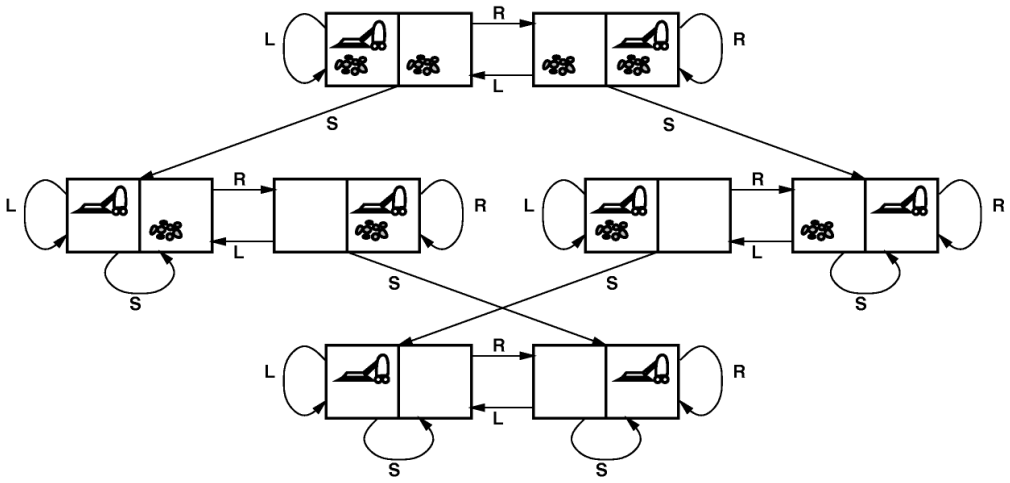
UMETNA INTELIGENCA

laboratorijske vaje: preiskovanje

Prostor stanj

Prostor stanj je:

- formalizem za predstavljanje problemov
- v obliki grafa, pri čemer
  - vozlišča ustrezajo problemskim situacijam
  - povezave ustrezajo dovoljenim akcijam



Predstavitev problema

Problem je definiran s:

- prostorom stanj
- začetnim stanjem (lahko več)
- končnim stanjem (lahko več)

Reševanje problema zahteva preiskovanje grafa:

- rešitev problema je pot od začetnega do končnega stanja
- optimizacijske probleme predstavimo tako, da povezavam v grafu dodamo cene
- cena rešitve je vsota vseh povezav vzdolž rešitvene poti

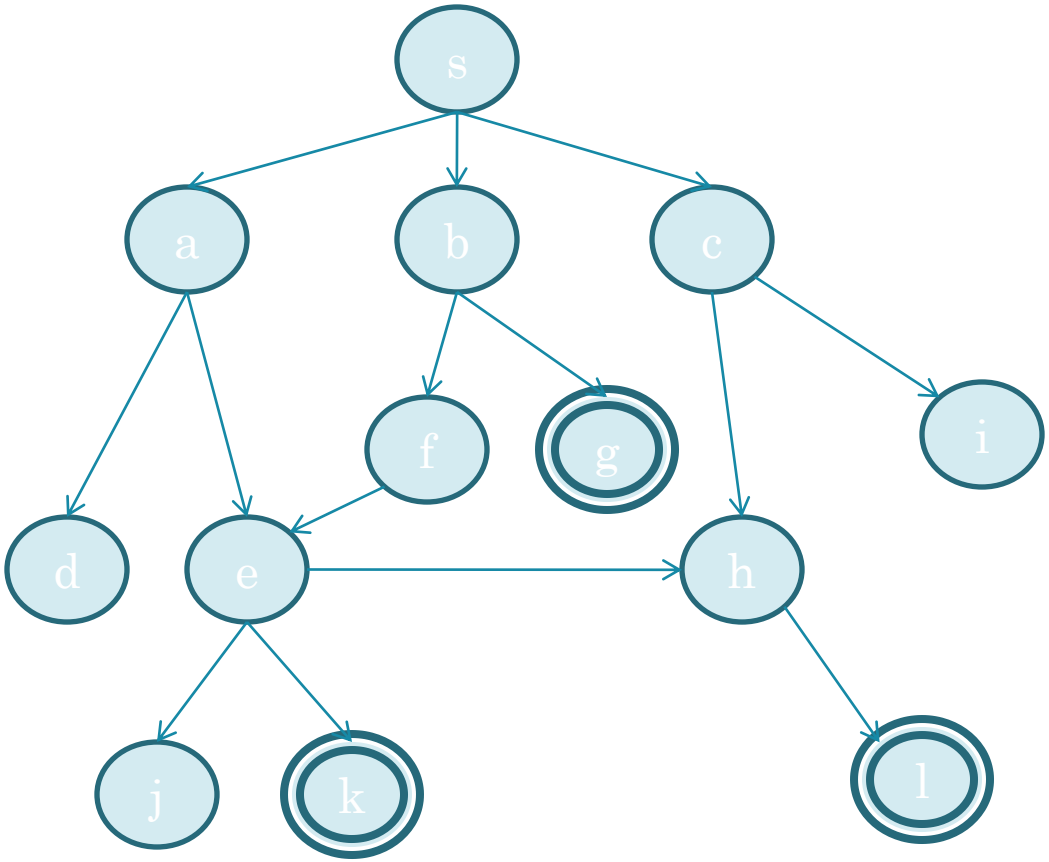
NEINFORMIRANO PREISKOVANJE

Osnovni strategiji za sistematično preiskovanje prostora stanj:

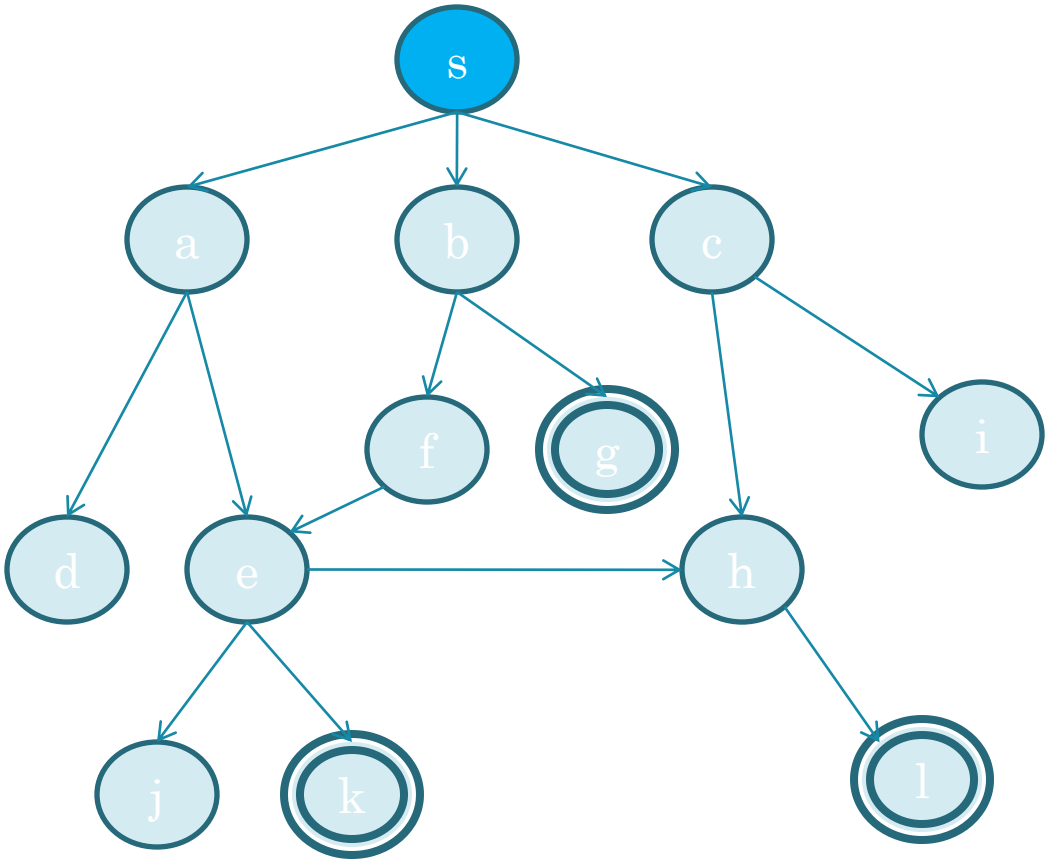
- iskanje v globino
  - med alternativami izbere tisto, ki je najdlje od začetnega stanja
  - najbolje prilega rekurzivnemu stilu programiranja
  - nevarnost zankanja
  - ni nujno, da najprej najde najkrajšo pot
  - časovna zahtevnost reda $O(b^m)$, prostorska zahtevnost reda $O(bm)$
- iskanje v širino
  - med alternativami izbere tisto najbližjo začetnemu stanju
  - vedno najprej najde najkrajšo pot
  - moramo voditi množico poti-kandidatov
  - časovna zahtevnost reda $O(b^{d+1})$, prostorska zahtevnost reda $O(b^{d+1})$

b – faktor vejanja grafa
d – globina najbližjega končnega stanja
m – maksimalna globina prostora stanj

PRIMER - ISKANJE V GLOBINO (1/9)

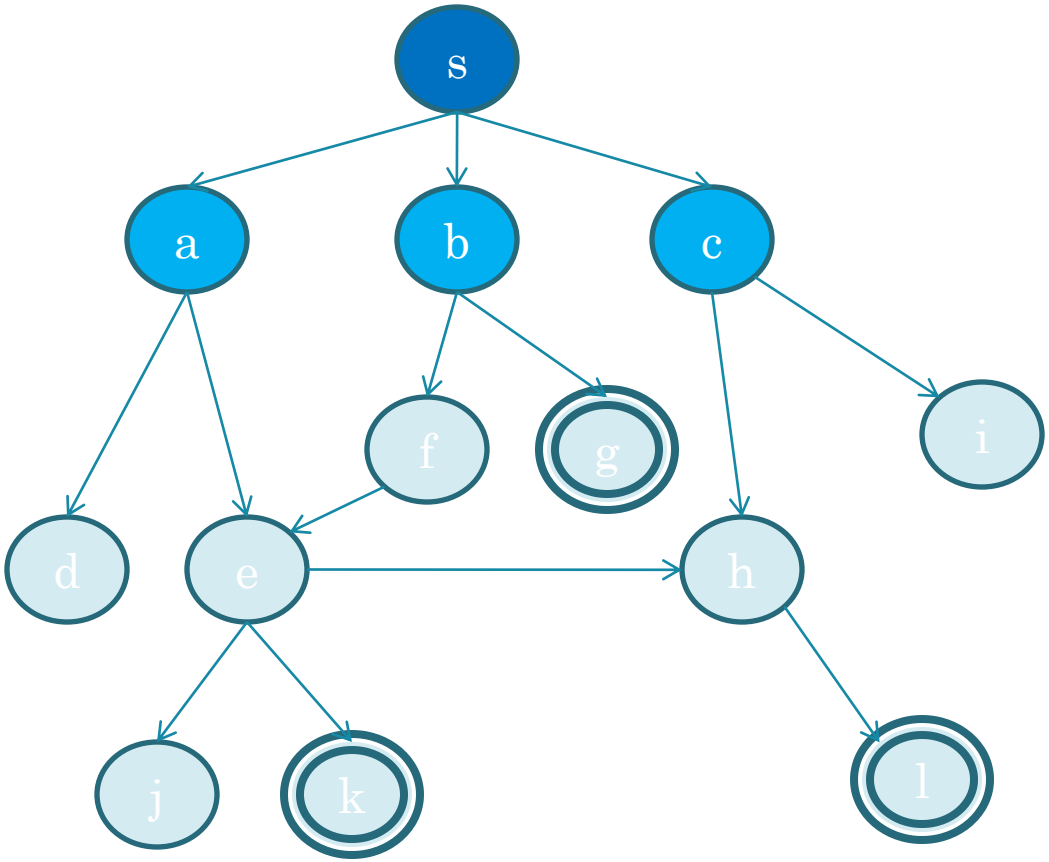


PRIMER - ISKANJE V GLOBINO (2/9)



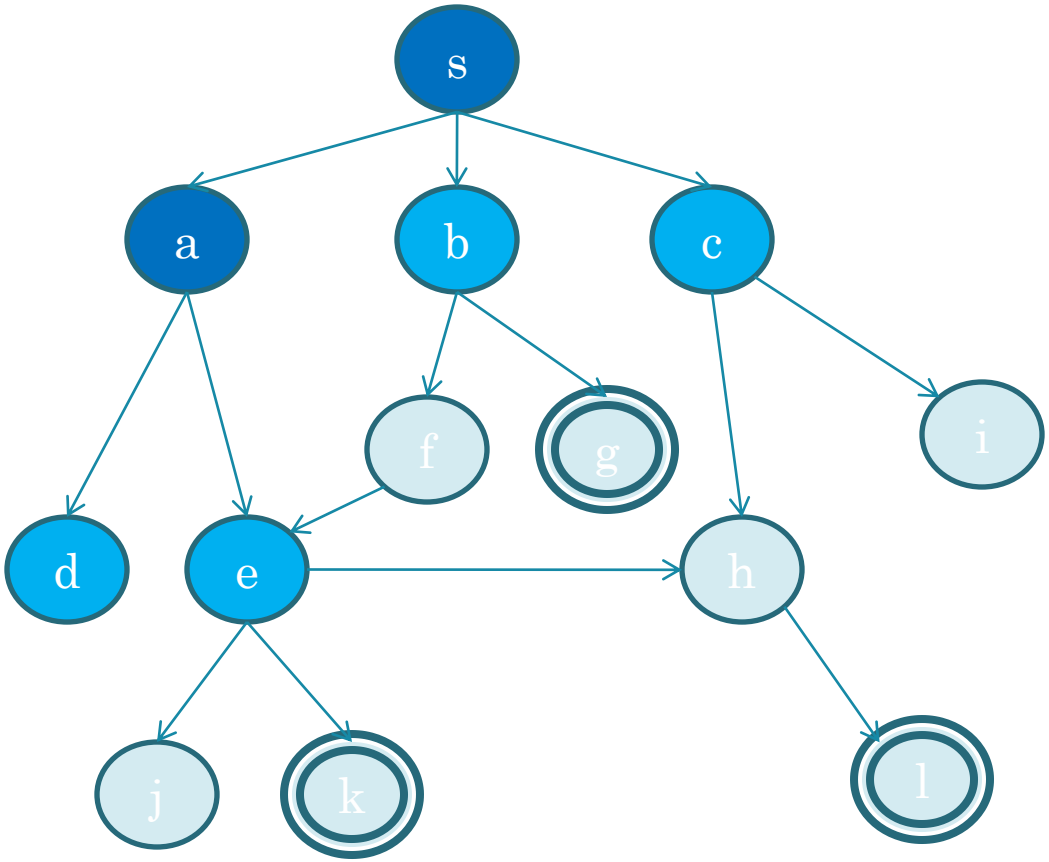
Open list: s ⟸ po vrsti obdelujemo

PRIMER - ISKANJE V GLOBINO (3/9)



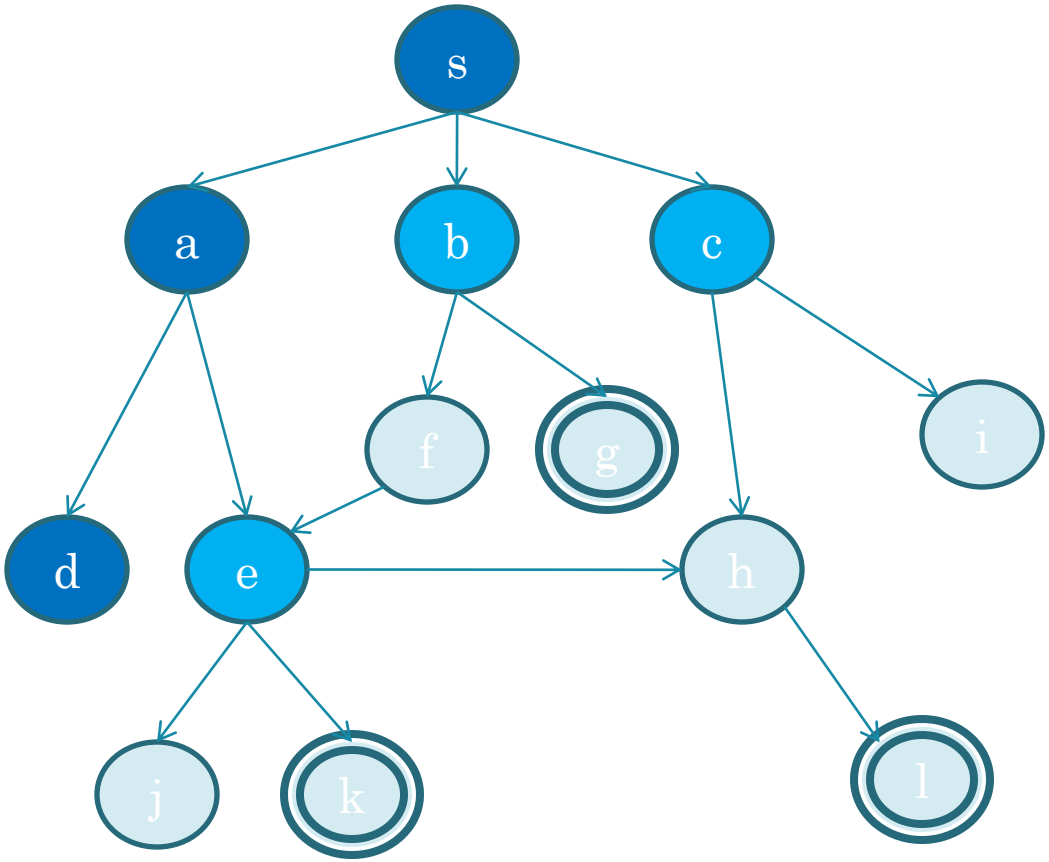
Open list: a, b, c ⇦ obdelano vozlišče odstranimo in dodamo njegove naslednike

PRIMER - ISKANJE V GLOBINO (4/9)



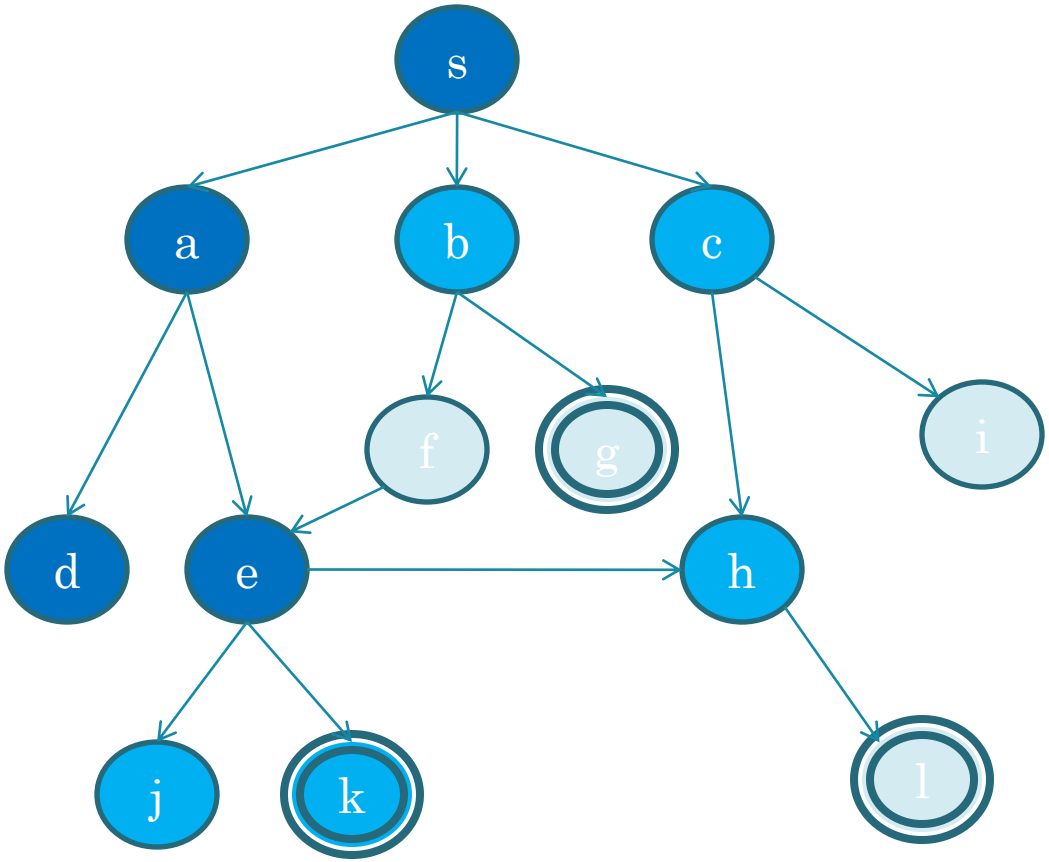
Open list: d, e, b, c ⇐ naslednike vozlišča ‘a’ smo dodali na **začetek** vrste

PRIMER - ISKANJE V GLOBINO (5/9)



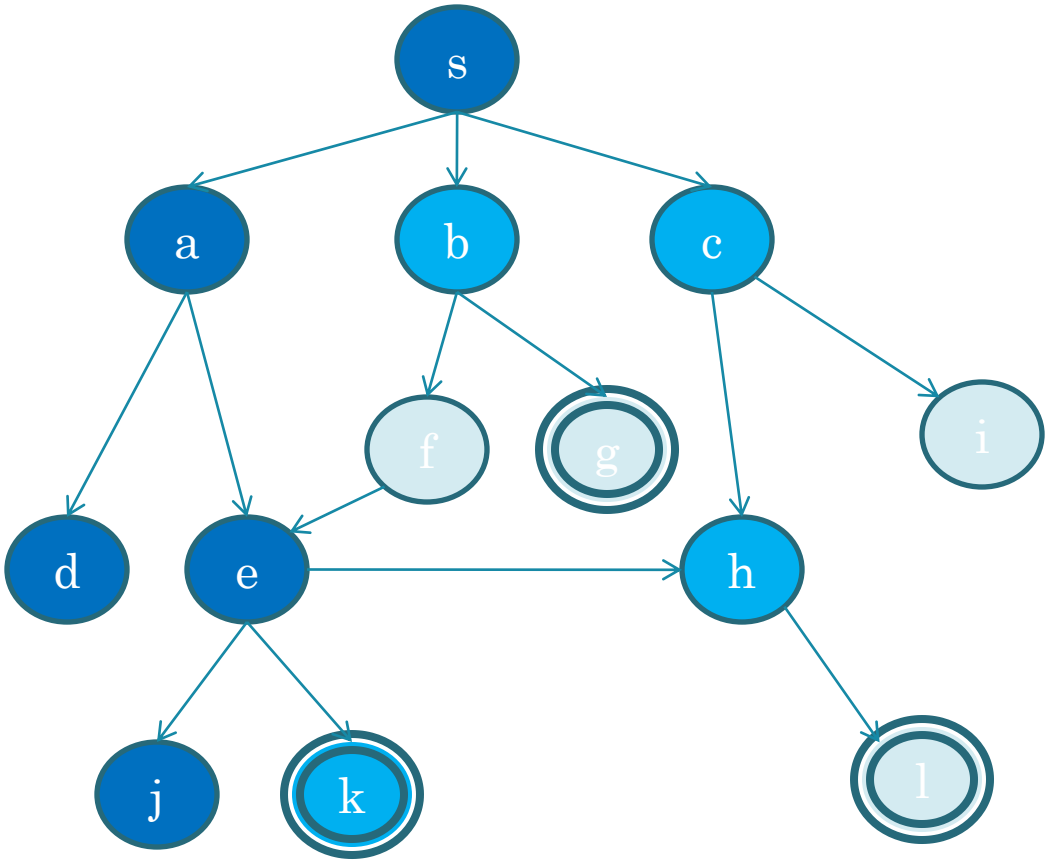
Open list: e, b, c

PRIMER - ISKANJE V GLOBINO (6/9)



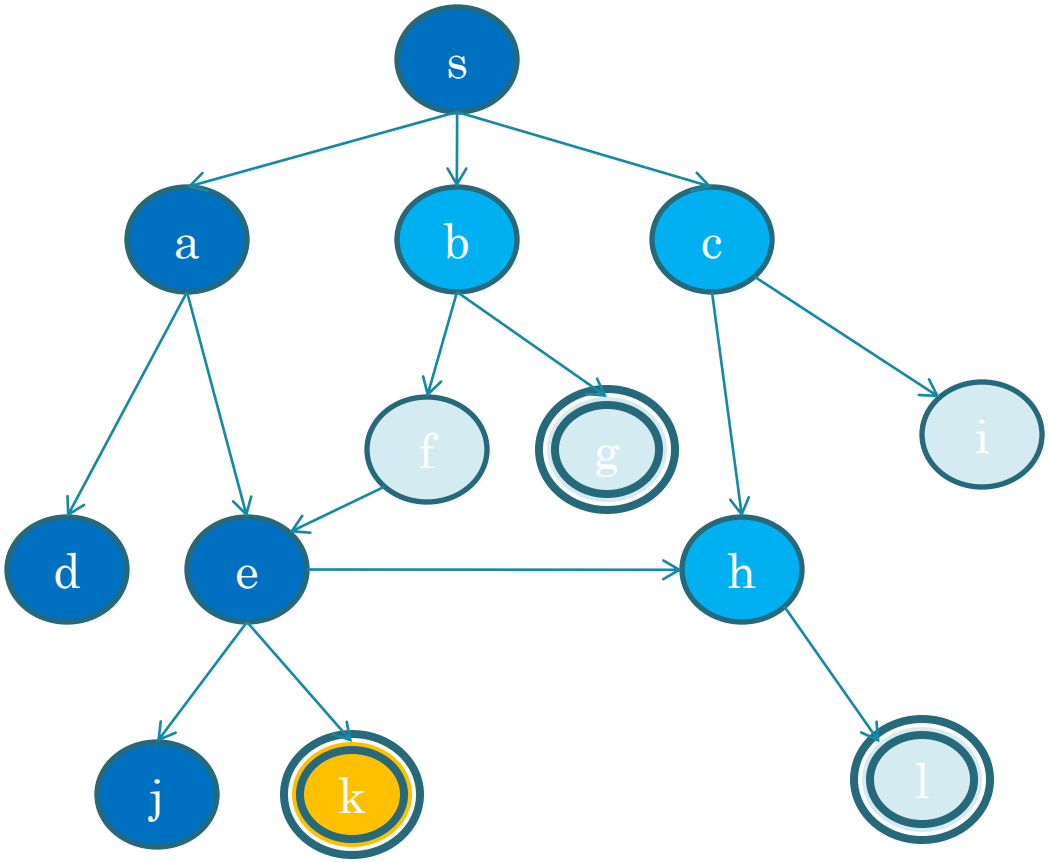
Open list: j, k, h, b, c

PRIMER - ISKANJE V GLOBINO (7/9)



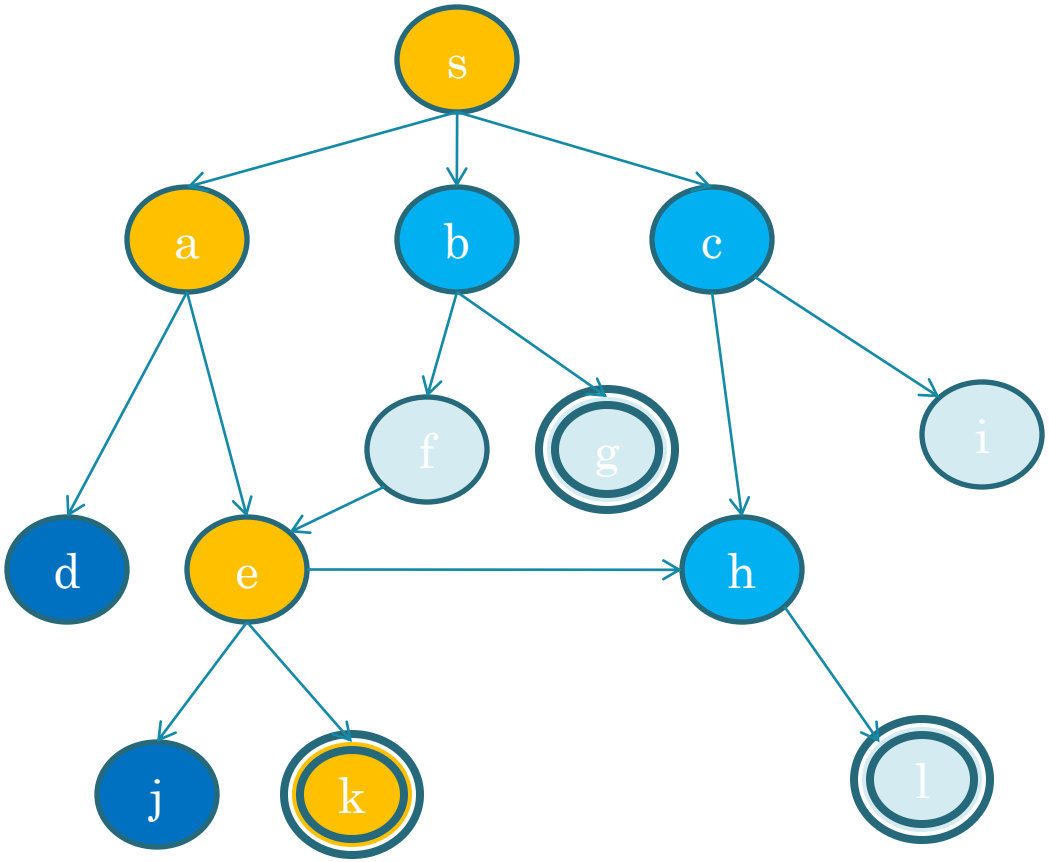
Open list: k, h, b, c

PRIMER - ISKANJE V GLOBINO (8/9)



Open list: h, b, c

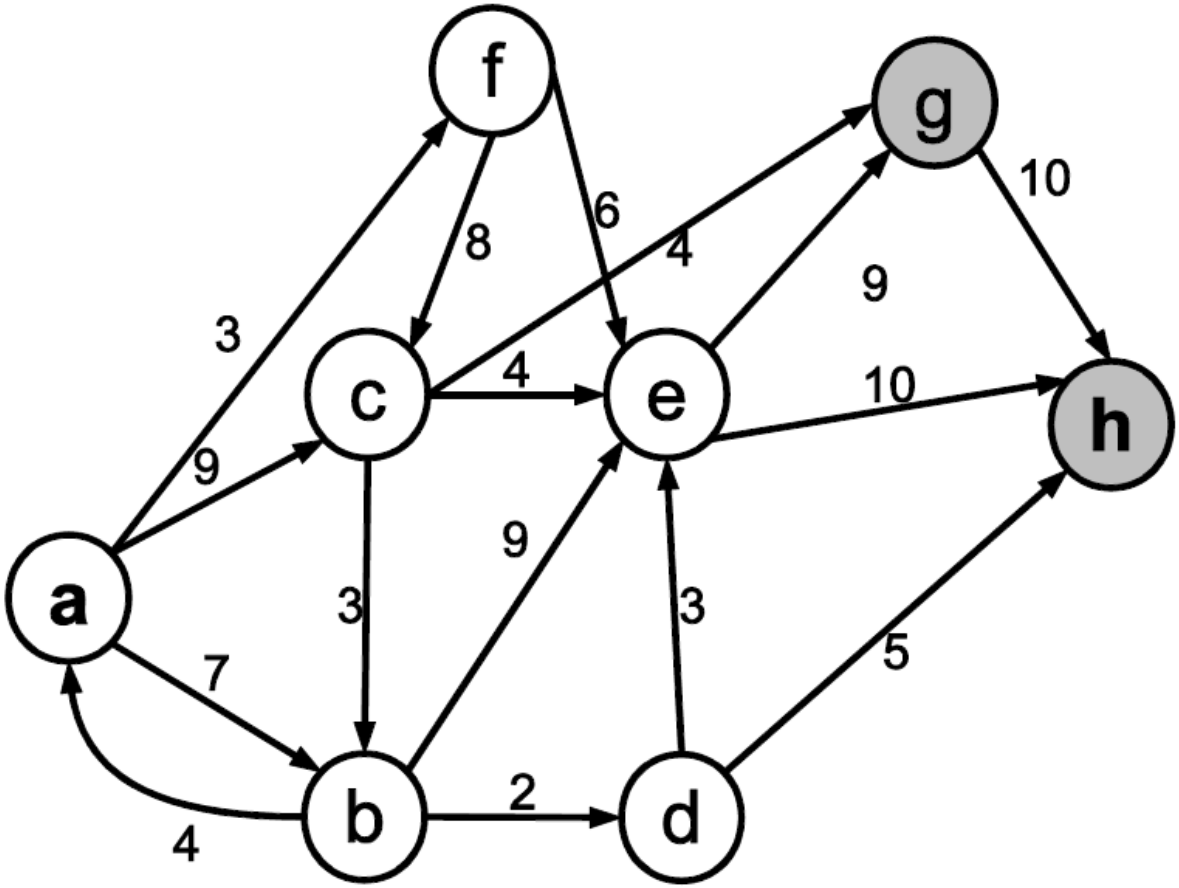
PRIMER - ISKANJE V GLOBINO (9/9)



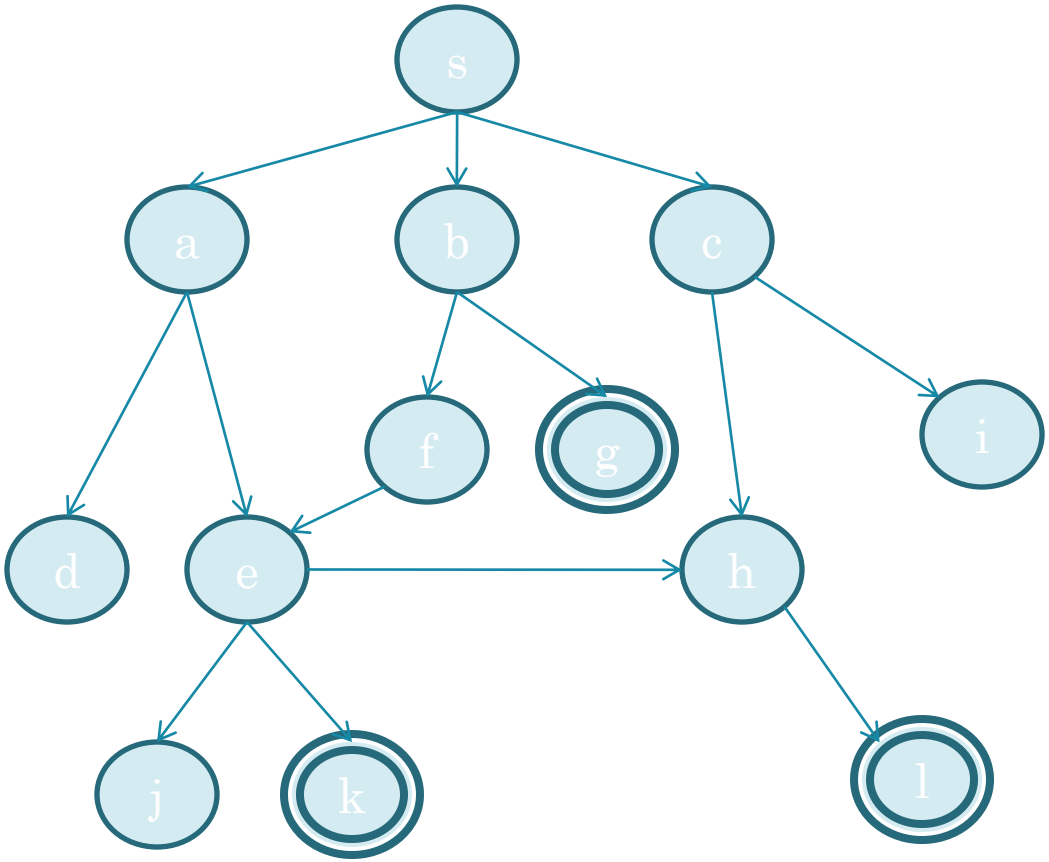
Rešitev: s, a, e, k

Samostojno delo

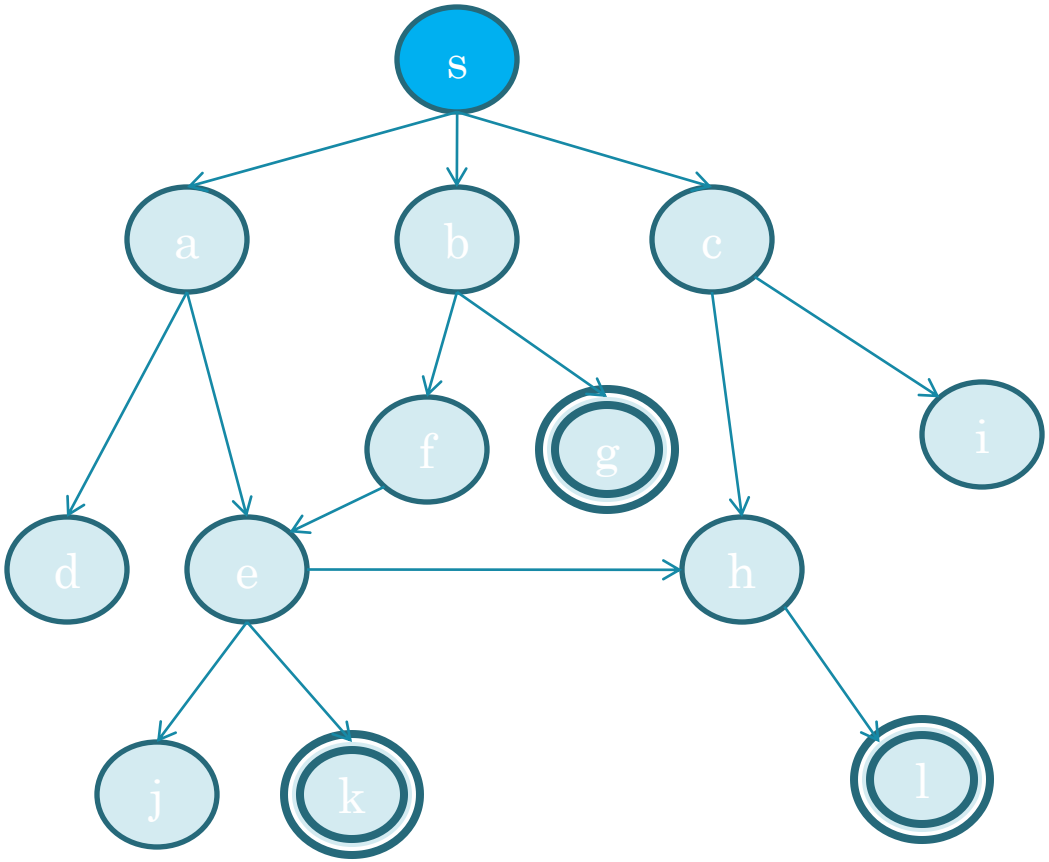
Za spodnji graf z začetnim vozliščem *a* in s končnima vozliščema *g* in *h* določi zaporedje razvijanja vozlišč, če uporabljamo preiskovanje v globino.



