A New Bayesian Rating System for Team Competitions

Sergey Nikolenko SERGEY@LOGIC.PDMI.RAS.RU
Steklov Mathematical Institute, 191023, nab. r. Fontanka, 27, St. Petersburg, Russia,
St. Petersburg Academic University, 194021, ul. Khlopina, 8, korp. 3, St. Petersburg, Russia

Alexander Sirotkin ALEXANDER.SIROTKIN@GMAIL.COM
St. Petersburg Institute for Informatics and Automation RAS, 199178, 14 Line VO, 39, St. Petersburg, Russia.
St. Petersburg Academic University, 194021, ul. Khlopina, 8, korp. 3, St. Petersburg, Russia

Abstract

We present a novel probabilistic rating system for team competitions. Building upon TrueSkill™, we change the factor graph structure to cope with the problems of TrueSkill™, e.g., multiway ties and variable team size. We give detailed inference algorithms for the new structure. Experimental results show a significant improvement over TrueSkill™.

1. Introduction

In the most general setting, probabilistic rating models are Bayesian models where the inference aims to find a linear ordering on a certain set given noisy comparisons of relatively small subsets of this set. There are applications of probabilistic rating models that do not relate to gaming and matches between players and/or teams: in general, a rating model is useful whenever there is no way to compare a large number of entities directly, but only comparisons with partially informative results are available (Zhang et al., 2010; Graepel et al., 2010; Stein et al., 2005; Stern et al., 2006). However, in what follows we will use the metaphor of matches between teams because the terminology fits this field very naturally, and our particular application does relate to ranking the results of tournaments.

Historically, the first probabilistic rating model was the Elo rating developed in the 1960s (Elo, 1978); it could process matches of two players, including draws. A different class of rating models is represented by Bradley–Terry models that, in their simplest form, assume that the win probability of a player or team is proportional to its "true" rating γ_i and then fit parameters γ_i (Stein et al., 2005; Rao & Kupper, 1967; Bradley & Terry, 1952; Hunter, 2004). The latest TrueSkill™ model recently developed in Microsoft Research (Graepel et al., 2007; Dangauthier et al., 2008) is based on Bayesian inference on factor graphs. It allows for matches between teams of several players, with the team roster changing between matches. In this paper, we present an updated version of TrueSkill™.

We have already identified several problems with the TrueSkill™ model that we briefly touch upon in Section 2 (Nikolenko & Sirotkin, 2010). In this paper, we provide, in Section 3, the inference algorithms for an enhanced version of the TrueSkill™ factor graph. We present the algorithms in more detail than the original paper (Graepel et al., 2007); we hope the algorithms in this paper are detailed enough to be easy to implement in software. Section 4 gives experimental results that show a significant improvement over TrueSkill™.

2. The TrueSkill™ rating system

The TrueSkill™ rating system developed by Herbrich, Minka, and Graepel (2007) performs Bayesian inference on a factor graph of special form. A sample factor graph is shown on Fig. 1; there, four teams have finished a match. Team 1 of two players won, teams 2 and 3, each of one player, drew in second place, and team 4, of two players, placed last.

In TrueSkill™, the problem is to compute posterior ratings given prior ratings and the results of a match given as a list of places (possibly equal in case of ties). We assume normal distributions on each layer of the factor graph, and these layers are as follows:

- $s_{i,j}$ represents the skill of player i from team j; it is normally distributed around $\mu_{i,j}$ with variance

Appearing in *Proceedings of the 28th International Conference on Machine Learning*, Bellevue, WA, USA, 2011.

$\sigma_{i,j}$, where $(\mu_{i,j}, \sigma_{i,j})$ are prior ratings;

- $p_{i,j}$ represents the performance of player i from team j in this match; it is conditionally normally distributed around the skill $s_{i,j}$ with variance β, where β is a model parameter;
- t_j represents the performance of team j; in TrueSkill™, team performance is the sum of player performances;
- d_j represents the difference in performance between teams who took neighboring places in the tournament; a tie between two teams corresponds to $|d_j| \le \varepsilon$ for some model parameter ε, and a win corresponds to $d_j > \varepsilon$.

In these variables, the general Bayesian inference problem looks like

$$p(\boldsymbol{s} \mid \pi) = \int\int\int p(\pi, \boldsymbol{d}, \boldsymbol{t}, \boldsymbol{p}, \boldsymbol{s}) d\boldsymbol{d} d\boldsymbol{t} d\boldsymbol{p} =$$
$$= \int\int\int p(\pi \mid \boldsymbol{d}) p(\boldsymbol{d} \mid \boldsymbol{t}) p(\boldsymbol{t} \mid \boldsymbol{p}) p(\boldsymbol{p} \mid \boldsymbol{s}) d\boldsymbol{d} d\boldsymbol{t} d\boldsymbol{p},$$

where π denotes match results, and boldface letters denote the corresponding vectors.

