GDC

Math for Game Programmers: Ranking Systems; Elo, TrueSkill and Your Own

Mario Izquierdo
Sr. Software Engineer at twitch

Ranking (`TrueSkill`)

map #1

#1 player

map #2

#2 player

#3 player

map #3

map #4

map #5

#4 player

player vs map

Ranking Systems

~ 30 mins talk

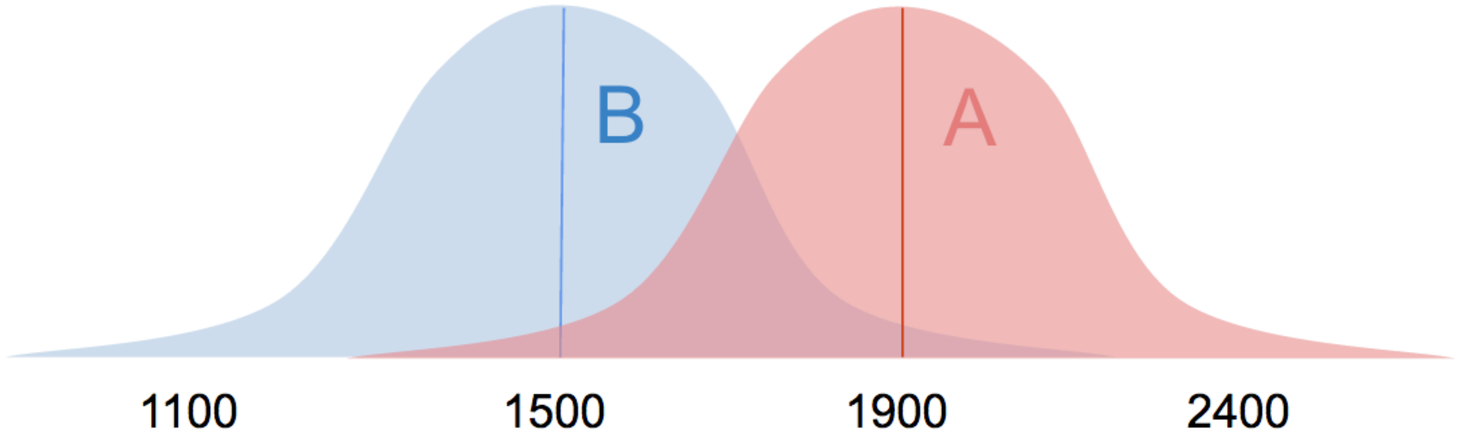
- **Elo**
- **TrueSkill**
- Practical Considerations

Elo Rating System

Árpád Élő (1903 - 1992)

- Physics profesor and master chess player.
- Elo's system constituted an improvement on the previous Harkness System.
- Elo's system was adopted by the FIDE (World Chess Federation) in **1970**.
- Published "The Rating of Chessplayers, Past and Present” in **1978**.
- Fun fact: Up until the mid-80’s, Elo himself made the rating calculations!

