II CAIM 2010
Segundo Congreso Argentino de Ingeniería Mecánica
San Juan - Noviembre 2010

Desarrollo de una estrategia de paralelización explícita para el método de red de vórtices inestacionario y no lineal

Luis Ceballos, Adrián Barone, Ariel Flores, Sergio Preidikman

Facultad de Ingeniería, Universidad Nacional de Río Cuarto
Ruta Nacional 36, km 601, Río Cuarto, Córdoba – Argentina
Tel/Fax: +54-3584676246 - E-mail: lceballos@ing.unrc.edu.ar

RESUMEN

En este trabajo se presenta el desarrollo de una estrategia de paralelización explícita para aumentar la eficiencia computacional de códigos computacionales que lo implementan el método de red de vórtices inestacionario y no lineal (NUVLM). Para ayudar a comprender la estrategia presentada, en la primera parte del trabajo se describe sucintamente el NUVLM, se muestran ejemplos de los resultados obtenidos con una implementación computacional secuencial del NUVLM y se presenta un algoritmo del NUVLM.

La estrategia de paralelización está orientada a reducir los tiempos de ejecución de las partes del NUVLM que más tiempo consumen. Mediante mediciones de tiempos de ejecución se detecta que porciones del código computacional actúan como "cuello de botella". También, se estudia como inciden el número de elementos de la discretización elegida para las mallas y la cantidad de pasos de la simulación sobre el porcentaje del tiempo total que insumen las distintas etapas de la implementación del NUVLM.

La estrategia de paralelización propuesta consiste en realizar una descomposición del dominio centrada en los datos de entrada sobre un modelo de arquitectura de memoria compartida. La implementación computacional se realizó con un código escrito en Fortran 90 que utiliza una biblioteca con el estándar de OpenMP.

En la sección de resultados se muestran mediciones de tiempo realizadas sobre códigos computacionales secuenciales. Las mediciones realizadas demuestran que la etapa de convección para generar las estelas es crítica porque consume el más alto porcentaje del tiempo total de ejecución. También se presentan resultados de mediciones de tiempos realizadas sobre ejecuciones de un programa paralelizado que utiliza varios hilos de ejecución. Estos resultados muestran importantes incrementos en la velocidad de ejecución. Por último, se presenta un análisis de performance y se muestran curvas de "speedup" y eficiencia obtenidas experimentalmente. Para el trazado de las curvas se realizaron ejecuciones del código paralelo utilizando una configuración de vehículo aéreo no tripulado de alas unidas. Las ejecuciones se realizaron utilizando una computadora de escritorio con un procesador con cuatro núcleos, y en todos los casos se demostró una muy buena eficiencia del código computacional que ha sido paralelizado.

Palabras Claves: Computación de alto desempeño, Memoria compartida, OpenMP, Aerodinámica, Método de red de vórtices inestacionario y no-lineal.

1. INTRODUCCIÓN

El trabajo presentado forma parte de un esfuerzo mayor orientado a desarrollar herramientas numéricas de alta fidelidad para estudiar problemas aeroservoelásticos inestacionarios y fuertemente no-lineales [1]-[5]. Para que las herramientas en desarrollo produzcan resultados confiables deben necesariamente emplear un modelo aeroservoelástico, y para ello deben incorporar un modelo aerodinámico, un modelo estructural y un sistema de control trabajando juntos como un único sistema dinámico. En un esfuerzo orientado a desarrollar herramientas computacionales del tipo mencionado se están siguiendo los lineamientos de Preidikman [6]. Integrantes del grupo al que pertenecen los autores de este trabajo han previamente desarrollado algunas herramientas computacionales con un modelo aerodinámico que implementa el método de red de vórtices inestacionario y no lineal (*non-linear unsteady vortex lattice method* o NUVLM).

En lo que respecta a los requerimientos computacionales, la implementación del NUVLM posee varias ventajas respecto de otros métodos numéricos basados en técnicas propias de la Mecánica de Fluidos Computacional (por ejemplo: elementos finitos, volúmenes finitos, etc.). No obstante, implementaciones "seriales" del NUVLM (comportamiento aerodinámico de vehículos aéreos con configuraciones de alas fijas, con alas que mutan, o con alas batientes) demostraron que el NUVLM requiere recursos computacionales altos. El uso de técnicas tales como recortar las estelas o aprovechar la simetría del problema permiten que las implementaciones computacionales del NUVLM puedan ejecutarse más rápido, a costo de restringir el tipo de problemas que pueden ser atacados [7], [8]. Por esta razón, el camino natural para lograr mayores velocidades de ejecución es desarrollar códigos computacionales que implementen el NUVLM utilizando técnicas de computación de alto desempeño (*High Performance Computing* o HPC).

Fritz y Long [9] presentaron herramientas computacionales que implementan el NUVLM usando técnicas de computación en paralelo sólo sobre el método utilizado para resolver sistemas de ecuaciones algebraicas lineales. En el presente trabajo se analizan que porciones del algoritmo NUVLM consumen más tiempo de ejecución y que partes son "naturalmente" paralelizables. Se describe, también, una estrategia de paralelización que permiten atacar las porciones más conflictivas o "cuellos de botella". La estrategia de paralelización se desarrolla de manera general y está pensada para ser aplicadas a casos en los que existen uno o varios cuerpos sumergidos en el seno de un fluido. Adicionalmente se muestran resultados de la implementación de una de las estrategias, utilizando una arquitectura de memoria compartida, OpenMP, y Fortran.

2. EL MÉTODO DE RED DE VÓRTICES

El modelo aerodinámico basado en el NUVLM permite modelar correctamente no-linealidades aerodinámicas asociadas con grandes ángulos de ataque, deformaciones estáticas, y flujos dominados por vorticidad en los que el fenómeno conocido como *vortex bursting* no ocurre. El modelo predice correctamente la emisión de vorticidad desde un cuerpo (o varios), inmerso en el seno de un fluido, hacia el campo del flujo. Esta vorticidad es transportada por el flujo de aire desde el cuerpo hacia el fluido y forma así las estelas. La distribución de la vorticidad en las estelas y la forma de las mismas son, también, parte de la solución del problema. El NUVLM es un método confiable y muy buen predictor de las cargas aerodinámicas inestacionarias y no-lineales.

