DELIS

Dynamically Evolving, Large-scale Information Systems

SP 5: Biologically Inspired Techniques for “Organic IT”

Report for months 25 - 36

Participants

UniBO, UPF, Telenor, RAL

Lead partner: Bologna (UniBO)

Information Society Technologies

SP5 Goals

Goals of SP5 “Biologically Inspired Techniques for Organic IT”

General

Identify, understand and reverse engineer techniques inspired by biological and social systems that display “self-*” properties. Deploy these in networked information systems

Specific

Consolidate and import BISON findings. Identify “nice” properties of biological and social systems. Relate found natural network “forms” to engineering “functions”

Identify desirable life-like properties - “Self-*”

Algorithms	Simulations / Tools	Implementations

Industrial Applications

Structure of SP5 “Biologically Inspired Techniques for Organic IT”

	Months							
WP	25-27	28-30	31-33	34-36	37-39	40-42	43-45	46-48
5.2	Evolved tinkering and degeneracy as engineering concepts			●				●
5.3	Bio-inspired design for dynamic solution spaces			●				●
5.4	Multi-scale topology evolution in natural and artificial networks			●				●
5.5	Identifying and promoting industrial applications			●				
5.6	The structure of tinkered landscapes			●				

● **= deliverable**

SP5 Deliverables Overview

Deliverables for 2006

D5.2.4: Modelling open source development networks (month 36)
D5.3.1: From biological and social algorithms to engineering solutions (month 36)
D5.4.2: Understanding and engineering ``multi-scale'' selection in evol.nets (month 36)
D5.5.1: Promising industrial applications in dynamically evolving networks (month 37)
D5.6.2: Integrated package for evolutionary dynamics of information networks including evolved design and landscape structure (month 36)

Deliverables planned for 2007

D5.2.5: Degeneracy and redundancy in self-organised systems (month 48)
D5.3.2: Applications of bio- and socio-inspired algorithms in info. Systems (month 48)
D5.4.3: Form and function in evolving information systems (month 48)

Goals (start month 0)

General

Explore ways of applying evolutionary computational strategies to the optimisation of pre-existing information systems. Facilitate the interaction between engineers and automatic systems in the construction of efficient information processing networks

Specific

Investigate the topological evolution of open source developer networks. Identify structures and processes.

Partners

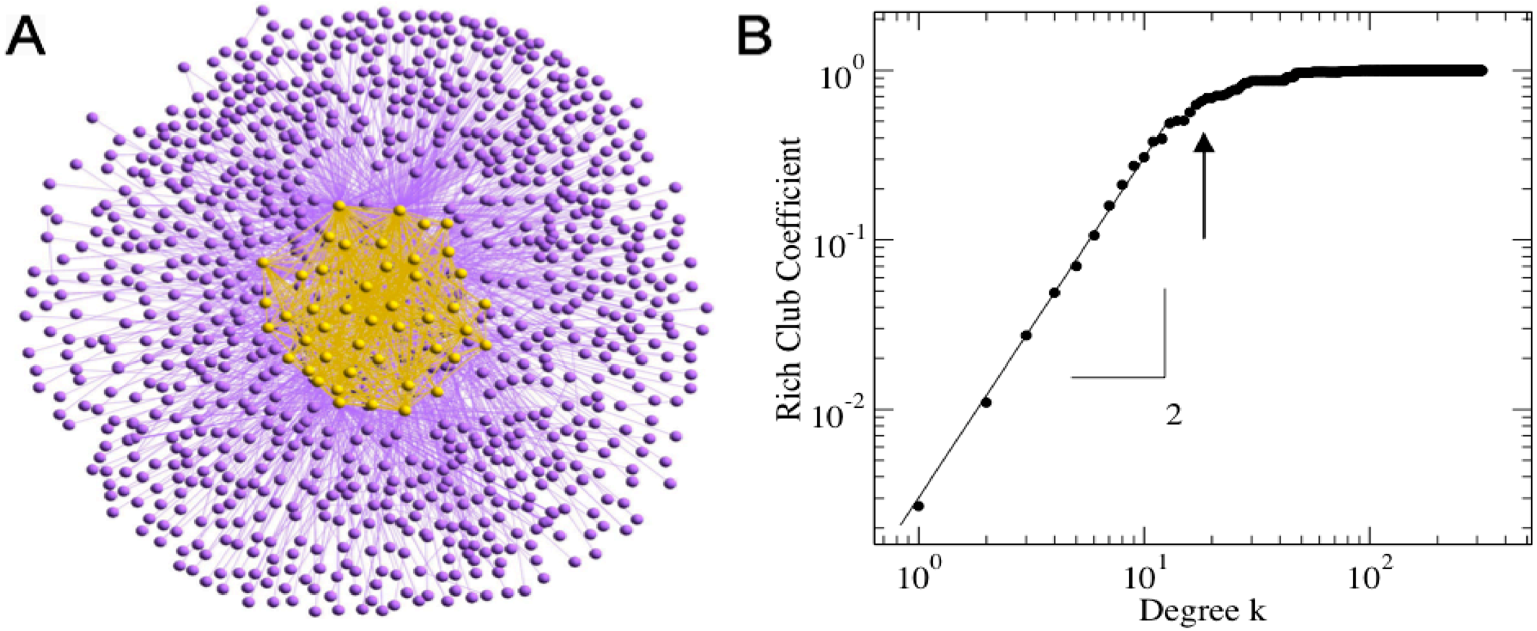
UPF, UniBO, Telenor

Results (from D5.2.4)

- Analysis of open source (OS) development community e-mail logs
- Shows strong hierarchy (rather than highly distributed)
- A “rich club” can be identified in a quantitative way
- Simple mode of preferential attachment with non-local evolution can reproduce this pattern
- More generally, first quantitative empirical evidence for the emergence of hierarchy in distributed networks of interacting agents

Rich Clubs in OS Communities

- Analysis of e-mail logs between open source developers for 120 different OS projects
- Filtered such that only e-mail related to bugs / development used
- Produces a weighted social network with weight indicating the amount of e-mails sent between pairs of developers
- Found power law distributions
- Hence there are a few pairs of members exchanging more e-mails than with the rest of the community.
- Analysis suggests these key members play the role of hubs since hey also have the largest number of connections
- Further, they form a “rich club”, connecting almost exclusively to each other rather than less active members



Correlations and rich-club phenomenon in the *Python* OS community. (A) Visualization of the rich-club where yellow balls depict hubs having $k > k_c$. (B) The rich-club coefficient scales with degree k and saturates once $k > k_c$. The pointing arrow indicates the crossover $k_c \approx 10$.

WP5.2 Summary

- Open source communities comprise a small set of hubs mediating the majority of e-mail traffic
- These hubs follow a “rich club” structure and can be automatically identified
- Counter to common view that OS is completely distributed, it is highly centralised
- Publications:
  - Sergi Valverde et al, (2006) “Self-Organization Patterns in Wasp and Open-Source Communities”, IEEE Intelligent Systems, 21(2), pp. 36-40
  - Sergi Valverde and Ricard V. Sole, “Self-organization and Hierarchy in Open-Source Social Networks”, *Submitted* to Physical Review E.
- Future: Metrics / tools for open source development communities, relating degeneracy in P2P systems (D5.2.5, month 48)

Goals (start month 19)

General

Develop tools and methods to translate / modify biologically and socially inspired algorithms for application in realistic information systems environments

Specific

Select a set of candidate algorithms and application domains. Use simulation and apply necessary tuning to produce acceptable performance

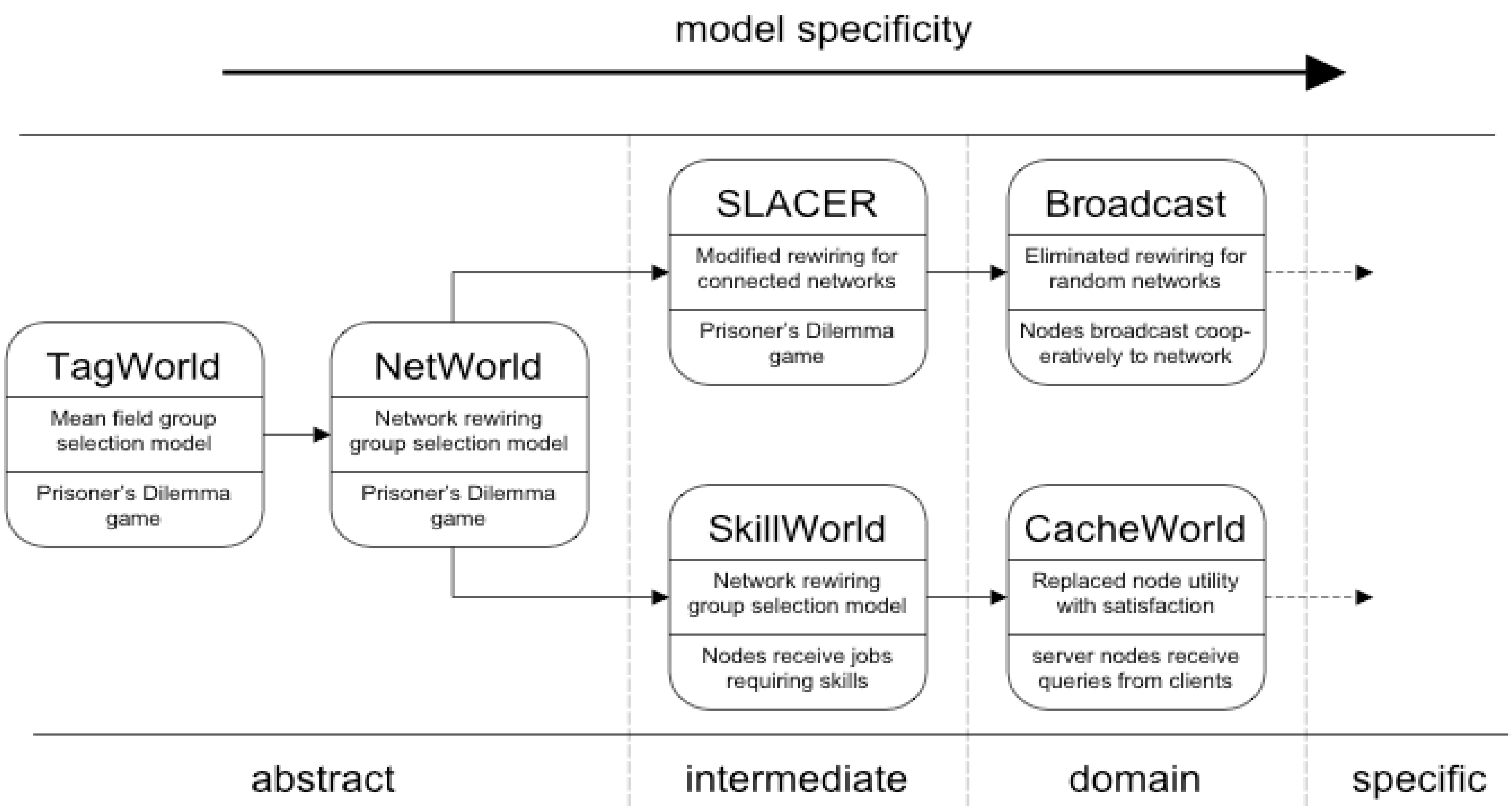
Partners

UniBO, RAL, UPF, Telenor

Results from D5.3.1

- Method / approach
  - model chains - from abstract / general models to concrete applications
  - towards “design patterns” of general approaches
  - Artificial Social Bootstrapping (ASB) idea
  - grounding / representing ``utility’’, dealing with utility lying
- Broadcast application
  - nodes initiate, receive and possibly pass-on messages
  - evolve passing behaviour selfishly
  - appears to self-organise around a critical threshold
- Cooperative content replication
  - server nodes form a dynamic overlay
  - neighbours mutually replicate content and share queries
  - nodes behave selfishly forming mutually beneficial alliances

Deriving specific applications via model chains



Tags, SLAC and SLACER

- P2P networks are usually open systems
  - Possibility to free-ride
  - High levels of free-riding can seriously degrade global performance
- We present simple protocols (SLAC & SLACER) that sustain high levels of cooperation despite selfish nodes
- We show that certain types of cheating and lying behavior do not necessarily destroy cooperation (on the contrary, may even improve, certain aspects, of it!)

Sociological Inspiration: Tag Systems

- Originated in Computational Sociology (John Holland 1992)
- Developed by Michigan group – Riolo, Axelrod, Cohen.
- Tags: observable social cues e.g. hairstyle, dress, accent...
- Tags evolve through copying and mutation (genes / memes)
- Limiting interactions between agents with similar tags leads to cooperative altruistic behaviour
- Agents characterized by Tag, Behavior (strategy), Utility
- Agents features in tag systems
  - Interaction restricted to agents with similar tag
  - Selfish optimization through copy of tag and behavior of better performing agents (both copied as one “package”)
  - Periodic mutation of tag and behavior

From Tag System to P2P

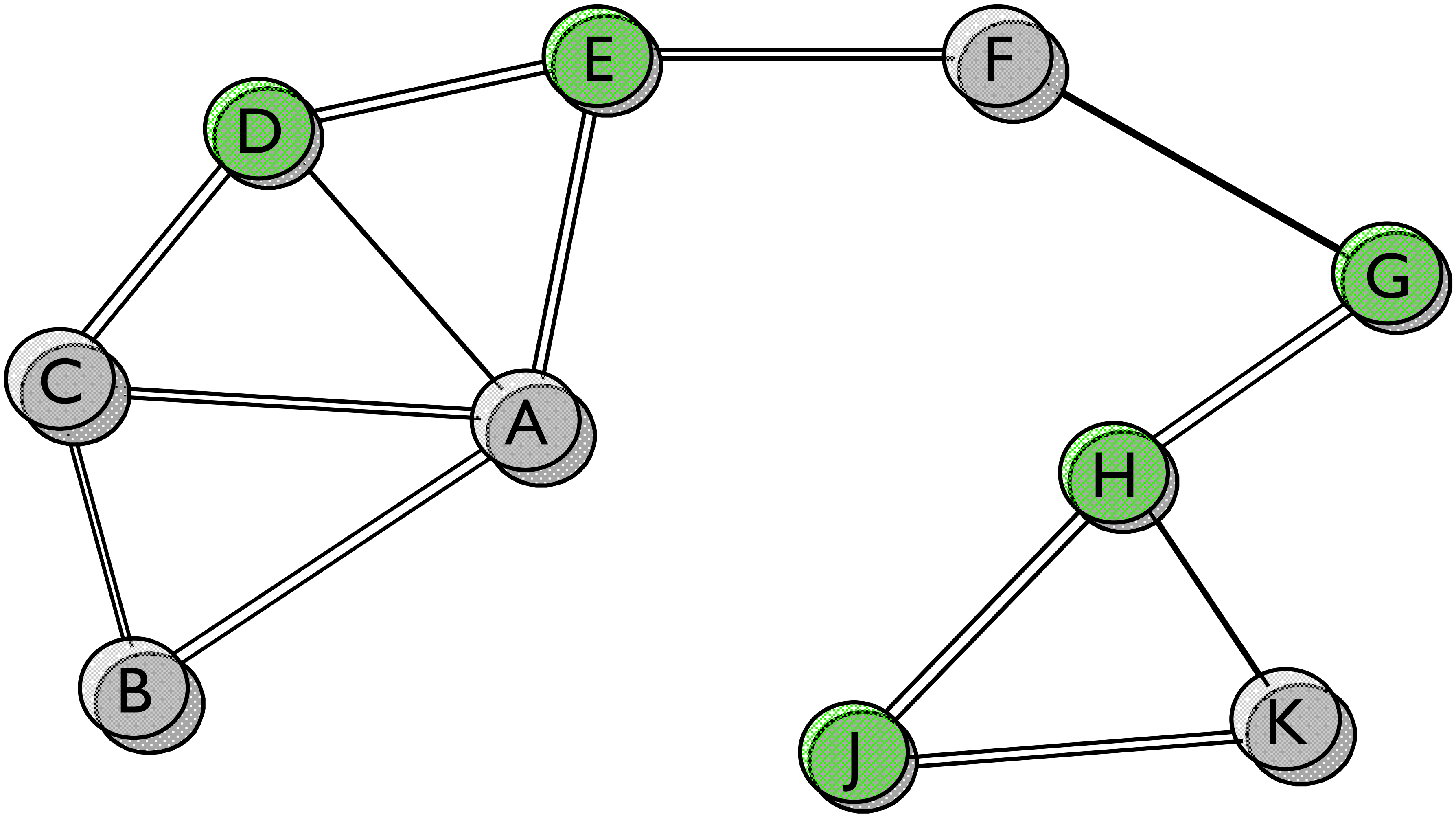
- Agents represented by nodes in an overlay network
- Tag represented by set of neighbors (view) in overlay
- Interaction between neighbors to achive an application task
- Behavior: Application behavior (i.e. share files or leech files)
- Utility: Evaluated at application level (i.e. number of files downloaded)

Node *p* periodically executes the following:

