

BEAM Event Handler

Hayden Atchley

October 23, 2021

LOADING THE DATA

BEAM can be configured to output an `events.csv` file for each iteration. For now, I just picked one for testing:

```
eventsCSV = "R/eventHandler/test/39.events.csv"
```

I then loaded the .csv file, Unfortunately, `read_csv` didn't get all the data types right, so I had to set them manually:

```
coltypes <- paste0("cdcdccddd",
                   "cdddddddc",
                   "dcccddddd",
                   "cdcccdldcc",
                   "ccccddcddc",
                   "cccc"
                   )

fullEvents <- read_csv(eventsCSV, col_types = coltypes)
```

NOTE I am in the process of creating a list with each column name and type in case the column order changes or different columns are written. It is not yet finished however.

Then I selected a few columns of interest, and added a couple more:

```
eventCols <- c("person",
               "time",
               "type",
               "mode",
               "legMode",
               "vehicleType",
               "vehicle",
               "arrivalTime",
               "departureTime",
               "departTime",
               "length",
               "numPassengers",
               "actType",
               "personalVehicleAvailable"
               )
```

```

fullEvents %<>% relocate(all_of(eventCols))

events <- fullEvents %>% select(all_of(eventCols))
events %<>% mutate(
  travelTime = arrivalTime - departureTime,
  avgSpeed = length / travelTime
)
events %<>% arrange(person, time)

```

TODO: I will work on code to read in various information from the rhFleet file like shift duration, etc. That info can be important for stats like utilization (passengers/hour/vehicle). For now, I just took the actual values:

```

rhHours <- 80000/3600 #add code to read from the file
rhNum <- 12

```

I also loaded some stats from UTA On Demand's monthly reports (the data is available in quarterly reports, I created the csv):

```

UTAAD <- read_csv("R/eventHandler/test/UTAADpilotinfo.csv")
UTAAD %>%
  my_kbl(digits = 3, align = 'lrrr')

```

Month	Avg wkday ridership	Utilization	Avg wait time
DEC	224	1.33	9
JAN	334	2.00	11
FEB	392	2.31	12
MAR	316	1.88	11
APR	275	1.52	10
MAY	105	0.67	8
JUN	162	1.10	9
JUL	155	1.10	9
AUG	193	1.50	12
SEP	214	1.60	12
OCT	200	1.70	13
NOV	169	1.70	13

As a side note, I stored some person IDs that provide useful demonstrations:

```

personRH <- 1067049 #normal ridehail
personRH2 <- 730554 #rh2
personReplan <- 1060618 #rh replanning
personModfail <- 1023248 #mode isn't realized by end of simulation

```

ANALYSIS

Event Types

A good place to start is with the event types:

```
countEvents <- events %>%
  group_by(type) %>%
  summarize(n = n())
countEvents %>%
  my_kbl(align = 'lr')
```

type	n
actend	80348
actstart	80163
arrival	80163
departure	86162
LeavingParkingEvent	68019
ModeChoice	81239
ParkingEvent	68016
PathTraversal	536014
PersonEntersVehicle	158062
PersonLeavesVehicle	152060
Replanning	890
ReserveRideHail	1224

Many of these are self-explanatory, but here is what I’ve gathered so far:

- **actstart/actend** list the person, time, and type of event
- **arrival/departure** list the person, time, and “legmode”
 - **legmode** according to the BEAM documentation is the overall trip mode, either realized (**arrival**) or to be attempted (**departure**)
- **PersonEntersVehicle/PersonLeavesVehicle** lists the person, vehicle, and time
- **ReserveRideHail** just lists the person and time
- **ModeChoice** lists the person, time, mode desired, length (distance) of intended trip, and if a personal vehicle was available when the mode choice was made
- **Replanning** just lists the person and time; I believe this is triggered when the original mode choice didn’t work for whatever reason, so the agent chooses a different mode (in a subsequent **ModeChoice** event)
- **PathTraversal** is an event for vehicles rather than for agents. It lists time, mode, vehicle type, vehicle, departure time, arrival time, length (distance), and number of passengers. It also lists comma-separated IDs of the links traversed. It does not list the person IDs of the passengers, but may be useful for stats like average trip time. However, these events relate to *sub-legs* of the trip, rather than the overall trip. I believe when it comes to transit modes there is one event per stop, based on the travel time of each such event.

