Journal of System Simulation

Volume 27 | Issue 8 Article 11

8-3-2020

Design of Unified Algorithm for Sampling MPDU

Ti Zhou
Beijing Aerospace Control Center, Beijing 100094, China;

Miao Yi
Beijing Aerospace Control Center, Beijing 100094, China;

Yang Jian
Beijing Aerospace Control Center, Beijing 100094, China;

Design of Unified Algorithm for Sampling MPDU

Abstract

Abstract: The standard of CCSDS AOS is used in space exploration because of the process of the standard airspace system. The spacecraft missions have used multi-protocol data unit (MPDU) to multiplex the package channels. The range of MPDU and the data are different in different spacecrafts. This process brings some problems in the maintenance and the correctness of the algorithms of virtual channel multiplexing in simulation system. The applications of CCSDS in the world were researched, and the general definitions of MPDU in these missions were described. *A circle memory was given to buffer the data of MPDU, two general formats in the process of sampling MPDU were researched, an efficient and maintenance-free algorithm for unified sampling MPDU was designed, the efficient of time was analyzed, which proves its correctness in theory*. The application of this algorithm was given, and the future application and work were pointed out.

Keywords

AOS Standard, multiplex, MPDU, sample, algorithm, correctness

Recommended Citation

Zhou Ti, Miao Yi, Yang Jian. Design of Unified Algorithm for Sampling MPDU[J]. Journal of System Simulation, 2015, 27(8): 1729-1734.

第 27 卷第 8 期 系统仿真学报© Vol. 27 No. 8
2015 年 8 月 Journal of System Simulation Aug., 2015

通用 MPDU 组帧算法设计

周倜，苗毅，杨健
（北京航天飞行控制中心 北京 100094）

摘要：航天系统的标准化令航天任务大量采用 CCSDS AOS 标准，该标准通过使用 MPDU 进行包信道复用，MPDU 大小和填入数据也随着航天器的不同而不同，给仿真系统带来了信道复用算法的维护效率与正确性保证等问题。研究了 CCSDS 在国内外的研究应用情况，介绍了航天任务中 MPDU 组帧的一般性规定，*提出了 MPDU 缓存队列的环形存储方式，研究了 MPDU 组帧过程的 2 种最一般形式，设计了一种高效、易于维护的通用 MPDU 组帧算法，给出了时效分析，并通过理论分析证明了该算法的正确性*，给出了本算法的具体应用情况并指出了未来的应用前景。

关键词：AOS 标准；复用；MPDU；组帧；算法；正确性
中图分类号：TP391 文献标识码：A 文章编号：1004-731X (2015) 08-1729-06

Design of Unified Algorithm for Sampling MPDU

Zhou Ti, Miao Yi, Yang Jian
(Beijing Aerospace Control Center, Beijing 100094, China)

Abstract: The standard of CCSDS AOS is used in space exploration because of the process of the standard airspace system. The spacecraft missions have used multi-protocol data unit (MPDU) to multiplex the package channels. The range of MPDU and the data are different in different spacecrafts. This process brings some problems in the maintenance and the correctness of the algorithms of virtual channel multiplexing in simulation system. The applications of CCSDS in the world were researched, and the general definitions of MPDU in these missions were described. *A circle memory was given to buffer the data of MPDU, two general formats in the process of sampling MPDU were researched, an efficient and maintenance-free algorithm for unified sampling MPDU was designed, the efficient of time was analyzed, which proves its correctness in theory*. The application of this algorithm was given, and the future application and work were pointed out.

