



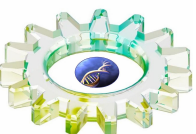
# Embryonic Models for Self-Healing Distributed Services

*Daniele Miorandi<sup>(1)</sup> , David Lowe<sup>(2)</sup> , Lidia Yamamoto<sup>(3)</sup>*

(1) CREATE-NET, [daniele.miorandi@create-net.org](mailto:daniele.miorandi@create-net.org)

(2) CRIN, Univ. of Technology, Sidney,  
[david.lowe@uts.edu.au](mailto:david.lowe@uts.edu.au)

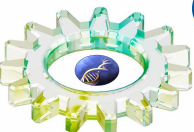
(3) Computer Science Dept., University of Basel,  
[lidia.yamamoto@unibas.ch](mailto:lidia.yamamoto@unibas.ch)





## Motivation and Setting

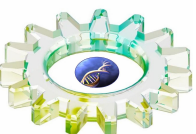
- ♦ We deal with distributed services, i.e., services whose execution involves a number of different tasks running on a plurality of interconnected machines (or nodes)
- ♦ Distributed services are used in a variety of application domains, including peer-to-peer file sharing, distributed databases and network file systems, distributed simulation engines and multiplayer games, pervasive computing environments





## Motivation and Setting

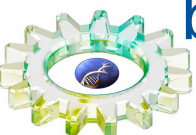
- ♦ In general, distributed services are difficult to manage. In particular, complexity issues arise with respect to the following operations:
  - Deployment
  - Configuration and setup
  - Reconfiguration after node or software faults
- ♦ Our target is to devise suitable strategies, based on self-organisation principles, to address such issues





# Inspiration

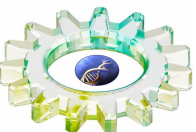
- ♦ We took inspiration from *embryological* (developmental) processes in biology
- ♦ An embryo made of initially identical cells (**stem cells**) develops into a full organism in which every cell assumes a different, specialized function, e.g. blood cells, skin cells, neurons.
- ♦ Stem cells are totipotent, i.e. unspecific and able to differentiate into the various cell types needed
- ♦ Developmental processes also encompasses growth, based on a cell division mechanism





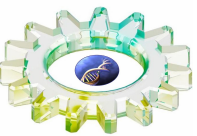
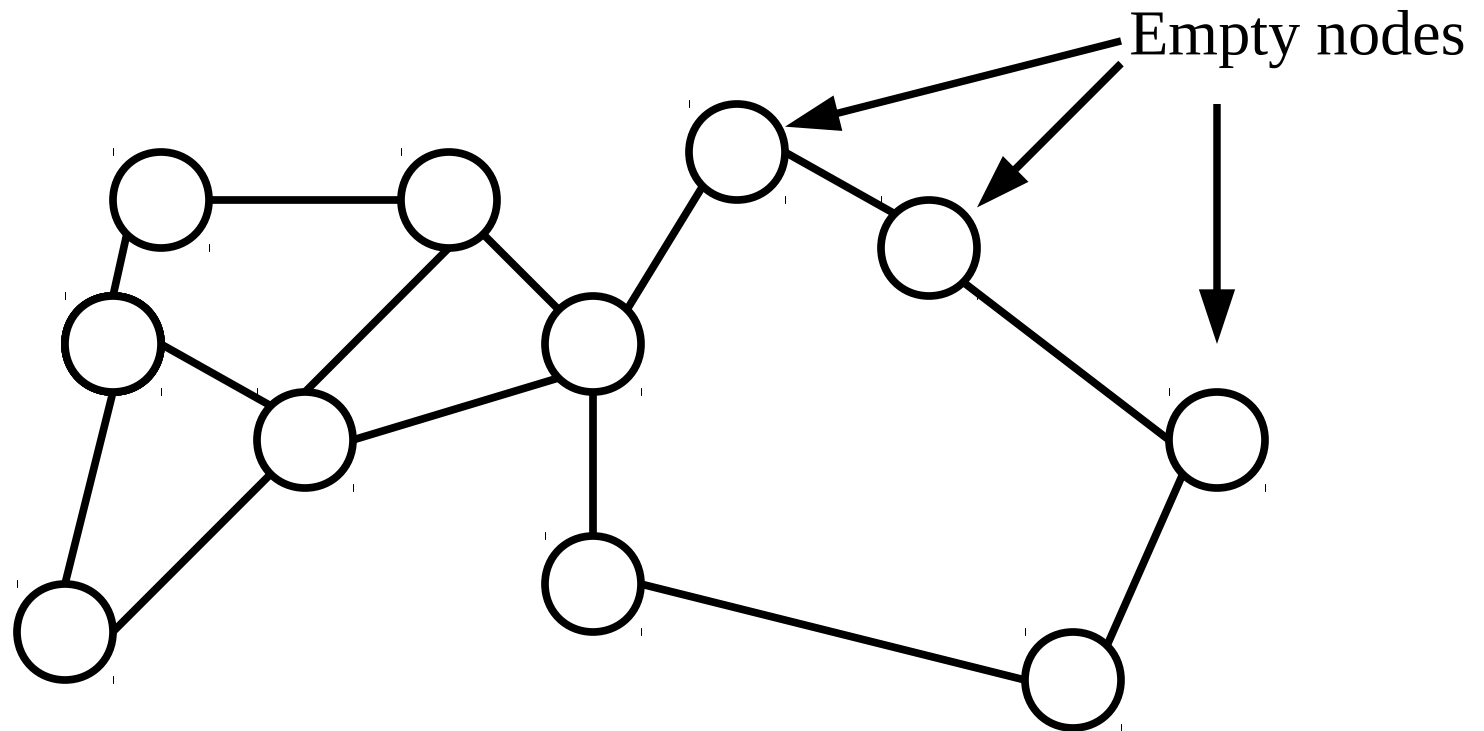
# Embryonic Software

- ◆ We aim at replicating embryogenetic processes in a distributed software setting
- ◆ Based on the notion of “**software stem cells**”
- ◆ Software stem cells contain a **genome** with a concise representation of the complete service process to be performed
- ◆ Software stem cells are initially totipotent and are designed to spread throughout the network by self-replication.
- ◆ These cells differentiate into the various components needed for performing the overall service.
- ◆ Adequate signalling mechanisms is needed to enable cells to exchange information about the state of their neighbours
- ◆ Upon detection of a fault in a neighbouring cell, they are able to re-enter the embryo state (unlike in biology), for differentiating again into the required functionalities
- ◆ We call this approach **EmbryoWare (Embryonic Software)**



