*SXF90*のビジュアル系ディバッガとしての *Psuite*

武知 英夫（阿南工業高等専門学校 機械工学科）
（サイバーメディアセンター利用指導員）
（a63059@center.osaka-u.ac.jp）

1 はじめに

かっての大型汎用機で365日殆んど毎日数値計算をやっていた経験がある。連日まともな解が出力されなかったのは勿論ではあるが、計算アルゴリズムのバグ取りに明け暮れた経験から、*Psuite*に関しては*SX-5*で運用が開始された時から興味があった。工学屋にとっては解をいかに早く出すかが全てで、高速化は二の次であることは申し上げるまでもない。この点で、*Psuite*のプログラミングの高速化支援よりも寧ろディバックにおけるビジュアル系機能を駆使した可視化機能の方を個人的にはより高く評価している。ここでは、*SX-5*の*f90*のプログラミング[1][2]から説明を進めて、*Psuite*の基本操作に至る個人的な経験を紹介することで、*f90*のディバックで現在何らかの壁を感じておられるユーザの一助になればと思う次第である。

2 サブルーティンアーカイブの作成と運用

実行毎にサブルーティン全てを再コンパイルするジョブをよく見かけるが、プログラムの編集効率や実行時間において有効な方法であるとは言い難い。元もと*SX-5*の*Psuite*はビジュアル系の設計であるために、システムディフォールトでは全てのサブルーティンを同一ディレクトリーに置いて一括コンパイルする方法をするように設定されているが、敢えてサブルーティンアーカイブ[3]を使用する実行方法を以下に紹介する。

```
sx54 /fs/user6/x60602/sx/sx_5/bolt<151>%f90 -c -o plsmb f90/plsmb.f *1
f90/plsmb.f:

f90: vec(1): plsmb.f, line 20: ループ全体をベクトル化する。
f90: vec(1): plsmb.f, line 20: ループ全体をベクトル化する。
f90: plsmb.f, plansm: There are 11 diagnoses.
sx54 /fs/user6/x60602/sx/sx_5/bolt<152>%ar r ~/libsublib.a ./plsmb  *2
sx54 /fs/user6/x60602/sx/sx_5/bolt<153>%ar t ~/libsublib.a          *3
asmatb
matinb
plsmb
prstb
setbak
lmulti
sx54 /fs/user6/x60602/sx/sx_5/bolt<154>%ar d ~/libsublib.a plsmb    *4
sx54 /fs/user6/x60602/sx/sx_5/bolt<155>%ar t ~/libsublib.a          *5
asmatb
matinb
prstb
setbak
lmulti
sx54 /fs/user6/x60602/sx/sx_5/bolt<156>%ar r ~/libsublib.a ./plsmb  *6
sx54 /fs/user6/x60602/sx/sx_5/bolt<157>%ar t ~/libsublib.a          *7
asmatb
matinb
prstb
setbak
lmulti
plsmb
```

図 *1* サブルーティンアーカイブへの挿入と削除

プログラム *plsmb.f (図 1 の*1)* をコンパイルオプション*-c* で *plsmb* へオブジェクト化する。このオブジェクトモジュール *plsmb (図 1 の*2)* を *SX-5* のアーカイブコマンド *ar*[1] の *ar r* オプションでライブラリ *libsublib.a* へ書き込むことが出来る。確認のために *ar t* 命令 *(図 1 の*3)* でアーカイブの構成メンバを参照する。アーカイブの編集の一例として、*(図 1 の*4)* で構成要素 *plsmb* を消去してみた。この消去を確認するために、*ar t (図 1 の*5)* を実行している。

後に述べる *Psuite* のディバッグモードで実行する場合は、上述の全てのサブルーティンを *f90 -c -C debug -optb -o plsm f90/plsmb.f* (ディバッグオプション**-C debug**) でコンパイルする必要がある。

