Supplementary material for "Multi-criteria journey planning on the passenger's device: design and validation for fifteen merged regional transit networks"

Yeray Garrido

This document complements the paper with material that does not fit in its pages: the pseudocode of the search and the proof of its first property, the constants and labels of the selection layer, the data format, further results, the detailed results of the comparison with OpenTripPlanner and the plausibility audit of the options. Section, table and algorithm numbers without the prefix S refer to the paper.

S1. Pseudocode of the search

Algorithm S2 gives the range iteration, Algorithm S3 the scan of one route and Algorithm S4 the relaxation of transfers. USEFUL(ℓ, p, u, j_0) is the pruning test of Section IV-D for a label ℓ at stop p after u rides, with at least j_0 more rides, using the alighting bounds after alighting and the boarding bounds before boarding. INSERT is Algorithm S1; every successful insertion of a live label into a bag of round k also inserts its shadow into the bags of the same kind for rounds $k+1, \ldots, K$.

Algorithm S1 Insertion of a label $\ell = (\tau, w)$ into a staircase bag b of capacity B.

if some $m \in b$ has $\tau_m \le \tau$ and $w_m \le w$ **then**
  return rejected
end if
remove from b every m with $\tau_m \ge \tau$ and $w_m \ge w$
insert ℓ keeping b sorted by τ (and decreasing w)
if $B = 1$ and $|b| > 1$ **then**
  remove position 1 ▷ keep the earliest
else if $|b| > B > 1$ **then**
  $i^* \leftarrow \arg\min_{1 \le i \le |b|-2}(w_{i-1} - w_i)$, ties to the largest i
  remove position i^* ▷ never 0 nor $|b| - 1$
end if

S2. Proof of the first property

We prove the first property of Section IV-G for range queries, which have no limit on the wait after a transfer. Plan queries also apply a two-hour limit on that wait; Section S3 discusses why the property can then fail.

a) Setting: The departure times of the window are sorted from latest to earliest and split into *groups* of consecutive times less than one minute apart, processed together; let $1, 2, \ldots$ be the groups in processing order and t_d the earliest time of group d. For $k \le K$ let $E_k(d)$ be the earliest arrival at the target over all journeys that leave the origin at t_d or later within the window and use at most k rides, with the walking and margins of Section III; a journey that leaves at t_d or later leaves in group d or in an earlier-processed one. We assume the FIFO property of every route, also across service days, and that each trip carries one delay, so that a moved trip keeps its order of stops and times. *Claim:* after group d is processed, the target bags $\mathcal{T}_1, \ldots, \mathcal{T}_k$ (with their shadows) contain a label with time $E_k(d)$, created in group d or in an earlier-processed one. The proof covers range queries as defined in Section III: departures within a window, without the deadline of arrive-by queries or the wait limit of plan queries.

b) Facts about the bags: (F1) A label is rejected only if a label of the bag is no later and walks no more, it removes only labels that are no earlier, and eviction never removes the earliest label of a bag (Algorithm S1). Hence, for a stop p, a kind of bag and a round u, the earliest time $m(p, u)$ over the bags of that kind at p in rounds $0, \ldots, u$ never increases during a run, except when the probe's labels are forgotten. (F2) Every label is the end of a journey prefix that leaves the origin in the current group or in an earlier-processed one and uses at most as many rides as the round of its bag; a shadow in round u copies a label of a round below u. (F3) By FIFO, if a prefix ends at p in a bag of some kind no later than another, any continuation of the second is also a continuation of the first and arrives no later; with ready labels this holds because margins are included in the time and depend only on the stop. (F4) A probe runs only while $\mathcal{T}_K$, which holds every target label or its shadow, is empty, so forgetting removes only labels of the current group, before any journey is known.

Algorithm S2 Range iteration over a departure window $[t, t + \Delta]$.

```
1: compute the bounds lb for the target (once per target and set of extra trips)
2: D ← all (τ0, p, τp): leaving the origin at τ0 ∈ [t, t + Δ] catches a departure from access stop p at τp
3: sort D by decreasing τ0; σ ← 0
4: for all groups G ⊆ D of departures less than 60 s apart do
5:     σ ← σ + 1                                   ▷ new stamp; older labels stay and prune
6:     probe ← (T_K is empty)
7:     ROUNDS(G, σ, probe)
8:     if probe and T_K is no longer empty then
9:         remove the labels with stamp σ (their shadows stay); ROUNDS(G, σ, false)
10:    end if
11:    rebuild a journey from each label of T_k with stamp σ; check it against its label
12: end for

13: procedure ROUNDS(G, σ, probe)
14:    for all (τ0, p, τp) ∈ G do
15:        INSERT(R0(p), (τp, access walk))
16:    end for
17:    for k = 1, ..., K while some stop received live labels in round k − 1 do
18:        for all routes r through such stops, from the first such position do
19:            SCANROUTE(r, k)
20:        end for
21:        for all extra trips do
22:            scan it as a moved trip (Algorithm S3, lines 18–21)
23:        end for
24:        if k < K then
25:            for all stops p with live labels in A_k(p) do
26:                TRANSFERS(p, k)
27:            end for
28:        end if
29:    end for
30: end procedure
```

