ソフトウェア工学 89−13
（1992. 12. 15）

状態空間モデルを背景とした制御仕様の視覚的獲得方式

浪岡 保男　笹氣 光一　伊藤 聡　中山 康子　水谷 博之
(株) 東芝 研究開発センター 情報・通信システム研究所

本稿では、制御仕様獲得方式の構想を述べる。本方式は、リハーサルベースの視覚的プログラミングを用い、設計者が制御対象となる設備の機器図形を用いて視覚的に目標状態を指示すると、プランニング手法により機器動作を推論して制御仕様を生成する。このプランニングでは、プラント設備の状態空間モデルを用いる。また、このモデル上での探索を効率的に行なう為に、目標状態に至る制御に関係のある機器の絞り込みを行なうスコープという概念を用いている。本方式は、従来の仕様記述方式に比べ機器動作イメージが把握し易く、設計者の思考レベルにより近い仕様記述方法を提供できる。

Visual programming for acquisition of sequence control specification

Yasuo Namioka　Koichi Sasaki　Satoshi Itoh　Yasuko Nakayama　Hiroyuki Mizutani
Research and Development Center, TOSHIBA Corporation
1, Komukai toshiba-cho, Saiwai-ku, Kawasaki, 210, JAPAN

This paper discusses a method which helps designers articulate sequence control specifications. The method features a rehearsal-based visual programming technique, in which designers specify the goal state of machinery using graphical representations. In this method, the system infers machine actions using a planning method and generates specifications. This planning is based on the state space model and the concept of *scope* for efficient search. The proposed method provides environments where designers can easily understand the behavior of plant machines and give instructions that are close to their thinking level.

1 まえがき

情報化が進む社会の中で、ソフトウェアの生産効率の向上は、重要な問題として取り上げられて久しい。プラント制御分野においても、プログラマブルコントローラ(PC)の適用により、制御ソフトウェアが複雑化、大規模化しており、ソフトウェアの生産性は重要な問題となっている。

プラント制御ソフトウェアの生産性向上の為、プラントの制御システム設計業務におけるエンジニアリングオートメーション化が推められている。シーケンス制御[1]ソフトウェアの開発では、上流工程で、ユーザからの要求を定義し、ラダー図形式やSFC(Sequential Function Chart)形式の制御仕様記述言語で記述された制御仕様を作成し、下流工程で、上流工程で作成した制御仕様から制御プログラムを作成する。これまで、下流工程に対する支援としては、知識処理手法[2]、プログラム変換[3]、部品合成[4]等の方式により制御ソフトウェアを自動生成する方法が試みられているが、上流工程への効果的な支援は、あまり行なわれていないのが現状である。

筆者等は、知識処理を設計問題に適用する試みの一つとして、シーケンス制御ソフトウェアの自動生成システムの研究[5]-[10]を行なっており、幾つかの実プラントへの適用も行なっている。このシステムの入力は、GRAFCET というグラフィック形の仕様記述方法を用いたSFC形式の言語で記述している。

自動生成システムの導入により、シーケンス制御ソフトウェア作成に要する時間は、全体としては減少している。しかし、設計者がソフトウェアを作成していた場合に許されていた曖昧な制御仕様記述が、自動生成システムでは許されない等の原因のために、制御仕様の記述に要する時間は逆に増加しており、シーケンス制御ソフトウェア作成に要する時間の中でも大きな比重を占めるようになってきた。また、制御仕様の品質や信頼性は、設計者がソフトウェアを作成していた場合と異なり、制御ソフトウェアの品質や信頼性に直接影響を及ぼすようになってきた。このため、制御仕様の作成効率・品質・信頼性の向上は、重要な課題となっている[11]。

一般に、SFC形式の制御仕様作成では、制御対象機器の動作を記述する機器動作部(ステップ)と、次の制御に移る為の遷移条件を記述する遷移条件部(トランジション)等を、一つ一つ入力して行なう。この作業においては、電子化する過程でのキー入力時の誤りや、上でも述べたように曖昧な記述が許されないなどの要因により作成時間が増大している。

また、SFC作成時に設計者は、まず機器や加工材料が次に取るべき目標状態をイメージし、続いてその状態に至る機器動作を考え、これをもとに制御仕様を作成する。この様に、SFCを作成する為に設計者は幾つかの思考過程を踏むが、機器構成が複雑になると機器の動作や状態を誤ってイメージしたことによる仕様記述の誤りが、特に経験の浅い設計者に起こり易く制御仕様の品質・信頼性の側面での問題となっていた。

