UDC 621.311.1-52: 621.398: 681.3.02

論文

59-B74

系統操作の正当性を自動検証する一手法

正員 松本 啓之亮 (三菱電機)

正員 坂口 敏明 (三菱電機)

1. ま え が き

系統操作は、給電所の管轄範囲の拡大に起因する業務量の増大のほかに、実際の操作が特定の時間に集中する傾向がある。このため操作手順作成の省力化や供給信頼度の面から迅速、確実な操作の実施を図るため系統操作の自動化が進められている。現在は系統復旧時の自動操作が研究開発の中心となっている⁽¹⁾。

系統操作の自動化の技術課題の一つとして、操作手順の自動作成の問題がある。これに対してはいくつかの手法が提案されている^{(2)~(4)}。文献に掲げたものは、命題論理や述語論理が主要な解析手法となっていて、論理式を証明するための基本的な定理証明システムを利用している。操作手順は、証明過程において作成される。このように自動作成、または計算機との対話によって作成された操作手順は、それが正しいかどうかのチェックがなされる。現状のシステムでは手順の正当性の検証は人間が行っており、計算機側にはこの機能は十分備わっていない。従って、手順の正当性を自動検証する手法を開発することが、系統操作の自動化を図るうえでも一つの重要課題となる。

本論文では時制論理 (temporal logic) を用いて、正当性の検証を自動化する一手法を提案する。与えられた操作手順が、ある満たすべき性質をすべて満足しているとき、その操作は正しいと定義する。例えば、平常時操作で操作中および操作後も常に停電が起らないとか、操作がいつか完了することが保証されるとかが、満たすべき性質に当たる。ここで、「常に」とか、「いつか」という概念が重要である。古典論理では、これらの概念は非常に取り扱いにくいものとなる。この状況をうまく記述する適当なものとして注目されているのが時制論理である。

以下では、まず時制論理の概念について述べ、操作

手順を逐次形プログラムとして形式的に定義し、手順の正当性をこのプログラム上で定義される時制論理式で表現する。そしてこの論理式を時制論理の定理を用いて計算機に自動的に証明させること (以下、このことを「自動証明」という) により正当性の検証を行なう。簡単な例題を用いて検証例を示す。検証の方法としては、上記のように、計算機が証明を行なうのではなく、検証項目を人間に提示してその判断を人間に委ねるというやり方も考えられるが、今後の発展性の面から考えて前述の方法を採用ことにする。

2. 時制論理の概要

ここでは、命題論理に時制の概念を付加したときに得られる時制論理について簡単に説明する。それ以外の詳細については文献(5)~(7)に譲ることにする。命題論理は命題記号 ($p_1, p_2, \dots, p_n, \dots$) と論理演算子 ($\wedge, \vee, \rightarrow, \sim$) とから定義される。命題論理での論理式は再帰的に導かれる。すなわち

(a) すべての命題記号 p_i は論理式である。

(b) もし p_i と p_j が論理式なら、

$p_i \wedge p_j$ (and) $p_i \vee p_j$ (or)

$p_i \rightarrow p_j$ (imply) $\sim p_i$ (not)

もまた論理式である。

(c) (a), (b) で定義されたものだけが論理式である。

時制論理は上記の体系に時制を表す演算子を付け加えたものであり、各時制演算子は

$\square p$ always すべての時点で p は真である。

$\diamond p$ sometime 将来いずれかの時点で p は真である。

$\bigcirc p$ next 次の時点で p は真である。

を表す。古典論理は論理関係を明確に規定するのに有効であるが、時間の前後関係を表現するものではないので、順序関係の記述には本質的に向いていない。一方、時制論理は時間の系列に対して命題を与えることができるので、事象の順序関係の記述に有効である。時制論理は離散的かつ無限の時間系列 σ というものを

Verification of Operation Sequence for Substation Switching Equipments. By Keinosuke Matsumoto, Member & Toshiaki Sakaguchi, Member (Mitsubishi Electric Corporation).

