Path Consistency with Minimum Edges

Chia-Jung Chang
Department of Computer Science and Information Engineering
National Taiwan University, Taipei, Taiwan

Abstract

The PC (Path Consistency) algorithm can be used to infer an undirected graph where the nodes represent the measured variables and the edges indicate the causality between these variables. However, its result depends on the order of the pairs of variables examined. To avoid this problem, we model this problem as an optimization problem and call it the PCME (Path Consistency with Minimum Edges) problem. The input of the PCME problem contains an initial undirected graph and a set of removal rules. Each rule can explain the relationship, represented as an edge, between a pair of nodes by some other specific path(s) connecting the two nodes. One edge can be removed if the relationship it represents is explained by a removal rule and the existence of the specific path(s). The problem is to find the minimum number of edges explaining all the edges in the initial undirected graph. We prove the NP-hardness of the problem and give a polynomial-time greedy algorithm with the best possible approximation ratio, unless NP has slightly super-polynomial time algorithms.

1 Introduction

A causal graph is a directed or an undirected graph where nodes represent measured variables and edges indicate the causality between pairs of variables. The PC (Path Consistency) algorithm [6] is for inferring a causal graph according to the conditional independence relations among measured variables. It is based on the idea that the high depencency between two nodes might be due to the direct cause-and-effect relationship, or the indirect relationship (a affects b and b affects c, but a and c also have high correlation). The PC algorithm is to remove the indirect relationships and to build a graph with only direct relationships. It starts from an undirected complete graph. Only the edges with high correlation (or mutual information) endpoints are maintained. One edge can be removed if there are alternative path(s) and the two endpoints are conditionally independent. In the last step, it assigns directions for some remaining edges according to the removed edges. It has been applied to infer gene regulatory networks using microarray expression data [5, 9].

In this article, we only consider the undirected part of the algorithm, which is the removal of the unwanted edges. Since the removal of one edge might depend on the existence of other edges, the inferred graph is affected by the order of the pairs of variables tested for the conditional independence. Steck [7] introduced the idea of ambiguous regions, which indicate maximal sets of inter-dependent edges. In the Hugin tool implementation [4], the ambiguities can be solved by user interaction, which requires prerequisite knowledge for the underlying structure. Abellán et al. [1] also proposed an algorithm that removes the edges following an order given by a Bayesian score. However, in previous works, the problem is not well modelled.

According to the Occam's razor theory, the graph with the minimum number of edges tends to be correct. Therefore, we model the original problem as an optimization problem. We say that the high depencency between two nodes can be **explained** by a graph if there is an edge between them or there are alternative path(s) with certain criteria (defined later). The problem becomes finding the graph with the minimum number of edges that can explain the depencency for all pairs of variables. We prove it to be an NP-hard problem and give a $\log |E|$ approximation algorithm, where $|E|$ is the number of edges in the zero order PC graph (explained later). We also prove the approximation ratio is the best possible unless **NP** has slightly super-polynomial time algorithms.

In the following, we introduce the PC algorithm and its problem of inter-dependency. Then, we give a formal definition of the PCME (Path Consistency with Minimum Edges) problem and its pre-processing steps. In Section 2, we shrink the solution space of the PCME problem and prove its NP-hardness by reducing the *Set Cover* problem [3] to it. In Section 3, we transform the PCME problem to another problem similar to the *Set Cover* problem by labelling the edges according to their roles in the removal rules and give a greedy algorithm to solve it. In Section 4, we prove the approximation ratio of the greedy algorithm. The discussion and the conclusion are in Section 5 and Section 6, respectively.

The PC Algorithm

The PC algorithm requires a method to calculate the conditional independence of two variables on a small set (usually one or two) of other variables. For example, the partial correlation coefficient and the conditional mutual information can be used to measure the conditional independence [5, 9]. A threshold is then set to tell if two nodes are conditional independent or not. The choice of the measurement of conditional independence or the threshold is independent of our problem. We denote the CI (conditional independence) of two variables i and j on a set S as $\mathrm{CI}(i, j|S) = \{\mathrm{T}, \mathrm{F}\}$.

