A++/P++ – Reference Examples
(version 0.7.2)

Daniel Quinlan
Lawrence Livermore National Laboratory
L-560
Livermore, CA 94550
925-423-2668 (office)
925-422-6287 (fax)
dquinlan@llnl.gov
Quinlan's Web Page: http://www.llnl.gov/casc/people/dquinlan
A++/P++ Web Page http://www.llnl.gov/casc/Overture/A++P++
A++/P++ Manual (postscript version)
A++/P++ Quick Reference Manual (postscript version)
LACC Number: LA-CC-96-1
LAUR Number: LA-UR-95-3273

August 16, 2000

August 16, 2000

Chapter 1

Examples: Code Fragments

This is a collection of example A++/P++ code fragments. It is intended to show some of the many ways that A++/P++ can be used. There are two sections, one on A++/P++ examples and the scond on P++ specific examples that demonstrate parallel features of P++.

1.1 A++/P++ Examples

These examples are common to both A++ and P++ array classes. They show a complex mix of operations taken from many A++/P++ codes.

```
#define BOUNDS_CHECK
#include "A++.h"

void main ()
   {
     int Array_Size = 100;

  // Index Constructor examples
     Index I ( 1 , Array_Size-2, 1 );  // position=1, count=Array_Size-2, stride=1
     Index J = I;                      // make an Index object J just like  I
     Index K = I-1;                    // make an Index object K just like I-1
     Index L = -I;                     // make L like I but with negative stride
     Index M = 5;                      // make Index object from integer index
     Index N;                          // build an uninitialized Index object
     N = I+1;                          // Index assignment to build N like offset of I

  // Array Constructor examples
     doubleArray A1 (Array_Size);
     floatArray B1 (Array_Size,Array_Size);
     doubleArray C1 (Array_Size,Array_Size,Array_Size);
     intArray D1 (Array_Size,Array_Size,Array_Size,Array_Size);
     floatArray E1 = B1;
     doubleArray F1 = B1(I-1,J);

     double *Fortran_Array_Pointer = new double [Array_Size+1][Array_Size];
     doubleArray G (Fortran_Array_Pointer,Array_Size,Array_Size+1);
```

```
 // Arrays for use in examples below
    doubleArray A (Array_Size,Array_Size);
    doubleArray B (Array_Size,Array_Size);
    doubleArray C (Array_Size,Array_Size);
    doubleArray D (Array_Size,Array_Size);
    double x = 42;

 // example of array-scalar assignment
    A = x;
    A (I) = x;
    A (I-1) = x * x;

// examples of array-array assignment operations and use of Index objects
   B = A;
   B = C = D = A;
   A (I,J) = B (J,J);
   A (I-1,J) = B (I+1,J);

// Scalar indexing
   A (0,12) = x;
   A (5,12) = A (0,12);
   x = A (1,12) + B(0,12);

// examples of array-array arithmitic operations
   A = B + (C * B - D) / A;
   A (I,J) += B (I,J) / C (I,J);
   A (I-1,J) *= B (I+1,J);

// examples of Jacobi relaxation (9-point stencil)
   A (I,J) = ( A (I+1,J+1) + A (I,J+1  ) + A (I-1,J+1) + A (I+1,J  ) +
               A (I-1,J  ) + A (I+1,J-1) + A (I,J-1  ) + A (I-1,J-1) ) / 8.0;

// examples of Jacobi relaxation (5-point stencil)
   A (I,J) = ( A (I,J+1) + A (I,J-1) + A (I+1,J) + A (I-1,J) ) / 4.0;

// more complex operations
   B (I,J) = ( A (I-1,J-1) * B (I+1,J+1) + C (I-1,J) * D (I,J+1) -
               D (I,J) * B (I,J) * ( A (I,J) - B (I,J) ) ) / ( C (I,J) + D (I,J) );

// examples of relational operator
   intArray Mask = B >= C;
   Mask = !B;
   Mask = !(B && C) != (!B | !C); // DeMorgan's Law

// example of replace operator
   A (I,J).replace ( B (I,J) <= 0.001 , 0.001 );
   A (I,J).replace ( A (I,J) <= C(I,J) , C(I,J) );

// simple example of "where" statement
   where ( B >= C)
        A = 0.001;

// more complex example of "where" used for multiple statement block
   where ( B(I,J) >= C(I,J) )
      {
        A(I,J) = ( A (I,J+1) + A (I,J-1) + A (I+1,J) + A (I-1,J) ) / 4.0;
        B(I,J) = 0.001;
        C(I,J) = 0.001;
      }

// examples of max function use
   x = max (B);
   A = max (B , C * B);
   A = max (B , C , A);

// examples of miscellaneous function use
   x = sum (B);
   A = cos (B) * sqrt (C);
   B(I,J) = (cos (B) * 2.0 )(I,J);

// examples of changing bases of array objects
   A.setBase (1);     // Force A to have indexing similar to Fortran array
   setGlobalBase (1); // Set all future arrays to have Fortran like base of 1
   A.setBase (x);
   A.setBase(x) = B;  // Shows value returned from setBase
   A.setBase (x,0);
   A.setBase (x*x,1);

// examples of bases and bound access
   Array_Size = A.dimension(0);
   printf ("Number of elements in A = %d \n",A.elementCount();
   for (int j = A.getBase(1); j <= A.getBound(1); j++)
        for (int i = A.getBase(0); i <= A.getBound(0); i++)
             A(i,j) = foo (i,j);

// examples of display functions
   A (I,J).display("This is A (I,J)");
   A = B + (C * D).display("This is C * D in expression A = B + (C * D)");
   (A = B * D).view("This is A = B * D");
   A.view("This is A (same view as above)");

// 2 ways to pass array objects by reference
   void foo ( const doubleArray & X );
   foo ( evaluate (A + B) );

   C = A + B;
   foo ( C );

// passing array objects by value requires no special handling
   void foobar ( const doubleArray X );
   foobar ( A + B );

// examples of fill functions
   A(I,J).fill(x);

   printf ("PROGRAM TERMINATED NORMALLY \n");
  }
```

