Learning chordal Markov networks by dynamic programming

Kustaa Kangas

Teppo Niinimäki
Mikko Koivisto

NIPS 2014 (to appear)

November 27, 2014

Probabilistic graphical models

Graphical model

- Graph structure $\mathcal{G}$ on the vertex set $V = \{1, \ldots, n\}$
- Represents conditional independencies in a joint distribution $p(X) = p(X_1, \ldots, X_n)$

Advantages

- Easy to read
- Compact way to store a distribution
- Efficient inference

Probabilistic graphical models

Directed models: Bayesian networks, ...

Undirected models: Markov networks, ...

Structure learning problem: Given samples from $p(X_1, \ldots, X_n)$, find a model that best fits the sampled data.

Probabilistic graphical models

Structure learning in chordal Markov networks: Find a chordal Markov network that maximizes a given decomposable score.

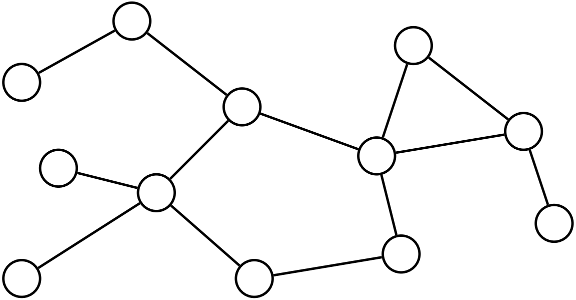
Prior work:

- Constraint satisfaction, Corander et al.
- Integer linear programming, Bartlett and Cussens

Our result: Dynamic programming in $O(4^n)$ time and $O(3^n)$ space for n variables.

- First non-trivial bound
- Competitive in practice

Markov networks

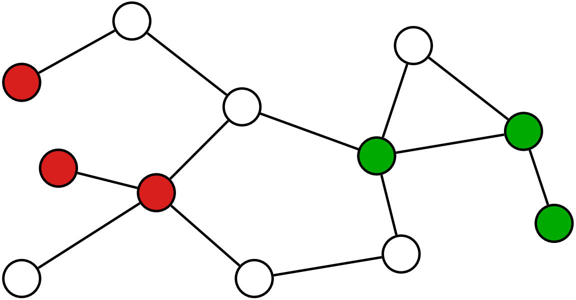


- Joint distribution $p(X) = p(X_1, \ldots X_n)$
- Undirected graph $\mathcal{G}$ on $V = \{1, \ldots, n\}$ with the Global Markov property: For $A, B, S \subseteq V$ it holds that

$$X_A \perp\!\!\!\perp X_B \mid X_S$$

if S separates A and B in $\mathcal{G}$.

Markov networks

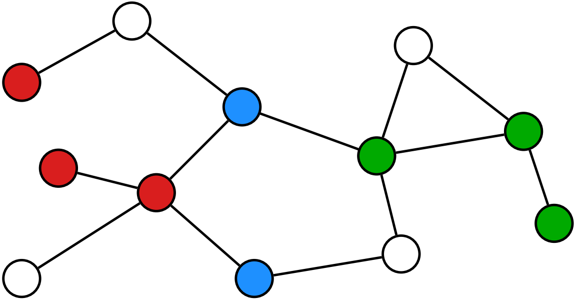


- Joint distribution $p(X) = p(X_1, \ldots X_n)$
- Undirected graph $\mathcal{G}$ on $V = \{1, \ldots, n\}$ with the Global Markov property: For $A, B, S \subseteq V$ it holds that

$$X_A \perp\!\!\!\perp X_B \mid X_S$$

if S separates A and B in $\mathcal{G}$.

