Team-Based Glicko-2 Rating with Per-Player Performance Weighting

1 Overview

This document describes a rating system for a competitive multiplayer game (4v4 or 5v5) based on the Glicko-2 algorithm, extended with:

- team-based matchmaking (two teams per match),
- per-player performance weighting (stronger contribution for players who carry, weaker for those who underperform),
- recent performance tracking via exponential moving average for short-term form detection,
- effective rating blending long-term Glicko-2 with recent performance,
- inactivity decay for rating deviation,
- team balancing algorithm with risk-adjusted scoring and configurable constraints.

Each player maintains a Glicko-2 rating state and is updated after every match, with the magnitude of the rating change modulated by their relative performance inside their own team. Recent performance is tracked separately to capture short-term form changes for improved matchmaking.

2 Player State

For each player i we maintain the following Glicko-2 parameters:

- rating R_i (e.g. initial value $R_0 = 1400$),
- rating deviation RD_i (e.g. initial value $\mathrm{RD}_0 = 350$),
- volatility σ_i (e.g. initial value $\sigma_0 = 0.06$),
- recent performance index $\mathrm{perfEMA}_i$ (exponential moving average of z-scores, initial value 0),
- performance games count $\mathrm{perfGames}_i$ (number of matches contributing to EMA, initial value 0).

Internally, Glicko-2 uses a transformed scale:

$$\mu_i = \frac{R_i - R_0}{173.7178}, \tag{1}$$

$$\phi_i = \frac{\mathrm{RD}_i}{173.7178}, \tag{2}$$

where $R_0 = 1400$ is the default initial rating.

These internal parameters $(\mu_i, \phi_i, \sigma_i)$ are used for updates. After each update we convert back to (R_i, RD_i) via:

$$R_i = 173.7178 \cdot \mu_i + R_0, \tag{3}$$

$$\mathrm{RD}_i = 173.7178 \cdot \phi_i. \tag{4}$$

3 Match Structure

Each match consists of two teams:

- Team A with player set $A = \{a_1, \ldots, a_n\}$,
- Team B with player set $B = \{b_1, \ldots, b_m\}$,

where $n, m \in \{4, 5\}$ for 4v4 or 5v5. The match outcome is:

- $S_A = 1, S_B = 0$ if Team A wins,
- $S_A = 0, S_B = 1$ if Team B wins,
- $S_A = S_B = 0.5$ for a draw.

4 Per-Player Performance Score

To incorporate the individual *performance score* p_i of each player i the following metric is defined:

$$p_i = \sum_{1}^{n} w_i \cdot stat_i \tag{5}$$

For a mode in which deaths, kills, damage and a general objective are tracked, one could define:

- $w_{\text{kill}} = 1$
- $w_{\text{death}} = -1$
- $w_{\text{dmg}} = 1/(hp\ equivalent)$

The system only needs *relative* performance within each team, so any monotonic transformation of p_i is acceptable.

5 Performance Scaling Within a Team

For each team T (either $T = A$ or $T = B$), with player set $T = \{i_1, \ldots, i_{|T|}\}$, we compute performance z-scores that will be used to scale rating changes.

5.1 Team Performance Statistics

$$\bar{p}_T = \frac{1}{|T|} \sum_{i \in T} p_i, \tag{6}$$

$$s_T = \sqrt{\frac{1}{|T|} \sum_{i \in T} (p_i - \bar{p}_T)^2} \tag{7}$$

Define the performance z-score for each player $i \in T$:

$$z_i = \frac{p_i - \bar{p}_T}{s_T}. \tag{8}$$

5.2 Sign-Aware Scaling Factor

The scaling factor depends on both the player's performance relative to teammates and the sign of the rating change from Glicko-2. Given $\Delta\mu_i = \mu_i^* - \mu_i$ (the rating change before performance adjustment), we compute:

$$f_i = 1 + \beta \cdot \text{sign}(\Delta\mu_i) \cdot z_i, \tag{9}$$

where:

- $\beta > 0$ controls sensitivity (typical value: $\beta = 0.2$),
- $\text{sign}(\Delta\mu_i) = +1$ if $\Delta\mu_i \geq 0$ (team won), -1 otherwise (team lost).

