Tractability in Structured Probability Spaces

Arthur Choi, Yujia Shen, Adnan Darwiche (UCLA)

NIPS, 2017

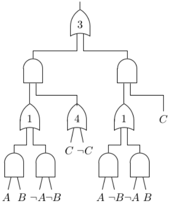
9th July, 2020

PSDD Overview

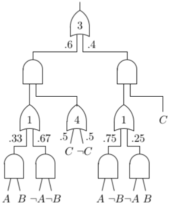
- To represent $Pr(\mathbf{X})$ where $Pr(\mathbf{x}) = 0$ for many $\mathbf{x}$. (Structured Space)

A	B	C	Pr
0	0	0	**0.2**
0	0	1	**0.2**
0	1	0	0.0
0	1	1	**0.1**
1	0	0	0.0
1	0	1	**0.3**
1	1	0	**0.1**
1	1	1	**0.1**

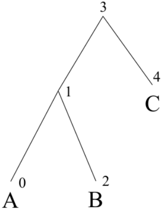
(a) Distribution



(b) An SDD



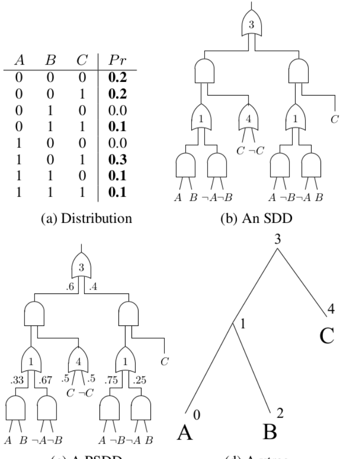
(c) A PSDD



(d) A vtree

PSDD Overview

- To represent $Pr(\mathbf{X})$ where $Pr(\mathbf{x}) = 0$ for many $\mathbf{x}$. (Structured Space)
- First step in construction: Construct a Boolean circuit (SDD) that captures the zero entries of the distribution.



A	B	C	Pr
0	0	0	**0.2**
0	0	1	**0.2**
0	1	0	0.0
0	1	1	**0.1**
1	0	0	0.0
1	0	1	**0.3**
1	1	0	**0.1**
1	1	1	**0.1**

(a) Distribution (b) An SDD

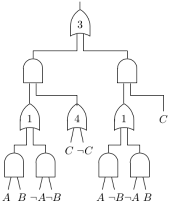
(c) A PSDD (d) A vtree

PSDD Overview

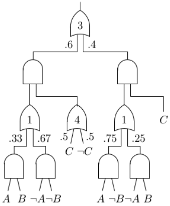
- To represent $Pr(\mathbf{X})$ where $Pr(\mathbf{x}) = 0$ for many $\mathbf{x}$. (Structured Space)
- First step in construction: Construct a Boolean circuit (SDD) that captures the zero entries of the distribution.
- Second step: Parameterize SDD, which induces a local distribution on the inputs of OR gates.

A	B	C	Pr
0	0	0	**0.2**
0	0	1	**0.2**
0	1	0	0.0
0	1	1	**0.1**
1	0	0	0.0
1	0	1	**0.3**
1	1	0	**0.1**
1	1	1	**0.1**

