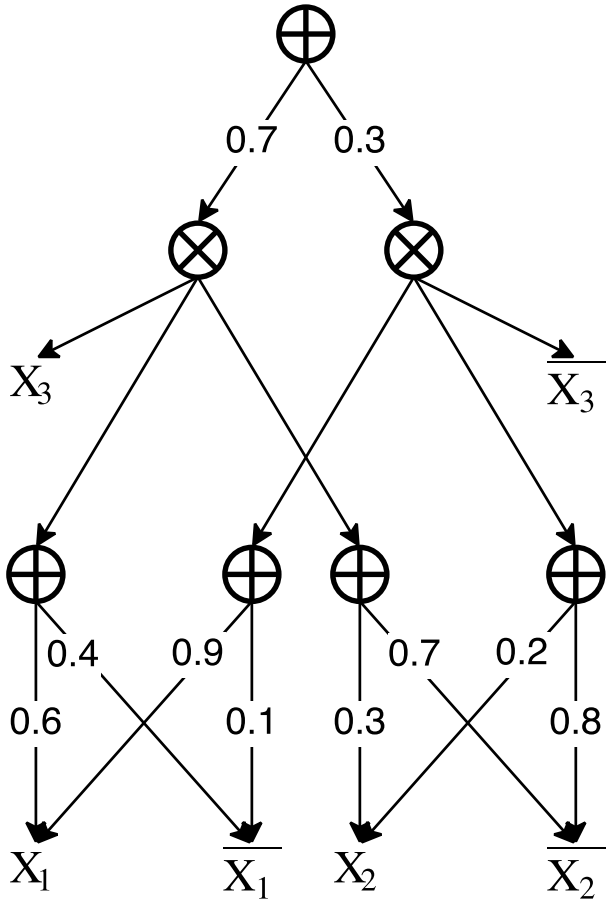
Maximum A Posteriori Inference in Sum-Product Networks

Jun Mei, Yong Jiang, Kewei Tu

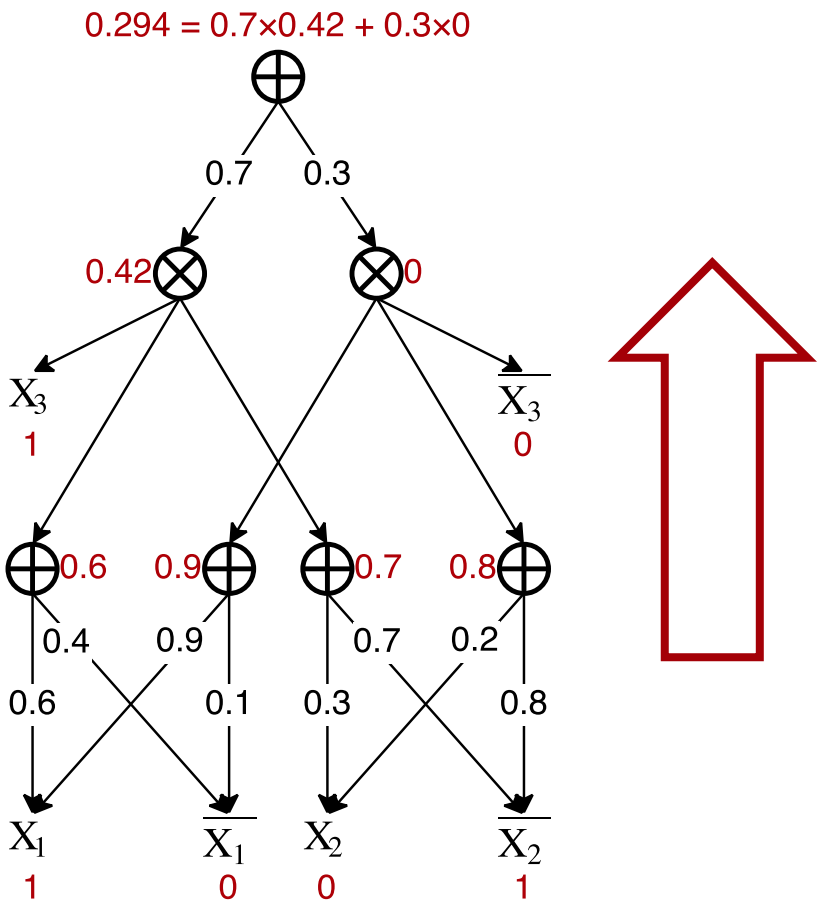
Sum-Product Networks (SPNs)

