ELSEVIER Artificial Intelligence in Medicine 16 (1999) 73–96

Artificial Intelligence in Medicine

Medical data mining using evolutionary computation

Po Shun Ngan [a,*], Man Leung Wong [b], Wai Lam [c], Kwong Sak Leung [a], Jack C.Y. Cheng [d]

[a] *Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong, Hong Kong*
[b] *Department of Computer Studies, Lingnan College, Hong Kong, Hong Kong*
[c] *Department of Systems Engineering and Engineering Management, The Chinese University of Hong Kong, Hong Kong, Hong Kong*
[d] *Department of Orthopaedics and Traumatology, The Chinese University of Hong Kong, Hong Kong, Hong Kong*

Received 11 November 1997; received in revised form 8 May 1998; accepted 7 July 1998

Abstract

In this paper, we introduce a system for discovering medical knowledge by learning Bayesian networks and rules. Evolutionary computation is used as the search algorithm. The Bayesian networks can provide an overall structure of the relationships among the attributes. The rules can capture detailed and interesting patterns in the database. The system is applied to real-life medical databases for limb fracture and scoliosis. The knowledge discovered provides insights to and allows better understanding of these two medical domains.

Keywords: Data mining; Evolutionary computation; Rule learning; Bayesian networks; Grammar based genetic programming

* Corresponding author.
E-mail addresses: psngan@cse.cuhk.edu.hk (P. Shun Ngan), mlwong@cse.cuhk.edu.hk (M. L. Wong), wlam@se.cuhk.edu.hk (W. Lam), ksleung@cse.cohk.edu.hk (K. S. Leung), jackcheng@cuhk.edu.hk (J. C. Y. Cheng)

0933-3657/99/$ - see front matter
PII: S0933-3657(98)00065-7

1. Introduction

Data mining aims at discovering novel, interesting and useful knowledge from databases [9]. Conventionally, the data is analyzed manually. Many hidden and potentially useful relationships may not be recognized by the analyst. Nowadays, many organizations including modern hospitals are capable of generating and collecting a huge amount of data. This explosive growth of data requires an automated way to extract useful knowledge. Thus, medical domain is a major area for applying data mining. Through data mining, we can extract interesting knowledge and regularities. The discovered knowledge can then be applied in the corresponding field to increase the working efficiency and improve the quality of decision making.

We developed a knowledge discovery system to extract knowledge from data. There are five steps in the system (Fig. 1). Real-life data are collected in the first step. Then, the data must be preprocessed before analysis can be started. The third and fourth step induce knowledge from the preprocessed data. The causality and structure analysis step learns the overall relationships between the variables. A resulting Bayesian network represents the knowledge structure. Based on this knowledge, the user can specify the grammar for the target rules to be discovered from data. This grammar is used for the rule learning step that learns a set of significant rules from the data. In the fifth step, the discovered knowledge is verified and evaluated by the domain experts. The domain experts may discover and correct mistakes in the discovered knowledge. On the other hand, the learned knowledge can be used to refine the existing domain knowledge. Finally, the learned Bayesian network is used to perform reasoning under uncertainty, and the induced rules are incorporated into an expert system for decision making.

In this paper, we present the two knowledge learning steps which are the core of the knowledge discovery system. They both employ evolutionary computation as the search algorithms. This paper is organized as follows. Section 2 introduces the

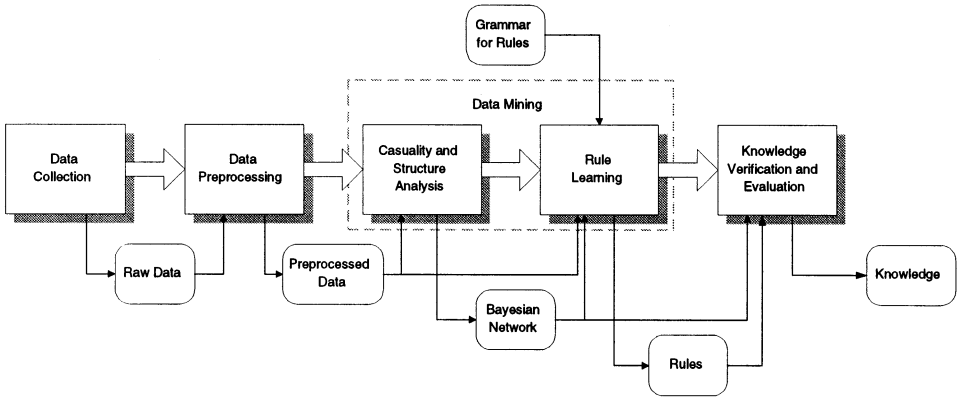

Fig. 1. The knowledge discovery process.

backgrounds on evolutionary computation, Bayesian network learning, and rule learning. Section 3 describes the approaches for learning Bayesian networks. The rule learning process is delineated in Section 4 and the details of the techniques are given in Section 5. The data mining system has been applied to two real-life medical databases. The results are presented in Sections 6 and 7 and the conclusion is presented in Section 8.

2. Backgrounds

2.1. Evolutionary computation

The term evolutionary computation is used to describe algorithms that simulate the natural evolution to perform function optimization and machine learning. They are based on the Darwinian principle of evolution through natural selection. The algorithms maintain a group of individuals to explore the search space. Examples of evolutionary computation include genetic algorithms (GA) [19,13], genetic programming (GP) [24,25], evolutionary programming (EP) [10,11] and evolution strategy (ES) [34,35]. GA uses a fixed-length binary bit string as an individual. Three genetic operators are used to search for better individuals. Reproduction operator copies the unchanged individual. Crossover operator exchanges bits between two parents. Mutation operator randomly changes individual bits. GP extends GA by using a tree structure as the individual. EP emphasizes on the behavioral linkage between parents and their offspring. Mutation is the only genetic operator in EP. There is no constraint on the representation in EP. ES emphasizes on the individual, i.e. the phenotype, to be the object to be optimized. A genetic change in the individual is within a narrow band of the mutation step size and the step size has self-adaptations.

Data mining can be considered as a search problem, which tries to find the most accurate knowledge from all possible hypotheses. Since evolutionary computation is a robust and parallel search algorithm, it can be used in data mining to find interesting knowledge in noisy environment.

2.2. Bayesian network learning

Bayesian network is a formal knowledge representation supported by the well-developed Bayesian probability theory. A Bayesian network captures the conditional probabilities between attributes. It can be used to perform reasoning under uncertainty. A Bayesian network is a directed acyclic graph. Each node represents a domain variable, and each edge represents a dependency between two nodes. An edge from node A to node B can represent a causality, with A being the cause and B being the effect. The value of each variable should be discrete. Each node is associated with a set of parameters. Let N_i denote a node and Π_{N_i} denote the set of parents of N_i. The parameters of N_i are conditional probability distributions in the form of $P(N_i|\Pi_{N_i})$ with one distribution for each possible instance of Π_{N_i}. Fig. 2 is

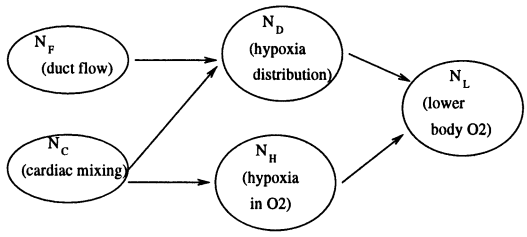

Fig. 2. A Bayesian network structure in a 'Blue' baby domain.

an example Bayesian network structure modeling a medical domain concerned with 'blue' baby diagnosis. This structure shows the causality relationships between the direction of duct flow, the degree of cardiac mixing, hypoxia distribution, hypoxia in O_2 and the degree of lower body O_2. More details of Bayesian networks can be found in [4,15,17].

