Факултет за електротехника и
информарички технологии - Скопје

Решавање на проблеми со помош на пребарување
Пребарување по длабочина

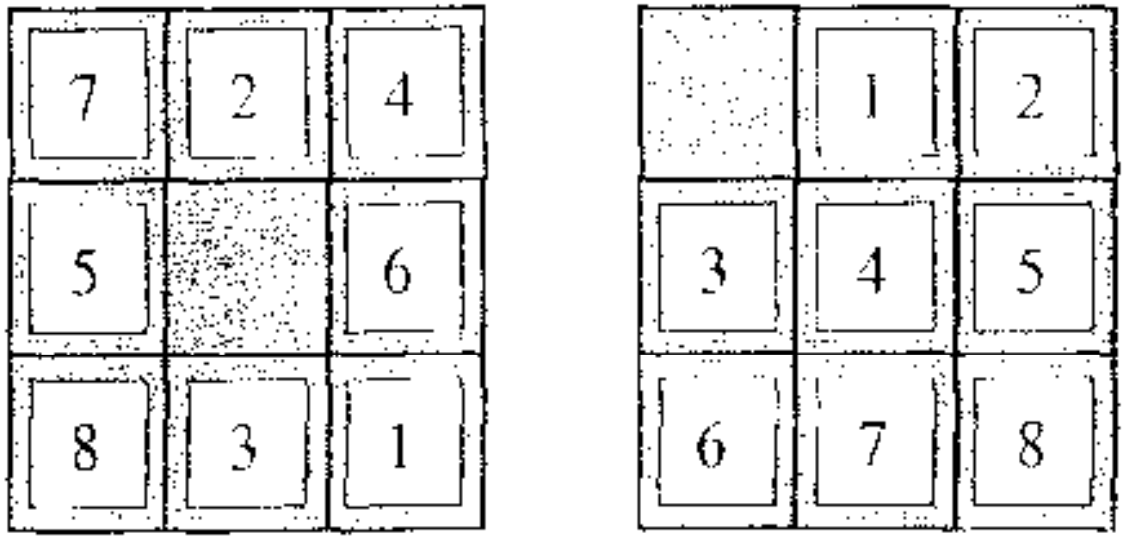
Изработил:
Билјана Милева - Бошкоска

Скопје, 18
22.10.2007

Видови на проблеми

- Пристапот на решавање-на-проблеми се аплицира на широко поле од работни околини. Ќе покажеме само неколку познати проблеми кои ги класифицираме во следните две категории: едноставни проблеми (*toy* problems) и реални проблеми (*real-world* problems).
- **Едноставниот проблем** е наменет да илустрира или изврши различни методи за решавање-на-проблемот. За него може да биде дефиниран концизен, егзактен опис. Тоа значи дека научниците може да го користат за споредба на перформансите на алгоритмите.
- **Реален проблем** е оној чие решение им е потребно на луѓето. За нив, не постои единствено дефинирање, туку се стремиме да дадеме општа формулација за истиот.

Едноставен проблем - Слагалица со 8 плочки



Стартна состојба и целна состојба

Едноставен проблем - Слагалица со 8 плочки

- **Слагалица со 8 плочки –** се состои од 3x3 табла на која се поставени осум нумерирани плочки и едно празно место за плочка. Плочката кој а е сосед на празното место може да се придвижи во празното место. Целта е да се постигне определена целна состојба, како на примерот даден на сликат. Стандардната формулација на проблемот е следна:
- **Состојби:** Описот на состојбата ја определува локацијата на секоја од осумте плочки и на празното место во едно од деветте места
- **Иницијална состојба:** Било која состојба може да се дефинира како иницијална состојба.
- **Функција на следбеник:** Таа генерира дозволени состојби кои се резултат од обидот на четирите акции (празното место се придвижува *лево, десно, надолу, нагоре*).
- **Тест за постигнување на целта:** Тој проверува дали моменталната состојба се совпаѓа со бараната цел.
- **Цена на патеката:** Секој чекор чини 1, така што цената на патот е бројот на чекори во патеката.

Реален проблем - пронаоѓање на патека

- Проблемот на **пронаоѓање на патека** се дефинира во помош на специфицирани локации и премин помеѓу нив по должината на врските кои ги соединуваат. Алгоритмите на **пронаоѓање на патека** се користат во најразлични апликации, како што се рутирање во компјутерски мрежи, планирање на воени операции, системи на планирање на авионски патувања итн. Обично е многу комплексно да се специфицираат овие проблеми. Нека разгледаме поедноставен пример на проблем на авионско патување на следниот начин:
- **Состојби:** секоја состојба е претставена со лпкација (на пример аеродром) и моментално време.
- **Иницијална состојба:** таа е претставена со проблемот кој се решава.
- **Функција на следбеник:** таа ги враќа состојбите кои резултираат од користење на некој закажан лет (кој евентуално би бил дополнително специфициран со класа на столче и локација), кој заминува подоцна од моменталното време од моменталниот аеродром до некој друг аеродром.
- **Тест за постигнување на целта:** дали сме стигнале до целта во некое претходно специфицирано време?
- **Цена на патеката:** таа зависи од паричната единица, времето на лет, процедури за царина и имиграција, квалитет на седиштето, време во денот, тип на авион, награди за frequent-flyer итн.

