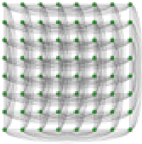
MAP Estimation with Perfect Graphs

Tony Jebara

July 21, 2009

1. Background
   - Perfect Graphs
   - Graphical Models
2. Matchings
   - Bipartite Matching
   - Generalized Matching
3. Perfect Graphs
   - nand Markov Random Fields
   - Packing Linear Programs
   - Recognizing Perfect Graphs
4. MAP Estimation
   - Proving Exact MAP
   - Pruning NMRFs
   - MAP Experiments
   - Conclusions

Background on Perfect Graphs

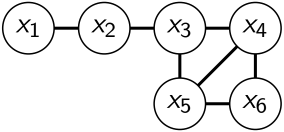
- In 1960, Claude Berge introduces perfect graphs and two conjectures
- Perfect iff every induced subgraph has clique # = coloring #
  - Weak conjecture: G is perfect iff its complement is perfect
  - Strong conjecture: all perfect graphs are Berge graphs
- Weak perfect graph theorem (Lovász 1972)
- Link between perfection and integral LPs (Lovász 1972)
- Strong perfect graph theorem (SPGT) open for 4+ decades

Background on Perfect Graphs




- SPGT Proof (Chudnovsky, Robertson, Seymour, Thomas 2003)
- Berge passes away shortly after hearing of the proof
- Many NP-hard and hard to approximate problems are P for perfect graphs
  - Graph coloring
  - Maximum clique
  - Maximum independent set
- Recognizing perfect graphs is $O(n^9)$ (Chudnovsky et al. 2006)

Graphical Models



- Perfect graph theory for MAP and graphical models (J 2009)
- Graphical model: a graph $G = (V, E)$ representing a distribution $p(X)$ where $X = \{x_1, \ldots, x_n\}$ and $x_i \in \mathbb{Z}$
- Distribution factorizes over graph cliques

$$p(x_1, \ldots, x_n) \quad = \quad \frac{1}{Z} \prod_{c \in C} \psi_c(X_c)$$

- E.g. $p(x_1, \ldots, x_6) = \psi(x_1, x_2)\psi(x_2, x_3)\psi(x_3, x_4, x_5)\psi(x_4, x_5, x_6)$

Canonical Problems for Graphical Models

- Given a factorized distribution

$$p(x_1, \ldots, x_n) \quad = \quad \frac{1}{Z} \prod_{c \in C} \psi_c(X_c)$$

- Inference: recover various marginals like $p(x_i)$ or $p(x_i, x_j)$

$$p(x_i) \quad = \quad \sum_{x_1} \cdots \sum_{x_{i-1}} \sum_{x_{i+1}} \cdots \sum_{x_n} p(x_1, \ldots, x_n)$$

- Estimation: find most likely configurations x_i^* or $(x_1^*, \ldots, x_n^*)$

$$x_i^* \quad = \quad \arg\max_{x_i} \max_{x_1} \cdots \max_{x_{i-1}} \max_{x_{i+1}} \cdots \max_{x_n} p(x_1, \ldots, x_n)$$

- In general both are NP-hard, but for chains and trees just P

Example for Chain

- Given a chain-factorized distribution

$$p(x_1, \ldots, x_5) = \frac{1}{Z}\psi(x_1, x_2)\psi(x_2, x_3)\psi(x_3, x_4)\psi(x_4, x_5)$$

- Inference: recover various marginals like $p(x_i)$ or $p(x_i, x_j)$

$$p(x_5) \propto \sum_{x_1}\sum_{x_2}\psi(x_1, x_2)\sum_{x_3}\psi(x_2, x_3)\sum_{x_4}\psi(x_3, x_4)\psi(x_4, x_5)$$

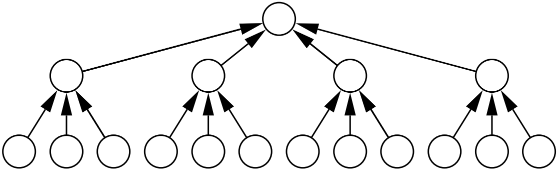
- Estimation: find most likely configurations x_i^* or $(x_1^*, \ldots, x_n^*)$

$$x_5^* = \arg\max_{x_5}\max_{x_1}\max_{x_2}\psi(x_1, x_2)\max_{x_3}\psi(x_2, x_3)\max_{x_4}\psi(x_3, x_4)\psi(x_4, x_5)$$

- The work distributes and becomes efficient!

