Distributedly Testing Cycle-Freeness

Heger Arfaoui[1] Pierre Fraigniaud[1]
David Ilcinkas[2] Fabien Mathieu[3]

[1]CNRS & Université Paris Diderot (LIAFA)
[2]CNRS & Université de Bordeaux (LaBRI)
[3]Alcatel-Lucent Bell Labs (LINCS)

GT Algorithmique Distribuée
January 12, 2015

Framework

Objective

Monitoring properties in large-scale distributed networks:

- Some nodes (possibly all) are queried
- Queried nodes execute a $O(1)$-round local distributed algorithm, producing a (small) local output
- A central authority gathers these local outputs and takes a global decision

Contexts of use

- Sensor networks with a base station
- Complexity theory in distributed computing

Framework

Objective

Monitoring properties in large-scale distributed networks:

- Some nodes (possibly all) are queried
- Queried nodes execute a $O(1)$-round local distributed algorithm, producing a (small) local output
- A central authority gathers these local outputs and takes a global decision

Contexts of use

- Sensor networks with a base station
- Complexity theory in distributed computing

Framework

Objective

Monitoring properties in large-scale distributed networks:

- Some nodes (possibly all) are queried
- Queried nodes execute a $O(1)$-round local distributed algorithm, producing a (small) local output
- A central authority gathers these local outputs and takes a global decision

Contexts of use

- Sensor networks with a base station
- Complexity theory in distributed computing

Framework

Objective

Monitoring properties in large-scale distributed networks:

- Some nodes (possibly all) are queried
- Queried nodes execute a $O(1)$-round local distributed algorithm, producing a (small) local output
- A central authority gathers these local outputs and takes a global decision

Contexts of use

- Sensor networks with a base station
- Complexity theory in distributed computing

Framework

Objective

Monitoring properties in large-scale distributed networks:

- Some nodes (possibly all) are queried
- Queried nodes execute a $O(1)$-round local distributed algorithm, producing a (small) local output
- A central authority gathers these local outputs and takes a global decision

Contexts of use

- Sensor networks with a base station
- Complexity theory in distributed computing

Property / Distributed language

We consider:

Properties

- Graph properties: large expansion, cycle-freeness
- Properties on labels: existence of a unique leader
- Mixed properties: $(\Delta + 1)$-coloring, existence of a spanning tree

More formally, distributed languages

A distributed language L is a set of labeled graphs (G, ℓ)

- G is a connected undirected graph
- $\ell : V(G) \to \{0,1\}^*$ is a function that labels each node v with the label $\ell(v)$.

Property / Distributed language

We consider:

Properties

- Graph properties: large expansion, cycle-freeness
- Properties on labels: existence of a unique leader
- Mixed properties: $(\Delta + 1)$-coloring, existence of a spanning tree

More formally, distributed languages

A distributed language L is a set of labeled graphs (G, ℓ)

- G is a connected undirected graph
- $\ell : V(G) \to \{0,1\}^*$ is a function that labels each node v with the label $\ell(v)$.

Precisions on the model

Local algorithms

Use of the $\mathcal{LOCAL}$ model:

- Distinct IDs
- Synchronous rounds, simultaneous start
- No failures, no dynamicity
- No congestion: unbounded message to each neighbor

Central algorithm

The central authority

- receives as input the multiset of the local outputs computed by the nodes
- runs a deterministic algorithm.

Precisions on the model

Local algorithms

Use of the $\mathcal{LOCAL}$ model:

- Distinct IDs
- Synchronous rounds, simultaneous start
- No failures, no dynamicity
- No congestion: unbounded message to each neighbor

Central algorithm

The central authority

- receives as input the multiset of the local outputs computed by the nodes
- runs a deterministic algorithm.