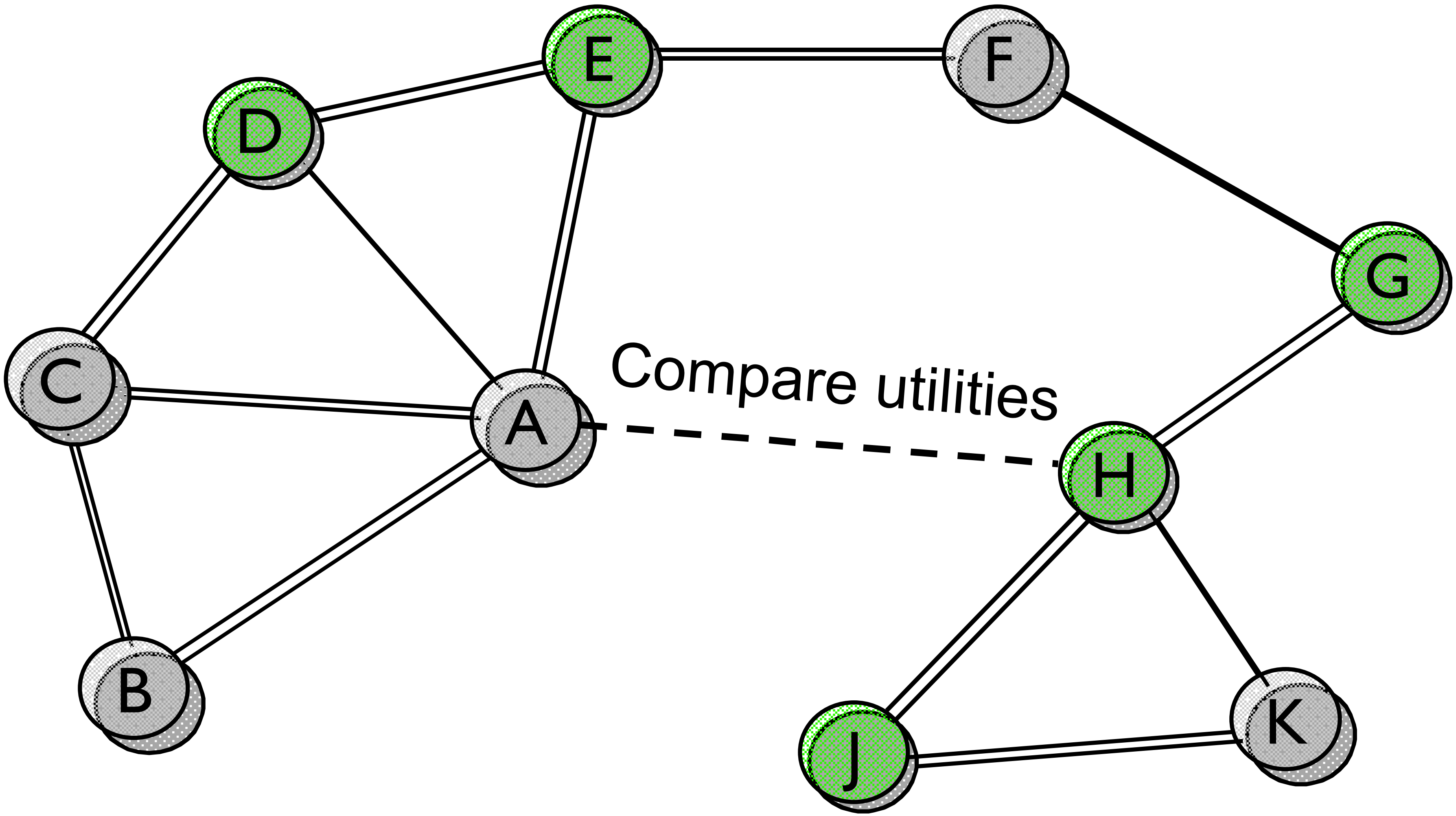
q = SelectPeer()
if utility$_q$ > utility$_p$
drop all current links
link to node *q* and copy its strategy and links
mutate (with low probability) strategy and links
fi

Peer selection based on a random overlay network (Newscast), whereas *copying*, *rewiring* and *mutating* are with respect to an application (strategy) over an "interaction network"

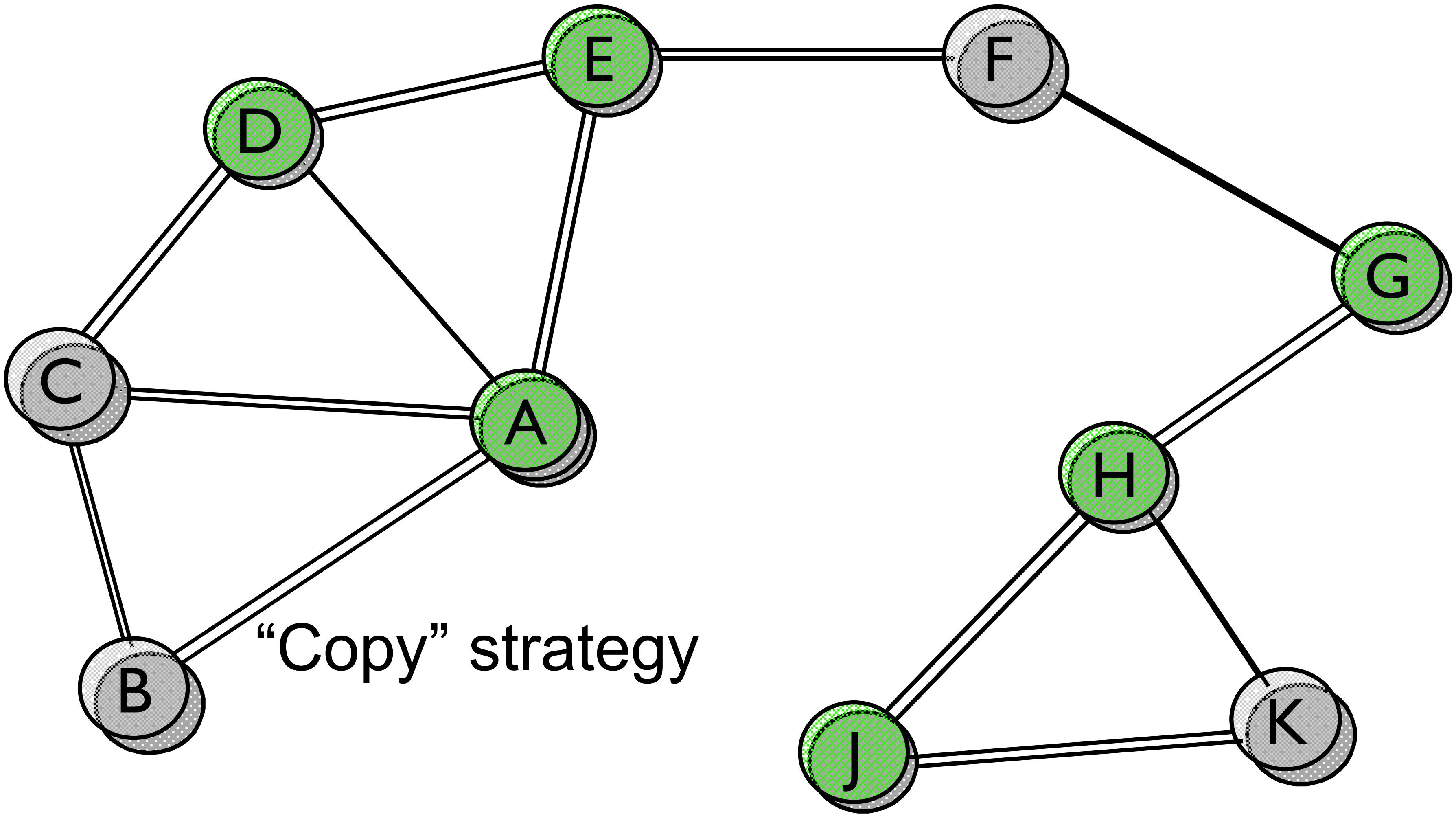
SLAC Algorithm: “Copy and Rewire”



SLAC Algorithm: "Copy and Rewire"

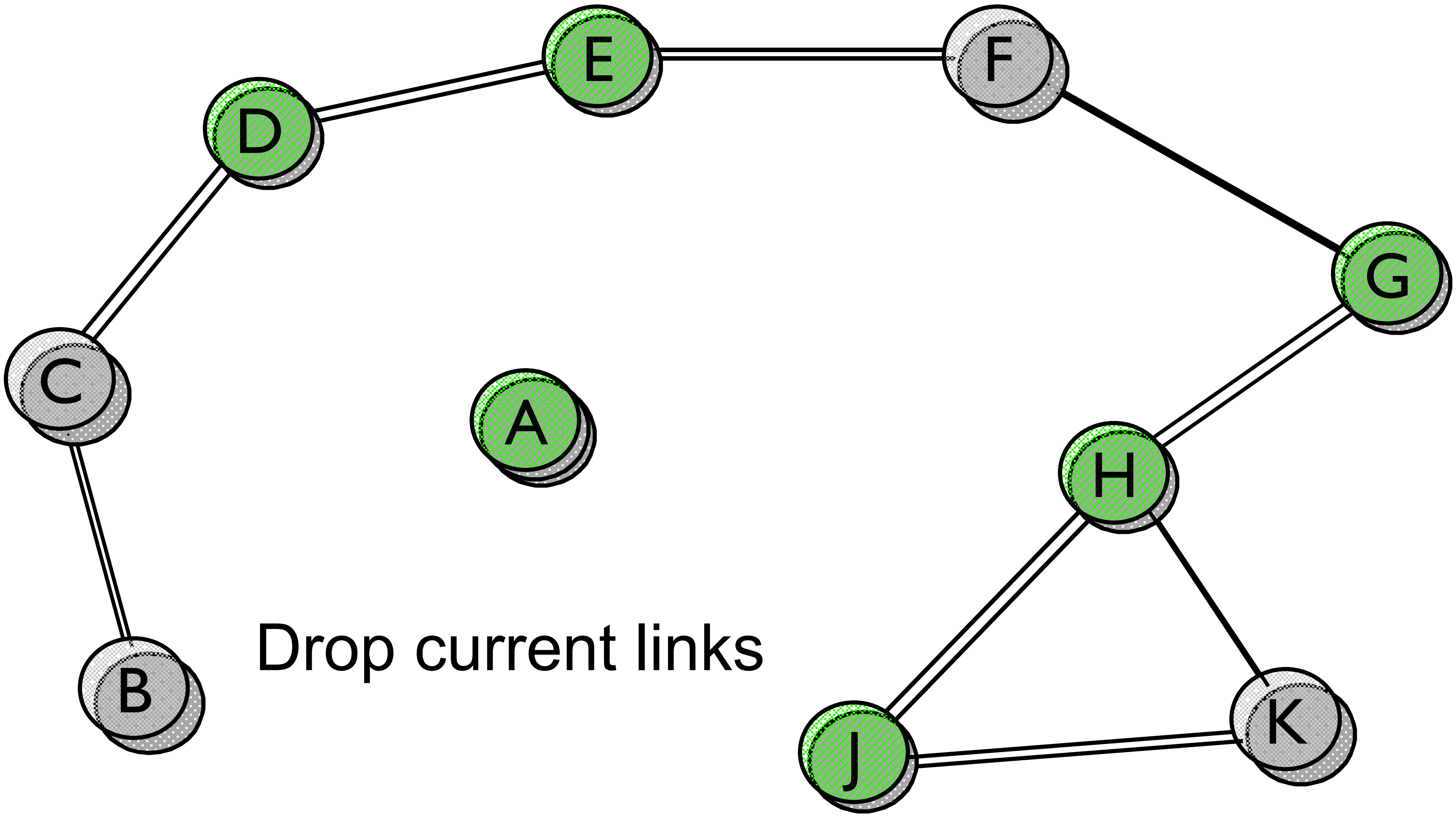


SLAC Algorithm: "Copy and Rewire"



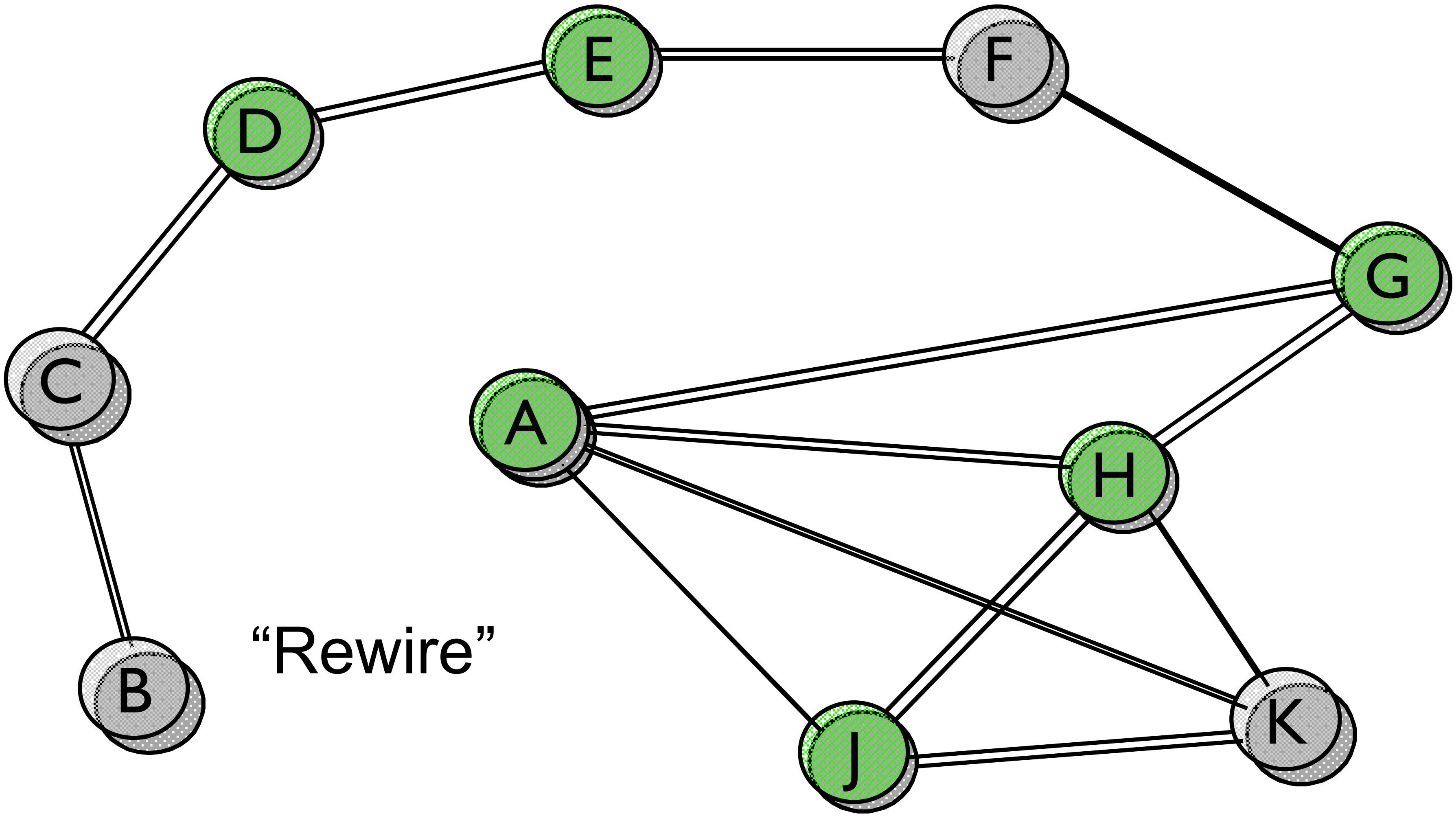
"Copy" strategy

SLAC Algorithm: "Copy and Rewire"

