Subject: Re: Algorithm for lat/lon searching
Posted by Gordon Sande on Fri, 18 Aug 2006 15:18:09 GMT
View Forum Message <> Reply to Message

On 2006-08-18 11:50:56 -0300, Paul van Delst <Paul.vanDelst@noaa.gov> said:

> Hello,
>
> I want to implement a global *land* surface emissivity database (as a
> LUT) into a radiative transfer code. Fir simplicity the database is
> simply gridded by lat/lon (land and sea). Due to memory limitations, I
> want to only keep the land gridboxes in my lookup table. Obviously,
> doing this complicates searching for the actual lat/lon element since
> they're no longer stored on a grid.
>
> What I'm looking for is a simple and/or quick method for searching a
> somewhat irregularly spaced database for particular points. In the IDL
> newsgroup there was recently a discussion above finding unique number
> pairs (lat->"high" portion of 64 bit int, lon->"low" portion) and I was
> thinking that would provide a searchable database. By converting the
> lat/lon pair to a unique number, e.g.
>
> JD Smith wrote:
> <IDL code follows>
>> epsilon=1.e-7 ; difference in degrees for equality
>> lat_lon = ulong64((lat+90.)/epsilon) + ishft(ulong64(lon/epsilon),32)
>
> the resultant lat_lon array being simple to search.
>
> An additional problem is that, since this data will be used for
> satellite data assimilation and satellites tend to scan "diagonally"
> across lat/lon, adjacent/close-by *geographical* grid elements will be
> accessed and it's not clear to me that the above lat/lon orgainisation
> will put elements separated by a short physical distance anywhere near
> each other in the lat_lon array.
>
> I will be playing with and testing this over the coming days, but I
> wanted to pick the brains of folks out there in advance.
>
> Thanks for any suggestions/advice,
>
> cheers,
>
> paulv
>
> p.s. Since the final code needs to be Fortran95, I set followups to
> comp.lang.fortran

Welcome to multiple key searching.

The granddaddy technique goes by the name of kd-trees. As in K Dimensional trees. When k=2 they are called quad trees. When k=3, oct trees. When ...

The problem is also called nearest neighbour searching with many geographers using natural neighbours as a variant. Also called associative searching or even content directed searching.

This has a large literature with much of the terminology very graph theoretic. Triangulation is an important problem for many so there is much discussion of that. Regular spatial arrangements are called crystals which is a whole field in physics. Geographic databases are pretty common.

If you like combinatorics there are a variety of space filling curves that can be used to keep things which are close in both (real) indices close in their single (referencing) index. The problem you are asking about.

And here you thought it was going to be a simple answer to a simple question!

Isn't this the sort of thing that outfits like NOAA are supposed to be experts in? Unfair question as you have to cross speciality boundaries and wade through arcane terminology. But seriously, there should be folks around there who know this sort of stuff.

Subject: Re: Algorithm for lat/lon searching
Posted by JD Smith on Fri, 18 Aug 2006 17:54:29 GMT

View Forum Message <> Reply to Message

On Fri, 18 Aug 2006 10:50:56 -0400, Paul van Delst wrote:

> Hello,
>
> I want to implement a global *land* surface emissivity database (as a LUT)
> into a radiative transfer code. Fir simplicity the database is simply
> gridded by lat/lon (land and sea). Due to memory limitations, I want to
> only keep the land gridboxes in my lookup table. Obviously, doing this
> complicates searching for the actual lat/lon element since they're no
> longer stored on a grid.

Here's a simple notion:

Why not develop a "whole earth grid" in whatever binning and projection is useful (an equal area projection comes to mind), run all your land points (only) through HIST_ND, store the resulting REVERSE_INDICES, and then, for a given lat/lon, look up its position in the multi-dimensional reverse index vector, and read out the emissivity data points. You don't say how

much information each of those emissivity data points would include, but
storing a reverse index vector is linear in the number of bins, and would
be much faster to access than sorting through constantly.

