ASTRA: A Tool for Abstract Interpretation of Graph Transformation Systems

Peter Backes Jan Reineke

Saarland University
Saarbrücken, Germany

Graph Transformation Systems

Graph Transformation System =

Start Graph + Set of Rules

Graph Transformation Systems

Graph Transformation System =

Start Graph + Set of Rules

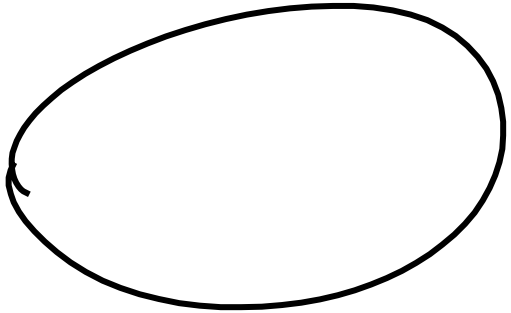
Rule: 

Graph Transformation Systems

Graph Transformation System =

Start Graph + Set of Rules

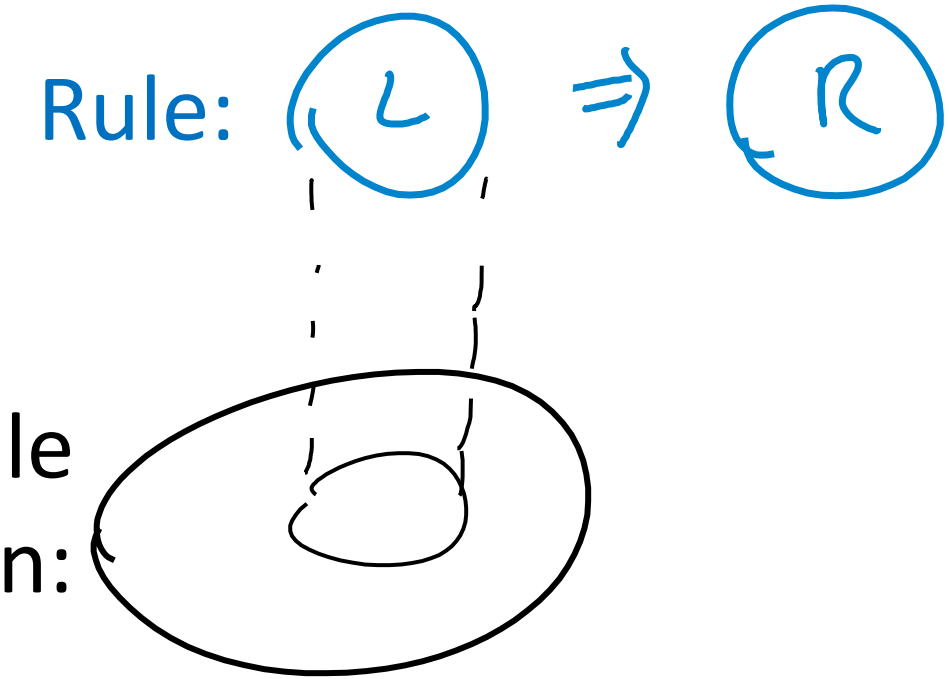
Rule: 

Rule Application: 

