Clinical Reasoning Automata for Simulated Patients

Froduald Kabanza[1] and Guy Bisson[2]

University of Sherbrooke
Sherbrooke, Québec J1K2R1, Canada
[1] Department of Computer Science
Froduald.Kabanza@usherbrooke.ca
[2] Faculty of Medicine
Guy.Bisson@Usherbrooke.ca

Abstract. In this paper we introduce *clinical reasoning automata* to model states and transitions about different cognitive processes that occur during a clinical reasoning activity. A state of the automaton represents a particular process in a complex patient diagnosis using influence diagrams encoding clinical knowledge about the case. Transitions model switch between diagnosis cognitive processes, such as collecting evidences, formulating hypothesis or explicitly asking for assistance at a given point during the reasoning process. That way, we can efficiently model tutoring feedback hints for clinical reasoning learning that are based not only on the clinical knowledge, but also on the sequencing of the tutoring processes.

1 Introduction

Medical diagnostic problems that are the focus of clinical reasoning are examples of complex decision making problems for which solutions depend not only upon clinical knowledge and medical experience, but also on well drilled strategies for collecting evidences about pathologies, analyzing them, formulating hypotheses and evaluating them based on collected evidences. Clinical reasoning learning (CRL) in medicine deals with teaching medical students to acquire the basics of such cognitive strategies and to apply them [6].

In many medical schools, CRL is taught using a problem-based learning approach (PBL). Students, in small groups (typically six to eight students) are presented a patient case and asked to diagnose it and propose a treatment or a management plan. Under the supervision of a professor, students will usually formulate initial hypotheses just after getting a brief presentation of the case, under the form of a textual problem statement, a live encounter with a patient simulated by a student playing a patient scenario prepared by a professor, or a live encounter with a real patient recruited from a clinic or hospital. Then they will begin to collect evidences by asking questions to the patient (e.g., the motive of the consultation, in which parts of the body he is experiencing problems, how long and frequently the symptoms occur, his life style, and so on), then they will proceed with a directed physical exam (e.g., sensing reflexes) and after that order laboratory/imaging tests (e.g., urine tests, blood tests, X-rays or MRI). The evidences are analyzed and then the hypotheses are reevaluated to discard those not fitting with the analysis, to keep those still uncertain, and to collect again new evidences in order to discard or confirm remaining hypotheses or generate new ones. This process of evidence collection, evidence analysis, hypothesis evaluation and hypothesis reformulation is iterated until the students are able to narrow their list of probable hypotheses to the one or two most probable diagnosis.

Students conduct their investigation based upon their declarative knowledge they have learned in PBL sessions and also on similar cases they have already seen in clinical settings or in previous CRL sessions. The clinical knowledge linking symptoms to diseases is at least in part a probabilistic exercise whereas the actions used to collect evidences (i.e., questions, physical exams and lab tests) involve tradeoffs about costs, harm to the patient, and the value of the information obtained from the tests with regard to the current hypotheses. The professor uses his own experience and knowledge to provide hints to the students during their investigation and sometimes to comment on variant of similar cases.

Because of the cognitive complexity of CRL, it is crucial that students have a varied, very well organized and structured knowledge base, and be exposed to as many cases as possible in their curriculum, since this is mostly an experience-based learning process. To foster the acquisition of their clinical reasoning skills, it is mandatory that the CRL involves small group sizes. One of the drawbacks is that this learning activity requires a lot of resources in terms of professor's availability for supervising the CRL. These professors are for the most part also physicians and/or researchers, and since most countries are faced with a severe shortage of physicians, the burden is even higher on them.

Many medical schools are therefore turning towards the use of software simulations to support at least in part the CRL activities. One possible strategy is to keep an affordable number of CRL courses directly supervised by professors and to have students practice on computer-based cases that are variants of those learned in classes, in groups or individually, with the assumption that computer-based cases will require little professor supervision or no supervision at all if the computer has enough intelligence capabilities to automatically provide the required feedback to students.

Providing feedback to students in computer-based CRL requires a system that not only is able to assess the student's strengths and weaknesses of his clinical reasoning process to solve a patient case, but is also able to apply metacognitive strategies to identify and correct flaws in his reasoning strategies, for instance, with regard to how he interleaves evidence collection and the hypothesis formulation processes. However, computer-based CRL systems proposed to date focus mostly on providing feedback with regard to the application of declarative clinical knowledge, with very little emphasis on feedback related to cognitive strategies about the sequencing of clinical reasoning sub-processes [3], [4], [9], [10].

