# Introduction to Probabilistic Machine Learning

Ralf Herbrich

Bayesian Ranking

# Overview

1. Ranking Problem
2. Probabilistic Ranking Models
3. TrueSkill: Expectation Propagation on Ranking Factor Graphs
4. TrueSkill Through Time

# Overview

1. **Ranking Problem**
2. Probabilistic Ranking Models
3. TrueSkill: Expectation Propagation on Ranking Factor Graphs
4. TrueSkill Through Time

# Motivation

- **Competition** is central to our lives
  - Innate biological trait
  - Driving principle of many sports

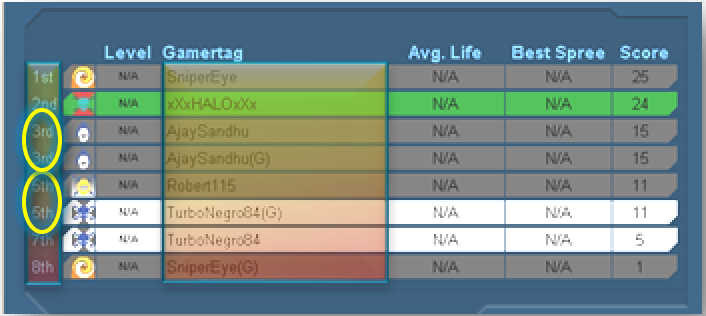
- **Chess rating** for fair competition
  - ELO: Developed in 1960 by Árpád Imre Élő (as a success to Harkness system)
  - Matchmaking system for Chess tournaments

- **Challenges** of online gaming
  1. Learn from few match outcomes efficiently
  2. Support multiple teams and multiple players per team
  3. Support draws and partial play as well as skill transfer over games

Árpád Imre Élő
(1903 – 1992)

# The Skill Rating Problem

- **Given**:
  - **Match outcomes**: Orderings among $k$ teams consisting of $n_1, n_2, \dots, n_k$ players.


- **Questions**:
  1. Skill $s_i$ for each player such that $s_i > s_j \Leftrightarrow P(\text{Player } i \text{ wins}) > P(\text{Player } j \text{ wins})$
  2. Global ranking among all players
  3. Fair matches between teams of players

# Overview

1. Ranking Problem
2. **Probabilistic Ranking Models**
3. TrueSkill: Expectation Propagation on Ranking Factor Graphs
4. TrueSkill Through Time

# Two-Player Match Outcome Model

- **Simple Two-Player Games**: Our data is the identity $i$ and $j$ of the two players and the outcome $Y \in \{-1, +1\}$ of a match between them.
  - **Bradley-Terry Model (1952)**: Model of a win of player $i$ given skills $s_i$ and $s_j$ is

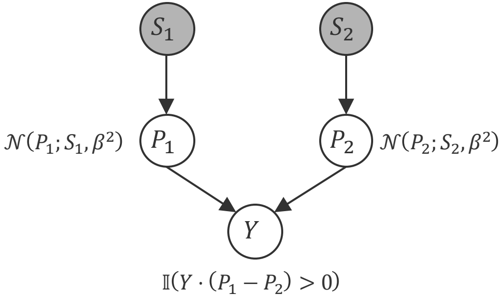
$$P\left(Y = 1 | s_i, s_j\right) = \frac{\exp(s_i)}{\exp(s_i) + \exp(s_j)} = \boxed{\frac{\exp\left(s_i - s_j\right)}{1 + \exp\left(s_i - s_j\right)}}$$

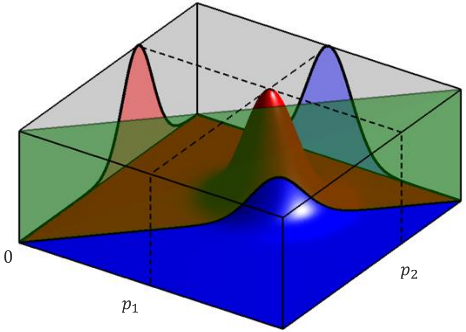
Logistic sigmoid in skill difference

  - **Thurstone Case V Model (1927)**: Model of a win given skills $s_i$ and $s_j$ is

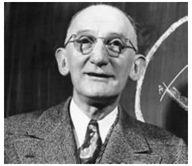
$$P\left(Y = 1 | s_i, s_j\right) = \boxed{\int_0^\infty \mathcal{N}\left(t; s_i - s_j, 2\beta^2\right) \mathrm{d}t}$$

