Task Fuel [B] (pal)

https://szkopul.edu.pl/problemset/problem/5g0vDW-MvMGHfWQqh56jQKx1/site/?key=statement

Fuel [B]

Memory limit: 128 MB

In the good old days, all n towns in Byteland were connected by a dense network of two-way roads. The king of Byteland decided to reduce the number of roads, and asked his Chief Computer Scientist for advice. Now, as a result of this decision, Byteland has only $n-1$ two-way roads connecting pairs of towns in such a way that there is exactly one route between any two towns in Byteland. All the roads are of the same length.

Equipped with a car whose tank has exactly enough fuel to drive on m roads in Byteland, Byteasar has decided to organize a trip that maximizes the number of visited towns. He can start his trip in any of the towns in Byteland, and, similarly, his trip can end anywhere-not necessarily back at the starting town. In maximizing the number of visited towns, Byteasar is allowed to drive multiple times on the same road, in the same or the reverse direction. Your task is to help Byteasar by finding the maximum number of towns that can be visited on one full tank of fuel.

Input

The first line of the input contains two integers, n and m ($2 \le n \le 500\,000$, $1 \le m \le 200\,000\,000$), where n is the number of towns in Byteland (each numbered uniquely from $\{1, \ldots, n\}$), and m is the number of roads that can be traveled on one tank of fuel.

The next $n-1$ lines describe Byteland's road network. Each of these lines contains two integers, a and b ($1 \le a, b \le n$), indicating that towns a and b are connected with a two-way road.

Output

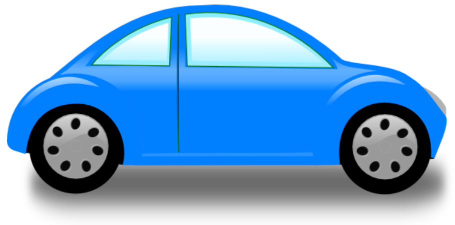
The output consists of one line containing exactly one integer: the maximum number of towns that can be visited on one full tank of fuel.

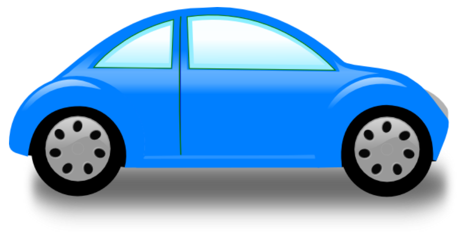
Send Feedback

Example

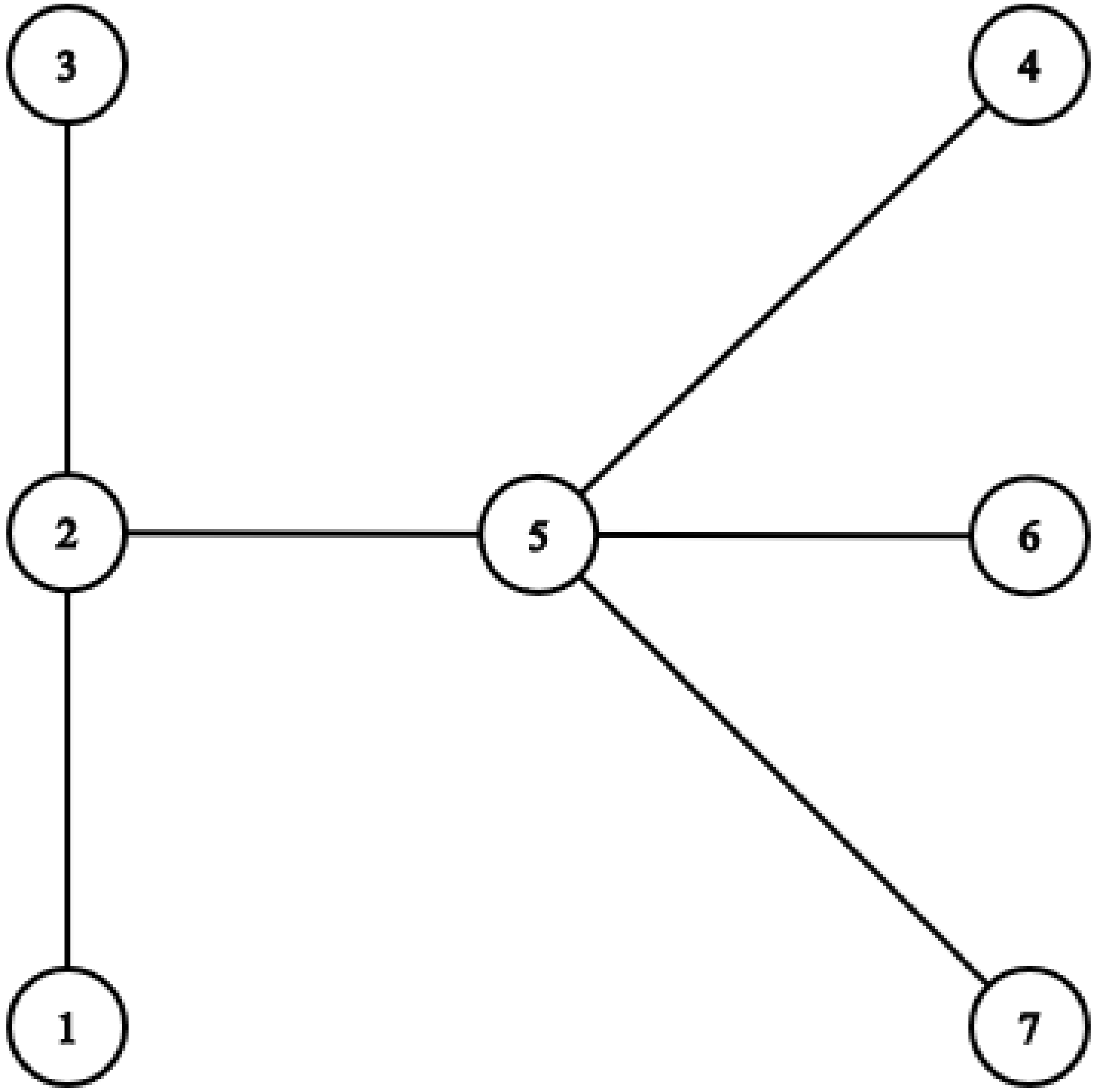
سامان قصد گشتن کشوری را دارد که شهرها و جاده‌های آن به شکل یک درخت n راسی است. برنامه‌ی سامان این است که با هواپیما به شهری دلخواه از این کشور سفر کند، سپس ماشینی کرایه کند و شروع به گشتن بکند و در نهایت در هر شهری که گشت و گذارش (یا بنزین ماشینش) تمام شد، با هواپیما به خانه بازگردد. بودجه‌ی او به اندازه‌ی خرید مقداری بنزین است که با آن می‌تواند حداکثر m جاده را بپیماید.

سامان می‌خواهد طوری مسیر گشت و گذارش را انتخاب کند که تعداد شهرهایی که از آن‌ها حداقل یک بار بازدید کرده بیشینه شود. الگوریتمی بهینه ارائه دهید که این بیشینه تعداد شهرهای ممکن را بدست آورد.

