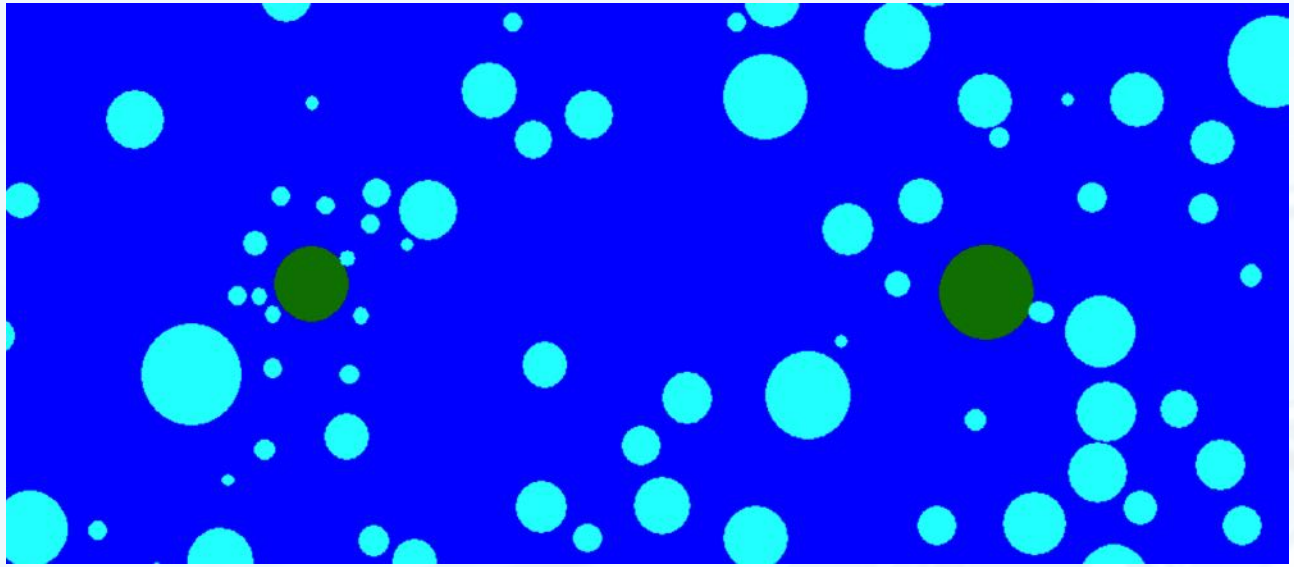
数据结构与算法（Python）-期末大作业

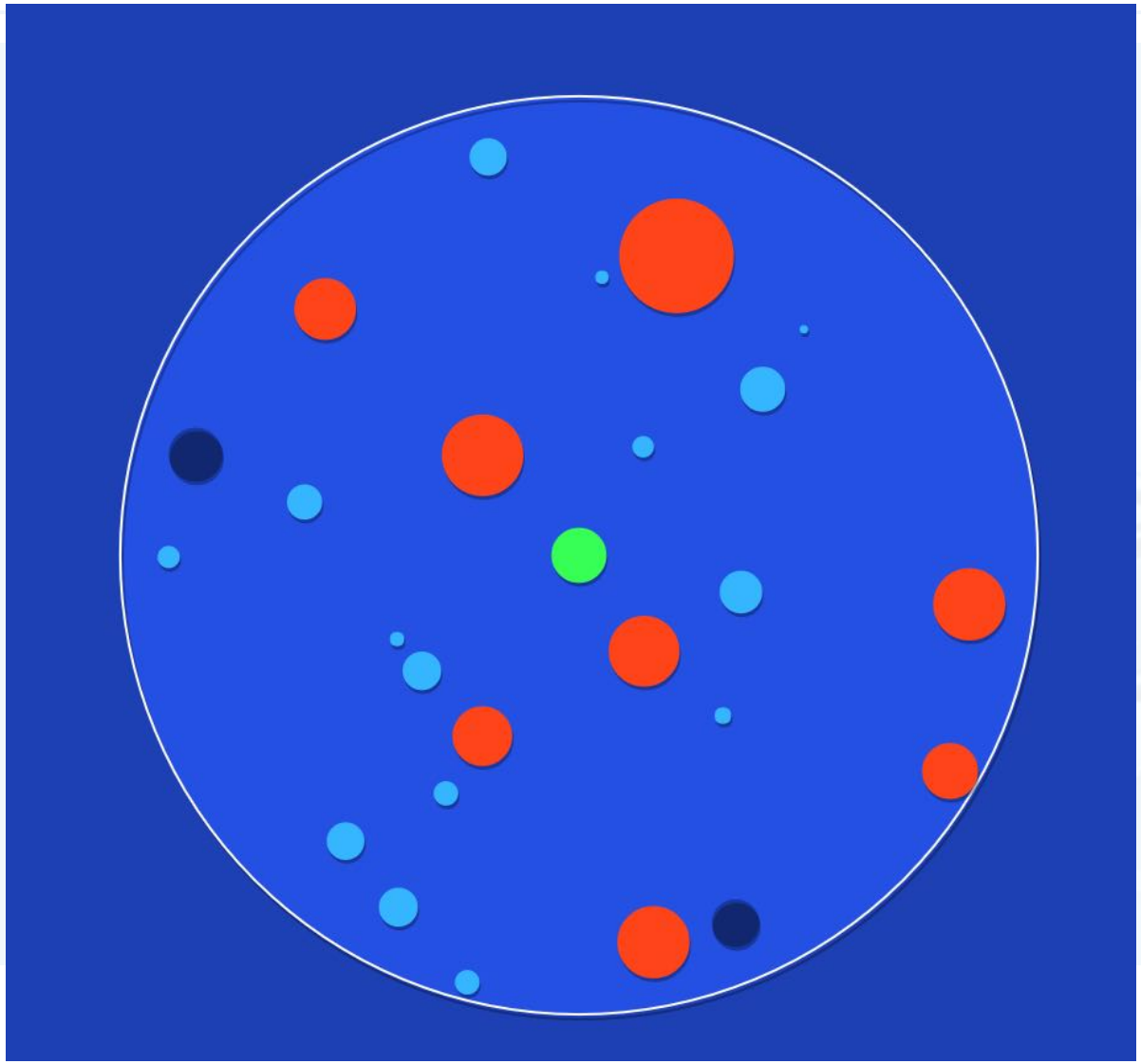
陈斌 gischen@pku.edu.cn 北京大学地球与空间科学学院

期末大作业：星际吞噬/Osmo

- 任务描述
- 组队
- 作业评分标准
- SESSDSA Osmo算法竞赛规则
- 实习作业时间进度
- 小组算法开发指南

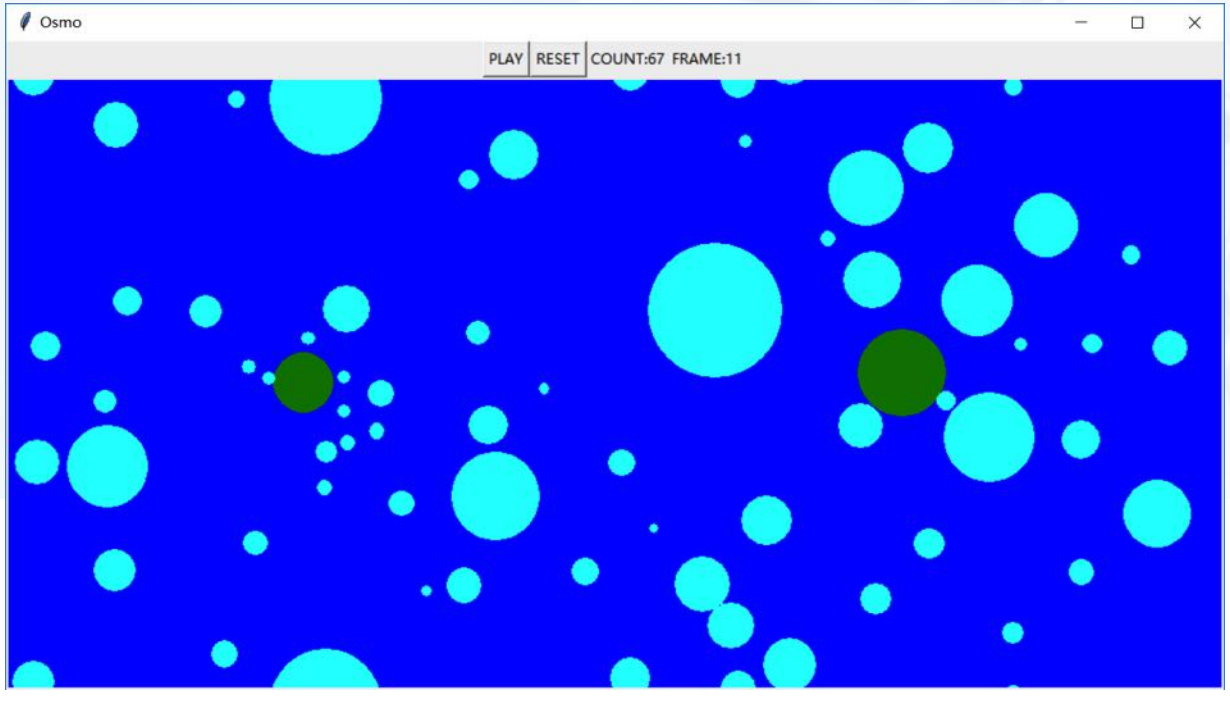


游戏原型：Osmos（ Hemisphere Games ）



星际吞噬Osmo

- 一个回合制AI对抗游戏
- 两队AI分别控制0/1号两个星体，号码在开始游戏时随机分配
- 每个星体有一定的半径，半径决定其面积，星体的质量正比于面积。
- 星体互不重叠且能够以一定速度运动。
- 完全信息决策



Osmo：场地World

- **场地**

  场地大小为1000*500

- **初始位置和运动**

  双方初始分别在左右两侧，场地的四等分点位置，初始速度为0。

- **其他星体**

  场地中除了玩家星体外还将随机分布CELLS__COUNT个星体,分别按照半径分为大、中、小三类

Osmo：场地World（续）

编号

随机分布的星体获得递增的编号。星体被完全吸收后，标记其死亡并不再更新;每次在弹射产生新的星体后,使新的星体获得一个新的编号

漂浮与边界

在没有外界影响的情况下，所有星体将做匀速直线运动；而当星体部分穿出游戏场地的边界时，穿出的部分将速度不变地移动至场地相对的一侧

