

# ISYE 3232 FINAL REPORT

Abba Otieno Ndomo

December 6, 2025

## Queueing Analysis of Intersection with Pedestrians

We analysed vehicle data from a signalised intersection with three approach streams and observable pedestrian crossings. For each car, we parsed arrival, stop and exit times. **We defined inter-arrival time as the difference between consecutive arrivals, service time as the time-in-the-intersection (exit minus stop), and waiting time as stop minus arrival.** The objective was to model the system as a single queue with a single server using QPLEX, then compare predicted waiting times to observed waits in order to assess the impact of pedestrians on vehicle delay.

```
# Helper functions
def _to_time(series: pd.Series) -> pd.Series:
    return pd.to_datetime(series.astype(str).str.strip(), format="%H:%M:%S", errors="coerce")

def _to_seconds(series: pd.Series) -> pd.Series:
    return pd.to_timedelta(series, errors="coerce").dt.total_seconds()

def _replace_nonpositive_with_min_positive(series: pd.Series) -> pd.Series:
    series = series.replace([np.inf, -np.inf], np.nan).dropna()
    positives = series[series > 0]
    if positives.empty:
        return series
    floor = positives.min()
    fixed = series.copy()
    fixed.loc[fixed <= 0] = floor
    return fixed

def load_vehicle_sheet(xl: Path, sheet_name: str) -> Dict[str, pd.Series]:
    df = pd.read_excel(xl, sheet_name=sheet_name)

    arrival = _to_time(df["Arrival Time"])
    stop = _to_time(df["Stop Sign Arrival"])
    exit_ = _to_time(df["Exit Time"])

    proc_col = _to_seconds(df["Processing time"])
    computed_proc = (exit_ - stop).dt.total_seconds()
    service = pd.Series(np.where(proc_col > 0, proc_col, computed_proc))
```

Figure 1: Snapshot of helper methods for cleaning our dataset

We cleaned the data by replacing any non-positive inter-arrival, entry, service or waiting times with the minimum positive value observed in that

series and removing missing values. We then formed empirical PMFs by binning times into **5-second intervals, capping support at 30 bins (0–150s)** to keep the QPLEX state space tractable. These PMFs were supplied to the `StandardMultiserver` model with `number_of_servers = 1`. After a 200-step burn-in, we sampled over 400 steps, truncating the state space at 400 states.

**For the combined vehicle stream, the mean inter-arrival time was approximately  $E[A] \approx 7.3\text{ s}$ , and the mean service time  $E[S] \approx 15.0\text{ s}$ , giving an arrival rate  $\lambda \approx 0.137 \text{ cars/s}$  and traffic intensity  $\rho = \lambda E[S] \approx 2.05 > 1$ .** QPLEX returned a truncated steady-state distribution with  $E[N] \approx 4066$  and  $E[Q] \approx 4065$ , and Little's law implied  $W_q \approx 29,700\text{ s}$  ( $\sim 8.25\text{ h}$ ). **These values are not physically realistic, since observed waits are on the order of tens of seconds.** Instead, they indicate that, under the assignment's **one-queue/one-server** framing, the system is **theoretically unstable:** pedestrian-inflated service times combined with observed arrivals exceed single-server capacity. Thus, QPLEX qualitatively signals **that pedestrians significantly reduce effective capacity**, but realistic quantitative predictions would require multiple servers (e.g. **lanes or phases**), **reduced service times, or lower arrival rates.**

### **Snapshot of the dataset (vehicles, combined across streams)**

- Inter-arrival (n=297): mean  $\sim 5.24\text{s}$ , median  $\sim 3\text{s}$ , std  $\sim 7.32\text{s}$ .
- Service (n=298): mean  $\sim 14.9\text{s}$ , median  $\sim 13\text{s}$ , std  $\sim 10.1\text{s}$ .
- Waiting (n=298): mean  $\sim 96.3\text{s}$ , median  $\sim 81\text{s}$ , std  $\sim 87.1\text{s}$ .
- Per-sheet stats are in the earlier cells ( `per_sheet` ).

Figure 2: Dataset snapshot