









TECHNICAL NOTE

Pyntacle: a parallel computing-enabled framework for large-scale network biology analysis

Luca Parca ¹, Mauro Truglio¹, Tommaso Biagini ¹, Stefano Castellana ¹, Francesco Petrizzelli ^{1,2}, Daniele Capocéfalo ¹, Ferenc Jordán ³, Massimo Carella ⁴ and Tommaso Mazza ^{1,*}

¹IRCCS Casa Sollievo della Sofferenza, Laboratory of Bioinformatics, Viale Cappuccini 1, 71013, San Giovanni Rotondo (FG), Italy; ²Department of Experimental Medicine, Sapienza University of Rome, Piazzale Aldo Moro 5, 00185, Rome, Italy; ³Balaton Limnological Institute, Centre for Ecological Research Klebelsberg Kuno 3, 8237 Tihany, Hungary and ⁴IRCCS Casa Sollievo della Sofferenza, Laboratory of Medical Genetics, Viale Padre Pio 7d, 71013, San Giovanni Rotondo (FG), Italy

*Correspondence address. Tommaso Mazza, IRCCS Casa Sollievo della Sofferenza, Laboratory of Bioinformatics, San Giovanni Rotondo (FG), Italy. E-mail: t.mazza@css-mendel.it  <http://orcid.org/0000-0003-0434-8533>

Abstract

Background: Some natural systems are big in size, complex, and often characterized by convoluted mechanisms of interaction, such as epistasis, pleiotropy, and trophism, which cannot be immediately ascribed to individual natural events or biological entities but that are often derived from group effects. However, the determination of important groups of entities, such as genes or proteins, in complex systems is considered a computationally hard task.

Results: We present Pyntacle, a high-performance framework designed to exploit parallel computing and graph theory to efficiently identify critical groups in big networks and in scenarios that cannot be tackled with traditional network analysis approaches.

Conclusions: We showcase potential applications of Pyntacle with transcriptomics and structural biology data, thereby highlighting the outstanding improvement in terms of computational resources over existing tools.

Background

Interactive systems are commonly represented as graphs (or networks), which are mathematical representations of “elements” (nodes) and their relationships (edges). The semantics of relationships is specific for each graph and completely defines its expressiveness. Protein interaction networks, for example, represent physical interactions as edges and proteins as nodes; metabolic networks wire metabolites whenever these participate in the same biochemical reactions; regulatory networks are directed graphs, where the directionality of relationships matters. Thus, a link exists between 2 molecules if there is evidence either of regulatory activity by a transcription factor onto a gene or of post-translational modifications. These, together with sev-

eral other kinds of networks, such as RNA, signaling, neuronal, trophic, and co-expression networks, are the concrete signs of an exceptional growth of molecular interaction data and, hence, of an intense research activity in the field of network medicine [1].

Network medicine is a relatively new discipline that exploits graph theory to identify key molecules in the human disease [2], together with their hidden molecular relationships. The general aim is that of reverse-engineering the mechanisms of pathogenesis of complex disorders and traits, whereby the etiology is notoriously convoluted. The disease is, in fact, a network where diseases are nodes and links represent relationships between the disease-associated cellular components. De-

Received: 24 March 2020; Revised: 10 July 2020

© The Author(s) 2020. Published by Oxford University Press GigaScience. This is an Open Access article distributed under the terms of the Creative Commons Attribution License (<http://creativecommons.org/licenses/by/4.0/>), which permits unrestricted reuse, distribution, and reproduction in any medium, provided the original work is properly cited.

termining such links would help identify the molecular relationships between phenotypes and the reasons for certain comorbidities and would positively affect diagnosis, treatment, and drug multi-purposing.

Certain kinds of biological networks share the feature of having a few relatively highly connected nodes, often called “hubs,” suggesting that the molecules represented by hubs should play special biological roles. The first hypothesis of network medicine is that most known disease genes, which are non-essential, lie in the periphery of these networks and are far from hubs. On the contrary, at least in human cells, hub molecules are encoded by essential genes [3]. A database, Database of Essential Genes (DEG), exists that reports essential genes for some bacteria, archaea, and eukaryotes [4]. An interesting speculation is that, because of their many links, hubs are reasonably associated with disease genes [5–7], which in turn, by virtue of the local hypothesis of network medicine, exhibit increased tendency to interact with each other, all being involved in the same disease. Thus, molecular networks are not random but tightly organized on the basis of specific principles, according to which the effect of a central gene, which is eventually aberrant, reverberates on the gene products of neighboring genes in its network. Hence, the expression of a disease phenotype rarely results from an individual aberrant gene, rather from the harmonized effects of groups of related genes. This holds true also for other types of networks, ranging from ecological to evolutionary and chemical networks.

Graph theory draws upon various tools to identify the most central elements, i.e., the key molecules, in a network. Here, the concept of centrality is synonymous with importance, even if the term has been declined differently in the literature. A topologically important node may be a hub, a “bottleneck,” namely, a node that lies in many pathways, or one that is “close” to most other nodes. Despite the above description, local (i.e., regarding nodes or edges) and global (i.e., regarding the entire network) properties of networks are unlikely to completely explain the functioning of complex systems because they either fail to take into account or underestimate the effects that groups of important nodes may jointly exert on these systems. Node 2 in Fig. 1 has 7 ties, and it is the highest-degree node in this example network. It is connected with 7 unimportant nodes because these exhibit low degree values. Node 9 is the second-most connected node with only 1 fewer edges than Node 2, but 2 of its neighbors, i.e., Nodes 16 and 21, are the third- and fourth-ranked nodes by degree, with 5 and 4 ties, respectively. Thus, although Node 2 is top ranked by degree, it may not be the most functionally central node. Whether this assertion is true strictly depends on the purposes for which a network is being studied.

More interestingly, the “network parsimony” principle of network medicine, according to which causal molecular pathways often coincide with the shortest molecular paths between known disease-associated components, implies that it is fundamental to find the nodes that lie within the highest number of pathways in networks because these are more likely to be functionally critical [1]. The betweenness centrality index [8] is the most suitable for this task. Node 21 in Fig. 1 is the top-ranked node by betweenness. This was expected because it lies in the exact middle of the network, which, in turn, exhibits a quasi-tree topological node organization. But even if Node 21 belongs to almost all shortest paths of the network, it ranks only fourth by degree because it is not individually much connected. Node 9 is the second node by betweenness, with a score very close to that of Node 21, but, on the contrary, it ranks second by degree (Supplementary Data S1). Whether the most important node is 2, 21, or 9 depends on the aims and context of the study.

Table 1: Group centrality metrics calculated for the example network.

Group	Degree	Betweenness	Closeness*
{2, 21}	0.5	0.39	0.58
{2, 9}	0.59	0.43	0.69
{21, 9}	0.36	0.35	0.38
{2, 9, 21}	0.71	0.45	0.75

Higher scores indicate higher centrality. * The “minimum” method was used to measure the distance from the group to an outside node.

Table 2: KPP-Neg and KPP-Pos metrics calculated for the example network

Group	DF (0.66)	m-reach*	DR
{2, 21}	0.87	79.2%	0.65
{2, 9}	0.91	95.8%	0.72
{21, 9}	0.84	62.5%	0.53
{2, 9, 21}	0.93	95.8%	0.74

DF (Neg) achieves its maximum value of 1.0 when the graph consists entirely of isolated nodes. m-reach (Pos) is a count of the number of unique nodes reached by any member of the group in m links or fewer. DR (Pos) achieves a maximum value of 1 when every non-group node is adjacent to ≥ 1 member of the group. * The m parameter of the algorithm was set to 2. The percentage of nodes reached by the group, including the group nodes, is reported.

Whenever >1 node exhibits similar topological scores, as in this case, or when a co-responsibility for a phenotype is suspected, studying groups and their centrality might be a reasonable option. In 1999, Everett and Borgatti expanded the definition of degree, betweenness, and closeness to groups of nodes [9]. Calculating these indices for the following groups: {2, 21}, {9, 21}, {2, 9}, the latter achieved the highest scores. Moreover, considering the group made by all 3 nodes, only degree and closeness increased significantly in respect to {2, 9} (cf. Table 1).