Барање на решение

- Штом дефиниравме неколку проблеми, следно треба да најдеме решение за нив. Тоа се прави преку пребарување во просторот на состојби.
- Понатаму ќе се занимаваме со неколку технологии на пребарување кои користат експлицитно **дрво на пребарување** кое се генерира од иницијалната состојба и функцијата на следбеник кои заедно го сочинуваат просторот на состојби. Во општ случај, може да имаме *граф за* пребарување наместо дрво за пребарување кога до иста состојба може да дојдеме со користење на повеќе патеки.

Стратегии за униформно пребарување

- Започнуваме со пет **стратегии за униформно пребарување** познато и под името **слепо пребарување.** Ќе разгледаме неколку стратегии од овој тип на пребарување.
- Овој термин означува дека немаме дополнителни информации за состојбите надвор од оние дефинириани во проблемот. Овие стратегии единствено може да генерираат наследник и да извршат разликување помеѓу целна состојба и меѓу состојба. Оние стратегии кои знаат дали една меѓу состојба е „понадежна“ од колку некоја друга меѓу состојба се нарекуваат **стратегии на информирано пребарување или хеуристичко пребарување.**

Терминологија

Започнуваме со терминологијата која ќе ја користиме при објаснување на алгоритмите за пребарување.

Да започнеме со Посетена состојба наспроти Проширена. За една состојба велиме дека е посетена, ако кога патеката ја достигне таа состојба (односно јазелот кој ја опишува таа состојба) , таа состојба се додаде на Q (низа од јазли кои ја сочинуваат патеката). Значи ако состојбата е било каде, во било кој јазел Q, тогаш таа состојба велиме дека е посетена. Или, ние сме го посетиле јазелот, сме го сместиле во Q, но се уште не сме го разгледале внимателно.

Состојбата M е проширена ако кога патеката до таа состојба е симната од Q. Во тој момент, наследниците на M се посетуваат и патеките до нив се додаваат во Q.

Во принцип, една состојба може да биде посетена повеќе пати. Штом еден јазел е проширен еднаш, немаме потреба да го прошируваме повторно.
Многу е битно да се прави разлика помеѓу овие два термини.

Повеќе за Посетените листи, ке слушнете на предавање.

По длабочина:

1. Земете го првиот елемент од Q
2. Додадете проширување на патеката пред Q

По широчина:

1. Земете го првиот елемент од Q
2. Додадете проширување на крајот пред Q

Прв најдобар:

1. Земете го „најдобриот“ (мерено по хеуристичката вредност на состојбата) елемент од Q
2. Додадете проширување на патеката било каде во Q (може да е поефикасно да го чуваме Q подреден на некаков начин за да може полесно да се најде „најдобриот“ елемент)

Пребарувањето на Прв најдобар (исто така познат под името „алчен“) е хеурисчко (информирано) пребарување кое ја користи вредноста на дефинираната хеуристичкна функција за да го води пребарувањето. Тоа нема да гарантира пронаоѓање на „најдобрата“ патека, на пример, најкратката патека до целта. Хеуристиката се користи со надеж дека ќе не однесе до најбрзо завршување на пребарувањето или до релативно добра состојба.

Пребарувањето Прв најдобар може да се имплементира на следниот начин: изберете ја „најдобрата“ патека (измерена со некоја хеуристичка вредност) од сите Q и додадете проширување некаде во Q. Па во секој чекор, ние ќе го разгледуваме јазелот со најдобра хеуристичка вредност.

Треба да се забележи дека во најлош случај, ова пребарување ќе ги прегледа сите патеки кои ги прегледуваат и алгоритмите по длабочина и по широчина, но редот на прегледување ќе биде различен, а со тоа и конечната патека ќе биде различна во општ случај.

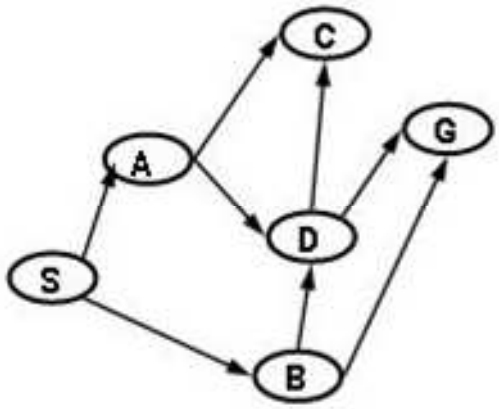
Пребарување по длабочина

Започнуваме со пребарување по длабочина, при што ќе користиме листи на посетени јазли.

