

The Sum-Product Algorithm Explained

Ralf Herbrich
Amazon

Overview

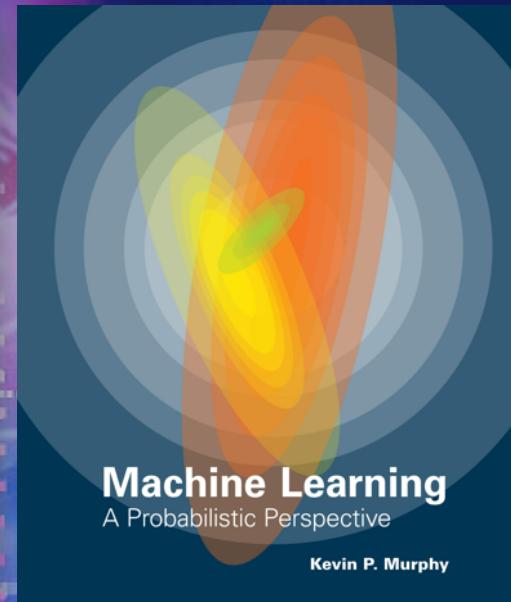
- Theory
 - Inference in Factor Graphs
 - Approximate Message Passing
- Applications @ Microsoft
 - TrueSkill: Gamer Rating and Matchmaking
 - TrueSkill Through Time: History of Chess

Background Material

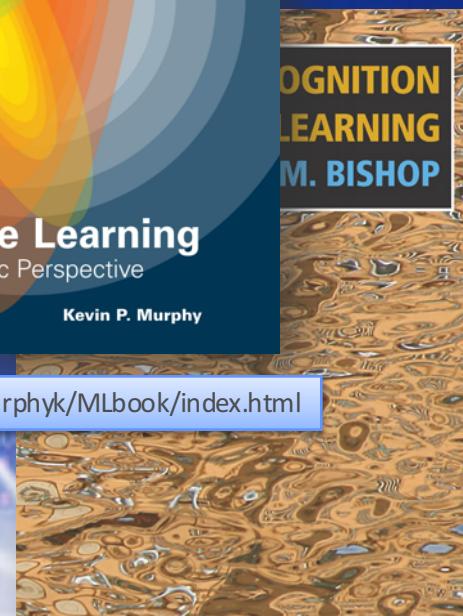
Coursera



<http://www.cs.ucl.ac.uk/staff/d.barber/bmml/>



<http://www.cs.ubc.ca/~murphyk/MLbook/index.html>



<http://research.microsoft.com/en-us/um/people/cmbishop/PRML/index.htm>

Overview

- **Theory**
 - Inference in Factor Graphs
 - Approximate Message Passing
- Applications
 - TrueSkill: Gamer Rating and Matchmaking
 - TrueSkill Through Time: History of Chess

Graphical Models

- **Definition:** Graphical representation of joint probability distribution
 - Nodes:  = Variables
 - Edges: Relationship between variables
- **Variables:**
 - Observed Variables: Data
 - Unobserved Variables: ‘Causes’ + Temporary/Latent
- **Key Questions:**
 - (Conditional) Dependency: $p(a, b|c) \stackrel{?}{=} p(a|c) \cdot p(b|c)$
 - Inference/Marginalisation: $p(a, b) = \sum_c p(a, b, c)$

Factor Graphs

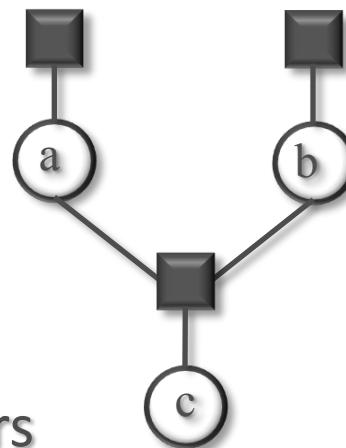
- **Definition:** Graphical representation of product structure of a function (Wiberg, 1996)
 - Nodes: ■ = Factors ○ = Variables
 - Edges: Dependencies of factors on variables.

- **Semantic:**

$$p(\mathbf{x}) = \prod_f f(\mathbf{x}_{V(f)})$$

- Local variable dependency of factors

$$p(a, b, c) = f_1(a) \cdot f_2(b) \cdot f_3(a, b, c)$$



Factor Graphs and Bayes' Law

- Bayes' law

$$p(s|y) \propto p(y|s) \cdot p(s)$$

- Factorising prior

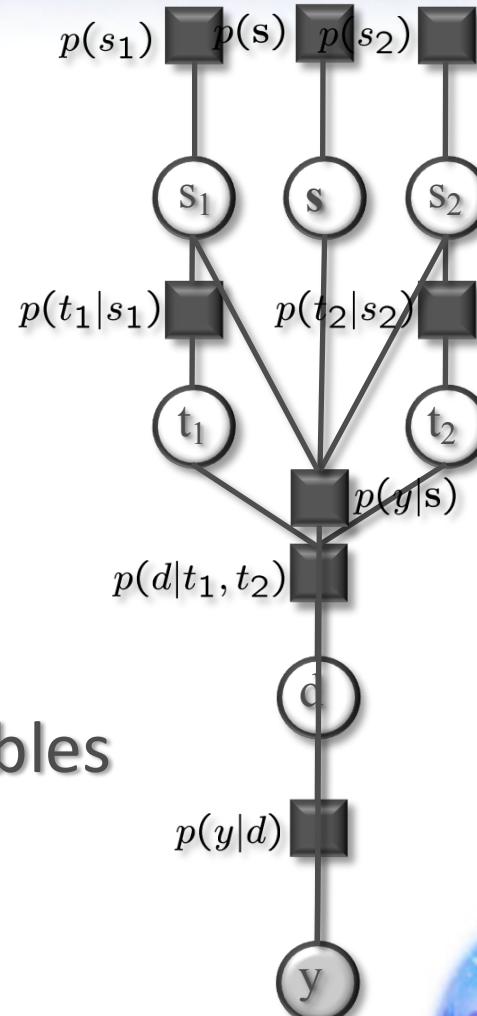
$$p(s) = p(s_1) \cdot p(s_2)$$

- Factorising likelihood

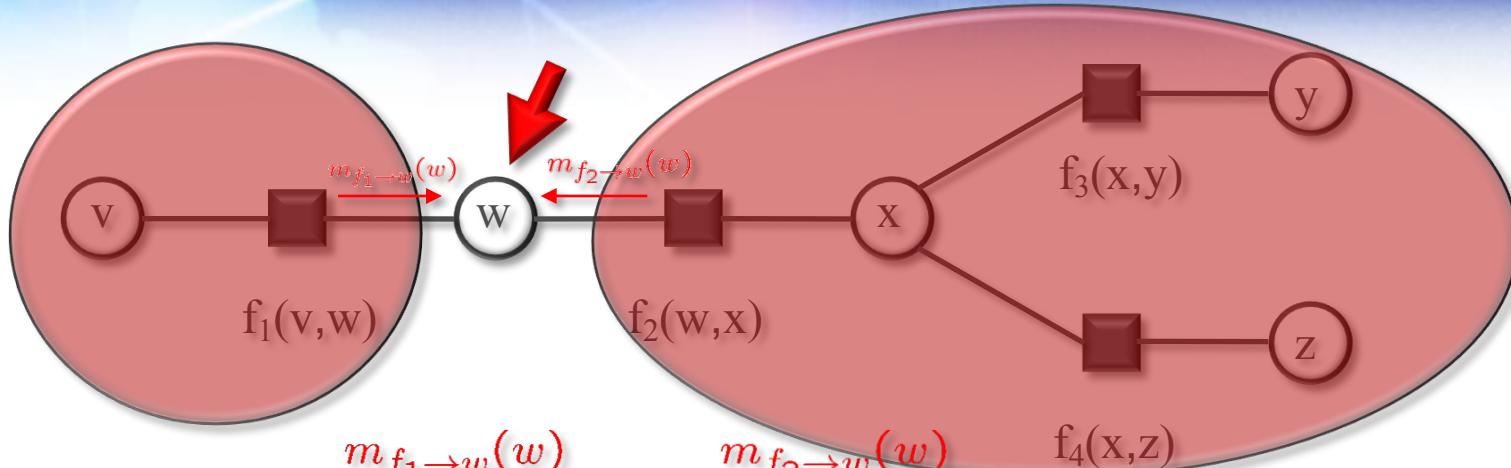
$$p(y, t, d|s) = \prod_i p(t_i|s_i) \cdot p(d|t_1, t_2) \cdot p(y|d)$$

- Inference: Sum out latent variables

$$p(y|s) = \sum_{\mathbf{t}} \sum_d p(y, \mathbf{t}, d|s)$$



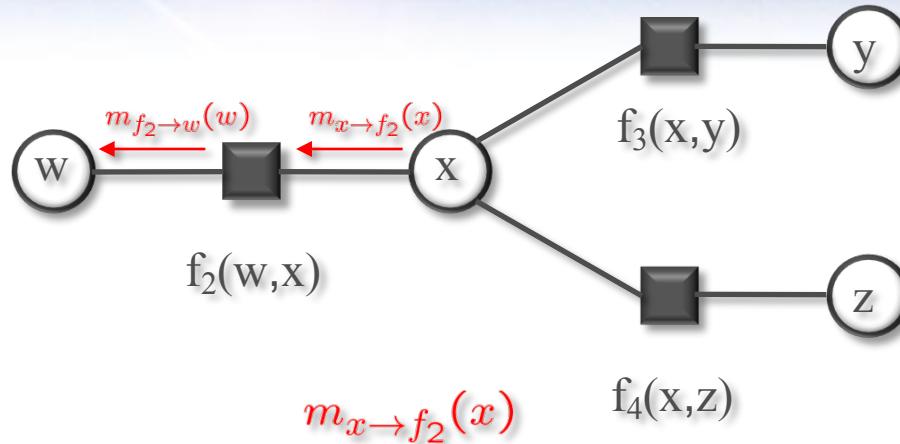
Factor Trees: Separation



$$p(w) = \left[\sum_{v} \sum_{x} \sum_{y} \sum_{z} f_1(v, w) f_2(x, w) f_3(x, y) f_4(x, z) \right]$$

Observation: Sum of products becomes product of sums of all messages from neighbouring factors to variable!