En flujos sobre superficies sólidas donde el número de Reynolds es alto, se genera vorticidad por efectos viscosos en capas muy delgadas, llamadas capas límites, que están pegadas a las superficies del cuerpo. Los efectos viscosos son responsables de la existencia de las capas límites. Parte de esta vorticidad es emitida desde los bordes filosos de los sólidos y transportada por el fluido, formando las estelas. El campo de velocidades asociado con toda esta vorticidad interactúa con la llamada corriente libre: mientras las condiciones de borde de no-penetración y no-deslizamiento son satisfechas sobre las superficies sólidas generadoras de vorticidad, la vorticidad en las estelas se mueve libremente en el fluido de forma tal que no se produzcan saltos de presión a través de las estelas.

El método de red de vórtices inestacionario esta basado en la idea de representar las capas límites y las estelas mediante sábanas vorticosas. Nos referiremos a estos dos tipos de sábanas vorticosas como "sábanas adheridas" (*bound-vortex sheets*) y "sábanas libres" (*free-vortex sheets*).

El flujo asociado con la vorticidad en la estela cercana al sólido afecta el flujo alrededor del mismo sólido y por lo tanto las cargas actuantes sobre ella. Debido a que la vorticidad presente en las estelas en un instante dado fue generada y convectada desde el ala en un tiempo anterior, las cargas aerodinámicas dependen de la historia del movimiento; las estelas contienen la "historia". El campo de velocidades, asociado con la vorticidad existente en un punto del espacio, decae con la distancia a dicho punto; en consecuencia, a medida que la vorticidad en la estela va siendo transportada flujo abajo, su influencia decrece y por lo tanto se dice que "el historiador" va perdiendo memoria. Más detalles acerca de los fundamentos matemáticos y de la implementación numérica del NUVLM pueden consultarse en las referencias [2],[6],[10].

2.1. Implementación computacional

En el NUVLM se reemplaza la sábana vorticosa adherida a la superficie del sólido y la sábana vorticosa libre por una red de segmentos vorticosos rectos y de longitud finita. La red de segmentos vorticosos de la sábana adherida divide la superficie del sólido en elementos de área que tienen forma de rectángulo, de paralelogramo, o en el caso más general de un cuadrilátero con todos sus lados diferentes.

La implementación numérica de este modelo requiere que la geometría del cuerpo se defina y de una manera particular: las distintas partes componentes del cuerpo son representadas mediante un conjunto de superficies. Estas superficies se definen mediante mallas formadas por paneles y nudos. En la Figura 1 se presenta una malla que define una geometría muy simple referida a una placa plana y rectangular, mientras que en la Figura 2 se muestra una geometría más compleja que corresponde a un vehículo aéreo no tripulado con una configuración de alas unidas (JW UAV).

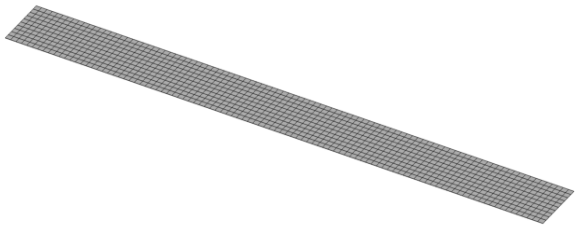

Figura 1 Malla de paneles de una geometría simple: placa rectangular

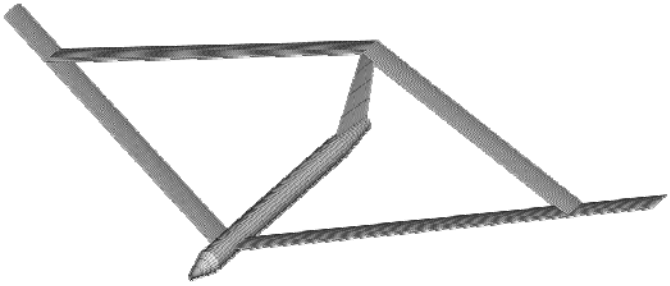

Figura 2 Malla de paneles generada para una configuración de UAV con alas unidas [1]

En cada uno de los paneles se define un punto de control que ese encuentra en el centroide de área del elemento. En el punto de control se impone la condición de no penetración en su versión discretizada. Esta imposición resulta luego en un sistema de ecuaciones algebraico lineal. La resolución de este sistema de ecuaciones permite obtener el valor de circulación que esta asociado a cada uno de los segmentos vorticosos utilizados para describir aerodinámicamente la superficie de la geometría.

En una última etapa se calculan las cargas aerodinámicas sobre la/s superficie/s sustentadora/s. Para cada elemento, se debe hallar primero la presión en el punto de control y luego multiplicarla por el área del elemento y por el vector unitario normal. La versión inestacionaria de la ecuación de Bernoulli se usa para calcular la distribución de la presión sobre la superficie de las alas. Finalmente, se suman las fuerzas y los momentos de dichas fuerzas actuantes en todos los elementos.

2.2. Generación de estelas

En esta subsección se muestra un ejemplo de generación de estelas para el caso de un ala que comienza a moverse impulsivamente. Para generar las estelas, hay que realizar una convección de las partículas de fluido que se encuentran sobre los bordes filosos del ala: borde de fuga y punteras. La convección se hace calculando el desplazamiento de las partículas que se produce en un intervalo de tiempo Δt.

En la Figura 3 se muestra un esquema de la evolución de las estelas para los primeros dos pasos de tiempo de una simulación. El ejemplo mostrado en la figura, corresponde al de un ala rectangular discretizada con 40 paneles (4 paneles a lo largo de la cuerda por 10 paneles a lo largo de la envergadura). En la Figura 3a se muestra el instante inicial t = 0, en el cual no existe la estela. En esa figura se destacan con puntos negros las partículas que están sobre los bordes filosos que emiten estela. A estas partículas se les realiza la convección durante el primer paso de tiempo, y como resultado se obtiene una fila de paneles que conforma la estela en el instante t = Δt (ver Figura 3b). Durante el segundo paso tiempo, se realiza el convección sobre las partículas que pertenecen a la estela y sobre las partículas que están sobre el borde de fuga y las punteras (partículas destacadas con puntos negros en la Figura 3b). El resultado al final del segundo paso de tiempo (t = 2Δt) está esquematizado en la Figura 3c, en la cual se muestran las dos filas de las estela obtenidas por "convección". Para continuar con los pasos siguientes, se procede de manera similar, esto es, realizando el proceso de convección sobre todas las partículas que pertenecen a la estela, y sobre las partículas que están sobre los bordes filosos que emiten estela.

