ハイパフォーマンス
コンピューティング 91－30
（２００２．８．２３）

大規模分散計算環境シミュレータの設計と実装

柴田俊介† 大野和彦† 中島浩†

本稿では，分散計算を行なう大規模分散計算機環境を複数の計算機上で仮想的に構築し，その中で分散計算プログラムのテストを行なうことができるシミュレータ Anastasia を提案する．

Anastasia は，クライアント／サーバシステム上で仮想世界のシミュレーションを行なう．仮想世界での計算性能やネットワーク遅延に合わせたイベントスケジューリングをおこなうため，仮想計算ノード間の通信は Anastasia が介在し，中継する．ユーザが必要な作業は，この中継のために分散計算システムの通信をおこなう関数を単純に置換することのみであり，非常に小さい労力でシミュレーションをおこなうことができる．

実際に分散計算プログラムを利用し，元のプログラムに対してわずかな修正で利用できる事，実際と比べて 16 ノードで 48%オーバヘッドでシミュレーションができることを示した．

Design and Implementation of large scale computational environment simulator

Syunsuke Shibata ,† Kazuhiko Ohno † and Hiroshi Nakashima†

We propose a simulator named Anastasia for large scale distributed computation system as a easily usable development and evaluation tool.

Ananstasia simulates a large scale distributed computation environment on a client/server system. It also captures all the communication messages between the processes to schedule events in the virtual environment of computation nodes and networks. Since almost all the part of programs in the target system are direcly executed by the processes, both required modification of the programs and simulation overhead are quite small.

We implemented Anastasia and evaluated its performance using the WDC distributed computation system. The result shows that a 16 node system is simulated with a small event scheduling overhead about 48%.

1. はじめに

近年，PC の性能とインターネットをはじめとするネットワーク技術の発達により，大規模な計算能力が得られる分散計算の手法が注目されている．グローバルコンピューティングとも呼ばれ，現在世界中で暗号解析[1] や π の計算[2]，地球外生命体探査[3] などで分散計算が広く利用されている．しかし，分散計算の性能は計算を管理するプログラムに大きく依存し，その評価や改良を実際の大規模かつ広域の分散環境でおこなうのは極めて困難である．

本稿では，このような大規模かつ広域な分散計算機環境を，小規模の計算機上で仮想的に構築し，分散計算プログラムのシミュレーションを可能にする大規模分散計算機環境シミュレータ Anastasia を提案する．

† 豊橋技術科学大学
Toyohashi University of Technology

Anastasia は，ユーザが自由に大規模な計算機環境（計算サーバやネットワーク等）を設計し，その中で分散計算プログラムをテストすることを目指す．プログラムのテストであるので，実際に使用するであろう分散計算プログラムをシミュレーションに利用するという方法で Anastasia を設計した．実装において，プログラムに対してシミュレーション用に修正が必要になるが，その変更は僅かである．

実際に，われわれの研究室で作成されたクライアント／サーバ型の分散計算システム WDC(Widely Distributed Computation)[4] を用いて，評価を行なった．その結果，シミュレーション用に通信部分の関数に対して完全な 1 対 1 の変更を行なうだけでシミュレーションに利用できる事，16 ノードからなるシステムについて，Anastasia を利用する事によるオーバヘッドが小さい事を確認した．

以後，2 章で Anastasia の特徴を述べ，続いて 3 章で設計，4 章で実装，5 章で評価という構成をとる．

2. シミュレータ Anastasia の概要

分散計算システムに代表されるような，現実には大規模かつ広域に分散されている計算機環境を，小規模の計算機上で仮想的に構築する 大規模分散計算機環境シミュレータ Anastasia を提案する．Anastasia 上では，ユーザが作成した分散計算プログラムを評価／テストを行なう事ができる．

Anastasia の特徴は次のとおりである．

2.1 シミュレーション環境の自由な設計

ユーザは想定するシミュレーション環境とその中の挙動（計算機の参加や離脱等）を自由に設計する事ができる．この情報を記したものをシナリオと呼び，ユーザが想定するシミュレーション環境の設計図である．この中にユーザの希望する配置の計算機やネットワークの構成を記述し，実行時の入力として与える事により様々な環境下での実験ができる．☆

