ペトリネットに基づく 高フレキシブル FA 制御システム†

村田 智 洋*・薦 田 憲 久*・松 本 邦 顕*

A Petri-Net Based FA (Factory Automation) Controller for Flexible and Maintainable Control Specifications

Tomohiro MURATA*, Norihisa KOMODA* and Kuniaki MATSUMOTO*

In this paper, a new type controller based on a Petri-Net like control language named C-net is proposed. By using C-net, control specifications can be easily described as a directed graph according to a time chart representing work station's actions. Complicated control specifications of work stations can be easily comprehended and changed.

C-net can be directly executed by a micro-processor based controller named SCR (Station Controller). A C-net interpreter executes a defined C-net by interpreting it cyclically (this cycle is called an execution cycle). In real time control systems, a response time of the controller is an essential factor. An efficient execution schema of C-net interpretation called a limited scanning method is also presented. In order to evaluate the efficiency of this schema, a performance evaluating model is built. Through the analysis of this model, the expression for calculating a response time of SCR is formulated, and this theoretical response time shows good coincidence with measured response time. By using the presented method SCR can execute the C-net model within 15 msec/execution cycle (on HMCS 68000), which describes control specifications of 10 machine tools-work station, where one of actions averages finishes every half a second. (This execution time doesn't depend on a scale of a C-net model.)

SCR has been successfully applied to several applications. A parts assembly station which consists of one industrial robot and six machine tools is described as an example. In this application, initial software development man-hour was reduced about 50% compared with relay ladder diagram method. Besides, maintenances as trouble shooting and changing of control specifications have been achieved easily.

Key Words: Petri-Net, FA (factory automation), controller, sequence control programing, interpreter

1. 緒 言

近年、ファクトリーオートメーション (FA) の分野においては中種中量生産あるいは多種小量生産を目的とする新しい生産システム形態が現れて来ている。本論文では、設備統括制御レベル¹⁾の複雑化にフレキシブルに対応するための制御ソフトウェアの構成について論ずる。以後、設備統括制御レベルをステーション制御レベルと呼ぶことにする。

ステーション制御レベルでは、非同期・並行的に動作する複数の機器を互いに同期、非他させながら制御する必要がある。従来、このような制御には、各機器の起動条件をハードワイヤードロジックで組んだりレー制御盤やラダーダイアグラムによるプログラム機能を備えた PC (プログラマブル・コントローラ) が統括制御装置として広く用いられている²⁾。

しかし、PC の本質はリレーによるハードワイヤードロジックの並列処理性をソフト的に実現したものであり、個々の制御出力を決定するためのプロセス入力 of 組み合せ (起動条件) の成立を周期的に調べる方法をとっている。この方法は、計算機の内部処理としては

† 第 9 回計測自動制御学会システムシンポジウムで発表 (昭和 58・8)

* (株)日立製作所システム開発研究所 川崎市麻生区王禅寺 1099

* Systems Development Laboratory, Hitachi, Ltd. Kawasaki

(Received April 6, 1984)

(Revised June 23, 1984)

なじみやすい方法であるが、PC では、これをそのまま制御仕様としてラダーダイアグラムで記述させるため、ステーションの動作の流れが断片的にしか記述できず、人間にとって制御仕様が非常にわかりにくくなる。このため、アプリケーションの頻繁な変更に伴うシーケンスの修正や機器の異常停止などの診断、および制御系の内部状態の再設定は、ラダーダイアグラムを組んだ人間があたっても難しい作業となる³⁾。

また、ステーションでの作業の高度化に伴い、ロボット、NC 機械などのインテリジェンスを有する機器との間の情報交換や上位コンピュータへの管理情報の提供など、ラダーダイアグラムで表現しにくい機能も多くなっている。この意味で PC はステーション制御レベルのコントローラとして十分な機能をもっているとはいえない。一方、情報処理機能の向上を目的として、最近マイクロコンピュータをベースとしたジョブショップ用計算機にセンサベースを備えた汎用マイクロコントローラが供給されてきている⁴⁾。しかし、プログラミングサポートは、ラダーダイアグラムに手続言語を組合せたレベルであり、上記の問題点を解決するものにはなっていない。