c) The journey to follow: Fix d and k with $E = E_k(d)$ finite, and among the journeys that achieve E choose one, J, with the fewest rides $k^* \le k$. If J leaves in an earlier-processed group d', then $E_k(d') \le E \le E_k(d')$, the claim for d' gives a target label with time E in rounds at most k, and by (F1) and (F4) it is still there. Otherwise J leaves in group d. We follow the last run of group d (the probe, if no second run follows it) and call the target bags *settled* when $\mathcal{T}_1, \ldots, \mathcal{T}_k$ hold a label with time E; once settled they stay so, by (F1) and because nothing is forgotten after the last run starts. Write the events of J as $e_1, e_2, \ldots$: ready to board at a stop, alighted at a stop, and at the target, each with its time t_e and its number of rides so far u_e.

d) Blocking labels: Let L be a label at the stop of an event e with time at most t_e, in a bag of the right kind and of round at most u_e, that is not live. By (F2) and (F3), replacing J's prefix by L's gives a journey that leaves at t_d or later and arrives by E. If L is a shadow, or a probe label's shadow, that journey uses fewer than k^* rides, contradicting the choice of J. If L is not a shadow, it belongs to an earlier-processed group d', since the labels of the current run are live; that journey leaves in d' with at most k^* rides, so $E_k(d') = E$ and, by the claim for d', the target bags are settled. So a label that is not live and is no later than J at one of its events either cannot exist or settles the bags.

e) Pruning: A label (τ, w) offered for an event e, with $\tau \le t_e$ in a round $u \le u_e$, is pruned only if the test of Section IV-D holds for every j, in particular for the number j^* of rides that J still takes after e. By the admissibility proposition and (F3), $\rho_{j^*} = \tau + \mathrm{lb}_{j^*}(p) \le t_e + (E - t_e) = E$, so it is finite, and a range query has no deadline. If a target label of round $u + j^* \le k^*$ has a time $\tau' \le \rho_{j^*} \le E$, it is the end of a journey that leaves at t_d or later with at most k rides, so $\tau' = E$; if it is not a shadow the bags are settled, and if it is one, the journey it copies uses fewer than k^* rides and arrives by E, contradicting the choice of J. Finally, a known arrival more than 30 minutes before $\rho_{j^*} \le E$ with at most $u + j^* \le k$ rides would contradict the definition of E, so the walking slack never prunes it. Hence such a label is pruned only when the bags are settled.

f) Induction along J: We show that, after the step of the run that handles e (the start of the departure for the access, the route scans of round u_e for an alighting, the transfers of round u_e for a ready event, the egress for the target), either the bags are settled or $m(p_e, u_e) \le t_e$ for the stop p_e and kind of e. Given the latter, the earliest label that achieves $m(p_e, u_e)$ is, by the paragraph on blocking labels, live, or the bags are settled. A live label is expanded once its round is complete: arrive bags are filled only by the route scans of their round and ready bags only by its transfers, so the label is still the earliest of its bag when

Algorithm S3 Scan of route $r = (p_0, \ldots, p_n)$ in round k, from position i_0.

procedure SCANROUTE(r, k)
  $R \leftarrow \emptyset$ ▷ riders: staircase of (trip, walking) on trips running on time
  for $i = i_0, \ldots, n$ **do**
    for all $(trip, w) \in R$ **do**
      $\ell \leftarrow (\tau_{\text{arr}}(trip, p_i), w)$;
      if USEFUL($\ell, p_i, k, 0$) **then**
        INSERT($\mathcal{A}_k(p_i)$, ℓ); if p_i is an egress stop, INSERT($\mathcal{T}_k$, ℓ + egress)
      end if
    end for
    for all live $\ell = (\tau, w) \in \mathcal{R}_{k-1}(p_i)$, from the earliest **do**
      if no rider with walking $\leq w$ leaves p_i before τ **and** USEFUL($\ell, p_i, k-1, 1$) **then**
        $trip \leftarrow$ earliest on-time trip of r leaving p_i at or after τ (binary search, next-trip table)
        if $k = 1$, require that leaving the origin for it falls within the window; if $k > 1$ in a plan query, require a wait of at most 2 h
        insert $(trip, w)$ into R if no rider dominates it
      end if
    end for
  end for
  for all trips of r moved by real-time data on days $d-1, d, d+1$ **do** ▷ one by one
    for $i = i_0, \ldots, n$ **do**
      alight at p_i as in lines 4–7 if the trip is being ridden
      board at p_i from the live ready label that walks least among those ready before its actual departure
    end for
  end for
end procedure

Algorithm S4 Transfers from stop p after round k.

procedure TRANSFERS(p, k)
  for all live $(\tau, w) \in \mathcal{A}_k(p)$ with USEFUL($(\tau, w), p, k, 1$) **do**
    $s \leftarrow$ comfort slack of p if this is the comfortable pass, else 0
    INSERT($\mathcal{R}_k(p)$, $(\tau + \max(60\,\text{s} + s, m_{\min}), w)$) ▷ $m_{\min}$: minimum transfer time asked
    for all links (p, p', m) with $m \leq 600$ **do**
      INSERT($\mathcal{R}_k(p')$, $(\tau + \max(m/v + 60\,\text{s} + s', m_{\min}), w + m/v)$) ▷ v: walking speed; s': comfort slack after walking, if comfortable
    end for
  end for
end procedure

it is expanded.

