

**UNIVERSITETI I PRISHTINËS "HASAN PRISHTINA"**  
**FAKULETI I SHKENCAVE MATEMATIKE-NATYRORE**

Departamenti: Matematikë | Programi: Shkencë kompjuterike



**TEMA:** DFS (Depth-First Search) Algoritmi

**LËNDA:** Inteligjencë Artificiale

**Punuan:**

Anisa Mazreku

Blerona Aliu

Erdonesa Kuqi

## **Abstrakti**

DFS (akronimi për Depth-First Search) është një algoritmë kërkimi që fillon nga një nyjë rrënjë dhe eksploron sa më thellë në secilën degë para se të kthehet prapa. Përdoret shpesh për kërkimin e rrugëve në një graf ose për të vizituar të gjitha nyjet në një komponentë të lidhur. Algoritmi mund të implementohet si një funksion rekursiv ose duke përdorur një strukturë të të dhënave stack për të mbajtur shënime të nyjeve që duhet të vizitohen. Aplikohet në fusha të ndryshme si matematik, shkencë kompjuterike etj. Nuk është i plot as nuk ka kosto të mirë optimale dhe ka kompleksitet të larë në disa raste dhe është i implementueshëm në Java.

**Keywords:** DFS (Depth-First Search), Algoritëm, Përdorimi, Implementimi

## Përmbajtja

1	Hyrja .....	4
2	Cka është algoritmi Depth first search?.....	5
3	Aplikimet DFS.....	6
3.1	Zbulimi i ciklit në një grafik .....	6
3.2	Gjetja e rrugës .....	6
3.3	Renditja topologjike .....	6
3.4	Për të provuar nëse një grafik është dypalësh .....	7
3.5	Gjetja e komponentëve të lidhur fort të një grafiku .....	7
3.6	Zgjidhja e enigmave me vetëm një zgjidhje, siç janë labirintet .....	7
3.7	Web crawlers.....	7
3.8	Gjenerimi i labirintit.....	8
3.9	Kontrollimi i modelit.....	8
3.10	Backtracking.....	8
4	Katër kriteret e vlerësimit .....	9
4.1	Plotëshmeria .....	9
4.2	Kosto optimale .....	9
4.3	Kompleksiteti kohor .....	9
4.4	Kompleksiteti hapësinor.....	10
5	Dokumentumi .....	10
6	Referencat .....	11
7	Tabelat .....	11

## 1 Hyrja

Inteligjenca artificiale (AI) përfshin krijimin e sistemeve inteligjente që mund të kryejnë detyra që zakonisht kërkojnë inteligjencë njerëzore. Një aspekt i rëndësishëm i AI është aftësia për të kërkuar zgjidhje për problemet. Algoritmet e kërkimit janë në qendër të shumë aplikacioneve të inteligjencës artificiale, duke filluar nga përpunimi i gjuhës natyrore dhe njohja e imazhit deri te lojrat dhe robotika. Algoritmet e kërkimit përdoren për të gjetur një zgjidhje për një problem duke eksploruar një sërë zgjidhjesh të mundshme. Këto algoritme kërkojnë sistematikisht nëpër një hapësirë të madhe zgjidhjesh të mundshme, duke vlerësuar secilën prej tyre derisa të gjendet një zgjidhje e kënaqshme ose të jenë ekzekutuar të gjitha zgjidhjet e mundshme. Ka shumë lloje të ndryshme të algoritmeve të kërkimit, secila me pikat e forta dhe të dobëta të veta. Në këtë kontekst, është thelbësore të kuptohen parimet e algoritmeve të kërkimit në AI, llojet e ndryshme të algoritmeve të kërkimit dhe se si ato mund të zbatohen për të zgjidhur probleme komplekse. Në këtë punim ne do të flasim për algoritmin depth first search. Një algoritm i cili përdoret për zgjidhjen e shumë problemeve komplekse [1].