Keywords: AOS Standard; multiplex; MPDU; sample; algorithm; correctness

引言

随着航天事业的蓬勃发展，航天器从初期仅传输遥控、遥测数据，到现在增加传送实验科学数据、有效载荷和图像话音等数据。传输的数据复杂，传输率高，信息交换量大。为了适应以统一数据流传送信息的新趋势[1-2]，航天任务采用 CCSDS 建议的相关内容定义上/下行数据传输格式。目前国内外主要的航天任务均采用 CCSDS 建议的相关规定开发软硬件产品，例如国际空间站项目[3-5]和地球观测系统(EOS)[6-7]。航天任务大量使用 CCSDS 建议[8]，CCSDS 的高级在轨系统(AOS)协议已广泛应用于载人航天和探月工程等航天计划中。

收稿日期：2015-04-28 修回日期：2015-07-03；
作者简介：周倜(1981-)，男，湖南岳阳，工程师，博士，研究方向为软件工程与系统仿真；苗毅(1976-)，男，陕西神木，高工，硕士，研究方向为航天测控软件设计与系统仿真；杨健(1982-)，男，四川通江，硕士，工程师，研究方向为航天测控软件设计与系统仿真。

为了能使不同类型的数据共享同一数据链路，CCSDS AOS 采用了两级多路复用机制：包信道复用和虚拟信道复用。通常通过总线传送到高速复接器的数据都以数据源包等形式占用一个或多个虚拟信道，然后将虚拟信道数据形成虚拟信道数据单元，各信道的数据复接在一起后，传送给 S 波段发射机[9]。

以下行数据为例，采用 CCSDS AOS 推荐的信道访问数据单元(CADU)格式，将 CTU 传递给数据处理单元的遥测数据源包 EPDU 组成 MPDU(多路协议数据单元)，MPDU 填充在 VCDU 的数据单元区，VCDU 附加同步码组成 CADU。

MPDU 是利用多路复用业务把多个源包数据按一定格式组合在一起，以使多个源包共用一个虚拟信道传输。在航天任务中已采用该复用模式进行上行和下行虚拟信道复用。但是 MPDU 的大小在不同任务中是不一样的，甚至在同一任务中，不同用途的飞行器所使用的 MPDU 包大小也不相同，这就给仿真系统开发和上行数据组 MPDU 包带来了非常多的重复工作。在以往的载人航天任务仿真系统中，MPDU 组包过程与 MPDU 包大小和待存入的 EPDU 包大小关系密切，控制逻辑复杂，移植过程困难。以某航天仿真系统为例，MPDU 组帧程序多达 231 行，if-else 语句多达 20 个，在向其他航天任务仿真系统迁移时，改造难度非常大，也不利于任务准备期间的维护。

国内外的相关研究主要集中在 AOS 数据系统方案设计、不同帧长的产生概率与 MPDU 包信道复用之间的关系模型等航天器信道设计关键技术[9-13]，没有专门研究 MPDU 组帧算法。为了适应航天任务 MPDU 包组包的复杂要求，需要开发出统一高效的 MPDU 组包算法，以解决仿真系统 MPDU 包组帧复杂和移植效率低下的难题。

在开发某航天任务仿真系统时，为解决其上的多目标飞行器 MPDU 组帧算法复杂、不易维护的问题，本文对 MPDU 的结构和组帧过程进行了研究，提出了一个通用的高效算法。

1 多路协议数据单元(MPDU)

航天器的下行遥测多路复用的过程为：依据遥测模式分别对遥测源包 EPDU 进行多路复用，生成多路协议数据单元(MPDU)，填充入虚拟信道数据单元(VCDU)数据单元区中。如果填入 MPDU 数据区末尾的源包 EPDU 超出数据区长度，则将超出部分顺序填入下一个 MPDU 数据区中。如果从用户得不到足够的源包，多路复用功能可以产生适当长度的填充包(格式为 CCSDS 包)，填入 MPDU 中。最短的填充包长度为 7 字节(6 字节导头，1 字节填充数据)，如果一个 MPDU 中所需的填充数据小于 7 字节，则产生一个长为 7 字节的填充包，此包填满这个比 PDU，然后溢出到下一个 MPDU 中。

