Pograмación Paralela y Distribuida

Licenciatura en Ciencias de la Computación

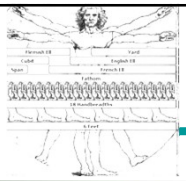
UNCuyo – Facultad de Ingeniería

Ministerio de Educación, Ciencia y Tecnología
Universidad Tecnológica Nacional
Facultad Regional Mendoza

LICPaD
Laboratorio de Investigación en Cómputo Paralelo/Distribuido

UNCUYO UNIVERSIDAD NACIONAL DE CUYO | FACULTAD DE INGENIERÍA

Early Adopters Awarded
Fall 2011 (NSF/IEEE)



Programación Paralela y Distribuida

Evaluación y Rendimiento de las Aplicaciones Paralelas

Unidad 4

Ministerio de Educación, Ciencia y Tecnología
Universidad Tecnológica Nacional
Facultad Regional Mendoza

LICPaD
Laboratorio de Investigación en Cómputo Paralelo/Distribuido

UNCUYO UNIVERSIDAD NACIONAL DE CUYO | FACULTAD DE INGENIERÍA

Contenido

- Introducción
- Speedup
- Eficiencia
- Balanceo de Carga
- Uso de los índices
- Análisis y Sintonización

Contenido

- Introducción
- Speedup
- Eficiencia
- Balanceo de Carga
- Uso de los índices
- Análisis y Sintonización

Introducción

- Diseño Paralelo
  - Las aplicaciones deben diseñarse de acuerdo al paradigma paralelo
    - Modelos de algoritmos paralelos
    - Librerías de comunicación
      (Dado que por lo general trabajaremos sobre sistemas distribuidos)
    - Aspectos adicionales:
      - Descomposición, asignación, granularidad, grado de concurrencia, balanceo de carga, escalabilidad, etc.
- Alto rendimiento
  - Las aplicaciones deben optimizarse para proveer un comportamiento eficiente
  - El comportamiento de la aplicación puede depender del conjunto de datos de entrada o del ambiente de ejecución

Introducción

- Depuración y Optimización
  - Una vez que un programa paralelo ha sido diseñado, implementado y testeado, debe evaluarse la calidad de su ejecución
  - Si el rendimiento obtenido está por debajo del esperable y/o el aceptable, la paralelización de la solución habrá sido en vano

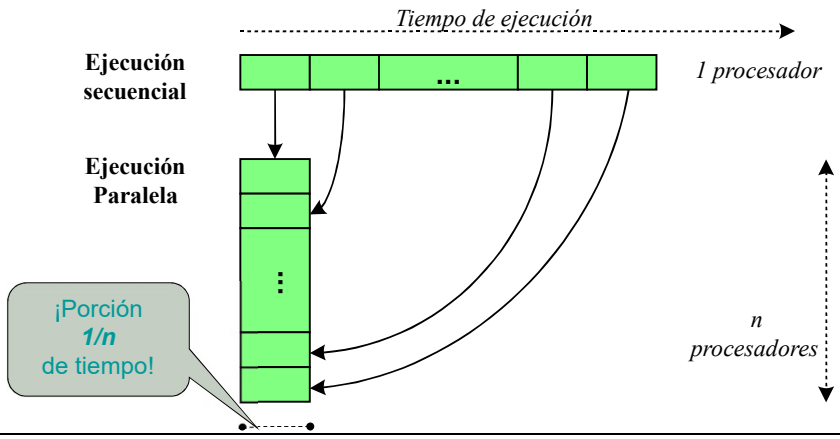
Introducción

- La idea general y básica del cómputo paralelo es que ***n*** procesadores o nodos deberían proveer una velocidad computacional ***n*** veces mayor a la provista por un nodo simple, es decir:

El problema debería ser
resuelto en un período
1/n
del tiempo insumido por un uniprocesador

Introducción

Gráficamente:



Introducción

- Sin embargo, ésta es una situación ideal que en la práctica no siempre se cumple.
  - Límites de rendimiento teóricos o prácticos
    - No obstante son superiores a los límites de los uniprocesadores.
    - Por ello, constituyen una plataforma ideal para aplicaciones que requieren altas prestaciones.
- Factores que limitan a las aplicaciones paralelas:
  - **Períodos** en los que algunos procesadores permanecen **ociosos**
  - **Cálculos redundantes** en cada nodo (recálculo de constantes, por ejemplo)
  - **Comunicación** entre procesos
  - Regiones del programa no paralelizables, como **inicialización** y **finalización** de la aplicación.

Introducción

¿Cómo diagnosticar los problemas de rendimiento que presentan las aplicaciones paralelas?

