Bayesian network learning by compiling to weighted MAX-SAT

James Cussens, University of York

Mathematical Aspects of Graphical Models, Durham, 2008-07-07

Outline

Model selection as combinatorial optimisation

- Model selection for Bayesian networks (using a decomposable score) is *combinatorial optimisation*.
- In this work the score is marginal likelihood with a Dirichlet parameter prior.
- $P(D|G) = \prod_{i=1}^{n} \prod_{j=1}^{q_i} \frac{\Gamma(\alpha_{ij})}{\Gamma(n_{ij}+\alpha_{ij})} \prod_{k=1}^{r_i} \frac{\Gamma(n_{ijk}+\alpha_{ijk})}{\Gamma(\alpha_{ijk})}$
- $\mathrm{Score(G)} \stackrel{\mathrm{def}}{=} \log P(D|G) = \sum_{i=1}^{n} \mathrm{Score}_i(\mathrm{Pa}_i(G))$.
- For each variable choose high-scoring parents subject to the constraint that no cycle is formed.

The basic idea

- Given that BN model selection is combinatorial optimisation . . .

The basic idea

- Given that BN model selection is combinatorial optimisation ...
- ... we can use state-of-the-art algorithms for combinatorial optimisation ...

The basic idea

- Given that BN model selection is combinatorial optimisation . . .
- . . . we can use state-of-the-art algorithms for combinatorial optimisation . . .
- . . . if we are prepared to do a little encoding.

Outline

The SAT problem

- Is a given set of propositional clauses (a CNF formula) satisfiable?

$$\overline{x_{12}} \vee \overline{x_{23}} \vee x_{13}$$

$$x_{12} \vee x_{23} \vee \overline{x_{13}} \qquad \text{OK} :(x_{12}, x_{23}, x_{13}), (x_{12}, \overline{x_{23}}, x_{13}), \ldots$$

The SAT problem

- Is a given set of propositional clauses (a CNF formula) satisfiable?

$$\overline{x_{12}} \vee \overline{x_{23}} \vee x_{13}$$

$$x_{12} \vee x_{23} \vee \overline{x_{13}} \qquad \text{OK} :(x_{12}, x_{23}, x_{13}), (x_{12}, \overline{x_{23}}, x_{13}), \ldots$$

$$x_{12}$$

$$x_{23} \qquad \text{OK} :(x_{12}, x_{23}, x_{13})$$

The SAT problem

- Is a given set of propositional clauses (a CNF formula) satisfiable?

$$
\begin{array}{rl}
\overline{x_{12}} \vee \overline{x_{23}} \vee x_{13} & \\
x_{12} \vee x_{23} \vee \overline{x_{13}} & \text{OK} : (x_{12}, x_{23}, x_{13}), (x_{12}, \overline{x_{23}}, x_{13}), \ldots \\
x_{12} & \\
x_{23} & \text{OK} : (x_{12}, x_{23}, x_{13}) \\
\overline{x_{13}} & \text{Unsatisfiable}
\end{array}
$$

- x_{12}, x_{23} and x_{13} are called *atoms*. (Short for atomic formulae.)

The weighted MAX-SAT problem

- Add weights to each clause (to get weighted CNF).
- Each assignment has a cost: the sum of the weights of the unsatisfied clauses.
- An infinite cost gives a 'hard' clause. (In practice a big number is used.)
- Goal: find an assignment with minimal cost.

$$
\begin{array}{rl}
9999 & \overline{x_{12}} \vee \overline{x_{23}} \vee x_{13} \\
9999 & x_{12} \vee x_{23} \vee \overline{x_{13}} \\
12 & x_{12} \\
34 & x_{23} \\
1 & \overline{x_{13}}
\end{array}
$$

Weighted MAX-SAT as mode finding for log-linear distributions

- Given weighted CNF $\lambda_1 C_1, \lambda_2 C_2, \ldots$
- Define $f_i(\mathbf{x}) = 1$ if $\mathbf{x}$ breaks clause C_i; else $= 0$
- $P(\mathbf{x}) = Z^{-1} \exp\left(\sum_i -\lambda_i f_i(\mathbf{x})\right)$

This connection has been exploited by those working on *Markov logic* where weighted *first-order* clauses are used.

