保持序列的随机性与不可预测性，防止被干扰方推测出跳频规律。此外，优化过程中还需考虑通信双方的同步需求，通过合理的序列生成和更新机制，确保在动态调整时不影响数据传输的连续性与稳定性。工程实现中，可结合 FPGA 或高速 DSP 平台，实现毫秒级甚至微秒级的序列更新速度，使系统能够在干扰模式突变时迅速做出响应，从而显著提升抗干扰性能。

4.2 智能频谱感知与干扰规避

智能频谱感知技术是抗干扰的核心手段之一，其基于认知无线电理念，通过对频谱使用状态和干扰信号分布的实时监测，动态调整通信频点选择。与传统频谱扫描不同，智能频谱感知不仅能获取当前频谱占用情况，还能利用深度学习模型对干扰趋势进行预测，从而提前采取规避措施。例如，通过卷积神经网络（CNN）提取频谱能量分布特征，结合循环神经网络（RNN）对时序数据进行分析，可以准确预测未来时刻某一频段的干扰概率。在此基础上，系统可在干扰发生前将通信跳转至相对干净的频段，实现主动规避而非被动应对。该方法特别适合干扰变化频繁且强度不均的环境。此外，智能频谱感知还可与地理信息和历史干扰数据库结合，形成环境特征感知与长期策略优化机制，使跳频通信在不同区域、不同时间段均能保持较高的可用性与稳定性。

4.3 信号处理与干扰抑制技术

信号处理与干扰抑制技术通过算法手段直接在接收端削弱干扰分量、增强有用信号，从而提高系统的抗干扰能力。自适应滤波器可根据输入信号与参考信号的相关性，动态调整滤波系数以消除干扰分量，特别适用于窄带干扰抑制。空时自适应处理（STAP）利用天线阵列的空间分集特性，将空间滤波与时间滤波相结合，在空域和时域同时抑制干扰，对多源干扰和干扰信号方向可分辨的场景效果显著。此外，多用户检测技术可以在多址通信环境中区分并恢复目标信号，降低多址干扰对通信质量的影响。在工程应用中，这些方法往往与前端低噪声放大、频域均衡、信道估计等模块配合使用，形成多层级的干扰抑制链路。通过优化算法复杂度和硬件实现方式，可以在保证处理性能的前提下减少计算延迟，从而在实时通信中实现高效的干扰抑制与信号增强。

5 系统实现方法与工程应用

5.1 基于认知无线电的抗干扰架构设计

基于认知无线电的架构赋予跳频通信自适应感知与动态优化能力，由频谱感知、跳频序列生成、干扰检测决策和信号处理等模块构成。频谱感知模块实时获取频谱占用与干扰分布，序列生成模块依据感知结果和规则动态调整跳频序列，干扰检测模块结合干扰信息和链路质量快速决策，信号处理单元执行滤波、均衡、调制解调等操作。在硬件实现上，采用FPGA+DSP混合架构可兼顾高速并行处理与灵活控制，实现毫秒级响应；模块化和可重构设计则方便不同任务快速切换算法与配置，提高系统适应性。

5.2 基于机器学习的干扰识别与预测

机器学习可提升干扰识别与预测的准确性与实时性。卷积神经网络（CNN）可提取干扰信号的时频特征，长短期记忆网络（LSTM）可分析时间序列的长期依赖，从而预测干扰趋势。该组合模型可在极短时间内完成干扰分类与频段干扰概率估计，为跳频策略提前调整提供依据。工程实现中，可将模型部署在嵌入式 GPU 或高性能 DSP 中进行边缘计算推理，避免延迟与带宽占用，并通过在线微调保持在不同环境下的高准确率。借助这种机制，系统不仅能在干扰发生时迅速响应，还能提前规避，显著提高通信稳定性与抗干扰性能。

6 结语

复杂电磁环境对跳频通信的稳定性与安全性提出了前所未有的挑战。通过将自适应跳频、频谱感知、干扰抑制、多天线协同与智能预测等技术有机结合，能够显著提升系统的抗干扰能力与通信质量。未来，随着人工智能和认知无线电技术的持续发展，跳频通信的抗干扰能力将在更广泛的场景中得到验证与应用，为高可靠通信提供坚实保障。

参考文献

[1] 马松,王家豪,张蔺,等.基于深度学习的非扩跳频通信体制窄带干扰抑制技术[J].电子科技大学学报,2024,53(06):862-870.

