Análisis de rendimiento del algoritmo SGP4/SDP4 para predicción de posición orbital de satélites artificiales utilizando contadores de hardware

Federico J. Díaz[1], Fernando G. Tinetti[2,3], Nicanor B. Casas[1], Graciela E. De Luca[1], Sergio M. Martin[1], Daniel A. Giulianelli[1]

[1]Universidad Nacional de La Matanza
Florencio Varela 1903 - San Justo, Argentina
[2]III-LIDI, Facultad de Informática, UNLP
Calle 50 y 120, 1900, La Plata, Argentina
[3]Comisión de Inv. Científicas de la Prov. de Bs. As.
fedediazceo@gmail.com, fernando@info.unlp.edu.ar, {smartin, ncasas, gdeluca, dgiulian}@ing.unlam.edu.ar

Abstract. Durante los últimos 25 años, la predicción de posición orbital de los cuerpos en orbitas cercanas y medianas a la tierra, se calculo mediante el conjunto de algoritmos de la familia SGP (acrónimo de "Simplified General Perturbations"). En la última década, se produjo un incremento considerable en la cantidad de satélites artificiales, aumentando también el número de objetos inutilizados en órbita (comúnmente llamados "Basura Espacial"). Este incremento requiere un mayor esfuerzo computacional de los algoritmos utilizados. Para aprovechar en forma más eficiente los recursos computacionales actuales, puede ser necesario optimizar los algoritmos mencionados, e incluso plantear una solución paralela. El análisis aquí propuesto pretende determinar el rendimiento del algoritmo, identificando las zonas de cálculo intensivo, utilizando contadores de hardware para medir transferencias de memoria cache y rendimiento.

Keywords: SGP4, SDP4, Satélites, Contadores de hardware, Cálculo de Rendimiento, Basura espacial, Optimización, Modelado de orbitas.

1 Introducción

En forma casi ininterrumpida, durante las últimas 5 décadas, la población orbital de satélites aumento en forma gradual y constante. La red de seguimiento de satélites del gobierno de Estados Unidos, ha realizado en total desde el año 1957, el seguimiento de más de 26000 cuerpos creados por el hombre en órbita. En la figura 1.1 se ve un modelado de la cantidad de cuerpos estimados orbitando la tierra.

Esta red actualmente realiza seguimiento de 8000 cuerpos en forma simultánea (el resto ya no se encuentra en órbita), y solo el 7% de esos cuerpos son satélites

funcionales, es decir aproximadamente 560[1]. De los 8000 cuerpos entonces, 7460 aproximadamente, son lo denominado Basura Espacial ("Space Debris"). Estos cuerpos, potencialmente peligrosos, pueden provocar colisiones entre satélites funcionales, y/o reingresar a la atmosfera y caer en zonas pobladas, lo cual provocaría daños importantes en las mismas. El seguimiento constante de estos cuerpos, es primordial, además, para la actividad espacial, dado que es necesario a veces realizar correcciones de trayectorias para evitar colisiones.

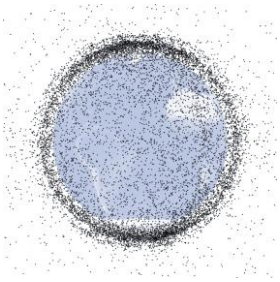

Fig. 1.1. Modelo de basura espacial y satélites en órbita cercana a la tierra. Adaptado de [1]

1.1 El problema computacional

En el año 1980 se propuso en [2] un modelo determinístico, que no requiera de un esfuerzo computacional demasiado intenso, permitiendo utilizarlo en múltiples cuerpos simultáneamente. La implementación original de este modelo, fue el algoritmo SGP[2].

El modelo actual utilizado es el SGP4, un derivado del SGP para orbitas cercanas a la tierra, y el SDP4, un derivado del SGP para orbitas superiores a los 5000 km de altura. Estos algoritmos, se utilizan para modelar orbitas cercanas a la tierra. Poseen una precisión de aproximadamente 1km, pero el error del cálculo aumenta entre 1km y 3km por cada día de predicción [3], dependiendo del cuerpo que se observe. Para reducir el error del algoritmo, se deben tomar muestras iniciales de elementos orbitales, y estas muestras se ingresan al algoritmo en un formato estándar de dos líneas[3], como veremos más adelante.

La precisión del algoritmo aumenta a medida que disminuyen los incrementos de tiempo con los que se realiza el cálculo. Pero esto conlleva a un aumento de la potencia computacional necesaria para obtener el resultado.