- A rooted weighted DAG consisting of
  - *leaves (indictors)*,
  - *sum* and *product* nodes.



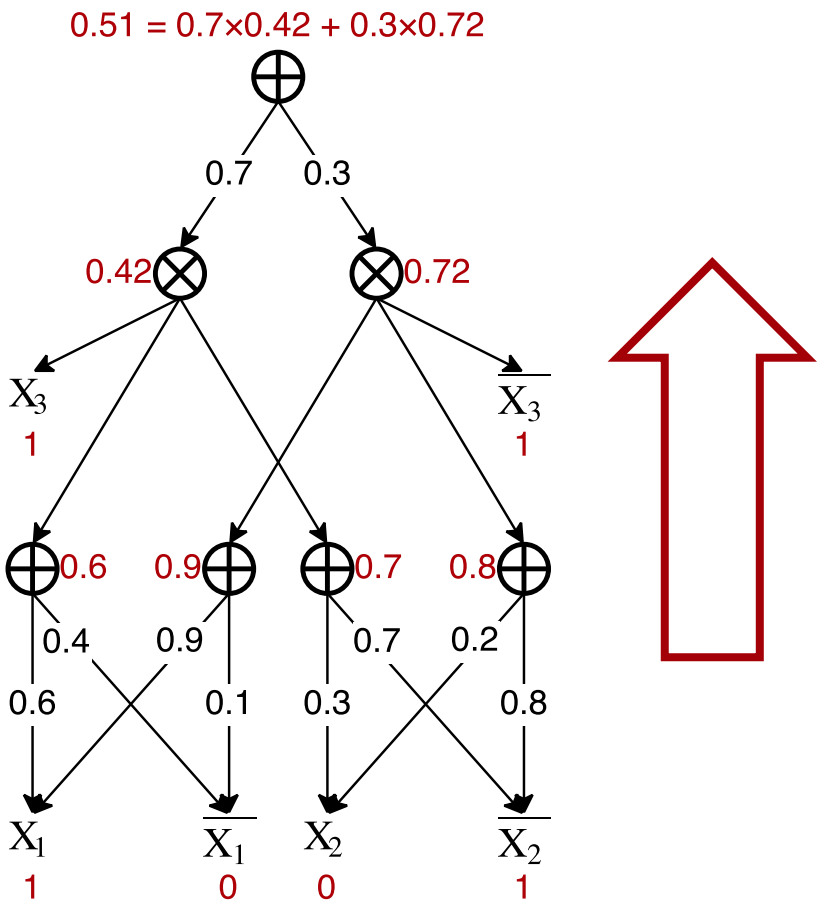
Sum-Product Networks: Probability

- $P(X_1 = 1, X_2 = 0, X_3 = 1)$
- Set
  - $X_1 = 1, \bar{X}_1 = 0$
  - $X_2 = 0, \bar{X}_2 = 1$
  - $X_3 = 1, \bar{X}_3 = 0$



