

ビジュアルデバッガにおけるデータ構造の表現法[†]

2E-5

田中 義一 上原 稔 森 秀樹^{††}東洋大学 工学部 情報工学科^{†††}

1 はじめに

近年益々高度化されるハードウェア技術に対して、その性能を十分に引き出すための並行処理への必要性が高まっている。しかし、並行プログラムの開発は逐次プログラムに比べて困難であり、特にデバッグにおいてはその作業を円滑に行えるような環境が必要とされる。我々のプロジェクトでは並行プログラミング言語 NET/C[3]の開発を行うと共に、その生産性を高めるためのプログラミング環境の開発も進めている。この研究の一環としてエディタとデバッガを統合した視覚的・対話的システム[4]を開発した。

並行プログラムのデバッグ作業においてプログラムの動作理解を円滑に行うためにはデータおよびその構造の変化を迅速に把握することが重要となり、その手段として情報の視覚化が要求される。しかし、データ構造の視覚化は要素数の増加や構造の複雑化によって表示領域や描画速度の面で無理が生じてくる。

本稿では、並行プログラミング言語 NET/C の視覚的プログラミングシステムの一つとして定義されているビデオ形式のビジュアルデバッガに適合し、なおかつ大規模で複雑なデータ構造に対してもマルチプルビューやスケール機能により柔軟に対応できるような表現法を紹介する。

2 NET/C

NET/C はネットワークでモデル化される問題を自然な形で記述することができ、なおかつそれを並行実行できる並行プログラミング言語である。NET/C のモデルはアクター(処理単位)とストリーム(通信路)から形成されるネットワークモデルで表現され、その処理はネットワークで構成されるアーキテクチャ上において最大限の威力を発揮することができる。

また、NET/C では仮想機械上のシミュレーションにより処理を実現することによって機種に依存しない汎用性とデバッグに有利な環境を提供している。

3 NET/C のプログラミング環境

NET/C のプログラミング環境では、プログラムの記述からデバッグに至るまでの一連の作業環境を GUI により実現している。プログラムの記述はアクターのシンボルを配置し、互いの関係(ストリーム)を定義することで行なう。デバッグ時にはストリームに流れるデータをアニメーションで表示することでアクター間のやり取りを直観的に認識することができる。

本システムにおけるアニメーションの操作は家庭用ビデオに類似したものであり、実行の再生、停止、逆再生、および速度調整を極めて簡単に行なうことができる。

4 情報の視覚化

NET/C で取り扱うデータには数値、文字、配列、構造体など様々な形式のものがあり、デバッグ時にはそれらの変化を監視する必要がある。このうち数値や文字などは値が直接ストリームを流れることになるが、配列や構造体の場合にはそれを示すポインタの流れとして取り扱われる。そのためストリーム内のデータを単純に表示しただけでは、それがポインタの場合にはその内容までは確認することができない。また、データがリスト構造になっている場合にはその内容だけでなく構造の変化を知ることが重要となる。

そこで、本システムではデータ構造を表示するための機能として VIPS[1] を参考にしたツリー表示と、それに加えてテーブル表示機能を用意し、更にスケール機能やヒストリー機能を追加することでスケール問題や動的なアニメーション表示にも対応した環境を実現した。

- 概略ツリー表示(図 1-A)
データの全体的な構造のみをツリー状に表示する機能
- 詳細ツリー表示(図 1-B)
データの構造に加えて各ノードの詳細情報を表示する機能
- テーブル表示(図 2)
線形データにおける各ノードの内容をテーブル状に表示する機能

[†]Representation of Data Structure for Visual Debugger^{††}Yoshikazu TANAKA, Minoru UEHARA, and Hideki MORI^{†††}Department of Information and Computer Sciences, Toyo University

