A PC-Style Markov Blanket Search for High Dimensional Datasets

Joe Ramsey

February 9, 2006

Technical Report No. CMU-PHIL-177

Department of Philosophy
Carnegie Mellon
Pittsburgh, Pennsylvania 15213

A PC-Style Markov Blanket Search for High Dimensional Datasets[1]

Joseph D. Ramsey
jdramsey@andrew.cmu.edu
Carnegie Mellon University
Department of Philosophy
Technical Report, 2/9/2006

For classification in high-dimensional datasets, it is often helpful to know not just the Markov blanket MB(t) of the target variable t but also the Markov blanket DAG MBD(t) of t. The Markov Blanket Fan Search (MBFS) is a local adaptation of the PC algorithm (Spirtes *et al.*, 2000) that searches directly for possible MBD(t) from conditional independence information over a set of causally sufficient acyclically related variables in a dataset D containing t. MBFS is scalable, just-in-time, and allows adjacency search to be performed in flexible order. The algorithm requires one parameter, maximum depth of search, though conditional independence tests used by the algorithm may require other parameters, such as significance level. Simulation results for datasets of up to 10,000 variables are given.

1. Introduction.

For classifying a variable t in a high dimensional dataset D over variables V, it is helpful first to be able to reduce the number of variables being used in the classification to a small subset V' of V \ {t} that contains all of the information needed to accomplish the classification with reasonable accuracy. The optimal way to select such a subset V' is to calculate the Markov blanket of t, MB(t)—that is, the uniquely smallest set of variables in V \ {t} such that I(s, t | S) for each s in V \ (S ∪ {t}), where I is a conditional independence oracle over the variables in V. If V is causally sufficient and faithful, and if the true causal graph over V is G, then MB(t) is equal to Parents(t) ∪ Children(t) ∪ Parents(Children(t)) in G. One could estimate MB(t) by first estimating the causal graph G' over V and then extracting MB(t) by finding the parents, children, and parents of children of t in G', though for large datasets this is unreasonably slow. The only sensible strategy is to estimate MB(t) locally. Within this strategy, one has a choice to either estimate MB(t) alone, without providing an estimate of the causal structure over MB(t) ∪ {t}, or to further provide an estimate of this causal structure itself. In the growing literature on Markov blanket search, there are relatively few algorithms that attempt to do the latter; in this report, a well-motivated algorithm of this latter sort is proposed, Markov Blanket Fan Search (MBFS), and its performance, scalability, and accuracy explored.

MBFS follows the theoretical direction of the PC algorithm, assuming causal sufficiency of variables in V and acyclicity of G. In the Markov blanket search literature, it is often commented the PC algorithm would be an ideal estimator of Markov blankets, if it could tractably be applied to the problem. Since MBFS follows exactly the advise of the PC algorithm for the adjacency search (though in a novel arrangement) and uses the same method for orienting edges, it makes sense to list ways in which MBFS is like PC. MBFS is correct for the same reasons that PC is correct, suitably restricted. The worst case complexity can be calculated in the same way for MBFS as for PC, though due to its restricted scope it is one polynomial degree less complex. Like PC, MBFS is just-in-time, though in a modular manner

[1] Many thanks to everyone who has commented on, contributed to, or made use of this work, most especially Clark Glymour, Peter Spirtes, and Xue Bai. Some early versions of the MBFS algorithm can be found in Bai et al. (2004a), Bai et al. (2004b), and Bai et al. (2005).

that allows the trace of the algorithm to be decomposed easily. Also, the output of MBFS is, like PC, a pattern, though a restricted algorithm is needed to show how Markov blanket DAGs can be extracted from these patterns. Given these and other similarities, if PC is an good estimator of Markov blankets, one expects MBFS to be an good estimator of Markov blankets as well, though simulation results do bear this out fairly convincingly.

