Bisimulation and Modal Logic in Distributed Computing

Tuomo Lempiäinen

Distributed Algorithms group, Department of Computer Science, Aalto University

(joint work with Lauri Hella, Matti Järvisalo, Antti Kuusisto, Juhana Laurinharju, Kerkko Luosto, Jukka Suomela and Jonni Virtema)

Computational Logic Day 2016

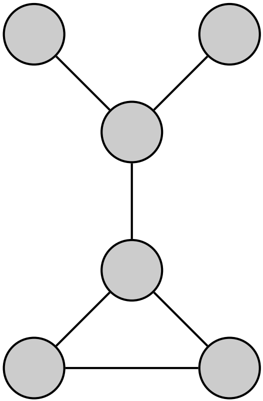
December 8, 2016 @ Aalto University

Publications

Brief overview of two papers:

- Hella, Järvisalo, Kuusisto, Laurinharju, Lempiäinen, Luosto, Suomela and Virtema:
  Weak models of distributed computing, with connections to modal logic
  PODC 2012, *Distributed Computing* 2015
- Lempiäinen:
  Ability to count messages is worth $\Theta(\Delta)$ rounds in distributed computing
  LICS 2016

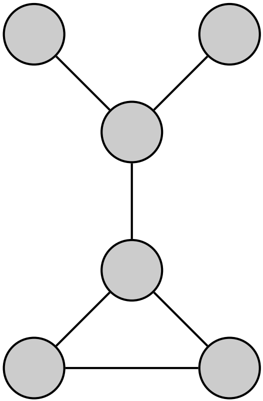
The model of computation



A simple finite undirected graph, whose each node is a deterministic state machine that

- runs the same algorithm,
- can communicate with its neighbours,
- produces a local output.

The model of computation

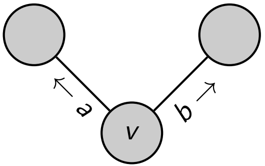


A simple finite undirected graph, whose each node is a deterministic state machine that

- runs the same algorithm,
- can communicate with its neighbours,
- produces a local output.

Anonymous nodes $\Rightarrow$ a weak model of computation.

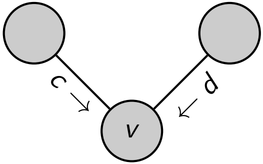
Communication in synchronous rounds



In every round, each node v

1. sends messages to its neighbours,
2. receives messages from its neighbours,
3. updates its state.

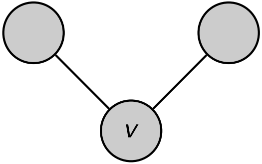
Communication in synchronous rounds



In every round, each node v

1. sends messages to its neighbours,
2. receives messages from its neighbours,
3. updates its state.

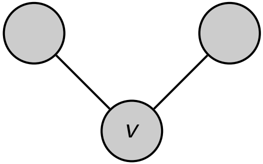
Communication in synchronous rounds



In every round, each node v

1. sends messages to its neighbours,
2. receives messages from its neighbours,
3. updates its state.

Communication in synchronous rounds

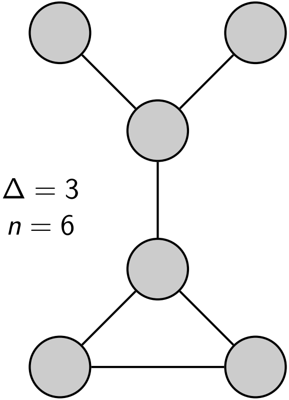


In every round, each node v

1. sends messages to its neighbours,
2. receives messages from its neighbours,
3. updates its state.

Eventually, each node halts and announces its own local output.

Focus on communication, not computation

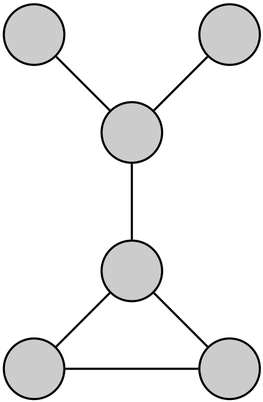


The running time of an algorithm is the *number of communications rounds*.

The running time may depend on two parameters:

- the maximum degree of the graph, Δ,
- the number of nodes, n.