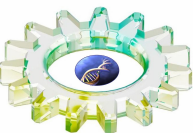
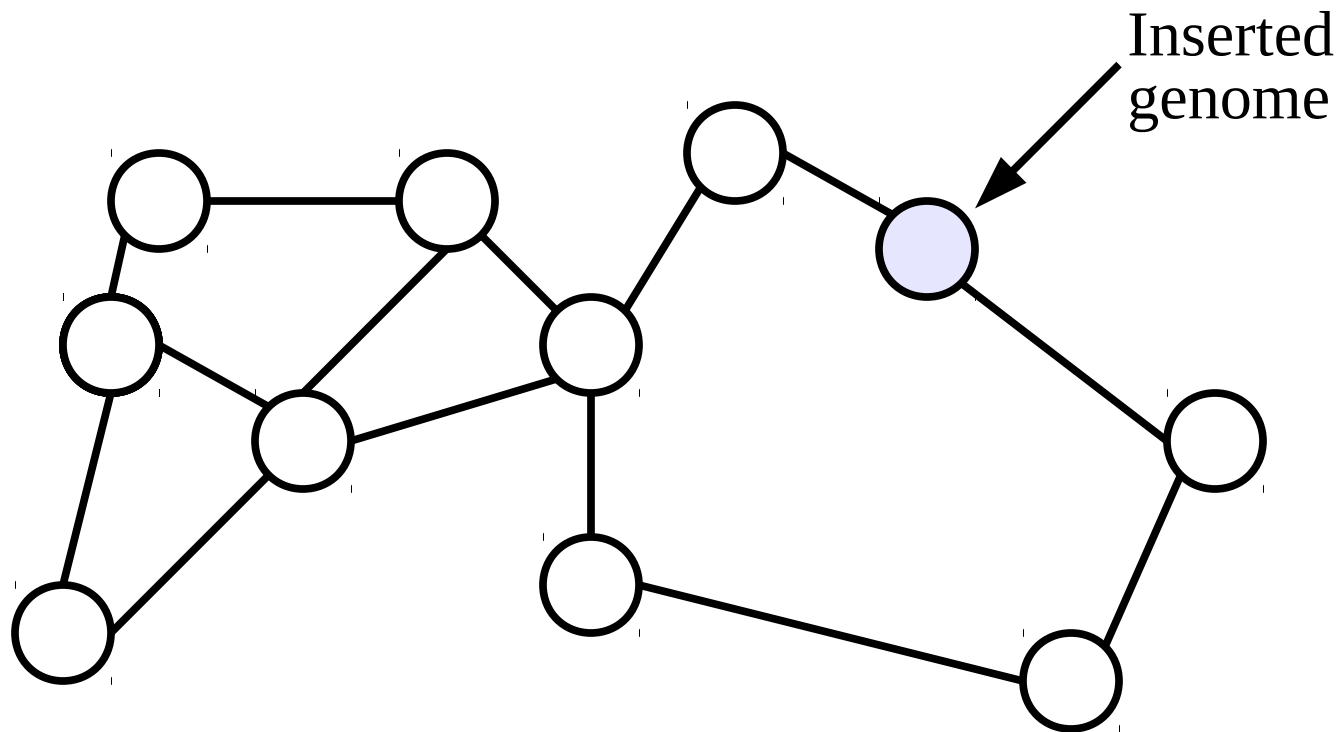


## How would it look like?





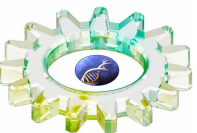
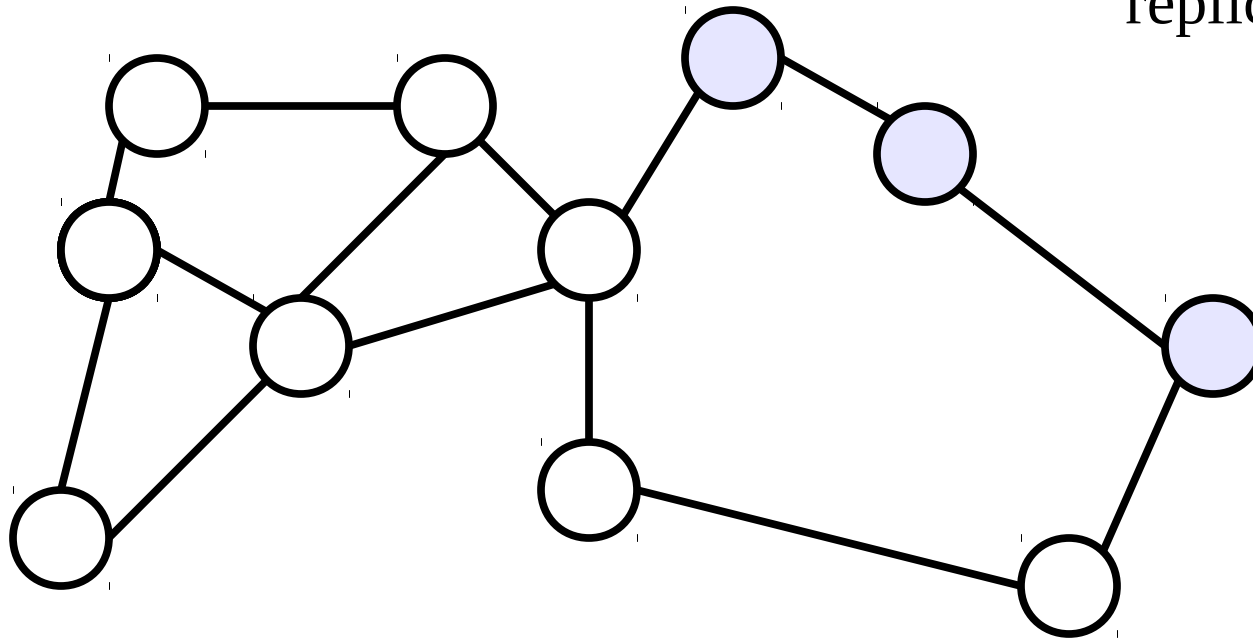
## How would it look like?





## How would it look like?

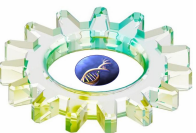
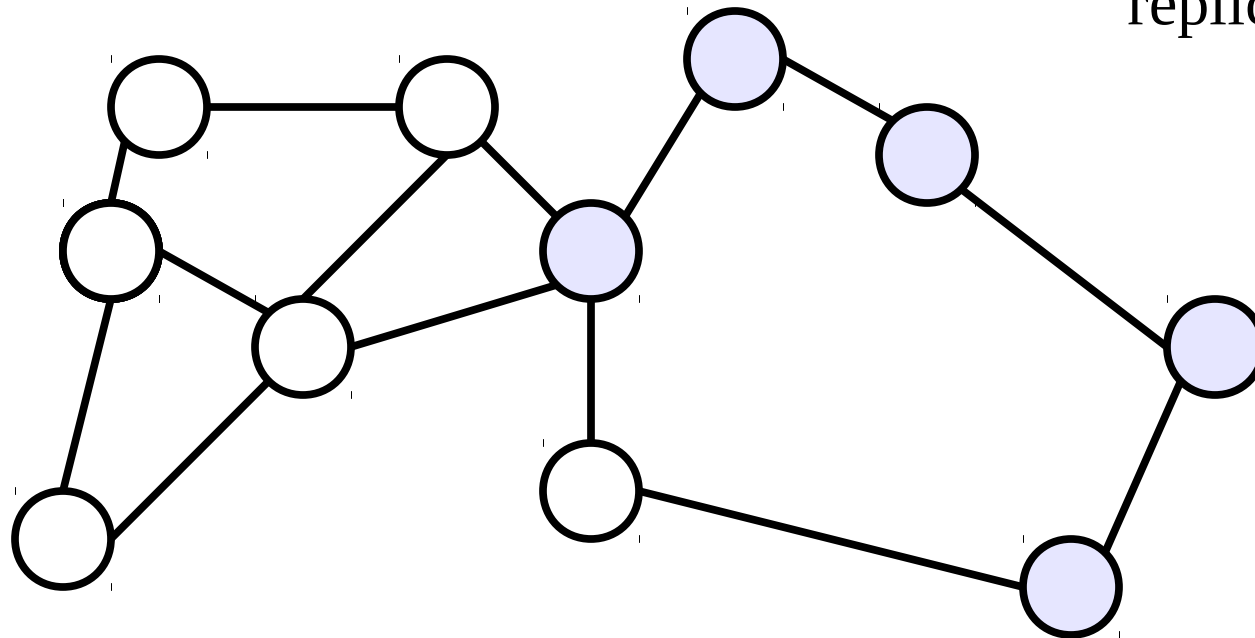
Genome replicates





