Hasso Plattner Institut
Digital Engineering · Universität Potsdam

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

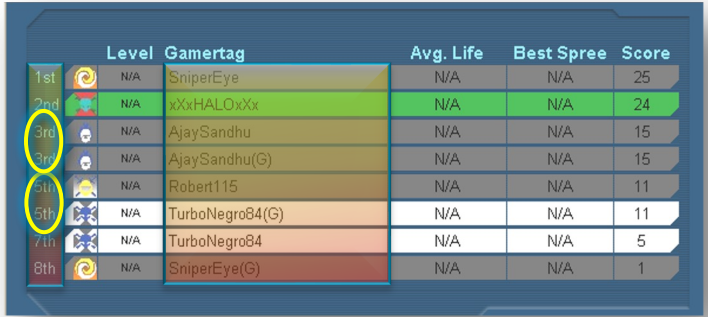
Árpád Imre Élő
(1903 – 1992)

The Skill Rating Problem

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

Team		Score
1st	Red Team	50
2nd	Blue Team	40

	Level	Gamertag	Avg. Life	Best Spree	Score
1st	10	BlueBot	00:00:49	6	15
1st	7	SniperEye	00:00:41	4	14
1st	9	ProThepirate	00:01:07	3	13
1st	10	dazdemon	00:00:59	3	8
2nd	10	WastedHarry	00:00:41	4	17
2nd	3	Ascla	00:00:37	2	10
2nd	9	Antidote4Losing	00:00:41	2	9
2nd	12	Blackknight9	00:00:48	3	4



	Level	Gamertag	Avg. Life	Best Spree	Score
1st	N/A	SniperEye	N/A	N/A	25
2nd	N/A	xXxHALOxXx	N/A	N/A	24
3rd	N/A	AjaySandhu	N/A	N/A	15
3rd	N/A	AjaySandhu(G)	N/A	N/A	15
5th	N/A	Robert115	N/A	N/A	11
5th	N/A	TurboNegro84(G)	N/A	N/A	11
7th	N/A	TurboNegro84	N/A	N/A	5
8th	N/A	SniperEye(G)	N/A	N/A	1

- **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

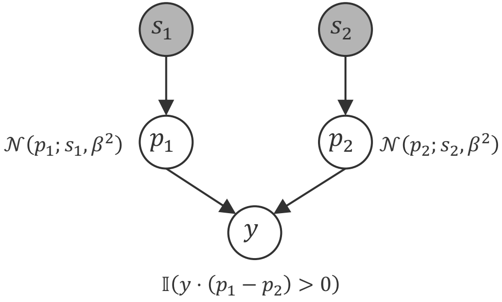
$$P\left(y = 1 | s_i, s_j\right) = \frac{\exp(s_i)}{\exp(s_i) + \exp\left(s_j\right)} = \boxed{\frac{\exp\left(s_i - s_j\right)}{1 + \exp\left(s_i - s_j\right)}}$$

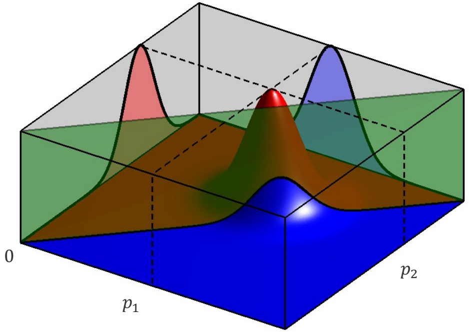
Logistic sigmoid in skill difference

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

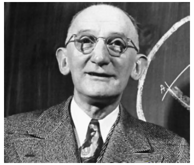
$$P\left(y = 1 | s_i, s_j\right) = \boxed{\int_0^{\infty} \mathcal{N}\left(t; s_i - s_j, 2\beta^2\right) dt}$$

Probit sigmoid in skill difference





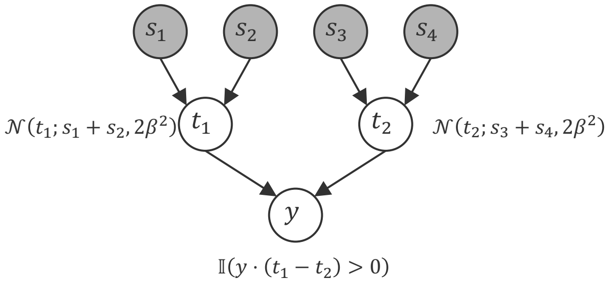
Ralph A. Bradley
(1923 – 2001)



