# High Performance Traffic Assignment Based on Variatonal Inequality

Students: XIAO Yujie (CityU), SHI Zhenmei (HKUST)
Mentors: Cheng Liu (ORNL), Kwai Wong (UTK)

## Abstract

Variational Inequality (VI) is a mathematical problem that is widely applied to equilibrium problems in different fields. This project focuses on modeling the transportation assignment problem using variational inequality and solving it in a highly efficient way. The solver implements on GPU using parallel computing.

The problem can be classified into two categories Static Traffic Assignment (STA) and Dynamic Traffic Assignment. STA, in this project, refers to solving for the user equilibrium which is a Nash equilibrium about the travel cost of each user without considering about demand changing over time. DTA refers to solving for the dynamic user equilibrium where the cost of travel is minimized for every user, in a continuous period of time.

## Static Traffic Assignment

*Nonlinear Complementarity Problem* (NCP)

Given a mapping $F: R_+^n \to R^n$, the NCP ($F$) is to find a vector $x \in R^n$ satisfying $0 \le x \perp F(x) \ge 0$.

*Static traffic assignment* (STA):

The STA problem in the NCP formulation as following:

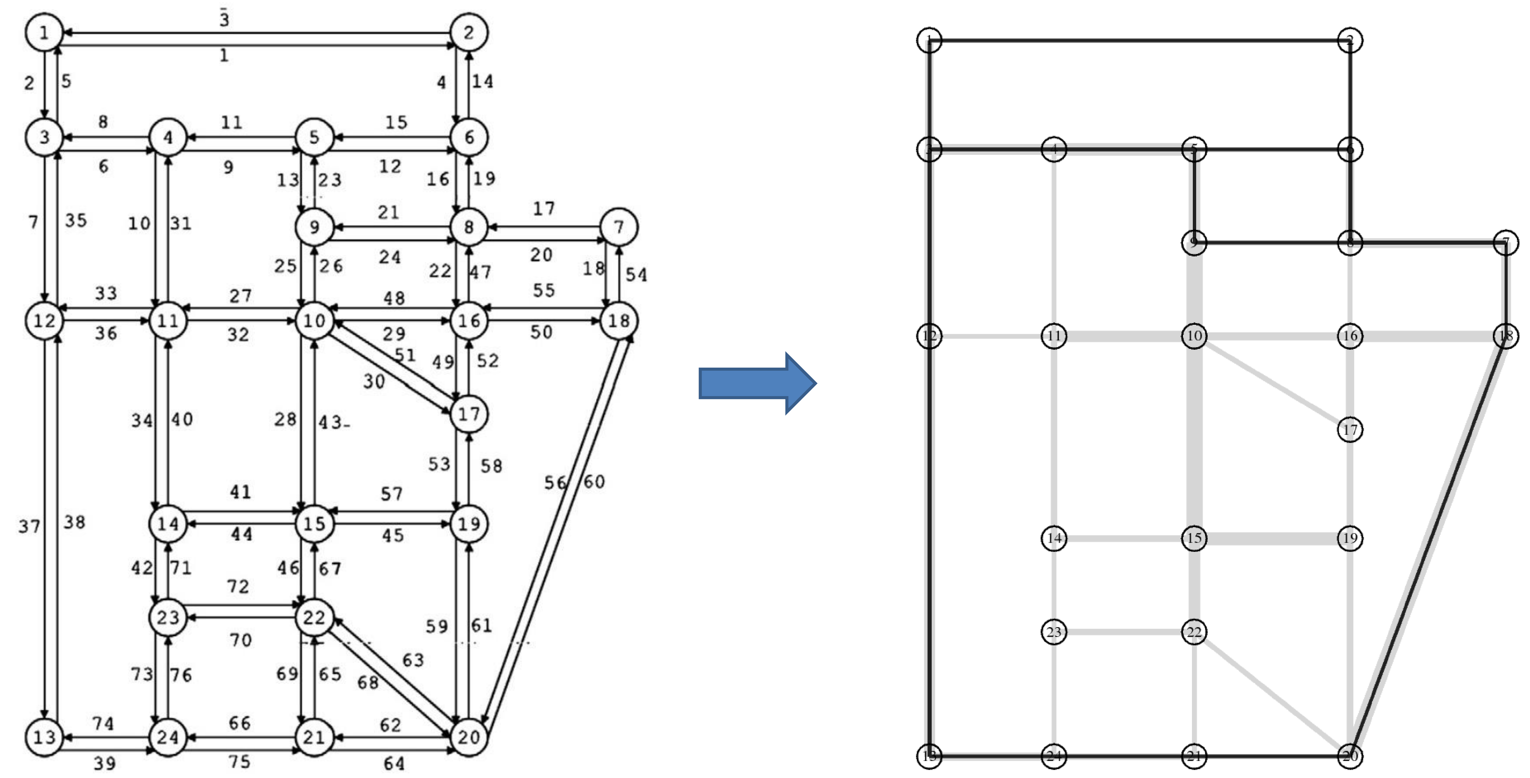
$$0 \le C_p(h) - u_w \perp h_p \ge 0, \quad \forall w \in \mathcal{W} \text{ and } p \in \mathcal{P}_w;$$
$$\sum_{p \in \mathcal{P}_w} h_p = d_w(u), \quad \forall w \in \mathcal{W},$$
$$u_w \ge 0, \quad w \in \mathcal{W}.$$
$$\mathbf{F}(h,u) \equiv \begin{pmatrix} C(h) - \Omega^T u \\ \Omega h - d(u) \end{pmatrix},$$

