Informative Priors for Graphical Model Structure

James Cussens, University of York
jc@cs.york.ac.uk
(joint work with Nicos Angelopoulos)

Supported by the UK EPSRC MATHFIT programme

Use of structural priors

- "The use of structural priors when learning BNs has received only little attention in the learning community."

  (Langseth & Nielsen, 2003)

- "The standard priors over network structures are often used not because they are particularly well-motivated, but rather because they are simple and easy to work with. In fact, the ubiquitous uniform prior over structures is far from uniform over [Markov equivalence classes]"

  (Friedman & Koller, 2003)

Exploiting experts

". . . in the context of knowledge-based systems, or indeed in any context where the primary aim of the modeling effort is to predict the future, [uniform] prior distributions are often inappropriate; one of the primary advantages of the Bayesian approach is that it provides a practical framework for harnessing all available resources including prior expert knowledge."

(Madigan et al, 1995)

The problem with experts

"Notwithstanding the preceding remarks, eliciting an informative prior distribution on model space from a domain expert is challenging." (Madigan et al, 1995)

Hard constraints

- Imposing a total ordering on variables or blocks
- Limiting the number of parents
- Banning/requiring specific edges.

Assuming link independence

$$\mathrm{pr}(M) \propto \prod_{e \in \mathcal{E}_P} \mathrm{pr}(e) \prod_{e \in \mathcal{E}_A} (1 - \mathrm{pr}(e))$$

(Buntine, 1991; Cooper & Herskovits, 1992;
Madigan and Raftery, 1994)

Edit distance from prior network

Let M differ from the expert's prior network by δ arcs, then

$$\text{pr}(M) = c\kappa^{\delta}$$

($\approx$ Madigan and Raftery, 1994; Heckerman et al, 1995)

