Estudio de algoritmos de calendarización para flujos de trabajo

Fernando Aguilar Reyes

Tesina para obtener el título de Ingeniero en Computación

Instituto Tecnológico Autónomo de México
Río Hondo #1, Progreso Tizapán, Del. Álvaro Obregón, 01080
México, Distrito Federal

6 de enero de 2014

Índice general

Índice de figuras

Capítulo 1

Introducción

Cada día, los cómputos son más complejos. Hay bancos que procesan millones de transacciones diarias. Las películas de animación requieren vastos tiempos de cómputo. Muchos proyectos de computación científica requieren hacer numerosos cálculos para llegar a resultados pertinentes. ¿Qué tienen en común todas estas aplicaciones? Ya hemos mencionado que toman mucho tiempo calcularse. Entonces, es natural pensar que se puede distribuir el gran trabajo que requieren estos proyectos entre varias computadoras para disminuir el tiempo total de ejecución y de esta forma, tener una solución escalable.

Lograr esta paralelización requiere un esfuerzo por parte del desarrollador de la solución. Aunque estas técnicas de paralelización son muy efectivas, éstas son aplicadas cuando el problema a resolver ha sido bien definido y cuando sólo hay una instancia del problema. Ahora bien, cuando la solución involucra varios pasos que están relacionados entre sí, por ejemplo, que el programa C requiera de la salida del programa A y del programa B para que pueda funcionar. O, cuando estamos definiendo los grandes bloques de solución del problema, hay una cierta secuencia que debemos seguir y, dentro de dicha secuencia, hay algunos pasos que podemos resolver de manera concurrente. Por lo tanto, estamos haciendo un *modelo* de nuestro problema.

Este modelo también es llamado *flujo de trabajo*. De manera muy abstracta, un modelo de trabajo es un conjunto de pasos que modelan la ejecución de un proceso. Este concepto

ha sido aplicado en numerosas áreas. En el ámbito de los negocios, se utiliza diagramas dibujados con los bloques y reglas del *Bussiness Process Execution Language* para modelar procesos de negocio.

En el ámbito de las ciencias en computación, los flujos de trabajo son aplicados para modelar problemas que requieran varios pasos para su solución. Algunos flujos de trabajo requieren mucho tiempo tiempo de ejecución para sus pasos, como son el caso de los flujos de trabajo científicos. Otros flujos de trabajo son muy simples pero se requieren que se ejecuten muchos de éstos. De esta forma, es desable distribuir la ejecución de éstos flujos de trabajo entre varias computadoras. Si bien es posible paralelizar algunos pasos de la ejecución de nuestro flujo de trabajo utilizando las técnicas antes mencionadas, hay restricciones de orden que se deben respetar, por lo cual, se debe planear la ejecución del flujo de trabajo.

Hay varias maneras de hacer esta planeación de la ejecución, también llamada *calendarización*. Cabe aclarar que definimos la calendarización como la asignación de recursos a tareas de tal modo que se ejecuten todas las tareas. Con ello, se desea encontrar una forma óptima de hacer esta calendarización, como reducir el tiempo de ejecución total del flujo de trabajo. Sin embargo, con la aparición del cómputo en la nube, es posible ejecutar nuestro flujo de trabajo con otras restricciones, como minimizar el presupuesto necesario para la ejecución del flujo afectando el tiempo de ejecución.

En esta tesina se hace un estudio de los principales algoritmos de calendarización de flujos de trabajo, con énfasis en los algoritimos utilizados en cómputo distribuido, en especial en cómputo en la nube. En el capítulo 2 se habla de un estudio detallado del concepto de flujos de trabajo y su aplicación en computación. El capítulo 3 trata los principales enfoques de cómputo distribuido para ejecutar estos flujos. En el capítulo 4 se hace un estudio de los principales algoritmos de calendarización de los flujos. Finalmente, en el capítulo 6 discutiremos algunas conclusiones sobre el análisis de estos algoritmos.

