数理解析研究所講究録
1489巻 2006年 223-228

非同期分散システムにおける故障検知器と故障計数器について

坂田 敦 (Atsushi Sakata) *　　小野 廣隆 (Hirotaka Ono) †
定兼 邦彦 (Kunihiko Sadakane) †　　山下 雅史 (Masafumi Yamashita) †

* 九州大学大学院システム情報科学府　† 九州大学大学院システム情報科学研究院
*†Depatment of Computer Science and Communication Engineering,
Graduate School of Information Science and Electrical Engineering, Kyushu University

1 はじめに

分散システムにおいて，プロセスの故障はをいかに扱うかは，高信頼システム・アルゴリズム設計における重要な問題であり，実際分散システムの耐故障性を向上させるための研究が盛んに行われている．

分散システムの耐故障性を議論する際によく注目されるのが非同期分散システムである．非同期分散システムにおいてはプロセスの故障を考慮すると多くの分散問題は決定性アルゴリズムでは解けないことが知られている [3].

ここで重要なのは，[3] で定義されている非同期分散システムではプロセスの処理速度や通信遅延など時間に関する制約が全くされていないという点である．問題の非可解性はこのようなシステムにおいては故障しているプロセスと「遅い」プロセスとを見分けることができないことに起因する．しかし，実際のシステムではプロセスの処理速度や通信遅延に関する制約を用いることにより，故障したプロセスをある程度検知できるため，上記の仮定では非可解とされた問題も，事実上可解となることがある．このような故障プロセスに関する検知情報を抽象化し議論するために提案されたのが故障検知器である [2].

故障検知器はどのプロセスが故障しているかという情報を与えるオラクルである．ただし，上述のような動機より，故障検知器は必ずしも正確な故障状況を知らせる必要は無い．[2] では故障検知器はその性能によってクラス分けがされている．どのクラスの故障検知器を用いれば問題が解けるかが分かれば，そのような故障検知器を実装できれば問題が解けるということになる．そのため，対象となる分散問題を解くために必要かつ十分な「最弱」の故障検知器に関する考察が理論的，また実装的興味から盛んになされている ([1], [4] など).

本研究では，最弱性の観点から故障計数器を提案し，その性能と性質について考案する．故障検知器が故障と疑われるプロセスのIDを出力するのに対し，故障計数器はIDでは無く故障数のみを出力する．この故障計数器に故障検知器と同様に性能を定義し，これと故障検知器を比較する．それにより，故障検知器では定義されない新たなクラスの有無を調べる．

2 準備

以下では取り扱う非同期分散システムのモデル，故障検知器の定義について述べる．

2.1 モデル

本節の定義の多くは [2] に基づく．

2.1.1 非同期分散システム

この論文で扱われている非同期分散システムのモデルは以下の通りである．システムは n 個のプロセスの集合 $\Pi = \{p_1, p_2, \cdots, p_n\}$ で構成される．プロセスは計算を行うプログラムをモデル化したものであり，その実行速度には下限が無いものとする．任意の2プロセス間に通信路が存在し，プロセス同士はこの通信路上でメッセージをやり取りし通信を行う．送られたメッセージはかならず届くが遅延時間に上限は無いとする．

説明のために大域時計 $\mathcal{T} = \{0, 1, 2, \cdots\}$ を考える．各プロセスはこれを参照できない．

2.1.2 故障

プロセスは停止故障を起こし，修復はされないものとする．故障パタン F は $\mathcal{T}$ から 2^{Π} への関数であり，$F(t)$ は時刻 t において故障しているプロセスの集合を表す．$crashed(F) = \bigcup_{t\in\mathcal{T}} F(t)$ で F において故障を起こすプロセスを表す．また，$correct(F) = \Pi - crashed(F)$ で F において故障しないプロセスを表す．ただし，$correct(F) \neq \emptyset$ とする．

2.2 故障検知器

2.2.1 定義

故障検知器履歴 H_D は $\Pi\times\mathcal{T}$ から 2^{Π} への関数で，$H_D(p,t)$ はプロセス p が時刻 t に疑っているプロセスの集合を表す．$p \neq q$ であるならば，$H_D(p,t) \neq H_D(q,t)$ であってもよい．

故障検知器 $\mathcal{D}$ は各故障パタン F から $\mathcal{D}(F)$ への関数である．$\mathcal{D}(F)$ は故障パタン F において故障検知器から出力され得るすべての故障検知器履歴の集合を表す．

故障検知器を実装に関して定義するには通信遅延時間などネットワークの性質について言及しなければならないため，故障検知器を完全性 (completeness) と正確性 (accuracy) の2点で特徴づける．そうすることによって，議論を抽象化することができる．

2.2.2 故障検知器の性質

2つの完全性と4つの正確性を定義する ([2]).

