. Estos implementan diversas políticas como pueden ser económicas, basadas en la información obtenida de la monitorización de los recursos o mediante mecanismos de predicción basados en información histórica o heurísticas.

Por otro lado hay varios planificadores para recursos de altas prestaciones (clusters) y que trabajan de forma externa a los sistemas de colas, por ejemplo MAUI scheduler [12], EASY [19] o OAR Scheduler [17]. Estos planificadores implementan diversas políticas y soportan mecanismos como por ejemplo las advanced reservations. También se pueden encontrar diversos sistemas de monitorización que dan soporte para Grid como pueden ser NWS [16], Mercury [13] o Globus MDS [9].

A pesar de esto no conocemos ningún proyecto que implemente la coordinación de todos estos niveles tal y como proponemos en el proyecto eNANOS. Del mismo modo tampoco conocemos ninguna herramienta que nos permita analizar los workloads en los tres niveles de forma integral.

4. eNANOS Local Scheduler

El eNANOS Scheduler es el responsable de ejecutar y controlar los trabajos que hay encolados en el sistema de colas mediante la API que proporciona LoadLeveler. Este planificador se comunica con el NANOS-RM para obtener información relacionada con el comportamiento de las aplicaciones y de los recursos locales, y con el eNANOS Broker para proporcionar información al nivel superior. Adicionalmente, está previsto que el planificador se comunique con un servicio de predicción y un sistema de información que están en desarrollo. También proporciona información de log y monitoriza los trabajos proporcionando una descripción detalla de los workloads en formato XML.

El modulo principal del planificador se encarga de controlar todos los trabajos que encuentra en el sistema de colas de LoadLeveler. Como es habitual, el funcionamiento es cíclico, periódicamente el planificador realiza las siguientes funciones básicas:

- Consulta los trabajos de sistema de colas
- Actualiza la información del sistema
- Evalúa la política de planificación
- Ejecuta los trabajos seleccionados

La configuración básica del scheduler se realiza a través de un fichero de configuración. Se pueden definir parámetros como puertos de comunicación, nombres de ficheros, o el nivel de multiprogramación inicial.

La comunicación con los otros componentes de la arquitectura del sistema se realiza mediante dos threads. El primero se encarga de interactuar con el NANOS-RM. Éste recibe información del runtime sobre el nivel de multiprogramación admitido por los nodos u otra información relacionada con las aplicaciones y el hardware, por ejemplo la carga de los nodos, o las CPU's que utiliza cada thread OpenMP. El segundo notifica los cambios relacionados con la ejecución de los trabajos o con los recursos que puedan ser relevantes para el nivel Grid. También puede recibir instrucciones del broker pero como actualmente esta parte está en desarrollo, de momento la comunicación entre los dos componentes se realiza indirectamente mediante el jobmanager de Globus y variables de entorno.

Actualmente el scheduler implementa las siguientes políticas de planificación: FIFO, Backfilling [20] y CPUM-based. La política basada en Backfilling se implementa a partir del tiempo de ejecución límite especificado por el usuario. La política CPUM-based utiliza la información del NANOS-RM para determinar cuando un nodo acepta la ejecución de nuevas aplicaciones. La información del runtime y el nivel de multiprogramación nos permiten planificar la siguiente aplicación de la cola siguiendo una política del tipo FIFO (según el instante de envío del trabajo). Estamos trabajando en la implementación de una política de Backfilling basada en la predicción realizada por un sistema de predicción externo.