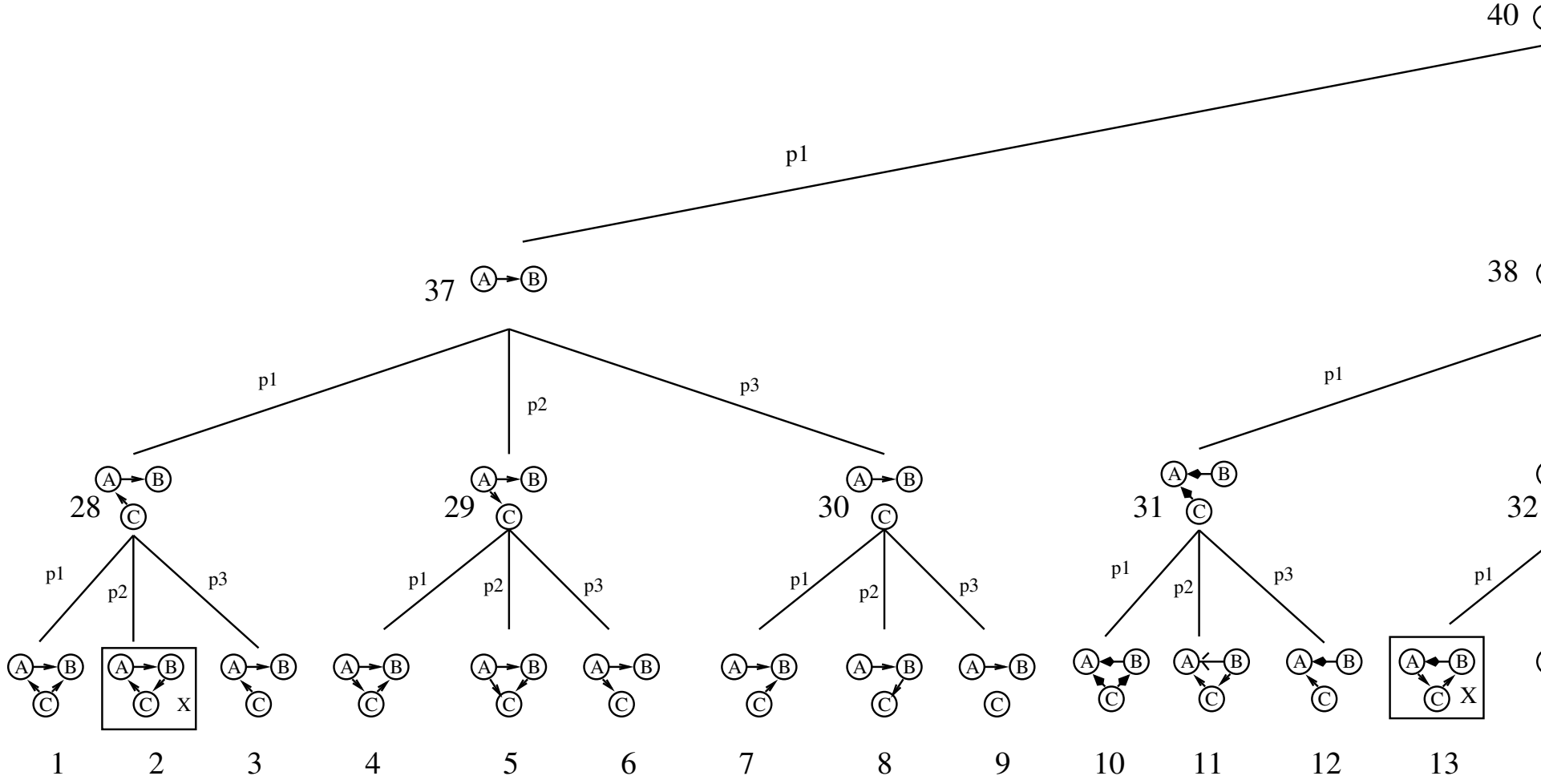
Priors over CART trees

- A Bayesian CART algorithm. Denison et al, *Biometrika* 1998

- Bayesian CART model search. Chipman et al, *JASA* 1998

"Instead of specifying a closed-form expression for the tree prior, $p(T)$, we specify $p(T)$ implicitly by a tree-generating stochastic process. Each realization of such a process can simply be considered a random draw from this prior. Furthermore, many specifications allow for straightforward evaluation of $p(T)$ for any T and can be effectively coupled with efficient Metropolis-Hastings search algorithms . . ." (Denison et al)

A graphical-model-generating stochastic process



Stochastic logic programs implement model-generating stochastic processes

1. Write a logic program which defines a set of models:

   - BN is a Bayesian network if . . .
   - $\forall BN : bn(BN) \leftarrow digraph(BN) \wedge acyclic(BN)$
   - `bn(BN) :- digraph(BN), acyclic(BN).`

2. Add parameters to define distribution over models to get a stochastic logic program (SLP).

SLPs for MCMC

- The tree gives a natural neighbourhood structure to the model space . . .

- . . . which we exploit to construct a proposal distribution based on the prior.

