Shortest Paths: Applications, Variations and Optimization

Natashia Boland

Irina Dumitrescu

Mathematics and Statistics
University of Melbourne
&
HEC Montreal, Canada Research Chair in
Distribution Management

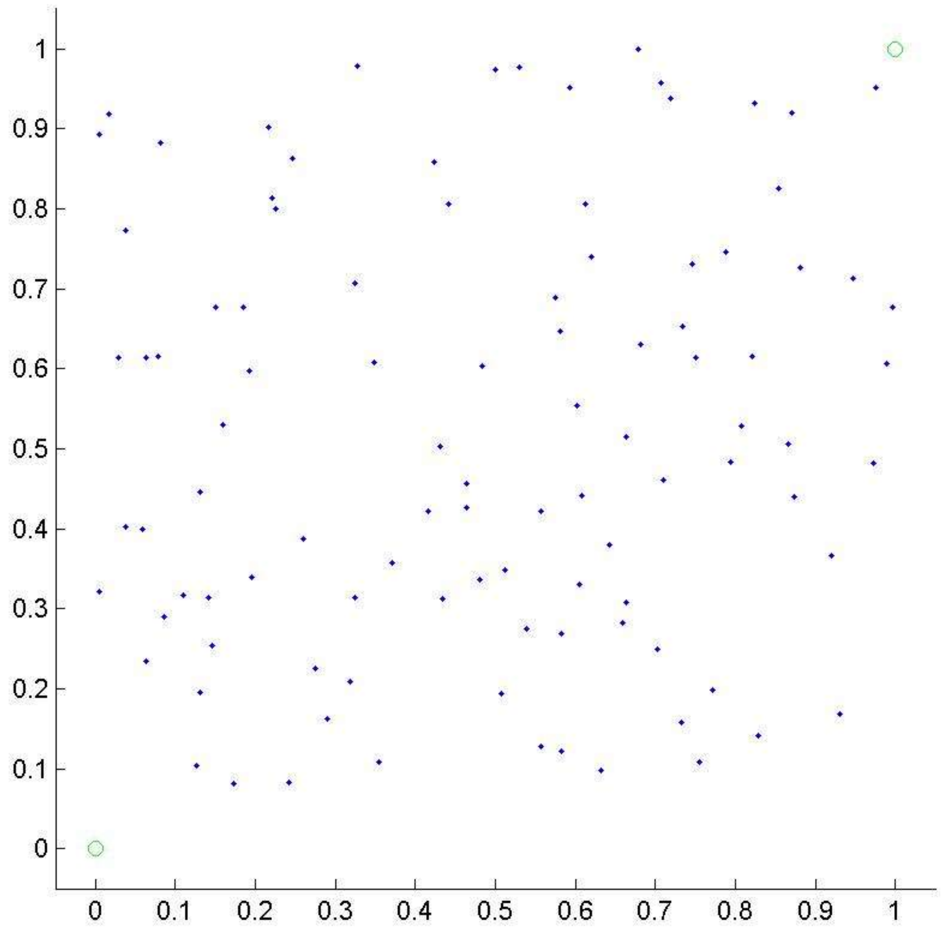
Outline

- Applications of shortest paths
  - Minefield path planning
  - Approximating piecewise linear functions
  - Systems of difference constraints
  - DNA sequence alignment
- Optimization: linear programming formulation
- Variations of shortest paths
  - Resource constraints
  - Elementary paths

Minefield Path Planning

- Find safe path through naval minefield
  - Given starting point and end point
  - Location of mines is known accurately
- Each mine is considered a potential threat
- Risk from a mine decreases with distance from it
- Risk at each point in space is total risk from all mines
- Total risk for a path is accumulated along the path
- Minimize risk

100 Random Mines



A Risk Model

- The risk (probability of damage) from the ith threat (mine) $r_i(x)$ at any point x in space is proportional to the inverse square of the distance from it:

$$r_i(x)=\sigma_i(d_i(x))^{-2}$$

threat-dependent constant

distance from x to mine i

- The total risk is the sum of the risks:

$$r(x) = \sum_i r_i(x)$$

The Risk Model

- Loosely speaking, the probability of successfully completing the path P, defined as a set of points in space, is

$$\prod_{x \in P} (1\text{-}r(x))$$

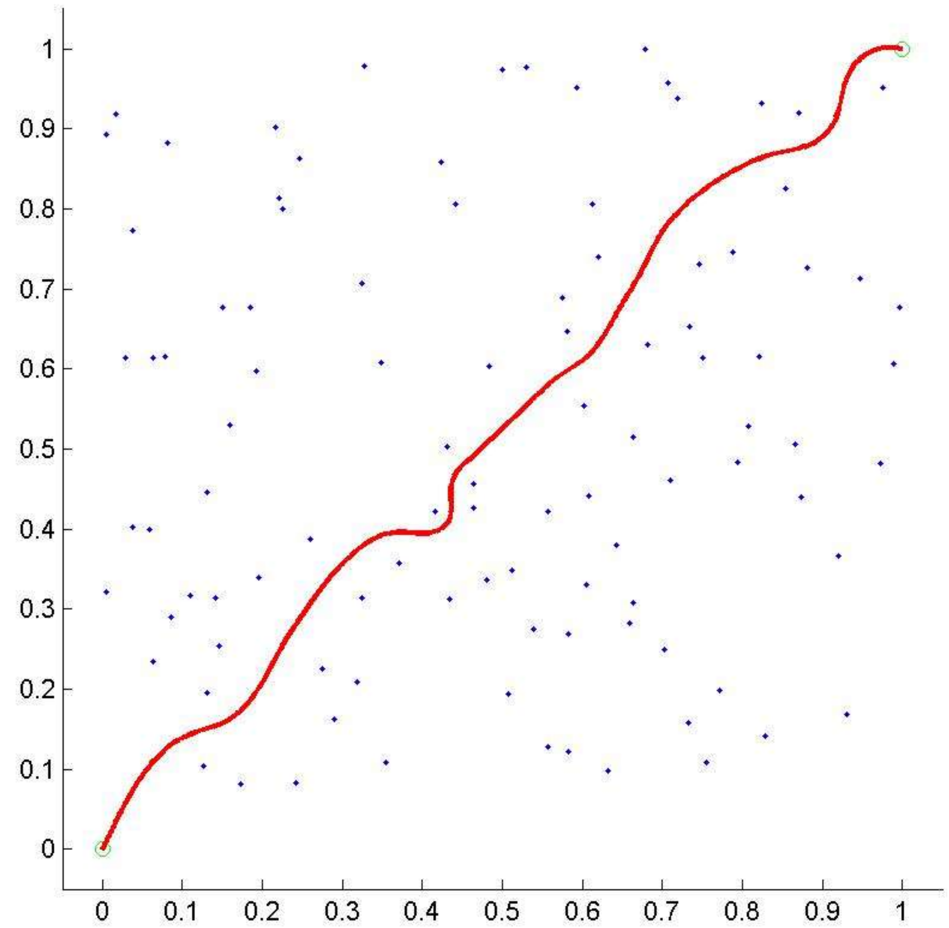
where $r(x)$ is the risk at point x.

- Maximizing this is equivalent to minimizing

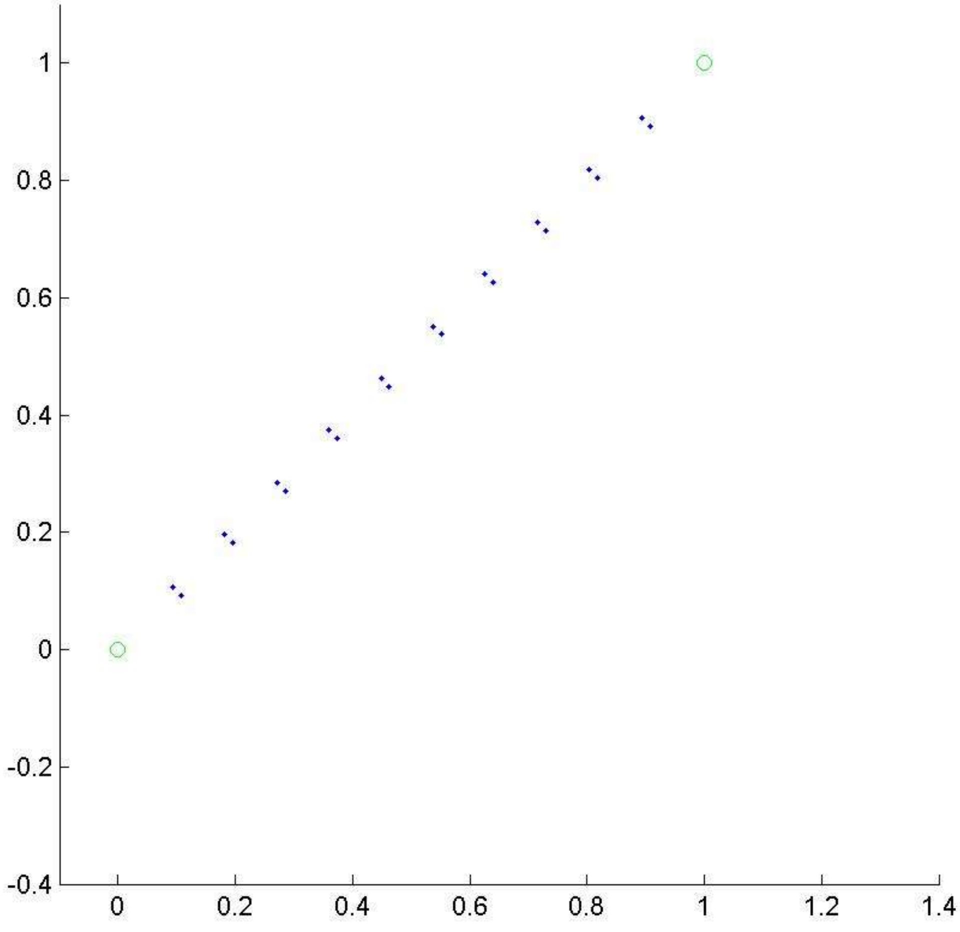
$$\int_{x \in P} c(x)dx = \int_{x \in P} (-log(1\text{-}r(x)))\ dx$$

where $c(x)$ is the “additive” risk at point x.

