# Initializing the Traffic Assignment Problem by Zone Aggregation and Disaggregation

**Corresponding author**
Keith Ruddell
Department of Engineering Science
The University of Auckland
Private Bag 92019
Auckland 1142
New Zealand
tel. +64 21 303 558
krud006@aucklanduni.ac.nz

Andrea Raith
Department of Engineering Science
The University of Auckland
Private Bag 92019
Auckland 1142
New Zealand
tel. +64 9 373 7599 ext 81977
a.raith@auckland.ac.nz

13 November 2013

**Abstract**

The traffic assignment problem (TAP) is a key element of many urban transport models. Given trip demand in the form of a matrix indexed by origin and destination, TAP assigns flow to the traffic network to achieve a user equilibrium where no individual traveler may reduce their trip cost by changing route. Many iterative algorithms have been developed to solve TAP; these are usually initialised by all-or-nothing assignment where all flow is assigned to shortest paths, disregarding congestion.

We develop a new method of initialization based on the solution of a simplified traffic assignment problem formed by aggregating origin and destination nodes. This simplified problem can be solved in a fraction of the time needed for the original problem. The user equilibrium of this aggregated problem can then be mapped onto the space of path flows of the original traffic assignment problem. With a suitable choice of aggregation the time required for an iterative TAP algorithm to converge can be greatly reduced. Including the time taken for initialization, our method gives convergence in some cases more than twice as fast as from all-or-nothing initialization. We also propose methods for the automatic aggregation of origins and destinations.

Our results show that the performance of path-based TAP algorithms is strongly dependent on the choice of initial solution.

The traffic assignment problem (TAP) is a central component in regional transport planning models. Its solution is a traffic equilibrium. This is meant to represent the aggregate behaviour of economically rational travellers seeking the 'shortest' path (in the sense of travel time or some combination of travel time and other costs) from origin to destination. The user equilibrium principle (Wardrop's condition) states that no traveler may reduce their journey time by changing route. Equivalently, all routes between a given origin and destination that are in use at equilibrium are equal-shortest.

In evaluation frameworks such as the four-stage model *(1)*, traffic assignment is one of the more computationally demanding stages. Hence it is of practical interest how to quickly compute equilibrium solutions.

As solving the traffic assignment problem to high precision is computationally demanding, techniques have been developed to simplify the problem while preserving the accuracy of certain outputs. Barton and Hearn *(2)* reported on a number of *ad hoc* network simplification strategies in use in the 1970s. Chan (*3*,*4*) proposed a method of aggregation over nodes and links that aims to preserve travel times between origin and destination nodes. His methods give coarse approximations to traffic assignment with less computational cost. Hearn's *(5)* 'transfer decomposition' takes a connected component of the network and constructs two traffic assignment subproblems for traffic internal and external to that component. The full problem is solved in a similar way to Benders decomposition, equilibrating the internal and external problems in turn.

Our own 'centroid aggregation' (CA) method goes beyond these earlier simplification techniques as it includes a disaggregation procedure which outputs a feasible solution in the original (full) traffic assignment problem. This initial feasible solution, when used to hot-start a path based equilibration solver, can lead to a significant reduction in the time required to solve TAP when compared to the standard all-or-nothing initial assignment.

The central idea of the centroid aggregation method is that reducing the number of zones (origins and destinations) has a *very* large effect on the computational effort required to solve TAP. Grouping together zones with $\gamma$ zones per group should give a simplified problem that solves in approximately $\frac{1}{\gamma^2}$ the time; as there are $\frac{1}{\gamma^2}$ as many origin-destination pairs, there are only $\frac{1}{\gamma^2}$ as many shortest path calculations to carry out. So we can quickly compute an equilibrium solution on the aggregated network which should map back to a good (though not equilibrated) traffic assignment on the detailed network.

After defining TAP in our own notation, we present the centroid aggregation method — both its aggregation and disaggregation procedures. We then briefly examine the problem of choosing, for a given network, a suitable aggregation of zone centroids, as well as presenting the results of some tests on real world instances of TAP.

This paper is a development of initial work presented at the 2012 ORSNZ conference *(6)*.

## THE TRAFFIC ASSIGNMENT PROBLEM

Our formulation of TAP is adapted from Patriksson's *(7)* book. Let $\mathcal{N}$ denote the set of nodes in our network, and $\mathcal{A} \subset \mathcal{N}^2$ its set of *arcs* or *links*.

In a traffic network, traffic flows to and from a special set of non-traversable nodes called *zone centroids*, denoted $\mathcal{Z}$. We will write $n$ for the number of nodes, $m$ for the number of arcs and $z$ for the number of zones. The *demand matrix* or *trip table* is a $z \times z$ matrix $\mathbf{D}$ with non-negative entries.

A *route* or *path* between two nodes $u$ and $v$ is a finite sequence of arcs where the from-node of the first arc is $u$, the to-node of each arc is the from-node of the next arc and the to-node of the last arc is $v$. The set of routes from $p \in \mathcal{Z}$ to $q \in \mathcal{Z}$ is $\mathcal{R}_{pq}$. We define the set of all routes $\mathcal{R} := \bigcup_{(p,q)\in\mathcal{Z}^2} \mathcal{R}_{pq}$. The *flow* along any route $r$ is $h_r$.

The *arc cost* functions (also known as volume-delay or link-flow functions) $t_a : \mathbb{R}^+ \to \mathbb{R}^+$ are positive, non-decreasing and convex for all $a \in \mathcal{A}$. These functions may represent travel time, cost or a combination of these.

