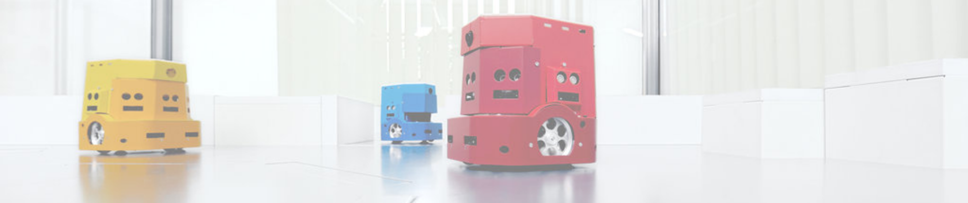
Obstacle Avoidance Algorithms



Miroslav Kulich

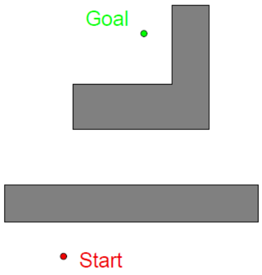
Czech Technical University in Prague
Czech Institute of Informatics, Robotics and Cybernetics
Intelligent and Mobile Robotics Group

http://imr.ciirc.cvut.cz/people/Mirek

Bug algorithms

Insect inspired

- Point robot operating on the plane
- Only local knowledge of the environment and a global goal
- Known direction to goal
- Otherwise local sensing (walls/obstacles and encoders)
- Robot can measure distance $d(x, y)$ between points x and y
- Reasonable world
    - finitely many obstacles in any finite area
    - a line will intersect an obstacle finitely many times
    - Workspace is bounded

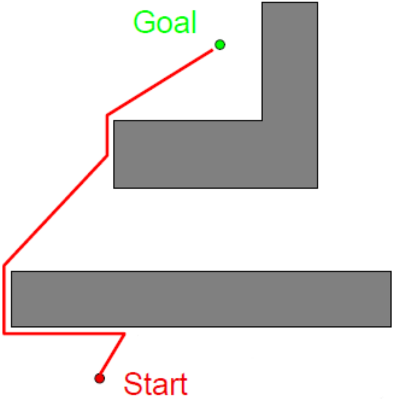


Beginner's strategy

,,Bug0" algorithm

- Known direction to goal
- Otherwise local sensing

1. Head toward goal.
2. Follow obstacles until you can head toward goal again.
3. Continue.



What can go wrong? Find a map that will foil Bug 0.

Assume a left-turning robot. Turning direction might be decided beforehand.

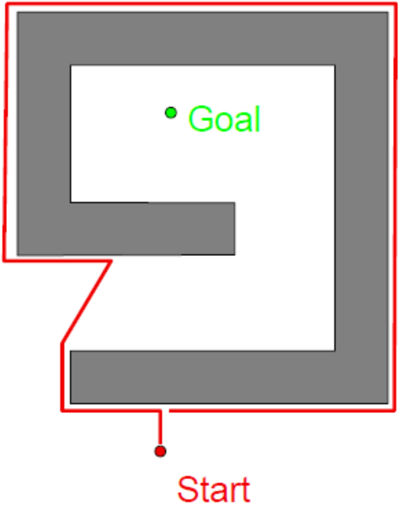
Beginner's strategy

,,Bug0" algorithm

1. Head toward goal.
2. Follow obstacles until you can head toward goal again.
3. Continue.

How can we improve Bug 0?

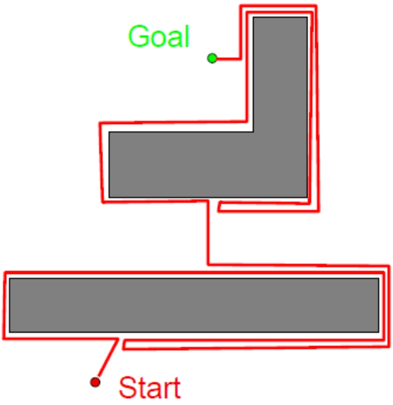
- Add memory
  - What information is available?
- Encoders
  - Keep track of robot's own motion



Bug 1

- Known direction to goal
- Otherwise local sensing
  - wall/obstacles and encoders

1. Head toward goal.
2. If an obstacle is encountered, circumnavigate it AND remember how close you get to the goal.
3. Return to that closest point and continue.



