

# Bayesian network learning by compiling to weighted MAX-SAT

James Cussens, University of York

SML, 2008-04-25

Work to be presented at UAI '08

# Outline

Introduction

Learning Bayesian networks

Weighted MAX-SAT

Compiling to weighted MAX-SAT

Results

# Motivation

- ▶ A similar motivation to Siegfried ...
- ▶ Why write (yet another) BN learning algorithm when weighted SAT search can do it?
- ▶ It's hard to get MCMC to converge for Bayesian model averaging so use search to do it.

# Outline

Introduction

**Learning Bayesian networks**

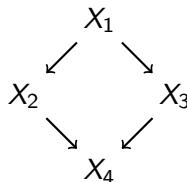
Weighted MAX-SAT

Compiling to weighted MAX-SAT

Results

## Bayesian network learning

$X_1$	$X_2$	$X_3$	$X_4$
0	1	0	0
1	1	0	0
0	0	1	1
1	0	1	0
0	1	1	1
0	1	0	0
1	1	0	0
0	0	1	1
1	0	1	0
0	1	1	1



## BN structure learning by maximising the (marginal) likelihood

- ▶ Marginal likelihood:  $P(D|G) = \int_{\theta} P(D|G, \theta)P(\theta)d\theta$
- ▶  $\text{Score}(G) \stackrel{\text{def}}{=} \log P(D|G)$ .
- ▶ Search for  $\arg \max_G \text{Score}(G)$
- ▶ Looking for DAGs  $G$  such that  $\text{Score}(G)$  is a small negative number.

## Scoring Bayesian networks

$$\text{Score} \left( \begin{array}{c} X_1 \\ \swarrow \quad \searrow \\ X_2 \quad X_3 \\ \searrow \quad \swarrow \\ X_4 \end{array} \right) = \text{Score}_1 \left( \begin{array}{c} X_1 \end{array} \right) + \dots$$

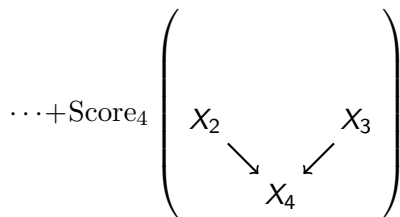
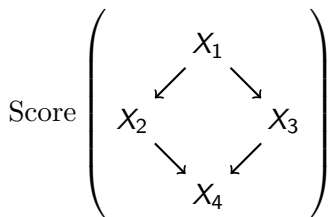
## Scoring Bayesian networks

$$\text{Score} \left( \begin{array}{c} X_1 \\ \swarrow \quad \searrow \\ X_2 \quad X_3 \\ \searrow \quad \swarrow \\ X_4 \end{array} \right) \quad \dots + \text{Score}_2 \left( \begin{array}{c} X_1 \\ \swarrow \\ X_2 \end{array} \right) + \dots$$

# Scoring Bayesian networks

$$\text{Score} \left( \begin{array}{c} X_1 \\ \swarrow \quad \searrow \\ X_2 \quad X_3 \\ \searrow \quad \swarrow \\ X_4 \end{array} \right) \quad \dots + \text{Score}_3 \left( \begin{array}{c} X_1 \\ \searrow \\ X_3 \end{array} \right) + \dots$$

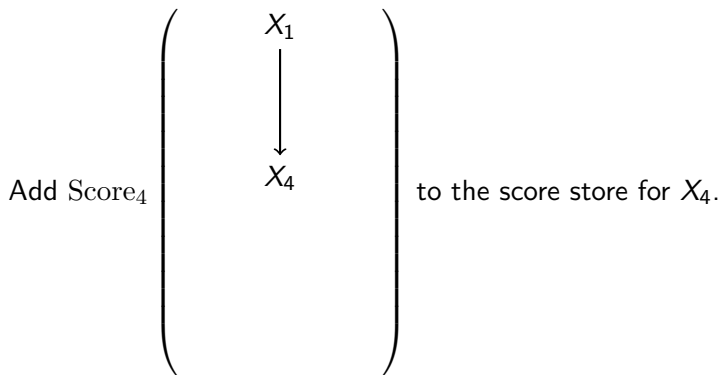
## Scoring Bayesian networks



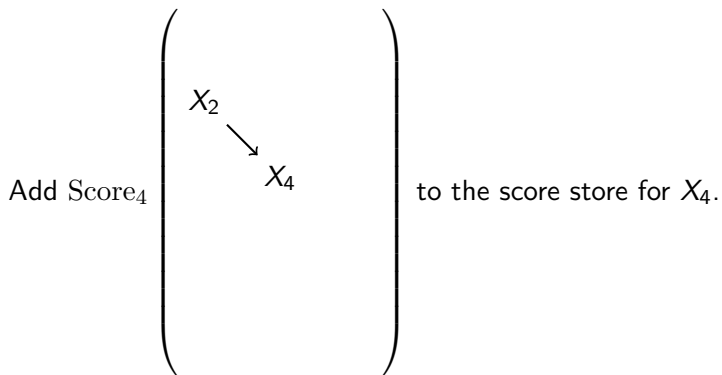
## Pre-computing 'family' scores (e.g. for $X_4$ )

Add  $\text{Score}_4$   $X_4$  to the score store for  $X_4$ .