- Con la interpretación de los diferentes **índices de rendimiento**

Introducción

- A través de los años se ha definido un conjunto de índices para medir la “bondad” de las soluciones paralelas
- Sin embargo, aisladamente ninguno de ellos puede proporcionar una medida completamente fiel del rendimiento real del sistema
- Por ello deben considerarse distintos índices para medir diferentes aspectos

Introducción

- **Parámetros de interés**
  - Tiempo de ejecución
  - Escalabilidad
  - Eficiencia
  - Requerimientos de memoria
  - Throughput
  - Latencia de la red
  - Índices de entrada/salida
  - Throughput de red
  - Costos de diseño
  - Costos de implementación
  - Costos de verificación
  - Reusabilidad
  - Requerimientos de hardware
  - Portabilidad
  - Costos de mantenimiento

Introducción

- La importancia de cada factor es relativa a la naturaleza del problema
- Sin embargo, algunos parámetros son generales
  - Modelos matemáticos que formalizan cualidades
- Permiten una comprensión básica del comportamiento general de las aplicaciones
- Permiten comparar entre ejecuciones de programas o diferentes implementaciones del mismo programa

Introducción

- Índices de rendimiento de mayor interés
  - Speedup
  - Eficiencia
  - Balanceo de Carga

- Los estudiaremos a continuación…

Contenido

- Introducción
- Speedup
- Eficiencia
- Balanceo de Carga
- Uso de los índices
- Análisis y Sintonización

Speedup

- Interés en el desarrollo de soluciones paralelas:
  - Velocidad con la que se resuelve el problema
- Considerando el tiempo insumido en un único procesador podría ponderarse el tiempo insumido en un sistema paralelo

Speedup

- El Speedup se define de la siguiente manera:

$$Speedup(n) = \frac{Tiempo\ de\ ejecucion\ utilizando\ un\ unico\ procesador}{Tiempo\ de\ ejecucion\ utilizando\ n\ nodos\ en\ paralelo}$$

- Sea $T(x)$ el tiempo de ejecución de una aplicación con x procesadores:

$$Speedup(n) = \frac{T(1)}{T(n)}$$

- $Speedup(n)$ representa el incremento de velocidad con la utilización del sistema paralelo

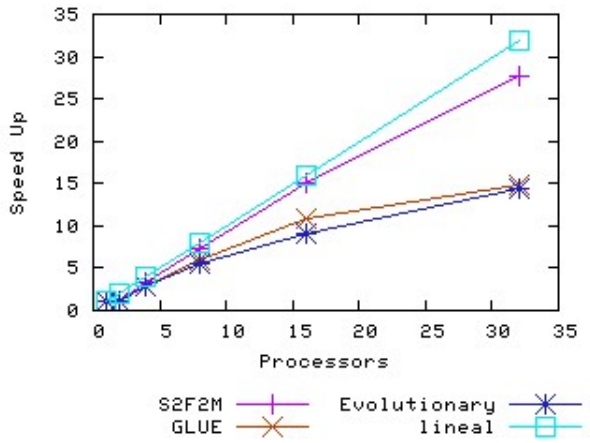
Speedup

- Speedup Máximo
  - El máximo speedup posible con n procesadores normalmente es n.
  - **"Speedup Lineal"**

$$Speedup(n) \leq \frac{T(1)}{\frac{T(1)}{n}} = n$$

  - Condiciones:
    - El cómputo puede dividirse en tareas de igual duración asignadas a procesadores diferentes sin overhead adicional en la solución paralela.

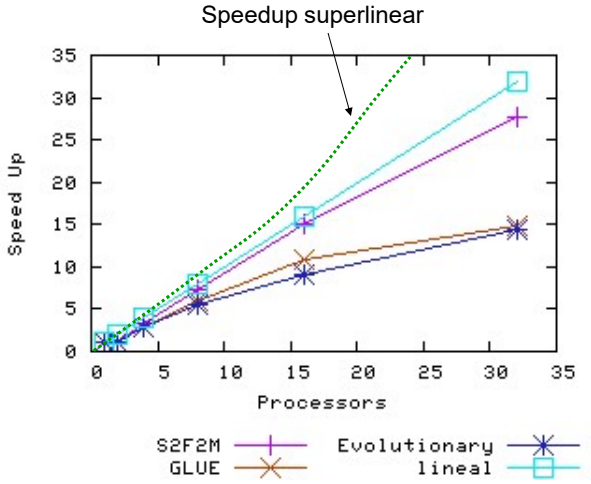
Speedup



Speedup

- Speedup Superlinear
  - El *Speedup*(*n*) puede alcanzar valores mayores a *n*.
    - Condiciones:
      - Existencia de memoria extra en el sistema paralelo.
    - ¡Cuidado!
      - T(1) debe ser el tiempo de la aplicación secuencial, es decir no el tiempo paralelo en 1 máquina.