- Takes longer to run.
- Requires more computational effort.

Bug 1 more formally

Let $q_0^L = q_{start}$
$i = 1$
loop
  repeat
    from q_{i-1}^L move toward q_{goal}
  until goal is reached or obstacle encountered at q_i^H
  if goal is reached **then**
    exit
  end if
  repeat
    follow boundary recording point q_i^L with shortest distance to goal
  until q_{goal} is reached or q_i^H is re-encountered
  if goal is reached **then**
    exit
  end if
  Go to q_i^L
  if move toward q_{goal} moves into obstacle **then**
    exit with failure
  else
    $i = i + 1$

Quiz - Bug 1 analysis

What are upper/lower bounds on the path length that the robot takes?

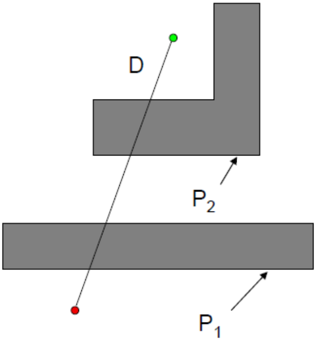
D = straight-line distance from start to goal
P_i = perimeter of the i^{th} obstacle

Lower bound

What is the shortest distance it might travel?

Upper bound

What is the longest distance it might travel?



Quiz - Bug 1 analysis

What are upper/lower bounds on the path length that the robot takes?

D = straight-line distance from start to goal

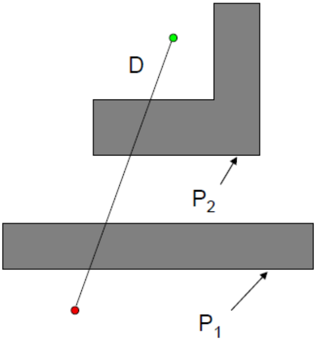
P_i = perimeter of the i^{th} obstacle

Lower bound

What is the shortest distance it might travel? D

Upper bound

What is the longest distance it might travel?



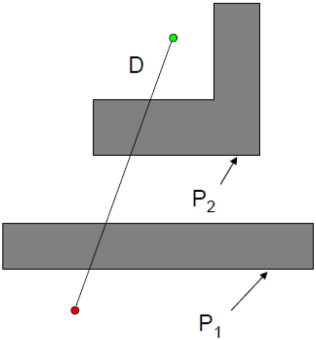
Quiz - Bug 1 analysis

What are upper/lower bounds on the path length that the robot takes?

D = straight-line distance from start to goal
P_i = perimeter of the i^{th} obstacle

Lower bound
What is the shortest distance it might travel? D

Upper bound
What is the longest distance it might travel? $D + 1.5 \sum_i P_i$



Quiz - Bug 1 analysis

What are upper/lower bounds on the path length that the robot takes?

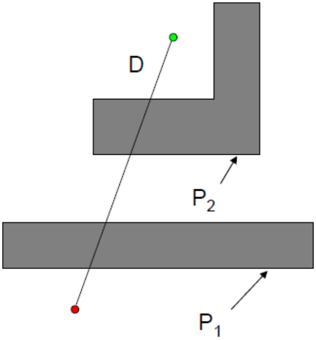
D = straight-line distance from start to goal
P_i = perimeter of the i^{th} obstacle

Lower bound

What is the shortest distance it might travel? D

Upper bound

What is the longest distance it might travel? $D + 1.5 \sum_i P_i$

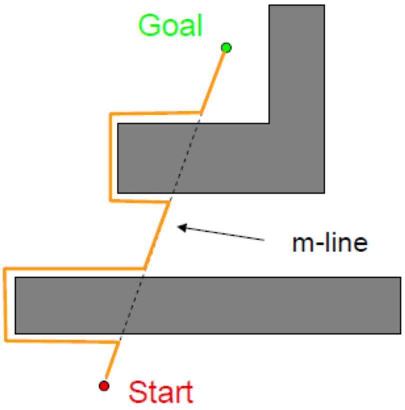


What is an environment where the upper bound is required?

A better bug?

