コード視覚化手法を用いたMPIプログラム開発環境のユーザインタフェース

副島直樹† 桑原寛明† 國枝義敏†
†立命館大学情報理工学部

1 はじめに

クラスタシステム普及の反面，その上で動作する並列プログラムを組むことのできる技術者が不足している．その主な原因として，複数のノード間での通信を含む並列プログラムの複雑な動作を，ソースコードより読み取ることが困難であるという点が挙げられる．プログラムの動作イメージを持たずに開発を進めることは困難である．そのため，ユーザが常に並列プログラムの全体像や動作イメージを持ちつつ開発できる機能が，開発環境に求められる．

ユーザのプログラム理解を支援する手法の1つとしてプログラムの可視化が用いられる．プログラムの可視化により，プログラムの処理ではなく，流れや構成の表現に特化した出力が得られる．可視化図形に可視化元のソースコードを関連付けて管理することにより，ユーザは可視化図形より指定した箇所のソースコードを参照できる．これにより，可視化表現を用いると同時に処理を示す事ができる．

本稿で提唱する手法ではMPIプログラミング初心者へのアプローチに重点を置く．可視化手法を用い，MPIプログラムの特殊なプログラム構造を開発者に意識させない出力を行う．これにより並列プログラムの全体像を簡潔な図表でユーザに与える．さらに通信の記述をサポートすることにより，MPIプログラミング初心者の負担を減らす．

本研究ではソースコードの静的解析により，ツリー状のデータ構造にプログラム解析情報を格納し，実際のソースコードとリンクしたデータ構造をシステム内に持たせることにより，プログラムを表わす簡潔な図形表示上でのコーディングを可能にする直感的なGUIを構築する．

2 関連研究

プログラムの可視化手法を用いた既存研究として[1], [2]等が挙げられる．これらの研究は本研究と同様にプログラムの構造や動作の理解支援を目的としている．本研究では可視化により得られた図形を出力のみに用いず，入力インタフェースとしても利用する．

A Graphical User Interface of MPI Program Development Environment using Code Visualization.
Naoki Soejima†, Hiroaki Kuwabara†and Yoshitoshi Kunieda†
†College of Information Science and Engineering, Ritsumeikan Univ.

また，グラフィカルな操作によりMPIプログラミングをサポートする研究の先駆けとしては[3]がある．本研究ではプログラムの可視化により得られる図形表示を操作の基盤とした．これにより，プログラムの全体像を見通しつつ開発を進められる環境の実現を図った．

3 MPIプログラミングに伴う問題点

MPIプログラムではPCクラスタの各ノード上で実行される並列タスクごとにランクと呼ばれる識別番号が割り当てられる．MPIプログラミングにおいては，このランクを保持する変数を制御文の条件式に用いることにより，並列タスク個別のプログラムコードを記述することができる．その結果，これらの個別の処理を記述したコードが同じファイルに混在して記述されることとなるので，MPIプログラムのソースコードは煩雑で読みにくくなる．

MPIプログラムの開発時に，開発者は通信に関わるバグの解決に最も多くの労力を費やすことが多い．様々な要因により通信時のバグは発生しうるが，通信関数の対応関係を明確にし，データの流れを把握することにより，バグの回避と円滑なデバッグ作業が期待できる．

4 可視化手法によるMPIプログラムへのアプローチ

可視化手法を用いることにより，前節に示したMPIプログラミングにおける問題点の解決をする．提唱するコード視覚化GUIの実装する機能を以下に示す．

(1) ノード別に実行されうるコードの表示切替

本システムはユーザの選択したランクでのみ実行されうるコードに絞って表示を行うことにより，ユーザのプログラム理解を支援する．ユーザプログラムの静的解析によりランクを保持する変数を割り出し，その変数が用いられる制御文の条件式の演算を行うことにより，指定されない実行コードを枝切りした上で描画を行なう．

図1に特定ノードを指定しない場合，図2にランク=0のノードを，図3にランク=1のノードをそれぞれ指定した場合のサンプルプログラムの描画例を示す．図中の縦軸は解析元ソースプログラムの行数を表わす．また，各図形の説明と分岐文の条件式を図1の図中に示す．これらの注釈は実際の出力には含まれない．