松本啓之亮: 正員, 三菱電機株式会社中央研究所
坂口敏明: 正員, 三菱電機株式会社中央研究所

前提としている。系列 $\sigma = l_0, l_1, \dots$ とは状態の無限列である。また、ある系列

$\sigma = l_0, l_1, \dots$ に対して、

$\sigma(i) = l_i, l_{i+1}, \dots$

として σ を i だけずらした系列を表す。ある命題 p に対して、系列 σ 上での評価は

$p = p|\sigma = p|l_0$

となる。 $p|\sigma$ は σ の最初の状態 l_0 における p の真値のことである。

次に、時制演算子についてみる。 w をある論理式とするとき、 $\square$ (always) 演算子に対して、

$\square w|\sigma = \text{true}$

とは、 $k \geq 0$ であるすべての k について

$w|\sigma(k) = \text{true}$

のことである。 $\diamond$ (sometime) 演算子に対しては、

$\diamond w|\sigma = \text{true}$ とは、

$w|\sigma(k) = \text{true}$

となる $k (\geq 0)$ が存在することである。 $\bigcirc$ (next) 演算子に対しては、次式である。

$\bigcirc w|\sigma = w|\sigma(1)$

時制論理式は、このようにして系列 σ 上で評価される。(以下では、論理式の表記は系列 σ を陽に表現しないものとする)。古典論理ではいまの時点に対して記述が与えられるのに対し、時制論理では時間の系列 $\{l_0, l_1, \dots, l_n, \dots\}$ に対して記述を与えることができ、この系列上での、あるシステムの性質を容易に表すことができる。例えば、

$p \rightarrow \square q \dots \dots \dots (1)$

は「 p が現時点で真ならば、 q は現時点以後すべての時点で真である」ということを意味する。また $\diamond p$ は上記のように、将来のいずれかの時点で真になればよく、必ずしもずっと真でありつづける必要はない。すなわち、

$p \rightarrow \diamond q \dots \dots \dots (2)$

は「もし p が現時点で真ならば、 q はいつか真になる」ということを表す。 $\bigcirc$ を用いて連続する二時点の関係を記述することができる。例えば、

$p \rightarrow \bigcirc q \dots \dots \dots (3)$

は「 p が現時点で真ならば、次の時点で q は真となる」とことを表す。時制演算子を重ねて用いることもできる。例えば、

$p \rightarrow \diamond \square q \dots \dots \dots (4)$

は「 p が現時点で真ならば、ある時点以後ずっと q が真となる」を表す (その時点以前には断続的に q が真であったかもしれない)。また

$p \rightarrow \square \diamond q \dots \dots \dots (5)$

は「 p が現時点で真ならば、いつでも q が真となることがある。つまり無限回 q が真となりうる」を表す。

このように $\square$ を用いて、safeness (いつも起こる) を、また、 $\diamond$ を用いて liveness (いつかは起こる) の性質を表すことができる。

3. 逐次形プログラムによる操作手順の表現

操作手順は各操作の実行前に満たさねばならない条件と、ある設備に対してなされる操作内容を規定している。例えば、しゃ断器の両端の電圧の位相差がある範囲内でない投入してはいけないなどが、条件に相当する。操作としては、開閉器の状態を単に開から閉に、またはその逆にすることである。

操作手順を記述するのに、FORTRAN に代表される逐次形プログラムを用いる。このようなプログラムは、各ステップにラベル付けをしてそれを節点に、またラベル間の遷移を枝に対応させることにより、有向グラフとして表すことができる。遷移は一般に図1のように書くことができる。この図において、 y はプログラム変数、 $c(y)$ は遷移 α を起こすための条件、 f は遷移 α が起ったときに y の変換を与える関数である。遷移のための条件として何も制約がない場合もありうる。図1の遷移は次のように表すことができる。