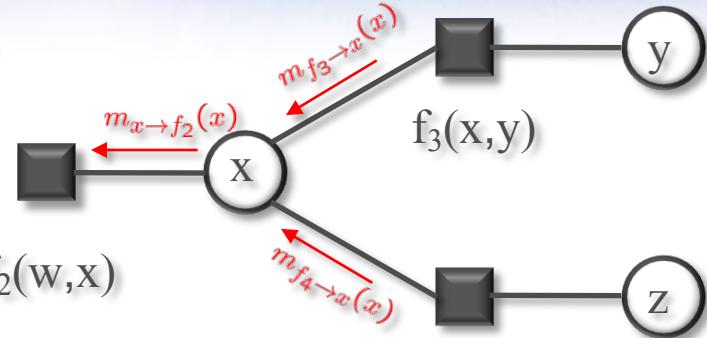
Messages: From Factors To Variables



$$m_{f2 \rightarrow w}(w) = \sum_x \sum_y (\sum_z f_2(w, x) f_3(x, y) f_4(x, z))$$

Observation: Factors only need to sum out all their local variables!

Messages: From Variables To Factors



$$m_{x \rightarrow f_2}(x) = \left[\sum_y \left[\sum_z f_3(y), y \right] \right] \left[\sum_z f_4(x, z) \right]$$

Observation: Variables pass on the product of all incoming messages!

The Sum-Product Algorithm

- Three update equations (Aji & McEliece, 1997)

$$p(t) = \prod_{f \in F_t} m_{f \rightarrow t}(t)$$

$$m_{f \rightarrow t_1}(t_1) = \sum_{t_2} \sum_{t_3} \cdots \sum_{t_n} f(t_1, t_2, t_3, \dots) \prod_{i>1} m_{t_i \rightarrow f}(t_i)$$

$$m_{t \rightarrow f}(t) = \prod_{f_j \in F_t \setminus \{f\}} m_{f_j \rightarrow t}(t)$$

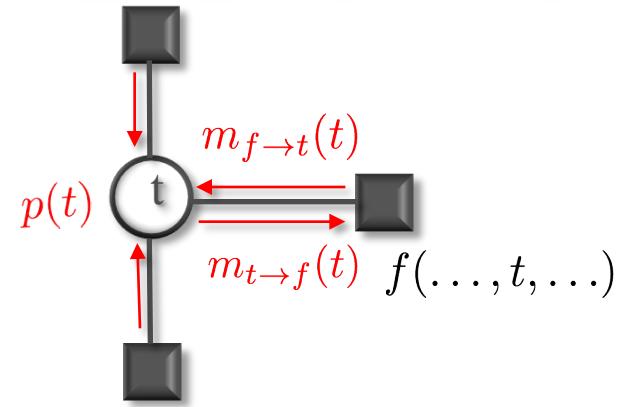
- Update equations can be directly derived from the distributive law.
- Calculate all marginals at the same time!
- Only need to pass messages twice along each edge!

Practical Considerations II

- Redundant computations:

$$p(t) = \prod_{f \in F_t} m_{f \rightarrow t}(t)$$

$$m_{t \rightarrow f}(t) = \prod_{f_j \in F_t \setminus \{f\}} m_{f_j \rightarrow t}(t)$$



- Caching: Only store $p(t)$ and $m_{f \rightarrow t}(t)$, then

$$m_{t \rightarrow f}(t) = \frac{p(t)}{m_{f \rightarrow t}(t)}$$

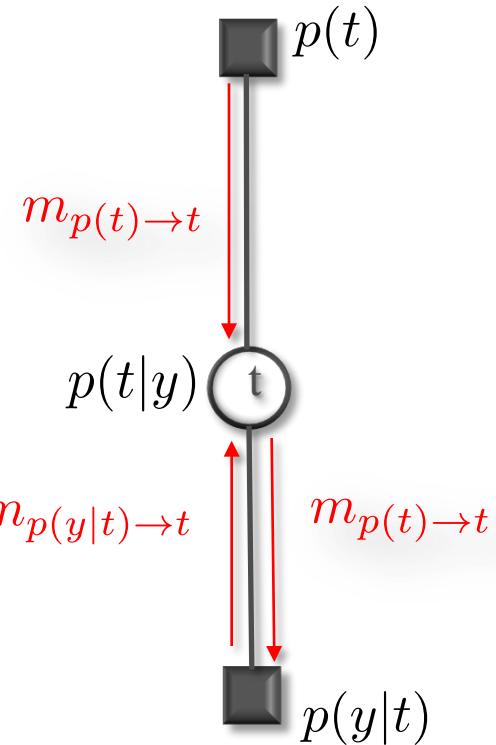
A Bayesian Interpretation

- Recall Bayes' Law:

$$p(t|y) \propto p(y|t) \cdot p(t)$$

- Prior and Data Messages:

$$\begin{aligned} p(t|y) &\propto m_{p(y|t) \rightarrow t} \cdot m_{p(t) \rightarrow t} \\ &\propto m_{p(y|t) \rightarrow t} \cdot m_{t \rightarrow p(y|t)} \end{aligned}$$



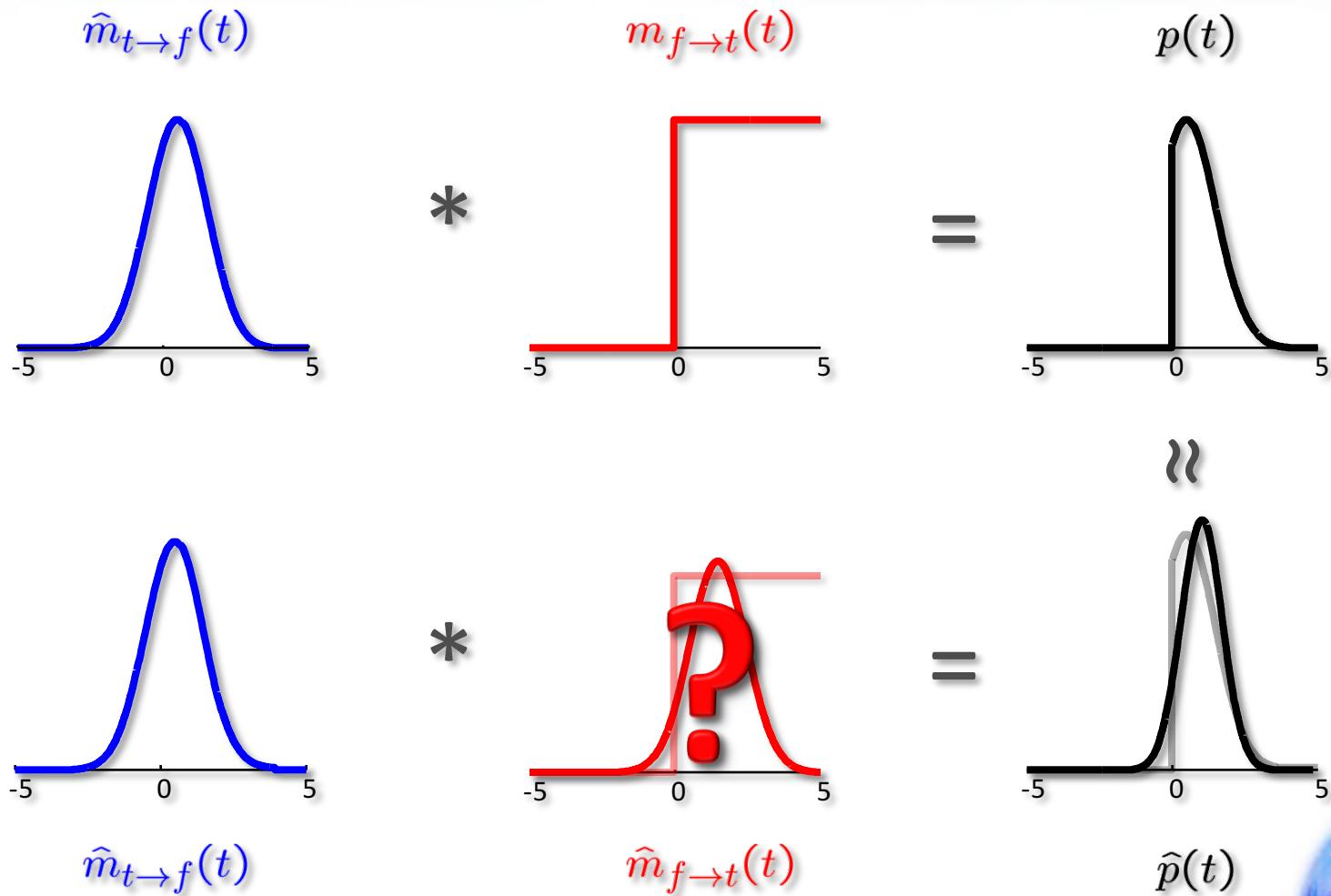
Message passing is separating the likelihood and prior into outgoing and incoming message!