A *traffic network* is a tuple $(\mathcal{N}, \mathcal{Z}, \mathcal{A}, \boldsymbol{t})$, consisting of a node set, zone set, arc set and set of arc cost functions.

The variables in our formulation are the route flows $h_r$. We write $\boldsymbol{h}$ for the vector of all route flows. We define the vector of link flows $\boldsymbol{f}$ by

$$f_a(\boldsymbol{h}) := \sum_{\{r\in\mathcal{R}:a\in r\}} h_r \ \ \forall a \in \mathcal{A}. \qquad [1]$$

That is, the flow along an arc is the sum of all route flows over all routes that traverse that arc.

If we have a traffic network and a non-negative vector $\boldsymbol{h}$ of route flows then path lengths as a function of path flows are defined by $c_r(\boldsymbol{h}) := \sum_{a\in r} t_a(f_a(\boldsymbol{h}))$. We write $\pi_{pq}(\boldsymbol{h})$ for the length of the shortest path from $p$ to $q$.

Using the notation we have just defined, we may state Wardrop's user equilibrium condition as the following: for all origin-destination pairs $(p, q)$ and all routes $r \in \mathcal{R}_{pq}$, if $h_r > 0$, then $c_r(\boldsymbol{h}) = \pi_{pq}(\boldsymbol{h})$. Alternatively,

$$h_r(c_r(\boldsymbol{h}) - \pi_{pq}(\boldsymbol{h})) = 0 \ \ \forall r \in \mathcal{R}_{pq}, \ \forall (p, q) \in \mathcal{Z}^2. \qquad [2]$$

PROBLEM 1 (Traffic Assignment Problem (TAP)). Given a traffic network $(\mathcal{N}, \mathcal{Z}, \mathcal{A}, \boldsymbol{t})$ and a demand matrix $\mathbf{D}$, find a path flow vector $\boldsymbol{h}$ that satisfies [2] together with the feasibility constraints

$$\sum_{r\in\mathcal{R}_{pq}} h_r = \mathbf{D}_{p,q} \qquad \forall (p, q) \in \mathcal{Z}^2 \text{ and} \qquad [3]$$

$$h_r \geq 0 \qquad \forall r \in \mathcal{R}. \qquad [4]$$

In words, the demand for travel between each origin-destination pair must be equal to the sum of route flows over all routes connecting that origin-destination pair [3], and the route flows are all nonnegative [4].

The equilibrium point, expressed as route flows, is in general not unique. However, with the additional assumption that the network is strongly connected, the equilibrium travel times $\pi_{pq}(\boldsymbol{h})$ are unique. Under the further assumption that the arc cost functions are strictly increasing, we get unique link flows at equilibrium *(7)*.

**Solving TAP**

The purpose of this paper is not to compare different TAP algorithms but to demonstrate an alternative method of initialization for existing algorithms. We test CA with two path based algorithms, although we see no reason why it could not be adapted for origin-based algorithms. See *(8)* for a fuller list of contemporary TAP algorithms.

Path-based equilibration uses column generation to overcome the immense number of possible routes in a traffic network. We denote the set of active routes, those with non-zero flow, between $p$ and $q$ by $\overline{\mathcal{R}}_{pq}$. The left-hand side of [2] — called *excess cost* of the route $r$ — is reduced at every iteration of a path-based algorithm.

We tested our initialization method with two path-based equilibration algorithms. Path equilibration *(9, 10)* shifts flow from the longest active route to the shortest current route between each origin-destination pair in turn. The projected gradient algorithm *(11)* redistributes flow among all active paths between a given origin and destination according to a local line search.

As our measure of convergence, we use relative gap, defined by

$$\mathrm{Rgap}(\boldsymbol{h}) := 1 - \frac{\sum\limits_{(p,q)\in\mathcal{Z}^2} \pi_{pq}(\boldsymbol{h}) \cdot \mathbf{D}_{p,q}}{\sum\limits_{r\in\mathcal{R}} c_r(\boldsymbol{h}) \cdot h_r}, \qquad [5]$$

in accordance with current practice *(11, 12)*. It has several advantages over other convergence measures *(13)*. In particular, a given $\mathrm{Rgap}$ value represents the same level of accuracy across different networks, demand scenarios and TAP algorithms.

## THE CENTROID AGGREGATION METHOD

It is easily recognised that in order for traffic assignment models to be tractable they must use a simplified representation of the real transport infrastructure; we simply cannot take into account every piece of data when modelling route choice. An obvious way of simplifying is the use of zones as origins and destinations. Zone centroids stand in for the many and varied origins and destinations spread across a whole district. In practice the size of zones is determined by how detailed a survey is possible in the study area, rather than by computational concerns *(1)*.

The proposed centroid aggregation (CA) method works by first solving a simplified traffic assignment problem with some zone centroids grouped together — all zones in a group treated as part of a larger *super-zone*. The aggregation function $\alpha$ determines which zones are grouped together. Under this grouping, we obtain a simplified network $(\hat{\mathcal{N}}, \hat{\mathcal{Z}}, \hat{\mathcal{A}}, \hat{\boldsymbol{t}})$ and trip matrix $\hat{\mathbf{D}}$, by a process described below in section 2.1.