- Access: the search offers to $\mathcal{R}_0$ the access labels of every departure of the group, including J's.
- Ride: a live ready label at p with time at most t_e boards, in the next round, the earliest trip on time that it can catch, found by binary search, so by FIFO it reaches every later stop no later than J's trip if that trip is on time. A trip that real-time data moves is scanned on its own and boarded from the live ready label that walks least among those ready before its departure; its arrival times do not depend on which of them boards it, so they are offered as if the label at p had. If it is skipped, a rider that walks no more already leaves p before it is ready, and that rider's trip arrives no later. In round 1 a trip may be boarded only if leaving the origin for it falls within the window, which J's trip and any earlier one caught from a label of group d satisfy.
- Transfer and egress: a live arrive label at the alighting stop of J with time at most t_e yields, in the same round, a ready label at that stop after the margin or at a linked stop after the walk and the margin, and a target label after the egress walk, each no later than J's.

Each label offered in this way is, by (F1), inserted or rejected by one no later, and, by the paragraph on pruning, pruned only when the bags are settled; this gives the statement for the next event. At the target it gives a label with time at most E in $\mathcal{T}_u$ for some $u \leq k^*$, hence exactly E, or settled bags. This proves the claim for every group, that is, to the minute.

g) Rebuilding: A target label with the current stamp becomes a journey of the answer, rebuilt through the parent each label stores. That the rebuilt journey matches its label is checked, not proved: the search compares its arrival and walking with

Algorithm S5 Lower bounds to the target per number of remaining rides (v: walking speed).

$\mathrm{lb}_0^{\mathrm{alight}}(p) \leftarrow$ egress walk from p (∞ if none); $\mathrm{lb}_0^{\mathrm{board}}(p) \leftarrow \infty$
for $j = 1, \ldots, K$ **do**
  for all routes and extra trips $(p_0, \ldots, p_n)$, fastest segments δ_i **do**
    $\beta \leftarrow \infty$
    for $i = n, \ldots, 1$ **do**
      $\beta \leftarrow \min\big(\beta, \mathrm{lb}_{j-1}^{\mathrm{alight}}(p_i)\big) + \delta_{i-1}$
      $\mathrm{lb}_j^{\mathrm{board}}(p_{i-1}) \leftarrow \min\big(\mathrm{lb}_j^{\mathrm{board}}(p_{i-1}), \beta\big)$
    end for
  end for
  for all stops p **do**
    $\mathrm{lb}_j^{\mathrm{alight}}(p) \leftarrow \min\big(\mathrm{lb}_0^{\mathrm{alight}}(p), \mathrm{lb}_j^{\mathrm{board}}(p)\big)$
    for all links (p, p', m) with $m \leq 600$ **do**
      $\mathrm{lb}_j^{\mathrm{alight}}(p) \leftarrow \min\big(\mathrm{lb}_j^{\mathrm{alight}}(p), \frac{m}{v} + \mathrm{lb}_j^{\mathrm{board}}(p')\big)$
    end for
  end for
end for

the label's and drops it otherwise, and no check failed in the experiments (the enumeration and the reference search report any such journey).

h) The second property: We do not prove the second property. When no bag evicts a live label, insertion keeps every label that no other dominates, and the argument above suggests that every point of the trade-off within the walking slack is found; the exhaustive enumeration of Section VII-B checks it on small networks (Table S6).

S3. The two-hour rule in plan queries

Plan queries do not board a vehicle that would make the passenger wait more than two hours after a transfer. This limit breaks the monotonicity used above: a ready label that is earlier and walks no more than another may be unable to board a vehicle that the later label could board within two hours, so rejecting the later label by dominance can lose a journey. Removing the limit from the search does not remove the loss, because the selection layer then discards the journey with the long wait (Section V) and the later label is still dominated. Section VII-F therefore compares the two ways of applying the limit; it does not bound the loss against an exact search with the limit, which we have not implemented.

S4. Constants

Tables S1 and S2 list every constant of the core that affects which journeys are found or shown, with the value used in production and in all the measurements of the paper. All but the last two rows of Table S2 live in one configuration structure; those two are compile-time limits of the real-time update.

S5. Option labels

Table S3 gives the labels of Section V, in order of precedence; each option receives the first that applies.

