Solving Angry Birds with Reinforcement Learning

Richard Kwasnicki & Julius Gonsior

February 23, 2017

Final presentation for INF-PM-FPA Profilmodul Forschungsprojekt Anwendung

Angry Birds - The Game

Angry Birds - The Game

- Artillery game
- Release date: December 2009 (iPhone)
- With $>$ 2 billion downloads in total most popular mainstream game
- Revenue of Rovio Entertainment in 2015: 142 million euro

Principle

- Primary aim: elimination of all pigs
- Secondary aim: maximize points (3 stars)
- Destroyed/damaged objects (ice, wood, stone, pig) and remaining birds = score

Which types of birds do exist?

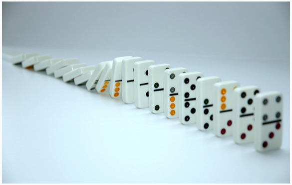
In the first episode “Poached Eggs’ exist:

Picture	Type	Strength	On-Click effect
	Red (Red)	Nothing	Nothing
	Blue (Jim, Jake & Jay)	Ice	Triplication
	Yellow (Chuck)	Wood	Speed-up
	Black (Bomb)	Stone	Explosion
	White (Matilda)	Explosive egg	Drops egg

More types in later episodes …

Why does it matter for AI?

- Predict outcome of physical actions
- No complete knowledge of the world
- Select best action out of $(640 \times 480)^{birds \times taps}$
- $3Birds \rightarrow 1.68 \times 10^{1646}$
- Planning over multiple shots



$\rightarrow$ Competition was born

The competition

- Yearly competition during IJCAI
- Main goal: AI better than Human
- Unknown, newly created levels
- 4 Rounds (Elimination, highest points)
- See more on http://aibirds.org/

Existing Solutions/Approaches

Existing Solutions/Approaches

- DataLab Birds (Data Science Lab Prague, Czech Republic, 1018230 Points)
  - Heuristics for different good actions
- AngryBER (Data Science, Ioannina, Greece, 935330 Points)
  - Machine Learning
  - Calculate expected reward
- Plan A+ (Computer Engineering, Sejong, Korea, 1002380 Points)
  - Two strategy's depending on object breakability
  - Manually tuned parameters (heuristic)
- AngryHEX (974670 Points)
  - ASP Knowledge Base (heuristic)
  - Scene encoding into logic assertions

Our Approach

What do we want to do different?

- Existing solutions (mainly): manual identification of rules or creation of heuristics/scores for good shots
- Our aim: automatic identification of a heuristic which means:
  - Experimental shooting
  - Learn what makes a good combination (planning)
  - Application of learned knowledge in unknown levels

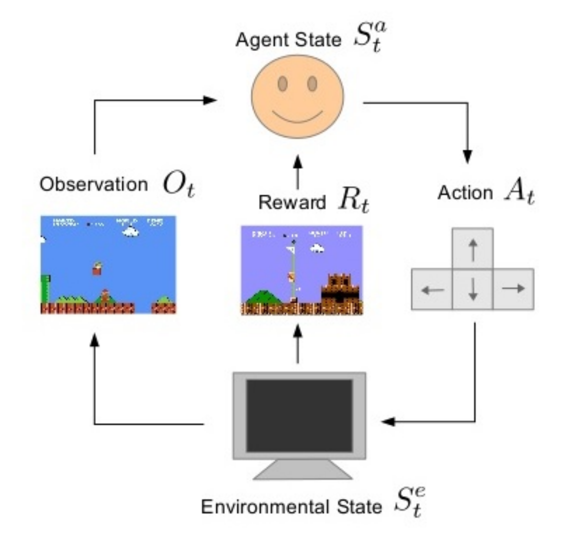
$\rightarrow$ Reinforcement Learning

Reinforcement Learning

Origin

- Field of machine learning
- Behaviorist psychology: animal learning
- Math: optimal control

What is the main principle of Reinforcement Learning?



Q-Learning (1)

Bellmann $\rightarrow$ Value Iteration:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha_t \times \left(r_{t+1} + \gamma \times \max_a Q(s_{t+1}, a) - Q(s_t, a_t) \right)$$

Old Value

Parameter:

- s … state
- a … action

	 12 31	 14 52	 15 72	 15 65
s_0	0	-1	27	3
s_1	5	-1	12	-1
s_2	-1	17	2	-10
s_3	-1	-1	-1	-1
s_4	-1	-1	-1	-1

Q-Learning (1)

Bellmann $\rightarrow$ Value Iteration:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha_t \times \left(r_{t+1} + \gamma \times \max_a Q(s_{t+1}, a) - Q(s_t, a_t) \right)$$

Learning rate

Parameter:

- s … state
- a … action

	 12 31	 14 52	 15 72	 15 65
s_0	0	-1	27	3
s_1	5	-1	12	-1
s_2	-1	17	2	-10
s_3	-1	-1	-1	-1
s_4	-1	-1	-1	-1

Bellmann → Value Iteration:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha_t \times \left(r_{t+1} + \gamma \times \max_a Q(s_{t+1}, a) - Q(s_t, a_t) \right)$$

Reward

Parameter:

- s … state
- a … action

	 12 31	 14 52	 15 72	 15 65
s_0	0	-1	27	3
s_1	5	-1	12	-1
s_2	-1	17	2	-10
s_3	-1	-1	-1	-1
s_4	-1	-1	-1	-1

Q-Learning (1)

Bellmann $\rightarrow$ Value Iteration:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha_t \times \left(r_{t+1} + \gamma \times \max_a Q(s_{t+1}, a) - Q(s_t, a_t) \right)$$

Discount factor

Parameter:

- s … state
- a … action

	12 31	14 52	15 72	15 65
s_0	0	-1	27	3
s_1	5	-1	12	-1
s_2	-1	17	2	-10
s_3	-1	-1	-1	-1
s_4	-1	-1	-1	-1

Bellmann $\rightarrow$ Value Iteration:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha_t \times \left(r_{t+1} + \gamma \times \max_a Q(s_{t+1}, a) - Q(s_t, a_t) \right)$$

Estimate of optimal future value

Parameter:

- s … state
- a … action

	 12 31	 14 52	 15 72	 15 65
s_0	0	-1	27	3
s_1	5	-1	12	-1
s_2	-1	17	2	-10
s_3	-1	-1	-1	-1
s_4	-1	-1	-1	-1

Bellmann $\rightarrow$ Value Iteration:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha_t \times \left(r_{t+1} + \gamma \times \max_a Q(s_{t+1}, a) - Q(s_t, a_t) \right)$$

Parameter:

- s … state
- a … action

	 12 31	 14 52	 15 72	 15 65
s_0	0	-1	27	3
s_1	5	-1	12	-1
s_2	-1	17	2	-10
s_3	-1	-1	-1	-1
s_4	-1	-1	-1	-1

Bellmann → Value Iteration:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha_t \times \left(r_{t+1} + \gamma \times \max_a Q(s_{t+1}, a) - Q(s_t, a_t) \right)$$

Parameter:

- s … state
- a … action

	12 31	14 52	15 72	15 65
s_0	0	-1	27	3
s_1	5	-1	12	-1
s_2	-1	17	2	-10
s_3	-1	-1	-1	-1
s_4	-1	-1	-1	-1

Q-Learning (1)

