A Fast Prize-Collecting Steiner Forest Algorithm for Functional Analyses in Biological Networks

A fast Prize-collecting Steiner Forest algorithm for Functional Analyses in Biological Networks

Murodzhon Akhmedov[1,2,3], Alexander LeNail[1], Francesco Bertoni[3], Ivo Kwee[2,3], Ernest Fraenkel[1], and Roberto Montemanni[2]

[1] Department of Biological Engineering, Massachusetts Institute of Technology, Cambridge, Massachusetts, USA
{akhmedov, lenail, fraenkel-admin}@mit.edu
[2] Dalle Molle Institute for Artificial Intelligence Research (IDSIA-USI/SUPSI), Manno, Switzerland
{murodzhon, roberto}@idsia.ch
[3] Institute of Oncology Research (IOR), Bellinzona, Switzerland
{francesco.bertoni, ivo.kwee}@ior.iosi.ch

Abstract. The Prize-collecting Steiner Forest (PCSF) problem is NP-hard, requiring extreme computational effort to find exact solutions for large inputs. We introduce a new heuristic algorithm for PCSF which preserves the quality of solutions obtained by previous heuristic approaches while reducing the runtime by a factor of 10 for larger graphs. By decreasing the draw on computational resources, this algorithm affords systems biologists the opportunity to analyze larger biological networks faster and narrow their analyses to individual patients.

Keywords: Prize-collecting Steiner Forest

1 Introduction

The Prize-collecting Steiner Tree Problem (PCST) is a widely studied problem in the combinatorial optimization literature [2]. In PCST, we are given a graph $G = (V, E)$ such that vertices are assigned prizes $\forall v \in V : p(v) \in \mathbb{R}^+$ and edges are weighted with costs $\forall e \in E : c(e) \in \mathbb{R}^+$. The objective is to find a tree $T = (V_t, E_t)$ which minimizes:

$$c(T) = \sum_{e \in E_t} c(e) + \sum_{v \notin V_t} p(v) \tag{1}$$

Vertices with prize $p(v) = 0$ are referred to as Steiner nodes, and vertices with $p(v) > 0$ are called terminal nodes [7].

By adding an artificial *root* node to the input graph with edges from *root* to every other node with weight ω, we construct an augmented graph $G' = (V \cup root, E \cup E_{root})$, where $E_{root} = e_{(root, v \in V)}$ with associated cost function $\forall e \in E_{root} : c'(e) = \omega$. The solution to the Prize-collecting Steiner Forest Problem

(PCSF) on graph G is the solution to PCST on G' with a slightly modified objective function [8]:

$$c(T) = \sum_{e \in E_t \setminus E_{root}} c(e) + \sum_{v \notin V_t} p(v) + \sum_{e \in E_{root}} c'(e) \tag{2}$$

The parameter ω regulates the number of selected outgoing edges from the root, which determines the number of trees in the forest. The final forest is obtained by removing the *root* from T as well as the edges emanating from it, such that the single tree solution connected by *root* becomes a solution of disconnected components. A desirable forest can be obtained by running the method for different values of ω.

The PCSF problem from combinatorial optimization maps nicely onto the biological problem of finding differentially enriched sub-networks in the interactome of a cell. An interactome is a graph in which vertices represent biomolecules and edges represent the known physical interactions of those biomolecules. We can assign prizes to vertices based on measurements of differential expression of those cellular quantities in a patient sample and costs to edges from confidence scores for those intra-cellular interactions from experimental observation (high confidence means low edge cost), yielding a viable input to the PCSF problem. Vertices of the interactome which are not observed in patient data are not assigned a prize and become the Steiner nodes.

An algorithm to approximately solve the Prize-Collecting Steiner Forest problem can then be applied against this augmented interactome, resulting in a set of subgraphs corresponding to subsections of the interactome in which functionally related biomolecules may play an important concerted role in the differentially active biological process of interest. Thus, PCSF, when applied in a biological context, can be used to predict neighborhoods of the interactome belonging to the key dysregulated pathways of a disease.

The PCSF problem is NP-hard, and therefore requires tremendous amounts of computation to determine exact solutions for large inputs. In biology, large networks are the norm. For example, the human Protein-Protein Interaction (PPI) network has some 15,000 nodes and 175,000 edges [1]. Adding metabolites for a more complete interactome yields inputs of some 36,000 nodes and over 1,000,000 edges. We require efficient algorithms to investigate these huge biological graphs for each patient, which requires solving PCSF many times.

We present a heuristic algorithm for PCSF which outperforms other heuristic approaches from the literature in computational efficiency for larger graphs by a factor of 10, while preserving the quality of the results. Our algorithm, MST-PCSF, builds from the ideas presented in [24, 25], using a greedy clustering pre-processing step which divides large input graph into smaller clusters, and a heuristic to find approximate solutions for each cluster.

