

Algoritmos de planificación dinámica en entornos Grid

Bertogna, Leandro
Departamento de Ciencias de la Computación
Universidad Nacional del Comahue
mlbertog@uncoma.edu.ar

Naiouf Marcelo , De Giusti Armando
Instituto de Investigación en Informática LIDI
Universidad Nacional de La Plata
{degiusti,mnaiouf}@lidi.info.unlp.edu.ar

Resumen

Debido a la gran cantidad de recursos de hardware y software que componen los sistemas Grid, cada uno con características y políticas propias, la gestión del sistema se torna compleja. Un aspecto clave que permite obtener la máxima potencia del sistema de manera eficiente se encuentra en el algoritmo de planificación de tareas. Debido al dinamismo de los recursos y las organizaciones que los contienen los planificadores tradicionales no son una opción válida. Los nuevos planificadores deben contemplar aspectos tales como heterogeneidad de recursos, adaptación dinámica al entorno y altos costos de comunicación. En este trabajo se presenta una línea de investigación sobre algoritmos de planificación Grid, se analiza una propuesta de planificación y se describen aspectos que se encuentran en desarrollo.

1. Introducción

Debido a la gran cantidad de recursos de hardware y software que componen los sistemas Grid, cada uno con diferentes características y complejidades, se torna imperioso simplificar y automatizar de manera eficiente su administración. Tareas de gestión rutinarias que requieren recursos humanos especializados deberían disminuir en forma considerable para transformarse sólo en definición de políticas, que sistemas autónomos de gestión de infraestructura cumplieran o hicieran cumplir.

Recursos pertenecientes a varios dominios administrativos son compartidos bajo políticas definidas localmente. Estas políticas regulan cómo se debe compartir, a quién se le permite acceder y bajo qué condiciones se accede a los recursos. El problema que se encuentra detrás del concepto Grid es cómo coordinar estos recursos compartidos y solucionar problemas en organizaciones virtuales, dinámicas, multi-institucionales [1, 2].

Un punto clave para la gestión de recursos en los sistemas Grid se encuentra en la capa intermedia (*middleware*): esta coordina recursos e intercambia información sin importar su ubicación física. Cuando alguna aplicación utiliza Grid y realiza requerimientos de hardware para su ejecución esta capa de software es la encargada de buscar, seleccionar e instanciar los recursos apropiados en donde realizar su despliegue.

La componente que interviene de forma activa de este proceso es el planificador de Grid o metaplanificador. Este permite acceder a los recursos distribuidos en distintas organizaciones físicas y comunicarse con los distintos administradores de recursos locales. Los administradores de recursos se convierten de esta manera en los proveedores de servicio y el metaplanificador los coordina y utiliza, en base a criterios preestablecidos.

Para lograr el máximo potencial de los recursos distribuidos, de manera efectiva y eficiente, los algoritmos que utilice el metaplanificador son de fundamental importancia. Desafortunadamente los algoritmos de planificación de sistemas paralelos y distribuidos tradicionales en general disponen de recursos homogéneos y dedicados, y por lo tanto no logran satisfacer los requerimientos de este nuevo entorno.

Este trabajo presenta una línea de investigación relacionada con el análisis e implementación de un modelo de planificación dinámico para sistemas Grid. En la segunda sección se describirán conceptos de planificación para estos entornos, en la tercera sección se presentan la arquitectura y el algoritmo en el que se trabaja actualmente finalizando con aspectos abiertos de la investigación y resultados esperados.

2. Algoritmos Grid

Los algoritmos de planificación han sido estudiados como problemática en los sistemas tradicionales de paralelismo y sistemas distribuidos, como los sistemas de multiprocesadores simétricos (SMP), masivamente paralelos (MPP) o clusters de Workstation (COW). Cada uno de estos algoritmos de planificación tomó en cuenta características propias de los sistemas para lograr máxima eficiencia. En la Tabla 1 se puede observar la evolución de estas características [3].

