KYBERNETIKA — VOLUME *40* (2004), NUMBER 3, PAGES 333 – 348

BUILDING ADAPTIVE TESTS USING BAYESIAN NETWORKS[1]

Jiří Vomlel

We propose a framework for building decision strategies using Bayesian network models and discuss its application to adaptive testing. Dynamic programming and $AO^\star$ algorithm are used to find optimal adaptive tests. The proposed $AO^\star$ algorithm is based on a new admissible heuristic function.

Keywords: Bayesian networks, adaptive testing, heuristic search

AMS Subject classification: 68T37

1. BAYESIAN NETWORKS

Bayesian networks are probabilistic graphical models that are capable of modelling domains comprising uncertainty. They were introduced to the field of expert systems by Pearl [17] and Spiegelhalter and R. P. Knill-Jones [21]. The first applications were an expert system for electromyography Munin [2] and the Pathfinder system [8]. Since then Bayesian networks were successfully applied in several areas. Strength of graphical models is not only that they enable efficient uncertainty reasoning with hundreds of variables (e. g. using the method of Lauritzen and Spiegelhalter [13]), but also they help humans to understand better the modelled domain. This is mainly due to their comprehensible representation by use of directed acyclic graphs representing dependencies between domain variables. See [14] where some recent applications of Bayesian networks are discussed.

Bayesian network consists of an directed acyclic graph (DAG) $G = (V, E)$, to each node $i \in V$ corresponds one random variable X_i with a finite set $\mathbb{X}_i$ of mutually exclusive states and a conditional probability table (CPT) $P(X_i \,|\, (X_j)_{j \in pa(i)})$, where $pa(i)$ denotes the set of parents of node i in graph G. See Figure 1 for an example of Bayesian network.

Bayesian network encodes qualitative and quantitative knowledge. Quantitative knowledge is represented by CPTs, while qualitative is encoded by use of a DAG. The DAG implies certain conditional independence relations between variables $(X_i)_{i \in V}$.

[1]This work was supported by the Grant Agency of the Czech Republic through grant No. 201/02/1269.

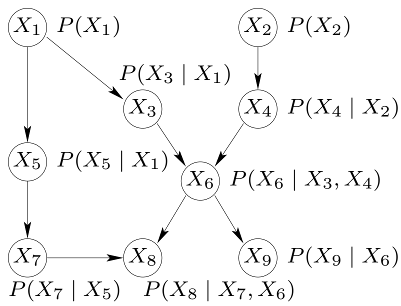

Fig. 1. An example of a Bayesian network.

A concept called *d-separation*, introduced by Pearl [19], can be used to read the conditional independence statements from a DAG.

In order to define *d-separation* we need few graphical concepts. A *path* in an undirected graph H is a sequence of nodes $\{n_1, \ldots, n_k\}$ such that $(n_i, n_{i+1}), i = 1, \ldots, k-1$ are edges in graph H and for $i, j = 1, \ldots, k$ and $i \neq j$ it holds that $n_i \neq n_j$. A *trail* in an oriented graph G is a sequence of nodes that forms a path in the undirected version of G, i. e. when the directions of arrows are ignored. We say that edges meet *head-to-head* in a node n_i of a trail $\{n_1, \ldots, n_k\}$ if there are directed edges $n_{i-1} \rightarrow n_i$ and $n_i \leftarrow n_{i+1}$ in the graph G.

Two different variables X_i and X_j are *d-separated* by $\mathcal{Y}$ if, for all trails between nodes i and j there is an intermediate node k such that edges either

- do not meet head-to-head in k and $X_k \in \mathcal{Y}$, or
- meet head-to-head in k and neither X_k nor any of its descendants belongs to $\mathcal{Y}$.

X_i, X_j are *conditionally independent* given a set of variables $\mathcal{Y}$ if $P(X_i \,|\, \mathcal{Y}) = P(X_i \,|\, \mathcal{Y}, X_j)$. It is required that all variables X_i, X_j d-separated by a set $\mathcal{Y}$ are conditionally independent given $\mathcal{Y}$ in a probability distribution P represented by the Bayesian network model. It is not difficult to show that the joint probability distribution $P(X_i)_{i \in V})$ satisfying the above property and having its CPTs equal to $P(X_i \,|\, (X_j)_{j \in pa(i)}), i \in V$ is unique and equals to the product of the CPTs, i. e.

$$P((X_i)_{i \in V}) \quad = \quad \prod_{i \in V} P(X_i \,|\, (X_j)_{j \in pa(i)}) \,.$$

The representation of the joint probability distribution by a Bayesian network has several advantages. One of the most important properties is that it allows computationally efficient reasoning with new evidence.

For a detailed introduction to Bayesian networks we refer to Jensen [10].

2. BUILDING STRATEGIES USING PROBABILISTIC MODELS

A *strategy* describes steps that the user should perform in order to achieve a required goal. For example, a step can be: the user performs an action, the user makes an observation, or the user answers a question. Since outcomes of steps are uncertain each strategy must specify the next step the user should do for all possible combinations of outcomes of previous steps. Thus a strategy can be represented by a directed tree. It is convenient to define two types of nodes in the tree – *chance nodes* and *terminal nodes*. Each chance node corresponds to a single step of a strategy. Terminal nodes are leaves of the tree where the strategy terminates. One session corresponds to a path in the tree, i. e. to a sequence of steps, starting at the root of the tree and ending at a terminal node. In Figure 2 we present an example of an adaptive test consisting of two questions. Ovals denote chance nodes. Diamonds denote terminal nodes. Each chance node is labelled by the corresponding step. Every edge coming out from a chance node is labelled by an outcome of the step corresponding to that node. The strategy represented by the tree is: *If the answer to the first question X_2 is correct then the second question is X_3 otherwise the second question is X_1.*

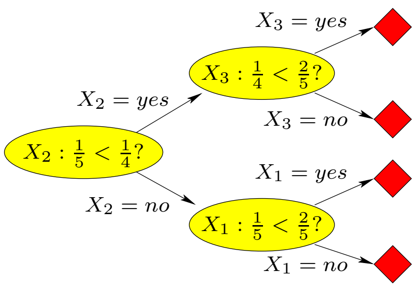

Fig. 2. Example of a strategy.