In 2006, Borgatti introduced 2 other classes of metrics for groups that were meant to assess the ability of groups either to disrupt a network, when removed, or to efficiently spread information through a network. These were defined as “Key-Player Problem/Negative” (KPP-Neg) and “Key-Player Problem/Positive” (KPP-Pos), respectively [10]. Note that similar concepts were also covered in other research fields and scientific contexts [11, 12], where specific search strategies [13, 14] were implemented. KPP-Neg and KPP-Pos were calculated for the same groups and reported in Table 2. It is interesting to notice that {2, 9} is still the most important group in terms of disruption potential and connectivity. Their scores were slightly lower than those of group {2, 9, 11}, meaning that even here Node 11 does not contribute significantly to the centrality of {2, 9}.

What remains to be verified is whether any other group exists that exhibits similar or higher centrality values. Considering the small network size, the option of running a “brute-force” algorithm to search the absolute best group(s) among all possible ones is computationally feasible, in place of a “greedy optimization” search, as suggested by Borgatti [10]. In this case, the best group of size 2 for all metrics is still {2, 9}, whereas {5, 9} reaches 100% of non-group nodes and is ranked first by m-reach. However, because none of the centrality scores of {2, 9} equaled their maximum possible values, we again applied the brute-force search to groups of increasing sizes, 3–6. We thus found that degree and closeness reached their absolute maximum scores, i.e., 1, equally with 2 groups {2, 9, 11, 16, 21}, {2, 9, 10, 16, 21} of size 5, meaning that nodes 10 and 11 are in-

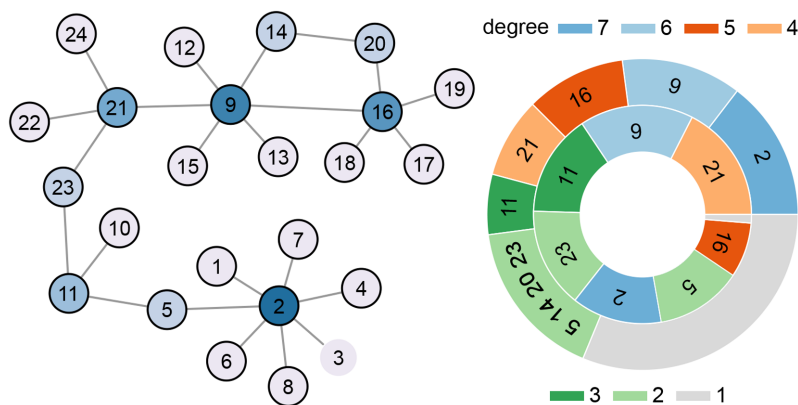


Figure 1: Left: Example network. The darker the blue color, the higher the degree of nodes. Right: Pie chart of the most central nodes. The outer circle reports the highest-degree nodes (counterclockwise, blue through gray). The inner circle represents the highest betweenness nodes, from light orange to gray, counterclockwise. Node names are reported within circle sectors. Sector width is proportional to the degree (outer) and betweenness (inner) values of nodes. Gray sectors contain unimportant nodes, i.e., nodes with unitary degree and negligible betweenness values.

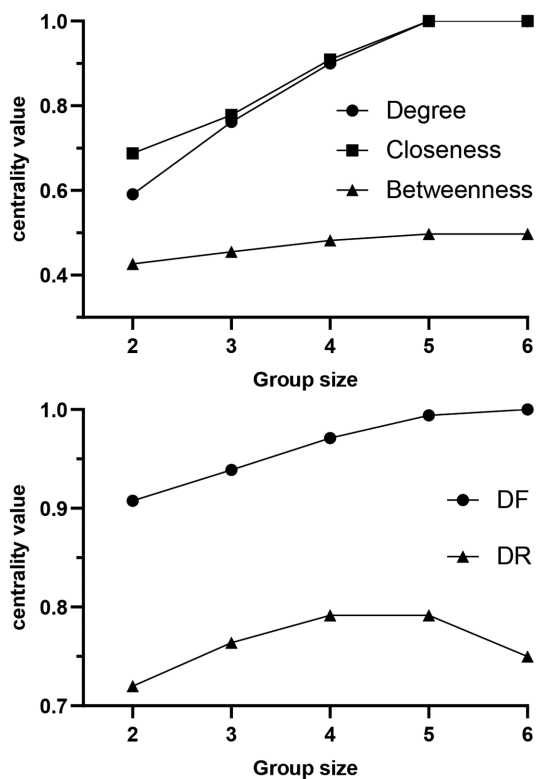


Figure 2: Brute-force search algorithm applied to all groups of sizes 2–6. For any size, the maximum scores obtained for (top) group degree, group closeness, and group betweenness and (bottom) DF and DR are plotted.

terchangeable and equally important; betweenness obtained its maximum score (0.497) with the group {2, 9, 11, 16, 21} (Supplementary Data S2). The best group by DF is {2, 9, 11, 14, 16, 21}, which achieves the score of 1. The groups {2, 9, 10, 16, 21} and {2, 9, 11, 16, 21} equally obtained the best DR score (0.792). It is worth noticing that DR and betweenness do not reach their absolute maximum scores, which however are plausibly the highest possible for this network, because groups of bigger sizes exhibit lower scores (Fig. 2). It is also interesting to notice that Nodes 2, 9, and 21 are included in all groups described above,

thereby highlighting their central roles in the network (Supplementary Data S3).

Computing the nestedness, which consists in verifying whether sets of nodes recur in groups of increasing sizes, could confirm the importance of Nodes 2, 9, and 21. Hence, if larger sets contain smaller sets, higher values of nestedness may be a proxy for identifying upstream/master regulators through the key nodes of the smallest groups. One way to calculate the nestedness of the example network is by the Nrow metrics [15, 16]. Nrow is defined as the average percentage of nodes from smaller sets that are contained in larger sets, taking all possible pairs of sets. Thus, after computing all the best sets of increasing sizes, from 2 to 5, for each group centrality metric but m-reach, Nodes 2 and 9, and not 21, resulted in being nested in all sets, regardless of their size (Fig. 3 and Supplementary Data S2). The same evidence emerged with the key-player metrics (Supplementary Data S3). The nestedness scores were generally quite high, meaning that nodes are not interchangeable among groups, i.e., there are few equally important nodes. The group {2, 9} is definitely important from a topological point of view, and its discovery would not have been immediately hypothesized without this investigation because Nodes 2 and 9 are 5 links apart.

This “practical” introduction aims at introducing the theory underlying Pyntacle. A toy model was used to describe the main features, outline a possible analytical pathway, and highlight how Pyntacle may help extract valuable information from real-world networks. The rest of the article thus presents (i) the software and its main components; (ii) its design and implementation; (iii) details of how to use it; (iv) benchmarks, assessed on real and simulated networks of increasing sizes, in comparison with a similar software package; and (v) 2 real-world case studies.

Pyntacle

Pyntacle is an open-source network analysis framework that was originally designed to tackle the Borgatti Key-Player Problem [10] efficiently through the identification of maximally reachable or disruptive groups of nodes. Contrary to similar software packages that either analyze networks with standard global and local topological metrics [17, 18] or provide the users with limited tools to detect important groups of nodes [19], Pyntacle adopts optimized heuristic algorithms and parallel computing strate-