The main task of learning Bayesian network from data is to automatically find directed edges between the nodes, such that the network can best describe the causalities. Once the network structure is constructed, the conditional probabilities are calculated based on the data. The problem of Bayesian network learning is computationally intractable [7]. However, Bayesian network learning can be implemented by imposing limitations and assumptions. For instance, the algorithms of Chow and Liu [5] and Rebane and Pearl [33] can learn networks with tree structures, while the algorithms of Herskovits and Cooper [18,8] and Bouckaert [3] require the variables to have a total ordering. More general algorithms include Heckerman et. al. [16], Spirtes et. al. [39] and Singh and Valtorta [36]. More recently, Larranaga et. al. [30,29] has proposed algorithms for learning Bayesian networks using GA's.

2.3. Rule learning

A rule is a sentence of the form 'if *antecedents*, then *consequent*'. Rules are commonly used in expressing knowledge and are easily understood by human. Rule learning is the process of inducing rules from a set of training examples. Classical algorithms in this field include AQ15 [32] and CN2 [6]. Previous works in rule learning using evolutionary computation mainly use GA [19,13]. There are two different approaches. In the Michigan approach [20,2], each individual in the GA corresponds to a rule, while in the Pittsburgh approach [37,38] it corresponds to a *set* of rules. The system REGAL [12] uses the Michigan approach and a distributed genetic algorithm to learn first-order logic concept descriptions. It uses a selection operator, called Universal Suffrage operator, to achieve the learning of multi-modal concepts. Another system GABIL [23] uses the Pittsburgh approach. It can adaptively allow or prohibit certain genetic operations for certain individuals. GIL [22] also uses the Pittsburgh's approach and utilizes 14 genetic operators. These operators perform generalization, specialization or other modifications to the individuals at the rule set level, the rule level and the condition level.

3. Causality and structure analysis

In the proposed knowledge discovery process (Fig. 1), the causality and structure analysis process induces a Bayesian Network from the data. The learning approach is based on Lam and Bacchus's work [27,26] on employing the minimum description length (MDL) principle to evaluate a Bayesian Network. EP is employed to optimize this metric in order to search for the best network structure.

3.1. The MDL metric

The MDL metric measures the *total description length* $D_t(B)$ of a network structure B. A better network has a smaller value on this metric. Let $N = \{N_1, \ldots N_n\}$ denote the set of nodes in the network (and thus the set of variables, since each node represents a variable), and Π_{N_i} denote the set of parents of node Ni. The total description length of a network is the sum of description lengths of each node:

$$D_t(B) = \sum_{N_i \in N} D_t\left(N_i, \prod_{N_i}\right) \tag{1}$$

This length is based on two components, the *network description length* D_n and the *data description length* D_d:

$$D_t\left(N_i, \prod_{N_i}\right) = D_n\left(N_i, \prod_{N_i}\right) + D_d\left(N_i, \prod_{N_i}\right) \tag{2}$$

The formula for the network description length is:

$$D_n\left(N_i, \prod_{N_i}\right) = k_i \log_2(n) + d(s_i - 1) \prod_{j \in \prod_{N_i}} s_j \tag{3}$$

where k_i is the number of parents of variable N_i, s_i is the number of values N_i can take on, s_j is the number of values a particular variable in Π_{N_i} can take on, and d is the number of bits required to store a numerical value. This is the description length for encoding the network structure. The first part in the addition is the length for encoding the parents, while the second part is the length for encoding the probability parameters. This length can measure the simplicity of the network.

The formula for the data description length is:

$$D_d\left(N_i, \prod_{N_i}\right) = \sum_{N_i \prod_{N_i}} M\left(N_i, \prod_{N_i}\right) \log_2 \frac{M\left(\prod_{N_i}\right)}{M\left(N_i, \prod_{N_i}\right)} \tag{4}$$

where M(·) is the number of cases that match a particular instantiation in the database.

This is the description length for encoding the data. A Huffman code is used to encode the data using the probability measure defined by the network. This length can measure the accuracy of the network.

3.2. Combining MDL and EP

We combined the MDL metric and EP for Bayesian network learning [28,41]. The flowchart in Fig. 3 shows the process. Each individual represents a network structure, which is a directed acyclic graph (DAG). A set of individuals is randomly created to make up the initial population. Each graph is evaluated by the MDL metric described above. Then, each individual produces a child by performing a number of mutations. The child is also evaluated by the MDL metric. The next generation of population is selected among the parents and children by tournaments. Each DAG B is compared with q other randomly selected DAGs. The tournament score of B equals to the number of rivals that B can win, that is, the number of DAGs among those selected that have higher MDL scores than B. In our setting, $q = 5$. One half of DAGs with the highest tournament scores are retained for the next generation. The process is repeated until the maximum number of generations is reached. The number of the maximum number of generations depends on the complexity of the network structure. If we expect a simple network, the maximum number of generations can be set to a lower value. The network with the lowest MDL score is output as the result.

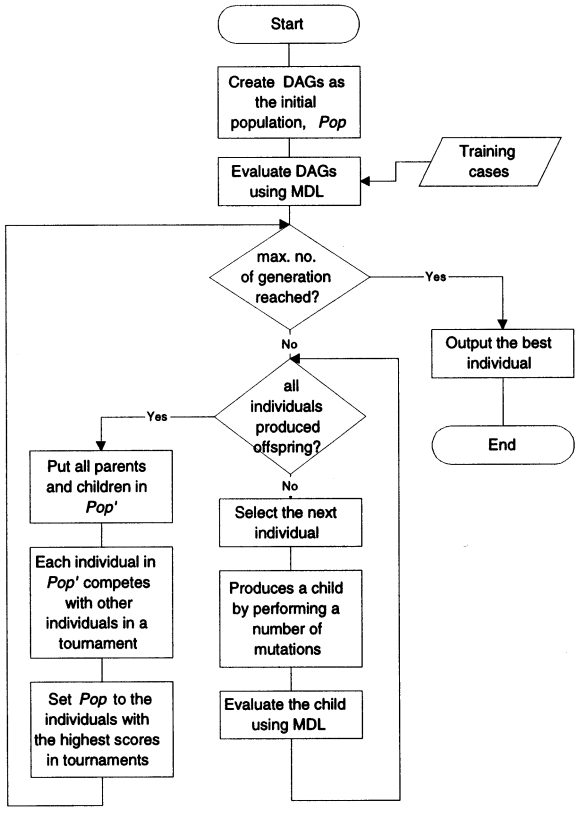

Fig. 3. The flowchart of the Bayesian network learning process.

