Optimizing Seattle Bus Transport Routes with Graph-Theoretic Methods

Giorgio Micaletto Federico Giorgi Taylor Woodward

Abstract

We present a graph-theoretic framework for redesigning Seattle's bus network informed by Average Daily Traffic (ADT) data. The transportation network is modeled as a weighted graph whose edge weights combine geometric cost, normalized busyness, hub-to-hub connectivity, and estimated service frequency via a normalized multi-objective function. We (i) compute a Minimum Spanning Tree (MST) to obtain a connected backbone of minimal aggregate cost under the chosen weights; (ii) augment the backbone by reinserting strategically chosen redundant edges near high-hub areas; (iii) perform route reduction via a greedy route-merging heuristic with geometric penalties; and (iv) place stops under spacing and accessibility constraints. We formalize assumptions, state and prove key properties (MST existence/optimality, invariance under affine rescaling of weights, correctness of greedy stop spacing on paths, and termination/computational complexity of the merging procedure), and discuss limitations and extensions (temporal dynamics, capacity, and demand modeling). The presentation is rigorous but intentionally readable, reflecting the scope of a project rather than a full monograph.

Contents

1 Introduction and Motivation

Urban congestion imposes significant economic and social costs. Public transit—particularly buses—can mitigate these costs if routes, frequencies, and stop placement align with latent demand and urban geometry. Designing such networks is a multi-criteria optimization problem with competing objectives (coverage, cost, reliability, and accessibility). Graph-theoretic methods offer an explicit mathematical language and algorithmic toolkit to reason about these trade-offs [1, 2].

We study Seattle, WA, leveraging publicly available ADT and geospatial data. Our aim is to produce a principled baseline procedure: a minimal-cost backbone (MST) augmented for robustness, then consolidated into a manageable number of routes with disciplined stop spacing. The framework is modular and

admits richer demand, capacity, and temporal layers.

2 Data and Preprocessing

We assume a cleaned geospatial dataset of road segments with ADT (Average Daily Traffic) measurements. Each record provides segment geometry and daily volume.

Coordinate system. For accurate distance computations, all geometries are projected to a local, metric CRS (e.g., EPSG:26910, UTM Zone 10N).

Normalization. Let ADT_e denote the daily traffic for edge e. We map to a unitless busyness score $b_e \in [0,1]$ via a linear rescaling:

$$b_e = \frac{\mathrm{ADT}_e - \min(\mathrm{ADT})}{\max(\mathrm{ADT}) - \min(\mathrm{ADT})}.$$

3 Notation, Assumptions, and Model

3.1 Graph model and notation

Let $G = (V, E)$ be the (undirected) road graph.[1] For $e = \{u, v\} \in E$:

- $L_e > 0$: segment length, with
$$\bar{L} := \frac{1}{|E|} \sum_{e \in E} L_e.$$
- $b_e \in [0,1]$: normalized busyness.
- $h_v := \sum_{e \sim v} b_e$: hub score at node v; write h_u, h_v for endpoints.
- f_e: estimated (unitless) service frequency proxy on e.

3.2 Assumptions

Assumption 1 (Temporal homogeneity)**.** *Traffic is time-invariant at our modeling resolution: $b_e(t) = b_e$ for all t.*

[1] Directionality can be handled by symmetrization or, if necessary, by moving to arborescences; see Remark 1.

Assumption 2 (Static infrastructure)**.** *The road graph G is fixed and connected; closures/accidents are ignored.*

Assumption 3 (Capacity abstraction)**.** *Vehicle capacities and crowding dynamics are abstracted into f_e (no hard capacity constraints).*

These assumptions yield a tractable baseline. Section 6 discusses relaxations.