Types of decision

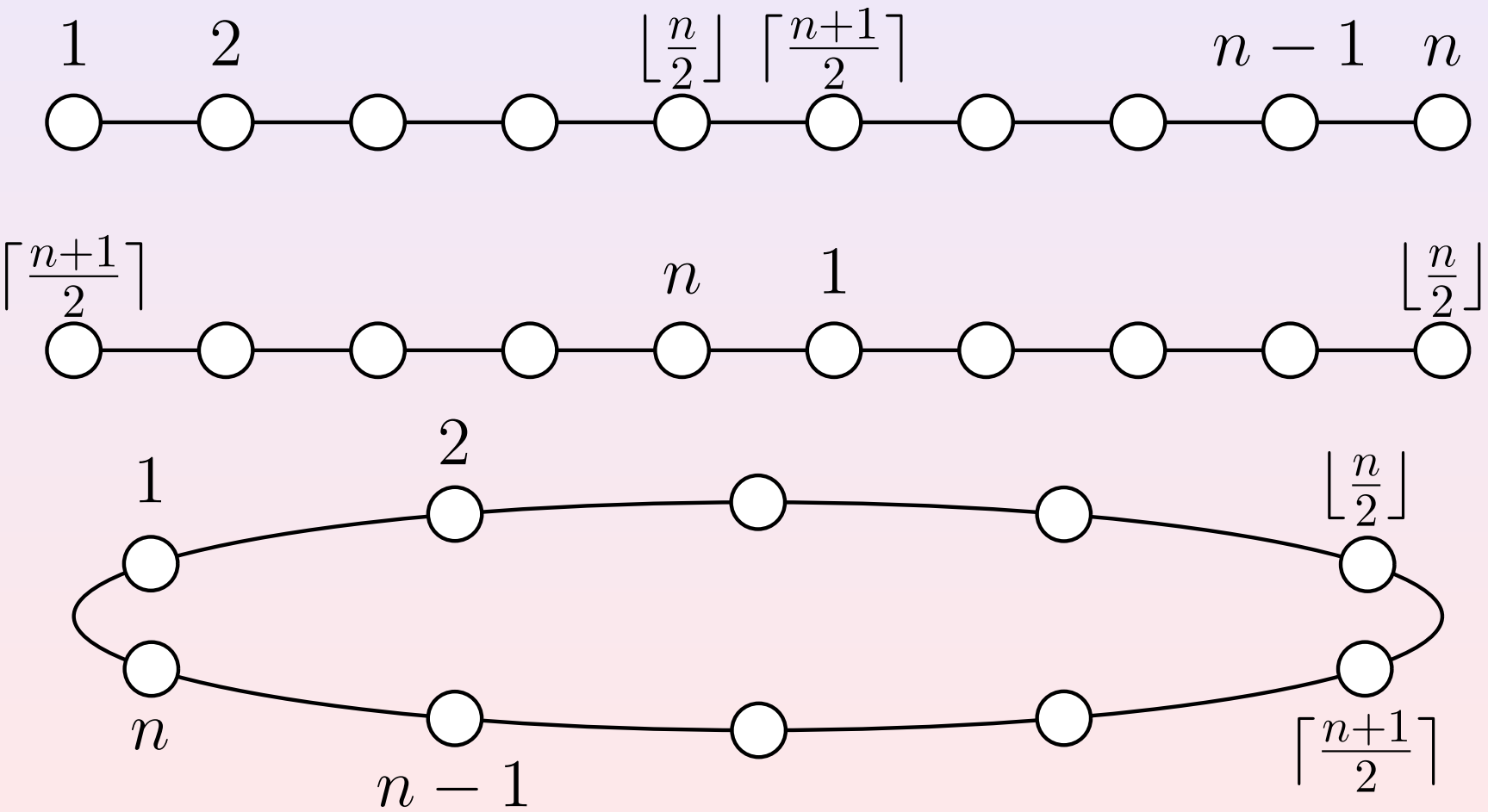
	#queried nodes	type of output	decision mechanism	gap	error	certificates
property testing	$o(n)$	$O(\log n)$ bits	algorithm	ϵ-far	yes	no
dist. decision	n	yes/no	$\forall$ yes / $\exists$ no	none	no	no
dist. testing	n	$O(\log n)$ bits	algorithm	none	no	no
dist. verification	n	yes/no	$\forall$ yes / $\exists$ no	none	no	yes
dist. certification	n	$O(\log n)$ bits	algorithm	none	no	yes

Distributed decision (cycle-freeness)

	#queried nodes	type of output	decision mechanism	success probability	certificates
dist. decision	n	yes/no	$\forall$ yes / $\exists$ no	impossible	no

Distributed decision (cycle-freeness)

	#queried nodes	type of output	decision mechanism	success probability	certificates
dist. decision	n	yes/no	$\forall$ yes / $\exists$ no	impossible	no



Distributed testing (cycle-freeness)

	#queried nodes	type of output	decision mechanism	success probability	certificates
dist. testing	n	$\leq \lceil \log \Delta \rceil$ bits	algorithm	deterministic	no

Local algorithm:

- outputs the degree of the node

Decision mechanism:

- YES $\Longleftrightarrow \sum$ outputs $= 2n - 2$

Main question answered by this work

Are $\log \Delta$ bits really necessary?

(The answer is essentially "yes".)

Distributed testing (cycle-freeness)

	#queried nodes	type of output	decision mechanism	success probability	certificates
dist. testing	n	$\leq \lceil \log \Delta \rceil$ bits	algorithm	deterministic	no

Local algorithm:

- outputs the degree of the node

Decision mechanism:

- YES $\Longleftrightarrow \sum$ outputs $= 2n - 2$

Main question answered by this work

Are $\log \Delta$ bits really necessary?

(The answer is essentially "yes".)

Distributed testing (cycle-freeness)

	#queried nodes	type of output	decision mechanism	success probability	certificates
dist. testing	n	$\leq \lceil \log \Delta \rceil$ bits	algorithm	deterministic	no

Local algorithm:

- outputs the degree of the node

Decision mechanism:

- YES $\Longleftrightarrow \sum$ outputs $= 2n - 2$

Main question answered by this work

Are $\log \Delta$ bits really necessary?

(The answer is essentially "yes".)

Distributed testing (cycle-freeness)

	#queried nodes	type of output	decision mechanism	success probability	certificates
dist. testing	n	$\leq \lceil \log \Delta \rceil$ bits	algorithm	deterministic	no

Local algorithm:

- outputs the degree of the node

Decision mechanism:

- YES $\Longleftrightarrow \sum$ outputs $= 2n - 2$

Main question answered by this work

Are $\log \Delta$ bits really necessary?

(The answer is essentially "yes".)

