Fast Anticipatory Reasoning for Computing Anticipatory Systems

Takahiro Koh, Yuichi Goto, and Jingde Cheng
Department of Information and Computer Science, Saitama University,
255 Shimo-Okubo, Sakura-ku, Saitama, 338-8570, Japan
+81-48-858-3785 - {t_koh, gotoh, cheng}@aise.ics.saitama-u.ac.jp -
http://www.aise.ics.saitama-u.ac.jp/index-e.html

Abstract
Anticipatory reasoning engine is an indispensable component for computing anticipatory systems. In practical computing anticipatory systems, the efficiency of anticipatory reasoning engine is a key issue to satisfy the real-time requirements from applications. FreeEnCal: a forward reasoning engine for general-purpose purpose is a hopeful candidate for anticipatory reasoning engines. However, the current FreeEnCal is not efficient enough for practical computing anticipatory systems. This paper presents a new implementation of FreeEnCal improved by adopting fast algorithms, and shows its efficiency by comparing the improved FreeEnCal with the old one. The improved FreeEnCal can be used as an anticipatory reasoning engine of practical computing anticipatory systems.
Keywords : Computing anticipatory system, Anticipatory reasoning, Unification, Fast algorithm, Forward reasoning engine.

1 Introduction

The concept of an anticipatory system first proposed by Rosen in 1980s [25]. Rosen considered that “an anticipatory system is one in which present change of state depends upon future circumstance, rather than merely on the present or past” and gave a first definition of an anticipatory system as “a system containing a predictive model of itself and/or its environment, which allows it to change state at an instant in accord with the model’s prediction to a latter instant.” Until now, anticipatory systems have been discussed and developed by scientists from various disciplines [1, 4, 9, 10, 13, 23]. Dubois defined a computing anticipatory systems as “a system which computes its current states in taking into account its past and present states but also its potential future states” and introduced the concepts of strong and weak anticipation [11, 12].

Any computing anticipatory system must have the ability of anticipate future event or events whose occurrence and truth are uncertain at the point of the current time. An adaptive way to make a computing anticipatory system have the ability of anticipation is to develop an anticipatory reasoning engine for general-purpose working based on various logic systems to underlie temporal reasoning such as temporal logic systems [2, 14, 15, 16, 29] as its component because reasoning is the process to draw new conclusions from given premises which are already known facts or previously assumed hypothesis and temporal logics is used to the logic basis of reasoning with time-related propositions [5]. On the other hand, the efficiency of anticipatory reasoning engines is a key issue in practical computing anticipatory systems because a practical computing anticipatory system must be able to perform any anticipation to draw enough effective conclusions within an acceptable time to satisfy the real-time requirements from applications from the viewpoints of software reliability and information security [18]. However, there is no efficient anticipatory reasoning engine yet.

FreeEnCal [8], which is a forward reasoning engine for general purpose, is a hopeful candidate of anticipatory reasoning engines. From the viewpoint of generality, FreeEnCal can perform anticipatory reasoning based on various logic systems to underlie temporal reasoning. However, the current FreeEnCal is not efficient enough to draw enough effective conclusions within an acceptable time. Some approaches to improve the efficiency of forward reasoning engines were studied [19, 20]. The present authors have developed fast algorithms for the duplication checking process [19] and the derivation process [20]. As the results of these works, the execution time of these processes is dramatically shortened. Therefore, it is expected that the efficiency of FreeEnCal can be dramatically improved by adopting the two fast algorithms. However, there is no such implementation of FreeEnCal.

To implement a practical anticipatory reasoning engine for computing anticipatory systems, this paper presents a new implementation of FreeEnCal improved by adopting the two fast algorithms. The paper also shows that the improved implementation of FreeEnCal can be used as a practical anticipatory reasoning engine by comparing the improved implementation of FreeEnCal with the old one.

The rest of this paper is organized as follows: Section 2 explains about computing anticipatory systems and anticipatory reasoning engines. Section 3 explains FreeEnCal. Section 4 explains our methods to improve the efficiency of FreeEnCal. Section 5 presents some experimental results. Section 6 discusses our experimental results. Some conclusions are given in section 7.