Capítulo 2

Flujos de trabajo

2.1. Definición y ejemplos

Un flujo de trabajo es un conjunto de pasos que modelan la ejecución de un proceso [6]. En la literatura especializada, los flujos de trabajos también son conocidos como *workflows*. En particular, se estudian a los flujos de trabajo utilizados para vislumbrar la ejecución de un proceso de cómputo. A continuación, se muestran algunos ejemplos de estos flujos de trabajo.

2.1.1. Anotación de proteínas

En el proyecto *e-Protein*, realizado por la Escuela Imperial de Londres, se realizó un flujo de trabajo para la anotación de proteínas. El objetivo del proyecto [7] era la identificación y anotación de partes de proteínas que expliquen su estructura y su función. En la figura 2.1 se muestra el flujo de trabajo desarrollado, donde las cajas representan los programas que son ejecutados para cada paso del proceso de anotación, y las líneas que conectan a las cajas representan las dependencias de datos entre los programas, es decir, si una caja tiene una línea que apunta a ella, significa que dicho programa depende de otro porgrama determinado por el otro extremo de la flecha.

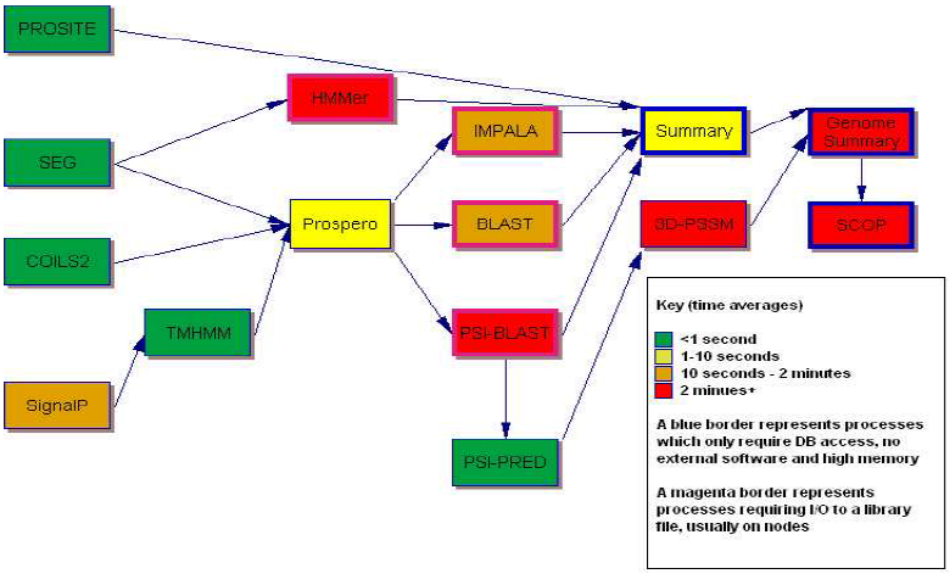

Figura 2.1: Flujo de trabajo para la anotación de proteínas, diseñado en la Escuela Imperial de Londres para el proyecto *e-Protein*.

2.1.2. Curvas de amenaza sísmica

Otro notable ejemplo es el proceso para generar curvas de amenaza que describen las probabilidades de que ocurra un temblor en una determinada área. Para elaborar estas curvas, los científicos del Centro de Terremotos del Sureste de California (SCEC por sus siglas en inglés) tienen que realizar una gran cantidad de simulaciones para que sus resultados puedan ser combinados y expresados en la curva de amenaza [5]. El flujo de trabajo para generar la curva de amenaza se muestra en la figura 2.2.