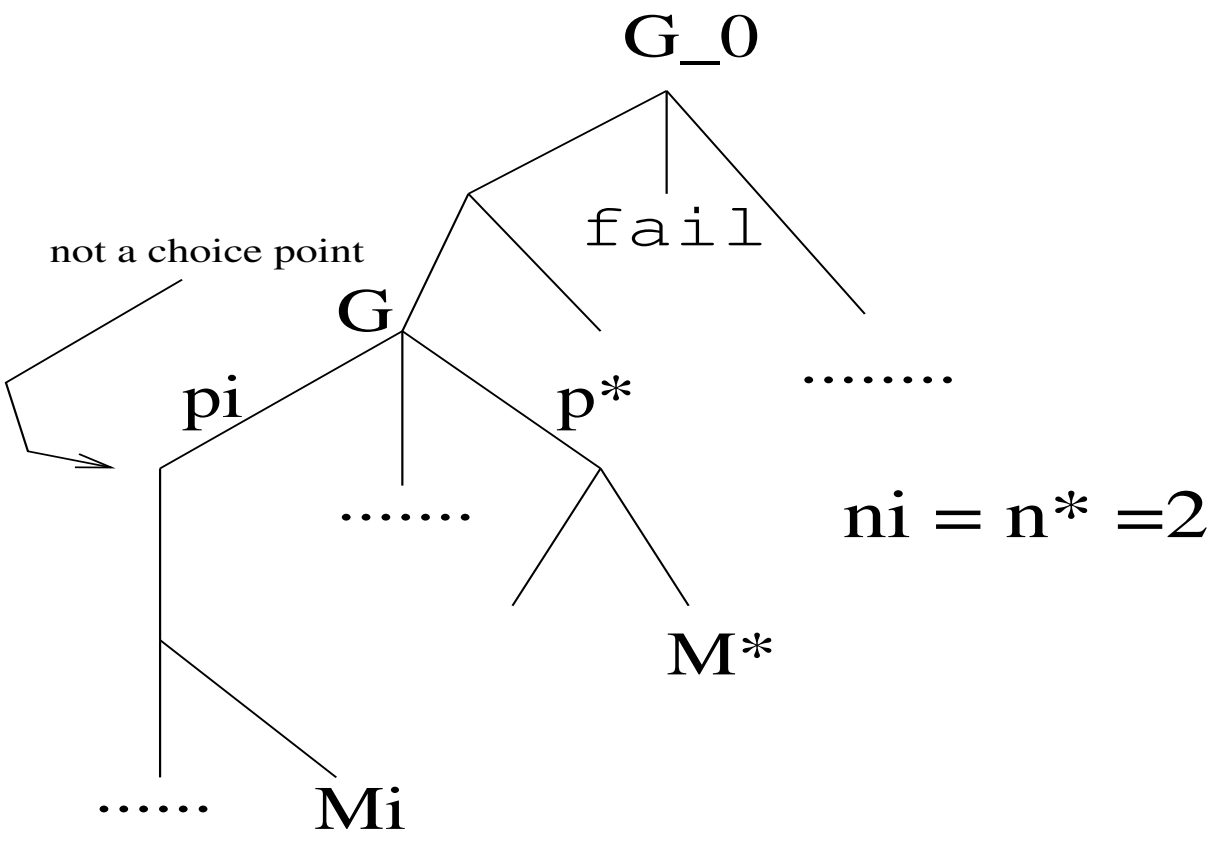
The proposal mechanism

1. Backtrack one step to the most recent choice point in the probability tree

2. We then probabilistically backtrack as follows: If at the top of the tree stop. Otherwise backtrack one more step to the next choice point with probability p_b.

3. Once we have stopped backtracking choose a new leaf/model M^* from the choice point by selecting branches according to their probabilities attached to them. However, in the first step down the tree we may not choose the branch that leads back to the current leaf/model M^i.

Bouncing around the tree



The acceptance probability

If M^* is a failure then $\alpha(M^i, M^*) = 0$ else:

$$\alpha(M^i, M^*) = \min\left\{p_b^{(n^*-n^i)}\frac{1-p^i}{1-p^*}\frac{P(D|M^*)}{P(D|M^i)}, 1\right\}$$

Better mixing with a cyclic transition kernel

- We cycle through the values $p_b = 1 - 2^{-n}$, for $n = 1, \ldots, 28$,
- so that on every 28th iteration, there is a high probability of backtracking all the way to the top of the tree.

It works . . . eventually!

M	$\widehat{p_4}$	$\widehat{p_5}$	$\widehat{p_6}$	p
BN_{22}	0.668	0.690	0.704	0.702
BN_{20}	0.176	0.150	0.145	0.146
BN_{19}	0.144	0.152	0.143	0.145
BN_4	0.007	0.005	0.005	0.005
BN_5	0.002	0.001	0.002	0.002
BN_1	0.001	0.001	0.001	0.001
BN_{14}	0	0	0	0
BN_{10}	0.001	0	0	0
BN_{11}	0	0	0	0

