設計自動化 64－13
(1992. 10. 23)

リアルワールドコンピューティング用 知的フォールトトレラントシステムの構成

阿部 茂樹　　亀山充隆　　樋口 龍雄

東北大学工学部

〒980 仙台市青葉区荒巻字青葉

あらまし　　ディジタル制御システムの高安全性を達成するためには、プロセッサなどのハードウエアのフォールトにとどまらず、リアルワールドに関する情報を用いて制御対象の異常、人為的な操作ミス、外部環境などに起因するフォールトまでカバーすることが重要である。本報告では、システムが安全に動作するために必要なリアルワールドに関する知識を用いて、広い範囲のフォールトをカバーできる3重化に基づく知的フォールトトレラントシステムの構成法を検討している。冗長な2つのモジュールは、外界の情報を途絶し、あらかじめ保有する実システムに関する知識のみを用いてリアルワールドの状態を推定するシミュレータにより構成される。これにより共有する外界の情報を用いることなく各モジュール間の独立性を保てるため、高い安全性を有するシステムが実現できる。

和文キーワード　安全性, 知的フォールトトレランス, 多重化冗長システム, プロダクションシステム, シミュレータ, ディジタル制御

Design of an Intelligent Fault-Tolerant System for Real-World Computing

Shigeki ABE　　Michitaka KAMEYAMA　　Tatsuo HIGUCHI

Faculty of Engineering,　Tohoku University

Aoba,　Aramaki,　Aoba-ku,　Sendai 980,　Japan

Abstract　　To achieve the safety of intelligent digital system for real-world applications, not only the hardware faults in the processors but also any other faults and errors related to the real world such as sensor faults, actuator faults and human errors must be removed. From this point of view, an intelligent fault-tolerant real-world computing system is proposed based on triple-modular redundancy. The system consists of a master processor that performs the actual control operations and two redundant modules which simulate real-world process together with the control operations using knowledge-based inference strategy. To realize the independency between the triplicated modules, the simulation for error detection and recovery is performed without actual external sensor signals used in the master processor.

英文 key words　Safety,Intelligent fault tolerance,Multiplicated redundant system,P roduction system,Simulator,Digital control

1　まえがき

マイクロコンピュータなどを組み込んだエレクトロニクス機器の発展に伴い、FA、カーエレクトロニクス、ロボットシステムなどの応用においてディジタル制御システムのインテリジェント化が進行している。このような応用では、厳しい環境中で使用される場合が多く、故障や誤動作の発生が極めて重大な事態を引き起こす可能性がある。従って、その信頼性・安全性を向上させることは重要な問題となる [1] [2]。特に、故障や誤動作が生じた場合それを確実に検出し、少なくても危険状態に陥らないような安全なシステムを構築することが重要である [3]。

高安全化を実現する手法として、故障や誤動作が生じた場合でも、あらかじめ定められた安全な状態となるように設計されたフェールセーフシステムの有効性が知られており、信号機の制御や鉄道などに採用されている [4]。しかしながら、プロセッサなどのハードウエアのみに発生するフォールトを対象とする場合がほとんどであることや、安全な状態が固定化できる場合しか用いることができないなどフォールトをカバーできる範囲が狭いという問題がある。これに対し、さらにシステムの安全性を向上させるためには、プロセッサ内部のフォールトのみにとどまらずシステムが安全に動作するリアルワールドの情報を網羅し、制御対象の異常、人為的な操作ミス、外部環境などに起因するフォールトまでカバーできる総合的な安全システムを構成することが要求される。

本報告では、以上のような観点から、システムが安全に動作するために必要なリアルワールドに関する知識をも用いて、広い範囲のフォールトをカバーできるシステムの構成法について検討している。ここで、実際の入力信号、制御装置、制御対象、環境情報などに基づいたシステムの正常動作が既知あるいは予測できるような場合、あらかじめコンピュータにその情報を知識として格納できれば実際のシステムとの照合により異常状態を検出することができると考えられる [5]。この誤り検出法では、従来の０、１、論理値レベルでの一致・不一致による誤り検出とは異なり、”ある処理果に対し事象レベルの照合により、通常のものとは異なることを検出し、事前知識に従った妥当なものに回復させる”という処理の妥当性を考慮した事象レベルでの誤り検出が可能となる [6]。一例として、シーケンス制御を取り上げシミュレーションにより本システムの有効性を確認した結果を示している。