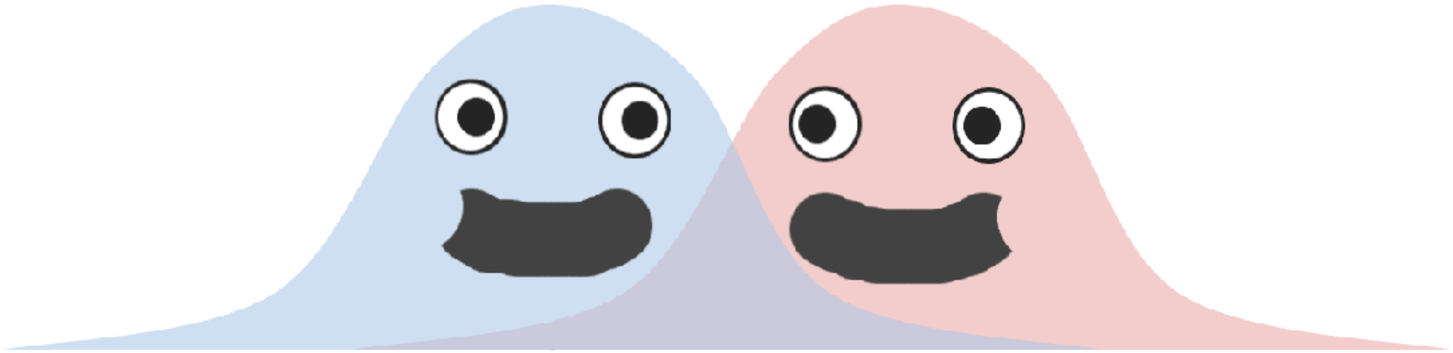
Elo Rating System: Normal Distribution



Assumption: Chess performance is a normally distributed **random variable.**

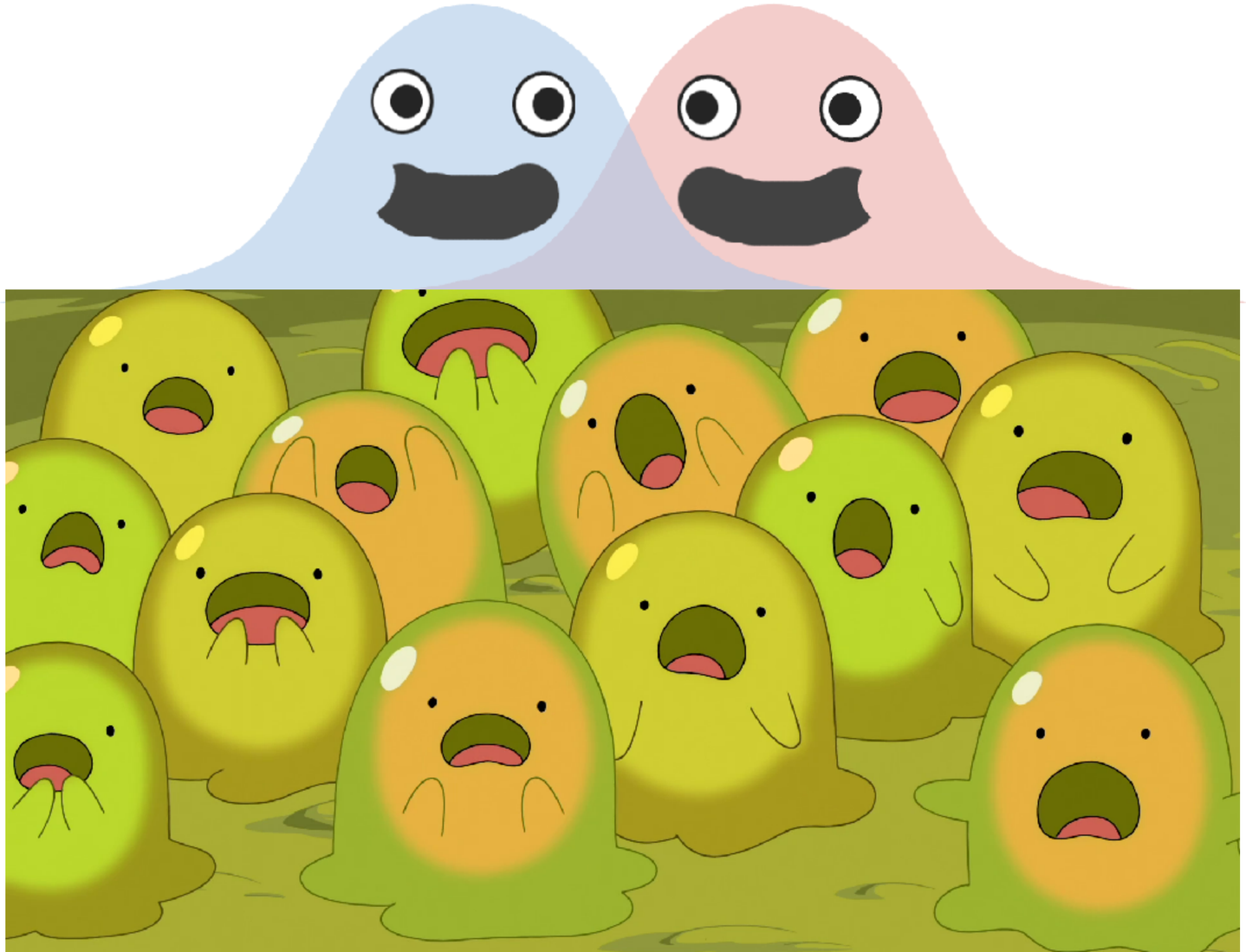
Using some simplifications (i.e. constant standard deviation) makes easy to calculate the **Expected** score of a match (probability of win) for two given player skill levels.

Elo Rating System: ~~Normal Distribution~~ “Slime Curve”



In the eyes of ELO, you are all “slime people”

Elo Rating System: ~~Normal Distribution~~ “Slime Curve”



Elo Rating System: Formula

After a given match, rating points are transferred between players:

`RatingDiff = (Score - Expected) * K-factor`

Where:

`Score` is 0 = loss, 0.5 = draw, 1 = win
`Expected` is 0 to 1, the probability of winning
`K-factor` is a constant for maximum change (update “speed”)

Elo Rating System: Formula

After a given match, rating points are transferred between players:

```
RatingDiff = (Score - Expected) * K-factor
```

Much of the trick is in figuring out what the **Expected** result of a game is. The original ELO system uses the following formula (from the Normal dist.):

```
Expected[A] = 1/(1+10^(Rating[B-A]/400))
```

Elo Rating System: Formula

After a given match, rating points are transferred between players:

```
RatingDiff = (Score - Expected) * K-factor
```

Much of the trick is in figuring out what the **Expected** result of a game is. The original ELO system uses the following formula (from the Normal dist.):

```
Expected[A] = 1/(1+10^(Rating[B-A]/400))
```

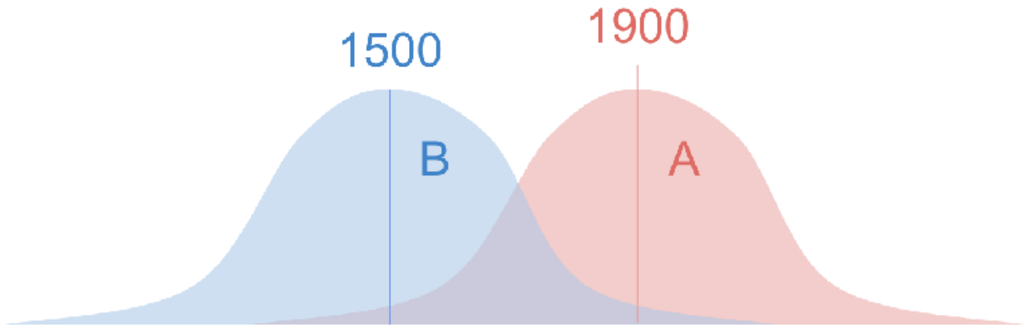
Which gives Player **A** the chances of winning for each **Rating[B-A]**

0: **50%**, 100: **64%**, 200: **76%**, 300: **85%**, 400: **91%**, 500: **95%**, 600: **97%**

Elo Rating System: Formula

For example, with ratings **Bob: 1500** and **Alice: 1900**

```
Expected[Bob]   = 0.09
Expected[Alice] = 0.91
```



with K-factor of **32**, the update for **Bob** is:

Bob wins (Score=1):

```
(1 - 0.09) * 32 = +29
```

Bob draws (Score=0.5):

```
(0.5 - 0.09) * 32 = +13
```

Bob loses (Score=0):

```
(0 - 0.09) * 32 = -3
```

Outcomes for **Bob** vs **Alice** (**9%** chance of winning):

Win	+ 29
Draw	+13
Loss	- 3

Elo Rating System: Comments

- Widely used and **well understood**.
- Only works for **1vs1**.
- It's **simplicity** is also its great strength,
  being able to calculate and understand the algorithm makes it feel “fair".
- The **K-factor** needs to be adjusted for new vs experienced players.
- Nowadays there are many **different implementations**, almost none of them follows Elo's original suggestions precisely.
- New players can take a long time to converge to their correct skill rating.