Approximate Message Passing

- **Problem:** The exact messages from factors to variables may not be closed under products.
- **Solution:** Approximate *each* marginal as well as possible in using a divergence measure on beliefs.
- **General Idea:** Leave-one out approximation

$$\begin{aligned}\hat{p}(t) &= \operatorname{argmin}_{\hat{p}}, D \left[m_{f \rightarrow t} \cdot \hat{m}_{t \rightarrow f}, \hat{p} \right] \\ \hat{m}_{f \rightarrow t}(t) &= \frac{\hat{p}(t)}{\hat{m}_{t \rightarrow f}(t)}\end{aligned}$$

Approximate Message Passing



Divergence Measures

- **Kullback-Leibler Divergence:** Expected log-odd ratio between two distributions:

$$\text{KL}(p, q) := \sum_t p(t) \log \left(\frac{p(t)}{q(t)} \right)$$

- **Minimizer for Exponential Families:** Matching the moments of the distribution $p(t)$!
- **General α -Divergence:**

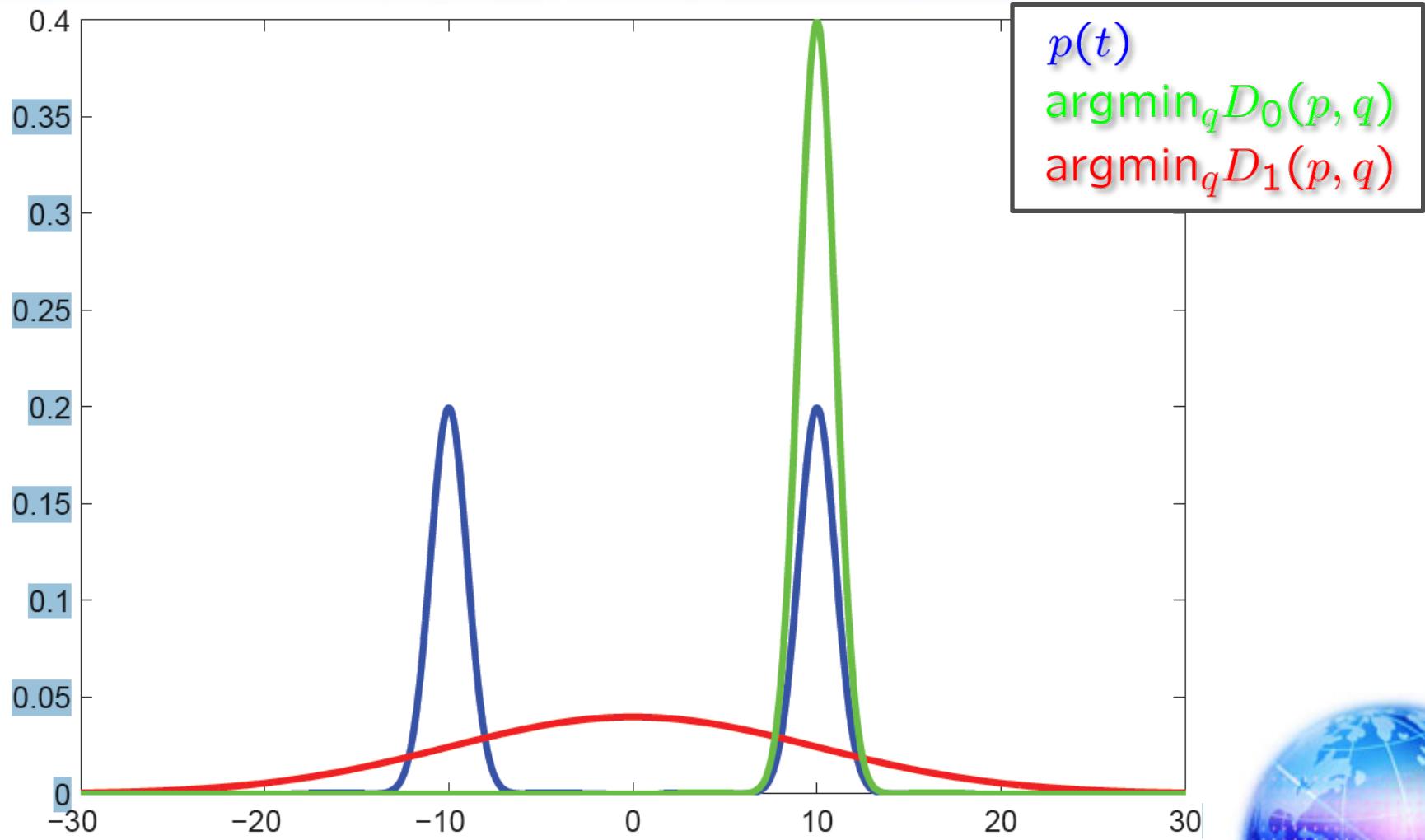
$$D_\alpha(p, q) := \frac{1 - \sum_t \frac{p^{\alpha-1}(t)}{q^{\alpha-1}(t)}}{\alpha(1 - \alpha)}$$

- **Special Cases:**

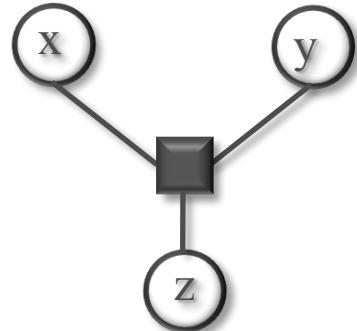
$$D_0(p, q) = \text{KL}(q, p)$$

$$D_1(p, q) = \text{KL}(p, q)$$

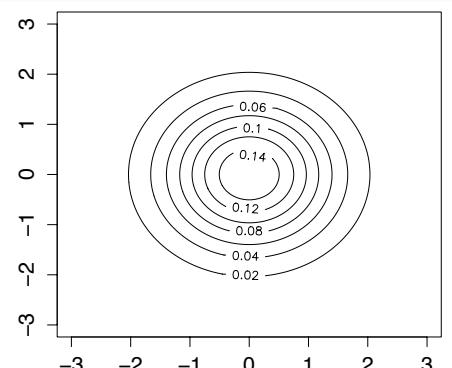
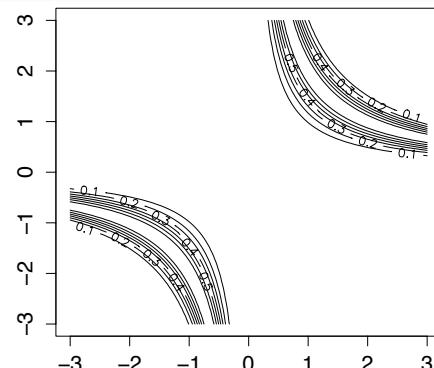
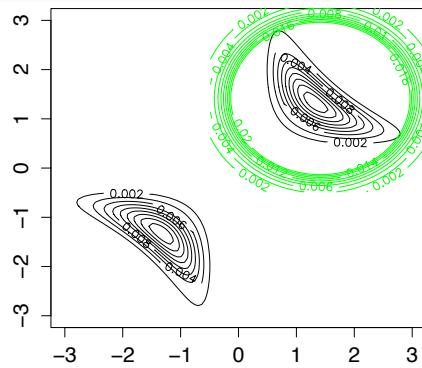
α -Divergence in Pictures



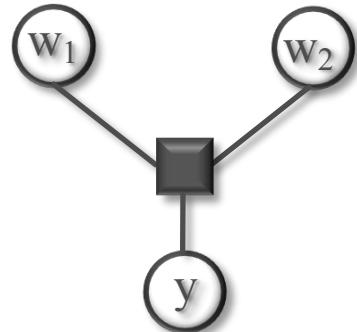
When to use which α -Divergence?



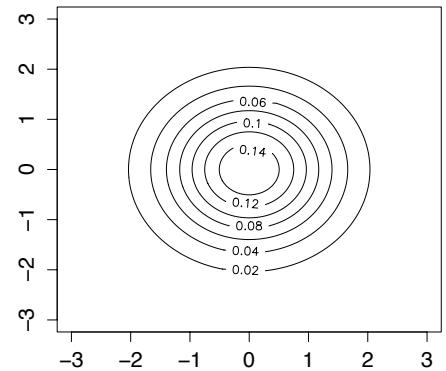
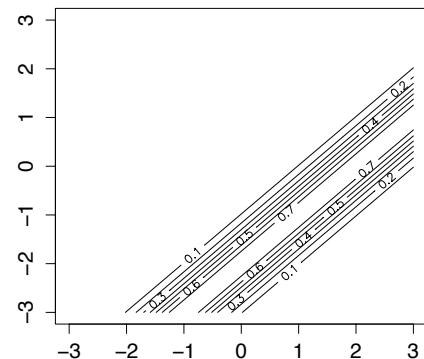
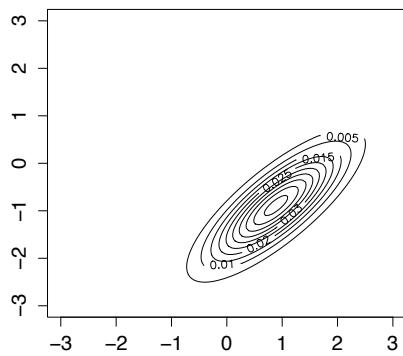
$\alpha=0$ resolves multi-modality in the posterior at the expense of too much certainty!



When to use which α -Divergence?