Probit sigmoid in skill difference



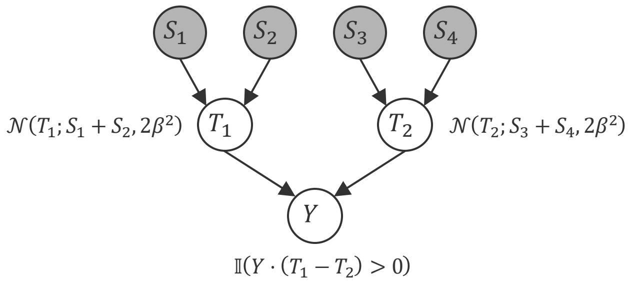

Ralph A. Bradley
(1923 – 2001)



Louis Leon Thurstone
(1887 – 1955)

# Two-Team Match Outcome Model

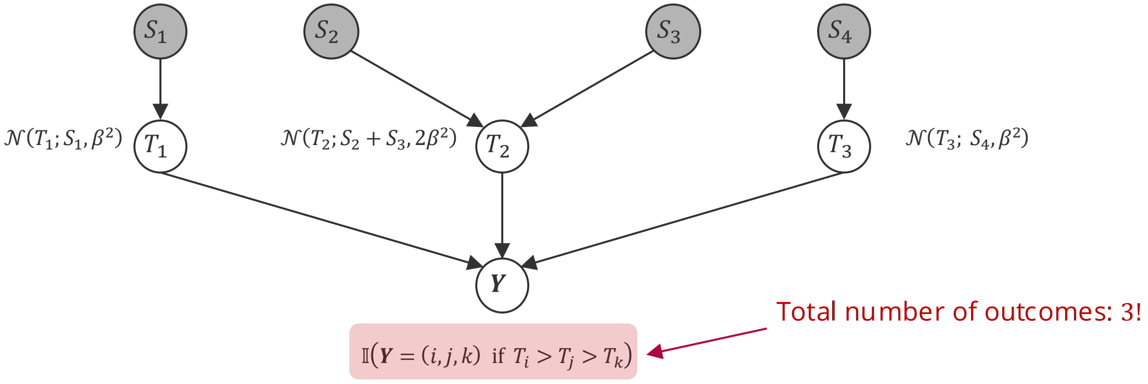
- **Team Assumption**: Performance of a team is the sum of the performances of its players




- **Pro**: Games where the team scores are additive (e.g., kill count in first-person shooter)
- **Con**: Games where the outcome is determined by a single player (e.g., fastest car in a race)
- **Observation**: Match outcomes correlate the skills of players
  - **Same Team**: Anti-correlated
  - **Opposite Teams**: Correlated

# Multi-Team Match Outcome Model

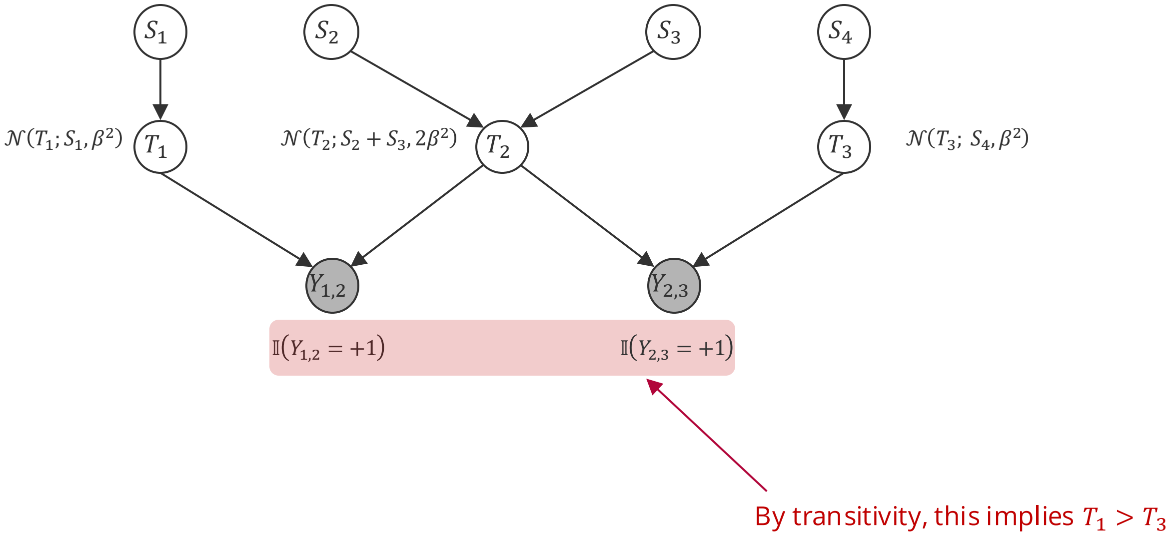
- **Possible Outcomes**: Permutations $\boldsymbol{Y} \in \{1,2,3\}^3$ of teams