Bellmann → Value Iteration:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha_t \times \left(r_{t+1} + \gamma \times \max_a Q(s_{t+1}, a) - Q(s_t, a_t) \right)$$

Parameter:

- s … state
- a … action

	 12 31	 14 52	 15 72	 15 65
s_0	0	15	27	3
s_1	5	-1	12	-1
s_2	-1	17	2	-10
s_3	-1	-1	-1	-1
s_4	-1	-1	-1	-1

Q-Learning (2) - Exploration rate

Law of Effect aka Trial & Error with parameter ε (exploration rate)

```
public Action getNextAction() {
        int randomValue = randomGenerator.nextInt(100);
        Action action;
        if (randomValue < explorationRate * 100) {
            action = qValuesDAO.getRandomAction();
        } else {
            action = qValuesDAO.getBestAction();
            }
        return action;
}
```

Q-Learning (3) - Delayed feedback

- Positive reward only on last shot
- First action affected after n games for n birds

Labyrint Example

- 5 x 5 World
- Possible Actions: Up, Right, Down, Left
- Rewards per move:
  - Green tile: +1
  - Red tile: -1
  - Default tile: -0.05

Live

Our Model for Angry Birds

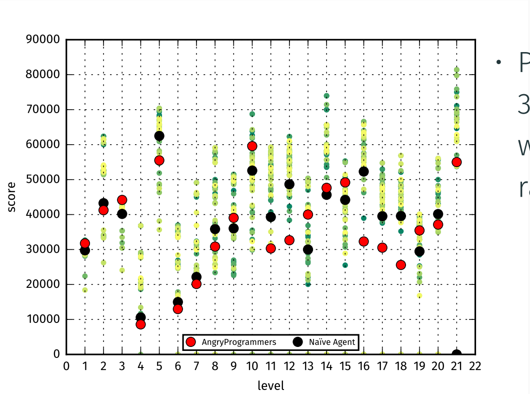
- State: Serialized screenshot with
  - rounded coordinates
  - object-type
  - object-shape
- Action: Shoot on center of an object (Limitation: object direct reachable)
- Reward: Score after successful finishing the level

State: RedBird 19 32 Rect … Wood 59 35 Rect Pig 54 29 Rect

Action: Wood 49 35 Rect Wood 54 31 Rect …

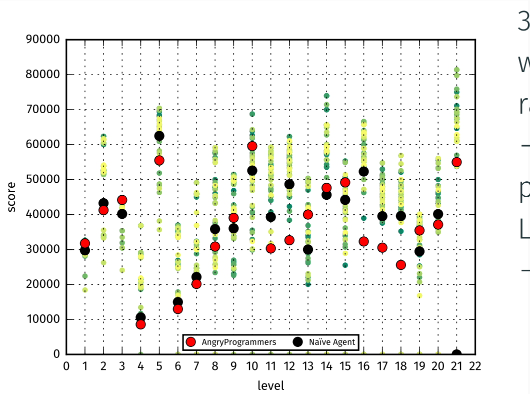
Observation of pure Reinforcement Learning Approach

Observation of pure Reinforcement Learning Approach



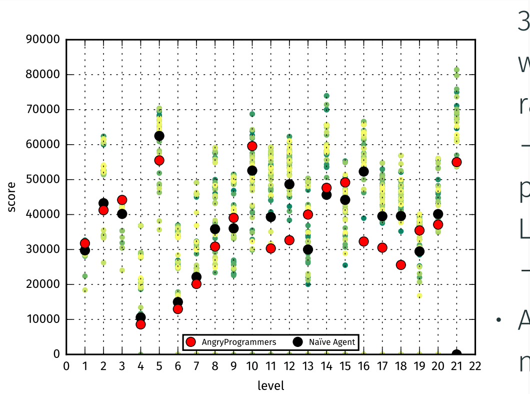
- Played levels after 1 week: 3.217 on a VM from the ZIH with 16 cores and 32GB ram

Observation of pure Reinforcement Learning Approach



- Played levels after 1 week: 3.217 on a VM from the ZIH with 16 cores and 32GB ram
  → much too less for proper Reinforcement Learning
  → delayed feedback!

Observation of pure Reinforcement Learning Approach



- Played levels after 1 week: 3.217 on a VM from the ZIH with 16 cores and 32GB ram
  → much too less for proper Reinforcement Learning
  → delayed feedback!
- Achieved scores similar to naïve agent
  → suggests that nothing was learned

Encountered practical Problems while implementing the Theory

Encountered Problems

Game is implemented as closed source Chrome plugin

- Chrome plugin and screenshot analysis needs lots of resources

  → couldn’t run more than 4 Chrome processes in parallel on a ZIH VM with 16 cores and 32GB ram

Encountered Problems

Game is implemented as closed source Chrome plugin

- Chrome plugin and screenshot analysis needs lots of resources
  - $\rightarrow$ couldn't run more than 4 Chrome processes in parallel on a ZIH VM with 16 cores and 32GB ram
- Game is slow: one shot takes $\sim$ 20 seconds
  - $\rightarrow$ 1.000.000 shots will take $\sim$ 232 days

Encountered Problems

Game is implemented as closed source Chrome plugin

- Chrome plugin and screenshot analysis needs lots of resources
  - $\rightarrow$ couldn't run more than 4 Chrome processes in parallel on a ZIH VM with 16 cores and 32GB ram
- Game is slow: one shot takes $\sim$ 20 seconds
  - $\rightarrow$ 1.000.000 shots will take $\sim$ 232 days
- Not possible to create new levels → overfitting!

Encountered Problems

Given Vision Module is not working exactly

- Object coordinates differ in each run

Encountered Problems

Given Vision Module is not working exactly

- Object coordinates differ in each run
- Wrong object recognition results in:

Encountered Problems

Given Vision Module is not working exactly

- Object coordinates differ in each run
- Wrong object recognition results in:
    - same action can lead to multiple states

Encountered Problems

Given Vision Module is not working exactly

- Object coordinates differ in each run
- Wrong object recognition results in:
    - same action can lead to multiple states
    - same target object didn't result always in same states

Encountered Problems

Given Vision Module is not working exactly

- Object coordinates differ in each run
- Wrong object recognition results in:
  - same action can lead to multiple states
  - same target object didn't result always in same states

→ increases exponentially search space

Encountered Problems

Possible Solutions:

- Reimplement game without GUI and actual API
  → takes a lot of time

Encountered Problems

Possible Solutions:

- Reimplement game without GUI and actual API
  → takes a lot of time
- Try to optimize/speed up existing game implementation
  → will never be as performant as reimplementation

Encountered Problems

Possible Solutions:

- Reimplement game without GUI and actual API
  → takes a lot of time
- Try to optimize/speed up existing game implementation
  → will never be as performant as reimplementation
- Limit search space
  → need to manually discard possible solutions

Encountered Problems

Possible Solutions:

- Reimplement game without GUI and actual API
  → takes a lot of time
- Try to optimize/speed up existing game implementation
  → will never be as performant as reimplementation
- Limit search space
  → need to manually discard possible solutions
- Use much more computating power

Encountered Problems

Possible Solutions:

- Reimplement game without GUI and actual API
  → takes a lot of time
- Try to optimize/speed up existing game implementation
  → will never be as performant as reimplementation
- Limit search space
  → need to manually discard possible solutions
- Use much more computating power