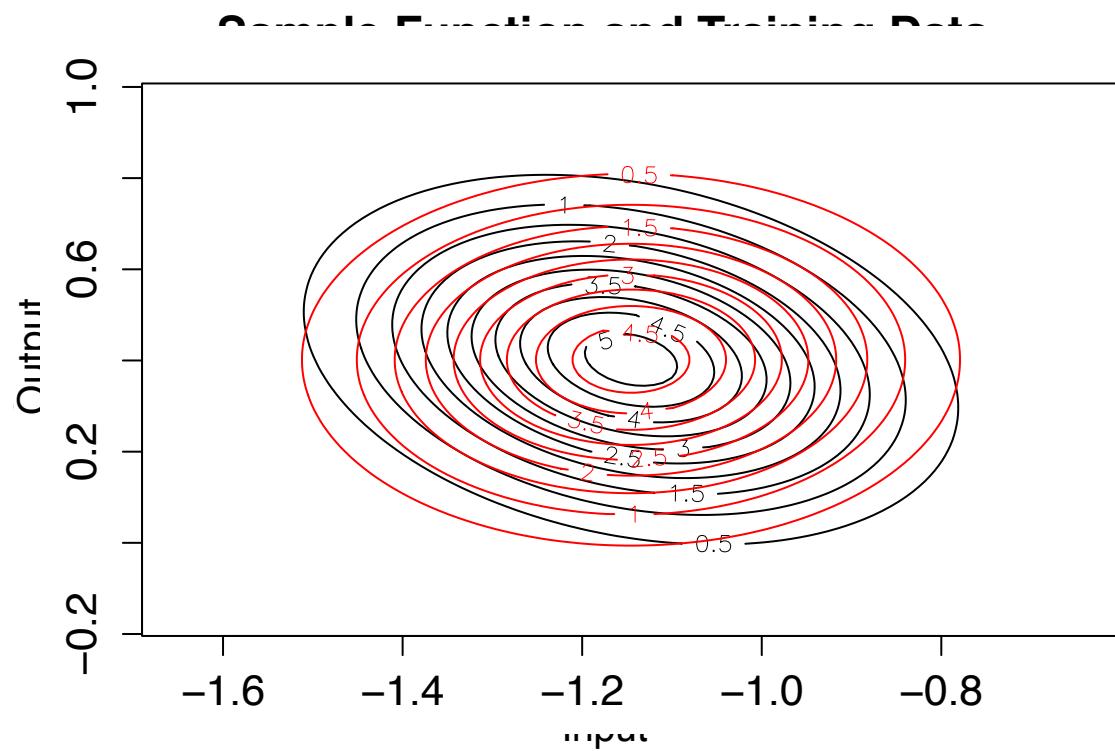


$\alpha=1$ captures all uncertainty for uni-modal posterior distributions!



Sample (ctd)

$$p(\mathbf{w}|y, \mathbf{x}) = \mathcal{N}(y; [\sin(3t); \sin(t) \cos(6t)]^T \mathbf{w}, 0.1) \cdot \mathcal{N}(\mathbf{w}; \mathbf{0}, \mathbf{I})$$



Overview

- Theory
 - Inference in Factor Graphs
 - Approximate Message Passing
- Applications @ Microsoft
 - TrueSkill: Gamer Rating and Matchmaking
 - TrueSkill Through Time: History of Chess

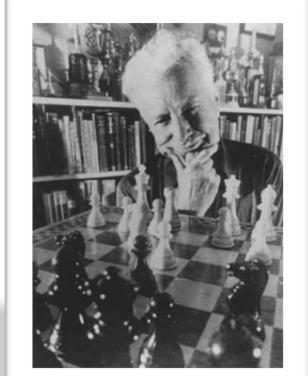
TrueSkill™

Joint work with Thore Graepel, Tom Minka & Phillip Treford



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
 - Matchmaking system for tournaments
- Challenges of online gaming
 - Learn from few match outcomes efficiently
 - Support multiple teams and multiple players per team



The Skill Rating Problem

- Given:
 - Match outcomes: Orderings among k teams

The image depicts a futuristic user interface for managing match outcomes and calculating skill ratings.

Match Outcomes:

- A large card shows the results of a match between the Red Team (Score: 50) and another team (Score: N/A). The Red Team is highlighted in orange.
- A smaller card shows the results of multiple matches, with the last match highlighted by a yellow border. The last match involved the Red Team (Score: 1), xXxHALOxXx (Score: 24), AjaySandhu (Score: 15), AjaySandhu(G) (Score: 15), Robert115 (Score: 11), TurboNegro84(G) (Score: 11), TurboNegro84 (Score: 5), and SniperEye(G) (Score: 1).

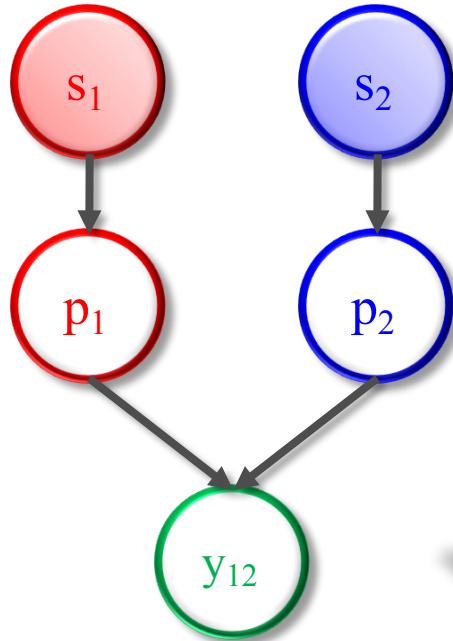
Skill Rating Problem:

- A large blue arrow points from the match outcomes towards a list of 17 players.
- The list of players and their current skill ratings is as follows:

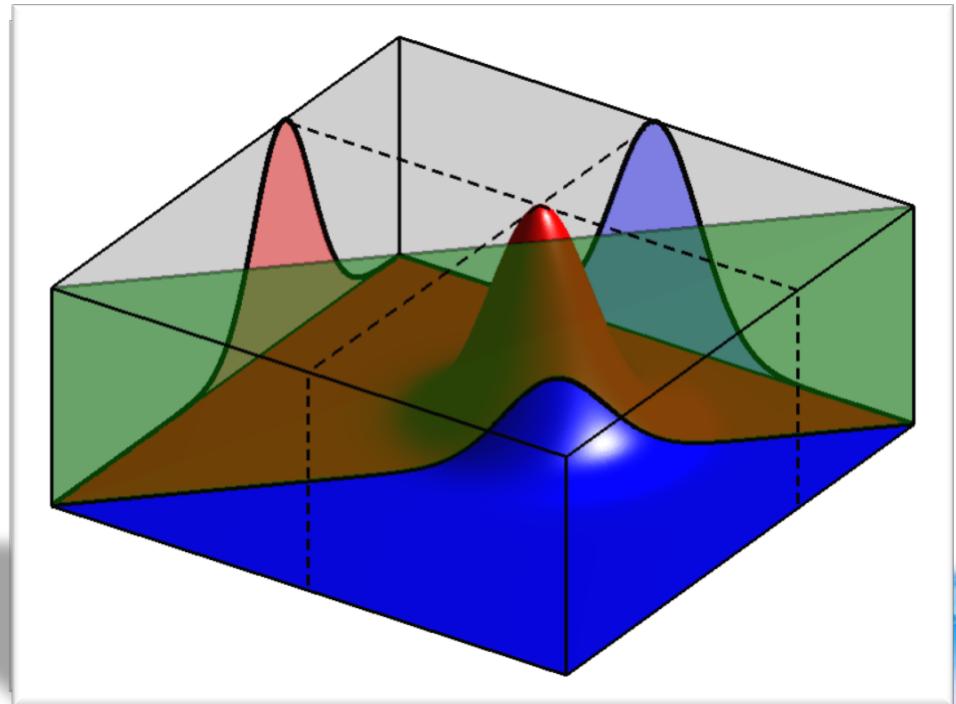
Rank	Player	Rating
1	SEWiCSYDE OWNs	27
2	FATAL REVENGE	26
3	Paranoia 1	25
4	Paulk	25
5	IxX OMG Xxl	25
6	BittyTom	25
7	brian 2007	24
8	SEXY MOZES	24
9	droplates	24
10	jaCKdaSaMuRai	24
11	II Me II	24
12	iamNightMare	24
13	a retarded007	24
14	Perfected Brit	24
15	THE MUFFIN MANx	24
16	TheVunit	23
17	Mr Sushi87	23

Two Player Match Outcome Model

- Latent Gaussian performance model for fixed skills
- Possible outcomes: Player 1 wins over 2 (and vice versa)