Strong Completeness: すべての *correct* プロセスがいつかはすべての故障するプロセスをずっと疑うようになる．
$\exists t \in \mathcal{T}$, $\forall i \in crashed(F)$, $\forall j \in correct(F)$, $\forall t' \geq t$: $i \in H_D(j,t')$

Weak Completeness: いつかはすべての故障するプロセスをずっと疑うようになる *correct* プロセスが存在する．
$\exists t \in \mathcal{T}$, $\forall i \in crashed(F)$, $\exists j \in correct(F)$, $\forall t' \geq t$: $i \in H_D(j,t')$

Strong Accuracy: 故障する前に疑われるプロセスは存在しない．
$\forall t \in \mathcal{T}$, $\forall i,j \in \Pi - F(t)$: $i \in H_D(j,t)$

Weak Accuracy: 決して疑われない *correct* プロセスが存在する．
$\exists i \in correct(F)$, $\forall t \in \mathcal{T}$, $\forall j \in \Pi - F(t)$: $i \notin H_D(j,t)$

Eventual Strong Accuracy: いつかは Strong Accuracy を満たすようになる．

Completeness	Accuracy			
	Strong	Weak	ES	EW
Strong	$\mathcal{P}$	$\mathcal{S}$	$\Diamond\mathcal{P}$	$\Diamond\mathcal{S}$
Weak	$\mathcal{Q}$	$\mathcal{W}$	$\Diamond\mathcal{Q}$	$\Diamond\mathcal{W}$

表 1: 故障検知器のクラス

Eventual Weak Accuracy: いつかは Weak Accuracy を満たすようになる．

2.2.3 故障検知器のクラスと強さ

2つの完全性と4つの正確性の組合せによって8つの故障検知器のクラスを定義する．それぞれの組合せに対応するクラスを表1にまとめる．

故障検知器の強弱関係は次のように定義される．あるクラス $\mathcal{D}$ の故障検知器をクラス $\mathcal{D}'$ の故障検知器に変換するアルゴリズムが存在するなら，かつその時に限り，$\mathcal{D}'$ は D より弱いと言い $\mathcal{D} \succeq \mathcal{D}'$ で表す．$\mathcal{D} \succeq \mathcal{D}'$ かつ $\mathcal{D}' \succeq \mathcal{D}$ である時，$\mathcal{D}$ と D' は同等であると言い，$\mathcal{D} \cong \mathcal{D}'$ で表す．

定理 1 $[2]\mathcal{Q} \succeq \mathcal{P}, \mathcal{W} \succeq \mathcal{S}, \Diamond\mathcal{Q} \succeq \Diamond\mathcal{P}, \Diamond\mathcal{W} \succeq \Diamond\mathcal{S}$. □

系 1 $[2]\ \mathcal{Q} \cong \mathcal{P}, \mathcal{W} \cong \mathcal{S}, \Diamond\mathcal{Q} \cong \Diamond\mathcal{P}, \Diamond\mathcal{W} \cong \Diamond\mathcal{S}$. □

3 故障計数器

故障検知器は故障と疑われるプロセスの ID 集合を与えるが，対象となる問題を解くために必要なのは故障したプロセスの ID 情報であるとは限らない場合が考えられる．そこで，故障と疑われるプロセスの数のみを与えるような故障計数器を定義する．

3.1 定義

故障計数器履歴 H_C は $\Pi\times\mathcal{T}$ から $\{0,1,\ldots,n-1\}$ への関数で，$H_C(p,t)$ はプロセス p が時刻 t に疑っている故障の数を表す．$p \neq q$ であるならば，$H_C(p,t) \neq H_C(q,t)$ であってもよい．