The space of all possible strategies can be represented by an AND/OR-graph [18], where AND nodes correspond to chance nodes and OR nodes to decision nodes. See Figure 3 for an example of the space of all possible test strategies for an adaptive test consisting of two questions when the bank of all possible questions $\mathcal{X}$ contains three questions. One strategy is highlighted. It starts with question X_2 and the second question is either X_3 or X_1 – selected according to the answer of X_2.

Let $\mathcal{S}$ denote the set of all strategies admissible for a given problem and $\mathcal{L}(\mathbf{s})$ denote the set of all terminal nodes of a strategy $\mathbf{s} \in \mathcal{S}$. An evaluation function $f : \cup_{\mathbf{s}\in\mathcal{S}}\mathcal{L}(\mathbf{s}) \mapsto \mathbb{R}$ is defined. The goal is to minimise this function at the end of a session. The outcomes of the steps proposed in a strategy $\mathbf{s}$ are unknown, only the probabilities $P(\mathbf{e}_\ell)$ of terminating in a node $\ell \in \mathcal{L}$ can be computed from the domain model represented by a Bayesian network. The expected value of the

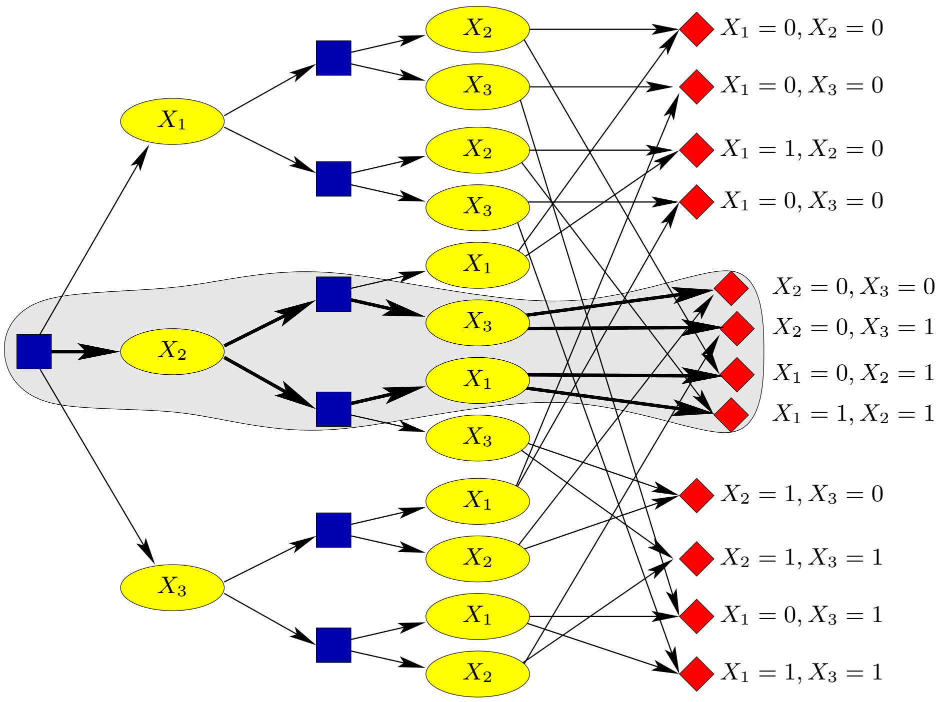

Fig. 3. The space of all possible test strategies consisting of two questions selected from a bank of questions containing three questions. One admissible test strategy is highlighted.

evaluation function is defined for each strategy $\mathbf{s} \in \mathcal{S}$ as

$$E_f(\mathbf{s}) \quad = \quad \sum_{\ell \in \mathcal{L}(\mathbf{s})} P(\mathbf{e}_\ell) \cdot f(\mathbf{e}_\ell)\,. \tag{1}$$

We search for a strategy $\mathbf{s}^\star \in \mathcal{S}$ minimising the value of $E_f(\mathbf{s})$ from all $\mathbf{s} \in \mathcal{S}$.

In this paper we apply this framework to adaptive testing. In Section 3 we describe a Bayesian network model that is used to model students solving a test. In Section 4 we discuss methods that can be used to design an adaptive test using a Bayesian network model. We propose an admissible heuristic function that can be used within an $AO^\star$ algorithm. We prove admissibility of the proposed heuristic function. The paper is concluded by an experiment that illustrate presented results.

3. ADAPTIVE TESTING

In this section we present an example of an adaptive test that was used to diagnose person's skills. The adaptive test uses Bayesian networks to model a tested student and the given questions.

Tests that are automatically tailored to the level of the individual examines are called *adaptive tests*. After each response on a question the system selects next question based on the answers of the previous questions. A simple example of adaptive test was presented in Figure 2. Since this approach requires computers for the test administration it is often referred to as *computerised adaptive testing* (CAT) [26, 23].

Almond and Mislevy [1] proposed to use graphical models for CAT. Their model consists of one *student model* and several *evidence models*, one for each task or question. Typically, a test designer specifies the tested skills $\mathcal{Y} = \{Y_1, \ldots, Y_k\}$ and a bank of questions $\mathcal{X} = \{X_1, \ldots, X_m\}$. We will use Y to denote multivariable $(Y_1, \ldots, Y_k)$ and $y = (y_1, \ldots, y_k)$ will denote a state of Y. The student model describes relations between student's skills, abilities, misconceptions. The knowledge about a student is expressed by use of a joint probability distribution $P(Y) = P(Y_1, \ldots, P_k)$, which is represented by a Bayesian network.