[2] 郑重,管芸笛,李文吉,等.多头注意力机制的卫星通信抗干扰波形决策方法[C]//中国高科技产业化研究会智能信息处理产业化分会.第十八届全国信号和智能信息处理与应用学术会议论文集.中国空间技术研究院通信与导航卫星总体部;国家航天局卫星通信系统创新中心;,2024:228-234.

[3] 李洪敏.浅议超短波无线电通信抗干扰技术的发展趋势[J].中国宽带,2024,20(10):100-102.

[4] 夏重阳,吴晓富,靳越.一种面向宽带跳频双动作的智能抗干扰决策算法[J].电讯技术,2024,64(09):1467-1473.

 DOI: https://doi.org/10.12349/iser.v6i9.7441

Design of constellation coverage analysis algorithm based on parallel computing

Xu Han[1] Erlong Wei[1] Feng Su[2] Sunan Liang[3]

1. China Electronics Technology Group Corporation, 54th Research Institute, Shijiazhuang, Hebei, 050081, China
2. China StarNet Network Application Research Institute Co., Ltd., Beijing, 100029, China
3. Beijing Hongyu Aerospace Technology Co., Ltd., Beijing, 100086, China

Abstract

With the rapid development of space technology, the number of satellite constellations continues to grow exponentially. This surge in coverage analysis demands processing massive datasets containing billions of target entities. Addressing this challenge requires efficient handling of such enormous data volumes. Our study proposes a parallel computing-based design for constellation coverage analysis. By decoupling coupled computational components and distributing them across multiple CPU cores, we significantly reduce computation time. Experimental results demonstrate that when data scales reach critical thresholds, this parallel computing approach achieves substantial reductions in processing time while maintaining operational efficiency.

Keywords

parallel computing; constellation coverage analysis; decomposition and decoupling

基于并行计算的星座覆盖分析算法设计

韩续[1] 韦二龙[1] 苏峰[2] 梁苏南[3]

1. 中国电子科技集团公司第五十四研究所，中国 · 河北 石家庄 050081
2. 中国星网网络应用研究院有限公司，中国 · 北京 100029
3. 北京宏宇航天技术有限公司，中国 · 北京 100086

摘　要

随着航天事业的快速发展，各类卫星星座数量会越来越多，各种覆盖分析都存在海量目标实体的计算需求，如何快速有效地处理海量数据，是星座覆盖分析过程中需要解决的问题。本文提出了一种基于并行计算的星座覆盖分析设计的方法，把耦合的计算实体拆分解耦，并将它们分配到不同的CPU内核进行并行计算，从而减少计算时间。实验结果表明当数据量达到一定的规模后，采用并行计算的方法能够大幅减少数据处理的时间，提高效率。

关键词

并行计算；星座覆盖分析；拆分解耦

1 背景

随着航天事业的快速发展，各类卫星星座数量会越来越多，航天应用场景中的覆盖分析计算、巨型星座仿真分析计算和轨道优化计算等计算任务日益繁重，处理海量的数据需要占用大量的计算机资源和时间，海量数据的快速处理需求与数据处理长耗时的矛盾越发突出。面对各种覆盖分析、碰撞分析中的海量目标实体快速计算需求，只有提升软件系统的整体计算效能，才能满足各种计算分析应用的需求，助力我国航天事业的发展。

【作者简介】韩续（1984–），男，中国河北石家庄人，硕士，工程师，从事航天运控与仿真，航天信息应用研究。

2 现状

2.1 卫星轨道计算发展现状

随着科技的进步，航天器的发展尤为迅速，在航天领域的研究中，人们精益求精，对轨道精度提出了更高的要求，但与此同时，也要面临更复杂的轨道计算过程。为了简化这个计算过程，国内外学者进行了实验研究并得到了多个结论。

进行卫星轨道计算往往需要使用北美防空联合司令部（NORAD）开发的 SGP4/SDP4 模型 [7]，SGP4（Simplified General Perturbations）是简化常规摄动模型，SDP4（Simplified Deep Space Perturbations）是简化深空摄动模型。对于轨道周期小于 225 分钟的近地球物体，可使用 SGP4 模型；对于轨道周期大于 225 分钟或者远离地球的物体，可以使用 SDP4 模型 [8]。

