Randomized Local Network Computing

Laurent Feuilloley[1] · Pierre Fraigniaud[2]

[1] ENS Cachan · Université Paris Diderot · Aalto University
[2] Université Paris Diderot

SPAA 2015 · Portland · June 15

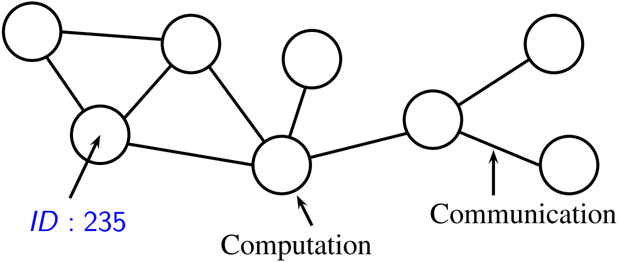
A derandomization theorem

Informally our theorem is :

- In a distributed computing model
- If a language can be checked locally with randomization
- Then :
  - If it can be constructed locally with randomization
  - Then it can be constructed locally without randomization

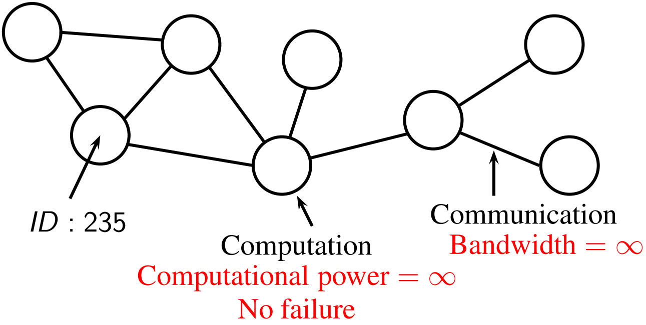
LOCAL model

- A network of machines
- Every vertex has a unique identifier



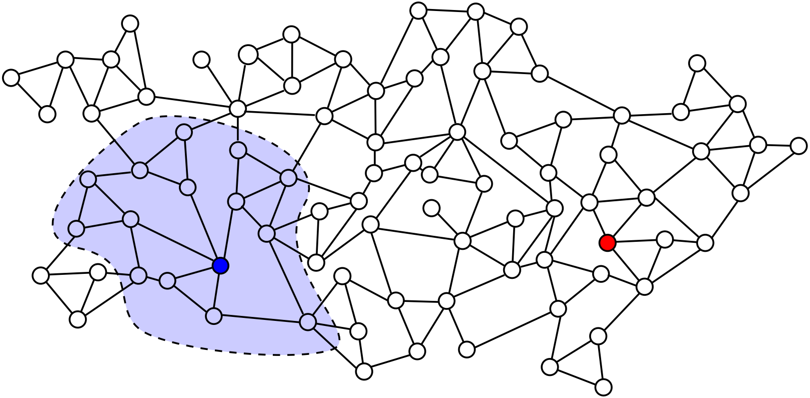
LOCAL model

- A network of machines
- Every vertex has a unique identifier



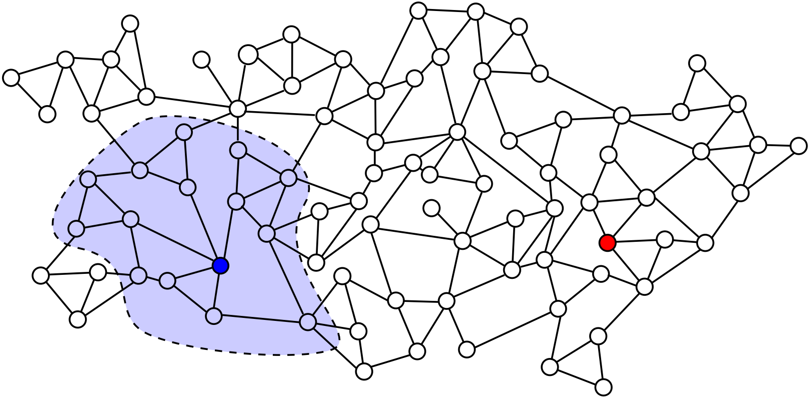
LOCAL model

First point of view : minimize the number of rounds



LOCAL model

Second point of view : local computation



max degree = Δ

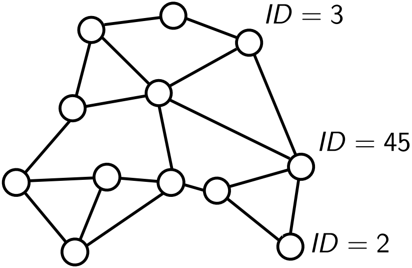
Theorem

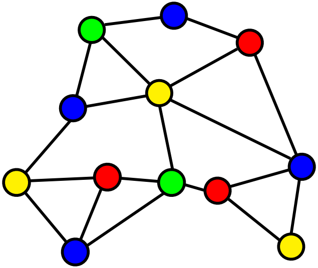
- In the LOCAL model
- If a language can be checked locally with randomization
- Then :
  - If it can be constructed locally with randomization
  - Then it can be constructed locally without randomization

