

# Exploiting independence for branch operations in Bayesian learning of C&RTs

Nicos Angelopoulos, James Cussens

University of York, UK

**Abstract.** In this paper we extend a methodology for Bayesian learning via MCMC, with the ability to grow arbitrarily long branches in C&RT models. We are able to do so by exploiting independence in the model construction process. The ability to grow branches rather than single nodes has been noted as desirable in the literature. The most singular feature of the underline methodology used here in comparison to other approaches is the coupling of the prior and the proposal. The main contribution of this paper is to show how taking advantage of independence in the coupled process, can allow branch growing and swapping for proposal models.

**Keywords.** Bayesian machine learning, classification and regression trees, stochastic logic programs

## 1 Introduction

In MCMC, Markov chain Mode Carlo, model learning [?,?], a prior and a proposal are exploited to derive a posterior over the space of all possible models explaining the data at hand. In most MCMC approaches the prior features as a computable function over models and it is only the proposal which is a process. That is a process generating one model, the *proposed* at each iteration, from the current model. Such approaches often allow only local perturbations to take place. In this way, the computation of both the proposal probability and the prior are kept simple.

The focus here is on MCMC over C&RTs, [?,?,?]. [?] and [?] use very similar moves, which are local. [?, p. 368] states that:

Allowing large branches to be pruned would require a converse step which would grow a similar branch in one move. Neither move type would be accepted often because of the large change in the tree structure . . .

However, this is an artifact of the separation of prior and proposal noted above. The ability to add,delete and modify whole branches with probability proportional to the prior can lead to better exploration of the model space by avoiding getting stuck in local modes. [?, p. 942] notes:

Once a tree has reasonable fit, the chain is unlikely to move away from a sharp local mode by small steps.

In contrast [?] have introduced a methodology that couples the proposal to the prior and they are both seen as part of the same process, that of moving between leaves on a single tree encompassing all possible models. The original framework as introduced therein allows for longer moves but has no provision for operations over arbitrary branches of the C&RTmodel. Instead, the construction process has to sequentially be rewound to a chosen point before resampling a new model completed from that point onwards. Here we introduce an improvement to this methodology and present some preliminary results run on a prototype system.

The remainder of the paper is structured as follows. Section 2 briefly reviews the Metropolis-Hastings MCMC. Section 3 presents a prior from the literature and shows how it can be captured as an SLP. Section 4 discusses independent branching in MCMC over model structures. In Section 5 we present some preliminary experimental results. Finally, in Section 6 are our concluding remarks.

## 2 MCMC

Markov chain Monte Carlo (MCMC) algorithms [?] are commonly used to estimate  $P(M | D)$  (posterior probability of models given data  $D$ ) from a prior distribution  $p(M)$ . Provided some weak conditions are met, and that the Markov chain is run for a sufficiently long period, then the visiting frequencies approach the true posterior probability of models. The posterior probability of a model is the probability that the particular model is the ‘true’ model. For an introduction of MCMC for machine learning see [?]. In constructing the Markov chain we assume the Metropolis-Hasting algorithm

0. Set  $i = 0$  and find  $M_0$ , an initial leaf of the SLD-tree, using the prior.
1. From  $M_i$  produce a candidate leaf  $M_*$ . Let the probability of reaching  $M_*$  be  $q(M_i, M_*)$ .
2. Let

$$\alpha(M_i, M_*) = \min \left\{ \frac{q(M_*, M_i)P(D|M_*)P(M_*)}{q(M_i, M_*)P(D|M_i)P(M_i)}, 1 \right\}$$

$$M_{i+1} = \begin{cases} M_* & \text{with probability } \alpha(M_i, M_*) \\ M_i & \text{with probability } 1 - \alpha(M_i, M_*) \end{cases}$$

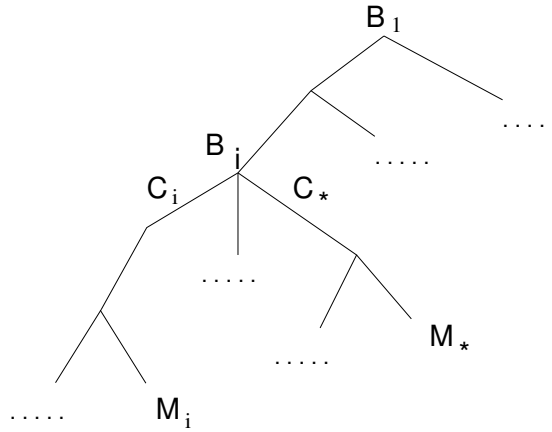
3. If  $i$  reached limit then terminate, else set  $i = i + 1$  and repeat from 1.