これらの問題の解決策として、ユーザインタフェースの操作方法の改善などを行なうことも重要であるが、筆者等は、動作イメージを把握し易い、設計者の思考レベルにより近い仕様記述環境の提案が重要と考え、本稿ではその構想を述べる。

ここで提案する制御仕様獲得方式では、リハーサルベースの視覚的プログラミングを用いており、制御対象となる設備を機器図形を用いて模式的に表示し、この機器図形等を用いて設計者が目標状態を視覚的に指示すると、状態空間モデルを用いて機器動作を推論(プランニング)し制御仕様を生成する。また、推論の効率をあげる為に、設計者によって視覚的に指示された制御目標に関係のある機器と関係の無い機器とを判別する知識として、スコープという概念を導入する。そして、スコープの概念を推論時に生かし易い構造を持つ状態空間モデルを提案する。

2 背景

2.1 視覚的プログラミング

視覚的プログラミングでは、アイコニックダイアグラムを用いる方法が信号処理プログラム等の分野で多く適用されているが、筆者等は、リハーサルによるプログラミングという方法に着目する。両方法の概要を次にあげる。

アイコニックダイアグラムを用いる方法： プログラム自体をアイコンを用いたダイアグラムで表す方法[12]。この方法では、データや関数や演算子等をアイコンを用いて表示する。プログラムはこれらのアイコンを用いてデータフローモデルに基づいたアイコニックダイアグラムで表される。プログラマがアイコニックダイアグラムを作成すると、システムがそのダイアグラムを解析してプログラムを生成する。

リハーサルによるプログラミング： アイコン(図形)を用いたリハーサルによるプログラム生成を行う方法[13],[14]。この方法では、プログラミング作業を舞台劇の創造になぞらえている。画面は舞台に相当し、アイコンは俳優等に相当する。プログラマがマウス等を用いてアイコンを操作してリハーサルすると、システムはそれを観察して適切なプログラミングコードを生成する。

両方法の特徴をふまえ、機器の動作イメージを把握し易いという観点から、リハーサルによるプログラミングという方法を選択する。 ただし、一般的なリハーサルによるプログラミングを制御の分野に適用すると、設計者が機器の動作と停止条件を一つ一つ指示して制御仕様を作成することになる。しかし、これでは従来のSFC形式の言語と同じレベルでの記述となる。この記述レベルについての考察は、ロボット工学におけるロボット言語の分類を参考にできる。井上等の分

類[16]によると、機器の動作と終了条件を指示する方法は、原始的動作レベルあるいは、構造的動作レベルとなる。本稿では、設計者の思考過程の上流により近い記述レベルとして、動作レベルよりも一層高度な、対象物状態レベルを選択する。

この記述レベルを制御の分野に適用すると、設計者が機器や加工材料の目標状態を図形を用いて指示すると、初期状態から目標状態に至るための機器の動作をシステムが推論して指示された部分の制御仕様を生成する。設計者は、この作業を繰り返すことにより、制御仕様を作成していくことができる。

2.2 プランニング

設計者によって指示された目標状態に至る機器動作を推論する為に、筆者等は、プランニングの手法を用いる。この手法には、状態空間モデル[15]に基づく探索による方法とポテンシャル関数による方法の2つに大きく分類できる[17]。両方法の概要を次にあげる。

状態空間モデルに基づく探索による方法: ロボットや機器の動きうる状態・空間を計算機内に記述し、記述された状態空間の中で、動作開始点から目標点までの経路をグラフ探索法などにより決定する方法。システムの例としては、STRIPS, ABSTRIPS, Theo[18]などがあげられる。この方法では、対象の規模が大きくなると、組合せ爆発等の問題から状態空間モデルの表現やモデル上のグラフ探索が難しくなる。このため、状態空間モデルの表現方法がこの方法の成功の鍵となる。

ポテンシャル関数による方法: 障害物からの距離に応じた反発力と、目標点からの距離に応じた引力を生じるようなポテンシャル関数を定義し、それに基づいて経路を決定する方法。扱った対象の代表例としてスタンフォード・マニピュレータがある。この方法では、最適なポテンシャル関数の定義が難しいことと、計算量が非常に多いことが課題である。

両方法の特徴から、後者では、多くの機器が存在するプラント設備では、ポテンシャル関数の定義が困難である。また、プラント設備では、設備内の機器変更がしばしば行なわれるが、前者では、状態空間モデルを制御対象の対象知識から知識コンパイル[19]で生成する方式を用いれば対処が比較的容易に行なえる。これらの観点から前者の状態空間モデルに基づく探索による方法を選択する。