Speedup



Speedup

- En resumen…
  - ¿Qué permite medir el Speedup?
    - La ganancia de incrementar la cantidad de recursos involucrados en el cómputo
  - ¿Cuál es un “buen Speedup”?
    - Cuanto más cercano se encuentre al Speedup lineal, mejor será el Speedup.
    - También debe considerarse el Speedup al crecer *n*, si es escalable

Speedup – Ley de Amdahl (1967)

- Factores que limitan el speedup:
  - Períodos ociosos
  - Cálculos redundantes
  - Comunicaciones
  - Cálculos meramente secuenciales

Speedup – Ley de Amdahl (1967)

- Sean
  - f la porción de tiempo requerido para procesar la región secuencial del programa
  - $T(1)$ la porción de tiempo de ejecución secuencial de la aplicación, es decir en un solo procesador
- El tiempo de ejecución con n nodos en paralelo será:

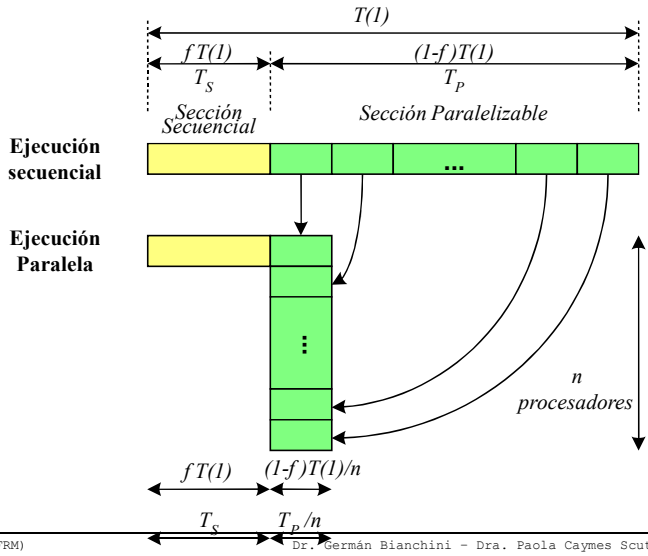
$$f * T(1) + (1 - f)\frac{T(1)}{n}$$

- Más intuitivamente…
  - T_S denota la región secuencial
  - T_P denota la región paralelizable
  - Y el tiempo de ejecución con n nodos será T_S + T_P

24

Speedup – Ley de Amdahl (1967)

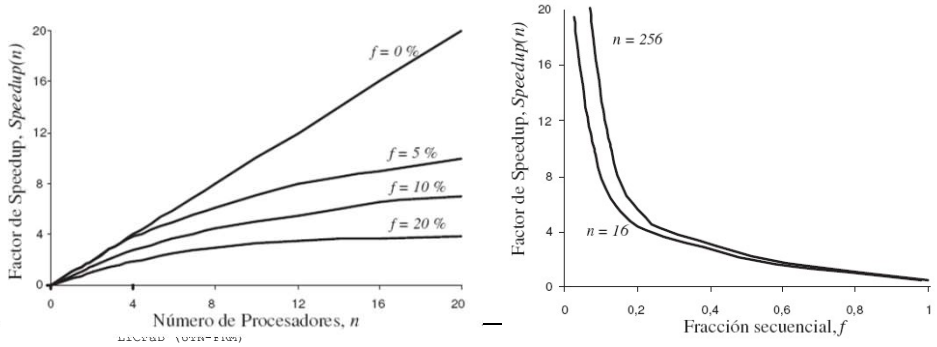
Gráficamente:



Speedup – Ley de Amdahl (1967)

- Ley de Amdahl

$$Speedup(n) = \frac{T_S + T_P}{T_S + \frac{T_P}{n}} = \frac{T(1)}{T_S + \frac{T_P}{n}} \leq \frac{T(1)}{T_S}$$



Speedup – Ley de Amdahl (1967)

- La fracción de cómputo ejecutada por procesos concurrentes debe ser una fracción sustancial del cómputo global para obtener un incremento de velocidad.
- El máximo speedup está limitado por 1/*f* :

$$\lim_{n \to \infty} Speedup(n) = \frac{1}{f}$$

Speedup - Ley de Gustafson-Barsis (1988)

- Amdahl propone un volumen de cálculo fijo, pero Gustafson plantea que el volumen del problema no es independiente del número de procesadores.
- Cuando el volumen del problema crece, lo hace "sólo" en su parte paralela, no en la secuencial. Por tanto, el cuello de botella tiende a cero.
- La ley de **Amdahl** se refiere a **procesos** con volumen de **cálculo fijo**, mientras que **Gustafson** se refiere a problemas cuyo volumen de **cálculo** puede **aumentar según** el número de **procesadores**.