PRIMER - ISKANJE V ŠIRINO (1/11)

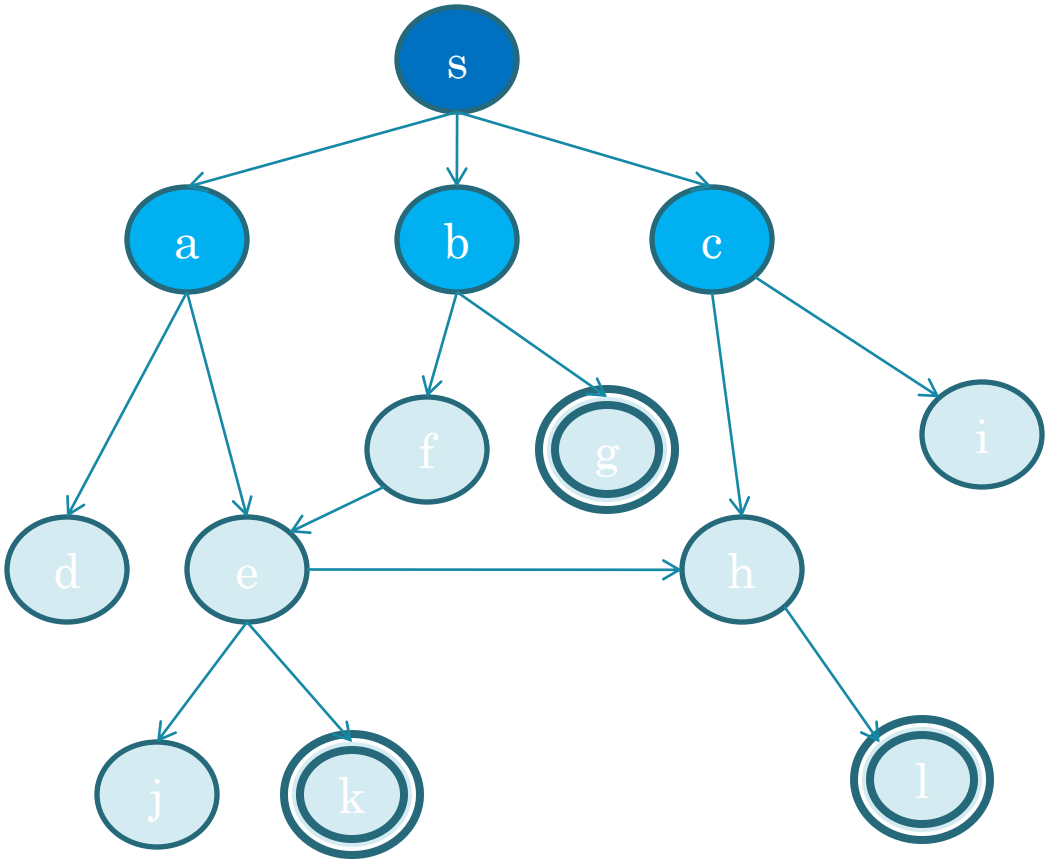


PRIMER - ISKANJE V ŠIRINO (2/11)



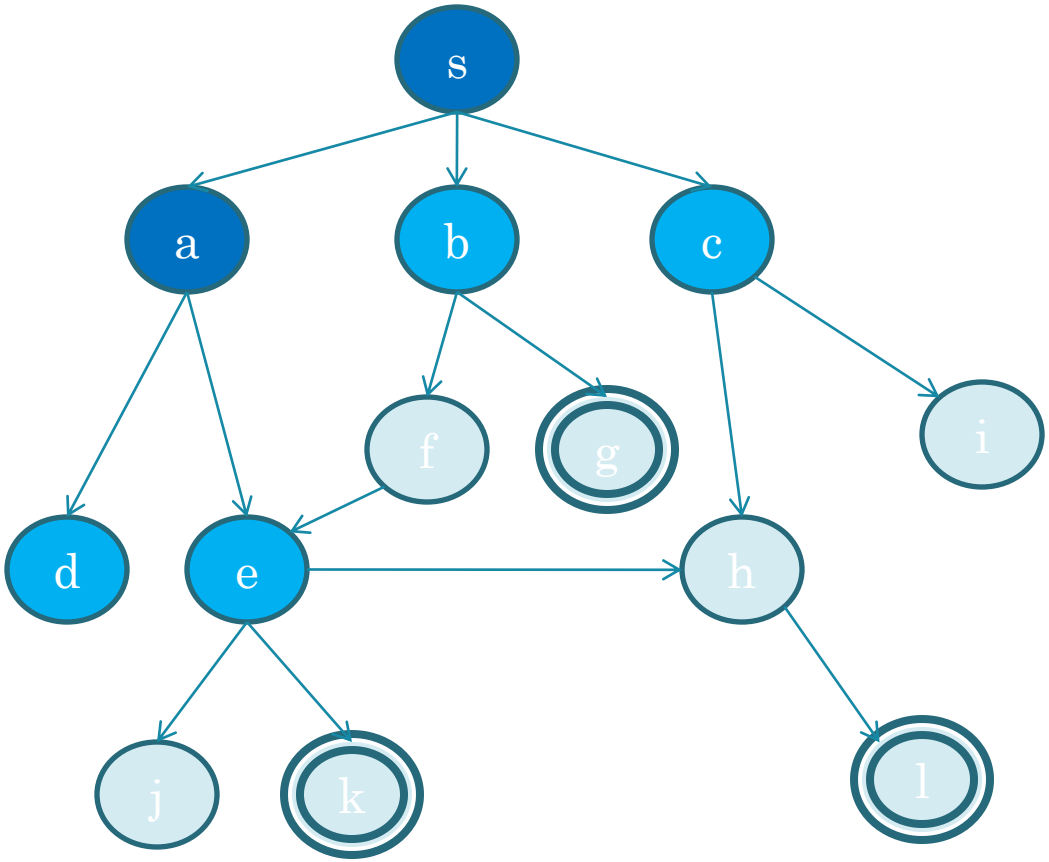
Open list: s

PRIMER - ISKANJE V ŠIRINO (3/11)



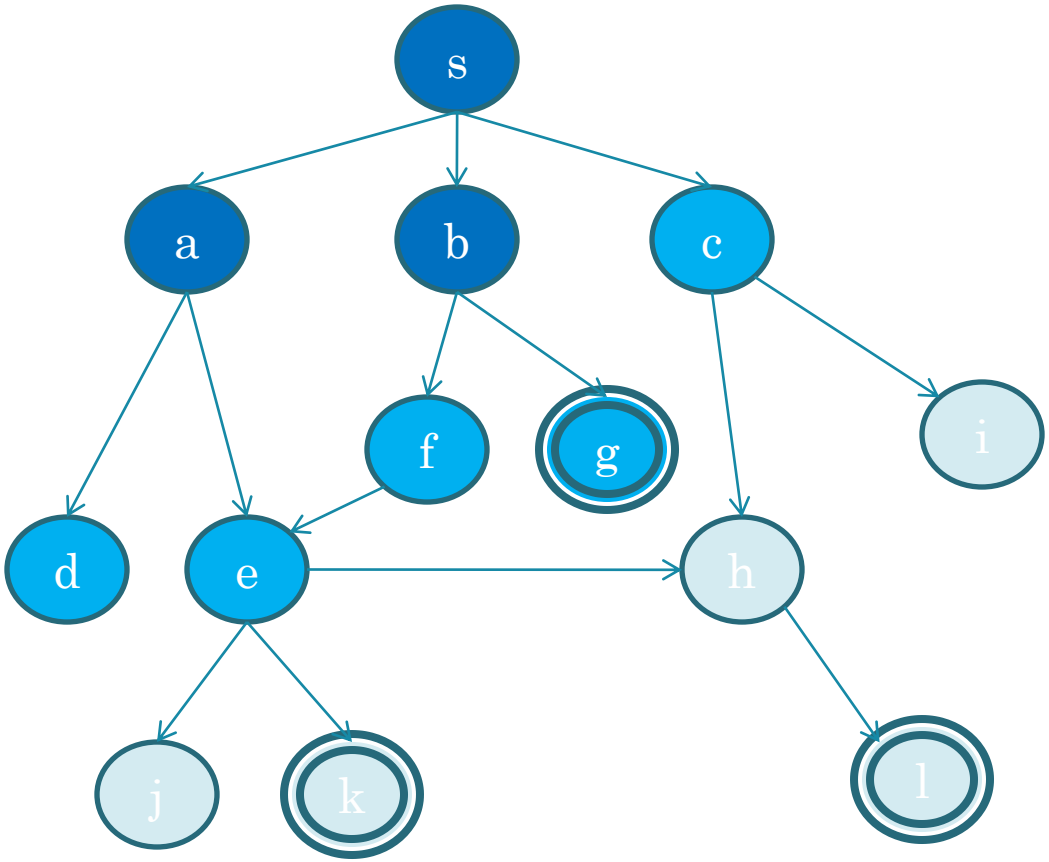
Open list: a, b, c

PRIMER - ISKANJE V ŠIRINO (4/11)



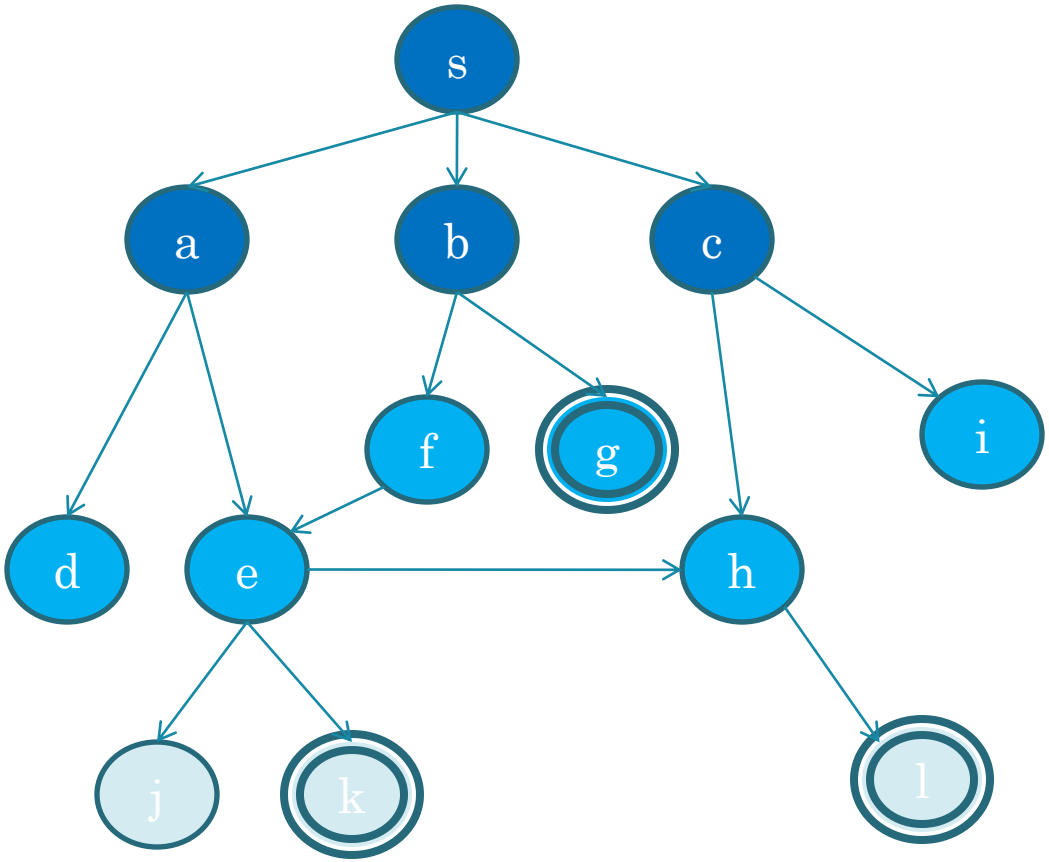
Open list: b, c, d, e ⟸ naslednike vozlišča ‘a’ smo dodali na **konec** vrste

PRIMER - ISKANJE V ŠIRINO (5/11)



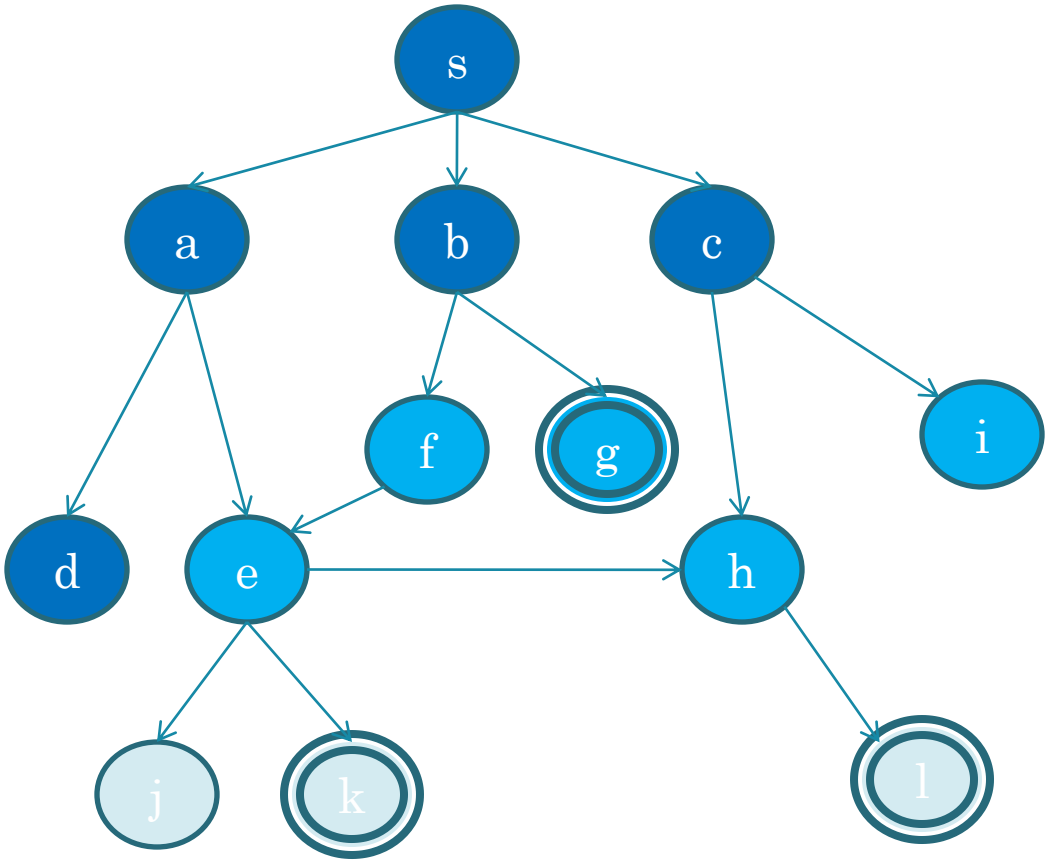
Open list: c, d, e, f, g

PRIMER - ISKANJE V ŠIRINO (6/11)



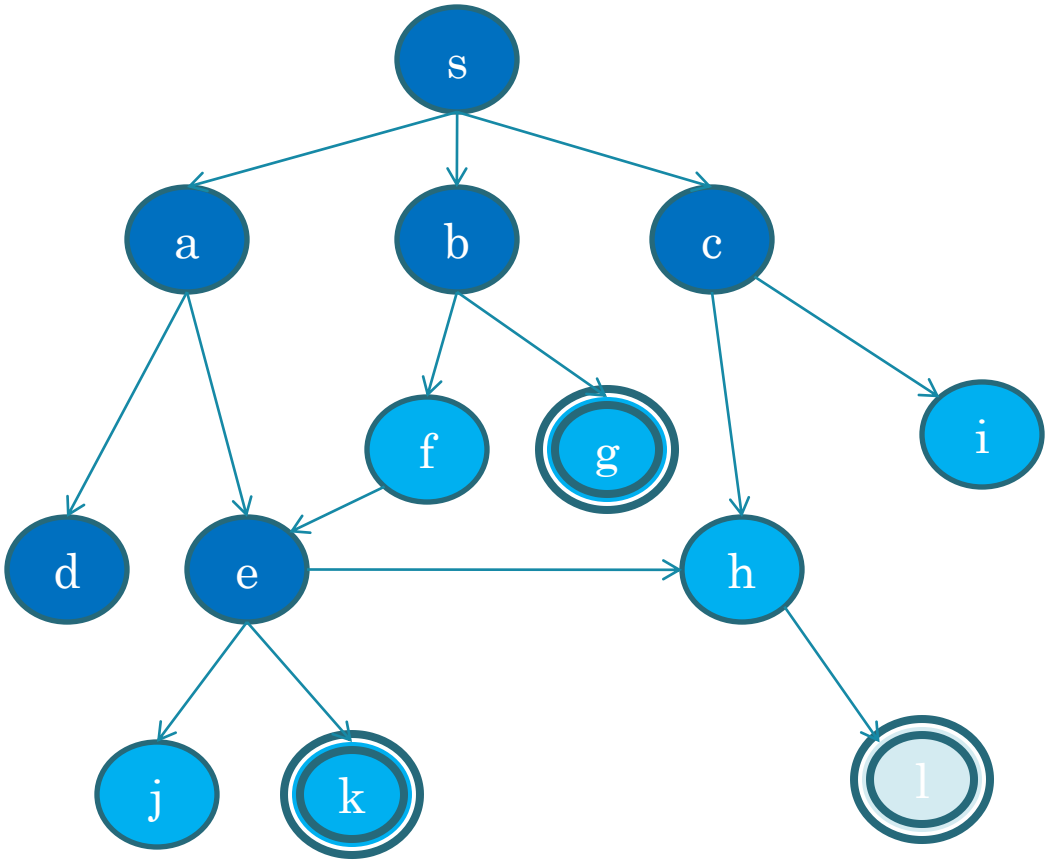
Open list: d, e, f, g, h, i

PRIMER - ISKANJE V ŠIRINO (7/11)



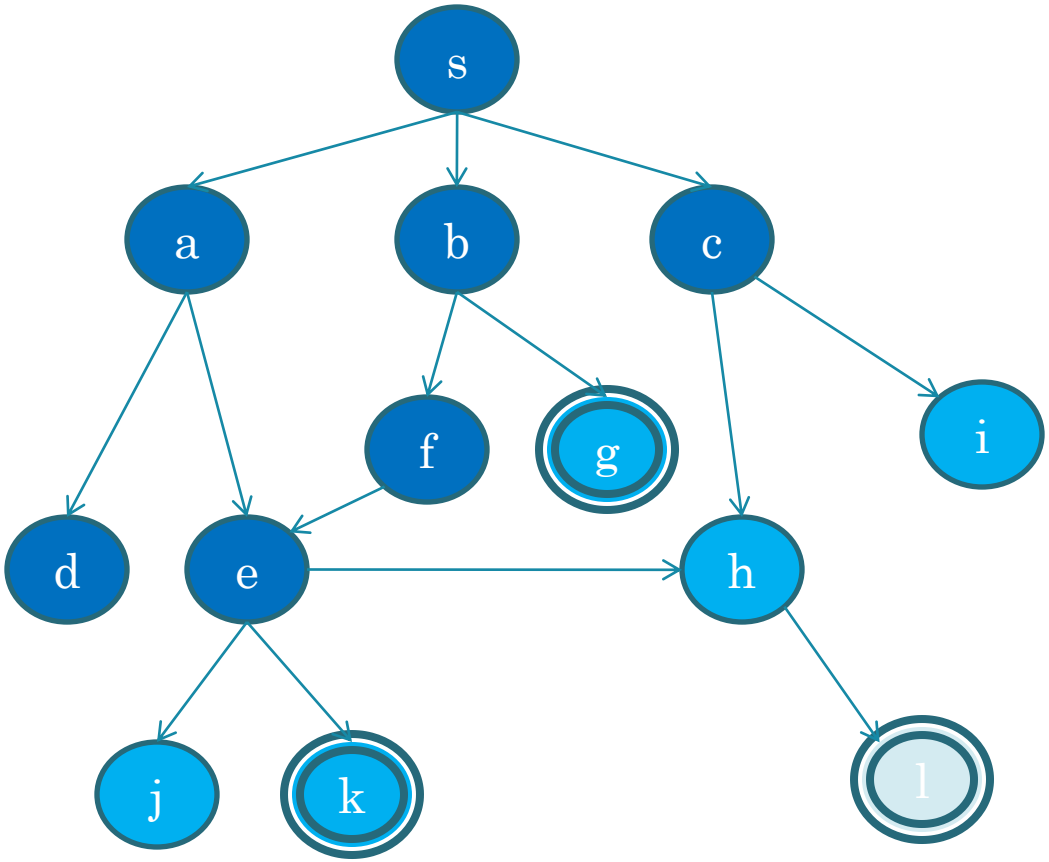
Open list: e, f, g, h, i

PRIMER - ISKANJE V ŠIRINO (8/11)



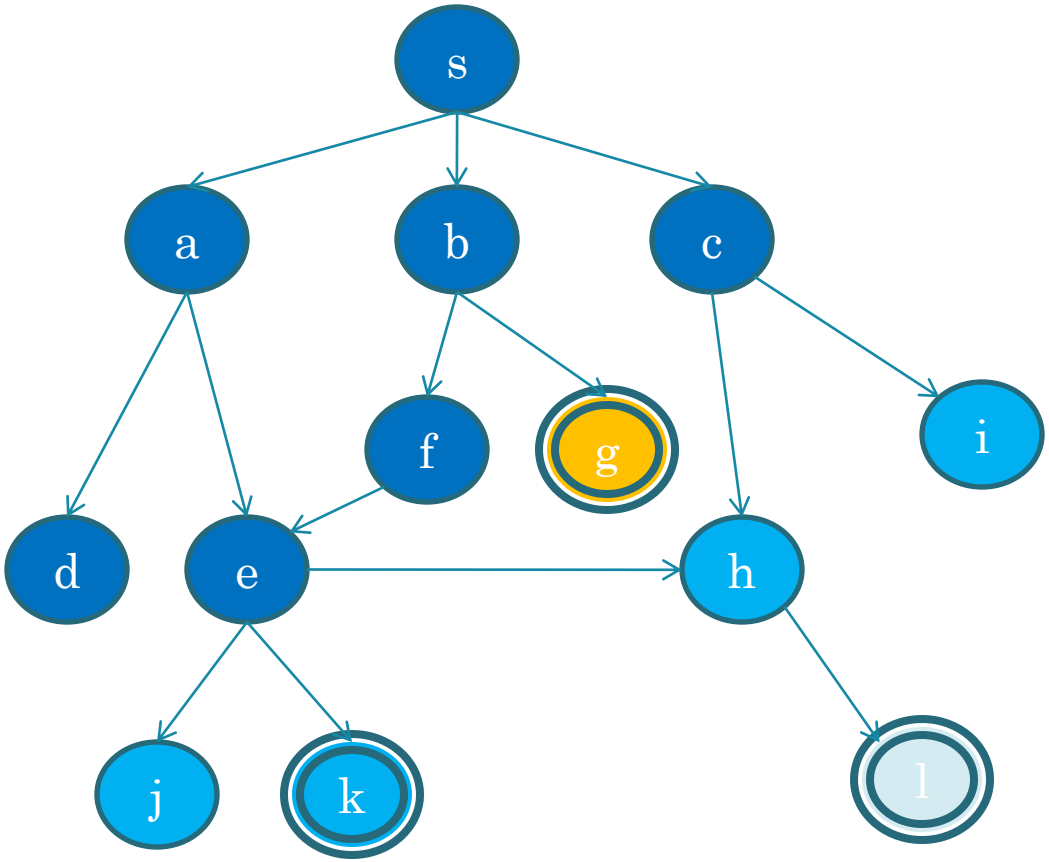
Open list: f, g, h, i, j, k, h

PRIMER - ISKANJE V ŠIRINO (9/11)



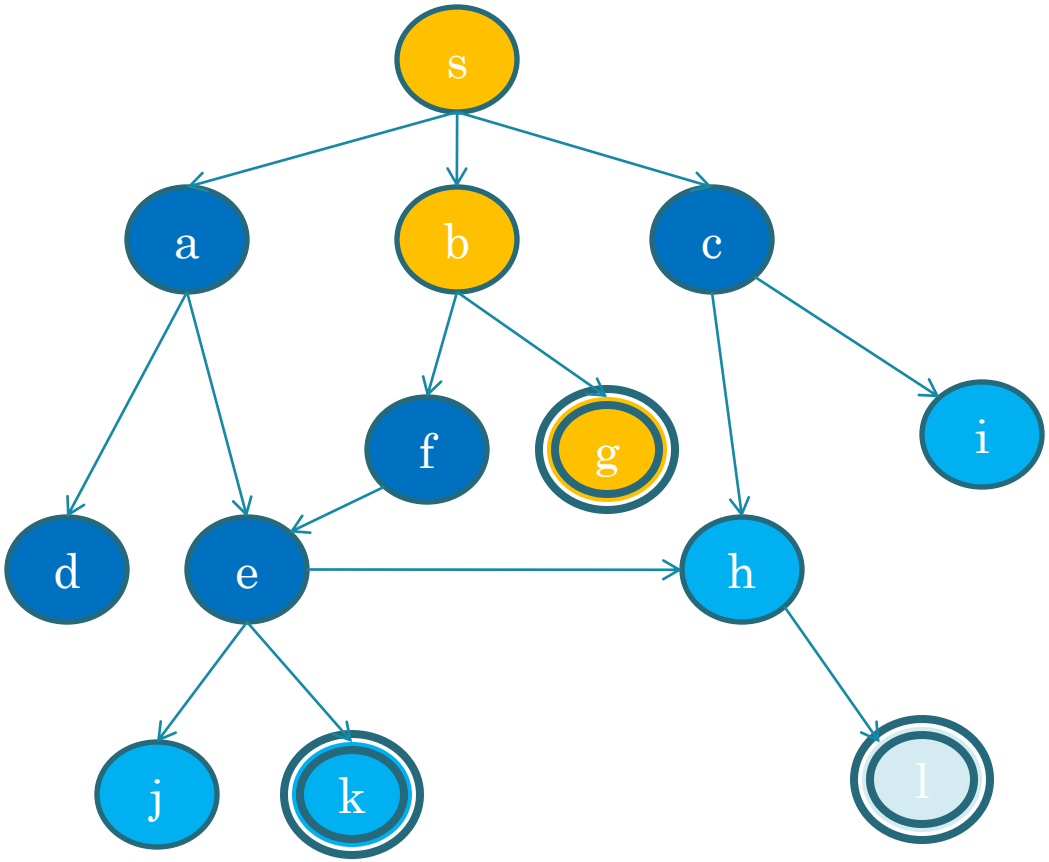
Open list: g, h, i, j, k, h, e

PRIMER - ISKANJE V ŠIRINO (10/11)



Open list: h, i, j, k, h, e

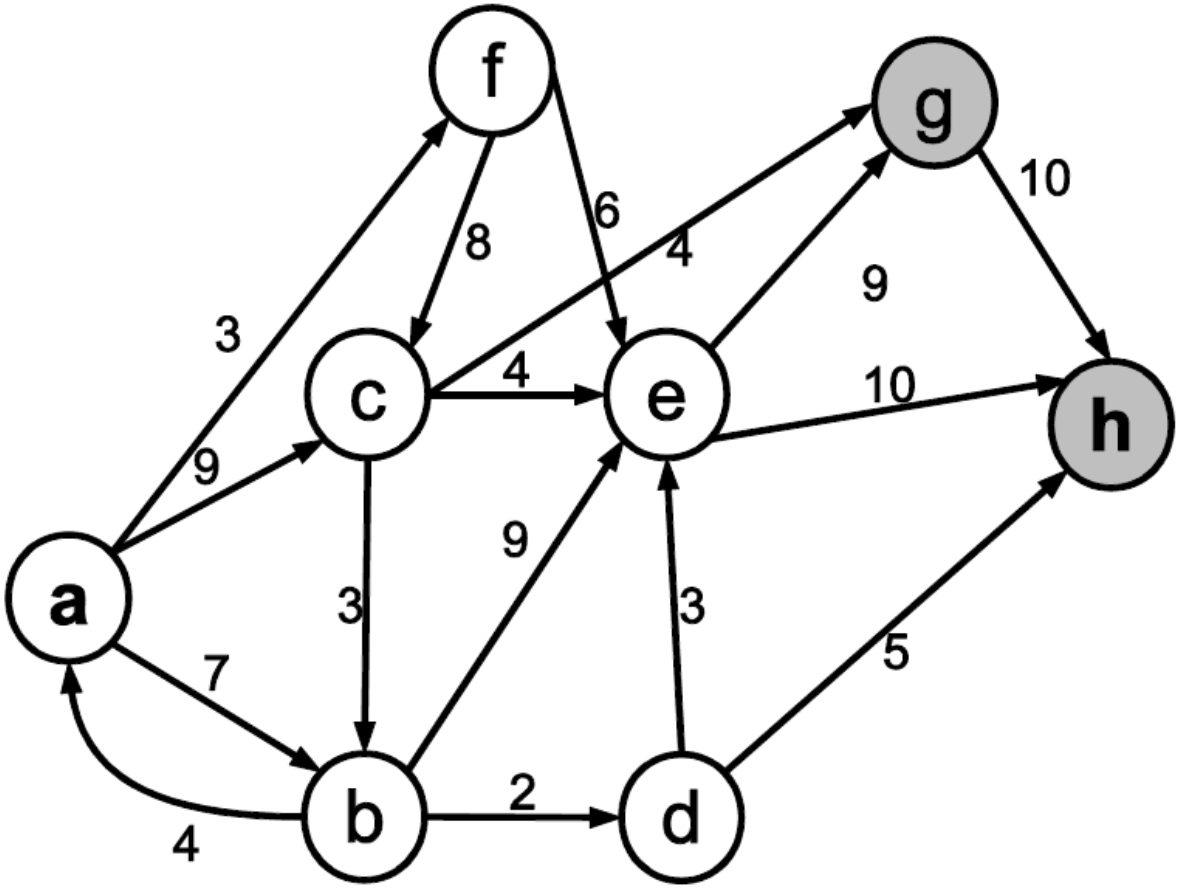
PRIMER - ISKANJE V ŠIRINO (11/11)



Rešitev: s, b, g

Samostojno delo

Za spodnji graf z začetnim vozliščem *a* in s končnima vozliščema *g* in *h* določi zaporedje razvijanja vozlišč, če uporabljamo preiskovanje v širino.