故障検知器 $\mathcal{C}$ は各故障パタン F から $\mathcal{C}(F)$ への関数である．$\mathcal{C}(F)$ は故障パタン F において故障検数器から出力され得るすべての故障計数器履歴の集合を表す．

Completeness	Accuracy			
	Strong	ES	*A	+A
Strong	$\mathcal{P}_\sharp$	$\Diamond\mathcal{S}_\sharp$	$\mathcal{I}$	$\mathcal{J}$
Weak	$\mathcal{Q}_\sharp$	$\Diamond\mathcal{Q}_\sharp$	-	-

表 2: 故障計数器のクラス

3.2 故障計数器の性質

2つの完全性と4つの正確性を定義する.

Strong Completeness: いつかはすべての *correct* プロセスが *crash* するプロセス数をずっと数え上げる.

$\exists t \in \mathcal{T}, \forall i \in correct(F), \forall t' \geq t : H_C(i,t') \geq |crashed(F)|$

Weak Completeness: いつかはいくつかの *correct* プロセスが *crash* するプロセス数をずっと数え上げる.

$\exists t \in \mathcal{T}, \exists i \in correct(F), \forall t' \geq t : H_C(i,t') \geq |crashed(F)|$

Strong Accuracy: 実際の *crash* 数より多く数えているプロセスは存在しない.

$\forall t \in \mathcal{T},\ \forall i \in \Pi - F(t)\ :\ H_C(i,t) \leq |F(t)|$

Eventual Strong Accuracy: いつかは実際の *crash* 数より多く数えているプロセスは存在しなくなる.

$\exists t \in \mathcal{T}, \forall i \in correct(F), \forall t' \geq t : H_C(i,t) \leq |F(t)|$

***Accuracy:** 実際の *crash* 数より多く数えるプロセスが存在するなら, *correct* プロセスかつせいぜい1つ.

$\exists t \in \mathcal{T}, \exists i \in \Pi, H_C(i,t) > |F(t)| \Rightarrow i \in correct(F), \forall t' \in \mathcal{T},\ \forall j \in \Pi \setminus \{i\}\ :\ H_C(j,t') \leq |F(t')|$

+Accuracy: 実際の *crash* 数より多く数えない *correct* プロセスが存在する.

$\forall t \in \mathcal{T},\ \exists i \in correct(F)\ :\ H_C(i,t) \leq |F(t)|$

3.3 故障計数器のクラス

2つの完全性と4つの正確性の組合せによる故障計数器のクラスを定義する. それぞれの組合せに対応するクラスを表2にまとめる. 故障計数器の強弱関係は故障検知器と同様に定義される.

4 故障検知器と故障計数器の比較

ここでは, 3節で定義した故障計数器のクラスがどの程度の強さを持っているかを故障検知器との比較により述べる. その比較結果を表3にまとめた. 以下では表中の強弱関係を証明する.

計数器	強弱	検知器	計数器	強弱	検知器
$\mathcal{P}_\sharp$	$\cong$	$\mathcal{P}$	$\mathcal{I}$	$\succ$	$\mathcal{S}$
$\mathcal{Q}_\sharp$	$\cong$	$\mathcal{Q}$	$\mathcal{J}$	$\prec$	$\mathcal{P}$
$\Diamond\mathcal{P}_\sharp$	$\cong$	$\Diamond\mathcal{P}$	$\mathcal{J}$	$\succ$	$\Diamond\mathcal{P}$
$\Diamond\mathcal{Q}_\sharp$	$\cong$	$\Diamond\mathcal{Q}$	$\mathcal{J}$	$\not\succeq$	$\mathcal{S}$
$\mathcal{I}$	$\prec$	$\mathcal{P}$			

表 3: 故障検知器と故障計数器の比較

/全てのプロセスが以下を実行/
/初期化/
(1) 任意の $i \in \Pi$ について, $output_i := \emptyset$;
/点呼/
while(){
(2)$k := H_C(i,t)$;
(3) すべての $j \in \Pi$ に対して $message(i,k)$ を送る.
(4)$n-k$ 個の $reply(j,k)$ が届くまで待つ.
(5) 届いたら, *reply* を送ってこなかった k 個のプロセスの集合を $output_i$ とする. }
/割り込み処理/
(6) すべての $j \in \Pi$ は $message(i,k)$ を受け取ったら i に対して $reply(j,k)$ を返す.