Markov networks

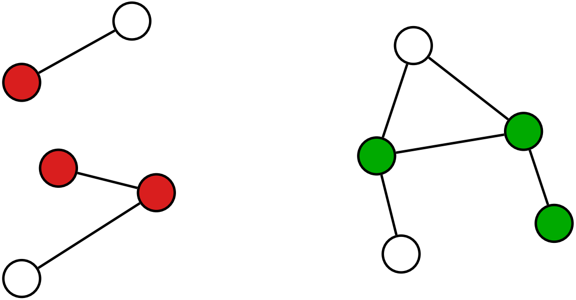


- Joint distribution $p(X) = p(X_1, \ldots X_n)$
- Undirected graph $\mathcal{G}$ on $V = \{1, \ldots, n\}$ with the Global Markov property: For $A, B, S \subseteq V$ it holds that

$$X_A \perp\!\!\!\perp X_B \mid X_S$$

if S separates A and B in $\mathcal{G}$.

Markov networks



- Joint distribution $p(X) = p(X_1, \ldots X_n)$
- Undirected graph $\mathcal{G}$ on $V = \{1, \ldots, n\}$ with the Global Markov property: For $A, B, S \subseteq V$ it holds that

$$X_A \perp\!\!\!\perp X_B \mid X_S$$

if S separates A and B in $\mathcal{G}$.

Markov networks

If p is strictly positive, it factorizes as

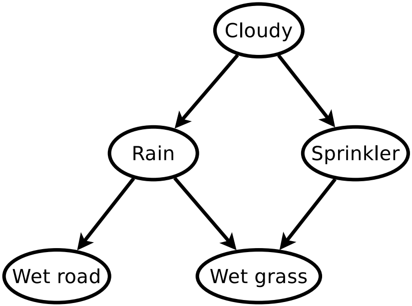
$$p(X_1, \ldots, X_n) = \prod_{C \in \mathcal{C}} \psi_C(X_C) \ ,$$

where

- $\mathcal{C}$ is the set of (maximal) cliques of $\mathcal{G}$
- ψ_C are mappings to positive reals
- $X_C = \{X_v : v \in C\}$

(Hammersley–Clifford Theorem)

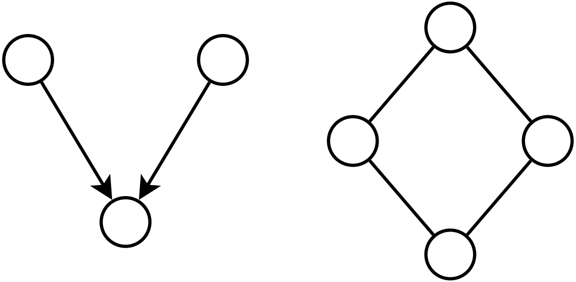
Bayesian networks



- Directed acyclic graph
- Conditional independencies by d-separation
- Factorizes:

$$p(X_1, \ldots, X_n) = \prod_{i=1}^{n} p(X_i \mid parents(X_i))$$

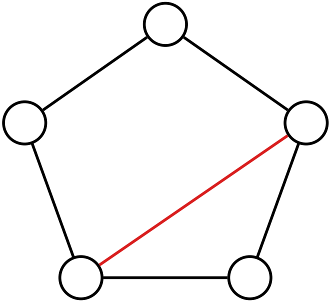
Bayesian and Markov networks



- Bayesian and Markov networks are not equivalent
- Chordal Markov networks are the intersection between the two

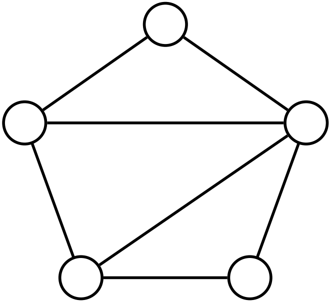
Chordal graphs

- A *chord* is an edge between two non-consecutive vertices in a cycle.
- An graph is *chordal* or *triangulated* if every cycle of at least 4 vertices has a *chord*.

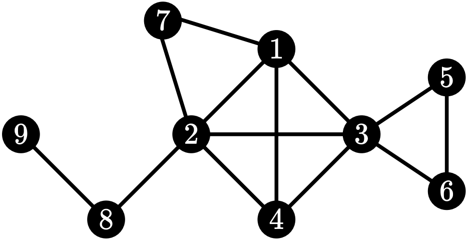


Chordal graphs

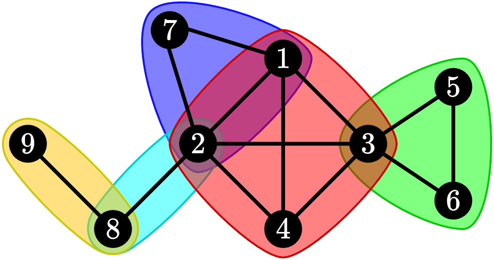
- A *chord* is an edge between two non-consecutive vertices in a cycle.
- An graph is *chordal* or *triangulated* if every cycle of at least 4 vertices has a *chord*.