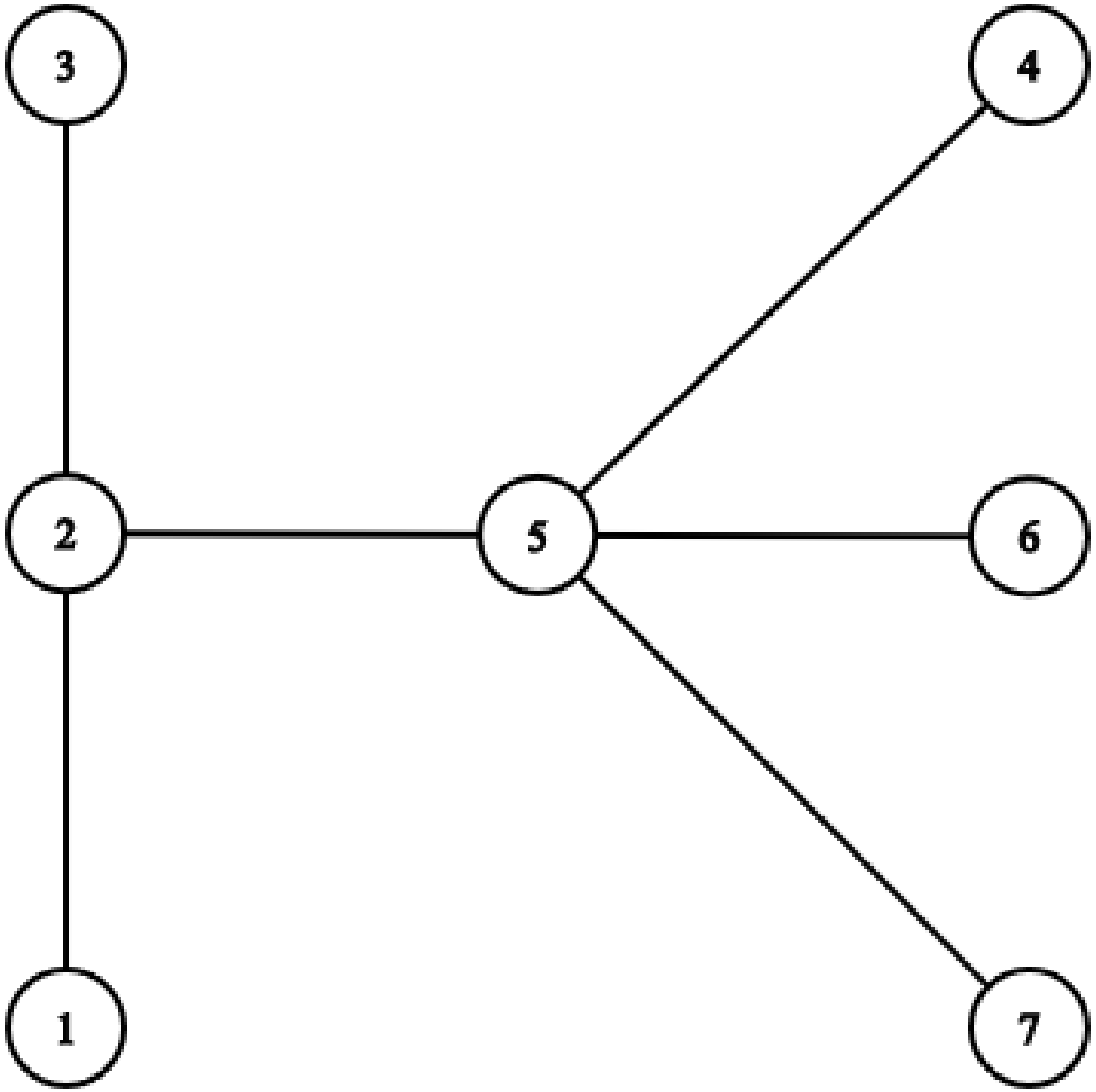




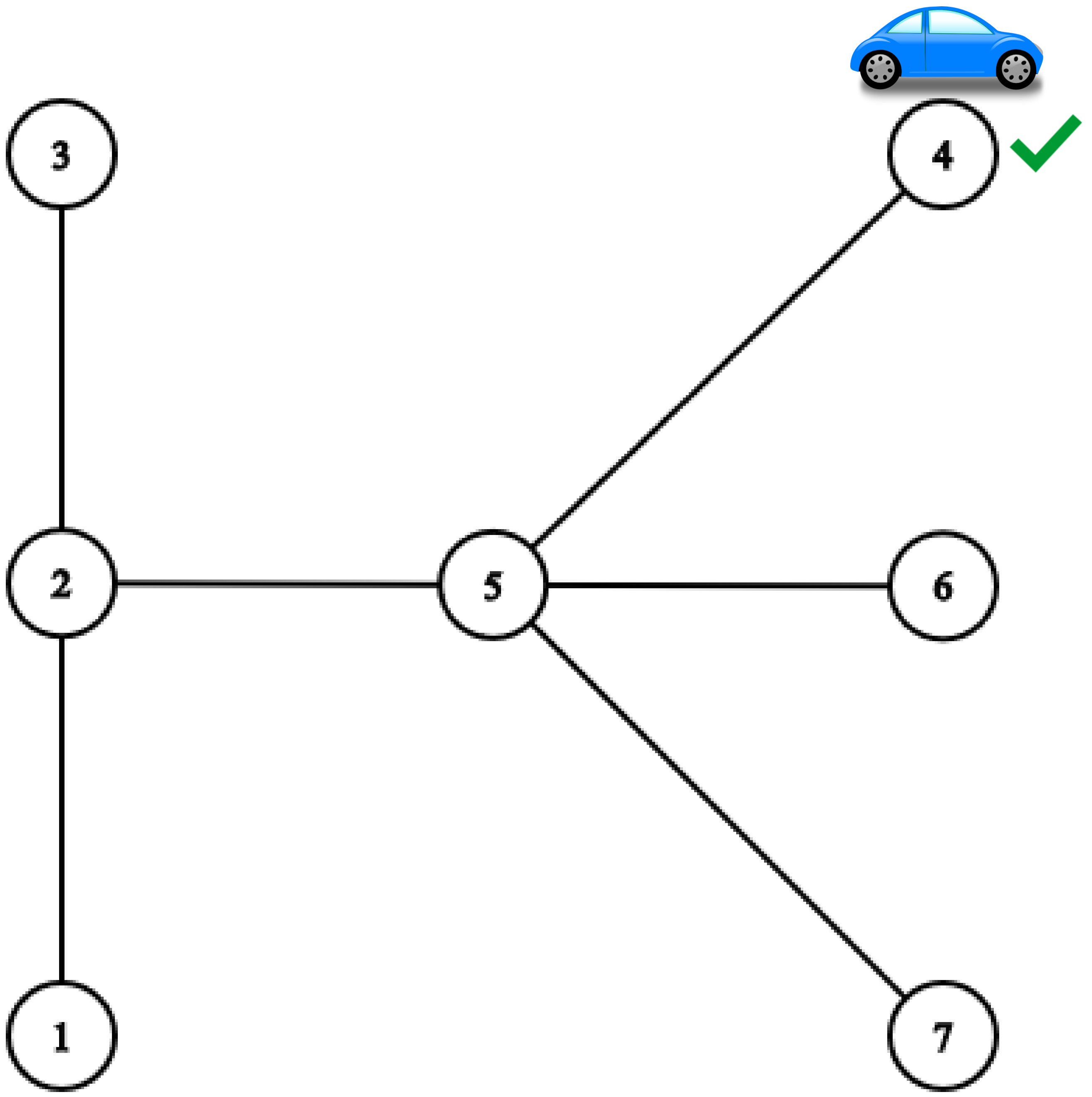
m



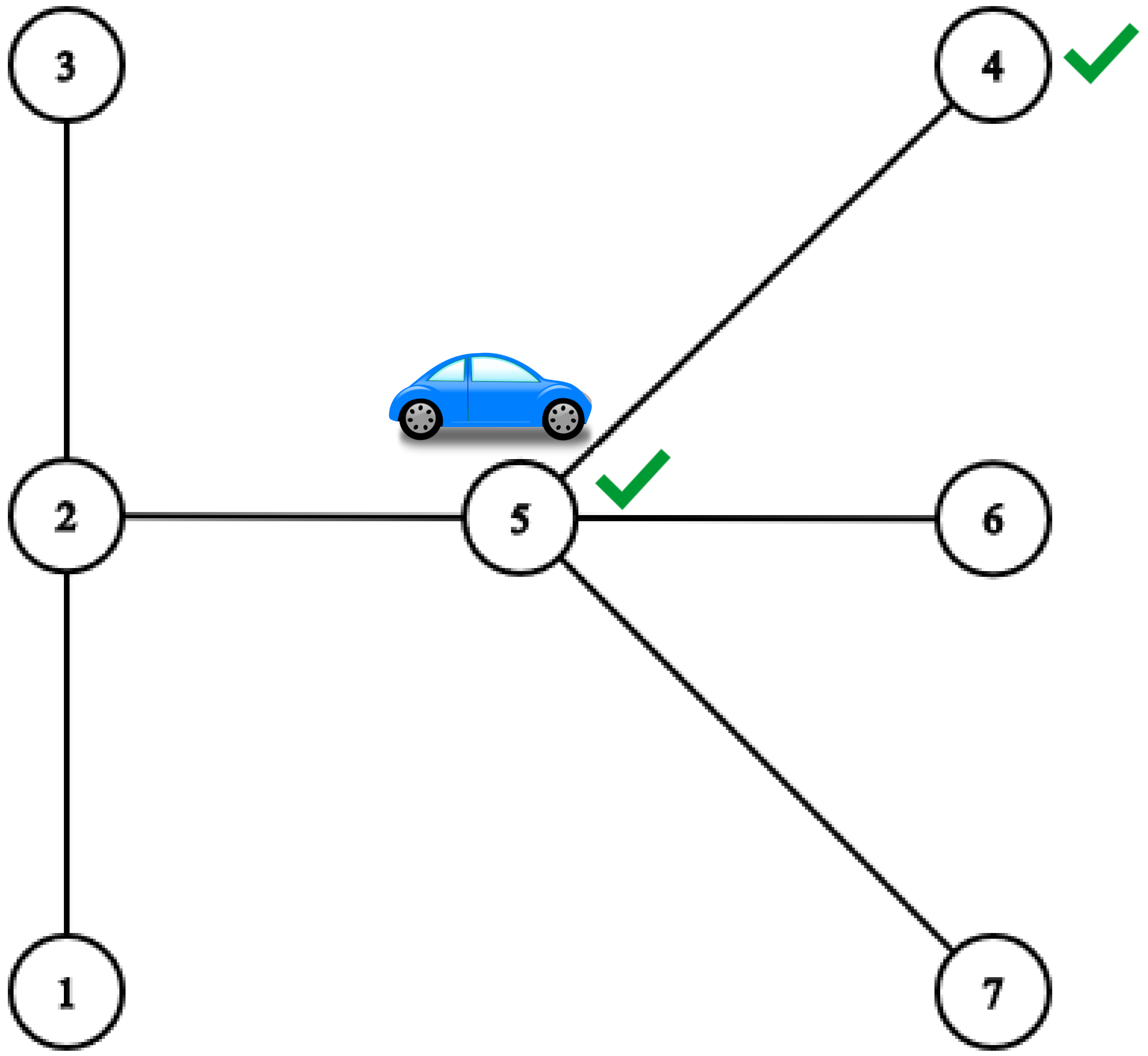
m=6



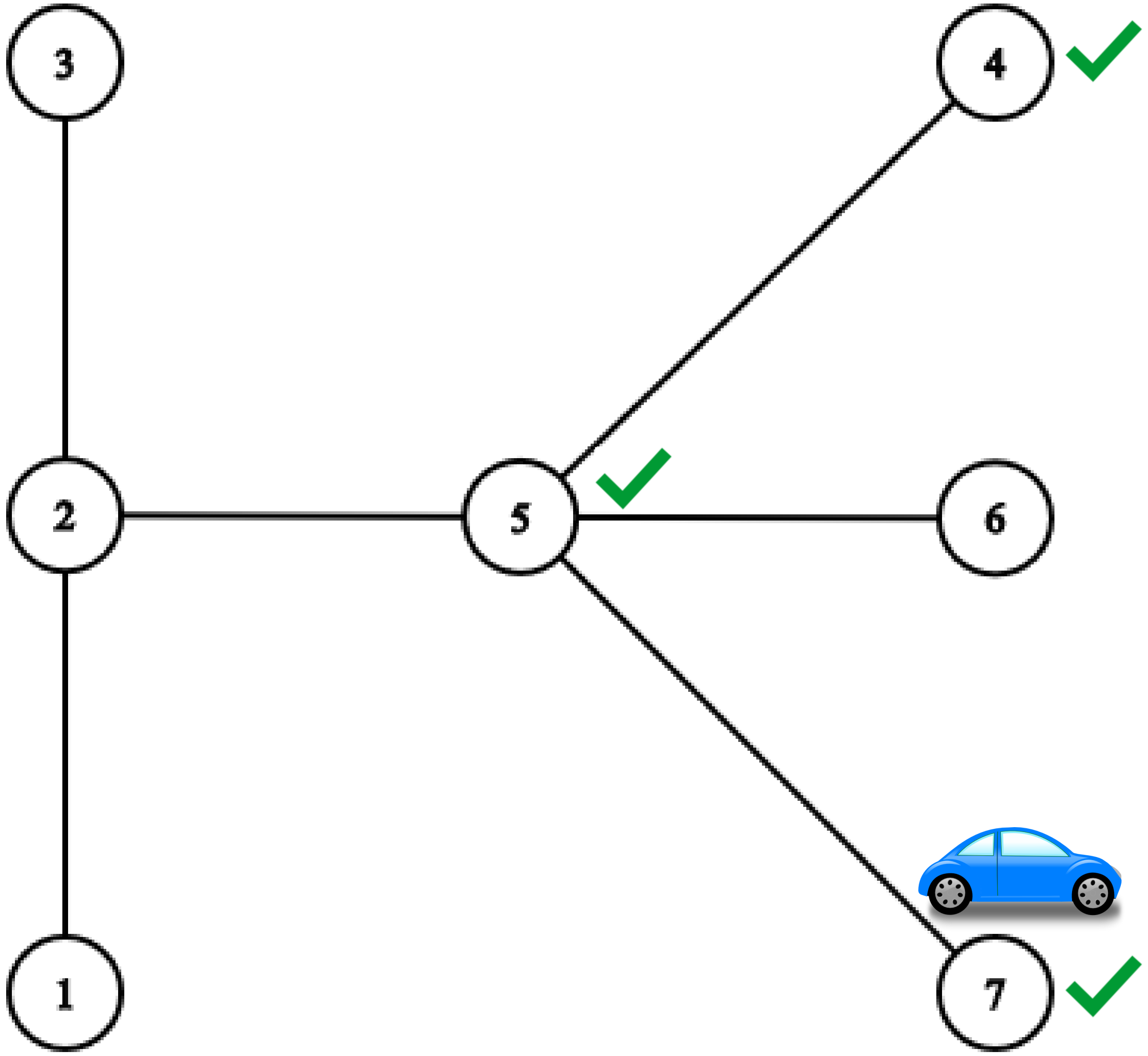
m=6



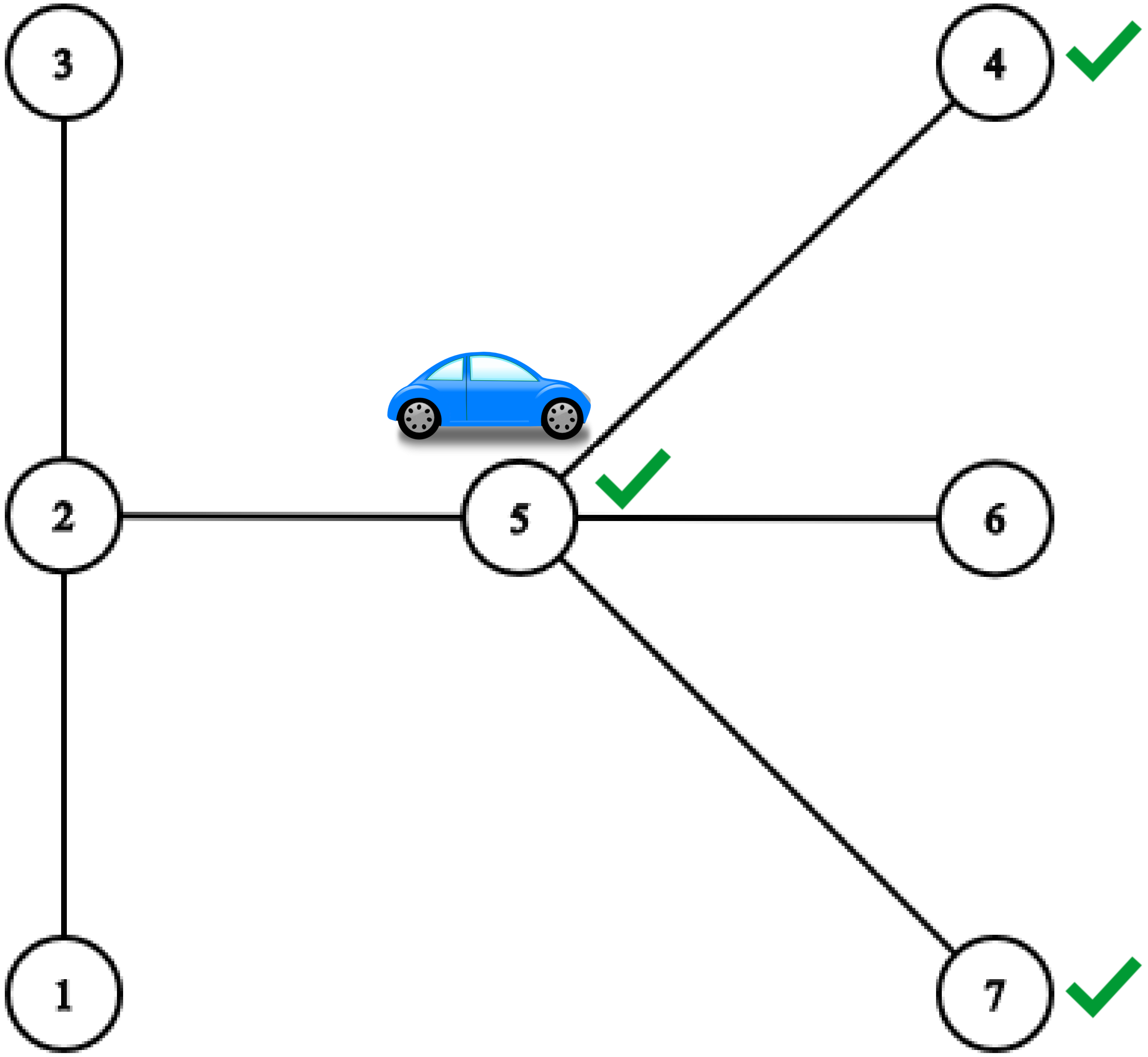
m=5



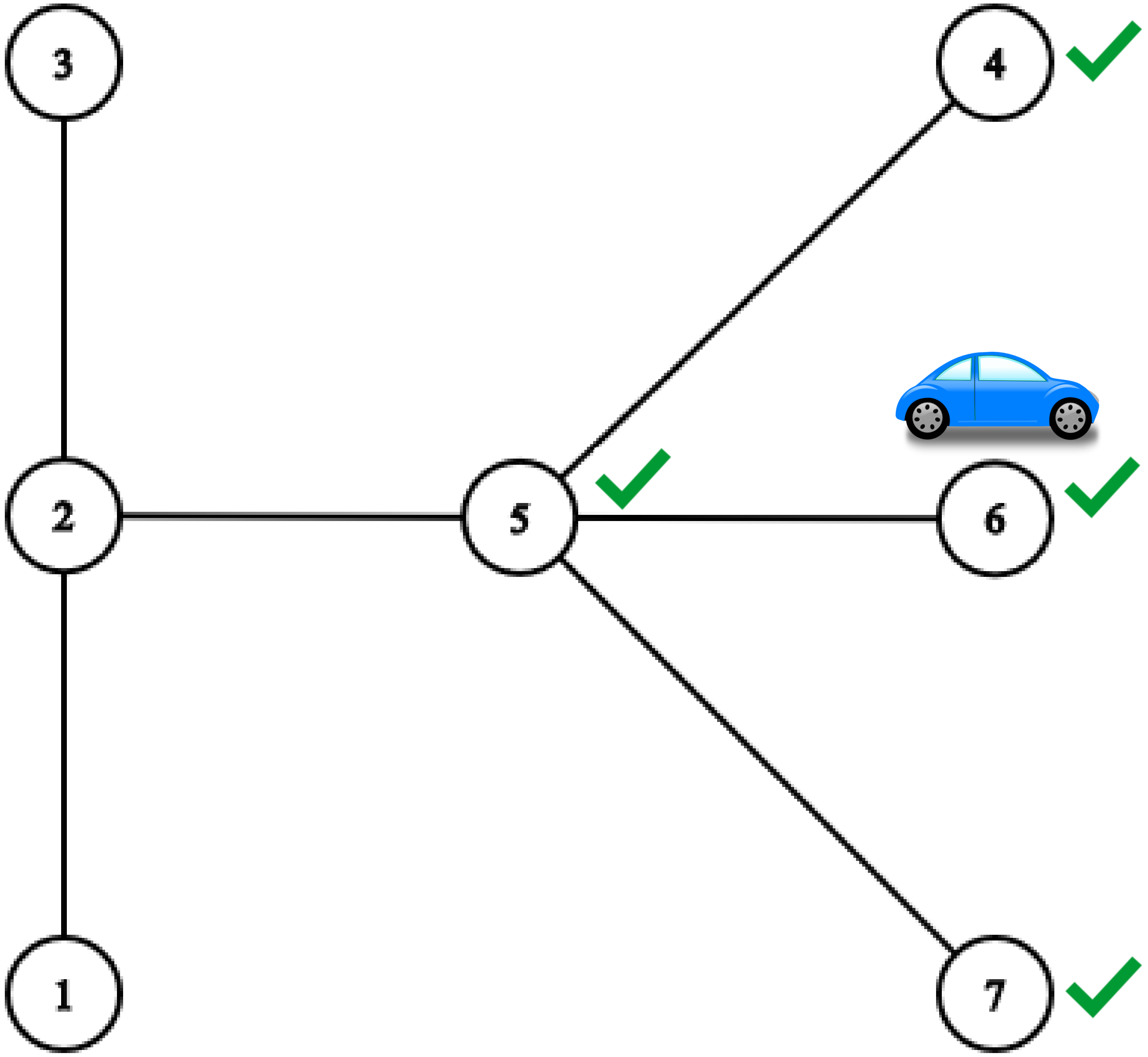
m=4



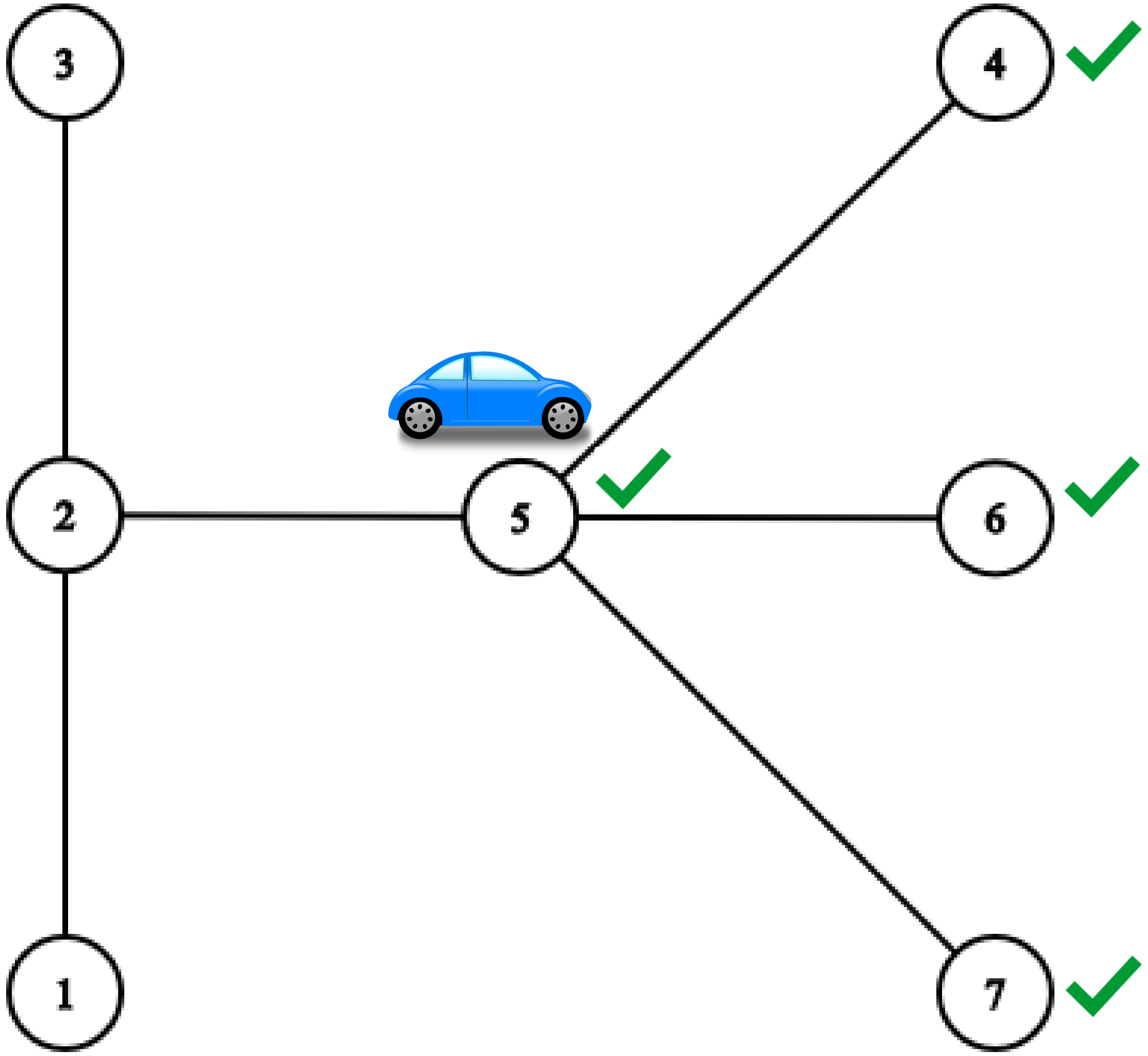
m=3



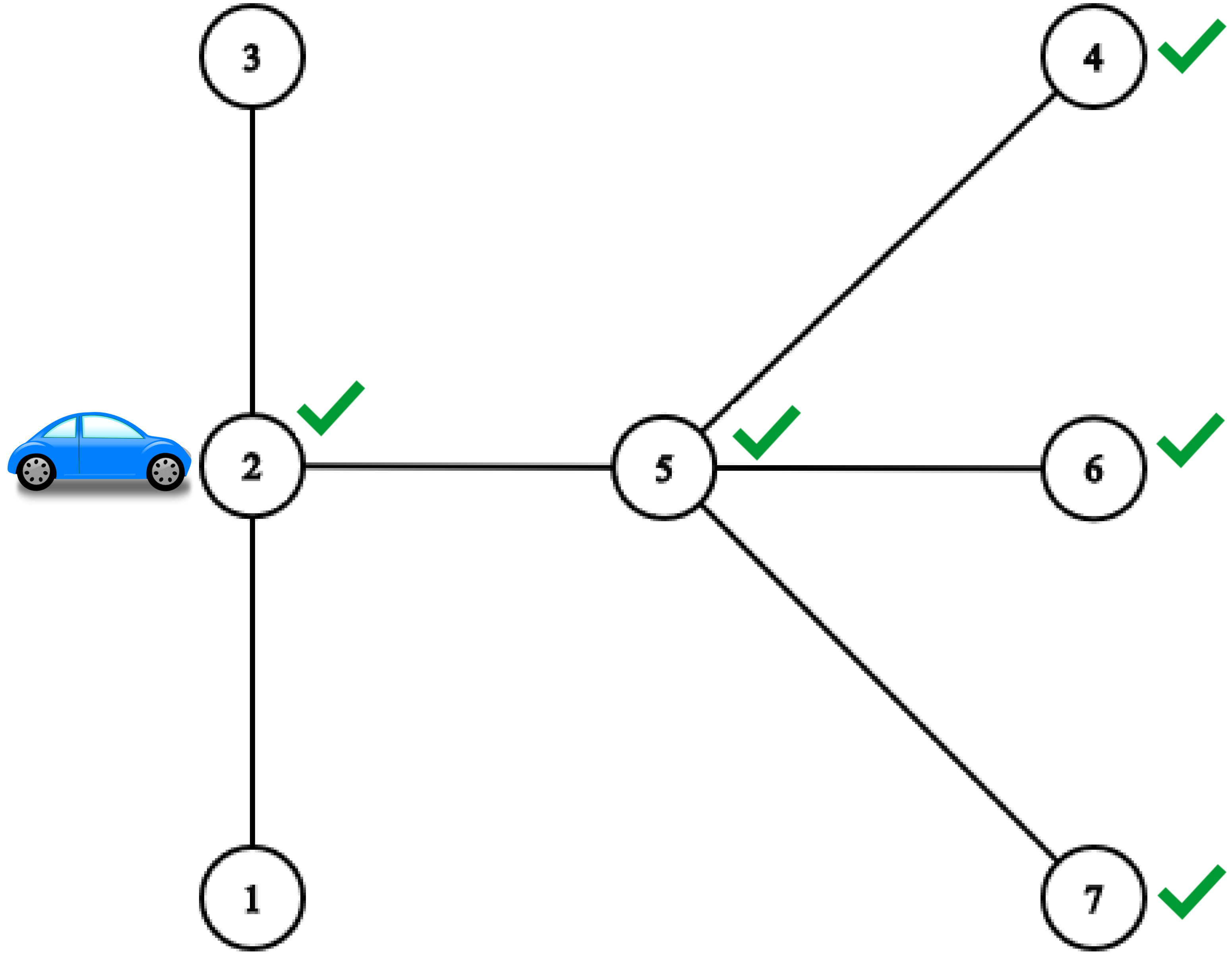
m=2



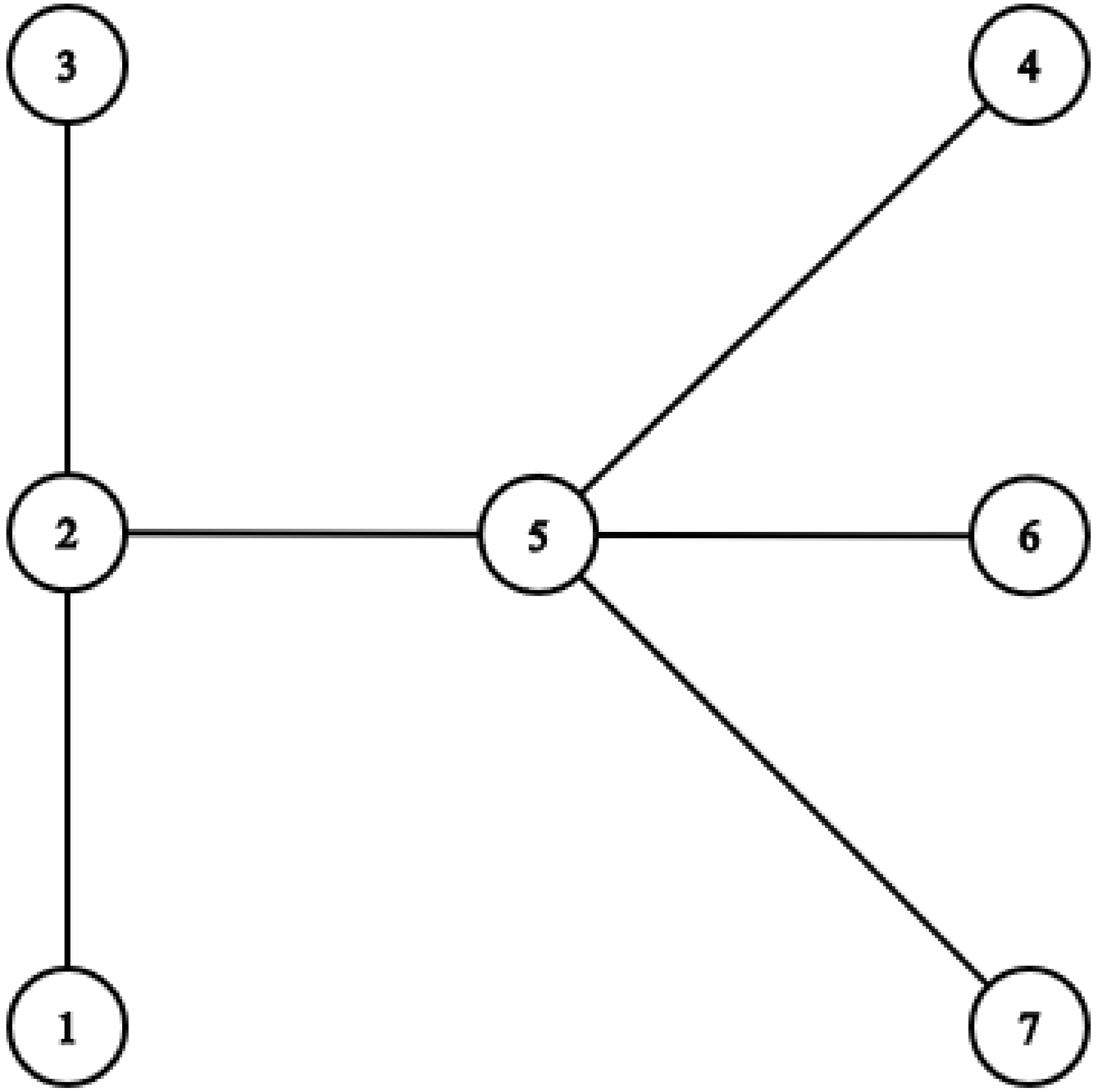
m=1



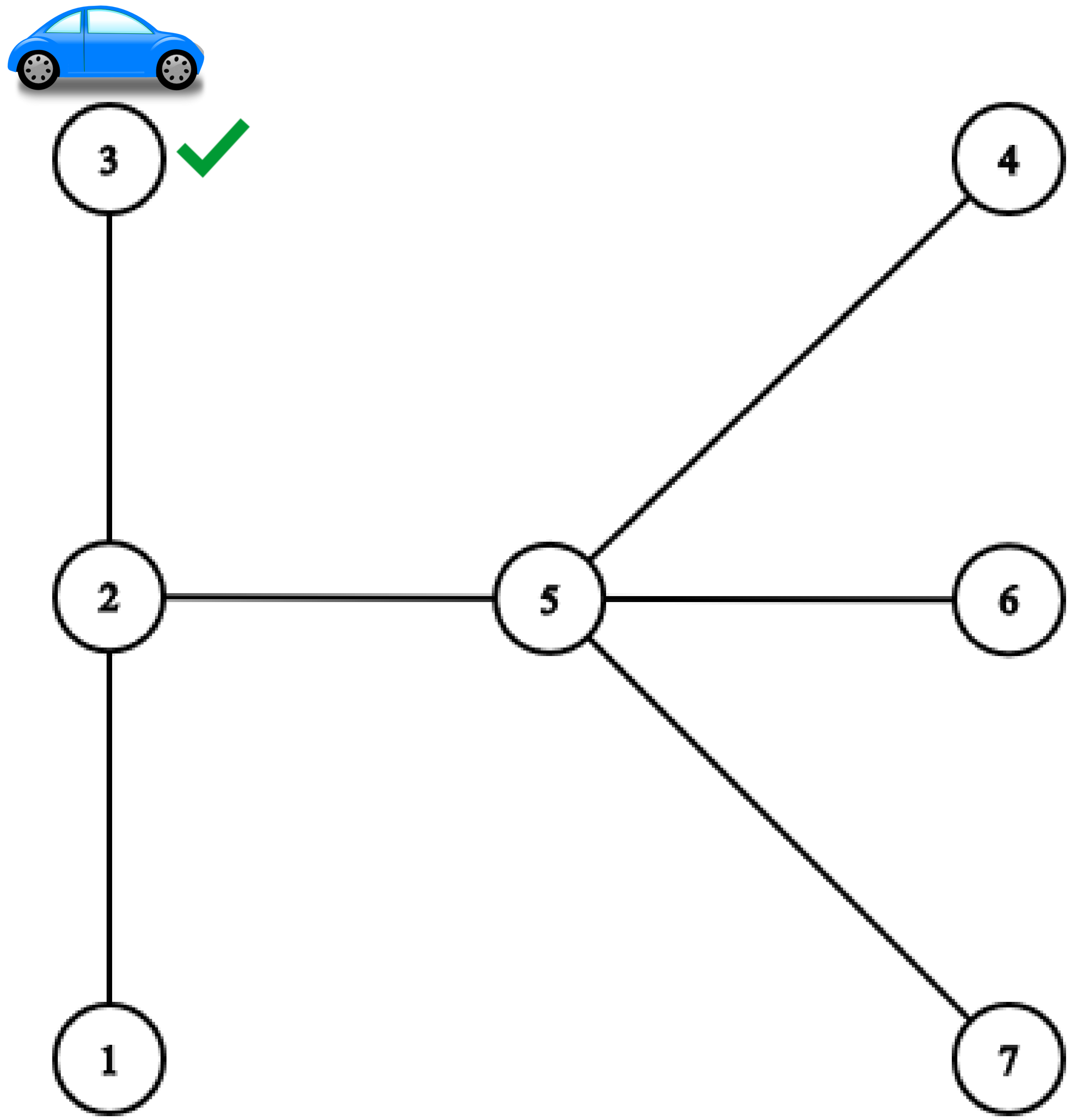
m=0



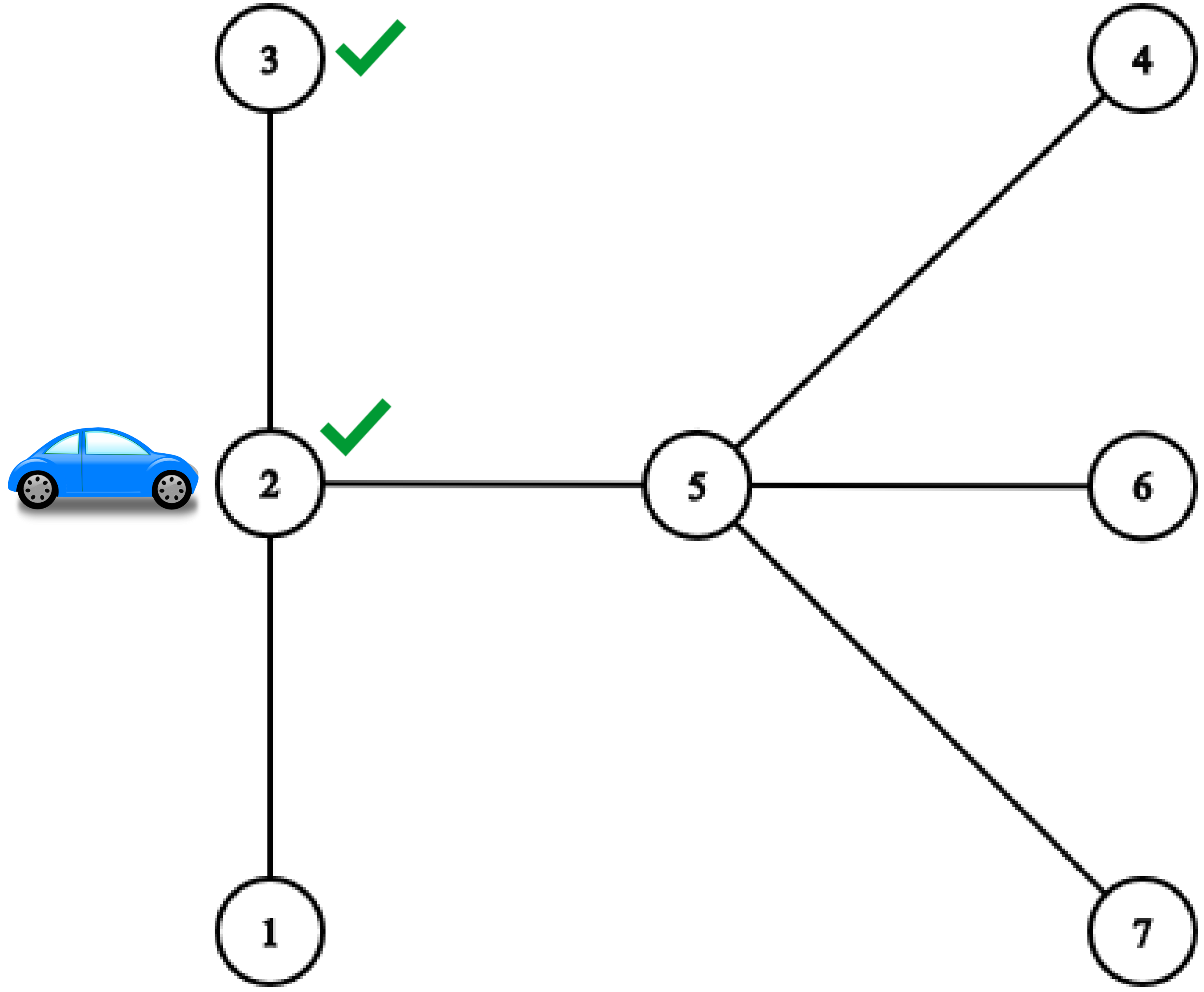
m=6



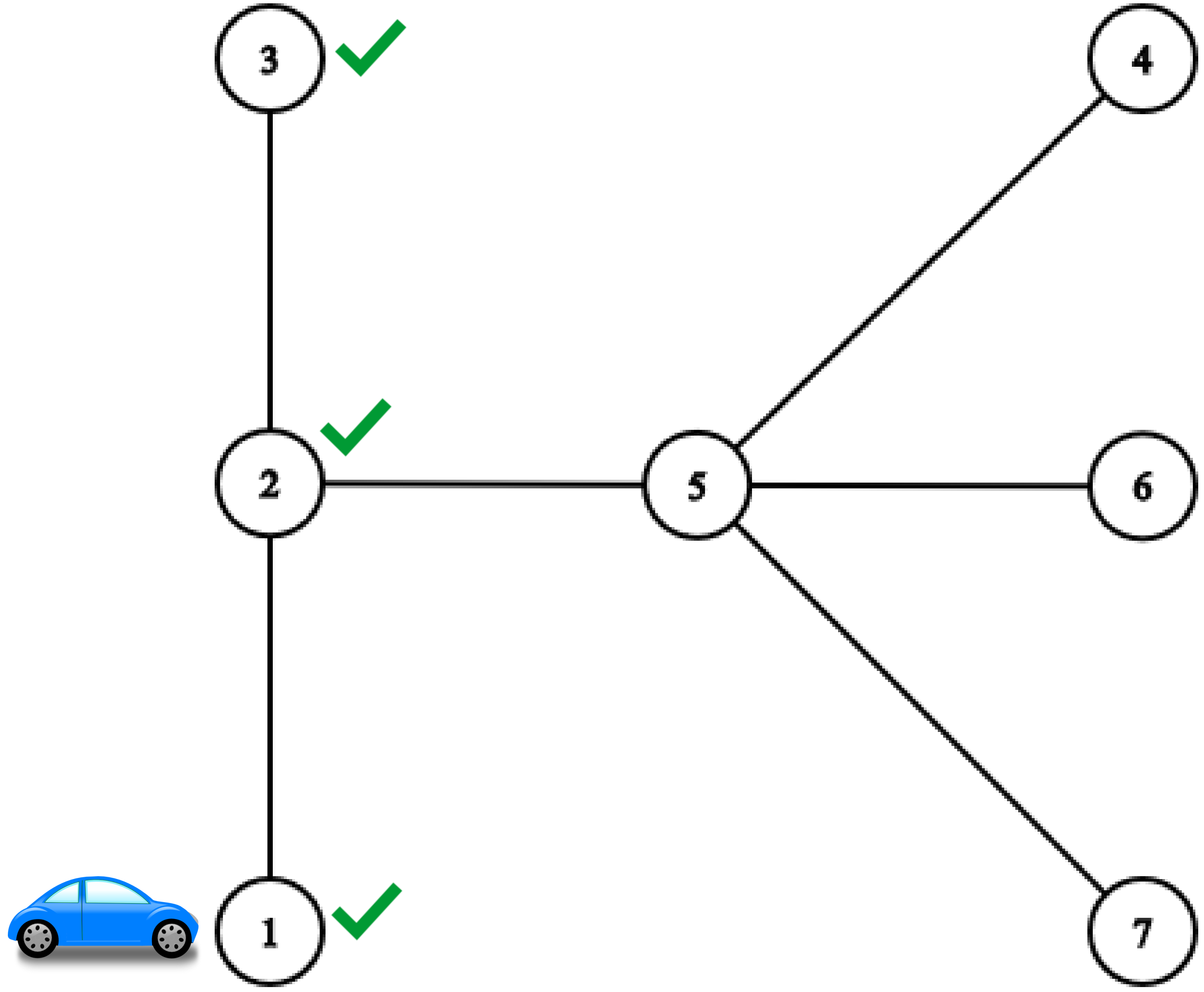
m=6



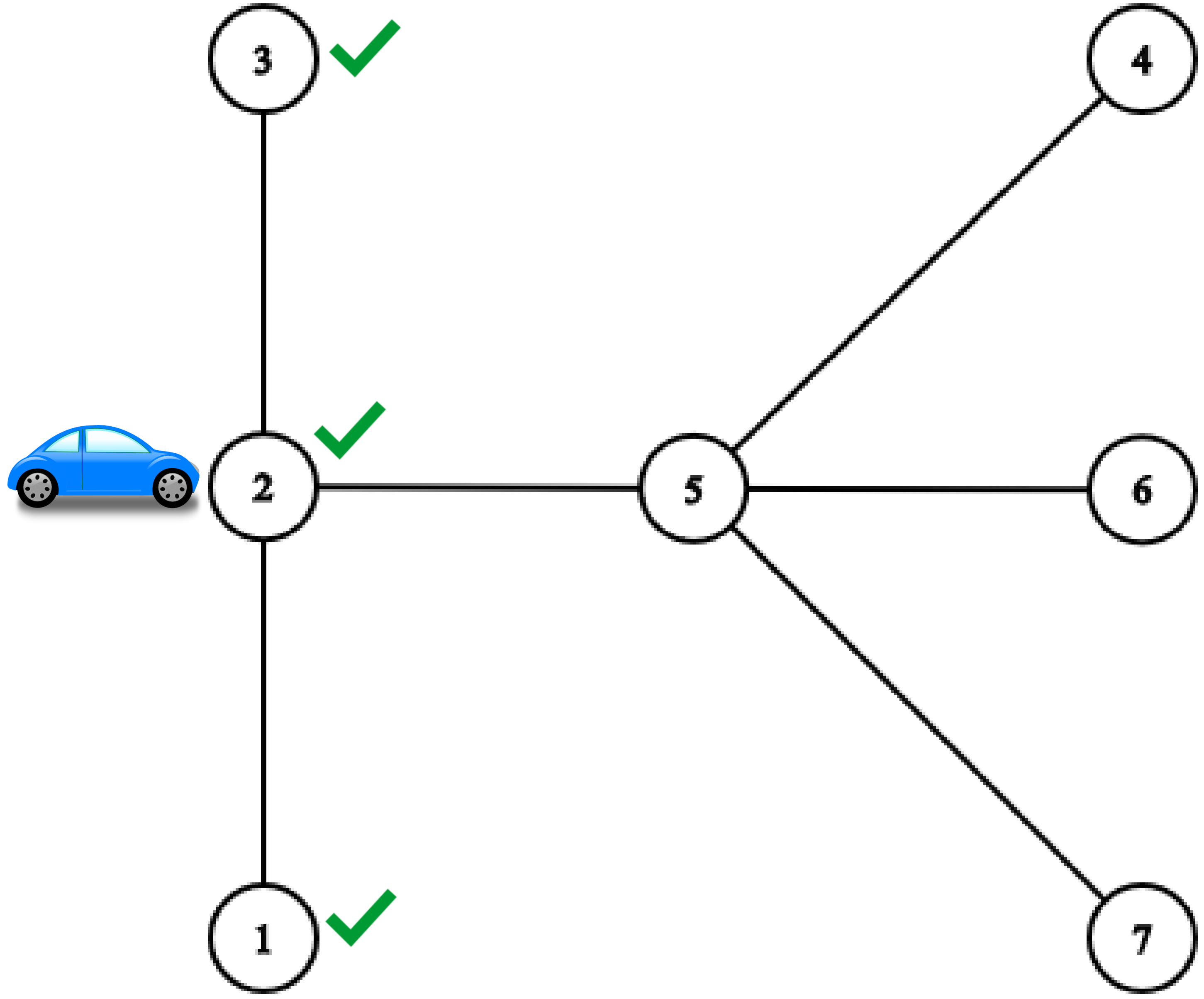
m=5