- **Easy** to **sample** for given skills but computationally difficult to "invert"!

# From Match Outcomes to Pairwise Rankings

- **Learning**: In the ranking setting, we observe multi-team match outcomes and want to infer the skills of all single players!
- **Idea**: Leverage the transitivity of the real line of latent scores!

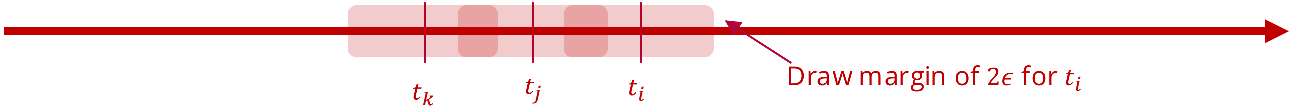



Introduction to Probabilistic Machine Learning

*Unit 6 – Bayesian Ranking*

# Modelling Draws and Partial Play

- **Draw Model**: Instead of $t_i > t_j$ for the winning team, we have three outcomes ($\epsilon > 0$)
  - **Team $i$ wins**: $t_i > t_j + \epsilon \Leftrightarrow t_i - t_j > \epsilon$
  - **Team $j$ wins**: $t_j > t_i + \epsilon \Leftrightarrow t_j - t_i > \epsilon$
  - **Teams draw**: $t_i \leq t_j + \epsilon \text{ and } t_j \leq t_i + \epsilon \Leftrightarrow |t_i - t_j| \leq \epsilon$
  - Pairwise draws in a chain **do not** model the actual event that all pairwise team performances are at most $\epsilon$ away from each other!




- **Partial Play**: If a player $i$ only participates for a fraction $\alpha_i \in [0,1]$ of the time in the match, then we model this assuming a linear contribution to the team skill by

$$T \sim \mathcal{N}(\cdot; \alpha_1 s_1 + \alpha_2 s_2, (\alpha_1^2 + \alpha_2^2)\beta^2)$$

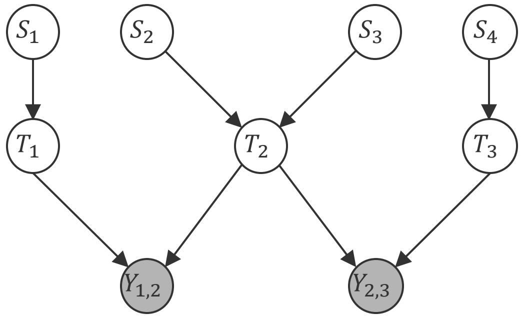
  - This only works if the fraction $\alpha_i$ is truly independent of the (predicted) match outcome!

