International Journal of Database Theory and Application
Vol.9, No.5 (2016), pp.21-36
http://dx.doi.org/10.14257/ijdta.2016.9.5.03

Exact Dominance Querying Algorithm on CP-nets

Guanlin Xin[1] and Jinglei Liu[2]

School of Computer and Control Engineering
Yantai University, Yantai 264005, China
[1]Master student; [2]Corresponding author
guanlin_xin@sina.com; jinglei_liu@sina.com

Abstract

CP-nets (*conditional preference networks*) *is a graphical model for representing qualitative conditional preference statements via ceteris paribus semantics interpretation. Dominance querying, the problem of determining whether an outcome is preferred to another on this graphical model, is of fundamental importance in preference querying. However, to date, the exact dominance querying algorithm with respect to any binary-valued CP-nets has not been given, hence designing an exact dominance querying algorithm is quite necessary. Fortunately, we discover that dominance querying problem is essentially a single source shortest path problem, and the induced graph of CP-nets is a sparse graph, as a result, dominance querying problem can be solved by Johnson algorithm which is suitable for solving the single source shortest path problem on sparse graph. As a byproduct, this algorithm can determine least flip numbers if an outcome dominates another outcome. At last, we present results of experiments that demonstrate the feasibility of our approach to dominance querying on acyclic CP-nets.*

Keywords: *conditional preference networks, ceteris paribus, dominance querying, single source shortest path problem, least flip numbers*

1. Introduction

Preference is a central aspect of decision making, common sense tells us that preference can guide human's behavior selection in many contexts. For instance, in early child-hood, when facing many toys, whereas our parent only prepare to buy only one for us, the key problem is to decide whether a toy is preferable to the other one on the basis of some preference criterions [1]. In preference decision, preference is primarily divided into two areas: quantitative preference and qualitative preference. In quantitative preference, the preference is expressed by a numerical value [2], and the bigger this value, the more preferred this object. By contrast, in qualitative preference, the preference is represented by a binary relation [3] and some orders. For example, I prefer tea and coffee at level 0.5 and 0.3 respectively is quantitative preference, whereas, I prefer coffee to tea is a qualitative preference. Qualitative preferences are often easier to express, whereas, expressing a specific quantitative level of preference values could be difficult owing to cognitive difficulties [4]. Consequently, it is a natural way to allow the user to express his preference in a qualitative way.

Among the preference languages, CP-nets (conditional preference networks) is a graphical model for compactly representing conditional and qualitative preference relation [5]. In this paper, we investigate two issues of CP-nets, *i.e.*, reviewing CP-nets application and designing dominance querying approach for CP-nets. The remainder of the paper is organized as follows. Following the brief introduction of preference and CP-nets, Section 2 provides some applications about CP-nets. In Section 3, we give a brief overview of some concepts related to CP-nets, and illustrate these concepts by a fashionable example. In Section 4, we present induced graph of CP-nets, and discuss the dominance querying

ISSN: 2005-4270 IJDTA

problem with respect to any binary-valued structure, which can be solved as a single source shortest path problem by designing Johnson algorithm. Section 5 contains the experimental evaluation of the proposed algorithm. Finally, in Section 6, we conclude the article and discuss a number of interesting directions for future theoretical research and applications.

2. CP-Nets Application

Preference handling has become an flourishing topic in many computational fields such as artificial intelligence (AI), databases (DB), decision theory (DT), and human-computer interaction (HCI). Preference models are needed in decision-support systems such as web-based recommender systems [6,7], in personalized query such as Preference-SQL [8]. Nearly all areas of AI deal with choice and decision making can thus benefit from these preferences handling approaches. Therefore, we first give the application of qualitative preference model — CP-nets.

2.1. CP-Nets in Assistant Decision

The need for acquiring efficient decision-making techniques, together with the desire to simplify the processes of knowledge acquisition, have motivated many researchers to design qualitative decision tools [9]. Fortunately, CP-nets is perfectly able to identify qualitative preference with conditional dependence relationship between these decision variables [5]. A growing application area for CP-nets is automated constraint-based product configuration, whose task is to assemble a number of components to a product, meanwhile satisfying the given compatibility constraints. Considering a simple example of assemble a personal computer, the first considering thing is computer brand, now, I strictly prefer ‘IBM’ to ‘DELL’ brand, while my preference between ‘laptop’ and ‘desktop’ type is conditioned on the selected brand: if IBM brand is selected, I prefer laptop type, otherwise I prefer desktop type if DELL brand is selected. There are four types of configuration for this computer, they are (IBM, laptop), (IBM, desktop), (DELL, laptop) and (DELL, desktop), how to choose the most preferred computer is a basic problem for the buyer. If we model this scenario by a CP-net, and by virtue of some reasoning techniques [2], we can easily help the buyer to determine which type of computer is the most preferred one. This example shows that CP-net can help customer or dealer to aid decision making, in addition, there are a variety of preference-aware systems in reality today, such as personal assistants and personalized user interfaces, in which preference-aware agents can help us shop, arrange photo albums and perform many other daily functions [10]. In particular, preferences are also ubiquitous in real life, for instance, when purchasing the cars, agent A might prefer a cheap car to an expensive car, and a car with bigger space may be more desirable at the same price. Thus, the automobile dealers need reason about the customers’ demand according to their preferences statement, and thereby offering corresponding car configuration to meet the customer’s demand. Aforementioned examples show that CP-nets model can help people to make decision, and greatly facilitate people’s behavioral selection.

2.2. CP-Nets in Preference Database Query

Recently, the need for incorporating preference query in database area is a very important issue in a variety of applications ranging from electronic commerce to personalized search engines, and a good deal of promising research works have been dedicated to this topic. Among which, the most interesting work is to extending standard SQL with preference facilities in order to provide personalized query [11], and perform preference query [12]. More precisely, the interest in the database context focuses on the notion of Top k [13] and Skyline preference query [14], and concentrates on the development of efficient methods for evaluating these queries [11,15]. Preference queries aim to retrieve objects that better match user's requirements from large databases, in literature [16], Ciaccia investigates the possibility of adopting CP-nets for database preference query, and introduces a new totalitarian semantics to query preference. Concurrently, within the database community, some work have been done to integrate preference handling into database query languages like preference SQL, which are built on a variety of preference constructors under a strict partial order semantics. Levandoski introduces a framework for extensible preference evaluation in database systems, and aims to support a wide-array of preference evaluation methods in a single extensible code base [17]. Beyond all that, some sophisticated algebraic optimization techniques and preference queries evaluation algorithms have been developed, for example, Endres attempts to transform queries over TCP-net (the extension of CP-nets) model into database preference queries systematically [18]. In a nutshell, being a preference model, CP-nets can enrich and expand standard SQL operations in database area so that it can realize corresponding personalized query operations.