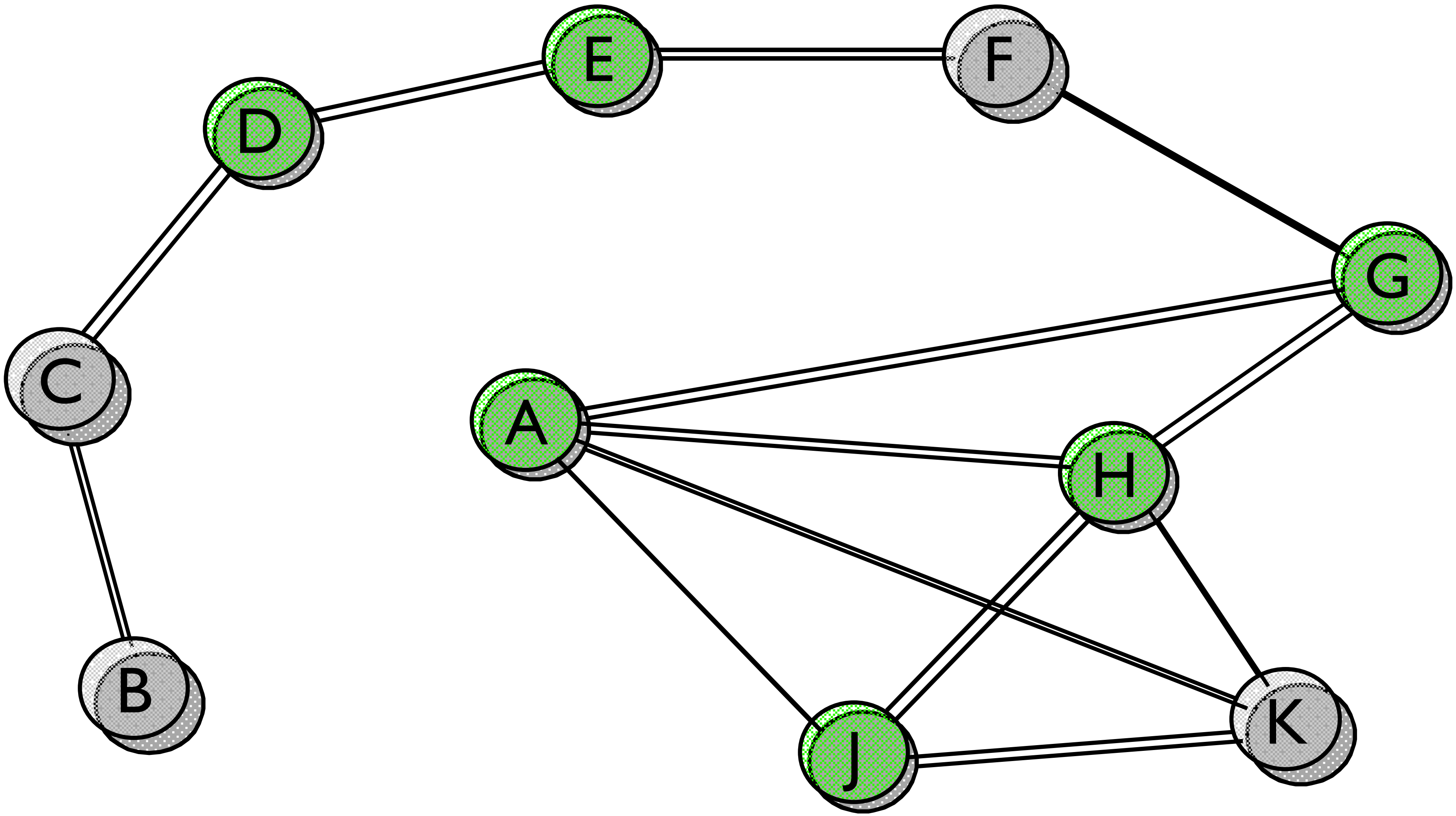


Drop current links

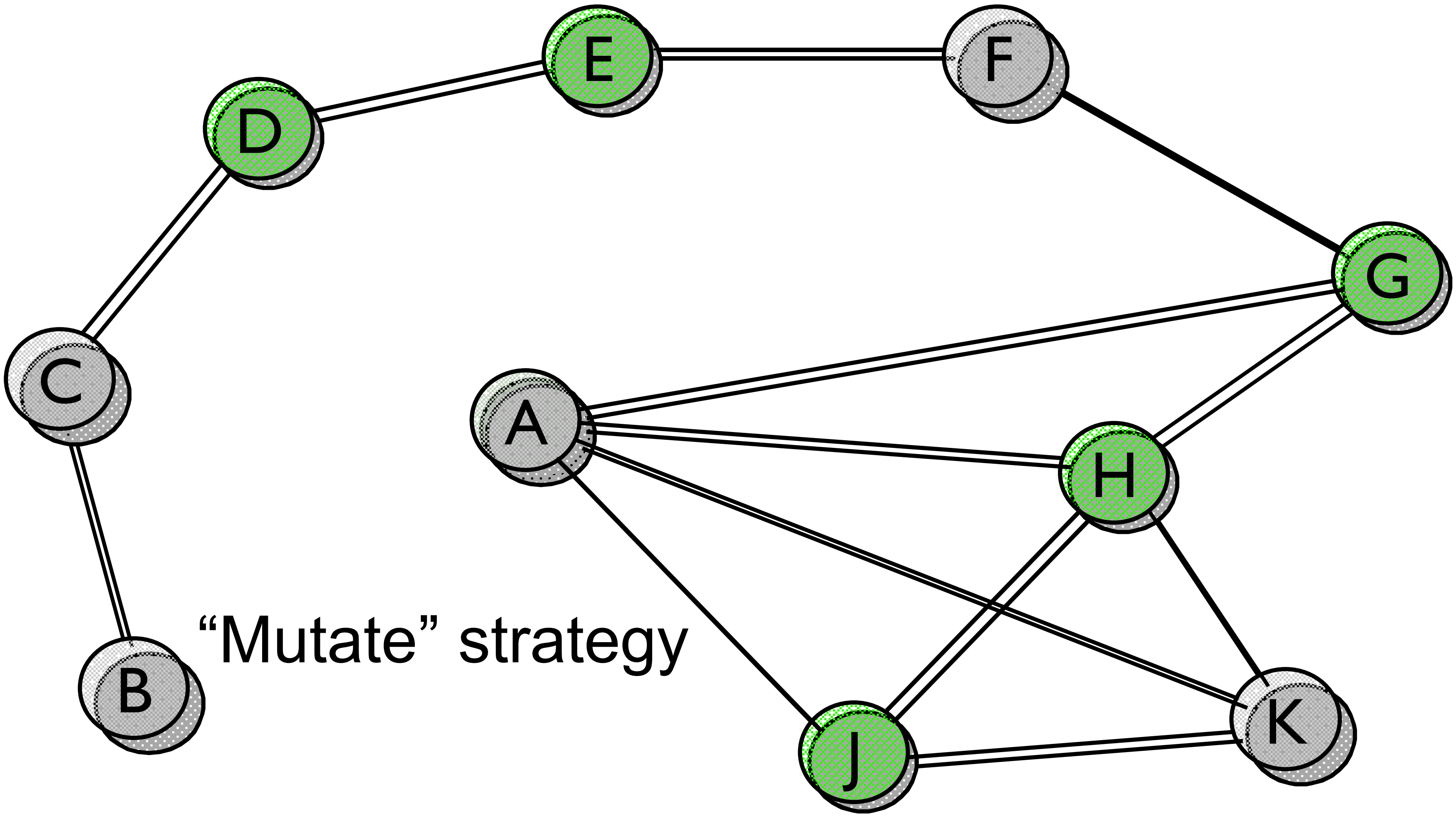
SLAC Algorithm: "Copy and Rewire"



SLAC Algorithm: “Mutate”

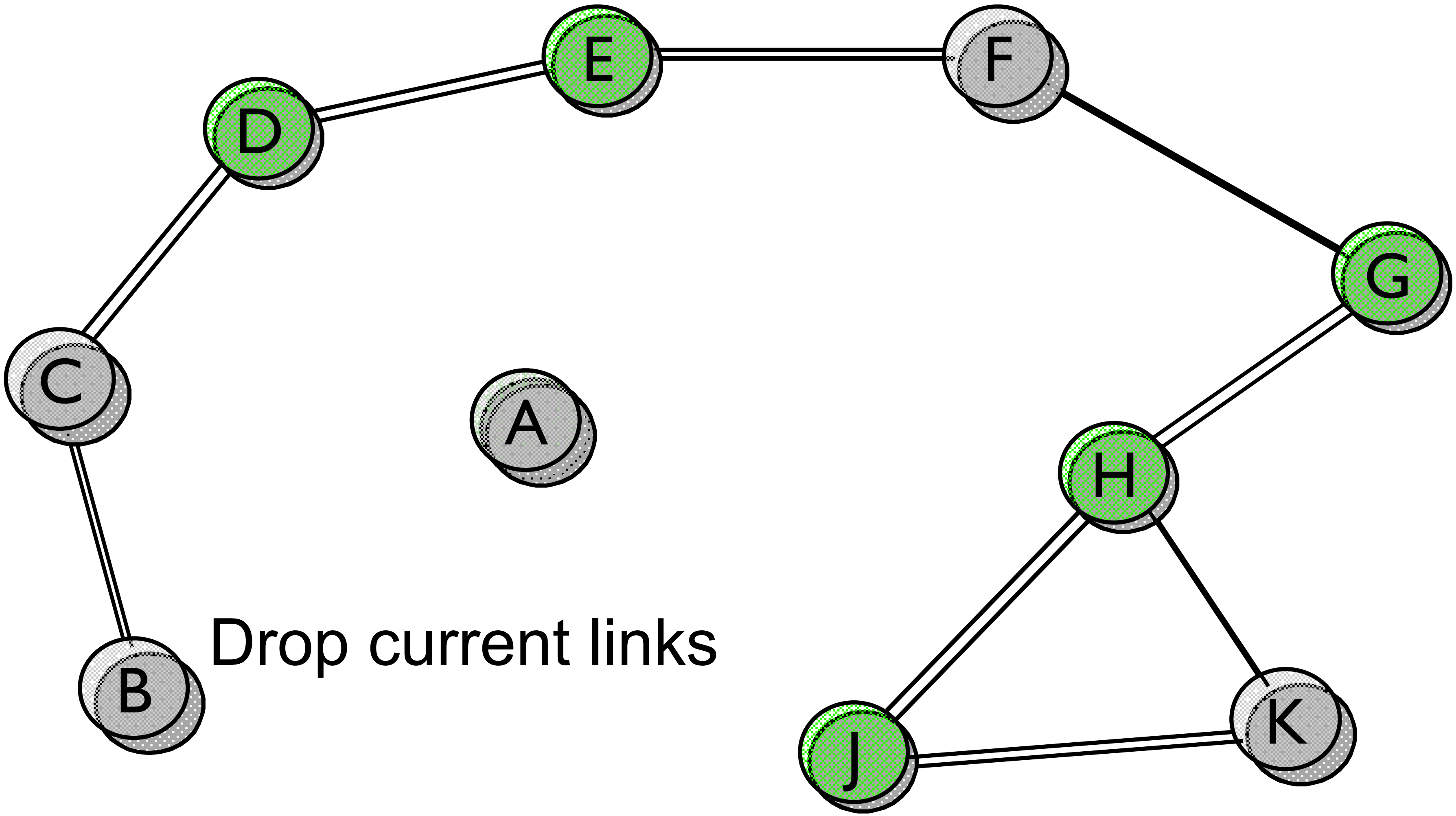


SLAC Algorithm: “Mutate”



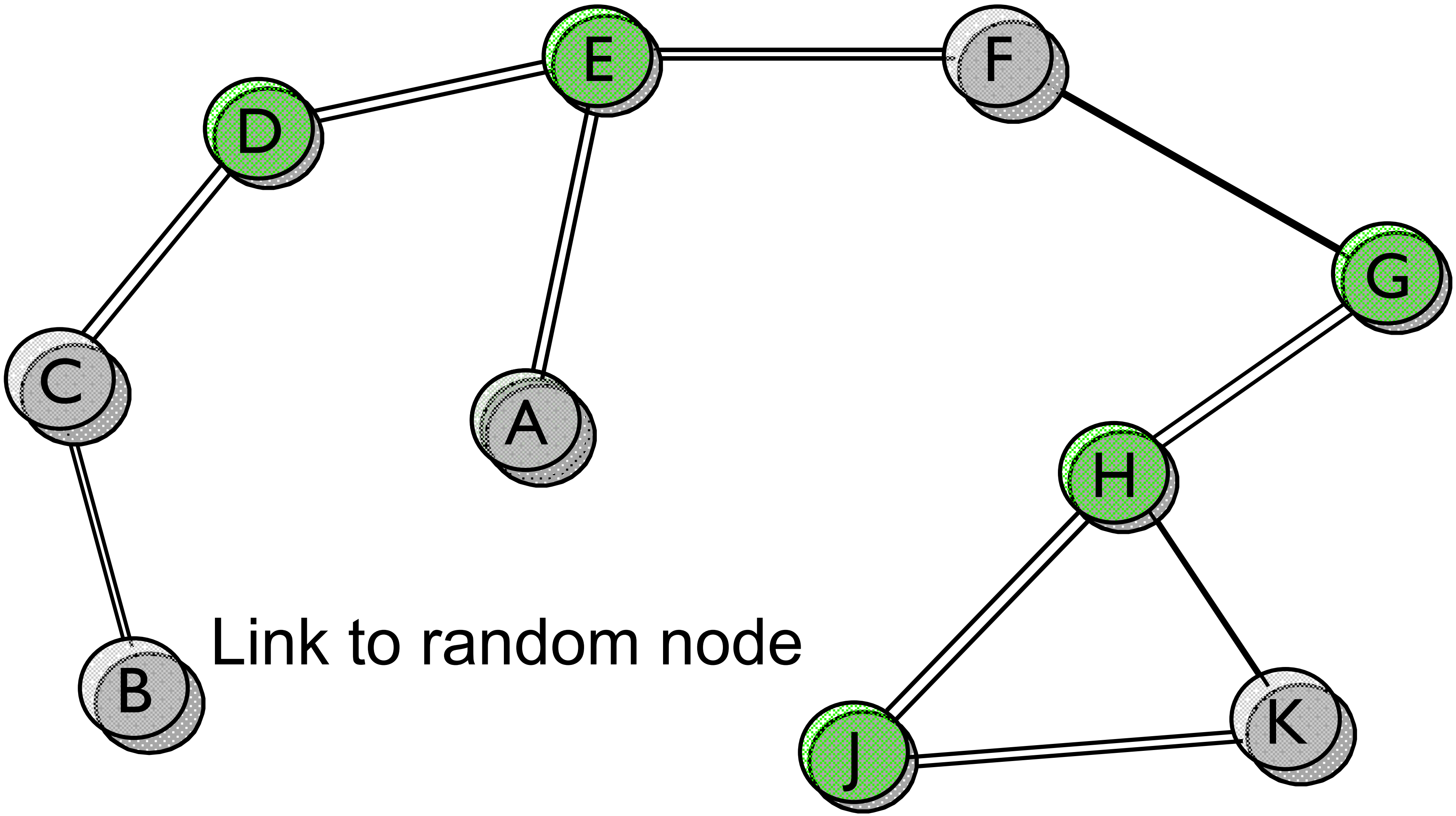
“Mutate” strategy

SLAC Algorithm: "Mutate"



Drop current links

SLAC Algorithm: "Mutate"



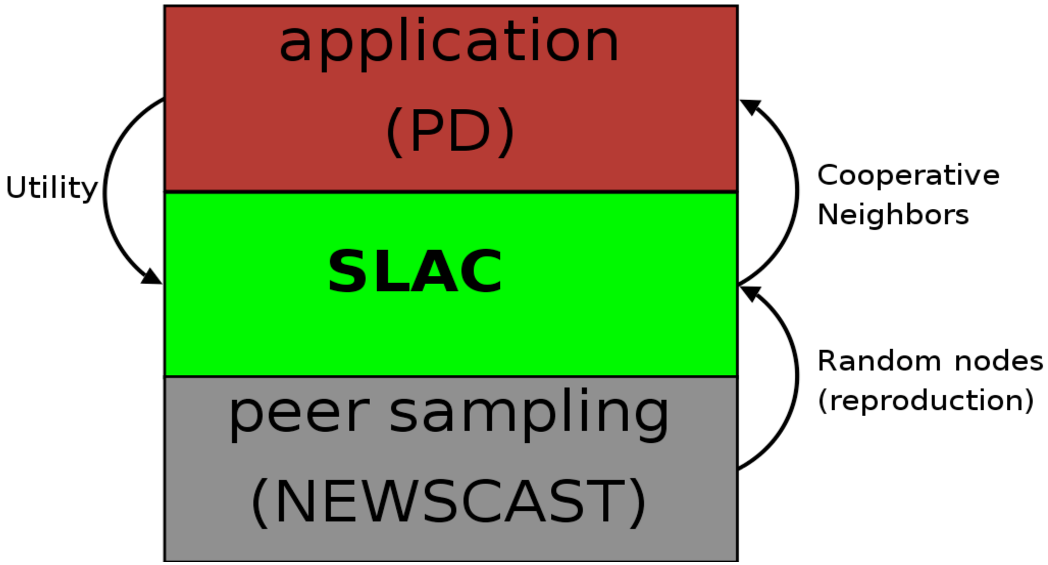
Link to random node

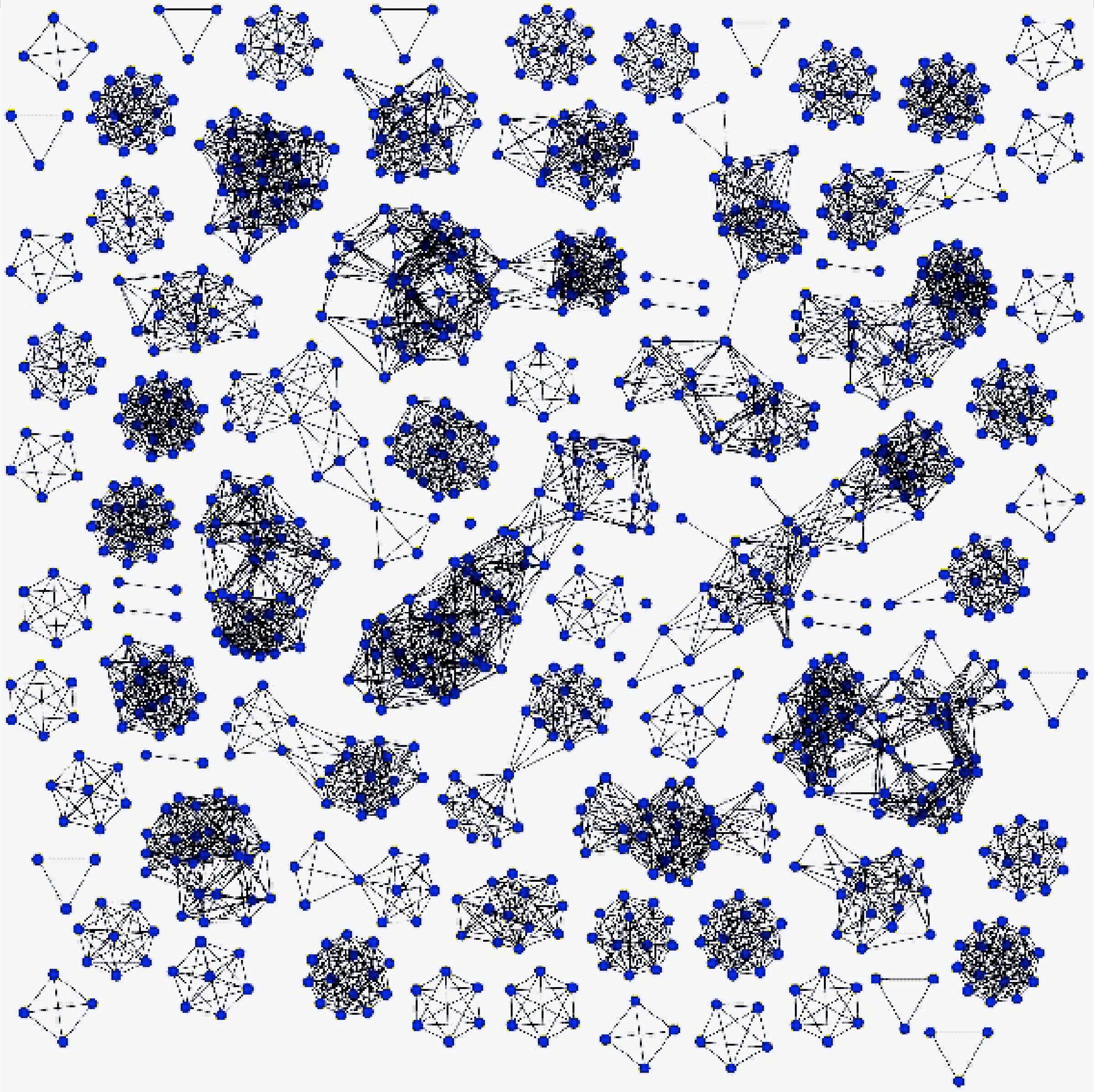
Prisoner’s Dilemma

- We tested SLAC with Prisoner’s Dilemma (PD)
  - Captures the conflict between “individual rationality” and “common good”
  - Defection (*D*) leads to higher *individual* utility
  - Cooperation (*C*) leads to higher *global* utility
  - *DC* > *CC* > *DD* > *CD*
- Prisoner’s Dilemma in SLAC
  - Nodes play PD with neighbors chosen randomly in the interaction network
  - Only pure strategies (always *C* or always *D*)
  - Strategy mutation: flip current strategy
  - Utility: average payoff achieved