It is not necessary to know the actual causal structure over MB(t) ∪ {t} in order to do classification for t; knowing MB(t) suffices for most classification algorithms, such as neural networks, support vector machines, decision trees, and so on. However, three points can be made in favor of the approach of MBFS in this regard. First, MBFS in simulation is an accurate and scalable estimator of MB(t), even if one ignores the fact that it returns causal structure over MB(t) ∪ {t}. A variety of algorithms have been proposed over the last several years specifically designed to calculate MB(t), including GS (Margaritis and Thrun, 1999), IAMB, InterIAMBPC, and InterIAMBPC-Eliminate (Tsamardinos *et al.*, 2002), Fast-IAMB (Yaramakala, 2004) MMMB (Tsamardinos *et al.*, 2003), HITON (Aliferis *et al.*, 2003), C5C (Frey *et al.*, 2003), Koller-Sahami (Koller and Sahami, 1996), and MBBC (Madden, 2002).[2] A side-by-side comparison of the performance, scalability, and accuracy of at least several of these algorithms with MBFS is a natural extension of the current work. Second, some classifiers in fact do require that one know the causal structure over MB(t) ∪ {t} to do classification of t given MB(t); a prominent case in point is Bayesian classification. One could of course first estimate MB(t) and then use a causal search algorithm, such as PC or GES, to estimate the graph over MB(t) ∪ {t}; however, MBFS yields this causal information directly.[3] Third, if one wants to control the value of t or to intervene on t, in which case it is extraordinarily helpful to know the causal structure over MB(t) ∪ {t} and not just MB(t) itself.[4]

The discussion will proceed as follows. Section 2 will define the notions of Markov blanket DAG and Markov blanket pattern. Section 3 will review the theory from Spirtes *et al.* (2000) needed to understand why MBFS algorithm works and will define MBFS algorithm itself. Section 4 will give simulation results for accuracy of estimation of MBD(t) by MBFS (for continuous and discrete datasets). Finally, Section 5 will sketch briefly extensions of this project to be carried out in the immediate future.[5]

2. Markov Blanket DAGs and Markov Blanket Patterns.

The Markov blanket DAG of a target t, MBD(t) may be understood to be the causal graph over MB(t) ∪ {t}, excluding edges among parents of t and among parents of children of t. Since MBFS will take

[2] Of these, only MBBC explicitly makes an attempt to estimate the causal structure over MB(t) ∪ {t}.

[3] It would be an interesting question, using a Monte Carlo approach, to see which approach, treated thoughtfully, gives better results. The two main questions are these. (a) Which methods give the best estimate of MB(t), irrespective of the estimated graph over MB(t) ∪ {t}? And, (b) For identical estimates of MB(t), which methods give the most accurate adjacency and orientation estimates for the graph over MB(t) ∪ {t}? Both of these questions can be addressed straightforwardly in simulation.

[4] The causal environment search algorithm described in Section 5, CEFS, would suffice for knowing the parents of t for purposes of controlling t.

[5] An implementation of MBFS used for this paper is in the Tetrad IV suite, http://www.phil.cmu.edu/projects/tetrad. Tetrad IV and the source code for Tetrad IV are copyrighted by Glymour, Spirtes, Scheines, and Ramsey and released under the GNU General Public license.

advantage only of conditional independence information over V, it will (like the PC algorithm) output a pattern—in this case, a *Markov blanket pattern*—for t, MBP(t), which is a graph that specifies an equivalence class of Markov blanket DAG's that are consistent with surveyed conditional independence information over V.[6] MBP(t) is a pattern that contains no extra nodes or edges with respect to Markov blankets for a particular target. This point may be made explicit by giving an algorithm for trimming a pattern to a Markov blanket pattern, where adj(x, G) is the set of nodes adjacent to x in G, and adj(x) is the set of adjacent nodes, where the graph with respect to which adjacency is being calculated is clear:

```
Trim-To-MBP(t, $\Pi$)
  1. PCPC $\leftarrow$ t $\cup$ adj(t) $\cup$ Parents(Children(t))
  2. for each x in V \ PCPC
  3.     for each y in adj(x)
  4.         remove x — y from $\Pi$
  5.     remove x from $\Pi$
  6. for each pair of variables x, y in Parents(t)
  7.     if x — y is in $\Pi$
  8.         remove x — y from $\Pi$
  9. for each pair of variables x, y in Parents(Children(t))
  10.    if x — y is in $\Pi$
  11.        remove x — y from $\Pi$
```