3 制御仕様獲得支援システムの構成

筆者等が提案する制御仕様獲得支援システムの構成を、図1に示す。本システムは、知識ベースと知識生成編集部から構成される。

知識ベースは、対象知識、状態空間モデル、制御仕様等を蓄積・管理するものである。対象知識は、制御対象となる設備について、設備や設備内の機器に関する構成、特性、動作、状態、運転知識などの知識を持つ。また、対象知識は Generic Model と Specific Model とに分かれる。Generic Model は、実際に扱う個々の設備に依存しない一般的な知識を持ち、Specific Model を生成・編集する為に用いる。Specific Model は、実際に扱う設備についての知識であり、状態空間モデルの生成や、制御仕様の生成・検証を行なう際に参照する。制御仕様は、上述したように SFC 形式で記述する。状態空間モデルについては、6章で述べる。

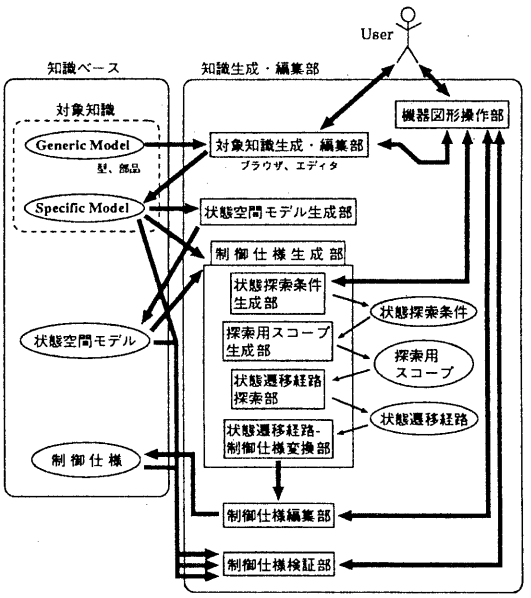

図 1: 制御仕様獲得システムの構成

知識生成・編集部において、機器図形操作部は、機器図形を用いて設計者に機器の状態や動作イメージを伝えるとともに、設計者からの機器図形を用いた指示を他の部分に伝える。対象知識生成・編集部は、Generic Model を用いて Specific Model の生成・編集を支援する。 状態空間モデル生成部は、Specific Model を用いて状態空間モデルを生成する。制御仕様生成部は、機器図形操作部から設計者の指示を受けとり、Specific Model、状態空間モデルを用いて指示された状況下で正常に動作する制御仕様を生成する。 制御仕様編集部は、制御仕様生成部から受けとった制御仕様を編集し知識ベースに登録する。 制御仕様検証部は、一度作成した制御仕様を幾つかの状況下で正常に動作するか検証する。

知識生成・編集部について、本稿では、機器図形操作部と制御仕様生成部の処理について述べる。

4 対象知識

制御対象に関する対象知識は、次にあげる知識と以下で述べる状態構成知識、動作知識、副次的状態遷移、スコープ知識により構成される。

物理構成: 設備内の物理的な構成。

特性シソーラス: 機器の特性による分類(プラント設備内にある機器に関するオントロジー)。

状態とセンサの関係: 機器状態とSFCの遷移条件部の記述との対応付け。

図形知識: 図形データ・位置と機器状態との対応付け。

運転知識: 幾つかの運転モードでの機器、プラント設備の可能な動作・状態、あるいは、異常な動作・状態の記述。

図2は、鉄鋼プラントの中のある一部を簡単化したプラント設備を機器図形を用いて模式的に表している。この設備では、「テンションリール」が鉄板を巻き取り「コイル」をつくると、「コイルカー」が、コイルを受け取り「スキッド1～3」に運搬する。本稿では、この例を用いて以下の説明を行なう。

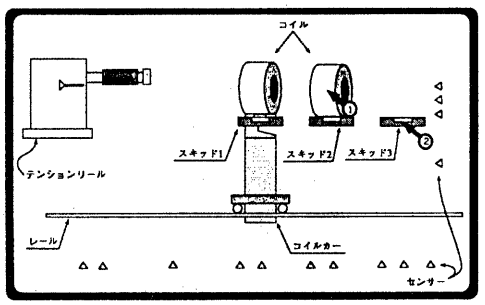

図 2: プラント設備と機器図形を用いた状態指示の例

4.1 状態構成知識

状態構成知識は、機器や設備の状態の表現する。この表現方法の概要を述べる。状態はプリミティブ状態、機器状態、設備状態の3つの階層に分けられる。プリミティブ状態は、表1(コイルカーの例)に示すような機器の状態を表現する為の最小単位である。また、表1のようにこのプリミティブ状態は、幾つかの状態項目(前後の位置、上下の位置、コイル搭載、..)に分類することができ、機器状態は、各状態項目のとるプリミティブ状態集合の直積の要素として表現する。設備状態は、各機器状態の集合の直積の要素として表現する。また、各プリミティブ状態には、表1のように、状態項目ごとに(ここでは数字を用いて)順序づけなどの関係づけを行なう。

