



统一交通时空数据格式: LibCity 中的原子文件

王静远

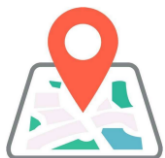
北京航空航天大学, 北京, 中国



- 为了统一表示不同类型的交通数据，LibCity 定义了五种原子文件，即交通数据中最小的五个信息单元。

文件名	信息	含义
xxx.geo	地理实体信息	描述地理空间中点、线、面三类实体的属性信息，如POI、路段、区域等。
xxx.usr	用户实体信息	描述参与运输的人员的属性，例如年龄、性别等。
xxx.rel	实体关系信息	描述实体之间的关系，例如路段之间的邻接关系。
xxx.dyna	交通状态信息	描述每个实体上交通系统的状态，例如每个路口的速度等。
xxx.ext	附加辅助信息	描述有助于交通预测的信息，例如天气、温度等。
config.json	配置信息	用于补充上表信息的说明。

.geo文件



点

< geo_id, type, coordinates, properties >



线



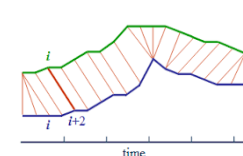
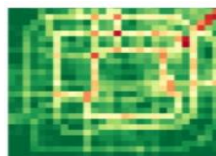
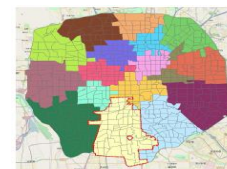
面

.usr文件



< user_id, properties >

.rel文件



< rel_id, type, origin_id, destination_id, properties >

.dyna

群体交通状态

人流/速度/路况

到达数量等

个体交通状态

个体出行轨迹

个体 OD 对等

关系的动态

连接关系的变化

OD 人流的变化等

.ext

描述有助于交通预测的信息，例如天气、温度等。

- 对于不同的流量预测任务，可能会使用不同的原子文件，一个数据集可能不包含所有六种原子文件。
- .geo、.usr、.rel、.dyna 和 .ext 的格式类似于由多列数据组成的 csv 文件。

geo_id	type	coordinates
773869	Point	[-118.32,34.15]
767541	Point	[-118.24,34.12]
...
769373	Point	[-118.32,34.10]

METR-LA.geo

rel_id	type	origin_id	destination_id	cost
0	geo	716328	716328	0.0
1	geo	716328	716331	4123.8
...
11752	geo	774207	774207	0.0

METR-LA.rel

dyna_id	type	time	entity_id	traffic_speed
0	state	2012-03-01T00:00:00Z	773869	64.375
1	state	2012-03-01T00:05:00Z	773869	62.667
...
7094303	state	2012-06-27T23:55:00Z	769373	61.778

METR-LA.dyna

Geo 表: 地理实体信息

<geo_id, type, coordinates, properties (multiple columns)>

- **geo_id**: 主键唯一地确定了一个地理实体（例如传感器、经纬度点、路段、区域等）。
- **type**: 地理实体的类型，一共有 “*Point*”, “*LineString*”, “*Polygon*” 三类。这三类与 Geojson 中的点、线和面的定义一致。
- **coordinates**: 由 Float 类型组成的数组或嵌套数组。使用 Geojson 的坐标格式描述地理实体的位置信息。
- **properties**: 描述地理实体的属性信息。如果有多个属性，您可以使用不同的列名来定义多列数据，例如 POI_name、POI_type。

Geo 表: 地理实体信息

<geo_id, type, coordinates, properties (multiple columns)>

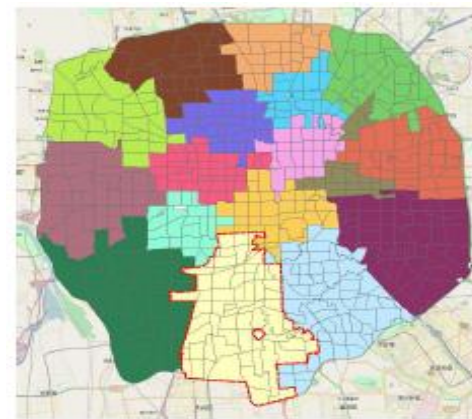
- Geojson 中的坐标表示格式: (经度在前, 纬度在后)
 - 点: [102.0, 0.5]
 - 线: [[102.0, 0.0], [103.0, 1.0], [104.0, 0.0], [105.0, 1.0]]
 - 面: [[[100.0, 0.0], [101.0, 0.0], [101.0, 1.0], [100.0, 1.0], [100.0, 0.0]]]



