Uber H3 : 육각형 계층의 인덱스 :: good

노트북:	IT관련_스크랩		
만든 날짜:	2019-04-08 오전 12:02	**수정한 날짜:**	2019-04-21 오후 12:01
태그:	GIS		
URL:	https://zzsza.github.io/data/2019/03/31/uber-h3/		

####
ㅇㅇ원문URL_ https://zzsza.github.io/data/2019/03/31/uber-h3/

<아래 스크랩 원문>

Uber H3 : 육각형 계층의 인덱스

31 Mar 2019 in Data on Geographic

- Uber의 그리드 시스템인 H3에 대한 글입니다

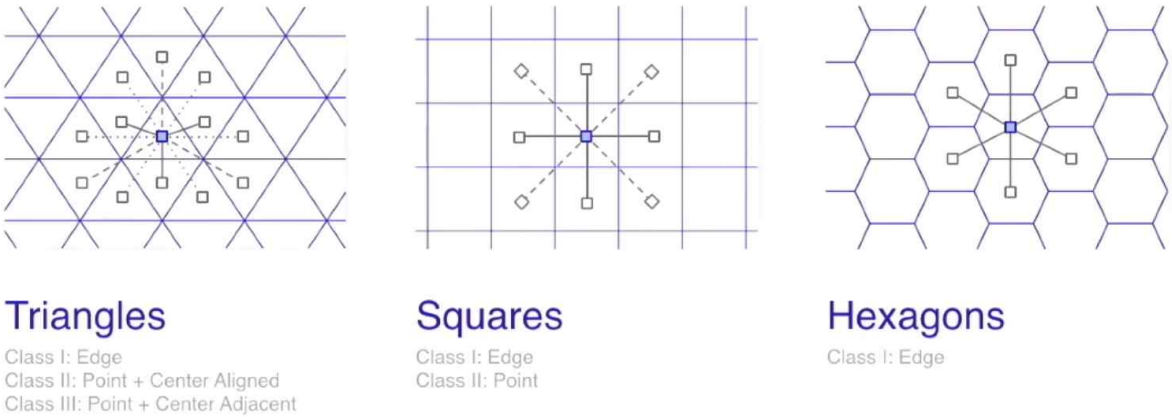
그리드 시스템

- Grid 시스템은 대용량 데이터를 분석하고, 지구의 영역을 구분 가능한 그리드셀로 분할할 때 중요
  - 한국 같은 경우 행정 구역 단위가 있지만(시군구동...) 이 단위는 행정을 위한 단위기 때문에 분석시 유용하지 않음
  - 강남구는 생각보다 길고, 큼. 강남역 왼쪽은 서초구
- 우버에선 ride price과 dispatch을 효율적으로 최적화하기 위해 그리드 시스템인 H3을 개발하고 오픈소스로 공개함

