

ФАКУЛТЕТ ЗА ИНФОРМАТИЧКИ НАУКИ И КОМПЈУТЕРСКО ИНЖЕНЕРСТВО

Неинформирано пребарување



Неинформирано пребарување

1. По прва широчина (прво по широчина)
2. По прва длабочина (прво по длабочина)
3. Со изедначена (унифицирана) цена
4. Со ограничена длабочина
5. Со итеративно одење во длабочина
6. Двонасочно

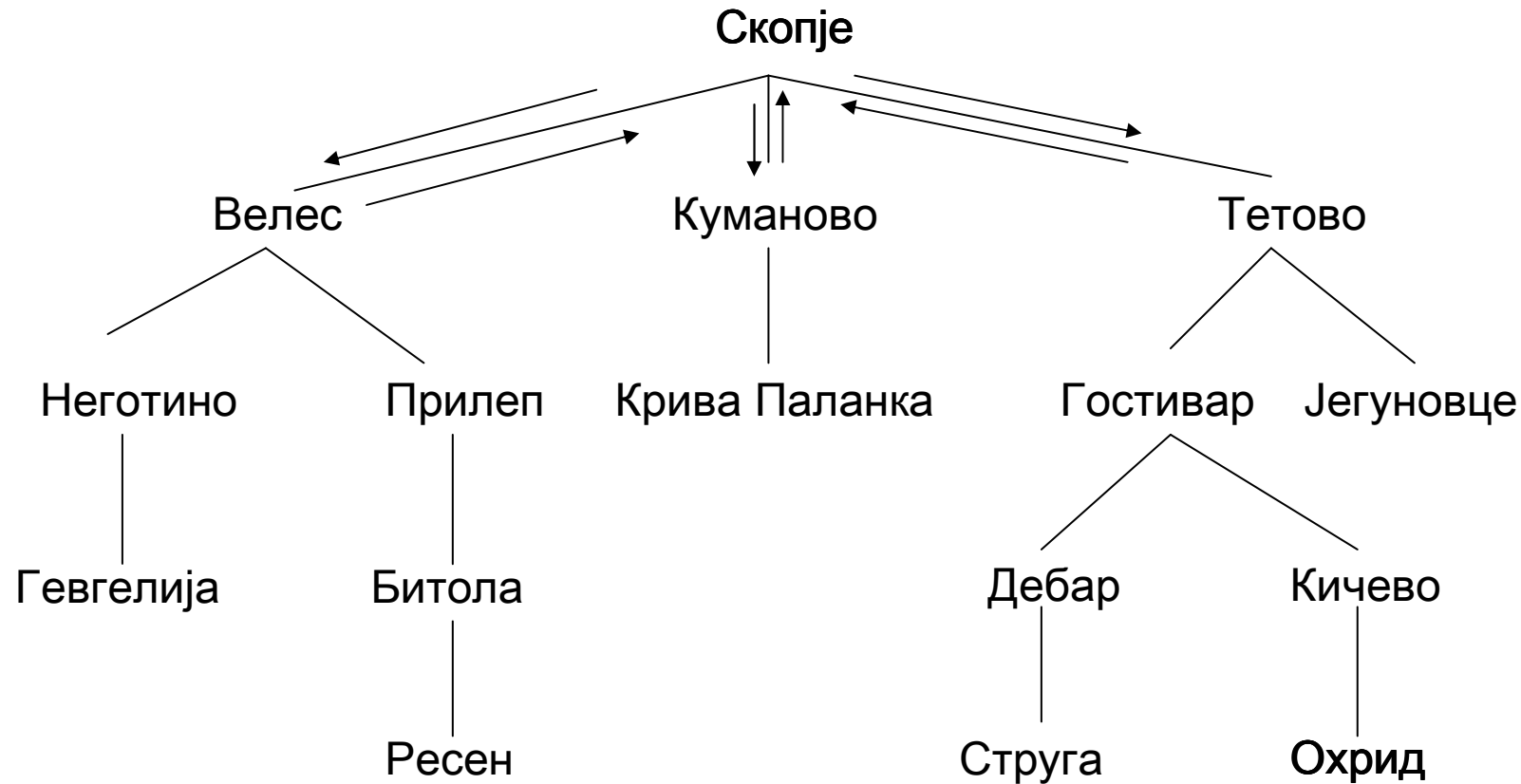
Пребарување по прва широчина (Breadth-first search)

- Го развива најплиткиот јазел, т.е оној јазел што е прв во листата неразложени јазли
- Новоразложените јазли се додаваат на крајот
- Базиран на FIFO редицата

Својства

- Ако b е конечно, алгоритмот е комплетен
- Временска комплексност (сложеност):
$$1+b+b^2+b^3+\dots +b^{d-1} = O(b^d)$$
- Просторна комплексност: $O(b^d)$
- Ако лаците имаат цена, тогаш не го наоѓа оптималното решение, но ако се претпостави дека цената на чекорите е еднаква, тогаш решението е оптимално.