There are a little bit difference for the steps of the PC algorithm between different applications and we adopted the steps in [9]. It starts from a complete undirected graph $G_c = (V, E)$ with nodes V representing the measured variables and edges E representing the causalities. Then,

1. Set $n = 0$
2. For each edge $e(i, j) \in E$, let $Adj(i, j) = \{k|e(i, k) \in E \text{ and } e(j, k) \in E\}$. If there exists a set $S \subseteq Adj(i, j)$ satisfying $|S| = n$ and $\mathrm{CI}(i, j|S) = \mathrm{T}$, remove $e(i, j)$ from E.
3. Set $n = n + 1$
4. Repeat step 2 and 3 until there is no edge $e(i, j) \in E$ with $|Adj(i, j)| \geq n$

A zero-order PC graph is the graph obtained after the iteration of $n = 0$. It is a graph where an edge indicates the node-node dependency (correlation or mutual information) is higher than some threshold. Usually the iteration stops after $n = 2$ and the algorithm results in a second-order PC graph; therefore, in many applications, the iteration time is assigned in the first place [5, 9].

Sometimes the removal of one edge may depend on the existence of another removable edge. For example, assume Figure 1(a) is a zero-order PC graph and $\mathrm{CI}(1, 2|3) = T$, $\mathrm{CI}(1, 4|3) = T$ and $\mathrm{CI}(1, 3|2) = T$. The removal of $e(1, 2)$ depends on the existence of $e(1, 3)$ and $e(2, 3)$, while the removal of $e(1, 3)$ depends on the existence of $e(1, 2)$ and $e(2, 3)$. We say that the edges $e(1, 2)$ and $e(1, 3)$ are inter-dependent on each other.

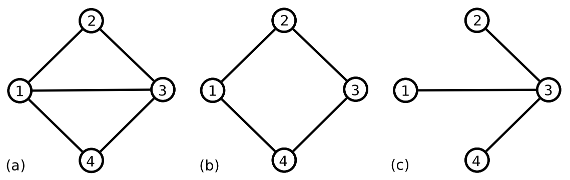

Figure 1: An example of inter-dependency

The order of testing CI for the pairs of nodes is thus important. If we test $e(1, 3)$ first, we will remove $e(1, 3)$ since $2 \in Adj(1, 3)$ and $CI(1, 3|2) = T$. Both $e(1, 2)$ and $e(1, 4)$ are irremovable after $e(1, 3)$ is removed because $Adj(1, 2)$ and $Adj(1, 4)$ are then empty. The algorithm will end up with Figure 1(b). On the other hand, if we test $e(1, 2)$ first and remove it, we will find $e(1, 3)$ is irremovable, and $e(1, 4)$ is removable. Now the algorithm results in Figure 1(c). We regard it as a better solution because it uses fewer edges to explain all the edges of the zero-order PC graph. We model the problem as an optimization problem.

The Path Consistency with Minimum Edges Problem

The idea is to list all possible rules for removing edges and to choose the minimum number of edges to remove all the other edges using these rules. We name the pair of the initial undirected graph (the zero-order PC graph) and the set of rules the **PC region**.

Definition 1. *The PC Region*
A PC region has the form (G, R) where $G = (V, E)$ is an undirected graph and R is a set of what we call as "*removal rules*". A removal rule has the form $((i, j), S)$ where

1. $i, j \in V$ and $e(i, j) \in E$,
2. $S \subseteq Adj(i, j) = \{k|e(i, k) \in E \text{ and } e(j, k) \in E\}$ and $|S| \geq 1$,
3. $\mathrm{CI}(i, j|S) = T$

A removal rule $((i, j), S)$ means that when all the edges from i to S and from j to S exist, $e(i, j)$ is removable.

We define the order of a removal rule as its $|S|$. For a first order removal rule $((i,j),k)$, the two edges $e(i,k)$ and $e(j,k)$ should be maintained to remove $e(i,j)$. We also define the order of a PC region as the maximum order among its removal rules. The order is a constant (i.e. 1 or 2) in most applications. Figure 2 gives an illustration of a first order PC region.