2.3. CP-Nets in Social Choice

In recent years, there has been a surge of interesting in applied computational complexity in society choice scenario (such as voting and negotiation). In these social choice, modeling users' preferences is an inevitable work, for this reason, an agent always represents its preference by specific preference model such as CP-net, which allows users to order their preference based on the different combinations of attributes. In literature [19], user's preferences are described by CP-net graphical model, which can be used in referring whether the negotiation outcome should be converge to a stable state (optimal outcome) based on Borda scoring heuristic information, so agent can easily obtain the most preferred outcome. Conitzer develops a framework for preference aggregation on multiple binary issues, where each of the agents' preferences is represented by a CP-net [20], and the aggregation rule is majority aggregation rule. Similarly, when voting for a candidate, Li proposes an efficient SAT-based approach to compute the majority winning alternatives, in which the ballot submitted by each individual reflects some aspect of his preferences [21], and the voting protocol is charged with aggregating these preferences into a collective decision. Mattei investigates the computational complexity of finding optimal bribery schemes in voting domains where each voter uses CP-net to represent their inter-dependencies preference for candidates [22]. Maran proposes a way to model voting influence when agents express their preferences by CP-nets, and define procedures of aggregating preferences for resisting the bribery [23]. Pini investigates a multi-agent scenario where agents express their preferences over a large set of decisions via soft constraints (CP-nets is a special case of soft constraint), and study their resistance to bribery attempts to influence aggregation result [24].

2.4. Challenges

Preference handling is a very important research issue in many fields [25], some scholars have studied computational aspects of CP-nets [5,26] and related issues (such as preference queried [16] and preference learning [4]) more carefully for some years, and obtained very fruitful results in complexity theory [26]. However, there does not exist a concrete exact dominance querying algorithm to determine whether an outcome is better than the other one, if two outcomes of arbitrarily binary-valued CP-net are given. That is to say, we does not determine whether existing an outcome sequence in which i-th outcome is preferred to (i+1)-th outcome. Hence, reviewing of CP-nets application and developing an exact dominance querying algorithm for any binary-valued CP-net, become the motivation of this paper.

2.5 Our Approach and Contributions

The main contributions of this article can be summarized as follows.

1. From graphical model knowledge representation and reasoning perspective, we survey the classification of preference research. More importantly, some key applications of CP-nets are given. From these applications, we can conclude that CP-nets is particular suitable for preference knowledge representation.

2. By studying induced graph of CP-nets, we observe that there exists the intimate connection between dominance querying problem and single source shortest path problem [27], and design an exact dominance querying algorithm — Johnson algorithm — with respect to any binary-valued structure CP-nets based on the sparsity of the induced graph of CP-net. Moreover, as a by-product, Johnson algorithm can also determine the minimal flip number for improving an outcome.

3. Some Definitions about CP-Nets

In this section, we give some background and definitions that will be used throughout the rest of the paper.

3.1 Basic Concepts

Definition 1 (Attribute of CP-nets) Let $V = \{X_1, X_2, \cdots, X_n\}$ be a set of variables (attributes), $Dom(X_i)$ be a finite domain associated with each variable X_i. Outcome space $\Omega = \times_{i=1}^{n} Dom(X_i)$ is all the possible outcomes, $o \in \Omega$ is an outcome of Ω. If the outcome o and o' are assigned the same values to all attributes except differing only one attribute, we think they have swappable relation.

In reality, preference for one variable always depends on the other variables assignment, while there exists a link from variable X_j to X_i, X_j is called a parent of X_i, we use $Pare(X_i)$ to denote the parents set of variable X_i, $Pare(X_i)$ can determine the user's preference order over all the possible values of $Dom(X_i)$.

Definition 2 Let $\succ$ be a binary relation over outcomes Ω.

1. $\succ$ is a strict partial order, that is

irreflexive: $\forall o \in \Omega \rightarrow o \gg o$,

antisymmetric: $(\forall o, o' \in \Omega)(o \succ o' \wedge o \neq o' \rightarrow o' \gg o)$,

transitive: $(\forall o, o', o'' \in \Omega)(o \succ o' \wedge o' \succ o'' \rightarrow o \succ o'')$.

We write $o \succ o'$ to denote that o strictly prefer to o'.

2. Given two outcomes $o, o' \in \Omega$,if $o \gg o'$ and $o' \gg o$, we say o and o' are incomparable.

If there exists two outcomes which are incomparable in Ω, the preference relation $\succ$ is called incomplete. If o is incomparable with o', we can write it with $o \perp o'$ (*i.e.*, neither $o \succ o'$ nor $o' \succ o$) [28].

In this paper, all the preference statements are represented by relational operator '$\succ$'.

3.2 Syntax of CP-Nets

The preference information captured by a CP-net can be viewed as a set of local preference assertions for specific single variable X_i, the preference assertion based on a single variable of CP-nets is preferentially independent of some other variables.

Definition 3 Let $V = \{X_1, X_2, \cdots, X_n\}$ be a set of variables, Ω be all the outcomes.

1. Conditional preference table (*CPT* or CP-table) describes user preferences over the values of $Dom(X_i)$ given every combination of its parent values $Dom(Pare(X_i))$. $CPT(X_i)$ is annotated at the vertex X_i of CP-net.

2. A CP-net is a directed graph $N = \langle V, CE \rangle$, in which, V is vertices set, CE is the set of directed edges which represents dependency relation between attributes. In every vertex $X_i \in V$, it is also annotated with by $CPT(X_i)$, which express the local preference on the values of its domain $Dom(X_i)$ given all possible combinations of values of its parents.

3.3 Semantics of CP-Nets

The formal semantics of CP-nets can be interpreted by a ceteris paribus interpretation and the worsening flipping sequence. we first give the definition of ceteris paribus on CP-nets and worsen flipping sequence.

