

Conflict-free Real-time AGV Routing

Beschreibung eines Routing-Algorithmus mit Zeitfenstern

Autonomes Fahren

AGV (Indoor)

- bekannte, kurze Strecken
- keine Hindernisse
- geringe Geschwindigkeiten
- bauliche Maßnahmen möglich
- gesetzlich erlaubt

PKW (Outdoor)

- hohe Geschwindigkeiten
- Vielzahl von Hindernissen
- unterschiedlicher Straßenbelag
- unterschiedliche Witterungsverhältnisse
- gesetzlich verboten

- Busbahn
- elektrische Deichsel

Spurführung

- Leitdraht
- Transponder
- Magnet
- Scanner
- Bildverarbeitung
- landgestützte Satellitennavigation (DGPS)

Kommunikation

- Induktive Datenübertragung (100 kHz)
- Infrarot
- Datenfunk (2,54 GHz)
 - geringe Störempfindlichkeit
 - uneingeschränkte Reichweite
 - hohe Übertragungsgeschwindigkeit

Hinderniserkennung

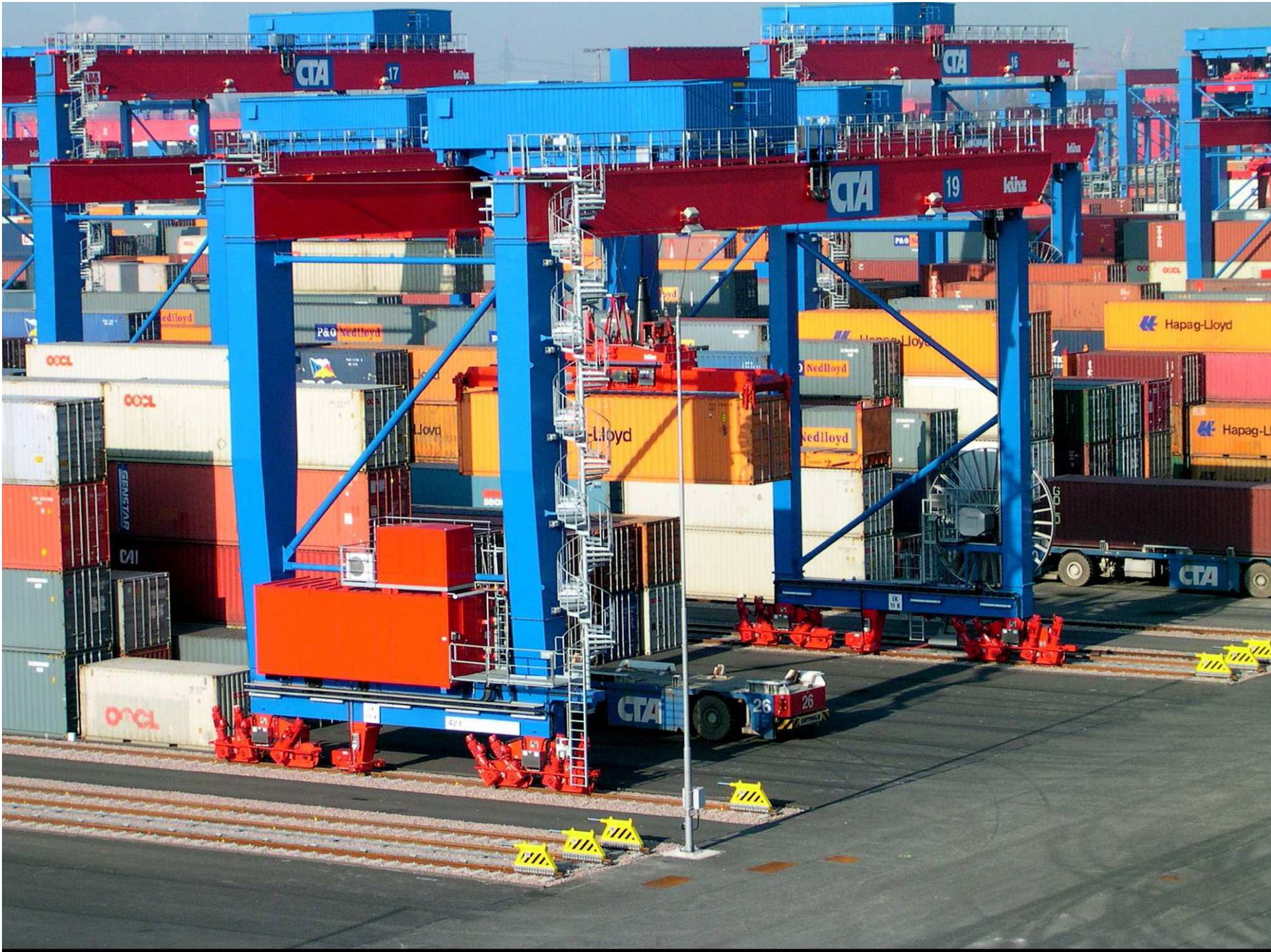
- Radar
- Bildverarbeitung
- Laser

Kontrolle

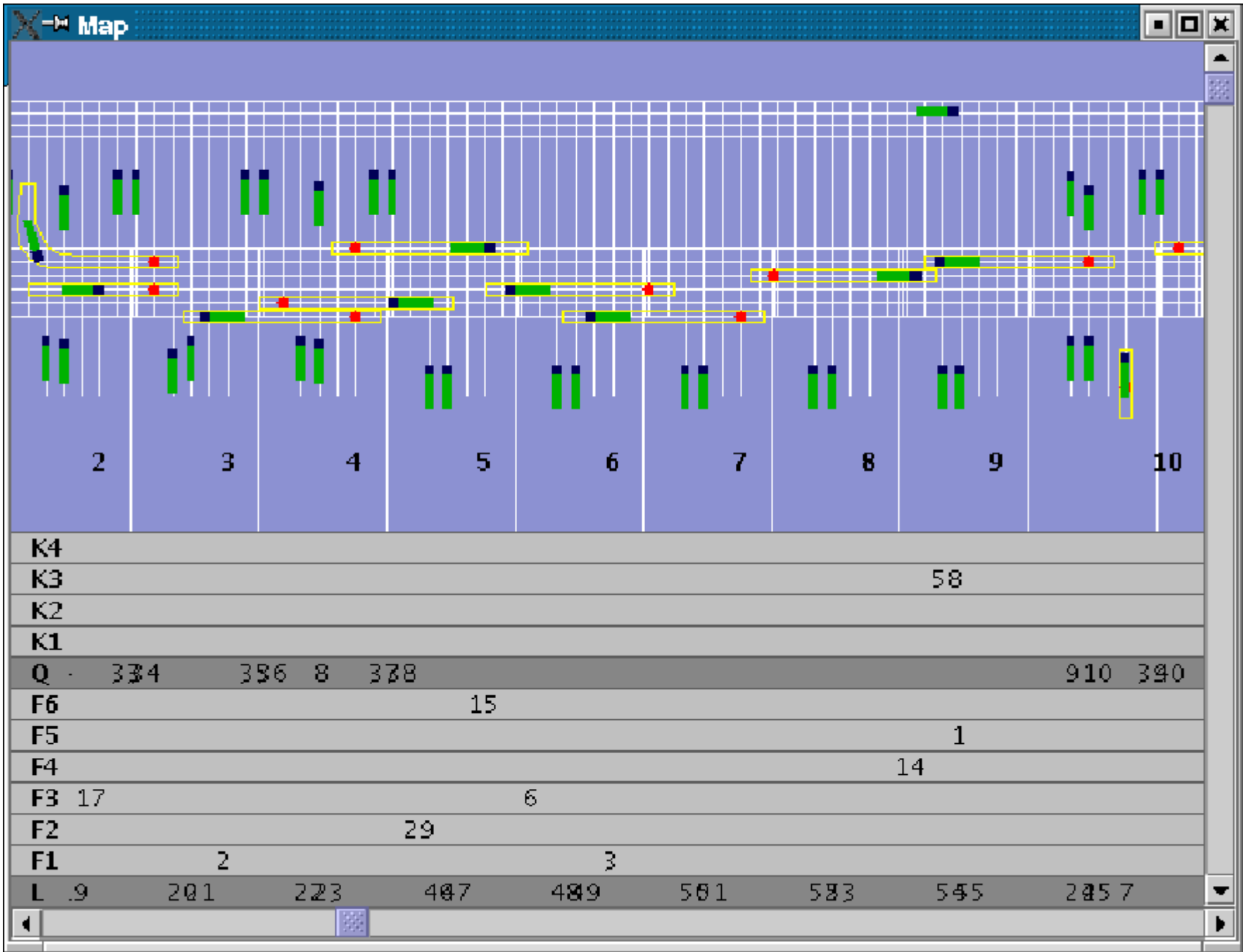
- Routing
- Kollisionsverhütung











Das Modell

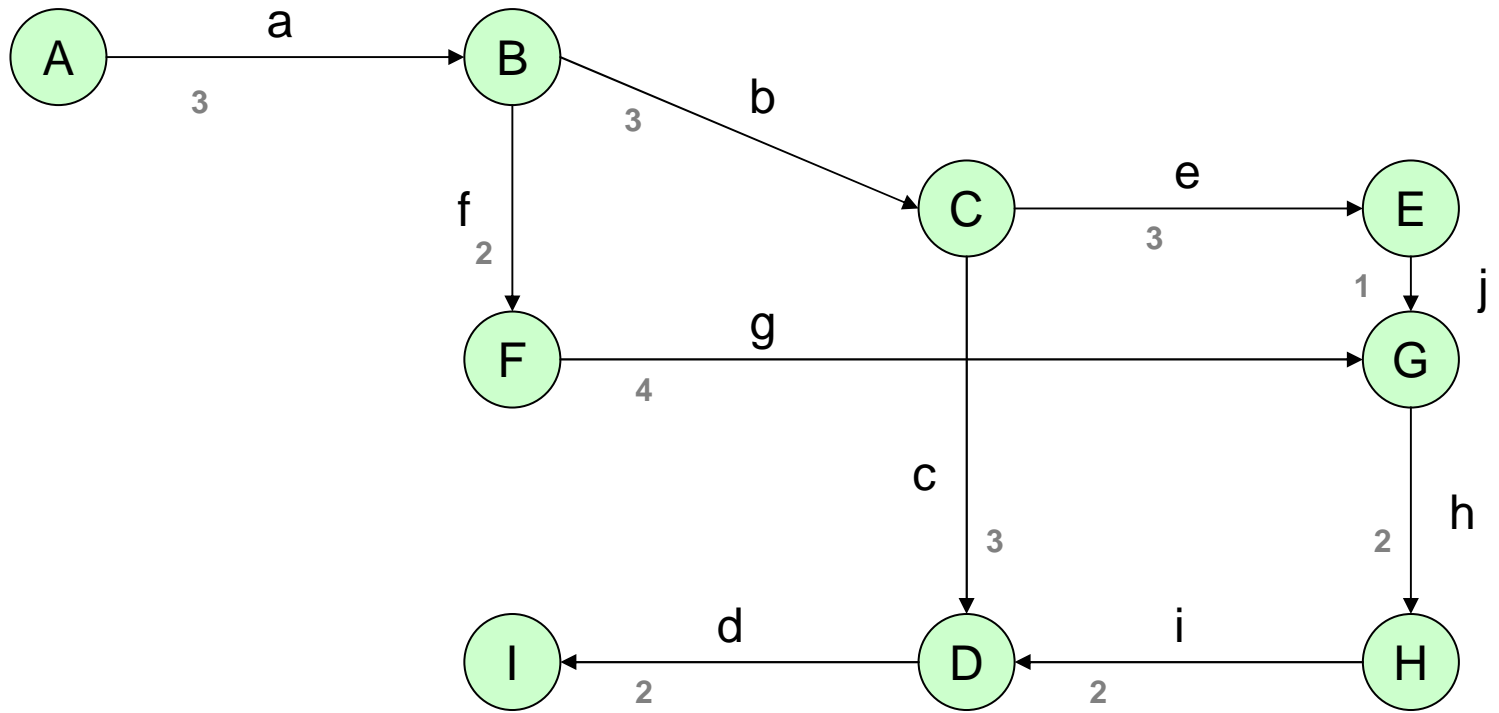
$G = (V, A)$ gerichteter Graph, der die Wege des Transportsystems repräsentiert

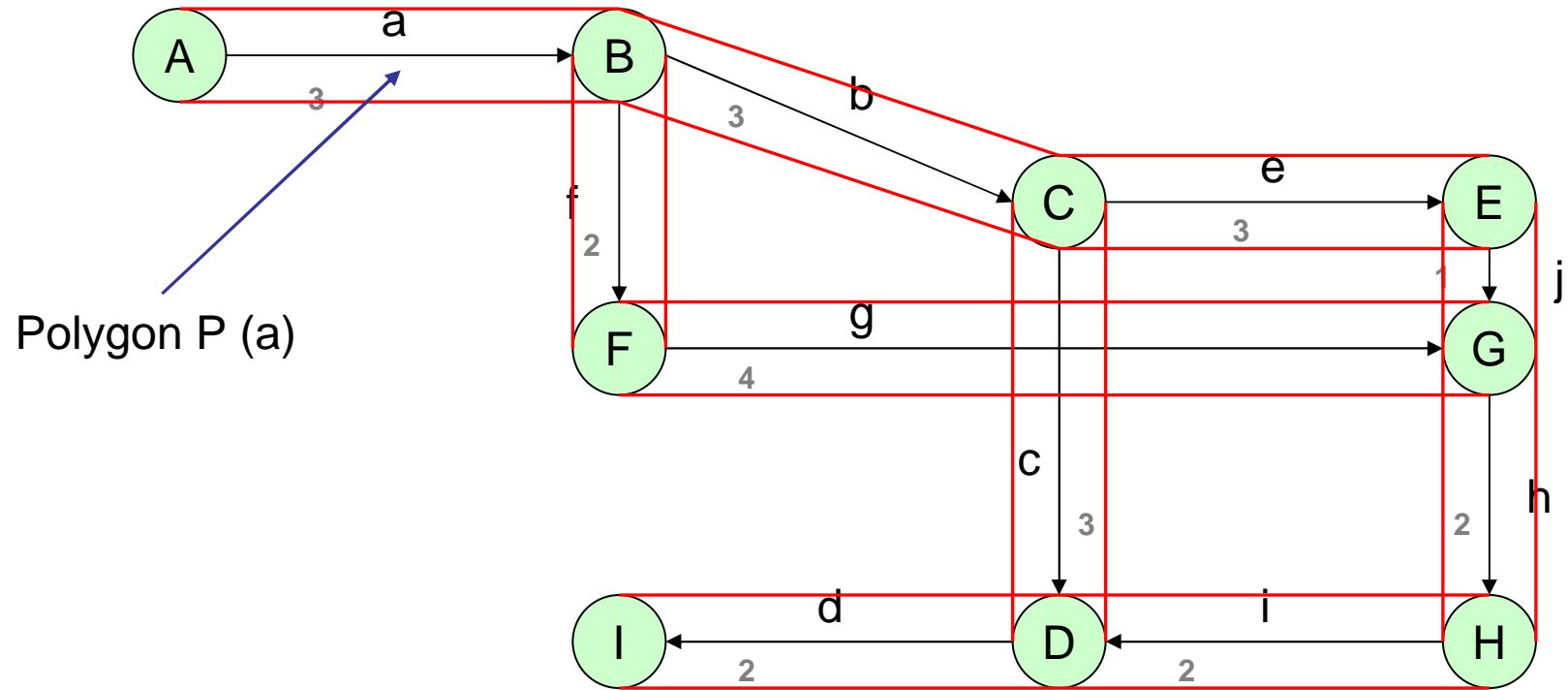
$T(\alpha)$ Fahrtzeit für Kante α

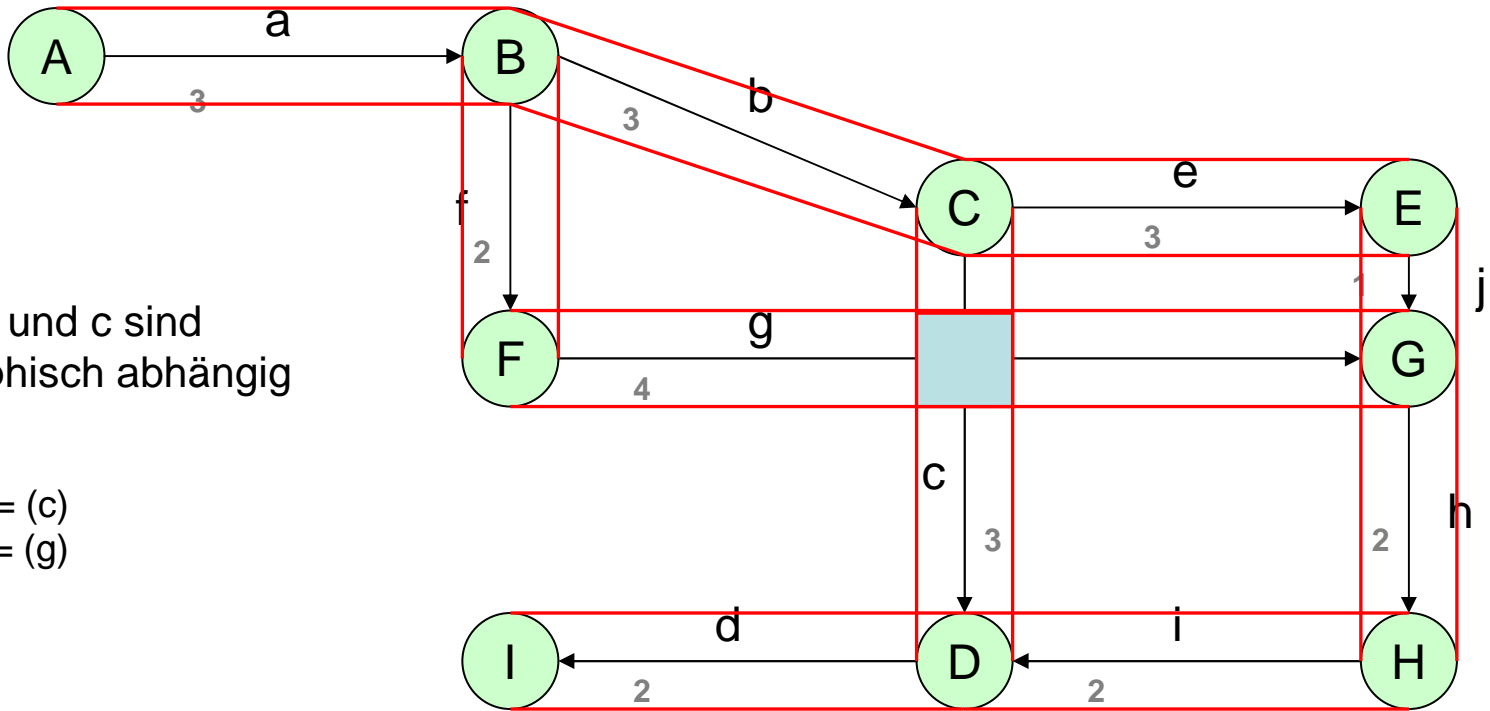
$\sigma = r_1, \dots, r_n$ Sequenz von Anfragen