## 2 Cka është algoritmi Depth first search?

Kërkimi i parë në thellësi (DFS) është një algoritëm popullor që përdoret në problemet e bazuara në grafik, duke përfshirë ato që hasen në inteligjencën artificiale. Është një teknikë e përdorur për të përshkuar ose kërkuar strukturën e të dhënave të grafikut ose pemës, duke vizituar të gjitha kulmet ose nyjet në grafik/pemë, duke filluar nga një nyje e veçantë dhe duke vazhduar sa më shumë që të jetë e mundur përgjatë secilës degë përpara se të kthehet prapa. DFS funksionon duke vizituar një nyje në një kohë, duke e shënuar atë si të vizituar dhe më pas duke eksploruar nyjet e saj ngjitur të pavizituara. Ai ndjek një qasje "thellësia e parë", që do të thotë se eksploron sa më shumë që të jetë e mundur përgjatë çdo dege përpara se të kthehet prapa. Ekzistojnë dy variante kryesore të DFS: DFS rekursive dhe DFS përsëritëse. Në DFS rekursive, një funksion përdoret për të përshkuar grafikun, ku funksioni thirret në mënyrë rekursive për çdo nyje të pavizituar derisa të mos ketë më nyje të pavizituara. Në DFS përsëritëse, një pirl përdoret për të mbajtur gjurmët e nyjeve që do të vizitohen dhe algoritmi vazhdon derisa pirlja të jetë bosh. Një nga avantazhet kryesore të DFS është se mund të përdoret për të gjetur një shteg zgjidhjeje nga një nyje e caktuar fillestare në një nyje të synuar, veçanërisht në problemet ku zgjidhja nuk është lehtësisht e dukshme. Sidoqoftë, DFS mund të bllokohet në një lak të pafund nëse ka një cikël në grafik, duke e bërë atë më pak efektiv në situata të caktuara. Në përgjithësi, DFS është një algoritëm i dobishëm në AI, veçanërisht në rastet kur zgjidhja nuk është lehtësisht e dukshme dhe është thelbësore të kërkon një hapësirë të madhe mundësish. një hyrje në lidhje me algoritmin e parë të kërkimit në thellësi [2].

Kërkimi i parë i thellësisë (DFS) është një algoritëm i përdorur gjerësisht në teorinë e grafikëve dhe inteligjencën artificiale, i përdorur për të përshkuar ose kërkuar një strukturë të dhënash grafiku ose peme. Algoritmi fillon në një nyje të caktuar dhe eksploron sa më shumë që të jetë e mundur përgjatë çdo dege përpara se të kthehet prapa. Teknika njihet si "thellësia e parë" pasi eksploron një shteg në thellësi përpara se të kalojë në shtegun tjetër. DFS mund të zbatohet si në mënyrë rekursive ashtu edhe në mënyrë përsëritëse dhe është e dobishme në aplikacione të ndryshme të AI, duke përfshirë përpunimin e gjuhës natyrore, njohjen e imazheve, luajtjen e lojërave dhe robotikën. Sidoqoftë, DFS mund të bllokohet në një unaz të pafundme nëse ka një cikël në grafik, duke e bërë atë më pak efikas në disa skenarë. Sidoqoftë, DFS është një mjet i vlefshëm për zgjidhjen e problemeve komplekse në AI që kërkojnë kërkimin e hapësirave të mëdha të zgjidhjeve të mundshme [3].

### 3 Aplikimet DFS

Në këtë pjesë do të flasim rreth aplikimit ose përdorimit të algoritmit DFS, pasi po të mos ishte i përdorur në shumë aspekte mbase nuk do të ishte kaq i njohur dhe i komplimentuar për funksionalitetin e tij.

#### 3.1 Zbulimi i ciklit në një grafik

DFS (Depth-First Search) është një algoritëm kërkimi që përdoret për të eksploruar dhe të analizojë grafitë. Në rastin e kërkimit të një cikli në një grafik, DFS mund të përdoret për të kuptuar nëse grafiku ka një cikël apo jo. Për të kuptuar se si mund të zbulohet një cikël në një grafik duke përdorur DFS, duhet të ndiqni këta hapa: Fillimisht, përzgjidhni një nyje fillestare dhe filloni DFS nga kjo nyje. Në kohën që eksploronin secilën nyje, shënoni atë si të vizituar. Nëse DFS takon një nyje që është tashmë shënuar si të vizituar, atëherë kjo do të thotë se kemi gjetur një cikël. Për të verifikuar nëse kjo është një cikël, ne duhet të kontrollojmë nëse ky nyje është në thellësi më të ulët se nyja aktuale në stekun DFS. Nëse është, atëherë kjo është një cikël. Nëse kjo nyje nuk është në thellësinë më të ulët, atëherë vazhdojmë DFS-në dhe eksplorojmë më tej. Nëse DFS përfundon dhe nuk kemi gjetur asnjë cikël, atëherë grafiku nuk ka cikël. Nëse keni gjetur një cikël, atëherë mund të shënoni të gjitha nyjet në cikël dhe ktheni listën e tyre si rezultat.

