特集

4. 応用の新展開

－メタコンピューティングへの応用－

高木浩光
名古屋工業大学

松岡　聡
東京工業大学

メタコンピューティング

1997年10月，RSA Data SecurityのThe RSA Data Security Secret-Key Challenge[1]において，distributed.netのチーム[2]が広域分散処理によって56bit RC5を破ったことが話題となった．このプロジェクトでは，問題を分割しクライアントに配布するサーバを中心に据え，部分問題を解くクライアントプログラムをWindowsやMacOSをも含む各種OS用のバイナリ形式で用意し，ボランティアを募って彼らにこれをダウンロードして実行してもらうことで，多数のクライアントに結果を報告させるという方法をとった．56bit RC5は250日ほどで破られ，この間50万もの異なるIPアドレスの計算機が参加したという．

このような，ネットワーク上の多数の計算機を包括的に1つの仕事のために使用するというアイデアは，「メタコンピューティング」[3]として知られ，近年，ネットワーキング技術の進歩とインターネットの爆発的な拡大により，現実のものとして再び注目されてきている．distributed.netの例のようなパソコンの利用のほかにも，スーパーコンピュータを利用した本格的なプロジェクトも立ち上がってきており，たとえば「I-WAY」プロジェクト[4),5]は，北米の17の組織にあるスーパーコンピュータを互いに接続して，この上で，60の参加グループにより開発されるさまざまな分野の科学技術計算アプリケーションを実行させようというものである．

ここ数年でメタコンピューティングが特に注目されるようになったもう1つの理由に，Javaの登場がある．distributed.netのプロジェクトがあらゆるOS用のバイナリを用意したように，メタコンピューティングの成功には異機種混成の環境にどう対処するかが1つの鍵となる．アプリケーションをJavaで記述しておくことで，1つのコードの提供だけで，Java VM（Virtual Machine）の動作する任意の環境でそれを動作させることができる．この点に着目してJava AppletによってRC5を破ろうというプロジェクトがいくつか存在する[6),7]．一般に，Javaによる計算はC言語などにより書かれた同じプログラムに比べて実行速度が遅いと考えられ，このような目的にはそぐわないのではないかと考えられがちであるが，これらのプロジェクトでは次のようにその意義を主張している．第1にJavaの実行性能は高度なJust-In-Time（JIT）コンパイラの登場により将来はFORTRAN，C並みに高速化されると期待できる．第2に，Java Appletを動作させられるWWWブラウザさえあれば，処理系の面倒なインストール作業を必要とせずに，誰でも簡単に実行させられるため，より多くのボランティアを募ることができる．

Javaによる汎用メタコンピューティングシステム

第3はJavaのセキュリティ機能にある．ここで，セキュリティの意義を明確にするために，メタコンピューティングの運用形態を**表-1**のように分類してみた．PVM（Parallel Virtual Machine）[8]やMPI（Message Passing Interface）[9]を通信ライブラリとして使用した，NOW（Network Of Workstations）[11]上の仮想並列計算機システム，またCondor[10]などのシステムは，使用する各計算機に利用アカウントを必要とするものであるため，計算機資源提供者とアプリケーション利用者は同一であり，それぞれ特定

の者に限られる．これに対して，Ninf[12]およびNetSolve[13]などのシステムは，利用アカウントを持たない不特定の者に対して，リモート手続き呼び出しによる計算機資源の利用を提供するものであるが，実行されるアプリケーションはあらかじめ管理者によってインストールされているものに限られる．一方，冒頭に述べたdistributed.netのようなアプローチは，不特定の者の計算機資源を利用しているが，ボランティア達がこのプロジェクトを信頼（提供されているクライアントプログラムが自分の計算機に害を及ぼさないと信頼）できる必要があるため，アプリケーション作成者と利用者は特定の者に限られる．

表-1　メタコンピューティング運用形態の分類

代表的なシステム	計算機資源提供者	アプリケーション作成者	アプリケーション利用者
PVM, MPI, Condor	特定	特定	特定
Ninf, NetSolve	特定	特定	不特定
distributed.net	不特定	特定	特定
Javelin, Ninflet	不特定	不特定	不特定