Definition 4 Let $N = \langle V, CE \rangle$ be a CP-net, if $o_1 \succ o_2$ entail $o_1x_i \succ o_2x_i$, we called that the preference $o_1 \succ o_2$ follow the ceteris paribus semantics. Namely, all else be equal.

An outcome sequence $o_1 \succ o_2 \succ \cdots \succ o_m$, where $o_i \in \Omega$ is called the worsen flipping sequence if each two successive outcomes have flip relation. Namely, outcome o_i and o_{i+1} only differ on one attribute and equal to every other attribute.

For instance, the statement "I prefer red wine to white wine if meat is served" asserts that, given two meals that differ only in the kind of wine served and both containing meat, the meal with a red wine is preferable to the meal with a white wine. As a second example, if $a_1 \succ a_0$, and $Dom(B)= \{b_0, b_1\}$, $Dom(C) = \{c_0, c_1, c_2\}$, we have the following preference statement:

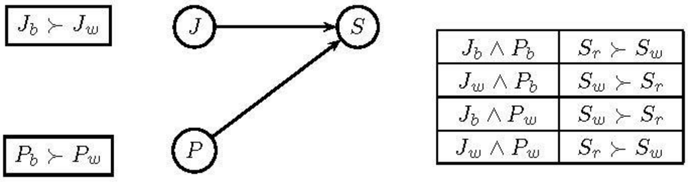
$$a_1b_0c_0 \succ a_0b_0c_0,\ a_1b_0c_1 \succ a_0b_0c_1,\ a_1b_0c_2 \succ a_0b_0c_2\,,$$
$$a_1b_1c_0 \succ a_0b_1c_0,\ a_1b_0c_1 \succ a_0b_1c_1,\ a_1b_1c_2 \succ a_0b_1c_2\,.$$

furthermore, if $a_1b_0 \succ a_1b_1$, we have

$$a_1b_0c_0 \succ a_1b_1c_0,\ a_1b_0c_1 \succ a_1b_1c_1,\ a_1b_0c_2 \succ a_1b_1c_2.$$

Example 1 (Evening Dress [5]) A typical CP-net is depicted in Figure 1, it describes the evening dress scenario based on preferences for three variables J, P and S, standing for the jacket, pants, and shirt, respectively. I unconditionally prefer black to white as a color for both the jacket and the pants. While my preference between the red and white shirts is conditioned on the combination of jacket and pants: if they are of different colors, then I prefer a white shirt, because a red shirt will probably make my outfit too gaudy. Otherwise, if the jacket and the pants have the same color, then a white shirt will make my outfit too colorless, so I prefer a red shirt.

As shown in Figure 1: $N = \langle V, CE \rangle$, where $V = \{J, P, S\}$, $Dom(J) = \{J_b, J_w\}$, $Dom(P) = \{P_b, P_w\}$, $Dom(S) = \{S_r, S_w\}$, $CE = \{<J, S>, <P, S>\}$. There exists three conditional preference tables — $CPT(J)$, $CPT(P)$, and $CPT(S)$, which are labeled in vertex J, P, and S respectively.



$J_b \wedge P_b$	$S_r \succ S_w$
$J_w \wedge P_b$	$S_w \succ S_r$
$J_b \wedge P_w$	$S_w \succ S_r$
$J_w \wedge P_w$	$S_r \succ S_w$

Figure 1. A CP-Net for "Evening Dress"

Proposition 1 Given an acyclic CP-net and two swappable outcomes o_1 and o_2 over Ω, the time complexity of determining whether $o_1 \succ o_2$ or $o_2 \succ o_1$ (swappable relation query) is $O(n)$.

Proof Obviously, swappable outcomes o_1 and o_2 have only one different variable, it can be determined which variable is different in $O(n)$ time, without lost of generality, provided that X_i is the variable in which o_1 and o_2 are different, looking up in $CPT(X_i)$ can make sure $o_1 \succ o_2$ or $o_2 \succ o_1$ easily.

CP-nets express preference statements by $CPT(X_i)$ explicitly, based on semantics of CP-nets — ceteris paribus, and preference transitivity (or worsen flipping sequence), we can derive plenty of preference relations which are characterized by following Proposition 2.

Ceteris paribus and flip relation (write by *FR*) have the following relationship.

Definition 5 If a pair of outcomes *o* and *o'* have swappable relation and $o \succ o'$, we say that *o'* and *o* have flip relation *FR*, written as *o'FRo*.

Proposition 2 Let $N = \langle V, CE \rangle$ be a binary-valued CP-net, u: $x_i \succ \overline{x_i}$ be a rule (statement) in $CPT(X_i)$, where $u \in Dom(Pare(X_i))$, then based on "ceteris paribus" semantics, we can obtained the flipping preference statement:

$$FR = \bigcup_{X_i \in V} \bigcup_{\varphi \in CPT(X_i)} \{ux_i y \succ u\overline{x_i} y \mid y \in Dom(V \backslash Pare(X_i) \backslash \{X_i\}\}.$$

Furthermore, based on transitive of preference, all the preference represented by CP-net can be defined by

$$\succ_N = TransitiveClosure(FR).$$

That is to say, $o_i \succ_N o_j$ if and only if there exists a worsen flipping sequence $o_i o_1 o_2 \cdots o_k \cdots o_j$ such that every successive two outcome $(o_k, o_{k+1}) \in FR$, or $o_k \succ o_{k+1}$.

4. Dominance Querying

The main purpose of any preference representation language is to support answering various queries about the preferences of the decision maker, CP-net is no exception. There are two fundamental types of queries:

- Outcome optimization query [5,29]: determine the best outcome among all the outcomes space Ω represented by CP-net, that is, we look for the most preferable outcome from outcome space Ω .
- Outcome comparison between a pair of outcomes [26]: namely, given outcomes o_1 and o_2, decide whether $o_1 \succ o_2$ or not.

The first problem have been solved by taking advantage of *forward sweep procedure* in time linear in the number of variables, however, the second problem has not been solved. Because CP-net expresses only partial order of outcome space Ω [5], and some outcomes may be incomparable, so dominance querying is not easy. In the following, we introduce the induced graph of CP-nets firstly, therefore, study dominance querying method for CP-nets.

4.1. Induced Graph of CP-Nets

According to Proposition 1, we can determine the dominance relation between a pair of swappable outcomes quickly. In order to judge the dominance relation between any two outcomes, we introduce the induced preference graph and dominance querying with respect to any structure of CP-nets.