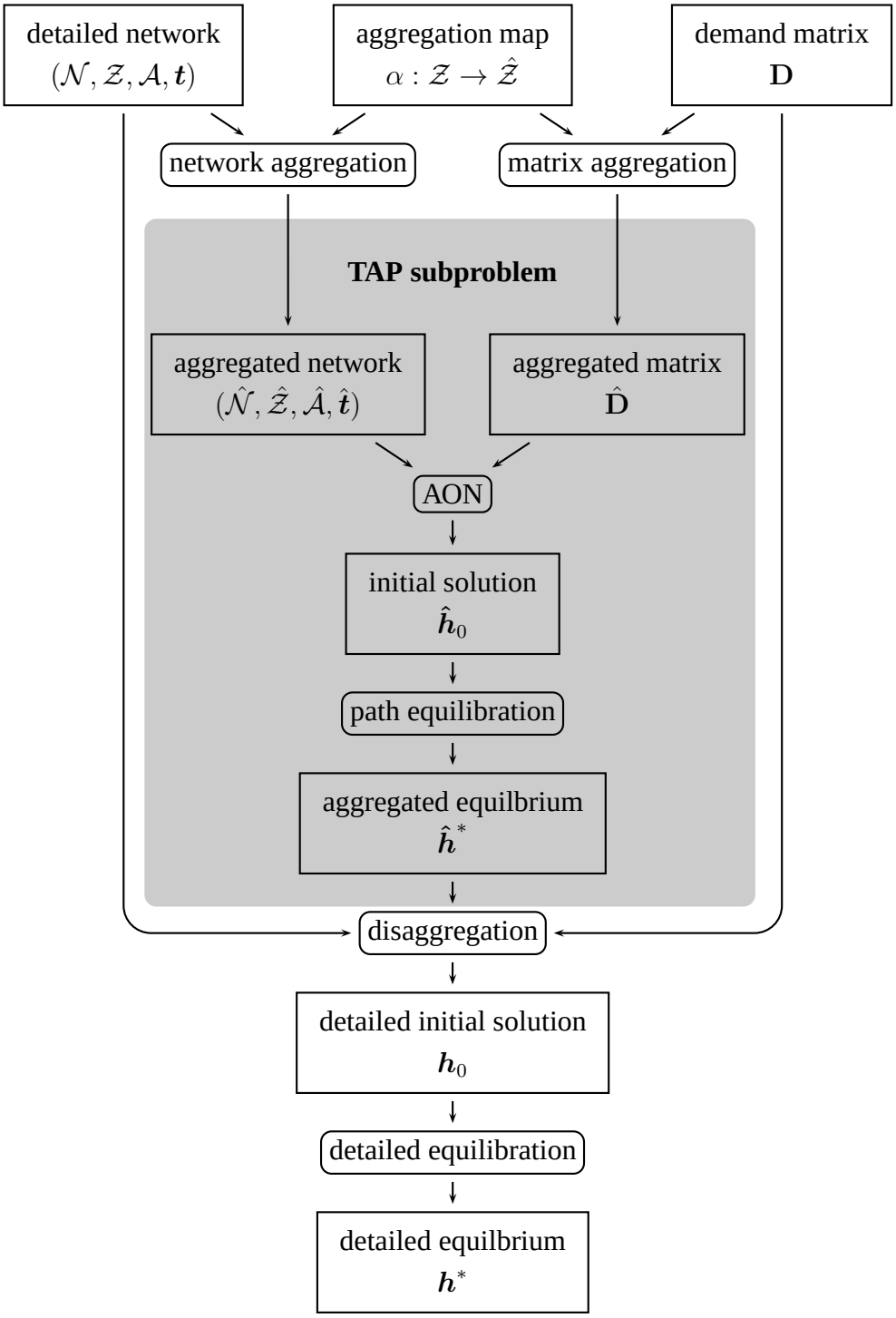



FIGURE 1: Flow diagram of CA method.

To find an equilibrium solution in the aggregated network we use all-or-nothing assignment for an initial solution followed by path equilibration, though any method that quickly returns a path flow equilibrium will do. For the purposes of mapping the solution back to the full network, we prefer a method giving a path flow solution, as opposed to a link flow solution. We use path equilibration. Finally, a solution in the full network is derived from the aggregated problem solution. This last step is called disaggregation.

Figure 1 shows the method in flow diagram form.

**Aggregation**

We describe the grouping of nodes with an aggregation function. The set of aggregated zones is denoted $\hat{\mathcal{Z}}$. We require that $|\hat{\mathcal{Z}}| < |\mathcal{Z}|$. The *aggregation function* is a surjective function $\alpha : \mathcal{Z} \to \hat{\mathcal{Z}}$. Two nodes $p$ and $q$ are grouped together if and only if $\alpha(p) = \alpha(q)$. Its inverse map, defined by

$$\alpha^{-1}(s) := \{p \in \mathcal{Z} : \alpha(p) = s\} \quad [6]$$

will be useful in defining the disaggregation procedure. We could also define the grouping in terms of a partition, as $\{\alpha^{-1}s : s \in \mathcal{Z}\}$ is a partition of $\mathcal{Z}$.

For notational convenience, we assume that the function $\alpha$ is extended to a function $\mathcal{N} \to \hat{\mathcal{N}}$, being the identity on $\mathcal{N} \setminus \mathcal{Z}$. The induced map on arcs $(u, v) \mapsto (\alpha(u), \alpha(v))$ is not necessarily injective, so we allow multiple parallel arcs in the aggregated network. The aggregated arc cost functions $\hat{\boldsymbol{t}}$ are inherited through this induced map on arcs: $t_{(u,v)} \mapsto \hat{t}_{(\alpha(u),\alpha(v))}$.