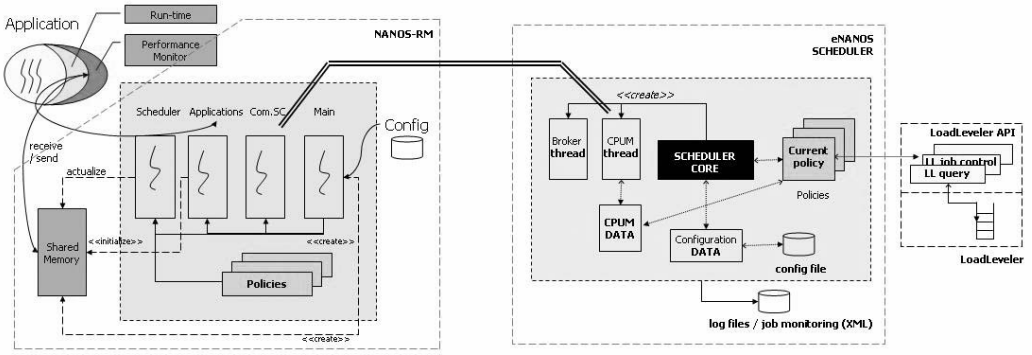

Figura 2. Arquitectura general y coordinación entre NANOS-RM y eNANOS Scheduler

Para poder realizar una planificación coordinada es muy importante la monitorización de los trabajos y de los recursos. En el proyecto eNANOS la idea es monitorizar los trabajos en todos los niveles de planificación pero de momento hemos implementado la monitorización de los workloads en los dos niveles inferiores. El eNANOS Scheduler proporciona información de la monitorización de las aplicaciones que puede ser utilizada por las políticas de planificación. Esta información también puede ser muy útil para el investigador ya que permite ver el comportamiento del workload. De este modo se puede disponer de información precisa para poder ajustar las políticas de planificación. La herramienta que se presenta en el apartado 6 es la encargada de proporcionar esta información de forma inteligible.

5. NANOS Resource Manager

El NANOS-RM es un gestor de procesadores configurable cuya misión principal es recibir las peticiones de las aplicaciones y distribuir los procesadores entre ellos. Debido a las características de los procesos/threads de AIX son las propias aplicaciones las que fuerzan la planificación decidida por el NANOS-RM.

El NANOS-RM también interactúa con el eNANOS Scheduler, le indica el número máximo de trabajos (o nivel de multiprogramación) que el NANOS-RM admite para que el sistema funcione sin tener sobrecarga. La idea es compartir la información necesaria para mejorar el rendimiento del sistema global sin penalizar el rendimiento de las aplicaciones individualmente.

Las aplicaciones se instrumentan dinámicamente para detectar comportamientos ineficientes como por ejemplo desbalanceos de la carga, speedups bajos, etc. El Performance Monitor es el encargado de esta tarea y está implementado como una librería que se carga dinámicamente. Los objetos que gestiona el NANOS-RM son: trabajos, tareas y CPU's. En el caso genérico, un trabajo es una aplicación MPI+OpenMP. Cada uno de los procesos MPI se considera una tarea e internamente puede pedir procesos.

En la Figura 2 se muestra la estructura general del NANOS-RM y la interacción con el eNANOS Scheduler. El funcionamiento del NANOS-RM es cíclico (asíncrono) y realiza las siguientes tareas básicas:

- Planificación de procesadores
- Control de la carga del sistema
- Detección dinámica de aplicaciones multinivel

Dispone de tres threads principales: uno se encarga de la interacción con el eNANOS Scheduler, otro se coordina con las aplicaciones y el último se encarga de realizar la planificación de los procesadores. La inicialización y configuración se realiza a través de la línea de comando.

Las políticas de planificación son dinámicas y tienen dos fases (multinivel). La primera fase es entre aplicaciones, implementa una política FIFO: 1 aplicación tiene N procesos MPI y por lo tanto

requiere de un mínimo de N procesadores. La segunda fase es entre procesos de una misma aplicación MPI+OpenMP e implementa dos posibles políticas: Equipartición y Dynamic Processor Balancing [4]. Esta última intenta balancear la carga entre los procesos MPI de una misma aplicación.

Para el análisis de rendimiento se detecta automáticamente la estructura iterativa de las aplicaciones para poder comparar. Internamente se mide el tiempo dedicado a cálculo y a ejecutar MPI. Se realizan medidas para evitar posibles picos de desbalanceo y se descartan iteraciones para evitar outliers. Más información sobre el NANOS-RM y las técnicas de balanceo de la carga puede encontrase en [4].

