分散プログラムの動作理解のための分散プロファイラ

3M−5

新垣 紀子　山崎 憲一　天海 良治
NTT ソフトウェア研究所

1　はじめに

コンピュータネットワーク技術の発達に伴い、複数のマシン上で複数のプロセスが通信をし合いながら1つの処理を行なう分散プログラムの開発の必要性が高まっている。しかし分散プログラムの開発は、プロセスや通信路の生成消滅など、プログラムの動作状況が動的に変化するので、従来の単独マシンのプログラム開発より、はるかに難しい。この問題を解決する手段として分散プログラムの実行状況モニタがある。これは、並列分散プログラムのデバッグの一手法として知られており、いくつか研究されている[1]。しかし、これらは、(1)OSの改造や専用ハードウェアが必要で汎用性がない、(2)ユーザプログラムの変更を必要とするため、ユーザへの負担がある、(3)モニタできる通信手段を限定している(例えば、ストリーム通信を取り扱えない)などの問題点がある。

我々は、既に上記の問題点を解決する分散プロファイラ[2]を提案した。これは、異機種分散ネットワークシステム上で動く分散プログラムの実行の様子を、わかりやすく表示して、プログラマの動作理解を支援するシステムである。今回このプロファイラを試作、評価したので報告する。

2　分散プロファイラの概要

2.1　分散プロファイラの設計方針

本プロファイラでは、分散プログラムの実行状況の表示、統計的計測などが可能である。以下では、これらの手段によって、分散プログラムの実行状況を把握することを、プロファイルと呼ぶ。

分散プロファイラの設計方針を以下に示す。

- 実行時にプロファイルを行なう。
  クライアント-サーバモデルのように止まらないプログラムや、バグにより無限ループに陥っているプログラムもプロファイル可能にするために、実行時のプロファイルをサポートする。
- 汎用性の高いシステムにする。
  OSの改造、専用ハードウェアなどを必要としないものとする。また、異機種分散ネットワークに対応する。
- 多くの分散プログラムをサポートする。
  現在分散プログラムは、UNIX上のCでソケットライブラリを用いて記述するのが最も一般的である。本分散プロファイラは、このような環境を対象とする。また、ソケットは、ストリーム通信(データの区切りの単位がなく、単にデータをバイトの列として扱う通信)として使われることが多いので、これをサポートする。
- プログラムの速度向上を図る際の支援をする。
  受信待ち時間などの定量的、統計的計測を行なう。

2.2　分散プロファイラの構成

実行時プロファイルを行なうには、実行履歴であるログを実行時にユーザプログラムから採取し、1ヶ所に収集する必要がある。このため分散プロファイラは、図1に示すように構成される。

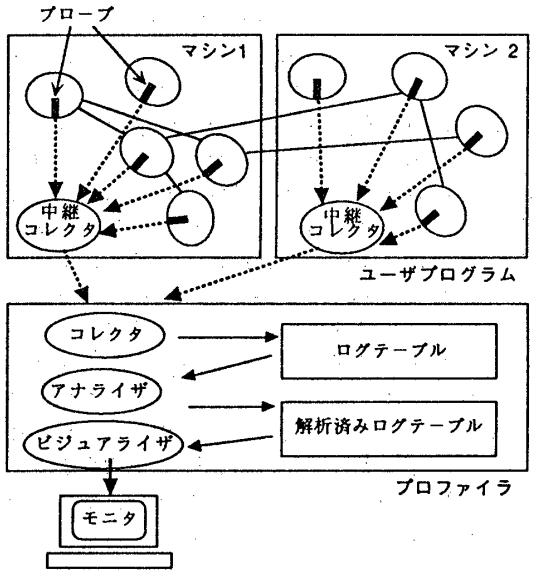

図1 分散プロファイラの構成

プローブは、ログを送るべき事象(以下イベントと呼ぶ)が発生すると、ログデータを作成し、中継コレクタへ送信する。本プロファイラではイベントを、プロセス生成消滅、プロセス間通信、受信待ちなどのCのライブラリ関数の呼びだしとする。プローブはインクルードファイルを用いて、これらのライブラリ関数をオーバライトすることで、ユーザプログラムに埋め込まれる。これにより、ログの採取と送信は、全てユーザ側で行なわれるため、OSの改造、専用ハードウェアなどは不要である。実際にプローブがログを送るのは、ログを採取すべき関数の実行が終了した時点である。これは、その関数の終了状態(Cライブラリのerrno)をログデータに含めて送信するためである。また、受信関数など待ち時間を測定したい関数については、関数の実行の前後にログの送信を行なう。なおログデータは、ホストマシン名、プロセスID、Lamportの論理時間[3]、イベントの種類及びストリームの送受信バイト数などから

A profiler for distributed programs
N.Shingaki, K.Yamazaki, Y.Amagai
NTT Software Laboratories

構成される。なお、ホストマシン名、プロセスID、Lamportの論理時間によりイベントが特定できるので、この3つ組をイベントIDと呼ぶ。

中継コレクタは、各マシンでプローブから受けとったログを一度蓄積し、一定数(現在は300個)に達した時点、あるいは一定時間(同5秒)ごとに、まとめてコレクタに送信する。中継コレクタでまとめて送信することにより、プローブから直接コレクタに送信するのに比べ、ネットワークを介する通信回数が減るため、ネットワークへの負荷を小さくする効果がある。

