space complexity of bfs and dfs

CSCI 373

	time	space
bfs	$O(b^d)$	?
dfs	$O(b^m)$	?

d — **solution depth**

the length of the shortest search path that leads to a final state

m — **maximum depth**

the length of the longest search path

b — **branching factor**

the maximum number of successors of a search node

space can be measured by the maximum number of nodes in the **container** at any given time

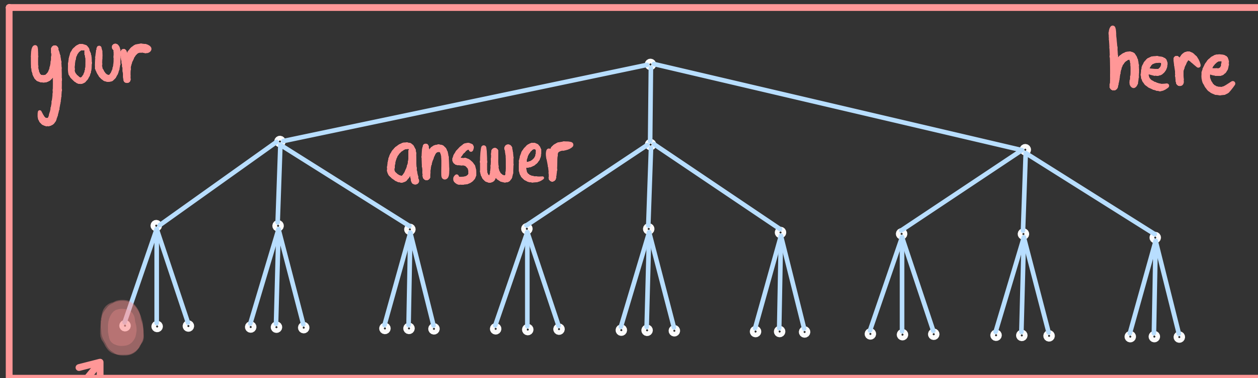
	time	space
bfs	$O(b^d)$	?
dfs	$O(b^m)$	?

d — solution depth: the length of the shortest search path that leads to a final state

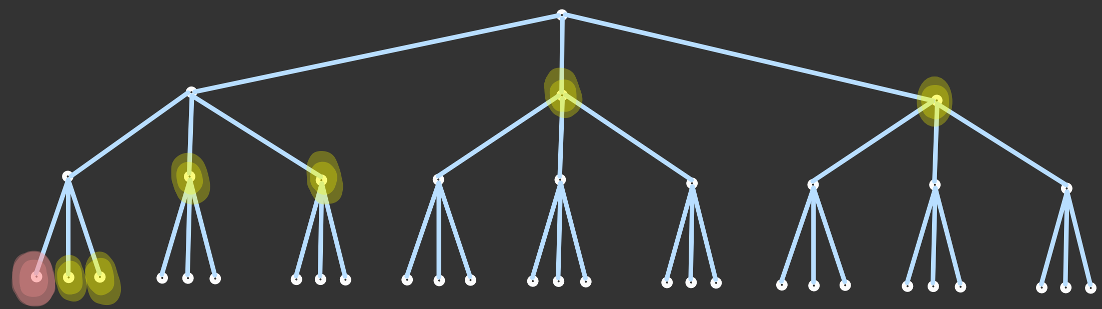
m — maximum depth: the length of the longest search path

b — branching factor: the maximum number of successors of a search node

consider dfs

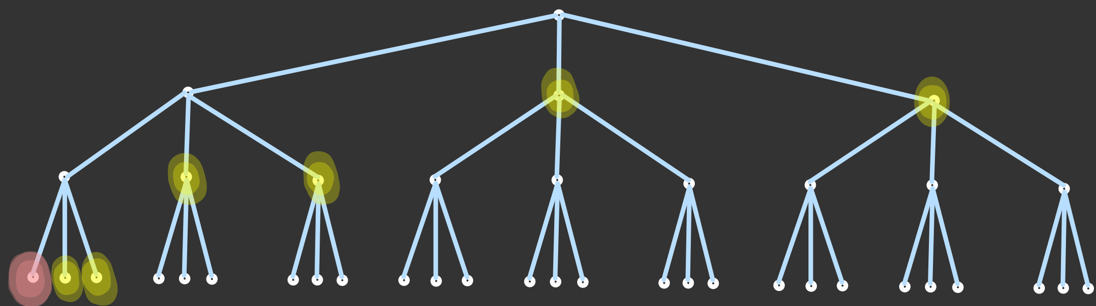


which nodes could be in the container at the point we visit the highlighted node?