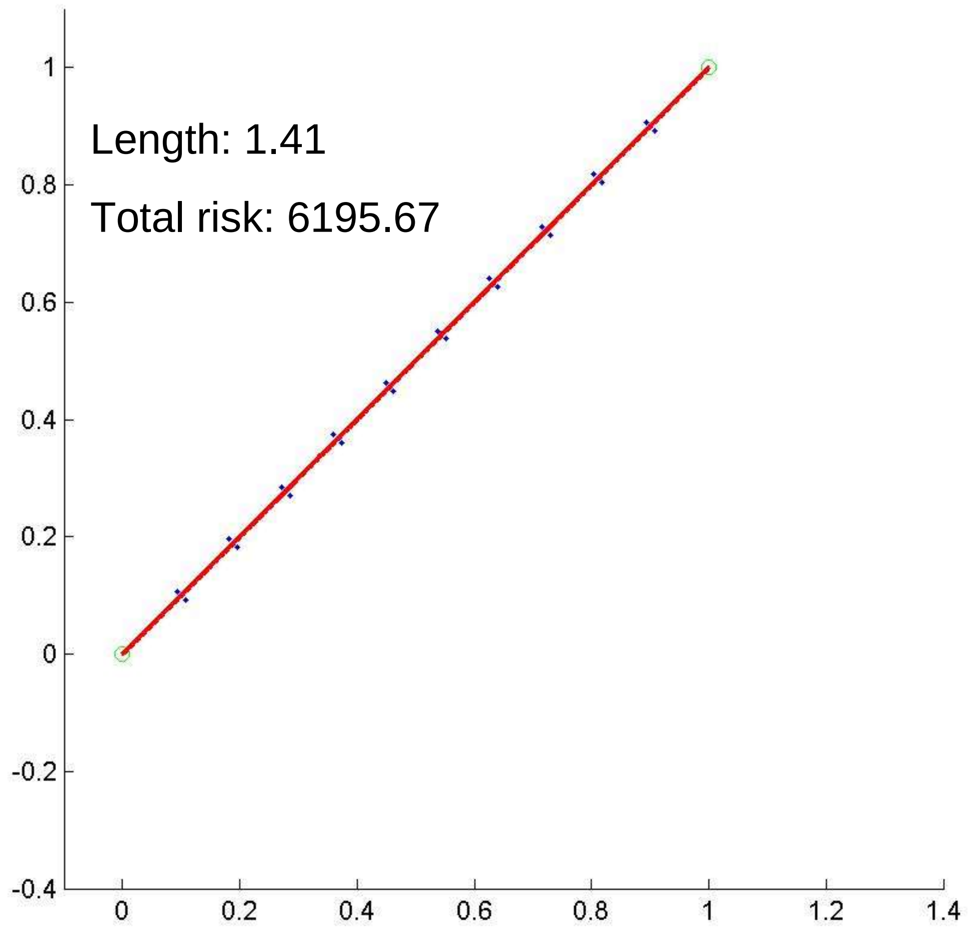
Optimal control path



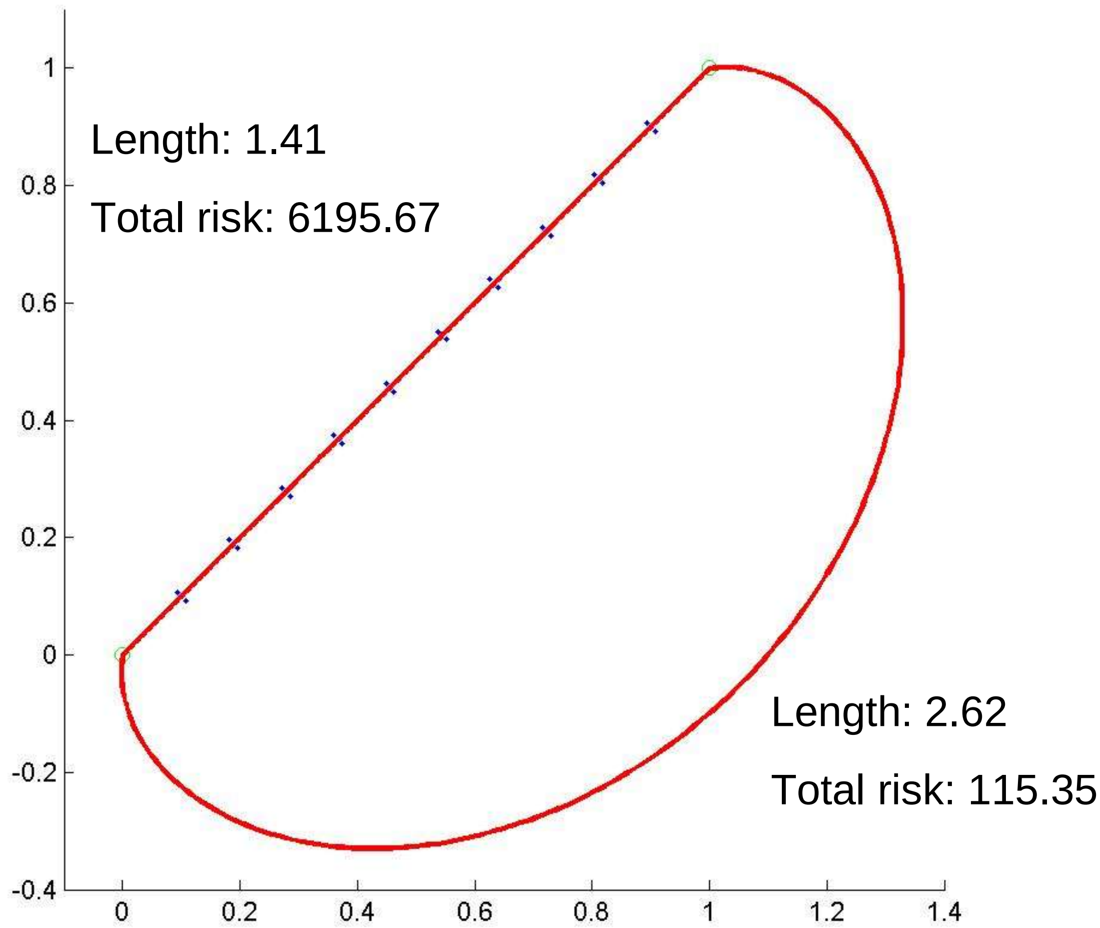
Corridor of Mines



A Local Optimum



Alternative



Local vs Global

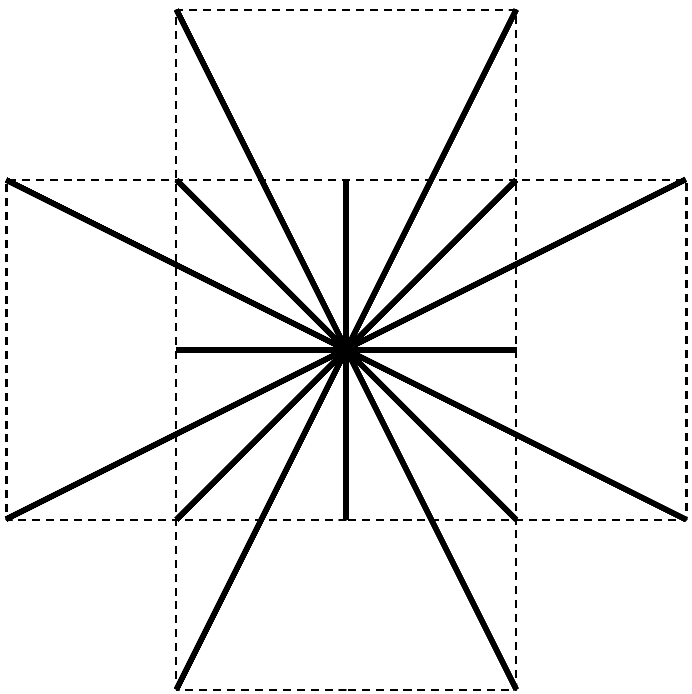
- Continuous approaches can only yield locally optimal solutions
- These may be far from globally optimal
- Space discretization gives an approximate problem – but means we can find a globally optimal solution

Network Model

- Discrete points in space form vertex set V
- Arcs A connect selected pairs of vertices
- Directed graph space discretization G=(V,A)

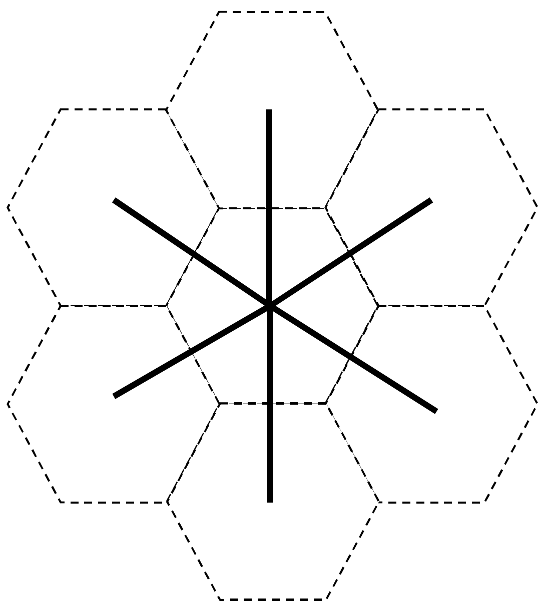
Regular Grid Discretization

- “Optimal Risk Path Algorithms”
    - Zabarankin, Uryasev and Pardalos



Hexagonal Grid Discretization

- "Path Planning For Unmanned Aerial Vehicles in Uncertain and Adversarial Environments"
  - Jun and D'Andrea



