<Introduction>

- Elo system에 대하여 설명
- Elo system에 대한 아이디어는 두 게임 플레이어들의 점수를 s1, s2로 지정.
    - 이후 s1, s2를 함수에 이용하여 선수들의 게임 결과에 대한 확률을 모델링한 것.
- 정규 분포를 따름
- 'i' 가 each player, 's' 는 skill and 'p' 는 performance, 'B'는 고정 변동값을 의미할때

$$P\left(p_1 > p_2 \middle| s_1, s_2\right) = \Phi\left(\frac{s_1 - s_2}{\sqrt{2}\beta}\right)$$

- 위의 식은 player 1이 이기는 확률로, player 1의 performance p1이 상대방의 performance p2를 능가할 확률과 같다.
- 게임이 끝난 이후, s1 과 s2는 업데이트 되며, s1 + s2 는 상수 유지
- player 1 이 이기면 y값은 1이 추가된다고 할 때, player 2가 이기면 y 값은 마이너스 1이 적용. 만약 무승부일경우, y는 0. (즉, $s1 \leftarrow s1 + y\Delta$, $s2 \leftarrow s2 - y\Delta$)

$$\Delta = \underbrace{\alpha\beta\sqrt{\pi}}_{K-\mathrm{Factor}} \left(\frac{y+1}{2} - \Phi\left(\frac{s_1 - s_2}{\sqrt{2}\beta}\right)\right)$$

- 위의 식에서 α 의 범위는 $0 < \alpha < 1$로, 이전까지의 점수/평가 vs 새 경기 점수에 대한 가중치를 결정
- 2명이 아닌 그 이상의 플레이어들이 함께 하는 게임 (Multiplayer game)에서 적용이 어려움 - 게임 결과는 팀 점수를 언급. 그러나 다음 매칭을 위해서는 각 팀원들의 개인적인 점수가 필요
- Multiplayer game 적용을 위하여 Trueskill system이 고안됨.

<Factor Graphs for Ranking>

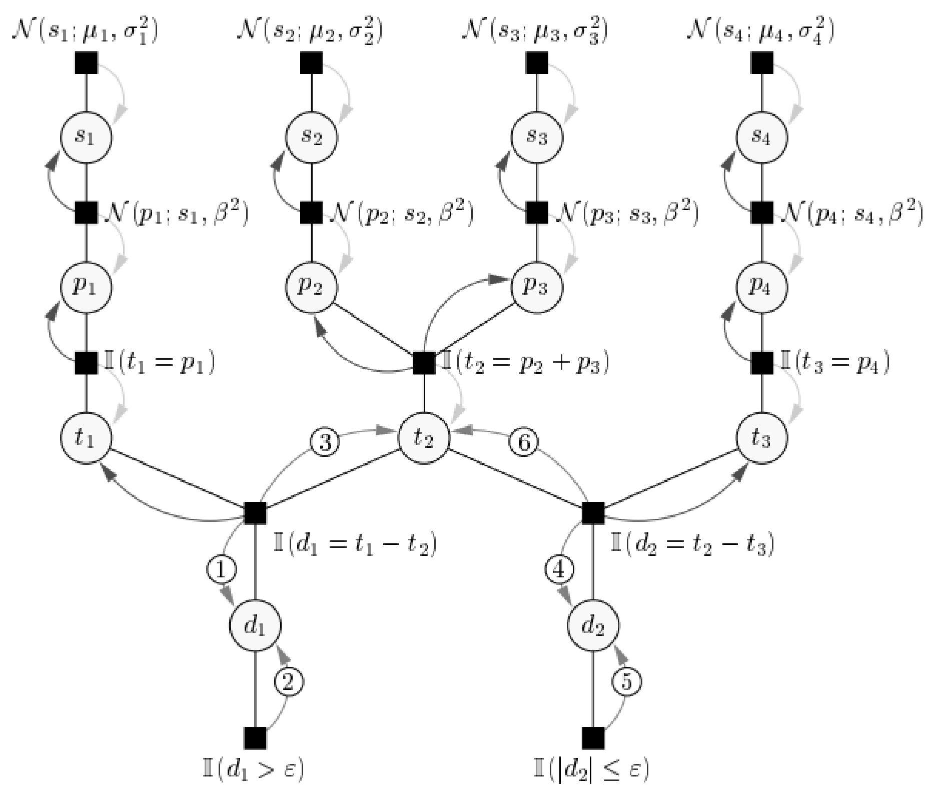
$$p\left(\mathbf{s} \middle| \mathbf{r}, A\right) = \frac{P\left(\mathbf{r} \middle| \mathbf{s}, A\right) p\left(\mathbf{s}\right)}{P\left(\mathbf{r} \middle| A\right)} .$$