多路协议数据单元由多路协议数据单元导头和多路协议数据单元包域组成，格式如图 1 所示。MPDU 的长度是固定的，通常是一个虚拟信道协议数据单元中数据域的长度。

MPDU 导头		MPDU 包域				
备用(5bits)	首导头指针(11bits)	前一源包#K 的结束	源包#K+1	…	源包#M	源包#M+1 的开始

图 1 MPDU 数据结构

首导头指针：MPDU 导头中的首导头指针直接指向第一个源包的起始位置，根据源包中包长度的标志就可以区分出每个独立的源包。

如果 MPDU 包域的第一个字节就是源包头的第一个字节，则首导头指针为 0。

如果 MPDU 中不包含源包包头，那么首导头指针域设为“不含包头”(二进制代号)。

如果 MPDU 中不包含任何有效的用户数据，也就是只包含填充包，那么首导头指针域设为“填充包”(二进制代号)。

如果一个源包的导头被分到两个 MPDU 中的情况（被分割的源包分在 MPDU（x）和 MPDU（x＋1）中），如果该源包是 MPDU（x）的第一个源包，那么 MPDU（x）的首导头指针域指向这个

源包首地址。如果不是 MPDU（x）的第一个源包，那么 MPDU（x+1）的首导头指针指向被分割源包的后一个源包头。

2 软件设计

本节首先对 MPDU 缓存队列进行设计，将其定义为一个较大的循环队列，然后分析了源包填入 MPDU 数据区时最一般的两种情况，针对 MPDU 填写的最一般情况设计了与任务型号无关的 MPDU 组帧算法，最后对算法的效率进行了分析，并对算法的正确性进行了证明。

2.1 MPDU 缓存设计

在软件中预留 n 个 MPDU 包大小的缓存循环存储新生成的 MPDU 包，如图 2 所示。其中，SendIndex 指向即将要发布的 MPDU 缓存区，SampIndex 指向即将要填入的 MPDU 缓存区当这两个指针指向同一片缓存时，表明 MPDU 缓存队列中无数据。当队列中有数据，并发送了一个 MPDU 数据包时，SendIndex 指向下一个缓存区。当 SampIndex 填入数据时，SampIndex 将在循环缓存区指向下一个待写入缓存，如果此时 SampIndex==SendIndex，则 SendIndex 指向下一个缓存区。

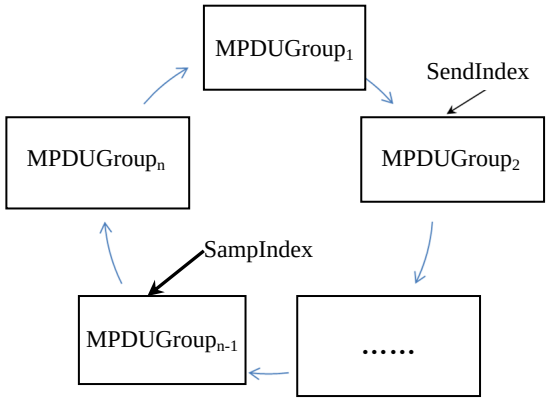

图 2 MPDU 循环缓存区

2.2 MPDU 组帧算法

在将数据源包填入 MPDU 包时，可能出现如图 3 所示的 2 种情况：(a) 给出了源包可以完全填入当前 MPDU 数据区的情况；(b) 给出了待填入源包需填入多个 MPDU 数据区的情况。根据 MPDU 导头填写要求，当数据源包导头填入 MPDU 源包时，只需考虑第一个 MPDU 源包导头是否已经填写，如果没有填写，则将当前待填入位置 i 填入 MPDU 导头 H 中。对于情况(a)(b)的数据区填写，如果有源包数据未填入 MPDU 包，则考虑该 MPDU 包能否装下剩余的源包数据，如果能够装下，则直接将源包数据拷入 MPDU 数据区，否则用数据填满 MPDU 数据区，并将剩余数据依次填入后续 MPDU 包中，直到无源包数据需要填写为止。