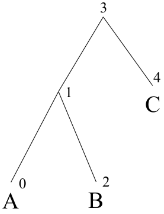
(a) Distribution



(b) An SDD



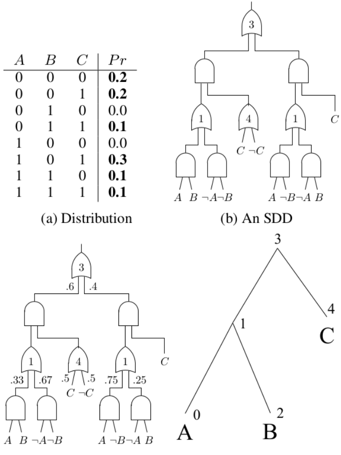
(c) A PSDD



(d) A vtree

PSDD Overview

- To represent $Pr(\mathbf{X})$ where $Pr(\mathbf{x}) = 0$ for many $\mathbf{x}$. (Structured Space)
- First step in construction: Construct a Boolean circuit (SDD) that captures the zero entries of the distribution.
- Second step: Parameterize SDD, which induces a local distribution on the inputs of OR gates.
- The probability of a complete instantiation x: Perform a bottom-up pass. Value of AND gate is product of its inputs. Value of OR gate is weighted sum of its inputs. Also, $\sum_{\mathbf{x}} Pr(\mathbf{x}) = 1$



A	B	C	Pr
0	0	0	**0.2**
0	0	1	**0.2**
0	1	0	0.0
0	1	1	**0.1**
1	0	0	0.0
1	0	1	**0.3**
1	1	0	**0.1**
1	1	1	**0.1**

(a) Distribution (b) An SDD

(c) A PSDD (d) A vtree

Route Distributions

- The setting: For an undirected graph G, $\mathbf{X}$ is a set of binary variables which correspond to edges in G. Instantiation $\mathbf{x}$ includes edge e iff the edge variable is set to true in $\mathbf{x}$.

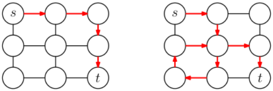
Route Distributions

- The setting: For an undirected graph G, $\mathbf{X}$ is a set of binary variables which correspond to edges in G. Instantiation $\mathbf{x}$ includes edge e iff the edge variable is set to true in $\mathbf{x}$.
- $\alpha_G = \mathbf{x_1} \vee \mathbf{x_2} \ldots$, where $\mathbf{x_i}$ correspond to routes in G.
  $Pr(\mathbf{X})$: **route distribution** iff $Pr(\mathbf{x}) = 0$ if $\mathbf{x} \not\models \alpha_G$

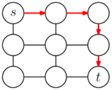
Route Distributions

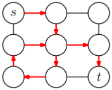
- The setting: For an undirected graph G, $\mathbf{X}$ is a set of binary variables which correspond to edges in G. Instantiation $\mathbf{x}$ includes edge e iff the edge variable is set to true in $\mathbf{x}$.
- $\alpha_G = \mathbf{x_1} \vee \mathbf{x_2} \ldots$, where $\mathbf{x_i}$ correspond to routes in G.
  $Pr(\mathbf{X})$: **route distribution** iff $Pr(\mathbf{x}) = 0$ if $\mathbf{x} \not\models \alpha_G$
- Simple Routes: No-loop paths in G. $\beta_G = \mathbf{x_1} \vee \mathbf{x_2} \ldots$, where $\mathbf{x_i}$ correspond to *simple routes* in G. Then, $\beta_G \models \alpha_G$
  Simple-route distribution: $Pr(\mathbf{x}) = 0$ if $x \not\models \beta_G$

Route Distributions

- The setting: For an undirected graph G, $\mathbf{X}$ is a set of binary variables which correspond to edges in G. Instantiation $\mathbf{x}$ includes edge e iff the edge variable is set to true in $\mathbf{x}$.
- $\alpha_G = \mathbf{x_1} \vee \mathbf{x_2} \ldots$, where $\mathbf{x_i}$ correspond to routes in G.
  $Pr(\mathbf{X})$: **route distribution** iff $Pr(\mathbf{x}) = 0$ if $\mathbf{x} \not\models \alpha_G$
- Simple Routes: No-loop paths in G. $\beta_G = \mathbf{x_1} \vee \mathbf{x_2} \ldots$, where $\mathbf{x_i}$ correspond to *simple routes* in G. Then, $\beta_G \models \alpha_G$
  Simple-route distribution: $Pr(\mathbf{x}) = 0$ if $x \not\models \beta_G$
- 