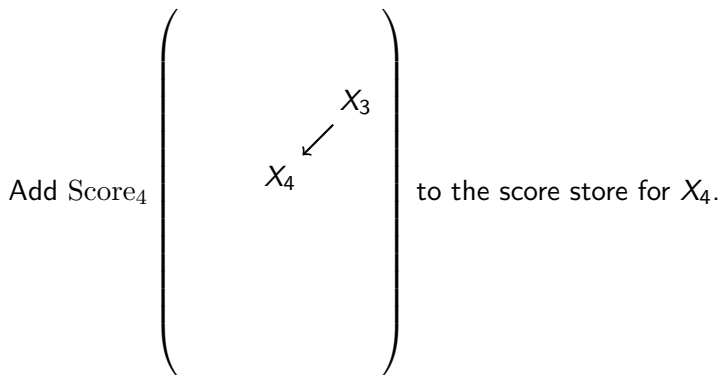
## Pre-computing 'family' scores (e.g. for $X_4$ )



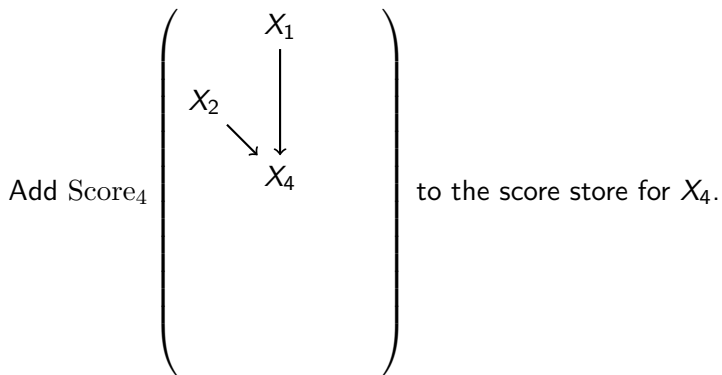
## Pre-computing 'family' scores (e.g. for $X_4$ )



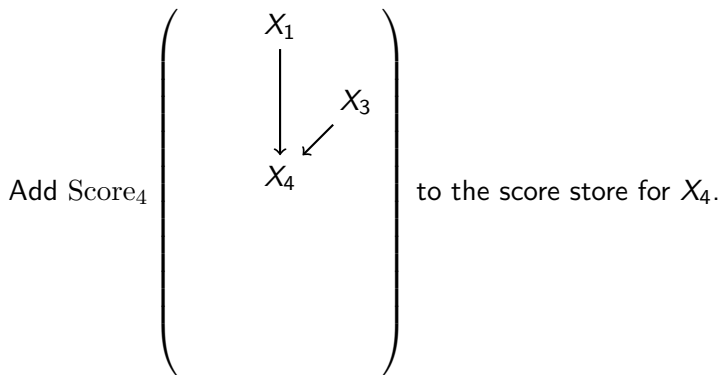
## Pre-computing 'family' scores (e.g. for $X_4$ )



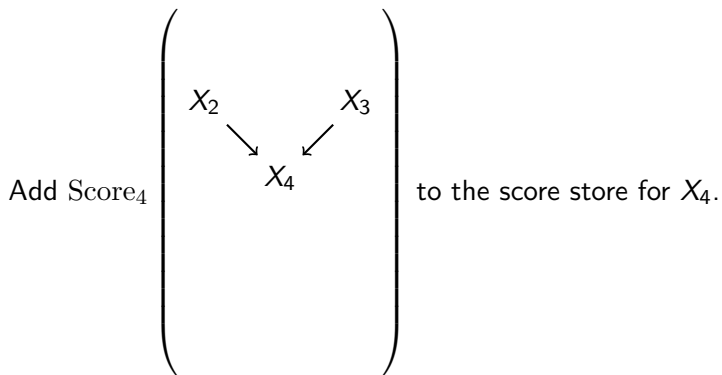
## Pre-computing 'family' scores (e.g. for $X_4$ )