- Team들을 반영하는 게임 결과에 대한 확률 모델
- 'r' = 게임 순위, 's' = 참여 플레이어의 실력/수준, 'A' = team의 모집단

$$P(\mathbf{r} | \{t_1, \ldots, t_k\}) = P\left(t_{r_{(1)}} > t_{r_{(2)}} > \cdots > t_{r_{(k)}}\right)$$

- 무승부를 고려하지 않을때에, 게임의 등수에 대한 확률
- 이 때, Rank 인 r 은 오름차순했다고 가정
- t_j = 팀 j의 퍼포먼스 능력 = 팀원들 퍼포먼스의 합
- 만약, 무승부일 경우일지라도, 팀의 퍼포먼스 점수는 ‘draw margin’(무승부 마진)으로 인해 다를 수 밖에 없기 때문에, 두 무승부 팀의 실력이 완전히 같은 것은 아니다.

- Gaussian density filtering을 사용하여 한 게임의 플레이어들의 스킬(s)을 측정할 수 있다.
- 만약 skill이 시간에 따라 다양해진다면, Gaussian dynamics factor을 사용 ($\mathcal{N}(s_{i,t+1}; s_{i,t}, \gamma^2)$ 여기에서 gamma^2 은 additive variance component.)



<Figure 1>

- 위의 그림은 TrueSkill system에 대한 factor graph 예시.
- 화살표는 인과 관계를 의미함. 연한 화살표는 위에서 아래로 실력 계산을 위해 사용되고, 어두운 색 화살표는 아래에서 위로 다음 매치를 위한 정보 전달을 위해 사용.

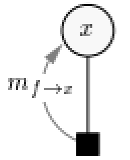
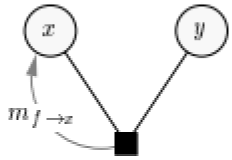
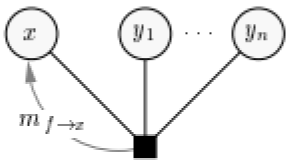
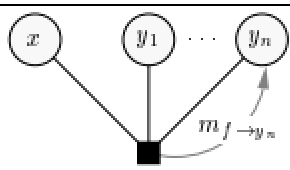
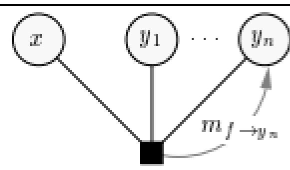
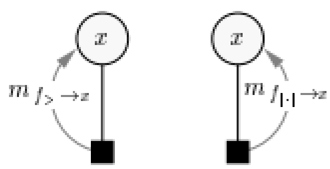
<Approximate Message Passing>

$$
\begin{aligned}
p\left(v_k\right) &= \prod_{f \in F_{v_k}} m_{f \rightarrow v_k}\left(v_k\right) \\
m_{f \rightarrow v_j\left(v_j\right)} &= \int \cdots \int f(\mathbf{v}) \prod_{i \neq j} m_{v_i \rightarrow f}\left(v_i\right) d\mathbf{v}_{\backslash j} \\
m_{v_k \rightarrow f}\left(v_k\right) &= \prod_{\bar{f} \in F_{v_k} \backslash\{f\}} m_{\bar{f} \rightarrow v_k}\left(v_k\right),
\end{aligned}
$$

