Parallel 3-dim FFT's for electronic structure calculations

Stefan Goedecker

Basic problems of FFT's:

- Low ratio between floating point operations
  and data (load/store's)
  3-dim FFT:
  - N^3 data points
  - $15N^3 \log_2(N)$ floating point operations
- Large data sets that do not fit into cache
- Highly nonlocal data access pattern
- Large amount of communication for parallel FFT

Multiple 1-dim FFT's for improved data locality

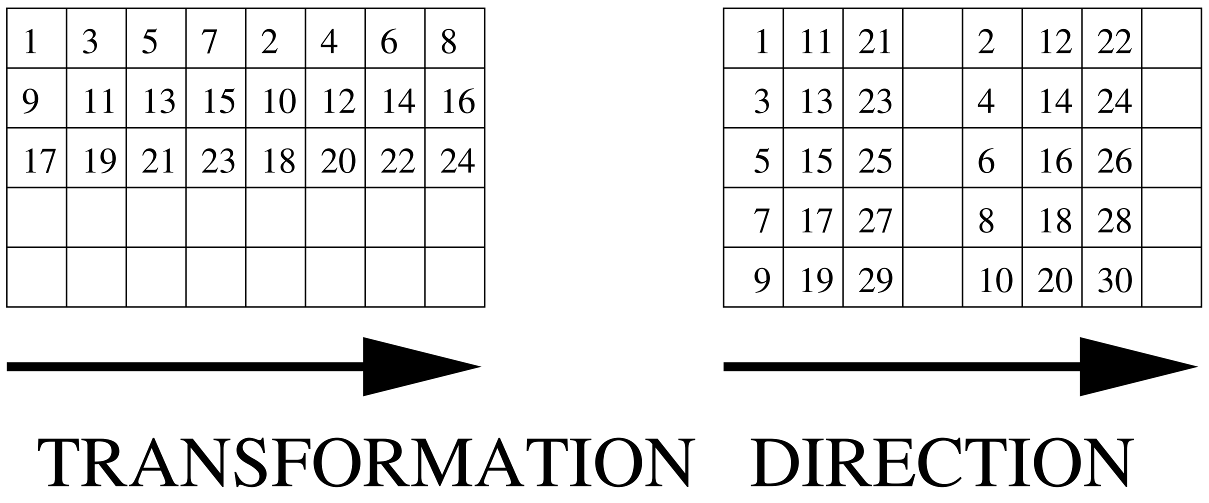


Figure 1: *The data access pattern for a multiple FFT, where five data sets of length eight are transformed. A Fortran column major ordering is assumed. On the left, the inner loop is over a single FFT sweep, resulting in a non-local data access pattern. On the right, the inner loop runs over the five data sets, leading to good spatial data locality.*

Rotation technique for a 3-dim FFT

Convention:

. i1, i2, i3 untransformed dimensions

. I1, I2, I3 transformed dimensions

i1, i2, i3

I3, i1, i2

I2, I3, i1

I1, I2, I3

Cache blocking on hierarchical memory computers

$$(i1, i2), i3 \rightarrow i12, i3 \rightarrow j, k, i3$$

$k = 1, ..., lot$
$j = 1, ..., m12 = n1 \times n2 / lot$
$m12 \times n3 \leq$ cache-size

OpenMP parallelization

Parallelize k loop

Performance results

Time (speedup) in seconds for a single 3-dim transform of size 128^3

- on DEC Alpha, 666 MHz (.41 sec gives 540 Mflops)

  Data from Philippe Blaise, Centre de Calcul CEA Grenoble

Numb. Proc.'s	DEC CXML	My OpenMP	My MPI	FFTW (serial/MPI)
serial	.36	.41	.94	.87
1	.87	.41 (1.)	.94 (1.)	1.31
2	.37	.25 (1.6)	.50 (1.9)	.99
4	.20	.16 (2.6)	.27 (3.5)	.45
8	.18		.17 (5.5)	.47
16	.13		.12 (7.8)	.47

- IBM Power3

  Data from Andrew Canning, NERSC, Berkeley

Numb. of Proc.'s	1	2	4	8	16
time (speedup)	.81 (1.)	.40 (2.)	.21 (3.9)	.12 (6.7)	.09 (9.0)

3-dim FFT algorithm for distributed memory

Input:	i1,i2,j3,(jp3)
multiple 1-dim FFT:	i1,I2,j3,(jp3)
Rotation:	I2,i1,j3,(jp3)
multiple 1-dim FFT:	I2,I1,j3,(jp3)
Rotation:	I1,I2,j3,(jp3)
Previous data set reformatted:	I1,J2,Jp2,j3,(jp3)
Copy:	I1,J2,j3,Jp2,(jp3)
MPI_ALLTOALL:	I1,J2,j3,jp3,(Jp2)
Previous data set reformatted:	I1,J2,i3,(Jp2)
multiple 1-dim FFT:	I1,J2,I3,(Jp2)
Copy:	I1,I3,J2,(Jp2)

