基于 Repast Simphony 平台的建模与仿真技术①

王宇宾

(河海大学 商学院, 南京 211100)

摘 要: 随着计算机技术的发展, 基于 Agent 的建模与仿真技术被认为是研究复杂系统的有效方法, Repast Simphony 平台为基于 Agent 的建模与仿真提供了有利条件. 针对如何利用 Repast Simphony 平台构建复杂系统仿真模型的问题, 重点介绍了 Repast Simphony 仿真平台, 分析了它的技术优势, 与同类仿真平台进行了比较分析, 并在分析平台主要类库的基础上, 总结出建模仿真的一般流程. 最后, 通过一个改进的 Schelling 模型仿真实例进一步阐述了利用平台构建复杂系统的设计与实现方法, 对使用 Repast Simphony 平台进行基于 Agent 的建模与仿真研究具有一定的指导意义.

关键词: Repast Simphony; 建模与仿真; 复杂系统; Schelling 模型

Modeling and Simulation Technology Based on Repast Simphony

WANG Yu-Bin

(Business School, Hohai University, Nanjing 211100, China)

Abstract: Agent-based modeling and simulation technology is considered to be an effective way to study complex systems, Repast Simphony platform provides favorable conditions for agent-based modeling and simulation. According to the problem of how to build simulation models of complex systems on Repast Simphony, this paper focuses on Repast Simphony simulation platform. Its technical superiorities is introduced. The similar simulation platforms are analyzed comparatively. The general simulation and modeling process is summarized through analyzing the main library for the platform. At last, an improved Schelling model simulation example is given for further describing the design and implementation for building complex systems based on Repast Simphony. It is of significance to use Repast Simphony for agent-based modeling and simulation research.

Key words: Repast Simphony; modeling and simulation; complex system; Schelling model

近年来, 基于 Agent 的建模与仿真方法和技术(Agent-based modelling and simulation, ABMS)在各学科领域中被广泛应用, 它在现实世界的复杂系统研究中具有不可替代的地位[1]. 同时, 各种建模仿真工具应运而生, 它们利用计算机技术为 ABMS 提供了良好的支持, 极大地促进了 ABMS 应用的发展, 为复杂系统理论的应用提供了可靠的方法和手段. 目前主流的建模与仿真工具包括 Repast, NetLogo 和 Swarm,它们各自有相应的仿真平台, 为构建基于 Agent 的仿真模型提供了功能强大的集成开发环境.

在众多仿真平台中, Repast 正被许多研究者使用并迅速发展更新. Repast Simphony 平台(以下简称 Repast S)是在 Repast 基础上发展而来的面向对象的、可视化多 Agent 建模与仿真平台. 自发布以来, 它已被广泛地应用于各个学科领域的复杂系统研究当中. 由于其良好的易用性和强大的扩展性, Repast S 被众多学者认为最具发展前景[2].

目前, Repast S 仿真平台正处于快速发展阶段, 其应用范围[3]涵盖了社会学、经济学、军事等众多领域, 但与平台使用相关的技术文献还比较少. 因此, 研究

① 收稿时间:2015-01-21;收到修改稿时间:2015-04-02

如何利用 Repast S 平台进行建模与仿真并构建出有研究价值的模型显得十分重要.

1 Repast Simphony平台分析

1.1 简介

Repast 起源于美国芝加哥大学社会科学计算研究所和 Argonne 国家实验室, 目前由非盈利组织 ROAD (Repast Organization or Architecture and Development)负责后续版本的维护与升级. Repast 提供了[4]一系列用来生成、运行、显示和收集数据的类库, 而且对运行中的模型进行“快照”, 记录模型在某一时刻中的实时状态, 并且可以把仿真模型运行过程中 Agent 状态动态演化过程生成视频资料.

Repast S 仿真平台实际上是将 Repast 仿真工具包嵌入 Eclipse 开发平台上的集成开发环境. 它是一款[5]高交互式并且容易学习和使用的, 基于 Java 编程建模的软件系统, 尤其适合在工作站和小计算集群中使用. 目前最新版本为 2.2, 发布于 2014 年 6 月, 本文基于最新版本进行分析和探究.

1.2 优势

Repast S 平台的优势有以下六点:

① 跨平台. Repast S 为不同的操作系统提供了不同下载版本, 支持在 Windows、Mac、Linux 平台上进行设计开发. 此外, 其跨平台性还表现在模型的使用上, 在一个平台发布模型后, 该模型可导入到其他平台上使用. 这一特点使模型得以分享, 极大地促进了 ABMS 的发展.