Classical approach used in educational and psychological testing since 1960's is item response theory (IRT) [15, 20]. Within this method the student is modelled by a single variable Θ. These models are suitable when the task is to grade students, but their application is problematic if more information about the tested student is required. In multidimensional IRT several variables are used to represent a student. The presented application of Bayesian network can be regarded as a generalisation of the multidimensional IRT. It brings two basic advantages:

- It can better reflect the student reasoning process and provides better insight into the modelled problem.
- The student model encode dependence between skills. Therefore adaptive tests can be substantially shortened while the test precision is kept.

Next we will briefly describe the learning process of a Bayesian network model used for testing basic operations with fractions [24].

First, a group of students from Aalborg University prepared paper tests that were given students of Brønderslev High School. Four elementary skills (addition, subtraction, multiplication, and comparison), four operational skills (cancelling, conversions between improper fractions and mixed numbers and vice-versa, common denominator), and abilities to apply operational skills to complex tasks were tested. The university students summarised the results as a list of data records. Several misconceptions were discovered and included as variables into the model.

Second, a model structure was learned using the PC-algorithm [22], implemented in Hugin [9]. It provided a first insight into the relations between skills and misconceptions. Then a "domain expert" explained some relations with the help of hidden variables and introduced certain constraints on edges. Applying different constraints on the resulting model the final model was learned, again using the PC-algorithm. The final model was calibrated using the EM-algorithm as proposed in [12].

In Figure 4 the final version of the student model is presented. Nodes in the first level (from bottom) correspond to the observed misconceptions, grey nodes in the second level to elementary skills, nodes with no fill in the second level correspond to operational skills, and grey nodes in the third level to application skills. The top node corresponds to a hidden variable. See [24] where a more detailed description can be found.

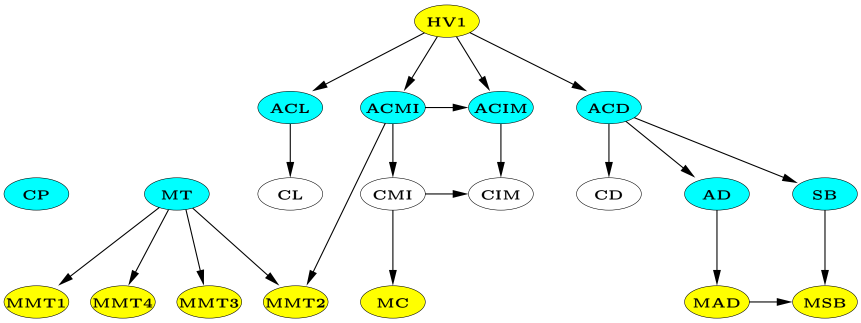

Fig. 4. Student model describing relations between skills and misconceptions.

For each question or task $X_j \in \mathcal{X}$ one evidence model is created by a domain expert. It describes the dependence of X_j on relevant skills from the student model. For every question the expert specifies which skills are related to the question. An example of a task is $\frac{1}{3} - \frac{1}{12} = \frac{4}{12} - \frac{1}{12} = \frac{3}{12} = \frac{1}{4}$. In order to be able to solve the task the student should have skills SB (subtraction) CL (cancelling), ACL (application of cancelling), CD (common denominator), ACD (application of common denominator), and should not have MSB (a misconception in subtraction). Thus, the relation between a variable T_j and related skills and misconceptions is described by a Boolean function. However, a student can make a mistake even if she has all abilities necessary to solve a given task and a correct answer does not necessarily mean that the student has all abilities. This uncertainty was modelled by a conditional probability distributions $P(X_j \mid T_j)$ estimated from the collected data. See Figure 5 for the model of the example task given above.

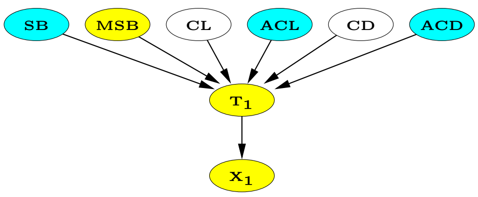

Fig. 5. Model describing relations between skills, misconceptions, and the example task.

The task models are connected to the student model when the corresponding tasks are solved by the tested student. Each time the corresponding variable X_j is instantiated with the answer provided by the student.

4. BUILDING ADAPTIVE TESTS

Typically, an adaptive test terminates after a given number of questions is answered or if sufficient information about the tested student is achieved. This defines the set of all possible test strategies $\mathcal{S}$. Every examiner tends to maximise information about the student at the end of a test. A way to formalise this preference is to aim at a probability distribution $P(Y_1, \ldots, Y_k)$ minimising the Shannon entropy at the end of the test [3]. The entropy of $P(Y_1, \ldots, Y_k)$ is defined[2] as

$$H(P(Y)) \quad = \quad - \sum_{y_1,\ldots,y_k} P(Y_1 = y_1, \ldots, Y_k = y_k) \cdot \log P(Y_1 = y_1, \ldots, Y_k = y_k)\,.$$

In every terminal node ℓ of a test strategy $\mathbf{s}$ we will compute entropy of the conditional probability distribution $P(Y_1, \ldots, Y_k \,|\, \mathbf{e}_\ell)$ given the evidence collected as far. Using substitution $f(\mathbf{e}_\ell) = H(P(Y_1, \ldots, Y_k \,|\, \mathbf{e}_\ell))$ formula (1) can be written as

$$E_H(\mathbf{s}) \quad = \quad \sum_{\ell \in \mathcal{L}(\mathbf{s})} P(\mathbf{e}_\ell) \cdot H(P(Y_1, \ldots, Y_k \,|\, \mathbf{e}_\ell))\,. \qquad (2)$$

The goal is to find a test strategy $\mathbf{s} \in \mathcal{S}$ minimising the expected entropy $E_H(\mathbf{s})$. We will call such a strategy an *optimal strategy* and denote it $\mathbf{s}^\star$.