This graph trimming step removes from the graph any node (and any edge attached to such a node) that is not in {t} ∪ Parents(t) ∪ Parents(Children(t)), and removed from that result any edges among parents or among parents of children of t; the end result of this trimming is the Markov blanket pattern. The Markov blanket DAGs contained in this pattern can then be extracted from this Markov blanket pattern (or, for that matter, from any pattern containing t) as follows:

```
List-Markov-Blanket-DAGs(t, $\Pi$)
  1. Make a copy $\Phi$ of $\Pi$.
  2. Meek-Orient($\Phi$)
  3. Trim-To-MBP($\Phi$, t)
  4. if $\Phi$ has no unoriented edges
  5.     return <$\Phi$>
  6. else
  7.     L $\leftarrow$ <>
  8.     Pick an edge x — y in $\Phi$.
  9.     Orient x $\rightarrow$ y in $\Phi$.
  10.    L $\leftarrow$ L + List-Markov-Blanket-DAGs($\Phi$, t)
  11.    Orient y $\rightarrow$ x in $\Phi$
  12.    L $\leftarrow$ L + List-Markov-Blanket-DAGs($\Phi$, t)
  13.    return L
```

Let Φ be a pattern over the nodes of G and let Φ' be a subgraph of Φ. The orientation step (step 2) uses

[6] For the definition of pattern, see Spirtes et al. (2000).

the complete PC orientation rule set due to Meek (1995), suitably restricted. That is, where A is a subset of the nodes of Φ' such that for each $v \in A$, $adj(v, \Phi) \subseteq A$[7],

```
Meek-Orient($\Phi$', t, A)
    1. while orientations can be made, for arbitrary a, b, c, and d in A:
    2.     If $a \rightarrow b$, $b — c$, $a \notin adj(c)$, and Is-Noncollider(a, b, c) then orient $b \rightarrow c$.[8]
    3.     If $a \rightarrow b$, $b \rightarrow c$, $a — c$, then orient $a \rightarrow c$.
    4.     If $a — b$, $a — c$, $a — d$, $c \rightarrow b$, $d \rightarrow b$, then orient $a \rightarrow b$.
    5.     If $a — b$, b in $\in adj(d)$ $a \in adj(c)$, $a — d$, $b \rightarrow c$, $c \rightarrow d$, then orient $a \rightarrow d$.[9]
```

Noncolliders for Meek-Orient step 2 are checked locally, as follows.[10]

```
Is-Noncollider(x, y, z, $\Phi$')
    if $x \in adj(z)$
        return false
    else if $\sim\exists\ S \subseteq adj(z) \cup adj(z)$ in $\Phi$' such that $y \notin S$ and $I(x, z \mid S)$
        return true
    else
        return false
```

A Markov blanket pattern is therefore a pattern Π containing the target t such that every variable and every adjacency in Π is in one of the graphs returned by List-Markov-Blanket-DAGs(Π, t) for pattern Π over V and every edge oriented in Π is so oriented in every graph of List-Markov-Blanket-DAGs(Π, t) in which $x \in adj(y)$.

In order to allow the MBFS algorithm to be applied to datasets that are not thoroughly well-behaved, bidirected edges are permitted in both Markov blanket DAG's and Markov blanket patterns. The PC algorithm as defined in Spirtes *et al.* (2000) forbids such edges, and relegates their theoretical discussion to the FCI algorithm, where it becomes obvious that they represent conflicts in the conditional independence information provided about the variables in V and possibly indicate the presence of latent variables. Nevertheless, in practice, bidirected edges in the context of MBFS are often the result of colliders being oriented too liberally, either because of false positive adjacencies in the final graph or because of false positive conditional independence judgments. In these and other situations, much of the output of MBFS can still be interpreted sensibly, so rather than not generating any output graph at all, bidirected edges are included in output graph, and it is left up to the user to

[7] This condition is needed for the local noncollider orientation.

[8] The local double-check of the noncollider for the orient-away-from-collider rule is heuristic; the lion's share of false orientations by Meek-Orient are due to incorrect judgments concerning noncolliders in this rule, and the local check is more conservative. The local noncollider check and the graphical noncollider check are both correct in the large sample limit.

[9] This last rule is only needed for the case where background knowledge has been specified; it is included here for generality.

[10] The local check is correct in the large sample limit and is a more conservative estimator in the short run of noncolliders than the graphical test. Alternatively, noncolliders may be checked as unshielded triples <x, y, z> that have not yet been oriented as colliders, as in Spirtes et al. (2000). This and the performance advantages of the local collider test, below, will be argued in a separate report for PC, FCI, and MBFS. For simulation comparisons of sepset orientation to this style of local orientation for colliders and noncolliders, see Ramsey (2006), forthcoming.

decide whether to keep these in the pattern, treat them as undirected, or remove them.