Gesucht ist eine Route von Startpunkt s_j zum Zielknoten t_j zum Zeitpunkt θ_j

$r_j = (s_j, t_j, \theta_j)$ Online-Anfrage eines AGV

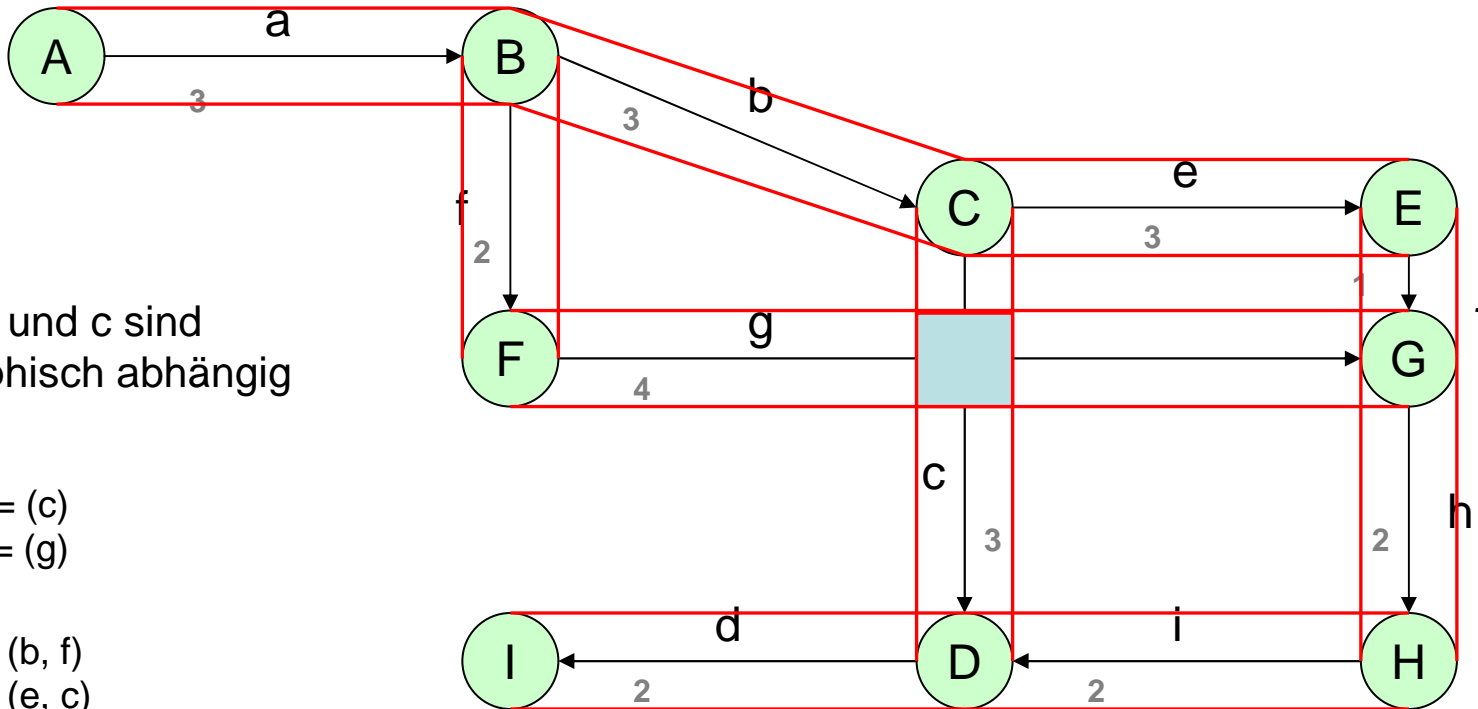






Kante g und c sind geographisch abhängig

confl (g) = (c)
confl (c) = (g)



Kante g und c sind
geographisch abhängig

confl (g) = (c)
confl (c) = (g)

Out (a) = (b, f)
Out (b) = (e, c)
Out (c) = (d)
Out (d) = nil
Out (e) = (j)
Out (f) = (g)
Out (g) = (h)
Out (h) = (i)
Out (i) = (d)
Out (j) = (h)

Das Problem

„Kürzeste Wege Problem mit Zeitfenstern“

c_α Kosten für Kante α
 F_α Menge der Zeitfenster für Kante α

Der Algorithmus

$(\alpha_L, d_L, I_L, \text{pred}_L)$ Label L

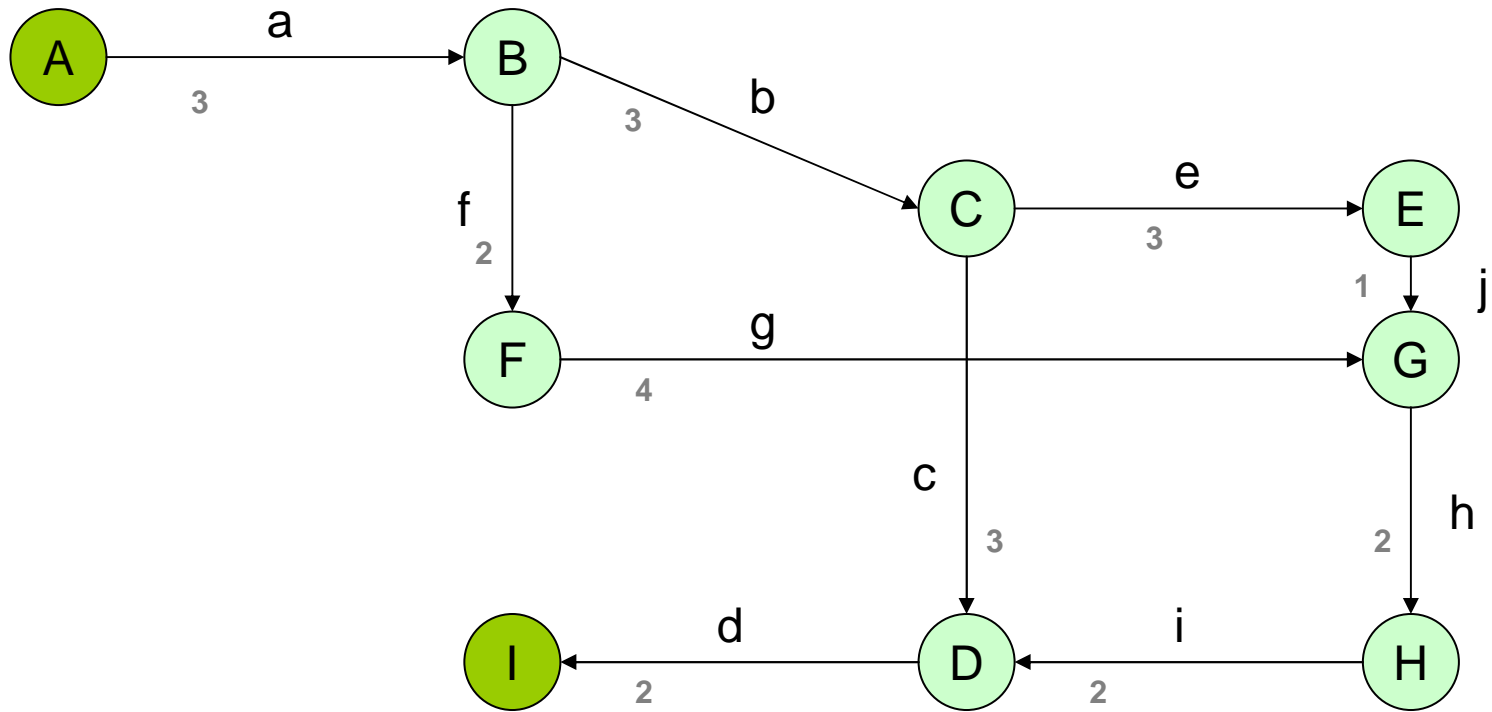
Jedes Label L repräsentiert einen Pfad vom Startknoten s zum Ende von α_L

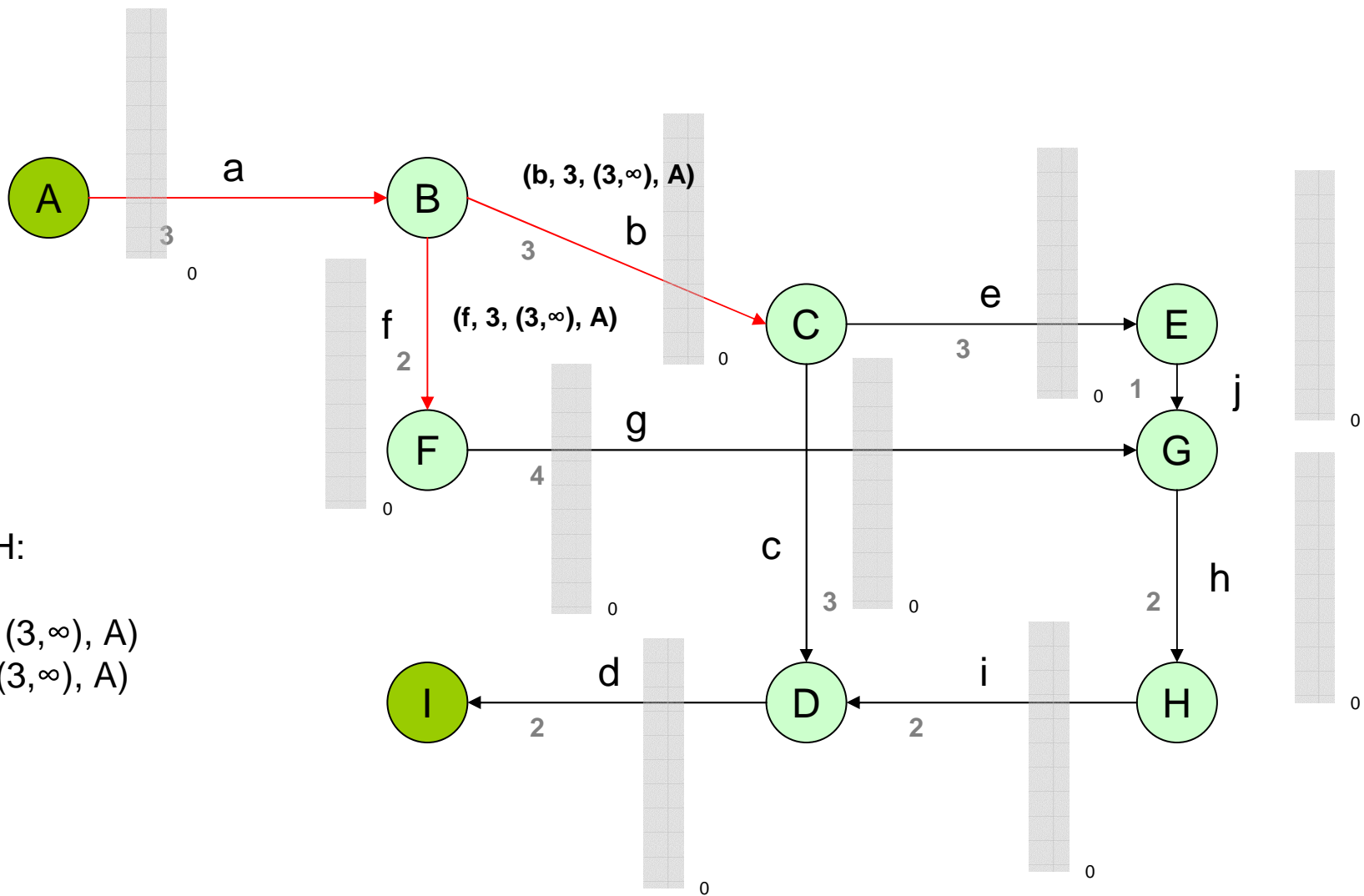
α_L Label für Kante α
 d_L Distanzwert (Reise- und Wartezeiten)
 I_L Labelintervall $I_L = (A_L, B_L)$
 pred_L Vorgänger von α_L

Definition: Ein Label L dominiert Label L', genau dann wenn...
 $d_L \leq d_{L'}$ und $I_{L'} \subseteq I_L$

- * Initialisierung.
- * Schleife
 - Beginn
 - * Label mit kleinstem d auswählen.
 - * Wenn keine Label vorhanden,
dann Stop.
 - * Wenn Ziel erreicht,
dann Stop.
 - * Für jedes Zeitfenster
 - * Label expandieren.
 - * Dominanztest.
 - Ende.
- * Zeitfenster aktualisieren.