4.1. Relation to sequential fault diagnosis

The problem defined above differs from *sequential fault diagnosis* [16] (or the test sequencing problem) in several aspects.

First, we allow imperfect questions, i.e., questions that have the conditional probability of at least one answer given a combination of values of skills different from one and zero.

Second, we terminate test after certain number of questions was asked, in contrast to sequential fault diagnosis, where a test terminates only if a faulty state was identified.

Third, we represent the modelled system by a probabilistic model over several variables probabilistically related to each other. In the sequential fault diagnosis the state of the system is modelled by a probability distribution over one variable.

Fourth, in our problem definition, the costs of all questions (tests) are equal.

4.2. Efficient computation of entropy

An important fact is that in any node n of a strategy the value of $H(P(Y|\mathbf{e}_n))$ can be computed efficiently exploiting the representation of Bayesian network $P(Y)$ by a junction tree [11].

The nodes of a junction tree corresponds to cliques of a moralized and triangularized graph of the Bayesian network. During the computation of entropy $H(P(Y|\mathbf{e}_n))$, first, the clique potentials are updated by the evidence $\mathbf{e}_n$, e. g., using a junction tree

[2]It will be convenient to have defined $\frac{0}{0} = 0$ and $0 \cdot \log 0 = 0$.

propagation method [11, 13]. Then the entropy is computed locally by adding entropy of potentials associated to cliques of the junction tree and subtracting entropy of potentials associated to separators of the junction tree. The complexity of this computation is proportional to the total clique size. In many real situations this allows to work with domains with hundreds of variables.

4.3. Myopic approach

In practice, quite often, a greedy approach is used to construct a myopically optimal test. A myopically optimal test is a test that consists of questions such that each question minimises the expected value of entropy after the question is answered.

Written formally, for each node n of a myopically optimal test $\mathbf{s}$ it holds that $\mathbf{e}_{m_i} = \mathbf{e}_n \cup \{X(\mathbf{e}_n) = x_i\}$, where for $i = 1, \ldots, t$ evidence $\mathbf{e}_{m_i}$ corresponds to child m_i of node n,

$$X(\mathbf{e}_n) \quad = \quad \arg\min_{X} \sum_{x} P(X = x \,|\, \mathbf{e}_n) \cdot H(P(Y \,|\, \mathbf{e}_n, X = x)) \,,$$

and t is number of states of $X(\mathbf{e}_n)$.

4.4. Strategy decomposition

Next we will propose a decomposition of a strategy $\mathbf{s}$. Latter it will be used to describe a search algorithm based on dynamic programming and to define an admissible heuristic function used during the search of an optimal strategy.

Let $\mathbf{s}'$ denote a strategy that is a substrategy[3] of an admissible strategy $\mathbf{s}$ and that has the same root ϑ as strategy $\mathbf{s}$. If $\mathbf{s}' \neq \mathbf{s}$ we will call $\mathbf{s}'$ an *incomplete strategy*. If v is a node of strategy $\mathbf{s}$ then by $\mathbf{s}^{\to v}$ we will denote the substrategy of $\mathbf{s}$ that has node v as its root and $\mathcal{L}(\mathbf{s}^{\to v}) \subset \mathcal{L}(\mathbf{s})$. We can do following decomposition of a strategy $\mathbf{s}$ (see Figure 6).

Using nodes r from the set of leaves of strategy $\mathbf{s}'$, $\mathcal{L}(\mathbf{s}')$, we get a set of strategies $\{\mathbf{s}^{\to r}, r \in \mathcal{L}(\mathbf{s}')\}$ where each strategy $\mathbf{s}^{\to r}$ is rooted in one node from $\mathcal{L}(\mathbf{s}')$. Note that $\cup_{r \in \mathcal{L}(\mathbf{s}')} \mathcal{L}(\mathbf{s}^{\to r}) = \mathcal{L}(\mathbf{s})$.

For leaves of a strategy[4] $\mathbf{s}$ we define the conditional expected entropy

$$E_H(\mathbf{e}_\ell) \quad = \quad H(P(Y \,|\, \mathbf{e}_\ell)) \,.$$

For evidence $\mathbf{e}_n$ associated with a non-leaf node n with set of children $ch(n)$ in $\mathbf{s}$ we define conditional expected entropy recursively

$$E_H(\mathbf{e}_n) \quad = \quad \sum_{m \in ch(n)} P(\mathbf{e}_m \,|\, \mathbf{e}_n) \cdot E_H(\mathbf{e}_m) \,.$$

Note that $E_H(\mathbf{s}) \equiv E_H(\emptyset)$. By $E^\star_H(\mathbf{e}_n)$ we will denote the conditional expected entropy of an optimal strategy $\mathbf{s}^\star$ given a evidence $\mathbf{e}_n$. Note that we can use the

[3] In the same sense as a subtree of a tree.

[4] For simplicity, we omit $\mathbf{s}$ in the index since the strategy should be clear from the context.