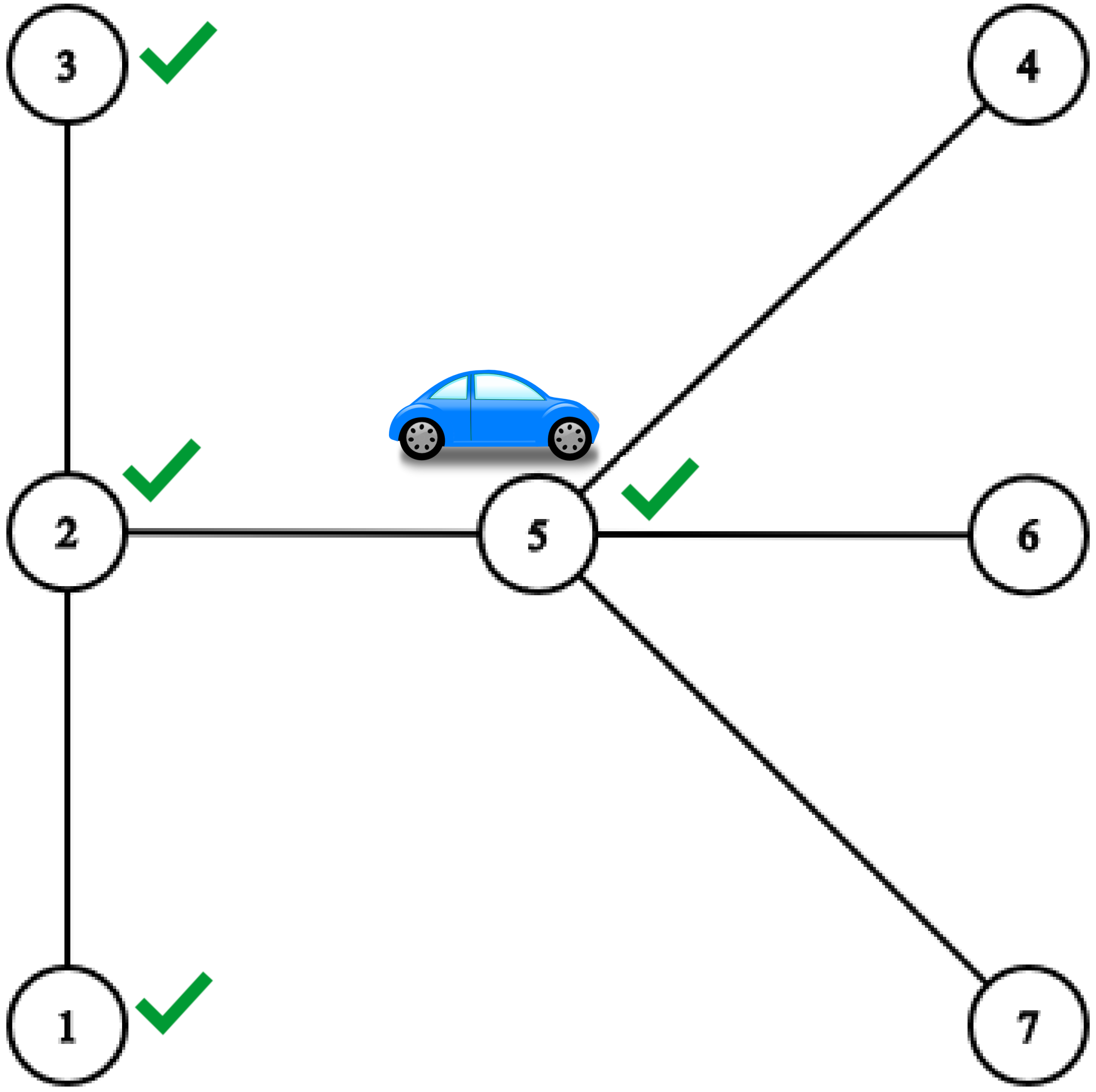
m=4



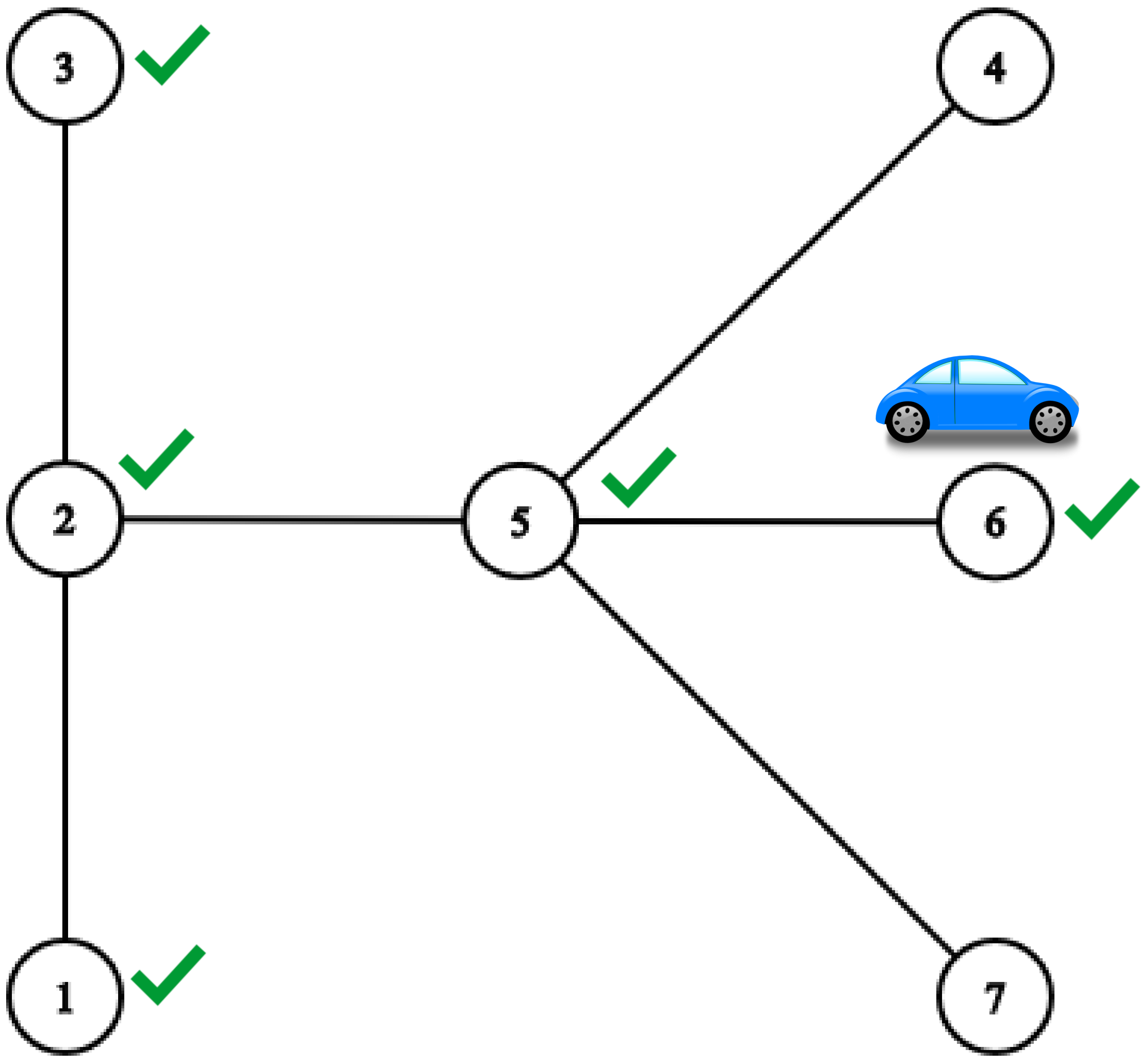
m=3



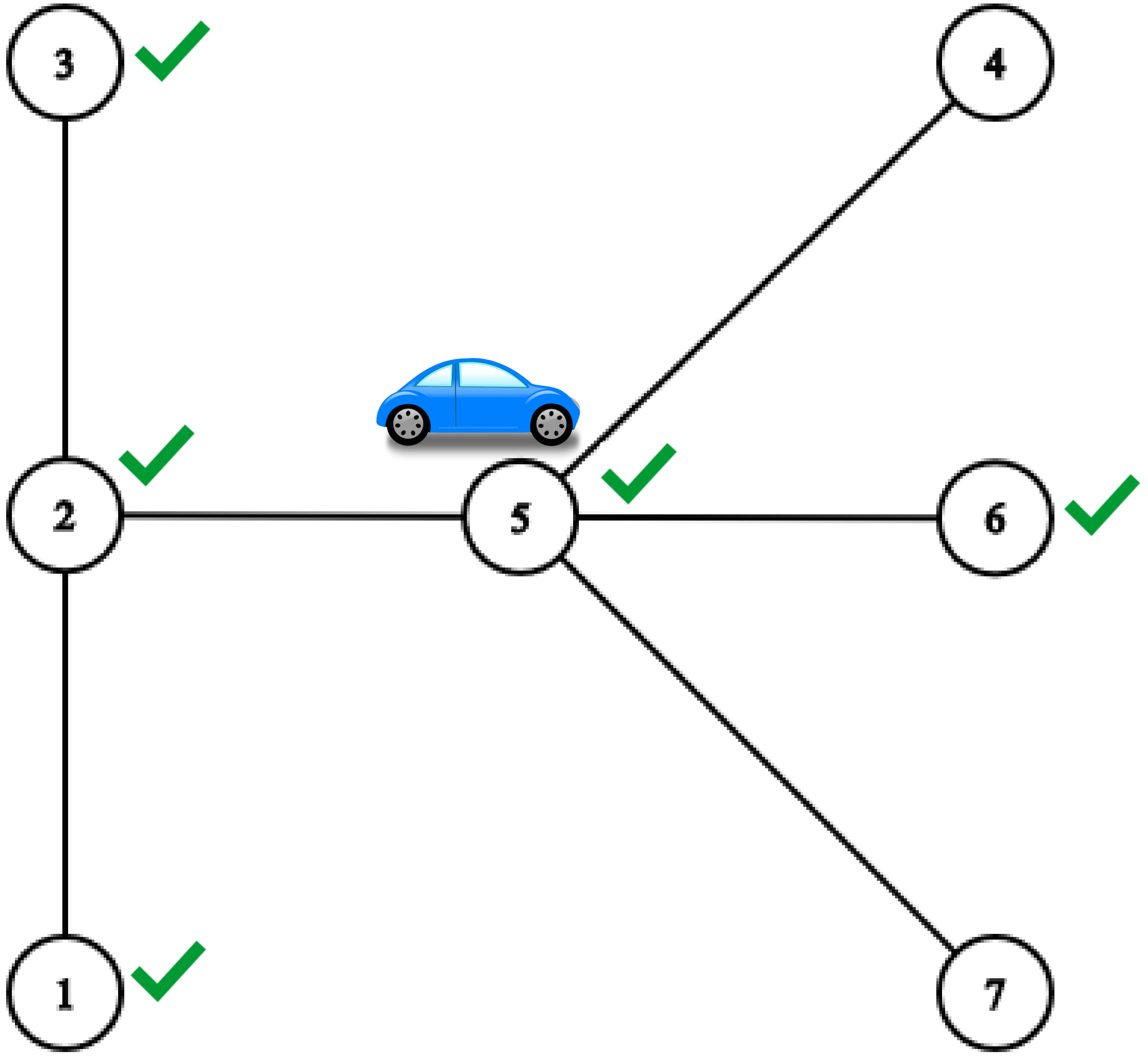
m=2



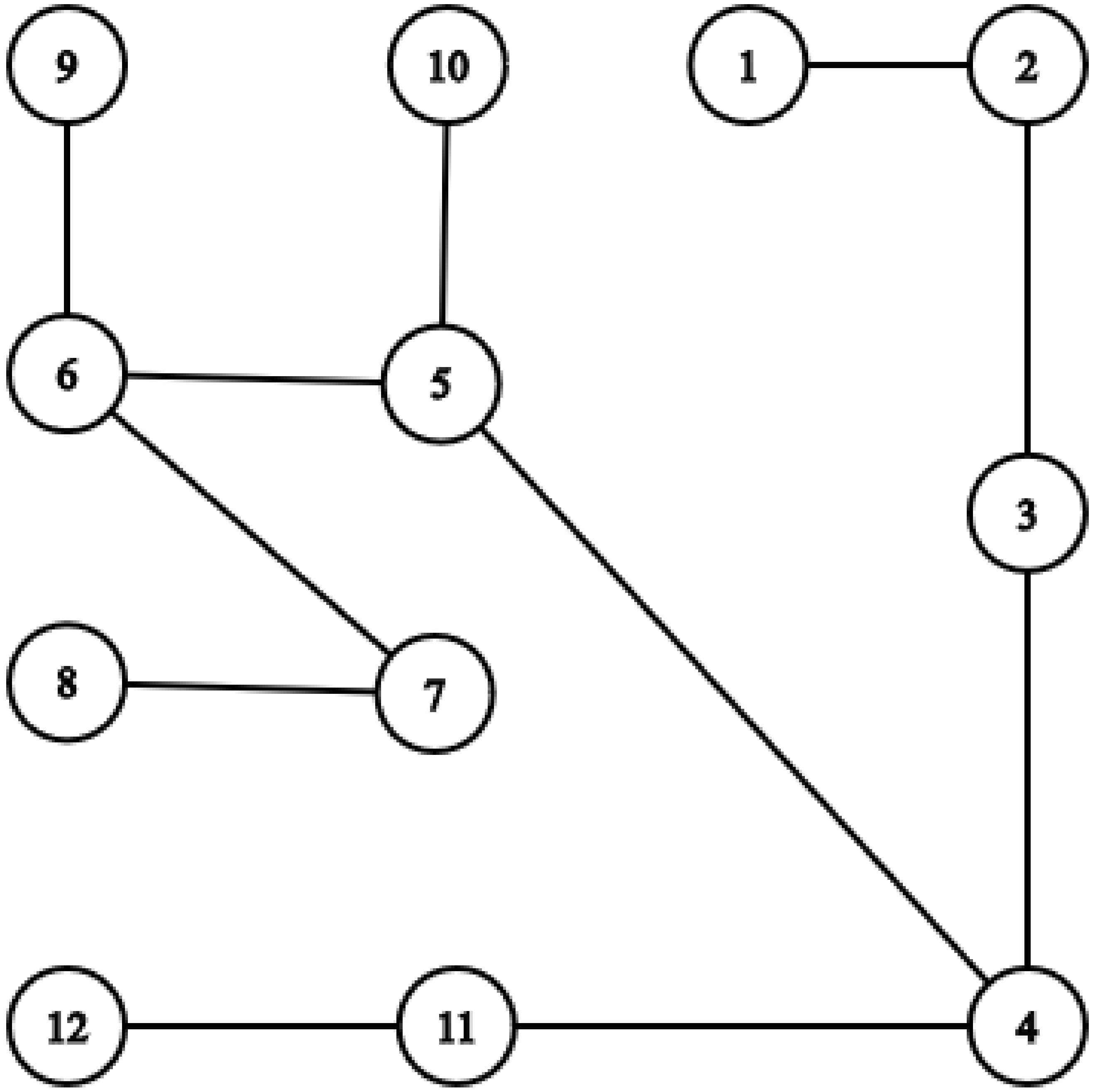
m=1



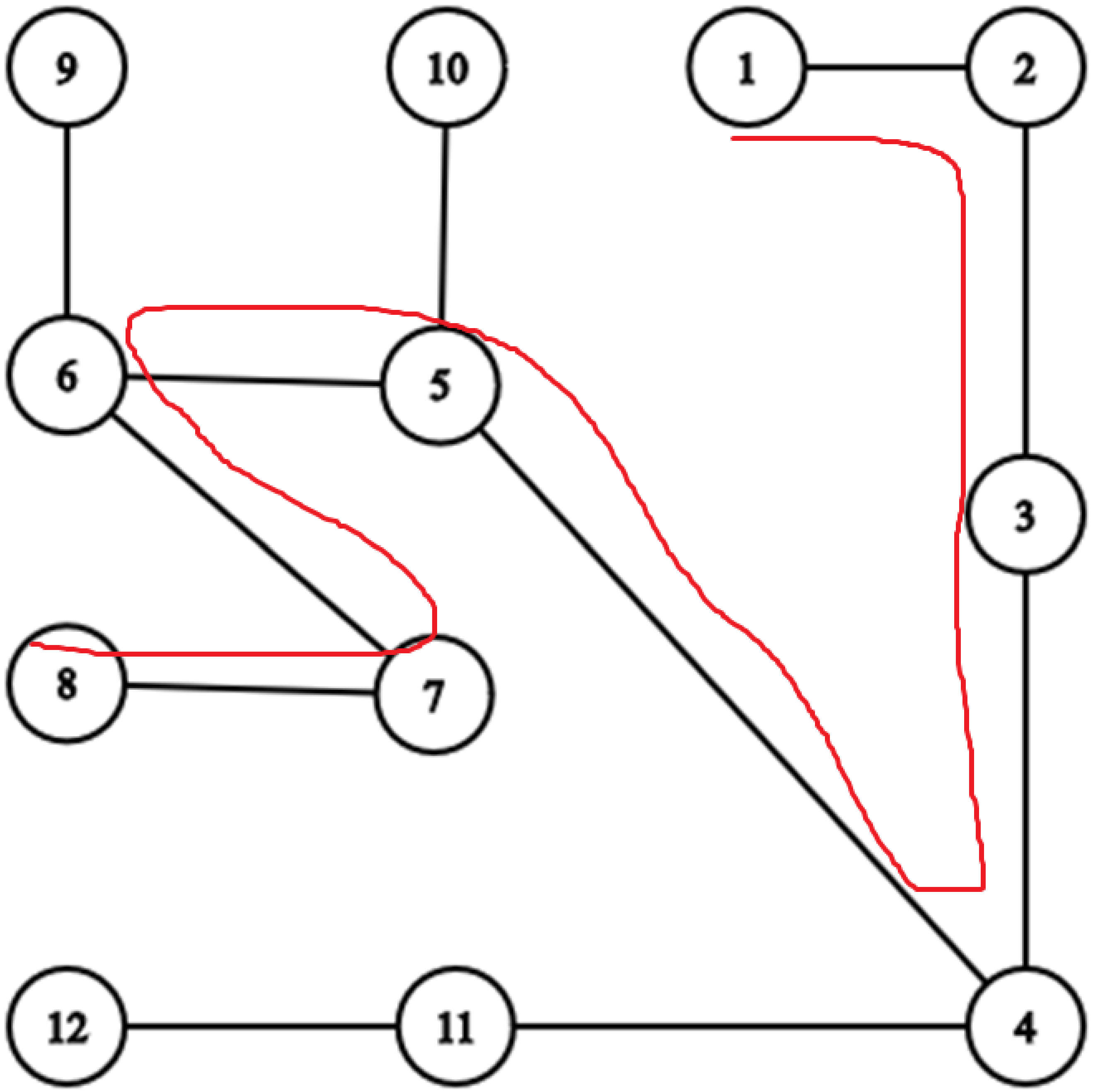
m=0



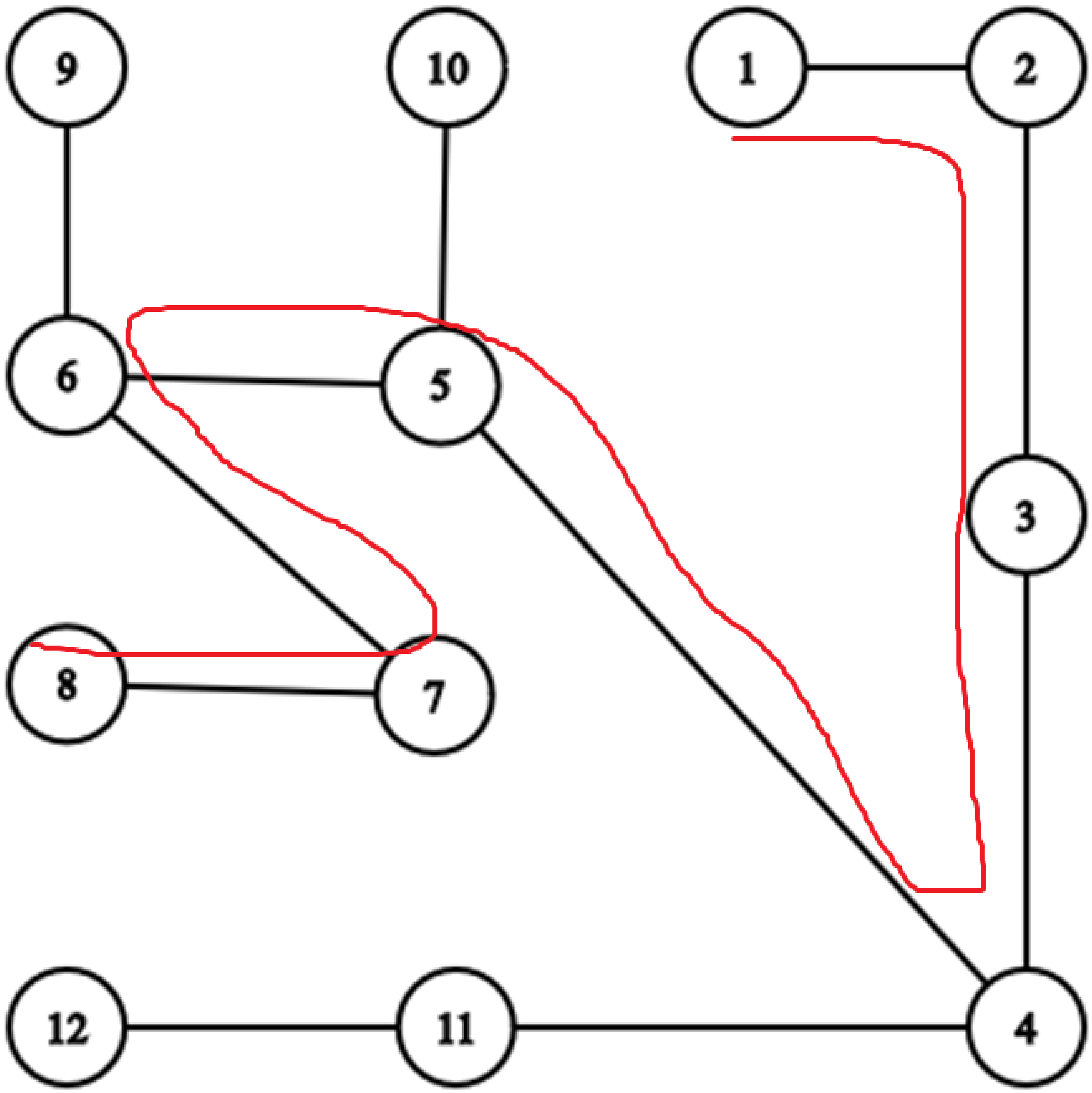
max length=?



max length=7

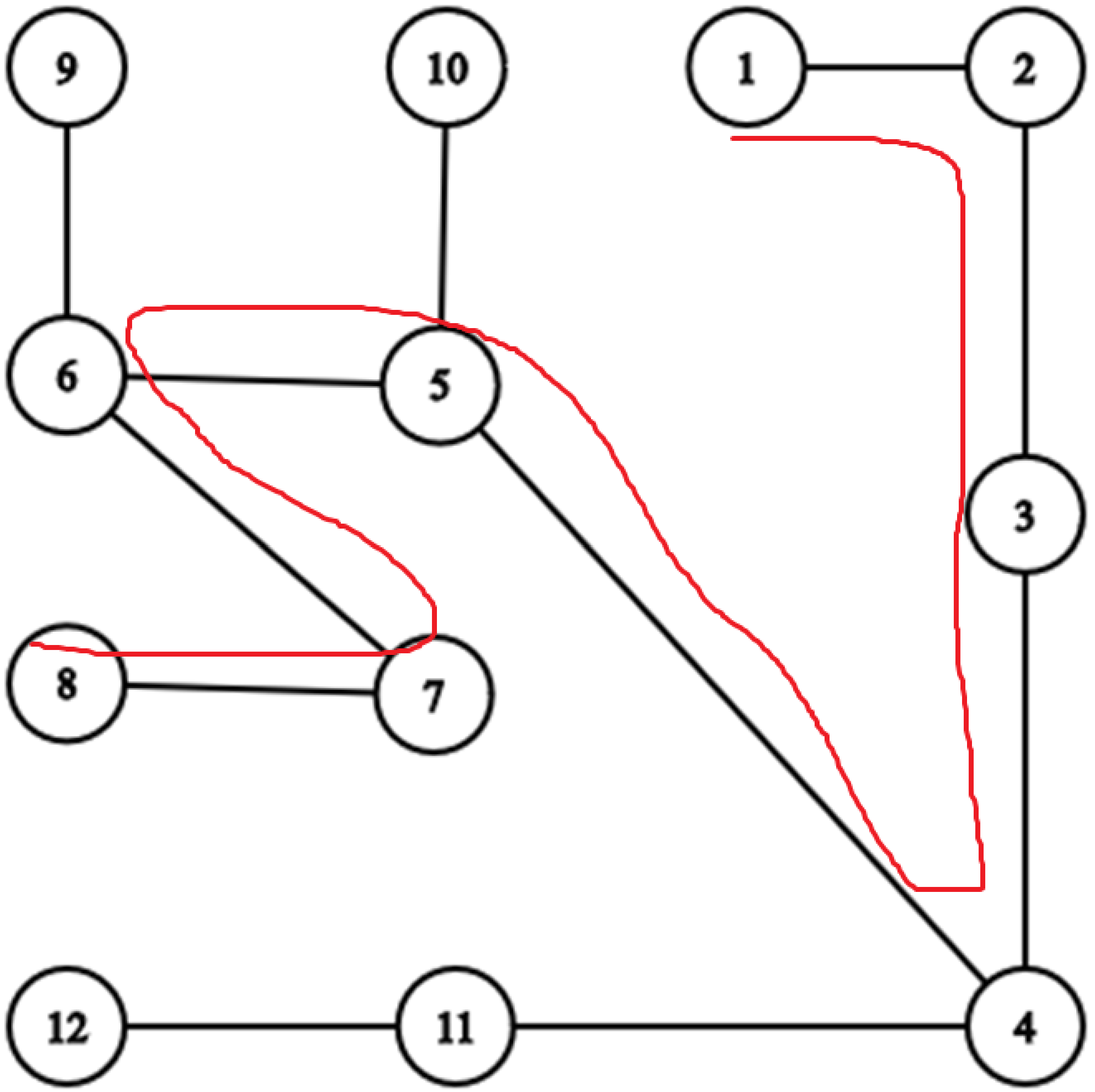


diameter=7



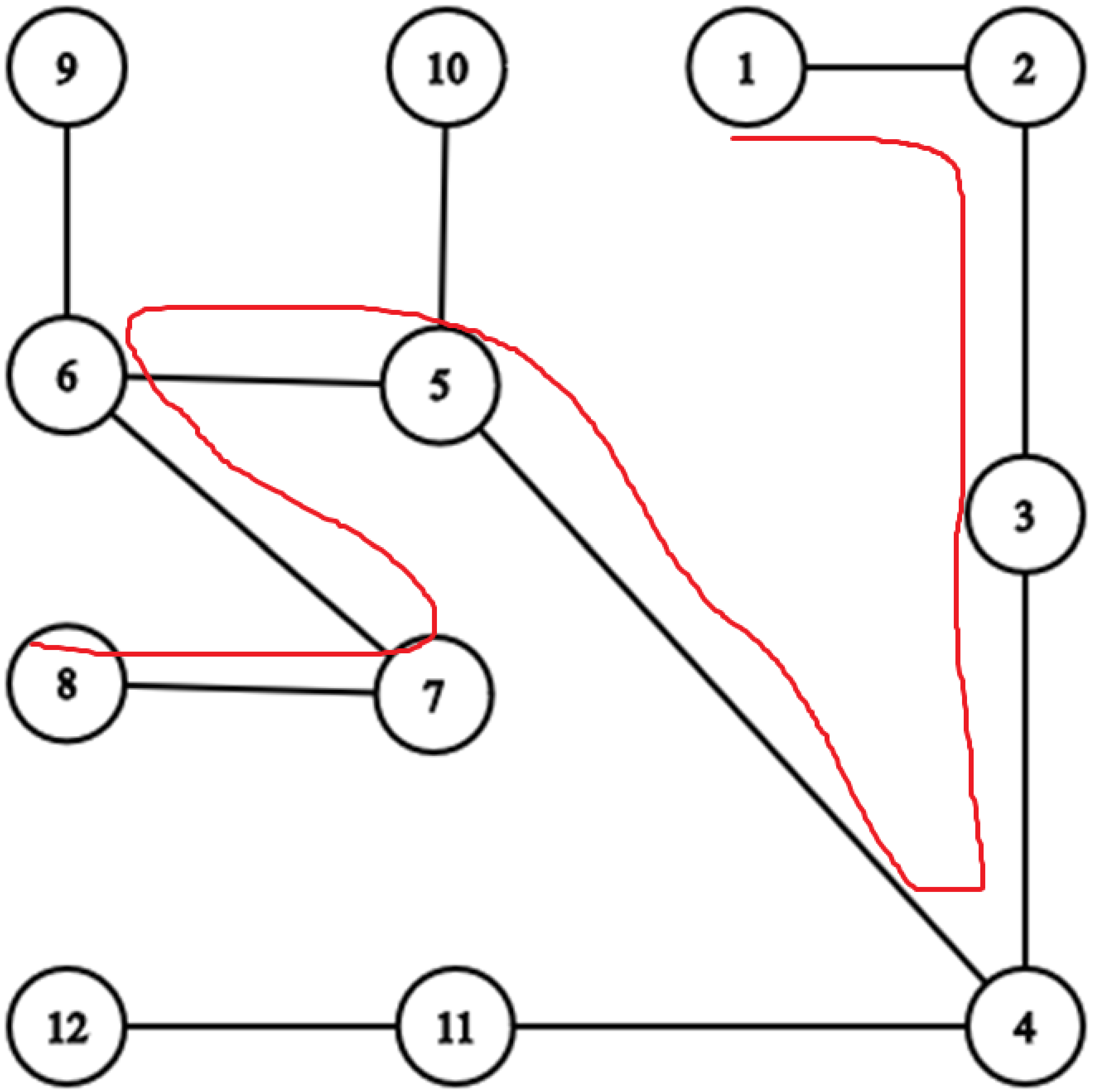
diameter=7

O(n)



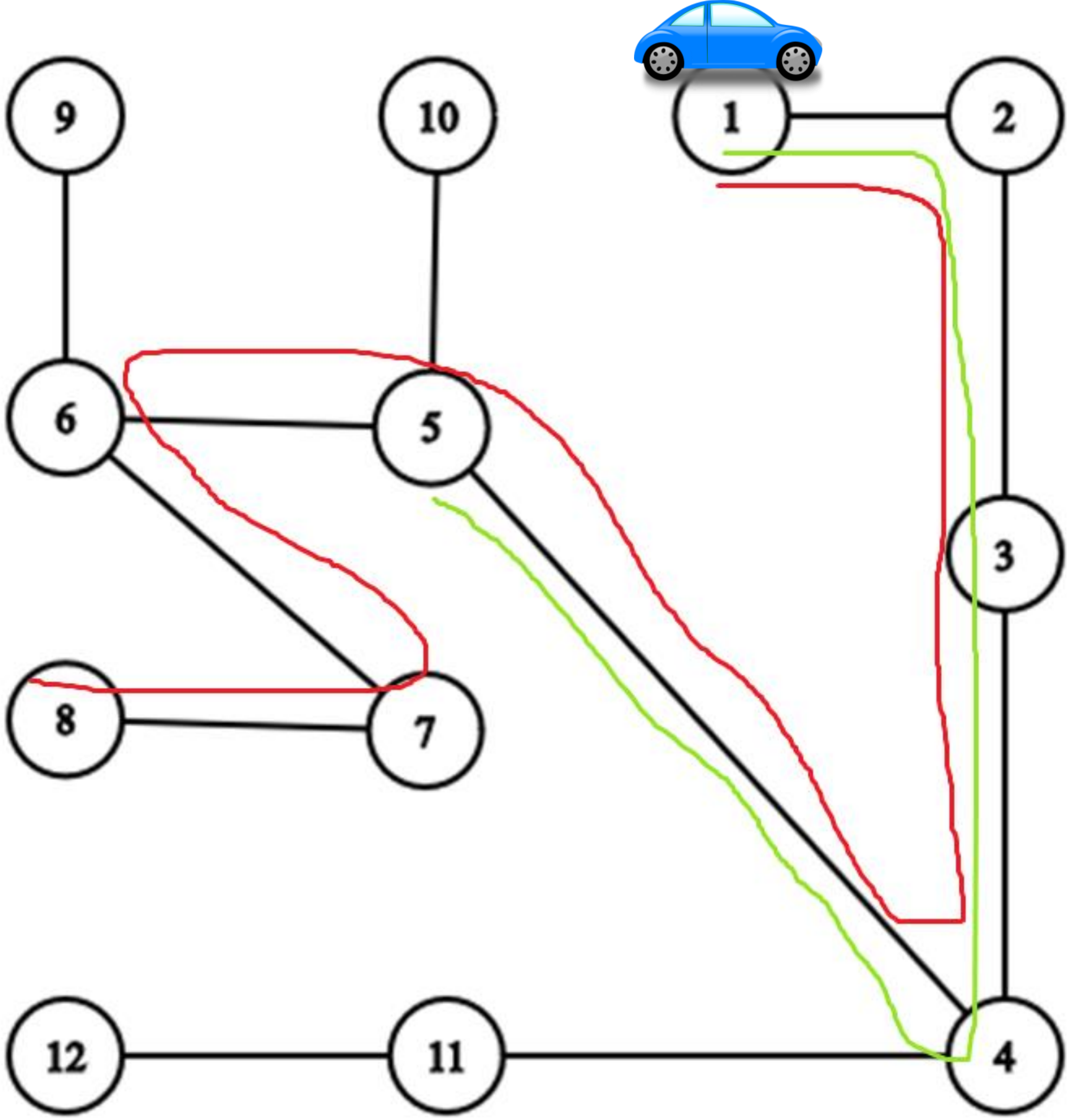
diameter=7

⛽ $m \leq d$



diameter=7

m=4



$m \leq d$ Answer: $m+1$ 😃

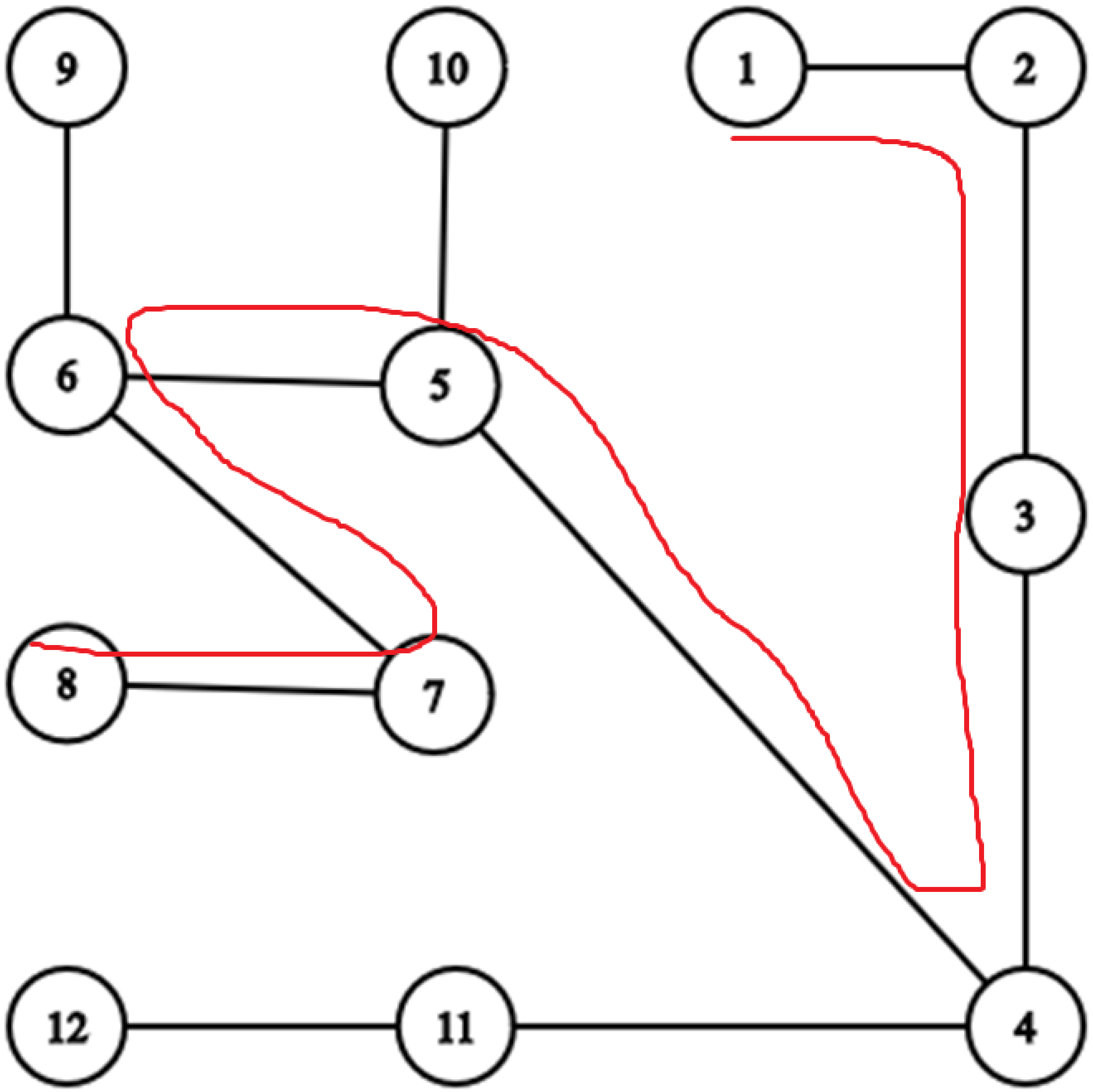
$m \leq d$ Answer: $m+1$ 😃

$m > d$ Answer: ?