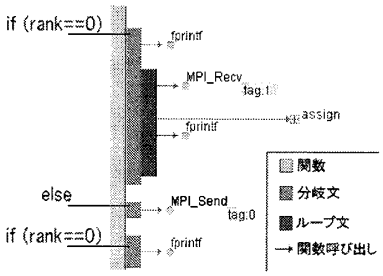

図 1: ランク指定無しの場合の描画と注釈

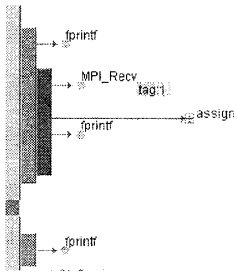

図 2: ランク=0 指定時

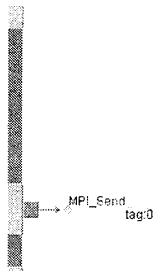

図 3: ランク=1 指定時

指定ノードで実行されない箇所は表示されず，また非表示箇所の編集を禁止している．また，静的解析によりランク保持変数の判定を行なうが，ユーザが任意の変数のランク保持判定を切り替えられる機能を提供する．

(2) 通信関数の挿入と管理

本 UI では簡潔なグラフィックで表されたプログラム上の任意の場所に通信関数を挿入する機能を提供する．

図形表示上の任意の場所をクリックすることにより通信関数挿入用のダイアログが呼び出される．次に任意の通信関数を選択し，その引数をフォームに入力することで操作は完了する．また，引数入力フォームに挿入位置のスコープ中の引数のデータ型に合致する変数をリストアップし，ユーザに選択させることにより，ユーザの入力負担を軽減すると共に間違った入力を防ぐ．

また 1 対 1 通信の識別番号である引数 tag をシステムで管理することにより，通信の対応関係を表示している．これにより 1 対 1 通信におけるバグを減らすことを意図している．新しい通信関数の挿入時に tag の値はシステムによって常に新しい値が割り当てられる．さらにプログラム中の通信関数の tag の値をユーザ操作により更新する機能を提供し，ユーザによる柔軟な操作を許している．

5 実装

本システムは構文解析器と UI 部の 2 つのプログラムにより構成される．図 4 に処理フローを示す．システムはまず，解析対象となるプログラムソース群を受け取る．はじめに構文解析器が渡されたソースファイル群の解析を行い，中間ファイルを出力する．次に UI 部がソースファイル群と，さらに対応する中間ファイル群を読込み，ツリー状のデータ構造を構成し，格納されたデータを元にプログラムを簡潔なグラフィックで描画する．UI の提供する機能によりユーザはデータツリー上の特定の情報を参照できると共に対応箇所のソースを編集できる．

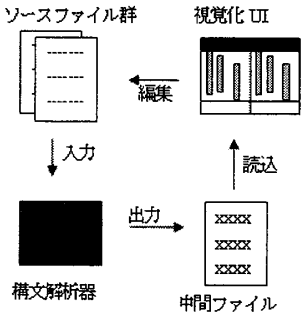

図 4: システム構成と処理フロー

6 おわりに

本研究ではプログラムの全体像を図示することにより，プログラム全体のイメージを持ちつつ，開発作業を進められる環境の構築を目的に，UI の構築を行なった．

現状ではユーザの直接入力によりソースファイルに変更が発生した場合，更新されたソースファイル全体の構文解析から再開せねばならず，ツリー構造の特性を活かしきれていないため，ソースコードの部分変更への対応が今後の課題として挙げられる．

参考文献

[1] 市川至，小野越夫，毛利友治: “プログラム可視化システムの概要”，情報処理学会第 38 回全国大会論文集，pp. 1237-1238，1989 年前期

[2] 佐藤竜也，田中二郎，志築文太郎，三末和男，高橋伸: “GUI を持つプログラムの理解支援のための可視化システム”，平成 18 年度 筑波大学第三学群情報学類 卒業論文，2007 年 2 月

[3] 酒寄保隆，三浦元喜，田中二郎: “GRIX: 並列プログラミングにおけるプロセッサ間通信のコーディング支援システム”，日本ソフトウェア科学会第 15 回大会論文集，pp. 381-384，1998 年 9 月