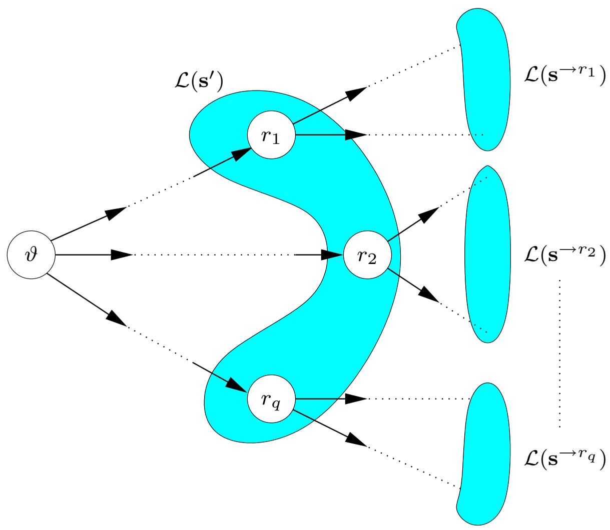

Fig. 6. Decomposition of strategy $\mathbf{s}'$.

partition of strategy $\mathbf{s}$ to compute the expected entropy of $\mathbf{s}$ using following formula

$$E_H(\mathbf{s}) \quad = \quad \sum_{r \in \mathcal{L}(\mathbf{s}')} P(\mathbf{e}_r) \cdot E_H(\mathbf{e}_r) \,. \tag{3}$$

4.5. Dynamic programming approach

Having this decomposition, we can easily construct a search algorithm based on the dynamic programming approach. The algorithm first evaluate all leaves ℓ of all possible strategies

$$E_H^\star(\mathbf{e}_\ell) \quad = \quad H(P(Y_1, \ldots, Y_k \,|\, \mathbf{e}_\ell))$$

and than processes upwards so that in each chance (AND) node n it computes

$$E_H^\star(\mathbf{e}_n) \quad = \quad \sum_{m \in ch(n)} P(\mathbf{e}_m \,|\, \mathbf{e}_n) \cdot E_H^\star(\mathbf{e}_m)$$

and in each decision (OR) node n it selects a child node minimising E_H

$$E_H^\star(\mathbf{e}_n) \quad = \quad \min_{m \in ch(n)} E_H^\star(\mathbf{e}_m) \,.$$

It finishes in the root node with $E_H^\star(\emptyset)$ being the minimal value of the expected entropy from all possible strategies. The optimal strategy $\mathbf{s}^\star$ can be easily traced up if the best children of decision nodes are stored.

A problem is that the search space is typically very large. Assume we aim at an optimal adaptive test of length n consisting of questions chosen from m possible questions. If we use dynamic programming approach we need to evaluate all nodes in a state graph. If all questions have two answers (e. g., correct and wrong), the number of expanded nodes is:

- $\sum_{i=1}^{n} 2^{i-1} \cdot \binom{m}{i-1}$ of decision (OR) nodes,
- $\sum_{i=1}^{n} 2^{i-1} \cdot \binom{m}{i-1} \cdot (m-i+1)$ of chance (AND) nodes, and
- $2^n \cdot \binom{m}{n}$ of leaves.

In each decision node a maximisation operation is performed, in each chance node an addition is performed, and in each leaf we compute the conditional entropy $H(P(Y|\mathbf{e}_\ell))$.

4.6. Admissible heuristics

One can try to avoid an extensive search by performing a top-down heuristic search driven by an admissible heuristic function. The goal of this section is to propose such an heuristic function. First, we will state and prove two lemmas that will be used to prove the main theorem.

Lemma 1. Let $P(A,B)$ be a probability distribution defined on Cartesian product of values of possibly multidimensional discrete variables A and B and $P(B=b) = \sum_a P(A=a, B=b)$. Then $H(P(A,B)) \geq H(P(B))$.

Proof. Since $P(B=b) = \sum_a P(A=a,B=b)$ and $P(A=a,B=b) \geq 0$

$$\begin{aligned} P(A=a,B=b) &\leq P(B=b) \\ \log P(A=a,B=b) &\leq \log P(B=b) \\ P(A=a,B=b)\cdot \log P(A=a,B=b) &\leq P(A=a,B=b)\cdot \log P(B=b). \end{aligned}$$

Since $H(P(A,B)) = \sum_{a,b} -P(A=a,B=b)\cdot \log P(A=a,B=b)$ we get

$$\begin{aligned} H(P(A,B)) &\geq \sum_{a,b} -P(A=a,B=b)\cdot \log P(B=b) \\ &\geq \sum_{b} -P(B=b)\cdot \log P(B=b) \\ H(P(A,B)) &\geq H(P(B))\,, \end{aligned}$$

which proves the lemma. □

Recall that $\cup_{r\in\mathcal{L}(\mathbf{s}')}\mathcal{L}(\mathbf{s}^{\rightarrow r}) = \mathcal{L}(\mathbf{s})$. Therefore $\sum_{r\in\mathcal{L}(\mathbf{s}')}\sum_{\ell_r\in\mathcal{L}(\mathbf{s}^{\rightarrow r})} P(\mathbf{e}_{\ell_r}) = 1$ and $P(\mathbf{e}_{\ell_r}), \ell_r \in \mathcal{L}(\mathbf{s}'), r \in \mathcal{L}(\mathbf{s}')$ is a probability distribution over the leaves of $\mathbf{s}$.

Thus we can define the entropy of the probability distribution over the leaves of strategy $\mathbf{s}$ in two equivalent ways

$$H(P(\mathbf{e}_\ell)) = \sum_{\ell \in \mathcal{L}(\mathbf{s})} -P(\mathbf{e}_\ell) \cdot \log P(\mathbf{e}_\ell) = \sum_{r \in \mathcal{L}(\mathbf{s}')} \sum_{\ell_r \in \mathcal{L}(\mathbf{s}^{\to r})} -P(\mathbf{e}_{\ell_r}) \cdot \log P(\mathbf{e}_{\ell_r}) .$$

It will be convenient to have defined also the entropy of the probability distribution over the leaves of strategy $\mathbf{s}'$

$$H(P(\mathbf{e}_r)) = \sum_{r \in \mathcal{L}(\mathbf{s}')} -P(\mathbf{e}_r) \cdot \log P(\mathbf{e}_r) .$$