TrueSkill Ranking System

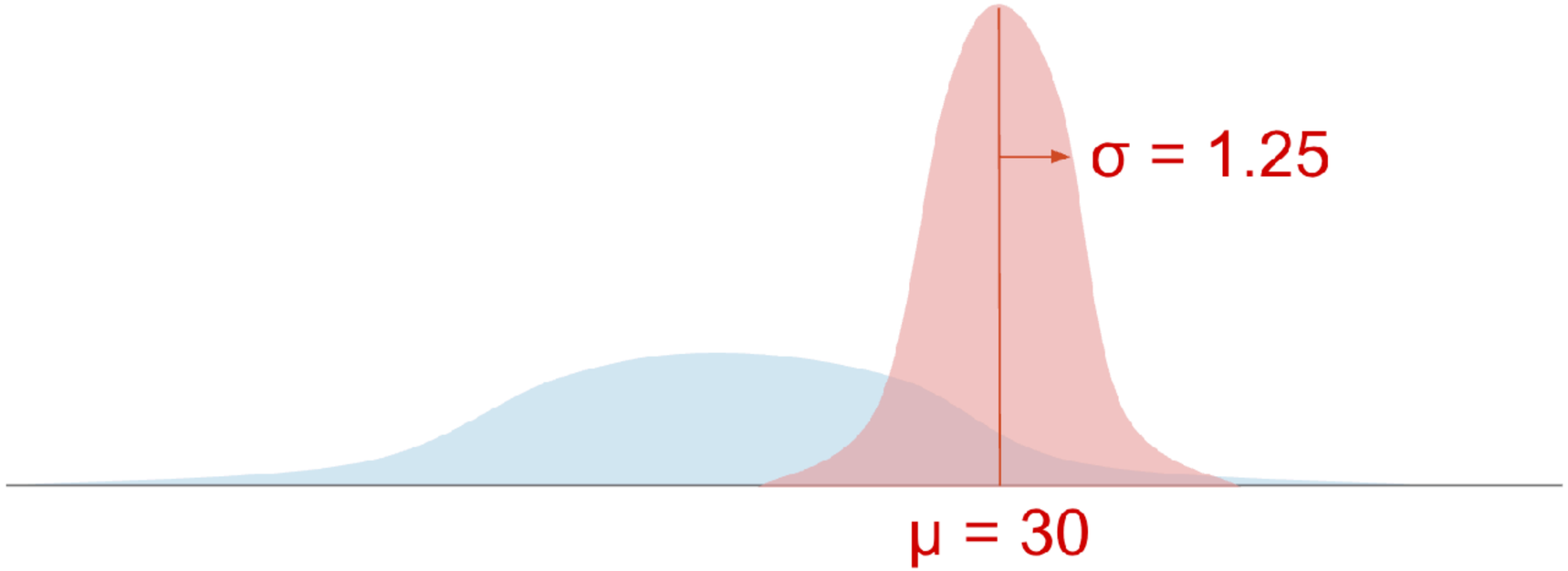
TrueSkill Ranking System

- Developed by Microsoft Research in 2005.
- Designed for matchmaking on Xbox Live.
- Improves upon Elo’s ideas.

TrueSkill: two variables μ, σ

μ: average skill

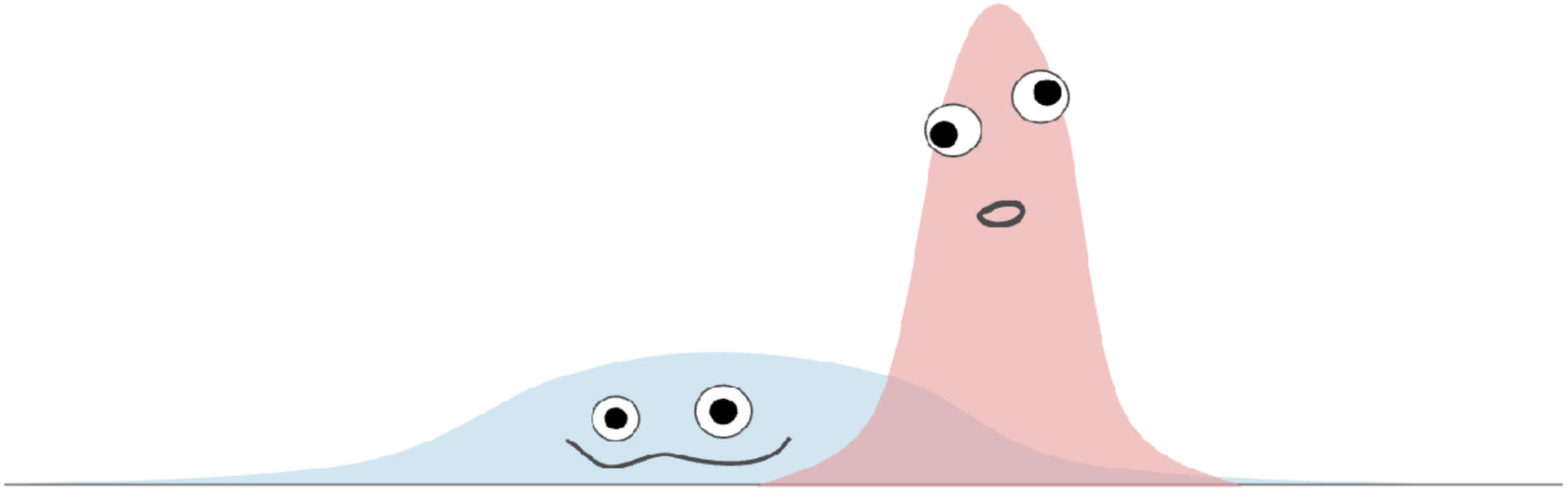
σ: sigma (degree of uncertainty)



TrueSkill: two variables μ, σ

μ: average skill

σ: sigma (degree of uncertainty)

