Intelligent Systems 2013

Planet Wars Bots



**Authors**

Anouk Boukema

Baudouin Duthoit

Gossa Lô

Mengxiao Qian

# Abstract

of 2-3 paragraphs

# Introduction

Introduction (to the problem, but also your solution, en some results. 2 pages)

# Background information

Background information: description of the game, the challenge, the IS framework, whatever is necessary to understand your paper. Here you would normally also summarise related work, but this is not required here as all the methods are in the textbook (1-2 pages)

## The game

In the Fall of 2010 there has been a google AI challenge. This challenge (probably) inspired some teachers at the VU which thought if they would simplify the game given by the contest they could change it into a perfect exercise for second years students of informatic related studies. This is how the game Planet Wars, which we used for this report, came into existence.

The creators of the Google challenge based their game on another game colleged Galcon which has been created by Phil Hassey. So indirectly this is also were our game came from.

### Galcon

As described on the Galcon site: “Galcon is an awesome high-paced multi-player galactic action-strategy game. You send swarms of ships from planet to planet to take over the galaxy.”

The gameplay is fairly easy. You control everything by mouse. You can select one or more of your own planets with which you want to attack. Then you just select the planet you want to attack or reinforce. By scrolling you can change the percentage of ships you want to send from your source planet to another planet. For example when you have 100 ships and you need 60 ships to concur an enemy planet you should change your percentage to 70, 80 or 90. Changing the percentage can be done at any time and as often as the player likes. The is not played in turns and therefor attacks can be made at any moment. Also, planets which are in possessed will gain the player more ships over time. The bigger the planet, the higher time rate the planet will give you a ship.

The goal of the game is to erase your enemy from the map.

### The simplification: Planet Wars

For the IA competition and for our course they simplified the game Galcon. The major change is that there isn’t any variation possible in the percentage of ships send from a planet. This percentage is now always 50%.  
The other adaptation is that Planet Wars plays in turns. Bigger planets still give more ships each turn, although this now doesn’t depend on the creation rate of the ships but of the growth rate of a planet. So each turn a possessed planet will generate a fixed number of new ships (dependent on the growth rate number of that planet). There are two possibilities when playing in turns.

#### Version 1 name???

Here each turn equals one move of a player. This means the other player can see the move his enemy made and adapt to that.

#### Version 2 name???

Now, for one turn both players have to make their moves. This means that they won’t know what which planet the other player chooses to attack. If both of the players choose the same planet to attack, they will start at the same time. Whom will win the planet is now dependent of the length from the source planet to the destination planet. The one that arrives first will concur it, but the one that arrives second might take it over again.

## The challenge

## The IS framework

## More???

# Research questions

Research question: Explain what you did, and what possible outcomes of your setup and contribution. That could e.g. be that you want to find out whether one methods works, or that it works better than other.  What do you mean by works better: Wins once, wins all the time, wins mostly, not alway loose, works faster, works better when time is restricted etc. (1 page)

Is a non-deterministic bot necessary to win planet wars?

Does the efficiency of a bot depend on the maps or the number of planets ?

Can we make a bot that wins a game without passing the time limit once?

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Wins 12/12 | tie | Crashes | Passed time limit | Ratio of planets concurred | Number of moves | Winning time 2/2: RandomBot | | Winning time 2/2: BullyBot | | Winning time 2/2: LookaheadBot | |
| FirstBot |  |  |  |  |  |  |  |  |  |  |  |  |
| Hillclimbing |  |  |  |  |  |  |  |  |  |  |  |  |
| Beamsearch |  |  |  |  |  |  |  |  |  |  |  |  |
| Adaptive |  |  |  |  |  |  |  |  |  |  |  |  |

# Experimental setup

We have made 4 different bots. Each of them is specialised in a different way. We will explain per bot how they are specialised and on what theory there speciality is based on (out of the lectures)

Experimental setup: Explain how you set up your experiments. What did you do, e.g. in terms of implementation (brief), but mostly in order to compare your different methods. Define your metrics. (2 pages)

## FirstBot

At the start of this course we decided to first make a bot which wasn´t based on the exercises. Just to experiment. That´s how we created the FirstBot.   
FirstBot is a fairly easy bot which only attacks enemy planets. We decided to implement one more principle in the FirstBot. FirstBot should consider if there aren´t any better options than attacking the enemy. By a better option we mean that FirstBot will gain more ships when he attacks a planet than it had cost him to concur the planet. We concluded this was only possible when the growth rate of a planet was bigger than the ships it possesses.