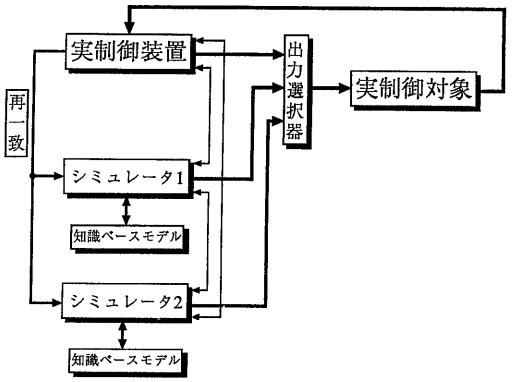


図 1: システム構成

2　システム構成

2.1　システム概要

入力信号、制御対象、環境情報など広い範囲のフォールトをカバーするためには、外界の情報を途絶し、あらかじめ保有する知識のみを用いてリアルワールドの状態を推定するシステムを個別に構成する必要がある。このシステムは、リアルワールドの情報に基づいて制御するシステムと同じ外界の情報を用いることがないため、外界の異常な情報に対する影響を避けることができる。また、多重化冗長システムを構成する場合最も重要な要素であるモジュール間の独立性を保つことにもなる。このことに着目した３重化に基づく知的フォールトトレラントシステムの構成を図1に示す。実制御装置と実制御対象からなる非冗長な実システムとそれの事前知識から作成したモデルに従って動作する２つのシ

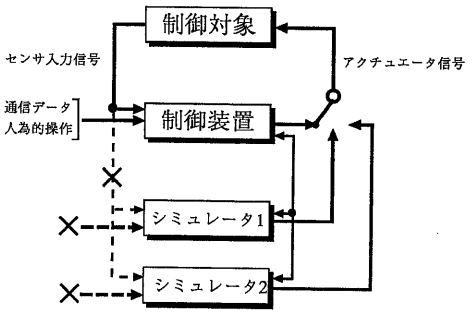


図 2: 3重化に基づく知的フォールトトレラントシステム

ミュレータから構成する。2つのシミュレータを用いている理由は、誤ったモジュールに対し誤りを検出するだけでなく、回復動作まで行う3重化システムを構成するためである。シミュレータの構成においても、N-バージョンプログラミングなどの手法を用いて独立性を保つ対策が必要であると考えられる [7]。

本システムの特長である広い範囲のフォールトをカバーできる原理を図2に示す。実システムは、リアルワールドの情報であるセンサ入力信号、人為的操作などに基づき制御アルゴリズムに従ってアクチュエータ信号を出力し制御対象を目標状態に変化させる。冗長な2つのシミュレータは、実システムが正常動作している時の入力信号、制御アルゴリズム、制御対象を数式や経験的知識を用いてモデル化し、実システムの動作をすべてソフトウェアに置き換えて構成する。このシステムの特徴は、シミュレータが外界の情報を途絶しあらかじめ内部に保有するモデルと知識のみを用いて実システムの入力信号、制御装置の内部状態やアクチュエータへの出力信号、制御対象の状態を推定する。このため、本構成では、各モジュールの処理を完全に独立化でき入力信号の誤り、制御対象の異常、人為的操作ミスなどのフォールトに対しても、システムの正常動作に関する知識を用いることによりカバーすることが可能となる。

しかしながら、実際のシミュレータは、実システムを正確にモデル化できない場合がほとんど

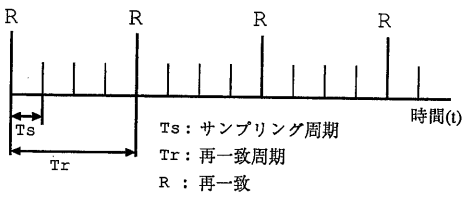


図 3: タイムチャート

であり、長時間外部情報を取り込まないままでいると、正常に動作している場合でもモデル誤差に起因するシミュレーション誤差や実システムの変動による影響を避けることができない。このため、すべてのモジュールが正常である場合、以後の処理に影響を与えるシミュレータの内部状態のすべてを図3のタイムチャートに示すように適当な周期で実システムの値と一致させることが必要となる。これを再一致(Resynchronization)と呼ぶ。本システムにおける再一致動作は、シミュレーション誤差によって正しい誤り検出が行われなくことを避ける重要な処理である。

