Cell Systems
Commentary

Comparing Technological Development and Biological Evolution from a Network Perspective

Koon-Kiu Yan,[1,8] Daifeng Wang,[2,3,8] Kun Xiong,[4] and Mark Gerstein[4,5,6,7,*]
[1]Department of Computational Biology, St Jude Children's Research Hospital, Memphis, TN 38105, USA
[2]Department of Biostatistics and Medical Informatics, University of Wisconsin - Madison, Madison, WI 53726, USA
[3]Waisman Center, University of Wisconsin - Madison, Madison, WI 53705, USA
[4]Department of Molecular Biophysics and Biochemistry, Yale University, New Haven, CT 06520, USA
[5]Program in Computational Biology and Bioinformatics, Yale University, New Haven, CT 06520, USA
[6]Department of Computer Science, Yale University, New Haven, CT 06520, USA
[7]Department of Statistics and Data Science, Yale University, New Haven, CT 06520, USA
[8]These authors contributed equally
*Correspondence: mark@gersteinlab.org
https://doi.org/10.1016/j.cels.2020.02.004

We compare the "patterns of mutation" in biological and technological networks. Negative selection at central nodes in biological networks has been widely reported; however, we show technological networks have an opposite trend. This suggests a potential contrast: biological evolution involves random tinkering, whereas man-made systems change according to rational planning.

The remarkable resemblance between the development of technology and the evolution of biological systems has fascinated generations of engineers, biologists, historians, and philosophers (Basalla, 1989; Arthur, 2011; Wagner and Rosen, 2014). Indeed, there are many apparent analogies. For instance, both biological and technological systems are adaptive, meaning their evolution is driven by some form of selection. While natural selection shapes the diversity of species, selection takes place in technological evolution in the form of the market, which combines various elements like physical constraints and customer requirements. Apart from such analogies, biological and technological evolutions share deeper commonality. For example, the "bursty" nature of biological evolution, punctuated equilibrium (Gould, 2002), has been reported in the evolution of various technological systems, such as software systems and programming languages (Gorshenev and Pis'mak, 2004; Valverde and Solé, 2015).

Perhaps, the most intriguing question is as follows: what is the reason behind the resemblance, given that random tinkering drives biological evolution while technological evolution involves, to a great extent, the plan of rational human designers? Is it conceivable to develop a unified framework or even discover overarching laws governing these two evolutionary processes? To compare and contrast the two systems, it is worthwhile to look in detail at their underlying connectivity networks. This is because the networks are a common framework that can be used to describe both biological and technological systems. By capturing the interactions between heterogeneous components in the corresponding systems, the underlying structure of the complex networks determines the function of the systems. A decade ago, Uri Alon and collaborators observed several common features in the organization of biological and engineering circuitry, such as modular organization and the existence of recurring elements called network motifs (Alon, 2003). He argued that the common architectures result from common design principles adopted by nature and human design. For instance, certain network motifs make a system tolerant to noise (Alon, 2007); such motifs are, therefore, widely found in biological and engineering networks for the sake of robustness. Despite a biological system and a man-made system presenting two similar solutions to an engineering problem, the routes to these solutions, or mechanisms, remains in question. To answer this question, instead of merely focusing on the convergent trends, it may be useful to re-examine certain differences.

Case Study: Evolutionary Patterns of Protein Interaction Network versus Package Dependency Network

Mapping the evolutionary patterns of components onto the underlying networks may shed light on the construction of similar solutions. As a case study, we focus on a piece of software, the statistical computing language R, a technological system built through the collaborative efforts of many statisticians and programmers. The evolution of R is captured by the so-called package dependency network, which specifies how the proper installation of a package depends on the installation of another package (A → B meaning package A depends on B) (Burns et al., 2019). Such a dependency exists because most programmers tend to reuse existing code rather than developing everything from scratch. While the topology of package networks by itself has interesting biological analogies (Yan et al., 2010; Pang and Maslov, 2013), here, we compare the evolution of package networks with its biological counterpart. A software package evolves through software update. Because most, if not all, updates are not cosmetic but aim to maintain or improve the function of a package, a higher rate of updates suggests that a package's function tend to change frequently. The tendency of a protein's function to change can be measured by dN/dS, i.e., the ratio of non-synonymous substitutions per site to synonymous substitutions per site, where $dN/dS < 1$ means a protein tends to reject mutations that can change its function and $dN/dS > 1$ means natural selection promotes functional changes in protein. While the rate of update is not compatible with dN/dS, the ways the two quantities vary within their