## How would it look like?

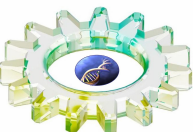
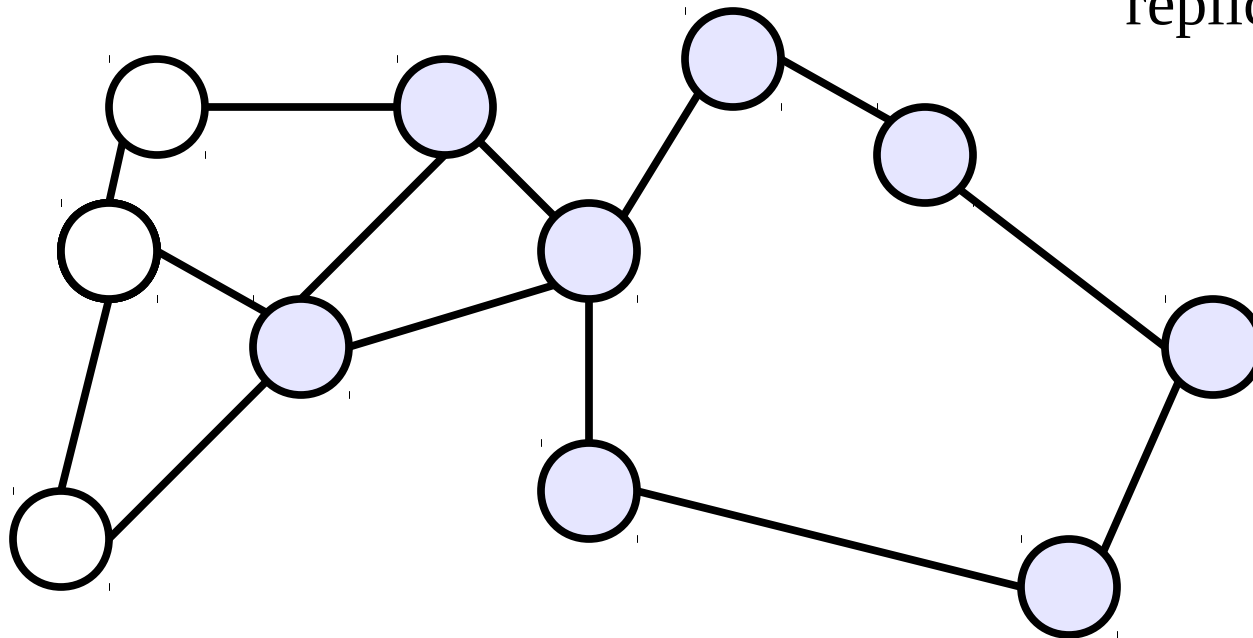
Genome replicates





## How would it look like?

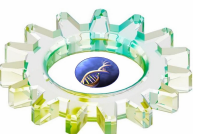
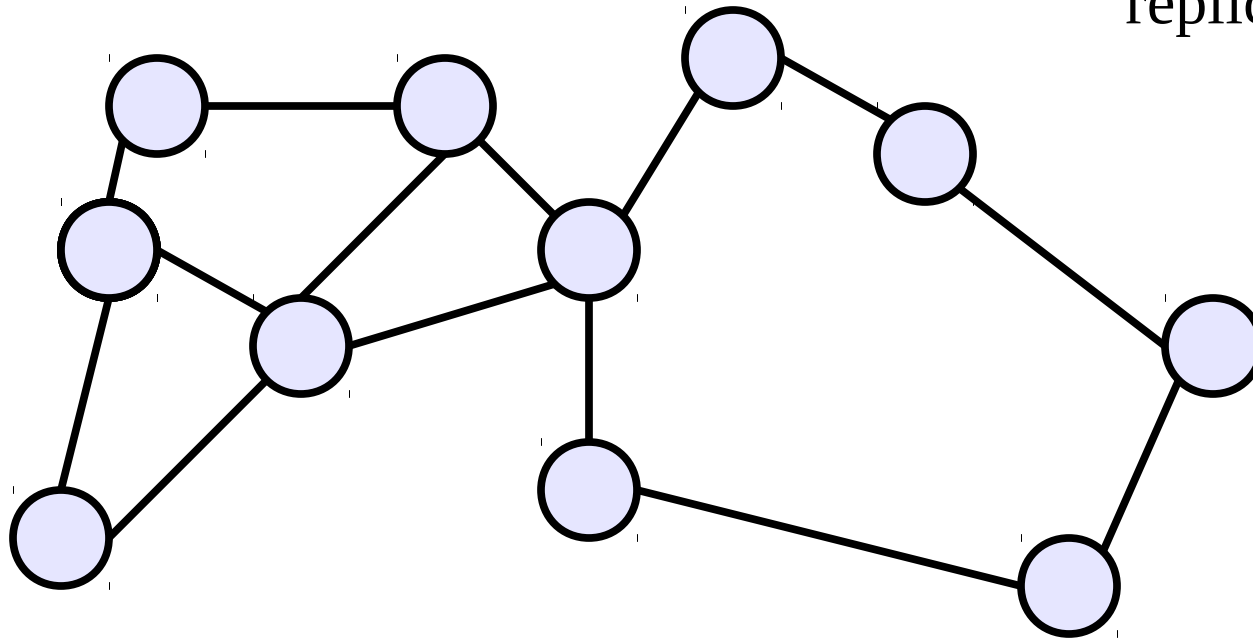
Genome  
replicates





## How would it look like?

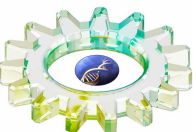
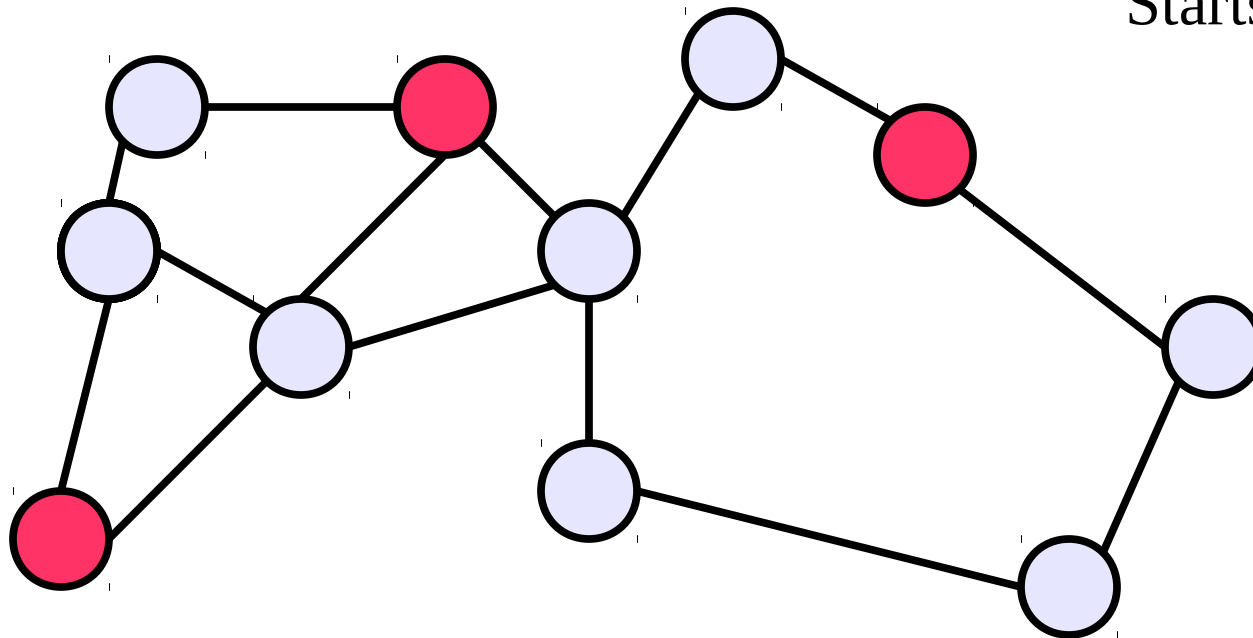
Genome  
replicates