We can see the mode choice distribution, as well as the legmode of **arrival** events:

```

modechoice <- events %>%
  filter(type == "ModeChoice") %>%
  group_by(mode) %>%
  summarize(n = n())

legMode <- events %>%
  filter(!is.na(legMode),
         type == "arrival") %>%
  group_by(legMode) %>%
  summarise(n = n())

modeComparison <- left_join(modechoice,
  legMode,
  by = c("mode" = "legMode")
) %>%
`colnames<-`(c("mode", "modechoice", "legmode")) %>%
mutate(
  replans = modechoice - legmode,
  replan_pct = replans / modechoice
)
modeComparison %>%
  my_kbl(digits = 3, align = 'lrrrr')

```

mode	modechoice	legmode	replans	replan_pct
car	67271	67197	74	0.001
drive_transit	812	174	638	0.786
ride_hail	580	508	72	0.124
ride_hail_pooled	644	461	183	0.284
walk	10513	10404	109	0.010
walk_transit	1419	1419	0	0.000

Interestingly, the number of **Replanning** events is lower than the sum of the **replans** column above:

```

events %>%
  filter(type == "Replanning") %>%
  nrow()

```

```
## [1] 890
```

```
modeComparison$replans %>% sum()
```

```
## [1] 1076
```

As it turns out, some people have their day “cut short”: they choose a mode but don’t arrive at their next event by the end of the simulation:

```

events %>%
  filter(person == personModedefail) %>%
  tail(n = 10) %>%
  select(1:5)

```

```
## # A tibble: 10 x 5
##   person   time type                mode legMode
##   <chr>   <dbl> <chr>                <chr> <chr>
## 1 1023248 80837 PersonEntersVehicle <NA> <NA>
## 2 1023248 80837 PersonEntersVehicle <NA> <NA>
## 3 1023248 81392 PersonLeavesVehicle <NA> <NA>
## 4 1023248 81455 PersonLeavesVehicle <NA> <NA>
## 5 1023248 81455 arrival                <NA> car
## 6 1023248 81455 actstart                <NA> <NA>
## 7 1023248 82218 ModeChoice                car <NA>
## 8 1023248 82218 actend                <NA> <NA>
## 9 1023248 82218 departure                <NA> car
## 10 1023248 82218 PersonEntersVehicle <NA> <NA>
```

Ridehail Stats

We can look at some of the stats pertaining to ridehail vehicles:

```
rhPassengers <- events %>%
  filter(type == "PathTraversal",
         vehicleType == "micro"
        ) %>%
  select(numPassengers) %>%
  table() %>%
  as_tibble() %>%
  `colnames<-`(c("numPassengers", "n")) %>%
  mutate(pct = n / sum(n))
rhPassengers %>%
  my_kbl(digits = 2, align = 'lrr', caption = "Trips by Occupancy")
```

Trips by Occupancy

numPassengers	n	pct
0	918	0.48
1	972	0.50
2	39	0.02

```
totRiders <- sum(as.integer(rhPassengers$numPassengers) * rhPassengers$n)
totRiders
```

```
## [1] 1050
```

```
utilization <- totRiders / rhHours / rhNum
utilization
```

```
## [1] 3.9375
```

```
events %>%
  filter(type == "PathTraversal",
         vehicleType == "micro"
```

```

    ) %>%
select(numPassengers, travelTime) %>%
group_by(numPassengers) %>%
summarise(
  mean = mean(travelTime) / 60,
  median = median(travelTime) / 60,
  min = min(travelTime) / 60,
  max = max(travelTime) / 60
) %>%
my_kbl(digits = 2, align = 'lrrr', caption = "Travel Times by Occupancy (min)")