2.1.3. Generación de facturas

Recientemente en México, el Sistema de Administración Tributaria emitió los lineamientos para que las operaciones de compra-venta entre personas físicas y morales puedan ser registradas por medio de facturas electrónicas. Para generar estos Comprobantes Fiscales Digitales, las empresas tienen que hacer procesos de validaciones de RFC y encriptar el

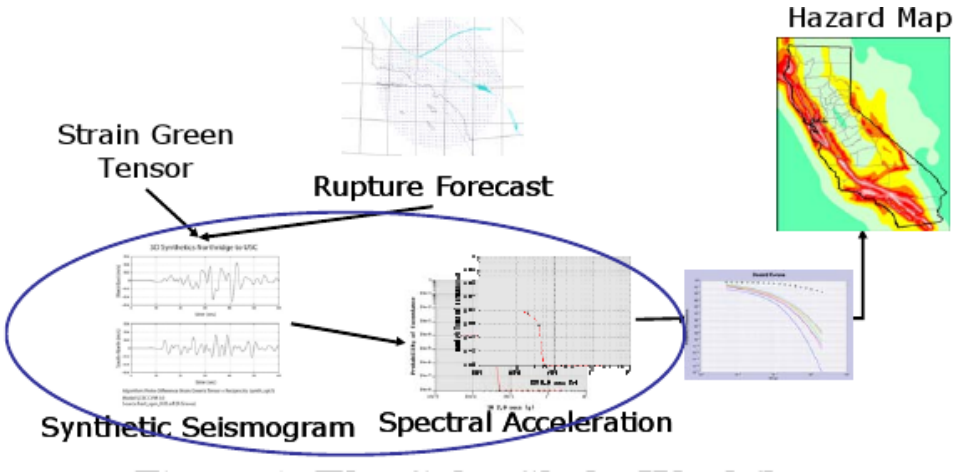

Figura 2.2: Flujo de trabajo para generar curvas de amenaza, elaborado por el SCEC.

contenido de la factura con un mecanismo de llave privada. Naturalmente, este proceso de generación de factura requiere de varias actividades. En la figura **??** se muestra el flujo de trabajo que utiliza una empresa para generar sus facturas. Este flujo se encuentra documentado en una tesina presentada por Alcerreca [3].

2.2. Estructura de los flujos de trabajo

Como hemos visto en los dos ejemplos anteriores, los flujos de trabajo presentados describen los *pasos* que necesitan para calcular la solución de un problema. En éstos, no se ha hablado sobre detalles para hacer los cómputos necesarios para cada flujo. Tampoco definen las plataformas de cómputo en que se ejecutan estos flujos. Tan sólo definen los grandes pasos para solucionar el problema y las dependencias que tienen estos pasos.

Ahora, es muy común que estas dependencias estén dictadas por los datos que requiere cada paso para funcionar. Sin embargo, hay situaciones en los que los pasos del flujo no requieren los datos del paso anterior para funcionar, sino que las dependencias están marcadas por el orden temporal que deben seguir estos pasos. Así, se puede notar una ambigüedad en la definición de un flujo de trabajo.