3.3. The mutation operators

Offspring in EP is produced by using a specific number of mutations. The probabilities of using 1, 2, 3, 4, 5 or 6 mutations are set to 0.2, 0.2, 0.2, 0.2, 0.1 and 0.1 respectively. The mutation operators modify the edges of the DAG. If a cyclic graph is formed after the mutation, edges in the cycles are removed to keep it acyclic. Our approach uses four mutation operators, with the same probabilities of being used:

1. Simple mutation randomly adds an edge between two nodes or randomly deletes an existing edge from the parent.
2. Reversion mutation randomly selects an existing edge and reverses its direction.
3. Move mutation randomly selects an existing edge. It moves the parent of the edge to another node, or moves the child of the edge to another node.
4. Knowledge-guided mutation is similar to simple mutation, however, the MDL scores of the edges guide the selection of the edge to be added or removed. The MDL metric of all possible edges in the network is computed before the learning algorithm starts. This mutation operator stochastically adds an edge with a small MDL metric to the parental network or deletes an existing edge with a large MDL metric.

4. Rule learning

The second step in our data mining process is to learn rules from the data. Our learning approach is based on generic genetic programming (GGP) [43,42,40], which is an extension of GP. It uses a grammar [21] to control the structures evolved in GP.

4.1. The GGP process

The flowchart in Fig. 4 shows the process of using GGP for rule learning. A grammar is provided as a template for rules. The algorithm starts with an initial population of randomly created rules using the user-defined grammar. One individual corresponds to one rule. Each rule is evaluated by a fitness function described in Section 5.1. Then, individuals are selected stochastically to evolve offspring by the genetic operators. Rules with higher fitness scores have higher chances of being selected. The three genetic operators, crossover, mutation and dropping condition are detailed in Section 4.4. In each generation, the number of new individuals evolved equals to the population size. Thus at this stage, the number of individuals is doubled. All individuals participate in a token competition and a replacement step, so as to eliminate similar rules and increase the diversity. These two steps are presented in Section 5.2. One half of the individuals with the higher fitness scores after token competition are passed to the next generation.

To estimate the fitness scores of individuals a data set is used in GGP. The data set should be partitioned into a training set and a testing set. Only the training set

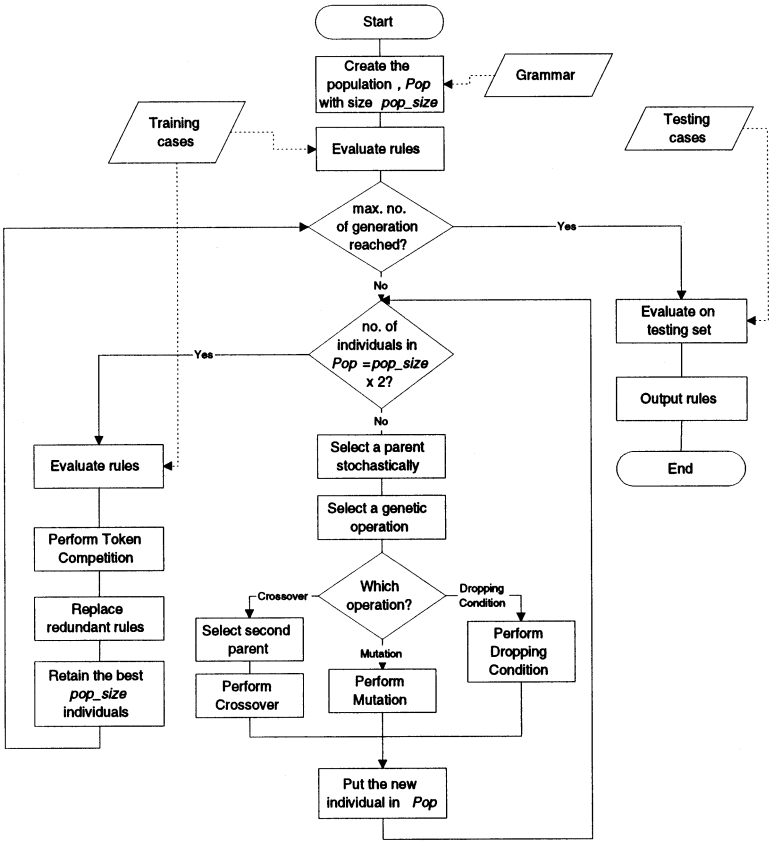

Fig. 4. The flowchart of the rule learning process.

is available for the learning process. After the maximum number of generations is reached, the discovered rules are further evaluated with the unseen testing set, so as to verify their accuracy and reject the rules that over-fit the training set. Our system uses 60% of the data for the training set and 40% for the testing set.

4.2. Grammar

The grammar specifies the rule structures to be evolved from GGP. The format of rules in each problem can be different. Thus, for each problem a specific grammar is written so that the format of the rules can best fit the domain. In general, the grammar specifies a rule is of the form 'if *antecedents* then *consequent*'. The antecedent part is a conjunction of attribute descriptors. The consequent part is an attribute descriptor as well. An attribute descriptor assigns a value to a nominal attribute, a range of values to a continuous attribute, or can be used to compare attribute values.

Table 1
An example grammar for rule learning

```
Rule → if Antes, then Consq.
Antes → Attr1 and Attr2 and Attr3
Attr1 → any|Attr1_descriptor
Attr2 → any|Attr2_descriptor
Attr3 → any|Attr3_descriptor
Attr1_descriptor → attr1=erc1
Attr2_descriptor → attr2 between erc2 erc2
Attr3_descriptor → attr3 Comparator Attr3_term
Comparator → |≠|<=|>=<|>
Consq → Attr4_descriptor
Attr4_descriptor → attr4=boolean_erc
```

For example, consider a database with four attributes. We want to learn rules about `attr4`, which is Boolean. The attribute `attrl` is nominal and coded with 0, 1 or 2. The attribute `attr2` is continuous between 0 and 200. The domain of `attr3` is similar to `attr2` and we want the rule to compare them. An example of the context free grammar for rule learning in given in Table 1. The symbols in cursive are the *non-terminals* and the other symbols are the *terminals*. A production rule of the form $\alpha \rightarrow \beta$ specifies that the non-terminals α can be expanded to β. $\alpha \rightarrow \beta|\gamma$ denotes$\{\alpha \rightarrow \beta,\ \alpha \rightarrow \gamma\}$. The symbols `ercl`, `erc2`, `erc3` and `boolean_erc` in this grammar are ephemeral random constants (ERCs). Each ERC has it own range for instantiation: `erc1` is within {0, 1, 2}, `erc2` and `erc3` are between 0 and 200, `boolean_erc` can only be T or F. In this grammar, the antecedent part consists of descriptors of all attributes. However, a rule does not need specifications for *all* attributes. The symbol 'any' is a generic descriptor that allows an attribute to be ignored in the rule. An attribute can be described either by its descriptor, or by 'any' such that it can be disregarded in the antecedent.

