Linux/Alpha活用講座

清水 尚彦　nshimizu@keyaki.cc.u-tokai.ac.jp

第17回
AlphaでJavaを使う

4月号というと多くの雑誌ではフレッシュマン特集を組んだりしています。Linuxに新たにチャレンジする新人さんたちも大勢いるでしょうし、その中の何人かは普通の人と違うLinuxを触ってみたいと思っているかもしれません。PCのプロセッサがIntelやAMDによってどんどんスピードアップしているのに比べて他社のプロセッサはずいぶん地味に見えるかもしれませんが、浮動小数点の演算性能を考えた場合にはAlphaは高いパフォーマンスを入手しやすい価格で提供している点で特別な存在です。

しかもLinuxを載せることによって、個人で自由に研究の対象としていじることができるRISCマシンとして活躍できます。ということで、新しくLinuxに興味を持った人たちにはぜひAlphaも選択肢として検討していただきたいと思っています。と、ここで書いてもきっとこの記事はAlphaに興味のある人しか読まないので宣伝効果はあまりなさそうな気がしますが、一応言っておかないと気がすまないのでした。

研究室のAlphaのマシンの1台(実は研究室でまともに動くx86マシンは、学生が勝手に持ちこんだもの1台と計算センターから借りている1台の計2台だけです)にはWindows NT Terminal Server Editionを載せていて、WBTクライアントから利用しています。研究室の予算で移動用の端末としてWindows CEのマシン「Sharp Telios」を購入したので、RDP clientをマイクロソフトのWEBからダウンロードしてインストールしました。時々CEをリセットするはめになりますが、思ったより安定して動作しています。ちゃんと日本語も表示できるのでWebのブラウズにはTeliosを用いることにしました。市販のWBTは思ったより高価なのでTelios+RDP clientは良い選択ではないかと思っています。

AlphaLinuxのニュースから

・Mozilla M12がリリースされました。私のところでは動作の確認は取れていません。Red Hat Linux用などでバイナリが配布されていると思いますが、Debian GNU Linuxでもalienコマンドを使ってバイナリを変換して使えるはずというコメントがニュースグループでなされていました。

・Compaqのテストドライブのサイトがアップデートされました。特に注目すべきはMyrinet接続された8ノードのBeowulfシステムです。8台のXP1000にLinux/Alphaを搭載しています。また、Linuxに加えてFreeBSDも試用できるようになりました。

・Java2 JDKがリリースされました。これもMozillaと同様に私のところでは動作の確認が取れていません。

PPPの設定

職場において過ごす時間が長いと、家での環境はなかなか整っていかないのですが、年末年始などしばらく過ごす場合に電子メールなどの環境を整えておかないと不便です。講義資料をPowerPointで作ろうとした時期が一時期あって、そのときにNTでメールの環境を整えていて我慢して使っていたのですが、私のマシンでのNTの不安定さは仕事に使えるレベルではなく、やはりLinuxに文章作成の環境を変えました。それでもメールとPDFの作成はNTで行っていたのですが、やはり不便なのでLinuxでメールの環境を作成しました。

メールの読み書きは大学の環境でも使っているmnewsを使うとして、問題はプロバイダとのメールの受け渡しとオフライ

ンでメールを書く環境の構築です。POPで読みだすにはfetchmailを素直に使えばいいのですが、オフラインでメールを書いておいてダイアルアップの時に送信するには、sendmailをそれなりに設定する必要が有ります。本屋でめぼしい本を探しても、なかなか私の要求に合うようなダイアルアップ環境のsendmailの設定をきちんと書いてある本は見つかりませんでした。いろいろな本やWebページの断片や、当然ですがPPP-HOWTOやCFのなどのドキュメントから私なりに作ったCF用のdefファイルを参考までに掲げておきます(リスト1)。一応それなりに動いているように見えます。何かのご参考になればと思います。このdefファイルを作成するにあたって掲げた条件は次のようなものです。

①オフラインでメールの登録ができ、ダイアルアップでまとめて送ること
②プロバイダのメールサーバを使い、正式なFromアドレスがつくこと
③Message-IDに不正なものを出さないこと
④fetchmailでメールの受け取りが可能なこと