Network Models

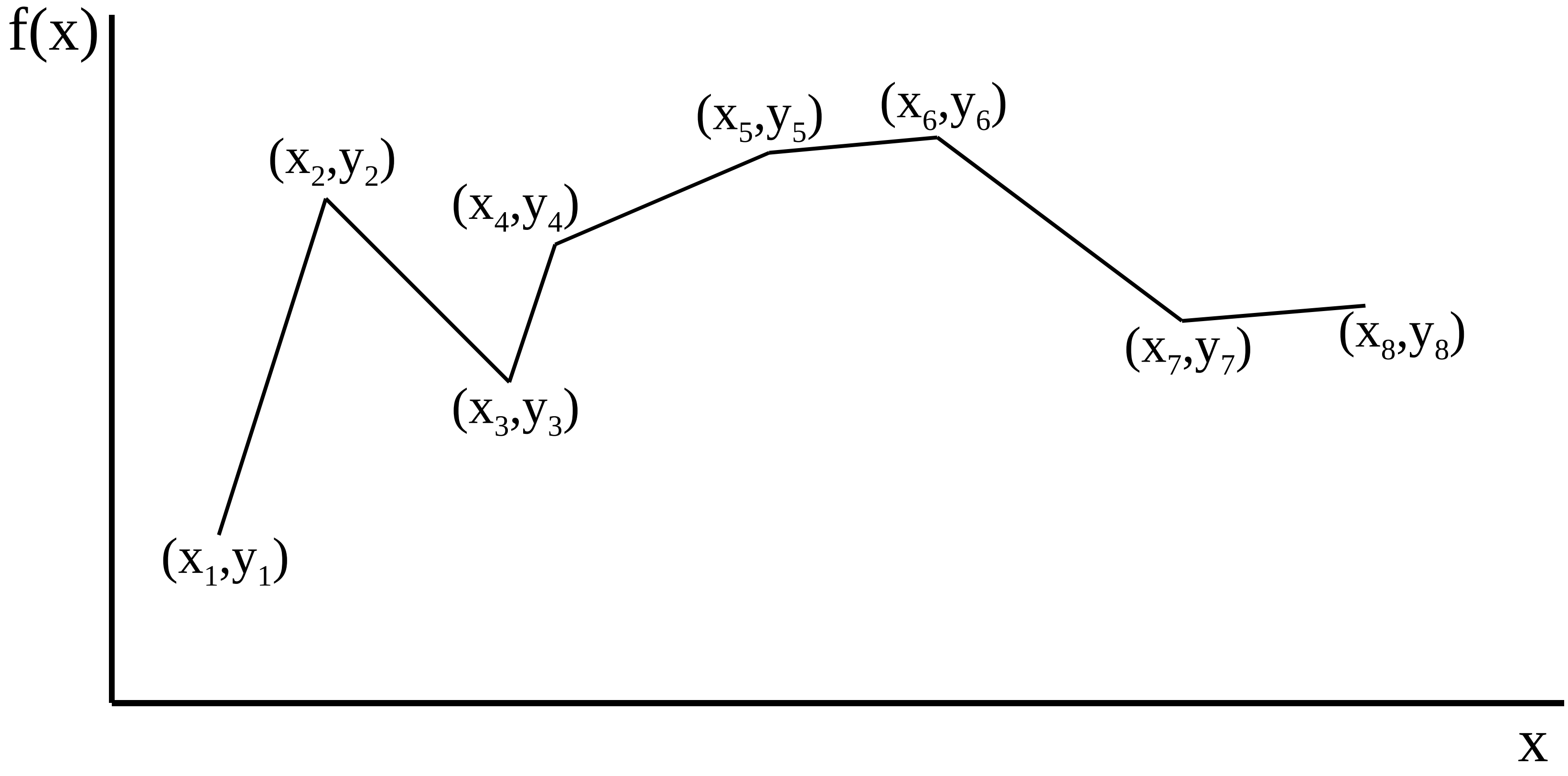
- Each arc has risk value given by the integral of c along the line segment joining the two vertices that form the endpoints of the arc

- Shortest path algorithms find minimum risk paths in graph G(V, A)

- These ideas can be applied to many similar situations
  - Submarine sonar avoidance path planning
  - Military aircraft radar avoidance path planning
  - Commercial aircraft flight planning
  - Highway or railway construction

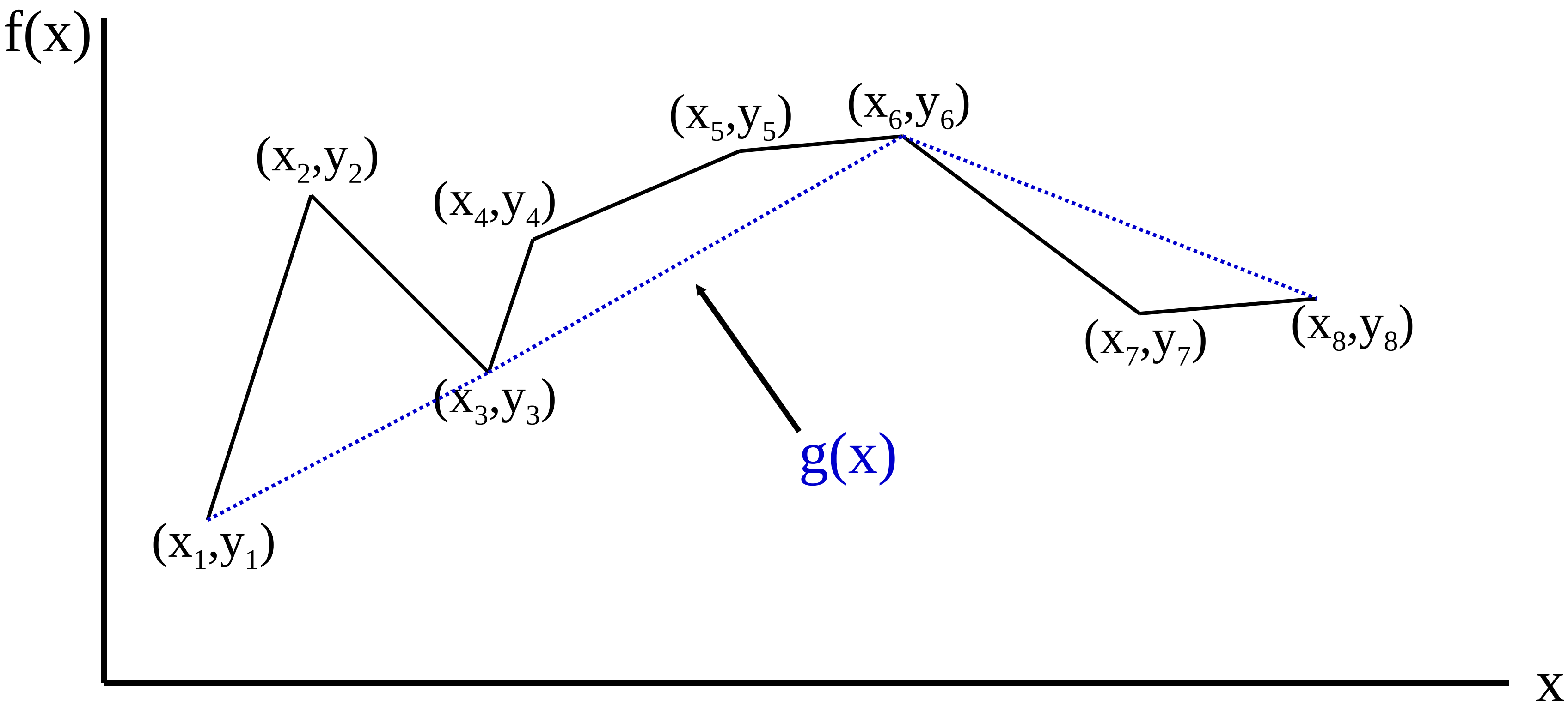
Function Approximation

- $f(x)$ a piecewise linear function defined by n points $(x_1,y_1), (x_2,y_2),\ldots,(x_n,y_n)$, where $x_1 \le x_2 \le \ldots \le x_n$