状態構成知識は、機器動作の意味づけ、機械自身や他の機械の動作による副次的な効果(副次的状態遷移)の意味づけ、運転知識の定義・表現、正常・異常状態の判断等を行なうための基盤となっている。

表 1: コイルカーの状態

状態項目	前後の位置		上下の位置		..
プリミティブ状態	状態名	数字	状態名	数字	..
	前進限	0	下降限	0	..
	停止位置0	1	コイル運搬高さ	1	..
	IL検出位置	2	上昇限	2	..
	減速位置1	3			..
	:	:			

4.2 動作知識

動作知識は一般に、動作名、変化する状態項目、状態の変化の仕方、動作の重み(運転方法の優先度を評価するために用いるその動作を行なうためのコスト)、この動作に対応して制御仕様の機器動作部に入るべき記述等を含む。コイルカーという機械を例にとると、「上昇」という動作は、図3のように表現する。

```
知識名:    コイルカー上昇
変化項目:  上下の位置
変化:      (元の状態の数字)+1→(新しい状態の数字)
重み:      1
機器動作部: コイルカー / 上昇
    :
```

図 3: コイルカーの動作知識「上昇」

4.3 副次的状態遷移

副次的状態遷移は、動作する機器自身あるいは他の機器の動作により、その機器の状態がどのように変化するかを表す。この知識は、状態遷移が発生するかどうかをチェックする「状態遷移チェック手順」と、この手順を起動するかどうかの前提条件となる「前提条件」と、状態遷移が発生した場合の状態変化を表す「作用」等からなる。

```
知識名          :コイルカー副次的状態遷移1

前提条件
 ・コイルカー
  ‥動作                    :   上昇
  ‥元の状態
    …コイル搭載             :   No
    …上下の位置             :   下降限
  ‥変化する状態
    …上下の位置             :   コイル運搬高さ

状態遷移チェック手順
 コイルカーの状態「前後の位置」  →   x
 位置xにあるスキッド             →   y
 スキッドyの状態「コイル支持」   →   z
        if (z == Yes)        then  状態遷移成立
                             else  状態遷移不成立
作用
 コイル搭載                 :   Yes.
```

図 4: コイルカー副次的状態遷移1

図4は、コイルカーに関する副次的状態遷移の例である。この知識の表現していることを言葉で表せば、「コイルを搭載していないコイルカーが前後の位置xで下降限からコイル運搬高さまで上昇する時、前後の位置がxのスキッドを検索する。そして、それがあれ

ば、そのスキッドがコイルを支持しているかどうかを検索する。ここで、コイルを支持しているならばこの条件は「真」になり、コイルカーはコイルを搭載した状態に遷移する。」となる。

5 スコープ

スコープには、基本スコープと探索用スコープがあり、探索用スコープは、8章で説明する状態空間モデル上での状態探索を行なう際に基本スコープから作成される。この知識により、状態経路探索を行なう際に着目する機器と無視する機器とを選別することができ、推論効率を向上できる。どのように選別するかは、5.2「探索用スコープの作成」で述べる。探索用スコープは、状態空間モデル上での状態探索を行なう場合に着目する機器のリストである。

5.1 基本スコープ

基本スコープは、制御対象がもつ幾つかの作業目的に対し、どの機械が各作業に参加するかを記述することにより表現される。言い換えれば、作業あるいは機能と機械とを結び付ける知識とも言うことができる。基本スコープの構成はプラント設備の作業あるいは機能の構成となる。図5は、図2のプラント設備に関する基本スコープの構成を表す。このプラント設備の例で扱う作業としては、加工材料の「運搬作業」と加工材料の「巻き取り作業」があり、さらに、この2つの機能を統括する「設備内作業」がある。この例では、コイルカーが設備内作業・スコープに登録されている。これは、コイルカーが加工材料を運搬する作業の他にテンションリールなどと協調して加工材料を巻き取る作業にも参加する為である。

