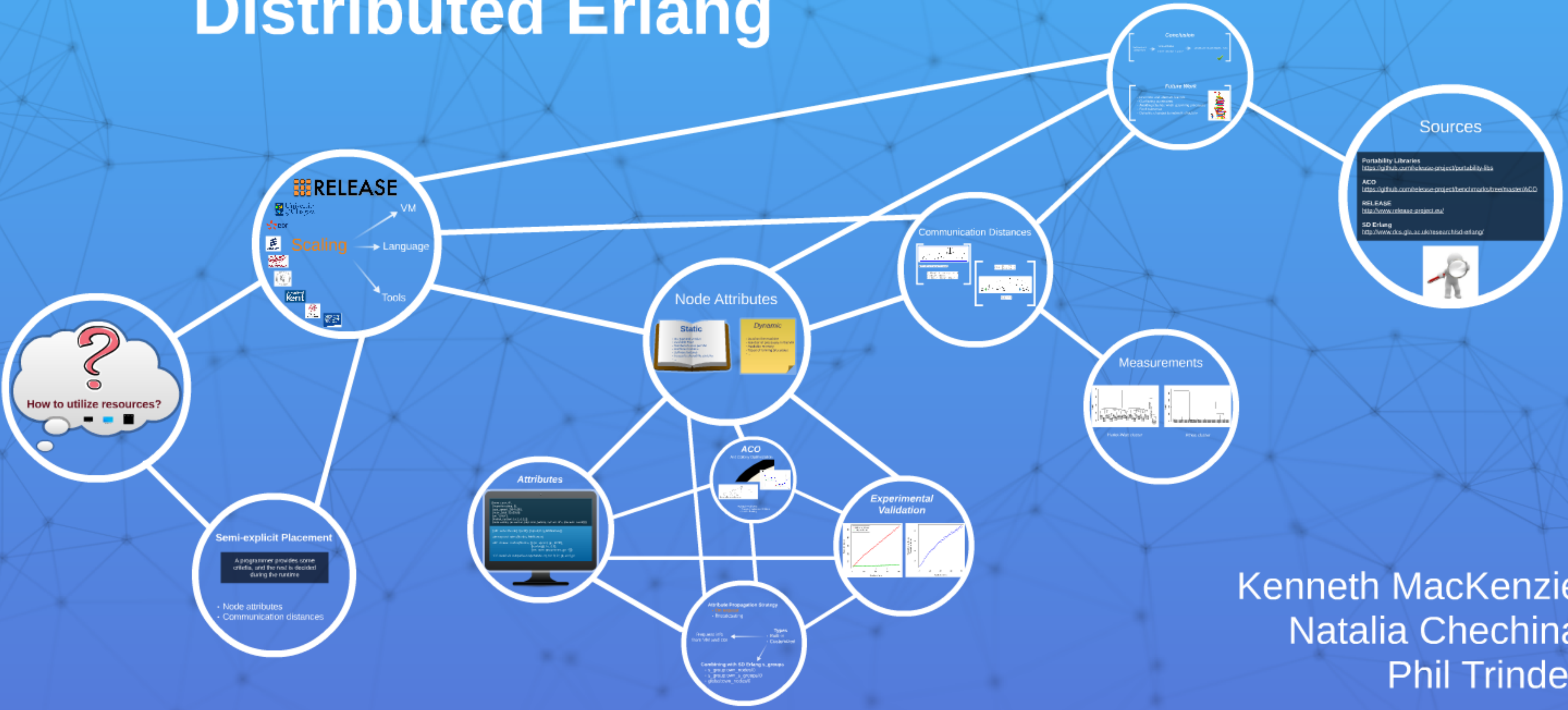
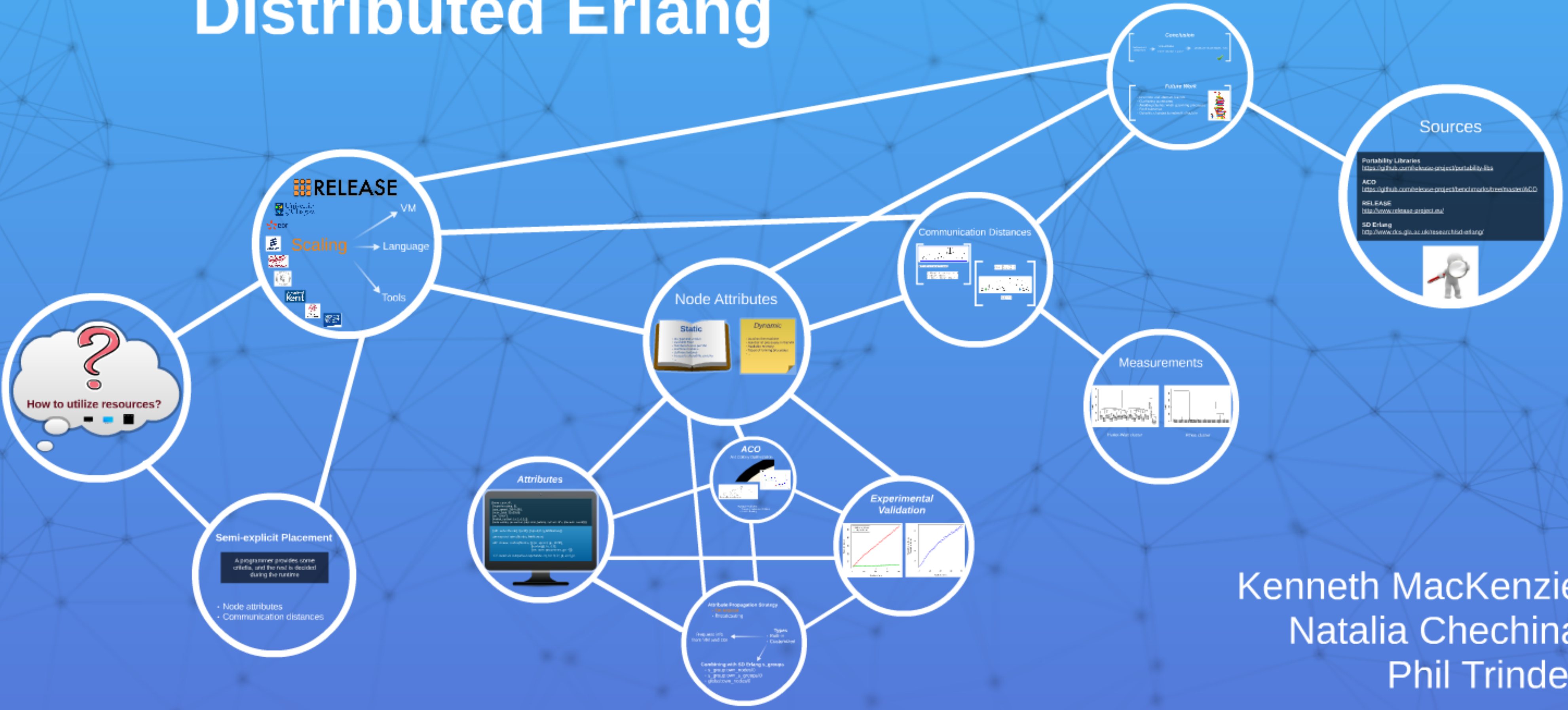
Performance Portability through Semi-explicit Placement in Distributed Erlang