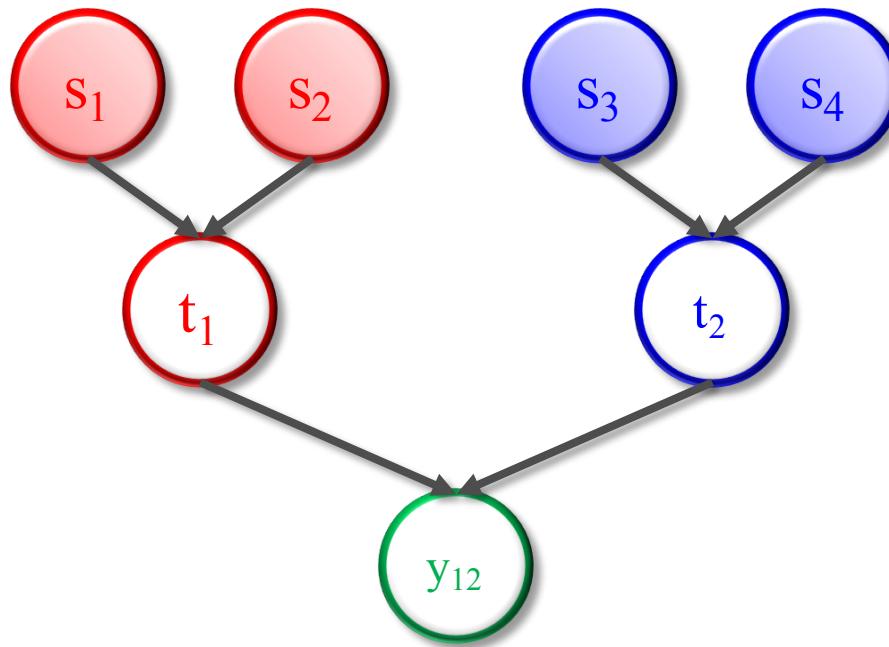


$$\mathbf{P}(y_{12} = (1, 2) | p_1, p_2) = \mathbb{I}(p_1 > p_2)$$



Two Team Match Outcome Model

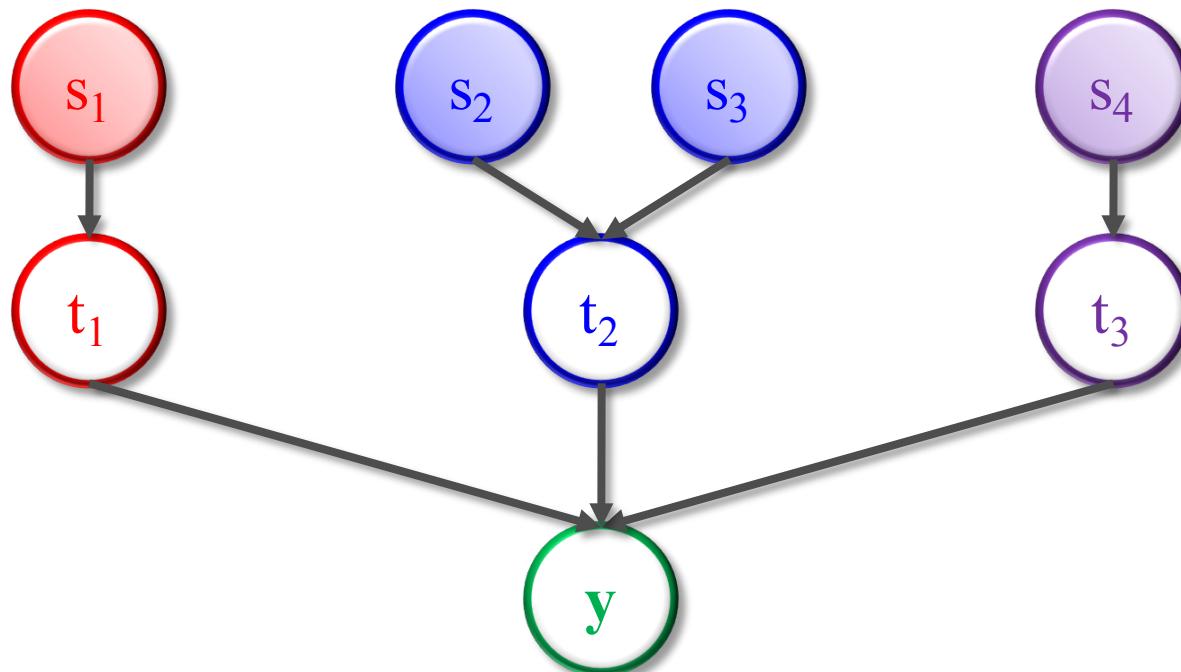
- Skill of a team is the sum of the skills of its members



$$\mathbf{P}(t_1 | s_1, s_2) = \mathcal{N} (t_1; s_1 + s_2, 2 \cdot \beta^2)$$

Multiple Team Match Outcome Model

- Possible outcomes: Permutations of the teams

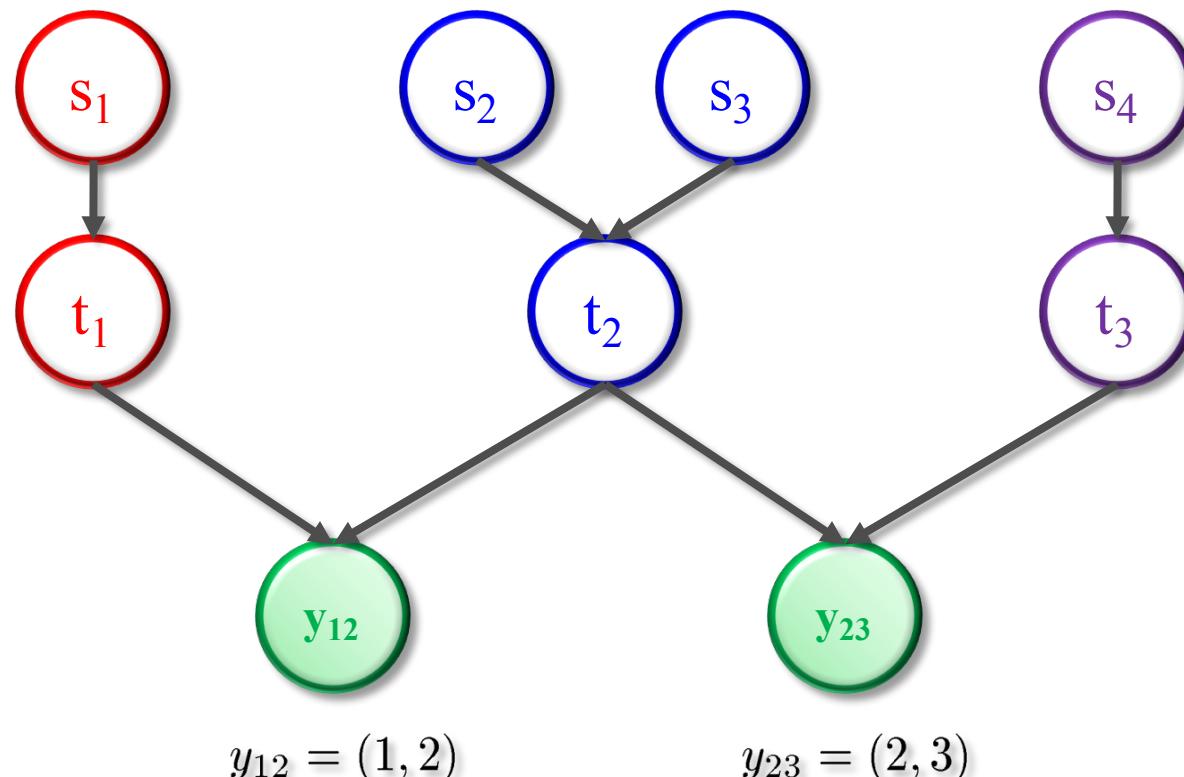


$$\mathbf{P}(y|t_1, t_2, t_3) = \mathbb{I}(y = (i, j, k)) \text{ where } t_i > t_j > t_k$$

Multiple Team Match Outcome Model

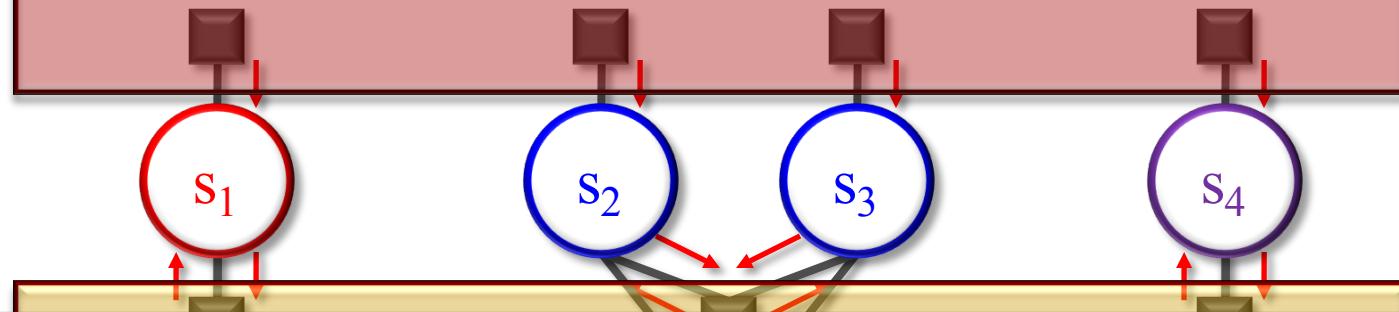
- But we are interested in the (Gaussian) posterior!

$$\mathbf{P}(s_i | \mathbf{y} = (1, 2, 3)) = \mathcal{N}(s_i; \mu_i, \sigma_i^2)$$