これに対して，計算機資源提供者，アプリケーション作成者，利用者それぞれが不特定であることを許すシステムがJavaをベースとしていくつか提案されている．不特定多数の計算機資源での任意のプログラムの実行を現実的なものとするためには，プログラムが安全に（提供された計算機に害を与えないように）実行されることが保証される必要があるが，Javaがちょうどこの特性を備えている．Javaは，

- ファイルやネットワークへのアクセス，スレッドに対する操作などの可否を制御するjava.lang.SecurityManagerクラス．
- 不正なメモリ番地へのアクセスを起こさないことを保証する，排除されたポインタ演算や添字チェックされる配列といった言語仕様．
- 不正なコードが含まれていないかチェックするbytecode verifier.
- 異なるClassLoaderからロードされた同名のクラスを別クラスとして扱う仕組み．

といったセキュリティのための機構を持っており，これらに基づくJavaの安全性は，その1つの応用である「Applet」のフレームワークにより実証されている．このような特徴から，不特定多数による不特定多数のためのメタコンピューティングシステムを実現する手段として，Javaが注目されている．

Appletによるメタコンピューティングシステム

このような観点に基づいて提案されたJava Appletによるメタコンピューティングシステムに，Javelin[14]，Charlotte[15]，JET[16]，IceT[17]，POPCORN[18]，Bayanihan[19]などがある．Javelinは，クライアント，ブローカ，ホストの3者から構成され（図-1），クライアントがアプリケーション利用者，ホストが計算機資源提供者に相当する．ブローカは，クライアントからの要求を適切なホストに割り当てる役割を担っている．クライアントは目的のアプリケーションをJava Appletとして記述し，そのクラスファイルをどこかのWWWサーバに置いておく（図中（2））．一方ホストはJava VMを持つWWWブラウザである．ブローカは小さなAppletとサーバプログラムから構成されている．クライアントは，あるアプリケーションを実行させたいとき，そのURLをブローカに登録する（図中（3））．一方ホストは，その計算機資源を提供したいときに，ブローカのWWWページにアクセスする（図中（1））．ブローカは，クライアントから要求されたAppletのURLを，利用可能なホストに渡してそのページを開くよう指示する（図中（4））．ホストのWWWブラウザはこの指示を受けるとそのURLのページを開く．その結果，クライアントのAppletがホスト上でロードされて実行される（図中（5），（6））．

Javelin以外のシステムも基本的には同様の仕組みを持つが，それぞれ次のような特色を持っている．POPCORNでは，アプリケーション利用者をBuyer，計算機資源提供者をSellerと呼び，Javelinのブローカに相当する部分をMarketと呼ぶ．MarketはCPU時間を売買する市場であり，提供した計算機資源で実際に計算が行われるとpopcoinという単位の通貨を獲得でき，次の機会にこれを支払って自分のアプリケーションを他のSellerの計算機上で実行することができる．この仕組みによって，信頼関係のない不特定の者の間でこのシステムを利用しても，公平に互いの資源を共有することができる．この仕組みにより，夜間の空いている時間帯に計算機資源を地球の裏側の昼間の地域に貸しだし，昼間は逆に借りるといった運用が現実的なものとなる．

Charlotteは，プログラミングモデルとして，fork-join型の並列プログラミングに基づいたDSM（Distributed Shared Memory）を用意している．このDSMはJavaの純粋なクラスライブラリとして用意されており，特別なVMの拡張やコンパイラを必要としていないため，そのままWWWブラウザの上で動作可能となっている．具体的には各プリミティブ型に対応するDSM版のクラス（たとえばfloatに対応するDfloatクラス）が用意されており，そのインスタンスに対するget()とset(data)でデータの読み出しと書き込みをするものとし，このget()，set(data)メソ

ッドが分散共有メモリとしてのコヒーレンス管理を行っている．Charlotteではコンシステンシモデルとして CR&EW（Concurrent Read and Exclusive Write）を採用している．