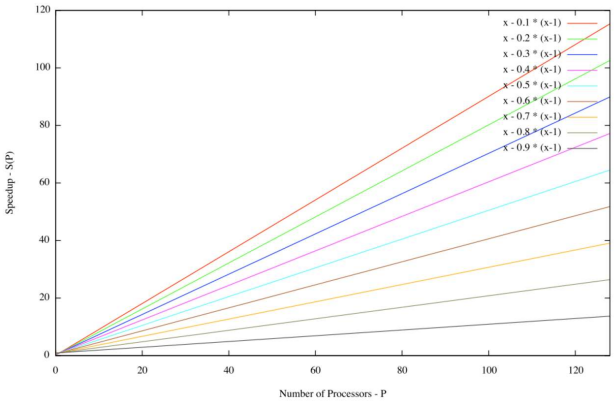
Speedup - Ley de Gustafson-Barsis (1988)

- Considerando la restricción del tiempo de ejecución paralela constante, el speedup será diferente numéricamente del speedup de Amdahl, y se denomina **Scaled Speedup** (speedup cuando el problema se escala)

$$Speedup_S(n) = n+(1-n)*f*T(1)$$

Siendo *n* el número de procesadores, $Speedup_S$ el scaled speedup y *f*T(1)* la parte no paralelizable del problema.

Speedup - Ley de Gustafson-Barsis (1988)



$$Speedup_S(n) = n+(1-n)*f*T(1)$$

LICPaD (UTN-FRM) Dr. Germán Bianchini – Dra. Paola Caymes Scutari

Contenido

- Introducción
- Speedup
- Eficiencia
- Balanceo de Carga
- Uso de los índices
- Análisis y Sintonización

Eficiencia

- En computación paralela, dados los costos, es crucial estudiar el uso que se ha hecho de los procesadores
- Puede determinarse a través de la eficiencia del sistema definida como:

$$Eficiencia = \frac{Tiempo\ de\ ejecucion\ utilizando\ un\ unico\ procesador}{Tiempo\ de\ ejecucion\ utilizando\ n\ nodos\ en\ paralelo \times n}$$

$$Eficiencia = \frac{T(1)}{T(n) \times n}$$

Eficiencia

- Eficiencia definida en función de Speedup:

$$Eficiencia = \frac{Speedup(n)}{n} \times 100\,\%$$

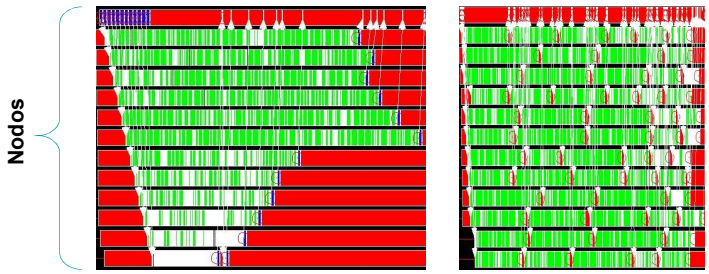
  - donde la eficiencia representa un porcentaje
  - Ejemplo:
    - los procesadores se utilizaron, en promedio, durante la mitad del tiempo
    - → la eficiencia es del 50 %

- La eficiencia alcanza el 100% cuando con *n* procesadores el *Speedup*(*n*) es *n*.

Contenido

Balanceo de Carga

Si cada nodo procesa una cantidad de tareas proporcional a su velocidad o disponibilidad, el sistema global se aprovecha mejor, ya que no hay nodos ociosos ni sobrecargados.



Ejecución desbalanceada Ejecución balanceada

Balanceo de Carga

- El balanceo/desbalanceo de carga puede estudiarse considerando los tiempos de cómputo y de ocio de los nodos involucrados
- En caso de observarse una ejecución desbalanceada, debe revisarse o ajustarse la técnica de descomposición y asignación utilizada

Balanceo de Carga

- Fuentes de overhead:
  - Interacciones entre procesos
  - Períodos ociosos
- Una asignación eficiente de tareas a procesadores debería considerar estas cuestiones para minimizar el efecto negativo que poseen.
- Sin embargo, la minimización de comunicaciones y/o períodos ociosos suele ir en detrimento de otras propiedades:
  - Agrupar tareas interdependientes suele desbalancear la carga
- Como consecuencia, la asignación de tareas y el balanceo de carga no resultan triviales

