清华大学第三届人工智能挑战赛

清华大学自动化系第十八届新生 C 语言大赛

参赛手册

指导单位

共青团清华大学委员会

清华大学学生科学技术协会

共青团清华大学自动化系委员会

承办单位

清华大学自动化系学生科协

大赛官网

目录

Part I

大赛概况

1 大赛宗旨

清华大学自动化系新生 C 语言大赛曾是学校二星级赛事，以“增加同学科技热情，培养团队协作精神，提高选手设计创新能力，丰富校园文化生活”为宗旨，至今已成功举办十七届。

比赛采用 AI 对战的方式，以 C++ 为开发语言，希望锻炼同学的编程能力，体验编程之乐。另外，比赛凝聚了自动化系最优秀的同学组成开发团队，全力开发出了富有趣味性、挑战性的比赛平台。

特别地，本届新生 C 语言大赛首次与自动化系必修课程《C++ 程序设计与训练》结合，边学边练边实战，让同学们更好地掌握程序设计的基础知识以及运用技巧。

2 往届赛题回顾

2007 贪吃蛇的围剿（贪吃蛇）
2008 八皇后的争夺（棋类）
2009 怒海争锋（弹幕）
2010 弹弹堂（泡泡堂）
2011 机器人大战（弹幕）
2012 仙境灵珠（格斗）
2013 王子复仇记（即时策略）
2014 天使羊（格斗）
2015 深蓝海战（即时策略）
2016 吞噬星空（agar）
2017 百团大战（棋类）
2018 塔阵兵锋（即时策略）
2019 狼烟四起（即时策略）
2020 碧水保卫战

3 本届赛题

3.1 概述

FC18 为四方势力在地图上进行回合制对战的策略游戏，玩家需要力求攻占其他玩家的塔、占领尽可能大的领地、消灭或俘虏其他玩家的兵团以获得胜利。每个玩家需要编写 AI，根据裁判程序提供的场地信息，决策己方势力在该回合的行动，并返回给裁判程序，以控制己方的行为。

3.2 背景

3.3 亮点

4 参赛流程

4.1 大赛日程

本比赛与清华大学第三届人工挑战赛（THUAI 3）赛程一致。比赛在春季学期第 4 周（2020 年 3 月 9 日）开放报名，在第 7 周（2020 年 4 月 5 日）会有预选排位赛（不影响晋级）进行，第 9 周周末（2020 年 4 月 18 日）在科展进行初赛（正选排位赛）结果展示。在一个月的时间内，选手们将在比赛中提升自己的 AI 的实力水平参与初赛。决赛预计将于春季学期第 12 周周末（2020 年 5 月 9 日）进行，前 8 名队伍将获得丰厚的奖金奖励以及课程加分，详情见下文的奖励办法。

4.2 报名方法

5 奖励办法

Part II

比赛规则

作为塔防游戏，每个势力在开始时各在一角拥有一座塔，塔周围的一定区域是自己的领地。玩家需要利用塔的生产力，完成生产兵团或升级塔的任务。生产出的作战兵团可以在场上移动，攻击其他势力的塔或兵团（减少他们的生命值）；生产出的工程兵团则可以完成修改地形、修理塔（恢复塔的生命值）的任务。其中，塔的等级越高，就会拥有更高的生产力、