A probabilistic inference algorithm for the TrueSkill™ model is given in (Graepel et al., 2007); it is based on the ideas of Minka (2001a; 2001b) that extend general message passing algorithms (MacKay, 2003). Messages are passed top-down (from player performances to differences between teams) and then back, each node passing a message to its neighbor whenever it has received messages from all its other neighbors. After passing every edge in both direction, updated distributions at the top nodes correspond to posterior distributions of the players' ratings. Message passing through the top layers of the factor graph is straightforward: since all distributions are normal, it is easy to compute the parameters of any weighted sum of them. On the other hand, the bottom layer requires an iterative message passing algorithm to approximate the posterior distributions of t_j. The problem is that after processing a node with the $\mathbb{I}(d_j > \varepsilon)$ or $\mathbb{I}(|d_j| \le \varepsilon)$ function, the message distribution ceases to be normal. TrueSkill™ solves this problem by approximating the result with a normal distribution and repeating the message passing iterations until convergence.

However, there are a few issues that the TrueSkill™ rating system fails to address properly. Our research concentrates mainly on two problems.

1. *Multiway ties.* In TrueSkill™, ties are treated by the $\mathbb{I}(|d_{i+1} - d_i| \le \varepsilon)$ node, and multiway ties are represented as chains of such nodes. This may lead to incorrect behaviour: if $|d_{i+1} - d_i| \le \varepsilon$ and $|d_{i+2} - d_{i+1}| \le \varepsilon$, it means that $|d_{i+2} - d_i| \le 2\varepsilon$, and the effect accumulates. Since victories between neighboring teams are represented as $|d_{i+1} - d_i| > \varepsilon$, it may even happen that the maximum likelihood value of a higher-placed team is smaller than for a lower-placed team. In the example on Fig. 1, the maximum likelihood value of team 4 performance t_4 may be arbitrarily close to that value of t_2, and if there was one more team tied with t_2 and t_3, it might exceed t_2. This situation is actually very likely to occur if the winning team was an underdog, and its prior distribution fell behind the prior distribution of team 4; we have seen this effect on our dataset. We address this problem with new factor graph structure (see Section 3).

2. *Variable team size.* Another feature of TrueSkill™ that we have not found to work well is the assumption that a team's performance is the sum of player performances. In many competitions and many comparison problems, an undersized team stands a very good chance against a full one, and measuring its performance as a sum of player performances gives the players of the smaller team (if they win, of course) a relatively unfair rating boost.

We propose to alleviate the latter problem by selecting a different function for the team performance as a function of player performances. It is very easy to replace the sum with any affine function. Moreover, we propose a simple way to approximate nonlinear functions: one can replace player performances with their estimates provided by the prior ratings μ_i. For instance, to approximate a team performance function $t = p_1^2 + p_2^2 + \ldots + p_n^2$ we replace it with $t = \mu_1 p_1 + \mu_2 p_2 + \ldots + \mu_n p_n$. The coefficients in this affine approximation can be given to the rating system immediately before processing a tournament, so they are allowed to depend on current player ratings.

3. The new factor graph and inference algorithm

To cope with the problem of large multiway ties, we add another layer to the TrueSkill™ factor graph. The extra layer unites tied teams into a single node that will participate in the approximate inference iterations on its own. The new variables l_k correspond to "average" performances for those subsets of teams that have shared a single place. If teams $t_1, \ldots, t_s$ have tied, we introduce a variable l which is related to t_i as $|l - t_i| \le \varepsilon$. Thus, on the bottom level all d_k's correspond to different places. In order to separate the places l_k, we require that $l_{k+1} - l_k = d_k > 2\varepsilon$.

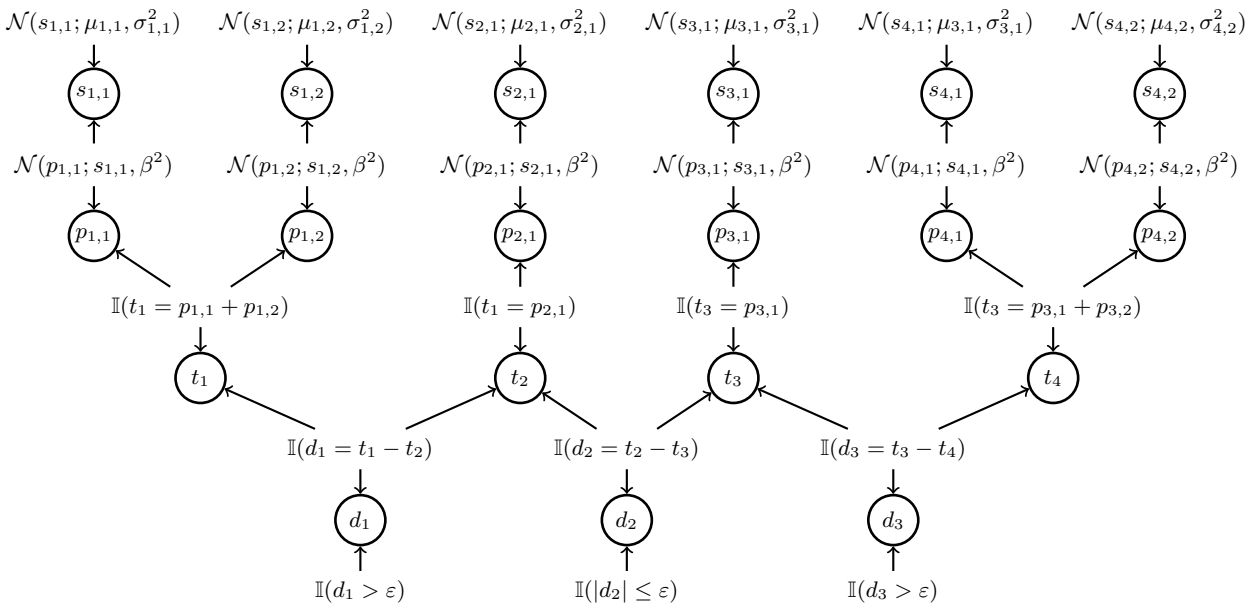

Figure 1. A sample TrueSkill™ factor graph: four teams, teams 1 and 4 have two players each; teams 2 and 3, one player each. Team 1 won, teams 2 and 3 drew behind it, and team 4 placed last.

