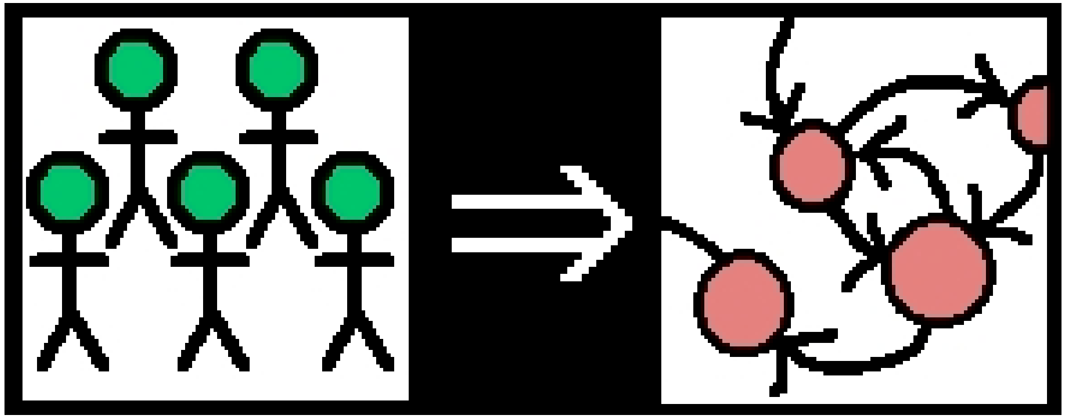
Socially Inspired Computing



Engineering with Social Metaphors

Cluster of Areas in SIC

- Social Simulation
- Evolutionary Computing
- Evolutionary Economics / Game Theory
- Artificial Life
- Artificial Societies

Emphasis

- Understanding
- Scientific / experimental
- General / abstract
- Interpretation of model key
- Computational simulation
- Emergence, Self-organisation
- Evolution, Decentralised, Scaling

Engineering

- Specified functions
- Known goals
- Technical constrains
- Practical implementation issues
- Top down, centralised, poor scaling
- Closed, Secure
- Fixed, non-adaptive

New Trend: Self-* Engineering

- Self-Organising, Self-Managing
- Self-Repairing, Self-Reoganising
- Emergent Function
- Decentralised, Open
- High Scalability
- Light Overheads

Basic Question

- Self-* has draw on biological inspiration
- But many Self-* problems look like sociological problems
- Can Self-* learn from socially inspired work?
- Can SIC learn from Self-* ?

Invited Speakers

- Next:
  - Mark Jelasity (Bologna)
- After Lunch (14:55):
  - Giovana Di Marzo Serugendo (Geneva)

Engineering with Sociological Metaphors: Examples and Prospects

www.davidhales.com

University of Bologna

This work is partially supported by the European Commission under the DELIS project

Background

- Many Self-* engineering issues can be thought of sociological questions:
  - Cooperation in open systems
  - Emergent social structures
  - Scalability, distributed implementation
  - Robustness

Examples - BitTorrent

BitTorrent system:

- P2P file sharing peer software
- Tens of millions of users
- Estimate 35% internet traffic
- Inspired by the tit-for-tat strategy popularised by political scientist Robert Axelrod (80’s) in PD tournaments
- WWI fraternisation over the trenches

Tit-for-Tat Strategy

- Start by cooperating
- Then copy behaviour of opponent in pervious interaction
- Hence, punish bad guys in the future
- Requires repeated interactions

Example - SLAC

SLAC algorithm:

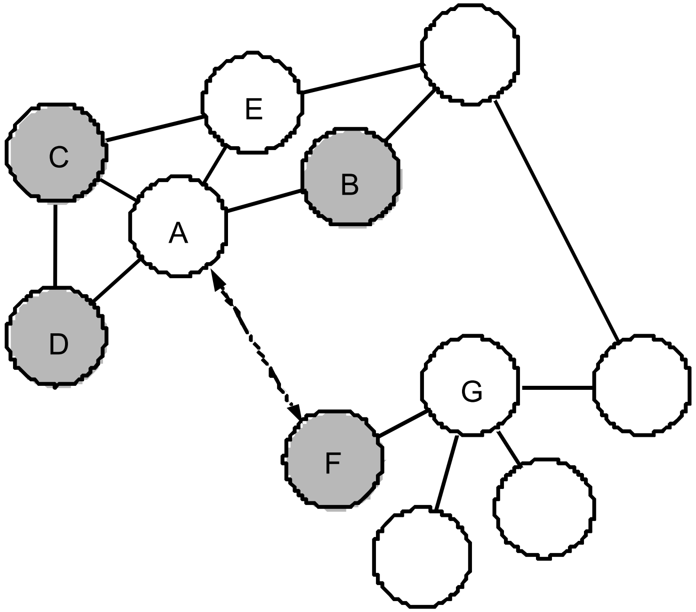
- Applying “tags” within a p2p network
- Translating an “evolutionary algorithm” into a network: *replication and rewiring*
- *Simulation* of file sharing scenario
- Inspired by *tag-based* cooperation models (old school tie effect) Holland/Axelrod/Riolo PD
- Works in one-time interactions

SLAC Algorithm

- Periodically each node:
  - Compares it’s performance (utility) with a randomly chosen other node
  - If other node has higher utility, copy that nodes view and behaviour
  - Mutate (add noise with low probability) to view and behaviour