2.2 ユーザプログラムを利用してのモデル化

Anastasia では，ユーザの作成した分散計算のプログラムを利用するというアプローチをとる．一般的にシミュレーションでは，シミュレーションさせるものに対してどのような振る舞いを起こすかといった**モデル**を記述しなければならない．しかし，Anastasia システムにおいては，ユーザは現実世界で使用するであろう分散計算プログラムを利用してシミュレーションをおこなうことができる．つまり，元のプログラムを利用する事で，ユーザがシミュレーション用モデルの設計作業を必要としないことを意味する．

分散計算のプログラムに対してシミュレーションの影響を加味するために，プログラムのソースコードに対して Anastasia 用の変更を必要とする．しかし，その労力は極めて小さく，かつ容易なものになっている．その理由は，変更点はホスト名やプログラム間の通信部分のみに絞っている☆☆からである．修正作業も該当するシステムコールや関数を完全に 1 対 1 に置換するだけであるので，手作業でもユーザの負担は非常に小さいものとなっている．将来的に，この修正を自動化することで，ユーザの負担を完全にゼロにする．

2.3 シミュレーション負荷の分散

元のプログラムをそのまま実行させるというアプローチであるため，シミュレーションするノード数が多くなればなるほど，プログラム実行のための計算能力を必要とする．一般に分散計算プログラムで解く問題は，非常に多くの計算パワーを必要とするため，シミュレーション用の小さな問題であったとしても現実と同じ負荷がかかる．そこで，負荷を分散させるために，Anastasia はクラスタ等の並列／分散計算機上での利用が可能である．すなわち，ユーザはシミュレーション規模に合わせて，シミュレーション用計算ノードを増やすだけで負荷の軽減を行なう事ができる．

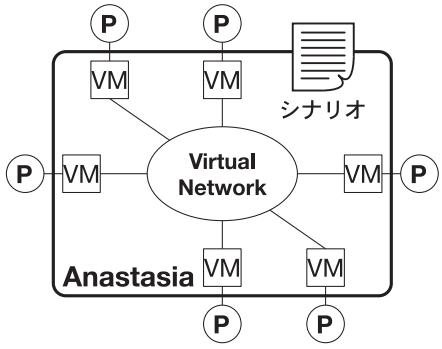

図 1 シミュレーションモデル
Fig. 1 Simulation Model

3. 設　　計

Anastasia システムの設計について説明する．3.1 で基本となるシミュレーションモデルを，3.2 では Anastasia システム自体の分散化について，3.3 では仮想情報を取り扱うためのプログラム側の対応について，3.4 で 3.1 から 3.3 までの設計を合わせた実装モデルの形について説明する．

3.1 シミュレーションモデル

Anastasia のシミュレーションモデルを**図 1** に示す．これは，元のプログラムをそのまま実行する部分と，シミュレーションとして制御される部分に分かれている．

計算プログラムをそのまま実行させるというアプローチから，計算プログラム（図 1 中の P）は，Anastasia の外でそれぞれ独立に，かつユーザのシミュレーションしたい数だけプロセスとして実行する．仮に，ユーザが 100 台の計算ノード環境を想定するなら，シミュレーション中では 100 個のプロセスが用意することになる．

Anastasia システムのシミュレーションモデル内を構成するのは，分散計算プログラムを起動して計算をおこなう**仮想計算機**（図 1 中の VM），インターネット等の計算機同士を接続する**仮想ネットワーク**（図 1 中の Virtual Network），シミュレーション世界の挙動に関してユーザが記述した**シナリオ**である．