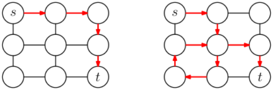
Route Distributions

- The setting: For an undirected graph G, $\mathbf{X}$ is a set of binary variables which correspond to edges in G. Instantiation $\mathbf{x}$ includes edge e iff the edge variable is set to true in $\mathbf{x}$.
- $\alpha_G = \mathbf{x_1} \vee \mathbf{x_2} \ldots$, where $\mathbf{x_i}$ correspond to routes in G.
  $Pr(\mathbf{X})$: **route distribution** iff $Pr(\mathbf{x}) = 0$ if $\mathbf{x} \not\models \alpha_G$
- Simple Routes: No-loop paths in G. $\beta_G = \mathbf{x_1} \vee \mathbf{x_2} \ldots$, where $\mathbf{x_i}$ correspond to *simple routes* in G. Then, $\beta_G \models \alpha_G$
  Simple-route distribution: $Pr(\mathbf{x}) = 0$ if $x \not\models \beta_G$
- 



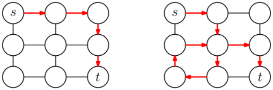
- To learn simple-route distributions using PSDD, compile β_G into an SDD and parameterize it.

Route Distributions

- The setting: For an undirected graph G, $\mathbf{X}$ is a set of binary variables which correspond to edges in G. Instantiation $\mathbf{x}$ includes edge e iff the edge variable is set to true in $\mathbf{x}$.
- $\alpha_G = \mathbf{x_1} \vee \mathbf{x_2} \ldots$, where $\mathbf{x_i}$ correspond to routes in G.
  $Pr(\mathbf{X})$: **route distribution** iff $Pr(\mathbf{x}) = 0$ if $\mathbf{x} \not\models \alpha_G$
- Simple Routes: No-loop paths in G. $\beta_G = \mathbf{x_1} \vee \mathbf{x_2} \ldots$, where $\mathbf{x_i}$ correspond to *simple routes* in G. Then, $\beta_G \models \alpha_G$
  Simple-route distribution: $Pr(\mathbf{x}) = 0$ if $x \not\models \beta_G$
- 

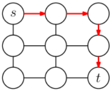
- To learn simple-route distributions using PSDD, compile β_G into an SDD and parameterize it.
- Scalability: Simple routes in graphs with as many as 100 nodes and 140 edges can be compiled. But, to handle larger problems, we can:

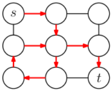
Route Distributions

- The setting: For an undirected graph G, $\mathbf{X}$ is a set of binary variables which correspond to edges in G. Instantiation $\mathbf{x}$ includes edge e iff the edge variable is set to true in $\mathbf{x}$.
- $\alpha_G = \mathbf{x_1} \vee \mathbf{x_2} \ldots$, where $\mathbf{x_i}$ correspond to routes in G.
  $Pr(\mathbf{X})$: **route distribution** iff $Pr(\mathbf{x}) = 0$ if $\mathbf{x} \not\models \alpha_G$
- Simple Routes: No-loop paths in G. $\beta_G = \mathbf{x_1} \vee \mathbf{x_2} \ldots$, where $\mathbf{x_i}$ correspond to *simple routes* in G. Then, $\beta_G \models \alpha_G$
  Simple-route distribution: $Pr(\mathbf{x}) = 0$ if $x \not\models \beta_G$
- 

- To learn simple-route distributions using PSDD, compile β_G into an SDD and parameterize it.
- Scalability: Simple routes in graphs with as many as 100 nodes and 140 edges can be compiled. But, to handle larger problems, we can:
  - advance the current SDD compilation technology, or

