聚合级 CGF 仿真中仿真应用程序的管理

谢永强，周保顺
（北京航空航天大学先进仿真技术航空科技重点实验室，北京 100083）

摘　要：对于大规模的聚合级 CGF 仿真，需要一定数量的计算机资源来实现，为了尽量减少占用的计算机资源和仿真中联邦成员的数量，也为了加强对这些资源的管理，针对具体的仿真应用，建立一个来管理聚合、解聚过程中需要的仿真程序的主控台是十分必要的，另外，为了加强对运行在所有机器之上的如此众多的**仿真应用程序的管理**，开发 **AST_SERVER** 和 **AST_CLIENT** 应用程序管理软件，可以提高聚合级 CGF 仿真项目的开发效率。

关键词：聚合级 CGF；应用程序管理；资源管理
中图分类号：TP391.9　　**文献标识码：**A　　**文章编号：**1004-731X (2006) S2-0293-03

Management of Applications in Aggregated Computer Generated Forces

XIE Yong-qiang, ZHOU Bao-shun
(Advanced Simulation Technology Key Lab, Beijing University of Aeronautics and Astronautics, Beijing 100083, China)

Abstract: A large number of computers are required in simulations of Aggregated Level Computer Generated Forces. Developing the Controlling Platform is essential ,not only for the purpose of reducing computer resource or the number of federates joined, but also for reinforcing the management of these resources, further more, for the purpose of reinforcing the *management of so many applications* running on all the computers. Developing application management software of AST_SERVER and AST_CLIENT can promote the efficiency of developing projects of aggregated level computer generated forces.

Key words: aggregated level computer generated forces; applications management; resource management

引　言

CGF(Computer Generated Force)分为平台级和聚合级两种。平台级CGF指的是每个仿真模型所描述的实体是单一的武器平台（如一辆坦克），主要用于小规模（比如营以下分队）的战术演练。但对于更大规模的分布交互仿真演练，比如多兵种，多实体参与的攻防对抗仿真系统，仅靠这些平台级的CGF实体是无法满足要求的。

究其原因主要有两个：

一是CGF模型本身的问题。平台级的CGF模型虽然进行了一定程度的简化，但如果一台计算机仿真太多的CGF实体，它本身的资源就会受到很大的限制，这样一来，要进行大规模的仿真，系统中就必须有大量的计算机节点。

二是网络的负载能力。由于网络带宽有限，实体节点过多必然导致网络上的信息数据量大增，使得信息传递延迟、数据丢失、误传的情况将不可避免地发生，进而会影响整个系统的运行效率，最终将会严重影响仿真结果的可信度。

聚合级CGF是通过对一定规模的作战单位（如坦克连）的作战行为进行足够的建模，使它在虚拟环境中不需要人的控制也能模拟完成与真实的作战单元相同的任务，主要用于分析与确认新型武器系统与新战术的有效性。因此，研究聚合级CGF具有非常重要的意义。

收稿日期：2006-05-10　　**修回日期：**2006-06-13
作者简介：谢永强(1981-)，男，山东，硕士，研究方向为分布交互仿真。

由陆装实验室分布仿真项目组和北京航空航天大学先进仿真技术实验室共同合作，开发了装甲兵聚合级CGF原型系统，初步实现了包括坦克师、团、营、连等多种建制的较大规模的仿真演练。兵力包括1个师、师中包含1个团、每团3个营、每营3个连，如果所有连级节点全部同时解聚会产生81个单车。另外，根据仿真要求还能进行规模的进一步扩展。要进行如此较大规模的仿真演练，就需要一定数量的机器资源来实现。如何对这些机器资源进行管理，以及如何对运行在这些机器之上的如此众多的仿真应用程序进行管理，是聚合级仿真中要解决的问题，针对前者开发的主控台基本实现了对现有资源的有效分配，针对后者开发了应用程序AST_SERVER和AST_CLIENT，它们分别运行在服务器端和各参与仿真的计算机上，不仅可用于聚合级仿真，也可用于节点较多的平台级仿真。

1　聚合级仿真中计算机资源的管理

1.1 应用背景

聚合级仿真与平台级仿真的一个显著区别就是参与仿真的节点更多，也更加复杂。比如一个聚合级仿真需要包括 1 个师、师中包含 1 个团、每团 3 个营、每营 3 个连，如果所有连级节点全部同时解聚会产生 81 个单车，由于每台计算机本身的性能不同，所以假设每台计算机运行 5 个结点，则需要 19 台计算机，此外仿真中还需要有 RTI 结点，二维态势显示节点，主控台、数据记录与回放等节点的加入，这就需要数目相当可观的计算机资源。但是考虑到仿真中当聚