Weighted MAX-SAT solvers

- Here are the SAT solving algorithms available in UBCSAT.
- 19 have weighted MAX-SAT variants

- Adaptive G2WSAT
- Adaptive G2WSAT+p
- Adaptive Novelty+
- Conflict-Directed Random Walk
- DDFW: Divide and Distribute Fixed Weights
- Deterministic Conflict-Directed Random Walk
- Deterministic Adaptive Novelty+
- G2WSAT: Gradient-based Greedy WalkSAT
- G2WSAT+p: Gradient-based Greedy WalkSAT with look-ahead
- GSAT: Greedy Search for SAT
- GSAT/TABU: GSAT with Tabu search
- GWSAT: GSAT with Random Walk
- HSAT: GSAT with History Information

Weighted MAX-SAT solvers

- HWSAT: HSAT with Random Walk
- IRoTS: Iterated Robust TABU Search
- Novelty
- Novelty+: Novelty with Random Walk
- Novelty++: Novelty with Diversification Probability
- Novelty+p: Novelty+ with look-ahead
- PAWS: Pure Additive Weighting Scheme
- RoTS: Robust Tabu Search
- R-Novelty
- R-Novelty+: R-Novelty with Random Walk
- RGSAT: Restarting GSAT
- RSAPS: Reactive SAPS
- SAMD: Steepest Ascent Mildest Descent
- SAPS: Scaling and Probabilistic Smoothing
- SAPS/NR: De-randomized version of SAPS
- Uniform Random Walk
- VW1: Variable Weighting Scheme One
- VW2: Variable Weighting Scheme Two
- WalkSAT
- WalkSAT/TABU: WalkSAT with TABU search

Outline

Choosing parents incurs a cost, but we must choose

- Create atoms: "X_i has parent set Pa"
- Create weighted clauses: $-\mathrm{Score}_i(\mathrm{Pa}) : \overline{X_i \text{ has parent set } \mathrm{Pa}}$
- Create 'hard' clauses:
  $(X_i \text{ has parent set } \mathrm{Pa}_{i1}) \vee (X_i \text{ has parent set } \mathrm{Pa}_{i2}) \vee \cdots \vee (X_i \text{ has parent set } \mathrm{Pa}_{im_i})$
- Choosing parents for each variable determines the DAG.

Ruling out cycles with a total order

- Encode variable orderings as well as DAGs (à la Friedman and Koller)
- Create $n(n-1)/2$ atoms: $\mathrm{ord}(X_i, X_j)$ meaning X_i and X_j are lexicographically ordered in the variable ordering.
- Create hard clauses:

$$\begin{aligned} X_j \text{ has parent set } \{X_i, X_k\} &\rightarrow \mathrm{ord}(X_i, X_j) \\ X_j \text{ has parent set } \{X_i, X_k\} &\rightarrow \overline{\mathrm{ord}(X_j, X_k)} \end{aligned}$$

- Create $n(n-1)(n-2)/3$ hard clauses:

$$\overline{\mathrm{ord}(X_i, X_j)} \vee \overline{\mathrm{ord}(X_j, X_k)} \vee \mathrm{ord}(X_i, X_k)$$
$$\mathrm{ord}(X_i, X_j) \vee \mathrm{ord}(X_j, X_k) \vee \overline{\mathrm{ord}(X_i, X_k)}$$

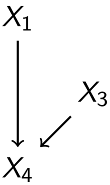
Outline

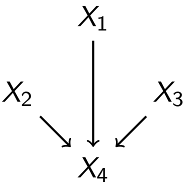
Pre-computing scores

- All weighted MAX-SAT solvers (that I know of) require all weights to be known before solving begins.
- So compute and store $\mathrm{Score}_i(\mathrm{Pa})$ for every variable *i* and candidate parent set Pa.
- I used a limit of 3 parents.
- With their more efficient code (and 4 dual-core machines) Silander and Myllymäki's bene system took 6 hours 16 minutes to compute *all* parent scores when there were 29 variables.
- In an example with 17 variables bene took under 18 seconds.