# Overview

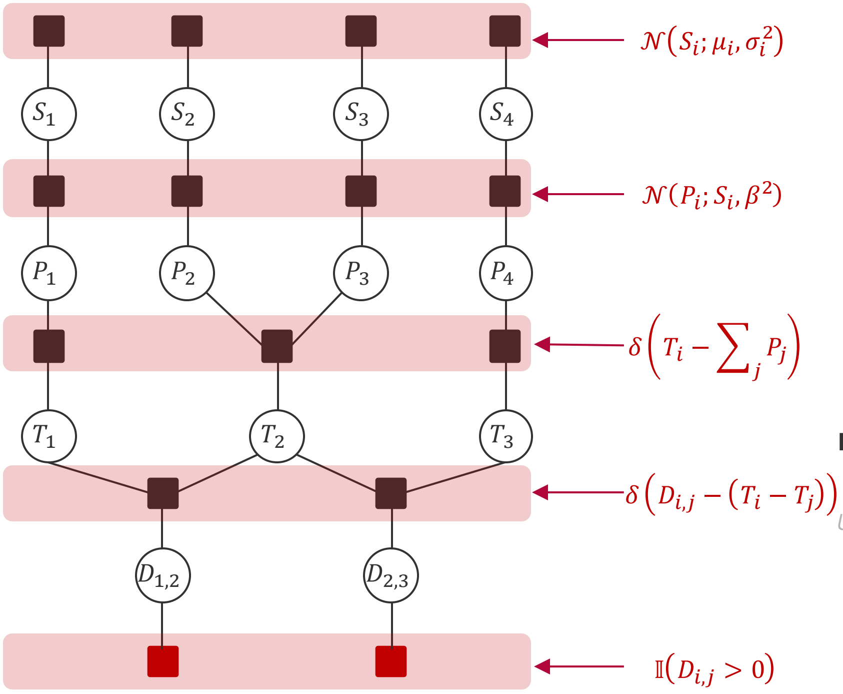
1. Ranking Problem
2. Probabilistic Ranking Models
3. **TrueSkill: Expectation Propagation on Ranking Factor Graphs**
4. TrueSkill Through Time

# TrueSkill Factor Tree

Bayesian Network




Factor Graph

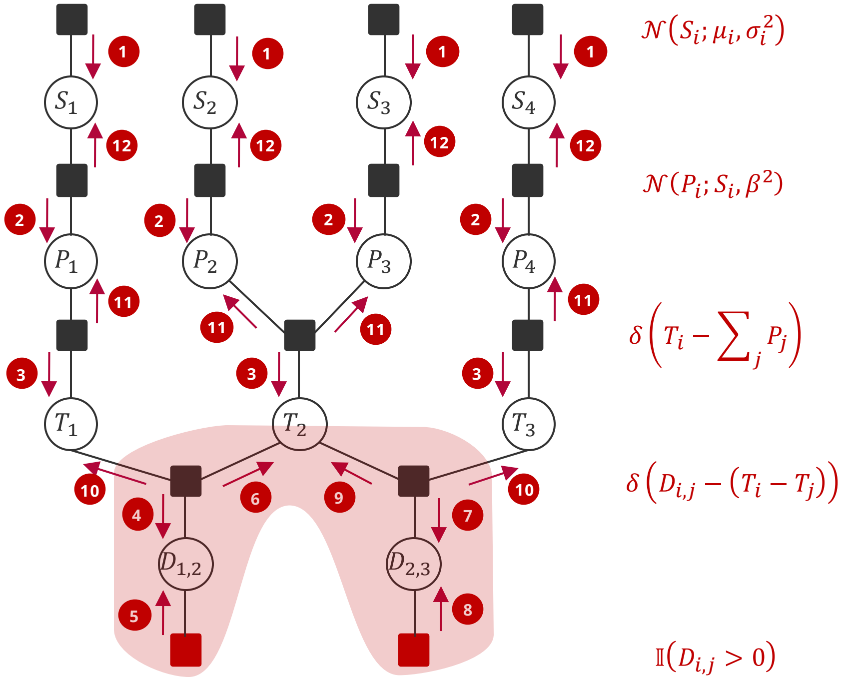



**Introduction to Probabilistic Machine Learning**

*Unit 6 – Bayesian Ranking*

# (Approximate) Message Passing in TrueSkill Factor Trees

TrueSkill Factor Graph

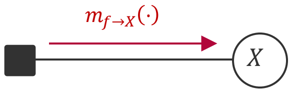



- Four Phases
  1. Pass prior messages (1)
  2. Pass messages *down* to the team performances (2 to 3)
  3. Iterate the approximate messages on the pairwise team differences (4 to 9)
  4. Pass messages back from *up* from team performances to player skill (10 – 12)

- Since this is a *tree*, the algorithm is guaranteed to converge!