En la Figura 4 se muestran ejemplos de estelas obtenidas con códigos desarrollados con anterioridad. En las Figura 4a se muestra la estela generada luego de 100 pasos de simulación para el caso una placa plana con un ángulo de ataque geométrico de 5 grados y un alargamiento con valor igual a 10. En la Figura 4b se muestra la estela generada luego de 100 pasos de simulación, para el caso de una configuración de JW UAV con un ángulo de ataque geométrico de 5 grados.

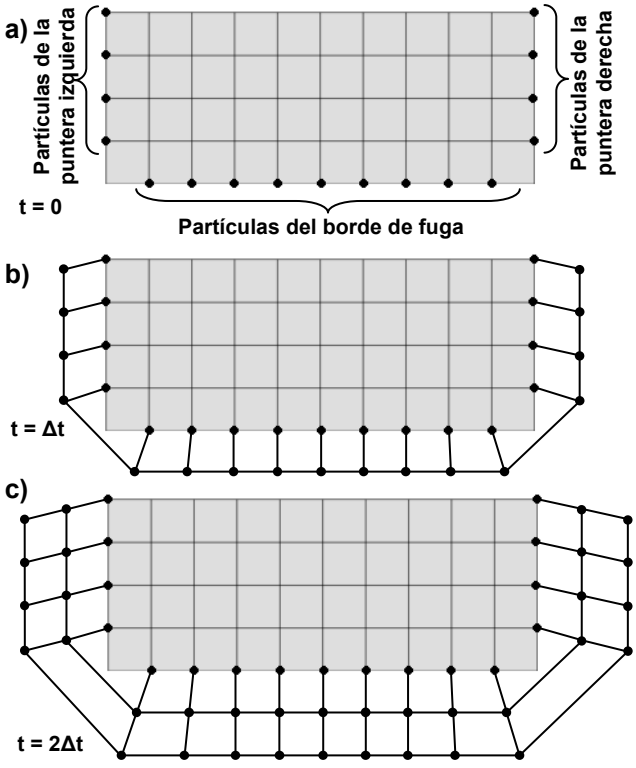

Figura 3 Esquema de la evolución de las estelas: a) t = 0, b) t = Δt y c) t = 2Δt

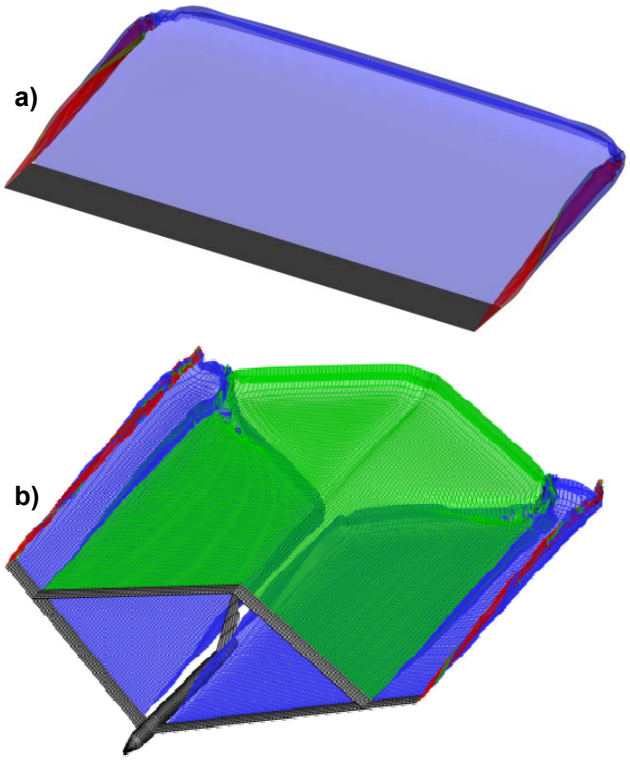

Figura 4 Ejemplos de estelas generadas para a) una geometría simple, y b) una configuración de JW UAV

2.3. Algoritmo del NUVLM

A continuación, se muestra un algoritmo del NUVLM que sido tomado como base para la implementación computacional paralelizada que se ha desarrollado en este trabajo:

1. *Lectura y ordenamiento de datos.*
2. *Cómputo de la matriz de coeficientes de influencia aerodinámicos.*
3. *Cómputo del lado derecho del sistema de ecuaciones.*
4. *Solución del sistema de ecuaciones: cálculo de la circulación.*
5. *Para cada paso de tiempo se realizan los siguientes pasos:*
   - 5.1. *Convección de las partículas de fluido.*
   - 5.2. *Cómputo del nuevo lado derecho del sistema de ecuaciones.*
   - 5.3. *Solución del nuevo sistema de ecuaciones.*
   - 5.4. *Cómputo de los coeficientes de presión y cargas.*
   - 5.5. *Almacenamiento de los resultados obtenidos en el paso de tiempo actual.*

3. ESTRATEGIA DE PARALELIZACION

En esta sección se presenta una estrategia que tienen como fin incrementar la velocidad de ejecución de un código computacional que implementa el NUVLM. La estrategia presentada consiste en una paralelización explícita del código computacional. La paralelización explícita se realiza sólo en aquellas porciones del código que consumen más tiempo durante su ejecución.

3.1. Identificación de los "cuellos de botella"

Experiencias de medición de tiempos de ejecución, realizadas con algunos códigos computacionales "secuenciales" que implementan el NUVLM, muestran que el mayor consumo de tiempo esta asociado al proceso de convección realizado para generar las estelas que se desprende desden desde los bordes filosos de un cuerpo sumergido en el seno de un fluido.

Las mediciones de tiempos de ejecución se realizaron mediante una técnica simple que, dadas las características que posee el NUVLM implementado, resulta efectiva a los fines de identificar los "cuellos de botella". La técnica de medición de tiempos consiste en adquirir la hora y fecha antes y después de cada una de las tareas listadas en el algoritmo presentado en la sección anterior, y luego por diferencia entre esas dos mediciones consecutivas, se obtiene el tiempo que ha consumido cada tarea.

En la sección de resultados se presentan mediciones de tiempos obtenidos de implementar la técnica antes descripta en un código que permiten simular casos como los mostrados en las Figura 4. Acumulando el tiempo empleado por las tareas repetitivas del algoritmo presentado en la subsección 2.3, se observa que el tiempo consumido por el proceso de convección de la estela supera ampliamente el tiempo empleado para ejecutar el resto de las tareas. También se concluye, de los casos presentados, que el porcentaje de tiempo total consumido por el proceso de convección de las estelas se ve poco influenciado por el tamaño de la malla elegida. Todos estos resultados confirman que, para disminuir los tiempos de ejecución, es necesario desarrollar una estrategia de paralelización explícita solamente en la etapa de convección.