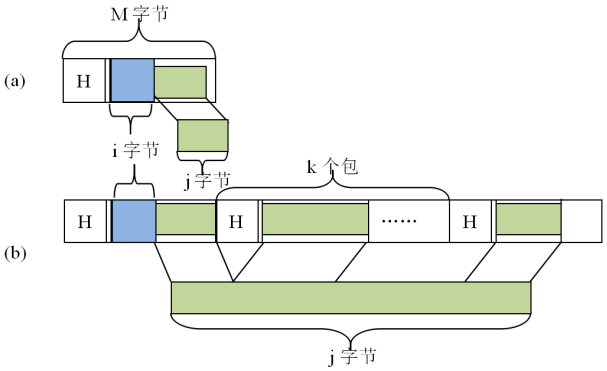

图 3 MPDU 包存储情况

为了算法设计方便，定义 MPDU_CNT 个 MPDU 缓存区为缓存区数组 szMPDUGroup[MPDU_CNT]，数组的元素为长度为 MPDU_LEN(即图 3 中 MPDU 包的总长 M 字节)的缓存。当前 MPDU 包待填写位置指针为 usMpduIndex，该指针从 MPDU 包头开始计数，即包含 MPDU 包头 H 的 2 字节。当前 MPDU 包头是否填写标记为 blDirty。当前待填入源包数据缓存为 szBuffer，全部数据长度为 in_usPackLen，剩余未填入 MPDU 包的数据长度为 s_usPackLen，已填入 MPDU 包的字节数为 s_uiIndex，本次拷入 MPDU 包的字节数为 s_usCopyLen。算法设计如下：

算法名称：MPDU 包填写算法

输入：

源包数据缓存 szBuffer[in_usPackLen]

输出：输入的源包数据已填入 MPDU 包缓存

序列的末尾。

过程：

1. 如果没有填写 MPDU 导头(blDirty==false)

1.1 则将 usMpduIndex-2 填入导头，blDirty←true.//填写导头

2. s_usPackLen←in_usPackLen; s_uiIndex←0;

3. 当 s_usPackLen>0 时，循环：

3.1 如果 s_usPackLen+usMpduIndex< MPDU_LEN

3.1.1 则 s_usCopyLen←s_usPackLen

3.1.2 否则 s_usCopyLen←MPDU_LEN-usMpduIndex

3.2 memcpy(szMPDUGroup [SampMIndex]+ usMpduIndex, szBuffer+s_uiIndex, s_usCopyLen);

3.3 usMpduIndex←usMpduIndex+usCopyLen;

3.4 s_usPackLen -= s_usCopyLen;

3.5 如果 usMpduIndex == MPDU_LEN

3.5.1 则 usMpduIndex←2;

3.5.2 SampMIndex= (SampMIndex+1)% MPDU_CNT;

3.5.3 如果 SampMIndex == SendMIndex

3.5.3.1 则 SendMIndex=(SendMIndex +1)% MPDU_CNT;

3.5.4 将当前 MPDU 包导头设置为不包含源包包头标志

3.5.5 MPDU 导头设置为未填写状态 (blDirty ← false)

本算法通过第 3 步的循环将源包数据顺序填入可用的 MPDU 包，当数据全部填入后算法终止。分析该算法可知，时间耗费主要在第 3.2 步缓存拷贝和第 3.5.2，3.5.3.1 步取余操作，长度为 j 字节的源包数据填入长度为 M 的 MPDU 包，循环体最多可能执行 j/(M-2)+2 次。假设内存拷贝需耗时 t_1，取余操作耗时 t_2，则本算法的时间耗费为 $O((t_1+2\times t_2)\times(j/(M-2)+2))\approx O((t_1+2\times t_2)\times j/M)$。由此可见，本算法的时间效率取决于源包和 MPDU 包的大小。在一个确定的任务中，MPDU 包的大小是固定的，源包则根据实际数据量大小而有所不同，因此本算法在一次任务中的时间耗费取决于源包大小。在不同任务中，该算法只需重新定义缓存区数组元素个数 MPDU_CNT，以及数组元素的长度 MPDU_LEN，即可适应新的任务组包需求。