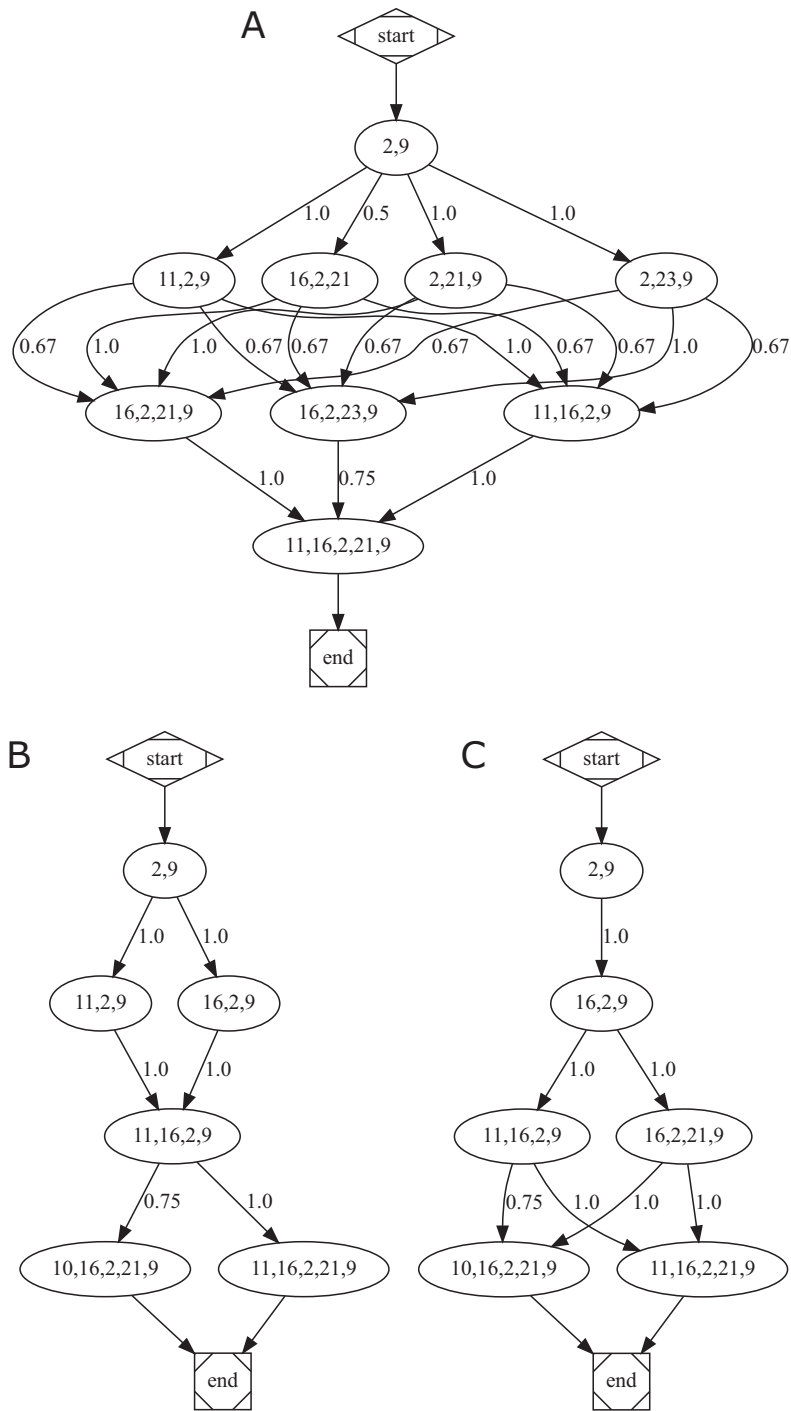


Figure 3: Nestedness graphs for group (A) betweenness, (B) closeness, and (C) degree centrality metrics. Nodes represent groups with top centrality values in respect to all other possible groups of nodes with same sizes. Edges connect groups when the bigger group contains ≥ 1 element of the smaller one. Edges are labeled with the overlap ratio between the elements of the connected groups.

gies to make the task of identifying key-player nodes feasible. It has the following attributes: (i) is available for Windows, Mac, and Linux OS; (ii) is available as both command line tool and API, with an easy and user-friendly interface for both input commands and results visualization; and (iii) allows the management of real-world graphs in a computationally efficient way.

Pyntacle is implemented in modules, each designed to analyze a particular aspect of a network. These can calculate global and local topological metrics (metrics module), the importance of groups of nodes (groupcentrality and keyplayer modules), search and analyze clusters of nodes (communities module), perform set operations between networks (set module), gen-

erate networks with different topological organizations (e.g., random, scale-free, and small-world networks, generate module), and convert and load/save networks using different data formats (e.g., adjacency matrix, edge list, SIF and dot, convert module).

Features

Centrality measures for groups

Pyntacle tackles the problem of identifying key-player nodes that, together, optimally diffuse something through a network or maximally disrupt or fragment a network when removed. It further extends the standard network centrality measures of degree, closeness, and betweenness (refer to [20] for a clear introduction and to [21] for further theoretical explanations) to groups rather than individual elements. To this regard, these methods are a direct generalization of the corresponding individual measures, in such a way that if, e.g., group degree and degree are applied to groups consisting of single elements, they yield identical results. The classes of algorithms are thus two: one that measures the importance of a set on the basis of its impact on the remaining nodes of a network and another that does it by considering the sole properties of the elements of a set.

The former class is composed by the DF (KPP-Neg; cf. Eq. 2 in Methods), DR (cf. Eq. 3), and m-reach (KPP-Pos; cf. Eq. 4) algorithms [10]. KPP-Neg measures the fragmentation of a network because of a set. KPP-Pos measures the overall cohesion that members of a set have with the remainder of the network. As described in the Methods section, DF measures the degree of reachability of a set of nodes, taking also into account the degree of cohesion of the set. m-reach counts the number of unique nodes reached by any member of a set in m links or fewer. DR is the weighted proportion of all nodes reached by the set, where nodes are inversely weighted by their minimum distance from the set.

The latter class is formed by the group-degree centrality measure, which accounts for the number of non-group nodes that are connected to group members (cf. Eq. 5 in Methods); the group-betweenness centrality measure, which measures the proportion of (shortest) paths connecting pairs of non-group members that pass through the group (cf. Eq. 6); and the group closeness, which sums the distances from the group to all vertices outside the group (cf. Eq. 7).

Search strategies for optimal sets

When the aim is not to quantify the centrality of a specific set of nodes but that of discovering which is/are the most central set(s) in a network, search heuristics might come in handy. In particular, Pyntacle implements a “greedy optimization” search heuristics presented in [10] and a brute-force combinatorial optimization search strategy (cf. Search algorithms section in Methods). The former progressively replaces the components of a starting random set with all other nodes of a graph, calculating one of the aforementioned centrality metrics for that group, and then stops when a suboptimal solution is obtained. The latter loops through all possible groups of a predefined size and returns only those exhibiting the best scores for any of the centrality measure. It is immediate that the computational complexity of the heuristic method is much lower than that of the exact method, at the cost of suboptimal solutions. The brute-force search yields exact solutions but is computationally impracticable for big networks. The choice of a heuristic approach is due to its scalability to large-scale networks, while exact solutions are provided

for smaller biological networks, for which there is no significant computational burden. It has to be noted that more efficient search strategies for large networks exist: Integer Linear Programming for exact solutions or metaheuristic approaches, such as population-based incremental learning methods [13].

Exploration of cross-talk pathways of sparse real-world networks

Real-world biological networks exhibit hierarchical organizations, where subnetworks (e.g., signaling pathways) are bridged by cross-talk links [22]. A number of developmental processes rely on cross-talk, where their aberrant regulation has been found to be associated with inflammatory response defects, as well as cancer and neurodegeneration [23, 24]. Together with the observation that causal molecular pathways often coincide with the shortest molecular paths between known disease-associated components (cf. the network parsimony principle [1]), these render the study of cross-talk in networks fundamental. Pyntacle eases the exploration of cross-talk by set operations on graphs. Individual networks can thus be compared (union, intersection, and difference) or merged and then studied topologically.

These networks are typically sparse and can be analyzed using algorithms that work best with graphs with a few edges. Pyntacle is optimized to work with increasingly large and complex networks. It lets the user assess the extent of sparseness of a network through mathematical indices, including the compactness and completeness indices [25, 26]. In addition, it chooses the best implementation of computationally heavy algorithms at run-time (e.g., the search for all the shortest paths), according to the available hardware (i.e., single or multi-core processors and GPU-enabled graphics cards) and some network global metrics, including the sparseness.

Data format compatibility and reporting

Pyntacle is compliant with the Cytoscape SIF data format and with the dot network data format. It can input and output adjacency matrices and edge lists as textual files, as well as serialized binary Python objects. Graph, node, or edge attributes can be imported/exported from/to file.

Pyntacle can report any analysis result in 2 formats: as textual files and as rich HTML files. In particular, the PyntacleInk module outputs an interactive, automatically generated web page that displays the graph, its attributes, and all the results of the analyses that were performed on it.

Implementation

Pyntacle is accessible via command line and exposes a Python API for fine-tuning its algorithms. It depends on iGraph [17] for handling the graph data structure and borrowing some basic local and global topological measures and network generators.

Heavy computations of new algorithms are just-in-time compiled to native machine instructions by Numba [27] and thus run on multi-process CPU or NVIDIA-compatible GPU hardware, if available in the hosting computing infrastructure (experimental feature only accessible through APIs in version 1.3). Differently from similar packages, this allows Pyntacle to process graphs with thousands of nodes, thus helping it manage, for example, the whole human transcriptome and other networks of comparable sizes. Moreover, GPU acceleration provides high-speed computing of Pyntacle’s algorithms, thereby making heavy and long-running tasks feasible.

