Equilibrium with link interactions

CE 392C

So far we have assumed that the travel time on a link is solely a function of that link's flow: $t_{ij}(x_{ij})$

In reality, a link's travel time may depend on other links' flow as well:

- Highways where overtaking is allowed
- Merges
- Queues spilling back

The second extension we are seeing to TAP allows these types of interactions to occur.

We're going back to the fixed demand problem; in Act II of the class we are not "stacking" the extensions to TAP. Might make an interesting course project, though.

Now, each link's travel time can depend on the flows on other links: $t_{ij}(\cdots, x_{ij}, \cdots)$. More compactly we can write $t_{ij}(\mathbf{x})$ as a function of the entire vector of link flows.

Everything else is the same as in the beginning: fixed demand, we seek a solution satisfying the principle of user equilibrium.

Obvious question: can we just modify the Beckmann function in some way?

Other questions: can we still guarantee existence and uniqueness of equilibrium solutions?

The Beckmann function is

$$\sum_{(i,j)\in A} \int_0^{x_{ij}} t_{ij}(x)\ dx$$

Can we just replace the upper limit of integration x_{ij} with $\mathbf{x}$, and integrate from the origin $\mathbf{0}$ to $\mathbf{x}$?

This would turn the regular integral into a line integral. However, line integrals generally depend on the path taken between the start and endpoints... so, the Beckmann function is no longer well-defined.

There is one special exception. This line integral is path-independent if $\mathbf{t}(\mathbf{x})$ is a conservative vector field.

For this to happen, we need the following symmetry condition:

For any feasible $\mathbf{x}$ and any two links (i, j) and (k, ℓ)

$$\frac{\partial t_{ij}}{\partial x_{k\ell}} = \frac{\partial t_{k\ell}}{\partial x_{ij}}$$

In this case, the Beckmann function is well-defined and all of the previous results hold.

However, in most cases the symmetric assumption is not reasonable to take. (Why?)

In the asymmetric case, the equilibrium problem cannot be formulated as the solution to a convex optimization problem.

However, the variational inequality approach still works, and equilibrium path flows $\hat{\mathbf{h}}$ still satisfy

$$\mathbf{c}(\hat{\mathbf{h}}) \cdot (\hat{\mathbf{h}} - \mathbf{h}) \leq 0$$

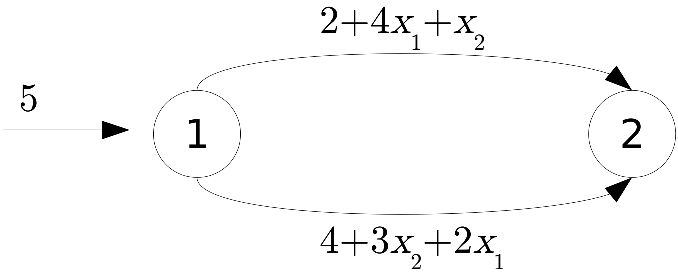
for all $\mathbf{h} \in H$ or equivalently equilibrium link flows $\hat{\mathbf{x}}$ satisfy

$$\mathbf{t}(\hat{\mathbf{x}}) \cdot (\hat{\mathbf{x}} - \mathbf{x}) \leq 0$$

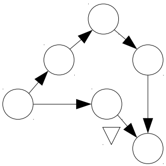
for all $\mathbf{x} \in X$.

This is a general pattern: even if we can't write down a convex optimization problem, we can often still write down a variational inequality. VIs are a more general modeling technique, but this generality has a cost.

EXAMPLES


$2+4x_1+x_2$
5
1
2
$4+3x_2+2x_1$

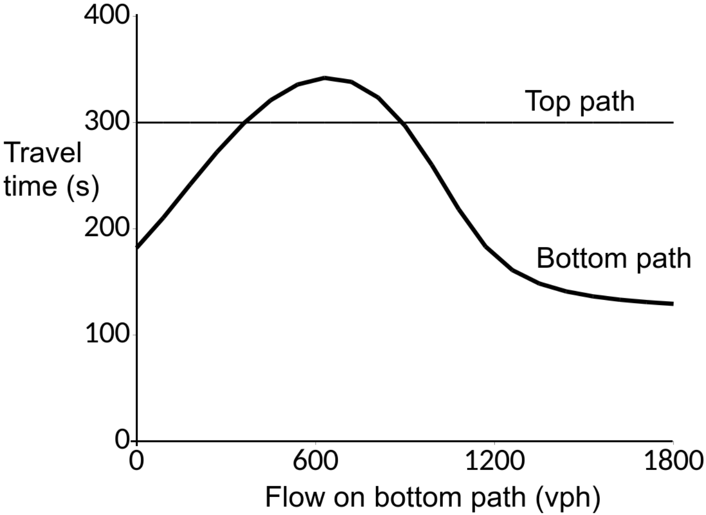
In this network, each link has a free-flow time of 1 unit and demand is 1800 vehicles.