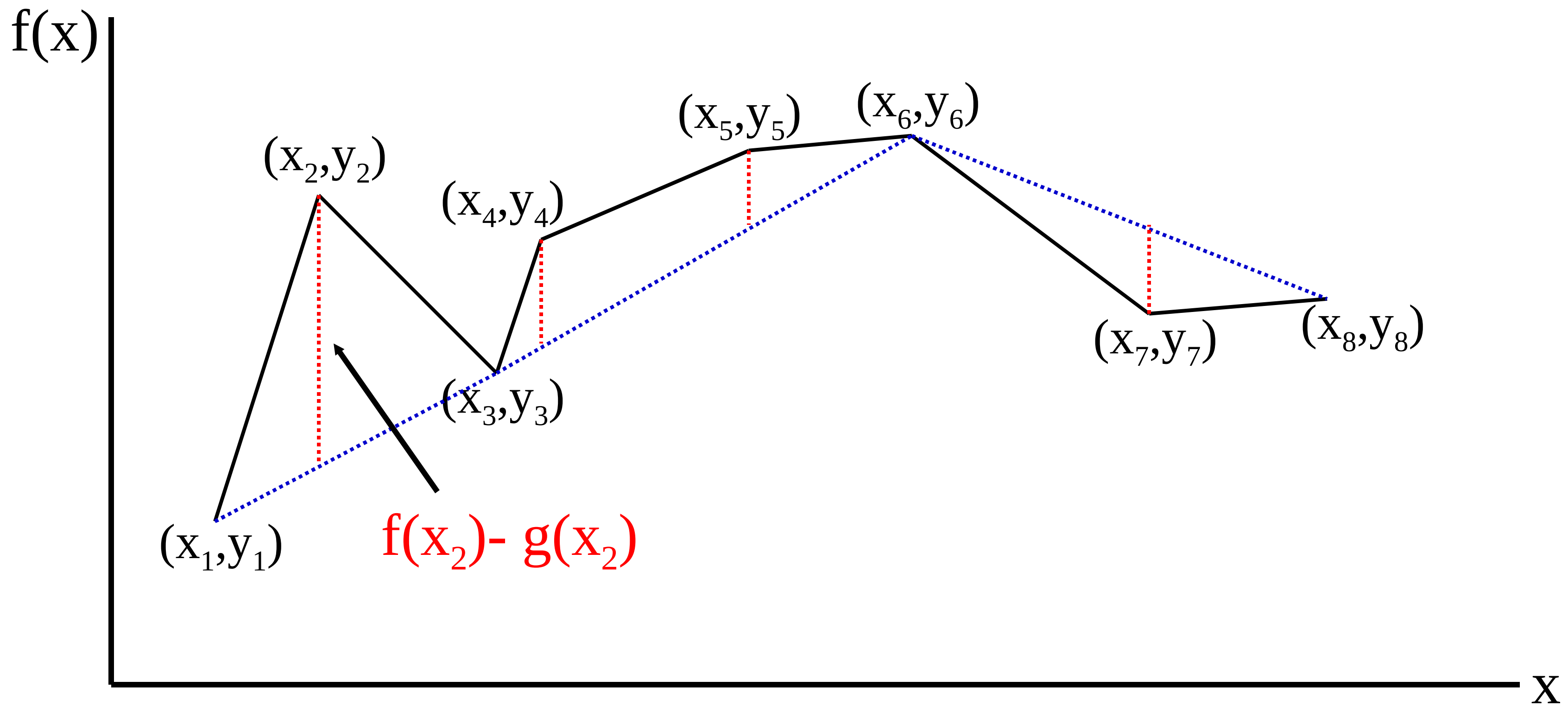
Function Approximation

- n is very large: approximate $f(x)$ by $g(x)$
- Restrict attention to $g(x)$ also piecewise linear defined by (x_1,y_1), (x_n,y_n) and a subset of $(x_2,y_2),\ldots,(x_{n-1},y_{n-1})$



How Good Is It?

- Quality of approximation: measure by $[f(x_i)-g(x_i)]^2$
- Find $g(x)$ to minimize $\sum_{i=1}^{n}[f(x_i)-g(x_i)]^2$ $\times\,\alpha$
- Trade off vs number of points used to define $g(x)$ $\times\,\beta$



Network Model

- Node i for each point $i=1,\ldots,n$
- Arc (i,j) defined for all $i,j=1,\ldots,n$ with $i < j$
- Arc (i,j) indicates possibility that $g(x)$ uses point (x_i,y_i) and then point (x_j,y_j), i.e. skips points

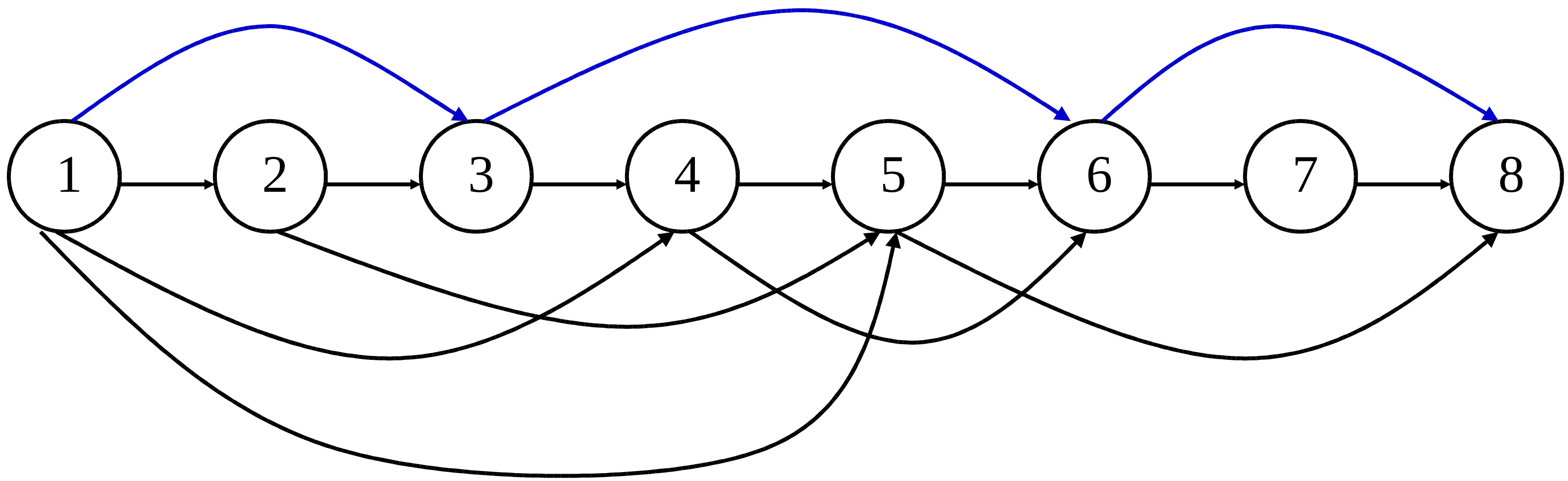
$$(x_{i+1},y_{i+1}),\ldots,(x_{j-1},y_{j-1})$$

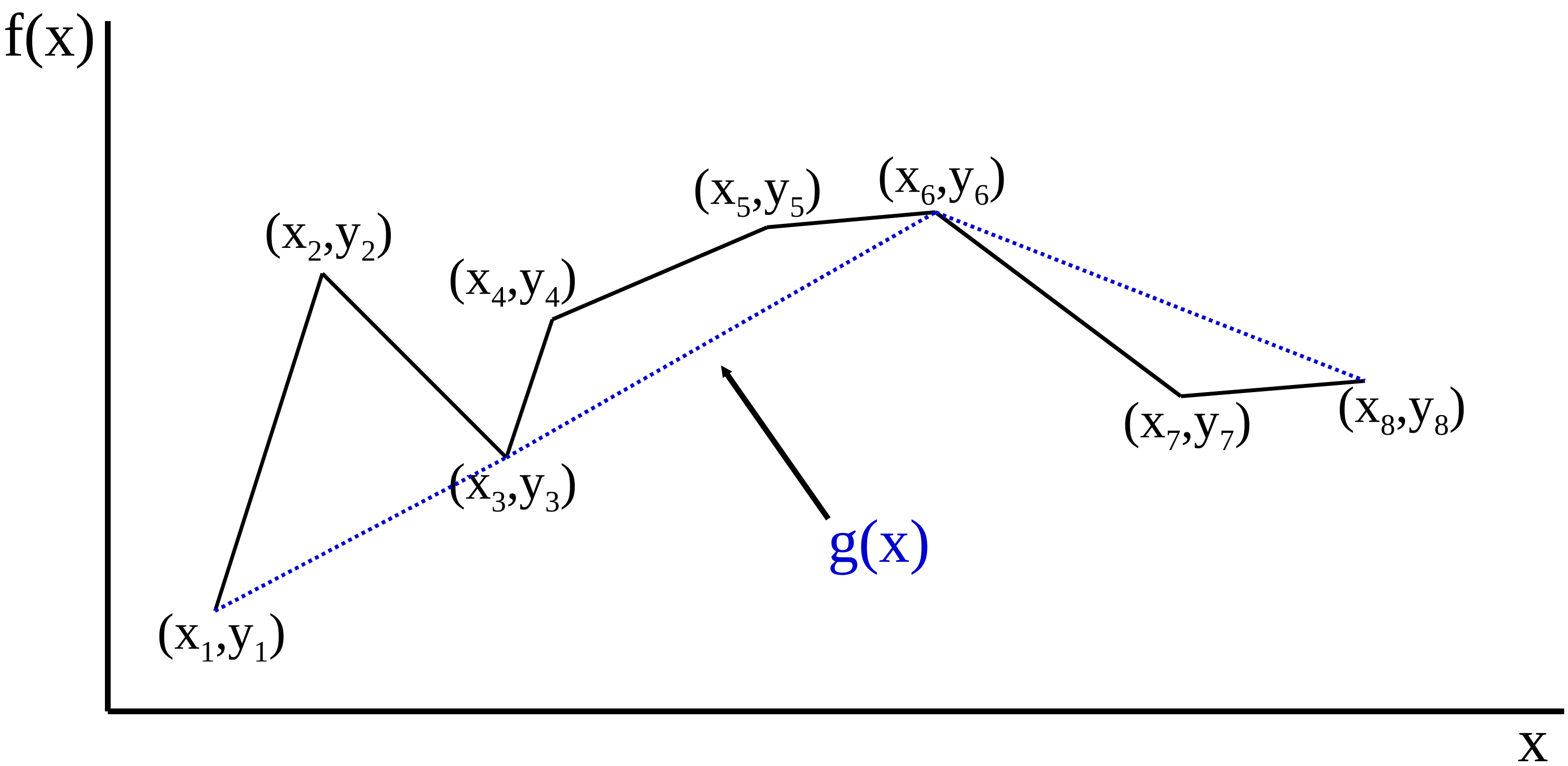
- Arc (i,j) has cost (length)