- 위 그림은 Message Passing에 관한 공식들
- $\boldsymbol{F}_{\boldsymbol{v}_{\boldsymbol{k}}}$ 는 변수 v_k 와 관련한 factor들의 set이다.
- $\mathbf{v}\backslash j$ 는 벡터 v의 j번째 component를 제외한 components
- True Skill Factor Graph는 순환적인 것이 아닌 반복되는 것!
  - Figure 1에서 2번과 5번 메시지를 제외한 나머지 메시지들은 1 -dimensional Gaussians 따른다.
- 기대 전파 알고리즘 (Expectation propagation)는 확률 분포에 대한 근사값을 찾는 것으로, 대상 분포에 인수분해 구조를 활용하는 반복적인 접근 방식을 사용한다.
- Moment matching : 만약 q(x)가 가우시안 N(x|μ, Σ) 을 따르면, Kullback-Leibler divergence 'KL(p||q)' 는 μ, Σ 되면서 p(x)의 평균과 p(x)의 공분산과 동등해진다. 이 때, p(x) 는 intractable한 확률 분포를 의미하며, q(x)는 tractable한 분포를 의미한다.

$$\hat{p}\left(d_i\right)=\hat{m}_{f \rightarrow d_i}\left(d_i\right) \cdot m_{d_i \rightarrow f}\left(d_i\right) \quad \Leftrightarrow \quad \hat{m}_{f \rightarrow d_i}\left(d_i\right)=\frac{\hat{p}\left(d_i\right)}{m_{d_i \rightarrow f}\left(d_i\right)}.$$

  -

Factor	Update equation
 $\mathcal{N}(x; m, v^2)$	$\pi_x^{\text{new}} \leftarrow \pi_x + \frac{1}{v^2}$ $\tau_x^{\text{new}} \leftarrow \tau_x + \frac{m}{v^2}$
 $\mathcal{N}(x; y, c^2)$	$\pi_{f\to x}^{\text{new}} \leftarrow a\left(\pi_y - \pi_{f\to y}\right)$ $\tau_{f\to x}^{\text{new}} \leftarrow a\left(\tau_y - \tau_{f\to y}\right)$ $a := \left(1 + c^2\left(\pi_y - \pi_{f\to y}\right)\right)^{-1}$ $m_{f\to y}$ follows from $\mathcal{N}\left(x; y, c^2\right) = \mathcal{N}\left(y; x, c^2\right)$.
 $\mathrm{I}(x = \mathbf{a}^\top \mathbf{y})$	$\pi_{f\to x}^{\text{new}} \leftarrow \left(\sum_{j=1}^{n} \frac{a_j^2}{\pi_{y_j} - \pi_{f\to y_j}}\right)^{-1}$ $\tau_{f\to x}^{\text{new}} \leftarrow \pi_{f\to x}^{\text{new}} \cdot \left(\sum_{j=1}^{n} a_j \cdot \frac{\tau_{y_j} - \tau_{f\to y_j}}{\pi_{y_j} - \pi_{f\to y_j}}\right)$
 $\mathrm{I}(x = \mathbf{b}^\top \mathbf{y})$	 $\mathrm{I}(y_n = \mathbf{a}^\top[y_1, \cdots, y_{n-1}, x])$ $\mathbf{a} = \frac{1}{b_n} \cdot \begin{bmatrix} -b_1 \\ \vdots \\ -b_{n-1} \\ +1 \end{bmatrix}$
 $\mathrm{I}(x > \varepsilon)$ $\mathrm{I}(\lvert x \rvert \le \varepsilon)$	$\pi_x^{\text{new}} \leftarrow \frac{c}{1 - W_f\left(d/\sqrt{c}, \varepsilon\sqrt{c}\right)}$ $\tau_x^{\text{new}} \leftarrow \frac{d + \sqrt{c} \cdot V_f\left(d/\sqrt{c}, \varepsilon\sqrt{c}\right)}{1 - W_f\left(d/\sqrt{c}, \varepsilon\sqrt{c}\right)}$ $c := \pi_x - \pi_{f\to x}, \quad d := \tau_x - \tau_{f\to x}$