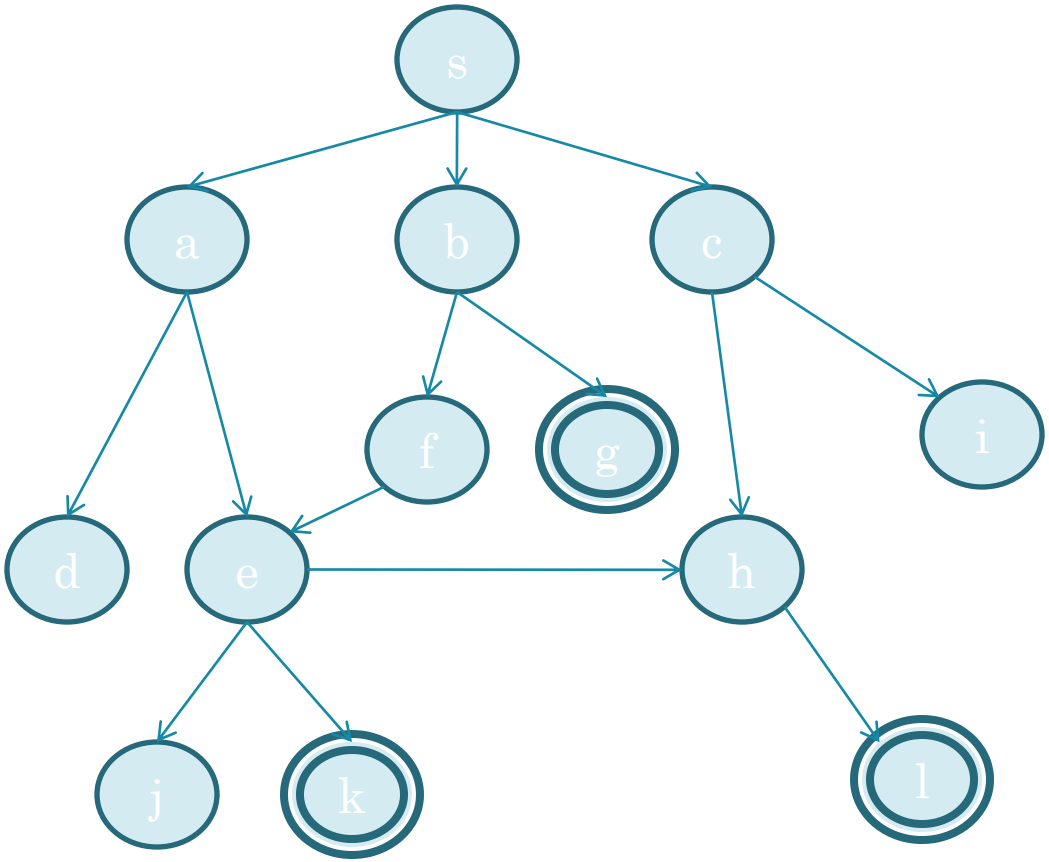
NEINFORMIRANO PREISKOVANJE

Iterativno poglabljanje

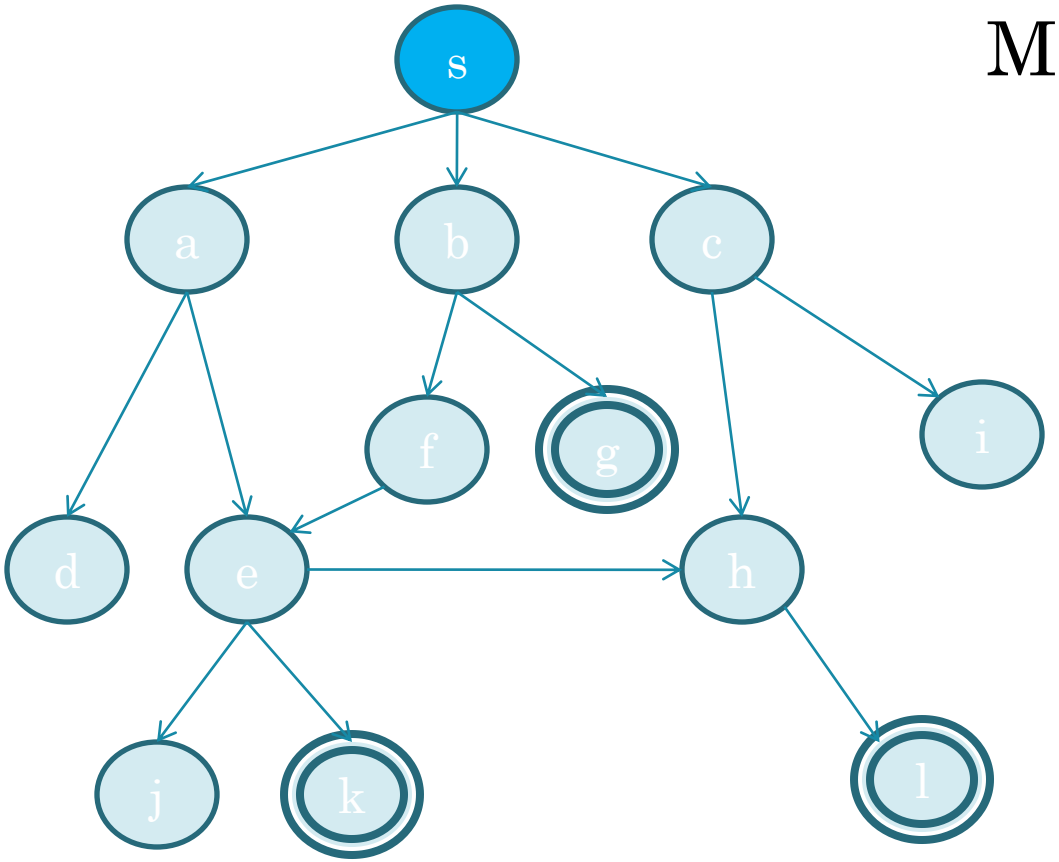
- iskanje v globino z omejeno globino, ki jo iterativno podaljšujemo
  - kombinira prednosti iskanja v globino in iskanja v širino
  - pomnilniško manj zahtevno
  - vedno najprej najde najkrajšo pot
  - časovna zahtevnost reda $O(b^d)$, prostorska zahtevnost reda $O(bd)$

b – faktor vejanja grafa
d – globina najbližjega končnega stanja

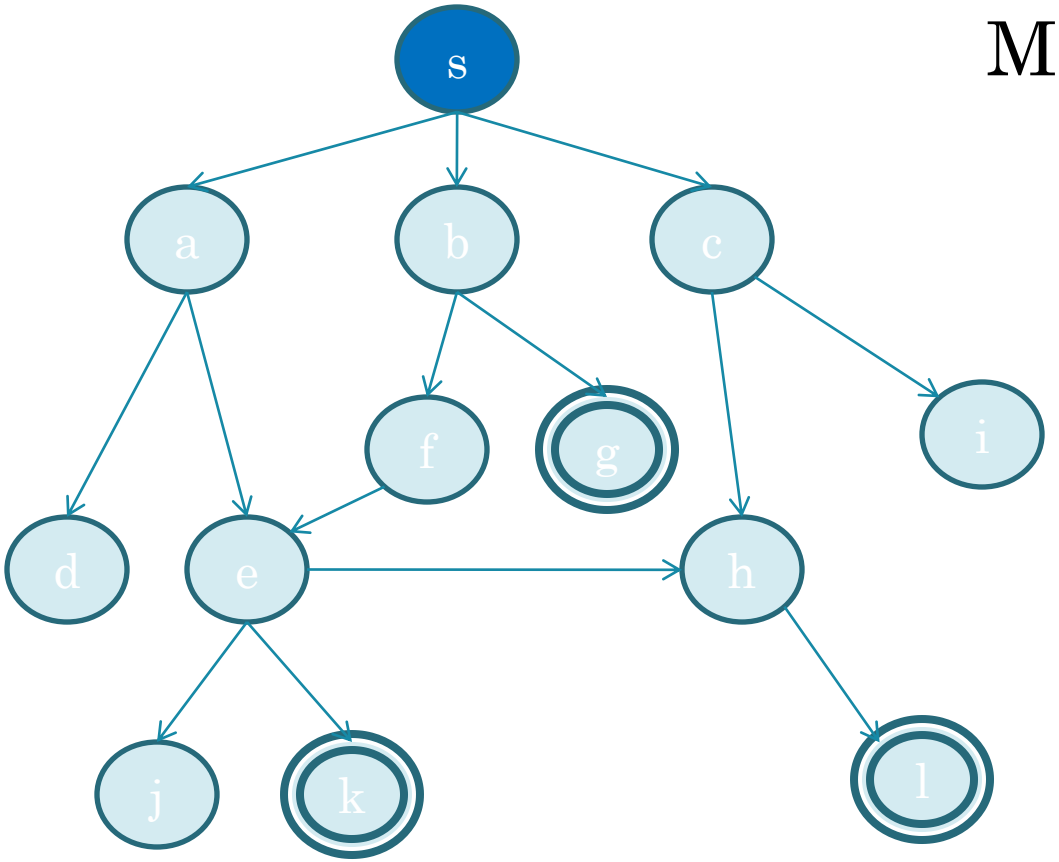
PRIMER – ITERATIVNO POGLABLJANJE (1/17)



PRIMER – ITERATIVNO POGLABLJANJE (2/17)

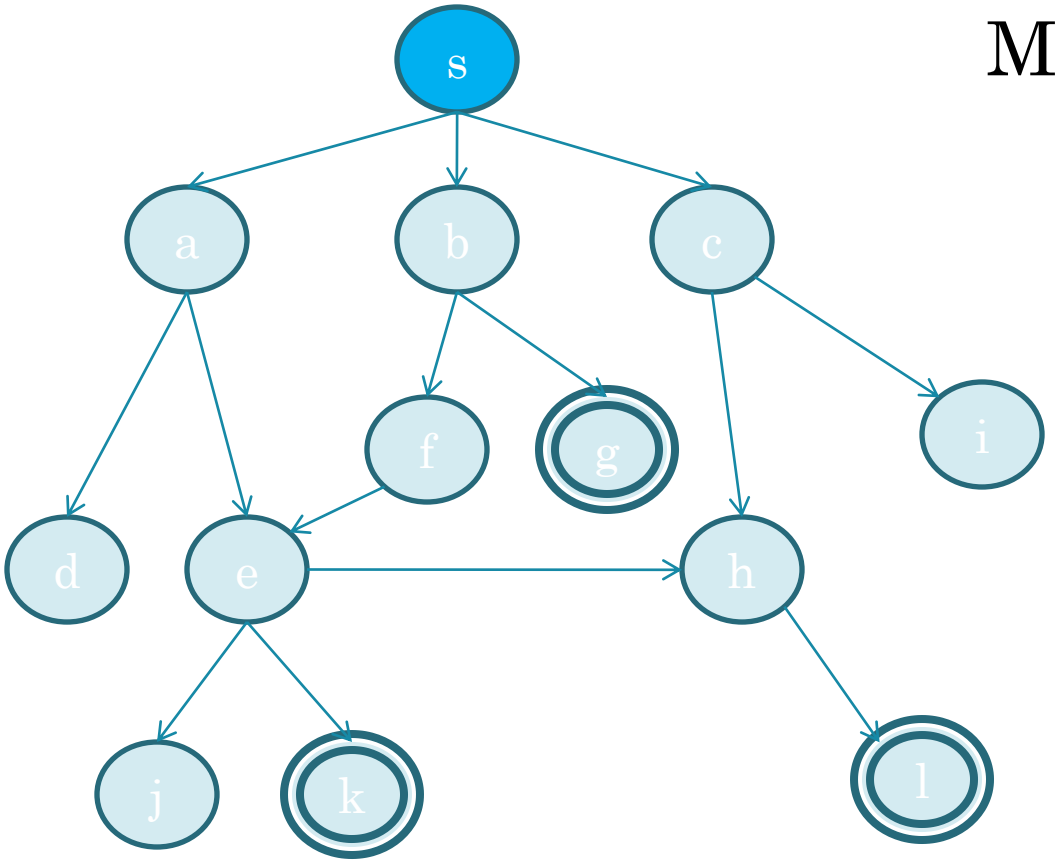


PRIMER – ITERATIVNO POGLABLJANJE (3/17)

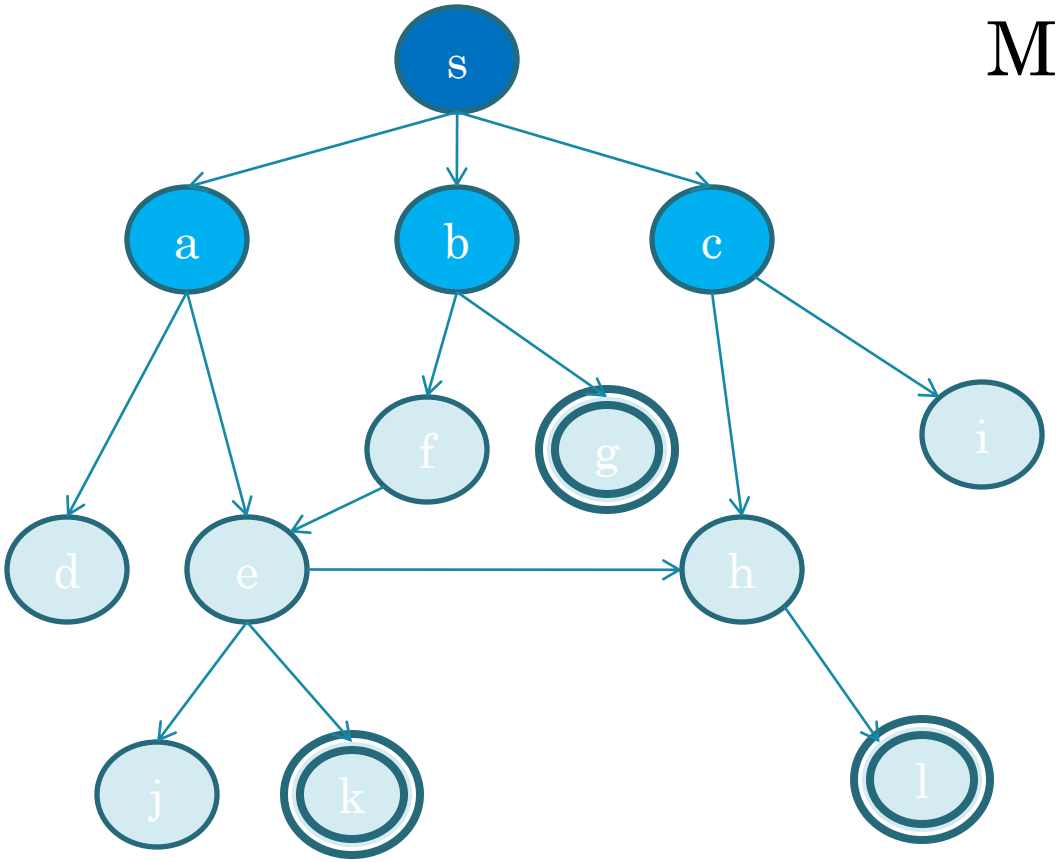


Meja 0

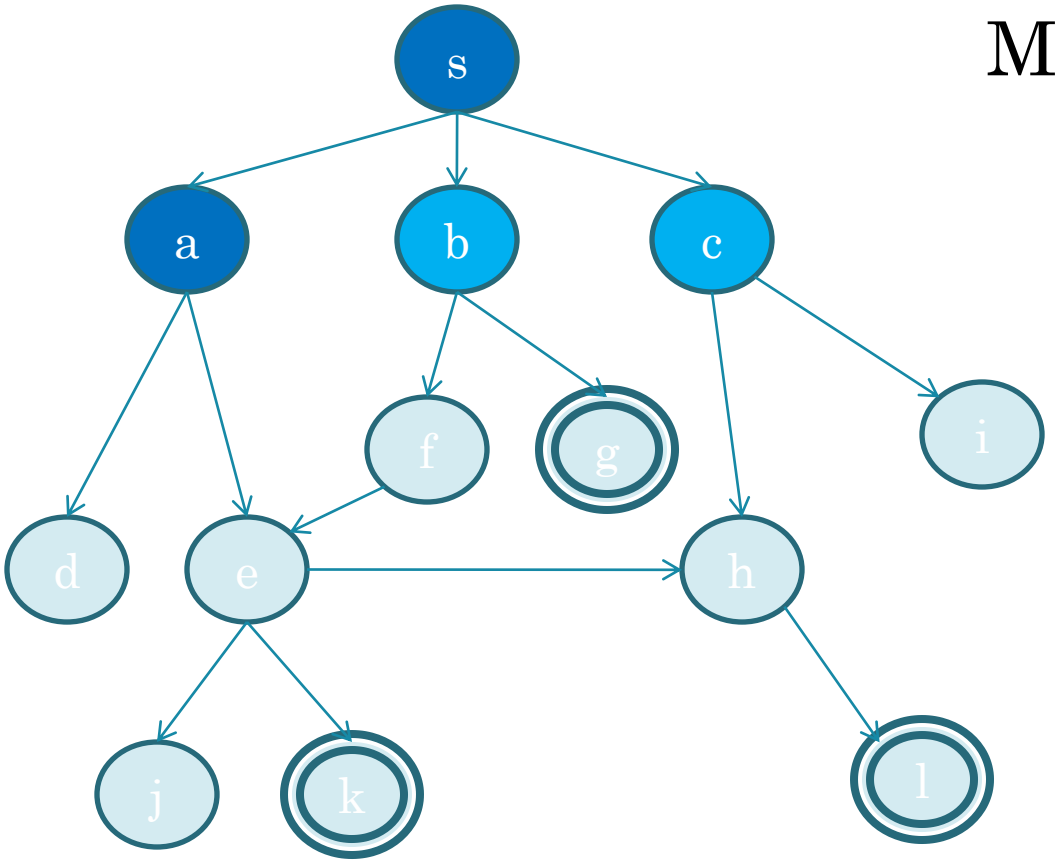
PRIMER – ITERATIVNO POGLABLJANJE (4/17)



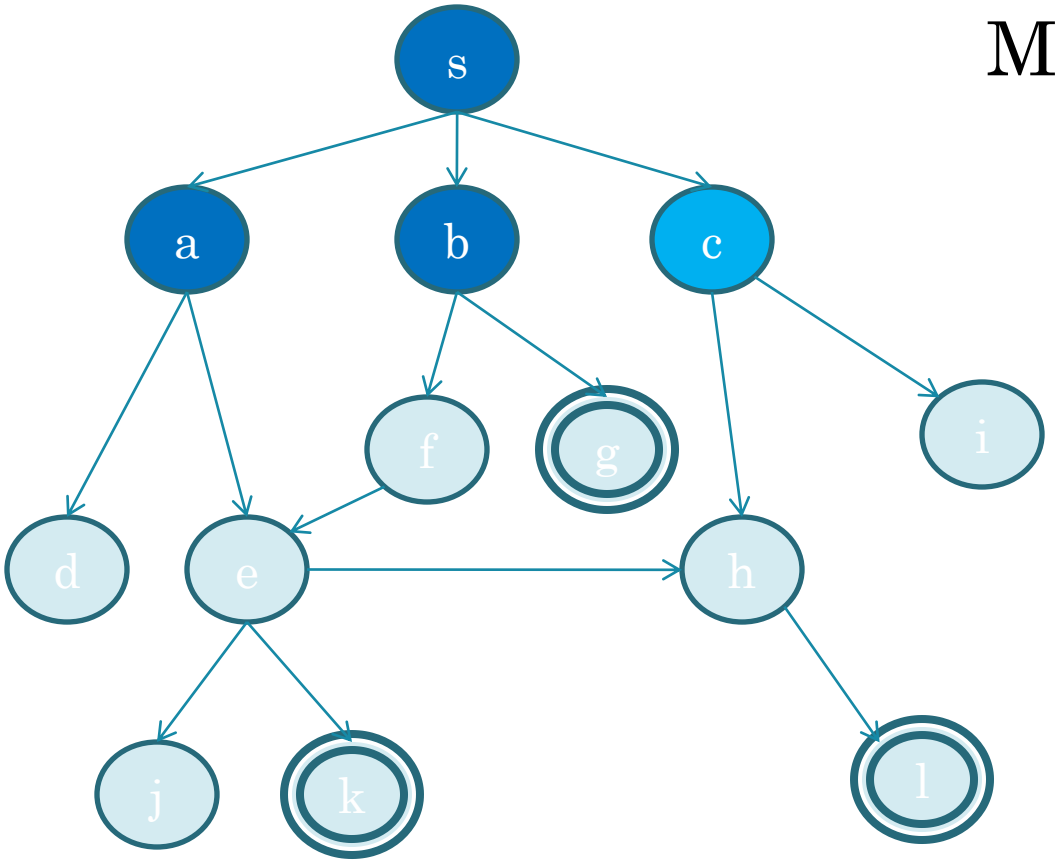
PRIMER – ITERATIVNO POGLABLJANJE (5/17)



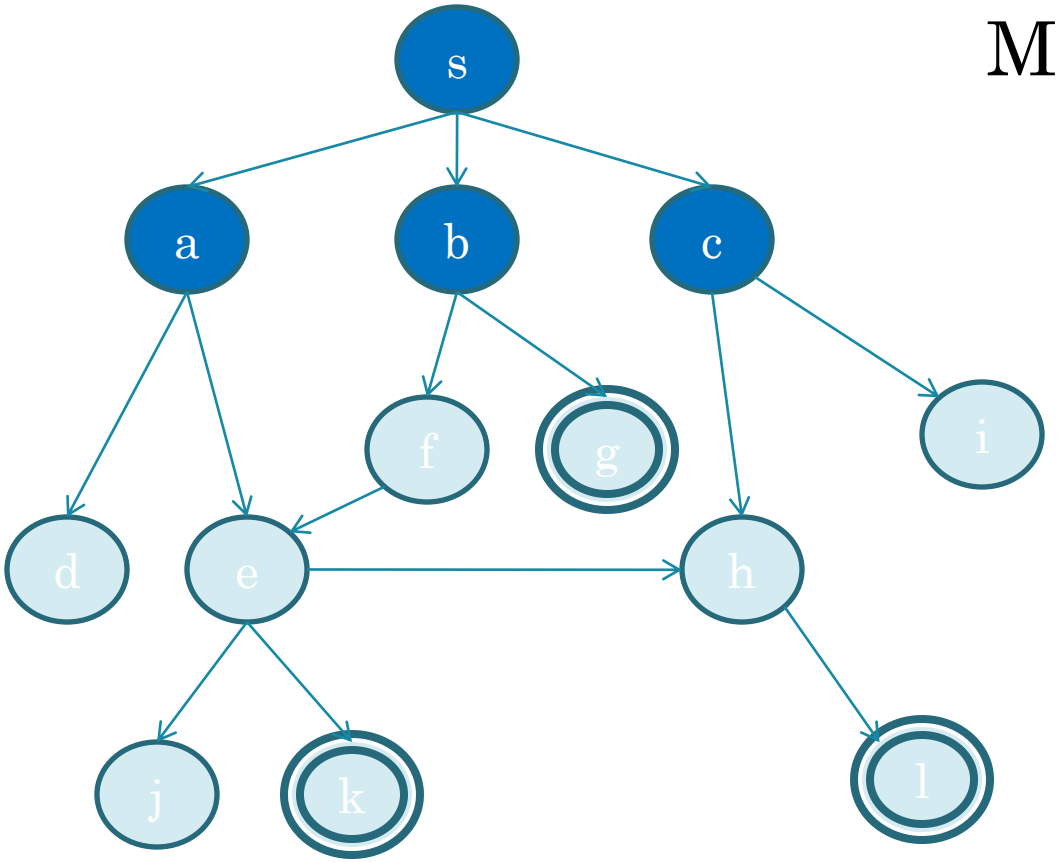
PRIMER – ITERATIVNO POGLABLJANJE (6/17)



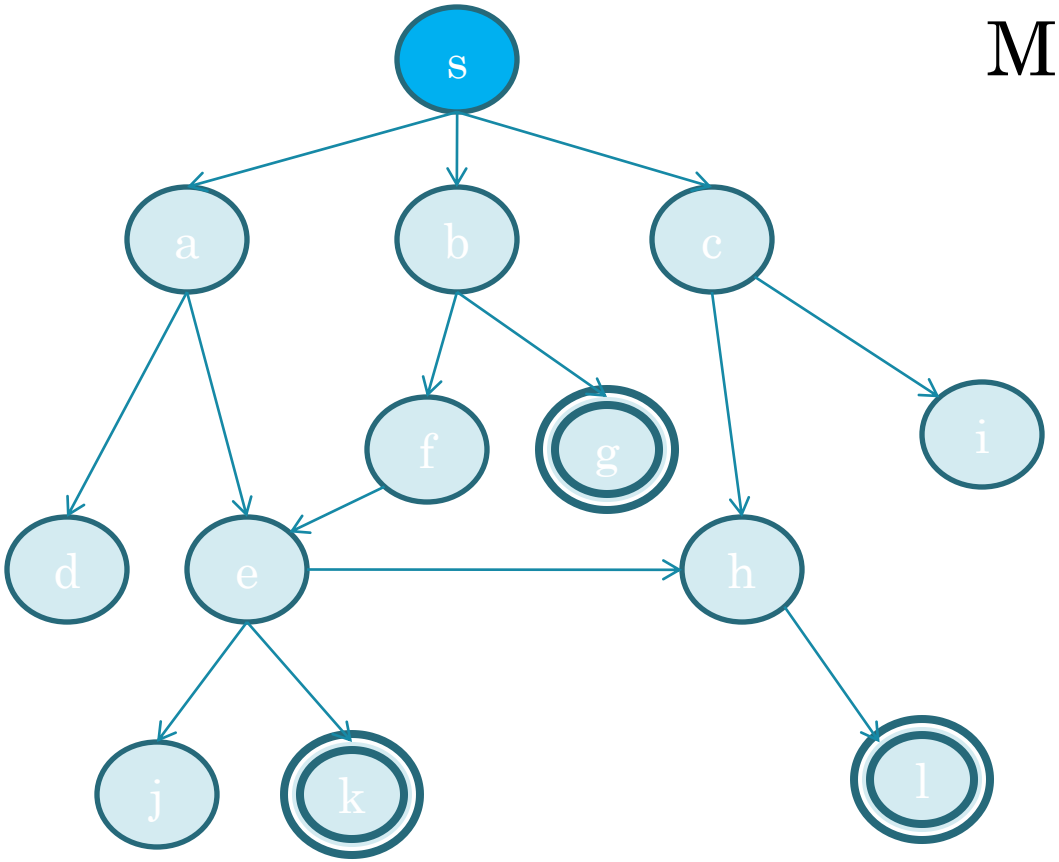
PRIMER – ITERATIVNO POGLABLJANJE (7/17)



PRIMER – ITERATIVNO POGLABLJANJE (8/17)

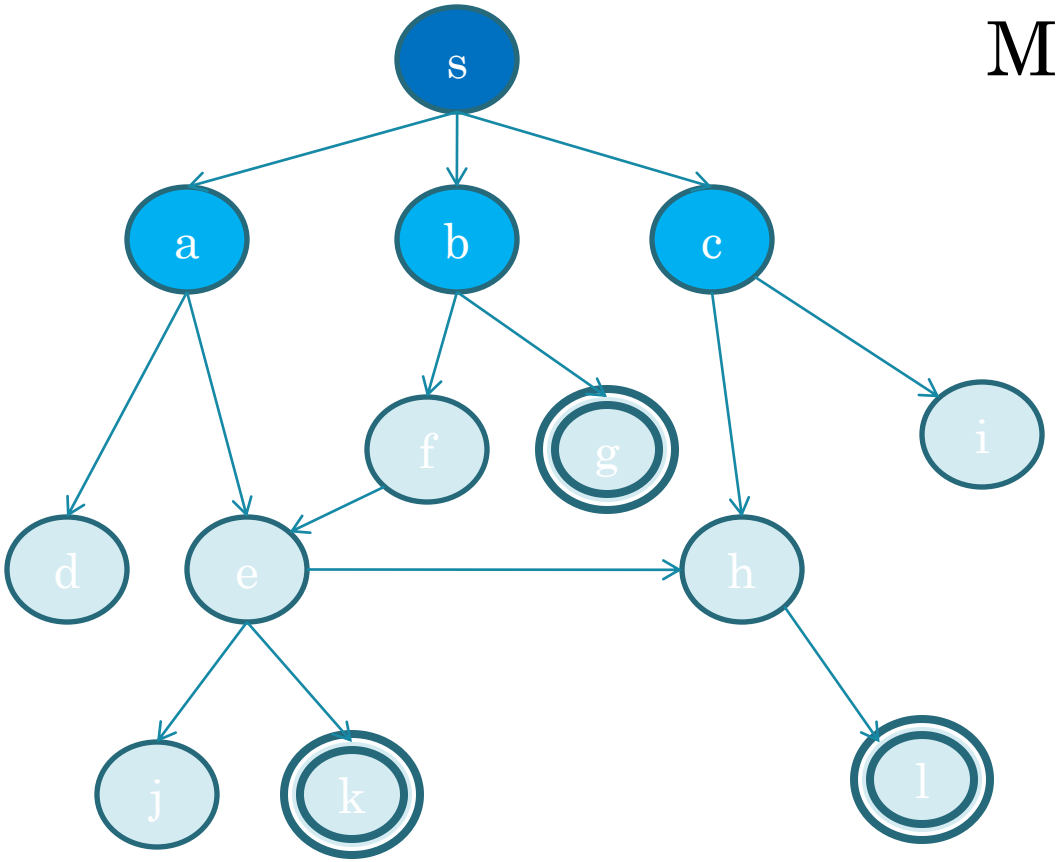


PRIMER – ITERATIVNO POGLABLJANJE (9/17)

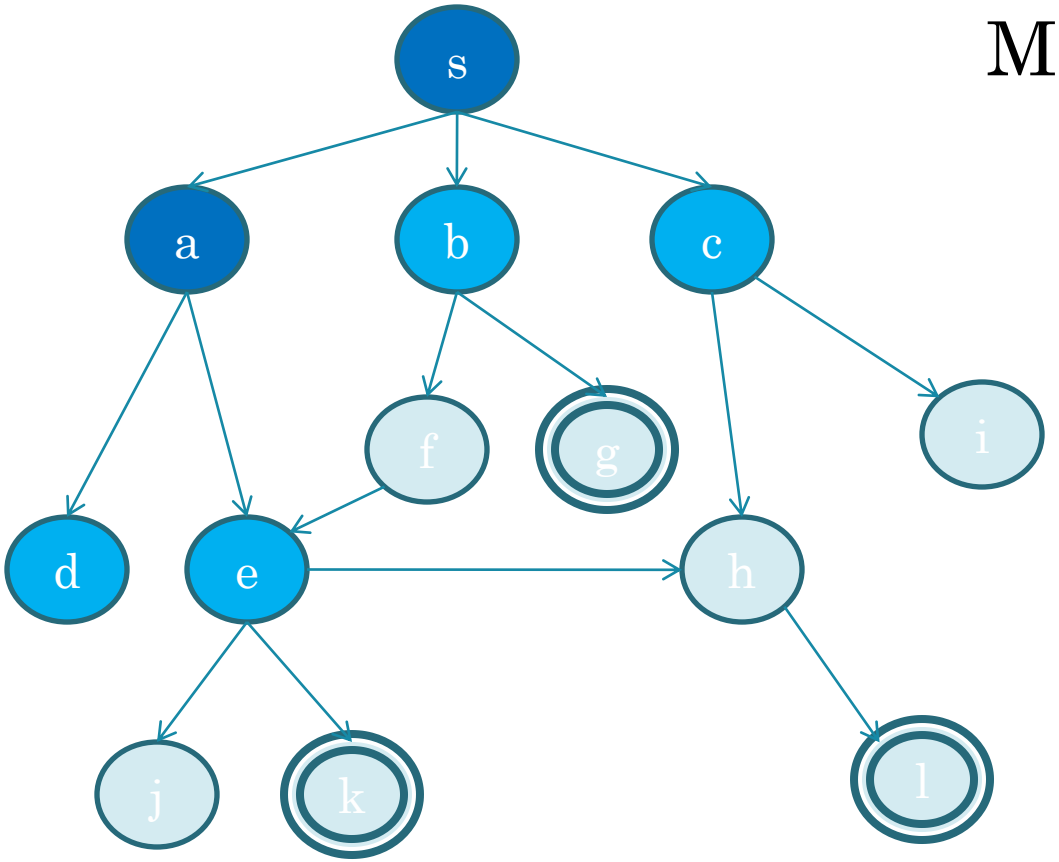


Meja 2

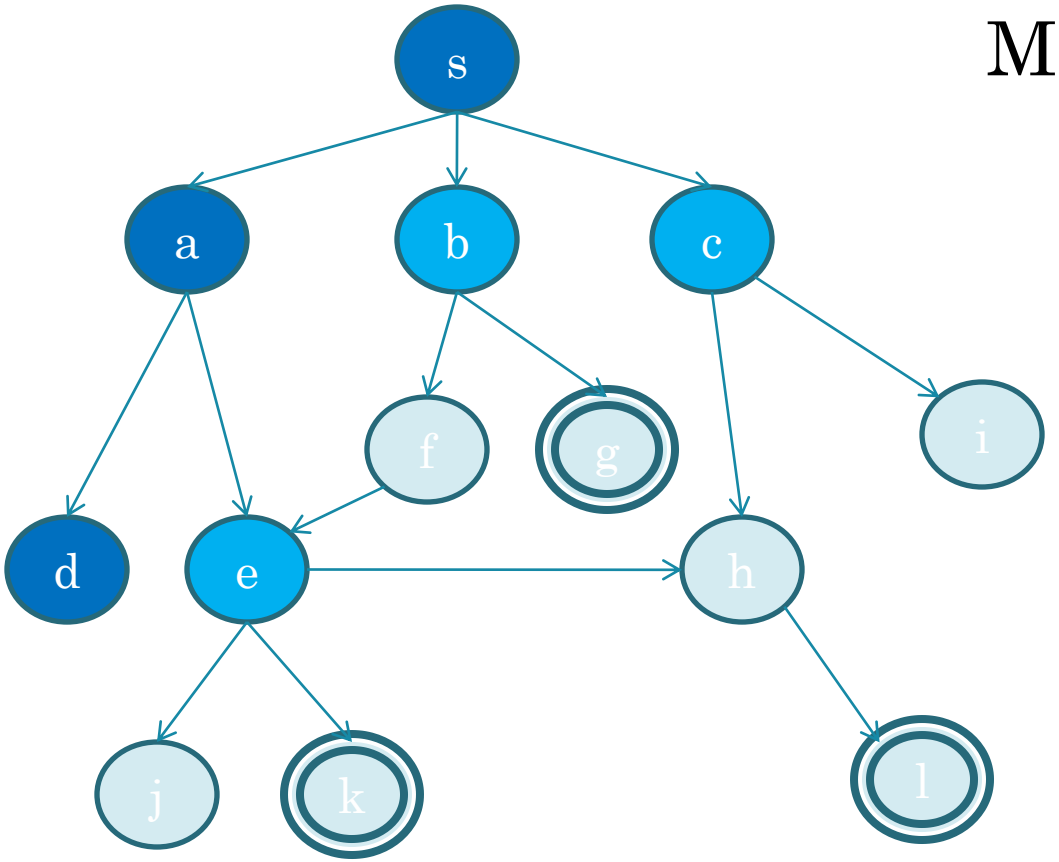
PRIMER – ITERATIVNO POGLABLJANJE (10/17)



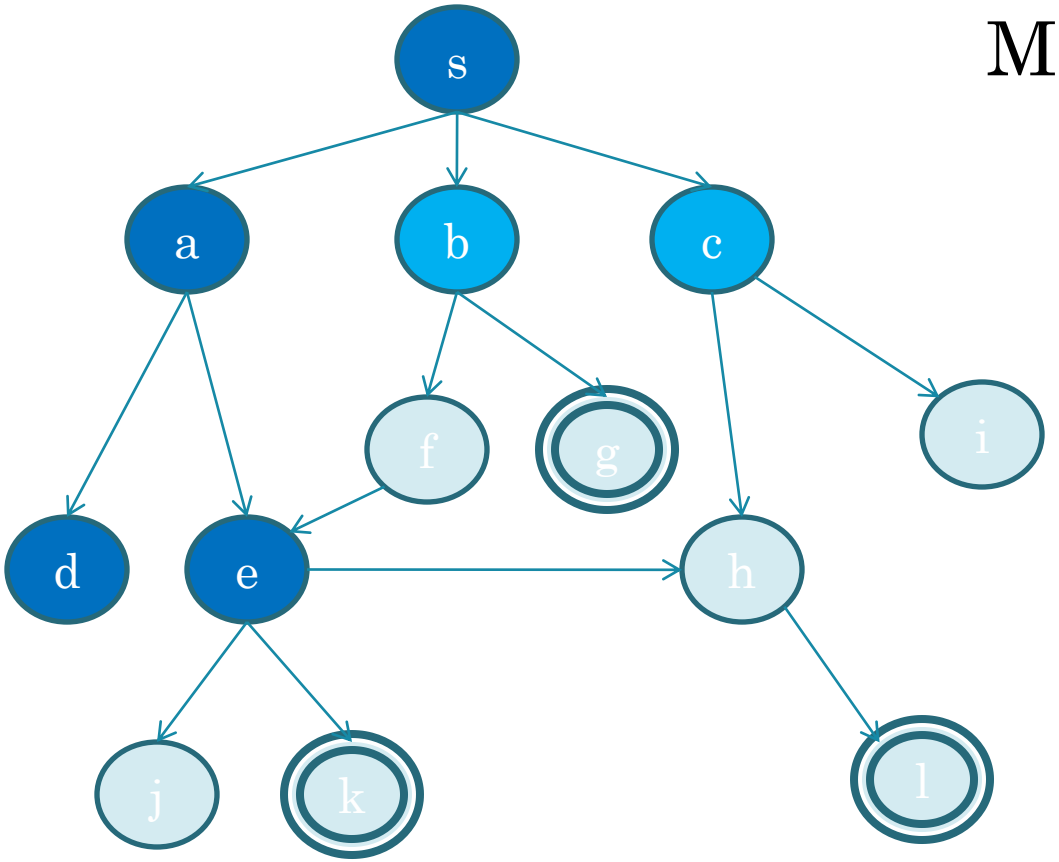
PRIMER – ITERATIVNO POGLABLJANJE (11/17)



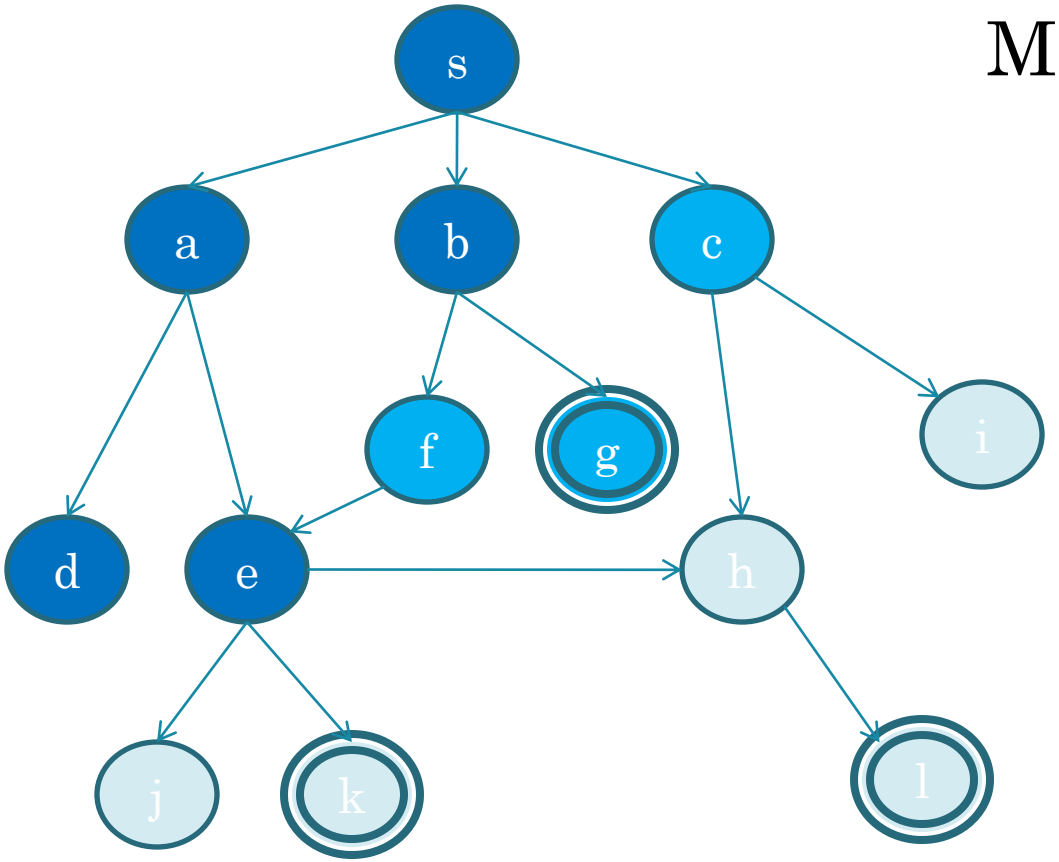
PRIMER – ITERATIVNO POGLABLJANJE (12/17)