The demand matrix for the aggregated problem is computed by

$$\hat{\mathbf{D}}_{\hat{p},\hat{q}} := \sum_{\substack{p \in \alpha^{-1}(\hat{p}) \\ q \in \alpha^{-1}(\hat{q})}} \mathbf{D}_{p,q} \quad \forall (\hat{p}, \hat{q}) \in \hat{\mathcal{Z}}^2. \quad [7]$$

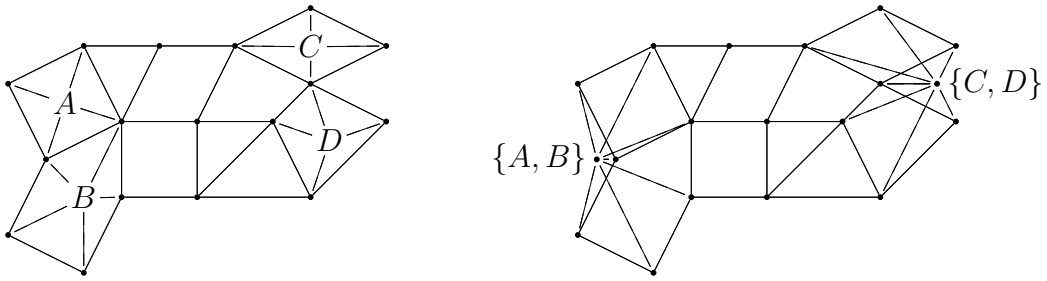
We can then define the aggregated TAP on the network $(\hat{\mathcal{N}}, \hat{\mathcal{Z}}, \hat{\mathcal{A}}, \hat{\boldsymbol{t}})$ with trip matrix $\hat{\mathbf{D}}$, and find a path flow equilibrium $\hat{\boldsymbol{h}}^*$ using a path based solver like path equilibration.

By way of example, figure 2*(a)* shows a network with four zones. Centroid connectors are shown as dashed lines. In figure 2*(b)*, the zones have been aggregated under the partition $\{\{A, B\}, \{C, D\}\}$.

**The centroid partitioning problem**

CA takes the aggregation function $\alpha$ as an input to the solution process. To endogenize the aggregation function, we need a way of automatically computing a good partition of the zone set $\mathcal{Z}$.

PROBLEM 2 (Centroid partitioning problem). Given a traffic network $(\mathcal{N}, \mathcal{Z}, \mathcal{A}, \boldsymbol{t})$ and associated trip table $\mathbf{D}$, find an aggregation map $\alpha : \mathcal{Z} \to \hat{\mathcal{Z}}$ to reduce the computation time required to equilibrate the traffic assignment problem using centroid aggregation.




*(a)* Network with four centroids *(b)* Aggregated network with two superzones

FIGURE 2: Zone aggregation

Problems of partitioning networks and metric spaces arise in many areas: data clustering for statistical analysis, mesh partitioning for parallel computing, facility location problems, *etc*. For a detailed account of how several such methods may be applied to the centroid partitioning problem for TAP, see Ruddell *(14)*. These methods include a modified $k$-means algorithm, data clustering based on a range of metrics and repeated minimum-cuts.

Here we discuss a greedy distance-based heuristic and the spectral partitioning method. These two methods showed greatest promise of all those tested.

Overall, we wish to minimise the computational effort needed to solve TAP. Of course we cannot hope to optimize for computation time over all the possible partitions of the zone set. We must introduce intermediate objectives — properties of an aggregation function that make it likely to be effective.

*Greedy approach*

The first approach we tested was a simple greedy algorithm:

**Algorithm 1** Greedy Aggregator Algorithm

**Input:** traffic network $(\mathcal{N}, \mathcal{Z}, \mathcal{A}, \boldsymbol{t})$, zone aggregation diameter $\rho$.
Set $\hat{\mathcal{Z}} = \mathcal{Z}$, the list of unassigned zones
**while** $\hat{\mathcal{Z}}$ non-empty **do**
  Choose a centre $z_0 \in \hat{\mathcal{Z}}$, place it in a new grouping $\mathcal{Z}_0$ and remove it from $\hat{\mathcal{Z}}$.
  **for** all $z \in \hat{\mathcal{Z}}$ **do**
    **if** $\mathrm{d}(z, z_0) \leq \rho$ **then**
      Put $z$ in $\mathcal{Z}_0$ and remove it from $\hat{\mathcal{Z}}$.

For each super-zone, this algorithm starts with a centre zone and then adds ungrouped zones that are within a distance $\rho$ of those already added. The distance measure can be free-flow travel time, on-the-ground distance or number of links, but in our case we use the free-flow travel time.

*Spectral Partitioning*

The traffic network may be treated as a graph by disregarding the directions of the arcs. The graph Laplacian $\mathbf{L} = \mathbf{A} - \mathbf{D}$ where $\mathbf{A}$ is the adjacency matrix and $\mathbf{D}$ is the diagonal degree matrix. As the graph Laplacian is singular, it has zero as an eigenvalue.

Spectral partitioning is due to Fiedler *(15)* and essentially consists of the computation of the smallest non-zero eigenvalue of the graph Laplacian and one of its eigenvectors. The projection of this eigenvector onto the set $\{-1, 1\}^n$ gives a partition of $\mathcal{N}$ into two subsets. See Newman *(16)* for a fuller explanation. The METIS package *(17)* uses repeated spectral partitioning to supply a starting $k$-way cut which it then improves using the Kernighan-Lin algorithm.