Filtering 'family' scores

If

Score_4 (


) $>$ Score_4 (


)

then throw RHS score away.

Outline

BNs and datasets

- Datasets of size 100, 1000 and 10000 were produced by forward sampling from the following 7 BNs.

Name	n	max $\|\mathbf{Pa}\|$	max r
Mildew	35	3	100
Water	32	5	4
alarm	37	4	4
asia	8	2	2
carpo	60	5	4
hailfinder	56	4	11
insurance	27	3	5

Size of WCNF

- These are sizes for an alternative encoding using a partial order over variables.

Data	atoms	clauses	lits
ca_2	8,609	226,406	661,551
ca_3	7,368	221,365	651,469
ca_4	19,932	269,367	747,473
ha_2	3,325	170,009	509,305
ha_3	3,842	171,400	512,087
ha_4	6,849	181,545	532,377
in_2	982	18,926	56,049
in_3	1,477	20,346	58,889
in_4	4,355	30,344	78,885

The MaxWalkSAT algorithm

```
while still_trying:
    somehow_assign_truth_values_to_all_atoms
    while cost <= target:
        c = random_choice(unsat_clauses)
        lits = lits_of(c)
        if random_flip:
            lit = random_choice(lits)
        else:
            lit = lowest_cost_flip(lits)
        flip_truth_value(lit)
        update_cost
```

Running MaxWalkSAT

```
newmaxwalksat version 20 (Huge)
seed = 99955222
cutoff = 10000000
tries = 100
numsol = 1
targetcost = 503040
heuristic = best, noise 50 / 100, init initfile
allocating memory...
clauses contain explicit costs
numatom = 6848, numclause = 181544, numliterals = 529296
wff read in
                                           average                    average        mean
   lowest     worst    number                 when                       over       flips
     cost    clause    #unsat     #flips     model     success            all       until
 this try  this try  this try   this try     found        rate          tries      assign
   506076     16968        56   10000000         *           0              *           *
   501973     23318        56    2913803   2913803          50       12913803  12913803.0

total elapsed seconds = 75.428415
average flips per second = 171206
number of solutions found = 1
mean flips until assign = 12913803.000000
mean seconds until assign = 75.428415
mean restarts until assign = 2.000000
ASSIGNMENT ACHIEVING TARGET 503040 FOUND
```

Nature of the search space

- If the current assignment of truth values to the atoms breaks at least one hard clause, then this assignment does *not* correspond to a DAG.
- The search (temporarily) visits cyclic graphs and 'graphs' were a variable's parent set may be undefined.
- Breaking hard constraints is OK; they will be fixed eventually.

Outline

Searching for high scoring BNs

Data	True	Ancestor	Total order	Long	> True
Mi_2	-7,786	-5,711	-5,708	-5,705	Y
Mi_3	-63,837	-47,229	-47,194	-47,120	Y
Mi_4	-470,215	-409,641	-410,159	-408,282	Y
Wa_2	-1,801	-1,488	-1,486	-1,484	Y
Wa_3	-13,843	-13,293	-13,284	-13,247	Y
Wa_4	-129,655	-129,274	-128,916	-128,812	Y
al_2	-1,410	-1,368	-1,368	-1,336	Y
al_3	-11,305	-11,599	-11,501	-11,339	N
al_4	-105,303	-107,205	-106,503	-105,907	N

Searching for high scoring BNs

Data	True	Ancestor	Total order	Long	> True
as_2	-247	-241	-241	-241	Y
as_3	-2,318	-2,312	-2,312	-2,312	Y
as_4	-22,466	-22,462	-22,462	-22,462	Y
ca_2	-1,969	-1,849	-1,852	-1,824	Y
ca_3	-17,739	-17,938	-17,891	-17,731	Y
ca_4	-173,682	-175,832	-176,456	-174,605	N