Bayanihanは，ボランティアによる計算を指向しており，できるだけ多くのボランティアを集めるためか，計算の過程を見せることに重点が置かれている．Javelinにおけるブローカに相当する仕組みはなく，問題ごとに個別のサーバが用意され，ボランティア達がこの問題のページにアクセスすることで計算が行われる．システムはいわゆるMaster-Workerパターンに基づいており，中央のサーバプログラムがMasterとして，AppletがWorkerとして動作する．Workerには対応するGUIのフレームワークが用意されており，プログラマは，目的のアプリケーションの計算ルーチンと，その結果を表示するメソッドを実装する．

また国内においても，昨年開催された「Javaに関する技術・応用・表現大賞」の技術部門に，早稲田大学から同様にJava Appletによる分散処理を実現する作品が応募されている[20]．

専用のJavaフレームワークによるメタコンピューティング

筆者らもJavaによるメタコンピューティングシステムとしてNinflet[21]を提案している．NinfletはAppletベースではなく，専用のスタンドアロンアプリケーションとして実装している．Ninfletは独自のClassLoaderとSecurityManagerを実装しており，Appletベースでは実現不可能な細かなセキュリティポリシーを持っている．たとえば，AWTによるウィンドウの作成や，スレッド操作を禁止している．

Ninfletシステムは，Server，Dispatcher，Clientの3つから構成される．このうちServerとDispatcherはJavelinにおけるホスト，ブローカに相当する．計算資源提供者は，WWWブラウザを操作するのではなく，Serverをデーモンプロセスとして常時稼働させておく．Serverでは，DispatcherからのNinflet発行要求の受け入れの許可／拒否の条件を設定することができる．計算資源提供者は，たとえば，自分がその計算機を使用していない間だけNinfletの受け入れを許可するよう設定することができる．不許可に設定されている場合，DispatcherはそのServerにNinfletのインスタンス化を依頼しない．タイマー機能により，不在にする夜間，休日だけ許可するといったことや，スクリーンセーバが動作している間だけ許可するといったことも可能である．受け入れ設定が不許可に切り替えられたとき，すでに実行中のNinfletが存在した場合，そのNinfletに対しては退去命令が発行される．退去命令が発行されたとき，

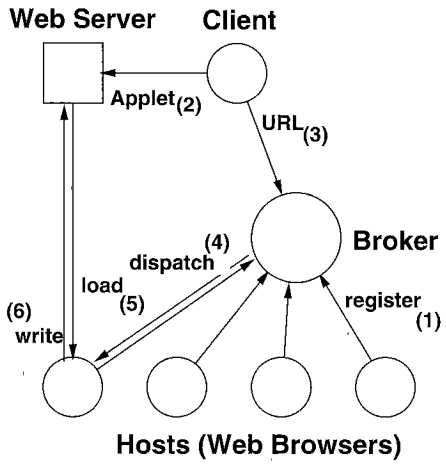

図-1 Java Appletを利用したメタコンピューティングシステムの基本構成

Ninfletは計算の途中結果を退避させることができる．退避は，Dispatcherにより指示されたところに対してなされる．退避されたNinfletは，Dispatcherにより再割当される別のServer上で再実行を開始することができる．

Appletベースのシステムには，計算資源提供者がインストール作業を必要としないという利点があったが，継続して資源提供をするには常時WWWブラウザを動作させて指定のページを開いておかなければならない．これに対しNinfletシステムは，いったんインストールがすめば，常時バックグランドプロセスとして動作し，計算機所有者を煩わせることがない．したがって，個人の計算機だけでなく，ワークステーションあるいはPCクラスタ上に仮想並列計算機を構築するためにも利用することができる．

Javaによる他の分散・並列計算システム

他にも，不特定の者のためのメタコンピューティングシステムではないが，Javaによる分散・並列処理をサポートするシステムがいくつか提案されている．JPVM[22]は，PVM風のメッセージパッシングインタフェースを実現するJavaのクラスライブラリであり，PVMとは互換性がない．JavaPVM[23]は，Javaで書かれたPVMへのインタフェースであり，PVMと互換性がある．ただしnative methodを使用して実装されている．JAVAR[24),25)]，JAVAB[26]はループなどに内在する並列性を，Javaのスレッドを用いた並列プログラムに変換する．JAVARはソースコードレベル変換するプリプロセッサで，JAVABはバイトコードからバイトコードへ変換する．