Graph Transformation Systems

Graph Transformation System =

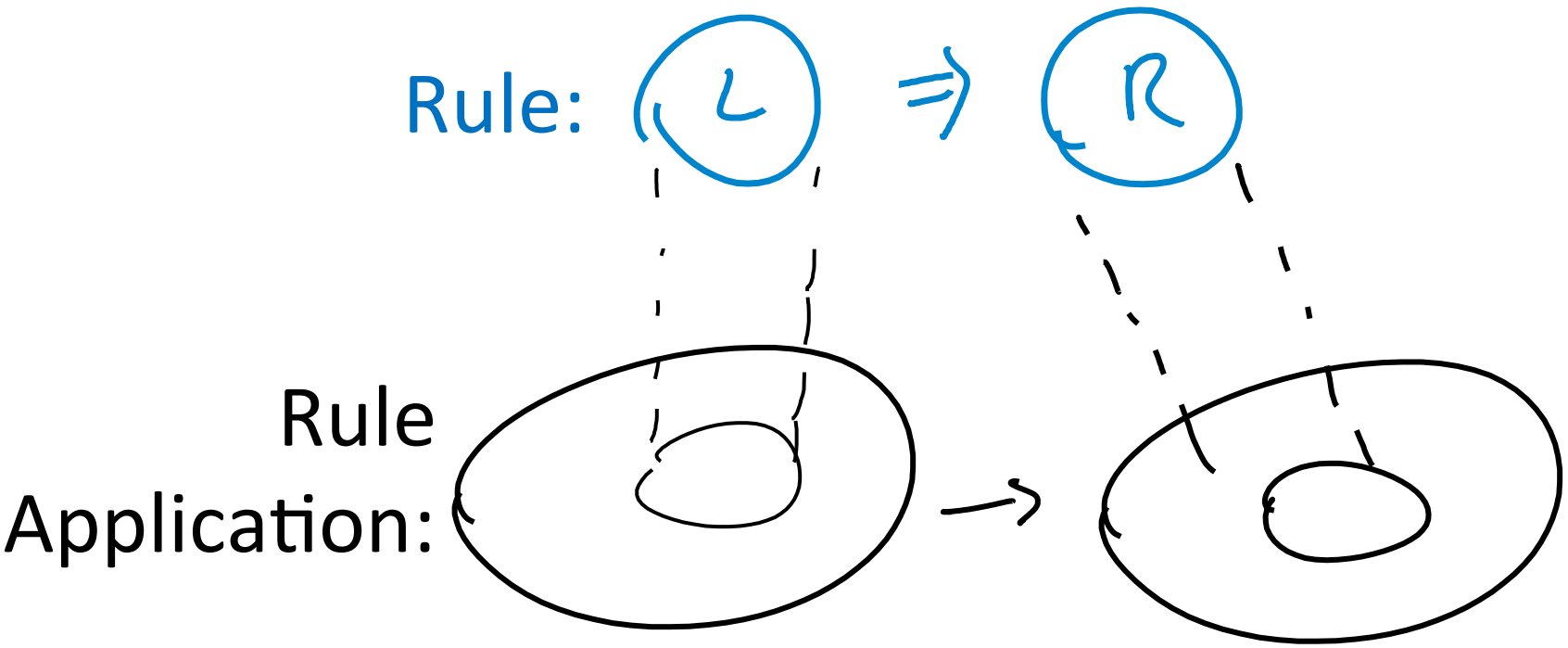
Start Graph + Set of Rules



Graph Transformation Systems

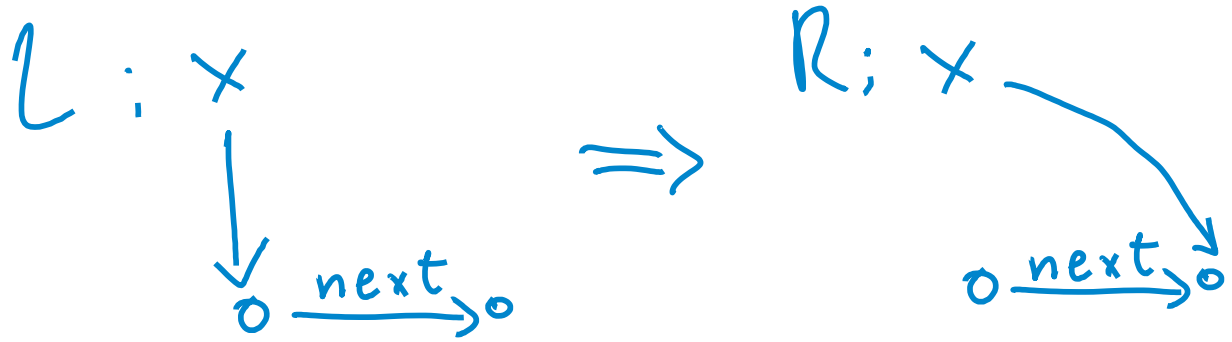
Graph Transformation System =

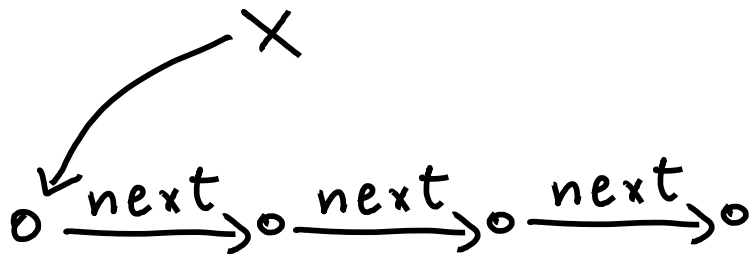
Start Graph + Set of Rules



Example: List Traversal

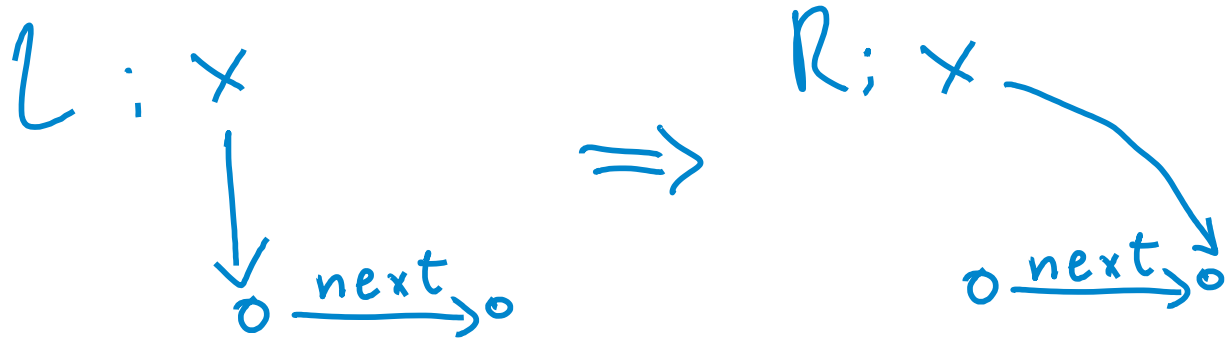
Rule:

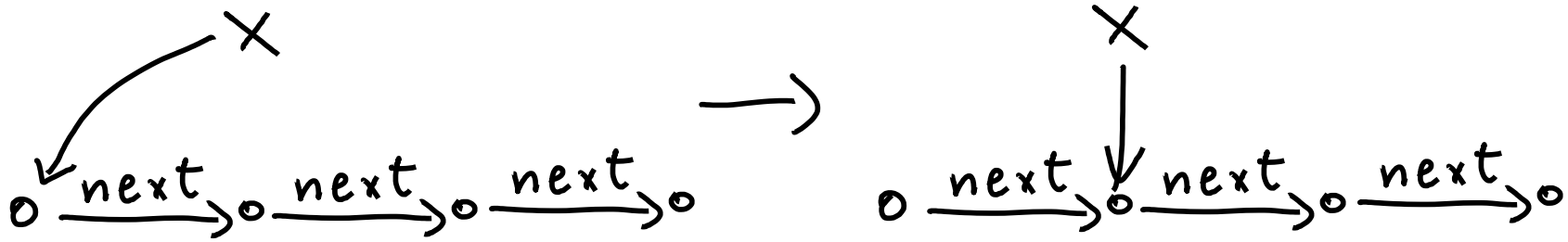




Example: List Traversal

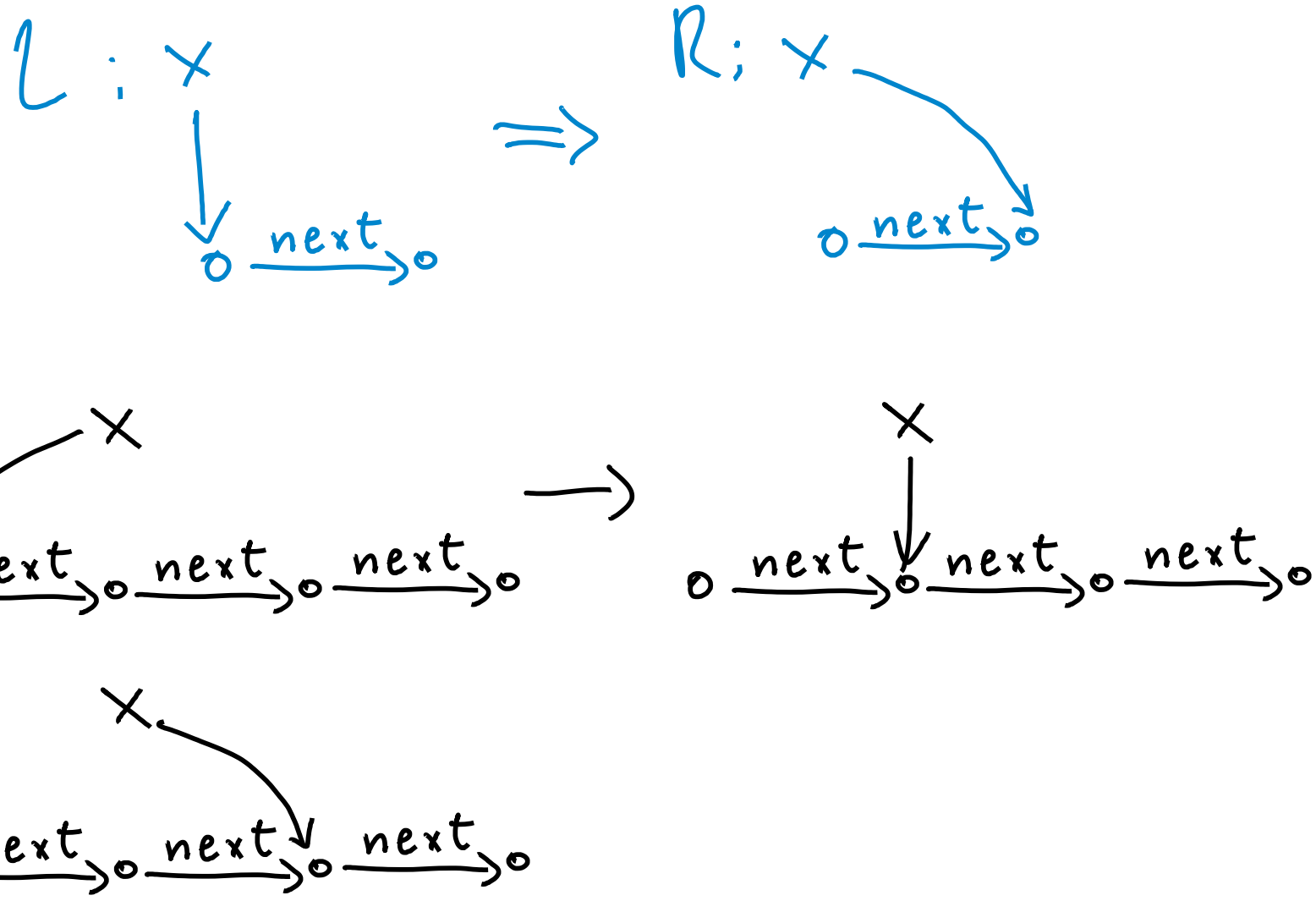
Rule:





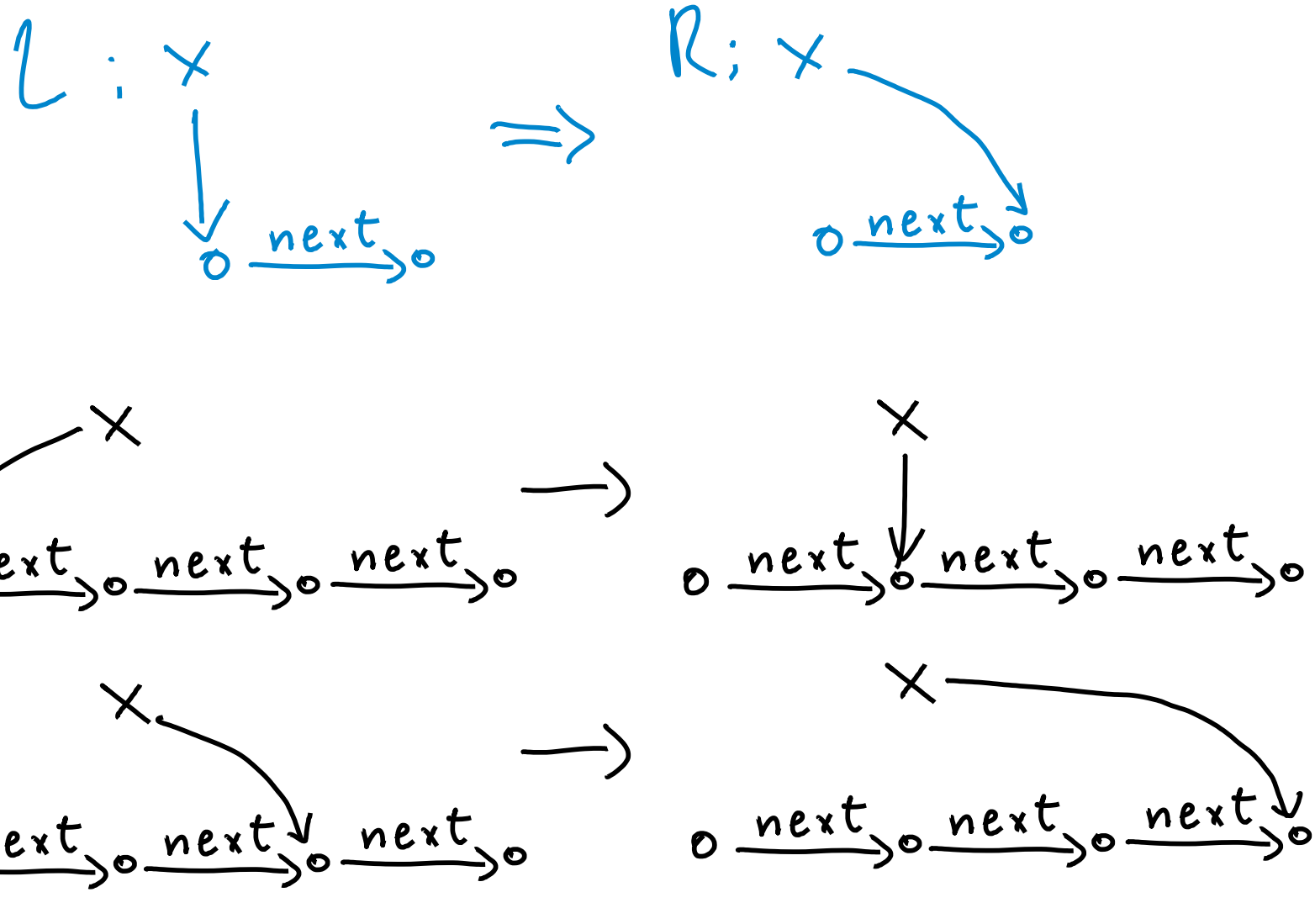
Example: List Traversal

Rule:

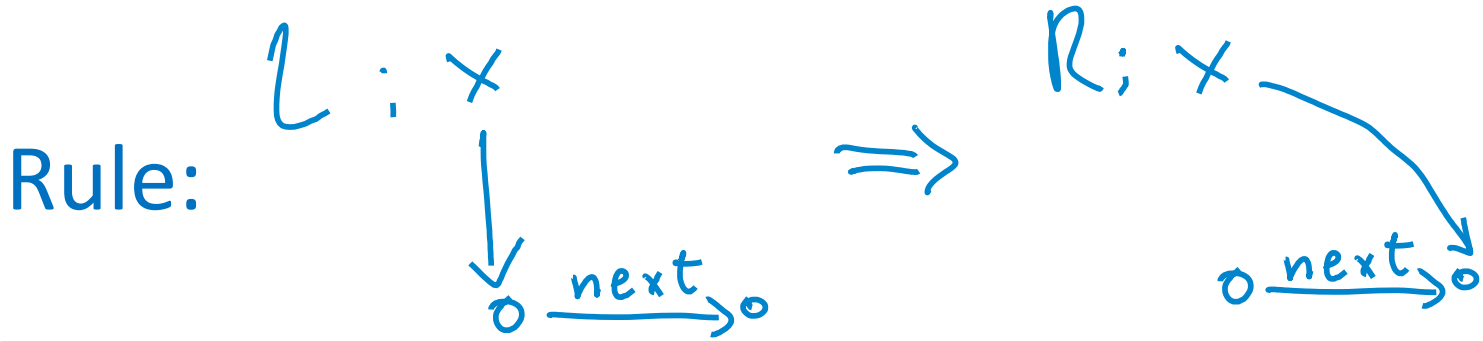


Example: List Traversal

Rule:

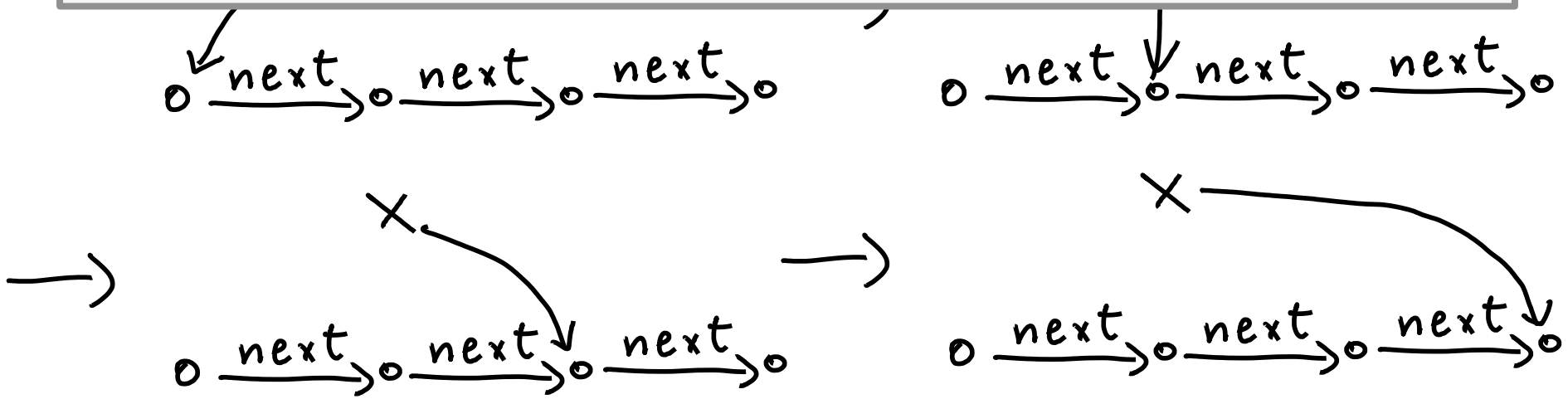


Example: List Traversal

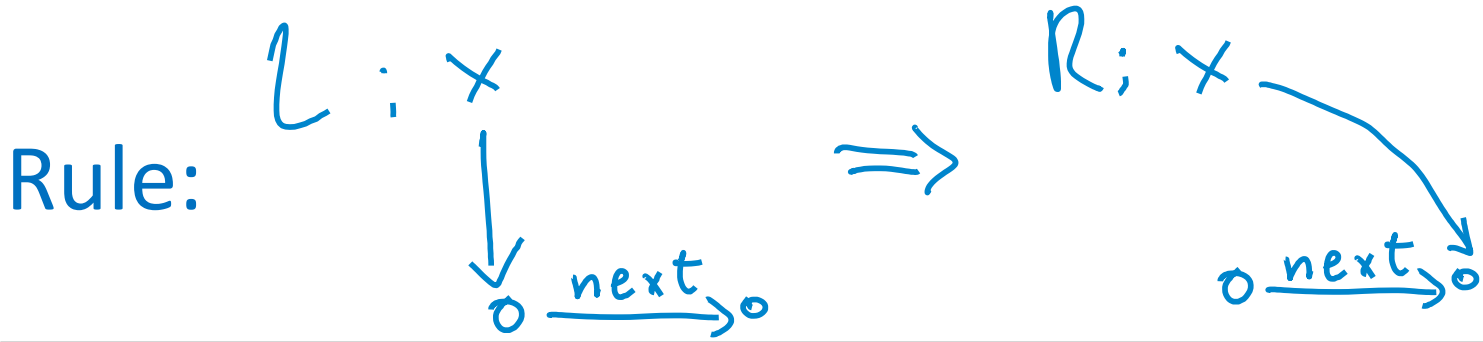


Semantics of Graph Transformation System =

All graphs reachable from start graph via rule applications

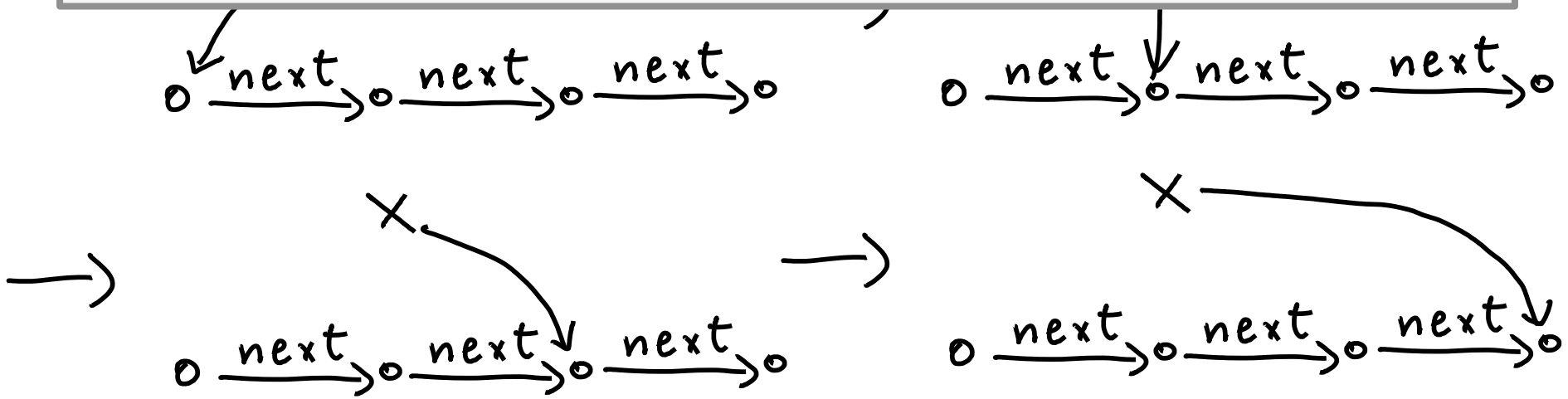


Example: List Traversal



Semantics of Graph Transformation System =

All graphs reachable from start graph via rule applications



Graph Transformation

Useful to model many kinds of computations:

- dynamic message-passing systems,
- biological processes,
- business processes,
- heap-manipulating programs, etc.

Our Main Area of Application: Dynamic Message-Passing Systems

Characteristics:

Our Main Area of Application: Dynamic Message-Passing Systems

Characteristics:

- Unbounded number of concurrent processes

Our Main Area of Application: Dynamic Message-Passing Systems

Characteristics:

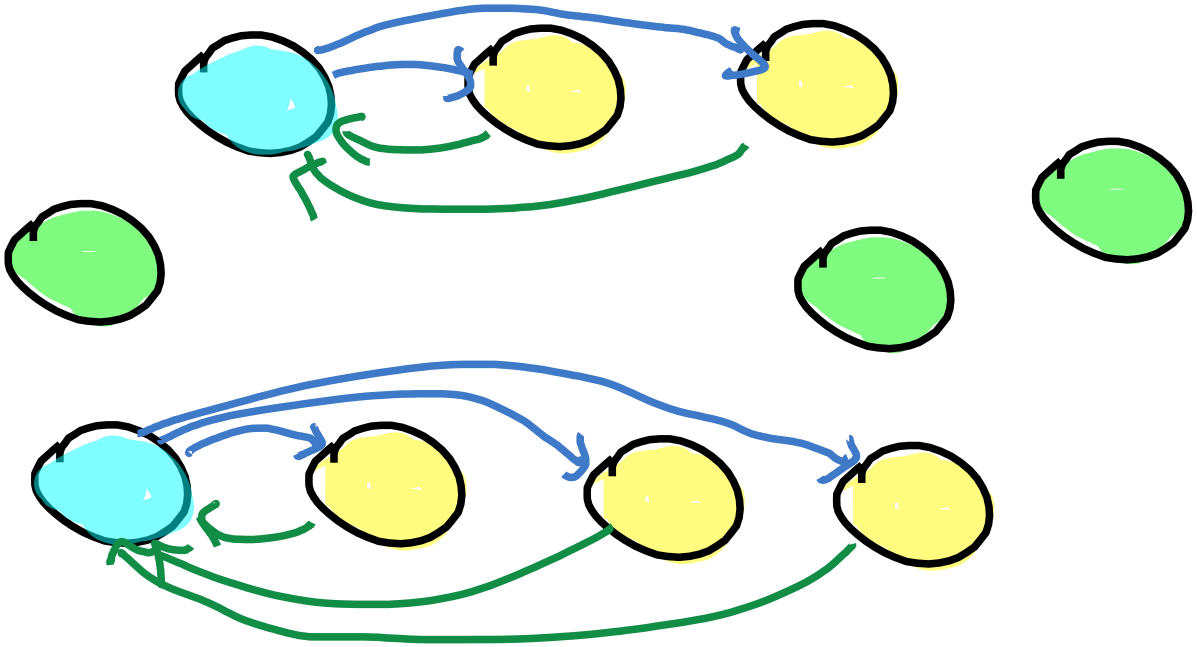
- Unbounded number of concurrent processes
- Processes dynamically construct and destruct communication topologies

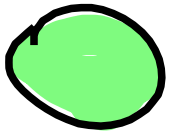
Our Main Area of Application: Dynamic Message-Passing Systems

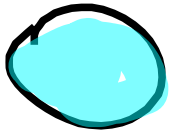
Characteristics:

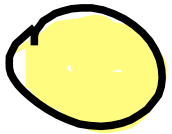
- Unbounded number of concurrent processes
- Processes dynamically construct and destruct communication topologies
- Each process executes a finite state protocol; makes transitions based on its local view of the system state