为了在理论上证明本算法的正确性，下面将对 MPDU 导头正确填写的命题进行证明。

命题 算法正确填写 MPDU 导头。

证明：(1) 源包填入 MPDU 源包时，如果当前 MPDU 源包没有填写导头，根据首导头指针的定义可知，前 i 个字节中没有源包的的起始位置，由于本次源包填入时缓存的第一个字节即是源包的起始位置，因此算法的第 1 步将填写本 MPDU 第一个源包的起始位置。由于 usMpduIndex 指向 MPDU 的写入位置(包含导头)，而导头指针是以 MPDU 数据区的起始位置为基准，所以在第 1.1 步需将 usMpduIndex-2 填入导头，当前导头填写标志被置为已填写 blDirty==true。

(2) 如果源包能全部填入当前 MPDU 包且未填满当前包时，usMpduIndex < MPDU_LEN，第 3.5 步不可进入。后续源包填写时，由于当前 MPDU 包已填写(blDirty==true)，当前 MPDU 包头不会被改变。

(3) 如果源包能全部填入当前 MPDU 包且填满当前包时，进入第 3.5 步，将下一 MPDU 缓存置为当前 MPDU 包(第 3.5.2~3.5.3 步 SampMIndex 后移一块)。第 3.5.4 步将 MPDU 导头置为不含源包导头，但是在第 3.5.5 步将填写标志设为未填写 blDirty==false。通过循环，可将(b)中的后续的 k+1 个 MPDU 包的包头置为无源包包头。由于每次填满 MPDU 数据区时，算法在第 3.5.2~3.5.3 步将指向下一个可填写的 MPDU 包，因此循环结束时，当前 MPDU 包一定未填满。当下一个源包到来时，可填入当前 MPDU 包，由于当前 MPDU 包导头填写标志为未填写，所以该新到源包为第一个源包，根据(1)可在 MPDU 导头正确填入第一个源包的起始位置。

3 算法对比分析

在以往某航天器仿真系统中，MPDU 组帧程

序多达 231 行，if-else 语句多达 20 个，其主要思想是根据 MPDU 包大小和源包大小进行分支控制，每个分支确定源包向 MPDU 包序列的填写方式。由于以往该系列的源包最大为 600 字节左右，MPDU 包为 200B 左右，理清可能的分支个数及种类的复杂度不太大，但是本次航天任务中源包最大为 500B 左右，而其中的一个航天器的 MPDU 包数据区为 235B，另一个航天器的 MPDU 包数据区为 436B。原算法在向该航天任务仿真系统迁移时，不同航天器均需根据实际控制情况分别编写各自专有的 MPDU 组帧算法，每个航天器对应的 MPDU 组帧代码量与原仿真系统基本相当，改造难度非常大，不利于任务准备期间的维护。后续航天任务的最大源包可达 4000B，如果采用原算法思想编写组帧控制程序，其复杂程度和代码量将呈指数上升，因此新算法在程序通用性和可维护性方面优势明显。本文使用 C++程序实现了通用 MPDU 组帧算法，下表给出了新旧算法具体实现程序的应用对比分析结果。

表 1 新旧组包程序比较

	某仿真系统原 MPDU 组包程序	采用新算法的 MPDU 组包程序
程序行数	231	30
if-else 语句数	20	4
通用性	不通用，根据源包和 MPDU 包大小重新分析、编写控制流程	通用
可维护性	差，任何一个分支错误都非常难以察觉，修改困难	好，各种可能均归约到循环处理，修改方便
适应性	源包或 MPDU 包大小变化都将引起控制程序修改	算法不受包大小变化的影响
测试	每个分支均需测试，测试等价类多，难以全部验证	分支少，测试等价类少，易于验证

在 SUN 工作站(SUNW, Sun-Fire V890; sparc; sun4u)、内存 32GB、操作系统为 SunOS5.10 的条件下运行某任务的仿真系统，在实际运用环境中随机截取连续的 4500 次组帧状态，得到组帧时间与采样点的分布如图 4 所示，其中纵坐标为组帧时间耗费，单位为 ns。由图可以看出，组帧的时间耗费集中在 200ns 左右，极少数在 1~2ms 之间。

