2021 DOE Vehicle Technologies Office
Annual Merit Review

Computation of Metropolitan-Scale, Quasi-Dynamic Traffic Assignment Models Using High Performance Computing

Jane Macfarlane
Lawrence Berkeley National Laboratory
June 22, 2021

Project ID: eems087

BERKELEY LAB

CITY OF SAN JOSE
CAPITAL OF SILICON VALLEY

Overview

TIMELINE

- Start: September 2019
- End: May 2021
- 100 % complete

BUDGET

- Total project funding
- $300k / 1 year

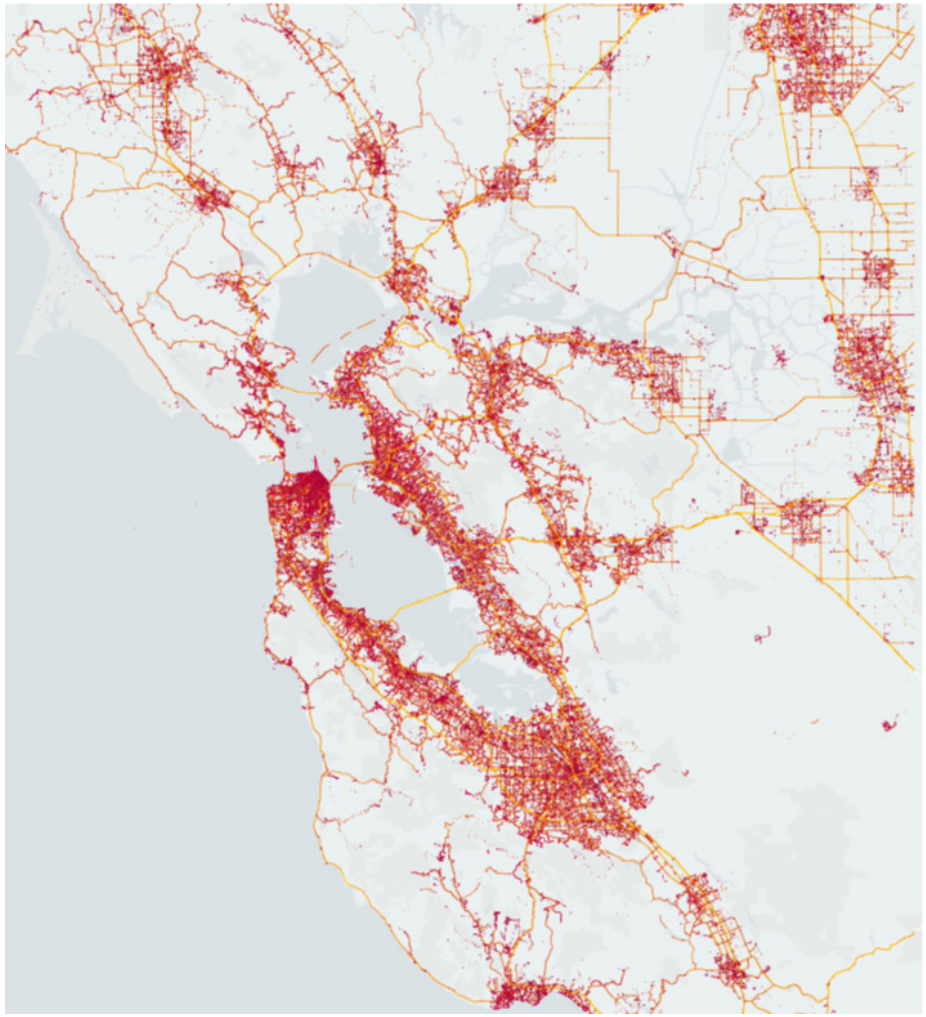
PARTNERS

- City of San Jose

BARRIERS

- Traffic assignment has traditionally focused only on user equilibrium travel time, not energy use at the system level.
- Static traffic assignment is not realistic as it cannot capture time-dependent traffic demand and ensure the continuity of traffic flows.
- Traditional dynamic traffic assignment models have limitations in terms of scalability and computational efficiency.

Relevance and Project Objectives



Overall Goal:

- Use of high-performance computing to address the compute load of traffic assignment methodologies and optimize for energy use in large scale networks

Objective:

- Develop traffic assignment computational solutions that provide more realistic results for evaluating traffic in large scale urban areas by representing demand dynamics over significantly smaller time steps than traditional approaches.
- Develop traffic assignment computational solutions that run in significantly reduced time frames by using parallel algorithms on high- performance computing.
- Introduce additional optimization objectives to the standard user travel time evaluations.

Impact:

- Enable computationally-efficient solutions for large-scale transportation planning and operation decisions that consider energy and travel time.

Approach

- Develop a metropolitan-scale, quasi-dynamic, parallel traffic assignment model that runs in significantly shorter compute times than alternative models

- Investigate time-based and energy-based optimization with a user focus and a system level focus

- Compare network metrics, energy, and mobility metrics across three optimization objective functions:
  - ✓ User equilibrium travel time (UET) based,
  - ✓ System optimal travel time (SOT) based, and
  - ✓ System optimal fuel use (SOF)

Consider Three Optimizations

$$T(q_i)_{UE} = t_0(1 + \alpha * (q_i/c_i)^{\beta}) \quad (1)$$

$$T(q_i)_{SO} = T(q_i)_{UE} + \frac{\partial T(q_i)_{UE}}{\partial q_i} * q_i \quad (2)$$

$$F(q_i)_{SO} = f_i + \frac{\partial f_i}{\partial q_i} * q_i \quad (3)$$

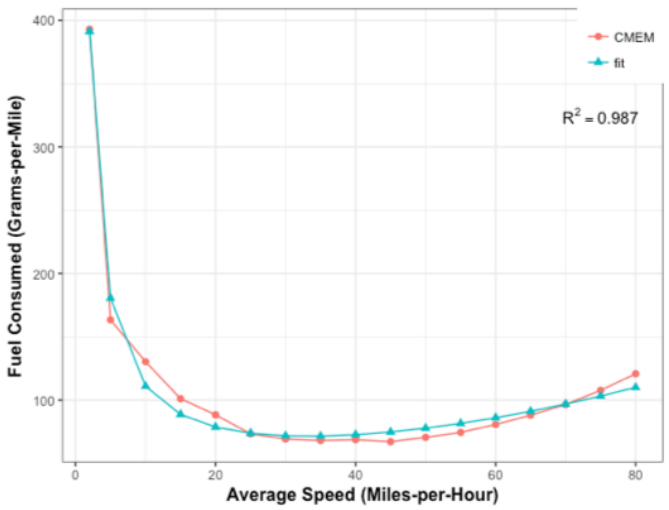
where: t_0 is the free flow link travel time, q_i is link flow, c_i is link capacity, v_{fi} is the free flow link speed and α and β are BPR coefficients 0.15 and 4 respectively. For system optimal fuel, f_i is the fuel consumption in grams/veh specified as

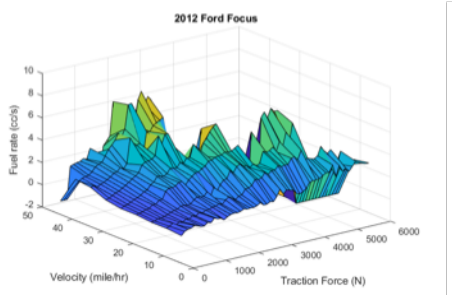
$$f_i = l_i(A + \frac{B}{v_i} + Cv_i^2)$$

where l_i is the length of link, A, B, C are constants estimated from real world drive cycles $A = -6.54170583e-03, B = 1.90215003, C = 1.58863662e-05$, and v_i is the congested speed defined as $v_i = \frac{v_{fi}}{1+\alpha*(q_i/c_i)^{\beta}}$.

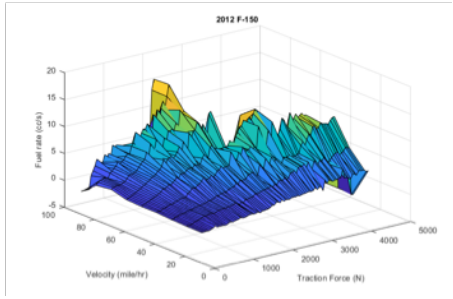
T_{UE} – User Equilibrium Travel Time
T_{SO} – System Optimal Travel Time
F_{SO} – System Optimal Fuel