In this formulation, find the candidate paths between each Origin – Destination (OD) pair is important.

*Algorithm and Implementation*

**Step 1** Use shortest path algorithm to find 7 paths for each OD pair. Here the solver uses nvGRAPH package in CUDA library which runs on GPU.
**Step 2** Convert all data in to NCP formulation.
**Step 3** Use NCP FBLSA Algorithm to solve the problem with given error bound. Here the solver uses Siconos package which is non-smooth numerical simulation and cuSPARSE library which also runs on GPU.

*Example and Results*



For the network as shown in the upper left graph, the assignment result can be shown on the graph. The running time can be reduced to one third compared with Frank-Wolfe algorithm in this example. GPU acceleration is significant.

## Dynamic Traffic Assignment

*Dynamic traffic assignment* (DTA) models determine departure rates, departure times and route choices over a given planning horizon.

*Dynamic network loading* (DNL) is a critical procedure for obtaining the traffic delay operator, which maps a traversal time to the departure time. In this project, the DNL procedure is approximated by a system of ODEs:

$$\frac{dx_{a_i}^p(t)}{dt} = g_{a_{i-1}}^p(t) - g_{a_i}^p(t) \quad \forall p \in \mathcal{P}, i \in [1, m(p)]$$
$$x_{a_i}^p(0) = x_{a_i}^{p,0} \in \Re_+^1 \quad \forall p \in \mathcal{P}, i \in [1, m(p)]$$
$$h_p^{\tau,k}(t) = g_{a_1}(t + D_{a_1}[x_{a_1}(t)])\left(1 + D'_{a_1}[x_{a_1}(t)]\dot{x}_{a_1}\right)$$
$$g_{a_{i-1}}^p(t) = g_{a_i}^p(t + D_{a_i}[x_{a_i}(t)])\left(1 + D'_{a_i}[x_{a_i}(t)]\dot{x}_{a_i}(t)\right) \quad \forall p \in \mathcal{P}, i \in [2, m(p)]$$

*Differential variational inequality* (DVI) is a particular branch of variational inequality(VI). And we employ it to formulate the DTA problem.

*Fixed point iteration*:

The original DVI problem is restated as a fixed point iteration in the computing process as following:

$$h_p^{k+1}(t) = \left[h_p^k(t) - \alpha \Psi_p\left(t, h^k\right) + v_{ij}\right]_+ \quad \forall (i,j) \in \mathcal{W}, p \in \mathcal{P}_{ij}$$
$$\sum_{p \in \mathcal{P}_{ij}} \int_{t_0}^{t_f} \left[h_p^k(t) - \alpha \Psi_p(t, h^k) + v_{ij}\right]_+ dt = Q_{ij} \quad \forall (i,j) \in \mathcal{W}$$

*Algorithm*

For each OD pair, given $h_0$

While $||h^{k+1} - h^k|| / ||h^k|| >= \varepsilon$
ODE($h^k$) = makeOde($h^k$)
x = sol. to ODE($h^k$)
for i = 1:m
$\xi_1 = t + D_1[x_1(t)]$
$\xi_{i+1} = \xi_i + D_i[x_i(\xi_i)]$
End for