战斗力、生命值上限。塔也可以攻击敌方的兵团，如果塔内有己方兵团驻扎，则塔的战斗力会更强。

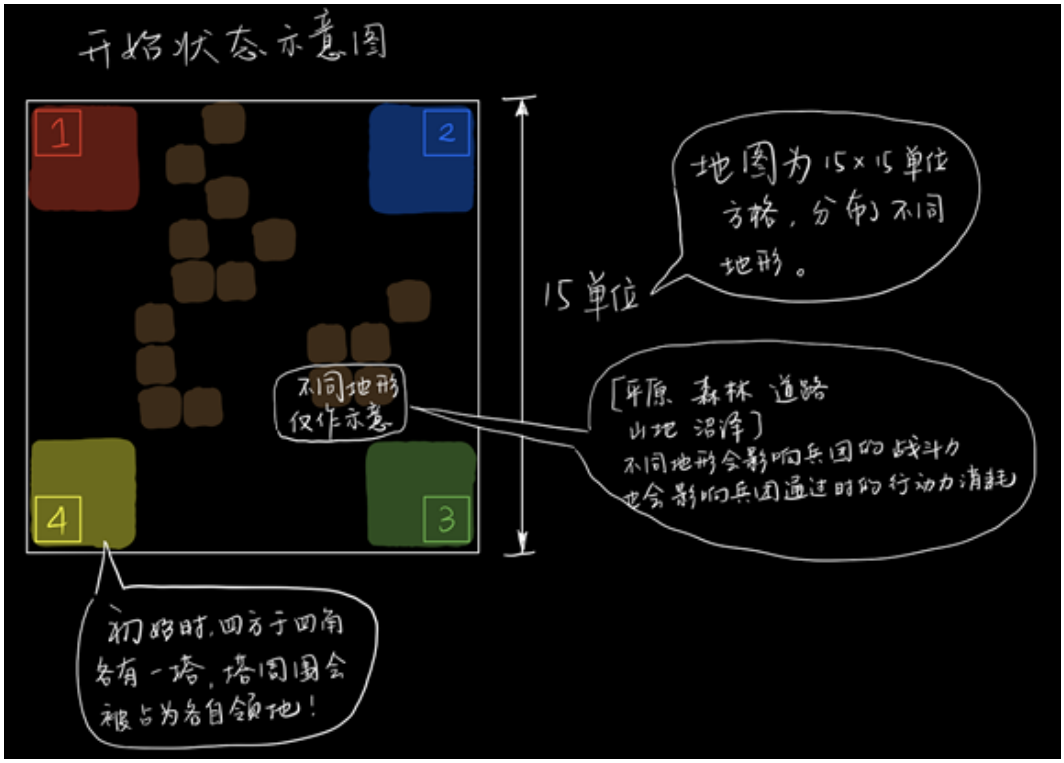

图 1: 游戏场地示意图：开始状态

6 角色说明

游戏中有以下角色：塔 *Tower*，作战兵团（分为战士 *Warrior*、弓箭手 *Archer*、法师），工程兵团（分为建设者 *Builder*、开拓者 *Extender*）。

6.1 塔

6.1.1 概述

塔是在地图上固定的，具有对一定范围内（距离为 2 及以内，即 5*5 的方形内）敌对势力塔或兵团自动攻击能力的建筑，每个势力最初时拥有 1 座塔，每个地图方格内最多有 1 座塔，每个势力最多拥有 10 座塔。当某势力没有塔时，该势力将被判为失败。地图上每个方格都有属于每个势力的占有属性值。每回合都将进行方格所属势力的判定，方格将属于 4 个势力中对该方格占有属性最高的一方。占有属性值由修建防御塔产生。防御塔可向周围

地图附加己方的占有属性值，该属性值随着与防御塔距离的增加而衰减，并在 5 格外衰减为 0，具体数据见表1。若有 2 方占有属性值同为最高，则该方格判定为过渡区域，过渡区域不允许建筑防御塔。