$\dots \dots \dots$
 $l_i \quad c(y) \rightarrow y := f(y); \text{go to } l_j$
 $l_j \quad \dots \dots \dots$

もし $c(y)$ としての制約がない場合には

$\dots \dots \dots$
 $l_i \quad y := f(y); \text{go to } l_j$
 $l_j \quad \dots \dots \dots$

以下に操作手順の例を示す。これは図2に示す送電線の 1L 停止操作である。

A 線 1L 停止操作手順表

順序	操作開閉器名	操作	確認事項
1	A 線 1L CB 220	off	電流確認
2	A 線 1L LS 223	off	

この操作手順は次のようなプログラムにより表現さ

図1 プログラム
 上での遷移 $\textcircled{l_i} \xrightarrow[c(y) \rightarrow y := f(y)]{\alpha} \textcircled{l_j}$

Fig. 1. The transition between labels.

図2 送電線 A

Fig. 2. Transmission line A.

れる。但し、 y_1, y_2 はそれぞれしゃ断器 CB 220 および断路器 LS 223 の開閉状態 ($y_i = \text{true}$ のとき閉, $y_i = \text{false}$ のとき開) を、また y_3 は A 線 1 L の電流値を表す変数とする。

```

 $y_1 := \text{true}, y_2 := \text{true}, y_3 := i (i \neq 0)$ 
 $l_0 \quad y_1 := \text{false}; \text{go to } l_1$ 
 $l_1 \quad y_3 = 0 \rightarrow y_2 := \text{false}; \text{go to } l_2$ 
 $l_2 \quad \text{halt}$ 

```

単一の電気所での操作手順は、このような逐次形プログラムで十分表現できる。なぜなら、並行した同時操作がないからである。しかし複数の電気所にまたがる開閉器群を操作する場合には、同時一括操作が行なわれる可能性がある。これに対処するためには通常の逐次形プログラムだけでは不十分であり、並列処理を記述できるプログラムが必要となる。そのような表現方法については今後の研究課題として考えてゆきたい。

4. 系統操作の正当性を検証するための時制論理

前項の方法によって、与えられた操作手順をプログラムとして形式的に表現し、操作手順の正当性をこのプログラム上で定義される時制論理式として表す。そしてこの論理式を時制論理における定理を用いて証明することにより正当性の検証を行なう。具体的な例を用いて説明する。

〈4・1〉 平常時の母線切替操作手順の検証 図3に示すような変電所で BUS1 から BUS2 ヘループ切替を行なう場合について考えてみる。このとき操作手順が正当であるかどうかを調べるための検証項目として次の事項が挙げられる。

- (1) 操作中および操作後も停電が起こらない。
- (2) 断路器による負荷電流の開閉がない。
- (3) 操作終了時に与えられた最終状態が実現される。

操作手順がこれらの性質を満たしていれば、その操作は正当であると考えられる。これを検証するのに時制論理を用いる。時制論理では無限の時間系列を対象としているので、与えられた操作手順をプログラムで表現するとき最後に無条件ループを付け加えて、図4のように考える。但し $[\]$ は何も操作しないことを示す。また図3に示す開閉器の名前そのものをその開閉状態を表す変数（例えば LS1 が閉なら $LS1 = \text{true}$ 、開なら $LS1 = \text{false}$ ）とみなす。このとき無停電の条件は、電源側から負荷側に至る経路が少なくとも一つ存在することと等価である。従って、これを論理式で表現す

図3 変電所モデル

Fig. 3. An example of substation.

図4 遷移グラフ

Fig. 4. Transition graph.

