UNIVERSITETI I PRISHTINËS “HASAN PRISHTINA”FAKULTETI I SHKENCAVE MATEMATIKO-NATYROREDEPARTAMENTI I MATEMATIKËSPROGRAMI: SHKENCA KOMPJUTERIKE

 **PUNIM SEMINARIK**Lënda: Intelegjenca Artificiale   
TEMA: Breadth First Search

**Punuar nga: Albina Grajqevci, Diellza Bajraktari, Dalina Elshani**

**Prill 2023**

Përmbajtja

[1. Hyrje 2](#_Toc133177333)

[2. Si punon algoritmi? 3](#_Toc133177334)

[3. Advantazhet dhe disadvantazhet e BFS 3](#_Toc133177335)

[4. Krahasimi me algoritmet tjera të kërkimit 4](#_Toc133177336)

[4. Shembull 5](#_Toc133177337)

[6. Pseudokodi 7](#_Toc133177338)

[7. Përdorimi 7](#_Toc133177339)

[8. Përmbledhje 8](#_Toc133177340)

[9.Referencat 9](#_Toc133177341)

# Hyrje

Inteligjenca Artificiale (AI) është një fushë e shkencës kompjuterike që merret me krijimin e makinave inteligjente që mund të kryejnë detyra që zakonisht kërkojnë inteligjencë njerëzore. Një nga detyrat themelore në AI është kërkimi i zgjidhjeve për problemet. Një algoritëm kërkimi është një komponent themelor i shumë sistemeve të AI dhe ndihmon në gjetjen e zgjidhjeve optimale ose afërsisht optimale për problemet.

Breadth first search është një algoritëm i përdorur gjerësisht në shkencën kompjuterike për kalimin ose kërkimin nëpër një grafik ose strukturë të dhënash peme. Përdoret për të eksploruar të gjitha nyjet në një thellësi të caktuar përpara se të kaloni në nivelin tjetër të thellësisë. Në këtë prezantim, ne do të mbulojmë bazat e mënyrës se si funksionon kërkimi në fillim dhe disa nga aplikimet e tij.

BFS funksionon duke përdorur një strukturë të dhënash në radhë për të mbajtur gjurmët e kulmeve që duhen vizituar. Fillon duke vizituar kulmin fillestar dhe duke vendosur kulmet e tij ngjitur në radhë. Më pas ai heq kulmin e parë nga radha dhe viziton kulmet e tij ngjitur që nuk janë vizituar ende. Ky proces vazhdon derisa të vizitohen të gjitha kulmet në grafik.

BFS ka një kompleksitet kohor prej O(V+E) ku V është numri i kulmeve dhe E është numri i skajeve në grafik. Kjo është për shkak se në skenarin më të keq, BFS duhet të vizitojë çdo kulm dhe skaj në grafik.

# Si punon algoritmi?

Algoritmi funksionon duke filluar nga një nyje e caktuar, më pas duke vizituar të gjithë fqinjët e saj përpara se të kalojë në nivelin tjetër të nyjeve. Kjo do të thotë që ne eksplorojmë të gjitha nyjet në një thellësi të caktuar përpara se të kalojmë në nivelin tjetër

BFS fillon në nyjen rrënjë ose një kulm fillestar dhe eksploron të gjitha nyjet në nivelin aktual përpara se të kalojë në nivelin tjetër. Ai mban një radhë të nyjeve që duhen vizituar, me kulmin fillestar si nyjen e parë në radhë. Ndërsa vizitohet çdo nyje, nyjet e saj ngjitur vendosen në radhë.

Më pas, algoritmi heq nyjen e parë nga radha dhe viziton nyjet e saj ngjitur. Ajo shënon çdo nyje të vizituar për të shmangur vizitën e saj përsëri. Ky proces vazhdon derisa radha të zbrazet, që do të thotë se të gjitha nyjet në grafik janë vizituar.