Во табелата дадена е содржината на Q и на листата на посетени јазли во текот на секој циклус на пребарувањето. Јазлите во Q се напишани во обратен редослед, а плавата боја означува нов додаден јазел (патека). На десната страна даден е графот кој го пребаруваме и на кој ќе ја означуваме секоја состојба која сме ја посетиле во секој чекор.

Земете го првиот елемент од Q. Додадете екстензија на патеката пред Q.

	Q	**Посетени**
1		
2		
3		
4		
5		

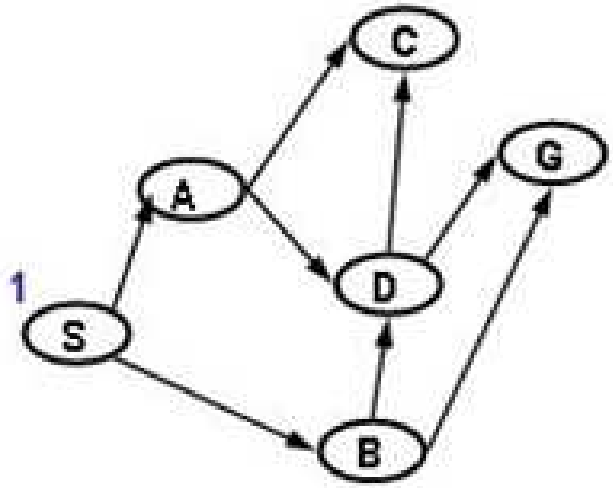


Додадените патеки се означени со плаво.
Патеките ги претставуваме во обратен редослед. Состојбата на јазелот е првиот елемент од листата.

Првиот чекор е да се иницијализира Q со еден јазел кој одговара на почетната состојба (во овој случај тоа е S) и да се иницијализира листата на посетени јазли со почетниот јазел.

Го земаме првиот елемент од Q. Дадаваме проширување на патеката пред Q.

	Q	Посетени
1	(S)	S
2		
3		
4		
5		



Додадените патеки се означени со плаво.
Патеките ги претставуваме во обратен редослед. Состојбата на јазелот е првиот елемент од листата.

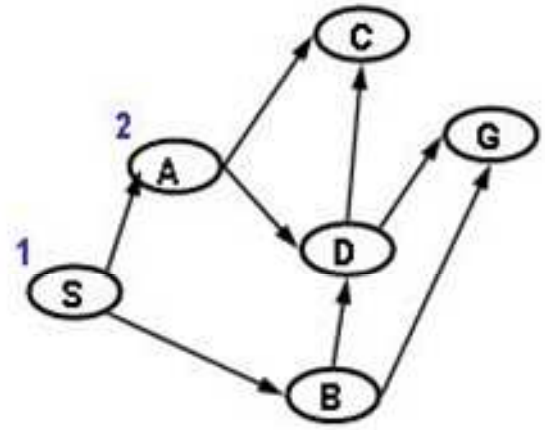
Го избираме првиот елемент од Q, кој е во суштина иницијален јазел, го отстрануваме од Q, ја прошируваме неговата патека кон неговите следни состојби (доколку тие не биле претходно посетени) и ги додаваме јазлите пред Q. Исто така овие состојби кои кореспондираат на овие нови јазли ги додаваме во листата на посетени јазли. Ја добиваме состојбата претставена во редот 2 на табелата.

Забележете дека јазлите кои следат, би можеле да се додадат во Q и во друг редослед. Тоа би било абсолутно точно. Обично ќе ги додаваме јазлите во Q по азбучен редослед, доколку не е специфициран некаков друг алгоритам за додавање на јазлите. Ова е сосема произволно решение.

Потоа го бираме првиот јазел од Q, чија состојба е A, и го повторуваме процесот, проширувајќи ги јазлите кон C и D и поставувајќи ги пред Q.

Го земаме првиот елемент од Q. Дадаваме проширување на патеката пред Q.

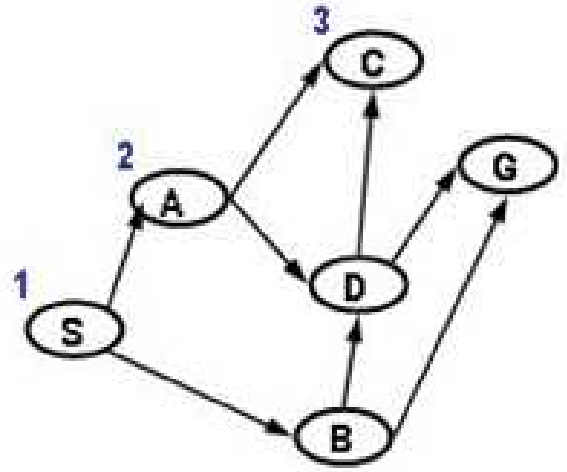
	Q	Посетени
1	**(S)**	**S**
2	**(AS)(BS)**	**A, B, S**
3		
4		
5		



Додадените патеки се означени со плаво.
Патеките ги претставуваме во обратен редослед. Состојбата на јазелот е првиот елемент од листата.