Anastasia システムの外で実行される計算プログラ

☆ 現状では，仮想ホスト名と仮想 IP アドレスといった仮想ホスト情報を設計する事ができる．

☆☆ 計算部分や管理部分といった重要な部分には修正を加えない．

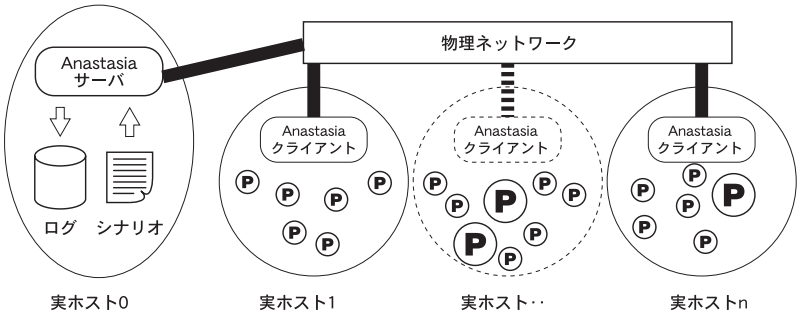

図 2 Anastasia の実装モデル
Fig. 2 Implementation model of Anastasia

ムには，それぞれ対応する仮想計算ノードが割り当てられる．プログラムの動作はこの VM のパラメータに合わせて仮想環境上での動作として調整する．仮想計算機同士は仮想ネットワークによって結ばれている．計算プログラム同士での通信は，一旦 Anastasia システムを中継して通信させる．中継させる事により，通信に対してシミュレーションの影響を加味させる．

よって，Anastasia でのシミュレーションは，(1) 実プログラムが動いた影響を考慮した VM，(2) 実プログラム同士の通信を中継した Virtual Network，(3) ユーザの設計した仮想計算機の出現時間・通信や仮想世界のネットワークの状態変化等を記述したシナリオ，の 3 つを利用して行なう．

3.2 Anastasia システムの分散化

Anastasia システムはプログラムをそのまま実行させるため，シミュレーションする計算ノード数が多くなればなるほど，物理計算資源（CPU の能力やメモリ）の不足を引き起こすことが予想される．そこで，物理的な資源不足の解消とシミュレーションの負荷分散のために，Anastasia は小規模な並列／分散計算機環境で動作するクライアント／サーバ型のアプリケーションとして設計する．クライアント／サーバ型としたのは，ノードの数に制限を持たせず，簡単に追加ができるようにするためである．

3.3 プログラムの Anastasia への対応

Anastasia システムの作成する仮想環境において，分散計算プログラムが対応しなければならない事項は以下であると考えられる．

(1) Anastasia を中継する通信
(2) 仮想ホスト名／ IP アドレス／ポート番号取得
(3) 仮想時間の取得

本来のプログラムのシステムコールのままでは，仮想情報は取得できない．そこで，Anastasia システムの仮想ホスト情報や仮想時間を取得する関数を用意し，元のシステムコールを Anastasia 用の関数にすり替えることで，仮想環境の情報を取得する．つまりシステムへ問い合わせをせずに，Anastasia に問い合わせを行なって，仮想情報を得るようにする．

3.4 Anastasia システムの実装モデル

3.1 から 3.3 までの設計をまとめた Anastasia システムの実装モデルは，**図 2** に示すようなクライアント／サーバ型のプログラムである．一つのサーバプログラムと複数のクライアントプログラムによって構成され，サーバと複数のクライアント同士は物理ネットワークを介して互いに通信をおこなう．それぞれ，以下の役割を持つ．

3.4.1 Anastasia サーバ

Anastasia サーバは以下の役割を持つ．

- サーバは全ての Anastasia クライアントと接続して，各 Anastasia クライアントが担当する部分仮想世界の情報を回収する
- シミュレーションを行なうための情報を集中的に管理し，シミュレーションを行なう

Anastasia サーバはユーザが記述したシナリオと呼ばれる仮想世界情報を入力し，仮想世界を構築する．Anastasia サーバでのシミュレーションの結果は，最終的にログとして出力される．

3.4.2 Anastasia クライアント

Anastasia クライアントは以下の役割を持つ．

- 分散計算プログラムを実際に動作させ，担当する全ての計算プログラムの CPU 使用時間やユーザ時間を監視する．さらにこの情報を Anastasia サーバに送る
- 分散計算プログラムに対して仮想ホスト名や仮想 IP アドレスといった仮想世界情報，さらに仮想時間を提供する
- 分散計算プログラム同士が通信を行なう際の仮想世界への出入口となる

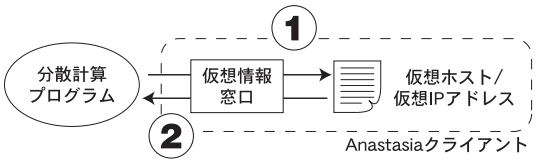

図 3 仮想情報取得
Fig. 3 Obtaining virtual information

各 Anastasia クライアントはシミュレーションを行なう仮想世界の一部分をそれぞれ担当し，実プログラムに対して Anastasia システムのインターフェースとなる．