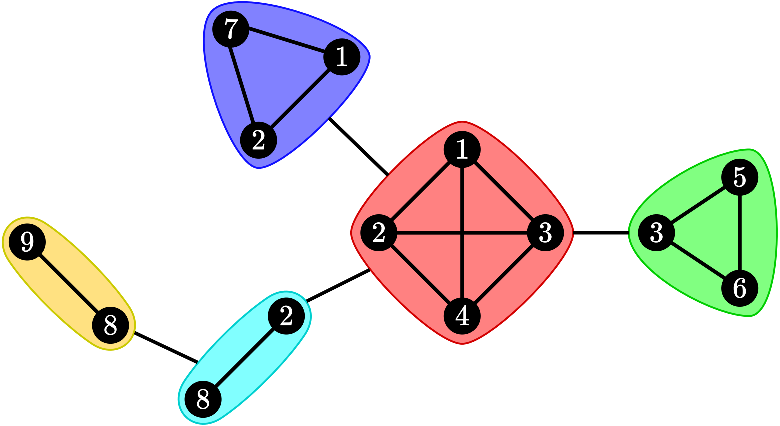
Clique tree decomposition



Clique tree decomposition

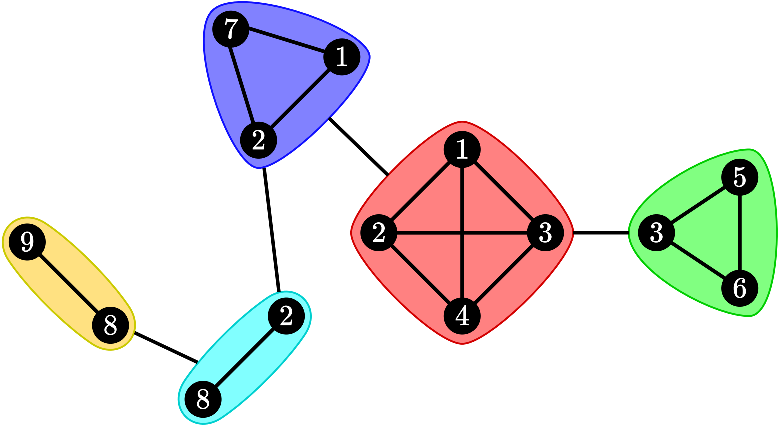


Clique tree decomposition



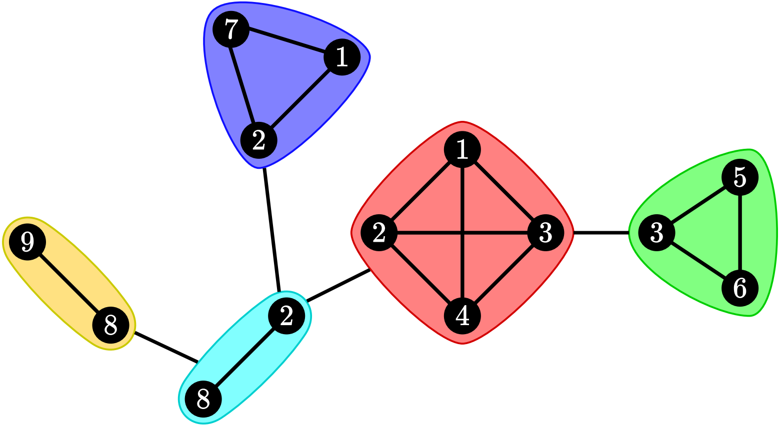
Running intersection property: For all $C_1, C_2 \in \mathcal{C}$, every clique on the path between C_1 and C_2 contains $C_1 \cap C_2$.