Table 2
An example derivation of rules

```
   Rule
⇒ if Antes, then Consq.

⇒ if Attr1 and Attr2 and Attr3, then Consq.

⇒ if Attr1_descriptor and Attr2_descriptor and Attr3_descriptor, then
   Attr4_descriptor.
⇒ if attr1=erc1 and attr2 between erc2 erc2 and attr3 Comparator Attr3_
   term, then attr4=boolean_erc.
⇒ if attr1=erc1 and attr2 between erc2 erc2 and attr3 ≠ erc3, then attr4=
   boolean_erc.
⇒ if attr1=0 and attr2 between 100 150 and attr3 ≠ 50, then attr4=T.
```

The grammar is used to derive rules to make up the initial population. The *start symbolic* the first symbol of the first line of the grammar. From the start symbol, a complete derivation is performed. Table 2 is an example of how a rule is derived from the grammar. This grammar allows the following rules:

- `if attr1=0 and attr2 between 100 and 150 and attr3` $\neq$ `50, then attr4=T.`
- `if attr1=1 and any attr3` $\geq$ `attr2, then attr4=F.`

GGP provides a powerful knowledge representation and allows a great fiexibility on the rule format. The representation of rules is not fixed but depends on the grammar. The descriptor is not restricted to compare attributes with values. Rather, the descriptors can be comparisons between attributes. Rules with other formats can be learned, provided that the suitable grammar is supplied. Moreover, rules with the user-desired structure can be learned because the user can specify the required rule format in the grammar.

4.3. Use of causality model and temporal order

The use of grammar can ensure syntactical correctness in the rule, but not semantic correctness. It is desirable to eliminate meaningless rules in the search process. This requires a certain degree of knowledge on the causality between the attributes. The causality and structure analysis steps in our data mining module can provide this knowledge. The Bayesian network may provide an overview of the relationships among the attributes. For example, if we know that attribute A is not related to any other attributes, then we do not need to learn rules about A. If we know attribute B should depend on attributes C and D, then we can specify a rule format like 'if $<$attribute C descriptor$>$ and $<$attribute D descriptor$>$, then $<$attribute B descriptor$>$'.

The temporal order among attributes can also provide knowledge to increase the learning efficiency. For example, in a medical domain, the rule 'if treatment is plaster, then diagnosis is radius fracture' is inappropriate. This rule does not make sense, because an operation is taken based on the treatment, not the other way round. In general, an event that occurs later will not be a cause of an event that occurred earlier! Thus, we can order the attributes according the temporal relationship. The grammar should be designed such that an attribute is not placed in the 'if' part if it occurs later then the attribute in the 'then' part. This temporal order can be represented easily using grammar. Both causality model and temporal order may significantly reduce search space and prune meaningless rules.

4.4. Genetic operators

The search space is explored by generating new rules using three genetic operators: crossover, mutation and a newly defined operator called dropping condition. A rule is composed of attribute descriptors. The genetic operators try to change the descriptors in order to search for better rules. Rank selection [13]

method is being used to select the parents. The probabilities of using crossover, mutation and dropping condition in our system are 0.5, 0.4 and 0.1, respectively.

Crossover is a sexual operation that produces one child from two parents. One parent is designated as the primary parent and the other one as the secondary parent. A part of the primary parent is selected and replaced by another part from the secondary parent. Suppose that the following primary and secondary parents are selected:

```
if attr1=0 and attr2 between 100 and 150 and attr3≠50, then
attr4=T.
if attr1=1 and any attr3≥attr2, then attr4=F.
```

The underlined parts are selected for crossover. The offspring will be

```
if attr1=0 and attr2 between 100 and 150 and attr3≥attr2, then
attr4=T.
```

The replaced part is selected randomly from the primary parent, hence genetic changes may occur either on the whole rule, on several descriptors, or on just one descriptor. The replacing part is also selected randomly, but under the constraint that the offspring produced must be valid according to the grammar. If a conjunction of descriptors is selected, it will be replaced by another conjunction of descriptors, but never by a single descriptor. If a descriptor is selected, then it can only be replaced by another descriptor of the same attribute. This can maintain the validity of the rule.

Mutation is an asexual operation. The genetic changes may occur on the whole rule, several descriptors, one descriptor, or the constants in the rule. A part in the parental rule is selected and replaced by a randomly generated part. The new part is generated by the same derivation mechanism using the same grammar. Similar to crossover, because the offspring have to be valid according to the grammar, a selected part can only mutate to another part with a compatible structure. For example, the parent

```
if attrl=0 and attr2 between 100 and 150 and attr3≠50, then
attr4=T.
```

may mutate to

```
if attrl=0 and attr2 between 100 and 150 and attr3=40, then
attr4=T.
```

Due to the probabilistic nature of GP, redundant constraints may be generated in the rule. For example, suppose that the actual knowledge is 'if $A < 20$ then $X = \mathrm{T}$'. We may learn rules like 'if $A < 20$ and $B < 20$, then $X = \mathrm{T}$'. Of course, this rule is correct, however, it does not completely represent the actual knowledge. Dropping condition is an operator designed to generalize the rules. The rule can be generalized if one descriptor in the antecedent part is dropped. Dropping condition selects randomly one attribute descriptor, and then turns it into 'any'. That particular attribute is no longer considered in the rule, hence the rule can be generalized.

The reproduction operator is not used in our approach. In conventional GP, an individual can exploit its genetic material through the use of the reproduction operator. Good individuals can reproduce themselves in the population and gradually dominate the population. However, in our system, we do not want a good rule

to replicate itself. Rather, we need to diversify the population in order to find several good rules. Hence reproduction is not used. Our system will only keep one copy for each good individual through token competition.

5. Novel techniques for rule learning

Other than using GGP as the search algorithm, other techniques are needed so as to efficiently learn multiple interesting rules from the database. These techniques are described in the following section.

5.1. Evaluation of rules

Completeness and consistency are conventionally used as the evaluation metric. However, a complete rule covering all the database records is unrealistic in a real-life situation. The support-confidence framework [1] is employed instead. *Support* measures the coverage of a rule. It is a ratio of the number of records covered by the rule to the total number of records. *Confidence factor* (*cf*) measures the consistency of a rule. It is the ratio of the number of records matching both the consequence and the antecedents to the number of records matching only the antecedents.

In the evaluation process, each rule is checked with every record in the training set. Three statistics are counted. The number *antes_hit* is the number of records matching the antecedents (the 'if' part), *consq_hit is* the number of records that match the consequent (the 'then' part), and *both_hit is* the number of records that obey the whole rule (both the 'if' and the 'then' parts).

The confidence factor *cf is* the fraction *both_hit*/*antes_hit*. However, a high confidence factor of a rule does not mean that the rule behaves significantly different from the average. Therefore, we need to consider the average probability of consequent *(prob)*. The value *prob is* equal to *consq_hit*/*total*, where *total* is the total number of records in the training set. This value measures the confidence for the consequence under no particular antecedent.

We defined *cf_part* as:

$$cf\_part = cf \times \log\left(\frac{cf}{prob}\right) \tag{5}$$

This value is based on two factors: *cf* and *cf*/*prob*. The log function measures the order of magnitude of the ratio *cf*/*prob*. *A* high value of *cf_part* requires the rule to have a high confidence *(cf)* and *cf is* higher than the average probability *(prob)*.

Support is another measure that we need to consider. A rule can have a high accuracy but may be found by chance and cover only a few training examples. These kind of rules do not have enough support. The value of *support* is defined as *both_hit*/*total*. If *support* is below a user-defined minimum, *min_support*, the confidence factor of the rule should not be considered.