## How would it look like?

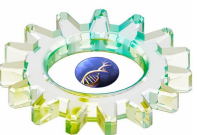
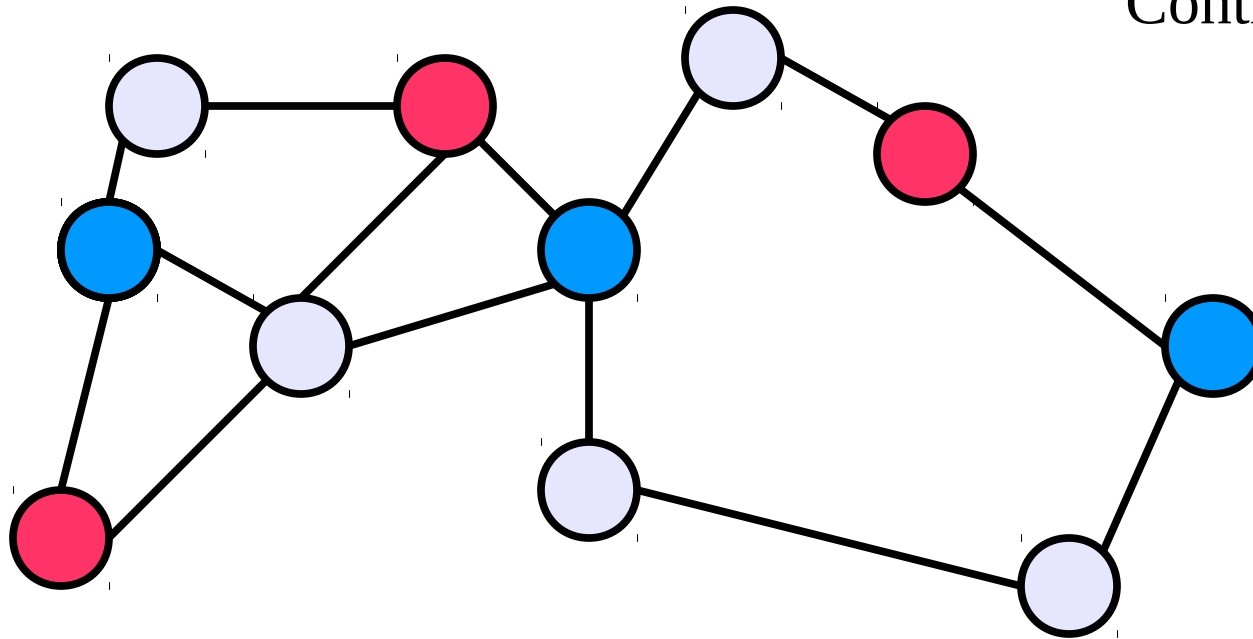
Differentiation  
Starts





## How would it look like?

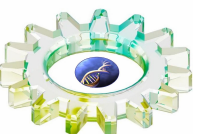
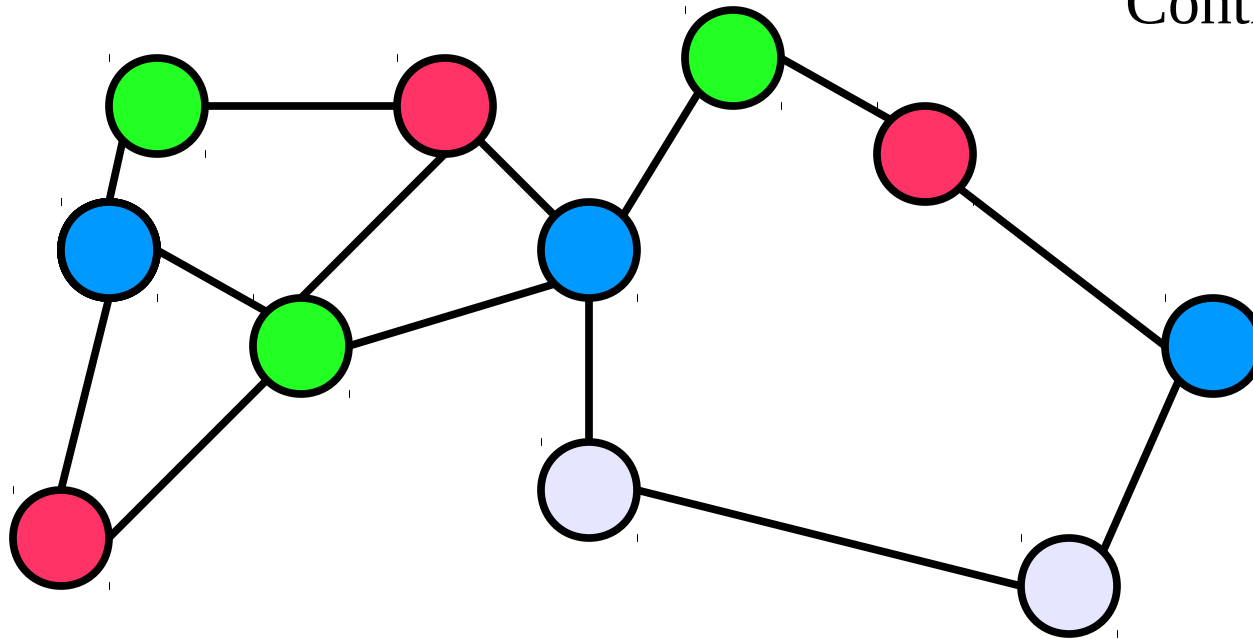
Differentiation  
Continues





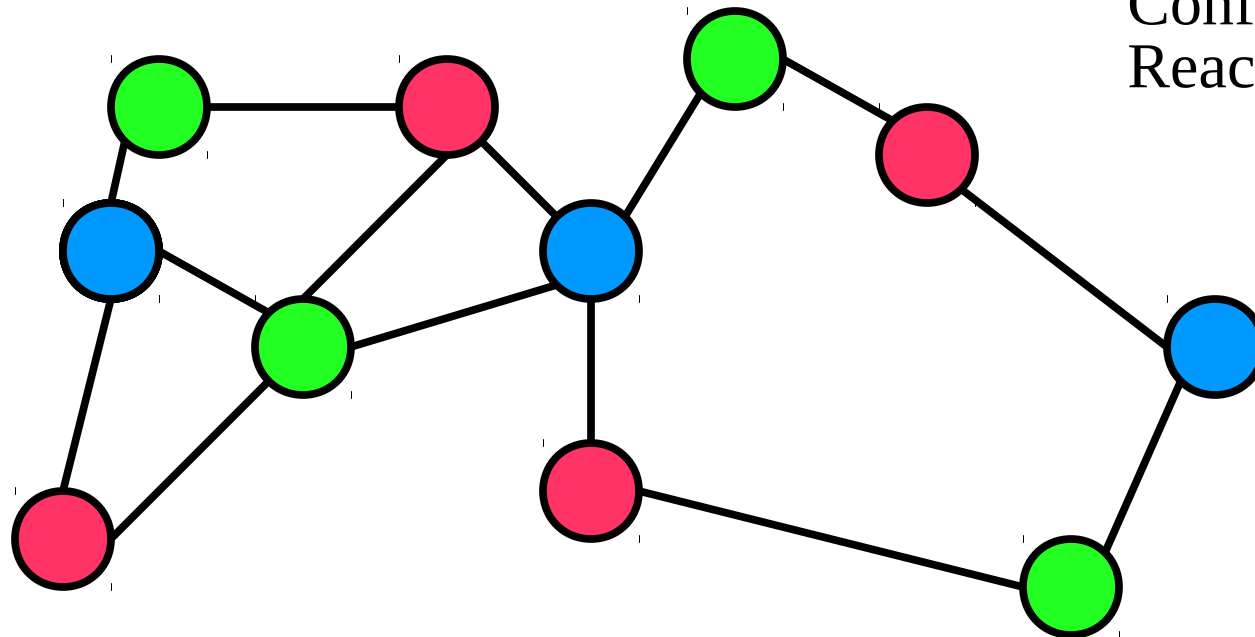
## How would it look like?

