The technical impact of social links in free software development

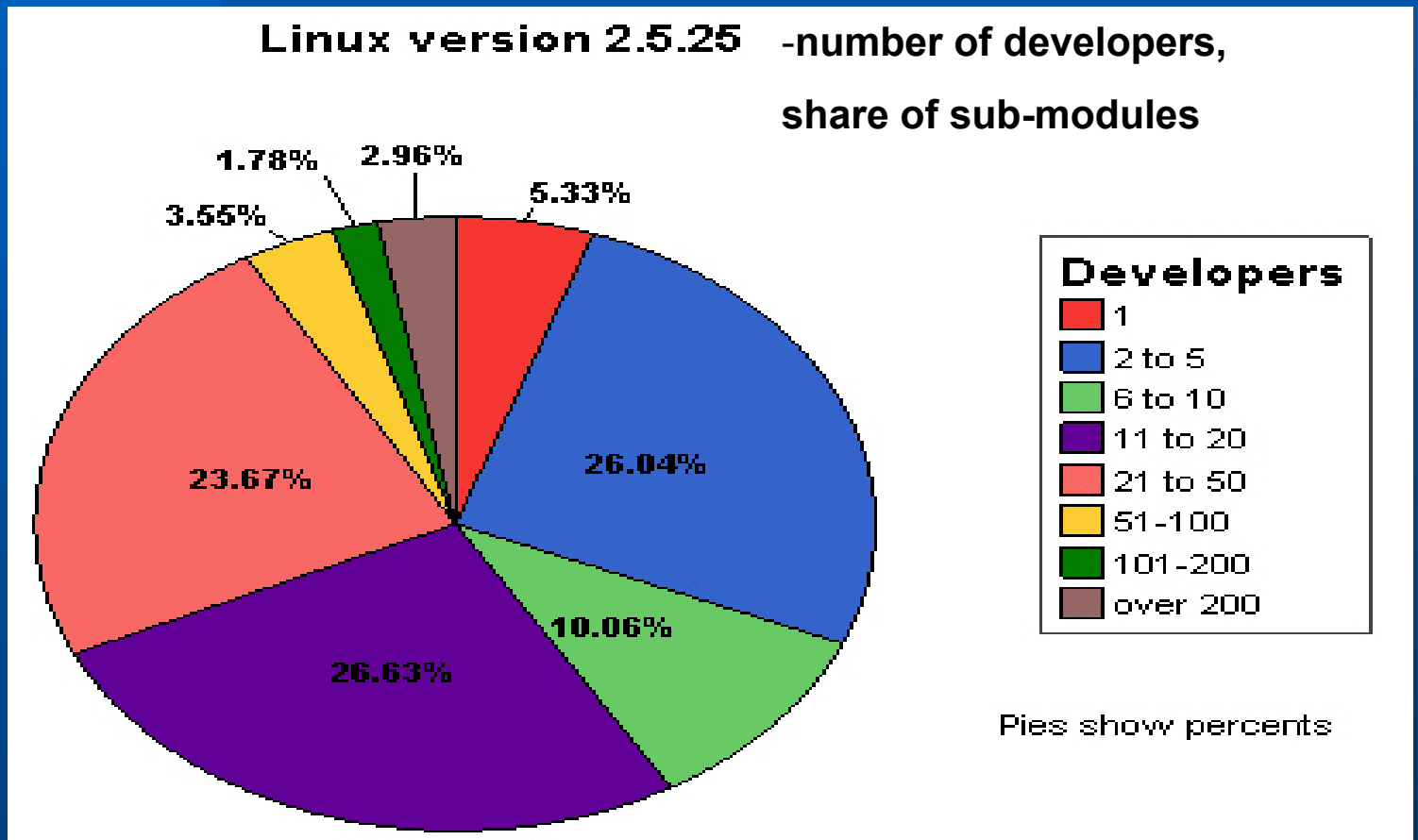
Oxford Workshop on Libre Software, Oxford Internet Institute, June 25, 2004