これらのハードワイヤードロジック代替のシーケンスプログラミング言語から脱却し、システム設計者の思考内容をより抽象化して表現できる方法として、非同期・並行システムの動作モデルであるペトリネット⁵⁾、⁶⁾がある。ペトリネットの表現をシーケンス制御の仕様記述に取り入れたものに GRAFCET⁷⁾がある。しかし GRAFCET は仕様記述言語であり、制御プログラムのインプリメント方法までは規定していない。ペトリネットを直接制御システムのプログラム言語に応用する試みとして、MFG⁸⁾、⁹⁾やペトリネットに基づく新しい PC¹⁰⁾の提案がなされている。これらの研究は、シーケンス制御ソフトウェアの実現方法の一つの新しい規範を与えるものである。しかし、これらはシーケンス制御系の設計解析支援や従来 PC の ON・OFF 制御機能の代替を目的としており、作業要求の識別や異常監視、条件分岐や繰返し動作の処理、大規模なネットの取り扱いなどの機能において、制御システムとしては実験システムのレベルに留まっている。

また、制御仕様の複雑化に伴うコントローラの処理ステップ数も増えており、従来 PC²⁾ (10~100 ms/実行サイクルのものが多く) 以上の高速処理を実現する方式が求められている。これに対し、ペトリネットモデルの処理方式と処理応答性の関係については、シミュレーション用の小規模なペトリネットについて、処理時間の実測結果に基づく定性的評価¹¹⁾しか行われて

おらず、実時間制御機能を付加したペトリネットに基づくコントローラについての検討はなされていない。

このような問題を解決するため本論文では、まず、ペトリネットに制御情報処理機能を付加した C-net (コントロール・ネット) による非同期・並行システムの制御仕様記述方式を提案する。つぎに、C-net の処理系としての C-net グラフ構造情報を利用した高速インタプリタ方式を提案し、実時間制御に必要な処理応答性についてモデルを用いて解析し、提案方式の実用性を明らかにする。最後に C-net に基づく制御システム SCR (ステーション・コントローラ) の構構と、実システムへの適用例を示す。

2. ペトリネットに基づく非同期・並行システムの動作仕様記述

2.1 ペトリネットモデル

非同期・並行システムのモデルとして、ペトリネットのサブクラスであるセーフなペトリネット (注1) が多く用いられる。セーフなペトリネット (以後 PN と記す) は、

$$PN = (P, T, I, O, M) \quad (1)$$

の5項組で表わされる⁵⁾。ここで P はプレースと呼ばれる状態 P_i の集合、T はトランジションと呼ばれる遷移 T_i の集合、I は遷移 T_i とその入力側プレースを対応させる入力関数 $I(T_i)$ の集合、O は遷移 T_i とその出力側プレースを対応させる出力関数 $O(T_i)$ の集合、M は PN の内部状態であるトークンの配置を表わすマージングベクトル (行ベクトル) で、 P_i にトークンがあるとき第 i 成分 1、ないとき 0 である。PN 内のトークンはつぎの規則で移動する。

(1) 入力側プレースのすべてにトークンが存在し、出力側プレースのすべてにトークンが存在しないトランジションは発火可能であるという。

(2) 発火可能なトランジションが発火すると、すべての入力側プレースのトークンは消滅し、すべての出力側プレースにトークンが発生する (これをトークンの通過と呼ぶ)。

なお、 $C = (P, T, I, O)$ を PN の構造と呼ぶ。このとき (1) 式は $PN = (C, M)$ と表わせる。

2.2 非同期・並行システムの制御仕様記述方式 C-net

PN において、機器群の単位動作 (部品をつかむ、置くなどの個々の動作ステップ) をプレース P_i に対

(注1) セーフなペトリネットとはペトリネットのプレースに一つのトークンの存在しか許さないもので解析が容易なため多く研究されている。

応させると、機器群の各单位動作の同期・排他関係がPNにおけるトークンの動作規則(1)によりプレース P_i とトランジション T_i の接続関係として陽に記述できる。また、機器群の動作の非同期・並行性はトークンの動作規則(2)によりPN中のトークンの動きとして自動的に実現される。そこで、プレース P_i で単位動作を起動したり、動作の完了を判定するための記述が行えるようにし、トランジション T_i で機器群の単位動作以外のオペレータなどの操作入力による制御規則が記述できるようにすればPNを機器群の直接制御に適用することが可能になる。以下では、PNに制御機能をもたせるための拡張を行う。