Route Distributions

- The setting: For an undirected graph G, $\mathbf{X}$ is a set of binary variables which correspond to edges in G. Instantiation $\mathbf{x}$ includes edge e iff the edge variable is set to true in $\mathbf{x}$.
- $\alpha_G = \mathbf{x_1} \vee \mathbf{x_2} \ldots$, where $\mathbf{x_i}$ correspond to routes in G.
  $Pr(\mathbf{X})$: **route distribution** iff $Pr(\mathbf{x}) = 0$ if $\mathbf{x} \not\models \alpha_G$
- Simple Routes: No-loop paths in G. $\beta_G = \mathbf{x_1} \vee \mathbf{x_2} \ldots$, where $\mathbf{x_i}$ correspond to *simple routes* in G. Then, $\beta_G \models \alpha_G$
  Simple-route distribution: $Pr(\mathbf{x}) = 0$ if $x \not\models \beta_G$
- 



- To learn simple-route distributions using PSDD, compile β_G into an SDD and parameterize it.
- Scalability: Simple routes in graphs with as many as 100 nodes and 140 edges can be compiled. But, to handle larger problems, we can:
  - advance the current SDD compilation technology, or
  - use hierarchical maps and distributions.

Hierarchical Route Distributions

- A route distribution can be represented hierarchically if we impose hierarchy on the underlying graph.

Hierarchical Route Distributions

- A route distribution can be represented hierarchically if we impose hierarchy on the underlying graph.
- **Hierarchical maps**: Partition nodes of G as $N_1, \ldots N_m$ into m *regions/clusters*. These regions partition edges $\mathbf{X}$ into

Hierarchical Route Distributions

- A route distribution can be represented hierarchically if we impose hierarchy on the underlying graph.
- **Hierarchical maps**: Partition nodes of G as $N_1, \ldots N_m$ into m *regions/clusters*. These regions partition edges $\mathbf{X}$ into
  - $\mathbf{B}$: Edges crossing the regions.

Hierarchical Route Distributions

- A route distribution can be represented hierarchically if we impose hierarchy on the underlying graph.
- **Hierarchical maps**: Partition nodes of G as $N_1, \ldots N_m$ into m *regions/clusters*. These regions partition edges $\mathbf{X}$ into
  - $\mathbf{B}$: Edges crossing the regions.
  - $\mathbf{A_1}, \ldots, \mathbf{A_m}$: Edges inside a region.

Hierarchical Route Distributions

- A route distribution can be represented hierarchically if we impose hierarchy on the underlying graph.
- **Hierarchical maps**: Partition nodes of G as $N_1, \ldots N_m$ into m *regions/clusters*. These regions partition edges $\mathbf{X}$ into
  - $\mathbf{B}$: Edges crossing the regions.
  - $\mathbf{A_1}, \ldots, \mathbf{A_m}$: Edges inside a region.
- Represent $Pr(X)$ using a set of smaller route distributions, *Decomposable route distribution*: $Pr(\mathbf{x}) = Pr(\mathbf{b}) \prod_{i=1}^{m} Pr(\mathbf{a_i}|\mathbf{b_i})$

Hierarchical Route Distributions

- A route distribution can be represented hierarchically if we impose hierarchy on the underlying graph.
- **Hierarchical maps**: Partition nodes of G as $N_1, \ldots N_m$ into m *regions/clusters*. These regions partition edges $\mathbf{X}$ into
  - $\mathbf{B}$: Edges crossing the regions.
  - $\mathbf{A_1}, \ldots, \mathbf{A_m}$: Edges inside a region.
- Represent $Pr(X)$ using a set of smaller route distributions, *Decomposable route distribution*: $Pr(\mathbf{x}) = Pr(\mathbf{b}) \prod_{i=1}^{m} Pr(\mathbf{a_i}|\mathbf{b_i})$
- Graph G_B: Each N_i is a single node.
  Subgraph G_{b_i}: From G, keep edges $\mathbf{A_i}$ and the edges set positively in b_i (used to enter and exit N_i). Local map for region i.
  So, G_B is an abstraction of G and G_{b_i} are subsets of G.