点



线



面

Geo 表: 地理实体信息

<geo_id, type, coordinates, properties (multiple columns)>

geo_id	type	coordinates
773869	Point	[-118.31828, 34.15497]
767541	Point	[-118.23799, 34.11620]
767542	Point	[-118.23818, 34.11640]
717447	Point	[-118.26772, 34.07248]
717446	Point	[-118.26572, 34.07142]

METR_LA.geo

geo_id	type	coordinates	venue_category_id	venue_category_name
0	Point	[-74.003,40.733]	4bf58dd8d48988d1e7931735	Music Venue
1	Point	[-73.975,40.758]	4bf58dd8d48988d176941735	Gym / Fitness Center
2	Point	[-74.003,40.652]	4bf58dd8d48988d1e4931735	Bowling Alley
3	Point	[-73.980,40.726]	4bf58dd8d48988d118941735	Bar
4	Point	[-73.967,40.756]	4bf58dd8d48988d11d941735	Bar

Foursqaure.geo

- Geojson 中的坐标表示格式: (经度在前, 纬度在后)
 - 点: [102.0, 0.5]
 - 线: [[102.0, 0.0], [103.0, 1.0], [104.0, 0.0], [105.0, 1.0]]
 - 面: [[[100.0, 0.0], [101.0, 0.0], [101.0, 1.0], [100.0, 1.0], [100.0, 0.0]]]

Usr 表: 用户实体信息

<usr_id, properties (multiple columns)>

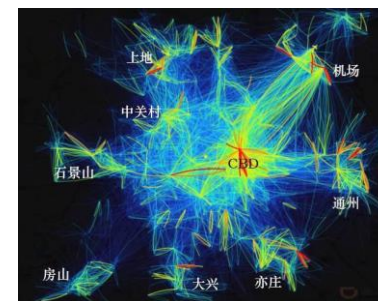
- **usr_id**: 主键唯一确定一个用户实体。
- **properties**: 描述用户实体的属性信息。如果有多个属性, 可以使用不同的列名来定义多列数据, 例如性别、出生日期。

usr_id
0
1
2
3
4

Foursqaure.usr



用户画像



出行偏好

Rel 表: 实体关系信息

<rel_id, type, origin_id, destination_id, properties (multiple columns)>

- **rel_id**: 主键唯一确定一对实体之间的关系。
- **type**: 关系的类型, 分为 “usr”, “geo” 两类。表明关系是基于地理实体还是用户实体。
- **origin_id**: 关系来源的 ID, 在 Geo 表或 Usr 表中。
- **destination_id**: 关系目标的 ID, 在 Geo 表或 Usr 表中。
- **properties**: 描述关系的属性信息。如果有多个属性, 可以使用不同的列名来定义多列数据。

Rel 表: 实体关系信息

<rel_id, type, origin_id, destination_id, properties (multiple columns)>

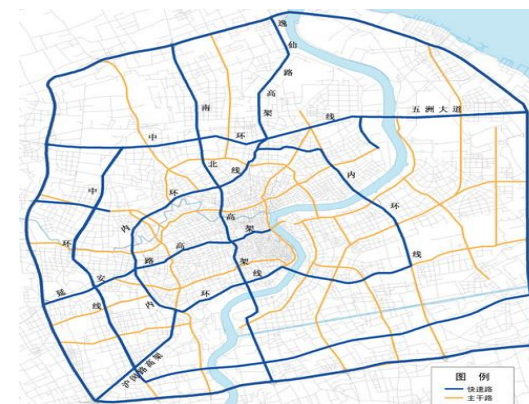
rel_id	type	origin_id	destination_id	cost	geo_id	type	coordinates
0	geo	716328	716328	0	773869	Point	[-118.31828, 34.15497]
1	geo	716328	716331	4123.8	767541	Point	[-118.23799, 34.11620]
2	geo	716328	716337	5179.6	767542	Point	[-118.23818, 34.11640]
3	geo	716328	716339	7245.5	717447	Point	[-118.26772, 34.07248]
4	geo	716328	716939	4785.1	717446	Point	[-118.26572, 34.07142]