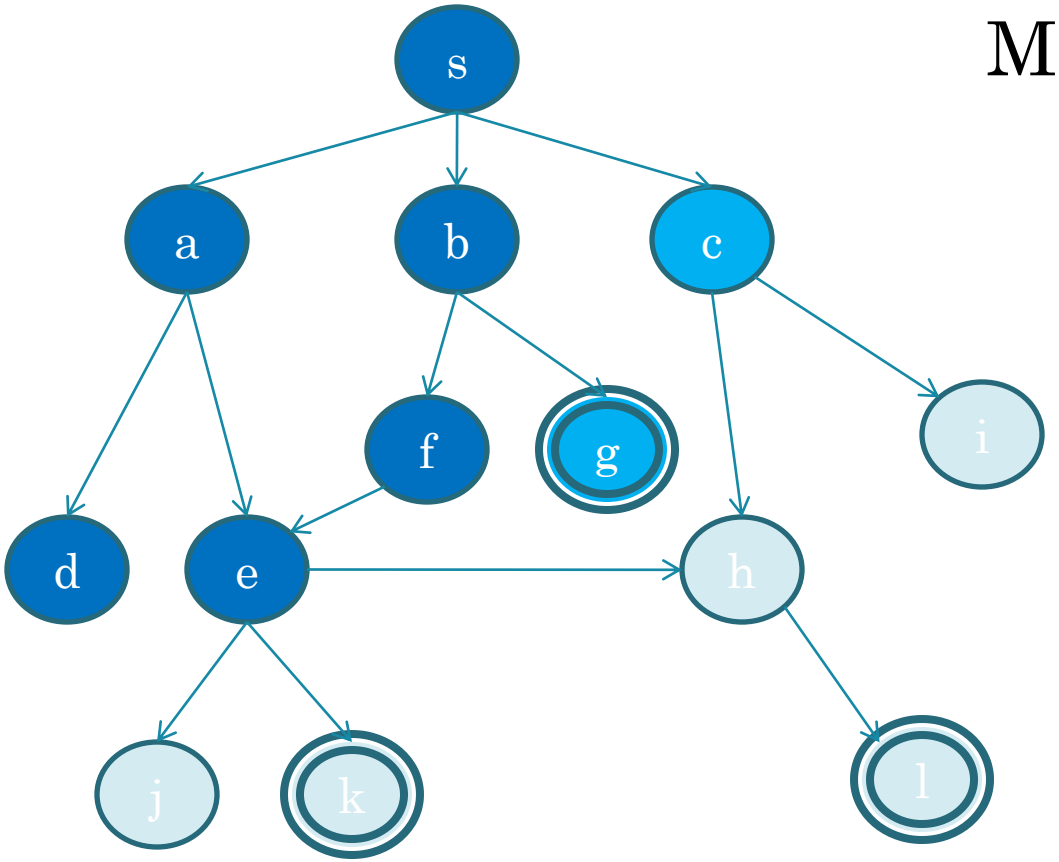
PRIMER – ITERATIVNO POGLABLJANJE (13/17)



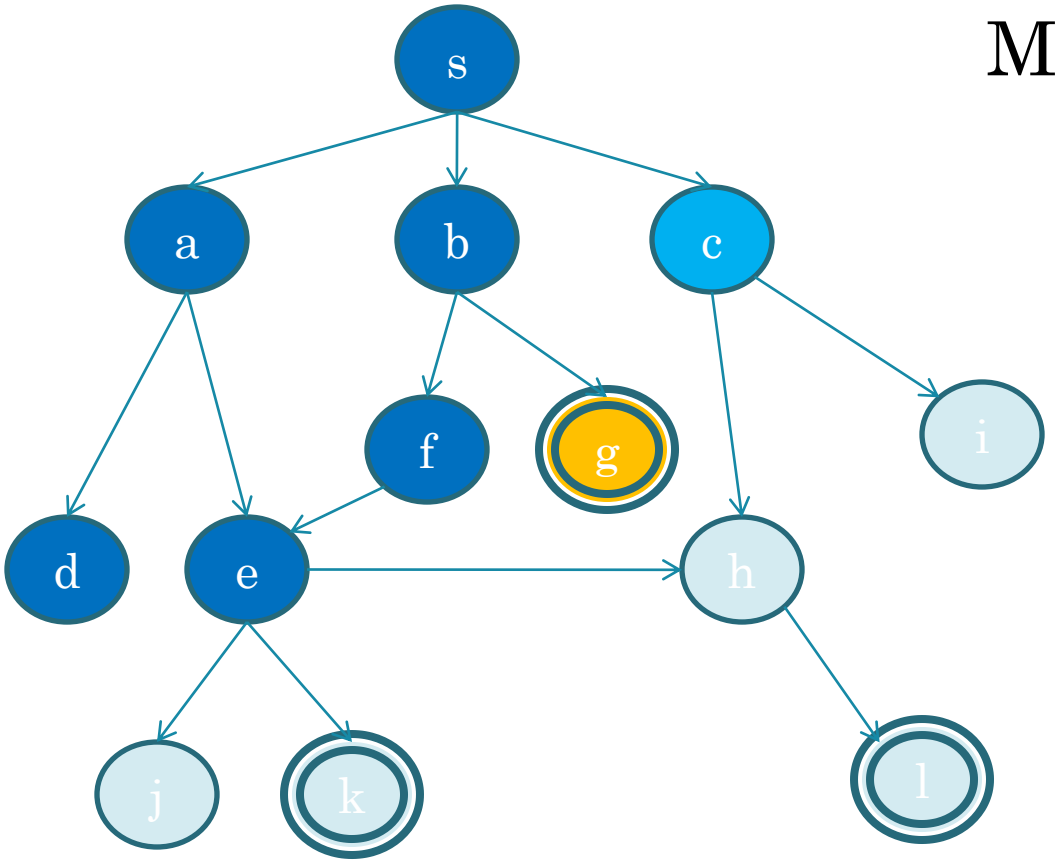
PRIMER – ITERATIVNO POGLABLJANJE (14/17)



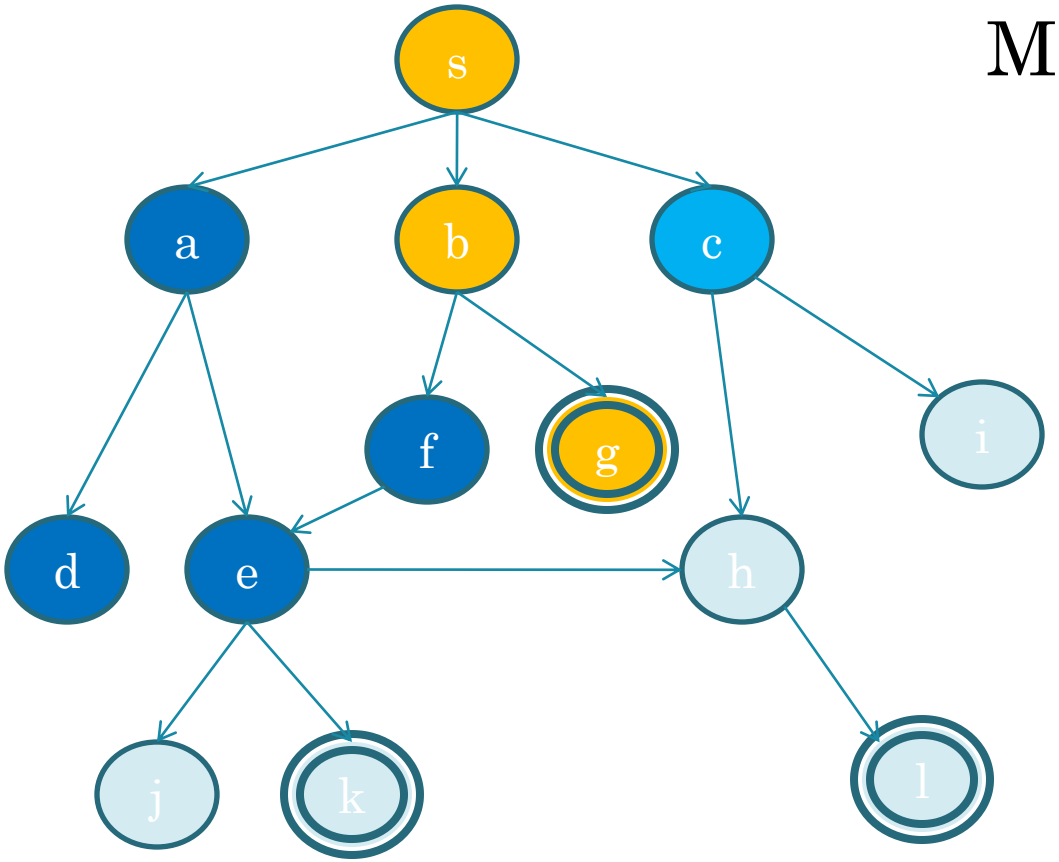
PRIMER – ITERATIVNO POGLABLJANJE (15/17)



PRIMER – ITERATIVNO POGLABLJANJE (16/17)



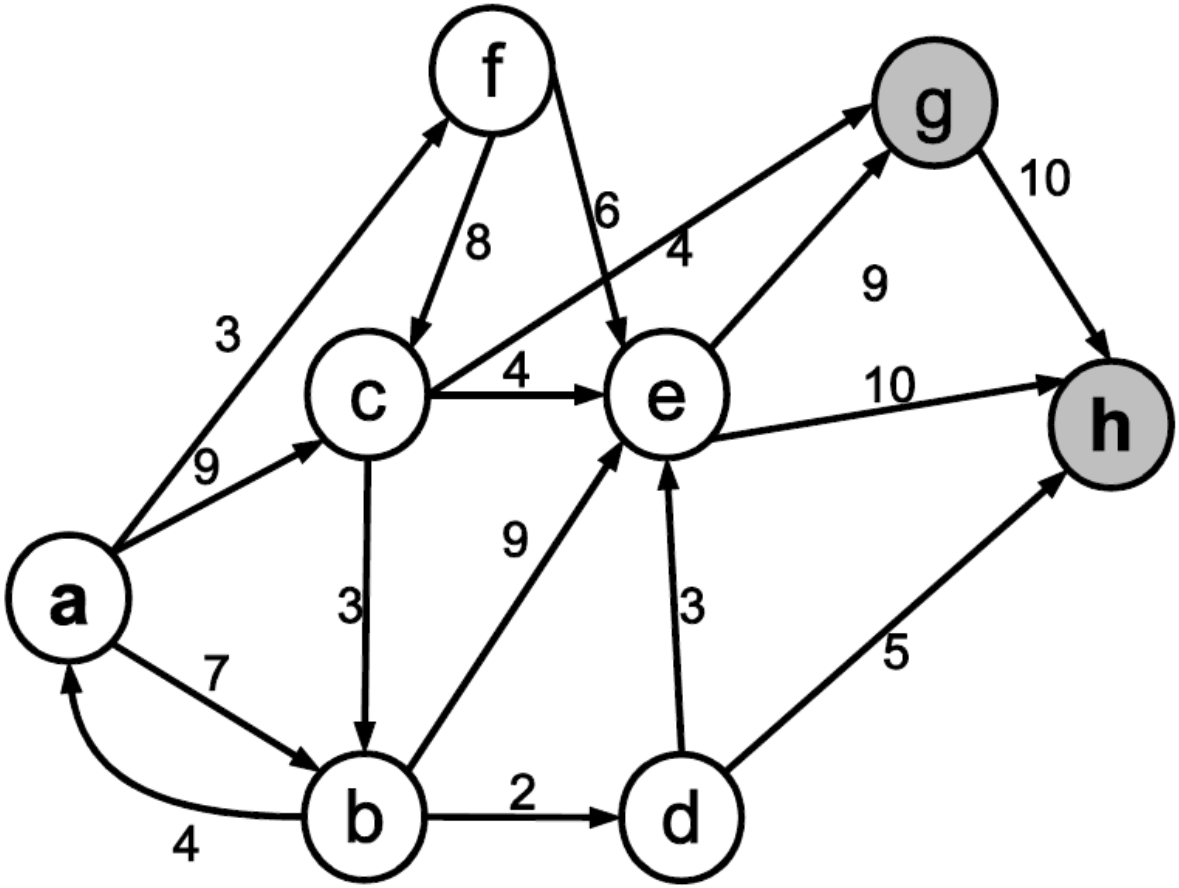
PRIMER – ITERATIVNO POGLABLJANJE (17/17)



Rešitev: s, b, g

Samostojno delo

Za spodnji graf z začetnim vozliščem *a* in s končnima vozliščema *g* in *h* določi zaporedje razvijanja vozlišč, če uporabljamo iterativno poglabljanje.



HEVRISTIČNO PREISKOVANJE

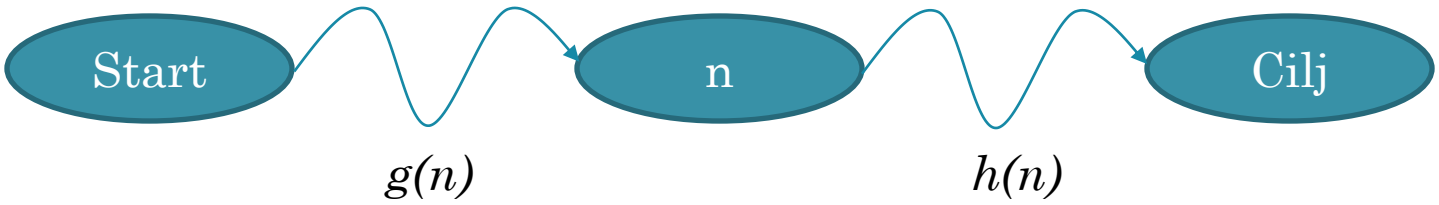
Če je prostor stanj velik, nam grozi kombinatorična eksplozija.

Moramo se zadovoljiti s preiskovanjem skromne podmnožice celotnega prostora stanj.

Uporabljamo hevristične ocene za omejevanje in usmerjanje iskanja v smeri najbolj obetavnega vozlišča.

Kot hevristična cenilka služi funkcija *f(n)*, ki ocenjuje "težavnost" vozlišča *n*.

HEVRISTIČNO PREISKOVANJE



g(n) je cena najboljše poti od začetnega vozlišča do vozlišča *n*

h(n) je ocena cene optimalne poti od vozlišča *n* do končnega vozlišča

Požrešno usmerjeno iskanje: $f(n) = h(n)$
(Greedy best-first search)

A*: $f(n) = g(n) + h(n)$

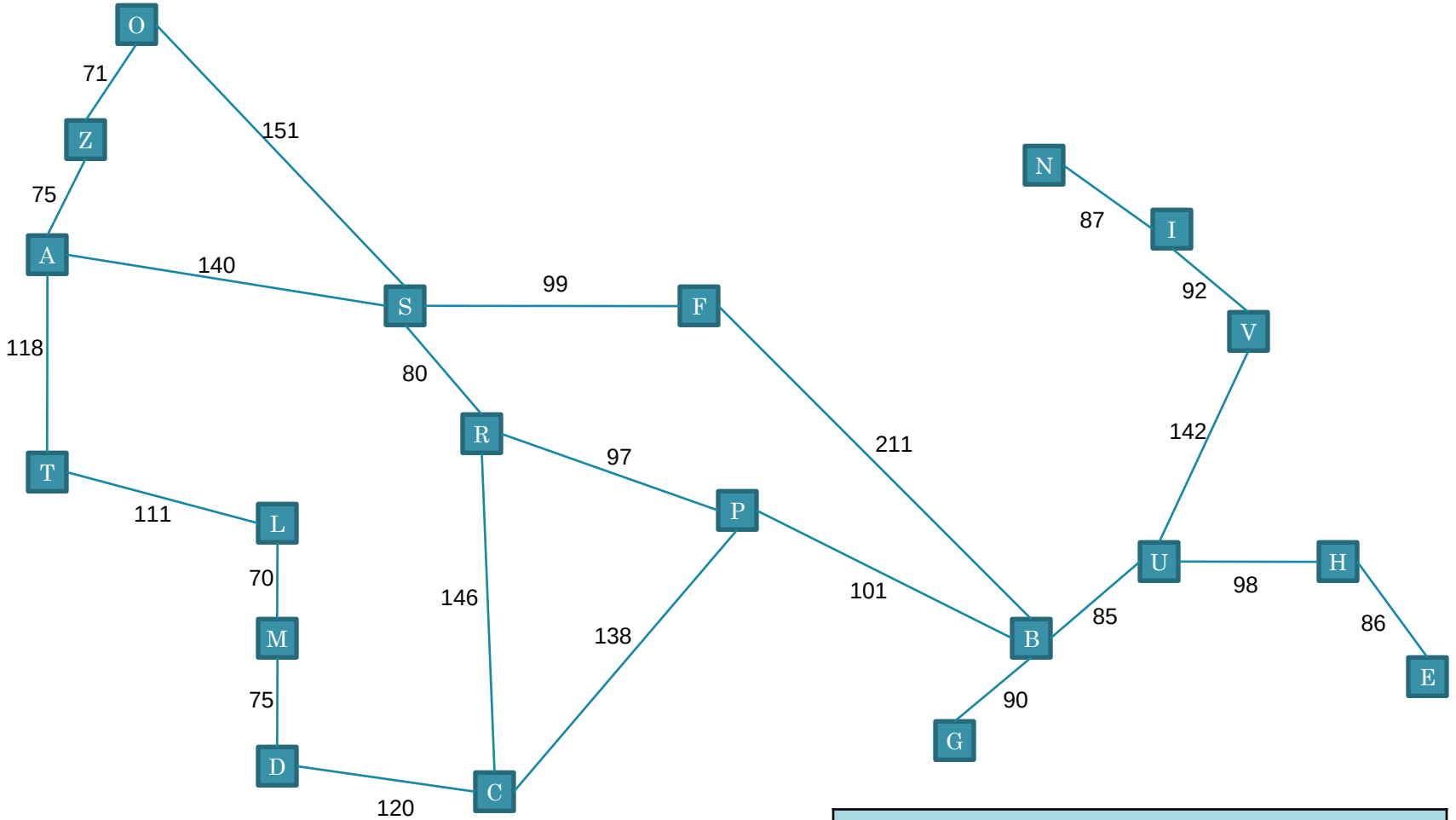
Dopustnost

- algoritem je dopusten (admissable), če vedno najde optimalno rešitev, če ta obstaja
- predpostavimo, da je za vsako vozlišče *n* v prostoru stanj $h^*(n)$ cena optimalne poti od *n* do najbližjega končnega vozlišča
- algoritem A*, ki uporablja hevristično funkcijo *h(n)*, tako da je za vsak *n*

$$h(n) \leq h^*(n)$$

  je dopusten.
- vsaka hevristična funkcija *h(n)*, ki ne precenjuje razdalje do cilja, je dopustna.
- trivialna, a neuporabna hevristična funkcija h(n) = 0 spremeni A* v iskanje v širino

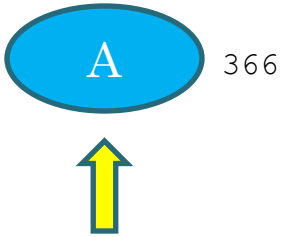
PRIMER - POŽREŠNO ISKANJE (1/7)



h(n) = Razdalja po zračni liniji do mesta B							
A	366	F	176	M	241	S	253
B	0	G	77	N	234	T	329
C	160	H	151	O	380	U	80
D	242	I	226	P	100	V	199
E	161	L	244	R	193	Z	374

PRIMER - POŽREŠNO ISKANJE (2/7)

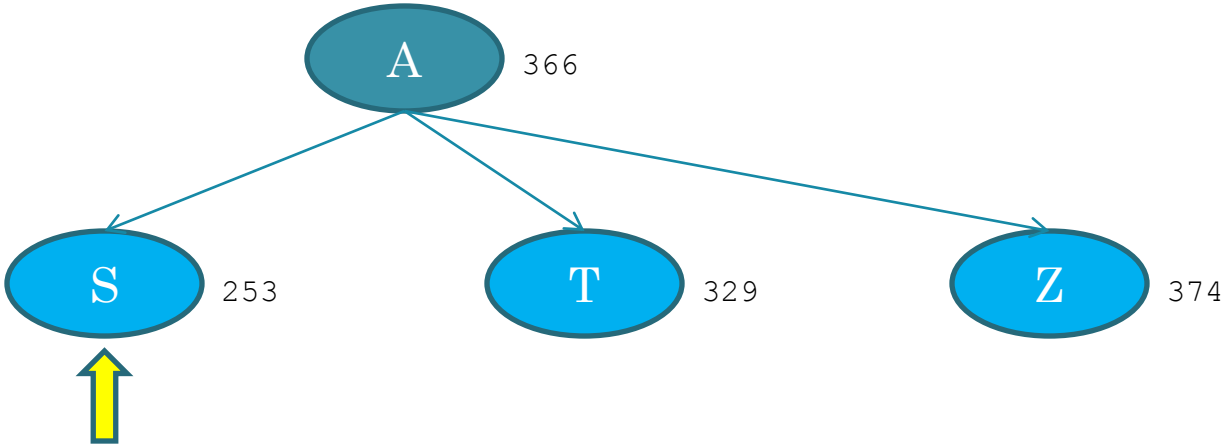
h(n) = Razdalja po zračni liniji do mesta B							
A	366	F	176	M	241	S	253
B	0	G	77	N	234	T	329
C	160	H	151	O	380	U	80
D	242	I	226	P	100	V	199
E	161	L	244	R	193	Z	374



366

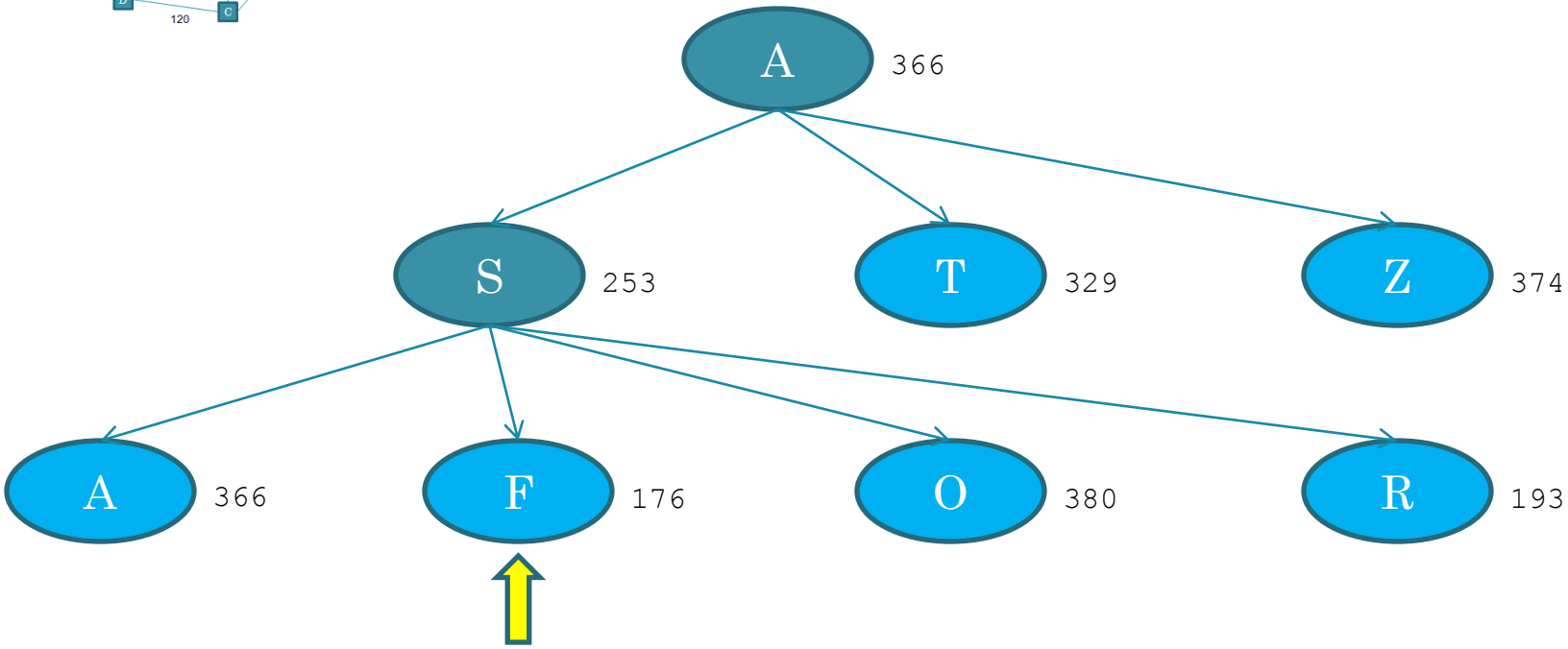
PRIMER - POŽREŠNO ISKANJE (3/7)

h(n) = Razdalja po zračni liniji do mesta B							
A	366	F	176	M	241	S	253
B	0	G	77	N	234	T	329
C	160	H	151	O	380	U	80
D	242	I	226	P	100	V	199
E	161	L	244	R	193	Z	374



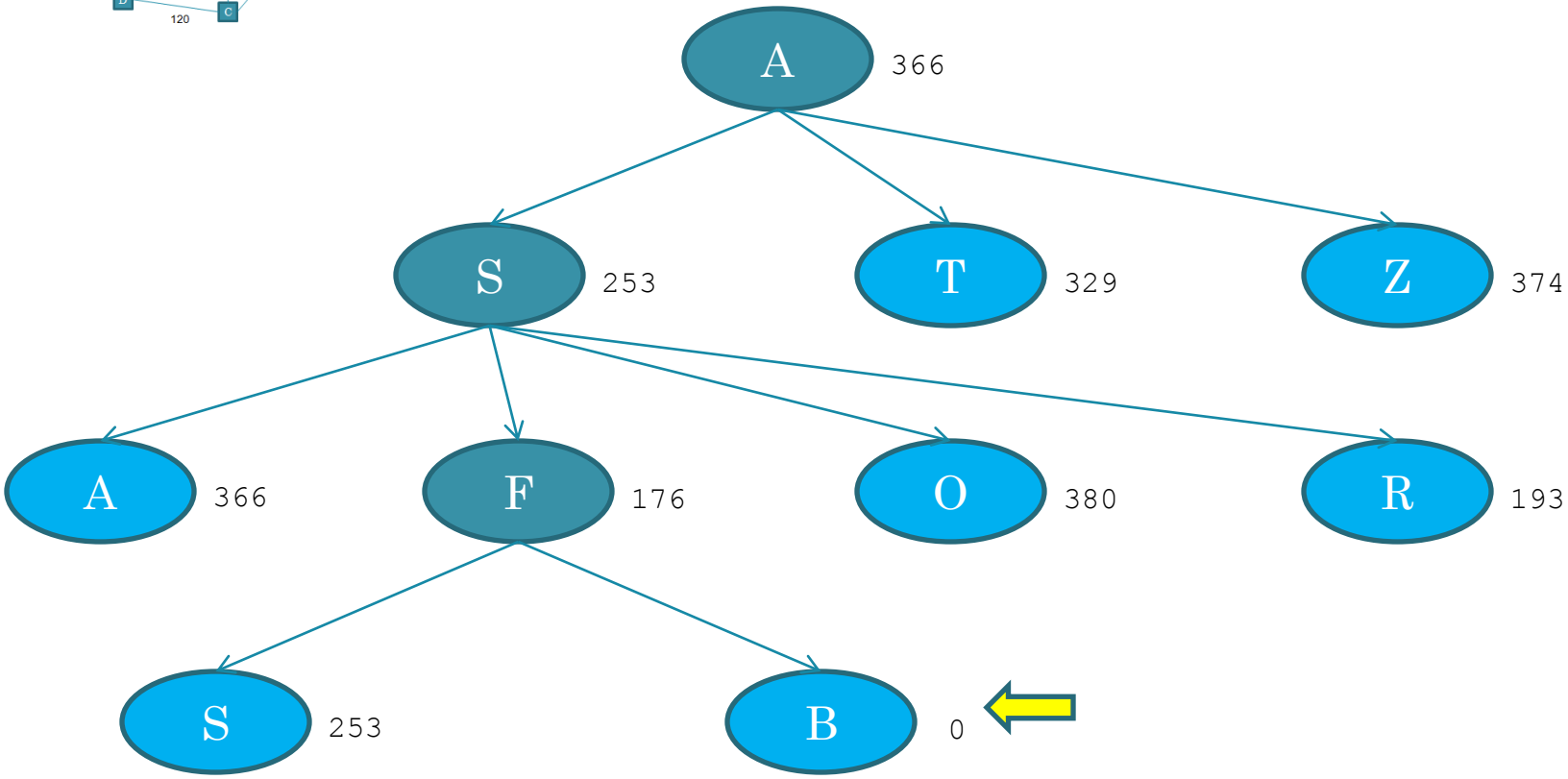
Primer - požrešno iskanje (4/7)

h(n) = Razdalja po zračni liniji do mesta B							
A	366	F	176	M	241	S	253
B	0	G	77	N	234	T	329
C	160	H	151	O	380	U	80
D	242	I	226	P	100	V	199
E	161	L	244	R	193	Z	374



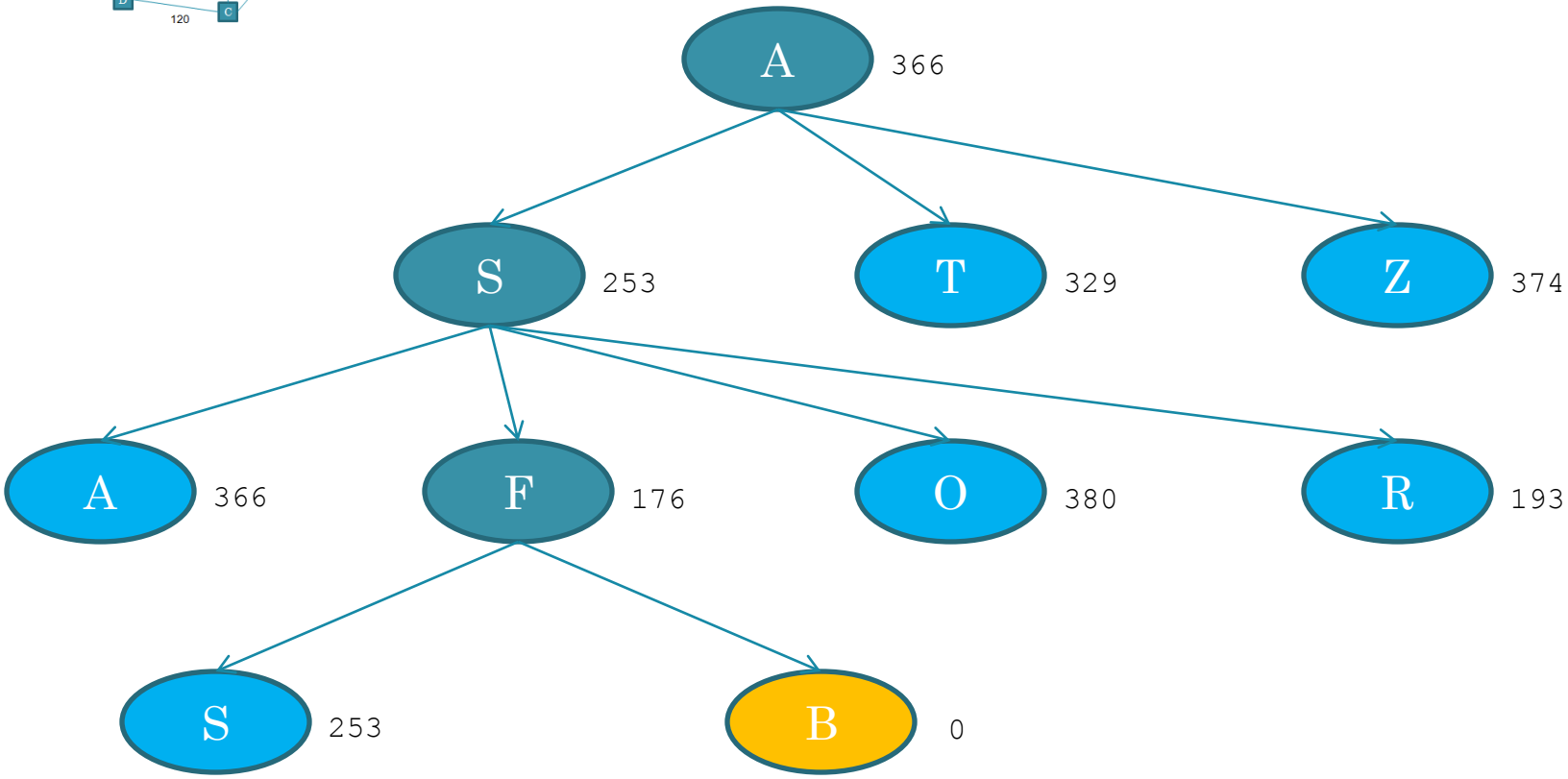
Primer - požrešno iskanje (5/7)

h(n) = Razdalja po zračni liniji do mesta B							
A	366	F	176	M	241	S	253
B	0	G	77	N	234	T	329
C	160	H	151	O	380	U	80
D	242	I	226	P	100	V	199
E	161	L	244	R	193	Z	374