このdefから作成したsendmail.cfで問題なさそうに見えるのですが、心配だったのでマシンのドメイン名を削っておいたところsendmailの実行にすごく時間がかかるようになってしまいました。これはドメイン名を探してsendmailがタイムアウトしてリトライするためなので、おかしなMessage-IDが出ていかないことを確認してローカルなドメイン名を復活させました。要するに/etc/hostsに自宅LANの環境のFQDNもどきを登録しただけなのですが、sendmailの実行時間はこれだけですっきりとします。

sendmail自身はこの設定で問題がなくなったように見えますが、ダイアルアップの時にいちいち手動でsendmail -qと打つのは面倒です。それにfetchmailだって自動で実行したいと思います。そこで、ip-upというスクリプトを変更してダイアルアップが成功したときにはfetchmailとsendmail -qを自動で実行するようにしました。また、ppp-goのオプションに-mが指定されたときにはfetchmailとsendmailが終わったらppp-offを呼びだして接続を切るように、さらに、-nが指定されたらleafnodeのfetchnewsでネットニュースの読みこみを行うようにppp-goとip-upにそれぞれ条件を追加しています。これはppp-goに引数が与えられたときにpppdに渡すパラメータに追加したipparamを経由してip-upに引数を渡すことで実現しています。

リスト1　ダイヤルアップ用def

```
DEF_ID='@(#)so-net.def N.Shimizu 10Jan2000'
# for sendmail R8s
CF_TYPE=R8V8
VERSION_SEPARATOR=-
LOCAL_VERSION=ppp
OS_TYPE=linux
MY_DOMAIN='fa2.so-net.ne.jp'
# for the fetchmail local delivery
MY_ALIAS='$j'
# use official from address
FROM_ADDRESS='$m'
RECIPIENT_GENERIC=yes
REWRITE_GENERIC_FROM=yes
# relay to the provider's mail server
DIRECT_DELIVER_DOMAINS='none'
DEFAULT_RELAY='relay:[mail.fa2.so-net.ne.jp]'
# do not send at the first time
CON_EXP='True'
SMTP_MAILER_FLAG_ADD=e
RELAY_MAILER_FLAG_ADD=e
# delete message id from out-going mails
SMTP_MAILER_FLAG_SUB=M
RELAY_MAILER_FLAG_SUB=M
```

Javaを使おう

通常のプログラム言語としてCよりはFortranの方が直感的で分かりやすいと思うのですが、新しい世代のプログラム言語の概念を学ぶには若干(?)役不足なので、もう少し新しい言語を使えるようになっていると便利です。x86の世界では商用のコンパイラを含めさまざまな言語が使えるのですが、Alphaの世界では商用のコンパイラのサポートはそれほど活発ではありません。私はなにも数多くの言語を学ぶ必要はあまりなく、厳選したいつくかの言語の概念を学べば十分だと思っています。

その学ぶべき言語の中でも有力なものとしてJavaがあります。実はAlphaLinuxのWebページでJDK 1.2のポーティングがされたという情報を得て今回はその記事を書こうと思っていたのですが、私のところではうまく動かすことができずにいます。このJDKを動かすにはシェアードライブラリにTru64 UNIXのものが必要なような気がしています。あまり、こればかりに時間を取っていられないのでさっさとあきらめてフリーのJavaの実装であるkaffeを試してみました。kaffeについては

http://www.kaffe.org/

を参照ください(画面1)。Debianのパッケージにはkaffeは入っていませんでしたので上記URLからkaffe-1.0.5.tar.gzを取ってきて展開し、

```
# ./configure
# make;make install
```

で簡単にインストールができました。ただし、このようにインストールしたkaffeはJITが無効になってしまっています。JIT用の定義ファイルはソースツリーのなかにあるのですが、サポートがとまったまま今のkaffeには適合しなくなっています。簡単なプログラムのコンパイルと実行は十分できるのですが、JITがないシステムでは性能的に物足りませんね。ともあれ、きちんと動作するかどうか簡単なプログラムで試してみましょう。

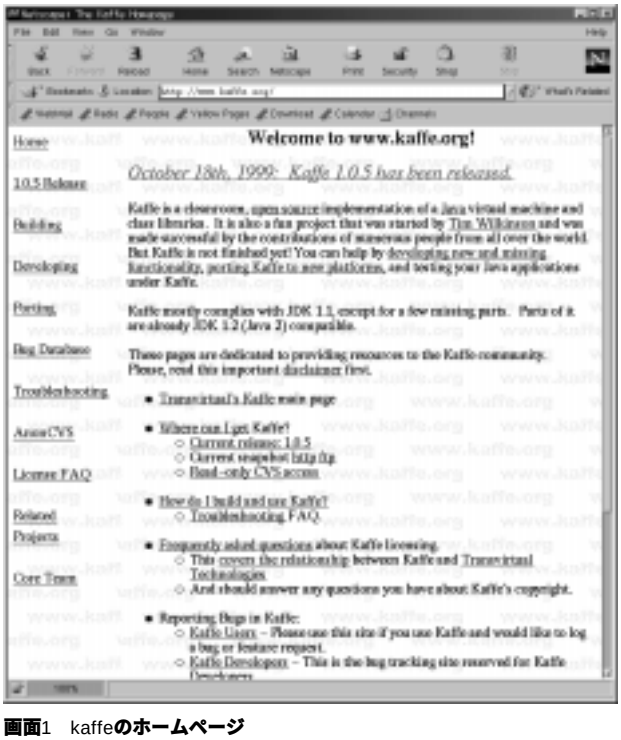
```
public class Hello {
 public static void main (String argv[]) {
  System.out.print("Hello World!\n");
 }
}
```

