An Enhanced Travel Demand Modeling Framework with a Post-Processing Technique Executed through a Feedback Mechanism

Amit Kumar
Purdue University

Dr. Srinivas Peeta
Purdue University

4th Transportation Research Board Conference on Innovations in Travel Modeling (ITM)
Tampa, Florida, April 30th-May 2nd 2012

May 1st 2012
Overview

- Introduction
- Problem Statement
- Proposed Methodology
- Computational Experiments
- Insights
Introduction
Introduction

Overall planning and implementation process

**PREANALYSIS PHASE**
- Problem Identification
- Formulation of Goals and Objectives
- Data Collection
- Generation of Alternatives

**TECHNICAL ANALYSIS PHASE**
- Land Use –Activity System Model
- Urban Transportation Model System (UTMS)
- Impact Prediction Models
- Evaluation of Alternatives/scenarios

**POSTANALYSIS PHASE**
- Decision Making
- Implementation
- Monitoring
Introduction

- Urban Transportation Modeling System (UTMS) current practice

  - Four stage transportation modeling process

  - TRIP GENERATION
    (How many trips? $T_i$)

  - TRIP DISTRIBUTION
    (Where do they go? $T_{ij}$)

  - MODE CHOICE
    (By what mode? $T_{ijm}$)

  - TRAFFIC ASSIGNMENT
    (By what route? $T_{ijmk}$)

  Need for feedback loop
Introduction

- Feedback loop is not often used in current practice

  • Reasons:
    - Traffic assignment process is carried out without taking advantage of previously obtained results, hence requiring more computational time
    - Each loop iteration may result in different set of solution leading to non-convergence
Problem Statement

• To develop post-processing technique to solve the following practical issues associated with user equilibrium traffic assignment in 4 step planning process
  ◆ Stability problem: Flow is very sensitive to small changes in the network
  ◆ Consistency problem: Link far removed from alternative being studied has perceptible flow change
  ◆ Convergence: Stable convergence requires impractical number of iterations with conventional Frank-Wolfe algorithm
Proposed Methodology
Conceptual Methodology

1. Start
   
   Set $n=1$

   Input: Network topology, link performance functions

   Trip Generation

   Trip Distribution

   Mode Choice

   If $n>1$ yes

   Traffic Assignment

   Post Processing

   Convergence achieved? yes

   n=n+1

   If n>1 yes

   Stop

   No

   No
Post Processing Steps

Input:
Network topology, link performance functions, O-D demand, path definition and path flows

If n>1

Yes

O-D Prioritization

Perturbation assignment

No

Improve convergence using SMPA

Store path definition, path flows and link flows

Database

Database
Start

Set iter = 1

Update flow, cost and slope for links, and then paths for all O-D pairs

Is current O-D pair the last O-D pair?

Yes

Go to the next O-D pair

No

Is $|\mu - c_{av}| < \beta$?

Yes

Update flow, cost and slope for links, and then paths for the current O-D pair

No

Check for violation of non-negativity constraint. Backtrack and project any infeasible flow back to the feasible space.

Select the first O-D pair

Apply flow change for costlier path set

Compute $c_{av}$ and compute shortest path. Update feasible path set. Check if O-D pair can be skipped.

Set iter = iter + 1

Save path flows and path costs

Is $Ngap < \epsilon$?

Yes

No

Apply flow change for cheaper path set

Update flow, cost and slope for links, and then paths for all O-D pairs

Network topology, link properties

Stop

Ref: Kumar and Peeta (2010)
Flow Update Mechanism of SMPA

Flow (f)

Cost (C)

C_{av}

P

1 2 3 4 5 6

ITM 2012
Flow Update Mechanism of SMPA

Cost (C)

$C_{av}$

$\mu$

Flow (f)

1 2 3 4 5 6
Need For a Hybrid Approach

- SMPA is based on the Gauss-Seidel Decomposition technique
  - Updates path set for a single O-D pair at-a-time
  - Updates flows between all the paths between an O-D pair at-a-time
- The sequential (one-at-a-time based) technique is good for small and medium size networks but it is not good for large size networks
  - Reason: gain in higher rate of convergence per iteration is compensated by the computational time required for generating the shortest paths and updating the path set based on sequential approach

Ref: Kumar, Peeta and Nie (2012)
Hybrid version of SMPA

- Shortest paths are generated and sets of paths are updated for all the O-D pairs simultaneously.
- Paths for each O-D pair are equilibrated and flows are updated based on the sequential approach.
Flow Chart of SMPA

1. **Start**
   - Network topology, link properties
   - Set \( \text{iter} = 1 \)

2. **Initialize the network** (AON or warm start)
   - \( \text{Is } N_{gap} < \varepsilon ? \)
     - Yes
       - Update flow, cost, and slope for links, and then paths for all O-D pairs
     - No
       - Go to the next O-D pair

3. **Check if O-D pair can be skipped.**
   - Update flow, cost, and slope for links, and then paths for the current O-D pair

4. **Apply flow change for costlier path set**

5. **Apply flow change for cheaper path set**

6. **Check for violation of non-negativity constraint. Backtrack and project any infeasible flow back to the feasible space.**

7. **Compute \( c_{av} \) and compute shortest path. Update feasible path set. Check if O-D pair can be skipped.**

8. **Select the first O-D pair**
   - \( \text{iter} = \text{iter} + 1 \)
   - Save path flows and path costs

9. **Stop**

Reference: Kumar and Peeta (2010)
Flow Chart of SMPA Hybrid

Start

Set iter=1

Initialize the network (AON or warm start)

Is |μ-c_{av}| < β?