(1) プロセス入出力機能

制御系から機器群以後プロセスと呼ぶに対する接点出力や送信メッセージなどのプロセス入出力信号 x_i の集合をA、プロセスから制御系に対する接点入力や受信メッセージなどのプロセス入力信号 y_i の集合をEとする。 $(x_i$ や y_i は具体的にはビットパターンや文字列の組み合わせで定義される)。プレースの集合P、トランジションの集合TとA、Eを対応づける関数をつぎのように定義する。

$$\delta(P_i) = x_i \quad x_i \in A \quad (2)$$

$$\eta(P_i) = (y_{i1}, y_{i2}, \dots, y_{in}) \quad y_{i1}, \dots, y_{in} \in E \quad (3)$$

$$\varphi(T_i) = y_{ik} \quad y_{ik} \in E \quad (4)$$

プレース P_i にトークンが入ると、 $\delta(P_i)$ で定義されたプロセス出力信号 x_i が出力され、トークンは $\eta(P_i)$ で定義されたプロセス入力信号 $y_{i1} \sim y_{in}$ のいずれかが入力されるまで P_i にとどまるとする。これにより P_i で一つの単位動作の起動と動作結果(リターンコード)の判定が可能となる。また T_i は $\varphi(T_i)$ で定義されたプロセス入力信号 y_{ik} が入力されるまでトークンを通過させない(T_i が閉じている)という規則を新たに導入するとプロセス入力信号で制御系の動作(トークンの動き)を外から制御できる。これに対応して、 P_i 、 T_i の状態 u_i 、 v_i をつぎのように定義する。

$$u_i = \begin{cases} 0 \text{ (動作中)}; P_i \text{ からプロセス信号 } x_i \text{ を出力したとき} \\ i_n \text{ (完了)}; P_i \text{ でプロセス入力信号 } y_{in} \text{ を検出したとき} \end{cases}$$

$$v_i = \begin{cases} 0 (T_i \text{ が閉}); T_i \text{ でプロセス入力信号 } y_{ik} \text{ が検出されないとき} \\ 1 (T_i \text{ が開}); T_i \text{ でプロセス入力信号 } y_{ik} \text{ を検出したとき} \end{cases}$$

なお(3)式で $\eta(P_i)$ が複数のプロセス入力信号 $y_{i1} \sim y_{in}$ と対応する理由は、動作完了の状態として正常完了または複数の異常完了状態のうちの一つを取りうることによる。以上の定義のもとにC-net(以下CN

と記す)をつぎのように定義する。

$$CN = (C, \eta, \delta, \varphi, U, V, M) \quad (5)$$

(5)式においてCは2.1で述べたPNの構造と同一でありプロセスの動作シーケンス仕様から決まる。 η, δ, φ は前述のプロセス入出力の仕様から決まる。Mは2.1で述べたPNのマーキングベクトルと同一である。ボックス状態の集合U、ゲート状態の集合VとMを合わせてCNの内部状態と呼ぶ。CをあらためてCMの構造と呼びCの構成要素であるP、Tをプロセス入出力機能を導入した意味でそれぞれボックス、ゲートと呼ぶ。またPとTの接続関係を定義する枝をアークと呼ぶ。プロセス入出力仕様を記述する η, δ, φ はそれぞれボックス条件、制御出力、ゲート条件と呼ぶ。またCNにおけるトークンの移動規則はつぎのように定義する。

(a) すべての入力側ボックスにトークンが存在しその状態が正常完了であり、すべての入力側ボックスにトークンが存在せず、かつゲートが開状態のときそのゲートは発火可能である。

(b) 発火可能なゲートが発火すると、すべての入力側ボックスのトークンは消滅し、すべての出力側ボックスにトークンが発生する。この際出力側ボックスに対応づけた制御出力が出力されるとともに、出力側ボックスの状態は動作中となる(ただし、出力側ボックスが存在しない場合は入力側ボックスのトークンが消滅するのみとする)。

これらの拡張によりPNでプロセスの直接制御が可能となる。制御出力 $\delta(P_i)$ が定義されたボックスを特に動作ボックス(act box)と呼ぶ。

(2) データ処理機能³⁾

PNを制御仕様記述に適用するためにはプロセス入出力処理機能だけでは不十分であり、条件分岐などの情報処理機能をもたせる必要がある。これに対し、ボックスおよびゲートにつぎの諸機能をもたせる。