Rishab Aiyer Ghosh

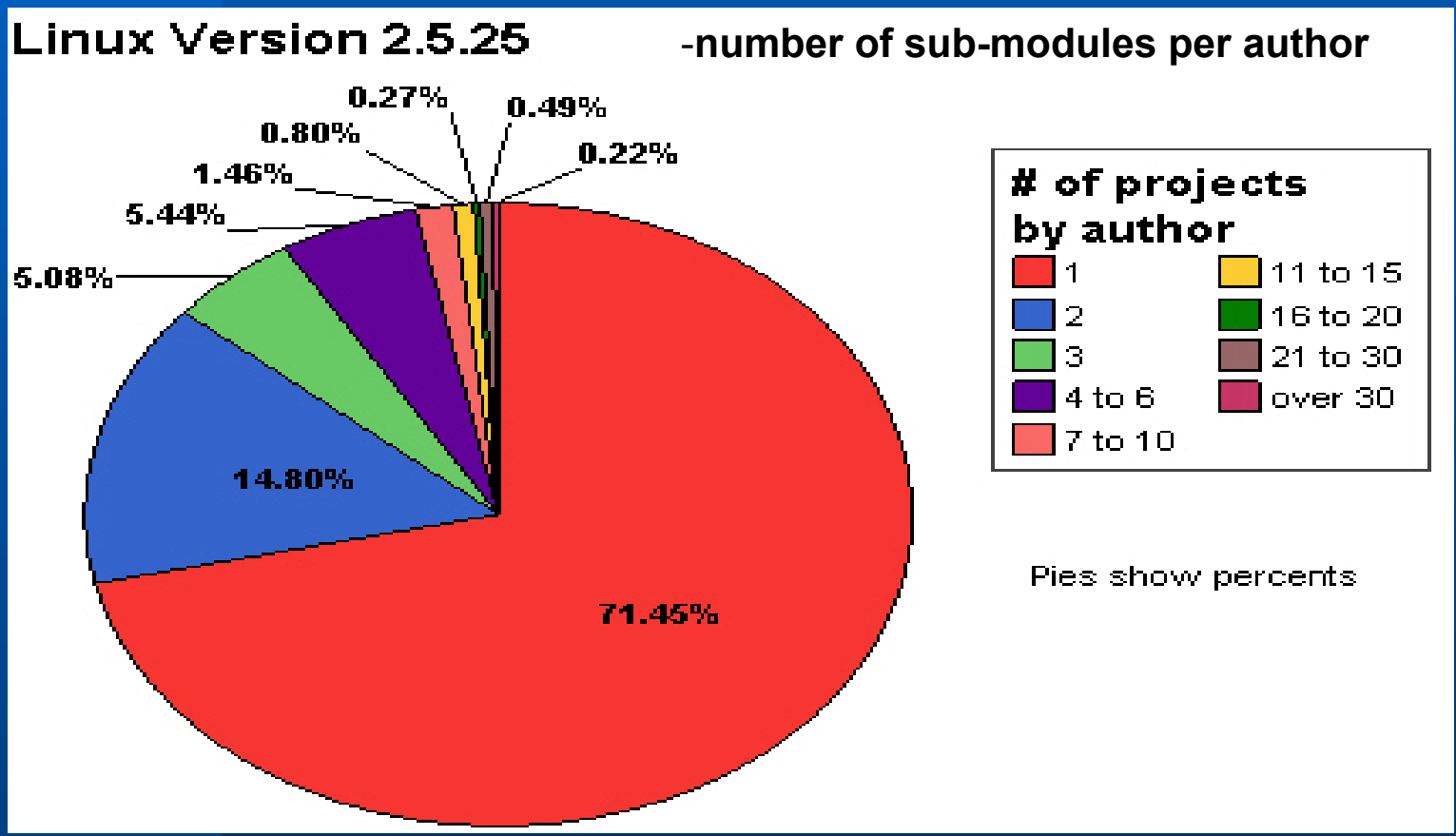
rishab@dxm.org

MERIT/Infonomics, University of Maastricht

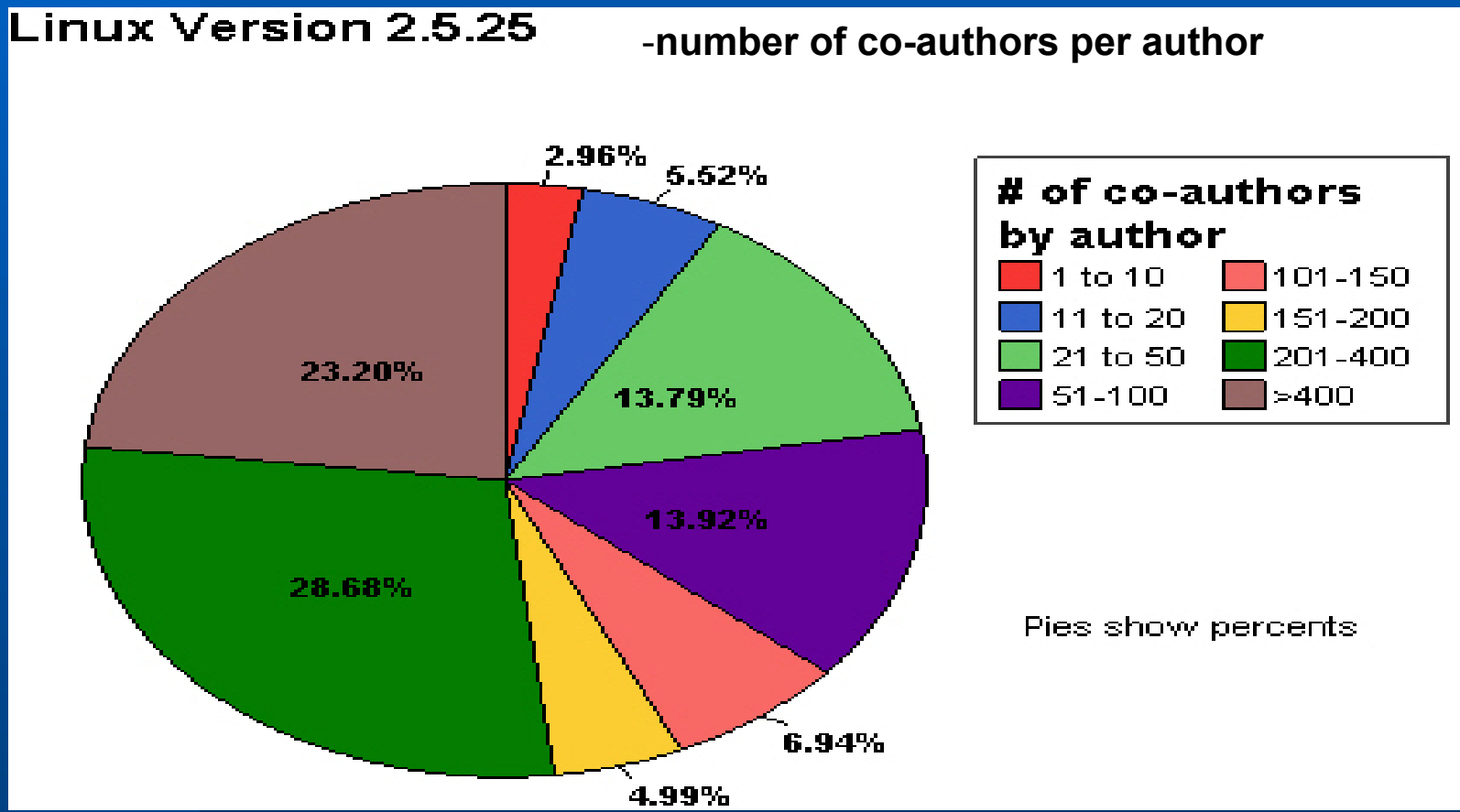
Some data on the Linux kernel



Some data on the Linux kernel



Some data on the Linux kernel

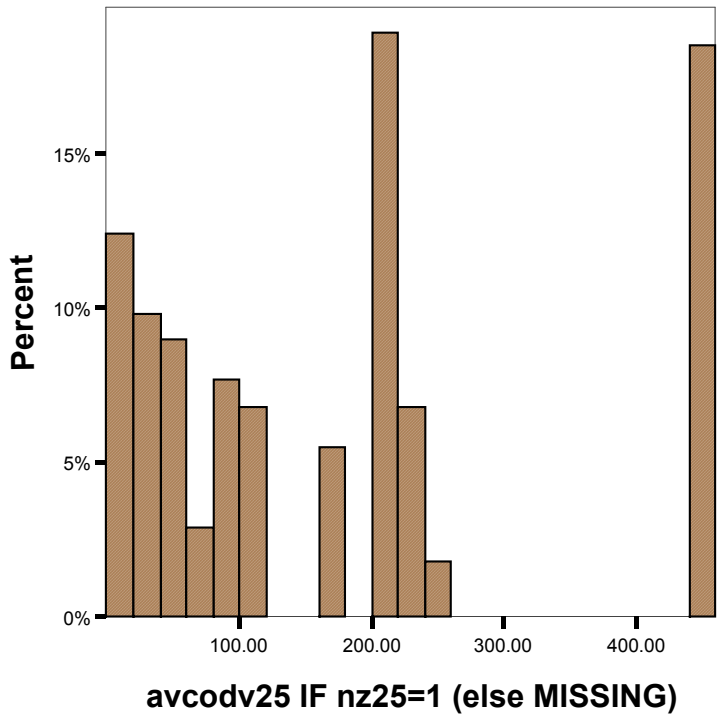


Social choice of modules?

- Most modules have just a few authors
- Most authors have contributed to just one module
- However, these lone contributions are not made to 1-author modules
- New contributors choose modules with many other contributors

Mean developers per module

Mean number of "co-developers" per module for 1-module contributors: over 45% have more than 200 co-developers per module.