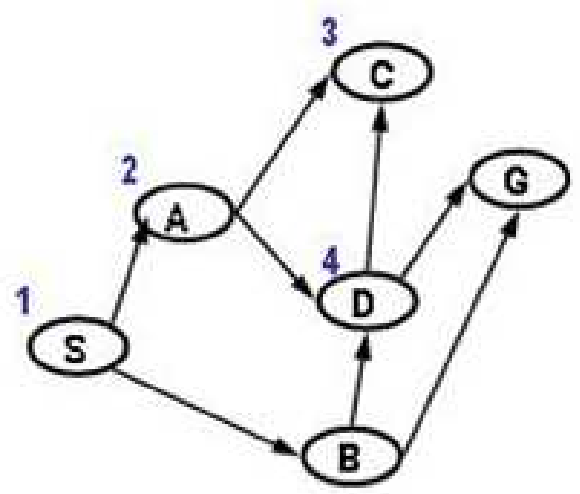
Го земаме првиот јазел, чија состојба е C, и при тоа забележуваме дека оваа состојба нема наследници, па соодветно, нема нови јазли за додавање.

	Q	Посетени
1	(S)	S
2	(AS)(BS)	A, B, S
3	(CAS)(DAS)(BS)	C, D, B, A, S
4		
5		



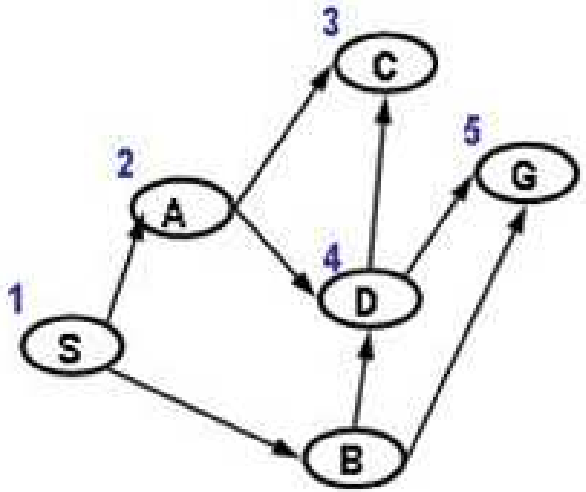
Го бираме првиот јазел од Q, чија состојба е D, и разгледуваме проширување кон состојбите C и G, но C е веќе на листата на посетени јазли, па не го додаваме тоа проширување. Ја додаваме патеката кон G пред Q.

	Q	Посетени
1	(S)	S
2	(AS)(BS)	A, B, S
3	(CAS)(DAS)(BS)	C, D, B, A, S
4	(DAS)(BS)	C, D, B, A, S
5		



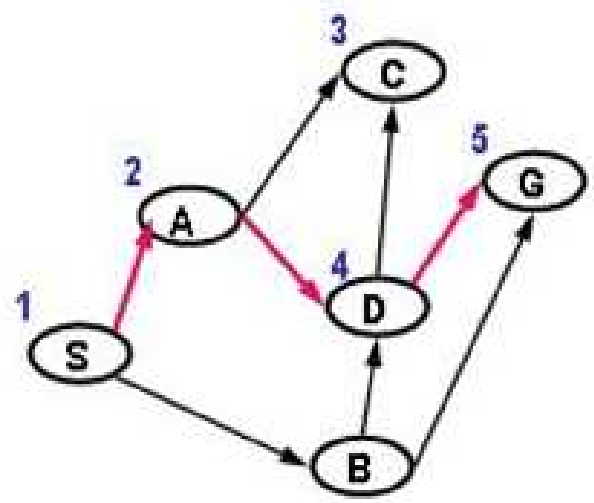
Го бираме првиот јазел од Q, чија состојба е G, т.е. нашата цел, па овдека застануваме и ја враќаме патеката.

	Q	Посетени
1	(S)	S
2	(AS)(BS)	A, B, S
3	(CAS)(DAS)(BS)	C, D, B, A, S
4	(DAS)(BS)	C, D, B, A, S
5	(GDAS)(BS)	G, C, D, B, A, S

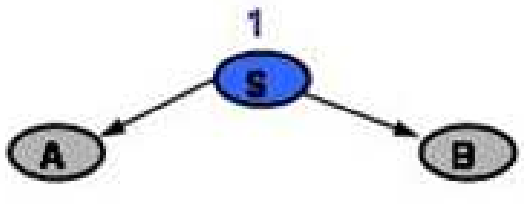


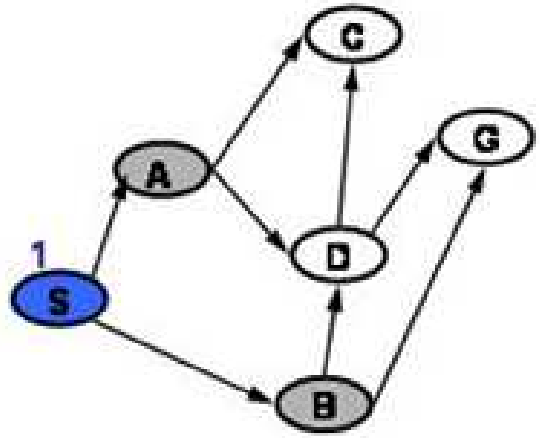
Последната патека која сме ја вратиле назад од S кон A потоа кон D и кон G.

