s-birds: A Heuristic Engine based Learner Bot for the Angry Bird Problem

Sourish Dasgupta, Siddharth Kothari, Sacheendra Talluri, Saheb Motiani
Dhirubhai Ambani Institute of Information & Communication Technology, India
sourish_dasgupta@daiict.ac.in, sids.aquarius@gmail.com, undead.rokr@gmail.com,
saheb210692@gmail.com

Abstract

s-birds is an Angry Bird playing bot developed by the Agent Research Lab at DA-IICT, India. *s-birds* tries to mimic a human expert and is based on Naïve Agent (2012 Angry Birds Competition winner) through rote learning. It also learns by taking advice from a heuristic engine that follows a set of core principles of structural physics.

1 Introduction

s-birds[1] is an Angry Bird agent that has been developed by a section of the Agent Research Lab at DA-IICT, India. *s-birds* is a hybrid agent in the sense that: (a) it tries to *rote learn* from the performance of the Naïve Agent (2012 Angry Bird Competition winner) and two expert Angry Bird players (students of DA-IICT, India), (b) it tries to *learn by taking advice* from a Heuristic Engine developed by our team, and (c) it tries out newer possibilities by exploring more number of candidate trajectories. *s-birds* is greedy in the sense that it tries to maximize the score per bird (i.e. per shot) while killing maximum number of pigs per shot. It also has the capacity of adapting if it fails to qualify a particular level. The qualifying version of *s-birds* only has heuristic engine since there was an error correction that was needed to be done during convertion of trajectory angles into sling points. In the subsequent sections we describe each of the components of *s-birds* together with a general outline of how each component works.

2 Approach Overview

The principal components of *s-birds* are: (i) learner, (ii) terrain detector, (iii) heuristic engine, and (iv) trajectory calculator. The learner is responsible for: (i) rote learning the behavior of a trainer for all the 21 levels of the Angry Bird game, (ii) generate additional trajectories and collect the score for each for a given graphic structure of pigs, terrain, and blocks and a given bird type, and (iii) maps new structures to known structures for a given bird type during playing the game. The first two roles form the training phase

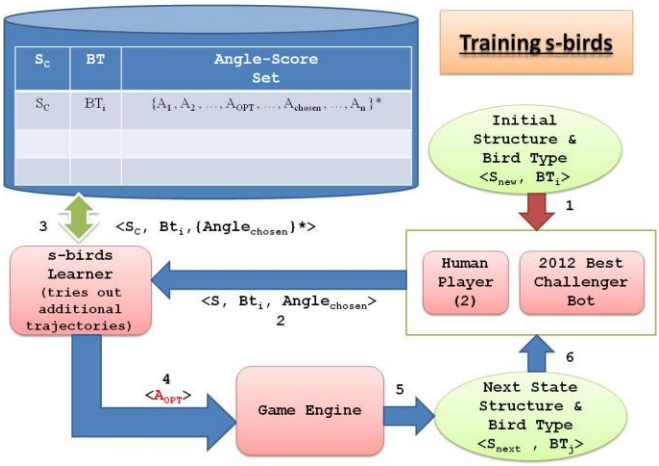

Figure 1: Training phase of *s-birds*

by the learner.

The heuristic engine is a simplified version of a physics engine and is based on some simple principles of physics. It has two aspects: (i) *Bottom Attack Principle* (BAP), and (ii) *Top Attack Principle* (TAP). BAP assumes that striking the most vulnerable block near the base of the structure would not only kill maximum number of pigs but will also cause maximum destruction to the structure by making the structure fall down like a pack of cards. The vulnerable blocks are called *pivotal blocks*. On the other hand, TAP assumes that striking a significantly displaceable heavy block at the top of structure can cause maximum damage by bringing this top block down on top of other blocks. Currently *s-birds* adopts BAP.

The trajectory calculator calculates the angle (and thereby the sling position) at which a bird should be shot to hit a pivotal block.

2.1 Enhanced Rote Learner

As mentioned earlier *s-birds* memorizes the behavior of its trainers in a table having three columns: (i) observed compressed structure, (ii) bird type used by trainer, and (iii) a <angle, score> pair where the score obtained is recorded along with the angle chosen by the trainer. This is done on every shot for every level. To enhance the training data the learner makes a range of 80 degrees up with respect to the

[1] The name of the bot comes from its creators' names all of which coincidentally start with an 's'.

line of sight and shoots the bird used by the trainer at all angles within that range that are in increment of 2 degrees. It records the <angle, score> pair of each of these shots.

2.2 Heuristic Engine

The heuristic engine calculates the pivotal block on the basis of a measure called *support*. The measure tries to understand the stability that is provided by any block to all the pigs. If support is high then the block cannot be a pivotal block since hitting it will not cause much damage. To calculate support we designed two measures: (i) *weighted block distance*, and (ii) *alignment score*. The weighted block distance of any block A to another block B is computed from the number of blocks that lies on the euclidean straight line joining the two blocks and the relative mass of each of these blocks. We assign the lowest mass of 0 to air colums (air blocks) and highest of 3 to stone blocks. Pigs are considered as blocks but during weighted block distance computation between a pig block and any other block we do not count the pig block itself. The higher is the weighted block distance from a block to a target block (say pig block) the lower is the probability of hitting the target block due to dampening of the force component that may propagate. Alignment score, on the other hand, measures the stable equlibrium of a block path from a source block to a destination block. Alignment of a particular block can be either vertical or horizontal depending on the length: breadth value. If a block path has more number of vertically aligned blocks (i.e. higher aligment score) then it has lower equilibrium than another path having more horizontal blocks (i.e. lower alignment score). A higher alignment score is preferable for a block path since that has higher probability of leading to greater destruction.