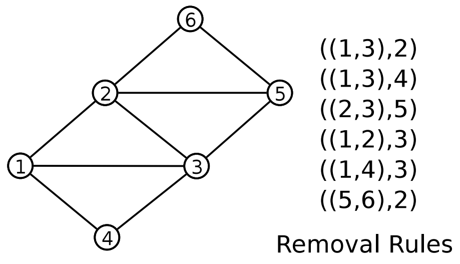

Figure 2: An example of a first order PC region

For a rule $((i,j),S)$, we name an edge with one endpoint in S and the other endpoint in $\{i,j\}$ as the "**explanatory edges**" and name the edge $e(i,j)$ as the "**dependent edge**", because the observed dependency between nodes i and j can be explained by the rule and the existence of the explanatory edges and $e(i,j)$ is thus removable. Simply put, the PCME problem is to find a minimum set of edges which can explain the relationships for all edges in the original graph, either by the selected edges themselves or by their explanatory edges.

Problem 1. The *PCME* Problem
Given a PC region (G,R), find the minimum set of edges $E_{min} \subseteq E$ such that for all $e \in E$, either $e \in E_{min}$ or there is a rule $r \in R$ such that e is its dependent edge and all its explanatory edges are in E_{min}.

Detecting the PC Region

It takes polynomial time to detect the PC region with a constant order. Given the measurement of multiple variables, we can take the following steps to detect the PC region (G,R):

1. Find the zero-order PC graph G_0 with the PC algorithm and let $G = G_0$.
2. Set $n = 1$ and the removing rules $R = \{\}$.
3. For each $e(i,j) \in E$, let $Adj(i,j) = \{k|e(i,k) \in E \text{ and } e(j,k) \in E\}$.
4. For each $e(i,j) \in E$, find all sets $S \subseteq Adj(i,j)$ satisfying $|S| = n$ and $\mathrm{CI}(i,j|S) = \mathrm{T}$. Join the rule $((i,j),S)$ to R for each S. Then set $Adj(i,j) = Adj(i,j) - \bigcup S$.
5. Set $n = n + 1$.
6. Repeat the steps 4 and 5 until there is no $e(i,j)$ with $|Adj(i,j)| \geq n$ or n reaches the threshold.

The threshold of n equals the order of the PC region. The PC region produced by this process is the input of the original PCME problem.

2 NP-hardness of the PCME Problem

Before proving its NP-hardness, we first shrink the solution space by labelling the edges in E with four categories. The categories are important for both the proof of its NP-hardness and the greedy algorithm. We define the categories according the roles of the edges in the removal rules.

Shrinking the Solution Space by Labelling the Edges

The first category E_1 contains the edges that don't appear as dependent edges in any rules. They must be maintained because there is no rule to remove them. The edges $e(2,5)$, $e(2,6)$, $e(3,4)$ and $e(3,5)$ in Figure 2 are in E_1. **The second category** E_2 contains the edges that only appear as dependent edges and at least one of their removal rules' explanatory edges are all in E_1. Since their explanatory edges are maintained, the connection of the edges have been explained. They should be removed directly because the removal of no edge depends on them. The edge $e(5,6)$ in Figure 2 is in E_2. **The third category** E_3 contains the edges appear as both explanatory and dependent edges and at least one of their rules' explanatory edges are all in E_1. The relationships of the edges have been explained as in E_2, but since they are the explanatory edges of some other edges, the existence of them can explain and may remove their dependent edges. The edge $e(2,3)$ in Figure 2 is in E_3. The edges left are in **the fourth category** E_4. The edges $e(1,2)$, $e(1,3)$ and $e(1,4)$ in Figure 2 are in E_4. It is trivial that it takes linear time to label the edges.

In a PC region, the selection of the edges in E_1 and E_2 is certain. The optimization of the PCME problem only depends on the selection of the edges in E_3 and E_4, so the solution space can be limited to the union of E_3 and E_4.

Problem 2. The Stage 2 *PCME* Problem
Given the labelled edges of the original PCME problem, the new problem is to find the minimum number of edges $E_{min} \subseteq E_3 + E_4$ such that for all $e \in E_4$, either $e \in E_{min}$ or there is a rule $r \in R$ such that e is its dependent edge and all its explanatory edges are in $E_{min} + E_1$.

The solution of the original PCME problem is the union of the solution of the stage 2 PCME problem and E_1. From now on, the solution space and the solution follow the definition of the stage 2 PCME problem.