- siblings
- siblings of ancestors

- siblings
- siblings of ancestors

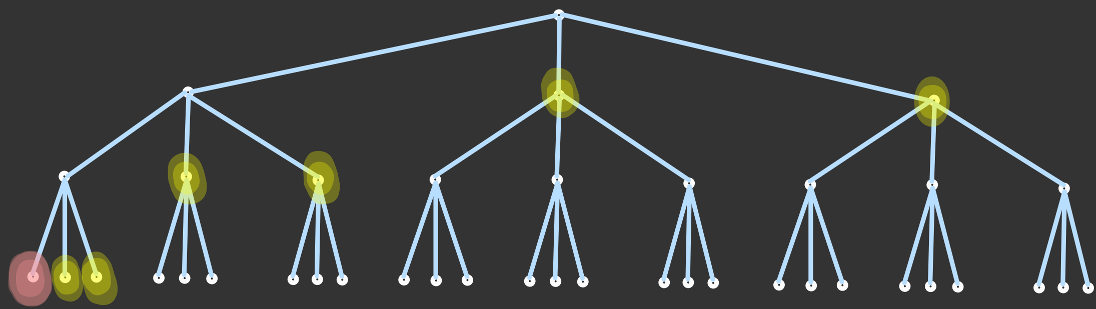


your answer here

how many nodes is this, if the highlighted node is at

depth k ?

- siblings
- siblings of ancestors

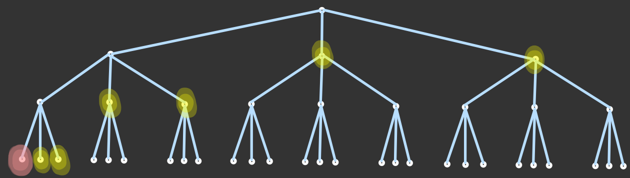


$$\begin{array}{r} b-1 \\ b-1 \\ +\ b-1 \\ \hline k(b-1) \end{array}$$

how many nodes is this, if the highlighted node is at

depth k ?

$O(bk)$



$$\begin{array}{r} b-1 \\ b-1 \\ +\ b-1 \\ \hline k(b-1) \end{array}$$

how many nodes is this, if the highlighted node is at depth k?

$O(bk)$

given this observation, what is the maximum number of nodes in the **container** at any given time?

your answer here

d solution depth

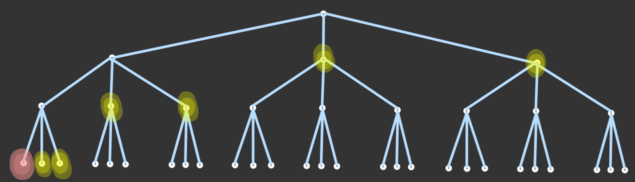
the length of the shortest search path that leads to a final state

m maximum depth

the length of the longest search path

b branching factor

the maximum number of successors of a search node



$$\begin{array}{r} b-1 \\ b-1 \\ +\ b-1 \\ \hline k(b-1) \end{array}$$

how many nodes is this, if the highlighted node is at depth k?

$O(bk)$

given this observation, what is the maximum number of nodes in the **container** at any given time?

$O(bm)$

d — **solution depth**
the length of the shortest search path that leads to a final state