> p.s. Since the final code needs to be Fortran95, I set followups to
> comp.lang.fortran

Oh, well, that's a different story then, since you'd have to write
your own HISTOGRAM from scratch ;). I guess it depends on your
desired bin size, then. If you can bin the whole earth (or whatever
portion thereof you're discussing) into say, 2^16 points, then map a
given LAT/LON (or other more convenientq re-projected coordinate pair)
to two 8bit integers (aka 1 16bit int, similar to what I showed for
64bit integers), you could keep a simple 65536 element hash table,
each element of which would point to the data contained (using
whatever F95 magic may or may not exist to do that: pointers to a
linked lists would come to mind for us C programmers). You could
implement more than 8bit of gridding per dimension, at the cost of
memory. 10bits each (1024 elements) would only occupy about 4MB.

If you need much finer gridding, you could still use a coarse hashed
grid of this form to get to the general area, but then perform a finer
grained (e.g. bounding box) search through the data points pointed to
in the indicated grid cell (or small set of adjacent grid cells).

JD

Subject: Re: Algorithm for lat/lon searching
Posted by news.qwest.net on Fri, 18 Aug 2006 19:17:01 GMT

View Forum Message <> Reply to Message

"JD Smith" <jdsmith@as.arizona.edu> wrote in message
news:pan.2006.08.18.17.54.28.887505@as.arizona.edu...
> On Fri, 18 Aug 2006 10:50:56 -0400, Paul van Delst wrote:
..
> Here's a simple notion:
>
> Why not develop a "whole earth grid" in whatever binning and projection is
> useful (an equal area projection comes to mind), run all your land points
> (only) through HIST_ND, store the resulting REVERSE_INDICES, and then, for
> a given lat/lon, look up its position in the multi-dimensional reverse
> index vector, and read out the emissivity data points.

That is a good solution, and one that I have employed in the past.
I would just point out that it gives a square of data (or in general
rectangle
in lat and lon), so some points along the diagonal are farther away than

others.
If you want to get a constant radius disk, you will have to calculate
distances
(perhaps something like what I suggested in my other post).

Having said that I doubt that it would make much of a difference, and I
would
go with the square grid approach.

Cheers,
bob

Subject: Re: Algorithm for lat/lon searching
Posted by Paul Van Delst[1] on Fri, 18 Aug 2006 21:23:34 GMT
View Forum Message <> Reply to Message

Hello,

JD Smith wrote:
> On Fri, 18 Aug 2006 10:50:56 -0400, Paul van Delst wrote:
>
>> Hello,
>>
>> I want to implement a global *land* surface emissivity database (as a LUT)
>> into a radiative transfer code. Fir simplicity the database is simply
>> gridded by lat/lon (land and sea). Due to memory limitations, I want to
>> only keep the land gridboxes in my lookup table. Obviously, doing this
>> complicates searching for the actual lat/lon element since they're no
>> longer stored on a grid.
>
> Here's a simple notion:
>
> Why not develop a "whole earth grid" in whatever binning and projection is
> useful (an equal area projection comes to mind), run all your land points
> (only) through HIST_ND, store the resulting REVERSE_INDICES, and then, for
> a given lat/lon, look up its position in the multi-dimensional reverse
> index vector, and read out the emissivity data points. You don't say how
> much information each of those emissivity data points would include, but
> storing a reverse index vector is linear in the number of bins, and would
> be much faster to access than sorting through constantly.
>
>> p.s. Since the final code needs to be Fortran95, I set followups to
>> comp.lang.fortran
>
> Oh, well, that's a different story then, since you'd have to write
> your own HISTOGRAM from scratch ;).

Well, the final code that accesses the LUT needs to be f95, but I can prepare the datafile
  offline however I like. If I only have to histogram the data once, and store those
indices in the datafile along with the data itself, that's fine.

If I understand what you're saying, the problem is that I can't have a "whole earth grid"
- I only want land data points. I.e. I might have an array of 720x360 (lon x lat) that I
would apply a land/sea mask to. That would give me, say, 30% of the original data where
there is no longer any regular grid (except within continental areas I guess).

> I guess it depends on your
> desired bin size, then. If you can bin the whole earth (or whatever
> portion thereof you're discussing) into say, 2^16 points, then map a
> given LAT/LON (or other more convenientq re-projected coordinate pair)
> to two 8bit integers (aka 1 16bit int, similar to what I showed for
> 64bit integers), you could keep a simple 65536 element hash table,
> each element of which would point to the data contained (using
> whatever F95 magic may or may not exist to do that: pointers to a
> linked lists would come to mind for us C programmers). You could
> implement more than 8bit of gridding per dimension, at the cost of
> memory. 10bits each (1024 elements) would only occupy about 4MB.