Definition 6 Let $N = \langle V, CE \rangle$, then $N' = \langle \Omega, IE \rangle$ is the directed graph of *N*, where Ω is shown in Definition 1, *IE* is the directed edges set which consists of swappable outcomes and for any directed edge $e \in IE$, sink of *e* is preferred to source of *e*. We say *N'* is the induced graph of *N*.

Figure 2 is the induced graph of CP-net as shown in Figure 1. Now, we study the counting properties of the induced graph of CP-net, which can measure the size of induced graph of CP-net, and will be used in the algorithm complexity analysis in Corollary 2 of Section 4.

Proposition 3 (sparse of induced graph) Let $N' = \langle \Omega, IE \rangle$ be the induced graph of binary-valued CP-net *N*, that is, $|V| = n$ and $|Dom(X_i)| = 2$, then *N'* is a sparse directed graph.

Proof We first resolve the cardinality of vertices set Ω and edges set *IE*. Obviously, $|\Omega| = 2^n$, because outcome *o* consists of *n* variables, and each variable has only one assignment differing other swappable outcome, so *o* has *n* swappable outcomes. Therefore, all the number of pair swappable outcomes in Ω is $n * 2^n$. Due to the symmetry of swappable outcomes, and directed property of edges in *N'*, *i.e.*, there is only one directed edge to connect a pair of swappable outcomes, so the number of edges is $n * 2^n / 2 = n * 2^{n-1}$. With the $|\Omega|$ and *IE* at hand, the ratio of the number of edges to the number of vertices is $n * 2^{n-1} / 2^n = n / 2$, therefore *N'* is a sparse graph.

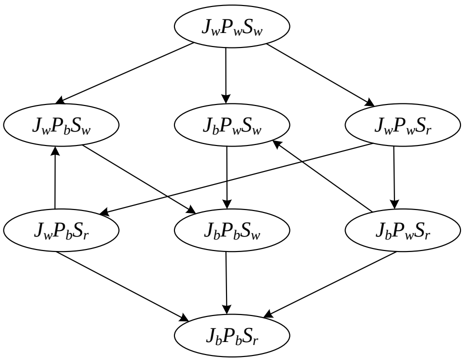


Figure 2. The Induced Graph of CP-Net

4.2. Dominance Querying Problem

Definition 7 Let $N' = \langle \Omega, IE \rangle$ be the induced graph of CP-net *N*, we call that *o* dominant *o'* (written as $o \succ o'$) if there exists a directed path from *o'* to *o*, where *o, o'*$\in \Omega$. The procedure of judging whether *o* dominates *o'* is called dominance querying.

Proposition 4 Let $N = \langle V, CE \rangle$ be a CP-net, $N' = \langle \Omega, IE \rangle$ be its induced graph, $o, o' \in \Omega$, then,

1. *IE* = *FR*, where *FR* is flip relation preference relation as shown in Proposition 2.

2. If *o* and *o'* are incomparable $(o \gg o'$ and $o' \gg o)$, then the number of paths from *o'* to *o* is 0; If $o' \succ o$, there exists one path at least from *o* to *o'*.

Example 2 Considering the induced graph of CP-net as shown in Figure 2, let o_1 = "$J_wP_wS_w$", o_2 = "$J_wP_wS_r$", o_4 = "$J_wP_bS_w$", o_7 = "$J_bP_wS_r$", then the number of paths from o_4 to o_7 is 0, there is only one path from o_2 to o_4, the number of paths from o_1 to o_4 is 2.

Definition 8 Let $N' = \langle \Omega, IE \rangle$ be the induced graph of CP-net *N*, $o, o' \in \Omega$, we write *Distance*(*o, o'*) to denote the length of the shortest path from *o* to *o'*. If there is no path from *o* to *o'*, we let *Distance*(*o, o'*) = ∞.

Proposition 5 If *o* and *o'* are incomparable, then *Distance*(*o, o'*) = ∞; if *oFRo'*, then *Distance*(*o, o'*) = 1; if $o' \succ o$, then *Distance*(*o, o'*) $\geqslant$ 1 and also is a bounded integer.

Given an acyclic CP-net *N*, dominance querying $o \succ o'$ can be treated as a searching procedure for an worsen flipping sequence from the more preferred outcome *o* to the less preferred outcome *o'*, this intuitive phenomena inspire us investigate dominance querying problem by the related approach which harness the technologies described in the Single Shortest Path Problem.

4.3. Dominance Querying Based on SSSP

Definition 9 (Single Source Shortest Path, SSSP) Let $N' = \langle \Omega, IE \rangle$ be the induced graph of CP-net *N*, we want to find a shortest path from a given source vertex $u \in \Omega$ to every other vertex $v \in \Omega$ \{*u*}, this problem is called the single source shortest path problem.

Many other problems can be solved by the algorithm for the SSSP, including the following variants: (1) Point to point shortest path problem [30]: finding a shortest path from given pair of endpoint *u* to *v*, if we solved the single source problem with source vertex *u*, we can solve this problem easily. (2) All pairs shortest paths problem [31]: finding all the shortest path for every pair of vertices *u* and *v*. This problem can be solved by running a single source shortest path algorithm once from each vertex $u \in \Omega$.

Theorem 1 For the induced graph *N'* of binary-valued CP-net, $o, o' \in \Omega$ and $o \neq o'$, if 1 ≤ *Distance*(*o*, *o'*) ≤ 2^n , then $o' \succ o$. That is, if the length of shortest path from *o* to *o'* is neither 0 nor more than 2^n , then $o' \succ o$; if the length of shortest path is more than 2^n , namely, there is no path from *o* to *o'*, then $o' \gg o$. In which *Distance*(*o*, *o'*) denotes the length of shortest path from *o* to *o'*.

Proof For the induced graph of binary-valued CP-net, there may exist 2^n improving flips at most from vertex *o* to *o'*, so the path of shortest path is not more than 2^n . In addition, because of *o* ≠ *o'*, then *Distance*(*o*, *o'*) ≠ 0.

Corollary 1 For the induced graph *N'* of binary-valued CP-net, $o, o' \in \Omega$ and *o*≠*o'*, if *Distance*(*o*, *o'*)= *k*, therefore we can flip outcome *o* *k* numbers to get a more preferred outcome *o'*.