m — **maximum depth**
the length of the longest search path

b — **branching factor**
the maximum number of successors of a search node

	time	space
bfs	$O(b^d)$	?
dfs	$O(b^m)$	$O(bm)$

d — **solution depth**

the length of the shortest search path that leads to a final state

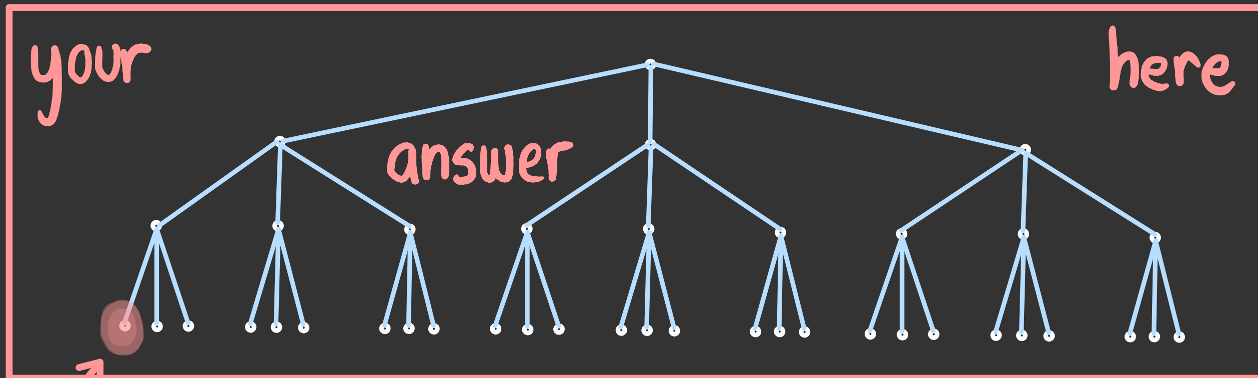
m — **maximum depth**

the length of the longest search path

b — **branching factor**

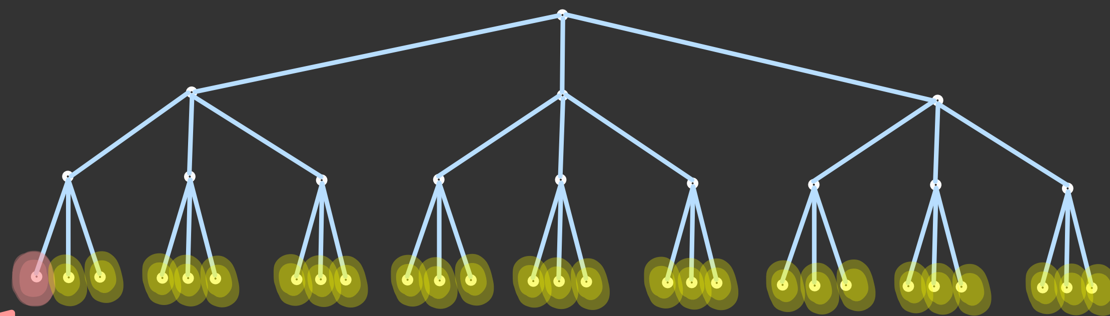
the maximum number of successors of a search node

now consider bfs



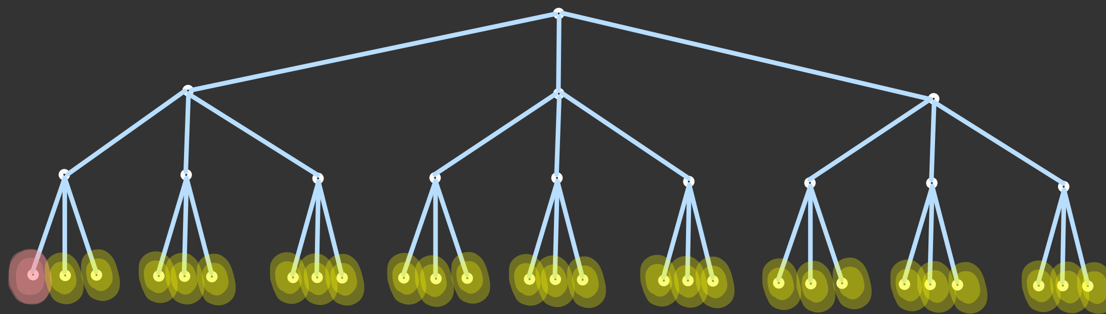
which nodes could be in the container at the point we visit the **highlighted** node?

now consider bfs



which nodes could be in the container at the point we visit the highlighted node?

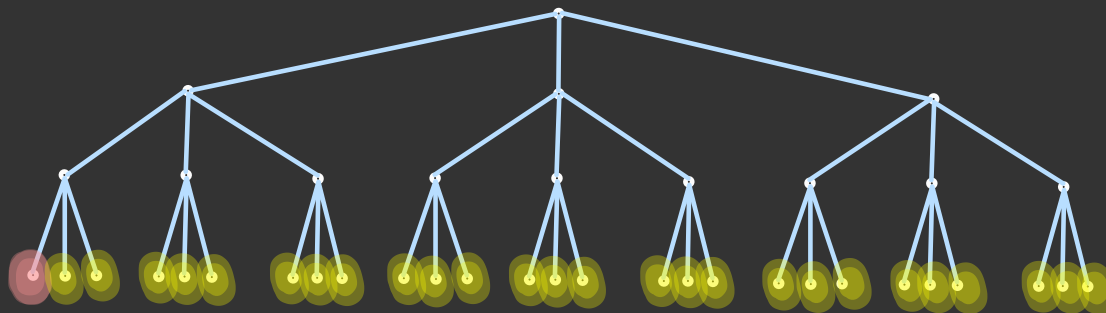
now consider bfs



how many nodes is this, if the highlighted node is at depth k?