Arquitectura	DSM/MPP	COW	GRID
Cronología	Fines de los 70's	Fines de los 80's	Mediados de los 90'
Conexión de los sistemas	Bus, switches	LAN, ATM	WAN/Internet
Costo de Interconexión	Muy Bajo / Despreciable	Bajo/ Usualmente no despreciable	Alto/ No despreciable
Heterogeneidad de Conexión	No	Bajo	Alto
Heterogeneidad de Nodos	No	Bajo	Alto
Grupo de Recursos	Predeterminado y Estático	Predeterminado y estático	Indeterminado y dinámico
Política de administración de recursos	Única	Única	Múltiple
Única Imagen del sistema	Si	Si	No

Tabla 1. Características de los sistemas de cómputo paralelos y distribuidos

Si bien se pueden tomar algunos modelos de planificación tradicional, las características propias de los sistemas Grid hace que esta tarea en muchos casos sea difícil de implementar. A continuación se describen en detalle las particularidades de entornos Grid:

Heterogeneidad y Autonomía

A pesar de que la heterogeneidad no es nueva en los algoritmos de planificación, aún es un gran desafío para su diseño. En los sistemas Grid debido a que los recursos son distribuidos en múltiples dominios sobre Internet, los nodos de cómputo, datos e interconexión varían entre organizaciones.

En los sistemas paralelos y distribuidos tradicionales, los recursos de computación generalmente están administrados en un punto central. El planificador tiene información completa de las tareas en ejecución y pendientes, y la carga de trabajo de los nodos, además de administrar colas de tareas y el grupo de recursos. De esta manera puede predecir el comportamiento de los recursos y es capaz de asignar tareas de acuerdo a requerimientos de rendimiento. En los sistemas Grid los recursos son autónomos y el planificador no tiene control de los recursos, y no puede influir en las políticas de las organizaciones, por lo que se torna difícil estimar el tiempo de ejecución de cada tarea en los distintos nodos de cómputo.

Dinamismo de Rendimiento

Realizar una planificación depende de las estimaciones de rendimiento que pueden proveer recursos candidatos, especialmente cuando el algoritmo es estático. Los algoritmos Grid trabajan en ambientes dinámicos donde el rendimiento de los recursos disponibles cambia constantemente. El cambio proviene de la autonomía de los recursos de cómputo, y de la competencia de las aplicaciones por estos recursos. En el caso de los sistemas Grid los recursos no son asignados con exclusividad a las aplicaciones Grid.

Selección de recurso y Separación de Datos

En los sistemas tradicionales, el código de ejecución y los datos de entrada/salida se encuentran generalmente en el mismo sitio, o las fuentes de datos y el destino de salida se determinan antes de que la aplicación inicie su ejecución. De esta manera, el costo de la transferencia de datos es mínimo o es constante, y en los algoritmos de planificación rara vez se considera. Los sistemas Grid involucran distintos recursos de cómputo, sitios de almacenamiento y redes de ancho limitado. Los sitios de cómputo son seleccionados por el planificador de acuerdo a modelos de rendimiento y estatus de los recursos. Además, la capacidad de conexión cobra mayor importancia ya que no solo es limitada sino que es compartida por otras tareas que no necesariamente son de la aplicación Grid.

Los desafíos que se han mencionado ponen obstáculos significativos al diseño e implementación eficiente y efectiva de los sistemas de planificación Grid. A pesar de esto, algunos de los conceptos de los planificadores tradicionales pueden servir en aspectos puntuales a medida que se desarrollan nuevos planificadores Grid [4].

3. Línea de Investigación

La contribución de esta línea de trabajo se encuentra en el desarrollo de un planificador de sistemas Grid. El desarrollo del planificador consta de distintos módulos, donde los dos más importantes son los que tienen la responsabilidad de asignar las tareas. El primero, único en el sistema y con una visión completa y el segundo, con tantas instancias como organizaciones componen al sistema Grid, con una visión local de los recursos.