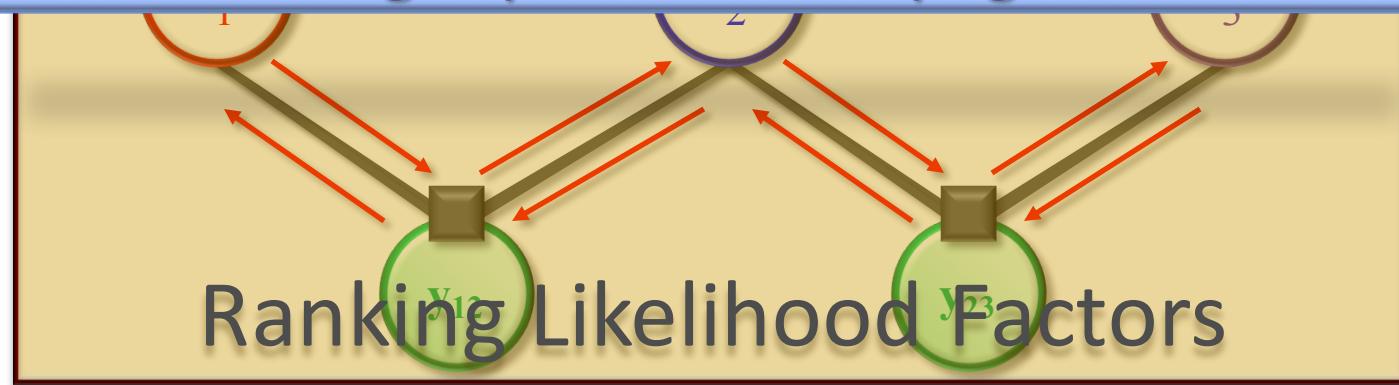
Efficient Approximate Inference

Gaussian Prior Factors



Fast and efficient approximate message passing
using Expectation Propagation

Ranking Likelihood Factors



Applications to Online Gaming

- Leaderboard
 - Global ranking of all players

$$\mu_i - 3 \cdot \sigma_i$$

- Matchmaking
 - For gamers: Most uncertain outcome

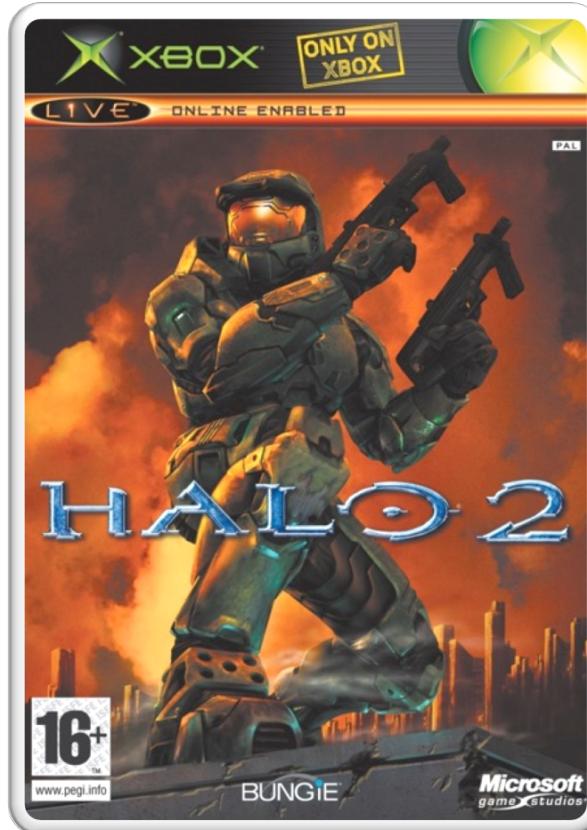
	Level	Gamertag	Avg. Life	Best Spree	Score	
1st	10	BlueBot	00:00:49	6	15	SEWICSYDE OWNS
1st	7	SniperEye	00:00:41	4	14	FATAL REVENGE
1st	9	ProThepirate	00:01:07	3	13	Paranoia 1
1st	10	dazdemon	00:00:59	3	8	Paulk
2nd	10	WastedHarry	00:00:41	4	17	IxX OMG Xxl
2nd	3	Ascla	00:00:57	4	17	BittyTom
2nd	9	Antidote4Losing	00:00:41	2	9	brian 2007
2nd	12	BlackKraak9	00:00:48	0	0	SEXY MOZES
						droplates
						jaCKdaSaMuRai
						II Me II
						iamNightMare
						a retarded007
						Perfected Brit
						THE MUFFIN MANx
						TheVunit
						Mr Sush87

$$P(p_i \approx p_j | \mu_i = \mu_j = 0, \sigma_i^2 + \sigma_j^2 = 0)$$

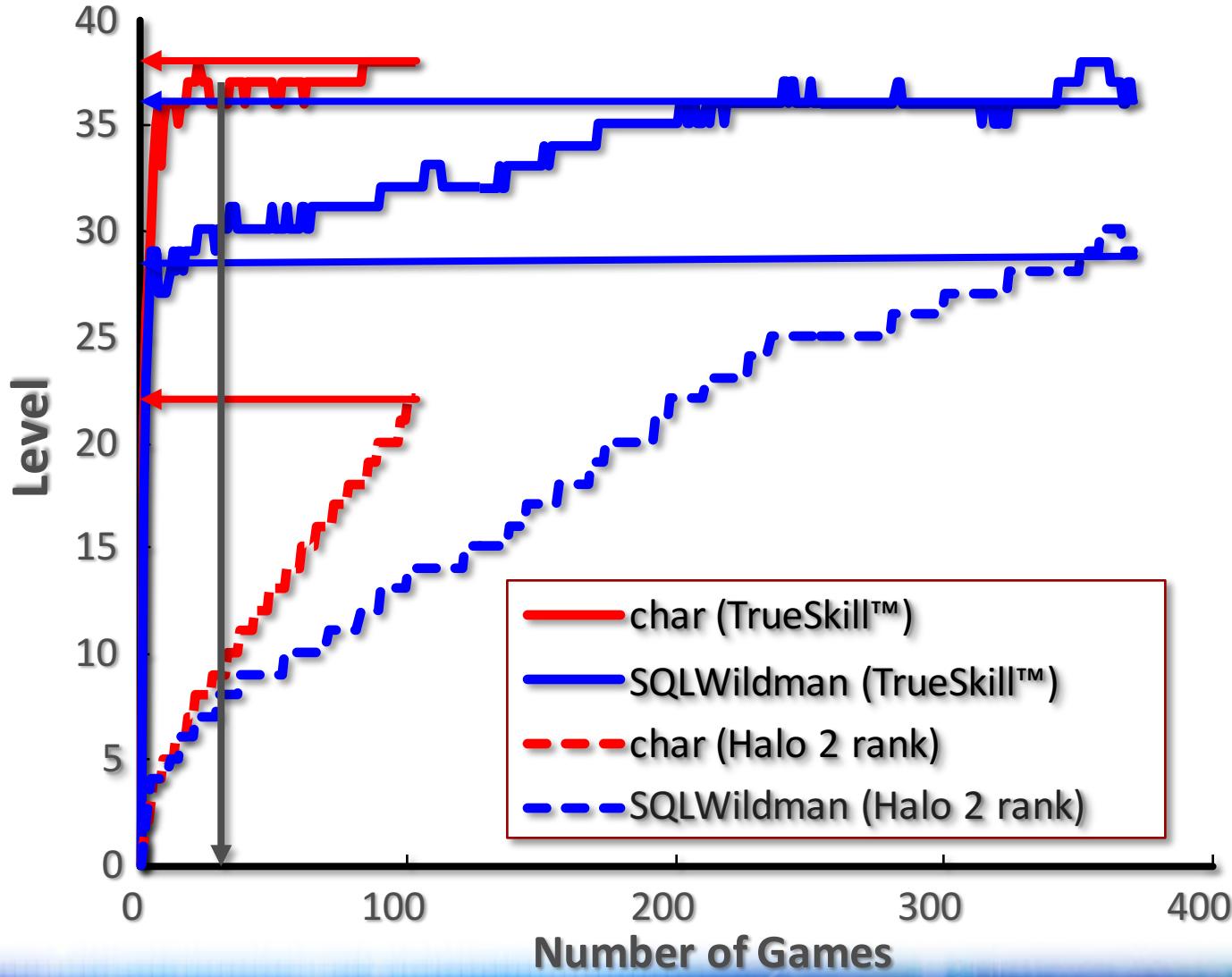
53	53	53	53	53	53	53
53	53	53	53	53	53	53
53	53	53	53	53	53	53