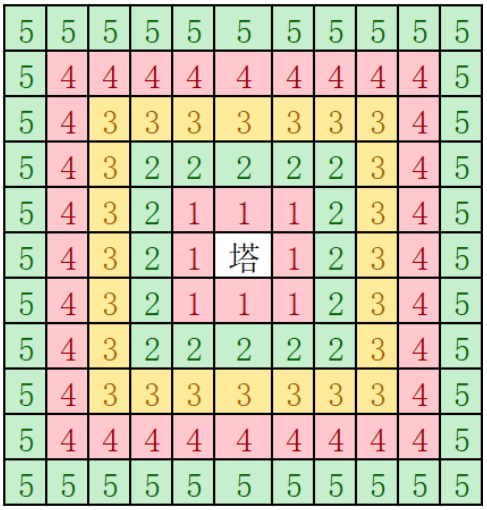

图 2: 距离计算示意图

表 1: 塔周围领地占有值

距防御塔距离 d	0	1	2	3	4	5	6+
施加占有属性值	inf	100	80	50	20	10	0

塔有以下属性：

(1) 等级 N。

(2) 生产力 W_N。

(3) 等级生命力上限 HP_N，当前生命力 hp。

(4) 等级战斗力 F_N，实际战斗力 f。

```
1
2 struct TowerInfo {
3
4   TTowerID      ID;    //防御塔ID
5   TPlayerID     ownerID;  //所属玩家ID
6   TPoint        position;    //位置
7   TProductPoint productPoint;  //生产力
```

```
  TProductPoint productConsume;  //当前生产任务尚需完成的剩余生产力值
  TBattlePoint  battlePoint;    //战斗力
  THealthPoint  healthPoint;    //生命值
  TLevel        level;         //等级
  productType   pdtType;      //当前生产任务类型
};

```

防御塔结构体

6.1.2 具体说明

1) 塔的等级 N 为 1-8 的正整数，从等级 N 升级到 $N+1$ 需要消耗的生产力值为 $40 \cdot N$。

2) 塔会根据自身的等级状况获得一个生产力数值 W_N，代表其一个回合能产生的生产力，具体的数据如表2所示。生产力 W_N 每回合更新，如果该回合没有使用生产力，下一回合不会累积。在同一时刻，塔可以选择一种生产任务（包括生产战士、生产弓箭手、生产法师、生产建造者、生产开拓者、升级塔自身）每一种任务需要消耗不同的生产力点数，具体情况见表3。兵团生产任务完成之后，塔所在的方格将立即生成一个对应兵团。防御塔所在方格内允许存在多个兵团，不必遵循同一单元格仅能存在一个兵团的限制。在 struct TowerInfo 中，可以访问到该塔当前未完成的生产任务 productType pdtType; 和该任务有待完成的工作余量 TProductPoint productConsume;

3) 等级生命力上限 HP_N 仅与等级正相关，具体的数据如表2所示。实际生命力 hp 会受到进攻而减小（具体结算方式如式2和式5所示）、由于建设者的维护而增加。当实际生命力 hp 被进攻方降低至 0 以下时，对方获得 5 分的击杀分，我方丧失该塔，塔等级下降 4 级：等级下降后，若不足 1 级，则塔消失，该单位恢复原来地形；若塔尚存在，则降级后的塔主权归对方所有（塔被俘虏）。

4）实际战斗力 f 为战斗中的战斗力。f 是 F_N、当前生命值 hp、等级生命值上限 HP_N、兵团驻扎情况带来战斗力增益 f_c（具体增益规则如表4所示）的函数。具体计算方法如下：

$$f = F_N \cdot \frac{hp}{HP_N} + \Sigma f_b \tag{1}$$

兵团攻击塔时，生命值的结算方式如下：

$$\begin{aligned} hp_{\text{塔}} \ -= \quad & 30 \cdot k_c \cdot e^{0.04(f_{\text{兵}} - f_{\text{塔}})}; \\ hp_{\text{兵团}} \ -= \quad & 28 \cdot e^{0.04(f_{\text{塔}} - f_{\text{兵}})}, \text{当兵团非弓箭手} \\ hp_{\text{兵团}} \ -= \quad & 0, \text{当兵团为弓箭手} \end{aligned} \tag{2}$$

塔只能攻击距离为 2 及以内的兵团。塔攻击兵团时，生命值的结算方式如下：

$$\begin{aligned} hp_{\text{兵团}} \ -= \ & 30 \cdot e^{0.04(f_{\text{塔}} - f_{\text{兵}})}, \text{对于所有兵团种类} \\ hp_{\text{塔}} \ -= \ & 0 \end{aligned} \tag{3}$$

5）兵团驻扎（一定要给兵团添加过驻扎到塔的命令）到塔，则被塔护卫。如果兵团驻扎到塔，则对该兵团发起的进攻会与塔结算，而不是与该兵团。但是，如果进攻发起方是防御塔，服从之前的规则，不存在塔与塔结算这一说，这次攻击兵团判定失败（塔不可以攻击驻扎在别人塔中的兵团）。具体见9。