Outline

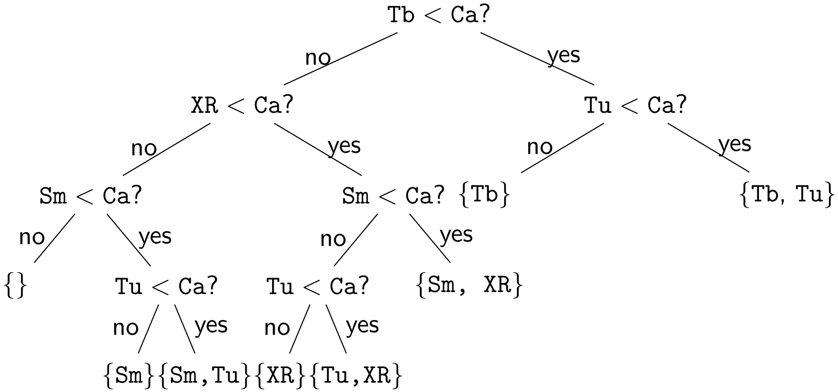
Working directly on total orders

- Given a total ordering the best parents for each variable are easy to find.

Parent sets for `Cancer`

{Tb, Tu}	-2.24772935188
{Tb}	-3.00976537207
{Sm, XR}	-8.07036732971
{Tu, XR}	-9.37534407212
{XR}	-9.38063760741
{Sm, Tu}	-21.6756460345
{Sm}	-21.6903150436
{}	-25.2333385745

Decision tree for choosing parents



Encoding as WCNF

$$
\begin{array}{rcl}
2 & : & \overline{(\mathtt{Tb} < \mathtt{Ca})} \vee \overline{(\mathtt{Tu} < \mathtt{Ca})} \\
3 & : & \overline{(\mathtt{Tb} < \mathtt{Ca})} \vee (\mathtt{Tu} < \mathtt{Ca}) \\
8 & : & (\mathtt{Tb} < \mathtt{Ca}) \vee \overline{(\mathtt{XR} < \mathtt{Ca})} \vee \overline{(\mathtt{Sm} < \mathtt{Ca})} \\
9 & : & (\mathtt{Tb} < \mathtt{Ca}) \vee \overline{(\mathtt{XR} < \mathtt{Ca})} \vee (\mathtt{Sm} < \mathtt{Ca}) \vee \overline{(\mathtt{Tu} < \mathtt{Ca})} \\
9 & : & (\mathtt{Tb} < \mathtt{Ca}) \vee \overline{(\mathtt{XR} < \mathtt{Ca})} \vee (\mathtt{Sm} < \mathtt{Ca}) \vee (\mathtt{Tu} < \mathtt{Ca}) \\
21 & : & (\mathtt{Tb} < \mathtt{Ca}) \vee (\mathtt{XR} < \mathtt{Ca}) \vee \overline{(\mathtt{Sm} < \mathtt{Ca})} \vee \overline{(\mathtt{Tu} < \mathtt{Ca})} \\
21 & : & (\mathtt{Tb} < \mathtt{Ca}) \vee (\mathtt{XR} < \mathtt{Ca}) \vee \overline{(\mathtt{Sm} < \mathtt{Ca})} \vee (\mathtt{Tu} < \mathtt{Ca}) \\
25 & : & (\mathtt{Tb} < \mathtt{Ca}) \vee (\mathtt{XR} < \mathtt{Ca}) \vee (\mathtt{Sm} < \mathtt{Ca})
\end{array}
$$

Initial results with ‘order-only’ encoding

- Using the irots solver and the new encoding get:
  - Score of -132,951 for insurance 10,000 dataset. Beats best previous score of -133,934 and score of true BN which is -133,489.
  - Score of -497,652 for hailfinder 10,000 dataset. Beats best previous score of -498,739.