S6. Query times, memory and variants

Fig. S1 gives the distribution of the plan-query times of Table II (region-wide set, data of 21 November, native and WebAssembly alternating query by query). WebAssembly took 1.40 to 1.53 times as long as native code at the 50th to 99th percentiles of the region-wide set and 1.33 to 1.49 times in the local set. Table S4 breaks down the memory of Section VII-D, as reported by the core; with the step-free walking table the linear memory is smaller, because that table is. The format's limit of about one million stops (20 bits in the walking table) is far beyond what memory on a phone would allow.

a) Other capacities: The core with $B = 1$ compares arrival times only, like rRAPTOR (Delling, Pajor and Werneck, 2015), and with $B = 16$ it overflows in only 18 of the 200 range queries; both keep the walking slack, stamps, shadows and the other rules, so they are variants of our core rather than implementations of those algorithms. Run in one interleaved pass with production (Table S5), the core with sixteen labels and no bounds took 3.2 times as long as production [3.0, 3.4] for 99% instead of 87% of the pooled frontier; with bounds, sixteen labels cost only 1.12 times the time of four, so the capacity is limited by working memory (Section VII-D) rather than by time. The core with one label and bounds took 0.63 times the time of production but returns the fastest journey per departure and number of rides whatever it walks, which is why only 539 of its 1,000 plan answers are those of production.

TABLE S1
CONSTANTS OF THE PLANNER (PRODUCTION VALUES): SEARCH, WALKING, TRANSFERS AND PLAN.

Constant	Value	Meaning
Search		
Maximum rides K	6	Rounds of the search
Bag capacity B	4	Labels per bag (stop, round and kind)
Departure grouping	60 s	Departures less than this apart are run together
Walking slack	30 min	Labels that cannot arrive within this of the fastest known journey with no more rides are pruned
Maximum wait after a transfer	2 h	In plan queries; range queries have no limit
Departure list	16,384	Departures per window chunk; if exceeded, the window is halved
Candidates	192	Journeys kept before selection
Walking		
Walking speed	1.2 m/s	833 ms per metre
Access and egress distance	1,200 m	From a point to a stop, along streets
Transfer distance	600 m	Between two stops
Far access	2,500 m, $\times 1.3$	When no stop is within 1,200 m: the nearest stop whose straight-line distance times 1.3 is at most 2,500 m
Access stops	256	Stops considered at each end
Walking-table grid	100 m	Cell side; access and egress distances are corrected from the cell centre
Stops per cell	128	Only stops that bring a route no nearer stop brings, nearest first; the links of a stop are pruned in the same way (Section III)
Transfers		
Same-stop margin	60 s	To board again at the stop where one alighted
Walking-transfer margin	60 s	Added to the walking time
Comfort slack, road	2 min	After alighting at a stop of a bus or coach network (comfortable pass)
Comfort slack, walking	1 min	After a walking transfer (comfortable pass); the larger slack applies
Plan		
First window	30 min	Then doubled while fewer than five distinct arrival times are found
Maximum window	4 h	
Look-ahead	24 h	For the existence check and the nearest journey of the diagnosis
Options shown	5	

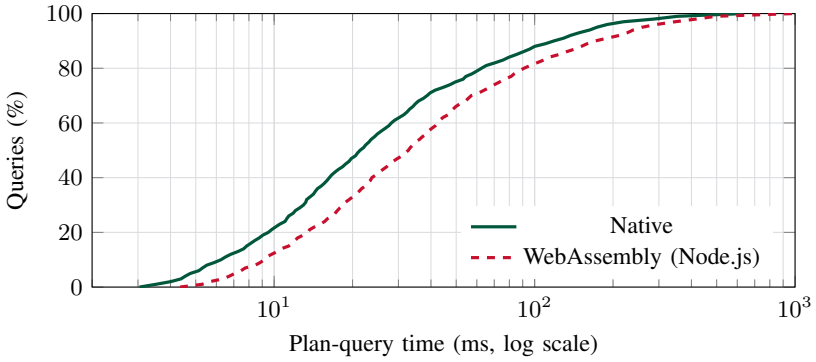


Fig. S1. Cumulative distribution of plan-query times, region-wide set (1,000 queries, builds alternating query by query).

S7. EXHAUSTIVE ENUMERATION

Table S6 gives the results of the exhaustive enumeration of Section VII-B.

S8. DATA FORMAT AND MEMORY

A pack is a sequence of little-endian 32-bit integers used in place, without parsing, covered by a CRC-32 checksum; the walking table packs each entry in one integer (stop in 20 bits, metres in 12) and names the packs it was built for, so data of different days cannot be mixed. Opening a pack checks sizes, checksum, every index, times that never go backwards and that no trip of a route overtakes another; afterwards the search does no bounds checking, and a malformed pack is rejected rather than read outside memory (Section VII-B). The core requests memory at start-up and never again, in *arenas* whose owner adds up beforehand everything it will need, so no search can run out of memory. Lists have explicit limits: a window with more than 16,384 departures is halved, a full candidate list (192 journeys) drops the worst and says so, and an answer over 512 KiB is replaced by an error rather than truncated. Every constant that changes results lives in one configuration structure, and the core contains no network name.

TABLE S2
CONSTANTS OF THE PLANNER (PRODUCTION VALUES): SELECTION, REAL-TIME DATA AND LIMITS.