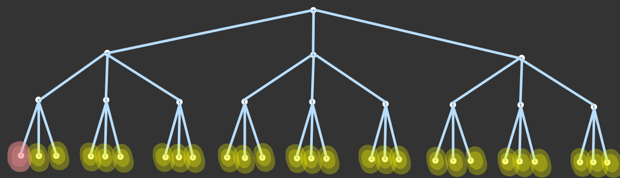
your answer here

now consider bfs



how many nodes is this, if the highlighted node is at depth k?

$O(b^k)$



how many nodes is this, if the highlighted node is at depth k?

$O(b^k)$

given this observation, what is the maximum number of nodes in the

container

at any given time?

your answer here

d solution depth

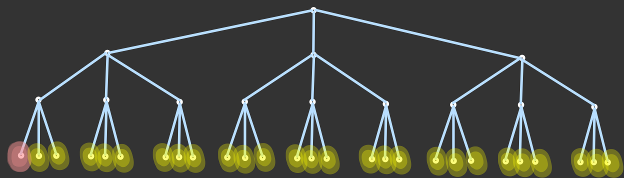
the length of the shortest search path that leads to a final state

m maximum depth

the length of the longest search path

b branching factor

the maximum number of successors of a search node



how many nodes is this, if the highlighted node is at depth k?

$O(b^k)$

given this observation, what is the maximum number of nodes in the **container** at any given time?

$O(b^d)$

---

d — **solution depth**

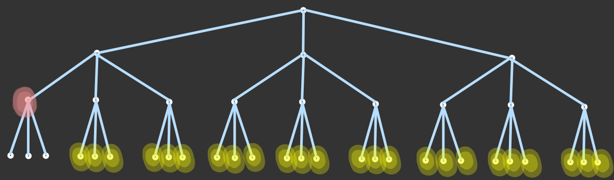
the length of the shortest search path that leads to a final state

m — **maximum depth**

the length of the longest search path

b — **branching factor**

the maximum number of successors of a search node



how many nodes is this, if the highlighted node is at depth k?

$O(b^{k+1})$

given this observation, what is the maximum number of nodes in the **container** at any given time?

or perhaps... $O(b^{d+1})$

d — **solution depth**
the length of the shortest search path that leads to a final state

m — **maximum depth**
the length of the longest search path

b — **branching factor**
the maximum number of successors of a search node

	time	space
bfs	$O(b^d)$	$O(b^d)$
dfs	$O(b^m)$	$O(bm)$

d — **solution depth**: the length of the shortest search path that leads to a final state

m — **maximum depth**: the length of the longest search path

b — **branching factor**: the maximum number of successors of a search node

	time	space
bfs	$O(b^d)$	$O(b^d)$
dfs	$O(b^m)$	$O(bm)$

what matters more in practice: space or time?

your opinion here

d solution depth

the length of the shortest search path that leads to a final state

m maximum depth

the length of the longest search path

b branching factor

the maximum number of successors of a search node

	time	space
bfs	$O(b^d)$	$O(b^d)$
dfs	$O(b^m)$	$O(bm)$

what matters more in practice: space or time?