你可以为塔添加以下操作：

(1) 生产：在命令中指定命令类型为塔命令，塔操作类型为生产，塔 ID，塔生产任务类型（<enum productType> ）。

(2) 攻击：在命令中指定命令类型为塔类型，塔操作类型为攻击，塔 ID，塔的攻击对象序号，即可指定某一做塔攻击某一个对象。

另外，请注意：

(1) 每回合，每座塔最多可以添加一个操作。如果该回合玩家给一个塔添加了多个操作，则第二个及以后的操作会被自动忽略，且占用 50 个操作的余额（相当于填了废志愿），所以请不要添加大于一个操作。

(2) 如果你的操作是无效操作，则也不会起任何作用。无效操作包括试图攻击自己的对象、试图攻击超出攻击范围的对象、不存在对应序号的对象等。

(3) 如果上一回合塔选择了生产，但未完成该生产任务，本回合选择攻击，则上一回合的生产进度会被保留，这样在下一回合假如继续添加生产该任务的命令，会在上一回合基础上继续生产。

(4) 若防御塔的某个生产任务尚未完成，玩家又指定了新任务，则未完成的生产任务的完成度将被缓存起来，然后进行新的任务。之后再选择有一定完成度的任务的时候，只需要完成未完成的部分，即完成剩余的生产力消耗值。即如果上一回合塔选择了生产任务 A，但未完成该生产任务 A，本回合选择生产任务 B，则上一回合的生产进度也会被保留，这样在下一回合假如继续添加生产任务 A 的命令，会在上一回合基础上继续生产。但是，由于接口所限，你只能通过 TowerInfo 访问到最近一次尚未完成的任务类型和余量，所以我们推荐一旦开始一种生产任务，中途不要切换别的生产任务（虽然你仍可以正常地生产它们）。

6.2 作战兵团

在同一个时间，同一个地图方格（防御塔所在方格除外）内的作战兵团、工程兵团数量各自不能超过 1 个。当某个方格内同时存在一个势力的一个作战兵团和工程兵团，则称工程兵团被作战兵团护卫。任何战斗，都优先与作战兵团结算。若在某次战斗中，该护卫队中的作战兵团阵亡。此时进行一次判定，若敌方作战兵团也在作战中阵亡，则护卫队中的工程兵团仍属于原来的玩家；若敌方作战兵团未阵亡，则护卫队中的工程兵团将被敌方俘虏，所属玩家变更为敌方。

6.2.1 概述

兵团有三个种类：战士（近战单位）、弓箭手（远程单位）、法师（高级进攻单位）。其中，弓箭手可以远程轰炸其他单位且自身不受伤害。法师具有较高的移动力。

作战兵团有以下属性：

(1) 行动力 M_c。

(2) 满血战斗力 F_c，实际战斗力 f。

(3) 生命力上限 HP_c，当前生命力 hp。

(4) 攻击距离 d_c。

(5) 所属玩家 ID。

```
15 struct CorpsInfo
16 {
17   //不需要，如果不存在就不录入信息了bool  exist;    //是否存在
18   TPoint  pos;    //兵团坐标
19   int   level;    //兵团等级
20   TCorpsID    ID; //兵团ID
21   THealthPoint  HealthPoint;  //生命值
22   TBuildPoint   BuildPoint;   //劳动力
23   TPlayerID   owner;      //所属玩家ID
24   corpsType       type;           //兵团种类
25   TMovePoint      movePoint;      //行动力
26   battleCorpsType   m_BattleType; //战斗兵用
27   constructCorpsType  m_BuildType;  //建造兵用
28 };
29
```

兵团结构体

6.2.2 具体说明

1）行动力 M_c 表示兵团在某一回合的进行行动的能力，具体见表4。兵团从一个单元格移动到另一个相邻单元格（即上下左右四个方向）将消耗一定行动力。（取决于单元格地形情况，计算方式为：一次移动经过的两个单元格的行动力消耗取平均值，并向上取整，具体见表5）值得注意的是，如果移动后行动力至少还有 1 点则则还能发起进攻，而一旦选择进攻将消耗所有行动力。行动力在新回合开始将重置。

2）生命力上限 HP_c 仅与兵团种类有关，具体见表4。实际生命力 hp 会受到进攻而减小（具体结算方式如式5所示），直至实际生命力 hp 被进攻方降低至 0 以下时，兵团死亡，对方获得 5 分的击杀分。