For instance, DxR is one of the most used commercial CRL software with a database of about a hundred clinical cases, covering both patient diagnosis and management and illustrates this kind of approach [3]. However, DxR has no built-in student's model other than a simple decision tree checking whether the student has reached the correct diagnosis within a given deadline and has considered all relevant hypotheses.

The decision tree is explicitly specified by a professor who authors the case. Consequently, students using DxR autonomously learn mostly by reinforcement as the decision tree indicates whether they have the right hypotheses or not, but with little ongoing feedback about their reasoning processes. Other commercial CRL such as VIPS have the same limitations [10].

The Adele [4] and COMET [9] systems provide more advanced automated tutoring feedback by using an influence diagram (i.e., a Bayesian network with actions and action utilities) to model the clinical knowledge about the patient. In fact the influence diagram represents expert knowledge about how symptoms probabilistically relate to diseases and how evidence-collect actions or clinical-knowledge-apply actions relate to evidences. It is essentially the same kind of Bayesian network used in medical decision support systems to recommend diagnosis, to collect new evidences, or treatment to clinicians in real-life cases [7], [8]. The generation of tutoring feedbacks is done by comparing the recommendations obtained from the network to actions, evidences or hypotheses found by the students and providing hints accordingly.

More specifically, at any step of the interaction between the student and the CRL interface that simulates a patient, we can recognize the evidence collected by the student and we can also determine the hypothesis he is currently working on either by asking them to the student directly, or by inferring them. Thus, given the current evidences as collected by the students we can use Bayesian inference algorithms to determine the most likely hypotheses from the expert point of view and compare them with the hypotheses currently inferred by the student. Based on the comparison, we can for example give hints to the students about whether he is on the right track or not, or give him clinical knowledge that is most appropriate to arrive at the evidence. Conversely, given the current hypotheses collected by the student, we could use again Bayesian inference algorithms to determine the most valuable evidences to collect next, and exploit this information to give hints about which evidences to collect next and how.

However, this approach does not model CRL strategies about how to interleave and refine the different phases of evidence collection, evidence analysis, hypothesis evaluation and hypothesis reformulation. A professor supervising CRL will provide hints not just aimed at instructing the student how to apply clinical knowledge, but also hints aimed at sharing with him his “rules of thumb” about how to interleave the different phases of evidence collection, evidence analysis, hypothesis evaluation and hypothesis reformulation. For instance, he may remind him when it is appropriate to stop the current evidence collection in order to re-evaluate the hypotheses. He may also explain why at a given step it’s better to collect evidences that rule out one current hypothesis as opposed to evidences that confirm a current active hypothesis, for instance as a measure to minimize the cost of evidence collection and not jeopardize the patient health. He may also intervene at certain points to provide structured knowledge about variant cases related to the one under study. For instance after a student has correctly interpreted evidences gathered from a lab test based on the current patient’s data, the professor may discuss patient cases for which the same test results would have yield a different interpretation. Such strategic interventions are not easily conveyed by Bayesian networks; in fact none of the above systems addresses them.

We introduce *clinical reasoning automata* (CRA) in our model on top of influence diagrams to fill this gap. These are hybrid finite state automata representing allowed sequences and refinements of the different phases of evidence collection, evidence analysis, hypothesis evaluation and hypothesis reformulation. The automaton has a finite number of states, but each state represents a clinical reasoning sub-process, that is, the system continues to evolve within a state, hence the qualification “hybrid”. A process modeled by a state has access to a global influence diagram encoding clinical knowledge as in Adele [4] and COMET [9], and local state-based production rules to provide feedback. This provides a means for modeling sequenced feedback rules, and also leads to a modular specification of tutoring feedbacks.

In the next section we present the system that we currently develop to simulate patients and foster clinical reasoning cognitive strategies. Then we discuss the implementation of tutoring feedbacks in the CRA. After a conclusion, we discuss future developments.

2 Patient Simulation System

Our patient simulator, TeachMed, has a graphic user interface (GUI) that provides functionalities similar to DxR [3] and VIPS [10]. The student GUI allows the student to select a patient case, access the patient’s medical record, interview the patient, proceed with a physical exam on a 3D model of the patient (Fig. 1), interact with a laboratory module to order tests (Fig. 2), and finally treat and manage the patient. Professors use another GUI to author patient cases.