$q(M_i, M_*)$  is the transition probability, i.e. the probability of proposing  $M_*$  from  $M_i$ ,  $P(D | M)$  is the likelihood that the data were generated by  $M$  and  $P(M)$  is the prior probability of  $M$ .

This compellingly simple framework has been used by [?,?,?] to learn C&RTs. In these approaches the prior is a function on model structures and the proposal as a process of local moves that allow the growth, shrink, and change of/at a single node. Although the need of longer moves has been recognised as potentially beneficial to avoiding local modes of the posterior, the limitation to local moves is a pragmatic one, stemming from the separation of the proposal from the

prior. To keep the computation of the ratios efficient and the  $\alpha$  value to levels that encourage mixing in the chain, these approaches are compelled to use local moves.

[?] introduced a framework which integrates both the prior and the proposal into a single process that affords simplifications on the computation of  $\alpha$ . This framework facilitates longer moves, but suffers from the sequentiality of the building process. A diagram of the process is shown in Fig. 1. At iteration  $i$  we propose  $M_*$  from  $M_i$  by means of operations on the SLD-tree. First we backtrack to  $B_i$ , an ancestor node of  $M_i$ , and then sample from that node onwards. A general case of this is illustrated in Fig. 1. The exact way  $M_*$  is chosen can vary



**Fig. 1.** Reaching  $M_*$  from  $M_i$  in the SLD-tree.

Since it is operating on the tree of all models, the SLD-tree, the algorithm needs to undo all choices up to  $B_i$  irrespectively of what part of the model, here a C&RT, it had built. In Section 4 we will show how the algorithm can be enhanced to exploit independence in the building process. This will then be used to run MCMC experiments that allows operation over arbitrary branches.

### 3 A C&RT Prior

Our approach to defining priors is related to a line of research which was independently initiated by [?] and [?]. The basic idea is to specify a prior distribution over a space of models with a *stochastic program* rather than by some closed-form expression:

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. [?]

The prior used in [?] grows a C&RT tree by starting with a single leaf node and then repeatedly splits each leaf node  $\eta$  with a probability  $\alpha(1+d_\eta)^{-\beta}$ , where  $d_\eta$  is the depth of node  $\eta$  and  $\alpha$  and  $\beta$  are prior parameters set by the user to control the size of trees. Unsplit nodes become leaves of the tree. If a node is split, the splitting rule for that split is chosen uniformly. An abbreviated fragment of the SLP which expresses this prior is given in Table 1.

```

1 - Sp: [Sp]: cart( Data, D, A/B, leaf(Data) ).
Sp      : [Sp]: cart( Data, D, A/B, Node ) :-
          Node = node(F,V,L,R),
          branch( Data, F, V, LData, RData ),
          D1 is D + 1,
          NxtSp is A * ((1 + D1) ^ -B),
          [NxtSp] : cart( LData, D1, A/B, L ),
          [NxtSp] : cart( RData, D1, A/B, R ).

```

Table 1. SLP defined prior

The stochastic program can be used to generate term structures that represent C&RTmodels such as the ones shown in Fig. 2.