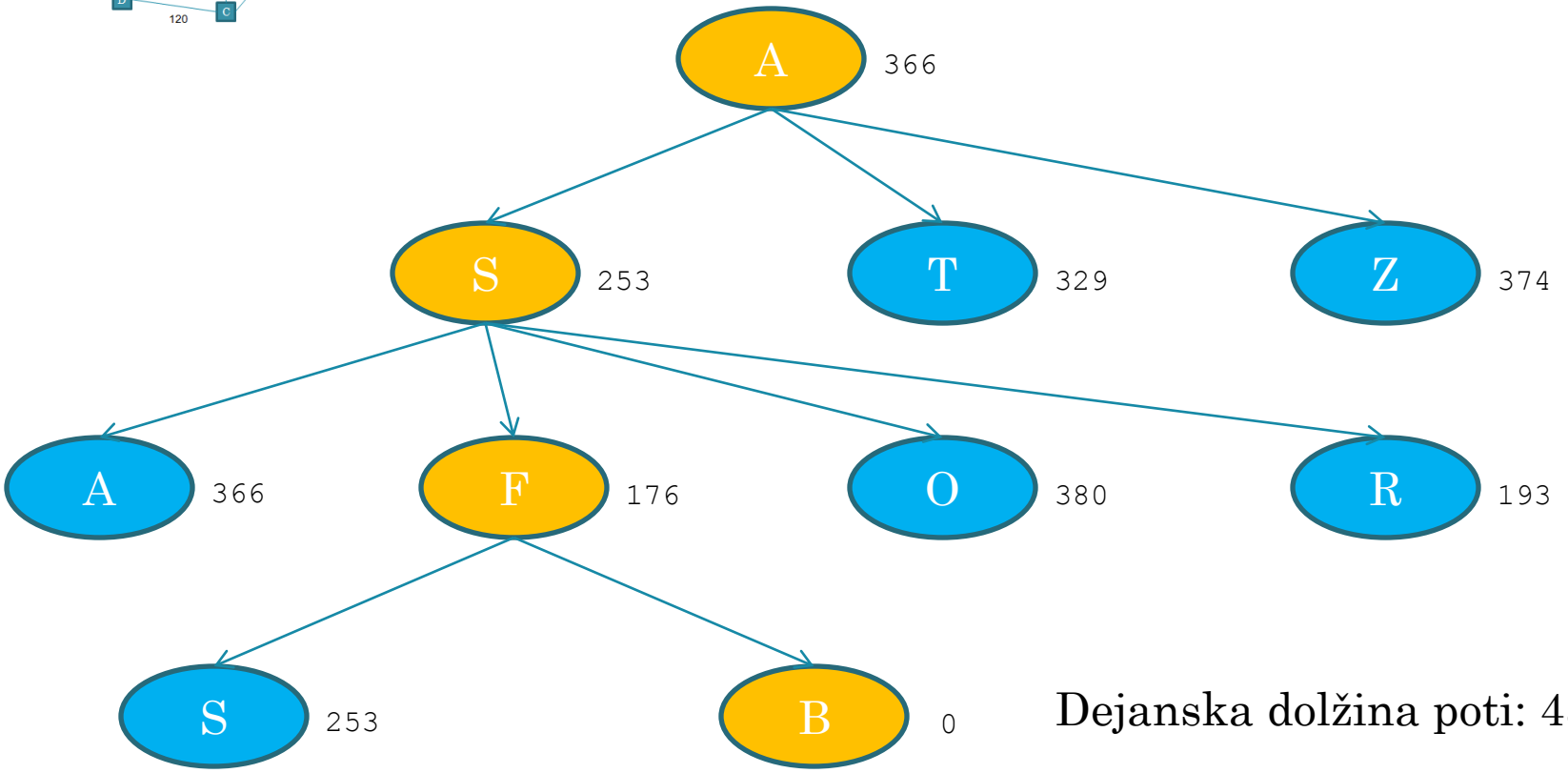
Primer - požrešno iskanje (6/7)

h(n) = Razdalja po zračni liniji do mesta B							
A	366	F	176	M	241	S	253
B	0	G	77	N	234	T	329
C	160	H	151	O	380	U	80
D	242	I	226	P	100	V	199
E	161	L	244	R	193	Z	374



Primer - požrešno iskanje (7/7)

h(n) = Razdalja po zračni liniji do mesta B							
A	366	F	176	M	241	S	253
B	0	G	77	N	234	T	329
C	160	H	151	O	380	U	80
D	242	I	226	P	100	V	199
E	161	L	244	R	193	Z	374

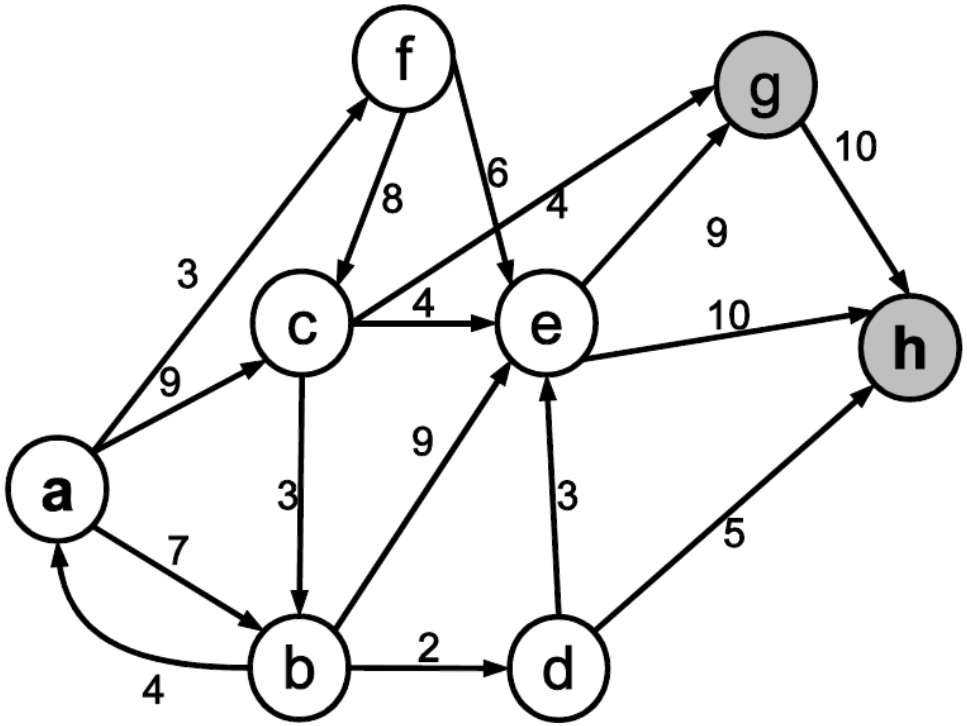


Dejanska dolžina poti: 450

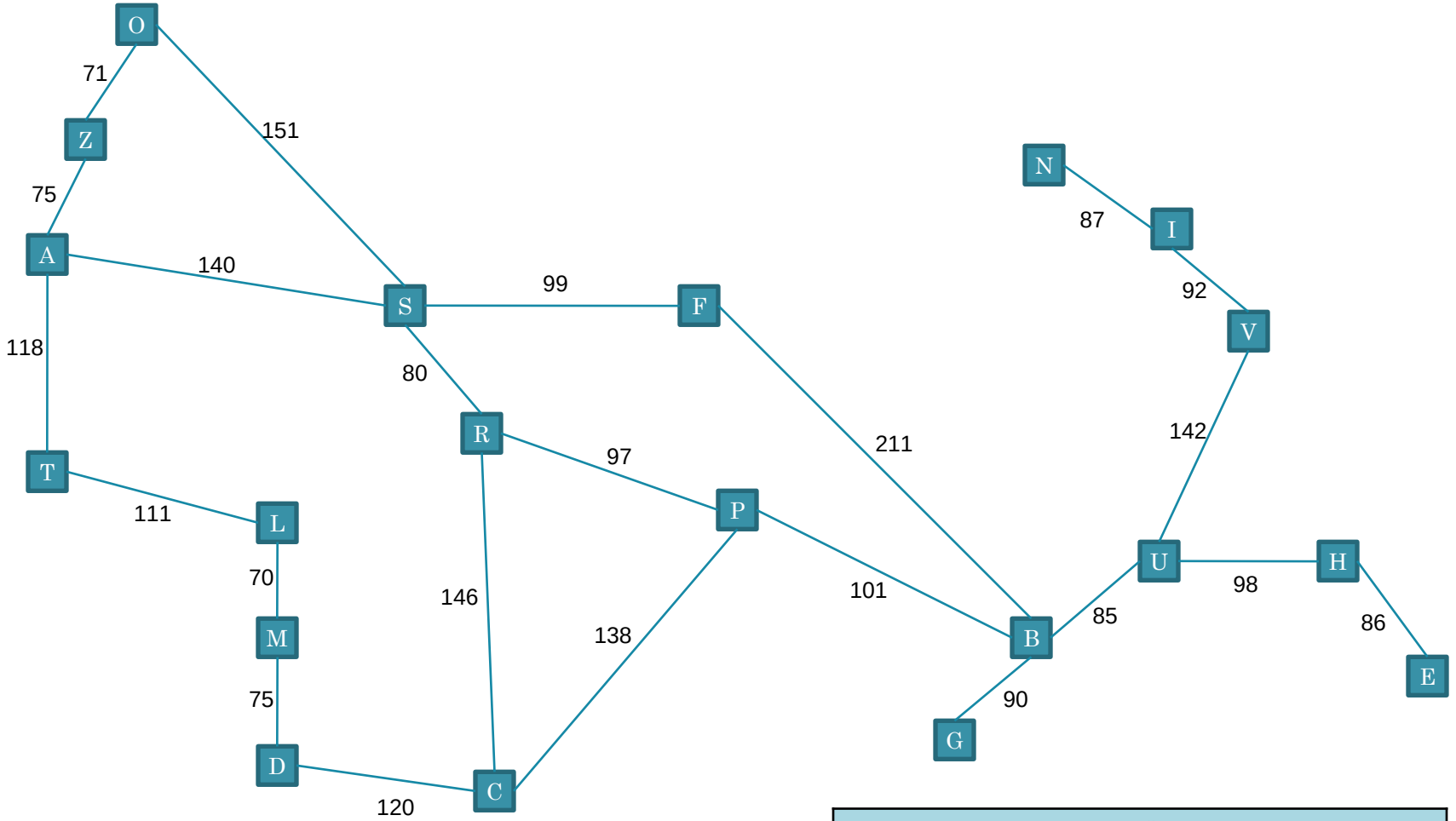
SAMOSTOJNO DELO

Za spodnji graf z začetnim vozliščem *a* in s končnima vozliščema *g* in *h* določi zaporedje razvijanja vozlišč, če uporabljamo požrešno iskanje s hevristično oceno iz spodnje tabele:

a	b	c	d	e	f	g	h
8	2	4	3	9	12	0	0

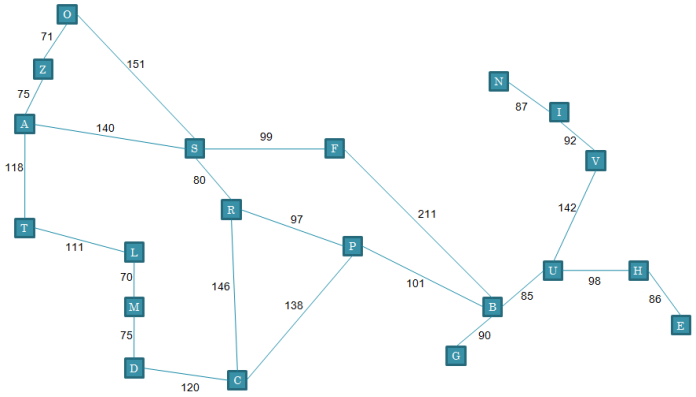


Primer A* (1/9)

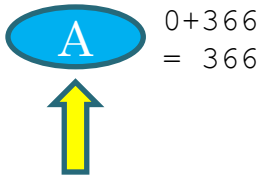


h(n) = Razdalja po zračni liniji do mesta B							
A	366	F	176	M	241	S	253
B	0	G	77	N	234	T	329
C	160	H	151	O	380	U	80
D	242	I	226	P	100	V	199
E	161	L	244	R	193	Z	374

Primer A* (2/9)

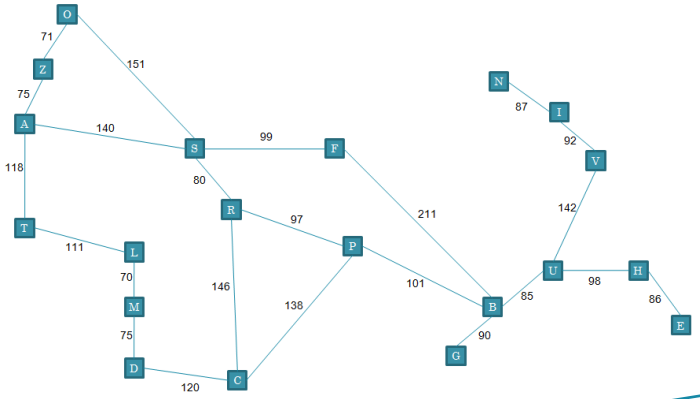


h(n) = Razdalja po zračni liniji do mesta B							
A	366	F	176	M	241	S	253
B	0	G	77	N	234	T	329
C	160	H	151	O	380	U	80
D	242	I	226	P	100	V	199
E	161	L	244	R	193	Z	374

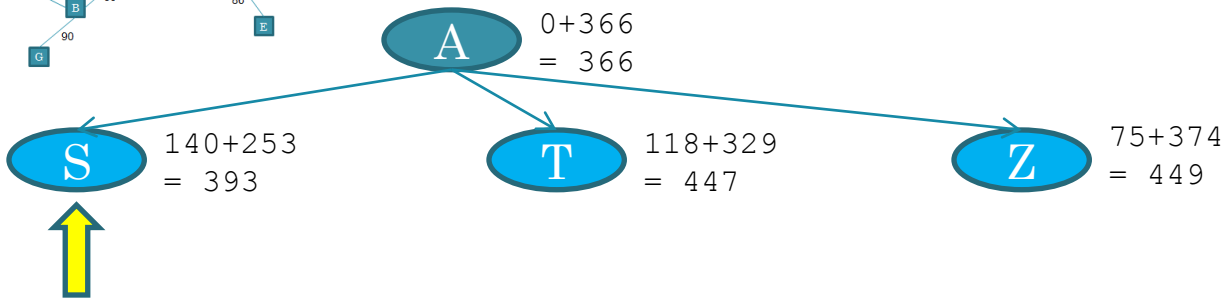


0+366
= 366

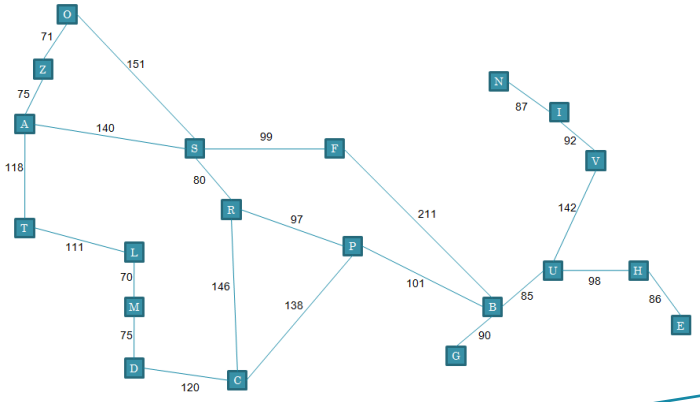
Primer A* (3/9)



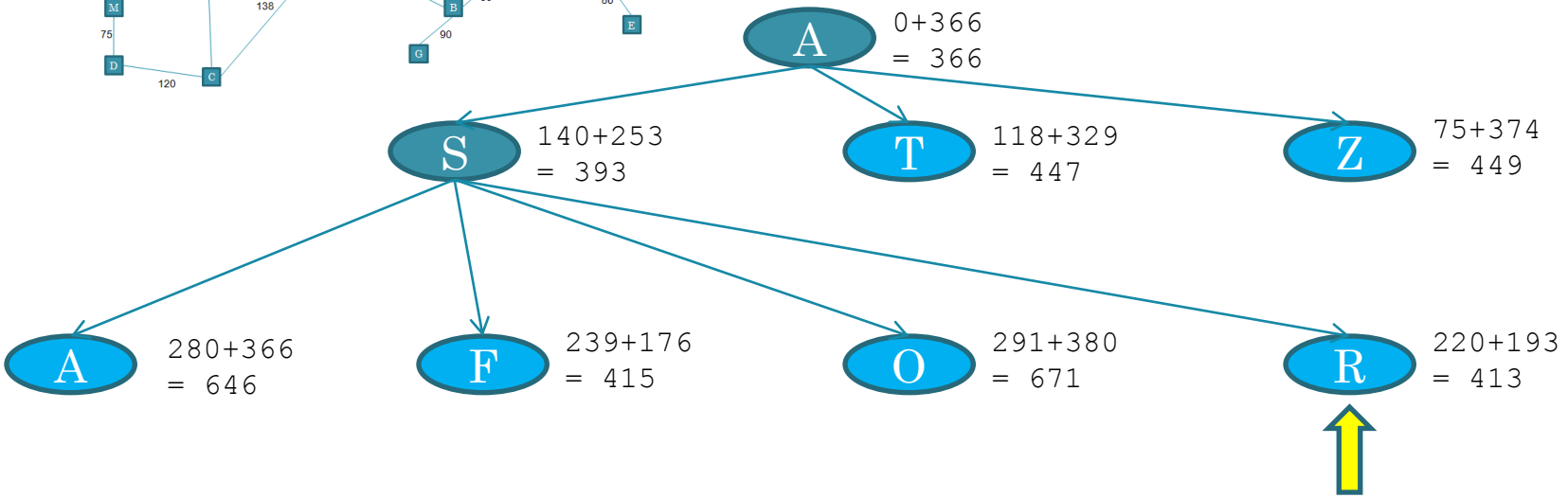
h(n) = Razdalja po zračni liniji do mesta B							
A	366	F	176	M	241	S	253
B	0	G	77	N	234	T	329
C	160	H	151	O	380	U	80
D	242	I	226	P	100	V	199
E	161	L	244	R	193	Z	374



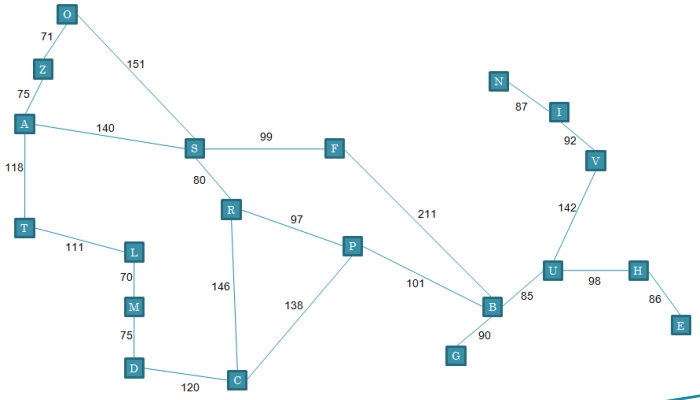
Primer A* (4/9)



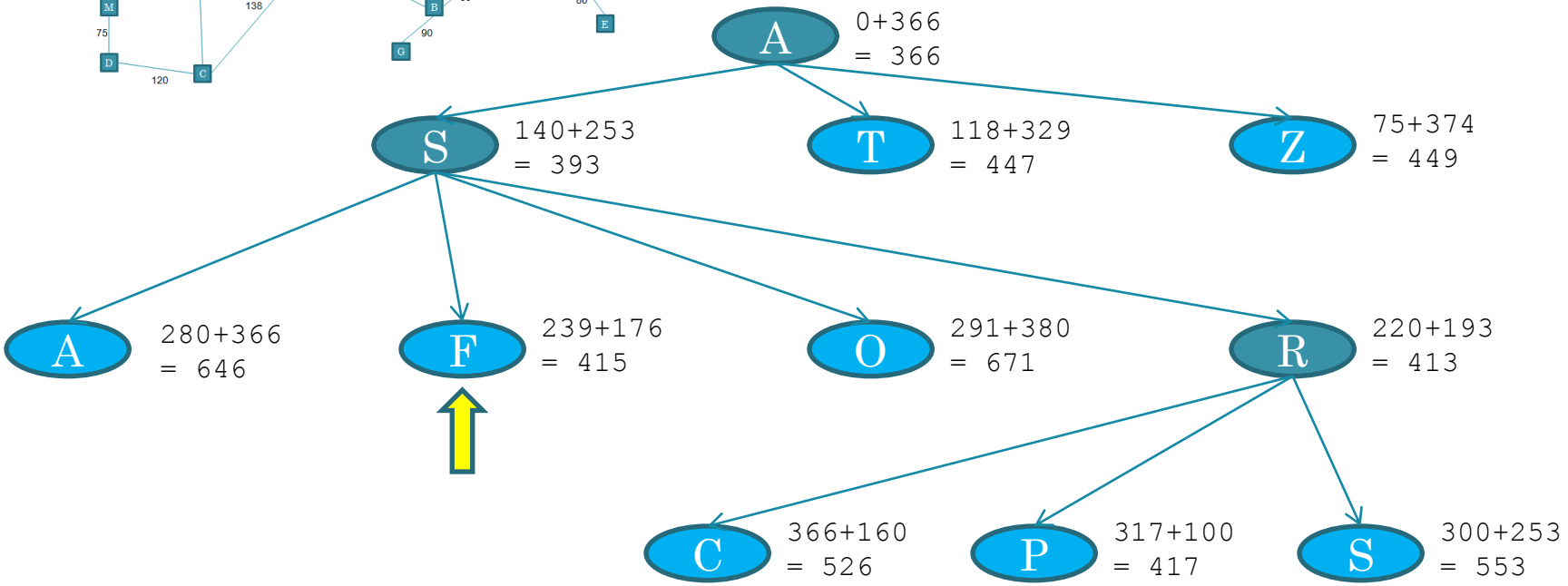
h(n) = Razdalja po zračni liniji do mesta B							
A	366	F	176	M	241	S	253
B	0	G	77	N	234	T	329
C	160	H	151	O	380	U	80
D	242	I	226	P	100	V	199
E	161	L	244	R	193	Z	374



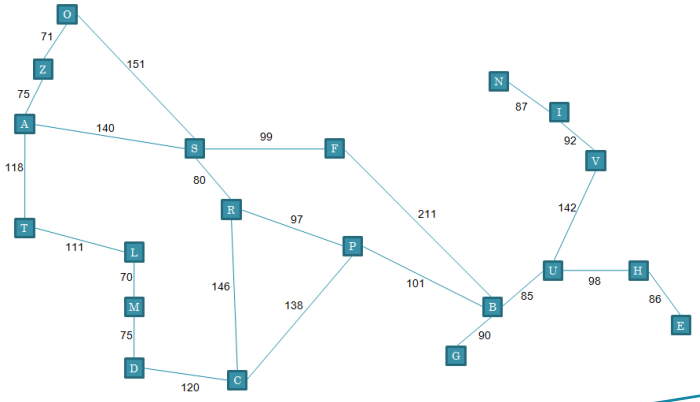
Primer A* (5/9)



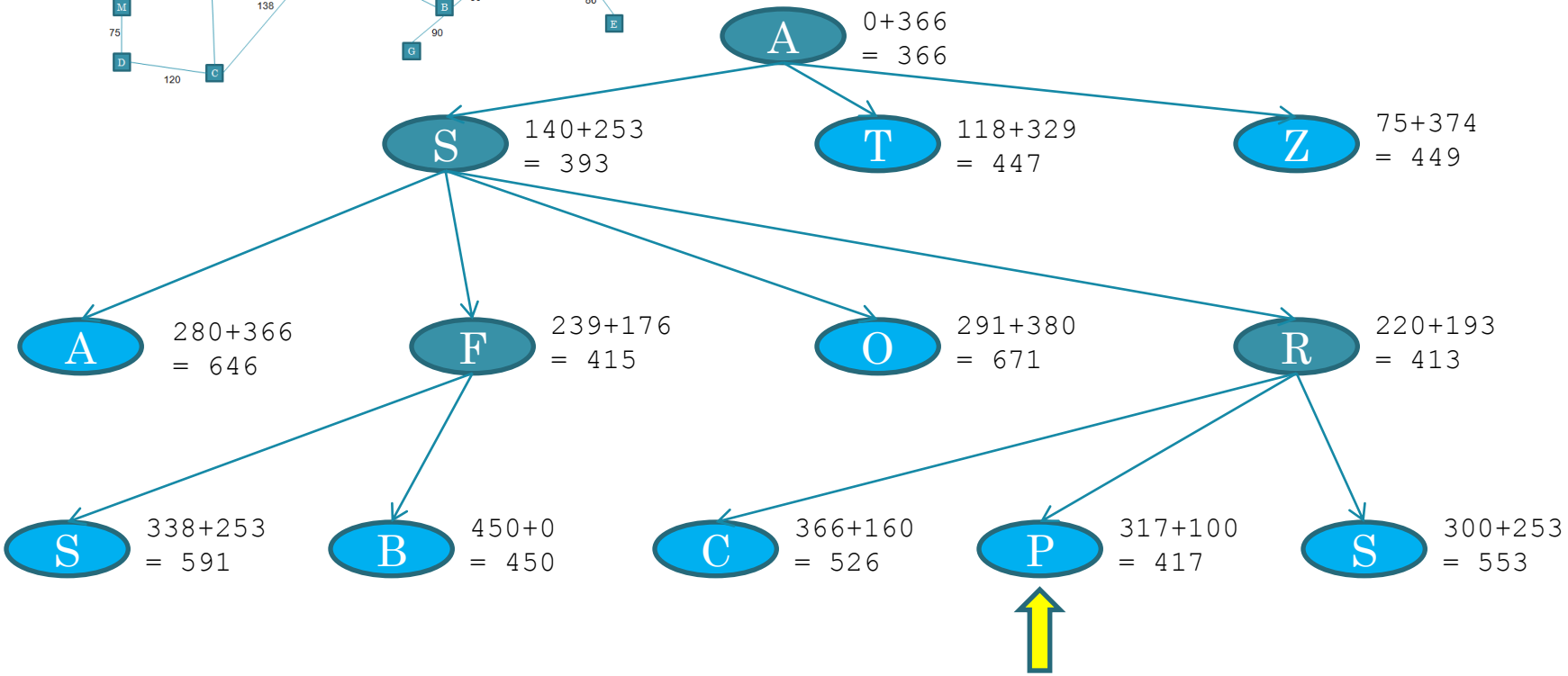
h(n) = Razdalja po zračni liniji do mesta B							
A	366	F	176	M	241	S	253
B	0	G	77	N	234	T	329
C	160	H	151	O	380	U	80
D	242	I	226	P	100	V	199
E	161	L	244	R	193	Z	374



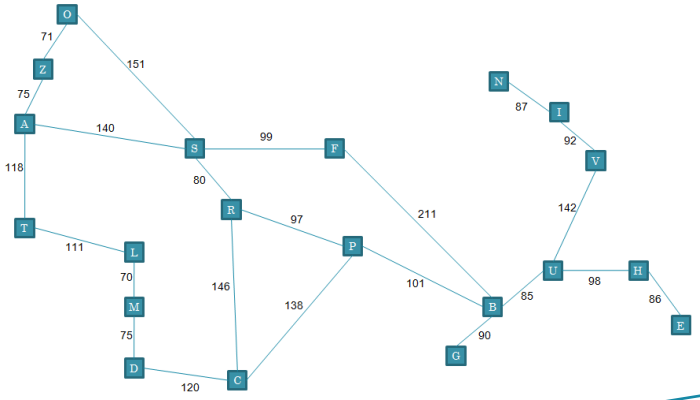
Primer A* (6/9)



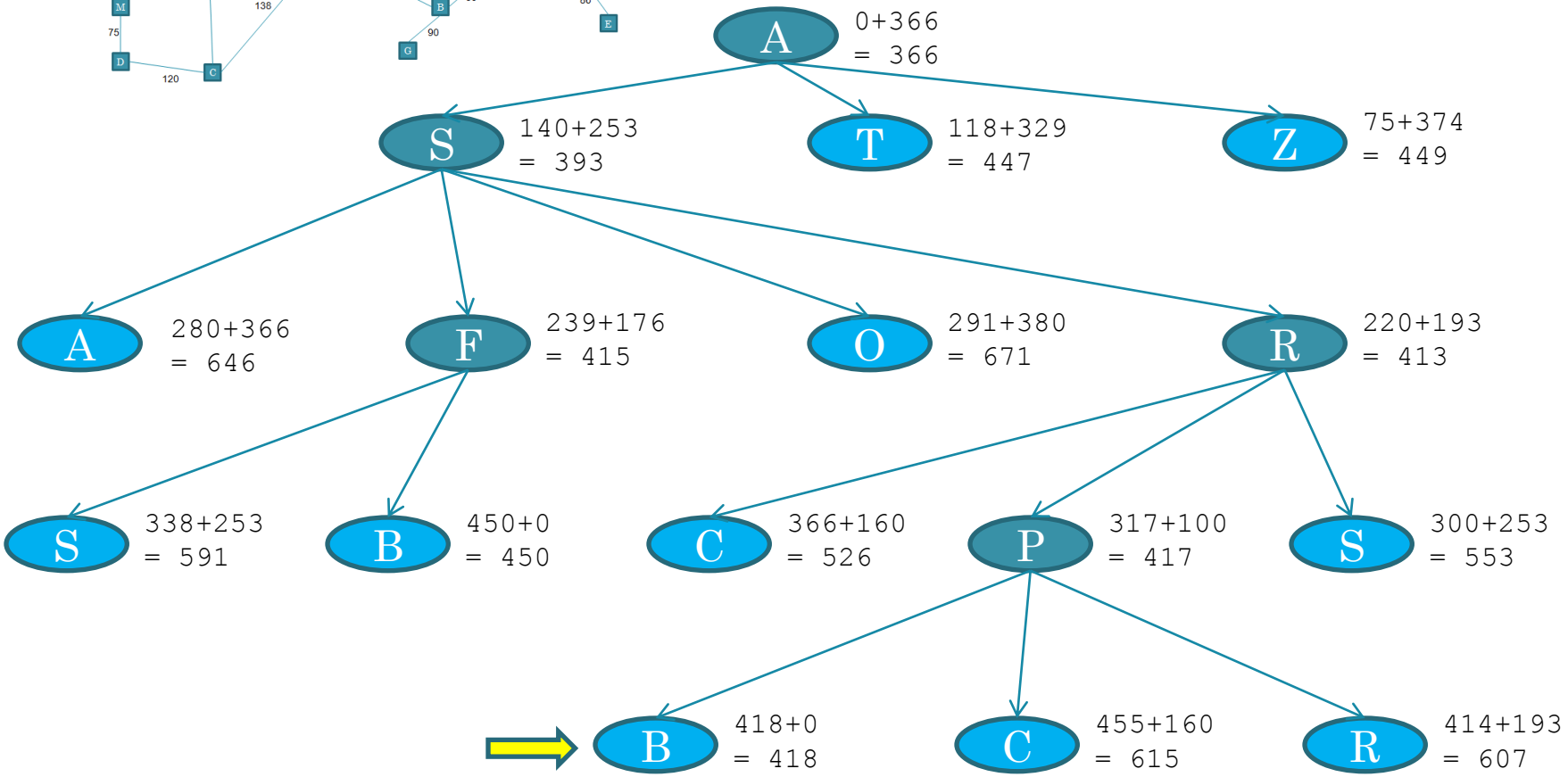
h(n) = Razdalja po zračni liniji do mesta B							
A	366	F	176	M	241	S	253
B	0	G	77	N	234	T	329
C	160	H	151	O	380	U	80
D	242	I	226	P	100	V	199
E	161	L	244	R	193	Z	374



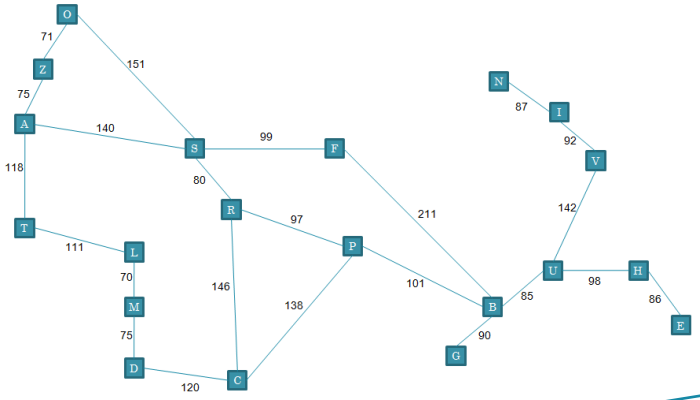
Primer A* (7/9)



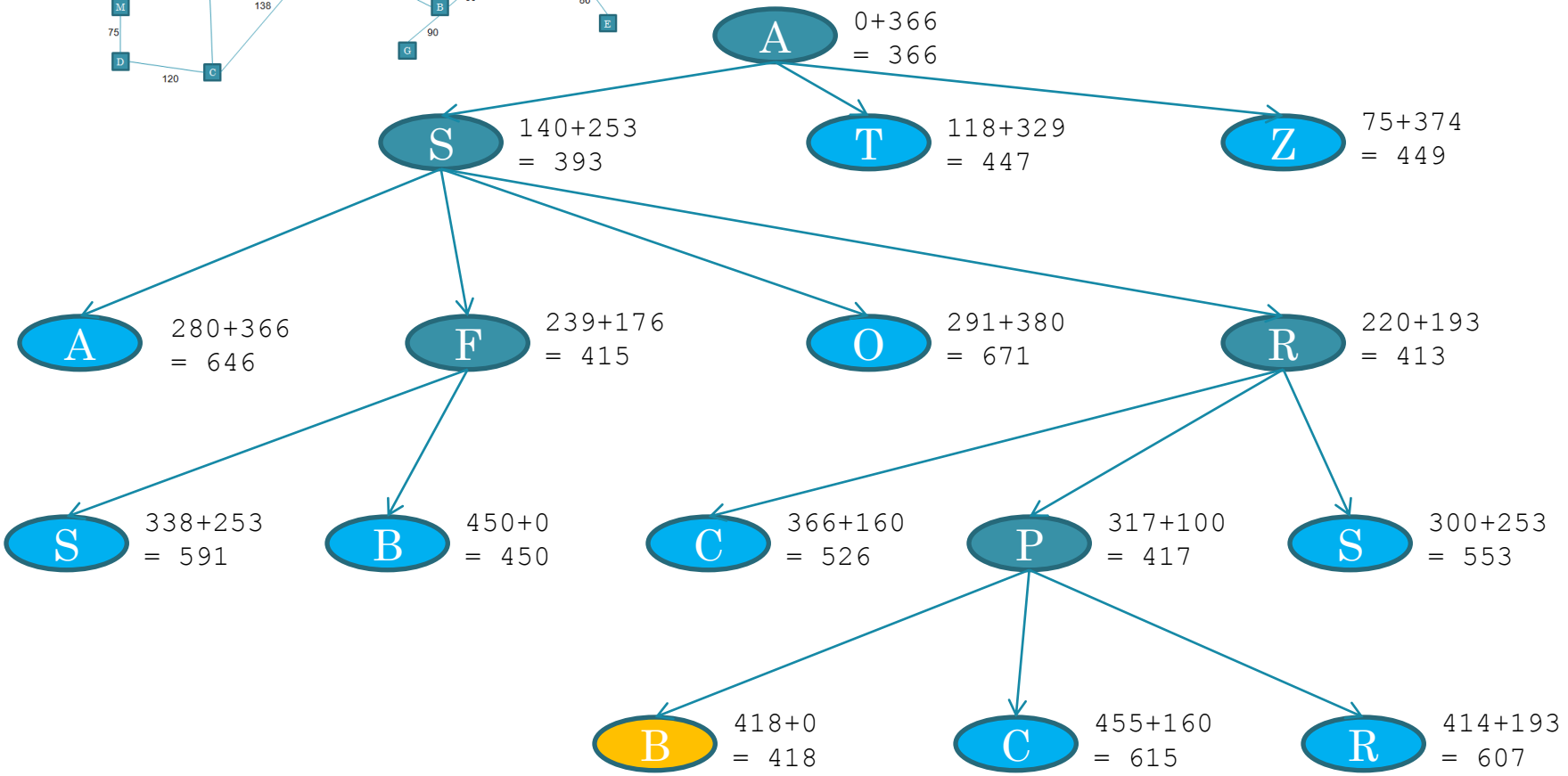
h(n) = Razdalja po zračni liniji do mesta B							
A	366	F	176	M	241	S	253
B	0	G	77	N	234	T	329
C	160	H	151	O	380	U	80
D	242	I	226	P	100	V	199
E	161	L	244	R	193	Z	374



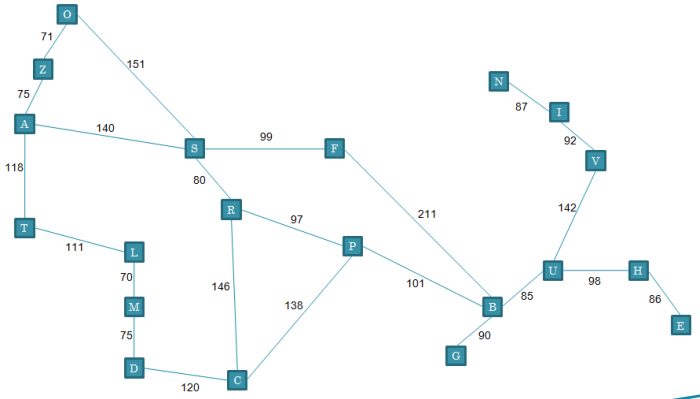
Primer A* (8/9)