Mediante herramientas de performance, y contadores de hardware, se realizó un estudio sobre una implementación específica de los algoritmos, para determinar qué tipo de optimizaciones podrían aplicarse, y evaluar el rendimiento de las funciones que ejecuta.

---

[1] El número exacto no es de libre conocimiento, dado que los satélites militares no se encuentran listados en informes públicos

[2] SGP: Del inglés "Simple General Perturbations"

[3] TLE: Del ingles "Two line element"

2 El modelo de seguimiento de satélites SGP

En el presente análisis, no se pretende realizar una exposición completa del modelo, que ya está perfectamente definido en [4], pero si vamos a realizar una introducción teórica a los elementos que lo componen. También vamos a exponer las razones de porque es necesario un modelo determinístico reducido en lugar de un modelo matemático incremental que contemple todas las variables posibles.

2.1 La necesidad de un modelo simple

Si queremos realizar el seguimiento de un satélite en particular, necesitamos modelar su comportamiento orbital. En una primera aproximación, se puede realizar el cálculo de la suma de todos los vectores fuerza que se aplican sobre el satélite, y calcular instante a instante, su posición en la órbita terrestre.

Esta aproximación es lo que se conoce como integración numérica. Las fuerzas que interactúan en un satélite debajo de los 5000 km de altura (llamadas orbitas de 225 minutos), no son solo gravitatorias, también tenemos el frenado atmosférico y la presión solar, definidos como:

Frenado atmosférico. Fuerza de rozamiento que surge de la interacción del cuerpo del satélite con la atmosfera terrestre. Es significante para cuerpos en órbitas con periodos menores a 225 minutos (distancias a la superficie menores a 5000 km). [5]

Presión solar. La radiación solar genera un efecto de "presión" acumulativa en los objetos, definida por la capacidad reflectiva de la superficie donde la radiación impacta [6].

Las *ventajas* de esta solución, es que si logramos realizar incrementos pequeños de tiempo, la precisión del resultado obtenido será muy buena.

Las *desventajas* son que tenemos que conocer exactamente todas las fuerzas aplicadas al satélite en cada instante. Necesitamos muchos recursos computacionales para resolver el problema, porque si queremos calcular la posición de un satélite, dado su estado inicial, tendremos que calcular todos sus estados intermedios desde ese estado inicial.

2.2 La solución: Modelo de perturbaciones simples

En los años 80, finalmente se opto por definir un modelo que tenga en cuenta las perturbaciones en la órbita. Realizando ciertas simplificaciones, se favorece el tiempo de cálculo vs. la precisión de los resultados.

Las *perturbaciones* en una órbita, son todos aquellos efectos físicos que modifican la posición orbital de un satélite. El frenado atmosférico y la presión solar anteriormente definidos, son *perturbaciones.* Este modelo, es un modelo determinístico, que permite calcular la posición y la velocidad de un satélite para un tiempo elegido sin la necesidad de una integración numérica.

Las *consideraciones* del modelo son las siguientes: La *masa* del satélite respecto a la de la tierra es *despreciable***,** y la *órbita* del satélite se considera *cercana* a la tierra y de baja excentricidad[4]

Luego se extendió la teoría para soportar orbitas con periodos mayores a los 225 minutos. En estas orbitas, el frenado atmosférico es inexistente, pero comienzan a predominar perturbaciones como las producidas por la resonancia gravitatoria entre la el sistema tierra-luna, el sistema tierra-sol, y la presión solar. A esta modificación, que pretende abarcar todos los cuerpos artificiales orbitando la tierra, se la denominó *SGPD4*[5]

2.3 Elementos de SGPD4

En la figura 2.1, pueden apreciarse los elementos del modelo, llamados "Elementos Keplerianos" [7] [6]. Básicamente, con 6 elementos, se puede definir la posición de un objeto en una órbita terrestre para un tiempo determinado.

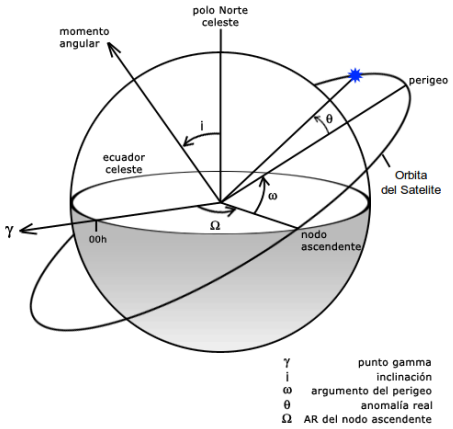

Fig. 2.1 Elementos keplerianos para un satélite en órbita. Adaptado de [8]

Donde:

Excentricidad (e) define la forma de la elipse, describiendo que tan elongada es comparada a una circunferencia (no indicada en el diagrama).

