## Breadth first search 1. Background 2. Data structures 3. Procedure 4. Results and interpretation 1. Background At start: we have a digraph G = (V, E) and a starting node (source node) s from the digraph. Goal: we want to know all nodes that are reachable from s. One way to do this is by performing a graph search. If all other nodes are reachable, then the search is also a traversal. Reminders • y is reachable from x when there is a directed path from x to y• y is adjacent to x when edge (x, y) exists • a path's length is the number of edges in the path The distance $\delta(x, y)$ between from x to y is is the length of shortest path from x to y. New:

Example		
	(D) (F) (H)	

x		A	В	C.	0	F	H
adjacen	t to <i>x</i>	D, B	С	C,A,F	B,F	BjC	C
reachat	ble from <i>x</i>	D, B, C, A, F	с, А, D В, F	, C,A,F B,D	, Β, F, C, A, D	B,C, A,D,F	C, A, F D, B
paths fr	om D to C	(D	F B	c) (	D, B, C)	(D, F,	٤)
length			2				
			2		2	d	
δ(D,	८) = २						
Results	from Brea	dth-first-searc	ch (BFS):				
• all • the	reachable n distance $\delta$	odes from $s$ $(s, y)$ of all rea	achable no	odes from <i>s</i>			
• a ro	ooted tree v	vhose root is a	s which ir	ncludes all r	eachable nodes	from s (the	BF-tree)
2. Data s	tructures						
In a graph s	search a no	ode is either o	liscovere	d or undisc	overed.		
<b>Q</b> : What do	we mean	when we say	a node x	chas been	discovered?		
<b>A</b> : We mea	n that a pa	th from s to x	has bee	n found.			

In any graph search

start with all nodes undiscovered, except s

- progress is made by moving along edges and discovering nodes that have been undiscovered

- graph searches differ in the order in which they move along edges

In BFS the status of a node is monitored by assigning it a color:

- a white node is undiscovered

- a gray node has been discovered, but it may have adjacent nodes that are undiscovered (white)

 a black node has been discovered and all nodes adjacent to it have been discovered (either gray or black)

Note: progression of a node's color: white  $\rightarrow$  gray  $\rightarrow$  black

## To perform BFS, for each node we have the following attributes:

• x.d distance of node x from source s:  $x.d = \delta(s, x)$ 

• x.colour = color of node

•  $x.\pi$  = parent of node x in BF tree

• x.Adj set containing nodes that are adjacent to x

## In BFS we maintain a queue of gray nodes.

## A queue is a one-dimensional data structure that has two ends: the head (front end) and the tail (back end).

• ENQUEUE $(O, x)$ : item x i	s put into <i>O</i> at 1	the tail	
	o pur mio 🤤 ur		
• DEQUEUE( $Q$ ): the item at	t the head of $Q$	is removed and returned	
A queue is said to function on a	a first-in-first-o	out (FIFO) basis.	
Example			
Start with empty queue:	Q: tai	I head	
operation		quouo	
		queue	
ENQUEUE( Q, 7 )	tail	7 head	
ENQUEUE(Q,2)	tail	ہ head	
	tail	ລຸ head	
ENQUEUE( Q, 4 )			
	tail	4 2 head	