4. 実　　装

設計で述べたシステムの実装を行なった．実装は，(1)Anastasia サーバ／クライアントの基本的実装, (2)仮想ホスト情報の取得，(3)Anastasia を経由する通信，について行なった．

4.1 Anastasia サーバ／クライアント

Anastasia サーバ／クライアントは ソケットインターフェースを用いた TCP ／ IP アプリケーション☆である．現状の実装は，Anastasia サーバ／クライアント間の通信と，以下で説明する機能である．

4.2 仮想ホスト情報の取得

図 2 で示した Anastasia システム上で，分散計算プログラムは仮想ホスト名や仮想 IP アドレスを利用可能にしつつ，実行されなければならない．

Anastasia クライアントは，**図 3** のように仮想情報取得用のソケットを準備する．分散計算プログラムは，ローカルホストに存在する Anastasia クライアントの仮想情報取得用のソケットに接続する事で，各情報を取り出す方法とした．☆☆

分散計算プログラム側での実現は，プログラム内のシステムコールや関数を Anastasia 用の関数に置き換えるという手法で行なった．入れ換えに必要なシステムコールや関数は，仮想ホスト名に対応する仮想 IP の取得やポート番号である．以下，入れ替える関数と実装について説明する．

4.2.1 仮想ホスト名の取得

ホスト名取得のシステムコール `gethostbyname()` を Anastasia 用の仮想ホスト名取得関数の `ANA_gethostbyname()` に置き換える．以下の動作により対応する仮想 IP アドレスを得ており，各番号は図 3 の番号に対応する．得られる仮想 IP アドレスは元の `gethostbyname()` の返り値と同じ構造のデータである．

(1) ローカルホスト内の Anastasia クライアントの仮想情報用の窓口に接続し，仮想ホスト名に対して問い合わせを行なう．
(2) Anastasia クライアントはテーブルを参照して対応する仮想 IP アドレスを返却する．

4.2.2 ポート番号

サーバプログラムがポート番号を設定する `bind()` は `ANA_bind()` に置き換える．流れは，図 3 の (1) でローカルホスト内の Anastasia クライアントの仮想情報用の窓口に接続し，仮想ポート番号を登録するだけである．

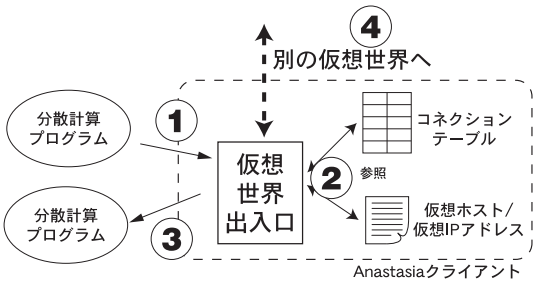

図 4 仮想ホスト間通信
Fig. 4 Communication between virtual hosts

4.3 仮想ホスト間通信

4.2 の仮想情報をもとに，Anastasia システムを通過させての通信についても実装を行なった．仮想世界の情報取得の時と同様に，Anastasia クライアントは，**図 4** のように 仮想世界出入口ソケットを用意する．分散計算プログラム同士の通信の際は，必ずこのソケットにアクセスする．分散計算プログラム側の対応も，4.2 同様に関数を入れ替える方法である．以下，入れ替える関数と実装について説明する．

4.3.1 仮想ホストへの接続

ほかのホストに接続するシステムコール `connect()` は`ANA_connect()` に置き換えて目的の仮想ホストへ接続を行なう．この時，`ANA_gethostbyname()` により取得した仮想 IP アドレスを利用する．図 4 での流れを以下に示す．番号は図 4 の番号に対応する．

(1) ローカルホスト上の Anastasia クライアントの仮想世界の入力ポートに接続を行なう．
(2) Anastasia クライアントは，送信された仮想 IP アドレスと仮想ポート番号を見て，仮想ホスト／IP アドレスのテーブルを参照し，目的の仮想ホストを探す．

☆ Anastasia に対応する分散計算プログラムも，ソケットインターフェースを用いた TCP ／ IP アプリケーションである．
☆☆ 現状は，仮想ホスト情報のみであるが，仮想時間も同じように取得する予定である．