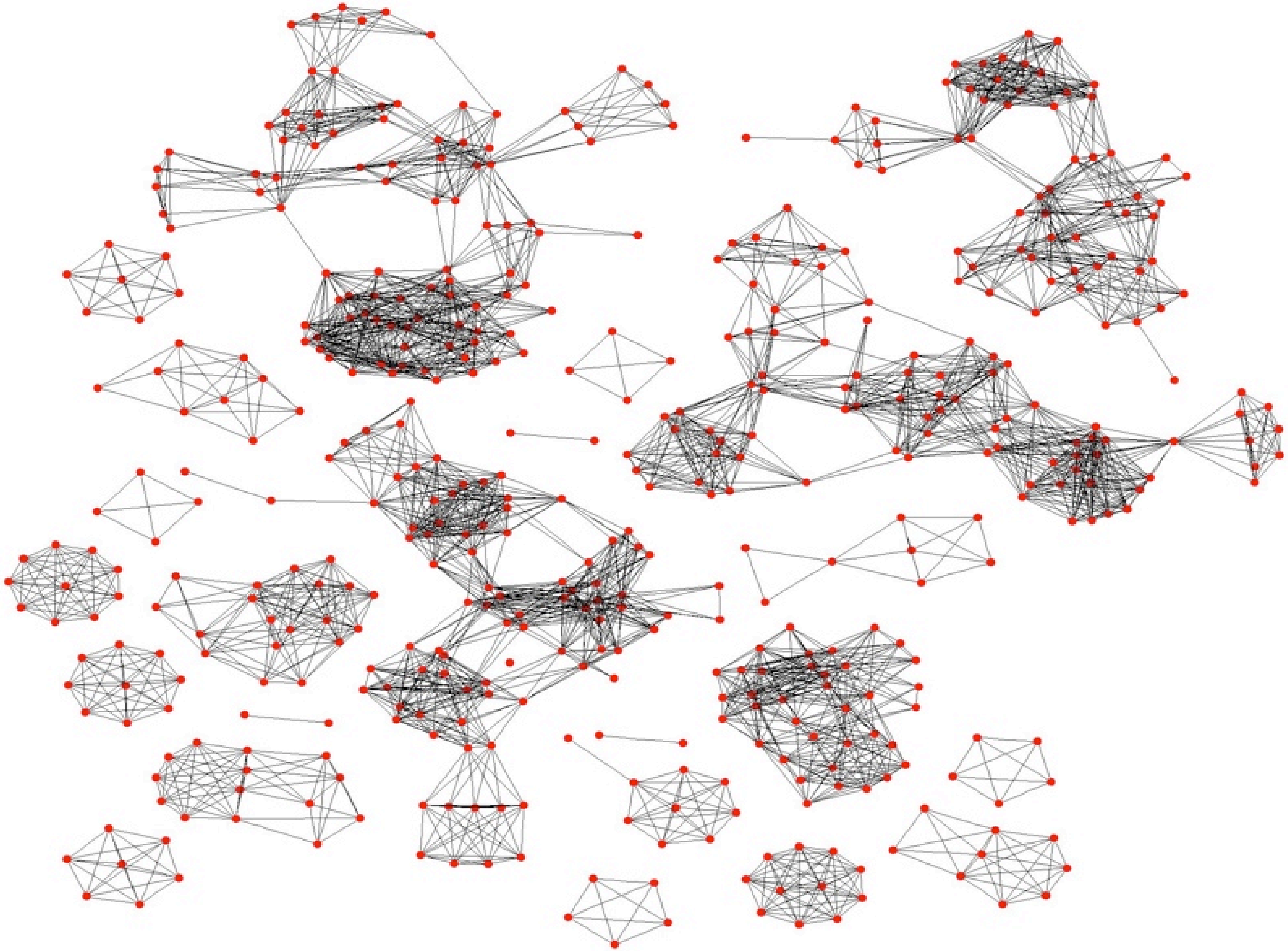
SLAC Architecture

- 3 layers architecture
- Random sampling
  - Newscast
- Cooperation and topology
  - Slac
- Application task
  - PD