Differentiation  
Continues

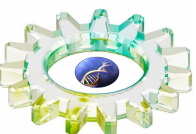




## How would it look like?



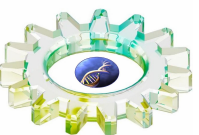
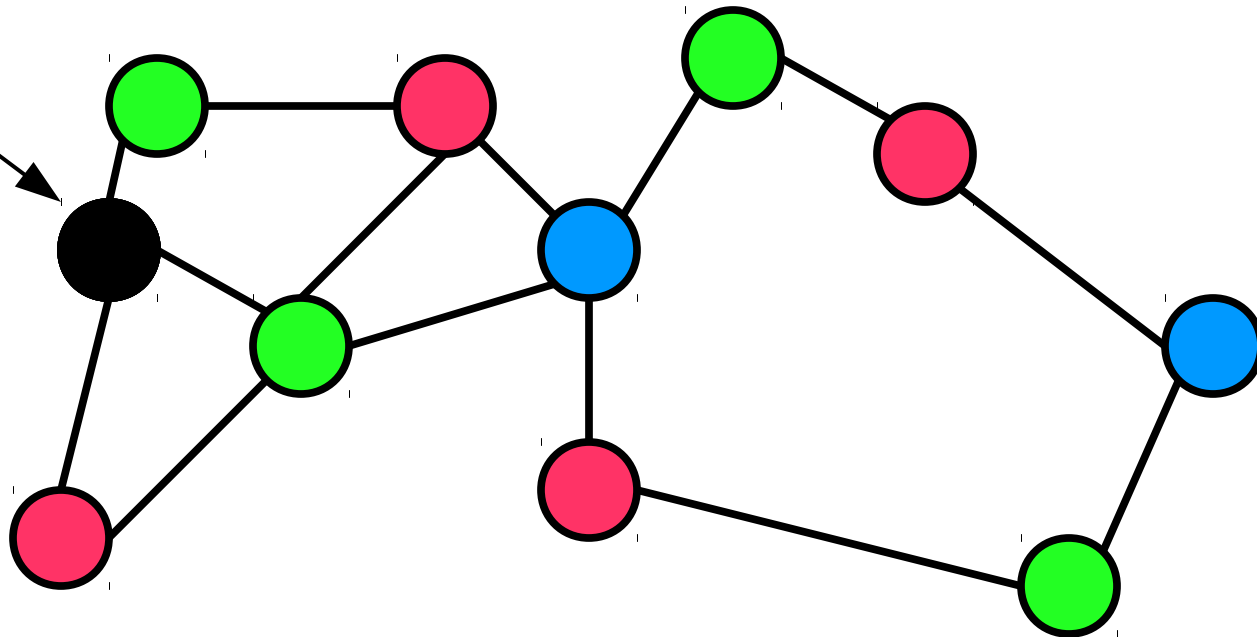
Stable  
Configuration  
Reached





## How would it look like?

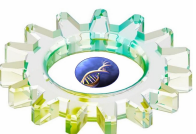
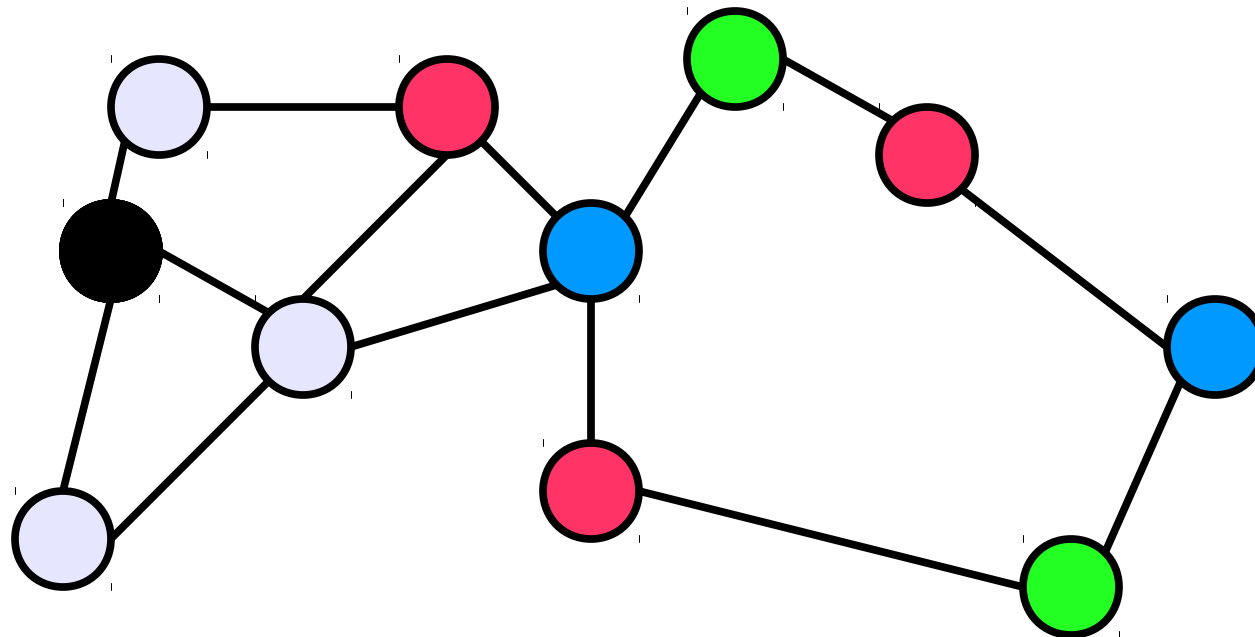
Node  
fails