Data Driven Energy Model

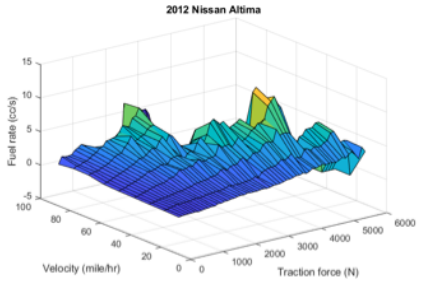




Compact Vehicle

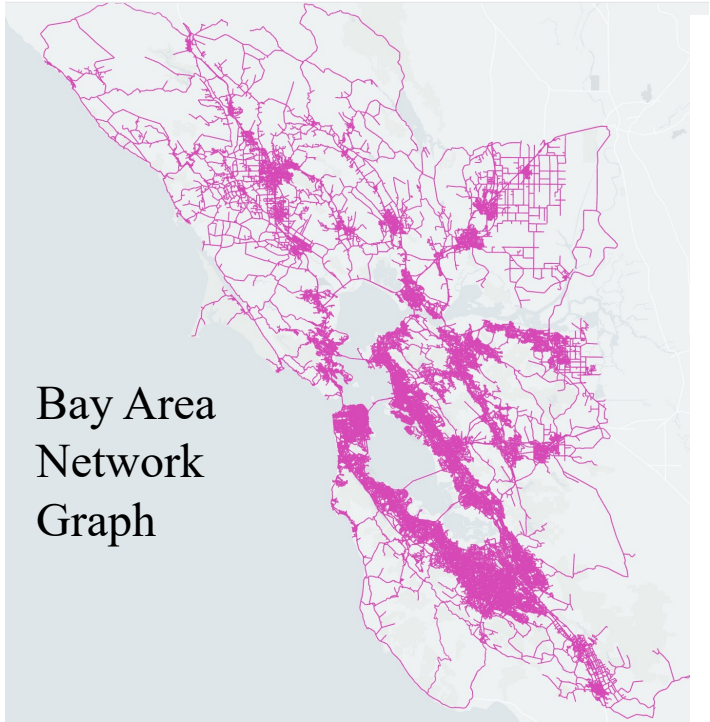


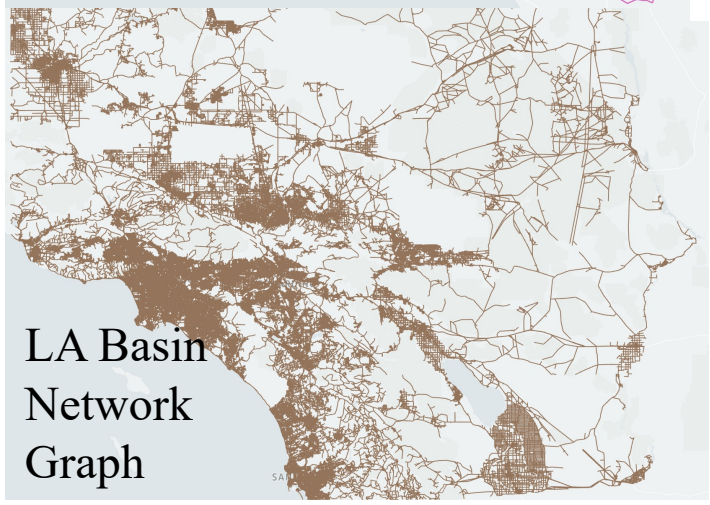
Mid Size Vehicle



MD Truck

Quasi Dynamic Traffic Assignment (QDTA) Algorithms





Algorithm 1: Quasi-dynamic traffic assignment

Data: Network graph $G(\mathcal{V}, \mathcal{A})$
Time step length T
Total time step counts N
Original travel demand of all time steps $\boldsymbol{d}^o = \{\boldsymbol{d}^o(t_i)\}$, with $i \in \{0, ..., N-1\}$

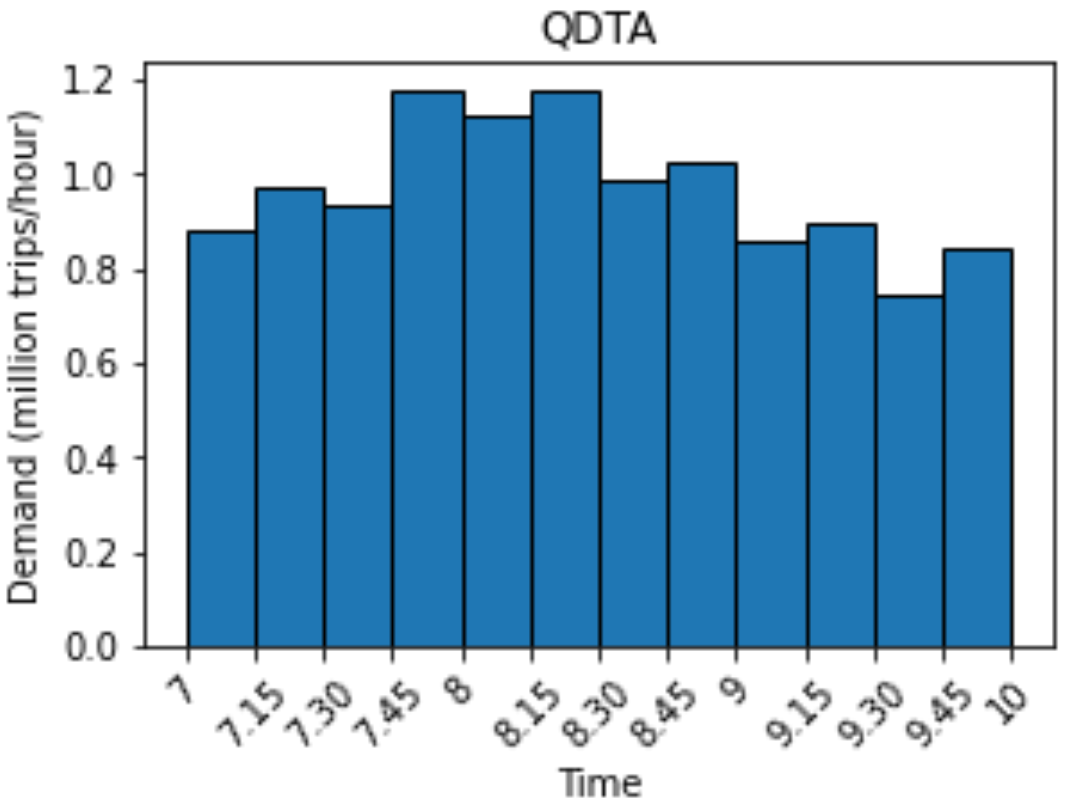
Initialize $\boldsymbol{d}^r(t_0) =$ empty nested associative array ; `// No residual demand in the first time step`

for $i = 0;\ i < N;\ i \mathrel{+}= 1$; `// Sequential discrete time step simulation`
do
  $\boldsymbol{d}(t_i) = \boldsymbol{d}^o(t_i) + \boldsymbol{d}^r(t_i)$; `// Get original and residual demand`
  $\boldsymbol{h}^{STA}(t_i) =$ Traffic_assignment$(G,\ T,\ \boldsymbol{d}(t_i))$; `// Path flow assignment using STA approximation`
  $\boldsymbol{h}(t_i), \boldsymbol{d}^r(t_{i+1}) =$ Residual_demand$(G,\ T,\ \boldsymbol{h}^{STA}(t_i))$; `// Truncate path flows and get residual demand based on time step length`
end
Result: $\boldsymbol{h} = \boldsymbol{h}(t_i)$, with $i \in 0, ..., N-1$; `// Path flow for all time steps`

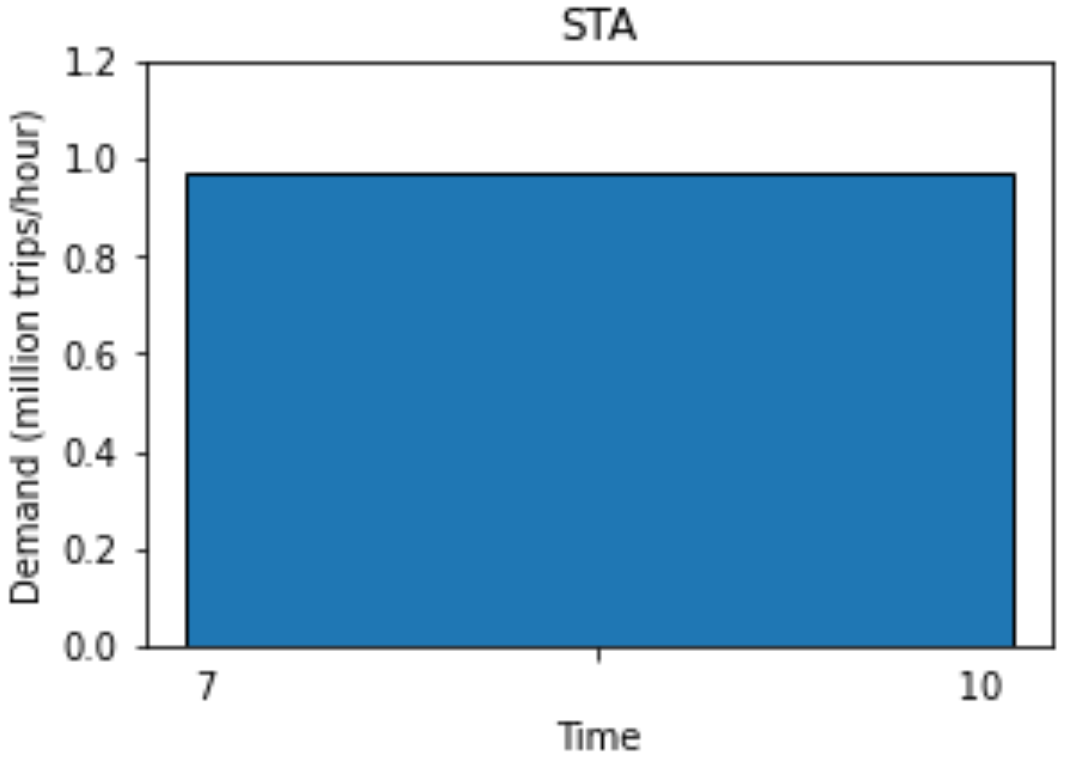
STA-based flow solution is first obtained using the Frank-Wolfe's algorithm (Algorithm 2, Algorithm 3). Path truncation is performed in each iterative step of the Frank-Wolfe's algorithm (Algorithm 3). A last round of route truncation to obtain a more accurate flow assignment and residual demand to be carried over to the next time step (Algorithm 4).

Detail of algorithms 2 -4 included in technical backup.