We define our fitness function to be:

$$raw\_fitness = support, \quad \text{if} \quad support < min\_support \tag{6}$$

$$raw\_fitness = w_1 \times support + w_2 \times cf\_part, \quad \text{otherwise}$$

where the weights w_1 and w_2 are user-defined to control the balance between the confidence and the support. The values are set to 1 and 8, respectively so that the system prefers a rule with good confidence to a rule with good support.

5.2. Token competition

One important requirement of a rule learning system is to learn as many interesting rules as possible. This can be modeled as the search for multiple solutions. We follow the Michigan approach [20,2] where each individual represents one rule. The individuals in the population combined together represent a rule set. The token competition [31] technique is employed to achieve the *niching* [14] effect, so that good individuals in different niches are maintained in the population. Token competition has an advantage that it does not need to define and compare the similarity between individuals. It simply regards two individuals to be similar if they cover the same records.

In the natural environment, once an individual has found a good place for living, it will try to exploit this niche and prevent other newcomers to share the resources, unless the newcomer is stronger than it is. Hence, the other individuals are forced to explore and find their own niches. In this way, the diversity of the population is increased.

Based on this mechanism, we assume that each record in the training set can provide a resource called token. If a rule can match a record, it will set a flag to indicate that the token is seized. As a result, other weaker rules cannot get the token. The priority of receiving tokens is determined by the strength of the rules. A rule with a high score on *raw_fitness* can exploit the niche by seizing as many tokens as it can. The other rules entering the same niche will have their strength decreased because they cannot compete with the stronger rule. The fitness score of each individual is modified based on the token it can seize. The modified fitness is defined as:

$$\text{modified\_fitness} = raw\_fitness \times count/ideal \tag{7}$$

where *count* is the number of tokens that the rule actually seized, and *ideal is* the maximum number of tokens that it can seize, which is equal to the number of records that the rule matches.

As a result of token competition, there are rules that did not seize any token. These rules are redundant as all of its records are already covered by the stronger rules. They can be replaced by new individuals. Introducing these new individuals can inject a larger degree of diversity into the population, and provide extra chances for generating good rules.

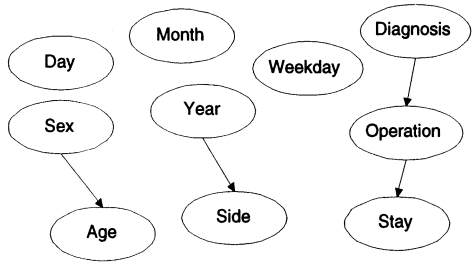

Fig. 5. The best network structure for the fracture database.

6. Results on the fracture database

The described data mining technology has been applied to a real-life medical database consisting of children with limb fractures, admitted to the Prince of Wales Hospital of Hong Kong during the period 1984–1996. This data can provide information for the analysis of child fracture patterns. This database has 6500 records and eight attributes, which are listed in Table 3.

6.1. Results of causality and structure analysis

The relationships among the attributes are analyzed by learning a Bayesian network. We have used a typical population size of 50 to run for 100 generations. The execution time was 45 min on a Sun Ultra 1/140. The best network structure is drawn in Fig. 5. Day, month, weekday and year refer to different parts of the admission date.

Table 3
Attributes in the fracture database

Name	Type	Description	Possible value
Sex	Nominal	Sex	'M' or 'F'
Age	Numeric	Age	Between 0 and 16 years old
Admday	Date	Admission date	Between 1984 and 1996; Divided into four parts: day, month, year and weekday
Stay	Numeric	Length of staying in hospital	Between 0 and 1000 days. Discretized into 18 non-uniform ranges.
Diagnosis	Nominal	Diagnosis of fracture	10 different values, based on the location of fracture
Operation	Nominal	Operation	'CR' (Simple closed reduction), 'CR + K-wire' (closed reduction with K-wire), 'CR + POP' (closed reduction with POP), 'OR' (open reduction) or Null (no operation)
Surgeon	Nominal	Surgeon	One of 61 surgeons or Null if no operation
Side	Nominal	Side of fracture	'Left', 'Right', 'Both' or 'Missing'

Table 4
Summary of the rules for the fracture database

About	No. rules	*cf* (%)			*cf*/*prob*			*support* (%)		
		mean	max	min	mean	max	min	mean	max	min
Diagnosis	2	45.6	51.4	39.8	1.6	1.7	1.4	9.2	10.0	8.4
Operation	8	42.6	74.0	28.0	2.0	2.9	1.1	5.4	16.2	3.2
Stay	7	71.1	81.1	47.0	2.5	7.0	1.4	4.5	8.7	3.1

This network shows three chains of causalities. The first chain shows that the length of staying in hospital depends on the operation, the operation in turn depends on the diagnosis. Another edge in the network is between sex and age. Although by common sense, sex should not be the cause of age, further analysis showed that in the database, the age is correlated with sex. Female patients are more likely to be in the younger age group (age 0–7), and male patients are more likely to be in the elder age group (11–15). There is another edge between year and side and this result is quite surprising. The conditional probabilities are investigated and two interesting points are revealed.

- The probabilities for the side equal to 'both' are exceptionally low for the years 1984, 1988 and 1992.
- The probabilities for the side equal to 'missing' are high for year 1995 and 1996, while for other years, these probabilities are low.

This phenomenon cannot be explained reasonably. We suspected that different notations are used in recording the side for different years.

6.2. Results of rule learning

Based on the learned Bayesian network, we observed a causality model between diagnosis, operation and stay and we wished to learn about these attributes. In addition, the temporal order gives extra knowledge on how the rules should be formulated. The attributes can be divided into three time stages: a diagnosis is first given to the patient, then an operation is performed, and after that the patient stays in the hospital. This knowledge leads to three causality models. Firstly, sex, age and admission date are the possible causes of diagnosis. Secondly, these three attributes and diagnosis are the possible causes of operation and surgeon. Thirdly, length of stay has all other attributes as the possible causes. A grammar (Appendix A) is written as a template for these three kind of rules. We have used a population size of 300 to run for 50 generations in the rule learning step. The execution time was ~ 3 h on a Sun Ultra 1/140 for the 6500 records. The results are listed in Table 4.

Two interesting rules regarding the diagnosis are found. The one with the highest confidence is:

```
If age is between 2 and 5, then diagnosis is Humerus. (cf=
51.436%)
```

The confidences of the rules about diagnosis are $\sim 40-50\%$. This is partly because there are no actual strong rules affecting the value of diagnosis. However the ratio *cf*/*prob* shows that the patterns discovered deviated significantly from the average. We found that humerus fractures are the most common fracture for children between 2 and 5 years old and radius fractures are the most common fractures for boys between 11 and 13.

Eight interesting rules about operation are found. The one with the highest confidence is:

```
If age is between 0 and 7, and admission year is between 1988 and
1993, and diagnosis is Radius, then operation is CR+POP. (cf=
74.05%)
```

These rules suggest that radius and ulna fractures are usually treated with CR + POP (i.e. plaster). An operation is usually not necessary for a tibia fracture. Open reductions are more common for children >11 years of age, while younger children (<7 years of age) have a higher chance of not needing operations. We did not find any interesting rules about surgeons, as the surgeons for operation are more or less randomly distributed in the database.