Hierarchical Route Distributions

- A route distribution can be represented hierarchically if we impose hierarchy on the underlying graph.
- **Hierarchical maps**: Partition nodes of G as $N_1, \ldots N_m$ into m *regions/clusters*. These regions partition edges $\mathbf{X}$ into
  - $\mathbf{B}$: Edges crossing the regions.
  - $\mathbf{A_1}, \ldots, \mathbf{A_m}$: Edges inside a region.
- Represent $Pr(X)$ using a set of smaller route distributions, *Decomposable route distribution*: $Pr(\mathbf{x}) = Pr(\mathbf{b}) \prod_{i=1}^{m} Pr(\mathbf{a_i}|\mathbf{b_i})$
- Graph G_B: Each N_i is a single node.
  Subgraph G_{b_i}: From G, keep edges $\mathbf{A_i}$ and the edges set positively in b_i (used to enter and exit N_i). Local map for region i.
  So, G_B is an abstraction of G and G_{b_i} are subsets of G.
- 

Hierarchical Simple Routes

- $Pr(\mathbf{B})$: Captures routes across regions; $Pr(\mathbf{A_i}|\mathbf{b_i})$: Capture routes within a region. So, total distributions $= 1 + \sum_{i=1}^{m} 2^{|B_i|}$

Hierarchical Simple Routes

- $Pr(\mathbf{B})$: Captures routes across regions; $Pr(\mathbf{A_i}|\mathbf{b_i})$: Capture routes within a region. So, total distributions $= 1 + \sum_{i=1}^{m} 2^{|B_i|}$
- $\gamma_G =$ Boolean expression obtained by disjoining $\mathbf{x}$ that correspond to simple routes that are also simple w.r.t G_B. Then γ_G. Then $\gamma_G \models \beta_G \models \alpha_G$.

Hierarchical Simple Routes

- $Pr(\mathbf{B})$: Captures routes across regions; $Pr(\mathbf{A_i}|\mathbf{b_i})$: Capture routes within a region. So, total distributions $= 1 + \sum_{i=1}^{m} 2^{|B_i|}$
- γ_G = Boolean expression obtained by disjoining $\mathbf{x}$ that correspond to simple routes that are also simple w.r.t G_B. Then γ_G. Then $\gamma_G \models \beta_G \models \alpha_G$.
- Hierarchical simple route distribution: If $Pr(\mathbf{B})$ represents simple route distribution for G_B and $Pr(\mathbf{A_i}|\mathbf{b_i})$ represent simple route distribution for G_{b_i}, then $Pr(\mathbf{X})$ is a simple route distribution for G.

Hierarchical Simple Routes

- $Pr(\mathbf{B})$: Captures routes across regions; $Pr(\mathbf{A_i}|\mathbf{b_i})$: Capture routes within a region. So, total distributions $= 1 + \sum_{i=1}^{m} 2^{|B_i|}$
- $\gamma_G =$ Boolean expression obtained by disjoining $\mathbf{x}$ that correspond to simple routes that are also simple w.r.t G_B. Then γ_G. Then $\gamma_G \models \beta_G \models \alpha_G$.
- Hierarchical simple route distribution: If $Pr(\mathbf{B})$ represents simple route distribution for G_B and $Pr(\mathbf{A_i}|\mathbf{b_i})$ represent simple route distribution for G_{b_i}, then $Pr(\mathbf{X})$ is a simple route distribution for G.
- $Pr(x) = 0$ if $x \not\models \gamma_G$.

Hierarchical Simple Routes

- $Pr(\mathbf{B})$: Captures routes across regions; $Pr(\mathbf{A_i}|\mathbf{b_i})$: Capture routes within a region. So, total distributions $= 1 + \sum_{i=1}^{m} 2^{|B_i|}$
- $\gamma_G =$ Boolean expression obtained by disjoining $\mathbf{x}$ that correspond to simple routes that are also simple w.r.t G_B. Then γ_G. Then $\gamma_G \models \beta_G \models \alpha_G$.
- Hierarchical simple route distribution: If $Pr(\mathbf{B})$ represents simple route distribution for G_B and $Pr(\mathbf{A_i}|\mathbf{b_i})$ represent simple route distribution for G_{b_i}, then $Pr(\mathbf{X})$ is a simple route distribution for G.
- $Pr(x) = 0$ if $x \not\models \gamma_G$.
- If more than 2 variables of $\mathbf{B_i}$ are true in some x, then $Pr(x) = 0$.