図 1: アルゴリズム 1

4.1 クラス $\mathcal{P}_\sharp, \mathcal{Q}_\sharp, \Diamond\mathcal{P}_\sharp, \Diamond\mathcal{Q}_\sharp$

定理 2 $\mathcal{P}_\sharp \cong \mathcal{P}, \mathcal{Q}_\sharp \cong \mathcal{Q}, \Diamond\mathcal{P}_\sharp \cong \Diamond\mathcal{P}, \Diamond\mathcal{Q}_\sharp \cong \Diamond\mathcal{Q}$

証明.

クラス $\mathcal{P}_\sharp, \mathcal{Q}_\sharp, \Diamond\mathcal{P}_\sharp, \Diamond\mathcal{Q}_\sharp$ の故障計数器でそれぞれ $\mathcal{P}, \mathcal{Q}, \Diamond\mathcal{P}, \Diamond\mathcal{Q}$ の故障検知器を模倣できることを示す. 逆は明らかに成り立つ.

図1の変換アルゴリズムを用いる. アルゴリズムの概要は以下の通りである. あるプロセス $i \in \Pi$ の故障計数器が $H_C(i,t)$ を出力している時に他のプロセスすべてにメッセージを送り, その返事を返してきたプロセス以外を故障と判断する. 特に (2)-(5) を' 点呼' と呼ぶことにする.

以下ではまず, このアルゴリズムを用いると Strong Accuracy, Eventual Strong Accuracy を持つ故障計数器はそれぞれ Strong Accuracy, Eventual Strong Accuracy を持つ故障検知器に変換されることを示す. また, 同様に Strong Completeness, Weak Completeness を持つ故障計数器がそれぞれ Strong Completeness, Weak Completeness を持つ

故障検知器に変換されることを示す.

・**Strong Accuracy**

背理法で示す. Strong Accuracy が満たされないと仮定すると, $\exists i, j \in \Pi - F(t) : j \in output_i$ を満たすような時刻 t が存在する. ここで, 時刻 t の $output_i$ が決定した時の k を考える. この時, i は (2) において j 以外の $n-k$ 個のプロセスから $reply$ を受け取ったことになる. ここで, 計数器の Strong Accuracy より, 時刻 t ですでに k 個のプロセスは確実に $crash$ しているので, $message(i,k)$ を受け取る $crash$ していないプロセスはせいぜい $n-k$ 個である. 仮定より, 時刻 t では $j \notin F(t)$ であるから, j の $reply$ 無しには $n-k$ 個の $reply$ はそろわない. よって矛盾.

・**Eventual Strong Accuracy**

故障計数器が Eventual Strong Accuracy を満たすので, ある時刻 t が存在し, t 以降は実際の故障数よりも多く数えるプロセスは存在しない. この時, 前述の Strong Accurasy と同様の証明より, t 以降に行われる点呼では任意の $output_i$ は故障していないプロセスを含まない. よって, Eventual Strong Accuracy は満たされる.

・**Strong Completeness**

すべての $p \in crashed(F)$ が crash する時刻を t とする. 計数器の Strong Completeness より, ある時刻 t' が存在し, すべての $q \in correct(F)$ が $t'' \geq t$ において $H_C(i,t'') \geq |crashed(F)|$ となる. よって, $t'' \geq t$ で行われる点呼では, すべての $p \in crashed(F)$ が $reply$ を出さず, $p \in output_i$ となる. よって Strong Completeness は満たされる.

・**Weak Completeness**

Strong Completeness の証明と同様.

以上より, $\mathcal{P}_\sharp \cong \mathcal{P}, \mathcal{Q}_\sharp \cong \mathcal{Q}, \Diamond\mathcal{P}_\sharp \cong \Diamond\mathcal{P}, \Diamond\mathcal{Q}_\sharp \cong \Diamond\mathcal{Q}$ である. □

定理 2 より, (Eventual)Strong Accuracy を持つようなクラスでは ID で故障情報を得ることと数で故障情報を得ることには差が無いことが分かる.

4.2 クラス $\mathcal{I}$

定理 3 $\mathcal{I} \prec \mathcal{P}$

証明.