ると、次式となる。

$$\begin{aligned}
 & Q1(LS1, LS2, LS3, LS4, LS5, \\
 & \quad LS6, CB1, CB2, CB3) \\
 & = (CB2 \wedge LS1 \wedge LS3 \wedge CB3) \\
 & \quad \vee (CB2 \wedge LS2 \wedge LS4 \wedge CB3) \\
 & \quad \vee (CB2 \wedge LS1 \wedge LS5 \wedge CB1 \wedge LS6 \\
 & \quad \quad \wedge LS4 \wedge CB3) \\
 & \quad \vee (CB2 \wedge LS2 \wedge LS6 \wedge CB1 \wedge LS5 \\
 & \quad \quad \wedge LS3 \wedge CB3) \dots \dots \dots (6)
 \end{aligned}$$

図3で電源側から負荷側に至る経路は四つ考えられる。Q1の各項は、それぞれ四つの経路に対応しており、それらの経路において電源側から順番に遭遇する開閉器を並べたものである。従って、少なくとも一つの経路上にある、すべての開閉器が閉、すなわち true なら $Q1 = \text{true}$ となる。もし Q1 が操作中を通じて「常に」真であれば、どの時点においても停電が起こらないことが保証される。これを時制論理の論理式で書き表すと、

$$\begin{aligned}
 & \vdash \text{at } l_0 \\
 & \rightarrow \Box Q1(LS1, LS2, LS3, LS4, LS5, \\
 & \quad LS6, CB1, CB2, CB3) \dots \dots (7)
 \end{aligned}$$

となる。これはもし手順が l_0 にあれば、すなわち操作を開始したならば、無停電の条件が常に満たされているということを意味する。従って、この論理式を自動証明して真であれば、最初の検証項目が満たされていることが分かる。

第二番目の検証項目について考える。断路器は対応

するし、断器が開であるか、または分路があれば操作可能となる。具体的に図3の各断路器について、それが操作可能となる条件を列挙すると次のようになる。

断路器	操作可能となる条件
LS1	LS2 $\wedge$ CB1 $\wedge$ LS5 $\wedge$ LS6
LS2	LS1 $\wedge$ CB1 $\wedge$ LS5 $\wedge$ LS6
LS3	LS4 $\wedge$ CB1 $\wedge$ LS5 $\wedge$ LS6
LS4	LS3 $\wedge$ CB1 $\wedge$ LS5 $\wedge$ LS6
LS5	$\sim$ CB1
LS6	$\sim$ CB1