```

Travel Times by Occupancy (min)

numPassengers	mean	median	min	max
0	3.40	3.36	0.02	11.93
1	2.72	1.13	0.02	17.82
2	3.08	3.02	0.05	13.12

We can also look at the average time between reserving a ridehail and the next event, whether that's entering the ridehail or replanning and choosing a different mode (note that times are given in minutes):

```

rhTimes <- events %>%
  arrange(person, time) %>%
  mutate(
    rhReserveTime = ifelse(
      type == "ReserveRideHail" & person == lead(person),
      lead(time) - time,
      NA
    ),
    rhReserveOutcome = ifelse(
      type == "ReserveRideHail" & person == lead(person),
      lead(type),
      NA
    )
  ) %>%
  filter(!is.na(rhReserveTime))

rhTimes %>%
  group_by(rhReserveOutcome) %>%
  summarise(
    mean = mean(rhReserveTime) / 60,
    median = median(rhReserveTime) / 60,
    min = min(rhReserveTime) / 60,
    max = max(rhReserveTime) / 60
  ) %>%
  my_kbl(digits = 2, caption = "RideHail Reservation Outcomes", align = 'lrrr') %>%
  add_header_above(c(" " = 1, "Time until outcome (min)" = 4))

```

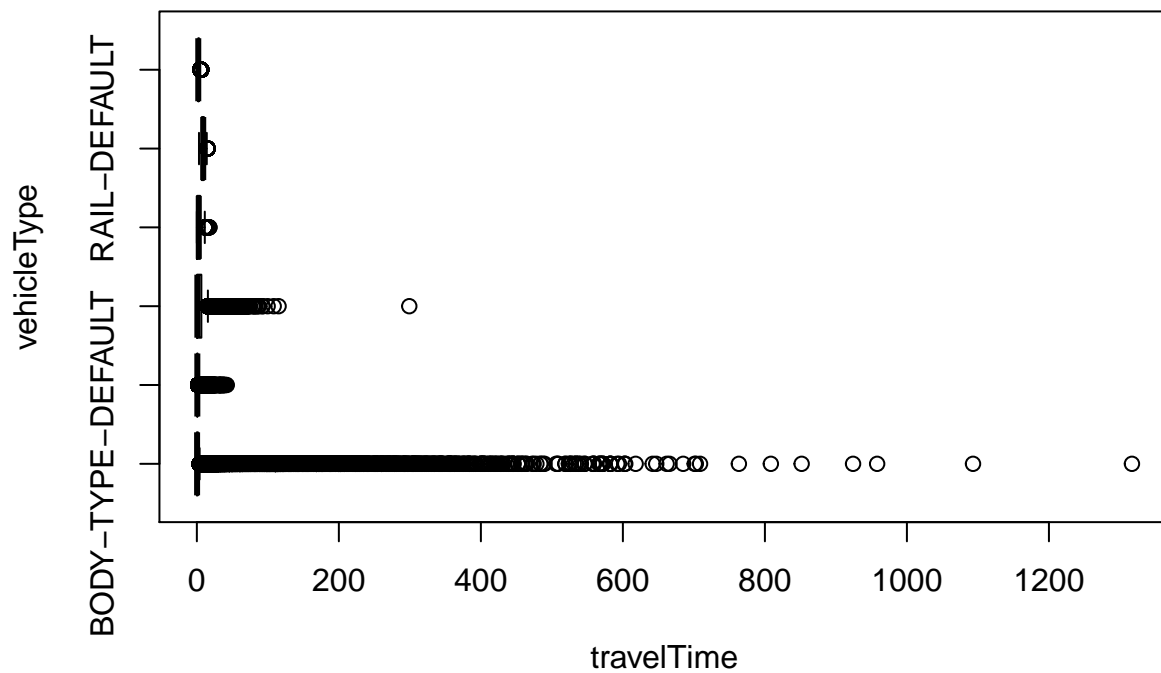
RideHail Reservation Outcomes

rhReserveOutcome	Time until outcome (min)			
	mean	median	min	max
PersonEntersVehicle	5.78	5.60	0.25	16.98
Replanning	2.14	2.27	0.08	3.50