Demand Profile Comparison



Quasi Dynamic Traffic Assignment
15 minute time step

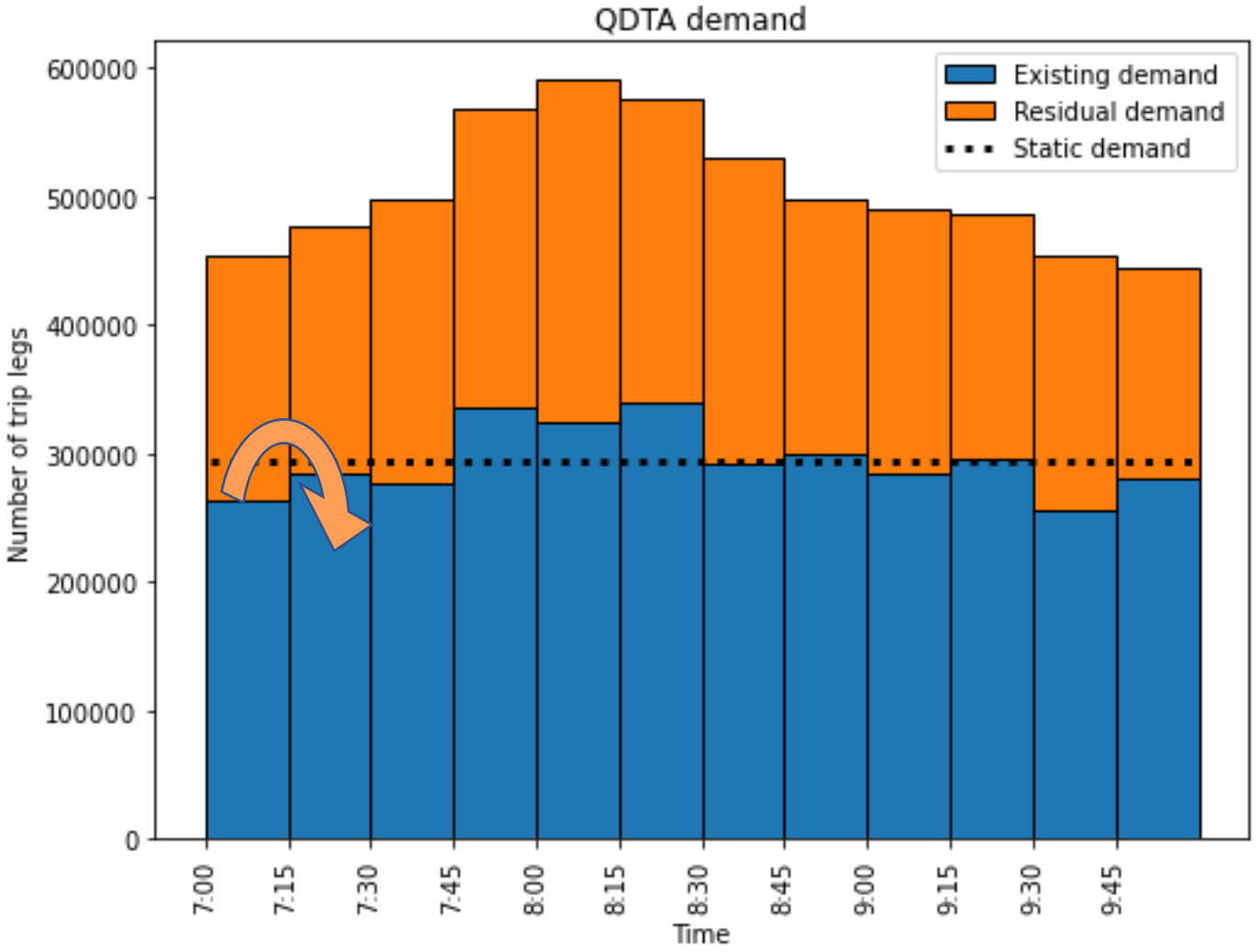


Static Traffic Assignment
3 hour time step

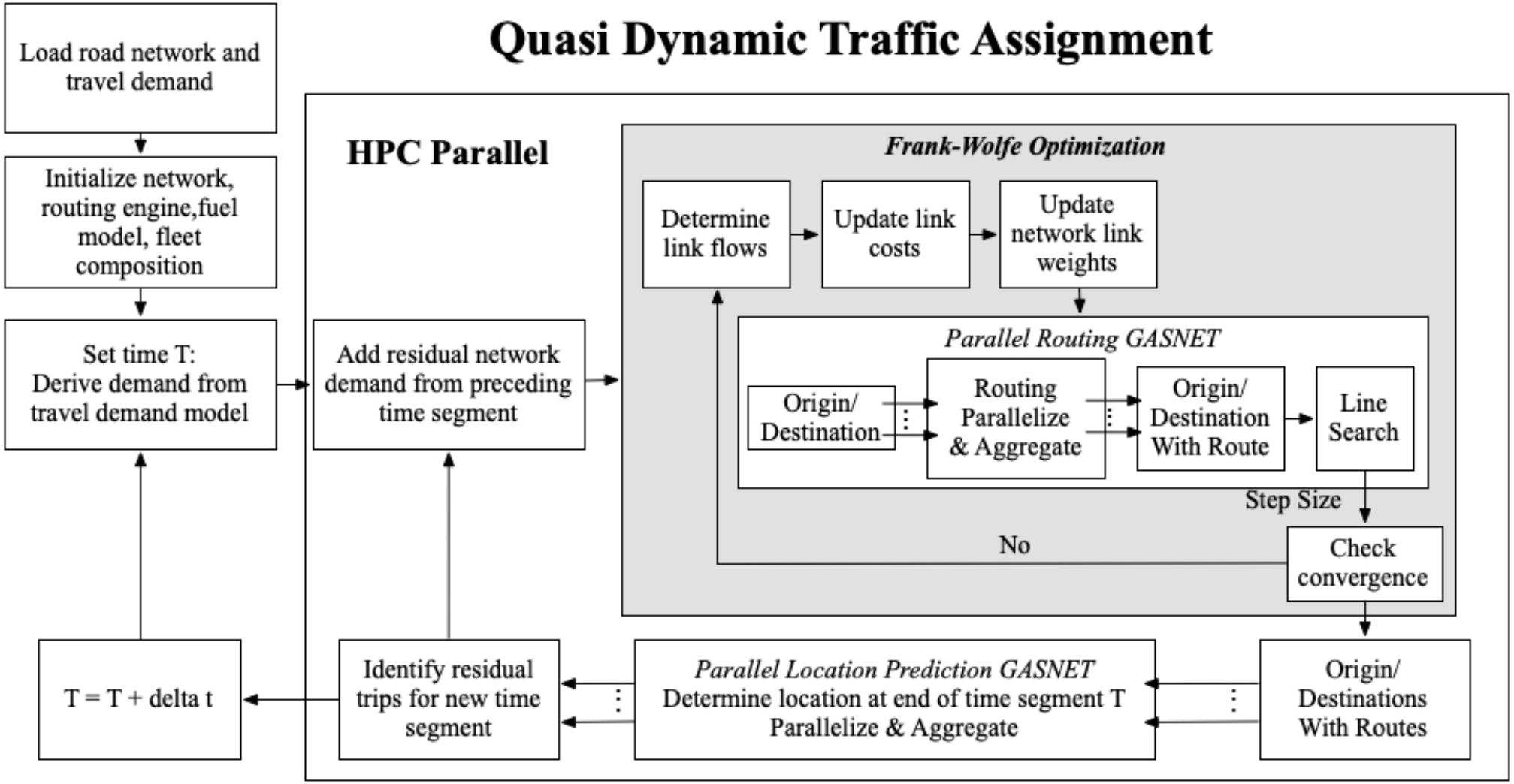
Key Element of QDTA : Residual Demand

The QDTA approach improves on STA in three key areas:

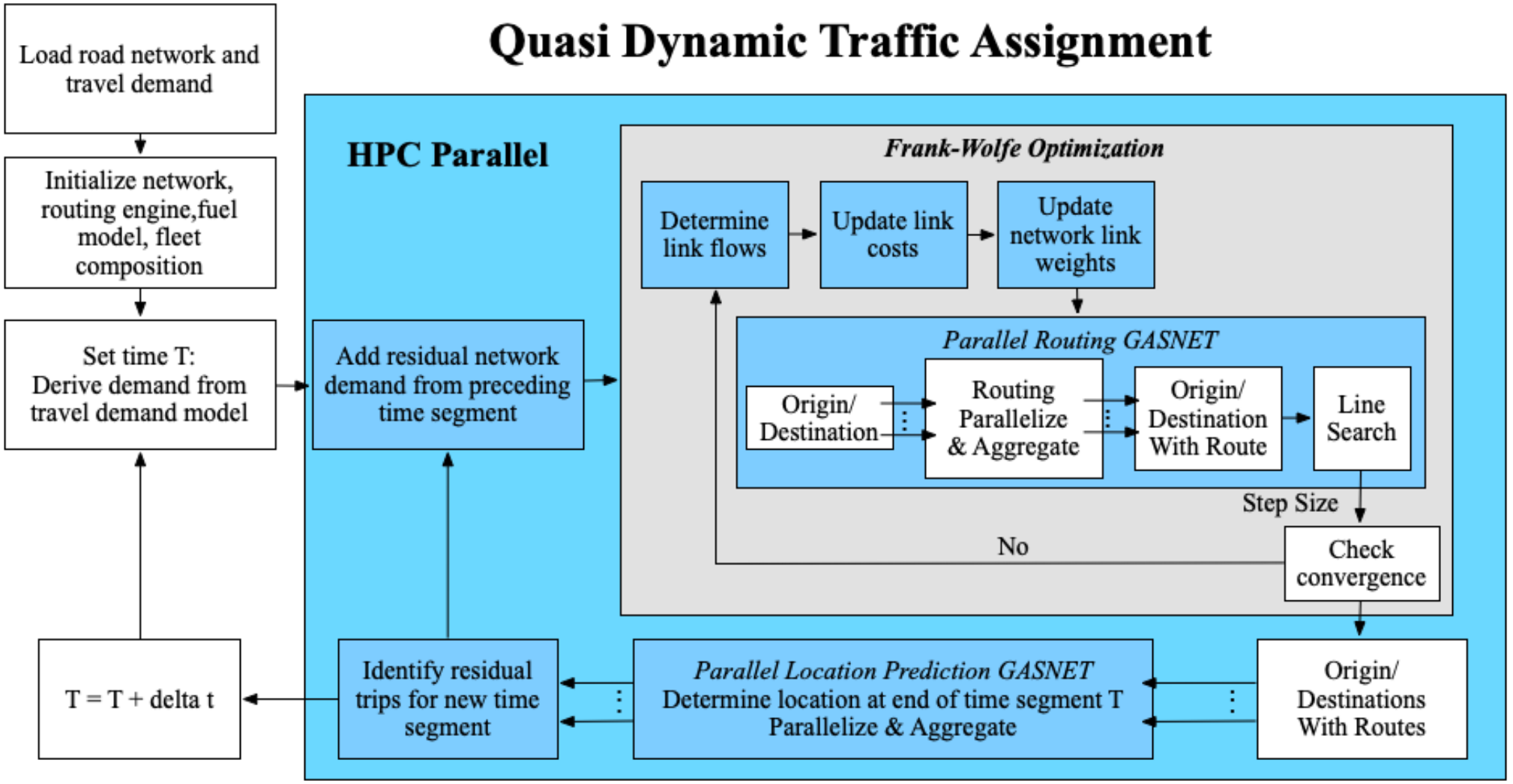
1. QDTA captures the temporal variation in the demand
2. QDTA ensures only the active part of the demand is assigned in a time interval
3. QDTA ensures residual demand is carried over across multiple time steps