For the induced graph of CP-net, if the length of the shortest path from outcome *o* to *o'* is *k* (*Distance*(*o*, *o'*) = *k* < 2^n), then $o' \succ o$. Starting from outcome *o*, we can only flip *k* times to get outcome *o'*, therefore, dominance querying problem could be phrased as a shortest path problem or path existence problem. We note here the similarity between dominance querying and SSSP, this similarity and the properties of induced graph of CP-nets in fact prompt us to present a version of Johnson algorithm to execute dominance querying.

Algorithm 1 is the procedure for determining whether $t \succ s$ based on computing the shortest path length from source vertex *s* to all the other vertices.

Algorithm 1 Dominance Query (CP-nets $N = \langle V, CE \rangle$, Outcome *s*, *t*)

```
Input: CP-nets $N = \langle V, CE \rangle$, arbitrary source outcome $s$ and destination outcome $t$
Output: Whether $t \succ s$ according to the preference order represented by $N$
int $A[m][m]$ ;
    // $m = 2^n$, define adjacency matrix $A$[ ][ ]
int $Distance[m]$;
    // Define distance array Distance[ ] to store all the shortest paths length from $s$ to $o \in \Omega$, i.e.
      $Distance[t]$ is the shortest path length from $s$ to $t$
A ← $Initialization(V, CE)$;
    // Construct adjacency matrix for $N'$
$Distance$[ ] ← $Johnson(A, s)$;
if   $Distance[t] \neq \infty$   then    // $Distance[t]$ is shortest path length from $s$ to $t$
    $t \succ s$
else
    $t$ » $s$
end
```

The main running time is dominated by the Procedure 1, the adjacency matrix *A* of induced graph *N'* can be regarded as a materialized view [32] which only need be solved one time and store in memory for other program invoking, so we ignore the time cost of Procedure *Initialization*(). In each iteration of Procedure 1, we perform a procedure to extract minimal value which is stored in *D*[ ] array. If the min-priority queue *D*[ ] in Johnson's algorithm is implemented by a binary heap [33], it will yield a total running time $O(m'\log n')$ [30, 34], where *n'* is the number of vertices of induced graph, *m'* is the number of edges of the induced graph *N'*.

Theorem 2 SSSP problem from a source vertex *s* in a graph with *n'* vertices and *m'* edges can be solved in $O(m'\log n')$ time, and used O(*n'*) extra space [30].

Corollary 2 Dominance querying algorithm of binary-valued CP-net *N* with *n* attributes by Algorithm 1 is soundness, and its time complexity is $O(2^n * n^2)$.

Proof Obviously, if there is a shortest path from vertex o_i to o_j, then the value of $Distance[o_j]$ is not ∞, so $o_j \succ o_i$ thus doing the dominance querying for a pair of outcomes is soundness. In addition, as known from Theorem 2, the time complexity of SSSP algorithm is $O(m'\log n')$, by Proposition 3, the number of edges $m' = n * 2^{n-1}$, the number of vertices $n' = 2^n$, therefore, $O(m'\log n')$=$O(n * 2^{n-1} * \log 2^n) = O(2^n * n^2)$, this means that we can fulfill dominance querying in time $O(2^n * n^2)$.

```
Procedure Initialization($V$, $CE$)
Input: Binary valued CP-net $N = \langle V, CE \rangle$
Output: Adjacency matrix $A[m][m]$, $m = 2^n$
int $A[m][m]$;
OutcomeSpace $\Omega' \leftarrow \varnothing$;
$\succ_V \leftarrow \varnothing$;
for each $X_i \in V$ do
    $\succ_{Xi} \leftarrow \varnothing$;
    for each $(\varphi = (u : x_{i1} \succ x_{i2}) \in CPT(X_i)$ do
        $\succ \varphi \leftarrow \varnothing$;
        for each $y \in Dom(V \setminus Pare(X_i) \setminus \{X_i\})$ do
            $\succ \varphi \leftarrow (\succ \varphi \cup \{(ux_{i1}y, ux_{i2}y)\})$;
            $\Omega' \leftarrow (\Omega' \cup \{ux_{i1}y, ux_{i2}y\})$;
        end
    $\succ_{Xi} \leftarrow (\succ_{Xi} \cup \succ \varphi)$;
    end
    $\succ_V \leftarrow (\succ_V \cup \succ_{Xi})$;
end
    // translate preference relation $\succ_V$ into adjacent matrix $A$
foreach $o_i, o_j \in \Omega'$ do
    if $(o_i, o_j) \in \succ_V$ then
        $A[i][j] \leftarrow 1$;
    else
        $A[i][j] \leftarrow \infty$;
    end
end
return $A$;
```

```
Procedure Johnson($A$, outcome $s$)
Input: Adjacency matrix $A$ of $N'$, and a source outcome $s$
Output: The set of shortest paths from vertex $s$ to all other vertices $D[u]$, $u \in \Omega'$
int $D[\,]$;
    // $D[u]$ is the shortest distance path array from $s$ to $u$
OutcomeSpace $Q \leftarrow \Omega'$;
    // $\Omega'$ is the outcome space of CP-net $N$
$D[s] \leftarrow 0$;
forall the $v \in \Omega' \setminus \{s\}$ do
    $D[v] \leftarrow \infty$;
end
while $Q \neq \varnothing$ do
    $u \leftarrow \arg\min_{q \in Q} (D[q])$;
    // $u$ is the vertex whose $D[\,]$ value is minimal
    $Q \leftarrow Q \setminus \{u\}$;
    foreach $v \in Q \wedge A[u][v] = 1$ do
        $D[v] \leftarrow \min(D[v], D[u] + A[u][v])$;
    end
end
return $D$;
```

Example 3 Find the shortest path from vertex o_1 to other vertexes in graph $N' = \langle \Omega, IE \rangle$ (as shown in Figure 2) by Algorithm 1, where $\Omega = \{o_1, o_2, o_3, o_4, o_5, o_6, o_7, o_8\}$, o_1 =“$J_wP_wS_w$”, o_2 =“$J_wP_wS_r$”, o_3 =“$J_wP_bS_r$”, o_4 =“$J_wP_bS_w$”, o_5 =“$J_bP_bS_w$”, o_6= “$J_bP_bS_r$”, o_7 =“$J_bP_wS_r$”, o_8 =“$J_bP_wS_w$”. IE = {<o_1, o_2>, <o_2, o_3>, <o_3, o_4>, <o_4, o_5>, <o_5, o_6>, <o_1, o_4>, <o_1, o_8>, <o_2, o_7>, <o_3, o_6>, <o_7, o_6>, <o_7, o_8>, <o_8, o_5>}.