すなわち、LS1 については、LS1 の状態が変化する前後で LS2, CB1, LS5, LS6 は閉の状態にとどまっていればよい。ここで、Q2 を

$$\begin{aligned}
 & Q2(LS1, LS2, LS3, LS4, LS5, LS6, \\
 & \quad CB1, CB2, CB3) \\
 & = \{ \{ (LS1 \wedge \sim LS1) \vee (\sim LS1 \wedge LS1) \} \wedge \{ (LS2 \wedge \sim LS2) \vee (\sim LS2 \wedge LS2) \} \wedge \{ (CB1 \wedge \sim CB1) \vee (\sim CB1 \wedge CB1) \} \wedge \{ (LS5 \wedge \sim LS5) \vee (\sim LS5 \wedge LS5) \} \wedge \{ (LS6 \wedge \sim LS6) \vee (\sim LS6 \wedge LS6) \} \} \quad (1) \\
 & \vee \{ \{ (LS2 \wedge \sim LS2) \vee (\sim LS2 \wedge LS2) \} \wedge \{ (LS1 \wedge \sim LS1) \vee (\sim LS1 \wedge LS1) \} \wedge \{ (CB1 \wedge \sim CB1) \vee (\sim CB1 \wedge CB1) \} \wedge \{ (LS5 \wedge \sim LS5) \vee (\sim LS5 \wedge LS5) \} \wedge \{ (LS6 \wedge \sim LS6) \vee (\sim LS6 \wedge LS6) \} \} \quad (2) \\
 & \vee \{ \{ (LS3 \wedge \sim LS3) \vee (\sim LS3 \wedge LS3) \} \wedge \{ (LS4 \wedge \sim LS4) \vee (\sim LS4 \wedge LS4) \} \wedge \{ (CB1 \wedge \sim CB1) \vee (\sim CB1 \wedge CB1) \} \wedge \{ (LS5 \wedge \sim LS5) \vee (\sim LS5 \wedge LS5) \} \wedge \{ (LS6 \wedge \sim LS6) \vee (\sim LS6 \wedge LS6) \} \} \quad (3) \\
 & \vee \{ \{ (LS4 \wedge \sim LS4) \vee (\sim LS4 \wedge LS4) \} \wedge \{ (LS3 \wedge \sim LS3) \vee (\sim LS3 \wedge LS3) \} \wedge \{ (CB1 \wedge \sim CB1) \vee (\sim CB1 \wedge CB1) \} \wedge \{ (LS5 \wedge \sim LS5) \vee (\sim LS5 \wedge LS5) \} \wedge \{ (LS6 \wedge \sim LS6) \vee (\sim LS6 \wedge LS6) \} \} \quad (4) \\
 & \vee \{ \{ (LS5 \wedge \sim LS5) \vee (\sim LS5 \wedge LS5) \} \wedge \{ (LS6 \wedge \sim LS6) \vee (\sim LS6 \wedge LS6) \} \wedge \{ (CB1 \wedge \sim CB1) \vee (\sim CB1 \wedge CB1) \} \} \quad (5) \\
 & \vee \{ \{ (LS6 \wedge \sim LS6) \vee (\sim LS6 \wedge LS6) \} \wedge \{ (CB1 \wedge \sim CB1) \vee (\sim CB1 \wedge CB1) \} \} \quad (6) \\
 & \vee \{ \{ (LS1 \wedge \sim LS1) \vee (\sim LS1 \wedge LS1) \} \wedge \{ (LS2 \wedge \sim LS2) \vee (\sim LS2 \wedge LS2) \} \wedge \{ (LS3 \wedge \sim LS3) \vee (\sim LS3 \wedge LS3) \} \wedge \{ (LS4 \wedge \sim LS4) \vee (\sim LS4 \wedge LS4) \} \wedge \{ (LS5 \wedge \sim LS5) \vee (\sim LS5 \wedge LS5) \} \wedge \{ (LS6 \wedge \sim LS6) \vee (\sim LS6 \wedge LS6) \} \} \quad (7) \\
 & \dots\dots\dots (8)
 \end{aligned}$$

で定義する。各項の意味は次のとおり。① LS1 の状態が閉から開へ、あるいは開から閉へ変化する (すな

わち $\{(LS1 \wedge \sim LS1) \vee (\sim LS1 \wedge LS1)\} = \text{true}$) と
き、LS2, CB1, LS5, LS6 はそれぞれ LS1 の操作の
前後で閉の状態にとどまっている ($(LS2 \wedge \sim LS2) =$
 true など)。②～⑥についても同様。⑦は断路器の操
作が無いことを意味している。従って第二番目の検証
項目を時制論理式で表すと、次式となる。

$$\begin{aligned}
 & \vdash \text{at } l_0 \\
 & \rightarrow \square Q2(LS1, LS2, LS3, LS4, LS5, \\
 & \quad LS6, CB1, CB2, CB3) \dots (9)
 \end{aligned}$$

図3の例で最終状態を例えば、LS1: 開, LS2: 開,
LS3: 開, LS4: 閉, LS5: 開, LS6: 開, CB1: 開,
CB2: 閉, CB3: 閉とし、また Q3 を

$$\begin{aligned}
 & Q3(LS1, LS2, LS3, LS4, LS5, LS6, \\
 & \quad CB1, CB2, CB3) \\
 & = \sim LS1 \wedge LS2 \wedge \sim LS3 \wedge LS4 \wedge \sim LS5 \\
 & \quad \wedge \sim LS6 \wedge \sim CB1 \wedge CB2 \wedge CB3 \dots (10)
 \end{aligned}$$