Find Opportunities to Parallelize

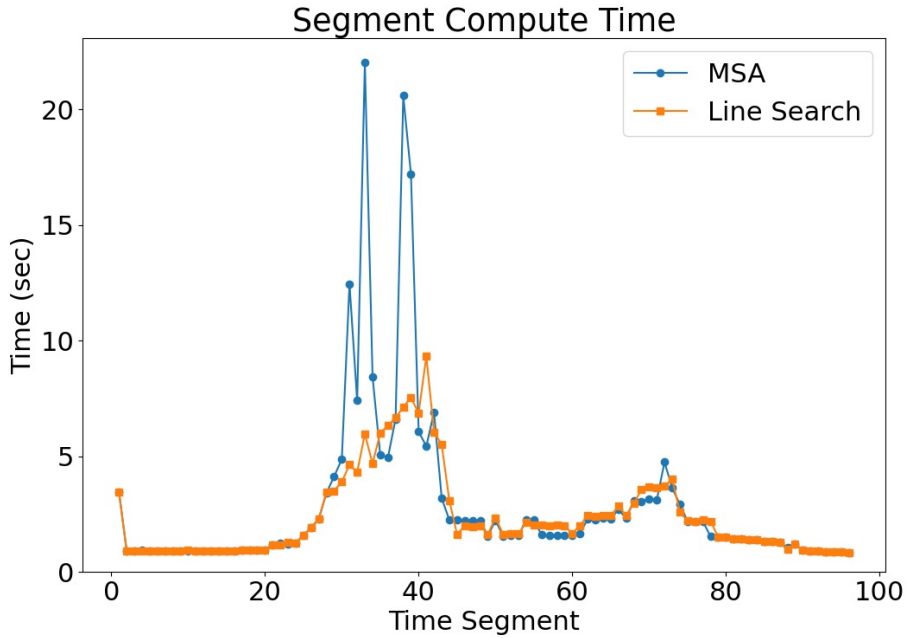


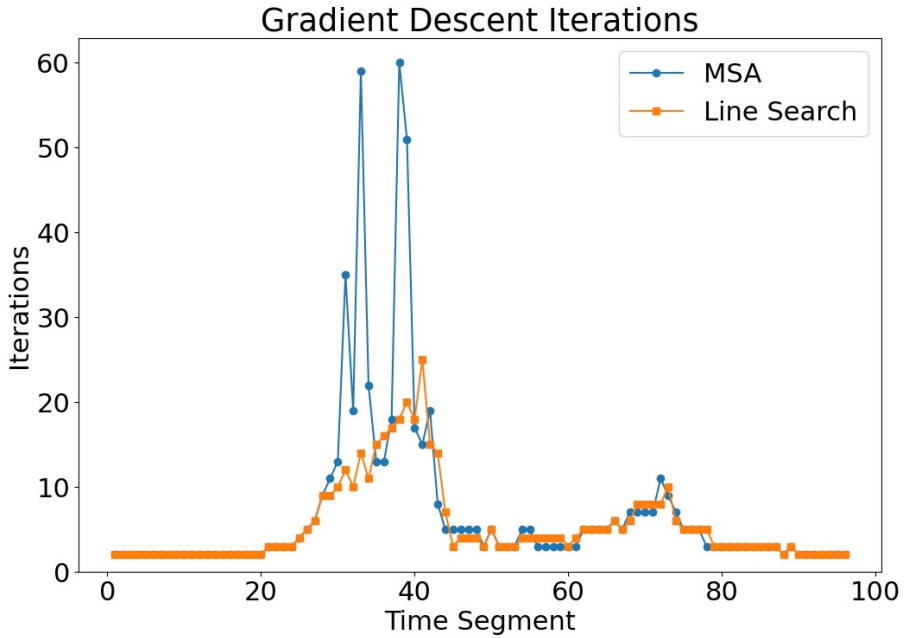
Find Opportunities to Parallelize



Quasi-Newton Line Search vs. Method of Successive Averages (MSA)

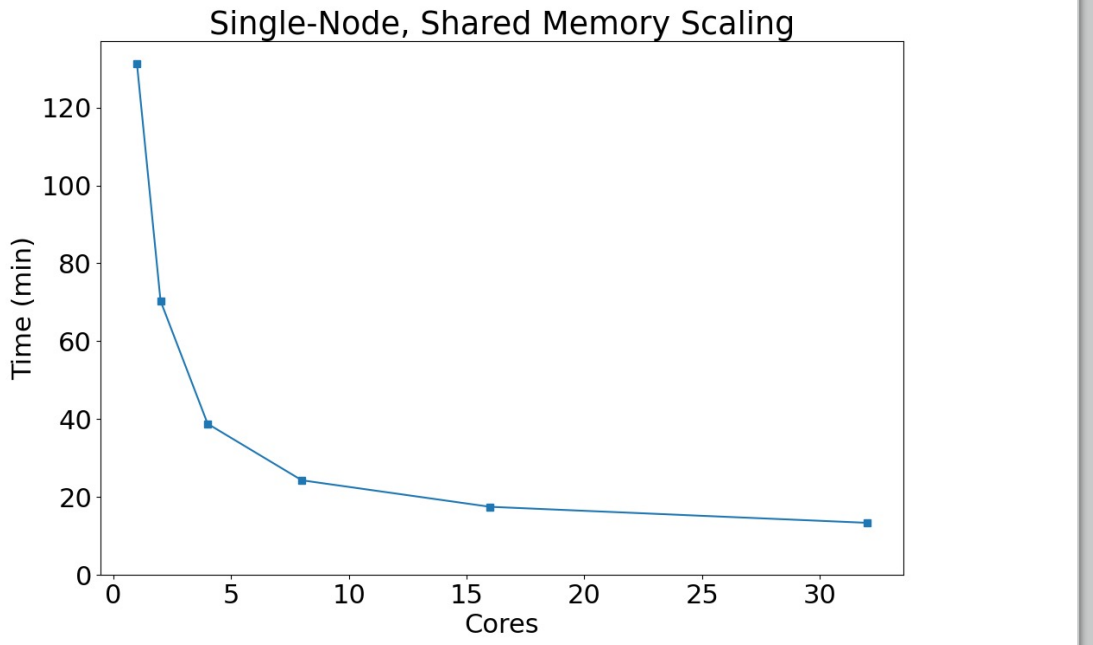
During 7-11am Peak Congestion:	34% Performance Improvement vs MSA
Overall (24 Hours):	16% Performance Improvement vs MSA

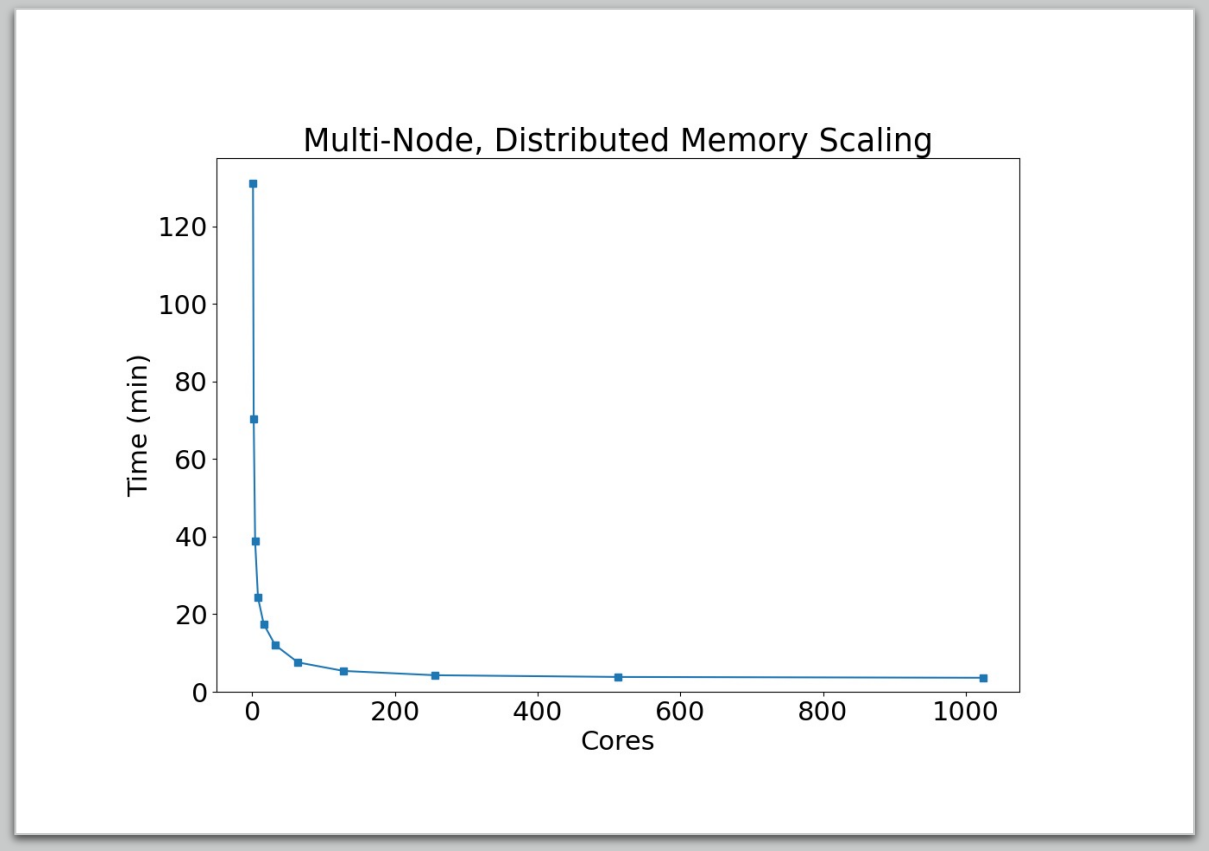




Parallel Compute Performance – SF Bay Area Network

Original Serial Run Time with 19 million trips over 2 million links: over 2 hours
Enabled **10x** Performance Speedup using 32 cores on a single node: 14 minutes
Enabled **36x** Performance Speedup (overall) using 32 nodes (1,024 cores): 4 minutes