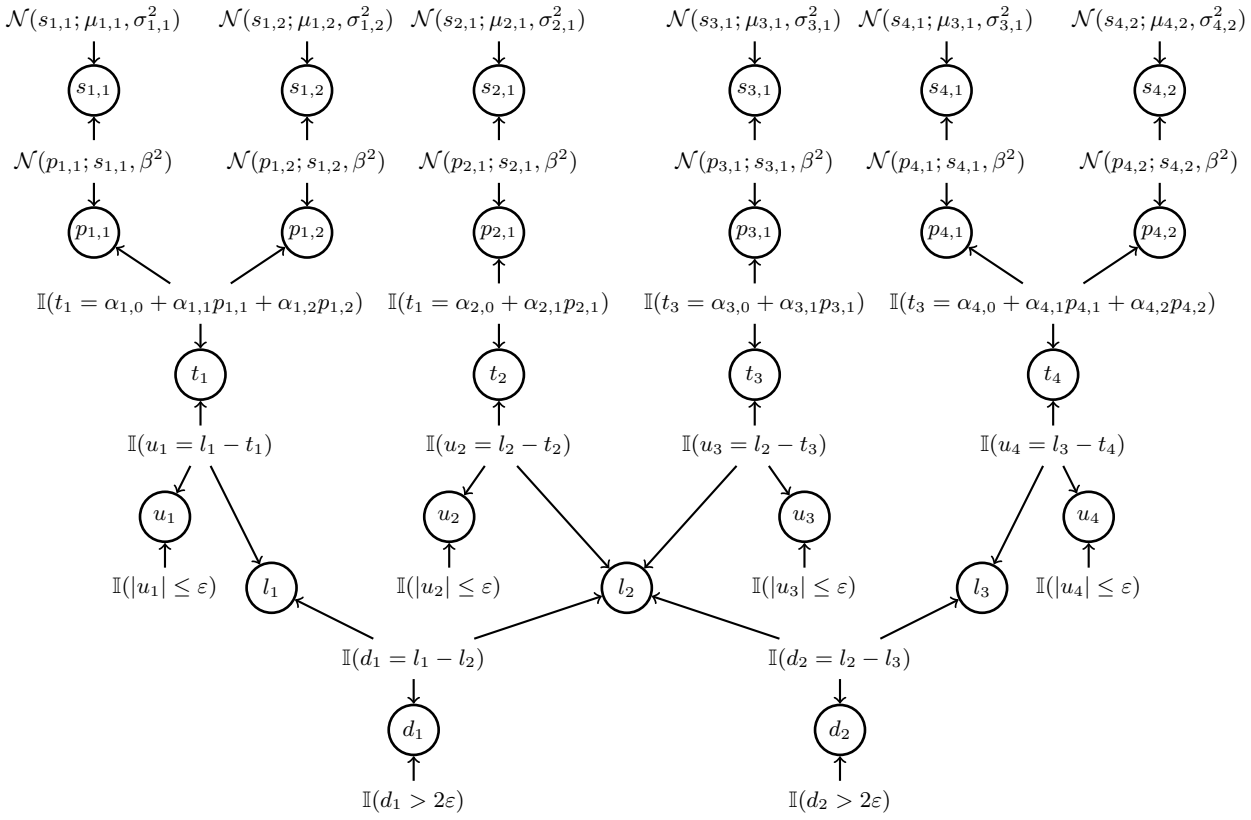

Figure 2. A new factor graph corresponding to the case shown on Figure 1.

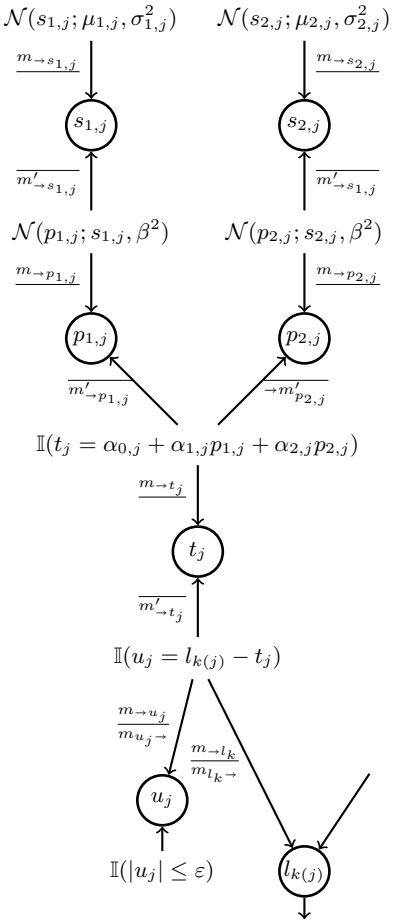

Figure 3. Messages in algorithms 1 and 3 for a team of two players. Messages m correspond to Algorithm 1; m', to Algorithm 3.