The Kernighan-Lin algorithm *(18)* is also covered well in Newman's *(16)* book. It is a hill-climbing heuristic for improving the cut-value of a $k$-way balanced cut. It works by examining all pairs of nodes $(u, v)$ where $u$ and $v$ are on opposite sides of the cut, and considering the change in the cut-value if $u$ and $v$ were swapped. At each iteration it swaps a sequence of nodes that improves the cut value.

**Disaggregation**

Once we have solved TAP on the aggregated network $(\hat{\mathcal{N}}, \hat{\mathcal{Z}}, \hat{\mathcal{A}}, \hat{\boldsymbol{t}})$, the next step is to derive a detailed path flow solution $\boldsymbol{h}_0$ from the aggregated equilibrium solution $\hat{\boldsymbol{h}}^*$.

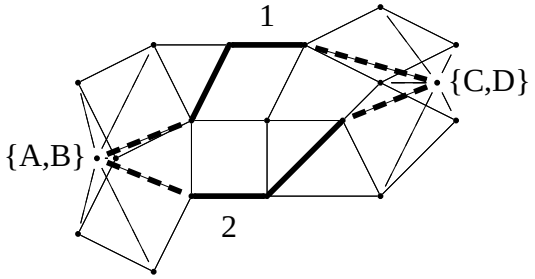
We refer to the problem of dividing the flow from one super-zone to another among the different origin-destination pairs as *untangling*. It can be expressed as an instance of the transportation problem *(19)*. The transportation problem finds the cheapest way to transport commodities from sources to sinks. It assumes that the cost of transportation from a given source to a given sink is linear in the amount transported and independent of all other variables.

The sources are the routes in the simplified solution $\hat{r} \in \overline{\mathcal{R}}_{\hat{p}\hat{q}}$, supplying the flow that they carry. The sinks are the origin-destination pairs with origin in the origin super-zone and destination in the destination super-zone: $(p, q) \in \alpha^{-1}(\hat{p}) \times \alpha^{-1}(\hat{q})$. We define the cost $c_r$ of using route $r$ to get from $p$ to $q$ as the distance from $p$ to the first non-zone node along $r$ plus the distance from the last non-zone node to $q$.

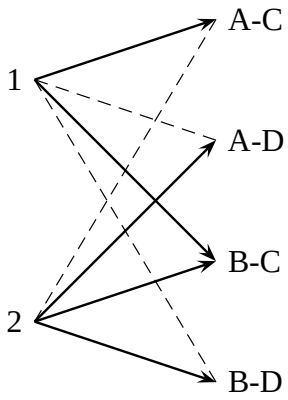
Figure 3*(a)* shows the active paths in the equilibrium solution of the simplified network of figure 2. Figure 3*(b)* shows non-zero allocations of a basic feasible solution of the transportation problem (in the form of a tree) and figure 3*(c)* shows the resulting untangled routes in the original network.

Once we have applied the untangling procedure to every aggregated origin-destination pair, the only unsatisfied demands are between zones grouped in the same super-zone. This traffic does not appear in the aggregated network, so needs to be assigned *ex nihilo* — we use all-or-nothing assignment.

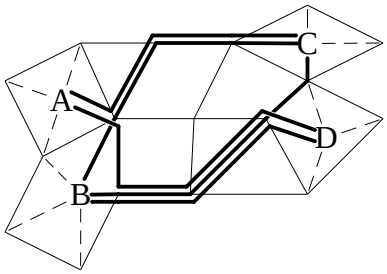
Implicit in the untangling process is a correspondence between paths in the aggregated




*(a)* Active routes in equilibrium from $\{A, B\}$ to $\{C, D\}$

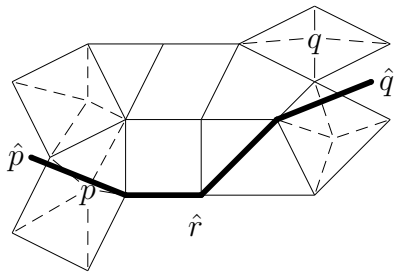



*(b)* Transportation subproblem optimal allocation

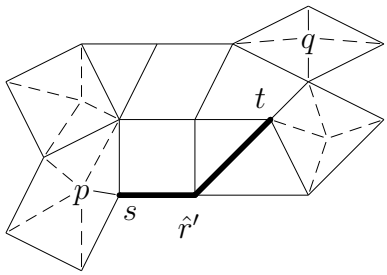



*(c)* Active routes between untangled OD pairs

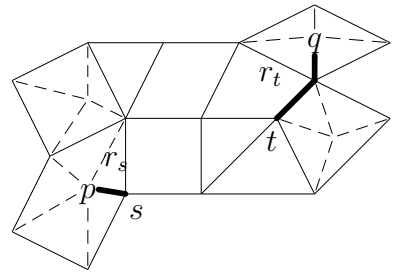
FIGURE 3: Allocating flow to OD pairs and resulting routes in the full network



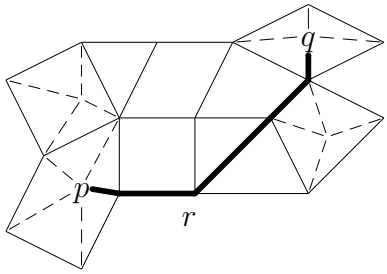

*(a)* Path from aggregated solution $\hat{r}$




*(b)* Top- and tailed: $\hat{r}'$




*(c)* Paths $r_s$ and $r_t$ between origin and main route, and main route and destination




*(d)* The full path $r$

FIGURE 4: Transferring a path from the aggregated to the full network

network and those in the full network. This is similar to how a human might navigate the traffic network, first choosing a route along major roads and then solving the more local problem of how to get on and off the major roads at either end of the journey. We already know that the aggregated arc set differs from the detailed arc set only at centroid connectors, so we could start by mapping each arc in the reduced route to the full arc from which it derives. This produces a path connecting one of the zones mapping to the aggregated origin zones to one of the zones mapping to the aggregated destination zone. The problem is that these might not be the zones that we wish to connect.