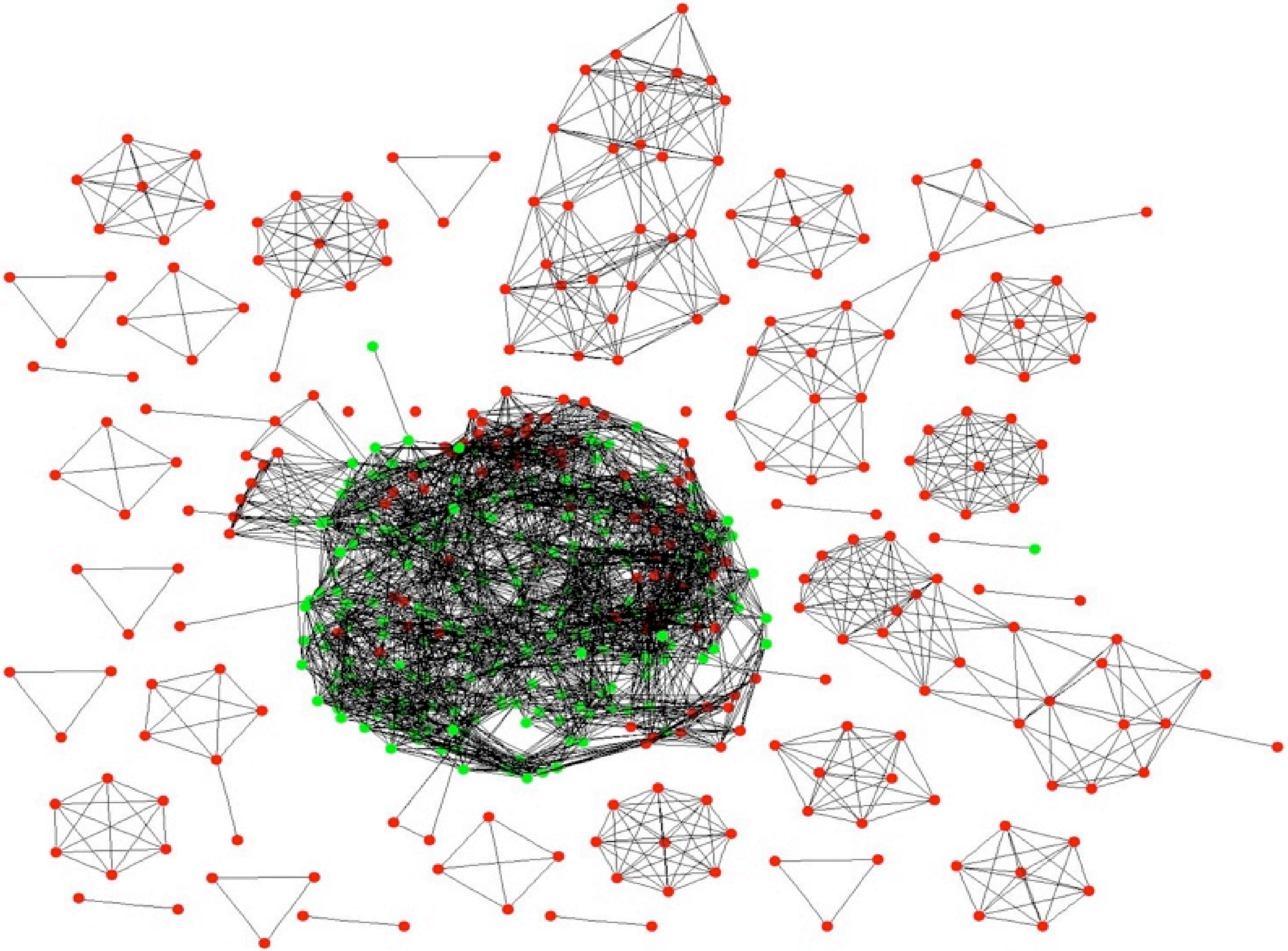




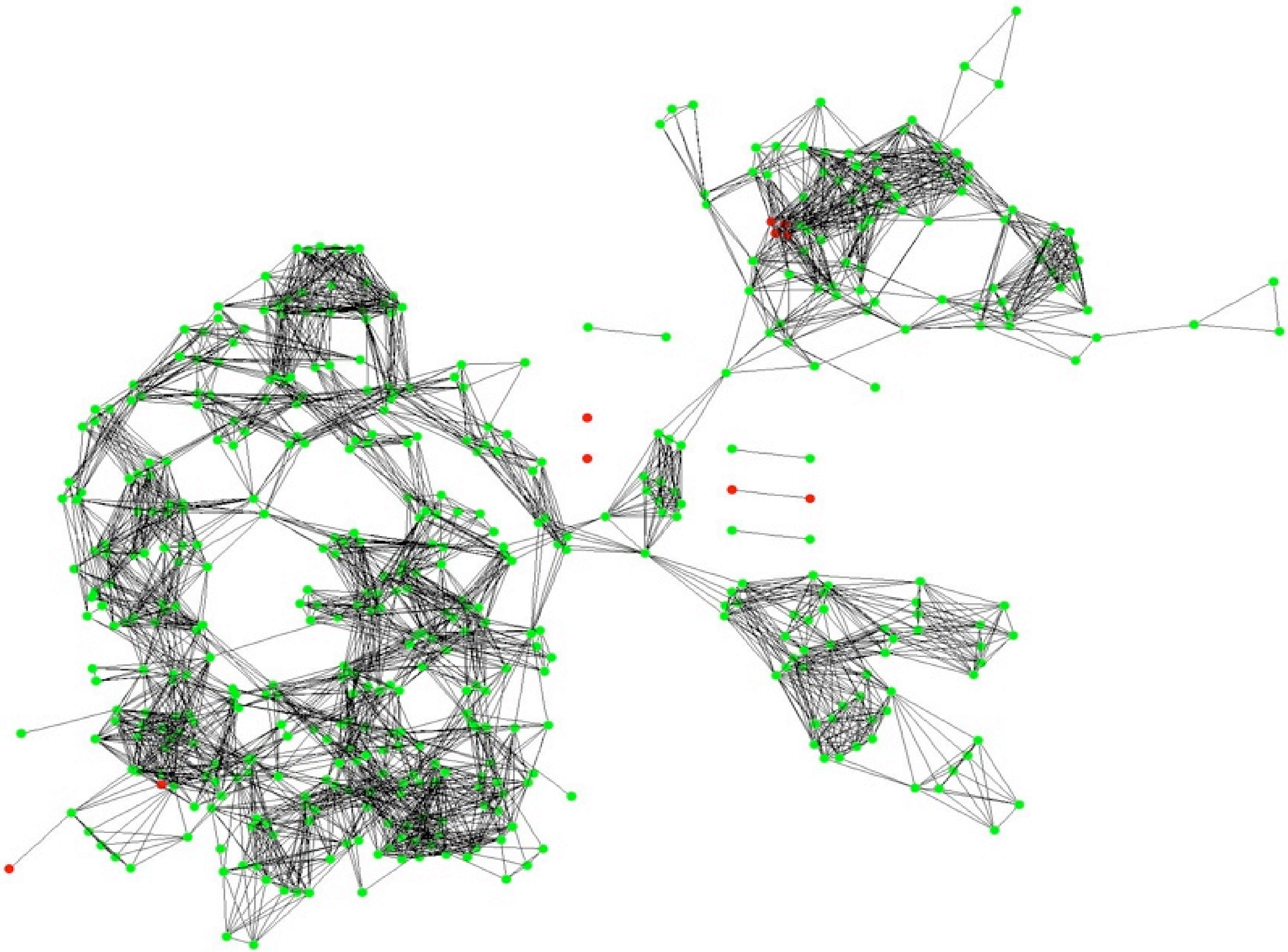
Cycle 180: Small Defective Clusters



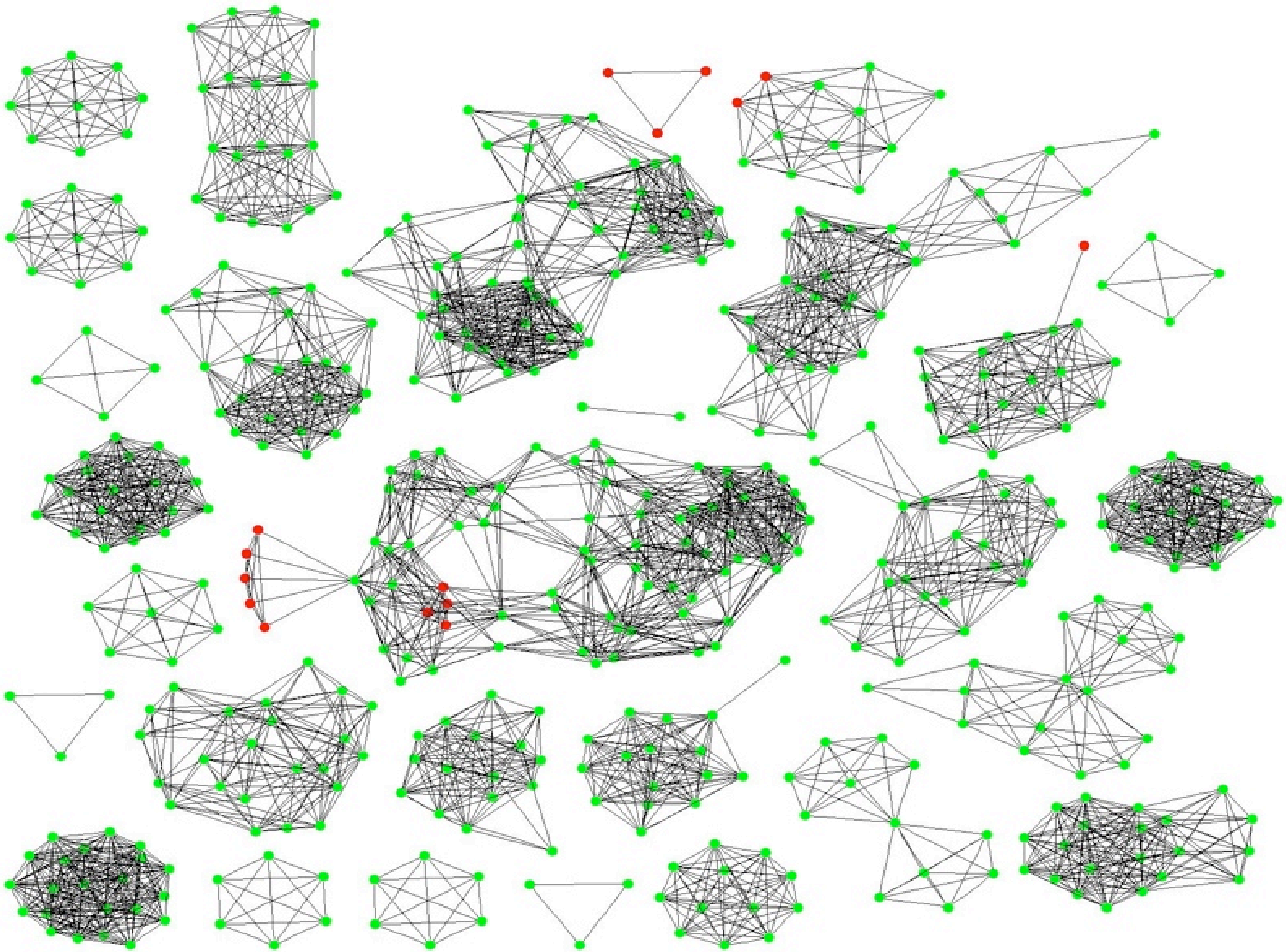
Cycle 220: Cooperation Emerges



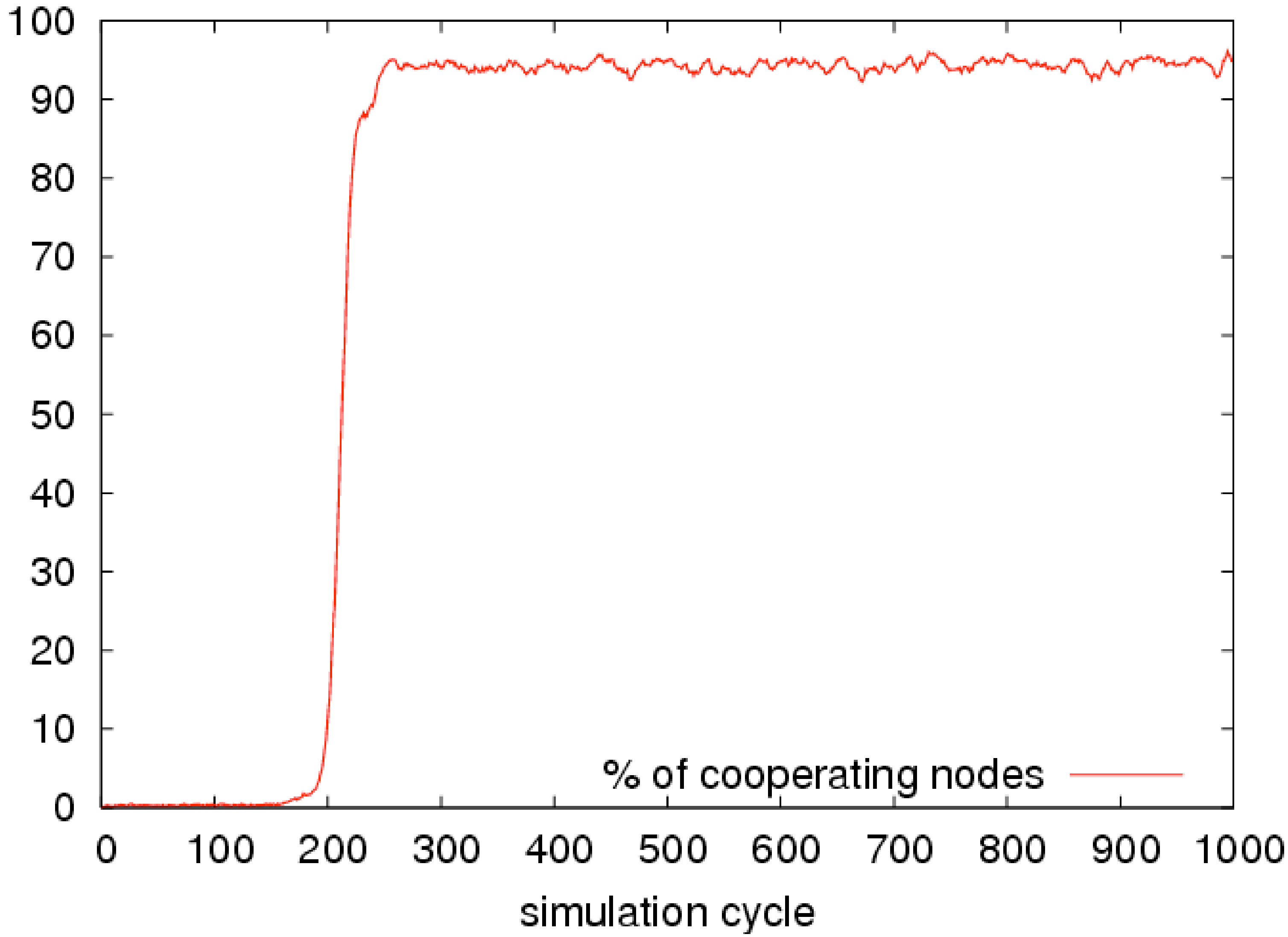
Cycle 230: Cooperating Cluster Starts to Break Apart



Cycle 300: Defective Nodes Isolated, Small Cooperative Clusters Formed



Cooperation Trend



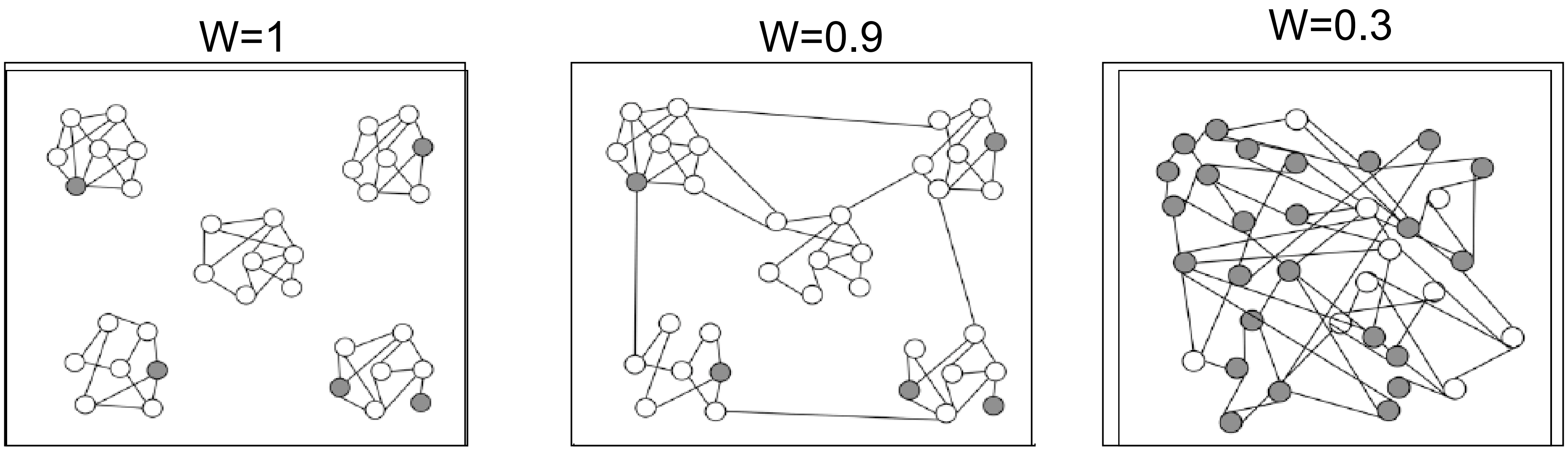
Why Does SLAC Work?

- SLAC produces very high levels of cooperation
- Nodes “move” throughout the network to find better neighborhoods
- This results in an evolution of the (interaction) network
- Group-like selection between clusters
  - Clusters of cooperating nodes grow and persist
  - Defecting nodes tend to become isolated

Tradeoff between topology and cooperation

UniBO, UPF, Telenor, RAL

- SLAC rewiring mechanism lead to high level of network partitioning
- SLACER: When isolating nodes not all the links are drop. Each link is dropped with given probability *W*
- Parameter *W* represents a tradeoff between network randomness and cooperation level
  - *W*=1: high cooperation, high partitioning
  - *W*=0.9: high cooperation, small world like topology
  - Low *W*: low cooperation, random like topology



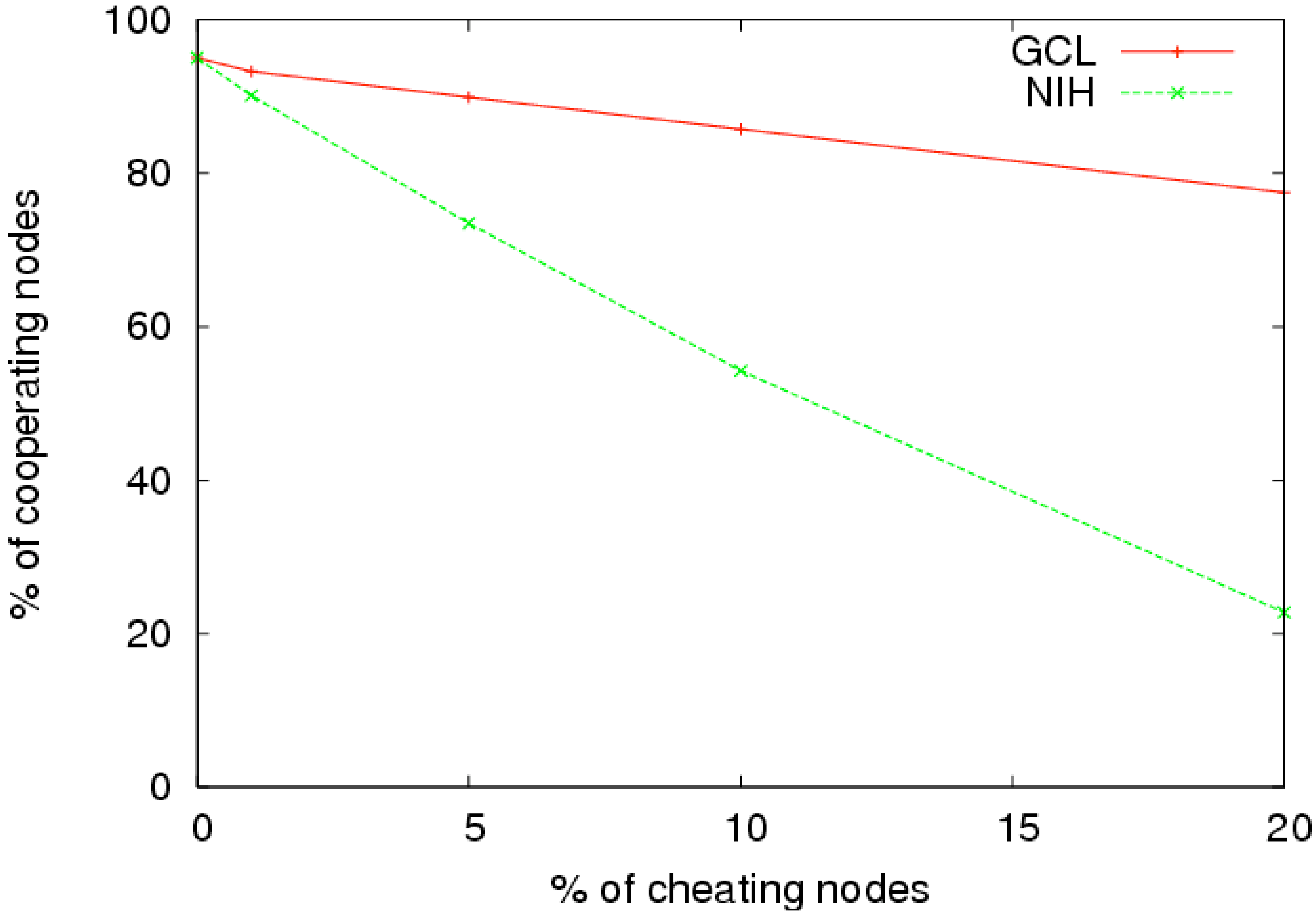
Cheating in SLACER

- SLACER requires nodes to *honestly* report their states (strategy, utility, links)
- What happens if some of the nodes lie in an effort to cheat the system? Will this destroy cooperation?
- We consider two types of cheating:
  - Greedy Cheating Liars (GCL) that want to exploit the system in order to increase their utilities
  - Nihilists (NIH) that want to destroy cooperation in the system and don't care about their own utilities

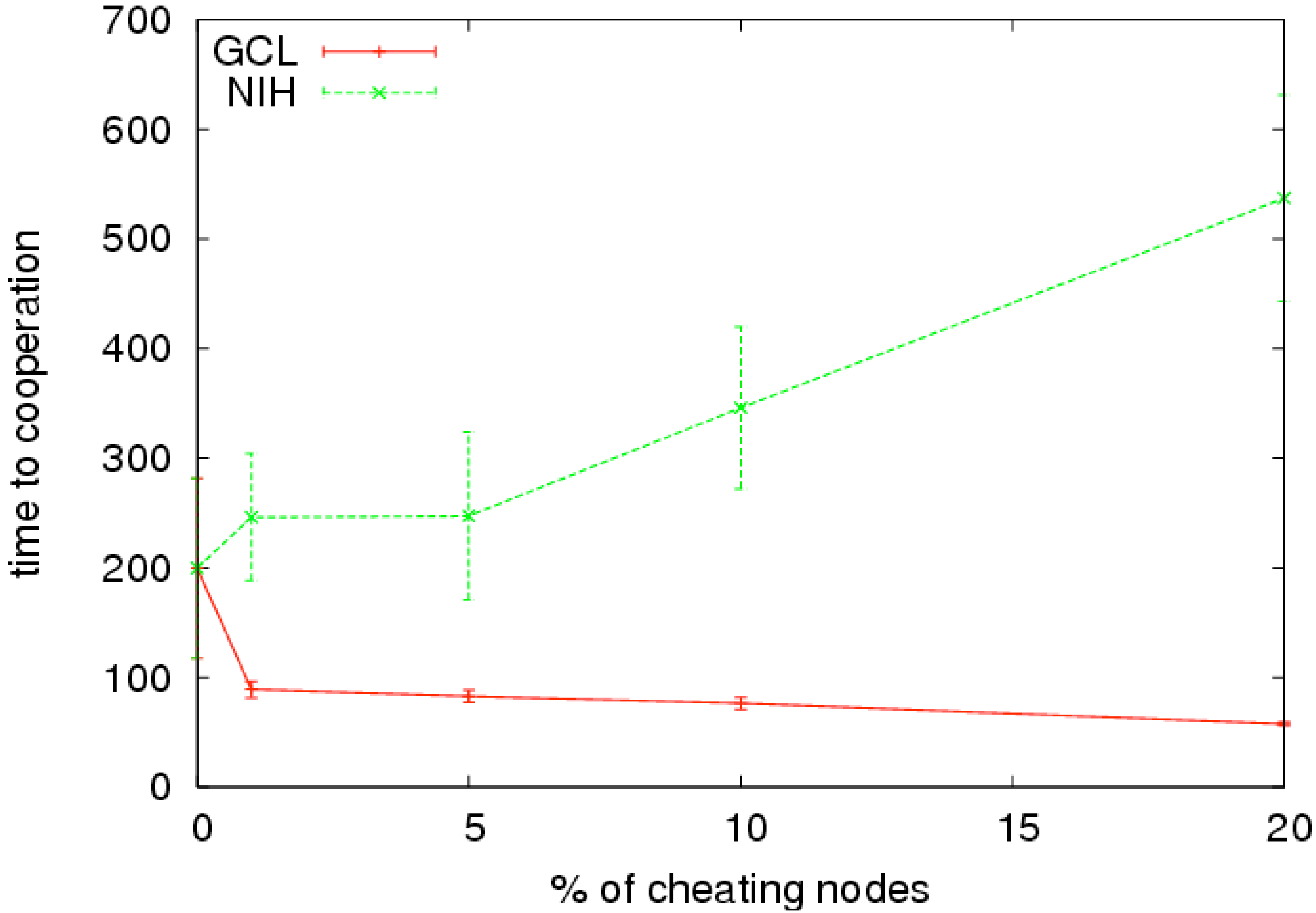
Greedy Cheating Liars

- GCL nodes:
  - Always report high utility (lying)
  - Always report strategy *C* (lying)
  - Always play strategy *D*
  - Move away when they are not completely surrounded by cooperators (i.e. utility < T)
  - In this manner, GCL nodes try to surround themselves with cooperating nodes (”suckers”) to exploit
- NIH nodes:
  - Always report high utility (lying)
  - Always report strategy *D*
  - Always play strategy *D*
  - Move away when they are surrounded by only defectors (i.e. utility = P)
  - In this manner, NIH nodes try to turn cooperating nodes to defectors then move away

