

Graph Optimization with Mixed Integer Programming

Ian Chen



Optimization problems in the wild

- Recommendation systems
- Resource allocation
- Unsupervised learning
- **Traffic engineering (routing)**
- **Traveling salesman**



Outline

Traffic engineering

Integer linear programs

Branch and Bound

Traveling salesman

Conclusion



Jupyter Notebook!



https://colab.research.google.com/github/SIGma-UIUC/meetings/blob/main/FA26/Gurobi/notebook.ipynb



Traffic engineering

Integer linear programs

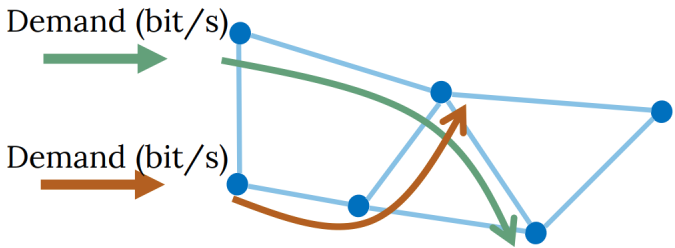
Branch and Bound

Traveling salesman

Conclusion



Route demands in the WAN

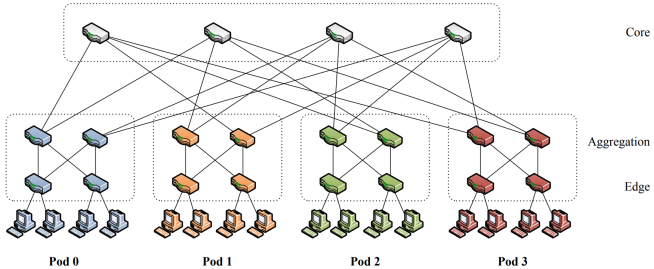


SWAN (Microsoft), B4 (Google)

Pooria Namyar et al. “Solving Max-Min fair resource allocations quickly on large graphs”. In: *NSDI '24*. 2024, pp. 1937–1958

Traffic engineering (cont.)

- Servers $\ll$ racks $\ll$ aggregation node $\ll \ldots$



- How should flows be routed from nodes i to j...
  - to minimize latency
  - to maximize bandwidth
  - to be maximally robust to failures

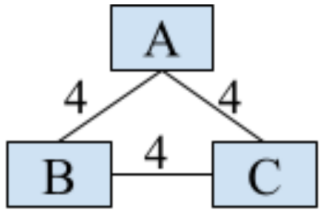
Radhika Niranjan Mysore et al. “PortLand: a scalable fault-tolerant layer 2 data center network fabric”. In: SIGCOMM ’09. ACM, Aug. 2009, pp. 39–50

Jupiter traffic engineering

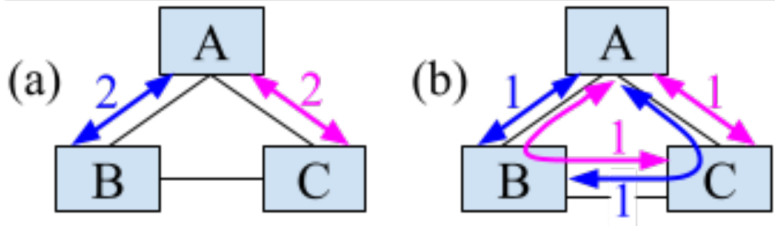
- input: $\approx$ 10 aggregation blocks
- input: a bundle of parallel links between blocks
- input: predicted traffic matrix between blocks
- constraint: flows must reach destination within 2 hops (latency)

Leon Poutievski et al. "Jupiter evolving: transforming google's datacenter network via optical circuit switches and software-defined networking". In: SIGCOMM '22. ACM, Aug. 2022, pp. 66–85

Ex: Jupiter traffic engineering



Capacity	Demand	
	Predicted	
A ↔ B: 4 A ↔ C: 4 B ↔ C: 4	A ↔ B: 2 A ↔ C: 2	



Leon Poutievski et al. "Jupiter evolving: transforming google's datacenter network via optical circuit switches and software-defined networking". In: SIGCOMM '22. ACM, Aug. 2022, pp. 66–85

Jupiter traffic engineering (cont.)