Conclusion
Future Work
Sources
Portability Libraries
https://github.com/release-project/portability-libs
ACO
https://github.com/release-project/benchmarks/tree/master/ACO
RELEASE
http://www.release-project.eu/
SD Erlang
http://www.dcs.gla.ac.uk/research/sd-erlang/
RELEASE
Scaling
VM
Language
Tools
Communication Distances
Node Attributes
Static
Dynamic
Measurements
How to utilize resources?
Semi-explicit Placement
A programmer provides some criteria, and the rest is decided during the runtime
• Node attributes
• Communication distances
Attributes
ACO
Experimental Validation
Attribute Propagation Strategy

Kenneth MacKenzie
Natalia Chechina
Phil Trinder

Performance Portability through Semi-explicit Placement in Distributed Erlang


How to utilize resources?
RELEASE
Scaling
VM
Language
Tools
Semi-explicit Placement
A programmer provides some criteria, and the rest is decided during the runtime
• Node attributes
• Communication distances
Node Attributes
Static
Dynamic
Attributes
ACO
Experimental Validation
Communication Distances
Measurements
Conclusion
Future Work
Sources
Portability Libraries
https://github.com/release-project/portability-libs
ACO
https://github.com/release-project/benchmarks/tree/master/ACO
RELEASE
http://www.release-project.eu/
SD Erlang
http://www.dcs.gla.ac.uk/research/sd-erlang/