合级结点解聚时，下级节点才开始参与此次仿真，如果聚合级节点未解聚，则它的下级节点都处于空闲状态，因此这些空闲的节点就可以作为与聚合级节点同样编制的节点解聚后的资源使用。但是要想使用同一批资源，这就要求两个聚合级节点不能同时处于解聚状态(以两个连共用同一批平台级资源为例，资源共享示意图见图 1)

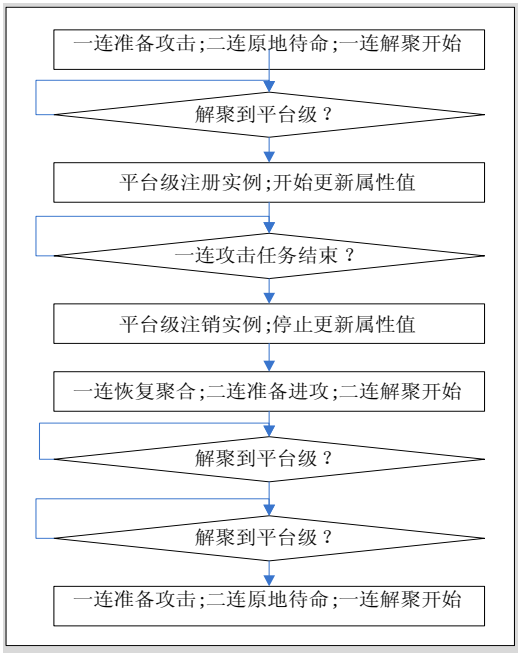

图 1 资源共享示意图

1.2 主控台程序的开发

一般在聚合级分布交互仿真中，存在师、团、营、连和单车等不同编制的实体，其中一般把师、团、营看作聚合级实体，存在于仿真的整个过程中，而连和单车则随着上级的解聚而产生，聚合而结束，所以它们的仿真程序存在着有实例运行和空闲两种状态。主控台程序的开发实质是为了重复利用连、单车等应用程序，以达到减少联邦成员个数，从而减少网络计算机数量的目的。

当仿真开始后，主控台程序的主要操作流程如下 (以主控台对连解聚实现为例，对营同理)

1、首先主控台要向平台级结点发送一个获取信息交互类 GetFederateInfo，平台级程序收到后将自己的 IP 地址 IPAddress，主机名 HostName，联邦成员名 FederateName 以交互类 FederateInfoRes 的形式发送给主控台。

2、当主控台从网络上收到解聚命令，并且目前有足够多的平台级程序处于空闲状态时，才允许解聚。然后从交互类 CurrentState 中将连级的 ID、连指挥车位置、速度和本连队形数据存储到本地，再调用 Init()函数根据位置和队形解算各个平台级单车的初始化位置、速度。

3、主控台再解析连长车的 ID，然后调用动态分配 ID 模块得到各平台级的 ID。

4、主控台以交互类 InitFederate 形式对平台级程序进行分配和初始化，然后向连长车发送解聚应答交互类，到此解聚过程结束。

5、当平台级实体被摧毁时，发送 InstanceLogout 交互类;当连级聚合时，连向平台级实体发送聚合交互类 CompanyAggOrder，如果平台级程序满足聚合条件则发送聚合应答交互到连长车，再发送实例注销 InstanceLogout 交互类到主控台，使主控台对该平台级程序的管理状态置为空闲状态。

2 仿真应用程序管理软件的开发

2.1 AST_SERVER 与 AST_CLIENT 的开发背景

聚合级仿真以及平台级仿真都是多联邦成员加入联邦后进行的仿真，节点众多，在演示过程中要进行一次仿真演练，就要许多人工手动进行的操作，比如启动应用程序、加入联邦，在仿真结束后还要执行退出联邦、关闭应用程序。由于聚合级仿真通常是几个部门联合进行的仿真，有的仿真程序的初始配置还需要进行更多的手动操作，不仅繁琐、费时，而且还容易出现操作错误，遗漏，有时多人同时来操作，导致加入顺序混乱，有可能使仿真结果不正确，有时还要重新操作，特别是在调试过程中，这种问题更是明显，直接影响了程序开发人员的情绪和开发进度，有时有的节点没有问题，也要负责维护这个节点的人员参与调试的工作，这样又导致了人力资源的浪费。为了解决以上问题，北航先进仿真技术实验室开发了 AST_SERVER 服务器端和 AST_CLIENT 客户端两个程序，以减轻仿真程序开发人员在调试和演示过程中大量重复性的工作，提高开发效率，使得程序开发人员不必头疼复杂的操作，只要简单的操作，就能执行一次仿真演练。(AST_SERVER 与 AST_CLIENT 程序在仿真网中分布图见图 2)