Balanceo de Carga

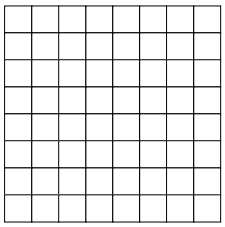
- Una ejecución balanceada depende de:
  - La división de tareas
  - Las características de las tareas
  - Las interacciones
  - El modelo de algoritmo paralelo
  - El tipo de asignación
    - Las técnicas de asignación más utilizadas para lograr balanceo de carga son:
      - Asignación Estática
        - Asignación basada en Partición de Dominio
      - Asignación Dinámica
        - Esquema centralizado
        - Esquema distribuido

Balanceo de Carga

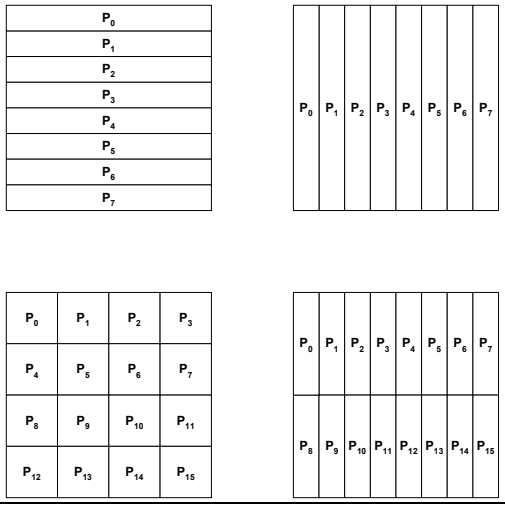
- Asignación Estática
  - **Partición de Dominio por Bloques**
    - Asignación de porciones contiguas uniformes a diferentes procesos
    - Dado un arreglo *d*-dimensional, se distribuye entre los procesos de manera tal que cada proceso recibe un bloque contiguo de entradas, un subconjunto del arreglo.
    - *Aplicabilidad*: localidad de interacción, i.e. el cálculo de cada elemento a lo sumo requiere los valores de los elementos vecinos

Balanceo de Carga

Matriz



Puede particionarse de diferentes maneras

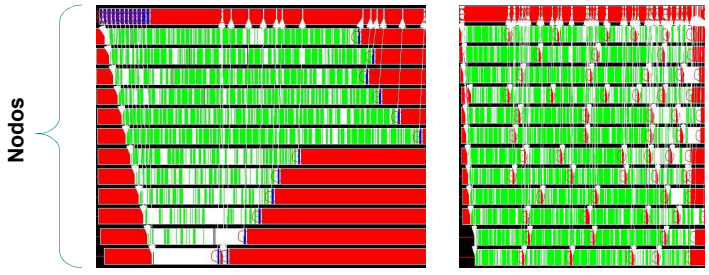


Balanceo de Carga

- Asignación Dinámica
  - **Esquema Centralizado**
    - Todas las tareas son mantenidas por un proceso especial (o subconjunto de procesos)
    - Para la asignación de las tareas normalmente se sigue un esquema Master/Worker, donde el Master es el proceso que administra el conjunto de tareas, y los Workers son los procesos que dependen del Master para obtener las tareas
      - Asignación completa
      - Asignación bajo petición

Balanceo de Carga

Si cada nodo procesa una cantidad de tareas proporcional a su velocidad o disponibilidad, el sistema global se aprovecha mejor, ya que no hay nodos ociosos ni sobrecargados.



Ejecución desbalanceada **Ejecución balanceada**

Balanceo de Carga

- Asignación Dinámica
  - **Esquema Distribuido**
    - El conjunto de tareas se distribuye entre los procesos
    - Los procesos intercambian tareas dinámicamente para balancear la ejecución
    - Supera el bottleneck del esquema centralizado de asignación de tareas
    - Parámetros críticos:
      - Coordinación/identificación de procesos intervinientes en un intercambio
      - Volumen del trabajo transferido
        - < volumen → > interacciones para completar datos
        - > volumen → el emisor puede quedar ocioso
      - Instante en que se realiza la transferencia
        - Cuando el receptor está ocioso
        - Cuando el receptor está por terminar su trabajo previo. Se anticipa.

Contenido

- Introducción
- Speedup
- Eficiencia
- Balanceo de Carga
- Uso de los índices
- Análisis y Sintonización

Uso de los índices

- Los índices de rendimiento permiten ponderar la calidad de diversas características de un programa

¿Cómo utilizarlos?