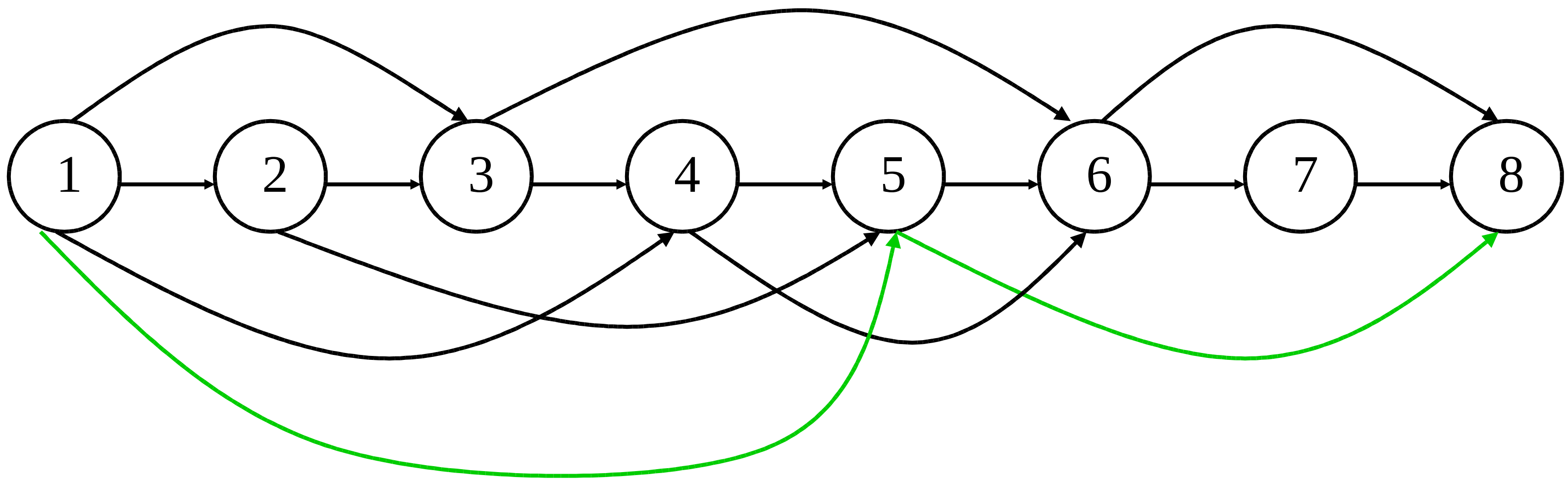
$$\alpha \sum_{k=i+1}^{j-1} [f(x_k)-g(x_k)]^2 + \beta$$

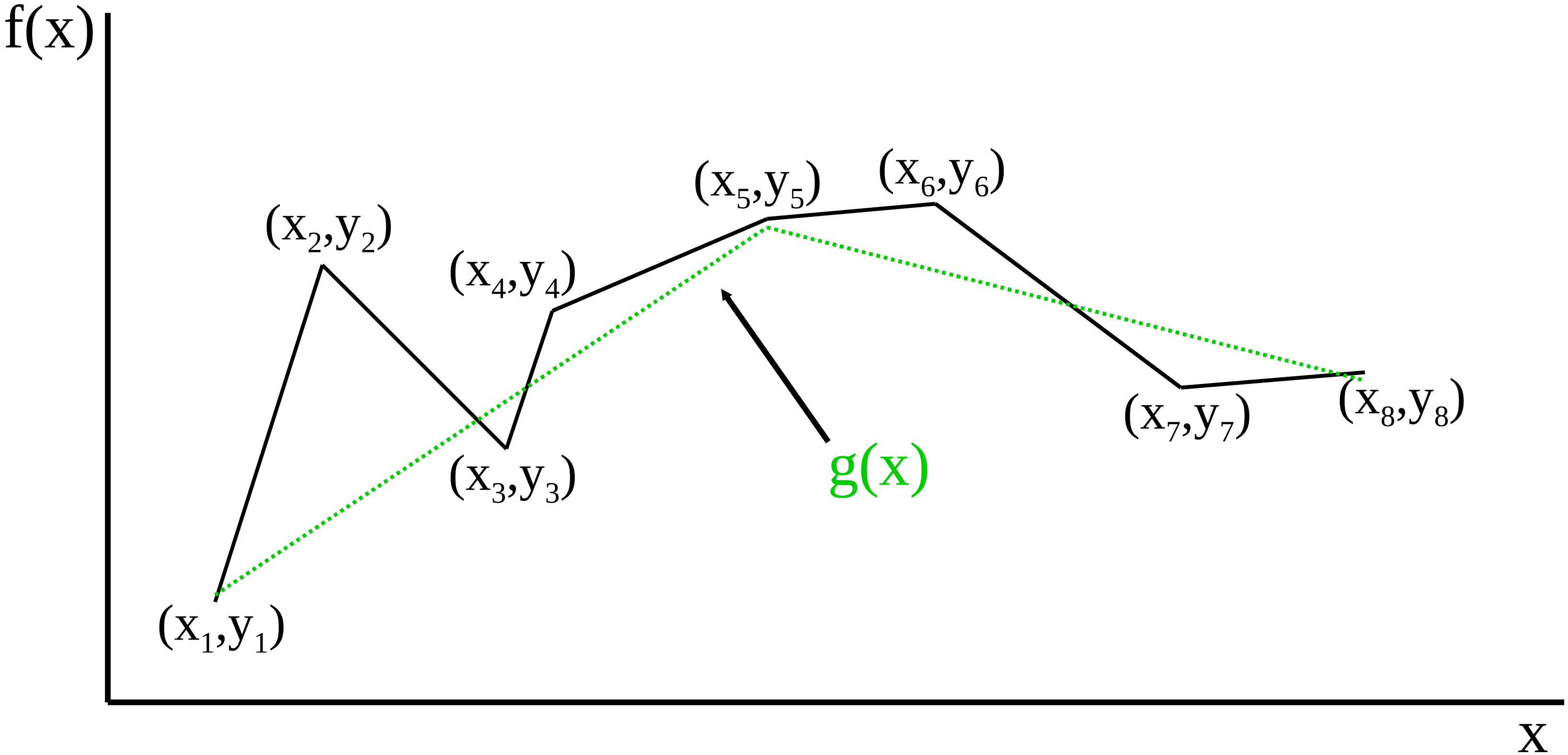
(only if $j \neq n$)

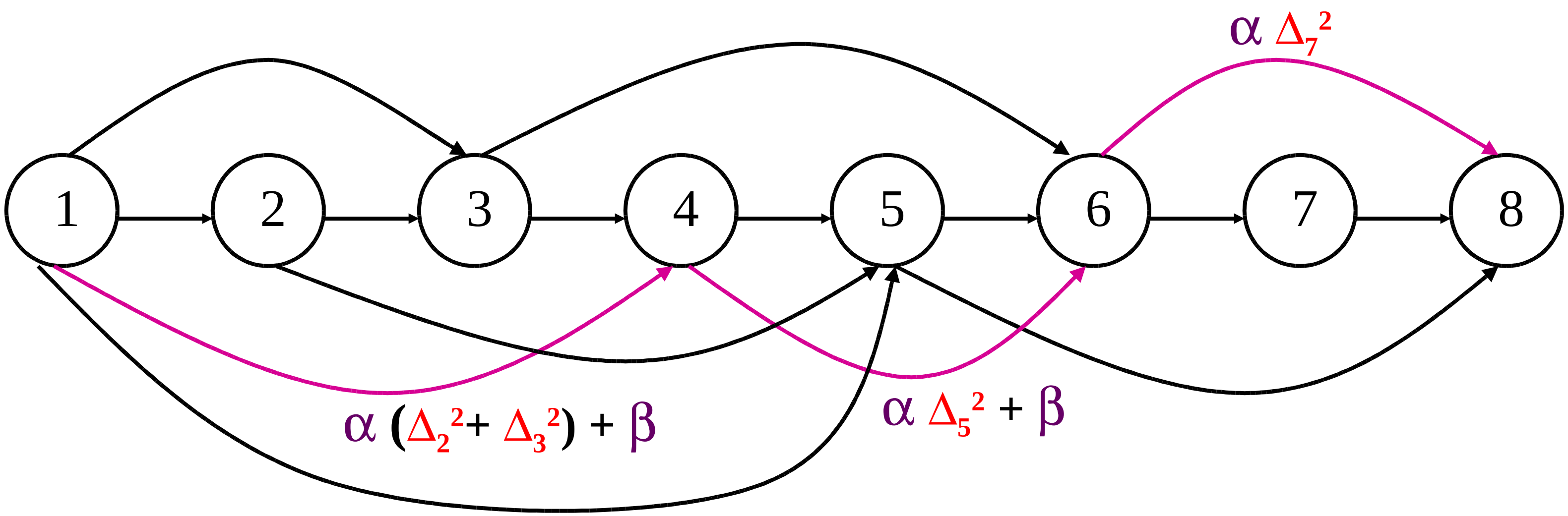
- Find shortest path from node 1 to node n


