Subject: Re: Looking for tetrahedra. Searching sorted lists.
Posted by Karl Schultz on Mon, 15 Aug 2005 16:43:13 GMT
View Forum Message <> Reply to Message

On Sun, 07 Aug 2005 21:37:19 -0700, cgguido wrote:

> Hi all,
>
> I was wondering if anybody can suggest a fast algorithm (perhaps using
> the magic of HISTOGRAM :-o) to find tetrahedra. Let me explain.
>
> I need to find quadruplets of mutually nearest neighbours id's in a
> nearest neighbour list.
>
> I start with a list of nearest neighbours id's, NN which is a 2xn (n ~
> 6e6 yep!) intarr and is sorted so that for any i:
>
> NN[0,i] lt N[1,i] and
> NN[0,i] le NN[0,i+1]
>
> example input:
>
> 0 2
> 0 5
> 0 34
> 1 2 *
> 1 3 *
> 1 4 *
> 1 56
> 2 3 *
> 2 4 *
> 3 4 *
> 3 9
> 3 12
> ...
>
> What I would like is an output of the form 4 x m (m is whatever it is)
> and each row contains the sorted list of ids for each quadruplet.
>
> example output:
>
> 1 2 3 4 <-- Are all neabours to each other.
> ...
>
> Currently my code takes (too) many hours to do this on Xeon 2GHz with
> 1.5GB ram and idl 6.0.
>
> Any help or suggestions would be much appreciated!!

>
> Gianguido
>
> PS: I can post my code if you think it might help.

One thing worth pointing out is that nearly all mesh-related functions in IDL are implemented in C because those algorithms call for a lot of looping and other operations that are not efficient in IDL.

Since you've got an algorithm implemented in IDL and an apparent need to run it repeatedly on lots of data that takes hours to process with IDL, this may be a good example of something to translate to C and package up as a DLM.

You might post the IDL code to see if anyone can spot any obvious "low-hanging fruit". There may be some simple operations that you are doing that are very expensive and can be addressed without understanding or changing the main algorithm. For example, how are you collecting the quads? If you are just appending them to an array to create a new array, that can be costly.

I also can't understand your problem well enough to suggest that IDL code might be able to do it fast enough. The code might better describe what you are doing.

But my gut feel after making a lot of guesses and assumptions is that you may need a C DLM to get more than two or three orders of magnitude improvement.

Karl