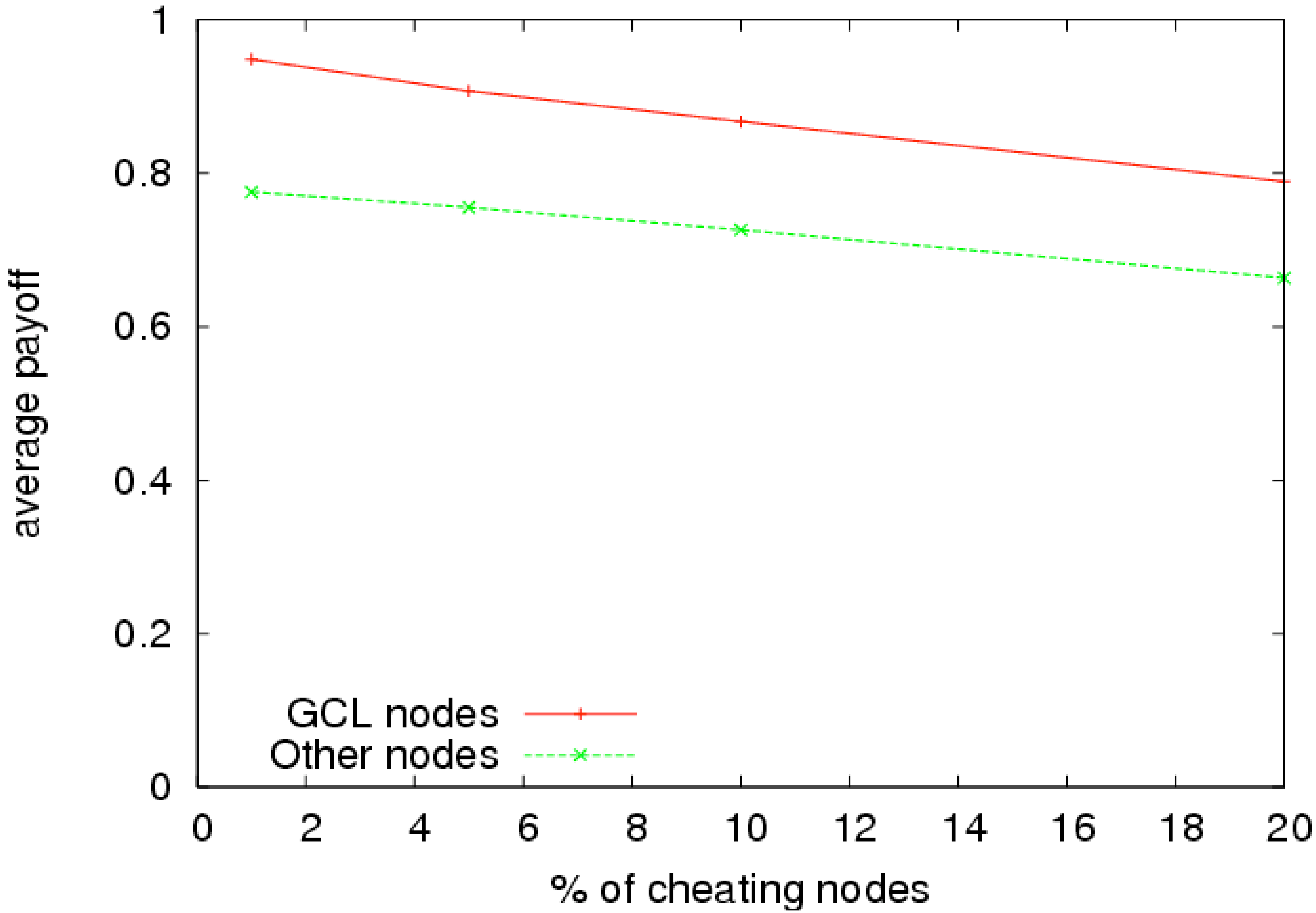
Cooperation in the Presence of Cheaters



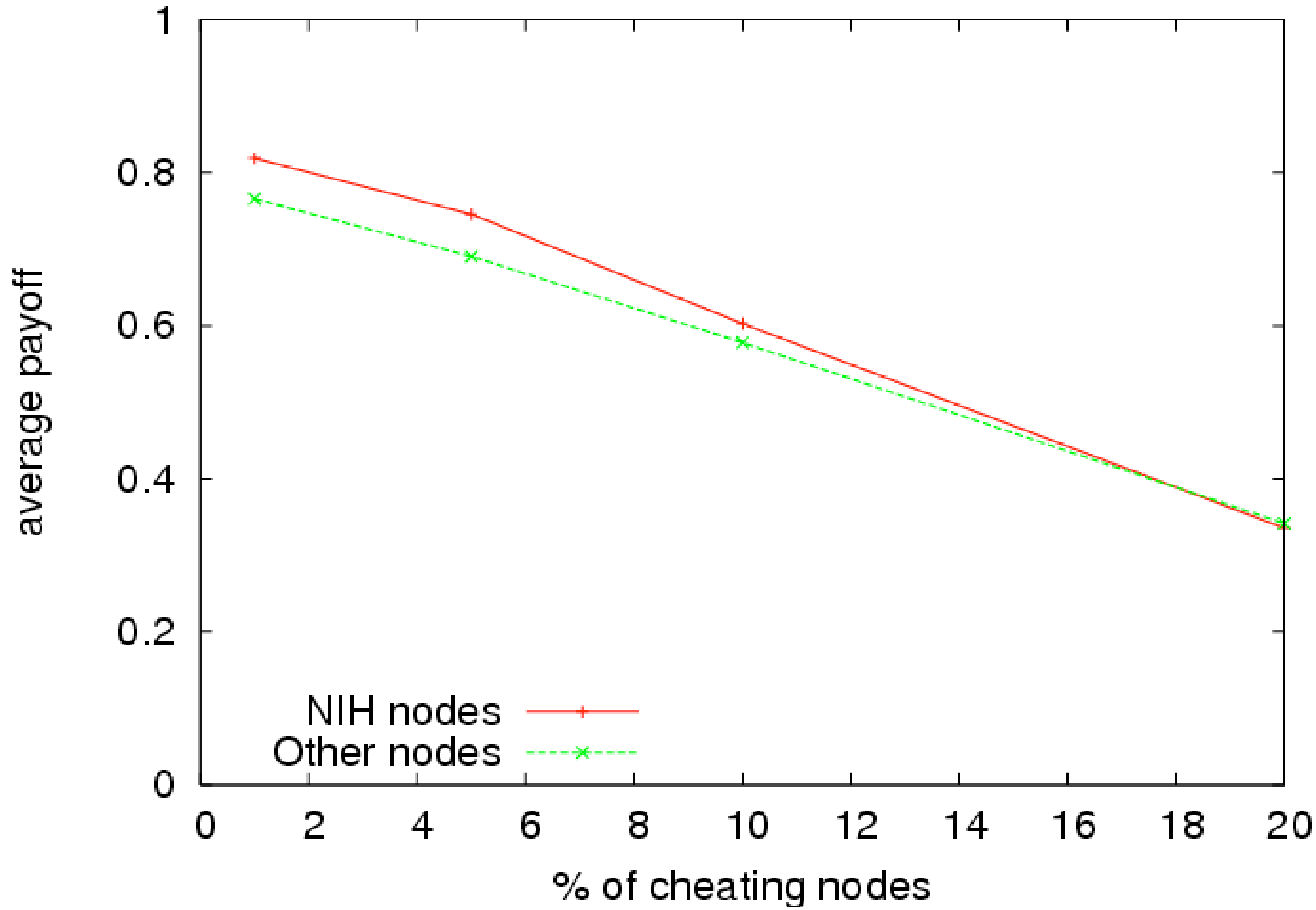
Time to Cooperation with Cheaters



Utilities in the Presence of GCL Nodes



Utilities in the Presence of NIH Nodes



A Few Comments

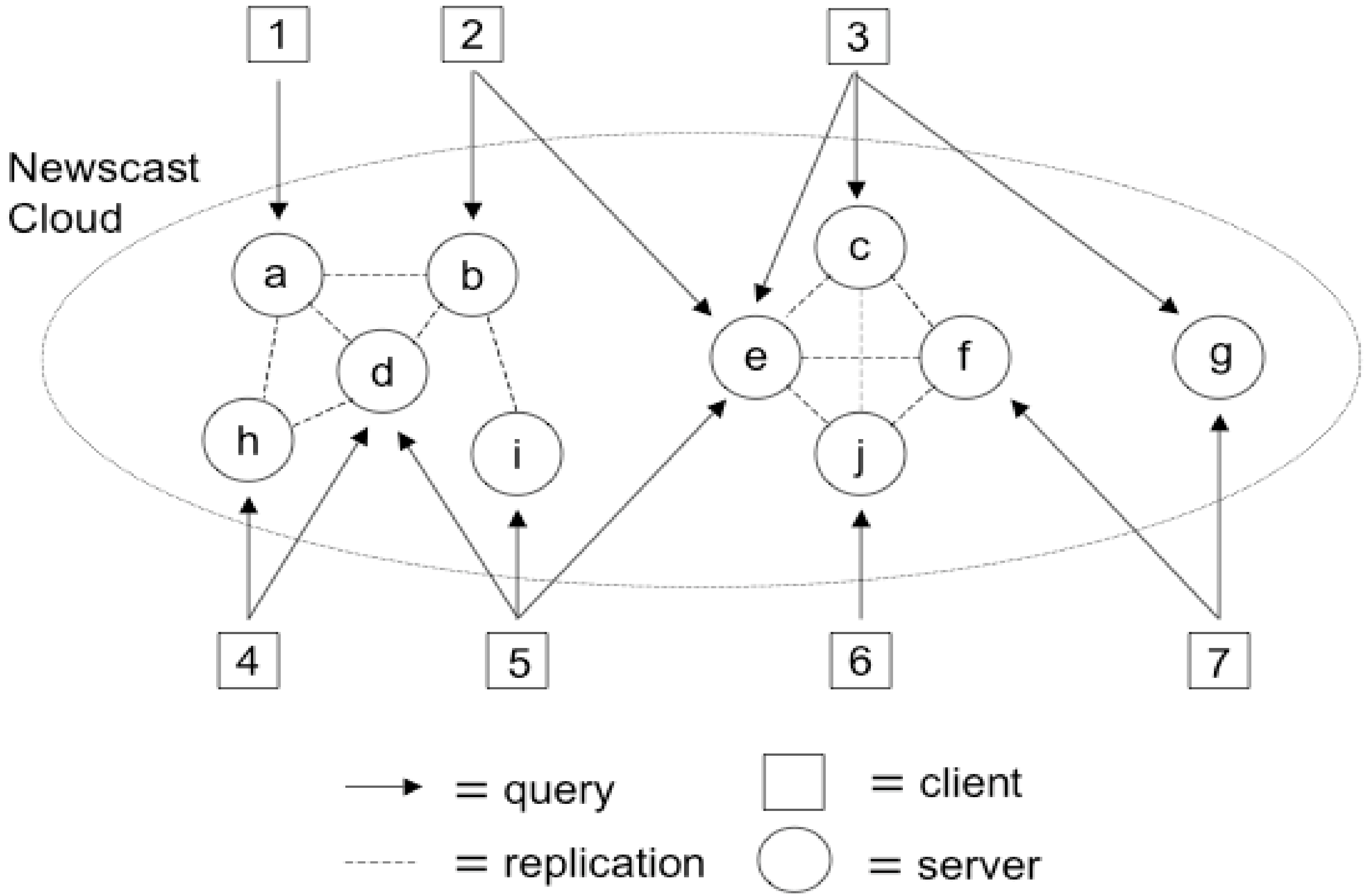
- SLACER can tolerate a high percentage of GCL nodes
- GCL nodes degrade global performance gracefully
- Yet, NIH nodes degrade performance significantly
- Interestingly, increasing percentage of GCL nodes *decreases* the time to cooperation
- GCLs might be seen as “taxing” the general population in return for more rapid cooperation
- Perhaps protocols can be designed to function despite cheating nodes rather than strive to detect and block them

Distinction between Free-Riding and Cheating

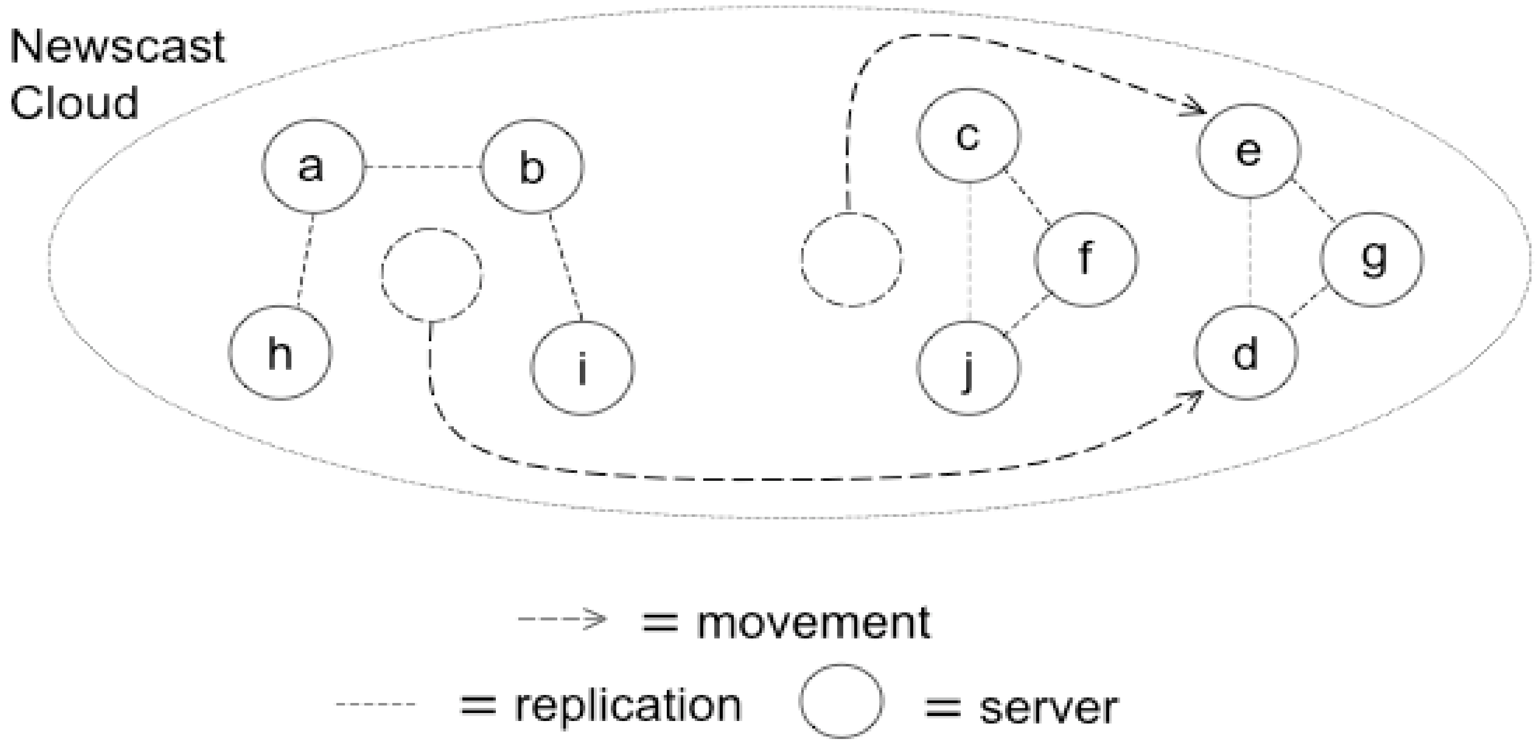
- Copying (and mutation) applied to normal behavior
- Cheating behavior limited to a (fixed) percentage of nodes and does not spread
- “Normal behavior” (including possible free-riding) akin to running good clients in a P2P system (like BitTorrent)
- “Cheating behavior” akin to running hacked versions of the P2P client
- Typically, these hacked versions remain limited to a small group “in the know” and are not made widely available to others

CacheWorld – Cooperative Content Replication

- Cooperative Content Replication
- Assuming protocols for
  - Replicating content between nodes
  - Redirecting queries (requests for content) between nodes
  - Peer sampling over a population of nodes
- Simple protocol for cooperatively coordinating these services to maximise system capacity
- With incentives for nodes to cooperate
- Dynamically adjusting to varying load and node entry and exit


1
2
3
Newscast
Cloud
a
b
c
d
e
f
g
h
i
j
4
5
6
7
= query
= replication
= client
= server

WP5.3: Biologically and socially inspired design for dynamic solution spaces

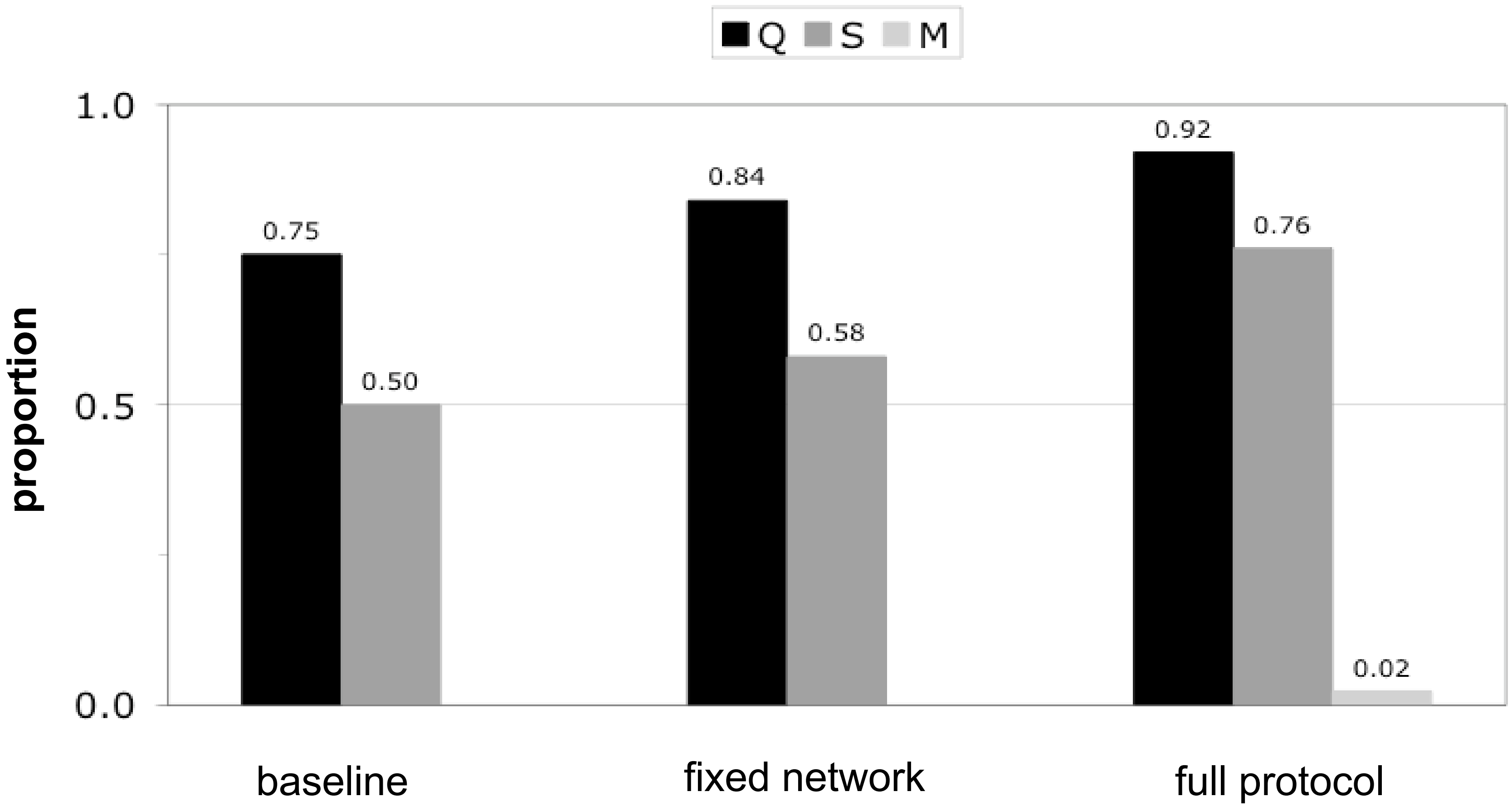


CacheWorld outline protocol

Passive thread	Active thread
on receiving a query q, node i: if not overloaded service q directly else if neighbors > 0 and q is not already a redirected query j ← selectRandomNeighbor() redirect q to j end if	periodically each node i: if not satisfied drop all neighbor links if Ci < directly received queries j ← selectRandomPeer(); link to j's neighbors and j end if end if