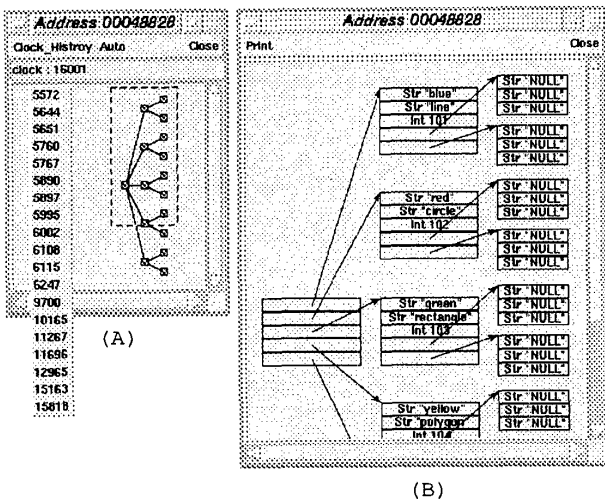


図 1: ツリー表示

概略ツリー表示はストリーム上のアドレスを選択することで表示され、トレース時にはその構造の変化とともに再描画が行なわれる。詳細ツリー表示は概略ツリー上の任意のノードを選択することで、そのノードをルートとした詳細表示が行なわれる。

Figure 2 is a table window titled 'Address 00048828'. It shows a table with columns for memory addresses and data fields. The data is as follows:

Address	Field 1	Field 2	Field 3	Field 4	Field 5
Address 00049868	Str *blue	Str *line	Int 101	Array 00051288	Array 00049868
Address 00051328	Str *red	Str *circle	Int 102	Array 00051508	Array 00051460
Address 00051418	Str *green	Str *rectangle	Int 103	Array 00051640	Array 00051560
Address 00051908	Str *yellow	Str *polygon	Int 104	Array 00051788	Array 00051756
Address 00051988	Str *brown	Str *text	Int 105	Array 00051928	Array 00051810

図 2: テーブル表示

プログラムで線形データを使う場合、線形に繋がれたデータは同一の構造体によって作られることが多く、各構造体の内容を比較する場合にはツリーよりもテーブルによって表示した方が対応する要素同士を比較し易く、そしてまた表示領域も節約することができる。

4.1 スケーリング機能

線形データのツリー表示では、構造体中の要素数の増大に伴って必要となる表示領域が拡大し描画速度も低下してしまう。そこで本システムでは要素数の大きさによって限定した表示を行なう。要素が特定数（ユーザ定義）を越えるノードについてはリストボックスを使って表示し、その子ノードの表示は行なわない。子ノードはリストボックスからポインタを選択することによって表示する。

また、ツリーの深さについても同様に、その大きさを指定することで限定した表示を行なうことができる。

4.2 ヒストリー機能

データ構造を理解するための近道は、その構造の変化を把握することである。ヒストリー機能は、構造に変化のあったクロックを記録しておき、後からでも即座に過去の状態を確認できるようにするための機能である。また、記録されている状態を連続的に表示することも可能である。

4.3 インスペクター

本システムでは、プログラムの実行を中断した時にデータの内容を対話形式で変更するための機能としてインスペクターを実装した。これにより、プログラムの繰り返し実行による時間的な無駄を大幅に削減することができる。

データの変更は、ストリーム上もしくは詳細ツリー表示やテーブル表示の中で変更したいデータを選択し、開かれたダイアログの中で新たな値を入力することで行なわれる。

5 まとめ

本研究では、NET/C のプログラミングシステムにデータ構造の表示機能を追加し、それに対してスケーリング機能やヒストリー機能を付加することで動的なアニメーション表示やスケーリング問題にも対応したデータ構造表示を実現した。

今後は、特定のストリームに着目したデータ表示、インスペクターの機能強化、そしてブラックボックスの実装など、より対話的な機能の実現を考えている。

参考文献

- [1] Takao Shimomura and Sadahiro Isoda. Linked-List Visualization for Debugging. In *IEEE Software*, pp. 44-51, May 1991.
- [2] Sven Moen. Drawing Dynamic Trees. In *IEEE Software*, pp. 21-28, July 1990.
- [3] Minoru Uehara. NET/C: Toward the Fine Grained UNIX-Like OS. In *OOS'94*, pp. 453-456, Springer-Verlag, December 1994.
- [4] 田中 義一, 他. NET/C のためのビジュアルデバッグ環境の開発. 日本ソフトウェア科学会第 12 回大会論文集, pp. 129-132, September 1995.