Computational Performance for QDTA and Hybrid QDTA/Mobiliti Simulation

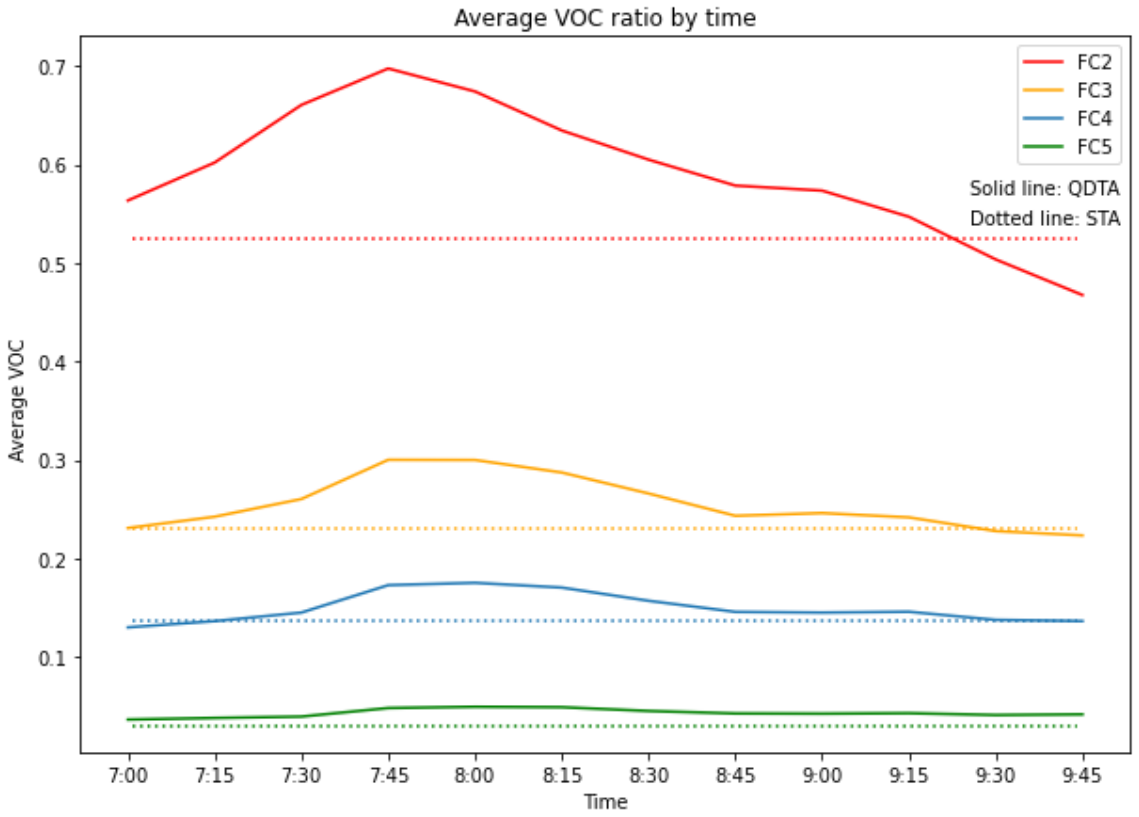
- Hybrid QDTA/Mobiliti Simulation:
  - Generate the QDTA defined routes
  - Simulate using Mobiliti (eems_037)

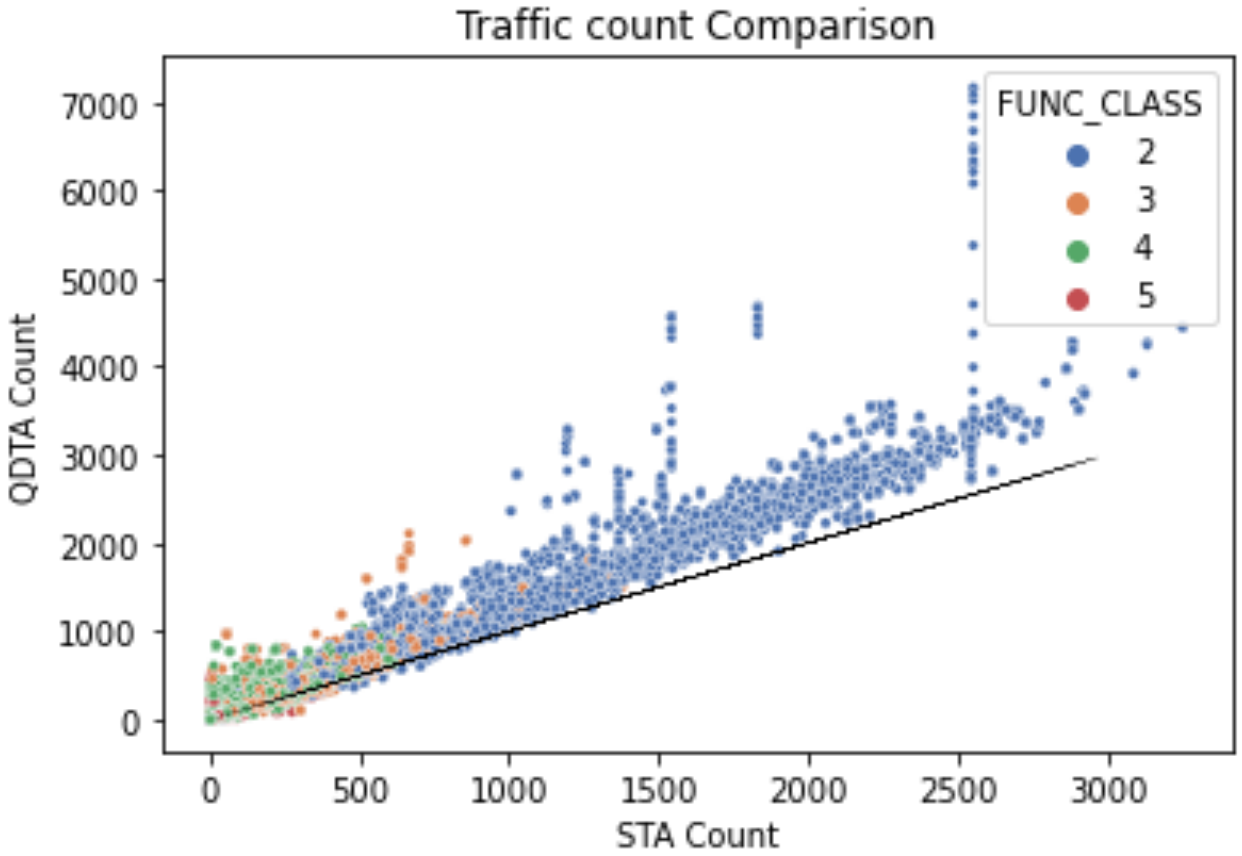
Scenario	Link Model	Compute Time (seconds)	Network & Demand Size
Bay Area UET QDTA	BPR	231	1M Links, 19M Trips
Bay Area Hybrid Simulation	BPR	33	
Bay Area Hybrid Simulation	Queue & Storage	93	
Los Angeles UET DTA	BPR	596	2M Links, 40M Trips
Los Angeles Hybrid Simulation	BPR	143	
Los Angeles Hybrid Simulation	Queue & Storage	570	

Allows comparison to other types of route assignment, e.g. part of the fleet is dynamically routed or impact of network changes during simulation

Average Velocity over Capacity

Classified by Functional Class of Link





System Metrics

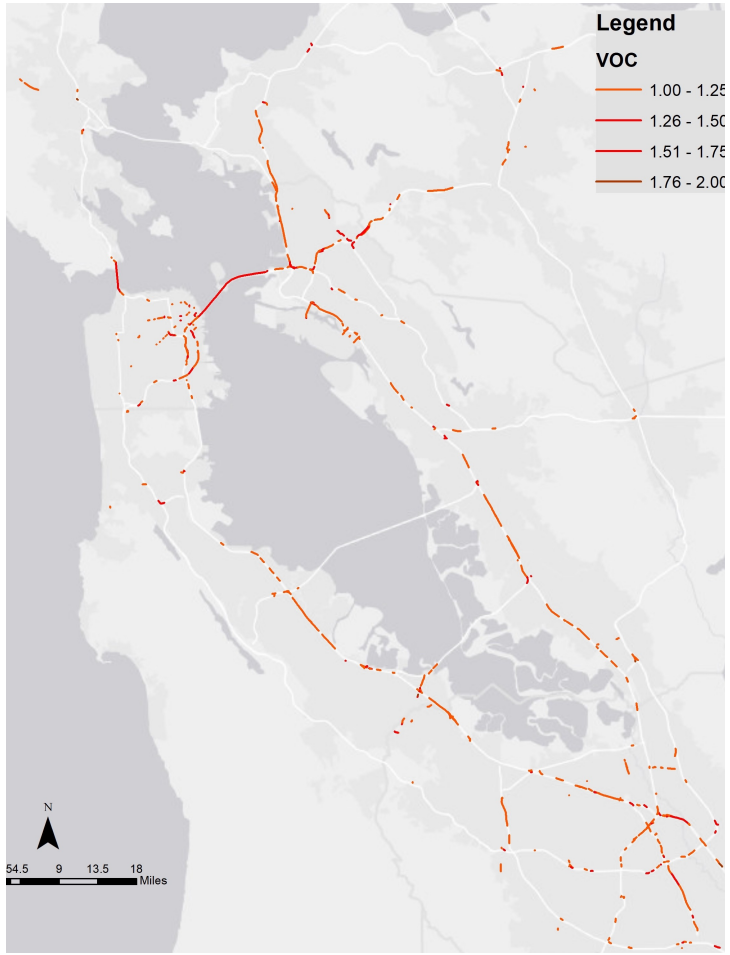
System Level

Category	VMT (in millions)		Average VOC		VHD (in thousands)	
	STA	QDTA	STA	QDTA	STA	QDTA
FC2	16.44	18.11	0.52	0.69	17.57	96.84
FC3	4.67	5.01	0.23	0.30	2.65	14.43
FC4	4.96	5.25	0.14	0.17	1.13	6.42
FC5	2.43	2.60	0.03	0.48	0.38	17.78
Total	28.51	30.98	-	-	21.95	135.49