The full path is derived by first ignoring the first and last arcs of the path $\hat{r}$ in the aggregated network, to give the 'main route' part of the full path $\hat{r}'$. The shortest paths from the origin $p$ to the start $s$ of $\hat{r}'$ and from the end $t$ of $\hat{r}'$ to the destination $q$ are found and chained together with $\hat{r}'$ to give the full path $r$. It may be that there are loops — nodes visited more than once — in the path $r$ thus derived, but they are simple to remove.

Figure 4 illustrates how the full path is derived from the aggregated-network path. Figure 4*(a)* shows the path $\hat{r}$ between super-zones $\hat{p}$ and $\hat{q}$. Figure 4*(b)* shows $\hat{r}'$, the result of removing the

centroid connectors from $\hat{r}$. The upstream and downstream endpoints $s$ and $t$ are marked. Figure 4*(c)* shows the shortest paths $p \to s$ and $t \to q$, and figure 4*(d)* shows the three parts put together to form $r$.

From our untangled solution an iterative TAP algorithm should converge much quicker than from a standard all-or-nothing initial solution.

Of all the steps in aggregation and disaggregation, the most complex is the derivation of the full path between two zones given the super-zone to super-zone path taken from the aggregated solution. This is due to the use of the shortest path routine, which is called twice for every origin-destination pair. Using an appropriate implementation of Dijkstra's shortest path algorithm *(20)*, we arrive at an overall complexity for the aggregation and disaggregation methods of CA of $O(z^2 m \log n)$, which is of the same order as path based TAP algorithms such as path equilibration *(14)*. Therefore CA should scale no worse than path based algorithms for large instances of TAP.

## RESULTS

We tested our algorithm on three medium-sized networks from the website of Bar-Gera *(21)*, representing Barcelona, Chicago and Winnipeg. Table 1 shows some statistics of the test problems. The computer used had an Intel® Core™2 Duo CPU E8400 @ 3.00GHz $\times$ 2 processor with 4GB RAM running Ubuntu 12.04. All algorithms were implemented in C++. Our program automates the entire Centroid Aggregation method, including partitioning of the network, aggregation and disaggregation.

The baseline, table 2, is all-or-nothing initialization (AON), against which we tested each partitioning method across a range of coarseness parameters. The more zones in each super-zone the coarser the aggregation. For the greedy aggregator the coarseness parameter is the diameter $\rho$. The unit of measurement for $\rho$ is minutes as all distances are represented, through the arc cost functions, as times. The distances between neighbouring zones differs greatly across our example networks. For the spectral partitioning algorithm the coarseness parameter is the mean super-zone size, the average number of zones per super-zone.

For each network and each path-based algorithm we present the time required to reach an equilibrium (to within a relative gap of $10^{-6}$) after AON initialisation as well as the percentage reduction in time taken to reach the same relative gap using centroid aggregation to initialise across a range of parameters for the partitioning methods. The times include generation of the partition, aggregation of the network, equilibration of the simplified problem (only to relative gap $< 10^{-5}$), disaggregation and convergence of the relevant algorithm on the full network.

Equilibration times using CA with the greedy aggregator are shown in table 3. Note the massive reduction in CPU time for Winnipeg with a 9min diameter on path equilibration. Figure 5 shows the progression of relative gap over time for this case. Rather than starting with a small relative gap, the CA-initialised run stars with a larger relative gap but converges at a faster rate and more steadily than the run initialised by all-or-nothing assignment. Figure 6 shows the relative

TABLE 1: Characteristics of test networks

| Network | Nodes | Links | Zones | OD pairs |
|---|---|---|---|---|
| Barcelona | 929 | 2522 | 110 | 7922 |
| Chicago | 933 | 2950 | 387 | 93135 |
| Winnipeg | 1040 | 2836 | 147 | 4344 |

TABLE 2: Time (s) to converge to relative gap $< 10^{-6}$ with all-or-nothing initialisation

| Network | PE with AON | PG with AON |
|---|---|---|
| Barcelona | 22.4 | 31.2 |
| Chicago | 180 | 320 |
| Winnipeg | 71.0 | 100 |

gap over time for Barcelona with a 9min diameter on projected gradient, the instance that took the longest relative to AON iniitialisation. We see that even in this worst case the rate of convergence is very similar under CA and AON. The 10min diameter gave good results for Barcelona, Chicago and Winnipeg; the mean number of zones per super-zones for these partitions were 10.0, 2.63 and 4.08 respectively. This suggests that coarseness is not the only important statistic of an effective aggregation map.

Using METIS, we tested spectral partitioning, refined by the Kernighan-Lin algorithm. In table 4, we see that this gave very good results for Barcelona under the path equilibration algorithm. Also, CA with spectral aggregation maps gave a reduction in computation time for Winnipeg and Chicago at all levels of coarseness.

Overall we sometimes see significant improvement in run time and even in the worst cases, the increase in time required is never more than 20%. Our disaggregation mapping belongs to a family of methods for projecting equilibrated solutions of similar traffic assignment problems onto good feasible starting points for equilibration. What we have shown is that hot-starting can be a time-saver even when a related TAP problem is formed, equilibrated and projected onto the full network for the sole purpose of hot-starting; the time required by these computations is outweighed by the time saved in reaching equlibrium in the full network.