space

d **solution depth**

the length of the shortest search path that leads to a final state

m **maximum depth**

the length of the longest search path

b **branching factor**

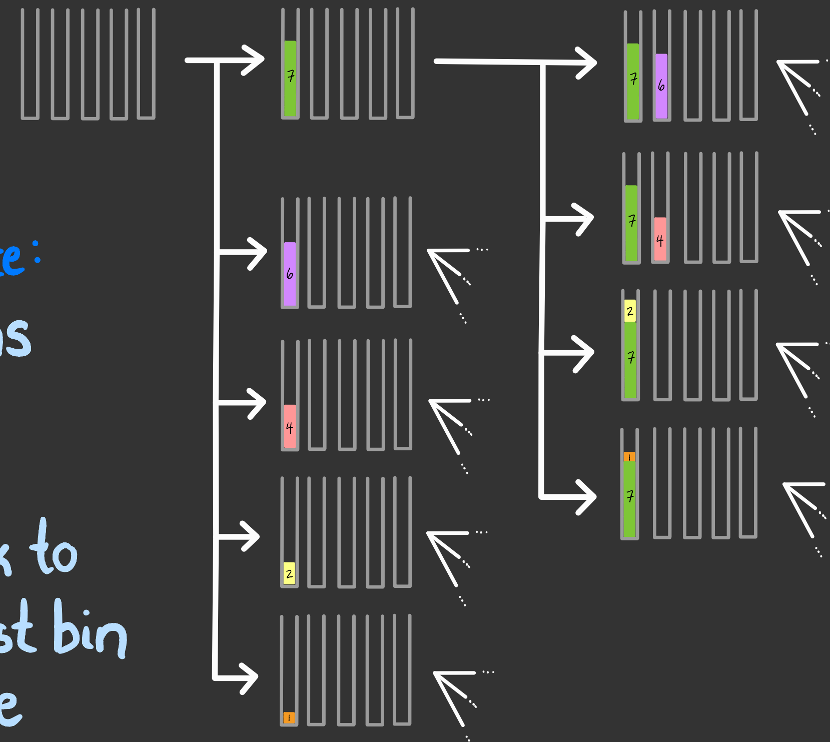
the maximum number of successors of a search node

why space?

consider this state machine for bin packing

initial state: empty bins

action: add a block to the leftmost bin with space



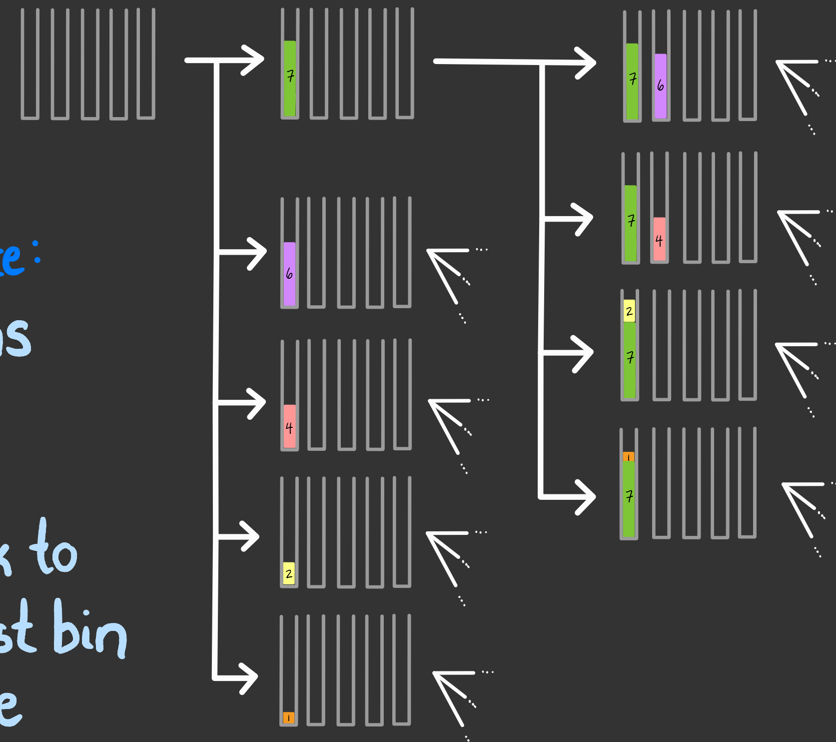
if there are n blocks, then branching factor b = ?

solution depth d = ?

why space?

initial state:
empty bins

action:
add a block to the leftmost bin with space



if there are n blocks, then branching factor $b = n$
solution depth $d = n$

why space?

	time	space
bfs	$O(b^d)$	$O(b^d)$
dfs	$O(b^m)$	$O(bm)$

so the space required by bfs is $O(n^n)$

even for just 10 blocks, we require $10^{10} \approx$ 10GB of memory (and that's if we can cram a search into a byte) why space?

	time	space
bfs	$O(b^d)$	$O(b^d)$
dfs	$O(b^m)$	$O(bm)$

this seriously limits the applicability of bfs in practice