The PyntacleInk visualizer exploits HTML5, Javascript, and Sigma to produce an interactive representation of the input

graph, its base metrics, and a graphic rendering of the results of most of Pyntacle's algorithms (KPP, group centrality, graph generation, set operations, community detection, Fig. 6B). A graph can be displayed using different layouts (i.e., Random, Circular, ForceAtlas, Fruchterman-Reingold), and the canvas render enables visualization and smooth interaction with graphs $\leq 5,000$ nodes in size, using a web browser of a standard desktop PC. All the information about a graph and the analyses that were performed on it are stored in a JSON file; this dictionary is updated with new information whenever a new run of analysis is performed on the same graph, allowing the user to simultaneously explore the results of different algorithms and—through the use of timestamps—the results of the same algorithm run with different parameters over time. Any graphical representation can be exported as vector graphics (SVG) or PNG screenshots.

Finally, Pyntacle is fully compatible with Jupyter Notebook.

Benchmarks

Compared with the keyplayer 1.0.3 R package [28] and KeyPlayer 1.44 [29], Pyntacle has the following attributes: (i) is available for Windows, Mac, and Linux OS; (ii) is available as both command line tool and API; and (iii) allows management of real-world graphs in a computationally efficient way.

Wall-clock time comparisons of Pyntacle and keyplayer, when searching for optimal kp-sets of some real and simulated graphs, are shown in Fig. 4. Noteworthy is that KeyPlayer is not rigorously testable here because it is a Windows-only GUI-based application.

Random networks were generated according to the Erdős-Rényi model. Six random networks, 3 with 100 nodes and 3 with 1,000 nodes, were generated. These 2 groups of networks differed for their wiring probability, which varied as 0.3, 0.5, and 0.7. This probability is a kind of weighting function, which ranges from 0 to 1, with bigger numbers producing denser networks. Four other real networks were used: the network representing strong advice-seeking ties in a global consulting company [10] (32 vertices and 55 edges); the parasite-host food web of the Carpinteria Salt Marsh Reserve (128 vertices and 1,198 edges) [15]; the *Caenorhabditis elegans* connectome (a modified version of the network published in [30], 279 vertices and 1,960 edges); and a high-quality *C. elegans* protein-protein interaction network (3,303 vertices and 5,561 edges, downloaded from the Agile Protein Interactomes DataServer [APID] [31, 32]).

Wall-clock times were measured 3 times for each network and centrality algorithm. DR, m-reach, and DF were the only 3 algorithms in common between the 2 software packages. The sub-optimal sets of size 2 were determined by both software packages using their own implementations of the greedy optimization search algorithm (cf. Search algorithms in the Methods section). Starting from the 100-node random networks, Pyntacle computed all indices in fractions of seconds (or a few seconds for DF), irrespective of the wiring probability. keyplayer computed the same indices of the same networks in 4–9 minutes. Considering the 1,000-node networks, keyplayer completed the computation of all indexes in >1 day, while Pyntacle took a few minutes to 5 hours (DF). Similarly, real networks were analyzed in fractions (or tenths for DF) of seconds by Pyntacle and in a few seconds to 1 hour by keyplayer, which took >1 day to analyze the APID network, as opposed to Pyntacle, which ran for a few minutes to 17 hours. Generally, Pyntacle was 40–3,900 times faster than keyplayer, depending on the test.

The brute-force search algorithm yields exact solutions at the cost of an intrinsic combinatorial complexity. However, its com-

putational load can be split into parallel processors. In Pyntacle, the best solutions are obtained after the enumeration of all possible groups of nodes and the calculation of their topological indices. Calculations are in fact independent from each other and hence suitable to being executed in parallel. When applied to our test networks with the aim of calculating the DR index, we verified that the smaller ones (≤ 100 nodes) have benefited from parallel execution only to a limited extent. While the strong advice-seeking ties in global consulting company network exhibited the best speedup with the use of 4 computing cores (1.76 \times), before decreasing its performance, the net execution time improvement consisted in fact of only 82 milliseconds, on average (Fig. 5C). Similarly, 100-node random networks, proportionally to the rewiring probability, achieved the best speedup values with 16 cores ($\sim 8\times$), with an improvement of just ~ 4 seconds (Fig. 5B). As expected, bigger networks benefited from parallel execution increasingly with the number of nodes. The Carpinteria network achieved the best speedup record ($\sim 11\times$) with 16 cores, although saving just 7 seconds of computation, while the connectome peaked at $\sim 25\times$ with 32 cores (Fig. 5A). The computations of 1,000-node random networks scaled well up to 16 nodes, exhibiting comparable speedups of $\sim 7\times$, $\sim 6\times$, and $\sim 6\times$ when the rewiring probability was varied from 0.3 to 0.5 and 0.7, respectively. The bigger APID network exhibited the best performance with 32 cores, achieving a speedup of $\sim 29\times$ and terminating the computation 23 hours earlier than the non-parallel run (Fig. 5D).

Although these are far from being linear speedups, the advantage and efficacy of parallel computing strategies is evident for big networks. These results can be reproduced using a Docker image available from the Pyntacle website.

Analyses

Case Study 1—protein-protein interaction interface

NADH dehydrogenase [ubiquinone] flavoproteins 1 and 2 (NDUFV1 and NDUFV2) are 2 core subunits of the mitochondrial respiratory Complex 1 [33]. Their interaction is mediated by 138 interface residues (Fig. 6A).

We have built a network whose edges linked interacting residues of the 2 proteins with the aim of identifying key residues at the interface between the 2 proteins and whose mutations might significantly affect their interaction (Fig. 6B). Thus, we computed several local topological metrics for these residues, e.g., degree, betweenness, closeness, radiality, and a few others, but none of them were shown to correlate appreciably with the contribution provided by each residue (Supplementary Fig. S1) on the NDUFV1-NDUFV2 interaction energy ($\Delta\Delta G$, expressed in kcal/mol and calculated with FoldX [34], see Methods): a maximum Pearson correlation of 0.32 was observed between $\Delta\Delta G$ and betweenness.

We then applied Pyntacle to the network, searching for the best positive and negative key-player sets of size 2 (colored in blue and red, respectively, in Fig. 6, Supplementary Data S4). The residues Glu161 and Tyr46 of NDUFV1 were identified as the best negative key players in the network (according to both F and DF metrics), namely, their removal was estimated to maximally fragment the network and thus potentially hamper the interaction between the 2 proteins. This was further confirmed by their energetic contributions to the interaction when mutated to alanine ($\Delta\Delta G +0.9$ and $+3.7$ kcal/mol, respectively, for Glu161 and Tyr46). Moreover, Glu161 of NDUFV1, paired with Leu234 of NDUFV2, and Cys125 of NDUFV1, paired with Tyr46, were identi-

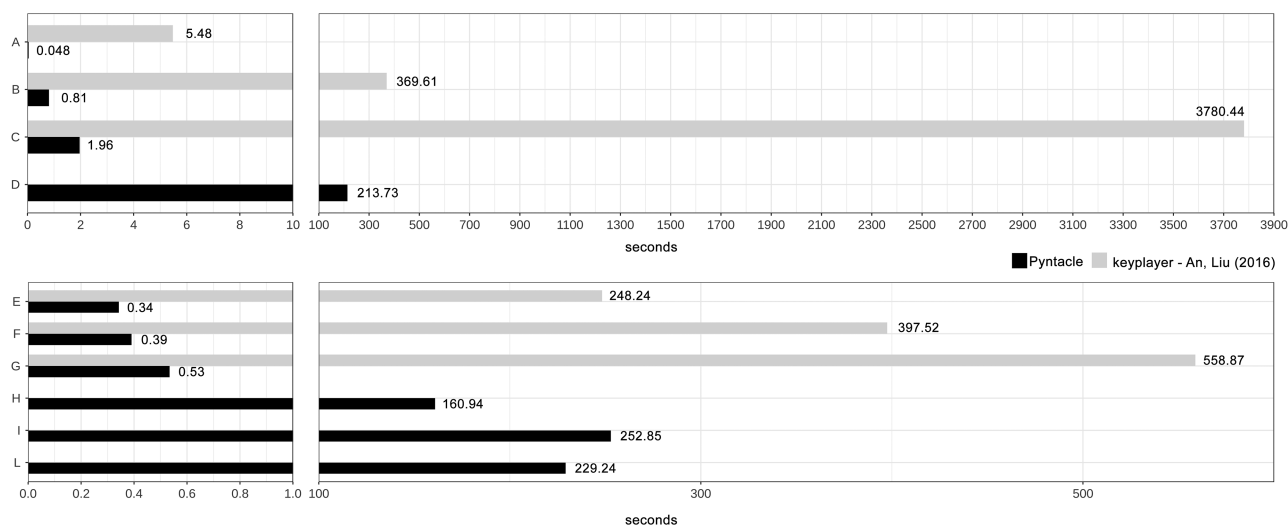


Figure 4: Greedy optimization search, metrics: DR. (A) Strong advice-seeking ties in global consulting company [10]; (B) parasite-host food web of the Carpinteria Salt Marsh Reserve [15]; (C) *C. elegans* connectome; (D) high-quality *C. elegans* protein-protein interaction network (APID); Erdős-Rényi random networks with (E-G) 100 nodes and rewiring probability $p = 0.3, 0.5,$ and 0.7 ; (H-L) 1,000 nodes and $p = 0.3, 0.5,$ and 0.7 .