$m \leq d$	Answer: $m+1$ 😃
$m > d$	Answer: ? 😢😭😩

diameter=7

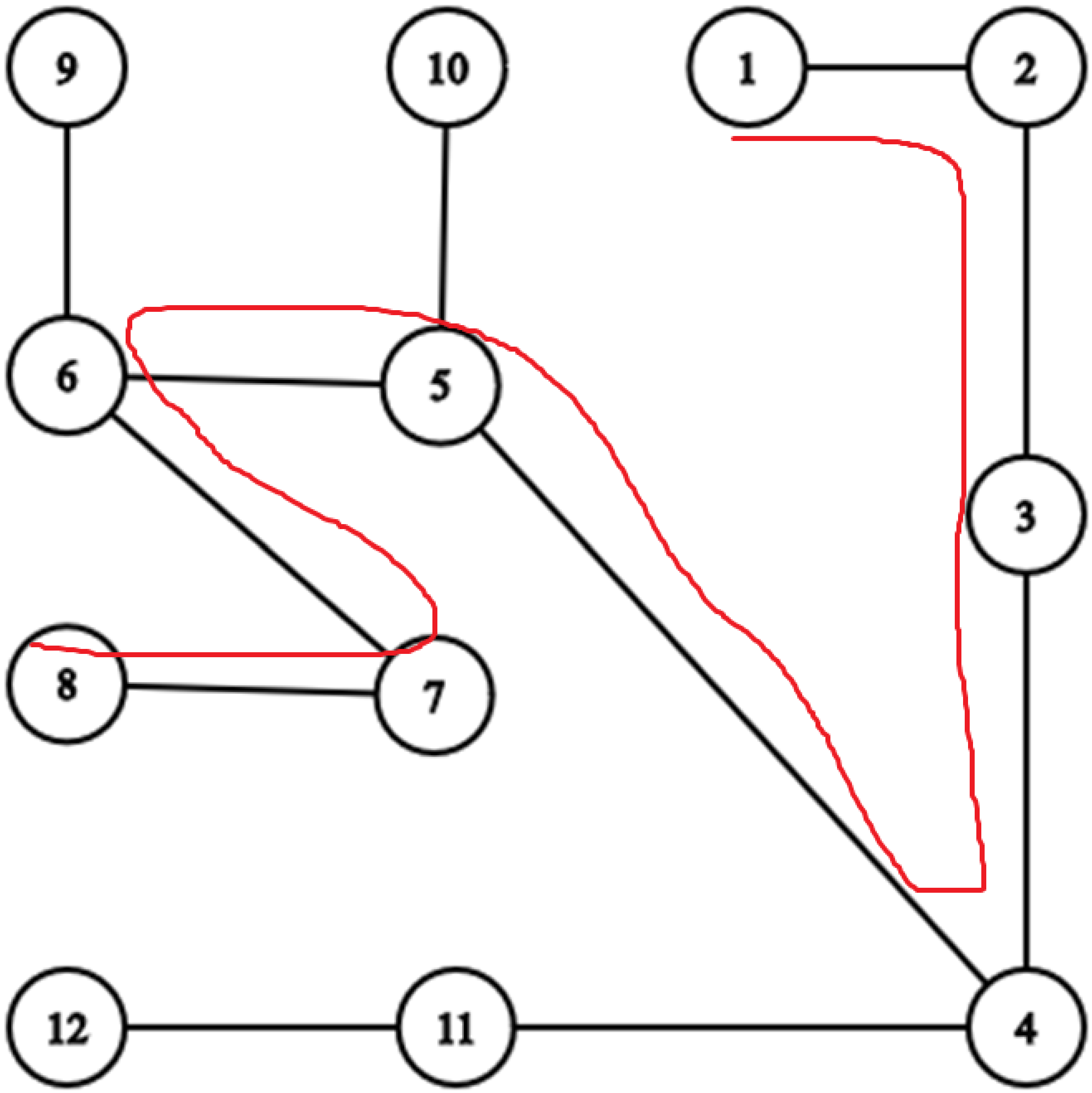
m=11



diameter=7

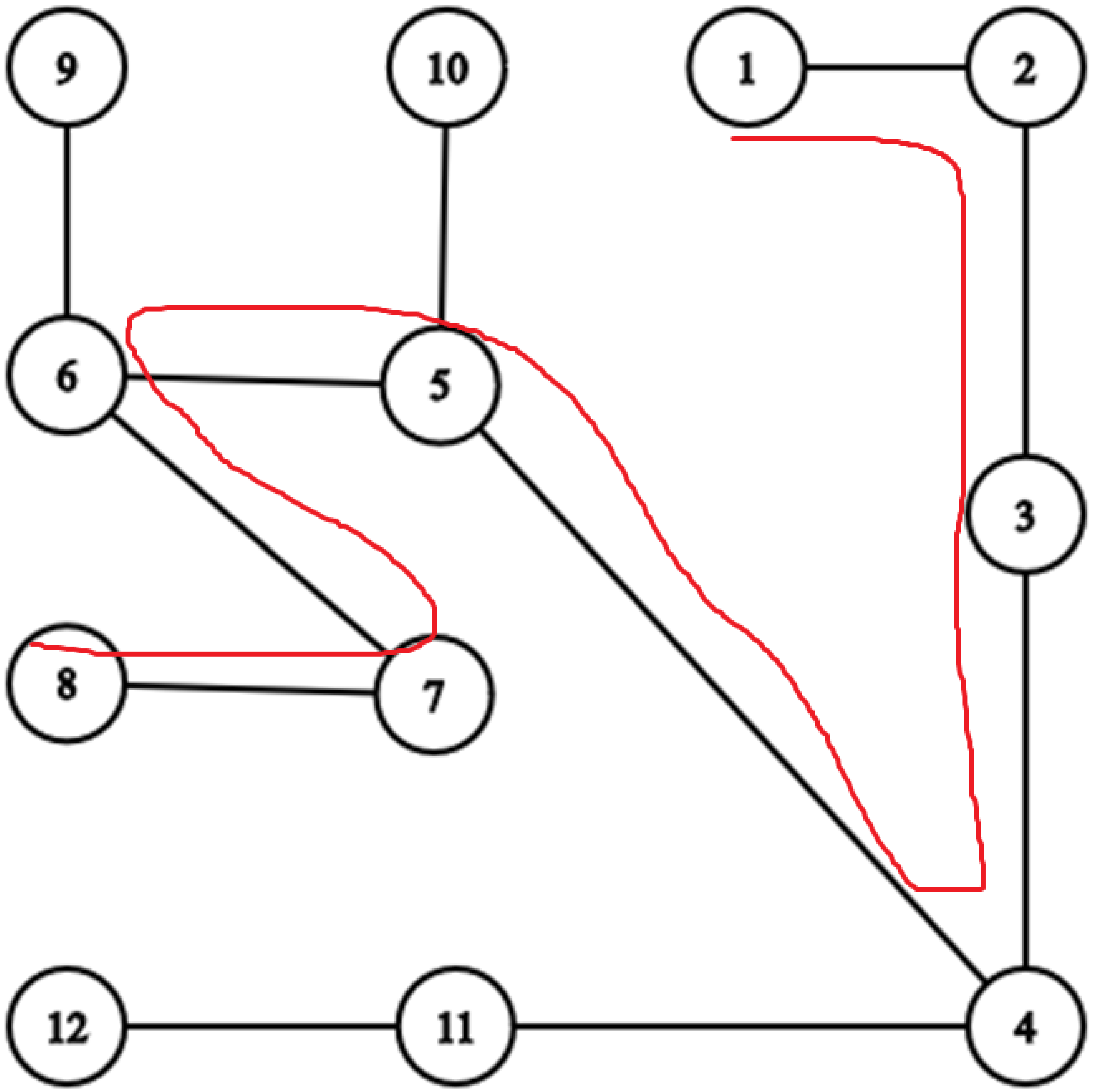
m=11

Answer ≥ 8



diameter=7

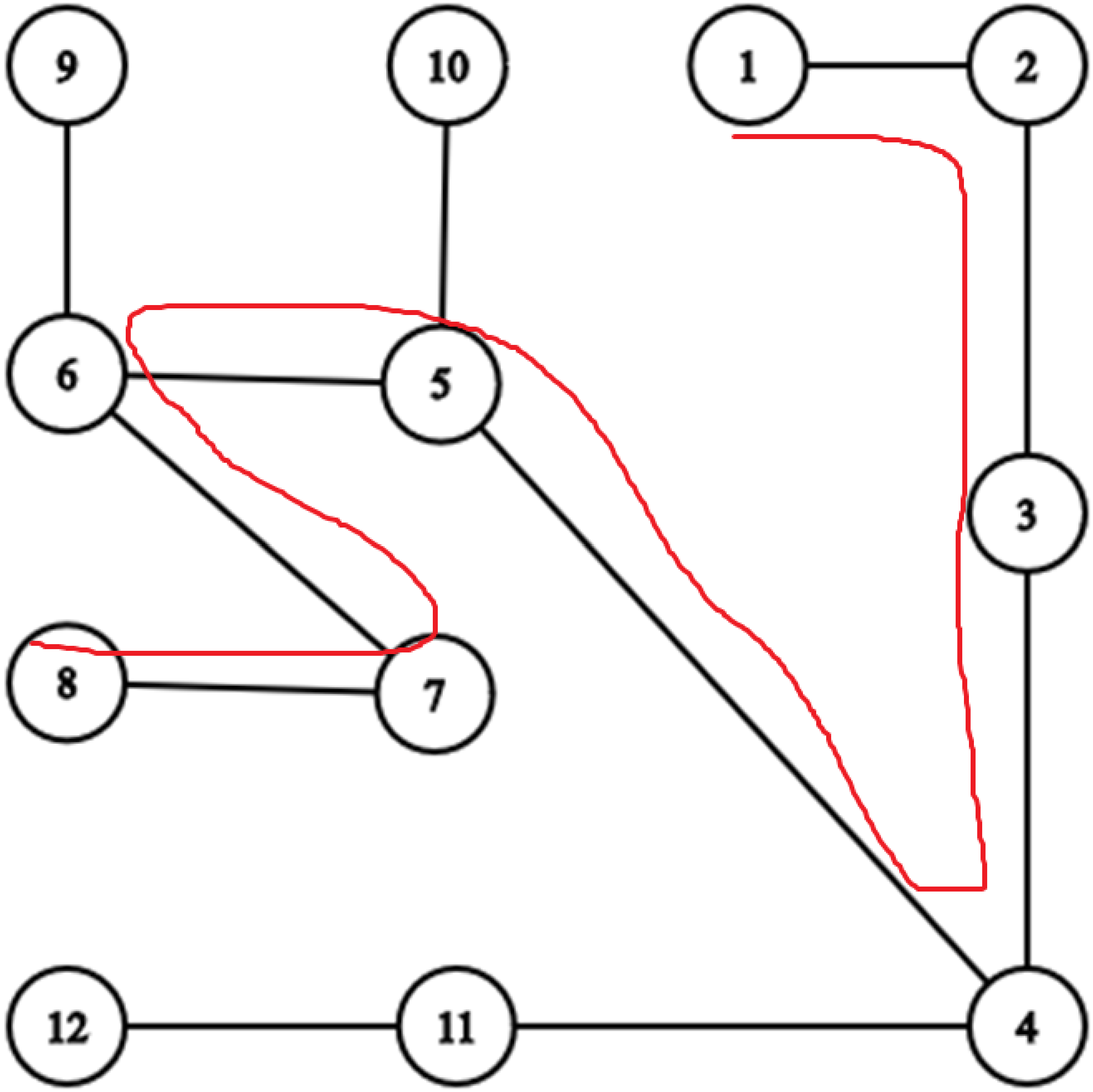
m=11



diameter=7

m=11

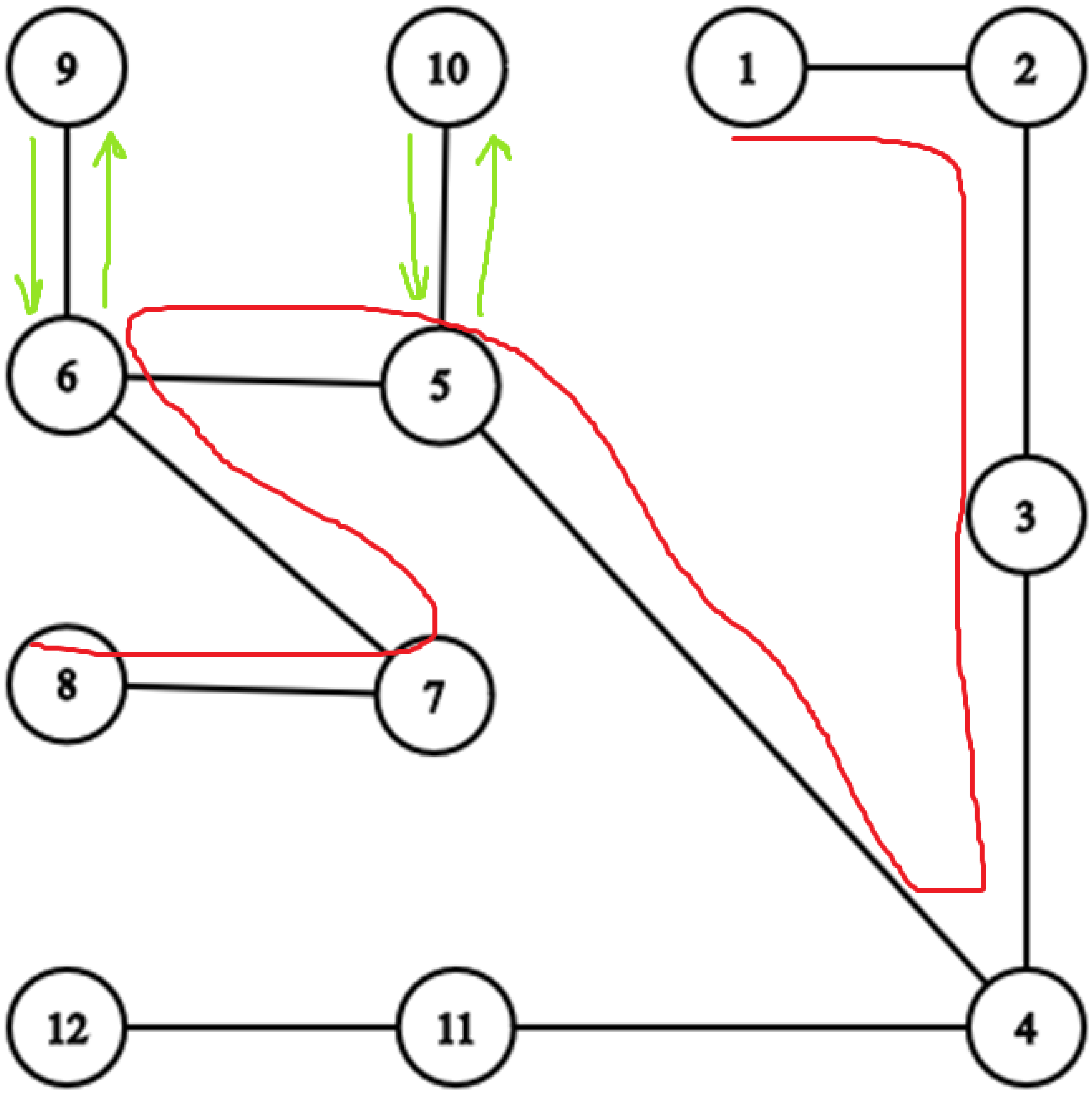
m − d = 4



diameter=7

m=11

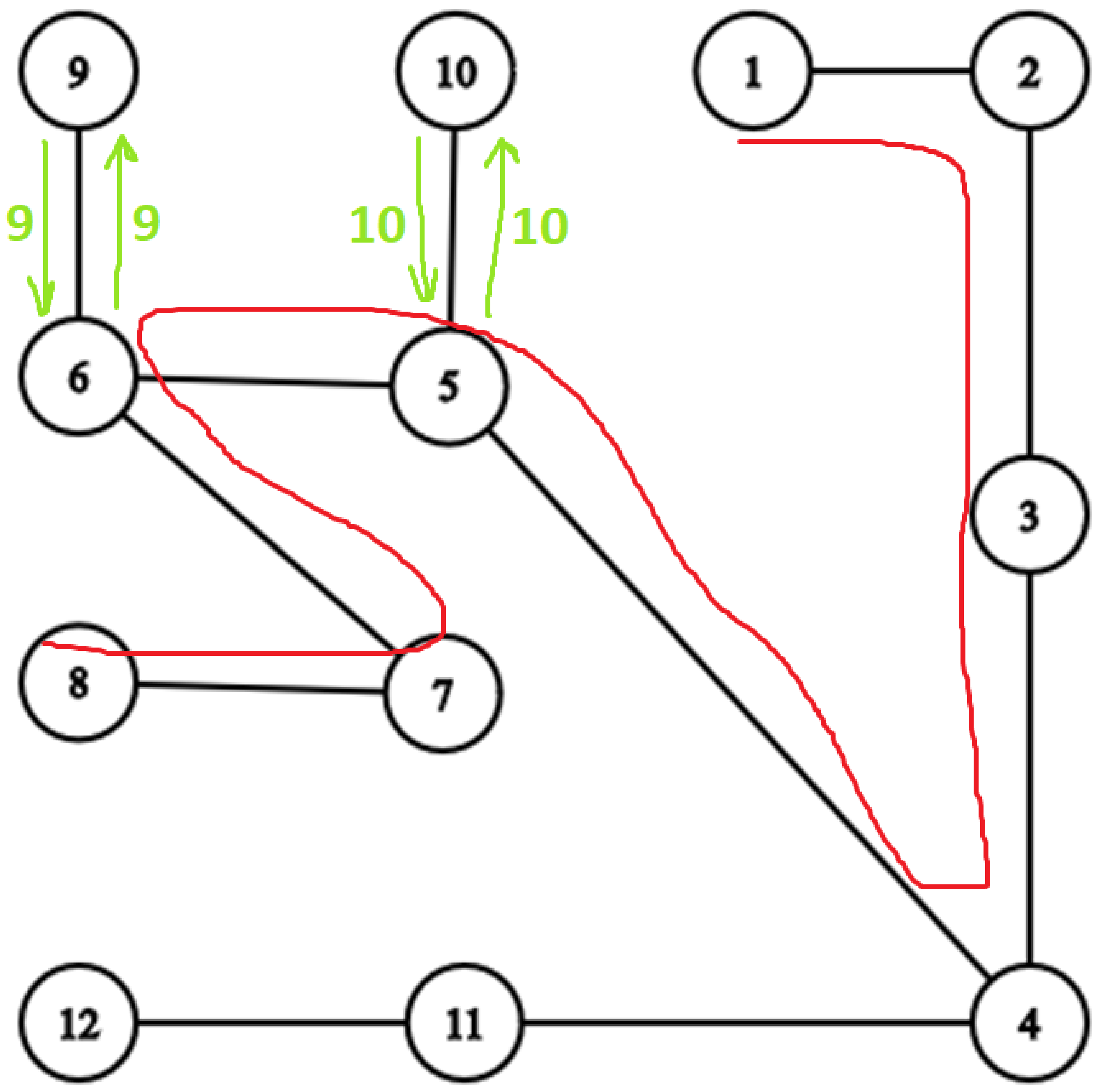
$m - d = 4$



diameter=7

m=11

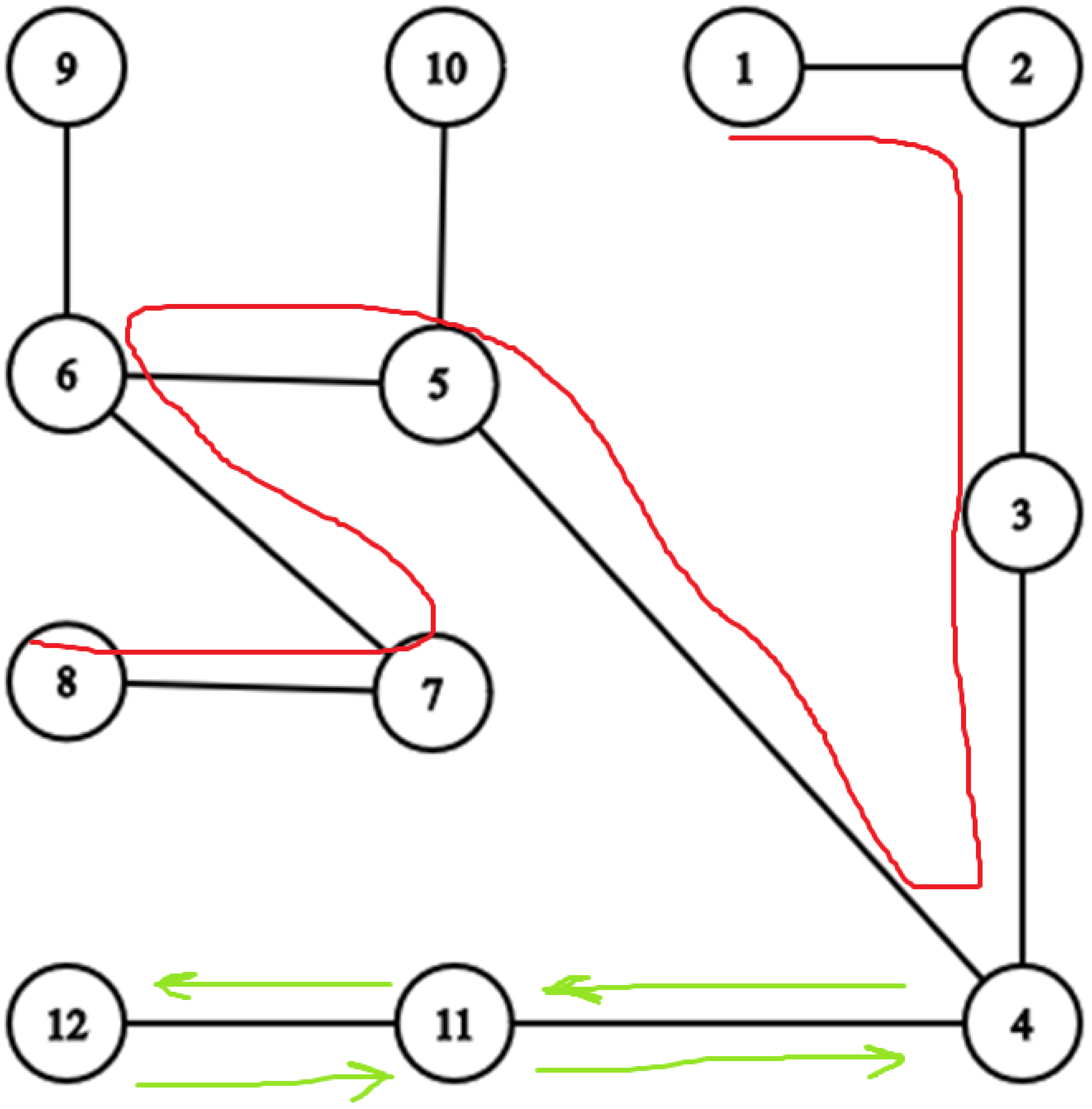
$m - d = 4$



diameter=7

m=11

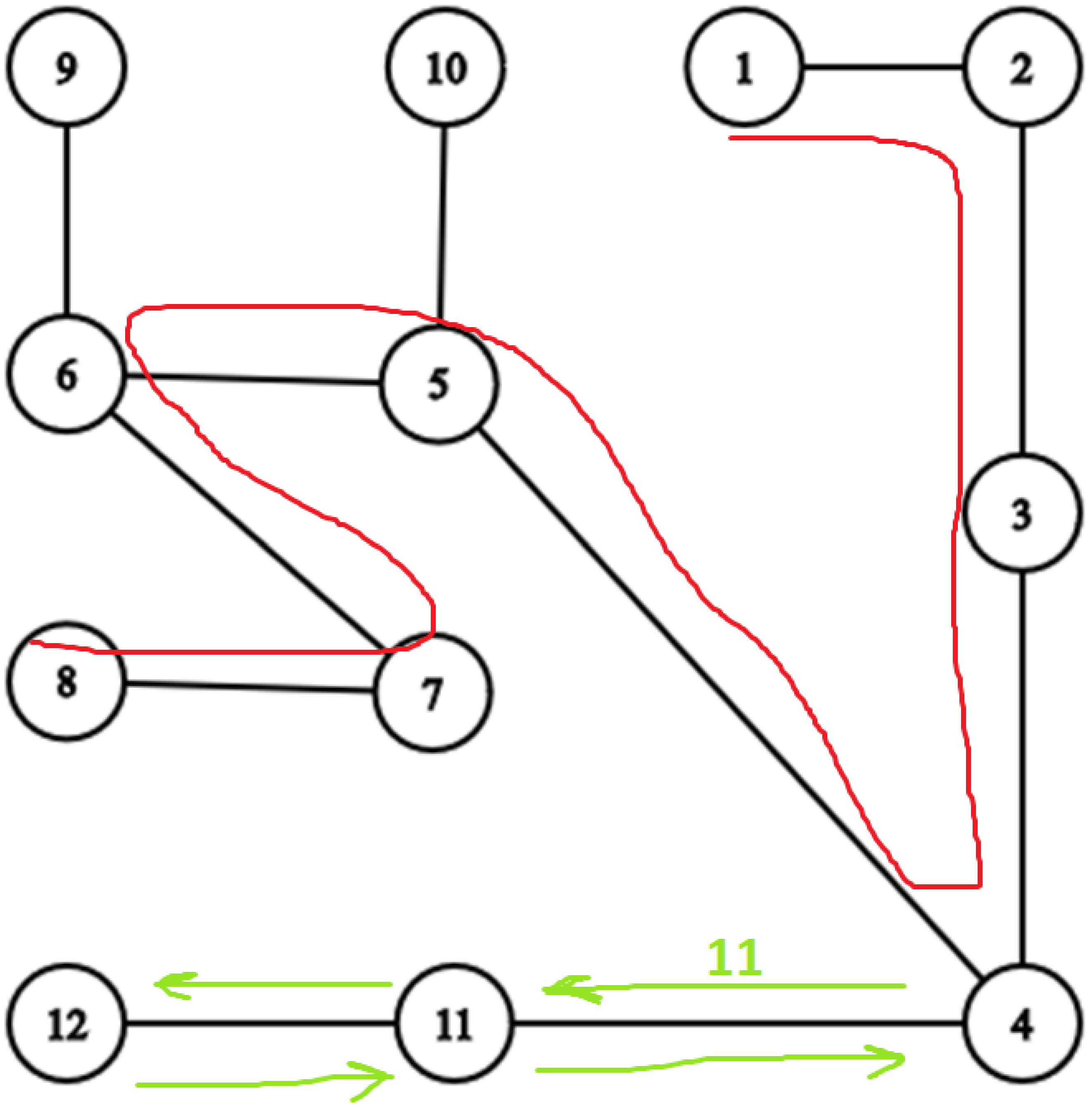
m − d = 4



diameter=7

m=11

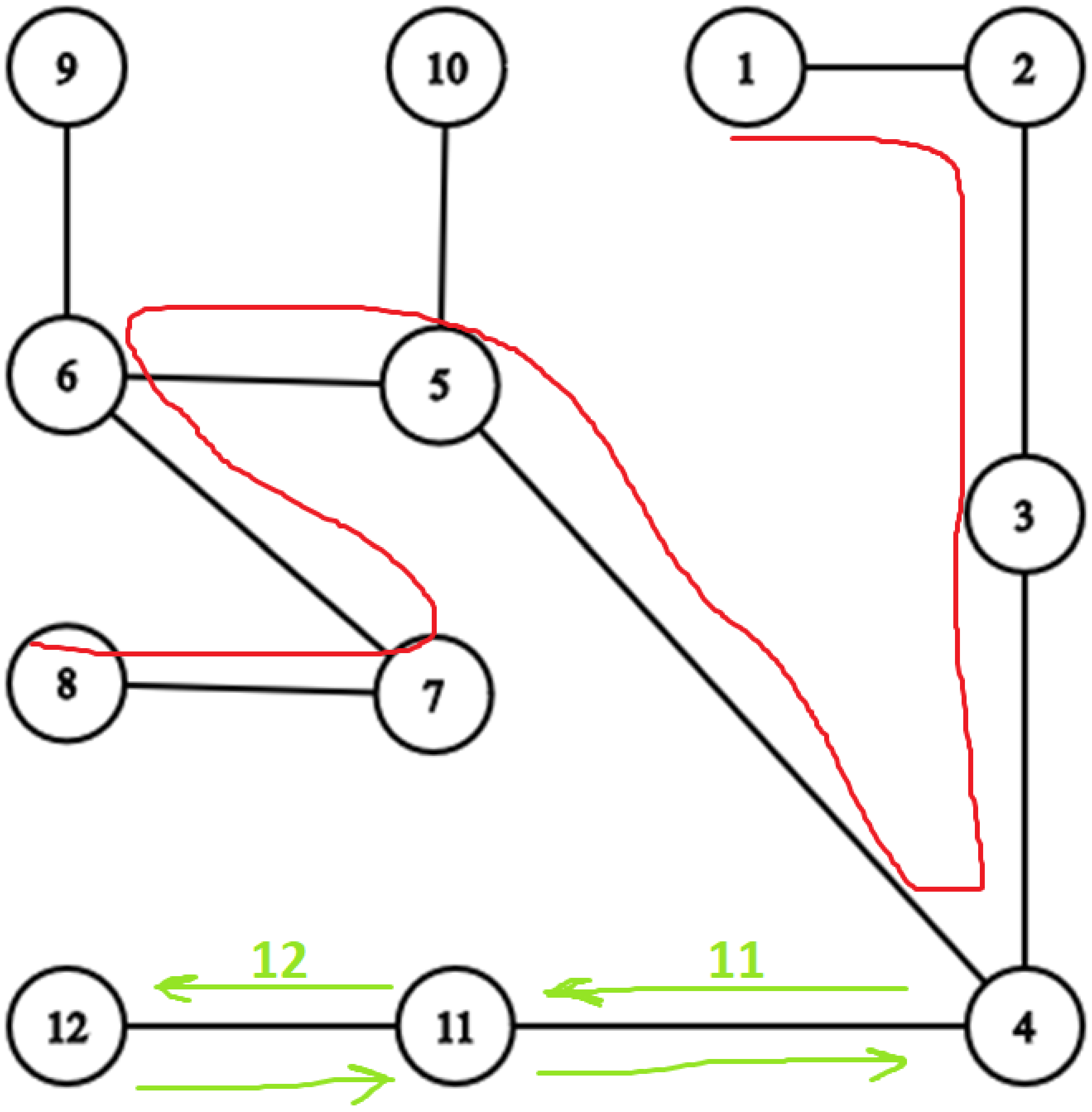
$m - d = 4$



diameter=7

m=11

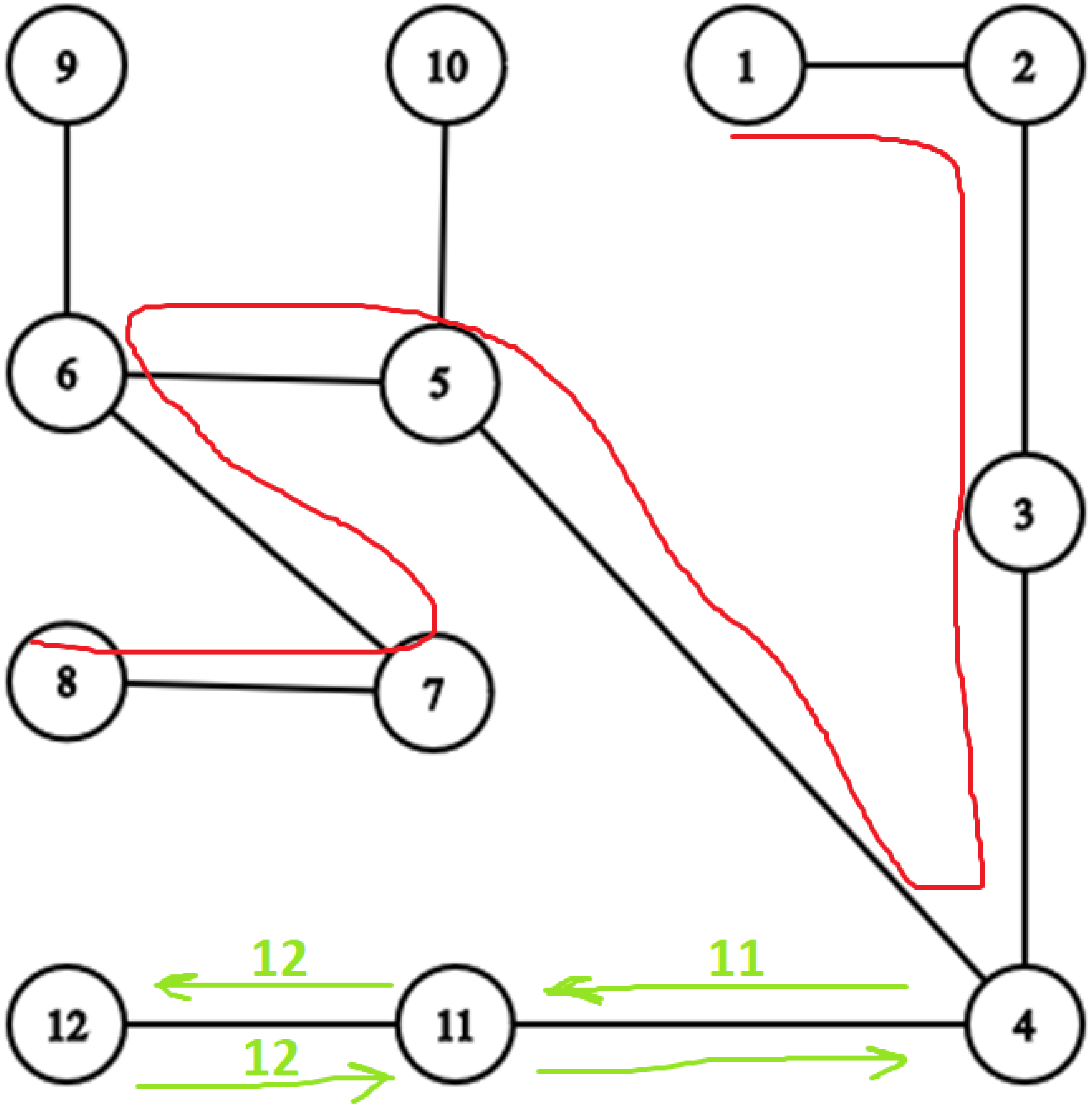
$m - d = 4$



diameter=7

m=11

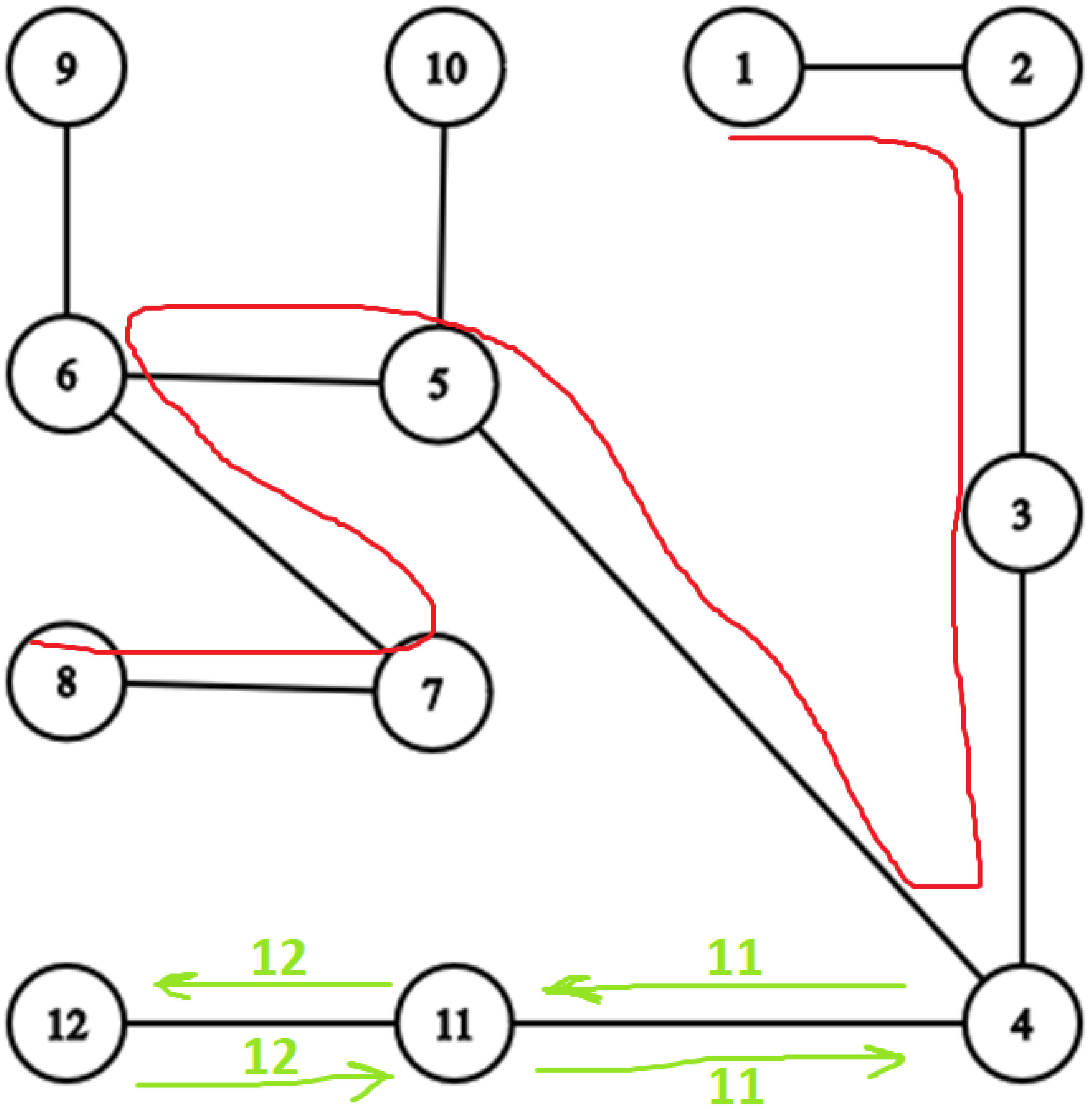
$m - d = 4$



diameter=7

m=11

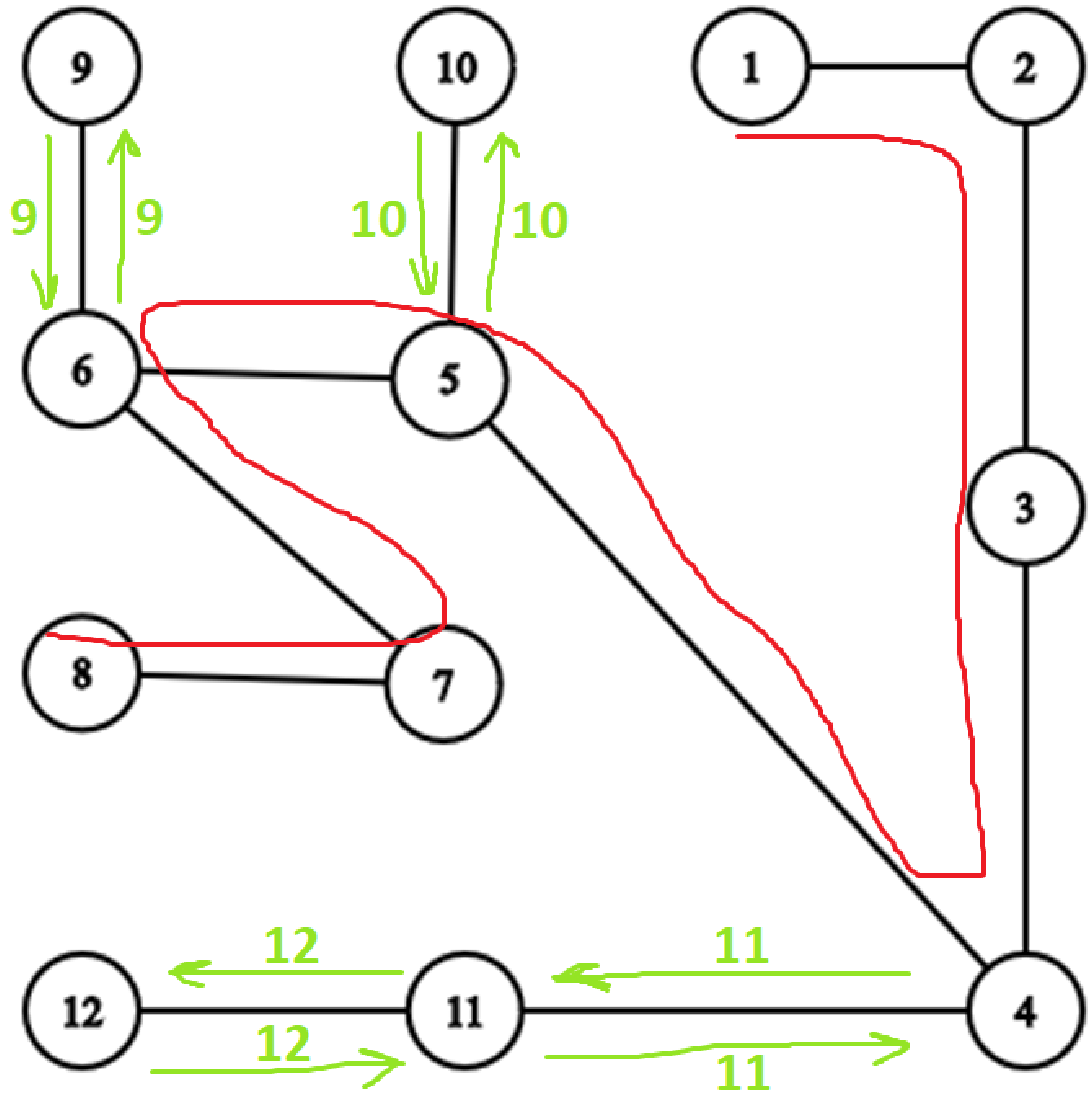
m − d = 4



diameter=7

m=11

m – d = 4

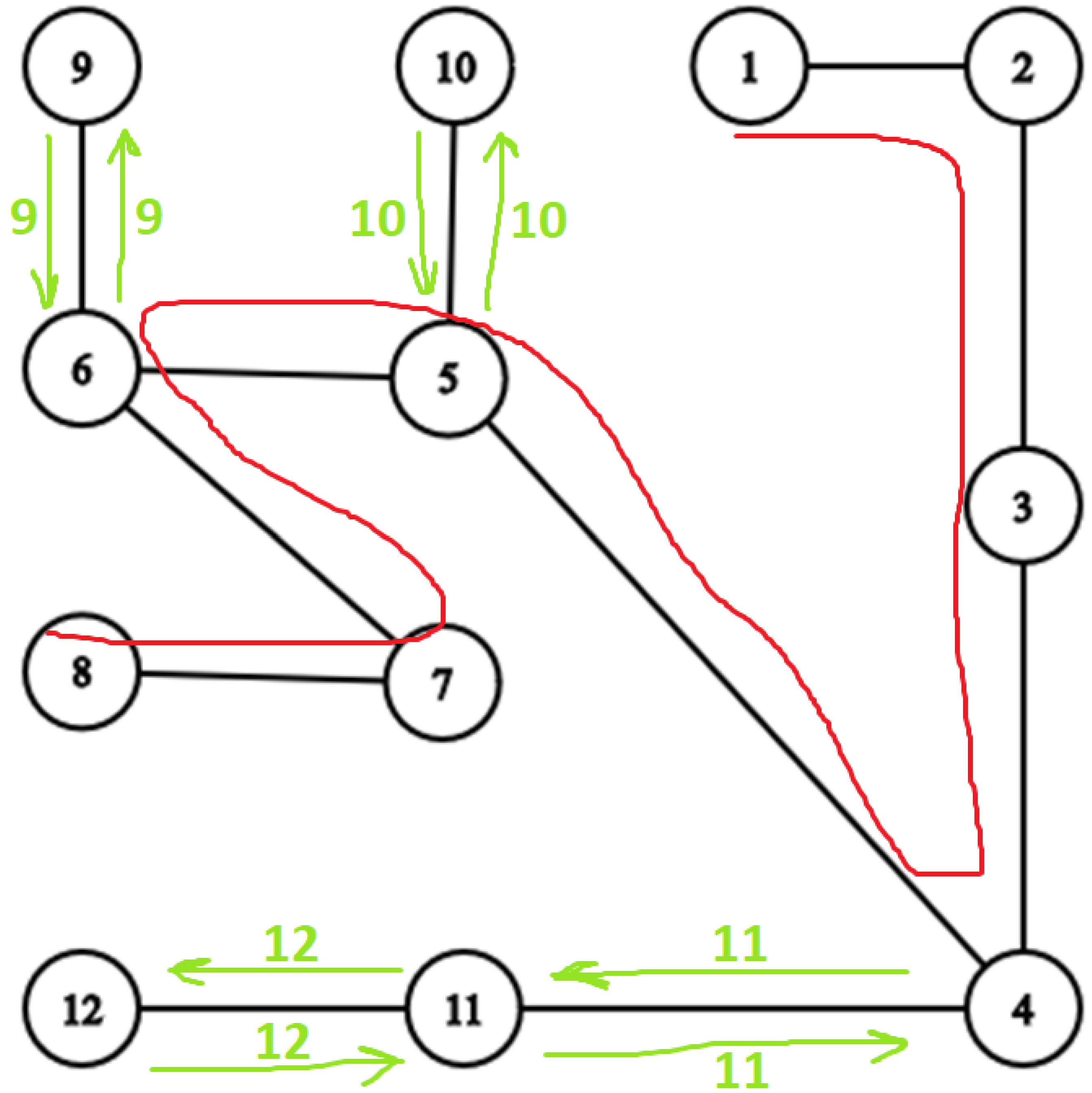


d+1

diameter=7

m=11

$m - d = 4$

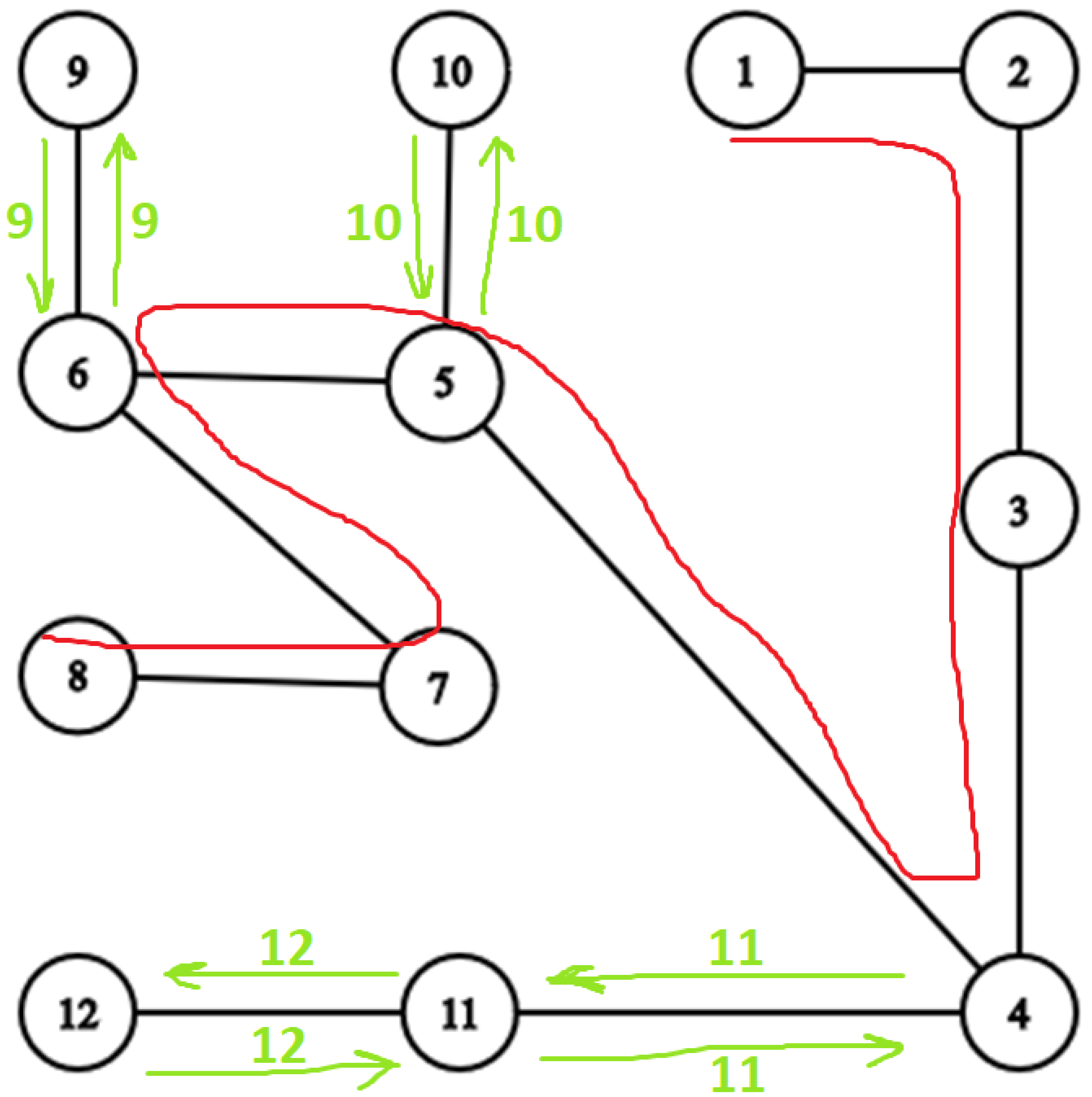


$$d+1+\frac{m-d}{2}$$

diameter=7

m=11

$$m-d=4$$

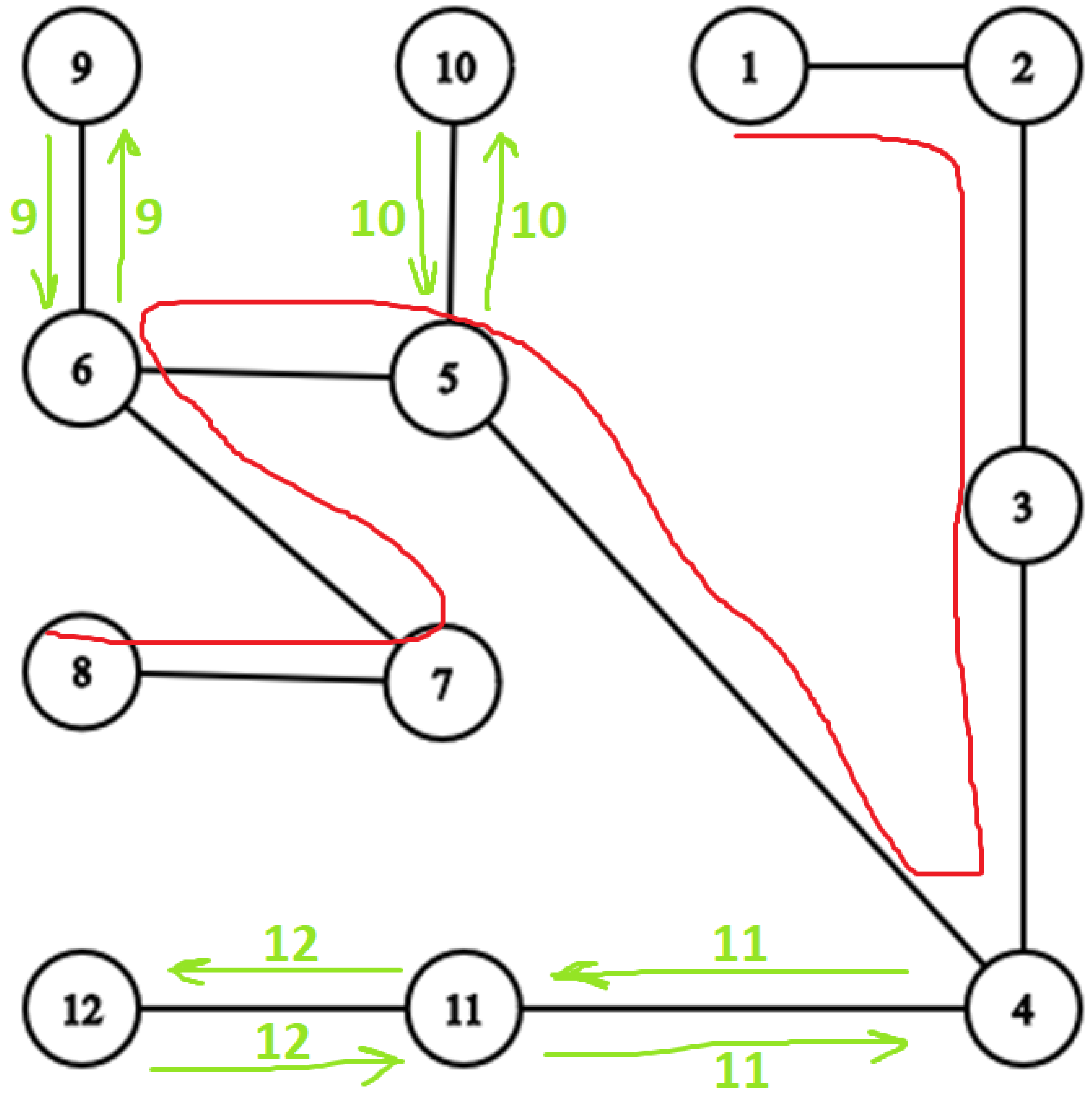


$$d+1+\frac{m-d}{2}$$

diameter=7

m=10

$$m-d=3$$

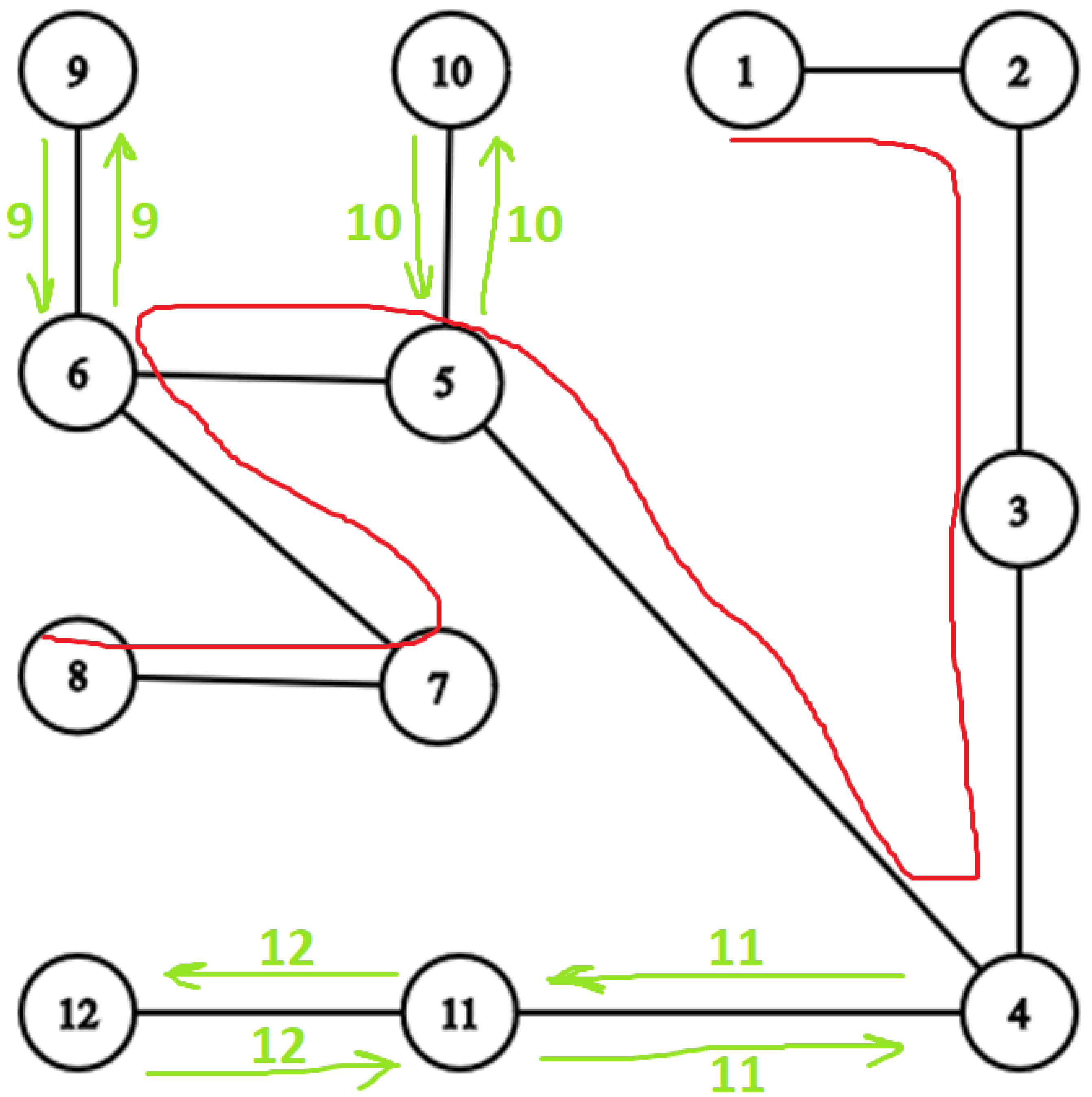


$$d+1+\left\lfloor \frac{m-d}{2} \right\rfloor$$

diameter=7

m=10

$$m-d=3$$

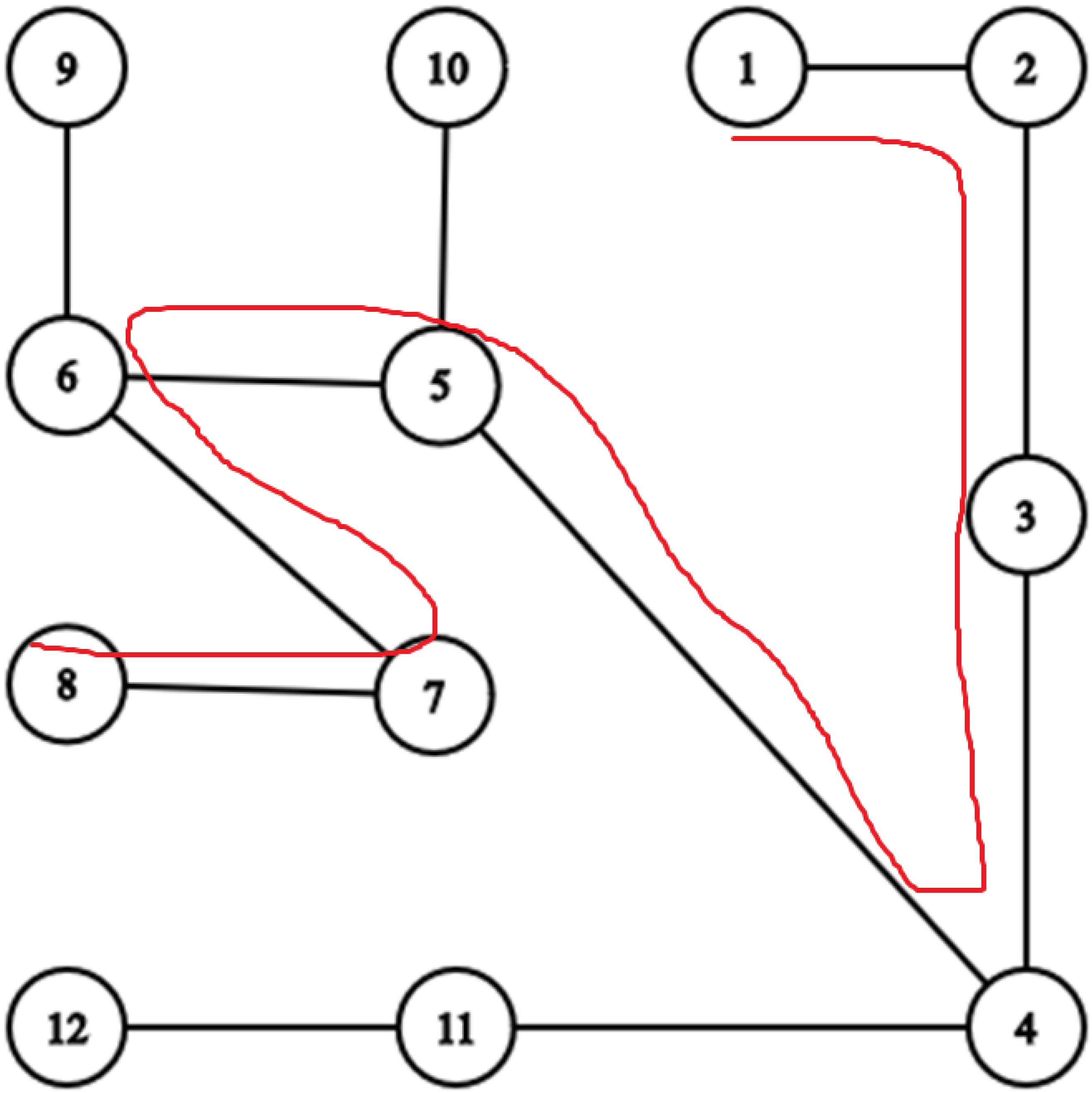


Any special case?

diameter=7

m=100

$m - d = 93$

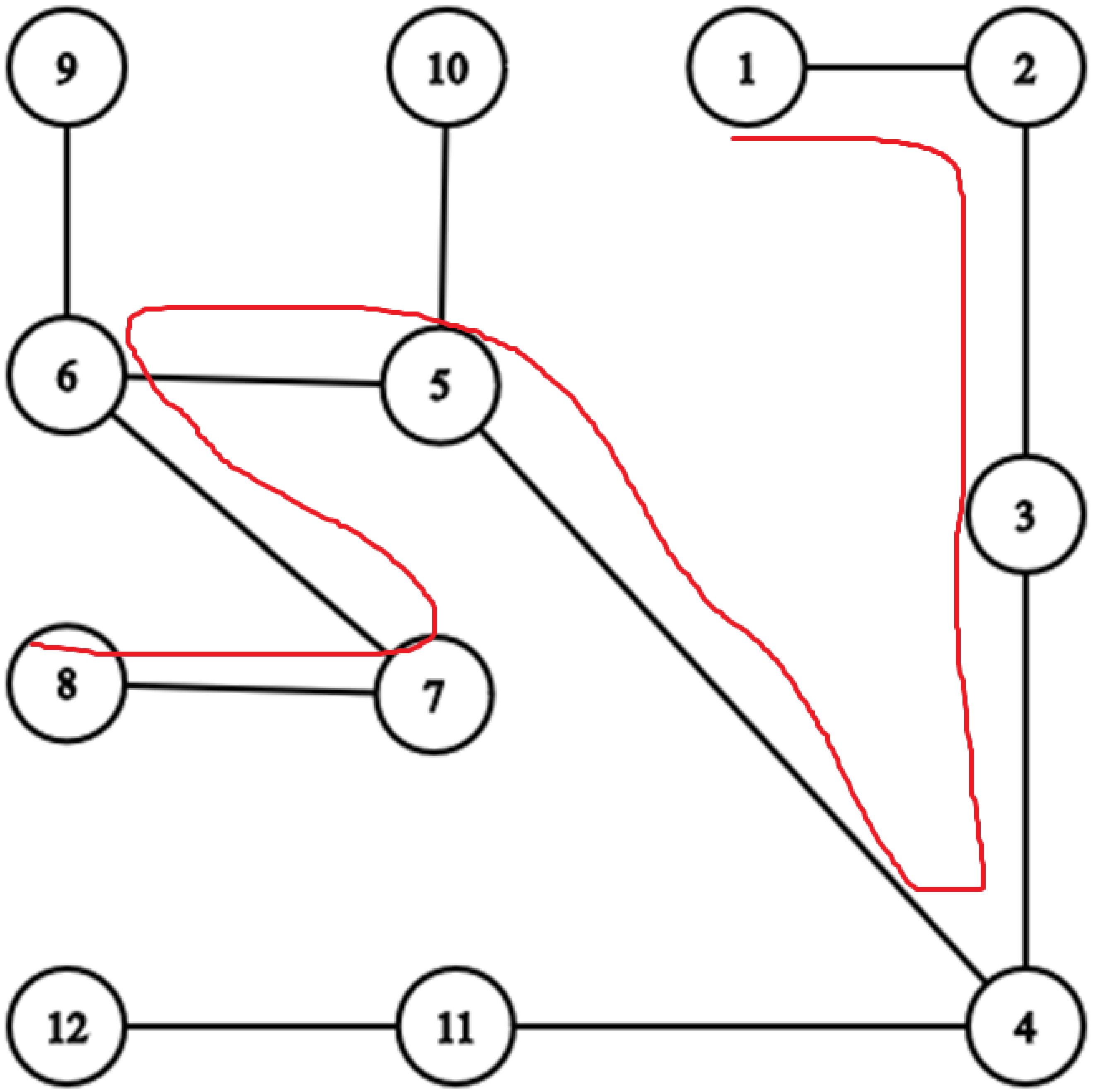


$$d+1+\left\lfloor\frac{m-d}{2}\right\rfloor$$

diameter=7

m=100

