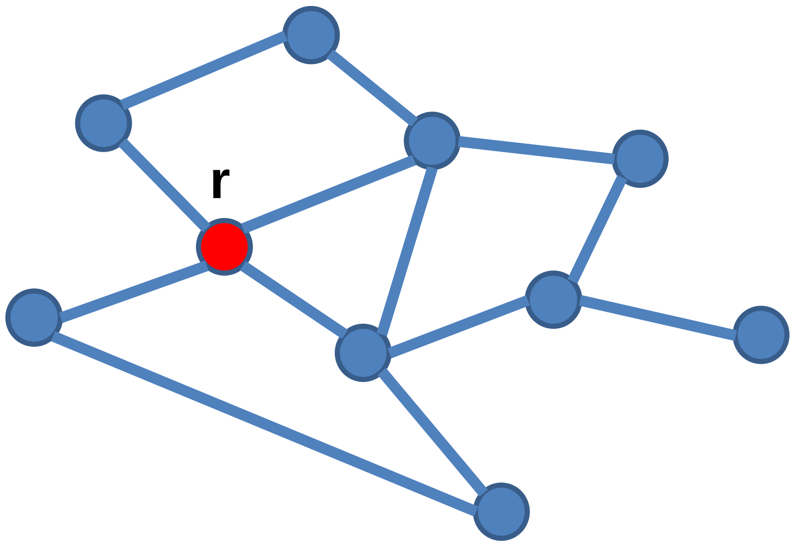
The First Fully Polynomial Stabilizing Algorithm for BFS Tree Construction*

A. Cournier, S. Rovedakis, V. Villain

* This work has been supported in part by the ANR project SPADES

Context

Undirected and connected network

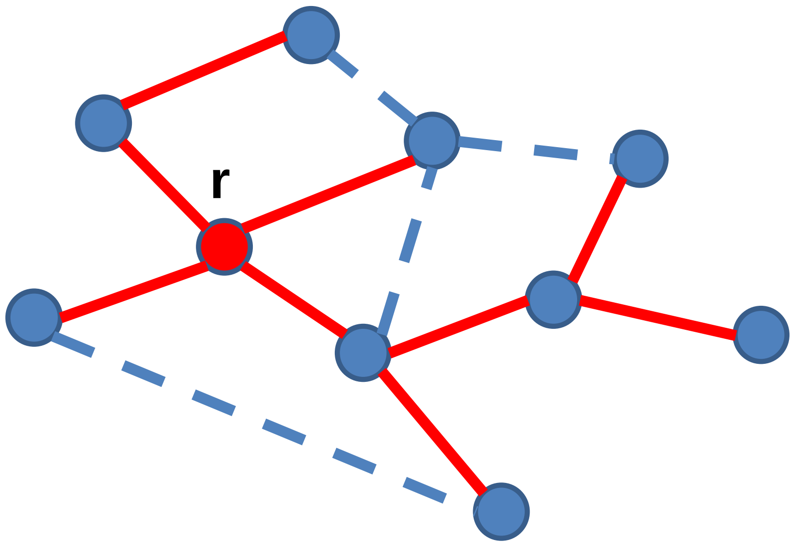


- Anonymous network with a distinguished root
- Each processor can distinguish its adjacent links

- Local shared memory
- Distributed **unfair** daemon

Context

Undirected and connected network



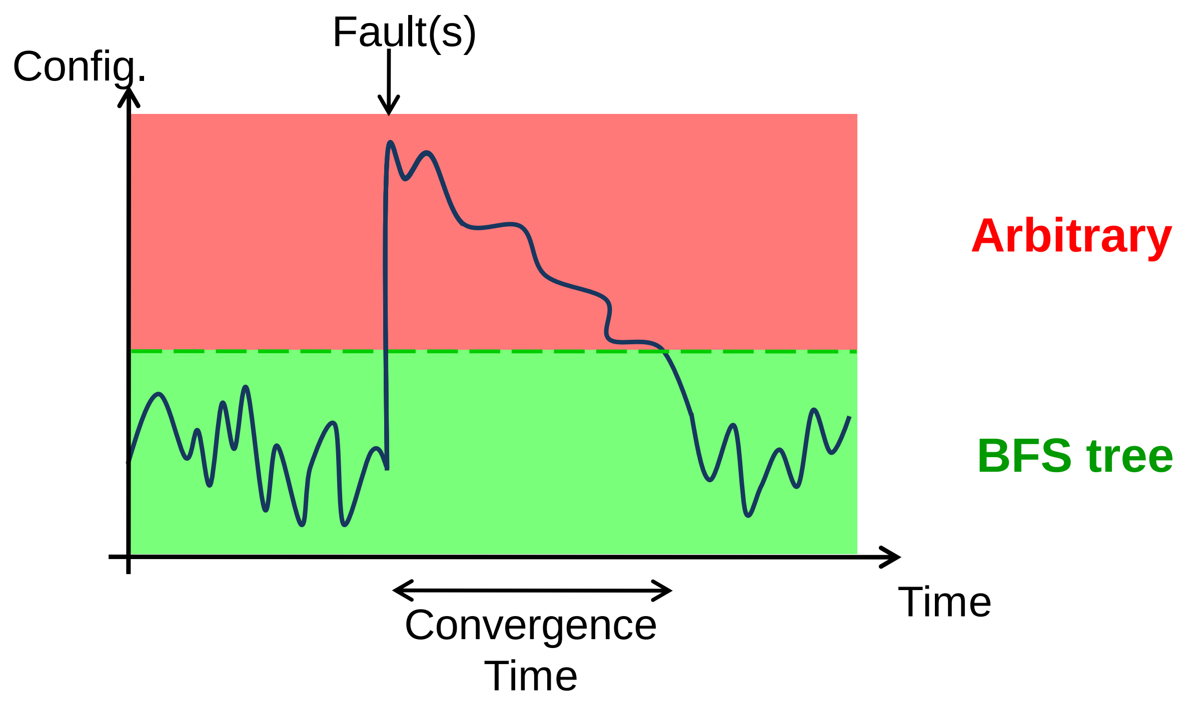
- Anonymous network with a distinguished root
- Each processor can distinguish its adjacent links