3.2. Estrategia de paralelización

La estrategia consiste en realizar la convección simultánea de distintas partículas en un paso de tiempo de simulación determinado. Las partículas pertenecen a la estela que ha sido generada hasta el paso actual de la simulación. Debido a que es posible saber con anterioridad cual es la cantidad total de partículas a convectar, la distribución de tareas puede hacerse de forma equitativa entre los diferentes hilos de ejecución disponibles.

En la Figura 5 se muestra un esquema de la evolución de las estelas para los primeros dos pasos de tiempo de una simulación. En esta figura se ilustra la estrategia aplicada durante la convección de la estela generada por el borde de fuga del ala presentada en el ejemplo de la Figura 3, en la sección 2.2 de este trabajo. Esta estrategia es aplicable también a la generación de las estelas de las punteras, pero para lograr claridad en este ejemplo eso se ha omitido. También, a los fines de lograr simplicidad, en el ejemplo de la

Figura 5 se considera que la convección es realizada por solo dos hilos de ejecución, pero la idea de esta estrategia puede ser generalizada para ser aplicada utilizando n hilos de ejecución.

La coloración diferente de los puntos que están sobre el borde de fuga y sobre la estela generada representa la manera en que se divide entre los dos hilos de ejecución la tarea de realizar la convección.

En la Figura 5a se muestra el instante inicial t = 0, en el cual no existe desarrollo de la estela. En esa figura se destacan con puntos negros las partículas que están sobre las punteras del ala y, con puntos azules y rojos las partículas que están sobre el borde de fuga del ala. Debido a las características del NUVLM es posible "convectar" en simultáneo, mediante dos hilos de ejecución diferentes, las partículas destacadas con puntos de color azul y rojo.

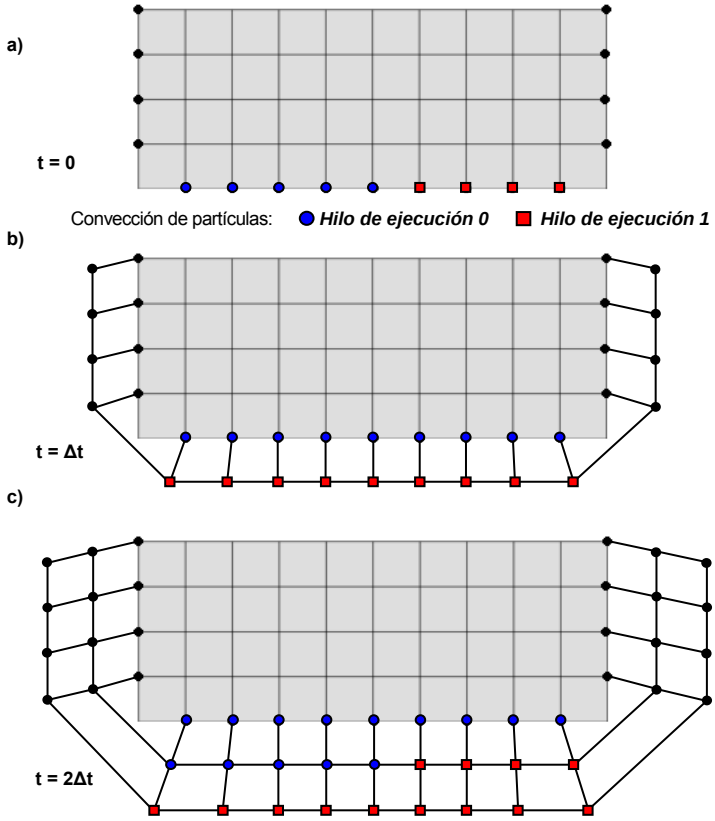

Figura 5: Esquema que ilustra la estrategia 4, para los pasos: a) t = 0, b) t = Δt y c) t = 2Δt.

Durante el primer paso de tiempo, y como resultado de la convección se obtiene una fila de paneles que conforma la estela en el instante t=Δt (Figura 5b). Es importante destacar que antes de pasar al siguiente paso de la simulación es necesario haber generado las estelas de todos los bordes filosos emisores existentes, esto es, punteras y borde de fuga. Notar que en este instante la cantidad de partículas a "convectar" ha cambiado respecto de la cantidad que había en el paso anterior, por lo que en este paso se plantea una nueva división de tareas.

La división se realiza repartiendo igual cantidad de partículas para que sean "convectadas" por cada hilo de ejecución. Esta división de tareas se remarca en la Figura 5b con puntos coloreados de color azul y rojo.

Luego de realizar la convección de las partículas, se obtiene el resultado del segundo paso de tiempo de simulación (t = 2Δt) que está esquematizado en la Figura 5c, en la cual se muestran las dos filas de la estela obtenida. En esa figura se presenta una nueva división de tareas entre los dos hilos de ejecución.

Para continuar con los pasos siguientes, se procede de manera similar, esto es, realizando el proceso de convección sobre todas las partículas que pertenecen a las estelas de las punteras, y dividiendo la cantidad de partículas en igual cantidad para que sea convectada en cada hilo de ejecución.

Si ocurriera que la división no sea entera, se asigna el resto de la división al primer hilo de ejecución. Como regla general, para el caso en que se disponen p hilos de ejecución y la división no es entera, se procede a repartir equitativamente el resto de la división entre los primeros hilos de ejecución.

4. RESULTADOS

En esta sección se presentan resultados de distintas mediciones de tiempos hechas sobre la versión secuencial y paralela del código desarrollado que implementa el NUVLM. Exceptuando los casos que en los que se haga una salvedad, todos los programas han sido escritos con Fortran 90 y compilados para ser ejecutados sobre un sistema operativo Gentoo Linux. La versión con paralalelización explícita fue desarrollada utilizando las bibliotecas de OpenMP que están embebidas en el compilador de Fortran utilizado (GNU Gfortran). Estos programas paralelizados implementan la estrategia que fue descripta en la sección anterior.

En todos los casos, los programas fueron ejecutados en una computadora de escritorio con una memoria RAM DDR2 de 2 Gb con un bus de 800 MHz y un procesador con una velocidad de reloj de 2.4 GHz, con tecnología de 4 núcleos, un bus del sistema de 1066 MHz y memoria cache L2 de 8Mb.