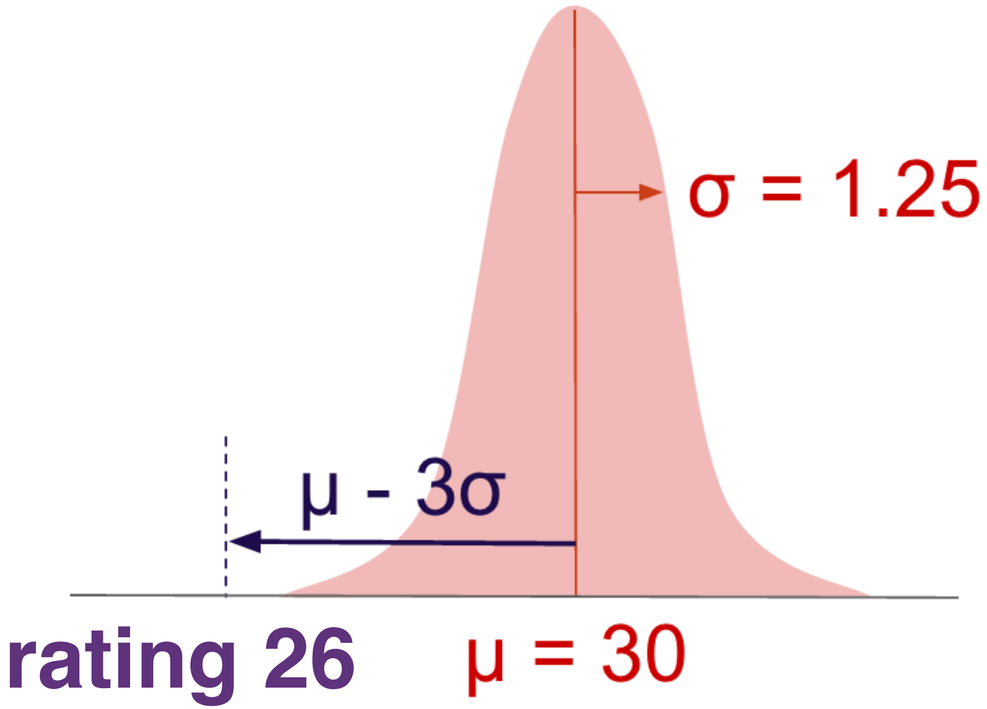


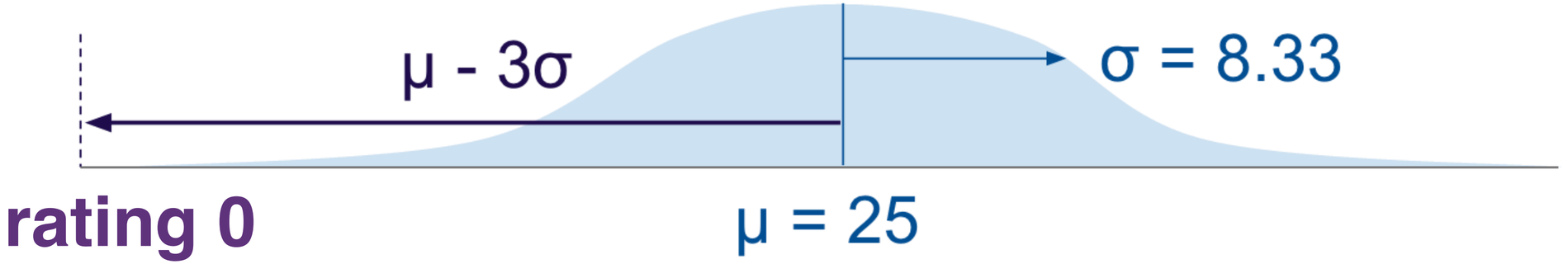
New player slimes are “short and fat”

Advanced players are “tall and thin”

TrueSkill: Visible Rating is μ - 3σ

- TrueSkill suggest using a very conservative number **μ - 3σ**
- Actual skill is **98%** likely to be **more** than the visible rating



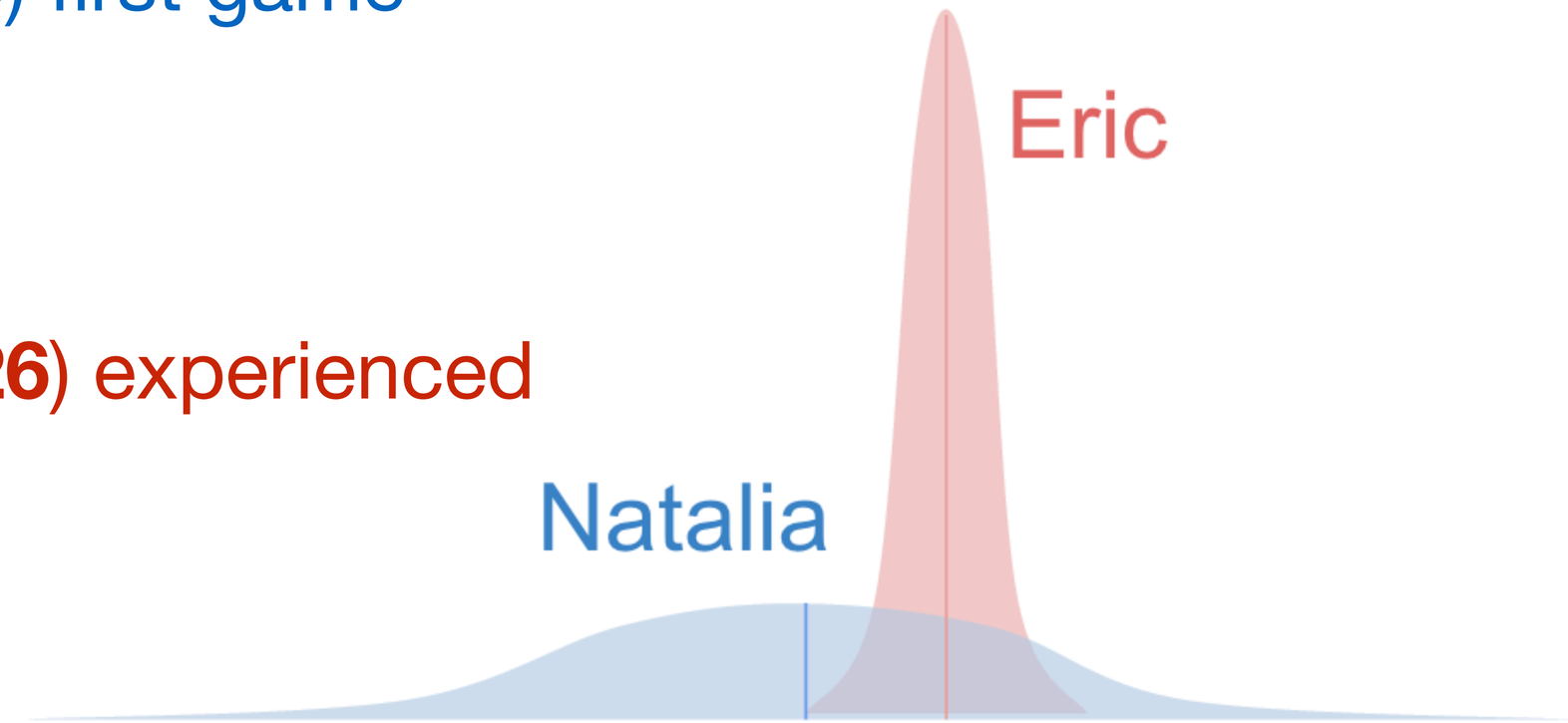


TrueSkill: Skill Update, 1vs1 (simplest case)