1
2
3
4
5
6
7
8

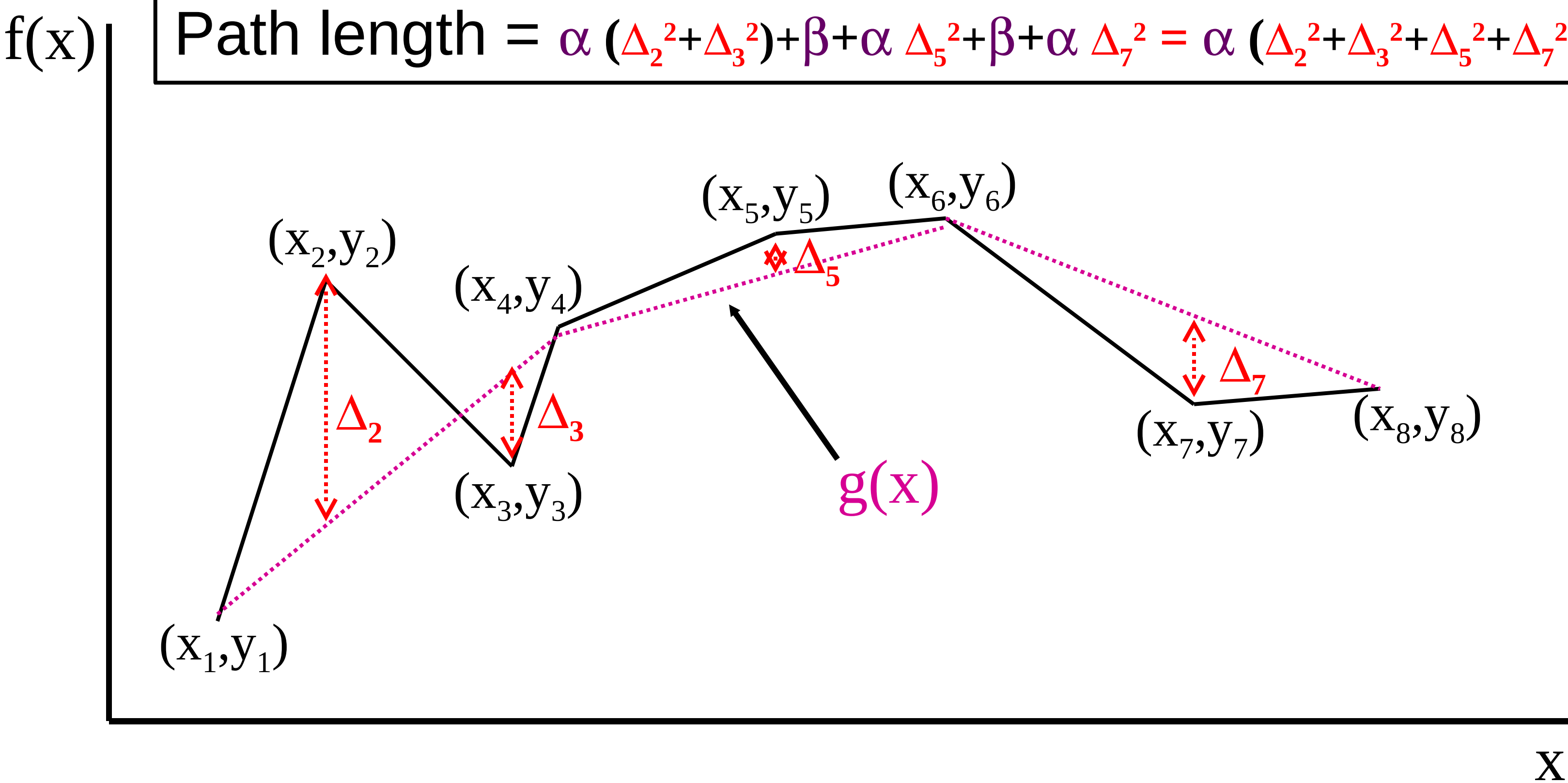

f(x)
x
(x_1,y_1)
(x_2,y_2)
(x_3,y_3)
(x_4,y_4)
(x_5,y_5)
(x_6,y_6)
(x_7,y_7)
(x_8,y_8)
g(x)


1
2
3
4
5
6
7
8


f(x)
(x1,y1)
(x2,y2)
(x3,y3)
(x4,y4)
(x5,y5)
(x6,y6)
(x7,y7)
(x8,y8)
g(x)
x



$$\text{Path length} = \alpha\,(\Delta_2^2+\Delta_3^2)+\beta+\alpha\,\Delta_5^2+\beta+\alpha\,\Delta_7^2 = \alpha\,(\Delta_2^2+\Delta_3^2+\Delta_5^2+\Delta_7^2)+2\beta$$



Systems of Difference Constraints

- Variables $x_1, x_2, \ldots, x_n$
- Constraints $x_{j_k} - x_{i_k} \leq b_k$, $k=1,\ldots,m$
- Does this system of constraints have a feasible solution, or not?

Applications

- Workforce scheduling
  - Fluctuating hourly or daily demands
  - Cyclical demands
  - Consistent shift structure or work patterns, e.g. 5 days on, 2 days off per week
  - Minimize number of workers needed to meet demands
- Other similar, e.g. trucks, maintenance

Example

- Variables x_1, x_2, x_3, x_4
- Constraints

$$x_3 - x_4 \leq 5$$

$$x_4 - x_1 \leq -10$$

$$x_1 - x_3 \leq 8$$

$$x_2 - x_1 \leq -11$$

$$x_3 - x_2 \leq 4$$

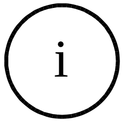
- Does this system of constraints have a feasible solution, or not?

$+\alpha$ $+\alpha$ $+\alpha$ $+\alpha$ for any α

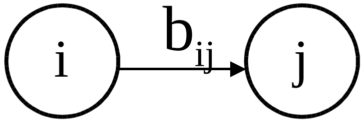
- $x_1=11$, $x_2 = 0$, $x_3 = 4$, $x_4 = 1$

Network Model

- Node i for each variable x_i



- Arc for each constraint



$$x_j - x_i \leq b_{ij}$$

- Nodes $N=\{1,\ldots,n\}$

  Arcs $A=\{(i_k,j_k) : k=1,\ldots,m\}$

  Length of arc (i_k,j_k) is b_k

- This is called the *constraint graph*

Example: Constraint Graph

- Variables x_1, x_2, x_3, x_4
- Constraints

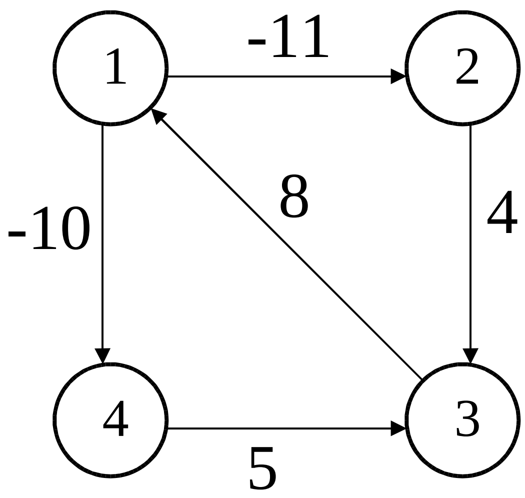
$$x_3 - x_4 \leq 5$$

$$x_4 - x_1 \leq -10$$

$$x_1 - x_3 \leq 8$$

$$x_2 - x_1 \leq -11$$

$$x_3 - x_2 \leq 4$$



Feasibility Condition

System of difference constraints is feasible

$\Updownarrow$

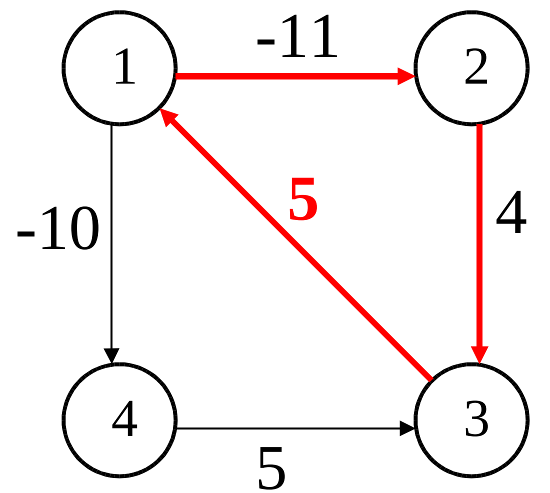
Constraint graph has no negative length cycles

Modified example

$$x_3 - x_4 \le 5$$
$$x_4 - x_1 \le -10$$
$$x_1 - x_3 \le \mathbf{5}$$
$$x_2 - x_1 \le -11$$
$$x_3 - x_2 \le 4$$

$$\left.\begin{aligned} x_2 - x_1 &\le -11 \\ x_3 - x_2 &\le 4 \\ x_1 - x_3 &\le 5 \end{aligned}\right.$$
$$\overline{\quad 0 \le -2 \quad}$$

Negative cycle constraints



Feasible Example

- Augmented constraint graph:
  - Node 0
  - Arcs (0,i) for all i=1,…,n of length 0
- Find shortest path from 0 to all nodes

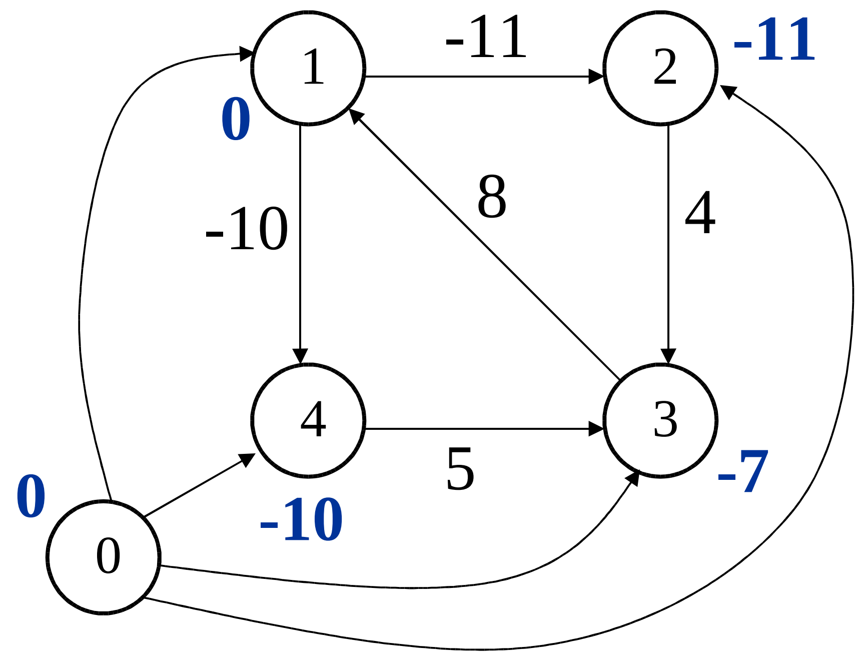
Solution:

$x_1 = \mathbf{0}$

$x_2 = \mathbf{-11}$

$x_3 = \mathbf{-7}$

$x_4 = \mathbf{-10}$



Path Optimality

- $d(j)$ = label on node j
- Optimality means $d(j) – d(i) \leq$ length of (i,j)
- Solution $x_j = d(j)$
- $x_j – x_i = d(j) – d(i) \leq$ length of $(i,j) = b_{ij}$

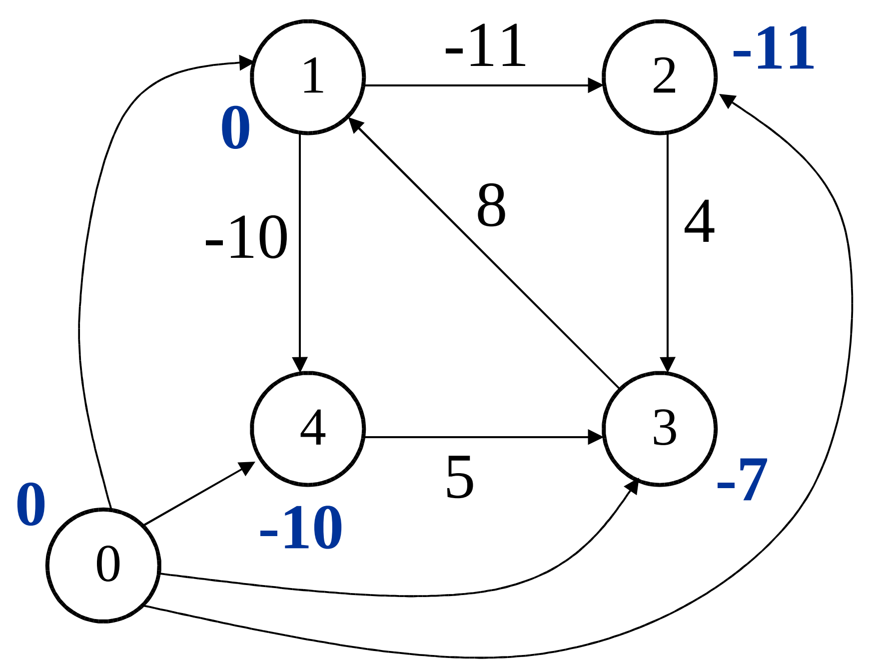
$$x_3 – x_4 \leq 5$$
$$x_4 – x_1 \leq -10$$
$$x_1 – x_3 \leq 8$$
$$x_2 – x_1 \leq -11$$
$$x_3 – x_2 \leq 4$$

$x_1 = \mathbf{0}$

$x_2 = \mathbf{-11}$

$x_3 = \mathbf{-7}$

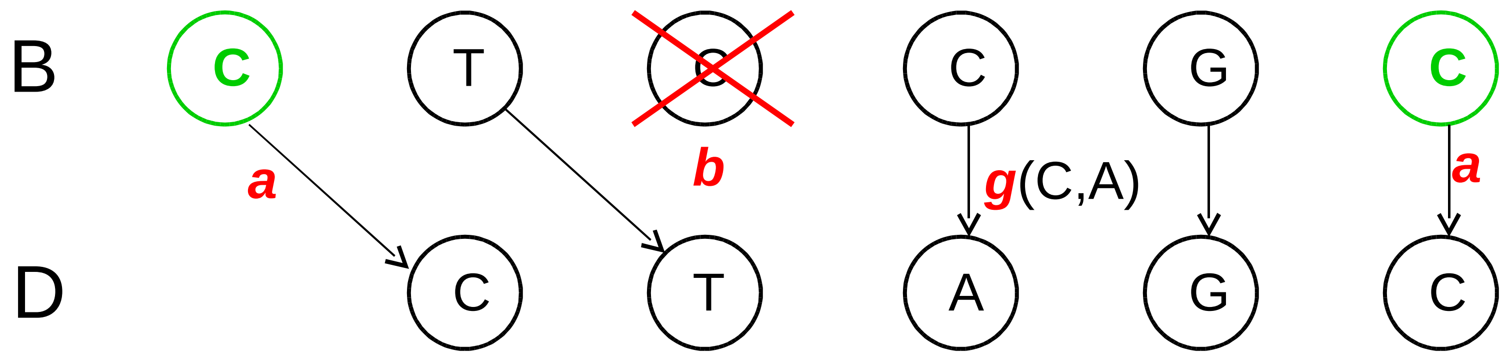
$x_4 = \mathbf{-10}$



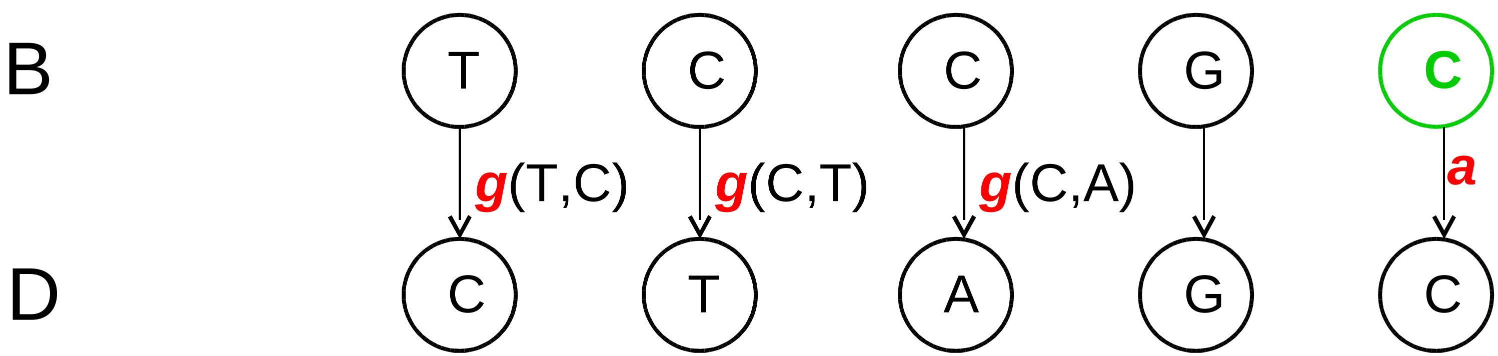
DNA Sequences

- Two sequences B and D
  - e.g. B = TCCG, D = CTAGC
- Convert B to D in a “minimal” way
  - Align matching elements in sequence
  - Insert new elements into B (cost *a*)
  - Delete elements from B (cost *b*)
  - Mutate elements in B (cost depends on elements, e.g. to convert C to G costs *g*(C,G))

Example



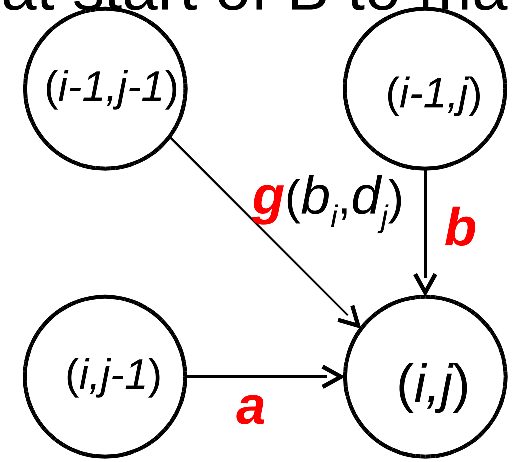
$$\text{Cost} = 2a + b + g(\text{C,A})$$