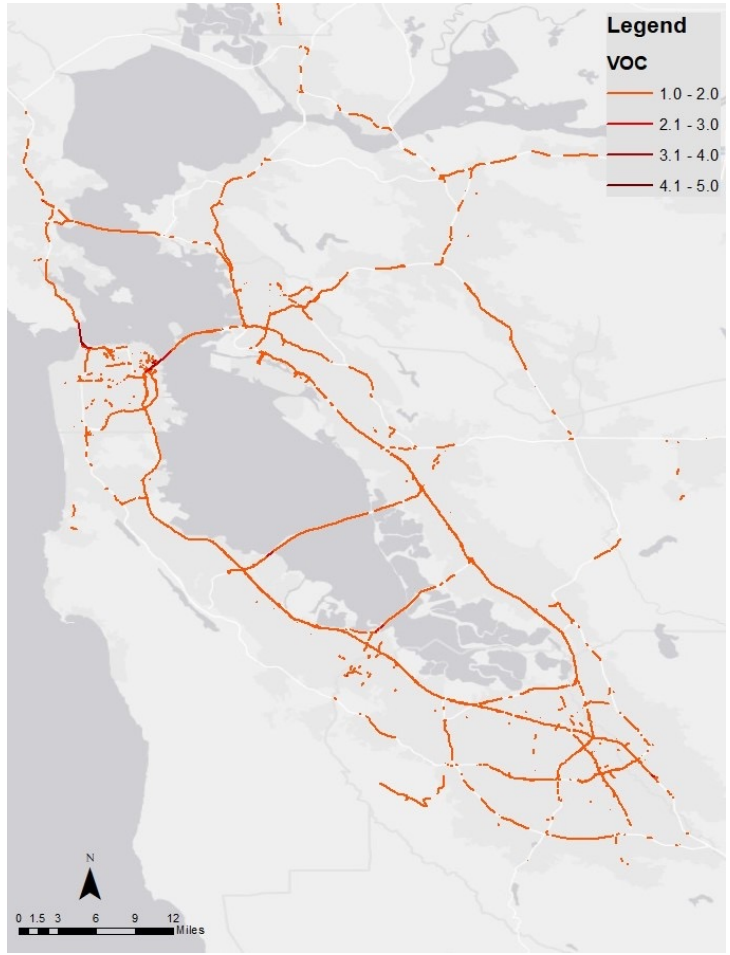
Congested Network

Category	Length (Km)		VMT(millions)		Average VOC	
	STA	QDTA	STA	QDTA	STA	QDTA
FC2	131	553	1.77	7.56	1.15	1.26
FC3	38	88	0.17	0.42	1.18	1.28
FC4	14	56	0.04	0.15	1.19	1.21
FC5	5	28	0.01	0.06	1.17	1.25
Total	188	725	2.0	8.19	-	-

Static Traffic Assignment Underestimates Congestion



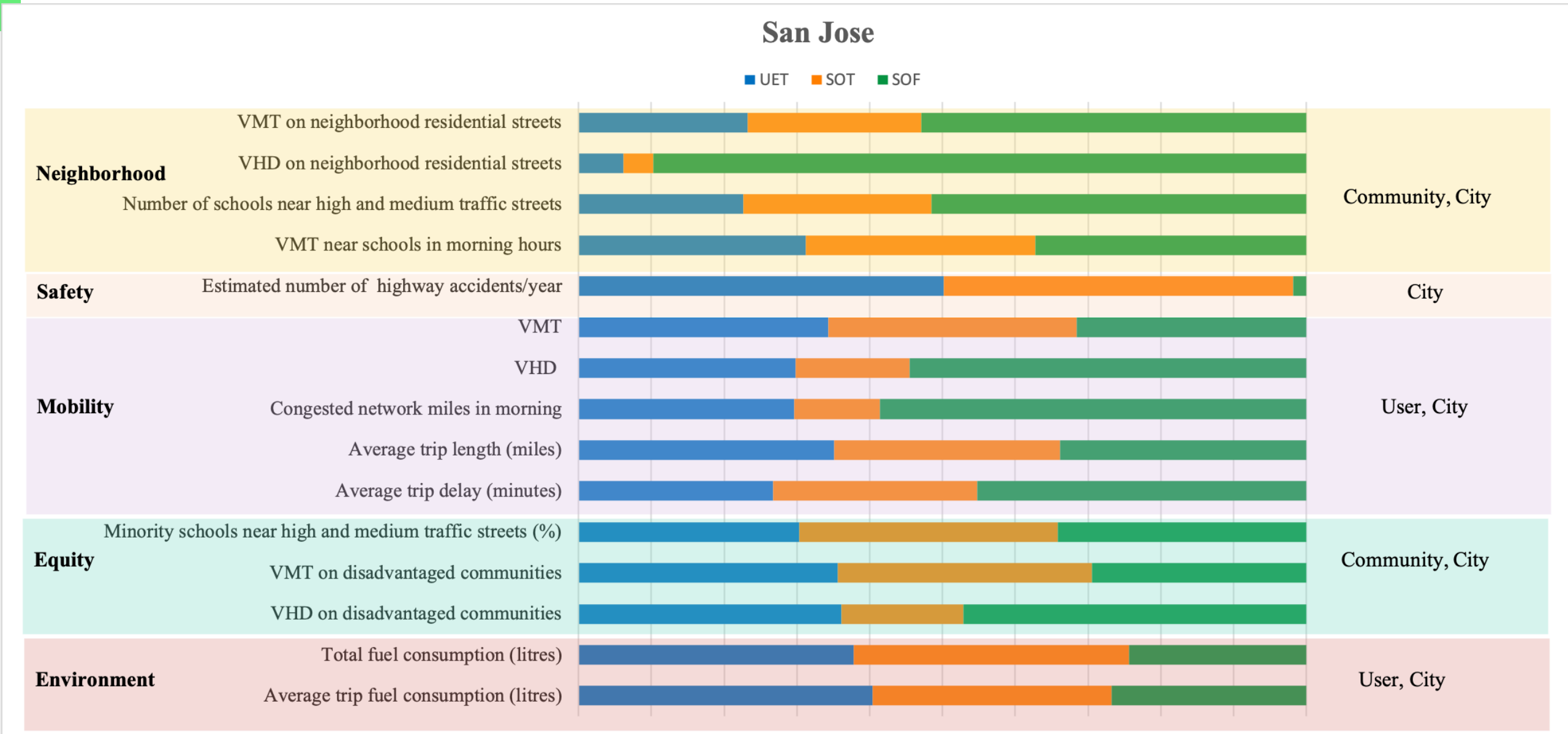
Static Traffic Assignment



Quasi Dynamic Traffic Assignment

- 8am to 9am traffic patterns
- VOC higher on a large part of the network

Evaluating the impacts on Cities



Evaluating the impacts on Cities

Theme: Neighborhood

- VMT disproportionately increases on neighborhood residential streets with SOT and SOF derived routes.
  - While the total system VMT reduces with SOT and SOF, the residential VMT increases for both cases. Neighborhood residential streets account for 4% of the total VMT in UET which increases to 5% and 11% with SOT and SOF respectively.
- The number of schools exposed to high and medium traffic increases significantly with SOF due to the shift to local roads.
  - Exposure to high and medium traffic occurs for 9% of schools in UET. This percentage slightly increases with SOT and doubles with SOF.

Theme: Mobility

- System VMT decreases with SOT and SOF compared to UET.
- VHD decrease with SOT and increase with SOF.