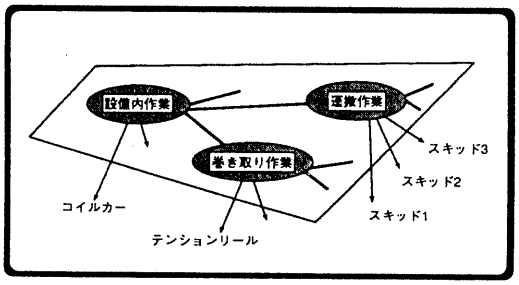

図 5: 基本スコープの構成

5.2 探索用スコープの作成

ここでは、基本スコープを、設備内作業、巻き取り作業、運搬作業等をノードとした木構造で表す。基本スコープから探索用スコープを作成する手順の概要は、次のようになる。

1. 8章で説明する状態空間モデル上で状態探索を行なう際に状態探索条件を作成するが、その条件に登録された機器と、基本スコープに登録されている機器とのマッチングを取る。
2. マッチングが取れた基本スコープがそれぞれ連結となるような root を含む最小部分木を求める。
3. 最小部分木に含まれる全ての基本スコープに登録された機器の中で能動的な特性を持つ (自らの動作を持つ) 機器を探索用スコープに登録する。

6 状態空間モデル

6.1 状態空間モデルの構成

状態空間モデルは、機器の状態遷移を扱う機器の状態空間モデルとプラント設備の状態群とを分割し、プラント設備状態とその要素の機器状態とにリンク (以下、スコープ枝) を張った構造を持つ。状態空間モデルの構成を図6に示す。

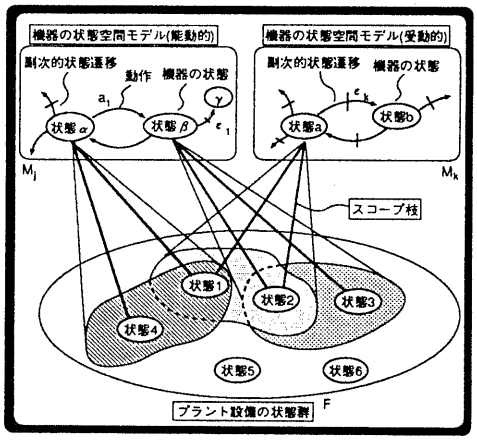

図 6: プラント設備の状態空間モデル

本モデルでは、機器の状態が遷移する要因として、機器の動作と副次的状態遷移の2つを扱う。機器の状態空間モデルは、機器状態をノードとし、動作と副次的状態遷移の2種類の有向枝を持つ有向グラフである。

機器の状態空間には機器の特性により2種類のモデルが存在する。能動的な特性を持つ機器の状態空間モデルは、動作の枝と副次的状態遷移の枝とを持つが、受動的な特性を持つ機器の状態空間モデルは、副次的状態遷移の枝だけを持つ。

設備の状態群は設備状態のノードの集合であるが、各設備状態ノード間は直接的には接続しない。また、機器状態のノード、及び、設備の状態ノードには、運転知識に照合して機器の状態として正常な (運転知識に抵触しない) もののみを扱う。

ここで、状態空間モデルの状態ノードの数について考察する。扱う設備内に n 個の機器があり、その中のある機器 M_j $(j = 1,..,n)$ の状態項目が l 個あり、各状態項目に登録されたプリミティブ状態の数を、それぞれ、$|E_i|$ $(i = 1,..,l)$ とする。

すると、機器 M_j の状態ノードの数 $|S_j|$ は、次式のように、機器の状態項目が持つ状態数の直積の要素数から異常な状態の数 b_j を取り除いた数となる。

$$|S_j| = \prod_{i=1}^{l} |E_i| \quad - b_j \tag{1}$$

また、設備の持つ正常な状態の数 $|F_0|$ は次の式で表すことができる。

$$|F_0| = \prod_{j=1}^{n} |S_j| \quad - I_0 \tag{2}$$

ここで、I_0 は、個々の機器の状態としては正常であるが、設備の状態としては異常であるものの数である。

6.2 状態遷移方法

本状態空間モデル上の状態遷移は図7に示す手続きを用いて行なう。図の中では、図6のプラント設備の「状態1」からプラント設備の「状態2」への遷移を例にとり手続きを述べている。