$\mathcal{I} \preceq \mathcal{P}$ であることは容易に分かる. よって, ここでは $\mathcal{I} \not\succeq \mathcal{P}$ のみ示す.

背理法で示す. 任意の故障パタンにおいて, クラス $\mathcal{I}$ に属する計数器の任意の履歴に対してクラス $\mathcal{P}$ に属する検知器への帰着アルゴリズム A が存在すると仮定し, 矛盾を導く.

時刻 0 で 1 つを除いてすべてのプロセスがクラッシュし, その後は故障が起こらないような故障パタンを F_1 とし, その時の A のランを r_1 とする. F_1 において $correct$ なプロセスを p とし, p が時刻 0 から $n-1$ 個の故障を永遠に疑っているような計数器履歴を考える. このような履歴は $\mathcal{I}$ に属す. 仮定より, A の出力は Strong Completeness を満たすので, ある時刻 t_p が存在し, t_p 以降 p は $\Pi \setminus \{p\}$ を出力し続けなければならない. r_1 では, p には 1 通のメッセージも届かない. このようなラン r_1 はすべてのプロセスについて考えられる.

次にすべてのプロセスが $correct$ であるような故障パタン F_2 を考え, その時のランを r_2 とする. この時, あるプロセス p が時刻 0 から永遠に故障数 $n-1$ を数え, 他のプロセスは永遠に故障数 0 を数えるとする. この時, p に宛てられたすべてのメッセージが上で定義した t_p を過ぎるまで届かないと仮定すると, p にとっては r_1 と同じ状況になり, 時刻 t_p に $\Pi \setminus \{p\}$ を出力する. これは Strong Accuracy を満たさない. よって, 矛盾.

以上より, 故障数任意の環境下においてはクラス $\mathcal{I}$ の計数器ではクラス $\mathcal{P}$ の検知器を模倣できない. □

定理 4 $\mathcal{I} \succ \mathcal{S}$

証明.

まず, $\mathcal{I} \succeq \mathcal{S}$ を示す.

クラス $\mathcal{I}$ の故障計数器でクラス $\mathcal{S}$ の故障検知器を模倣できることを示す. 図 1 のアルゴリズムで変換可能である. アルゴリズムの出力がクラス $\mathcal{S}$ のそれぞれの性質を満たすことを示す.

・**Weak Accuracy**

故障数を多く数えるプロセスを i とする. この時, 任意の $j \in \Pi \setminus \{i\}$ は定理 2 の証明より, 故障していないプロセスを疑うことは無い. また, プロセス i も自分自身を疑うことは無いので, すべてのプロセスが i を決して疑わない. i は $correct$ プロセスであるから, Weak Accuracy は満たされる.

・**Strong Completeness**

$\mathcal{I}$ は Strong Completeness を満たすので, 定理 2 と同様の証明が可能.

次に, $\mathcal{S} \not\succeq \mathcal{I}$ を示す.

背理法で示す. 任意の故障パタンにおいて, $\mathcal{S}$ に属する検知器の任意の履歴に対して $\mathcal{I}$ に属する計数器への帰着アルゴリズム A が存在すると仮定し, 矛盾を導く.

まず，時刻 0 でプロセス p が故障し，他のプロセスは *correct* であるようなような故障パタン F_1 を考え，その時のランを r_1 とする．p 以外のプロセスが時刻 0 から永遠に p だけを疑うような検知器履歴を考える．このような履歴は $\mathcal{S}$ に属す．仮定より，A の出力は Strong Completeness を満たすので，ある時刻 t が存在し，t 以降 p 以外のすべてのプロセスは 1 以上を出力し続ける．これは時刻 t までにプロセス間でどのようなメッセージのやりとりがあったとしても起こる．

次にすべてのプロセスが *correct* であるような故障パタン F_2 を考え，その時のランを r_2 とする．このとき，p 以外のプロセスに関しては r_1 の時と同じ検知器履歴で，p の履歴は常に空集合であるような履歴を考える．このような履歴は $\mathcal{S}$ に属す．

このとき，p から他のプロセスへのメッセージはすべて時刻 t を過ぎてから届くと仮定すると，p 以外のプロセスは r_1 と同じ状況になり，時刻 t に 1 以上を出力する．これは多く数えるプロセスはせいぜい 1 つという*Accuracy を満たさない．よって，矛盾．

□

4.3 クラス $\mathcal{J}$

定理 5 $\mathcal{J} \prec \mathcal{P}$

証明.