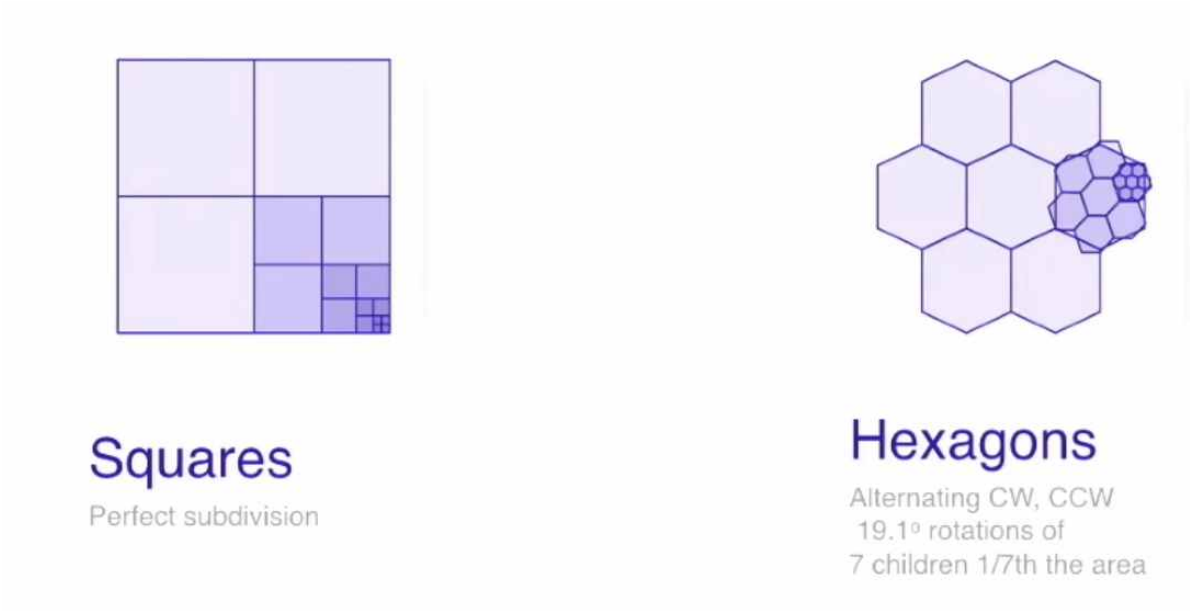
유튜브 영상

- H3 Youtube
- Surge Pricing 하며 겪은 이슈
  - 우버에서 처음엔 도시 단위로 오퍼레이션 했는데, Boundary Effect가 생김
    - Surge Cliffs에서 취소가 생김
  - Pantom Demand: 유령 수요(너무 넓은 지역)
  - 프랑스는 도시가 매우 복잡하고 잘게 쪼개져 있음
  - 도시보다 작은 단위가 필요함을 깨달음

- 왜 육각형(헥사곤)을 사용했는가?
  - Smooth gradients of demand를 구현할 수 있음
  - Clear center of demand
  - Dynamic neighborhoods
  - 아래 관점으로 여러 실험
    - Neighbor Traversal(이웃 순회)
      -