In this way the engine computes the support of a particular block to all the pig blocks and creates a vector of *m* dimension (a dimension corresponds to a particular support value for a particular block path) where *m* is the number of pigs and the values are sorted according to the position of the pigs from left to right. The angular distance between the *m* block paths is also calculated to assess how close the pigs are to each other. If the pigs are close enough with respect to a particular source block then the chance is high that picking that block as a pivotal block will cause maximum damage as compared to another source block which might have the pigs far apart with respect to itself (considering that both has equivalent support for all the block paths leading to the pigs). Finally, all the support vectors representing their corresponding source blocks are then compared with an imaginary best case base zero vector. This base vector corresponds to an imaginary block where all the pig blocks are coincidental. The engine then selects the block closest to the imaginary base block as the best pivotal block.

After selecting the pivotal block the engine then computes the *penetration distance* from the pivotal block to all the surface blocks. The penetration distance is a measure of how difficult it will be for a particular bird type to crush into the pivotal block by breaking through the surface block. This measure depends again on the weighted block distance value of the path from the pivotal point to a particular surface block and also on the type of the bird. A bigger and heavy bird can penetrate a particular block path that a lighter and smaller bird may not. Once the computation is over the engine selects the surface block that has minimum penetration distance. After that a trajectory is calculated by the trajectory calculator and is given as input to the learner. It may be so that the pivotal block is itself a surface block in which case this step is not necessary. Only the trajectory is calculated.

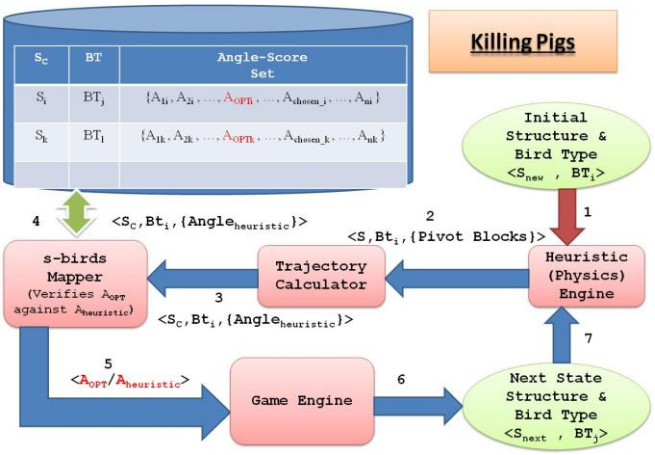

Figure 2: *s-birds* game playing phase

2.3 Learner at Play

The learner first observes a new structure and using the GMC converts it into a compressed structure. It then matches the structure and the bird type that it has to use in its rote learned table. If a sufficiently close match is found (*s-birds* uses a very tight threshold) then the learner mimics the behavior of trainer by picking the angle that creates maximum score for that structure and that bird type. If it fails to find a match then the learner turns back to the heuristic engine and feeds it with the new structure and the bird type. The heuristic engine returns a trajectory as an advice to the learner. The learner follows the trajectory and shoots. This entire process is repeated in the next shot where the destroyed structure becomes a new structure to be observed by the learner. If a learner fails any level then it knows that the fault was with the trajectory advised by the heuristic engine since the training data has no noise. In this situation it then chooses the next best pivotal block trajectory path as suggested by the learner. For all shots suggested by the heuristic engine it rote learns that trajectory for future reference and/or update.

3 Team Profile

The team consists of four members - Sourish Dasgupta, Siddharth Kothari, Sacheendra Talluri, Saheb Motiani.

3.1 Sourish Dasgupta

Sourish Dasgupta received his PhD degree in Computer Science from University of Missouri - Kansas City, USA in 2011. Currently he holds an Assistant Professor position in Dhirubhai Ambani Institute of Information & Communication Technology (DA-IICT), Gandhinagar, India. His research interests are in the areas of Service Oriented Computing, Distributed Multi-Agent Systems and Semantic Web.

3.2 Siddharth Kothari

Siddharth Kothari finished his B.Tech, ICT at Dhirubhai Ambani Institute of Information and Communication Technology (DA-IICT) in 2013. He wrote his first game agent for Planetwars AI Challenge in 2010. He has been hooked on since. His broad interests lie in the fields of Statistical Machine Learning and Operating Systems.

3.3 Sacheendra Talluri

Sacheendra is currently a second year student pursuing B.Tech, ICT at Dhirubhai Ambani Institute of Information and Communication Technology (DA-IICT). His area of interest lies in the field of distributed multi agent systems. He is currently working in the Agent Research Lab led by Professor Sourish Dasgupta at DA-IICT.

3.4 Saheb Motiani

Saheb Motiani is currently a senior year student pursuing B.Tech, ICT at Dhirubhai Ambani Institute of Information and Communication Technology (DA-IICT). He has been a Google Summer of Code student in 2012 and 2013. His interests are geared towards Agent Systems and Web APIs. He is currently working in the Agent Research Lab led by Professor Sourish Dasgupta at DA-IICT.