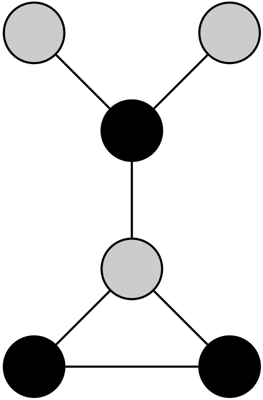
Graph problems



We study *graph problems* where

- the problem instance is the communication graph $G = (V, E)$,
- a solution is a mapping $S\colon V \to Y$ from nodes to local outputs.

Graph problems



$Y = \{0, 1\}$

We study *graph problems* where

- the problem instance is the communication graph $G = (V, E)$,
- a solution is a mapping $S \colon V \to Y$ from nodes to local outputs.

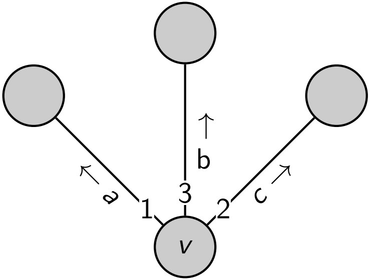
Often the solution is an encoding of a subset of vertices or edges of the graph.

One typical example is the *minimum vertex cover*.

PODC 2012: seven variants of the model

Options for sending messages:

- a port number for each neighbour (V),

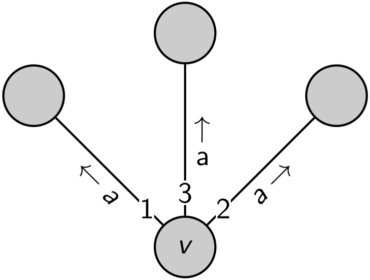


Node v sends a **vector** (a, c, b).

PODC 2012: seven variants of the model

Options for sending messages:

- a port number for each neighbour (V),
- broadcast the same message to all neighbours (B).



Node v **broadcasts** message a.

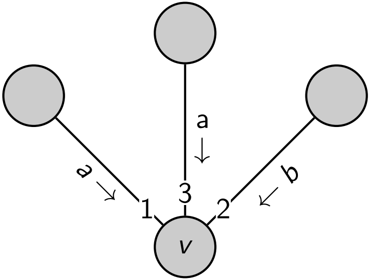
PODC 2012: seven variants of the model

Options for sending messages:

- a port number for each neighbour (V),
- broadcast the same message to all neighbours (B).

Options for receiving messages:

- a port number for each neighbour (V),



Node v receives a **vector** (a, b, a).

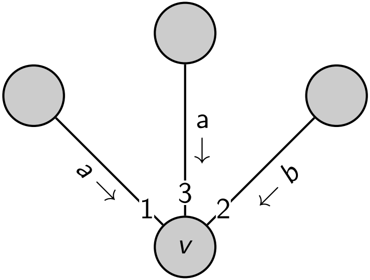
PODC 2012: seven variants of the model

Options for sending messages:

- a port number for each neighbour (V),
- broadcast the same message to all neighbours (B).

Options for receiving messages:

- a port number for each neighbour (V),
- receive a multiset of messages (M),



Node v receives a **multiset** $\{a, a, b\}$.

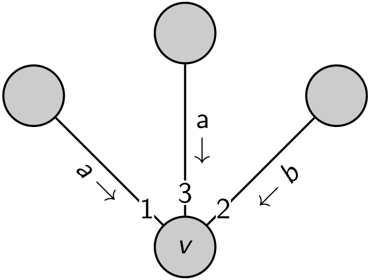
PODC 2012: seven variants of the model

Options for sending messages:

- a port number for each neighbour (V),
- broadcast the same message to all neighbours (B).

Options for receiving messages:

- a port number for each neighbour (V),
- receive a multiset of messages (M),
- receive a set of messages (S).



Node v receives a **set** $\{a, b\}$.

PODC 2012: seven variants of the model

Options for sending messages:

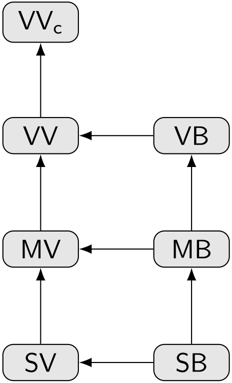
- **a port number for each neighbour (V)**,
- broadcast the same message to all neighbours (B).

Options for receiving messages:

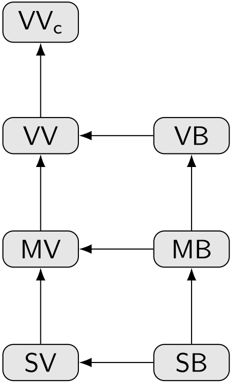
- **a port number for each neighbour (V)**,
- receive a multiset of messages (M),
- receive a set of messages (S).