Constant	Value	Meaning
Short rides		
Short ride	3 min	Rides this short may be replaced by walking
Short-ride walk	1,500 m	Longest walk that replaces them
Short-ride lateness	5 min	At most this later arrival (or earlier departure)
Fastest restored	2 min	The truly fastest option comes back if nothing is within this
Discarding and ranking		
Duration limit	$180\% + 15$ min	Of the shortest candidate
Extra rides	2	Over the candidate with fewest rides
Walking tolerance	3 min	In practical dominance, plus 3 min per ride saved
Fewer rides, later	1 min	A candidate with fewer rides may arrive this much later
Fewer rides, earlier	20 min	… and depart this much earlier
Transfer penalty	5 min	Per transfer, in the score
Missing probability	50%	Of a risky transfer, in its expected loss
Maximum expected loss	30 min	Per risky transfer (also when there is no later vehicle that day)
Less-walking gain	1 min	Minimum saving for the “less walking” option
Fill spread	2 h	Fill-in options arrive at most this after the first
Variant window	5 min / 20 min	Same main vehicle arriving / departing this close is a variant
Delayed label	2 min	Minimum delay for the “delayed” label
Real-time data and limits		
Delays per update	1,024	Per `live` request
Extra trips	64	With 2,048 stop visits in total
Request / answer	128 KiB / 512 KiB	JSON buffers
Maximum delay	$\pm$12 h	Larger delays or advances in a `live` update are rejected
Extra-trip match	5 min	A ride re-planned on an extra trip is matched to the extra trip of its line leaving its stop closest in time, at most this apart

TABLE S3
OPTION LABELS, IN ORDER OF PRECEDENCE.

	Label	Condition
1	Last of the day	On some ride, nothing of the network leaves later that day from the boarding stop towards the alighting stop or the next stop
2	Tight transfer	Some transfer is risky
3	Delayed	Some vehicle is two or more minutes late
4	Line alert	It uses a line with a service alert
5	Your line	It uses a favourite line
6	Fastest	It truly arrives earliest (departs latest, arrive-by)
7	Fewer transfers	Chosen for fewer rides
8	Less walking	Chosen for walking least
9	Safe transfer	No risky transfer, and the fastest has one
10	Leaves later	Same lines as the first, N minutes later or earlier
11	Other network	It uses a network not selected

S9. BUILDS, SCENARIOS AND MALFORMED INPUT

The native and WebAssembly builds gave byte-identical answers to every plan query of Section VII-B, and the WebAssembly module rebuilt from the measured source revision is byte-identical to the one in production. 186 hand-written scenarios, each a small network with an exact expected answer for one rule, run on both builds, the native one with the address and undefined-behaviour sanitisers, without errors. A fuzzer feeds corrupted packs, walking tables, links and requests: every corrupted file is rejected or answered without reading outside memory, except three walking tables with absurd cell sizes that overflow an integer in an operation the application does not use and in the access distances of a query from a point. The measured revision accepts such tables; the production tables use 100 m cells and are checksummed.

S10. SENSITIVITY OF THE SELECTION LAYER

Table S7 gives, for each threshold of the selection layer of Section V halved and doubled in turn, how many of the 1,979 plan answers with options change (2,000 queries of Section VII-C, data of 21 November). When a ranking constant or the comfort slack changes the first option, the change follows the constant. Halving the transfer penalty gives a first option with one ride more in all 55 answers that change, arriving earlier in 52 (median 3.9 min); doubling it, one ride fewer in all 80, arriving later in 75 (median 8 min). Halving the probability of missing a risky transfer gives more risky transfers in 63 of 65, arriving earlier in 46 (median 7.5 min); doubling it, fewer in 53 of 54, arriving later in 49 (median 14 min). Halving the maximum

TABLE S4
MEMORY OF THE CORE WITH THE PRODUCTION DATA OF 21 NOVEMBER 2026 (8,242 STOPS).

Part	Bytes
Packs of the fifteen networks (used in place)	5,432,948
Walking table (used in place)	8,893,216
Tables of the merged network, extra trips, records and names	776,140
Network (`world_bytes`)	**15,102,304**
Bag labels: 8,242 stops × 6 rounds × 2 bags × 4 labels × 32 B	12,658,176
Bag headers, bounds, next-trip table, round lists, departures and candidates	2,956,040
Search (`planner_bytes`)	**15,614,216**
WebAssembly linear memory	**31,981,568**

TABLE S5
VARIANTS OF THE CORE ON THE SAME QUERIES (NATIVE, DATA OF 21 NOVEMBER, ONE INTERLEAVED PASS). SAME: PLAN ANSWERS IDENTICAL TO PRODUCTION. FRONTIER: SHARE OF THE POOLED FRONTIER OF THE 200 RANGE QUERIES. RATIO: TOTAL PLAN TIME RELATIVE TO PRODUCTION, WITH 95% INTERVAL OVER QUERIES.

