一种环境理解的分布式群体仿真任务划分方法

周文平,唐好选,季振洲
(哈尔滨工业大学计算机科学与技术学院,黑龙江哈尔滨 150001)

摘 要: 本文通过引入环境结构因素,提出了一种适用于多层次复杂环境的自适应任务划分算法.自动读取场景模型并通过理解转换为连通邻接区域集,然后对区域进行快速粗粒度划分,有效提高划分性能;自然消除了被障碍隔离的相邻区域个体间的感知计算,大大减少了节点间通信量,使之更适合于大规模群体仿真应用.实验结果表明该算法的划分代价和执行性能均较优.文中设计了一种适合该划分算法的分布式仿真模型,基于该模型的分布式系统对室内多层楼宇或室外场景大规模群体仿真均具有较高仿真性能,相同规模群体的仿真性能与仿真节点数成线性关系表明系统具有良好的可扩展性.

关键词: 分布式仿真; 群体仿真; 任务划分; 图形处理单元
中图分类号: TP391 **文献标识码:** A **文章编号:** 0372-2112 (2014)12-2448-09
电子学报 URL: http://www.ejournal.org.cn **DOI:** 10.3969/j.issn.0372-2112.2014.12.017

An Environment Structure Understanding Task Partition Method for Distributed Crowd Simulation

ZHOU Wen-ping, TANG Hao-xuan, JI Zhen-zhou
(*School of Computer Science and Technology, Harbin Institute of Technology, Harbin, Heilongjiang* 150001, *China*)

Abstract: The paper proposed an adaptive partition method considering environmental structural factors, which is applicable for multilayered complex environment. Simulation scene is automatically extracted to walkable adjacent areas, and a coarse granularity partition based on regions is applied to get shorter execution time. The inter-individual perceptual computing of any two individuals separated by obstacle between two adjacent regions is negligible, so it efficiently reduces the inter-node communication cost and makes the algorithm more suitable for large scale crowd simulation. The results show the proposed algorithm gets lower cost and higher performance. An efficient distributed simulation model is designed for the partition method, and a distributed system based on the model gets higher simulation performance on both inner door and out door scene. The performance of system with the same crowd size linearly increases with the increase of compute nodes, which proves high scalability of the system.

Key words: distributed simulation; crowd simulation; task partition; graphic processing unit (GPU)

1 引言

兼顾行为真实性和仿真实时性是大规模群体仿真面临的难题,采用混合并行方式解决此问题是并行人群仿真的重要研究分支[1].群体仿真研究可基于 Agent 或流体进行[2,3],基于 Agent 的群体能够更丰富的表达个体特性和行为方式,相关 AI 算法亦可提高 Agent 的智能[4].因此本文采用基于 Agent 的方式,通过混合并行方式提高仿真性能和扩展性.

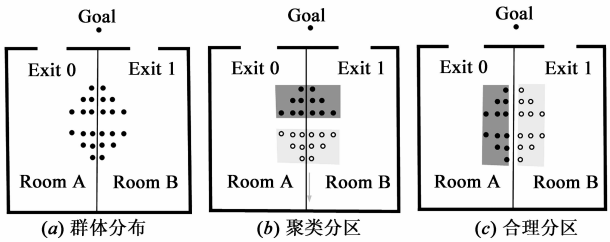
并行人群仿真的关键问题是任务划分,当前研究存在两方面不足:

(1)人群仿真通常与特定场景相关联,如逃生模型和室内环境密切相关,虚拟战场和户外地形相关等.当前研究常忽略环境这一重要因素,以开放无障碍环境作为研究假设,常以平面的欧式距离作为算法基础[2,5],不适合多层次复杂环境.

(2)划分代价[5]问题.对于图 1(a)表示的初始群体分布,因未考虑环境结构因素,聚类算法[2,6]得到的任务划分结果如图 1(b)所示,群体被分割成上下两组,其通讯代价为两个分区邻近边界的个体数加权值远大于 0.由于被障碍物隔离的个体之间感知关联较弱(或可忽略),按照图 1(c)将群体沿墙体左右分区显得更为合理,由于障碍隔离,两分区邻近边界的个体间通讯被忽略,故而通讯代价趋近于 0.在紧急逃生的室内虚拟环

收稿日期:2014-02-10;修回日期:2014-08-31;责任编辑:覃怀银
基金项目:国家自然科学基金(No.61175027)

境场景中,这种现象尤为突出.



(*a*) 群体分布　(*b*) 聚类分区　(*c*) 合理分区

图1　仿真环境对分区算法的约束

由此可知,环境结构因素是任务划分过程不可或缺的关键因素之一.引入环境结构因素,将问题空间映射到按区域进行划分的过程域,以此设计出的算法既能满足多层次复杂环境任务划分的要求,同时又能获得更好的分区代价.

2　相关工作

任务划分算法主要包括两大类:基于 Agent 个体分组的划分算法和基于区域分割的划分算法.

在基于 Agent 个体分组的划分算法研究方面:

最简单直接的方法是随机个体划分方法(Random Division of Agents,RDA)[7],采用 Agent 个体数目随机均分的方式进行任务划分.该方法主要优点实现简单,不足是各分组的 Agent 分布散乱,很多近邻关联的 Agent 分布在不同的计算节点使得各计算节点间的通信代价很高.

其次是基于图划分的方法(RBP/LP/CRP)[8],算法的主要思想是通过 Agent 个体感知关联生成无向图,然后对图进行划分.其中二分迭代分区法(RBP 算法)的时间复杂度为 $O(N^3(P-1))$,这使得该方法的执行性能较低.

还有一种是基于聚类的任务划分方法(K-means Cluster Algorithm,KCA)[6],该算法通过各 Agent 顶点平面距离的 K-Means 聚类实现.此方法以欧式距离作为聚类条件,在复杂环境(如楼层内)中,紧贴墙的对立面的个体间欧式距离很近,却因被墙体阻碍,其实际距离却很远(可能需绕过门到达墙的对面侧),因此欧式直线距离并不能真实反映个体之间近邻关系,造成聚类结果的巨大偏差,导致划分达不到预想的结果.