3.3 Multi-objective edge weight

We combine geometry, busyness targeting, hub connectivity, and service effort in a normalized weight:

$$w(e) = \alpha \frac{L_e}{\bar{L}} + \beta \phi(b_e) + \gamma \psi(h_u, h_v) + \delta \chi(f_e), \tag{1}$$

with nonnegative policy parameters $\alpha, \beta, \gamma, \delta$. Here $\phi(\cdot)$ denotes a busyness penalty, $\psi(\cdot,\cdot)$ hub connectivity and $\chi(\cdot)$ service effort. We use the concrete choices

$$\phi(b) := |b - b^\star|,$$
$$\psi(h_u, h_v) := -\frac{h_u + h_v}{\bar{h} + \varepsilon},$$
$$\chi(f) := f,$$

where $b^\star \in [0,1]$ is the desired traffic level, $\bar{h}$ is the mean hub score, and $\varepsilon > 0$ is a small stabilizer. We set

$$f_e \;=\; f_0 + S \cdot \frac{b_e + h_u + h_v}{3(\bar{h} + 1)}, \tag{2}$$

so higher demand/centrality increases the frequency proxy. All terms are unitless, enabling (1) to be interpreted as a scalarized multi-objective surrogate.

Proposition 1 (Affine Invariance of MST)**.** *Fix any $a > 0$ and $b \in \mathbb{R}$, and define*

$$w'(e) := a\, w(e) + b$$

for all e. Any MST under w is also an MST under w', and vice versa.

Proposition 2 (Existence and Uniqueness of MST)**.** *If G is connected and $w(e) \in \mathbb{R}$ for all e, an MST exists. If, in addition, all edge weights are distinct, the MST is unique.*

Proposition 3 (Backbone Optimality)**.** *Let T be an MST of (G, w). Then T is connected and has minimal total weight among all spanning subgraphs with $|V| - 1$ edges. Thus T is a minimum-cost* backbone *under* (1)*.*

Proofs are deferred to Appendix A.

Remark 1. *If directionality is essential, replace MST with a minimum spanning* arborescence *rooted at a depot (Edmonds' algorithm). Our use of undirected G is justified for symmetric segments or when we first symmetrize one-way streets into bi-directional cost proxies.*

4 Algorithms

We outline the four stages: construction and weighting, MST backbone, augmentation, and consolidation (merging + stops). Throughout, $n := |V|$ and $m := |E|$.

4.1 Graph construction and weighting

Graph Construction is executed by Algorithm 1.

Algorithm 1 BuildWeightedGraph

1: Input: road segments with geometry and ADT
2: Project to metric CRS; compute L_e
3: Normalize ADT to $b_e \in [0, 1]$
4: Build $G = (V, E)$; compute hub scores $h_v = \sum_{e \sim v} b_e$
5: Compute f_e via (2)
6: Compute $w(e)$ via (1)
7: **return** weighted graph (G, w)

Complexity: $\mathcal{O}(m + n)$ arithmetic plus geodesic length computations.

4.2 MST backbone

We compute an MST T of (G, w) (Kruskal or Prim).

- *Kruskal:* $\mathcal{O}(m \log m)$ (sorting) with near-linear union–find.
- *Prim (binary heap):* $\mathcal{O}(m \log n)$; (Fibonacci heap): $\mathcal{O}(m + n \log n)$.

4.3 Backbone augmentation by hub edges

We add robustness using Algorithm 2 by reinserting a budgeted set of non-tree edges that connect high-hub areas.

Algorithm 2 AugmentBackbone

1: Input: MST T, full graph (G, w), hub scores $\{h_v\}$, budget B
2: Candidate set $\mathcal{C} \leftarrow \{e = \{u, v\} \in E \setminus E(T) : h_u + h_v \text{ high}\}$
3: Score each $e \in \mathcal{C}$ by $\sigma(e) := \lambda_1 \frac{h_u + h_v}{h+1} - \lambda_2 w(e)$
4: Add the top-B edges by σ to T to obtain H
5: **return** augmented backbone H

Complexity: sorting $\mathcal{C}$ is $\mathcal{O}(|\mathcal{C}| \log |\mathcal{C}|) \subseteq \mathcal{O}(m \log m)$.