Kenneth MacKenzie
Natalia Chechina
Phil Trinder

RELEASE

University of Glasgow

EDF

ERICSSON

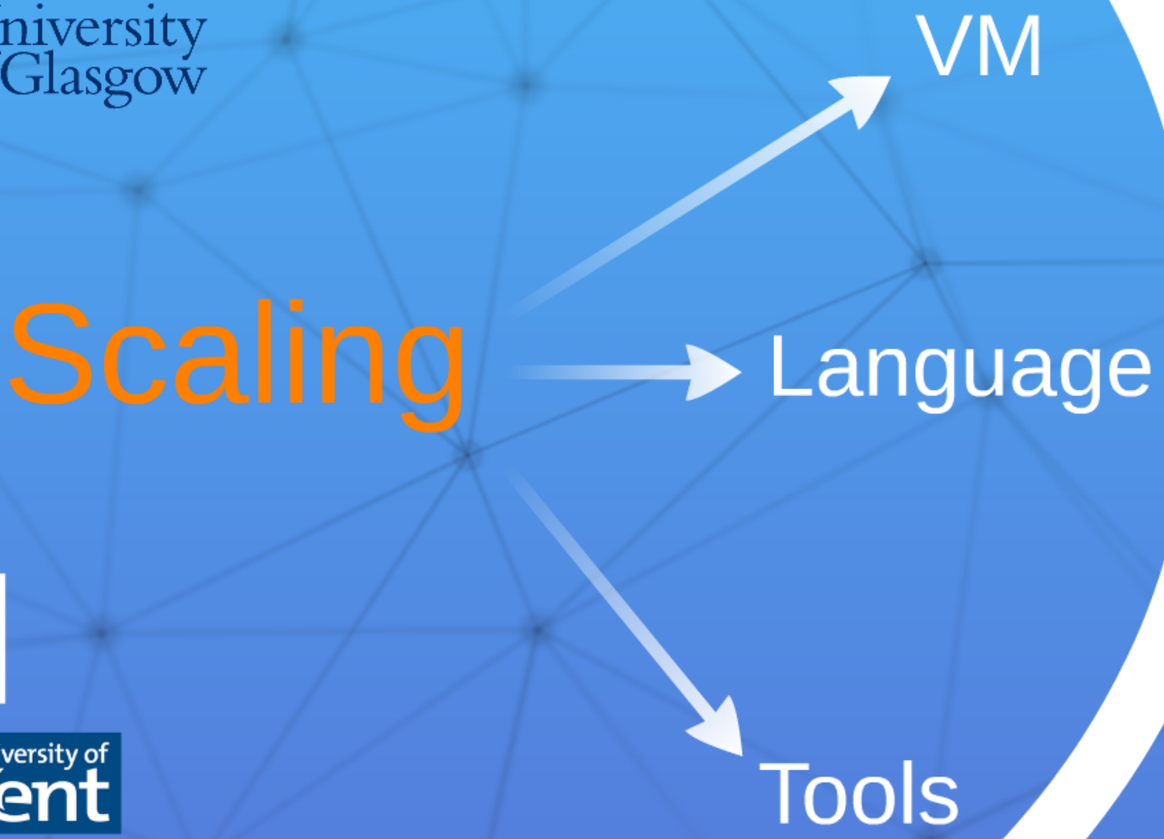
Erlang SOLUTIONS

ICCS

University of Kent

UPPSALA UNIVERSITET

HERIOT WATT UNIVERSITY



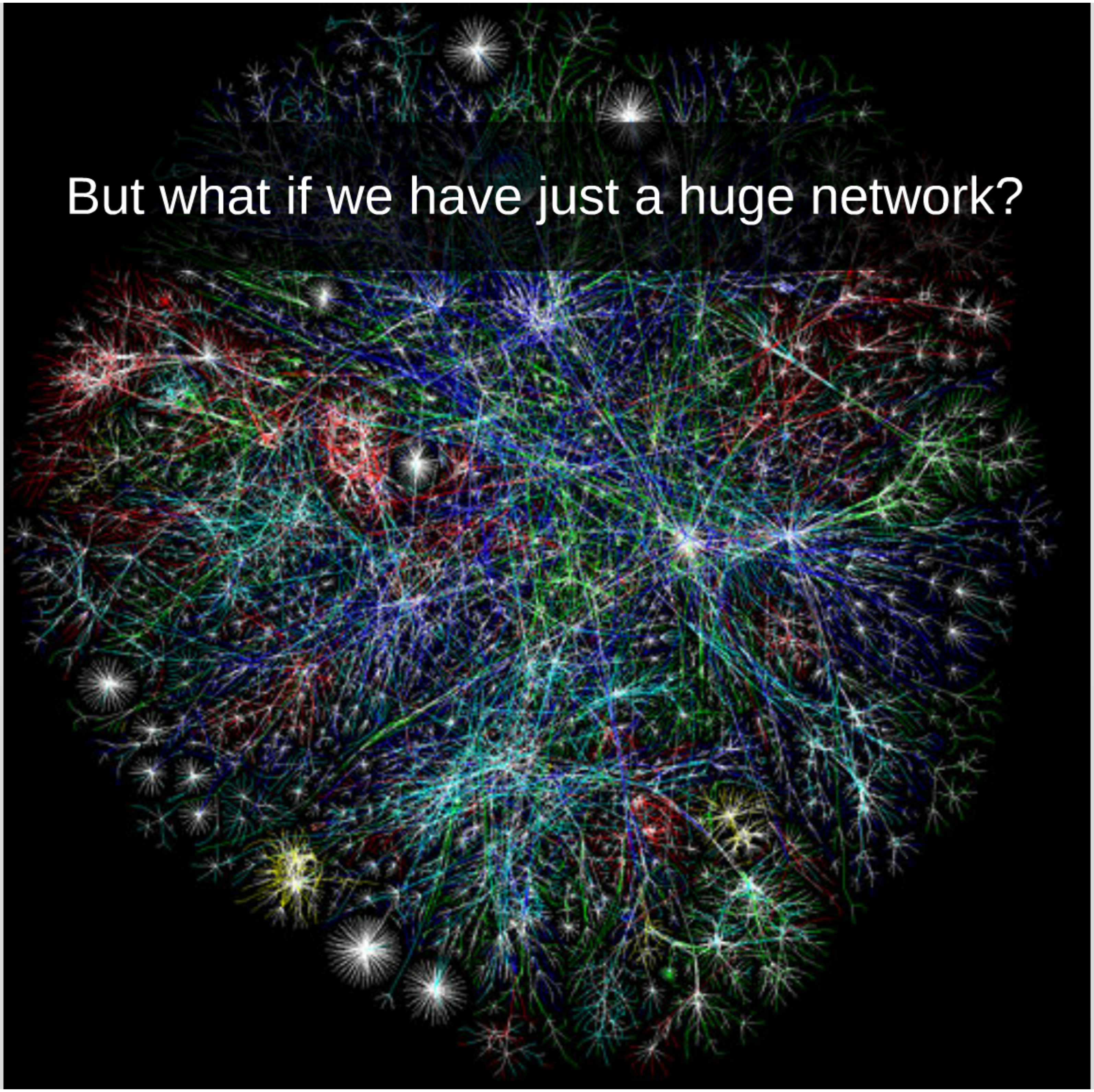
Language

How to utilize resources?

Small network -- Fine!

Structured network -- Great!

But what if we have just a huge network?