3）实际战斗力 f 为战斗中的战斗力。f 是满血战斗力 F_c、当前生命值 hp、生命值上限 HP_c、兵团所处地形情况带来战斗力增益 f_t（具体增益规则如表5所示）的函数。具体计算方法如下：

$$f = F_c \cdot \frac{hp}{HP_c} + f_t \tag{4}$$

兵团只能攻击在攻击范围内的对象，不同兵团的攻击距离 d_c 如表4所示。兵团 B 受到兵团 A 攻击时，生命值的结算方式如下：

$$\begin{aligned} hp_B -= &\quad 30 \cdot e^{0.04(f_A - f_B)}; \\ hp_A -= &\quad 28 \cdot e^{0.04(f_B - f_A)}, \text{当兵团 A 非弓箭手} \\ hp_A -= &\quad 0, \text{当兵团 A 为弓箭手} \end{aligned} \tag{5}$$

4）补充说明：若 A 兵团发起对 B 兵团的进攻操作且 B 兵团被消灭，A 兵团存活，则 A 兵团（除弓箭手外）会移动到 B 兵团所在的方格。若 A 兵团为弓箭手则不会移动。

作战兵团能进行的操作有：在地图上移动、驻扎己方势力的塔、对敌方军团发起进攻、对敌方防御塔发起进攻。

6.3 工程兵团

工程兵团分为：建造者、开拓者。建造者用来进行特定的工程建造，而开拓者则用于修建新的防御塔。工程兵团是脆弱的功能性单位。在工程兵团没有受到作战兵团的护卫时，任何敌方作战兵团对其的进攻操作，会直接将其俘虏。俘虏时，直接将所属玩家更改为发起该次进攻的作战兵团的所属玩家，本回合就可以直接操控它。

工程兵团与作战兵团共用 CorpsInfo 结构体。其中，兵团坐标、兵团 ID、行动力 M_c、所属玩家 ID 为共有属性；生命值 HP、战斗兵兵种为作战兵团属性；工程兵兵种为工程兵团属性。

1）行动力 M_c 表示兵团在某一回合的进行行动的能力，具体见表4。在移动后，建造者、开拓者的行动力至少为 1 的情况下才能进行建造的操作。且一旦进行建造的操作，行动力都会被清空，直到下一个回合才会重置。

6.3.1 建造者

开拓者有关参数为行动力 M_c，劳动力 B 和玩家所属 ID。

2）劳动力 B 是建造者特有的属性值，表示能够进行工程建造的次数。建造者的劳动力大小初始值为 3。每回合可以消耗劳动力，对所在的单元格进行某项工程建设。建造者发起操作后，建造者扣除一点劳动力。若劳动力为 0，则该建造者单位立刻消失（阻塞赋值）。

3）建造者可以进行地形修改（只能实现平原-森林的互换）和防御塔维修（单次修理将恢复防御塔该等级最大生命值的 1/3（向下取整））两种操作，两种操作各自需要 1 点劳动力消耗。发起地形修改操作时，建造者必须位于欲修改地形的方格上。发起防御塔维修操作时，建造者必须位于欲维修的防御塔的方格上。地形修改是我方小回合所有命令输入结束统一修改（相当于数电非阻塞赋值），而生命值修复则是立刻进行（相当于数电阻塞赋值），修复后当回合之后塔的命令可以用新的生命值。

6.3.2 开拓者

开拓者有关参数为行动力 M_c 和玩家所属 ID。

4）开拓者可以进行防御塔建造的工作。在开拓者被生产出来时，生产其的防御塔等级下降 1（本来为 1 则不再下降）。开拓者可以在己方领土的任一无防御塔的方格上进行防御塔建造，发起建造操作时必须位于目标单元格上。建造操作是立即完成的，且会使得开拓者单位立刻消失。