- Local shared memory
- Distributed **unfair** daemon

- Construction of a BFS tree

Stabilizing systems

- Self-stabilizing systems [Dijkstra, 1974]



- A snap-stabilizing system, regardless of the initial state of the processors, always behaves according to its specification. [Bui et al, 1999]

State of the art

Construction	Paper	Round	Step	Memory	Silent
BFS	[Arora, Gouda 90]	$O(N^2)$	-	$O(\log(n))$	Yes
	[Dolev, Israeli, Moran 90]	$O(d)$	-	$O(\Delta \log(n))$	Yes
	[Afek, Kutten, Yung 91]	$O(n^2)$	-	$O(\log(n))$	Yes
	[Ducourthial, Tixeuil 03]	$\Theta(d)$	$O(n(Max+d)^n)$	$O(\log(n))$	Yes
	[Awerbuch, Kutten, Mansour, Patt-Shamir, Varghese 93]	$O(D)$	$\Omega(2^{D/2})$	$O(\log^2(n))$	Yes
	[Johnen 97]	$\Omega(d^2)$	-	$O(\log(\Delta))$	No
	[Burman, Kutten 07]	$O(d)$	-	$O(\log^2(n))$	Yes
	[Datta, Larmore, Vemula]	$O(n)$	$\Omega(n^{\log(n)})$	$O(\log(n))$	Yes
	[Cournier, Devismes, Villain 09]	$\Theta(d^2+n)$	$O(\Delta n^3)$	$O(\log(n))$	No
Any	[Chen, Yu, Huang 91]	$O(n)$	$\Omega(2^n)$	$O(\log(n))$	Yes
	[Kosowski, Kuszner 05]	$O(n)$	$\Theta(n^2 d)$	$O(\log(n))$	Yes
	[Cournier 09]	$\Theta(n)$	$\Theta(n^2)$	$O(\log(n))$	Yes
DFS	[Collin, Dolev 94]	$O(dn \Delta)$	-	$O(n \log(\Delta))$	Yes
	[Cournier, Devismes, Villain 05]	$O(n^2)$	$O(n^3)$	$O(\log(n))$	Yes
	[Cournier, Devismes, Petit, Villain 06]	$O(n)$	$O(n^2)$	$O(n \log(n))$	Yes
	[Cournier, Devismes, Villain 09]	$O(n)$	$O(\Delta n^3)$	$O(\log(\Delta+n))$	No

Motivation

Definition [*fully polynomial algorithm*]: It is a stabilizing algorithm with

- a round complexity $O(d^a)$ and,
- a step complexity $O(n^b)$

with d the diameter and n the size of the network

Question:

Does there exists a *fully polynomial* stabilizing algorithm to construct a spanning tree ?

- Polynomial step complexity
- with a round complexity $O(d^a)$

Existing approaches

<u>Goal:</u>

Construction of a spanning tree satisfying fully polynomial constraints

- **Approach 1 [Huang, Chen 92]**
  - Classical strategy to construct a a spanning tree
    - Root has a zero level
    - Each processor hooks on to its neighbor of lowest level
- **Problems**
  - The valid tree grows **quickly** in terms of **rounds**
  - Invalid trees are deleted **slowly**
    - **New processors can join invalid trees**

Existing approaches

Goal:

Construction of a spanning tree satisfying fully polynomial constraints

- **Approach 2 ([KK 05] and [Cournier 09])**
  - Classical strategy to construct a a spanning tree
    - Root has a zero level
    - Each processor hooks on to its neighbor of lowest level
  - Invalid trees are frozen
- **Problems**
  - Invalid trees are deleted **quickly** (*no processor can join*)
  - The valid tree grows **slowly** (*wait for tree deletions*)

Goal

- **A forest** (*constraint on levels*)
  - **A valid tree**
  - **Invalid trees**

- **Duality**
  - The valid tree must grow quickly (in terms of rounds and steps)
  - Invalid trees must be deleted quickly too

⇨ Need to develop a mechanism to deal with this duality (*question-answer mechanism*)