2.2 誤り検出

入力信号および処理結果の誤り検出は、2つのシミュレータの推定値と、実システムの値との比較により行われる。ここで、シミュレータは実システムと同一のプロセッサを用いることを前提としているので、そのフォールトも考慮しなければならない。基本的動作は、あらかじめ設定された許容誤差ε_1に基づき実システムの状態rとシミュレータの状態iを比較する。その結果（1）式を満たす場合には、モデル誤差に起因するシミュレーション誤差であると見なし再一致を行い、（2）式の場合は誤動作が発生したものと見なしてシミュレータの値を用いて実システムの状態を回復する。

$$|r-i| \leq \varepsilon_1 \qquad (1)$$

$$|r-i| > \varepsilon_1 \qquad (2)$$

実システムの正常値をm、シミュレーション誤差をε_2とすれば（3）式が成立する。

$$|i-m| \leq \varepsilon_2 \quad (3)$$

システムが許容できる安全限界εが与えられているとすれば、(4)式の関係が成り立たなければならない。

$$|r-m| \leq \varepsilon \quad (4)$$

(1)、(3)、(4)式よりε_1とϵ_2の間には(5)式の関係が満たされなければならない。

$$|r-m| \leq |r-i| + |i-m| \leq \varepsilon_1 + \varepsilon_2 < \varepsilon \quad (5)$$

ここで、ε_1とϵ_2の間には(6)式が成り立たなければ、実制御システムが正常に動作しているにもかかわらず異常であると判断されることになる。

$$\varepsilon_1 \geq \varepsilon_2 \quad (6)$$

このような誤り検出により、どれか一つのモジュールが異常であると判断された時は一時フォールトに対し正常なモジュールの値を用いて状態回復をサンプリング周期内に行う。また、実システムに固定フォールトが発生した場合は、入出力をシミュレータ側に切換えて制御を行う。

3 シミュレータの構成

2つのシミュレータは、リアルワールドの動作に対しモデル誤差に起因するシミュレーション誤差をできるだけ小さくするように構築することが要求される。しかし、システムが大規模・複雑化するほど不完全情報や断片的知識を扱うことが多くなると予想され、完全にモデル化することが困難となる。このような場合、モデル化可能な部分と共に経験的な知識を用いて構築せざるを得ない。ここでは、図4で示されるようなプロダクションシステムで構成する。この利点としては、経験的・断片的知識や環境情報などを表現し易いことやルールの修正・追加・削除が簡単であるなどソフトウエア開発が容易であることが挙げられる。

制御装置や制御対象の動特性あるいは環境情報は、あらかじめプロダクションメモリ(PM)に格納する。この動作は、まず、シミュレータの現在の状態をワーキングメモリ(WM)に格納する。推論部では、WMのデータに基づき次にシミュータがどのような制御をするかPMから選択し、その情報をWMに返す。

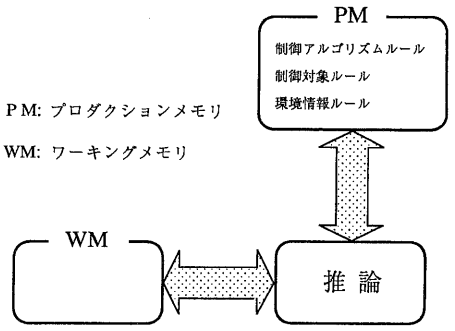


図4: シミュレータの構成

4 適用例

4.1 シーケンス制御

一例として、図5にシーケンス制御システムについて具体的に考察する。ここで、制御対象や制御対象に関する情報はプロダクションルールで表現しており、シミュレータを一貫してプロダクションシステムで構築するため制御アルゴリズムもプロダクションルールで表現している。このシステムの制御アルゴリズムは、以下のように与えられるものとする。