图 5 给出了各时间耗费区间之间的比例，该图表明算法组帧速度非常快，400ns 以下的采样点达到总数的 79.20%。

图 6 给出了各时间耗费区间之间的比例。

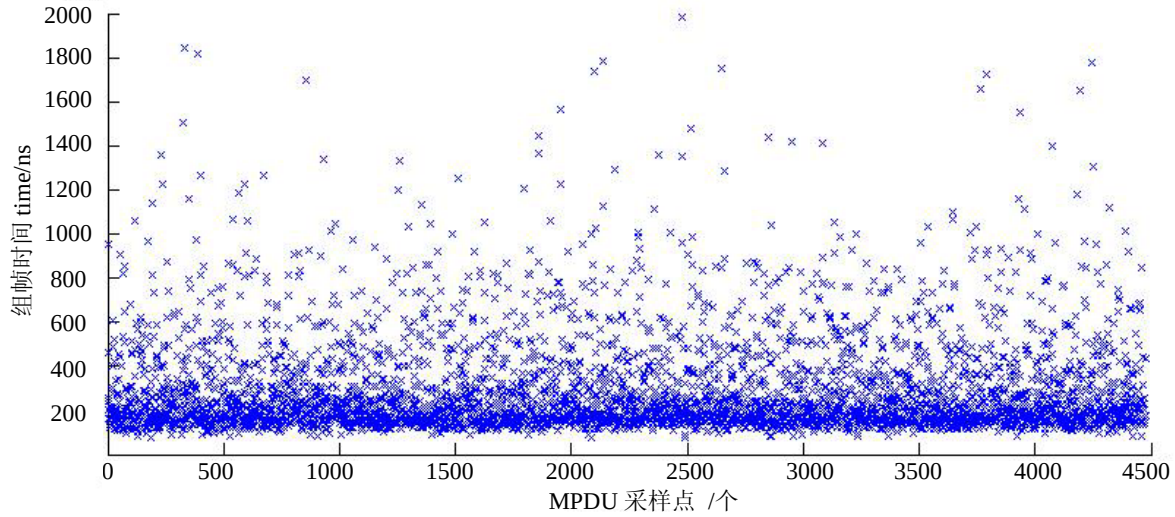

图 4 采样点的组帧时间分布

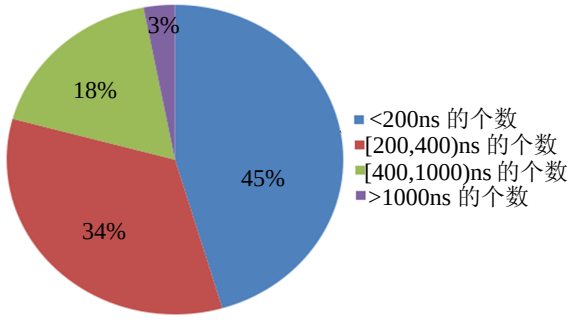

图 5 MPDU 组帧算法耗时分布

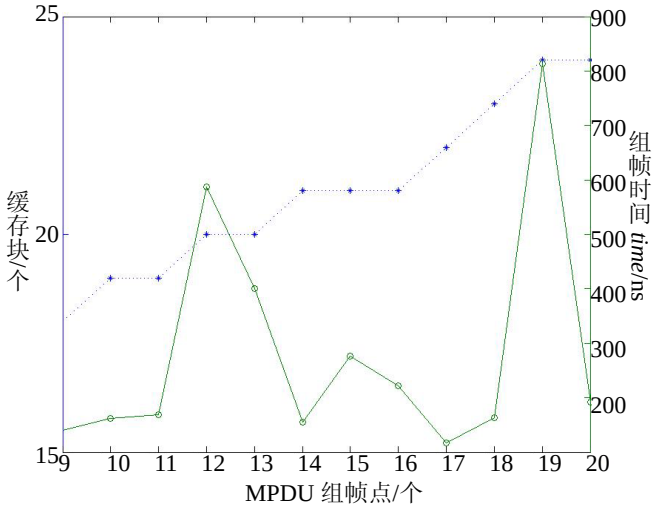

图 6 跨缓存块与组帧时间的关系

点达到总数的 79.20%。给出了第 1009~1020 个采样点之间组帧填写缓存块与组帧耗时之间的关系。图中左边的纵坐标标注了组帧点所使用的缓存块(即算法中的 SampIndex)，在图中以*标注采样点所对应的 SampIndex，并用虚线连接相邻各点。图中右侧纵坐标标注了组帧的时间耗费，单位 ns，在图中以圆圈标注采样点所对应的时间耗费，并用实线连接相邻点的时间耗费。由图可以很明显的看出，在同一个缓存块中存放源包数据时时间耗费较小，一般在 200ns 左右；在跨缓存块存放源包数据时，时间耗费会同步上升。

4 结论

本算法简洁高效，独立于 MPDU 和源包的大小，算法实现后已应用于多飞行器仿真系统，各类 MPDU 包组包正确，圆满完成了该航天任务的各项联试工作，有力保障了该任务的顺利进行。目前该算法程序已直接迁移到其他航天仿真系统，未加修改即可很好的适应源包大小在 14B~4000B 之间的大跨度范围 MPDU 组帧过程，时间效率高，满足飞行控制的各项联试要求。该算法可直接应用于其它航天任务上行/下行的 MPDU 组帧过程，具有很好的通用性，后续可在未来的空间站等其它航天任务仿真系统和上行 MPDU 组帧过程中推广使用。

参考文献:

[1] 姜昌，范晓玲. 航天通信跟踪技术导论[M]. 北京：北京工业大学出版社, 2003.
[2] 黄薇，张纪生. CCSDS AOS 建议介绍[J]. 飞行器测控学报, 2000，19(3): 37-46.
[3] Adrian J. Hooke. CCSDS Advanced Orbiting Systems: International Data Communications Standards for the Space Station Freedom [J]. IEEE Network (S0890-8044)，1990, 4(5): 13-16.