6. Soporte para análisis de workloads

El eNANOS Scheduler internamente se encarga de monitorizar los trabajos y los recursos. El planificador proporciona un fichero en formato XML para cada trabajo que gestiona. En estos ficheros XML describen la ejecución del trabajo y siguen un esquema que incluye la siguiente información (atributos más importantes):

- Identificador del trabajo a nivel local (StepID asignado por LoadLeveler)
- Identificador del trabajo a nivel Grid (JobID asignado por el eNANOS Broker)
- Nombre del workload
- Nombre del trabajo
- Topología de la aplicación (número de tareas MPI y procesos OpenMP)
- Conjunto de eventos

Cada uno de los eventos de un trabajo incluye el instante en el que se produce el evento (timestamp), la identificación del evento, una descripción y la información necesaria asociada al evento. El conjunto de los eventos nos permite determinar el ciclo de vida de los trabajos con una precisión de segundos (escala de los timestamps). Un conjunto de estos ficheros XML describen un workload ya que uno de los atributos permite identificarlo. Actualmente los workloads incluyen la información correspondiente a los dos niveles de planificación inferiores de la jerarquía planteada. La jerarquía básica del esquema XML planteado se muestra en la Figura 3, desde el nivel Grid hasta el nivel thread.

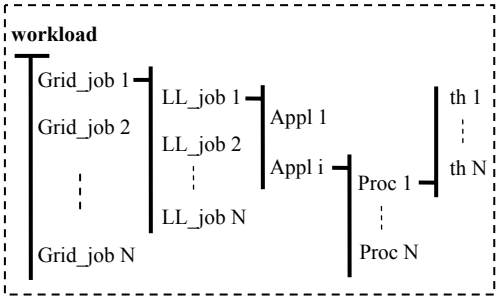

Figura 3. Jerarquía básica del esquema XML

Para poder visualizar y analizar los workloads hemos implementado una herramienta que a partir de estos ficheros XML genera una traza que puede ser visualizada con el software de visualización y análisis Paraver. La herramienta lo primero que hace es fusionar los ficheros XML correspondientes a varios trabajos en un solo fichero que contiene toda la información del workload. A partir de la información de las tareas y del conjunto de eventos generamos la traza.

Para las trazas utilizamos dos tipos de records (líneas de las trazas de Paraver): records de estado y records de evento. Los records de eventos corresponden directamente a los eventos que se describen en el fichero XML. Los records de estado definen el estado de un trabajo en un intervalo de tiempo determinado. Estos records se determinan a partir de cada par de eventos y del tipo de estos.

7. Evaluación

7.1. Arquitectura

La evaluación se ha realizado en un sistema IBM con dos configuraciones: un IBM RS-6000 SP con 8 nodos de 16 Nighthawk Power3 @375Mhz (192 Gflops/s) con 64 Gb de RAM y un IBM p630 con 9 nodos de 4 p630 Power4 @1Ghz (144 Gflops/s) con 18 Gb de RAM. Hay disponible un total de 446 Gflops y 1,8 TB de disco duro. Todos los nodos están conectados mediante un SP Switch2

funcionando a 500 MB/seg. El sistema operativo es AIX 5.1 con el sistema de colas LoadLeveler.

7.2. Análisis numérico del workload

En este apartado presentamos el mecanismo de análisis de workloads sin disponer de herramientas de soporte. Para ello utilizaremos dos workloads ejecutamos en un nodo de 16 CPU's. Estos workloads están compuestos por aplicaciones NAS [15] del tipo MZ y de la clase A. Posteriormente se presentan las ventajas de utilizar las herramientas de soporte para analizar los workloads.