で定義すると、最後の検証項目は次の時制論理式で表
現される。

$$\begin{aligned}
 & \vdash \text{at } l_0 \\
 & \rightarrow \diamond \square Q3(LS1, LS2, LS3, LS4, LS5, \\
 & \quad LS6, CB1, CB2, CB3) \dots (11)
 \end{aligned}$$

次に、これらの時制論理式の証明法について考え
る。(7), (9)式に対しては時制論理体系内の不変原
理 (Invariance Principle)⁽⁷⁾ と呼ばれている定理を用
いる。

不変原理

もし (ア) Q が最初に真, すなわち

$$\vdash \text{at } l_0 \rightarrow Q(y_1, \dots, y_n) \text{ かつ}$$

(イ) Q がプログラム p の任意の遷移につ
いて真でありつづける, すなわち

$$\vdash \text{at } l \wedge Q(y_1, \dots, y_n) \rightarrow Q(f(y_1), \dots, f(y_n))$$

ならば、Q は常に真である。

$$\vdash \text{at } l_0 \rightarrow \square Q(y_1, \dots, y_n)$$

また、(11)式の証明は、

$$\begin{aligned}
 & \vdash \text{at } l_e \\
 & \rightarrow Q3(LS1, LS2, LS3, LS4, LS5, LS6, \\
 & \quad CB1, CB2, CB3) \dots\dots\dots (12)
 \end{aligned}$$

を証明することにより導かれる。

今回開発した自動証明システムは、記号処理に適し
た LISP 言語⁽⁸⁾で書かれており、(7), (9)式のよう
な不変性の証明に対しては不変原理の条件部(ア)、
(イ)が成立するかどうかをみる。このため操作手順を
プログラム化したものの内容を検討し、あらゆる遷移
についてそれが実行可能かどうか、可能であればその
前後で不変性が保たれているかどうかを点検してい

Sequence Switch Action				Sequence Switch Action			
1	CB2	off		1	CB1	in	
2	CB3	off		2	LS5	in	
3	LS1	off		3	LS6	in	
4	LS3	off		4	LS2	in	
5	LS2	in		5	LS1	off	
6	LS4	in		6	LS4	in	
7	CB2	in		7	LS3	off	
8	CB3	in		8	CB1	off	
				9	LS5	off	
				10	LS6	off	

<pre> (defprop operation (l1 l2 l3 l4 l5 l6 l7 l8) sequence) (defprop l1 CB2 switch) (defprop l1 nil action) (defprop l2 CB3 switch) (defprop l2 nil action) (defprop l3 LS1 switch) (defprop l3 nil action) (defprop l4 LS3 switch) (defprop l4 nil action) (defprop l5 LS2 switch) (defprop l5 t action) (defprop l6 LS4 switch) (defprop l6 t action) (defprop l7 CB2 switch) (defprop l7 t action) (defprop l8 CB3 switch) (defprop l8 t action) (defprop Q1 (or (and CB2 LS1 LS3 CB3) (and CB2 LS2 LS4 CB3) (and CB2 LS1 LS5 CB1 LS6 LS4 CB1) (and CB2 LS2 LS6 CB1 LS5 LS3 CB3))) form) </pre>				<pre> Q1 is not satisfied at 1. Q2 is not satisfied at 2. </pre>			
---	--	--	--	--	--	--	--

(a) (b)

図 5 証明システムへの入力

Fig. 5. Inputs for the theorem prover.