Distributed verification (cycle-freeness)

	#queried nodes	type of output	decision mechanism	success probability	certificates
dist. verification	n	yes/no	$\forall$ yes / $\exists$ no	deterministic	$\Theta(\log n)$ bits

Certificates:

- choose an arbitrary node r
- node v's certificate is its distance to r

Local algorithm:

- YES $\Longleftrightarrow$ both are true
  - if cert. $= 0$, then all neighbors have cert. 1
  - if cert. is $x \neq 0$, then exactly one neighbor has cert. $x-1$ and all others have cert. $x+1$

Distributed verification (cycle-freeness)

	#queried nodes	type of output	decision mechanism	success probability	certificates
dist. verification	n	yes/no	$\forall$ yes / $\exists$ no	deterministic	$\Theta(\log n)$ bits

Certificates:

- choose an arbitrary node r
- node v's certificate is its distance to r

Local algorithm:

- YES $\Longleftrightarrow$ both are true
  - if cert. $= 0$, then all neighbors have cert. 1
  - if cert. is $x \neq 0$, then exactly one neighbor has cert. $x - 1$ and all others have cert. $x + 1$

Distributed verification (cycle-freeness)

	#queried nodes	type of output	decision mechanism	success probability	certificates
dist. verification	n	yes/no	$\forall$ yes / $\exists$ no	deterministic	$\Theta(\log n)$ bits

Certificates:

- choose an arbitrary node r
- node v's certificate is its distance to r

Local algorithm:

- YES $\Longleftrightarrow$ both are true
  - if cert. $= 0$, then all neighbors have cert. 1
  - if cert. is $x \neq 0$, then exactly one neighbor has cert. $x-1$ and all others have cert. $x+1$

Distributed certification (cycle-freeness)

	#queried nodes	type of output	decision mechanism	success probability	certificates
dist. certification	n	2 bits	algorithm	deterministic	2 bits

Certificates:

- choose an arbitrary node r; node v's certificate is
  - if $v = r$, then 3
  - if $v \neq r$, then its distance to r modulo 3

Local algorithm:

- outputs the pair (b, b') where
  - if cert. $= 3$, then $b = 1$ and, $b' = 1$ iff all neighbors have cert. 1
  - if cert. is $x \neq 3$, then $b = 0$ and, $b' = 1$ iff exactly one neighbor has cert. $x - 1$ and all others have cert. $x + 1$

Decision mechanism:

- YES $\iff \sum b = 1$ and $\bigwedge b' = 1$

Distributed certification (cycle-freeness)

	#queried nodes	type of output	decision mechanism	success probability	certificates
dist. certification	n	2 bits	algorithm	deterministic	2 bits

Certificates:

- choose an arbitrary node r; node v's certificate is
  - if $v = r$, then 3
  - if $v \neq r$, then its distance to r modulo 3

Local algorithm:

- outputs the pair (b, b') where
  - if cert. $= 3$, then $b = 1$ and, $b' = 1$ iff all neighbors have cert. 1
  - if cert. is $x \neq 3$, then $b = 0$ and, $b' = 1$ iff exactly one neighbor has cert. $x - 1$ and all others have cert. $x + 1$

Decision mechanism:

- YES $\iff \sum b = 1$ and $\bigwedge b' = 1$

Distributed certification (cycle-freeness)

	#queried nodes	type of output	decision mechanism	success probability	certificates
dist. certification	n	2 bits	algorithm	deterministic	2 bits

Certificates:

- choose an arbitrary node r; node v's certificate is
  - if $v = r$, then 3
  - if $v \neq r$, then its distance to r modulo 3

Local algorithm:

- outputs the pair (b, b') where
  - if cert. $= 3$, then $b = 1$ and, $b' = 1$ iff all neighbors have cert. 1
  - if cert. is $x \neq 3$, then $b = 0$ and, $b' = 1$ iff exactly one neighbor has cert. $x - 1$ and all others have cert. $x + 1$

Decision mechanism:

- YES $\iff \sum b = 1$ and $\bigwedge b' = 1$

Distributed certification (cycle-freeness)

	#queried nodes	type of output	decision mechanism	success probability	certificates
dist. certification	n	2 bits	algorithm	deterministic	2 bits

Certificates:

- choose an arbitrary node r; node v's certificate is
  - if $v = r$, then 3
  - if $v \neq r$, then its distance to r modulo 3

Local algorithm:

- outputs the pair (b, b') where
  - if cert. $= 3$, then $b = 1$ and, $b' = 1$ iff all neighbors have cert. 1
  - if cert. is $x \neq 3$, then $b = 0$ and, $b' = 1$ iff exactly one neighbor has cert. $x - 1$ and all others have cert. $x + 1$

Decision mechanism:

- YES $\iff \sum b = 1$ and $\bigwedge b' = 1$

Our main result

For the cycle-freeness decision problem:

	#queried nodes	type of output	decision mechanism	success probability	certificates
dist. decision	n	yes/no	$\forall$ yes / $\exists$ no	impossible	no
dist. testing	n	$\leq \lceil \log \Delta \rceil$ bits	algorithm	deterministic	no
dist. verification	n	yes/no	$\forall$ yes / $\exists$ no	deterministic	$\Theta(\log n)$ bits
dist. certification	n	2 bits	algorithm	deterministic	2 bits
dist. testing [this paper]	n	$\geq \lceil \log \Delta \rceil - 1$ bits	algorithm	deterministic	no

Model

$\mathcal{LOCAL}$ model

- Pairwise distincts IDs
- Synchronous fault-free rounds (& simultaneous wake-up)
- Messages of unlimited size

Distributed testing

- All nodes execute a t-round local distributed algorithm, producing a local output (t is a constant)
- A central authority gathers the outputs as a multiset to produce a global decision

Theorem to be proved

Every distributed tester for cycle-freeness in connected max.-degree-Δ graphs has output size at least $\lceil \log \Delta \rceil - 1$ bits.

Model

$\mathcal{LOCAL}$ model

- Pairwise distincts IDs
- Synchronous fault-free rounds (& simultaneous wake-up)
- Messages of unlimited size

Distributed testing

- All nodes execute a t-round local distributed algorithm, producing a local output (t is a constant)
- A central authority gathers the outputs as a multiset to produce a global decision

Theorem to be proved

Every distributed tester for cycle-freeness in connected max.-degree-Δ graphs has output size at least $\lceil \log \Delta \rceil - 1$ bits.