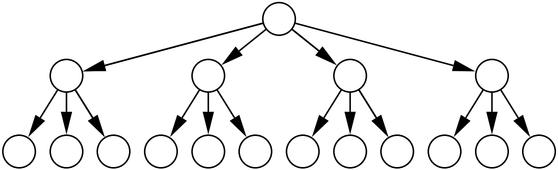
Canonical Problems on Trees

- The idea of distributed computation extends nicely to trees
- On trees (which subsume chains) do a collect/distribute step
- Alternatively, can perform distributed updates asynchronously
- Each step is a sum-product or a max-product update



Canonical Problems on Trees

- The idea of distributed computation extends nicely to trees
- On trees (which subsume chains) do a collect/distribute step
- Alternatively, can perform distributed updates asynchronously
- Each step is a sum-product or a max-product update



MAP Estimation

- Let's focus on finding most probable configurations efficiently

$$X^* = \arg\max p(x_1, \ldots, x_n)$$

- Useful for image processing, protein folding, coding, etc.
- Brute force requires $\prod_{i=1}^n |x_i|$
- Efficient for trees and singly linked graphs (Pearl 1988)
- NP-hard for general graphs (Shimony 1994)
- Approach A: relaxations and variational methods
  - First order LP relaxations (Wainwright et al. 2002)
  - TRW max-product (Kolmogorov & Wainwright 2006)
  - Higher order LP relaxations (Sontag et al. 2008)
  - Fractional and integral LP rounding (Ravikumar et al. 2008)
  - Open problem: when are LPs tight?
- Approach B: loopy max product and message passing

Max Product Message Passing

1. For each x_i to each X_c: $m_{i \to c}^{t+1} = \prod_{d \in \mathrm{Ne}(i) \setminus c} m_{d \to i}^t$
2. For each X_c to each x_i: $m_{c \to i}^{t+1} = \max_{X_c \setminus x_i} \psi_c(X_c) \prod_{j \in c \setminus i} m_{j \to c}^t$
3. Set $t = t + 1$ and goto 1 until convergence
4. Output $x_i^* = \arg\max_{x_i} \prod_{d \in \mathrm{Ne}(i)} m_{d \to i}^t$

- Simple and fast algorithm for MAP
- Exact for trees (Pearl 1988)
- Exact for single-loop graphs (Weiss & Freeman 2001)
- Local optimality guarantees (Wainwright et al. 2003)
- Performs well in practice for images, turbo codes, etc.
- Similar to first order LP relaxation
- Recent progress
  - Exact for matchings (Bayati et al. 2005)
  - Exact for generalized b matchings (Huang and J 2007)

Bipartite Matching

	Motorola	Apple	IBM
"laptop"	0\$	2\$	2\$
"server"	0\$	2\$	3\$
"phone"	2\$	3\$	0\$

$$\rightarrow C = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}$$

- Given W, $\max_{C \in \mathbb{B}^{n \times n}} \sum_{ij} W_{ij} C_{ij}$ such that $\sum_i C_{ij} = \sum_j C_{ij} = 1$
- Classical Hungarian marriage problem $O(n^3)$
- Creates a very loopy graphical model
- Max product takes $O(n^3)$ for exact MAP (Bayati et al. 2005)

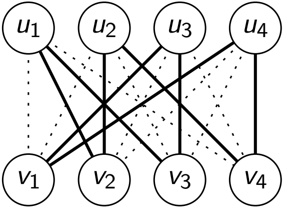
Bipartite Generalized Matching

	Motorola	Apple	IBM
"laptop"	0$	2$	2$
"server"	0$	2$	3$
"phone"	2$	3$	0$

$$\rightarrow C = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}$$

- Given W, $\max_{C \in \mathbb{B}^{n \times n}} \sum_{ij} W_{ij} C_{ij}$ such that $\sum_i C_{ij} = \sum_j C_{ij} = b$
- Combinatorial b-matching problem $O(bn^3)$, (Google Adwords)
- Creates a very loopy graphical model
- Max product takes $O(bn^3)$ for exact MAP (Huang & J 2007)

