SACS^2: Dynamic and Formal *S*afety *A*nalysis Method for *C*omplex *S*afety *C*ritical *S*ystem

Kwang Yong Koh* and Poong Hyun Seong
Department of Nuclear and Quantum Engineering, Korea Advanced Institute of Science and Technology, 371-1, Guseong-dong, Yuseong-gu, Daejeon, 305-701, Republic of Korea
**Corresponding author: goeric1@kaist.ac.kr*

1. Introduction

Fault tree analysis (FTA) is one of the most widely used safety analysis technique in the development of safety critical systems. However, over the years, several drawbacks of the conventional FTA have become apparent. One major drawback is that conventional FTA uses only static gates and hence can not capture dynamic behaviors of the complex system precisely. Although several attempts such as dynamic fault tree (DFT) [1], PANDORA [2], formal fault tree (FFT) [3] and so on, have been made to overcome this problem, they can not still do absolute or actual time modeling because they adapt relative time concept and can capture only sequential behaviors of the system. Second drawback of conventional FTA is its lack of rigorous semantics. Because it is informal in nature [4], safety analysis results heavily depend on an analyst's ability and are error-prone. Finally reasoning process which is to check whether basic events really cause top events is done manually and hence very labor-intensive and time-consuming for the complex systems [5].

In this paper, we propose a new safety analysis method for complex safety critical system in qualitative manner. As illustrated in Fig. 1, we introduce several temporal gates based on timed computational tree logic (TCTL) [6] which can represent quantitative notion of time. Then, we translate the information of the fault trees into UPPAAL query language and the reasoning process is automatically done by UPPAAL [7] which is the model checker for time critical system.

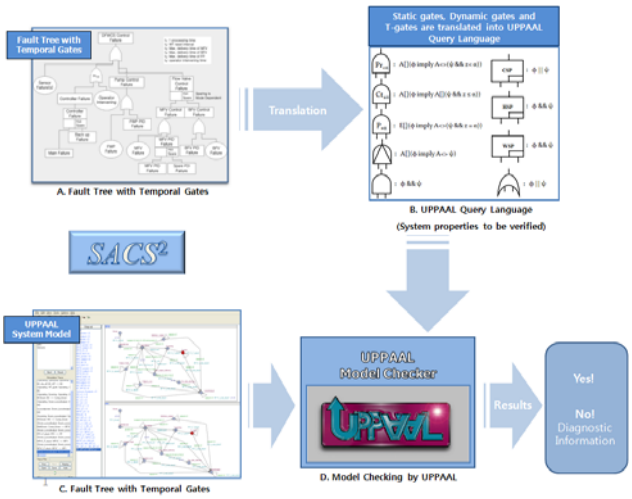


Fig. 1. Overview of proposed approach.

2. Definition of Fault Tree Gates

In this section we introduce new temporal gates based on TCTL to describe dynamic behaviors of system which changes its states as time goes on, and define the temporal gates, some dynamic gates and static gates in terms of 'Name', 'Symbol', 'semantic' and 'meaning'.

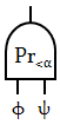
2.1 Definition of Temporal gates

We develop several temporal gates and make a concrete semantic for each of them based on TCTL syntax. The partial results are illustrated in Fig. 3.

Name: Promptness gate

Symbol: $Pr_{<\alpha}$

Graphical Notation:



Semantic: $\mathbf{AG}[\phi \rightarrow \mathbf{AF}_{<\alpha}\psi]$ (α is any rational number from $\mathbf{Q}$)
where $\phi, \psi ::= p \mid \alpha \mid \neg\phi \mid \phi \vee \psi \mid \mathbf{E}[\phi \mathbf{U} \psi] \mid \mathbf{A}[\phi \mathbf{U} \psi]$
$p \in AP$, atomic propositions,
$z \in D$, formula clocks,
$\alpha \in \Psi(C \cup D)$, constraints over formula clocks and automata clocks

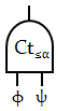
Meaning: ψ occurs within α time units after ϕ has occurred.

a) Promptness gate

Name: Continuity gate

Symbol: $Ct_{\geq\alpha}$

Graphical Notation:



Semantic: $\mathbf{AG}[\phi \rightarrow \mathbf{AG}_{\geq\alpha}\psi]$ (α is any rational number from $\mathbf{Q}$)
where $\phi, \psi ::= p \mid \alpha \mid \neg\phi \mid \phi \vee \psi \mid \mathbf{E}[\phi \mathbf{U} \psi] \mid \mathbf{A}[\phi \mathbf{U} \psi]$
$p \in AP$, atomic propositions,
$z \in D$, formula clocks,
$\alpha \in \Psi(C \cup D)$, constraints over formula clocks and automata clocks

Meaning: ψ continues for at least α time units after ϕ has occurred.

b) Continuity gate

Fig. 2. Definitions of some temporal gates.

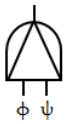
2.2 Definition of Dynamic gates

In other to use existing dynamic and static gates in the work, they should be represented in a form of computation tree logic (CTL). We redefine them in CTL and some of them are illustrated in Fig. 4.