2 Anticipatory Reasoning Engine

An anticipatory reasoning is a reasoning to draw new, previously unknown and/or unrecognized conclusions about some future event or events whose occurrence and truth are uncertain at the point of time when the reasoning is being performed [5]. Note that the above definition of anticipatory reasoning refers to a point of time to be considered as the reference time. Obviously, an anticipatory reasoning is meaningful only if it is done about a dynamic world where some thing changes along time [5].

Anticipatory reasoning engine is a key component to make an anticipation in computing anticipatory system. Anticipatory reasoning engine should be a forward reasoning engine with capable of reasoning with time-related propositions. Anticipatory reasoning should be forward rather than backward because when we perform an anticipatory reasoning, it is impossible to set some future event or events as a goal or sub-goal because occurrence and truth are uncertain at the time point of the reasoning is being performed because anticipatory reasoning is the reasoning to draw new, previously unknown and/or unrecognized conclusions about some future event or events whose occurrence and truth are uncertain at the point of time when the reasoning is being performed. Anticipatory reasoning engine should be capable of reasoning with time-related propositions because anticipation is the action to make some future event known in advance, especially on the basis of special knowledge. It is a notion must relate to a point of time to be considered as the reference time. For any anticipation, both the anticipated thing and its truth must be unknown before the completion of that anticipation. Thus, logic systems to underlie temporal reasoning are required as the logical basis of anticipatory reasoning engines [5]. On the other hand, it varies from person to person that which logic system/axiomtic system is a suitable for a target application. We think that temporal relevant logics [5] and their extensions are suitable for anticipatory reasoning, but someone does not think so. She/he may want to use other logic systems. Therefore, anticipatory reasoning engines should be capable of reasoning based on various logic systems to underlie temporal reasoning.

The first anticipatory reasoning engine was proposed and developed in 2006 [21]. It is based on an extended version of EnCal: an automated forward deduction system for general-purpose entailment calculus [3]. However, from the viewpoint of generality, the first anticipatory reasoning engine was not enough because it can deal with only logic systems with at most two temporal binary operators and six unary temporal operators, and inference rules of modus ponens and temporal generalization. It is impossible to add other temporal operators and inference rules to the first anticipatory reasoning engine because EnCal cannot not deal with operators and inference rules defined and given by users.

3 FreeEnCal: A Forward Reasoning Engine with General Purpose

FreeEnCal is a hopeful candidate to implement an anticipatory reasoning engine for computing anticipatory systems. FreeEnCal is a forward reasoning engine for general purpose. FreeEnCal can interpret and perform inference rules defined and given by its users, draw fragments of various classical and/or non-classical logic systems formalized as different formal systems, draw empirical theorems of various formal theories constructed based on various logic systems, and perform deductive, inductive, and abductive reasoning automatically [8, 17]. One of the greatest difference between EnCal and FreeEnCal is the ability to deal with operators, and inference rules defined and given by users. FreeEnCal can deal with temporal reasoning based on not only existing logic systems but also new logic systems to be proposed in the future.

FreeEnCal takes the followings as input data: a set of logical formula as premises, a set of inference rules specified by users and some termination conditions, and outputs all logical formulas deduced from the premises and/or deduced logical formulas under the termination conditions. FreeEnCal performs the following four processes repeatedly [8].

Inference rule selection process: it selects an inference rule from the set of given inference rules.

Derivation process: it deduces logical formulas by applying the selected inference rule to each ordered tuple of logical formulas given as premises and previously deduced logical formulas if the inference rule is appliable to the ordered tuple.

Duplication checking process: it finds all of the deduced logical formulas which are duplicated of a given premise or a previously deduced conclusion.

Adding process: it adds all unduplicated logical formulas into the set of previously deduced logical formulas.

The flowchart of FreeEnCal is shown in figure 1.

FreeEnCal for a practical computing anticipatory system must get enough effective conclusions in an acceptable time. However, the current FreeEnCal is not efficient enough because of the inefficiency of the derivation process and the duplication checking process [19, 20].