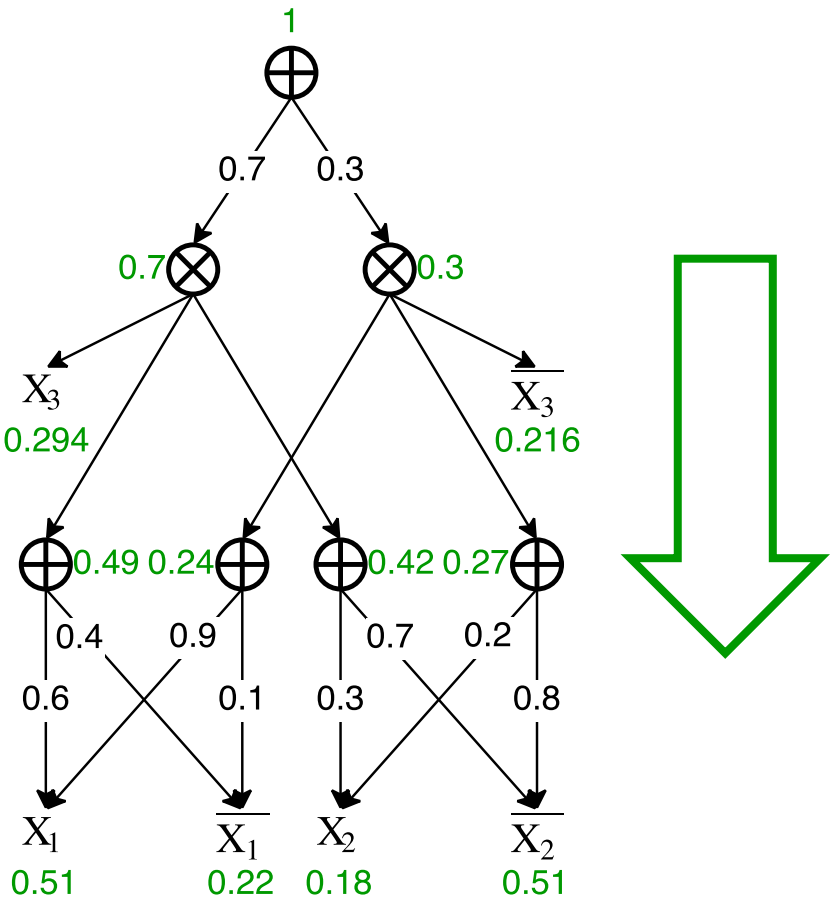
Sum-Product Networks: Marginal

- $P(X_1 = 1, X_2 = 0)$
- Set
  - $X_1 = 1, \bar{X}_1 = 0$
  - $X_2 = 0, \bar{X}_2 = 1$
  - $X_3 = 1, \bar{X}_3 = 1$



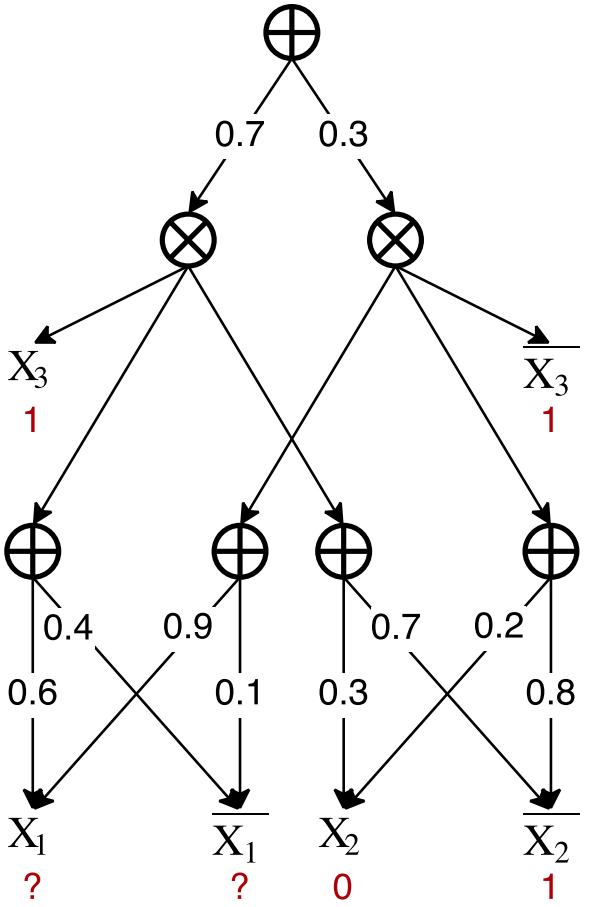
Sum-Product Networks: Differential trick