New approach

Hybrid approach: Reinforcement Learning and Heuristics

- Difference to previous solution: drastically reduced search space by limiting possible target actions

Hybrid approach: Reinforcement Learning and Heuristics

- Difference to previous solution: drastically reduced search space by limiting possible target actions
- Instead of proposing all possible target objects, low/high trajectories and tap time a preselection of most promising targets

Hybrid approach: Reinforcement Learning and Heuristics

- Difference to previous solution: drastically reduced search space by limiting possible target actions
- Instead of proposing all possible target objects, low/high trajectories and tap time a preselection of most promising targets
- Reinforcement learning shall again select, based on the current state, which preselected action to take

Hybrid approach: Reinforcement Learning and Heuristics

- Difference to previous solution: drastically reduced search space by limiting possible target actions
- Instead of proposing all possible target objects, low/high trajectories and tap time a preselection of most promising targets
- Reinforcement learning shall again select, based on the current state, which preselected action to take
- Drawback: learned strategy is limited by preselected actions
  → no totally new strategies

Hybrid approach: Reinforcement Learning and Heuristics

Heuristics used for preselection of possible targets:

- Big round objects 

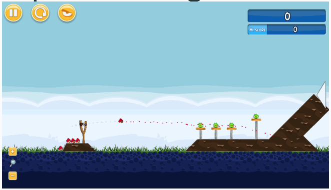
Hybrid approach: Reinforcement Learning and Heuristics

Heuristics used for preselection of possible targets:

- Big round objects 
- TNT 

Hybrid approach: Reinforcement Learning and Heuristics

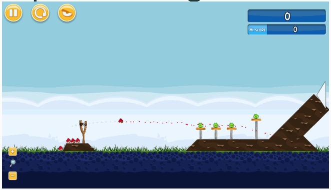
Heuristics used for preselection of possible targets:



- Big round objects 
- TNT 
- Multiple pig shot

Hybrid approach: Reinforcement Learning and Heuristics

Heuristics used for preselection of possible targets:



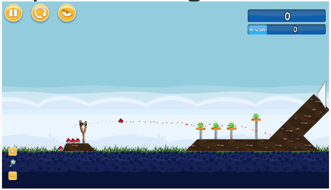
- Big round objects 
- TNT 

- Multiple pig shot

  - Score depending on following factors:
    - Number of objects above

Hybrid approach: Reinforcement Learning and Heuristics

Heuristics used for preselection of possible targets:



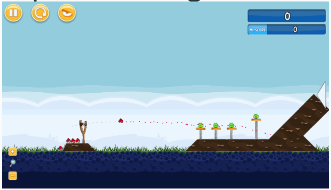
- Big round objects 
- TNT 

- Multiple pig shot

  - Score depending on following factors:
    - Number of objects above
    - Number of objects in trajectory

Hybrid approach: Reinforcement Learning and Heuristics

Heuristics used for preselection of possible targets:

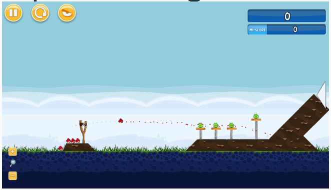


- Big round objects 
- TNT 
- Multiple pig shot

  - Score depending on following factors:
    - Number of objects above
    - Number of objects in trajectory
    - Number of objects to the right

Hybrid approach: Reinforcement Learning and Heuristics

Heuristics used for preselection of possible targets:



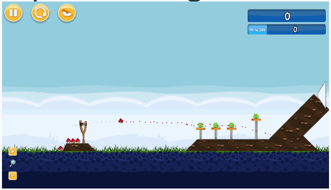
- Big round objects
- TNT
- Multiple pig shot

  - Score depending on following factors:
    - Number of objects above
    - Number of objects in trajectory
    - Number of objects to the right
    - Number of objects below

Hybrid approach: Reinforcement Learning and Heuristics

Heuristics used for preselection of possible targets:

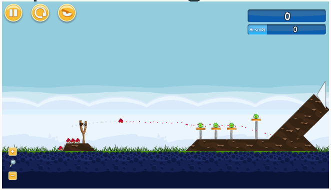
- Big round objects
- TNT
- Multiple pig shot



- Score depending on following factors:
  - Number of objects above
  - Number of objects in trajectory
  - Number of objects to the right
  - Number of objects below
  - How many shots are left

Hybrid approach: Reinforcement Learning and Heuristics

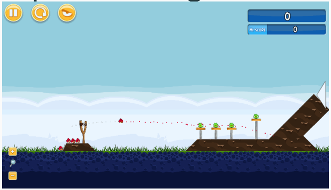
Heuristics used for preselection of possible targets:



- Big round objects
- TNT
- Multiple pig shot
  - Score depending on following factors:
    - Number of objects above
    - Number of objects in trajectory
    - Number of objects to the right
    - Number of objects below
    - How many shots are left
    - Material

Hybrid approach: Reinforcement Learning and Heuristics

Heuristics used for preselection of possible targets:

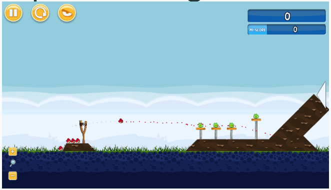


- Big round objects
- TNT
- Multiple pig shot

  - Score depending on following factors:
    - Number of objects above
    - Number of objects in trajectory
    - Number of objects to the right
    - Number of objects below
    - How many shots are left
    - Material
    - Orientation

Hybrid approach: Reinforcement Learning and Heuristics

Heuristics used for preselection of possible targets:



- Big round objects
- TNT
- Multiple pig shot

  - Score depending on following factors:
    - Number of objects above
    - Number of objects in trajectory
    - Number of objects to the right
    - Number of objects below
    - How many shots are left
    - Material
    - Orientation
    - Distance to pigs

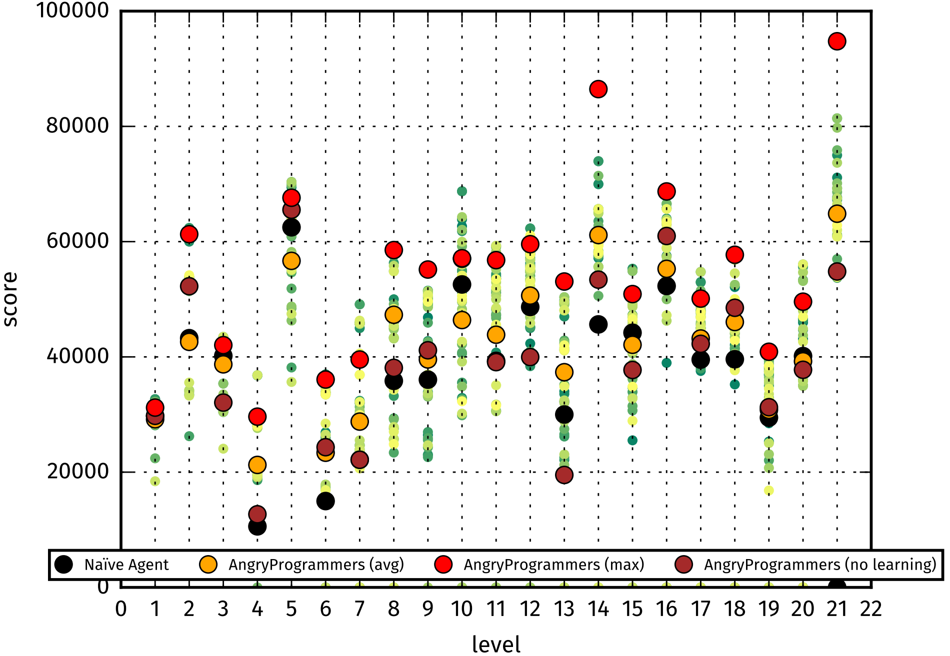
Observation again

Observation of Reinforcement Learning and Heuristic

- Played levels after 2 weeks: 5172

Observation of Reinforcement Learning and Heuristic

- Played levels after 2 weeks: 5172
- 1–2 tries needed to solve first level, later level needed comparably much more tries


score
level
Naïve Agent
AngryProgrammers (avg)
AngryProgrammers (max)
AngryProgrammers (no learning)
0
20000
40000
60000
80000
100000
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22

Ready for competition yet?