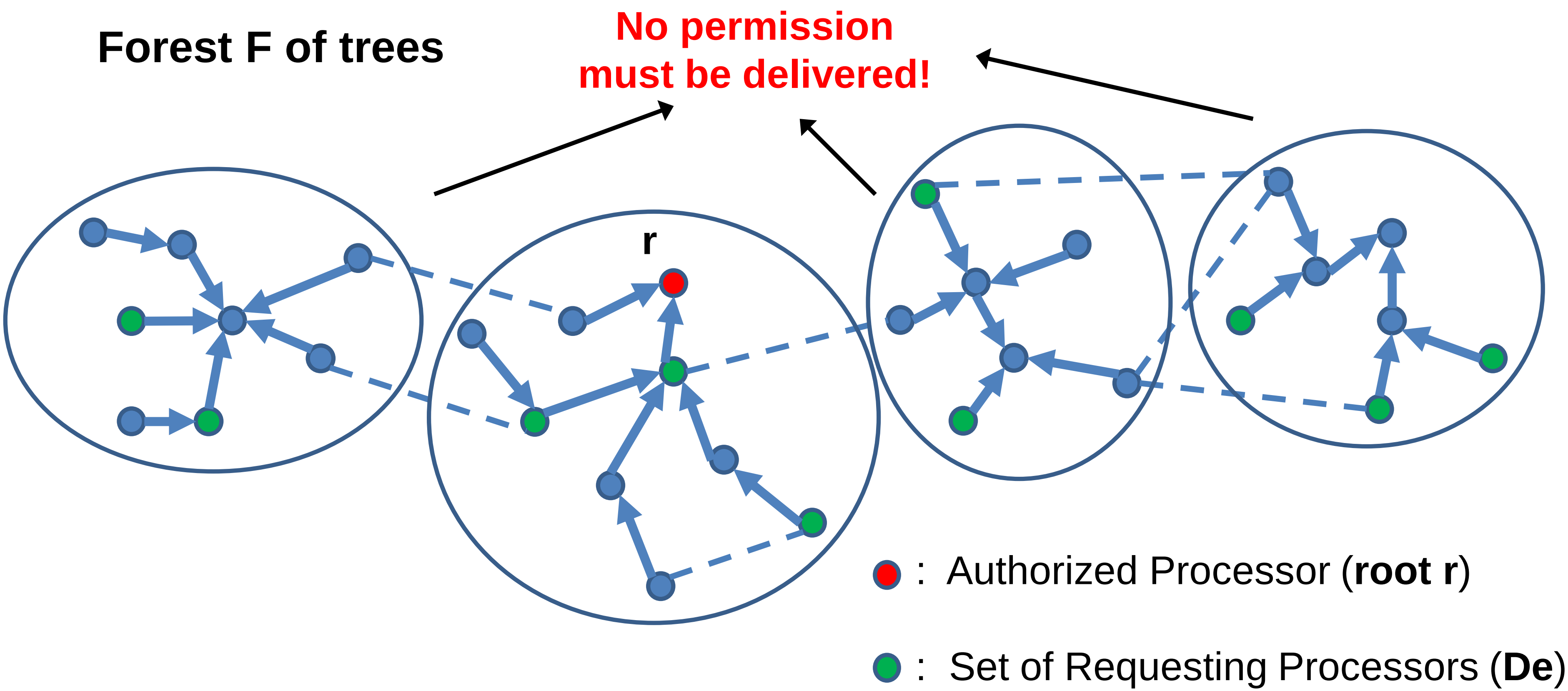
Our contribution

Construction	Paper	Round	Step	Memory	Silent
BFS	[Arora, Gouda 90]	$O(N^2)$	-	$O(\log(n))$	Yes
	[Dolev, Israeli, Moran 90]	$O(d)$	-	$O(\Delta \log(n))$	Yes
	[Afek, Kutten, Yung 91]	$O(n^2)$	-	$O(\log(n))$	Yes
	[Ducourthial, Tixeuil 03]	$\Theta(d)$	$O(n(Max+d)^n)$	$O(\log(n))$	Yes
	[Awerbuch, Kutten, Mansour, Patt-Shamir, Varghese 93]	$O(D)$	$\Omega(2^{D/2})$	$O(\log^2(n))$	Yes
	[Johnen 97]	$\Omega(d^2)$	-	$O(\log(\Delta))$	No
	[Burman, Kutten 07]	$O(d)$	-	$O(\log^2(n))$	Yes
	[Datta, Larmore, Vemula 08]	$O(n)$	-	$O(\log(n))$	Yes
	[Cournier, Devismes, Villain 09]	$\Theta(d^2+n)$	$O(\Delta n^3)$	$O(\log(n))$	No
	[Cournier, Rovedakis, Villain 19]	$O(d^2)$	$O(n^6)$	$O(\log(n))$	Yes
Any	[Chen, Yu, Huang 91]	$O(n)$	$\Omega(2^n)$	$O(\log(n))$	Yes
	[Kosowski, Kuszner 05]	$O(n)$	$\Theta(n^2 d)$	$O(\log(n))$	Yes
	[Cournier 09]	$\Theta(n)$	$\Theta(n^2)$	$O(\log(n))$	Yes
DFS	[Collin, Dolev 94]	$O(dn \Delta)$	-	$O(n \log(\Delta))$	Yes
	[Cournier, Devismes, Villain 05]	$O(n^2)$	$O(n^3)$	$O(\log(n))$	Yes
	[Cournier, Devismes, Petit, Villain 06]	$O(n)$	$O(n^2)$	$O(n \log(n))$	Yes
	[Cournier, Devismes, Villain 09]	$O(n)$	$O(\Delta n^3)$	$O(\log(\Delta+n))$	No

Stabilizing BFS algorithm

(Question-Answer problem)