3 サブルーティンアーカイブのリンクと実行

次に、ディレクトリー *(図 2 の*1)* にあるメインプログラム *elast.f* のコンパイル *(図 2 の*2)* を実行する際に、サブルーティンアーカイブ *libsublib.a* をディレクトリー */usr1/x60602* からリンクすることをオプションで指定する。

```
sx54 /fs/user6/x60602/sx/sx_5/bolt<178>%cd /usr1/x60602/sx/sx_5/bolt/fort        *1
sx54 /usr1/x60602/sx/sx_5/bolt/fort<179>%dir .
合計 4350
drwxr-xr-x    3 x60602   prosou         4608 Jul 14 18:49 ./
drwxr-xr-x   10 x60602   prosou          512 Aug  5 20:22 ../
drwxr-xr-x    2 x60602   prosou         4608 Jul 14 18:49 OPT/
-rw-------    1 x60602   prosou        15234 Mar  8  1998 elast.f
-rwx------    1 x60602   prosou         3488 Mar  8  1998 input.dat*
sx54 /usr1/x60602/sx/sx_5/bolt/fort<180>%f90 elast.f -L/usr1/x60602 -lsublib     *2
elast.f:

f90: vec(3): elast.f, line 21: ベクトル化できないループである。
f90: vec(3): elast.f, line 26: ベクトル化できないループである。
f90: vec(1): elast.f, line 36: ループ全体をベクトル化する。
f90: opt(1800): elast.f, line 125: 行列積ループをライブラリ呼び出しに置換した。
f90: vec(3): elast.f, line 181: ベクトル化できないループである。
f90: elast.f, _MAIN: There are 23 diagnoses.
f90: vec(2): elast.f, line 190: ループの一部をベクトル化する。
f90: elast.f, setbak: There is 1 diagnosis.
sx54 /usr1/x60602/sx/sx_5/bolt/fort<181>%dir .
合計 4290
drwxr-xr-x    3 x60602   prosou         4608 Aug  5 20:35 ./
drwxr-xr-x   10 x60602   prosou          512 Aug  5 20:22 ../
-rw-r--r--    1 x60602   prosou         1896 Jul 14 18:51 .pproj
drwxr-xr-x    2 x60602   prosou         4608 Jul 14 18:49 OPT/
-rwxr-xr-x    1 x60602   prosou      1057747 Aug  5 20:35 a.out*
-rw-r--r--    1 x60602   prosou         1683 Jul 14 18:49 a.out.mak
-rw-r--r--    1 x60602   prosou         1683 Jul 14 18:34 a.out.mak.old
-rw-r--r--    1 x60602   prosou         1703 Jul 14 17:12 a.out.mon.out
-rw-------    1 x60602   prosou        15234 Mar  8  1998 elast.f
-rwx------    1 x60602   prosou         3488 Mar  8  1998 input.dat*
sx54 /usr1/x60602/sx/sx_5/bolt/fort<183>%a.out < input.dat                       *3
0********** plane stress ********
 NODT  48  ,NELT  69  ,KOX  8  ,KOY  7  ,nf  6
 READ(*,11) (KAKOM(I  1  ,J  4  )  1  ,J=1,3),T(I  1  )  1.000000
 READ(*,11) (KAKOM(I  2  ,J  4  )  2  ,J=1,3),T(I  2  )  1.000000
 READ(*,11) (KAKOM(I  3  ,J  4  )  2  ,J=1,3),T(I  3  )  1.000000
 READ(*,11) (KAKOM(I  4  ,J  4  )  3  ,J=1,3),T(I  4  )  1.000000
0   66       -0.1719E-01   0.1171E+01  -0.3751E-01        0.1172E+01  -0.1837E-01      91.8
0   67        0.1089E+00   0.1044E+01  -0.6089E-01        0.1048E+01   0.1050E+00      93.7
0   68        0.1729E-01   0.1024E+01  -0.2421E-01        0.1024E+01   0.1671E-01      91.4
0   69        0.1984E-01   0.1047E+01  -0.3115E-01        0.1048E+01   0.1890E-01      91.7
sx54 /usr1/x60602/sx/sx_5/bolt/fort<184>%
```

図 *2* サブルーティンアーカイブをリンクする f90 プログラムの実行

[1]sx-5 の man ar で参照

ここでは、コンパイルのオプションで実行モジュール名を指定をしないディフォールト *a.out* (図 *2* の **3)* を実行し、データファイル *input.dat* を入力とする計算結果を得ている。

4 *Psuite* の起動と実行

Psuite の実行では、以上のようにプログラムの実行が一応完結することが必要である。*Psuite* では、ディバックモードの *f90* でコンパイルし、リンクされた実行モジュール *a.out* を会話的に実行させながらチューニングが処理されるために、*sx-5* での実行が起動と同時に停止するモジュールでは *Psuite* の機能を起動する以前にアボートしてしまう事態になる。