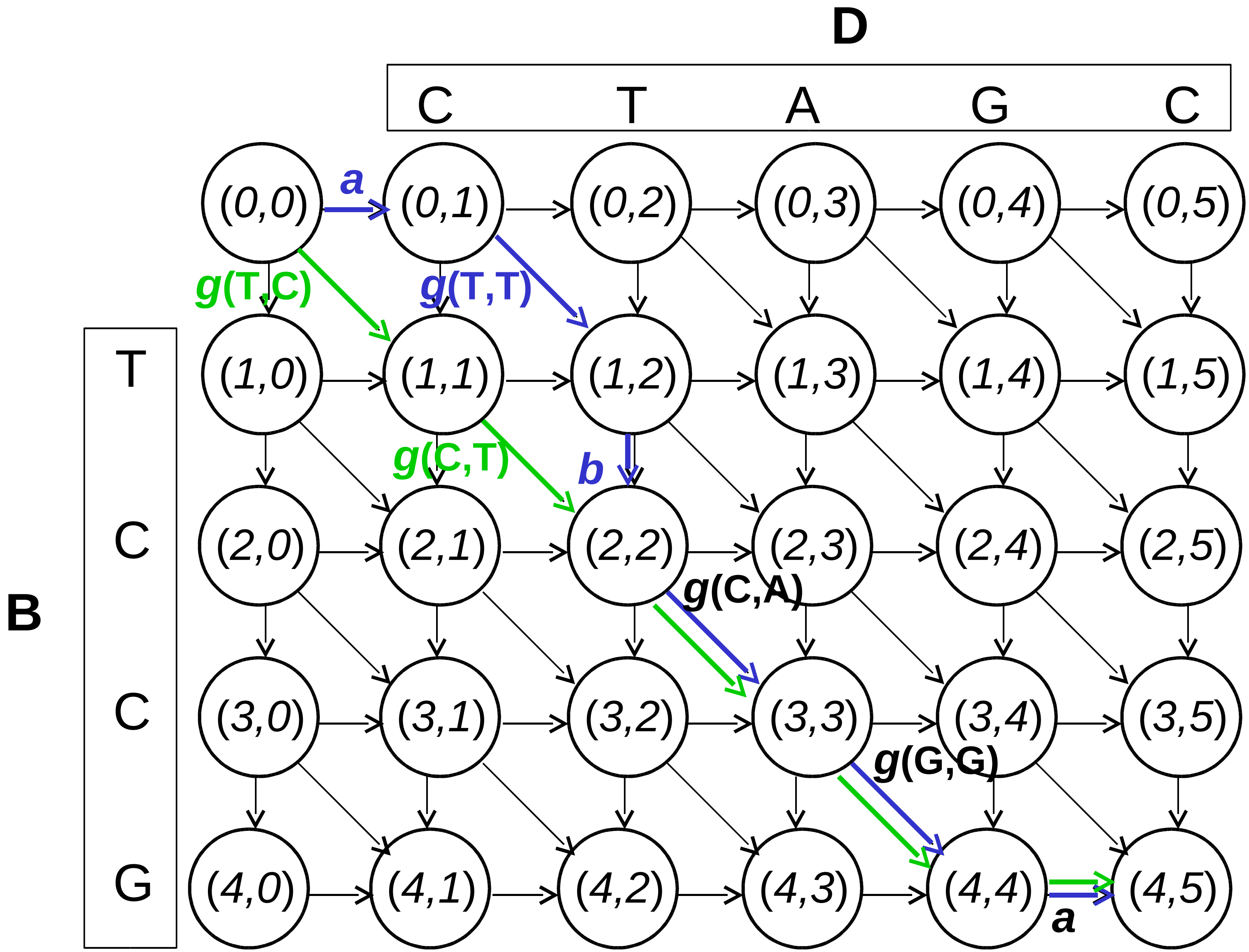
$$\text{Cost} = a + g(\text{T,C}) + g(\text{C,T}) + g(\text{C,A})$$

Network Model

- Sequence B has elements $b_1,\ldots,b_N$, sequence D has elements $d_1,\ldots,d_M$
- Node for each pair (i,j), $i=0,1,\ldots,N$, $j=0,1,\ldots,M$
  - represents decision about matching element b_i with d_j
  - $(0,0)$ is the start node, with nothing matched
  - "$j=0$" indicates deleting element at start of B up to b_{i+1}
  - "$i=0$" indicates adding new elements at start of B to match up to d_{j+1}
- Arc for each option:
  - Match (diagonal arc)
  - Insert new element (horizontal arc)
  - Delete element (vertical arc)



Example Network



(Assume
g(X,X)=0)

Shortest path model

- Each path from (0,0) to (N,M) represents a different way to convert sequence B to sequence D
- Least cost path corresponds to best match

Other Applications

- There are numerous applications of shortest path problems
- Shortest paths are often solved as subproblems in other procedures
- See “Network Flows” by R. Ahuja et al., Chapter 4, and references therein

Linear Programming

- The shortest path problem can be formulated as a network flow

  Digraph $G=(V,A)$

  Arc costs or lengths c_{ij}, for all $(i,j) \in A$

  Path start node $s \in V$, end node $t \in V$, $s \neq t$

- Variables $x_{ij} = 1$ if (i,j) in path, 0 otherwise

- Net flow into node $i = \sum_{j\,:\,(j,i) \in A} x_{ij} - \sum_{j\,:\,(i,j) \in A} x_{ij}$

LP Formulation

$$\min \sum_{(i,j) \in A} c_{ij} x_{ij}$$

s.t.

$$\sum_{j : (j,i) \in A} x_{ji} - \sum_{j : (i,j) \in A} x_{ij} = \begin{cases} -1, & i=s \\ 0, & i \neq s,t \\ 1, & i=t \end{cases} \qquad \forall i \in V$$

$$x \geq 0$$

Duality Results

- LP dual has variables u_i for each node $i \in V$
- u_i can be interpreted as the length of path from node s to node i
- LP dual constraints are difference constraints!
- LP is bounded below if and only if there is no negative length cycle
- LP optimality conditions correspond to the usual path optimality conditions
- Can deduce that the LP formulation has the integrality property, i.e. 0-1 solutions naturally

Time Windows

- A path in a network $G=(V,A)$ may represent a route for a delivery vehicle
- In this case, the time that the vehicle makes each delivery may be constrained
- T_{ij} = time needed for vehicle to drive from node i to node j (may includes unloading at i)
- $[a_i,b_i]$ = time interval must arrive at node i in
- t_i = time arrive at node i
- If (i,j) in path, then $t_j = \max\{t_i + T_{ij}, a_j\}$
- Require $t_i \le b_i$ for all $i \in V$

Integer LP Formulation

$$\min_{x \in \{0,1\}^{|A|},\, t} \quad \sum_{(i,j) \in A} c_{ij} x_{ij}$$

s.t.

$$\sum_{j\,:\,(j,i) \in A} x_{ji} \;-\; \sum_{j\,:\,(i,j) \in A} x_{ij} = \begin{cases} -1, & i = s \\ 0, & i \neq s,t \\ 1, & i = t \end{cases} \qquad \forall i \in V$$

$$t_j \geq t_i + T_{ij} - (b_i + T_{ij} - a_j)(1 - x_{ij}), \; \forall (i,j) \in A$$

$$a_i \leq t_i \leq b_i \;\; \forall i \in V$$

Additive Arc Resources

- A path in a network $G=(V,A)$ may represent a route for a delivery vehicle
- The petrol consumed by the vehicle may be constrained by its tank capacity, Q
- D_{ij} = petrol consumed on arc (i,j)
- d_i = total petrol consumed visiting all customers on the route up to and including i
- If (i,j) in path, then $d_j = d_i + D_{ij}$
- Require $d_i \leq Q$ for all $i \in V$ (or just $d_t \leq Q$)
- Special case of time windows $[0,Q]$

ILP Formulation

$$\min_{x \in \{0,1\}^{|A|}} \sum_{(i,j) \in A} c_{ij} x_{ij}$$

s.t.

$$\sum_{j\,:\,(j,i) \in A} x_{ji} - \sum_{j\,:\,(i,j) \in A} x_{ij} = \begin{cases} -1, & i=s \\ 0, & i \neq s,t \\ 1, & i=t \end{cases} \qquad \forall i \in V$$

$$\sum_{(i,j) \in A} D_{ij} x_{ij} \leq Q$$

Additive Node Resources

- A path in a network $G=(V,A)$ may represent a route for a delivery vehicle
- The quantity that can be carried on the vehicle may be constrained by its capacity, Q
- D_i = demand quantity for node i
- d_i = total demand for all customers preceding on the route, including i
- If (i,j) in path, then $d_j = d_i + D_j$
- Require $d_i \leq Q$ for all $i \in V$ (or just $d_t \leq Q$)
- Special case of additive arc resource

ILP Formulation

$$\min_{x \in \{0,1\}^{|A|}} \sum_{(i,j) \in A} c_{ij} x_{ij}$$

s.t.

$$\sum_{j:(j,i) \in A} x_{ji} - \sum_{j:(i,j) \in A} x_{ij} = \begin{cases} -1, & i=s \\ 0, & i \neq s,t \\ 1, & i=t \end{cases} \qquad \forall i \in V$$

$$\sum_{(i,j) \in A} D_i x_{ij} \leq Q - D_t$$

Vector Resources

- Arc (i,j) may use r^k_{ij} units of resource k
- Resource limits R^k for each resource k
- q^k_i = cumulative units of resource k used on path nodes from s to i
- If (i,j) in path, then $q^k_j = q^k_i + r^k_{ij}$
- Require $q^k_t \leq R^k$ for all k
- (Alternatively, some resources may be time window type, not just additive)

ILP Formulation

$$\min_{x \in \{0,1\}^{|A|}} \sum_{(i,j) \in A} c_{ij} x_{ij}$$

s.t.

$$\sum_{j\,:\,(j,i) \in A} x_{ji} - \sum_{j\,:\,(i,j) \in A} x_{ij} = \begin{cases} -1, & i=s \\ 0, & i \neq s,t \\ 1, & i=t \end{cases} \qquad \forall i \in V$$

$$\sum_{(i,j) \in A} r^k_{ij} x_{ij} \leq R^k, \ \forall \text{resources } k$$

Renewable Resources

- A path may represent a sequence of work for a crew in a “job network”, perhaps including several duty periods with rests in between
- The resource can be replenished at some special nodes, $R \subseteq V$
- If (i,j) in path and $j \notin R$, then $d_j = d_i + D_j$
- So if $j \in R$ then free to re-set $d_j = 0$
- Require $d_i \leq Q$ for all $i \in V$

Elementary Paths

- Shortest path problems often arise as sub-problems in column generation
- In these cases there may be negative length cycles – a shortest elementary path (no repeated nodes) is sought
- This can be modelled by using a vector of additive resources, one for each node i
- Resource $r^k_i=1$ if $k=i$, 0 otherwise $\forall i,k \in V$
- Resource limit $R^k=1$ for each $k \in V$

Other Variations

- Multi-criteria optimization
- Dynamic shortest paths e.g. with time-dependent travel times, such as due to peak-hour congestion
- Subset disjoint paths
- Many others……