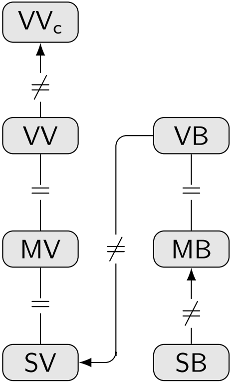
We can require the outgoing and incoming port numbers to be consistent $\Rightarrow$ the *port-numbering model* ($\mathsf{VV_c}$).

PODC 2012: a hierarchy of complexity classes



PODC 2012: a hierarchy of complexity classes





Theorem

$\mathsf{SB} \subsetneq \mathsf{MB} = \mathsf{VB} \subsetneq \mathsf{SV} = \mathsf{MV} = \mathsf{VV} \subsetneq \mathsf{VV_c}$.

PODC 2012: connections to modal logic

The constant-time variant of each of the seven complexity classes can be characterised by a modal logic such that there is a canonical one-to-one correspondence between algorithms and modal formulas.

PODC 2012: connections to modal logic

The constant-time variant of each of the seven complexity classes can be characterised by a modal logic such that there is a canonical one-to-one correspondence between algorithms and modal formulas.

Example: graded modal logic (GML),

$$\varphi := q_n \mid (\varphi \wedge \varphi) \mid \neg\varphi \mid \Diamond\varphi, \mid \Diamond_{\geq k}\varphi,$$

where q_n are proposition symbols and $k \in \mathbb{N}$.

$$\begin{aligned} G, v \models q_n \quad & \text{iff} \quad \text{degree}(v) = n, \\ G, v \models \Diamond_{\geq k}\varphi \quad & \text{iff} \quad \big|\{w \in V : (v, w) \in E \text{ and } G, w \models \varphi\}\big| \geq k. \end{aligned}$$

PODC 2012: connections to modal logic

The constant-time variant of each of the seven complexity classes can be characterised by a modal logic such that there is a canonical one-to-one correspondence between algorithms and modal formulas.

Example: graded modal logic (GML),

$$\varphi := q_n \mid (\varphi \wedge \varphi) \mid \neg\varphi \mid \Diamond\varphi, \mid \Diamond_{\geq k}\varphi,$$

where q_n are proposition symbols and $k \in \mathbb{N}$.

$$\begin{aligned} G, v \models q_n \quad &\text{iff} \quad \text{degree}(v) = n, \\ G, v \models \Diamond_{\geq k}\varphi \quad &\text{iff} \quad \big|\{w \in V : (v, w) \in E \text{ and } G, w \models \varphi\}\big| \geq k. \end{aligned}$$

GML corresponds to the complexity class MB (receive a multiset, send by broadcasting).

PODC 2012: connections to modal logic

In each variant of modal logic, one can characterise definability by a variant of bisimulation.

A nonempty relation $Z \subseteq V \times V'$ is a *graded bisimulation* between $G = (V, E, \tau)$ and $G' = (V', E', \tau')$ if the following conditions hold.

1. If $(v, v') \in Z$, then $v \in \tau(q_n)$ iff $v' \in \tau'(q_n)$ for each q_n.
2. If $(v, v') \in Z$ and $X \subseteq E(v)$, then there is a set $X' \subseteq E'(v')$ such that $|X'| = |X|$ and for each $w' \in X'$ there is a $w \in X$ with $(w, w') \in Z$.
3. If $(v, v') \in Z$ and $X' \subseteq E'(v')$, then there is a set $X \subseteq E(v)$ such that $|X| = |X'|$ and for each $w \in X$ there is a $w' \in X'$ with $(w, w') \in Z$.

We use bisimulation to derive the separation results between the complexity classes.

The relationship of MV and SV

The simulation results used to show the equivalence of complexity classes do not increase the running time, except for one:

Theorem (PODC 2012)

Assume that there is an MV*-algorithm* $\mathcal{A}$ *that solves a problem* Π *in time* T*. Then there is an* SV*-algorithm* $\mathcal{B}$ *that solves* Π *in time* $T + 2\Delta - 2$*.*

Is this result tight?