# Advantazhet dhe disadvantazhet e BFS

Disa nga advantazhet e përdorimit të BFS janë:

1. Plotësia: Algoritmi BFS është i garantuar për të gjetur një zgjidhje nëse ekziston. Kjo do të thotë që nëse ka një rrugë nga nyja fillestare në nyjen e qëllimit, algoritmi përfundimisht do ta gjejë atë.
2. Optimaliteti: Algoritmi BFS do të gjejë shtegun më të shkurtër midis nyjeve të fillimit dhe qëllimit nëse peshat e skajeve janë të gjitha të njëjta. Është i vetmi algoritëm që garanton optimalitet kur të gjitha kostot e skajeve janë të njëjta.
3. Thjeshtësia: Algoritmi BFS është relativisht i lehtë për t'u zbatuar dhe kuptuar. Kërkon vetëm një radhë për të ruajtur nyjet që do të vizitohen.

Disa nga disadvantazhet e përdorimit të BFS janë:

1. Kërkesat e memories: Algoritmi BFS ruan të gjitha nyjet e vizituara në memorie, gjë që mund të jetë problem kur kërkoni hapësira të mëdha. Kërkesa e memories për BFS është proporcionale me numrin e nyjeve në pemën e kërkimit.
2. Kompleksiteti kohor: Algoritmi BFS ka një kompleksitet kohor prej O(b^d), ku b është faktori i degëzimit dhe d është thellësia e pemës së kërkimit. Kjo do të thotë që algoritmi mund të jetë shumë i ngadalshëm për hapësira të mëdha kërkimi.
3. Kërkime të tepërta: Algoritmi BFS mund të vizitojë të njëjtën nyje disa herë, gjë që mund të jetë e kotë dhe të ngadalësojë kërkimin.

# Krahasimi me algoritmet tjera të kërkimit

1. Kërkimi i parë në thellësi (DFS):

Algoritmi DFS eksploron sa më shumë që të jetë e mundur përgjatë çdo dege përpara se të kthehet prapa. Ai përdor një pirg për të mbajtur gjurmët e nyjeve që do të vizitohen. Ndryshe nga BFS, DFS nuk garanton plotësinë ose optimalitetin, por shpesh është më i shpejtë dhe përdor më pak memorie se BFS. DFS është i dobishëm kur hapësira e kërkimit është e madhe dhe e thellë, dhe qëllimi ka të ngjarë të gjendet në thellësinë e pemës së kërkimit.

1. Kërkimi me kosto uniforme (UCS):

Algoritmi UCS zgjeron nyjen me koston më të ulët të rrugës g(n) midis të gjitha nyjeve të pazgjeruara. UCS garanton optimalitet, por mund të eksplorojë shtigje nënoptimale nëse kostot e skajit nuk janë uniforme. UCS është i dobishëm kur kostoja e arritjes së qëllimit është shqetësimi kryesor.

3. Kërkim A\*:

Algoritmi A\* zgjeron nyjen me f(n) = g(n) + h(n) më të ulët midis të gjitha nyjeve të pazgjeruara, ku g(n) është kostoja e rrugës nga nyja fillestare në nyjen n dhe h(n) është vlerësimi heuristik i kostos nga nyja n tek nyja e qëllimit. A\* garanton optimalitet dhe shpesh është më efikas se UCS nëse disponohet një heuristikë e mirë. A\* është i dobishëm kur hapësira e kërkimit është e madhe dhe kostoja e arritjes së qëllimit ndryshon shumë.

Ngjashmëritë:

1. Të gjitha këto algoritme kërkimi synojnë të gjejnë një shteg nga një nyje fillestare në një nyje qëllimi në një hapësirë të caktuar kërkimi.
2. Të gjitha përdorin një strukturë të dhënash (p.sh., queue, stack, or priority queue) për të mbajtur gjurmët e nyjeve që do të vizitohen.
3. Të gjitha ato mund të zbatohen në mënyrë përsëritëse ose rekursive.

Dallimet:

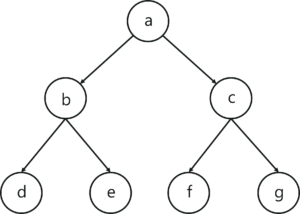
1. BFS garanton plotësinë dhe optimalitetin, ndërsa DFS nuk garanton asnjëren.
2. UCS garanton optimalitet, por mund të eksplorojë shtigje nënoptimale nëse kostot e skajit nuk janë uniforme.
3. A\* garanton optimalitet dhe mund të jetë më efikas se UCS nëse disponohet një heuristikë e mirë.
4. BFS dhe UCS përdorin një radhë për të mbajtur gjurmët e nyjeve që do të vizitohen, ndërsa DFS përdor një pirg.
5. A\* përdor si koston e arritjes së një nyje, ashtu edhe një vlerësim heuristik të kostos së mbetur për të arritur qëllimin, ndërsa algoritmet e tjera përdorin vetëm koston e arritjes së një nyje.