3. The Markov Blanket Fan Search (MBFS).

The Markov Blanket Fan Search (MBFS) uses the assumptions and methods of the PC algorithm to search concentrically outward from a target to estimate its Markov blanket DAG. The assumptions are (1) that variables V being searched over are causally sufficient (i.e., that every common cause of distinct v, w $\in$ V is in V), (2) that the distribution D over V is faithful to the true causal graph G over V, and (3) that there are no cycles in G.

The algorithm for MBFS is divided into three stages: an adjacency search phase, an edge orientation phase, and a graph trimming phase. The adjacency search phase makes use of the following lemma:[11]

Lemma 1: Let G be the a causal graph over variables V (a DAG), let G' be an undirected graph over V such that if x and y are adjacent in G, x and y are adjacent in G'. Then z and w are nonadjacent in G' iff z is d-separated from w given S, for some subset S of either adj(z) \ {w} or of adj(w) \ {z} in G'.

Proof. Assume without loss of generality that z is not a descendant of w in G. Then by Lemma 3.3.9 of Spirtes *et al.* (2000),[12] z is d-separated from w given parents(w), a subset of adj(w) \ {z}. ∴

This is the principle used for adjacency search in PC; in the adjacency search of MBFS, the principle is applied concentrically. First, the target t is added to the graph G, and then a "fan" is constructed about t by constructing an edge from t to each node x associated with (i.e., not unconditionally independent of) t. By conditioning on sets of adjacents of increasing size ("depth"), the algorithm attempts to remove as many of the edges adjacent to t as possible, up to a fixed maximum depth, d_{max}. The list A of visited target nodes is initialized by adding t to it. At the end of this step, it is established that all edges t — x still in the graph cannot have been removed by conditioning on subsets of adj(t) containing no more than d_{max} variables.

The same procedure is now applied to each variable in adj(t). That is, for each variable x $\in$ adj(t), a fan is constructed about x, removing edges whenever a subset S of at most d_{max} edges is encountered such that I(x, y | S), for some y in adj(x) and S $\subseteq$ adj(x). As each node is visited by the fan construction procedure, it is added the list A. At the end of this step, it is established that all edges t — x cannot be removed, either by conditioning on subsets of adj(t) or by conditioning on subsets of adj(x); they are therefore, by Lemma 1, edges in the true undirected graph.[13]

[11] A different formulation, less amenable to Markov blanket search, is used in the original paper on the PC algorithm, Spirtes and Glymour (1991).

[12] Lemma 5.1.1 could also be used.

[13] Notice that this is different from the procedure for PC, in that PC applies the principle in Lemma 1 at each depth to every node in the graph, for each adjacent edge of that node, so that all depth 0 test are done before all depth 1 tests, and so on. The adjacency search for MBFS does depth 0, 1, and so on, in sequence for each node before moving on to the next node, doing the bookkeeping needed to avoid having to revisit nodes already visited. Obviously a causal search over a set of variables V could be done using the MBFS approach and would yield the same results as PC in the large sample limit; the comparison of these two methods has not yet been carried out.

There are at this point most likely several edges of the form $x' — y'$ where $x' \subseteq \text{adj}(t)$ and $y' \subseteq \text{adj}(x')$.[14] It is not known at this point whether these y' might be parents of children of t. To determine whether they might be, the fan-construction procedure is applied to each node in $\text{adj}(\text{adj}(t)) \setminus (\text{adj}(t) \cup \{t\})$. Care is taken not to construct edges to any node in A, since such an edge will already have been removed at an earlier stage in the algorithm. Once this third fan-construction step is finished, any variable y' still in $\text{adj}(\text{adj}(t)) \setminus (\text{adj}(t) \cup \{t\})$ must be on a length-two undirected path $t — x' — y'$ in the true graph and might be a parent of a child.

In order to obtain the final Markov blanket pattern, edges among nodes visited by the Construct-Fan method are oriented by first orienting colliders and then applying Meek-Orient. The collider orientation is as follows, where Π is a complete pattern over the variables of G, Π' is a subgraph of Π, and A is a subset of the nodes of Π' such that for each $v \in A$, $\text{adj}(v, \Pi) \subseteq A$,[15]

```
Orient-Colliders($\Pi'$, A)
    for each triple x *-* y *-* z where x, y, z in A
        if Is-Collider(x, y, z, $\Pi'$)
            Orient x*-*y*-*z as x*->y<-*z
```

```
Is-Collider(x, y, z, $\Pi'$)
    if adj(x, z) in $\Pi'$
        return false
    else if $\sim\exists\ S \subseteq \text{adj}(x) \cup \text{adj}(z)$ in $\Pi'$ such that $y \in S$ and $I(x, z \mid S)$[16]
        return true
    else
        return false
```