Lemma 2. Let $\mathbf{s}'$ be an incomplete strategy and let u_r be the number of leaves of strategy $\mathbf{s}^{\to r}$ for $r \in \mathcal{L}(\mathbf{s}')$. Then

$$H(P(\mathbf{e}_\ell)) - H(P(\mathbf{e}_r)) \leq \sum_{r \in \mathcal{L}(\mathbf{s}')} P(\mathbf{e}_r) \cdot \log u_r .$$

P r o o f. Since entropy is maximised by a uniform distribution it holds for each $r \in \mathcal{L}(\mathbf{s}')$ that

$$H(P(\mathbf{e}_{\ell_r} \,|\, \mathbf{e}_r)) \leq H(\frac{1}{u_r}) = \log u_r$$

Therefore

$$\sum_{r \in \mathcal{L}(\mathbf{s}')} P(\mathbf{e}_r) \cdot H(P(\mathbf{e}_{\ell_r} \,|\, \mathbf{e}_r)) \leq \sum_{r \in \mathcal{L}(\mathbf{s}')} P(\mathbf{e}_r) \cdot \log u_r \tag{4}$$

The left hand side of inequality (4) can be rewritten as

$$\begin{aligned}
&\sum_{r \in \mathcal{L}(\mathbf{s}')} P(\mathbf{e}_r) \cdot H(P(\mathbf{e}_{\ell_r} \,|\, \mathbf{e}_r)) \\
&= \sum_{r \in \mathcal{L}(\mathbf{s}')} P(\mathbf{e}_r) \cdot \sum_{\ell_r \in \mathcal{L}(\mathbf{s}^{\to r})} -\frac{P(\mathbf{e}_{\ell_r})}{P(\mathbf{e}_r)}) \cdot \log \frac{P(\mathbf{e}_{\ell_r})}{P(\mathbf{e}_r)} \\
&= \sum_{r \in \mathcal{L}(\mathbf{s}')} \sum_{\ell_r \in \mathcal{L}(\mathbf{s}^{\to r})} -P(\mathbf{e}_{\ell_r}) \cdot \log P(\mathbf{e}_{\ell_r}) + \sum_{r \in \mathcal{L}(\mathbf{s}')} P(\mathbf{e}_r) \cdot \log P(\mathbf{e}_r) \\
&= H(P(\mathbf{e}_\ell)) - H(P(\mathbf{e}_r)) ,
\end{aligned}$$

which, when substituted to inequality (4), proves the lemma. □

Theorem 1. Assume an incomplete strategy $\mathbf{s}'$ having entropy $H(P(Y \,|\, \mathbf{e}_r))$ in each leaf $r \in \mathcal{L}(\mathbf{s}')$. Then for the expected entropy of any strategy $\mathbf{s} \in \mathcal{S}$ such that strategy $\mathbf{s}'$ is its substrategy it holds that

$$E_H(\mathbf{s}) \geq \sum_{r \in \mathcal{L}(\mathbf{s}')} P(\mathbf{e}_r) \cdot (H(P(Y \,|\, \mathbf{e}_r)) - \log u_r) . \tag{5}$$

P r o o f. Using the decomposition of strategy $\mathbf{s}$ proposed in Subsection 4.2 we can write the definition of $E_H(\mathbf{s})$ (given by formula (2))

$$\begin{aligned} E_H(\mathbf{s}) &= \sum_{r \in \mathcal{L}(\mathbf{s}')} \sum_{\ell_r \in \mathcal{L}(\mathbf{s}^{\to r})} P(\mathbf{e}_{\ell_r}) \cdot H(P(Y \mid \mathbf{e}_{\ell_r})) \\ &= \sum_{r \in \mathcal{L}(\mathbf{s}')} \sum_{\ell_r \in \mathcal{L}(\mathbf{s}^{\to r})} \sum_{y} -P(\mathbf{e}_{\ell_r}) \cdot P(y \mid \mathbf{e}_{\ell_r}) \cdot \log \frac{P(y, \mathbf{e}_{\ell_r})}{P(\mathbf{e}_{\ell_r})} \\ &= H(P(Y, \mathbf{e}_\ell)) - H(P(\mathbf{e}_\ell)) \,. \end{aligned} \tag{6}$$

From Lemma 1 it follows that

$$H(P(Y, \mathbf{e}_\ell) \quad \geq \quad H(P(Y, \mathbf{e}_r)) \tag{7}$$

and from Lemma 2 we have

$$-H(P(\mathbf{e}_\ell)) \quad \geq \quad -H(P(\mathbf{e}_r)) - \sum_{r \in \mathcal{L}(\mathbf{s}')} P(\mathbf{e}_r) \cdot \log u_r \,. \tag{8}$$

Substituting inequalities (7) and (8) to formula (6) we finally get

$$\begin{aligned} E_H(\mathbf{s}) &\geq H(P(Y, \mathbf{e}_r)) - H(P(\mathbf{e}_r)) - \sum_{r \in \mathcal{L}(\mathbf{s}')} P(\mathbf{e}_r) \cdot \log u_r \\ &\geq \sum_{r \in \mathcal{L}(\mathbf{s}')} P(\mathbf{e}_r) \cdot H(P(Y \mid \mathbf{e}_r)) + \sum_{r \in \mathcal{L}(\mathbf{s}')} \sum_{y} P(y, \mathbf{e}_r) \cdot \log P(\mathbf{e}_r) \\ &\quad - H(P(\mathbf{e}_r)) - \sum_{r \in \mathcal{L}(\mathbf{s}')} P(\mathbf{e}_r) \cdot \log u_r \\ &\geq \sum_{r \in \mathcal{L}(\mathbf{s}')} P(\mathbf{e}_r) \cdot H(P(Y \mid \mathbf{e}_r)) - \sum_{r \in \mathcal{L}(\mathbf{s}')} P(\mathbf{e}_r) \cdot \log u_r \,, \end{aligned}$$