在国内，空间碎片轨道预测的精度分析使用了 SGP4/SDP4 模型，指出在处理中高近圆轨道时，预测的最大偏差不大于 3 千米，且运算速度快，但计算目标的轨道近地点高度小于 500 千米时，其计算结果会产生较大的偏差 [1]。在计算一千多个不同类型的空间目标中使用了SGP4/SDP4模型，指出对于低轨目标而言，轨道高度越低预报误差越大；对于高轨目标而言，轨道高度越高预报误差越大；对于椭圆轨道目标而言，近地点高度越低误差越大 [3]。在国外，对 GPS 卫星使用 SGP4 模型进行了轨道预报，其预报结果与历书、精确星历计算结果对比后，说明 SGP4 的轨道预报误差在历元时刻前后 15 天内是可以接受的，越接近历元时刻，误差越小，历书计算结果精确度越高 [6]。

鉴于 SGP4/SDP4 模型在轨道计算中起到了不可忽视的作用，为了让研究内容更具备现实意义，本文将采用 NORAD 公布的双行元文件进行 SGP4 轨道仿真实验。

2.2 并行计算技术发展现状

自 2005 年以来，单个处理器内核的计算能力并没有显著提高，主处理器的时钟频率仍停留在 2-4 GHz 的范围内，处理器的总功耗约为 100 瓦，这是计算机可控散热的最大功率。因此，多核或多处理器并行是提高计算性能的有效途径，我们需要根据硬件设计的变化对程序进行并行调整，充分利用计算机的计算能力来加快程序的运算速度。

在处理器中引入多核 / 集群并行计算（并行计算）技术是提高算法程序计算速度的有效途径，其基本思想是将程序的计算任务进行划分，分配到不同的内核中进行计算，以达到单位时间内做更多计算的目的，从而缩短整体计算时间。

3 研究内容

航天技术快速发展至今，卫星星体数量达到了巨大的规模，人们通过庞大的星体群获得了更多更有效的数据用于生产研究，这些数据在多个领域的发展中发挥了关键作用。但在获得利好的同时，也面临着数据处理的严峻考验。

一方面，由于星体数量越来越多，进行星座覆盖分析时需要处理的数据量也随之剧增；另一方面，生产研究对星座覆盖分析的要求已经变得越来越高，不论是覆盖分析面积的增大，还是分析计算精度的提高都需要额外处理巨大的数据量。然而，受限于 CPU 单个核心的计算能力，CPU 单核心串行计算的方法在庞大的数据量面前已经是捉襟见肘的情形，完成一次覆盖分析便需要耗费很多时间，无法满足当今生产研究期待的快速完成星座覆盖分析的需求。

为了保证覆盖分析计算的精度，本文提出了闭合区域网格点的划分方法以及闭合区域可见性的计算方法；为了提高星座覆盖分析的计算速度，本文提出了星座轨道、星座覆盖的并行计算方法，并验证了星座轨道、星座覆盖的并行分组计算能力以及并行计算后的数据整合能力。

3.1 闭合区域网格点划分

任意多边形闭合区域的网格点划分过程如下所示。

(1) 算法输入：

a) 闭合区域边界的顶点集合

b) 网格的划分精度

(2) 算法输出：划分后的网格点集合

(3) 理论基础：

a)Delaunay 三角剖分算法 [5]，即所有三角形的外接圆均满足空圆性质的三角剖分

b)Bovier-Watson 算法，即通过点插入法构建 Delaunay 三角形横截面的算法

(4) 算法处理过程：

a) 基于 Delaunay 三角剖分算法，将闭合区域划分为三角形集合

步骤 1：计算区域的边界框，创建两个超三角形作为初始三角形网格，根据边界曲线创建边界点；

步骤 2：将边界点逐个插入三角形网格。每次插入后，使用 Bovier-Watson 算法计算新的 Delaunay 三角网格；

步骤 3：删除步骤 1 中插入的辅助点（并删除与之相关的三角形），以获得所有边界点的 Delaunay 三角网格；

步骤 4：在一定条件下将重心插入三角形网格区域内的三角形中，并根据 Bowyer-Watson 算法进行调整，得到新的 Delaunay 三角形交点。重复上述操作，直到所有的重心都被放置到三角形网格中，然后就得到了该区域完整的 Delaunay 三角形网格；