En este trabajo de investigación se analiza un entorno para aplicaciones paralelas. El sistema grid selecciona e interconecta un grupo de máquinas donde la aplicación pueda ejecutarse de manera eficiente. La eficiencia se mide desde el punto de vista de la aplicación y del uso de los recursos que consume. Para que sea factible instanciar máquinas con parámetros y configuración determinados, en cada máquina física del sistema se instancian una o mas máquinas virtuales, siendo estos los recursos que administran los planificadores.

El modelo de aplicación paralela es master-worker. Todas las tareas realizan el mismo tipo de cálculo y envían /reciben la misma cantidad de datos todo esto en forma independiente una de otras. Este esquema es muy común en simulaciones o aplicaciones graficas. Se asume también que cada vez que un nuevo grupo de máquinas se publica el sistema se caracteriza y se realizan pruebas de rendimiento.

3.1 Metaplanificador

En el trabajo [5] el metaplanificador, según el requerimiento de cómputo y comunicación de una aplicación paralela, selecciona recursos y los conecta en una red virtual. Encontrar un grupo de recursos que puedan trabajar cooperativamente en un entorno distribuido de manera eficiente no es

una tarea trivial. Uno de los objetivos es que esta búsqueda fuera lo más rápida y la solución fuera lo más aproximada a la óptima.

Para solucionar el problema de una manera analítica, los recursos disponibles tanto de cómputo como de red fueron mapeados a un grafo: los recursos de cómputo como nodos y los enlaces de comunicación como aristas. El peso de los nodos y aristas correspondieron a características de las máquinas y ancho de banda respectivamente. Las aristas recibieron menos peso cuanto mas ancho de banda disponían y el peso de los nodos se calculó a través de una función, por ejemplo costo económico, poder de cómputo, etc.

La estrategia se divide en dos etapas, donde la primera consiste en seleccionar a un grupo de clusters. En esta etapa se usa una heurística para encontrar al grupo de equipos óptimo, tomando en cuenta la comunicación y el poder de cómputo. Una vez que se obtiene este grupo comienza la segunda etapa: por cada cluster se realiza un análisis y se evalúa cuántas máquinas será conveniente tomar del grupo en cada caso. Para este caso, se calcula la tasa de transferencia de la red y se compara con la cantidad de datos que el conjunto de máquinas transmitirá. Si se tomaran más máquinas de lo debido el umbral de rendimiento caería, habiendo máquinas que deban esperar para poder transmitir los datos computados estando ociosas en ese momento.

3.2 Planificador local

El planificador local [5,6] también conocido en algunos trabajos como “*binder*” es el módulo determinado de instanciar las aplicaciones en los recursos seleccionados. Provee los datos de entrada, inicia la ejecución y monitorea la ejecución de las aplicaciones. Un ejemplo de este caso es Globus GRAM (Grid Resource Allocation and Management) [7].

En esta investigación además de tener la funcionalidad básica, el planificador local hace también de gestor de los clusters de máquinas virtuales. En los planificadores tradicionales de cluster, como Torque o SGE, los recursos son estáticos y predeterminados. En este caso, el planificador local coordina con el metaplanificador para asignar recursos en forma dinámica.

Otra funcionalidad que brinda el planificador local es la de gestión dinámica en tiempo de ejecución aprovechando la funcionalidad de migración en vivo de máquinas virtuales. Esta característica se aplica sin interrupción de las aplicaciones que se estuvieran ejecutando en ese momento sobre las máquinas migradas. Si el metaplanificador ofreciera la ejecución de nuevas tareas con bajo requerimiento de entrada y salida de red, y el planificador local tuviera gran porcentaje de máquinas ociosas podría liberar ancho de banda consolidado máquinas virtuales en un solo equipo. Este desplazamiento reduciría el rendimiento de las aplicaciones liberando ancho de banda, y como resultado de esta capacidad libre en la red el planificador sería capaz de instanciar máquinas ociosas con máquinas virtuales para la nueva aplicación. Esta solución aumenta el porcentaje de utilización de equipos y reduce el tiempo total de ejecución del sistema, siempre y cuando el tiempo de penalización en la baja de rendimiento sea menor que la ganancia de la incorporación de nuevas máquinas.