which corresponds to inequality (5) we wanted to prove. □

Remark 3. Assume the set of admissible strategies $\mathcal{S}$ consists of all test strategies n questions long. If each question has two possible outcomes then each test strategy corresponds to a tree that has $u = 2^n$ leaves. Even before the search starts we can, using the Theorem 1, constraint the optimal value of the expected entropy by

$$E_H(\mathbf{s}^\star) \quad \geq \quad H(P(Y)) - n \log 2 \,.$$

This provides a natural interpretation of entropy in the context of adaptive testing. If our knowledge about a student is represented by a probability distribution $P(Y)$ with entropy $H(P(Y))$ then we would need at least $\frac{H(P(Y))}{\log 2}$ perfect questions, with two outcomes each, to yield precise information about a student knowledge state.

Theorem 1 naturally leads to a lower bound on $E_H(\mathbf{s})$ given probability distributions $P(Y \mid \mathbf{e}_{\ell_r})$ in the leaves $r \in \mathcal{L}(\mathbf{s}')$ of an incomplete strategy. For each incomplete strategy $\mathbf{s}'$ we can define

$$\underline{E}_H(\mathbf{s}') \quad = \sum_{r \in \mathcal{L}(\mathbf{s}')} P(\mathbf{e}_r) \cdot (H(P(Y \mid \mathbf{e}_r)) - \log u_r) \ .$$

It is a consequence of Theorem 1 that $\underline{E}_H(\mathbf{s}')$ is an admissible heuristics.

4.7. *AO** algorithm

We can use heuristics $\underline{E}_H(\mathbf{s}')$ in an $AO^\star$ algorithm [18]. In each step of the algorithm we select a strategy $\mathbf{s}'$ that has minimal value from all strategies expanded as far. Then we expand a non-expanded node n that is a child of a node from the selected strategy $\mathbf{s}'$. We set the value of $E_H(\mathbf{e}_n) = \underline{E}_H(\mathbf{e}_n)$ and using recursive formula (3) we recompute the values of all ancestor nodes of n.

The algorithm can proceed in a similar manner as it is described in [25], where an $AO^\star$ algorithm was used to search a troubleshooting strategy minimising the expected cost of repair of a device. The only difference are the admissible heuristics. The usage of admissible heuristics guarantee that the first expanded complete strategy is an optimal strategy [18].

5. EXPERIMENTS

In the experiments probability distribution $P(Y)$ modelled a student solving basic operations with fractions. It was represented by a Bayesian network having the structure presented in Figure 4. A bank $\mathcal{X}$ of twenty question was available.

We have performed experiments with this model on a real data set consisting of 149 data records, one for each tested student [6, 24]. The model was learnt from all but one data records and then tested on the remaining one. This was repeated for each data record. In Figure 7 we present how in average the entropy of conditional probability of skills $H(P(Y \mid \mathbf{e}))$ evolved for three different strategies - ascending, descending[5], and myopic. The straight line on the left hand side corresponds to the value of heuristics $\underline{E}_H(\emptyset)$ computed in the root node with no evidence as far. No strategy can get bellow this line.

Assume we want to construct a test of length $n = 10$. The straight line on the right hand side is parallel to the former. If a strategy ever crosses this line then it cannot be better than the myopic strategy after answering ten questions and consequently it cannot be optimal. Thus during the search of an optimal strategy using the $AO^\star$ algorithm exploiting heuristics proposed in this paper no strategy crossing this line is further expanded. In our example, the $AO^\star$ algorithm would stop expanding the ascending and descending strategies after the seventh questions.

More detailed results of experiments with myopic adaptive test of basic operations with fractions were presented in [24]. Bayesian network used in myopic adaptive

[5] The questions in these two strategies were simply chosen according to their index in the ascending and descending order, respectively.

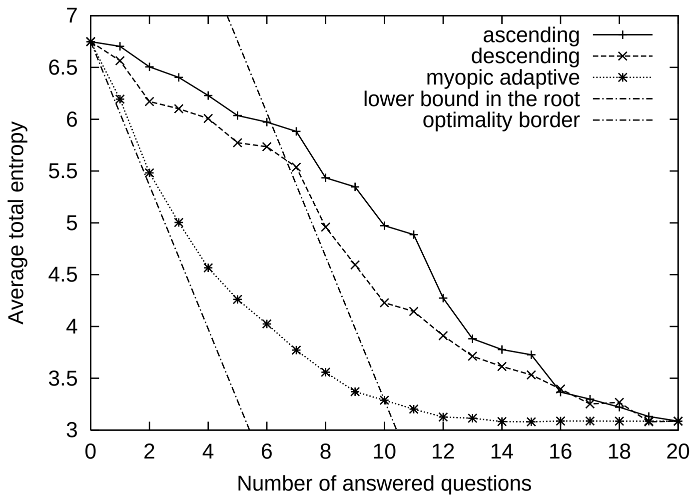

Fig. 7. Comparison of test strategies.

tests provided good predictions of skills. In average more than 90% of skills were correctly predicted after seven questions were answered. In paper and pencil tests twenty questions were typically needed to get the same prediction quality. See also [5] and [4] for results of experiments with Bayesian networks applied to adaptive diagnosis and tutoring.

6. CONCLUSIONS AND FUTURE WORK

In this paper we have proposed a framework for building decision strategies. We first learn a probabilistic model of the domain of interest and then we search for a best decision strategy. When building a probabilistic model we can exploit expert knowledge of the modelled domain as well as collected data. Another approach is to learn the best strategy in a integrated fashion [7]. This approach uses collected data only.