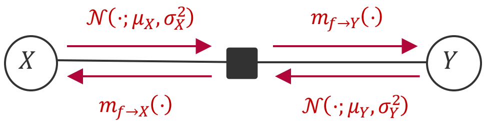
# Message Update Equations

### Gaussian Factor $\mathcal{N}(X;\mu,\sigma^2)$



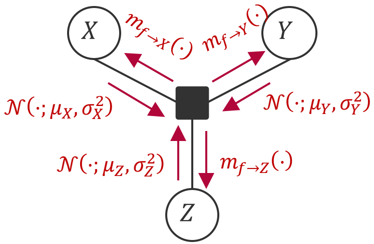

$$m_{f\to X}(x) = \mathcal{N}(x;\mu,\sigma^2)$$

### Gaussian Mean Factor $\mathcal{N}(Y;X,\beta^2)$



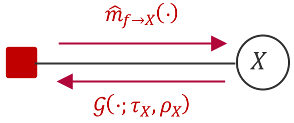

$$m_{f\to Y}(y) = \mathcal{N}(y;\mu_X,\sigma_X^2+\beta^2)$$
$$m_{f\to X}(x) = \mathcal{N}(x;\mu_Y,\sigma_Y^2+\beta^2)$$

### Weighted Sum Factor $\delta\big(Z-(aX+bY)\big)$




$$m_{f\to Z}(z) = \mathcal{N}(z;a\mu_X+b\mu_Y,a^2\sigma_X^2+b^2\sigma_Y^2)$$
$$m_{f\to Y}(y) = \mathcal{N}(y;(\mu_Z-a\mu_X)/b,(\sigma_Z^2+a^2\sigma_X^2)/b^2)$$
$$m_{f\to X}(x) = \mathcal{N}(x;(\mu_Z-b\mu_Y)/a,(\sigma_Z^2+b^2\sigma_Y^2)/a^2)$$

### Between Factor $\mathbb{I}(l \le X < u)$




$$\hat{m}_{f\to X}(x) = \mathcal{G}\left(x;\sqrt{\rho_X}\cdot\frac{V}{1-W}+\tau_X\cdot\frac{W}{1-W},\rho_X\cdot\frac{W}{1-W}\right)$$

Correction functions

$V := v_{l\sqrt{\rho_X},u\sqrt{\rho_X}}\left(\frac{\tau_X}{\sqrt{\rho_X}}\right)$

$W := w_{l\sqrt{\rho_X},u\sqrt{\rho_X}}\left(\frac{\tau_X}{\sqrt{\rho_X}}\right)$

of doubly-truncated Gaussians

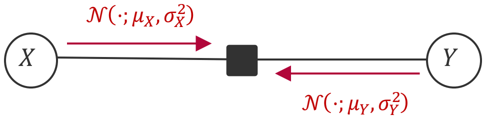
# Factor Normalizations

$$Z_{f_i} = \frac{\sum_{\{\boldsymbol{x}_{\mathrm{ne}(f_i)}\}} f_i(\boldsymbol{x}_{\mathrm{ne}(f_i)}) \prod_{j\in\mathrm{ne}(f_i)} \hat{m}_{X_j\to f_i}(x_j)}{\sum_{\{\boldsymbol{x}_{\mathrm{ne}(f_i)}\}} \prod_{j\in\mathrm{ne}(f_i)} \hat{m}_{f_i\to X_j}(x_j)\cdot \hat{m}_{X_j\to f_i}(x_j)}$$

## Gaussian Factor $\mathcal{N}(X;\mu,\sigma^2)$

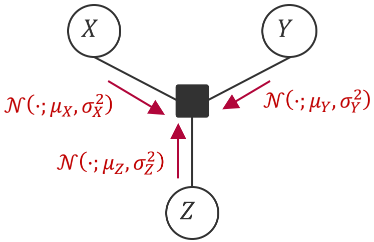

$$Z_f = 1$$

## Gaussian Mean Factor $\mathcal{N}(Y;X,\beta^2)$



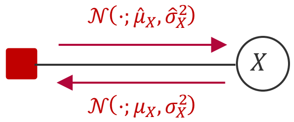

$$Z_f = \frac{1}{\mathcal{N}(\mu_X;\mu_Y,\sigma_X^2+\sigma_Y^2+\beta^2)}$$

## Weighted Sum Factor $\delta\big(Z-(aX+bY)\big)$




$$Z_f = \frac{1}{|a|\cdot|b|\cdot[\mathcal{N}(\mu_Z;a\mu_X+b\mu_Y,\sigma_Z^2+a^2\sigma_X^2+b^2\sigma_Y^2)]^2}$$