Clamp f_i to stay within a reasonable range:

$$f_i = \min\{f_{\max}, \max\{f_{\min}, f_i\}\}, \tag{10}$$

where typical values are $f_{\min} = 0.5$ and $f_{\max} = 1.5$.

5.3 Behavior

This formula ensures:

- **Wins** ($\Delta\mu_i > 0$):
  - Good performers ($z_i > 0$): $f_i > 1 \rightarrow$ gain more rating.
  - Poor performers ($z_i < 0$): $f_i < 1 \rightarrow$ gain less rating.
- **Losses** ($\Delta\mu_i < 0$):
  - Good performers ($z_i > 0$): $f_i < 1 \rightarrow$ lose less rating.
  - Poor performers ($z_i < 0$): $f_i > 1 \rightarrow$ lose more rating.

Note that $E[f_i] = 1$ across the team (since $E[z_i] = 0$), so the system redistributes rating changes based on performance without inflating or deflating the total.

6 Team Aggregated Rating for Glicko-2

Although Glicko-2 is defined for individual vs individual matches, we can approximate team games by treating each player's opponent as the *aggregated rating* of the opposing team.

For a team T with players $i \in T$, define its internal mean and deviation:

$$\mu_T = \frac{1}{|T|} \sum_{i \in T} \mu_i, \tag{11}$$

$$\phi_T = \sqrt{\frac{1}{|T|^2} \sum_{i \in T} \phi_i^2}. \tag{12}$$

In a match between Team A and Team B, each player in Team A will treat a single opponent with parameters (μ_B, ϕ_B), and analogously each player in Team B sees (μ_A, ϕ_A).

7 Single-Opponent Glicko-2 Update

This section recalls the standard Glicko-2 update for a player i facing a set of opponents j. In our application, each player has exactly one opponent: the aggregated opposing team.

7.1 Expected Score

For a player with parameters $(\mu_i, \phi_i, \sigma_i)$ and a single opponent $(\mu_{\text{opp}}, \phi_{\text{opp}})$, define

$$g(\phi_{\text{opp}}) = \left(1 + \frac{3\phi_{\text{opp}}^2}{\pi^2}\right)^{-1/2}. \tag{13}$$

The expected score for player i is

$$E_i = \frac{1}{1 + \exp\left(-g(\phi_{\text{opp}})\left(\mu_i - \mu_{\text{opp}}\right)\right)}. \tag{14}$$

7.2 Variance Term v

For a single opponent, the variance term simplifies to

$$v = \left[g(\phi_{\text{opp}})^2 E_i(1 - E_i)\right]^{-1}. \tag{15}$$

7.3 Delta Term Δ

Given the actual score s_i for player i (1 if their team wins, 0 if their team loses, 0.5 for draw), define

$$\Delta = v \cdot g(\phi_{\text{opp}}) \cdot (s_i - E_i). \tag{16}$$

7.4 Volatility Update

The volatility parameter σ_i is updated by solving a one-dimensional optimization problem. Let

$$a = \ln(\sigma_i^2), \tag{17}$$

and define an iterative scheme (in this repo implementation Illinois) to find the new a' such that

$$f(a') = 0, \tag{18}$$

where

$$f(x) = \frac{e^x(\Delta^2 - \phi_i^2 - v - e^x)}{2(\phi_i^2 + v + e^x)^2} - \frac{x - a}{\tau^2}. \tag{19}$$

Here τ is a system parameter controlling how quickly volatility can change.