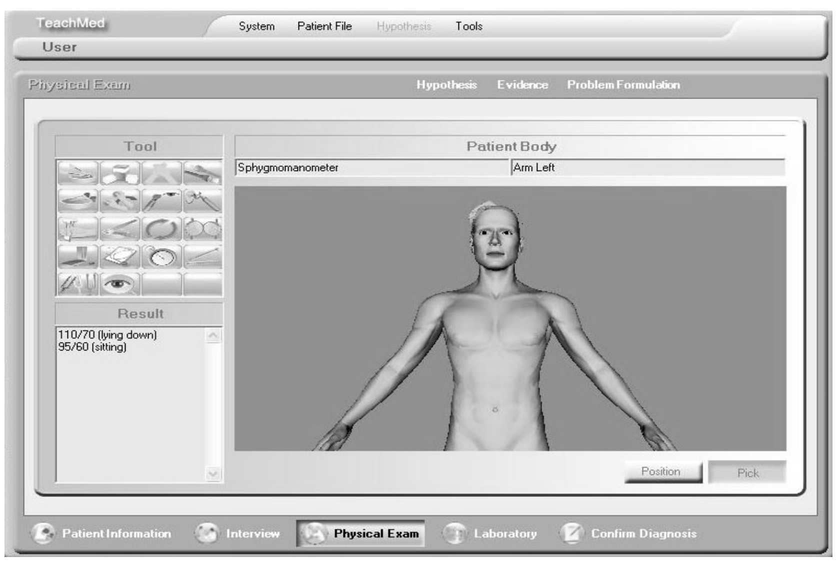

Fig. 1. The student decides to take the patient's blood pressure

3 Automated Tutor

A key original component of TeachMed is an automated tutor that monitors the student's actions to provide rapid feedback. Tutoring feedback is implemented based on a *clinical influence diagram*, the *current system state*, *feedback rules* triggered by the current system state, and a *clinical reasoning automaton* that sequences clinical reasoning sub-processes and corresponding feedback rules.

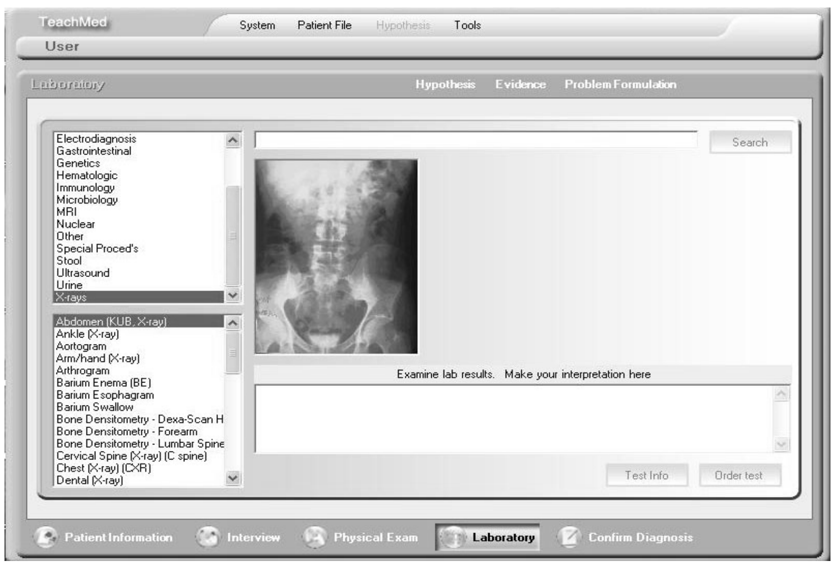

Fig. 2. Abdominal plain x-ray ordered by the student

3.1 Clinical Influence Diagram and Bayesian Inference Processes

A clinical influence diagram is a Bayesian network encoding the clinical diagnostic knowledge from an expert point of view. It specifies causal links between evidences, symptoms and pathologies (with corresponding probability distributions), as well as actions that investigate these symptoms (with corresponding utilities). Fig. 3 shows a fragment of an influence diagram we use for a case of acute diarrhea. Probabilities and utility specifications are omitted for clarity. The fragment illustrates only some nodes with few evidence-collection actions (questions to the patient and lab tests). The entire model for the diarrhea case contains 70 nodes, including physical exam actions and curriculum-knowledge application actions to collect evidences.