b) 根据网格划分精度进一步处理所有的三角形集合

遍历所有的划分后的三角形，判断每个三角形的最长边是否大于网格划分精度；

若三角形的最长边大于网格划分精度，则将此三角形作为闭合区域再次执行步骤 1 的三角形划分算法；

若三角形的最长边小于网格划分精度，则将此三角形添加到最终的划分结果中；

c) 计算三角形最终划分结果集合中每个三角形的中心点，得到划分后的网格点集合

根据计算三角形中点的算法，计算所有划分后的三角形的中心点；

d) 汇总所有的网格点得到最终的算法输出。

3.2 单个卫星对单个网格点的可见时间计算方法

单个卫星对单个网格点的可见时间计算过程包括数据输入和算法处理两个部分。

3.2.1 数据输入

输入要计算可见性的开始时间和结束时间；输入卫星的轨道参数，通过轨道计算方法求出卫星轨道数据，由此可得到卫星轨道上每个点的位置信息以及卫星经过这个点的时间信息；最后输入地球模型数据，通过模型数据确定地球

的椭圆方程；最后输入地球表面单个网格点的经纬度位置坐标，连接卫星和网格点，得到一条直线。

3.2.2 算法处理

基于地球中心 ICRF 坐标系建立笛卡尔坐标系 F，在坐标系 F 中，对于某个时刻 t，计算卫星在时刻 t 时的位置 $P_0(x_0,y_0,z_0)$，计算网格点在时刻 t 时的位置 $P_1(x_1,y_1,z_1)$，通过卫星和网格点的位置坐标 $F_0(x,y,z)$: $\frac{x-x_0}{x-x_1}=\frac{y-y_0}{y-y_1}=\frac{z-z_0}{z-z_1}$，可得出 P_0P_1 的直线方程，这个直线方程随卫星的位置改变而改变。

使用解析几何方法可以求解得到地球椭球方程 $F_1(x,y,z)$ 与直线方程 $F_0(x,y,z)$ 的随时间变化的关系，假设直线 F_0 与椭球 F_1 有两个交点，其中一个交点为网格点，另一个交点为 $P_2(x_2,y_2,z_2)$，卫星与两个交点相连，可得到两条线段 $|P_0P_1|$ 和 $|P_0P_2|$，由此可将可见性问题转化为数学问题，即：设 $d=|P_0P_1|-|P_0P_2|$，当 $d<0$ 时，表示网格点在线段 P_0P_2 之间，此时卫星与网格点可见；当 $d=0$ 时，表示交点 P_1 与 P_2 重合，直线与椭球相切，此时卫星与网格点恰好可见；当 $d>0$ 时，表示网格点在线段 P_0P_2 的延长线上，此时卫星与网格点不可见。

在实际情况中，由于卫星轨迹、地球自转都是连续量，所以对于 $d=F_3(t)$ 建立的函数也是一个连续函数。因此可以采用布伦特方法找到函数 $d=F_3(t)$ 中，$d\leq 0$ 时自变量 t 的取值范围，该范围即卫星与网格点的所有可见时间段。

3.3 闭合区域可见性计算方法

任意闭合区域的可见性计算过程如下所示。

3.3.1 算法输入

a) 卫星的轨道参数以及轨道计算方法

b) 可见性计算时间，包括：开始时间、结束时间、计算条件：瞬时通信不考虑光速

c) 覆盖区域的划分后的网格点集合

d) 计算精度参数，包括：时间精度、误差范围

3.3.2 理论基础

a) 约束算法

b) 牛顿 - 拉斐逊方法 [4]

3.3.3 算法处理过程

a) 遍历所有的网格点

步骤 1: 根据单个卫星对单个网格点计算方法的求解得到该网格点的所有可见时间段；

步骤 2: 单个网格点可见性计算过程示意图如 图 1 所示:

b) 汇总所有卫星对所有的网格点的可见时间段，得到最终的算法输出。

3.4 星座轨道并行计算处理方法

在星座轨道并行计算中输入卫星轨道根数、开始时间、结束时间、时间步长、积分器等；对数据进行处理计算后，输出卫星星历数据。

由于每个卫星的轨道计算是相互独立的，所以可以根据要计算的卫星轨道数量和当前环境的处理器核心数，得到划分给每个并行计算子模块所要执行的计算任务。当每个并行计算子模块计算完毕后，即可将计算结果输出到结果收集进程（即主进程）。