In order to orient x*-*y*-*z at this point in the algorithm as a collider using the local method, it is necessary that adj(x) contain a superset of the true adjacents to x and that adj(z) contain a superset of the true adjacents to z, for Lemma 1 to apply. Restricting Orient-Colliders and Meek-Orient to variables in A with respect to Π' accomplishes this.[17]

Note that Orient-Colliders (like Orient-Noncolliders) diverges from the use of sepsets in the PC algorithm to orient colliders. Sepset orientation may be used instead if desired. To orient colliders using sepsets, each time an edge $x — y$ is removed because $I(x, y \mid S)$, the set $\{x, y\}$ is mapped to S in a sepset mapping, Sepset, and in the collider orientation step $x — y — z$ is oriented as a collider just in case y is not in Sepset($\{x, z\}$). In simulation testing (reported separately), the use of local collider orientation as above leads to fewer false positive collider orientations for small sample sizes. Both methods are correct in the large sample limit.

[14] It's possible that all adjacents to t have been removed.

[15] This condition is needed for the local noncollider orientation. Note that over the set A, orientation for MBFS is identical to orientation for PC.

[16] Strictly speaking, it is necessary only to examine subsets of adj(X) and of adj(Z), since by assumption cycles do not occur. However, examining all subsets of $\text{adj}(X) \cup \text{adj}(Z)$ allows for slightly better accuracy when tested for PC in general and is not expensive.

[17] For sepset collider orientation, this is not an issue; sepsets are only recorded for edges that are removed from the graph.

Finally, the graph is trimmed to a Markov blanket pattern using Trim-To-MBP.

Using these helper methods, the algorithm may be easily stated. It takes the following arguments: V (a list of variables), I(x, y | Z) (an independence relation over V), $t \in V$ (the target variable), and d_{max} (the maximum size of any conditioning set considered for any conditional independence test performed). Variables internal to the algorithm are: G (a graph, to be constructed), A (a list of variables visited by the Construct-Fan method), t, v, w, and y (variables in V), and T (a set of variables in V). The algorithm is as follows:

```
MBFS(V, I, t, $d_{max}$):
    $G \leftarrow \emptyset$, $A \leftarrow \emptyset$
    Add t to G
    do Construct-Fan(t, V, I, G, A, $d_{max}$)
    for each v in adj(t)
        do Construct-Fan(v, V, I, G, A, $d_{max}$)
    for each w in adj(adj(t)) \ A
        do Construct-Fan(w, V, I, G, A, $d_{max}$)
    do Orient-Colliders(G, A)
    do Meek-Orient(G, t, A)
    do Trim-To-MBP(G, t)
    return G
```

```
Construct-Fan(y, V, I, G, A, $d_{max}$)
    $A \leftarrow A \cup \{y\}$
    for each v in V \ A
        if ~I(y, w | $\emptyset$)
            Add y — v to G
    $d \leftarrow 1$;
    while ($d \leq d_{max}$ and $d \geq |adj(y, G)|$)
        loop: for each v in adj(y, G) \ A
            for each $T \subseteq adj(y, G) \setminus \{v\}$ such that $|T| = d$
                if I(v, x | T)
                    remove y — v from G
                    continue loop
        $d \leftarrow d + 1$
```

The graph G returned by MBFS is, as mentioned above, an MB pattern that contains: (a) the target t, the parents and children of t, and the parents of the children of t; (b) all edges among t, parents of t, children of t, and parents of children of t (Some of these edges may not be directed); (c) possibly some extra nodes and edges to account for the possibility that, if some edges t — v were actually oriented as t → v, these nodes and adjacencies would be required in the MBD(T); and, (d) no nodes or adjacencies or undirected edges that do no belong in some Markov blanket DAG consistent with independence facts supplied by I. There may also be some bidirected edges in G if the independence oracle I is inconsistent with the assumptions that the true graph is acyclic and causally sufficient, for reasons explained at the end of Section 2.

The set of possible Markov blanket DAG's consistent with the final graph (trimming such extra nodes and edges out where appropriate) may be generated using method List-Markov-Blanket-DAGs, above, with the added direction that if bidirected edges are discovered in the output, these may at the discretion of the user be left in the output and ignored or oriented as undirected before applying the procedure.