,,Bug 2" algorithm

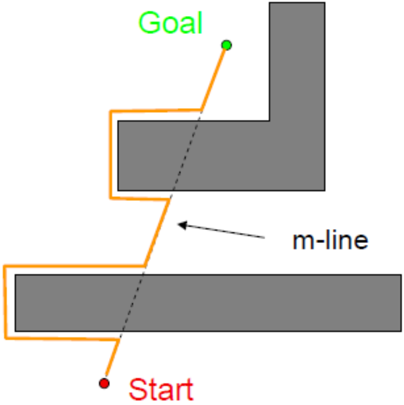
1. Head toward goal.
2. If an obstacle is on the way, follow it until you hit the m-line again.
3. Leave the obstacle and continue toward the goal.



A better bug?

,,Bug 2" algorithm

1. Head toward goal.
2. If an obstacle is on the way, follow it until you hit the m-line again.
3. Leave the obstacle and continue toward the goal.

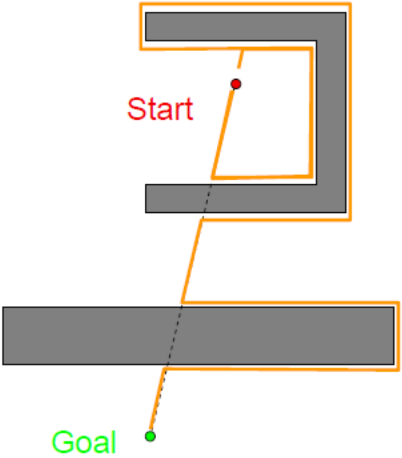


What can go wrong? Find maps that will foil Bug 2.

A better bug?

Whoops! Infinite loop

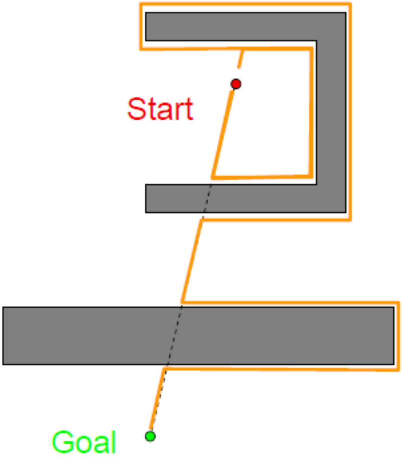
1. Head toward goal.
2. If an obstacle is on the way, follow it until you hit the m-line again closer to the goal.
3. Leave the obstacle and continue toward the goal.



A better bug?

Whoops! Infinite loop

1. Head toward goal.
2. If an obstacle is on the way, follow it until you hit the m-line again closer to the goal.
3. Leave the obstacle and continue toward the goal.



Is this algorithm better or worse than Bug 1?

Bug 2 more formally

Let $q_0^L = q_{start}$
$i = 1$
loop
  repeat
    from q_{i-1}^L move toward q_{goal} along the m-line
  until goal is reached or obstacle encountered at q_i^H
  if goal is reached **then**
    exit
  end if
  repeat
    follow boundary
  until q_{goal} is reached or q_i^H is re-encountered or m-line is re-encountered, x is not q_i^H,
  $d(x, q_{goal}) < d(q_i^H, q_{goal})$ and way to goal is unimpeded
  if goal is reached **then**
    exit
  end if
  if q_i^H is reached **then**
    return failure
  else
    $q_i^L = m$
    $i = i + 1$
    continue
  end if
end loop

Head-to-head comparison

Draw world in which Bug 2 does better than Bug 1 (and vice versa)

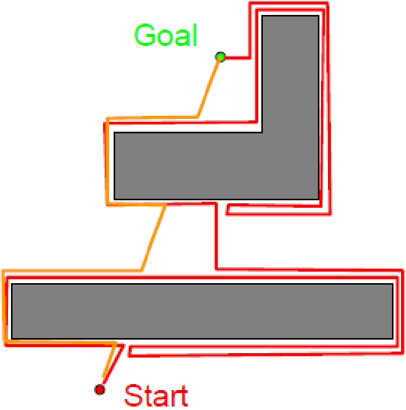
Bug 2 beats Bug 1

Bug 1 beats Bug 2

Head-to-head comparison

Draw world in which Bug 2 does better than Bug 1 (and vice versa)