3.	Procedure IIII IIII IIII IIIIIIIIIIIIIIIIIIIII
De	escription of BFS:
10	
	BREADTH-FTRST-SEARCH
	1. Mark s as discovered. Mark the distance as $d = 1$ .
	2. Discover all nodes that are a distance of d from s.
	<b>3.</b> Add 1 to $d$ .
	4. If there are still nodes that are reachable from s, then
	repeat steps 2 and 3.
);	
О. Ц	w do we know that there are still nodes that are reachable from s2
	JW do we know that there are still hodes that are reachable hold s?
Psei	Idocode:
1	BREADTH-FIRST-SEARCH $(s, G)$
2	executes a breadth first search on graph G starting from
3	source node s
4	
5	forEach node x in G
6	$x.color = white, x.d = \infty, x.\pi = NIL$
8	end
	/* Give source node appropriate values. */
9	A second with the contrast of
9 10	$s.color = gray, \ s.d = 0$
9 10 11	s.color = gray, s.d = 0 $\triangleright$ initialize a queue in $Q$
9 10 11 12	s.color = gray, s.d = 0 $\triangleright$ initialize a queue in $Q$ ENQUEUE(Q,s)
9 10 11 12 13	s.color = gray, $s.d = 0$ $\triangleright$ initialize a queue in $Q$ ENQUEUE $(Q, s)$ while $Q$ is not empty DESCRIPTION
9 10 11 12 13 14	$s.color = gray, s.d = 0$ $\triangleright \text{ initialize a queue in } Q$ $ENQUEUE(Q,s)$ $while Q \text{ is not empty}$ $x = DEQUEUE(Q)$
9 10 11 12 13 14 14 15 16	s.color = gray, s.d = 0 $\triangleright$ initialize a queue in Q ENQUEUE(Q,s) while Q is not empty x = DEQUEUE(Q) for Each node u in x Adi
9 10 11 12 13 14 14 15 16 	s.color = gray, s.d = 0 $\triangleright$ initialize a queue in Q ENQUEUE(Q,s) while Q is not empty x = DEQUEUE(Q) forEach node y in x.Adj if y.color == white then
$\begin{array}{c} 9\\ 9\\ -10\\ 11\\ 12\\ 13\\ 14\\ -15\\ 16\\ -17\\ 18\\ \end{array}$	s.color = gray, s.d = 0 $\triangleright$ initialize a queue in Q ENQUEUE(Q,s) while Q is not empty x = DEQUEUE(Q) forEach node y in x.Adj if y.color == white then /* Node y is undiscovered. */
9 10 11 12 13 14 15 16 	$s.color = \operatorname{gray}$ , $s.d = 0$ $\triangleright$ initialize a queue in $Q$ $ENQUEUE(Q,s)$ while $Q$ is not empty $x = DEQUEUE(Q)$ forEach node $y$ in $x.Adj$ if $y.color = =$ white then $/*$ Node $y$ is undiscovered. $*/$ $y.color = \operatorname{gray}$ , $y.d = x.d + 1$ , $y.\pi = x$
9 10 11 12 13 14 15 16 	$s.color = \operatorname{gray}$ , $s.d = 0$ $\triangleright$ initialize a queue in $Q$ $ENQUEUE(Q,s)$ while $Q$ is not empty $x = DEQUEUE(Q)$ forEach node $y$ in $x.Adj$ if $y.color = =$ white then $/*$ Node $y$ is undiscovered. $*/$ $y.color = \operatorname{gray}$ , $y.d = x.d + 1$ , $y.\pi = x$ $ENQUEUE(Q, y)$
9 10 11 12 13 14 15 16 	s.color = gray, s.d = 0 $\triangleright$ initialize a queue in $Q$ ENQUEUE $(Q, s)$ while $Q$ is not empty x = DEQUEUE(Q) forEach node $y$ in $x.Adj$ if $y.color ==$ white then /* Node $y$ is undiscovered. $*/y.color = gray, y.d = x.d + 1, y.\pi = xENQUEUE(Q, y)end$
9 10 11 12 13 14 15 16 	s.color = gray, $s.d = 0$ $\triangleright$ initialize a queue in $Q$ ENQUEUE $(Q, s)$ while $Q$ is not empty x = DEQUEUE(Q) forEach node $y$ in $x.Adj$ if $y.color ==$ white then /* Node $y$ is undiscovered. $*/y.color = gray, y.d = x.d + 1, y.\pi = xENQUEUE(Q, y)endendend$
9 10 11 12 13 14 15 16 	s.color = gray, s.d = 0 $\triangleright$ initialize a queue in Q ENQUEUE(Q,s) while Q is not empty x = DEQUEUE(Q) forEach node y in x.Adj if y.color == white then /* Node y is undiscovered. */ y.color = gray, y.d = x.d + 1, y.\pi = x ENQUEUE(Q,y) end end x.color = black end

Rema	arks																
1. The	e enti	re gra	ph G	is giv	ven a	s an a	argum	nent.	This	is inte	nded	to re	prese	ent the	set	of ver	tices
and th	ne ad	jacen	cy set	ts x.A	<i>dj</i> for	each	node	э.									
2. Init	ializa	tion is	s done	e in <mark>fo</mark>	orEac	h loo	p of li	ne 5.									
3. At I	ine 1	2, s is	the c	only e	eleme	nt in	Q. He	ence i	n the	first it	eratio	on of t	the w	hile lo	ор, х	c = s.	
4. In e	each	iterati	on of	the v	/hile l	оор											
- )	k is re	emove	d fror	n Q	Voro		tiante	م ام ا	ho fo	r E o ob	loon	oflin	0.16				
ة - ر -	all no x is e	des a ventua	ally co	plored	x are d blac	inves k	stigate	a in t	ne to	reach	юор	of IIn	e 16				
5. A n	ode	can or	nly be	add	ed on	ce to	Q.										
Examp	ole																
				A.			$\bigcirc$			×							
				Ŷ		7	Y.		~	5							
				1.				/									
				$\mathcal{A}$			E			d							
							U	-		M							
Execu	ute B	READ	TH-F	IRST	_SEA	RCH	l with	s = D	)								
							_										