① 作業要求の発生検出と識別

対応づけられた作業要求信号入力が発生したとき、ボックス内に作業要求信号の内容に応じた番号を有するトークンを発生させる受信ボックス(receive box)をもうける。発生したトークンはC-netにおける初期トークンとなる。

② トークンの条件分岐

ボックス条件 $\eta(P_i)$ のそれぞれのプロセス入力 y_i の発生に対し、トークンがつぎに移動すべきボックスを指定することにより、トークンの分岐が行えるようにする。またゲートに対応づけたゲート条件 $\varphi(T_i)$ により、発火させるべきゲートを選択する。これによ

り，異常処理などへの条件分岐が可能となる。

③ タイマー機能

ボックス内に入ってきたトークンを指定時間だけ留めておくタイマーボックス (timer box) をもうける。また，ボックスに対応づけたボックス条件 $\eta(P_i)$ の許容成立待ち時間を指定し，それ以内に $\eta(P_i)$ が成立しない場合の検出機能を動作ボックスにもたせる。これらは，ディレイタイマ，ウォッチドグタイマとして用いる。

④ 類似繰り返し作業の処理機能

動作シーケンスとしては，同じでも，具体的動作のパラメータが異なる繰り返し作業（たとえば，位置の異なるネジ締作業の繰り返しなど）の要求に対し，それぞれ異なる番号をもつサブトークンを繰り返しの数だけ発生させるソースボックス (source box) をもうける。また繰り返しシーケンスの区間指定のためにカウントゲート (count gate) をもうける。カウントゲートは発火回数が指定された繰り返し回数と等しくなるまでは入出力ボックス内のトークンを消滅させる処理のみを行い，等しくなった時点で出力側ボックスへトークンを移動させる。これにより類似繰り返し動作の記述が容易にできる。

⑤ そのほかの機能

CN の階層的定義を許す階層ボックス (hierarchy box)，プロセス入出力を行わずトークンの排他区間を定義するために用いるダミーボックス (dummy box)，ユーザサブルーチンを実行するプログラムボックス (program box) がある。

CN は以上の機能を有するボックス，およびゲートにより構成される。

(3) 記述例

Fig. 1(a) はワークステーションの例である。このステーションではロボット A がパーツフィーダより部品を取り，組み立テーブル上で 2 個の部品を組む間にロボット B がネジフィーダからネジを吸着し，それぞれの部品について 4 カ所のネジ締めを行う。このステーションの動作シーケンスを CN で記述した例を Fig. 1(b) に示す。各ボックスにはボックス名を，各ゲートにはゲート番号をそれぞれ付けて記述する。受信ボックス “RCEVE” で作業要求信号を検知すると “RCEVE” にトークンが発生し，トークンが CN 中を移動する。そして，それぞれの動作ボックスに対応づけられた制御出力を行い，その完了応答としてボックス条件の成立を判定する。なおプロセス入出力の詳細内容

は省略する。ロボット A, B の同期条件はゲート 4 で，排他関係はダミーボックス “TABLE” で記述している。また，ロボット A, B の繰り返し回数と区間は，それぞれソースボックス “REPT 1” (2 個のサブトークンを発生) とカウントゲート 5 (2 回の発火をカウント)，およびソースボックス “REPT 2” (4 個のサブトークンを発生) とカウントゲート 8 (4 回の発火をカウント) の組み合わせによって記述している。

3. C-net インタプリタ

実時間制御に CN を適用するに際しては，CN の処理系である CN インタプリタでいかに高速に CN を実行するかが応答性の面で重要である。

$$CN = (C, \eta, \delta, \varphi, U, V, M)$$

において，2.2 で述べた CN おけるトークンの移動規則 (a)，(b) はつぎのように定式化される。いま， $T_i, P_i, P_j \in C$ とするとき，

$$\begin{aligned} (1) \quad & M_i = 1 \quad (\text{for } P_i \in I(T_i)) \\ & \text{かつ } U_i = 1 \quad (\text{for } P_i \in I(T_i)) \\ & \text{かつ } M_j = 0 \quad (\text{for } P_j \in O(T_i)) \\ & \text{かつ } V_i = 1 \end{aligned}$$

のとき T_i は発火可能である。

$$\begin{aligned} (2) \quad & \text{発火可能なゲート } T_i \text{ が発火するとき，} \\ & M_i = 0 \quad (\text{for } P_i \in I(T_i)) \\ & \text{かつ } M_j = 1 \quad (\text{for } P_j \in O(T_i)) \end{aligned}$$