	Q	Посетени
1	(S)	S
2	(AS)(BS)	A, B, S
3	(CAS)(DAS)(BS)	C, D, B, A, S
4	(DAS)(BS)	C, D, B, A, S
5	(GDAS) (BS)	G, C, D, B, A, S



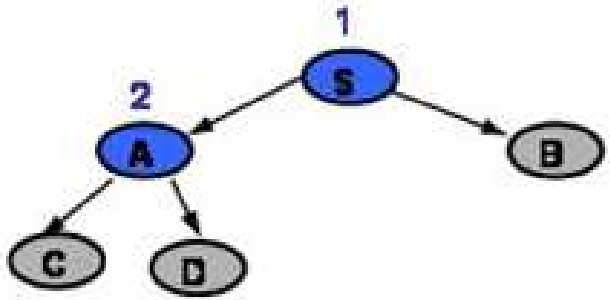
Скицирањето на содржината на Q може да стане малку монотоно, иако секогаш дозволува да се следат перформансите на алгоритмот во детали. Друг начин за да се визуелизираат едноставните пребарувања е да се нацрта дрвото кое се пребарува, како што е покажано овде, покажувајќи го резултатот на првата екпанзија на примерот кој го разгледувавме.

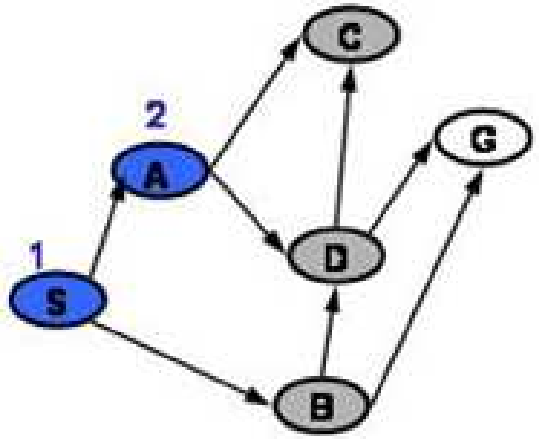




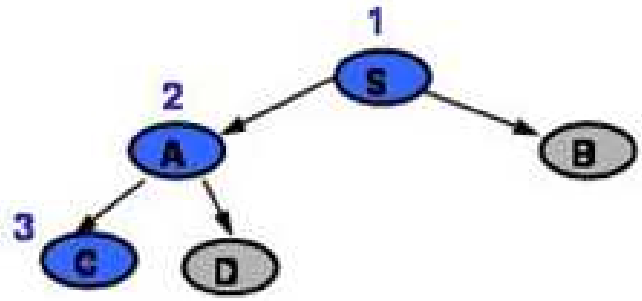
Броевите го покажуваат редоследот на извлекување од Q (проширување)
Темно плавите = посетени и проширени
Светло сивите = посетени

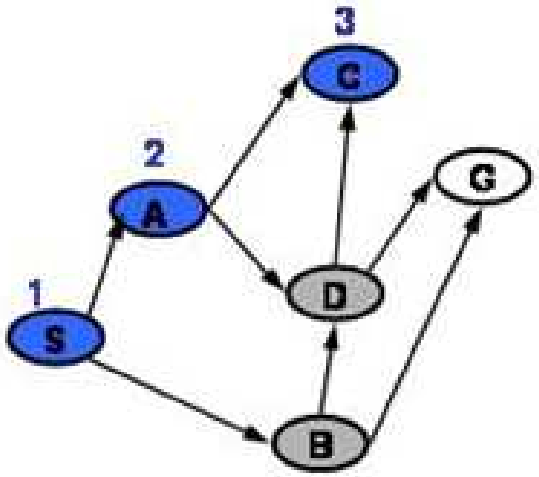
На овој начин, го воведовме лево кон десно одлучувањето кои јазли да се прошират – ова е чисто произволно. Кораспондира точно со произволната одлука за кој јазел да се додаде прво во Q. Одлучуваме да го прошириме јазелот чија состојба е A, и која завршува со посета на C и D.