- def: a commodity (s, t, d) is a demand d between blocks s and t
- input: a network $G = (V, E)$, capacities c_e, commodities (s_k, t_k, d_k)
- decision variables: for every $p : s_k \rightsquigarrow t_k$
  - x_p^k the load commodity k should put on on path p
- objective: minimize max-utilization U on any arc

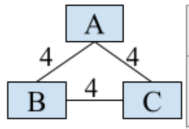
Leon Poutievski et al. "Jupiter evolving: transforming google's datacenter network via optical circuit switches and software-defined networking". In: SIGCOMM '22. ACM, Aug. 2022, pp. 66–85

Jupiter traffic engineering (cont.)

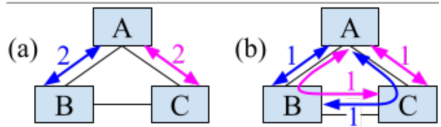
$$
\begin{aligned}
\min_{x,U} \quad & U \\
\text{s.t.} \quad & \sum_{p \in P_k} x_p^k = d_k \qquad \forall k \\
& \sum_{k} \sum_{p \in P_k : e \in p} x_p^k \leq U c_e \qquad \forall e \\
& x_p^k \geq 0 \qquad \forall p \in P_k
\end{aligned}
$$

Leon Poutievski et al. “Jupiter evolving: transforming google’s datacenter network via optical circuit switches and software-defined networking”. In: SIGCOMM ’22. ACM, Aug. 2022, pp. 66–85

Jupiter traffic engineering (cont.)



Capacity	Demand	
	Predicted	Actual
A ↔ B: 4 A ↔ C: 4 B ↔ C: 4	A ↔ B: 2 A ↔ C: 2	A ↔ B: 4 A ↔ C: 2



- Robustness to inaccurate predictions
- Maximum-utilization for (a) is 1
- Maximum-utilization for (b) is 0.75

Leon Poutievski et al. “Jupiter evolving: transforming google’s datacenter network via optical circuit switches and software-defined networking”. In: SIGCOMM ’22. ACM, Aug. 2022, pp. 66–85

Jupiter traffic engineering (cont.)

- Hyper-parameter $S \in (0, 1]$

$$
\begin{aligned}
\min_{x,U} \quad & U \\
\text{s.t.} \quad & \sum_{p \in P_k} x_p^k = d_k \qquad \forall k \\
& \sum_k \sum_{p \in P_k : e \in p} x_p^k \le U c_e \qquad \forall e \\
& x_p^k \le d_k \frac{\min_{e \in p} c_e}{\sum_{q \in P_k} \min_{e \in q} c_e} \left(\frac{1}{S} \right) \qquad \forall k, p \in P_k \\
& x_p^k \ge 0 \qquad \forall k, p \in P_k
\end{aligned}
$$

Leon Poutievski et al. “Jupiter evolving: transforming google’s datacenter network via optical circuit switches and software-defined networking”. In: SIGCOMM ’22. ACM, Aug. 2022, pp. 66–85



Traffic engineering

Integer linear programs

Branch and Bound

Traveling salesman

Conclusion



bipartite matching

- Given a bipartite graph $G = (L \cup R, E)$, find a maximum *matching*.

$$\begin{aligned} \max_{x} \quad & \sum_{e \in E} x_e \\ \text{s.t.} \quad & \sum_{u \in N(v)} x_{uv} \leq 1 \qquad \forall v \in L \cup R \\ & x_{uv} \in \{0, 1\} \end{aligned}$$



bipartite matching (cont.)

- Given a bipartite graph $G = (L \cup R, E)$, find a maximum *matching*.

$$\begin{aligned} \max_{x} \quad & \sum_{e \in E} x_e \\ \text{s.t.} \quad & \sum_{u \in N(v)} x_{uv} \leq 1 \qquad \forall v \in L \cup R \\ & 0 \leq x_{uv} \leq 1 \end{aligned}$$

- Theorem: best fractional LP solution equals the best integer optimum

vertex cover

- Given a graph $G = (V, E)$, find the smallest *vertex cover*

$$\begin{aligned} \min_{x} \quad & \sum_{v \in V} x_v \\ \text{s.t.} \quad & x_u + x_v \geq 1 \qquad \forall uv \in E \\ & x_v \in \{0, 1\} \end{aligned}$$

vertex cover (cont.)

- Solving the LP relaxation:

$$\begin{aligned} \min_{x} \quad & \sum_{v \in V} x_v \\ \text{s.t.} \quad & x_u + x_v \geq 1 \qquad \forall uv \in E \\ & 0 \leq x_v \leq 1 \end{aligned}$$

- So... how does this help us?



vertex cover (cont.)