Seven interesting rules about length of stay are found. The one with the highest confidence is:

```
If admission year is between 1985 and 1996, and diagnosis is
Femur, then stay is more than 8 days. (cf=81.11%)
```

The rules about the length of stay suggest that femur and tibia fractures are serious injuries with patients having to stay longer in hospital. If open reduction is used, the patient requires more time to recover because the wound has been cut open for operation. If no operation is needed, it is likely that the patient can return home within 1 day. Normally, radius fractures require a shorter recovery time.

The results have been evaluated by the medical experts. The causality model matches with general knowledge. The doctor decides a treatment based on the type of fracture, and the treatment affects the recovery. Previous analyses on fracture patterns only gave an overall injury pattern. Our system automatically uncovered relationships between different attribute values. The rules provide interesting patterns that were not recognized before now. The analysis gives an overview of the important epidemiological and demographic data of the fractures in children. It clearly demonstrated the treatment pattern and rules of decision making. It can provide a good monitor of the change of pattern of management and the epidemiology if the data mining process is continued longitudinally over the years. It also helps to provide the information for setting up a knowledge-based instruction system to help young doctors in training to learn the rules in diagnosis and treatment.

7. Results on the scoliosis database

The data mining process has been applied to the database of scoliosis patients. Scoliosis refers to the spinal deformation, where a patient suffering from this has

Table 5
Attributes in the scoliosis database[a,b,c]

Name	Explanation	Possible value
Sex	Sex	'M' or 'F'
Age	Age	positive integer
Lax	Joint Laxity	integer between 0 and 3
lstCurveT1	Whether 1st curve started at vertebra T1	Y or N
lstMCGreater	Whether the degree of 1st Major Curve is greater the the 2nd Major Curve	Y or N
L4Tilt	Whether vertebra L4 is tilted	Y or N
lstMCDeg	Degree of 1st Major Curve	positive integer
2ndtMCDeg	Degree of 2nd Major Curve	positive integer
lstMCApex	Apex of 1st Major Curve	any vertebra
2ndMCApex	Apex of 2nd Major Curve	any vertebra
Degl	Degree of 1st Curve	positive integer
Deg2	Degree of 2nd Curve	positive integer
Deg3	Degree of 3rd Curve	positive integer
Deg4	Degree of 4th Curve	positive integer
Class	Scoliosis Classification	K-I, K-II, K-III, K-V, TL, L
Mens	Period of Menstruation	positive integer
TSI	Trunk Shift (in cm)	positive integer
TSIDir	Trunk Shift Direction	null, left or right
RI	Risser Sign	integer between 0 and 5
Treatment	Treatment	Observation, surgery or bracing

[a] Vertebras are coded with Tl-T12 or L1-L5.
[b] Trunk shift measures the displacement of the curve.
[c] Risser sign measures the maturity of the patient.

one or several curves in his spine. Among them, the curves with severe deformations are identified as major curves. The database stores measurements on the patients, such as the number of curves, curve location, degrees and directions. It also records the maturity of the patient, the class of scoliosis and the treatment. The database has 500 records. According to the domain expert, 19 attributes are useful and extracted from the database in the preprocessing step (Table 5).

7.1. Results of causality and structure analysis

We have used a population size of 50 and a maximum number of generations of 1000 to run in the causality and structure analysis. The execution time was 3 min on a Sun Ultra 1/140. The best Bayesian network structure is shown in Fig. 6. The right part of the network shows that sex implies menstruation, and menstruation implies age, and age in turn implies `RI`. The network also shows that `TSIDir` can imply `TSI` because if `TSI` direction is null, `TSI` should be zero.

The main part of the network shows that 2ndMCDeg can imply 2ndMCApex and lstMCGreater. This is because if 2ndMCDeg = 0, the patient does not have the second major curve, and thus 2ndMCApex must be zero and the first major curve must be the greater curve. The value of 2ndMCDeg also implies the degree of the second curve (Deg2), because if the patient has two major curves, most of the time the second major curve is the second curve. The value of lstMCDeg is affected by lstMCGreater and Deg2. When the degree of first major curve is greater than the second curve, lstMCDeg is most likely large. When Deg2 is large, the first major curve is most likely going to be the second curve. The value of lstMCDeg can imply Deg1 because when the value of lstMCDeg is small, the degree first curve is not large. Deg2 can imply the value of L4Tilt and Deg3, while Deg3 can imply lstCurveT1. If the degree of the second curve is large, then usually L4 is tilt. If the patient does not have the second curve, then he will not have the third curve. Moreover, if he has at least three curves, then most of the time, the deformation will start at the first vertebra T1. The network also shows that the value of treatment mainly depends on lstMCDeg. On the other hand, Class depends on Deg2.

7.2. Results of rule learning

The medical experts are interested to discover more about the classification and treatment of scoliosis. Scoliosis can be classified as Kings, Thoracolumbar (TL) and Lumbar (L), while Kings can be further subdivided into K-I, II, III, IV and V. Treatment can be observation, surgery and bracing. The determinations of these two attributes are complicated. Although the induced Bayesian network provides valid and useful relationships, the domain expert is more interested in finding relationships between classification and the attributes lstCurveTl, lstMC-Greater, L4Tilt, lstMCDeg, 2ndMCDeg, lstMCApex and 2ndMCApex, and relationships between treatment and age, laxity, degrees of the curves, maturity of

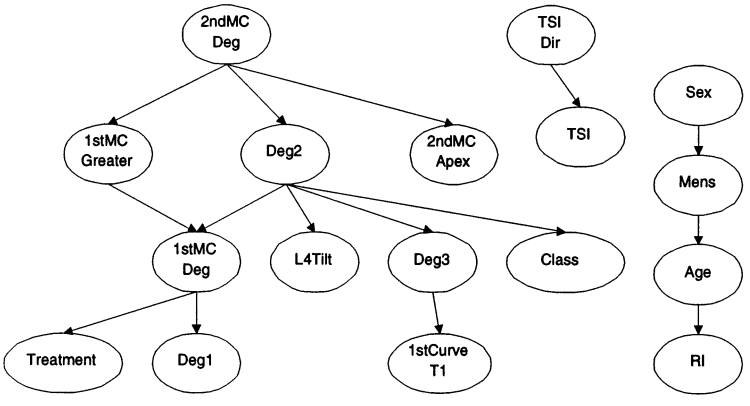

Fig. 6. The best network structure for the scoliosis database.

Table 6
Results of the rules for scoliosis classification

Class	No. Rules	*cf* (%)			*support* (%)			*prob* (%)
		mean	max	min	mean	max	min	
King-I	5	94.84	100	90.48	5.67	10.73	0.86	28.33
King-II	5	80.93	100	52.17	6.61	14.38	1.07	35.41
King-III	4	23.58	25.87	16.90	1.56	2.58	0.86	7.94
King-IV	2	24.38	29.41	19.35	1.18	1.29	1.07	2.79
King-V	5	54.13	62.50	45.45	0.97	1.07	0.86	6.44
TL	1	41.18	41.18	41.18	1.50	1.50	1.50	2.15
L	3	54.04	62.50	45.45	2.00	2.79	1.07	4.51

the patient, displacement of the vertebra and the class of scoliosis. This domain knowledge can easily be incorporated in the design of the rule grammar. There are two types of rules, one for the classification of scoliosis and the other for suggesting treatment. The grammar is outlined in Appendix B.

