
B2SIM DESIGN OVERVIEW

OVERVIEW OF BLOONS TD BATTLES 2

Multiplayer versus game where players must send “bloons” to overwhelm their opponent while simultaneously popping bloons sent their way.

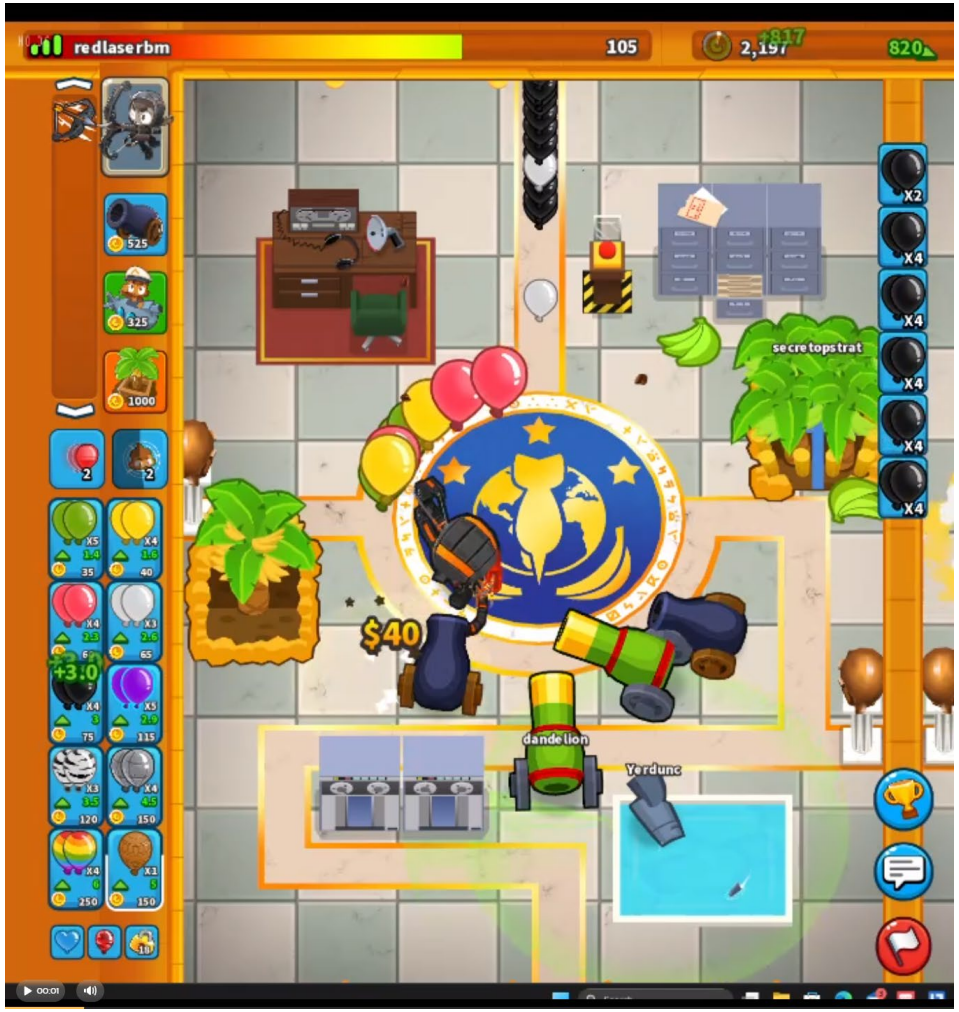
A game usually consists of up to 40 rounds. Each player has a limited selection of bloons they can choose to send, and this selection becomes increasingly more powerful as the game proceeds.

Optimal gameplay usually comprises building income early by sending weaker bloons to sustain sending a lot of more costly, more lethal bloons later in the game.

GENERATING INCOME

Two primary sources of income:

- **Eco** – Cash gained every 6 seconds. The player can send bloons which cost money now but increase this value, allowing for more money later. Usually a cost-effective source of income generation.
- **Farms** – A source of income that pays out money evenly over the course of a round and can be upgraded as the game progresses to generate more money. The effectiveness of farms vary depending on round lengths.
- Top players normally focus on developing *both* income sources simultaneously early on before stopping eco at around R20 to develop powerful farms.



PROBLEM OVERVIEW

Problem – It is difficult to rigorously determine how to best balance developing the two sources of income since their effectiveness can vary from game to game. While acquiring raw game experience develops strong intuition, it is difficult in the heat of the moment of a given b2 match to objectively obsess the optimality of a given farm strategy.

Stakeholders – Top-level players of Bloons TD Battles 2, who desire a way to quickly and objectively analyze the effectiveness of farming strategies without needing to repeatedly play the game.

Task – Given the state of the game in the present time and a *flowchart* consisting of a prescribed strategy for using eco sends and purchasing/selling/upgrading farms, we want to determine instantaneously the state of the game at a given time in the future.