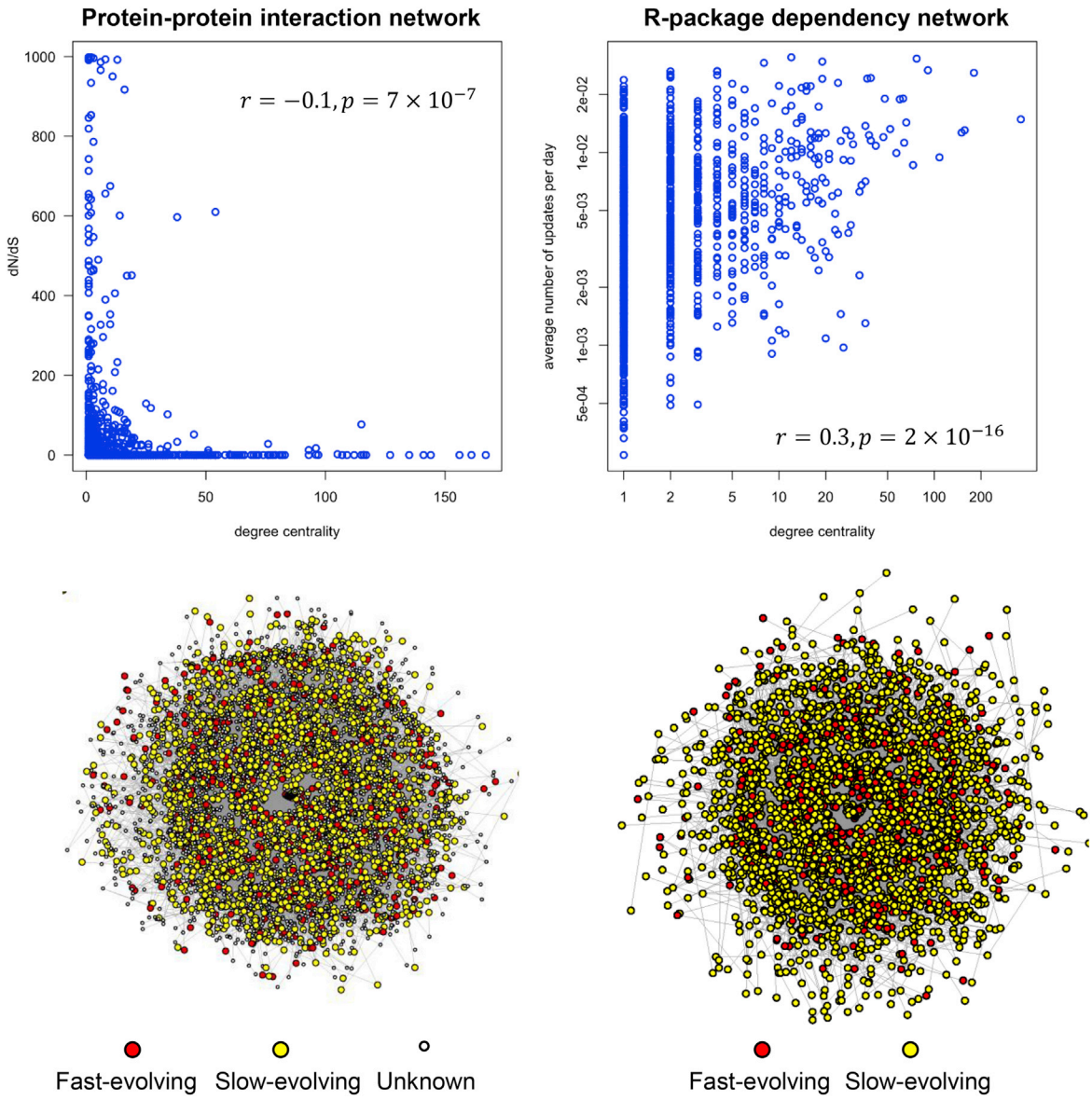

Figure 1. Different Evolutionary Patterns in a Biological Network versus in a Technological Network

The left panel shows the human protein-protein interaction (PPI) network based on Kim et al. (2007) for a direct comparison with the R package dependency network on the right. The bottom left shows the human PPI network (red for dN/dS > 1, yellow for dN/dS < 1, and white for lacking a measurement of dN/dS), and the top left shows the degree centrality (x axis) versus the dN/dS (y axis) of each protein in the network. The bottom right shows the R package dependency network generated by an in-house script (red for update rate > 0.01 and yellow for update rate < 0.002), and the top right shows the degree centrality (x axis) versus the rate of update of each R package (y axis). The rate of update is obtained by parsing the log files in CRAN (https://cran.r-project.org/) with our in-house script. Central nodes in the human PPI network are under strong selective constraints (tend to reject changes in function), whereas central nodes in the R package network change more often. In other words, network centrality and rate of evolution are negatively correlated in biological networks, such as the human PPI network, but positively correlated in technological networks, such as the R package network. Degree centrality of a node is the number of its interacting neighboring nodes in the network. The igraph package (https://igraph.org) was used to construct the R package dependency network and calculate the centralities. We also normalized the update of a package by the size of the package and found a positive correlation between the normalized update and the raw update values. This shows that the update rate of a package is not greatly determined by the size of the package. All of the data and scripts associated with this figure are available from archive2.gersteinlab.org/proj/netessaypkg.