Пример за наоѓање на патот од Скопје до Охрид



Како изгледа листата?

Јазли што се во листата за
разложување

- Скопје,
- Велес, Куманово, Тетово,
- Неготино, Прилеп,
- Крива Паланка,
- Гостивар, Јегуновце,
- Гевгелија,
- Битола,
- Крива Паланка (неразложлив)
- Дебар, Кичево
- Гевгелија (неразложлив)
- Ресен,
- Струга,
- Охрид.

Листа на целосно разложените
јазли

- Скопје,
- Велес,
- Куманово,
- Тетово,
- Неготино,
- Прилеп,
- Крива Паланка,
- Гостивар,
- Гевгелија,
- Битола,
- Дебар,

Мерки на комплексноста при примената на алгоритмот по прва широчина:
b = 3; d = 4, m = 5

Проценка на Расел и Норвиг за влијанието на длабочината на дрвото врз времето и меморијата *

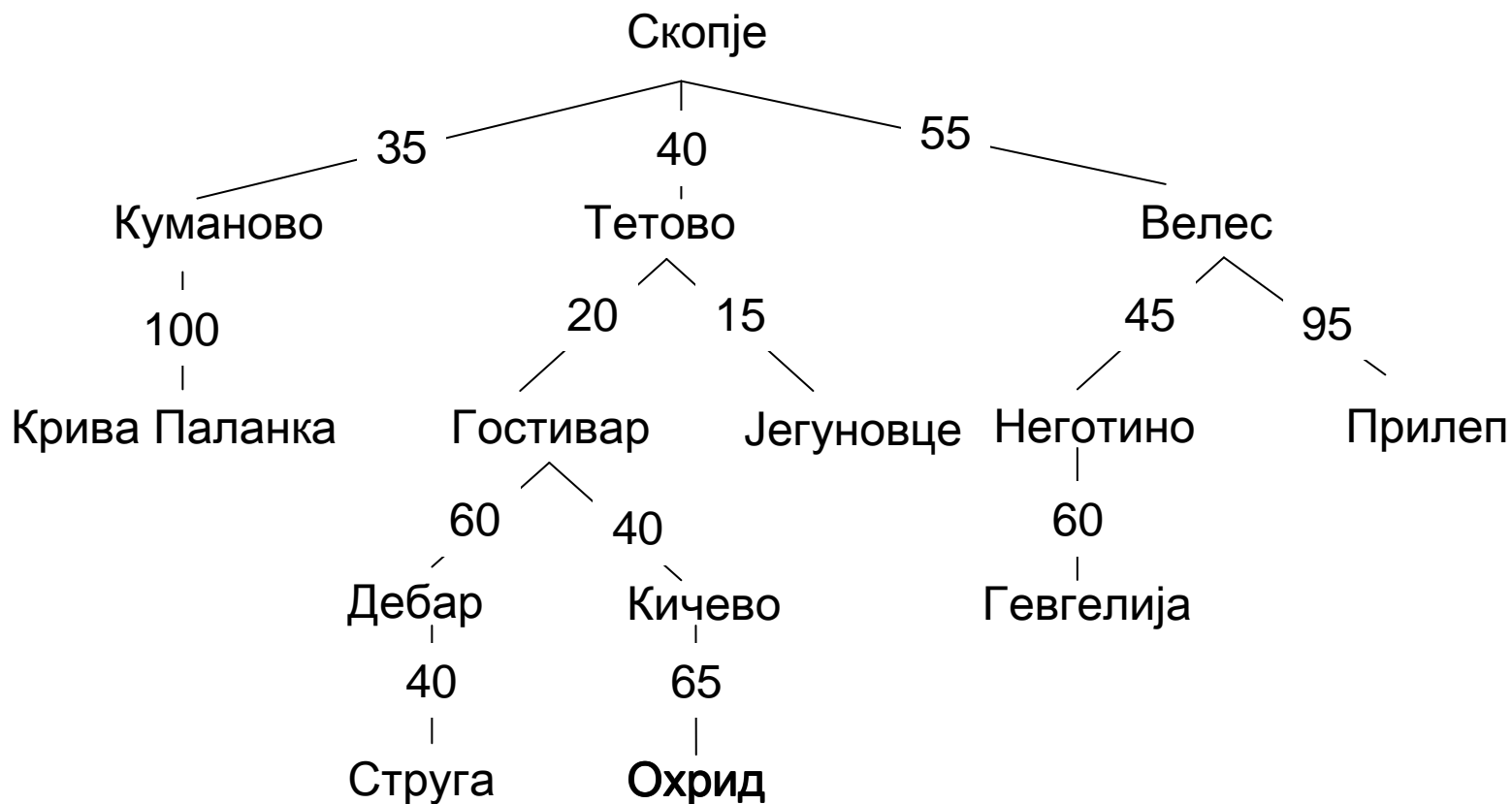
Depth	Nodes	Time	Memory
2	1100	.11 seconds	1 megabyte
4	111,100	11 seconds	106 megabytes
6	10^7	19 minutes	10 gigabytes
8	10^9	31 hours	1 terabytes
10	10^{11}	129 days	101 terabytes
12	10^{13}	35 years	10 petabytes
14	10^{15}	3,523 years	1 exabyte