Name: Priority-AND (PAND)gate

Symbol: PAND

Graphical Notation:



Semantic: **AG**[ϕ → **AF** ψ]
where ϕ, ψ ::= p | ¬ ϕ | ϕ ∨ ψ | **EX** ϕ | **E**[ϕ **U** ψ] | **A**[ϕ **U** ψ]
p ∈ AP, atomic propositions,

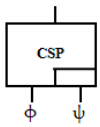
Meaning: ψ occurs at some point in time after ϕ has occurred.

a) PAND gate

Name: Cold Spare (CSP) gate

Symbol: CSP

Graphical Notation:



Semantic: ϕ || ψ (treated as 'or' gate in the proposed approach)
where ϕ, ψ ::= p | ¬ ϕ | ϕ ∨ ψ | **EX** ϕ | **E**[ϕ **U** ψ] | **A**[ϕ **U** ψ]
p ∈ AP, atomic propositions,

Meaning: Output occurs if and only if all spare events (inputs) occur.

b) CSP gate

Fig. 3. Definitions of some dynamic gates.

3. Translation from Fault Tree into UPPAAL Query Language

All the information of a fault tree is translated into UPPAAL query language for automatic verification. First, fault tree gates are translated to corresponding UPPAAL query language based on transition rules between TCTL and UPPAAL query language. In Fig. 4, corresponding expressions of fault tree gates are illustrated.

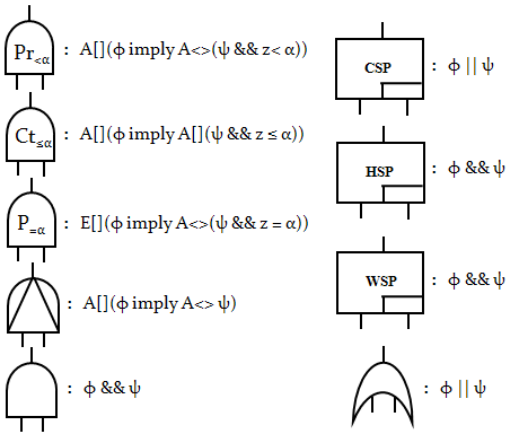


Fig. 4. Fault tree gates and their corresponding expressions in UPPAAL Query.

Digital feed-water control system (DFWCS) which is Benchmark system in [8] is analyzed with fault tree analysis and partial results of the analysis (partial fault tree) and its translated result are illustrated in Fig. 5.

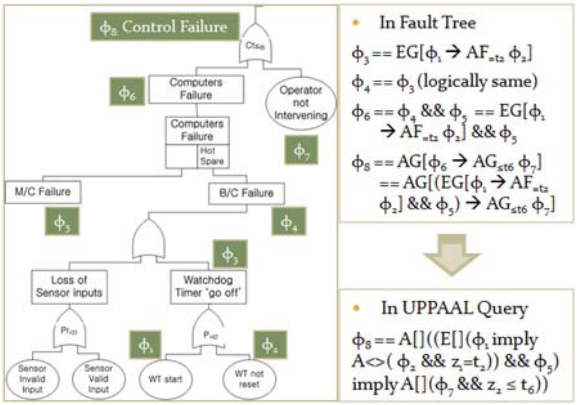


Fig 5. Exampled fault tree and translated result.

4. Conclusions

We demonstrated that the proposed approach was useful for providing formal, automated and qualitative assistance in informal safety analysis. Although a tool to automate our method is being under development, we expect that the method become promising to even large scale complex system with tool support.

REFERENCES

[1] J. B. Dugan, S. J. Bavuso and M. A. Boyd, Dynamic fault-tree models for fault-tolerant computer systems, IEEE Transactions on Reliability, Vol.41(3), p.363, 1992.
[2] M. Walker and Y. Papadopoulos, Synthesis and analysis of temporal fault trees with PANDORA: The time of Priority AND gates, Nonlinear Analysis: Hybrid Systems, Vol.2(2), p368, 2008.
[3] F. Ortmeier, W. Reif, G. Schellhorn, A. Thums, B. Hering, and H. Trappschuh, Safety analysis of the height control system for the Elbtunnel, Reliability Engineering and System Safety, Vol.81(3), p259, 2003.
[4] N.G. Leveson, Safeware: system safety and computers, Addison- Wesley, New York, 1995.
[5] F. Ortmeier and G. Schellhorn, Formal Fault Tree Analysis – Practical Experiences, Electronic Notes in Theoretical Computer Science, Vol.185, p139, 2007.
[6] B.Berard and et al., Systems and Software Verification Model-Checking Techniques and Tools, Springer-Verlag, Berlin Heidelberg, 2001.
[7] UPPAAL Help, version 4.0.7, released in November 2008. http://www.uppaal.com/
[8] T. Aldemir, M.P. and et al., Dynamic Reliability Modeling of Digital Instrumentation and Control Systems for Nuclear Reactor Probabilistic Risk Assessments, NUREG/CR-6942, U.S. NRC, Washington, D.C., 2007.