Clique tree decomposition



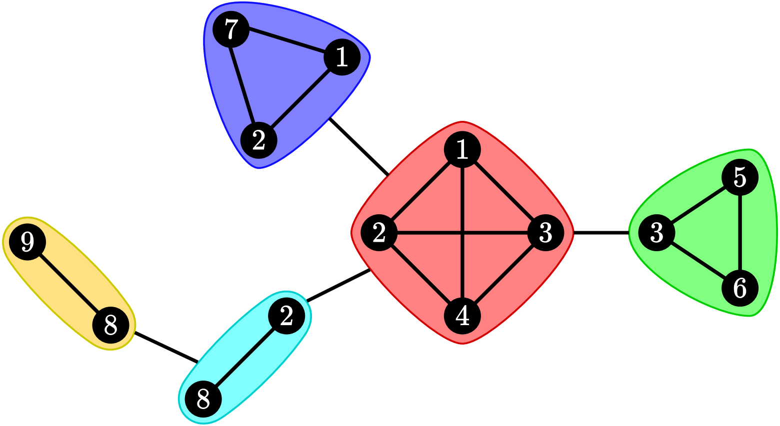
Running intersection property: For all $C_1, C_2 \in \mathcal{C}$, every clique on the path between C_1 and C_2 contains $C_1 \cap C_2$.

Clique tree decomposition



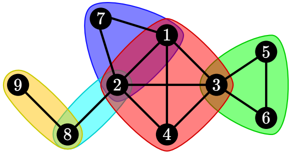
Running intersection property: For all $C_1, C_2 \in \mathcal{C}$, every clique on the path between C_1 and C_2 contains $C_1 \cap C_2$.

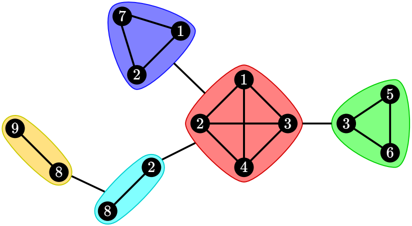
Clique tree decomposition



Separator: Intersection of adjacent cliques in a clique tree. Every clique tree has the same multiset of separators.

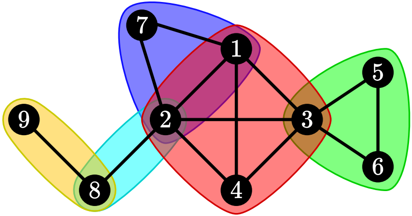
Clique tree decomposition





Theorem: A graph is chordal if and only if it has a clique tree.

Chordal Markov networks



- $\psi_i(X_{C_i}) = p(C_i)/p(S_i)$
- Factorization becomes

$$p(X_1, \ldots, X_n) = \prod_{C \in \mathcal{C}} \psi_C(X_C) = \frac{\prod_{C \in \mathcal{C}} p(X_C)}{\prod_{S \in \mathcal{S}} p(X_S)} \,,$$

where $\mathcal{C}$ and $\mathcal{S}$ are the sets of cliques and separators.

Structure learning

Given sampled data D from $p(X_1, \ldots X_n)$, how well does a graph structure $\mathcal{G}$ fit the data?

Common scoring criteria decompose as

$$score(\mathcal{G}) = \frac{\prod_{C \in \mathcal{C}} score(C)}{\prod_{S \in \mathcal{S}} score(S)}$$

Each $score(C)$ is the probability of the data projected to C, possibly extended with a prior or penalization term.

e.g. maximum likelihood, Bayesian Dirichlet, ...

Structure learning

Structure learning problem in chordal Markov networks:

Given $score(C)$ for each $C \subseteq V$, find a chordal graph $\mathcal{G}$ that maximizes

$$score(\mathcal{G}) = \frac{\prod_{C \in \mathcal{C}} score(C)}{\prod_{S \in \mathcal{S}} score(S)} \, .$$

We assume each $score(C)$ can be efficiently computed and focus on the combinatorial problem.