② 支持多种方式建模. 除了支持编写代码(如 Java, Groovy, ReLogo 等)的传统方式实现模型之外, Repast S 还支持使用可视化流程图(FlowChart)来设计仿真流程, 以”point–and–click”方式构建出模型.

③ 强大的算法支持. Repast S 不仅提供了传统算法的支持, 而且还提供了多种人工智能算法的支持, 如常用的遗传算法和神经网络算法. 这些算法是通过其内置的开源算法类库包来实现的, 分别使用遗传算法包 JGAP 和神经网络算法包 JOONE, 在构建模型时只需调用相应的算法类库即可实现算法.

④ 对 GIS 的支持. Repast S 为基于 Agent 的建模与仿真提供了相应的 GIS 支持库, 开发者可以再平台上结合 GIS 工具集成开发.

⑤ 批处理(Batch Run)和数据收集(Data Collection). 在仿真中通常要进行多次不同参数的输入来获取相应的大量数据, 这个过程如果以手工完成无疑会十分费时费力. Repast S 提供的批处理功能可以让研究者通过一次性配置实验所需的不同输入参数, 让平台自动进行仿真模拟, 最后利用数据收集功能对每一次的仿真结果进行记录.

⑥ 支持构建 3D 的表现形式. Repast S 提供与 3D 表现相关的工具包, 通过构建 3D 模型, 研究人员可以更精确地观察系统中 Agent 的活动情况.

1.3 比较分析

除了 Repast S 以外, 目前主流的仿真平台还包括 Swarm 和 NetLogo,下面将对以上几款仿真平台作比较分析, 如表 1 所示. 在基于 Agent 的建模与仿真实践操作过程中, 仿真平台的选择通常需要从易用性和功能性两个方面进行权衡[6]. 首先从平台易用性分析, NetLogo[7]提供了简单且有力的 Logo 语言、内置图形化接口和丰富的支持文档, 可用于支持在网格环境中具有局部交互的 Agent 系统建模. Swarm[8]是“框架类库”平台, 为基于 Agent 建模提供了组织设计模型的概念框架和相应的软件类库, 建模者需要了解平台相关的类库文档, 然后在框架之下编写代码来构建多 Agent 系统, 建模者无疑需要更多的编程技术支持才能构建出想要实现的模型. 当前 Repast S 平台已经支持通过绘制流程图的方式构建 Agent 模型, 建模者可以先设计出模型的整体结构, 然后再编写代码实现模型细节上的需求. 因此, 在易用性上NetLogo和Repast S 平台比 Swarm 更胜一筹. 另一方面, 从平台的功能性分析, 在 Repast 和 Swarm 上建模者可以在类库的基础上进行扩展, 以满足实际需求. 其中 Repast 是纯 Java 实现的, 可以方便地把第三方的 Java 类库直接引入平台中使用, 同时 Repast S 中也内置了遗传算法类库 JGAP, 神经网络算法等类库 Joone, 因此, Repast S 的扩展性要比 Swarm 更加强大. 虽然在 NetLogo 上也能使用 Java 语言对其进行外部控制或者扩展功能, 但就平台本身而言, NetLogo 并未提供相关的可计算模型类库. 因此, NetLogo 在功能方面显得比较有限. 所以, 从功能上分析, Repast S 平台最具优势.

表 1 仿真平台比较

	NetLogo	Swarm	Repast S
易用性	一般 (Logo 语言、内置图形化接口)	较差 (仅支持编写代码实现)	较好 (流程图方式建模, 支持多种编码语言)
内置计算模型	无	遗传算法 Breeder; 分类器系统 CW; 神经网络算法 NeuroLib; 目标交叉算法包 BP–CT	遗传算法类库 JGAP; 神经网络包 JOONE; 回归算法;
实验支持	一般(多场景实验自动化管理)	较差(没有自动化工具, 需要编写程序)	较好(菜单驱动, 支持批处理)

在建模仿真中, 通常需要根据研究的实际需求来选择仿真平台. 上述三个仿真平台在不同领域中均有可供借鉴的模型. 但综合来看, Repast S 是目前最具活力的多 Agent 建模与仿真平台, 它吸取了 Swarm 的核心理念, 同时借鉴了其他仿真平台的友好建模方式. 此外, Repast 是一个开源的仿真平台并持续地改进已有功能. 同时, Repast 项目组还维护着一个活跃的用户讨论邮件组, 任何人都可以加入邮件组来获得技术支持. 正是在其良好的发展前景之下, Repast S 平台被越来越多的研究者采用来进行多 Agent 的建模与仿真研究.