Question-Answer Problem



Allowed tree: tree rooted at a processor **r**

Question-Answer Problem:

Deliver a permission to a *requesting* processor in an *allowed* tree

Question-Answer Algorithm

Satisfies

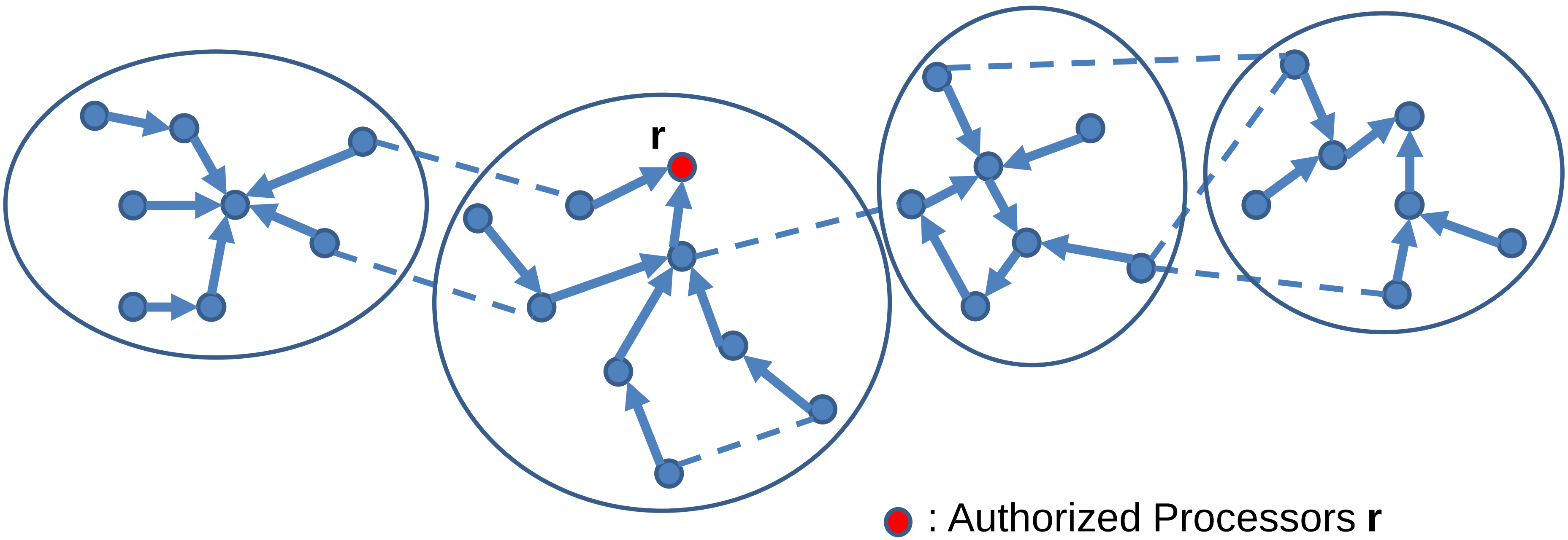
1. Deliver a permission **only** to requesting processors in allowed tree
2. If the closest requesting processor(s) of the allowed tree is at height ***k***, it will receive a permission in 2***k*** rounds
3. And in polynomial number of steps

Stabilizing BFS algorithm

(Spanning tree construction)

Tree construction problem

A set of connected components F



Allowed tree: tree rooted at a processor **r**

⇨ **Only Processor r in AP** (*Spanning tree*)

Tree construction

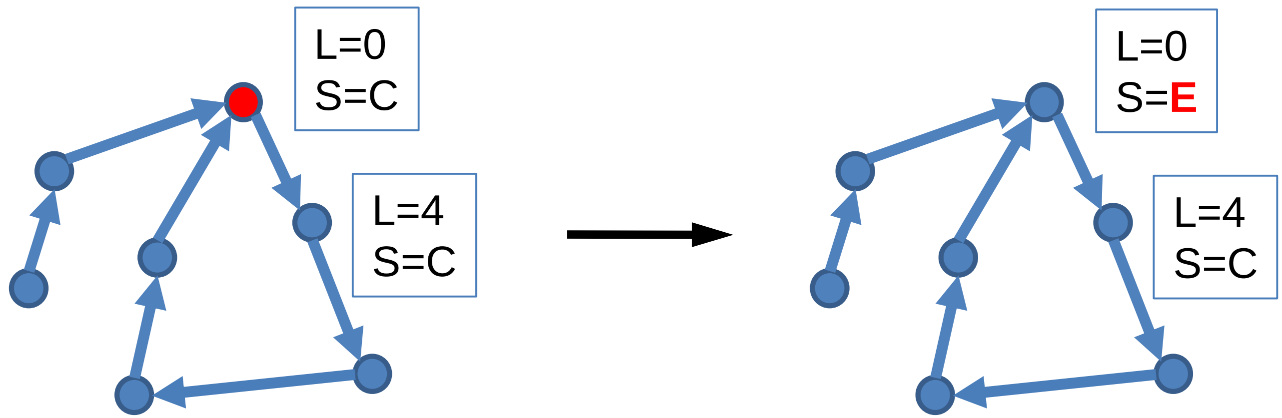
- **Normal root (**processor **r)**
  - No parent
  - Zero level for r ($r.L = 0$)
  - Status to C
- **Abnormal root**
  - x has a parent
  - Faulty level ($x.L \leq (x.P).L$)