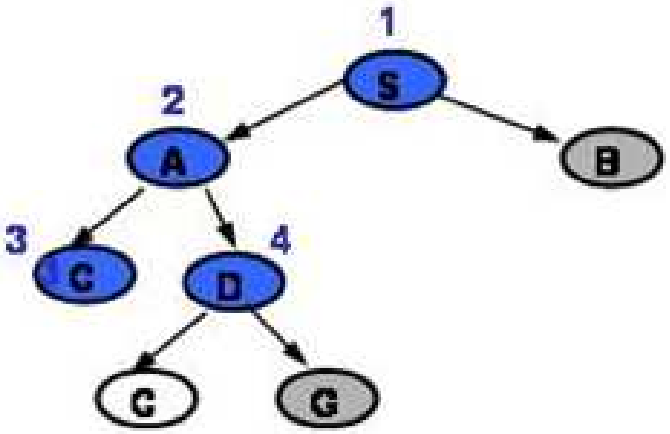


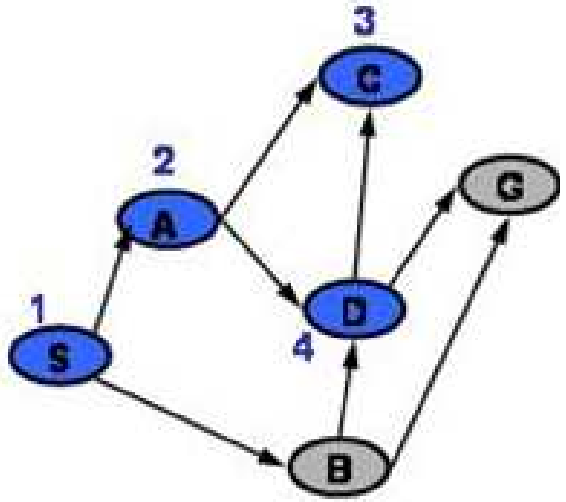
Сега го прошируваме јазелот кој кореспондира со C, кој нема наследници, па не можеме да одиме подлабоко. Во овој момент, се оди **кон назад или се враќаме по истиот пат назад** до јазелот родител и го прошируваме следниот непроширен наследник на родителот. Ако нема такви на даденото ниво, продолжуваме наназад до родителот се додека не наидеме на некој јазел кој не е проширен. Прогласуваме неуспех (failure) ако неможеме да најдеме јазел кој не е проширен. Во нашиот случај ги наоѓаме јазлите A, односно D кои не се проширени.





Значи го прошируваме јазелот D. Забележете дека состојбите C и G се достапнни преку D. Но бидејќи веќе го имаме посетено C, не додаваме јазел кој кореспондира на таа патека. Дадаваме нов јазел кој кораспондира со патеката на G.

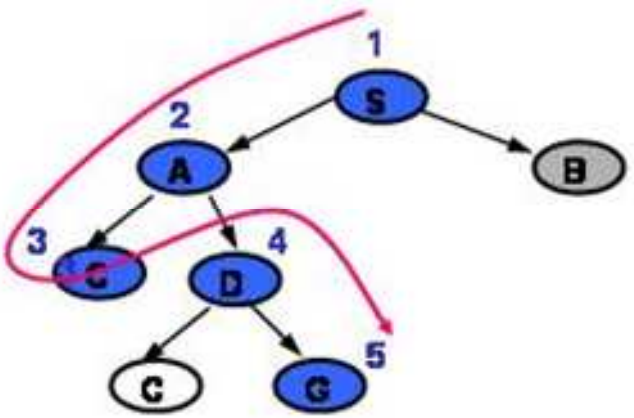


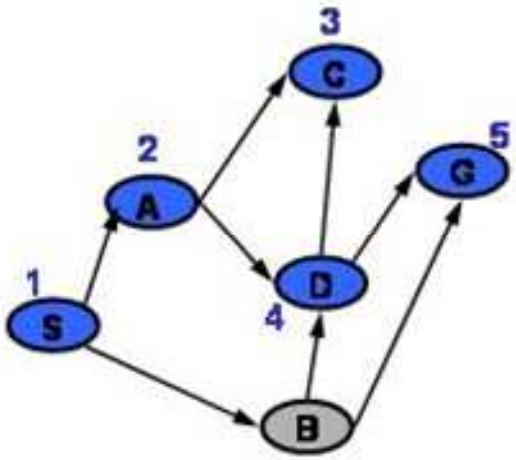


Забелешка: C не е посетен повторно

Сега го прошируваме G.

Ваквиот начин на приказ на пребарување по длабочина е најчест. Во суштина, во овој приказ може најдобро да видиме зошто пребарувањето се вика по длабочина. Розевата стрелка покажува секвенца на проширувања за време на пребарувањето и може да се забележи дека секогаш оди длабоко во дрвото колку што е тоа можно. Исто така, може да видиме и зошто често пати ова пребарување се нарекува **пребарување со враќање по истиот пат назад**. Но сепак, да се има во предвид дека ова е само различен приказ на истото пребарување на Q-базираниот алгоритам.