(3) 目的の仮想ホストが同じ仮想世界内にいれば，Anastasia クライアントからサーバが待機しているポート番号に向かって接続をおこなう．
(4) もし，目的の仮想ホストが別の仮想世界に存在するのであれば，送信元の情報を付けて目的の Anastasia クライアントへ送る．

目的ホストまでの接続が完了すると接続の対応を示す**コネクション**がコネクションテーブルに登録される．

4.3.2 通信の入出力

通信における入出力関数として，C の標準入出力関数であるファイルディスクリプタを使用した `read()/write()` とファイルストリームポインタを使用した `fprintf()`, `fgets()` に対応する Anastasia システム用の関数にする．この関数内では，通信の際のバイト数をヘッダ情報として添付して送信している．受信する際には，情報を展開してもとの通信の情報を取り出す．読み込み／書き込みのデータの流れは図 4 のようになる．各番号は図中の番号に対応する．

(1) プログラム側から，Anastasia クライアントに対して書き込み／読み込みを行なう．
(2) Anastasia クライアントはデータが流れてきたソケットとコネクションテーブルを見て，データが流れる目的地を探す．
(3) 目的のホストが同じ仮想世界内なら，目的プログラムに対してデータを書き込む／読み込む
(4) 目的の仮想ホストがほかの仮想世界であれば，担当する Anastasia クライアントに送信する．

4.3.3 通信の終了

通信の終了は，コネクションのどちらか一端がクローズを行なった時に行なわれる．一方が切れたらコネクションテーブルからもう一方も切断し，コネクションテーブルから取り除く．

5. 評　　価

評価では，実際の分散計算プログラムを Anastasia 用プログラムに修正してシミュレーション環境で実行できること，通信が Anastasia システムを通過させていることによるオーバヘッドについて示す．

5.1 評 価 方 法

評価方法は，われわれの研究室で作成した分散計算システム WDC を利用する．方法は，最初に WDC のソースコードを Anastasia 用に修正・コンパイルしたものを用意する．このシミュレーション用プログラムと元のプログラムとで，同じ問題を与え， Anastasia システム上でも正常に動作する点と， Anastasia システムを通過するためのオーバヘッドについての評価をクライアント数を変えて行なった．

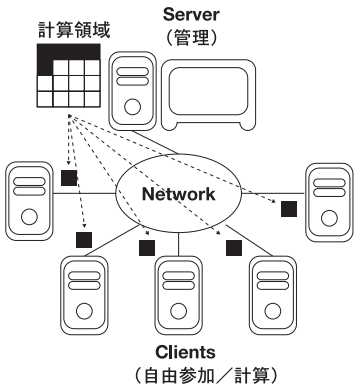

図 5 WDC システム
Fig. 5 WDC system

5.1.1 分散計算システム WDC

WDC は TCP ／ IP によるクライアント／サーバ型の分散計算システムであり，**図 5** のように，サーバにより分割された計算領域をクライアントが自由参加して取得し計算を行なうシステムである．流れは以下のとおりである．

(1) サーバが稼動する．
(2) クライアントが稼動．サーバに接続して参加し，計算領域を得る．
(3) クライアントは接続を切って，計算をおこなう．
(4) クライアントはサーバに接続して結果を返し，次の計算領域を得る．
(5) 3 に戻る．
(6) サーバは返却された結果を集計し，終了条件を満たせば終了する．

この評価で使用した問題は暗号解読である．ただし計算領域内に解は存在しない問題を用い，領域全てをクライアントが計算し尽くすまでを計測した．

5.2 プログラムの修正

元の WDC プログラムに対して， Anastasia 用に変更するには，設計で述べたようにホスト名や接続に関する部分と，通信の入出力を行なっている部分に対して修正を加える．WDC では TCP ／ IP のソケットインターフェースで通信している．WDC のソースコードの行数は約 2200 行であり，そのうち修正を加えたのは 52 行であった．52 行はホスト名取得，接続，通信の入出力関数のいずれかであり，これらは全て Anastasia 用の関数に完全に 1 対 1 で置換する事で Anastasia システムを介した通信と動作を行なうことができた．このことから，容易に分散計算プログラムを Anastasia システムに利用できると言える．