4.4 Route generation, merging, and stop placement

We treat each path in H with high aggregate demand as a candidate route, then greedily merge routes to reduce redundancy.

Merging objective. For routes r_1, r_2 with endpoints set E_1, E_2, define

$$\begin{aligned} C(r_1, r_2) = \; & \lambda_d \min_{x \in E_1, y \in E_2} d(x, y) \\ & + \lambda_\theta \Theta(r_1, r_2) \\ & + \lambda_L L(r_1 \cup r_2) \\ & + \lambda_R \Pi(r_1, r_2), \end{aligned}$$

where d is endpoint proximity, Θ penalizes sharp turns (average absolute angle), L is length of the merged geometry, and Π penalizes restricted/forbidden edges.

Algorithm 3 GreedyRouteMerging

Input: initial route set $\mathcal{R}_0$ from H, target count R_{target}
$k \leftarrow |\mathcal{R}_0|$, $t \leftarrow 0$, $\mathcal{R} \leftarrow \mathcal{R}_0$
while $k > R_{\text{target}}$ **do**
  Compute $C(r_i, r_j)$ for all unordered pairs in $\mathcal{R}$
  $(i^\star, j^\star) \leftarrow \arg\min_{i<j} C(r_i, r_j)$
  Merge $r_{i^\star}, r_{j^\star}$ into $\widehat{r}$; update $\mathcal{R} \leftarrow \mathcal{R} \setminus \{r_{i^\star}, r_{j^\star}\} \cup \{\widehat{r}\}$
  $k \leftarrow k - 1$, $t \leftarrow t + 1$
end while
return $\mathcal{R}$

Lemma 1 (Termination and Complexity)**.** *Algorithm 3 terminates in $|\mathcal{R}_0| - R_{target}$ iterations. A naive implementation is $\mathcal{O}(|\mathcal{R}|^2)$ per iteration; with a priority queue over pairwise costs and lazy updates, the total runtime is $\mathcal{O}(|\mathcal{R}_0|^2 \log |\mathcal{R}_0|)$ up to geometry evaluation costs.*

Stop placement on routes. Let $\Delta_{\max} > 0$ be the maximum allowed inter-stop distance; let $\epsilon > 0$ be a spatial merge tolerance for near-duplicate stops.

Algorithm 4 GreedyStopSpacing (per route)

Input: route polyline P, spacing $\Delta_{\max}$, merge tolerance ϵ
Initialize stops at mandatory points (endpoints, key junctions)
Traverse P; whenever the distance from the last stop reaches $\Delta_{\max}$, place a new stop
Merge any pair of stops within ϵ
return stop set $S(P)$

Proposition 4 (Feasibility and Minimality on a Path)**.** *On a simple path of length L, Algorithm 4 returns a feasible set with $\max_i d(s_i, s_{i+1}) \le \Delta_{\max}$. Moreover, among all feasible sets that respect endpoints, the greedy placements achieve the minimum cardinality $\lceil L/\Delta_{\max} \rceil + 1$.*

5 Diagnostics and Results (Illustrative)

Let $\widehat{\mathcal{R}}$ be the merged route set and $S = \bigcup_{r \in \widehat{\mathcal{R}}} S(r)$ the stop set.

Backbone size. The MST T uses $|V| - 1$ edges; augmentation adds B edges. Report $|E(T)|, |E(H)|$.

Route consolidation. Initial routes $|\mathcal{R}_0|$ reduced to $|\widehat{\mathcal{R}}| = R_{\text{target}}$. Report average route length, number of turns (proxy: average absolute bearing change), and mean inter-stop distance.

Wait-time proxy. For each edge e, define $\widehat{W}_e = 60/f_e$ (minutes). Summarize $\{\widehat{W}_e\}$ by mean/median and quantiles; visualize as a heatmap over H.

Coverage proxy. Report percent of high-b_e segments within ρ meters of a route/stop.

Note: Replace placeholders with your computed quantities; include your figures (e.g., `initial_routes.png`, `bus_routes.html`, `bus_routes_with_stops.html`) as static/interactive supplements.

