Subject: what is the largest Array size please? ("out of memory" error)
Posted by nixnixnix on Wed, 19 Mar 2008 17:11:11 GMT
View Forum Message <> Reply to Message

I am dealing with large amounts of data and was attempting to dimension an array of the form

typedef WithDeepCopy<Array <doubleArray> > d2Array;

where doubleArray is defined as

typedef WithDeepCopy<Array <double> > doubleArray;

the d2Array enables me to have large fast grids of double precision floats and refer to it conveniently as

myArray[x][y]

Most of the time this works fine but when I tried to dimension a grid of doubles with

number of x = 13393
number of y = 5951

it falls over with "Out of memory" error. Now I would expect this to happen as the array size approaches 2GB but this array size appears to be a "mere" 617MB

I have 4GB (3.2GB usable) of memory on my PC and it wasn't anywhere near all used up. It fell over about 2/3 of the way through dimensioning.

Is this by design or a fault?

Nick

Subject: Re: what is the largest Array size please? ("out of memory" error)
Posted by mirek on Fri, 21 Mar 2008 23:49:21 GMT
View Forum Message <> Reply to Message

nixnixnix wrote on Wed, 19 March 2008 13:11
typedef WithDeepCopy<Array <double> > doubleArray;

There can be 2 issues involved...

First, Array<double> is not optimal - this will need about

N * max(16, sizeof(double)) + N * sizeof(void *) * 1.33

memory.

Try using Vector<double> instead. You might also try SetCount or Shrink to get that "1.33" down to 1.

Then, the memory, at least for Vector (or "sizeof(void *") need to be continous for single Vector (or Vector<void *> inside Array). On Win32, this might be a problem.

Anyway,

Quote:
array size appears to be a "mere" 617MB Smile"

well:

13393*5951*(16 + 1.33 * 4) = 1.7GB

Which looks like some 2GB limitation

Mirek

Subject: Re: what is the largest Array size please? ("out of memory" error)
Posted by nixnixnix on Sat, 22 Mar 2008 16:51:43 GMT
View Forum Message <> Reply to Message

Thanks Mirek,

I'll replace it with a straight memory allocation when I get time as its the sort of object that gets allocated and initialised once and then queried lots.

Cheers,

Nick

Subject: Re: what is the largest Array size please? ("out of memory" error)
Posted by mirek on Sat, 22 Mar 2008 19:28:07 GMT
View Forum Message <> Reply to Message

nixnixnix wrote on Sat, 22 March 2008 12:51Thanks Mirek,

I'll replace it with a straight memory allocation when I get time as its the sort of object that gets allocated and initialised once and then queried lots.

Cheers,

Nick

Well, it will be intersting how this will evolve. IMO, due to continuous area problem, I would still consider allocating one block per line...

Mirek

Subject: Re: what is the largest Array size please? ("out of memory" error)
Posted by nixnixnix on Sat, 22 Mar 2008 21:11:14 GMT
View Forum Message <> Reply to Message

Ah - nice idea! You mean make a grid of values by using an array (Y) of pointers to lines (X) of values (Z) and so keep the speed of access whilst keeping the size to a minimum (overhead of extra 8 bytes per line max). I like it!

Thanks,

Nick

p.s. I just went over the 2Gb mark again with non-gridded data but will attempt to apply the same logic.

Subject: Re: what is the largest Array size please? ("out of memory" error)
Posted by nixnixnix on Wed, 26 Mar 2008 12:10:45 GMT
View Forum Message <> Reply to Message

Hi Mirek,

So I've made a class to act as a container for a 2d array.

There appears to be a problem with the destructor but I can't see that there is anything wrong it. Is this something to do with th way that UPP does garbage collection?

I have an Array of Turbine objects which is defined

WithDeepCopy<Array <Turbine> > m_turbs;

and each Turbine object contains several Grid2d objects. Whenever I try to delete the Turbine

objects, I get an error saying there is heap corruption.