## How would it look like?

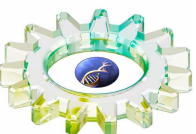
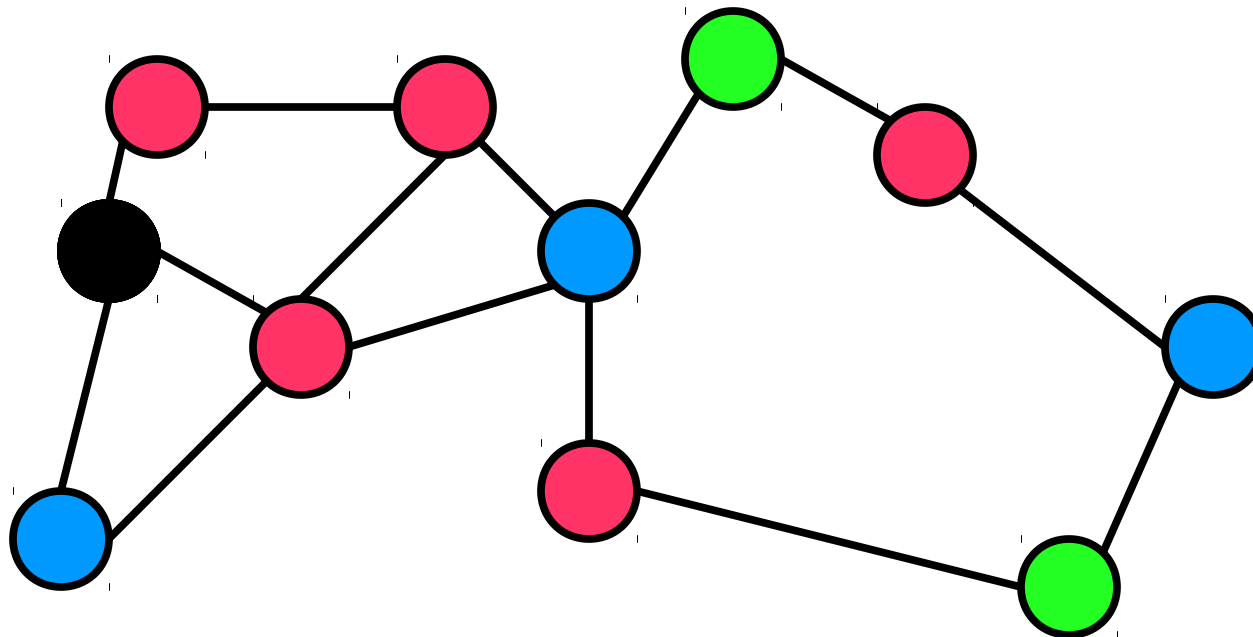
Reconfiguration  
triggered:  
back to the  
stem cell  
state





## How would it look like?

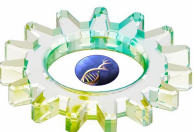
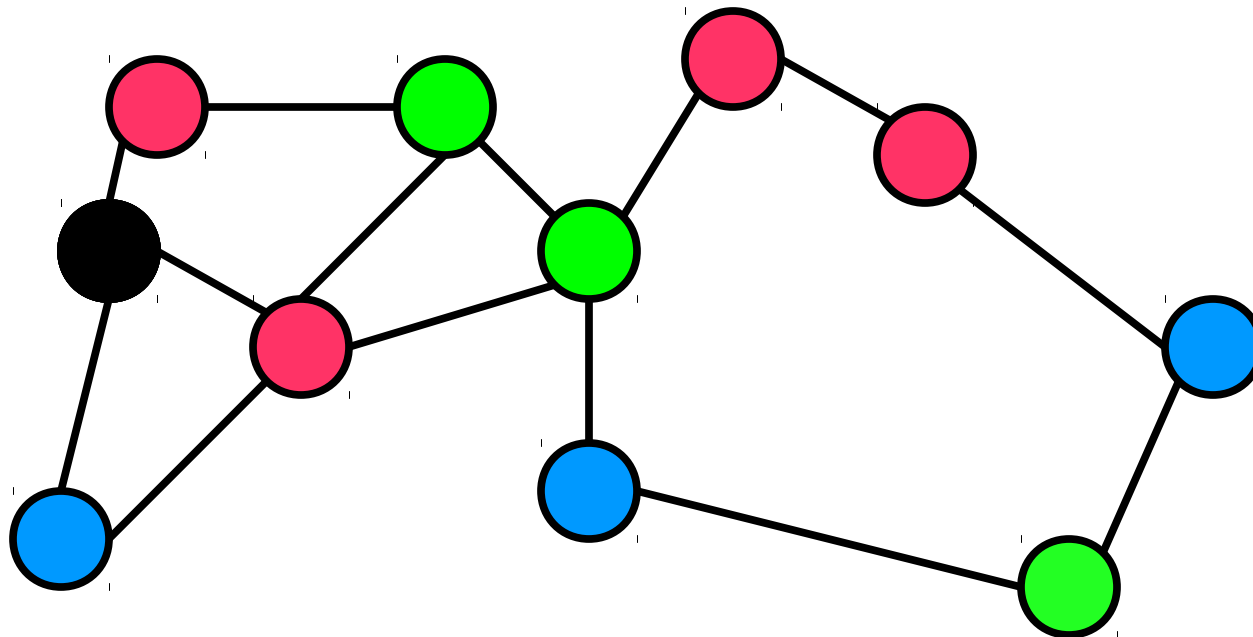
Reconfiguration  
continues





## How would it look like?

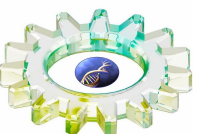
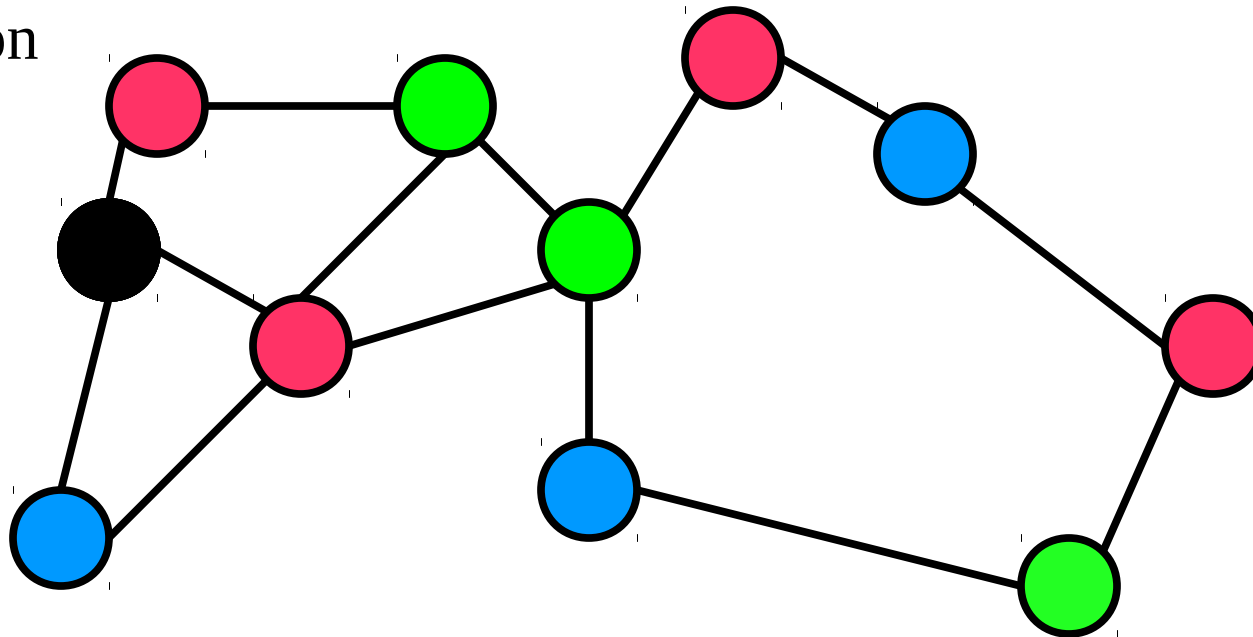
Reconfiguration  
continues