We have proposed an admissible heuristic function that can be used in the construction of a most informative adaptive test of a fixed length, e. g., within an $AO^\star$ algorithm. In future we intend to implemented the $AO^\star$ algorithm and perform additional experiments with real data that would show how significant is the reduction of the searched space.

In many applications (e. g. in troubleshooting man-made devices or in medical diagnosis) the observations do not have equivalent costs and miscellaneous criteria should be used to terminate strategies. This makes the search of an optimal strategy more complicated [25] and provides a motivation for further research.

(Received October 3, 2003.)

REFERENCES

[1] R. G. Almond and R. J. Mislevy: Graphical models and computerized adaptive testing. Appl. Psychological Measurement *23* (1999), 3, 223–237.

[2] S. Andreassen, F. V. Jensen, S. K. Andersen, B. Falck: V. Kjærulff, M. Woldbye, A. R. Sørensen, A. Rosenfalck, and F. Jensen: MUNIN — An expert EMG assistant. In: Computer-Aided Electromyography and Expert Systems, Elsevier Science Publishers (J. E. Desmedt, ed.), Amsterdam 1989.

[3] M. Ben-Bassat: Myopic policies in sequential classification. Trans. Comput. *27* (1978), 2, 170–174.

[4] C. Conati, A. S. Gertner, K. VanLehn, and M. J. Druzdzel: On-Line Student Modeling for Coached Problem Solving Using Bayesian Networks. In: Proc. of the Sixth Int. Conf. on User Modeling (UM97) (A. Jameson, C. Paris, and C. Tasso, eds.), Chia Laguna, Sardinia, Italy, 1997.

[5] E. Millán and J. L. Pérez-de-la-Cruz: A Bayesian Diagnostic Algorithm for Student Modeling and its Evaluation. User Modeling and User Adapted Interaction, *12* (2002), 2–3, 281–330.

[6] L. Būtėnas, A. Brilingaitė, A. Čivilis, X. Yin, and N. Zokaitė: Computerized Adaptive Test Based on Bayesian Network for Basic Operations with Fractions. Student Project Report, Aalborg University, 2001, http://www.cs.auc.dk/library.

[7] R. Greiner, A. J. Grove, and D. Roth: Learning cost-sensitive active classifiers. Artificial Intelligence *130* (2002), 2, 137–174.

[8] D. Heckerman, E. Horwitz, and B. Nathwani: Towards normative expert systems: Part I, the Pathfinder project. Methods Inform. Medicine *31* (1992), 90–105.

[9] Hugin Explorer, ver. 6.0. Comput. Software 2002, http://www.hugin.com.

[10] F. V. Jensen: Bayesian Networks and Decision Graphs. Springer-Verlag, New York – Berlin – Heidelberg 2001.

[11] F. V. Jensen, S. L. Lauritzen, and K. G. Olesen: Bayesian updating in recursive graphical models by local computation. Comput. Statist. Quarterly *4* (1990), 269–282.

[12] S. L. Lauritzen: The EM-algorithm for graphical association models with missing data. Comput. Statist. Data Anal. *1* (1995), 191–201.

[13] S. L. Lauritzen and D. J. Spiegelhalter: Local computations with probabilities on graphical structures and their application to expert systems (with discussion). J. Roy. Statist. Soc. Ser. B *50* (1988), 157–224.

[14] S. L. Lauritzen: Some Modern Applications of Graphical Models. In: Highly Structured Stochastic Systems (P. J. Green, N. L. Hjort, and S. Richardson, eds.), Oxford University Press, 2002.

[15] F. M. Lord: A theory of test scores. Psychometrica Monograph No. *7* (1952).

[16] K. R. Pattipati and M. G. Alexandridis: Application of heuristic search and information theory to sequential fault diagnosis. IEEE Trans. Systems Man Cybernet. *20* (1990), 4, 872–887.

[17] J. Pearl: Reverend Bayes on inference engines: a distributed hierarchical approach. In: Proc. AAAI National Conference on AI, Pittsburgh 1982, pp. 133–136.

[18] J. Pearl: Heuristics – Intelligent Search Strategies for Computer Problem Solving. Addison-Wesley, Reading, MA 1984.

[19] J. Pearl: Fusion, propagation and structuring in belief networks. Artificial Intelligence *29* (1986), 3, 241–288.

[20] G. Rasch: Probabilistic Models for Some Intelligence and Attainment Tests. Technical Report, Danish Institute for Educational Research, Copenhagen 1960.

[21] D. J. Spiegelhalter and R. P. Knill-Jones: Statistical and knowledge-based approaches to clinical decision-support systems. J. Roy. Statististics Soc. Ser. A *147* (1984), 35–77.

[22] P. Spirtes, C. Glymour, and R. Scheines: Causation, Prediction, and Search (Lecture Notes in Statist 81), Springer-Verlag, Berlin 1993.

[23] W. J. Van Der Linden and C. A. W. Glas: Computerized Adaptive Testing: Theory and Practice. Kluwer Publishers, Dordrecht 2000.

[24] J. Vomlel: Bayesian networks in educational testing. Int. J. of Uncertainty, Fuzziness and Knowledge Based Systems, *12* (2004), Supplementary Issue 1, 83–100.

[25] M. Vomlelová and J. Vomlel: Troubleshooting: NP-hardness and solution methods. Soft Comput. J. *7* (2003), 5, 357–368.

[26] H. Wainer, D. Thissen, and R. J. Mislevy: Computerized Adaptive Testing: A Primer. Second edition. Mahwah, Lawrence Erlbaum Associates, N. J. 2000.

Jiří Vomlel,

Laboratory for Intelligent Systems, University of Economics, Ekonomická 957, 148 01 Prague 4. Czech Republic and

Institute of Information Theory and Automation, Academy of Sciences of the Czech Republic, Pod vodárenskou věží 4, 182 08 Prague 8. Czech Republic.

e-mail: vomlel@utia.cas.cz