Workshop on Artificial Intelligence in Practice

Part 1: AI Targets and Applications in Technics and Logistics

Sebastian Iwanowski
FH Wedel (D)
Erasmus Workshop at Fontys University, Eindhoven (NL)

Section 2a: Algorithms of Dijkstra and A*

Uninformed Search Strategy

Dijkstra‘s Algorithm for Edge-Valued Graphs

For all edges (u,v) there is a value:
length (u,v) := edge evaluation from node u to node v

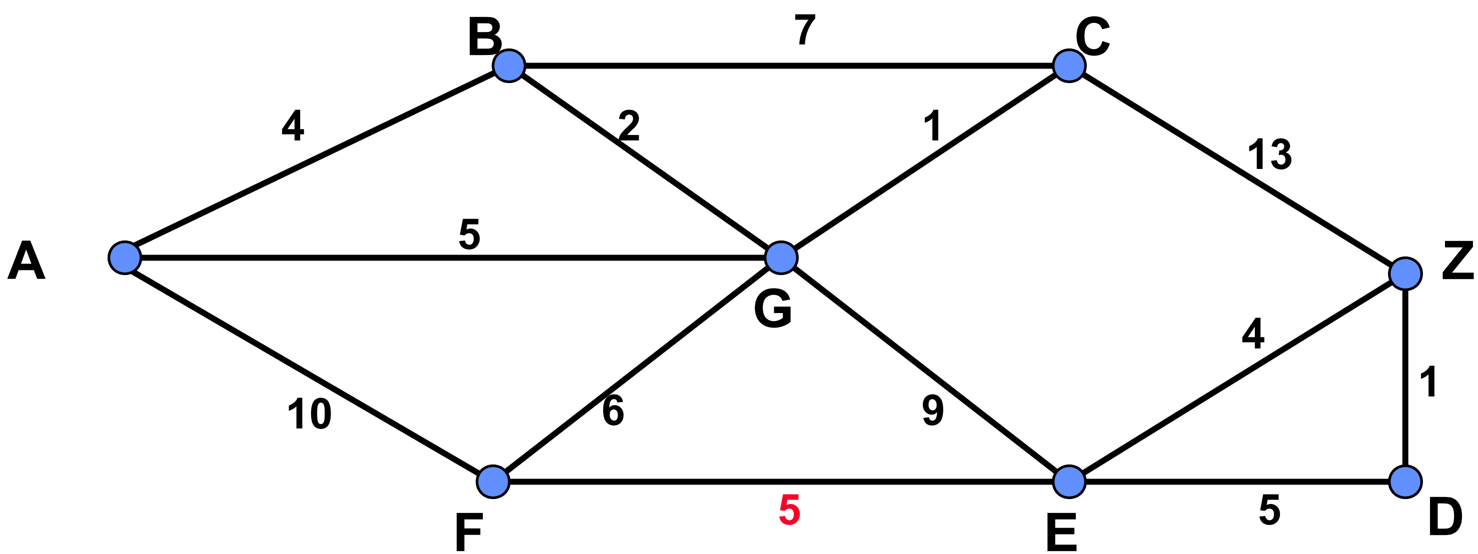
Prerequesite for edge values: All edge values must be nonnegative.

Algorithm for search of shortest path from A to B (path with minimal sum of edge values):

- Let *Done* be a set initialized by {A}. Set *label* (A) := 0.
  Let *NotFinallyComputed* be a set initially consisting of all other nodes.
  Set *label* (n) := *length* (A,n) for all nodes n directly adjacent to A (the „neighbors“).
  Set *label* (v) := ∞ for all nodes n not directly adjacent to A.

- Repeat:
  Choose node u from *NotFinallyComputed* such that *label* (u) is minimal
  and move u from *NotFinallyComputed* to *Done*.
  Update all nodes n from *NotFinallyComputed* that are directly adjacent to u:
  label (n) := min {*label* (n), *label* (u) + *length* (u,n)}.
  until u = B

Example for Dijkstra‘s algorithm



Shortest path from G to Z: G → E → Z (13 units)

node (path length from G, direct predeccesor responsible for latest update):

A(5,G)		A(5,G)		A(5,G)						
B(2,G)		B(2,G)								
C(1,G)										
D(∞)	→	D(∞)	→	D(∞)	→	D(∞)	→	D(∞)	→	D(14,E)
E(9,G)		E(9,G)		E(9,G)		E(9,G)		E(9,G)		
F(6,G)		F(6,G)		F(6,G)		F(6,G)				
Z(∞)		Z(14,C)		Z(14,C)		Z(14,C)		Z(14,C)		Z(13,E)

Informed (heuristic) Search Strategies

requires the following additional information for each node:

estimation value h (node) as a lower bound for the distance left to the destination

- must be easy to compute (e.g. by geographic coordinates)
- must guarantee that each actual path to the destination is not shorter

h() gives a nonnegative value: The smaller the value, the closer is the distance to the destination

Implementation of this principle: A* Algorithm

- applies Dijkstra‘s algorithm to this principle
- chooses for shift from *NotFinallyComputed* to *Done* not the node with minimal label, but rather the node with minimal sum of label plus estimation value

red = difference to Dijkstra

Informed (heuristic) Search Strategy

A* Algorithm for Edge-Valued Graphs with node heuristic

For all edges (u,v) there is a value:
length (u,v) := edge evaluation from node u to node v

For all nodes n there is a value:
$h_B(n)$:= lower bound for distance to B

Prerequesite for edge values: All edge values must be nonnegative.