	Plan queries (1,000)					Range queries (200)	
Variant	Median (ms)	95th (ms)	Total (s)	Ratio	Same	Total (s)	Frontier (%)
$B = 1$	43.5	221.5	67.7	1.36 [1.28, 1.45]	539	4.2	14.3
$B = 1$ + bounds	16.5	118.3	31.5	0.63 [0.61, 0.65]	539	2.3	14.3
$B = 4$	69.8	475.4	131.8	2.64 [2.50, 2.79]	989	16.7	86.2
$B = 4$ + bounds	**24.0**	**189.2**	**49.9**	1	**1,000**	**5.8**	**87.1**
$B = 16$	78.2	589.5	158.6	3.18 [3.00, 3.36]	979	22.6	99.3
$B = 16$ + bounds	25.5	206.4	55.8	1.12 [1.09, 1.14]	982	6.8	99.4

expected loss gives more risky transfers in 27 of 28 and earlier arrivals in 16 (median 16 min); doubling it, fewer in 13 of 14 and later arrivals in 12 (median 36 min). Doubling the comfort slack at road stops changes the first option in 199 answers, arriving later in 94 and at the same time in 92, with fewer rides in 44 and more in 18; with the same departure, arrival and rides, its legs or walking change in 196 more. The short-ride threshold, a filter, changes the first option in 41 and 47 answers, mostly in rides (one more in 25 when halved, one fewer in 29 when doubled).

S11. PLAUSIBILITY AUDIT

The reference search checks that journeys are possible and optimal; it does not check that the five options make sense to a person. The audit tool of Section VII-H runs plan queries between random street points (within a few hundred metres of a stop, at most 40 km apart, on a random day of the week ahead) and counts the options a passenger would find odd. Table S8 gives the counts for the seed used to tune the rules and for three seeds that were never used, with the data of 21 November. The examples printed by the tool (three per category and seed) show that long waits and journeys with four or more vehicles come from rural areas with a bus every hour or two and from journeys between districts that need that many vehicles. Every option that another option of the same list beats in time has more risky transfers in the option that beats it, so the selection keeps the safer one on purpose.

S12. COMPARISON WITH OPENTRIPPLANNER: DETAILS

This section gives the details of Section VII-G.

a) Setup and measures: The fifteen packs of 21 November 2026, measured with revision `3b769a2` of the core like every other table, were exported to GTFS by the functions that build the packs, keeping the trips of the day before, the day and the day after the query day; a separate reader of the binary packs confirmed that the feeds hold the same trips, stops, times to the second and service days. OTP's walking model adds time at crossings, on stairs and in lifts, which made it walk at 0.94 to 1.13 m/s on level streets; these extras were switched off so that it walks at 1.2 m/s, as our table does. In its debugging mode, OTP's itinerary filters mark the itineraries they would remove instead of removing them. The 200 range queries are those of Section VII-E (one-hour windows starting between 06:00 and 21:00, same generator, seed and data). All processes were warmed up with the same 20 queries. The *window* convention of Table VI is called *strict* in the result files; under it, the journeys that a later OTP departure hides count as found only by us. Besides bootstrap intervals over queries, exact binomial (Clopper–Pearson) intervals are used where stated.

TABLE S6

RANGE SEARCH AGAINST AN EXHAUSTIVE ENUMERATION ON 1,000 RANDOM SMALL NETWORKS (8,000 QUERIES). EXACT EARLIEST: QUERIES IN WHICH THE EARLIEST ARRIVAL WITH EVERY NUMBER OF RIDES WAS RETURNED. FULL BAG: QUERIES IN WHICH A BAG EVICTED A LIVE LABEL; JOURNEYS MAY BE MISSING ONLY THERE. WITH $B = 1$ THE SEARCH COMPARES ARRIVAL TIMES ONLY AND DOES NOT SEARCH THE WALKING TRADE-OFF, SO ONLY THE FIRST ROW APPLIES: THE TOOL'S LOG FOR THIS CAPACITY REPORTS THE 4,524 JOURNEYS THAT WALK LESS AS MISSING AND 312 RETURNED JOURNEYS THAT WALK MORE AS DOMINATED, AND NO OTHER PROBLEM (`CORRECCION.JSON` IN THE ARCHIVE).

	$B = 4$	no slack	$B = 2$	$B = 1$
Exact earliest arrivals	8,000	8,000	8,000	8,000
per departure (31,206 journeys)	8,000	8,000	8,000	8,000
Queries with a full bag	5	16	617	–
Missing non-dominated journeys	2	12	574	–
Dominated returned within the slack	0	0	9	–

TABLE S7

SENSITIVITY OF THE SELECTION LAYER: PLAN ANSWERS (OF 1,979 WITH OPTIONS) THAT CHANGE WHEN A CONSTANT IS HALVED OR DOUBLED. TIMES: DEPARTURE, ARRIVAL OR RIDES OF THE FIRST OPTION; FIRST: THE WHOLE FIRST OPTION, WALKING AND LEGS INCLUDED; ORDER: THE ORDERED LIST OF OPTIONS (DEPARTURE, ARRIVAL, RIDES); SET: THE SAME LIST UNORDERED.