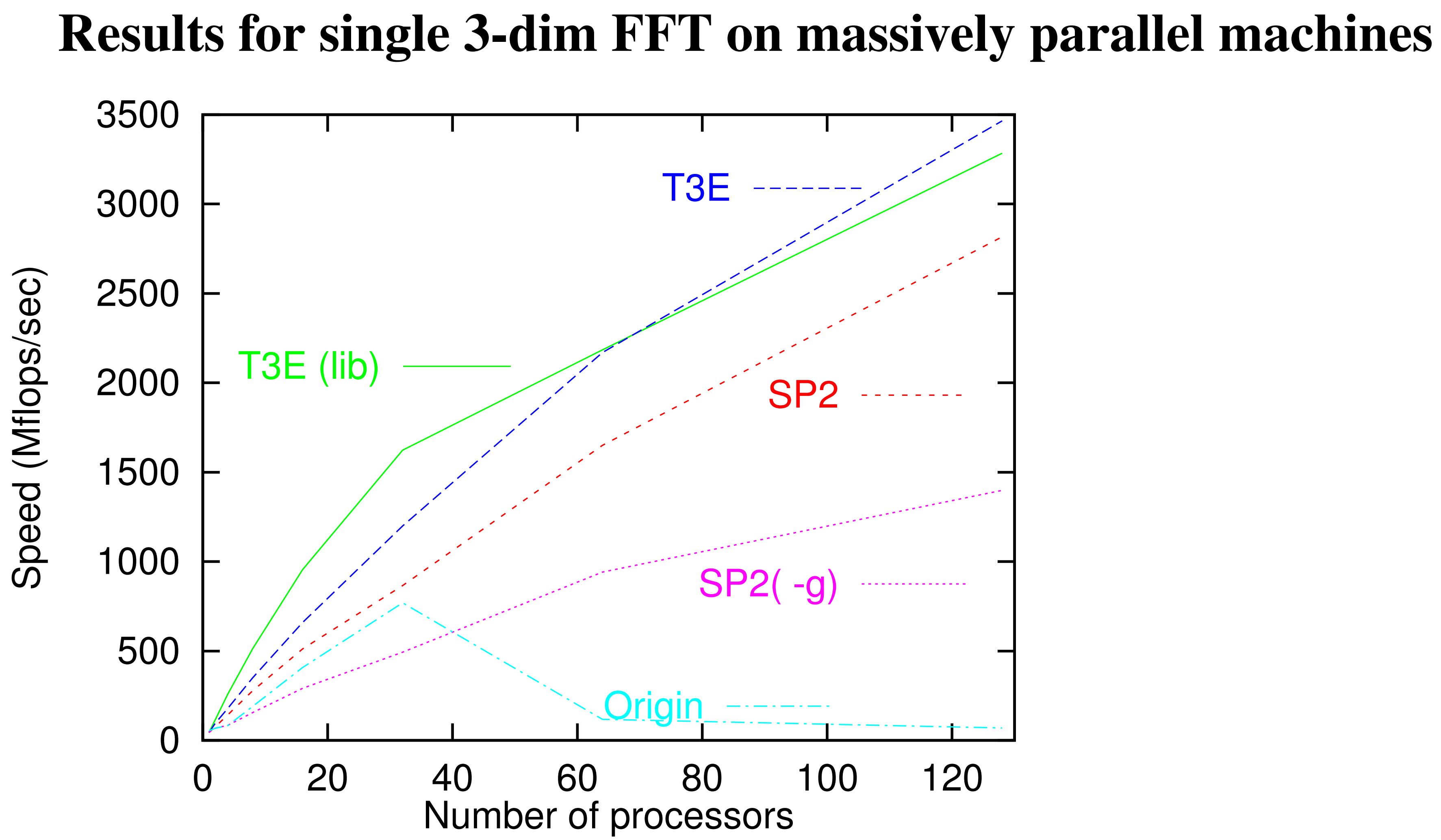
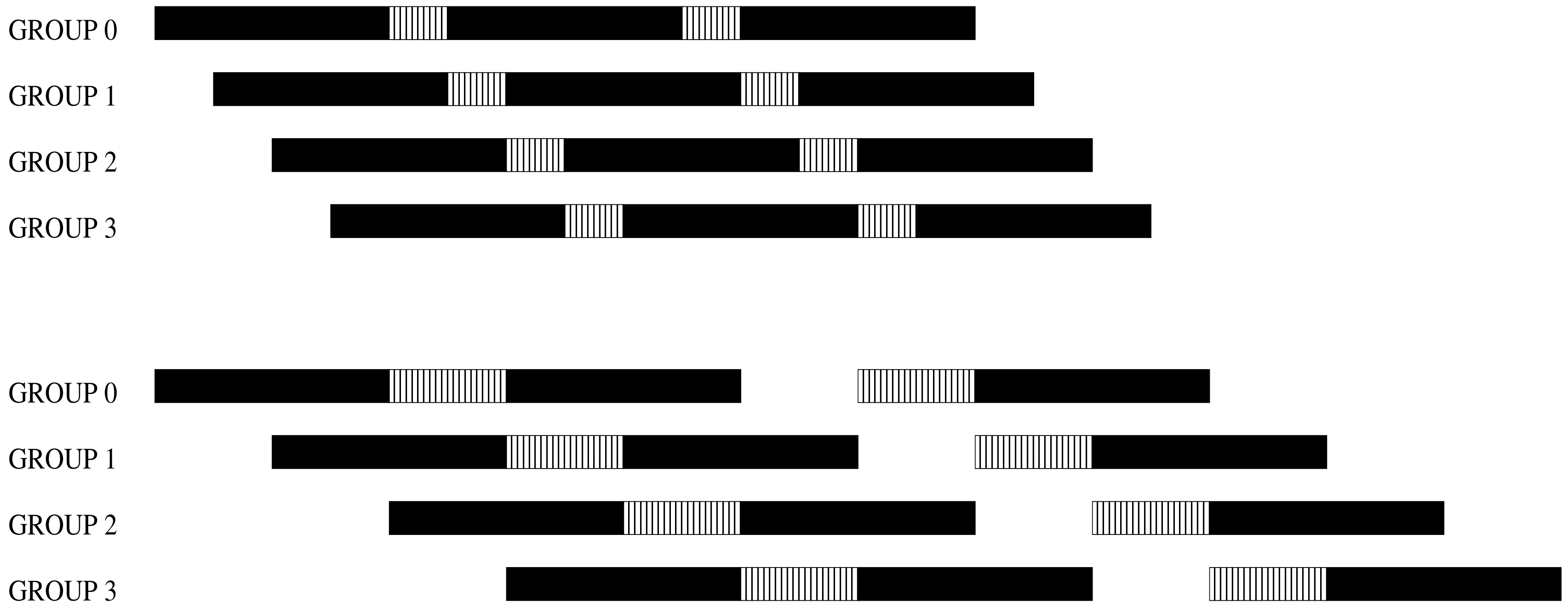