Once a' is found, the updated volatility is

$$\sigma_i' = \exp\left(\frac{a'}{2}\right). \tag{20}$$

7.5 Intermediate and Final Rating Deviation

First compute the intermediate deviation:

$$\phi_i^* = \sqrt{\phi_i^2 + \sigma_i'^2}. \tag{21}$$

Then compute the new deviation:

$$\phi_i' = \left(\frac{1}{\phi_i^{*2}} + \frac{1}{v}\right)^{-1/2}. \tag{22}$$

7.6 Updated Rating Mean

Finally, the updated mean (before performance weighting) is

$$\mu_i^* = \mu_i + \phi_i'^2 \cdot g(\phi_{\text{opp}})\,(s_i - E_i). \tag{23}$$

At this point, the standard Glicko-2 update would set $(\mu_i', \phi_i', \sigma_i') = (\mu_i^*, \phi_i', \sigma_i')$.

8 Performance-Scaled Glicko-2 Update

To incorporate individual performance, we apply the Glicko-2 update as above to obtain intermediate values $(\mu_i^*, \phi_i', \sigma_i')$ and then modify only the rating mean based on the sign-aware scaling factor f_i.

8.1 Definition of the Scaled Update

Let

$$\Delta\mu_i = \mu_i^* - \mu_i \tag{24}$$

be the Glicko-2 change in the internal rating mean.

Compute the sign-aware scaling factor:

$$f_i = \min\left\{f_{\max}, \max\left\{f_{\min}, 1 + \beta \cdot \text{sign}(\Delta\mu_i) \cdot z_i\right\}\right\}. \tag{25}$$

The final mean after performance scaling is:

$$\mu_i' = \mu_i + f_i \cdot \Delta\mu_i. \tag{26}$$

The deviation and volatility are not scaled by performance:

$$\phi_i' \text{ as computed in the standard Glicko-2 update,} \tag{27}$$
$$\sigma_i' \text{ as computed in the standard Glicko-2 update.} \tag{28}$$

Thus, performance influences the *magnitude* of the rating change for the match but not the uncertainty parameters.

8.2 Optional Clamping

To avoid extreme rating jumps, one may clamp the resulting change:

$$\Delta\mu_i' = \mu_i' - \mu_i, \tag{29}$$

and then limit it by

$$\Delta\mu_i^{\text{final}} = \min\{\Delta_{\max}, \max\{-\Delta_{\max}, \Delta\mu_i'\}\}, \tag{30}$$

with some maximum change $\Delta_{\max} > 0$. The final rating mean becomes

$$\mu_i^{\text{final}} = \mu_i + \Delta\mu_i^{\text{final}}. \tag{31}$$

9 Full Per-Match Update Algorithm

For each match between Team A and Team B:

1. For every player i, retrieve $(R_i, \text{RD}_i, \sigma_i)$ and convert to (μ_i, ϕ_i).
2. Compute team internal parameters (μ_A, ϕ_A) and (μ_B, ϕ_B).

3. Compute performance scores p_i for all $i \in A \cup B$.
4. For each team $T \in \{A, B\}$:
   (a) Compute $\bar{p}_T$, s_T, z_i for $i \in T$.
5. For each player $a_i \in A$:
   (a) Let $(\mu_{\text{opp}}, \phi_{\text{opp}}) = (\mu_B, \phi_B)$.
   (b) Let $s_i = S_A$ (team outcome).
   (c) Perform the standard single-opponent Glicko-2 update using $(\mu_i, \phi_i, \sigma_i)$, $(\mu_{\text{opp}}, \phi_{\text{opp}})$, and s_i, obtaining $(\mu_i^*, \phi_i', \sigma_i')$.
   (d) Compute $\Delta\mu_i = \mu_i^* - \mu_i$.
   (e) Compute sign-aware scaling factor $f_i = 1 + \beta \cdot \text{sign}(\Delta\mu_i) \cdot z_i$, clamped to $[f_{\min}, f_{\max}]$.
   (f) Set $\mu_i' = \mu_i + f_i \cdot \Delta\mu_i$.
   (g) Optionally clamp $\mu_i' - \mu_i$ to a maximum magnitude.
6. For each player $b_j \in B$, repeat the same procedure with $(\mu_{\text{opp}}, \phi_{\text{opp}}) = (\mu_A, \phi_A)$ and $s_j = S_B$.
7. Convert each player's updated internal parameters back to (R_i, RD_i):

$$R_i = 173.7178 \cdot \mu_i' + R_0,$$
$$\text{RD}_i = 173.7178 \cdot \phi_i'.$$

10 Recent Performance Tracking

To capture short-term form changes (hot streaks, cold streaks), we maintain a recent performance index for each player.