1.4 类库分析

通过调用仿真平台提供的类库, 我们能够方便地使用仿真平台提供的功能. Repast S 的强大功能是建立在 Repast

框架工具包的类库基础上的, 分析 Repast 的类库有助于建模者进一步了解建模与仿真过程的实现和运行机制. 在一般的建模与仿真中主要使用到以下类.

① 引擎类(Engine). 该类的使用贯穿于整个仿真过程, 是最重要仿真调度类. Repast 的仿真调度策略结合了时间步长与事件步长. 每一个时间步称为一个”tick”.事件的发生一般依赖于 Agent 的调用, 在 Agent 的方法步中可以加入时间表(Schedule)标识来注明该事件何时被调度执行. 其中包括事件的发生时间(starttime),发生间隔(interval)和优先级(priority).

② 空间类(Space). 在多 Agent 的建模与仿真中, 空间是 Agent 生存和活动的区域, 在类库 Space 中有四种空间可供模型使用, 包括连续空间(Continuous Space), 地理信息系统(GIS), 网格(Grid)和网络(Network).建模者可以根据实际需要使用不同的空间或者将多种空间结合使用.

③ Agent 类. 在 Repast 中没有提供明确的 Agent 类, 因为 Agent 类和面向对象编程语言中的 Object 类十分相似, 建模者可以根据实际需要创建一个或多个 Agent 类, 并定义其属性和行为.

④ 随机数生成类(Random). 在仿真中随机数的生成对仿真结果的影响十分重要. 在类 RandomHelper 的帮助下, 用户可以迅速生成模型需要的随机数, 如得到服从均匀分布, 服从 ζ 分布和服从 β 分布的随机数.

⑤ 输入和输出类. 在 Repast S 中有两种仿真模型的运行方式, 分别为批处理方式和非批处理方式. 其中, 批处理方式指是通过读取参数文件自动执行多次仿真实验并输出相应的结果. 当中主要使用的类包括 Batch 和 Parameter. 非批处理方式是指通过提供的图形用户界面(GUI)设置模型参数以及在运行时期动态修改参数运行仿真程序并产生实验数据.

1.5 建模与仿真流程

Repast S 平台有两种状态, 一种是设计态, 另一种是运行态, 分别对应于建模与仿真两个阶段. 在不同的状态下, 两者的界面也不相同.

在建模阶段, 我们可以以编写代码和绘制流程图的方式来构建模型. Repast S 的建模实现以基于 ABMS 方法为基础, 结合了 Repast S 平台的特性, 建模步骤有以下 5 步:

① 根据所研究的复杂系统抽象出其中的 Agent, 并分析 Agent 的属性, 所处环境结构以及 Agent 之间的关系和交互行为.

② 设计 Agent 环境. 选择合适的空间类来描述 Agent 之间的关系, 创建 Context 环境并为其添加空间的信息.

③ 实现系统模型包含的 Agent, 并定义其属性, 关系及行为.

④ 选择合适的输入、输出方式来设置仿真程序的参数和收集数据.

⑤ 运行仿真程序, 根据仿真结果修正系统模型.

在仿真阶段, Repast S 的仿真运行流程如图 1 所示.

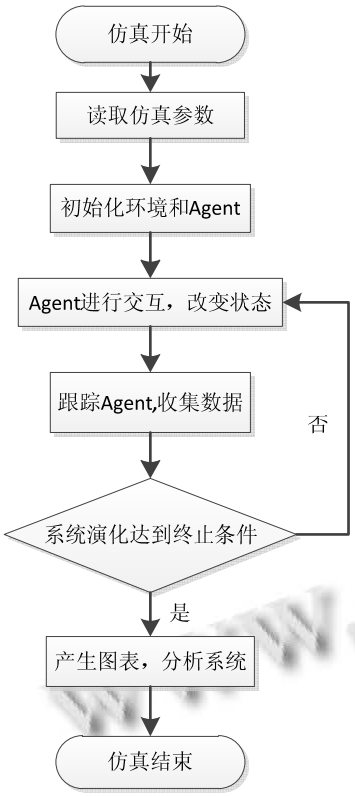


图 1　Repast S 平台仿真流程

2　基于Repast S的仿真模型实现

为了更好地说明使用 Repast S 平台建模与仿真的基本方法和步骤, 本文利用 Repast S 平台中提供的 Schelling 基本模型进行阐述. 该模型的实现参考了文献[9]对 Schelling 模型的改进. 在模型中加入 Agent 的死亡与繁衍特性, 同时使模型中的 Agent 类型增加到三种, 使之更贴近现实世界.