- $\frac{\partial P(X_1=1,X_2=0)}{\partial X}$
- $\frac{\partial P(X_1=1,X_2=0)}{\partial X_3} = 0.294$
  $= P(X_1 = 1, X_2 = 0, X_3 = 1)$
- $\frac{\partial P(X_1=1,X_2=0)}{\partial \bar{X}_3} = 0.216$
  $= P(X_1 = 1, X_2 = 0, X_3 = 0)$



Maximum A Posteriori Inference (MAP)

- Given Q, e, find
  $\arg\max_q P(Q = q, E = e)$
- Example:
  - $\arg\max_{x_1} P(X_1 = x_1, X_2 = 0)$
  - X_1: Query variable
  - X_2: Evidence variable
  - X_3: Hidden variable



Background

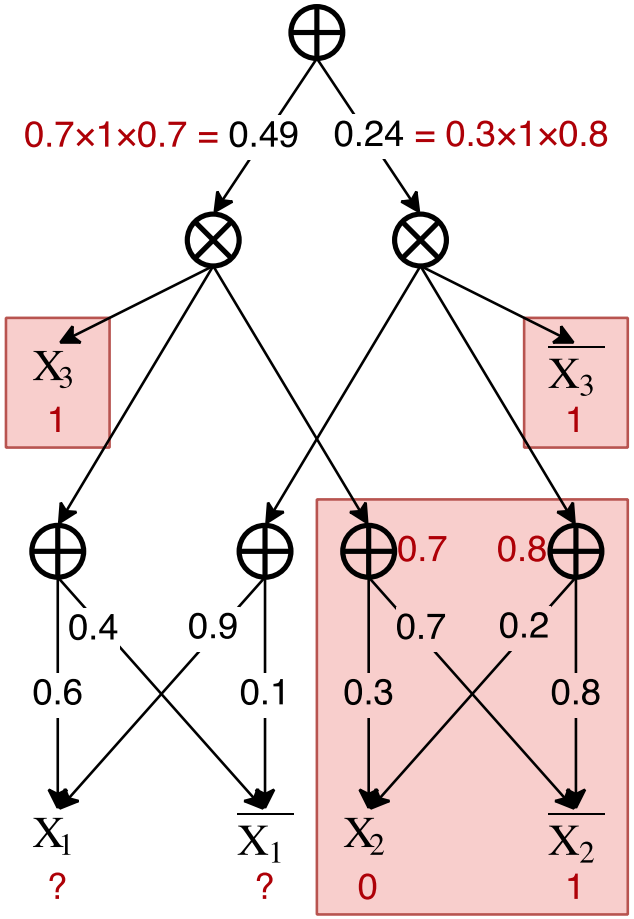
Theoretical Results

MAX Inference

Approximation Complexity

MAX Inference

- $\arg\max_x P(X = x)$
  - No evidence variables
  - No hidden variables
- Reduce MAP to MAX
  1. Bottom-up pre-calculation
  2. Remove nodes in $E \cup H$
- We focus on MAX, because
  - the reduction is in linear time
  - the two SPNs represent the same distribution over Q



Approximation Complexity

- Theorem: It's NP-hard to approximate MAP in SPNs to $2^{|S|^{\delta}}$ for any $0 \leq \delta < 1$.

- Proof sketch:
  - It's NP-hard to approximate MAP in tree-structured Bayesian Networks (BNs) to $2^{|S|^{\epsilon}}$ for any $0 \leq \epsilon < 1$. [De Campos 2011]
  - Given a tree-structured BN, we can construct an SPN in linear size representing the same distribution in linear time.