		Halved				Doubled			
Constant	Value	Times	First	Order	Set	Times	First	Order	Set
Walking tolerance	3 min	1	2	594	594	1	6	629	628
Duration limit (excess)	80%	1	4	230	230	2	3	66	66
Duration slack	15 min	1	1	59	59	1	2	50	50
Extra rides	2	1	38	250	250	3	12	50	50
Fewer rides, later	1 min	0	5	15	15	0	10	37	37
Fewer rides, earlier	20 min	1	1	191	191	0	0	204	204
Less walking, gain	1 min	0	0	12	12	0	0	23	23
Fill spread	2 h	0	0	521	521	0	0	253	253
Short ride	3 min	41	61	270	270	47	64	235	234
Transfer penalty	5 min	55	55	187	63	80	81	287	70
Missing probability	50%	65	66	228	136	54	54	220	100
Maximum expected loss	30 min	28	28	105	51	14	14	50	18
Comfort slack, road	2 min	100	197	390	344	199	395	599	545

b) Results: The variant with bags of 16, no walking slack and no grouping of departures finds the same share as production to within half a point at every tolerance and within 0.3 points at 0, 60 and 120 s. Restricted to the 181 of 200 queries in which OTP locates both ends, the shares are 93.8% and 82.0% (93.7% and 85.7% in the second sample); with a two-hour cap on the wait between rides, as in plan queries, 94.3% and 79.4% (93.8% and 83.5%).

c) Agreement under the same walking distances: Our table's metres are converted at 833 ms/m, with the 60 s slack. In the second sample, OTP with its default pruning leaves 19 of the 12,501 journeys of ours that its walking allows unmatched within 120 s (0.15%), and recovers 2 of them without its pruning of one transfer per pattern; in the main sample it returns 2,318 of 2,320 within 60 s. With the production table, our planner returns 3,224 of the 3,364 OTP journeys within 60 s and 3,317 within 120 s. The production table differs from the unpruned one in keeping only useful stops and at most 128 instead of 256 stops per cell; the loss is the combined effect of both.

d) Where the answers differ: Table S9 classifies every discrepancy of the frontier. With the same extracts, the two planners link stops to the streets and find paths differently: on 400 random links of our table between 100 and 600 m, OTP's path is 2% longer at the median, and a tenth of the links are more than 20% longer; 8% of them exceed OTP's 500 s transfer limit, and 7% of 400 random access walks of up to 1,200 m exceed its 1,000 s. Our limits are distances and OTP's are times, so near a limit a walk fits in one planner and not in the other, and a tight transfer is made by one and missed by the other. Some stops are not linked to OTP's streets at all, so that from a point next to them OTP reaches no stop, whereas our table walks to them in a straight line with a 30% detour; it does the same where the extracts have no streets (in Burgos and Aragon), and there OTP does not find the place. Without the pruning of our walking table, our share of the edge-neutral frontier rises from 94.1% to 95.1% (from 93.9% to 94.9% in the second sample). Six discrepancies of the second sample, all missing from OTP, remain unexplained. A random sample of 30 discrepancies was checked by hand; in the 25 of the edge-neutral comparison the hand classification agrees with the automatic one, after one hand label (an egress walk first read as a transfer) was corrected on comparing the two, so the check was not blind. The other 5, which only the strict comparison counts as missing from OTP, are all hidden by a later departure. No discrepancy came from stations (with coordinates at both ends, grouping platforms into stations plays no part) or from OTP's filters.

e) Single answers: Of the 142 queries in which both planners have a journey leaving in the window, the earliest arrival is the same within a minute in 97, earlier with ours in 28 (median 5 min, largest 2.0 h) and earlier with OTP in 17 (median

TABLE S8
PLAUSIBILITY AUDIT: FINDINGS PER SEED, IN ALL OPTIONS SHOWN AND, IN PARENTHESES, IN FIRST OPTIONS (400 QUERIES EACH; SEED 1 IS THE ONE USED TO TUNE THE RULES).

Seed	1	2	3	4
Queries with options	400	400	399	395
Options shown	1,445	1,495	1,434	1,506
Transfer waiting more than 25 minutes	253 (97)	232 (80)	276 (100)	219 (64)
Four or more vehicles	194 (58)	174 (50)	197 (64)	151 (50)
Two consecutive rides on the same line	27 (12)	38 (9)	27 (3)	36 (9)
Ride that walking would match	3 (1)	4 (1)	5 (2)	3 (2)
Beaten in time by a riskier option of the list	38 (12)	31 (12)	31 (14)	43 (15)
Twice as long as the first option	1	1	0	0

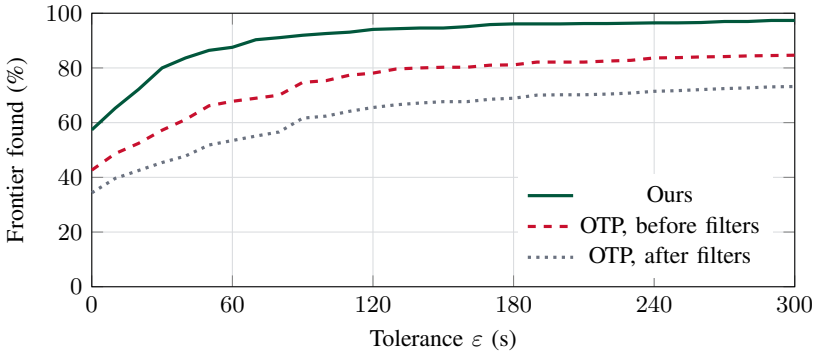


Fig. S2. Share of the edge-neutral pooled frontier (795 journeys, 200 queries) found within a tolerance ε. In the second sample the difference is positive at every tolerance from 0 to 300 s.