Bug 2 beats Bug 1

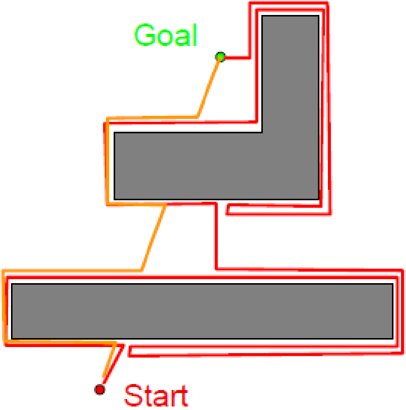


Bug 1 beats Bug 2

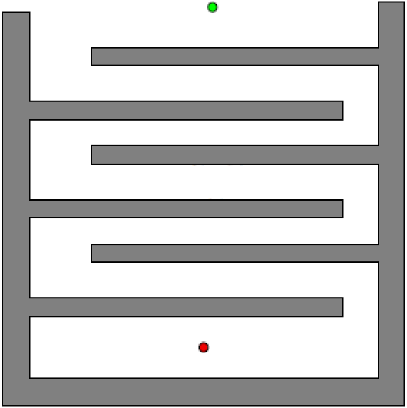
Head-to-head comparison

Draw world in which Bug 2 does better than Bug 1 (and vice versa)

Bug 2 beats Bug 1



Bug 1 beats Bug 2



Bug 1 vs. Bug 2

- Bug 1 is an exhaustive search algorithm
  - it looks at all choices before committing
- Bug 2 is a greedy algorithm
  - it takes the first thing that looks better
- In many cases, Bug 2 will outperform Bug 1, but.
- Bug 1 has a more predictable performance overall.

Quiz - Bug 2 analysis

What are upper/lower bounds on the path length that the robot takes?

D = straight-line distance from start to goal

P_i = perimeter of the i^{th} obstacle

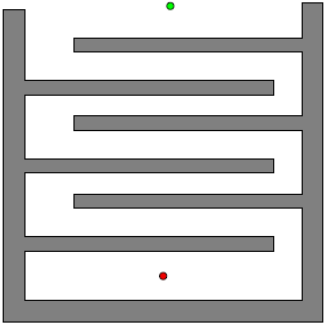
Lower bound

What is the shortest distance it might travel?

Upper bound

What is the longest distance it might travel?

n_i = # of m-line intersection of the i^{th} obstacle



Quiz - Bug 2 analysis

What are upper/lower bounds on the path length that the robot takes?

D = straight-line distance from start to goal

P_i = perimeter of the i^{th} obstacle

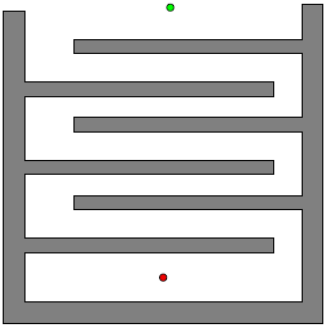
Lower bound

What is the shortest distance it might travel? D

Upper bound

What is the longest distance it might travel?

n_i = # of m-line intersection of the i^{th} obstacle



Quiz - Bug 2 analysis

What are upper/lower bounds on the path length that the robot takes?

D = straight-line distance from start to goal

P_i = perimeter of the i^{th} obstacle

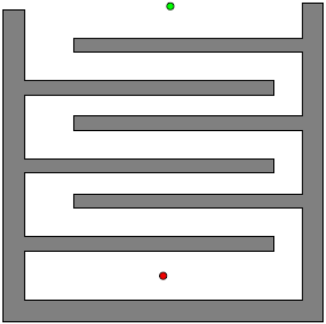
Lower bound

What is the shortest distance it might travel? D

Upper bound

What is the longest distance it might travel? $D + 1.5 \sum_i \frac{n_i}{2} P_i$

$n_i = \#$ of m-line intersection of the i^{th} obstacle



Quiz - Bug 2 analysis

What are upper/lower bounds on the path length that the robot takes?

D = straight-line distance from start to goal
P_i = perimeter of the i^{th} obstacle

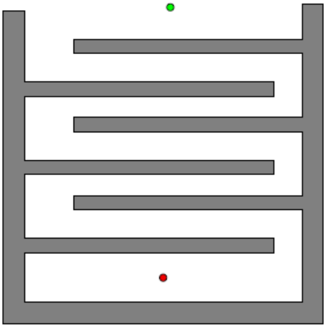
Lower bound

What is the shortest distance it might travel? D

Upper bound

What is the longest distance it might travel? $D + 1.5 \sum_i \frac{n_i}{2} P_i$

n_i = # of m-line intersection of the i^{th} obstacle



What is an environment where the upper bound is required?