Model

$\mathcal{LOCAL}$ model

- Pairwise distincts IDs
- Synchronous fault-free rounds (& simultaneous wake-up)
- Messages of unlimited size

Distributed testing

- All nodes execute a t-round local distributed algorithm, producing a local output (t is a constant)
- A central authority gathers the outputs as a multiset to produce a global decision

Theorem to be proved

Every distributed tester for cycle-freeness in connected max.-degree-Δ graphs has output size at least $\lceil \log \Delta \rceil - 1$ bits.

Usefulness of the identifiers

ID-oblivious

The algorithm's output does not depend on the IDs.

Order-invariant

The algorithm's output only depends on the relative order of the IDs.

ID-dependent

The algorithm's output freely depends on the IDs.

Usefulness of the identifiers

ID-oblivious

The algorithm's output does not depend on the IDs.

Order-invariant

The algorithm's output only depends on the relative order of the IDs.

ID-dependent

The algorithm's output freely depends on the IDs.

Usefulness of the identifiers

ID-oblivious

The algorithm's output does not depend on the IDs.

Order-invariant

The algorithm's output only depends on the relative order of the IDs.

ID-dependent

The algorithm's output freely depends on the IDs.

ID-oblivious, $t = 0$

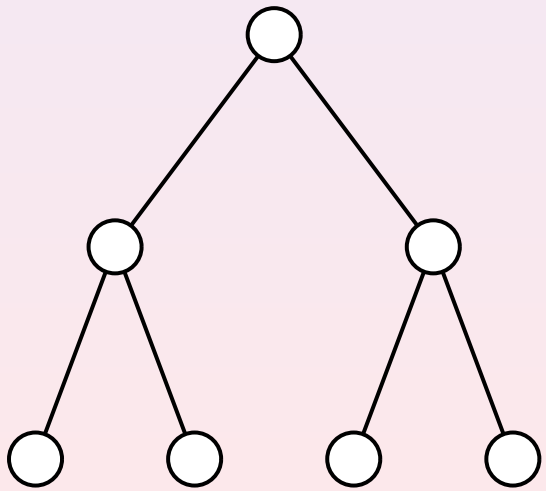
Sketch of the proof

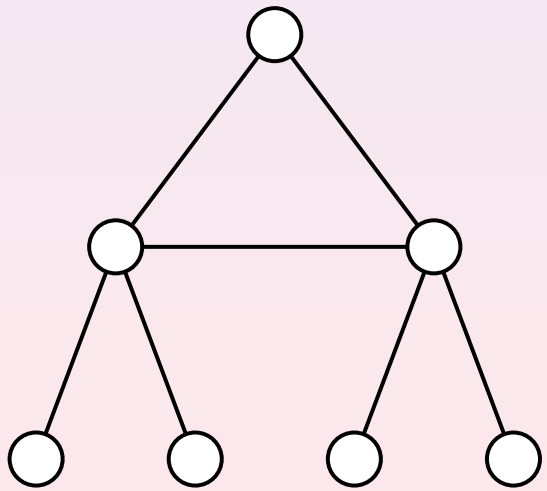
- $< \lceil \log \Delta \rceil - 1$ bits $\Rightarrow$ same output for degrees i and $j > i$
- construction of two almost identical graphs
  - a tree with $x + z$, resp. y, nodes of degree i, resp. j
  - a non-tree with x, resp. $y + z$, nodes of degree i, resp. j

ID-oblivious, $t = 0$

Sketch of the proof

- $< \lceil \log \Delta \rceil - 1$ bits $\Rightarrow$ same output for degrees i and $j > i$
- construction of two almost identical graphs
  - a tree with $x + z$, resp. y, nodes of degree i, resp. j
  - a non-tree with x, resp. $y + z$, nodes of degree i, resp. j





Generalizing to arbitrary (constant) t

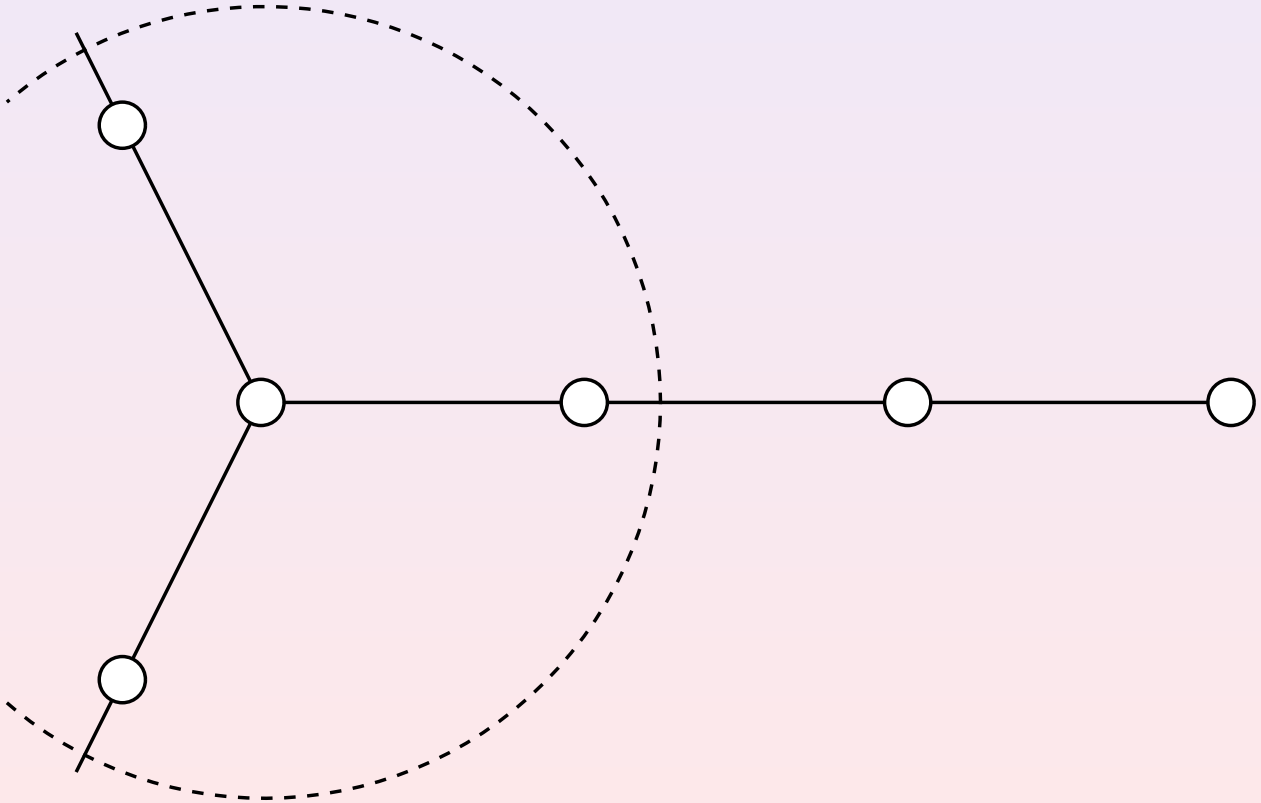
Use of subdivided trees:

