GLIF: Decision steps with missing data

A. Veselý, D. Buchtela, J. Zvárová

Centre of Biomedical Informatics, Institute of Computer Science AS CR, Praha

Objective

Medical guidelines are used for standardization of medical care and for decision support. GLIF (Guidelines Interchange Format) [1] is one of the computer interpretable representations of medical guidelines used for sharing of medical knowledge among institutions and across computer applications. Medical decisions represented in GLIF model with decision steps depend on evaluation of conditions attached to the decision variants.

If values of some parameters contained in a condition are not known, an effective algorithm that would enable to determine if the condition must be valid cannot be valid or if its validity, due to the missing data, cannot be determined, is needed. In the paper we present more efficient algorithm based on the use of 3-valued logic and standard resolution algorithm.

GLIF model

GLIF model could be represented in a form of an oriented graph [2,3]. The nodes of the graph are guidelines steps and the edges represent continuation from one step to the other. Guidelines steps are ***action step***, ***state step***, ***decision step***, ***branch and synchronization steps*** (see fig.1).Branching conditions ($\alpha_1 \ldots \alpha_n$) that belong to some edge coming from decision step can be one of the following types: ***strict in***, ***strict out***, ***rule in and out***.

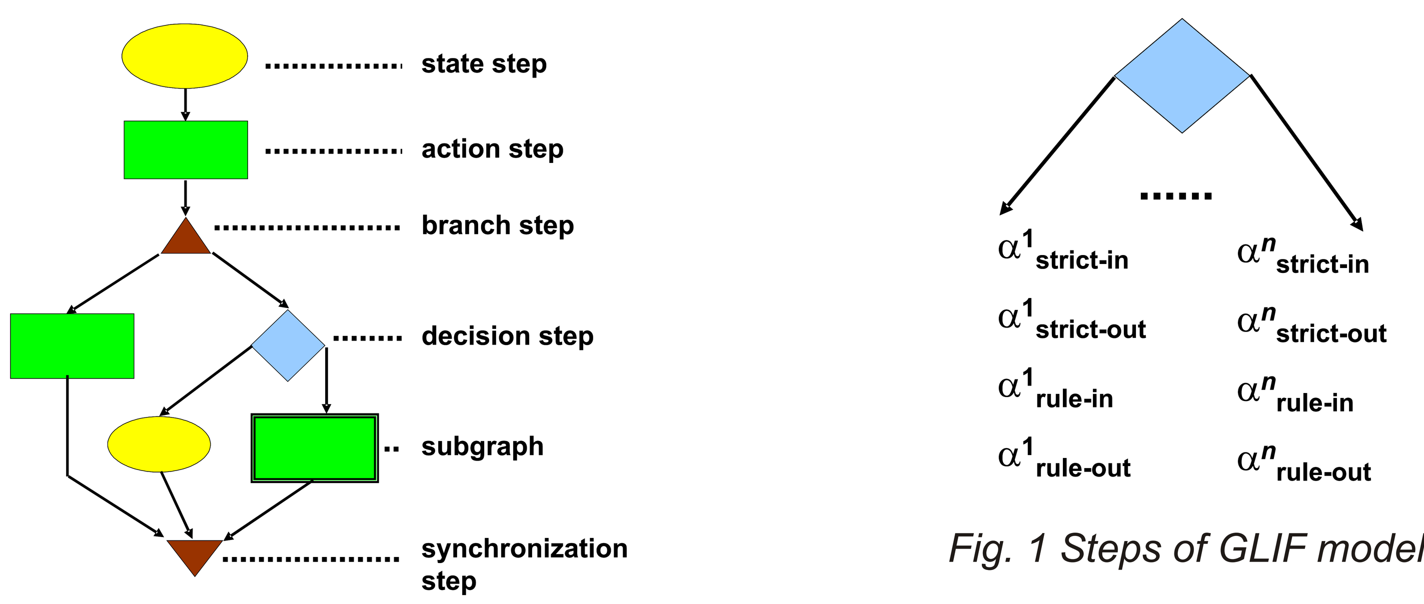

Fig. 1 Steps of GLIF model

Evaluation of conditions in the decision step with missing values in 3-valued logic