Bipartite Generalized Matching



- Graph $G = (U, V, E)$ with $U = \{u_1, \ldots, u_n\}$ and $V = \{v_1, \ldots, v_n\}$ and $M(.)$, a set of neighbors of node u_i or v_j
- Define $x_i \in X$ and $y_i \in Y$ where $x_i = M(u_i)$ and $y_i = M(v_j)$
- Then $p(X, Y) = \frac{1}{Z} \prod_i \prod_j \psi(x_i, y_j) \prod_k \phi(x_k)\phi(y_k)$ where $\phi(y_j) = \exp(\sum_{u_i \in y_j} W_{ij})$ and $\psi(x_i, y_j) = \neg(v_j \in x_i \oplus u_i \in y_j)$

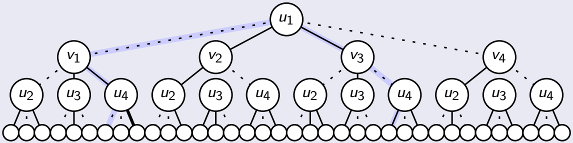
Bipartite Generalized Matching

Theorem (Huang & J 2007)

Max product on G converges in $O(bn^3)$ time.

Proof.

Form unwrapped tree T of depth $\Omega(n)$, maximizing belief at root of T is equivalent to maximizing belief at corresponding node in G



□

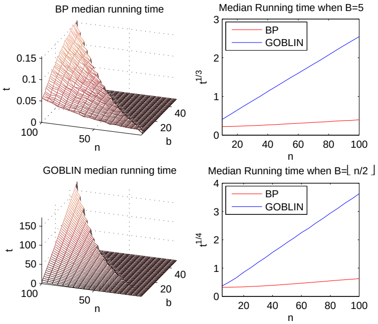
Theorem (Salez & Shah 2009)

Under mild assumptions, max product 1-matching is $O(n^2)$.

Bipartite Generalized Matching

- Code at http://www.cs.columbia.edu/~jebara/code

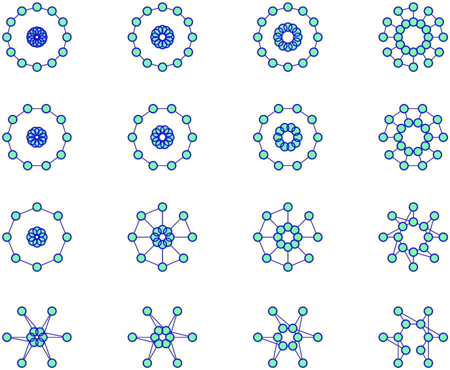
Generalized Matching



Applications:
unipartite matching
clustering (J & S 2006)
classification (H & J 2007)
collaborative filtering (H & J 2008)
semisupervised (J et al. 2009)
visualization (S & J 2009)

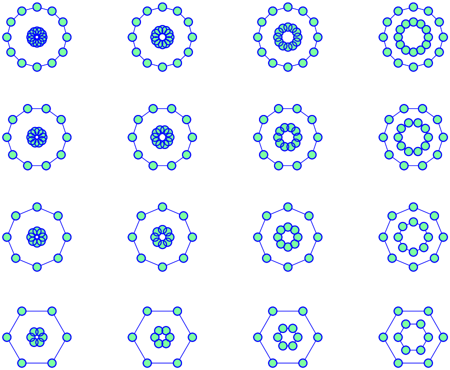
Max product is $O(n^2)$ on dense graphs (Salez & Shah 2009)
Much faster than other solvers

Unipartite Generalized Matching



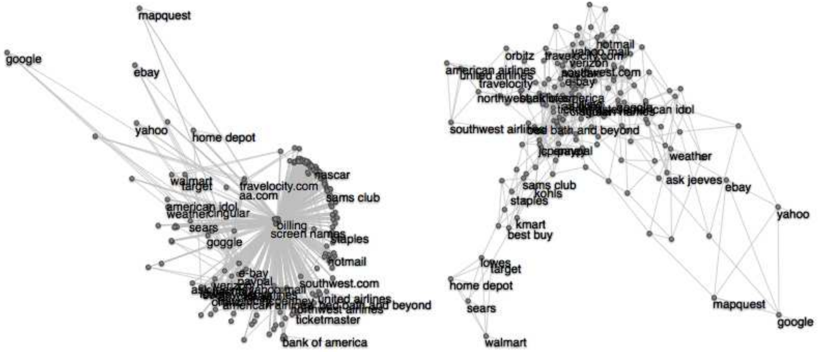
- Above is k-nearest neighbors with $k = 2$

Unipartite Generalized Matching



- Above is unipartite b-matching with $b = 2$

Unipartite Generalized Matching



- Left is k-nearest neighbors, right is unipartite b-matching.