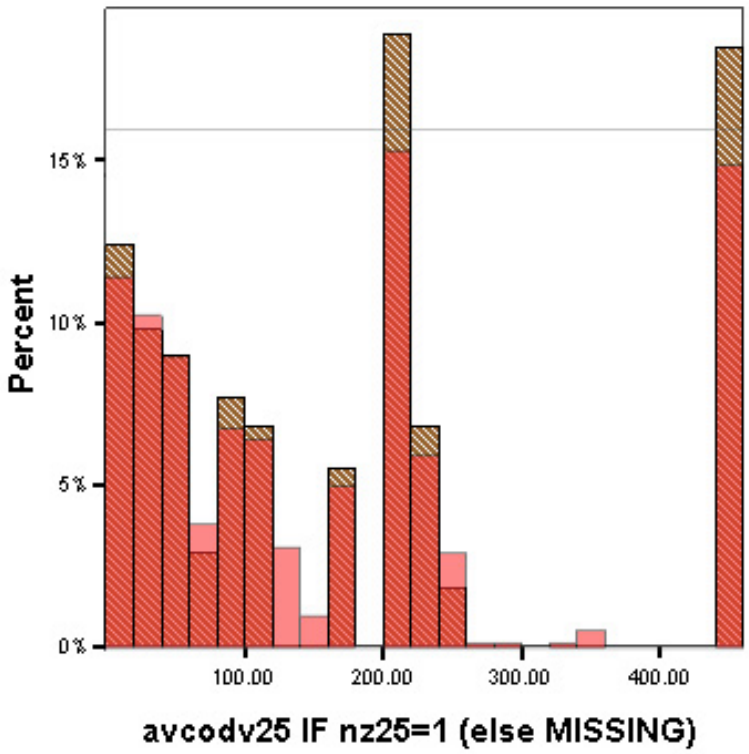


Mean developers per module

Mean number of "co-developers" per module:

- for 1-module contributors: over 45% have more than 200 co-developers per module.

- for 2-module contributors (red overlay), only a slight change. I.e. 2nd modules are not much smaller than 1st modules contributed to.



Mean developers per module

Correlation: number of modules authored by average number of developers per module

		avdev
v1.0	(n=158)	-0.225
v2.0.30	(n=618)	-0.222
v2.5.25	(n=2263)	-0.145

(Pearson 2-tailed: correlations are significant at the 0.01 level)

Developer numbers / module size

- Contributors don't necessarily know how many co-authors they have / will have for a given module
- Code size of module is easily available
- Size is an explicit proxy for "importance"
- Hypothesis: developers are attracted to "important" projects, partly because they have many other developers

Developer numbers / module size

Module size highly correlated to number of authors

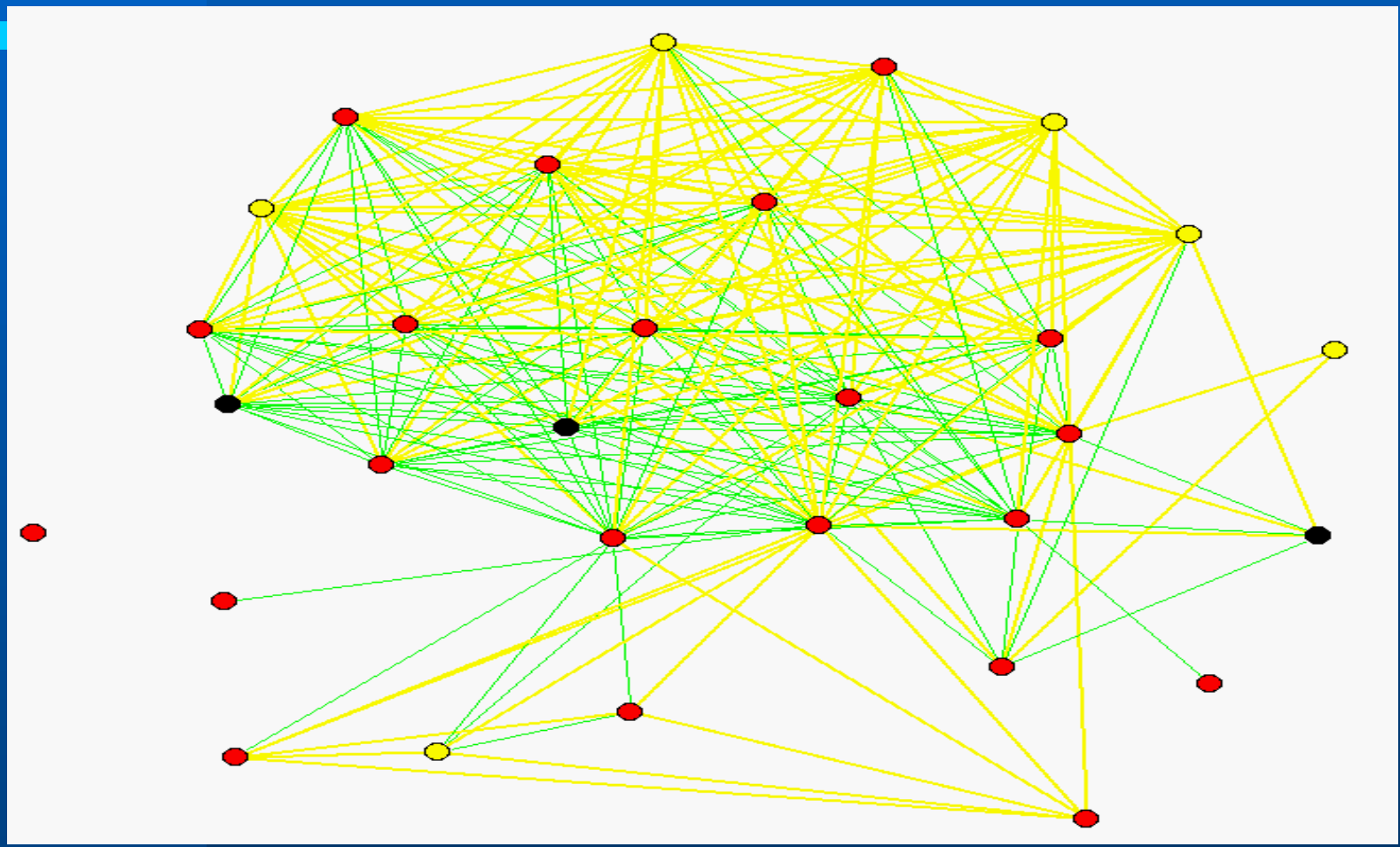
		Correlation
v1.0	(n=30)	0.890
v2.0.30	(n=60)	0.892
v2.5.25	(n=169)	0.894

