

Predicting bottlenecks from a floor plan

|V| = 388 nodes
|E| = 427 edges

A building’s floor plan and a school’s timetable together determine, mathematically, where crowding will happen — before a single student walks through it. We converted the new building’s architectural drawings into a network, drove it with the real school timetable, and asked: where does it jam, and can reshuffling which class uses which room fix it?

PROCESS

Turning a floor plan into a graph.

We built custom software that lets you draw directly on top of an imported floor plan: click to place a node at every room, junction, and staircase landing, then connect them with edges. Each edge is automatically weighted by its measured length and the walking speed measured in Phase 1 (1.47 m/s); staircase edges get a corrected weight for the extra time height costs. The result: 388 nodes and 427 weighted edges across both floors, with staircases linking the two levels together.

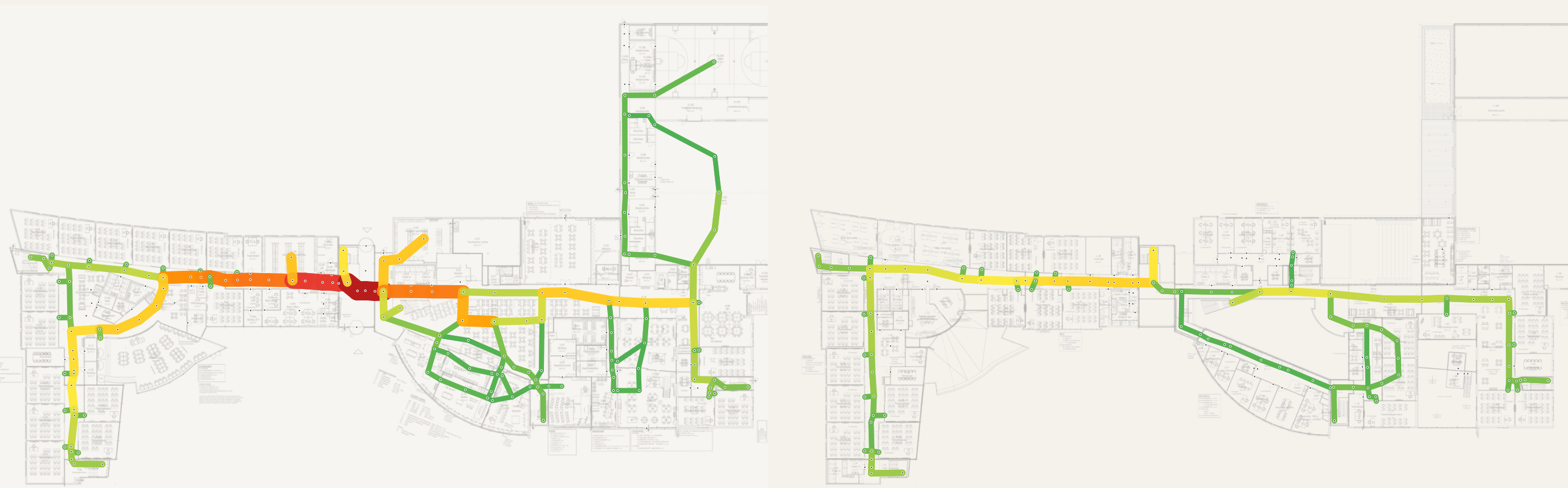
52,000 trips a week.

The current school’s real weekly timetable — translated onto the new building’s room layout — was parsed into more than 52,000 individual student trips: student A, room 0.27 to room 1.73, at 11:05. Each trip is routed along its shortest path with Dijkstra’s algorithm, and we count how many times each corridor segment gets walked across the full week. That count, per edge, is the load map below.

1 GRAPH GENERATION SOFTWARE



2 CORRIDOR LOAD, GROUND FLOOR & FIRST FLOOR



WHAT WE FOUND

- 1 One hall carries 62% of everyone.**
Across all 52,000 simulated trips, 62% pass through a single central hall. This isn’t a scheduling artifact — it’s the building’s shape. The new building is essentially two wings joined by one hub: any trip between wings, between floors, or to the entrance has to cross that space on either floor. No room reassignment changes this.
- 3 A load distribution that's 6.6× uneven.**
The busiest corridor segment carries 6,513 passages a week — 6.6× the network average. The Gini coefficient measures this concentration on a 0–1 scale (0 = perfectly even, 1 = one edge carries everything); the original layout scores **0.585**.

- 2 The busiest staircase narrows to under a metre.**
Of seven staircases, five are open to students. The one nearest the hall — the obvious route upstairs — is five people wide at the bottom but funnels to roughly 0.8 m at the top landing, exactly where the most people try to pass. The old building spread students across effectively 16 staircase flights; this one offers 5.
- 4 The bottleneck is structural, not incidental.**
Every one of the 18 heaviest-loaded edges sits on a route into or out of the central hall or its staircase. There is no second, independent bottleneck elsewhere in the graph to relieve pressure on — the hall has no substitute.

CAN ROOM ASSIGNMENT FIX IT?

Of the building’s 51 rooms, 21 have a fixed function — labs, gyms, workshops — and can’t move. For the remaining 30, we searched for the room assignment that spreads load most evenly, using **Simulated Annealing**: an algorithm that tries thousands of swapped layouts, occasionally accepting a worse one on purpose so it doesn’t get stuck in a mediocre local solution, and gradually cools down to converge on a good one. Each candidate layout costs ~18 seconds to fully re-simulate; 500 iterations took about 2.5 hours to converge.

A real improvement, but a small one — that’s the finding itself. The total number of trips doesn’t change; optimisation only redistributes it. Five independent runs from different starting points converged to scores within 3% of each other but different layouts: there’s no single “correct” assignment, only a family of comparably good ones. Scheduling can redistribute load at the margins; it can’t undo a building’s shape.

Metric	Original	Optimised	Change
Gini coefficient	0.585	0.568	–3%
Bottleneck edges (>μ+2σ)	18	14	–4
Maximum edge load / wk	29,825	27,926	–6.4%
Mean edge load / wk	4,881	4,868	–0.3%

Simulated Annealing, T₀ = 100, α = 0.98, 500 iterations, λ = 10 penalty weight.

The school is currently considering closing two outer staircases, on the assumption that the central ones make them redundant. We re-ran the simulation with those staircases removed: their share of inter-floor traffic doesn’t disappear — it gets pushed onto the already-overloaded central staircase, making the worst bottleneck in the building meaningfully worse. This is the kind of decision this model can answer before it’s made, not after.