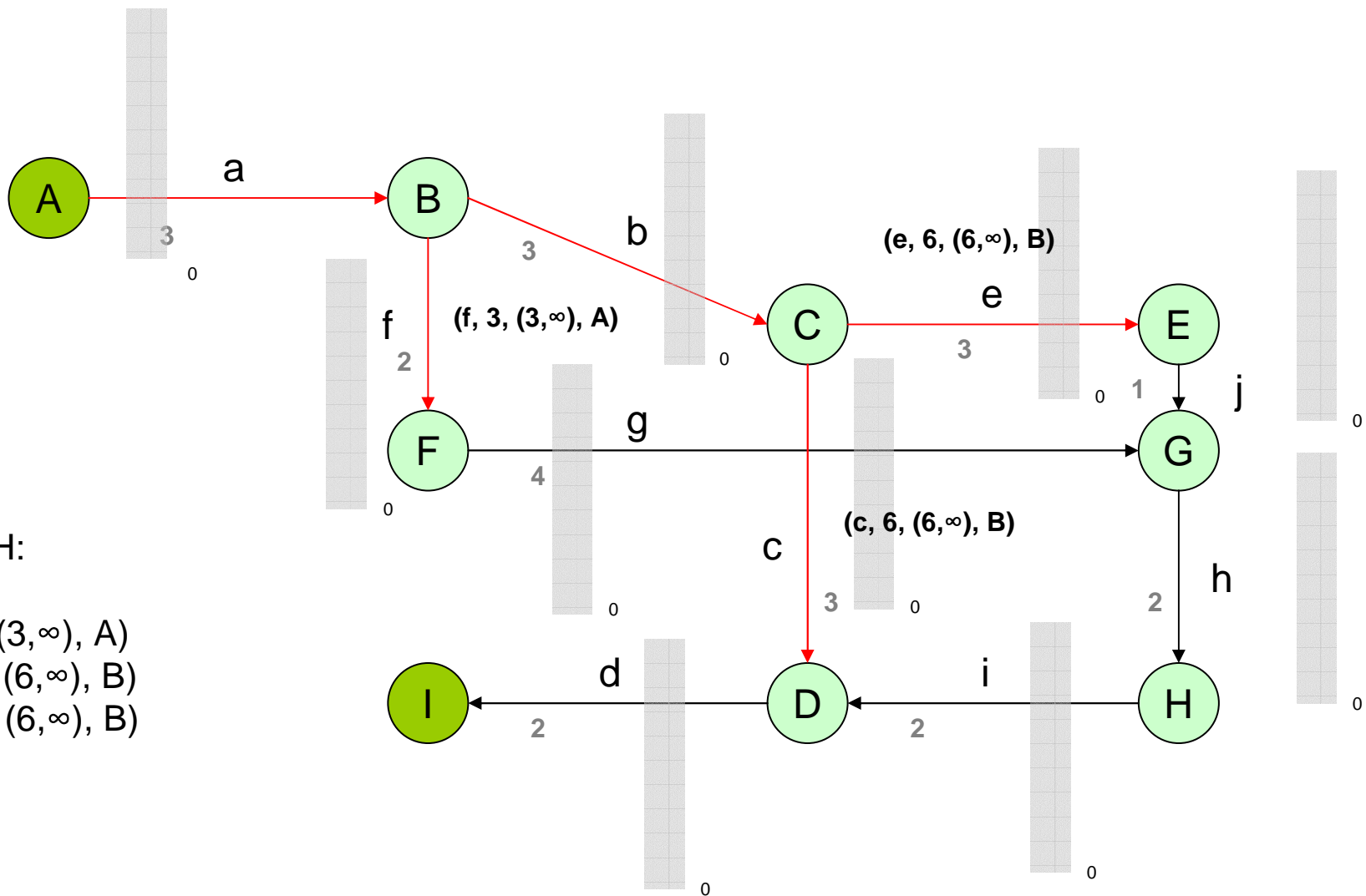
1. Anfrage: $r = (A, I, 0)$





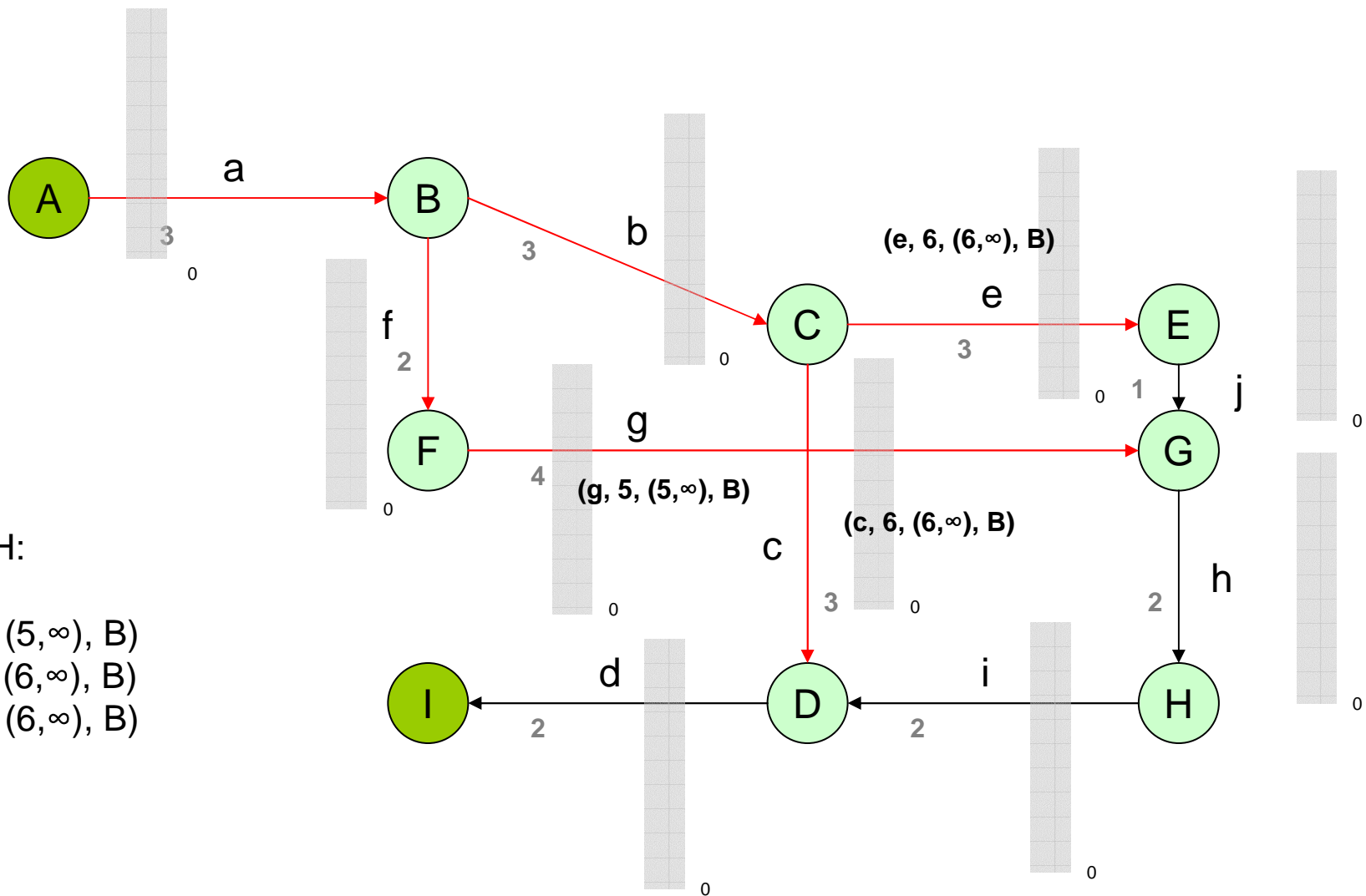
Queue H:

1. $(b, 3, (3, \infty), A)$
2. $(f, 3, (3, \infty), A)$
- 3.
- 4.
- 5.



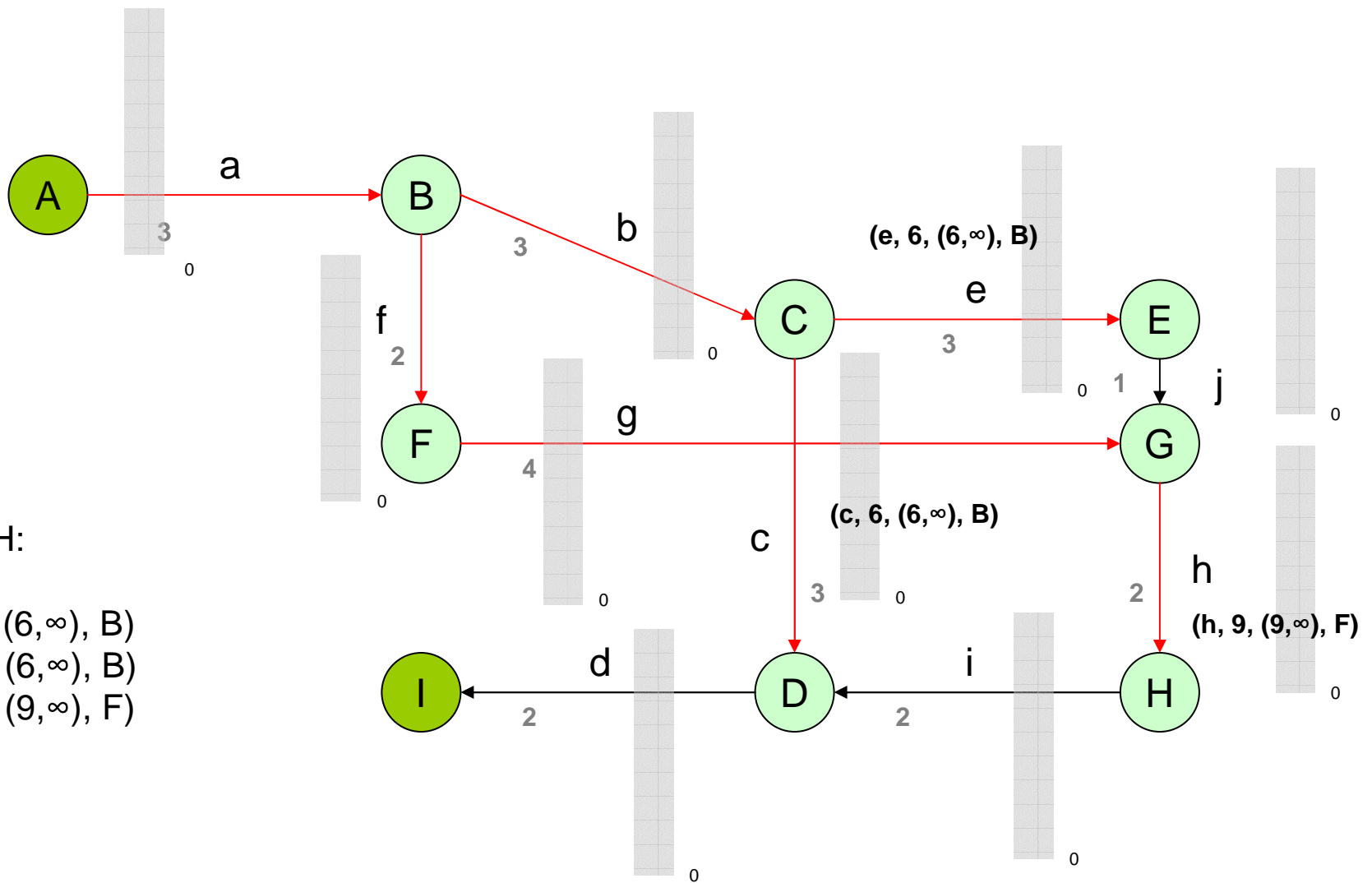
Queue H:

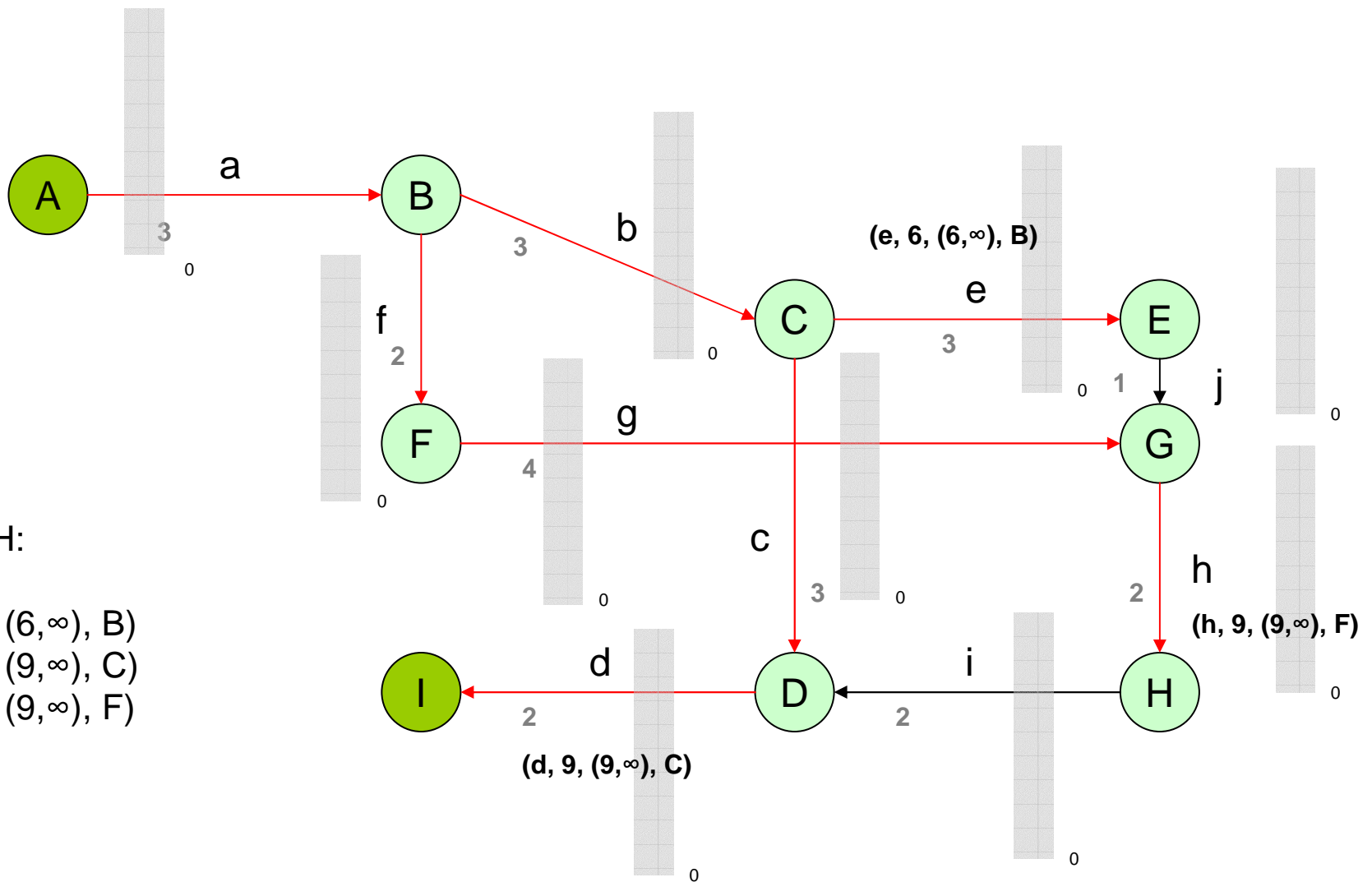
1. $(f, 3, (3, \infty), A)$
2. $(c, 6, (6, \infty), B)$
3. $(e, 6, (6, \infty), B)$
- 4.
- 5.



Queue H:

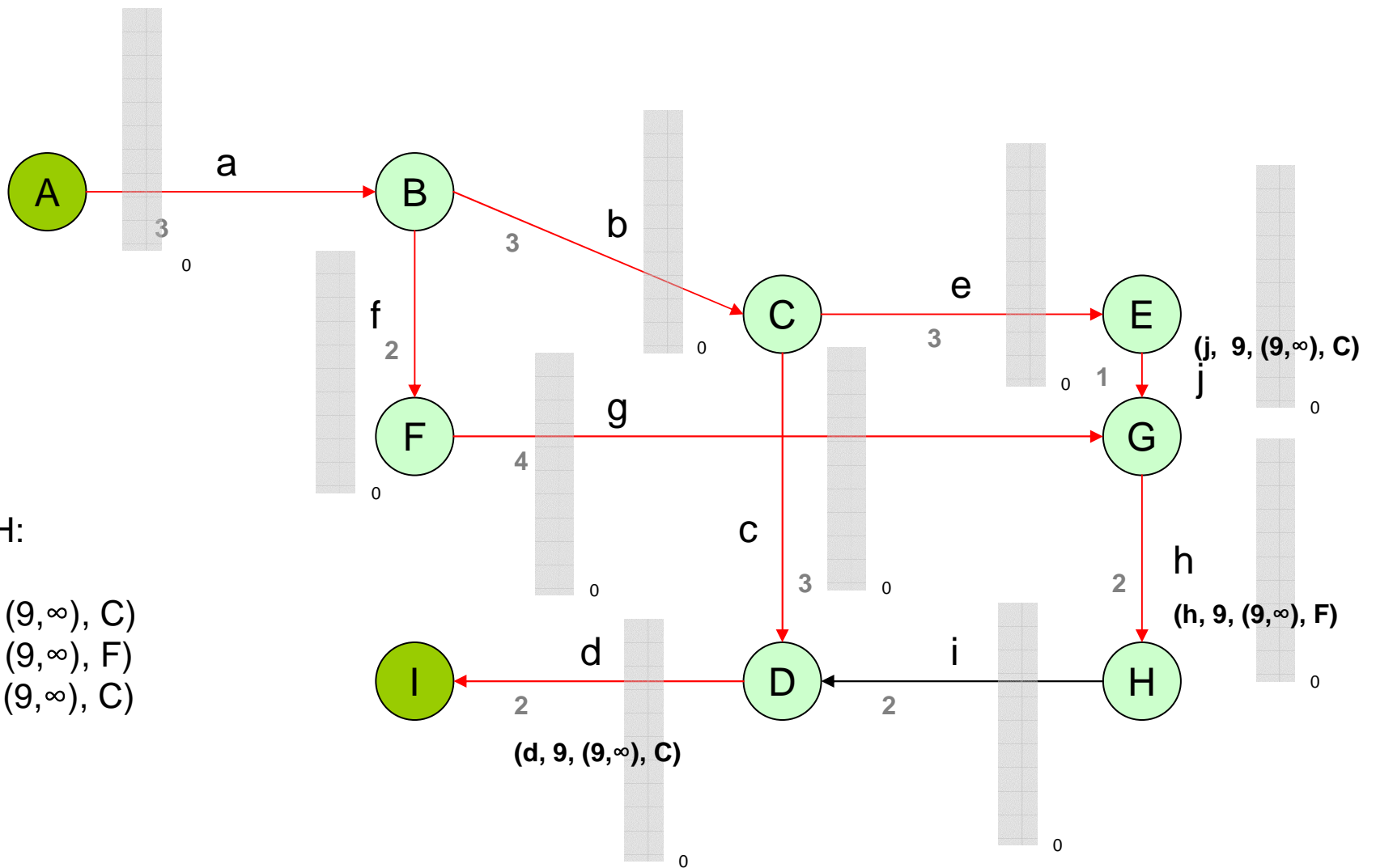
1. $(g, 5, (5, \infty), B)$
2. $(c, 6, (6, \infty), B)$
3. $(e, 6, (6, \infty), B)$
- 4.
- 5.