## Between Factor $\mathbb{I}(l \le X < u)$




$$\Phi(z) := \int_{-\infty}^{z} \mathcal{N}(x;0,1)\,\mathrm{d}x$$

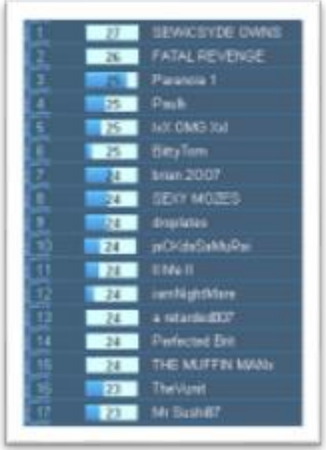
$$Z_f = \frac{\Phi\left(\frac{u-\mu_X}{\sigma_X}\right)-\Phi\left(\frac{l-\mu_X}{\sigma_X}\right)}{\mathcal{N}(\hat{\mu}_X;\mu_X,\sigma_X^2+\hat{\sigma}_X^2)}$$

# Decision Making: Match Quality and Leaderboards

- **Match Quality**: Decide if two players $i$ and $j$ should be matched
  - **Idea**: Pick the pair $(i, j)$ where the following quality is highest

$$\text{Quality}(i,j) = P\left(P_i \approx P_j | \mu_i - \mu_j, \sigma_i^2 + \sigma_j^2\right) = \lim_{\epsilon \to 0} P\left(\left|P_i - P_j\right| \leq \epsilon | \mu_i - \mu_j, \sigma_i^2 + \sigma_j^2\right)$$

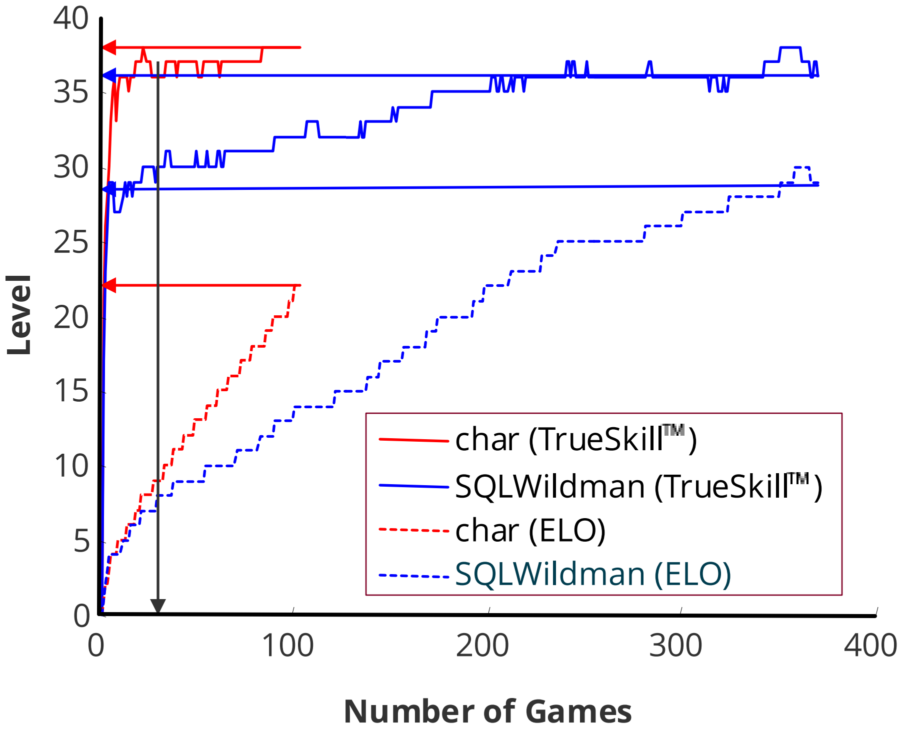
  - **Observation**: This pair $(i, j)$ approximately maximizes the information (entropy!) of the predicted match outcome because it gets closest to 50% winning probability



- **Leaderboard**: Decide how to display the best to worst player
  - **Observation**: There is an asymmetry in making a ranking mistake
    - **Cheap**: Ranking a truly good player lower than they should be (why?)
    - **Expensive**: Ranking a truly bad player higher than they should be (why?)
    - The loss minimizer of this decision process is a **quantile** $\mu - k \cdot \sigma$ with $k > 0$

# Experimental Results

- **Data Set: Halo 2 Beta**
  - 3 game modes
    - Free-for-All
    - Two Teams
    - 1 vs. 1
  - > 60,000 match outcomes
  - ≈ 6,000 players
  - 6 weeks of game play
  - Publically available