(1) 水槽1と水槽2は、上限センサ(S1,S3)がオンになるまで流入する。

(2) 水槽2は上限センサがオンになったらヒータをオンする。

(3) 温度センサがオンになったらヒータをオフする。

(4) 水槽1と水槽2は、同時に排出を開始する。

(5) 水槽1と水槽2は、下限センサ(S2,S4)がオフになるまで排出する。

(6) 水槽1と水槽2の両方の排出が停止したら、水槽3のミキサを動作させる。

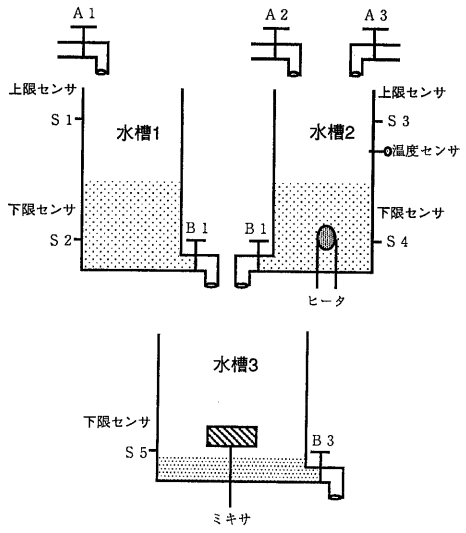


図 5: シーケンス制御モデル

表 1: 水槽 1 の時間的入力制限

時刻 t (hour)	0	1	2	3	4	5	6	7	8	9	10	11
流量 c (l/min)	2.5	2.5	2.5	2.0	2.0	1.5	1.0	1.0	1.5	2.0	2.0	1.5
時刻 t (hour)	12	13	14	15	16	17	18	19	20	21	22	23
流量 c (l/min)	1.0	1.0	1.5	1.5	2.0	1.5	1.5	1.0	1.0	1.5	2.0	2.0

(7) ミキサが１０分間動作したら停止する。

(8) 水槽３の下限センサがオフするまで排出する。

(9) 水槽３の排出が停止したら（１）にもどる。

このシステムでは、制御対象に関する情報として水槽１の流入量がおおよそ表1のように時間的変化することが経験的にわかっているものとする。

ここで、水槽１のシミュレータ動作は、表２のプロダクションルールで表される。但し、*は0,1 の任意の値である。今、水位の上昇フラグ $up1$、下降フラグ $dw1$ を以下のように定める。

$$\begin{cases} up1 = 1 \text{ かつ } dw1 = 0) & \text{水位上昇中のとき} \\ up1 = 0 \text{ かつ } dw1 = 1) & \text{水位下降中のとき} \end{cases}$$

表 2: 水槽１の制御アルゴリズムに関するルール

	up1	dw1	S1	S2	dw2	up3	A1	B1
R1	1	0	0	*	*	*	1	0
R2	1	0	1	*	*	*	0	0
R3	0	0	1	*	*	*	0	0
R4	0	1	*	1	0	*	0	0
R5	0	1	*	1	1	0	0	0
R6	0	1	*	1	1	1	0	1
R7	0	1	*	1	*	1	0	1
R8	0	1	*	0	*	1	0	0
R9	0	0	*	0	*	1	0	0
R10	1	0	0	0	*	0	0	0

まず、水槽間に依存がない処理、例えば現在のワーキングメモリの情報が水位が上昇中で上限センサ (S1) がオフの時, 制御アルゴリズムルールは表2より [R 1]

- if($up1$=1 & $dw1$=0 & S1=0) then {A1=1,B1=0}

が選択される。ここで、A1,B1 はアクチュエータ信号である。制御対象ルールは、表１の入力制限があることから現在の時刻を考慮し、水位の変化に関するルールを時刻毎に作成する必要がある。現在、時刻（t＝８）とすると

- if(A1=1 & B1=0 & t=8) then y=y+1.5

が選ばれ水位 y を更新する。また、センサ状態は、水位 y の値に基づき更新される。下限センサは y=20、上限センサは y=120 で変化するものとすると以下のルールが作成される。

- if(y $<$ 20) then S1=0,S2=0
- if(y $\geq$ 20 and y $<$ 120) then S1=0,S2=1
- if(y $\geq$ 120) then S1=1,S2=1

次に、水槽間に依存関係がある上限センサ (S1) がオン状態になり下降状態に移行する場合を考える。このシステムでは、制御アルゴリズム（４）より水槽１と水槽２は同時に排出しなければならないことや排出するためには水槽３が

それを許可している状態であることなど水槽間で制約条件が必要となる。そこで、水槽2の下降状態($dw2$=1)、水槽3の排出許可状態($up3$=1)を条件として加えると、この時の制御アルゴリズムルールは[R 6]

- if($up1$=0 & $dw1$=1 & S2=1 & $dw2$=1 & $up3$=1) then {A1=0,B1=1}

が選択される。

また、水槽2の動作でヒータにより液体を一定温度上昇させる必要がある場合、シミュレータではヒータがオンしてから目標温度に到達する経過時間に関する事前知識から以下のようなルールを構築することも必要となる。例えば、ヒータの状態を heat(0,1) とし、ヒータがオンしてからの経過時間を th とする。

