Synthesizing Action Deceptive Strategy in Two-player Strategy Computer Game: A Case Study

Mohammed Tousif Zaman, Abhishek N. Kulkarni and Jie Fu

May 16, 2020

1 Abstract

We extend a typical relic retrieval scenario from the popular game of Age of Empires II (AoE) to demonstrate the application of action deception. In this scenario, a team of three units of player 1 (P1)—a villager, a knight and an archer—is tasked to retrieve a relic from the enemy's (P2) territory, which is being protected by three P2 watch towers. The challenge presented by the scenario is that neither the villager can retrieve the relic by himself (as he would be killed by the watch towers) nor can the armed units destroy the watch towers by themselves. In this situation, we compute an action deceptive strategy for P1, which ensures almost-surely (with probability one) that P1 will be able to retrieve the relic. The key advantages of this approach are that it allows efficient use of game resources—P1 can retrieve the relic using 3 units, whereas the traditional AI would end up sending an army of 15 units to take down towers—as well as it relieves the traditional `PlayerManager` from the complex task planning for individual units. This provides a better and more "intelligent" game AI design.

2 Relic Retrieval Scenario

AoE II: Introduction Age of Empires II (AoE) is a real-time strategy game designed by Microsoft Age of Empires Wiki. The game consists of several players (AI and/or human) who compete with each other to fight and defeat their enemies. The player who vanquishes all their enemies wins the game. There are several complex winning conditions in AoE that involve collection of relics (a special object in AoE), defeating all enemies in a war and so on. In this work, we shall draw inspiration from one winning condition, where the player who retrieves all the relics wins.

Limitations of AI in AoE Traditionally, the game AI consists of a script called `PlayerManager` which is tasked with managing the economy (spawning new villagers, assigning roles to villagers, managing trade, buildings etc.), military (strategically assigning units, decide when and where to attack etc.) and so on. This happens in a sequential manner wherein during every turn of i-th player the AI assigns a task along with the necessary resources, trajectory and action queue to a single unit or a platoon. This often results in an inefficent behavior of the units that resembles a brute force approach rather than an intelligent one (see Top 10 Things AoE2 Player Hate and similar).

Relic Retrieval Scenario In a game where all players compete to retrieve all relics to win, it is customary for the first player finds a relic to protect it using watch towers Watch Tower Placements. A watch tower is a type of a military building which fires arrows at the (arbitrarily chosen) enemy units within its range. This allows the player to gather sufficient resources to enable the feature that allows the player to collect the relic.

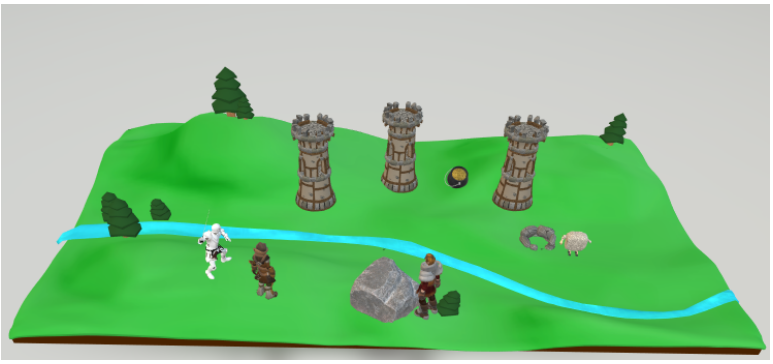

Figure 1: Representation of the scenario

In this work, we consider one such scenario in which P1 has three units—a villager, a knight and an archer—who are tasked by `PlayerManager` to retrieve the relic that is protected by three watch towers of P2 as seen in Figure 1.

Deviating from the traditional AoE gameplay, we allow the `PlayerManager` to allocate initial resources that the team may use at its discretion to accomplish the task. Furthermore, we allow the villager to gather more resources while on the mission which the team can use, again at their discretion. These extensions are in contrast to the traditional AoE gameplay, where all gathered resources and their expenditure must be planned by the `PlayerManager`. We find this modification to be intuitive as it separates team-level mission planning from player-level mission planning, thereby estabishing a hierarchy of decision-makers which is commonly observed in administrative structures in real-life.

