



Stanford Streaming Supercomputer StreamFlo

Massimiliano Fatica

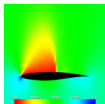


Motivations

StreamFlo is the Brook version of FLO82, a FORTRAN code written by Prof. Jameson, for the solution of the inviscid flow around an airfoil.

The main objectives of this porting are:

- Evaluate performances of a typical CFD code on the Streaming Supercomputer. The structure of the code is similar to TFlow and the algorithm is found in many compressible flow solvers.
- Evaluate the issues involved in porting legacy FORTRAN codes to Brook



Governing Equations

StreamFlo solves the Euler equations:

$$\frac{d}{dt} \int_{\Omega} w dx dy + \int_{\partial\Omega} (f dy - g dx) = 0,$$

where w , f and g are respectively the vector of unknowns and the inviscid flux in the x and y directions.

$$w = \begin{Bmatrix} \rho \\ \rho u \\ \rho v \\ \rho E \end{Bmatrix}, \quad f = \begin{Bmatrix} \rho(u - u_0) \\ \rho u(u - u_0) + p \\ \rho v(u - u_0) + pu \\ \rho E(u - u_0) + pu \end{Bmatrix}, \quad g = \begin{Bmatrix} \rho(v - v_0) \\ \rho v(u - u_0) \\ \rho v(v - v_0) + p \\ \rho E(v - v_0) + pv \end{Bmatrix}$$

This set of equation is completed by an equation of state for the pressure

$$p = (\gamma - 1) \rho \left[E - \frac{1}{2}(u^2 + v^2) \right]$$

A cell centered finite volume formulation is used to solve the equations together with a multigrid acceleration. Time integration is performed by a five stage Runge-Kutta scheme.

Code organization

External driver controls the multigrid strategy and the multigrid data movement (transfer from fine to coarse grid and vice versa). Data is stored in a 1D array which holds all the multigrid levels.

On each multigrid level, the code operates on 2D grids where it needs to compute the time step. Most of the operations are performed on stencils (3x3 and 5x5).

The code uses O-meshes: one direction is periodic, while the other direction has the profile surface and the far field boundary. Brook has been modified to implement operators with independent boundary conditions in each logical direction.

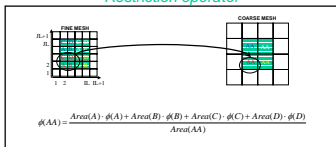
From FORTRAN to Brook

To port a FORTRAN code to Brook there are two essential steps:

- Define the data layout and arrange the data in streams
- Define the computational kernels to be performed on the streams

The following example shows how the original FORTRAN code for the restriction operator in the multigrid is written in Brook

Restriction operator



FORTRAN implementation

```

C ----- TRANSFER THE SOLUTION TO A COARSER MESH -----
C ----- TRANSFER THE SOLUTION TO A COARSER MESH -----
DO N=1,4
  JJ = J-1
  DO J=J-2,2
    JJ = JJ + 1
    II = I + 1
    DO I=I-2,2
      II = II + 1
      WWRB(J,I) = (VOL(I,J) + VOL(I,J-1) + VOL(I,J+1) + VOL(I-1,J) + VOL(I-1,J-1) + VOL(I-1,J+1) + VOL(I+1,J) + VOL(I+1,J-1) + VOL(I+1,J+1)) / 9
    END DO
  END DO
END DO

```

Brook: define the data movement

```

// Fine mesh: flow is a 2D stream of shape (nx+2,ny+2)
// Coarse mesh: flow_coarse is a 2D stream of shape (nx/2+2,ny/2+2)
// Select interior points on the fine mesh
streamDomain(flow_flow,2,2,nx+1,2,ny+1)
streamDomain(vol_vol,2,2,nx+1,2,ny+1)
// Generate a stream with the groups (A,B,C,D), (A,B,C,D), (A,B,C,D), (A,B,C,D)
streamGroup(local_flow2d_flow,STREAM_GROUP_HALO,2,2,2);
streamGroup(local_vol2d_vol,STREAM_GROUP_HALO,2,2,2);
// Apply restriction operator to generate stream with (AA,AA,AA,AA)
TransferFieldFineCoarse(local_flow2d_flow,local_vol2d_vol,coarse_flow)

```

Brook: define the computation

```

kernel void TransferFieldFineCoarse(flow2d_s_flow_flow,local2d_s_vol_outflow_flow_coarse_flow)
{
  coarse_flow_rho = ( vol[0]* fine_flow[0][0]*rho + vol[1]* fine_flow[0][1]*rho
                    + vol[3]* fine_flow[1][0]*rho + vol[3]* fine_flow[1][1]*rho ) /
                    ( vol[0][0] + vol[0][1] + vol[1][0] + vol[1][1] );
  .....
}

```

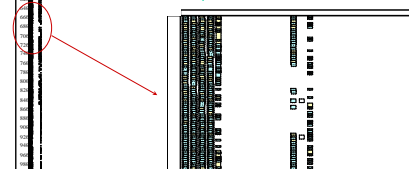
Preliminary results

This shows the schedule of the kernel for the computation of the diffusive flux using a symmetric limited positive H-CUSP scheme.

Set up of the 5x5 stencil: %20 of utilization



Computation of the flux: %90 of utilization



This kernel currently runs at 37 GFLOPS. Improving the setup of the stencils (that is now utilizing only 20% of the resources), will bring the sustained GFLOPS up to 50.

Conclusions

- The interaction between the application developers and the language development group has helped insure that Brook can be used to code real scientific application.
- StreamFlo is in the final stages of development: all the subroutines have been coded in Brook and the results are being validated with the FORTRAN version.
- The code is very clean and easy to understand.
- Initial performance studies are very encouraging.

Future work

Continue work on StreamFlo to improve its capabilities:

- 3D version
- Viscous flow

Evaluate the performance on the multinode simulator

Start the porting of other applications:

- LES
- Purple benchmark (PPM,...)