Movimiento medio es el equivalente a una velocidad media del satélite. La unidad es revoluciones orbitales por día. Está relacionada con el eje semi-mayor o el radio de la órbita. (No indicada en el diagrama).

Inclinación (i) Angulo comprendido entre el plano de la órbita, y el plano de referencia, medido en el nodo ascendente

AR[7] de nodo ascendente (Ω) orienta en forma horizontal el nodo ascendente de la elipse (donde el cuerpo realiza un tránsito ascendente respecto al plano de referencia), toando como base el punto vernal del marco de referencia.

---

[4] En oposición a una órbita de decaimiento rápido [9]

[5] Se denomina también SDP4 al algoritmo que realiza seguimiento de satélites en orbitas superiores a los 225 minutos, el nombre SGPD4 es una conjunción de ambos modelos

[6] En la referencia indicada, se consideran 8 elementos, dado que el tiempo (Época) y el frenado atmosférico(se indica como opcional), se consideran elementos del modelo

Argumento del perigeo (ω) define la orientación de la elipse en el plano orbital como un ángulo medido desde el nodo ascendente hacia el perigeo (el punto más cercano del satélite a la tierra)
Anomalía real (θ) representa el ángulo real en el plano de la elipse entre el perigeo y la posición del satélite en un tiempo determinado. A este tiempo se le llama "Época"[8]
En lugar de la anomalía real, se suele medir la anomalía media
Anomalía media es un ángulo que matemáticamente varia en forma lineal respecto del tiempo, pero no se corresponde correctamente con la anomalía real.
La *anomalía media* se puede convertir a una *anomalía real*

2.4 Limitaciones del modelo SGPD4

Como mencionamos previamente, al realizar simplificaciones en la predicción de las orbitas para optimizar el uso de recursos computacionales, estamos también, introduciendo un error en los resultados. Con tal de corregir ese error, el modelo necesita puntos de referencia para cada objeto que predice. Mediante la utilización de puntos de referencia, se pueden seguir utilizando las simplificaciones establecidas. Estos puntos de referencia, se generan mediante observaciones (directas o por radar), y se computan en un formato estándar de dos líneas, llamado TLE.

2.5 TLE: Elementos de dos líneas

Los TLE (del inglés "*Two line elements*"), se generan para cada elemento que se quiera predecir su trayectoria con el modelo SGPD4. Un TLE generado para este modelo, se realiza utilizando las mismas simplificaciones que el modelo establece, con lo cual no puede ser usado en otro modelo de seguimiento. Un TLE *solo* sirve para el modelo SGPD4. En la figura 2.2 podemos ver un ejemplo de TLE con sus elementos

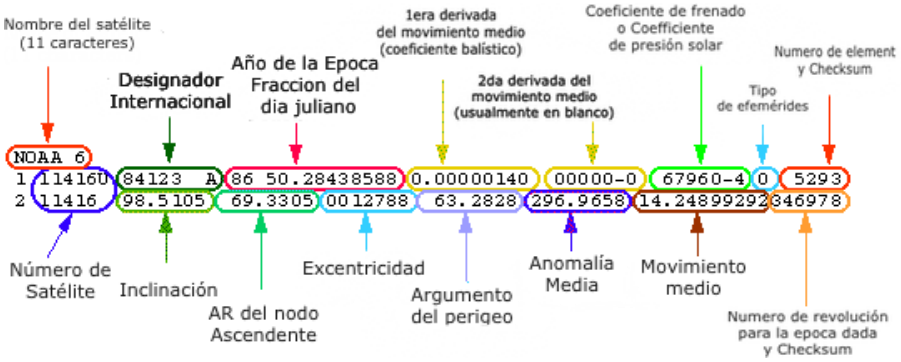

Fig. 2.2. Ejemplo de un TLE con sus elementos. Adaptado de [10]

---

[7] **AR:** Ascensión Recta
[8] Representa el tiempo en el cual se quiere realizar la predicción de la órbita

Estos parámetros condicen con los parámetros antes mencionados que el algoritmo SGPD4 necesita para predecir una órbita. Mediante observaciones se determinan, y se colocan en este formato. Para cada satélite que se quiera predecir su órbita, se necesita tener su respectivo TLE. [11]

La importancia del modelo, no radica tanto en su precisión, sino en el volumen de datos observacionales que se generan para cada satélite. La red de observación y seguimiento de satélites del gobierno de Estados Unidos mantiene una constante renovación de los TLE para todos los objetos mayores a 1cm en órbita terrestre. La cantidad de datos disponibles es lo que le da relevancia al modelo, con lo cual una optimización paralela podría producir mejoras significativas en el aprovechamiento de los recursos utilizados.