-
    - 위에서부터 순서대로 네 개의 그림은 standard Gaussian integrals이다.
    - 마지막 그림은 moment matching 의 결과이다.
- Marginal 근사값이 변화지 않을 때까지 앞에서 설명한 메세지 반복을 계속 해야한다.

<Experiments and Online Service - Halo 2 Beta Test>

- TrueSkill과 ELO의 성능을 비교하기 위하여 Halo2의 4가지 유형에 매치에 적용시켰다.
- Halo2의 match type은 다음과 같다
    - Free For All(8인 개인전)
    - Small teams(4vs4 팀전)
    - Head To Head(1:1)
    - Large teams(8vs8 팀전)

$$\text{draw probability} = \Phi\left(\frac{\varepsilon}{\sqrt{n_1 + n_2}\,\beta}\right) - \Phi\left(\frac{-\varepsilon}{\sqrt{n_1 + n_2}\,\beta}\right) = 2\Phi\left(\frac{\varepsilon}{\sqrt{n_1 + n_2}\,\beta}\right) - 1,$$

- ε은 factor node에 대한 draw margin으로, 팀들간의 무승부 비율(“empirical draw probability”)을 집계하여 구한다.
- 위의 draw probability식에 근거하여 ε을 무승부의 가능성과 연관짓는다.
- 여기서 n1, n2는 각각 두팀의 선수의 수이다.

[TrueSkill vs ELO]

- Setting: α=0.07, Gaussian performance distribution으로 TrueSkill과 ELO를 비교한다.
    - Elo scale에서 24 K factor에 부합하는 것으로, 안정적이고 좋은 dynamics로 여겨진다.
- 팀 게임 혹은 두 팀이상의 게임을 진행할 때는 ‘Duelling heuristic’ 사용
    - 팀 결과와 모든 선수 개개인의 퍼포먼스를 비교하여 Δ값을 계산한 뒤, Δ의 평균을 선수 점수에 업데이트 시킨다.
-

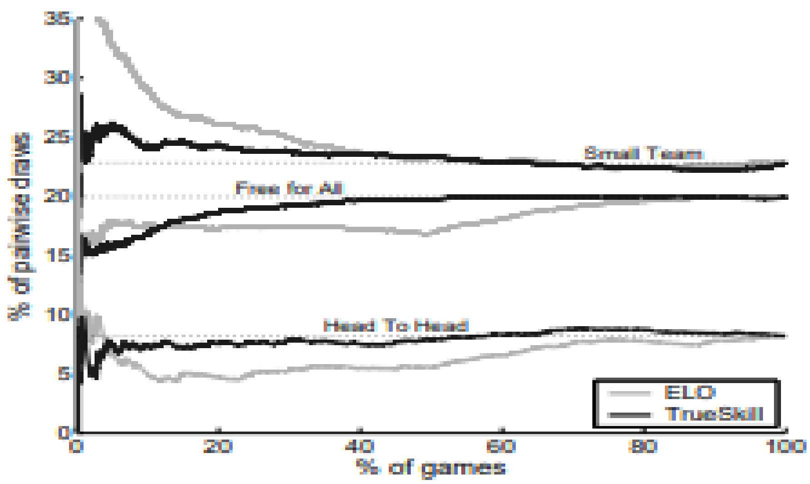
	ELO full	TrueSkill full	ELO “challenged”	TrueSkill “challenged”
Free for All	32.14%	**30.82%**	38.30%	**35.64%**
Small Teams	**34.92%**	35.23%	42.55%	**37.17%**
Head to Head	33.24%	**32.44%**	40.57%	**30.83%**
Large Teams	39.49%	**38.15%**	44.12%	**29.94%**