2.1 问题背景

Schelling 模型[10]是由经济学家 Thomas C. Schelling 针对微观个体行为如何演变为社会的宏观居住隔离问题而提出的. 在 Schelling 模型中, 有两种不同类型的人群, 每人根据其邻居构成来选择迁移或者留下继续居住, 并且每个人总是会不断寻找让他满意的邻居构成位置居住. Schelling 模型假设人们更愿意与自己同类型的人相处在一起, 给定每个人一定的同类邻居期望比例, 当人们发现附近的同类邻居少于期望比例时, 就会选择搬迁到一个新的地方居住; 附近的同类邻居大于或等于期望比例时, 人们则选择留下来继续居住. Schelling 模型证明了即使在两类人群相对比较能容忍与对方相处的前提下, 仍然会出现大范围居住隔离现象.

2.2 Agent 的环境和属性

2.2.1 Agent 的环境

实例中 Agent 的环境空间由网格(Grid)类实现. 网格是一种适合研究多个 Agent 在一定区域内活动的相对离散的空间结构. 网格中的格子代表 Agent 的居住地, 每个格子只能容纳一个 Agent,格子在网格中的位置可以通过坐标位置来获取.

2.2.2 Agent 的属性

根据上述问题背景分析可知, Agent 的主要属性包括: 当前年龄(currentAge), 死亡年龄(maxAge), 类型(type)以及同类邻居期望比例(percentLikeNeighbors). 其中 Agent 的当前年龄随着时间步的增加而增加; 每个 Agent 有不同的死亡年龄, 在仿真参数设置的死亡年龄区间内随机确定; Agent 的类型在仿真初始化时随机确定, 不同类型的 Agent 在网格中呈现不同的颜色; Agent 的同类邻居期望比例值相同, 由仿真参数所决定.

2.3 Agent 的关系和行为

2.3.1 Agent 的关系

模型使用 Moore 邻域结构来描述 Agent 之间的关系. 如图 1 所示, 当 Agent 位于图 1 中 A 的位置时, 其邻居分别为位于它四周 B 位置上的 Agent.

2.3.2 Agent 的行为

Agent 的行为主要有搬迁和死亡. 具体描述如下: 在每一个”tick”周期中, Agent 先判断身边的同类邻居比例与期望比例的关系, 当前者小于后者时, Agent 不满意现在的居住地并搬迁到一个新的网格位置中; 当前者大于等于后者时, Agent 保持所在位置. 然后判断 Agent 是否达到死亡年龄, 如果达到则该 Agent 从网格中消失, 同时在相同的网格位置中生成一个新的随机类型的 Agent 以替换死去的 Agent.在 Repast S 平台中编写 Java 代码实现上述设定. 鉴于篇幅所限, 这里仅给出关键代码框架.

代码如下所示:

```
public class Agent {
    private int maxAge, currentAge, type;
    private double percentLikeNeighbors;
    @ScheduledMethod(start=0,interval=1)
    //从第 0 个“tick”开始, 以 1 个“tick”为间隔, 循环调用
    public void step() {
```

```
                ......
        boolean lookingForNewSite = true;
        while(lookingForNewSite){
            // 查询网格中的 Moore 邻居
            MooreQuery<Agent> query = new MooreQuery<Agent>(grid,this);
            ......
            }
            // 满足期望比例, 不搬迁
            if (numLikeMe / neighborCount >= percentLikeNeighbors){
                lookingForNewSite = false;
            }
            else{
                //Agent 搬到新的位置
                grid.moveTo(this, x,y);
            }
        }
        if (currentAge >= maxAge)
            this.die(); //执行死亡行为
    }
}
```

2.4 设置仿真参数

仿真实验通过设置三种不同的同类邻居期望比例值来探讨微观主体的行为偏好对宏观现象的涌现作用. 设置 2000 个 Agent 在 50*50 的网格区域中交互演化, 具体仿真参数设置如表 1 所示.