The population size used in the rule learning step is 100 and the maximum number of generations is 50. The execution time was $\sim$1 h on a Sun Ultra 1/140. The results of rule learning from this database are listed below.

1. Rules for scoliosis classification.

An example of this kind of rules is:

```
if lstMCGreater=N and lstMCApex=T1-T8 and 2ndMCApex=L3-L4,
then King-I. (cf=l00%)
```

For each class of scoliosis, a number of rules are mined. The results are summarized in Table 6. For King-I and II, the rules have high confidence and generally match with the knowledge of medical experts. However, there is one unexpected rule for the classification of King-II. Under the conditions specified in the antecedents, our system found a rule with a confidence factor of 52% that was classified under King-II. However, the domain expert suggests that the class should be King-V! After analysing the database, we revealed that serious data errors existed in the current database and that some records contained an incorrect scoliosis classification.

For King-III and IV, the confidence of the rules discovered is $\sim$20%. According to the domain expert, one common characteristic for these two classes is that there is only one major curve or that the second major curve is insignificant. However, there is no rigid definition for a 'major curve' and the concept of 'insignificant' is fuzzy. These all depend on a doctor's interpretation. Due to the lack of this important information, the system is unable to find accurate rules for these two classes. Another problem is that only a small number of patients in the database where classified according to King-III or IV (see *prob* in Table 6). The database cannot provide a large number of cases for training. Similar problems also existed for King-V, TL and L.

For the class King-V, TL and L, the system found rules with confidence around 40% to 60%. Nevertheless, the rules for TL and L show something different in

comparison with the rules suggested by the clinicians. According to our rules, the classification always depends on the *first major curve*, while according to the domain expert, the classification depends on the *larger major curve*. After discussion with the domain expert, it is agreed that the existing rules are not defined clearly enough, and our rules are more accurate than them. Our rules provide hints to the clinicians to re-formulate their concepts.

2. Rules about treatment.

A typical rule of this kind is:

```
If age=2-12 and Deg1=20-26 and Deg2=24-47 and Deg3=27-52 and
Deg4=0,then Bracing. (cf=100%)
```

The results are summarized in Table 7. The rules for observation and bracing have very high confidence factors. However, the support is not high, showing that the rules cover only fragments of the cases. Our system prefers accurate rules to general rules. If the user prefers more general rules, the weights in the fitness function can be tuned. For surgery, no interesting rule was found because only 3.65% of the patients were treated with surgery.

The biggest impact on the clinicians from the data mining analysis of the scoliosis database is the fact that many rules set out in the clinical practice are not clearly defined. The usual clinical interpretation depends on the subjective experience. Data mining revealed quite a number of mismatches in the classification on the types of Kings curves. After a careful review by the senior surgeon, it appears that the database entries by junior surgeons may be inaccurate and that the data mining rules discovered are in fact more accurate! The classification rules must therefore be quantified. As a result, the rules discovered can help in the training of younger doctors and act as an intelligent means to validate and evaluate the accuracy of the clinical database. An accurate and validated clinical database is very important for helping clinicians to make decisions, to assess and evaluate treatment strategies, to conduct clinical and related basic research, and to enhance teaching and professional training.

8. Conclusion

We have presented a data mining system that is composed of five steps. The third and fourth steps are detailed. They both employ evolutionary computation as the search algorithms. Causality and structure analysis focuses on the general causality

Table 7
Results of the rules about treatment

Type	No. Rules	*cf* (%)			*support* (%)			*prob* (%)
		mean	max	min	mean	max	min	
Observation	4	98.89	100	95.55	3.49	6.01	1.07	62.45
Bracing	5	79.57	100	71.43	1.03	1.29	0.86	24.46
Surgery	0	–	–	–	–	–	–	3.65

model between the *variables* while rule learning captures the specific behavior between particular *values* of the variables.

Our system is particularly suitable to the analysis of real-life databases that cannot be described completely by just a few rules. Building a complete model for such a database is difficult and usually results in a complicated model. We have used a Bayesian network to give a causality model. The Bayesian network is easy to understand while it has a well-developed mathematical model. Moreover, in many real-life situations, the available rules are general guidelines with many exceptional cases. The rule learning step aims to learn such kinds of knowledge. It compares the confidence of the rule with the average probability and a search for patterns significantly deviated from the normal. Token competition is used so as to learn as many rules as possible. Furthermore, knowledge from domain experts can be very useful to data mining. The use of grammar allows the domain knowledge to be easily and effectively utilized. On one hand, grammar can prune the search space with meaningless rules, while on the other hand, it can ensure that the output knowledge is in the user desired format.

The system has been applied to two real-life medical databases. The results can provide interesting knowledge as well as suggest refinements to the existing knowledge. We also found unexpected results that led to the discovery of errors in the database. In the fracture database, the system automatically uncovered knowledge about the age effect on fracture, the relationship between diagnoses and operations, and the effect of diagnoses and operations on the length of stay in the hospital. In the scoliosis database, we have discovered new information regarding the classification and treatment of scoliosis. This discovered knowledge will lead to a refinement of existing knowledge.

Acknowledgements

This work was partially supported by Hong Kong RGC CERG Grant CUHK 4161/97E and CUHK Engineering Faculty Direct Grant 2050151. The authors wish to thank Chun Sau Lau and King Sau Lee for preparing, analyzing and implementing the rule learning system for the scoliosis database.

Appendix A. The grammar for the fracture database

This grammar is not completely listed. The grammar for the other attribute descriptors is similar to the part of the grammar in lines 11–19.

```
1: Rule → Rule1 |Rule2| Rule3
2: Rule1 → if Antesl, then Consq1.
3: Rule2 → if Antes1 and Antes2, then Consq2.
4: Rule3 → if Antes1 and Antes2 and Antes3, then Consq2.
```

```
5: Antes1 → Sex1 and Age1 and Admday1
6: Antes2 → Diagnosis1
7: Antes3 → Operation1 and Surgeon1
8: Consq1 → Diagnosis_descriptor
9: Consq2 → Operation_descriptor | Surgeon_desriptor
10: Consq3 → Stay_descriptor
11: Sex1 → any | Sex_descriptor
12: Sex_descriptor → sex=sex_const
13: Admday1 → any | Admday_descriptor
14: Admday_descriptor → admday_day between day_const
day_const
15: Admday_descriptor → admday month between month_const
month_const
16: Admday_descriptor → admday_year between year_const
year_const
17: Admday_descriptor → admday weekday between weekday_const
weekday_const
18: Diagnosisl → any | Diagnosis_descriptor
19: Diagnosis_descriptor → diagnosis is diagnosis_const
```

Appendix B. The grammar for the scoliosis database

This grammar is not completely listed. The grammar for the other attribute descriptors is similar to the part of the grammar in lines 7–12.

```
1: Rule → Rule1 | Rule2
2: Rule1 → if Antes1, then Consq1.
3: Rule2 → if Antes2, then Consq2.
4: Antes1 → 1stCurveT1 lstMCGreater and L4Tilt and lstMCDeg
  and 2ndMCDeg and lstMCApex and 2ndMCApex
5: Antes2 → Age and Lax and Deg1 and Deg2 and Deg3 and Deg4 and
  Mens and RI and TSI and ScoliosisType
6: Consq1 → ScoliosisType_descriptor
7: 1stMCGreater → any | lstMCGreater_descriptor
8: 1stMCGreater_descriptor → lstMCGreater=boolean_const
9: 1stMCDeg → any | lstMCDeg_descriptor
10: 1stMCDeg_descriptor → lstMCDeg between deg_const
  deg_const
11: 1stMCApex → any | lstMCApex_descriptor
12: 1stMCApex_descriptor → 1stMCApex between Apex_const
  Apex_const
```

References

[1] Agrawal R., Imielinski T., Swami A. Mining association rules between sets of items in large databases. Proceedings of the 1993 International Conference on Management of Data (SIGMOD 93), 1993:207–216.