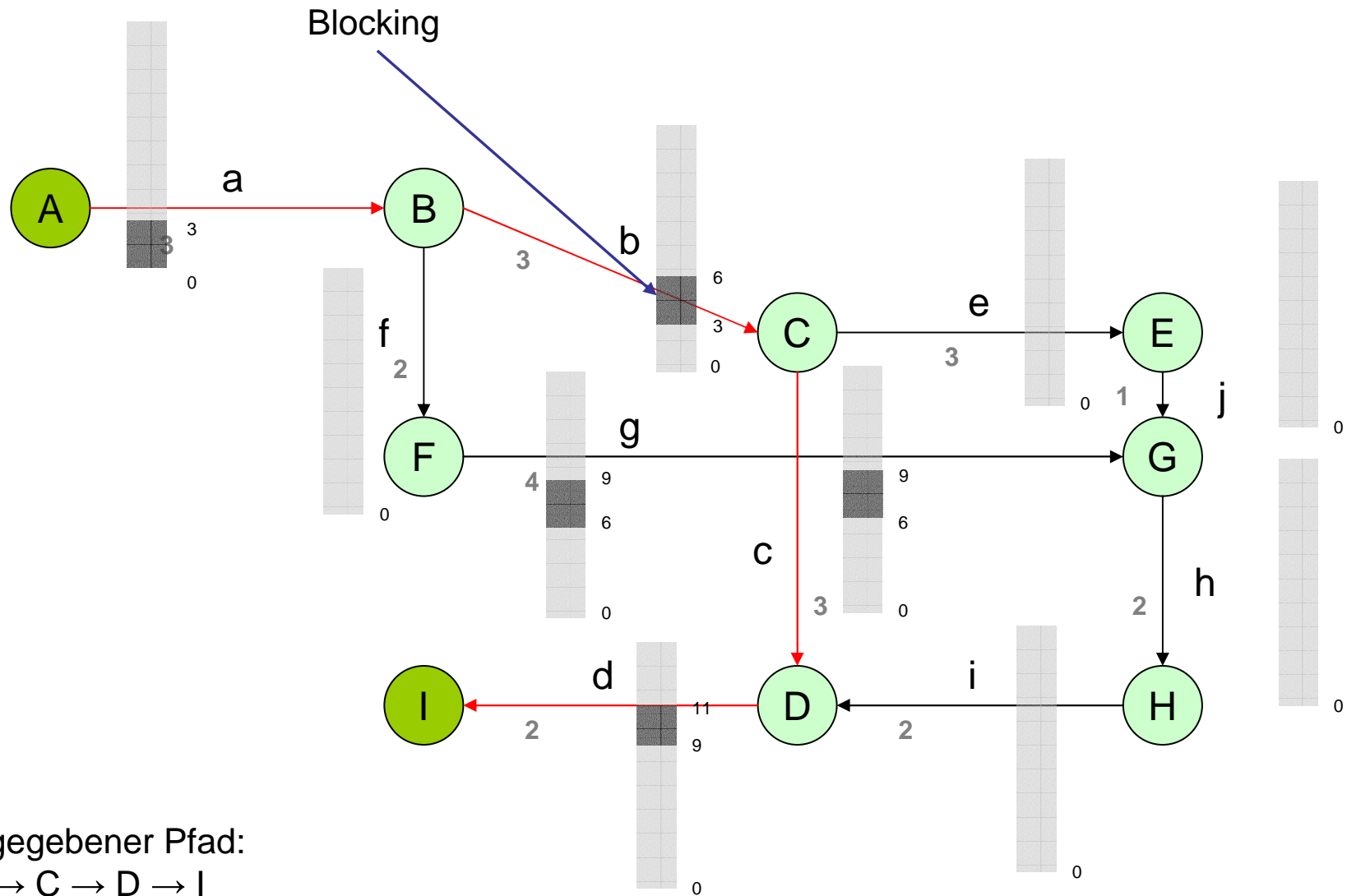




Queue H:

1. (e, 6, (6, ∞), B)
2. (d, 9, (9, ∞), C)
3. (h, 9, (9, ∞), F)
- 4.
- 5.

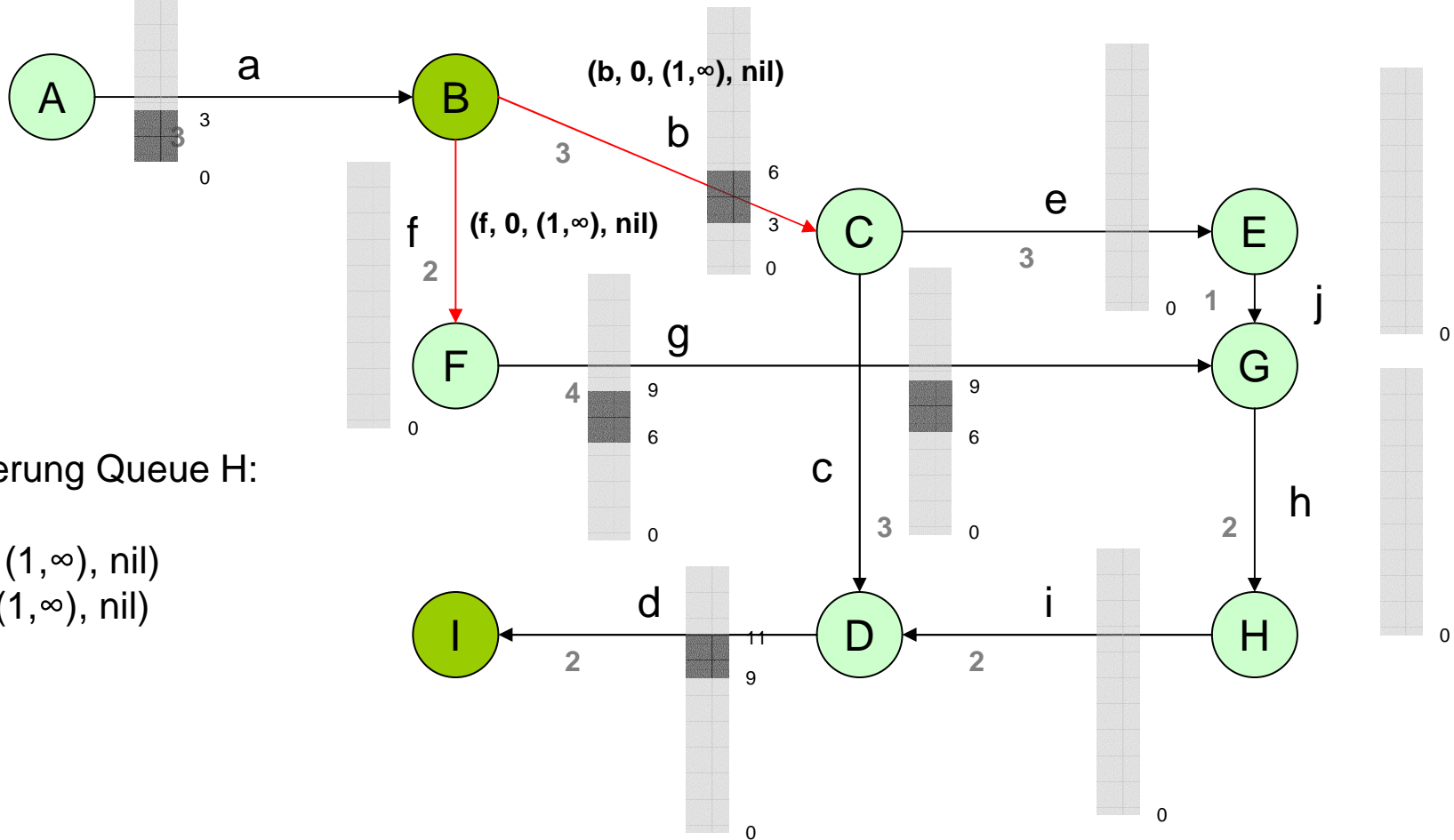




Zurückgegebener Pfad:
 $A \rightarrow B \rightarrow C \rightarrow D \rightarrow I$

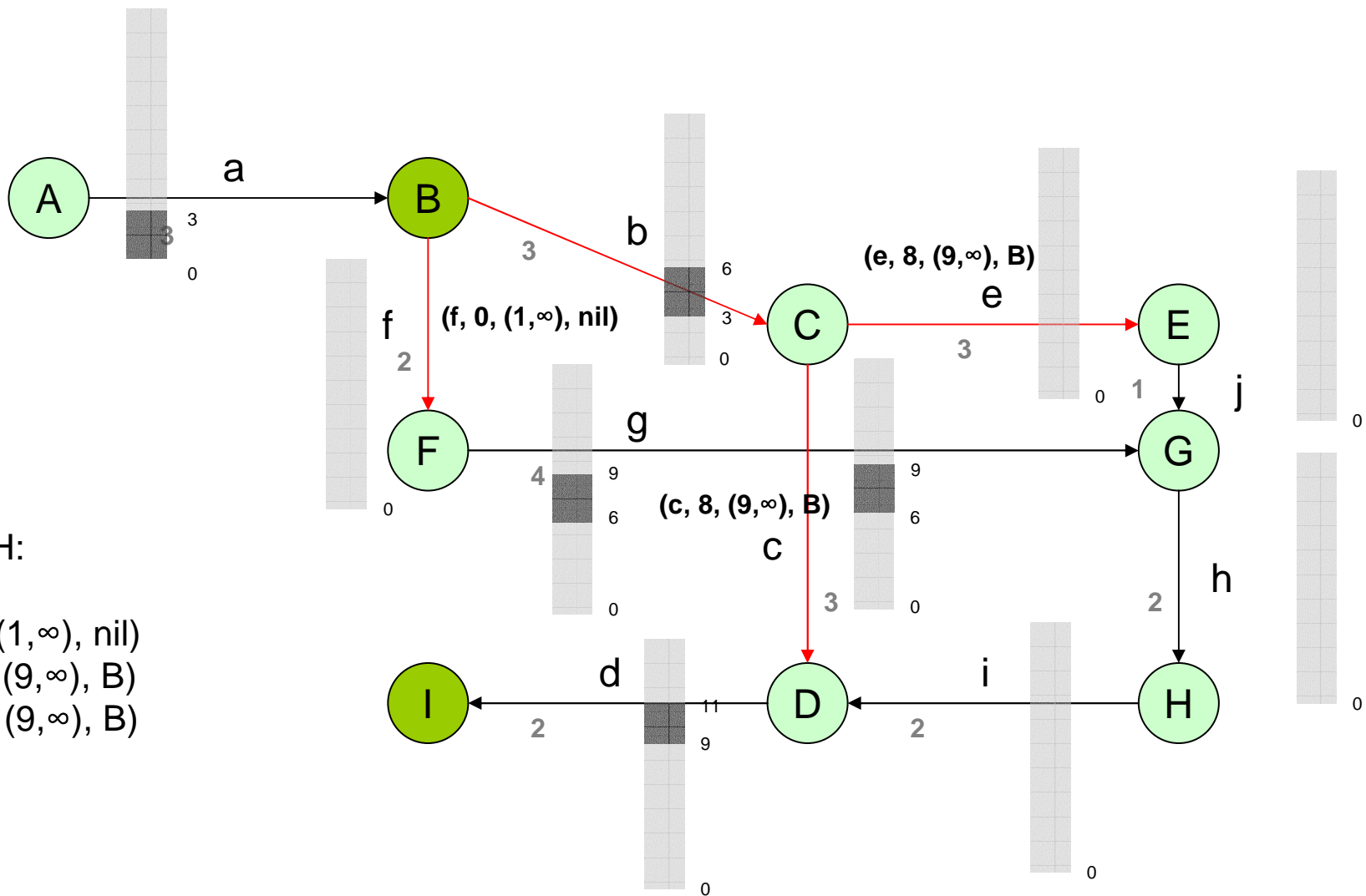
Fahrtzeit: 11 Zeiteinheiten
 davon Wartezeit: 0 Zeiteinheiten

