ELO Learning

THE ELO ALGORITHM

ZERO DAY

Last updated: 19 August 2025
University of Pretoria

Name	Student number
RM (Rene) Brancon	u22556771
NF (Nigel) Mofati	u22528084
TM (Tukelo) Mokwena	u22536800
S (Saskia) Steyn	u17267162
NG (Ntokozo) Tonga	u22506773

Team contact:
ZeroDay0D4y@gmail.com

Contents

1. Single Player Mode

XP Implementation:

The single player XP algorithm is designed to calculate xp for a user after completing a question or a set of questions.

It rewards:

- Correctness (whether the answer was right)
- Speed (bonus for faster completion)
- Level scaling (higher levels yield proportionally less XP)
- Gatekeeping (more XP if the user is further from their next level)

The final XP is designed to be positive, fair, and scalable with user progress.

Core Functions
calculateTimeReward(actualTimeSeconds)

Provides a bonus proportionate to how quickly the player answered compared to maxTimeSeconds.

Formula:

$$Time\ Reward\ =\ max(0,\ maxTime\ -\ actualTime)/maxTime$$

Range:

0 (slowest) to 1 (fastest)

calculateLevelReward(currentLevel)

Adjusts XP to make leveling harder at higher levels.

Formula:

$$Level\ Reward\ =\ 1/(1\ +\ \alpha\ *\ currentLevel)$$

α -> turnable constant

calculateGateKeepingComponent(currentXP, nextLevelXP)

Provides bonus XP when the player is far from the next level.

Formula:

$$Gatekeeper = \beta * ((nextLevelXP - currentXP)/nextLevelXP)$$

β -> Gate keeping scaling constant

calculateSinglePlayerXP()

The main function that combines all the components.

Formula:

$$XP = CA * ((XPGain * CA) + (XPGain * TimeReward) + (XPGain * levelReward) + (XPGain * Gatekeeper)) * 0.3$$

Where:

- CA = Correctness (0 or 1)
- XPGain = base XP for the question
- scalingFactor =

ELO Rating Implementation:

In single-player mode, questions act as “opponents” with their own ELO ratings. When a player answers a question, both the player’s and question’s ELO rating are updated based on the outcome.

How it works:

1. Question Difficulty: Each question has an ELO rating representing its difficulty.
2. Expected Outcome: System calculates probability of player getting question correct.
3. Actual Outcome: Player either gets it right or wrong.
4. Rating Update: Both player and question ratings adjust accordingly.

Core Functions:

- calculateExpectedRating(ratingA, ratingB, alpha = 400)
    - Calculates the expected win probability for ratings A vs ratingsB

    - Formula:

      $$Expected = 1 / (1 + 10^{((ratingsA - ratingsB)/alpha)})$$

    - Returns: Probability between 0 and 1

- updateEloRating({ rating, expected, actual, kFactor = 40})
    - Updates a single ELO rating based on expected vs actual outcome
    - Formula:

      $$delta = kFactor * (actual - expected)$$

      $$newRating = max(0, round(rating + change * scalingFactor))$$

    - Returns: Updated rating (minimum 0)

- updateSinglePlayerEloPair({ playerRating, questionRating, isCorrect, })
    - Updates both player and question ELO ratings simultaneously
    - Returns:

```
{
  "newPlayerElo": number,
  "newQuestionElo": number,
  "playerEloChange": number,
  "questionEloChange": number,
}
```

Parameters:

- playerRating : Current player ELO
- questionRating: Current question difficulty ELO
- isCorrect: Boolean outcome
- alpha = 400: ELO scaling constant
- playerK = 40: Player rating change sensitivity
- questoinK = 24: Question rating change sensitivity
- totalNumberOfQuestions = 2: Scaling divisor

Formula:

$$expectedP = 1 / (1 + 10^{((questionRating - playerRating)/400)})$$

$$deltaP = playerK * (score - expectedP) / totalNumberOfQuestions$$

$$deltaQ = questionK * ((1 - score) - expectedQ)/totalNumberOfQuestions$$

$$newPlayerElo = clamp(playerRating + deltaP, minRating, maxRating)$$

$$newQuestionElo = clamp(questionRating + deltaQ, minRating, maxRating)$$

Benefits:

- Questions become more accurately rated over time.
- Players face appropriately challenging content.
- Automatic difficulty balancing.
- Separate K-factors allow different adaptation rates.

2. Multiplayer Mode

Overview

This ELO XP algorithm combines:

- ELO rating expectations (how strong players are relative to each other)
- Actual performance (how well players did in a specific match)

The goal is to distribute XP fairly:

- Players gain more XP when they beat stronger opponents.
- Players gain less XP when they underperform against weaker opponents.
- Minimum XP protection ensures fair gameplay

Core Functions:

- **calculateExpected(p1,p2)**
  - Uses the same `calculateExpectedRating` function as single-player mode.Calculates the expected win probabilities based on player ratings.
  - **Formula:**

$$p1_{expected} = 1/1 + 10^{(p2-p1)/\alpha}$$

$$p2_{expected} = 1 - p1_{expected}$$

    - Where p1 is player 1’s rating, p2 is player 2’s rating, α = 400 is the ELO scaling constant

- **calculateMultiplayerXP(xpTotal, expected1, expected2, score1)**
    - Distributes XP between two players based on ELO expectations and match outcome.
    - **Parameters:**
        - xpTotal: Total XP pool to distribute
        - expected1, expected2: Expected win probabilities for each player
        - Score1: Match outcome (1= player1 wins, 0.5 = draw, 0 = player2 wins)
    - **Formula:**

$$xp1 = scaledTotal * (score1 - expected1)$$

$$xp2 = scaledTotal * (score2 - expected2)$$

$$if\ xp1 <= 0 : xp1 = scaledTotal * minXPFraction \,\&\, xp2 = scaledTotal - xp1$$
$$if\ xp2 <= 0 : xp2 = scaledTotal * minXPFraction \,\&\, xp1 = scaledTotal - xp2$$

    - **Returns:**
        - **[Math.round(xp1), Math.round(xp2)]**

Benefits:

- Players gain more XP when beating stronger opponents
- Players lose less XP when losing to stronger opponents
- Encourages playing against higher-rated opponents
- Guaranteed minimum XP prevents frustration

Tuning Constants

Single-Player XP:

- α (alpha) = 0.1 - Level scaling sensitivity
- β (beta) = 0.05 - Gatekeeping bonus scaling
- maxTimeSeconds = 30 - Maximum time for questions
- Final Scaling = 0.3 - XP reduction factor

Single-Player ELO:

- α (alpha) = 400 - ELO scaling constant
- playerK = 40 - Player rating change factor
- questionK = 24 - Question rating change factor

- scalingFactor = 1.0 - Overall scaling multiplier
- totalNumberOfQuestions = 2 - Scaling divisor
- minRating = 0, maxRating = Infinity

Multiplayer:

- α (alpha) = 400 - ELO scaling constant
- K-factor = 40 - Rating change sensitivity
- multiplayerScalingFactor = 0.4 - Base XP scaling
- minXPFraction = 0.15 - Minimum XP protection (15% of total)