METR_LA.rel

METR_LA.geo



社交网络



道路网络

Dyna 表: 交通状态信息

<dyna_id, type, time, entity_id(multiple columns), properties(multiple columns)>

- **dyna_id**: 主键唯一地确定了 Dyna 表中的一条记录。
- **type**: 状态类型, 分为个体交通状态 “*trajectory*” (对于基于轨迹的任务) 和群体交通状态 “*state*” (对于交通状态预测任务)。
- **time**: 时间信息, 使用 ISO-8601 标准中的日期和时间组合表示法, 如: 2020-12-07T02:59:46Z。
- **entity_id**: 描述状态是基于哪个实体产生的, 即地理实体或用户实体的 ID。
- **properties**: 描述记录的属性信息。如果有多个属性, 可以使用不同的列名来定义多列数据, 比如速度数据和流量数据。

Dyna 表: 交通状态信息

- 类型: 群体交通状态 “*state*”
- 细分为: 基于空间点 / 基于路段 / 基于区域 / 基于网格 / 基于 OD / 基于网格间 OD
- 格式: **<dyna_id, state, time, entity_id, properties>**
 - 为了便于使用, *entity_id* 列随着细分的群体交通状态的变化而变化。
- 表中的行应根据 *<entity_id>* 进行聚合, 具有相同 *<entity_id>* 的行按 *<time>* 排序。

Dyna 表: 交通状态信息

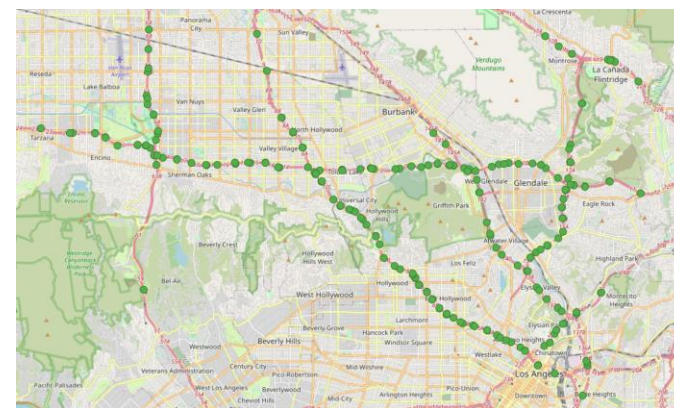
- 类型: 基于空间点 / 基于路段 / 基于区域的群体交通状态
- 格式为: **<dyna_id, state, time, entity_id, properties>**
- 对于可以进行一维编号的实体, 如传感器、路段、区域等。entity_id 为对应的实体 ID, 列名为 [entity_id], 文件后缀名为 .dyna。

dyna_id	type	time	entity_id	traffic_speed
0	state	2012-03-01T00:00:00Z	773869	64.375
1	state	2012-03-01T00:05:00Z	773869	62.66667
2	state	2012-03-01T00:10:00Z	773869	64
3	state	2012-03-01T00:15:00Z	773869	0
4	state	2012-03-01T00:20:00Z	773869	0

METR_LA.dyna

geo_id	type	coordinates
773869	Point	[-118.31828, 34.15497]
767541	Point	[-118.23799, 34.11620]
767542	Point	[-118.23818, 34.11640]
717447	Point	[-118.26772, 34.07248]
717446	Point	[-118.26572, 34.07142]

METR_LA.geo



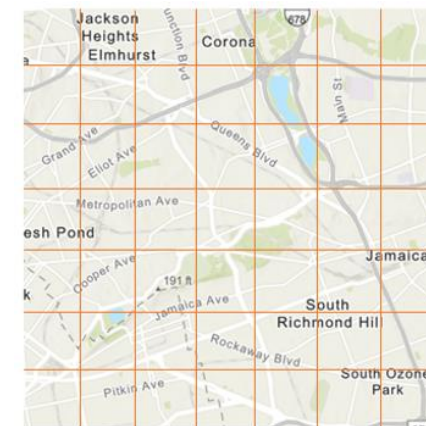
基于空间点

Dyna 表: 交通状态信息

- 类型: 基于网格的群体交通状态
- 格式: **<dyna_id, state, time, entity_id, properties>**
- 对于基于网格的交通数据, entity_id 扩展为 [row_id, column_id] 两列, 文件后缀为 .grid。