図7が示す状態遷移手続きは、状態空間モデル内の状態数に依存しない。また、副次的状態遷移の有向枝の重みを「0」とすれば、状態空間モデルは、プラント設備の状態ノードと遷移妥当性のチェック機能を備えた有向枝で構成された有向グラフと見なすことができる。よって、状態空間モデル上の状態経路探索は、重み付き有向グラフの最短路問題に帰着でき、基本的にはDijkstraのアルゴリズム[20]を用いて解くことができる。

```
procedure STATE TRANSITION
begin
 ・状態1を表す作業用ベクトル f_t = (s_1, .., s_n) 作成
 ・機器 M_j 選択
 ・ f_t 中の M_j の状態 s_jα に連結された動作 a_1 を選択
 ・ s_jβ → f_tj   /* f_t の M_j に対応する要素 f_tj を
       a_1 によって遷移する状態 s_jβ に書き換える */
 ・ if { s_jβ に登録されている
       副次的状態遷移 ε_1 が成立する } then begin
    ・ s_jγ → f_tj /*s_jγ : ε_1 によって遷移する状態 */
    ・ while{ε_1 に関連する他の機器 M_k(k ≠ j) の
          副次的状態遷移 ε_k に対して処理する } do
      ・ s_jb → f_tj /*s_jb:ε_k によって遷移する状態 */
    end
  end
 ・ if { f_t = (s_1, .., s_n) が正常状態である } then
    ・状態遷移 (状態1 → 状態2) の成立。
end.
```

図 7: 設備状態の状態状態遷移手続き

7 スコープ導入の効果

状態経路探索に Dijkstra のアルゴリズムを用いた場合、このアルゴリズムの計算時間は、探索の対象となるネットワークの点の数 $|V|$ を用いて、最悪、

$$O(|V|^2) \tag{3}$$

であることが知られている[20]。ここでは、設備の正常な状態の数 $|F_0|$ が $|V|$ に相当する。 ここで、スコープの概念を導入した場合に、探索の対象となる状態数は、次の式で表すことができる。

$$|F_a| = \prod_{j=1}^{n} C_j |S_j| \quad - I_a \tag{4}$$

$$C_j = \begin{cases} 1 & M_j \in \text{探索用スコープ} \\ \frac{1}{|S_j|} & M_j \notin \text{探索用スコープ} \end{cases} \tag{5}$$

ここで、式(2)と式(4)を比べると,

$$\frac{|F_a|}{|F_0|} = \frac{\prod_{j=1}^{n} C_j |S_j| \quad - I_a}{\prod_{j=1}^{n} |S_j| \quad - I_0} \simeq \prod_{j=1}^{n} C_j \tag{6}$$

このことは、スコープを用いて探索にかかる計算時間のオーダーを、「スコープで取り除かれた機器の状態数の積の2乗分の1」に削減できることを示している。

8 制御仕様の獲得過程

図2のプラント設備を例に、本稿が提案する方式を用いた制御仕様の獲得過程を述べる。この過程で設計者は、主に、初期状態と目標状態の指示を行なうだけで、制御仕様を得ることができる。

1. 初期状態の特定
   標準的な状態をあらかじめ幾つか用意する。設計者は、適切な標準状態を選択し、これを仕様入力を開始したい状態になるよう、図形を用いた指示で機器の状態を修正することにより、初期状態を特定する。連続して仕様作成する場合は、前回の目標状態を初期状態として用いることもできる。

2. 目標状態の指示
   図2の①と②の矢印は、目標状態の指示例である。この例では、スキッド2が支持しているコイルをスキッド3に移す、すなわち、「スキッド2にコイルが無く、かつ、スキッド3にコイルがある」状態を指示している。このように、目標状態の特徴的な部分を指示するだけでも良い。

3. 状態探索条件の作成

図2の指示から得られる目標状態が満たすべき条件は、この例では表2のようになる。本稿ではこの条件を状態探索条件と呼ぶ。このような条件で指示を行なう場合、指示された目標状態は、一般にこの条件を満たすプラント設備状態の集合となる。状態経路探索では、この条件を満たす状態の中で最も初期状態に近い(すなわち、状態経路を連結する有向枝の重みの合計(TW)が最小の)ものを、最終的な目標状態として選択する。

表 2: 状態探索条件の例

機械	状態項目	状態値
スキッド2	コイル支持	No
スキッド3	コイル支持	Yes

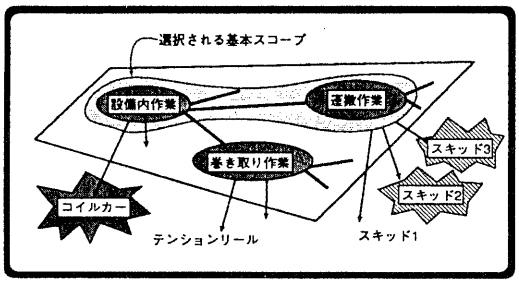

図 8: 探索用スコープ

4. 探索用スコープの作成

探索用スコープは、5.2で説明した手順で作成する。図8は、探索用スコープを決定する様子を表している。まず、状態探索条件と図5の基本スコープの双方のスキッド2、スキッド3により、基本スコープ「運搬作業」のマッチングがとれる。続いて、「運搬作業」とrootを含む最小部分木を作成すると基本スコープ「設備内作業」と「運搬作業」が選択される。そして、この2つの基本スコープに登録されている能動的な機器として、コイルカーが探索用スコープに登録される。