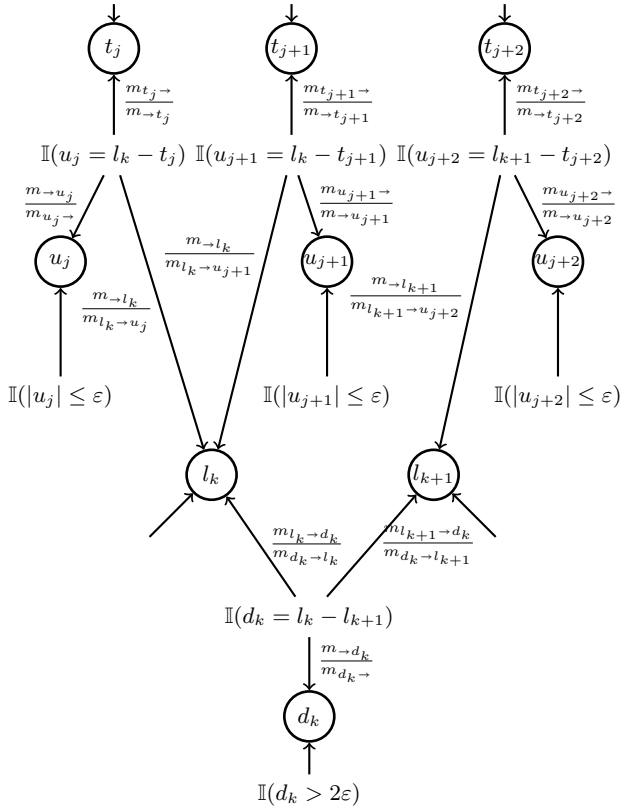

Figure 4. Messages in algorithm 2.

A sample resulting factor graph is shown on Fig. 2. For the team performance function, we use an abstract affine function with coefficients $\alpha_{i,j}$.

Inference in our model is very similar to TrueSkill™. As in TrueSkill™, we assume the players' performance distribution to be normal. We need the following operations on normal distributions: addition, multiplication by a constant, multiplication of two normal distributions, division of two normal distributions (all operations are assumed to include renormalization). These operations suffice for all upper layers of the factor graph. We give formulas for approximate computations in two cases, a draw and a victory.

We characterize a normal distribution with its mean μ and variance σ. Addition and multiplication by a constant change the parameters as follows:

$$\begin{aligned} \mu_{sum} &= \mu_1 + \mu_2, & \sigma_{sum} &= \sqrt{{\sigma_2}^2 + {\sigma_1}^2}, \\ \mu_{\cdot\alpha} &= \alpha\mu, & \sigma_{\cdot\alpha} &= |\alpha|\sigma. \end{aligned}$$

In the algorithm, we often need to calculate products of messages, i.e., products of probability densities coming from different nodes. The messages in our case contain a normal distribution; formulas for the product and ratio of two normal densities, we get

$$\begin{aligned} \mu_{prod} &= \tfrac{\mu_1{\sigma_2}^2 + \mu_2{\sigma_1}^2}{{\sigma_2}^2 + {\sigma_1}^2}, & \sigma_{prod} &= \tfrac{\sigma_2\sigma_1}{\sqrt{\sigma_1^2 + \sigma_2^2}}, \\ \mu_{div} &= \tfrac{\mu_1\sigma_2^2 - \mu_2\sigma_1^2}{\sigma_2^2 - \sigma_1^2}, & \sigma_{div} &= \tfrac{\sigma_2\sigma_1}{\sqrt{|\sigma_2^2 - \sigma_1^2|}}. \end{aligned} \tag{1}$$

Finally, we give formulas for the approximate inference algorithm (see Appendix for the derivation). There are two cases; in one, we have to "bounce back" from a node x with the $\mathbb{I}(x > \varepsilon)$ function attached to it; in the other, from a node with the $\mathbb{I}(|x| \le \varepsilon)$ function attached. All other situations in the factor can be computed exactly. Consider the original message $m = (\mu, \sigma)$; approximating the result with a normal distribution, we get

$$\mathbb{I}_{>\varepsilon}(m) = \frac{p_>}{m}, \quad \mathbb{I}_{\le\varepsilon}(m) = \frac{p_\le}{m}, \tag{2}$$