With perfect independence information (i.e., in the large sample limit), MBFS is correct in that: (i) after step 7, all edges among nodes in A are edges in the true undirected graph G' over V, (ii) all colliders oriented in step 8 are correctly oriented, and (iii) any implied orientations made in step 9 are correctly oriented. Were the search expanded outward, it might be possible to orient additional colliders that in step 9 might lead to additional orientations over G' restricted to A, but these orientations are not available within the restriction of fan construction visits to A.

If the degree of the true graph is bounded above by k, then the number of conditional independence tests of the adjacency search phase with unlimited depth is bounded above by $(1+k+k^2)\sum_{i=0}^{k}\binom{n-1}{i}$, where n is the number of nodes in the graph. For a given depth d, however, the number of conditional independence tests of the adjacency search is bounded above by $(1+k+k^2)\sum_{i=0}^{d}\binom{n-1}{i}$; for fixed maximum degree this is equal to $c\sum_{i=0}^{d}\binom{n-1}{i}$ for some c, implying (since the algorithm is dominated by the adjacency search) that for d = 2 the algorithm is worst-case quadratic in n, a suggestion that can be cross-checked by plotting the timing data in Tables 1 and 2.[18]

4. Accuracy of Markov Blanket Estimation.

Simulation results are given for accuracy of estimation of MBP(t) by MBFS for continuous and binary datasets of 500, 1000, 2000, and 10,000 variables, respectively, each with 1000 cases.

The procedure is as follows. For each type of dataset and for each dimension n, a graph with n nodes and n edges was selected from a uniform distribution over DAG's of n nodes,[19] a random model was constructed (SEM for continuous, binary Bayes for discrete) with randomly selected parameters, and a dataset was simulated from this model with 1000 cases. 25 random variables were then chosen from each dataset, and the Markov blankets for these variables were estimated using MBFS. For continuous datasets, the conditional independence test was Fisher Z with specified alpha; for discrete datasets, the conditional independence test was chi square,[20] with specified alpha. All searches were performed at a depth of 2, since the true graphs were known to be sparse.

[18] Tighter average case bounds can be calculated by taking into account the fact that, e.g., by the time one gets to depth 3 in the Construct-Fan method, all of variables that have been discovered to be independent of the target conditional on sets of 0, 1, or 2 other variables have been removed from the list of adjacencies. In general, when MBFS is run at depth 3 on data generated from a sparse model, it returns fairly quickly even at high dimensions, so the worst case is rarely an issue.

[19] The Markov chain method used for generating random DAGs was an adaptation of Melancon et al. (2000). The number of iterations of the chain was limited to 3 x 10^9 due to time constraints for the 10,000-node examples.

[20] The multivariate chi square test used for these simulations calculates degrees of freedom as in Feinberg (1994), p. 142.

Table 1*. Average performance measurements of MBFS estimating MBD(t) (depth 2) for randomly simulated continuous data, from generative SEM models with number of edges equal to the dimension and randomly selected parameter values, each average over 25 randomly selected target variables t in the data set. Each row represents a new simulated dataset from which 25 targets were chosen. For variable meanings, see text.*

Dim	α	SIZE	FP	FN	PFP	PFN	CFP	CFN	PCFP	PCFN	TIME
500	0.0001	2.1	0.0	0.7	0.0	0.1	0.0	0.1	0.0	0.5	1.0
500	0.001	3.0	0.1	0.4	0.0	0.0	0.0	0.1	0.0	0.3	1.3
500	0.01	3.0	0.1	0.4	0.0	0.0	0.0	0.1	0.0	0.3	1.3
1000	0.0001	2.8	0.0	0.8	0.0	0.2	0.0	0.2	0.0	0.5	2.6
1000	0.001	3.5	0.2	0.8	0.0	0.2	0.0	0.1	0.1	0.5	3.1
1000	0.01	2.4	0.4	1.0	0.1	0.2	0.0	0.3	0.0	0.6	3.9
2000	0.0001	3.2	0.0	0.8	0.0	0.4	0.0	0.1	0.0	0.4	7.4
2000	0.001	3.0	0.2	1.2	0.0	0.1	0.0	0.4	0.0	0.9	5.5
2000	0.01	2.4	1.8	0.8	0.2	0.1	0.2	0.3	0.4	0.6	20.9
5000	0.0001	3.4	0.2	1.0	0.2	0.3	0.0	0.1	0.0	0.6	28.5
5000	0.001	3.5	0.2	1.3	0.0	0.3	0.0	0.3	0.0	0.8	34.1
5000	0.01	2.9	3.3	0.5	0.5	0.1	0.4	0.1	0.5	0.4	153.3
10000	0.0001	3.2	0.0	1.1	0.0	0.2	0.0	0.2	0.0	0.8	83.4
10000	0.001	2.6	0.3	0.7	0.1	0.1	0.0	0.1	0.0	0.6	118.0
10000	0.01	2.4	2.4	0.5	0.3	0.2	0.3	0.1	0.9	0.3	726.5