在基于区域分割的任务划分方法研究方面:

最常规的方法是按状态划分方法(Division By State,DBS),即针对 K 个计算节点将虚拟环境划分为 K 个区域,每个区域单独划分到一个结算节点.Zhou B 等[9]将虚拟环境划分成连续的区域,实现了基于分组行为的群体分布式仿真.Lozano M 等[10]提出一种基于遗传算法的任务划分方法.文献[5]对比研究了各 DBS 方法,总结出 QHull 凸包方法迄今最优.DBS 方法较个体分组的划分方法在性能和划分代价都有较大提升,其不足在于以平面欧式距离为划分基础不适合多层次复杂环境,同时忽略环境障碍对通信关联的影响,使得分区通信代价很高.

其次是均匀网格的区域划分方法,通过将平面空间网格化,然后对网格单元随机均匀分配进行任务划分.Quinn 等[11]采用此方法研究了社会力模型的并行实现,采用主/从方式协同计算,每个从节点负责相同数量的网格.Reynolds[12]使用类似 Quinn 的方法,在性能上有所提升,其不足在于对网格单元的随机均匀分配,会导致与 RDA 方法类似的缺陷.

近期提出的网格聚类算法的区域划分方法(Grid-based Partition Algorithm,GPA)[2],是对文献[6]个体分组的聚类算法研究工作的延续.该方法增加了群组因素的计算,对群组模型的群体仿真具有较好的应用效果,其优点是预先对环境进行网格化,以网格单元进行粗粒度划分,有效提高了算法性能.不足之一在于只针对平面环境,不适用于多层次复杂环境;其另一不足和文献[6]中 KCA 算法一样,在非开放场景,由于环境障碍,欧式距离并不能真实反映个体之间近邻关系,造成聚类划分达不到预想的结果.

本文提出方法与上述方法相比主要有两点改进:(1)分区结果自适应生成,并能够有效应用于多层次复杂环境;(2)引入环境因素消除被障碍隔离的相邻区域个体间的感知计算,能够获得更好的划分代价.

此外,国内毛天露等[3]提出了一种基于语义环境的任务划分算法.该算法适用于多层次复杂场景和基于流的群体仿真,但需要特殊建模工具为虚拟环境添加语义信息,任务划分中需要人为调整过渡区域来实现对通用区域的划分,且未涉及负载平衡及分区间通讯量等相关描述.

3　算法总体目标

本算法主要实现如下目标:(1)可保持每次递归运算过程中各个计算节点负载均衡;(2)与两个或两个以上区域存在感知交叉关联的个体数目最少;(3)更新周期小于 250ms,仿真渲染频率大于 30fps.

3.1　任务划分的问题定义

基于环境理解的任务划分问题可定义为:群体中数目为 M 的个体集合 Agents 定义为 $\text{Agents}=\{\text{Agents}_i \mid i=1,\cdots,M\}$,虚拟环境按自然结构被算法自动理解并划分为邻接 Cell 区域集合 Cells 定义为 $\text{Cells}=\{\text{Cells} \mid i=1,\cdots,N\}$,任务划分的目标为将区域集合中的 N 个 Cell 区域划分到具有 K 个计算节点的分布式机群 $\text{Servers}=\{\text{Server}_l \mid l=1,\cdots,K\}$ 中,使得各计算节点内的 Cell 区域包含的 Agent 个体总数趋于平衡且计算节点之间消息

通讯量达到最小.

3.2 任务划分的优化目标

把区域集合 Cells 中的 Cell_i 设置为顶点,为相邻接区域之间添加一条无向边,可将区域集合转化为邻接区域无向连通图(adjacent regions undirected connected graph),定义为 ARUCG(Cells, E).图中各顶点的权值大小设置为区域中包含的 Agent 个体数,表示区域负载.顶点间的边权值设置为相邻区域间有感知关联的个体数目,表示相邻区域间需要交互的通信量.相关定义如下:

定义 1 划分数组 P:划分结果采用一个长度为 N 的数组 P 表示,对每一个 $\text{Cell}_i \in \text{Cells}$, $P[\text{Cell}_i]$是一个 1 到 K 的整数,表明 Cell_i 属于哪个划分.

定义 2 ARUCG(Cells, E)的划分子图为:$\text{ARUCG}_l(\text{Cells}_l, E_l)$, $l = 1, \cdots, L$,则对于 Cells_l 中的任一点 Cell_i 有 $P[\text{Cell}_i] = l$,若 $e(u,v) \in E_l$,则有 $P[u] = P[v] = l$.

定义 3 划分 P 的边界边集合定义为:$\text{EB}_p = \{e(u,v) \mid u \in \text{ARUCG}_{\text{li}} \wedge v \in \text{ARUCG}_{\text{lj}} \wedge \text{li} \neq \text{lj}\}$.

依定义,最终提出 P 划分的优化目标函数为式(1):

$$\begin{cases} |\text{Cells}_{\text{li}}| \approx |\text{Cells}_{\text{lj}}|, \text{li} = 1, \cdots, L \wedge \text{lj} = 1, \cdots, L \wedge \text{li} \neq \text{lj} \\ \text{cost}_P = \sum e(u,v), \quad e(u,v) \in \text{EB}_P \\ \min(\text{cost}_P) \end{cases} \tag{1}$$

以下通过两个具体仿真场景阐述基于任务划分的分布式群体仿真问题.

场景 1 室内紧急逃生场景

室内环境场景可抽象为区域集合 Cells,邻接 Cell 区域之间的入口集合定义为 $\text{Portals} = \{\text{Portal}_i \mid i = 1, \cdots, p\}$ 以及疏散的可用逃生出口集合定义为 $\text{Exits} = \{\text{Exit}_i \mid i = 1, \cdots, e\}$,我们用 $\text{Sum}[\text{Cell}_i]$表示 Cell_i 中的 Agent 个体数,则区域集合所对应的个体数集合为 $\text{AgentsCountSet} = \{\text{Sum}(\text{Cell}_i) \mid i = 1, \cdots, n\}$.