Yes

Update flow, cost and slope for links, and then paths for the current O-D pair

Go to the next O-D pair

No

Is Ngap < ε?

Yes

Update the path sets for all the O-D pairs

Select the first O-D pair

Save path flows and path costs

Select the first O-D pair

Compute c_{av}. Check if O-D pair can be skipped.

iter=iter+1

No

Apply flow change for costlier path set

Apply flow change for cheaper path set

Check for violation of non-negativity constraint. Backtrack and project any infeasible flow back to the feasible space.

Update flow, cost and slope for links, and then paths for the current O-D pair

Update flow, cost and slope for links, and then paths for all O-D pairs

Stop

ITM 2012
Post Processing Steps

Input:
Network topology, link performance functions, O-D demand, path definition and path flows

If n>1

Yes

O-D Prioritization

Perturbation assignment

Database

No

Improve convergence using SMPA

Store path definition, path flows and link flows

Database
Perturbation Assignment

- Used when traffic assignment needs to be carried out with slightly different input

- Used in three cases
  - When O-D demand changes
  - When network topology changes
  - When link properties change

- Can be used only with path-based algorithms
Perturbation Assignment

- If old O-D demand = 0
  - If new O-D demand = 0
    - Skip this O-D pair
  - Else
    - Assign all demand to shortest path
  - End

- Else
  - new O-D demand
    - new path flow = \( \left( \frac{\text{new O-D demand}}{\text{old O-D demand}} \right) \times \text{old path flow} \)
  - End
Perturbation Assignment

- Used when traffic assignment needs to be carried out with slightly different input
- Used in three cases
  - When O-D demand changes
  - When network topology changes
  - When link properties change
- Can be used only with path-based algorithms
Perturbation Assignment

- When some links are added
  - New efficient paths may come into existence which are not in the path sets
    - Such paths are generated during path set update process of traffic assignment algorithm (SMPA)

- When some links are deleted
  - Some paths in the path set may get deleted
    - This may be problematic
    - It can be tackled in two ways
Perturbation Assignment

- Two ways to avoid the problem arising due to the deletion of paths from the path set
  - Method 1: Distribute the flow of deleted path among other paths between the O-D pair connected by the deleted path
  - Method 2: Proper order of evaluation of network improvement alternatives/scenarios
    - Run the model for the scenario having smallest number of links and use this result for warm starting the other scenarios
Method 2 to avoid the problem arising due to the deletion of paths from the path set

First
• Run the model for network 2
• Warm start uses solution from network 2

Second
• Run the model for network 1 using warm start
• Save the solution

Network 1
Network 2
Post Processing Steps

Input:
Network topology, link performance functions, O-D demand, path definition and path flows

If n>1

Yes

O-D Prioritization

Perturbation assignment

Database

No

Improve convergence using SMPA

Store path definition, path flows and link flows

Database
Deciding the order in which O-D pairs are brought into equilibration process

Enables to achieve faster and stable convergence
Two types of approaches for the flow update

- Simultaneous Approach (All-at-a-time algorithms)
  - Equilibrates all O-D pairs at a time

- Sequential Approach (One-at-a-time algorithms)
  - Equilibrates one O-D pair at a time sequentially
  - Has been found to achieve convergence faster
  - The sequence of O-D pairs will affect the rate of convergence and the solution vector

How to decide this sequence
Methodology for O-D Prioritization

Start

Input:
Network topology,
link performance functions, O-D demand

Run the algorithms without O-D prioritization

Decide the possible ways of O-D prioritization

Run the algorithms with different ways of O-D prioritization

Compare the convergence rates

Compare the solution stability

Select the best way of O-D prioritization

Stop
Possible ways of O-D prioritization

- Based on the O-D demand
  - In ascending order
  - In descending order
- Based on the free flow travel time (proxy for proximity)
  - In ascending order
  - In descending order
- Combined measure of O-D demand and free flow travel time
Proposed Methodology

- Combined measure of O-D demand and free flow travel time
  - Normalization factor calculated as
    \[ nf = \frac{\text{mean[demand]}}{\text{mean[FF Travel time]}} \]
  - Final weight calculated as
    \[ wtn = wt \times nf \]
  - Weight \( wt \) is varied to find its optimum value
  - [priority index vector] = [demand vector] + wtn \times [vector of FF travel time]
  - Then prioritization done in increasing/decreasing order of elements of priority index vector
Computational Experiments
Computational Experiments

- **Test network**
  - Borman Corridor Network
    - 197 Nodes
    - 460 Links
    - 1681 O-D pairs with non-zero demand
Measure of Convergence