Example: Car Platooning

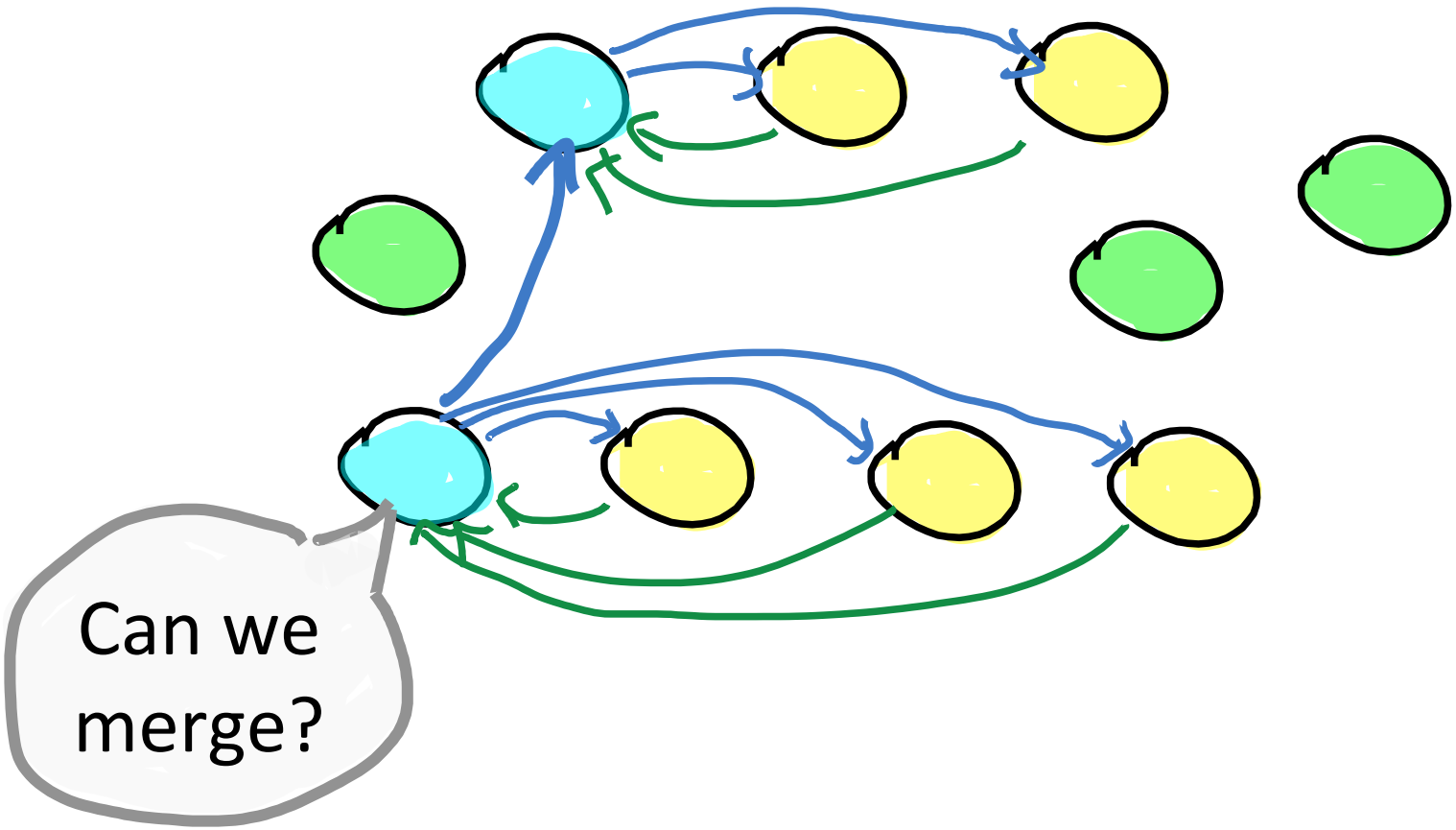


Free Agent = 

Platoon Leader = 

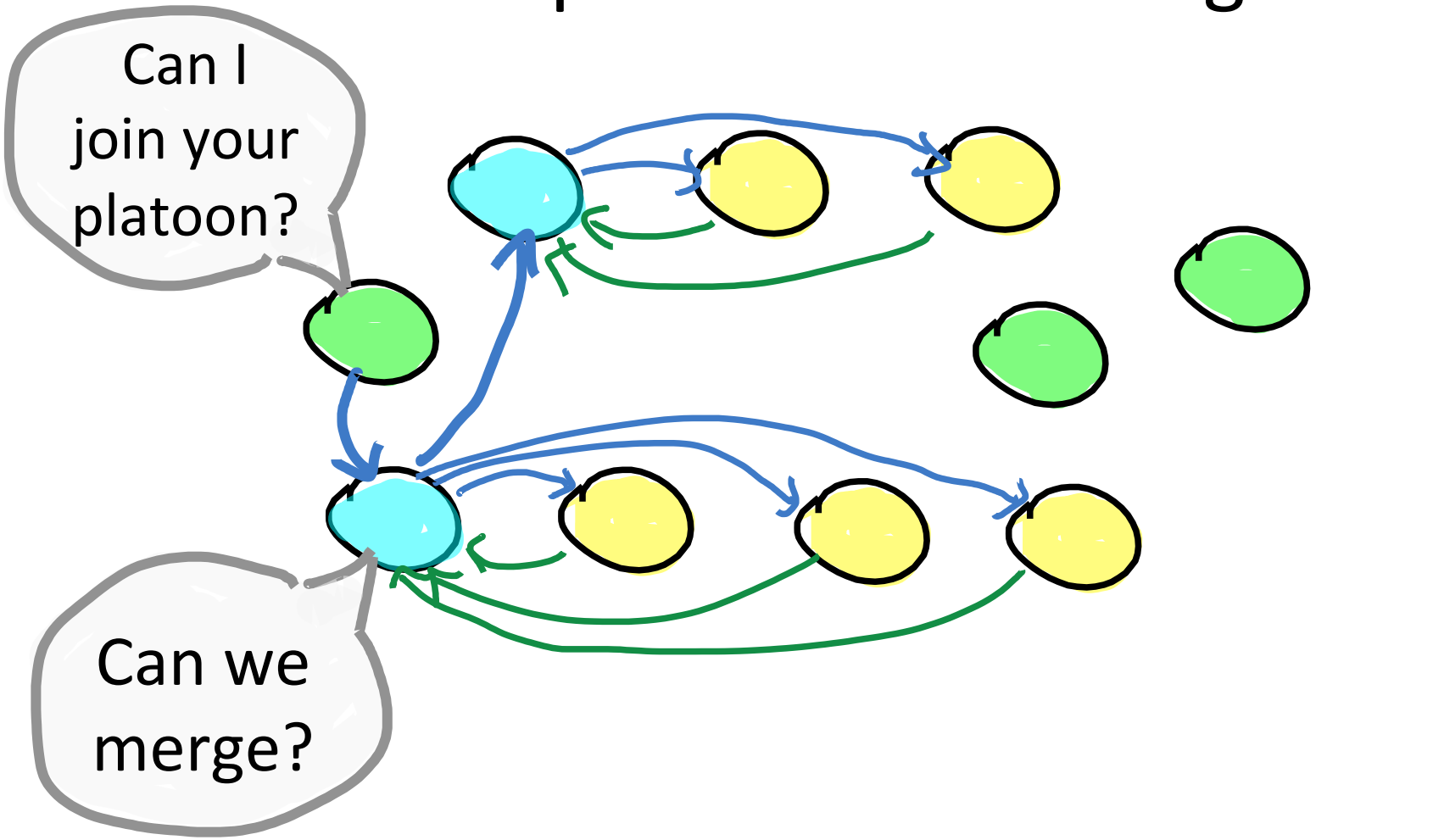
Follower = 

Example: Car Platooning

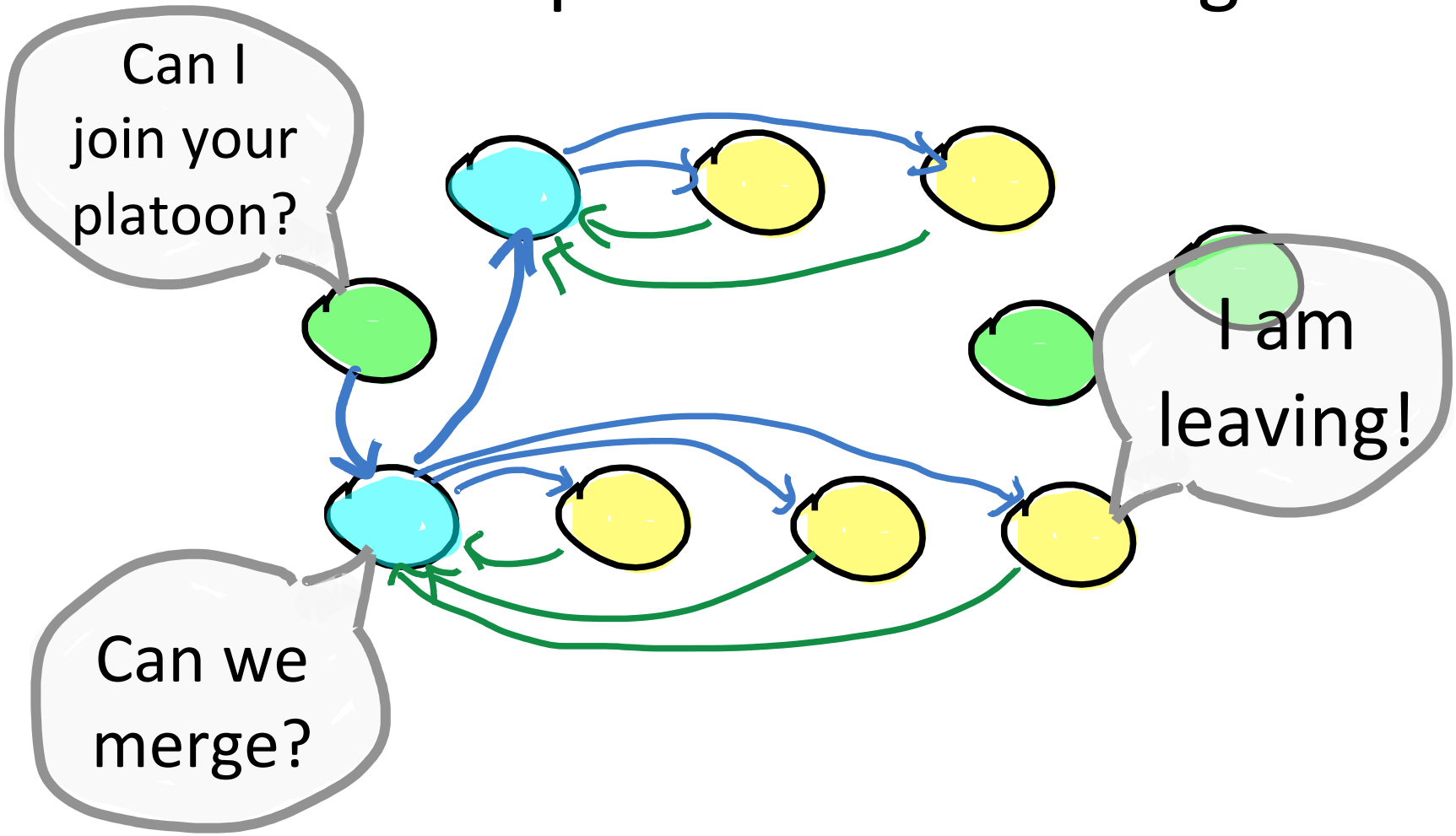


Free Agent = Platoon Leader = Follower =

Example: Car Platooning



Example: Car Platooning

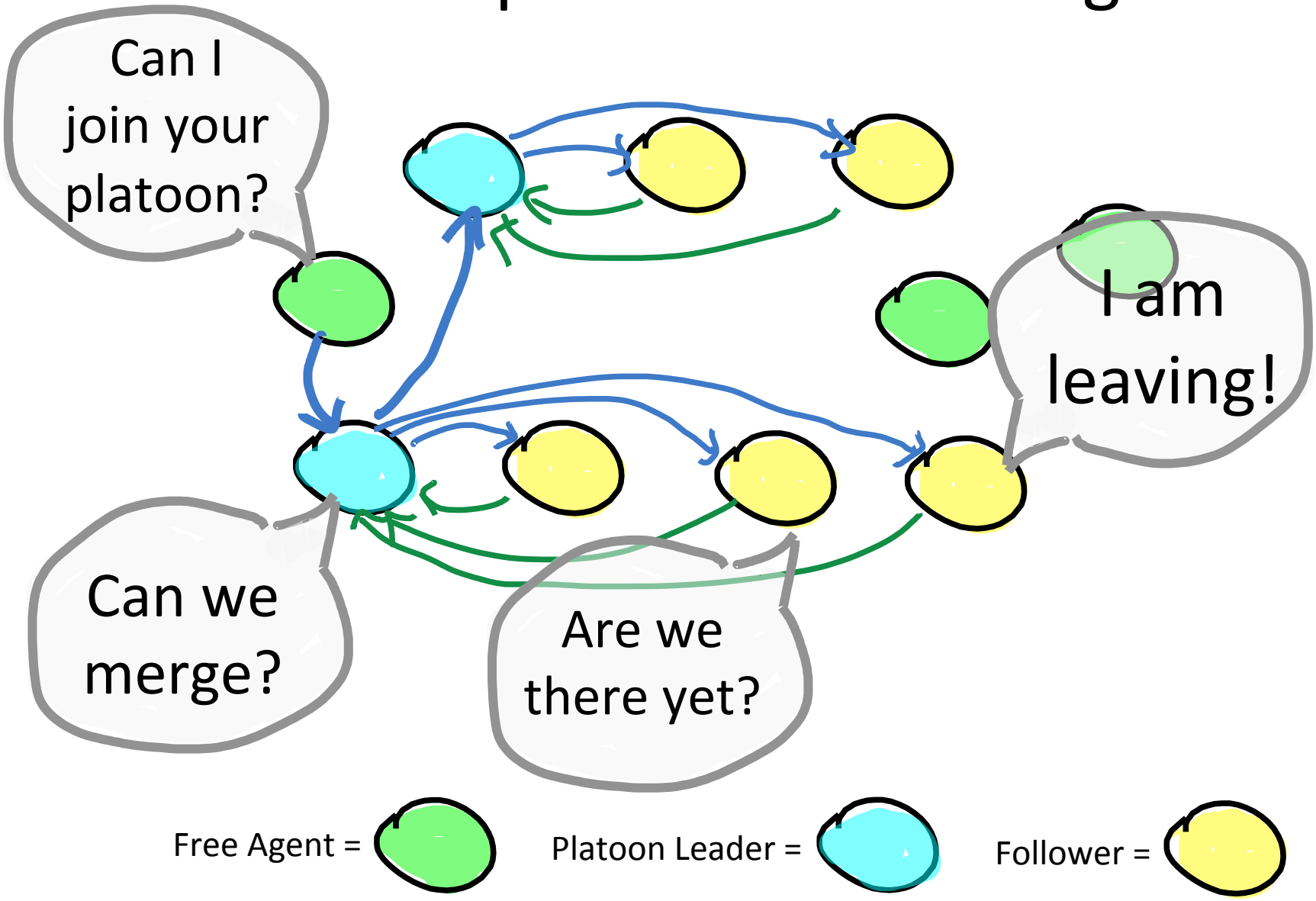


Free Agent =