Fig. 1 (a) A station

Fig. 1 (b) C-net representation of sequence control specification for the station

かつ $U_i=0$ (for $P_i \in O(T_i)$)

かつ $\text{out}(\delta(P_j))$ (for $P_i \in O(T_i)$)

ここで $\text{out}(\delta(P_j))$ はボックス P_j に対応づけた出力イベントをプロセスに出力することを意味する。

CN インタプリタが行うべきことは、実時間で CN 中の (1) の条件を満たすゲート (発火可能ゲート) を選び出し、選んだゲートについて (2) の処理を行うことである。このための素朴な方法は、すべてのゲートについて (1) の条件をチェックし、発火可能となったゲート T_i についてのみ (2) の処理を行うことである。しかし、この方法では、常にすべてのゲートについて発火可能の判定を行うため、CN の規模、すなわち CN に含まれるボックス、ゲートの数が大きくなると (1) の発火可能判定処理に要する時間が CN の規模にほぼ比例して増加するという欠点がある。これに対し、CN の構造 C、および CN の状態 U、M の情報を利用することにより (1) の発火可能判定処理時間を大幅に削減することができる。以下、限定スキャン方式と呼ぶ CN 高速インタプリタ方式を提案する。本アルゴリズムは Fig. 2 に示すように S1~S4 の 4 つの処理ステップからなり、それぞれ 4 つのスタック H1~H4 を用いる。準備として、スタック H1 に $M_i=1$ かつ $U_i=0$ となるボックス P_i を積んでおく。(H1 中の P_i の数を h_1 で示す)。

S1; (完了判定処理)

すべての $P_i \in H1$ について、 $U_i=1$ を満たす P_i のみ H1 から除き H2 に積む。(H2 中の P_i の数を h_2 で示す。)

S2; (判定候補ゲート登録処理)

すべての $P_i \in H2$ について、 $M_i=1$ かつ $P_i \in I(T_j)$ あるいは $M_i=0$ かつ $P_i \in O(T_j)$ を満たす T_j を求め H3 に積む P_i を H2 から除く。なお P_i は少なくとも一つの出力ゲートに接続しており、 T_j は必

ず求まる。(H3 中の T_j の数を h_3 で示す。)

S3; (発火可能判定処理)

すべての $T_j \in H3$ について、CN におけるトークンの移動規則 (1) を満たす T_j (発火可能ゲート) のみ H3 から除き、H4 に積む。(H4 中の T_j の数を h_4 で示す。)

S4; (発火処理)

すべての $T_j \in H4$ について、CN におけるトークンの移動規則 (2) を適用し、 $P_i \in I(T_j)$ を H2 に積み $P_i \in O(T_j)$ を H1 に積んだ後、 T_j を H4 から除く。以上の S1~S4 の処理では、ゲート T のうちで入力ボックス $I(T_j)$ の少なくとも一つのボックスのボックス条件が新たに成立したか、あるいは出力ボックス $O(T_j)$ の少なくとも一つのボックスで新たにトークンが消滅したゲート T_j のみについて CN におけるトークン移動規則 (1) を判定する。これにより、プロセスの動作によりトークンの通過が起きる可能性の生じたゲートのみについて発火可能判定を行うため、CN インタプリタの処理時間が削減される。

4. CN インタプリタの応答性評価

処理すべき CN についてつぎの量を定義する。

$$\varepsilon = \frac{\text{総アーク数}}{\text{総ボックス数}} = \frac{m_a}{m_b} \quad (6)$$

$$\sigma = \frac{\text{総ボックス数} \times 2}{\text{総ゲート数}} = \frac{2m_b}{m_g} \quad (7)$$

ε は単位ボックス当り平均入出力アーク数、 σ は単位ゲート当り平均入出力ボックス数である。(7) 式の分子が 2 倍になっているのは、一つのボックスを入力ボックスおよび出力ボックスとして 2 回数えることを意味する。さて、 h_1 は、CN 中の $U_i=0$ なるボックス P_i (動作中ボックスと呼ぶ) の最大数である。 h_1 は、プロセス内で並行に動作している機器の最大数と見な