respective networks should reflect how each type of network evolves. As shown in Figure 1, the central nodes in the package dependency network, i.e., the packages that are prerequisite to many others, have higher update rates than peripheral packages. However, in the protein-protein interaction (PPI) network of human, central proteins have smaller dN/dS than peripheral proteins, suggesting that the central proteins tend to reject functional changes more often than peripheral proteins (Figure 1). In other words, node centrality and the tendency of a node's function to change are negatively correlated in the PPI network, but positively correlated in the R package dependency network.

More Insights on Biological Evolution versus Technological Evolution

What are the reasons and implications for this observation? Andreas Wagner formalized the idea of a genotype network in both biological and technological systems (Samal et al., 2010; Raman and Wagner, 2011). The term is used to describe the connections among all genotypes of a system in which two genotypes are connected if one is able to become another via a certain evolutionary step (e.g., a mutation). Because a genotype can be a network of constituents by itself, the genotype network could be pictured as a network of networks. Network evolution is essentially described by a trajectory on the genotype network. In this setting, biological and technological evolution results in different trajectories. Their outcomes are respectively natural and the human-driven ways for innovation. Mediated by random mutations, innovation is possible in biological systems only if they exhibit a certain level of mutational robustness, i.e., the effects of certain mutations are neutral rather than lethal. The outcomes of innovation are also subject to biases in mutation and natural constraints (Smith et al., 1985; Carroll 2008). In PPI networks, the hubs are negatively selected because they have many underlying constraints (Fraser et al., 2002; Hahn and Kern, 2005); in contrast, the peripheral nodes provide room for innovation, and they are in general "hotspots" under positive selection (Kim et al., 2007). Nevertheless, whether the number of interacting partners can fully explain the high selective pressure is debatable due to other underlying cofactors such as protein expression (Bloom and Adami, 2003).

Intuitively, it could also be difficult to update a central node for technological evolution. When a component becomes the prerequisite of others, the component should resist changes because any change can disrupt the function of other components. This kind of "lock-in" phenomenon is quite common in technical systems—e.g., the wide use of QWERTY keyboard—and it is generally referred to generative entrenchment (Schank and Wimsatt, 1986). Nevertheless, what we observed in the package network is different. Instead of a lock-in, the intense use of a package exposes weak spots that hamper performance, forcing software engineers to innovate the package. The situation is analogous to highway networks: the road planner takes measures to ensure construction in Manhattan does not paralyze the city, and one sees comparatively more construction on highly used bottlenecks (e.g., the George Washington Bridge in New York City) compared with out-of-the-way thoroughfares. The absence of lock-in is probably because it is more effective to perform the update, as nowadays disruption could be reduced by using tools such as a version control system. To a certain extent, the existence of central nodes in the package network results from some code being frequently used by many disparate processes; thus, it is cost effective for software engineers to recycle existing code, though it may lower the innovability of the system. It is ironic that extra effort has to be spent for constant updates and alternative innovations. Of course, the possibility of frequently updating a software system owes to the flexible nature of software—technological systems closely tied to particular physical or cultural constraints may present another set of challenges.

We could picture the biological (tinkerer) and technological (designer) evolution as an optimization problem. Despite designers sometimes employing a "trial and error" method, both tinkerer and designer explore the corresponding genotype networks with similar underlying objectives but employ different criteria when balancing constraints. The difference between tinkerer and designer suggests that, as an optimization process, no approach optimizes all objectives (cost effectiveness and mutational robustness in this case), and thus trade-offs are unavoidable in both biological and technological systems. This is essentially the conventional wisdom—there's no free lunch (Lander, 2011).

Conclusion

In short, we have presented a side-by-side comparison on the evolution of a technological network and a biological network. We observed a contrast in the correlation between centrality and the tendency for functional changes that could be explained in terms of the trade-offs between cost effectiveness and mutational robustness. Stepping out of the biological domain thus provides a new perspective on biological evolution. The formal theory of evolution was originated from biology, but since then, the ideas of Darwin have penetrated and transformed many disciplines. Our findings in the R systems is not a singular case; a positive correlation between package centrality and package update rate has been observed in other software systems (Yan et al., 2010; Myers, 2003). It will be instructive to check whether other technical systems show similar behavior, even for engineered biological systems. For example, the idea of tracking updates of R packages can also be further applied to help design engineered systems in such fields as synthetic biology. For instance, a version control system has been developed to track the engineering history of synthetic cells (Luzardo et al., 2019). Finally, contemporary sequencing technology enables biologists to potentially investigate genome evolution in every extant species. Similarly, in the technological domain, via documentation and code that can be readily parsed by computers, engineers have an unprecedented opportunity to study the evolution of designed systems. Given the resources, perhaps studying the evolution of systems in different domains will, in return, benefit both disciplines jointly.