个体逃生任务可以设定为$\langle \text{Agent}_i, \text{Exit}_j \rangle$,即 Agent_i 朝着最近的逃生出口 Exit_j 方向紧急逃离.仿真的初始状态为人群随机分布在各区域内,则 Cell_i 的负载即为 $\text{Sum}[\text{Cell}_i] = \text{Random}()$,所有区域的总负载用函数 g 表示,见式(2),仿真的终止条件为各区域的人员均已疏散完毕,也即 $g = 0$.

$$g = \sum_{i=1}^{n} \text{Sum}(\text{Cell}_i) \tag{2}$$

任务划分是将任务$\langle \text{Agent}_i, \text{Exit}_j \rangle$划归到各计算节点 Server_l 中的过程.根据定义 2 可知 Cells_l 为划分区域子集,Cells_l 各区域的个体任务将分配给 Server_l 计算节点,则 Server_l 的负载可由函数 L_{inner}求得,见式(3).

$$L_{\text{inner}}(\text{Cells}_l) = \sum_{\text{Cell}_i \in \text{Cells}_l} \text{Sum}(\text{Cell}_i) \tag{3}$$

则 Cells_l 中各区域包含的所有 Agent 个体集合定义为 $\text{Agents}_l = \{\text{Agent}_i \mid \text{Pos}(\text{Agent}_i) \text{ in } \text{Cell}_j \wedge \text{Cell}_j \in \text{Cells}_l\}$,其中 $\text{Pos}(\text{Agent}_i)$为 Agent_i 的空间位置坐标,最终所得 Server_l 分配任务可表示为式(4).

$$\text{Task}_l(\text{inner}) = \{\langle \text{Agent}_i, \text{Exit}_j \rangle \mid \text{Agent}_i \in \text{Agents}_l, \text{Exit}_j \in \text{Exits}\} \tag{4}$$

场景 2 室外群体漫游场景

室外环境场景可抽象为区域集合 Cells,邻接区域之间的入口集合 Portals 以及预先指定的漫游目标集合定义为 $\text{Aims} = \{\text{Aim}_j \mid j = 1, \cdots, \alpha\}$,区域集合所对应的个体数集合为 AgentsCountSet.个体漫游任务可以指定为$\langle \text{Agent}_i, \text{Aim}_j \rangle$,即 Agent_i 朝着指定的目的地 Aim_j 方向漫游前行.仿真的初始状态为人群随机分布在区域内,即 $\text{Sum}[\text{Cell}_i] = \text{Random}()$,仿真的终止条件为群体各成员均已到达目标地址,即为式(5)

$$|\text{Pos}[\text{Agent}_i] - \text{Pos}[\text{Aim}_j]| < \varepsilon \tag{5}$$

其中 $\text{Pos}[\text{Aim}_j]$为目标地址 Aim_j 的空间位置坐标,收敛值 ε 为指定的 Agent_i 到达目标地点 Aim_j 的距离最小阈值.

任务划分是将任务$\langle \text{Agent}_i, \text{Aim}_j \rangle$划归到各计算节点 Server_l 中的过程.Server_l 的负载可由函数 L_{outer}求得,见式(6).

$$L_{\text{outer}}(\text{Cells}_l) = \sum_{\text{Cell}_i \in \text{Cells}_l} \text{Sum}(\text{Cell}_i) \tag{6}$$

最终所得 Server_l 分配任务可表示为式(7).

$$\text{Task}_l(\text{outer}) = \{\langle \text{Agent}_i, \text{Aim}_j \rangle \mid \text{Agent}_i \in \text{Agents}_l, \text{Aim}_j \in \text{Aims}\} \tag{7}$$

4 算法描述

本节主要描述基于环境理解的任务划分算法(Environment Structure Unstanding Task Partition, EnvSUTP),EnvSUTP 算法在构建 CPGs (Cells and Portal Graphs)图的基础上,针对 Cells 进行任务划分.算法的基本输入为三维场景,人群在场景的分布,算法的最终输出结果为场景的区域集合 Cells 及区域集合的划分 P.算法流程为:

(a)首先采用 CPGs 方法对三维场景 S_{3D}进行结构提取,获得组成场景的 Cell 区域集合;

(b)通过 Cell 区域间邻接关系构建邻接区域无向图 ARUCG(Cells, E);

(c)随机生成仿真人群,通过人群分布状态对 ARUCG(Cells, E)的各顶点值和边权值进行求解;

(d)按优化目标函数对带权无向图进行 P 划分.

4.1 环境结构提取

将场景分割成 Cells-and-Portal 的 CPGs 方法常被用于人群仿真的环境处理过程中[13~16].通过一种体元化

的 CPGs 算法来完成环境的几何提取及自动化结构理解,可快速建立体元与 Cell 区域之间的对应关系,方便个体与环境的频繁交互.

首先,从任意场景模型文件中提取环境的空间几何信息,用〈BBox, nVertex, Vertexs, nTris, Tris〉五元组表示,其中 BBox 为顶点集的最小包围盒,nVertex 为顶点数,Vertexs 为顶点坐标集,nTris 为三角面片数,Tris 为三角面片顶点索引集.

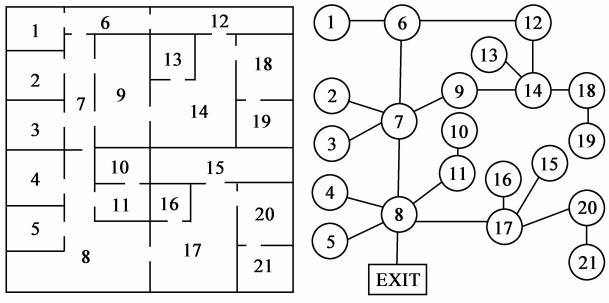
其次,对五元组体素化,获取场景的体元三维数组由 VoxelsGrid[WIDTH][LENGTH][HEIGHT]表示.通过将三角面片体素化映射到体元,建立环境和体元之间的关联关系.映射的精度由预设的体元单位大小决定.

