制御系設計支援システムにおける マクロデータフローモデルを用いた統一プログラミング環境

5ZB－08

上野 晃嗣, 合田 憲人, 原 辰次
東京工業大学総合理工学研究科[1]

1 はじめに

現在、制御工学は自動車、航空機からハードディスクドライブまでのあらゆる製品に応用され、不安定特性の解消、応答改善、システムの自動調節などを可能にしている。制御系解析・設計作業とは、これらの制御対象システムの特性を解析し、これに対して最適なコントローラを設計する作業であり、現在では支援CADツールの使用が欠かせないものとなっている。

これらのツールには用途と作業フェーズに応じて多様なものが存在し、それぞれ扱うデータ構造が異なるが、ユーザはそれらを用いた処理を複数組み合わせて実行する必要がある。また、処理順序や与えるパラメータなどについて試行錯誤的な作業を行う必要がある。

従来、このような作業をCUI(Character User Interface)を用いて行う方法が多かった[1]が、この方法には以下のような問題があった。

- 作業が複雑で困難
- ツール間のデータ形式の変換をユーザが行う必要がある

一方、ソフトウェアのGUI化は視認性、操作性の向上に大きな効果があることは数々の研究で確かめられている。しかし、制御系CADシステムのGUI化は特定のツールに特化して行われ、他のツールとの組み合わせを困難にするなどの、柔軟性の低下を招いている。

また、処理の実行手順を試行錯誤的に決定するための支援機構は今まで提案されてこなかった。

これらの問題に対し、本稿では制御系解析・設計支援CADシステムにおける統一的プログラミング環境を提案する。提案するプログラミング環境は従来のCADツールとは独立して存在し、以下のような特徴を持つ。

- ビジュアルプログラミングの導入によるユーザの作業の容易化
- 異なるツール間でのデータ形式の自動変換
- 様々な制御系解析・設計ツールとの柔軟な組み合わせ
- 再実行機能による試行錯誤的作業の支援

2 統一プログラミング環境

本章では、本稿が提案する統一プログラミング環境について述べる。

制御系解析・設計作業では、ユーザは、図1に示すような複数のツールの実行手順を決定し、解析・設計作業を進める。

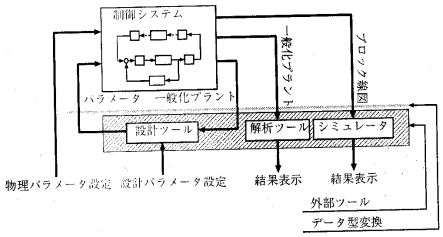

図1: 制御系解析・設計作業

例えば図1の例では、ユーザは、設計ツールによりパラメータを制御システムに与え、それによる特性の変化を解析ツールやシミュレータで観測する。次にその結果によって再びパラメータを変更する等の処理を繰り返す。

提案するプログラミング環境では、このような作業の効率を高めるため、独自に提案するマクロデータフローモデルを用いたビジュアルプログラミング機能を提供する。このモデルは、任意の部分の再実行が可能であり、試行錯誤的な作業を効率良く支援する。

図1にあるようなツールは、多くの場合独立した外部プログラムとして用意されることが多いが、本プログラミング環境では、これらの外部プログラム(ツール)を呼び出すことが可能である。また、その際のデータ形式の変換は自動化されているため、ユーザはデータ変換などについての繁雑な作業の必要が無くなる。

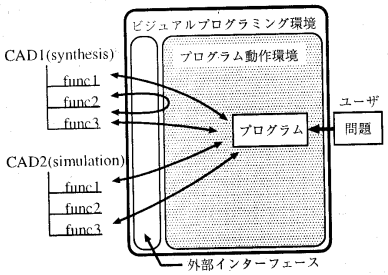

図2: プログラミング環境の構成

このような外部ツールとの柔軟な組合せを実現するために、提案する環境は、図2に示すように、外部機

[1]A Unified Programming Environment for Computer Aided Control System Design, Using a Macrodataflow Model.
Kouji UENO, Kento AIDA, Shinji HARA
Interdisciplinary Graduate School of Science and Engineering, Tokyo Institute of Technology.