Experimental Setup

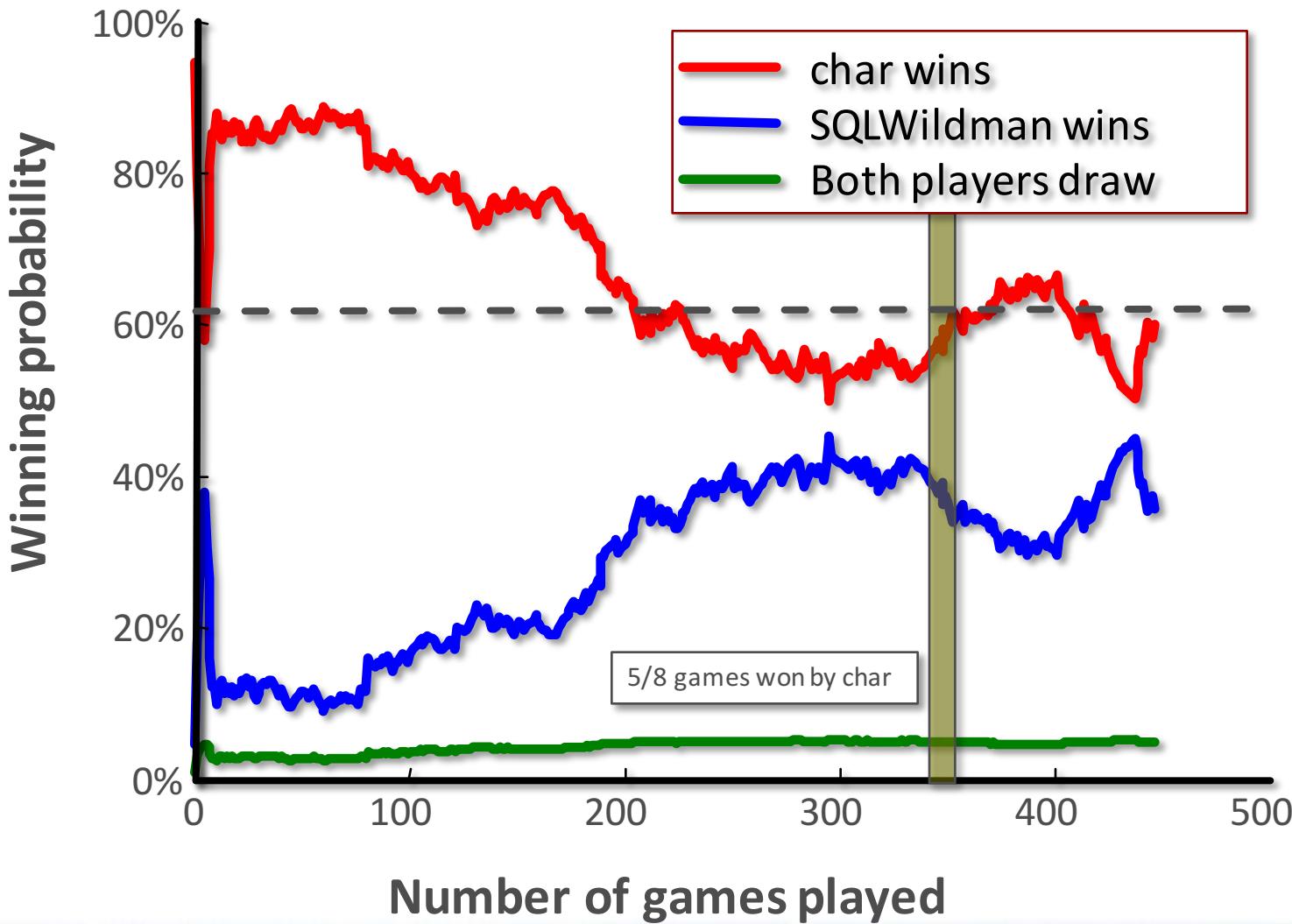
- Data Set: Halo 2 Beta
 - 3 game modes
 - Free-for-All
 - Two Teams
 - 1 vs. 1
 - > 60,000 match outcomes
 - ≈ 6,000 players
 - 6 weeks of game play
 - Publically available



Convergence Speed



Convergence Speed (ctd.)



Xbox 360 & Halo 3

- **Xbox 360 Live**

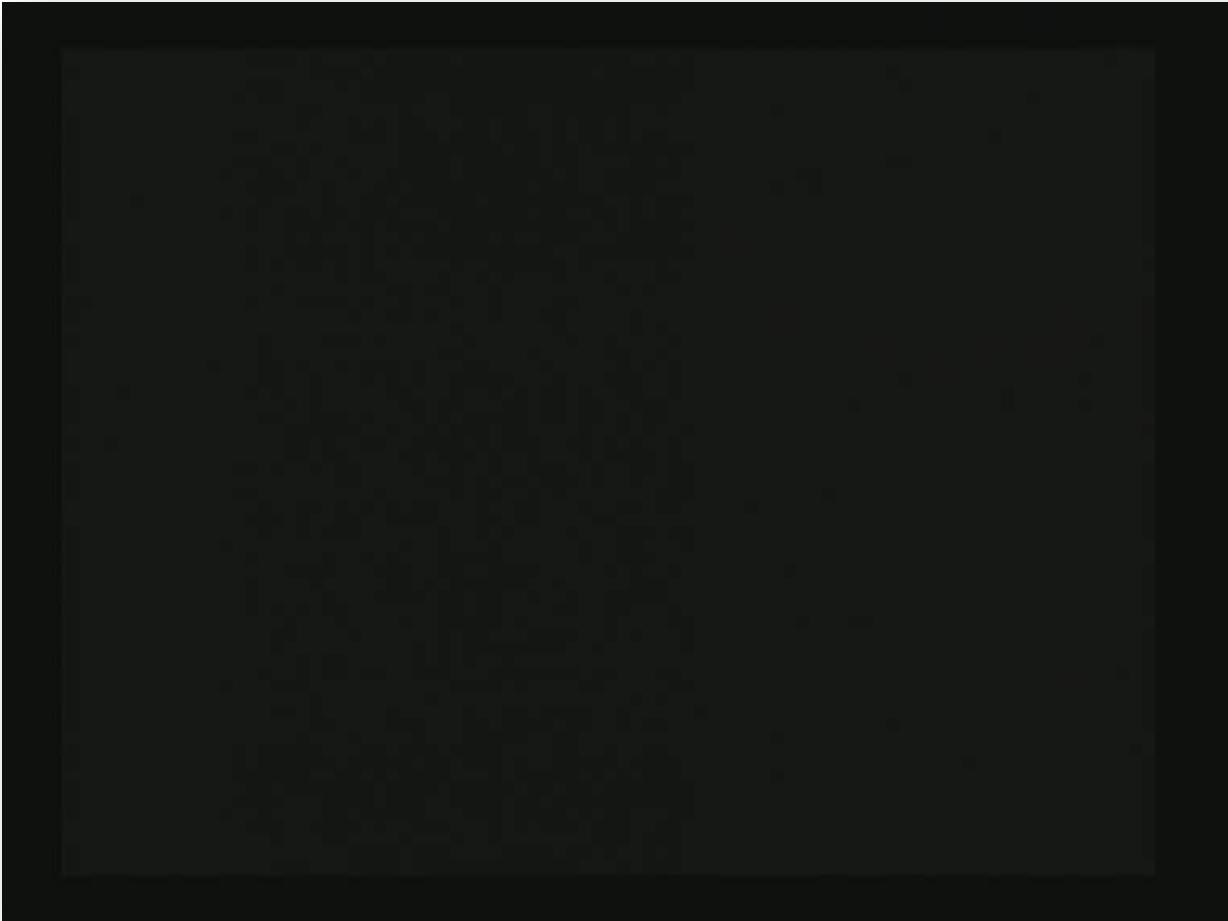
- Launched in September 2005
- Every game uses TrueSkill™ to match players
- > 10 million players
- > 2 million matches per day
- > 2 billion hours of gameplay

- **Halo 3**

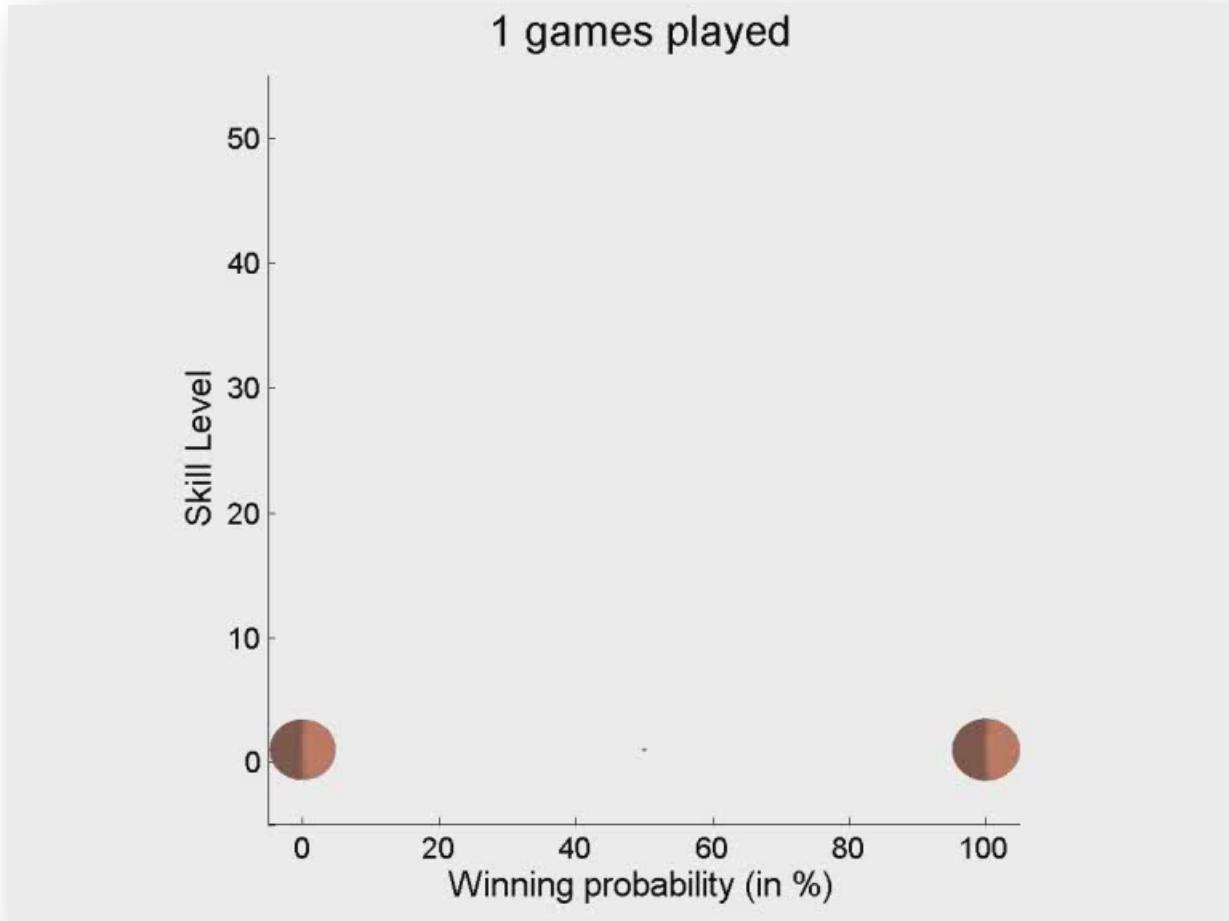
- Launched on 25th September 2007
- Largest entertainment launch in history
- > 200,000 player concurrently (peak: 1,000,000)