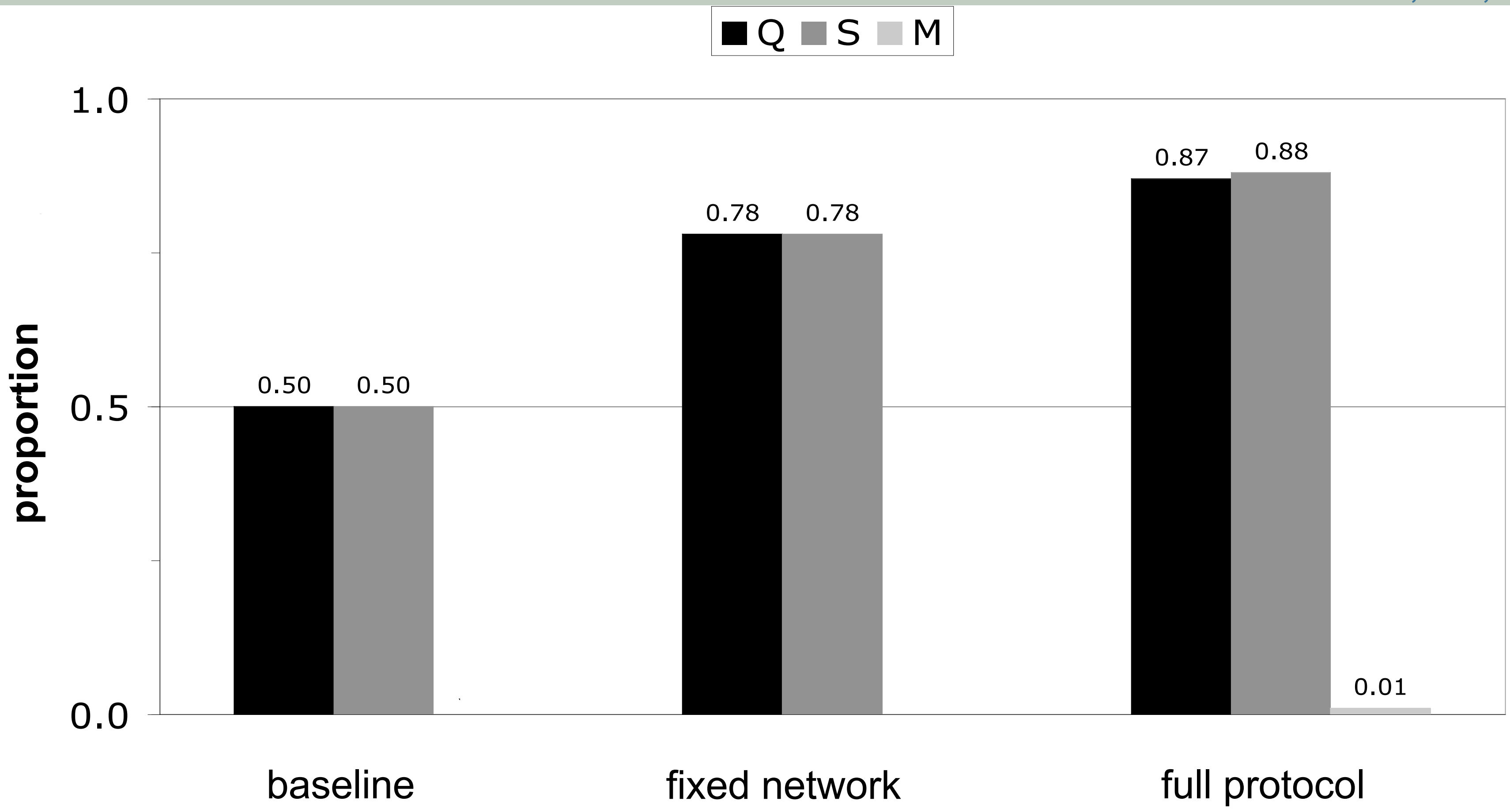
- Capacity (C) and load for each node specify different scenarios
- maximum number of neighbours (k) currently defined exogenously (typically low, k < 10)
- nodes are satisfied if all queries submitted to them are answered (over a given period - the load cycle)
- each node associated with a single unique content item that is replicated between linked neighbours

WP5.3: Biologically and socially inspired design for dynamic solution spaces



Q = queries answered, S = satisfied nodes, M = movement

(very simple scenario, half nodes underloaded, half overloaded, k = 1)



Q = queries answered, S = satisfied nodes, M = movement

(less simple scenario, half nodes underloaded, half overloaded, k = 4)

CacheWorld Summary

- Very initial results, with simple load / capacity scenarios
  - Nodes replicate and serve a single “content” item
  - Not modelling cost of replication process
  - Fixed loads and capacities
- Hence more realistic scenarios needed and comparison with existing protocols (on-going work with RAL)
- Still not tested with malicious nodes, pure freeriders and churn. But reasonably confident will degrade gracefully
- Current results not has good has hoped, however
- Experimenting with
  - conditional acceptance of new links (e.g. only if node is under or overloaded)
  - simple “loyalty” approach (where preference is given to older links) could lead to much better results but this is on-going. Interesting this could link to a lot of work from “evolutionary economics” (Kirman’s Marseille Fish Market studies / models)

ASB Idea

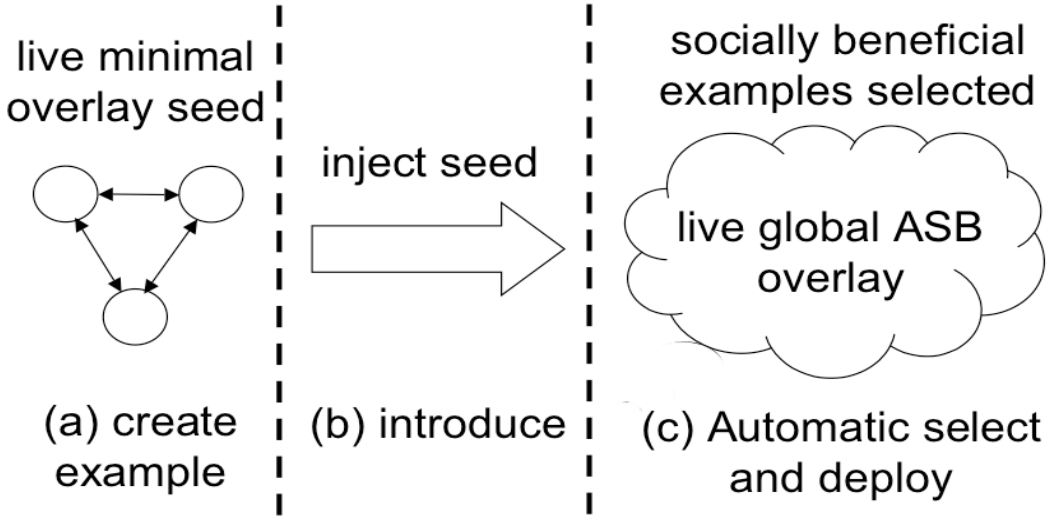
- Current problem of open P2P protocol deployment
- Requires massive user intervention (client discovery, download, evaluation)
- Out-protocol communication concerning client discovery
- Idea: Automatic Social Bootstrapping (ASB)
- A meta P2P protocol
- Dynamic (run time) deployment of P2P protocols
- Automatic selection of protocols that are socially beneficial (increase the collective utility of nodes)
- Given some application supplied utility
- For a given API

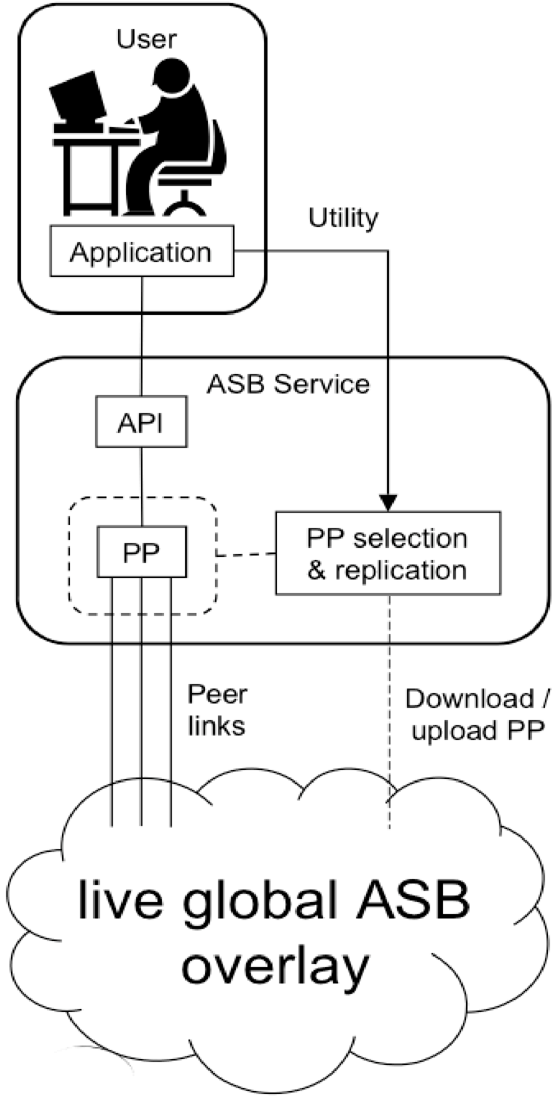
WP5.3: Biologically and socially inspired design for dynamic solution spaces

UniBO, UPF, Telenor, RAL

Periodically each node:

$i \leftarrow$ this node
$j \leftarrow$ SelectRandomNode() // select a random node j
if $U_j \gg U_i$ then // utility of j higher?
  $PP.protocol_i \leftarrow PP.protocol_j$ // copy protocol of j to i add j
  $PP.view_i \leftarrow PP.view_j \cup j$ // copy view of j to i add j





WP5.3 Summary

- Begun to develop a method for adapting models toward implementation
- Initial work on broadcast and cooperative content replication looks promising
- Publications:
  - Hales, D. and Arteconi, S. (2006) SLACER: A Self-Organizing Protocol for Coordination in P2P Networks. IEEE Intelligent Systems 21(2):29-35
  - Hales, D. and Babaoglu, O. (2006) Towards Automatic Social Bootstrapping of Peer-to-Peer Protocols. ACM SIGOPS Operating Systems Review 40(3)
  - Arteconi, S., Hales, D., Babaoglu, O. (submitted) Greedy Cheating Liars and the Fools Who Believe Them. CMOT Journal
  - Arteconi, S, Hales, D., Babaoglu, O. (2006) Broadcasting at the Critical Threshold. DELIS-TR-0373. To be submitted to SASO 2007
- Future: More realistic scenarios, experimentation with malicious nodes, possible implementation, further apps. from D.5.5.1 (D5.3.2, month 48)

WP5.4 Multi-Scale topology evolution in natural and artificial networks

Goals (start month 13)

General

Explore processes of general network evolution in both natural and artificial systems - determine and harness both the form and function of multi-level evolution for engineering

Specific

Develop dynamical analysis techniques for evolving networks. Identify natural and artificial networks that demonstrate selection / topology evolution at different levels. Relate network “forms” to desirable network “functions”

Partners

UPF, UniBO, Telenor

Results (from D5.4.2)

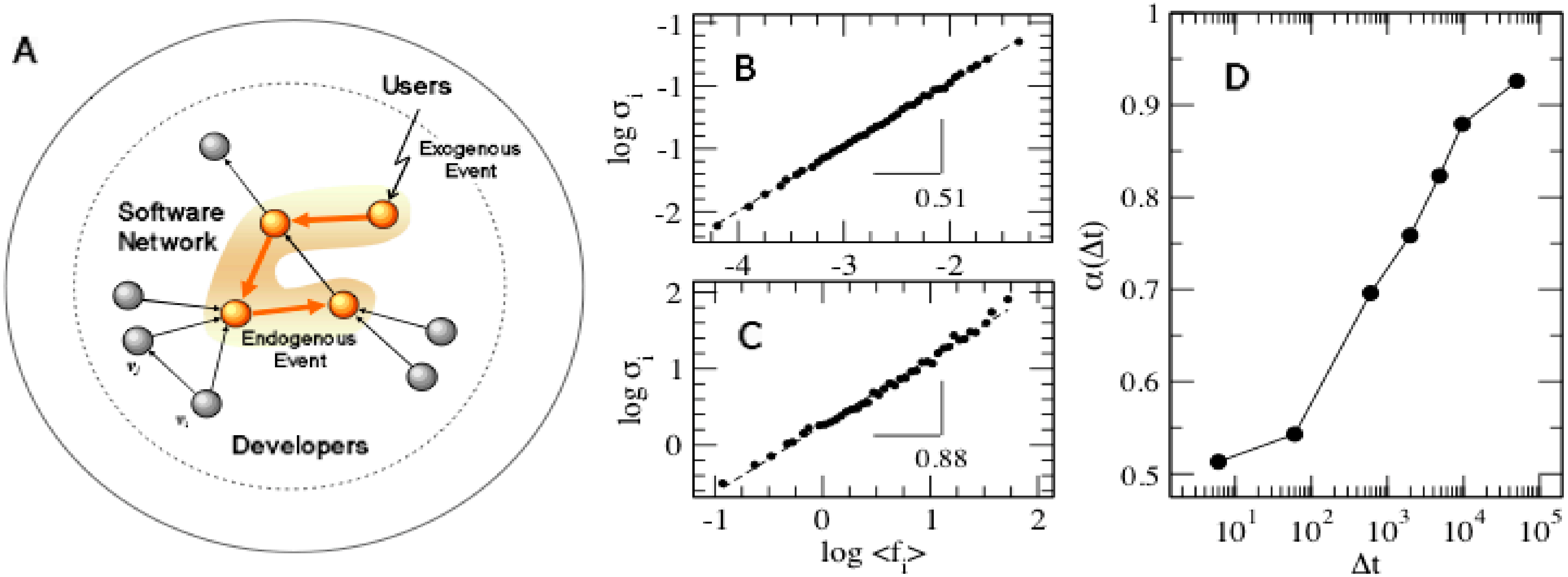
- Understanding Multi-scale Evolution of Open Source Software
  - Analysis of CVS databases to aid understanding of software development process in OS systems
  - Evidence of endogenous and exogenous change at different time-scales
  - Method for determining files likely to change together
  - Might be useful when refactoring legacy software
- Relating models of emergent multi-level selection
  - Framework for comparison of “group selection” models
  - Previous SLACER and SkillWord models (from SP4/SP5)
  - Recent novel bio / socio group selection models
  - Possibility of generalised design pattern(s) with conditions of application

Change in Open Source Projects

- Analysis of CVS files of Open Source projects
- Measuring patterns of file changes over different timescales
- Evidence that short-term changes (order of hours) follow an endogenous process similar to that found in circuits / physical internet
- Long-term changes (order of year) follow an exogenous (user driven) process similar to that found in human systems
- Appears to be two distinct kinds of change process that can be identified
- Same tools can be used to identify sets of files that change together - potentially the modules of the system

WP5.4 Multi-Scale topology evolution in natural and artificial networks

UniBO, UPF, Telenor, RAL



(A) Schematic representation of an OSS community. (B) Scaling of fluctuations with average change activity for the software project XFree86 at Δt = 6 hours (top) and Δt = 9600 hours (bottom). (D) Dependence of the observed α exponent with the measurement window Δt.