1.2 P++ Specific Examples

This section presents some examples that are specific to parallel P++ operations. These example deal directly with the distributions of array objects onto the multiple processors available within the parallel environment.

```
#define BOUNDS_CHECK
#include <A++.h>

int main(int argc, char** argv )
   {
     Index::setBoundsCheck (On);
     int numberOfProcessors = 128;
  // P++ looks for the application name if "" is specified
     Optimization_Manager::Initialize_Virtual_Machine("",
             numberOfProcessors,argc,argv);

  // Example of using a partition object (assume number of procesors is >= 64)
     Partition_Type Partition_A (64);   // Build partition object which uses processors 0-63
     floatArray A(100,100,Partition_A); // Build array using default "block-block" distribution
                                        // across the processors represented by Partition_A.

  // Example of distribution onto subrange of processors
     Range ProcessorSubrange_B (27,37);
     Partition_Type Partition_B (ProcessorSubrange_B); // Build partition object which uses processors 27-3
     floatArray B(100,100,Partition_B);                // Build an array distributed "block-block"over
                                                       // processors 27-37

  // Simple example of alignment specification
     Range all;               // Default range object implies "all" of wherever it is used
     floatArray C (100,100);  // Build array "block-block" over all processors
     floatArray D = C (0,all); // Align D with boundary of C

  // Simple example of array redistribution
     Range all;                                        // Default range object implies "all" of wherever it
     Range ProcessorSubrange_E (45,83);
     Partition_Type Partition_E (ProcessorSubrange_E); // Build partition object which uses processors 45-8
     floatArray E (100,100,Partition_E);               // Build array "block-block" over processors 45-83
     floatArray F = E (0,all);                         // Align F with boundary of E
     Partition_E.SpecifyProcessorRange(Range(2-12));   // Redistribute E and F on to processors 2-12
                                                       // note that F is STILL aligned with the boundary of

  // More complex redistribution example.  This example builds a collection
  // of different sized arrays each associated with the same partitioning object.
  // then the arrays are all repartitioned through simple manipulation of the
  // partition object.  the arrays are initially distributed onto processor 0,
  // then on an increassing number of processors until all processor are used,
  // then repartitioned onto a decreassing number of processors untill finally
  // distributed only on processor zero.
     int Size = 10;
     Partitioning_Type Partition (Range(0,0));
     doubleArray Temp_A(Size,Partition);
     doubleArray Temp_B(Size*2,Partition);
     doubleArray Temp_C(Size/2,Partition);
     doubleArray Temp_D(Size*2,Partition);
     doubleArray Temp_E(Size/2,Partition);
     doubleArray Temp_F(Size,Partition);
   int i;
   for (i=0; i < Communication_Manager::Number_Of_Processors; i++)
        Partition.SpecifyProcessorRange (Range(0,i));  // redistribute all arrays associated with "Partition"
   for (i=0; i < Communication_Manager::Number_Of_Processors; i++)
        Partition.SpecifyProcessorRange (Range(i,numberOfProcessors-1));

// Example using scalar indexing on local part of distributed array
   intArray v(100);
   int ibas = v.getLocalBase(0);
   int ibnd = v.getLocalBound(0);
   Optimization_Manager::setOptimizedScalarIndexing (On);
   for (int i=ibas; i<=ibnd; i++)
        v(i) = i;
   Optimization_Manager::setOptimizedScalarIndexing (Off);

// Example of getting local A++ array within P++ distributed array
   floatArray X (100,100);  // distributed array
   floatSerialArray X_local = X.getLocalArray();  // Deep copy of local data
   floatSerialArray X_local (X.getLocalArray(),SHALLOW_COPY);  // Shallow copy of local data
   floatSerialArray X_local (X.getLocalArray(),DEEP_COPY);  // Deep copy of local data
   floatSerialArray *X_pointer_to_local = X.getSerialArrayPointer(); // pointer to local data
```