- Max values suggest that reinforcement learning is a suitable promising method for solving of Angry Birds

Ready for competition yet?

- Max values suggest that reinforcement learning is a suitable promising method for solving of Angry Birds
- But: good results aren't reproducible yet because the really good shots haven't been remembered
  → delayed feedback in Q-learning

Ready for competition yet?

- Max values suggest that reinforcement learning is a suitable promising method for solving of Angry Birds
- But: good results aren't reproducible yet because the really good shots haven't been remembered
  → delayed feedback in Q-learning
- for competition the overfitting still needs to be checked

Ready for competition yet?

- Max values suggest that reinforcement learning is a suitable promising method for solving of Angry Birds
- But: good results aren't reproducible yet because the really good shots haven't been remembered
  → delayed feedback in Q-learning
- for competition the overfitting still needs to be checked
- not yet implemented strategy for solving unknown level in competition modus

Outlook

What would we do different if we were starting all over?

- Reimplement the game in a more machine learning friendly way

What would we do different if we were starting all over?

- Reimplement the game in a more machine learning friendly way
    - API to get current objects

What would we do different if we were starting all over?

- Reimplement the game in a more machine learning friendly way
    - API to get current objects
    - Direct output of computed resulted state after shooting an object (no useless wait for pretty animation to finish)

What would we do different if we were starting all over?

- Reimplement the game in a more machine learning friendly way
  - API to get current objects
  - Direct output of computed resulted state after shooting an object (no useless wait for pretty animation to finish)
- Can't discourage to not use Reinforcement Learning, result of second approach looked promising
  → leads to speculation that first approach probably would have worked with more iterations

Questions?