For example:

Natalia:
μ 25, σ 8.33 (**rating 0**) first game

Eric:
μ 30, σ 1.25 (**rating 26**) experienced

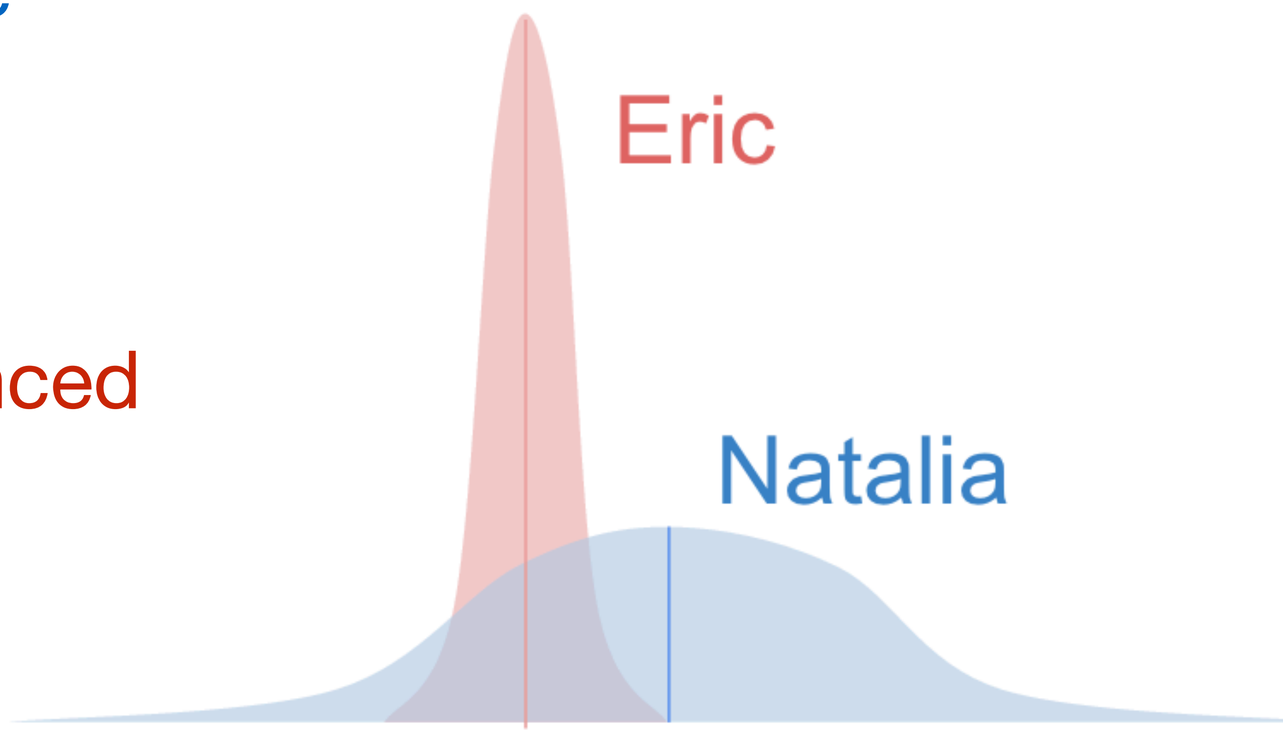


TrueSkill: Skill Update, 1vs1 (simplest case)

For example: Big surprise! Natalia wins!!!

Natalia:
μ 25, σ 8.33 (**rating 0**) first game
μ 33, σ 5.97 (**rating 15**) win :D

Eric:
μ 30, σ 1.25 (**rating 26**) experienced
μ 29, σ 1.25 (**rating 25**) loss :(



TrueSkill: Formulas 1vs1 (simplest case)

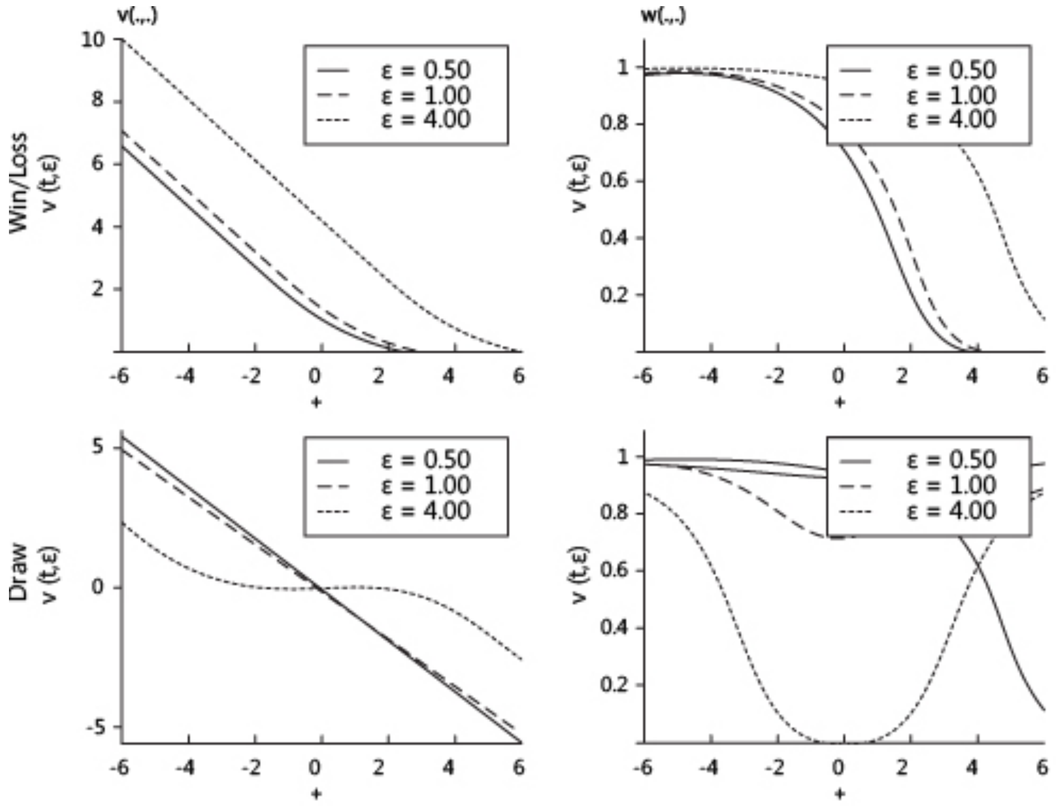
$$\mu_{winner} \leftarrow \mu_{winner} + \frac{\sigma^2_{winner}}{c} \cdot v\left(\frac{(\mu_{winner} - \mu_{loser})}{c}, \frac{\varepsilon}{c}\right)$$