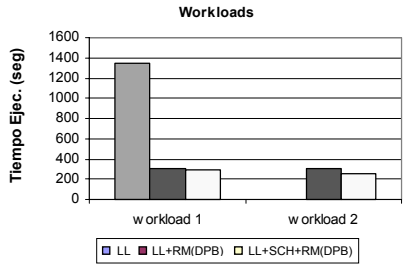

Figura 4. Tiempos de ejecución de los workloads

Los workloads utilizados están formados por 6 aplicaciones NAS, algunas de ellas BT y otras SP de la clase A. El número de tareas MPI y threads OpenMP no es el mismo para todas las aplicaciones. En la Figura 4 se muestra el tiempo de ejecución de los dos workloads con tres tipos de configuraciones: LL, LL+RM(DPB) y LL+SCH+RM(DPB). La primera corresponde a la configuración por defecto del sistema en la cual se aplica la política de backfilling a los trabajos encolados. La segunda utiliza también política de backfilling para planificar los trabajos pero las aplicaciones son gestionadas por el NANOS-RM aplicando la política de Dynamic Processor Balancing (DPB) para balancear la carga. En el último caso además de balancear las aplicaciones con el NANOS-RM también se utiliza el eNANOS Scheduler (en coordinación con el NANOS-RM) para realizar la planificación.

En este caso la información que disponemos es el tiempo de ejecución de los workloads (Figura 4) y de las aplicaciones que componen los workloads individualmente (Figura 5). Con esta información analítica podemos apreciar ciertas características de los workloads. Por ejemplo, en el primer workload con la configuración de LL se obtiene un tiempo de ejecución muy superior al obtenido con las otras configuraciones (del orden de seis veces más elevado). No podemos determinar los motivos por los cuales el tiempo de ejecución es tan elevado con esta configuración, sólo podemos hacer ciertas hipótesis. En las otras dos configuraciones, el tiempo de ejecución es inferior y bastante similar entre ellas. Realizando balanceo de la carga conseguimos una mejora muy importante, pero de forma coordinada con el planificador local todavía se consiguen mejores resultados. De estos datos deducimos que el balanceo de la carga ofrece mejoras y que el planificador en coordinación todavía más, pero no sabemos nada del comportamiento de los workloads.

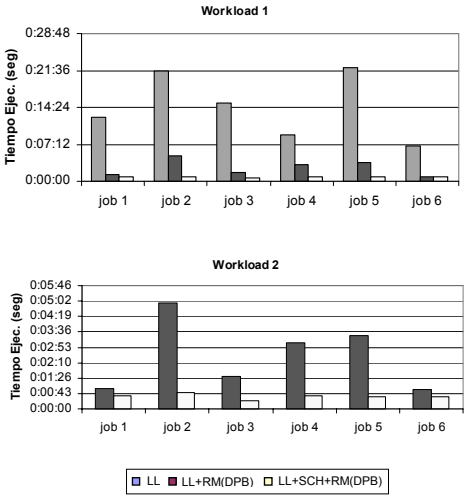

Figura 5. Tiempos de ejecución de las aplicaciones

En la Figura 5 podemos apreciar los tiempos de ejecución de las aplicaciones que componen los workloads de forma individual. Vemos como los tiempos de respuesta de las aplicaciones son en general mucho más reducidos utilizando los planificadores coordinados. Así pues, en este caso el planificador local también reduce el tiempo de respuesta de las aplicaciones y permite aumentar el throughput.

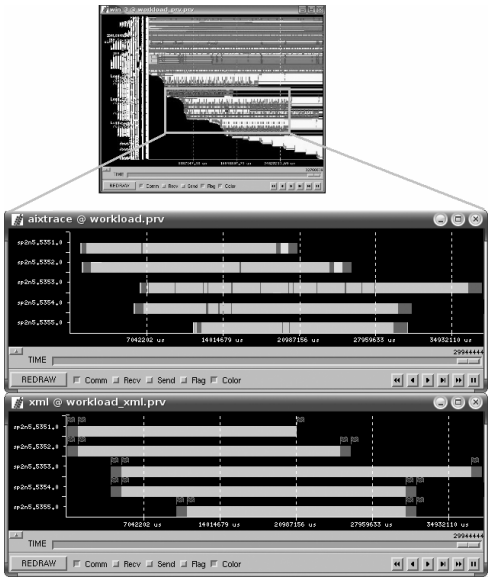

Figura 6. Trazas de un workload, mediante AIX trace y la herramienta presentada (XML)