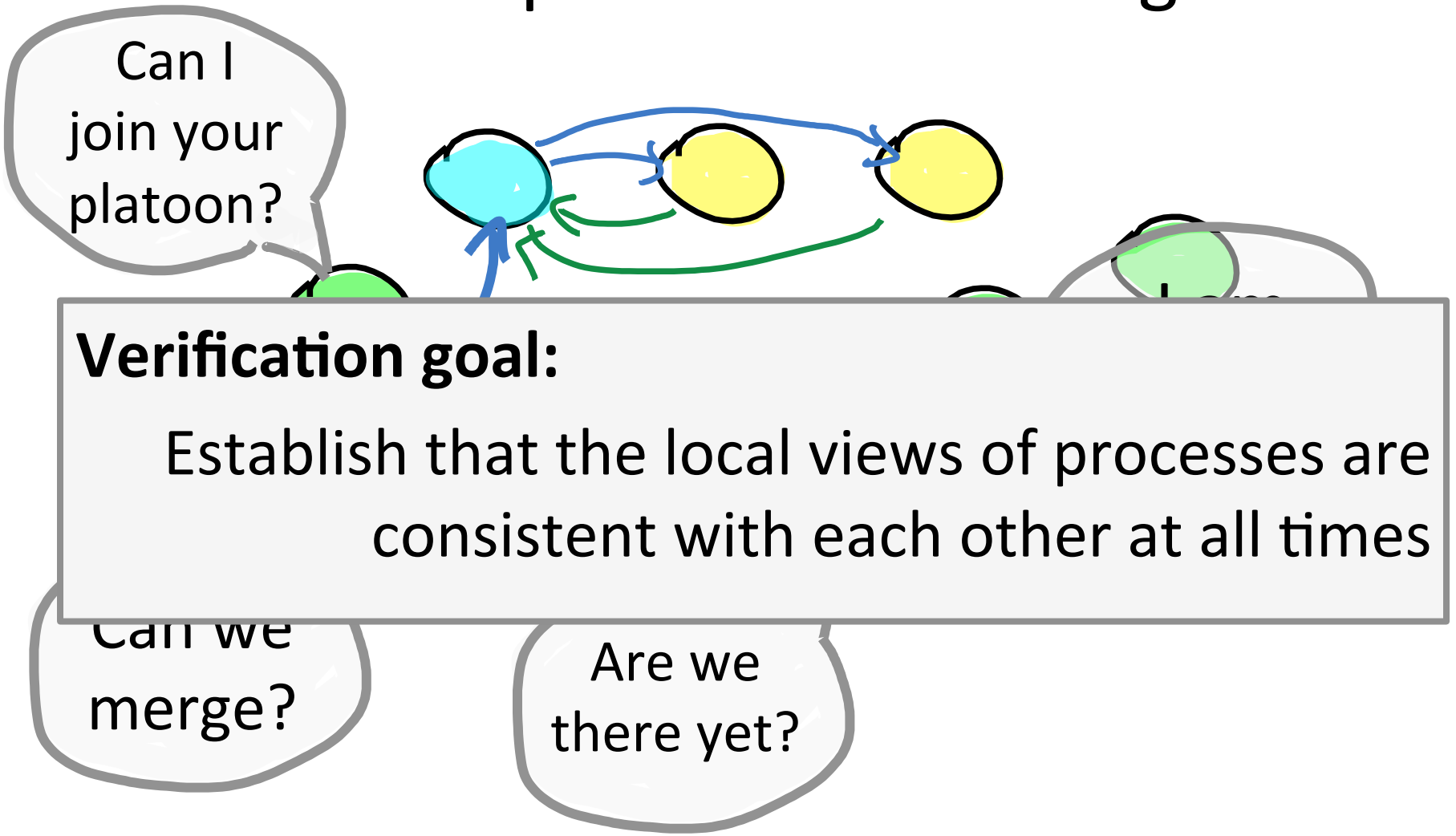
Platoon Leader =

Follower =

Example: Car Platooning



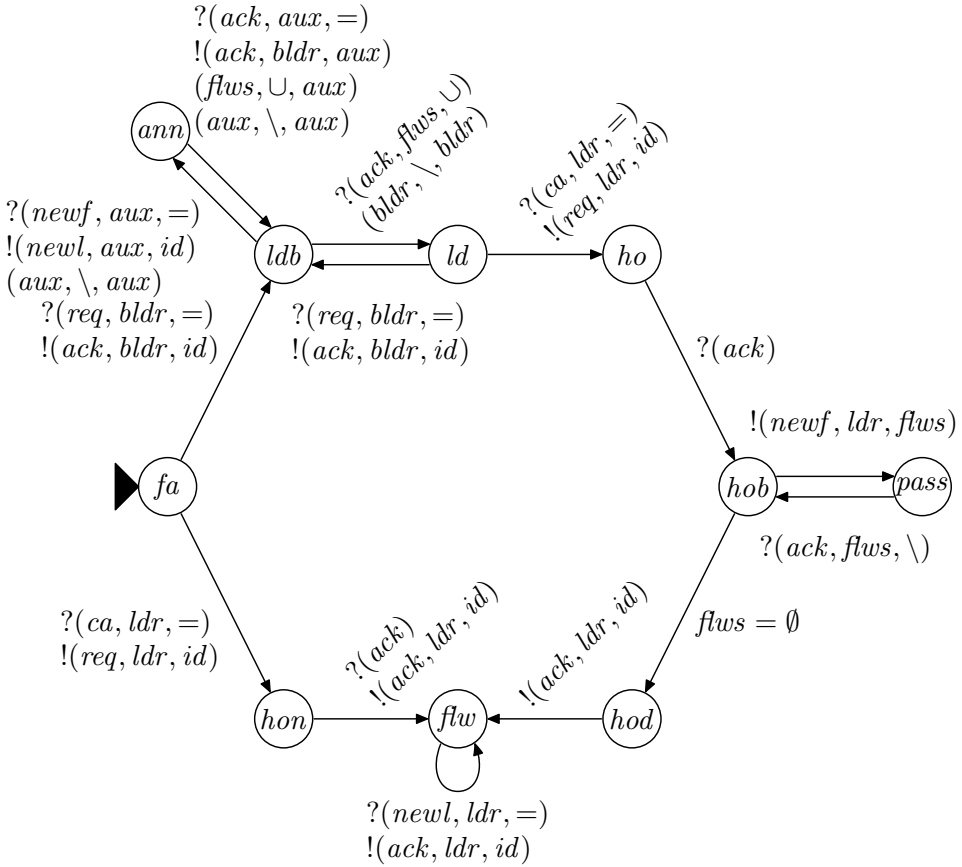
Example: Car Platooning



Example:

Merge Protocol from Car Platooning

All processes concurrently execute the following protocol + new processes may appear arbitrarily:



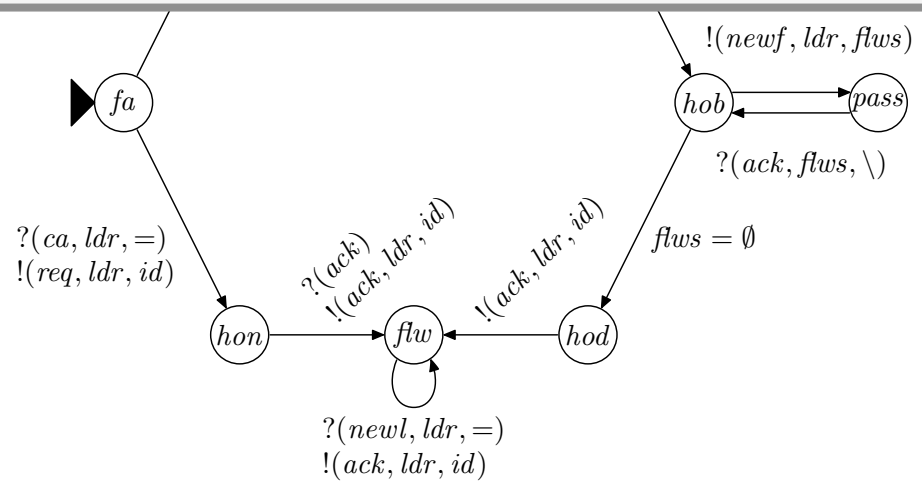
Example:

Merge Protocol from Car Platooning

All processes concurrently execute the following protocol + new processes may appear arbitrarily:

Simple translation into graph transformation system:

- One rule for each transition in the automaton
- One rule for process creation



Cluster Abstraction

Assumption:

Protocols designed by humans establish invariants that are “local in nature”.