Louis Leon Thurstone
(1887 – 1955)

Two-Team Match Outcome Model

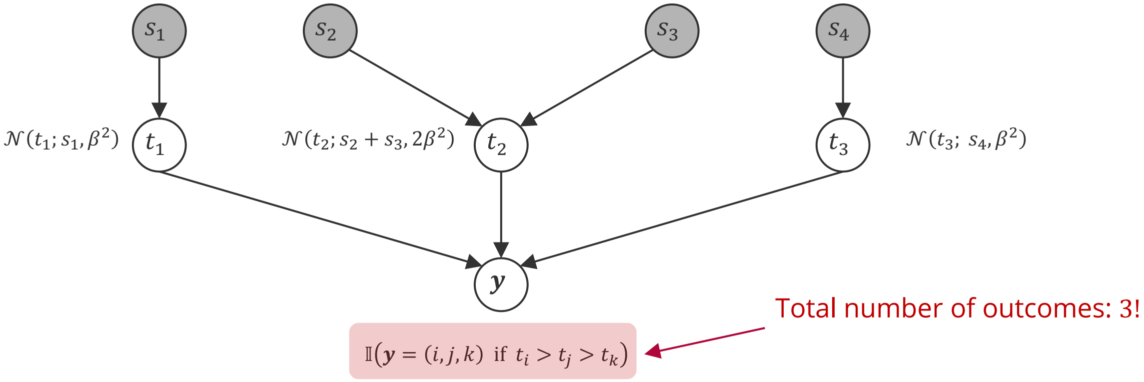
- **Team Assumption**: Performance of a team is the sum of the performances of its players



- **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



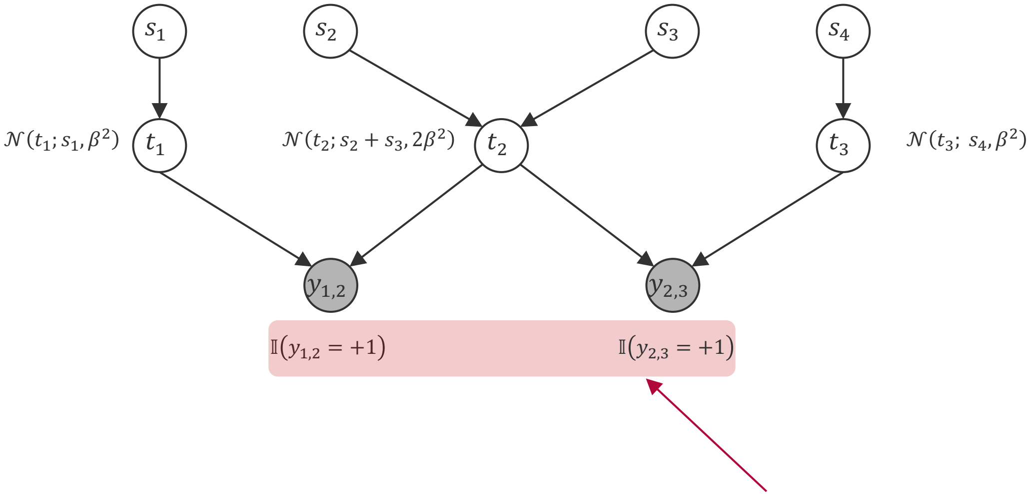
- **Easy** to **sample** for given skills but computationally difficult to ”invert”!

Introduction to Probabilistic Machine Learning

Unit 5 – Bayesian Ranking

From Match Outcomes to Pairwise Rankings

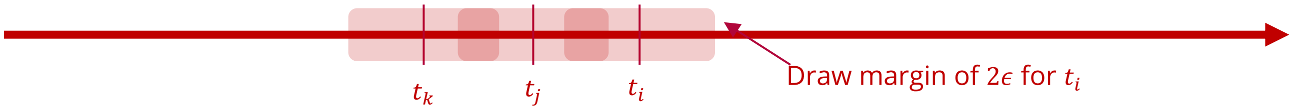
- **Learning**: In the ranking setting, we observe multi-team match outcomes and want to infer the skills of all single players!
- **Idea**: Leverage the transitivity of the real line of latent scores!