ARMI[27]はRMIにおいて非同期メソッド呼び出しを実現する．非同期メソッド呼び出しはHORB[28]でも提供されているが，HORBでは，1つのリモートリ

ファレンスオブジェクトに対して同時に1つしか非同期メソッド呼び出しをすることができない．これに対し，ARMIでは呼び出しごとにfutureオブジェクトを返すことにより，同時に複数の呼び出しが可能となっている．

Javaによるメタコンピューティングの将来

科学技術計算のためにJavaを利用することは果たして妥当だろうか？ 現状ではJava VMの実行速度はこのために十分なものとはいえない．にもかかわらず，Javaをそのような目的で利用する気運は高まっており，専門のワークショップもいくつか開催されている[29),30)]．

たしかに，Javaを用いる理由がセキュリティやプログラミングの容易さだけにあるのなら，別の専用言語を設計するのもよいだろう．しかし，普及度，今後の発展速度，プログラマの習熟度，関心度などを考えると，新たな言語を興すよりJavaを活用する方が現実的と考えられる．実行速度の問題は，HotSpotなどの先進的なJITコンパイラや，FORTRANなどにおける既存の最適化技術の適用によって改善されるだろう．ただし，Java固有の問題も多く存在する．たとえば，キャッシュを意識したブロッキングやプリロード系のソースレベルでの最適化に効果が出ない場合があることが我々の実験によって確認されている．また，UMA-SMP，NUMA-SMP，MPPなどの環境でいかにして最適化するかといった課題もある．

今後，Javaによるメタコンピューティングを実用的なものとするには，そのアーキテクチャデザインの研究を進めると同時に，数値計算に特化したJITコンパイラや，数値計算に向いたJava言語の拡張およびAPIの拡張の研究を進める必要がある．我々も，自己反映計算によりこれらの問題を解決するOpenJITシステム[31)]の提案・研究開発を始めている．