2 Related Work

The authors in [2] initially studied the prize-collecting traveling salesman problem, and proposed a 3-approximation algorithm. A 2-approximation with $O(n^3 \log n)$

running time is proposed in [3] by using the primal-dual method, improved by [4] to $O(n^2 \log n)$ execution time. That runtime was maintained in [5], which improved the approximation factor to $2 - \frac{2}{n}$.

Exact methods were devised in [6,7] using mixed integer linear programming, and a branch-and-cut algorithm based on directed edges was proposed to solve the model. An exact row generation approach was presented in [9] using a new set of valid inequalities.

A relax-and-cut algorithm was studied in [10] to develop effective Lagrangian heuristic. A multi-start local search algorithm with perturbations was integrated with variable neighborhood search in [11]. Seven different variations of PCST were studied in [12], and polynomial algorithms were designed for four of them. Some lower bound and polyhedral analyses were performed in [13–16]. A tabu-search metaheuristic, and a combination of memetic algorithms and integer programming were employed in [17, 18] for PCST.

The application of the PCST approach in Biology has led to important results. The specific biological problems were to identify functions of molecules [19], to find protein associations [20], and to reconstruct multiple signaling pathways [21]. We have been developing heuristic and matheuristic PSCT algorithms [24, 25] for similar applications, and in this study we aim to extend these ideas to a forest approach in order to make more realistic biological inferences.

3 Algorithm: MST-PCSF, a fast heuristic algorithm for the PCSF problem

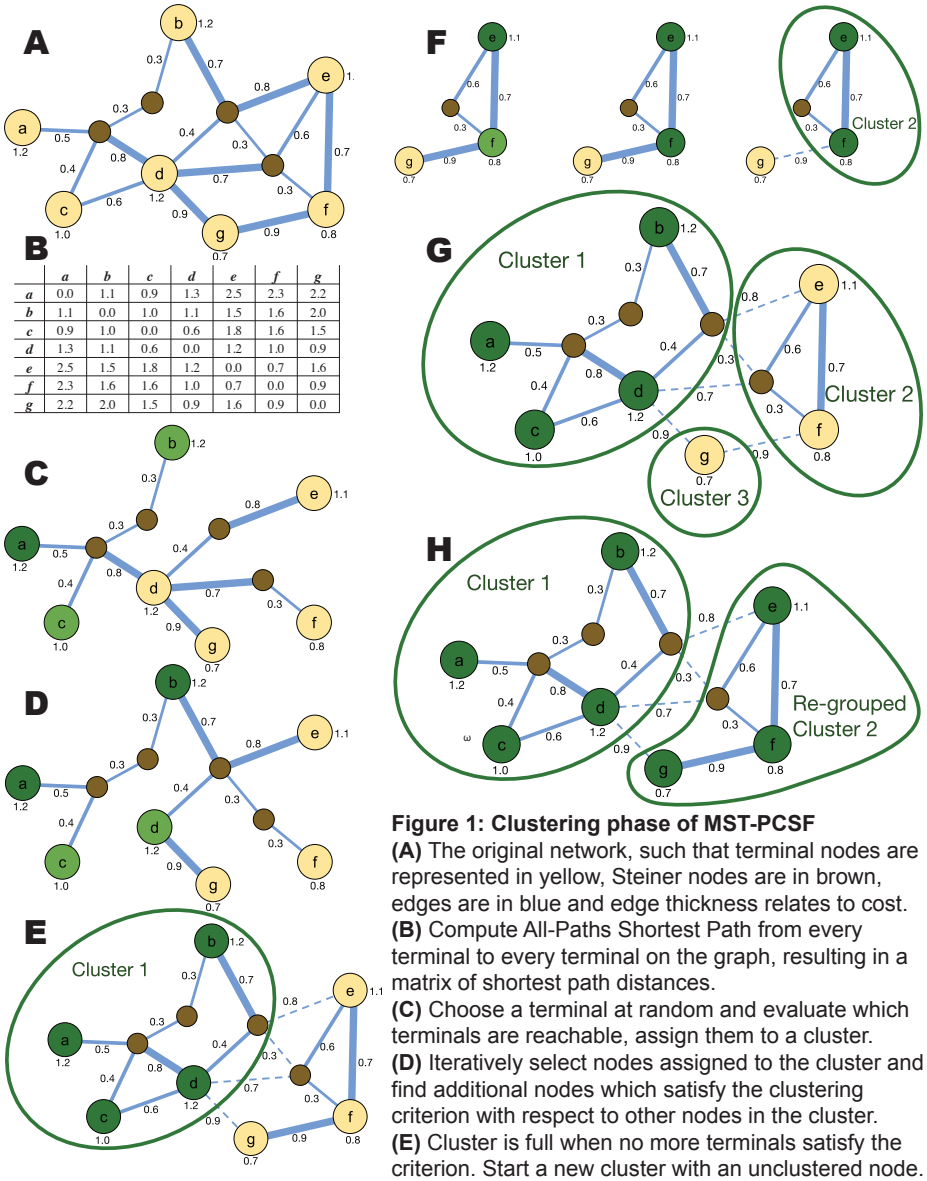
The proposed MST-PCSF heuristic algorithm is composed of two distinct phases. First, we cluster the input graph to transform a global problem into a set of smaller local problems. Second, we bypass the intractability of PCSF by instead solving the Minimum Spanning Tree Problem (MST) on an altered representation of the input graph. These simple ideas dramatically reduce the time needed to obtain high quality solutions to the Prize-collecting Steiner Forest Problem.