The influence diagram is implemented using Smile Bayesian network library [2]. Evidences gathered by the student are recorded into a *student evidence table*. Given this table, we use Smile library to compute the posterior probability for a set of hypotheses from the expert point of view and rank them according to their likelihood, that is, the *expert hypothesis table*. Intuitively, assuming the influence diagram correctly models the expert clinical knowledge, the expert would infer these hypotheses from the given evidences. On the other hand, based on his evidence table, the student will infer his own hypothesis table (i.e., the *student hypothesis table*) which may or may not match the expert hypothesis table.

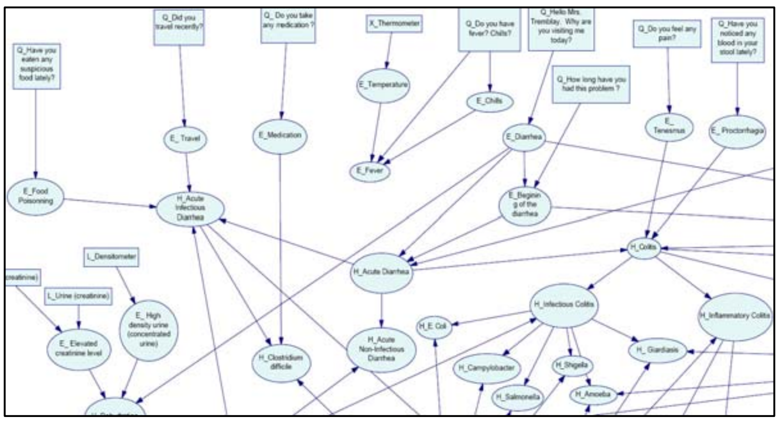

Fig 3. Fragment of the Bayesian network for the acute diarrhea case

In principle, the next best evidence-gathering process should be one yielding a value of information that exceeds the cost of gathering information. However, in general it is very difficult to express the utilities of evidence-gathering actions with a level of accuracy that takes into account all possible evidence observations [7],[8]. Therefore, we do not rely entirely on the utilities specified for evidence-gathering actions to provide feedback about the next evidence-gathering process. Rather we use these utilities as heuristic input to feedback rules that provide the actual hints about the next evidence-gathering steps. More specifically, based on the current student's evidence and hypothesis tables, we use Smile library to compute evidence-collection actions and store them into a *next student evidence-action table*, on top of which we can express facts that are preconditions of feedback rules. For instance, we can specify a feedback rule stating that "if not acute-infection-diarrhea and urine test's value of information is less than blood test no more by *x* threshold, then perform urine test".

In order to provide feedback TeachMed needs to determine the current hypothesis the student is trying to confirm or reject (i.e., the *student's hypothesis goal*) or the current evidence he is trying to collect (i.e., the *student's evidence goal*). We could obtain this information by simply asking it to the student, but this would result in too much intervention, disturbing the tutoring strategy, particularly for advanced students who just need occasional help. Instead, based on the observed student's actions (i.e., questions he asks, physical exams he makes, or lab tests he orders), we use Smile library to infer the evidence he is trying to collect, similarly for the student's hypothesis goal. If the output of this goal recognition inference is below a validity threshold, we prompt a question to the student to get directly from him the goal he is working on. If the recognition output is a list of at most three elements with valid probability above the threshold, the question looks like "are you trying to establish x, y or z?" Otherwise, the question is a direct one such as "what hypothesis are you trying to establish?"

In practice, experienced practitioners formulate hypotheses and evidences unconsciously, particularly in the early stages of their cognitive investigation [6]. They start to think about them explicitly only in later stages as their reasoning sub-processes become more involved, requiring them to weigh the benefits and disadvantages of the final decisions they are about to make (e.g., ordering a costly MRI versus waiting for the symptoms to develop further in order to enable clearer evidences). Nevertheless, when teaching, it is a good strategy to require novice students to express explicitly their current hypotheses and the evidences they have found, even during the early stages of their investigation; that way, the professor can evaluate for example whether their evidence gathering is indeed guided by their hypotheses and their validity. For this purpose, TeachMed's menu provides the possibility for the student to update directly his *evidence table* and his *hypothesis table*, if this is a requirement set by the professor when he authored the case. Nevertheless, for tutoring feedback we cannot entirely rely on these tables because students may forget to record the hypothesis in trying to collect evidences based on hypotheses in their minds but not in the table. In fact, TeachMed still invokes goal-recognition to infer the hypothesis the student may be working on and if different from those in his edited table, an appropriate feedback will be triggered.