dyna_id	type	time	row_id	column_id	risk	geo_id	type	coordinates	row_id	column_id
0	state	2013-01-01T00:00:00Z	0	0	0	0	Polygon	[]	0	7
1	state	2013-01-01T01:00:00Z	0	0	0	1	Polygon	[]	0	8
2	state	2013-01-01T02:00:00Z	0	0	0	2	Polygon	[]	0	10
3	state	2013-01-01T03:00:00Z	0	0	0	3	Polygon	[]	0	11
4	state	2013-01-01T04:00:00Z	0	0	0	4	Polygon	[]	0	12

NYC_RISK.grid



NYC_RISK.geo

基于网格

Dyna 表: 交通状态信息

- 类型: 基于 OD 的群体交通状态
- 格式: **<dyna_id, state, time, entity_id, properties>**
- 对于基于 OD 的交通数据, entity_id 扩展为 **[origin_id, destination_id]** 两列, 文件后缀名为 .od。

dyna_id	type	time	origin_id	destination_id	flow	geo_id	type	coordinates
0	state	2012-03-01T00:00:00Z	0	1	345	0	Point	[-118.31828, 34.15497]
1	state	2012-03-01T00:05:00Z	0	2	277	1	Point	[-118.23799, 34.11620]
2	state	2012-03-01T00:10:00Z	0	3	64	2	Point	[-118.23818, 34.11640]
3	state	2012-03-01T00:15:00Z	0	4	0	3	Point	[-118.26772, 34.07248]
4	state	2012-03-01T00:20:00Z	1	2	0	4	Point	[-118.26572, 34.07142]

Data.od

geo_id	type	coordinates
0	Point	[-118.31828, 34.15497]
1	Point	[-118.23799, 34.11620]
2	Point	[-118.23818, 34.11640]
3	Point	[-118.26772, 34.07248]
4	Point	[-118.26572, 34.07142]

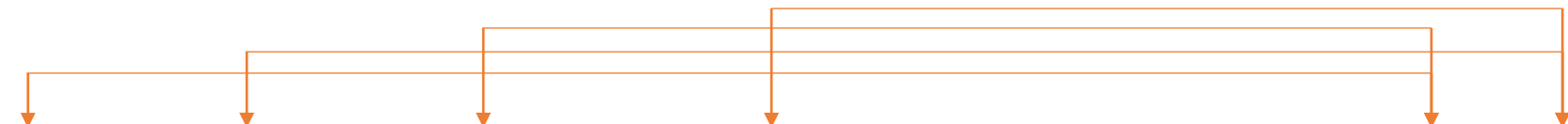
Data.geo



基于 OD

Dyna 表: 交通状态信息

- 类型: 基于网格间 OD 的群体交通状态
- 格式: **<dyna_id, state, time, entity_id, properties>**
- 基于网格间 OD 的流量数据, entity_id 扩展为四列 [origin_row_id, origin_column_id, destination_row_id, destination_column_id], 文件后缀名为 .gridod。



dyna_id	type	time	origin_row_id	origin_column_id	destination_row_id	destination_column_id	geo_id	type	coordinates	row_id	column_id
0	state	2013-01-01T00:00:00Z	0	1	2	1	0	Polygon	[]	0	7
1	state	2013-01-01T01:00:00Z	0	1	2	1	1	Polygon	[]	0	8
2	state	2013-01-01T02:00:00Z	0	1	2	1	2	Polygon	[]	0	10
3	state	2013-01-01T03:00:00Z	0	1	2	1	3	Polygon	[]	0	11
4	state	2013-01-01T04:00:00Z	0	1	2	1	4	Polygon	[]	0	12