Background

Theoretical Results

Algorithmic Results

Exact Solver

Approximate Solver

Experiments

Exact Solver

- Basic idea:
  - Reduce MAP to MAX in linear time.
  - Search the whole space but prune any subspace that cannot contain the solution.
- Example:
  - Find $\arg \max_{x_1,x_2,x_3} P(X_1 = x_1, X_2 = x_2, X_3 = x_3)$
  - Try $X_1 = 0$
    - Try $X_2 = 0$
      - Try $X_3 = 0$
      - Try $X_3 = 1$
    - Try $X_2 = 1$ …

Marginal Checking

- Prune a subspace if the samples in it sum up to less than the current best result.
- The summation of samples in a subspace is computed by the marginal inference in SPNs, which can be done in linear time.
- Example:
  - Suppose the current best result is $\hat{x}$
  - Try $X_1 = 1$, prune if $P(X_1 = 1) \leq P(X = \hat{x})$
    - Try $X_2 = 0$, prune if $P(X_1 = 1, X_2 = 0) \leq P(X = \hat{x})$
    - …

Forward Checking

- Check one-step forward.
- Try $X_1 = 1$, check all the following simultaneously in linear time using *differential trick*:
  - $P(X_1 = 1, X_2 = 0) \leq P(X = \hat{x})$?
  - $P(X_1 = 1, X_2 = 1) \leq P(X = \hat{x})$?
  - $P(X_1 = 1, X_3 = 0) \leq P(X = \hat{x})$?
  - $P(X_1 = 1, X_3 = 1) \leq P(X = \hat{x})$?

Ordering

- Q: Which variable should be tried earlier?
- A: The variable with fewer remaining values.

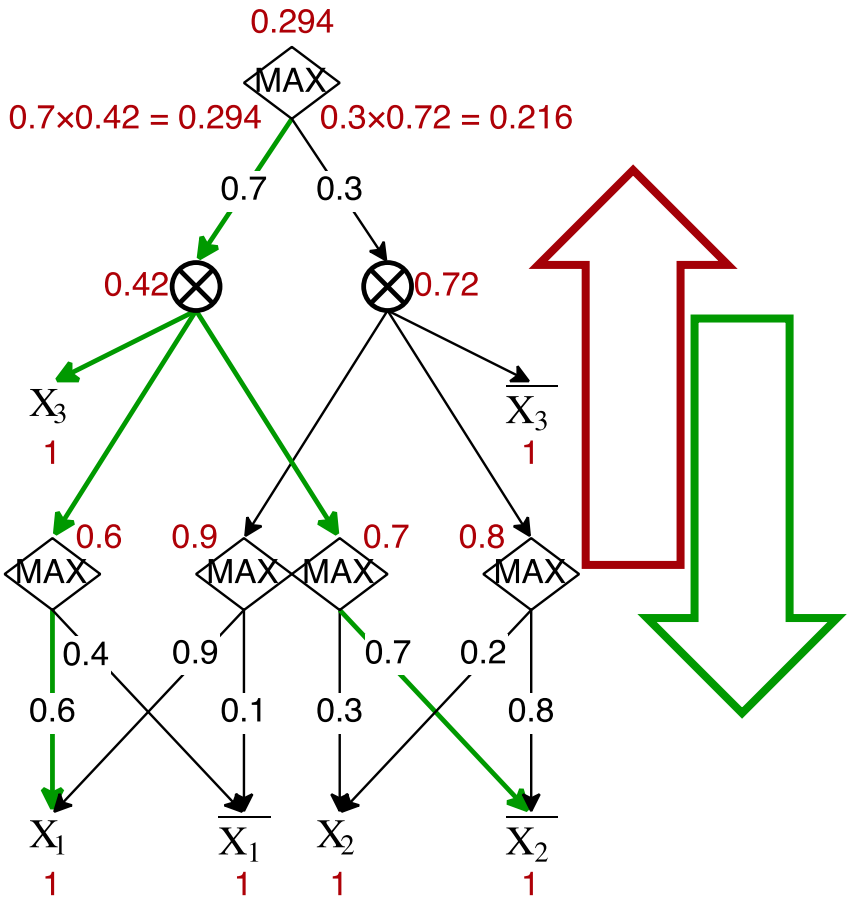
- Q: Which value of a variable should be tried earlier?
- A: The value with higher marginal probability.