4 Improving the Efficiency of FreeEnCal

To improve the efficiency of FreeEnCal, we focused on the derivation process and the duplication checking process because these processes are the most time consuming processes. The intrinsic operation of those processes is unification, that is to check whether two logical formulas can be unified or not. In the derivation process, unification is used to check whether an inference rule is appliable to an ordered tuple of logical formulas. Refer [20] for the precise definition of the derivation process. In the duplication checking process, unification is used to check whether a deduced conclusion is a duplicate conclusion or not. Refer [19] for the precise definition of the duplication checking process.

FreeEnCal treats sets of logical fomulas. Thus, FreeEnCal checks a number of pairs of logical fomulas whether they can be unified. The current FreeEnCal is implement based on a naive algorithm. The naive algorithm works very slowly

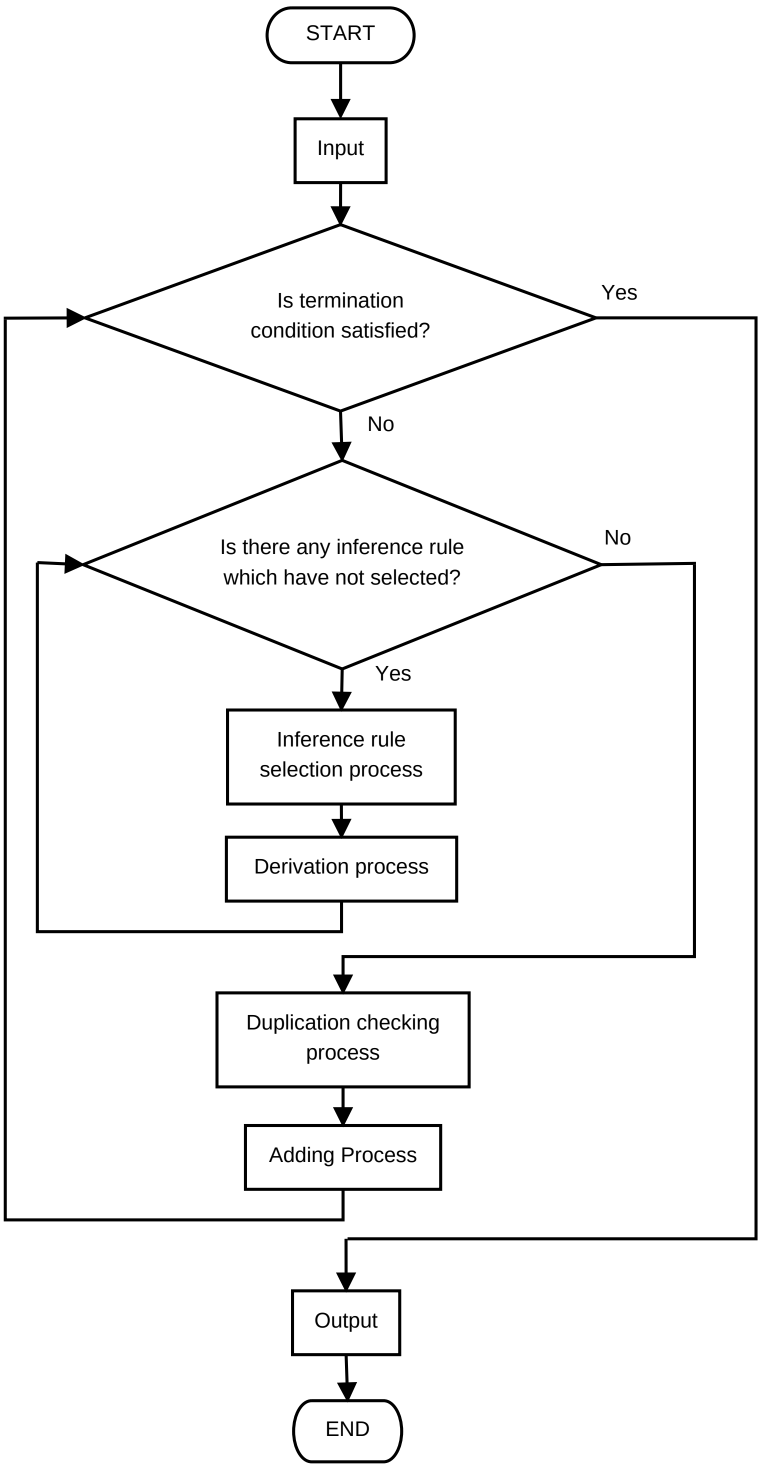
START
Input
Is termination condition satisfied?
Yes
No
Is there any inference rule which have not selected?
No
Yes
Inference rule selection process
Derivation process
Duplication checking process
Adding Process
Output
END