The adjacency matrix A of induce graph N' is as follows:

$$A = \begin{bmatrix} 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

After running *Johnson*() algorithm, *Distance*[o_1] = [∞, 1, 2, 1, 2, 3, 2, 1],
in the same manner, we can get

Distance[o_2] = [∞, ∞, 1, 2, 3, 2, 1, 2], *Distance*[o_3] = [∞, ∞, ∞, 1, 2, 1, ∞, ∞],

Distance[o_4] = [∞, ∞, ∞, ∞, 1, 2, ∞, ∞], *Distance*[o_5] = [∞, ∞, ∞, ∞, ∞, 1, ∞, ∞],

Distance[o_6] = [∞, ∞, ∞, ∞, ∞, ∞, ∞, ∞], *Distance*[o_7] = [∞, ∞, ∞, ∞, 2, 1, ∞, 1],

Distance[o_8] = [∞, ∞, ∞, ∞, 1, 2, ∞, ∞].

Because $Distance[o_1][o_j]$ $(j\neq 1) \neq \infty$, so $o_j \succ o_1$ $(j\neq 1)$.

5. Experimental Evaluation

5.1. Experimental Environment

Platform In this section, we report an extensive empirical study over a real and synthetic data set to examine our proposed approach and the approach of [35]. All the experiments were conducted on a PC with a 3.20-GHz Intel Pentium, 2 GB of internal memory, running Windows 7 Operating system. The algorithm is implemented in VC++ 9.0 and SMILE C++ library which is a structural modeling, inference, and learning engine.

Parameters We have conducted sensitivity experiments versus the number of attributes $|V|$, the size of the each attribute domain $Dom(X_i)$, the number of parents of attribute $|Pare(X_i)|$.

Metrics Each experiment measures the total query processing time required by each execution algorithm examined. The performance are measured by the total running time *TotalTime* for dominance querying among all the outcome Ω , and the average running time *AverageTime* for a pair of outcome *o* and *o'*, of which, the total time consists of all the time consuming for obtaining all the dominance relation $\succ_N$, the relationship between *TotalTime* and *AverageTime* is

$$TotalTime = AverageTime * |\Omega|.$$

5.2. Experiments on Real Data

For the experiments on real data, all the datasets are derived from the Bayesian networks provided by the University of Pittsburgh decision systems laboratory (available at http:// genie.sis.pitt.edu/ and http://www.bnlearn.com/bnrepository/). We transform these BNs into CP-nets by modifying their conditional probability table to CP-tables. For instance, if in BNs, $Dom(A) = \{a_1, a_2, a_3\}$, and $P(A = a_1) = 0.2$, $P(A = a_2) = 0.5$, $P(A = a_3) = 0.3$, then the transformed CP-table is $CPT(A) = \{a_2 \succ a_3 \succ a_1\}$.

We give the experiment result on ASIA network (sometimes called LUNG CANCER) and SACHS networks. Specifically, ASIA is designed for by probabilities graphical model inferring [36], whereas, SACHS networks is a causal protein-signaling networks derived from multi parameter single-cell data [37]. On these real datasets, the response times (*ms*) in evaluating dominance querying are illustrated by Table 1.

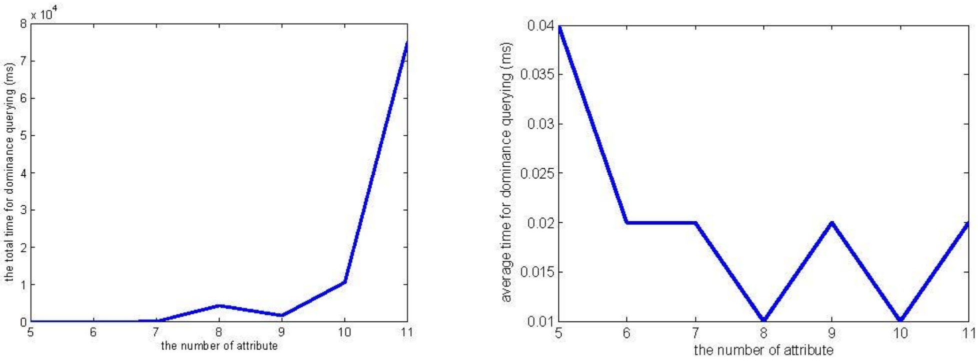
Table 1. Algorithm Performance for ASIA and SACHS Network

Networks	Networks parameters						Running time (*ms*)	
	$\|V\|$	$\|E\|$	*AveDe*	*MaxDe*	*#Para*	$\|\Omega\|$	*ToTi*	*AveTi*
ASIA	8	8	2.00	2	18	256	349	0.05
SACHS	11	17	3.09	3	178	177147	$4.31*10^{11}$	0.04

In Table 1, *AveDe* is Average degree of network, *MaxDe* is maximum in-degree, *#Para* is number of parameters, *ToTi* is the total time consumption for solving all the dominance querying time between all feasible outcome, *AveTi* is the average time consumption for dominance querying.

5.3. Experiments on Synthetic Data

For synthetic data, we randomly generate some binary-value CP-nets by varying the number of variables $|V|$, the size of attribute domain $|Dom(X_i)|$, the structure of the network and the conditional preference table, and see how it affects the performance of the proposed algorithm, here $|V|$ was set to 5, 6,7, 8, 9, 10 and 11 respectively, $|Dom(X_i)|$ is set to 2. The response time for solving dominance querying is illustrated by Figure 3.



(a) Total Time for All Dominance Querying (b) Average Time for Each Dominance Querying

Figure 3. Performance of Dominance Querying

As we can see from above figure, with the increase of $|V|$, the total query time is also increasing, it is because of the squarely increase of ordered pair outcomes, but the average time for each dominance querying is varied not much.