- replace each edge by a path of length $2t + 1$
- consider the vector of outputs from the ball

Generalizing to arbitrary (constant) t

Use of subdivided trees:

- replace each edge by a path of length $2t + 1$
- consider the vector of outputs from the ball



The case of the subdivided graphs

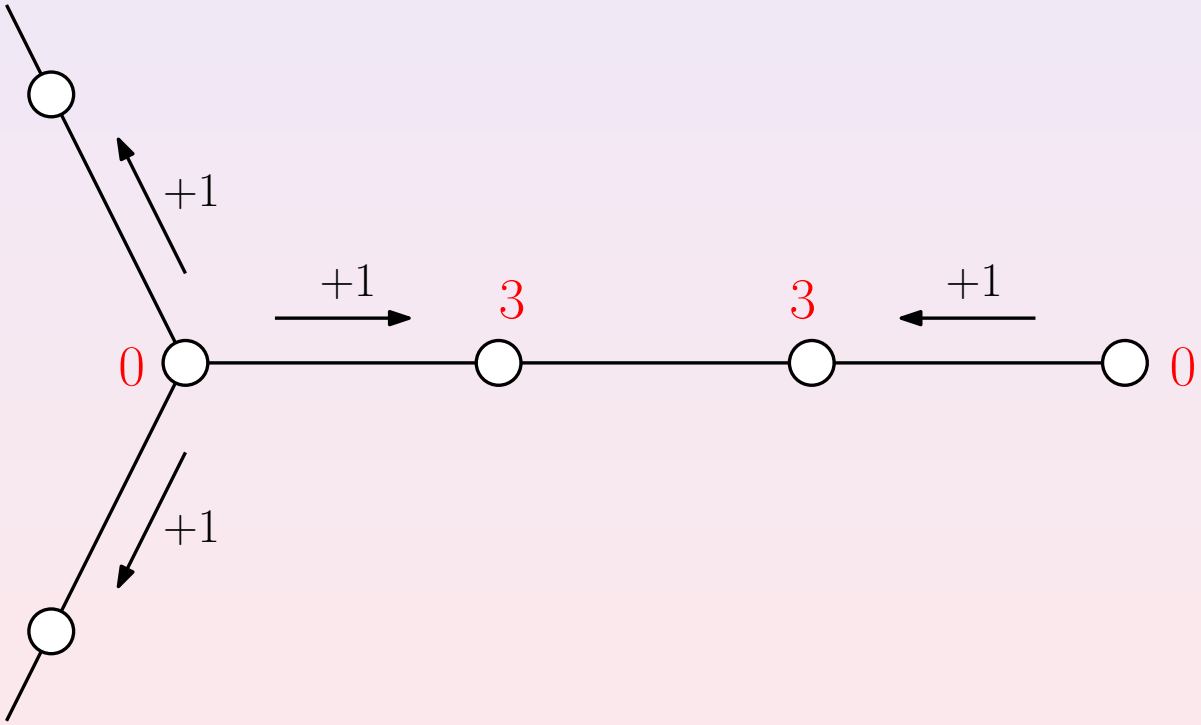
Lemma

Only four outputs are sufficient in subdivided graphs!

The case of the subdivided graphs

Lemma

Only four outputs are sufficient in subdivided graphs!



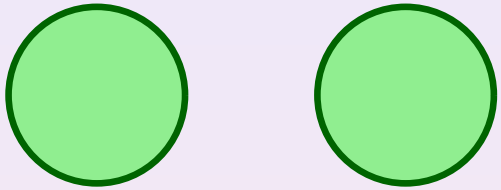
Nodes of degree different from 2 distribute their degree.

A solution

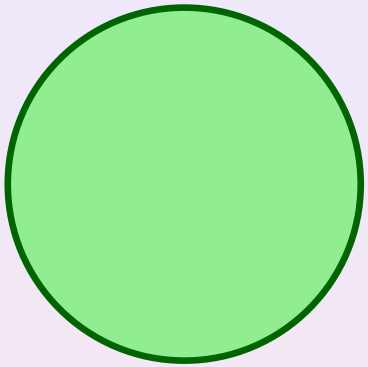
Hiding the trees into the forest!