Unipartite Generalized Matching

	p_1	p_2	p_3	p_4
p_1	0	2	1	2
P_2	2	0	2	1
p_3	1	2	0	2
p_4	2	1	2	0

$$\rightarrow C = \begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix}$$

- $\max_{C \in \mathbb{B}^{n \times n}, C_{ii}=0} \sum_{ij} W_{ij} C_{ij}$ such that $\sum_i C_{ij} = b$, $C_{ij} = C_{ji}$
- Combinatorial unipartite matching is efficient (Edmonds 1965)
- Makes an LP with exponentially many blossom inequalities
- Max product exact if LP is integral (Sanghavi et al. 2008)

$$p(X) = \prod_{i \in V} \delta\left(\sum_{j \in \mathrm{Ne}(i)} x_{ij} \le 1\right) \prod_{ij \in E} \exp(W_{ij} x_{ij})$$

Back to Perfect Graphs

- Max product and exact MAP depend on the LP's integrality
- Matchings have special integral LPs (Edmonds 1965)
- How to generalize beyond matchings?
- Perfect graphs imply LP integrality (Lovász 1972)

Lemma (Lovász 1972)

For every non-negative vector $\vec{f} \in \mathbb{R}^N$, the linear program

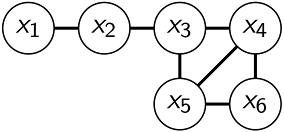
$$\beta = \max_{\vec{x} \in \mathbb{R}^N} \vec{f}^\top \vec{x} \text{ subject to } \vec{x} \geq 0 \text{ and } A\vec{x} \leq \vec{1}$$

recovers a vector $\vec{x}$ which is integral if and only if the (undominated) rows of A form the vertex versus maximal cliques incidence matrix of some perfect graph.

Back to Perfect Graphs

Lemma (Lovász 1972)

$$\beta \;=\; \max_{\vec{x}\in\mathbb{R}^N} \vec{f}^{\top}\vec{x} \text{ subject to } \vec{x} \geq 0 \text{ and } A\vec{x} \leq \vec{1}$$



$$A \;=\; \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix}$$

nand Markov Random Fields

- Lovász's lemma is not solving $\max p(X)$ on G
- We have $p(x_1, \ldots, x_n) = \frac{1}{Z} \prod_{c \in C} \psi_c(X_c)$
- How to apply the lemma to any model G and space X?
- Without loss of generality assume $\psi_c(X_c) \leftarrow \frac{\psi_c(X_c)}{\min_{X_c} \psi_c(X_c)} + \epsilon$
- Consider procedure to convert G to $\mathcal{G}$ in NMRF form
- NMRF is a nand Markov random field over space $\mathbf{X}$
    - all variables are binary $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$
    - all potential functions are pairwise nand gates $\psi_{ij}(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i, \mathbf{x}_j) = \delta(\mathbf{x}_i + \mathbf{x}_j \leq 1)$

nand Markov Random Fields

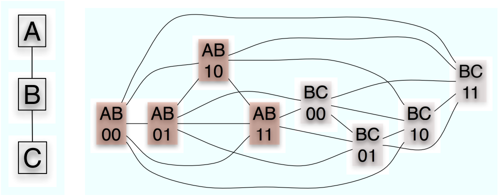


Figure: Binary graphical model G (left) and nand MRF $\mathcal{G}$ (right).

Initialize $\mathcal{G}$ as the empty graph
For each clique c in graph G do
    For each configuration $k \in X_c$ do
        add a corresponding binary node $\mathbf{x}_{c,k}$ to $\mathcal{G}$
        for each $\mathbf{x}_{d,l} \in \mathcal{G}$ which is incompatible with $\mathbf{x}_{c,k}$
            connect $\mathbf{x}_{c,k}$ and $\mathbf{x}_{d,l}$ with an edge

nand Markov Random Fields

- Obtain the following distribution from $\mathcal{G}$
  $\rho(\mathbf{X}) = \prod_{c \in C} \prod_k^{|X_c|} e^{f_{c,k} \mathbf{x}_{c,k}} \prod_{d \in C} \prod_{l=1}^{|X_d|} \Phi(\mathbf{x}_{c,k}, \mathbf{x}_{d,l})^{\mathbf{E}(\mathbf{x}_{c,k}, \mathbf{x}_{d,l})}$
  where $f_{c,k} = \log \psi_c(k)$.
- Cardinality of $\mathcal{G}$ is $|\mathbf{X}| = \sum_{c \in C} \left(\prod_{i \in c} |x_i|\right) = N$
- If node $\mathbf{x}_{c,k} = 1$ then clique c is in configuration $k \in X_c$.
- Clearly surjective, more configurations $\mathbf{X}$ than X
- Nand relationship prevents inconsistent settings $\sum_k \mathbf{x}_{c,k} \leq 1$