where

$$\begin{aligned}
\mu_{>} &= \mu + \frac{\sigma e^{-\frac{(\mu-\varepsilon)^2}{2\sigma^2}}}{\alpha_{>}\sqrt{2\pi}},\\
\sigma^2_{>} &= \mu^2 + \sigma^2 + \frac{\sigma(\varepsilon+\mu)e^{-\frac{(-\varepsilon+\mu)^2}{2\sigma^2}}}{\alpha_{>}\sqrt{2\pi}} - \mu^2_{>},\\
\alpha_{>} &= \tfrac{1}{2} - \tfrac{1}{2}\mathrm{erf}\left(\frac{(\varepsilon-\mu)}{\sqrt{2}\sigma}\right),\\
\mu_{\leq} &= \mu + \frac{\sigma(e^{-\frac{(\epsilon-\mu)^2}{2\sigma^2}} + e^{-\frac{(\epsilon+\mu)^2}{2\sigma^2}})}{\alpha_{\leq}\sqrt{2\pi}},\\
\sigma^2_{\leq} &= \mu^2 + \sigma^2 - \sigma\frac{(\varepsilon+\mu)e^{-\frac{(-\varepsilon+\mu)^2}{2\sigma^2}} + (\varepsilon-\mu)e^{-\frac{(\varepsilon+\mu)^2}{2\sigma^2}}}{\sqrt{2\pi}\alpha_{\leq}} - \mu^2_{\leq},\\
\alpha_{\leq} &= \tfrac{1}{2}\mathrm{erf}\left(\frac{(\epsilon+\mu)}{\sqrt{2}\sigma}\right) - \tfrac{1}{2}\mathrm{erf}\left(\frac{(-\epsilon+\mu)}{\sqrt{2}\sigma}\right),
\end{aligned} \tag{3}$$

and division is done according to (1).

The entire inference algorithm is shown in Algorithms 1, 2, and 3 (we have broken it up into three parts for readability). In all algorithms, m is the number of teams; n_j, the number of players in team j; r, the number of different places (i.e., the number of vertices l_k); $T(k)$, the set of indices of teams who shared the k^{th} place (i.e., the set of t_j's that are immediately connected to l_k in the factor graph); $k(j)$, the place team j has occupied. Messages are denoted by $m_{x\to y}$; each message is identified with a pair of normal distribution parameters $(\mu_{x\to y}, \sigma_{x\to y})$, and this notation is used interchangeably. In operations on distributions, we implicitly assume that the result may have be renormalized; however, formulas (1)-(3) are normalized already. We illustrate all messages occurring in the algorithms on Figures 3 and 4.

Algorithm 1 computes the prior distributions on team performances and the initial approximation to $m_{\to l_k}$; it is illustrated on Fig. 3 by messages m. Algorithm 2 is the main cycle of approximate message passing; the messages are shown on Fig. 4. Finally, Algorithm 3 propagates the results of approximate message passing and outputs updated ratings for every player; it is shown on Fig. 3 with messages m'.

Algorithm 1 Initialization.

Input: $(\mu_{i,j}, \sigma_{i,j})$, $1 \le j \le m$, $1 \le i \le n_j$.
for $j = 1$ **to** m **do**
  for $i = 1$ **to** n_j **do**
    $m_{\to s_{i,j}} := (\mu_{i,j}, \sigma_{i,j})$
    $m_{\to p_{i,j}} := (\mu_{i,j}, \sqrt{\sigma^2_{i,j} + \beta^2})$
  end for
  $m_{\to t_j} := \alpha_{0,j} + \sum_{i=1}^{n_j} \alpha_{i,j} m_{\to p_{i,j}}$
  $m_{\to u_j} := (0, \sigma_{\to t_j})$
  $m_{u_j \to} := \mathbb{I}_{\leq\varepsilon}(m_{\to u_j})$
  $m_{t_j \to} := m_{u_j \to} m_{\to t_j}$
end for
for $k = 1$ **to** r **do**
  $m_{\to l_k} := \prod_{j \in T(k)} (m_{t_j} - m_{u_j \to})$
end for
return $\{m_{\to l_k}\}_{k=1}^{r}$, $\{m_{\to p_{i,j(i)}}, m_{\to s_{i,j(i)}}\}_{i=1}^{n}$.

4. Experimental results

We have tested our approach and compared it with TrueSkill™ on a database of tournaments from an intellectual game in which teams of (usually) six players compete in solving problems. In fact, this work has resulted from trying to apply TrueSkill™ to this rating problem. The dataset consists of 680 tournaments featuring about 50000 players. Most of the players participated in very few tournaments, but there was a core subset of players who played many tournaments across a wide variety of teams. The teams in the dataset's tournaments are limited to six players, but smaller teams are very common. An interesting challenge was provided by teams that listed more than six players in the roster; this meant that a team had had substitutions during the tournament. In these cases, we normalize the team performance function so that the total team performance approximately matches the performance of a team consisting of top six players of the roster, but keep each player with a nonzero coefficient in the sum (otherwise, his or her rating will not change after the tournament).

Some teams have had a very stable roster playing many tournaments together; however, in general players often switch teams, and most teams are hard to define in terms of a roster. A tournament usually has a fixed set of possible different positions (possible results), about 30-60 positions in total (which means that the teams had to solve 30-90 problems and were ranked by the number of correctly solved problems, ties not broken). However, the number of teams in a tournament could be very large, exceeding 1000 in some cases. Therefore, multiway ties, sometimes spanning across dozens of places, were common.

We have compared the predictive power of two versions of TrueSkill™ and three versions of our approach.

1. TSa is basic TrueSkill™: the team performance is computed as the sum of player performances.
2. TSb is TSa, but the team performance is computed as the average of player performances.
3. In TS2a, team performance is the sum of player performances and variance is variable; only the factor graph is different from TrueSkill™.
4. TS2b is TS2a with a specifically tuned function of team performance.