定理 3 と同様の証明が可能. □

定理 6 $\mathcal{J} \succeq \Diamond\mathcal{P}$

証明.

クラス $\mathcal{J}$ の故障計数器でクラス $\Diamond\mathcal{P}$ の故障検知器を模倣できることを示す.

・Strong Completeness

すべての $p \in crashed(F)$ が crash する時刻を t とする．計数器の SC より，すべての $q \in correct(F)$ について $H_C(q,t') \geq |crashed(F)|$ であるような時刻 $t' > t$ が存在し，t' 以降常にこの不等式が成り立つ．よって，t' 以降に点呼が行われるならば，その点呼をするプロセスは故障を実際の故障数以上に数えている．この時，すべての $p \in crashed(F)$ が *reply* を出さず，かつ，必ず $n-k$ 個の *reply* がそろい $p \in suspect_q$ となる．この *suspect* がすべてのプロセスに送られ，それがいつかは届くのでこの点呼の結果ではすべてのプロセスがすべての $p \in crashed(F)$ を疑う．

次に，t' 以降に点呼を行うプロセスが存在することを示す．計数器の$+Accuracy$ より，絶対に故障を多く数えないプロセス i が存在する．したがって，そのプロセスは絶対に (10) を行わない．また，i' は必ず t' 以降に初めて H_C が $|crashed(F)|$ になるので，(11) より，必ず t' 以降に点呼を行う．

t' 以降の点呼では *reply* は必ずそろうので (5) で待ち続けることは無く，i は永遠に点呼を繰り返し *suspect* をすべてのプロセスに送り続ける．ここで，t' 以前に始まった点呼で送られた *suspect* がすべて届いた時刻を t'' とすると，すべてのプロセスは t'' 以降に最初の *suspect* が届いてからずっとすべての $p \in crashed(F)$ を疑い続けるようになるので，Strong Completeness は満たされる．

・Eventual Strong Accuracy

故障数を多く数えたプロセスは (10) により点呼を行わなくなる．よって，t' 以降に点呼を永遠に続けられるのは故障数を正しく数えているプロセスのみ．このプロセスの行う点呼では故障していないプロセスは *suspect* に入らない．いつかは$+Accuracy$ で存在の保証されている，故障数を正しく数えるプロセス以外は (10) を行い点呼しなくなる．よって，Eventual Strong Accuracy も満たされる． □

定理 7 $\Diamond\mathcal{P} \not\preceq \mathcal{J}$

証明.

故障数任意という環境下ではクラス $\Diamond\mathcal{P}$ の故障検知器ではクラス $\mathcal{J}$ の計数器を模倣できないことを示す.

背理法で示す．任意の故障パタンにおいて，$\Diamond\mathcal{P}$ に属する検知器の任意の履歴に対して $\mathcal{J}$ に属する計数器への帰着アルゴリズム A が存在すると仮定し，矛盾を導く．

まず，時刻 0 で 1 つを除いてすべてのプロセスが停止するような故障パタン F_1 を考え，その時のランを r_1 とする．生き残っているプロセス p が時刻 0 から永遠に自分以外の全てのプロセスを疑うような検知器履歴を考える．このような履歴は◇P に属す．仮定より，A の出力は Strong Completeness を満たすので，ある時刻 t_p が存在し，t_p 以降 p は $n-1$ 以上を出力し続けなければならない．r_1 では p には永遠に 1 通のメッセージも届かない．このようなラン r_1 はすべてのプロセスについて考えられる．

次にすべてのプロセスが *correct* であるような故障パタン F_2 を考え，その時のランを r_2 とする．すべてのプロセスが時刻 0 から他のすべてのプロセスを疑っており，各プロセスは上で定義した t_p 以降にずっと出力が空集合になるような履歴を考える．このような履歴は $\Diamond\mathcal{P}$ に属す．各プ