We now formalize the gameplay and player capabilities and objectives below. Figure 2 summarizes different regions in the game.

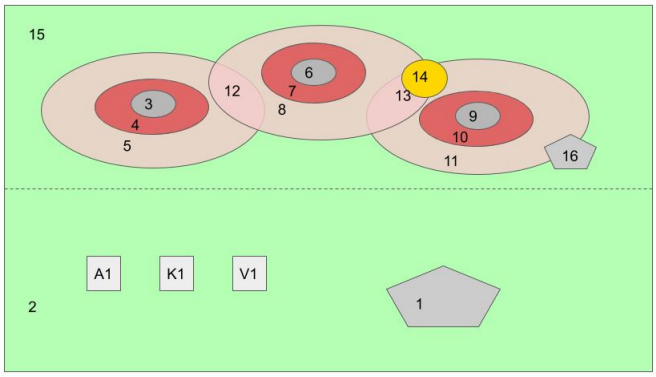

Figure 2: Different regions in the scenario

- There are two players; P1 and P2, in the game. P1 has three units, namely a villager, a knight and an archer. P2 has three watch towers as seen in Figure 2.
- The world[1] is divided into two regions, P1's territory (region 2) and P2's territory (region 15).
- There are two stone mines in the world, one is located in P1's territory (1) and the other in P2's territory (16).
- The stone mine in P2's territory (16) is within the range of at least one watch tower (9).
- The relic (14) is located in P2's territory and is protected by at least two watch towers (6 and 9).
- P1's team starts with 100 stone and 100 health for each unit.
- Villager is capable of building a tower or a castle. A tower costs 150 stone and a castle costs 500 stone.
- Neither knight nor archer is capable of destroying a watch tower by themselves or together.
- The P2's territory (region 15) is split in two parts by the range (regions 5, 11). That is, no unit can cross behind the enemy lines without being attacked by watch towers.

[1]Here, by world we mean the region in which the team is required to stay within. Such a constraint is desirable as violating it could result in other consequences such as P2 initiating a war at another location or similar.

- The terrain is such that P1 cannot build a watch tower or a castle in P1's territory and in P2's territory, the only place to build a castle is within a watch tower's range (regions 8, 11, 13).
- Each watch tower has two regions: inner region with greater accuracy (-10 health per hit) and outer region with lower accuracy (-5 health per hit).
- A watch tower begins to attack enemy unit if it trespasses the protected outer/inner region. If there are multiple units in the protected regions, the unit that poses higher threat to the tower is more likely to be attacked and so on. The threat posed by P1 is ordered as: Knight > Archer > Villager.

Given the game rules, we discuss how an action-deceptive strategy could be successful in collecting the relic, despite it being protected by watch towers. Note that, a traditional AoE AI would require around 15 infantry units to bring down a set of 3 watch towers.

Example of Successful Strategy As described in scenario description, the P1 team has no trivial strategy to retrieve the relic. In fact, the only way a team of 3 can succeed in their task is by building a castle in the range of watch tower. As castle has very large amount of hit points, castle would destroy all three watch towers.

However, building a watch tower is non-trivial. Because, if P2 knows all action capabilities of P1, then a winning strategy of P2 would be to kill villager first, whenever villager is in the range. Thus, the initial state of the game is, in fact, losing for P1 in the game with perfect information.

However, in our scenario we assume that P2 does not know that *P1 can build a castle*, and that *P1's archer can suicide to give bonus health to villager*. Ignorant of these two actions, P2 would assess more risk to watch towers is from knight and archer. Thus, it would choose to target them over the villager, thereby allowing villager to safely be smuggled into P2's territory—*a deceptive move*.