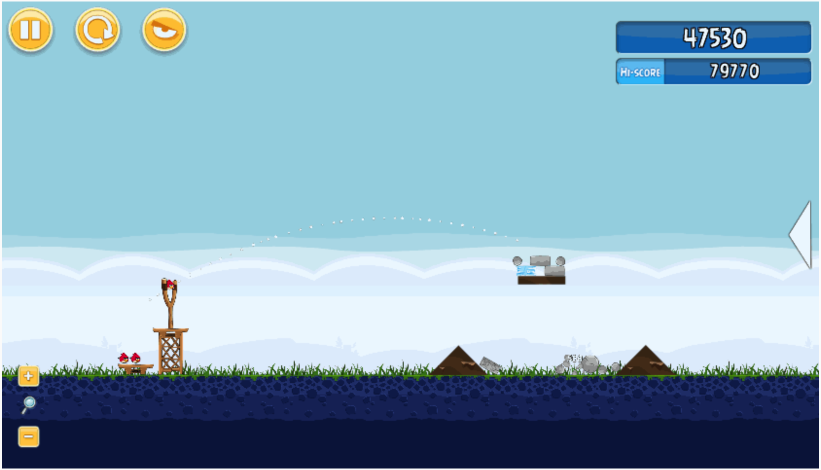
Level 14

Level 14

Level 14

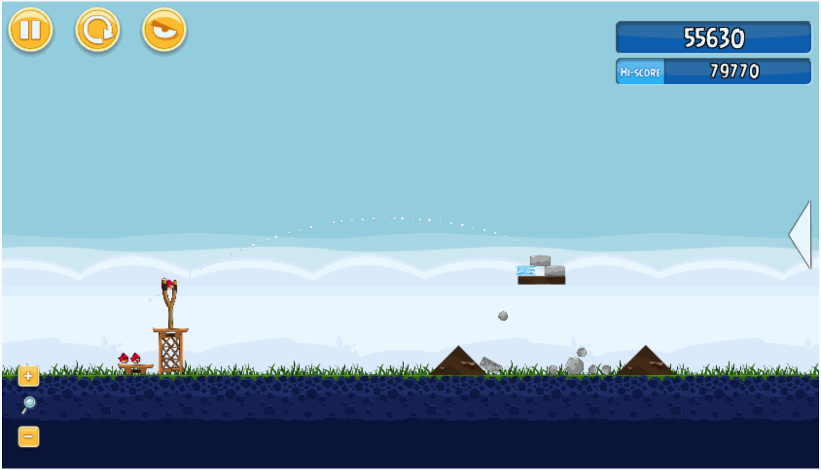
Level 14

Level 14



Level 14

Level 14

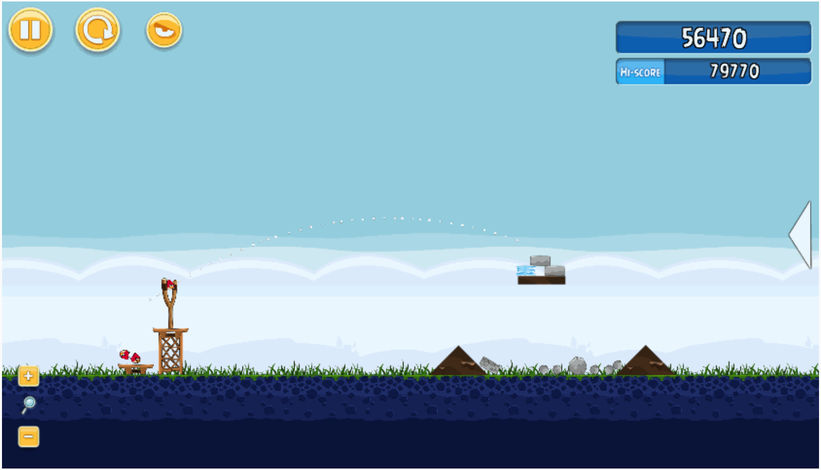


Level 14

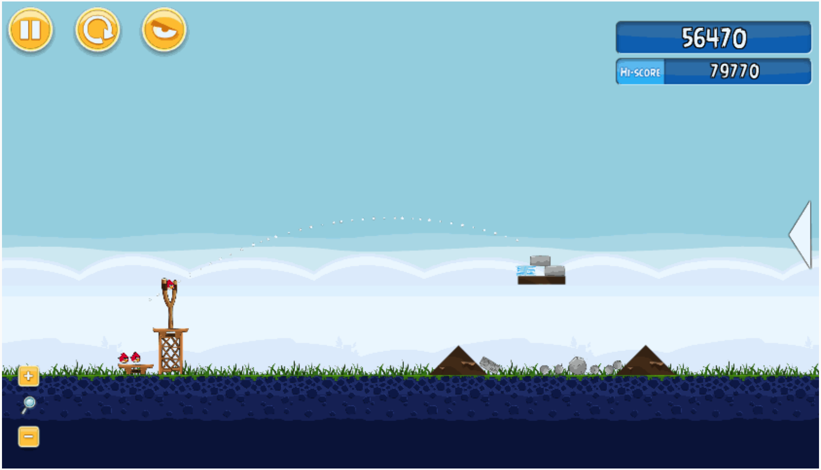
Level 14

Level 14

Level 14



Level 14



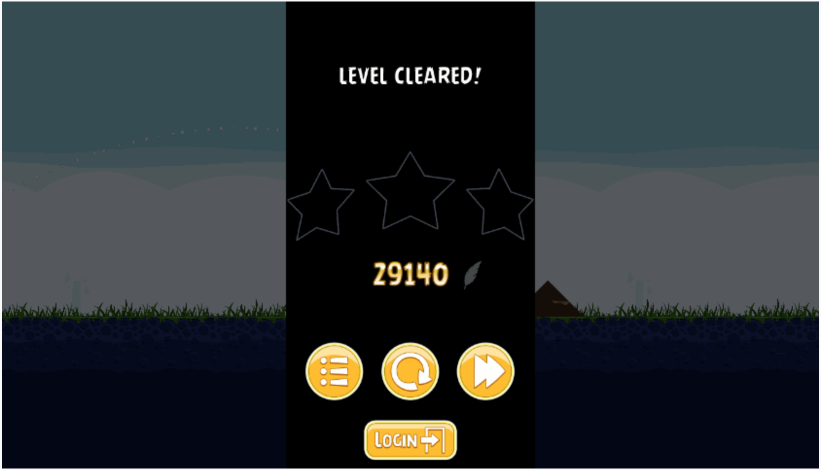
Level 14

Level 14

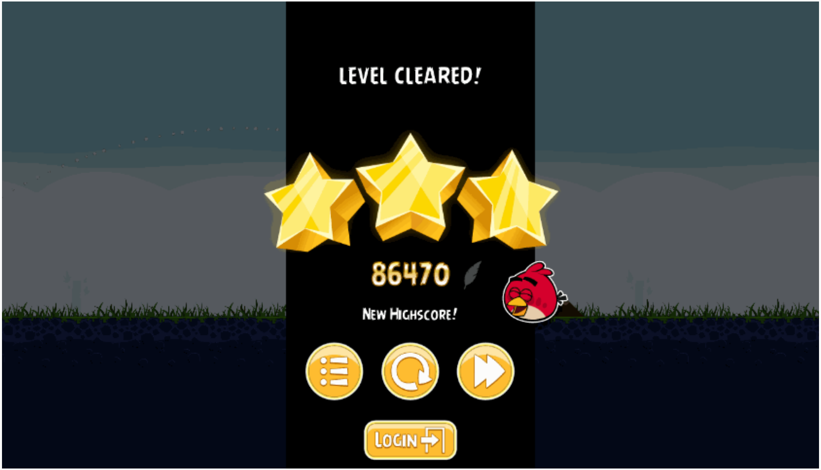
Level 14

Level 14