Uso de los índices

- La evaluación de un algoritmo paralelo se realiza en dos etapas:
  - ***A priori***: comprende el estudio teórico
    - Permite identificar posibles errores o elementos mejorables en la implementación (aspectos inherentes)
  - ***A posteriori***:
    - Comprende el estudio experimental
      - Permite interpretar los datos considerados en la evaluación y contrastar con los resultados esperados de acuerdo al estudio teórico
    - Permite realimentar el análisis teórico
      - (Es decir, *"a priori"* para ejecuciones siguientes)

Uso de los índices

- Estudio experimental
  - Comprende varios pasos:
    1. Diseño de los experimentos
    2. Determinar los factores que pueden influir en los resultados
    3. Determinar las métricas que deben obtenerse
    4. Ejecución de los experimentos
    5. Interpretación de los resultados

Uso de los índices

- Estudio experimental
  1. **Diseño de los experimentos**
     - Definir el rango
       - Número de procesadores
       - Tamaño del problema
     - Determinar qué comportamiento se espera
       - Resultados esperados (tipo, características, indicadores)
     - Comparar los resultados experimentales con los teóricos
       - Fuentes posibles de errores:
         - Estudio teórico
         - Implementación
         - Diseño o ejecución de los experimentos

Uso de los índices

- Estudio experimental
  2. **Determinar los factores que pueden influir en los resultados**
     - Características del conjunto de datos
       - Caso más favorable
       - Caso menos favorable
       - Caso promedio
     - Estado (carga) del sistema
     - Costo de acceso a los datos
       - *Swapping*: puede requerirse según el tamaño de lo datos
       - Al distribuir, la gestión de memoria mejora notablemente

Uso de los índices

- Estudio experimental
  3. **Determinar las métricas que deben obtenerse**
     - **Speedup y Eficiencia**
       - Tiempo de ejecución
       - Cantidad de nodos
       - Tamaño del problema
     - **Balanceo de Carga**
       - Tiempo de cómputo de cada nodo
       - Tiempo ocioso de cada nodo
       - Tiempo de comunicación
     - **Escalabilidad**
       - Es fundamental la realización de experimentos de diferentes tamaños
       - La analizaremos en función de:
         - Speedup, eficiencia y balanceo de carga
         - Características algorítmicas de la aplicación

Uso de los índices

- Estudio experimental
  4. **Ejecución de los experimentos**
     - Recolección de datos de ejecución
  5. **Interpretación de los resultados**
     - Los datos recolectados se utilizan para interpretar las expresiones de rendimiento
     - Los datos se organizan en tablas que consideren:
       - Tamaño de los datos
       - Cantidad de nodos
       - Tiempo de ejecución
     - Y se grafican
       - Para apreciar comportamientos normales
       - O detectar anomalías que hay que explicar y/o corregir

Uso de los índices

- Estudio experimental
  - Experimentos complementarios
    - Individualizar partes más costosas
      - Tomar tiempos en distintas partes del programa
      - Mejorar la algorítmica y programación
    - Por ejemplo, pueden considerarse regiones
      - Cómputo
      - Comunicaciones
      - Y contrastarlas con su tiempo conjunto
    - Para detectar problemas de
      - Sincronización o desequilibrio
      - Solapamiento entre cómputo y comunicación
      - Desbalanceo de carga
      - Desequilibrio entre diversas partes del cómputo

Contenido

- Introducción
- Speedup
- Eficiencia
- Balanceo de Carga
- Uso de los índices
- Análisis y Sintonización

Análisis y Sintonización

Rendimiento de las aplicaciones paralelas

- Indices para evaluar el comportamiento:
  - Speedup
  - Eficiencia
  - Balanceo de carga

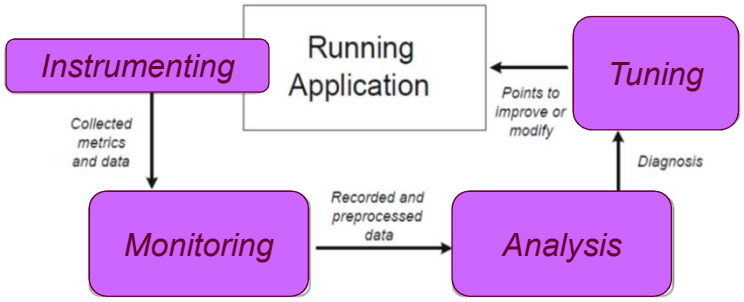
  } Ninguno provee información específica o sugerencias para solucionar los problemas de rendimiento

Análisis y Sintonización de Rendimiento

- El **Proceso de Sintonización** comprende una serie de pasos:
  - Monitorización
  - Análisis
  - Sintonización