3 min); the difference in direction is not significant (two-sided sign test, $p = 0.14$; $p = 0.07$ for the 128 against 100 of the second sample). The test leaves out the queries in which only one planner has a journey in the window, 34 against 0 in the main sample and 188 against 2 in the second, which all favour ours. The largest difference in OTP's favour, 11.1 h, comes from the pruning of the walking table: the stop that gives OTP an evening connection was dropped from our table because a nearer stop serves the same route, and our planner, without it, only reaches the target the next morning; with the unpruned table it finds the evening connection. In 34 queries only our planner has a journey in the window: in 16, an OTP journey leaving after the window arrives as early with no more rides; in 5, OTP does not find the origin or the destination, outside its streets; in 7, it reaches no stop from one end; and in the other 6, its walking limits leave it with no connection in the window. In none of the 200 does OTP alone have one, and in 2 of the 1,000 of the second sample, both for a walk that our table measures as longer than 1,200 m and OTP as shorter.

f) Sensitivity to the walking model: With OTP's default walking model, our planner finds 96.9% of the edge-neutral frontier [95.0, 98.4] and OTP 59.7% [52.3, 67.4], and ours has the earlier arrival in 50 queries against 12 ($p < 10^{-5}$). Our search with the unpruned table returns all 503 OTP journeys that our walking distances allow (within 60 s), the production table 490 of them, and OTP all 1,821 of ours that its walking allows.

g) Footprint and speed: At the 95th percentile the WebAssembly module took 91 against 115 ms for OTP after its filters in the main sample and 105 against 97 ms in the second; it gave the same answer as the native build, byte for byte, in all 1,200 queries. The Node.js process used 99 MB (the core reports 30.7 MB for its data structures with these data); the OTP process 2.1 GB, a figure that reflects the heap limit it was started with rather than what it needs. Our packs and walking table take 14.3 MB uncompressed (5.9 MB with gzip). These numbers measure different things (a pipe against HTTP, one process against a server designed to serve many users) and are not a benchmark of OTP.

h) Further threats to validity: The re-timing uses OTP's own street search, checked against its transfers; queries in which OTP reports that it reaches no stop from one end are counted as out of its reach. Of the 17 journeys of ours that OTP leaves unmatched in the second sample for reasons other than its transfer pruning, one checked by hand is an 858 s egress walk that OTP's street search makes but its transit search does not use. Times were measured on a shared four-core virtual machine and include building answers of different sizes.

TABLE S9

WHY A FRONTIER JOURNEY (EDGE-NEUTRAL, 120 S) IS MISSING FROM ONE PLANNER. ALL: EVERY CASE OF THE 200 QUERIES, CLASSIFIED BY CHECKING EACH LEG OF THE OTHER PLANNER'S JOURNEY AGAINST THE WALKING RULES OF THE ONE THAT MISSES IT. SAMPLE: CASES CHECKED BY HAND. SECOND: THE SECOND SAMPLE OF 1,000 QUERIES. WALKING: THE LEG DOES NOT FIT THE LIMIT, MISSES A CONNECTION, OR MAKES THE JOURNEY MORE THAN TWO MINUTES WORSE.

	Missing from ours			Missing from OTP		
Cause	All	Sample	Second	All	Sample	Second
Walking: access leg	11	0	103	33	0	167
Walking: egress leg	23	6	66	45	10	212
Walking: transfer	5	2	26	57	4	200
No stop within OTP's reach	–	–	–	29	0	123
Place outside the street data	–	–	–	9	1	36
Walking-table pruning (ours)	8	2	63	–	–	–
One transfer per pattern (OTP)	–	–	–	1	0	0
Not explained	0	0	0	0	0	6
Search, stations, filters	0	0	0	0	0	0
Total	47	10	258	174	15	744

TABLE S10

RANGE-QUERY TIME (200 QUERIES, ONE-HOUR WINDOW, WARMED UP, SAME MACHINE) AND RESIDENT MEMORY AT THE END OF THE RUN. CONTEXT ONLY: OTP IS A SERVER ON THE JAVA VIRTUAL MACHINE, STARTED WITH AN 8 GIB HEAP LIMIT, AND ANSWERS OVER HTTP; OURS ANSWERS THROUGH A PIPE. DATA: WHAT EACH PLANNER LOADS (OUR PACKS AND WALKING TABLE, UNCOMPRESSED; OTP'S GRAPH).

	Median (ms)	95th pct. (ms)	Memory (MB)	Data (MB)
Ours, native	14.1	68.5	32	14.3
Ours, WebAssembly (Node.js)	20.5	91.4	99	14.3
OTP, before filters	50.3	116.4	2,081	181.5
OTP, after filters	49.1	114.8	2,081	181.5