⇨ **Forest** (constraint on levels, i.e., $p.L=(p.P).L+1$))

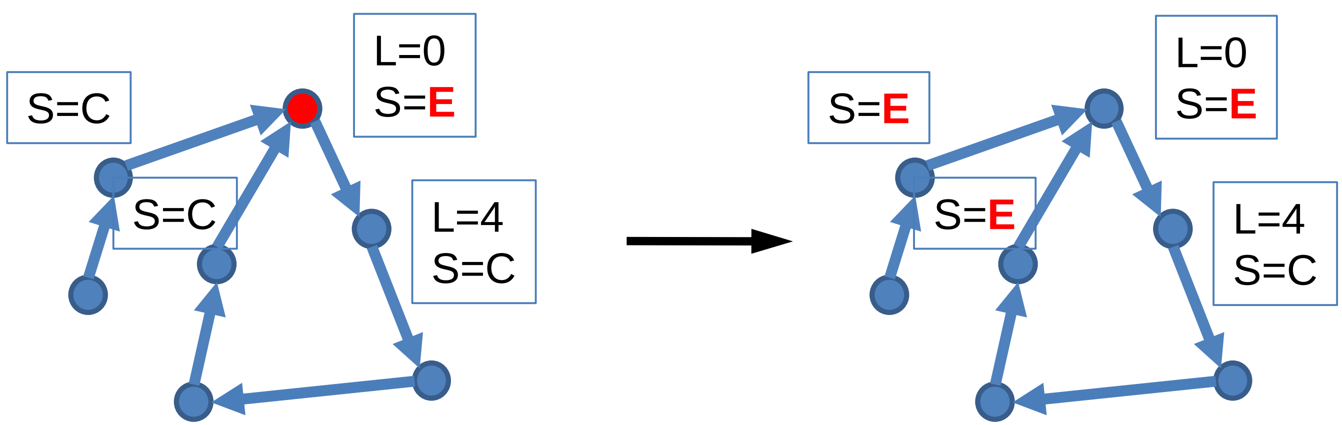
- **Normal tree** ⇨ tree rooted at **r**
- **Abnormal tree** ⇨ tree rooted at an abnormal root

Tree construction

- Detection of an abnormal root

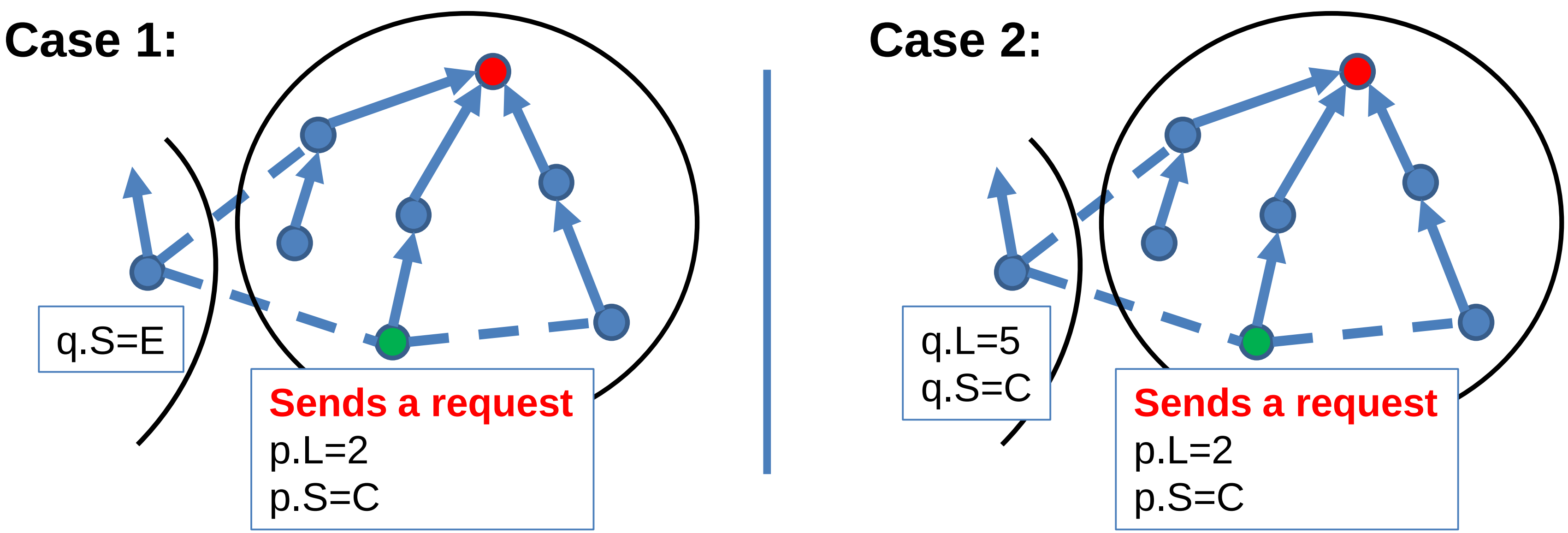


- Propagation of Status E in abnormal trees



Tree construction

- **Permission:** allows connections of new processors to a normal tree
- Ask of a permission
  - **Case 1:** a neighbor q in an abnormal tree (*q in Status E*)
  - **Case 2:** a neighbor q with high level (*q.L > p.L+1*)