By transitivity, this implies $t_1 > t_3$

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 ϵ 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

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

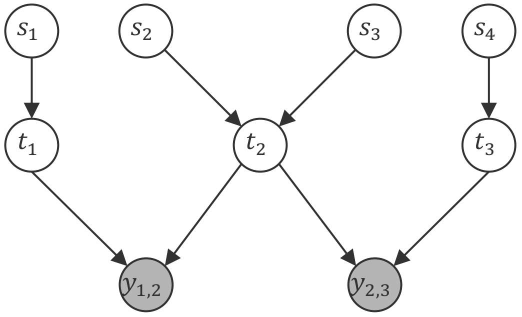
  - This only works if the fraction α_i is truly independent of the (predicted) match outcome!

Overview

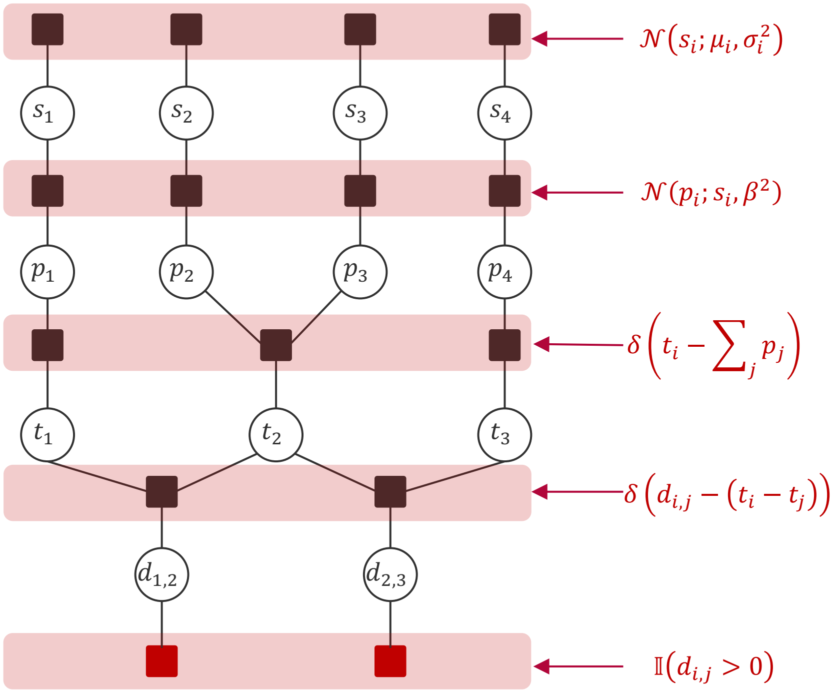
1. Ranking Problem
2. Probabilistic Ranking Models
3. **TrueSkill: Expectation Propagation on Ranking Factor Graphs**
4. TrueSkill Through Time