$$\mu_{loser} \leftarrow \mu_{loser} - \frac{\sigma^2_{loser}}{c} \cdot v\left(\frac{(\mu_{winner} - \mu_{loser})}{c}, \frac{\varepsilon}{c}\right)$$

$$\sigma^2_{winner} \leftarrow \sigma^2_{winner} \cdot \left[1 - \frac{\sigma^2_{winner}}{c^2} \cdot w\left(\frac{(\mu_{winner} - \mu_{loser})}{c}, \frac{\varepsilon}{c}\right)\right]$$

$$\sigma^2_{loser} \leftarrow \sigma^2_{loser} \cdot \left[1 - \frac{\sigma^2_{loser}}{c^2} \cdot w\left(\frac{(\mu_{winner} - \mu_{loser})}{c}, \frac{\varepsilon}{c}\right)\right]$$

$$c^2 = 2\beta^2 + \sigma^2_{winner} + \sigma^2_{loser}$$



β^2 (unknown) is the variance of performance around the skill.
ε is the “draw margin”, that can be adjusted for each game mode.
v(…) and w(…) are explained through the plots (exact definition is not public).

TrueSkill: Formulas 1vs1 (simplest case)

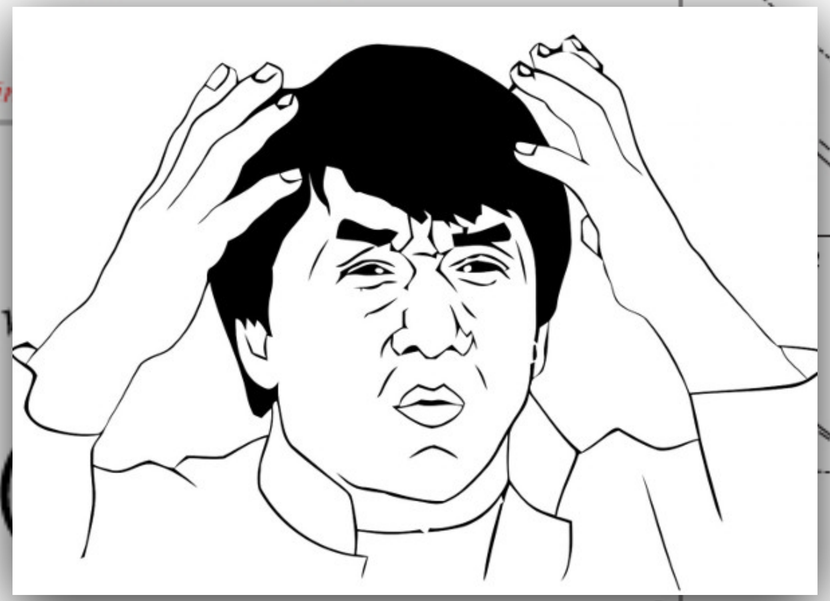
$$\mu_{winner} \leftarrow \mu_{winner} + \frac{\sigma^2_{winner}}{c} \cdot v\left(\frac{(\mu_{winner} - \mu_{loser})}{c}, \frac{\varepsilon}{c}\right)$$

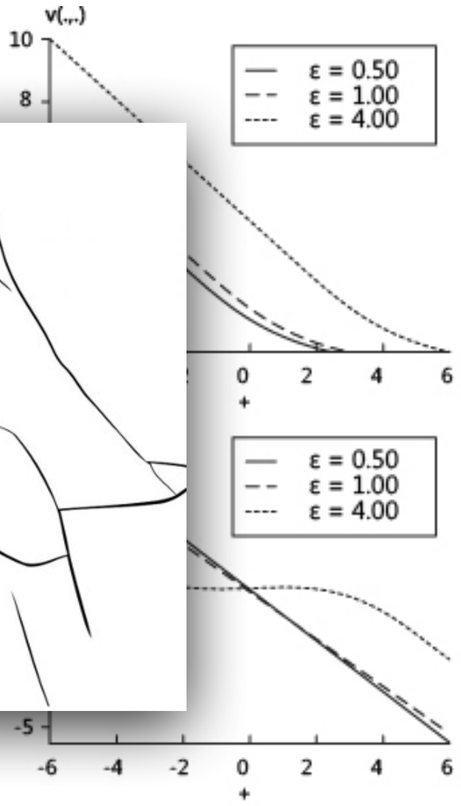
$$\mu_{loser} \leftarrow \mu_{loser} - \frac{\sigma^2_{loser}}{c} \cdot v\left(\frac{(\mu_{win}}{c}\right.$$

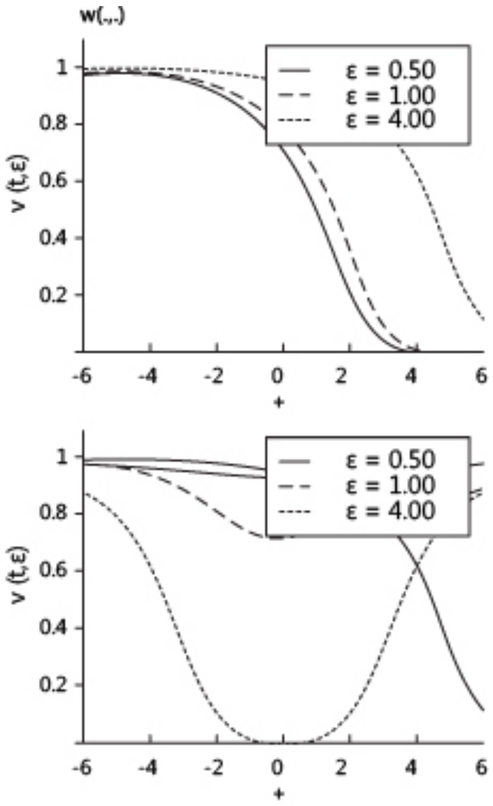
$$\sigma^2_{winner} \leftarrow \sigma^2_{winner} \cdot \left[1 - \frac{\sigma^2_{winner}}{c^2} \cdot v\right.$$

$$\sigma^2_{loser} \leftarrow \sigma^2_{loser} \cdot \left[1 - \frac{\sigma^2_{loser}}{c^2} \cdot w\right.$$

$$c^2 = 2\beta^2 + \sigma^2_{winner} + \sigma^2_{loser}$$







Yeah, it’s complicated … but don’t worry, it’s all in the Interweb

TrueSkill: Comments

- **Flexible**, can model many different types of competitive games.
- **Quickly Converges** to the player's true skill (only a few games).
- Calculations are very **complex**. Although today's computers can handle it, this may confuse players and sometimes seen "unfair".
- Makes easy to model new players (initial rating 0 and uncertainty).
- It is **proprietary** and may require a license to be used, a great open alternative is the Glicko system (although limited to 1v1).