Proof of NP-hardness

In this subsection, we prove that the *Set Cover* problem [3] can be reduced to the $PCME$ problem. Since the *Set Cover* problem is NP-Complete, the $PCME$ problem is NP-hard.

Definition 2. *The Set Cover Problem*
Given a universe U of n element $\{1, ..., n\}$ and m covering sets of elements $CS = \{s_1, ..., s_m\}$ whose union is U, find a minimum subset of CS whose union is also U.

Figure 3 gives an example of the set cover problem, in which $U = \{1, ..., 5\}$ and $CS = \{a, b, c, d\}$. We can choose the sets b and c to cover the universe $\{1...5\}$.

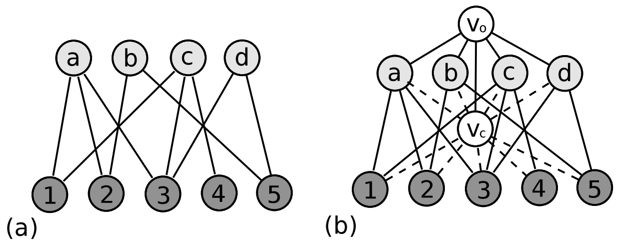

Figure 3: (a) An example of the set cover problem (b) The transformation of the set cover problem to the PCME problem (without showing the removal rules). Only the selection of the dash lines should be considered.

Given the universe of n elements $U = \{1...n\}$ and the collection of m sets $CS = \{s_1...s_m\}$, the idea is to construct a PC region (G, R) and to map the solution of its PCME problem to the solution of the set cover problem.

G starts with a centroid node v_c and R starts with an empty set. Then,

1. Add m nodes corresponding to the m sets $\{s_1...s_m\}$. (We call them the set nodes.)
2. Add edges from v_c to all the set nodes (the set edges).
3. Add a node v_o and add edges from v_o to v_c and to the set nodes.
4. Add rules $((v_c, s_i), v_o)$ for all the set nodes s_i.
5. Add n nodes corresponding to the n elements $\{1...n\}$ (the element nodes).
6. Add edges from node v_c to all the element nodes (the element edges).
7. For each covering set s_i and for each element $k \in s_i$, add the edge $e(s_i, k)$ and the rule $((v_c, k), s_i)$.

The edges added in step 3 and 7 belong to E_1 because they are not the dependent edges of any removal rules. They must be selected, so we only consider the set edges and the element edges, which belong to E_3 and E_4, respectively. Figure 3(b) illustrates the transformation of the set cover problem in Figure 3(a).

Lemma 1. Any solution of the set cover problem can be transformed to a solution of its corresponding PCME problem with the same size.

Proof. Given a solution of the set cover problem, we can select the set edges corresponding to the sets in the solution. All the other set edges and the element edges are removed. Any set edge $e(v_c, s_i)$ can be removed because it belongs to E_3. All the element edges can be removed because any element k is covered by some set s_i in the solution and the element edge $e(v_c, k)$ can thus be explained by the rule $((v_c, k), s_i)$ and the existence of $e(s_i, k)$, which belongs to E_1, and $e(v_c, s_i)$, which is selected as described.

□

Lemma 2. Any solution of the constructed PCME problem can be transformed to a solution of the original set cover problem with fewer or the same size.

Proof. We first prove that for any solution of this PCME problem, we can find another solution using only the set edges and having fewer or equal number of edges. When there is still element edges in the solution, pick one of them, say $e(v_c, k)$. Since the element k is in some set, say s_i, we can replace $e(v_c, k)$ with $e(v_c, s_i)$ as another solution.

The element edges $e(v_c, k)$ can be explained by the rule $((v_c, k), s_i)$ and the existence of $e(s_i, k)$ and $e(v_c, s_i)$.

Since there are no element edges in the solution, every element edge $e(v_c, k)$ should be explained by at least one rule $((v_c, k), s_i)$ and the existence of the set edge $e(v_c, s_i)$. This means $k \in s_i$ by definition and we choose s_i into the solution of the set cover problem. Since every k is covered, the collection of the sets corresponding to the set edges is a solution of the set cover problem. $\square$

It is trivial that all the reduction and the transformation steps run in polynomial time. The number of the selected E_3 edges for the solution of this PCME problem is exactly the same as the number of sets for the solution of the set cover problem. We thus prove that the *PCME* problem is NP-hard.