El modelo aerodinámico implementado en las herramientas computacionales ha sido validado en un trabajo anterior [2]; allí pueden consultarse comparaciones realizadas entre resultados de la herramienta y resultados disponibles en la literatura para geometrías simples y con soluciones clásicas de la teoría de perfiles delgados.

4.1. Tiempos consumidos y parámetros que influyen sobre cada tarea

En esta subsección se presentan resultados de mediciones de tiempo que muestran que el proceso de convección de la estela es la tarea que más tiempo consume en el código de NUVLM (paso 5.1 del algoritmo mostrado en la subsección 2.3 de este trabajo). Las ejecuciones de los programas aquí presentadas se realizaron para diferentes geometrías y sirven para mostrar: 1) como se distribuye el tiempo consumido por las tareas o subrutinas del programa, y 2) como los porcentajes de tiempo insumidos por las diferentes tareas son influenciados por la discretización de la geometría empleada y por la cantidad de pasos de tiempo de simulación realizados.

Para los casos ejecutados se describen las características de las geometrías empleadas y se muestran los resultados de medir el tiempo total empleado por la ejecución del programa y el porcentaje del tiempo total consumido por las distintas tareas que realiza el programa. El tiempo total se expresa en horas y los tiempos consumidos por las tareas se dan como un porcentaje del tiempo total empleado para la ejecución de todo el programa.

4.1.1. Influencia de la densidad de la malla aerodinámica

En esta subsección se muestran resultados de mediciones de tiempos de ejecución del programa en su versión secuencial simulando el comportamiento aerodinámico de una geometría simple que consistente en una placa plana con alargamiento 20 que fue discretizada en seis mallas diferentes. Los resultados de mediciones de tiempos de esta subsección 4.1.1 fueron obtenidos con una versión del programa compilado mediante Intel® Fortran para ser ejecutado en un sistema operativo Microsoft® Windows® XP®.

En la Tabla 1 se muestran detalles de las mallas aerodinámicas empleadas en cada ejecución del programa secuencial. Las mallas se caracterizan por el número de paneles a lo largo de la cuerda y el número de paneles a lo largo de la envergadura (ver primera fila de la tabla). Otros datos que se presentan son la cantidad total de paneles, la cantidad total de nudos, la cantidad de nudos que se encuentran sobre los bordes filosos emisores de estela y la cantidad de pasos de tiempo de simulación necesarios para lograr que el largo de la estela desprendida sea aproximadamente diez veces la dimensión de la cuerda.

Tabla 1 Ejecución del programa secuencial. Características de las mallas y cantidad de pasos de tiempo

Malla aerodinámica →	4 x 80	5 x 100	6 x 120	10 x 200	13 x 260	15 x 300
Cantidad total de paneles	320	500	720	2000	3380	4500
Cantidad total de nudos	405	606	847	2211	3654	4816
Cantidad de “nudos emisores”	87	109	131	219	285	329
Cantidad de pasos de simulación	40	50	60	100	130	150

En la Tabla 2 se presentan los resultados de mediciones de tiempo correspondientes a las ejecuciones del programa asociadas a los casos descriptos en la Tabla 1. En la primera columna se describen las tareas más relevantes para las cuales se realizaron mediciones de tiempo. Los resultados presentados muestran cuan importante es el tiempo empleado en la etapa de convección de la estela frente al resto de las tareas. También puede observarse que para estas geometrías, el porcentaje del tiempo total que insumen las distintas tareas está poco influenciada por los cambios en las mallas empleadas.

Tabla 2 Ejecución del programa secuencial. Resultados de mediciones de tiempos totales y parciales

Malla aerodinámica →		**4 x 80**	**5 x 100**	**6 x 120**	**10 x 200**	**13 x 260**	**15 x 300**
Tiempo total de ejecución en horas →		0.08	0.26	0.68	8.37	31.15	63.66
Porcentajes	Generar la matriz de coef. de influencia	0.18	0.14	0.11	0.07	0.05	0.04
	Generar los lados derechos	10.74	10.25	10.02	10.57	10.26	10.04
	Resolver el sistema de ecuaciones	0.10	0.10	0.11	0.20	0.30	0.36
	Realizar la convección de las estelas	86.93	87.48	87.94	87.29	87.56	87.73
	Calcular las cargas	1.91	1.93	1.77	1.84	1.82	1.82

En la Tabla 2 puede observarse que para las mediciones de tiempos realizadas, la etapa de convección consumió entre un 87 y un 88 % del tiempo total de ejecución. La tarea que le sigue en importancia es la etapa que se encarga de generar los nuevos lados derechos del sistema de ecuaciones algebraico lineal, la cual consumió entre un 10 y un 11 % del tiempo total, mientras el resto de las etapas en conjunto insumió menos del 2 % del tiempo total. Dada la marcada uniformidad de los resultados porcentuales para las distintas mallas podemos concluir que los porcentajes del tiempo de ejecución para las etapas más críticas no son influenciados por el número de elementos utilizados.

4.1.2. Influencia de la cantidad de pasos de simulación

A continuación se presentan resultados que muestran como influye la cantidad de pasos de tiempo de las simulaciones sobre la distribución porcentual del consumo de tiempo. Los resultados mostrados corresponden a ejecuciones del programa secuencial. En todas las ejecuciones se observó que el proceso de convección de la estela insume mucho más tiempo que todo el conjunto de las otras tareas.

En la Tabla 3 se muestran resultados de mediciones de tiempo efectuadas sobre el programa secuencial, para simulaciones de 25, 50, 75 y 100 pasos de tiempo. La geometría utilizada en estas ejecuciones es siempre la misma y posee las características de la malla de 10x200 paneles que se lista en la Tabla 1. Puede observarse que la distribución de los tiempos consumidos por las tareas cambia según la cantidad de pasos de simulación realizados. La tarea de convección, con solo 25 pasos de simulación ya insume el 58% del total. Esto crece de manera importante a medida que crece la cantidad de pasos, alcanzado un 87 % para 100 pasos. Este incremento del porcentaje de tiempo insumido por la tarea de convección, trae consigo un decremento en la participación porcentual en el tiempo total por parte de las otras tareas. La segunda tarea en importancia en cuanto a consumo de tiempo presenta una caída desde un 22 a un 11% al incrementar de 25 a 100, la cantidad de pasos de simulación.