3.1 Greedy Clustering transforms a global problem into a set of local problems

The intuition behind the clustering phase of the algorithm is that, in the context of biological interaction networks, we anticipate finding within the expression data groups of dysregulated terminal nodes joined by moderately high confidence edges and Steiner nodes, forming independent high-prize, low-cost patches in the input graph. Simply put, we anticipate the clusters exist quite strongly in the data, and constituent trees from the optimal solution forest will be split between these clusters, but will not span multiple clusters. Clustering therefore is a sensible first step.

PCSF has proven more apt for the problem of highlighting disease-relevant networks than PCST because in a tumorous cell, for example, multiple functional pathways may be simultaneously active and dysregulated, but non-overlapping.

If we seek to find a tree, we are forced to incorporate spurious edges in our solution. If we seek to find a set of disconnected trees, however, we only select the edges needed to connect the high-prize terminals.



	a	b	c	d	e	f	g
a	0.0	1.1	0.9	1.3	2.5	2.3	2.2
b	1.1	0.0	1.0	1.1	1.5	1.6	2.0
c	0.9	1.0	0.0	0.6	1.8	1.6	1.5
d	1.3	1.1	0.6	0.0	1.2	1.0	0.9
e	2.5	1.5	1.8	1.2	0.0	0.7	1.6
f	2.3	1.6	1.6	1.0	0.7	0.0	0.9
g	2.2	2.0	1.5	0.9	1.6	0.9	0.0

Figure 1: Clustering phase of MST-PCSF
(A) The original network, such that terminal nodes are represented in yellow, Steiner nodes are in brown, edges are in blue and edge thickness relates to cost.
(B) Compute All-Paths Shortest Path from every terminal to every terminal on the graph, resulting in a matrix of shortest path distances.
(C) Choose a terminal at random and evaluate which terminals are reachable, assign them to a cluster.
(D) Iteratively select nodes assigned to the cluster and find additional nodes which satisfy the clustering criterion with respect to other nodes in the cluster.
(E) Cluster is full when no more terminals satisfy the criterion. Start a new cluster with an unclustered node.
(F) Repeat steps **B-E** until there are no remaining unclustered nodes from the graph in **A**.
(G) The raw clustering of the graph in **A** may have many singletons and doubletons.
(H) Merge singleton and doubleton clusters with their nearest cluster, return the final clustering

The clustering is performed by considering the pairwise relationship of terminal nodes. Two terminal nodes i and j are clustered together if they satisfy the strict clustering criterion: $D_{i,j} < p(i)$ & $D_{i,j} < p(j)$, which worked best of those we tested for biological networks. After clustering the graph, singleton and doubleton clusters are merged with their nearest neighbor clusters, since those subgraphs harbor very little biological information on their own.

Algorithm 1 MST-PCSF, Clustering phase

Initizalization:
Set $U = \{v : p(v) > 0\}$, the set of 'unassigned terminals'.
Set $A = \varnothing$, the set of 'assigned terminals'.
Vector C such that $|C| = |U|$ and $C_i \leftarrow 0$ for all $i : 1...U$
Matrix $D \in \mathbb{R}^{|U|\times|U|}$ such that $D_{i,j}$ is the weighted distance of the shortest path from v_i to v_j for all pairs $v_i, v_j \in U$
$clusterID \leftarrow 0$

Algorithm
while U is not $\varnothing$ **do**
  $clusterID \leftarrow clusterID$ +1;
  Remove arbitrary vertex v_i from U and insert it into A.
  while A is not $\varnothing$ **do**
    Remove arbitrary vertex v_a from A.
    $C_a = ClusterID$
    for each $v_u \in U$ **do**
      if ($D_{a,u} < p(v_a)$ & $D_{a,u} < p(v_u)$ **then**
        Remove v_u from U and insert it into A.
      end if
    end for
  end while
end while
for each singleton and doubleton cluster $G_k = (V_k, E_k)$ **do**
  Let $G_{min_k} = min \sum_{v_i \in V_k} \sum_{v_j \notin V_k} D_{ij}$, the closest subgraph to G_k.
  Consolidate G_k with the nearest cluster G_{min_k}
end for

Output terminals set A, assignment vector C representing the final clustering.