Stage

- After setting $X_1 = 1$, the problem becomes $\arg\max_{x_2,x_3} P(X_1 = 1, X_2 = x_2, X_3 = x_3)$.
- This is an MAP inference.
- If we reduce the MAP to MAX, then we decrease the size of the SPN and therefore speed up subsequent computation.

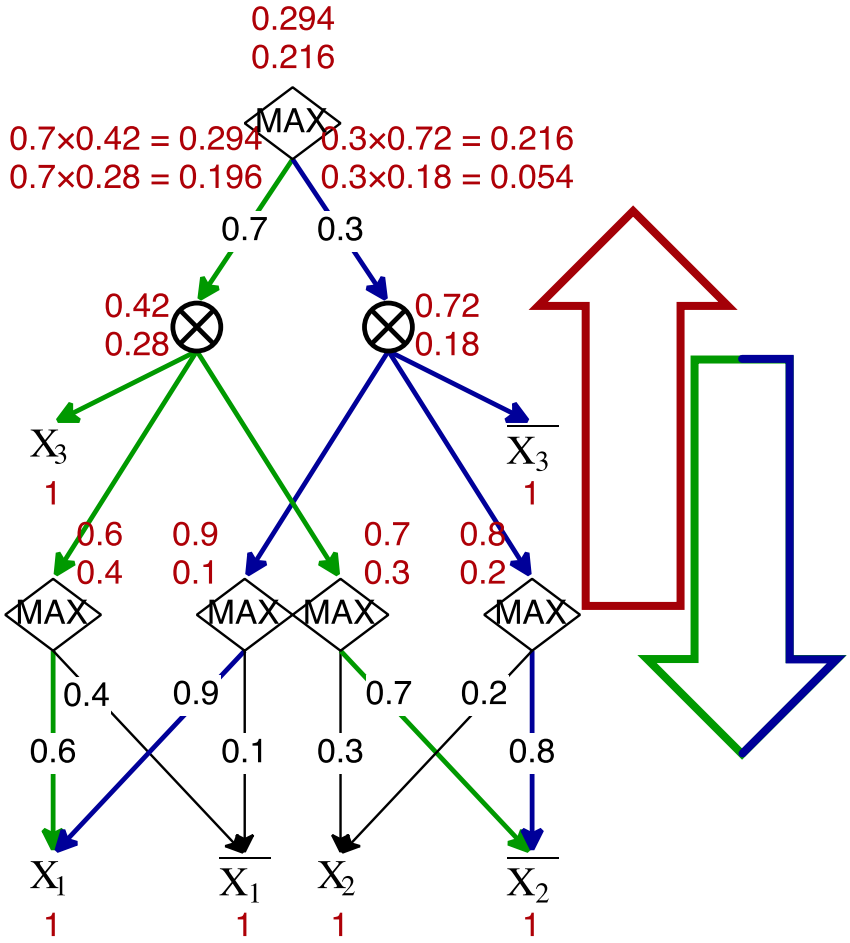
Approximate Solver

- Best Tree Method [Poon and Domingos 2011]
  1. Replace *sum* with *max*
  2. Bottom-up calculation
  3. Then, top-down choosing
- Example:
  - Output is
    - $X_1 = 1, X_2 = 0, X_3 = 1$



K-Best Tree Method

- Find the K best trees and compare their solutions
  - Time complexity is still linear in the SPN size.
- Intuition:
  - The solution often corresponds to trees with large values.
- Example ($K = 2$):
  - Output is
    - $X_1 = 1, X_2 = 0, X_3 = 1$
    - $X_1 = 1, X_2 = 0, X_3 = 0$