Some Results

- Hierarchical simple-route distribution can be represented by a data structure with size $O(2^{|B|} + \sum_{i=1}^{m} 2^{|A_i|}|B_i|^2)$

Some Results

- Hierarchical simple-route distribution can be represented by a data structure with size $O(2^{|B|} + \sum_{i=1}^{m} 2^{|A_i|}|B_i|^2)$
- Represent $Pr(B)$ and $Pr(A_i|b_i)$ using PSDDs.

Some Results

- Hierarchical simple-route distribution can be represented by a data structure with size $O(2^{|B|} + \sum_{i=1}^{m} 2^{|A_i|}|B_i|^2)$
- Represent $Pr(B)$ and $Pr(A_i|b_i)$ using PSDDs.
- Let $Pr(\mathbf{X})$ be decomposable route distribution, $Pr(\mathbf{X}|\gamma_G)$ be a hierarchical simple-route distribution, α be a query. Then the error of the query $Pr(\alpha|\gamma_G)$ rel. to $Pr(\alpha)$ is

$$\frac{Pr(\alpha|\gamma_G) - Pr(\alpha)}{Pr(\alpha|\gamma_G)} = Pr(\kappa_G)\left[1 - \frac{Pr(\alpha|\kappa_G)}{Pr(\alpha|\gamma_G)}\right]$$

where $\kappa_G = \beta_G \wedge \neg\gamma_G$.

Some Results

- Hierarchical simple-route distribution can be represented by a data structure with size $O(2^{|B|} + \sum_{i=1}^{m} 2^{|A_i|}|B_i|^2)$
- Represent $Pr(B)$ and $Pr(A_i|b_i)$ using PSDDs.
- Let $Pr(\mathbf{X})$ be decomposable route distribution, $Pr(\mathbf{X}|\gamma_G)$ be a hierarchical simple-route distribution, α be a query. Then the error of the query $Pr(\alpha|\gamma_G)$ rel. to $Pr(\alpha)$ is

$$\frac{Pr(\alpha|\gamma_G) - Pr(\alpha)}{Pr(\alpha|\gamma_G)} = Pr(\kappa_G)\left[1 - \frac{Pr(\alpha|\kappa_G)}{Pr(\alpha|\gamma_G)}\right]$$

  where $\kappa_G = \beta_G \wedge \neg\gamma_G$.
- When simple routes are also simple in G_B? Since $Pr(\gamma_G) + Pr(\kappa_G) = 1$, then if $Pr(\gamma_G) \approx 1$, then we expect the hierarchical distribution to be accurate.

Compiling Routes

- To compile a PSDD for hierarchical simple routes in G:

Compiling Routes

- To compile a PSDD for hierarchical simple routes in G:
  - First compile SDD for each N_i, taking edges A_i and B_i

Compiling Routes

- To compile a PSDD for hierarchical simple routes in G:
  - First compile SDD for each N_i, taking edges A_i and B_i
  - Then compile an SDD representing simple routes of the abstracted graph G_B.

Compiling Routes

- To compile a PSDD for hierarchical simple routes in G:
  - First compile SDD for each N_i, taking edges A_i and B_i
  - Then compile an SDD representing simple routes of the abstracted graph G_B.
  - Parameterize all the SDDs to get $m + 1$ PSDDs.

Compiling Routes

- To compile a PSDD for hierarchical simple routes in G:
  - First compile SDD for each N_i, taking edges A_i and B_i
  - Then compile an SDD representing simple routes of the abstracted graph G_B.
  - Parameterize all the SDDs to get $m + 1$ PSDDs.
  - Multiply all the component PSDDs to get a single PSDD over the structured space of hierarchical simple-routes.