Travel Times

We can also look at travel times (again in minutes):

```
events %>%
  filter(!is.na(travelTime)) %>%
  mutate(travelTime = travelTime / 60) %>%
  group_by(vehicleType) %>%
  boxplot(travelTime ~ vehicleType,
          data = .,
          horizontal = T)
```



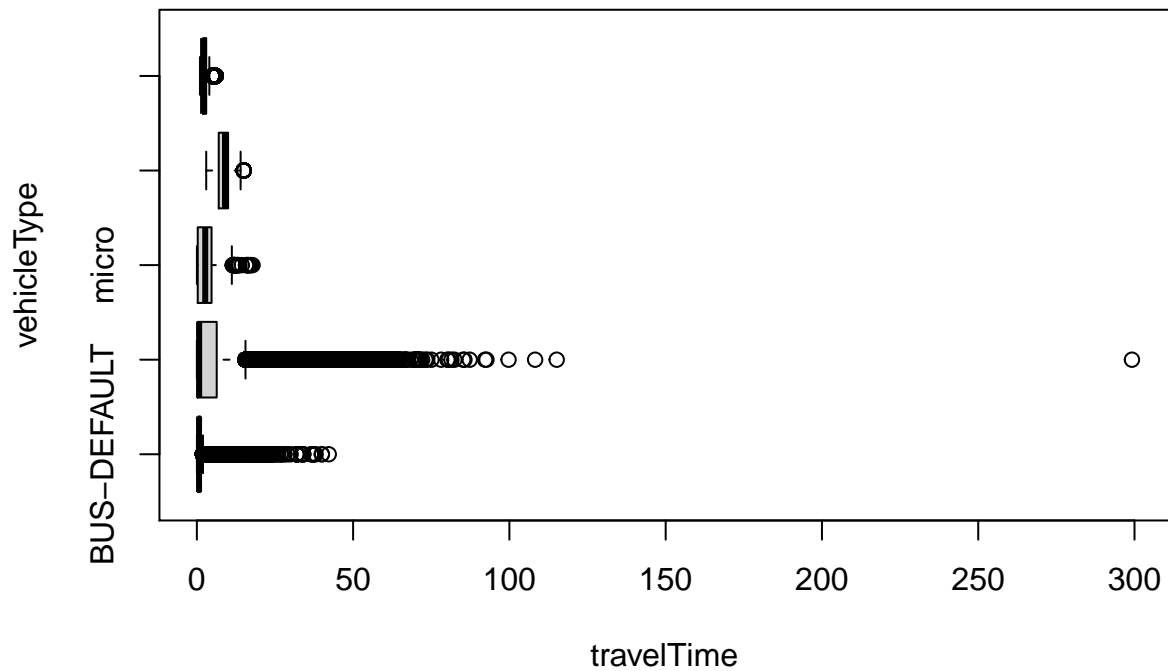
We may have some issues related to routing and/or since it seems people are willing to walk upwards of 20 hours. However, we can still look at subsets of the data:

```
events %>%
  filter(!is.na(travelTime),
```

```

    !grepl("BODY", vehicleType)
  ) %>%
mutate(travelTime = travelTime / 60) %>%
group_by(vehicleType) %>%
boxplot(travelTime ~ vehicleType,
        data = .,
        horizontal = T)

```



```

events %>%
  filter(!is.na(travelTime),
         !grepl("BODY", vehicleType),
         travelTime < (3 * 3600)
  ) %>%
mutate(travelTime = travelTime / 60) %>%
group_by(vehicleType) %>%
boxplot(travelTime ~ vehicleType,
        data = .,
        horizontal = T)

```