Assume $\alpha_i \in \{\alpha_1 \ldots \alpha_n\}$ is a condition built up from simple propositions (binary parameters) $P_1, \ldots, P_n$, $P=\{P_1, \ldots, P_n\}$. Assume $P_{k1}, \ldots, P_{kr} \in P$ are propositions with known truth value **t** and $P_{l1}, \ldots, P_{ls} \in P$ are propositions with known truth value **f**.

Assume we would like to know the following.

a) Condition α is valid whatever the truth values of the unknown simple propositions from P are.

b) Condition α is not valid whatever the truth values of the unknown simple propositions from P are.

c) neither a) nor b) is true.

Assume $P_1, \ldots, P_n$ are simple propositions of 3-valued Lukasiewiecz logic with truth values 0, ½, 1. Logical formulae are built up from simple propositions (propositional variables) $P_1, \ldots, P_n$ by means of logical operators (connectives) $\wedge, \vee, \neg$ and the truth values of compound formulae are evaluated using the following expressions

$\alpha \vee \beta = \max(\alpha, \beta)$

$\alpha \wedge \beta = \min(\alpha, \beta)$

$\neg\alpha = 1 - \alpha$

The truth value of a formula is interpreted according to an ontology Ω:

$\alpha = 1$ iff is valid in ontology Ω

$\alpha = 0$ iff is not valid in Ω

$\alpha = ½$ iff the validity of in is unknown Ω

An assignment $\omega(P_1, \ldots, P_n) = (\omega_1(P_1), \ldots, \omega_n(P_n))$, of truth values 0 or 1 to simple propositions $P_1, \ldots, P_n$ defines the state ω of ontology Ω.

An assignment $\tau(P_1, \ldots, P_n) = (\tau_1(P_1), \ldots, \tau_n(P_n))$, of truth values 0, 1, ½ to simple propositions $P_1, \ldots, P_n$ can be viewed as our incomplete knowledge about the ontology state ω:

a) If $\tau_i(P_i)=1$, then we know that P_i is valid in the state ω.

b) If $\tau_i(P_i)=0$, then we know that P_i is not valid in the state ω.

c) If $\tau_i(P_i)=½$, then we do not know if P_i is valid in the state ω or not.

We will say that ontology state $\varpi(P_1, \ldots, P_n)$ is compatible with an assignment $\tau(P_1, \ldots, P_n)$ if from $\tau_i(P_i)=1$ follows $\omega_i(P_i)=1$ and from $\tau_i(P_i)=0$ follows $\omega_i(P_i)=0$ for all $i=1, \ldots n$.

The definition of logical operators implies that if for an assignment $\tau(P_1, \ldots, P_n)$ the truth value $\alpha=1$, then α must be true in all ontology states compatible with $\tau(P_1, \ldots, P_n)$. Similarly if $\alpha=0$, then α cannot be valid in any ontology state compatible with $\tau(P_1, \ldots, P_n)$.

If the truth value of α is ½, the following cases are possible:

1. There exist compatible ontology states in which α is valid and there exist compatible ontology states in which α is not valid (e.g. if $\tau(P_1, P_2)=(1, ½)$, $\alpha=P_1 \wedge P_2 = ½$ then α is valid in the compatible state $\omega=(1, 1)$ and is not valid in the compatible state $\omega=(1, 0)$).
2. α is valid in all compatible ontology states (e.g. $\tau(P_1)=½$, $\alpha=P_1 \vee \neg P_1 = ½$).
3. α is not valid in any compatible ontology state (e.g. $\tau(P_1)=½$, $\alpha=P_1 \wedge \neg P_1 = ½$).

