Single flow routing through network

Pramesh Kumar

IIT Delhi

March 5, 2025

Suggested reading

- Goetschalckx Chapter 6

Vehicle routing classification

The classification is based on three characteristics:

1. whether one or more vehicles are used
   - if one vehicle is used then then there is no assignment decision on which task has to be executed by which vehicle.
2. whether the status of vehicle is required before and after the task execution
   - The class of problems in which status of the vehicle is not required is called origin-destination flow routing. Single vehicle routing between two locations is shortest path problem.
3. whether the problem considers multiple time periods
   - Dynamic routing problems consider multiple time periods

Number of Vehicles	Single	Multiple
Vehicle Status		
Request-Only (Flow)	Single Flow (SPP)	Multiple Flows (NFP)
Prior & Post (Vehicle)	Single Vehicle Roundtrip (TSP)	Multiple Vehicles Roundtrip (VRP)

Figure: Routing problem classification (Goetschalckx Chapter 6)

Shortest path

- Fundamental problem with numerous applications.
- Appears as a subproblem in many network flow algorithms.
- Easy to solve.

Outline

Shortest path problem

Definition (Path cost). The cost of a directed path $P = (i_1, i_2, ..., i_k)$ is the sum of cost of its individual links, i.e., $c(P) = \sum_{i=1}^{k-1} c_{i,i+1}$.

Definition (Shortest Path Problem). Given $G(N, A)$, link costs $c : A \mapsto \mathbb{R}$, and source $s \in N$, the shortest path problem (also known as single-source shortest path problem) is to determine for every non-source node $i \in N \backslash \{s\}$ a shortest cost directed path from node s.

OR

Definition (Shortest Path Problem). Given $G(N, A)$, link costs $c : A \mapsto \mathbb{R}$, and source $s \in N$, the shortest path problem is to determine how to send 1 unit of flow as cheaply as possible from s to each node $i \in N \backslash \{s\}$ in an uncapacitated network.

LP formulation

Primal

$$\min_{\mathbf{x}} \sum_{(i,j)\in A} c_{ij}x_{ij}$$

$$\text{s.t.} \sum_{j\in FS(i)} x_{ij} - \sum_{j\in BS(i)} x_{ji} = \begin{cases} n-1 & \text{if } i=s \\ -1 & \forall i \in N\backslash\{s\} \end{cases}$$

$$x_{ij} \geq 0, \forall (i,j) \in A$$

Types of shortest path (SP) problems

1. *Single-source shortest path*: SP from one node to all other nodes (if exists)
   1.1 with non-negative link costs.
   1.2 with arbitrary link costs.
2. *Single-pair shortest path* SP from between one node and another node.
3. *All-pairs shortest path* SP from every node to every node.
4. *Various generalizations of shortest path*:
   - Max capacity path problem
   - Max reliability path problem
   - SP with turn penalties
   - Resource-constraint SP problem
   - and many more

Outline

Single-source shortest path

Assumptions

1. Network is directed
2. Link costs are integers
3. There exists a directed path from s to every other node (can be satisfied by creating an artificial link from s to other nodes)
4. The network does not contain a negative cycle.

Remark. For a network containing a negative cycle reachable from s, the above LP will be unbounded since we can send an infinite amount of flow along that cycle.

Can SP contain a cycle?

1. It cannot contain negative cycles.
2. It cannot contain positive cycles since removing the cycle produces a path with lower cost.
3. One can also remove zero weight cycle without affecting the cost of SP.

Label setting and label correcting algorithms

- Shortest path algorithms assign tentative distance label to each node that represents an upper bound on the cost of shortest path to that node.
- Depending on how they update these labels, the algorithms can be classified into two types:
  1. Label setting
  2. Label correcting
- Label setting algorithms make one label permanent in each iteration
- Label correcting algorithms keep all labels temporary until the termination of the algorithm.
- Label setting algorithms are more efficient but label correcting algorithms can be applied to more general class of problems.

Dijkstra's algorithm

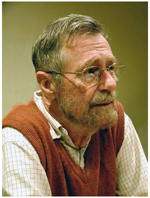
A label setting algorithm

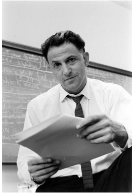
Input: Graph $G(N, A)$, costs c, and source s
Output: Optimal cost labels d and predecessors $pred$
procedure $\textsc{Dijkstra}(G, c, s)$
    $S \leftarrow \phi; T \leftarrow N$
    $d(i) \leftarrow \infty, \forall i \in N\{s\}; d(s) \leftarrow 0$
    $pred(i) \leftarrow \texttt{NA}, \forall i \in N\backslash\{s\}; pred(s) \leftarrow 0$
    while $T \neq \phi$ **do**
        Choose a node i with minimum $d(i)$ from T
        $S \leftarrow S \cup \{i\}; T \leftarrow T\backslash\{i\}$
        for $j \in FS(i)$ **do**
            if $d(j) > d(i) + c_{ij}$ **then**
                $d(j) \leftarrow d(i) + c_{ij}$
                $pred(j) \leftarrow i$
            end if
        end for
    end while
end procedure

Label correcting algorithm

Input: Graph $G(N, A)$, costs c, and source s
Output: Optimal cost labels d and predecessors $pred$
procedure $\textsc{LabelCorrecting}(G, c, s)$
    $SEL = \{s\}$
    $d(i) \leftarrow \infty, \forall i \in N\{s\}; d(s) \leftarrow 0$
    $pred(i) \leftarrow \texttt{NA}, \forall i \in N \setminus \{s\}; pred(s) \leftarrow 0$
    while $SEL \neq \phi$ **do**
        Remove an element i from SEL
        for $j \in FS(i)$ **do**
            if $d(j) > d(i) + c_{ij}$ **then**
                $d(j) \leftarrow d(i) + c_{ij}$
                $pred(j) \leftarrow i$
                if j not in SEL **then**
                    $SEL = SEL \cup \{j\}$
                end if
            end if
        end for
    end while
end procedure

Origins of above algorithms





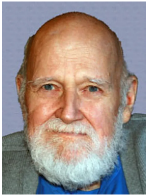


Figure: (From left to right) Edsger W. Dijkstra, Richard E. Bellman, Lester Randolph Ford Jr. (Pictures sources: Wiki, stanford.edu, and independent.com/)

Thank you!