Tabla 3 Tiempo insumido con el programa usando una geometría simple y distintas cantidades de pasos de tiempo

Cantidad de pasos de simulación →		**25**	**50**	**75**	**100**
Tiempo total de ejecución en horas →		0.25	1.33	3.85	8.42
Porcentajes	Generar la matriz de coef. de influencia	2.02	0.40	0.14	0.06
	Generar los lados derechos	21.95	16.95	13.18	10.70
	Resolver el sistema de ecuaciones	3.08	0.63	0.23	0.11
	Realizar la convección de las estelas	58.81	76.43	83.53	87.34
	Calcular las cargas	14.05	5.54	2.89	1.76

En la Figura 6a se muestran resultados de mediciones de tiempo efectuadas sobre el programa secuencial, para simulaciones de 1 a 200 pasos de tiempo. En estas simulaciones se utiliza una configuración de JW UAV como la que se muestra en la Figura 2. Esta geometría esta discretizada con una malla de 1425 paneles y 1660 nudos, y posee 130 nudos repartidos sobre los borde filosos que intervienen en la emisión de estelas. En la Figura 6 se muestra de manera gráfica como influye la cantidad de pasos de simulación en la distribución de tiempos consumidos por las diferentes rutinas o partes del código computacional. En la figura, puede observarse claramente que en menos de 20 pasos de simulación, la etapa de convección comienza a ser la etapa predominante en el consumo de tiempo. En el caso de simulaciones con muy pocos pasos de tiempo (menores a 6), se observa que la etapa que más tiempo consume es la resolución del sistema de ecuaciones lineales algebraicas (ver Figura 6b). A los fines de hacer diseño en ingeniería, siempre será necesario hacer simulaciones con muchos pasos de tiempo (200 pasos o más), por lo que puede concluirse que la etapa de convección siempre será la etapa más costosa y, por lo tanto, es allí donde debe concentrarse el esfuerzo de paralelizar explícitamente el código computacional a los fines de lograr programas con mayores velocidades de ejecución.

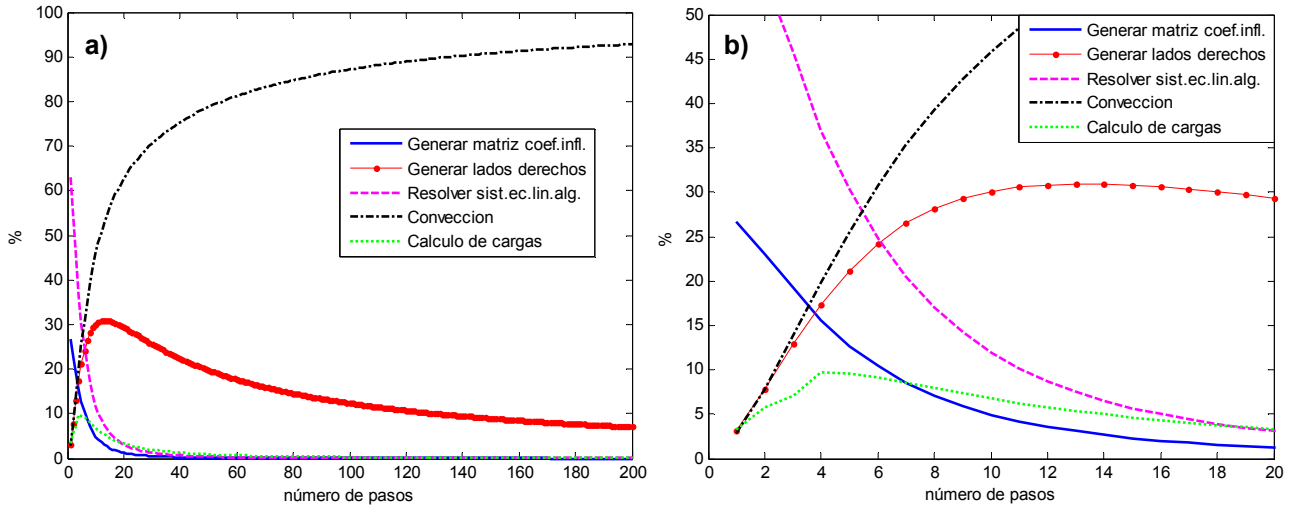

Figura 6 Influencia de la cantidad de pasos de simulación sobre la distribución de tiempos, simulaciones utilizando una configuración de JW UAV: a) simulaciones hasta 200 pasos y b) detalle de simulaciones hasta 20 pasos

4.2. Implementación de la estrategia

En esta sección se presentan resultados de mediciones de tiempo obtenidas al ejecutar el programa en su versión paralelizada. Estos programas poseen implementada la estrategia descripta en la sección 3. Se realizaron ejecuciones utilizando 2, 3 y 4 hilos de ejecución. Los resultados se presentan en la Tabla 4 donde se muestran los valores medidos de tiempos totales y parciales. Los tiempos totales se expresan en horas y los tiempos parciales se presentan como un porcentaje del tiempo total medido. Adicionalmente, en la primera columna de tiempos se muestran mediciones hechas sobre la versión secuencial del programa.

El consumo de tiempo del programa secuencial es tomado como referencia para indicar variaciones en el consumo de tiempo de las versiones paralelizadas. Las ejecuciones realizadas presentan un comportamiento esperado: cuando se usan 2, 3 ó 4 hilos de ejecución se observa un importante incremento de la velocidad de ejecución respecto del caso secuencial.

En la Tabla 4 se muestran resultados obtenidos al medir tiempos sobre ejecuciones del programa en su versión en paralelo. En todas las ejecuciones, la cantidad de pasos de simulación es 200 y la geometría utilizada es siempre la misma (la configuración de JW UAV descripta en la subsección anterior). En la última fila de resultados de la Tabla 4 se muestran variaciones porcentuales de los tiempos consumidos por las ejecuciones del programa paralelo, referidas al tiempo consumido por la versión secuencial del programa. La utilización de versiones paralelizadas con 2, 3 y 4 hilos de ejecución muestra una importante disminución en los tiempos totales de ejecución (entre un 49 y un 72%).

Tabla 4: Tiempo insumido con el programa secuencial y paralelo usando distintas cantidades de hilos de ejecución

	Número de hilos de ejecución →	Secuencial	2	3	4
	Tiempo total de ejecución en horas →	51.20	26.33	18.44	14.57
Porcentajes	Generar la matriz de coef. de influencia	<0.01	0.01	0.01	0.01
	Generar los lados derechos	7.06	13.11	18.38	23.33
	Resolver el sistema de ecuaciones	0.01	0.02	0.03	0.04
	Realizar la convección de las estelas	92.83	86.67	81.31	76.28
	Calcular las cargas	0.08	0.15	0.22	0.28
	Variación respecto de la versión secuencial →		48.6%	64.0%	71.5%

Al incrementar el número de hilos, la convección de la estela disminuye su porcentaje de participación sobre el tiempo total (93% para la secuencial y 76% usando cuatro hilos), lo que trae aparejado un incremento en la participación porcentual del resto de las tareas, entre las que se destaca al generación de lados derecho que se incrementa desde el 7%, en la versión secuencial del programa, al 23%, en la versión paralelizada con 4 hilos.