Structure learning

Bruteforce solution:

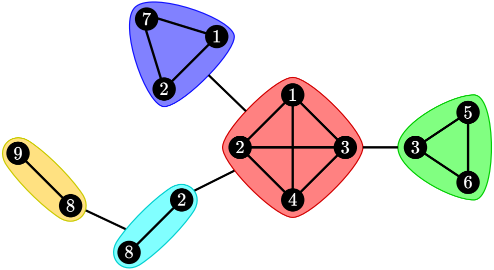
- Enumerate undirected graphs
- Determine which are chordal
- For each chordal $\mathcal{G}$, find a clique tree to evaluate $score(\mathcal{G})$
- $O^*(2^{\binom{n}{2}})$

Structure learning

We denote $score(\mathcal{T}) = score(\mathcal{G})$ when $\mathcal{T}$ is a clique tree of $\mathcal{G}$.

- Every clique tree $\mathcal{T}$ uniquely specifies a chordal graph $\mathcal{G}$.
- We can search the space of clique trees instead.

Recursive characterization



Let $\mathcal{T}$ be rooted at C with subtrees $\mathcal{T}_1, \ldots, \mathcal{T}_k$ rooted at $C_1, \ldots, C_k$. Then,

$$score(\mathcal{T}) = score(C) \prod_{i=1}^{k} \frac{score(\mathcal{T}_i)}{score(C \cap C_i)}$$

Recurrence

For $S \subset V$ and $\varnothing \subset R \subseteq V \setminus S$,

let $f(S, R)$ be the maximum $\mathit{score}(\mathcal{G})$ over chordal $\mathcal{G}$ on $S \cup R$ such that S is a proper subset of a clique.

Then, the solution is given by $f(\varnothing, V)$.

Recurrence

For $S \subset V$ and $\varnothing \subset R \subseteq V \setminus S$,

let $f(S, R)$ be the maximum $score(\mathcal{G})$ over chordal $\mathcal{G}$ on $S \cup R$ such that S is a proper subset of a clique.

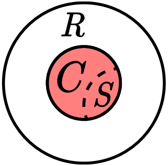
Then, the solution is given by $f(\varnothing, V)$.

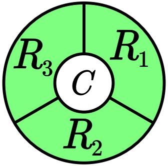
$$f(S, R) = \max_{\substack{S \subset C \subseteq S \cup R \\ \{R_1, \ldots, R_k\} \sqsubset R \setminus C \\ S_1, \ldots, S_k \subset C}} score(C) \prod_{i=1}^{k} \frac{f(S_i, R_i)}{score(S_i)} .$$

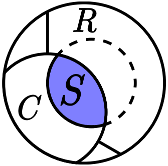
Recurrence

$$score(\mathcal{T}) = score(C) \prod_{i=1}^{k} \frac{score(\mathcal{T}_i)}{score(C \cap C_i)}$$

$$f(S, R) = \max_{\substack{S \subset C \subseteq S \cup R \\ \{R_1, \ldots, R_k\} \sqsubset R \setminus C \\ S_1, \ldots, S_k \subset C}} score(C) \prod_{i=1}^{k} \frac{f(S_i, R_i)}{score(S_i)}$$



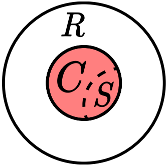


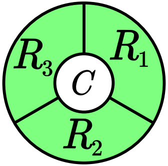


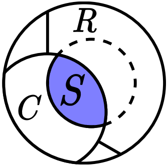
Recurrence

$$score(\mathcal{T}) = score(C) \prod_{i=1}^{k} \frac{score(\mathcal{T}_i)}{score(C \cap C_i)}$$

$$f(R) = \max_{\substack{\emptyset \subset C \subseteq R \\ \{R_1, \ldots, R_k\} \sqsubset R \setminus C \\ S_1, \ldots, S_k \subset C}} score(C) \prod_{i=1}^{k} \frac{f(S_i \cup R_i)}{score(S_i)}$$







Recurrence

$$f(S,R) = \max_{\substack{S \subset C \subseteq S \cup R \\ \{R_1,\ldots,R_k\} \sqsubset R \setminus C \\ S_1,\ldots,S_k \subset C}} score(C) \prod_{i=1}^{k} \frac{f(S_i,R_i)}{score(S_i)}$$