My initial thoughts were along that line. Put together a list of land position keys based
on your packing the lat/lon into a single number, use that algorithm to compute the
required key, and then search the hash table for the emissivity data values.

Whenever I need to do any searching I always try to get a second (and third, ...) opinion. :o)

The spatial index searching that Gordon Sande mentioned might also be the go (if i can
figure out how to do it).

cheers,

paulv

--
Paul van Delst   Ride lots.
CIMSS @ NOAA/NCEP/EMC   Eddy Merckx
Ph: (301)763-8000 x7748
Fax:(301)763-8545

Subject: Re: Algorithm for lat/lon searching
Posted by Paul Van Delst[1] on Fri, 18 Aug 2006 21:26:02 GMT
View Forum Message <> Reply to Message

Gordon Sande wrote:
> On 2006-08-18 11:50:56 -0300, Paul van Delst <Paul.vanDelst@noaa.gov> said:

>
>> Hello,
>>
>> I want to implement a global *land* surface emissivity database (as a
>> LUT) into a radiative transfer code. Fir simplicity the database is
>> simply gridded by lat/lon (land and sea). Due to memory limitations, I
>> want to only keep the land gridboxes in my lookup table. Obviously,
>> doing this complicates searching for the actual lat/lon element since
>> they're no longer stored on a grid.
>>
>> What I'm looking for is a simple and/or quick method for searching a
>> somewhat irregularly spaced database for particular points. In the IDL
>> newsgroup there was recently a discussion above finding unique number
>> pairs (lat->"high" portion of 64 bit int, lon->"low" portion) and I
>> was thinking that would provide a searchable database. By converting
>> the lat/lon pair to a unique number, e.g.
>>
>> JD Smith wrote:
>> <IDL code follows>
>>> epsilon=1.e-7 ; difference in degrees for equality
>>> lat_lon = ulong64((lat+90.)/epsilon) + ishft(ulong64(lon/epsilon),32)
>>
>> the resultant lat_lon array being simple to search.
>>
>> An additional problem is that, since this data will be used for
>> satellite data assimilation and satellites tend to scan "diagonally"
>> across lat/lon, adjacent/close-by *geographical* grid elements will be
>> accessed and it's not clear to me that the above lat/lon orgainisation
>> will put elements separated by a short physical distance anywhere near
>> each other in the lat_lon array.
>>
>> I will be playing with and testing this over the coming days, but I
>> wanted to pick the brains of folks out there in advance.
>>
>> Thanks for any suggestions/advice,
>>
>> cheers,
>>
>> paulv
>>
>> p.s. Since the final code needs to be Fortran95, I set followups to
>> comp.lang.fortran
>
> Welcome to multiple key searching.
>
> The granddaddy technique goes by the name of kd-trees. As in K Dimensional
> trees. When k=2 they are called quad trees. When k=3, oct trees. When ...
>

> The problem is also called nearest neighbour searching with many
> geographers
> using natural neighbours as a variant. Also called associative searching or
> even content directed searching.
>
> This has a large literature with much of the terminology very graph
> theoretic.
> Triangulation is an important problem for many so there is much discussion
> of that. Regular spatial arrangements are called crystals which is a whole
> field in physics. Geographic databases are pretty common.
>
> If you like combinatorics there are a variety of space filling curves
> that can
> be used to keep things which are close in both (real) indices close in
> their
> single (referencing) index. The problem you are asking about.
>
> And here you thought it was going to be a simple answer to a simple
> question!
>
> Isn't this the sort of thing that outfits like NOAA are supposed to be
> experts in? Unfair question as you have to cross speciality boundaries
> and wade through arcane terminology. But seriously, there should be folks
> around there who know this sort of stuff.

There probably are, but there's much less red tape involved emailing this newsgroup than to broadcast email seeking help where I work. :o) But seriously, I will start asking around.

cheers,

paulv

p.s. And thanks for the info/suggestions above.

--
Paul van Delst Ride lots.
CIMSS @ NOAA/NCEP/EMC Eddy Merckx
Ph: (301)763-8000 x7748
Fax:(301)763-8545