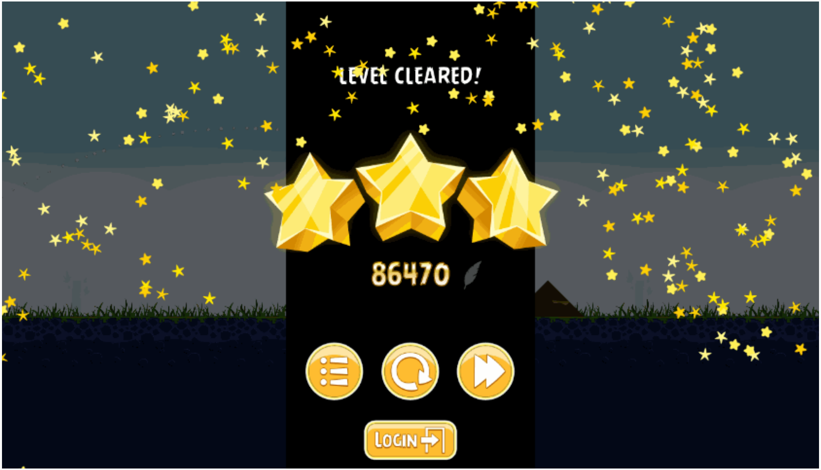
Level 14



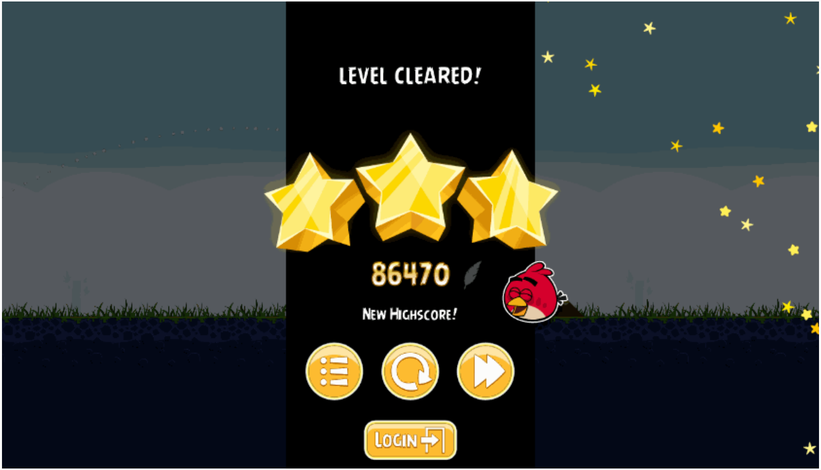
Level 14



Level 14



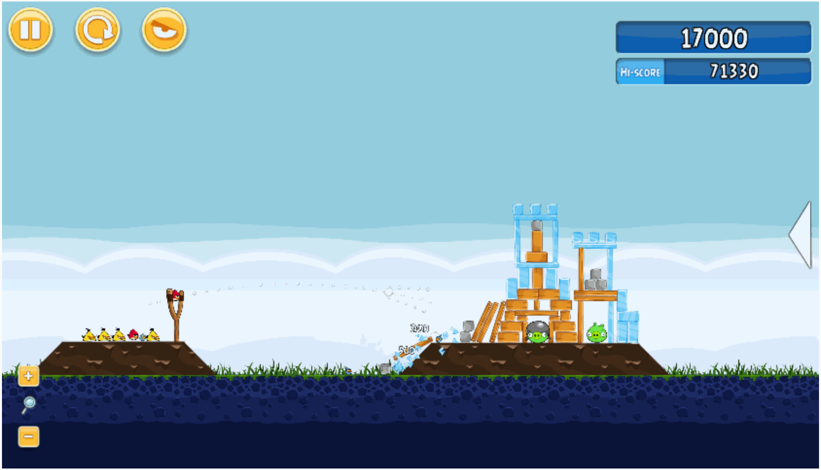
Level 14



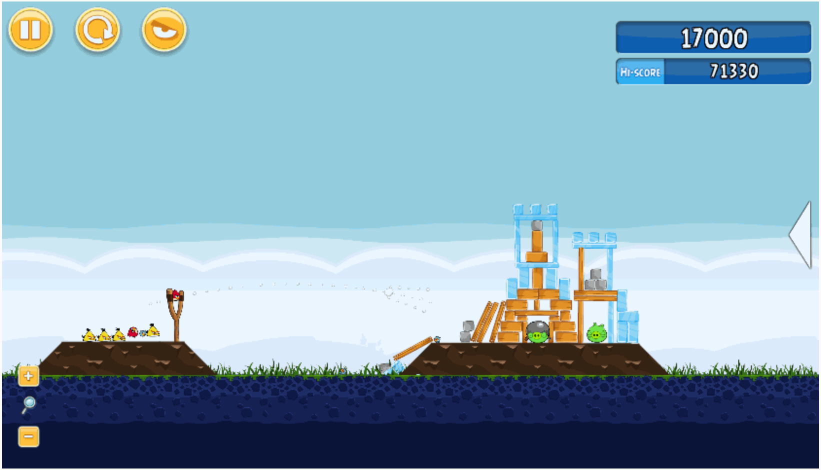
Level 21

Level 21

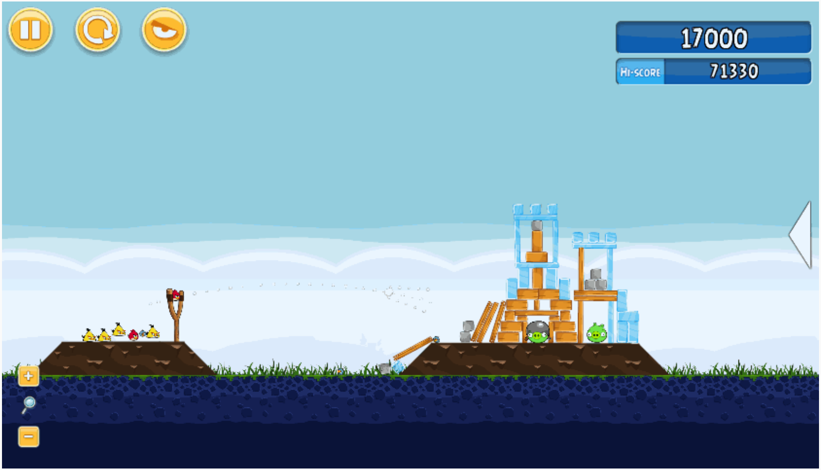
Level 21



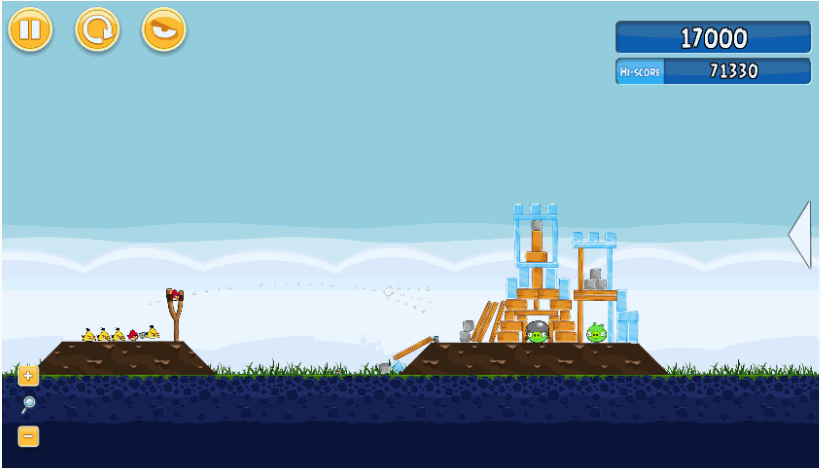
Level 21



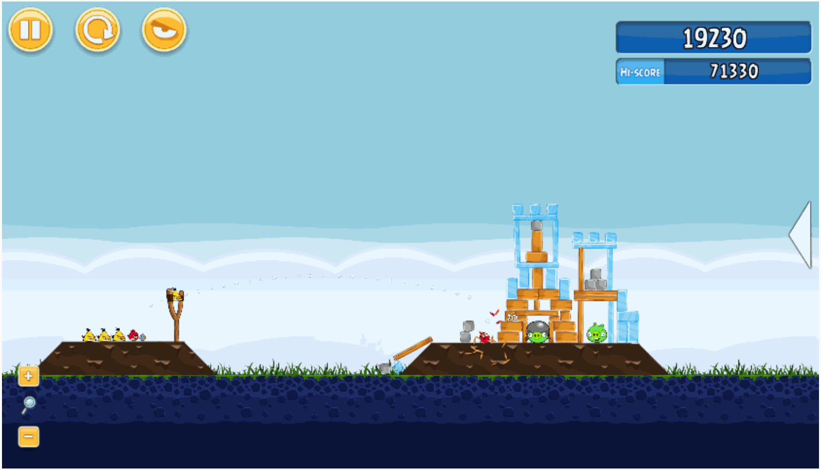
Level 21