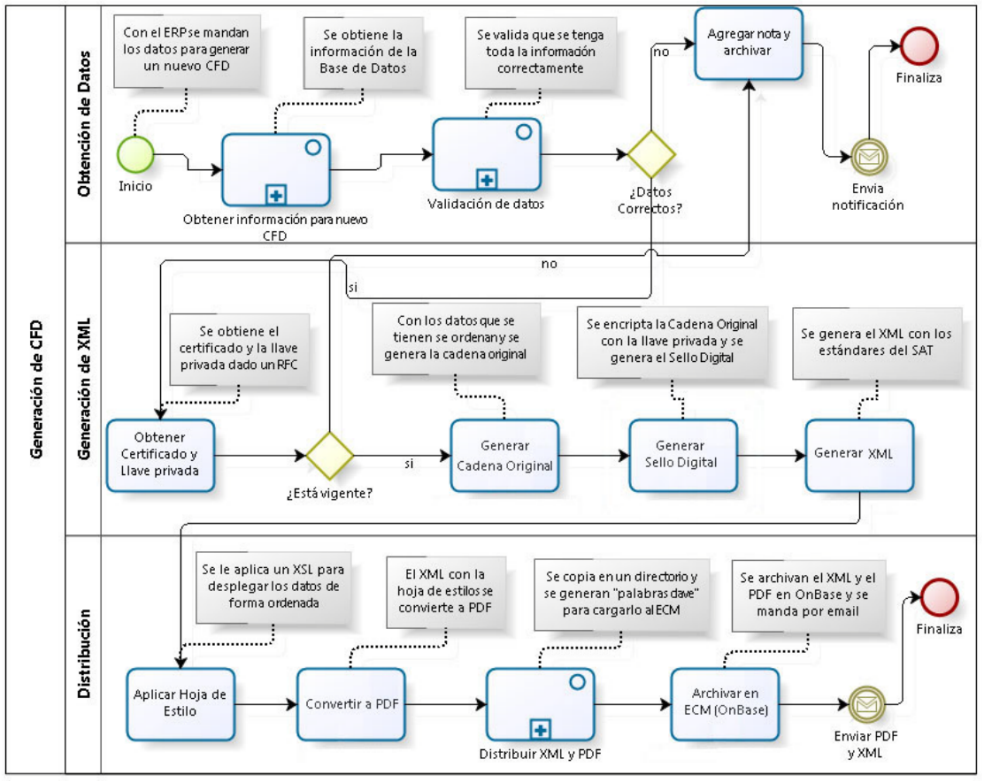

Figura 2.3: Flujo de trabajo para generar facturas electrónicas, utilizado por una empresa de cementos.

2.2.1. Perspectivas de los flujos de trabajo

El trabajo de Van der Aalst et al. [8] identifica una serie de perspectivas vislumbrados en las definiciones de las especificaciones de un flujo de trabajo en las que se basan los sistemas que administran la ejecución de éstos. Las cuatro perspectivas se enuncian a continuación:

- **Perspectiva de control de flujo.** Describe las relaciones de las actividades (pasos) con estructuras de control, tales como: secuencia, decisión, ejecución de actividades en paralelo y punto de sincronización conjunta[1].
- **Perspectiva de datos.** En ella, los flujos de trabajo describen las entradas y salidas de datos, tanto de ejecución como de control, que se tienen en cada actividad del flujo. También se toma en cuenta los datos locales a cada actividad; es decir, que sólo son necesarios dentro del contexto de ésta.
- **Perspectiva de recursos.** Muestra cuáles son los recursos con los que se cuentan para ejecutar el flujo de trabajo y la forma en que estos recursos se encuentran organizados. Estos recursos pueden ser desde entidades de cómputo hasta roles con responsabilidades específicas cumplidas por actores humanos.
- **Perspectiva operacional.** Aquí se detallan las operaciones elementales necesarias en cada actividad para ejecutar el flujo de trabajo. Estos detalles incluyen las transferencias de datos entre las operaciones y su correspondencia en programas.

Cabe aclarar que estas perspectivas están relacionadas entre sí de modo que el control de flujo es la base en la que descansan las demás perspectivas. Esto es porque la perspectiva de datos requiere que el control de flujo tenga los datos de entrada y salida como prueba de que se cumplieron las pre y post-condiciones de cada actividad, respectivamente; la perspectiva de recursos define con qué se ejecutarán y almacenará los datos del flujo de trabajo; mientras que la perspectiva operacional trata los detalles sobre cómo se utilizan *físicamente* los recursos, los datos y los programas a lo largo del flujo de trabajo.