7.3. Análisis y visualización con herramientas

Con el estudio anterior del workload hemos podido apreciar cierta información mediante datos cuantitativos. Con el uso de trazas podemos analizar el comportamiento de las aplicaciones de forma sencilla, rápida y eficaz. En la Figura 6 se muestra el aspecto de las trazas, cada fila corresponde a una aplicación en ejecución, el color más claro corresponde al estado "running" y el más oscuro al estado "idle". La traza de más arriba es directamente la obtenida mediante AIX trace [10] y convertida al formato de Paraver. Más abajo se muestran sólo los trabajos del workload de la misma traza. La última es la obtenida a partir de los ficheros XML de monitorización mediante la herramienta que hemos desarrollado. Se puede comprobar cómo las dos trazas tienen la misma estructura, aunque puede haber alguna pequeña diferencia por la escala.

En la Figura 7 se muestra una traza correspondiente al detalle de la ejecución del workload con la configuración LL. En este caso las filas representan los threads que están en ejecución en las CPU's. Se puede apreciar como los threads van cambiando de estado constantemente en intervalos de tiempo muy cortos. Esto es debido principalmente al uso inadecuado de las CPU's del nodo, llegando a saturarlas y por lo tanto provocando una ejecución ineficiente. En este caso se producen un número muy elevado de cambios de contexto entre threads que pelean para conseguir las CPU. Usando el balanceo de la carga evitamos estas circunstancias y además, con el planificador local evitamos saturar tanto las CPU's y mejorar el rendimiento.

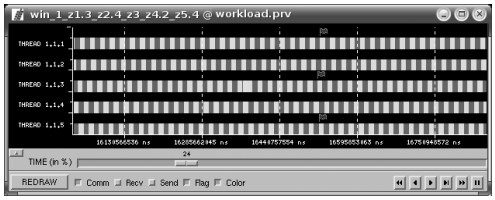

Figura 7. Detalle de la ejecución de una aplicación

Teniendo en cuenta que con los dos sistemas podemos obtener casi la misma información, el principal motivo por el cual hemos optado por este mecanismo es el reducido tamaño de las trazas obtenidas. En la Figura 8 se muestra una comparativa de tamaño entre los diferentes tipos de trazas para workloads de diferentes duraciones (traza binaria completa en formato Paraver, con sólo los datos del workload y la traza obtenida a partir de los ficheros XML). También hay que tener en cuenta que el objetivo que perseguimos es monitorizar los tres niveles de planificación comentados anteriormente. Sería muy complicado fusionar trazas de AIX trace de diversos nodos o de diferentes sistemas. También hemos tenido en cuenta otros factores, por ejemplo el hecho que XML es un formato estándar, o la facilidad de lectura, manejo y conversión a otros formatos.

	binaria paraver	paraver filtrada	paraver (XML)
10s	12 MB	8 MB	1,5 KB
1 min	150 MB	132 MB	40 KB
10 min	670 MB	640 MB	300 KB

Figura 8. Comparativa de tamaños de trazas

8. Conclusiones y trabajo futuro

En este artículo hemos presentado un mecanismo que nos permite visualizar y analizar workloads, mostrando el comportamiento de las aplicaciones que lo componen. También hemos presentado los resultados de la evaluación de workloads reales, tanto analíticamente como mediante trazas. En la evaluación de los workloads hemos comprobado que la planificación coordinada entre eNANOS Scheduler y NANOS-RM nos proporciona mejoras tanto en el tiempo de ejecución del workload como en el tiempo de respuesta de las aplicaciones. Hemos visto como las trazas, que se obtienen a partir de los ficheros XML que describen los workloads, son un buen mecanismo para analizarlos. En comparación con otros posibles mecanismos, hemos visto que nuestra propuesta tiene ventajas, sobre todo por el reducido tamaño de las trazas y la posibilidad de unificar la información de diversos niveles de planificación en una única traza.