3.2 Solving MST on an altered representation of the input graph bypasses the complexity of PCSF

In the second phase of the algorithm, we bypass the complexity of PCSF by finding instead the tree covering every terminal node with minimum total edge costs, called the Minimum Spanning Tree. The altered representation of the original network is the set of complete subgraphs composed exclusively of terminals from each cluster, in which each edge is weighted with the shortest path distance between the terminal nodes. We can solve MST on this representation quickly and

then project the solution back into the original graph, and finally, disconnecting nodes too expensive to retain in the solution.

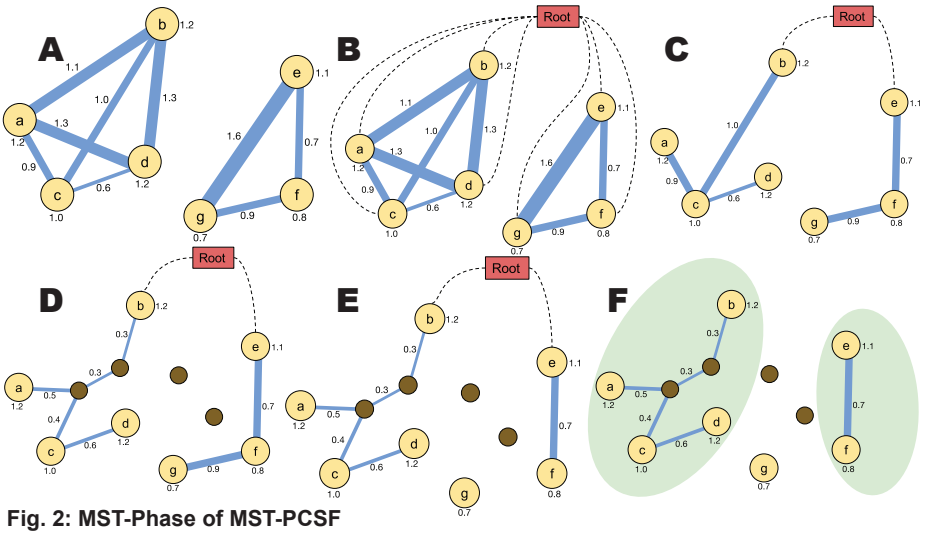


Fig. 2: MST-Phase of MST-PCSF
(A) Construct the complete graph from the terminal nodes in the clustered subgraphs
(B) Add an artificial root node with an edge from that root to every other nodes with cost ω. or use the already-existing node specified as the root by the user.
(C) Determine the MST of this augmented graph by Prim's algorithm.
(D) Interpolate the Steiner nodes from the original graph.
(E) Prune all leaves for which the prize of the leaf is less than the cost of the edge.
(F) Output a forest of subgraphs for further investigation.

Algorithm 2 MST-PCSF, MST phase

Initialization:
For each subgraph G_s, construct the complete subgraph of the terminal nodes in G_s called $G'_s{=}(V'_s, E'_s)$, such that each edge $\in E'_s$ is weighted with the cost of the shortest path between the terminals it connects in G_s.

Algorithm
Add a vertex called *Root* and edges from *Root* to each terminal with cost ω.
This produces a new graph which is the union of each of the G'_s subgraphs and *Root*.
Find the minimum spanning tree of the new graph to obtain $MST = (V_{mst}, E_{mst})$
Interpolate the omitted Steiner nodes in the edges of MST from G
for each leaf node $v \in MST$ and associated parent node v' **do**
  if $p(v) <$ connection cost **then**
    remove v from MST ;
  end if
end for
Remove *Root* from MST leaving behind a forest of subgraphs we call F.

Output F, an approximate solution to the PSCF problem on graph G.

The algorithmic steps in (Figure 2 A-D) can be repeated several times in order to decrease the tree cost further, by adding the Steiner nodes incorporated in the solution tree (Figure 2-D) into the cliques constructed in the next iteration (Figure 2-A), yielding better results at each iteration until convergence. However, we only perform a single iteration of those steps for our results in this work.