6 Limitations and Extensions

Temporal dynamics. Assumption 1 is strong; integrating peak/off-peak slices (or real-time feeds) permits time-varying $b_e(t)$ and $f_e(t)$.

Capacity and crowding. Assumption 3 ignores vehicle capacity, load-dependent dwell, and reliability; add constraints or penalties to $\chi(f_e)$, or formulate a bilevel design–assignment model.

Demand heterogeneity. Demographic and land-use layers (population, jobs, equity) can shape ϕ and f_e.

Geometry realism. Penalize U-turns and excessive curvature in Θ; favor line/loop archetypes.

Directed operations. For one-way streets or asymmetric travel times, use arborescences or strongly connected augmentations (Remark 1).

7 Reproducibility (Data & Code)

Scripts: `main.py`. Data: `SDOT_data.geojson`. Outputs: `bus_routes.html`, `bus_routes_with_stops.html`, `initial_routes.png`. Running the pipeline should (i) build the weighted graph, (ii) produce an MST and augmented backbone, (iii) generate routes and stops, and (iv) export static/interactive visualizations.

References

[1] T. H. Cormen et al. *Introduction to Algorithms*. 3rd. MIT Press, 2009.

[2] L. R. Ford and D. R. Fulkerson. *Flows in Networks*. Princeton University Press, 1962.

A Proofs and Auxiliary Results

Proof of Proposition 1

Let $w'(e) = a\,w(e) + b$ with $a > 0$ constant. For any cut C of G, an edge e has smaller w than e' iff it has smaller w' than e' (strict order preserved by positive affine maps). The cut property characterizing MSTs depends only on the ordering of edges crossing cuts, hence the set of MSTs is identical under w and w'. □

Proof of Proposition 2

Existence follows from standard MST theory for connected graphs with real weights (e.g., Kruskal/Prim construct an MST). If all edge weights are distinct, the cycle/cut properties imply uniqueness. □

Proof of Proposition 3

By definition, an MST is a connected acyclic subgraph with $|V| - 1$ edges and minimum total weight among such subgraphs. This is precisely a minimal-cost backbone under (1). □

Proof of Lemma 1

Each iteration strictly reduces the route count by one, so the algorithm halts after $|\mathcal{R}_0| - R_{\text{target}}$ iterations. Naively recomputing all pairwise costs takes $O(|\mathcal{R}|^2)$ per iteration; maintaining a heap of candidate merges with lazy invalidation yields total $O(|\mathcal{R}_0|^2 \log |\mathcal{R}_0|)$, plus geometry updates. □

Proof of Proposition 4

For feasibility, the greedy rule never allows a gap $> \Delta_{\max}$ by construction. For minimality on a simple path of length L, any feasible set has at least $\lceil L/\Delta_{\max} \rceil + 1$ stops (a packing argument). Greedy achieves this bound by placing a new stop exactly when needed, starting at an endpoint. Merging within ϵ preserves feasibility if merges do not create gaps $> \Delta_{\max}$. □

B Practical Parameterization

- **Scalarization:** Start with $\alpha = 1$ (length baseline), $\beta \in [0.2, 1]$ to target $b^\star$, $\gamma \in [0.1, 0.5]$ to favor hubs (negative term), and $\delta \in [0.1, 0.5]$ to temper frequency costs.
- **Frequency proxy:** Choose f_0 as the minimum acceptable headway mapped to the unitless scale, S as sensitivity to demand.
- **Augmentation budget:** Pick B to obtain empirical 2-edge-connectedness on high-hub subgraphs or to meet a reliability KPI.
- **Merging weights:** Tune $(\lambda_d, \lambda_\theta, \lambda_L, \lambda_R)$ to penalize sharp turns/U-turns and restricted edges; calibrate against a few hand-checked routes.
- **Stops:** Typical $\Delta_{\max} \in [300, 500]$ m; $\epsilon \in [25, 50]$ m for deduplication.