S1: completion judgement of activated boxes

S2: listing up of candidate boxes for enable test

S3: enable testing

S4: firing

H1: stack of activated boxes

H2: stack of completed and marking changed boxes

H3: stack of candidate gates for enable test

H4: stack of enable gates

ε : average number of input and output arcs per box

σ : average number of input and output boxes per gate

μ : completion rate of activated boxes per execution cycle

μ' : firing rate per execution cycle

Fig. 2 Configuration of a limited scanning method

せる。動作中ボックスのうち、S1～S4の処理（1サイクル）が行われる間に完了条件 η が成立するボックスの割合を μ （完了率）とする。 h_2 は、1サイクル中に、新たにトークンが発生したか、あるいは消滅したボックスの平均数である。 h_3 は1サイクル中に発火可能判定処理S3の判定候補となるゲートの平均数である。 h_4 は発火可能判定の結果、トークンの移動が可能と判定されたボックスの平均数である。 $\mu' = h_3/h_4$ を発火率と呼ぶ。判定候補ゲート登録処理S2では、トークンが新たに発生したボックスの出力側ゲートと、トークンが新たに消滅したボックスの入力側ゲートをすべてH3に積むから、 h_2 と h_3 の間には $h_3 = \varepsilon h_2/2$ の関係がある。ゲート当りの平均入力ボックス数と平均出力ボックス数はほぼ等しいと見なせるので、発火処理S4では、 $\sigma h_4/2$ のボックス中のトークンが消滅し、 $\sigma h_4/2$ のボックスにトークンが発生する。これらはそれぞれ h_1 および h_2 に戻される。

さて、プロセス内の機器の単位動作時間（制御系から制御指令 δ が出力されて完了条件 η が成立するまでの時間）は通常S1～S4の1サイクルの処理時間に比べずっと大きいため、 $h_1 \sim h_4$ は平衡状態にあると考えてよい。この場合、次式が成り立つ。

$$h_1 = h_1(1 - \mu) + \frac{\sigma}{2} h_4 \quad (8)$$

$$h_2 = h_1 \mu + \frac{\sigma}{2} h_4 \quad (9)$$

$$h_3 = \frac{\varepsilon}{2} h_2 \quad (10)$$

$$h_4 = \mu' h_3 \quad (11)$$

(8)～(11)式が $h_1 = h_2 = h_3 = h_4 = 0$ 以外の有意な解をもつためには、(8)～(11)式の係数行列式=0、すなわち次式の成立が必要である。

$$\begin{vmatrix} -\mu & 0 & 0 & \sigma/2 \\ \mu & -1 & 0 & \sigma/2 \\ 0 & \varepsilon/2 & -1 & 0 \\ 0 & 0 & \mu' & -1 \end{vmatrix} = 0 \quad (12)$$

(12)式より $\mu(1 - \mu'\varepsilon\sigma/2) = 0$ を得る。 $\mu \neq 0$ より

$$\mu' = 2/\varepsilon\sigma \quad (13)$$

が成立する。(13)式を平衡条件と呼ぶ。このとき(8)～(11)式および(13)式より h_2, h_3, h_4 を h_1 を用いて表わすと次式を得る

$$h_2 = 2\mu h_1 \quad (14)$$

$$h_3 = \varepsilon\mu h_1 \quad (15)$$

$$h_4 = 2\mu h_1/\sigma \quad (16)$$

S1～S4の1実行サイクルの処理時間 Z は、S1, S2の1ボックス当り処理時間を t_j, t_i 、S3, S4の1ゲ-

ート当り処理時間を t_e, t_f とすると次式で与えられる。

$$\begin{aligned} Z &= h_1 t_j + h_2 t_i + h_3 t_e + h_4 t_f \\ &= (t_j + (2t_i + \varepsilon t_e + 2t_f/\sigma)\mu) h_1 \end{aligned} \quad (17)$$

t_j, t_i, t_e, t_f は任意のCNについて、CNインタプリタの各処理ステップS1～S4の処理時間を実測して求めることができる。 ε, σ の値はCNの構造から決まる。また μ, h_1 の値はプロセス内の機器のスピード、および並行に動きうる機器数から決まる。以後 h_1 を P で表わし、 ε, σ をCNの構造パラメータ、 μ, P をプロセスの構造パラメータと呼ぶ。

