OS 資源ビューの仮想化を用いた分散システムテストベッド

西川賀樹[†] 大山恵弘[††] 米澤明憲[†]

1. はじめに

P2P、ネットゲーム等の分散システムの開発において、多くの実機や仮想マシンを用いる方法が広く利用されている。しかし、多数の実機を用いる方法では、実機の準備に大きな経済的コストと管理コストがかかる。仮想マシンを用いる方法にも、既存の仮想マシンの多くは大量の資源を消費するため、1 計算機上に実現できる実行環境が高々数個から数十個に限られるという問題がある。

そこで本研究では、1 計算機上に多数の仮想環境を生成する機能を提供し、分散システムの開発を支援するミドルウェアを提案する。ユーザは仮想分散環境のノードとネットワークに関する仕様を記述してミドルウェアに与える。ミドルウェアはその仕様を基にして仮想分散環境の構築と管理を行う。また仮想環境のネットワークトポロジの変化や通信状況の提示を行う GUI、仮想環境上で動作している多数のアプリケーションをまとめて管理・制御できるデバッグ支援機構を提供することにより開発の支援を行う。そのミドルウェアは、プロセスレベルの仮想化技術を用いているので 1 個の仮想ノード当たりの消費資源が極めて少ない。そのため軽量であり現在の平均的スペックの PC 上で 50〜100 個の仮想環境が現実的な速度で動作する。具体的には、システムコールを監視・制御することにより、ファイルシステムとネットワークへの操作を仮想化する。そのミドルウェアは、上記の仮想化によって行える開発作業を対象とし、より低い層の仮想化（ハードウェアやネットワークパケットの仮想化）を必要とするテストは対象としない。

2. 提案するミドルウェア

提案するミドルウェアは、ファイルとネットワークに関して実環境から隔離された仮想環境の中でプロセスを実行する機能を提供する。ファイルに関しては、Chroot と同様に、実環境のものとは異なるファイル名前空間を仮想環境の中のプロセスに対して見せる。ネットワークに関しては、各仮想環境に仮想的なアドレスを割り当てることができる。仮想的なアドレスは connect, accept, bind などの通信システムコールに影響を与える。GUI はアプリケーションプロセスをモニタすることによって得た通信やプロセスの状態等の情報を基にアニメーションを表示する。デバッグ支援機構については、ユーザからの入力または SIGSEGV シグナルの発生によって仮想環境上の任意のアプリケーションプロセスにデバッガをアタッチさせる。また複数のデバッガプロセス起動時にもユーザに一元的なインターフェースを提供する。

3. 記述言語

ユーザが与えたいネットワークトポロジ、検証に関するパラメータ、仮想環境に関する設定を表現するための記述言語を今回設定した。まずトポロジ、検証に関するパラメータにおける記述の仕様を以下に示す。

```
/*ノードの生成*/
create = nodeA[type]:...:nodeN[type];
/*仮想 IP アドレスの設定*/
nodeA[addr] = [virtual_IP][real_IP];
/*伝送路の設定*/
link = nodeA:nodeB[latency];
/*ノードの生成・削除*/
nodectl = nodeA[time][time];
/*仮想ファイル名前空間の設定*/
nodeA[file] = [virtual_path][real_path];
```

トポロジの指定は、まず create においてノード名を英数字でその後に大括弧内で type を指定する。type はアプリケーションやルータ、ファイアーウォール等動作するノードの役割を記述する。次に仮想ノードの仮想 IP アドレスとそれに対応する実 IP アドレスを記述する。そして link でノード間に伝送路を作成したい二つのノード名を示し、大括弧内に遅延時間を記

† 東京大学大学院情報理工学系研究科コンピュータ科学専攻
†† 電気通信大学電気通信学部情報工学科

述する。また実行中にノードの生成・削除を行う為に nodectl でノード名、次の大括弧内に削除までの時間を、次の大括弧内に生成までの時間を指定する。これにより実行中にトポロジを変化させた時の動作の検証を行う事ができる。最後に仮想ノードのファイル名前空間について仮想パスと実パスの対応を記述し定義する。