コレクタは、中継コレクタから収集したログをアナライザへ渡す。アナライザは、ログの送信順への並べ替えと、ストリーム通信の対応づけを行なう。まずログを、イベントIDを用いて因果関係のある順に並べ替える。これは、異なるプロセスからログを収集するため、イベントの発生順にログが届かない可能性があるためである。次に並べ替え済みデータを使って、送信イベントと受信イベントの対応づけを行なう。ここで問題となるのは、ストリーム通信の場合、一般に送信イベント(fprintfなど)と受信イベント(fscanfなど)が1対1に対応しないということである。これまでの分散プログラムのモニタでは、この扱いを避け、メッセージ通信のみをサポートしていた。本プロファイラでは、ストリームはデータの到着順が保証されるという特性を利用して、次のような方法によりストリーム通信をサポートしている。コレクタ側で、各ストリームごとに、ログデータを用いてそのストリームへの送信バイト数と、イベントIDを組にしてリストにする。『ストリームからの受信』のイベントのログがコレクタに到着すると、対応するストリームのリストを探す。次にそのリストの最初の要素の送信バイトから受信バイトを引く。これが正ならば、その送信イベントを対応させる。負ならば、次の要素を取り、同様に行なう。負の場合には、1つの受信に複数の送信が対応することになる。

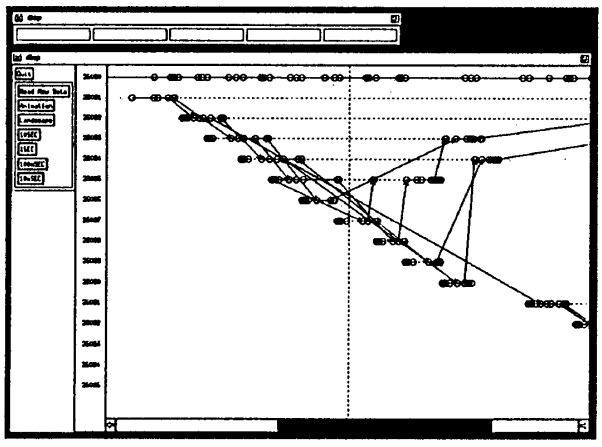

図2 分散プロファイラの表示例

ビジュアライザは、アナライザで解析したデータをXウィンドウ上に表示する。現在、時間-プロセスダイアグラムと、プロセス状態ダイアグラムが表示可能である。図2に前者の表示例を示す。なお現在、実行時表示は実装されていない。

3 プロトタイプの評価

プローブイフェクトとは、デバッガなどの影響で、プログラムの動作が変化することを言う。分散プログラムの場合、複数のプロセスを同時に実行するので、プローブイフェクトによる通信のタイミングの変化、実行時間への影響などが問題になる。これについて評価するため、ログ当たりの送信時間を次のような方法で測定した。まずユーザプログラムとして、2台のマシン間で、数バイトのデータを互いに転送し合うプログラムを作成した。このユーザプログラムにプローブを組み込み、2台のSUN4上で動作させ、別のSUN4にコレクタとアナライザを置き、ログを収集した。この状態で測定を行なった。次に比較実験を行なうために、プローブを改造して、ログをファイルへ書き出すようにし、同様に測定した。この測定結果を表に示す。

	CPU時間	経過時間
中継コレクタへ送信する場合	0.63	1.9
ファイルへセーブする場合	0.77	1.2

表 ログ送信1回あたりの時間(単位: ミリ秒)

CPU時間はあまり変わらないが、経過時間は本プロファイラでは、ファイルに書き出す方法の1.6倍となった。ログをファイルに書き出す方法は、プログラム終了後にログを1ヶ所に集めて解析するため、実行時プロファイルができない。本分散プロファイラは、前述のようにクライアント-サーバモデルのプログラムなどを実行時にプロファイルできるため、ファイルのセーブに比べて遅いが十分有用であると思われる。また現在、プローブ-中継コレクタ間は、TCPを用いているが、ログ送信時間を小さくすることにより、速い通信方法を検討する。

4 おわりに

本稿では、分散プログラムの実行状況をわかり易く表示することで、動作理解を支援するシステム、分散プロファイラの概要を述べ、プロトタイプの簡単な評価実験を行なった。今後は、現在実現していない統計情報の測定機能を実装する。また、大規模分散プログラムの動作理解を支援する方法についての研究を進める。

参考文献

1) C. E. McDowell, D. P. Helmbold: Debugging Concurrent Programs, *ACM Computing Surveys*, Vol.21, No.4, December 1989.

2) 新垣紀子, 山崎憲一, 天海良治: 分散プロファイラについて, 第24回情報科学若手の会シンポジウム, 1991年7月.

3) L. Lamport: Time, Clocks, and the Ordering of Events in a Distributed System, *Communications of the ACM*, Vol.21, No.7(July 1978). pp.558-565.