L2b Pipeline example: capacity and verdicts

Why does raising one server change nothing and raising another change everything?

the capacity service and the example model, after the demo has been run once

owner: Roar Georgsen

status: approved without external review, 2026-08-27

functional view: how the number is made

① read the servers and wiring
the projection, via the router

servers and their wiring

② build the flow network
one edge per server, its throughput

the flow network

③ find the maximum flow
entry to exit servers: the capacity

the residual network

④ find the source-side cut
saturated servers nearest the entry

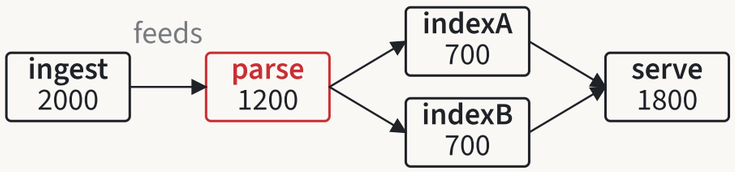
capacity and bottleneck

⑤ compare with each limit
PASS if capacity >= limit, else FAIL

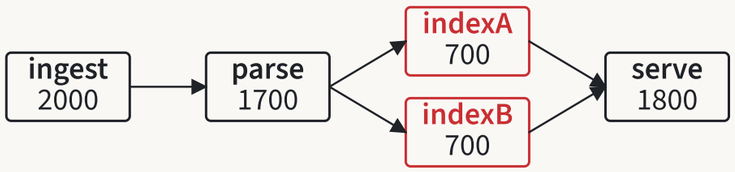
known issue: limits of validity
exact for the idealised model,
an upper bound for a real one that
meets the assumptions, and not for
capacity planning. The assumptions:
flow is conserved at every server
even partitioning across branches
perfect load balancing
no queueing, no latency coupling
stationary load, no bursts
unlimited connections

the wiring, in queries per second

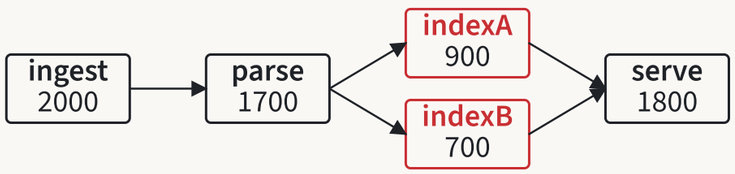
shipped: 1200, PIPE-R1 FAIL



parse to 1700: 1400, PIPE-R1 FAIL



then indexA to 900: 1600, PIPE-R1 PASS



verdicts and reasons

⑥ raise a server outside the cut
ingest 2000 to 3000: nothing moves

the same number, the same cut

⑦ raise a server inside the cut
parse 1200 to 1700: the cut migrates

then indexA 700 to 900: capacity 1600,
PIPE-R1 PASS, and PIPE-R1.4 on indexB
still fails, 700 against its allocated 750

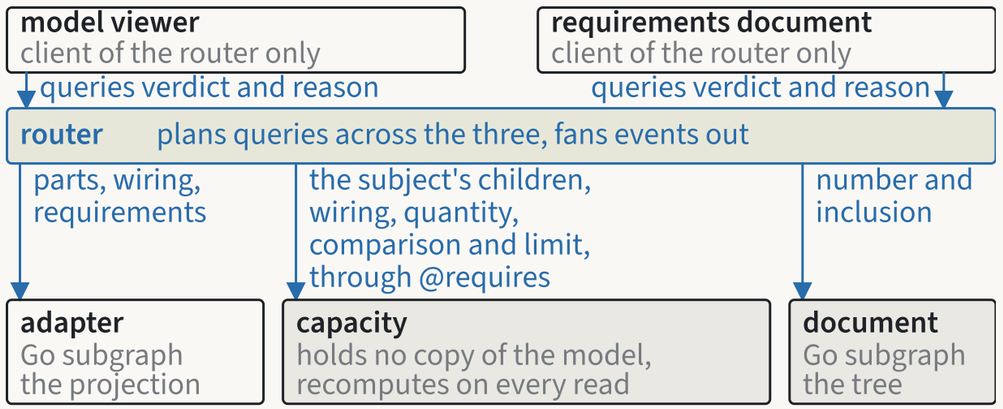
the arithmetic

capacity = the maximum flow, entry to exit servers, each
server one edge sized by its throughput = the minimum cut:
min over a chain, sum over parallel branches

state	cut	capacity, PIPE-R1					derived
shipped	parse	1200 FAIL	PASS	FAIL	FAIL	FAIL	PASS
ingest 3000	parse	1200 FAIL	PASS	FAIL	FAIL	FAIL	PASS
parse 1700	indexA indexB	1400 FAIL	PASS	PASS	FAIL	FAIL	PASS
indexA 900	indexA indexB	1600 PASS	PASS	PASS	PASS	FAIL	PASS

derived, in order: ingest, parse, indexA, indexB, serve
limits: 1500 on the pipeline, ingest, parse and serve,
750 on each index server. Row 2 is box 6, row 4 is box 7:
PIPE-R1.4 fails at 700 against its allocated 750 while
PIPE-R1 passes
reasons capacity 1200 against 1500, limited by parse
capacity 1400 against 1500, limited by indexA, indexB
capacity 1600 against 1500, limited by indexA, indexB
all values known, from the running example,
guarded by a unit test

where it runs: three services, one router



decisions

- ★ **AD-0006** idealised model
  only throughput limits a server
- ★ **AD-0024** reasons from fixed templates
  a reason carries no word of the model
- ★ **AD-0007** maximum flow, source-side cut
  any wiring, the nearest cut is unique
- ★ **AD-0025** the document owns its structure
  it owns order, nesting and numbering

the record: the article series under docs/articles, the capacity model in article 6.
Sibling sheets: L0, L1, L2a. Decision records: AD-0006, AD-0007, AD-0024, AD-0025.

- a component
- owned by the example
- the router and what passes through it
- a failing verdict or the bottleneck
- ★ a decision record, its id and one line of rationale (the cookbook types stars by concern)
- ① the reading path
- 1200 a known value