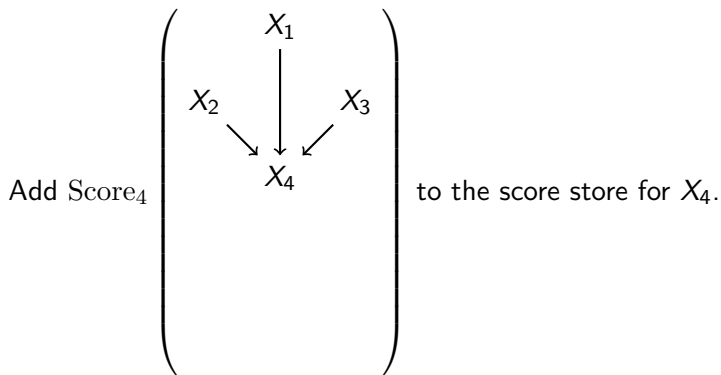
## Pre-computing 'family' scores (e.g. for $X_4$ )



## Pre-computing 'family' scores (e.g. for $X_4$ )



## Pre-computing 'family' scores (e.g. for $X_4$ )



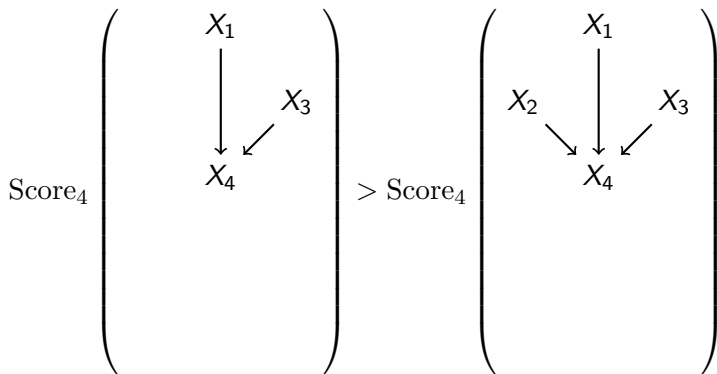
## Family scores for some datasets

With a limit of at most 3 parents for each variable.

Name	$n$	scores	Time taken for $N = \dots$		
			$10^2$	$10^3$	$10^4$
Mildew	35	230,300	1,678	1,942	4,502
Water	32	159,744	22	123	1,093
alarm	37	288,859	38	208	1,501
asia	8	512	0	0	0
carpo	60	2,056,800	544	1,811	13,892
hailfinder	56	1,555,456	852	2,974	22,666
insurance	27	79,704	29	97	749

## Filtering 'family' scores

If



then throw RHS score away.

## Results of filtering

Dat	$N = \dots$					
	$10^2$		$10^3$		$10^4$	
	max	$n$	max	$n$	max	$n$
Mi	977	3,515	17	163	47	465
Wa	44	482	44	573	49	961
al	75	907	184	1,928	473	6,473
as	10	41	24	107	31	161
ca	350	5,068	352	3,827	2,089	16,391
ha	22	244	77	761	435	3,768
in	28	279	95	774	496	3,652

## BN learning as constrained optimisation

- ▶ Why not just pick the highest scoring parents for each variable?
- ▶ Because (probably) that would produce a *cyclic* graph.
- ▶ BN learning is a finite domain constraint problem.
  - ▶ Map each BN variable to a constrained variable; its candidate parent sets are its values.
  - ▶ Infeasible assignments are those which cause cycles.

# Outline

Introduction

Learning Bayesian networks

**Weighted MAX-SAT**

Compiling to weighted MAX-SAT

Results

# Motivation

- ▶ It's quite a big problem.
- ▶ So try out an approach that has been shown to work on big problems.

# SAT, MAX-SAT and weighted MAX-SAT

**SAT** Given propositional clauses, set the truth values of the atoms to satisfy all clauses; or show that no satisfying assignment exists.

**MAX-SAT** Given propositional clauses, satisfy as many clauses as possible.

**Weighted MAX-SAT** Given weighted propositional clauses, maximise the sum of the weights of satisfied clauses.

**Weighted MAX-SAT** (Equivalently) Minimise the sum of the weights of unsatisfied clauses.

# 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
```

# Outline

Introduction

Learning Bayesian networks

Weighted MAX-SAT

**Compiling to weighted MAX-SAT**

Results

## Encoding: the atoms

- ▶ For each pair of atoms  $a$ ,  $b$ , whether  $a$  is an ancestor of  $b$  is represented by a propositional atom.
- ▶ Each choice of parents for each variable is represented by a propositional atom.

c Atom 1 means Bronchitis  $\rightarrow$  Cancer

c Atom 2 means Bronchitis  $\rightarrow$  Dyspnea

c Atom 118 means that Cancer has exactly these parents:  
Dyspnea, Tuberculosis, Bronchitis.

## Encoding: the hard clauses

- ▶ For all  $a, b, c$ :  $(a \rightarrow b) \wedge (b \rightarrow c) \Rightarrow (a \rightarrow c)$
- ▶ Every variable must have some choice of parents.