Data.gridod

Data.geo

Dyna 表: 交通状态信息

- 类型: 个体交通状态 “*trajectory*”
- 细分为: GPS 轨迹 / 基于路段的轨迹 / POI 轨迹
- 格式: **<dyna_id, type, time, entity_id, (traj_id), properties>**
 - entity_id 列应该是用户实体 ID。
 - traj_id 列表示同一用户的多条轨迹的数量 (从 0 开始) , 如果用户只有一条轨迹, 该列可以为空。
- 表中的行按照 <entity_id> 进行聚合, <entity_id> 相同的行按 <traj_id> 排序, <traj_id> 相同的行按 <time> 排序。

Dyna 表: 交通状态信息

- 类型: GPS 轨迹
- 格式: **<dyna_id, type, time, entity_id, (traj_id), coordinates, properties>**
- <coordinates> 列是 GPS 点的经纬度。

dyna_id	type	time	entity_id	traj_id	coordinates	current_state	usr_id
0	trajectory	2014-08-03T18:29:00Z	810	0	"[104.115353,30.64392]"	1	810
1	trajectory	2014-08-03T18:29:40Z	810	0	"[104.113091,30.642129]"	1	811
...	812
21	trajectory	2014-08-03T18:53:23Z	810	0	"[104.076552,30.626844]"	0	813
22	trajectory	2014-08-03T18:13:00Z	810	1	"[104.106701,30.6916]"	1	814
...	814
241	trajectory	2014-08-03T18:16:00Z	11919	7	"[104.100816,30.706191]"	1	814

chengdu.dyna

chengdu.usr



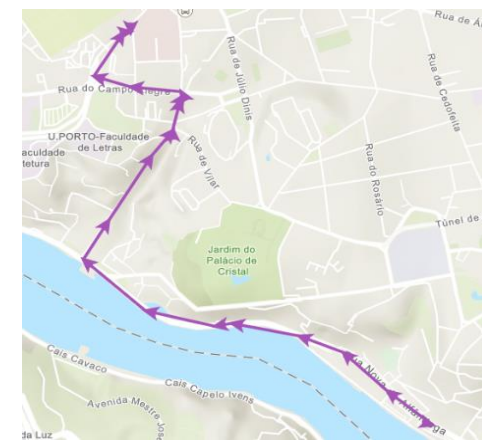
GPS 轨迹

Dyna 表: 交通状态信息

- 类型: 基于路段的轨迹
- 格式: **<dyna_id, type, time, entity_id, (traj_id), location, properties>**
- <location> 列的内容是 geo_id, 对应于 geo 表中的一个路段实体。

usr_id	dyna_id	type	time	entity_id	location	geo_id	type	coordinates
0	0	trajectory	2009-01-17T20:27:37Z	0	0	0	LineString	[[-122.7323, 47.8899], [-122.7321, 47.8903]]
1	1	trajectory	2009-01-17T20:27:38Z	0	1	1	LineString	[[-122.7321, 47.8903], [-122.7318, 47.8910]]
2	2	trajectory	2009-01-17T20:27:39Z	0	2	2	LineString	[[-122.7318, 47.8910], [-122.7313, 47.8921]]
3	3	trajectory	2009-01-17T20:27:40Z	0	2	3	LineString	[[-122.7313, 47.8921], [-122.7307, 47.8933]]
4	4	trajectory	2009-01-17T20:27:41Z	0	3	4	LineString	[[-122.7307, 47.8933], [-122.7302, 47.8944]]

Data.usr Data.dyna Data.geo



基于路段的轨迹

Dyna 表: 交通状态信息

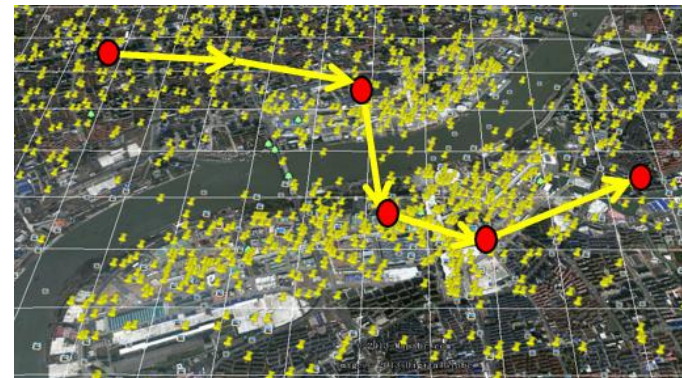
- 类型: POI 轨迹
- 格式: **<dyna_id, type, time, entity_id, (traj_id), location, properties>**
- <location> 列的内容是 geo_id, 对应于 geo表中一个 POI 空间点实体。