[2] Booker L, Goldberg DE, Holland JH. Classifier systems and genetic algorithms. Artif Intell 1989;40:235–82.

[3] R.R. Bouckaert. Properties of belief networks learning algorithms. Proceedings of the Conference on Uncertainty in Artificial Intelligence, 1994:102–109.

[4] Charniak E. Bayesian networks without tears. AI Mag 1991;12(4):50–63.

[5] Chow CK, Liu CN. Approximating discrete probability distributions with dependence trees. IEEE Trans Inf Theory 1968;14(3):462–7.

[6] Clark P, Niblett T. The CN2 induction algorithm. Mach Learn 1989;3:261–83.

[7] Cooper GF. The computational complexity of probabilistic inference using Bayesian belief networks. Artif Intell 1990;42:393–405.

[8] Cooper GF, Herskovits E. A Bayesian method for the induction of probabilistic networks from data. Mach Learn 1992;9:309–47.

[9] Fayyad U.M., Piatesky-Shapiro G., Smyth P. From data mining to knowledge discovery: An overview. AI Mag., 1996:37–51.

[10] Fogel DB. An introduction to simulated evolutionary optimization. IEEE Trans Neural Netw 1994;5:3–14.

[11] Fogel L., Owens A., Walsh M. Artificial Intelligence through Simulated Evolution. New York: Wiley, 1966.

[12] Giordana A, Neri F. Search-intensive concept induction. Evol Comput 1995;3:375–416.

[13] Goldberg D.E. Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley, Reading, MA, 1989.

[14] Goldberg D.E., Richardson J. Genetic algorithms with sharing for multimodal function optimization. Proceedings of the second International Conference on Genetic Algorithms, 1987:41–49.

[15] Heckerman D. Bayesian Networks for Knowledge Discovery, chapter 11. Cambridge, MA: MIT Press, 1996:273–306.

[16] Heckerman D, Geiger D, Chickering DM. Learning Bayesian networks: the combination of knowledge and statistical data. Mach Learn 1995;20(3):197–243.

[17] Heckerman D, Wellman MP. Bayesian networks. Commun ACM 1995;38(3):27–30.

[18] Herskovits E., Cooper G. KUTATO: An entropy-driven system for construction of probabilistic expert systems from databases. Technical Report KSL-90-22, Knowledge Systems Laboratory, Medical Computer Science, Stanford University, 1990.

[19] Holland J.H. Adaptation in Natural and Artificial Systems. Cambridge, MA: MIT Press, 1992.

[20] Holland J.H., Reitman J.S. Cognitive systems based on adaptive algorithms. In: Waterman D.A., Hayes-Roth F., editors. Pattern-Directed Inference Systems. New York: Academic Press, 1978.

[21] J. E. Hopcroft, J. D. Ullman. Introduction to automata theory, languages, and computation. Reading, MA: Addison-Wesley, 1979.

[22] Janikow CZ. A knowledge-intensive genetic algorithm for supervised learning. Mach Learn 1993;13:189–228.

[23] De Jong KA, Spaers WM, Gordon DF. Using genetic algorithms for concept learning. Mach Learn 1993;13:161–88.

[24] Koza J.R. Genetic Programming: on the programming of computers by means of natural selection. Cambridge, MA: MIT Press, 1992.

[25] Koza J.R. Genetic Programming II: automatic discovery of reusable programs. Cambridge, MA: MIT Press, 1994.

[26] Lam W. Bayesian network refinement via machine learning approach. IEEE Trans. Pattern Anal Mach. Intell., 1998.

[27] Lam W, Bacchus F. Learning Bayesian belief networks—an approach based on the MDL principle. Comput Intell 1994;10(3):269–93.

[28] Lam W., Wong M.L. Leung M.S., Ngan P.S. Discovering probabilistic knowledge from databases using evolutionary computation and minimum description length principle. Genetic Programming: Proceedings of the Third Annual Conference, 1998.
[29] Larranaga P, Poza M, Yurramendi Y, Murga R, Kuijpers C. Structure learning of Bayesian network by genetic algorithms: A performance analysis of control parameters. IEEE Trans Pattern Anal Mach Intell 1996;18(9):9.
[30] Larranage P, Kuijpers C, Murga R, Yurramendi Y. Learning Bayesian network structures by searching for the best ordering with genetic algorithms. IEEE Trans Syst Man Cybern Part A: Syst Humans 1996;26(4):487–93.
[31] Leung K.S., Leung Y., So L., Yam K.F. Rule learning in expert systems using genetic algorithm: 1, concepts. Proceedings of the Second International Conference on Fuzzy Logic and Neural Networks, Iizuka, Japan, 1992: 201–204.
[32] Michalski R.S., Mozetic I., Hong J., Lavrac N. The multi-purpose incremental learning system AQ15 and its testing application to three medical domains. Proceedings of the Fifth National Conference on Artificial Intelligence, 1986:1041–1045.
[33] Rebane G., Pearl J. The recovery of causal poly-trees from statistical data. Uncertainty in Artificial Intelligence 3. Amsterdam: North-Holland, 1989:175–182.
[34] Rechenberg I. Evolution Strategy: Optimization of technical systems by means of biological evolution. Stuttgart: Fromman-Holzboog, 1973.
[35] Schwefel H.P. Numerical Optimization of Computer Models. New York: Wiley, 1981.
[36] Singh M., Valtorta M. An algorithm for the construction of Bayesian network structures from data. Proceedings of the Conference on Uncertainty in Artificial Intelligence, 1993:259–265.
[37] Smith S.F. A Learning System based on Genetic Adaptive Algorithms. PhD thesis, University of Pittsburgh, 1980.
[38] Smith S.F. Flexible learning of problem solving heuristics through adaptive search. Proceedings of the Eighth International Conference On Artificial Intelligence. San Mateo, CA: Morgan Kaufmann, 1983.
[39] Spirtes P., Glymour C., Scheines R. Causation, Prediction and Search. Berlin: Springer, 1993.
[40] Wong M.L. Evolutionary program induction directed by logic grammars. PhD thesis, The Chinese University of Hong Kong, 1995.
[41] Wong M.L., Lam W., Leung K.S. Using evolutionary computation and minimum description length principle for data mining of probabilistic knowledge, submitted to IEEE Trans. Pattern Anal. Mach. Intell., 1997.
[42] Wong ML, Leung KS. Inducing logic programs with genetic algorithms: the genetic logic programming system. IEEE Expert 1995;10(5):68–76.
[43] Wong ML, Leung KS. Evolutionary program induction directed by logic grammars. Evol Comput 1997;5:143–80.