る。証明システムに与える入力、プログラム化された操作手順(順序および操作内容)、開閉器の初期状態と最終状態、ある性質が不変であることを表した論理式および最終的に達成させたい性質を示す論理式である。

実際に証明システムに与える入力の一部を図5に示す。例えば、図6(a)の操作手順は(*)のようになる。LISPでは t はtrueに、nilはfalseに対応する。またQ1の論理式は(**)のような形式で与えられる(defpropは変数に値をセットするためのLISP関数である)。しかしこれらの中には人間が直接入力するのではなく、計算機により自動導出できるものがある。それらについては自動化をより進める段階で考慮したい。

この例題に対して具体的ないくらかの操作手順について、その正当性を上述の証明システムに検証させた検証結果を図6に示す。この図の(a),(b),(c)は、それぞれ検証項目の(1),(2),(3)に反する場合に相当する。またその原因がどの操作によるものかも検証結果は示している。(a)の操作手順は停電切替の場合の手順であるが、この例題ではループ切替を考えているので、この手順は正当ではないという結論になる。(b)はCB1とLS5のインターロック条件に反する手順である。また(c)は停止母線側の断路器LS3をoffし忘れている場合である。(d)がこの場合の正しい手順の例となる。これらの結果をみていると、検証はそれを行なうことにより、あたかも不完全なプログラムを、完全なものへ仕上げてゆく過程であるとみなすことができる。

〈4.2〉 事故時復旧操作手順の検証 原理的には平常時操作と同様に各開閉器に対してその開閉状態を表す変数を割り当て、満たすべき性質を時制論理式と

Sequence Switch Action				Sequence Switch Action			
1	LS5	in		1	LS5	in	
2	LS6	in		2	LS6	in	
3	CB1	in		3	CB1	in	
4	LS2	in		4	LS2	in	
5	LS1	off		5	LS1	off	
6	LS4	in		6	LS4	in	
7	CB1	off		7	LS3	off	
8	LS5	off		8	CB1	off	
9	LS6	off		9	LS5	off	
				10	LS6	off	

The state of LS3 differs from its final state. All given requirements are satisfied.

(c) (d)

図 6 検証結果

Fig. 6. The result of verification.

して表現すれば、操作手順の検証を行なうことができる。二次系の復旧操作について考えると、原則としてループをつくらず放射状に復旧し、いかなる箇所にも過負荷が起こらないようにしなければならない。これらの要件を開閉状態を表す変数で直接表現するのは困難であるので、特殊な関数を導入する必要がある。

まず、ループをつくらないという条件を考えるために、対象系統の開閉器を枝(開閉器が閉であれば枝が存在し、開であれば枝は存在しないものとする)で、母線や送電線、変圧器を点で表して系統をグラフ化する。このときそのグラフの点の数を $\alpha 0$ 、枝の数を $\alpha 1$ 、第0ベッチ数⁽⁹⁾(付録参照)を $p 0$ とすると、ループがないための必要十分条件は、(13)式となる。

$$p 1 = \alpha 1 - \alpha 0 + p 0 = 0 \dots\dots\dots (13)$$

ここで、 $p 1$ は第1ベッチ数と呼ばれるものである。開閉状態を表す変数ベクトルを Y とすると、(13)式は次のように書くことができる。

$$p 1(Y) = \alpha 1(Y) - \alpha 0 + p 0(Y) = 0 \dots (14)$$

但し, $\alpha 1(Y)$ は枝の数を, $p0(Y)$ は第 0 ベッチ数を与える関数である (点の数 $\alpha 0$ は開閉状態に依らず一定である)。

次に, 過負荷が起こらないという性質を Y を用いて, どのように表すかについて考える。開閉器の開閉状態が与えられれば, 系統構成が確定する。従って, 潮流計算を行なうことにより, 各送電線や変圧器を流れる有効電力が求まる。従って各送電線や変圧器の送電許容値を $mi(i=1, \dots, n)$ とすると, 過負荷が起こらないという性質は次のように表される。

$$(f1(Y) \leq m1) \wedge (f2(Y) \leq m2) \wedge \dots \wedge (fn(Y) \leq mn) \dots \dots \dots (15)$$