- Normalized Gap

\[
\text{Normalized Gap} = \frac{(\sum \sum \sum \sum \ldots) - (\sum \sum \sum \sum \ldots)}{(\sum \sum \sum \sum \ldots)}
\]

Weighted average excess cost

Convergence level: \( \text{Ngap} = 1E-06 \)
Deciding the optimal O-D prioritization criteria
Computational Experiments

- Deciding the optimal weight for the combined measure of O-D prioritization

![Graph showing CPU time (sec) vs Weight](image)
Computational Experiments

- Deciding the optimal weight for the combined measure of O-D prioritization
Deciding the optimal O-D prioritization criteria
Computational Experiments

- Benefit of warm start and O-D prioritization
  - 10 percent of the total O-D pairs were randomly selected
  - For the selected O-D pairs trip demand was adjusted by 10 percent
  - Traffic assignment was done by
    - SMPA alone
    - SMPA with optimal O-D prioritization
    - SMPA with warm start (using results of base case: original O-D demand)
    - SMPA with optimal O-D prioritization and warm start
Computational Experiments

- Benefit of warm start and O-D prioritization

![Graph showing computational experiments results](image-url)
Insights
Insights

- Efficient traffic assignment algorithm such as SMPA in combination with post processing technique can provide more stable solution.

- O-D prioritization can help to achieve higher level of convergence in lesser time.

- O-D prioritization leads to more stable values of link flows (from the early iterations) compared to non-prioritized algorithms.

- The best criterion for O-D prioritization varies across algorithms and from one network to another.

- O-D prioritization with warm start provides better basis for evaluation of alternatives and scenario analysis.
Questions?

Thank You
Additional Slides
User Equilibrium

- The Beckmann’s formulation (Transformation) for user equilibrium –

$$\min \text{ } F(P) = \sum_{a} \int_{0}^{F_{\text{low}(a)}} L_{ij}(P) \, dp$$

Subject to -

$$\sum_{j} \ L_{ij} = L_{ij} \quad (\text{Non-negativity constraint})$$

$$L_{ij} \geq 0 \quad (\text{Non-negativity constraint})$$

$$F_{\text{low}(a)} = \sum_{k} \sum_{s} \sum_{a} \ P_{ikj} \ L_{ij} \quad F_{\text{low}(a)} = 1 \quad \text{(flow on the arc a)}$$

Where,

- $F_{\text{low}(a)}$ = Flow on the arc $a$
- $L_{ij}$ = Cost of traversing arc $a$ (function of flow on arc $a$)
- $P_{ikj}$ = Path flow of path $k$ between an O-D pair $r-s$
- $L_{ij}$ = Demand for O-D pair $r-s$
The objective function decomposed into three parts:

\[ F(r,s) = \sum_{\text{hmm}} \int_0^{\text{hmm}} \text{F}_{\text{hmm}}(r,s) \text{F}_{\text{hmm}} + \sum_{\text{hmm}} \int_0^{\text{hmm}} \text{F}_{\text{hmm}} + (\Delta \text{F}_{\text{hmm}} - \Delta \text{F}_{\text{hmm}}) + \sum_{\text{hmm}} \int_0^{\text{hmm}} \text{F}_{\text{hmm}}(r,s) \text{F}_{\text{hmm}} \]

Where,
- \( \text{hmm} \) = Set of costlier paths, comprising of paths having cost greater than the average cost of the OD pair \( r-s \)
- \( \text{hmc} \) = Set of cheaper paths, comprising of paths having cost lesser than the average cost for the OD pair \( r-s \)
- \( \text{fla} \) = Flow on the arc \( a \)
- \( \text{tc} \) = Cost of traversing arc \( a \) (function of flow on arc \( a \))
- \( A \) = Set of arcs of the network

The third part represents the arcs which do not belong to any path in the feasible set of the O-D pair \( r-s \) being equilibrated

Ref: Kumar and Peeta (2010)
Flow Update Mechanism of SMPA

- The move direction for the flow update process in SMPA is:

\[ \Delta \bar{P}_{it} = \min \left\{ \bar{P}_{it}, \sum_{k \in \text{route}} \left( \frac{P_{it}^k (\bar{P}_{it}) - P_{it}^k (\bar{P}_{it})}{P_{it}^k (\bar{P}_{it})} \right) \right\} \]

\[ \Delta \bar{P}_{it} = \frac{\sum_{k \in \text{route}} \left( \bar{P}_{it}^k (\bar{P}_{it}) - \bar{P}_{it}^k (\bar{P}_{it}) \right)}{P_{it} (\bar{P}_{it}) \cdot \sum_{k \in \text{route}} \frac{1}{P_{it}^k (\bar{P}_{it})}} \]

- Flow update mechanism:

\[ \bar{P}_{it} \to \bar{P}_{it} - \Delta \bar{P}_{it}, \quad \bar{P}_{it} \in \bar{P} \quad (1) \]

\[ \bar{P}_{it} \to \bar{P}_{it} + \Delta \bar{P}_{it}, \quad \bar{P}_{it} \in \bar{P} \quad (2) \]

Ref: Kumar and Peeta (2010)