2. Anfrage: $r = (B, l, 1)$



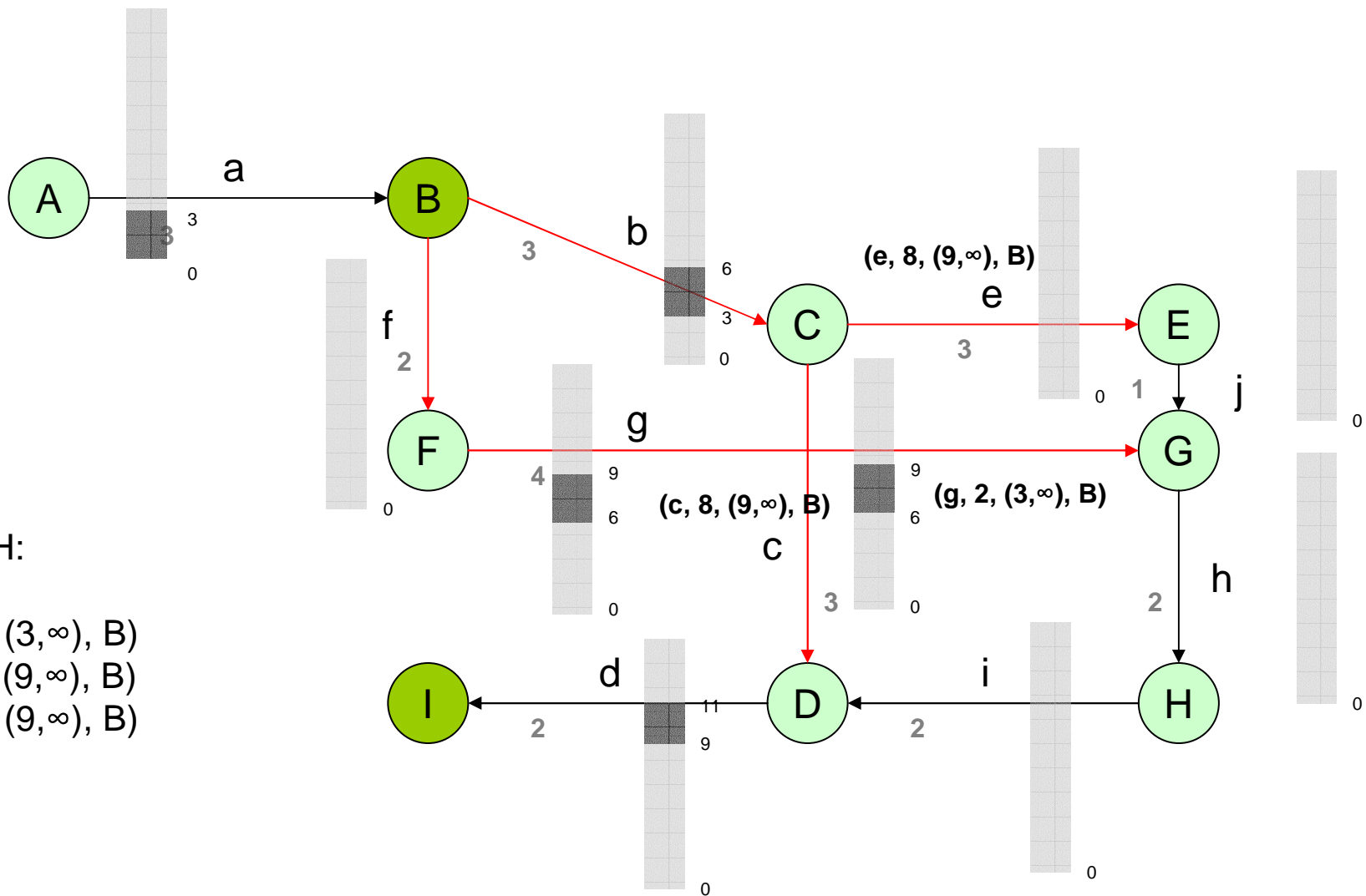
Initialisierung Queue H:

1. $(b, 0, (1, \infty), \text{nil})$
2. $(f, 0, (1, \infty), \text{nil})$
- 3.
- 4.
- 5.



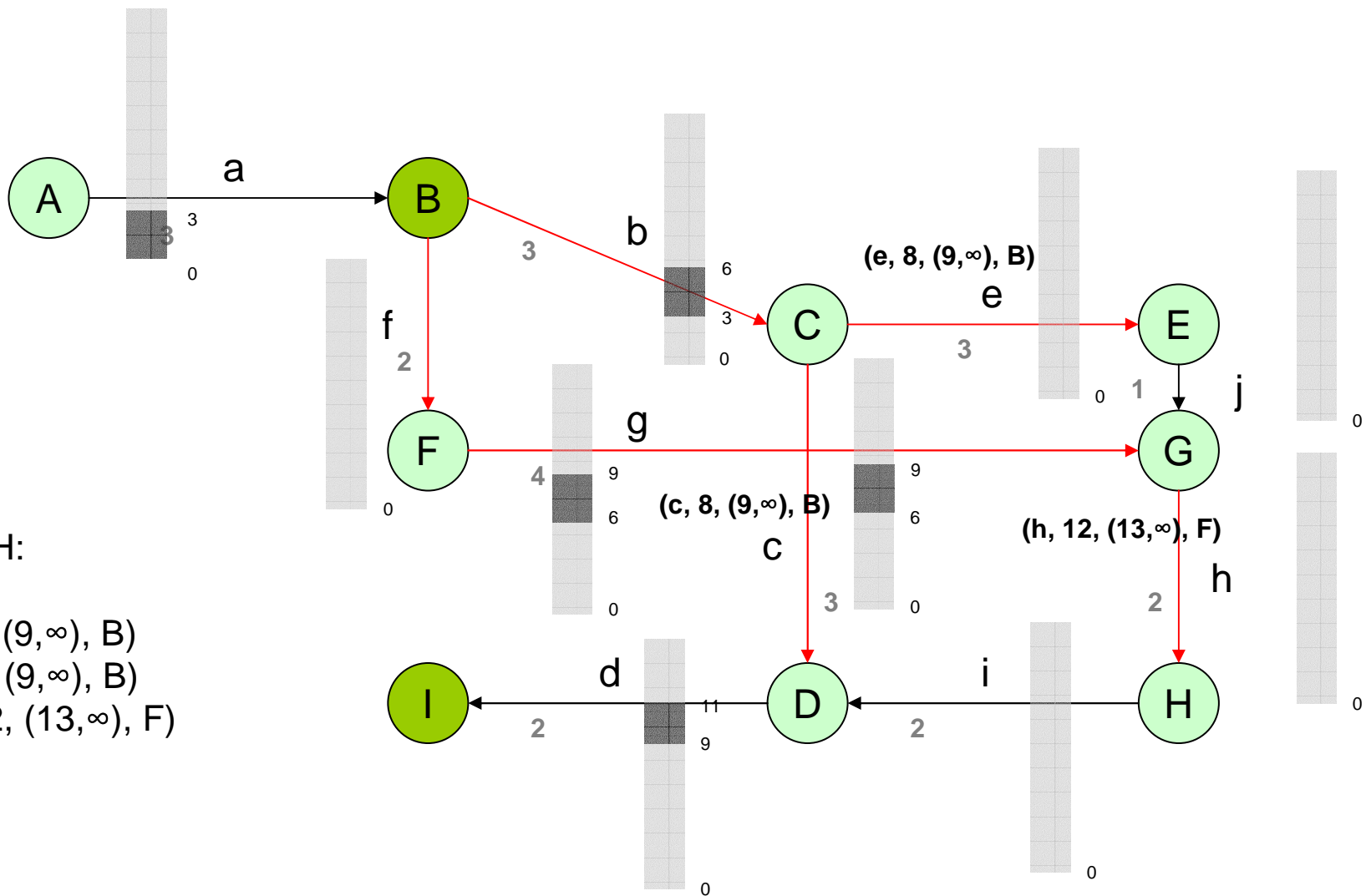
Queue H:

1. $(f, 0, (1, \infty), \text{nil})$
2. $(c, 8, (9, \infty), B)$
3. $(e, 8, (9, \infty), B)$
- 4.
- 5.