Figure 3.11 Time and memory requirements for breadth-first search. The numbers shown assume branching factor $b = 10$; 10,000 nodes/second; 1000 bytes/node.

Пребарување со изедначена цена (Uniform-cost search)

- Го развива „најефтиниот“ неразложен јазел
- Новоразложените јазли се додаваат на крајот, а цената на сите поминати јазли се сумира
- Комплетен, ако цената на секој чекор надминува некоја однапред дефинирана цена ϵ
- Временската и просторната комплексност се помали од $O(b^{1+\lceil C^*/\epsilon \rceil})$, каде C^* е цената на оптималното решение, а цената на секој чекор надминува ϵ
- $O(b^{1+\lceil C^*/\epsilon \rceil})$ може многу да го надминува b^d

Илустрација на пребарувањето со изедначена цена



Мерки на комплексноста при примената на алгоритмот за пребарување по изедначена цена: **b = 3; d = 4, m = 4**

Како е формирано дрвото (1)?

- Критериум: минимална вкупна цена.
- Ако постојат два збира со еднаква вредност, се избира јазелот со помала длабочина.

Куманово: $c_1 = c(\text{Скопје, Куманово}) = 35$

Тетово: $c_2 = c(\text{Скопје, Тетово}) = 40$

Велес: $c_3 = c(\text{Скопје, Велес}) = 55$

Јегуновце: $c_4 = c_2 + c(\text{Тетово, Јегуновце}) = 55$

Како е формирано дрвото (2)?

Гостивар: $c_5 = c_2 + c(\text{Тетово, Гостивар}) = 60$

Неготино: $c_6 = c_3 + c(\text{Велес, Неготино}) = 100$

Кичево: $c_7 = c_5 + c(\text{Гостивар, Кичево}) = 100$

Дебар: $c_8 = c_5 + c(\text{Гостивар, Дебар}) = 120$

Крива Паланка:

$c_9 = c_1 + c(\text{Куманово, Крива Паланка}) = 135$

Како е формирано дрвото (3)?

Прилеп: $c_{10} = c_3 + c(\text{Велес, Прилеп}) = 150$

Струга: $c_{11} = c_8 + c(\text{Дебар, Струга}) = 160$

Гевгелија:

$c_{12} = c_6 + c(\text{Неготино, Гевгелија}) = 160$

Охрид: $c_{13} = c_7 + c(\text{Кичево, Охрид}) = 165$

- За да може постојано да се разложува токму тој јазел што има најмала збирна цена, неопходно е да се меморираат сите претходни патишта.
- Тоа е основната причина за големата комплексност.