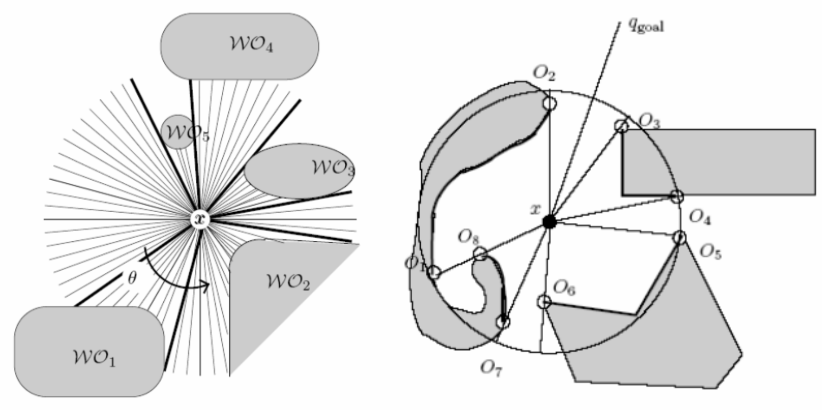
Tangent bug

A more realistic Bug

- As presented: global beacons plus contact-based wall following
- The reality: we typically use some sort of range sensing device that lets us look ahead (but has finite resolution and is noisy)
- Now, let us assume we have a range sensor...

Intervals of Continuity

- Tangent Bug relies on finding endpoints O_i of finite, continuous segments of ρ_R



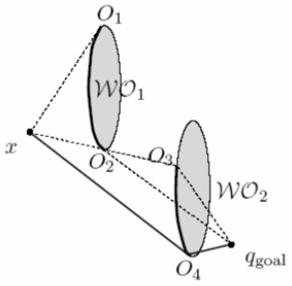
Tangent bug

Basic ideas

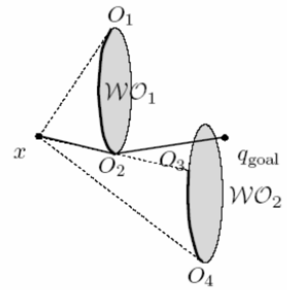
- Motion-to-Goal (two variations)
  - Move towards the goal until an obstacle is sensed between the robot and the goal
  - Move towards the O_i that maximally decreases a heuristic distance, e.g. $d(x, O_i) + d(O_i, q_{goal})$
- Follow obstacle
  - Started if the robot cannot decrease the heuristic distance
  - Continuously moves towards the on the followed obstacle in the same direction as the previous motion-to-goal
  - Back to motion-to-goal when it is ,,better" to do so

Heuristic example

At x the robot knows only what it sees and where the goal is,



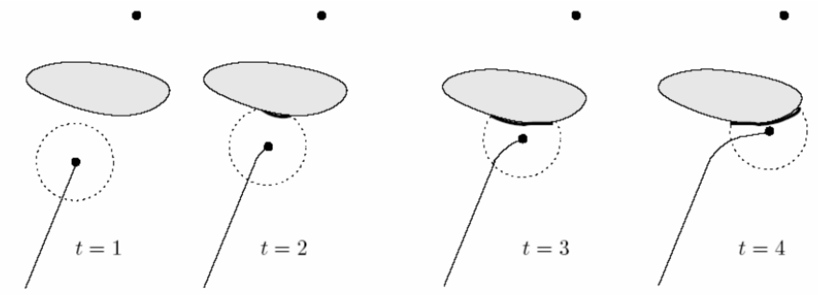
so it moves toward O_2. Note that the line connecting O_2 and goal passes through an obstacle.



so it moves toward O_4. Note that some ,,thinking" was involved and the line connecting O_4 and the goal passes through an obstacle.

Choose the point O_i that minimizes $d(x, O_i) + d(O_i, q_{goal})$.

Motion-To-Goal example

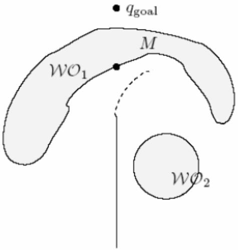


Choose the point O_i that minimizes $d(x, O_i) + d(O_i, q_{goal})$.