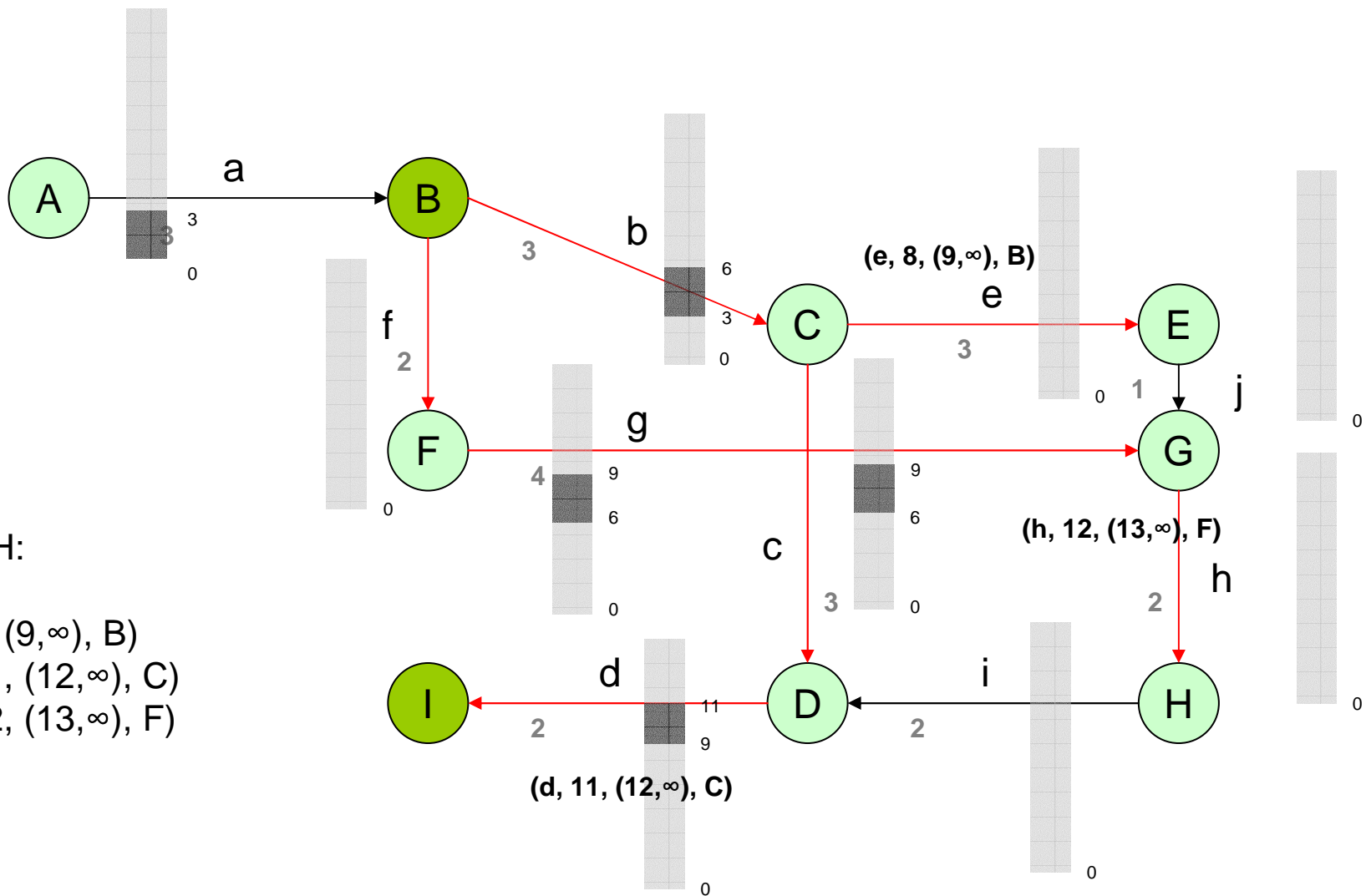
Queue H:

1. (g, 2, (3, ∞), B)
2. (c, 8, (9, ∞), B)
3. (e, 8, (9, ∞), B)
- 4.
- 5.



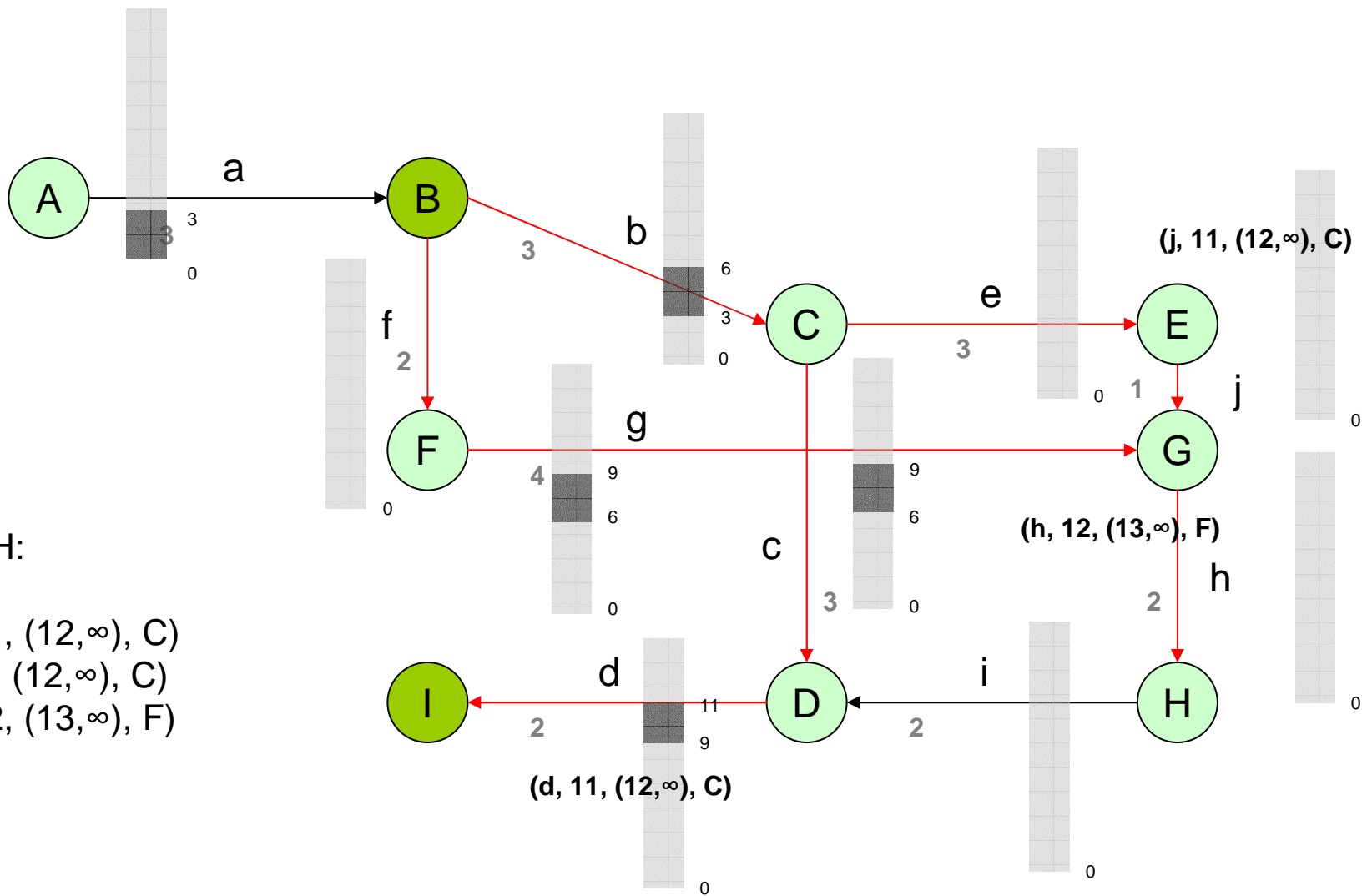
Queue H:

1. (c, 8, (9,∞), B)
2. (e, 8, (9,∞), B)
3. (h, 12, (13,∞), F)
- 4.
- 5.



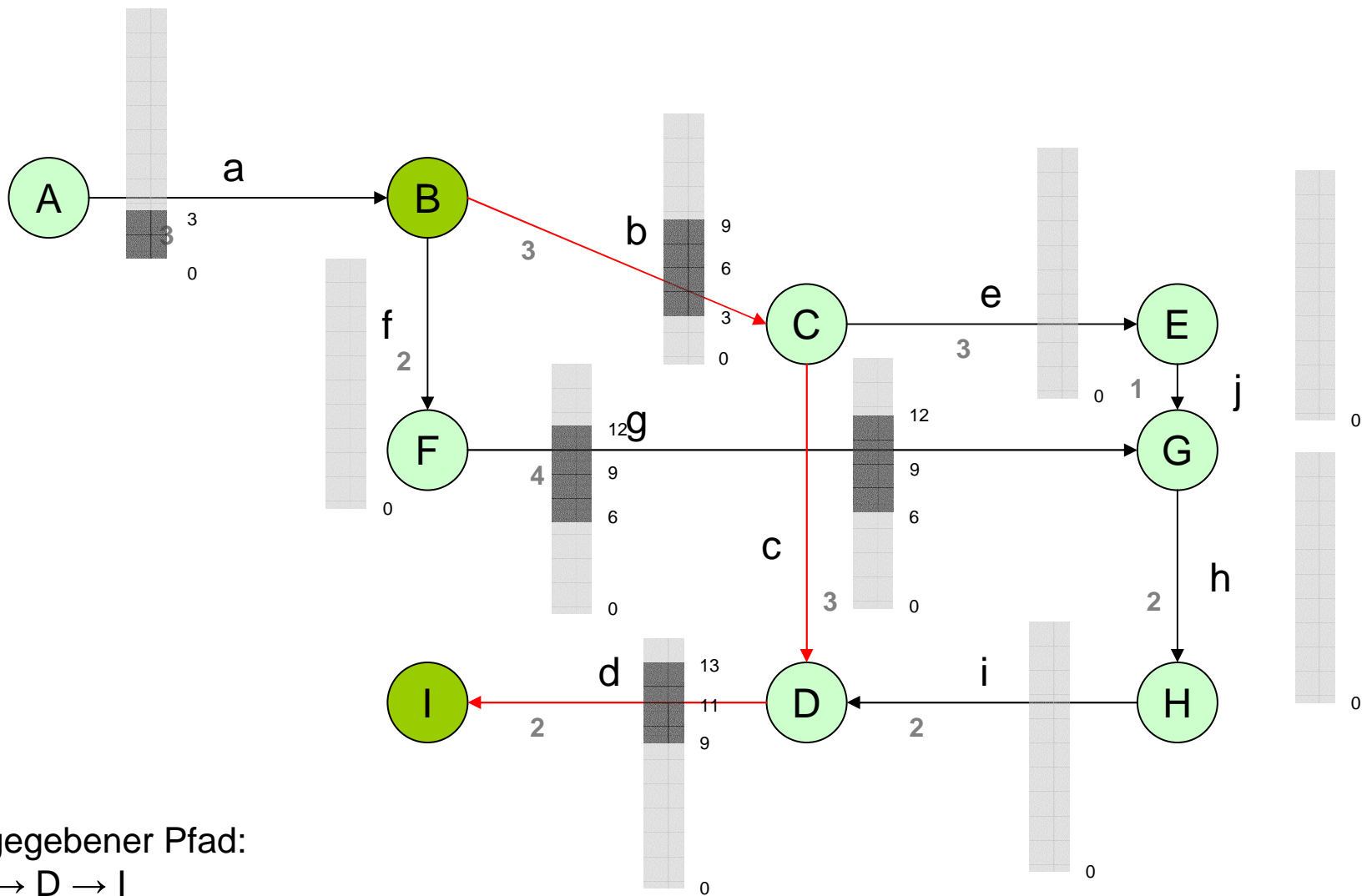
Queue H:

1. $(e, 8, (9, \infty), B)$
2. $(d, 11, (12, \infty), C)$
3. $(h, 12, (13, \infty), F)$
- 4.
- 5.



Queue H:

1. $(d, 11, (12, \infty), C)$
2. $(j, 11, (12, \infty), C)$
3. $(h, 12, (13, \infty), F)$
- 4.
- 5.



Zurückgegebener Pfad:
 B → C → D → I

Fahrtzeit: 13 Zeiteinheiten
 davon Wartezeit: 5 Zeiteinheiten