$m - d = 93$

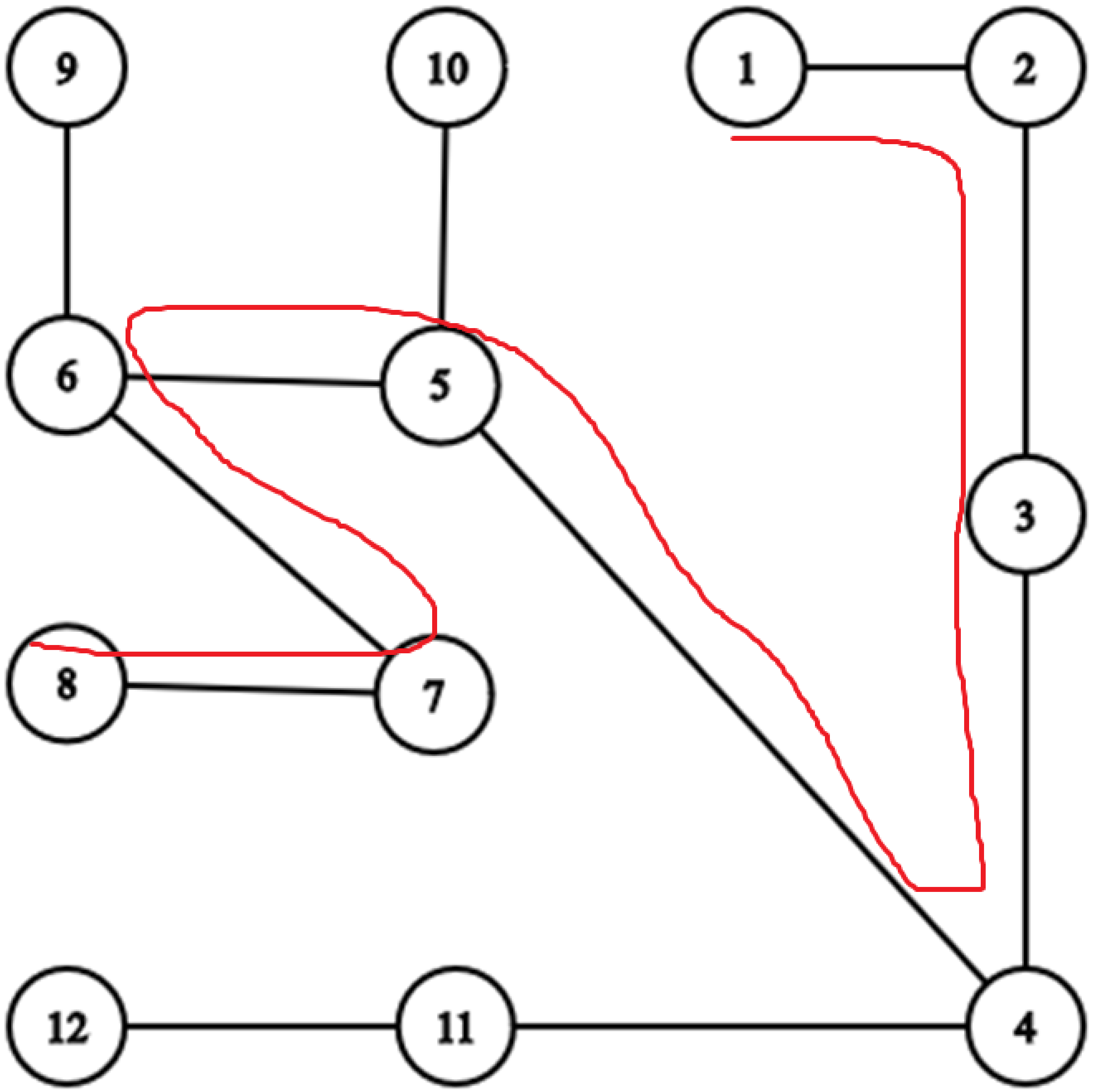


$$\min\left(d+1+\left\lfloor\frac{m-d}{2}\right\rfloor,\ \ n\right)$$

diameter=7

m=100

$m - d = 93$



$m \leq d$ Answer: $m+1$

$m > d$ Answer: $\min\left(d+1+\left\lfloor \frac{m-d}{2} \right\rfloor,\ n\right)$

😊😊😊

How to get diameter?

naïve approach

- Run dfs/bfs from every node

naïve approach

- Run dfs/bfs from every node

  $O(n(V+E))=O(n(n+n-1))=O(n^2)$

naïve approach

- Run dfs/bfs from every node

$$O(n(V+E))=O(n(n+n-1))=O(n^2)$$

Is it good enough? 🤔

naïve approach

- Run dfs/bfs from every node

$$O(n(V+E))=O(n(n+n-1))=O(n^2)$$

Is it good enough? 🤔

But we can do better than that… 😌

greedy approach

- pick an arbitrary vertex *s*

greedy approach

- pick an arbitrary vertex ***s***
- find farthest vertex ***r*** to s

greedy approach

- pick an arbitrary vertex ***s***
- find farthest vertex ***r*** to s
- find farthest vertex ***t*** to r

greedy approach

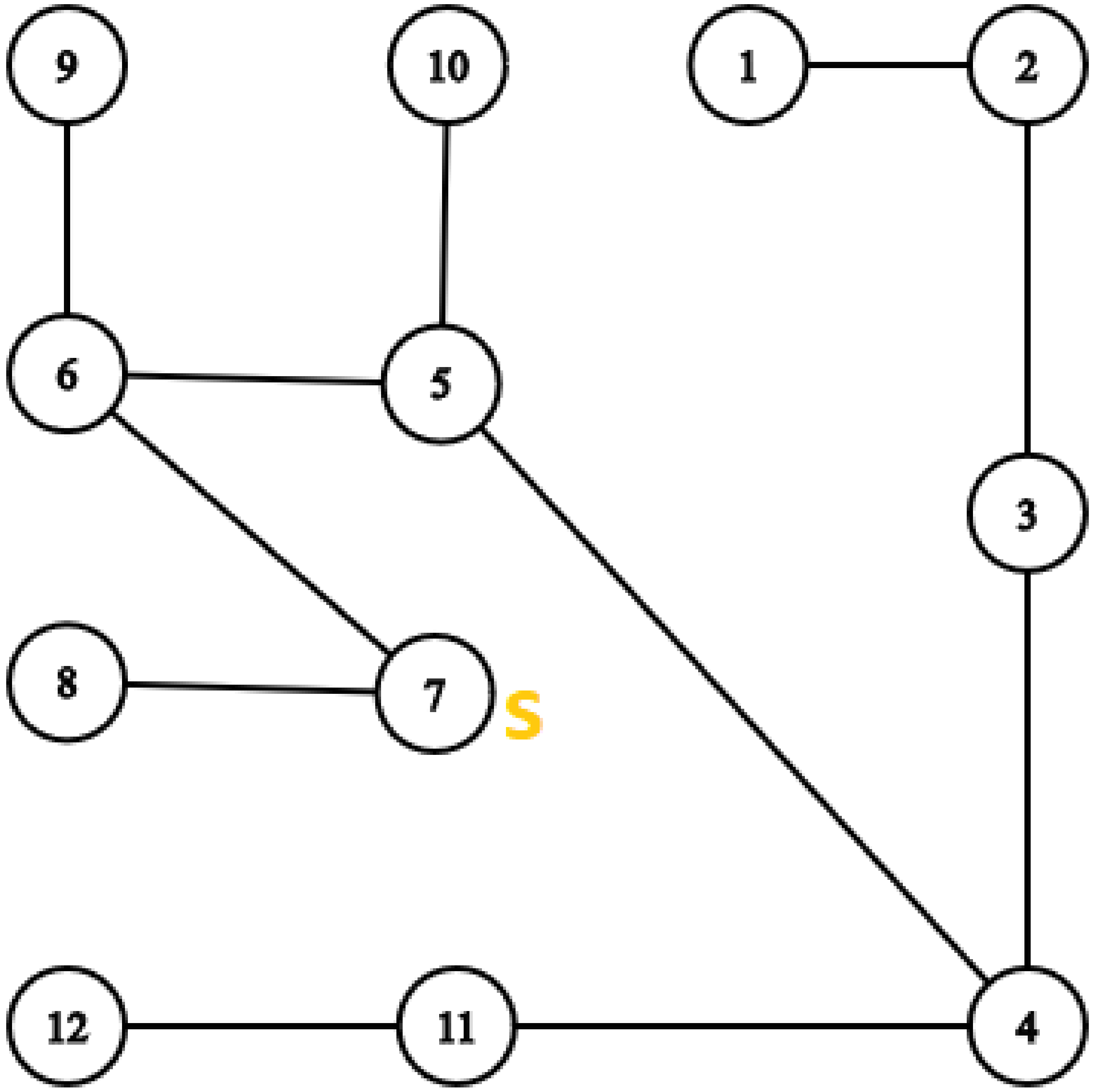
- pick an arbitrary vertex ***s***
- find farthest vertex ***r*** to s
- find farthest vertex ***t*** to r
- t to r path is a diameter!

greedy approach

- pick an arbitrary vertex *s*
- find farthest vertex *r* to s
- find farthest vertex *t* to r
- t to r path is a diameter!

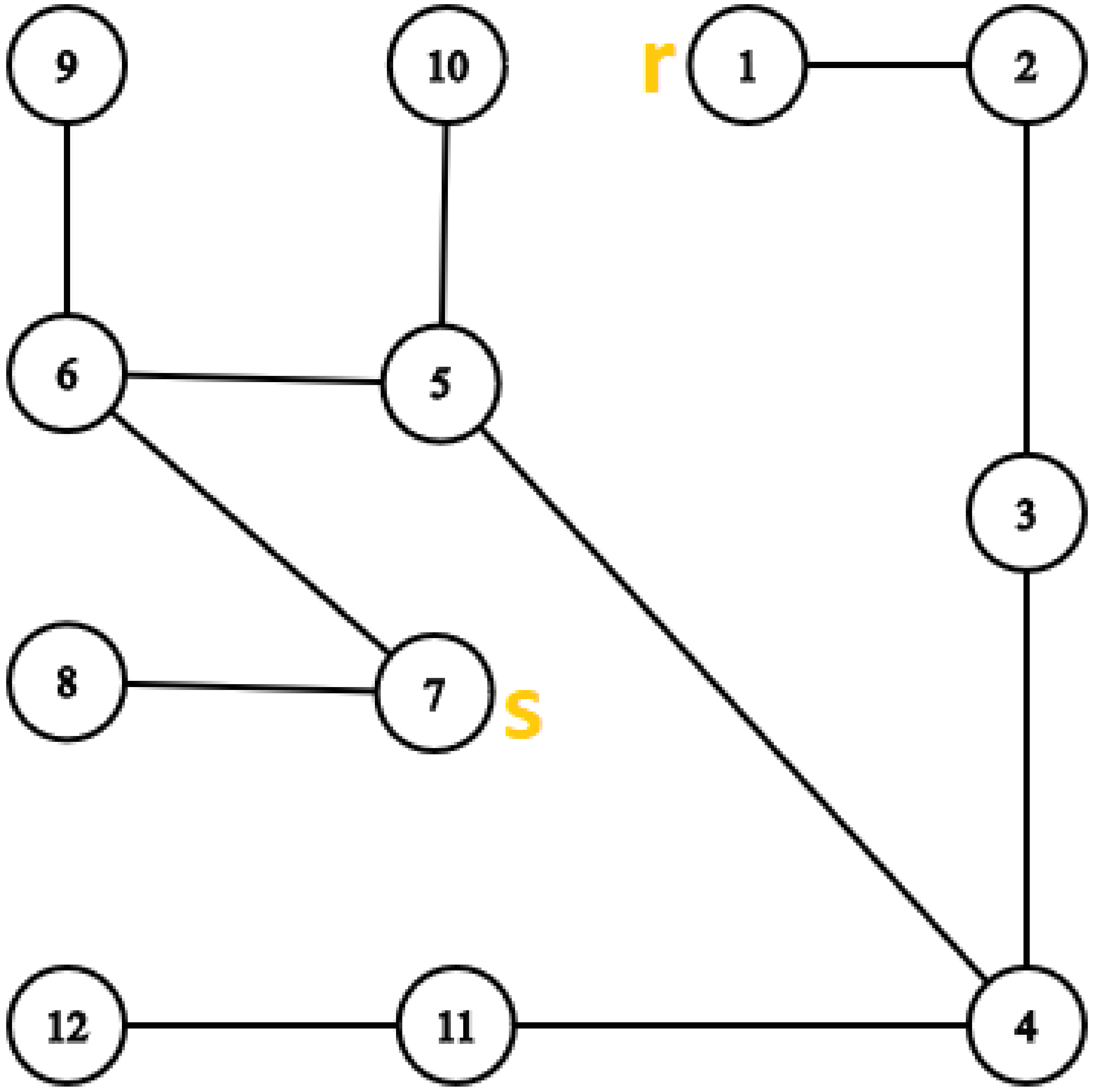
greedy approach

- pick an arbitrary vertex ***s***
- find farthest vertex ***r*** to s
- find farthest vertex ***t*** to r
- t to r path is a diameter!