Semi-explicit Placement

A programmer provides some critetia, and the rest is decided during the runtime

- Node attributes
- Communication distances

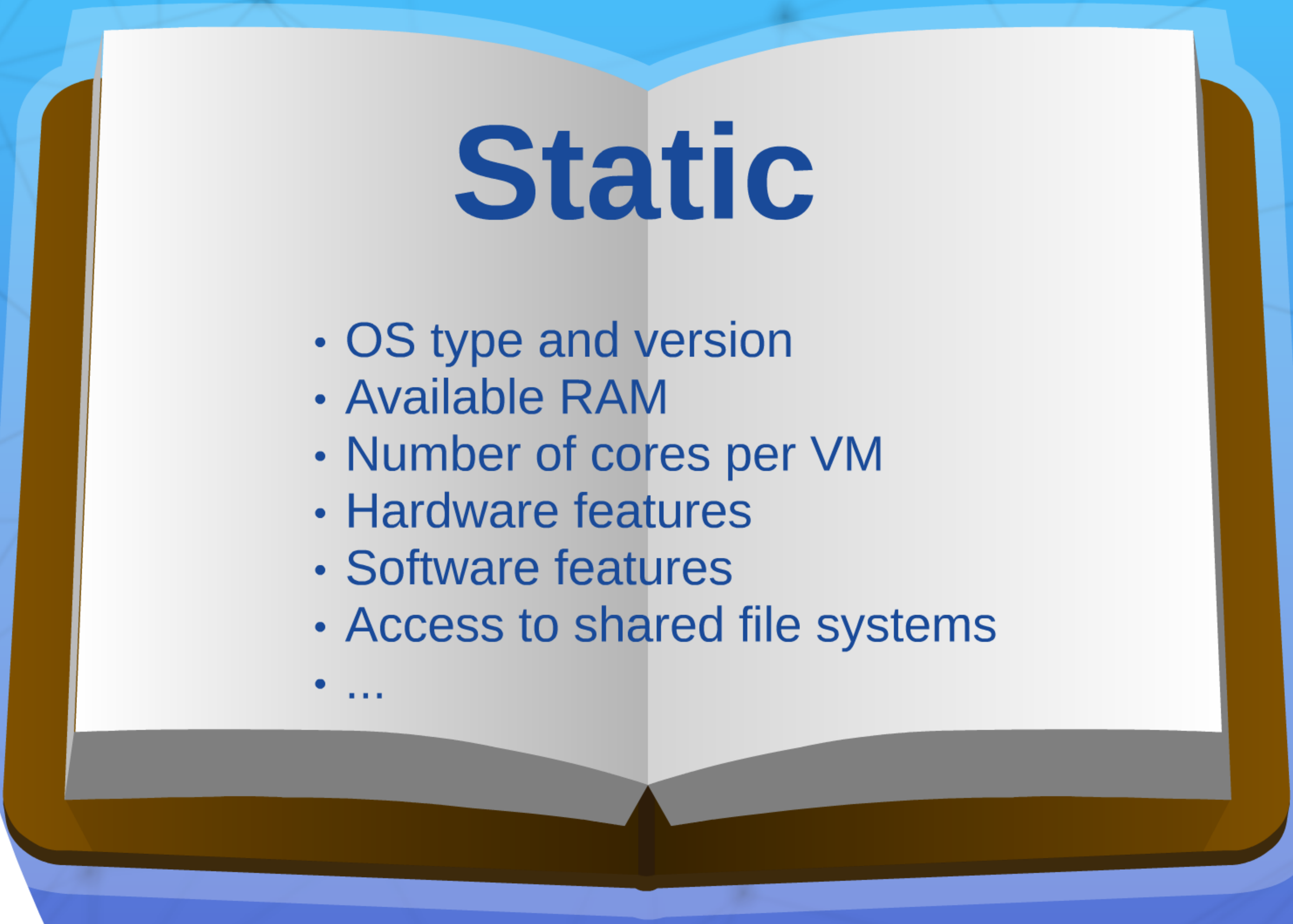
Node Attributes

Static

- OS type and version
- Available RAM
- Number of cores per VM
- Hardware features
- Software features
- Access to shared file systems
- ...

Dynamic

- Load on the machine
- Number of processes in the VM
- Available memory
- Types of running processes
- ...