Fig. 1: The flowchart of FreeEnCal

because it performs a lot of unnecessary comparisons, that is, comparing the same prefixes of logical formulas in polish notation. There is no need to compare the same prefixes more than one time. For example, there are two sets of logical formulas $P = \{\rightarrow \vee abb, \rightarrow \vee ab \wedge ab\}$ and $Q = \{\rightarrow \wedge cdc, \rightarrow \wedge cd \vee cd\}$ where $\rightarrow$, $\vee$, and $\wedge$ are binary connectives and a, b, c and d are sentential variables. In this case, it should compare the first logical formula of P and the first logical formula of Q. It is clear that no more comparisons are needed because prefix "$\rightarrow \vee$" does not match to "$\rightarrow \wedge$". Therefore, every pair of logical formulas is not unifiable. However, the naive algorithm compares such same prefixes again and again. The key idea of our fast algorithms are reducing such unnecessary comparison [19, 20]. Refer [19] and [20] for the precise definition of the fast algorithm for the duplication checking process and the derivation process respectively. These approaches are a kind of branch and bounds method so that there is no difference between their time complexities. However, both of these algorithms redeced the execution time of the processes. Especially, the algorithm for the duplication checking algorithm dramatically reduced the execution time of the duplication checking process and it may no longer a time consuming process [19]. On the other hands, the algorithm for the derivation process reduced the execution time but it was not so dramatic like the one on the duplication checking process [20]. Therefore, these algorithms are useful to improve the efficiency of an anticipatory reasoning engine based on FreeEnCal.

5 Experimental Results

In order to achieve a practical anticipatory reasoning engine for a practical computing anticipatory system, we implemented FreeEnCal with the fast algorithms shown in section 4. To evaluate the efficiency of the improved FreeEnCal, we measured the execution time of the improved and the old FreeEnCal to deduce conclusions from axioms of some temporal logic systems and deontic logic systems on a PC server (DELL^{TM}PowerEdge^{TM}2950, dual Intel® Xeon® L5420 processor 2.5GHz with 32 GB main memory). We used logical formulas of Kt [2], LTL [22, 27], T_0Ec [5], DEc [28], and T_0DEc [6] as the premises. They are axiomatic systems of the temporal logic systems, linear temporal logic systems, temporal relevant logic systems, deontic relevant logic systems, and temporal deontic relevant logic systems respectively. In the experiment, adjunction which is an inference rule is excluded with T_0Ec, DEc, and T_0DEc because it is clearly meaningless for anticipation. In addition, we should put limitation on the number of deduced conclusions because the number of the deduced conclusions may become infinite. Therefore, we limited the degree of the nest of a logical connective [8] in a logical formula. Table 1 shows the limitation of the degrees for each experiment.

G, H, F, P_t, and N indicate future-tense always operator, past-tense always operator, future-tense sometime operator, past-tense sometime operator, next operator respectively. O and P_d indicate obligation operator and permission operator

Table 1: Limitations of the Degrees

	$\rightarrow$	$\wedge$	$\neg$	G	H	F	P_t	N	O	P_d
Kt	2	1	∞	1	1	1	1	-	-	-
LTL	2	∞	∞	1	-	1	-	∞	-	-
T_0Ec	2	1	1	1	1	1	1	-	-	-
DEc	2	1	∞	-	-	-	-	-	2	1
T_0DEc	2	1	1	1	1	1	1	-	1	1

respectively.

Table 2 shows the number of deduced conclusions.