Considerations To Make Your Own Ranking System

What makes a Good Ranking System?

What makes a Good Ranking System?

- Credibility (fair/unfair)

What makes a Good Ranking System?

- Credibility (fair/unfair)
- Easy to implement

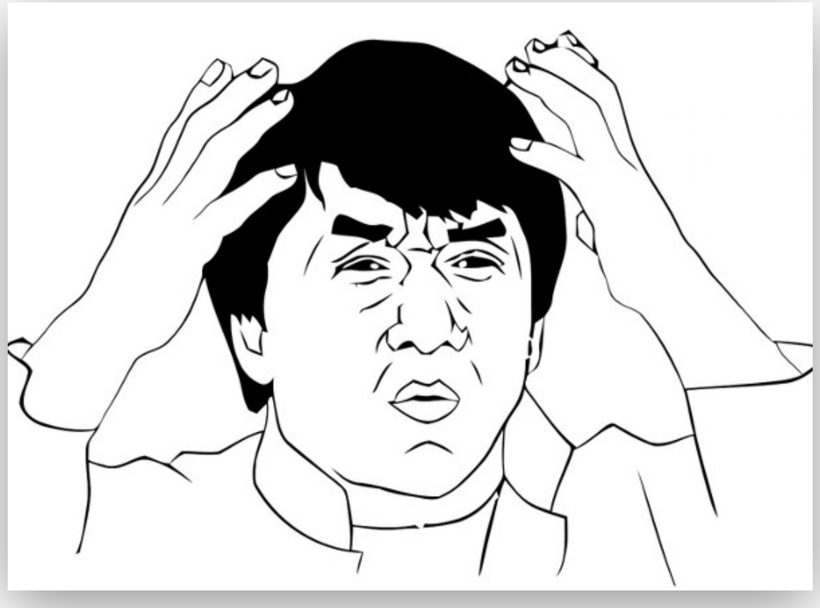
What makes a Good Ranking System?

- Credibility (fair/unfair)
- Easy to implement
- Depending on the case also …
  - Quality Matchmaking
  - Accurate Predictions
  - Fun (approachable, feel of mastery, status)
  - Many other details …

Lets review a few practical issues:

- Complexity
- Subjectivity
- Inflation
- Cold Start
- Time Decay
- The Fun Factor
- Gaming the System
- Margin of Victory
- Home advantage
- 50% win ratio
- Beyond Games

Complexity



- More precise and flexible is more complex.
- Hard to understand can feel “unfair”.
- Complex systems are easier cheat.
- Simplicity goes a long way, Elo is still widely used world wide.

Subjectivity

- Can handle a great deal of complexity.
- May also feel “unfair”.
- Even algorithmic systems like ELO have subjective elements like K-factor, or assuming that performance is normally distributed.

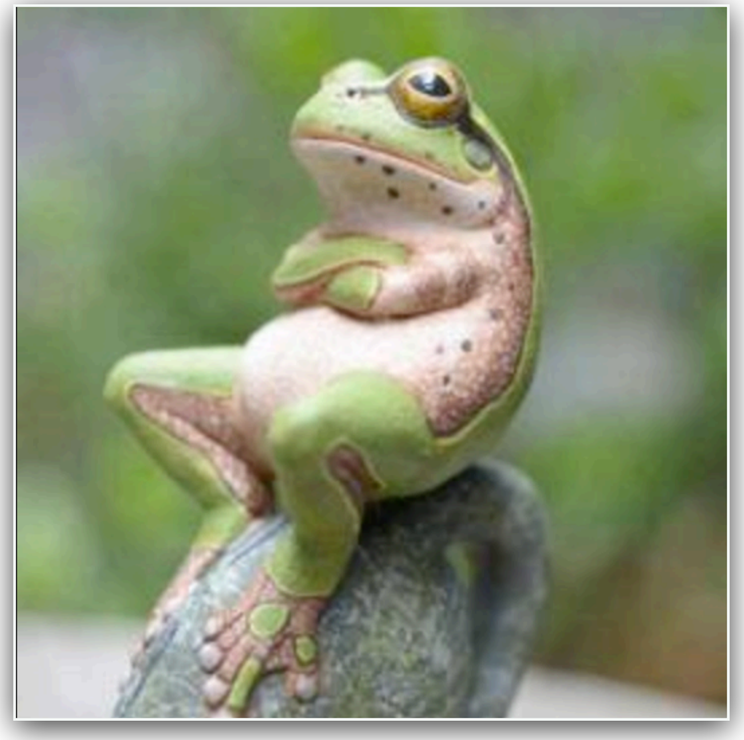
Inflation/Deflation of scores

- It is commonly believed that chess top level modern ratings are inflated, which makes hard to compare players from different ages.
- Sometimes need to inject points or modify system variables to adjust average scores.
- In practice, most games don't really have issues with inflation.

Cold Start

- We don’t know the skill of new players.
- TrueSkill and Glicko solve this problem by modeling uncertainty.
- Elo can solve it with K-factor.
- Placement matches are very useful.