In our scenario, we assume P1 does not have enough resources to build a castle. Thus, the villager must collect stone from both the stone mines before building a castle. When P1 collects stone from first mine (either region 1 or 16), P2 does not update his risk assessment. However, when P1 mines the second stone mine, P2 knows that P1 is trying to build a castle and assign greatest risk to be from a villager. Thus, the P1 team must prioritize collecting stone from 16 before 1. This is a decision that *should* be made automatically by the action deception algorithm.

Finally, we note that it is not possible for the villager to complete building the castle without dying. Hence, the suicide action of Archer/Knight can be used to give a health boost to villager. Again, if Archer commits suicide to boost the Knight, P2 would become aware and choose to kill villager before any other unit.

3 Formal Modeling of Game Scenario

To define a formal model of the game scenario, we must define an implementation of (a) transition system, and (b) specification of P1. We start by formalizing the specification first, as it will help us define the set of AP required to define transition system.

Specification: Informally, P1's task is to retrieve the relic and bring it back to P1's region. In LTL, this can be written as follows,

$$\varphi = \Diamond(\mathsf{R}_{\mathsf{relic}} \wedge \Diamond \mathsf{R}_{\mathsf{P1}}),$$

where $\mathsf{R}_{\mathsf{relic}}$ and R_{P1} are two APs representing whether the villager is in region 14 and in region 2, respectively.

Transition System: The game scenario can be captured as a transition system,

$$TS = \langle S, Act, T, AP, L \rangle,$$

where

- Every state in the set of states, S, is represented as the tuple, $s = (s_1, s_2)$ where

  $s_1 = (\mathsf{pos}_{\mathsf{villager}}, \mathsf{pos}_{\mathsf{knight}}, \mathsf{pos}_{\mathsf{archer}}, \mathsf{health}_{\mathsf{villager}}, \mathsf{health}_{\mathsf{knight}}, \mathsf{health}_{\mathsf{archer}}, \mathsf{res})$,
  $s_2 = (\mathsf{health}_{\mathsf{T1}}, \mathsf{health}_{\mathsf{T2}}, \mathsf{health}_{\mathsf{T3}})$

  where res remembers whether villager has collected resources from region 1, region 16 or not.

- The set of actions, Act, is tabulated in Table 1.
- The transition function is implicitly captured by the defined actions.
- $AP = \{\mathsf{R}_{\mathsf{relic}}, \mathsf{R}_{\mathsf{P1}}\}$ is the set of atomic propositions.
- $L : S \rightarrow 2^{AP}$ is the labeling function such that $\mathsf{R}_{\mathsf{relic}}(s) = \top$ if the $s.\mathsf{pos}_{\mathsf{villager}} = 14$ and $\mathsf{R}_{\mathsf{P1}}(s) = \top$ if the $s.\mathsf{pos}_{\mathsf{villager}} = 2$.

Unit	Action List
Archer, A	attack, move, suicide, NOP
Knight, K	attack, move, NOP
Villager, V	build, move, collect, NOP
Watch Tower, T	attack, NOP

Table 1: Representation of units and available actions

NOP: stands for No Operation. The unit just remains in rest at its location.

Combinatorially, the size of the transition system state space can be computed as

$$|S| = 16^3 \times 3^3 \times 4 \times 3^3 \times 2 \times 3 \times 8 = 191102976 \approx 19 \times 10^6$$

Note: Calculation considers turn, DFA and I-Graph states.

It is possible to use some domain specific simplifications such as "whenever villager is in regions 1, 2, 15, the archer and knight will also be within either of the three regions" reduce the state space and further reducing one of the watch towers decreases the states, substantially. For instance, with above observation, a tighter bound on size can be given as

$$|S| = 10^3 \times 3^3 \times 4 \times 3^2 \times 2 \times 3 \times 8 \approx 46.65 \times 10^6$$

Finally, instead of using combinatorial procedure, we plan to construct an initialized transition system iteratively. This only adds the reachable vertices in the graph to the transition system. We expect this to reduce substantially the size of transition system.