#### 3.2 Gjetja e rrugës

Ne mund ta specializojmë algoritmin DFS për të gjetur një shteg midis dy kulmeve të dhëna  $u$  dhe  $z$ . Për të gjetur një shteg midis dy kulmeve të dhëna  $u$  dhe  $z$  në një grafik  $G$ , mund të specializojmë algoritmin DFS në këtë mënyrë: Thirrni  $DFS(G, u)$  me  $u$  si kulm fillestar. Përdorni një pirg  $S$  për të mbajtur gjurmët e shtegut midis kulmit fillestar dhe kulmit aktual. Fillimisht shtypni kulmin fillestar në  $S$ . Gjatë kërkimit DFS, në kohën që eksploronin secilën nyje, shtypni atë në  $S$ . Sapo të ndeshet kulmi i destinacionit  $z$ , ktheni shtegun si përmbajtjen e pirgut  $S$  duke filluar nga kulmi fillestar  $u$ . Nëse DFS përfundon dhe nuk kemi gjetur kulmin e destinacionit  $z$ , atëherë nuk ka shteg midis kulmeve  $u$  dhe  $z$ . Kjo do të na japë shtegun midis kulmit fillestar  $u$  dhe kulmit destinacion  $z$  në grafikun  $G$ . Pirgu  $S$  do të mbajë gjurmët e shtegut dhe do të kthehet në fund si shtegu midis kulmeve  $u$  dhe  $z$ . Nëse nuk gjendet shteg midis kulmeve, atëherë kthehet një mesazh gabimi.

#### 3.3 Renditja topologjike

Renditja topologjike është një koncept i rëndësishëm në teorinë e grafeve dhe përdoret kryesisht për të gjetur një renditje valide të topologjisë së një grafi të orientuar aciklik (DAG - Directed Acyclic Graph). Në një DAG, topologjia është një renditje e gjithë nyjeve të grafit në një mënyrë që për çdo drejtim të rënditur midis dy nyjeve, nyja e parë ndodhet para nyjes së dytë. Në kuptimin e punës, kjo do të

thotë që nuk mund të fillojmë një detyrë para se të kemi përfunduar detyrat përkatëse që varen nga ajo. Në fushën e programimit, renditja topologjike mund të përdoret për t'i kryer detyrat e ndryshme të kompilimit në një skedar në një rend të përcaktuar, në mënyrë që të jenë të sigurt se secila detyrë kryhet në rendin e duhur.

### 3.4 Për të provuar nëse një grafik është dypalësh

Saktësisht, për të provuar nëse një grafik është dypalësh, mund të përdoret një algoritëm DFS ose BFS. Në fillim, mund të zgjidhni një kulm të caktuar si kulm fillestar dhe ta ngjyrosni me një ngjyrë, duke filluar nga ngjyra e parë, thjesht për qëllime ilustruese le të themi se ngjyra e parë është e kuqe. Pastaj, duke përdorur DFS ose BFS, mund të vizitohen të gjitha kulmet e grafikut dhe çdo herë që vizitohet një kulm i ri, ai ngjyroset me ngjyrën e kundërt të prindit të tij. Nëse gjatë këtij procesi dy kulme të së njëjtës ngjyre lidhen me njëri-tjetrin, atëherë grafiku nuk është dypalësh. Nëse, nga ana tjetër, mund të vizitohet çdo kulm dhe nuk hasen dy kulme të njëjta me ngjyrë që janë të lidhura me njëri-tjetrin, atëherë grafiku është dypalësh.

### 3.5 Gjetja e komponentëve të lidhur fort të një grafiku

Një graf i drejtuar është një strukturë matematikore që përbëhet nga një grup kulmesh dhe anësh, ku secila anë është një lidhje mes dy kulmeve, dhe kjo lidhje është e orientuar në një drejtim të caktuar. Kjo do të thotë se në grafikun e drejtuar, ndryshon renditja e kulmeve që janë të lidhura nga një anë. Një graf i drejtuar quhet i lidhur fort nëse ka një shteg nga çdo kulm në grafik në çdo kulm tjetër.