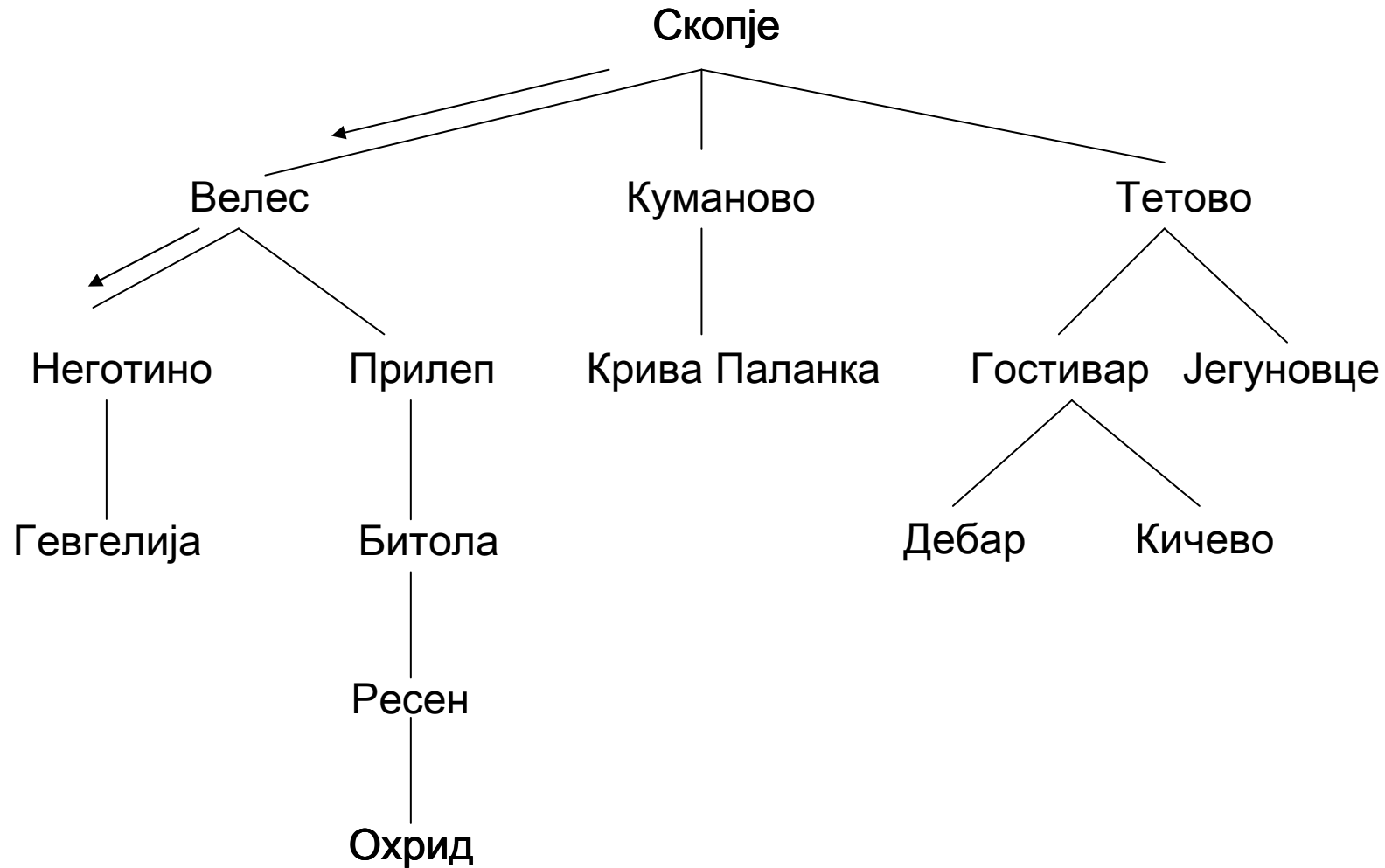
Пребарување по прва длабочина (Depth-first search)

- Го развива најдлабокиот јазел
- Новоразложените јазли се ставаат на почетокот на листата
- Базиран на LIFO редицата

Својства

- Алгоритамот не е комплетен, доколку во просторот има јамки
- Временска комплексност: $O(b^m)$
- Просторна комплексност: $O(bm)$
- Ако лаците имаат цена, тогаш не го наоѓа оптималниот пат