表 2 仿真参数设置

仿真参数	设置值
初始 Agent 数	2000
死亡年龄	[80,100]
网格大小	50×50
同类邻居期望比例(%)	33, 50, 80

2.5 仿真运行与结果分析

按照上述步骤, 把 Schelling 模型搭建起来后, 我们从 Repast S 平台中启动模型, 进入仿真运行状态. 从图 2 中我们可以看到 Repast S 平台提供了可交互的图形用户界面, 用户能方便地控制仿真系统的运行, 同时仿真模型中的各项数据变化也能通过图表直观地反馈给用户, 如图 2 中的折线图反映了当前仿真系统中各类 Agent 数量的实时变化情况. 特别地, 我们通过实现平台的 Style3D 接口, 让系统的演化在 3D 效果下呈现, 用户能够更精确地观测系统中每个 Agent 的变化情况.

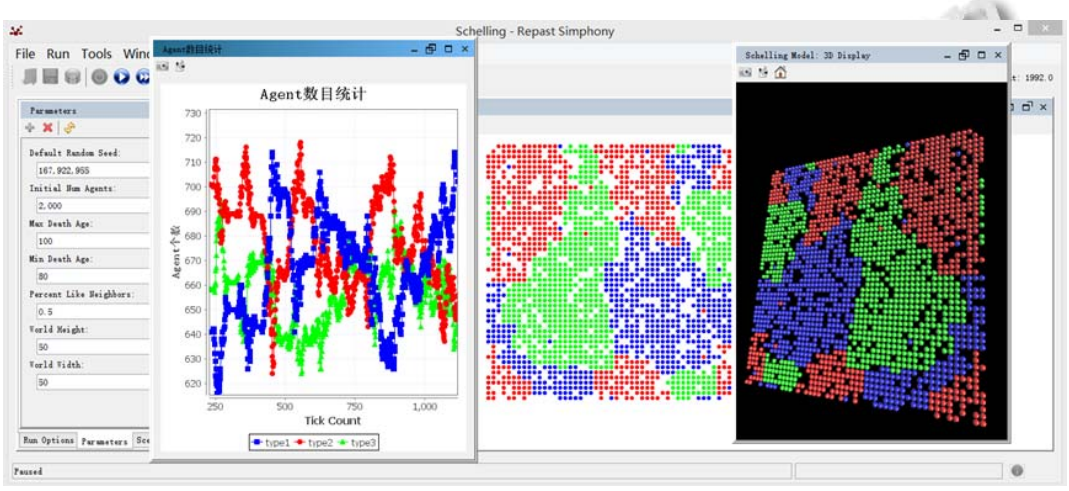
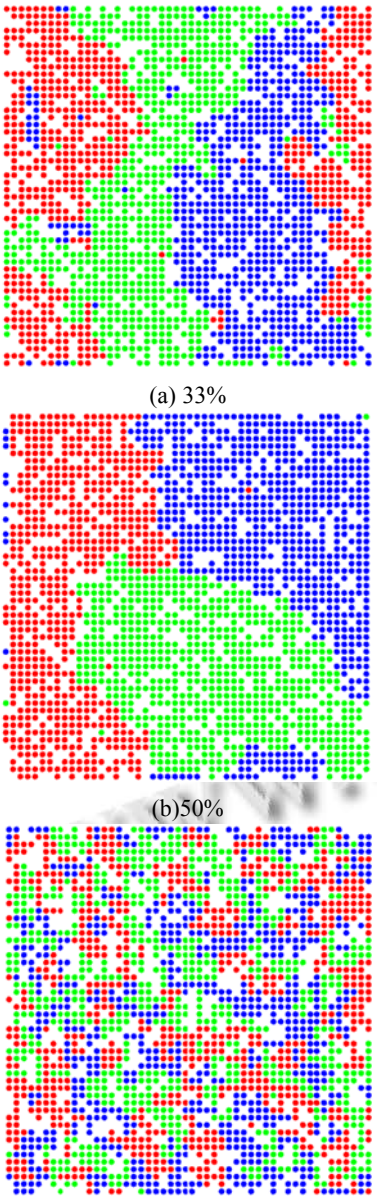