然后,标示边界体元并构建各体元到边界距离场.

最后,通过分水岭算法(WaterShed Transform)对距离场进行处理,得到 Cells.

4.2　构建邻接区域无向图

在任务划分过程中需要实时记录邻接区域集合中各 Cell 区域包含的个体群,以及相邻 Cell 区域入口处具有相互感知关联的个体数目,为此需要实现环境的管理与群体状态的维护.图 2(a)显示的是一个示例室内环境的平面图.



(a) 室内示例场景平面图　　(b) 邻接区域无向图

图2　从场景平面图提取无向图

针对图 2(a)的示例场景,通过 4.1 节方法求得虚拟环境被划分为 Cells.每个 Cell 区域包含特定的个体群,维持邻接 Cell 区域的相关信息.其中 Cell 区域由多边形表示,Portal 入口由边表示.通过计算可得 Cell 区域的通路连接关系,用邻接区域表存储,以此构建的邻接区域无向图如图 2(b)所示.

4.3　群体管理及无向图权值维护

通过无向图分割方式进行任务划分,需计算无向图的顶点值和边权值.人群分布状态随仿真过程动态演进,因此需对无向图权值进行动态更新维护.每个个体在某一时刻唯一归属于一个 Cell 区域,通过判定 Cell 区域与个体的空间包含关系,可将个体划分到对应的 Cell 区域,具体由算法 1 实现,其中 CellIDs[nAgents]记录个体的区域编号索引列表.

算法 1　人群状态更新

```
输入:Agents,Cells,VoxelsGrid[WIDTH][LENGTH][HEIGHT]
输出:CellIDs[nAgents]
1.FOR i from 1 to N step = 1
2.    CellIDs[i] ← -1;
3.END FOR
4.FOR each Agent_i ∈ Agents DO
5.    int vid←calVoxelID(Pos[Agent_i],VoxelsGrid);
6.    int cid←getCellID(vid);
7.    CellIDs[i] ←cid;
8.END FOR
```

4.3.1　Cell 区域内部个体数计算

ARUCG(Cells, E)的顶点权值表示为区域中个体的数目,决定区域内共需消耗的计算量大小.通过算法 2 求解,可得每 Cell 区域包含 Agent 个体数.首先将 AgentsCountSet 数组置零,然后遍历个体计数.

算法 2　Cell 区域个体计数

```
输入:CellIDs[nAgents]
输出:AgentsCountSet
1.FOR i from 1 to N step = 1
2.    AgentsCountSet[i]←0;
3.END FOR
4.FOR i from 1 to N step = 1
5.    cid←CellIDs[i];
6.    AgentsCountSet[cid] + + ;
7.END FOR
```

4.3.2　相邻区域间感知关联个体数计算

ARUCG(Cells, E)的边权值表示为相邻 Cell 区域间感知关联个体数,决定了分区之后区域间的通信量大小.设 Agent 的感知半径为 R,仅需对所有 Portal 入口边的 $\pm 2R$ 范围内的 Agent 进行关联判定,从而求得分区之间关联个体的数目.

如图 3 所示,两相邻 Cell 区域 $ABCD$ 与 $CDEFG$ 之间

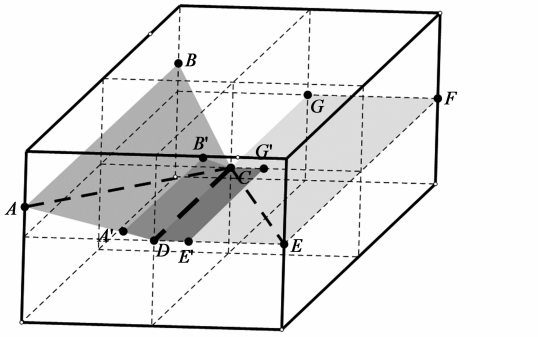


图3　Portal近邻关联

的 Portal 入口为 CD 边.相邻区域并不一定在同一平面,我们设定 $A'B'CD$ 为 Cell 区域内的临界区域,$CDE'G'$ 为 Cell 区域 $CDEFG$ 内的临界区域,两个区域合并为沿 Portal 边关联的临界区域.由于 Cell 区域为空间多边形构成,所有顶点不能保证在同一平面,所以可按如下规则进行临界区域求解:以 Portal 边 CD 及同多边形区域内距离 C 或 D 点最近的多边形顶点 A 为平面,以 D 为起点,正向为区域内侧方向求法向,D 点沿法向 $2R$ 的距离处即为 A',C 点沿法向 $2R$ 的距离处即为 B',同理可求 E',G'.

令 eEdges[nPortals]为由入口连接的相邻区域之间有感知关联的个体数列表,lCID,rCID 为入口连接的相邻两区域的编号索引,lbox,rbox 为两区域的临界区域包围盒,lvoxels,rvoxels 分别为临界区域映射的体元列表.通过算法 3 计算所得相邻区域个体关联数即为无向图的边权值.

4.4 无向图 P 划分

为了使划分结果达到 3.2 节的优化目标,可采用文献[17]提出的算法对无向图进行划分求解,得到满足负载均衡和分区间通信量最小化的划分结果为 $P[\text{Cell}_i]$.

5 分布式仿真应用

根据 EnvSUTP 算法的特点,设计出一种合适的分布式群体仿真模型.

5.1 分布式仿真模型

分布式群体仿真包含三类节点 CC(Control Center)主控节点,CN(Compute Node)计算节点,Viewer 视景节点.系统按 $P[\text{Cell}_i]$结果分配各区域给各 CN 节点,在划分过程产生的每一条分割边表示该边相邻两个 CN 节点之间存在通信.仿真系统结构图如图 4 所示.