4 Results

We compare the performances of MST-PCSF and the message passing algorithm (MSGP), a broadly used heuristic algorithm for PCST and PCSF. The MSGP algorithm has been used to predict unknown protein associations [20], find hidden components of regulatory networks [22], and reconstruct cell-signaling pathways [21]. We test the performances of these algorithms on small benchmark instances from the literature, as well as medium and large networks generated from real biological data. We use the default parameters for MSGP, except the reinforcement parameter g, which is set to 0.001 as in [21]. The computational studies are performed on a server equipped with an AMD Opteron(tm) Processor 6320 and shared memory of 256 GB. A single core is used while running the experiments. We run the algorithms for values of $\omega = \{1, 2\}$ for medium and large instances, and we used $\omega = \{5, 8\}$ for small instances due to their higher edge costs.

The first of these domains is a set of benchmark instances for PCST algorithms from the literature [11] called the D instance set, which is composed of smaller networks of roughly 1,000 nodes and 25,000 edges. However, since the algorithms we compare solve PCSF, we modify the D instances by adding a *root* node beforehand, and specify they use that node as their root. For these small inputs, MSGP provides slightly higher upper bounds while MST-PCSF outperforms MSGP in average running time. (Table 1)

The second domain is phosphoproteomic data from the Glioblastoma patients in [21]. These graphs are obtained by using differentially expressed genes as terminals by mapping them onto the human PPI network. As our PPI, we use STRING (version13), in which the network edges have a experimentally-derived confidence score $s(e)$ [1]. The low confidence edges with $s(e) < 0.5$ are removed to improve the reliability of the findings. We convert edge confidence into edge cost: $c(e) = max(0.01, 1 - s(e))$. For these networks, the upper bounds of MST-PCSF and MSGP are comparable for both values of ω while MST-PCSF is an order of magnitude faster. (Table 3)

Our largest graphs are generated by mapping the phosphoproteomic data from the Breast Cancer patients in [23] onto an integrated interactome of proteins and metabolites, resulting in networks with 36,897 nodes and 1,016,288 edges. Here as well, each network represents a single patient (Table 2). For these large networks, MST-PCSF arrives at similar results in a tenth the time.

Table 1. Performances of MST-PCSF and MSGP for the D instances [21]. The performance of the message passing algorithm [20] and the proposed heuristic are displayed under $MSGP$ and $MST\text{-}PCSF$ for $\omega = \{5, 8\}$. We report the upper bounds obtained from the methods under OBJ column. The running times of the methods are provided in seconds under $t(s)$ column.

				$\omega = 5$				$\omega = 8$			
				MSGP		MST-PCSF		MSGP		MST-PCSF	
Instance	V	E	T	OBJ	t(s)	OBJ	t(s)	OBJ	t(s)	OBJ	t(s)
D01-A	1001	1255	5	21	0.44	21	0.04	26	0.44	26	0.04
D01-B	1001	1255	5	25	0.44	25	0.04	40	0.44	40	0.03
D02-A	1001	1260	10	46	0.44	46	0.05	58	0.44	58	0.04
D02-B	1001	1260	10	50	0.44	50	0.05	80	0.44	80	0.04
D03-A	1001	1417	83	630	0.65	631	0.53	787	0.70	795	0.54
D03-B	1001	1417	83	782	0.66	782	0.56	1131	0.67	1156	0.53
D04-A	1001	1500	250	928	0.64	942	0.86	1164	0.63	1174	0.80
D04-B	1001	1500	250	1129	0.66	1137	0.85	1574	0.83	1591	0.84
D05-A	1001	1750	500	1777	0.85	1791	1.90	2130	0.84	2191	1.84
D05-B	1001	1750	500	2108	0.78	2125	2.09	2787	0.82	2832	2.12
D06-A	1001	2005	5	21	0.61	21	0.04	26	0.62	26	0.06
D06-B	1001	2005	5	25	0.60	25	0.04	40	0.62	40	0.05
D07-A	1001	2010	10	46	0.67	46	0.05	58	0.68	58	0.08
D07-B	1001	2010	10	50	0.67	50	0.11	80	0.68	80	0.06
D08-A	1001	2167	83	630	0.82	640	0.64	763	0.85	789	0.62
D08-B	1001	2167	83	753	0.84	759	0.66	994	1.56	1012	0.66
D09-A	1001	2250	250	918	1.73	927	0.98	1080	0.95	1117	0.94
D09-B	1001	2250	250	1099	1.29	1119	1.03	1367	1.03	1403	1.03
D10-A	1001	2500	500	1569	1.21	1598	2.36	1705	1.37	1742	2.36
D10-B	1001	2500	500	1812	1.13	1839	2.44	2058	1.25	2074	2.43
D11-A	1001	5005	5	21	1.53	21	0.08	26	4.11	26	0.07
D11-B	1001	5005	5	24	1.57	24	0.07	36	4.12	36	0.08
D12-A	1001	5010	10	42	1.61	42	0.10	50	3.66	50	0.12
D12-B	1001	5010	10	43	1.56	44	0.11	50	2.57	52	0.10
D13-A	1001	5167	83	454	4.72	473	1.13	463	2.48	476	1.09
D13-B	1001	5167	83	501	2.46	514	1.09	510	3.11	530	1.10
D14-A	1001	5250	250	615	3.32	640	1.68	624	3.89	655	1.69
D14-B	1001	5250	250	675	4.16	698	1.67	690	2.10	711	1.69
D15-A	1001	5500	500	1057	3.72	1069	3.68	1076	2.43	1083	3.73
D15-B	1001	5500	500	1124	2.69	1144	3.65	1147	2.73	1156	3.74
D16-A	1001	25005	5	18	30.12	19	0.28	22	22.92	22	0.28
D16-B	1001	25005	5	19	22.28	21	0.27	22	24.87	24	0.27
D17-A	1001	25010	10	28	27.29	30	0.40	31	20.27	33	0.39
D17-B	1001	25010	10	28	28.28	30	0.40	31	30.67	33	0.43
D18-A	1001	25167	83	225	16.16	249	3.96	231	27.75	253	3.96
D18-B	1001	25167	83	233	19.73	257	3.93	238	22.58	260	3.95
D19-A	1001	25250	250	314	16.42	342	5.88	321	32.38	345	5.87
D19-B	1001	25250	250	320	29.35	350	5.88	323	29.35	353	5.91
D20-A	1001	25500	500	542	29.84	546	11.97	545	26.37	549	12.00
D20-B	1001	25500	500	543	19.81	547	11.99	545	32.22	550	11.95
			mean	531	7.05	541	1.84	623	7.91	637	1.84
			std	573	10.28	579	2.83	698	11.36	709	2.83