/すべてのプロセスが以下を実行/
/初期化/
(1) 任意の $i \in \Pi$ について，$output_i := \emptyset$;
(2)$suspect_i := \emptyset$;
/点呼の繰り返し/
while(){
(3)$k := H_C(i,t)$;
(4) すべての $j \in \Pi$ に対して $message(i,k)$ を送る.
(5)$n-k$ 個の $reply(j,k)$ が届くまで待つ.
(6) 届いたら，$reply$ を送ってこなかった k 個のプロセスの集合を $suspect_i$ とする.
(7) すべてのプロセスに $suspect_i$ を送る.
}
/$output$ の決定/
while(){
(8)$output_i :=$ 最後に受け取った $suspect_j$; }
/割り込み/
(9) すべての $j \in \Pi$ は $message(i,k)$ を受け取ったら i に対して $reply(j,k)$ を返す.
(10) 送った $message(i,k)$ に対し $n-k$ 個より多く $reply(j,k)$ が届いたら，以降は (7) だけをくり返す.
(11) 計数器の値 H_C が変化したら (3) へ．ただし，すでに (10) を実行していれば (11) は無視する.

図 2: アルゴリズム 2

ロセスに宛てられたすべてのメッセージがそのプロセスの t_p を過ぎるまで届かないと仮定すると，すべてのプロセスが r_1 と同じ状況になり A は時刻 t_p に $n-1$ を出力する．これは多く数えないプロセスが 1 つは存在という+Accuracy を満たさない．よって，矛盾.

以上のことから，故障数任意の環境下においては◇P の検知器では $\mathcal{J}$ の計数器を模倣できない. □

系 2 $\mathcal{J} \succ \Diamond\mathcal{P}$

証明.
定理 6 と 7 から言える. □

4.4 $\mathcal{S} \not\preceq \mathcal{J}$

定理 8 故障を多く数えるプロセスが 2 つ以上あるクラスの故障計数器ではクラス $\mathcal{S}$ の故障検知器を模倣できない.

証明.
定理 7 と同様の証明を行う.
定理 7 の証明で考えた r_1 を 2 つのプロセス $p,q \in \Pi$ についてそれぞれ考える．また，r_2 と同様の故障パタンを考え，p,q がそれぞれ時刻 0 から永遠に $n-1$ 個の故障を数えるような計数器履歴を考える．p,q に宛てられたすべてのメッセージが $max\{t_p,t_q\}$ まで届かないとすると，p,q はそれぞれ時刻 t_p と t_q に $\Pi \setminus \{p\}$ と $\Pi \setminus \{q\}$ を出力する．これは Weak Accuracy を満たさない. □

系 3 $\mathcal{S} \not\preceq \mathcal{J}$

証明.
定理 8 から言える. □

5 まとめと今後の課題

故障したプロセスの ID 集合を返す故障検知器に対し，故障したプロセスの数のみを返す故障計数器を定義した．また，故障計数器のクラスと故障検知器のクラスの強弱関係について考察し，クラス $\mathcal{P},\mathcal{Q},\Diamond\mathcal{P},\Diamond\mathcal{Q}$ に関しては故障検知器の返す故障プロセスの ID 情報をそのまま数情報に変換しただけの故障計数器と同等の能力を持つことが分かった．また，$\mathcal{P}$ と $\mathcal{S}$ の間に $\mathcal{I}$，$\mathcal{P}$ と $\Diamond\mathcal{P}$ の間に $\mathcal{J}$ というクラスが存在することを示した.

今後の課題としては，故障検知器のクラス $\mathcal{S},\mathcal{W}$ と同等の能力を持つ故障計数器のクラスを見つける必要があると考えている．また，クラス $\mathcal{I}$，$\mathcal{J}$ がその問題を解く最弱の故障計数器となるような問題の考察も行いたい.

参考文献

[1] T. D. Chandra, V. Hadzilacos, and S. Toueg. The weakest failure detector for solving consensus. J. ACM, 43(4):685-722, 1996.

[2] T. D. Chandra and S. Toueg. Unreliable Failure Detectors for Reliable Distributed Systems. J. ACM, 43(2):225-267, 1996.

[3] M. J. Fischer, N. A. Lynch, and M. S. Paterson. Impossibility of distributed consensus with one faulty process. J. ACM, 32(2):374-382, 1985.

[4] R. Guerraoui. Non-Blocking Atomic Commit in Asynchronous Distributed Systems with Failure Detectors. Distributed Computing, 15:17-25, 2002.