item	0						
	0	1	2	3	4		
Q	D	FB	CF	С	A		
A.d	00	Ø	00	00	3		
B.d	00	1	1	1			
C.d	<b>0</b> 40	00	る	a	a		
D.d	0	0	0	0	0	0	 <u> </u>
F.d	20	1	1	1	1	Gr	
H.d	00	Co	00	00	00		
Α.π	NIL	NIL	NIL	NIL	С		
Β.π	NIL	D	D	D	D		
С.π	NIL	NIL	В	В	В		
D.π	NIL	NIL	NIL	NIL	NIL		
F.π	NIL	D	D	D	D		
Η.π	NIL	NIL	NIL	NIL	NIL		
A.color	white	white	white	white	arov		
B.color	white	aray	black	block	glay		
C.color	white	white	grav	drav	block		
D.color	gray	black	black	black	hlack		
F.color	white	gray	grav	black	hlack		
H.color	white	white	white	white	white		
					winte		

⊢ınal	resu	ts:											
x		A	В	С		D	F		F	4			
x.d		3	1	2		0	1		¢	)			
Χ.π		С	D	В		NIL	[	D	Ν	۱L			
X.CC	olor												
		black	black	b	lack	black		black	<u>د</u>	whi	te	 	
4. Re	esult	s and i	nterpreta	ation									
Q: Ho	ow do	we know	w that <i>x.d</i> is	s the len	gth of	f the short	est pa	ath fro	m s to	<b>) Χ</b> , δ	(s, x)?		
Q: Ho	ow do	we know	w that <i>x.d</i> is	s the len	gth of Q in in	f the short	est pa	ath fro	m s to	) <i>Χ</i> , δ	( <i>s, x</i> )?		
<b>Q</b> : Ho <b>A</b> : Th	ow do ne noo	we knov des are p	w that <i>x.d</i> is out into the	s the len queue (	gth of Q in in	f the shortencreasing o	est pa order	ath fro of the	m s to their	ο <i>x</i> , δ dista	( <i>s, x</i> )? ance.		
<b>Q</b> : Ho <b>A</b> : Th	ow do ne noo	we knov des are p	w that <i>x.d</i> is out into the	s the len queue (	gth of Q in in	f the shorte	est pa order	ath fro of the	m s to	ο <i>x</i> , δι dista	( <i>s, x</i> )? ance.		
Q: Ho A: Th Fror	ow do ne noo m exa	we know des are p ample:	w that <i>x.d</i> is	s the len queue (	gth of Q in in	f the shorte	est pa order	ath fro of the	m s to their	ο <i>x</i> , δ dista	( <i>s, x</i> )? ance.		
Q: Ho A: Th Fror Q	ow do ne noo m exa :	we know des are p ample: A	w that <i>x.d</i> is out into the	s the len queue (	gth of Q in in	f the short ncreasing o	est pa order D	ath fro of the	m s to	ο <i>x</i> , δι dista	( <i>s, x</i> )? ance.		
Q: Ho A: Th Fror Q	ow do ne noo m exa : d	we know des are p ample: A 3	w that <i>x.d</i> is out into the C	s the len queue ( F 1	gth of Q in in	f the short ncreasing of B	est pa order D	ath fro of the	m s to	ο x, δι dista	( <i>s, x</i> )? ance.		
Q: Ho A: Th Fror Q	ow do ne noo m exa : d	we know des are p ample: A 3	w that <i>x.d</i> is out into the	s the len queue o F 1	gth of Q in in	f the short ncreasing o B 1	est pa order D 0	ath fro of the	m s to	ο <i>x</i> , δι dista	( <i>s, x</i> )? ance.		
Q: Ho A: Th Fror Q xc. o Q: Ho	ow do ne noo m exa : d	we know des are p ample: A 3 we know	w that <i>x.d</i> is out into the C 2 w what nod	s the len queue o F 1 es are re	gth of Q in in	f the short ncreasing of B 1 able from s	est pa order D 0 ?	ath fro of the	m s to	ο x, δι dista	( <i>s, x</i> )? ance.		
Q: Ho A: Th Fror Q xc. Q: Ho A1: If	ow do ne noo m exa : d ow do x is r	we know des are p ample: A 3 we know	w that <i>x.d</i> is out into the C Q w what nod	s the len queue ( F 1 es are re	gth of Q in in eacha	f the short ncreasing of B 1 able from s ot white.	est pa order D 0 ?	ath fro of the	m s to	ο x, δι dista	( <i>s, x</i> )? ance.		
Q: Ho A: Th Fror Q x Q: Ho A1: If A2: If	ow do ne noo m exa c d w do x is r x is r	we know des are p ample: A 3 we know eachable	w that <i>x.d</i> is out into the C Q w what nod e from <i>s</i> the	s the len queue of F 1 es are ro en <i>x.colo</i>	gth of Q in in eacha pr is n	f the short ncreasing of B 1 able from s ot white.	est pa order D 0 ?	ath fro of the	m s to	ο x, δι dista	( <i>s, x</i> )? ance.		
Q: Ho A: Th Fror Q x A1: If A2: If	ow do ne noo m exa : d w do x is r x is r	we know des are p ample: A 3 we know eachable	w that <i>x.d</i> is out into the C Q w what nod e from <i>s</i> the	s the len queue of F 1 es are re en <i>x.colo</i> en <i>x.colo</i>	gth of Q in in eacha or is n not ∞	f the short acreasing of B 1 able from s ot white.	est pa order D 0 ?	ath fro of the	m s to	ο x, δι dista	( <i>s, x</i> )? ance.		