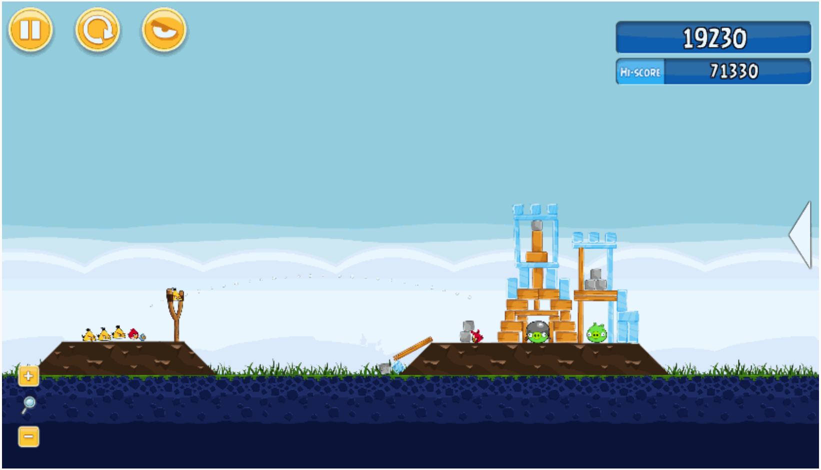
Level 21



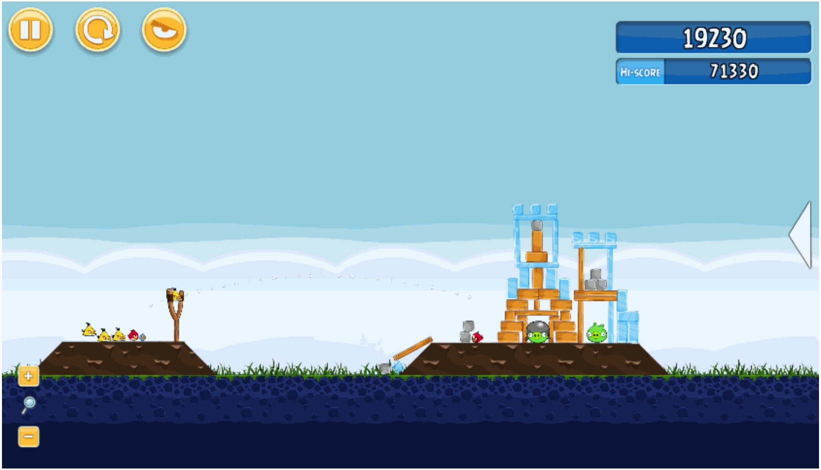
Level 21



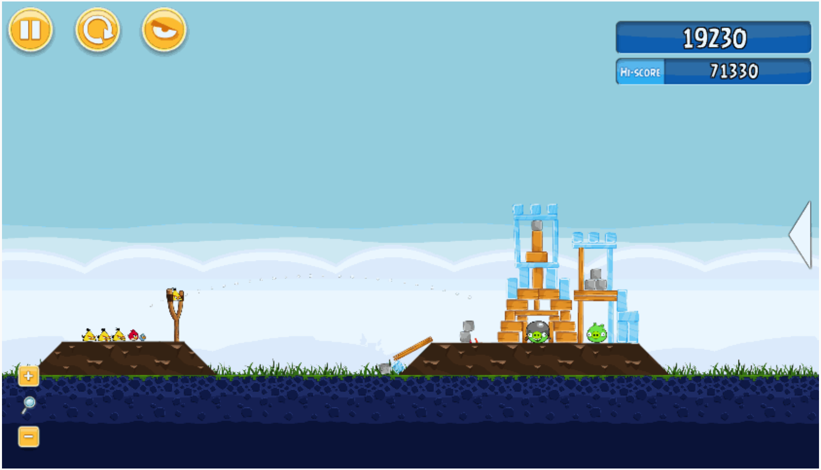
Level 21



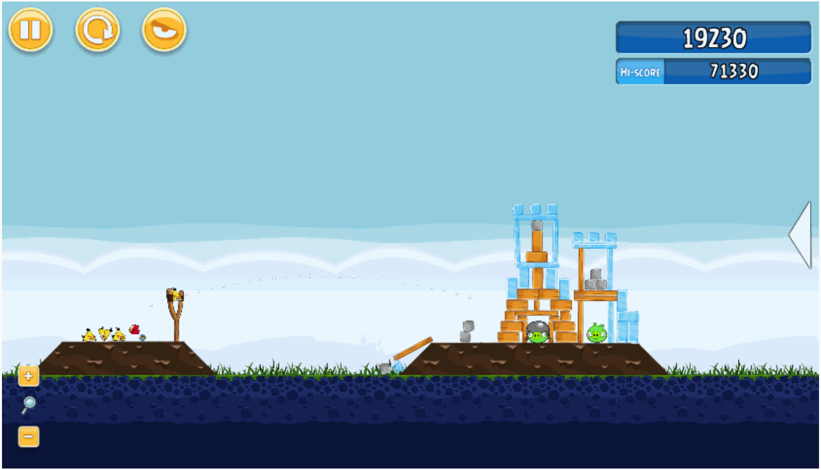
Level 21



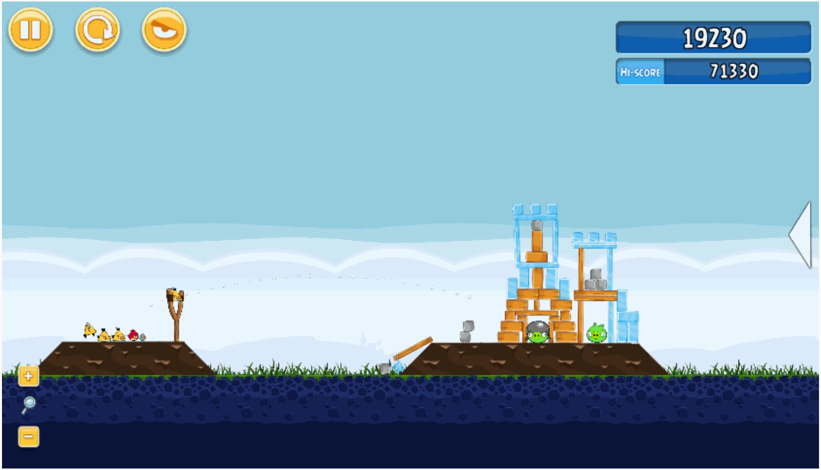
Level 21



Level 21



Level 21



Level 21

Level 21

Level 21

Level 21

Level 21

Level 21

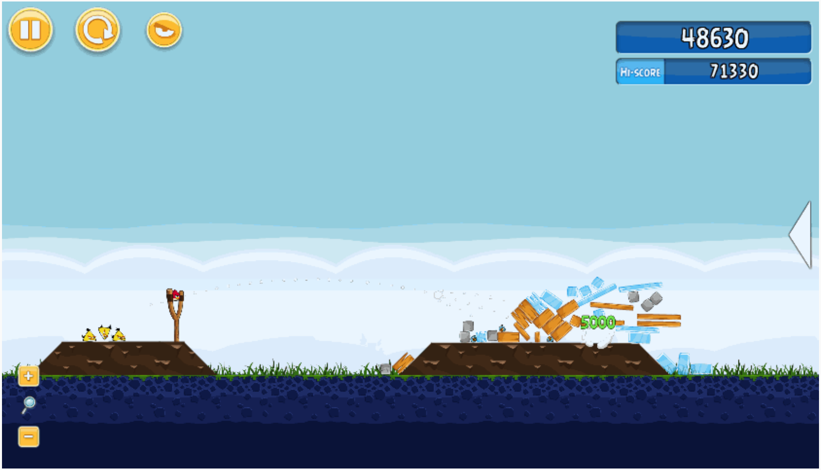
Level 21

Level 21