TrueSkill Factor Graphs

Bayesian Network

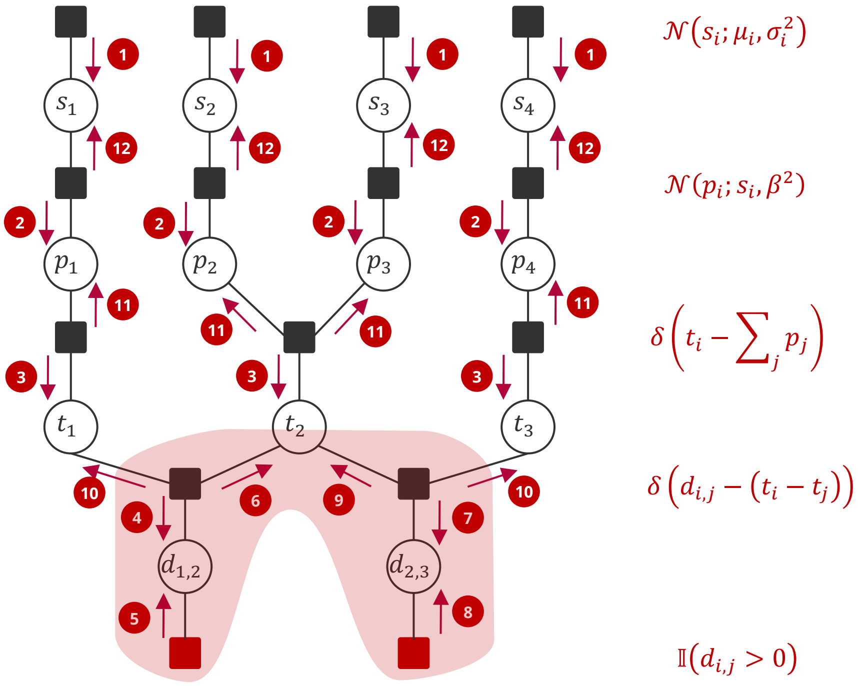


Factor Graph



(Approximate) Message Passing in TrueSkill Factor Graphs

TrueSkill Factor Graph

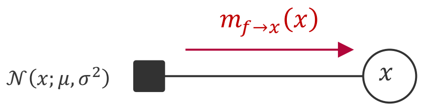


- Four Phases
  1. Pass prior messages (1)
  2. Pass messages *down* to the team performances (2 to 3)
  3. Iterate the approximate messages on the pairwise team differences (4 to 9)
  4. Pass messages back from *up* from team performances to player skill (10 – 12)

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

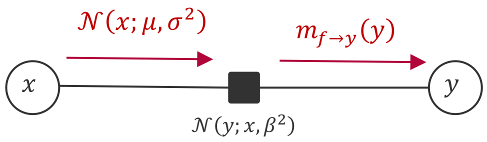
Message Update Equations

Gaussian Factor



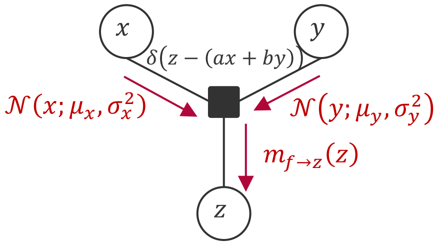
$$m_{f\to x}(x) = \mathcal{N}(x;\mu,\sigma^2)$$

Gaussian Mean Factor



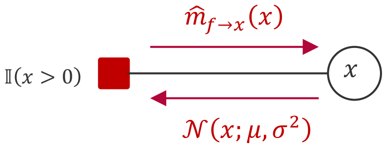
$$m_{f\to y}(y) = \int \mathcal{N}(y;x,\beta^2)\cdot\mathcal{N}(x;\mu,\sigma^2)\,dx = \mathcal{N}(y;\mu,\sigma^2+\beta^2)$$

Weighted Sum Factor



$$m_{f\to z}(z) = \mathcal{N}\left(z; a\mu_x + b\mu_y, a^2\sigma_x^2 + b^2\sigma_y^2\right)$$

Greater-Than Factor



$$\hat{m}_{f\to x}(x) = \frac{\hat{p}(x)}{m_{x\to f}(x)} = \frac{\mathcal{N}(x;\hat{\mu},\hat{\sigma}^2)}{\mathcal{N}(x;\mu,\sigma^2)}$$

Mean and variance of a truncated Gaussian $\mathcal{N}(x;\mu,\sigma^2)$

Truncated Gaussians

- **Truncated Gaussians**. *A truncated Gaussian given by $p(x) \propto \mathbb{I}(x > 0) \cdot \mathcal{N}(x; \mu, \sigma^2)$ has the following three moments*

$$Z(\mu, \sigma^2) = \int_{-\infty}^{+\infty} p(x)\, dx = 1 - F(0; \mu, \sigma^2)$$

Follows from definition of F

$$E[X] = \int_{-\infty}^{+\infty} x \cdot p(x)\, dx = \mu + \sigma \cdot v\left(\frac{\mu}{\sigma}\right)$$

Additive update that goes to zero as $\frac{\mu}{\sigma} \to \infty$

$$\mathrm{var}[X] = \int_{-\infty}^{+\infty} (x - E[X])^2 \cdot p(x)\, dx = \sigma^2 \cdot \left(1 - w\left(\frac{\mu}{\sigma}\right)\right)$$