Пример за наоѓање на патот од Скопје до Охрид



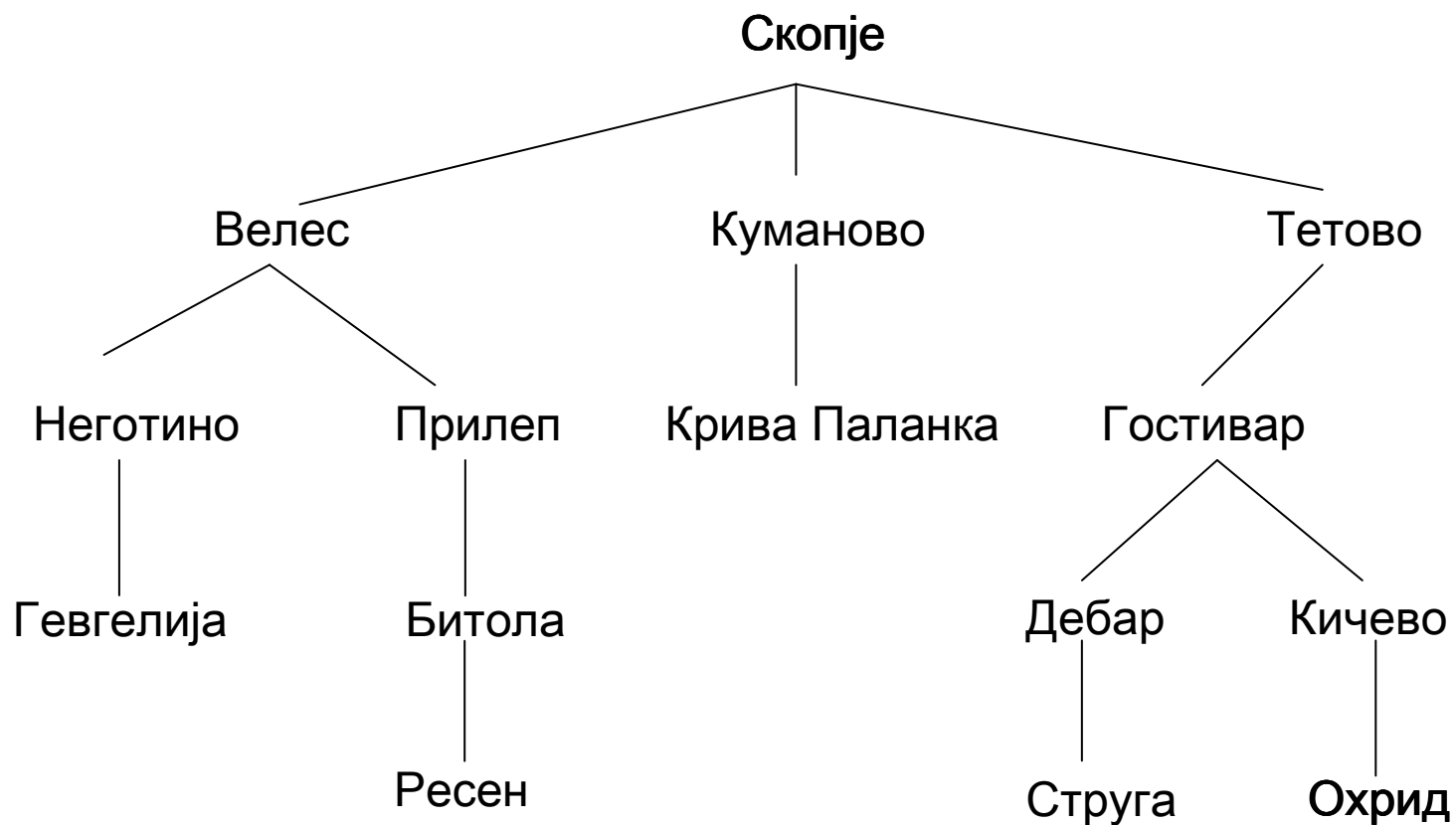
Пребарување со ограничена длабочина (Depth-limited search)

- Може да се изедначи со пребарувањето по прва длабочина при што длабочината d не надминува однапред дефинирана вредност l
- l се вика дијаметар на просторот за пребарување

Својства на пребарувањето со ограничена длабочина

- Некомплетен и не гарантира оптималност
- Временска комплексност: $O(b^l)$
- Просторна комплексност: $O(bl)$
- Пребарувањето може да заврши заради тоа што:
 - Не постои решение
 - Отсекувањето е превисоко, т.е. не постои решение со толку мала длабочина