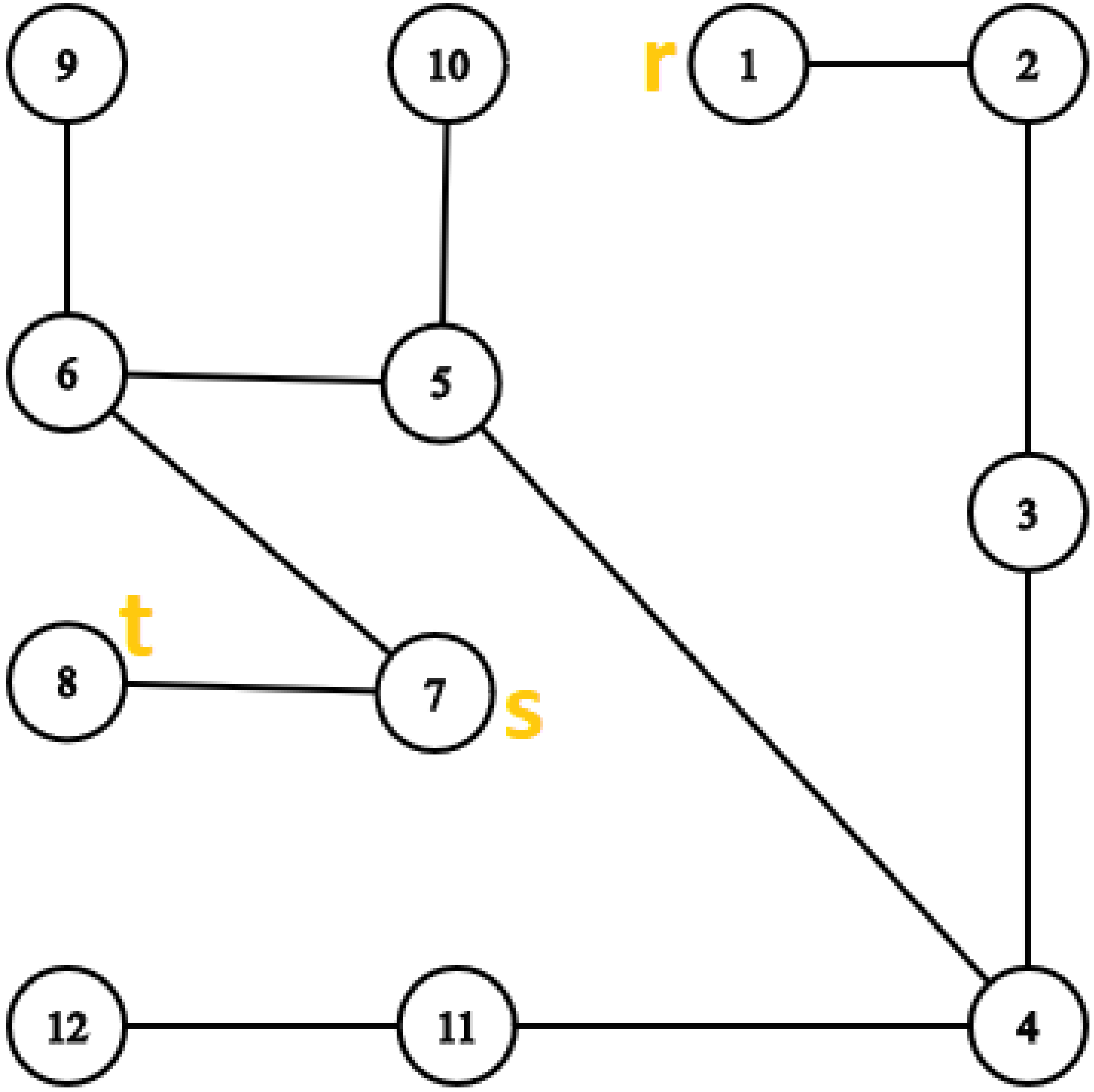
greedy approach

- pick an arbitrary vertex ***s***
- find farthest vertex ***r*** to s
- find farthest vertex ***t*** to r
- t to r path is a diameter!



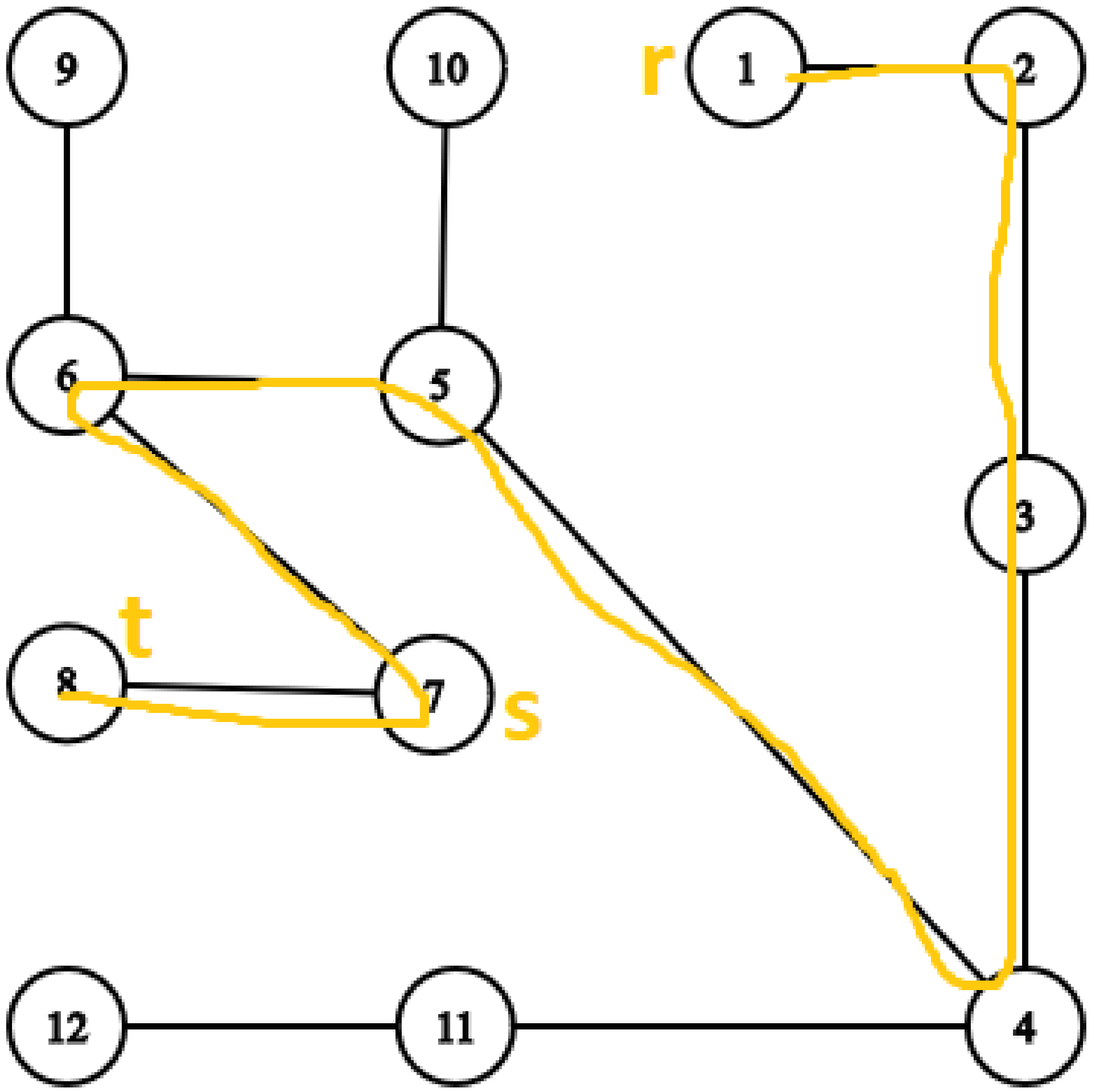
greedy approach

- pick an arbitrary vertex ***s***
- find farthest vertex ***r*** to s
- find farthest vertex ***t*** to r
- t to r path is a diameter!



greedy approach

- pick an arbitrary vertex ***s***
- find farthest vertex ***r*** to s
- find farthest vertex ***t*** to r
- t to r path is a diameter!