4. 実装の詳細

本研究は分散システム開発における第一段階のテストを容易にするシステムの提供を目的の一つとしている。よって、より簡単に導入・利用できるようにするために、ptrace システムコールを用いてユーザレベルで実装を行う。ptrace によりシステムコールをフックし、各システムコールの引数を書き換えることにより、ファイルシステムとネットワークに関して仮想化されたビューを構築する。ファイルシステムの仮想化は、各仮想ノードに作られるファイル群の実体を実ファイルシステム上に置き、open・stat 等のファイルパスを引数に持つシステムコールをフックし仮想パスを実パスに書き換えることにより実現する。ネットワークの仮想化は、各仮想ノードに仮想 IP アドレスを設定して、connect・accept・bind 等の通信システムコールの引数の仮想的な IP アドレスとポート番号を実際の IP アドレスとポート番号に書き換えることにより実現する。監視は一つのアプリケーションの実行による一つもしくは複数のプロセスに対して、一つの監視プロセスが監視する。また一つのマスタープロセスによって全ての監視プロセスの制御を行う。GUI については touchgraph[1] のライブラリを用いて実装を行った。被監視プロセスが socketcall ・write・read 等の通信関連システムコールを発行したり、起動・終了・停止等の状態の変化が起こった場合に監視プロセスがマスタープロセス経由で GUI プロセスと通信を行うことで仮想環境の状況を提示することができる。デバッグ支援機構はコマンドラインによるユーザの入力または SIGSEGV シグナルの発生によって起動する。実装には GDB を用いており起動するとユーザの指定した、もしくは SIGSEGV の発生した被監視プロセスに GDB をアタッチさせる。この際、複数のプロセスが ptrace を用いて一つのプロセスを監視することはできないので監視プロセスは一旦被監視プロセスからディタッチし GDB プロセスにアタッチする。GDB プロセスが終了すると再び監視プロセスは被監視プロセスにアタッチする。また全ての GDB プロセスを管理するプロセスを作成し、このプロセスによりユーザからの入力や GDB からの出力を制御する。これにより複数 GDB プロセスを端末から一元的に使用することができる。

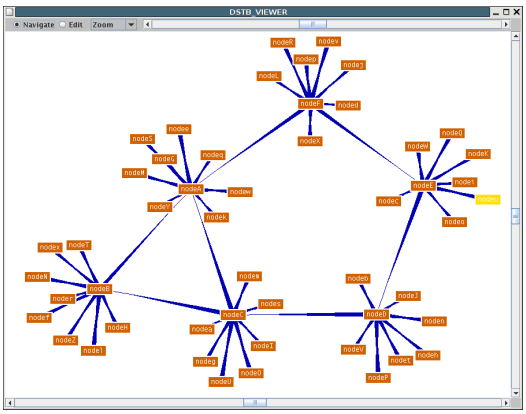

図 1 touchgraph ライブラリを用いて実装した GUI

5. 実　　験

今回 Gtk-Gnutella という P2P ソフトウェアを利用して本システムの実験を行った。Gtk-Gnutella は GUI により操作が可能なファイル共有ソフトである。実験には CPU が Core Duo 2.0GHz 、メモリが 2GB のマシンを用いた。1 台の PC 上に、Gtk-Gnutella を実行する仮想ノードを 60 個立ち上げた。60 個の Gtk-Gnutella は仮想環境上において、現実的な性能で問題なく動作しファイルの検索や交換等も正常に行うことができた。

6. おわりに

本研究ではプロセスレベル仮想化技術を用いて仮想環境を構築し、仮想環境上で動作している多数のアプリケーションを監視・制御することにより全てのアプリケーションに対して一元的に効率良くデバッグを行えるシステムを実装した。今後、NAT 等のように分散システムで問題となる可能性のあるネットワーク環境のエミュレーション機能や、分散システムのパラメータの設定を支援する為の機構を実装する予定である。それによりデバッグ支援に加えアプリケーションの動作の検証や設定が行える事を目指す。

参 考 文 献

1) : touchgraph. http://www.touchgraph.com/.