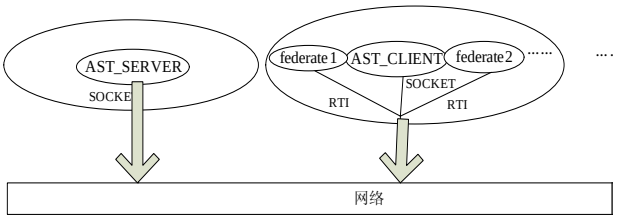

图 2 AST_SERVER 与 AST_CLIENT 分布图

2.2 AST_SERVER 的功能及实现

其中服务器端程序 **AST_SERVER** 运行在服务器上，通过 socket 编程实现，用来管理参与仿真运行的所有程序，然后统一设置所有程序的启动顺序和启动时间间隔。仿真开始前，各台机器上都运行一个客户端 AST_CLIENT，把本机要运行的联邦成员程序信息通知 AST_SERVER，然后 AST_SERVER 从联邦成员加入顺序文件中读取数据，对本次仿真中的所有客户端程序加入顺序进行排序，再通过 socket 向各台机器上运行的客户端程序 AST_CLIENT 发送启动应用程序命令，当仿真结束后，再向各个客户端程序发送一次

关闭所有应用程序的命令。(AST_SERVER 运行图见图 3)

图 3　AST_SERVER 运行图

2.3 AST_CLIENT 的功能及实现

每当计算机启动后，设置客户端程序 AST_CLIENT 为自动运行状态，然后和服务器进行 socket 连接，成功后，将本机所有的联邦成员信息发送到 AST_SERVER。如果参与仿真的联邦成员有变化，也可以更改后，再重新向 AST_SERVER 发数据进行更新。之后，等待 AST_SERVER 发来的启动命令。当客户端收到后，按照收到的启动顺序和时间延时计算启动该 程序的时间，然后到时启动仿真程序。另外，仿真程序也设置为启动自动加入联邦状态。当所有的仿真启动应用程序命令都发出去之后，大白方开始向各台机器发送仿真初始化、同步和开始等命令后仿真开始。当仿真结束时，AST_CLIENT 收到关闭应用程序命令，将所有的仿真程序关闭，仿真程序在关闭之前自动退出联邦，RTI 最后一个被关闭。

2.4 AST_SERVER 与仿真应用程序之间的通讯

把 AST_SERVER 作为服务器端，当仿真应用程序启动后，作为程序客户端，在仿真应用程序中也加入简单的 socket 编程模块，当仿真应用程序启动后保持与服务器程序 AST_SERVER 的连接。为了减少网络上的数据量，仿真应用程序每 0.2 秒向 AST _SERVER 发送一个联邦成员当前的状态数据，如果 AST_SERVER 1 秒内没有收到仿真应用程序发来的数据，则表示当前应用程序处于故障状态，或处于死循环、死机状态。服务器端 AST_SERVER 程序会显示所有这些程序的运行状态。

2.5 程序的自动运行和状态的保存

为了最大程度的减轻操作人员的负担，在开发过程中进行了如下操作：

在 AST_CLIENT 端，我们保存了要参与仿真运行的应用程序的路径和名称，不需要每次都进行设置，但可以按照需要进行更改，在 AST_SERVER 端，用文件的方式保存了此次仿真中需要的联邦成员的名称和启动顺序，每次当 AST_CLIENT 与 AST_SERVER 进行连接后，将数据发给 AST_SERVER。AST_SERVER 可以根据文件中的数据对所有的联邦成员的启动顺序和延时时间进行自动设置，操作非常简便。

3　结论

在聚合级仿真项目的开发过程中，主控台程序对聚合解聚过程的仿真起到了重要作用，能够准确的进行仿真程序资源的动态分配和回收再利用，极大的减少了仿真中需要的计算机数量。另外，AST_SERVER 与 AST_CLIENT 自动管理程序的开发，较好的实现了大规模聚合级仿真程序的管理，也可以用在联邦成员较多的平台级仿真中，从而提高仿真项目的开发效率。

参考文献：

[1] 王杏林，郭齐胜，徐如燕，杨立功等. 基于多 agent 的聚合级 CGF 系统的体系结构研究[J]. 计算机工程与应用, 2001，19: 1-2.
[2] 柏彦奇，龚传信等. 聚合级分布交互式作战仿真体系结构研究[J]. 计算机仿真, 1999, 4: 1-2.
[3] 郭齐胜，杨立功，徐如燕，王杏林等. 基于 HLA 的聚合级 CGF 初探 [J]. 计算机工程与应用, 2001.
[4] 庞国峰. 分布式虚拟战场环境中计算机生成兵力系统的研究[D]. 北京航空航天大学, 2001.
[5] 博嘉科技编写. Visual C++6.0/Internet Programming Instance 网络编程实作教程［M］. 北京: 希望电子出版社, 2001.