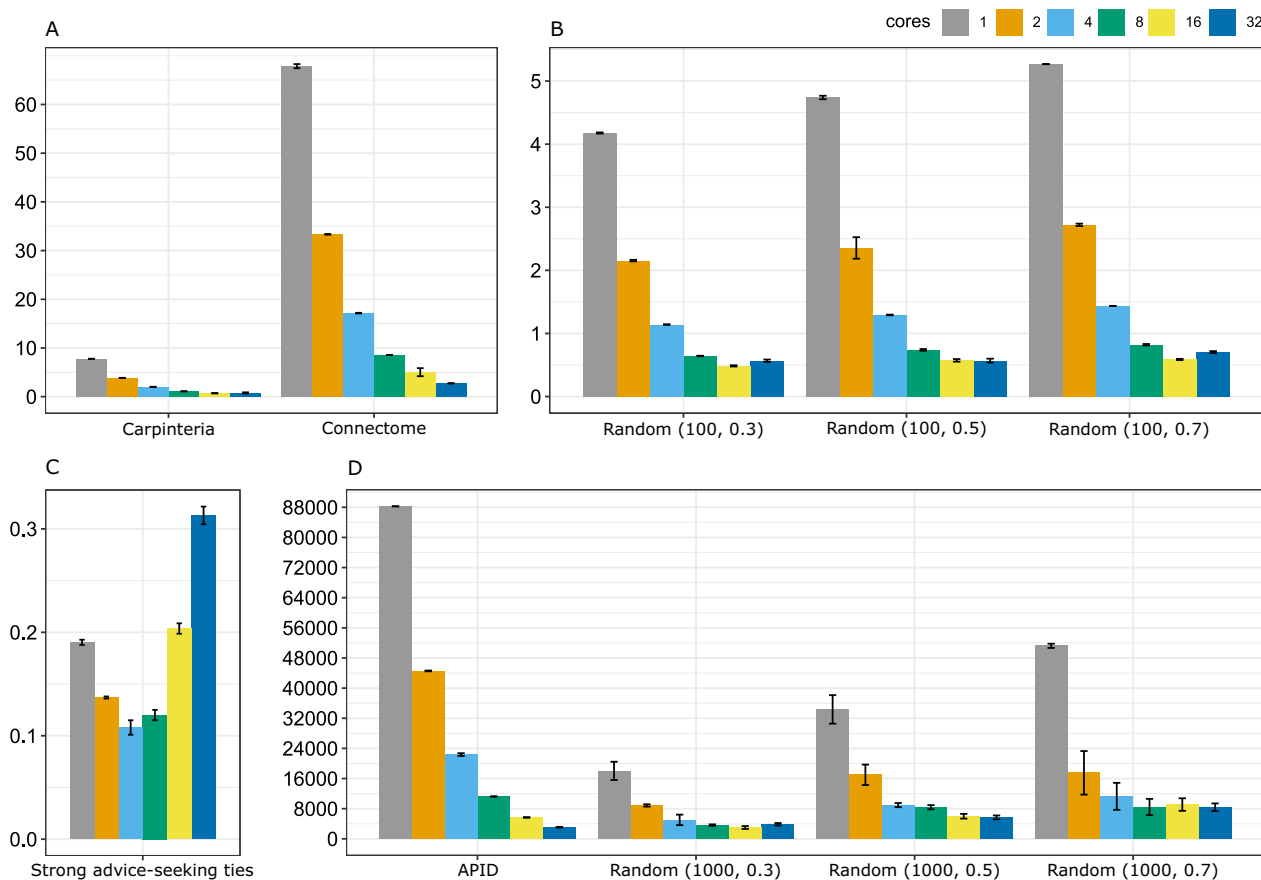


Figure 5: Improvement of execution times (in seconds) of Pyntacle using parallel computing on different networks using increasing numbers of computing cores. Bars represent mean and SD values.

fied as the best positive key-player pairs (respectively calculated with the DF and m-reach metrics), namely, they resulted in being immediately reachable from the remaining network by direct links or indirect links joining close neighbor residues.

Contrary to Glu161 and Tyr46, Leu234 and Cys125 neither exhibited a significant $\Delta\Delta G$ when mutated to alanine nor were characterized by high values of local topological metrics. For these reasons, they would have been overlooked by all other

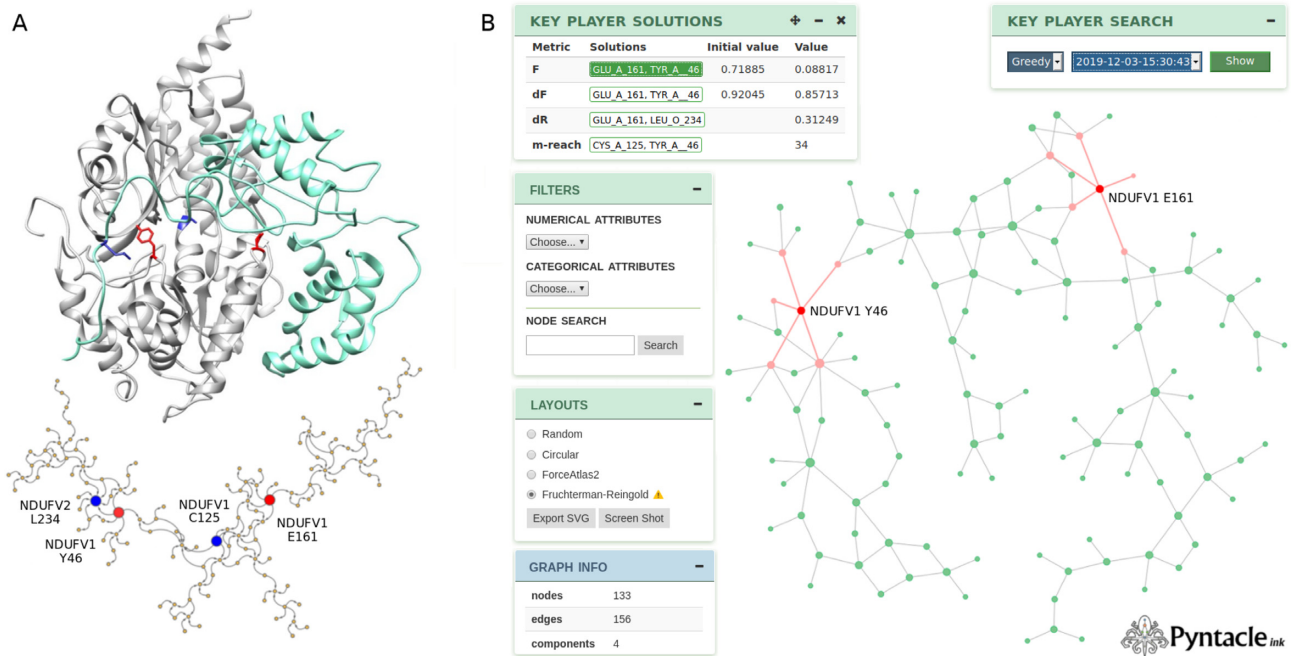


Figure 6: A: Representation of the interaction between NDUFV1 (white) and NDUFV2 (cyan) (PDB id 5xtld). The residues forming the interaction interface are represented as a graph (bottom) connecting residues close in space. Positive and negative key players are colored in blue and red, respectively, in both the interaction structure and interaction network. B: The PyntacleInk viewer. Different menus can be used to (i) visualize all the analyses that have been performed on a network, (ii) visualize the network general metrics, (iii) filter nodes by attributes, and (iv) change the overall layout of the network.

techniques, even the more accurate and computationally intensive, such as alanine scanning [35]. Other than being computationally demanding and impracticable for large-scale analyses, alanine scanning is known to be blind to residues that are chemically similar to alanine, thereby ignoring their epistatic features, which are critical in some regions of the interaction interface.