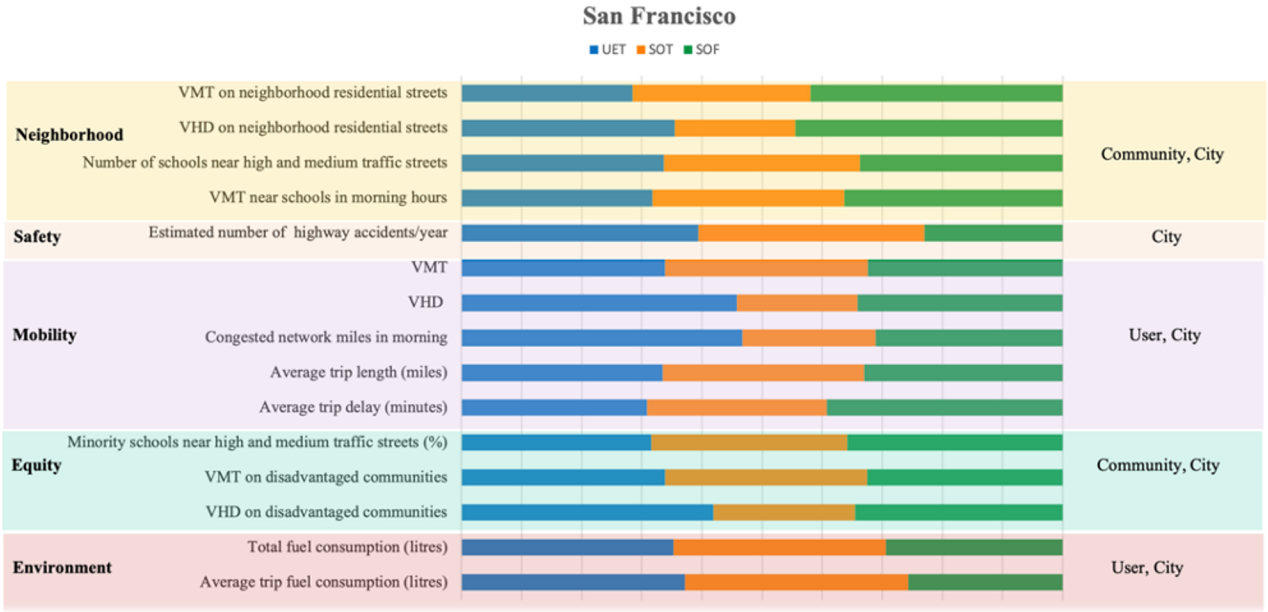
Evaluating the impacts on Cities

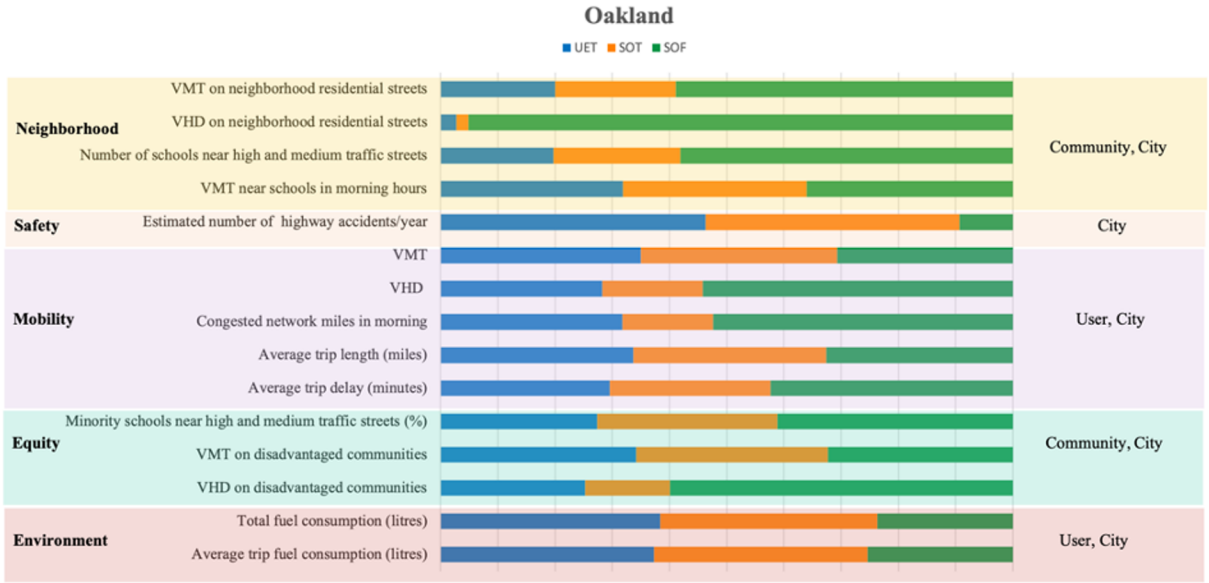
Theme: Equity

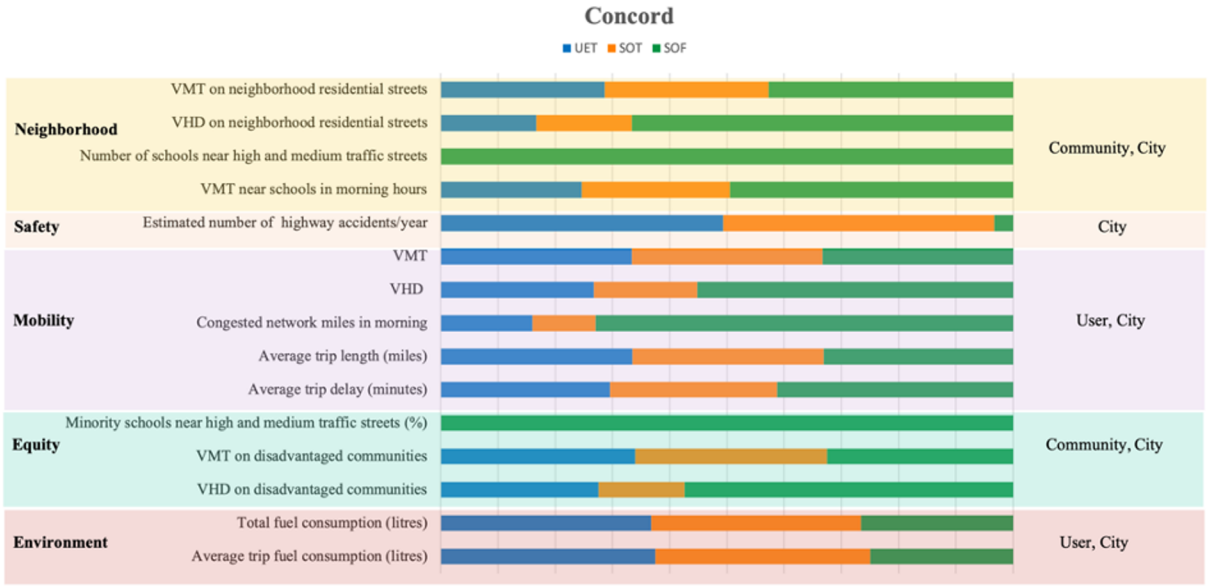
- Minority schools bear disproportionate impacts of traffic exposure.
  - 22% of schools are categorized as minority schools in San Jose. The proportion of schools affected by this predicted exposure is 41% in UET.
  - This percentage further increases by at least 5 percentage points with SOT and SOF.

- Consistent with total VMT reduction trends, VMT in disadvantaged communities reduces with SOT and SOF routing.
  - Note that the population in disadvantaged communities constitutes 32% of the total, but account for 40% VMT and 55% VHD with UET. A This reflects the tendency of these communities to be located near highways with high flow rates.
  - These percentages are reduced with SOT and SOF optimizations.

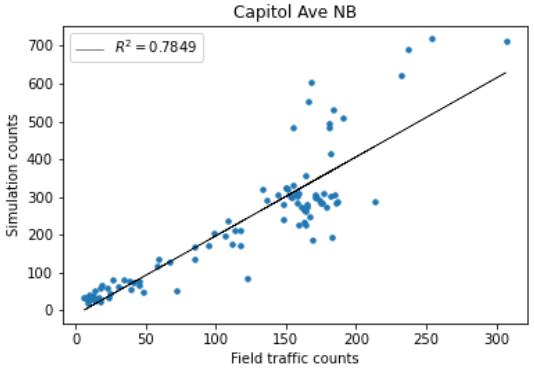
Comparison Across Cities

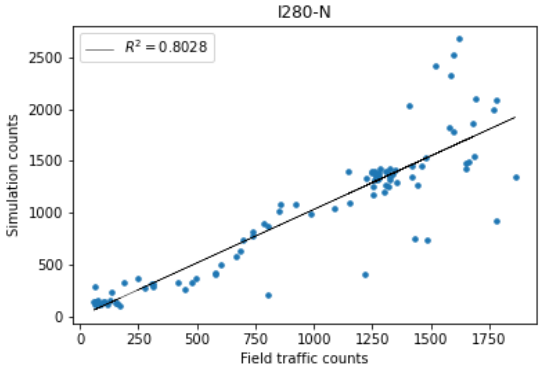


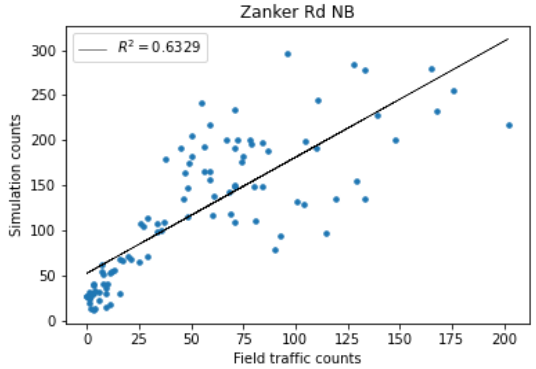


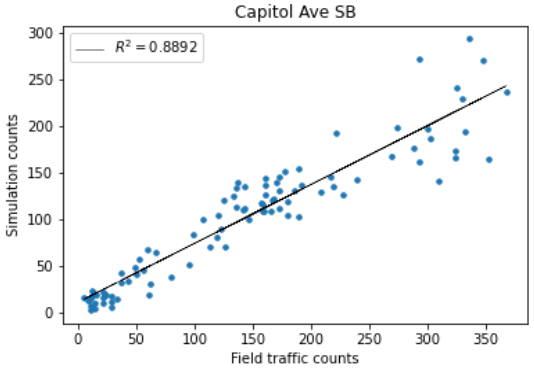


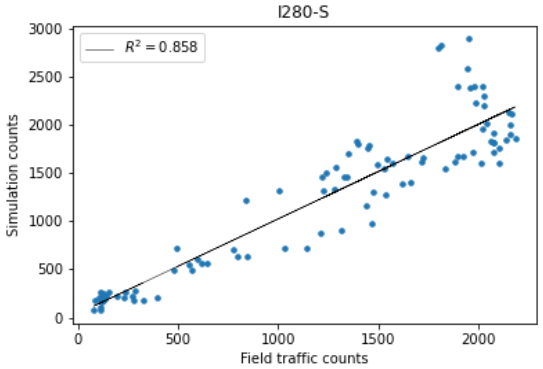
Validation – San Jose Vehicle Counts, PeMS