## CONCLUSION

We have proposed a new method for initializing iterative TAP solvers and described how it works by aggregating origin and destination zones to form a simplified traffic assignment problem whose solution may be mapped to an initial feasible solution of the original problem. Our tests show that the idea of using the solution of a simplified TAP problem to initialise a TAP solver has merit. In

TABLE 3: Percentage reduction (increase) in time to converge to relative gap $< 10^{-6}$ with greedy aggregator

| | Diameter (min) | | | | |
|---|---|---|---|---|---|
| | 8 | 9 | 10 | 11 | 12 |
| | Path equilibration (%) | | | | |
| Barcelona | 18 | 7 | 17 | (8) | 17 |
| Chicago | 14 | 8 | 29 | 24 | 7 |
| Winnipeg | 3 | 71 | 32 | 1 | 29 |
| | Projected gradient (%) | | | | |
| Barcelona | 3 | (16) | 8 | 6 | (1) |
| Chicago | 11 | (2) | 18 | 14 | 1 |
| Winnipeg | 5 | 23 | 25 | (3) | 1 |

TABLE 4: Percentage reduction (increase) in time to converge to relative gap $< 10^{-6}$ with Spectral/K.-L. aggregation by METIS

| | Mean super-zone size | | | | | |
|---|---|---|---|---|---|---|
| | 2 | 3 | 4 | 5 | 6 | 7 |
| | Path equilibration (%) | | | | | |
| Barcelona | 4 | 14 | 17 | 16 | 39 | 13 |
| Chicago | 11 | 1 | 14 | 15 | 19 | 20 |
| Winnipeg | 22 | 17 | 41 | 13 | 25 | 30 |
| | Projected gradient (%) | | | | | |
| Barcelona | 0 | 13 | 4 | 5 | 21 | 6 |
| Chicago | 4 | (3) | 1 | 0 | (2) | 2 |
| Winnipeg | 28 | 36 | 49 | 11 | 44 | 5 |

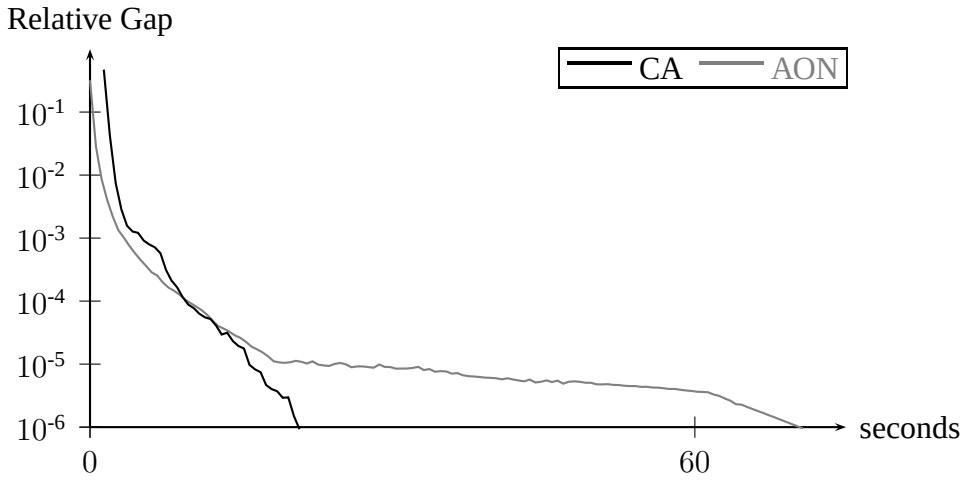



FIGURE 5: Relative gap vs CPU time, for Winnipeg test network, best case

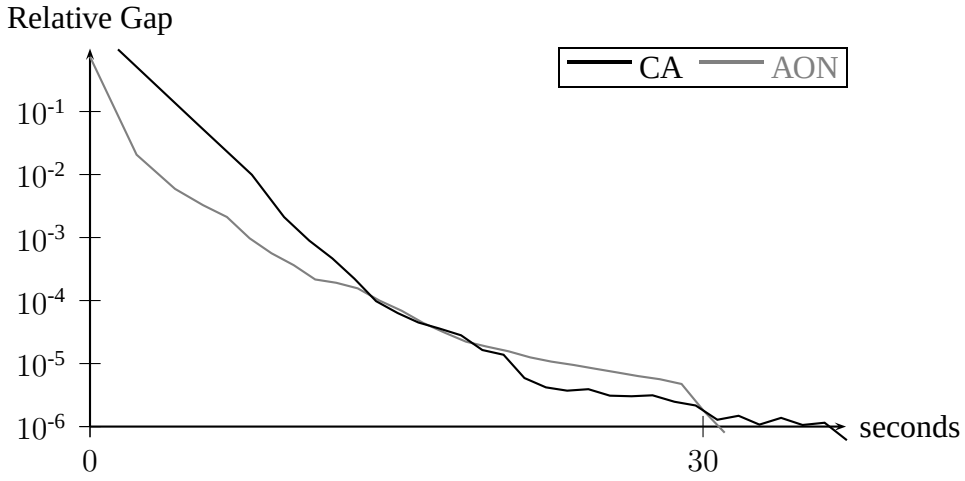



FIGURE 6: Relative gap vs CPU time, for Barecelona test network, worst case

the best cases path equilibration takes less than half the time to converge to a relative gap of $10^{-6}$ than with all-or-nothing initialization. In the worst cases the time taken to converge is no more than 10% longer.