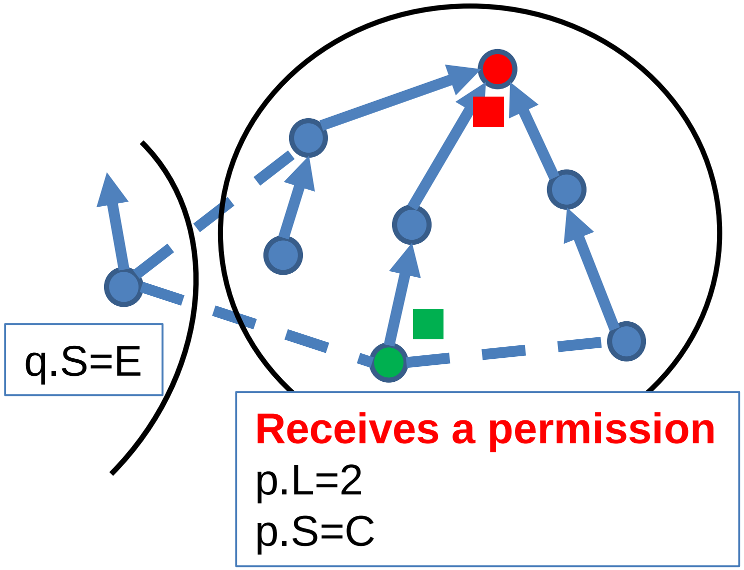
Tree construction

- **Construction of a BFS**

  Connection to the neighbor:

  – of lowest level

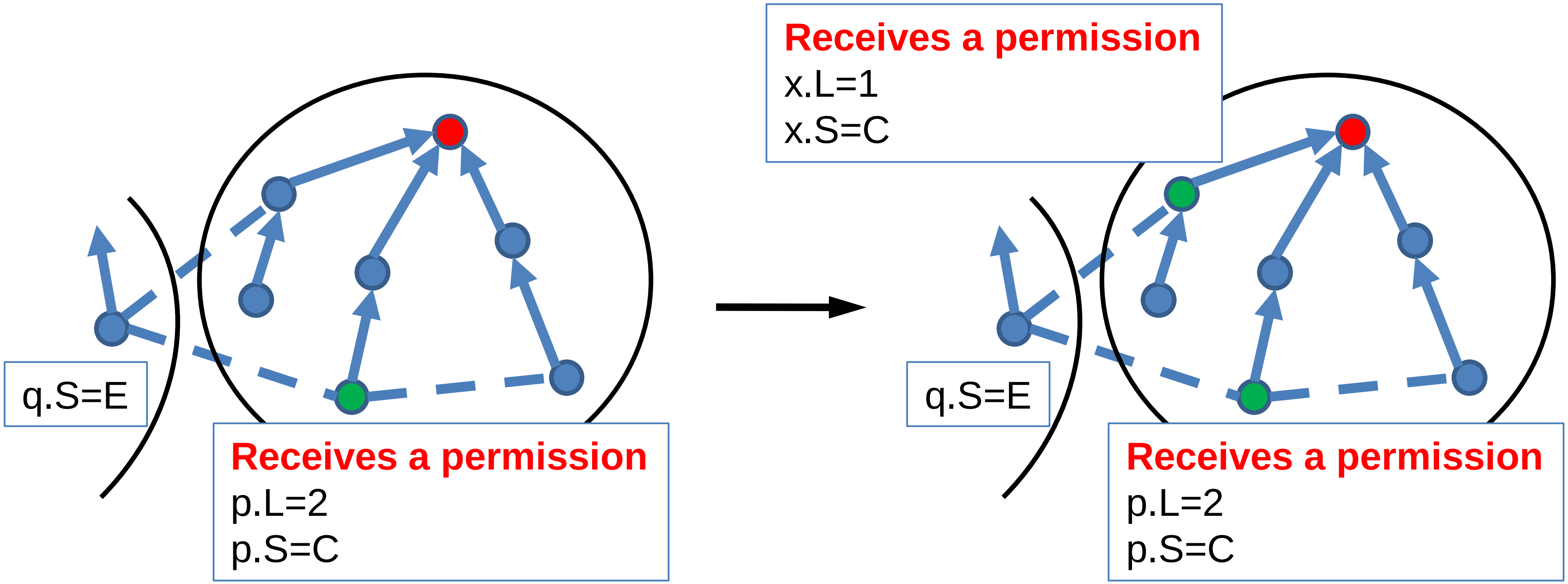
  – with a permission (*obtained via Algorithm 2*)



Tree construction

- **Construction of a BFS**

  Connection to the neighbor:
  - of lowest level
  - with a permission (*obtained via Algorithm 2*)