Multiplicative update that goes to 1 as $\frac{\mu}{\sigma} \to \infty$

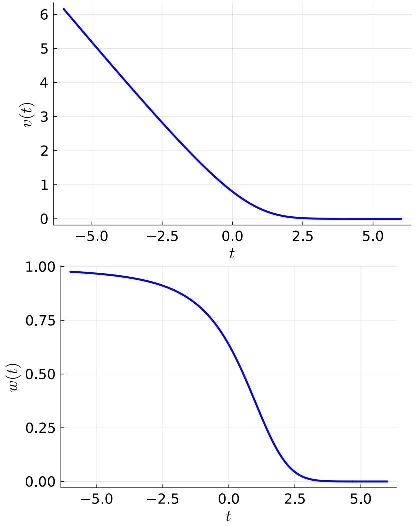
where the probit $F(t; \mu, \sigma^2) := \int_{-\infty}^{t} \mathcal{N}(x; \mu, \sigma^2)\, dx$ and

$$v(t) := \frac{\mathcal{N}(t; 0,1)}{F(t; 0,1)}$$

Converges to $-t$ as $t \to -\infty$

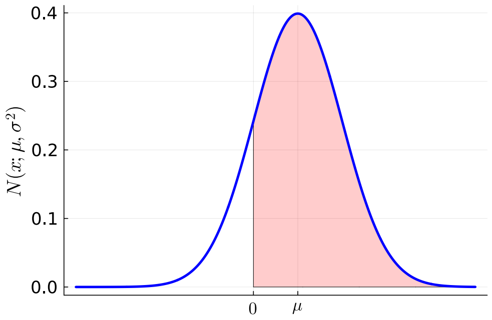
$$w(t) := v(t) \cdot [v(t) + t]$$

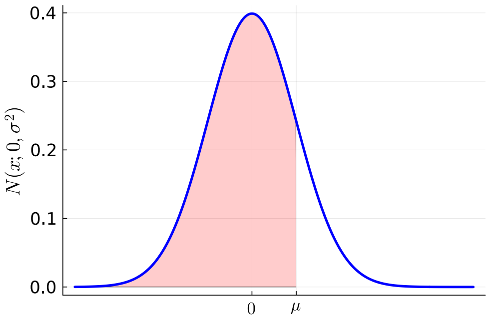
- This can be generalized to an arbitrary interval $[a, b]$ where the Gaussian is truncated!



Derivation of Mean Update Equation

$$Z(\mu,\sigma^2) = \int_0^{+\infty} \mathcal{N}(x;\mu,\sigma^2)\,dx = 1 - F(0;\mu,\sigma^2)$$





Definition of the derivative of integral!

$$\frac{d}{d\mu}Z(\mu,\sigma^2) = \frac{d}{d\mu}\int_0^{+\infty}\left(\frac{x-\mu}{\sigma^2}\right)\cdot\mathcal{N}(x;\mu,\sigma^2)\,dx$$

$$\frac{d}{d\mu}Z(\mu,\sigma^2) = \frac{d}{d\mu}\int_{-\infty}^{\mu}\mathcal{N}(x;0,\sigma^2)\,dx = \mathcal{N}(\mu;0,\sigma^2)$$

$$\frac{1}{\sigma^2}\cdot\left[\int_0^{+\infty} x\cdot\mathcal{N}(x;\mu,\sigma^2)\,dx - \mu\cdot\left(1-F(0;\mu,\sigma^2)\right)\right] = \mathcal{N}(\mu;0,\sigma^2)$$

$$\mathcal{N}\left(\frac{\mu}{\sigma};0,1\right) = \sigma\cdot\mathcal{N}(\mu;0,\sigma^2)$$

Pulling the derivative into the integral and apply chain rule

$$\left[\int_0^{+\infty} x\cdot\frac{\mathcal{N}(x;\mu,\sigma^2)}{\left(1-F(0;\mu,\sigma^2)\right)}\,dx\right] = E[X] = \mu + \sigma\cdot\frac{\mathcal{N}\left(\frac{\mu}{\sigma};0,1\right)}{F\left(\frac{\mu}{\sigma};0,1\right)}$$

$$1 - F(0;\mu,\sigma^2) = F\left(\frac{\mu}{\sigma};0,1\right)$$

Normalization Constant for Outcome Probabilities

- **Skill marginals** are normalized 1D Gaussians by design!
- **Problem**: For the probability of the observed match, we want to compute the total normalization constant

$$Z = \int_{-\infty}^{+\infty} \cdots \int_{-\infty}^{+\infty} \prod_i p(s_i) \cdot \prod_i p(p_i|s_i) \cdot \prod_j p(t_j|\mathbf{p}) \cdot \prod_j p\left(d_{j,j+1}|t_j, t_{j+1}\right) \cdot \prod_j \mathbb{I}\left(d_{j,j+1} > 0\right)\, d\mathbf{s}d\mathbf{p}d\mathbf{t}d\mathbf{d}$$