5.4. Some Works about Dominance Query

Among the research of preference handling, preference representation is the most basic work, just because majority of preference applications are expressed by CP-nets [15, 28], so how to solve the fundamental problems — dominance querying over graphical model CP- nets — are essential and pre-requisite. Boutilier summarizes the previous works and specifies the syntax, semantics and some applications of CP-nets such as multimedia document presentation [5], but some basic problems still need to be solved. First, the algorithm of dominance querying with respect to any binary-valued structure CP-nets has not given, with the exception of some special cases such as CP-nets with tree or polytree structure conditional dependencies. Although Goldsmith and his co-authors have proven that dominance querying and consistency of CP-nets are PSPACE-complete [26] by the abstruse STRIPS planning technique [38], unfortunately, they did not provide an intuitive concrete algorithm to demonstrate what they used technique. Recently, Santhanam reduces dominance querying of CP-nets to reachability analysis, and provides an encoding of CP-nets in the form of the Kripke structure for CTL (computational tree logic) [39], thus, dominance querying with acyclic CP-net can be fulfilled by a specific model checker for CTL [40]. More recently, [41] proposes a heuristic approach to make dominance querying in arbitrary acyclic multi-valued CP-nets, their proposed hamming distance approach guides the search process efficiently and allows significant reduction of search space when there is no flip sequence possible, meanwhile, without impacting soundness or completeness of the search process. However, among the above mentioned approaches, they either did not make the complexity analyze for their proposed algorithm [40, 41], or only proved the proposed algorithm complexity but no concrete algorithm [26]. More seriously, most of dominance querying algorithm is only suitable for restricted CP-net structure. However, our exact dominance querying algorithms are suitable for arbitrary acyclic CP-net structure. Moreover, we give its deterministic time complexity analysis as shown in Corollary 2.

6. Summary and Outlook

In this paper, by constructing induced graph of CP-nets and studying its properties, we show that the problem of dominance querying with respect to any binary-valued structure is a single source shortest path problem essentially, as a result, dominance querying problem can be solved by Johnson algorithm, this result lay the theoretical foundation for probing some other properties (such as consistency [26], satisfiability [5]) for CP-nets based on dominance querying.

Further research as follows: (1) studying some other query task, such as skyline query, Top-k query on CP-nets [17]. (2) studying how to efficient store all the preference relation $\succ_N$ as shown in literature [42].

Acknowledgements

The authors would like to thank the anonymous reviewers for their valuable comments. This work was supported by Natural Science Foundation of China (61572419, 61403328, 61403329), Shandong Province Natural Science Foundation (ZR2013FM011, ZR2013FQ023, ZR2013FQ020, ZR2014FL009), and Shandong Province Science and Technology Plan of Education (J14LN23).

References

[1] J. Liu and S. Liao, “Expressive efficiency of two kinds of specific CP-nets”, Information Sciences, vol. 295, no. 2, **(2015)**, pp. 379–394.

[2] J. Liu, “Research on CP-nets and its expressive power”, Acta Automatica Sinica, vol. 37, no. 3, **(2011)**, pp. 290–302.

[3] G. Santhanam, S. Basu, and V. Honavar, “Representing and reasoning with qualitative preferences for compositional systems”, Journal of Artificial Intelligence Research, vol. 42, no. 3, **(2011)**, pp. 211–274.

[4] Y. Chevaleyre, F. Koriche, J. Lang, J. Mengin, and B. Zanuttini, “Learning ordinal preferences on multiattribute domains: The case of CP-nets”, in Preference Learning, Springer-Verlag New York, **(2011)**, pp. 273–296.

[5] C. Boutilier, R. Brafman, C. Domshlak, H. Hoos, and D. Poole, “CP-nets: A tool for representing and reasoning with conditional ceteris paribus preference statements”, Journal of artificial intelligence research, vol. 21, no. 1, **(2004)**, pp. 135–191.

[6] M. Yu, X. Han, X. Gou, J. Yu, F. Lv, and J. Li, “Content-based social network user interest tag extraction”, International Journal of Database Theory & Application, vol. 8, no. 2, **(2015)**, pp. 107–118.

[7] M. de Gemmis, L. Iaquinta, P. Lops, C. Musto, F. Narducci, and G. Semeraro, “Learning preference models in recommender systems”, in Preference Learning, Springer, **(2011)**, pp. 387–407.

[8] W. Kießling and G. Köstler, “Preference SQL: Design, implementation, experiences”, Proceedings of the 28th international conference on Very Large Data Bases, **(2002)** August 990–1001; Hong Kong, China.

[9] R. Brafman, C. Domshlak, and S. Shimony, “Qualitative decision making in adaptive presentation of structured information”, ACM Transactions on Information Systems, vol. 22, no. 4, **(2004)**, pp. 503–539.

[10] J. Goldsmith and U. Junker, “Preference handling for artificial intelligence”, AI Magazine, vol. 29, no. 4, **(2008)**, pp. 9–12.

[11] K. Stefanidis, G.Koutrika, and E.Pitoura, “A survey on representation, composition and application of preferences in database systems”, ACM transactions on database systems, vol. 36, no. 3, **(2011)**, pp. 19:1–19:45.

[12] A. Arvanitis and G. Koutrika, “PrefDB: Supporting preferences as first-class citizens in relational databases”, IEEE Transactions on Knowledge and Data Engineering, vol. 26, no. 6, **(2014)**, pp. 1430–1446.

[13] I. Ilyas, G. Beskales, and M. Soliman, “A survey of top-k query processing techniques in relational database systems”, ACM Computing Surveys, vol. 40, no. 4, **(2008)**, pp. 11:1–11:58.

[14] J. Lee and S.-w. Hwang, “Toward efficient multidimensional subspace skyline computation”, The VLDB Journal, vol. 22, no. 4, **(2013)**, pp. 1–17.

[15] F. S.Pereira and S. de Amo, “Evaluation of conditional preference queries”, Journal of Information and Data Management, vol. 1, no. 3, **(2010)**, pp. 503–518.

[16] P. Ciaccia, “Querying databases with incomplete CP-nets”, in Multidisciplinary Workshop on Advances in Preference Handling, Vienna, Austria, **(2007)**.

[17] J. J. Levandoski, A. Eldawy, M. F. Mokbel, and M. E.Khalefa, “Flexible and extensible preference evaluation in database systems”, ACM Transactions on Database Systems, vol. 38, no. 3, **(2013)**, pp. 17:1–17:43.

[18] M. Endres and W.Kießling, “Transformation of TCP-net queries into preference database queries”, Proceedings of the ECAI 2006 Multidisciplinary Workshop on Advances in Preference Handling, Italy, **(2006)**.

[19] R. Aydogan, T. Baarslag, K. Hindriks, C. Jonker, and P. Yolum, “Heuristic-based approaches for CP-nets in negotiation”, Proceedings of the 4th International Workshop on Agent-based Complex Automated Negotiations, Taipei, Taiwan, **(2011)**.