[1]Esta estructura de control también es conocida como *join synchronization*

2.3. Calendarización como control de flujo

El hecho de que la perspectiva de control de flujo sea la base de las demás perspectivas indica que la forma en que controla la ejecución del flujo determina de manera fundamental el rendimiento de la ejecución total de una instancia del flujo. Por lo tanto, es de vital importancia encontrar métodos de calendarización que permitan encontrar correspondencias entre recursos y actividades que cumplan con los requisitos dictados en las especificaciones examinadas en la perspectiva del control del flujo y que también maximicen el rendimiento de la ejecución de todo el flujo en general.

En este trabajo, se establecerán las siguientes consideraciones para limitar el estudio de los algoritmos de calendarización, a saber:

- Los flujos de trabajo están representados como *grafos dirigidos acíclicos*, con el objetivo de simplificar el estudio
- Las tareas (o actividades) de los flujos de trabajo son consideradas *atómicas*, i.e., una tarea no puede ejecutarse incompletamente. Algunos sistemas de administración de ejecución de flujos de trabajo tienen mecanismos para lidiar con estos errores, pero el estudio de éstos queda fuera de los límites de este trabajo.
- La información de las *demás perspectivas* se puede requerida, dependiendo si el mecanismo de calendarización lo considere necesario para tomar mejores decisiones.

Capítulo 3

Cómputo distribuido

Los flujos de trabajo requieren de recursos de cómputo para su ejecución. En los ejemplos anteriores se mencionaron aplicaciones científicas que, por su naturaleza, requieren vasto tiempo de ejecución para llevarse a cabo. Por otro lado, también existen aplicaciones de negocio que, si bien son computacionalmente sencillas, se requiere ejecutar una gran cantidad de instancias de estos flujos de trabajo. Por ello, ejecutar cualquiera de estos flujos de trabajo en una sola computadora resulta prohibitivo. Así, se hace uso de varias computadoras para distribuir el esfuerzo para correr estos grandes procesos.

Dependiendo de cómo estén organizados estas computadoras, se definen los enfoques para correr los flujos de trabajo con cómputo distribuido.

3.1. Enfoques de cómputo para flujos de trabajo

Diversos enfoques se han aplicado para distribuir la ejecución de un flujo de trabajo entre varias computadoras. De acuerdo a Buyya et al., los enfoques de cómputo más importantes para los flujos de trabajo son los *clusters*, los *grids* y las *nubes* [4]. A continuación, explicaremos cada uno de los enfoques.

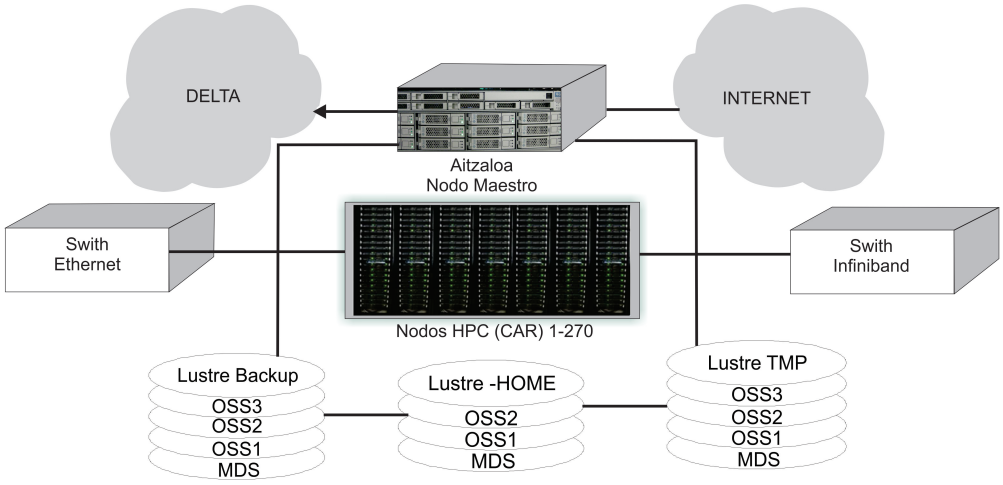

Figura 3.1: Topología del cluster Aitzaloa

3.1.1. Clusters

Los *clusters* son sistemas distribuidos, paralelos, compuestos de varias computadoras que son vistas como un único recurso de cómputo [4].