3 Análisis de rendimiento

Para la ejecución del modelo, se utilizo la implementación del algoritmo publicada en [12], llamada "Dundee SGPD4", y está basada en [13]. Esta implementación permite la lectura de los formatos TLE, y está escrita en el lenguaje de programación C

Para el problema en particular se decidió utilizar la información (TLE) de 12 objetos (satélites y basura espacial), obtenidos de [14]. Utilizando las simplificaciones del modelo, resulta menos costoso realizar modelados de orbitas, con lo cual podemos realizar predicciones con incrementos de tiempo pequeños, para reconstruir una órbita. Para estos doce cuerpos, se realizo una modelado de la órbita en incrementos de 0.1 segundos (dt), utilizando 144000 incrementos (n), lo que da un total de

$$dt * n = 14400\ segundos\ =\ 10\ días\ de\ predicciones \quad (1)$$

3.1 Análisis previo de la ejecución del algoritmo

En la tabla 3.1 claramente se ve, que el mayor porcentaje de tiempo de ejecución se encuentra repartido entre 3 funciones, siendo *sgdp4* la que concentra casi el 73% del tiempo de ejecución.

Tabla 1.1. Salida del comando gprof para el algoritmo Dundee SGPD4 prediciendo posiciones para 12 cuerpos (se omiten las funciones no relevantes en %)

El intervalo de muestreo es de 0.01 segundos.				
% Tiempo	Segundos acumulados	Segundos por función	llamadas	Nombre de la función
72.75	0.40	0.40	2005693	sgdp4
12.73	0.47	0.07	2005693	kep2xyz
9.09	0.52	0.05	1201421	compute_LunarSolar
5.46	0.55	0.03	1201421	SGDP4_dpper

También vemos, que la función *kep2xyz* fue llamada la misma cantidad de veces que *sgdp4*. Esta función lo que realiza es la conversión final de coordenadas de elementos keplerianos, a coordenadas cartesianas, pero no es relevante desde el punto de vista funcional del algoritmo, porque es un paso de conversión.

Otra función a tener en cuenta es *compute_LunarSolar,* ya que esta función, encargada de calcular las perturbaciones lunares y solares para satélites en orbitas mayores a los 5000 km, es llamada para 7 de los cuerpos elegidos, que se encuentran en orbitas superiores a la altitud mencionada. En la tabla 3.2 podemos ver las funciones seleccionadas para analizar

Tabla 3.2. Funciones candidatas a optimizar

% Tiempo	Segundos acumulados	Segundos por función	Llamadas	Nombre de la función
72.75	0.40	0.40	2005693	sgdp4
9.09	0.52	0.05	1201421	compute_LunarSolar

3.2 Análisis mediante contadores de hardware

Para analizar las funciones en detalle, vamos a hacer uso de la biblioteca PAPI (del inglés *"Performance Application Programmer Interface"*) [15].

Vamos a analizar ambas funciones, desde el punto de vista de la utilización de cache y cantidad de Flop de ejecución por segundo (FLOPs). De esta manera vamos a poder determinar con un número real, cual es la cantidad de operaciones estimadas necesarias para nuestra muestra de satélites. También, analizando los fallos de cache, vamos a poder utilizar esta información para detectar posibles optimizaciones a realizar.

La biblioteca PAPI, nos ofrece una cantidad de "Eventos" hardware, que podemos seleccionar para medir en áreas específicas de nuestro código. Estos eventos nos proporcionan información detallada de lo que está ocurriendo en el hardware cuando nuestro algoritmo se ejecuta. El análisis propuesto, es comparar las operaciones de punto flotante, con los fallos de cache, para comprobar si se obtiene una correlación entre estas dos medidas. Si esta correlación existe, entonces se podrá comprobar que el algoritmo necesita una optimización en memoria cache.