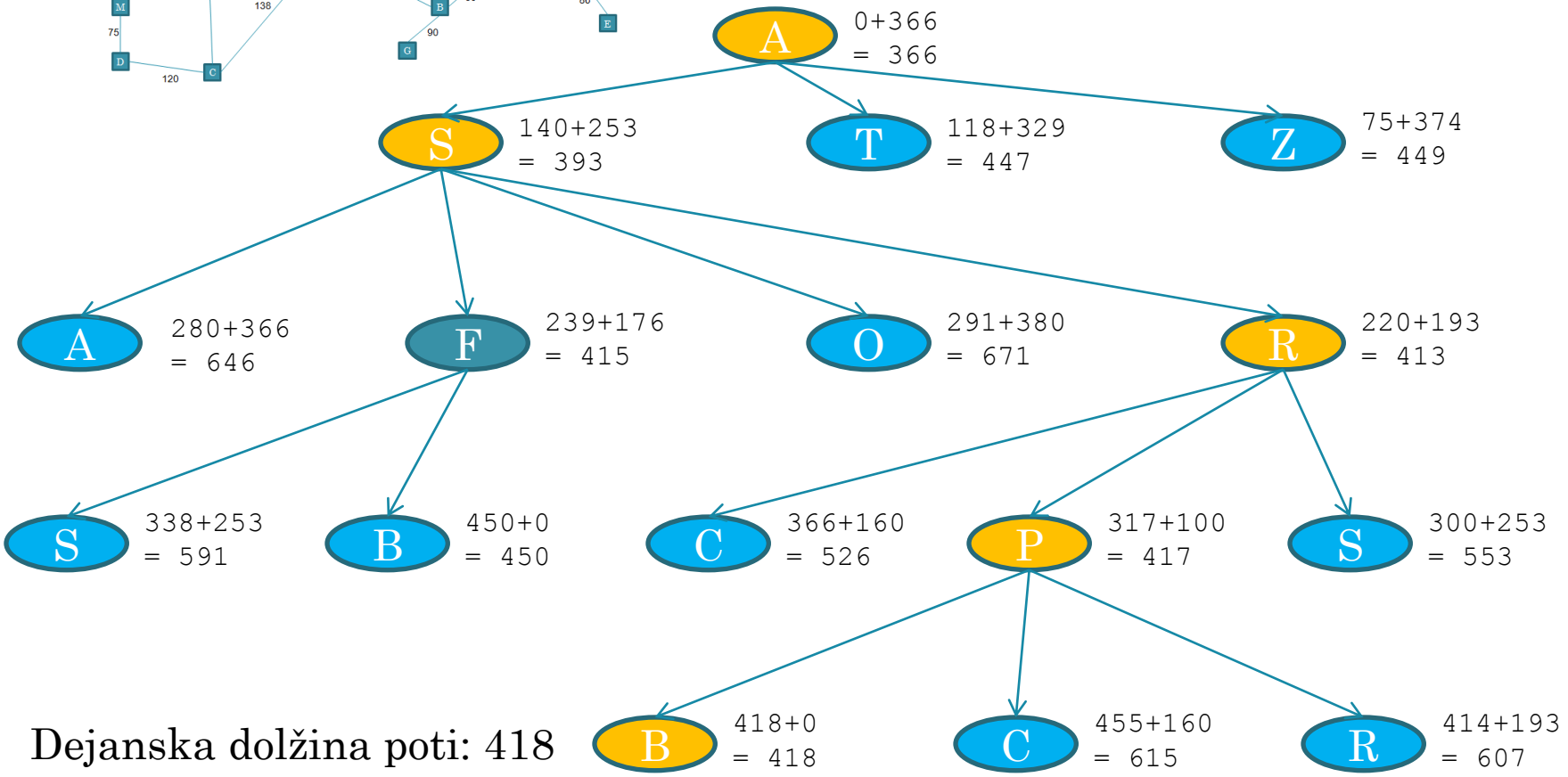
h(n) = Razdalja po zračni liniji do mesta B							
A	366	F	176	M	241	S	253
B	0	G	77	N	234	T	329
C	160	H	151	O	380	U	80
D	242	I	226	P	100	V	199
E	161	L	244	R	193	Z	374



Primer A* (9/9)

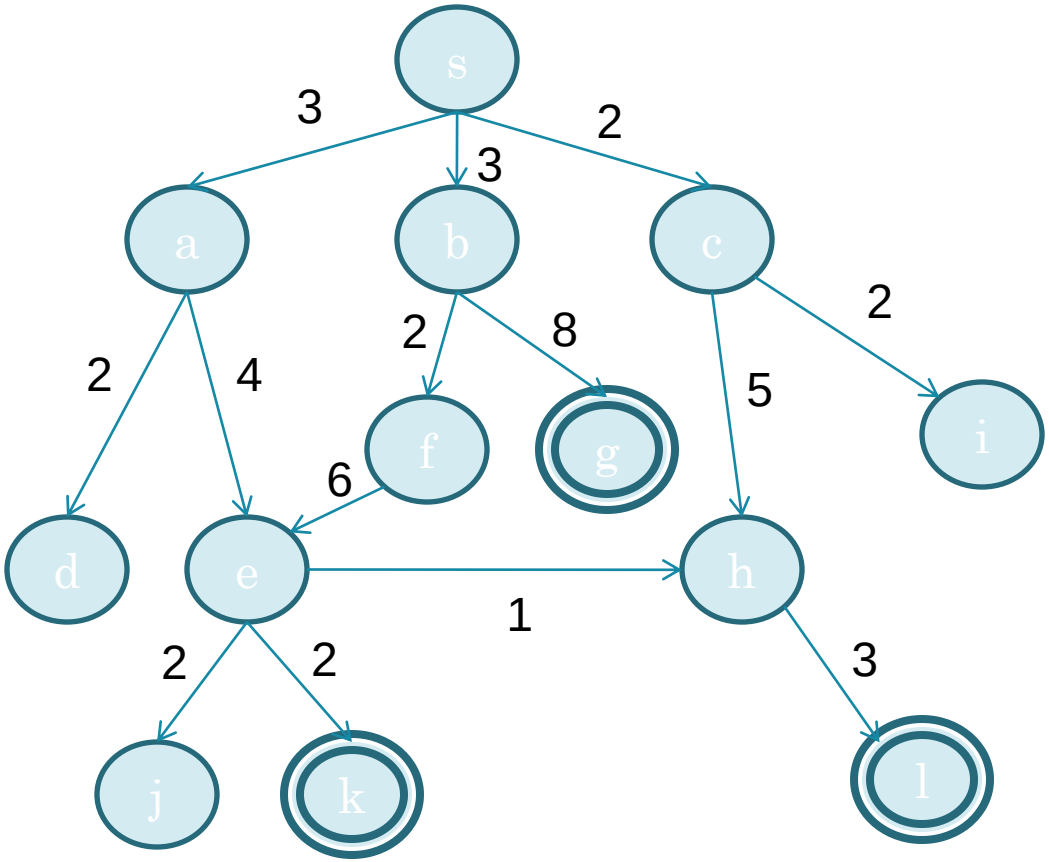


h(n) = Razdalja po zračni liniji do mesta B							
A	366	F	176	M	241	S	253
B	0	G	77	N	234	T	329
C	160	H	151	O	380	U	80
D	242	I	226	P	100	V	199
E	161	L	244	R	193	Z	374



Dejanska dolžina poti: 418

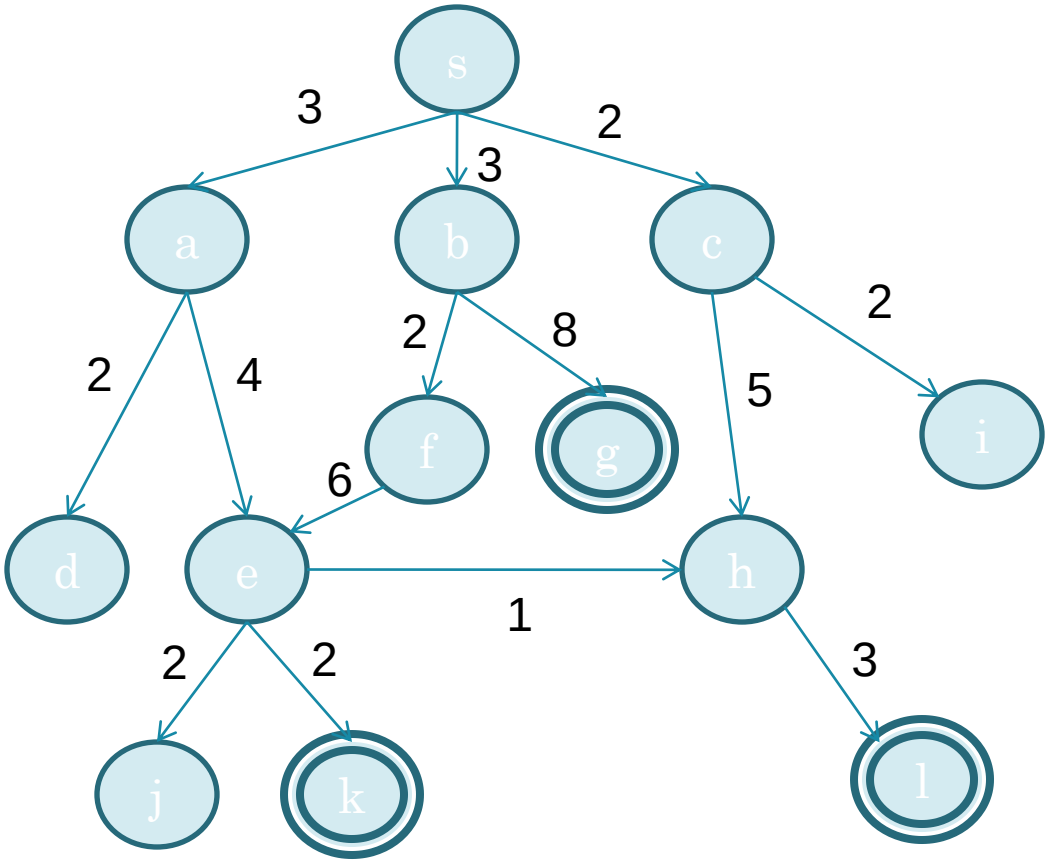
PRIMER – A* (1/9)



	s	a	b	c	d	e	f	g	h	i	j	k	l
h(n)	6	5	9	4	10	2	10	0	1	12	12	0	0

ali je *h(n)* dopustna?

PRIMER – A* (2/9)



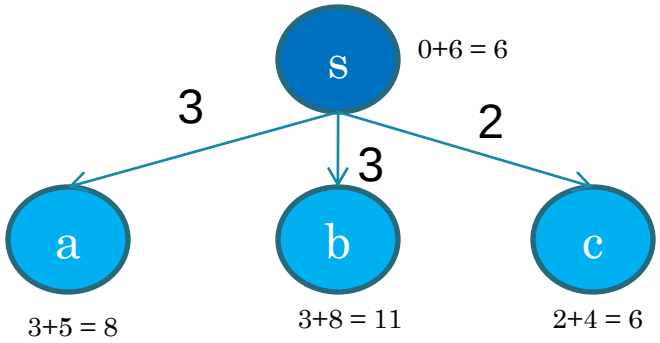
	s	a	b	c	d	e	f	g	h	i	j	k	l
h(n)	6	5	9 8	4	10	2	10 8	0	1	12	12	0	0

PRIMER – A* (3/9)

0+6 = 6

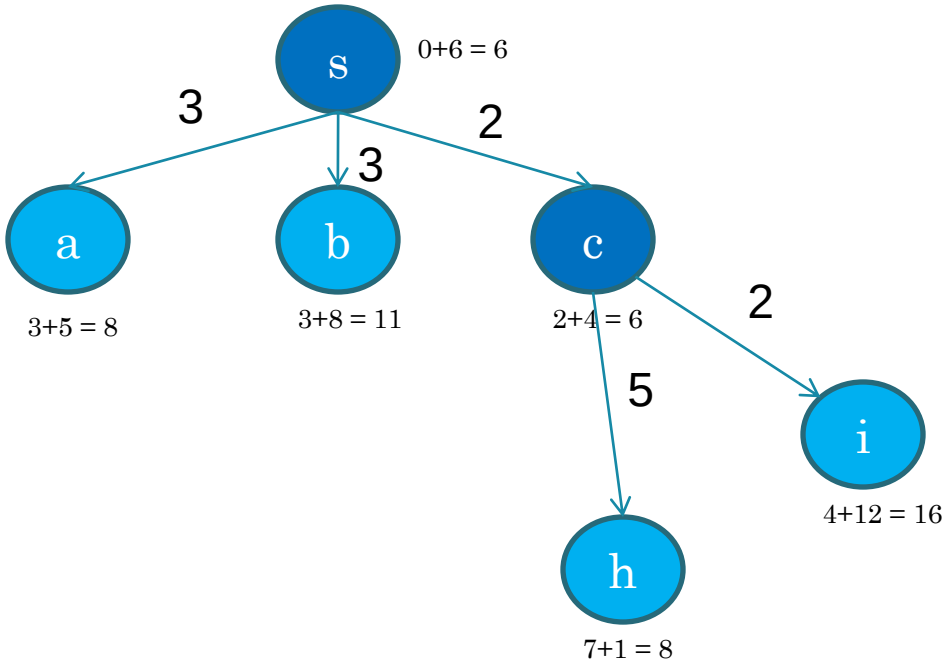
	s	a	b	c	d	e	f	g	h	i	j	k	l
h(n)	6	5	8	4	10	2	8	0	1	12	12	0	0

PRIMER – A* (4/9)



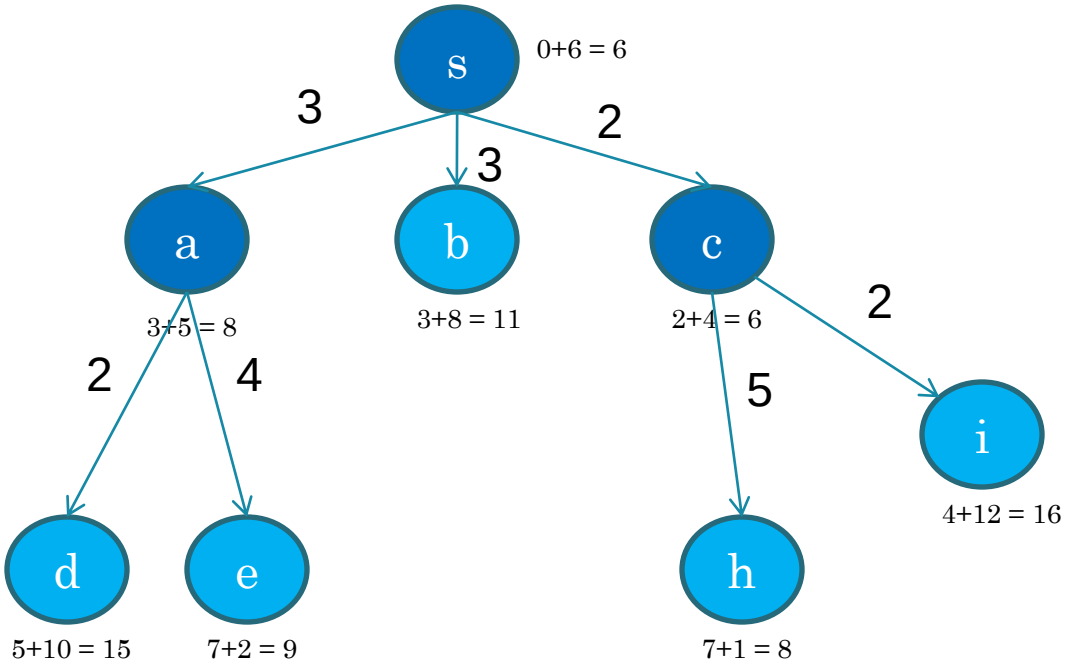
	s	a	b	c	d	e	f	g	h	i	j	k	l
h(n)	6	5	8	4	10	2	8	0	1	12	12	0	0

PRIMER – A* (5/9)



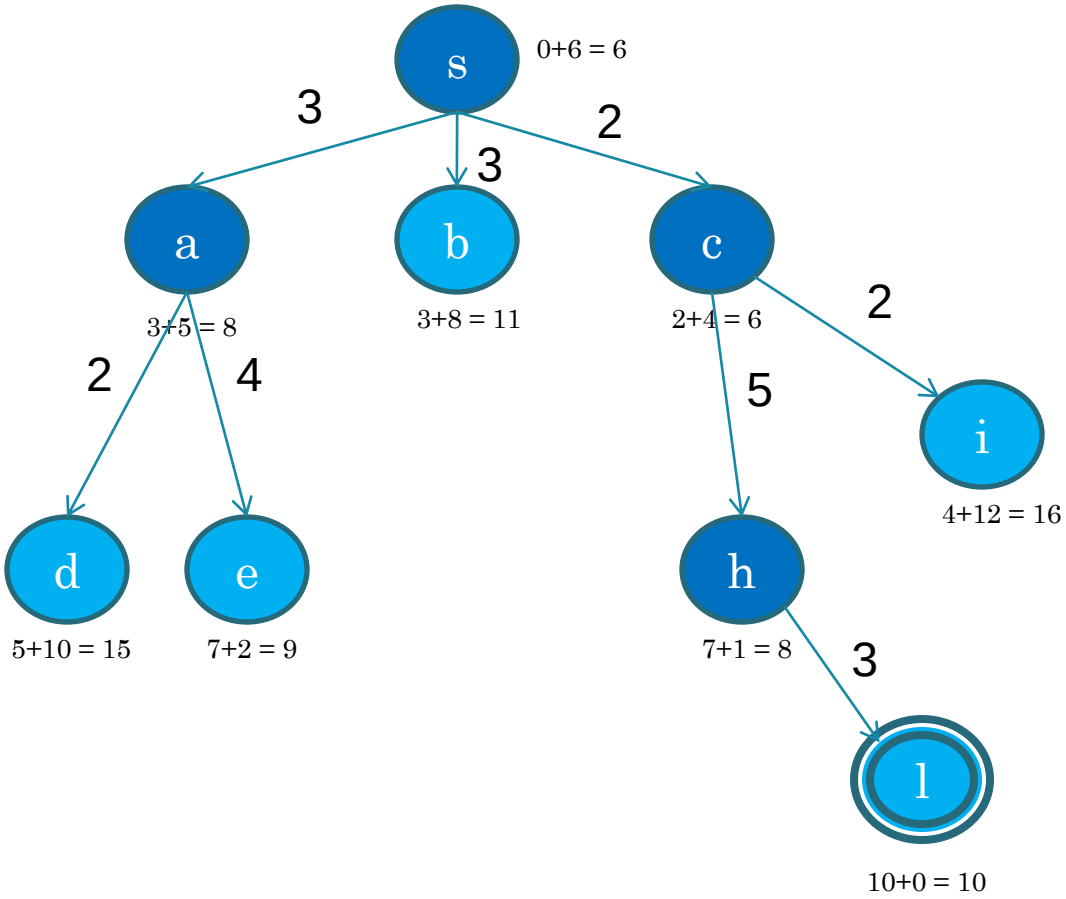
	s	a	b	c	d	e	f	g	h	i	j	k	l
h(n)	6	5	8	4	10	2	8	0	1	12	12	0	0

PRIMER – A* (6/9)



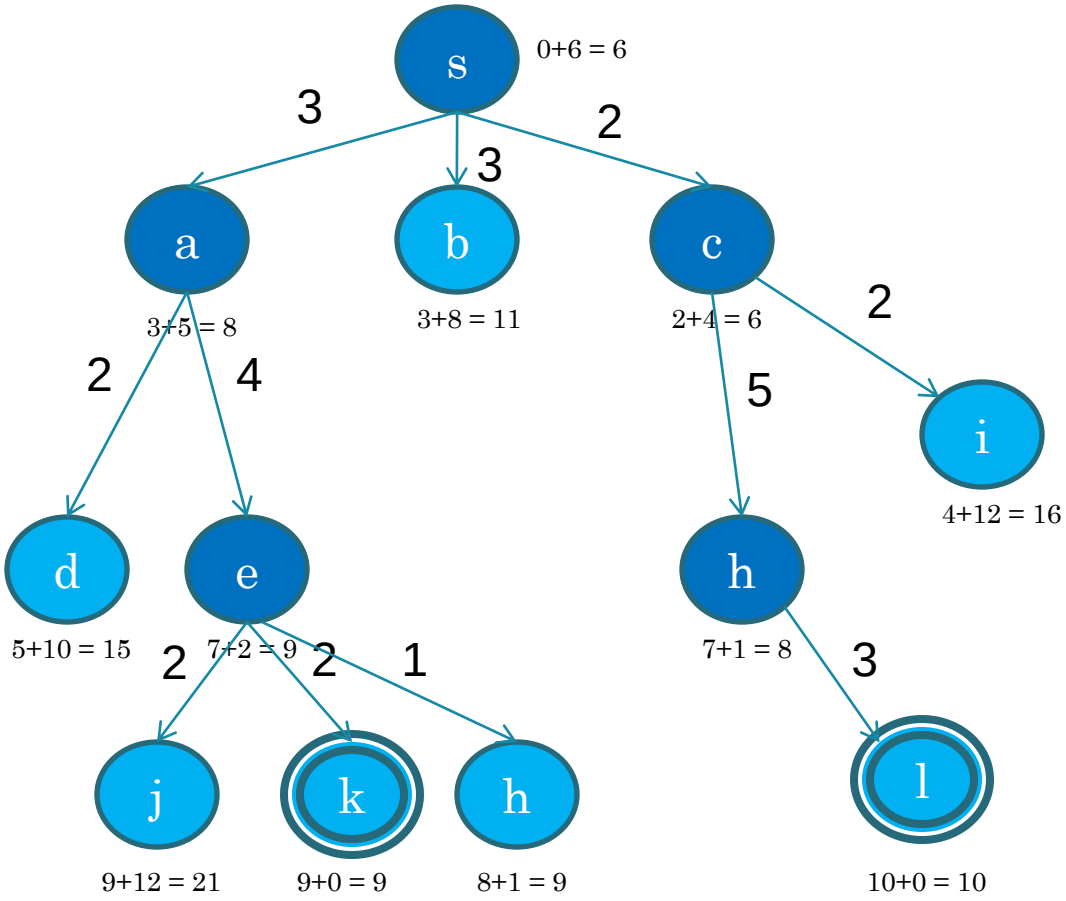
	s	a	b	c	d	e	f	g	h	i	j	k	l
h(n)	6	5	8	4	10	2	8	0	1	12	12	0	0

PRIMER – A* (7/9)



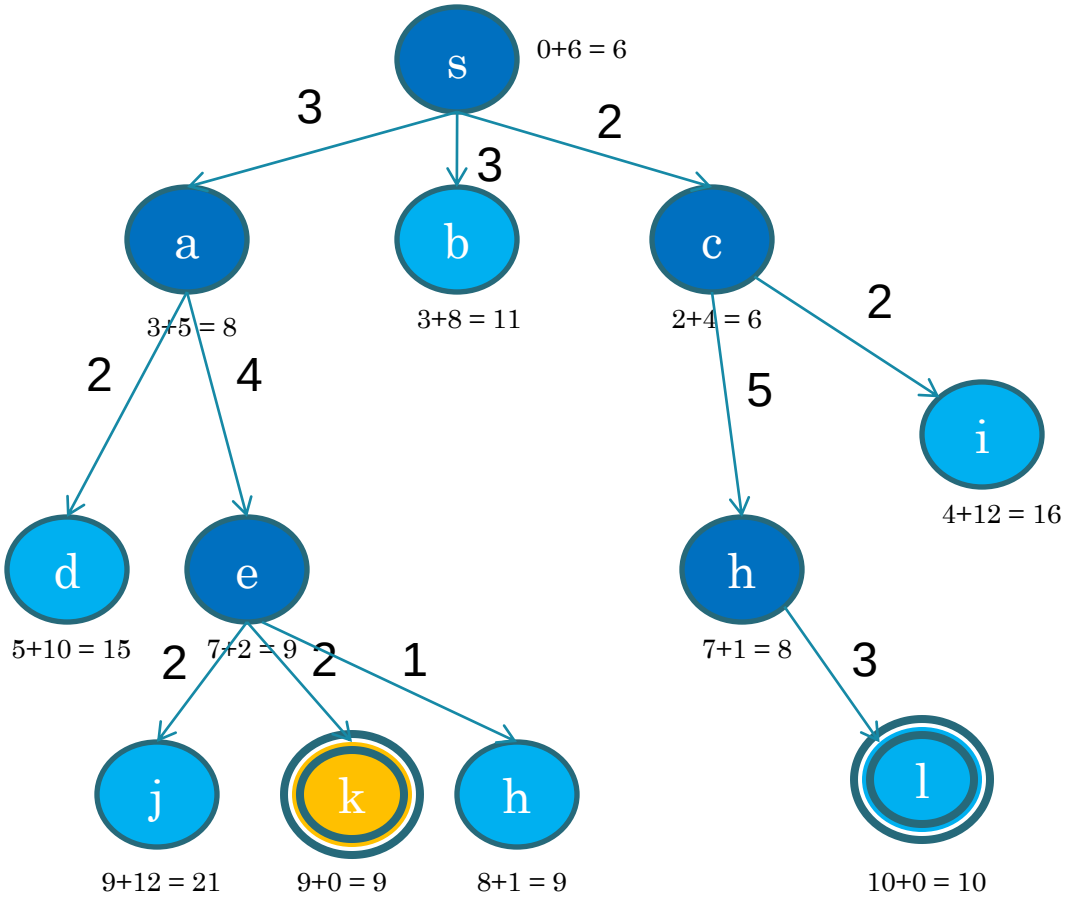
	s	a	b	c	d	e	f	g	h	i	j	k	l
h(n)	6	5	8	4	10	2	8	0	1	12	12	0	0

PRIMER – A* (8/9)



	s	a	b	c	d	e	f	g	h	i	j	k	l
h(n)	6	5	8	4	10	2	8	0	1	12	12	0	0

PRIMER – A* (9/9)

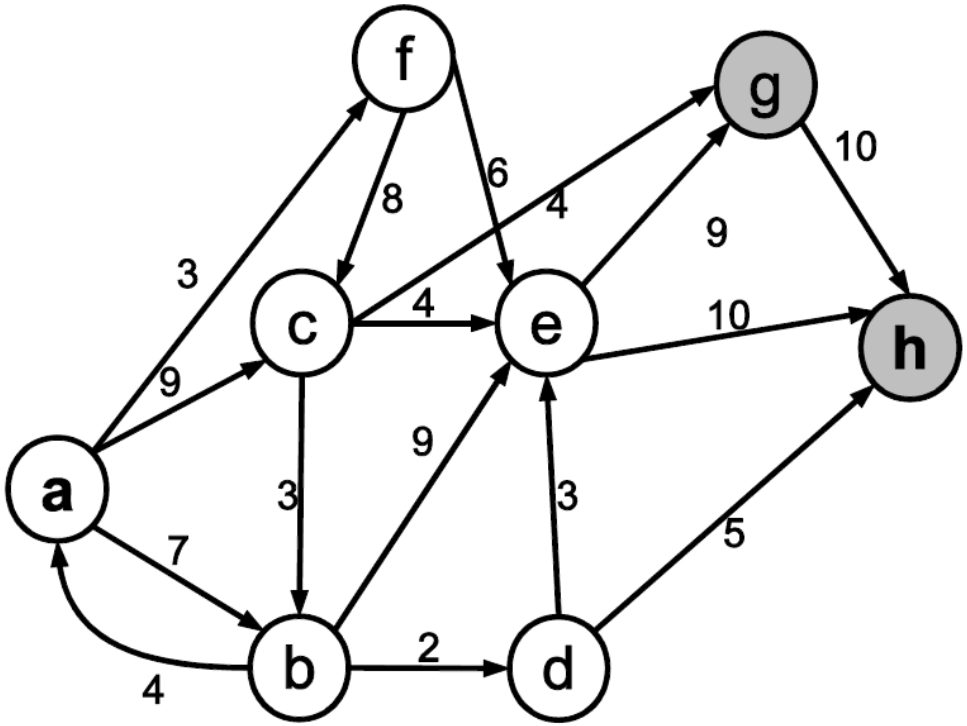


	s	a	b	c	d	e	f	g	h	i	j	k	l
h(n)	6	5	8	4	10	2	8	0	1	12	12	0	0

SAMOSTOJNO DELO

Za spodnji graf z začetnim vozliščem *a* in s končnima vozliščema *g* in *h* določi zaporedje razvijanja vozlišč, če uporabljamo A* s hevristično oceno iz spodnje tabele:

a	b	c	d	e	f	g	h
8	2	4	3	9	12	0	0



Izboljšave algoritma A*

Težava algoritma A* je prevelika poraba pomnilnika.

Izboljšave porabijo manj pomnilnika, a še vedno zagotavljajo optimalnost rešitve.

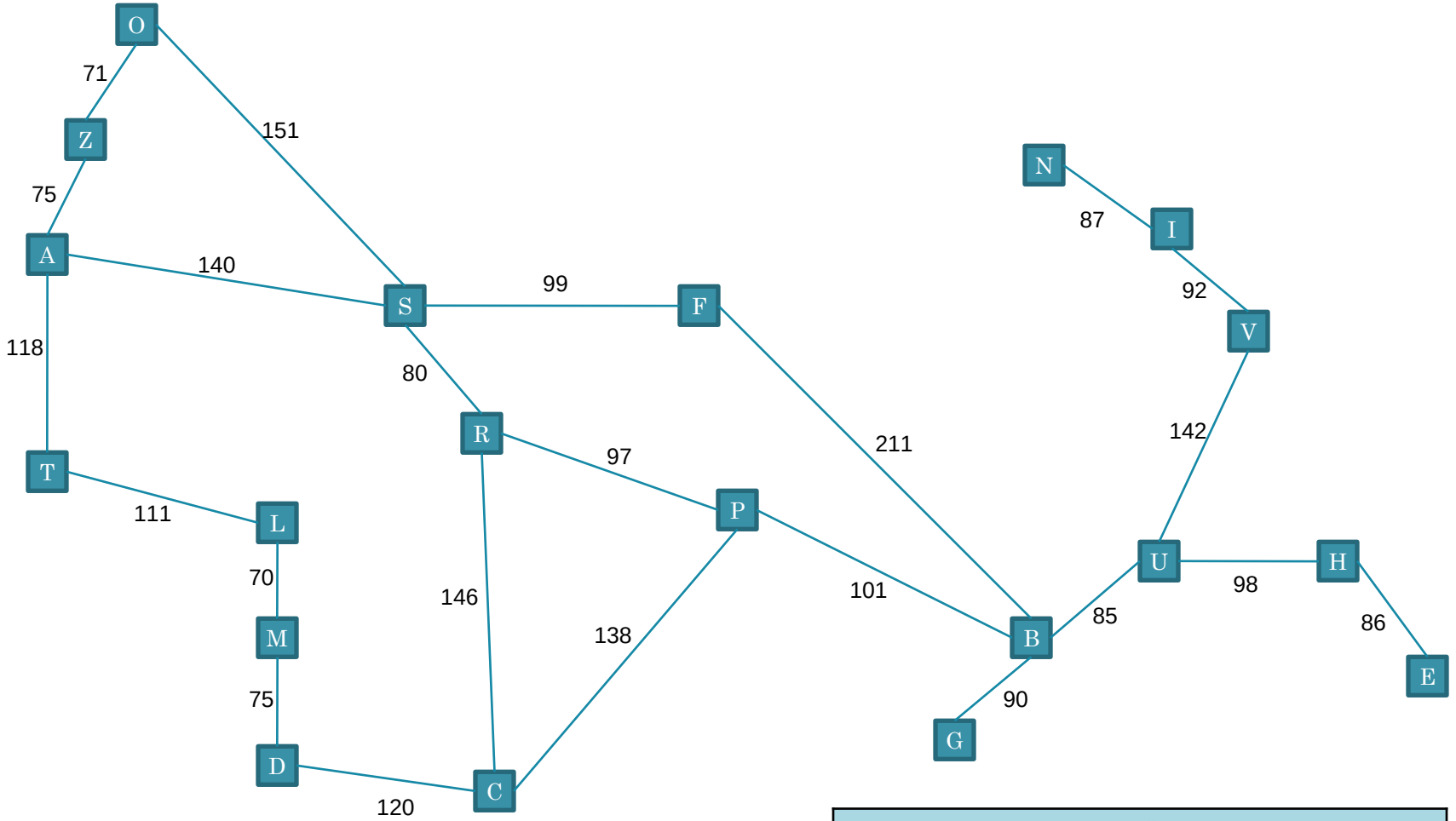
Iterative-deepening A* (IDA*):

- namesto povečevanja globine iskanja, povečuje vrednost hevristične ocene *f(n)*

Recursive best first search (RBFS):

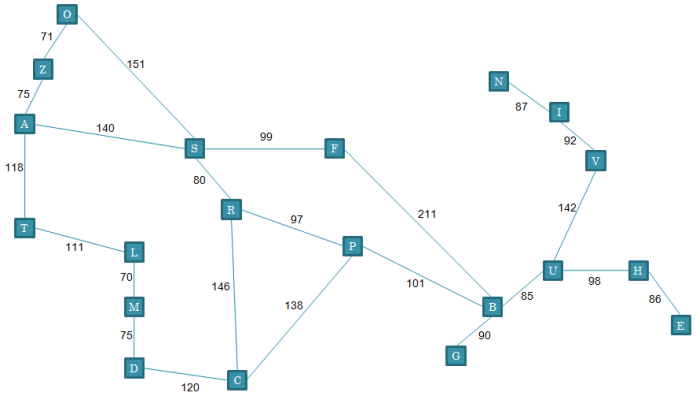
- shrani vrednosti vseh otrok na trenutni poti
- išče do meje sobratov
- pri vračanju si zapomni *f* vrednost najboljšega lista
- na podlagi zapomnjene *f* vrednosti ve, katere veje so perspektivne

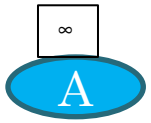
Primer RBFS (1/13)



h(n) = Razdalja po zračni liniji do mesta B							
A	366	F	176	M	241	S	253
B	0	G	77	N	234	T	329
C	160	H	151	O	380	U	80
D	242	I	226	P	100	V	199
E	161	L	244	R	193	Z	374

Primer RBFS (2/13)

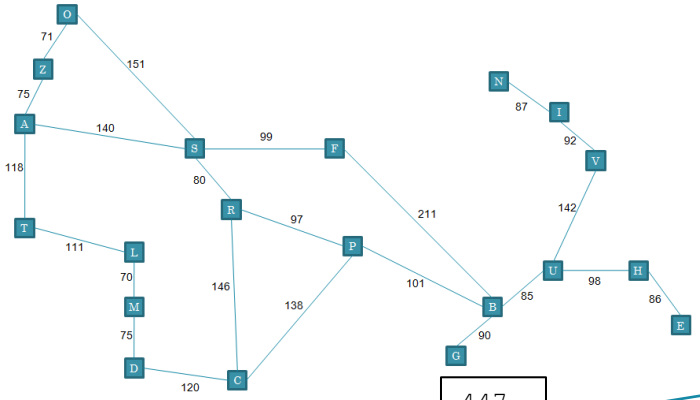




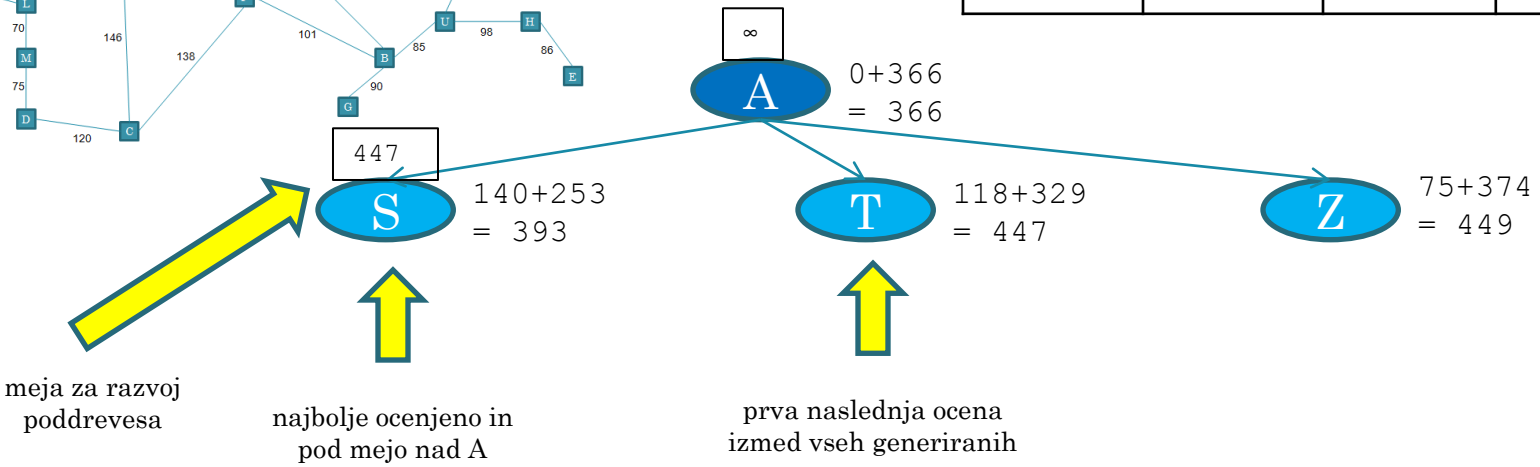
0+366
= 366

h(n) = Razdalja po zračni liniji do mesta B							
A	366	F	176	M	241	S	253
B	0	G	77	N	234	T	329
C	160	H	151	O	380	U	80
D	242	I	226	P	100	V	199
E	161	L	244	R	193	Z	374