Algorithm 2 Approximate inference.

Input: $\{m_{\to l_k}\}_{k=1}^{r}$.
for $k = 1$ **to** $r-1$ **do**
  $m_{l_k \to d_k} := m_{\to l_k}$
  $m_{l_{k+1} \to d_k} := m_{\to l_{k+1}}$
end for
repeat
  for $k = 1$ **to** $r-1$ **do**
    $m_{\to d_k} := m_{\to l_k} - m_{\to l_{k+1}}$
    $m_{d_k \to} := \mathbb{I}_{>2\varepsilon}(m_{\to d_k})$
    $m_{d_k \to l_k} := m_{d_k \to} + m_{l_{k+1} \to d_k}$
    $m_{d_k \to l_{k+1}} := m_{l_k \to d_k} - m_{d_k \to}$
  end for
  for $k = 1$ **to** r **do**
    for $j \in T(k)$ **do**
      $m_{l_k \to u_j} := m_{d_{k-1} \to l_k} m_{d_k \to l_k} \prod_{i \in T(k) \setminus j} m_{u_i \to l_k}$
      $m_{\to u_j} := m_{\to l_k} - m_{t_j \to}$
      $m_{u_j \to} := \mathbb{I}_{\leq \varepsilon}(m_{\to u_j})$
      $m_{u_j \to l_k} := m_{t_j \to} - m_{u_j \to}$
    end for
    $m_{l_k \to} := \prod_{j \in T(k)} m_{u_j \to l_k}$
  end for
  $m_{l_1 \to d_1} := m_{l_1 \to}$
  for $k = 2$ **to** $r-1$ **do**
    $m_{l_k \to d_{k-1}} := m_{l_k \to} m_{d_k \to l_k}$
    $m_{l_k \to d_k} := m_{l_k \to} m_{d_{k-1} \to l_k}$
  end for
  $m_{l_r \to d_{r-1}} := m_{l_r \to}$
until convergence
for $j = 1$ **to** m **do**
  $m_{\to t_j} := m_{l_{k(j)} \to u_j} - m_{u_j \to}$.
end for
return $\{m_{\to t_j}\}_{j=1}^{m}$.

5. TS2c is TS2b with fixed variance $\sigma = 300$.

In TS2b and TS2c, we compute team performance as the average performance of its players with penalties for missing players (2% per missing player). If the roster exceeds 6 (max number of players in the team at any moment, but substitutions are allowed, so rosters can get up to 9), we normalize team performance to the average of (a priori) best six players but leave all players in the mix (thus, t is the sum of $p_{i,j}$'s with identical weights of (avg of top 6 $\mu_{i,j}$'s)/(sum of all $\mu_{i,j}$'s)). Assuming $m_{i,j}$'s are sorted, the team skill is given by

$$t_i = \begin{cases} \frac{\sum_{j=1}^{n_i} p_{i,j}}{n_i} \cdot (0.88 + 0.02 n_i), & \text{if } n_i \leq 6, \\ \sum_{j=1}^{n_i} p_{i,j} \cdot \frac{\sum_{j=1}^{6} \mu_{i,j}}{6 \sum_{j=1}^{n_i} \mu_{i,j}}, & \text{if } n_i > 6. \end{cases} \quad (4)$$

Algorithm 3 Propagating the results.

Input: $\{m'_{\to t_j}\}_{j=1}^{m}$, $\{m_{\to p_{i,j(i)}}, m_{\to s_{i,j(i)}}\}_{i=1}^{n}$.
for $j = 1$ **to** m **do**
  for $i = 1$ **to** n_j **do**
    $m'_{\to p_{i,j}} := \frac{1}{\alpha_{i,j}} \left[m'_{\to t_j} - \alpha_{0,j} - \sum_{i' \neq i} \alpha_{i',j} m_{\to p_{i',j}} \right]$
    $m'_{\to s_{i,j}} := \mu_{\to p_{i,j}}, \sqrt{\sigma^2_{\to p_{i,j}} + \beta^2}$
    $m_{i,j} := m'_{\to s_{i,j}} m_{\to s_{i,j}}$
  end for
end for
return $\{m_{i,j}\}_{i,j}$.

Table 1. Comparing predictive power: an entry a_{ij} shows the number of tournaments on which model i has had better predictions than model j. Numbers do not add up to 300 due to tournaments with exactly similar predictions.

	TSa	TSb	TS2a	TS2b	TS2c
TSa	0	57	15	9	8
TSb	237	0	9	3	2
TS2a	281	289	0	57	56
TS2b	288	294	218	0	100
TS2c	289	297	229	178	0

where n_i is number of players in team i.

Players start out with $\mu = 3000$ and $\sigma = 1000$ each (except in TS2c, where σ is fixed to be 300). We have fixed player variability $\beta = 400$. We input tournament results to rating systems in chronological order and attempt to predict team performances for the next tournament based on all previous tournaments.