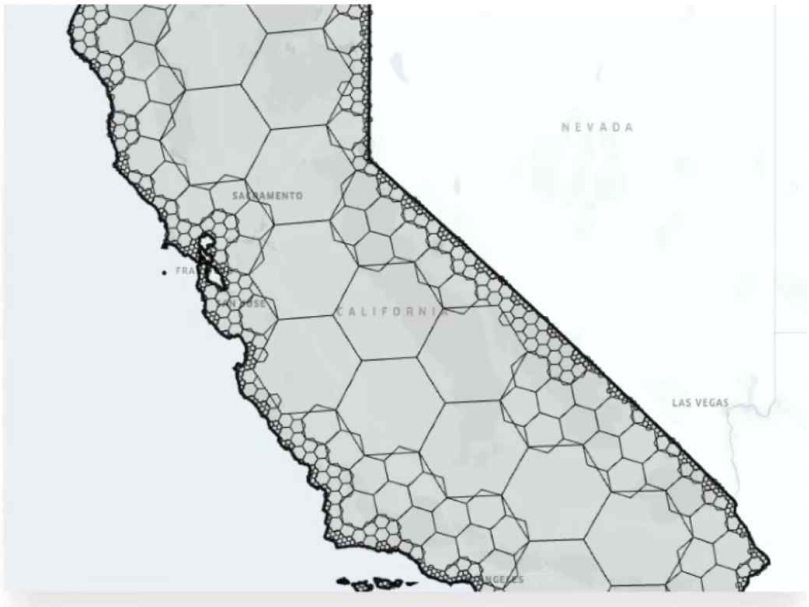


    - Subdivision(재조합)
      -



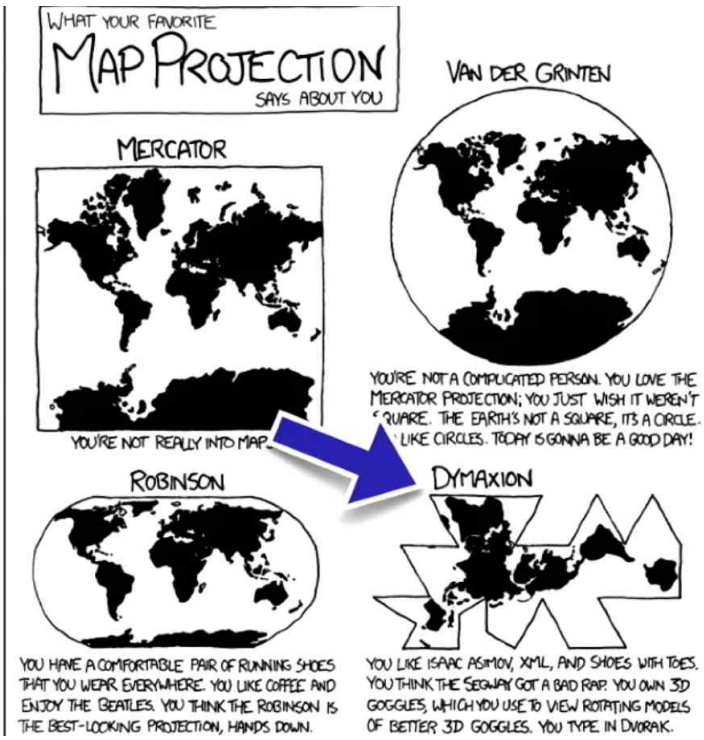
      - 사각형이 완벽하게 재조합이 되나, 헥사곤은 그렇진 않음. 우버는 완전 똑같을 필요는 없다고 함(약간의 에러를 감수)
      -

California

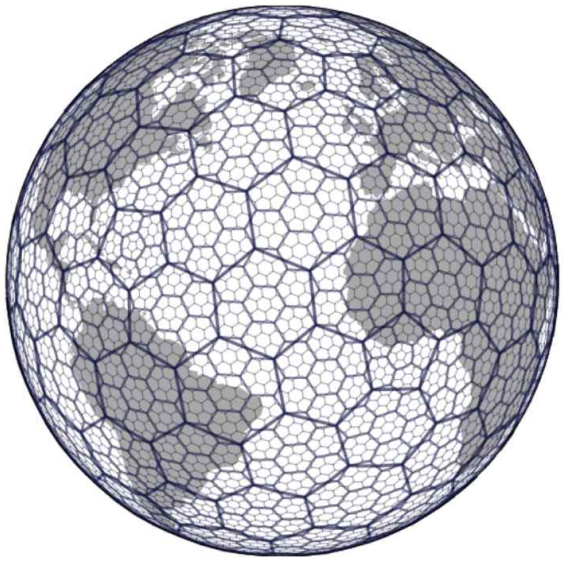


- Distortion(왜곡)
  - 헥사곤을 선택한 중요한 이유 중 하나
  - 지구는 sphere(구체)고, 평평하지 않음. 그러나 grid는 평평함. Map Projection을 통해 평평해짐
  -