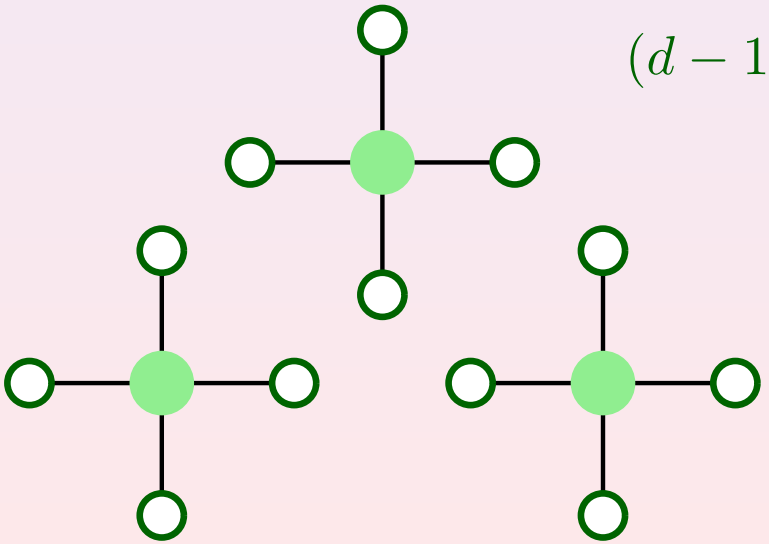
A solution

Hiding the trees into the forest!