Услов: $d < 4$

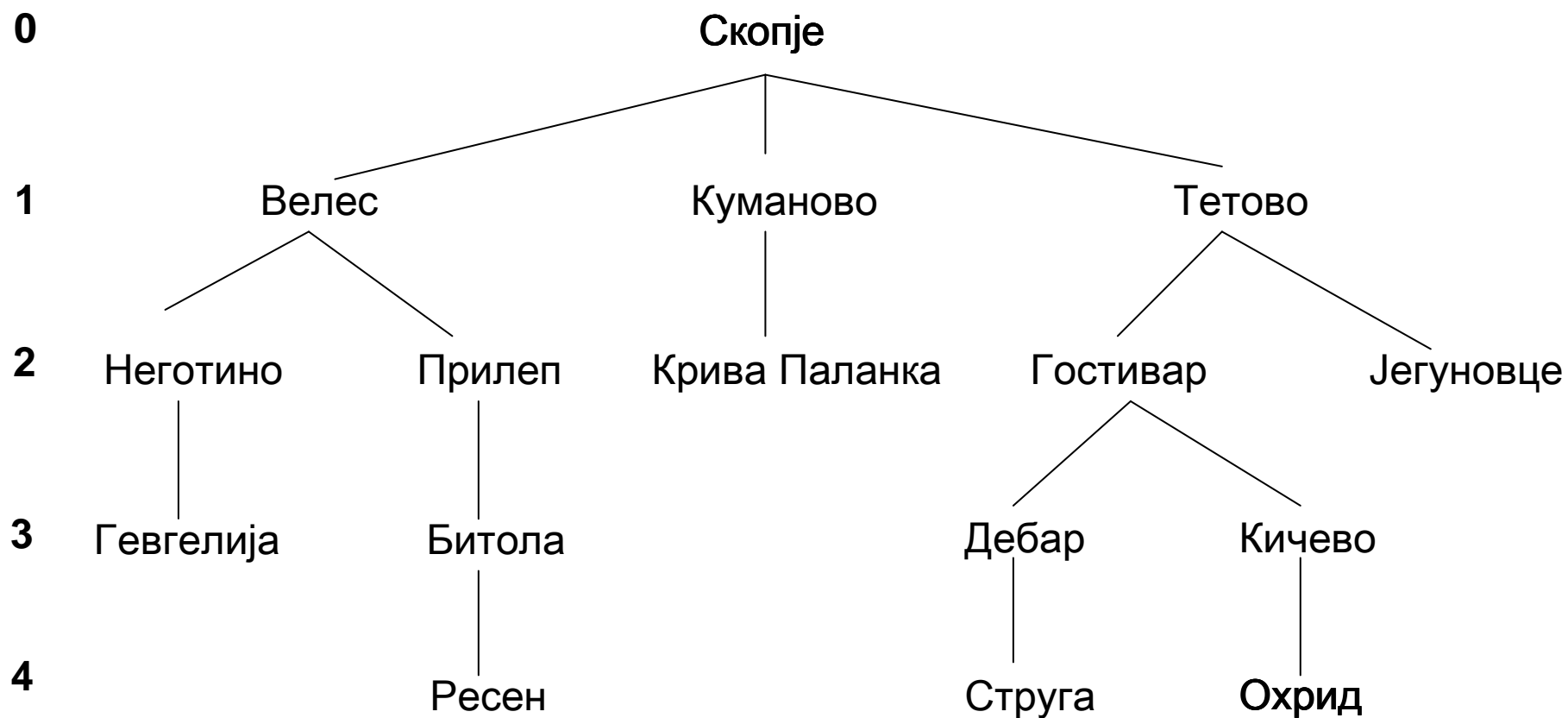


Забелешка: За помала длабочина, алгоритмот не наоѓа решение

Пребарување со постепено зголемување на длабочината (iterative deepening depth-first search)

- Пребарување со ограничена длабочина, при што длабочината постепено расте од 0 до d
- Разложувањето на јазлите од повисоките нивоа е повеќекратно, но сепак, овој алгоритам е b пати побрз од пребарувањето по прва ширина
- Комплетен и го наоѓа оптималното решение, доколку постои