Projections



-



- H3를 사용하면 지리 데이터를 분석해 여러 결정을 내릴 수 있음

그리드를 사용한 분석 사례

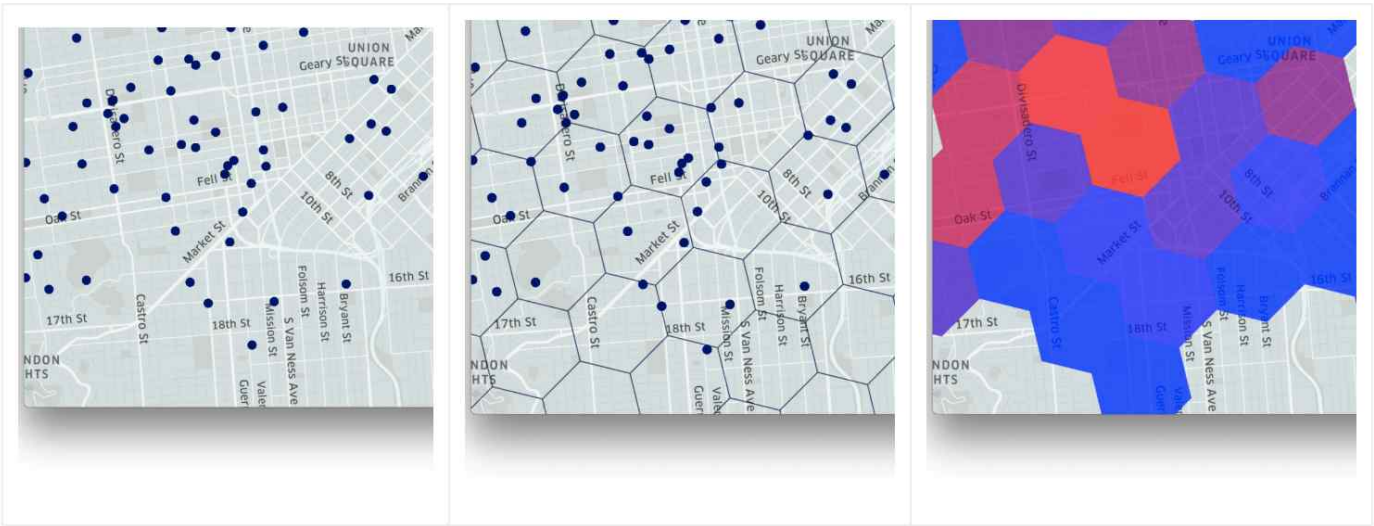


Figure 2. The maps above depict the process of bucketing points with H3: cars in a city; cars in hexagons; and hexagons shaded by number of cars.

- 매 순간 라이더가 라이딩을 요청하고, 운전자는 여행을 시작, 배고픈 사용자는 음식을 요청
  - 각 이벤트는 특정 위치에서 발생
- 이런 이벤트를 분석해 시장에 대해 더 잘 이해하고 최적화할 수 있음
- 도시의 특정 지역에서 공급보다 수요가 많아 가격을 조정하거나 특정 드라이버에게 가까운 거리에 승차 요청이 있다고 알릴 수 있음
- 도시 전체 데이터를 분석해야 하고, 미세한 단위로 수행되어야 함
- 헥사곤은 quantization error를 최소화함

H3

- Hexagonal global grid system의 장점 + hierarchical indexing 시스템을 결합하기 위해 H3을 만듬
- 지구상 3차원 위치에서 2차원 점으로 이동하려면 투영(Projection)이 필요
  - Mercator Projection이 유명한데, 크기 왜곡이 발생해 셀의 영역이 달라짐
    - 정사각형 그리드는 여러 계수가 필요로 함
  - 결국 지도 투영은 이십면체(Icosahedron)를 중심으로 하는 gnomonic projection을 사용함
    - 20면체는 다양한 방법으로 펼쳐져 2차원 지도를 생성할 수 있으나, H3는 전개하지 않고 20 면에 그리드를 배치해 geodesic discrete global grid 시스템을 만듬
- 육각형
  - 육각형은 중심점과 이웃 점 사이에 단 하나의 거리를 가지고 있음
  - 그라디언트에 대한 분석/스무딩을 단순화함
- H3 그리드
  - 122개의 기본 셀을 지구상에 배치하고 한 면에 10개의 셀을 배치해 구성함
  - 16개의 해상도(resolution)을 지원
-