Theorem

For $f_{c,k} > 0$, MAP estimate $\mathbf{X}^*$ of $\rho(\mathbf{X})$ yields $\sum_k \mathbf{x}^*_{c,k} = 1$ for all cliques $c \in C$.

Packing Linear Programs

Lemma

The MAP estimate for $\rho(\mathbf{X})$ on $\mathcal{G}$ recovers MAP for $p(X)$

- Relaxation of MAP on $\rho(\mathbf{X}) \equiv$ set packing linear program
- If graph $\mathcal{G}$ is perfect, LP efficiently solves MAP

Lemma

For every non-negative vector $\vec{f} \in \mathbb{R}^N$, the linear program

$$\beta = \max_{\vec{x} \in \mathbb{R}^N} \vec{f}^\top \vec{x} \text{ subject to } \vec{x} \geq 0 \text{ and } A\vec{x} \leq \vec{1}$$

recovers a vector $\vec{x}$ which is integral if and only if the (undominated) rows of A form the vertex versus maximal cliques incidence matrix of some perfect graph.

Packing Linear Programs

- For general graph G, MAP is NP-hard (Shimony 1994)
- Convert G to $\mathcal{G}$ (polynomial time)
- If graph $\mathcal{G}$ is perfect
  - Find maximal cliques (polynomial time)
  - Solve MAP via packing linear program (polynomial time)

Theorem

MAP estimation of any graphical model G with cliques $c \in C$ over variables $\{x_1, \ldots, x_n\}$ producing a nand Markov random with a perfect graph $\mathcal{G}$ is in P and requires no more than $O\left(\left(\sum_{c \in C}\left(\prod_{i \in c}|x_i|\right)\right)^3\right)$.

Packing Linear Programs

- For general graph G, MAP is NP-hard (Shimony 1994)
- Convert G to $\mathcal{G}$ (polynomial time)
- If graph $\mathcal{G}$ is perfect (? time)
  - Find maximal cliques (polynomial time)
  - Solve MAP via packing linear program (polynomial time)

Theorem

MAP estimation of any graphical model G with cliques $c \in C$ over variables $\{x_1, \ldots, x_n\}$ producing a nand Markov random with a perfect graph $\mathcal{G}$ is in P and requires no more than $O\left(\left(\sum_{c \in C}\left(\prod_{i \in c}|x_i|\right)\right)^3\right)$.

Packing Linear Programs

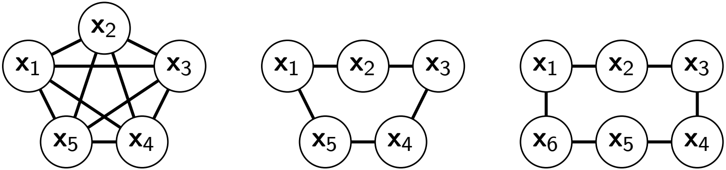
- For general graph G, MAP is NP-hard (Shimony 1994)
- Convert G to $\mathcal{G}$ (polynomial time)
- If graph $\mathcal{G}$ is perfect (polynomial time!!!)
  - Find maximal cliques (polynomial time)
  - Solve MAP via packing linear program (polynomial time)

Theorem

MAP estimation of any graphical model G with cliques $c \in C$ over variables $\{x_1, \ldots, x_n\}$ producing a nand Markov random with a perfect graph $\mathcal{G}$ is in P and requires no more than $O\left(\left(\sum_{c \in C} \left(\prod_{i \in c} |x_i|\right)\right)^3\right)$.