All these issues are overcome with Pyntacle, which makes it possible to look for topologically important groups of residues between proteins efficiently and regardless of their chemical structure.

Case Study 2—miRNA-miRNA interaction network

MicroRNAs (miRNAs) are small RNA molecules (18–25 nucleotides) able to regulate gene expression levels through different cellular mechanisms, the most important of which is that a miRNA can recognize different messenger RNAs as targets and, at the same time, one of those targets can be recognized by multiple miRNAs. Owing to the renowned role played by miRNAs in tumorigenesis and cancer progression [36], we have analyzed a miRNA interaction network of patients affected by breast cancer.

Expression data of healthy and tumor tissue samples of 87 patients affected by breast cancer were retrieved from The Cancer Genome Atlas (TCGA) and were used to wire 487 miRNAs in 2 correlation networks, built on healthy and tumor samples, respectively. A functional association between any 2 miRNAs was assumed to exist if the absolute values of the Pearson correlation coefficients of their expression levels exceeded 0.5 (cf. Case Study 2 in Methods).

The healthy network was very dense and highly connected (average degree of 40.4), as opposed to the tumor network, which was mostly disconnected (average degree of 17.4). Both networks were analyzed with Pyntacle, which identified miR-1307-3p and miR-140-3p as negative key players and miR-136-5p, miR-484,

and miR-127-5p as positive key players of the healthy tissue network. These are all associated with the onset and development of breast cancer and some are even used as markers for prognosis [37–42] (Supplementary Data S5).

In the tumor network, Pyntacle identified miR-192-5p, miR-483-5p, and miR-577 as negative key players and miR-324-5p and miR-337-3p as positive key players, all involved in proliferation, cell migration, and metastasis of breast cancer [43–47] (Supplementary Data S6). It has to be noted that while these miRNAs individually have high betweenness values, it would not have been possible to infer a possible synergistic interaction between them without the key-player analysis of Pyntacle.

Discussion and Concluding Remarks

The motivations behind Pyntacle come from the constant growth of experimental data sets and from the increasing need to represent and analyze them with computationally efficient methods. This first release of Pyntacle has been designed with these aims and thus with the intent to help researchers from different scientific fields and with different levels of computing skills to approach network biology and benefit from its analytical tools.

We showed the attributes of Pyntacle and its versatility in dealing with different problems, all of which are traceable to possibly big networks of interactive elements. In one case study, Pyntacle was thus able to identify key amino acids that were greatly contributing to the formation of a protein-protein interaction interface. This otherwise time-consuming task was accomplished very efficiently by translating the problem into a network analysis task, compared with current approaches that take tens of minutes to handle even small interaction interfaces. Another similar and detailed case study can be found in [48]. In a second case study, Pyntacle was used to analyze the TCGA data

set of miRNA expression in breast cancer to build miRNA-miRNA networks. These were analyzed in search of miRNAs that were occupying key positions in the network, which were later recognized as already known biomarkers or responsible for the onset and progression of breast cancer.

In conclusion, Pyntacle represents a starting point for large-scale network biology studies. Being a modular framework, it will be expanded to handle weighted networks, in the near future, and directed networks, immediately after. These features, which are shared with some other key-player detection methods and tools [28, 49, 50], are relevant to make Pyntacle fully capable of analyzing all kinds of biological networks. It will also be enriched with new optimization search algorithms and with a new algorithm to compute the set nestedness. New applications and use-cases are envisaged, the currently most concrete being one concerning the analysis of trajectories of molecular dynamics simulation of proteins.

Potential Implications

The main problem concerning most of the currently available network analysis tools, which is also the main reason why we made Pyntacle, is that these do not handle networks of medium to big sizes efficiently. This issue not only relates to the big networks and the suitability and effectiveness of the currently available algorithms to analyze them but also their practicability over a reasonable time axis and in terms of required computing resources. This point is critical for most fields of research, from biology to medical and social sciences, where systems are naturally big and complex (e.g., the whole proteome, the diseasome, and socialnomics, to mention a few). In particular, a field that at the time of this writing is hitting the headlines, i.e., epidemiology, is greatly developing in terms of the capability to draw contagion maps and predict infection growth over time. These maps are actually networks where nodes are people and edges are relationships that occurred in recent and short periods of time. There are many ways these networks could be studied. One way could be determining the leading front of the infection, namely, a group composed of healthy people that are close to the affected people and highly social and to administer a vaccine to curb the infection. Another possibility would be that of determining the minimum possible number of communication routes to be closed at a national level to implement a clever lockdown. These and several other options may be implemented in Pyntacle using ad hoc algorithms and computing protocols tailored for big networks.

Methods

Key-player and group-centrality metrics

Pyntacle tackles the problem of identifying key-player nodes that, together, optimally diffuse something through a network or maximally disrupt or fragment a network when removed. The classes of algorithms are thus two: one that measures the importance of a set on the basis of its impact on the remaining nodes of a network and another that does it by considering the sole properties of the elements of a set. The former class, also known as KPP-Neg, measures the fragmentation of a network because of the removal of a set of nodes. It is composed by the F metrics:

$$F = 1 - \frac{\sum_k s_k(s_k - 1)}{n(n-1)}, \quad (1)$$

which are based on the size s_k of its components k ; and by the DF metrics:

$$DF = 1 - \frac{2 \sum_{i>j} (1/d_{ij})}{n(n-1)}, \quad (2)$$

where d_{ij} denotes the distance between the i th and the j th node. It ranges from 1, when all nodes are adjacent, to 0, when all nodes are isolates.

The latter class, also known as KPP-Pos, measures the overall cohesion that members of a set have with the remainder of the network and is made up of:

$$DR = \frac{\sum_j (1/d_{Kj})}{n}, \quad (3)$$

where n is the size of the graph and d_{Kj} denotes the minimum distance (shortest path) between any member i of the set K and the remaining nodes j in the graph. Similarly, the m -reach metric counts how many unique nodes can be reached from K in m steps or less. The formulation is:

$$C_K = \sum_{j \in V \setminus K} \cup \frac{1}{d_{ij}}, \quad (4)$$

where C_K ranges from 0 to $n - k$, n representing the size of the graph and k that of the considered set of nodes. It is important to notice that this index assumes that all paths of length m or less are equally important and that all paths longer than m are wholly irrelevant [10].

Pyntacle further extends the standard network centrality measures of degree, closeness, and betweenness to groups of nodes, in a way that if, e.g., group-degree and degree centrality measures are applied to groups consisting of single elements, they yield identical results. This class of metrics is formed by the group-degree centrality measure, which is intended as the number of non-group nodes that are connected to group members. Multiple ties to the same node are counted only once. It is defined as follows:

$$GD_K = \frac{\sum_{i \in K, j \in V \setminus K} a_{ij}}{\|V \setminus K\|}, \quad (5)$$

where $a_{ij} = 1$ when i and j are adjacent nodes, if $i \in K$ and $j \in \{V \setminus K\}$, and counting $a_{i,j}$ and $a_{v,j}$ only once $\forall v \in K$ when $a_{i,j} = 1$ and $a_{v,j} = 1$. Hence, the group-degree centrality ranges from 0 to 1 if the group K is completely isolated or fully connected to all other nodes. Group-betweenness centrality of a set K is defined as the number of shortest paths connecting any 2 nodes u and v passing through K over the number of all paths between the two.

$$GB_K = \frac{p_{u,v}(K)}{p_{u,v}}, \quad (6)$$

where u and v are any pair of nodes not belonging to the group K , $p_{u,v}(K)$ represents the number of shortest paths connecting u and v and that traverses K , while $p_{u,v}$ is the total number of shortest paths between u and v . Group closeness of a group K is defined as the sum of the minimum, maximum, and average distances from the nodes belonging to the group to all other nodes outside

the group.

$$GC_K = \frac{\sum_{j \in V \setminus K} \bar{d}_{Kj}}{\|V \setminus K\|}, \quad (7)$$

where \bar{d}_{ij} is the minimum, maximum, or average distance between nodes in K and all other nodes.