表 1 Anastasia の性能
Table 1 Performance of Anastasia

構成	オリジナル	Anastasia	オーバヘッド
1 Serv, 1 Cli	1301 sec	1335 sec	2.6%
1 Serv, 15 Cli	91 sec	135 sec	48%

5.3 オーバヘッド

Anastasia システムを中継することによるオーバヘッドを**図 1** に示す．1 サーバ 1 クライアントと 1 サーバ 15 クライアントでの環境☆で計測した．

6. 関 連 研 究

分散システム／グローバルコンピューティングをシミュレーションするという方法はいくつか提案されている．

グローバルコンピューティングのスケジューリングアルゴリズム評価として，Bricks[5),6)] がある．ネットワークトポロジ，計算サーバ，通信モデルをモジュール化し，パラメータを利用して実環境で計測される，計算ホストやネットワークの性能や外乱をパラメータとして設定し，グローバルコンピューティングのスケジューリング手法に対する評価を行なうモデルを提案している．

Grid 評価環境として，MicroGrid[7)] が挙げられる．クラスタ計算機上に Globus[8)] ベースの仮想 Grid システムを構築する Grid エミュレータである．われわれの Anastasia 同様に実プログラムをそのまま実行させるというアプローチである．しかし，時刻管理において MicroGird は実時間でのスケジューリングを行なうだけであり，プログラムが各仮想ホストの CPU をどれだけ利用するといったシミュレーションを行なっていない．

7. ま と め

本稿では，大規模な計算機環境をクラスタ上に仮想的に構築して分散計算プログラムのシミュレーションを可能とするシステム Anastasia を提案した．Anastasia は現実に用意できない大規模な計算機環境を小規模の計算機環境で実現するクライアント／サーバ型プログラムである．

シミュレーションには，現実に利用するプログラムに若干の変更を加えたものを利用する．変更によって，プログラムはユーザの記述した仮想化されたホスト名や IP アドレスを利用して動作させることができる．

☆ 各ノードは，Pentium II 450MHz, RAM 128M, 100M Ethernet で結合

実際にクライアント／サーバ型プログラムの分散計算システム WDC を用いた評価実験では，ほぼ完全な 1 対 1 の関数の置換によって容易に元のプログラムをシミュレーション用に修正・利用することができた．また，オーバヘッドは小さいものであることを示した．

今後は，Anastasia サーバが行なうシミュレーション部分の設計および実装を行なっていく．現実の計算性能やネットワークの状況はヘテロジニアスかつ動的であるため，分散計算プログラムの性能評価や改善のためのシミュレーションとして，動的なシミュレーション環境のモデリングを行なう必要がある．さらにこのシミュレーション環境での時間をプログラム側でも認識し，シミュレーションできるよう対応を行なっていく．

謝辞 本研究の一部は科学技術振興事業団・戦略的基礎研究事業「低電力化とモデリング技術によるメガスケールコンピューティング」による．

参 考 文 献

1) McNett, D.: distributed.et Project RC5. http://www.distributed.net/.
2) Percival, C.: PiHex A distributed effort to calculate Pi. http://www.cecm.sfu.ca/projects/pihex/.
3) P. Anderson, D.: SETI@home. http://setiathome.ssl.berkeley.edu/.
4) 田代友成: 分散計算システム WDC の設計と実装, 並列処理シンポジウム JSPP2001, pp. 271–278 (2001).
5) 中田 秀基 松岡 聡 長嶋 雲兵 竹房 あつ子: グローバルコンピューティングシミュレータの概要, 情報処理学会研究報告, Vol. 99, No. 21, pp. 31–36 (1999).
6) Aida, K., Tekefusa, A., Nakada, H., Matsuoka, S., Sekiguchi, S. and Nagashima, U.: Performance Evaluation Model for Scheduling in Global Computing Systems, *The International Journal of High Performance Computing Applications*, Vol. 14, No. 3, pp. 268–279 (2000).
7) Song, H. J., Liu, X., Jakobsen, D., Bhagwan, R., Zhang, X., Taura, K. and Chien, A. A.: The MicroGrid: a Scientific Tool for Modeling Computational Grids, *Supercomputing* (2000).
8) Globus: . http://www.globus.org/.
9) W. リチャード・スティーヴンス（篠田 陽一訳）: UNIX ネットワークプログラミング，トッパン (1992).