Table 2. The comparison results of the methods for the Breast Cancer network instances generated based on phosphoproteomic data in [23]. The performance of the message passing algorithm [20] and the proposed heuristic are displayed under $MSGP$ and $MST\text{-}PCSF$ for $\omega = \{1, 2\}$.

				$\omega = 1$				$\omega = 2$			
				MSGP		MST-PCSF		MSGP		MST-PCSF	
Instance	V	E	T	OBJ	t(s)	OBJ	t(s)	OBJ	t(s)	OBJ	t(s)
A2-A0CM	36892	1016411	122	35.71	1668	35.64	148	36.69	1221	36.89	131
A2-A0D2	36892	1016411	226	66.38	1653	66.50	260	67.53	2067	67.50	235
A2-A0EV	36892	1016411	69	18.29	1329	18.35	86	19.55	1780	19.60	78
A2-A0EY	36892	1016411	118	40.56	1962	40.70	158	41.77	2355	41.91	126
A2-A0SW	36892	1016411	60	14.45	1622	14.50	83	15.45	1550	15.50	68
A2-A0T6	36892	1016411	92	30.92	1615	31.05	119	32.18	1633	32.31	100
A2-A0YC	36892	1016411	182	48.04	1576	48.17	236	49.04	1844	49.17	195
A2-A0YD	36892	1016411	55	17.92	1573	17.92	78	18.92	1787	18.92	69
A2-A0YF	36892	1016411	165	48.31	2003	48.27	212	49.54	984	49.65	185
A2-A0YM	36892	1016411	236	61.89	1912	62.07	304	62.91	1560	63.19	244
A7-A0CE	36892	1016411	142	38.35	1486	38.42	176	39.61	1772	39.68	150
A7-A13F	36892	1016411	139	43.42	1365	43.57	181	44.42	1515	44.57	147
A8-A06Z	36892	1016411	112	36.24	1462	36.42	141	37.76	1456	37.94	120
A8-A079	36892	1016411	77	25.77	1599	25.96	93	26.77	2029	26.96	94
A8-A09G	36892	1016411	186	46.65	1882	46.61	222	47.61	2126	47.73	214
AN-A04A	36892	1016411	208	63.40	1828	63.41	254	64.58	1940	64.41	241
AN-A0FK	36892	1016411	81	21.82	1677	21.89	104	22.99	1535	23.06	98
AN-A0FL	36892	1016411	126	35.63	1891	35.87	157	36.75	1759	36.99	134
AO-A0JC	36892	1016411	76	22.32	1661	22.39	94	23.32	1844	23.39	84
AO-A0JE	36892	1016411	116	31.60	2214	31.70	151	32.88	1846	32.97	124
AO-A0JM	36892	1016411	216	59.34	2334	59.60	261	60.79	1971	61.09	222
AO-A126	36892	1016411	51	16.26	1595	16.30	72	17.26	1519	17.30	59
AO-A12B	36892	1016411	148	43.94	2091	43.94	193	45.88	1655	45.72	156
AO-A12D	36892	1016411	211	54.94	1492	55.09	265	56.00	1935	56.18	217
AO-A12E	36892	1016411	146	39.68	1943	39.78	183	40.98	2246	40.81	154
AO-A12F	36892	1016411	167	44.88	2282	45.16	228	46.02	2301	46.30	175
AR-A0TR	36892	1016411	120	32.08	1756	32.28	154	33.08	1830	33.28	128
AR-A0TT	36892	1016411	164	43.05	1872	43.33	223	44.22	1636	44.50	191
AR-A0TV	36892	1016411	201	57.08	2323	57.27	276	58.20	3149	58.44	208
AR-A0TX	36892	1016411	169	46.36	1929	46.48	225	47.73	2088	47.74	176
AR-A0U4	36892	1016411	154	41.16	2093	41.39	210	42.48	2044	42.70	162
AR-A1AP	36892	1016411	109	32.49	1569	32.50	152	33.52	1584	33.54	117