(Pearson 2-tailed: correlations are significant at the 0.01 level)

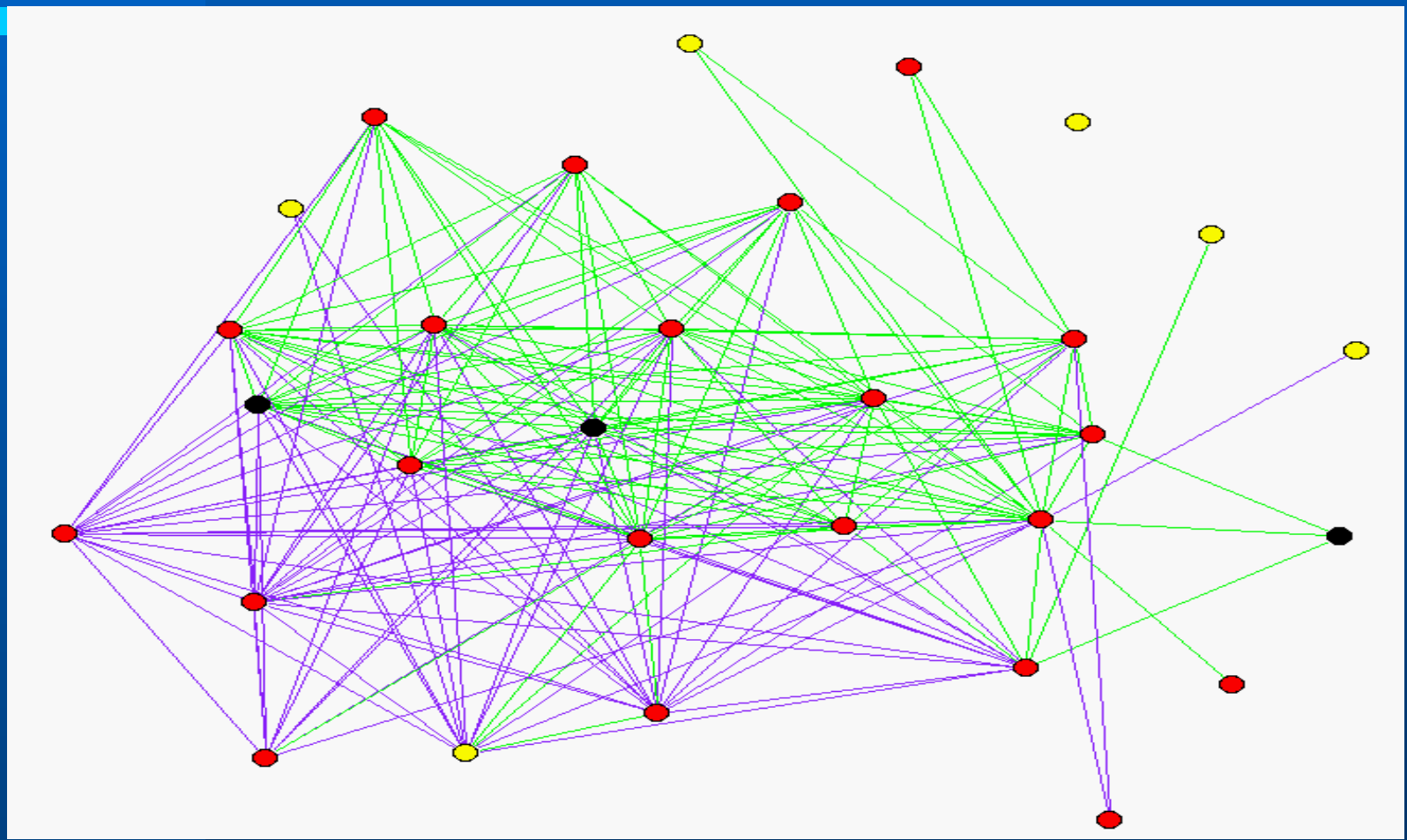
Module association

- Modules can be linked together to form a graph based on at least two criteria
- Authorship: one or more contributors are common to modules
- Code: one module depends on functions defined in another module
- It turns out that the co-incidence of these two attributes is quite high