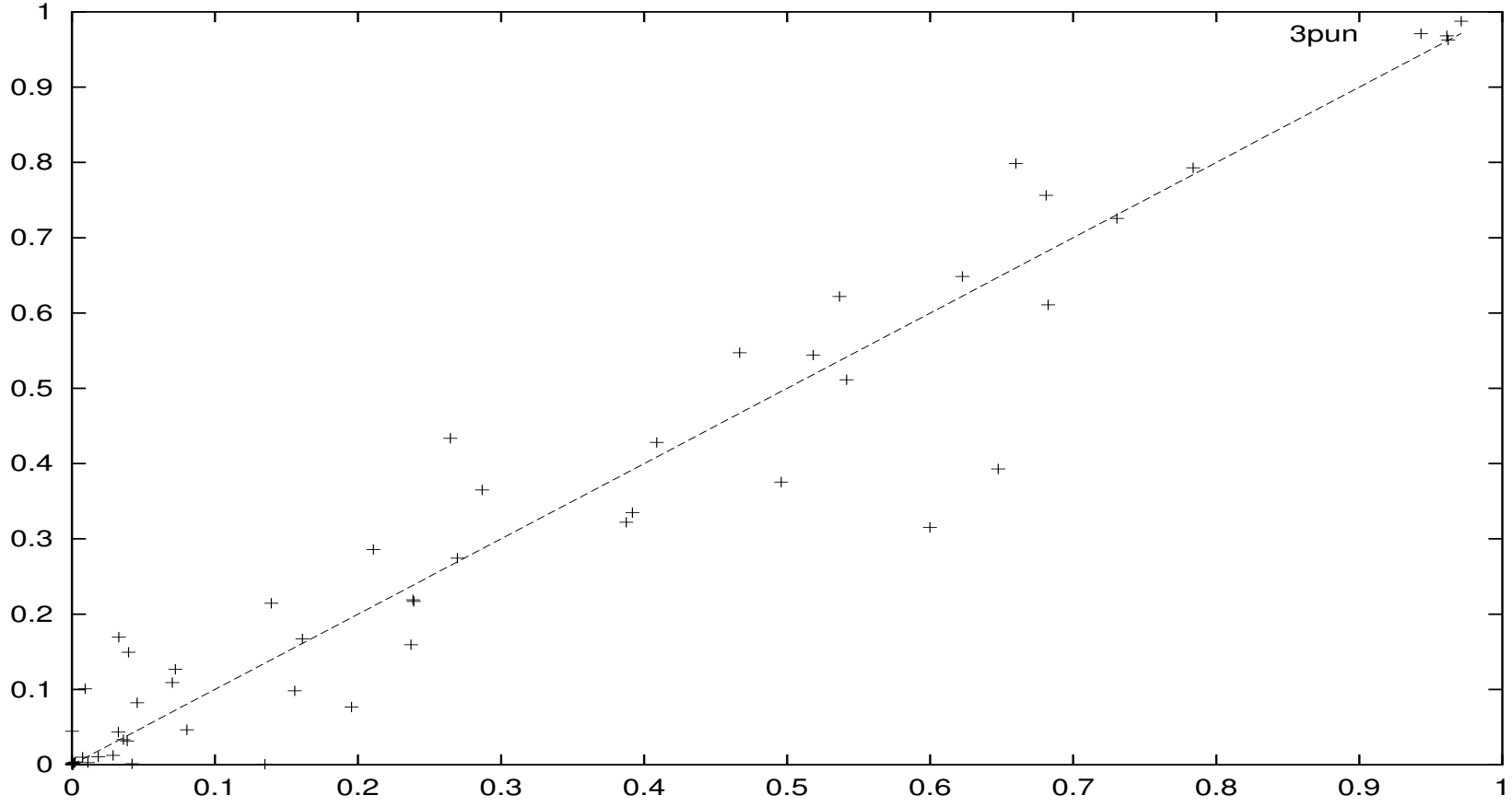
Estimated ($\widehat{p_i}$) and actual (p) posterior probabilities for the nine most probable 3-node BNs in $BNTREE$.

$\widehat{p_i}$ is the estimated probability after 10^i iterations.