- Algorithm needs to correct for non-normalization of some factors

$$Z_f \approx \frac{\sum_{\{\mathbf{x}_{\mathrm{ne}(f)}\}} f\left(\mathbf{x}_{\mathrm{ne}(f)}\right) \prod_{j \in \mathrm{ne}(f)} \widehat{m}_{x_j \to f}\left(x_j\right)}{\sum_{\{\mathbf{x}_{\mathrm{ne}(f)}\}} \prod_{j \in \mathrm{ne}(f)} \widehat{m}_{f \to x_j}\left(x_j\right) \cdot \prod_{j \in \mathrm{ne}(f)} \widehat{m}_{x_j \to f}\left(x_j\right)}$$

1. **Initialize**: Reset all marginals of all variables to $\mathcal{G}(0,0)$ and $\log Z = 0$
2. **Re-compute** all marginals and the log-normalization constant
   - For all factors, multiply all messages into marginals and add the normalization constant to $\log Z$
3. **Adjust** the log-normalization constant by the factor normalizations $\sum_f \log\left(Z_f\right)$

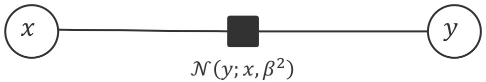
Factor (Log-) Normalization Equations

$$Z_{f_i} \approx \frac{\sum_{\{x_{\mathrm{ne}(f_i)}\}} f_i(\mathbf{x}_{\mathrm{ne}(f_i)}) \prod_{j\in\mathrm{ne}(f_i)} \widehat{m}_{x_j\to f_i}(x_j)}{\sum_{\{x_{\mathrm{ne}(f_i)}\}} \prod_{j\in\mathrm{ne}(f_i)} \widehat{m}_{f_i\to x_j}(x_j) \cdot \prod_{j\in\mathrm{ne}(f_i)} \widehat{m}_{x_j\to f_i}(x_j)}$$

Gaussian Factor

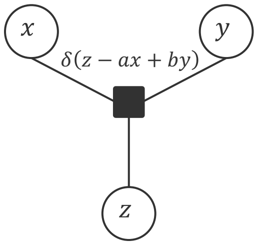
$$\log(Z_f) = 0$$

Gaussian Mean Factor



$$\log(Z_f) = \log\left(\int \frac{p(x)}{m_{f\to x}(x)}\,dx\right)$$

Weighted Sum Factor



$$\log(Z_f) = \log\left(\int \frac{p(x)}{m_{f\to x}(x)}\,dx\right) + \log\left(\int \frac{p(y)}{m_{f\to y}(y)}\,dy\right)$$

Greater-Than Factor

$$\log(Z_f) = \log\left(F\left(\frac{\mu_{x\to f}}{\sigma_{x\to f}}\right)\right) - \log\left(\int m_{x\to f}(x) \cdot m_{f\to x}(x)\,dx\right)$$

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 two players have equal skills

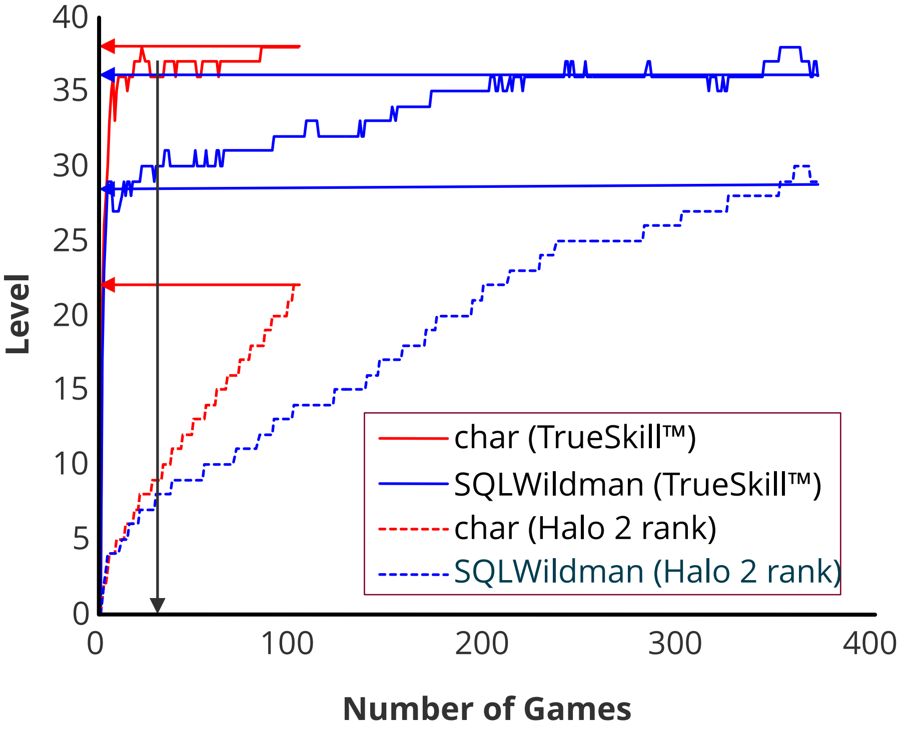
$$\text{Quality}(i,j) = \frac{P\left(p_i \approx p_j | \mu_i - \mu_j, \sigma_i^2 + \sigma_j^2\right)}{P\left(p_i \approx p_j | \mu_i - \mu_j = 0, \sigma_i^2 + \sigma_j^2 = 0\right)}$$