いくつかFAのプロセスの制御仕様をCNで記述したところ、ほとんどのCNが $\varepsilon \leq 4, \sigma > 2$ で記述可能であることが判明した。例として、モトローラ系16ビットマイクロプロセッサM68000上でS-PL/H¹⁶⁾で開発したCNインタプリタにおける $t_j \sim t_f$ の実測値を用い、 $m_a = 1, 100, m_b = 550, m_c = 500$ すなわち $\varepsilon = 2, \sigma = 2.2$ のCNを処理する場合のCNインタプリタ処理時間 Z を計算すると、

$$Z = (0.91 + 3.12\mu)P \text{ [ms]} \quad (18)$$

となる。

このCNについて $\mu = 1, 0.5$ の両ケースについて、 P を2から12まで変えながら、 Z を上記のインタプリタによって実測した結果をFig. 3に示す。実測結果は(18)式で与えられる直線と良く一致した。

μ の値は、C-netインタプリタの起動周期 τ と機器の平均単位動作時間 u により $\mu = \tau/u$ で与えられる。例えば50ms周期で起動されるC-netインタプリタで、平均単位動作時間0.5秒の機器（高速な機器に属する）を制御する場合は $\mu = 0.05/0.5 = 0.1$ であり、(18)式によれば $P = 10$ すなわち並行動作する機器10

Fig. 3 Response time of C-net interpreter

台の制御仕様を記述した CN を処理する場合の時間は $Z=12.2[\text{ms}]$ と求められる。これは、従来のライン制御用 PC で約 3~4K ステップのラダープログラム（比較的小規模のライン制御レベル）を処理する時間とほぼ等しい。従来の PC が、常にすべての処理ステップを実行する方式のためその処理時間が処理すべきステップ数にほぼ比例するのに対し、本方式では処理時間が C-net の規模 (m_a, m_b, m_p) に直接影響を受けないため、ステーションの制御ステップが大規模化するほど、本方式がより有利となる。

5. CN ベースコントローラ SCR (ステーションコントローラ) とその適用

5.1 SCR の構成

Fig. 4 にステーションコントローラ (SCR) のソフト構成を示す¹⁷⁾。SCR は CN エディタ, CN モニタ, CN 制御データ, CN インタプリタ, およびプロセスインターフェースからなる。

CN エディタ¹⁴⁾はプロセスの制御仕様を記述した CN をグラフィック端末より図形イメージで階層的に定義できるため、大規模な CN を容易に作成できる。CN モニタは CN 中のトークン配置とボックスの状態をグラフィック端末上にオンラインで表示する¹³⁾。これにより、制御系の内部状態が容易に把握できる。

CN 制御テーブルは、CN エディタで入力された CN データを CN インタプリタが実行可能な内部形式¹⁵⁾で格納しておくテーブルである。これらの構成を **Table 1**, **Table 2** に示す。

Fig. 4 Configuration of SCR

Table 1 Gate information

input boxes list	
output boxes list	
pointer to associated input events (gate condition)	
maximum count number for count gate	counter for count gate

Table 2 Box information

input gates list	
output gates list	
pointer to associated output events (control outputs)	
pointer to associated input events (box condition)	
maximum execution time	timer counter

CN インタプリタは 3 章で述べた方式により、CN 内のトークンを移動させながら、プロセスインターフェースを介して機器群の同期・排他制御を行う。CN インタプリタは M68000 上に S-pL/H¹⁶⁾ でインプリメントしており、プロセスインターフェースとしてデジタル入出力 (DI・DO) とシリアル通信ポート (RS 232-C) をサポートしている。

5.2 実システムへの適用例

SCR をロボットによる部品組立ステーションの制御システムに適用した。Fig. 5 にそのシステム構成を示す。ロボットは 8 組のハンドを取り換えることにより、7 種の部品の組立を行う。このシステムの制御仕様を CN で記述したところ、ボックス数、ゲート数とも約 1,000 の規模の CN となった。CN の構造パラメータは $\varepsilon=2.0$, $\sigma=2.1$ であり、プロセスの構造パラメータは $\mu=0.02$, $P=7$ である。SCR は各機器の制御装置 (PC, マイコンなど) に対し、DI, DO 各 100 点を介して起動制御を行う。CN インタプリタの起動は、リアルタイム OS 下で 100ms 周期で行っているが、インタプリタの処理時間は 10ms 以下であり、十分な処理応答性を実現している。また、制御