Search algorithms

When the aim is not to quantify the centrality of a specific set of nodes but to discover which is/are the most central set/s of nodes in a network, search heuristics might come in handy. In particular, Pyntacle implements a greedy optimization search heuristics presented in [10], and a brute-force combinatorial optimization search strategy. The former follows this naive algorithm:

Algorithm 1 Greedy optimization search

```

1: procedure GREEDYSEARCH(k: int, V: list, FUNC: lambda)
2:   score = 0.0
3:   scoredict = {} ← hashmap
4:   halt = false
5:   K = DRAWNODES(k, V)
6:   while not halt do
7:     for all u ∈ K do
8:       for all v ∈ V \ K do
9:         Ktemp = SWAP(K, u, v)
10:        scoretemp = FUNC(Ktemp)
11:        scoredict[(u, v)] = scoretemp
12:        ubest, vbest = GETPAIRSWITHBESTSCORE(scoredict)
13:        bestscore = scoredict[(ubest, vbest)]
14:        if bestscore > score then
15:          K = swap(K, ubest, vbest)
16:          score = bestscore
17:        else
18:          halt = true
19:   return (K, score)
20: end

```

FUNC is an appropriate key-player metrics; “drawnodes” is a function that randomly picks k nodes from V , which contains all nodes of the network; “swap(K, u, v)” substitutes the element u in K with v ; “getPairsWithBestScore” is a function that returns the pairs that yielded the best centrality score. This method progressively replaces the components of a starting random set K with all other nodes of a graph, calculating one of the aforementioned centrality metrics for that group and then stopping when a sub-optimal solution is obtained.

The brute-force combinatorial optimization search strategy implemented in Pyntacle loops through all possible groups of a predefined size (k) and returns only those exhibiting the best scores for any of the previous centrality measure (F). Even if the computation can be performed in parallel (the “for all” loop below), it is obvious that the computational complexity of the heuristic method is much lower than that of this method, at the cost of yielding suboptimal solutions. The algorithm below returns exact solutions but is computationally impracticable with big networks.

GENERATECOMBINATIONS is a function that generates all possible sets of nodes of size k picking nodes from V .

Algorithm 2 Brute-force search

```

1: procedure BRUTEFORCESEARCH(k: int, V: list, FUNC: lambda)
2:   bestscore = 0.0
3:   scoredict = {} ← hashmap
4:   allsets = GENERATECOMBINATIONS(k, V)
5:   for all set ∈ allsets do
6:     scoretemp = FUNC(set)
7:     scoredict[set] = scoretemp
8:   bestsets = GETSETSWITHBESTSCORE(scoredict)
9:   bestscore = scoredict[bestsets]
10:  return (bestsets, bestscore)
11:  end

```

Set operations on graphs

Graph union, $G1 \cup G2$, is implemented as $(V1 \cup V2, E1 \cup E2)$, namely, as the union of nodes (V) and edges (E). Graph intersection is defined as $G1 \cap G2 = (V1 \cap V2, E1 \cap E2)$, where only common nodes and edges are reported in the resulting graph. The difference between $G1$ and $G2$ results in a graph with nodes and edges only present in $G1$ and not in $G2$. Because the difference between graphs is not reciprocal, $G1 - G2 \neq G2 - G1$.

Case Study 1

Chains A and O of the Protein Data Bank (PDB) structure 5xtl were considered for the analysis of NDUFV1 and NDUFV2, respectively. First, the structure has been repaired (RepairPDB module of FoldX) and thus allows more relaxed residue side-chain rotamers and the solution of clashes. Then, residues located at the interaction interface and the interaction energy were determined with the AnalyseComplex module of FoldX. Alanine scanning of the interface residues was performed by substituting each amino acid with an alanine residue (Build-Model module). The interaction energy of each mutant has been calculated with the AnalyseComplex module to determine the $\Delta\Delta G$ of the mutant compared to the wild-type interaction structure. A residue-residue interaction network was built, which connected residues belonging to different chains in the complex if any of their non-hydrogen atoms (both backbone and side-chain) were within a 4.5 Å radius from each other. The network was then analyzed with the keyplayer module of Pyntacle and a greedy search algorithm was used to find the optimal key-player sets of size 2, considering all available metrics (F and DF , as KPP-Neg; and DR and m -reach, with m set to 2, as KPP-Pos).

Case Study 2

Expression levels of 547 miRNAs in 87 healthy and 87 tumor breast samples were retrieved from TCGA. Separately for healthy and tumor individuals, correlations of expression between any possible pairs of miRNAs (149,331 total pairs) were calculated by Pearson correlation coefficient. Only significant values that exceeded ± 0.5 were considered to represent edges connecting miRNAs in the healthy and tumor networks. In both networks, the best KPP-Pos and KPP-Neg sets of size 2 were sought using Pyntacle’s greedy optimization search and calculating F and DF , as KPP-Neg; and DR and m -reach, with m set to 1, as KPP-Pos.

Availability of Source Code and Requirements

- Project name: Pyntacle
- Project home page: <http://pyntacle.css-mendel.it>
- Operating systems: Linux, Mac, and Windows
- Programming language: Python 3.7+
- Other requirements: CUDA toolkit (optional)
- License: GNU GPL 3.0
- RRID:SCR_019030
- bio.tools ID: biotools:pyntacle

Source code is stored in GitHub. Installation procedures, tutorials, case studies, and a Docker container are all available from the Pyntacle website.

Availability of Supporting Data and Materials

An archival copy of the supporting data, for the reproduction of the case studies, is also available via the GigaScience repository, GigaDB [51].

Additional Files

Supplementary Figure S1.

Supplementary Data S1. Centrality metrics calculated for the network in Figure 1

Supplementary Data S2. Nestedness of the best sets by group-centrality metrics for the network in Figure 1

Supplementary Data S3. Nestedness of the best KP sets for the network in Figure 1

Supplementary Data S4. Local topological metrics and $\Delta\Delta G$ values of the residues represented in Figure 6A

Supplementary Data S5. Local topological metrics for the “healthy” network described in Case Study 2

Supplementary Data S6. Local topological metrics for the “tumor” network described in Case Study 2

Abbreviations

API: Application Programming Interface; APID: Agile Protein Interactomes; CPU: central processing unit; GPU: graphics processing unit; GUI: graphical user interface; KPP-Neg: Key-Player Problem/Negative; KPP-Pos: Key-Player Problem/Positive; miRNA: microRNA; NADH: nicotinamide adenine dinucleotide–hydrogen (reduced); PDB: Protein Data Bank; TCGA: The Cancer Genome Atlas.

Competing Interests

The authors declare that they have no competing interests.

Funding

This study was supported by the Italian Ministry of Health (Ricerca Corrente 2018-2020) and by the “5 × 1000” voluntary contribution.

Authors’ Contributions

L.P. designed and performed the experiments; M.T. implemented PyntacleInk and took care of the package maintenance and deployment; D.C. wrapped the iGraph modules and performed the benchmarks; T.B., S.C., and F.P. contributed to the case study analysis; M.C. contributed to the definition of the case stud-

ies; F.J. oversaw the implementation of the topology metrics; T.M. designed and implemented the software and oversaw the project.

Acknowledgments

We are grateful to Christoph Gohlke, from the Laboratory for Fluorescence Dynamics, University of California, for providing Windows unofficial binaries of the iGraph Python extension package; Juliana Pereira for testing and constructive discussions; and to NVIDIA Corporation for supporting this research. The results of Case Study 2 shown here are in whole or part based upon data generated by the TCGA Research Network.