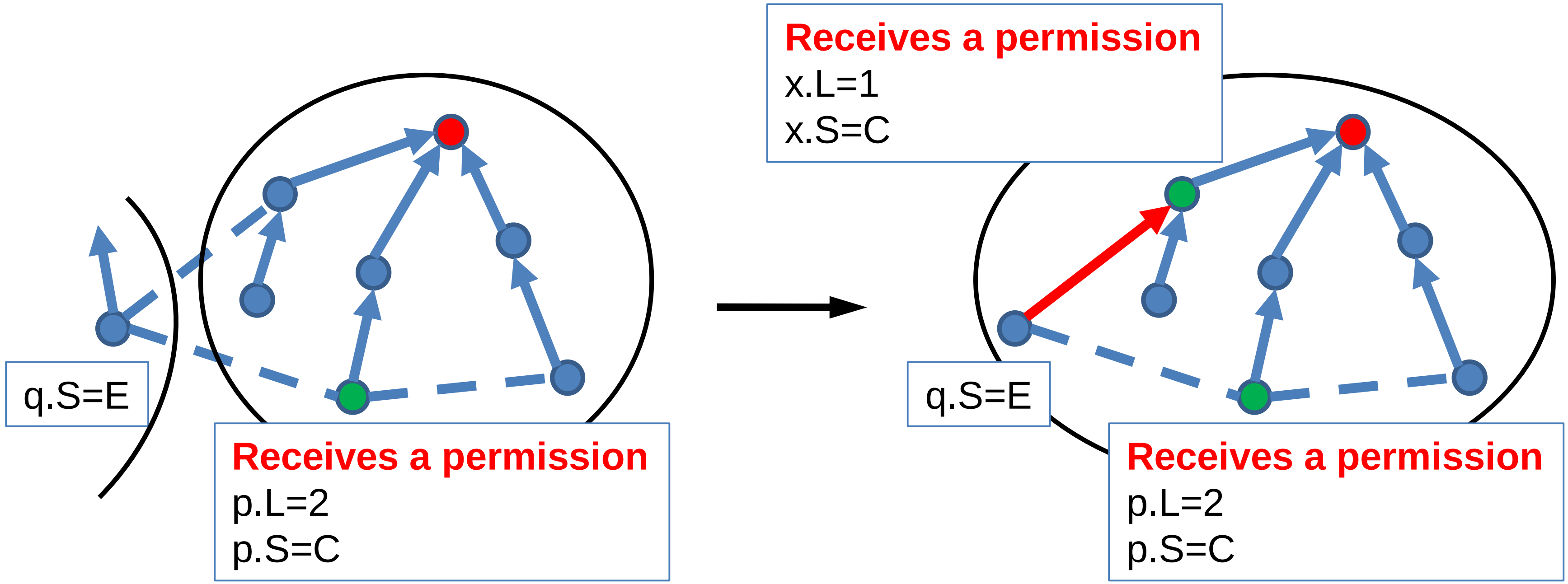
Tree construction

- **Construction of a BFS**

  Connection to the neighbor:

  – of lowest level

  – with a permission (*obtained via Algorithm 2*)



Complexity analysis

Rounds complexity

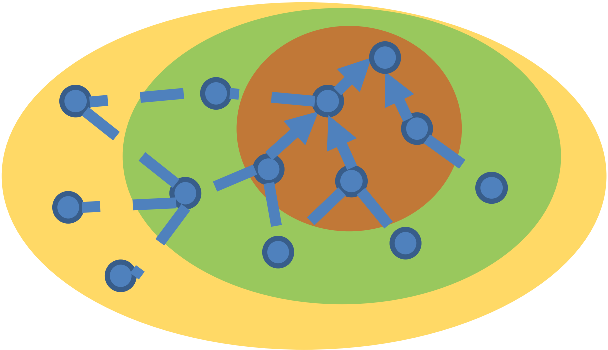
- Our algorithm constructs a BFS tree layer by layer, each new layer uses permissions
  - Processors closest to r at height $\boldsymbol{k}$, receive a permission in O($\boldsymbol{k}$) rounds (*Algorithm 2*)
  - There are at most $\boldsymbol{d}$ layers in a BFS tree

  ⇒ O($\boldsymbol{d}^2$) rounds to construct a BFS tree
  - Processors in abnormal trees connect to BFS tree in O($\boldsymbol{d}^2$) (*do not wait the end of propagation*)

⇒ In O($\boldsymbol{d}^2$) rounds a BFS tree is constructed

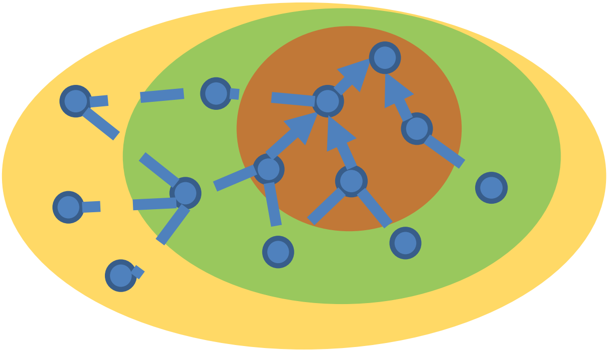
Step complexity

- Topological change:
  - Status changes from C to E
  - Connection to a new parent
- Only neighbors of an abnormal tree can join it (*permission is needed*)
  - At most 1 connection via the same neighbor



Step complexity

- Topological change:
  - Status changes from C to E
  - Connection to a new parent
- Only neighbors of an abnormal tree can join it (*permission is needed*)
  - At most 1 connection via the same neighbor



- Each processor produces at most $2\Delta+n$ topological changes
  - $< 2\Delta$ topological changes (while not in normal tree)
  - $< n$ topological changes (while in normal tree)