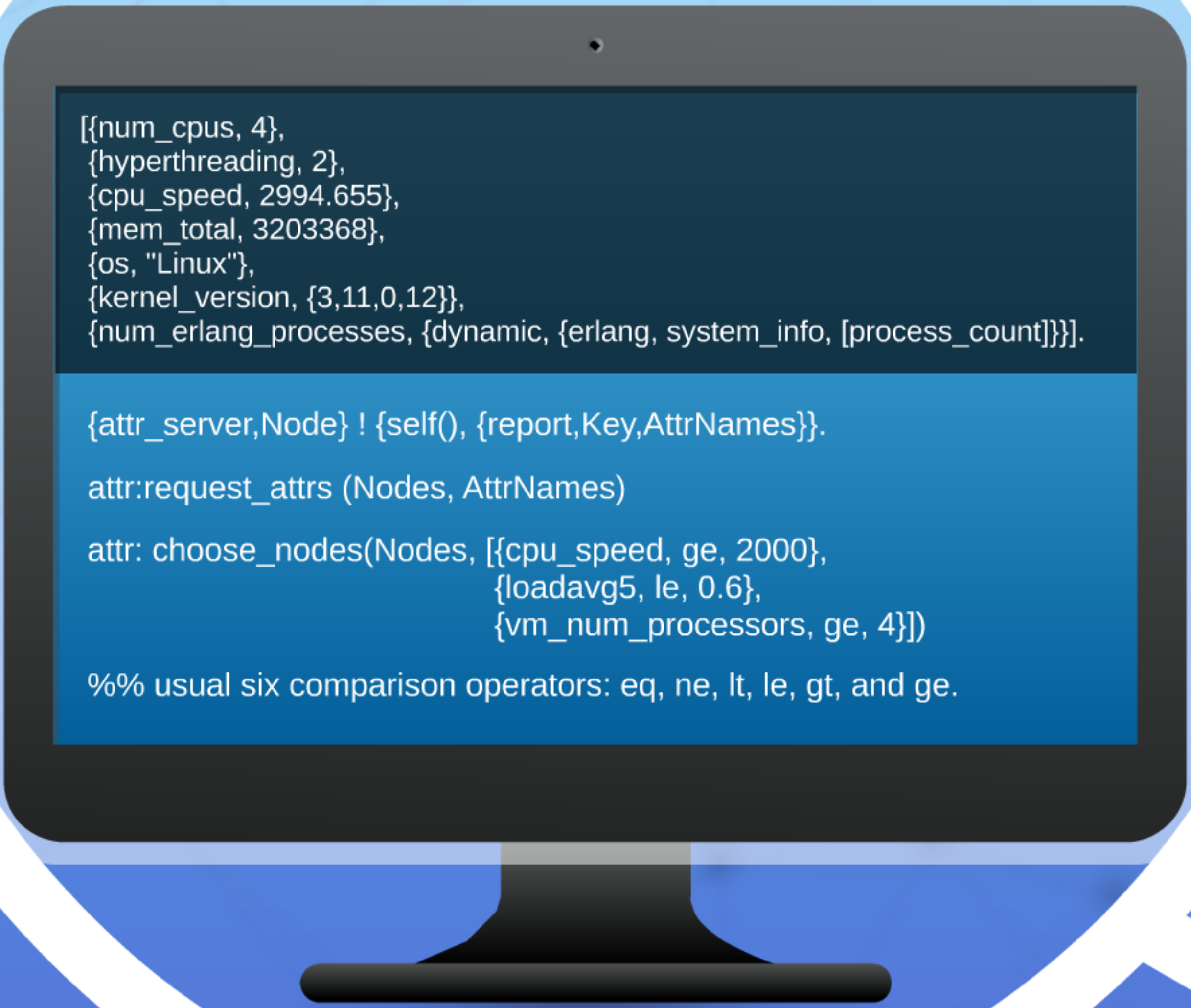
Static

- OS type and version
- Available RAM
- Number of cores per VM
- Hardware features
- Software features
- Access to shared file systems
- ...

Dynamic

- Load on the machine
- Number of processes in the VM
- Available memory
- Types of running processes
- ...

Attributes

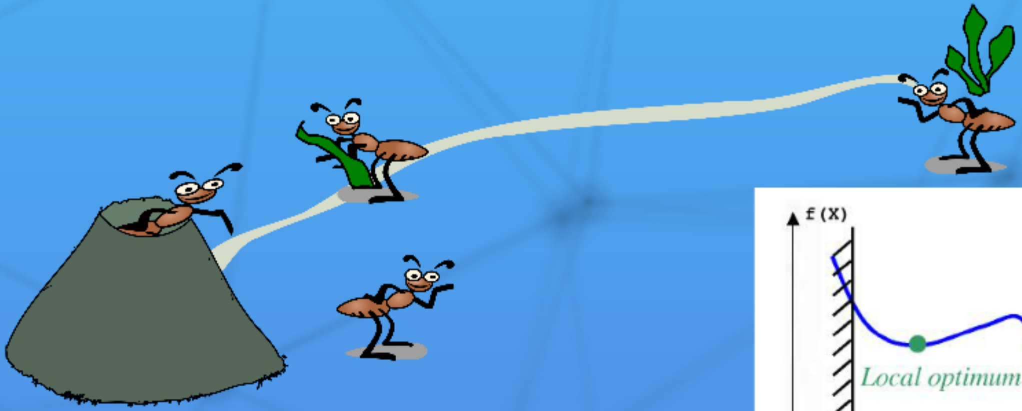


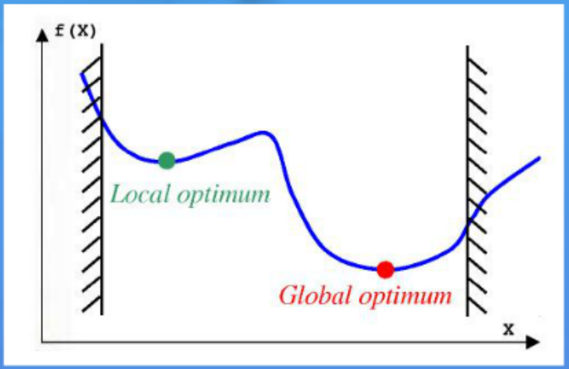
```
[{num_cpus, 4},
 {hyperthreading, 2},
 {cpu_speed, 2994.655},
 {mem_total, 3203368},
 {os, "Linux"},
 {kernel_version, {3,11,0,12}},
 {num_erlang_processes, {dynamic, {erlang, system_info, [process_count]}}}].
```

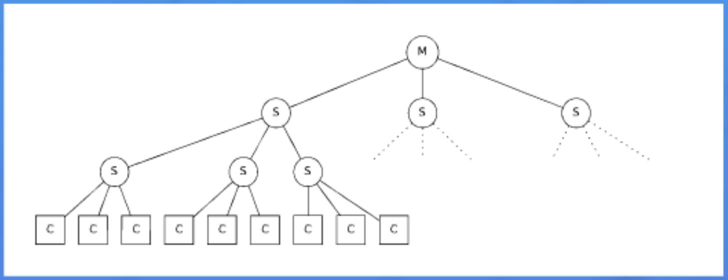
```
{attr_server,Node} ! {self(), {report,Key,AttrNames}}.

attr:request_attrs (Nodes, AttrNames)

attr: choose_nodes(Nodes, [{cpu_speed, ge, 2000},
                           {loadavg5, le, 0.6},
                           {vm_num_processors, ge, 4}])

%% usual six comparison operators: eq, ne, lt, le, gt, and ge.
```