Recurrence

$$f(S,R) = \max_{\substack{S \subset C \subseteq S \cup R \\ \{R_1,\ldots,R_k\} \sqsubset R \setminus C}} score(C) \prod_{i=1}^{k} \max_{S_i \subset C} \frac{f(S_i,R_i)}{score(S_i)}$$

Recurrence

$$f(S,R) = \max_{\substack{S \subset C \subseteq S \cup R \\ \{R_1,\ldots,R_k\} \sqsubset R \setminus C}} score(C) \prod_{i=1}^{k} \max_{S_i \subset C} \frac{f(S_i,R_i)}{score(S_i)}$$

$$h(C,R) = \max_{S \subset C} \frac{f(S,R)}{score(S)}$$

Recurrence

$$f(S,R) = \max_{\substack{S \subset C \subseteq S \cup R \\ \{R_1,\ldots,R_k\} \sqsubset R \setminus C}} score(C) \prod_{i=1}^{k} h(C, R_i)$$

$$h(C,R) = \max_{S \subset C} \frac{f(S,R)}{score(S)}$$

Recurrence

$$f(S,R) = \max_{\substack{S \subset C \subseteq S \cup R \\ \{R_1,\ldots,R_k\} \sqsubset R \setminus C}} score(C) \prod_{i=1}^{k} h(C, R_i)$$

Recurrence

$$f(S,R) = \max_{S \subset C \subseteq S \cup R} score(C) \max_{\{R_1,\ldots,R_k\} \sqsubset R \setminus C} \prod_{i=1}^{k} h(C, R_i)$$

Recurrence

$$f(S,R) = \max_{S \subset C \subseteq S \cup R} score(C) \max_{\{R_1,\ldots,R_k\} \sqsubset R \setminus C} \prod_{i=1}^{k} h(C, R_i)$$

$$g(C,U) = \max_{\{R_1,\ldots,R_k\} \sqsubset U} \prod_{i=1}^{k} h(C, R_i)$$

Recurrence

$$f(S,R) = \max_{S \subset C \subseteq S \cup R} score(C) g(C, R \setminus C)$$

$$g(C,U) = \max_{\{R_1,\ldots,R_k\} \sqsubset U} \prod_{i=1}^{k} h(C, R_i)$$

Recurrence

$$g(C, U) = \max_{\{R_1,\ldots,R_k\} \sqsubset U} \prod_{i=1}^{k} h(C, R_i)$$

Recurrence

$$g(C, U) = \max_{\{R_1,\ldots,R_k\} \sqsubset U} \prod_{i=1}^{k} h(C, R_i)$$

If $U = \varnothing$, then $g(C, U) = 1$ (empty product).

Recurrence

$$g(C, U) = \max_{\{R_1,\ldots,R_k\} \sqsubset U} \prod_{i=1}^{k} h(C, R_i)$$

If $U = \varnothing$, then $g(C, U) = 1$ (empty product).
Otherwise

$$g(C, U) = \max_{\{R_1,\ldots,R_k\} \sqsubset U} \prod_{i=1}^{k} h(C, R_i)$$

Recurrence

$$g(C,U) = \max_{\{R_1,\ldots,R_k\} \sqsubset U} \prod_{i=1}^{k} h(C,R_i)$$

If $U = \varnothing$, then $g(C,U) = 1$ (empty product).
Otherwise

$$g(C,U) = \max_{\varnothing \neq R_1 \subseteq U} \max_{\{R_2,\ldots,R_k\} \sqsubset U \setminus R_1} \prod_{i=1}^{k} h(C,R_i)$$

Recurrence

$$g(C,U) = \max_{\{R_1,\ldots,R_k\} \sqsubset U} \prod_{i=1}^{k} h(C,R_i)$$

If $U = \varnothing$, then $g(C,U) = 1$ (empty product).
Otherwise

$$g(C,U) = \max_{\varnothing \neq R_1 \subseteq U} h(C,R_1) \max_{\{R_2,\ldots,R_k\} \sqsubset U \setminus R_1} \prod_{i=2}^{k} h(C,R_i)$$