Copying a more successful node

Before

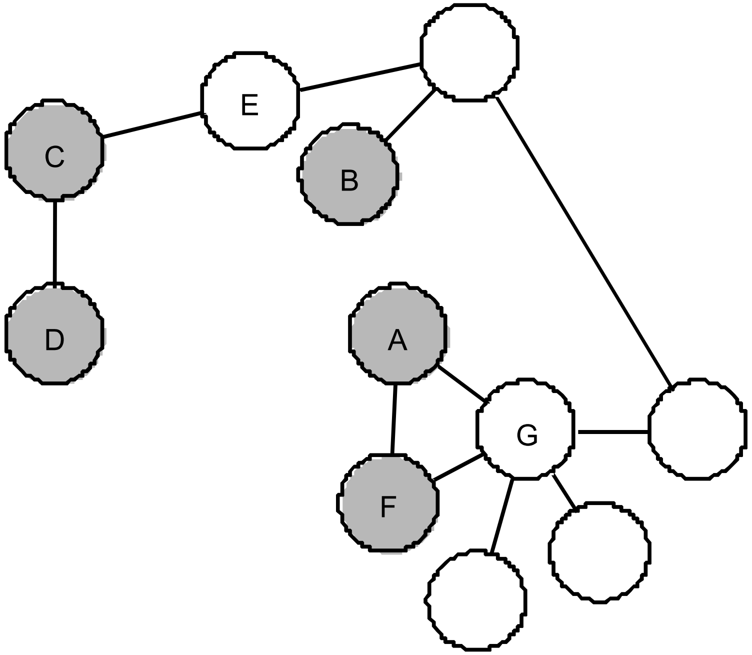


$F_u > A_u$

Where A_u = average utility of node A

After

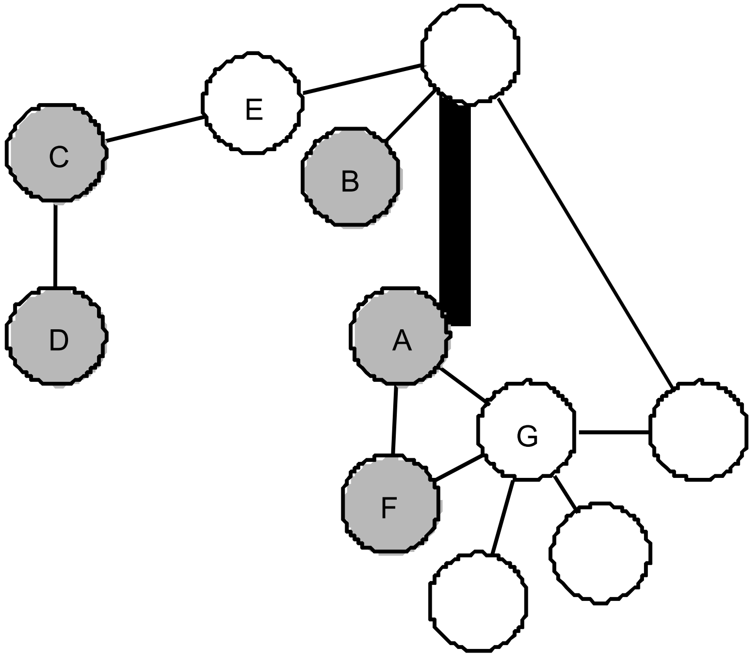
A copies F neighbours & strategy



In this case mutation has not changed anything

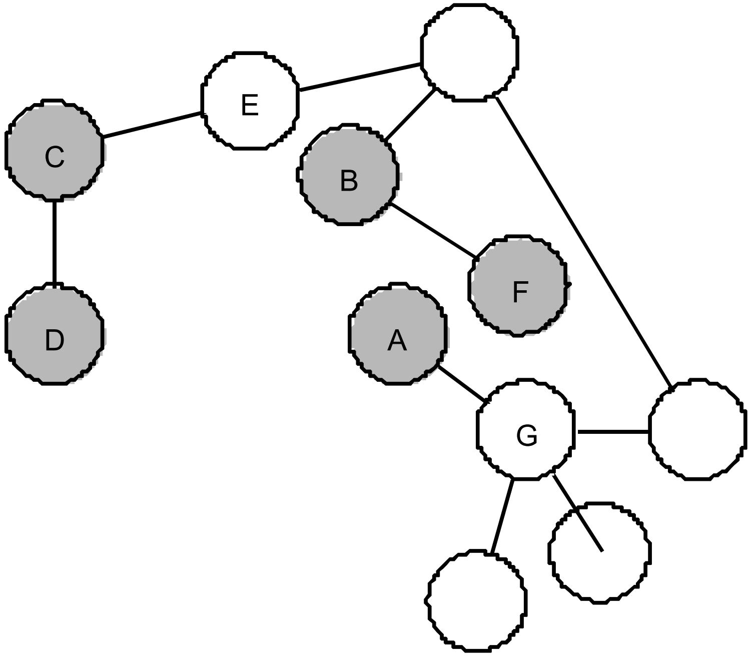
Random movement in the net

Before



Mutation applied to F's neighbourhood

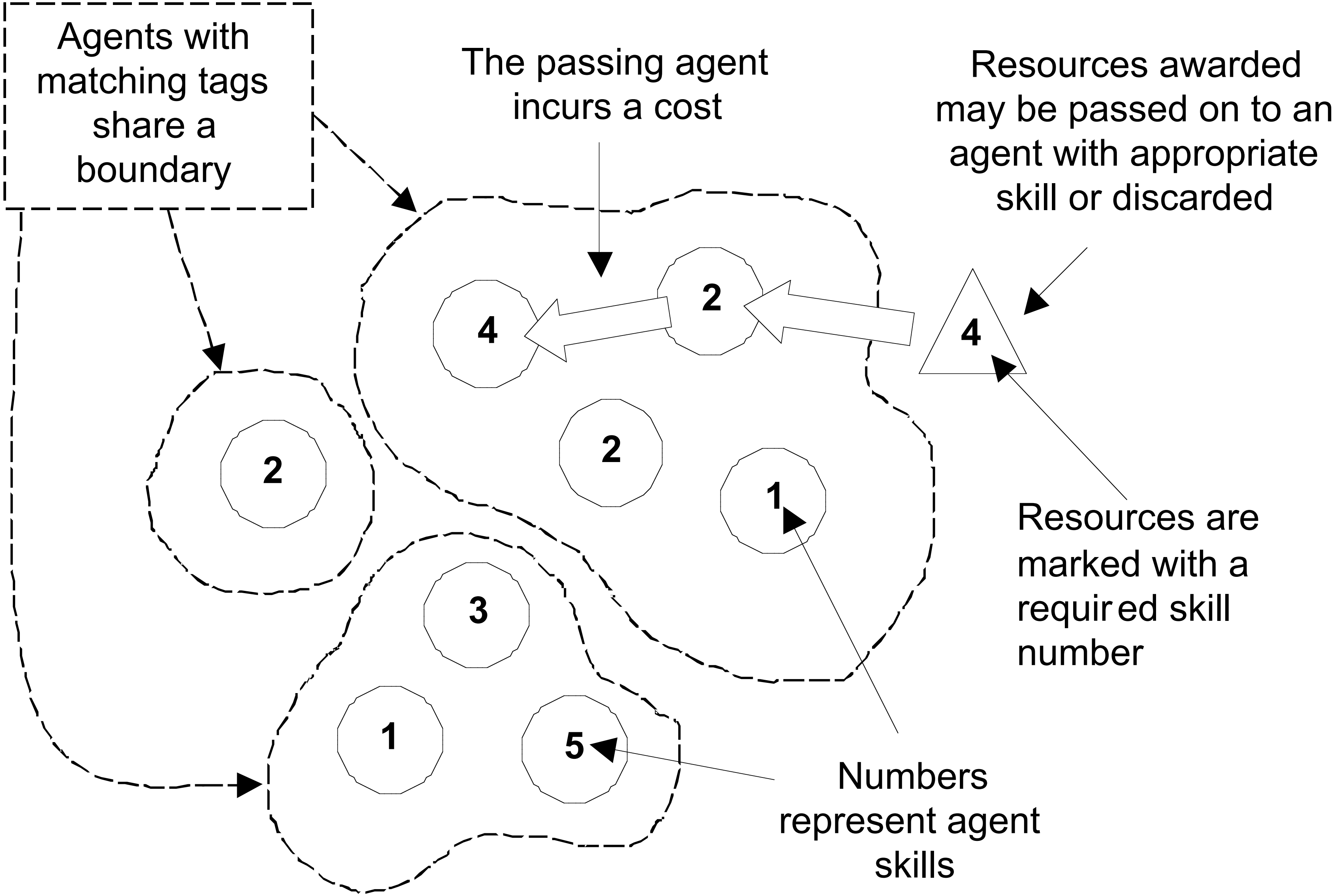
After

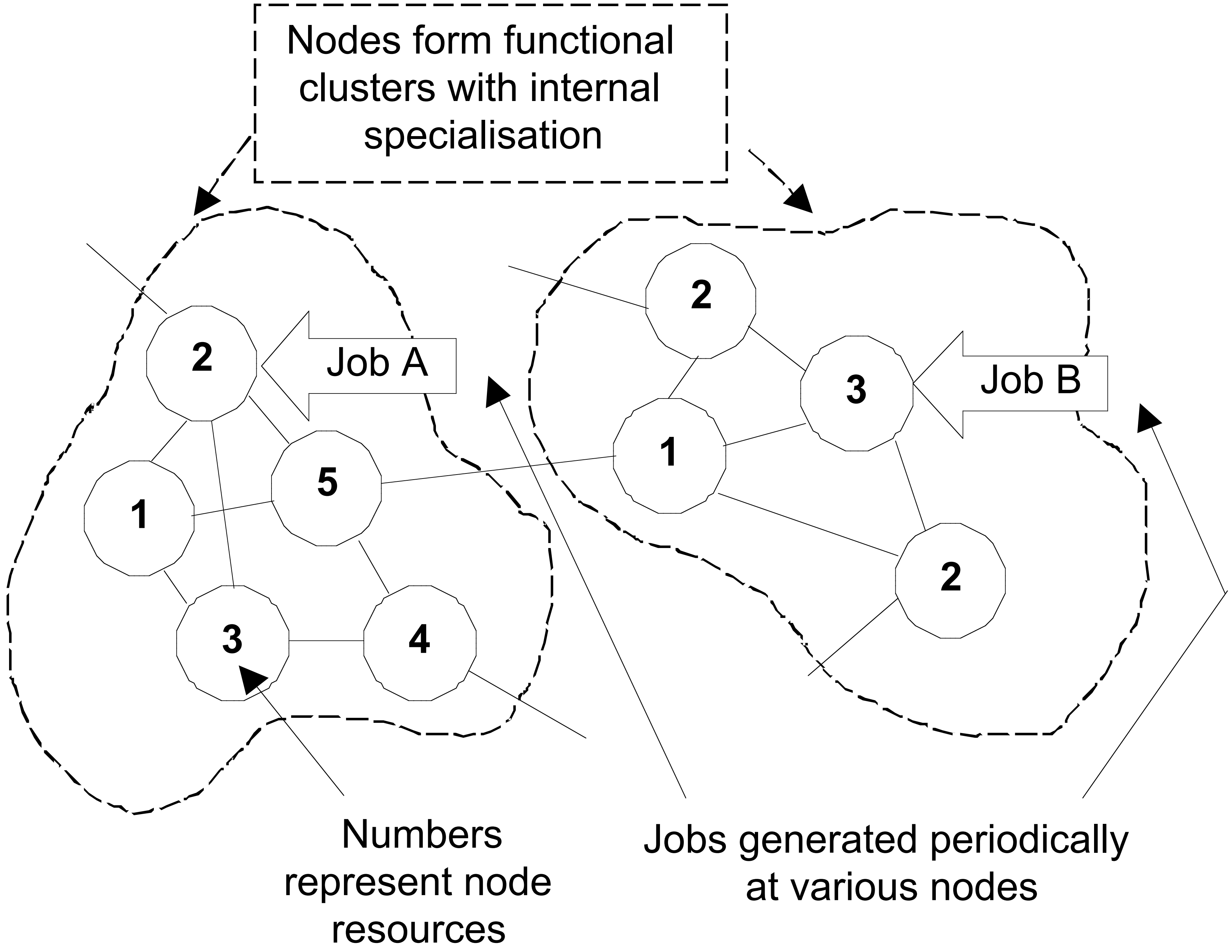


F is wired to a randomly selected node (B)

Prospects - Specialisation

- SLAC works for producing simple cooperation in PD and a file-sharing scenario
- It can also be applied to produce clusters of nodes with internal division of labour
- Previous tag models interpreted as “foraging tribes – harvesting resources”
- Can be translated into “nodes and jobs”


Agents with matching tags share a boundary
The passing agent incurs a cost
Resources awarded may be passed on to an agent with appropriate skill or discarded
4
2
4
2
2
1
3
1
5
Resources are marked with a requir ed skill number
Numbers represent agent skills


Nodes form functional clusters with internal specialisation
2
Job A
1
5
3
4
2
3
Job B
1
2
Numbers represent node resources
Jobs generated periodically at various nodes

Prospects – power in p2p

- Many social simulation work with evolving social networks
- Some demonstrate the emergence of *hierarchy* and *power*
- Both may be useful for many engineering problems in p2p

Engineering with Social Metaphors Discussion

- Is any of this really engineering?
- Are we really making use of social metaphors or is the link tenuous?
- Can general methods be developed to import techniques?
- How are mutation, replication, strategy and fitness concepts translated into deployable systems?