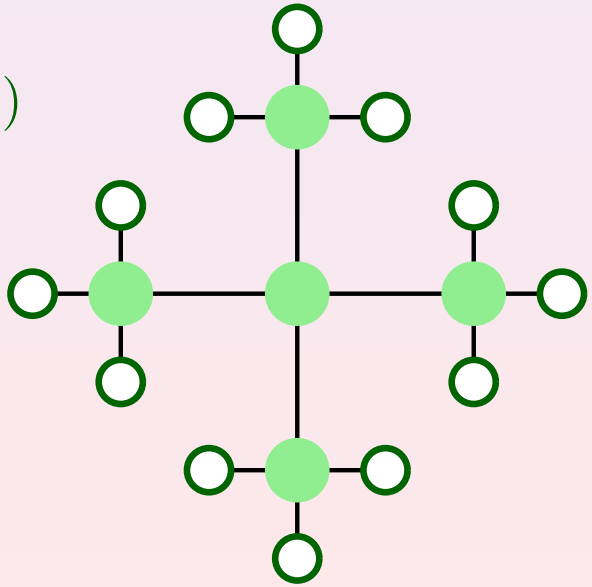


$$2 \cdot (2\pi R) = 2\pi(2 \cdot R)$$

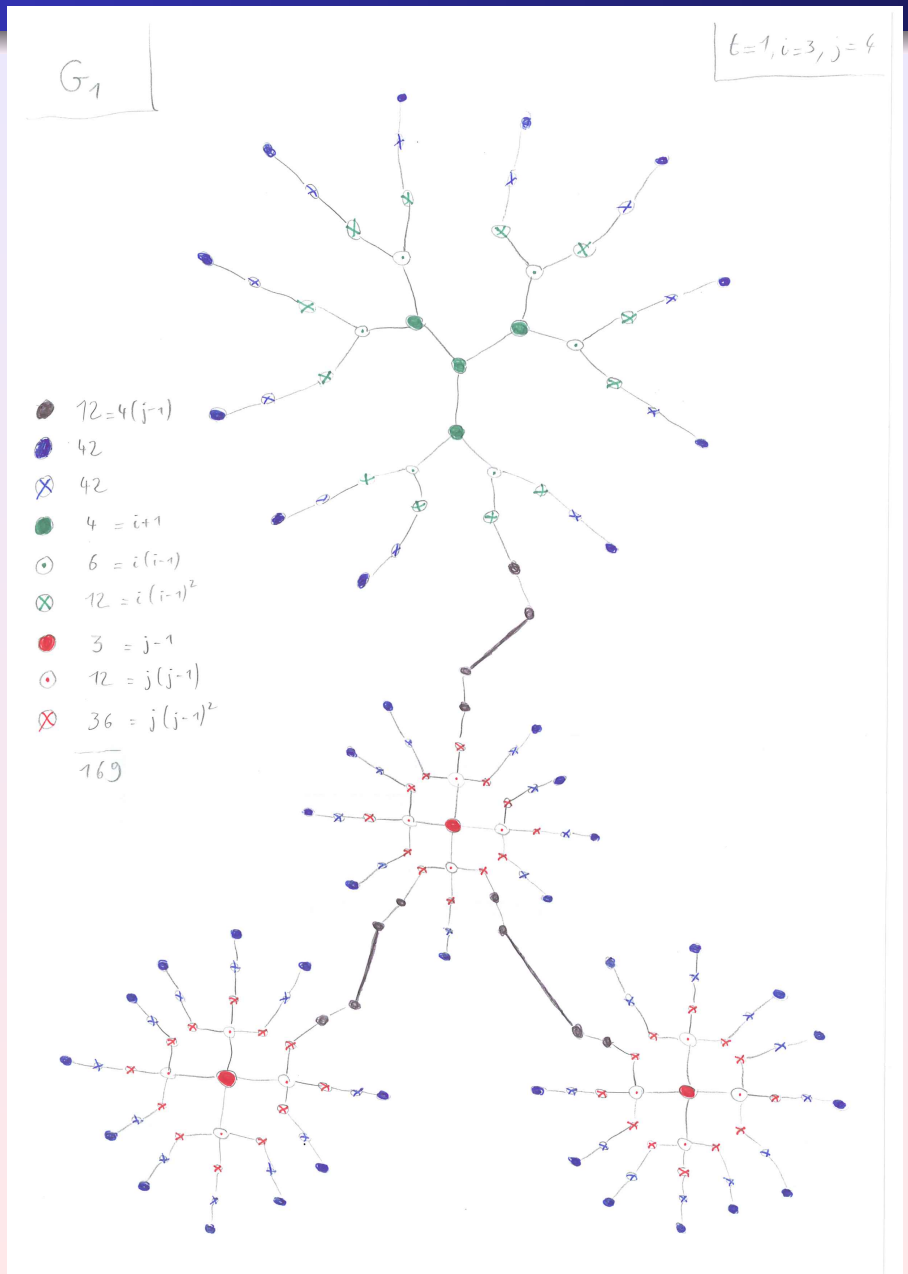


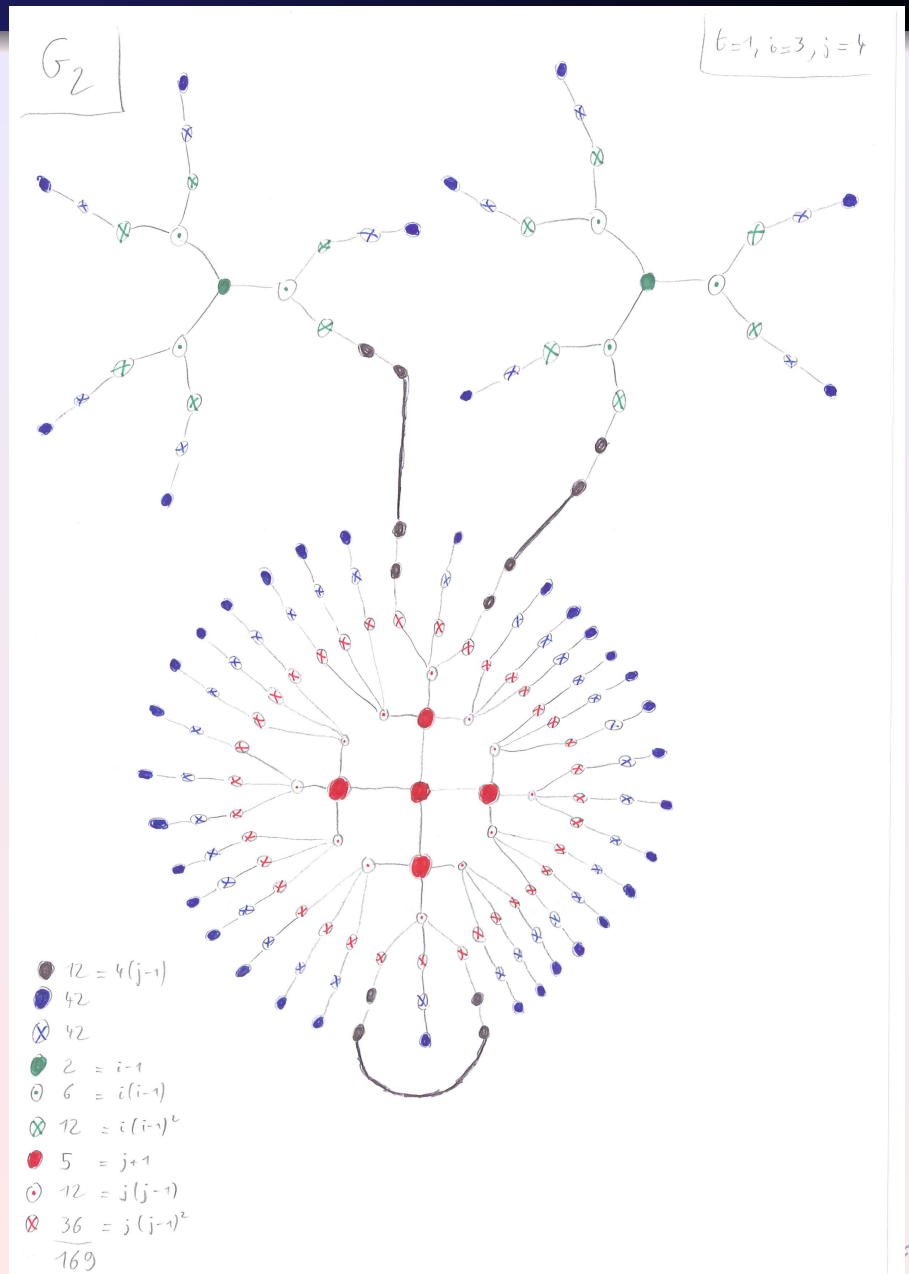


$$(d-1) \cdot d = d \cdot (d-1)$$



ID-oblivious solution

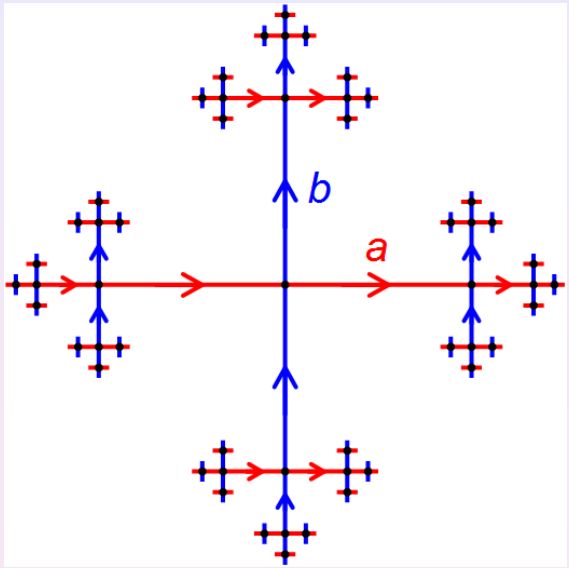




Towards an order-invariant solution

Definition: free group (wikipedia)

The free group F_S over a given set S consists of all expressions (a.k.a. words, or terms) that can be built from members of S, considering two expressions different unless their equality follows from the group axioms (e.g. $st = suu^{-1}t$, but $s \neq t$ for $s, t, u \in S$). The members of S are called generators of F_S.