  - **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



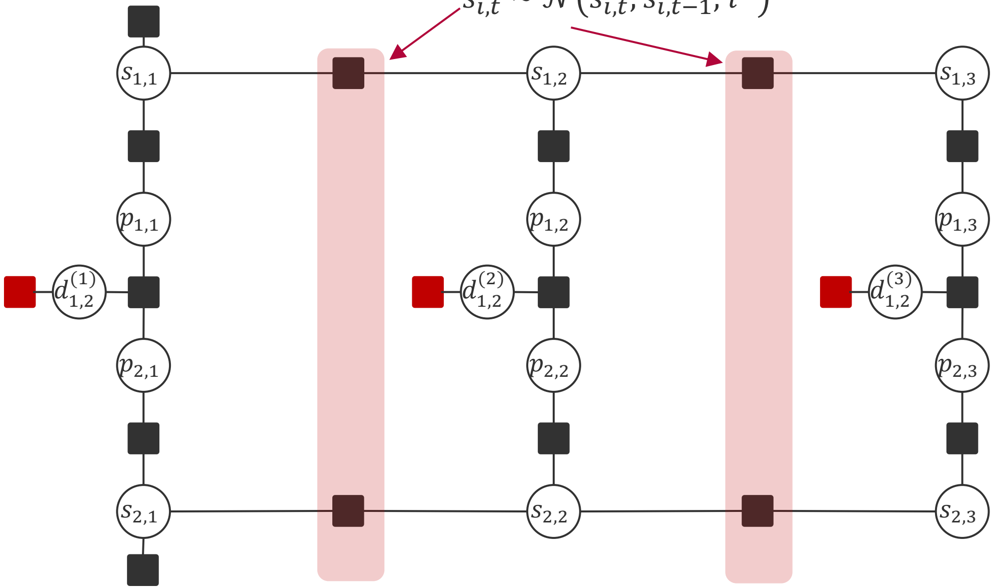
Overview

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

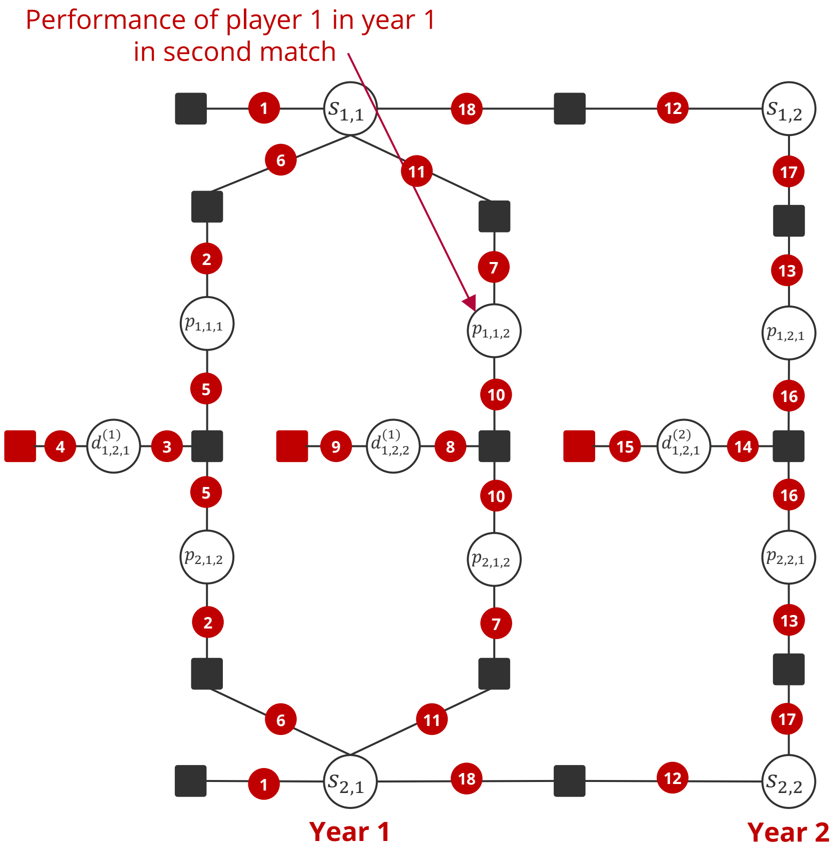
Skill Dynamics

- **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