ここで, $fi(Y)$ は各開閉器の開閉状態を与えたときに送電線 (あるいは変圧器) i に流れる有効電力の絶対値を与える関数である。

従って, (14), (15) 式が操作中「常に」成立するかどうか, また最終的に復旧方針どおりの系統構成になっているかを調べればよいことになる。検証方法としては, 特別な関数が導入されている点を除けば, 平常時操作のところで説明した方法と全く同一である。この他の検証項目についても, 新しい変数や関数を導入することにより論理式で表現して, これまでと同様の手法により検証できる。

5. あとがき

系統運用の効率化のために系統操作の自動化が図られている。しかし, しゃ断器や断路器の操作手順が正しいかどうかの検証は, 実質的にはオペレータがすべて判断しており, 計算機側にはそのための十分な機能がないのが現状である。操作手順においては, 満たすべき性質を表現するために, 時間的な概念である「常に」とか「いつか」とかが重要な役割を果す。このため本論文では, 操作手順が持っている時間的な性質を扱い, かつ手順が正当なものであるかを検証するための手法として時制論理を用いたわく組を提案した。

具体的には, 与えられた操作手順が, ある満たすべき性質をすべて満足しているとき, その操作は正しいと定義し, 次のような手続きにより手順の検証がなされる。まず, 操作手順を逐次形プログラムとして解釈し, 手順の満たすべき性質をこのプログラム上で定義される時制論理式で表現する。次に与えられた手順がこの論理式を満足するかどうかを調べることで正当性の検証がなされる。このように本手法は, 操作手順の検証問題を形式的に取り扱うことができ, 自動化を図るのに適している。しかもその過程は, すべて時

制論理体系により保証されたものになっている。ここで提案した手法では, 手順の満たすべき性質を表現した時制論理式は, 人間が与えなければならない。これらの時制論理式そのものも公理から導びき出すように完全自動化するには, 時制論理を組みこんだ定理証明システムが必要となるが, これは今後の研究課題としたい。(昭和 58 年 11 月 18 日受付)

文 献

- (1) 松本・坂口:「知識ベースに基づく電力系統復旧方式の決定法」電学論 B 103, 175 (昭 58-3)
- (2) 坂口:「電気所運転における機器操作手順の自動作成法」電気学会電力技術研究会資料 PE-81-44 (昭 56-7)
- (3) 松本・武田:「命題論理を用いた系統操作手順の決定」昭 56 電気学会大会 No. 880
- (4) 角, 他:「電力系統におけるスイッチ開閉シーケンスのベトリネットによる解析」システムと制御 26, 9, 605 (昭 57)
- (5) Z. Manna & A. Pnueli: "Verification of Concurrent Programs, Part I: The Temporal Framework", Report No. STAN-CS-81-836, Dept. of Computer Science, Stanford University, June, 1981
- (6) Z. Manna & A. Pnueli: "Verification of Concurrent Programs, Part II: Temporal Proof Principles", Report No. STAN-CS-81-843, Dept. of Computer Science, Stanford University, September, 1981.
- (7) Z. Manna, "Verification of Sequential Programs: Temporal Axiomatization", Report No. STAN-CS-81-877, Dept. of Computer Science, Stanford University, September, 1981
- (8) 黒川:「LISP 入門」(昭 57-6) 培風館
- (9) 小野寺: グラフ理論の基礎 (昭 43-9) 森北出版

付 録

第 0 ベッチ数 あるグラフが次の性質

「グラフの任意の点から他の任意の点に適当にいくつかの枝をたどれば到達することができる」を満たすとき, このグラフを連結グラフという。それ以外は非連結グラフと呼ばれ, いくつかの連結グラフの集りで構成されている。付図 1 はこの性質を満たさないから非連結グラフで, 3 個の連結グラフからなっている。非連結グラフが $p0$ 個の連結グラフからなるとき, その個数 $p0$ を第 0 ベッチ数という。従って付図 1 の $p0$ は 3 である。また連結グラフの $p0$ は 1 である。

付図 1 非連結グラフ

app. Fig. 1. A non-connected graph.