Table 2*. Average performance measurements of MBFS estimating MBD(t) (depth 2) for randomly simulated discrete data, from generative binary Bayes models with number of edges equal to the dimension and randomly selected parameter values, each average over 25 randomly selected target variables t in the data set. Each row represents a new simulated dataset from which 25 targets were chosen. For variable meanings, see text. Unlike the continuous case, the problem of simulating a random 10,000-variable dataset on a desktop computer was not solved in time for this report.*

Dim	α	SIZE	FP	FN	PFP	PFN	CFP	CFN	PCFP	PCFN	TIME
500	0.0001	3.8	0.0	2.4	0.0	0.6	0.0	0.6	0.0	1.5	0.4
500	0.001	3.0	0.0	1.6	0.0	0.4	0.0	0.4	0.0	1.0	0.5
500	0.01	3.1	0.2	1.5	0.0	0.2	0.0	0.4	0.0	1.0	0.6
1000	0.0001	3.5	0.0	2.1	0.0	0.6	0.0	0.6	0.0	1.2	0.9
1000	0.001	3.5	0.0	2.3	0.0	0.8	0.0	0.6	0.0	1.3	0.9
1000	0.01	2.4	0.4	1.1	0.0	0.2	0.0	0.3	0.0	0.8	1.7
2000	0.0001	3.0	0.0	2.2	0.0	0.6	0.0	0.5	0.0	1.3	2.0
2000	0.001	3.2	0.1	1.6	0.0	0.3	0.0	0.4	0.0	1.0	3.8
2000	0.01	2.3	0.5	1.6	0.0	0.3	0.0	0.5	0.0	1.1	4.5
5000	0.0001	2.4	0.1	1.3	0.0	0.2	0.0	0.4	0.0	1.0	10.7
5000	0.001	3.8	0.0	2.0	0.0	0.4	0.0	0.6	0.0	1.4	17.2
5000	0.01	2.7	0.6	1.4	0.0	0.5	0.0	0.3	0.0	0.7	36.2

Independently, the true Markov blanket for t was calculated graphically from the generative graph for each trial, and false positive and false negative statistics for nodes in MB(t), Parents(t), Children(t), and Parents(Children(t)) were calculated, along with running times of MBFS for each trial.

Averaged variable values for all trials are shown in Tables 1 and 2. The variables are as follows:

a) SIZE – Number of nodes in the true MB(t) (excludes target).

b) FP – False positive nodes with respect to the true MB(t), target excluded.

c) FN – False negative nodes with respect to the true MB(t), target excluded.

d) PFP – False positive parents with respect to the true MB(t).

e) PFN – False negative parents with respect to the true MB(t).

f) CFP – False positive children with respect to the true MB(t).

g) CFN – False negative children with respect to the true MB(t).

h) PCFP – False positive parents of children with respect to the true MB(t).

i) PCFN – False negative parents of children with respect to the true MB(t).

j) Time – Running time of MBFS for each trial, in seconds.[21]

The calculation of false positive and false negative counts is adjusted for the fact that MBFS returns a Markov blanket pattern and not a Markov blanket DAG as follows. With t the target variable, let M be the Markov blanket estimated by MBFS, and let M' be the true Markov blanket DAG extracted from the generative graph. Unoriented and bidirected edges in M that exist in M' are oriented as in M', and then the method Trim-To-MBP is called with M as argument, producing M*. False positive nodes, parents, children, and parents of children are then computed directly with respect to M*.