Análisis y Sintonización

- Proceso de Sintonización

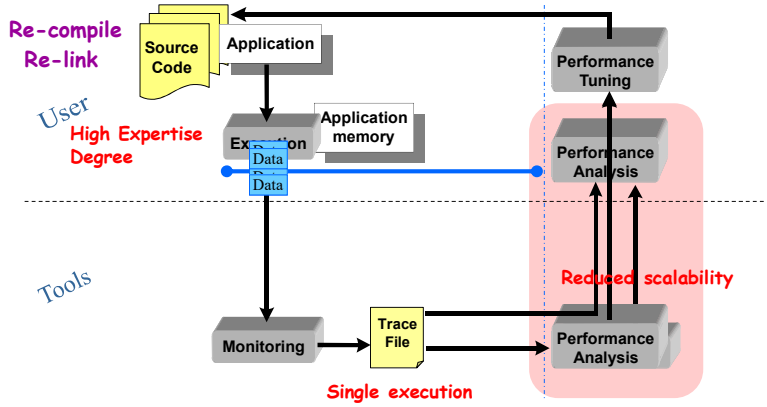


Análisis y Sintonización

- Clasificación de Herramientas de Performance
  - Herramientas de Monitorización
    - Tecnología para instrumentar la aplicación
    - Conjunto de herramientas para recolectar y procesar los datos
    - Intrusión
  - Herramientas de Análisis
    - Automatización de la evaluación de los datos monitoreados
    - Conocimiento de rendimiento
  - Herramientas de Sintonización
    - Introducir modificaciones en la aplicación
- Aproximaciones de Monitorización, Análisis y Sintonización
  - Análisis Clásico de Rendimiento
  - Análisis Automático de Rendimiento
  - Análisis Dinámico de Rendimiento
  - Sintonización Dinámica del Rendimiento

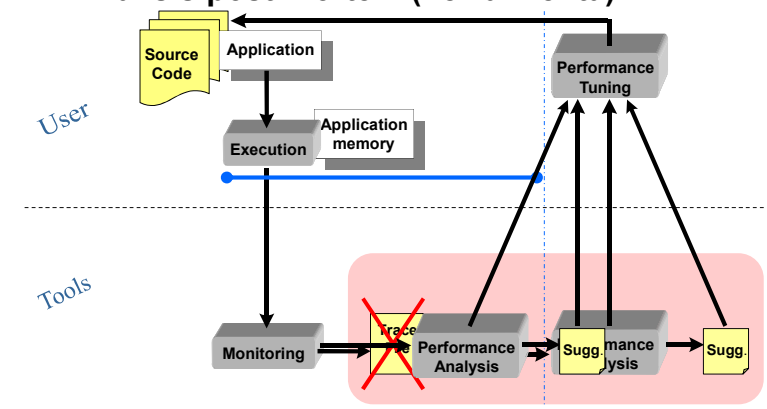
Análisis y Sintonización

- **Análisis Clásico** de Rendimiento
  - **Análisis post-mortem (usuario)**



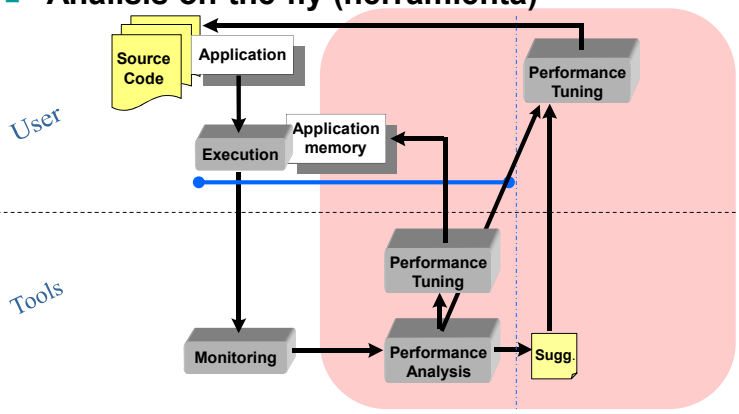
Análisis y Sintonización

- **Análisis Automático** de Rendimiento
  - **Análisis post-mortem (herramienta)**



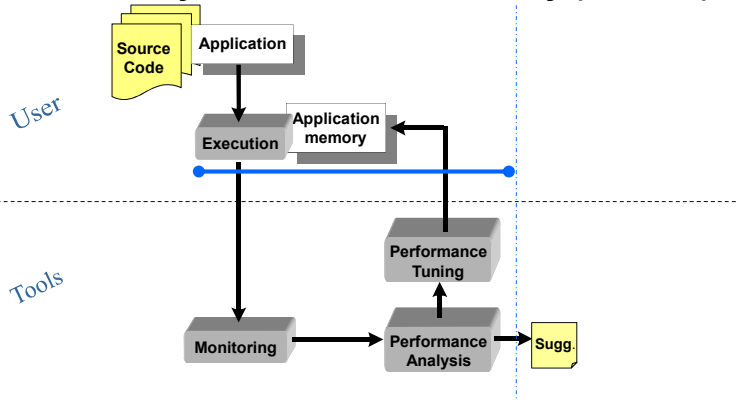
Análisis y Sintonización

- **Análisis Dinámico** de Rendimiento
  - **Análisis on-the-fly (herramienta)**



Análisis y Sintonización

- **Sintonización Dinámica** de Rendimiento
  - **Análisis y sintonización on-the-fly (herram.)**