[4] Johnson M J. Modeling of the Space Station Freedom data management system[C]// Global Telecommunications Conference, 1990, San Diego, CA:IEEE, 1990:572-578 .
[5] Joseph F. Smith. Overview of the Space Station Communications Networks [J]. IEEE Network (S0890-8044), 1990, 4(5): 22-28.
[6] Toby Bennett. EOS High Rate Telemetry Processing Components [C]// International telemetry conference, 1993, Las Vegas, NV:NASA, 1993 :485-491.
[7] Barbara Brown. High Performance CCSDS Processing Systems for EOS-AM Spacecraft Integration and Test [C]//International telemetry conference, 1995, Las Vegas, NV:NASA,1995: 519-525.
[8] 孙辉先，陈小敏. CCSDS 高级在轨系统及在我国航天器中的应用[J]. 航天器工程, 2003, 12(1): 12-18.
[9] 李佳航. 探析数据通信中的多路复用技术及其应用[J]. 网络安全技术与应用, 2013, 13(10): 42-43.
[10] 马西飞. CCSDS AOS 空间协议信道复用机制的 OPNET 仿真[D]. 上海: 上海交通大学, 2014.
[11] 赵运弢，潘成胜，田野. 长相关流量下的高级在轨系统帧复用效率研究[J]. 系统仿真学报，2013，25(5): 1130-1134.
[12] 赵运弢，潘成胜，田野，等. 一种有限缓存下的高级在轨多路复用等时帧生成模型[J]. 信息与控制，2010, 39(6): 68-72.
[13] 赵运弢，潘成胜，田野，等. 基于 CCSDS 高级在轨系统的 MPDU 复用效率研究[J]. 宇航学报. 2010, 31(4): 261-265.