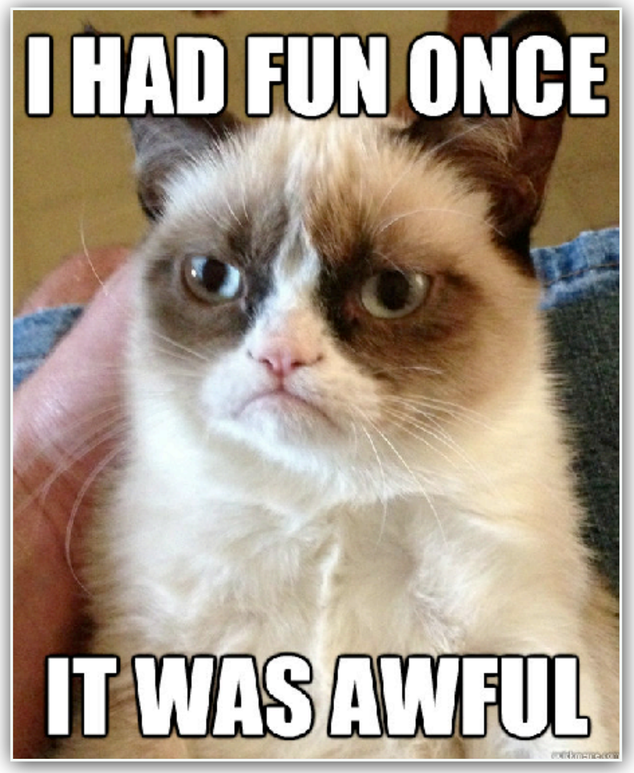
Time Decay



- Returning players may be out of practice.
- In Elo, the first games after a while may be frustrating.
- TrueSkill and Glicko can model Time Decay by increasing uncertainty (σ)

The Fun Factor

- The ranking can be brutally honest, and most players just want to **feel progression**.
- Use Ranking System for matchmaking, and **accumulative system** for progression.
- **Side-missions** (i.e. Hearthstone).
- **Locality** (play with the same group).
- **Hierarchy** (Silver, Gold, Diamond, Master).



Gaming the System

- Matchmaking should be random.
- Using Elo ratings on tournaments may discourage top players to participate.
- DCI (Magic: The Gathering) abandoned Elo on 2012 in favor of a new cumulative system named "Planeswalker Points"

Margin of Victory: Wins vs Points

- Most systems only count wins.
- Counting game points can help improve accuracy.
- But the margin can be manipulated for gambling, and can also promote unhealthy matches (i.e 15-0)

Home advantage

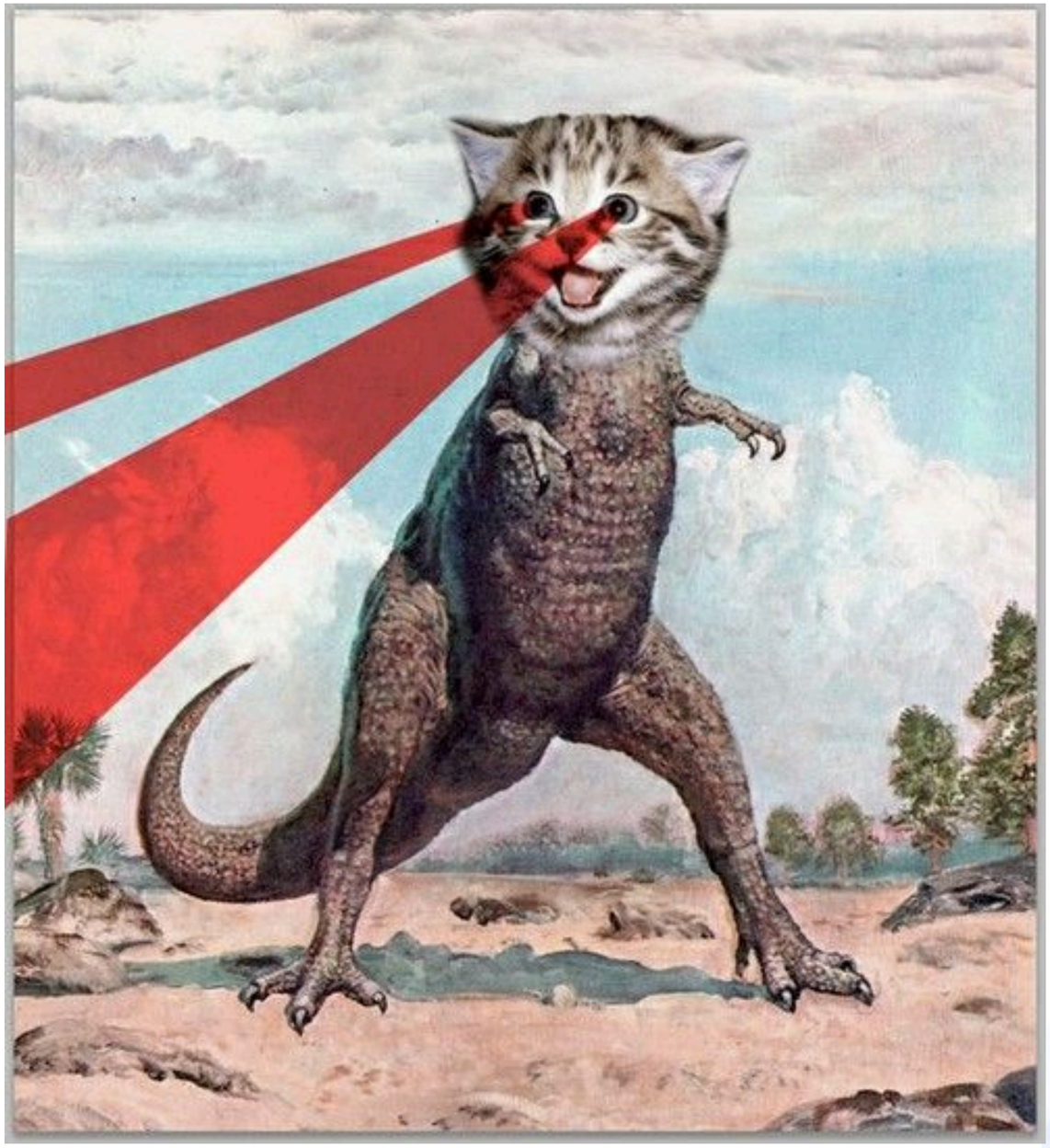
- In many sports like Basketball, playing home has greater chances of victory.
- This is just another variable that could be used to improve accuracy.
- In online games, this could be favorite maps, gear, etc.

The dreaded 50% win ratio

- A good Matchmaking System gives players a 50% chance of winning.
- But players like to win more often.
- Offer side activities (i.e. Starcraft Arcade) or non-competitive quick games.
- You could synthesize win streaks, at the cost of giving bad streaks later.

Beyond Games

- Ranking and rating systems are everywhere: Amazon, Yelp, Google search, etc.
- Zuckerberg's used a variation of Elo on his Facemash site, to rank Harvard's students.
- Any item that can be compared can be ranked (i.e. player vs map).



KEEP CALM AND LOVE CATS

Questions?