7 计分规则

在游戏进程中防御塔数量降为 0 的玩家，判定出局。第一位出局的玩家获得最低位次，第二位出局的玩家获得次低位次，依次类推。

当游戏进行至 300 回合后，场上还未出局的玩家将按照得分进行排名。

1. 防御塔得分：单个防御塔得分 = 防御塔等级数 *10。防御塔得分为所有单个防御塔得分之和。

2. 兵团得分：单个兵团得分 =4 分。兵团得分为所有单个兵团得分之和。

3. 击杀分：每当消灭一个敌方的作战兵团/塔/俘虏一个敌方的工程兵团时，都可以得 5 分。

得分相同的按防御塔攻占数、消灭敌方军团数、俘虏敌方军团数排名。若再相同随机决定排名。

另外规定：塔最多 10 座，作战兵团最多 10 个，工程兵团最多 10 个，一次最多添加 50 个命令。(如果己方兵团已经有 10 个，此时俘虏了一个敌方兵团，则敌方兵团直接消失，而不是转换为我方兵团。塔同理。)

8 你需要做什么

8.1 概述

所有玩家的 AI 都可以从 Info 中读取当前场上各方势力的兵团、塔的信息，并设计算法，并按照统一的接口 CommandList 给裁判程序回传命令，操控己方势力；在游戏中，选

手只需要在 ai.cpp 文件中的 void player_ai(Info& info) 函数中填写自己的代码，并最终只需要提交 ai.cpp 文件。

```
30 class CommandList
31 {
32 public:
33   void addCommand(commandType _FC18type, initializer_list<int>
   _FC18parameters);
34   void removeCommand(int n);  //【FC18】移除第n条命令
35   vector<Command> getCommand() { return m_commands; }  //【FC18】获取所有
   命令
36 };
```

添加命令有关代码

8.2 玩家添加命令示例

玩家通过 info.myCommandList.addCommmand(< 命令类型 >,< 参数列表: 参数 1, 参数 2...>) 添加命令。

8.2.1 塔

防御塔攻击兵团:info.myCommandList.addCommmand(towerCommand, {TAttackCorps, 本塔 ID, 目标兵团 ID})

防御塔设定生产任务:info.myCommandList.addCommmand(towerCommand, {TProduct, 本塔 ID, 生产任务类型 (见下方枚举类型)})

```
37 enum productType
38 {//                                                    生产回报
39   PWarrior = 0,       //生产战士          1star-战士兵团
40   PArcher = 1,       //生产弓箭手       1star-弓箭手兵团
41   PCavalry = 2,       //生产骑兵          1star-骑兵兵团
42   PBuilder = 3,       //生产建造者         1-建造者兵团
43   PExtender = 4,       //生产开拓者         1-开拓者兵团
44   PUpgrade = 5,       //塔升级任务        塔等级+1（max=8)
45   NOTASK = -1
46 };
```

生产任务类型

说明：新建的兵团需要从下一回合开始可以起作用。

8.2.2 作战兵团

移动：info.myCommandList.addCommmand(corpsCommand, {CMove, 本兵团 ID, 方向（Cup / Cdown / Cleft / Cright）})

兵团攻击兵团:info.myCommandList.addCommmand(corpsCommand, {CAttackCorps, 本兵团 ID, 目标兵团 ID})

兵团攻击防御塔:info.myCommandList.addCommmand(corpsCommand, {CAttackTower, 本兵团 ID, 目标防御塔 ID})

兵团驻扎在己方塔: info.myCommandList.addCommmand(corpsCommand, {CStation-Tower, 本兵团 ID })

说明：所有兵团添加进攻命令之后，之后添加的移动或其他进攻命令均会被忽略。在单个回合中，对于单个作战兵团，仅能添加：若干移动指令（也可以不移动）+ 一个攻击命令或驻扎己方塔的命令（二选一。添加攻击命令需要还有剩余 >0 的行动力，而添加驻扎命令则不需要。如果移动命令完成后，行动力为 0，则该回合无法进攻，但可以驻扎。如果输入的命令无效，例如不在攻击范围内、移动超出场地范围、攻击命令下达时没有剩余行动力等，无效命令会被直接忽略。）

8.2.3 工程兵团：建造者

移动：info.myCommandList.addCommmand(corpsCommand, {CMove, 本兵团 ID, 方向编号（Cup / Cdown / Cleft / Cright）})