proof of correctness

proof of correctness

picked ***s***

- case 1: ***s*** on diameter
- case 2: ***s*** not on diameter

proof of correctness

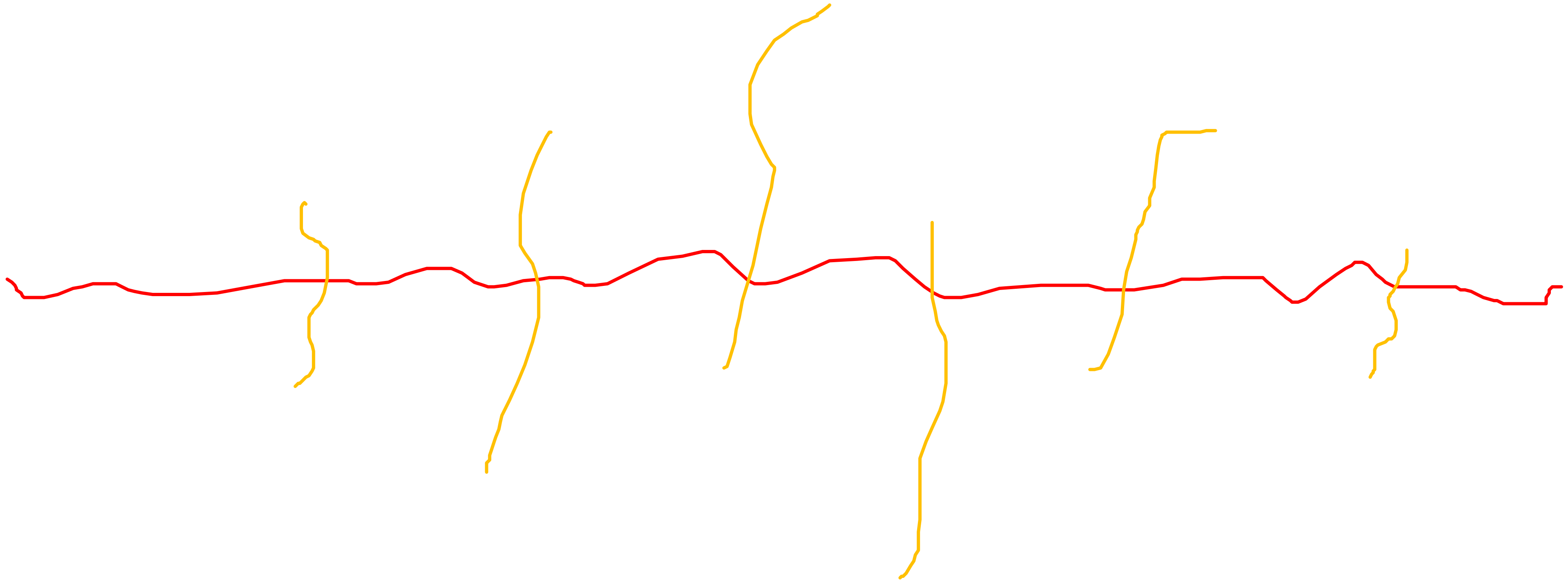
case 1: ***s*** on diameter

proof of correctness

case 1: ***s*** on diameter

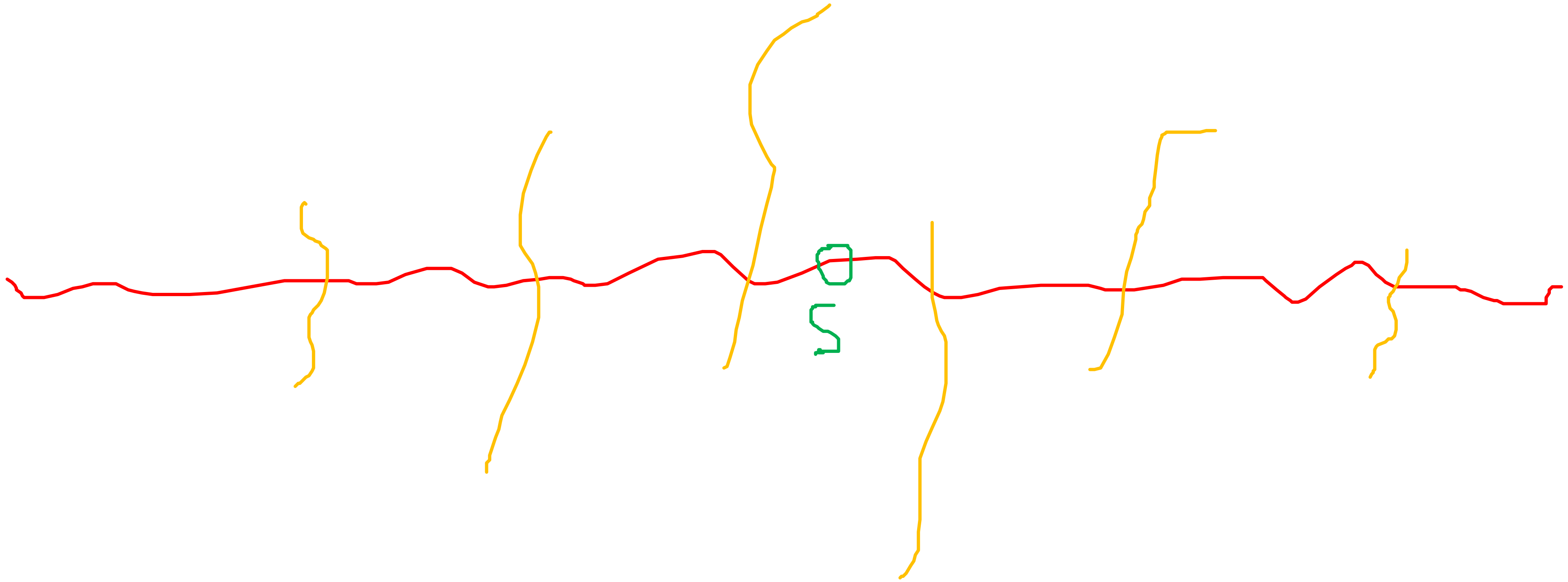
proof of correctness

case 1: ***s*** on diameter



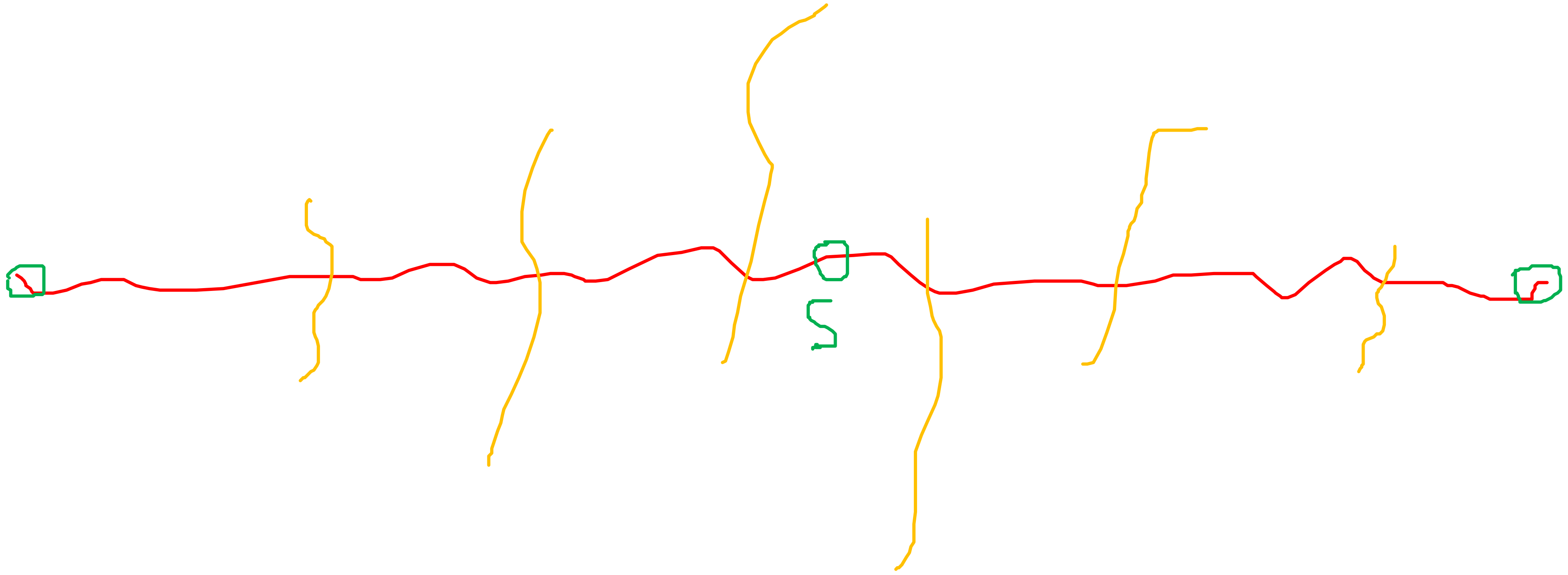
proof of correctness

case 1: ***s*** on diameter



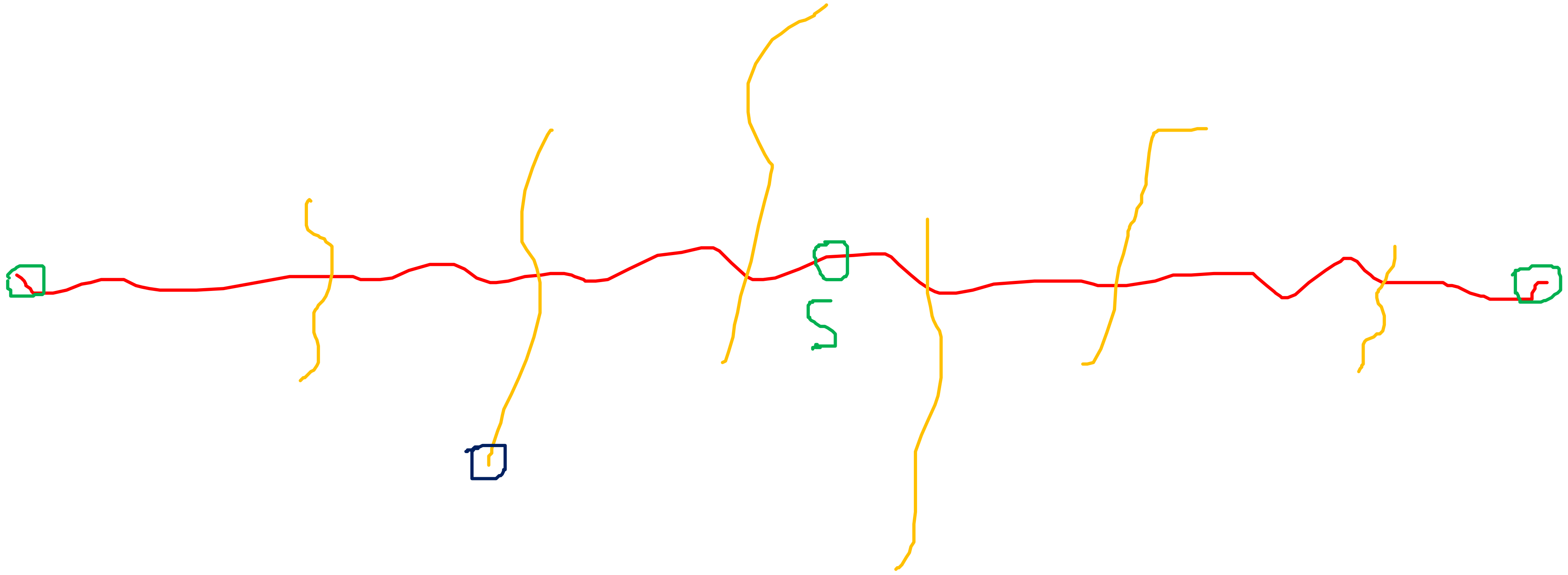
proof of correctness

case 1: ***s*** on diameter



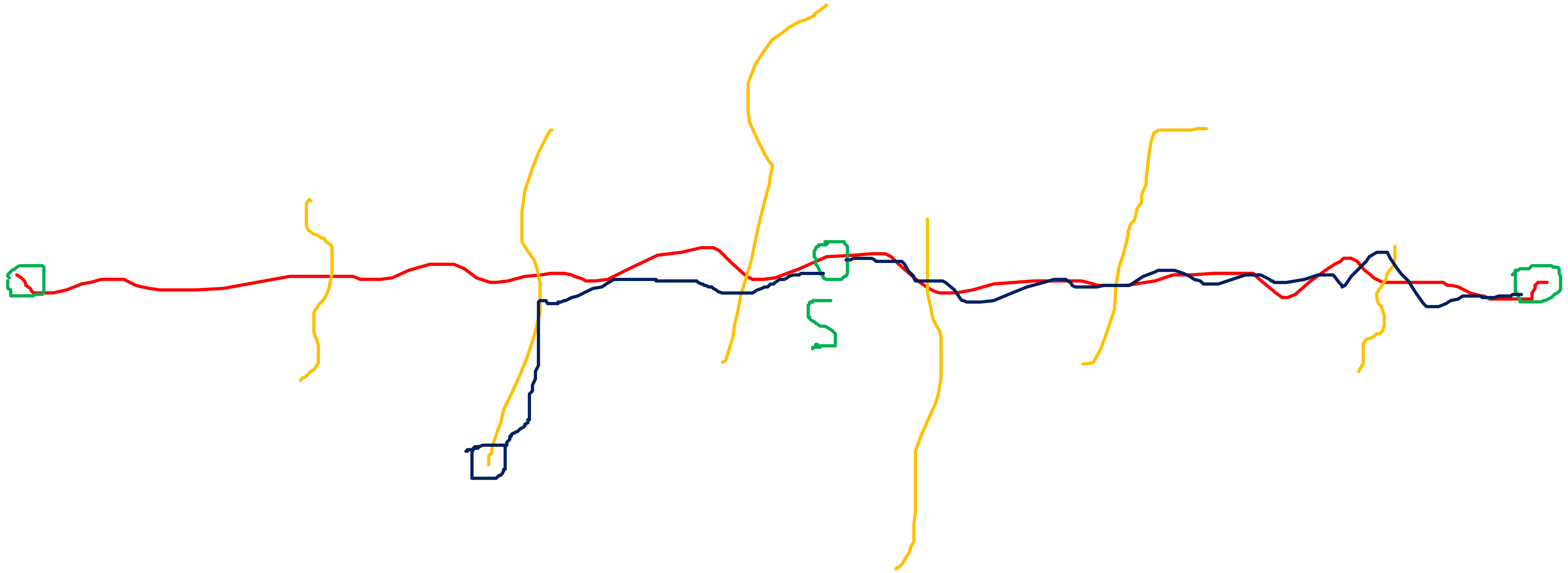
proof of correctness

case 1: ***s*** on diameter



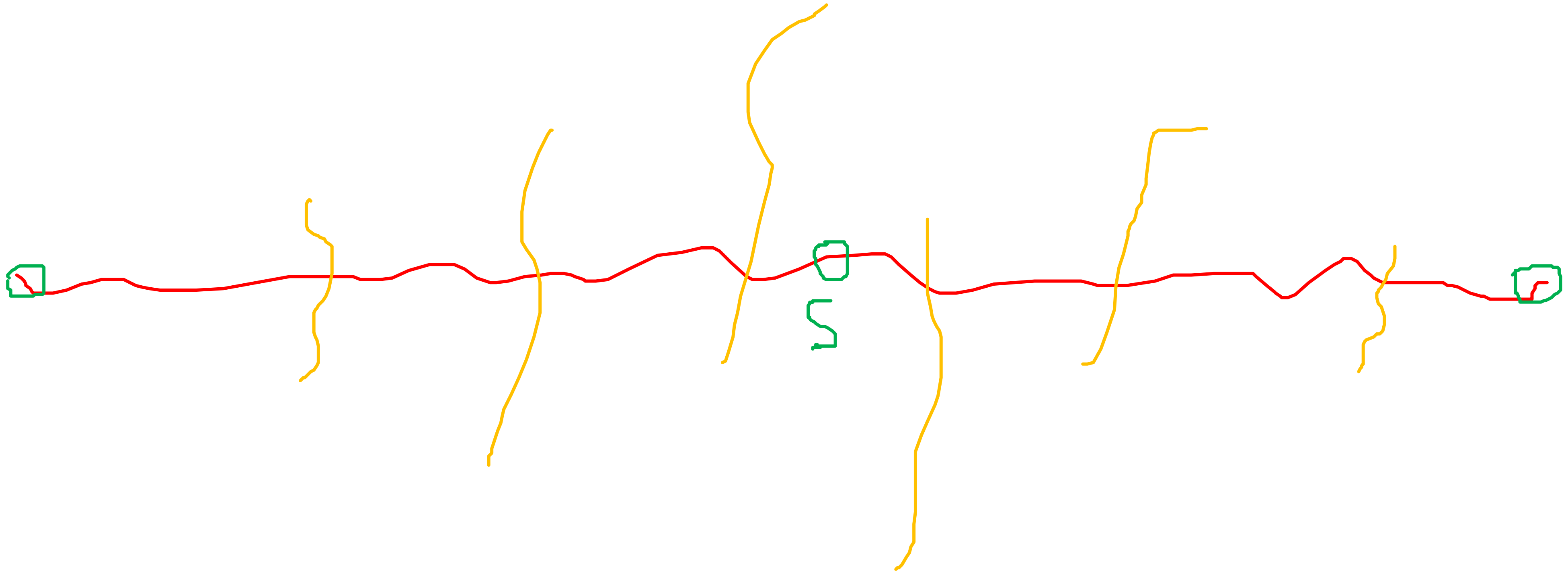
proof of correctness

case 1: ***s*** on diameter



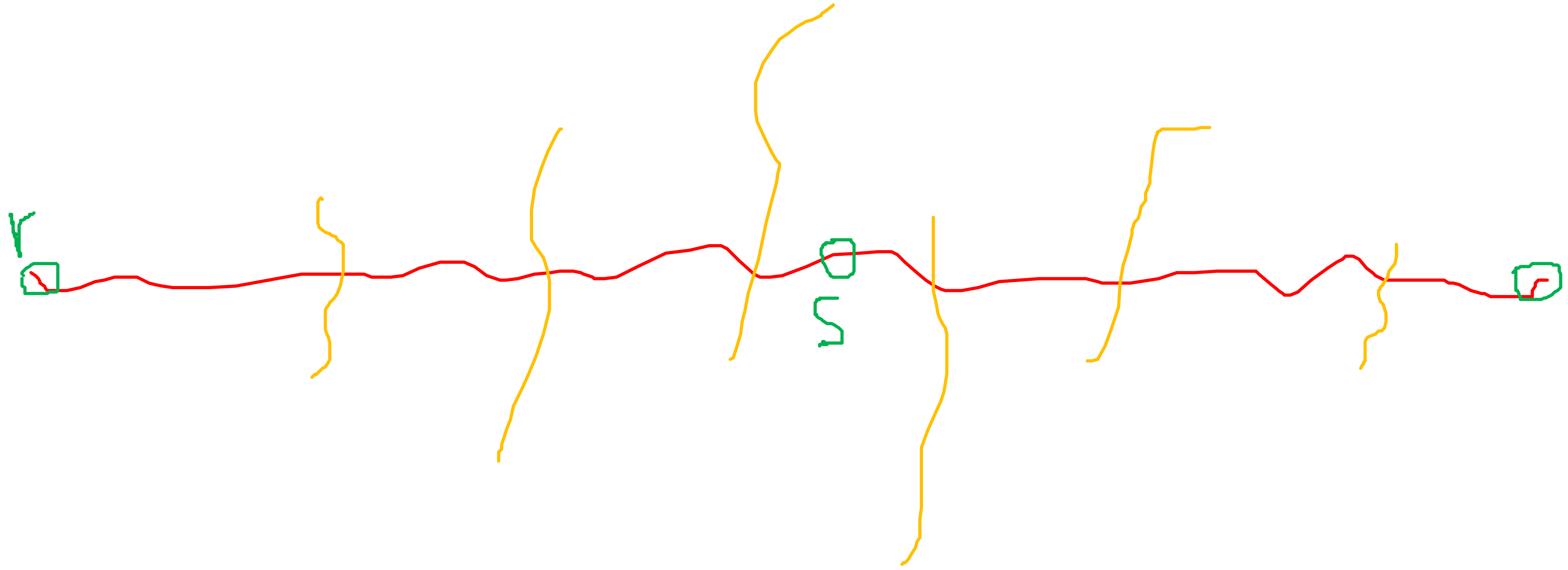
proof of correctness

case 1: *s* on diameter



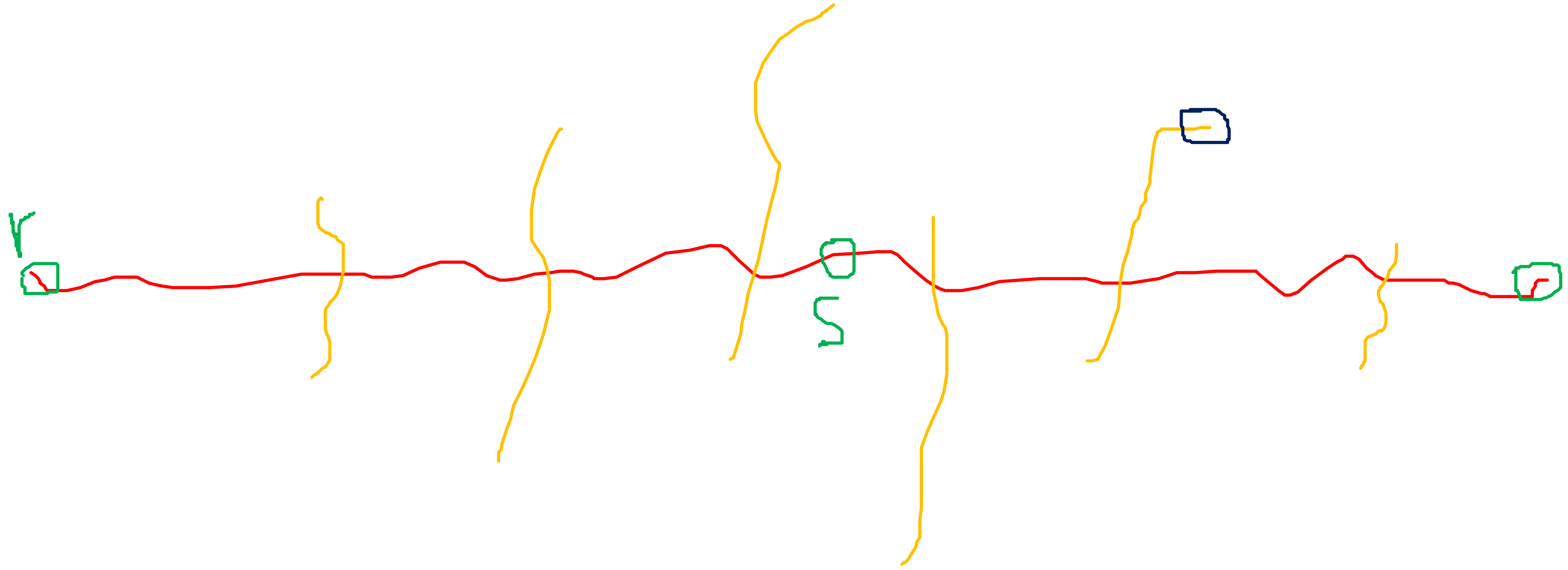
proof of correctness

case 1: ***s*** on diameter



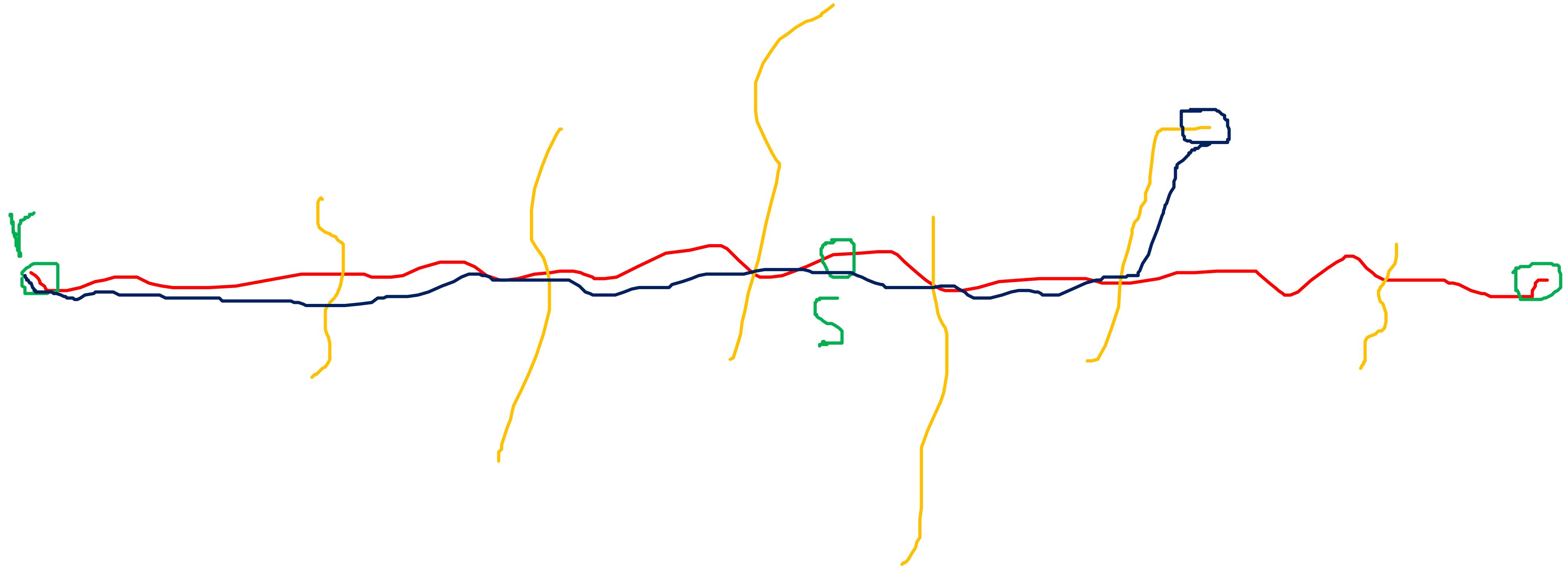
proof of correctness

case 1: ***s*** on diameter



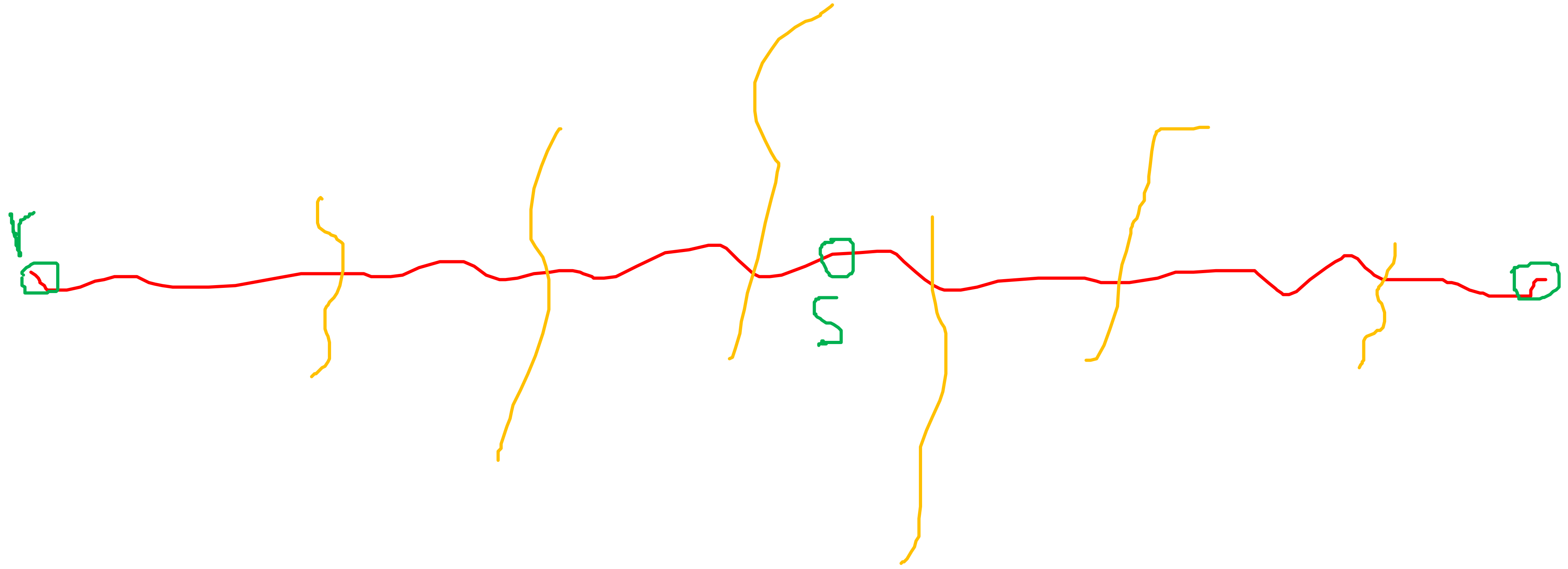
proof of correctness

case 1: ***s*** on diameter



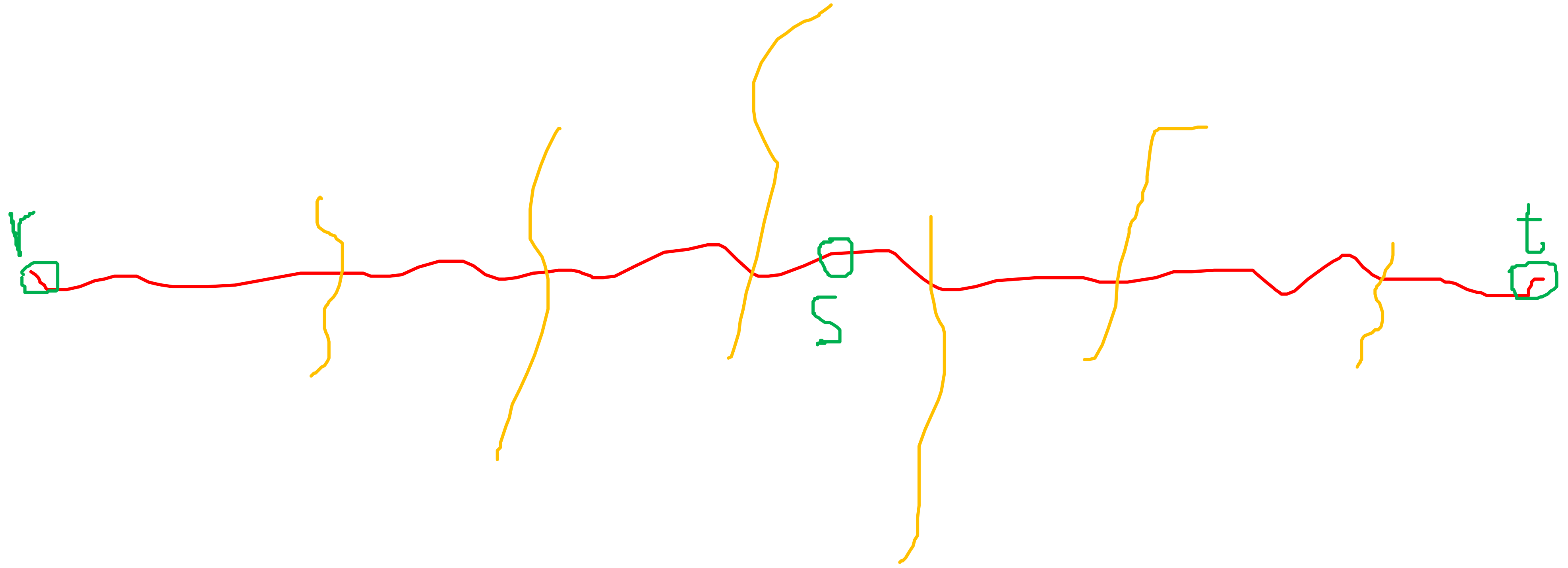
proof of correctness

case 1: ***s*** on diameter



proof of correctness

case 1: ***s*** on diameter



greedy approach

- pick an arbitrary vertex *s*
- find farthest vertex *r* to s
- find farthest vertex *t* to r
- t to r path is a diameter!

greedy approach

- pick an arbitrary vertex ***s***
- find farthest vertex ***r*** to s
- find farthest vertex ***t*** to r
- t to r path is a diameter!

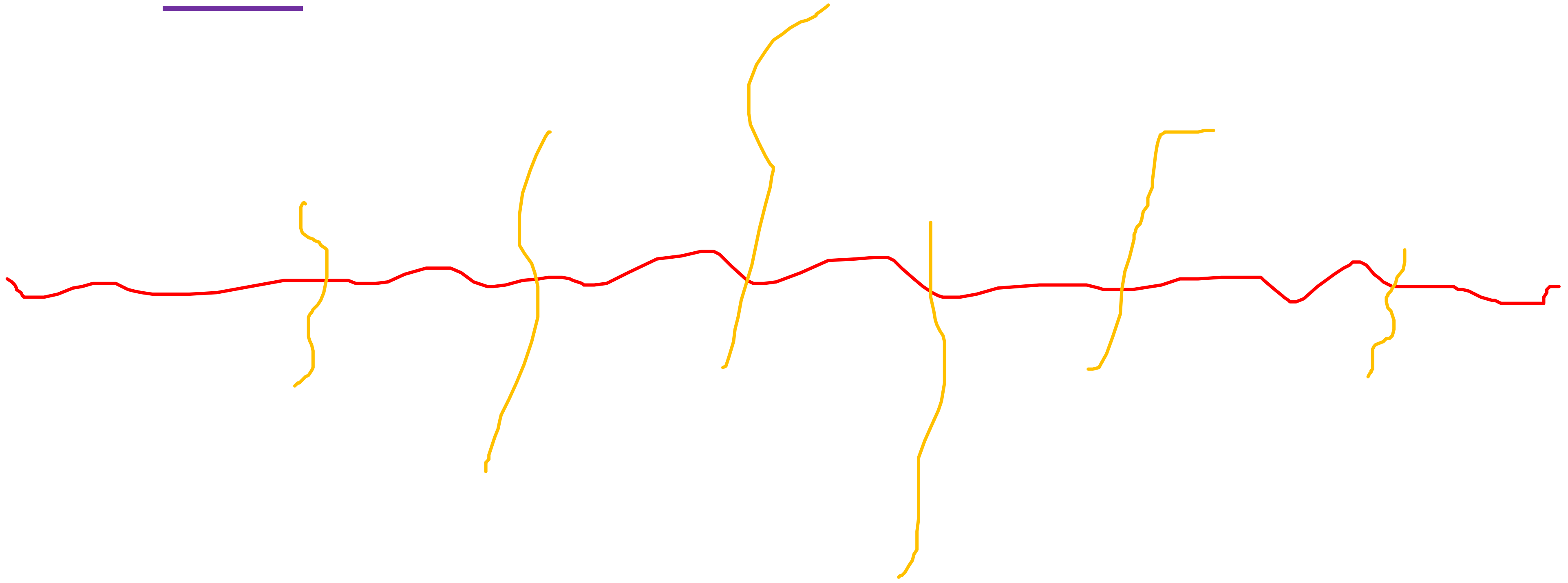
Case 1 proved
(***s*** on diameter)