The first conclusion we can draw from our computational tests is that initialization matters for TAP equilibration algorithms. When solving TAP repeatedly on the same network with different trip tables, or with only a few links added or removed; re-starting (cold starting) from AON assignment is probably not the wisest choice. It could be worthwhile to try out some other heuristics for warm-starting.

Though we have shown that an aggregation/disaggregation approach to initializing TAP can be worthwhile, there exist several ways in which the centroid aggregation method could be expanded and improved. We have not yet had the chance to test our centroid aggregation on larger networks, where there are even greater opportunities to speed up equilibration. It could be

adapted to initialize origin-based algorithms as well as path-based algorithms. The way that paths are mapped from simplified to full network could be refined so that the shortest paths are more likely to be active in the disaggregated solution. Also, it is natural to ask why some aggregation functions are more effective at speeding up convergence than other, and whether it is possible to quickluy compute an aggregation function that will guarantee a fast convergence. Finally, it should be possible to adapt our method to the dynamic traffic assignment problem, which is even more computationally demanding than plain TAP.

Overall, the centroid aggregation method is a promising idea for better ways to initialise traffic assignment solvers.

## ACKNOWLEDGMENTS

The authors thank Olga Perederieieva for sharing her implementations of different TAP algorithms. This research was partially supported by Marsden grant 9075 362506.

## REFERENCES

*(1)* Ortúzar, J. d. D. and L. G. Willumsen, *Modelling Transport*. Wiley, New York, 2001.

*(2)* Barton, R. R. and D. Hearn, *Network aggregation in transportation planning models: final report*. U.S. Dept. of Transportation, Research and Special Programs Administration, Washington, DC, 1979.

*(3)* Chan, Y., *Optimal Travel Time Reduction in a Transport Network: An Application of Network Aggregation and Branch and Bound Techniques*. Research report, Transportation Systems Division, M.I.T. Department of Civil Engineering, 1969.

*(4)* Chan, Y., A method to simplify network representation in transportation planning. *Transportation Research*, Vol. 10, No. 3, 1976, pp. 179–191.

*(5)* Hearn, D. W., Practical and theoretical aspects of aggregation problems in transportation planning models. In *Transportation Planning Models*, Elsevier, Amsterdam, 1984, pp. 257–287.

*(6)* Ruddell, K. and A. Raith, An Aggregational Approach to the Traffic Assignment Problem. In *Proceedings of the 46th Annual ORSNZ Conference*, Wellington, New Zealand, 2012.

*(7)* Patriksson, M., *The Traffic Assignment Problem: models and methods*. VSP, Netherlands, 1994.

*(8)* Inoue, S. and T. Maruyama, Computational Experience on Advanced Algorithms for User Equilibrium Traffic Assignment Problem and Its Convergence Error. *Procedia - Social and Behavioral Sciences*, Vol. 43, 2012, pp. 445 – 456.

*(9)* Dafermos, S. C. and F. T. Sparrow, The Traffic Assignment Problem for a General Network. *Journal of the National Bureau of Standards - B*, Vol. 738, No. 2, 1969, pp. 91–118.

*(10)* Leventhal, T., G. Nemhauser, and L. Trotter, A Column Generation Algorithm for Optimal Traffic Assignment. *Transportation Science*, Vol. 7, No. 2, 1973, pp. 168–176.

*(11)* Florian, M., I. Constantin, and D. Florian, A New Look at Projected Gradient Method for Equilibrium Assignment. *Transportation Research Record: Journal of the Transportation Research Board*, Vol. 2090, 2009, pp. 10–16.

*(12)* Bar-Gera, H., Origin-Based Algorithm for the Traffic Assignment Problem. *Transportation Science*, Vol. 36, No. 4, 2002, pp. 398–417.

*(13)* Slavin, H., J. Brandon, and A. Rabinowicz, Empirical Comparison of Alternative Equilibrium Assignment Methods. In *Proceedings of the European Transport Conference*, 2006.

*(14)* Ruddell, K., *Hot-Starting the Traffic Assignment Problem by Zone Aggregation and Disaggregation*. Master's thesis, University of Auckland, 2013.

*(15)* Fiedler, M., Algebraic connectivity of graphs. *Czechoslovak Mathematical Journal*, Vol. 23, No. 98, 1973, pp. 298–305.

*(16)* Newman, M. E. J., *Networks: An Introduction*. Oxford University Press, Oxford; New York, 2010.

*(17)* Karypis, G., *METIS:A Software Package for Partitioning Unstructured Graphs, Partitioning Meshes, and Computing Fill-Reducing Orderings of Sparse Matrices Version 5.1.0*. University of Minnesota, Department of Computer Science & Engineering, Minneapolis, 2013.

*(18)* Kernighan, B. W. and S. Lin, An Efficient Heuristic Procedure for Partitioning Graphs. *The Bell System Technical Journal*, Vol. 49, No. 1, 1970, pp. 291–307.

*(19)* Bertsekas, D. P., *Linear Network Optimization - Algorithms and Codes.* Cambridge, Mass: MIT Press, 1991.

*(20)* Even, S., *Graph Algorithms, 2nd Edition*. Cambridge University Press; Cambridge, New York, New York, NY, USA, 2012.

*(21)* Bar-Gera, H., *Transportation Test Problems*, 2013, `http://www.bgu.ac.il/~bargera/tntp/` Accessed March 15, 2013.