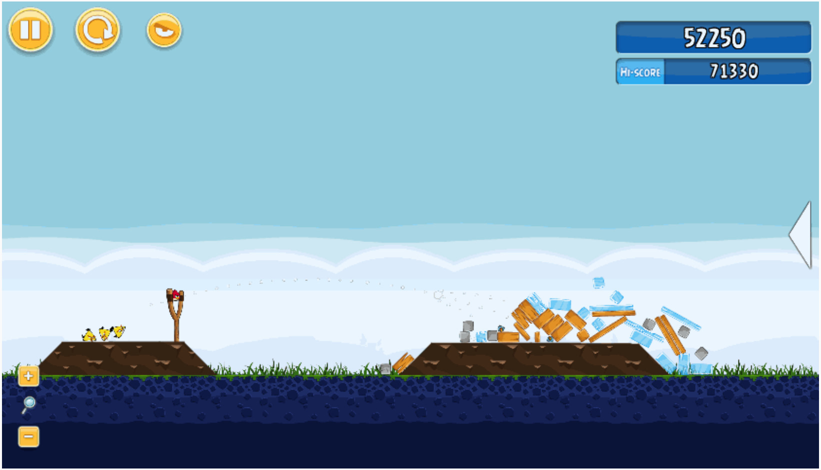
Level 21

Level 21

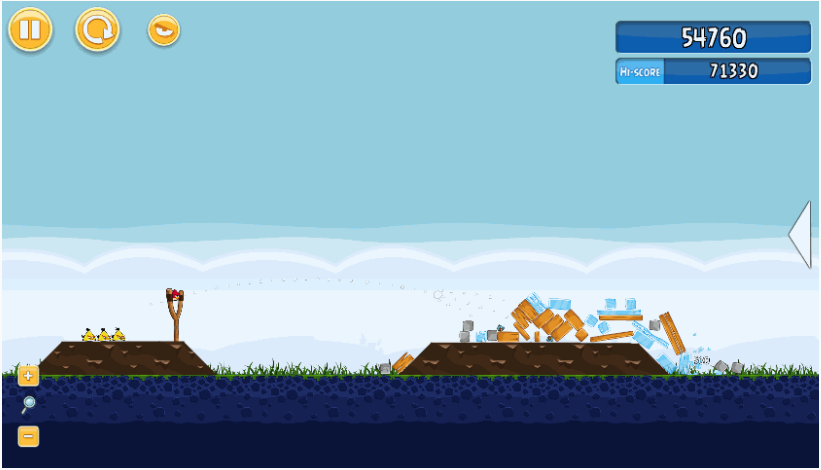
Level 21



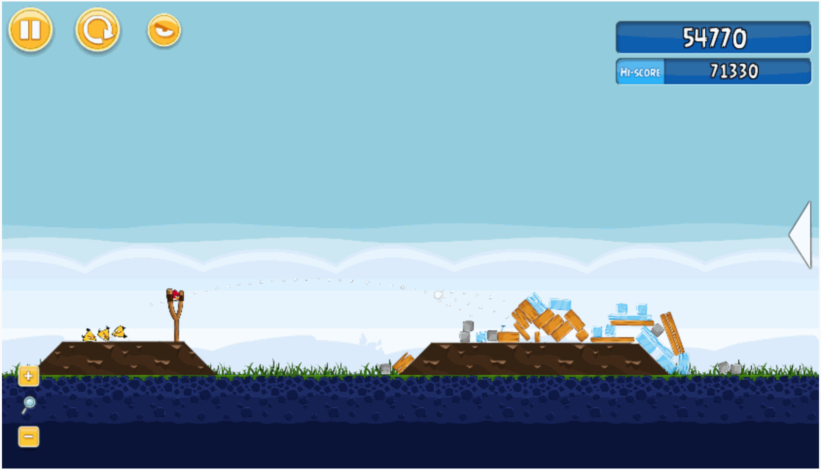
Level 21



Level 21



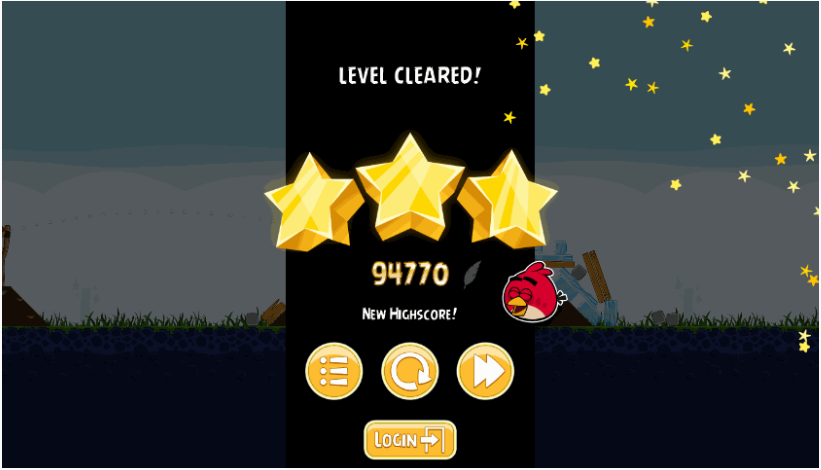
Level 21



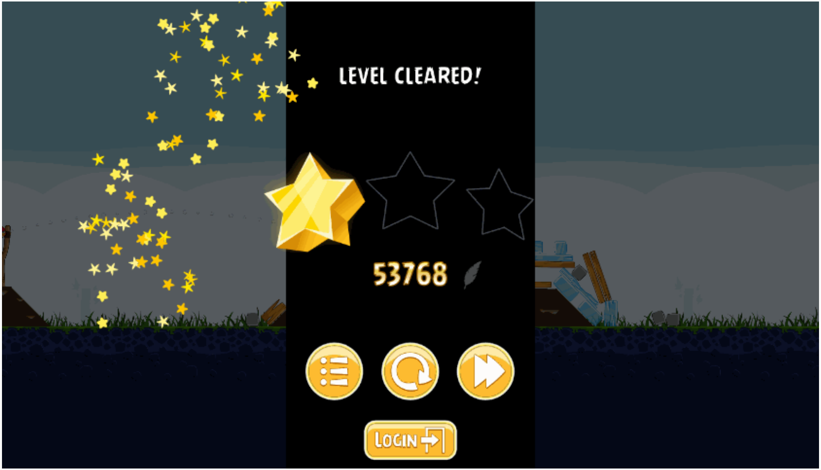
Level 21

Level 21

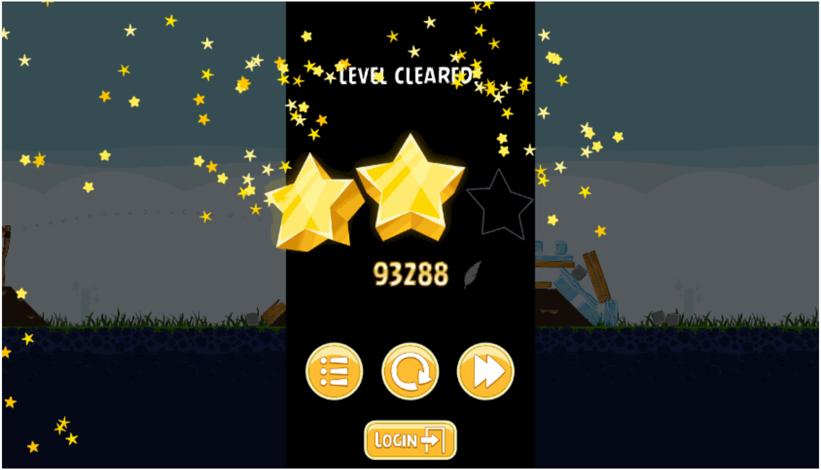
Level 21



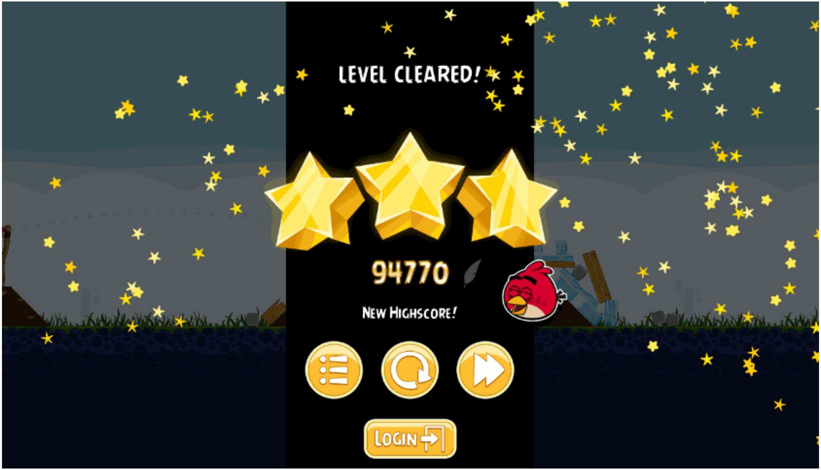
Level 21



Level 21



Level 21



Level 21