Construction vs Decision

Example : $(\Delta + 1)$-coloring

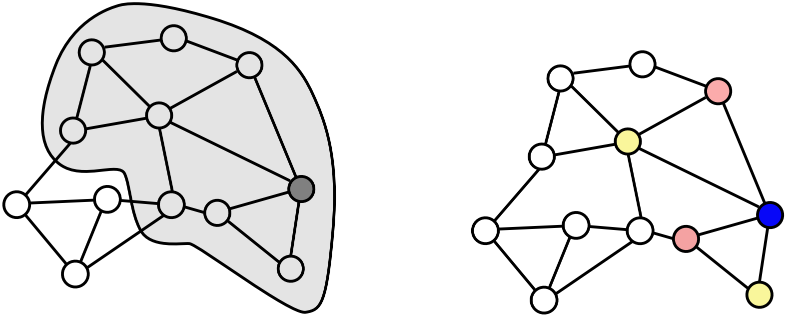
Construction from a global perspective.





Construction vs Decision

Construction from a local perspective.



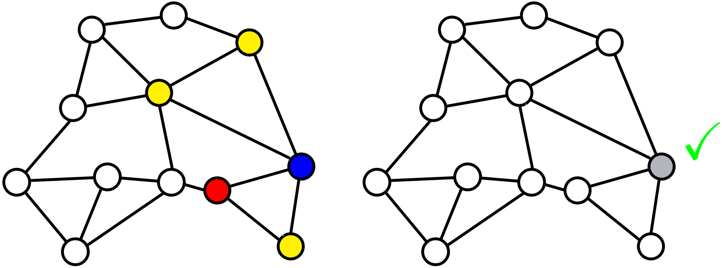
Construction vs Decision

Theorem (Linial'92) :

Constructing a $(\Delta + 1)$-colouring requires $\Omega(\log^* n)$ rounds.

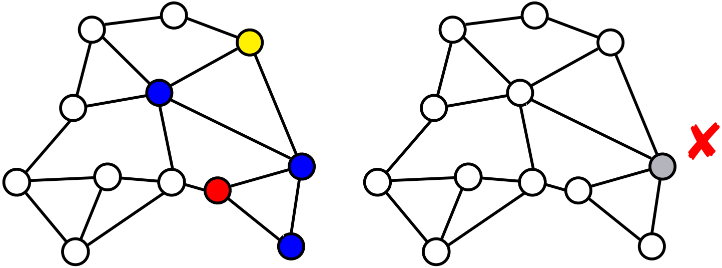
Construction vs Decision

Decision from a local perspective



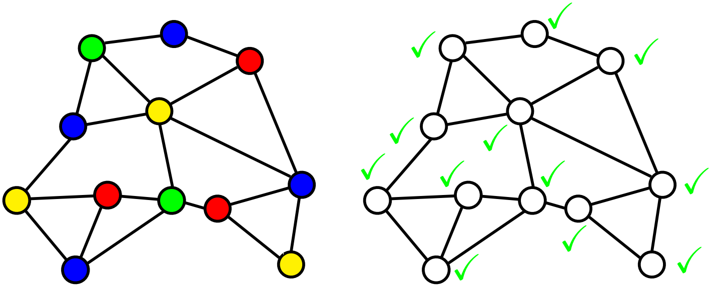
Construction vs Decision

Decision from a local perspective



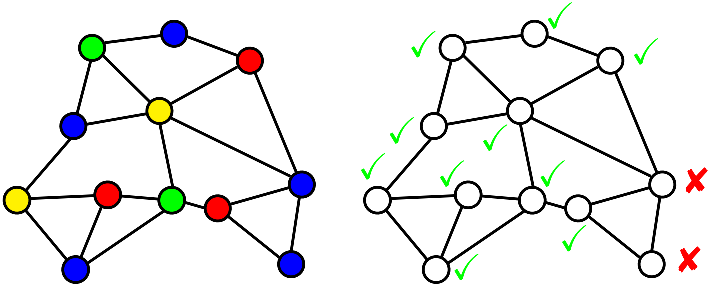
Construction vs Decision

Decision from a global perspective



Construction vs Decision

Decision from a global perspective

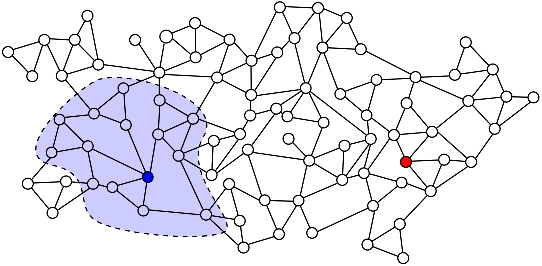


Theorem

- In the LOCAL model
- If a language can be checked locally with randomization
- Then :
    - If it can be constructed locally with randomization
    - Then it can be constructed locally without randomization

Locally ?

