Temporal logics

as a tool for reasoning about dynamic systems

Temporální operátory ○, □ , ◊ se (anglicky) nazývají

○ *nexttime* nebo jen *next* ,

□ *always* nebo *henceforth* a

◊ *sometime.*

Formule ○*A* , □ *A* a ◊*A* se (anglicky) čtou

○***A*** : *nextA* , česky *příště A*,

□ ***A*** : *alwaysA*, česky *vždy A*

◊***A*** : *sometimeA*, česky *někdy A*

Preference ¬ , ○ **,** □ **,** ◊ váží silněji než ∨, ∧ , → a ≡ má nejslabší prioritu.

LTL Formal system

Let $\boldsymbol{A}$, $\boldsymbol{B}$ be any formulas of the language with temporal operators. The following sets of (valid) formulas constitute **axioms of LTL**:

- (**Taut**) contains instances of all propositional tautologies,
- **(LTL 1)** $\neg\circ A \leftrightarrow \circ\neg A$,
- **(LTL 2)** $\circ(A \rightarrow B) \leftrightarrow (\circ A \rightarrow \circ B)$,
- **(LTL 3)** $\Box A \leftrightarrow (A \wedge \circ\Box A)$.

Derivation rules of LTL:

- (**modus ponens**) “From A and $A \rightarrow B$ derive B.
- (**next**) “From $\circ A$ derive $\circ A$.
- (**indukce**) “From $A \rightarrow B$ and $A \rightarrow \circ A$ derive $A \rightarrow \Box B$.

Some formulas that can be derived in LTL

The LTL system is correct and complete. For example following valid formulas can be derived:

a) $\Box A \rightarrow \Diamond A$,

b) $\circ A \rightarrow \Diamond A$,

c) $\Box(A \rightarrow B) \rightarrow (\Box A \rightarrow \Box B)$,

d) $\Box(A \rightarrow B) \rightarrow (\Diamond A \rightarrow \Diamond B)$,

e) $\Diamond\Diamond A \leftrightarrow \Diamond A$

f) $\Box(B \rightarrow \circ A) \rightarrow \Box(B \rightarrow \Diamond A)$.

Description of a complex dynamic system – well functionning institution

- Any submission will be some time delivered to the proper place (responsible clerk) **□(*submition* → ◊*delivered*)**,
- When a submition is delivered its processing will start in the next instant **□(*delivered* → ○*processed*)**,
- Any processed application will be once decided and that decision will final (it will never be revised any more)

  □(*processed* → ◊□*ready*).

These formulas characterize rules of functioning for the considered complex dynamic system. Let us denote this set of formulas the program **P** describing this system. It seems, that there cannot occur a situation, when the requirement is submitted but it is never ready. Is it really so? Let us prove that the formula ***submition* & ¬ ◊□ *ready*** can never be become true, because it is inconsistent with the considered program **P** .

Every submition will become ready!

i. $\Box(submition \rightarrow \Diamond delivered)$,
ii. $\Box(delivered \rightarrow \circ\, processed)$,
iii. $\Box(processed \rightarrow \Diamond\Box ready)$.

c) $\Box(A \rightarrow B) \rightarrow (\Box A \rightarrow \Box B)$,
d) $\Box(A \rightarrow B) \rightarrow (\Diamond A \rightarrow \Diamond B)$,
e) $\Diamond\Diamond A \leftrightarrow \Diamond A$
f) $\Box(B \rightarrow \circ A) \rightarrow \Box(B \rightarrow \Diamond A)$
g) $\Box A \rightarrow A$.

The provable formula f) allows to substitute the rule ii by the rule

iv. $\Box(delivered \rightarrow \Diamond\, processed)$

All the considered rules have the same structure, namely $\Box(A \rightarrow \Diamond B)$.

The provable formula f) allows to substitute the rule ii by the rule

$\Box(delivered \rightarrow \Diamond\, processed)$

All the considered rules have the same structure, namely $\Box(A \rightarrow \Diamond B)$.

First we prove that such rules can be "connected" :

From validity of the formulas $\Box(A \rightarrow \Diamond B)$ and $\Box(B \rightarrow \Diamond C)$ the following formula can be proven: $(A \rightarrow \Diamond C)$.

1.	$\Box(B \rightarrow \Diamond C) \rightarrow (\Diamond B \rightarrow \Diamond\Diamond C)$	derived property d)
2.	$\Box(B \rightarrow \Diamond C)$	2^{nd} assumption
3.	$(\Diamond B \rightarrow \Diamond\Diamond C)$	MP (1, 2)
4.	$(A \rightarrow \Diamond B)$	g) for the 1^{st} assumption
5.	$(A \rightarrow \Diamond\Diamond C)$	taut. for 3 and 4
6.	$(A \rightarrow \Diamond C)$	

$submition \rightarrow \Diamond\Box ready$

Further resources:

- Huth M., Ryan M.: *Logic in Computer Science*, Cambridge University Press, 2004
- Michael Fisher: *Introduction into Formal Methods Using Temporal Logic*, John Wiley & Sons, 2011
- FIRST PhD Autumn School on Modal Logic, November 10-11 2009, materials from the courses *Temporal Logics for Specification and Verification* (V. Goranko), **http://hylocore.ruc.dk/m4m6school.html**
- Program system SPIN