proof of correctness

case 2: ***s*** not on diameter

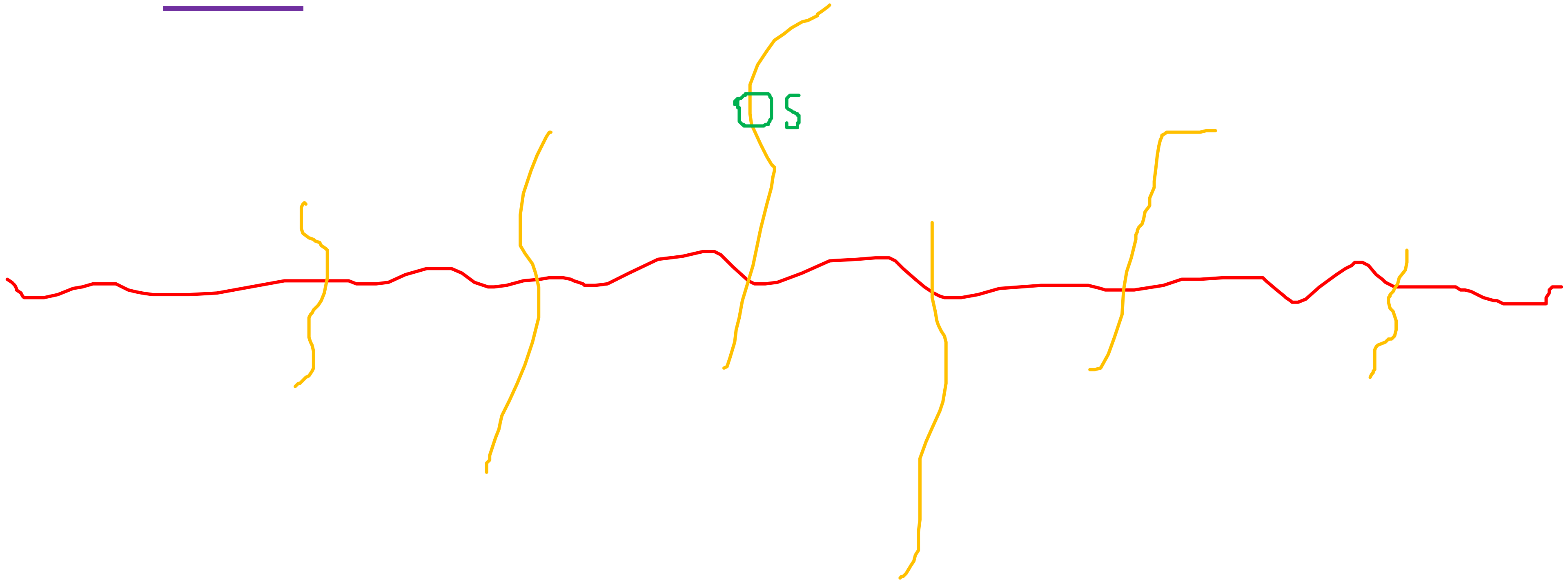
proof of correctness

case 2: ***s*** not on diameter



proof of correctness

case 2: ***s*** not on diameter



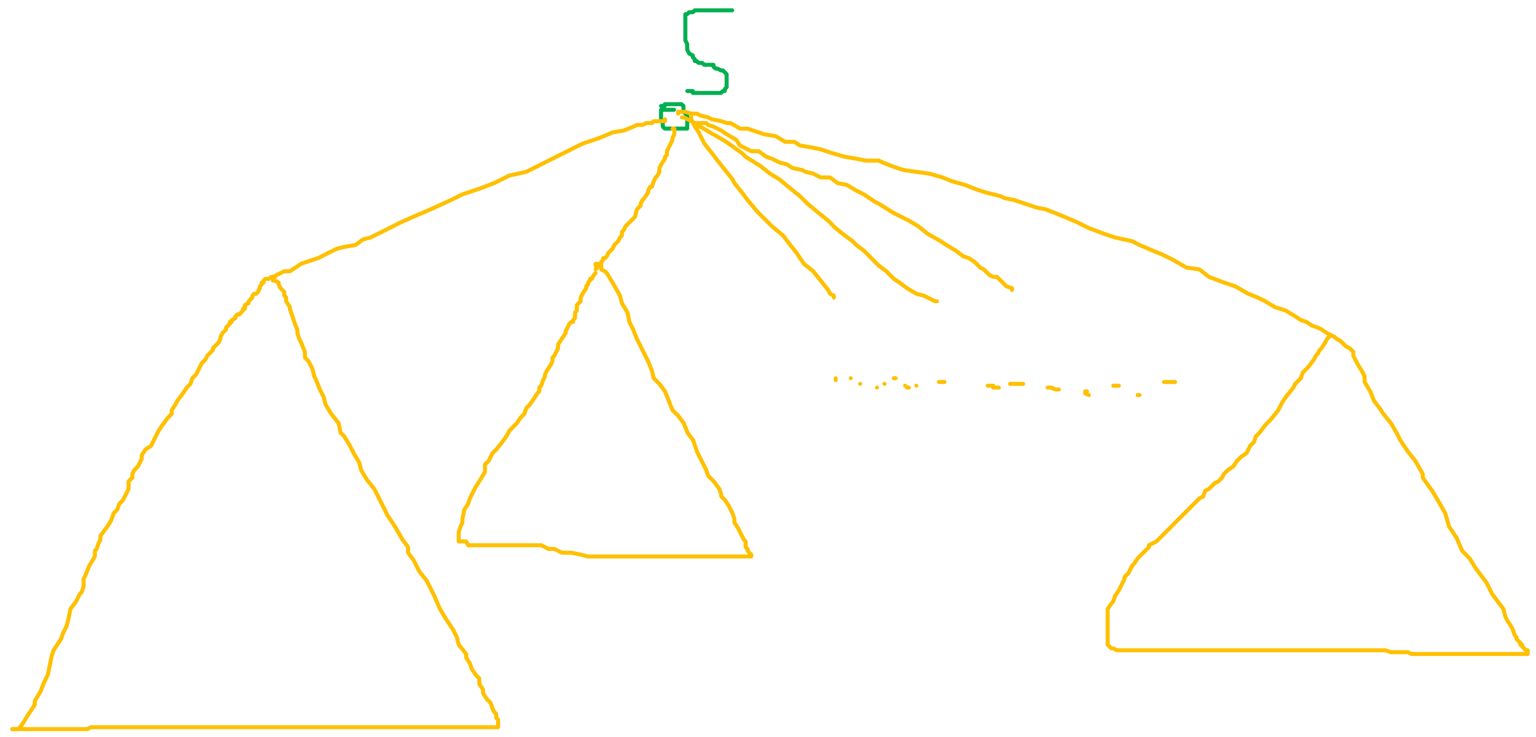
proof of correctness

case 2: ***s*** not on diameter

s

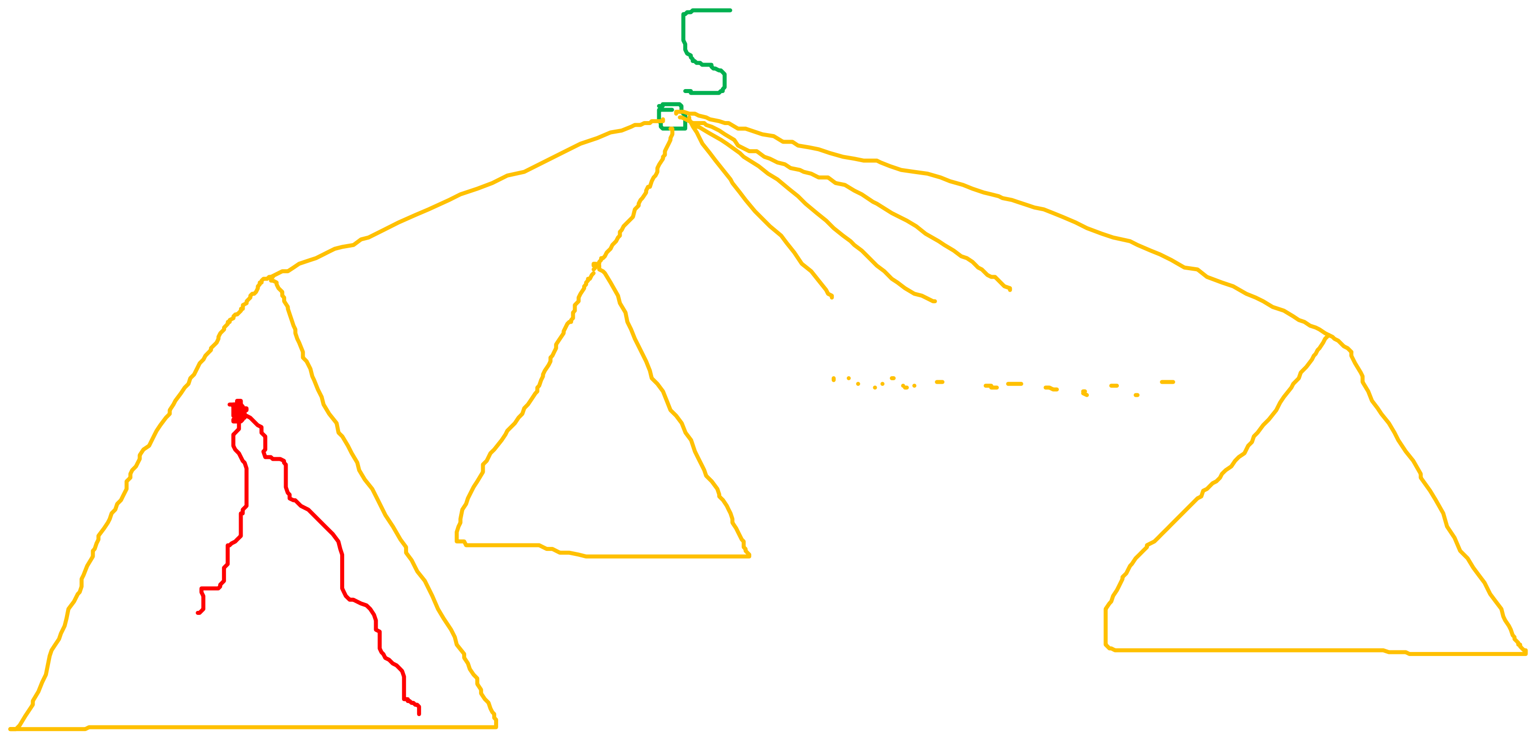
proof of correctness

case 2: ***s*** <u>not</u> on diameter



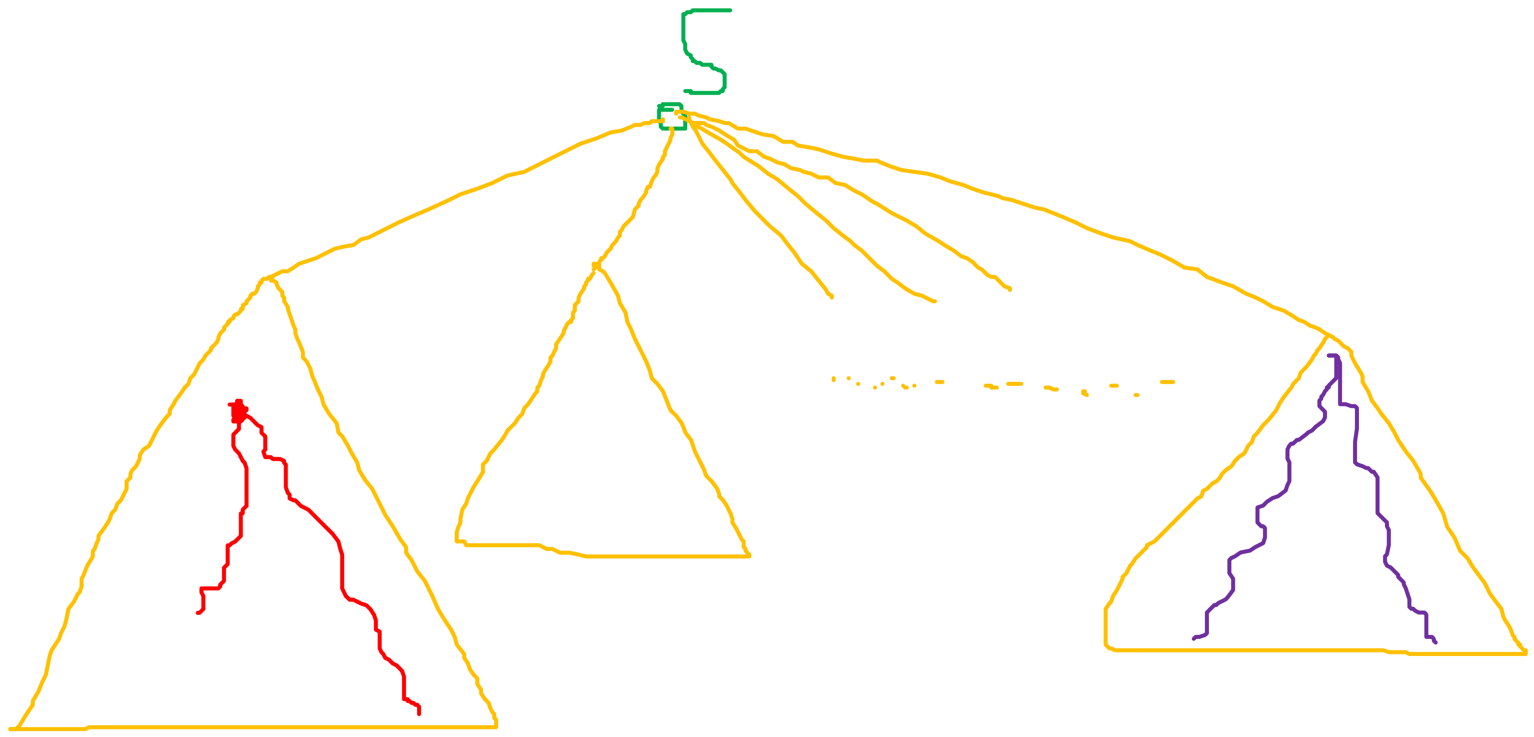
proof of correctness

case 2: ***s*** <u>not</u> on diameter



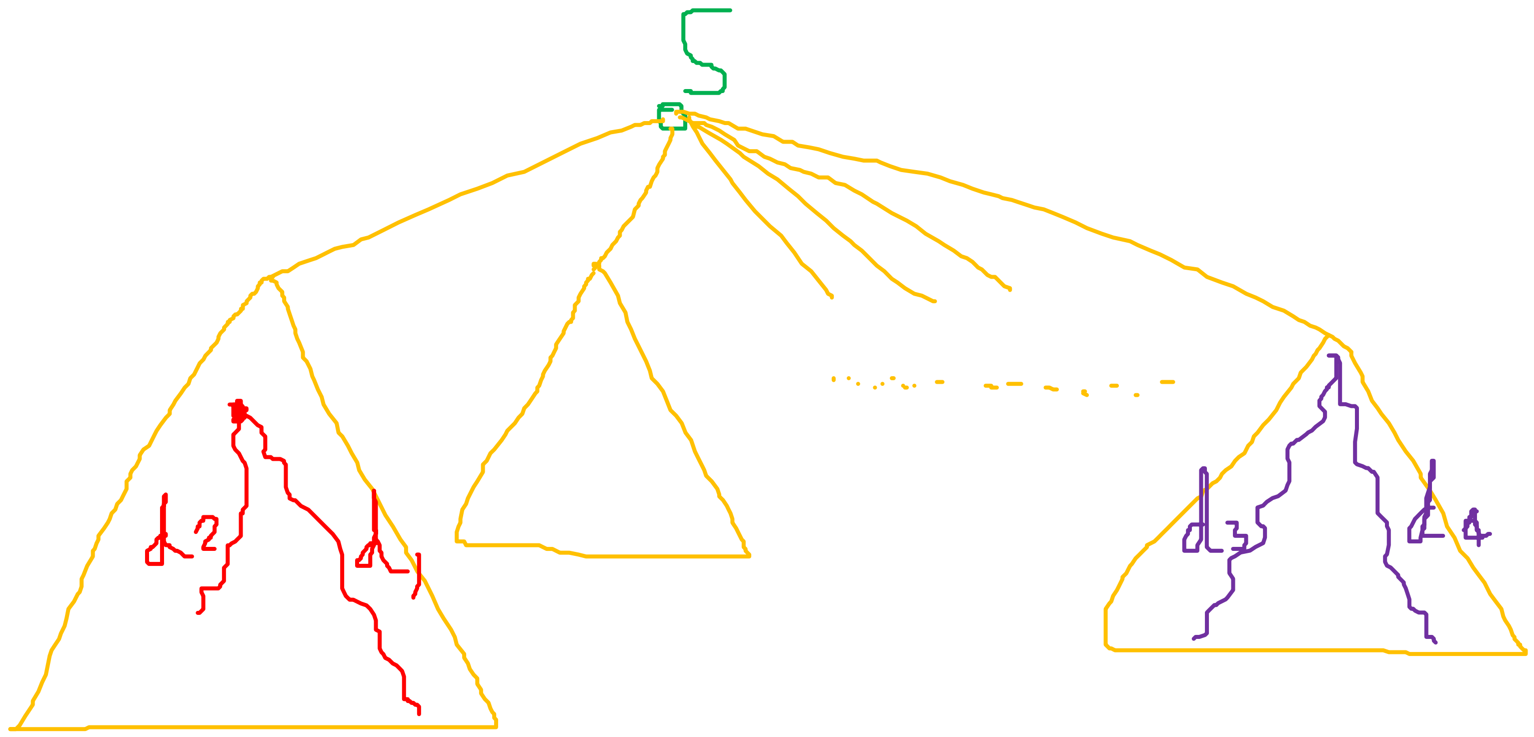
proof of correctness

case 2: ***s*** not on diameter



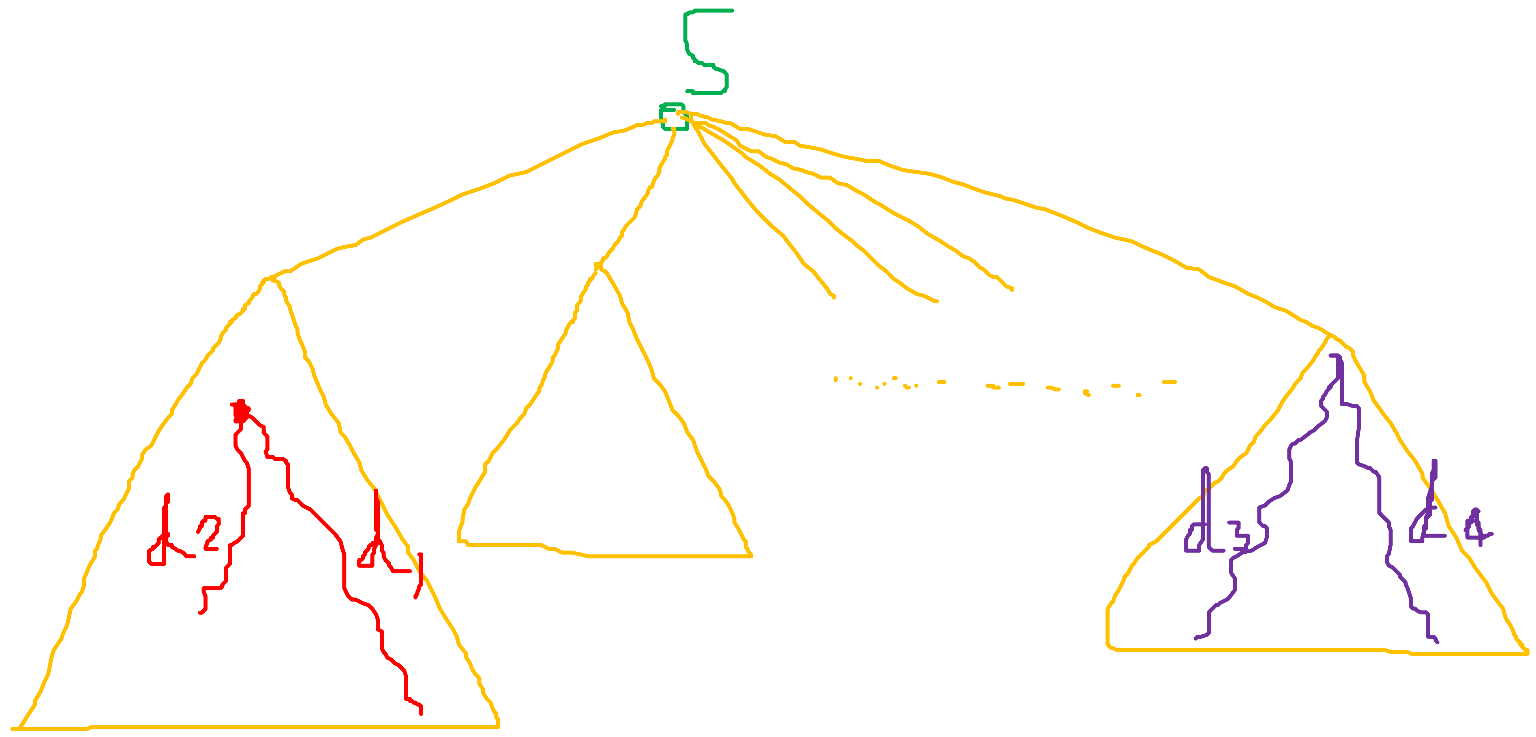
proof of correctness

case 2: ***s*** not on diameter



proof of correctness

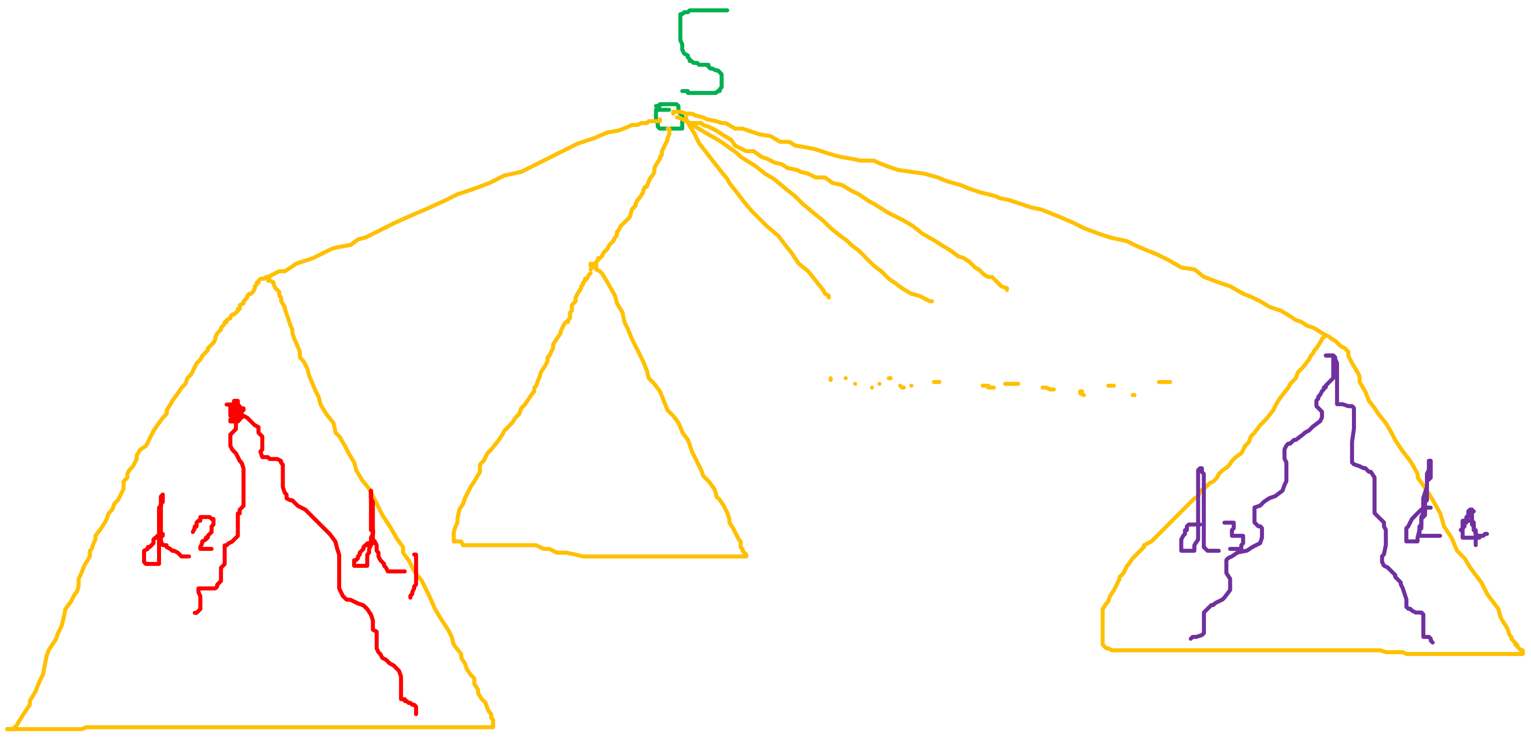
case 2: ***s*** <u>not</u> on diameter



$d_1 > d_3, d_4$

proof of correctness

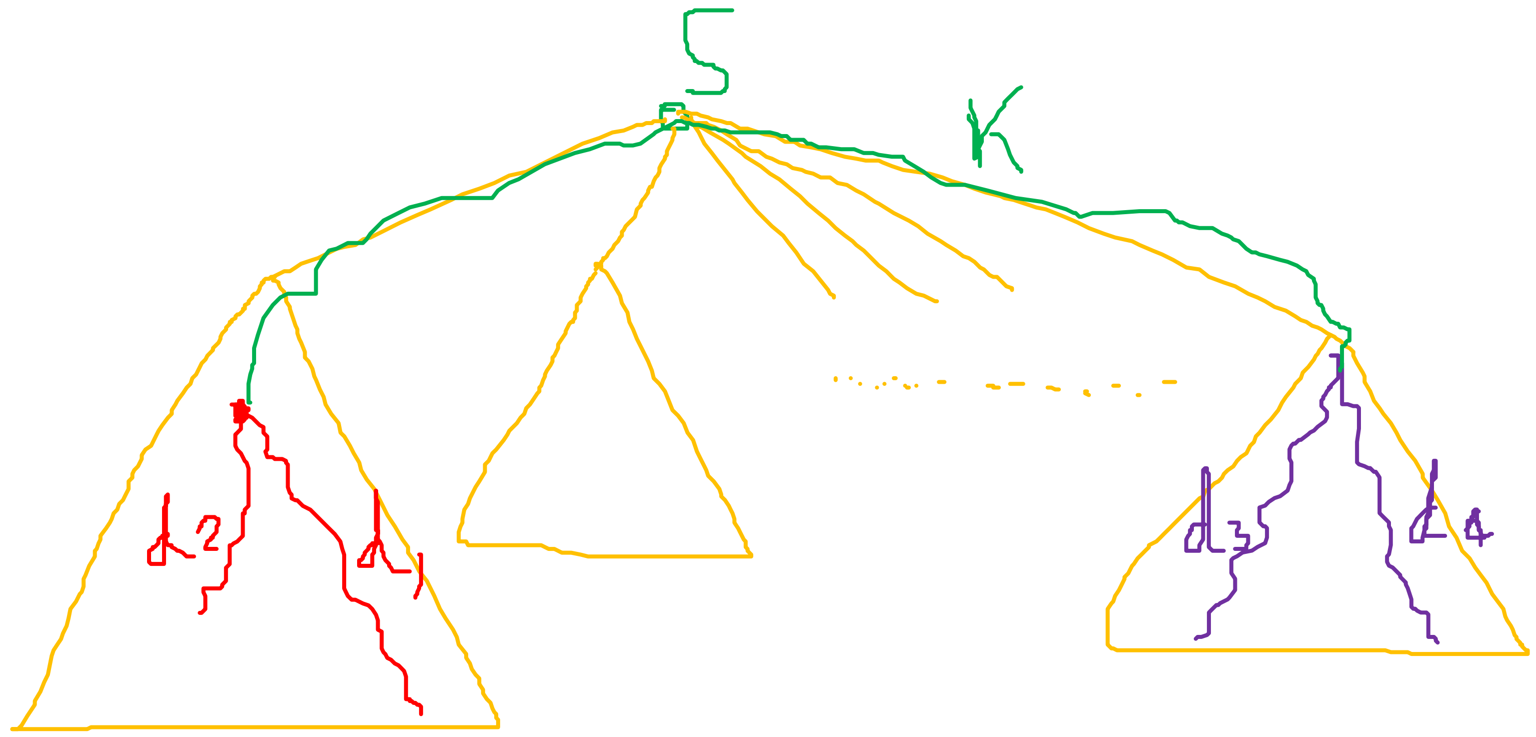
case 2: ***s*** <u>not</u> on diameter



$d_1 > d_3, d_4$ $d_3 + d_4 > d_1 + d_2$

proof of correctness

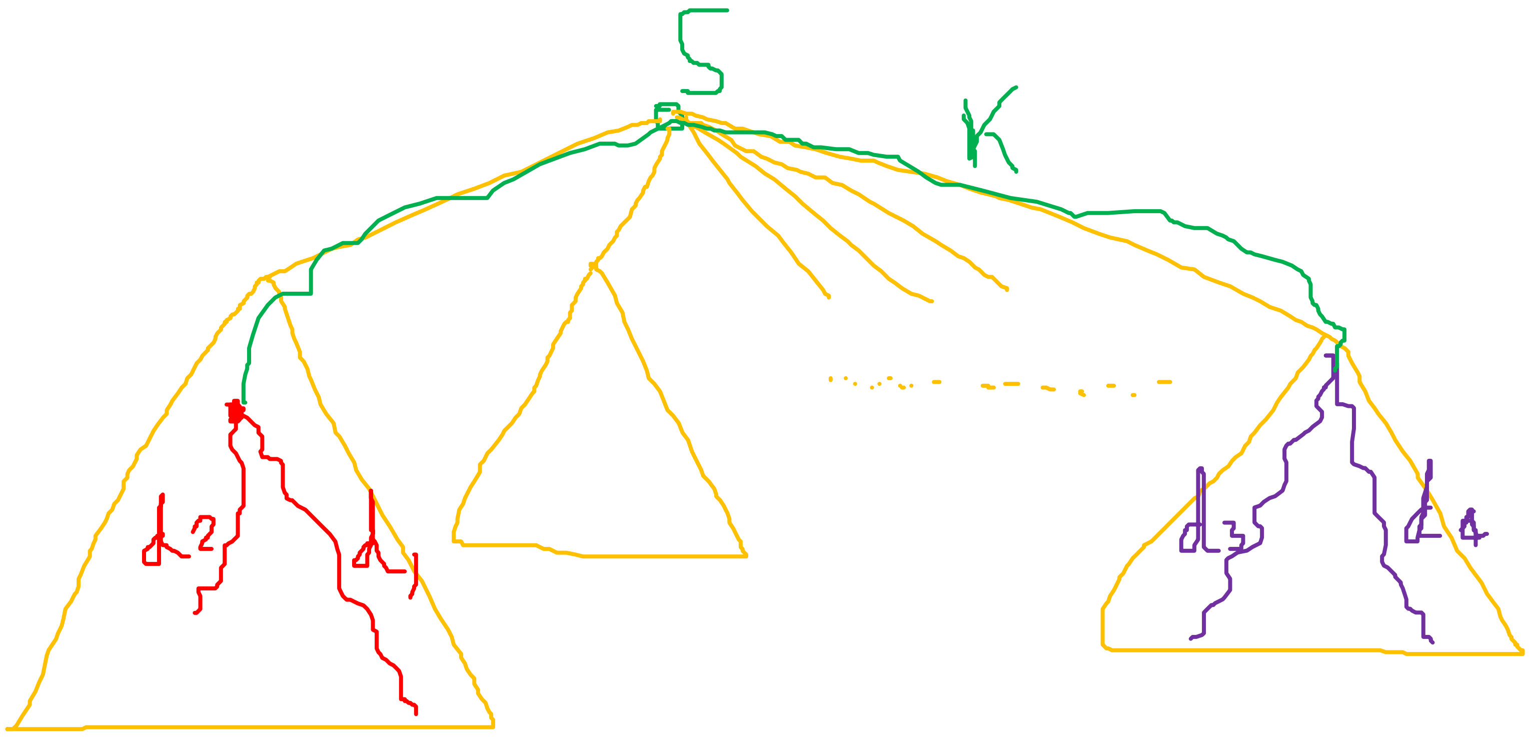
case 2: ***s*** <u>not</u> on diameter



$d_1 > d_3, d_4$ $\quad$ $d_1 + d_3 + K > d_3 + d_4 > d_1 + d_2$

proof of correctness

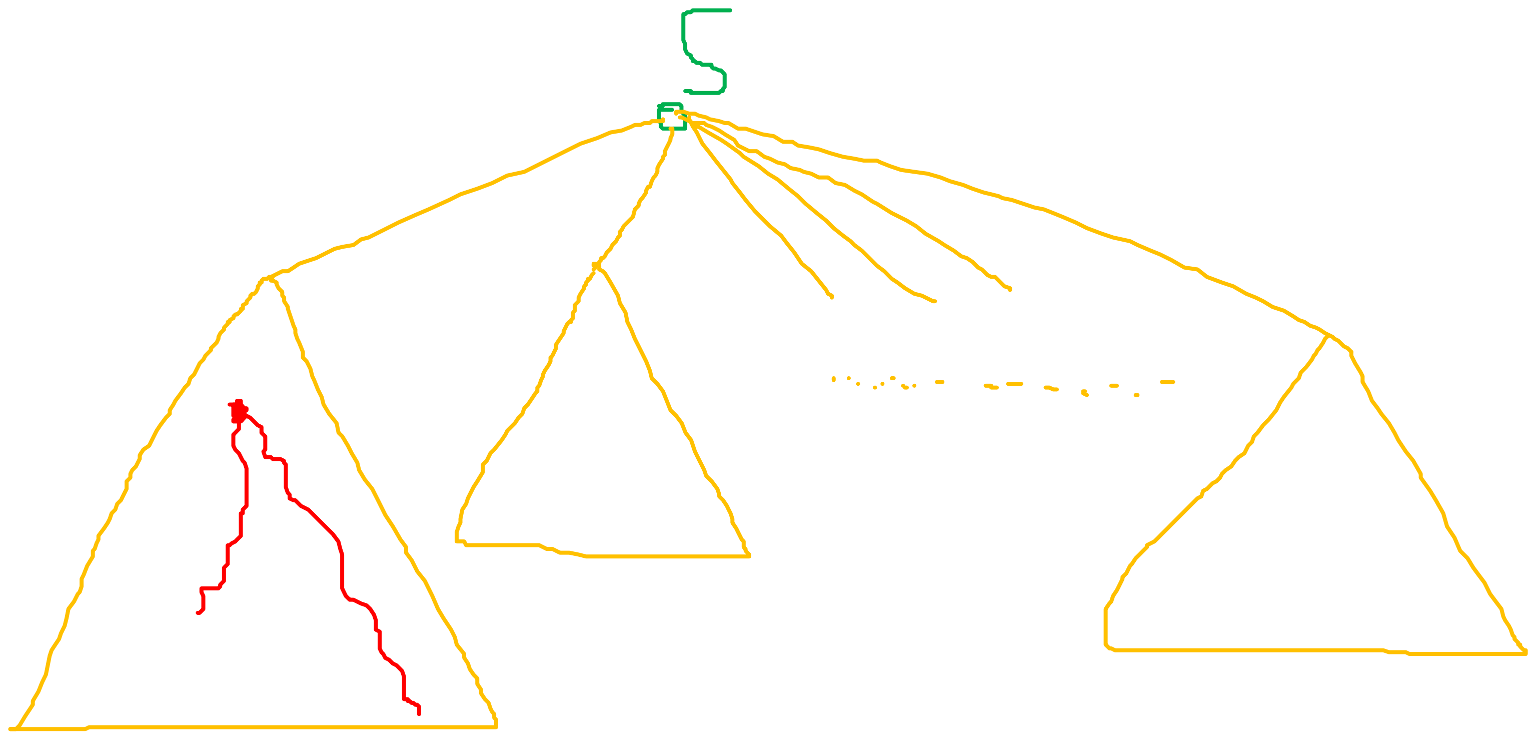
case 2: ***s*** not on diameter



$d_1 > d_3, d_4$ $\quad$ $d_1 + d_3 + K > d_3 + d_4 > d_1 + d_2$

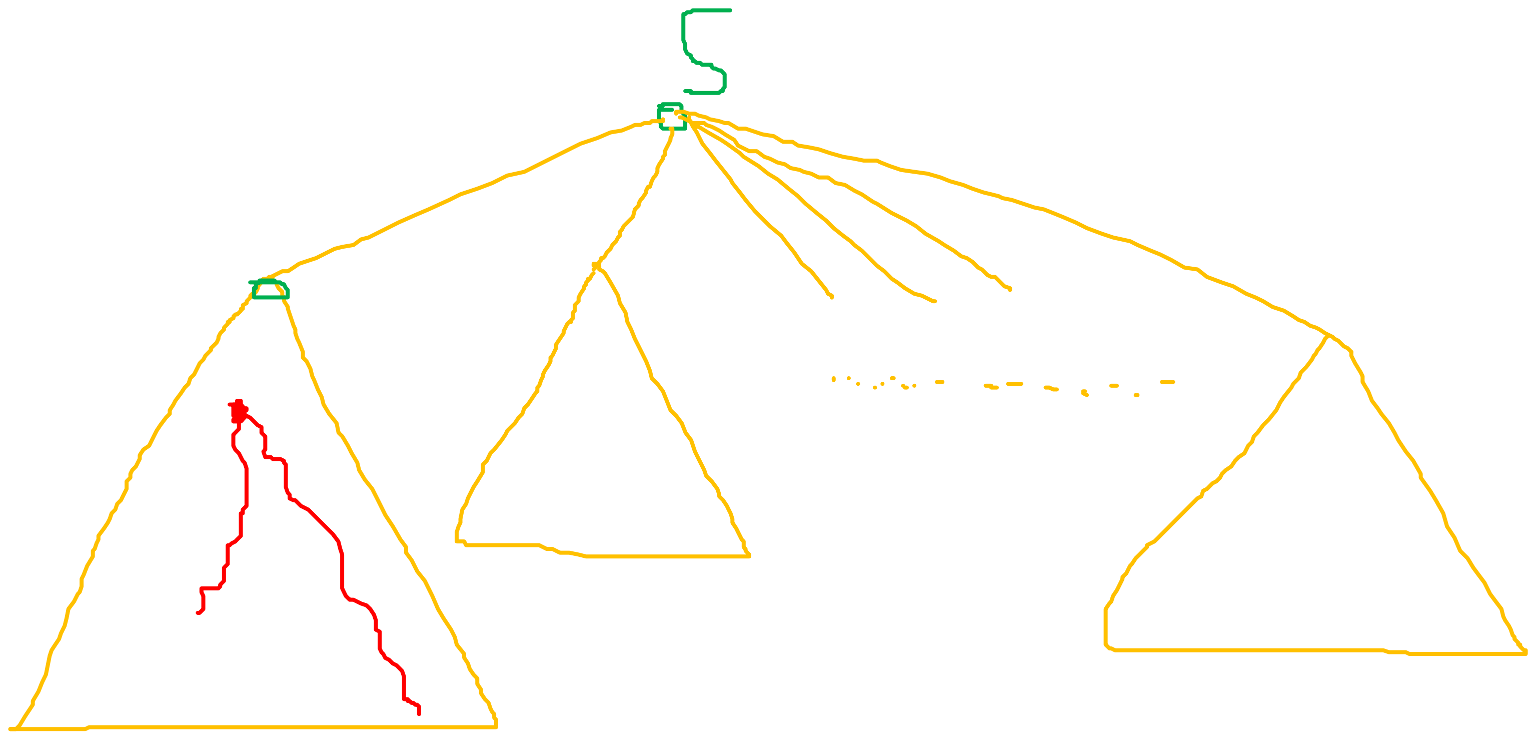
proof of correctness

case 2: ***s*** not on diameter



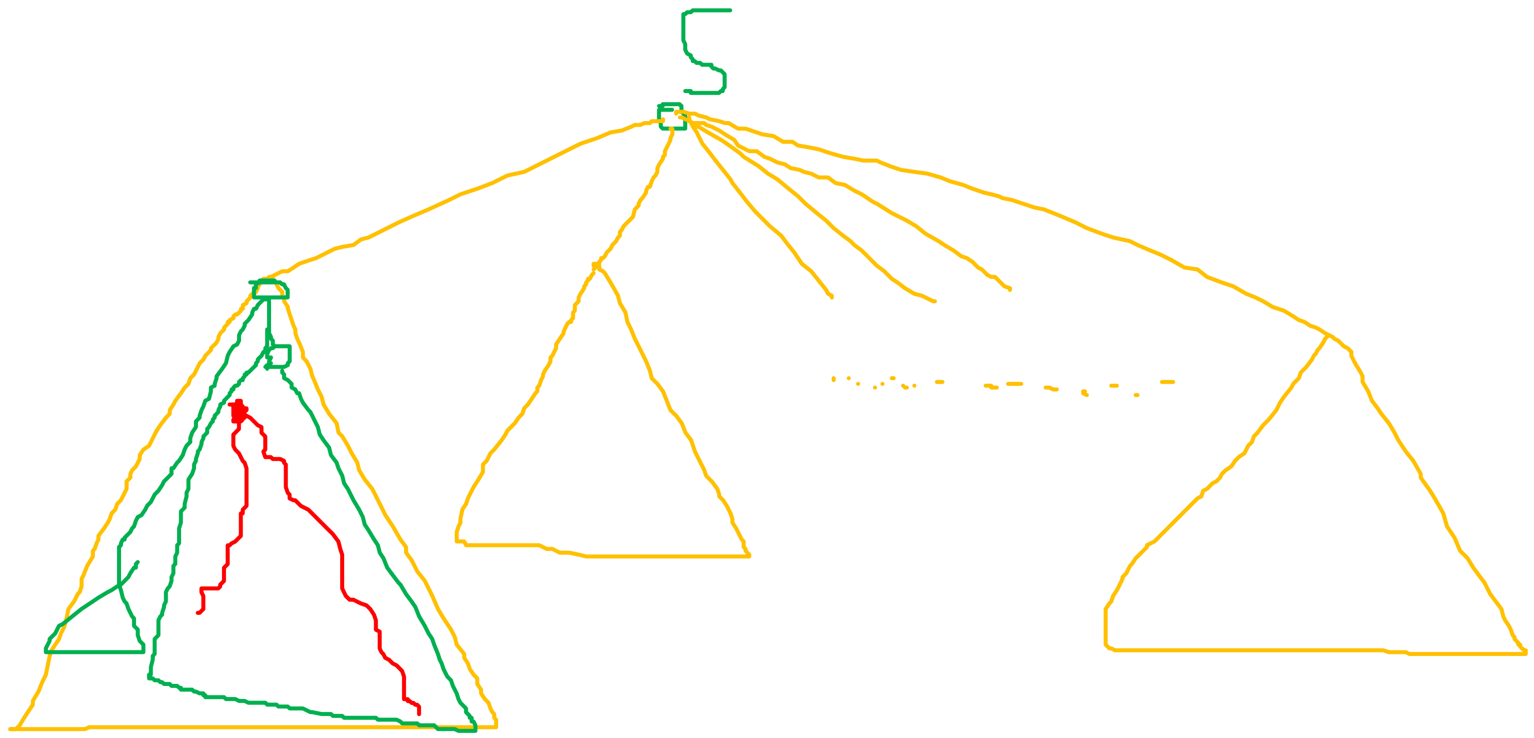
proof of correctness

case 2: ***s*** <u>not</u> on diameter



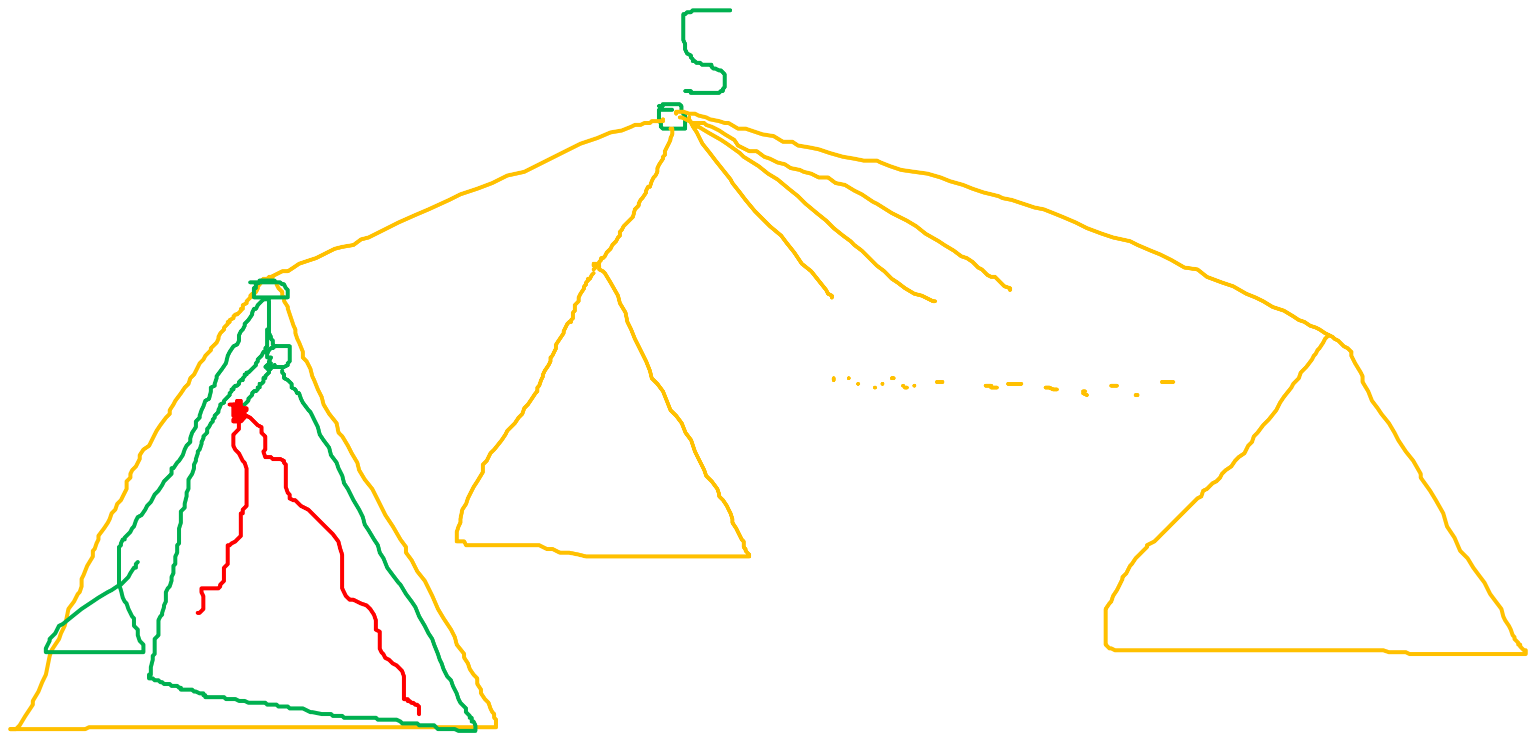
proof of correctness

case 2: ***s*** not on diameter



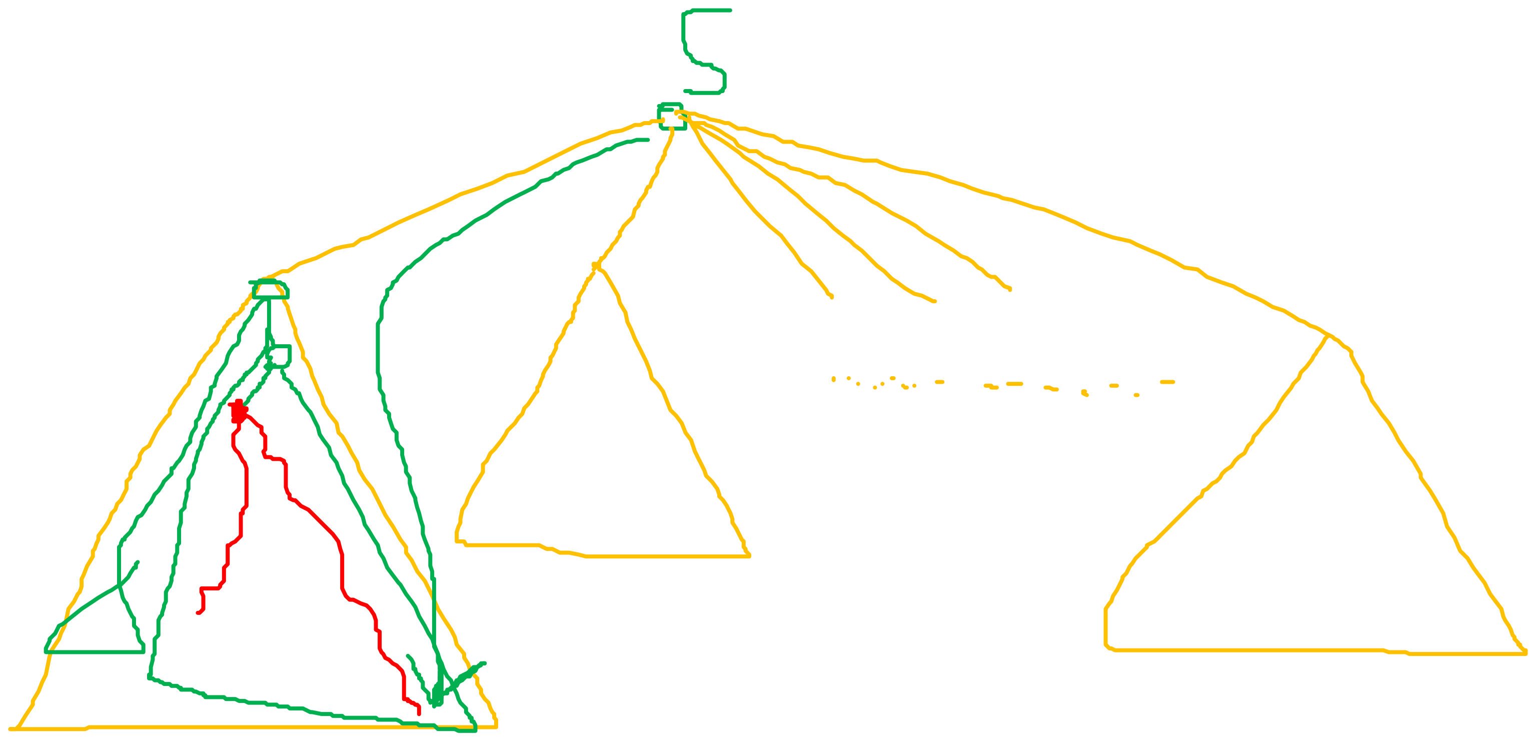
proof of correctness

case 2: ***s*** <u>not</u> on diameter



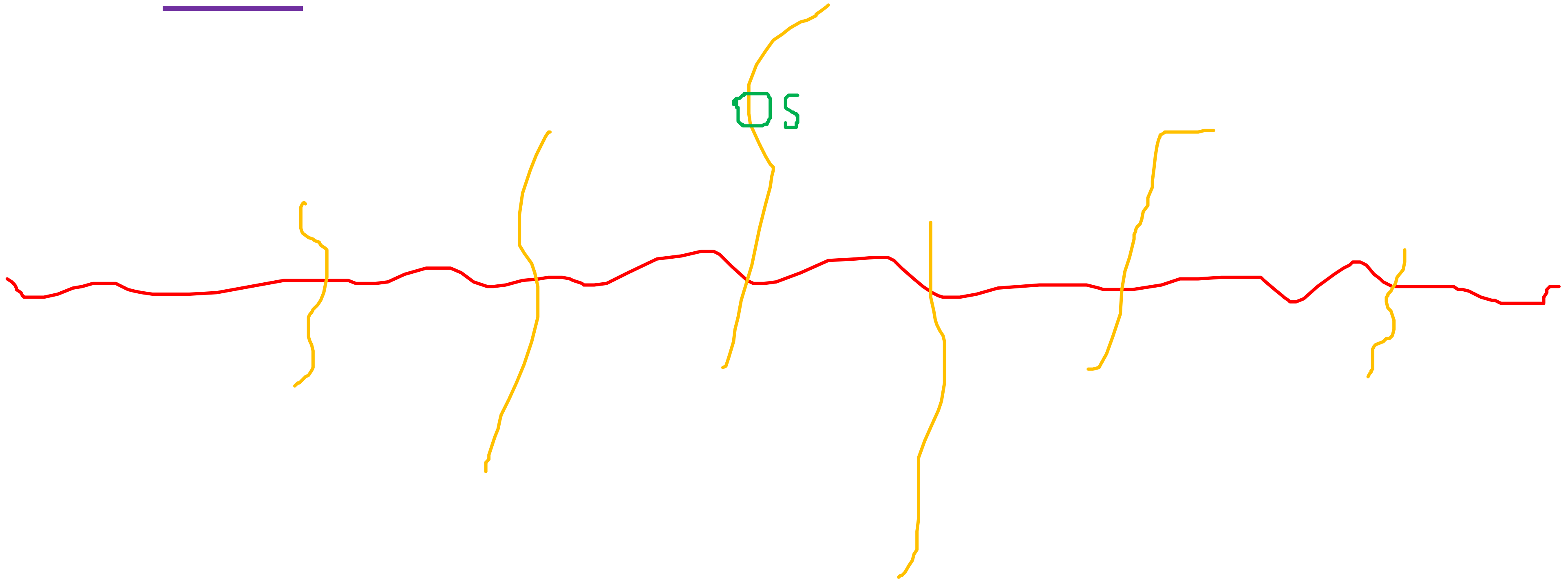
proof of correctness

case 2: ***s*** <u>not</u> on diameter



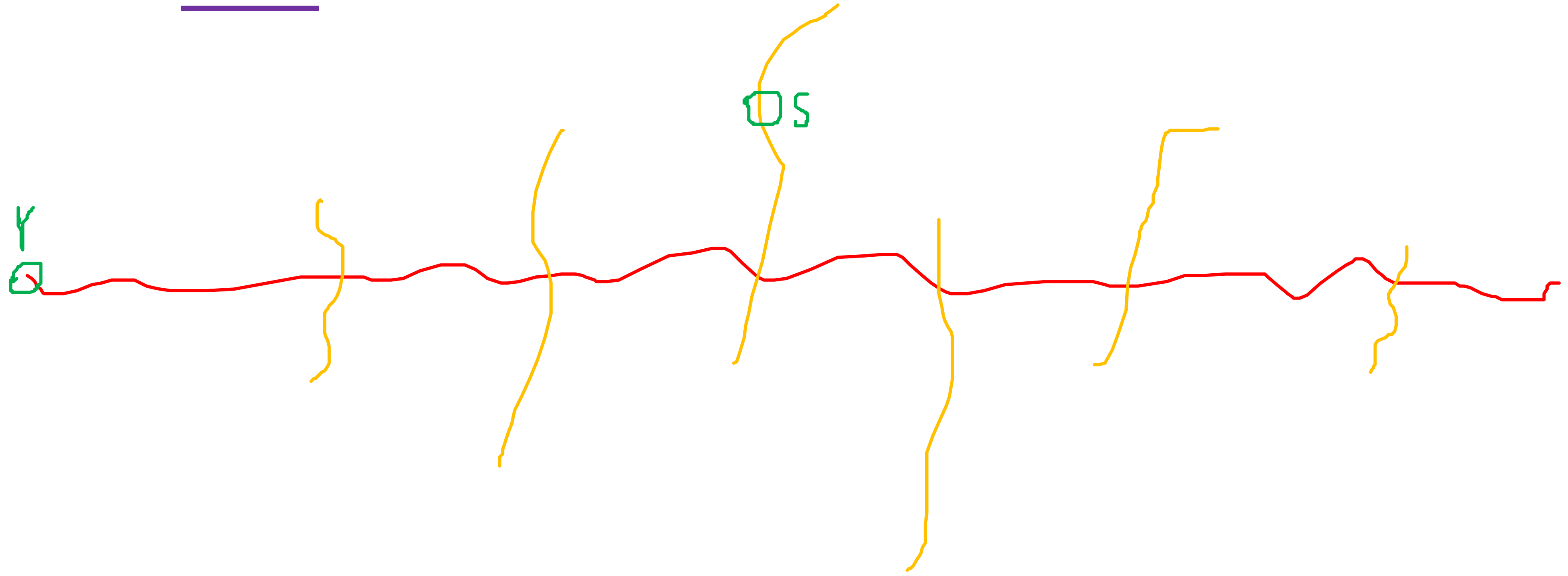
proof of correctness

case 2: ***s*** <u>not</u> on diameter



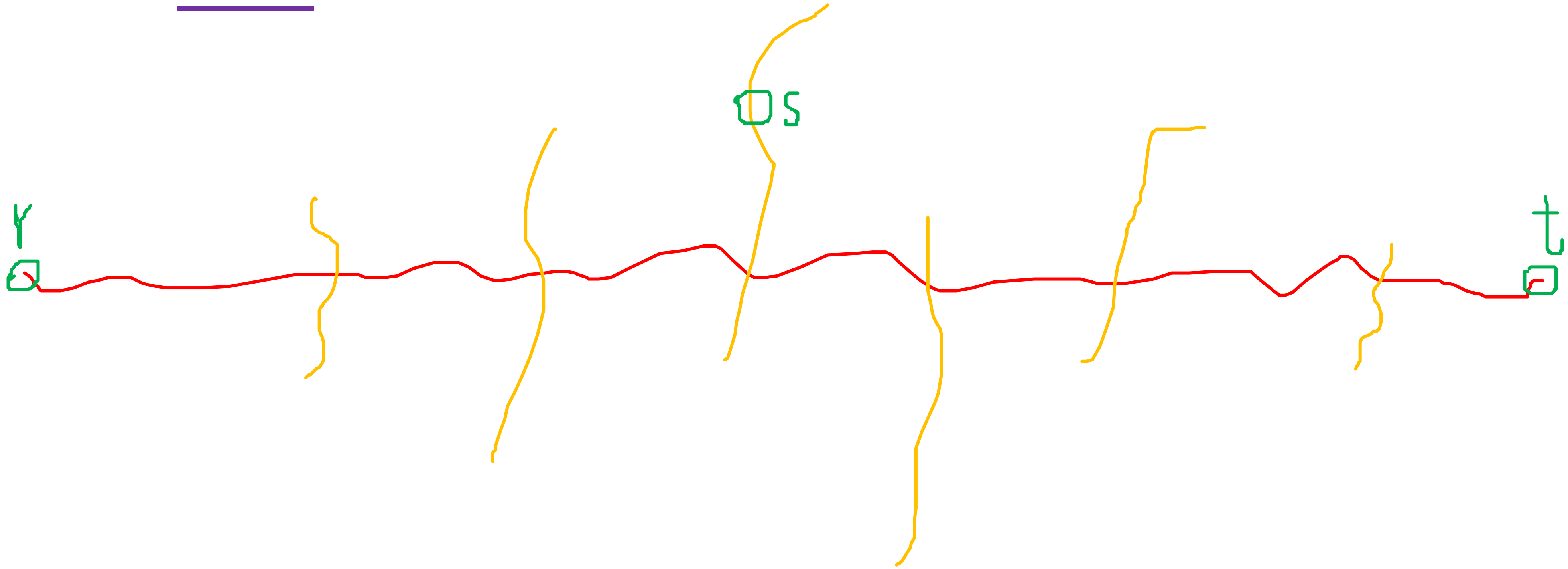
proof of correctness

case 2: ***s*** <u>not</u> on diameter



proof of correctness

case 2: ***s*** <u>not</u> on diameter



proof of correctness

picked s

- case 1: s on diameter
  proved ✓
- case 2: s not on diameter
  proved ✓

proof of correctness

picked *s*

case 1: *s* on diameter

proved ✓

case 2: *s* not on diameter

proved ✓

https://github.com/AmiriShavaki/Task-Fuel https://github.com/AmiriShavaki/Task-Fuel

Thank you!

💐💐💐