Theorem 3. The *PCME* problem is an NP-hard problem

3 A Greedy Algorithm

In Section 2, we mention that the solution space of the PCME problem is limited to the edges in E_3 and E_4. In Figure 4, we illustrate the stage 2 PCME problem of Figure 2.

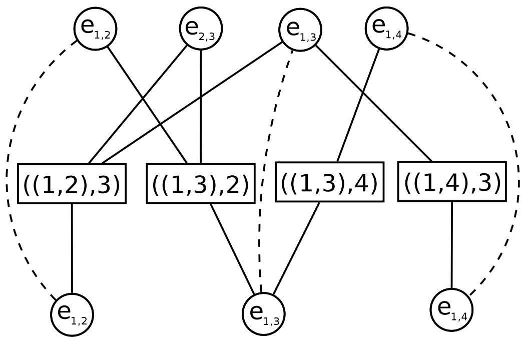

Figure 4: The transformation of the PCME problem in Figure 2

The nodes in the top layer represent the solution space, the edges in E_3 and E_4. The nodes in the down layer represent the edges in E_4, the relationships of which are still not explained. The rectangles in the middle layer represent the removal rules with dependent edges in E_4. Each removal rule rectangle has links to the top nodes representing its explanatory edges and a link to the down node representing its dependent edge. There are also links between the top nodes and the down nodes when they represent the same edges.

A down node can be covered by the selection of a top node with a link to it, or by the activation of a rectangle with a link to it. A rectangle is activated iff all top nodes with links to it are all selected. For example, to cover the down node $e_{1,2}$, one can select the top node $e_{1,2}$ or both the top nodes $e_{2,3}$ and $e_{1,3}$.

There are rectangles connecting to only one top nodes; because its other explanatory edges are in E_1 and have already been selected. For example, the rule $((1, 4), 3)$ has only one top node $e_{1,3}$ because its explanatory edge $e_{3,4}$ is in E_1. The edges in E_3 are in the top layer but not in the down layer because they have been explained by their explanatory edges. For example, the edge $e_{2,3}$ has been explained because its explanatory edges $e_{2,5}$ and $e_{3,5}$ of the rule $((2, 3), 5)$ are in E_1.

There are removal rules not in the rectangles. The explanatory edges of these rules are all in E_1 and the rules are thus activated; therefore, we don't have to put them in the illustration.

Problem 3. An Alternative Description of the Stage 2 *PCME* Problem
The PCME problem is to select the minimum number of the top nodes to cover all the down nodes, either by the selection of the top node representing the same edge, or by activating the rectangle that has a link to the down node. A rectangle is activated iff all the top nodes with links to it are selected.

We propose a greedy algorithm in Algorithm 1.

Algorithm 1 A Greedy Algorithm
```
1: Set scores for the top nodes as 1/(the number
   of extra down nodes it can cover)
2: while Not all down nodes are covered do
3:     Select one top node $v_{min}$ with the minimum score.
4:     Delete $v_{min}$, the explained down nodes and
       the rectangles connecting to them.
5:     Adjust score for the top nodes.
6:     Add $v_{min}$ to the solution
7: end while
```

Taking Figure 4 for example, in the beginning, the score of $e_{1,2}$ is 1 because the selection of it only cover its corresponding down node. It can cover $e_{1,3}$ only after the selection of $e_{2,3}$. The algorithm selects $e_{1,3}$ into its solution because its score is minimum:1/2. It can cover $e_{1,3}$ and $e_{1,4}$.

For each iteration of the greedy algorithm, the score of a top node may increase because some down nodes have been explained. For example, the score of $e_{1,4}$ rises from $1/2$ to ∞ after the selection of $e_{1,3}$. The score of a top node may also decrease because all the other explanatory edges are selected for some rectangles. For example, the score of $e_{2,3}$ drops from ∞ to 1 because it can explain $e_{1,2}$ with the rule $((1,2),3)$ after the selection of $e_{1,3}$. We give an illustration in Figure 5(a) after the first iteration .