LICS 2016: the simulation overhead is tight

Theorem

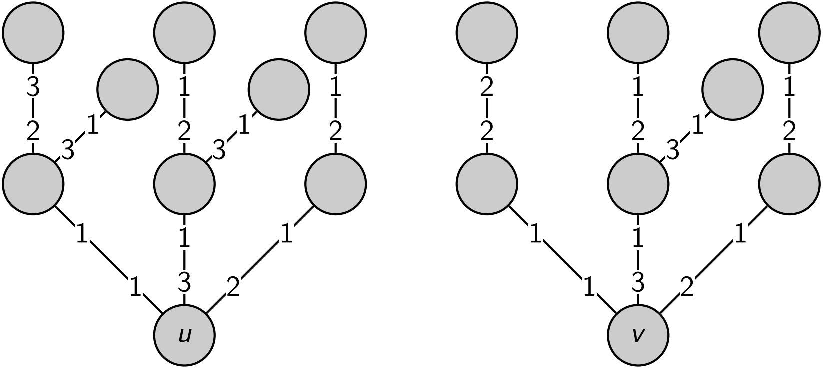
For each $\Delta \geq 2$ there is a port-numbered graph G_Δ with nodes u, v, w such that when executing any SV*-algorithm $\mathcal{A}$ in G_Δ, u receives identical messages from its neighbours v and w in rounds $1, 2, \ldots, 2\Delta - 2$.*

We can also separate the models by a graph problem:

Theorem

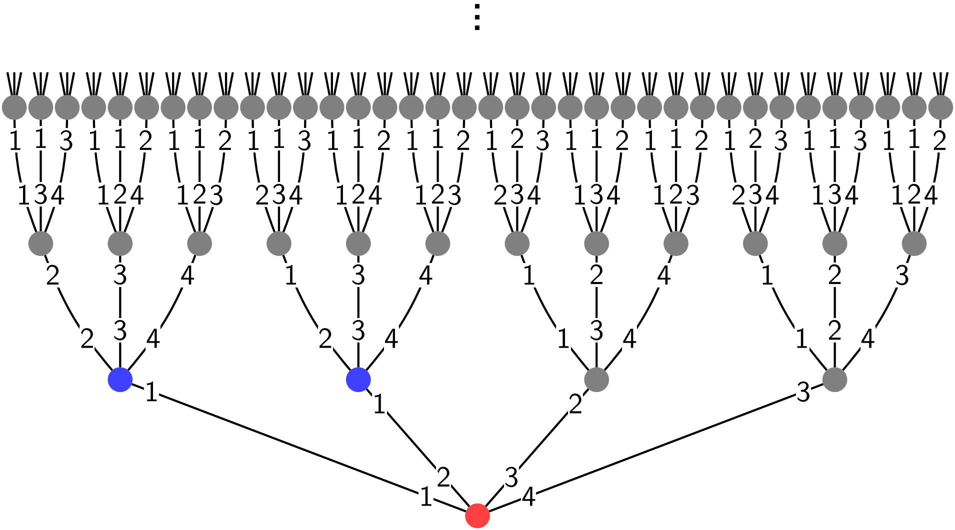
There is a graph problem Π that can be solved in one round by an MV*-algorithm but that requires at least $\Delta - 1$ rounds for all $\Delta \geq 2$, when solved by an* SV*-algorithm.*

Example: separating SV and MV



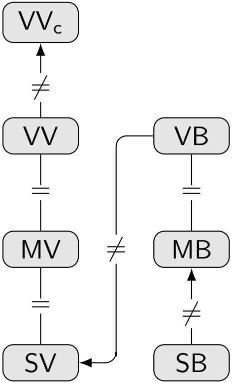
Output 1 if there is an even number of neighbours of even degree, 0 otherwise.

Generalisation: graph G_Δ (here $\Delta = 4$)



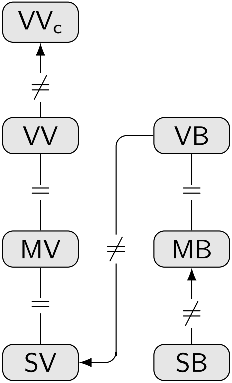
The blue nodes are bisimilar up to the distance $2\Delta - 2$.

Conclusion



- We defined seven complexity classes and characterised the containment relations.
- Each constant-time class corresponds to a variant of modal logic.
- Only in one case there is overhead in simulating a stronger model by a weaker one, and that overhead is unavoidable.

Conclusion



- We defined seven complexity classes and characterised the containment relations.
- Each constant-time class corresponds to a variant of modal logic.
- Only in one case there is overhead in simulating a stronger model by a weaker one, and that overhead is unavoidable.

Thanks! Questions?