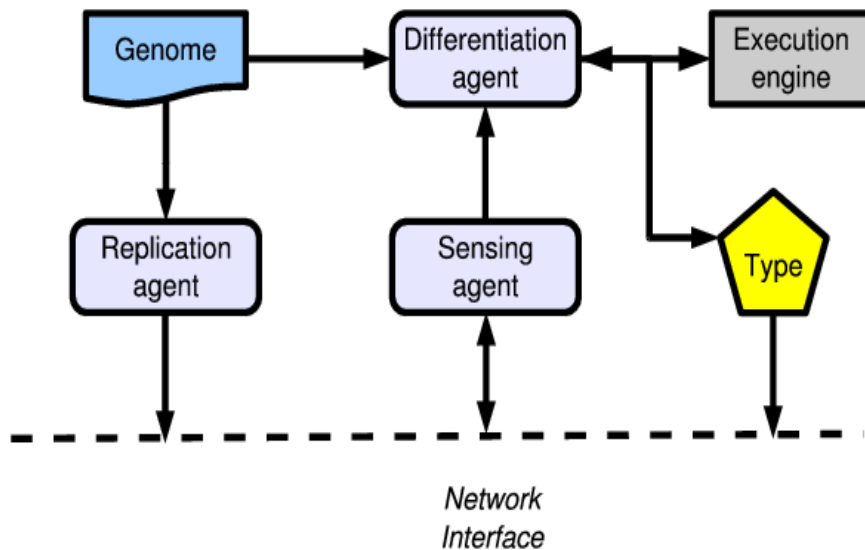
## How would it look like?

Stable  
configuration  
reached

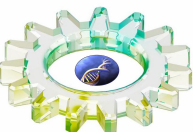




# EmbryoWare Architecture



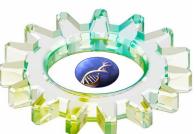
- Each cell contains:
  - A replication agent (replicates genome in neighbouring nodes)
  - A sensing agent (monitors the type of neighbouring cells)
  - A differentiation agent (performs the differentiation based on the policies specified in the genome and sends into execution the appropriate code)





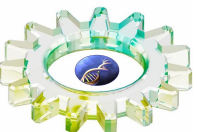
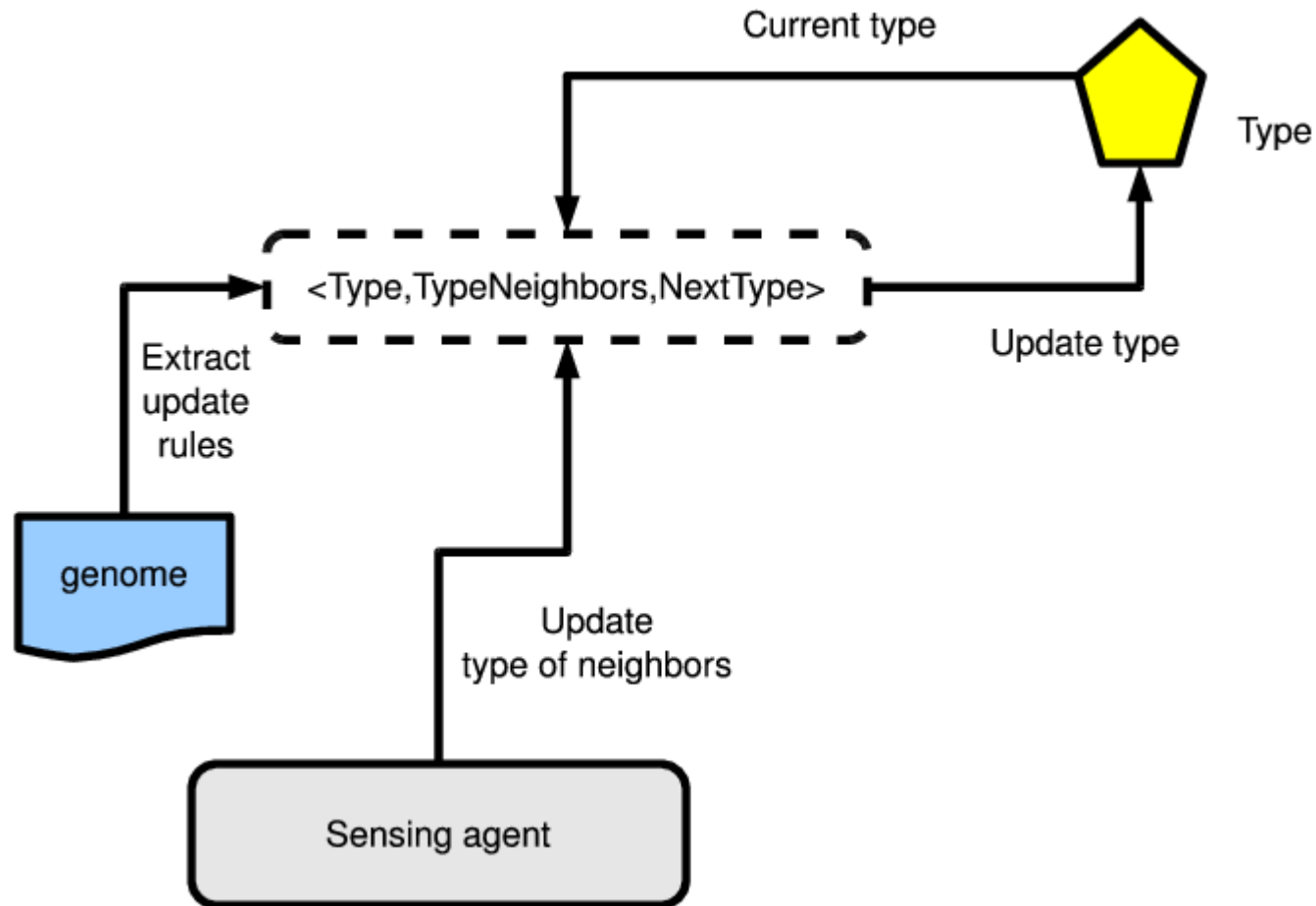
## EmbryoWare Architecture

- ▶ The genome includes:
  - (1) a description of the expected behaviour of the service as a whole;
  - (2) a description of the single tasks to be performed by cells;
  - (3) a set of rules for deciding, based on the current task and the task performed by neighbouring cells, which task is to be performed next.



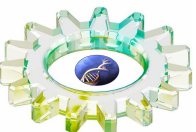
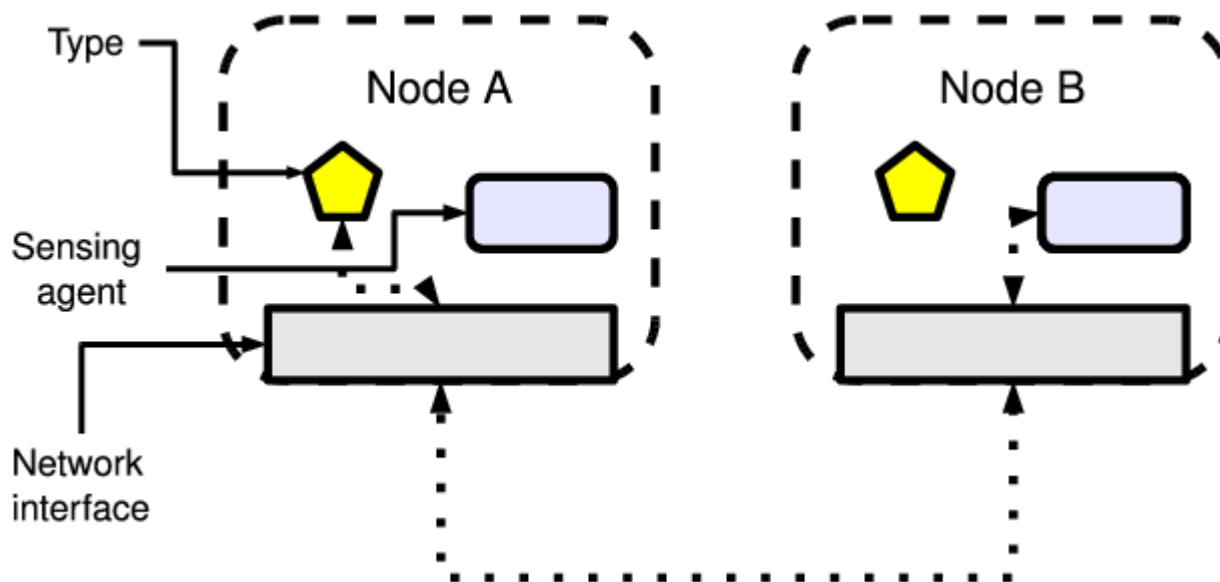