Un ejemplo de un cluster es la instalación de la Universidad Autónoma Metropolitana, campus Iztapalapa, llamada *Aitzaloa*, compuesta por 270 nodos de cómputo, cada uno equipado con dos procesadores Intel Xeon Quad-Core y 16GB en RAM; los nodos están conectados entre sí por medio de switches Ethernet e Infiniband. El cluster también cuenta con un sistema de archivos distribuido basado en Lustre. La capacidad real de cómputo del cluster Aitzaloa es de 18.4 teraFLOPS [2].

El detalle de la topología del cluster se puede apreciar en la figura 3.1, en donde podemos apreciar que los switches son los puntos de conexión entre el nodo maestro, el sistema de almacenamiento distribuido y los nodos de cómputo.

3.1.2. Grids

Los *grids* son sistemas distribuidos, paralelos, compuestos de computadoras autónomas y geográficamente distribuidas que pueden trabajar en conjunto o de manera independiente de

acuerdo a los objetivos, políticas y mecanismos de uno o varios administradores del sistema, es decir, un grid puede ser compartido entre varias instituciones [4].

El proyecto *LANCAD*[1] es un buen ejemplo, pues une el cluster *Aitzaloa* de la UAM, el cluster de la UNAM *KamBalam*, y el cluster *Xiuhcoatl* del CINVESTAV por medio de una red de fibra óptica instalada en las estaciones del Sistema de Transporte Colectivo Metro. La suma de la potencias reales de cada *nodo robusto* del grid es de 48.55 teraFLOPS [1].

En la figura 3.2 se muestra los tramos de línea de Metro que cuentan con fibra óptica para conectar cada uno de las supercomputadoras (clusters) de las tres instituciones.

3.1.3. Nubes

Las *nubes* (clouds) son sistemas distribuidos, paralelos, compuestos de computadoras o máquinas virtuales interconectadas que son aprovisionadas para usarse como uno o varios recursos de cómputo, de acuerdo a un contrato de nivel de servicio acordado entre el proveedor de la nube y el cliente [4].

Empresas nuevas y existences proveen servicios de cómputo en la nube, tales como GoGrid, Rackspace, Amazon, Microsoft, IBM, Oracle, entre otras. La forma en que operan es la siguiente: se paga cierta cantidad por utilizar servicios de cómputo o almacenamiento durante determinado tiempo. Así, los clientes no tienen que invertir grandes cantidades de dinero para contar con una gran infraestructura como en el caso de los los clusters y los grids.

3.2. Comparación de los enfoques

Si bien estos enfoques difieren, principalmente, en la forma en que están organizadas las unidades de cómputo, podemos enumerar algunas observaciones.

Primero, es natural ver que los grids están compuestos de clusters, pero también podría verse a un grid como un gran cluster. Hasta cierto punto este razonamiento parece correcto.