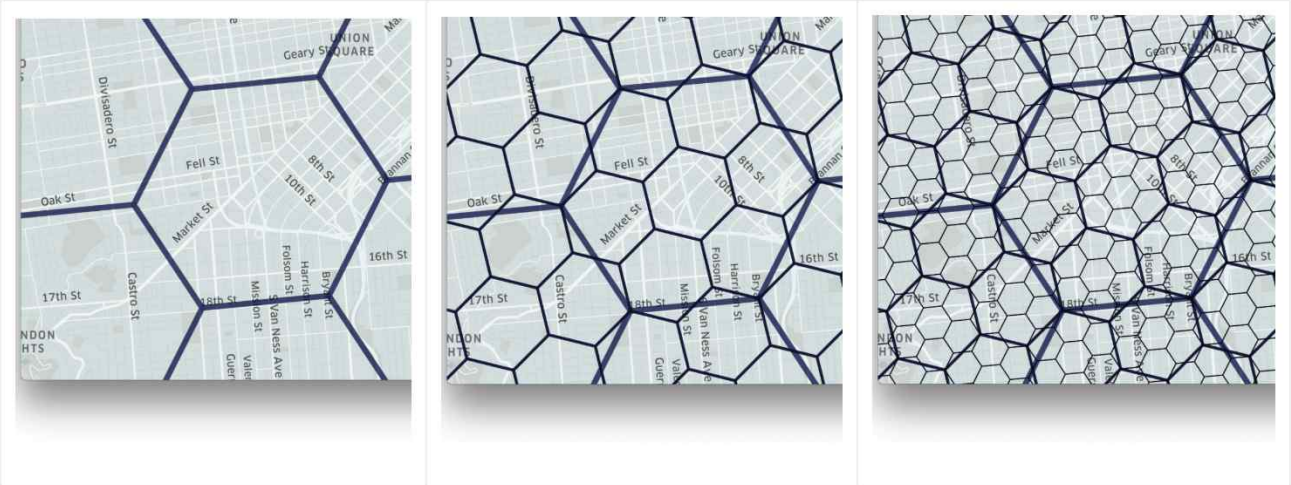


Figure 8. H3 enables the user to subdivide areas into smaller and smaller hexagons.

- 계층적 특성으로 인덱스의 해상도를 효율적으로 자르고 복구할 수 있음
- 육각형 색인은 64 비트 정수로 표현됨

H3 사용하기

- Python
- h3-py
- cc , make , cmake 가 깔려있는지 확인

```
which cc
/usr/bin/cc
which make
/usr/bin/make
which cmake
/usr/bin/cmake
```

- 설치

```
pip3 install h3
```

- geo_to_h3 함수
  - 위도, 경도, 해상도를 통해 h3 인덱스를 반환하는 함수
  - arg : lat 위도, lng 경도, hex resolution
-

```
from h3 import h3
h3_address = h3.geo_to_h3(37.3615593, -122.0553238, 5)
```

- h3_to_geo 함수
  - h3 인덱스를 통해 헥사곤의 중심점(lat, lng)을 반환하는 함수
-

```
hex_center_coordinates = h3.h3_to_geo(h3_address)
```

- h3_to_geo_boundary 함수
  - h3 인덱스를 통해 헥사곤의 boundary를 반환하는 함수
-

```
hex_boundary = h3.h3_to_geo_boundary(h3_address)
```

- k_ring_distance 함수
  - h3 인덱스를 통해 거리가 k 안에 있는 h3 인덱스를 반환하는 함수

-

```
h3.k_ring_distances(h3_address, 4)
```

- Folium과 결합해 Jupyter Notebook에서 사용하는 예시
- SQL에서 사용하고 싶으면, UDF를 만들어서 사용할 수 있음
- H3를 토대로 나온 데이터를 kepler.gl에서 시각화할 수 있음
- 관련 블로그 글 : Uber Kepler.gl : 지리 데이터 시각화 도구

Reference

- H3: Uber's Hexagonal Hierarchical Spatial Index
- H3 Github
- h3-py Github
- H3 Youtube

이 글이 도움이 되셨다면 추천 클릭을 부탁드립니다 :)

 Buy me a coffee