A natural metric for comparing rating systems is to count the pairs of teams whose relative standings a rating system has predicted correctly. Figure 5 shows how the error rate of the four rating systems changed over time. The graph depicts the average error rate over a sliding window of 50 tournaments as a function of the total number of tournaments processed; it is a measure of both prediction quality and learning speed. Table 1 lists the pairwise comparisons of all systems: an entry a_{ij} in this table shows in how many of the last 300 tournaments (the first 380 are treated as training data) the percentage of correctly predicted pairs is higher for rating system i than for j. Table 2 shows average pair prediction percentages across all tournaments and for the last 300, 150, and 50 tournaments. In general, TS2c shows the best results among competitors, but all three versions of TS2 far outperform TrueSkill™.

Table 2. The percentage of incorrectly predicted pairs of teams.

	All	Last 300	Last 150	Last 50
TSa	40.7	39.5	39.8	39.2
TSb	44.2	32.6	29.5	29.2
TS2a	28.1	22.9	21.2	20.4
TS2b	25.2	21.2	20.1	19.3
TS2c	25.6	20.6	19.7	18.7

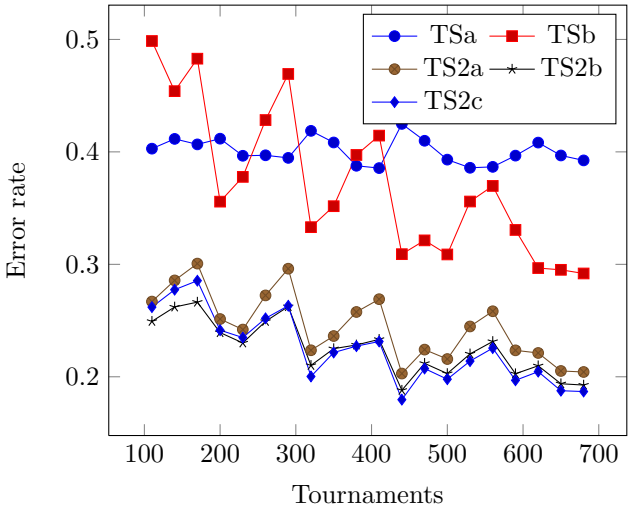

Figure 5. Average error rate over the sliding window of 50 tournaments.

5. Conclusion

In this paper, we have described a modification of the TrueSkill™ probabilistic rating system that addresses a major problem with TrueSkill™: incorrect handling of large multiway ties. We have presented a modified construction of the factor graph that takes care of this problem and inference algorithms for this modified construction. We have also given experimental results that show that our model outperforms TrueSkill™ when multiway ties are common.

Another conclusion is that we strongly encourage the users of TrueSkill™ to tune their team performance functions because the sum of player performances may be a very poor approximation to the team performance. We do not give our final version of the team performance function because it clearly is specific to our particular application; however, we do advise to make the extra effort of tuning it. Further work in this direction may include automated procedures for tuning the constants ε, β, and σ (for fixed variance versions).

Acknowledgements

Both authors acknowledge the support of the Russian Ministry of Education and Science through the Leading Scientists Program (Decree 220). Work of the first author has also been supported by the Russian Presidential Grant Programme for Young Ph.D.'s, grant no. MK-4089.2010.1, for Leading Scientific Schools, grant no. NSh-5282.2010.1, Federal Target Programme "Scientific and scientific-pedagogical personnel of the innovative Russia", and the Russian Fund for Basic Research. The second author also acknowledges the Russian Fund for Basic Research, grant no. 09-01-00861-a.

References

Bradley, R. A. and Terry, M. E. Rank analysis of incomplete block designs. I. The method of paired comparisons. *Biometrika*, 39:324–245, 1952.

Dangauthier, P., Graepel, T., Minka, T., and Herbrich, R. TrueSkill through time: Revisiting the history of chess. In Platt, J.C., Koller, D., Singer, Y., and Roweis, S. (eds.), *Advances in Neural Information Processing Systems 20*, pp. 337–344, Cambridge, MA, 2008. MIT Press.

Elo, A. *The Ratings of Chess Players: Past and Present*. New York: Arco, 1978.

Graepel, T., Minka, T., and Herbrich, R. TrueSkill(tm): A Bayesian skill rating system. In Schölkopf, B., Platt, J., and Hoffman, T. (eds.), *Advances in Neural Information Processing Systems 19*, pp. 569–576, Cambridge, MA, 2007. MIT Press.

Graepel, T., Candela, J. Q., Borchert, T., and Herbrich, R. Web-scale Bayesian click-through rate prediction for sponsored search advertising in Microsofts Bing search engine. In *Proceedings of the* 27^{th} *International Conference on Machine Learning*, pp. 13–20, 2010.

Hunter, David R. MM algorithms for generalized Bradley-Terry models. *The Annals of Statistics*, 32 (1):384–406, 2004.

MacKay, D. J. *Information Theory, Inference and Learning Algorithms*. Cambridge University Press, 2003.

Minka, T. Expectation propagation for approximate Bayesian inference. In Breese, Jack S. and Koller, Daphne (eds.), *Proceedings of the* 17^{th} *Conference on Uncertainty in Artificial Intelligence*, pp. 362–369, San Francisco, CA, 2001a. Morgan Kaufmann Publishers Inc.

Minka, T. *A family of algorithms for approximate Bayesian inference*. PhD thesis, Massachusetts Institute of Technology, 2001b.