4.1 *Psuite* の起動の前に

Psuite を起動する前に、クライアント側 (ここでは *vis01e0*) のユーザファイルを以下の要領で更新する必用がある。

- *.rhosts* に *vis01e0* を記入する。
- *psuite* のパスを指定する。*%setenv PATH ${PATH}:/usr/psuite*
- 手元の端末画面を *vis01e0* に貼り付ける。*%setenv DISPLAY "unix:0.0"*
- *Psuite* の初期化ファイルをホームディレクトリへコピーする。*%cp /usr/psuite/.psuite .*
- *Psuite* の初期化ファイル *.psuite* の項目に以下の値を設定する。
  *PSUITE*remotehost:sx5.center.osaka-u.ac.jp*　　*SX* のホスト名
  *PSUITE*remotedir:/fs/user6/x60602*　　*SX* の作業ディレクトリ
  *PSUITE*username:x60602*　　*SX* のアカウント名
- *Psuite* の起動はホームディレクトリで *psuite &*

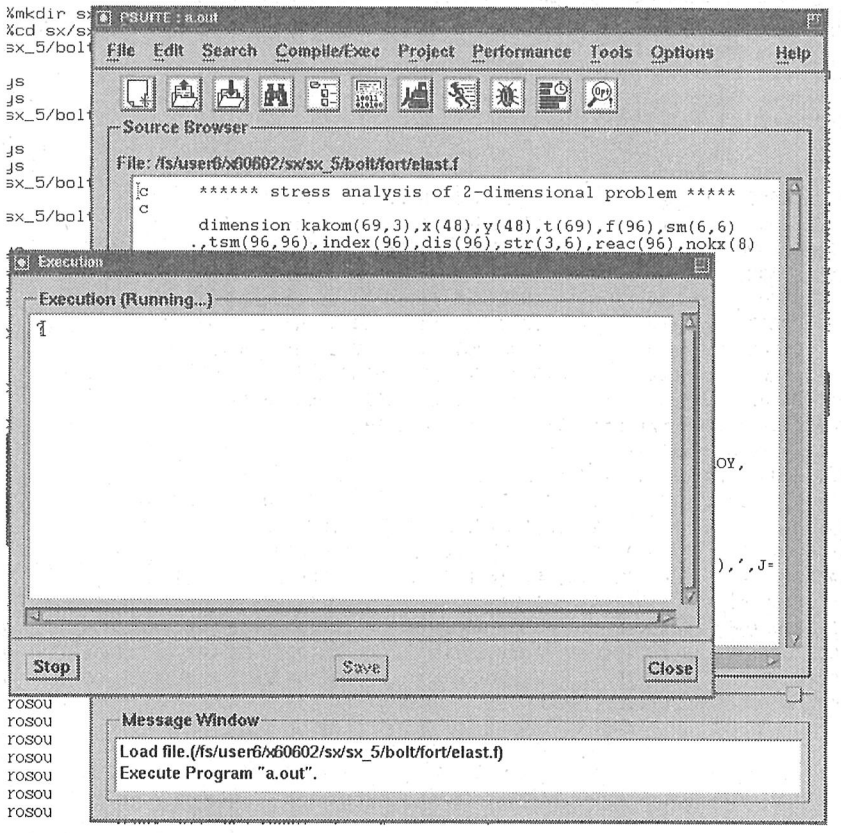

図 *3 Psuite* の起動と *Fortran 90* の実行ウィンドウ

4.2 *Psuite* の実行手順

Psuite の起動に続いて、以下の順でサブメニューからコンパイル、メイク&ビルド、ランを実行する。図3は *Psuite* の *RUN* 常態の画面を表示している。

- メニューバー *Open File* からソースファイルをオープンする。
- メニューバー *Compile/Exec* から *Set Compile Options* からコンパイル時のオプションを指定する。
- メニューバー *Build* から、*Makefile* の自動生成、コンパイルおよびリンクを実行する。
- メニューバー *Compile/Exec* の *Set Run Options* を選択し、実行オブジェクト名、入力ファイル名や出力ファイル名を指定する。
- メニューバー *Run* を押してプログラムを実行させる。