Perfect Graphs

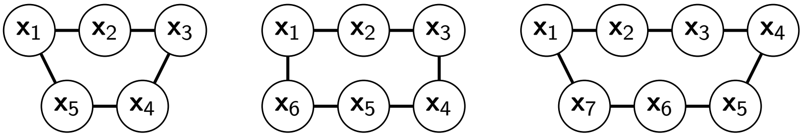
- To determine if $\mathcal{G}$ is perfect
  - Run algorithm on $\mathcal{G}$ in $O(N^9)$ (Chudnovsky et al. 2005)
  - or use tools from perfect graph theory to prove perfection
- Clique number of a graph $\omega(\mathcal{G})$: size of its maximum clique
- Chromatic number of a graph $\chi(\mathcal{G})$: minimum number of colors such that no two adjacent vertices have the same color
- A perfect graph $\mathcal{G}$ is a graph where every induced subgraph $\mathcal{H} \subseteq \mathcal{G}$ has $\omega(\mathcal{H}) = \chi(\mathcal{H})$



Perfect Not Perfect Perfect

Strong Perfect Graph Theorem

- A graph is perfect iff it is Berge (Chudnovsky et al. 2003)
- Berge graph: a graph that contains no odd hole and whose complement also contains no odd hole
- Hole: an induced subgraph of $\mathcal{G}$ which is a chordless cycle of length at least 5. An odd hole has odd cycle length.
- Complement: a graph $\bar{\mathcal{G}}$ with the same vertex set $\mathsf{V}(\mathcal{G})$ as $\mathcal{G}$, where distinct vertices $\mathbf{u}, \mathbf{v} \in \mathsf{V}(\mathcal{G})$ are adjacent in $\bar{\mathcal{G}}$ just when they are not adjacent in $\mathcal{G}$



odd hole even hole odd hole

Recognition using Perfect Graphs Algorithm

- Could use slow $O(N^9)$ algorithm (Chudnovsky et al. 2005)
- Runs on $\mathcal{G}$ and then on complement $\bar{\mathcal{G}}$
  - Detect if the graph contains a pyramid structure by computing shortest paths between all nonuples of vertices. This is $O(N^9)$
  - Detect if the graph contains a jewel structure or other easily-detectable configuration
  - Perform a cleaning procedure. A vertex in the graph is C-major if its set of neighbors in C is not a subset of the vertex set of any 3-vertex path of C. C is clean if there are no C-major vertices in the graph
  - Search for the shortest odd hole in the graph by computing the shortest paths between all triples of vertices
- Faster methods find all holes (Nikolopolous & Palios 2004)
- Less conclusive than Chudnovsky but can run on $N \geq 300$

Recognition using Strong Perfect Graph Theorem

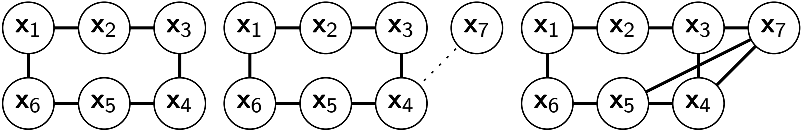
- SPGT implies that a Berge graph is one of these primitives
  - bipartite graphs
  - complements of bipartite graphs
  - line graphs of bipartite graphs
  - complements of line graphs of bipartite graphs
  - double split graphs
- or decomposes structurally (into graph primitives)
  - via a 2-join
  - via a 2-join in the complement
  - via an M-join
  - via a balanced skew partition
- Line graph: $\mathrm{L}(\mathcal{G})$ a graph which contains a vertex for each edge of $\mathcal{G}$ and where two vertices of $\mathrm{L}(\mathcal{G})$ are adjacent iff they correspond to two edges of $\mathcal{G}$ with a common end vertex

Recognition using Strong Perfect Graph Theorem

- SPGT and theory give tools to analyze graph
- Decompose using replication, 2-join, M-joins, skew partition...
- May help diagnose perfection when algorithm is too slow

Lemma (Replication, Lovász 1972)

Let $\mathcal{G}$ be a perfect graph and let $v \in V(\mathcal{G})$. Define a graph $\mathcal{G}'$ by adding a new vertex v' and joining it to v and all the neighbors of v. Then $\mathcal{G}'$ is perfect.