Here locally means constant number of rounds

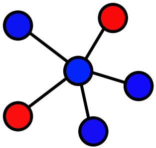


Coloring verification can be done locally → 1 round,
but coloring construction cannot→ $\log^* n$ rounds.

Locally ?

What can be constructed locally ?

$\rightarrow$ Weak coloring, fractional coloring, and some approximations



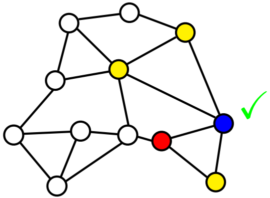
Theorem

- In the LOCAL model
- If a language can be checked in $O(1)$ rounds, with randomization
- Then :
  - If it can be constructed in $O(1)$ rounds with randomization
  - Then it can be constructed in $O(1)$ rounds without randomization

Languages and classes

- a language : is a set
  $\{(G,x)$ satisfying a property P $\}$
- A class is a set of languages

$\rightarrow$ LD $=$ the languages that can be checked in constant time deterministically.



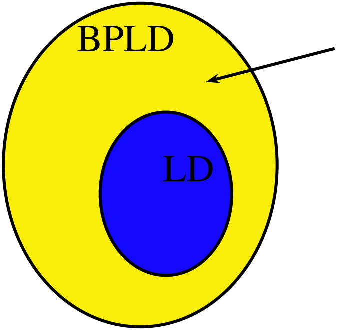
Languages and classes

BPLD = the languages that can be checked in constant time using randomization.

More precisely :
there exists a checker, and $p \in \left(\frac{1}{2}, 1\right]$ s.t. :

- If $(G, x) \in \mathcal{L}$, then Pr[all nodes accept] $\geq p$
- If $(G, x) \notin \mathcal{L}$ then Pr[a node rejects] $\geq p$

f-resilient tasks and BPLD



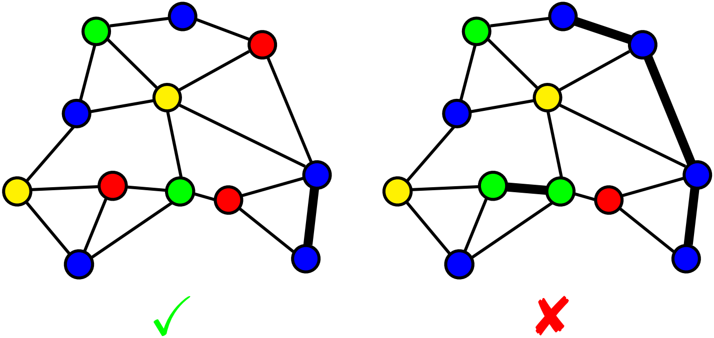
f-resilient tasks

$\mathcal{L}_f = \mathcal{L} \in$ LD

$+$ errors accepted on f nodes.

f-resilient tasks and BPLD

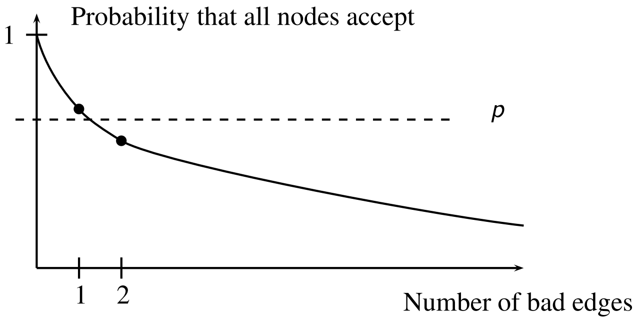
Coloring with one bad edge is ok, but not more.



f-resilient tasks and BPLD

Strategy :

- If the coloring is good, accept
- If the coloring is bad, reject with probability q



Back to the derandomization theorem

More formally the theorem is :

- In the LOCAL model
- If $\mathcal{L} \in$ BPLD
- Then :
  - If $\mathcal{L}$ can be constructed in $O(1)$ rounds with randomization
  - Then it can be constructed in $O(1)$ rounds deterministically

A glimpse of the proof

Two main steps :

- Using Ramsey theory to reduce to a special case (from Naor-Stockmeyer'93)
- Proving that locality prevent weird correlations

Further works

- LD $\rightarrow$ BPLD $\rightarrow$?
- Get a better understanding of randomization in network distributed computing.

Further works

- LD $\rightarrow$ BPLD $\rightarrow$?
- Get a better understanding of randomization in network distributed computing.

Thank you !