Figure 2: The parallel performance of a 128^3 FFT on the Cray T3E, IBM SP2 and SGI Origin2000. On the Cray we show both the performance of our implementation and that of the PCCFFT3D library, denoted by "lib"

Multiple 3-dim FFT’s on multiprocessor nodes

Overlap communication and computation



Multiple 3-dim convolutions on multiprocessor nodes

Application of local potential on wavefunction is a convolution

- FFT from Fourier into real space with zero padding to eliminate aliasing errors
- Multiplication of wavefunction with potential in real space
- FFT from real into Fourier space

Advantages:

- Since the data sets in Fourier space are 8 times smaller than in real space the amount of communication can be reduced
- Cache blocking can be done in a combined way for the last sweep in the initial FFT the multiplication with the potential in real space and the first sweep of the final FFT

Results for multiple 3-dim convolutions massively parallel machines

Table 1: Timings, [speed in Gflops] and (speedup) of the MPI and mixed OpenMP/MPI implementation on a Crat XT3 and a Compaq SC for 3-dim multiple FFTs.

	1 XT3 MPI	SC MPI	SC 1 mixed	SC 2 mixed	SC 4 mixed
1		2.91	2.93	1.72 (1.7)	0.84 (3.5)
2	1.0 [2.3]	1.63 (1.8)	1.62 (1.8)	0.84 (3.5)	0.45 (6.6)
4	.52 [4.6]	0.88 (2.5)	0.88 (3.3)	0.46 (6.3)	0.25 (11.9)
8	.25 [9.5]	0.54 (5.4)	0.47 (6.3)	0.25 (12.0)	0.14 (20.3)
16	.13 [19]	0.25 (11.7)	0.24 (12.3)	0.13 (22.9)	0.081 (36.4)
32	.071 [34]	0.13 (22.7)	0.13 (22.6)	0.075 (38.9)	0.050 (58.4)
64	.034 [70]	0.066 (43.8)	0.067 (43.7)	0.037 (79.0)	0.032 (91.8)
128	.018 [134]	0.040 (72.0)	0.036 (81.2)	0.019 (158)	0.018 (163)

Conclusion

- Even though standard single FFT's are difficult to parallelize, convolutions can give very high performance on massively parallel computers with fast networks

References:

- S. Goedecker: Comp. Phys. Commun. **76**, 294 (1993)
- S. Goedecker: SIAM Journal on Scientific Computing **18**, 1605 (1997)
- S. Goedecker, M. Boulet, T. Deutsch: Comp. Phys. Commun. **154** 105 (2003)

Parallelization of Wavelet based version of Abinit

2 types of datastructures

- Convolutions and fast wavelet transformations are not parallelized. Each processor treats one or several orbitals.

```
                              I,iorb,(jorb)
```

- In the orthogonalization part each processor has a fraction of the coefficients of all the wavefunctions. This datastructure is obtained from the previous one in the following way:

```
                          i,j,iorb,(jorb)

 Copy:                    i,iorb,j,(jorb)

MPI_alltoall              i,iorb,jorb,(j)

                          i,IORB,(j)
```