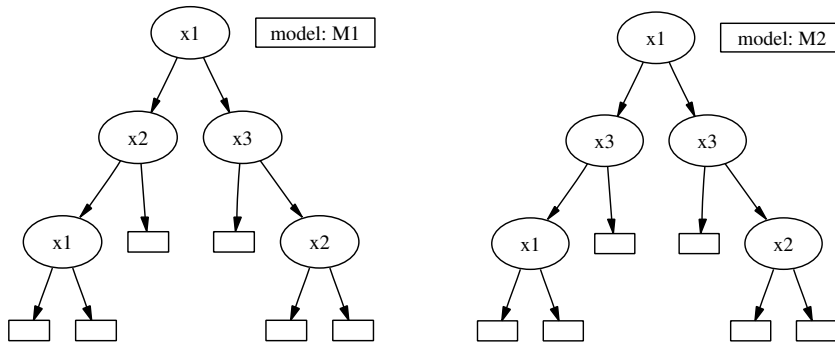


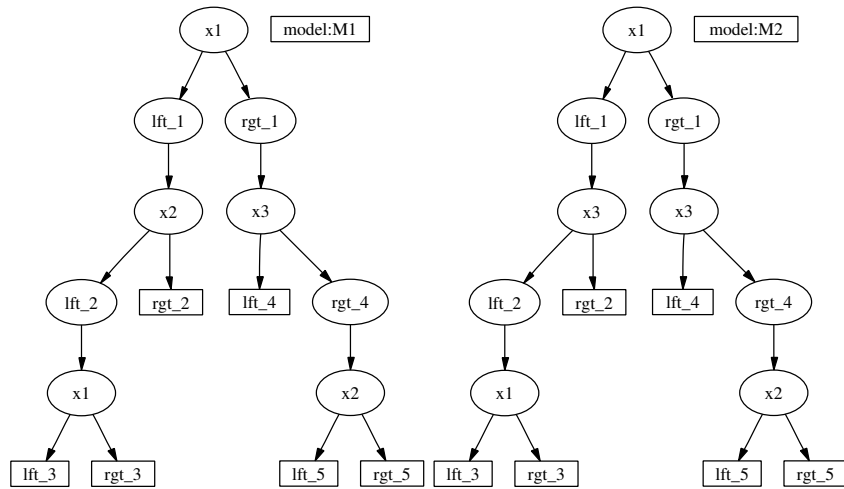
Fig. 2. Example C&RTmodels

## 4 Independent Branches

The models shown in Fig. 2 are, intuitively speaking, neighbours. This is because a single change on one node of one of the trees is all that is needed to produce

the other. In the work presented in [?] this requires rebuilding of branches that happen to be built after the node in question even if they are in part of the tree that is in branches that do not stem from the changed node.

In short, the last two lines of the program in Table 1 unnecessarily sequentialise the process of building the tree. In the way the program is executed, all choices for the left branch are considered to be *before* all the choices for the right branch.



**Fig. 3.** Proof trees for M1 and M2.

A path in the SLD tree can be viewed, in an alternative manner, as a proof tree, [?, p. 53]. The proof trees for  $M1$  and  $M2$  are shown in Fig. 3. Backtracking can then be seen as selecting internal nodes in these trees and providing alternative components to the proof.

Based on these ideas, we have allowed the user to declare by means of syntax such branches as independent. Execution can thus take advantage of the independence in such situations and effect proposal backtracking that take advantage of the structure of models.

## 5 Experiments

We have used the PIMA dataset to run some preliminary experiments. This dataset was also used in [?]. It contains 768 complete entries over 8 feature variables.

We run 250,000 iterations using the uniform choice backtracking proposal, and prior parameters:  $\alpha = .95$   $\beta = .8$ .

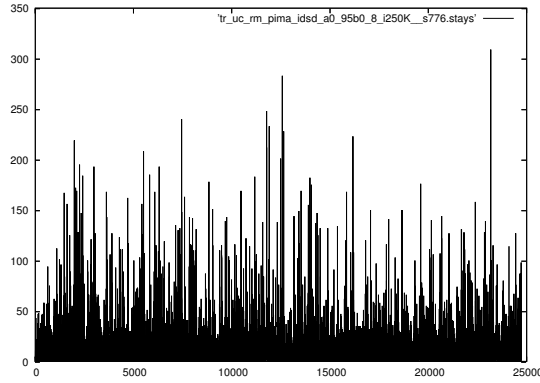


Fig. 5 shows the trajectory of stays. For model visited in the chain, we plot a bar whose  $y$  value is the number of iteration the chain stayed in that visit. It is clear that the chain does not spent inordinate amount of time in any specific model. Such a behaviour is usually a sign of poor mixing leading to inaccurate learning.