10.1 Performance Normalization Within Team

After each match, for each player i in team T, we already have the z-score:

$$z_i = \frac{p_i - \bar{p}_T}{s_T}. \tag{32}$$

To prevent extreme outliers from dominating, clip the z-score:

$$z_i^{\text{clip}} = \min\{z_{\max}, \max\{-z_{\max}, z_i\}\}, \tag{33}$$

where $z_{\max} = 3.0$ (representing $\pm 3\sigma$ range).

10.2 Exponential Moving Average

Update the recent performance index using exponential moving average with target window $N_{\text{window}} = 10$ games:

$$\text{perfEMA}_i' = \begin{cases} z_i^{\text{clip}} & \text{if perfGames}_i = 0 \text{ (first game)}, \\ (1 - \alpha_i) \cdot \text{perfEMA}_i + \alpha_i \cdot z_i^{\text{clip}} & \text{otherwise}, \end{cases} \tag{34}$$

where the smoothing factor is:

$$\alpha_i = \begin{cases} \frac{1}{\text{perfGames}_i + 1} & \text{if perfGames}_i < N_{\text{window}} \text{ (bootstrap)}, \\ \frac{2}{N_{\text{window}} + 1} & \text{if perfGames}_i \geq N_{\text{window}} \text{ (steady state)}. \end{cases} \tag{35}$$

Then increment: $\text{perfGames}_i' = \text{perfGames}_i + 1$.

10.3 Interpretation

- $\text{perfEMA}_i \approx 0$: player consistently performs at team average level.
- $\text{perfEMA}_i \approx +1$: player consistently outperforms teammates by $\sim 1\sigma$.
- $\text{perfEMA}_i \approx -1$: player consistently underperforms teammates by $\sim 1\sigma$.

11 Effective Rating

For matchmaking and team balancing, we blend the long-term Glicko-2 rating with recent performance.

11.1 Recent Rating

Define the recent rating as the base rating adjusted by recent performance:

$$R_i^{\text{recent}} = R_i + \beta \cdot \text{perfEMA}_i, \tag{36}$$

where $\beta = 80$ is the rating boost per 1σ of performance.

Cap the boost to prevent excessive deviation from long-term rating:

$$\Delta_{\max}^{\text{boost}} = \min\{2 \cdot \text{RD}_i, 200\}, \tag{37}$$

and clamp:

$$R_i^{\text{recent}} = R_i + \min\{\Delta_{\max}^{\text{boost}}, \max\{-\Delta_{\max}^{\text{boost}}, \beta \cdot \text{perfEMA}_i\}\}. \tag{38}$$

11.2 Effective Rating Blending

Blend long-term and recent ratings using RD_i as a trust factor:

$$w_{\text{RD}} = \frac{\text{RD}_i^2}{\text{RD}_i^2 + C^2}, \tag{39}$$

where $C = 80$ is the RD scale constant.

The blending weight is:

$$w_i = 0.5 \cdot w_{\text{RD}}. \tag{40}$$

The effective rating is:

$$R_i^{\text{eff}} = R_i + w_i \cdot (R_i^{\text{recent}} - R_i). \tag{41}$$

11.3 Behavior

- Low RD_i (established player): w_i is small, $R_i^{\text{eff}} \approx R_i$ (90% Glicko, 10% recent).
- High RD_i (uncertain player): w_i is large, more influence from recent performance (60% Glicko, 40% recent).