轨道并行计算算法流程图如 图 2 所示：

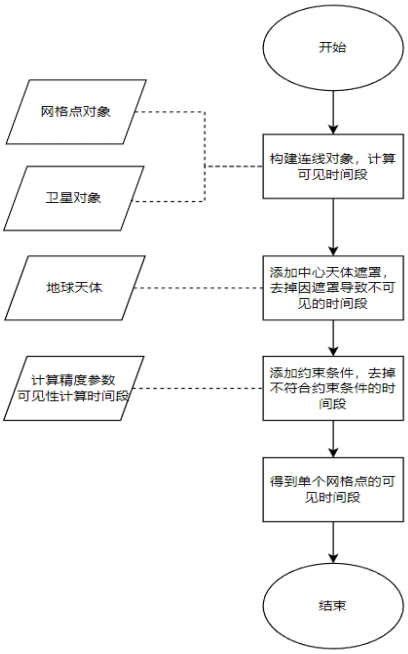

图 1 单个网格点可见性计算过程示意图

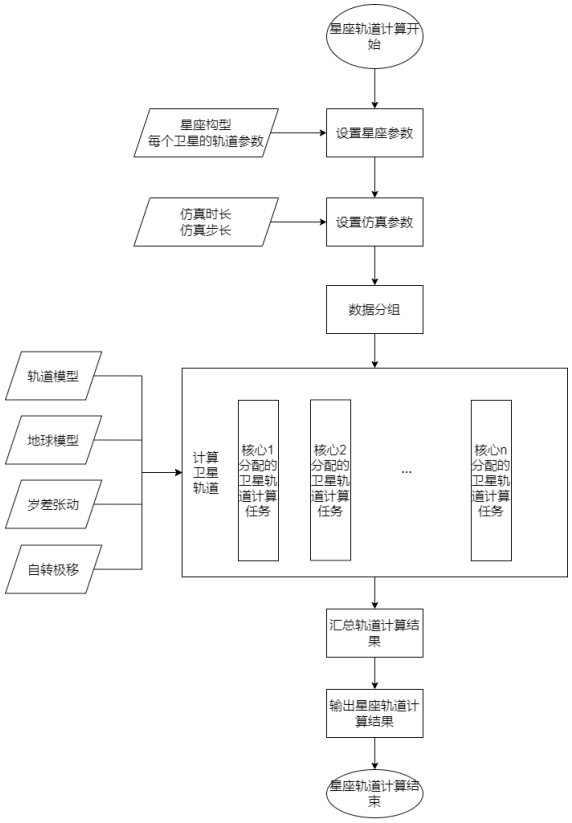

图 2 星座轨道并行计算算法流程图

图中，轨道模型采用 SGP4 算法模型。Vallado 等对 1980 年以来的多种 SGP4 版本进行了总结，提出了普遍认可的双行根数的时间系统和坐标系统。双行根数采用的时间系统为协调世界时 UTC，坐标系统为瞬时真赤道平春分点坐标系（TEME）。TEME 与 J2000 坐标系存在如下的变换关系：

$$\vec{r}_{J2000}=(R_P)^{-1}\cdot(R_N)^{-1}\cdot R_Z(-\Delta u)\cdot\vec{r}_{TEME}$$

式中：R_P为岁差矩阵，R_N为章动矩阵，R_Z为绕 Z 轴的旋转矩阵，Δu 为赤经章动。J2000.0 采用的时间系统为地球动力学时 TT，TT 与 UTC 和原子时 TAI 存在如下转换关系：

$$TT=TAI+32.184^s$$

$$TAI=UTC+\Delta AT$$

其中，ΔAT 为跳秒，具体可查阅 IERS 的公告。

3.5 星座覆盖并行计算处理方法

在星座覆盖并行计算中输入卫星星历、开始时间、结束时间、时间步长、覆盖区域以及覆盖特性值定义等，对数据进行处理计算后，输出星座对区域的覆盖特性值数据。

数据分割的方式是将区域的覆盖计算划分成多个小区域的覆盖计算，对于每个小区域，都需要计算整个星座对其的覆盖特性值数据。根据区域的划分数量和当前环境的处理器核心数，将所有区域覆盖计算任务分配到不同的内核中进行并行计算。