El análisis de MFlop que ejecuta esta función, se realiza mediante el evento *PAPI_FP_OPS*, que mide la cantidad de operaciones de punto flotante, y la función de biblioteca *PAPI_get_real_usec*, que funciona como un reloj real de máquina. Luego realizando una operación matemática, obtenemos los Flops

$$\frac{\text{Cantidad de FLOP}}{\text{Tiempo de ejecucion (s)}} = cantidad\ de\ flop\ por\ segundo(FLOPs) \quad (\mathbf{2})$$

Para el análisis de memoria, se utilizaran dos eventos, *PAPI_L1_DCM*, y *PAPI_L1_ICM*. El primero nos calcula la cantidad de fallos de cache de datos, y el

segundo la cantidad de fallos de cache en instrucciones. El evento *PAPI_L1_TCM,* es un evento que combina ambos fallos de cache en uno solo.

3.3 Análisis de las operaciones de punto flotante vs. fallos de cache de la función SDGP4

En la tabla 3.3 podemos ver, que en el sexto cuerpo (SGP4 simple, marcado en el recuadro), se observa un comportamiento significativo. Se producen pocos fallos de cache, pero se obtiene un alto nivel de MFlops. Esto ocurre porque la simplicidad del modelo permite una alta reutilización de las variables intervinientes, así como también un bajo tiempo de ejecución, maximizando la eficiencia del algoritmo.

Tabla 3.3. Tiempo (en microsegundos), FLOP, Flops calculados (expresados en MFlops) y fallos de cache L1 para la función SGDP4.

Modelo	**SGP4 normal**	**SGP4 normal**	**SGP4 normal**	**SGP4 normal**
Tiempo (µs)	413673	408677	409513	407684
FLOP	45718720	45260232	45426294	45508607
MFlops	111	111	111	112
Fallos de cache L1	4908969	4415024	4502586	4457233
Modelo	**SDP4 resonante**	**SGP4 simple**	**SDP4 resonante**	**SDP4 normal**
Tiempo (µs)	491014	76416	487726	521072
FLOP	69287673	5443941	69478753	76542298
MFlops	141	71	142	147
Fallos de cache L1	8279346	396473	7853828	9890781
Modelo	**SDP4 normal**	**SDP4 normal**	**SDP4 sincrónico**	**SDP4 sincrónico**
Tiempo (µs)	481717	522181	508021	499020
FLOP	67755484	79058341	77300346	75665619
MFlops	141	151	152	152
Fallos de cache L1	7460974	10583237	10314707	10449186

De aquí podemos concluir, que a mayores perturbaciones, mayores fallos de cache se obtendrán de la implementación. El cuerpo 5, y los cuerpos del 7 al 12, responden a orbitas superiores a los 225 minutos. En estas orbitas, tenemos un problema pseudo-incremental, en el cual tenemos que añadir las perturbaciones requeridas. El indicio del comportamiento del sexto cuerpo, nos marca el camino de donde el modelo necesita ser optimizado. Esto es, en el cálculo de las perturbaciones.

3.4 Análisis de la Función *compute_LunarSolar*

Para analizar esta función vamos a realizar una pequeña diferencia, y vamos a tener en cuenta por separado los fallos de cache de código, y de cache de datos. En la tabla 3.4 podemos ver que la diferencia de MFlops calculados entre el modelo SGP4 y el modelo SDP4, es básicamente el cálculo de las perturbaciones.

Tabla 3.4. Tiempo (en microsegundos), FLOP, MFlops calculados y fallos de cache L1 de código y datos para la función *compute_LunarSolar*. Se omiten los primeros 4 cuerpos, ya que responden al modelo SGP4 y no utilizan estas perturbaciones.

Modelo	**SDP4 resonante**	**SGP4 Simple**	**SDP4 resonante**	**SDP4 normal**
Tiempo (µs)	364346	0	364492	365418
FLOP	19758109	0	19780697	19742655
MFlops	54	0	54	54
Fallos de cache L1 código	2775539	0	2720791	2640799
Fallos de cache L1 datos	237326	0	186601	233858
Modelo	**SDP4 normal**	**SDP4 Normal**	**SDP4 sincrónico**	**SDP4 sincrónico**
Tiempo (µs)	365745	363005	364334	361646
FLOP	21028810	19644869	19787661	19264844
MFlops	57	54	54	53
Fallos de cache L1 código	2189155	2868253	2693908	2776276
Fallos de cache L1 datos	152355	250608	260512	273745

Este incremento de MFlops es del orden del 50% aproximadamente, debido a que los MFlops de esta función, deben sumarse a los MFlops de la función SDGP4.Los fallos de cache de código, pueden mejorarse mediante la agrupación de las soluciones.