修复防御塔: info.myCommandList.addCommmand(corpsCommand, {CRepair, 本兵团 ID})

修改地形：info.myCommandList.addCommmand(corpsCommand, {CChangeTerrain, 本兵团 ID, 目标地形 (见下方枚举类型)})

说明：仅支持地形“平原-森林”之间的相互转换。在单个回合中，对于单个建造者，仅能添加：若干移动指令（也可以不移动）+ 修复防御塔/修改地形（二选一。此时需要还有剩余 >0 的行动力。如果移动命令完成后，行动力为 0，则该回合无法修复/修改。如果输入的命令无效，无效命令会被直接忽略。）

修改地形需要在该回合结束之后统一起作用。如果该回合两个兵团对同一个地形执行了更改，以最后一个传入的更改为准。

```
47 enum terrainType
48 {
49   TRTower = 0,        //塔
50   TRPlain = 1,        //平原
51   TRMountain = 2,        //山地A
52   TRForest = 3,        //森林
53   TRSwamp = 4,        //沼泽
```

```
54    TRRoad = 5,         //道路
55 };
```

生产任务类型

8.2.4 工程兵团：开拓者

移动：info.myCommandList.addCommmand(corpsCommand, {CMove, 本兵团 ID, 方向（Cup / Cdown / Cleft / Cright）})

建立新防御塔：info.myCommandList.addCommmand(corpsCommand, { CBuild, 本兵团 ID })

说明：在单个回合中，对于单个开拓者，仅能添加：若干移动指令（也可以不移动）+ 建立新防御塔（此时需要还有剩余 >0 的行动力。如果移动命令完成后，行动力为 0，则该回合无法新建防御塔。一旦执行了新建防御塔命令后，兵团即消失。建立新防御塔的条件：该方格是己方领地，该方格尚未建立防御塔。）

新建的防御塔需要从下一回合开始可以起作用。

9 规则具体实现逻辑

(1) 兵团被攻击时，裁判程序的判定逻辑为：首先判断目标兵团是否驻扎到塔。若是，则与塔结算（但是，如果进攻发起方是防御塔，服从之前的规则，不存在塔与塔结算这一说，这次攻击兵团判定失败。塔攻击驻扎别人塔里的兵团是废命令，被忽略）；若否，则判断，是否是工程兵且被作战兵护卫。然后判断目标兵团是否是工程兵且被作战兵护卫。若是，则与作战兵结算，若否，则直接与目标兵团结算。

(2) 攻击塔命令（塔被占领或摧毁的情况）：塔所在方格的所有敌方兵团（不允许兵团进入别人塔所在方格，因此塔中所有兵团都是本塔方的兵团），都直接杀死，不管是否驻扎到塔。

(3) 攻击塔命令（塔被占领的情况）：若发起攻击的兵团与被攻击的塔同时生命值减为 0，则不会由攻方占领塔。这一点，我们与兵团攻击兵团时同时死亡的情况保持一致。

10 相关数据表

表 2: 塔等级表

等级 N	生产力 W_N	等级战斗力 F_N	等级生命力上限 HP_N
1	10	25	100
2	15	27	120
3	20	29	140
4	25	32	160
5	30	35	180
6	35	38	200
7	40	41	220
8	45	45	240

表 3: 塔生产任务表

生产任务	所需的生产力值
战士	40
弓箭手	60
法师	100
建造者	40
开拓者	40
升级（N 升级到 N+1）	N*40

表 4: 兵团参数表

兵种 Crops	战士	弓箭手	法师	建设者	开拓者
战斗力增益系数 f_c	2	2	4	NA	NA
攻城系数 k_c	0.4	0.7	0.5	NA	NA
攻击距离 d_c	1	2	1	NA	NA
行动力 M_c	2	2	4	2	2
生命力上限 HP_c	60	50	70	NA	NA
满血战斗力 F_c	36	30	44	NA	NA
劳动力 B	NA	NA	NA	3	NA
m_BattleType	0	1	2	NA	NA
m_BuildType	NA	NA	NA	0	1

表 5: 地形参数表

地形	平原	山地	森林	沼泽
地形战斗力增益 f_t	0	5	3	-3
地形行动力消耗 m_t	2	4	3	4