算法 3 计算相邻区域个体关联数

```
输入:Agents,Cells,VoxelsGrid[WIDTH][LENGTH][HEIGHT],Portals
输出:eEdges[nPortals]
1.FOR i from 1 to nPortals step = 1
2.    eEdges[i]←0;
3.END FOR
4.FOR eachPortal_i∈ Portals DO
5.    int lCID←getLCellID(Portal_i);
6.    int rCID←getRCellID(Portal_i);
7.    float* lbox←getLBox(lCID,Portal_i,2R);
8.    float* rbox←getLBox(rCID,Portal_i,2R);
9.    voxel* lvoxels←calVoxels(lbox,VoxelsGrid);
10.   voxel* rvoxels←calVoxels(rbox,VoxelsGrid);
11.   voxels←lvoxels ∪ rvoxels;
12.   FOR each voxel_j∈ voxels DO
13.      Agent * agents←findAgents(voxel_j);
14.      eEdges[i]←eEdges[i] + sum(agents)
15.   END FOR
16.END FOR
```

CC 节点主要功能包括三维场景的 CPGs 图构建,环境管理与群体管理、任务划分、维护每个 CN 节点的区域列表以及各区域所包含个体列表及运行过程中负载失衡后的任务重新调度.

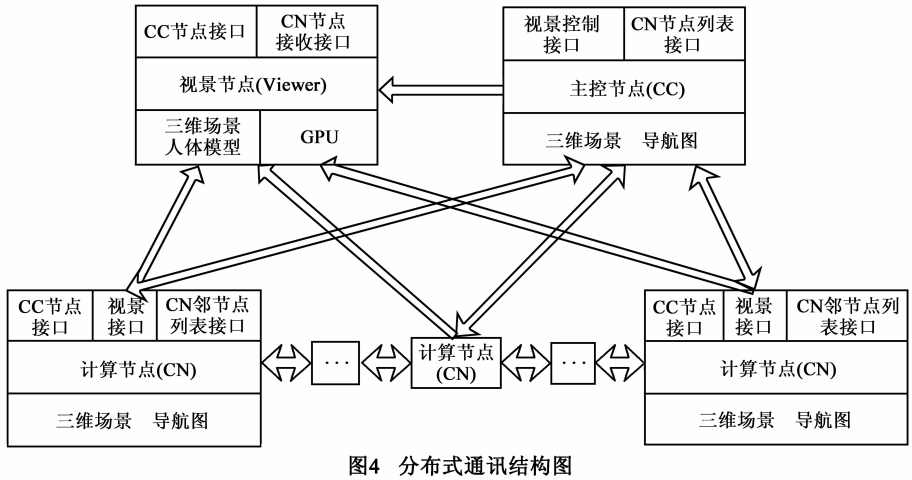


图4 分布式通讯结构图

CN 节点主要负责所分配的分区区域内个体的路径导航、行为运算及个体运动状态实时更新等功能.各 CN 节点接收来自 CC 主控节点分配的仿真任务,同时将运算状态反馈给主控节点,并将个体状态信息传递给 Viewer 节点进行渲染显示.CN 节点通过 Portal 入口的列表对应相邻 CN 节点列表维持彼此通讯.

Viewer 视景节点,实时采集来至 CC 节点的控制指令,收集各 CN 节点中个体的状态信息并对个体进行渲染.通过文献[18]中提出的一种基于运动捕捉人体动画的 GPU 几何实例化算法来实现群体的动画渲染.

5.2 潜在问题分析

任务划分过程可能存在两个潜在的问题,会严重的影响到分区后各运算节点的负载平衡.

问题 1　当 Cell 区域数小于 CN 计算节点个数时($n < k$),造成个别 CN 计算节点空负载运行.

可依次对包含最大个体数的 Cell 区域进行分割处理,以此为原则设计出算法 4 来完成相关工作.在 ARUCG(Cells, E)图中,每个区域都分配有唯一识别的编号,假设 RegIdx[n]为区域的编号索引,RegAgts[n]为相应区域包含的个体数目.可依次对最大数目 Cell 区域进行二分分割,最终可得合理的区域数.

算法 4　Cell 区域数小于计算节点数处理算法

输入:Agents, RegIdx[n], RegAgts[n], n, k
输出:RegIdx[n], RegAgts[n], n
1. WHILE($n < k$)
2.　按各环境区域包含个体的数目由高到低排序,RegDesc[n]←Sort(RegAgts[n]),同时更新索引 IdxTmp[n]←update(RegIdx[n]),使得 IdxTmp[i]与 RegDesc[i]一一对应;
3.　针对最大数目的区域 IdxTmp[0]的包围盒,采用二分分割方法将长边分割成两部分,此时增加的分割线段设置为新区域的 Portal 入口边,IdxTmp[0]裂变成两个新区域 Reg$A$, Reg$B$.将 Reg$A$ 的索引编号 Idx$A$ 设置为 IdxTmp[0],为 Reg$B$ 新建索引编号 Idx$B$ 为 $n+1$,同时统计各分割区域的个体数 Num$A$, Num$B$;
4.　分别移除 IdxTmp[0], RegDesc[0],同时使用插入排序,将 Num$A$, Num$B$ 降序插入到 RegDesc[$n+1$]当中,将 Idx$A$, Idx$B$ 插入 IdxTmp[$n+1$]对应的位置;
5.　RegIdx[n]←IdxTmp[n], RegAgts[n]←RegDesc[n];
6.　n←$n+1$,
7. END WHILE;
8. RETURN n, RegIdx[n], RegAgts[n]

问题 2　当 Cell 区域数大于服务器数时($n > k$),若存在 Cell 区域包含的个体数大于每个服务器应分得平均个体数时($\max_{i=0}^{n}(|\text{Cells}_i|) > (\sum_{i=0}^{n}|\text{Cells}_i|/k)$),会自然造成负载失衡.

对此类区域依次进行分割处理,为此设计了算法 5 进行求解.