- if($heat$=1 & th > 10) then {$heat$=0}
- if($heat$=1 & th < 10) then {$heat$=1}

これにより、温度センサの状態を予測できると共に、温度センサの異常状態を検出することも可能となる。

以上のように推論部では、WM の状態に基づき PM に保有する条件を満たすルールを順番に実行し WM を更新する。一般に、シミュレータを構築する時には、水槽間の制約条件や時間情報などを用いてルール化することが必要となる。

4.2 シミュレーション結果

シーケンス制御システムを用いて、本構成法の有効性をシミュレーションにより示す。実システムには、実際発生すると考えられる変動としての乱数を発生させ、それをシステムに加えている。ここでのシミュレータの値は、シミュレータ1のみを示すことにする。図6 (a) は、水槽1の再一致しない実システムとシミュレータの水位変化で、図6 (b) は、30周期毎に再一致を行った場合の結果である。これより、時間が経過するに従って水位の変化にずれを生じるのに対し、再一致により実システムと大きなずれを生じることなく制御できることがわかる。従って、すべてのモジュールが正常動作している場合、再一致を行うことによりいずれかにフォールトが発生しても正しく誤り検出を実行することができる。本構成では、サンプリング周期毎に誤り検出を実行することで、従来の3重化システムと同様にプロセッサ内部の一時的フォールトをカバーすることができる。図7 (a) は、通

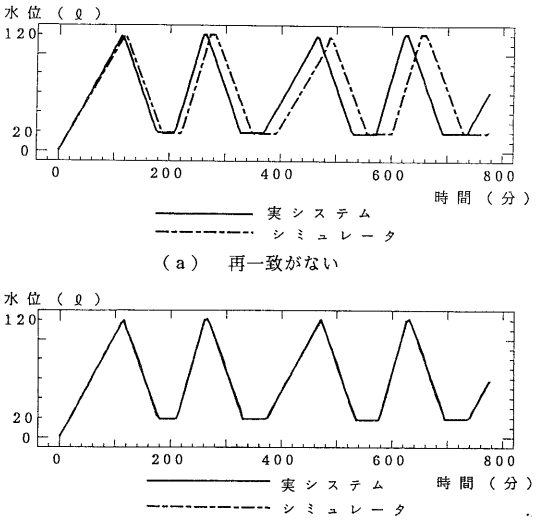
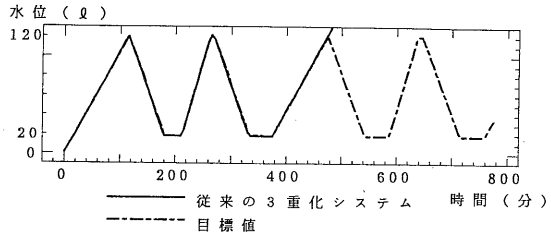


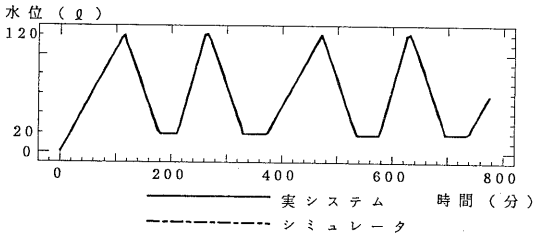
図6 水槽1の水位変化

常の3重化システム実システムの水槽1の上限センサ (S1) に固定的なフォールト (オフ状態) が発生した場合の水位の変化であり、図7 (b) は、本構成法によりシミュレータが固定的なフォールトを検出し、アクチュエータ信号をシミュレータ側に切り換え、シミュレータの知識のみによって制御した時の水位変化である。本システムでは、上限センサの故障で水槽から液体が溢れて危険状態に陥ることを、シミュレータの推定結果に置き換えることにより回避することが可能となる。また、図8 (a) は、通常の3重化システムで温度センサに故障が発生した場合、危険状態に陥る水槽2の温度変化を示している。これに対し図8 (b) は、本システムにおいてシミュレータの保有する時間情報を用いて制御した場合の温度変化である。これらのシミュレーション結果から、本システム構成によりプロセッサ