4.3. Análisis de *performance*

En esta sección se presenta un análisis empírico de la performance del código computacional que implementa la estrategia de paralelización desarrollada en este trabajo. Los modelos de performance utilizados en este trabajo se basan en la ley de Amdahl [11]. A continuación, se trazan curvas de speedup y eficiencia del código paralelizado para diferentes casos de ejecución.

El speedup se calcula realizando el cociente entre el tiempo de ejecución del programa secuencial y el tiempo de ejecución que insume el programa paralelizado al utilizar p hilos de ejecución o threads. En las Figura 7 se muestran de dos maneras los resultados de speedup obtenidos experimentalmente. En la Figura 7a se muestran curvas, para diferentes pasos de simulación, del speedup vs. el número de threads empleados. En la Figura 7b se muestran curvas, para diferente número de threads, del speedup vs. el número de pasos de simulación.

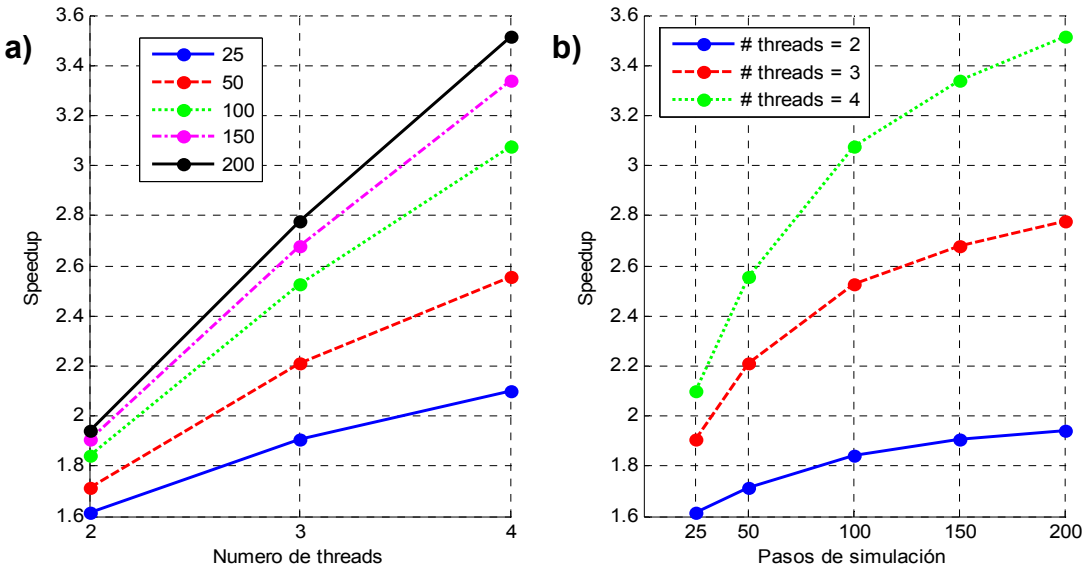

Figura 7 Speedup de la versión paralelizada del código: a) variando el número de threads para un número de pasos fijo y b) variando la cantidad de pasos de simulación para un número de threads fijo

Las gráficas de la Figura 7 muestran la ganancia de velocidad del código paralelizado respecto del código secuencial. Puede observarse que a medida que aumenta el número de hilos de ejecución utilizados, el speedup tiende a alejarse de su valor óptimo, p. Por ejemplo, en el caso de 200 pasos de simulación se puede observar que con dos, tres y cuatro threads, se obtiene aproximadamente el 97%, 93% y 88% del valor óptimo de speedup, respectivamente.

La eficiencia se calcula realizando el cociente entre el valor de speedup, para p threads, y el número de hilos de ejecución p. En la Figura 8 se muestra la eficiencia obtenida experimentalmente con el código paralelizado. En la Figura 8a se muestran curvas, para diferentes pasos de simulación, de la eficiencia vs. el número de threads empleados. En la Figura 8b se muestran curvas, para diferente número de threads, de la eficiencia vs. el número de pasos de simulación.

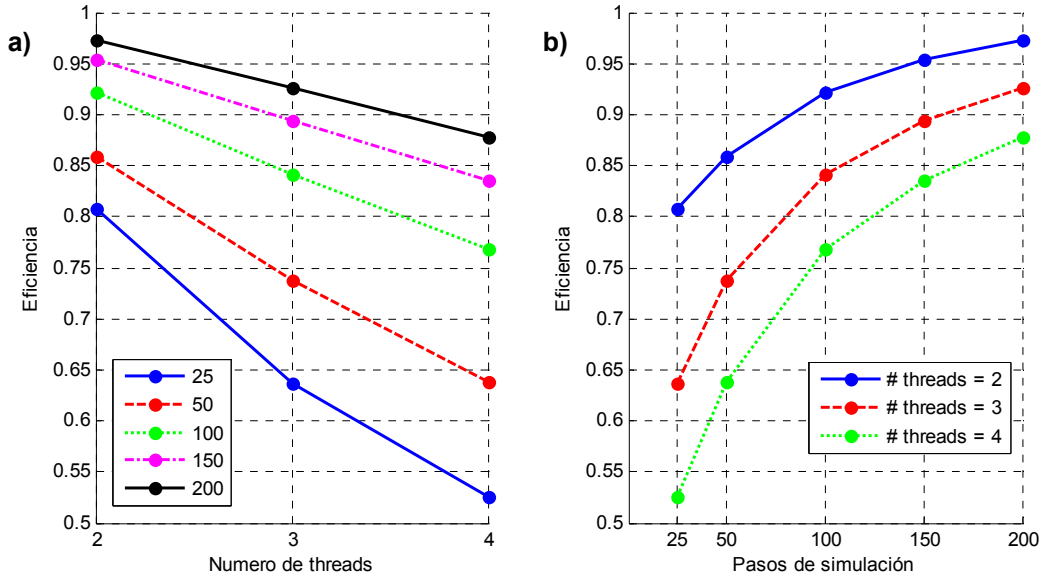

Figura 8 Eficiencia de la versión paralelizada del código: a) variando el número de threads para un número de pasos fijo y b) variando la cantidad de pasos de simulación para un número fijo de threads

Los resultados presentados en la Figura 8 nos dan una idea de la porción de tiempo que los procesadores se dedican a realizar un trabajo útil. En la Figura 8a, puede observarse una pérdida en eficiencia a medida que se aumenta el número de threads. Esta pérdida es más pronunciada cuando se realizan simulaciones con pocos pasos de tiempo. Por otro lado, puede observarse en la Figura 8b que para un número fijo de hilo de ejecución, la eficiencia del código computacional mejora según aumenta el número de pasos de tiempo de la simulación.