ACO

Ant Colony Optimization



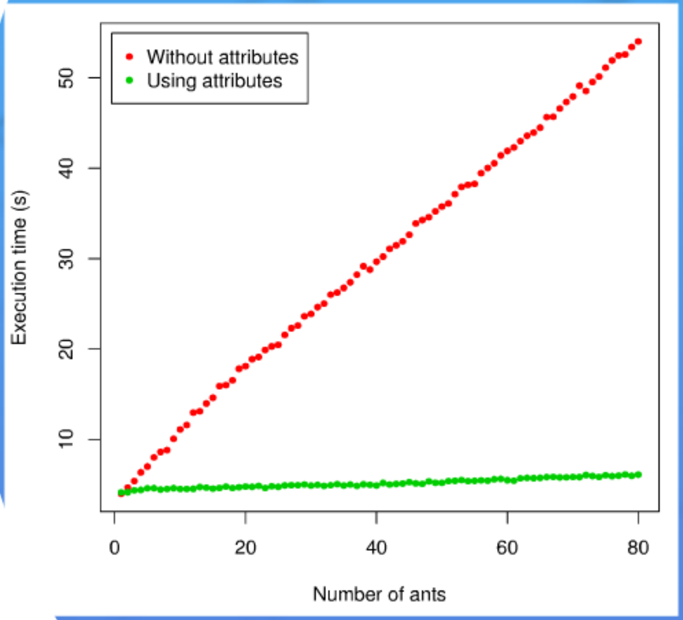


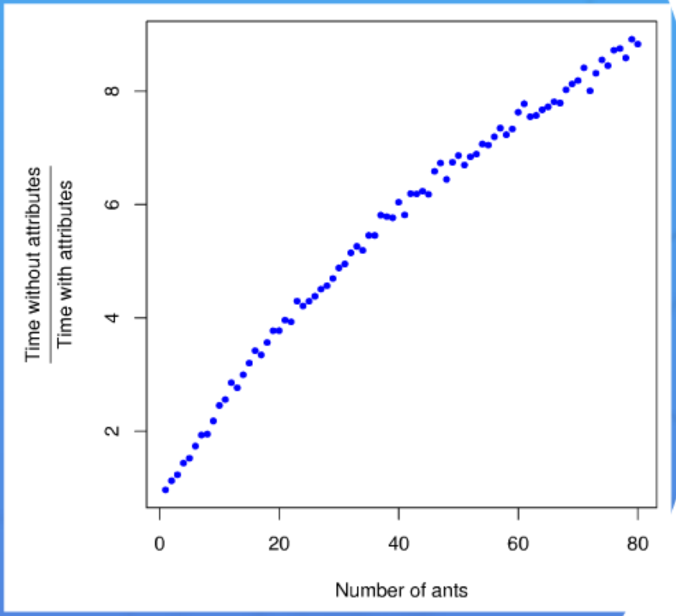


Related Problems:

- Traveling Salesman Problem
- Vehicle Routing

Experimental Validation





Attribute Propagation Strategy

- On request
- Broadcasting

Request info
from VM and OS

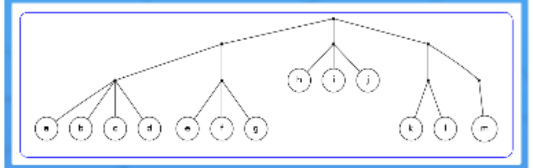
Types

- Built-in
- Customized

Combining with SD Erlang s_groups

- s_group:own_nodes/0
- s_group:own_s_groups/0
- global:own_nodes/0

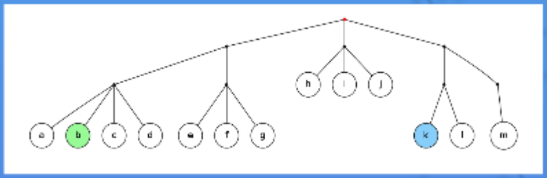
Communication Distances



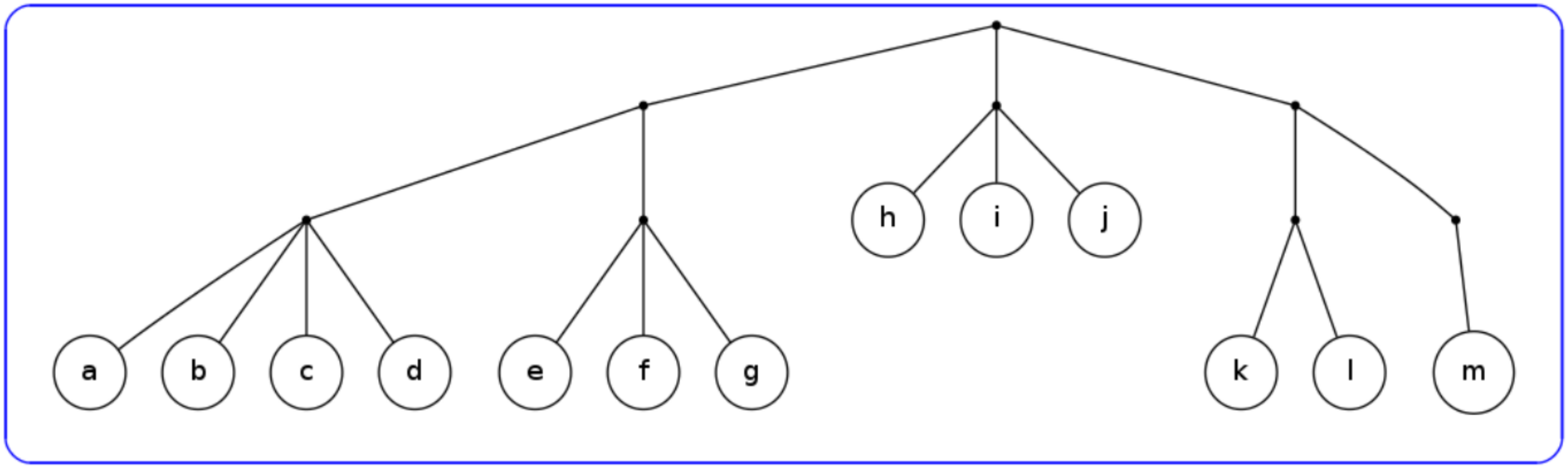
$d : X \times X \to \mathbb{R}^+ = \{x \in \mathbb{R} : x \geq 0\}$

(i) $d(x, y) = 0$ if and only if $x = y$
(ii) $d(x, y) = d(y, x)$
(iii) $d(x, z) \leq d(x, y) + d(y, z)$

$$[d(x, y) = \begin{cases} 0 & \text{if } x = y \\ 2^{-\ell(x,y)} & \text{if } x \neq y. \end{cases}]$$