このリストをHello.javaという名前で作成します。

```
% javac Hello.java
% java Hello
```

と打ちこんで無事

```
Hello World!
```



画面1　kaffeのホームページ

が出力されたら、とりあえずは動作OKでしょう。JITがなくてもこの程度のプログラムなら全く気になりませんね。次に数値計算に使えるものか実力を試してみましょう。それにはベンチマークが手っ取り早いので、例によってLinpackベンチマークを実行してみましょう。

```
http://phase.etl.go.jp/netlib/benchmnark/linpackjava
```

からベンチマークのプログラムを取ってきます。このページにはアプレットとして走るJavaのコードがありますが、今はコンソールで走らせるためにzipファイルを取ってきましょう。tar.gzのファイルもあるように書かれていますが、実はzipの方しかありませんから注意してください。さて、ここからLinpackJava.zipを取ってきたらこれをunzipコマンドなどで展開してソースコードであるLinpack.javaを取りだします。例によって

```
% javac Linpack.java
% java Linpack
```

で実行させてみましょう。いくつかのマシンで実行した結果を示します(表1)。この表の中でSS10はSUNのワークステーションですが、JITを使って高速化を計っています。それにしてはあまり速くはないのは古いマシンですからご愛嬌ということにしてください。悪いことにJITを使っていた場合の経過時間はさらに伸びています。

先ほどのURLにはベンチマーク結果がいろいろと発表されていますが、上位にある2000MFLOPSを超える値はおそらくタイマーの不正確さからくる値だと思われるので(こんな数値がでるくらいなら本来のLinpackベンチマークにおいてトップが取れます)無視するとしても、少なくとも数十MFLOPSは出てくれないと物足りません。また、このベンチマークではせっかくの21264を搭載したXP1000の性能が振るいません。

これはJITの無いインタープリタだけのJVMでは分岐予測の困難なコードになってしまい、アウトオブオーダーがきちんと働かないものと想像できます。いくらなんでも数値計算のアプリケーションとしてこの性能では満足できません。特にXP1000はFortranでは200MFLOPSを超える性能を実測してい

表1　JIT無しの実行結果

マシン	経過時間	実行時間	性能(MFLOPS)
XP1000	1.38	0.98	0.70
VT 5/600	1.04	0.60	1.14
EB164/300	2.0	1.21	0.57
SS10-30	3.64	0.73	0.94

るだけに差が歴然としすぎています。そこで、Alpha用の高速なJITとして定評のあるcacaoをインストールして試してみました。このJVMはまだサポートされるクラスに制限があるなど、互換性の面で不安があるので完全に頼りきるには心もとないのですが、Webページに示された高速性はとても魅力です。cacaoのWebページは

```
http://www.complang.tuwien.ac.at/java/cacao/
```

ですが、このWebページのインストールドキュメントはあまり親切ではないですね。私がやったのは、まずインストールのページからcacao.0.3.linux5_1.gzとclasses.zip.gzの2つのファイルをダウンロードします。そしてcacao.0.3.linux5_1.gzをgunzipで解凍した後/usr/local/bin/cacaoにリネームして

```
# chmod +x /usr/local/bin/cacao
```

と実行形式に直しました。classes.zip.gzは/tmpに置いて、まずgunzipで解凍した後展開します。

```
# gunzip /tmp/classes.zip.gz
# mkdir /usr/local/lib/java{,/classes}
# cd /usr/local/lib/java/classes;unzip -x /tmp/
classes.zip
```

cacaoの実行はjavaコマンドの代わりにcacaoとすれば良いので、例によって簡単な例題から実行してみましょう。

```
% cacao Hello
```

いかがですか？　このプログラムはきちんと動作すると思います。ところが、Linpackはエラーになって動作しません。配列のバウンダリチェックに引っかかっているようなメッセージが表示されています。そこで、-oldオプションをつけて異なるJITを使ってみるときちんと動作することが分かりました。新しいJITの問題点なのか、インストールした環境の問題なのかこれから切り分ける必要があります。

早速、cacaoのベンチマークをLinpackで行ってみましょう(表2)。最新のJITはもっと速くなっているはずですが、動かないものは仕方がないですからね。kaffeに比べて圧倒的に速くなっているのが分かると思います。このくらいの性能ならば普段使っても違和感がないのではないでしょうか？　ここでは21264のXP1000がもっとも高い性能を出しています。ちゃんとコンパイルされたコードであればこれが当然ですね。