内のフォールトのみだけではなく入力信号自体の誤り、制御対象の異常までもが検出できることがわかる。

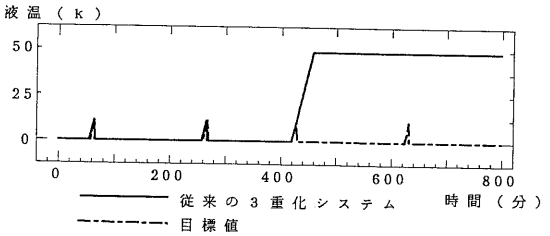


（a）従来の3重化システム

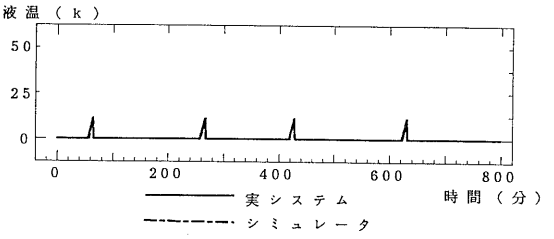


（b）本システム

図7　上限センサ（S1）に発生した固定フォールトの影響による水位変化



（a）　従来の3重化システム



（b）　本システム

図8　温度センサ故障による液温変化

5　複数種類のセンサを用いた多重化方式への拡張

本構成では、外界情報を途絶しているためあらかじめリアルワールドの情報を得る必要がある。しかしながら、リアルワールドでは、あらかじめ次の状態が既知あるいは予測できる制御だけではなく、現在の状況を把握するため逐次外部情報を取り入れることができるシステム構成が重要となる。

本システムの応用として、シミュレータを外部情報から途絶するのではなく、シミュレータにも各々にセンサを持たせることにより独自に外部情報を得て、それに基づきシミュレータを起動させるシステムを考える。一例として、安全車への応用で”信号のある交差点を右折する”ことを考える。

以下のように初期状態がセンサ情報により与えられる。

(初期状態)

場所 ： 交差点

信号機の状態： 青

対向車の有無： 無

歩行者の有無： 無

シミュレータのプロダクションメモリは以下の知識をもっているとする。

- ルール（1） if 赤信号か黄信号 then 停止
- ルール（2） if 青信号 then 進行
- ルール（3） if 交差点 then 右折
- ルール（4） if 右折 then 右方向指示ランプ
- ルール（5） if 右折かつ進行 then ハンドル右回転
- ルール（6） if 右折かつ対向車 then 停止
- ルール（7） if 右折かつ歩行者 then 停止

推論部では、初期状態より

ルール（2）⇒ルール（3）⇒ルール（4）⇒

ルール（５）の順でルールを適用し、安全に右折できるようにサポートする。すなわち、運転者が赤信号を見落としたり、対向車や歩行者の確認ができなかった場合など、人為的操作ミスに対し警告を出すなどして安全性を高めることができる。

6 むすび

28 多重化冗長システムを構成する場合、各モジュール間の独立性を高めることが重要である。そこで、冗長モジュールがシミュレータで構成された３重化システムの構成法を示した。この構成では、２つの冗長なモジュールは、外部情報をある一定の周期途絶しているため、その間は各モジュールを完全独立にできる。従って、入力信号の誤り、制御対象の異常、人為的な操作ミスまでもカバーできることをシミュレーションにより確認できた。

今後、本システムを実際のシステムに応用した場合の具体的評価を行うことが重要となる。さらに、シミュレータの保有する知識が危険の限界値でしか与えられないような制御対象などへの適用を検討していきたい。

参考文献

[1] 亀山、樋口："マイクロコンピュータの高信頼化", 計測と制御,24,4,pp.319-324(1985).

[2] 当麻:"システムの安全性, 高信頼性の考え方と評価", 計測と制御,24,4pp.290-295(1985).

[3] 奥村："フェールセーフシステムの構成", 電子通信学会誌,62,2,pp.198-205(1979).

[4] Akita,Nakamura：
"SAFTY AND FAULT-TOLERANCE IN COMPUTER-CONTROLLED RAILWAY SIGNALLING SYSTEMS",Dpendable Computing and Fault-Tolerant Systems,Vol.4,pp.107-131.

[5] 亀山、樋口：'" 知的フォールトトレランス", 電子情報通信学会誌,Vol.73,No.11,1990.

[6] 亀山、鄭、樋口：'" 知識工学的手法に基づくフォールトトレラントシステムのの構成とディジタル制御システムへの応用", 電子情報通信学会論文誌,Vol.70,No.12,pp.2679-2706,1987.

[7] L.Chen and A.Avizienis："N-version programming : A fault-tolerance approach to reliability of software operation",FTCS-8,pp.3-9(1978).

盛光印刷所