The data show, first of all, that MBFS is a tractable algorithm even up to a dimension 10,000 models for continuous data at alpha levels of 0.001 and below. False positives are well-controlled across the board for continuous and discrete models, except for dimension 10,000 models at alpha = 0.01. False negatives are well controlled for continuous models and less well controlled for discrete models, though the algorithm still provides useful information for such models. An interesting question for the discrete case is whether the estimated Markov blankets returned by MBFS are adequate for accurate classification, an issue that will be taken up separately.

5. Further Work.

This report is preliminary; several continuations are suggested for the near term. First, a simulation comparison of MBFS to alternative algorithms listed in Section 1 should be carried out to see how the various available algorithms scale up and how accurate they are at estimating true Markov blankets in simulation. Second, MBFS in the form described above will be applied to large, real datasets to see how well it performs in comparison to existing algorithms. Third, the lacuna in the simulation study above will be filled in, so that a simulation study for dimension-10,000 binary datasets can be carried out. Finally, a number of variant algorithms to MBFS that use the fan construction method in other ways will be implemented and tested.

[21] All trials were generated using the implementation of MBFS included in the version of Tetrad IV on 1/22/2006 (version 4573 in SVN) on a Dell GX280 with 2 GB RAM running Fedora Linux, using Java JRE 1.5.0.

References

Aliferis, C., I. Tsamardinos, and A. Statnikov (2003). HITON: a novel Markov blanket algorithm for optimal variable selection. TR, Vanderbilt University, DSL-03-08.

Bai, X., C. Glymour, R. Padman, P. Spirtes, J. Ramsey. (2004a). MB Fan Search Classifier for Large Data Sets with Few Cases. Technical Report CMU-CALD-04-102. School of Computer Science, Carnegie Mellon University (2004)

Bai, X., R. Padman, and E. Airoldi (2004b). Sentiment Extraction from Unstructured Text using Tabu Search-Enhanced Markov Blanket. Carnegie Mellon University, School of Computer Science, Technical Report CMU-ISRI-04-127.

Bai, X., R. Padman (2005). Tabu Search Enhanced Markov Blanket Classifier for High Dimensional Data Sets. Proceedings of INFORMS Computing Society. (to appear) Kluwer Academic Publisher.

Feinberg, S. (1994). The analysis of cross-classified data, 2nd ed. Cambridge, MA: MIT Press.

Frey, L., D. Fisher, I. Tsamardinos, C. Aliferis, and A. Statnikov (2003). Identifying Markov blankets with decision tree induction. Proceedings of the Third IEEE International Conference on Data Mining (ICDM'03) 0-7695-1978-4/03.

Madden, M. (2002). Evaluation of the performance of the Markov Blanket Bayesian Classifier algorithm. Technical report, Department of Information Technology, National University of Ireland, 2002.

Meek, C. (1995). Causal inference and causal explanation with background knowledge. In *Proceedings of Eleventh Conference on Uncertainty in Artificial Intelligence*. Montreal: Morgan Kaufman, 403-418.

Koller, D. and M. Sahami (1996). Toward optimal feature selection. In *International Conference on Machine Learning*, 284-292.

Margaritis, D. and S. Thrun (1999). Bayesian network induction via local neighborhoods. In S. Solla, T. Leen, and K. Muller, ed., *Proceedings of Conference on Neural Information Processing* Systems *(NIPS-12)*. MIT Press.

Melancon, G., I. Dutour, and M. Bousquet-Melou (2000). Random generation of DAGs for graph drawing. Dutch Research Center for Mathematical and Computing Science (CWI). Technical Report INS-R0005, Febuary.

Ramsey, J. (2006). Local Collider Orientation for PC and FCI. Carnegie Mellon University, Department of Philosophy. Technical Report. Forthcoming.

Spirtes, P., C.Glymour, and R. Scheines (2000). Causation, Prediction, and Search. Cambridge, MA: MIT Press.

Tsamardinos, I., and C. Aliferis (2003). Towards principled feature selection: Relevancy, filters and wrappers. In *Ninth International Workshop on Artificial Intelligence and Statistics*, Key West, Florida.

Tsamardinos, I., C. Aliferis, and A. Statnikov (2002). Algorithms for large scale Markov blanket discovery. In T*he 16th International FLAIR*S *Conference*, St. Augustine, FL.

Tsamardinos, I., C. Aliferis, and A. Statnikov (2003). Time and sample efficient discovery of Markov blankets and direct causal relations. KDD.

Yaramakala, Sandeep (2004). Fast Markov blanket discovery. M.A. Thesis, Computer Science, Iowa State University.