Nikolenko, S. I. and Sirotkin, A. V. Extensions of the TrueSkill$^{\text{TM}}$ rating system. In *Proceedings of the* 9^{th} *International Conference on Applications of Fuzzy Systems and Soft Computing*, pp. 151–160, 2010.

Rao, P. V. and Kupper, L. L. Ties in paired–comparison experiments: a generalization of the Bradley–Terry model. *Journal of the American Statistical Association*, 62:194–204, 1967.

Stein, A., Aryal, J., and Gort, G. Generalized Bradley–Terry models and multi-class probability estimates. *IEEE Transactions on Geoscience and Remote Sensing*, 43:852–856, 2005.

Stern, D., Herbrich, R., and Graepel, T. Bayesian pattern ranking for move prediction in the game of Go. In *Proceedings of the* 23^{rd} *International Conference on Machine Learning*, pp. 873–880, 2006.

Zhang, X., Graepel, T., and Herbrich, R. Bayesian online learning for multi-label and multi-variate performance measures. In *Proceedings of the Thirteenth Conference on Artificial Intelligence and Statistics AISTATS 2010*, volume 9, pp. 956–963, 2010.

Appendix

In the Appendix, we derive formulas (2).

Suppose that a node connected to either a $\mathbb{I}(> \varepsilon)$ or $\mathbb{I}(\leq \varepsilon)$ function in the factor graph receives a message; we assume that the message is a normal distribution with parameters μ and σ. Let us try to approximate the distribution corresponding to $m_{>\varepsilon}$ (the case when this node corresponds to a victory). To do that, we compute $p_> = mm_{>\varepsilon}$. The normalization constant is

$$\alpha_> = \int_{-\infty}^{+\infty} mm_{>\varepsilon}\mathrm{d}x = \int_{\varepsilon}^{+\infty} m\mathrm{d}x = \frac{1}{2} - \frac{1}{2}\mathrm{erf}\left(\frac{(\varepsilon-\mu)}{\sqrt{2}\sigma}\right).$$

Therefore, we can compute the mean

$$E(p) = \int_{-\infty}^{+\infty} \frac{xmm_{>\varepsilon}}{\alpha_>}\mathrm{d}x = \int_{\varepsilon}^{+\infty} \frac{xm}{\alpha_>}dx = \mu + \frac{\sigma e^{-\frac{(\mu-\varepsilon)^2}{2\sigma^2}}}{\alpha_>\sqrt{2\pi}}.$$

and the second moment

$$D(p) = \int_{-\infty}^{+\infty} \frac{x^2mm_{>\varepsilon}}{\alpha_>}\mathrm{d}x = \int_{\varepsilon}^{+\infty} \frac{x^2m}{\alpha_>}\mathrm{d}x = \mu^2 + \sigma^2 + \frac{\sigma(\varepsilon+\mu)e^{-\frac{(-\varepsilon+\mu)^2}{2\sigma^2}}}{\alpha_>\sqrt{2\pi}}.$$

Therefore,

$$\mu_> = E(p), \quad \sigma_> = D(p) - (E(p))^2.$$

Having obtained $p_> = mm_{>\varepsilon}$, we can now find $m_{>\varepsilon}$ as $p_>/m$ since we know how to compute the ratio of two normal distributions. We denote the resulting normal distribution as $\mathbb{I}_{>\varepsilon}(m)$.

For $m_{\leq\varepsilon}$ (the case of a tie), we find the parameters of $q = mm_{\leq\varepsilon}$. Again, the normalization constant is

$$\alpha_\leq = \int_{-\infty}^{+\infty} mm_{\leq\varepsilon}\mathrm{d}x = \int_{-\varepsilon}^{\varepsilon} m\mathrm{d}x = \frac{1}{2}\mathrm{erf}\left(\frac{(\epsilon+\mu)}{\sqrt{2}\sigma}\right) - \frac{1}{2}\mathrm{erf}\left(\frac{(-\epsilon+\mu)}{\sqrt{2}\sigma}\right),$$

and we get

$$E(q) = \int_{-\infty}^{+\infty} \frac{xmm_{\leq\varepsilon}}{\alpha_\leq}\mathrm{d}x = \int_{\varepsilon}^{+\infty} \frac{xm}{\alpha_\leq}\mathrm{d}x = \mu + \frac{\sigma(e^{-\frac{(\epsilon-\mu)^2}{2\sigma^2}} + e^{-\frac{(\epsilon+\mu)^2}{2\sigma^2}})}{\alpha_\leq\sqrt{2\pi}},$$

$$D(q_d) = \int_{-\infty}^{+\infty} \frac{d^2m_dm_{d>\varepsilon}}{\alpha_\leq}dd = \int_{\varepsilon}^{+\infty} \frac{d^2m_d}{\alpha_\leq}dd$$

$$= \mu^2 + \sigma^2 - \sigma\frac{(\varepsilon+\mu)e^{-\frac{(-\varepsilon+\mu)^2}{2\sigma^2}} + (\varepsilon-\mu)e^{-\frac{(\varepsilon+\mu)^2}{2\sigma^2}}}{\sqrt{2\pi}\alpha_\leq}.$$

Thus, we have computed the parameters of distribution q and are now able to find $m_{\leq\varepsilon}$ as q/m. We denote the resulting distribution as $\mathbb{I}_{\leq\varepsilon}(m)$.