[1]Laboratorio Nacional de Cómputo de Alto Rendimiento

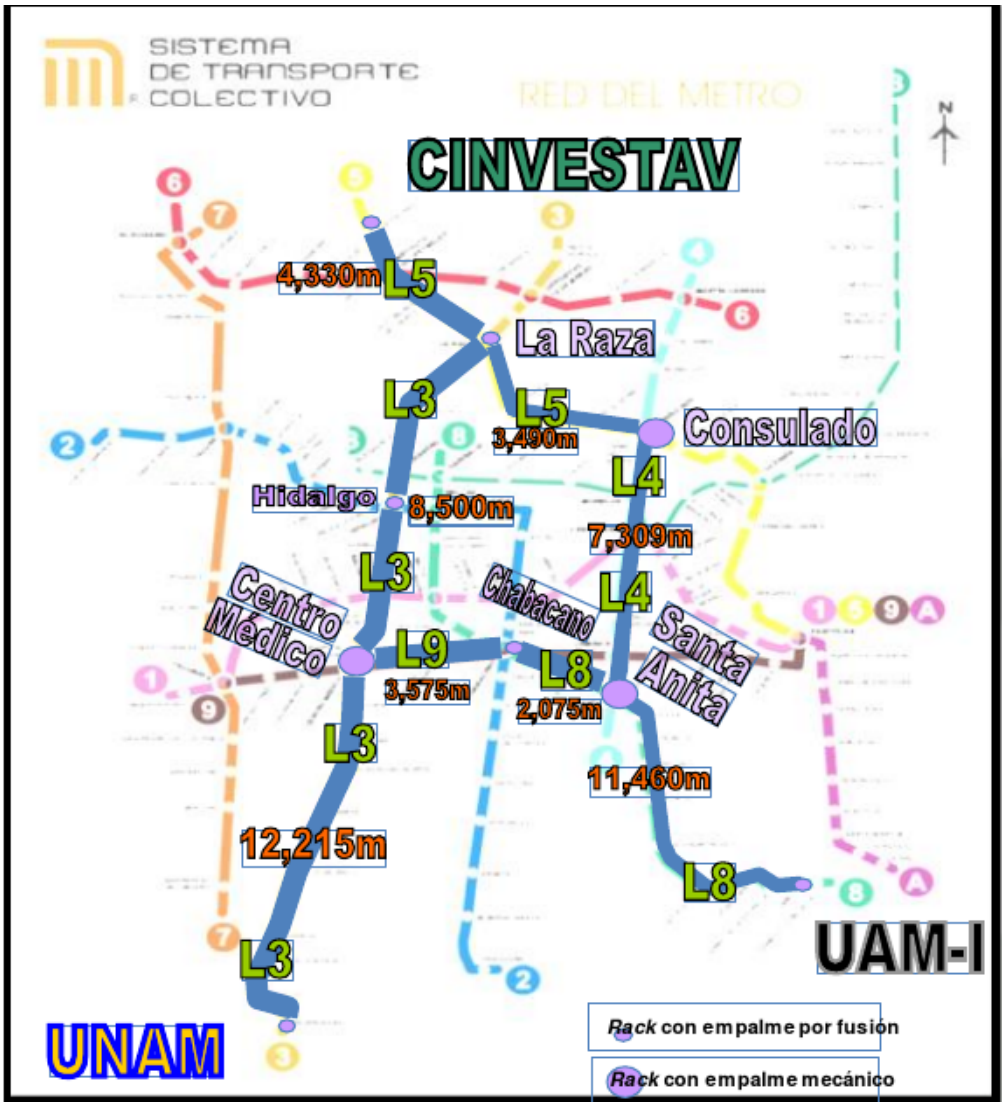

Figura 3.2: Detalle de la red de fibra óptica para conectar los nodos robustos del grid del proyecto LANCAD

Sin embargo, la diferencia importante entre clusters y grids radica en el hecho de que el grid es comúnmente administrado por varios técnicos de varias instituciones que, pueden tener objetivos y necesidades diversas, mientras que un cluster requiere de menos administradores y se asume una mayor flexibilidad para enfrentar los fallos de un cluster.