Table 2: The Number of Deduced Conclusoins

	Deduced Conclusions
Kt	1,980
LTL	2,472
T_0Ec	11,627
DEc	18,865
T_0DEc	1,023,526

Table 3 shows the execution time of the improved and old FreeEnCal.

Table 3: Execution Time

	Improved (sec)	Old (sec)	Ratio
Kt	6.6	56.7	8.6
LTL	7.1	96.1	13.5
T_0Ec	91.4	1.23×10^3	13.6
DEc	96.7	4.82×10^3	49.9
T_0DEc	2.99×10^4	2weeks+	40.4+

The ratio in table 3 is calculated by dividing the execution time of the improved one into the old one. In the experiment, the improved FreeEnCal is about 8.6 to 49.9 times faster than the old one. The improved FreeEnCal works clearly faster than the old FreeEnCal in every axiomatic system. In case of T_0DEc, the improved FreeEnCal finished to draw conclusions in 8.5 hours. However, the old FreeEnCal did not finish to draw conclusions in 2 weeks.

6 Discussion

From the experimental results, the improved FreeEnCal is a hopeful candidate to implement a practical anticipatory reasoning engine. A practical computing anticipatory system must be able to perform any anticipation to draw enough effective conclusions within an acceptable time to satisfy the real-time requirements from applications from the viewpoints of software reliability and information security [18]. It is very difficult question what is the yardstick for judging about "enough effective conclusions." That depends on each application. The improved one may not effective enough for some applications because we do not have enough data to judge whether the improved one is enough effective for any computing anticipatory system. However, we can expect that the improved one works faster than the old one. The possibility to deduce an effective conclusion increase if the number of deduce conclusions increases. In general, the execution time of a forward reasoning engine gets longer if the more conclusions are deduced. From the experimental results, the ratio of the execution time between the improved and the old one is getting bigger if the number of deduced conclusions bigger. Because of the above two reasons, the improved one is a hopeful candidate of a practical anticipatory reasoning engine.

It is possible to consider that the improved FreeEnCal works efficiently with other logic systems to underlie temporal reasoning. The improved FreeEnCal is more efficient than the old one in any logic systems in the experiment. The experiment covers five different logic systems. The result of T_0Ec shows the improved FreeEnCal works efficiently with logic systems temporal relevant logics. The result of DEc shows the improved FreeEnCal works efficiently with logic systems of deontic relevant logics. The result of T_0DEc shows the improved FreeEnCal works efficiently with logic systems of temporal deontic relevant logics. Temporal deontic relevant logic systems can be considered as extensions of temporal relevant logics and deontic relevant logics. From these experimental results, we can expect that the improved FreeEnCal works efficiently with other logic systems of extensions of temporal relevant logics and deontic relevant logics such as 3D spatio-temporal relevant logics [7]. The result of Kt and LTL shows the improved FreeEnCal works efficiently with logic systems of temporal logics and linear temporal logics respectively. From these experimental results, we can expect that the improved FreeEnCal works efficiently with their extensions such as computational tree logics [24]. Therefore, the improved FreeEnCal works efficiently in case with various logic systems to underlie temporal reasoning.

7 Concluding Remarks

Anticipatory reasoning engine is an indispensable component for computing anticipatory systems. In practical computing anticipatory systems, the efficiency of anticipatory reasoning engine is a key issue to satisfy the real-time requirements from applications. The paper has presented a new implementation of FreeEnCal improved by adopting the two fast algorithms. The paper has also shown that the improved FreeEnCal works efficiently with various logic systems to underlie temporal reasoning from some experimental results. Therefore, the improved implementation of FreeEnCal can be a hopeful candidate of a practical anticipatory reasoning engine.

For the future works, to show the efficiency of the improved FreeEnCal in a practical computing anticipatory system, the study of computing anticipatory systems, we are going to implement a prototype of a computing anticipatory system such as an anticipatory reasoning reacting system [7, 26] using the improved FreeEnCal. On the other hand, parallel computing is a hopeful way to reduce the execution time of anticipatory reasoning engine because it is shown that parallel computing is effective for forward deduction engine for anticipatory reasoning [18]. Therefore, we are going to develop a parallel version of the improved FreeEnCal.