AR-A1AS	36892	1016411	181	48.69	2111	48.91	246	49.93	2301	50.09	186
AR-A1AW	36892	1016411	116	31.48	1494	31.72	143	32.65	1831	32.89	124
BH-A0AV	36892	1016411	137	40.06	1918	40.20	163	41.84	1653	41.98	143
BH-A0BV	36892	1016411	220	57.43	1674	57.58	251	58.52	1641	58.58	228
BH-A0DG	36892	1016411	209	62.49	1707	62.58	278	63.56	1924	63.70	217
BH-A0E9	36892	1016411	107	35.57	1907	35.45	153	36.35	1412	36.45	115
BH-A18N	36892	1016411	176	47.57	1567	47.58	243	48.73	2306	48.87	184
BH-A18U	36892	1016411	72	23.87	1507	23.91	94	24.87	1281	24.91	80
C8-A12T	36892	1016411	87	20.25	1604	20.31	105	21.51	1735	21.57	95
C8-A12U	36892	1016411	87	27.03	1927	27.15	104	28.02	1598	28.15	95
C8-A12V	36892	1016411	94	26.38	2159	26.54	111	27.38	1958	27.54	101
C8-A130	36892	1016411	84	26.88	1078	27.06	124	27.88	1225	28.06	92
C8-A131	36892	1016411	187	53.48	1762	53.77	222	54.57	2113	54.80	195
C8-A138	36892	1016411	191	51.18	1939	51.14	235	52.13	1675	52.19	212
D8-A142	36892	1016411	135	41.35	1437	41.20	169	42.03	1051	42.20	143
E2-A154	36892	1016411	145	42.36	1773	42.50	174	43.47	1810	43.61	153
E2-A158	36892	1016411	169	49.12	2276	49.18	209	50.37	2372	50.38	177
			mean	39.91	1768	40.02	179	41.06	1801	41.17	150
			std	13.54	288	13.56	63	13.57	379	13.58	52

Table 3. Performances of MST-PCSF and MSGP for the Glioblastoma network instances generated from phosphoproteomic data from [21]. The first four columns provide instance name, number of total nodes, edges, and terminals for each network. The performance of the message passing algorithm [20] and the proposed heuristic are displayed under $MSGP$ and $MST\text{-}PCSF$ for $\omega = \{1, 2\}$. We report the upper bounds obtained from the methods under OBJ column. The running times of the methods are provided in seconds under $t(s)$ column.

				$\omega = 1$				$\omega = 2$			
				MSGP		MST-PCSF		MSGP		MST-PCSF	
Instance	V	E	T	OBJ	t(s)	OBJ	t(s)	OBJ	t(s)	OBJ	t(s)
GBM6	15357	175792	108	26.35	338	26.39	22	28.61	262	28.65	22
GBM8	15357	175792	115	25.45	233	25.48	23	27.45	204	27.48	23
GBM10	15357	175792	132	32.85	229	32.91	26	36.96	270	37.02	26
GBM12	15357	175792	135	32.50	228	32.57	27	35.03	234	35.10	27
GBM15	15357	175792	131	30.49	326	30.56	26	33.62	247	33.70	26
GBM26	15357	175792	122	28.42	270	28.46	24	31.02	237	31.04	24
GBM39	15357	175792	123	29.51	166	29.54	25	31.68	163	31.70	25
GBM59	15357	175792	123	26.54	249	26.53	24	29.56	274	29.55	25
GBM-pool	15357	175792	161	42.90	324	42.98	32	50.90	272	50.97	32
			mean	30.56	263	30.60	26	33.87	240	33.91	26
			std	5.33	57	5.35	3	7.08	37	7.10	3