5. 状態遷移経路の探索

状態遷移経路の探索では、図7の手続とDijkstraのアルゴリズムをマージした手続きを用いて図9のような状態経路を求める。スコープは、図7の手続きにおいて、探索用スコープに登録した機器のみを選択することで用いている。

状態遷移経路探索で現れる状態遷移の様子を、図10の③の矢印の示す経路の状態遷移を例に説明する。この例では、始め、コイルカー状態とスキッド2状態はそれぞれ(上下の位置: 下降限, 前後の位置: 停止位置2, コイル搭載:No)と(コイル支持:Yes)である。状態探索においてこの状態から、コイルカーの動作上昇が選択されると、上昇に伴うコイルカーの状態遷移は(上下の位置: 下降限→コイル運搬高さ)である。次いで、この動作に伴う副次的状態遷移1が成立することにより(コイル搭載:No → Yes)の状態遷移が起こる。また、コイルカーの動作上昇に関連して、スキッド2の副次的状態遷移が成立しスキッド2の状態遷移(コイル支持:Yes → No)が起こる。各状態遷移の結果からプラント設備全体の状態が決定され、上昇の重み「1」がこの経路の重みとなる。

ところで、目標状態が状態探索条件を満たす状態集合で示される為、初期状態と目標状態の間に必要な機器動作が多くなるほど、目標状態の競合、または、一つの目標状態に対する状態遷移経路の競合が発生する可能性が増す。この例の程度の探索では競合する可能性は低いが、もし、競合が起こった場合は、競合する状態遷移経路を図10のようなアニメーション等を用いて設計者に提示し選択を促す。

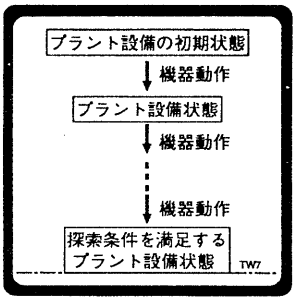

図 9: 状態遷移経路

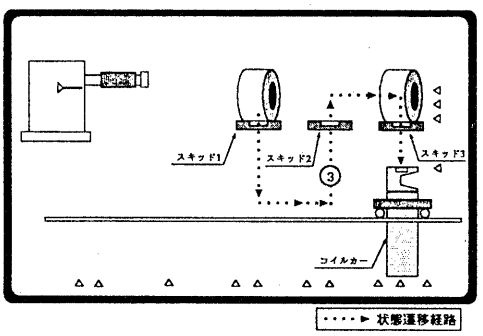

図 10: 推論により発見された状態と状態遷移経路

6. 状態遷移経路からSFC形式の制御仕様への変換

まず、状態遷移経路から、動作する機器、動作、動作の終了状態を抽出する。次に、対象知識の「動作」と「状態とセンサの関係」の知識から、抽出された項目に対応する制御仕様のための記述を検索し、次項図11の様な制御仕様を作成する。