$d(b, k) = 2^{-0} = 1$



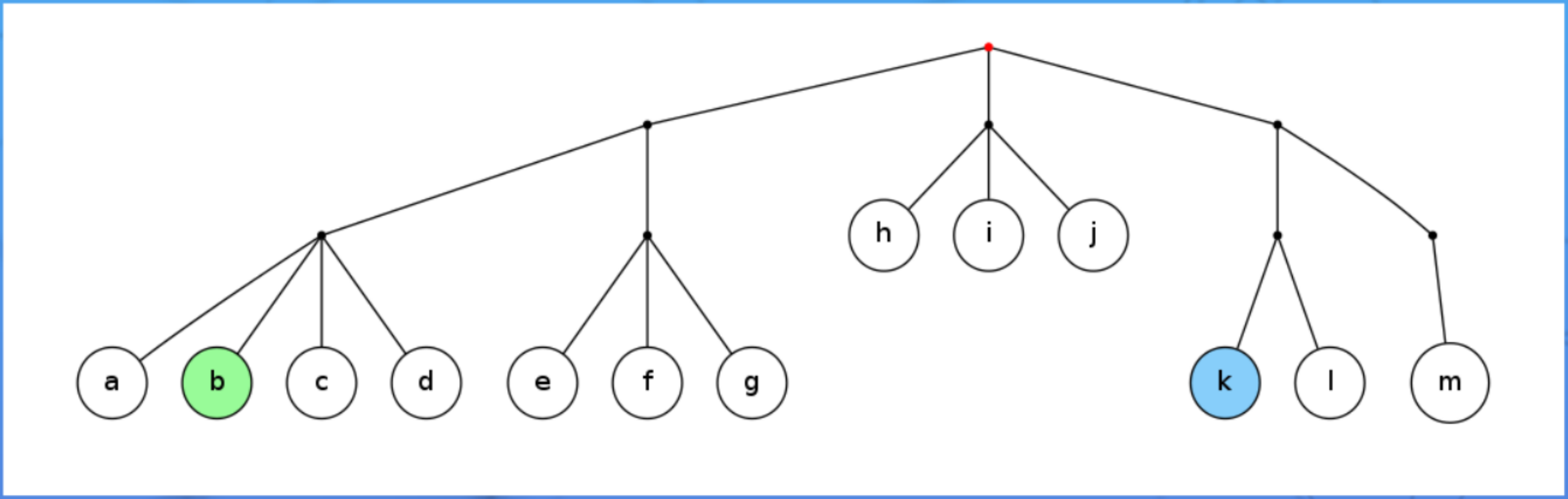
$$d : X \times X \to \mathbb{R}^+ = \{x \in \mathbb{R} : x \geq 0\}$$

(i) $d(x, y) = 0$ if and only if $x = y$

(ii) $d(x, y) = d(y, x)$

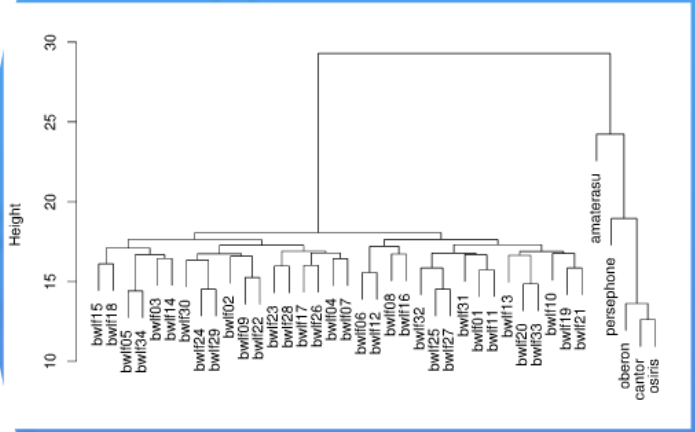
(iii) $d(x, z) \leq d(x, y) + d(y, z)$

$$[d(x,y) = \begin{cases} 0 & \text{if } x = y \\ 2^{-\ell(x,y)} & \text{if } x \neq y. \end{cases}]$$

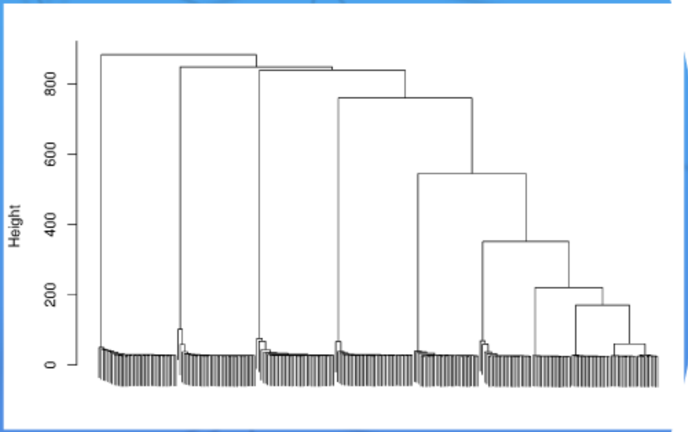


$$d(b,k) = 2^{-0} = 1$$

Measurements



Heriot-Watt cluster



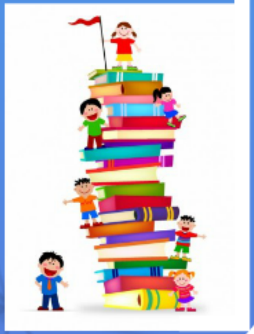
Athos cluster

Conclusion

Semi-explicit placement → Node attributes / Communication distance → attr:choose_nodes(Nodes, Pars) ✓

Future Work

- Concrete and abstract bounds
- Conflicting constraints
- Avoiding clashes when spawning processes
- Fault tolerance
- Dynamic changes to network structure