Segundo, tanto los grids como las nubes son físicamente muy similares: ambos consisten de varios clusteres interconectados y distribuidos geográficamente. Entonces ¿cuál es la diferencia? Lo que distingue a las nubes de los grids son dos características: (1) en una nube, se utilizan *máquinas virtuales* para distribuir el recurso. El usuario tiene la vista de una parte del grid como si fuera una computadora dedicada exclusivamente para éste. Por otro lado, en los grids, es común que el usuario acceda a él con una cuenta asignada por un administrador, de tal suerte que se tiene una vista de una gran máquina que es compartida entre varios usuarios. Muchos de los administradores de grids en el mundo administran los trabajos encargados por los usuario de tal modo que se ejecuten lo más rápido posible con los recursos disponibles en un tiempo dado. Los usuarios, en algunos casos obtienen acceso al grid y trabajan sus proyectos de manera gratuita o, en caso contrario, pagan una cuota por un tiempo fijo de cómputo. Pero, en el caso de las nubes, se agrega un modelo económico para acceder a los recursos llamado *pay as you go*, en donde el usuario paga una cuota por los recursos utilizados, sin alguna limitación mas que el presupuesto. Asi, el usuario podría pagar una gran cantidad para que su flujo de trabajo se ejecute en el menor tiempo posible o también podría pagar una menor cantidad sacrificando el tiempo total de ejecución del flujo.

Capítulo 4

Calendarización

La calendarización[1] es el proceso de asignación de recursos a tareas, de tal modo que se define un orden de ejecución de las tareas, teniendo lugar diferentes combinaciones de recursos y tareas. En este capítulo se define de la forma más elemental el problema de la calendarización para estudiar sus propiedades y también se explica la relación de este problema fundamental con la calendarización de flujos de trabajo.

4.1. Definición del problema

4.2. La complejidad de calendarizar

[1]También conocida en la literatura especializada como *scheduling*

Capítulo 5

Algoritmos de calendarización

5.1. Algoritmos de Mejor Esfuerzo

5.2. Algoritmos de Calidad en el Servicio

Capítulo 6

Software para la administración y ejecución de flujos de trabajo

6.1. Software orientado a clusters

6.1.1. Open Grid Scheduler

6.2. Software orientado a grids

6.2.1. SwinDew-G

6.2.2. Pegasus

6.3. Software orientado a nubes

6.3.1. ANEKA

6.3.2. SwinDew-C

Capítulo 7

Conclusiones

Bibliografía

[1] Cluster Híbrido de Supercómputo — Xiuhcoatl — Cinvestav. `http://clusterhibrido.cinvestav.mx/`. Consultado el 10 de noviembre de 2013.

[2] Laboratorio de Supercómputo y Visualización en Paralelo. `http://supercomputo.izt.uam.mx/infraestructura/aitzaloa.php`. Consultado el 10 de noviembre de 2013.

[3] Sebastian Alcerreca Alcocer. Emisión de Comprobantes Fiscales Digitales (CFD) - Desarrollo del sistema de emisión, distribución y almacenamiento de CFD de la corporación Montecito. Tesis de licenciatura, Instituto Tecnológico Autónomo de México, 2013.

[4] Rajkumar Buyya, Chee Shin Yeo, Srikumar Venugopal, James Broberg, and Ivona Brandic. Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility. *Future Generation computer systems*, 25(6):599–616, 2009.

[5] Ewa Deelman, Scott Callaghan, Edward Field, Hunter Francoeur, Robert Graves, Nitin Gupta, Vipin Gupta, Thomas H Jordan, Carl Kesselman, Philip Maechling, et al. Managing large-scale workflow execution from resource provisioning to provenance tracking: The cybershake example. In *e-Science and Grid Computing, 2006. e-Science'06. Second IEEE International Conference on*, pages 14–14. IEEE, 2006.

[6] J Octavio Gutierrez-Garcia and Kwang Mong Sim. Agent-based cloud workflow execution. *Integrated Computer-Aided Engineering*, 19(1):39–56, 2012.

[7] Angela O'Brien, Steven Newhouse, and John Darlington. Mapping of scientific workflow within the e-protein project to distributed resources. In *UK e-Science All Hands Meeting*, pages 404–409, 2004.

[8] Wil MP van Der Aalst, Arthur HM Ter Hofstede, Bartek Kiepuszewski, and Alistair P Barros. Workflow patterns. *Distributed and parallel databases*, 14(1):5–51, 2003.