Si se hace un análisis del TLE previo a la ejecución del algoritmo, se puede determinar qué tipo de modelo orbital utiliza. Sabiendo esto, podemos reordenar la ejecución de la predicción, para aprovechar el código de ejecución que se encuentra en cache, y solo los datos intervinientes se modificarían.

4 Conclusiones y trabajo futuro

Realizar optimizaciones de modelos orbitales, es una tarea compleja. Generalmente estos modelos involucran operaciones matemáticas complejas, y requieren un esfuerzo computacional significativo. En este primer análisis de rendimiento, se intento lograr obtener el punto de base para realizar optimizaciones futuras. Ese punto, debería ser la optimización de las funciones que calculan las perturbaciones

solares y lunares. Específicamente, la función *compute_LunarSolar* es la candidata principal a ser optimizada.

Inicialmente el modelo original, intentó ser una forma determinística de fijar posiciones de objetos en orbitas cercanas. Con el paso del tiempo, fue necesario introducir modificaciones de cálculo incremental para orbitas profundas, que trajeron consigo un incremento en la potencia computacional necesaria para resolver el problema.

Con este trabajo se intenta dar un paso hacia adelante en la dirección de la optimización del modelo predictivo. Al realizar un análisis objetivo desde el punto de vista de las operaciones en punto flotante, y del manejo de cache, la función mencionada produjo resultados que podrían indicar que es posible obtener mejoras en una optimización futura de la misma.

Se propone que el mayor esfuerzo de análisis, debería concentrarse en el desafío de obtener versiones paralelas de todas las perturbaciones calculadas posibles. Inclusive, la posibilidad de distribuir el trabajo computacional utilizando GP-GPU para este fin.

Referencias

1. Earth Observatory, NASA: Space Debris, http://earthobservatory.nasa.gov
2. Felix R. Hoots, Ronald L. Roehrich: SPACETRACK REPORT NO.3: Models for Propagation of NORAD Element Sets (Diciembre 1980)
3. T.S. Kelso: Real-World Benchmarking. Satellite Times, 3, no. 2. pp. 80-82 (Noviembre/Diciembre 1996)
4. Felix R. Hoots, Ronald L. Roehrich: SPACETRACK REPORT NO.3: Models for Propagation of NORAD Element Sets (Diciembre 1980)
5. E.M Gaposchkin and AJ. Coster: Analysis of Satellite drag. Lincoln Laboratory Journal, Massachusetts Institute of technology, vol. 1, no 2. (1998)
6. Nichols, E.F & Hull, G.F. (1903) The Pressure due to Radiation, The Astrophysical Journal,Vol.17 No.5, p.315-351.
7. The Radio Amateur Satellite Corporation, http://www.amsat.org/amsat/keps/kepmodel.html
8. Human Space Flight, NASA: Definition of Two-line Element Set Coordinate System, http://spaceflight.nasa.gov
9. John Kennewell: Satellite Orbital Decay Calculations. IPS Radio and space services, Australian Government, Bureau of Meteorology (1999)
10. Instrument Working Group, NASA: Keplerian Elements, http://earth-www.larc.nasa.gov/ceresweb/IWG/intro.html
11. T.S. Kelso: Frequently Asked Questions: Two-Line Element Set Format. Satellite Times, vol. 4, no. 3, pp 52-54 (Enero 1998).
12. Dundee satellite receiving station, Dundee University, http://www.sat.dundee.ac.uk/
13. Felix R. Hoots, Ronald L. Roehrich: SPACETRACK REPORT NO.3: Models for Propagation of NORAD Element Sets (Diciembre 1980)
14. Space-Track organization, United States Government, https://www.space-track.org/
15. Performance Application Programmer Interface, http://icl.cs.utk.edu/papi/
16. T.S. Kelso: Orbital Propagation, Part I. .Satellite Times, vol. 1, no. 1, pp 70-71 (Septiembre/Octubre 1994)
17 H. Karttunen, P. Kröger, H. Oja, M. Poutanen, K. J. Donner: Fundamental Astronomy, Springer - Verlag, Berlín (2007)
18 Fernando G. Tinetti, Armando De Giusti: "Procesamiento Paralelo. Conceptos de Arquitecturas y Algoritmos", Editorial Exacta, (1998).