PRIMER RBFS (3/13)



h(n) = Razdalja po zračni liniji do mesta B							
A	366	F	176	M	241	S	253
B	0	G	77	N	234	T	329
C	160	H	151	O	380	U	80
D	242	I	226	P	100	V	199
E	161	L	244	R	193	Z	374

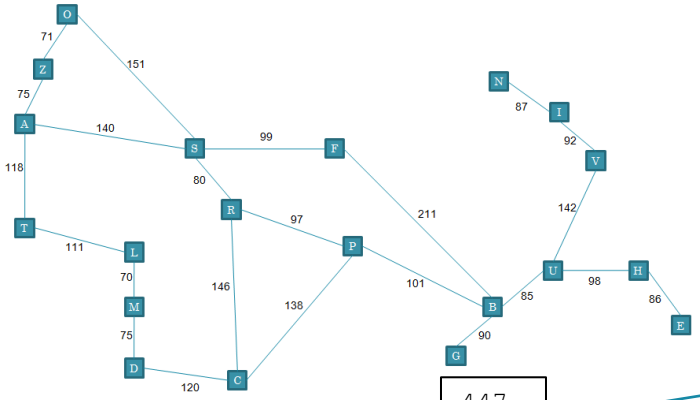


meja za razvoj poddrevesa

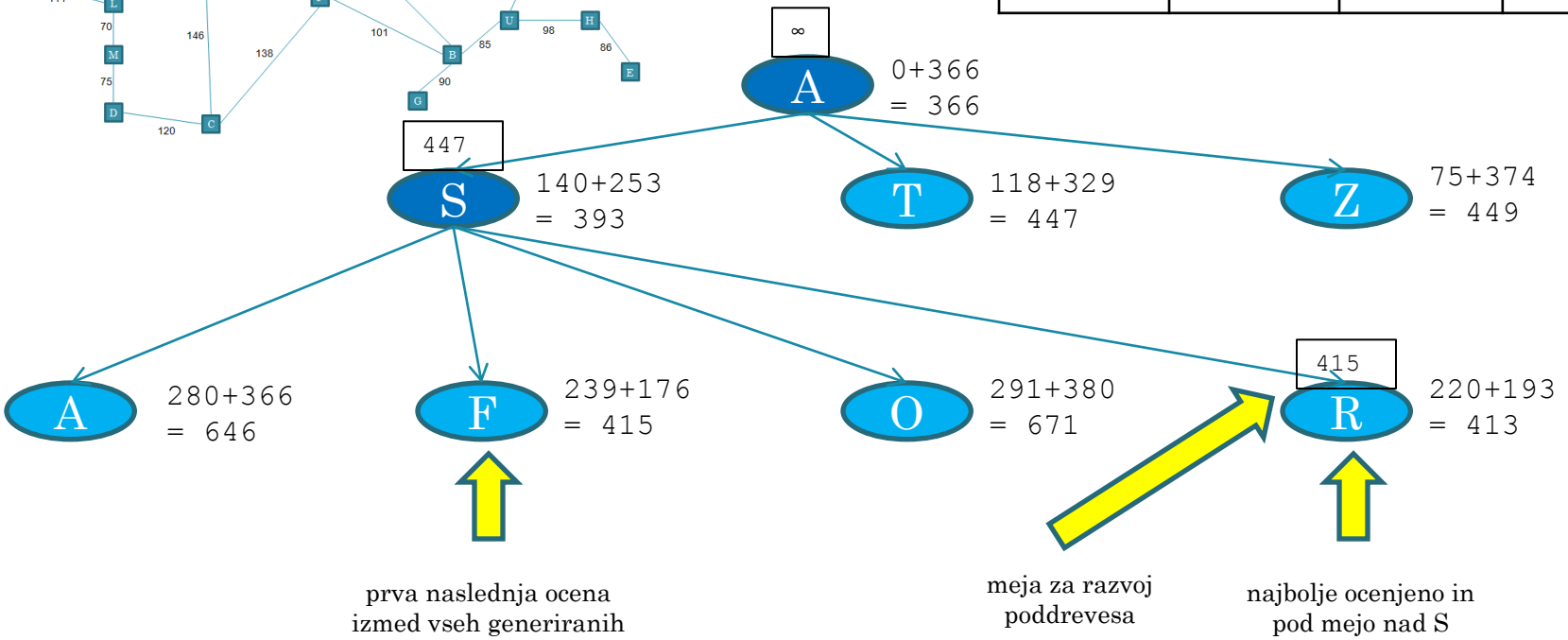
najbolje ocenjeno in pod mejo nad A

prva naslednja ocena izmed vseh generiranih

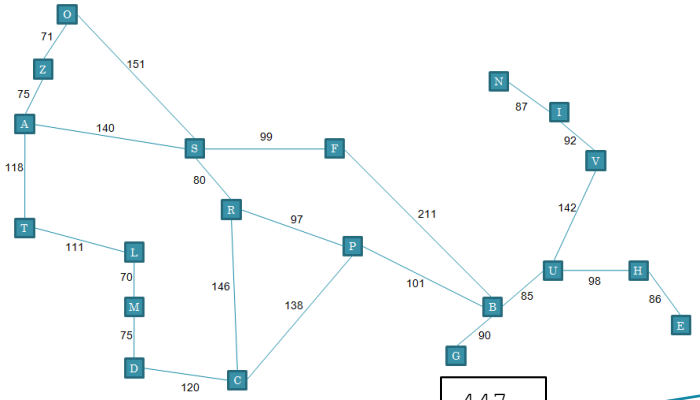
PRIMER RBFS (4/13)



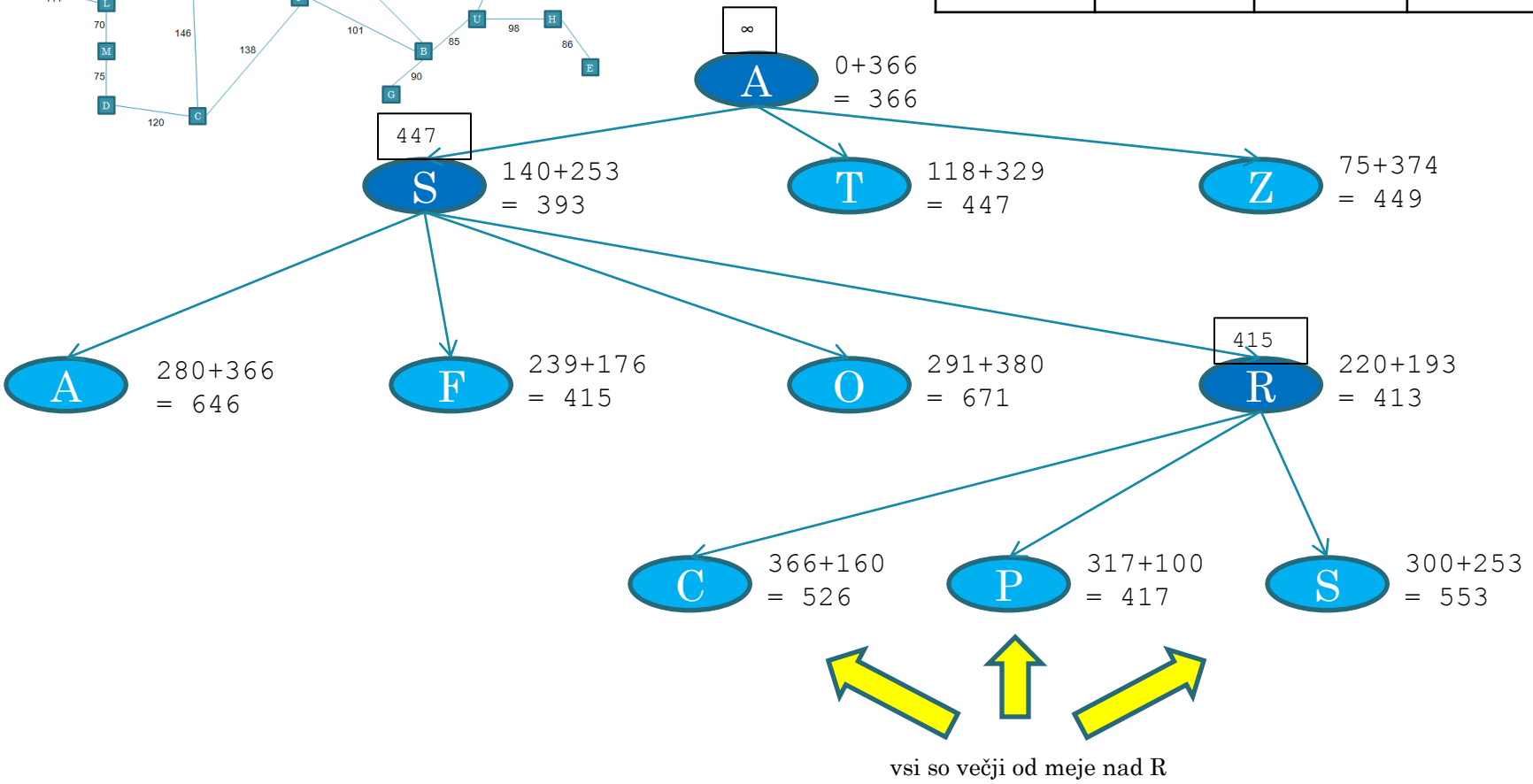
h(n) = Razdalja po zračni liniji do mesta B							
A	366	F	176	M	241	S	253
B	0	G	77	N	234	T	329
C	160	H	151	O	380	U	80
D	242	I	226	P	100	V	199
E	161	L	244	R	193	Z	374



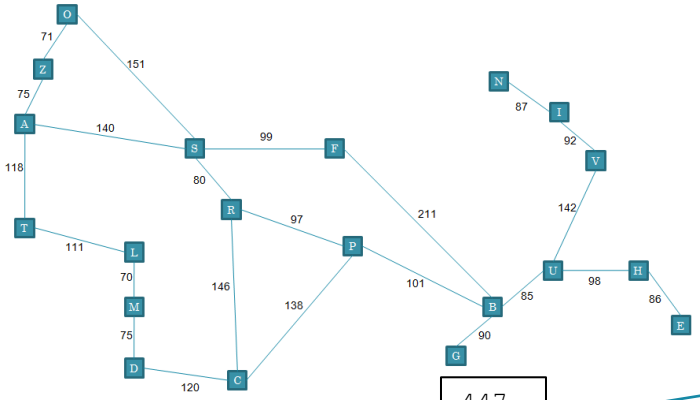
PRIMER RBFS (5/13)



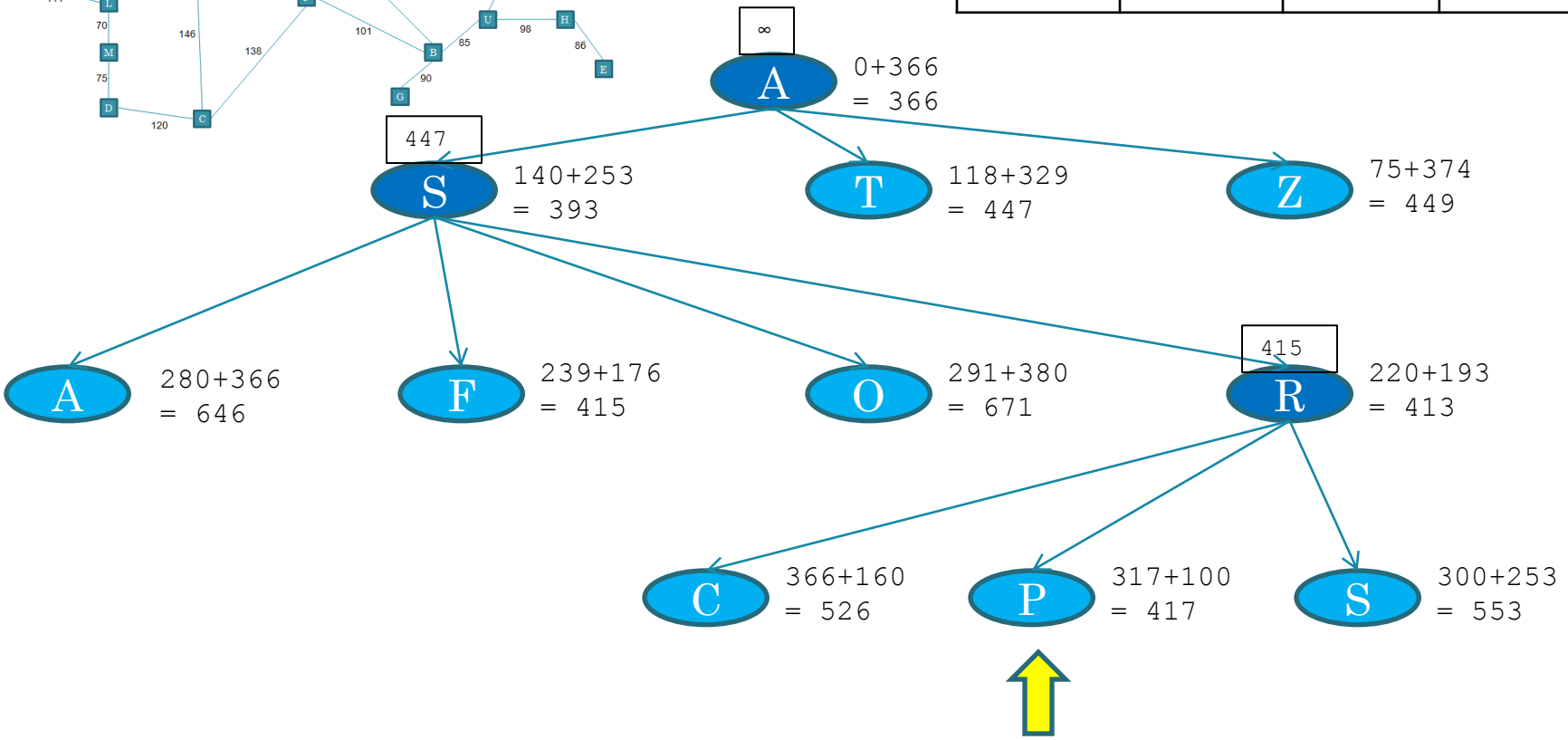
h(n) = Razdalja po zračni liniji do mesta B							
A	366	F	176	M	241	S	253
B	0	G	77	N	234	T	329
C	160	H	151	O	380	U	80
D	242	I	226	P	100	V	199
E	161	L	244	R	193	Z	374



PRIMER RBFS (6/13)

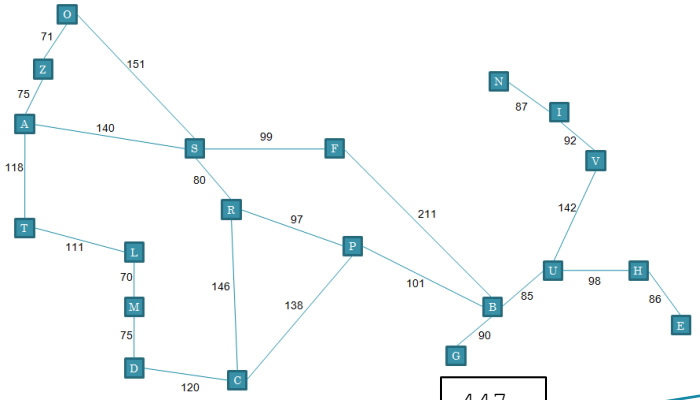


h(n) = Razdalja po zračni liniji do mesta B							
A	366	F	176	M	241	S	253
B	0	G	77	N	234	T	329
C	160	H	151	O	380	U	80
D	242	I	226	P	100	V	199
E	161	L	244	R	193	Z	374

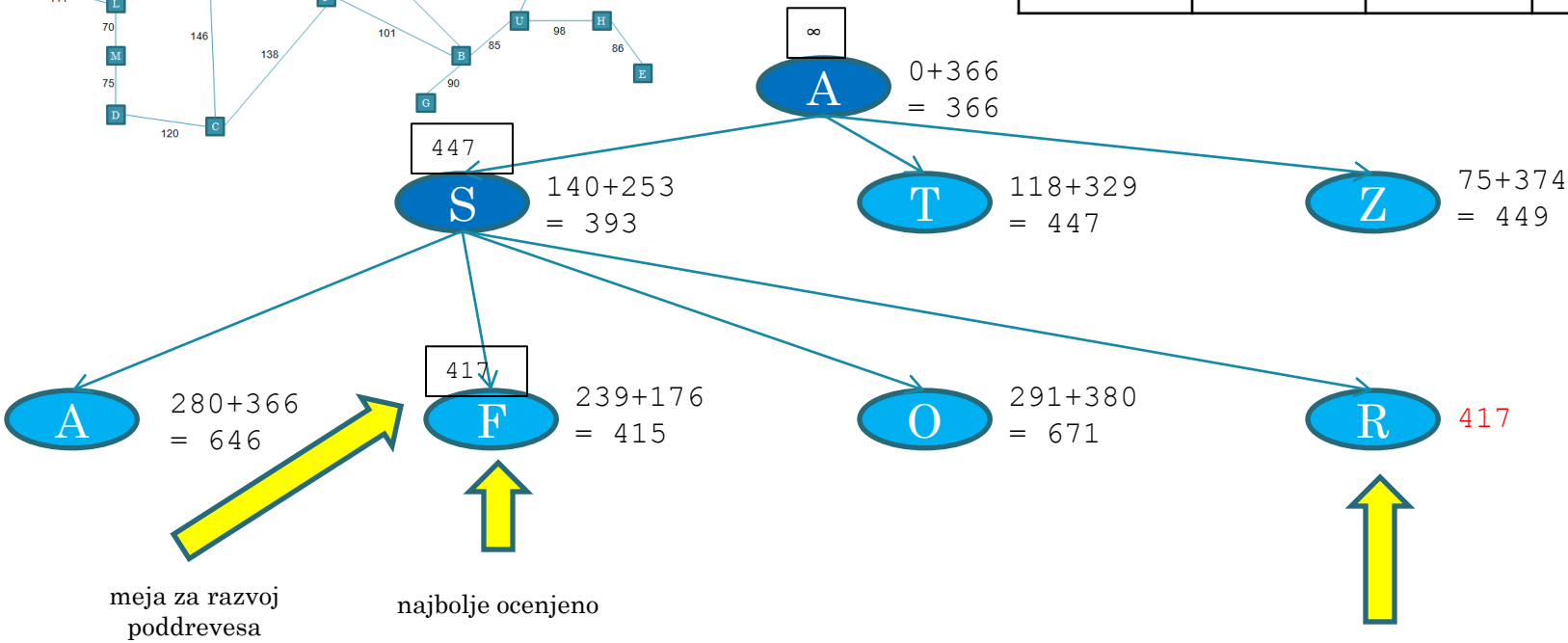


najboljša ocena poddrevesa

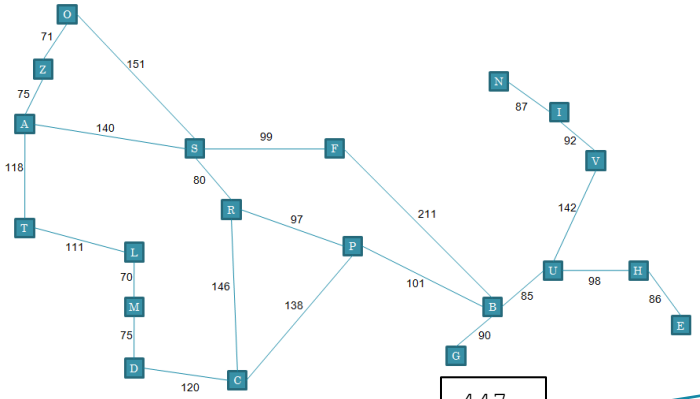
PRIMER RBFS (7/13)



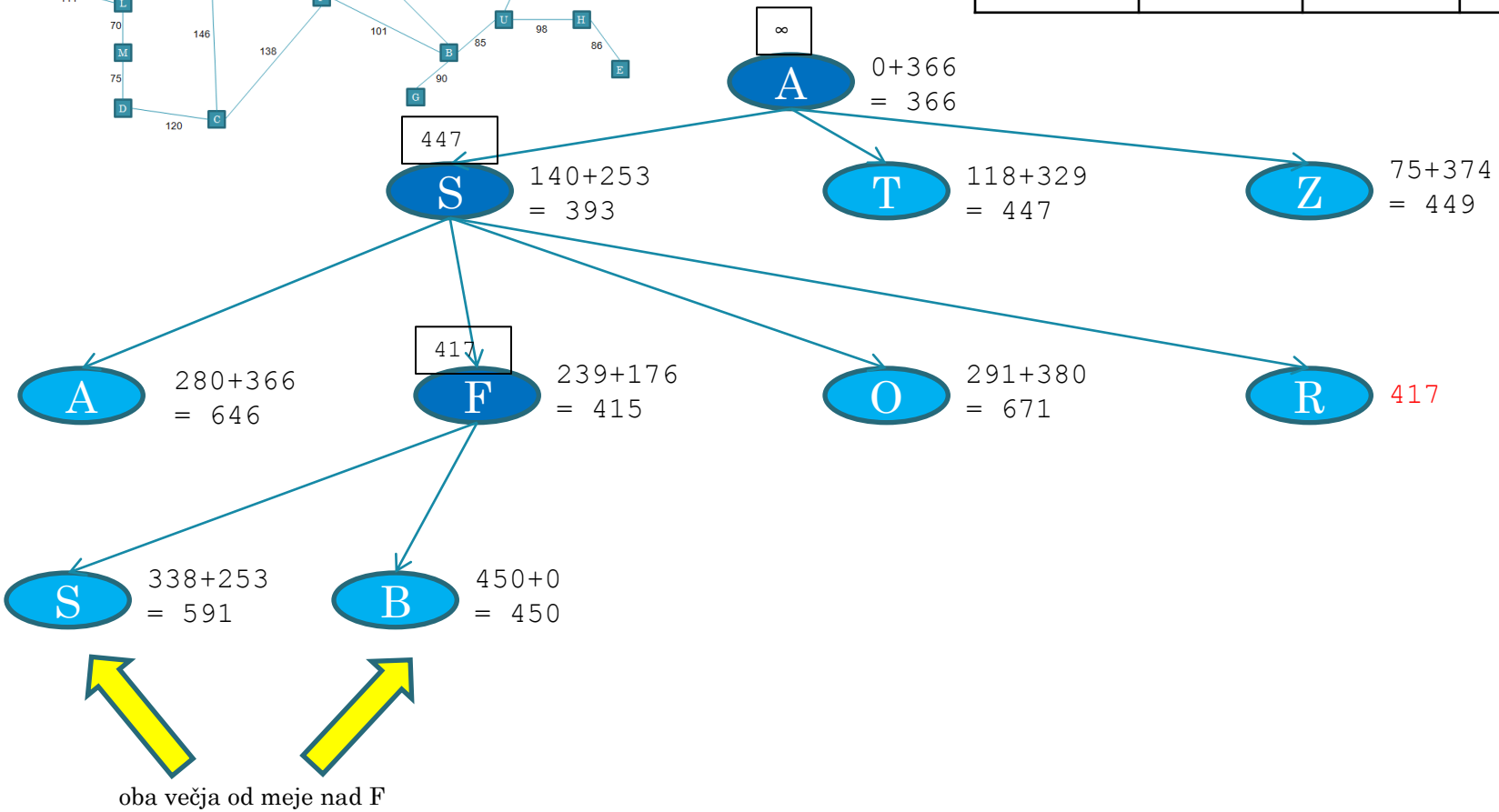
$h(n)$ = Razdalja po zračni liniji do mesta B							
A	366	F	176	M	241	S	253
B	0	G	77	N	234	T	329
C	160	H	151	O	380	U	80
D	242	I	226	P	100	V	199
E	161	L	244	R	193	Z	374



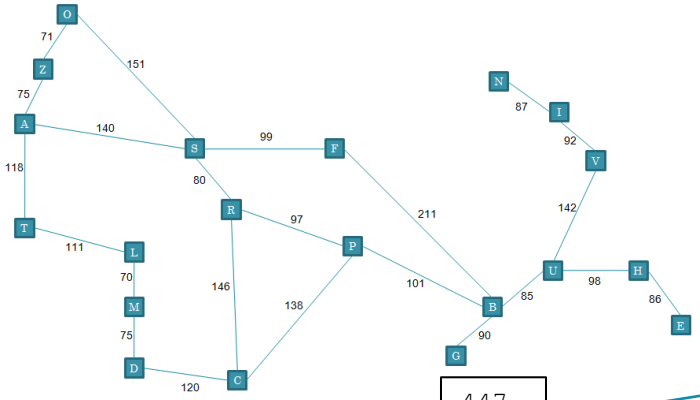
PRIMER RBFS (8/13)



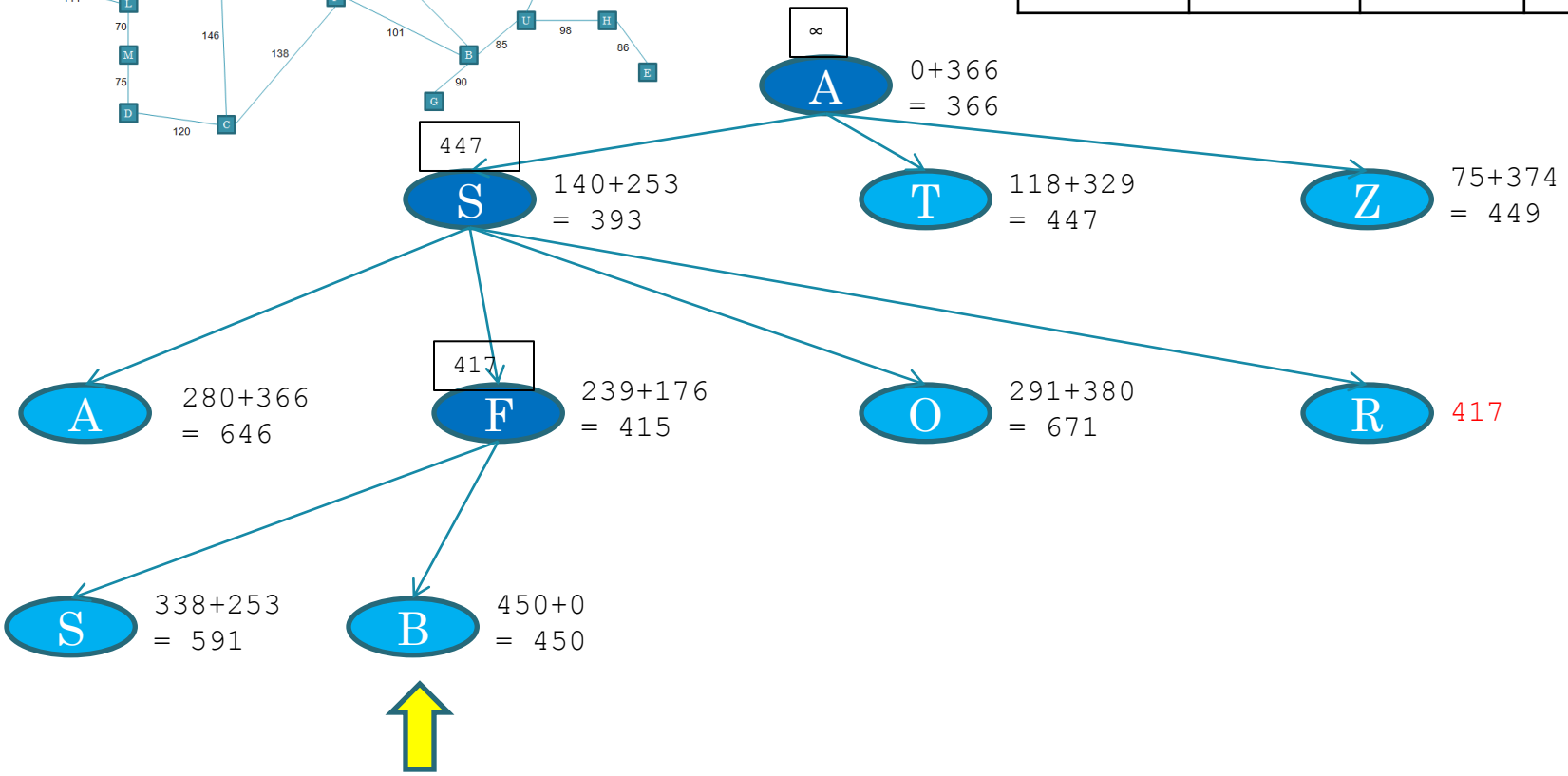
h(n) = Razdalja po zračni liniji do mesta B							
A	366	F	176	M	241	S	253
B	0	G	77	N	234	T	329
C	160	H	151	O	380	U	80
D	242	I	226	P	100	V	199
E	161	L	244	R	193	Z	374



PRIMER RBFS (9/13)

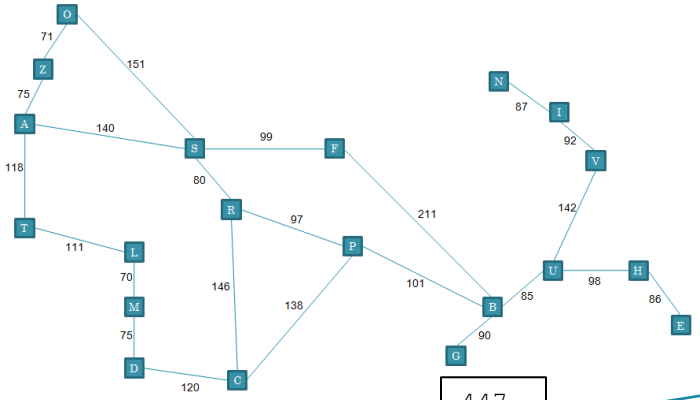


h(n) = Razdalja po zračni liniji do mesta B							
A	366	F	176	M	241	S	253
B	0	G	77	N	234	T	329
C	160	H	151	O	380	U	80
D	242	I	226	P	100	V	199
E	161	L	244	R	193	Z	374

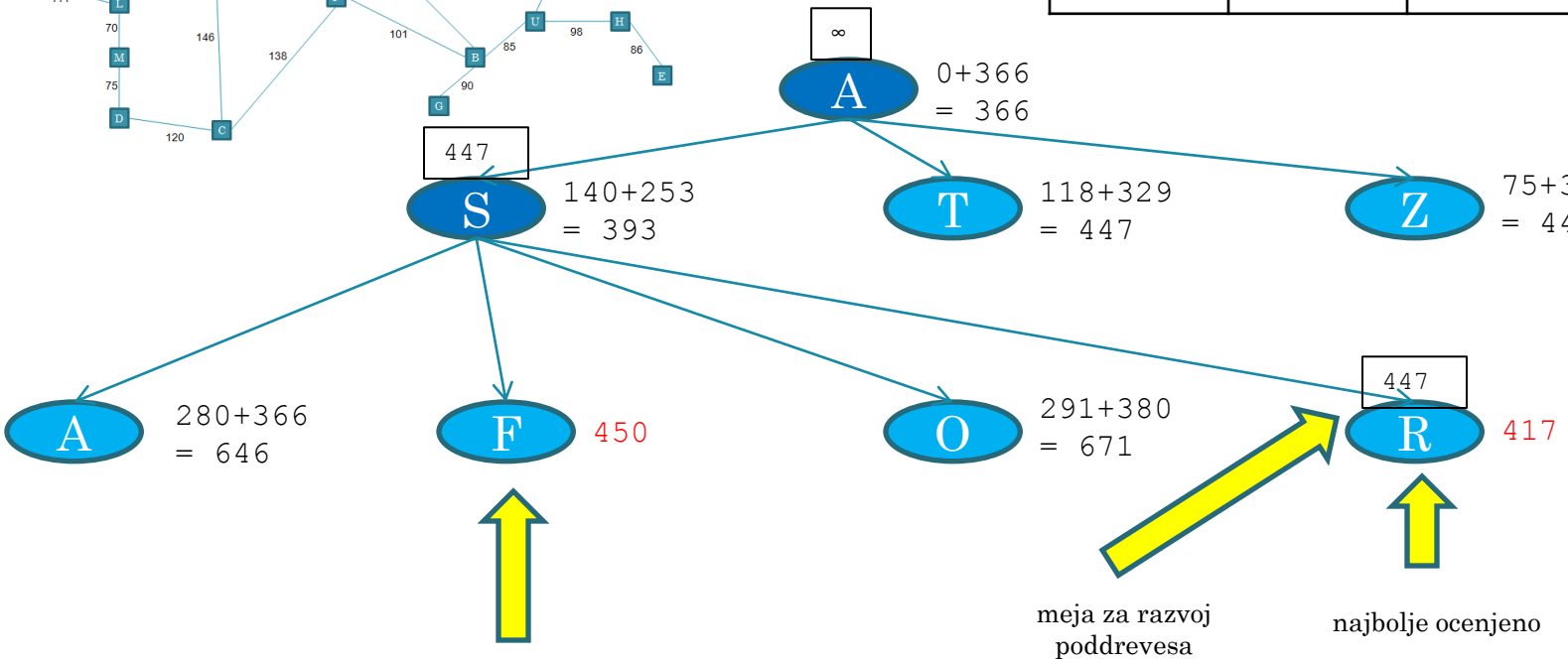


najboljša ocena poddrevesa

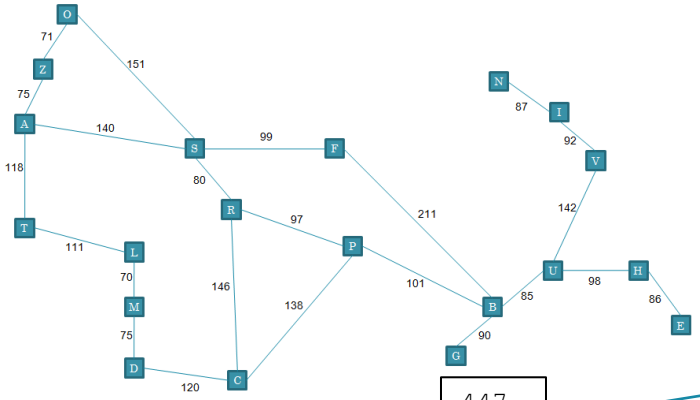
PRIMER RBFS (10/13)