能を利用するための外部インターフェース部と、利用できる機能を組み合わせて作業をプログラムし、実行するための、プログラム動作環境の二つに分かれた構成を取る。

3 マクロデータフローモデル

本章では本環境において用いるマクロデータフローモデル [2][3] の構造と動作について述べる。

提案する環境においてユーザはビジュアルプログラミングによって作業に必要な各処理の実行手順の定義を行う。また、処理結果に応じて設計パラメータや実行手順に対して様々な修正を行いながら、繰り返しプログラムを実行する。

これを実現するためのビジュアルプログラムの内部モデルは以下の要件を満たさなければならない。

- ユーザに提示するビジュアルプログラムに近い形であり、修正が容易
- 各処理の効率的な実行

これらの必要性に従い、本環境では独自のマクロデータフローモデルを用いる。

3.1 モデルの構成

本マクロデータフローモデルでは、ノードであるタスクと、有向エッジであるタスク間関係によって構成される (図 3)。エッジの矢印の元を上流、先を下流と呼ぶ事とする。

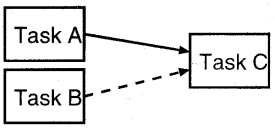

図 3: タスクとタスク間関係

タスクはプログラム切片を表し、本環境では主に単一の外部ツール呼び出しに対応する。

タスク間関係はタスク間の実行順序を表すもので、実行制約 (図 3 における点線) とデータ依存 (図 3 における実線) に分けられる。このうち、実行制約は、タスクが実行されるか否かを表す条件を示し、データ依存は、タスク内で使用されるデータの依存関係を表す。

タスク、タスク間関係などの各要素には、待機・実行・データ転送・実行抑制など、それぞれ複数の状態が定義されている。要素の状態は隣接要素の状態に応じて遷移し、これによってタスク実行の制御を行う。実行制約については、上流タスクの条件分岐によって状態を決定するが、データ依存は基本的に上流タスクの状態をそのまま反映する。タスクは、**上流からの入力データが揃い、実行制約により実行が抑制されていない場合**、実行を開始する。

3.2 再実行支援

本モデルは従来のデータフローモデルを用いた並列処理系 [3] などとは異なり、任意のタスクからのプログラムの再実行が可能であり、さらにタスクの再実行回数の最適化を実現している。

この最適化はデータフロー全体の構造に依存する形ではなく、タスク、タスク間関係などの各要素の独立動作の協調によって実現される機能なので、ユーザが、プログラム実行中にプログラムの修正 (要素の組み変え)、パラメータ変更、再実行などを自由に行なえるという利点がある。

本モデルでは、プログラムの再実行とは**任意のタスクに対してユーザが再実行指令を与える**ことと定義する。

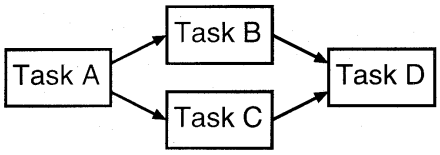

図 4: プログラム例

ここで図 4 は、既に 4 つ全てのタスクが一度実行済のプログラムであるとする。Task B に対して再実行指令が与えられた場合、Task D は、Task B の実行が終了して更新データが送られ次第、一度実行される。この時、Task D は Task C からの入力値については前回実行時の値を用いる。これを実現するため、**各タスクは毎回の実行時に全ての入力値を保存するものとする。**

一方、同図において Task A に再実行指令が与えられた場合、Task D は Task B, C どちらかの更新データが送られた時点で再実行されるのではなく、Task B, C 双方の実行結果を待ってから一度だけ実行されなければならない。これを実現するため、**各タスクは実行可能性の有無を表す状態を持ち、それをタスク間関係を通じて伝搬させる。また、各タスクは、上流のタスクが実行される可能性が存在する間は再実行を行わない。**

4 まとめ

本稿では、独自のマクロデータフローモデルを用いて制御系設計支援システムのための、統一プログラミング環境を提案した。

今後の課題としては、提案手法の実問題を用いた性能評価が挙げられる。

参考文献

[1] The MathWorks, Inc. MATLAB User's Guide. 1997.

[2] R. G. Babb II: Parallel Processing with Large-Grain Data Flow Techniques, IEEE Computer, 17, 7, 55-61, 1984.

[3] 本多, 岩田, 笠原: Fortran プログラム粗粒度タスク間の並列性検出手法, 電子情報通信学会論文誌, J73-D-I, 12, 951-960, 1990.