usr_id	dyna_id	type	time	entity_id	location	geo_id	type	coordinates
0	0	trajectory	2009-01-17T20:27:37Z	0	0	0	Point	[-122.7323, 47.8899]
1	1	trajectory	2009-01-17T20:27:38Z	0	1	1	Point	[-122.7321, 47.8903]
2	2	trajectory	2009-01-17T20:27:39Z	0	2	2	Point	[-122.7318, 47.8910]
3	3	trajectory	2009-01-17T20:27:40Z	0	2	3	Point	[-122.7313, 47.8921]
4	4	trajectory	2009-01-17T20:27:41Z	0	3	4	Point	[-122.7307, 47.8933]

Data.usr

Data.dyna

Data.geo



POI 轨迹

Ext 表: 附加辅助信息

<ext_id, time, properties (multiple columns)>

- **ext_id**: 主键唯一确定 Ext 表中的一条记录。
- **time**: 时间信息, 使用 ISO-8601 标准中的日期和时间组合表示法, 例如: 2020-12-07T02:59:46Z。
- **properties**: 描述记录的属性信息。

ext_id	time	temperature
0	2012-03-01T00:00:00Z	272.03
1	2012-03-01T00:05:00Z	271.46
2	2012-03-01T00:10:00Z	271.19
3	2012-03-01T00:15:00Z	271.07
4	2012-03-01T00:20:00Z	270.83

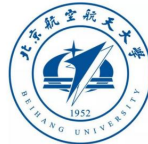
Data.ext

需要在配置文件中给出数据集中每一列的数据类型，有助于后续的数据处理。

Type	Description
geo_id	对应于 Geo 表中的 ID
usr_id	对应于 Usr 表中的 ID
rel_id	对应于 Rel 表中的 ID
time	符合 ISO-8601 标准的时间字符串
coordinate	符合 Geojson 格式坐标表示的字符串
num	实数
enum	枚举类字符串
other	其余的信息以字符串类型存储

- **配置文件用于补充描述上述五个表的信息。它以 json 格式存储，由 6 个 key 组成：geo、usr、rel、dyna、ext 和 info。**
 - **对于 geo, rel, dyna:**
 - 包含一个 include_types 键，其用一个数组来描述表中 <type> 列的值。之后，将每个 <type> 列的值作为一个键，描述该 <type> 类在表中包含哪些键以及键对应的数据类型。
 - **For usr, ext:**
 - 包含一个 properties 键，描述属性表中包含哪些键及其数据类型。
 - **For info:**
 - 包含数据集其他必要的统计信息。对于不同的交通预测任务，包含不同的内容，如路网结构大小。