The bottom path is shorter, but must yield to traffic on the top path.

The link performance functions for all links are constant (1) except for the yield link which depends on both the flow on its own link as well as on the link it yields to.

Don't worry about the exact equation for this formula, it comes from "gap acceptance" concepts.


400
300
200
100
0
Travel
time (s)
Top path
Bottom path
0
600
1200
1800
Flow on bottom path (vph)

There are three equilibrium solutions. Which is likely to occur?

Only two of the three are “stable” to small changes in link flow.

EXISTENCE

Does an equilibrium solution always exist?

Since the variational inequality is the same as before, as long as the link performance functions are continuous, an equilibrium exists.

MONOTONICITY AND UNIQUENESS

Last class, we saw a case where three equilibria existed.

Can we describe why this happened mathematically?

When we move flow from one path (say, h_1) to another h_2, intuitively the difference in path costs $c_2 - c_1$ should *increase*.

Let $\mathbf{c}(\mathbf{h})$ be the vector of path costs as a function of path flows.

This vector-valued function is *strictly monotone* if $(\mathbf{c}(\mathbf{h}') - \mathbf{c}(\mathbf{h})) \cdot (\mathbf{h}' - \mathbf{h}) > 0$ for all $h \neq h'$.

Note: this is stronger than requiring that $\mathbf{c}(\mathbf{h})$ is increasing in each of the path flows.

If $\mathbf{c}(\mathbf{h})$ is strictly monotone, then the equilibrium solution is unique.

Let $\hat{\mathbf{h}}$ be an equilibrium solution and let $\mathbf{h}$ be any other solution.

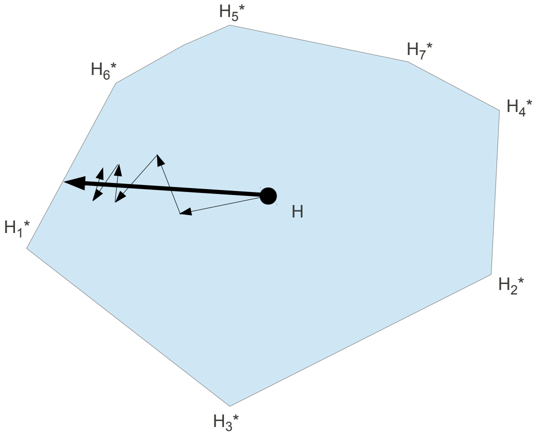
Then
$\mathbf{c}(\mathbf{h})(\mathbf{h}-\hat{\mathbf{h}}) = (\mathbf{c}(\mathbf{h})-\mathbf{c}(\hat{\mathbf{h}}))(\mathbf{h}-\hat{\mathbf{h}}) + \mathbf{c}(\hat{\mathbf{h}})(\mathbf{h}-\hat{\mathbf{h}}) > \mathbf{c}(\hat{\mathbf{h}})(\hat{\mathbf{h}}-\mathbf{h}) \geq \mathbf{0}$, so $\mathbf{h}$ is not an equilibrium.

SIMPLICIAL DECOMPOSITION ALGORITHM

Simplicial decomposition is more sophisticated than convex combination algorithms: rather than "forgetting" the $\mathbf{x}^*$ vectors from previous iterations, we will save them

The set $\mathcal{X} = \{\mathbf{x_1^*}, \mathbf{x_2^*}, \ldots, \mathbf{x_k^*}\}$ stores the $\mathbf{x}^*$ vectors from each iteration (ignoring duplicates).

Comparing simplicial decomposition and convex combinations



By combining multiple directions, simplicial decomposition can reach equilibrium faster.

Simplicial Decomposition Algorithm

This algorithm has two components: the **master algorithm** and a **subproblem**

Master algorithm:

1. Initialize the set $\mathcal{X} \leftarrow \emptyset$
2. Find shortest paths for all OD pairs.
3. Form the all-or-nothing assignment $\mathbf{x}^*$ based on shortest paths.
4. If $\mathbf{x}^*$ is already in $\mathcal{X}$, stop.
5. Add $\mathbf{x}^*$ to $\mathcal{X}$.
6. **Subproblem:** Find a restricted equilibrium $\mathbf{x}$ using only the vectors in $\mathcal{X}$.
7. Return to step 2.

Assuming we can solve the subproblem, this algorithm is guaranteed to converge. Why?