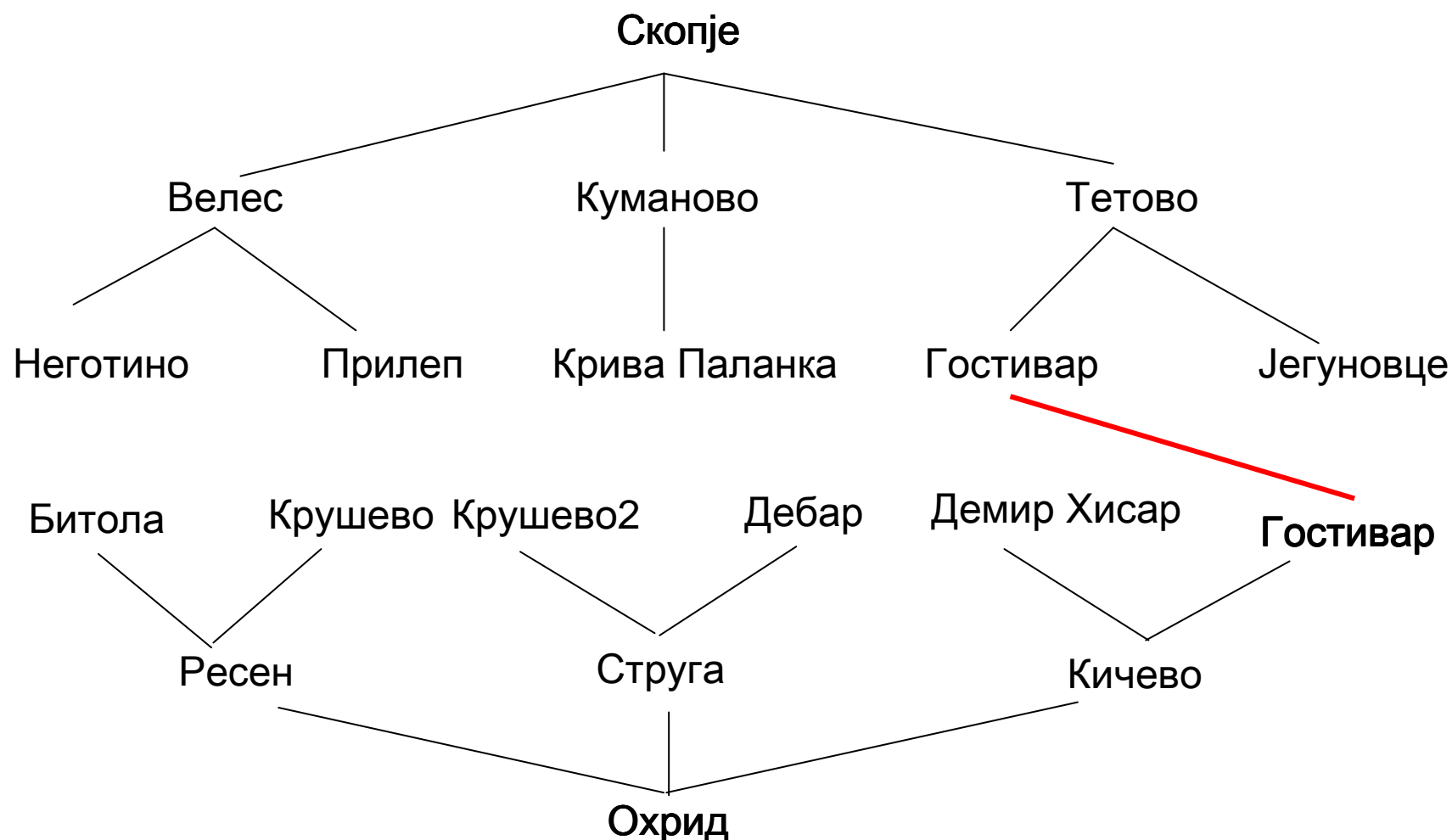
Постепено зголемување на длабочината



Двонасочно пребарување (bidirectional search)

- Најбрз неинформиран алгоритам, но неприменлив за секој проблем
- Основна мотивација: $2 \cdot b^{d/2} < b^d$
- Пребарување по прва широчина од почетокот кон средината и од крајот кон средината, до средба на половина пат, доколку таа постои
- Постојани проверки дали средбата е воспоставена
- Основни параметри: $\text{Pred}(n)$ и $\text{Succ}(n)$
- Комплексност $O(b^{d/2})$