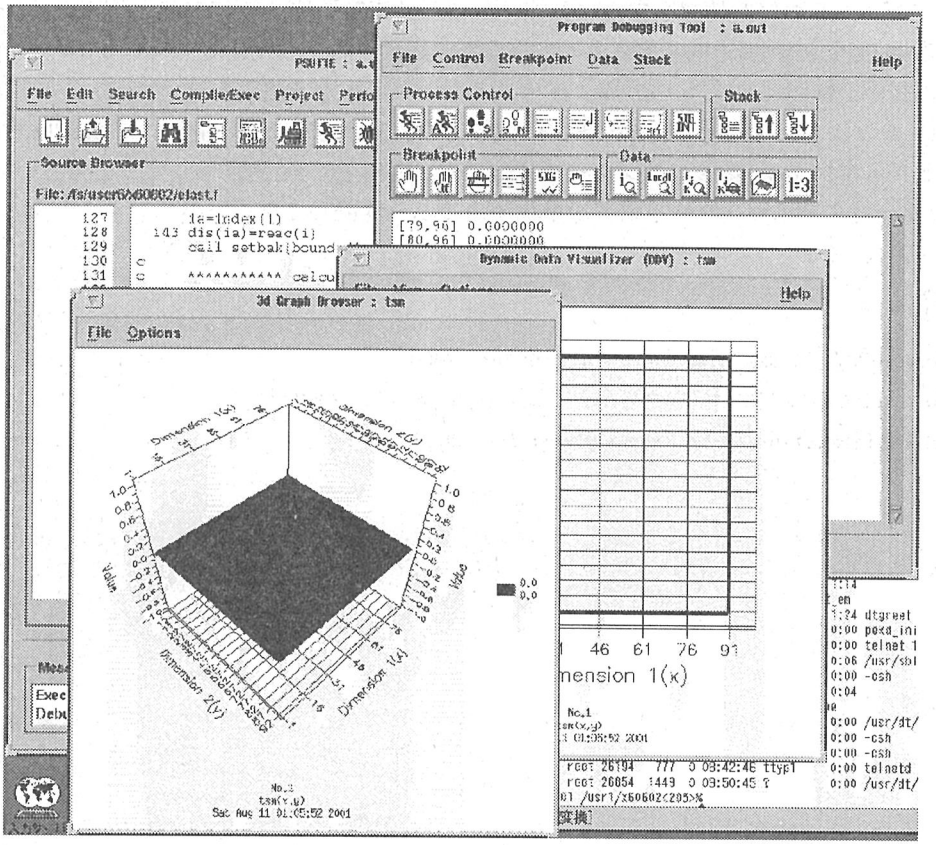

図 *4* ディバッグモードの *Psuite* による配列の *3-D* 表示

5 *Psuite* のディバッグモード実行

上の図4はディバッグモードでの実行例を表示している。*Psuite* のディバッグモードの特徴は、カーソルで一時停止位置を指定出来ることや、プログラムの一定箇所を実行しながら特定配列のトレースや図4のような三次元表示を全くソースプログラムの変更なしに処理出来る点にある。この二点だけを取っても、我々のようにソースプログラムの万行を編集し直し、大型汎用機で同じような作業を行ってきた者に取って、*Psuite* は衝撃的な *IT* 進化の時の流れと大いなる隔世の感を抱かせる存在でもある。

6 *Psuite* の実行と *Inter Net* 回線速度

大型汎用機で長年の放ったらかしにしていたバイハーモニックファンクションの数値微分方程式の収束問題も、この *Psuite* のおかげで、計算途中の二次元配列のディバッグを格段に短時間で解決できそうである。

幸いにも我々が使用する *10Mbit/sec LAN* では、問題なく *Psuite* の実行が可能である。只、これは通常の場合で、一旦この *10Mbit/sec LAN* に障害が発生した場合、ホストの *sx5.center.osaka-u.ac.jp* へのアクセスばかりではなく、クライアントとしての *vis01.center.osaka-u.ac.jp* まで反応しなることがある。いくらかのローカルファイルが手元に残っていても、一切の仕事が進まない状態に陥る。センターを絶対に利用しない研究者が信じて疑わないセンター不要論の根拠はこの状態を意味する。かっての大型汎用機は専用線のみを確かにサポートしていたし、当時の *NTT* は専用回線に対して通常の研究費では到底賄えきれない程の課金をしていた。しかし、この時代にセンターは専用線の品質までも保障する努力を払ってくれていたし、アプリケーションがリモート端末でフリーズするというようなことは無かった。勿論、現在センターがアクセスする全てのユーザの経路情報を完全に把握することは至難の技ではあるが、センターの資源を提供する業務である以上、この問題を回避することは出来ない筈である。*ODINS* と我々の *LAN* との距離は *100km* であり、その間の最低回線速度は *1.6Mbit/sec* である。センターが運用するアプリケーションの中でも比較的重たい *AVS* などは、回線速度の問題で正常動作させることは出来ない。*SINET* の幹線以外の支線では茨木市を中心とする半径 *70km* 前後にセンター資源の 100%利用限界が存在するようであり、いつまでも *serve yourself* 的なサービスではセンター不信者が増加することを危惧してならない。

参考文献

[1] NEC. “fortran90/sx 言語説明書”. *nec software Inc.*, 2000.

[2] NEC. “fortran90/sx プログラミングの手引”. *nec software Inc.*, 2000.

[3] NEC. “言語支援機能利用の手引”. *nec software Inc.*, 2000.