There are only finitely many all-or-nothing assignments.

After each iteration, we add another vector to $\mathcal{X}$.

Eventually, we'll have them all and the algorithm will terminate.

Furthermore, when the algorithm terminates, we have found the equilibrium. Why?

If the shortest paths with respect to **x** are already in $\mathcal{X}$, then the restricted equilibrium is also an unrestricted equilibrium.

So, everything hinges on the subproblem: Find a solution to the variational inequality when the feasible region is restricted to combinations of vectors in $\mathcal{X}$.

We can use Smith's algorithm to solve the subproblem.

Note: Smith's algorithm works whenever the link costs are monotone. This holds when there are no link interactions, so it can be used for the basic traffic assignment problem as well.

To see how close we are to solving the restricted variational inequality, define the *Smith gap* as follows:

$$\gamma_S = \sum_{\mathbf{x_i^*} \in \mathcal{X}} \left([\mathbf{t}(\mathbf{x}) \cdot (\mathbf{x} - \mathbf{x_i^*})]^+ \right)^2$$

This is similar to other gap measures (like γ or AEC) in that it is zero only if we are at restricted equilibrium, and positive otherwise. The Smith gap γ_S is helpful for proving convergence of Smith's algorithm.

Smith's algorithm

Given a solution $\mathbf{x}$, improve it in the following manner:

1. Find an improvement direction $\Delta\mathbf{x}$
2. Update $\mathbf{x} \leftarrow \mathbf{x} + \mu\Delta\mathbf{x}$, where μ is chosen such that γ_S is smaller after the update.

If μ is small enough, the new $\mathbf{x}$ will be feasible and γ_S will be lower. Often we test different values of μ until we find a "successful" one, e.g., 1, 1/2, 1/4, etc.

How do we find the improvement direction?

$$\Delta \mathbf{x} = \frac{\sum_{\mathbf{x}_i^* \in \mathcal{X}} \left[\mathbf{t}(\mathbf{x}) \cdot (\mathbf{x} - \mathbf{x}_i^*)\right]^+ (\mathbf{x}_i^* - \mathbf{x})}{\sum_{\mathbf{x}_i^* \in \mathcal{X}} \left[\mathbf{t}(\mathbf{x}) \cdot (\mathbf{x} - \mathbf{x}_i^*)\right]^+}$$

is an improvement direction; a small enough step will reduce γ_S.

Proof: Smith, 1983.

How many improvement directions do we take?

At some point, we stop finding improvement directions with the current set $\mathcal{X}$, then return to the master algorithm to add another vector to $\mathcal{X}$.

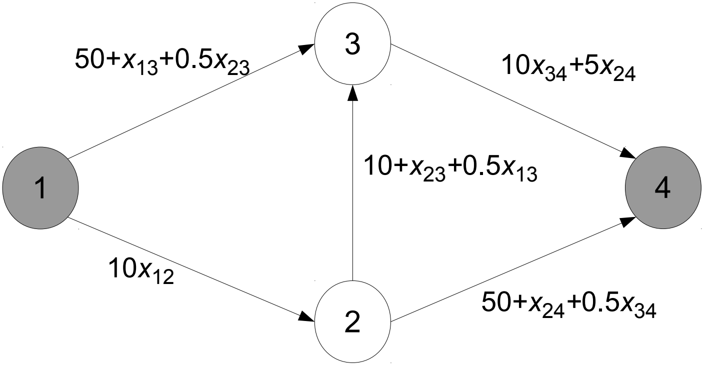
We'll stop when $\mathbf{x}$ is "close enough" to being a restricted equilibrium (based on γ_S).

Deciding when to stop the subproblem is a bit of an art. Often it doesn't pay to solve the subproblem to a high level of accuracy when $\mathcal{X}$ is still small

Master algorithm:

1. Initialize the set $\mathcal{X} \leftarrow \emptyset$
2. Find shortest paths for all OD pairs.
3. Form the all-or-nothing assignment $\mathbf{x}^*$ based on shortest paths.
4. If $\mathbf{x}^*$ is already in $\mathcal{X}$, stop.
5. Add $\mathbf{x}^*$ to $\mathcal{X}$.
6. **Subproblem:** Find a restricted equilibrium $\mathbf{x}$ using only the matrices in $\mathcal{X}$.
    1. Find the improvement direction $\Delta\mathbf{x}$
    2. Update $\mathbf{x} \leftarrow \mathbf{x} + \mu\Delta\mathbf{x}$, with μ sufficiently small (to reduce γ_S).
    3. Return to step 1 of subproblem unless γ_S is small enough.
7. Return to step 2.

Example



Demand is 6.