12 Inactivity Decay

Rating deviation increases during periods of inactivity to reflect growing uncertainty.

12.1 Activity Threshold

A player is considered active if they played at least $N_{\min} = 3$ rounds in the past D days.

12.2 Rating Period Decay

If a player is inactive, calculate the number of rating periods elapsed:

$$N_{\text{periods}} = \left\lfloor \frac{\Delta t}{T_{\text{period}}} \right\rfloor, \tag{42}$$

where Δt is days since last match and $T_{\text{period}} = 7$ days per rating period.

For each rating period, apply Glicko-2 standard decay in internal scale:

$$\phi_i' = \sqrt{\phi_i^2 + \sigma_i^2}. \tag{43}$$

Cap at maximum:

$$\phi_i' = \min\left\{ \phi_i', \frac{\text{RD}_{\max}}{173.7178} \right\}, \tag{44}$$

where $\text{RD}_{\max} = 350$.

13 Team Balancing Algorithm

The system includes an algorithm to create balanced teams from a lobby of players.

13.1 Player Score Computation

For each player i, we use the effective rating as the balancing score:

$$S_i = R_i^{\text{eff}} \tag{45}$$

The effective rating R_i^{eff} already incorporates uncertainty through RD-weighted blending of long-term Glicko-2 rating and recent performance (see Section 10). No additional risk adjustment is needed or applied.

13.2 Team Strength and Uncertainty

For a team T with players $\{i_1, \ldots, i_k\}$:

$$S_T = \sum_{i \in T} S_i, \tag{46}$$

$$U_T = \sqrt{\sum_{i \in T} \text{RD}_i^2}. \tag{47}$$

13.3 Objective Function

For a proposed team split into Team 0 and Team 1, the objective is:

$$J(T_A, T_B) = \left| \frac{S_{T_A}}{|T_A|} - \frac{S_{T_B}}{|T_B|} \right| + \lambda \left| \frac{U_{T_A}}{\sqrt{|T_A|}} - \frac{U_{T_B}}{\sqrt{|T_B|}} \right|, \tag{48}$$

where $\lambda = 0.8$ is the uncertainty balance weight.

Using averages ensures fair comparison for uneven teams (e.g., 4v3).

13.4 Constraints

- **Top-2 separation**: The top 2 players by effective rating must be on different teams.
- **Uneven team rule**: For odd player counts (e.g., 7 players → 3v4), the top player must be placed in the smaller team to compensate for numerical disadvantage.

13.5 Optimization

The algorithm enumerates all valid team combinations respecting constraints and selects the assignment minimizing $J(T_A, T_B)$.

Tie-breaking (in order):

1. Smallest pure rating difference $|\bar{R}_{T_A} - \bar{R}_{T_B}|$.
2. Smallest uncertainty difference $|U_{T_A} - U_{T_B}|$.

14 Design Notes

- Win/loss remains the primary driver of rating changes via the Glicko-2 core update. Performance only modulates the magnitude of this change.
- The average player on a team always has a scaling factor f_i close to 1, so the team's average rating change is close to what standard Glicko-2 would produce for a team-based approximation.
- The sign-aware formula ensures proper incentives:
  - In wins: strong performers gain more, weak performers gain less.
  - In losses: strong performers lose less, weak performers lose more.

  This correctly rewards carrying and penalizes poor play regardless of outcome.
- The performance score p_i is game-dependent and should be constructed so that roles (e.g. support vs. damage) are fairly evaluated relative to other players of the same role.
- Recent performance tracking captures short-term form independent of long-term rating. This addresses players on winning/losing streaks or skill improvements not yet reflected in Glicko-2.
- Effective rating is used only for matchmaking and balancing, not for the Glicko-2 update itself. This separation maintains rating system integrity while improving match quality.
- Inactivity decay ensures returning players are matched more cautiously until their skill level is re-established.
- Team balancing considers both skill and uncertainty, creating not just balanced ratings but also balanced confidence levels.