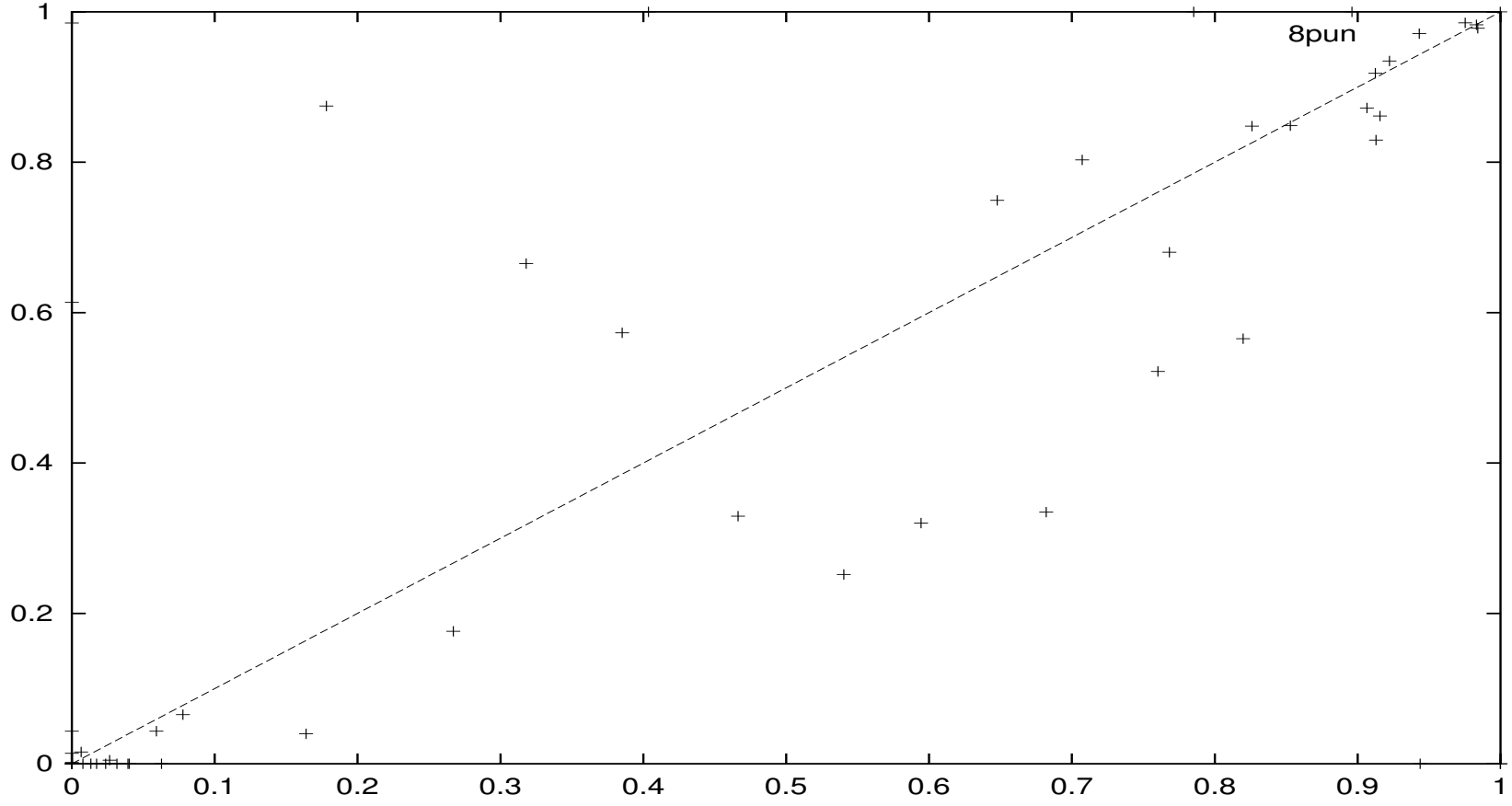
Real evaluation

- Generate 2295 datapoints from the *Asia* BN
- 783,702,329,343 BNs in model space
- Run MCMC for 500,000 iterations (no burn-in)
- Runtimes: 24 minutes - 55 minutes
- 2 runs for each 'setting': compare observed probabilities

Real evaluation - OK results



Real evaluation - hmmm



Markov equivalence classes

```
bn(RVs,BN) :-
        skeleton(RVs,Skel),
        essential_graph(Skel,Imms,EG), %could stop here
        bn(EG,Imms,BN),
        top_sort(BN,_).  %check for cycles
```

Way too many failures!

Logic program transformation

```
member(X,[X|_]).
member(X,[_|T]) :- member(X,T).
```

```
?- member(X,List),List=[_,_,_]
```

```
member(X,[X,_,_]).
member(X,[_,X,_]).
member(X,[_,_,X]).
```

SLP transformation for more efficient sampling

```
1/2 : member(X,[X|_]).
1/2 : member(X,[_|T]) :- member(X,T).
```

```
?- member(X,List),List=[_,_,_]
```

```
4/7 : member(X,[X,_,_]).
2/7 : member(X,[_,X,_]).
1/7 : member(X,[_,_,X]).
```

What about R?

- R calls C calls Prolog
- Where does the prior live? as an R object?
- The data should eventually be an R dataframe
- Begin with R as a 'wrapper'.