5. CONCLUSIONES

En este trabajo se presentó una estrategia para paralelizar explícitamente códigos computacionales que implementan el método inestacionario y no-lineal de red de vórtices. La estrategia se orientó a incrementar la velocidad de ejecución de la porción del código computacional que consume más tiempo. Para identificar los posibles “cuello de botella” se realizaron mediciones de tiempos correspondientes a ejecuciones de un código computacional secuencial desarrollado con anterioridad por los integrantes del grupo de trabajo al cual pertenecen los autores de este trabajo. En las ejecuciones realizadas se observó que la forma en que se distribuyen porcentualmente los consumos de tiempo, entre las distintas etapas de cálculo del NUVLM, no se ve influenciada por el número de elementos de las mallas aerodinámicas elegidas para discretizar la geometría de un problema. Se comprobó que la etapa de convección es crucial en cuanto al tiempo insumido por el NUVLM.

Una vez determinados los cuellos de botella, se desarrolló una estrategia de paralelización y se implementó en un código computacional. Mediante el uso del código paralelizado, y utilizando 2, 3 y 4 hilos de ejecución, se llevaron a cabo pruebas que mostraron notables incrementos en la velocidad de ejecución. Esto se traduce en importantes ahorros (entre un 49% y un 72%) del tiempo total de cálculo. Los análisis experimentales de performance realizados mostraron que la estrategia de paralelización explícita implementada computacionalmente tiene un decaimiento del speedup a medida que crece el número de threads usado. A pesar de ello las ejecuciones del código con una importante cantidad de pasos de simulación y realizadas con una simple PC de escritorio, con un procesador de cuatro núcleos, mostraron una muy buena eficiencia (cercana al 90%).

Como trabajos futuros están previstos: i) utilizar técnicas de teoría de la complejidad para realizar un estudio teórico de la performance del código computacional desarrollado, a fin de complementar el presente trabajo, y ii) explorar nuevas estrategias de paralelización explícita basadas en dividir las sábanas vorticosas en zonas que serán tratadas, durante la simulación, por diferentes hilos de ejecución.

6. REFERENCIAS

[1] L. Ceballos, S. Preidikman, y J. Massa, Generador Paramétrico de Geometrías de UAVs de Alas Unidas Orientado al Método No-Lineal e Inestacionario de Red de Vórtices, Mecánica Computacional, **27 -** 2983-3007, 2008.

[2] L. Ceballos, S. Preidikman y J. Massa, Herramienta Computacional para Simular el Comportamiento Aerodinámico de Vehículos Aéreos No Tripulados con una Configuración de Alas Unidas, Mecánica Computacional, **27 -** 3169-3188, 2008.

[3] L. Ceballos, S. Preidikman, C. Gebhardt y J. Massa, *Actas del V Congreso Argentino de Tecnología Espacial*, *Comportamiento Aeroelástico Inestacionario Y No-Lineal de Vehículos Aéreos No Tripulados de Alas Unidas: Herramienta para Relacionar el Modelo Aerodinámico con el Estructural*, Mar del Plata, Argentina, 2009.

[4] B. Roccia, S. Preidikman, y J. Massa, De la Biología a los Insectos Robots: Desarrollo de un Código Computacional Interactivo para Estudiar la Cinemática de Alas Batientes, Mecánica Computacional, **27 -** 3041-3058, 2008.

[5] B. Roccia, S. Preidikman, L. Ceballos y J. Massa, *Actas del V Congreso Argentino de Tecnología Espacial*, Implementación del Método de Red de Vórtices No-Lineal e Inestacionario para Estudiar la Aerodinámica de Micro-Vehículos Aéreos de Alas Batientes Inspirados en la Biología, Mar del Plata, Argentina, 2009.

[6] S. Preidikman, *Numerical Simulations of Interactions Among Aerodynamics, Structural Dynamics, and Control Systems*, Ph.D. Dissertation, Department of Engineering Science and Mechanics, Virginia Polytechnic Institute and State University, Blacksburg, VA, 1998.

[7] J. Katz y A. Plotkin, *Low-Speed Aerodynamics.* Cambridge Aerospace Series, Cambridge, UK, 2005.

[8] P. A. Ravetta, *Desarrollo de Simulaciones Numéricas para el Estudio Aeroelástico del Control de Actitud de Generadores Eólicos Medianos*, Tesis de Magíster, Facultad de Ingeniería, Universidad Nacional de Río Cuarto, Río Cuarto, Argentina, 2005.

[9] T. E. Fritz y L.N. Long, *42nd AIAA Aerospace Sciences Meeting and Exhibit, A Parallel, Object-Oriented Unsteady Vortex Lattice Method for Flapping Flight*, Ref. AIAA-2004-39., Reno, Nevada, 2004.

[10] P.Konstadinopoulos, D.T. Mook and A.H. Nayfeh, *AIAA Atmospheric Flight Mechanics Conference*, *A Numerical Method for General Unsteady Aerodynamics*, Ref. AIAA-81-1877, Albuquerque, New Mexico, 1981.

[11] Almeida, F., Giménez, D., Mantas, J. M. y Vidal, A. M., *Introducción a la computación paralela*, Cengage Learning Paraninfo, Madrid, España, 2008

Agradecimientos

Los autores de este trabajo desean agradecer a Ms.Sc. Ing. Julio Massa y Mg. Bruno Roccia por sus comentarios, sugerencias y aportes desinteresados a este trabajo.

Los autores, también, desean agradecer a la Secretaría de Ciencia y Técnica y a la Facultad de Ingeniería de la Universidad Nacional de Río Cuarto, a la Secretaría de Ciencia y Técnica y al Departamento de Estructuras de la Facultad de Ciencias Exactas, Físicas y Naturales de la Universidad Nacional de Córdoba, al Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET) y a la Agencia Córdoba Ciencia por el apoyo a los proyectos liderados por el Dr. Sergio Preidikman.