- Rounding a solution x^* to the LP
- Let $\bar{x}_v = I(x_v^* \geq 1/2) \in \{0, 1\}$
- Satisfies $\bar{x}_v + \bar{x}_u \geq 1$ for each $uv \in E$
- 2-approximation: $\sum_{v \in V} \bar{x}_v \leq 2 \sum_{v \in V} x_v^* \leq 2\mathrm{OPT}$



independent set

- Given a graph $G = (V, E)$, find the largest *independent set*

$$\begin{aligned} \max_{x} \quad & \sum_{v \in V} x_v \\ \text{s.t.} \quad & x_u + x_v \leq 1 \qquad \forall uv \in E \\ & x_v \in \{0, 1\} \end{aligned}$$



independent set (cont.)

- Solving the LP relaxation:

$$\begin{aligned} \max_{x} \quad & \sum_{v \in V} x_v \\ \text{s.t.} \quad & x_u + x_v \leq 1 \qquad \forall uv \in E \\ & 0 \leq x_v \leq 1 \end{aligned}$$



independent set (cont.)

- Suppose $G = K_n$, then maximum independent set has 1 vertex
- obs: LP solution is at least $n/2$
- proof: let $x_v = 1/2$ for all vertices
- no (easy) way to *round* the LP to be integral





Traffic engineering

Integer linear programs

Branch and Bound

Traveling salesman

Conclusion



General form of integer linear program

$$
\begin{aligned}
\min \quad & \boldsymbol{c}^T \boldsymbol{x} \\
\text{s.t.} \quad & A\boldsymbol{x} \leq \boldsymbol{b} \\
& x_i \in \mathcal{Z}
\end{aligned}
$$

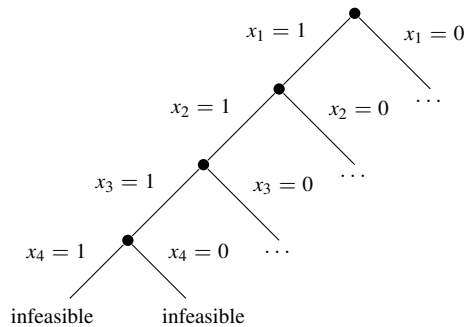
$$\Downarrow$$

$$
\begin{aligned}
\min \quad & \boldsymbol{c}^T \boldsymbol{x} \\
\text{s.t.} \quad & A\boldsymbol{x} \leq \boldsymbol{b} \\
& x_i \in \{0, 1\}
\end{aligned}
$$

Naïvety

- Enumerate all 2^n possible assignments
- Find the best out of all feasible solutions

$$\begin{aligned} \min \quad & -6x_1 - 2x_2 - 5x_3 - 2x_4 \\ \text{s.t.} \quad & 3x_1 + 2x_2 + 2x_3 + x_4 \leq 4 \\ & x_i \in \{0, 1\} \end{aligned}$$



Lower bounding and upper bounding solutions

For a minimization problem:

- LP relaxations lower bounds the true opt.
- Any found integral solution upper bounds the true solution
- Prune if a branch cannot beat the current best upper.

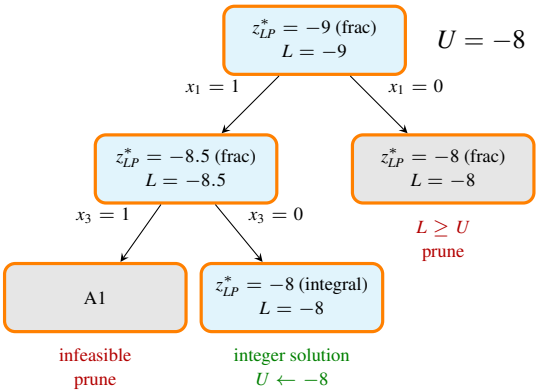


Branch-and-bound: pruning paths (example)

$$
\begin{aligned}
\min \quad & -6x_1 - 2x_2 - 5x_3 - 2x_4 \\
\text{s.t.} \quad & 3x_1 + 2x_2 + 2x_3 + x_4 \leq 4 \\
& x_i \in \{0, 1\}
\end{aligned}
$$



Branch-and-bound: pruning paths (example)



Optimal solution

$$x^* = (1, 0, 0, 1), \qquad U = -8$$

Algorithmic loop

Branch-and-bound (minimization) problem for node A, given lower bound L and best value U thus far:

- If $L \geq U$, then quit (prune the branch).
- Find (z^*_{LP}, x^*), update $L \leftarrow \max(L, z^*_{LP})$.
- If $L \geq U$, then quit (prune the branch).
- If x^* integral, update upper bound $U \leftarrow z^*_{LP}$ and return x^*.
- Else branch on some fractional x_j, searching

$$x_j = 0, \qquad x_j = 1$$

with local L and the globally latest U.