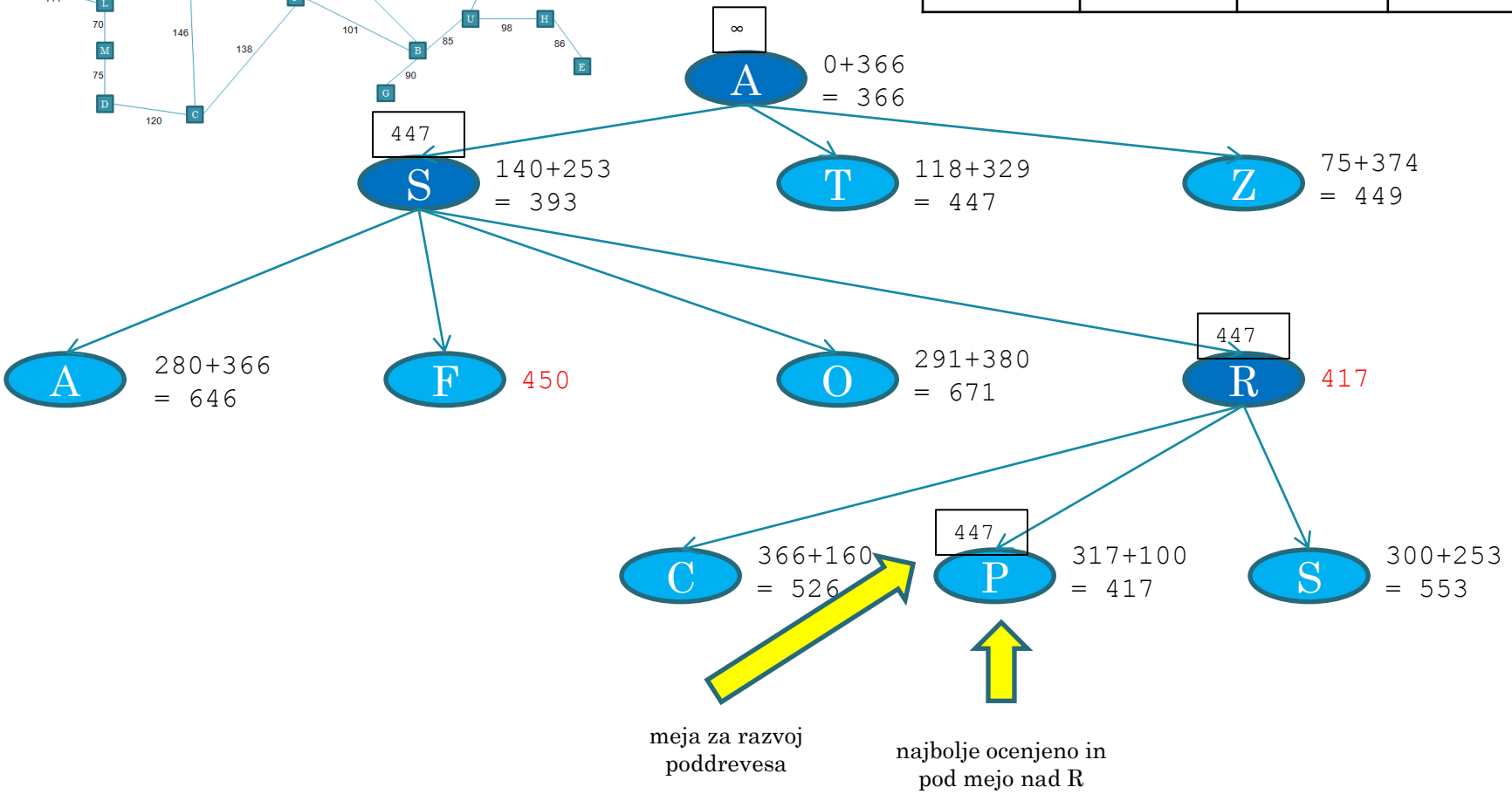
h(n) = Razdalja po zračni liniji do mesta B							
A	366	F	176	M	241	S	253
B	0	G	77	N	234	T	329
C	160	H	151	O	380	U	80
D	242	I	226	P	100	V	199
E	161	L	244	R	193	Z	374



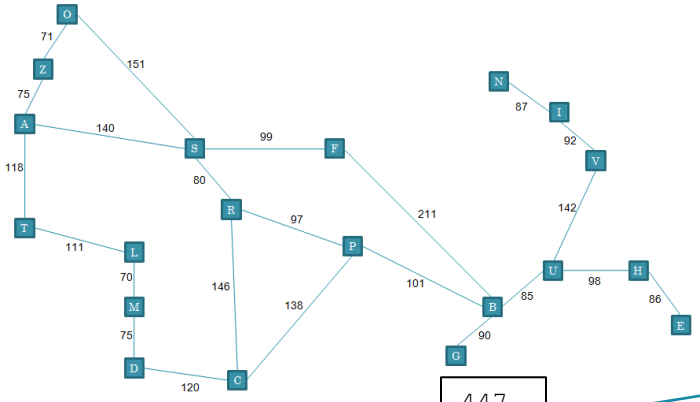
PRIMER RBFS (11/13)



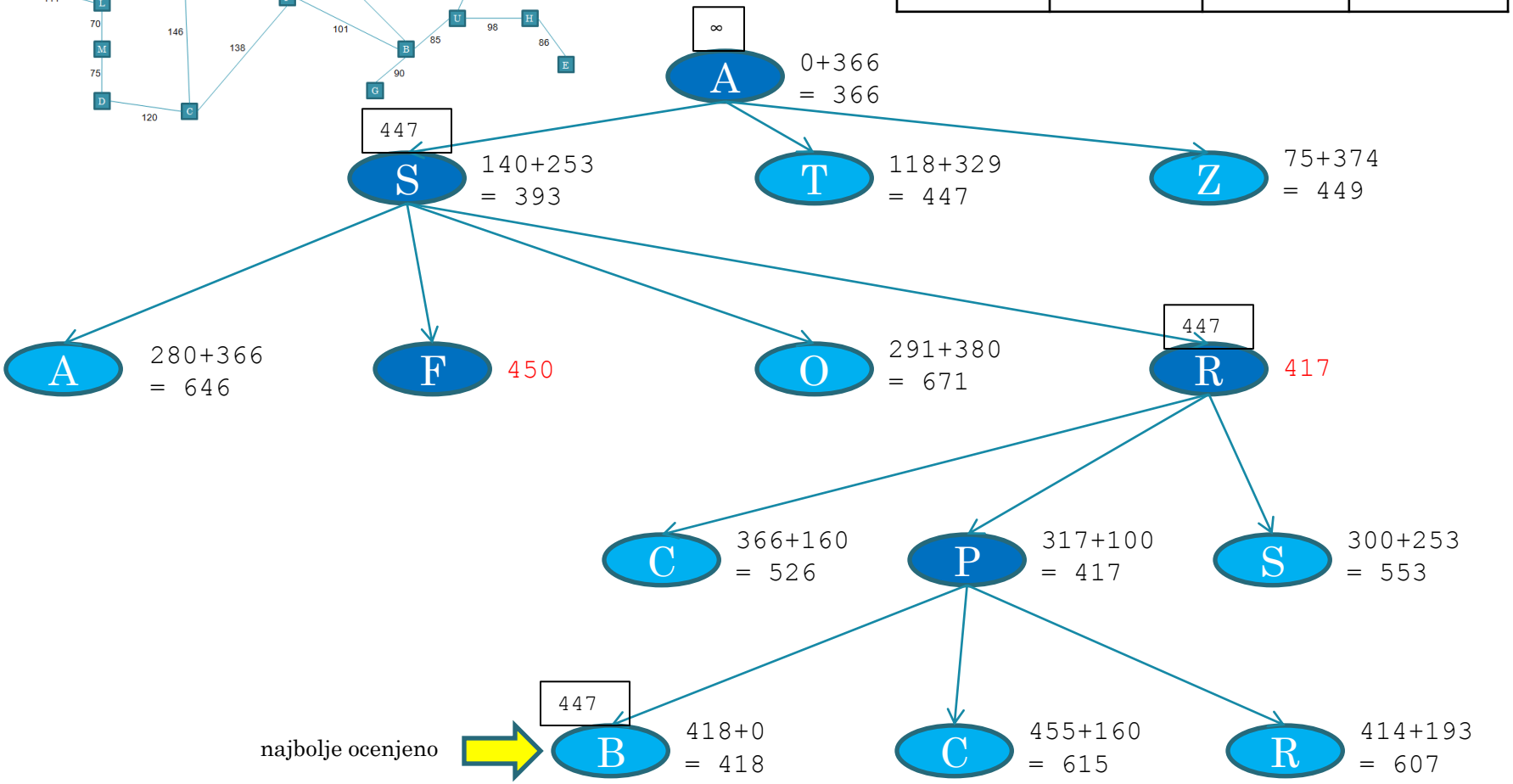
h(n) = Razdalja po zračni liniji do mesta B							
A	366	F	176	M	241	S	253
B	0	G	77	N	234	T	329
C	160	H	151	O	380	U	80
D	242	I	226	P	100	V	199
E	161	L	244	R	193	Z	374



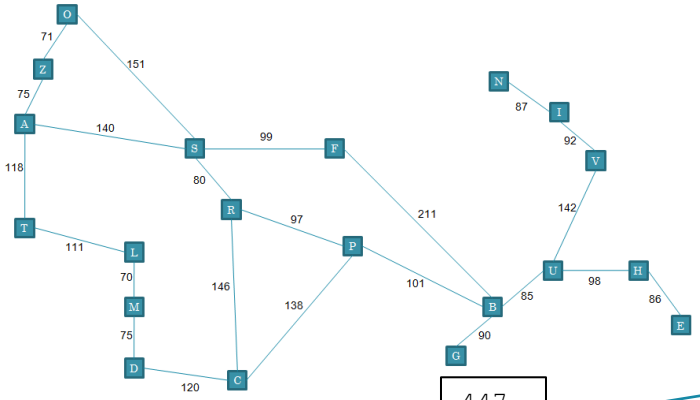
PRIMER RBFS (12/13)



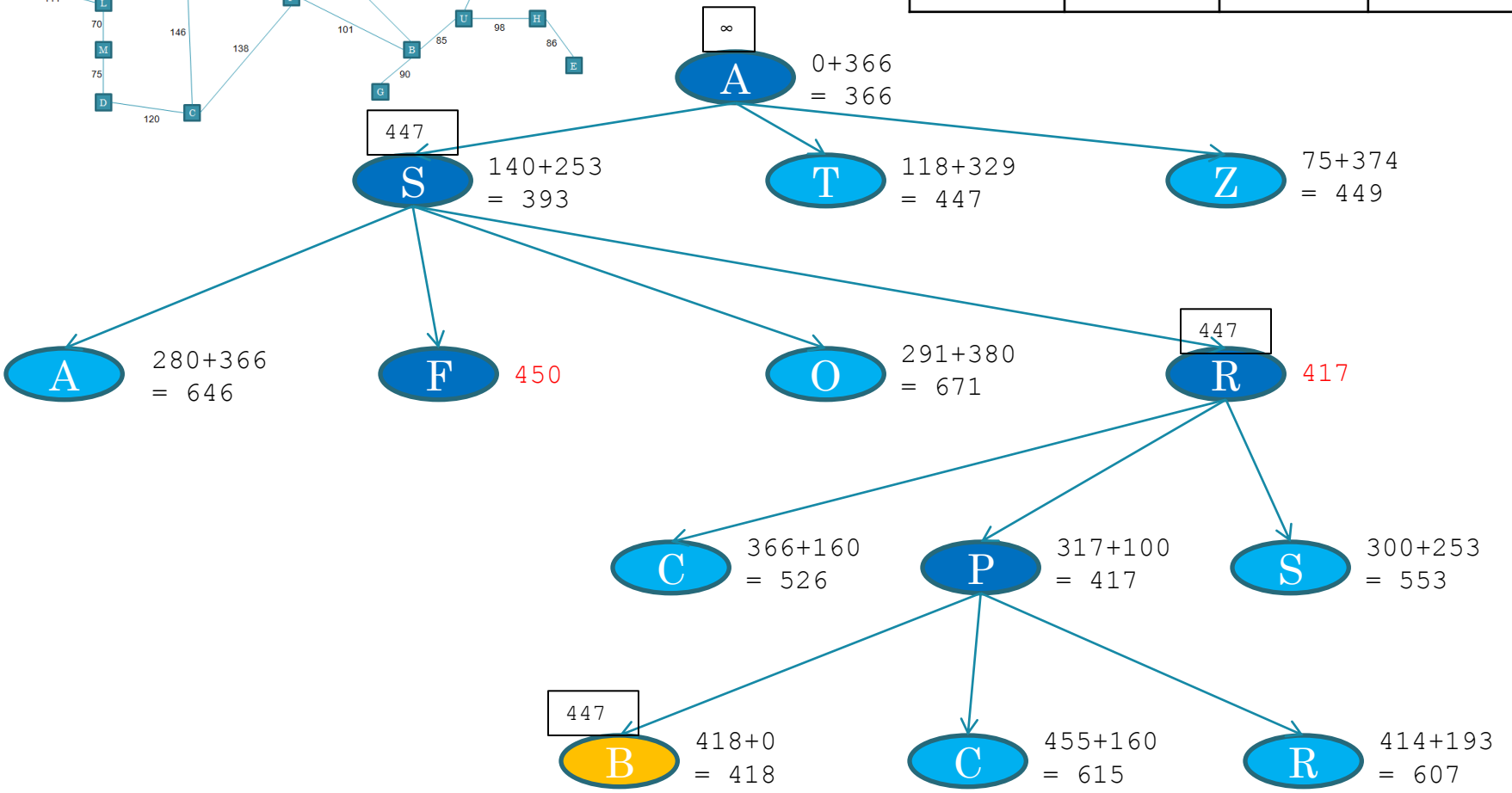
h(n) = Razdalja po zračni liniji do mesta B							
A	366	F	176	M	241	S	253
B	0	G	77	N	234	T	329
C	160	H	151	O	380	U	80
D	242	I	226	P	100	V	199
E	161	L	244	R	193	Z	374



PRIMER RBFS (13/13)



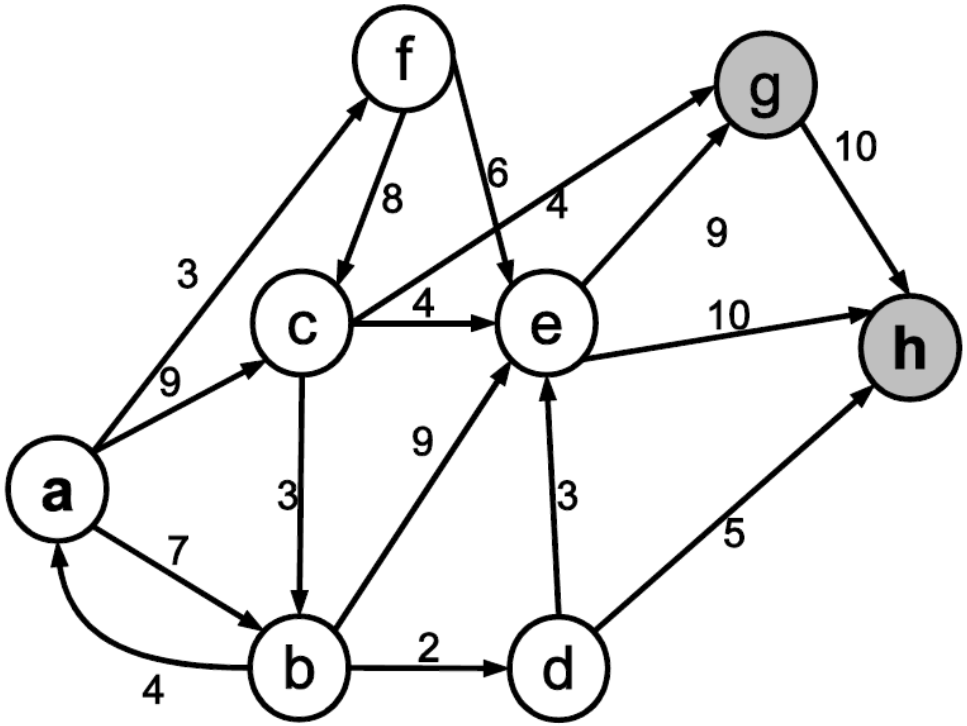
h(n) = Razdalja po zračni liniji do mesta B							
A	366	F	176	M	241	S	253
B	0	G	77	N	234	T	329
C	160	H	151	O	380	U	80
D	242	I	226	P	100	V	199
E	161	L	244	R	193	Z	374



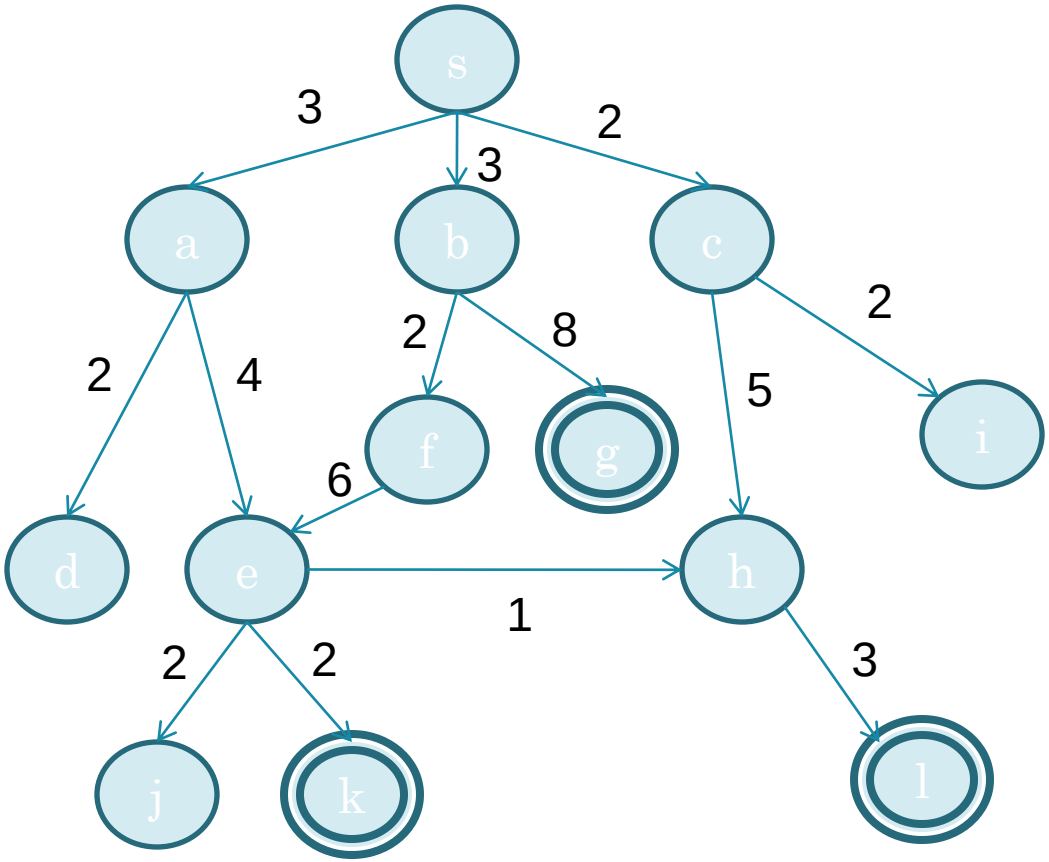
SAMOSTOJNO DELO

Za spodnji graf z začetnim vozliščem *a* in s končnima vozliščema *g* in *h* določi zaporedje razvijanja vozlišč, če uporabljamo RBFS s hevristično oceno iz spodnje tabele:

a	b	c	d	e	f	g	h
8	2	4	3	9	12	0	0

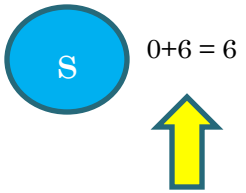


PRIMER – IDA* (1/13)



	s	a	b	c	d	e	f	g	h	i	j	k	l
h(n)	6	5	8	4	10	2	8	0	1	12	12	0	0

PRIMER – IDA* (2/13)



Meja $f(n) = 0$

	s	a	b	c	d	e	f	g	h	i	j	k	l
h(n)	6	5	8	4	10	2	8	0	1	12	12	0	0

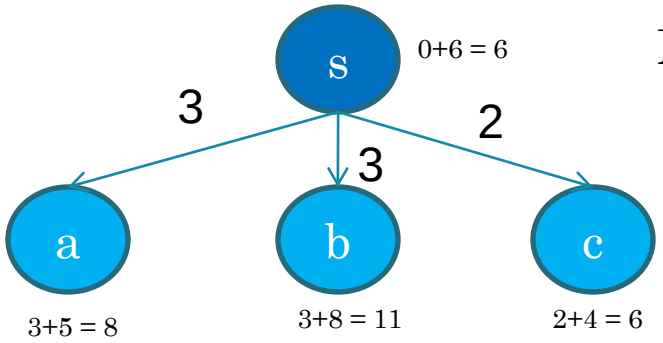
PRIMER – IDA* (3/13)

0+6 = 6

Meja $f(n) = 6$

	s	a	b	c	d	e	f	g	h	i	j	k	l
h(n)	6	5	8	4	10	2	8	0	1	12	12	0	0

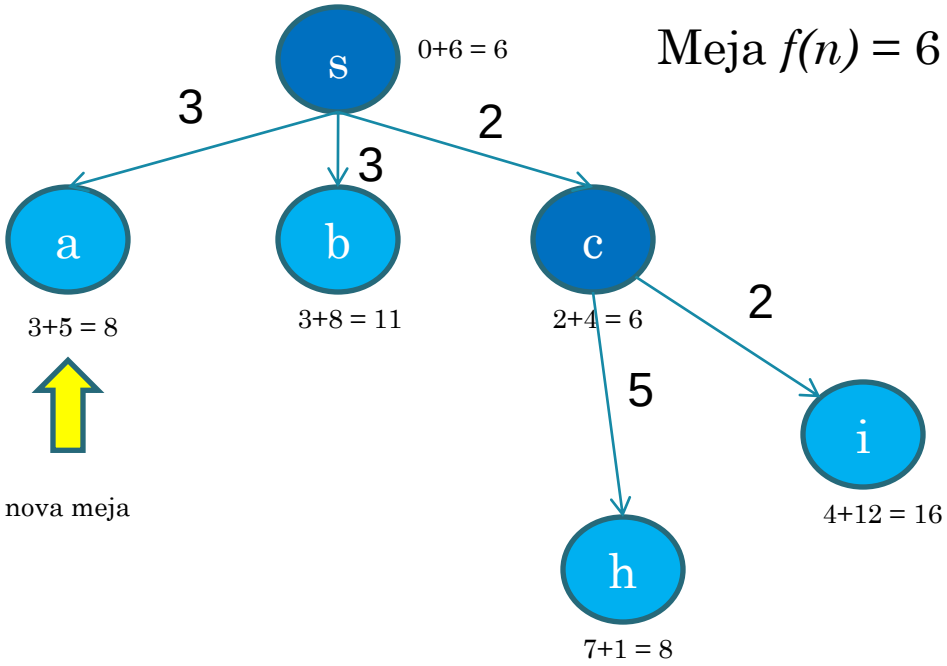
PRIMER – IDA* (4/13)



Meja $f(n) = 6$

	s	a	b	c	d	e	f	g	h	i	j	k	l
h(n)	6	5	8	4	10	2	8	0	1	12	12	0	0

PRIMER – IDA* (5/8)



	s	a	b	c	d	e	f	g	h	i	j	k	l
h(n)	6	5	8	4	10	2	8	0	1	12	12	0	0

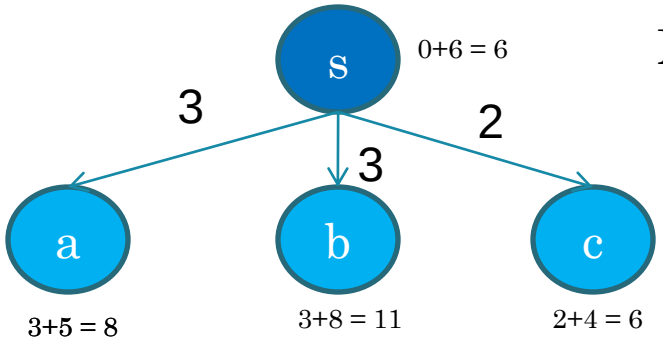
PRIMER – IDA* (6/13)

0+6 = 6

Meja $f(n) = 8$

	s	a	b	c	d	e	f	g	h	i	j	k	l
h(n)	6	5	8	4	10	2	8	0	1	12	12	0	0

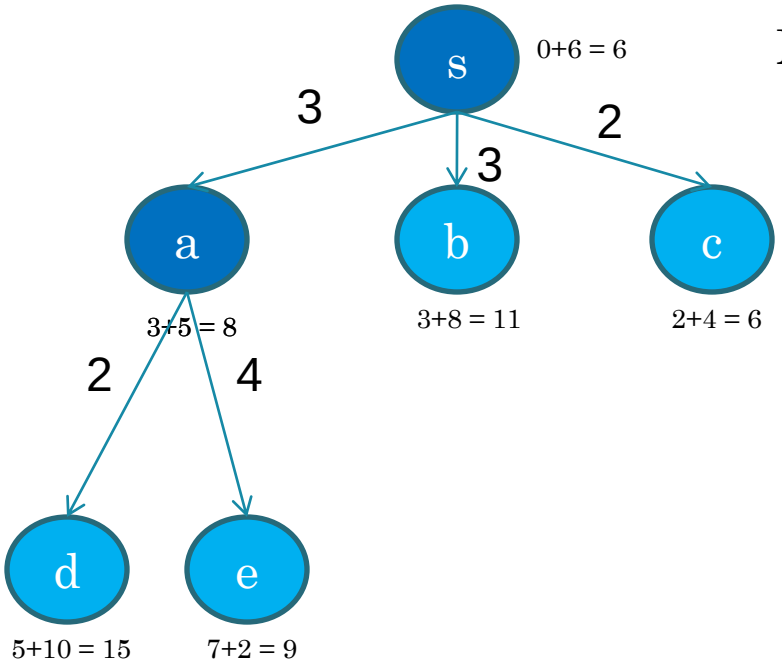
PRIMER – IDA* (7/13)



Meja $f(n) = 8$

	s	a	b	c	d	e	f	g	h	i	j	k	l
h(n)	6	5	8	4	10	2	8	0	1	12	12	0	0

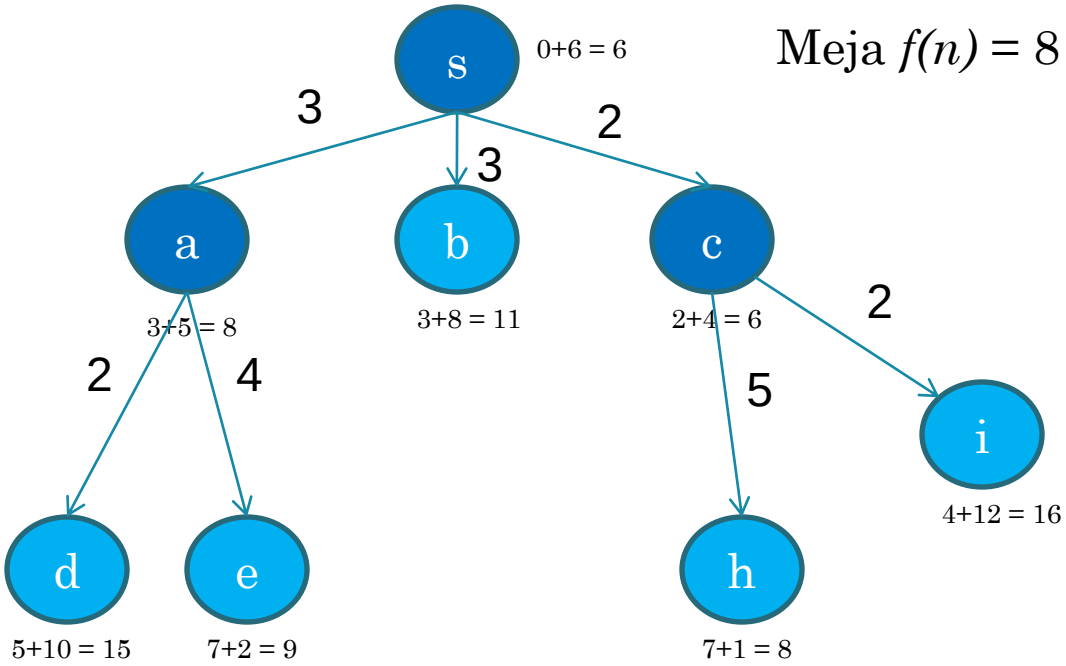
PRIMER – IDA* (8/13)



Meja $f(n) = 8$

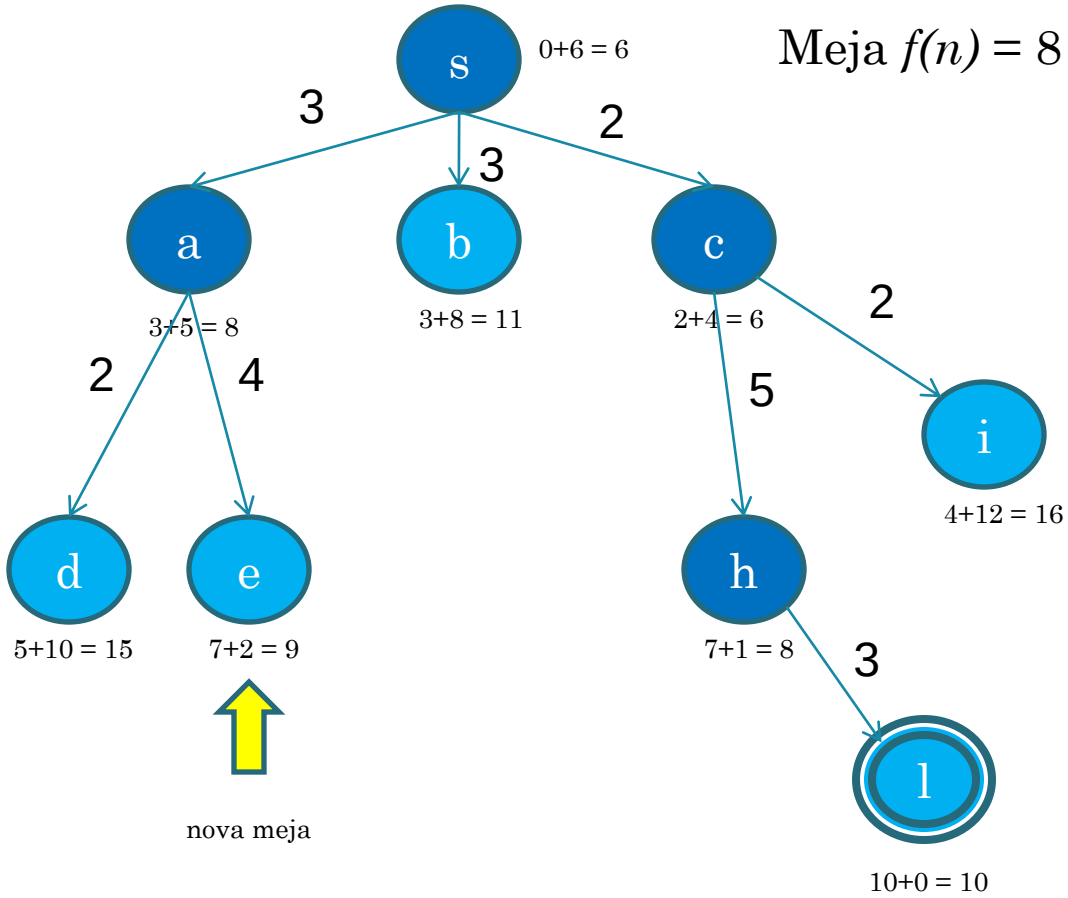
	s	a	b	c	d	e	f	g	h	i	j	k	l
h(n)	6	5	8	4	10	2	8	0	1	12	12	0	0

PRIMER – IDA* (9/13)



	s	a	b	c	d	e	f	g	h	i	j	k	l
h(n)	6	5	8	4	10	2	8	0	1	12	12	0	0

PRIMER – IDA* (10/13)



	s	a	b	c	d	e	f	g	h	i	j	k	l
h(n)	6	5	8	4	10	2	8	0	1	12	12	0	0

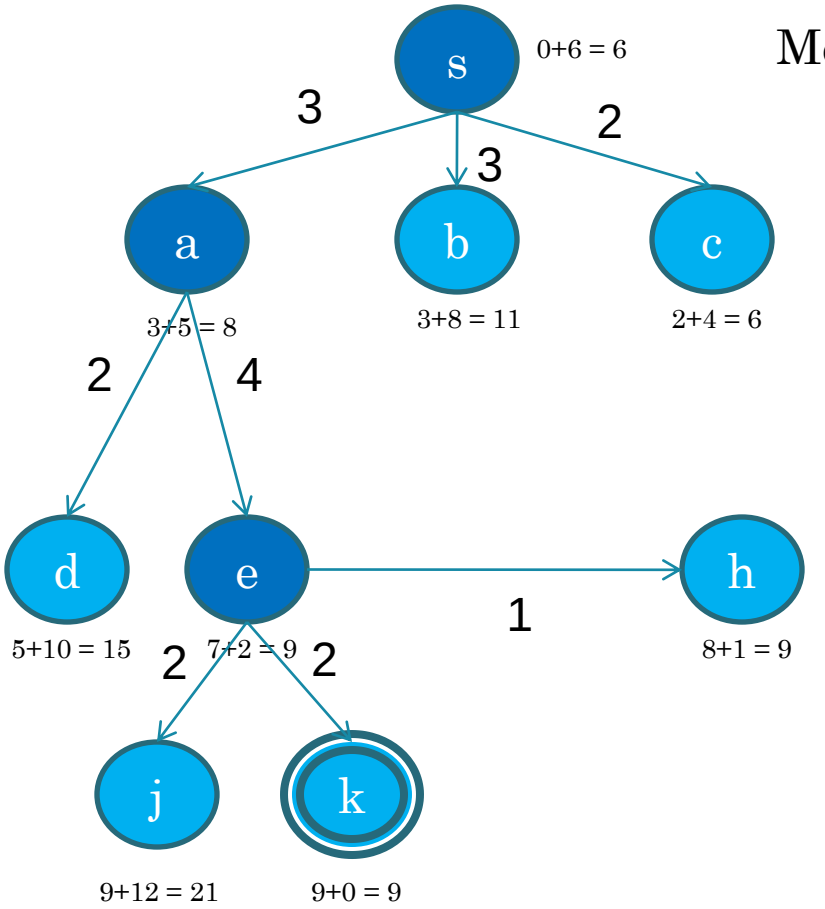
PRIMER – IDA* (11/13)

0+6 = 6

Meja $f(n) = 9$

	s	a	b	c	d	e	f	g	h	i	j	k	l
h(n)	6	5	8	4	10	2	8	0	1	12	12	0	0

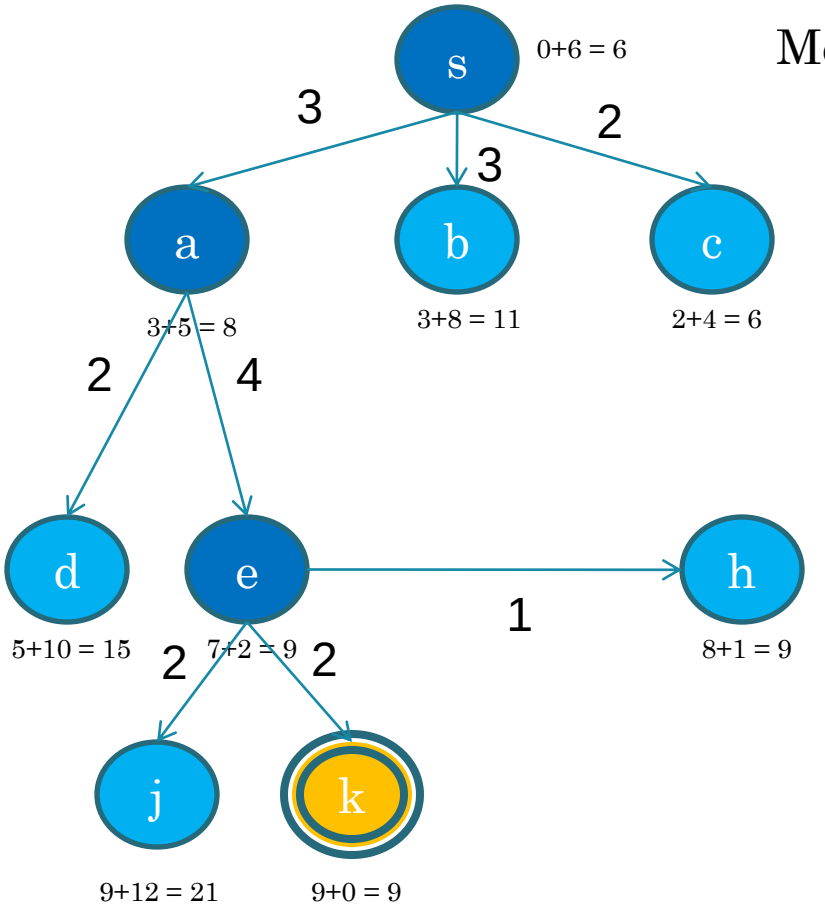
PRIMER – IDA* (12/13)



Meja $f(n) = 9$

	s	a	b	c	d	e	f	g	h	i	j	k	l
h(n)	6	5	8	4	10	2	8	0	1	12	12	0	0

PRIMER – IDA* (13/13)



Meja $f(n) = 9$

	s	a	b	c	d	e	f	g	h	i	j	k	l
h(n)	6	5	8	4	10	2	8	0	1	12	12	0	0

SAMOSTOJNO DELO

Za spodnji graf z začetnim vozliščem *a* in s končnima vozliščema *g* in *h* določi zaporedje razvijanja vozlišč, če uporabljamo IDA* s hevristično oceno iz spodnje tabele:

a	b	c	d	e	f	g	h
8	2	4	4	9	12	0	0