$D_p = \xi_m$

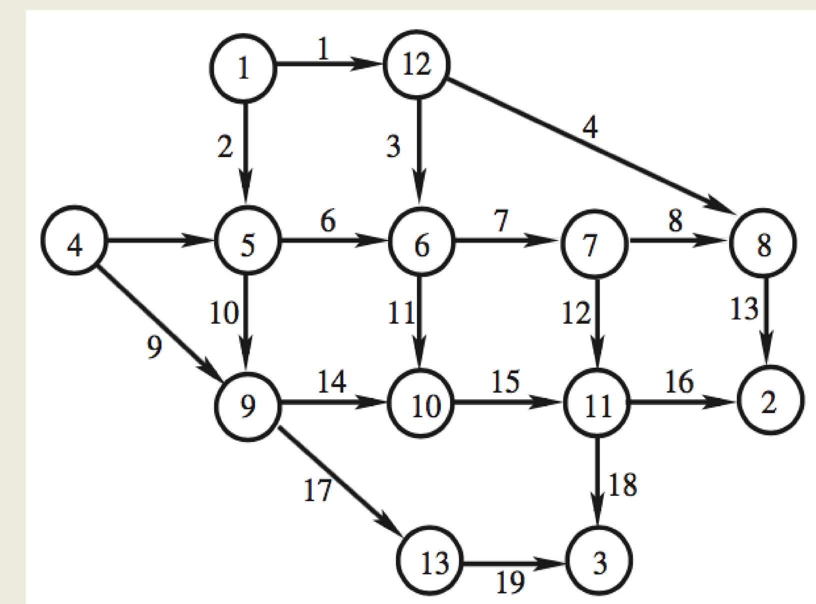
$\Psi(h^k) = F(D_p)$
$\sum \int [hk - \alpha\Psi(h^k) + v]_+ = Q$
v = zero of
$\sum \int [hk - \alpha\Psi(h^k) + v]_+ - Q$

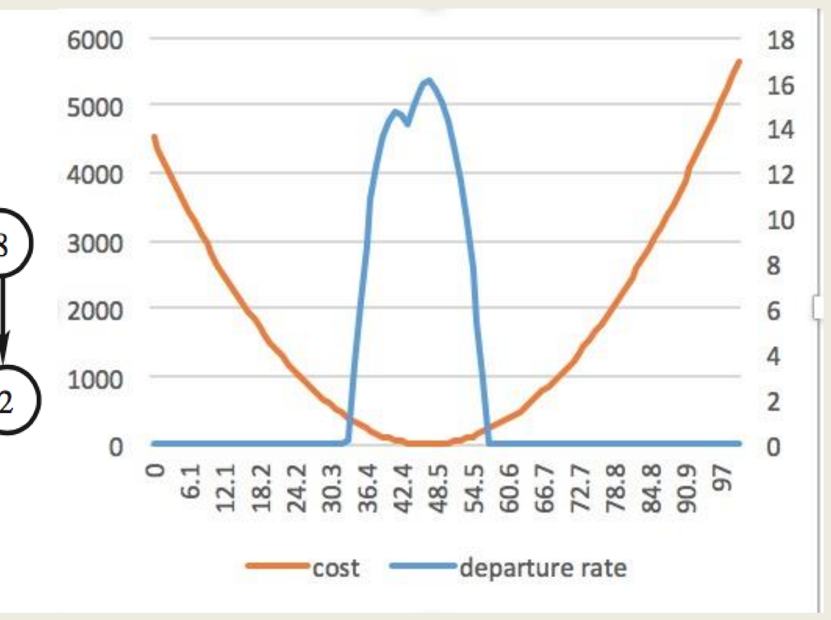
$h^{k+1} = [hk - \alpha\Psi(h^k) + v]_+$

End while

Cost = $\Psi(h)$

*Example and Results*






For the network as shown in the upper left graph, considering the characters of 19 different road, then the optimal choice of departure rate is presented in the graph in the upper right side.

## Future Work

For STA, the parallel computing is simply link CUDA library. Optimization in parallel method for this specific problem is still available.

For DTA, the work already done is still in the early stage, there remains a lot of work to optimize the algorithm, including verifying the convergence of the algorithm theoretically.

## Acknowledgements

Support from National Science Foundation, Joint Institute for Computational Sciences, University of Tennessee, Hong Kong University of Science and Technology, City University of Hong Kong, and Oak Ridge National Laboratory are gratefully acknowledged.