これらの値がFortranのLinpackベンチマークとどういう関係にあるか興味のある方もいると思います。1月号で示したように、CompaqのFortranではもう少し大きな200元の問題でVT5/600, XP1000はそれぞれ191MFLOPS, 263MFLOPSをマークしています。ちなみにg77ではそれぞれ50MFLOPS、234MFLOPSとなっているのでEV5のユーザーでg77を使っているならば2倍以内の性能差ということになります。CompaqのFortranを使うならば最大6倍ほども違うのでJAVAのメリットを見つけるのは難しいところですね。ただ、この結果はcacaoの古いJITを使っているということに注意してください。cacaoではソースの配布はまだされていないので新しいJITの問題点は追求できませんが、新しいものを使えればずっと高速になっているはずです。cacaoのプロジェクトではJITのオブジェクトの最適化も考えているようです。

表2　JITありの実行結果

マシン	経過時間	実行時間	性能(MFLOPS)
XP1000	0.05	0.01	57.22
VT 5/600	0.07	0.02	32.70
EB164/300	0.1	0.04	16.35

LinPackベンチマーク 2000年1月レポートから

ftp://phase.etl.go.jp/pub/netlib/benchmark/performance.psに最新のベンチマーク性能の報告が出ています。執筆時点では2000年1月が最新になっています。このレポートに掲載されている情報の中からベクトル機を除いたマシンの上位から10機種ほど拾ってリストにしてみます。

1. Compaq 8400 6/575 460MFLOPS (kf77, inline daxpy)
2. Compaq Alpha Server DS20/500MHz 440MFLOPS (kf77, inline daxpy)
3. Compaq 8200 6/575 440MFLOPS (kf77, inline daxpy)
4. HP N4000 440MHz 375MFLOPS (f77, inline daxpy)
5. HP V2500 440MHz 375MFLOPS (f77, inline daxpy)
6. IBM RS6000/397 160MHz 315MFLOPS
7. Compaq XP1000 500MHz 335MFLOPS (kf77)*1
8. DEC 8400 5/625 612MHz 287MFLOPS
9. Compaq Alpha Server DS20/500MHz 270MFLOPS
10. DEC 8200 5/625 612MHz 268MFLOPS

となっています。KAP Fortranのインライン展開を使ったものはさすがに速く、同じDS20でも1.6倍もの性能を叩き出します。ちなみにこのレポートに載っているx86の最速情報は

*1　この順位はおかしいですね。

PentiumII Xeon 450MHzの98MFLOPSです。JavaのJITで本当に100MFLOPS程度でているのかどうかこの結果を見る限り疑問に思っているのは私だけでしょうか？　正確なタイマを持つシステムで計測した性能が知りたいところですね。

A Linux/Alphaのディストリビューション

私はDebianをずっと使っていますが、さまざまなディストリビューションが増えてきています。中でもRed Hat Linuxの開発版をベースに日本でまとめられている、Kondara MNU/Linuxというディストリビューションをプリインストールするベンダも現れて注目しています。その他にはRed Hat 6.1やSuSE 6.3もAlpha版のリリースやアップデートがなされています。私のメインマシンで使っているDebianも2.1r4がリリースされています。

このようなパッケージの乱立は、それぞれのパッケージのバイナリの互換性さえ確保できていれば気にするほどのことではないのですが、互換性が取れないとバイナリで配布されているパッケージの利用が不便になります。正しくパッケージを動作させるためには、ライブラリとカーネルのバージョンの要求に注意してパッケージの利用をする必要があります。

画面2　Compaqの登録をするところ

A おわりに

KondaraにはAlpha用のバイナリもあるので、研究室のEB164にインストールしようと思って作業を進めていましたが、残念ながら現時点ではインストールに成功していません。インストーラが途中で固まってしまうのですが、診断用のコンソールがないので何が起きているのかも把握できないのが残念です。ネットワークインストールでなければうまくいくのかもしれません。次回までにインストールに成功したら簡単なレポートをしたいと思います。

さて、Compaqからまた Alpha/Linuxに大きなプレゼントが届きました。それは長らく待たれていたNetscapeです。NetscapeのTrue64版はもともとあって、Linuxで使うためにはCompaqのシェアードライブラリが必要だったのですが、Compaqでは必要なライブラリをセットにしてRPMパッケージにしてダウンロードできるようにしました。Debianのユーザーはダウンロード後に少し手作業が入りますが、ともかくきちんと動くブラウザが手に入ったということは素晴らしいことです。`http://www.compaq.com/partners/netscape/downloads/register_nav4_Linux.html`(画面2)からユーザー登録をしてダウンロードしてください。