Prerequesite for node heuristic $h_B(u)$ and actual path length $p_B(u)$ to B:

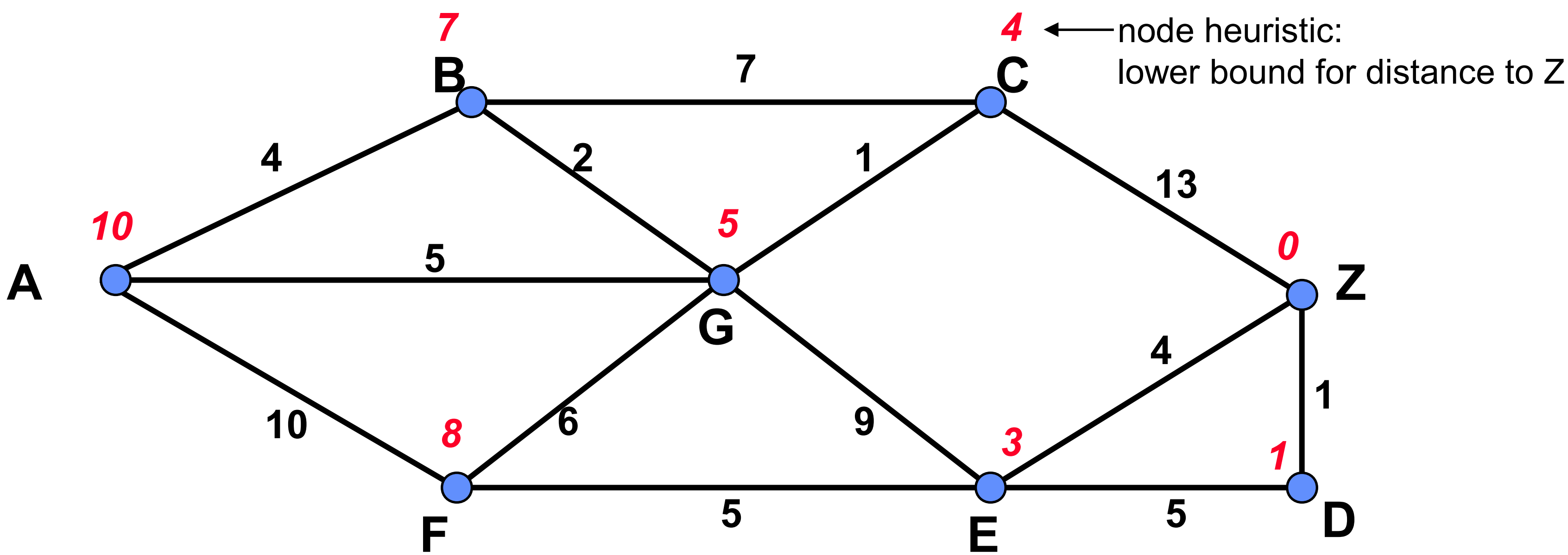
admissibility:	$h_B(u) \le p_B(u)$	(no overestimation)
monotonicity:	$h_B(u) \le h_B(v) + length(u,v)$	(triangle inequality)

Algorithm for search of shortest path from A to B (path with minimal sum of edge values):

- Let *Done* be a set initialized by {A}. Set *label* (A) := 0.
  Let *NotFinallyComputed* be a set initially consisting of all other nodes.
  Set *label* (n) := *length* (A,n) for all nodes n directly adjacent to A (the „neighbors")
  and *estimatedPathLength* (n) := *label* (n) + $h_B(n)$.
  Set *label* (v) := ∞ for all nodes n not directly adjacent to A and *estimatedPathLength* (v) := ∞
- Repeat:
  Choose node u from *NotFinallyComputed* such that *estimatedPathLength* (u) is minimal
  and move u from *NotFinallyComputed* to *Done*.
  Update all nodes n from *NotFinallyComputed* that are directly adjacent to u:
  label (n) := min {*label* (n), *label* (u) + *length* (u,n)}
  estimatedPathLength (n) := *label* (n) + $h_B(n)$ (if label has changed).

  until u = B

Example for A* Algorithm



Shortest path from G to Z : G → E → Z (13 units)

node (path length from G, direct predeccesor responsible for latest update, estimatedPathLength):

A(5,G,15)		A(5,G,15)		A(5,G,15)		A(5,G,15)
B(2,G,9)		B(2,G,9)				
C(1,G,5)						
D(∞)	→	D(∞)	→	D(∞)	→	D(14,E,15)
E(9,G,12)		E(9,G,12)		E(9,G,12)		
F(6,G,13)		F(6,G,14)		F(6,G,14)		F(6,G,14)
Z(∞)		Z(14,C,14)		Z(14,C,14)		Z(13,E,13)

2 steps saved from Dijkstra