References

[1] Henrik Andersson and Jonas Hoglin: An Anticipatory Application for Water Regulation, in Daniel M. Dubois (Ed.), *Computing Anticipatory Systems: CASYS '03 - 6th International Conference*, Vol. 718 of *AIP Conference Proceedings*, pp. 516–524. The American Institute of Physics, 2004.

[2] John P. Burgess: Basic Tense Logic, in D. Gabbay and F. Guenthner (Eds.), *Handbook of Philosophical Logic: Volume II: Extensions of Classical Logic*, pp. 89–133. Reidel, Dordrecht, 1984.

[3] Jingde Cheng: EnCal: An Automated Forward Deduction System for General-Purpose Entailment Calculus, in N. Terashima and E. Altman (Eds.), *IFIP96 - 15th World Conference on IT Tools*, pp. 507–517. Chapman & Hall, 1996.

[4] Jingde Cheng: Anticipatory Reasoning-Reacting Systems, *Proceedings of International Conference on Systems, Development and Self-organization*, pp. 161–165, 2002.

[5] Jingde Cheng: Temporal Relevant Logic as the Logical Basis of Anticipatory Reasoning-Reacting Systems, in Daniel M. Dubois (Ed.), *Computing Anticipatory Systems: CASYS 2003 - Sixth International Conference.* AIP Conference Proceedings, 2004.

[6] Jingde Cheng: Temporal Deontic Relevant Logic as the Logical Basis for Decision Making Based on Anticipatory Reasoning, *Proceedings of 2006 IEEE Annual International Conference on Systems, Man, and Cybernetics*, Vol. 2, pp. 1036–1041. The IEEE Systems, Man, and Cybernetics Society, 2006.

[7] Jingde Cheng, Yuichi Goto, and Natsumi Kitajima: Anticipatory Reasoning about Mobile Objects in Anticipatory Reasoning-Reacting Systems, in Daniel M. Dubois (Ed.), *Computing Anticipatory Systems: CASYS 2007 - 8th International Conference*, Vol. 1051 of *AIP Conference Proceedings*, pp. 244–254. The American Institute of Physics, 2008.

[8] Jingde Cheng, Shinsuke Nara, and Yuichi Goto: FreeEnCal: A Forward Reasoning Engine with General-Purpose, *KES '07: Proceedings of the 11th International Conference on Knowledge-Based Intelligent Information and Engineering Systems*, Vol. 4693 of *Lecture Notes in Artificial Intelligence*, pp. 444–452. Springer-Verlag, 2007.

[9] Arshia Cont, Shlomo Dubnov, and Gerard Assayag: Anticipatory Model of Musical Style Imitation Using Collaborative and Competitive Reinforcement Learning, *Anticipatory Behavior in Adaptive Learning Systems*, Vol. 4520 of *Lecture Notes in Computer Science*, pp. 285–306. Springer-Verlag, 2005.

[10] Daniel M. Dubois: Introduction to Computing Anticipatory Systems, *International Journal of Computing Anticipatory Systems*, Vol. 2, pp. 3–14, 1998.

[11] Daniel M. Dubois: Review of Incursive, Hyperincursive and Anticipatory Systems, in Daniel M. Dubois (Ed.), *Computing Anticipatory Systems: CASYS '99 - 3rd International Conference*, Vol. 517 of *AIP Conference Proceedings*, pp. 3–30. The American Institute of Physics, 2000.

[12] Daniel M. Dubois: Mathematical Foundations of Discrete and Functional Systems with Strong and Weak Anticipations, *Anticipatory Behavior in Adaptive Learning Systems*, Vol. 2684 of *Lecture Notes in Computer Science*, pp. 107–125. Springer-Verlag, 2004.

[13] Bertil Ekdahl: Anticipatory Systems as Linguistic Systems, in Daniel M. Dubois (Ed.), *Computing Anticipatory Systems: CASYS '99 - 3rd International Conference*, Vol. 517 of *AIP Conference Proceedings*, pp. 131–140. The American Institute of Physics, 2000.