References

1. Barabasi AL, Gulbahce N, Loscalzo J. Network medicine: a network-based approach to human disease. *Nat Rev Genet* 2011;**12**:56–68.
2. Goh KI, Cusick ME, Valle D, et al. The human disease network. *Proc Natl Acad Sci U S A* 2007;**104**:8685–90.
3. Jeong H, Mason SP, Barabási AL, et al. Lethality and centrality in protein networks. *Nature* 2001;**411**:41–2.
4. Luo H, Lin Y, Gao F, et al. DEG 10, an update of the Database of Essential Genes that includes both protein-coding genes and non-coding genomic elements. *Nucleic Acids Res* 2014;**42**:D574–80.
5. Wachi S, Yoneda K, Wu R. Interactome-transcriptome analysis reveals the high centrality of genes differentially expressed in lung cancer tissues. *Bioinformatics* 2005;**21**:4205–8.
6. Jonsson PF, Bates PA. Global topological features of cancer proteins in the human interactome. *Bioinformatics* 2006;**22**:2291–7.
7. Xu J, Li Y. Discovering disease-genes by topological features in human protein–protein interaction network. *Bioinformatics* 2006;**22**:2800–5.
8. Freeman LC. A set of measures of centrality based on betweenness. *Sociometry* 1977;**40**:35–41.
9. Everett MG, Borgatti SP. The centrality of groups and classes. *J Math Sociol* 1999;**23**(3):181–201.
10. Borgatti SP. Identifying sets of key players in a social network. *Comput Math Org Theor* 2006;**12**:21–34.
11. Boginski V, Commander CW. Identifying critical nodes in protein-protein interaction networks. In: Butenko S, Chaovalitwongse WA, Pardalos PM, eds. *Clustering Challenges in Biological Networks*. World Scientific; 2009.
12. Csermely P, Korcsmáros T, Kiss HJM, et al. Structure and dynamics of molecular networks: A novel paradigm of drug discovery: A comprehensive review. *Pharmacolo Ther* 2013;**138**:333–408.
13. Lalou M, Tahraoui MA, Kheddouci H. The critical node detection problem in networks: a survey. *Comput Sci Rev* 2018;**28**:92–117.
14. Walteros JL, Pardalos PM. Selected topics in critical element detection. In: Daras NJ, ed. *Applications of Mathematics and Informatics in Military Science*. New York, NY: Springer; 2012, doi:10.1007/978-1-4614-4109-0_2.
15. Capocéfalo D, Pereira J, Mazza T, et al. Food web topology and nested keystone species complexes. *Complexity* 2018;**2018**, doi:10.1155/2018/1979214.
16. Almeida-Neto M, Guimarães P, Guimarães PR, Jr, et al. A consistent metric for nestedness analysis in ecologi-

- cal systems: reconciling concept and measurement. *Oikos* 2008;**117**(8):1227–39.
17. Csárdi G, Nepusz T. The igraph software package for complex network research. *InterJ Complex Syst* 2006, **1695**(5):1–9.
 18. Hagberg A, Swart P, Chult DS. Exploring network structure, dynamics, and function using NetworkX. In: Varoquaux G, Vaught T, Millman J, eds. *Proceedings of the 7th Python in Science Conference (SciPy)*. 2008:11–16.
 19. Jacobs S, Khanna A, Madduri K, et al. influenceR: Software Tools to Quantify Structural Importance of Nodes in a Network. 2015. <https://cran.r-project.org/package=influenceR>. Access date: 7 January 2020.
 20. Borgatti SP, Everett MG. A Graph-theoretic perspective on centrality. *Social Netw* 2006;**28**:466–84.
 21. Freeman LC. Centrality in networks: I. Conceptual clarification. *Social Netw* 1979;**1**:215–39.
 22. Vert G, Chory J. Crosstalk in cellular signaling: background noise or the real thing?. *Dev Cell* 2011;**21**(6):985–91.
 23. Zolezzi JM, Inestrosa NC. Wnt/TLR Dialog in neuroinflammation, relevance in Alzheimer's disease. *Front Immunol* 2017;**24**(8):187.
 24. Qu X, Tang Y, Hua S. Immunological approaches towards cancer and inflammation: a cross talk. *Front Immunol* 2018;**20**(9):563.
 25. Menniti S, Castagna E, Mazza T. Estimating the global density of graphs by a sparseness index. *Appl Math Comput* 2013;**224**:346–57.
 26. Mazza T, Romanel A, Jordán F. Estimating the divisibility of complex biological networks by sparseness indices. *Brief Bioinformatics* 2010;**11**(3):364–74.
 27. Crist J. Dask & Numba: Simple libraries for optimizing scientific Python code. In: 2016 IEEE International Conference on Big Data (Big Data), Washington, D.C. IEEE; 2016:2342–3.
 28. An WH, Liu YH. An R package for locating key players in social networks. *R J* 2016;**8**(1):257–68.
 29. Borgatti SP. KeyPlayer 1.44. 2019. <http://www.analytictech.com/keyplayer/keyplayer.htm>. Accessed 6 August 2019.
 30. Towilson EK, Vértes PE, Ahnert SE, et al. The rich club of the *C. elegans* neuronal connectome. *J Neurosci* 2013;**33**(15):6380–7.
 31. Alonso-López D, Campos-Laborie FJ, Gutiérrez MA, et al. Agilè Protein Interactomes DataServer. 2019. <http://apid.dep.usal.es/>. Accessed 7 August 2019.
 32. Alonso-López D, Campos-Laborie FJ, Gutiérrez MA, et al. APID database: redefining protein-protein interaction experimental evidences and binary interactomes. *Database* 2019;**2019**, doi:10.1093/database/baz005.
 33. Guo R, Zong S, Wu M, et al. Architecture of human mitochondrial respiratory megacomplex I2III2IV2. *Cell* 2017;**170**:1247–57.
 34. Schymkowitz J, Borg J, Stricher F, et al. The FoldX web server: an online force field. *Nucleic Acids Res* 2005;**33**:W382–8.
 35. Weiss GA, Watanabe CK, Zhong A, et al. Rapid mapping of protein functional epitopes by combinatorial alanine scanning. *Proc Natl Acad Sci U S A* 2000;**97**:8950–4.
 36. Reddy KB. MicroRNA (miRNA) in cancer. *Cancer Cell Int* 2015;**15**:38.
 37. McGuires A, Brown JAL, Kerin J. Metastatic breast cancer: the potential of miRNA for diagnosis and treatment monitoring. *Cancer Metastasis Rev* 2015;**34**:145–55.
 38. Wang DY, Gendoo DMA, Ben-David Y, et al. A subgroup of miRNAs defines PTEN-deficient, triple-negative breast cancer patients with poorest prognosis and alterations in RB1, MYC, and Wnt signaling. *Breast Cancer Res* 2019;**21**(1):18.
 39. Pronina IV, Loginov VI, Burdennyy AM, et al. DNA methylation contributes to deregulation of 12 cancer-associated miRNAs and breast cancer progression. *Gene* 2017;**604**:1–8.
 40. Li Q, Yao Y, Eades G, et al. Downregulation of miR-140 promotes cancer stem cell formation in basal-like early stage breast cancer. *Oncogene* 2014;**33**:2589–600.
 41. Vos S, Vesuna F, Raman V, et al. miRNA expression patterns in normal breast tissue and invasive breast cancers of BRCA1 and BRCA2 germ-line mutation carriers. *Oncotarget* 2015;**6**:32115–37.
 42. Wang X, Zhu J. Mir-1307 regulates cisplatin resistance by targeting Mdm4 in breast cancer expressing wild type P53. *Thorac Cancer* 2018;**9**:676–83.
 43. Wang Z, Wang J, Yang Y, et al. Loss of has-miR-337-3p expression is associated with lymph node metastasis of human gastric cancer. *J Exp Clin Cancer Res* 2013;**32**(1):76.
 44. Zuo XL, Chen ZQ, Wang JF, et al. miR-337-3p suppresses the proliferation and invasion of hepatocellular carcinoma cells through targeting JAK2. *Am J Cancer Res* 2018;**8**:662–74.
 45. Yin C, Mou Q, Pan X, et al. MiR-577 suppresses epithelial-mesenchymal transition and metastasis of breast cancer by targeting Rab25. *Thorac Cancer* 2018;**9**:472–9.
 46. Cioce M, Valerio M, Casadei L, et al. Metformin-induced metabolic reprogramming of chemoresistant ALDHbright breast cancer cells. *Oncotarget* 2014;**5**:4129–43.
 47. Li JY, Jia S, Zhang WH, et al. Differential distribution of microRNAs in breast cancer grouped by clinicopathological subtypes. *Asian Pac J Cancer Prev* 2013;**14**:3197–203.
 48. Petrizzelli F, Biagini T, Barbieri A, et al. Mechanisms of pathogenesis of missense mutations on the KDM6A-H3 interaction in type 2 Kabuki syndrome. *Comput Struct Biotechnol* 2020;**18**:2033–42.
 49. Paudel N, Georgiadis L, Italiano G. Computing critical nodes in directed graphs. *J Exp Algorithmics* 2018;**23**:2.2.
 50. McGuires RM, Deckro RF, Ahner DK. The weighted key player problem for social network analysis. *Military Operations Res* 2015;**20**:35–53.
 51. Parca L, Truglio M, Biagini T, et al. Supporting data for “Pyntacle: a parallel computing-enabled framework for large-scale network biology analysis.” *GigaScience Database* 2020. <http://doi.org/10.5524/100779>.