## EmbryoWare Architecture (cont'd)





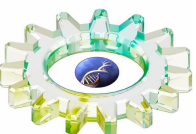
## EmbryoWare Architecture (cont'd)





## EmbryoWare: Methods

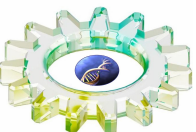
- ▶ Sensing, differentiation and replication processes are performed periodically
- ▶ Sensing and differentiation are done at the same rate
- ▶ Replication has a much longer period
- ▶ Various differentiation policies can be envisaged, from simple lookup table to probabilistic strategies





# Experimental Validation

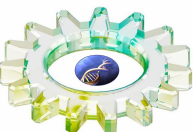
- ◆ Focus on a distributed network monitoring scenario
- ◆ Five cell types: stem, monitor, logger, alarm, analysis
- ◆ Each monitoring sample needs to be reported to a logging cell that should be no more than 2 hops away from the monitoring cell
- ◆ The logging cell will accumulate data samples and report them periodically to alarm cells.
- ◆ The network should contain 2 alarm cells for redundancy, but no more than 2 alarm cells.
- ◆ Each alarm cell should have two 1-hop neighbours that are analysis cells which it uses to assist in analysing the provided samples.
- ◆ In order to distribute the load associated with alarm condition evaluation, the cells taking on the alarm behaviour should change periodically.





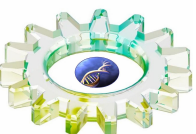
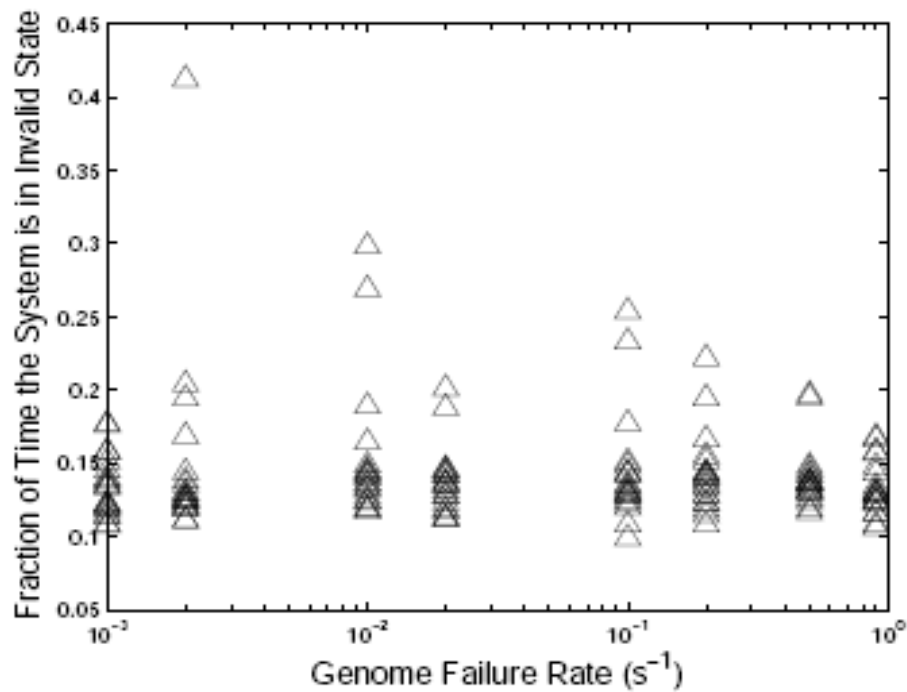
## Implementation Details

- ◆ Based on a probabilistic differentiation mechanisms
- ◆ Type is changed according to a given probability distribution depending on the current local state (self plus neighbours)
- ◆ Challenge is to achieve given network-level behaviours (e.g., presence of 2 alarm nodes) using only local information
- ◆ Implemented and tested in Matlab
- ◆ Varied the genome fault rate to assess robustness and self-healing properties
- ◆ Varied network size to assess scalability properties



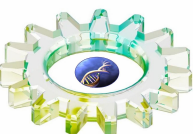
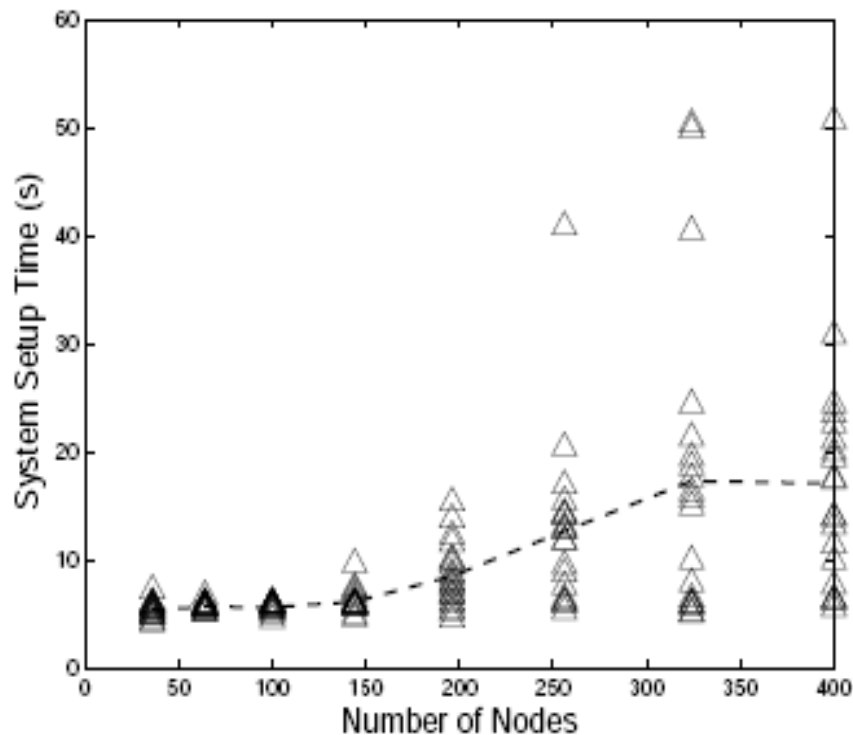


# Results and Assessment





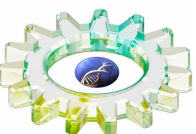
## Results and Assessment (cont'd)





## Insight from Simulations

- ♦ The optimal tuning of the differentiation parameters is dependant upon the network size
- ♦ Network fringe effects need to be considered in designing the genome behaviours
- ♦ *Hydra* behaviour: if we split the network of nodes into two isolated sub-networks, then the nodes differentiate in order to create a fully operational system in each sub-network, exhibiting a natural self-healing ability.





## Next Steps

- ♦ In EmbryoWare the complexity is moved to the design of appropriate differentiation policies in the genome
  - Explore application of evolutionary strategies for automating the genome design
- ♦ Implementation in a suitable multi-agent environment to test in real-world systems
  - To address impact of communication delays, synchronization issues etc.