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

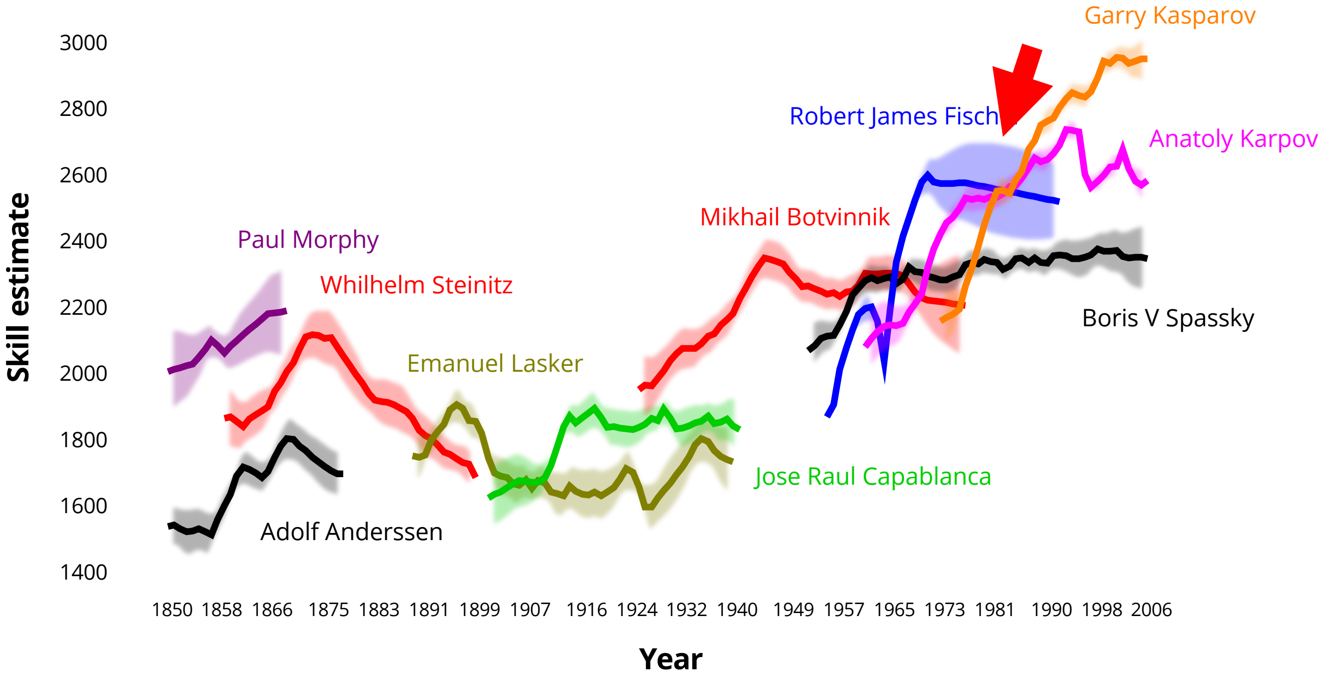


TrueSkill Through Time: Message Schedule



- 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

Introduction to Probabilistic Machine Learning

Unit 5 – Bayesian Ranking

See you next week!