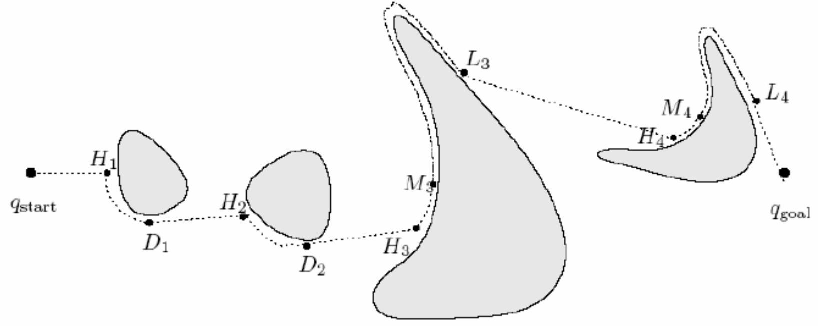
Boundary following

- Problem: What if this distance starts to go up?
- Answer: Start to act like a Bug and follow boundary!

- Move toward the O_i on the followed obstacle in the ,,chosen" direction while maintaining $d_{followed}$ and d_{reach}.
- $d_{followed}$ is the shortest distance between the sensed boundary and the goal
- d_{reach} is the shortest distance between blocking obstacle and goal (or my distance to goal if no blocking obstacle visible)
- Terminate when $d_{reach} < d_{followed}$

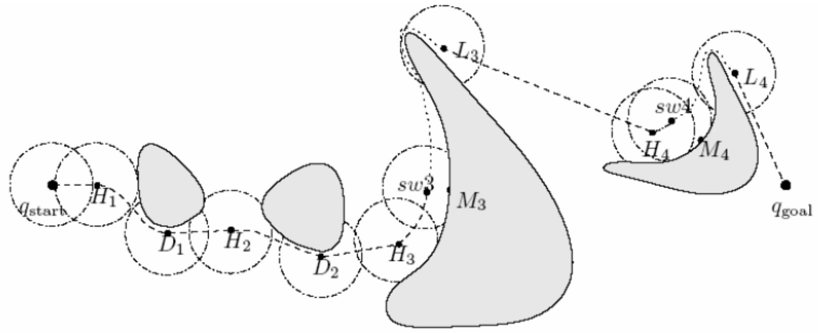


Example: zero sensor range

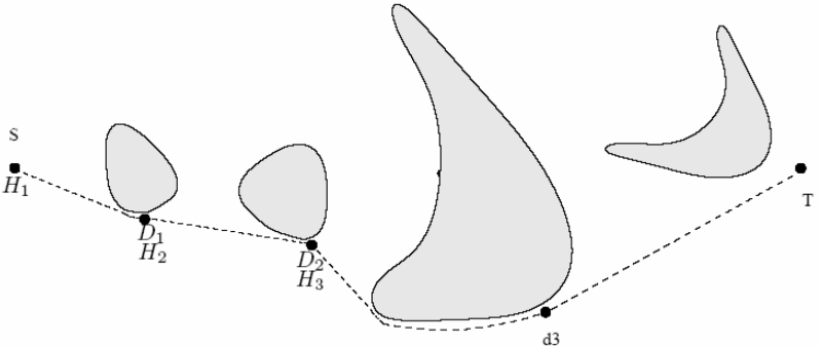


- Robot moves toward goal until it hits obstacle 1 at H_1.
- Pretend there is an infinitely small sensor range and the O_i, which minimizes the heuristic is to the right.
- Keep following obstacle until robot can go toward obstacle again.
- Same situation with second obstacle.
- At third obstacle, the robot turned left until it could not increase heuristic.

Example: Finite sensor range



Example: Infinite sensor range



Tangent bug algorithm

move towards the goal

repeat

Compute continuous range segments in view.

Move toward n in $\{T, O_i\}$ that minimizes

$h(x, n) = d(x, n) + d(n, q_{goal})$

until goal is encountered **or**

the value of $h(x, n)$ begins to increase

follow boundary continuing in same direction as before repeating

repeat

update $\{O_i\}$, d_{reach} and $d_{followed}$

until goal is reached **or**

a complete cycle is performed (goal is unreachable) **or**

$d_{reach} < d_{followed}$