当所有区域覆盖计算完成后，将计算结果数据归集到某个进程中，进行合并处理。数据合并处理完成后得到卫星星座对于整个区域的覆盖特性值数据。

星座覆盖分析并行计算算法流程图如 图 3 所示：

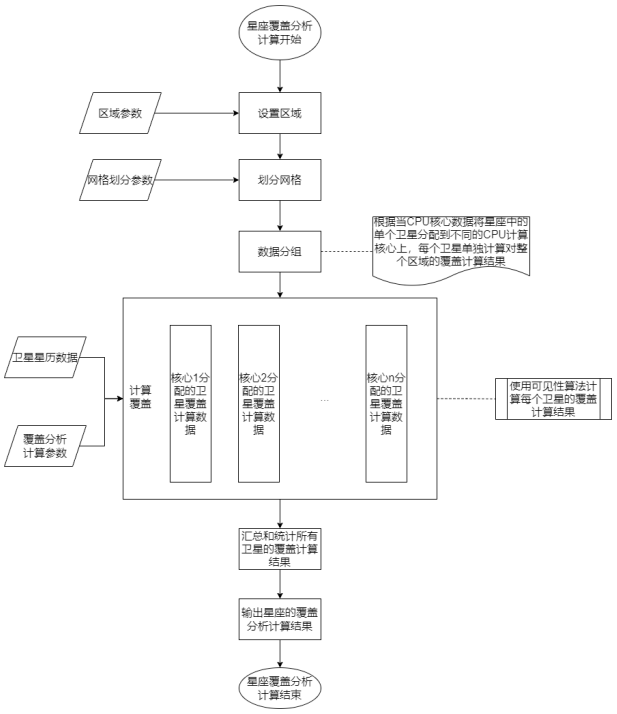

图 3 星座覆盖分析计算算法流程图

覆盖特性值数据是从覆盖数据中统计出来的，以覆盖特性值平均访问时间（Access Time）和平均重访时间（Revisit Time）为例，说明一下覆盖特性值得统计算法。经过可见性计算，区域中某个点被覆盖资源访问的时间是不连续的时间段序列，记为 Intervals，不被访问的时间也是不连续的时间段序列，记为 Gaps，则平均访问时间和平均重访时间的统计算法如下：

$$(\text{AccessTime})_{\text{average}}=\frac{\sum_{i=0}^{M}(\text{Intervals})_i}{M}$$

$$(\text{RevisitTime})_{\text{average}}=\frac{\sum_{i=0}^{N}(\text{Gaps})_i}{N}$$

其中，M 为所有可见时段的个数，N 为所有不可见时段的个数。

4 实验验证

4.1 卫星组数不同

（1）计算环境：

仿真使用的轨道参数为：半长轴 8000 千米、偏心率 0.0、轨道倾角 86.0° 、升交点赤经 0.0° 、近地点幅角 0.0° 、平近点角 0.0° 、定轨时间 2022 年 1 月 1 日 12:00:00UTCG。转换到双行元格式后的轨道参数下所示：

1 11801U 17001.50000000 .00000000 00000-0 00000-0 0 8

2 11801 86.0000 00.0000 0000 0.0000 0.000 12.1330 6

仿真时间：2022 年 1 月 1 日 12:00:00 UTCG~2022 年 1 月 2 日 12:00:00 UTCG

步长：60s；

操作系统：Windows 10 x64

CPU：英特尔酷睿 i7-10700，CPU 架构：Comet Lake-S，核心主频：3.8GHz, 加速频率：5.1GHz，三级缓存大小：16M，8 核心 16 线程，125W 热设计功耗。

GPU：NVIDIA GeForce GTX 860M， 具备 640 个 CUDA 核心，配备 4096MB 的显存容量。

（2）计算结果对比：

实验数据基于 NORAD 发布的双行轨道参数文件，设置覆盖区域范围：纬度环 -5° ~ 5° ，分辨率 1° ，分别读取 1 组，10 组，100 组，500 组，1000 组，5000 组，10000 组卫星在 release 环境下进行 SGP4 轨道仿真，对比运算效率与加速比，如表 1 所示。

4.2 覆盖范围不同

（1）计算环境：

模拟中使用的轨道参数如下：半主轴 8000 公里，偏心率 0.0，轨道倾角 86.0° ，节距长度 0.0° ，近地点幅度 0.0° ，平面近地点角 0.0° ，定点时间 2022 年 1 月 1 日 12:00:00 UTCG。转换为双列元格式后的轨道参数如下：

1 11801U 17001.50000000 .00000000 00000-0 00000-0