```
c Clause -1 -9 2: Bronchitis ->> Cancer
                  & Cancer ->> Dyspnea
                  => Bronchitis ->> Dyspnea
c Clause 58 59 ... 121:
                  since exactly one family for Cancer
```

## Encoding: the hard clauses (ctd.)

- ▶ Choosing parents implies ancestor relations: e.g. if  $a$  is chosen as one of the parents of  $b$ , then  $a$  is an ancestor of  $b$ .
- ▶ **No cycles**  $\forall a, b : \neg(a \rightarrow b) \vee \neg(b \rightarrow a)$

- c Clause -118 1: Cancer has parents  
Dyspnea, Tuberculosis, Bronchitis  
=> Bronchitis ->> Cancer
- c Clause -8 -9: not(Cancer ->> Bronchitis) or  
not (Bronchitis ->> Cancer)

## Encoding: the soft clauses

- ▶ For each choice of parents for each variable assert that the choice is false (a single negative literal clause).
  - ▶ But *weight* each such clause with  $-1$  times the score for this choice.
- c Soft clause 30 -118: since Cancer  
with parents Dyspnea, Tuberculosis, Bronchitis  
has score -30.275543 and scaling is 1

## Size of CNF problems

Data	atoms	clauses	lits
as_2	98	488	1,351
as_3	164	693	1,761
as_4	218	873	2,121
ca_2	8,609	226,406	661,551
ca_3	7,368	221,365	651,469
ca_4	19,932	269,367	747,473
in_2	982	18,926	56,049
in_3	1,477	20,346	58,889
in_4	4,355	30,344	78,885

# Outline

Introduction

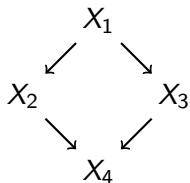
Learning Bayesian networks

Weighted MAX-SAT

Compiling to weighted MAX-SAT

**Results**

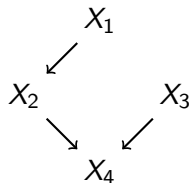
## Synthetic data



Sample  
 $\Rightarrow$

$X_1$	$X_2$	$X_3$	$X_4$
0	1	0	0
1	1	0	0
0	0	1	1
1	0	1	0
0	1	1	1
0	1	0	0
1	1	0	0
0	0	1	1
1	0	1	0
0	1	1	1

Learn  
 $\Rightarrow$



## It's fast

```
newmaxwalksat version 20 (Huge)  
numatom = 6848, numclause = 181544, numliterals = 529296
```

lowest	worst	number		
cost	clause	#unsat	#flips	
this try	this try	this try	this try	
506076	16968	56	10000000	
501973	23318	56	2913803	

```
total elapsed seconds = 75.428415  
average flips per second = 171206  
number of solutions found = 1  
ASSIGNMENT ACHIEVING TARGET 503040 FOUND
```

## Searching for high scoring BNs

Data	True	Target	no cycle_atom tries=100
Mi_2	-7,786	-6,611	-5,711
Mi_3	-63,837	-52,866	-47,229
Mi_4	-470,215	-459,874	-409,641
Wa_2	-1,801	-1,523	-1,488
Wa_3	-13,843	-13,304	-13,293
Wa_4	-129,655	-128,745	-129,274
al_2	-1,410	-1,383	-1,368
al_3	-11,305	-11,255	-11,599
al_4	-105,303	-105,226	-107,205

## Searching for high scoring BNs

Data	True	Target	no cycle_atom tries=100
as_2	-247	-245	-241
as_3	-2,318	-2,318	-2,312
as_4	-22,466	-22,466	-22,462
ca_2	-1,969	-1,952	-1,849
ca_3	-17,739	-17,821	-17,938
ca_4	-173,682	-175,349	-175,832

## Searching for high scoring BNs

Data	True	Target	no cycle_atom tries=100
ha_2	-6,856	-6,058	-5,998
ha_3	-55,366	-52,747	-53,059
ha_4	-503,040	-498,163	-506,219
in_2	-1,951	-1,715	-1,690
in_3	-14,411	-13,919	-14,105
in_4	-133,489	-132,968	-134,378

## Bayesian model averaging

- ▶ Just collect BNs (truth assignments) generated in the search  
...
- ▶ ... together with their scores (likelihoods).
- ▶ (And pipe through `sort -u`.)
- ▶ Just give zero probability to all other networks and normalise to get a posterior distribution over networks.
- ▶ Excellent results for asia, generally bad in other cases.