3.2 Student Model and Model Tracer

TeachMed maintains a *student model*, reflecting the student's progress information inferred from the influence diagram as explained above (i.e., student evidence table, expert evidence table, student's hypothesis table, the student's hypothesis goal, and the student's evidence goal), and student's input for feedback request (e.g., the student can ask TeachMed to help him decide between two interpretations of a lab test). A process, called the *student's model tracer*, monitors the student's actions and periodically invokes the update processes described above to maintain the student's model.

3.3 Feedback Rules and Feedback Generator

The *feedback generator* is a process that periodically checks the current student's model to trigger tutoring feedback to the student, using production rules that are preconditioned on data in the student's model and having consequents that are tutoring hints. These are "teaching" expert rules and can be as efficient as the available teaching expertise allows.

To illustrate, a feedback rule of the form "if student-asks-help (interpret, test, results) then hint-test-interpretation (test, results)" will trigger a hint helping the student about the interpretation of a lab test. Following Andes' approach [5], hints are implemented using template-based heuristic dialogues, with slots replaced by data in the student's model. With the previous example, TeachMed will engage a tree-structured dialog with the student by asking him his impressions about the interpretation and issuing hints until the dialog concludes in the correct interpretation. Feedback rules can also be triggered by TeachMed initiative by having rules with antecedents based upon the evaluation of the student's progress as reflected by the student's model.

3.4 Clinical Reasoning Automaton

A *clinical reasoning automaton* (CRA) is a state transition system that abstracts the evolution of the student's model under the student's actions. The feedback generator starts in a given state of the automaton. In the current state, it is synchronized with the model tracer monitoring the student's actions, focusing on one sub-goal, and using a set of feedback rules to help the student. A transition to a new state switches the sub-goal and the feedback rules to those of the new state. From another perspective, a state of the CRA represents a sub-process of the clinical reasoning (e.g., the student is collecting evidence for "Acute Diarrhea"). Therefore, a CRA state is a dynamic state in that, the student's model continues to evolve; for example, the evidence and hypothesis table continue to change under the student's actions. A transition signals the occurrence of an event that is sufficiently interesting (from the point of view of a given tutoring strategy) to switch a context (e.g., the student's has finished collecting evidences for the current student's evidence goal).

More formally, a CRA is a hybrid automaton [1], consisting of:

- A finite set of variables, which are variables composing the student's model, a *time* variable (counting the time elapsed in a state), and possibly additional variables defined by the patient-case author (e.g., a student performance variable).
- A finite set of states, consisting of the variables (*time* and student's model variables remain implicit), feedback rules and functions about each variable evolution in the state.
- A start state.
- A finite set of transition between states.

A transition is a triplet of the form "*(u, test, v)*", where *test* is a Boolean expression over variables in state *u* (implicit and/or explicit), possibly involving author-defined Boolean functions over these variables. The transition means that if the *test* holds (i.e., the transition is enabled), then the feedback generator moves into state *v*. If many transitions are simultaneously enabled, then it moves in a non deterministic fashion into one of the successor states.

The functions defining the evolution of the author-defined variables are defined by the case author. For the others, the functions are defined as follows. The time variable is initialized to zero upon entering the state and is automatically incremented as time passes. The student's model variables are automatically updated by the *model tracer*.

We have seen that feedback rules are production rules preconditioned on the state variables and that the consequents are normally tutoring hints for the student as explained in the previous subsection. As a matter of facts, the case author can also specify “feedback rules” having only the internal effect of updating author-defined variables (e.g., to update a performance score for the student).

There is no explicit notion of accepting a run for a task automaton. Goodness or badness of runs is conveyed to the learner via the feedback given in states. However, it’s possible to implement rules with a side effect of updating a performance score for the student. One can also specify transitions to states in which the feedback indicates complete success for the task or complete failure.

To illustrate, assuming that we are in a state monitoring the student collecting evidence for some symptom, we can have an outgoing transition labeled “hypothesis-goal-changed” (i.e., the model tracer has established that the student has switched to another hypothesis) to a state in which we watch whether the student remembers to reformulate his hypotheses. In this new state, if a student attempts any action not related to the evaluation and re-formulation of his hypotheses he will get feedback hints aimed at making him remember why at this stage he should reformulate his hypotheses. It could be possible to express this feedback without using a CRA transition to a new state, but then this will result in complex, unstructured feedback rules that not only become difficult to maintain, but also are executed less efficiently since at every stage the system has a large number of rules to consider. In contrast, CRA feedback rules only depend on states and in the current state there are only few of them to consider, that is, those active in that state. It is possible to encode feedback rules that depend on a trace (i.e., a sequence of states), by defining a variable that records relevant part of the trace into a state and specifying feedback rules that have facts in their antecedents that depend on the current value of the variable.