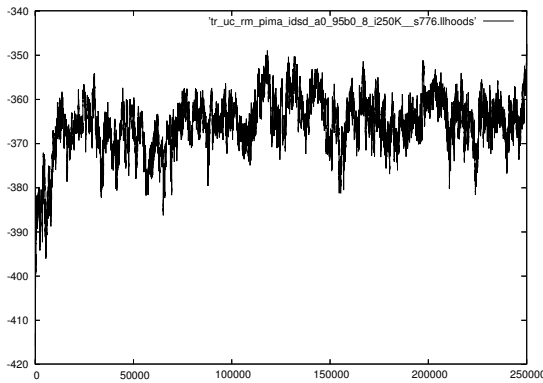
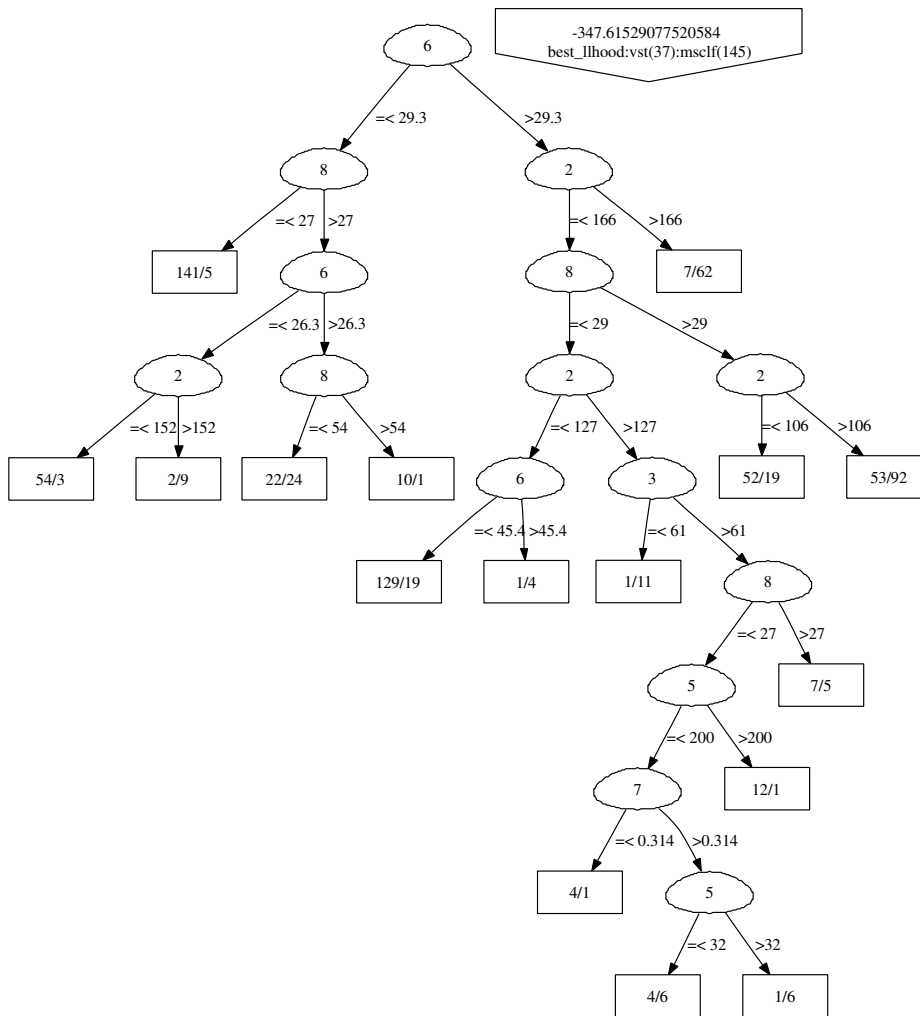


Fig. 5 shows the likelihood trajectory, where for each model visited we plot a point whose  $y$  value is the likelihood of the current model. The trajectory shown, shows a healthy movement to various likelihood neighbourhoods.

Finally Fig. 5 shows the model with best likelihood visited in the chain. Since we are concerned with learning a distribution over the models, the best likelihood model, should be misinterpreted as the learned model. In Bayesian approaches, model averaging can be used for finding such a model, or for doing statistical inference. The log-likelihood of the model in Fig 5 is -347.615.



## 6 Conclusions

We have shown that the coupling of prior and proposal can have another benefit apart from the ease of calculation of the acceptance probability  $\alpha$ . Namely, the common process can be used to operate on sub-parts of the model structure.

A particular case of operating on sub-parts of model structure was pursued further in the form of branch operation on C&RTmodels. This was possible by exploiting independence in the construction process. Our experimental results showed that the produced chains moved a good deal, and that most time was spent in high likelihood areas.

In the future we hope to experimentally quantify the gains from the independent branching when compared to the sequential case, and to automate the recognition of independent branches within our system, which is available from <http://www.cs.york.ac.uk/~nicos/sware/slps>.

## Acknowledgements

This work was supported by EPSRC grant GR/S30993/01 *Stochastic Logic Programs for MCMC*.