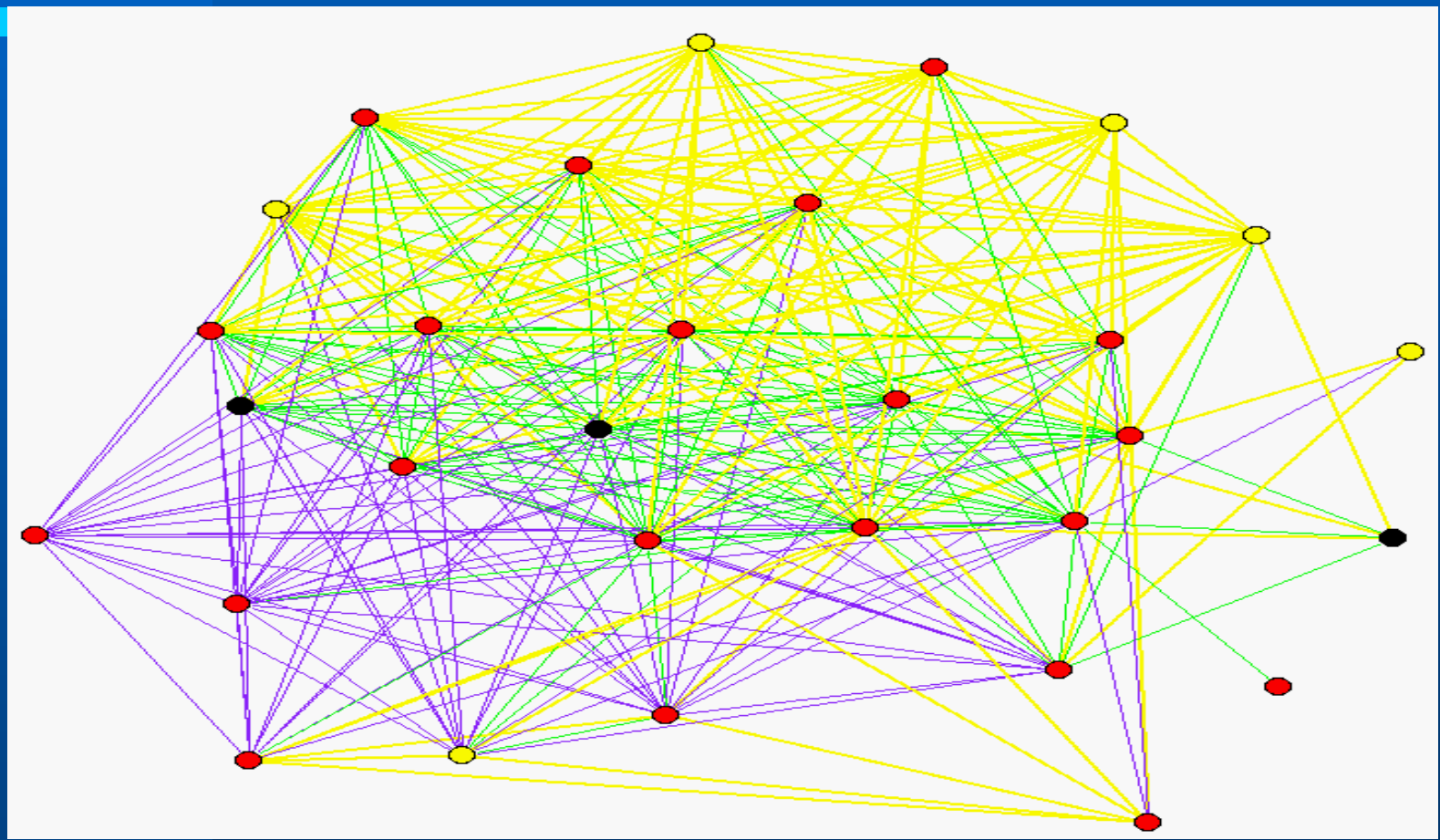
Module association: authors, v1.0



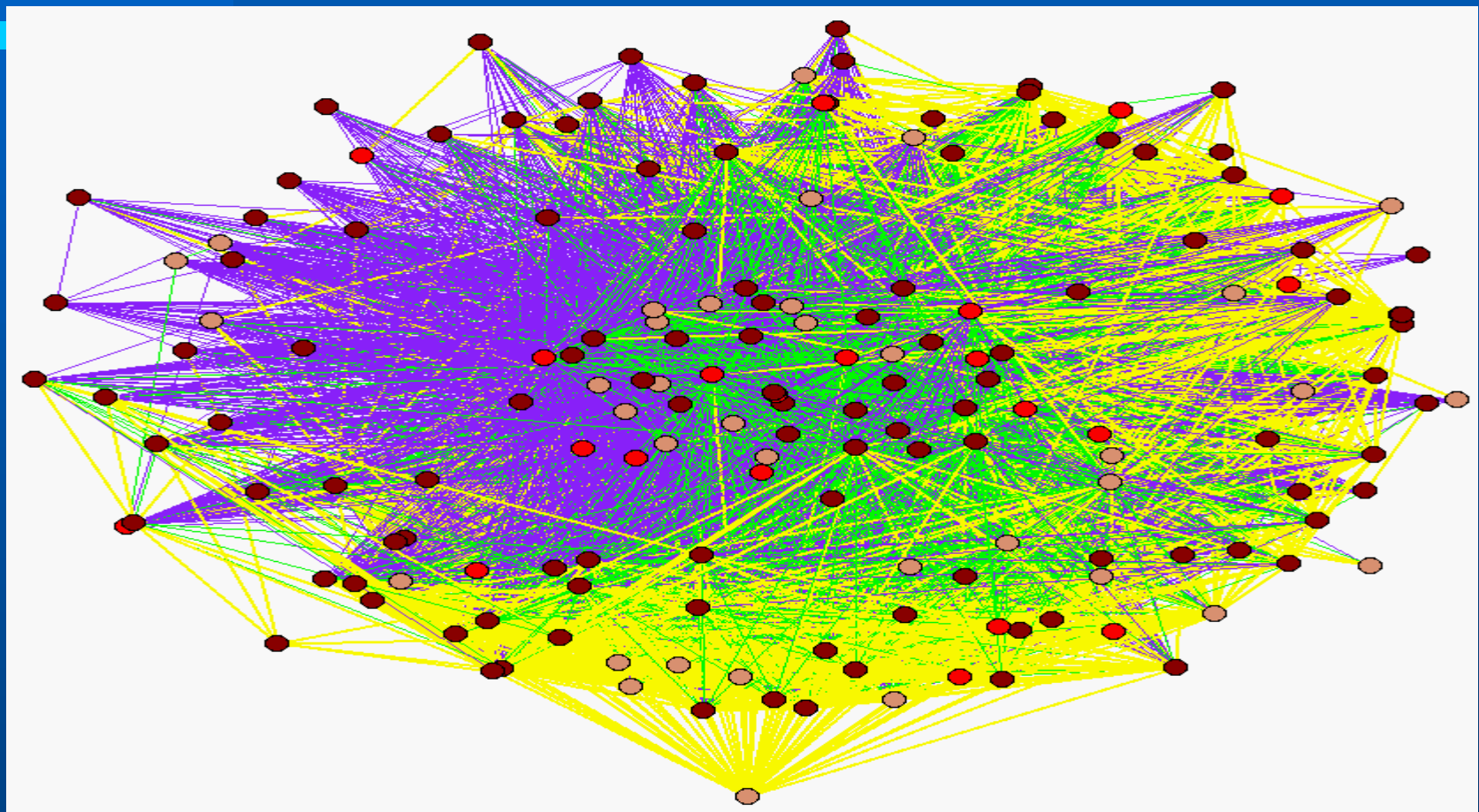
Module association: code, v1.0



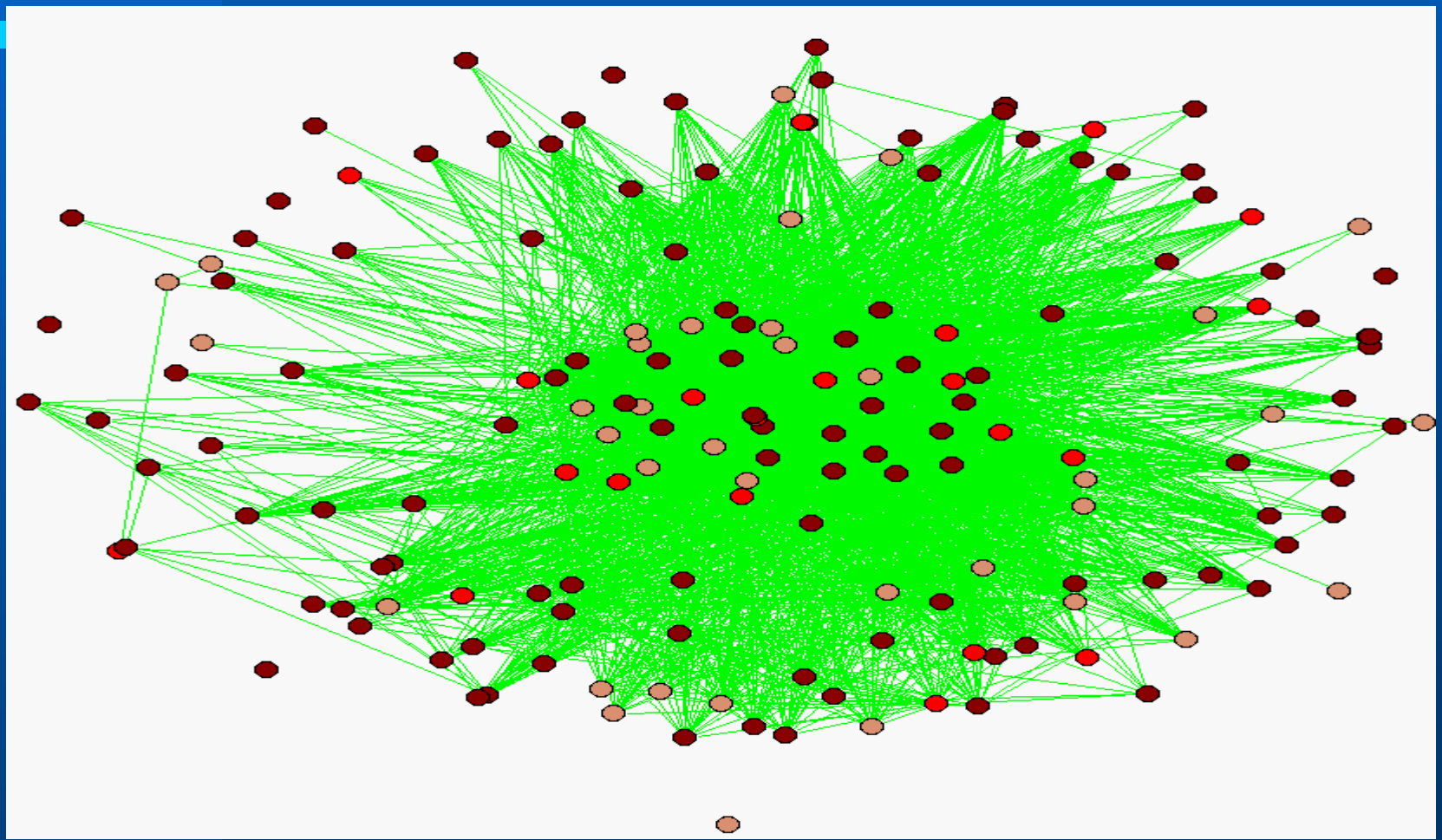
Module association: both, v1.0



Module association: both, v2.5.25



Module association: both, v2.5.25



Exploring the author-code link

- Strong degree of coincidence – between author link and code link
- Significant correlation between strength of author link and presence of code link
- Indications of dynamic impact of author link on future code links (and vice versa)

Exploring the author-code link

Phi 4-point similarity between binary variables for author link & code link:

v1.0	**(n=435)**	**0.122**
v2.0.30	**(n=1770)**	**0.254**
v2.5.25	**(n=14196)**	**0.341**

Exploring the author-code link

Spearman's rho: scalar variables for strength of author link against binary variable for presence of code link:

v1.0	**(n=435)**	**0.130**
v2.0.30	**(n=1770)**	**0.241**
v2.5.25	**(n=14196)**	**0.341**

(strength measured by number of common authors;
also tested with other strength measures)

Exploring the author-code link

- Analytical problems: no good fit model with regression, possibly due to highly skewed data
- Hard to select strength (rather than binary presence) variable for code dependency link
- Time lag between versions too big for dynamic analysis

Future steps

- Similar exercise using finer dynamic granularity (and possibly CVS data, for activity rather than cumulative authorship) may allow better interpretation
- Better data on code dependency (especially dependency strength) may help in identifying relationships

Tentative conclusions

- Significant relationship between social and technical links between modules
- Direction of causality is unclear
- Impact on code development is, however, potentially very high
- "tip of the iceberg" – e.g. code reuse (including dependency on "trivial" rather than "complex" functions) may be much higher with greater social links

Future steps

- Use of clustering algorithms may help identify predictor relationships
- Common authorship may predict code dependency group-wise more than pair-wise

Further information

- **www.flossproject.org**
- **CODD technical papers: orbiten.org/codd/**

  codd.berlios.de