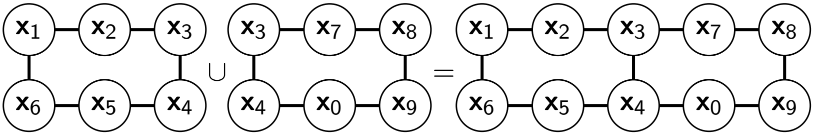


Recognition using Strong Perfect Graph Theorem

- SPGT and theory give tools to analyze graph
- Decompose using replication, 2-join, M-joins, skew partition...
- May help diagnose perfection when algorithm is too slow

Lemma (Gluing on Cliques, Skew Partition, Berge & Chvátal 1984)

Let $\mathcal{G}$ be a perfect graph and let $\mathcal{G}'$ be a perfect graph. If $\mathcal{G} \cap \mathcal{G}'$ is a clique (clique cutset), then $\mathcal{G} \cup \mathcal{G}'$ is a perfect graph.



Proving Exact MAP for Tree Graphs

Theorem (J 2009)

Let G be a tree, the NMRF $\mathcal{G}$ obtained from G is a perfect graph.

Proof.

First prove perfection for a star graph with internal node v with $|v|$ configurations. First obtain $\mathcal{G}$ for the star graph by only creating one configuration for non internal nodes. The resulting graph is a complete $|v|$-partite graph which is perfect. Introduce additional configurations for non-internal nodes one at a time using the replication lemma. The resulting $\mathcal{G}_{star}$ is perfect. Obtain a tree by induction. Add two stars $\mathcal{G}_{star}$ and $\mathcal{G}_{star'}$. The intersection is a fully connected clique (clique cutset) so by (Berge & Chvátal 1984), the resulting graph is perfect. Continue gluing stars until full tree G is formed. $\square$

Proving Exact MAP for Bipartite Matchings

Theorem (J 2009)

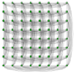
The maximum weight bipartite matching graphical model

$$p(X) = \prod_{i=1}^{n} \delta\left(\sum_{j=1}^{n} x_{ij} \leq 1\right) \delta\left(\sum_{j=1}^{n} x_{ji} \leq 1\right) \prod_{k=1}^{n} e^{f_{ik} x_{ik}}$$

with $f_{ij} \geq 0$ *has integral LP and yields exact MAP estimates.*

Proof.

The graphical model is in NMRF form so G and $\mathcal{G}$ are equivalent. $\mathcal{G}$ is the line graph of a (complete) bipartite graph (Rook's graph). Therefore, $\mathcal{G}$ is perfect, the LP is integral and recovers MAP. $\square$



Proving Exact MAP for Unipartite Matchings

Theorem (J 2009)

The unipartite matching graphical model $G = (V, E)$ *with* $f_{ij} \geq 0$

$$p(X) = \prod_{i \in V} \delta\left(\sum_{j \in Ne(i)}^{n} x_{ij} \leq 1\right) \prod_{ij \in E} e^{f_{ij} x_{ij}}$$

has integral LP and produces the exact MAP estimate if G *is a perfect graph.*

Proof.

The graphical model is in NMRF form and graphs G and $\mathcal{G}$ are equivalent. The set packing LP relaxation is integral and recovers the MAP estimate if $\mathcal{G}$ is a perfect graph. □

Pruning NMRFs

- Possible to prune $\mathcal{G}$ in search of perfection and efficiency
- Two optional procedures: **Disconnect** and **Merge**
- **Disconnect**: For each $c \in C$, denote the minimal configurations of c as the set of nodes $\{\mathbf{x}_{c,k}\}$ such that $f_{c,k} = \min_\kappa f_{c,\kappa} = \log(1+\epsilon)$. **Disconnect** removes the edges between these nodes and all other nodes in the clique $\mathbf{X}_c$.
- **Merge**: For any pair of unconnected nodes $\mathbf{x}_{c,k}$ and $\mathbf{x}_{d,l}$ in $\mathcal{G}$ where $\mathrm{Ne}(\mathbf{x}_{c,k}) = \mathrm{Ne}(\mathbf{x}_{d,l})$, combine them into a single equivalent variable $\mathbf{x}_{c,k}$ with the same connectivity and updates its corresponding weight as $f_{c,k} \leftarrow f_{c,k} + f_{d,l}$.
- Easy to get MAP for $\mathcal{G}$ from **Merge**(**Disconnect**($\mathcal{G}$))

Convergent Message Passing

- Instead of LP solver, use convergent message passing (Globerson & Jaakkola 2007) get faster solution

Input: Graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and θ_{ij} for $ij \in \mathcal{E}$.