### 3.6 Zgjidhja e enigmave me vetëm një zgjidhje, siç janë labirintet

DFS mund të përshtatet për të gjetur të gjitha zgjidhjet për një labirint duke përfshirë vetëm nyjet në shtegun aktual në grupin e vizituar.) Por në fakt, DFS nuk është një zgjidhje e mirë për zgjidhjen e enigmave si labirintet me vetëm një zgjidhje. DFS mund të zbulojë një zgjidhje të mundshme, por nuk mund të garantojë se ai do të gjejë zgjidhjen optimale në kohë të arsyeshme. Kjo është sepse DFS shkon në thellësi nëpër grafik dhe mund të rrezikojë të rri në një rrugë të gabuar ose në një cikël, duke humbur kohë dhe duke mos gjetur zgjidhjen optimale. Në vend të kësaj, algoritmet e kërkimit në gjerësi si BFS, A\* dhe algoritmi i kërkimit të greinjës (greedy search) janë më të mira për zgjidhjen e enigmave me vetëm një zgjidhje si labirintet. Këto algoritme janë më të sofistikuar dhe kanë aftësinë për të gjetur zgjidhjen optimale në kohë të arsyeshme.

### 3.7 Web crawlers

Kërkimi i parë në thellësi mund të përdoret në zbatimin e zvarritësve të uebit për të eksploruar lidhjet në një faqe interneti. Kërkimi i parë në thellësi mund të përdoret për eksplorimin e lidhjeve në një

faqe interneti në rastet kur dëshirojmë të gjejmë sa më shumë faqe të lidhura me faqen fillestare në kohë të shkurtër. Kështu, në rastin e zvarritësve të uebit, kërkimi i parë në thellësi përdoret për të shkuar thellë në një faqe të caktuar dhe pastaj për të vazhduar të eksploron faqet e lidhura me atë faqe në këtë drejtim.

### **3.8 Gjenerimi i labirintit**

Kërkimi i parë në thellësi mund të përdoret për të gjeneruar labirint të rastësishëm. Kërkimi i parë në thellësi mund të përdoret për të gjeneruar labirint të rastësishëm duke filluar me një kornizë të madhe me mure dhe pastaj duke zgjedhur një nyje të rastësishme për të ndjekur rrugën e labirintit. Kur ndeshet në mure ose në shtegun e krijuar më parë, algoritmi mund të zgjedhë një tjetër rrugë të rastësishme për të vazhduar ndjekjen. Kjo vazhdon deri sa të arrihet në fund të labirintit ose deri sa të krijojnë një labirint të plotë. Kjo është një metodë interesante për të gjeneruar labirintë të ndryshme dhe ka shumë aplikime në lojëra video, si dhe në robotikë dhe inteligjencë artificiale.

### **3.9 Kontrollimi i modelit**

Në kontrollin e modelit, kërkimi i parë në thellësi mund të përdoret për të eksploruar gjendjet e mundshme të modelit dhe për të verifikuar nëse ato plotësojnë vetitë e dëshiruara. Duke filluar nga një gjendje fillestare, algoritmi DFS mund të ndjekë gjendjet e mundshme të modelit, duke u përpoq për të gjetur një sërë gjendjesh që plotësojnë vetitë e dëshiruara. Kjo mund të ndihmojë në zbulimin e gabimeve në model dhe në përmirësimin e tij.

### **3.10 Backtracking**

Kërkimi i parë në thellësi mund të përdoret në algoritmet e backtracking. Algoritmet e backtracking janë algoritme për të zgjidhur problemele kombinatorike, si gjetja e një zgjidhjeje optimale për një probleme, dhe përdorin kërkimin në thellësi për të eksploruar hapat e mundshëm të zgjidhjes deri në gjetjen e një zgjidhjeje të vlefshme. Në këtë proces, nëse shikohet se zgjidhja e zgjedhur nuk plotëson kërkesat e problemit, algoritmi kthen prapa dhe provon zgjidhje të tjera deri në gjetjen e një zgjidhjeje të vlefshme ose përfundimin se nuk ka zgjidhje [4][6].



## 4 Katër kriteret e vlerësimit

Kriteret	Depth-first search
Plotshmeria?	No
Optimal Cost?	No
Time	$O(b^m)$
Space	$O(bm)$

Table 1. Katër kriteret e vlerësimit

### 4.1 Plotëshmeria

DFS nuk është i garantuar të jetë i plotë në gjetjen e shtegut të mundshëm në mes të dy kuptimeve në një graf të paorientuar me peshë (dhe kjo mund të jetë një problem në disa raste). Për shembull, në një graf të thjeshtë të paorientuar me tre pikë, A, B dhe C, nëse ekziston shtegu A-B-C me peshë 1 dhe shtegu A-C me peshë 10, DFS mund të fillon nga pikat A ose B, por nuk do të jetë në gjendje të kalojë nëpër pikën C përmes shtegut më të shkurtër, sepse nuk do të ketë ndonjë informacion për peshat e shtegëve. Megjithatë, në një graf të orientuar me peshë, DFS është i plotë në gjetjen e shtegut të mundshëm midis dy kuptimeve nëse ai ekzekutohet në mënyrë korrekte.