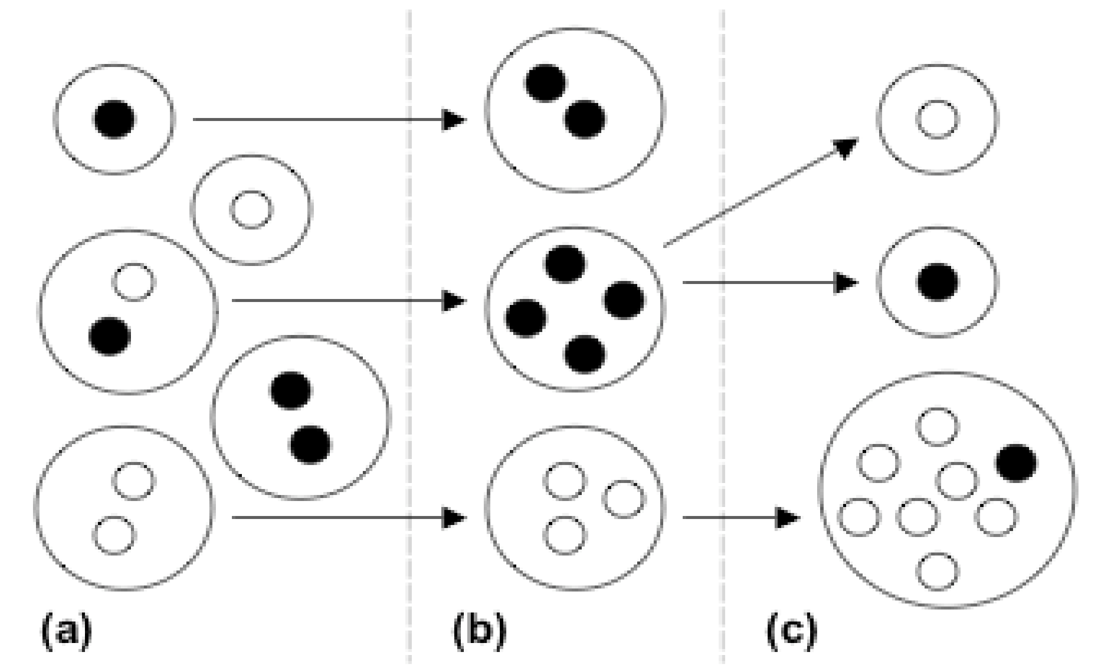
Group Selection Framework

Previous models from SP5 and novel group selection models are comparable within the following framework:

- *Group boundary* - a mechanism which restricts interactions between agents such that the population is partitioned into groups
- *Group formation* - a process which forms groups dynamically in the population
- *Migration* - a process by which agents may move between different groups
- *Conditions* - cost / benefit ratio of individual interactions and other conditions which are sufficient for producing group-level selection

WP5.4 Multi-Scale topology evolution in natural and artificial networks

UniBO, UPF, Telenor, RAL



Outline algorithm for tag model:

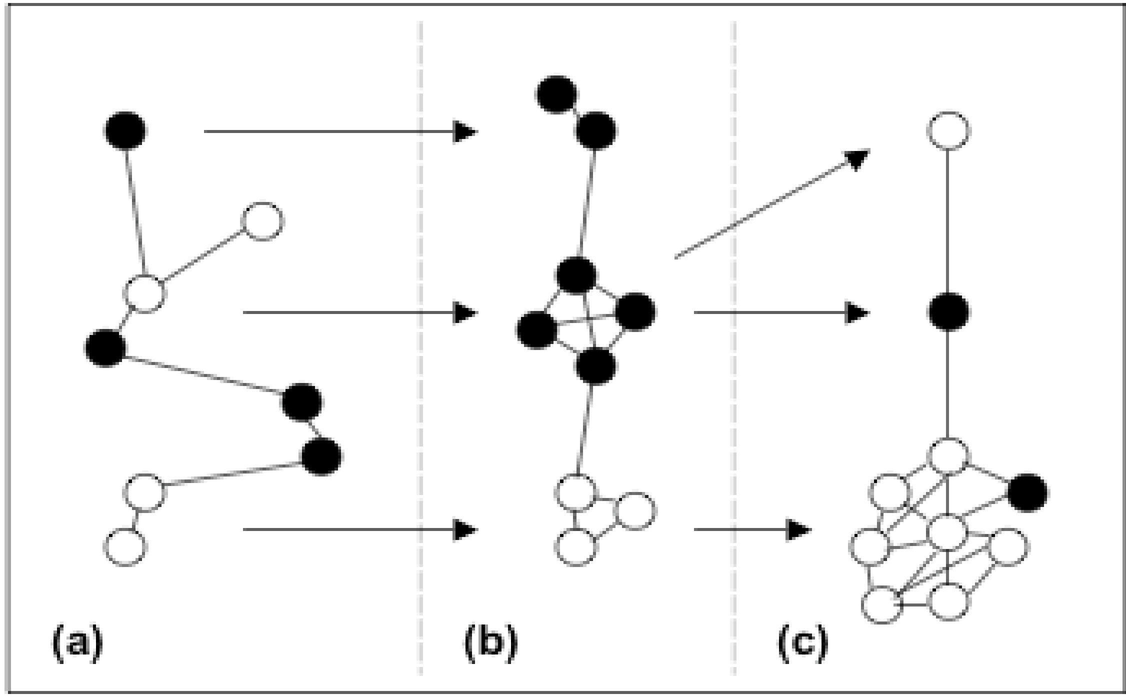
for each generation loop
interaction within groups (obtain fitness)
reproduce individuals based on fitness
with *Prob(mt)* individuals form new group
with *Prob(ms)* individuals flip strategy
end generation loop

Group boundary: tag stored by each individual defines group membership
Group formation and migration: probabilistic mutation of tag

Schematic of the evolution of groups in the tag model. Three generations (a-c) are shown. White individuals are pro-social (altruistic), black are selfish. Individuals sharing the same tag are shown clustered and bounded by large circles. Arrows indicate group linage. When ***b*** is the benefit a pro-social agent can confer on another and ***c*** is the cost to that agent then the condition for group selection of pro-social groups is:

b > c and mt >> ms

WP5.4 Multi-Scale topology evolution in natural and artificial networks



Outline algorithm for network model:

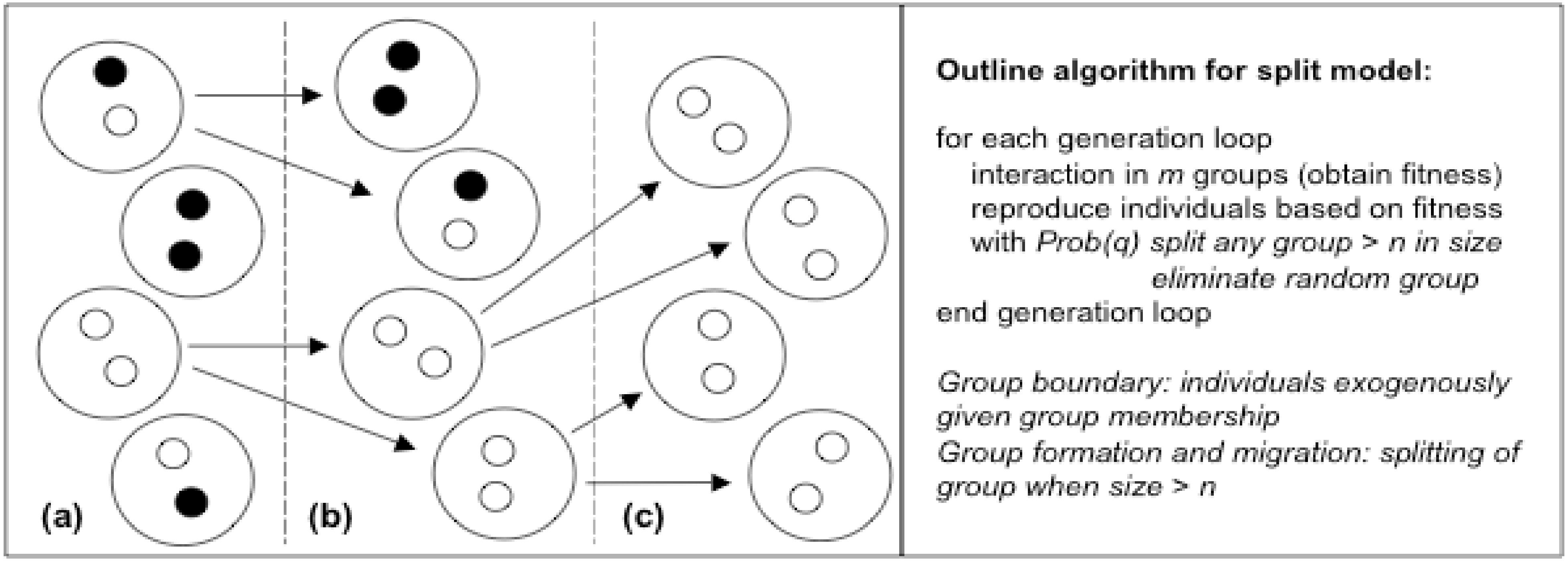
for each generation loop
    interaction within groups (obtain fitness)
    reproduce individuals based on fitness
    with *Prob(t) copy new links*
    with *Prob(mt)* individuals form new group
    with *Prob(ms)* individuals flip strategy
end generation loop

Group boundary: individuals directly linked in the network
Group formation and migration:copying of links probabilistically

Schematic of the evolution of groups in the network-rewire model. Three generations (a-c) are shown. Altruism selected when:***b > c and mt >> ms.*** When ***t = 1***, get disconnected components, when ***1 > t > 0.5***, get small-world networks

Santos F. C., Pacheco J. M., Lenaerts T. (2006) Cooperation prevails when individuals adjust their social ties. PLoS Comput Biol 2(10)

WP5.4 Multi-Scale topology evolution in natural and artificial networks



Schematic of the evolution of in the group-splitting model. Three generations (a-c) are shown. Altruism is selected if the population is partitioned into ***m*** groups of maximum size ***n*** and ***b / c > 1 + n / m.***

Traulsen, A. & Nowak, M. A. (2006). Evolution of cooperation by multilevel selection. Proceedings of the National Academy of Sciences 130(29):10952-10955.

WP5.4 Summary

- Automatic analysis of software development using CVS logs
- Framework for comparing group selection models, towards design pattern(s)
- Publications:
  - Valverde, S. (2006) Crossover from Endogenous to Exogenous Activity in Open-Source Software Development, accepted for publication in Europhysics Letters
  - Hales, D. (2006) Emergent Group-Level Selection in a Peer-to-Peer Network. Complexus 2006;3. 108-118.
- Future: Software metrics for OS development, group selection based design pattern(s), “motif” based network analysis (D5.4.3, month 48)

WP5.5 Industrial applications and knowledge transfer

Goals (start month 13)

General

Bridge between academic research (in DELIS SP5) and realities of industry (telecom). Identify possible ideas for patents, spin-offs, industrial projects

Specific

Identify activities and mechanisms with possible commercial and industrial applications

Partners

Telenor, UniBO, UPF

Results (from D5.5.1)

- Experience of patent process, R&D within a large telecoms corporation and formation of a recent spin-off company. Telenor.
- Barriers / opportunities for commercial exploitation. Academic v. commercial culture (peer recognition v. bottom line), low status of applications (often)
- Fully distributed power method. Possible application: PageRank and similar computations in distributed search engine (SP6). UniBo and Telenor
- Epidemic spread. Possible apps: Managing virus spread in networks, disease spread in human communities. Social network analysis (for social network sites), Innovation spreading (viral marketing), information flow within organisations. Telenor

Results (from D5.5.1)

- Analysis of structure of open-source communities. An understanding of the open source development process has the potential to contribute to improvements in the design and management of this OS process. UPF.
- Analysis of motifs in software graphs. Possible application for software development and maintenance through new software metrics (e.g. prediction of requirement for refactoring). Also intelligent software code searching. UPF
- Cooperative P2P protocols resistant to certain kinds of cheating and selfish behaviour in client nodes. Possible apps: cooperative spam & spyware countermeasures. Cooperative broadcasting, content replication. UniBo

WP5.6 The Structure of Tinkered Landscapes

Goals (start month 16)

General

Comparison of biological networks and engineered designs
Understand evolutionary mechanisms that make natural networks robust and have other differing properties. Produce simulator package.

Specific

Characterize topologies, functional constraints, fitness landscapes of existing networks. Relate knowledge to optimizing evolutionary rules / algorithms.

Partners

UPF, UniBO

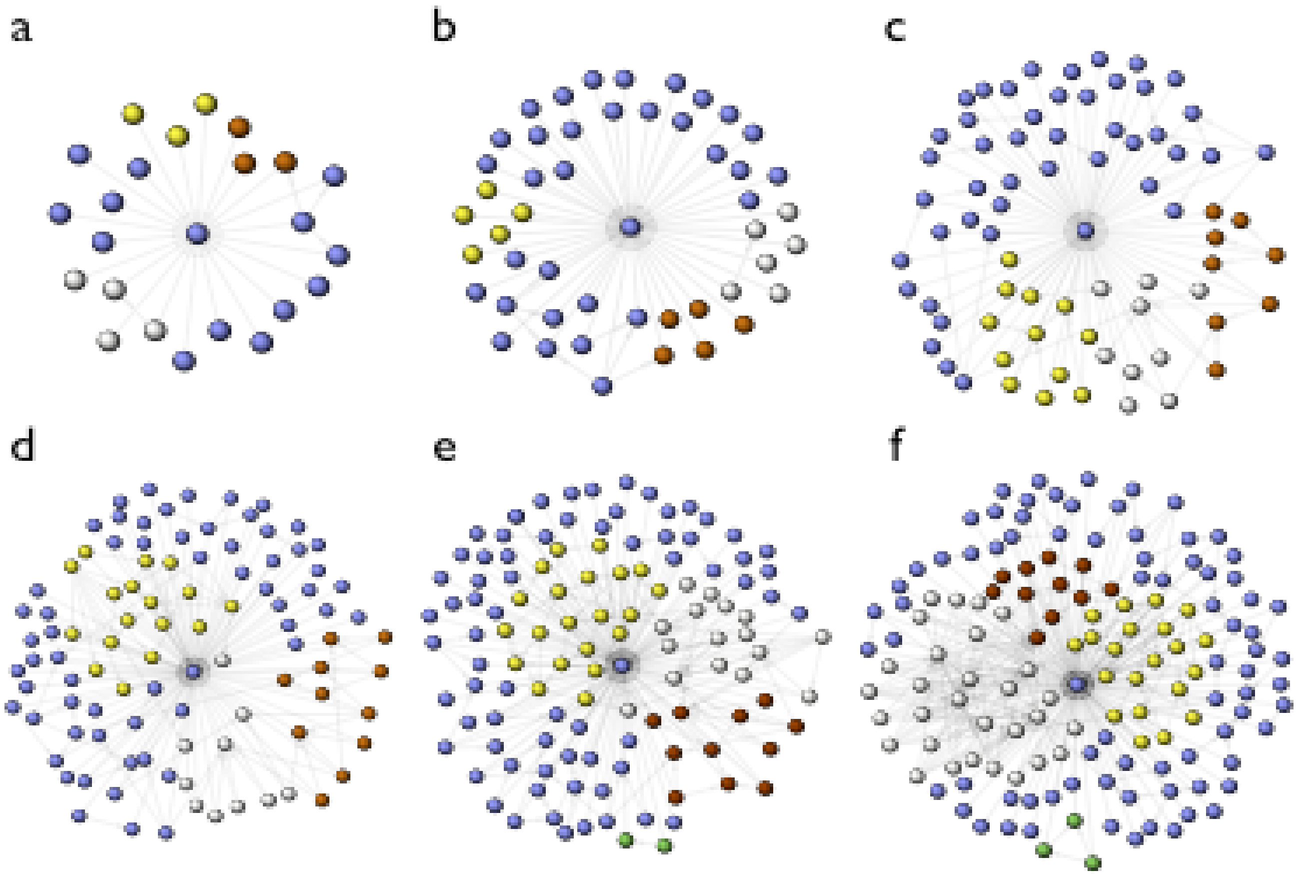
Results (from D5.6.2)

- Study of the Topology and Evolution of Patent Citation Networks
- Similar properties to scientific citation networks
- Evidence of network “modularity”
- Simple preferential attachment rule can reproduce structure
- Publications:
  - Valverde, S., Sole, R. V., Bedau, M., and Packard, N. H., ”Topology and Evolution of Technology Innovation Networks”, *submitted* to Phys. Rev. E.

Patent Citation Networks

- Study of the Topology and Evolution of Patent Citation Networks
- Captures a form of innovation and evolution of technology
- Dataset: US Patent and Trademark Office (http://www.uspto.gov/)
- Patents are grouped into modules - sets of nodes that exchange more links between them than with the rest of nodes
- Careful inspection of patents associated to nodes within a module reveal common functional traits
- The in-degree distribution for the patent citation network follows an extended power-law form
- Extended power-laws have been previously associated with a mixed attachment mechanism
- Mixed attachment mechanisms involve new nodes attaching to target nodes according to their degree and also at random

WP5.6 The Structure of Tinkered Landscapes



From (a) to (f), evolution of a patent subset related to computed tomography. The hub in the center corresponds to the precursor invention by G. Hounsfield (US patent 3778614).

SP5 Dissemination and Cooperation

Cooperation with other SP's

SP4-SP5 Game theory and evolutionary economics models
SP5-SP6 Cooperative distributed information sharing
CCT2, CSS-TW1 Meetings attended / organised

Cooperation with other projects

- BISON As described, extensive cooperation with concluded BISON
- NANIA EPSRC (UK) 5 year project – 2 collaborative meetings made in 2006, with Manchester / Entire NANIA group
- CATNETS On-going collaboration (FET STREP)
- ONCE-CS Complexity Network, ECCS'06 (2 posters, 1 paper)

Other

UniBo prominent in organisation of new SASO conference @ MIT, July 2007

Thank you!