Двонасочно пребарување

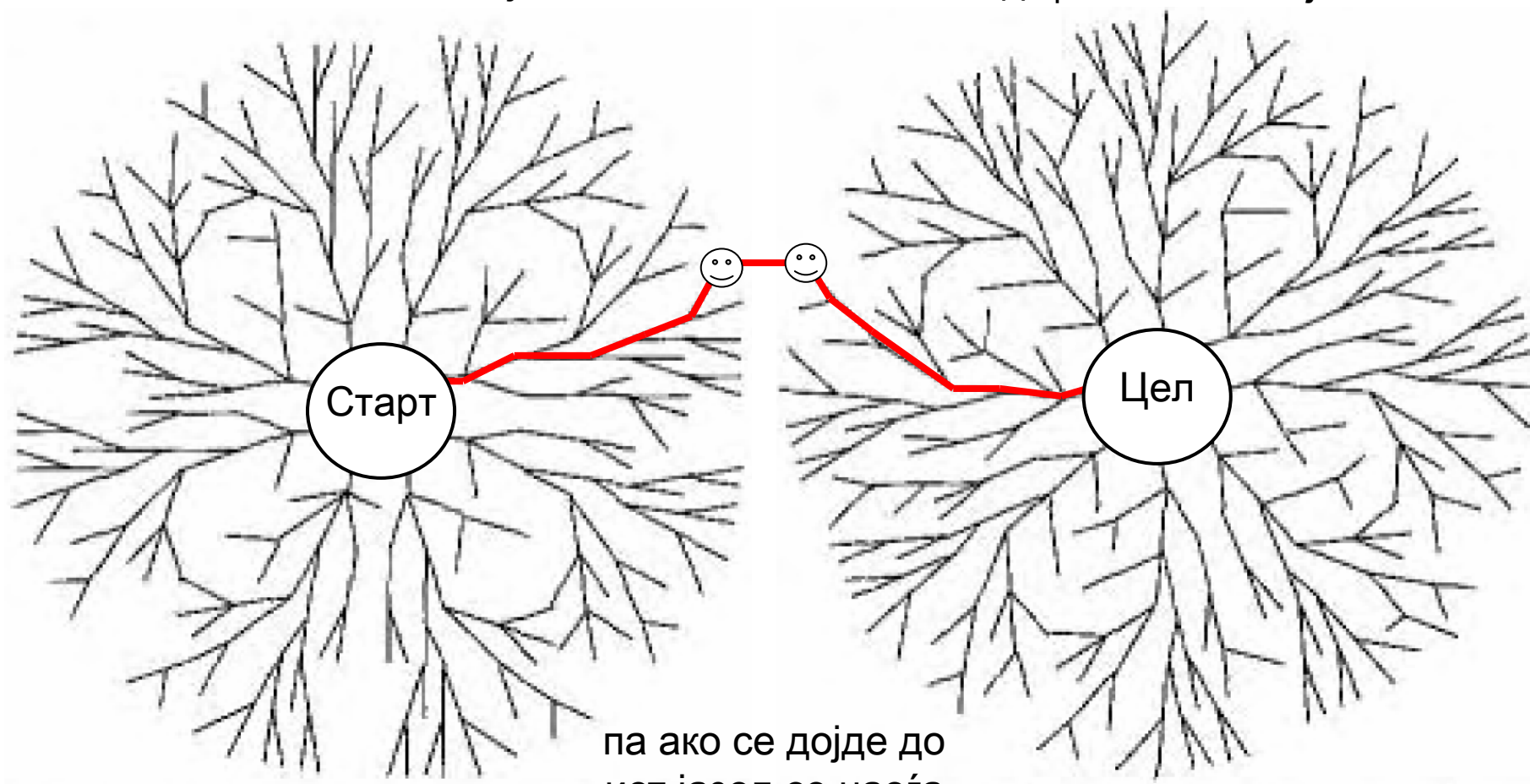


Забелешка: Ако има цена, може да се примени алгоритмот за унифицирана цена

Двонасочно пребарување

напред
од почетната состојба

назад
од целната состојба



па ако се дојде до
ист јазол се наоѓа
решението

Споредба на стратегиите за слепо пребарување

Критериум	breadth-first	uniform-cost	depth-first	depth-limited	iterative deepening	bidirectional
КОМПЛЕТНОСТ	да	да	не	да, ако $l \geq d$	да	да
време	$O(b^d)$	$O(b^d)$	$O(b^m)$	$O(b^l)$	$O(b^d)$	$O(b^{d/2})$
меморија	$O(b^d)$	$O(b^d)$	$O(bm)$	$O(bl)$	$O(bd)$	$O(b^{d/2})$
ОПТИМАЛНОСТ	да	да	не	не	да	да

b е фактор на разгранување, d е длабочината на решението,
 m е максималната длабочина на стеблото, l е ограничувањето на длабочината

Користена литература

Првенствено книгата:

- Artificial Intelligence, A Modern Approach
2nd edition, Russel and Norvig, страна 89

НО ИСТО ТАКА И СТАТИИТЕ КОИ ПОТЕКНУВААТ
ОД ОСНОВНАТА СТАТИЈА ЗА Search
Algorithms во англиската Википедија



Прашања?