Në përmbledhje, çdo algoritëm kërkimi ka pikat e forta dhe të dobëta të veta, dhe zgjedhja e cilit algoritëm të përdoret varet nga karakteristikat e hapësirës së kërkimit dhe kërkesat e problemit.

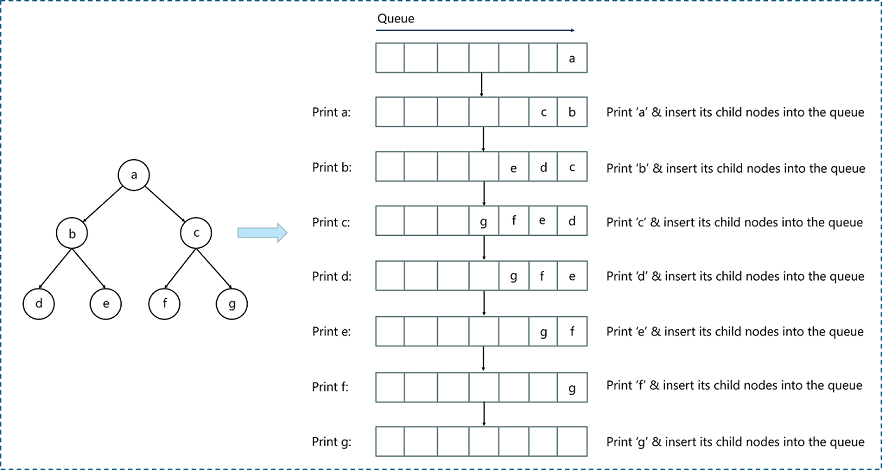
# Shembull

Para se te fillojmë duhet t’a dimë që struktura kryesore e të dhënave që përdoret është queue. Queue është një strukturë abstrakte e të dhënave që ndjek metodologjinë First-In-First-Out (të dhënat e futura së pari do të aksesohen së pari). Ai është i hapur në të dy skajet, ku një fund përdoret gjithmonë për të futur të dhëna (në radhë) dhe tjetri përdoret për të hequr të dhënat (dequeue).

Si shembull do t’a marrim grafin më poshtë dhe do t’a përdorim BFS për të kaluar nëpër graf.



Qëllimi ynë është të përshkojmë grafikun duke përdorur Algoritmin e Kërkimit Breadth-First.



Sqarim:

1. Cakto 'a' si nyjen rrënjë dhe fute atë në Radhë.
2. Nxirrni nyjen 'a' nga radha dhe futni nyjet fëmijë të 'a', d.m.th., 'b' dhe 'c'.
3. Shtypni nyjen 'a'.
4. Radha nuk është bosh dhe ka nyjen 'b' dhe 'c'. Meqenëse 'b' është nyja e parë në radhë, le ta nxjerrim atë dhe të fusim nyjet fëmijë të 'b', d.m.th., nyja 'd' dhe 'e'.
5. Përsëritni këto hapa derisa radha të zbrazet. Vini re se nyjet që tashmë janë vizituar nuk duhet të shtohen përsëri në radhë.