算法 5　Cell 区域个体数大于计算节点平均个体数处理算法

输入:RegIdx[n], RegAgts[n], n, k
输出:RegIdx[n], RegAgts[n], n
1. MaxNum←max(RegAgts[n])), AvgNum←sum(RegAgts[n])/k;
2. WHILE(MaxNum > AvgNum)
3.　已知最大数目的区域数组偏移为 offset,通过对 RegAgts[offset]的包围盒采用二分分割方法将长边分割成两部分.此时分割的线段增加为 portal 入口边,生成两个区域 Reg$A$, Reg$B$,将 Reg$A$ 的索引编号 Idx$A$ 设置为 RegIdx[offset],为 Reg$B$ 新建索引编号 Idx$B$ 为 $n+1$,同时统计各分割区域的个体数 Num$A$, Num$B$;
4.　设置 RegIdx[offset]←Idx$A$, RegAgts[offset]←Num$A$,数组末尾分别插入 Reg$B$, RegIdx[$n+1$]←Idx$B$, RegAgts[$n+1$]←Num$B$;
5.　MaxNum←max(RegAgts[n]);
6.　n←$n+1$;
7. END WHILE
8. RETURN RegIdx[n], RegAgts[n], n

在仿真时,如两个问题在一次仿真中同时出现,则可依次按序运行以上两个算法即可得到所需结果.

6 实验与讨论

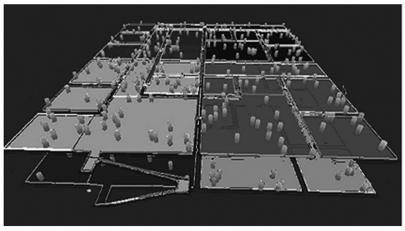
本节对 EnvSUTP 算法及分布式仿真模型进行仿真实验及相关性能评估.性能评估指标主要包含两类:延迟和吞吐率.参照文献[19]可设置最大可接受响应时间为 250ms,设置最低仿真动画帧率为 30fps(可接受的下限).采用的仿真平台为一组 9 台 Intel 平台机器(至强 4 核@2.50GHz, 3.8GB 内存, NVIDIA Quadro FX 580 显卡),其中一台作为 CC 主控节点,其余 8 台作为 CN 运算节点,一台 Intel 机器(三代 i5, 4 核@3.40GHz, GTX 760 显卡)作为 Viewer 群体动画渲染节点.所有机器的操作系统均采用内核版本为 2.6.32 的 Linux 系统,机组采用千兆路由互联组成内部局域网.

实验 1　分布式仿真实验及性能分析

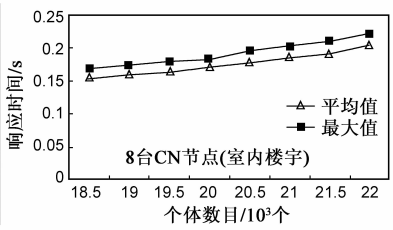
针对多层楼宇的室内环境,以 2 层楼为例,8 个 CN 计算节点的划分结果如图 5(a)所示,相同灰度填充的 Cell 划分到同一个计算节点,仿真个体采用随机均匀分布的方式自动生成.逃生个体采用 RVO(Relative Velocity Obstacle)速度障碍规则进行避碰处理.不同个体数在相同数目运算节点上的响应时间关系如图 5(b)所示,结果显示仿真性能与人群规模基本成平滑的线性关系.图中 2.2 万仿真个体在 8 台 CN 计算节点的平均响应时间接近 200ms,符合最大可接受响应时间 250ms,表明仿真具有较高性能.相同个体数在不同数目的运算节点上的响应时间关系如图 5(c)所示,结果显示相同规模人群的仿真性能与计算节点数成近线性关系,证明系统具有良好的可扩展性.

对于室外场景的分布式群体仿真,图 5(d)为城镇的人群日常活动的仿真结果,个体采用 RVO 速度障碍规则进行避碰处理,不同个体数在相同数目运算节点上的响应时间关系如图 5(d)所示,结果显示仿真性能与人群规模基本成平滑的线性关系.图中 2.2 万仿真个体在 8 台 CN 计算节点的平均响应时间介于 200ms ~ 250ms 之间,符合最大可接受响应时间约束,表明仿真依旧具有较高性能.对比图 5(e)和图 5(b),可知室外场景响应时间均略高于室内场景,是由于室外场景规

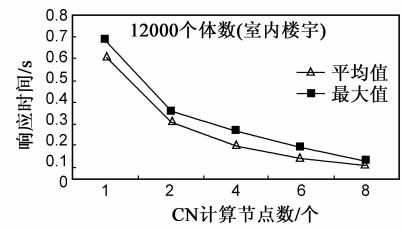
模大于室内场景,使得 Cell 区域相对较多,从而略有增加路径导航所需计算时间所致.相同个体数在不同数目运算节点上的响应时间关系如图 5(f)所示,结果显示相同规模人群的仿真性能与计算节点数成近线性关系,证明系统对室外环境仿真同样具有良好的可扩展性.