Análisis y Sintonización

Aproximación	Herramientas
Análisis Clásico	Tape/PVM, ParaGraph, Vampir, Damien, Pablo, XPVM
Análisis Automático	KappaPi, Paradise, AIMS
Análisis Dinámico	Paradyn
Sintonización DInámica	Autopilot, Active Harmony, AppLeS, PerCo, Mojo, MATE

Resumen

- Índices de Rendimiento
  - Necesidad
  - Utilidad
  - Aplicabilidad
  - Speedup
  - Eficiencia
  - Balanceo de Carga
- Utilidad y utilización de los índices de rendimiento
- Restricciones de los índices de rendimiento
- Aproximaciones para el análisis y la sintonización de aplicaciones paralelas
- Herramientas:
  - Análisis
  - Sintonización

Tarea para la casa…

- **Repasar los conceptos presentados**
- **Resolver los ejercicios de la Práctica relacionados con este tema**

Bibliografía

- **Almeida D., Giménez D., Mantas J., Vidal A.,** *"Introducción a la Programación Paralela".* Paraninfo Cengage Learning. 2008.
- **Bianchini G,** "*Wildland Fire Prediction Based on Statitical Analysis of Multiple Solutions*". Tesis de Doctorado. Universitat Autònoma Barcelona. 2006.
- **Buyya, R. et al**, "*High Performance Cluster Computing – Architectures and Systems (Volume 1)"*, Prentice Hall, 1999.
- **Dongarra J., Foster I., Fox G., Gropp W., Kennedy K., Torczon L., White A.,** "*Sourcebook of parallel computing*". Morgan Kaufmann Publishers. 2003.
- **Grama A., Gupta A., Karypis G., Kumar V.,** "*Introduction to Parallel Computing*". Pearson Addison Wesley. Second Edition. 2003.
- **Mattson, T., Sanders, B., Massingill, B.**, "*Patterns for Parallel Programming*", Addison-Wesley, 2004.
- **Wilkinson B., Allen M.,** "*Parallel Programming - Techniques and Applications Using Networked Workstations and Parallel Computers*". Pearson Prentice Hall. Second Edition. 2005.

Bibliografía

- **Caymes Scutari, P.** *"Extending the Usability of a Dynamic Tuning Environment".* Tesis de Doctorado. Universitat Autònoma Barcelona. 2007.
- **Espinosa, A., Margalef, T., Luque, E.,** "*Automatic Detection of Parallel Program Performance Problems*". Lecture Notes in Computer Science, vol. 1573, pp. 365-377, Springer-Verlag. junio de 1998.
- **Krishnan, S., Kale, L. V.**, "*Automating Parallel Runtime Optimizations Using Post-Mortem Analysis*". International Conference on Supercomputing, pp. 221-228. 1996.
- **Mayes, K.R., Lujan, M., Riley, G.D., Chin, J., Coveney, P.V. and Gurd, J.R.,** "*Towards Performance Control on the Grid".* Philosophical Transactions of the Royal Society of London Series A, Vol. 363, No. 1833, pp. 1793-1806, agosto de 2005.
- **Miller, B.P., Callaghan, M.D., Cargille, J.M., Hollingsworth, J.K., Irvin, R.B., Karavanic, K.L., Kunchithapadam, K., Newhall, T.,** "*The Paradyn Parallel Performance Measurement Tool".* IEEE Computer vol. 28. pp. 37-46. noviembre de 1995.
- **Morajko, A.** "*Dynamic Tuning of Parallel/Distributed Applications".* Tesis de Doctorado. Universitat Autònoma Barcelona. 2004.
- **Ribrel, R.L., Vetter, J.S., Simitci, H., Reed, D.A.**, "*Autopilot: Adaptive Control of Distributed Applications".* High Performance Distributed Computing 1998, pp. 172-179. Chicago, agosto de 1998.
- **Tapus, C., Chung, I-H., Hollingsworth, J.K.**, "*Active Harmony: Towards Automated Performance Tuning".* SC'02. noviembre de 2002.
- **Yan, J., Sarukhai, S.**, "*Analyzing Parallel Program Performance Using Normalized Performance Indices and Trace Transformation Techniques".* Parallel Computing, vol. 22, pp. 1215-1237. 1996