- 위 테이블은 prediction error를 보여주는 테이블이다.
- TrueSkill에는 아래의 식을 이용하여 매치메이킹의 기준을 만들었고, ELO에서는 ELO 점수(s1-s2)를 이용하였다.

$$q_{\text{draw}}\left(\beta^2, \mu_i, \mu_j, \sigma_i, \sigma_j\right) := \sqrt{\frac{2\beta^2}{2\beta^2+\sigma_i^2+\sigma_j^2}} \cdot \exp\left(-\frac{(\mu_i-\mu_j)^2}{2\left(2\beta^2+\sigma_i^2+\sigma_j^2\right)}\right)$$

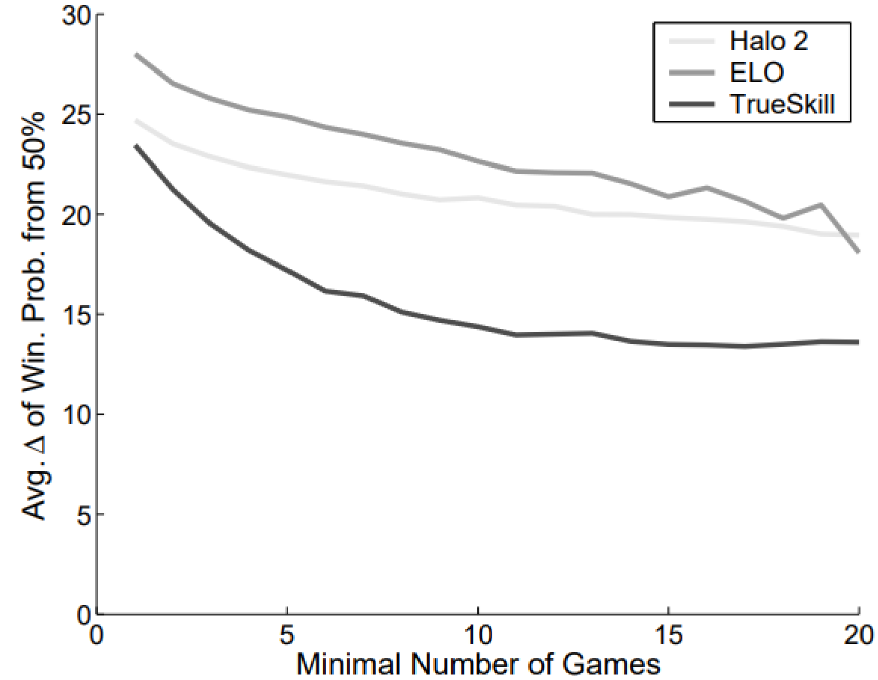
- ELO"challenged" 컬럼과 TrueSkill"challenged"컬럼을 통해 , tight한 matchmaking을 하는데에 TruSkill이 더 효과적이라는 것을 알 수 있다.