Definition: linearly ordered group (wikipedia)

A linearly ordered group is a group G equipped with a total order "$\leq$", that is translation-invariant: Let $x, y, z \in G$, we say that $(G, \leq)$ is a left-ordered group if $x \leq y$ implies $z\,x \leq z\,y$.

An order-invariant solution

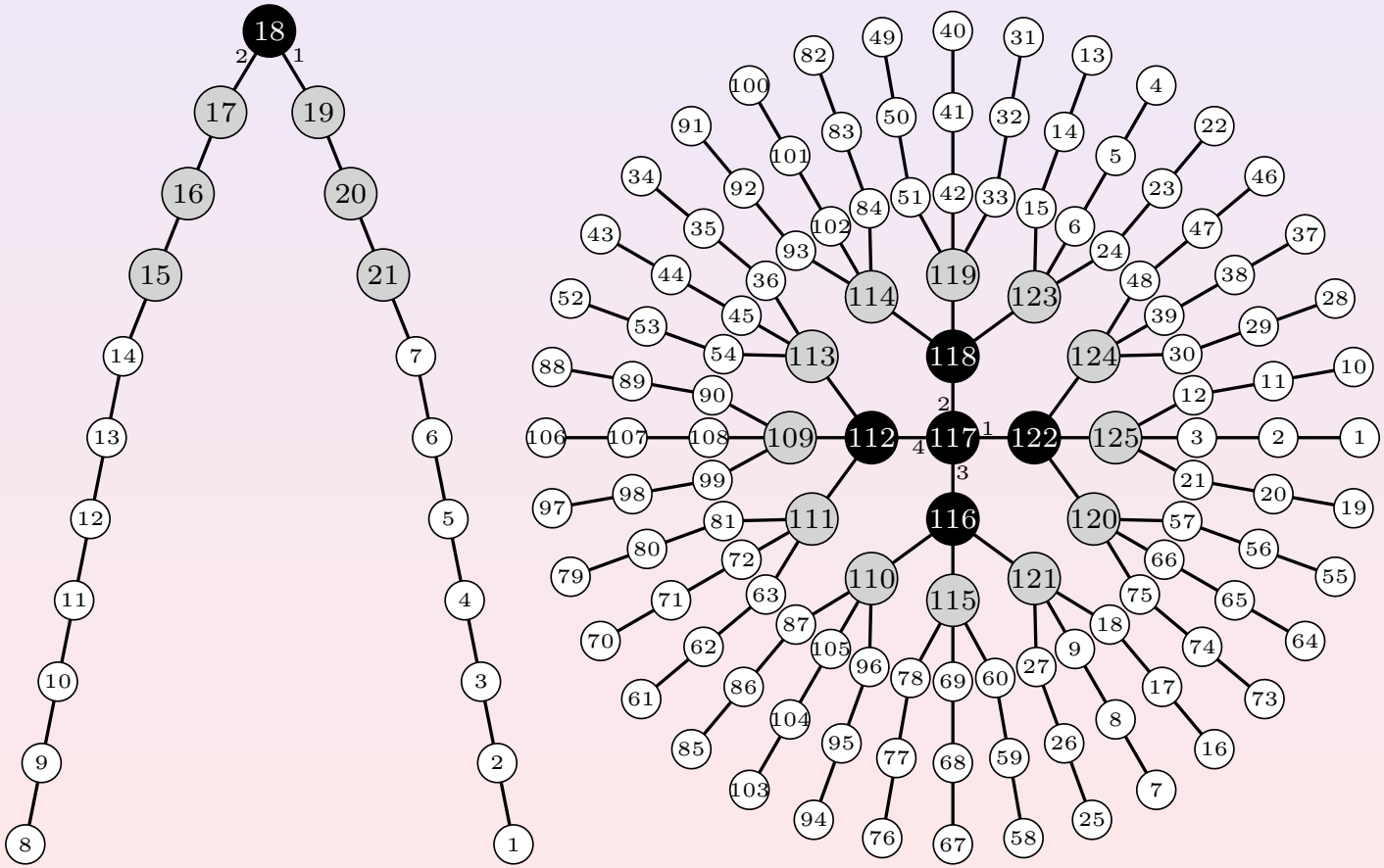
Theorem (at least from the 1940's)

Every free group is left-orderable.

An order-invariant solution

Theorem (at least from the 1940's)

Every free group is left-orderable.



The final argument

Theorem (of independent interest)

$\exists$ a solution $\Longrightarrow$ $\exists$ an order-invariant solution

For every non-negative integers k, t, Δ, and every language $\mathcal{L}$ defined on connected graphs with maximum degree Δ, and k-valued domain, if there exists a t-round construction algorithm $\mathcal{A}$ for $\mathcal{L}$, then there is a t-round order-invariant construction algorithm $\mathcal{A}'$ for $\mathcal{L}$.

Thank you

for your attention