4 Evaluation

TeachMed has been tested so far in laboratory. With a CRA reduced to just one single state our approach is similar to Adele’s [4] and COMET’s [9]. So far, our specifications demonstrate that CRA offer the advantages of specifying modular feedback rules – they are simpler to specify because organized depending on the current clinical reasoning sub-process— and also offer the possibility of modeling feedback hints about the sequencing of clinical reasoning sub-processes, which cannot be done easily with the above approaches

5 Conclusion

In our medical school, CRL are currently conducted live in small groups of eight students under the supervision of a professor who is a domain expert. One student is picked to play the role of a patient and another plays the role of the physician. The remaining students are observers but also actively participate in the discussion. The patient student behaves according to instructions prepared by the professor on paper based on pedagogical objectives that are fixed by the curriculum committee. The school is planning to start using TeachMed to complement these simulations in the fall of 2005 or the winter of 2006. The students will use TeachMed in an autonomous mode.

Besides this forthcoming deployment of TeachMed in our medical curriculum, future work includes the implementation of an interface between TeachMed and a real-world health record containing information about patients and how they were treated. As noted by Gertner *et al*. in their physics tutoring system [5], a Bayesian network implicitly models solutions, but not the solution process itself. Yet, being able to model the solution process can improve the quality of feedback given to the student. A similar observation holds in CRL, with the difference that the solving process is harder to formalize other than giving general strategies about the interleaving of the different clinical reasoning sub-processes. Nevertheless, we feel that the possibility of having one expert-solution, albeit not encompassing all experts' solutions, can be exploited to provide enriched feedback. In the more trivial case, we could illustrate the solution to the student as one example of expert's approach. Given that an electronic heath-record does not contain clinical reasoning traces, but contains just the solutions (i.e., the diagnosis and the patient treatment), the challenge will be to find techniques for inferring expert solving processes from expert solutions.

References

1. Alur, R., Courcoubetis, C., Henzinger, T., and Po. H.: Hybrid automata: an algorithmic approach to the specification and verification of hybrid systems. In: Proceedings of Hybrid Systems. Lecture Notes in Computer Science, Vol. 736, Springer Verlag, (1993):209-229.

2. Druzdzel, M.: SMILE: Structural Modeling, Inference, and Learning Engine and GeNIE: A Development Environment for Graphical Decision-Theoretic Models. In Proceedings of the Eleventh Conference on Innovative Applications of Artificial Intelligence, (1999) 902-903. (See http://www.sis.pitt.edu/~genie/ for system download).

3. DxR Clinician: http://www.dxrgroup.com/

4. Ganeshan, R., Johnson, L., Shaw, E. and Wood, B.: Tutoring Diagnostic Problem Solving. In: Proceedings of Conference on Intelligent Tutoring Systens, Lecture Notes in Computer Science, Vol. 1839, (2000) 33-42

5. Gertner, A., Conati, C. and VanLehn, K.: Procedural help in Andes: Generating Hints Using a Bayesian Network Student Model. In: Proceedings of National Conference on Artificial Intelligence, AAAI Press, (1998)106-111.

6. Gruppen, L. and Frohna, A.: Clinical Reasoning. International Handbook of Research in Medical education. Kluwer Academic (2000) 205–230

7. Heckerman, D., Horvitz, E. and Nathwani, B.: Towards Normative Expert Systems: Part I. The Pathfinder Project. In: Methods of Information in Medecine (1992): 90-105.

8. Onisko, A., Lucas, P. and Druzdzel M.: Comparison of Rule-Based and Bayesian Network Approaches in Medical Diagnostic Systems. In: Proceedings of the Eighth {Annual Conference on Artificial Intelligence in Medicine, Lecture Notes in Computer Sciences, vol. 2101, Springer Verlag, (2001): 281-292.

9. Suebnukarn, S. and Haddawy, P. Collaborative Intelligent Tutoring System for Medical Problem-Based Learning. In: Proceedings of the Ninth Conference on Intelligent User Interfaces, (2004)14-21.

10. VIPS: Virtual Internet Patient Simulator: https://www.swissvips.ch/fr/index.htm