### 4.2 Kosto optimale

DFS nuk është një algoritëm që garanton gjetjen e rrugës më të shkurtër midis dy pikave. Algoritmi fillon nga një pikë dhe shkon në një pikë tjetër të vlefshme të padalë, duke e shtuar atë në skajin e shtegut të vizituar. DFS vazhdon të kërkojë thellësi më tej nga pikat e reja, duke shtuar secilën pikë të vizituar në skajin e shtegut. Nëse një pikë nuk lidhet me pikën e kërkimit aktual, DFS kthehet mbasprap dhe vazhdon me pikën tjetër të mundshme.

Algoritmet e kërkimit të rrugës më të shkurtër janë të specializuara për këtë qëllim, siç është algoritmi i Dijkstrës, algoritmi A\* etj.

### 4.3 Kompleksiteti kohor

Kompleksiteti kohor i DFS është  $O(b^m)$ , ku b është faktori i degëve dhe m është thellësia maksimale e kërkimit në hapësirë. Faktori i degëve tregon numrin maksimal të degëve që mund të ketë një pikë, ndërsa thellësia maksimale e kërkimit në hapësirë tregon sa thellë mund të shkojë kërkimi në hapësirë për të gjetur zgjidhjen.

Kjo kompleksitet mund të jetë shumë i lartë në raste të caktuara, sidomos në grafe të dendura ose me shumë cikle. Në këto raste, algoritmet e tjerë kërkimi mund të jenë më të mira për të gjetur zgjidhjen e kërkuar me një kohë më të shkurtër. Megjithatë, DFS mbetet një algoritëm i dobishëm për kërkimin në grafe të thjeshta dhe mund të përdoret në shumë aplikacione të ndryshme.

## 4.4 Kompleksiteti hapësinor

Kompleksiteti hapësinor i DFS mund të shprehet në funksion të faktorëve si numri i nyjeve dhe thellësia maksimale e kërkimit në një pemë të kërkimit. Nëse kemi një pemë me faktorë deri në  $b$  dhe thellësi maksimale  $m$ , atëherë kompleksiteti hapësinor i DFS është  $O(bm)$ .

Kjo është pasi DFS mbajt një stivë që ruan gjurmët e shtegjeve që ai ka eksploruar, dhe ndërsa shkon në thellësi në pemë, ai mund të mbajë në memorie një numër të madh shtegjesh të gjata në stivë. Kështu që, hapësira që DFS kërkon në memorie është e drejtpërdrejtë e lidhur me thellësinë maksimale të kërkimit dhe faktorin  $b$  i lidhur me numrin e fëmijëve që ka çdo nyje në pemë. Në raste kur pemës i mungon struktura e duhur, DFS mund të përdorë hapësirë shumë më të madhe në memorie [6].

## 5 Dokumentumi

Klasat	
Graphh	Klasa implementon të gjithë metodat për funksionimin e algoritmit DFS
DFSExample	Kjo klasë demonstron një shembull të implementimit të algoritmin DFS nga metodat në klasën Graphh
Konstruktori	
Graphh(int v)	Krijon një grafik me kulme v dhe inicializon një listë boshe fqinjësie për qdo kulm.
DFSExample()	I krijuar nga java
Metoda	
void addEdge(int v, int ë)	Shton një skaj në grafik.
void DFS(int v)	Shënon të gjitha kulmet si të pa vizituara Thirr funksionin ndihmës rekursiv për të printuar kalimin e DFS.
void DFSUtil(int v, boolean[] visited)	Shënon nyjen aktuale si të vizituar dhe printon atë. Përsërit për të gjitha kulmet ngjitur me këtë kulm.
static void	main(String[] args)

Table 2. Dokumentimi i kodit në java për algoritmin DFS

## 6 Referencat

- [1] <https://www.javatpoint.com/search-algorithms-in-ai>
- [2] <https://www.programiz.com/dsa/graph-dfs>
- [3] <https://www.educative.io/answers/what-is-depth-first-search>
- [4] <https://www.geeksforgeeks.org/applications-of-depth-first-search/>
- [5] <https://aima.cs.berkeley.edu/>
- [6] S.S. Skiena. (2008). The Algorithm Design Manual, 2nd edition.

## 7 Tabelat

Table 1. Katër kriteret e vlerësimit

Table 2. Dokumentimi i kodit në java për algoritmin DFS