### Implementation

## HillclimbingBot

The basics of this bot are based on the hillclimbing search principle. For this search method the agent compares all the heuristic values he can choose from and chooses the path of his best option. In our case we defined these heuristic values by parameter D which indicates the difference between the ships our bot possesses and the ships the enemy possesses. This D may differ for each possible planet our bot can attack.

D planetA  = HissLossA – MyLossA + MyGrowthA - HisGrowthA

This formula indicates that our bot will favour to attack a planet of the enemy because this generates

a high D rate. If our bot doesn’t attack one of the enemies ships the D will most possibly be a negative value. This is because most of the times the growth rate of a planet is lower than the number of ships it houses.

All the possible D values our bot can find differ over the number of planets in the game and the number of planets our bot 1possesses. Since 27 planets is the maximum, the most D values our bot possibly has to create are 14\*13 =182.

Our bot will attack with the planet that, in combination with (one of) his planets made the highest D value.

### Implementation

## Comparing the methods

# Results

Results: describe your results in some kind of overview tables, and point the reader to the most significant and interesting results in a short text. (2 pages)

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Wins 12/12 | tie | Crashes | Passed time limit | Ratio of planets concurred | Number of moves | Winning time 2/2: RandomBot | | Winning time 2/2: BullyBot | | Winning time 2/2: LookaheadBot | |
| FirstBot |  |  |  |  |  |  |  |  |  |  |  |  |
| Hillclimbing |  |  |  |  |  |  |  |  |  |  |  |  |
| Beamsearch |  |  |  |  |  |  |  |  |  |  |  |  |
| Adaptive |  |  |  |  |  |  |  |  |  |  |  |  |

# Findings

Findings: As a separate step interpret the results, and give explanations for the results. (1 page)

# Conclusion

Conclusions: summarise what you did, and highlight the most inportant findings. (1 page)

The paper should describe the chosen methods and compare them analytically and emprically. Based on this analysis you should draw some generic conclusions. For this you should take (at least) the 4 bots you implemented and compare their performance by having them systematically play against each other. According to the different environments (but possibly not) different bots might outperform others. Define some interesting hypotheses and research questions, and use your analysis to verify or falsify them

1. Title page with title and authors, and an abstract of 2-3 paragraphs
2. Introduction (to the problem, but also your solution, en some results. 2 pages)
3. Background information: description of the game, the challenge, the IS framework, whatever is necessary to understand your paper. Here you would normally also summarise related work, but this is not required here as all the methods are in the textbook (1-2 pages)
4. Research question: Explain what you did, and what possible outcomes of your setup and contribution. That could e.g. be that you want to find out whether one methods works, or that it works better than other.  What do you mean by works better: Wins once, wins all the time, wins mostly, not alway loose, works faster, works better when time is restricted etc. (1 page)
5. Experimental setup: Explain how you set up your experiments. What did you do, e.g. in terms of implementation (brief), but mostly in order to compare your different methods. Define your metrics. (2 pages)
6. Results: describe your results in some kind of overview tables, and point the reader to the most significant and interesting results in a short text. (2 pages)
7. Findings: As a separate step interpret the results, and give explanations for the results. (1 page)
8. Conclusions: summarise what you did, and highlight the most inportant findings. (1 page)

Evaluate everything as structured as possible. Let each bot fight other bots, to test their abilities. This has to be done for each bot we made and in the same systematic way.