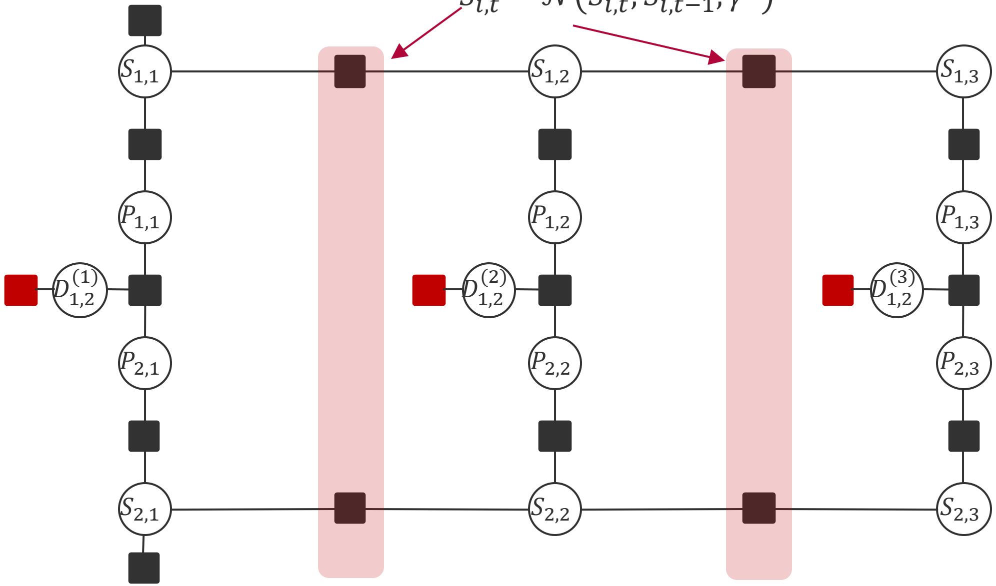

# Overview

1. Ranking Problem
2. Probabilistic Ranking Models
3. TrueSkill: Expectation Propagation on Ranking Factor Graphs
4. **TrueSkill Through Time**

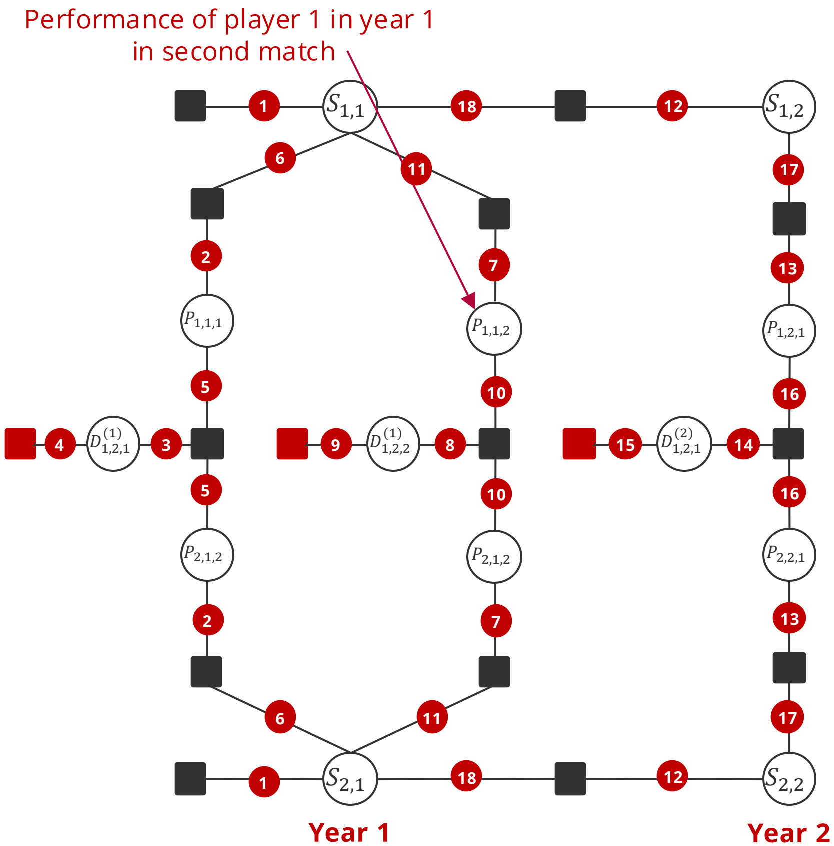
# Skill Dynamics

- **Dynamics**: In reality, skills of players evolve over time and are not stationary
  - **Idea**: Since we do not know which direction the skills evolve, assume that the skill of player $i$ at time $t$ depends on the skill of the same player at time $t-1$ via