Experiments

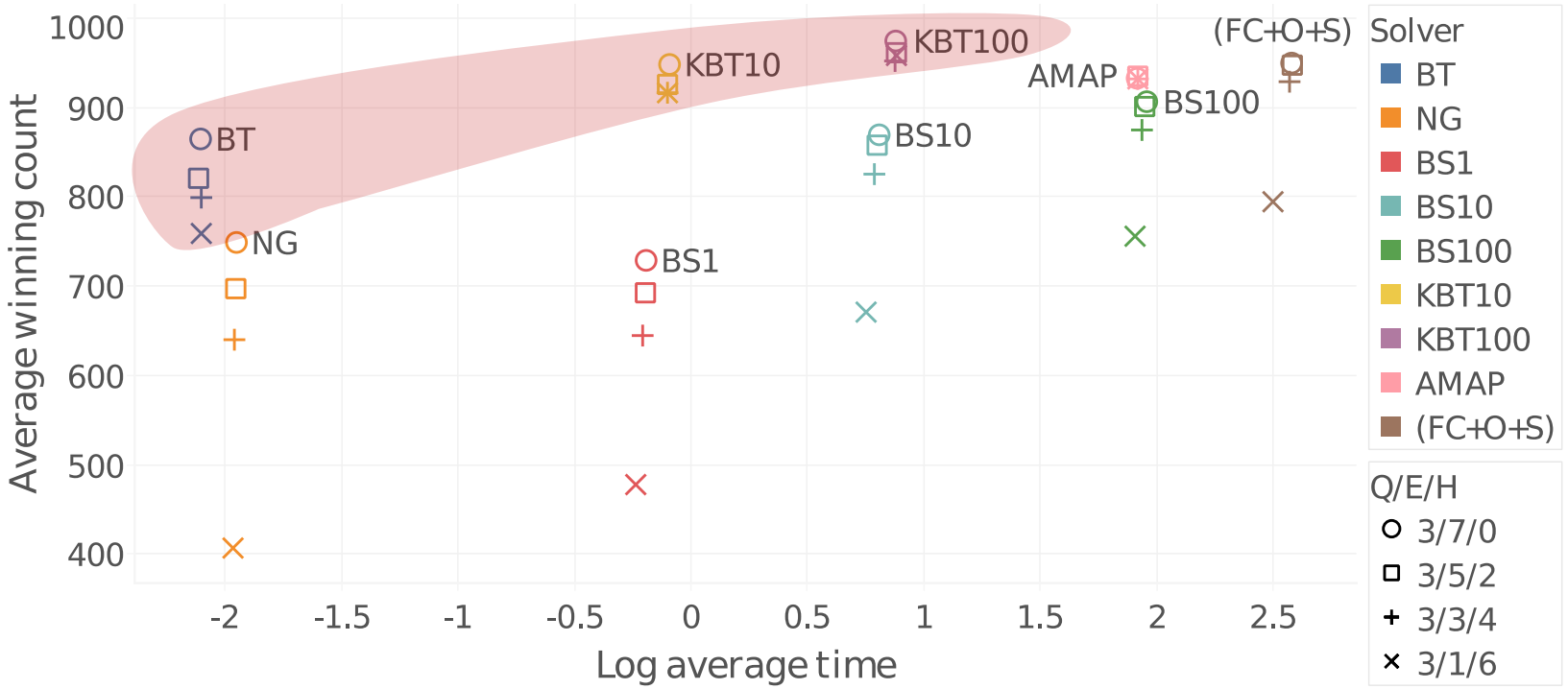
- SPNs are learned by LearnSPN on the 20 real-world datasets [Gens and Domingos 2013].
  - #Variables range from 16 to 1556
  - #Arcs range from 6471 to 2,598,116
- MAP problems are generated randomly.
- Running time is bounded for one MAP problem by 10 minutes.

Experiments: Exact Solver

- Our four techniques significantly speed up the searching process.
- Our best exact solver could handle SPNs with up to 1k variables and 150k arcs in ten minutes.

Experiments: Approximate Solver

- Our K-Best Tree approximate solver has better overall performance than existing methods.



Background

Theoretic Results

Algorithmic Results

Thank you!

Q&A

Code is available at https://github.com/shtechair/maxspn