<b>A</b> : The	root is s.	For each noc	de x, we	can constr	uct the rev	verse path	(from x to	s) using	9
parent	attribute >	κ.π.					· · · · · · · · · · · · · · · · · · ·		
x	A	В	С	D	F	Н			
Χ.π	С	D	В	NIL	D	NI	-		(
BF-tree	e from exa	ample:		B) Z					
		C			(F)				
		A							
Remar are as s	<b>k</b> : The pa short, but	ths from s to z none that are	x is the s e shorter	shortest ler r.)	ngth path.	(There ma	y be othe	er paths	that
Remar are as : Q: Cal	<b>k</b> : The pa short, but n we use	ths from s to z none that are Breadth-first-	x is the s shorter search c	shortest ler r.) on an undire	ngth path. ected grap	(There ma	y be othe	er paths	that
Remar are as s Q: Ca A: Yes	<b>k</b> : The pa short, but n we use s.	ths from s to z none that are Breadth-first-s	x is the s shorter search c	shortest ler r.) on an undire	ngth path. ected grap	(There ma	y be othe	er paths	that
Remar are as s Q: Car A: Yes Rema - if x t	k: The pa short, but n we use s. <b>rks for u</b> pelongs to	ths from <i>s</i> to <i>z</i> none that are Breadth-first-s ndirected gra	x is the s shorter search c aph: v belong	shortest ler r.) on an undir s to <i>x.Adj</i>	ngth path. ected grap	(There ma	y be othe	er paths	that
Remar are as s Q: Cal A: Yes Rema - if <i>x</i> i - if <i>x</i> i	k: The pa short, but n we use s. <b>rks for u</b> pelongs to s reachat	ths from <i>s</i> to <i>z</i> none that are Breadth-first-s ndirected gra b <i>y.Adj</i> , then <i>y</i> ole from <i>s</i> , the	x is the s shorter search c ph: belong:	shortest ler r.) on an undir s to <i>x.Adj</i> eachable fro	ngth path. ected grap	(There ma	y be othe	er paths	that
Remar are as s Q: Cal A: Yes Rema - if x k - if x i - <i>x.d</i> is	k: The pa short, but n we use s. <b>rks for u</b> belongs to s reachat s the dista	ths from <i>s</i> to <i>z</i> none that are Breadth-first-s ndirected gra b <i>y.Adj</i> , then <i>y</i> ble from <i>s</i> , the ance both fron	x is the s shorten search c belong: n s is re n s to x a	shortest ler r.) on an undir s to <i>x.Adj</i> eachable fro and from <i>x</i> t	ngth path. ected grap om <i>x</i> to <i>s</i>	(There ma	y be othe	er paths	that
Remar are as s Q: Car A: Yes Rema - if x t - if x i - x.d is	k: The pa short, but n we use s. <b>rks for u</b> belongs to s reachat s the dista	ths from <i>s</i> to <i>z</i> none that are Breadth-first-s ndirected gra b <i>y.Adj</i> , then <i>y</i> ble from <i>s</i> , the ance both fron	x is the s shorten search c belong n s is re n s to x a	shortest len r.) on an undire s to <i>x.Adj</i> eachable fro	ngth path. ected grap	(There ma	y be othe	er paths	that
Remar are as s Q: Cal A: Yes Rema - if x t - if x i - x.d is	k: The pa short, but n we use s. <b>rks for u</b> pelongs to s reachat s the dista	ths from <i>s</i> to <i>z</i> none that are Breadth-first-s ndirected gra b <i>y</i> . <i>Adj</i> , then <i>y</i> ble from <i>s</i> , the ance both fron	x is the s shorten search c belong n s is re n s to x a	shortest ler r.) on an undir s to <i>x.Adj</i> eachable fro and from <i>x</i> t	ected grap	(There ma	y be othe	er paths	that