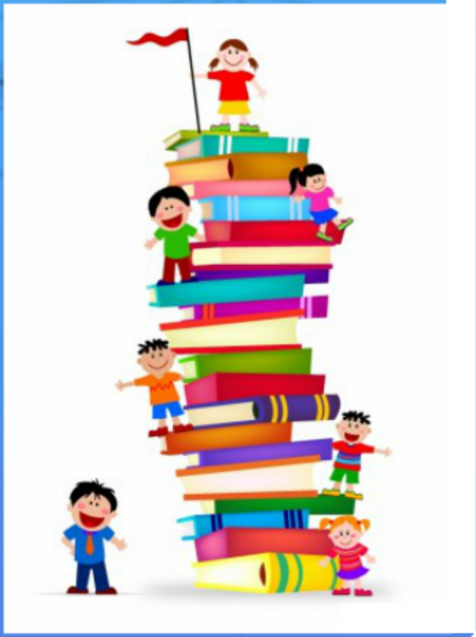
Conclusion

Semi-explicit placement → Node attributes / Communication distance → attr:choose_nodes(Nodes, Pars) ✓

Future Work

Future Work

- Concrete and abstract bounds
- Conflicting constraints
- Avoiding clashes when spawning processes
- Fault tolerance
- Dynamic changes to network structure



Sources

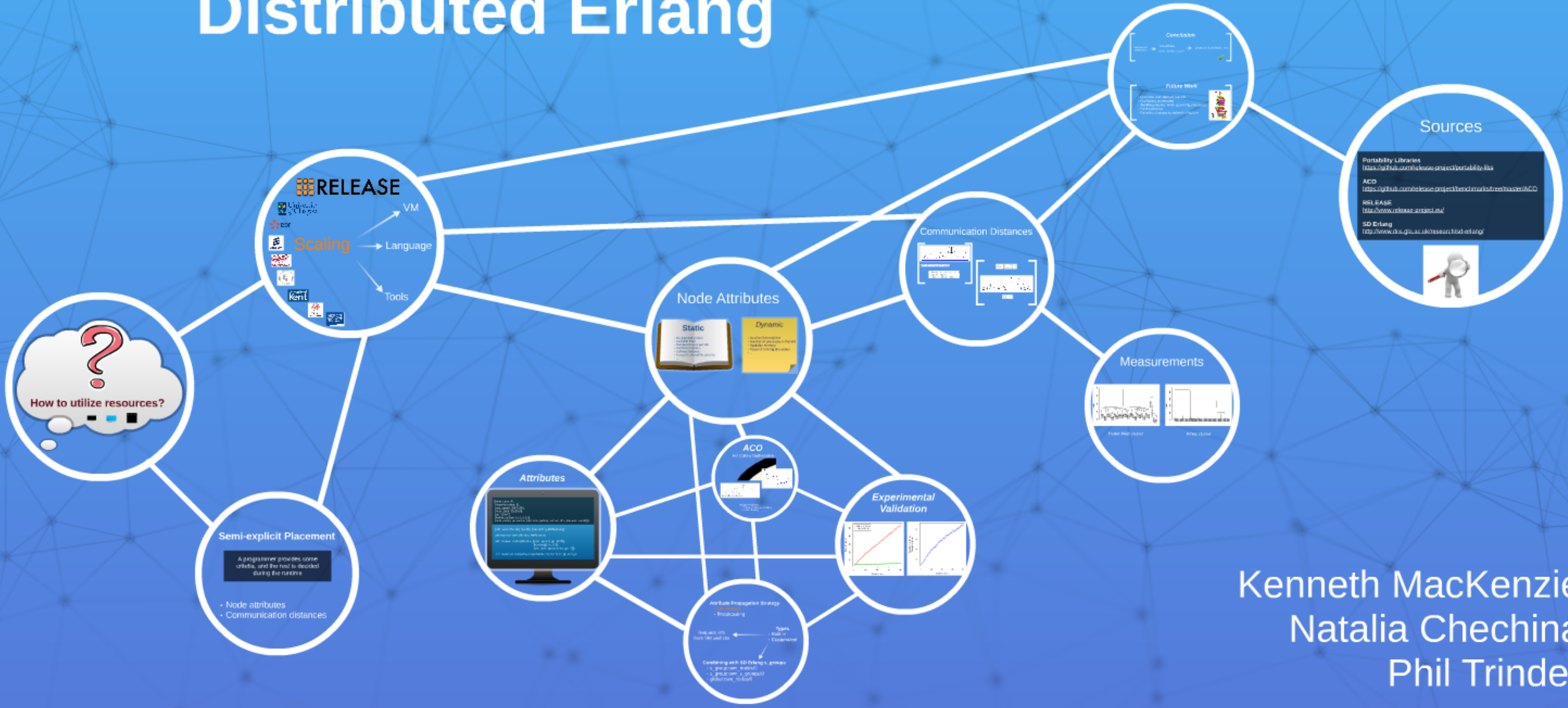
Portability Libraries
https://github.com/release-project/portability-libs

ACO
https://github.com/release-project/benchmarks/tree/master/ACO

RELEASE
http://www.release-project.eu/

SD Erlang
http://www.dcs.gla.ac.uk/research/sd-erlang/

Performance Portability through Semi-explicit Placement in Distributed Erlang


RELEASE
Scaling
VM
Language
Tools
How to utilize resources?
Semi-explicit Placement
A programmer provides some criteria, and the rest is decided during the runtime
• Node attributes
• Communication distances
Node Attributes
Static
Dynamic
Attributes
ACO
Experimental Validation
Communication Distances
Measurements
Conclusion
Future Work
Sources
Portability Libraries
https://github.com/release-project/portability-libs
ACO
https://github.com/release-project/benchmarks/tree/master/ACO
RELEASE
http://www.release-project.eu/
SD Erlang
http://www.dcs.gla.ac.uk/research/sd-erlang/

Kenneth MacKenzie
Natalia Chechina
Phil Trinder