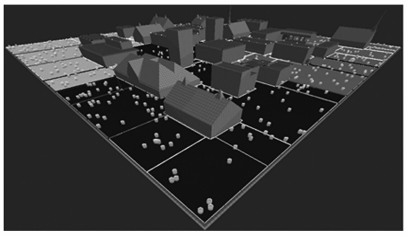
(a) 多楼层室内场景8分分区图



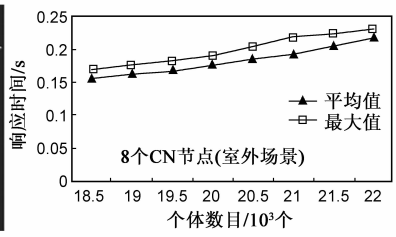
(b) 室内场景不同人群规模仿真响应时间



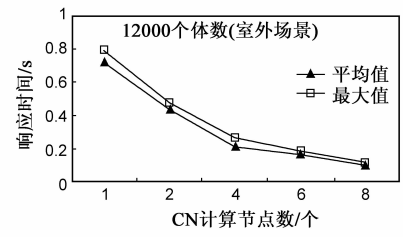
(c) 室内场景不同CN节点数仿真响应时间



(d) 室外场景仿真4分分区图



(e) 室外场景不同人群规模仿真响应时间



(f) 室外场景不同CN节点数仿真响应时间

图5 分布式仿真实验

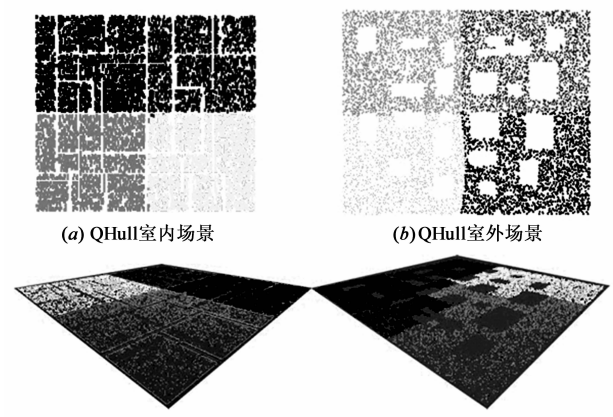
实验 2 与 QHull 算法对比分析

文献[5]对 GA 遗传算法、R-Tree 算法和 QHull 凸包算法等三种典型的基于平面区域划分算法进行了对比研究,得出 QHull 算法在任务划分代价和性能各方面表现最优.通过实现 QHull 算法与 EnvSUTP 算法进行对比.分别在图 5(a)的单层楼室内环境(因 QHull 不适用于多层次复杂环境而未使用多层楼宇)和图 5(d)的室外环境中随机生成 10000 规模的人群,通过 EnvSUTP 算法进行划分并求得划分代价值,再将生成的两组个体顶点坐标集导出文件后,通过 QHull 划分算法分别对各组导出顶点集进行划分并求值.划分代价采用与文献[10]一致的计算方式,见式(8),其中 ω_1,ω_2 权值均设置为 0.5.

$$H(P)=\omega_1\cdot\alpha(P)+\omega_2\cdot\beta(P),\omega_1+\omega_2=1 \qquad (8)$$

式中 P 为一种分区方案,$\alpha(P)$为分区方案 P 中所有的感知区域与两个及两个以上的分区区域有交叉的 Agent 的总和,$\beta(P)$为每个分区区域 Agent 数与分区区域平均 Agent 数之间的标准差,ω_1,ω_2 分别为各自的权重.

两种算法在 4 个 CN 计算节点的划分效果如图 6 所示.图 6(a)与 6(c)的室内分区对比中,6(c)中完全按房间区域结构规则划分,而 6(a)中右下所在房间区域左上角与左下角混入属于其他分区的个体.且 6(c)相邻区域共享边为两者之间的门入口边,总体较短,6(a)因未考虑环境结构,分区之间沿墙体均视为共享边,共享个体数目远多于 6(c).图 6(b)与 6(d)的室外场景分区对比中,可以看到 EnvSUTP 算法所得结果 6(d)中能够很好利用环境隔离的作用,使得相邻分区之间的共享边长达到最小,而 6(b)中左右两边分区沿共享边均布满个体.



(a) QHull室内场景 (b)QHull室外场景

(c) 本文算法室内场景 (d) 本文算法室外场景

图6 任务划分效果比较

求得相关划分代价结果如图 7(a)所示,实验显示 EnvSUTP 算法代价在室内场景及室外场景远低于 QHull 算法,且对紧急逃生室内环境代价优势尤为突出.结果验证了基于环境结构特点能够排除被障碍隔离的相邻区域个体间感知计算量,进而有效降低节点间同步通信量的特点.

QHull 任务划分算法是文献[5]中对比执行效率最高的算法,图 7(b)显示两种算法执行时间均不超过 20ms,说明两种算法都具有较高的性能.同时 EnvSUTP 算法对室内楼层和室外场景的执行时间均明显低于 QHull 算法,验证了基于环境区域的粗粒度的划分,能

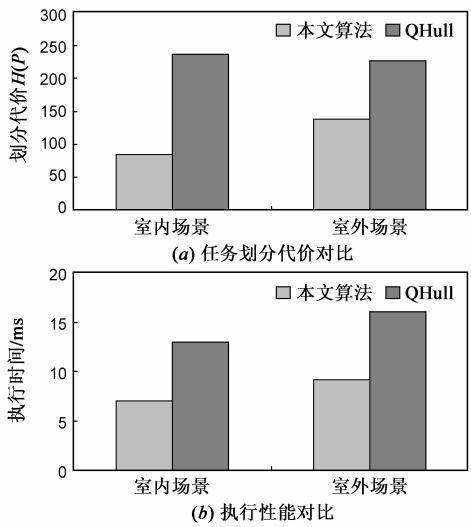


图7　与QHull算法对比分析

够达到更好的运算性能.

实验 3　与 GPA 算法对比分析

EnvSUTP算法通过增加任务划分粒度以提高算法性能的思路和 GPA 算法[2]类似,都具有较高性能.相关工作已经分析过 GPA 算法只适合平面环境,而 EnvSUTP 算法能满足多层次复杂环境的应用,有更好的环境适用性.本实验主要验证 EnvSUTP 算法和 GPA 算法在包含环境障碍平面环境中分区代价方面的优劣.以图 6(*a*)中 100m × 100m 单层楼室内环境为实验环境,以实验 1 中 EnvSUTP算法在单层楼室内环境导出顶点集作为 GPA 算法输入,均匀网格的单元格长度设为 5m,按 4 个 CN 计算节点数划分.

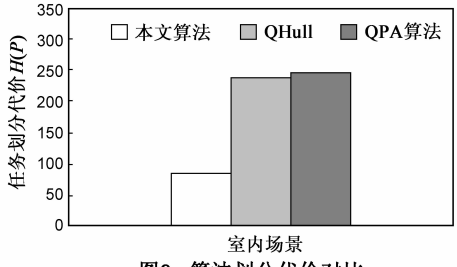


图8　算法划分代价对比

划分代价对比如图 8 所示,QPA 算法接近 QHull,都远高于 EnvSUTP 算法,验证了 GPA 算法中忽略环境障碍采用直线距离计算感知关联并不能真实反映个体间近邻关系,造成划分代价较高的不足.