Planteamos dos líneas básicas de trabajo futuro. Por un lado hay que añadir a la traza la información de todos los niveles de planificación, completando la coordinación entre niveles. Por el otro lado, hay que mejorar el esquema XML de la monitorización de los trabajos para soportar un nivel de detalle superior en las trazas (información relativa a CPU's, threads, etc.).

9. Agradecimientos

Este trabajo está realizado con el soporte del Ministerio de Ciencia y Tecnología, código TIN2004-07739-C02-01, y del proyecto europeo HPC-Europa con contrato 506079.

Referencias

[1] F. Berman, R. Wolski, et. al., "Application-Level Scheduling on Distributed Heterogeneous Networks", Proceeding of Supercomputing'96, 1996

[2] CEPBA Tools Web Site, http://www.cepba.upc.es/tools.htm

[3] J. Corbalán, R.M. Badia, J. Labarta, "Enanos Performance Tools for Autonomic Grid Resource Allocation Management", CEPBA-IBM Research Institute, 2002

[4] J. Corbalán, A. Duran, J. Labarta, "Dynamic Load Balancing of MPI+OpenMP applications", ICPP'04, 08-2004, Montreal, Quebec, Canada

[5] EZ-Grid Resource Brokerage System, http://www.cs.uh.edu/~ezgrid/

[6] J. Frey, T. Tannenbaum, M. Livny, I. Foster, S. Tuecke, "Condor-G: A Computation Management Agent for Multi-Institutional Grids". 10th HPDC, IEEE Press, Agosto 2001

[7] GridLab, A Grid Application Toolkit and Testbed, http://www.gridlab.org

[8] GridWay Project Web Site, http://www.gridway.org/

[9] Globus Information Services: Monitoring & Discovery (MDS) Web Site, http://www-unix.globus.org/toolkit/mds/

[10] IBM AIX trace facility Web Site, http://publib.boulder.ibm.com/infocenter/pseries/index.jsp

[11] IBM Research, Autonomic Computing, http://www.research.ibm.com/autonomic/

[12] MAUI Cluster Scheduler Web Site, www.clusterresources.com/products/maui

[13] Mercury Grid Monitoring System Web Site, http://www.lpds.sztaki.hu/mercury/

[14] K. Nadiminti, S. Venugopal, H. Gibbins, R. Buyya, "The Gridbus Broker Manual (v.2.0)", GRIDS Laboratory, http://www.gridbus.org/broker

[15] NASA Parallel Benchmarks Web Site, http://www.nas.nasa.gov/

[16] Network Weather Service (NWS) Web Site, http://nws.cs.ucsb.edu/

[17] OAR scheduler Web Site, http://oar.imag.fr/

[18] I. Rodero, J. Corbalán, R.M. Badia, J. Labarta, "eNANOS Grid Resource Broker", P.M.A. Sloot et al. (Eds.): EGC 2005, LNCS 3470, Amsterdam, Junio 2005, pp. 111-121

[19] J. Skovira, W. Chan, H. Zhon, "The EASY – LoadLeveler API Project ", LNCS 1162, 1996, pp. 41-47

[20] A.M. Weil, D. Feitelson, "Utilization, Predictability, Workloads and User Runtimes Estimates in Scheduling the IBM SP with Backfilling", In IEEE Trans. On Parallel and Distributed Syst. 12(6) pp.529-543, Jun 2001.

[21] R. Yahyapour, P. Wieder, "Grid Scheduling Architecture-Requirements", GGF GSA Research Group, Enero 2005