1. Initialize all messages to any value.
2. For each $ij \in \mathcal{E}$, simultaneously update

$$\lambda_{ji}(\mathbf{x}_i) \leftarrow -\frac{1}{2} \sum_{k \in \mathrm{Ne}(i) \setminus j} \lambda_{ki}(\mathbf{x}_i) + \frac{1}{2} \max_{\mathbf{x}_j} \left[\sum_{k \in \mathrm{Ne}(j) \setminus i} \lambda_{kj}(\mathbf{x}_j) + \theta_{ij}(\mathbf{x}_i, \mathbf{x}_j) \right]$$

$$\lambda_{ij}(\mathbf{x}_j) \leftarrow -\frac{1}{2} \sum_{k \in \mathrm{Ne}(j) \setminus i} \lambda_{kj}(\mathbf{x}_j) + \frac{1}{2} \max_{\mathbf{x}_i} \left[\sum_{k \in \mathrm{Ne}(i) \setminus j} \lambda_{ki}(\mathbf{x}_i) + \theta_{ij}(\mathbf{x}_i, \mathbf{x}_j) \right]$$

3. Repeat 2 until convergence.
4. Find $b(\mathbf{x}_i) = \sum_{j \in \mathrm{Ne}(i)} \lambda_{ji}(\mathbf{x}_i)$ for all $i \in \mathcal{V}$.
5. Output $\hat{\mathbf{x}}_i = \arg\max_{\mathbf{x}_i} b(\mathbf{x}_i)$ for all $i \in \mathcal{V}$.

Convergent Message Passing

Theorem (Globerson & Jaakkola 2007)

With binary variables $\mathbf{x}_i$, fixed points of convergent message passing recover the optimum of the LP.

Corollary

Convergent message passing on an NMRF with a perfect graph finds the MAP estimate.

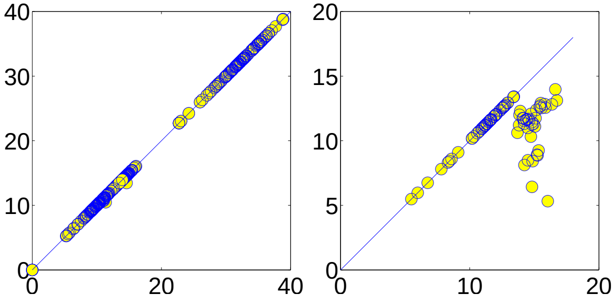
MAP Experiments

- Investigate LP and message passing for unipartite matching
- Exact MAP estimate possible via Edmonds' blossom algorithm
- Consider graphical model $G = (V, E)$ with $f_{ij} \geq 0$

$$p(X) = \prod_{i \in V} \delta\left(\sum_{j \in \mathrm{Ne}(i)}^{n} x_{ij} \leq 1 \right) \prod_{ij \in E} e^{f_{ij} x_{ij}}$$

- Compare solution found by message passing on the NMRF
- Try various topologies for graph G, perfect or otherwise

MAP Experiments



(a) Perfect Graphs (b) Random Graphs

Figure: Scores for the exact MAP estimate (horizontal axis) and message passing estimate (vertical axis) for random graphs and weights. Figure (a) shows scores for four types of basic Berge graphs while (b) shows scores for arbitrary graphs. Minor score discrepancies on Berge graphs arose due to numerical issues and early stopping.

Conclusions

- Perfect graph theory is fascinating
- It is a crucial tool for exploring LP integrality
- Many recent theoretical and algorithmic breakthroughs
- Integrality of LP is also crucial for exact MAP estimation
- MAP for any graphical model is exact if $\mathcal{G}$ is perfect
- Efficient tests for perfection, maximum clique and LP
- Can use max product or message passing instead of LP
- Perfect graphs extend previous results on MAP for
    - Trees and singly-linked graphs
    - Single loop graphs
    - Matchings
    - Generalized matchings

Further Reading and Thanks

- MAP Estimation, Message Passing, and Perfect Graphs, T. Jebara. *Uncertainty in Artificial Intelligence*, June 2009.
- Graphical Models, Exponential Families and Variational Inference, M.J. Wainwright and M.I. Jordan. *Foundations and Trends in Machine Learning*, Vol 1, Nos 1-2, 2008.
- Loopy Belief Propagation for Bipartite Maximum Weight b-Matching, B. Huang and T. Jebara. *Artificial Intelligence and Statistics*, March 2007.
- Thanks to Maria Chudnovsky, Delbert Dueck and Bert Huang.