Osmo：接触与吸收

当两个星体相接触（即二者中心距离小于二者半径之和）时，将会发生吸收过程，吸收过程保持质量守恒和动量守恒。被完全吸收的星体将会从对局中移除。

吸收和被吸收者的判定

当二者半径不等时，半径较小者被吸收

当二者半径在误差范围内相等，速度较大者被吸收

当二者半径和速度在误差内均相等，星体编号较小者被吸收

Osmo：接触与吸收（续）

当存在多个星体同时接触时

（1）由星体编号最小的星体开始判断，若此时与其接触的星体中有半径大于此星体的，则此星体被吸收。之后星体的判断中视为此星体已不存在。

（2）若一个星体同时被多个星体吸收，则按以下规则判定被何者吸收：

当各吸收星体中半径最大者唯一，则被半径最大者吸收。

当各吸收星体中半径最大者不唯一，则被速度最小者吸收。

当各吸收星体中半径最大且速度最小者不唯一，则被星体编号较小者吸收。

Osmo：弹射

星体可主动将自身的部分质量作为一个新的星体弹射出去

弹射角度可由玩家自由控制。弹射过程视为碰撞的逆过程，依然遵守质量和动量守恒

玩家可以利用在弹射星体过程中的反作用来控制自身运动，不断吞噬比自身小的星体，并躲避比自己更大的星体

玩家每一帧中最多可弹射1次星体，弹射星体的质量与本体质量之比为一恒定值EJECT_MASS_ RATIO ，相对自身速度为恒定值

Osmo：回合制

- 游戏时间按照tick均匀流逝
- 一场比赛tick总数有限制
- 星体喷射方向用theta或None表示
- 控制

每一帧内两个玩家根据相同的信息做出决策，然后依次运动