# Pseudokodi

Pseudokodi:

function BREADTH-FIRST-SEARCH(*problem*) returns a solution node or *failure*

*node*←NODE(*problem*.INITIAL)

if *problem*.IS-GOAL(*node*.STATE) then return *node*

*frontier*←a FIFO queue, with *node* as an element

*reached*←{*problem*.INITIAL}

while not IS-EMPTY(*frontier*) do

*node*←POP(*frontier*)

for each *child* in EXPAND(*problem*, *node*) do

*s*←*child*.STATE

if *problem*.IS-GOAL(*s*) then return *child*

if *s* is not in *reached* then

add *s* to *reached*

add *child* to *frontier*

return *failure*

Mekanizmi i funksionimit të algoritmit BFS mund të përmblidhet në hapat e mëposhtëm:

1. Filloni në nyjen rrënjë dhe shtoni atë në një radhë.
2. Ndërsa radha nuk është bosh, hiqni nyjen e parë nga radha.
3. Nëse nyja e hequr është nyja e qëllimit, përfundoni kërkimin dhe kthejeni shtegun në nyjen e qëllimit.
4. Përndryshe, shtoni të gjitha nyjet fqinje të nyjes së hequr në radhë.
5. Përsëritni hapat 2 deri në 4 derisa të gjendet nyja e qëllimit ose radha të jetë bosh.

# Përdorimi

Breadth first search ka shumë aplikime në shkencën kompjuterike, duke përfshirë:

* Algoritmet e shtegut më të shkurtër dhe të pemës me shtrirje minimale: BFS mund të përdoret për të gjetur shtegun më të shkurtër ndërmjet dy nyjeve në një grafik të papeshuar ose për të gjetur pemën minimale të shtrirjes në një grafik të ponderuar ku të gjitha peshat e skajeve janë të barabarta.
* Algoritmet e rrjedhës së rrjetit: BFS mund të përdoret për të zgjidhur probleme të ndryshme të rrjedhës së rrjetit, të tilla si problemi i rrjedhës maksimale dhe problemi i prerjes minimale
* Renditja topologjike: BFS mund të përdoret për të kryer renditjen topologjike në një graf jociklik të drejtuar (DAG-directed acyclic graph), i cili është një renditje lineare e kulmeve të tij që është në përputhje me skajet e drejtuara.
* Zvarritja dhe indeksimi i uebit: BFS mund të përdoret në algoritmet e zvarritjes së uebit për të indeksuar dhe marrë informacion nga World Wide Web. Algoritmi fillon në një URL rrënjë dhe viziton të gjitha lidhjet në faqe, duke i shtuar ato në një radhë të URL-ve që do të vizitohen.
* Zgjidhja e enigmave: BFS mund të përdoret për të zgjidhur lloje të ndryshme enigmash, si p.sh. enigma e pllakave rrëshqitëse dhe kubi i Rubikut. Në këto enigma, algoritmi fillon në gjendjen fillestare dhe kryen një kërkim të gjerë të të gjitha lëvizjeve të mundshme derisa të arrijë në gjendjen e qëllimit.
* Gjetja e rrugës në lojëra: BFS mund të përdoret për të gjetur shtegun më të shkurtër midis dy pikave në një botë loje. Algoritmi fillon në vendndodhjen aktuale të lojtarit dhe eksploron të gjitha pllakat aty pranë derisa të gjejë destinacionin.

Në përgjithësi, BFS është një algoritëm i gjithanshëm që mund të zbatohet për një gamë të gjerë problemesh në shkencën kompjuterike dhe më gjerë. Aftësia e tij për të gjetur shtegun më të shkurtër në një grafik të papeshuar dhe efikasiteti i tij në grafikë të rrallë e bëjnë atë një mjet të vlefshëm në shumë aplikacione.

# Përmbledhje

Si përfundim, breadth first search është një algoritëm i dobishëm për kalimin ose kërkimin nëpër një grafik ose strukturë të dhënash peme. Ai funksionon duke eksploruar të gjitha nyjet në një thellësi të caktuar përpara se të kaloni në nivelin tjetër. Ka shumë aplikime në shkencën kompjuterike, duke përfshirë algoritmet e rrugës më të shkurtër, gjetjen e komponentëve të lidhur, zbulimin e cikleve dhe gjenerimin e labirinteve.

# 9.Referencat

(https://www.javatpoint.com/breadth-first-search-algorithm)

(https://www.geeksforgeeks.org/breadth-first-search-or-bfs-for-a-graph/) (https://www.hackerearth.com/practice/algorithms/graphs/breadth-first-search/tutorial/)

(https://www.tutorialspoint.com/data\_structures\_algorithms/breadth\_first\_traversal.htm) (https://www.edureka.co/blog/breadth-first-search-algorithm/)