Assume the condition α and an assignment $\tau(P_1, \ldots, P_n)$ that describe our incomplete knowledge about ontology Ω are given. Obviously, the assertion about validity of α with regard to all possible assignments of truth values to missing values is equivalent to the assertion about validity of α in all ontology states compatible with assignment $\tau(P_1, \ldots, P_n)$. Thus for evaluation of condition with missing values the following algorithm can be used.

Algorithm

1. Evaluation of α in 3-valued logic.

Evaluate α in 3-valued logic:

a) If $\alpha=1$, α must be valid in Ω. Stop.

b) If $\alpha=0$, α cannot be valid in Ω. Stop.

c) If $\alpha=½$, go to the step 2.

2. Substitution $\alpha \to \alpha'$.

For all P_i, $i=1, \ldots, n$ do the following:

a) If P_i is contained in α and $\tau_i(P_i)=1$, substitute all occurrences of P_i with **t** (truth value **t**).

b) If P_i is contained in α and $\tau_i(P_i)=0$, substitute all occurrences of P_i with **f** (truth value **f**).

3. Minimization $\alpha' \to \alpha''$.

Use the following equivalences to minimize α':

$\neg\mathbf{t} \equiv \mathbf{f}$

$\neg\mathbf{f} \equiv \mathbf{t}$

$\gamma \vee \mathbf{t} \equiv \mathbf{t} \vee \gamma \equiv \mathbf{t}$

$\gamma \vee \mathbf{f} \equiv \mathbf{f} \vee \gamma \equiv \gamma$

$\gamma \wedge \mathbf{t} \equiv \mathbf{t} \wedge \gamma \equiv \gamma$

$\gamma \wedge \mathbf{f} \equiv \mathbf{f} \wedge \gamma \equiv \mathbf{f}$

4. Contradiction detection.

a) If $\neg\alpha''$ is contradiction of classical 2-valued logic, then α is valid for all possible assignments of truth values to missing values. Stop.

b) If α'' is contradiction of classical 2-valued logic, then α is not valid for any possible assignment of truth values to missing values. Stop.

c) If neither $\neg\alpha''$ nor α'' are contradictions, then for some possible assignments of truth values to missing values α is valid and for some of them α is not valid. Stop.

Whether formulas α'' or $\neg\alpha''$ are contradictions can be decided using the standard resolution algorithm or if the formulae have not many propositional variables, by means of truth table.

Conclusions

Assume $\alpha_i \in \{\alpha_1 \ldots \alpha_n\}$ is a condition of built up from simple parameters $P_1, \ldots, P_n$, $P=\{P_1, \ldots, P_n\}$. If the condition α_i **is fulfilled** independently **only on the known truth value** of P the designed decision algorithm can be used for evaluation of conditions even if it does not know truth values of all (unknown) parameters contained in the condition α_i. The method is based on **use of 3-valued logic** and is more suitable **for computer realizations.**

Reference

[1] Boxwala AA, Peleg M, Tu SW, Ogunyemi O, Zeng Q, Wang D, et al. GLIF3: A Representation Format for Sharable Computer-Interpretable Clinical Practice Guidelines. J Biomed Inform. 2004;37(3):147-61.

[2] Buchtela D, Peleska J, Vesely A, Zvarova J, Zvolsky M. Guideline Knowledge Representation Model (GLIKREM). European Journal for Biomedical Informatics, 2008, Volume 4 (2008) Issue 1. ISSN 1801-5603. Available at: http://www.ejbi.org/articles/200812/34/1.html.

[3] Vesely A, Zvarova J, Peleska J, Buchtela D, Anger Z. Medical Guidelines Presentation and Comparing with Electronic Health Record. International Journal of Medical Informatics 75 (3-4). 2006. 240-245.

Supported by the project 1M06014 ME CR.

Contact: Doc. Ing. Arnošt Veselý, CSc., vesely@euromise.cz
Institute of Computer Science AS CR, Dept. of Medical Informatics, Pod Vodárenskou věží 2,
182 07, Praha 8, Czech Republic