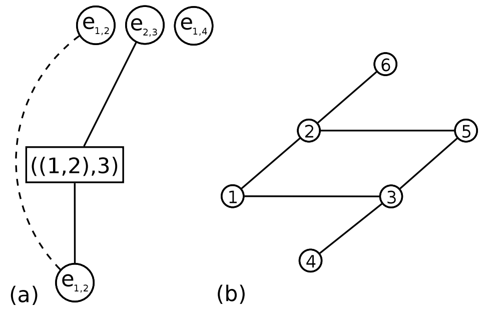

Figure 5: (a)The data structure after the selection of $e_{1,3}$ (b)The result of the greedy algorithm on Figure 2

The union of the edges in the solution of the greedy algorithm and the edges in E_1 is the final solution of the PCME problem. Figure 5 is the result of the labelling and the greedy algorithm for the PCME problem in Figure 2.

Complexity

Let $|E_{amb}| = |E_3 + E_4|$ and $|R_{amb}|$ be the number of removal rules having their dependent edges in E_4 (i.e. the rectangles). The top nodes can be stored in a heap map data structure, so each operation for extracting the minimum one or for adjusting the score of one node runs in $O(\log |E_{amb}|)$ time. In an iteration, let R be the rectangles to which the deleted top node v_{min} have links. The number of top nodes to be adjusted is $O(|R|)$ since we assume that the order of the PC region is constant. In addition, the total number of links between the top nodes and the rectangles is in $O(|R_{amb}|)$ due to the same reason. Since the links between v_{min} and R are used only for this iteration, the total number of operations is in $O(|R_{amb}|)$. Therefore, the time complexity of the greedy algorithm is in $O(|R_{amb}| \log |E_{amb}|)$.

4 Approximation

The proof of the approximation of the greedy algorithm for the PCME problem is similar to that for the set cover problem in [8], except the following lemma.

Lemma 4. In Algorithm 1, there are always top nodes that can cover at least one down node.

Proof. Each down node has a top node representing the same edge. Before all the down nodes are covered, at least the top nodes corresponding the uncovered down nodes can cover the down nodes. □

With this lemma, we can define the price of the down nodes for the greedy algorithm as: 1/(the number of the down nodes covered in the same iteration). The sum of the prices of all the down nodes are thus equal to the number of selected edges using the greedy algorithm, which we denote as N_{gre}. We denote the number of the selected edges for the optimal solution as N_{opt}. It is noted that the greedy algorithm always select the top node minimizing the price for each individual covered down node.

Theorem 5. The approximation ratio of the greedy algorithm for the simplified PCME problem is $\ln |E_4|$.

Proof. We order the down nodes as the order of the iterations they are covered. The ordered prices are denoted as $(C_1, ..., C_{|E_4|})$. In the iteration covering the k-th down node, there are more than $|E_4| - k + 1$ down nodes uncovered. We can always choose the N_{opt} top nodes of the optimal solution to cover all the uncovered down nodes. The price of the uncovered down nodes are thus at most $N_{opt}/(|E_4| - k + 1)$. Since C_k is minimum in the iteration, $C_k \le N_{opt}/(|E_4| - k + 1)$.
$N_{gre} = \sum_{k=1}^{|E_4|} C_k \le N_{opt} \sum_{k=1}^{|E_4|} \frac{1}{k} \le N_{opt}(\ln |E_4| + 1)$.

The proof is thus completed.

□

Theorem 6. $\ln |E_4|$ is the best-possible approximation ratio for the simplified PCME problem, unless **NP** has slightly super-polynomial time algorithms.

Proof. The proof is based on the inapproximability of the set cover problem. The set cover problem with n elements has $\ln n$ approximation ratio as its threshold, unless **NP** has slightly super-polynomial time algorithms (e.g. $n^{O(\log \log n)}$) [2].

The transformation in Section 3 can transform the set cover problem to a PCME problem with n edges labelled as E_4. If we can find an algorithm with approximation ratio better than $\ln |E_4|$, the solution can be directly applied to the original set cover problem. Since the sizes of the solutions of both problems are the same and $|E_4| = n$, we now have an algorithm for the set cover problem with approximation ratio better than $\ln n$, which contradicts the fact.