[14] Dov Gabbay, Marcelo Finger, and Mark Reynolds: *Temporal Logic: Mathematical Foundations and Computational Aspects*, Vol. 2, Oxford University Press, 2000.

[15] Dov Gabbay, Ian Hodkinson, and Mark Reynolds: *Temporal Logic: Mathematical Foundations and Computational Aspects*, Vol. 1, Oxford University Press, 1994.

[16] Antony Galton (Ed.): *Temporal Logics and their Applications*, Academic Press, London, 1987.

[17] Yuichi Goto, Takahiro Koh, and Jingde Cheng: A General Forward Reasoning Algorithm for Various Logic Systems with Different Formalizations, *KES '08: Proceedings of the 12th international conference on Knowledge-Based Intelligent Information and Engineering Systems, Part II*, pp. 526–535. Springer-Verlag, 2008.

[18] Yuichi Goto, Shinsuke Nara, and Jingde Cheng: Efficient Anticipatory Reasoning for Anticipatory Systems with Requirements of High Reliability and High Security, *International Journal of Computing Anticipatory Systems*, Vol. 14, pp. 156–171, 2004.

[19] Takahiro Koh, Yuichi Goto, and Jingde Cheng: A Fast Duplication Checking Algorithm for Forward Reasoning Engines, *KES '08: Proceedings of the 12th International Conference on Knowledge-Based Intelligent Information and Engineering Systems, Part II*, pp. 499–507. Springer-Verlag, 2008.

[20] Takahiro Koh, Yuichi Goto, and Jingde Cheng: A Fast Algorithm for Derivation Process in Forward Reasoning Engines, *International Journal of Computational Science*, Vol. 4, No. 3, pp. 219–231, 2010.

[21] Shinsuke Nara, Feng Shang, Takashi Omi, Yuichi Goto, and Jingde Cheng: An Anticipatory Reasoning Engine for Anticipatory Reasoning-Reacting Systems, *International Journal of Computing Anticipatory Systems*, Vol. 18, pp. 225–234, 2006.

[22] Amir Pnueli: The Temporal Logic of Programs, *SFCS '77: Proceedings of the 18th Annual Symposium on Foundations of Computer Science*, pp. 46–57. IEEE Computer Society, 1977.

[23] Marco A. Ramos, Alain Berro, and Yves Duthen: Distributed Anticipatory System, *Advanced Distributed Systems*, Vol. 3563 of *Lecture Notes in Computer Science*, pp. 443–451. Springer-Verlag, 2005.

[24] Mark Reynolds: An Axiomatization of Full Computation Tree Logic, *The Journal of Symbolic Logic*, Vol. 66, No. 3, pp. 1011–1057, 2001.

[25] Robert Rosen: *Anticipatory Systems - Philosophical, Mathematical and Methodological Foundations*, Pergamon Press, 1985.

[26] Feng Shang, Shinsuke Nara, Takashi Omi, Yuichi Goto, and Jingde Cheng: A Prototype Implementation of an Anticipatory Reasoning-Reacting System, in Daniel M. Dubois (Ed.), *Computing Anticipatory Systems: CASYS 2005 - 8th International Conference*, Vol. 839 of *AIP Conference Proceedings*, pp. 401–414. The American Institute of Physics, 2006.

[27] Aravinda P. Sistla and Edmund M. Clarke Jr.: The Complexity of Propositional Linear Temporal Logics, *Journal of the ACM*, Vol. 32, No. 3, pp. 733–749, 1985.

[28] Takahiro Tagawa and Jingde Cheng: Deontic Relevant Logic: A Strong Relevant Logic Approach to Removing Paradoxes from Deontic Logic, *PRICAI '02: Proceedings of the 7th Pacific Rim International Conference on Artificial Intelligence*, pp. 39–48. Springer-Verlag, 2002.

[29] Yde Venema: Temporal Logic, in Lou Goble (Ed.), *The Blackwell Guide to Philosophical Logic*, pp. 203–223, Oxford, 2001. Blackwell Publishers.