图 2 仿真运行界面

仿真结果如图 3 所示, 通过设置不同的同类邻居期望比例, 经过 10000 个"tick"周期后仿真系统将分别"涌现"出不同的现象. 从仿真结果中, 我们可以发现, 在较低的同类邻居期望比例(33%和 50%)下均出现了明显的"居住隔离"现象, 如图 3(a)和图 3(b)所示, 系统呈现出一种相对稳定的状态; 而在较高的同类邻居期望比例(80%)下没有出现明显的隔离现象, 如图 3(c)所示. 由于同类邻居期望比例很高, 系统中的 Agent 不断搬迁以寻找满意的居住地, 此时系统呈现出一种不稳定的状态, 不同类型的 Agent 在网格中持续大规模

地变动. 仿真结果可以解释为即便人们相对比较能容忍与不同类型的人们相处在一起, 如在 33%和 50%的同类邻居期望比例下, 还是会出现大范围的居住隔离现象. 相反, 如果人们有极端的同类型邻居偏好, 如在 80%的同类邻居期望比例下, 居住隔离现象反而不容易形成. 由此, 我们可以深刻地认识到, 宏观层次上发生的居住隔离现象, 可能并不是由于在微观层次上人们对周围环境的苛刻要求造成的. 正如 Thomas C. Schelling 所阐述的: 微观上的动机并不一定等同于宏观上的表现. 仿真模型揭示了宏观隔离现象与微观行为间的关系, 达到了实验目的.



(a) 33%

(b)50%

(c) 80%

图 3 不同同类邻居期望比例下的仿真结果

3 结语

Repast S 平台为用户提供了一种便捷的对复杂系统进行建模仿真分析的计算机实现方法. 通过定义微观主体的行为特征, 观察仿真过程中"涌现"出的宏观现象, 结合平台提供的图表和图形输出结果, 我们可以方便地评价各种模型, 策略以及方法的实施效果. 本文首先对 Repast S 平台进行简单介绍, 分析了平台的优势特点; 通过与其他仿真平台比较分析, 突出了 Repast S 全面的 ABMS 支持和良好的发展前景; 然后分析了平台提供的主要类库, 总结了利用 Repast S 平台建模仿真的基本步骤流程. 最后详细介绍了一个改进的 Schelling 模型仿真实例的实现过程. 仿真的运行揭示了现实生活中居住隔离的形成机制, 同时表现了 Repast S 平台优秀的建模与仿真支持.

文中仅给出了一个基本的 Schelling 模型实现案例, 要在具体问题研究中运用该模型还可以对复杂系统模型作进一步的扩展. 比如我们可以利用 Schelling 模型研究收入隔离现象, 此时, 只需重新定义 Agent 的相关属性来表现其收入和偏好特性即可, 不需要改动模型的整体框架. 事实上, Repast S 还为我们提供了许多常用的基本模型, 比如"糖域模型", "捕食者模型""元胞自动机"等. 在实际应用中我们可以根据所研究的复杂系统, 应用本文的思路对 Agent 的环境, 属性, 关系以及行为进行分析, 然后选用相似的基本模型进行扩展, 可以方便地实现复杂系统的建模与仿真, 这也是 Repast S 的可扩展性的具体体现.

参考文献

1 廖守亿,戴金海.复杂适应系统及基于 Agent 的建模与仿真方法.系统仿真学报,2004,(1):113–117
2 姜昌华,韩伟,胡幼华.REPAST——一个多 Agent 仿真平台.系统仿真学报,2006,(8):2319–2322
3 Macal CM, North MJ. Agent–based modeling and simulation. Winter Simulation Conference. 2009. 86–98.
4 郝成民,刘湘伟,郭世杰,等.Repast:基于 Agent 建模仿真的可扩展平台.计算机仿真,2007,(11):285–288
5 Repast 官网.http://repast.sourceforge.net/.
6 陈悦峰,董原生,邓立群.基于 Agent 仿真平台的比较研究.系统仿真学报,2011,(S1):110–116.
7 Netlogo 官网.http://ccl.northwestern.edu/netlogo/.
8 Swarm 官网 http://savannah.nongnu.org/projects/swarm.
9 Epstein JM. Generative social science: studies in agent–based computational modeling. Princeton University Press, 2006.
10 Schelling TC. Dynamic models of segregation. Journal of Mathematical Sociology, 1971, 1(2): 143–18.