Nick

```
class Grid2d
{
public:
 Grid2d(){m_nRows=m_nCols=0;m_ppData=NULL;}// default constructor
 Grid2d(const Grid2d& grid)       // copy constructor
 {
  m_nRows=m_nCols=0;
  m_ppData = NULL;

  if(grid.m_nRows==0 || grid.m_nCols==0)
  {
   return;
  }

  if(Dimension(grid.m_nCols,grid.m_nRows))
  {
   // copy values
   for(int i=0;i<grid.m_nCols;i++)
   {
    for(int j=0;j<m_nRows;j++)
    {
     this->Set(i,j,grid.m_ppData[i][j]);
    }
   }
  }
 }

 Grid2d& operator=(const Grid2d& grid)
 {
  m_nRows=m_nCols=0;
  m_ppData = NULL;

  if(grid.m_nRows==0 || grid.m_nCols==0)
  {
   return *this;
  }

  if(Dimension(grid.m_nCols,grid.m_nRows))
  {
   // copy values
   for(int i=0;i<grid.m_nCols;i++)
   {
    for(int j=0;j<m_nRows;j++)
```

```
   {
    this->Set(i,j,grid.m_ppData[i][j]);
   }
  }
 }

 return *this;
}

virtual ~Grid2d()
{
 Clear();
}

void Clear()
{
 if(m_nCols==0)
  return;

 for(int i=0;i<m_nCols;i++)
 {
  delete m_ppData[i];
 }
 delete m_ppData;
 m_ppData=NULL;
 m_nRows=m_nCols=0;
}

int GetRows(){return m_nRows;}
int GetCols(){return m_nCols;}

bool SetSize(int nCols,int nRows,double val=-99999)
 {return Dimension(nCols,nRows,val);}
bool Dimension(int nCols,int nRows,double val=-99999)
{
 if(m_ppData!=NULL)
 {
  if(nRows==m_nRows && nCols==m_nCols)
  {
   if(val>-99999)
   {
    for(int i=0;i<nCols;i++)
    {
     for(int j=0;j<nRows;j++)
     {
      m_ppData[i][j] = val;
     }
    }
```

```
  }

  return true;
 }

 Clear();
}

m_nCols = nCols;
m_nRows = nRows;

m_ppData = new double*[nCols];

if(m_ppData==NULL)
{
 m_nRows=m_nCols=0;
 return false; // failed memory allocation so return false
}

for(int i=0;i<m_nCols;i++)
{
 m_ppData[i] = new double[nRows];

 if(m_ppData[i]==NULL)
 {
  // we failed to allocate memory so destroy everything and return false
  for(  ;i<=0;i--)
  {
   delete m_ppData[i];
  }
  delete m_ppData;
  m_ppData = NULL;
  m_nRows=m_nCols=0;
  return false;
 }

 if(val>-99999) // init values
 {
  for(int j=0;j<m_nRows;j++)
  {
   m_ppData[i][j] = val;
  }
 }
}

 return true;
}
```

```
 double* operator[](int nCol)   {return GetCol(nCol);}

 inline double* GetCol(int nCol)
 {
  if(nCol>=0 && nCol<m_nCols)
   return m_ppData[nCol];
  else
   return m_ppData[0];
 }

 double Get(int nCol,int nRow)   {if(nRow>=0 && nCol>=0 && nRow<m_nRows &&
nCol<m_nCols)return m_ppData[nCol][nRow];}
 void Set(int nCol,int nRow,double val) {if(nRow>=0 && nCol>=0 && nRow<m_nRows &&
nCol<m_nCols)m_ppData[nCol][nRow]=val;}

 virtual void Serialize(Stream& s)
 {
  int nRows = m_nRows;
  int nCols = m_nCols;

  s % nCols % nRows;

  if(s.IsLoading())
  {
   if(!Dimension(nCols,nRows))
    return;
  }

  for(int i=0;i<m_nCols;i++)
  {
   for(int j=0;j<m_nRows;j++)
   {
    s % m_ppData[i][j];
   }
  }
 }

protected:

 // current size of grid
 int m_nRows;
 int m_nCols;

 // data values
 double** m_ppData;
};
```

Subject: Re: what is the largest Array size please? ("out of memory" error)
Posted by mirek on Wed, 26 Mar 2008 12:41:07 GMT
View Forum Message <> Reply to Message

catch(std::bad_alloc){

You can save the effort, U++ does not throw this one

Other than that, heap corruption is most often caused by writing past the end.

Mirek

Subject: Re: what is the largest Array size please? ("out of memory" error)
Posted by mrjt on Wed, 26 Mar 2008 13:04:59 GMT
View Forum Message <> Reply to Message

```
 double Get(int nCol,int nRow){if(nRow>=0 && nCol>=0 && nRow<m_nRows &&
nCol<m_nCols)return m_ppData[nRow][nCol];}
 void Set(int nCol,int nRow,double val){if(nRow>=0 && nCol>=0 && nRow<m_nRows &&
nCol<m_nCols)m_ppData[nRow][nCol]=val;}
```

If the array structure is [double*(columns)][double(rows)] then you have these backwards. If row count and column count are different Set will always write outside of the array on a copy op since you have this code:

```
        // copy values
  for(int i=0;i<grid.m_nCols;i++)
  {
//    memcpy(m_ppData[i], grid.GetCol(i), m_nRows * sizeof(double));

   for(int j=0;j<m_nRows;j++)
   {
    this->Set(i,j,grid.m_ppData[i][j]);
   }
  }
```

Additionally I believe the loop is redundant since you've just copied all the data with memcpy.

Edit: also I believe the delete calls in the contructor should be delete[]

James

Subject: Re: what is the largest Array size please? ("out of memory" error)
Posted by mrjt on Wed, 26 Mar 2008 14:42:22 GMT
View Forum Message <> Reply to Message

Since I've never done it before I thought I might try and write this as a Upp style container with

Pick semantics and template parameter. It's relatively straightforward, I've attached the code in case it's any use.

The only thing I can't figure out is how to make [][] work for non-const access, as I can't find a safe way of passing out the pointer to the row.

File Attachments

1) Grid2D.h, downloaded 299 times

Subject: Re: what is the largest Array size please? ("out of memory" error)
Posted by nixnixnix on Wed, 26 Mar 2008 17:11:01 GMT
View Forum Message <> Reply to Message

Doh!

Thanks James.

EDIT: didn't see your second reply at first. Will try it and get back to you. Thanks a lot for this. I kind of suspected that I should do the pick and deep copy thing but had no clue how to start.

Cheers,

Nick
