Neo4j - Finding a doctor on the way to find coffee

I love coffee. I love finding new coffee shops and trying out great coffee roasters. Some of my favorites are Great Lakes Coffee, Stumptown Roasters in NYC's Ace Hotel, Red Rooster (my choice for home delivery) and my go-to local coffee shop, The Grounds. The Grounds is owned by friends and you have to love their hashtag #LoveCoffeeLovePeople.

You are probably asking what this has to do with Neo4j or anything in general. Go pour yourself a cup of coffee and then come back to see where this leads.

In the Neo4j Uber H3 blog post, you saw how we could use the hexaddress to find doctors within a radius, bounding box, polygon search or even along a line between locations. The line between locations feature got me thinking what if I could get use that feature and combine it with turn-by-turn directions to find a doctor along the route. You never know when you may have had too much coffee and need to find the closest doctor. Or maybe you have a lot of event data (IED explosions for example) and you want to see which ones have occurred along a proposed route.

Let's see if we can pull this off. I remembered that I had written some python code in conjunction with the Google Directions API. The directions API takes in a start address and an end address. For example:

```
https://maps.googleapis.com/maps/api/directions/json?origin='1700 Courtho
use Rd, Stafford, VA 22554'&destination='50 N. Stafford Complex, Center S
t Suite 107, Stafford, VA 22556'&key='yourapikey'
```

The API returns a JSON document which includes *routes* and *legs*. Each leg has a **start_location** with a lat and lng value and an **end_location** with a lat and lng value. Check the link for details on the JSON format.

In my python code, I make a call to the Directions API. I then parse the routes and associated legs (*that just sounds weird*) to get the start lat/lng and the end lat/lng for each leg. I can then pass the pair into my Neo4j procedure (*com.dfauth.h3.lineBetweenLocations*), get the results and output the providers that are along that line. Here's an example:

```
neoQuery = "CALL com.dfauth.h3.lineBetweenLocations(" + str(prevLat) + ",
" + str(prevLng) + "," + str(endlat) + "," +  str(endlng) +") yield nodes
 unwind nodes as locNode match (locNode)(t:TaxonomyCode) return distinct
locNode.Address1 + ' ' + locNode.CityName +', ' + locNode.StateName as lo
cationAddress, locNode.latitude as latitude, locNode.longitude as longitu
de, coalesce(p.BusinessName,p.FirstName + ' ' + p.LastName) as practiceNa
```

```
me, p.NPI as NPI order by locationAddress;" #       print(neoQuery)
  result = session.run(neoQuery)        for record in result:         pri
nt(record["locationAddress"] + " " + record["practiceName"])
```

When I tried this from my house to The Grounds, I got these results:

On the 15.5 minute drive, I would pass about 10 doctors. Now the NPI data isn't totally accurate but you can see what is available along the route.

I thought it was cool. My python code (minus the API keys) are on Github. Refill your coffee and thanks for reading.