FFTJプロジェクト

FFTJ = Fastest Fourier Transform in Japan
(FFTW = Fastest Fourier Transform in the West に対抗して)

状況認識:

ISPACK はこれまで主にベクトル計算機上での高速化を念頭において設計してきた.

しかし, 近年, ベクトル計算機は利用しづらい環境になってきている. (ベクトル計算機を作り続けているのは NEC のみであり, 地球シミュレータや阪大・東北大などの大計センターあたりでしか使えない).

HPC業界はどんどん超並列スカラー計算機に移行していっている(京大大計センターもその一例). 次期(地球)シミュレータ(京速コンピュータ) もベクトル機にはなりえない筈.

従って，今後はこのような計算機環境を念頭にライブラリの開発をしていかざるをえない(個人的には比較的容易に理論ピーク性能近くに到達が可能なベクトル機は好きだったのだが).

これまで:

ライブラリの並列化自体はそれほど困難なことではない.

共有メモリ環境では OpenMP ディレクティブをプログラムに埋め込めばよいし，分散メモリ環境では MPI でプログラムを書けばよい(どちらもデファクトスタンダードになっているので，普通どこでも使える).

ISPACK でも，主要な変換ルーチンにおいてこれらの並列化を施したものを既に作成して公開してある.

しかし，問題はそう簡単ではない.

問題点:

スカラー計算機とベクトル計算機ではCPUのアーキテクチャが根本的に異なるので，ベクトル計算機向けに開発したプログラムそのままではスカラー計算機の性能を十分には引き出せない．

というわけで，並列化を追求する以前に，単一CPUでの性能を十分に引き出すべくプログラムの抜本的な設計変更が必要である．

そのためにはスカラー計算機のアーキテクチャをまず十分に理解しなければならない．

最近やっていること:

身近なスカラー計算機として, Intel Pentium4, Core のアーキテクチャの理解とその上でのプログラムの高速化に取り組んでいる.

これは, Intel Pentium4, Core (またはその互換 CPU) の計算機環境はいたるところで利用可能であり,かつ非常に高速である(プログラムによっては, 1CPUで数GFlopsの演算性能を引き出せる)ので, このCPU上でまず高速化しておけば将来的にも有利である(次期京大大計は Quad Core Opteron(Barcelona)になるらしいし).

また, Pentium4, Core のような広く流通しているCPUで高速化しておくことにより, 「誰にでも簡単に数値実験の追試のできる環境」を提供することが可能になりうると考える(追試のできない実験は実験ではない).

CPUのアーキテクチャを理解し，性能をフルに引き出そうとするならば，アセンブリ言語を理解することが必須である．

最近ではコンパイラもそれなりに賢くなってきてはいるが，Pentium4, CoreのSSE2(Streaming SIMD Extension 2)などを活用しようとするならば，アセンブリ言語でのプログラミングは避けて通れない．

というわけで，ISPACKの高速化に向けた準備段階として，アセンブリ言語の勉強を兼ねて，Pentium4, Core で高速に走る FFT をアセンブリ言語で開発中である(FFTJプロジェクト)．

現在，とりあえず長さ 512 までの FFTを作成してみてある．
benchFFT(`http://www.fftw.org/speed/`) の指標では
Pentium4 Xeon(Prestonia)) や Core 2 Duo Xeon(Woodcrest) でFFTW や Intel IPPS を凌駕している．

開発の方針:

いきなりアセンブリ言語で書くと訳分からなくなってしまうので，基本的アルゴリズムを考えた段階でまず FORTRAN77 で書いてみて，それからアセンブリ言語に書き換える．

こうすることによってFORTAN77プログラムがアセンブリ言語のプログラムの「注釈」のようになるし，バグフィックスが容易になる(基本的アルゴリズムに間違いがあるのか，それともアセンブリ化に誤りがあるのかが明解になる)．

また，もし将来的に PCの CPUのアーキテクチャが大きく変わり，現在書いているアセンブリコードが動かなくなっても(そんなことは当面無いと思われるが)，FORTRAN77コンパイラがあればスピードはともかくとして同じ機能が提供できる．

簡単なアセンブリコードの例を対応するFORTAN77コードと対応させて示す.