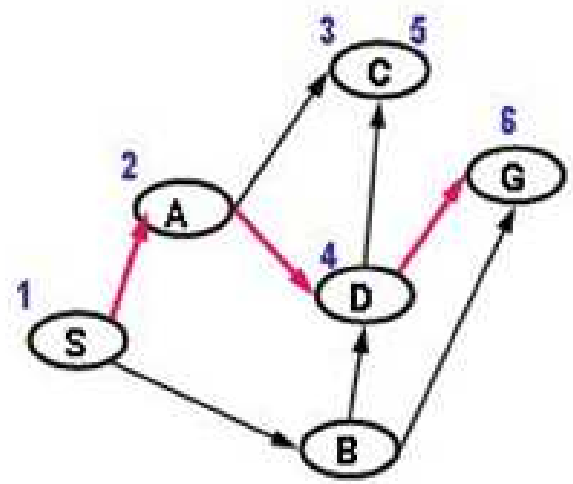




Може да го повториме процесот на пребарување по длабочина без листата на посетени јазли, и како што се гледа во овој случај, додаден е повторно пат до C во Q, кој кога ја користевме листата на посетени јазли, беше блокиран. Како вежба, можете да поминете низ овој алгоритам сами.

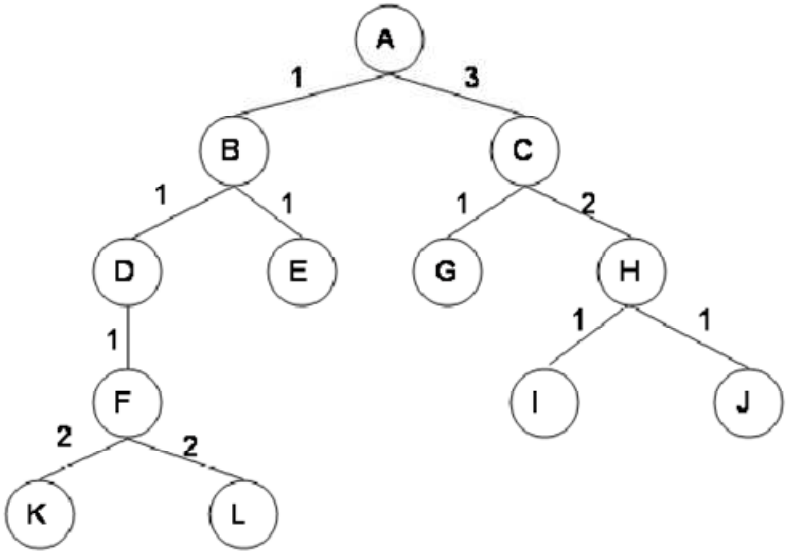
Во отсуство на листата на посетени јазли, повторно како барање се појавува да нема навлегување во циклуси, па ако сме посетиле некоја состојба во некоја патека, не треба повтроно да ја посетуваме истата во ниедно проширување на било која патека.

	Q
1	(S)
2	(AS)(BS)
3	(CAS)(DAS)(BS)
4	(DAS)(BS)
5	(CDAS)(GDAS)(BS)
6	(GDAS) (BS)



Задача 1

Да го разгледаме дрвото подолу.



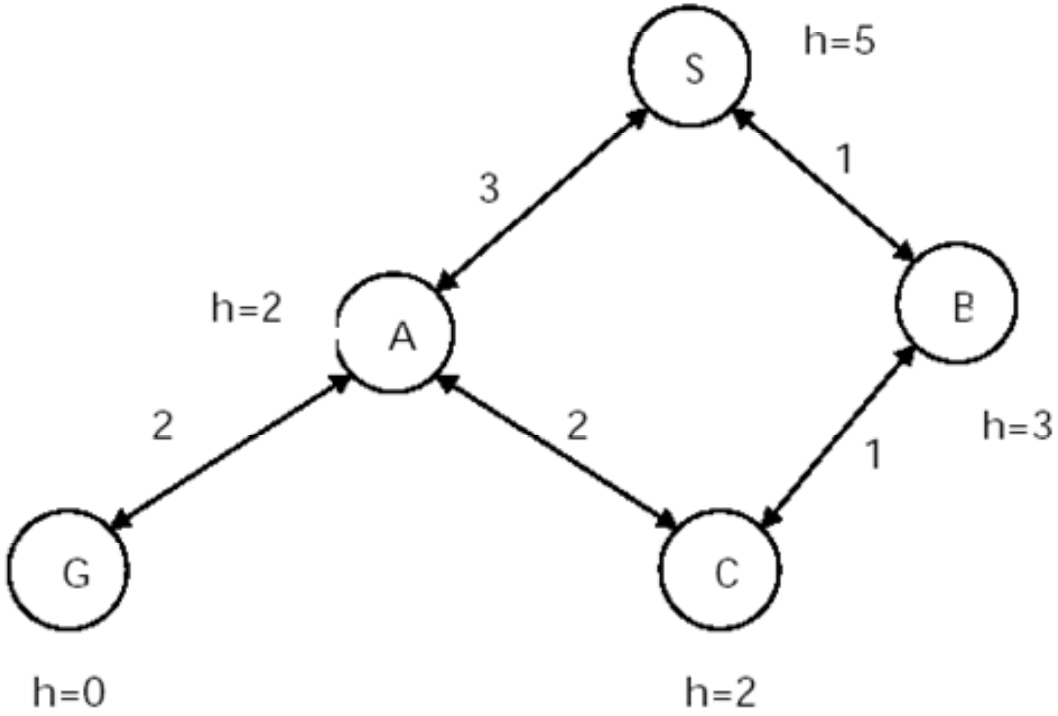
Броевите на врските нека претставуваат должини на врските. Нека претпоставиме дека јазлите се прошируваат по азбучен редослед кога не е специфициран друг вид на пребарување, и нека цел претставува јазелот G. Не се користат никакви проширени или посетени листи.