Esta implementación en las condiciones adecuadas llega a una mejora del 20% en el tiempo de cómputo de una aplicación paralela con las máquinas a un 100% de ejecución por el tiempo que se ejecuta la aplicación paralela.

4. Estudios Actuales y Resultados esperados

Actualmente se esta encuentra en desarrollo un simulador para probar distintos entornos y lotes de tareas y observar el comportamiento del planificador global y local. Se encuentra en estudio la

incorporación de nuevas heurísticas de planificación para tareas independientes con el objetivo de realizar una comparación más exhaustiva de la propuesta contra las alternativas habituales. Algunos de los algoritmos que se están analizando son Min-Max y Min-Min [10, 11].

Los trabajos que se están llevando a cabo actualmente en el módulo de administración local son la extensión no solo a relación cómputo comunicación sino a fijar parámetros para una caracterización más detallada del uso de máquinas virtuales como interferencia de rendimiento [8, 9]. Los resultados esperados de la incorporación de este conocimiento como nuevas reglas de inferencia en los módulos locales brindarán una mejor adaptación dinámica redundando en un mayor incremento en el rendimiento global del sistema.

4. Referencias

- [1] I. Foster, C. Kesselman and S. Tuecke, The Anatomy of the Grid: Enabling Scalable Virtual Organizations, in the International J. Supercomputer Applications, 15(3), pp. 200-220, fall 2001.
- [2] I. Foster and A. Iamnitchi, On Death, Taxes, and the Convergence of Peer-to-Peer and Grid Computing, in Proc. of 2nd International Workshop on Peer-to-Peer Systems (IPTPS'03), Berkeley, CA, SA, February 2003.
- [3] F. Dong and S. G. Akl, "Scheduling Algorithms for Grid Computing: State of the Art and Open Problems," Technical Report of the Open Issues in Grid Scheduling Workshop, School of Computing, University Kingston, Ontario, January 2006
- [4] A. Andrieux, D. Berry, J. Garibaldi, S. Jarvis, J. MacLaren, D. Ouelhadj and D. Snelling, "Open Issues in Grid Scheduling", Official Technical Report of the Open Issues in Grid Scheduling Workshop, Edinburgh, UK, October 2003.
- [5] Bertogna Leandro, Naiouf Marcelo, De Giusti Armando E., Luque Emilio. "Planificación dinámica de clusters a demanda en entornos Grid". CACIC 2007.
- [6] Y. Zhu, A Survey on Grid Scheduling Systems, Department of Computer Science, Hong Kong University of science and Technology, 2003
- [7] K. Czajkowski, I. Foster, N. Karonis, C. Kesselman, S. Martin, W. Smith, and S. Tuecke, A Resource Management Architecture for Metacomputing Systems, In D.G. Feitelson and L. Rudolph, editors, in Proc of the 4th Workshop on Job Scheduling Strategies for Parallel Processing, LNCS Vol. 1459 pp. 62–82, Orlando, Florida USA, March 1998.
- [8] Younggyun Koh, Rob Knauerhase, Paul Brett, Mic Bowman, Zhihua Wen, and Calton Pu, "An Analysis of Performance Interference Effects in Virtual Environments" in the IEEE International Symposium on Performance Analysis of Systems and Software. 2007.
- [9] J. N. Matthews, W. Hu, M. Hapuarachchi, T. Deshane, D. Dimatos, G. Hamilton, M. McCabe, and J. Owens, "Quantifying the performance isolation properties of virtualization systems," in ExpCS '07: Proceedings of the 2007 workshop on Experimental computer science, ACM, 2007
- [10] X. He, X. Sun and G. Laszewski, A QoS Guided Min-Min Heuristic for Grid Task Scheduling, in J. of Computer Science and Technology, Special Issue on Grid Computing, Vol.18, No.4, pp.442-451, July 2003
- [11] M. Wu, W. Shu and H. Zhang, Segmented Min-Min: A Static Mapping Algorithm for Meta-Tasks on Heterogeneous Computing Systems, in Proc. of the 9th Heterogeneous Computing Workshop HCW'00), pp. 375--385, Cancun, Mexico, May 2000.