[20] V. Conitzer, J. Lang, and L. Xia, “Hypercubewise preference aggregation in multi-issue domains”, Proceedings of the 22nd International Joint Conference on Artificial Intelligence, Barcelona, Catalonia, Spain, **(2011)**.

[21] M. Li, Q. Vo, and R. Kowalczyk, “Majority-rule-based preference aggregation on multi-attribute domains with structured preferences”, Proceedings of the 10th International Conference on Autonomous Agents and Multiagent Systems, vol. 11, **(2011)**, pp. 659–666.

[22] N. Mattei, M. S. Pini, F. Rossi, and K. B. Venable, “Bribery in voting with CP-nets,” Annals of Mathematics and Artificial Intelligence, **(2013)**, pp. 1–26.

[23] A. Maran, N. Maudet, M. S. Pini, F. Rossi, and K. B. Venable, “A framework for aggregating influenced CP-nets and its resistance to bribery”, Proceedings of the 27th AAAI Conference on Artificial Intelligence, **(2013)** 668–694.

[24] M. S. Pini, F. Rossi, and K. B. Venable, “Resistance to bribery when aggregating soft constraints”, Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems, **(2013)** 1301–1302; St. Paul, MN, USA.

[25] C. Domshlak, E. Hüllermeier, S. Kaci, and H. Prade, “Preferences in AI: An overview”, Artificial Intelligence, vol. 175, no. 7-8, **(2011)**, pp. 1037–1052.

[26] J. Goldsmith, J. Lang, M. Truszczynski, and N. Wilson, “The computational complexity of dominance and consistency in CP-nets”, Journal of Artificial Intelligence Research, vol. 33, no. 1, **(2008)**, pp. 403–432.

[27] J. B. Orlin, K. Madduri, K. Subramani, and M. Williamson, “A faster algorithm for the single source shortest path problem with few distinct positive lengths”, Journal of Discrete Algorithms, vol. 8, no. 2, **(2010)**, pp. 189–198.

[28] M. Pini, F. Rossi, K. Venable, and T. Walsh, “Incompleteness and incomparability in preference aggregation: Complexity results”, Artificial Intelligence, vol. 175, no. 7-8, **(2011)**, pp. 1272–1289.

[29] K. Purrington and E. Durfee, “NP-Completeness of outcome optimization for partial CP-nets”, Proceedings of the 23rd AAAI Conference on Artificial Intelligence, **(2008)** July 1826–1827; Chicago, Illinois, USA.

[30] T. Cormen, C. Leiserson, and C. Rivest, R.L.and Stein, Introduction to algorithms. Cambridge, Massachusetts London, England: The MIT press, **(2001)**.

[31] L. Planken, M. de Weerdt, and R. van der Krogt, “Computing all-pairs shortest paths by leveraging low treewidth”, Journal of Artificial Intelligence Research, vol. 43, **(2012)**, pp. 353–388.

[32] H. V. Jagadish, “A compression technique to materialize transitive closure”, ACM Transactions on Database Systems, vol. 15, no. 4, **(1990)**, pp. 558–598.

[33] M. L. Fredman and R. E. Tarjan, “Fibonacci heaps and their uses in improved network optimization algorithms”, Journal of the ACM, vol. 34, no. 3, **(1987)**, pp. 596–615.

[34] D. B. Johnson, “Efficient algorithms for shortest paths in sparse networks”, Journal of the ACM, vol. 24, no. 1, **(1977)**, pp. 1–13, 1977.

[35] H. Wang, X. Zhou, W. Chen, and P. Ma, “Top-k retrieval using conditional preference networks”, Proceedings of the 21st ACM International Conference on Information and Knowledge Management, New York, USA, **(2012)**, pp. 2075–2079.

[36] S. L. Lauritzen and D. J. Spiegelhalter, “Local computations with probabilities on graphical structures and their application to expert systems”, Journal of the Royal Statistical Society, Series B (Methodological), vol. 50, no. 2, **(1988)**, pp. 157–224.

[37] K. Sachs, O. Perez, D. Pe’er, D. A. Lauffenburger, and G. P. Nolan, “Causal protein-signaling networks derived from multiparameter single-cell data”, Science, vol. 308, no. 5721, **(2005)**, pp. 523–529.

[38] T. Bylander, “The computational complexity of propositional strips planning”, Artificial Intelligence, vol. 69, no. 1-2, **(1994)**, pp. 165–204.

[39] R. Meolic, T. Kapus, and Z. Brezocnik, “Actlw–An action-based computation tree logic with unless operator”, Information Sciences, vol. 178, no. 6, **(2008)**, pp. 1542–1557.

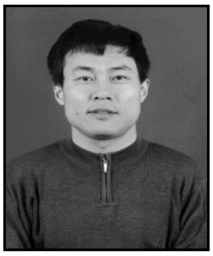
[40] G. R. Santhanam, S. Basu, and V. Honavar,“Dominance testing via model checking”, Proceedings of the 24th AAAI Conference on Artificial Intelligence, Atlanta, Georgia, USA, **(2010)**, pp. 357–362.

[41] M. Li, Q. Vo, and R. Kowalczyk, “Efficient heuristic approach to dominance testing in CP-nets”, Proceedings of the 10th International Conference on Autonomous Agents and Multiagent systems, Taipei, Taiwan, **(2011)**, pp. 353–360.

[42] R. Jin, N. Ruan, Y. Xiang, and H. Wang, "Path-tree: An efficient reachability indexing scheme for large directed graphs", ACM Transactions on Database Systems, vol. 36, no. 1, **(2011)**, pp. 7:1–7:44.

Authors

Guanlin Xin, she was born in Shandong in 1991. She is a graduate student in Yantai University, China. Her research direction is machine learning, her current research interest is graph model learning, such as Bayesian networks, conditional preference networks (CP-nets). Her research concentrates on the structure learning of the CP-nets, combined with probability statistics knowledge.



Jinglei Liu, he was born in Shanxi in 1970. He is an associate professor in Yantai University, China. His research direction includes artificial intelligence and theoretical computer science. At AI aspect, his current research interest are knowledge representation and reasoning for graph model, such as CP-nets, coalition structure graph, *etc.* At theoretical computer science aspect, his research concentrates on finite model theory.