ACKNOWLEDGMENTS

We thank an anonymous reviewer for critical insights and comments on the manuscript. M.G. acknowledges funding from the USA NSF Award #11660648 and the A.L. Williams Professorship.

REFERENCES

Alon, U. (2003). Biological networks: the tinkerer as an engineer. Science *301*, 1866–1867.

Alon, U. (2007). Network motifs: theory and experimental approaches. Nat. Rev. Genet. *8*, 450–461.

Arthur, B. (2011). The Nature of Technology: What it is and how it evolves, Reprint edition (Free Press).

Basalla, G. (1989). The Evolution of Technology (Cambridge University Press).

Bloom, J.D., and Adami, C. (2003). Apparent dependence of protein evolutionary rate on number of interactions is linked to biases in protein-protein interactions data sets. BMC Evol. Biol. *3*, 21.

Burns, B., Lamb, J., Boueri, P., and Qi, J. (2019). Exploring the Structure and Dependencies of an R Package. https://github.com/UptakeOpenSource/pkgnet.

Carroll, S.B. (2008). Evo-devo and an expanding evolutionary synthesis: a genetic theory of morphological evolution. Cell *134*, 25–36.

Fraser, H.B., Hirsh, A.E., Steinmetz, L.M., Scharfe, C., and Feldman, M.W. (2002). Evolutionary rate in the protein interaction network. Science *296*, 750–752.

Gorshenev, A.A., and Pis'mak, Y.M. (2004). Punctuated equilibrium in software evolution. Phys. Rev. E Stat. Nonlin. Soft Matter Phys. *70*, 067103.

Gould, S.J. (2002). The Structure of Evolutionary Theory, First Edition (Belknap Press).

Hahn, M.W., and Kern, A.D. (2005). Comparative genomics of centrality and essentiality in three eukaryotic protein-interaction networks. Mol. Biol. Evol. *22*, 803–806.

Kim, P.M., Korbel, J.O., and Gerstein, M.B. (2007). Positive selection at the protein network periphery: evaluation in terms of structural constraints and cellular context. Proc. Natl. Acad. Sci. USA *104*, 20274–20279.

Lander, A.D. (2011). Pattern, growth, and control. Cell *144*, 955–969.

Luzardo, J.T., Winterhalter, C., Widera, P., Kozyra, J., de Lorenzo, V., and Krasnogor, N. (2019). Linking Engineered Cells to Their Digital Twins: a Version Control System for Strain Engineering. bioRxiv. https://doi.org/10.1101/786111.

Myers, C.R. (2003). Software systems as complex networks: Structure, function, and evolvability of software collaboration graphs. Phys. Rev. E *68*, 046116.

Pang, T.Y., and Maslov, S. (2013). Universal distribution of component frequencies in biological and technological systems. Proc. Natl. Acad. Sci. USA *110*, 6235–6239.

Raman, K., and Wagner, A. (2011). The evolvability of programmable hardware. J. R. Soc. Interface *8*, 269–281.

Samal, A., Matias Rodrigues, J.F., Jost, J., Martin, O.C., and Wagner, A. (2010). Genotype networks in metabolic reaction spaces. BMC Syst. Biol. *4*, 30.

Schank, J.C., and Wimsatt, W.C. (1986). Generative Entrenchment and Evolution. PSA Proc. Bienn. Meet. Philos. Sci. Assoc. *1986*, 33–60.

Smith, J.M., Burian, R., Kauffman, S., Alberch, P., Campbell, J., Goodwin, B., Lande, R., Raup, D., and Wolpert, L. (1985). Developmental Constraints and Evolution: A Perspective from the Mountain Lake Conference on Development and Evolution. The Quarterly Review of Biology *60*, 265–287.

Valverde, S., and Solé, R.V. (2015). Punctuated equilibrium in the large-scale evolution of programming languages. J. R. Soc. Interface *12*, 20150249.

Wagner, A., and Rosen, W. (2014). Spaces of the possible: universal Darwinism and the wall between technological and biological innovation. J. R. Soc. Interface *11*, 20131190.

Yan, K.-K., Fang, G., Bhardwaj, N., Alexander, R.P., and Gerstein, M. (2010). Comparing genomes to computer operating systems in terms of the topology and evolution of their regulatory control networks. Proc. Natl. Acad. Sci. USA *107*, 9186–9191.