7　结论

文章提出一种基于环境结构特点的任务划分算法,同时设计了最适合该算法的分布式仿真模型.划分算法引入环境因素,从而适用于多层次复杂场景.通过自动化提取仿真场景结构并转化为 CPGs 图,结果可同时用于任务划分和群体导航,不增加过多额外开销.在人群规模较大时,以环境结构区域作为基本单元的粗粒度划分比直接对 Agent 进行划分具有更高的算法性能.通过环境排除被障碍隔离的感知关联计算可最大限度降低分区间通讯代价.实验结果显示算法具有较高的性能和极低的划分代价,相同规模人群的仿真响应时间与仿真节点数呈近线性关系显示系统具有良好的可扩展性.

参考文献

[1] Vigueras G, Orduña J M, Lozano M. Advances in Practical Applications of Agents and Multiagent Systems [M]. Berlin: Springer, 2010. 15 – 24.

[2] Wang Y, Lees M, Cai W. Grid-based partitioning for large-scale distributed agent-based crowd simulation[A]. Proceedings of the 2012 Winter Simulation Conference [C]. USA: IEEE Computer Society Press, 2012. 241 – 252.

[3] Tianlu Mao, Hao Jiang, Jian Li, et al. Parallelizing continuum crowds[A]. Proceedings of the 17th ACM Symposium on Virtual Reality Software and Technology[C]. New York: ACM, 2010. 231 – 234.

[4] Yilmaz E, Isler V, Yardimci Y C. The virtual marathon: parallel computing supports crowd simulations [J]. IEEE Computer Graphics & Applications, 2009, 29(4): 26 – 33.

[5] Vigueras G, Lozano M, Orduña J M, et al. A comparative study of partitioning methods for crowd simulations[J]. Applied Soft Computing, 2010, 10(1): 225 – 235.

[6] Wang Y, Lees M, Cai W, et al. Cluster based partitioning for agent-based crowd simulations [A]. Proceedings of the 2009 Winter Simulation Conference[C]. USA: IEEE Computer Society Press, 2009. 1047 – 1058.

[7] Lorek H, White M. Parallel bird flocking simulation[A]. Proceedings of BCS International Conference on Parallel Processing for Graphics and Scientific Visualization[C]. British: Computer Society, 1993. 1 – 13.

[8] Lui J, Chan M. An efficient partitioning algorithm for distributed virtual environment systems[J]. IEEE Transactions on Parallel and Distributed Systems, 2002, 13(3): 193 – 211.

[9] Zhou, B, Zhou, S. Parallel simulation of group behaviors[A]. Proceedings of the 2004 Winter Simulation Conference [C]. USA: IEEE Computer Society Press, 2004. 364 – 370.

[10] Lozano M, Orduña J M, Cavero V. A genetic approach for distributing semantic databases of crowd simulations[A]. Proceedings of the 21st International Parallel and Distributed Symposium[C]. USA: IEEE Computer Society Press, 2007. 1 – 8.

[11] Quinn M, Metoyer R, Hunter-Zaworski K. Parallel implementation of the social forces model[A]. Proceedings of the Sec-

ond International Conference in Pedestrian and Evacuation Dynamics[C]. Berlin: Springer, 2003. 63 – 74.
[12] Reynolds C. Big fast crowds on PS3[A]. Sandbox'06: Proceedings of the 2006 ACM SIGGRAPH Symposium on Videogames[C]. New York: ACM, 2006. 113 – 121.
[13] Haumont D, Debeir O, Sillion F. Volumetric cell-and-portal generation[J]. Computer Graphics Forum, 2003, 22(3): 303 – 312.
[14] Pettre J, Laumond J-P, Thalmann D. A navigation graph for real-time crowd animation on multilayered and uneven terrain [A]. First International Workshop on Crowd Simulation[C]. New York: Pergamon Press, 2005. 81 – 90.
[15] Pelechano N, Badler N. Modeling crowd and trained leader behavior during building evacuation [J]. IEEE Computer Graphics and Applications, 2006, 26(6): 80 – 86.
[16] Lerner A, Chrysanthou Y, Cohen-Or D. Efficient cells-and-portals partitioning [J]. Computer Animation & Virtual Worlds, 2006, 17(1): 21 – 40.
[17] G Karypis, V Kumar. Multilevel *k*-way partitioning scheme for irregular graphs[J]. Journal of Parallel and Distributed Computing, 1998, 48(1): 96 – 129.
[18] Zhou Wenping, Tang Haoxuan, Ji Zhenzhou. GPU instancing method for crowd animation based on motion capture data [J]. Journal of Computational Information Systems, 2013, 9 (11): 4459 – 4467.
[19] Vigueras G, Lozano M, Perez C, et al. A scalable architecture for crowd simulation: Implementing a parallel action server [A]. Proceedings of the 37th International Conference on Parallel Processing [C]. USA: IEEE Computer Society Press, 2008. 430 – 437.

作者简介

周文平 男, 1984 年生于江西吉安, 哈尔滨工业大学计算机科学与技术学院博士研究生. 主要研究方向为并行计算和虚拟现实.
E-mail: zhoulangtian@163.com

唐好选 男, 1971 年生于甘肃武威. 哈尔滨工业大学计算机科学与技术学院副教授. 主要研究方向为计算机图形学、虚拟现实、物联网和 3G 移动信息化技术.
E-mail: tanghx@hit.edu.cn

季振洲 男, 1965 年生于黑龙江哈尔滨, 哈尔滨工业大学计算机科学与技术学院教授、博士生导师. 主要研究方向为计算机体系结构和并行计算.
E-mail: jizhenzhou@hit.edu.cn