- Match Quality

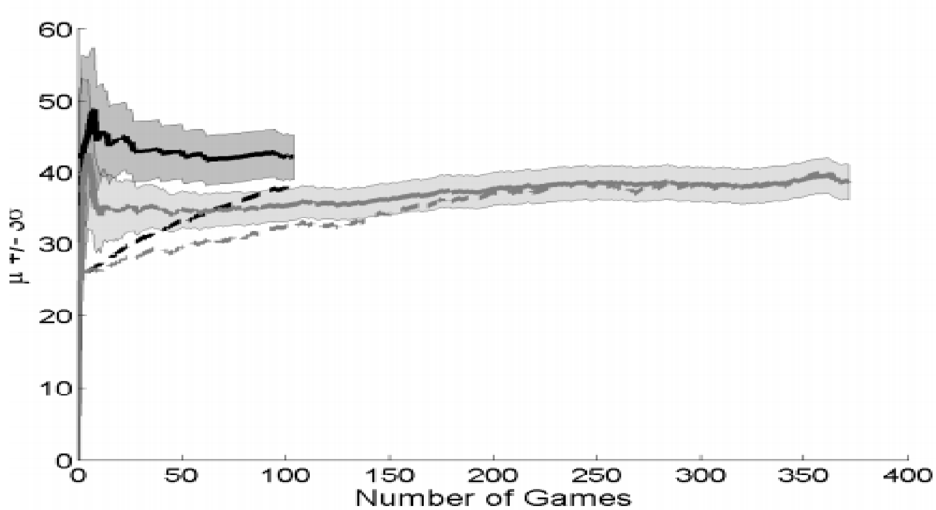


-
  - 비슷한 수준의 실력으로 유저를 매칭할 때의 기준
  - Elo vs TrueSKill
    - "Small Teams"에서는 적용이 안되었지만, "Free for All"과 "Head to Head"게임에서 TrueSkill 모델이 Elo 보다 우수함을 보여줌.

- Win Probability

- 

    - 우승 확률에 대한 것으로, 우승 확률이 높을 수록 너무 약한 플레이어와 매칭됨을 의미한다.
    - 각 플레이어의 우승 확률과 해당 게임의 선수의 최소 게임수에 따라 우승확률의 평균편차를 측정한다.
    - 위 plot은 “Head to Head” 게임에 대한 것으로, TrueSkill system 덕에 적은 게임으로 공정한 매치가 잘 이루어졌음을 보여준다.

- Convergence Properties
- 

- 위 그림은 “Free for All” 게임 모드에서 가장 실력이 좋은 두 선수의 수렴 그래프이다.(뚜렷한 선: TrueSkill, 점선: ELO)
- Elo는 천천히 목표에 천천히 수렴하는 반면에, TrueSkill은 자동적으로 학습률을 고르는 것을 볼수 있다.
- Tureskill은 선수들의 랭킹을 인코딩하기 위해 nlog(n) bits에 대한 information theoretic limit에 가까워진다(수렴).

<Experiments and Online Service - TrueSkill in Xbox 360 Live>

- Xbox Live에서
  - $\mu_0 = 25 \text{ and } \sigma_0^2 = (25/3)^2$
  - Positive skills는 약 99%에 상응한다.
  - 리더보드 상위권자들은 μ-3σ에 분포한다.
  - Pairwise matchmaking은 ε이 0으로 수렴할 때, 아래의 무승부 확률 공식을 이용한다.
    - $$q_{\text{draw}}\left(\beta^2, \mu_i, \mu_j, \sigma_i, \sigma_j\right) := \sqrt{\frac{2\beta^2}{2\beta^2 + \sigma_i^2 + \sigma_j^2}} \cdot \exp\left(-\frac{(\mu_i - \mu_j)^2}{2\left(2\beta^2 + \sigma_i^2 + \sigma_j^2\right)}\right)$$
- 흥미로운 발견
  - 기회(운)을 베이스로 하는 게임에서는 스킬의 분포가 작지만, 레이싱게임 같은 실력 베이스의 게임은 스킬의 분포가 크다.
  - Matchmaking과 Skill은 플레이어 들에게 피드백을 줄 수 있다.
  - Skill reset이 시행되면, 플레이어가 새로운 system의 게임에서 몇판을 연속적으로 지더라도 reset함으로써 갱신하고 그럼으로써 tight한 Matchmaking이 이루어진다.

<conclusion>

- TrueSkill은 Bayesian skill rating 알고리즘을 기반한 것으로, factor graph에서의 메세지 패싱된 추정값에 근거하고 엘로시스템을 능가하며 실전에서도 잘 적용된다.
- Factor graph는 넓은 범위의 멀티클래스와 랭킹 문제를 포함한 제약 분류 모델에 적용이 용이하다.
- Feature 벡터들은 랭킹 함수를 만드는 데 사용할 수 있다.