Config 文件



```
"geo":{
  "including_types":[
    "Point"
  ],
  "Point":{
    "poi_name":"other",
  }
}

"usr":{
  "properties":{
    "user_type":"enum",
    "birth_year":"time",
    "gender":"enum"
  }
},

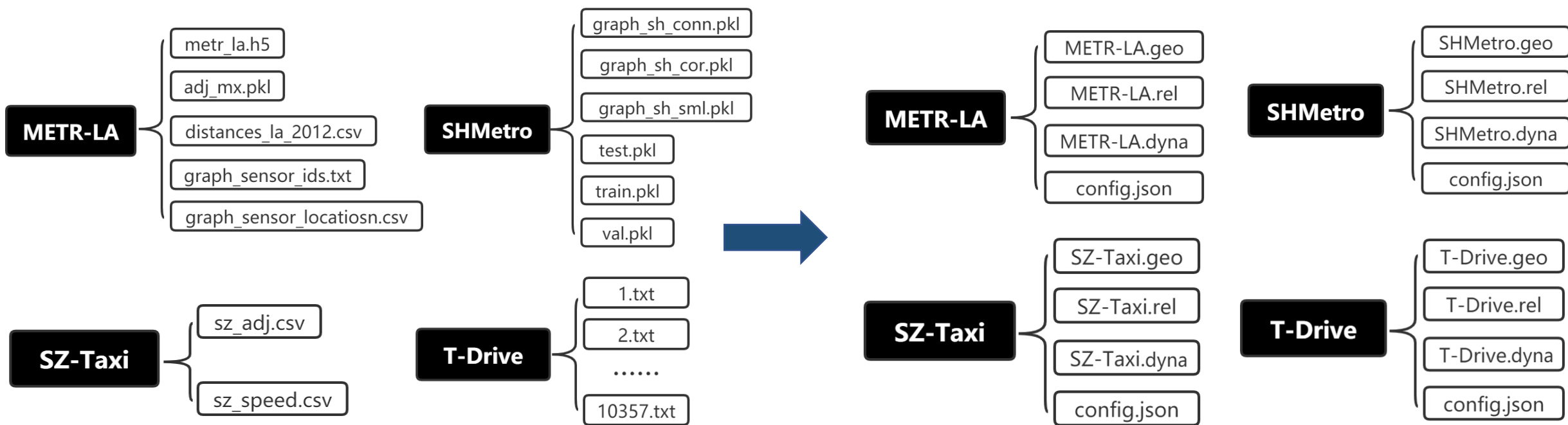
"rel":{
  "including_types":[
    "geo"
  ],
  "geo":{
    "link_weight":"num"
  }
},

"dyna":{
  "including_types":[
    "state"
  ],
  "state":{
    "entity_id":"geo_id",
    "traffic_speed":"num"
  }
},

"ext":{
  "properties":{
    "temperature":"num"
  }
},

"info": {
  "time_interval": 300,
}
```


- 我们基于上述原子文件格式已将 35 个交通数据集整合到交通预测开源算法库 **LibCity** 中，解决了交通数据集存储格式不一致的问题。



- LibCity 已将覆盖 11 个国家的 22 个城市的 35 个开源数据集转换为标准原子格式数据集。
- LibCity 还开源了原子文件转换脚本，供用户在转换自己的交通数据集时参考。



数据集全球覆盖情况

DATASET	#GEO	#REL	#USR	#DYNA	PLACE	DURATION	INTERVAL
METR-LA[19]	207	11,753		7,094,304	Los Angeles, USA	Mar. 1, 2012 - Jun. 27, 2012	5min
Los-loop[43]	207	42,849		7,094,304	Los Angeles, USA	Mar. 1, 2012 - Jun. 27, 2012	5min
SZ-Taxi[43]	156	24,336		464,256	Shenzhen, China	Jan. 1, 2015 - Jan. 31, 2015	15min
Loop Seattle[8, 9]	323	104,329		33,953,760	Greater Seattle Area, USA	over the entirety of 2015	5min
Q-Traffic[21]	45,148	63,422		264,386,688	Beijing, China	Apr. 1, 2017 - May 31, 2017	15min
PeMSD3[31]	358	547		9,382,464	California, USA	Sept. 1, 2018 - Nov. 30, 2018	5min
PeMSD4[13]	307	340		5,216,544	San Francisco Bay Area, USA	Jan. 1, 2018 - Feb. 28, 2018	5min
PEMSD7[31]	883	866		24,921,792	California, USA	Jul. 1, 2016 - Aug. 31, 2016	5min
PEMSD8[13]	170	277		3,035,520	San Bernardino Area, USA	Jul. 1, 2016 - Aug. 31, 2016	5min
PEMSD7(M)[38]	228	51,984		2,889,216	California, USA	weekdays of May and June, 2012	5min
PEMS-BA[19]	325	8,358		16,937,700	San Francisco Bay Area, USA	Jan. 1, 2017 - Jun. 30, 2017	5min
Beijing subway[41]	276	76,176		248,400	Beijing, China	Feb. 29, 2016 - Apr. 3, 2016	30min
M_dense[11]	30			525,600	Madrid, Spain	Jan. 1, 2018 - Dec. 21, 2019	60min
Rotterdam[14]	208			4,813,536	Rotterdam, Holland	135 days of 2018	2min
SHM-taxi[33]	228	22,244		1,021,008	Shanghai, China	Jul. 1, 2016 - Sep. 30, 2016	15min

数据集统计信息表



感谢聆听!

王静远

北京航空航天大学, 北京, 中国

jywang@buaa.edu.cn, <http://www.bigcity.ai>