$$S_{i,t} \sim \mathcal{N}\left(S_{i,t}; S_{i,t-1}, \gamma^2\right)$$

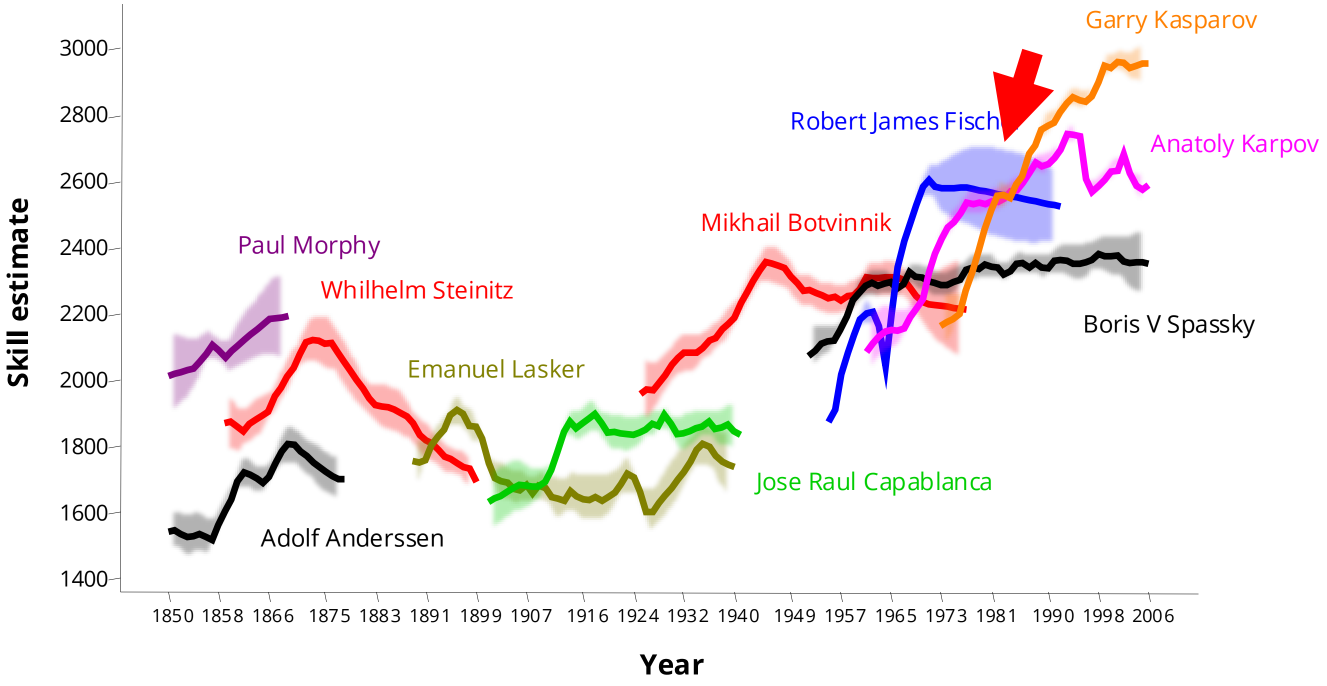


# TrueSkill Through Time: Message Schedule




- Four Phases
  1. **Prior (1)**: Send prior messages to each skill variable for the first year of a player
  2. **Annual Matches (2-11)**: Loop over all (2-player) matches in a year until the skill marginals for all active player in that year does not change (much) anymore
  3. **Forward Dynamics (12)**: Send skill dynamics messages forward in time from $t$ to $t + 1$ and keep running phase 2. (13 – 17).
  4. **Backward Dynamics (18)**: Send skill dynamics messages backward in time from year $t + 1$ to $t$ and keep running step 2. (2-11)
- Stop when no variable in the outer loop changes much anymore.

# TrueSkill-Through-Time: Chess Players




**History of Chess**
3.5M match outcomes
20 million variables
40 million factors

Hasso Plattner Institut
Digital Engineering · Universität Potsdam

# See you next week!