SIMULATING ECO

In more abstract terms, let $G_s = (C_s, E_s, F_s)$ represent the state of the game at time s, and let ζ_E, ζ_F represent flowcharts for eco'ing and farming respectively. Then, for $t > s$ and some Φ_t we have that $G_t = \Phi_t(s, G_s, \zeta_E, \zeta_F)$, and our goal is to write code for Φ_t so that G_t can be determined instantaneously.

Our strategy to build the general Φ_t is to first do it in simple cases, and from there expand into increasingly more complex scenarios.

Our first goal is to build Φ_t in the special case where there are no farms and the only class of ζ_E's considered is the class C consisting of constantly attempting to use *one* eco send repeatedly.

SIMULATING ECO

For fast simulation, we undertake the following procedure:

- Determine if the player can send an eco bloon at the *current time*
  - If they can, add the eco bloon to the queue, and fast forward to the *next time* they can send a bloon.
  - If they can't, fast forward to the first time that they *can*.
- If the next time to send eco is beyond the target time, record this time, but then fast forward only to the target time.
- The table on the right shows which time the simulator fast-forwards to based on what happens in the present.

Event	Time To Fast Forward To
Player is able to send & there's still space left in the queue afterwards	After the eco delay expires $(t + 0.15)$
Player is able to send & and this causes the queue to completely fill	After the first bloon in the queue leaves
Player can't send - Not enough cash	After the next eco tick $6\left(\left\lfloor \frac{t}{6} \right\rfloor + 1\right)$
Player can't send – Full eco queue	After the first bloon in the queue leaves

SIMULATING ECO

Eco payments occur every 6 seconds, and we want to avoid accidentally skipping over them. Let Φ_t^E be a function that performs the eco simulation process of the previous slide until target time t. To simulate over $(s, t]$, we do the following:

- While $s < t$…
  - $s_1 = \min\left(t, 6\left(\left\lfloor\frac{s}{6}\right\rfloor + 1\right)\right)$
  - Set $G_{s_1} = \Phi_{s_1}^E(s, G_s, \zeta_E, 0)$
  - If $s_1|6 = 0$, then set $G_{s_1}^C \mathrel{+}= G_{s_1}^E$
  - Set $s = s_1$
- Essentially, we stop the simulation at times divisible by 6 to award payments from eco.

Each bloon send has an associated *send duration* which determines how long it takes that send to leave the queue once it becomes first in line. In general, let s_i denote the send duration of the ith set in line and w the amount of time that the first send has spent being first in line. Then, the jth set in line will leave the queue at the time

$$t - w + \sum_{i=1}^{j} s_i$$

FARM PAYOUTS

In every round of a typical Battles 2 game, a special class of bloons called *natural bloons* will be sent by the game to both players. The bloons that get sent and the duration of time s_i spent sending those bloons is predetermined, but the actual length $r_i > s_i$ of a round depends on when the natural bloons are popped.

Most farms are determined completely by a tuple (P, F) which describes how many times F the farm will try to pay out over a round and the payout P the farm gives when it does pay out.

FARM PAYOUTS

Let $R_i = \sum_{k=0}^{i-1} r_k$ denote when round i starts. In general, farms have two rules concerning payouts over a given round i:

- If a farm is bought *before* the start of a round, it will payout P dollars at the times
  $\mathrm{R_i} + \mathrm{s_i}\frac{\mathrm{k-1}}{\mathrm{F}}$ for $1 \leq k \leq F$
- If a farm is bought *during* a round at some time t, it will payout P dollars at the times $\mathrm{t} + \mathrm{s_i}\frac{\mathrm{k}}{\mathrm{F}}$ for $1 \leq k < F\frac{R_i+s_i-t}{s_i}$

SIMULATING ECO & FARMS

Our next goal is to extend Φ_t to the case where the player may have farms but cannot make any changes to them.

In general, if we expect a payout of any kind to occur at a time t_i, we should simulate to t_i, award the payment, and then continue simulating until the time of next payout.

Thus, the simulation can be described as follows:

- Let $\{(t_i, P_i)\}_{i=1}^{N}$ be a sequence arranged in chronological order describing all payouts to occur within the time interval $(s, t]$. We assume $t_N = t$ and set $P_N = 0$ if no payout is expected at that time.
- Set $i = 0$. Then, while $\mathrm{i} < N$…
  - $i \mathrel{+}= 1$
  - Set $G_{t_i} = \Phi_{t_i}^{E}\left(t_{i-1}, G_{t_{i-1}}, \zeta_E, 0\right)$
  - Set $G_{s_1}^{C} \mathrel{+}= P_i$

SIMULATING ECO & FARMS

A positive consequence of our approach to simulation is that we can skip over times where nothing happens without compromising simulation accuracy.

$\zeta_E = constant\ grouped\ yellows, \zeta_F = 0$