Step complexity

As a consequence the time step complexity is

$O(\Delta m n^3 + m n^4) \leq O(n^6)$ steps

Conclusion

- Silent stabilizing algorithm to construct BFS tree in $O(d^2)$ rounds and $O(n^6)$ steps
  - Best comprise between round and step complexities
  - Questioning mechanism **reduces step complexity** (avoids useless requests), but **higher round complexity**

Perspectives

- Does there exist a stabilizing algorithm to construct a spanning tree in **O(*d*)** rounds with a polynomial step complexity ?
- Determine the problems which admit a fully polynomial algorithm

Thank you

Stabilizing BFS construction

- **Conditional composition of Algorithm 1 and 2**

- **Legitimate configuration (*Algorithm 1*)**

  For every p:
  - p.L=(p.P).L+1
  - For every neighbor q, |p.L-q.L| ≤ 1

  ⇨ No new request is generated (**Rem1**)

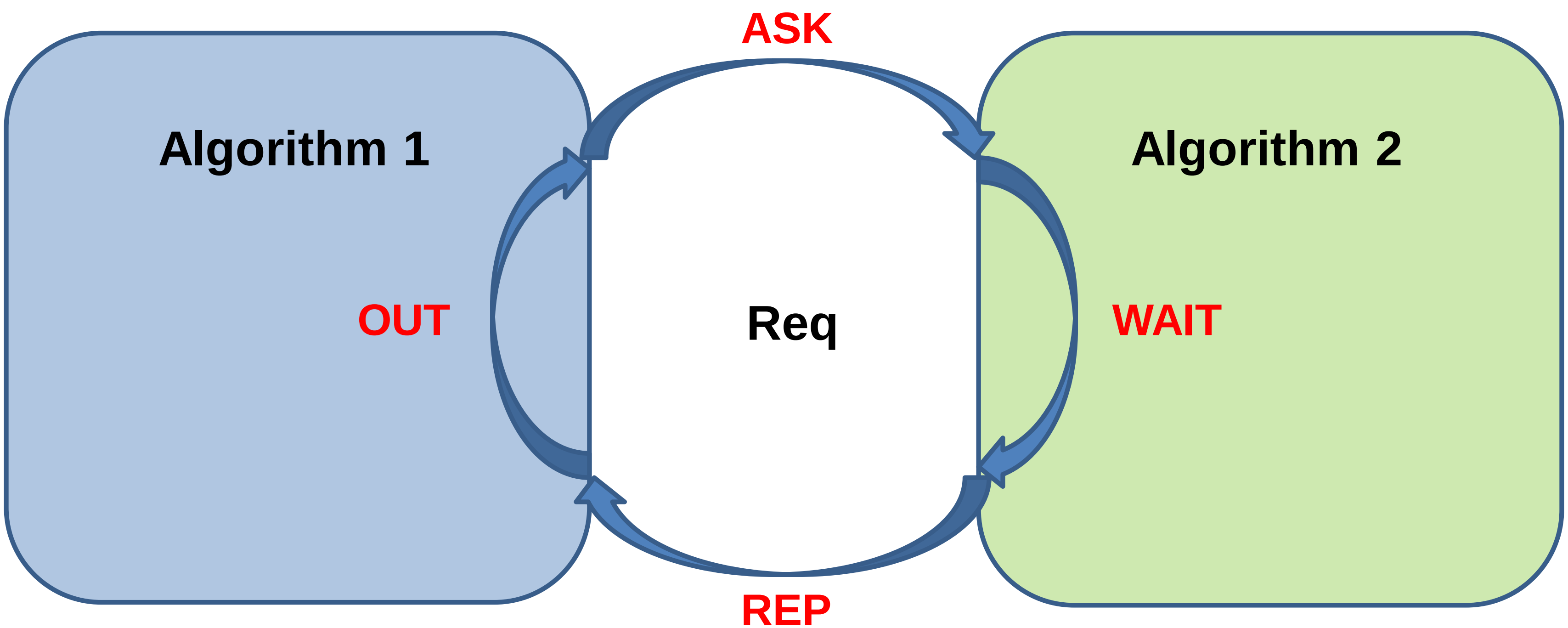
- **Silent Property**
  - Algorithm 1 is silent (*levels are correct, no new connection*)
  - Algorithm 2 is silent (*from Rem1*)

Stabilizing BFS tree algorithm

- Stabilizing BFS tree algorithm obtained by composition of Algorithm 1 and 2
- Use of a conditional composition (*use of a Predicate*)
  - Algorithm 2 executed before Algorithm 1
  - Algorithm 2 executed at p iff
    - p belongs to a normal tree
    - p belongs to a tree locally correct
    - p has lower height than its neighbor q

Algorithm outline

- Conditional composition of 2 stabilizing algorithms



Handles processor attachments

Delivers authorizations

- Algorithm 1 monitors Algorithm 2 using variable Req

Step complexity: Algorithm 2

- Each topological change produces at most Δ requests

(1) At most 2Δm+mn requests to construct a BFS tree
- – Since there are $2\Delta n+n^2$ topological changes (*Algorithm 1*)

(2) In $O(n^3)$ steps every requesting processor receives a permission
- – At least a requesting processor (nearest to r) receive a permission in $O(n^2)$ steps

Final step complexity

$$O(2\Delta m+mn) \times O(n^3) = O(\Delta mn^3+mn^4) \leq O(n^6) \text{ steps}$$