ここまでの例の中で、図10の③の矢印が示す経路からは、コイルカー、上昇、コイル運搬高さ

が抽出され、機器動作部として「コイルカー / 上昇」が、遷移条件部としては「コイルカー /(and 高さ 1 (not 高さ 2))」が検索される。

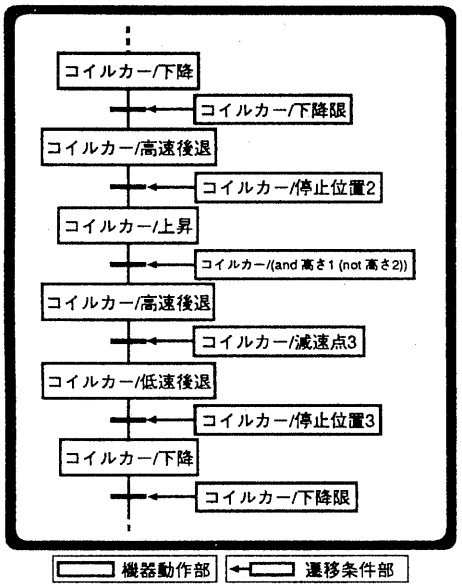

図 11: 得られたフロー形式の制御仕様

9 まとめ

本稿では、状態空間モデルを背景とした、視覚的な仕様獲得方式についての構想を述べた。本方式の特徴をあげると次のようになる。

- リハーサルベースの視覚的プログラミングを用いるため、制御対象の機器動作イメージが把握し易く、制御仕様の記述の誤記を防ぐことができる。
- リハーサルにおいて設計者の行なう指示のレベルを状態指示レベルに設定しており、主に初期状態と目標状態の指示を与えると、システムが初期状態から目標状態に至る機器動作を推論して制御則に即した制御仕様を生成する。このように、機器の動作を細かく指示しなくてもよく、生成される仕様も制御則に即しているので、既存の入力方法に比べ容易に、信頼性の高い制御仕様が得られる。
- 推論は、状態空間モデルを背景としたプランニングを用いており、探索手法やモデルの構造にスコープの概念を導入して効率良く行なっている。

本稿の提案する方式により、制御仕様作成にかかる時間の削減、制御仕様の品質及び信頼性の向上が期待できる。また、このことにより制御ソフトウェアの開発に要する時間の削減、品質の向上が期待できる。今後、本方式を用いた仕様作成支援システムをインプリメントしてこの方式の有効性を証明する予定である。

参考文献

[1] 関口、他著: シーケンス制御工学, 電気学会.

[2] 杉山健司: 知識処理による自動プログラミング, 情報処理, Vol.28, No.10, pp1297-1304,1987.

[3] 吉田紀彦: プログラム変換による自動プログラミング, 情報処理, Vol.28, No.10, pp1320-1328,1987.

[4] 古宮誠一: 部品合成による自動プログラミング, 情報処理, Vol.28, No.10, pp1329-1345,1987.

[5] Ono,Y.,et al.: Artificial Intelligence based Programmable Controller Software Designing, IEEE AI'88 Proceedings of the International Workshop on Artificial Intelligence for Industrial Applications, pp85-90, 1988.

[6] Nakayama,Y.,et al.: Model-based Automatic Programming for Plant Control, Proceedings of The Sixth IEEE Conference on Artificial Intelligence Applications, pp281-287, 1990.

[7] 貞重他：シーケンス制御ソフト自動生成システム －制御対象モデルの構築支援技術－, 情報処理学会第 39 回全国大会, 1S-2, 1989.

[8] Mizutani,H.,et al.:A Knowledge Representation for Model-Based High-Level Specification, Proeedings of The Seventh IEEE Conference on Artificial Intelligence Applications, pp124-128, 1991.

[9] 伊藤他：対象知識に基づく仕様動作説明技術, 情報処理学会第 43 回全国大会, 1E-4, 1991.

[10] Mizutani,H.,et al.:Automatic Programming for Sequence Control,Proceedings of the IAAI-92 Conference on Innovative Applications of Artificial Intelligence 4, pp315-331, 1992.

[11] 山本修一: 制御ソフトウェアを対象とした CASE の現状と動向, 情報処理, Vol.31, No.8, pp1057-1067,1990.

[12] David Lau-Kee, et al.: "VPL:An Active,Declarative Visual Programming System", Proceedings '91 IEEE Workshop on Visual Languages, pp40-46, Kobe Japan, October, 1991.

[13] J.E.Mears,C.E.Hughes and R.Braby:A Rehearsal-Based Authoring Environment for Visual Programming of Procedure Simulations,Visual Languages and Visual programming, pp159-184, Plenum Press, New York, 1990.

[14] M.Hirakawa, et al.: Interpretation of Icon Overlapping in Iconic Programming, Proceedings '91 IEEE Workshop on Visual Languages, pp254-259, Kobe Japan,October, 1991.

[15] Daniel E.Whitney:State Space Models of Remote Manipulation Tasks, IEEE Trans. on AC.,Vol.AC-14,No.6(1969)pp.617-623.

[16] 小笠原,井上：知能ロボット・プログラミングシステム COSMOS, 日本ロボット学会誌,2 巻 6 号,pp3-21,Des,1984.

[17] ロボット工学ハンドブック, 日本ロボット学会編, コロナ社, 1990.

[18] Tom M. Mitchell:Becomming Increasingly Reactive, Proceedings English National Conference on Artifitial Intelligence Volume Two, pp1051-1058, July ,1990.

[19] Ashok K.Goel:Knowledge Compilation(A Symposium), IEEE EXPERT, pp71-73, April, 1991.

[20] 伊理, 梶谷, 白川, 篠田, 他著: 演習グラフ理論, コロナ社.