参考文献
1) The RSA Data Security Secret-Key Challenge, http://www.rsa.com/rsalabs/97challenge/
2) DISTRIBUTED.NET -- Fastest Computer on Earth, http://www.distributed.net/
3) Catlett, C. and Smarr, L.: Metacomputing, Communications of the ACM, 35, pp.44-52 (1992).
4) DeFanti, T., Foster, I., Papka, M., Stevens, R. and Kuhfuss, T.: Overview of the I-WAY: Wide Area Visual Supercomputing, International Journal of Supercomputer Applications, 10 (1996).
5) Stevens, R.: The I-WAY and the Future of Distributed Supercomputing, http://www.crpc.rice.edu/CRPC/newsletters/fal95/director.html
6) RC5 Java Breaking Effort, http://rc5.mit.edu/
7) RC5 and Java, http://www.hewgill.com/rc5/
8) Sunderam, V. S.: PVM: A Framework for Parallel Distributed Computing, Technical Report ORNL/TM-11375d, Department of Math and Computer Science, Emory University (1990).
9) MPI: A Message-Passing Interface Standard, The International Journal of Supercomputer Applications and High Performance Computing Systems (1994).
10) Lizkow, M., Livny, M. and Mutka, M. W.: Condor - A Hunter of Idle Workstations, Proceedings of the 8th International Conference of Distributed Computing Systems (1988). http://www.cs.wisc.edu/condor/
11) Anderson, T. E., Culler, D. E. and Patterson, D.: A Case for NOW (Network of Workstations), IEEE Micro, Vol.15 (1) (Feb. 1995).
12) 中田, 高木, 松岡, 長嶋, 佐藤, 関口: Ninfによる広域分散並列計算, 並列処理シンポジウム JSPP'97 論文集, pp.281-288 (1997). http://ninf.etl.go.jp/
13) Casanova, H. and Dongarra, J.: NetSolve: A Network Server for Solving Computational Science Problems, In Proceedings of Supercomputing'96 (1996). http://www.cs.utk.edu/netsolve/
14) Cappello, P., Christiansen, B., Ionescu, M. F., Neary, M. O., Schauser, K. E. and Wu, D.: Javelin: Internet-Based Parallel Computing Using Java, Concurrency: Practice and Experience (Nov. 1997).
15) Baratloo, A., Karaul, M., Kedem, Z. and Wyckoff, P.: Charlotte: Metacomputing on the Web, In Proc. of the 9th International Conference on Parallel and Distributed Computing Systems (Sep. 1996). http://www.cs.nyu.edu/milan/charlotte/
16) Silva, L. M., Pedroso, H. and Silva, J. G.: The Design of JET: A Java Library for Embarrassingly Parallel Applications, Proceedings WOTUG'20－Parallel Programming and Java Conference, IOS Press, pp.210-228 (1997).
17) Gray, P. A. and Sunderam, V. S.: The IceT Environment for Parallel and Distributed Computing, In Proc. of 1997 International Scientific Computing in Object-Oriented Parallel Environments Conference (1997).
18) Camiel, N., London, S., Nisan, N. and Regev, O.: The POPCORN Project: Distributed Computation over the Internet in Java, In 6th International World Wide Web Conference (Apr. 1997). http://www.cs.huji.ac.il/~popcorn/
19) Sarmenta, L. F. G., Hirano, S. and Ward, S. A.: Web-Based Volunteer Computing Using Java, In Proc. of the 2nd International Conference on Worldwide Computing and its Applications (Mar. 1998). http://www.cag.lcs.mit.edu/bayanihan/
20) 福森, 浜中, 菅原, 吉川, 中山: JAVA アプレットによる分散処理の実現, Javaに関する技術・応用・表現大賞'97 (Aug. 1997). http://www.java-fj.or.jp/event/grandprix/97/
21) Takagi, H., Matsuoka, S., Nakada, H., Sekiguchi, S., Satoh, M. and Nagashima, U.: Ninflet: a Migratable Parallel Objects Framework using Java, ACM 1998 Workshop on Java for High-Performance Network Computing (Feb. 1998).
22) Ferrari, A.: JPVM－The Java Parallel Virtual Machine, http://www.cs.virginia.edu/~ajf2j/jpvm.html
23) JavaPVM－The Java to PVM Interface, http://www.isye.gatech.edu/chmsr/JavaPVM/
24) Bik, A. J. C. and Gannon, D. B.: Automatically Exploiting Implicit Parallelism in Java, Concurrency, Practice and Experience, Vol.9 (6), pp.579-619 (1997).
25) JAVAR－A Prototype Restructuring Compiler for Java, http://www.extreme.indiana.edu/~ajcbik/JAVAR/index.html
26) JAVAB－A Prototype Bytecode Parallelization Tool, http://www.extreme.indiana.edu/~ajcbik/JAVAB/index.html
27) Raje, R., Williams, J. and Boyles, M.: An Asynchronous Remote Method Invocation (ARMI) Mechanism for Java, Concurrency: Practice and Experience, Vol.9 (11), pp.1207-1211 (1997). http://klingon.cs.iupui.edu/~rraje/html/armi.html
28) Hirano, S.: HORB: Distributed Execution of Java Programs, Worldwide Computing and Its Applications, Springer Lecture Notes in Computer Science 1274, pp.29-42 (1997). http://www.horb.org/
29) Java for Scientific Computing, http://www.npac.syr.edu/projects/javaforcse/
30) ACM Workshop on Java for High-Performance Network Computing, http://www.cs.ucsb.edu/conferences/java98/
31) 情報処理振興事業協会（IPA）高度情報化支援ソフトウェア育成事業, 自己反映計算に基づくJava言語用の開放型Just-In-Timeコンパイラ OpenJITの研究開発, http://matsulab.is.titech.ac.jp/OpenJIT/

（平成10年2月26日受付）