Cluster Abstraction

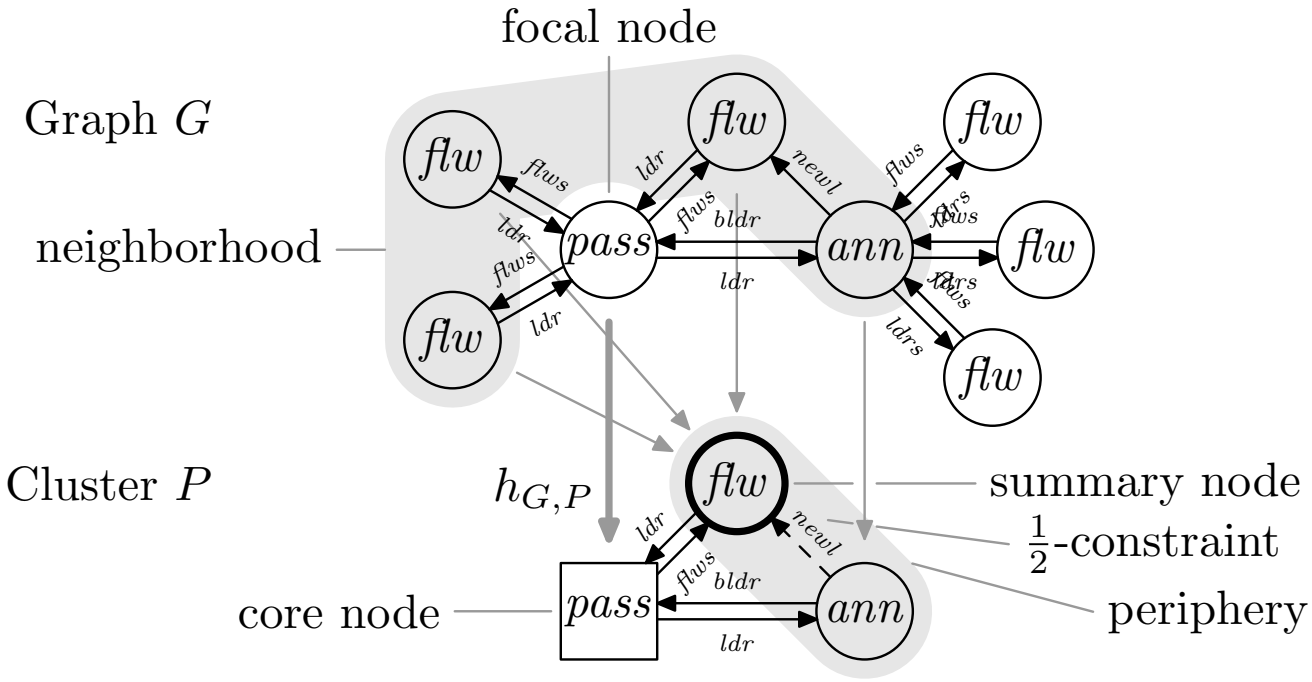
Assumption:

Protocols designed by humans establish invariants that are “local in nature”.

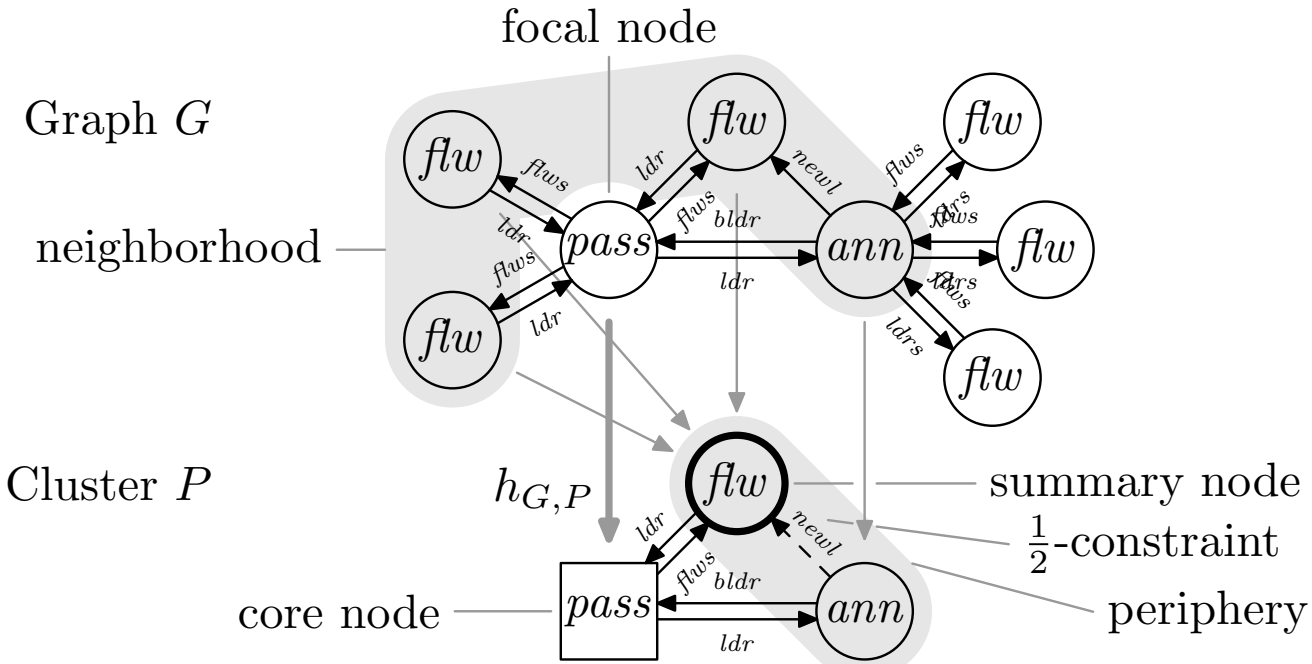
Cluster Abstraction:

Maintain for each process only its immediate environment.

Cluster Abstraction: Example



Cluster Abstraction: Example



Challenge:

Computing (best) abstract transformer for cluster abstraction

See [VMCAI 2015] for more details.

Astra 2.0

Implements cluster abstraction

- Input: Graph Transformation System S
- Output: Set of clusters overapproximating semantics of S

Free Software, available here:

http://www.rw.cdl.uni-saarland.de/~rtc/astra/

Some Experimental Results

Benchmark	**# clusters**	**# it.**	**time**	**v?**
Sync. leader-controlled merge	22509	135	9m	y
Sync. follower-controlled merge	24957	144	22m	y
Async. leader-controlled merge	142326	202	13136m	y
Async. follower-controlled merge	58023	157	3972m	y

Some Experimental Results

Benchmark	# clusters	# it.	time	v?
Sync. leader-controlled merge	22509	135	9m	y
Sync. follower-controlled merge	24957	144	22m	y
Async. leader-controlled merge	142326	202	13136m	y
Async. follower-controlled merge	58023	157	3972m	y
Firewall	31	5	0m	y
Firewall 2	45525	7	0m	n
Public/private servers	239	10	0m	y
Dining philosophers	41	7	0m	*
Resources	32	4	0m	y
Mutual exclusion	308	17	1m	y
Euler walks	18	3	0m	*
Linked list	7	3	0m	y
Circular buffer	152	17	3m	*
Red-black tree	263	11	10m	y
AVL tree	1876	38	757m	y

Now: Tool Demo