	Q
1	(A)
2	(BA)(CA)
3	(DBA) (EBA) (CA)
4	(FDBA) (EBA) (CA)
5	(KFDBA) (LFDBA) (EBA) (CA)
6	(EBA) (CA)
7	(CA)
8	(GCA) (H CA)

За дома: Да се определи листата на посетени јазли

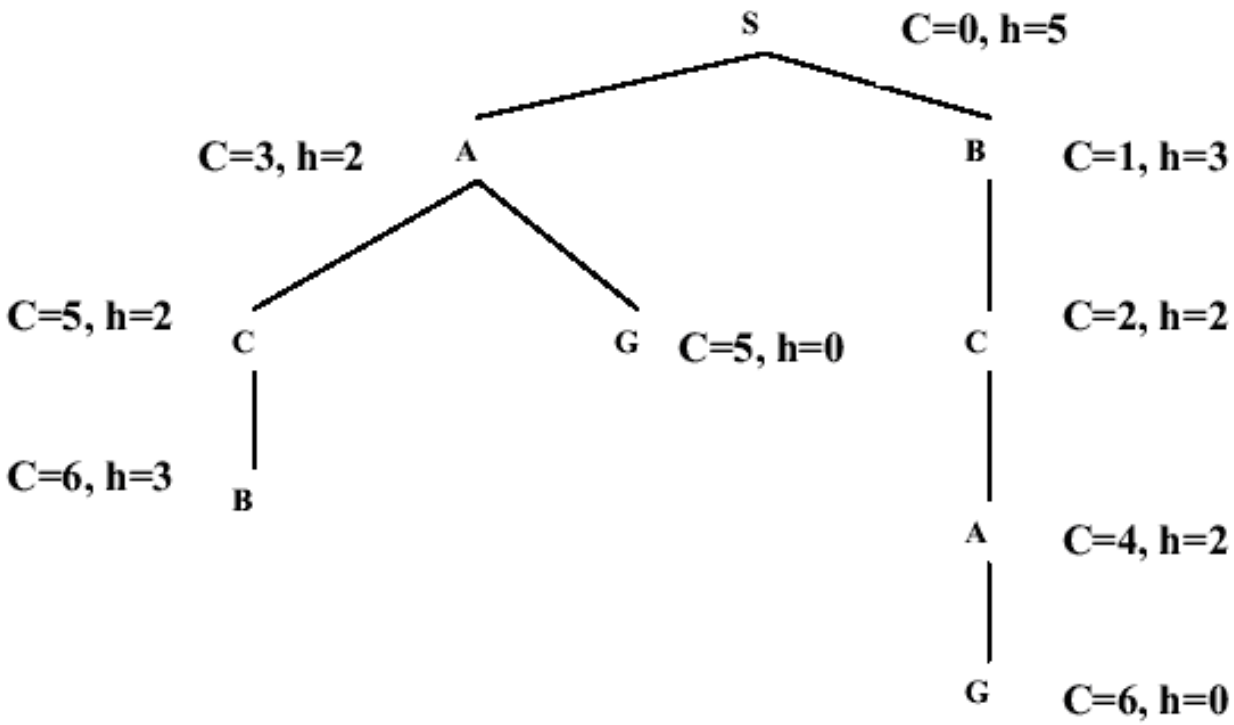
Задача 2

Подолу е даден граф кој треба да се пребара (започнувајќи од S и завршувајќи во G). Цените на врските се прикажани како хеуристички проценки на состојбите. Може да не ви се потребни сите информации дадени на сликата за сите типови на пребарување.



Нацртајте го комплетното пребарување на овој граф. Означете го секој јазел во дрвото со цената на патеката до тој јазел и со хеуристичката цена на тој јазел. Кога треба да се референцирате кон некој јазел, користете го името на соодветната состојбата и должината на патеката до таа состојба.

Решение



1.Извршете пребарување по длабочина со користење на листа на посетени јазли. Претпоставете дека децата на состојбите се подредени по азбучен редослед. Прикажете ја секвенцата на јазли кои се проширени со пребарувањето.

S0, A3, C5, G5. Забележете дека B6 не е експандирано бидејќи B е на листата на посетени јазли (ставено е таму при проширување на S0).

	Q	Посетени
1	(S)	S
2	(AS)(BS)	A, B, S
3	(CAS)(GAS)(BS)	G,C, D, B, A, S
4	(GAS)(BS)	G,C, D, B, A, S
5	(GAS)(BS)	C, D, B, A, S