Traffic engineering

Integer linear programs

Branch and Bound

Traveling salesman

Conclusion

modeling

- given a weighted digraph $G = (V, E, w)$, find the shortest tour visiting all nodes exactly once
- decision variables: $x_e \in \{0, 1\}$
- constraints: degree

$$\sum_{u \in N^-(v)} x_{u \to v} = 1 \qquad \sum_{u \in N^+(v)} x_{v \to u} = 1$$

- constraints: connected
  - order vertices in increasing, according to the edges (excluding vertex 1)
  - decision variables y_v for each v

$$x_{u \to v} = 1 \implies y_v \geq y_u + 1$$

  - representing as a linear constraint [1]

$$y_v - y_u - 1 \geq -n(1 - x_{u \to v})$$

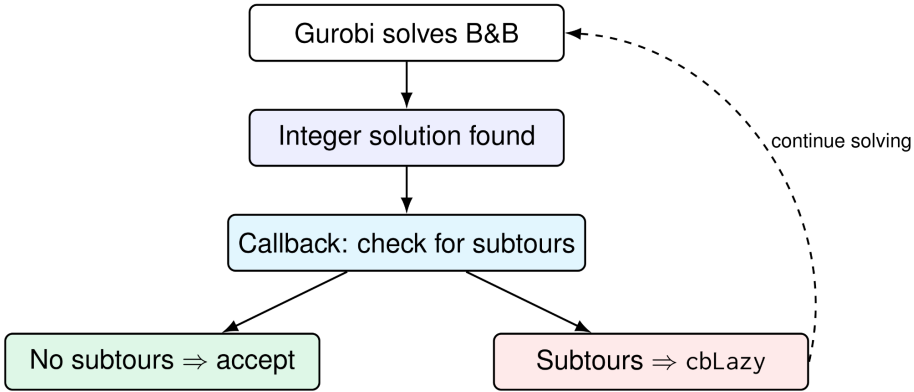
[1] Gurobi has special implication variables for pattern; here we do it by hand

formulation strength

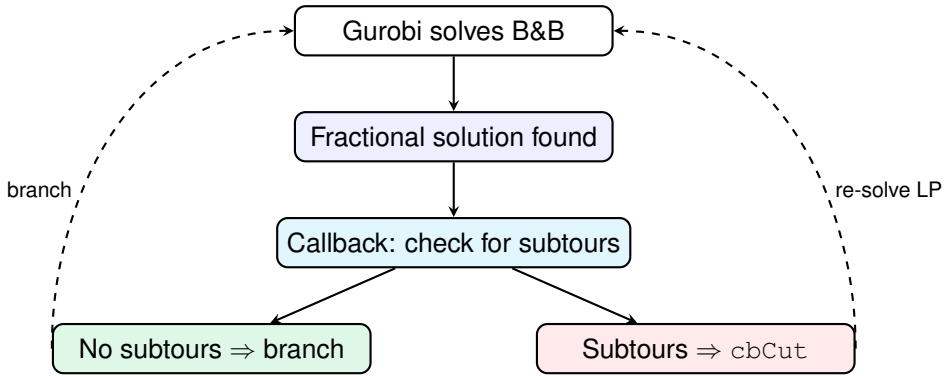
- large integrality gap: very small relaxed solutions means weak bounds
- enforce stronger topological constraints on the tour:
  - For each $\emptyset \subsetneq S \subsetneq V$, then at least one edge entering/leaving S
- much stronger constraints (smaller integrality gap)[2], but now we have exponentially many constraints to check

[2] in fact, if TSP weights are a metric, then LP is provable $3/2$-approximation

lazy constraints



branch-and-cut



Credits and further reading

- Slides closely based on material from Farouk's CS 498 AE
- Gurobi reference manual: docs.gurobi.com/projects/optimizer/en/13.0/index.html
- TSP developments: *Cook, William J. In pursuit of the traveling salesman: mathematics at the limits of computation. Princeton university press, 2015.*
- Google blog post on optimization: research.google/blog/google-research-2022-beyond-algorithmic-advances/
- Meta blog post on optimization: engineering.fb.com/2026/09/21/open-source/rebalancer-generic-high-performance-library-assignment-problems/
- LinkedIn blog post on optimization: www.linkedin.com/blog/engineering/scalability/dualip-solving-extreme-scale-linear-programs