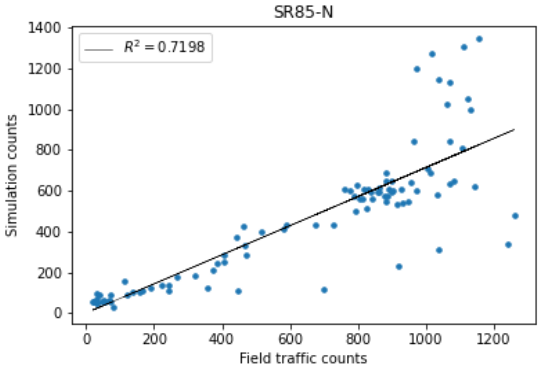


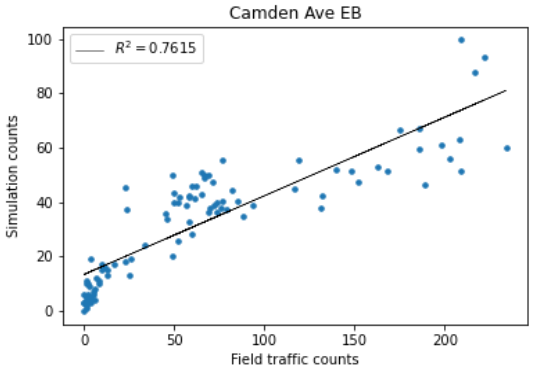


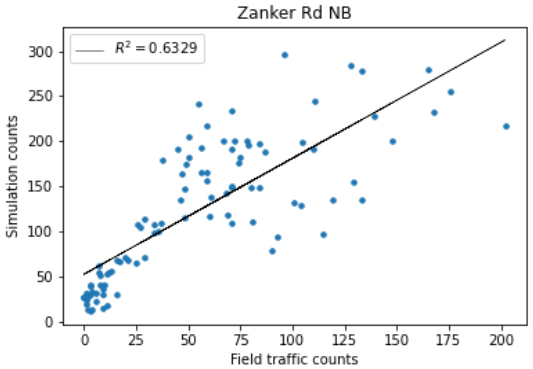


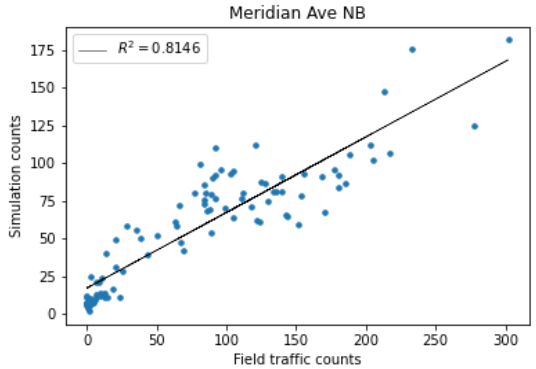












Response to Previous Comments

The reviewer noted the project was relevant as it applies HPC to large-scale network modeling and aims to support an eventual simulation framework

- ***We have integrated the work into the Mobiliti simulation platform, thus providing a capability to evaluate standard optimizations, such at User Equilibrium Travel Time and System Optimal Travel Time, in context of alternate approaches for optimizing traffic in large urban environments.***

The reviewer asked if planners would need supercomputers continually - as things change.

- ***The sister project to this, eems037, aims to generate surrogate models built by running many simulations to create training data for the purpose of obviating the need for HPC. We are also investigating moving the platform to a cloud computing environment.***

The reviewer noted our discussion regarding sensitivity to speed profiles when optimizing for fuel use.

- ***This is a topic of follow-on work in the eems037 project, in which we are exploring the appropriate level of fidelity for fuel use estimates. We are using real-world gps traces to inform these models in order to improve the cost measures for highly congested link states.***

Collaboration and Coordination

Uber

Challenges and Proposed Future Research

- Collaborate with the City of San Jose to include QDTA results into decision making for city
- Understand where the fuel focused objective function breaks down

- Further develop AI/deep learning solutions to the traffic management
  - Use QDTA as a fast solver to provide ensembles of training samples into our Mobiliti machine learning/ AI framework

Any proposed future work is subject to change based on funding levels.

Summary

- An implementation of a large-scale, urban traffic assignment that uses the newly developed Quasi Dynamic Traffic Assignment algorithms generated more realistic results for evaluating traffic than a standard multi-hour Static Traffic Assignment approach.
- The parallelized traffic assignment computational solution on HPC runs in significantly reduced time frames. **The Bay Area traffic assignment for a ~1M node network with 19M trip legs runs in ~4 minutes. The Los Angeles Basin traffic assignment for a ~2M node network with 40M trip legs runs in ~10 minutes.**
- Additional optimization objectives to the standard user travel time evaluation were also evaluated: system optimized travel time and system optimized fuel use.

Paper on ArXive https://arxiv.org/abs/2104.12911

Quasi-Dynamic Traffic Assignment using High Performance Computing

Cy Chan, Anu Kuncheria, Bingyu Zhao, Theophile Cabannes, Alexander Keimer, Bin Wang, Alexandre Bayen, Jane Macfarlane

Submitted to Transportation Research

Technical Backup

Algorithm 2

Algorithm 2: Traffic_assignment: using Frank-Wolfe's algorithm

Data: Network graph $G(\mathcal{V}, \mathcal{A})$
Time step length Δt
Travel demand of current step $\boldsymbol{d} \in \mathbb{R}_{\geq 0}^{\mathcal{V} \times \mathcal{V}}$
Free-flow travel time of each link $\boldsymbol{c}_0 = \{c_{0,a}\}$, with $a \in \mathcal{A}$

Take $\boldsymbol{h} = \text{All\_or\_nothing}(G, \Delta t, \boldsymbol{d}, \boldsymbol{c}_0)$; `// Set initial path flow in the free-flow condition`

while ***True*** ; `// Gradient descent step`
do
  $\boldsymbol{c} = \text{BPR}(\boldsymbol{\Delta h})$; `// Calculate the edge travel time`
  $\boldsymbol{h}^{AON} = \text{All\_or\_nothing}(G, \Delta t, \boldsymbol{d}, \boldsymbol{c})$; `// All-or-nothing path flow with new edge weights`
  $\alpha^{\star} = \arg\min_{\alpha} \text{Cost\_function}\left(\boldsymbol{\Delta}\left[\boldsymbol{h} + \alpha \cdot (\boldsymbol{h}^{AON} - \boldsymbol{h})\right]\right)$; `// Exact line search`
  $\boldsymbol{h}_{new} = \boldsymbol{h} + \alpha^{\star} \cdot (\boldsymbol{h}^{AON} - \boldsymbol{h})$; `// Update path flows`
  if *Converged* $(\boldsymbol{\Delta h}, \boldsymbol{\Delta h}_{new})$; `// Check convergence`
  then
    break
  end
  $\boldsymbol{h} = \boldsymbol{h}_{new}$;
end
Result: $\boldsymbol{h}$; `// Path flow using STA`

Algorithm 3

Algorithm 3: All_or_nothing: iterative step of the Frank-Wolfe Algorithm

Data: Network graph $G(\mathcal{V}, \mathcal{A})$
Time step length Δt
Travel demand of current step $\boldsymbol{d} \in \mathbb{R}_{\geq 0}^{\mathcal{V} \times \mathcal{V}}$
Edge travel time/cost $\boldsymbol{c} = \{c_a\}, a \in \mathcal{A}$

Initialize $\boldsymbol{h}$ = empty associative array ; // Initialize the path flow vector

for $d_{p,q} \in \boldsymbol{d} \mid d_{p,q} > 0$; // Iterate over non-zero elements
do
  r_{sp} = Get_shortest_path$(p, q, \boldsymbol{c}, G)$; // Get shortest path to destination
  r_{tp} = Truncate_path$(r_{sp}, \boldsymbol{c}, G, \Delta t)$; // Truncate path according to time step length
  $h_{r_{tp}} \mathrel{+}= d_{p,q}$; // Add trips to the path flow vector
end
Result: $\boldsymbol{h}$; // Path flow from all-or-nothing assignment

Algorithm 4

Algorithm 4: Residual_demand: trips that cannot finish in one assignment interval

Data: Network graph $G(\mathcal{V}, \mathcal{A})$
Time step length Δt_i
Intermediate path flow results from the STA $\boldsymbol{h}^{STA}(i)$
All paths used in the time step $\mathcal{R}(i) = \{\mathcal{R}_{p,q}\}$, with $(p,\ q) \in \mathbb{R}_{\geq 0}^{\mathcal{V}\times\mathcal{V}}$

Initialize $\boldsymbol{d}^r(i+1) =$ empty nested associative array ; // Initialize the residual demand vector, will be added to the demand of the next step
Initialize $\boldsymbol{h}(i) =$ empty associative array ; // Initialize the truncated path flow vector
Take $\boldsymbol{c} = BPR(\boldsymbol{\Delta h}^{STA}(i))$; // Edge travel time based on path/link flow from STA

for $r \in \mathcal{R}(i)$; // Parallelizable for loop
do
  $r_{tp} =$ Truncate_path$(r, \boldsymbol{c}, G, \Delta t_i)$; // Truncate path to what can be traversed in the time step
  $h_{r_{tp}}(i)\ \mathrel{+}= h_r^{STA}(i)$; // Populate the path flow vector
  if $r_{tp} \neq r$; // Flow has not reached its destination within the time slice
  then
    $s =$ Get_last_vertex(r_{tp});
    $q =$ Get_last_vertex(r);
    $d_{s,q}^r(i+1)\ \mathrel{+}= h_r^{STA}(i)$; // Add to residual demand
  end
end
Result: $\boldsymbol{h}(i), \boldsymbol{d}^r(i+1)$; // Path flow for the current time step using QDTA, and residual demand to be added to the next time step