Recurrence

$$g(C,U) = \max_{\{R_1,\ldots,R_k\} \sqsubset U} \prod_{i=1}^{k} h(C, R_i)$$

If $U = \varnothing$, then $g(C,U) = 1$ (empty product).
Otherwise

$$g(C,U) = \max_{\varnothing \neq R_1 \subseteq U} h(C, R_1) g(C, U \setminus R_1)$$

Recurrence

$$g(C, U) = \max_{\{R_1, \ldots, R_k\} \sqsubset U} \prod_{i=1}^{k} h(C, R_i)$$

If $U = \varnothing$, then $g(C, U) = 1$ (empty product).
Otherwise

$$g(C, U) = \max_{\varnothing \neq R \subseteq U} h(C, R) g(C, U \setminus R)$$

Recurrence

We have the split into three simpler recurrences:

$$f(S,R) = \max_{S \subset C \subseteq S \cup R} score(C) g(C, R \setminus C)$$

$$g(C,U) = \max_{\emptyset \subset R \subseteq U} h(C,R) g(C, U \setminus R)$$

$$h(C,R) = \max_{S \subset C} f(S,R) / score(S)$$

Dynamic programming in the increasing order of set size.

Space: $O(3^n)$
Time: $O(4^n)$

Efficient indexing

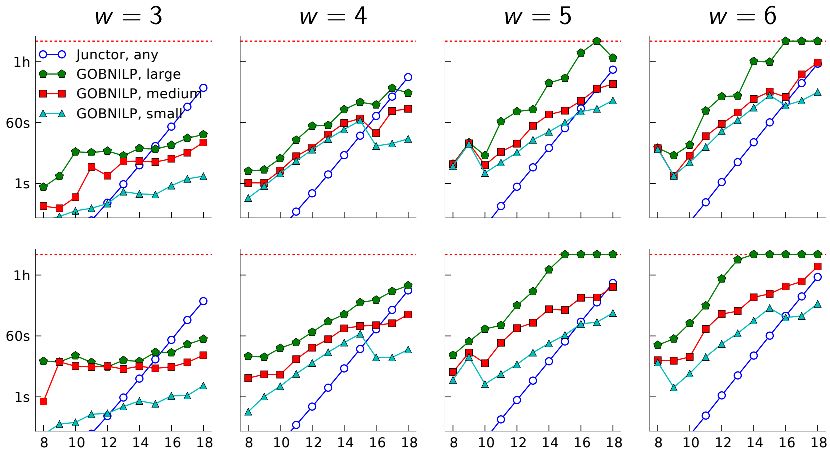
For each pair (A, B) compute the index

$$\sum_{v=1}^{n} 3^{v-1} \cdot I_v(A, B)$$

where

$$I_v(A, B) = \begin{cases} 1 \text{ if } v \in A, \\ 2 \text{ if } v \in B, \\ 0 \text{ otherwise.} \end{cases}$$

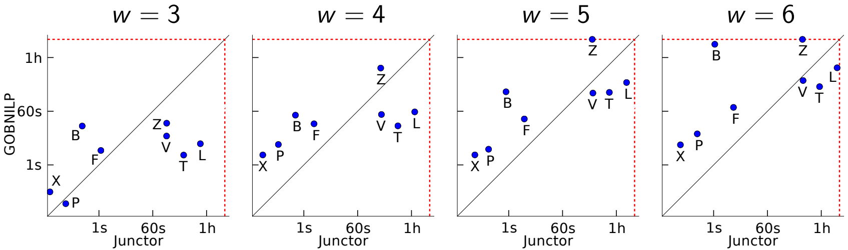
Experiments



Experiments

Dataset	Abbr.	n	m
Tic-tac-toe	X	10	958
Poker	P	11	10000
Bridges	B	12	108
Flare	F	13	1066
Zoo	Z	17	101

Dataset	Abbr.	n	m
Voting	V	17	435
Tumor	T	18	339
Lymph	L	19	148
Hypothyroid		22	3772
Mushroom		22	8124



Thank you!