玩家给出对星体的指令应当为：theta（表示沿theta角度喷射）或None（表示不喷射）

给出喷射命令后在1个tick中立即执行，并运动

Osmo：输入输出要求

- **编写一个类Player**

  输入为一个list,为当前的所有星体(包括玩家星体) ,下标为前述的星体编号。

- **每个星体数据为一个Cell**

  包含形如pos,veloc等参数。其中pos=[x，y]，veloc=[v_x，v_y]。

  x，y，v_x，v_y，radius分别代表星体的x坐标，y坐标，x速度分量，y速度分量,半径,均为浮点数。

- **要求返回一个值即弹射角度theta**

  采用弧度制,可为[0，2Π)内的任意浮点数。若不弹射,返回None

Osmo：运算时限

- **每个玩家总的运算时限为10秒**

  即对一个玩家，在每个tick中做出决策需要一定的运算时间

  在该场比赛中，这些运算时间之和不能超过10秒

- **对于一般的算法，基本不会出现超时的情况（如所给的示例算法）**

Osmo：胜负判定

- **下列情况立即胜利**
  - 对方星体被吞噬且己方存活
  - 对方运算时长耗尽且己方尚未耗尽
  - 对方代码出错
- **下列情况立即失败**
  - 己方星体被吞噬且对方存活
  - 己方运算时长耗尽且对方尚未耗尽
  - 己方代码出错
- **平局**
  - 双方星体被同时吞噬
  - 双方运算时长同时耗尽
- **下列情况立即结束，以双方各自的星体大小判断胜负**
  - Tick总数消耗完
  - （若大小相等则判为平局）