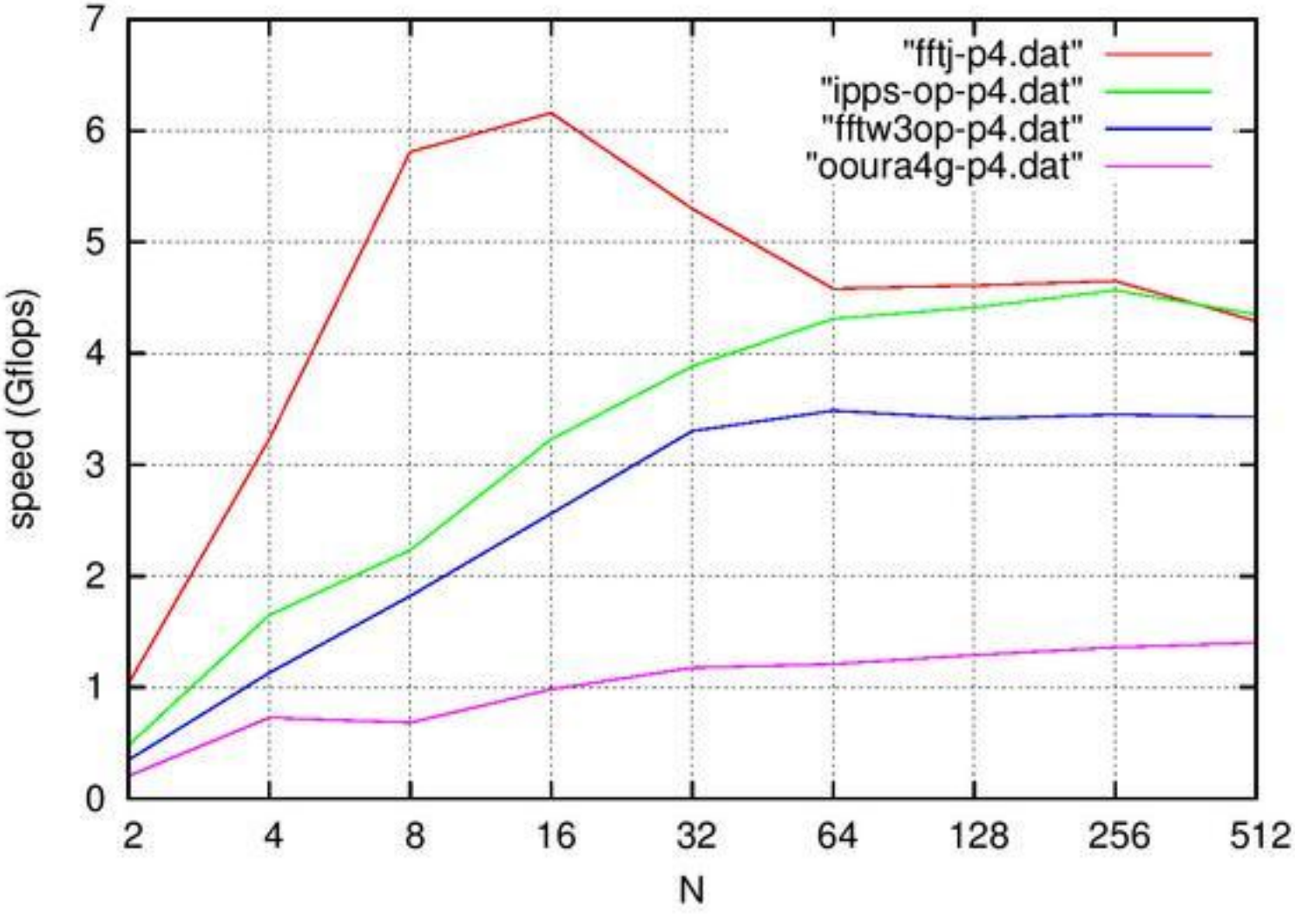
FORTRAN77:

```
SUBROUTINE SUB(A,B)
COMPLEX*16 A,B
B=B+A
END
```

Pentium4, Core用アセンブリ:

```
.globl sub_
sub_:
 movl 4(%esp), %eax # A
 movl 8(%esp), %ecx # B
 movapd (%eax), %xmm0
 movapd (%ecx), %xmm1
 addpd %xmm0, %xmm1
 movapd %xmm1, (%ecx)
 ret
```

FFTの速度比較: Intel Pentium4 Xeon 3.06GHz (Prestonia)



FFTの速度比較: Intel Core2 Duo Xeon 3.0GHz (Woodcrest)