□

5 Discussion

The Ambiguous Regions

For the PC problem, the sets of connected interdependent edges are called ambiguous regions [7]. The removal of one edge only depends on the edges in the same ambiguous region. Therefore, we can divide the original PCME problem into un-intersected ambiguous regions.

Given a PC region (G, R), at first, we group the removal rules in R by intersection of their edges, including the explanatory and the dependent edges. For each group, the union of the edges of the removal rules is a connected undirected graph. The pair of the graph and the removal rules is a smaller PC region. We can solve the PCME problem for each PC region separately.

There are edges not in any of the ambiguous regions. The removal of no edge depends on them and there is no rule to remove them. The combination of the solutions for all the ambiguous regions and these irremovable edges is an undirected graph that is the solution of the original PCME problem.

By splitting the PC region into smaller ambiguous regions, we can improve the time complexity of the greedy algorithm, which is $O(|R_{amb}| \log(\max(|E_{amb}|)))$ now, where $\max(|E_{amb}|)$ is the maximum $|E_{amb}| = |E_3 + E_4|$ among all ambiguous regions. The approximation ration is also better and is now $\ln \max(|E_4|)$, where $\max(|E_4|)$ is the maximum $|E_4|$ among all ambiguous regions.

The Edge Weights

In some applications, the edges have weights on them, for example, the degree of independence. One might ask to construct a graph to explain all edges with the edges having the minimum sum of weights. The original PCME problem is a special case where the weights of all edges are the same.

We can also solve the weighted PCME problem by adapting the greedy algorithm. We change the score of a top node as (the edge weight)/(the original score) and always select the top node with minimum score into the solution. The proof of its approximation ratio is almost the same, except the price for each down node become (the weight of selected top nodes)/(the number of down nodes explained in the same iteration).

6 Conclusion

The result of the PC algorithm is affected by the order of testing the pairs of variables. In this article, we model the problem as an optimization problem called the PCME problem. The problem is to find a graph with the minimum number of edges that can explain the conditional dependency for all pairs of nodes. We prove that it is an NP-hard problem and give a polynomial time greedy algorithm and prove its approximation ratio, which is the best possible one, unless **NP** has slightly super-polynomial time algorithms.

Acknowledgements

Chia-Jung Chang was supported in part by NSC grants 100-2221-E-002-131-MY3 and 101-2221-E-002-063-MY3. I would like to thank Professor Kun-Mao Chao for his valuable advice.

References

[1] Joaquín Abellán, Manuel Gómez-Olmedo, and Serafn Moral. Some Variations on the PC Algorithm. In *Probabilistic Graphical Models*, pages 1–8, 2006.

[2] Uriel Feige. A Threshold of $\ln n$ for Approximating Set Cover. *Journal of the ACM*, 45(4):634–652, July 1998.

[3] Michael R. Garey and David S. Johnson. *Computers and Intractability.* Freeman, New York, 1979.

[4] Anders L. Madsen, Michael Lang, Uffe B. Kjrulff, and Frank Jensen. The Hugin Tool

for Learning Bayesian Networks. In *In Proceedings of 7th European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty*, pages 594–605, 2003.

[5] Shigeru Saito and Katsuhisa Horimoto. Co-expressed Gene Assessment Based on the Path Consistency Algorithm: Operon Detention in Escherichia Coli. In *International Conference on Systems, Man and Cybernetics*, pages 4280–4286. IEEE, October 2009.

[6] Peter Spirtes, Clark N. Glymour, and Richard Scheines. *Causation, Prediction, and Search.* MIT Press, second edition, 2000.

[7] H. Steck. *Constraint-Based Structural Learning in Bayesian Networks using Finite Data Sets.* Ph.D. Thesis, Der Technischen Universität München, Munich, Germany, 2001.

[8] Vijay V. Vazirani. *Approximation Algorithms.* Springer-Verlag, Berlin, 2001.

[9] Xiujun Zhang, Xing-Ming Zhao, Kun He, Le Lu, Yongwei Cao, Jingdong Liu, Jin-Kao Hao, Zhi-Ping Liu, and Luonan Chen. Inferring Gene Regulatory Networks from Gene Expression Data by Path Consistency Algorithm Based on Conditional Mutual Information. *Bioinformatics (Oxford, England)*, 28(1):98–104, January 2012.