任务描述

- **编程：依托Osmo基础设施代码，用Python编写对战算法**

  根据当前场上态势，返回本方的喷射角度theta指令

  要求应用本课所学到的数据结构与算法，如栈、队列、链表、散列表、递归、动态规划、树、图等部分组合，并具有一定的复杂度和智能。

  要求代码结构清晰、格式规范、注释丰富。

- **报告：撰写算法实现过程的实验报告**

  包括算法思想阐述、程序代码说明、测试过程报告、小组分工和实验过程总结等4个部分

  要求实验报告图文并茂、内容丰富、结构清晰、写作规范、逻辑性强。

- **竞赛：参加SESSDSA Osmo算法竞赛**

  与其他小组的算法对战，根据输赢获得竞赛排名

  要求对战过程基本无bug、无异常

组队

- 分组进行实习作业，原则上每组4-5人，设组长1名
- 组队过程由组长确定开始，**确定后**组长开始招募组员

  组长确定原则：以自愿报名为主，自愿报名表单见课程网站。
- 组员招募遵循自愿原则，提倡**均衡**原则
- 组长负责：

  召集实习作业过程讨论会

  汇总代码和报告

  代表小组参加竞赛

作业评分标准

- **数算实习作业占总评的25%，即25分**
- **算法编程占9分**
- **实习报告占8分**
- **联盟内的竞赛排名占8分**

  参赛无bug无异常得3分；第1轮出线得5分；季军得6分；亚军得7分；冠军得8分

- **评分适用于全组同学**

  每组有额外3分加分，可由组长组织本组民主评议，奖励1～2名表现突出的组员（含组长）。

  另外，组长有权对实习过程中表现差的同学提出批评及降分建议。

SESSDSA Osmo算法竞赛规则

- 竞赛目标：采用**算法**指挥己方星体，改变其在场上**运动方向**，利用场上其他星体运动态势信息**计算**己方星体喷射方向，使自己长大，或对手被吞噬。
- 双方初始分别在左右两侧，场地的四等分点位置，初始速度为0。
- 玩家改变喷射方向，每帧可喷射一次。
- 系统预设置场地大小1000*500
- 系统预设置tick总数
- 根据前述的胜负规则来判断胜负

F18联盟 vs N18联盟

- 选课一共254人：18级134人；非18级120人
- 为了均衡实力，缩小小组规模和数量
- 将全体同学分为18级和非18级两个联盟
- 联盟内部按照世界杯赛制进行4轮比赛，决出冠亚军，获得相应奖励
- 最后可以进行联盟之间的冠军友谊赛、挑战赛，以及人机对战。

SESSDSA Osmo算法竞赛规则

- 赛前进行热身挑战赛，为了避免代码泄露，参加热身赛的小组可将代码发到对战平台，以获得对其他小组的对战结果和复盘数据
- 首先将同一个联盟的小组抽签分为东西南北4个区(N-E-W-S)
- 第一轮为区内竞赛，循环赛制，每区2组出线，决出八强

  第一轮的每场胜者积3分，负者0分，平局各积1分；每区2组出线
- 第二轮为淘汰赛决出四强：E1-W2, E2-W1, S1-N2, S2-N1

  第二轮开始，每场必决出胜负
- 第三轮为四强半决赛：E1/W2-S1/N2, E2/W1-S2/N1
- 第四轮为决赛：决出联盟冠亚季军，获得神秘奖品

实习作业时间进度

- 组长报名，然后正式组队
- 即日开始实习作业，开发算法，编程测试，热身挑战，撰写报告
  注意组员分工明确，协同合作
- 6月4日（周二）课上进行算法竞赛
- 6月11日（周二）前提交完整作业
  包括代码、实验报告
- （6月18日周二下午）闭卷考试

小组算法开发指南

- **详见github代码仓库**

  https://github.com/chbpku/osmo.sessdsa

- **每组编写1个类（Player）**

  Player根据场上态势返回对己方星体的喷射角度theta（None或0~2Π之间的值）

  __init__方法里的id用于区分两个玩家（即0或1）

  arg是预留参数，如果需要增加初始参数，可以对其进行改动以适应需要，也可以不改

  strategy方法用于决策，传入的allcells为list，表示场上尚存活的星体及相关参数

  Strategy返回的是theta或None

- **由于本作业提出时间短，一定存在不足之处，请向技术组反映咨询**

  说明文档仍在持续更新