Fig. 5 Application of SCR to Parts Assembly Station

プログラムの作成期間は、従来 PC を用いた場合と比べ約 50% 削減された。C-net エディタ、C-net モニタの機能により、従来時間のかかっていた異常の同定や制御仕様の変更にも迅速に対処できている。

6. 結 言

非同期・並行システムの抽象デモルであるペトリネットをステーション制御レベルのプロセス制御に適用するためにペトリネットにイベント処理機能、情報処理機能をもたせた C-net による制御仕様記述方式を提案した。また、C-net に基づく制御方式として実時間に十分な応答性能な達成するための C-net 高速インタプリタ方式を提案し、評価モデルに基づき、処理応答時間の評価式を明らかにした。この結果、平均単位動作時間 0.5 秒で並行に動作する機器 10 台からなるプロセスの制御仕様を記述した C-net を、その規模によらず 15 ms 以内で処理しうることを示し、複雑な制御仕様をもつ FA ステーション制御に十分適用できることを示した。

C-net にもとづく制御装置 SCR をロボットによる部品組立ステーションの制御に適用した結果、制御ソフトの開発工数の点で従来 PC 方式に比べ約 50% のマンパワー削減を実現し、その後のメンテナンスにも容易に対処できている。

最後に、本研究に関しご討議いただいた東京工業大学市川惇信教授に対し深く感謝の意を表します。また本研究の機会をお与え下さった、(株)日立製作所コンピュータ事業本部長三浦武雄博士、同システム開発研究所所長川崎惇博士、並びに本研究に関しご指導いただいた同システム開発研究所第 1 部部長春名公一博士に対し感謝致します。また SCR の開発および実システムへの適用に関しご討議、ご協力いただいた同大みか工場、土浦工場、習志野工場並びにシステム開発研究所の方々に感謝致します。

参 考 文 献

- 1) 成田：FA における計算・通信技術、計測と制御、22-11, 33/36 (1983)
- 2) 関口：PC における言語の動向、電気学会雑誌、101-10, 35/42 (1981)
- 3) 関口：シーケンス制御の現状と課題、オートメーション 27-5, 10/14 (1983)
- 4) 大西：PC が制御用パソコンと合体、日経メカニカル、1983年 2 月号、86/90 (1983)
- 5) J. L. Peterson: Petri-Net Theory and the Modeling of Systems, prentice Hall (1983)
- 6) 市川、小林：事象駆動形システムの表現と制御、計測と制御、21-10, 929/938 (1983)
- 7) 戸塚：シーケンスの新しい表現法—仏 ADEPA の新提案の紹介、機械設計、25-1, 127/132, 25-2, 111/125 (1981)
- 8) 長谷川、高橋：非連続システムのためのマーク流れ線図の提案、計測自動制御学会論文集、20-2, 122/129 (1984)
- 9) 高橋、長谷川：MFG に基づいたプログラマブル・コントローラ、第 21 回 SICE 学術講演会予稿集、No. 1404 (1982)
- 10) D. Chocron and E. Cerny: A Petri-Net-Based Industrial Sequencer, Proc. IEEE 1980 Int. Conf. and Exhibition on Industrial Control and Instrumentation, 18/22 (1980)
- 11) M. Silva and S. Velilla: Programmable Logic Controllers and Petri-Nets A Comparative Study, IFAC Software for Computer Control, 83/88 (1982)
- 12) 楠田、吉田：PC のアーキテクチャとソフトウェア技術、電気学会雑誌、102-4, 7/11 (1982)
- 13) 薦田、村田ほか：FA 用ステーションコントローラ(SCR)の提案、第 9 回システムシンポジウム予稿集、261/266 (1983)
- 14) 薦田、片山ほか：作業ステーション制御用ステーションコントローラ(SCR)用画面エディタ、第 28 回情報処理全国大会、IG-9 (1983)
- 15) M. Silva and R. David: SYNTHESE PROGRAMME-EE. DES AUTOMATISMES LOGIQUES. DECRITS. PAR RESEAUX DEPETRI, RAIRO-Automatize 13-4, 369/393 (1979)
- 16) 神野ほか：16ビットマイコン用高級言語 S-PL/H の開発と 68000 システムソフトウェアへの適用、マイクロコンピュータ応用国際コンファレンス '82 ⑤ 5-1, 193/202 (1982)
- 17) 松本ほか：自立分散 FA 制御システム、昭和 58 年電気学会東京支部大会講演論文集、2-S.7-5 (1983)