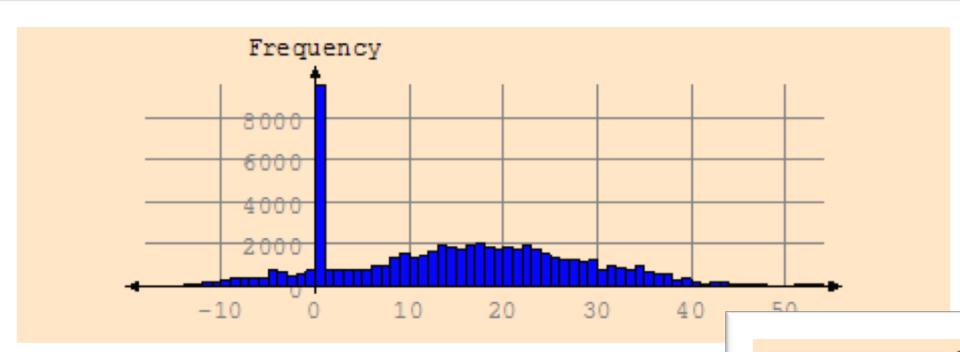
Halo 3 in Action



Halo 3 Public Beta Analysis

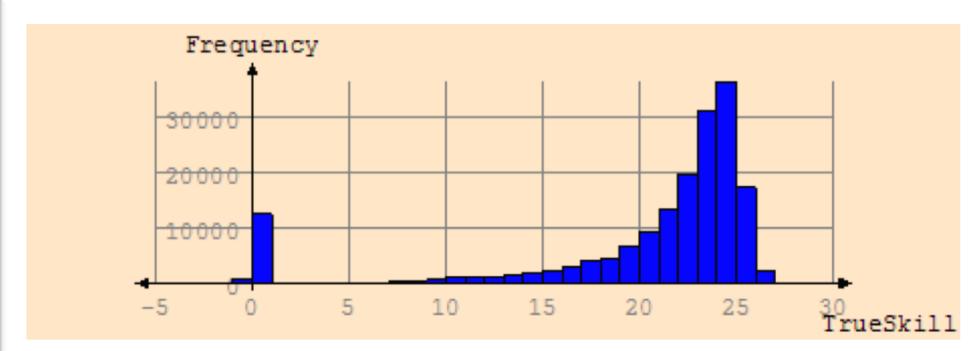
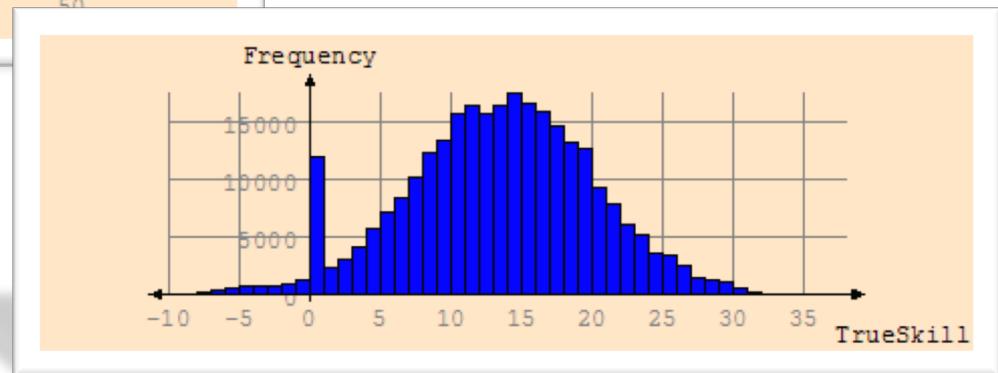


Skill Distributions of Online Games



Car racing (3-4 laps): 40 levels

Golf (18 holes): 60 levels

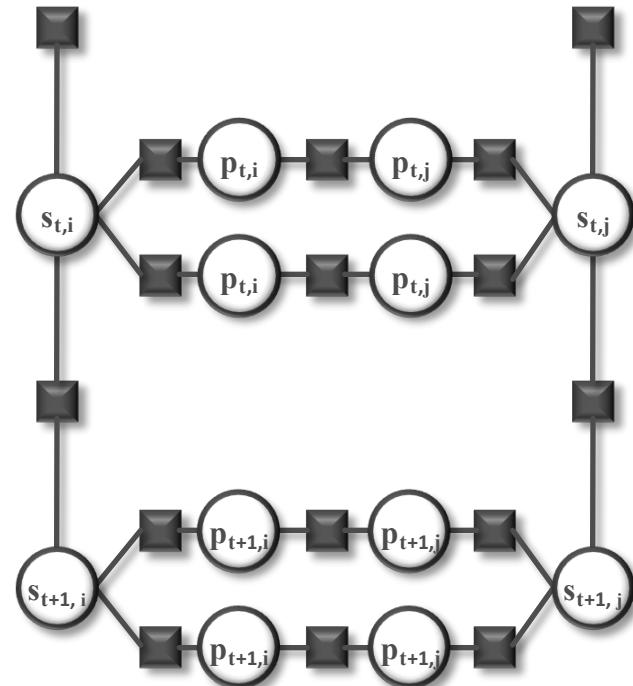


UNO (chance game): 10 levels

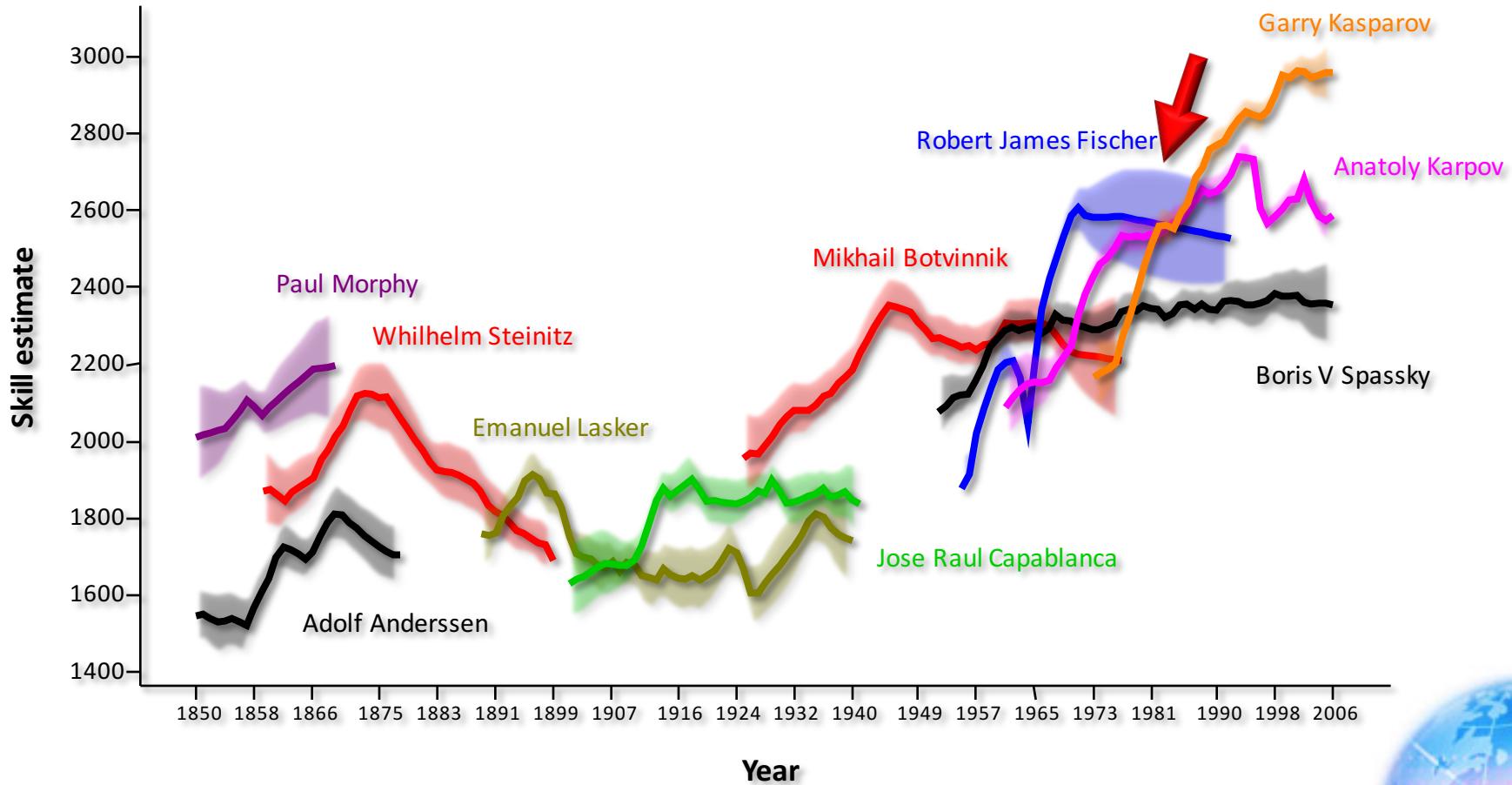


TrueSkill™ Through Time: Chess

- Model time-series of skills by smoothing across time
- History of Chess
 - 3.5M game outcomes (ChessBase)
 - 20 million variables (each of 200,000 players in each year of lifetime + latent variables)
 - 40 million factors



ChessBase Analysis: 1850 - 2006



Thanks!

1000101

101101101010101010001010101011010
01010001010101

10100101101101010101010101000101