Finally, we compare the results provided by MST-PCSF and MSGP in Glioblastoma patient networks for $\omega = 1$. We excluded the UBC gene from the interactome due to its high node degree. We took the union of output forests for these 9 instances for each algorithm. MST-PCSF provided a subgraph with 269 nodes and 301 edges and MSGP provided a subgraph of 286 nodes and 364 edges. We merged the subgraphs into one network, demonstrating the overlap between the solutions (Figure 3). The methods obtained solutions with 255 common nodes and 251 common edges. MST-PCSF recovered 89% of nodes and 83% of edges of the solution provided by MSGP. This result demonstrates that our algorithm does not merely recover similar quality solutions, but in fact, very similar solutions.

5 Conclusion

We present a new heuristic algorithm for the Prize-collecting Steiner Forest problem which supersedes the existing algorithms of which we are aware, particularly on larger-scale graphs common in the application-space of biology. The PCSF approach is well suited to the problem of predicting disease-relevant subnetworks from an interactome conditional on observed data gathered from patients. Our algorithm reduces the requisite computing time to solve PCSF which expedites existing research and also provides the capacity to analyze these data patient-by-patient, an operation which previously has been prohibitively computationally expensive.

Our algorithm accelerates the pace of relevant subnetwork imputation, which we hope will be a boon for all who apply the Prize-collecting Steiner Forest approach, in biology and elsewhere.

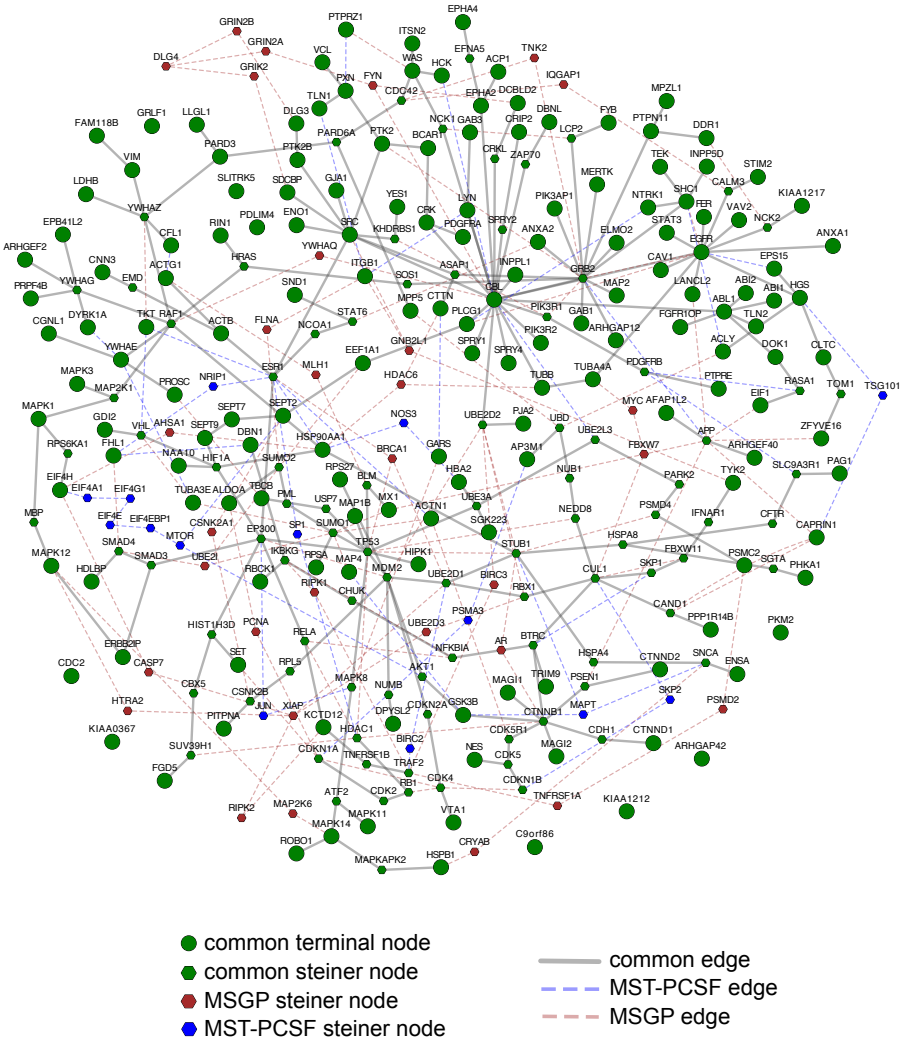


Figure 3: A visual representation of the relationship between the solutions obtained by the MST-PCSF and MSGP algorithms on Gliablastoma patient data.

6 Acknowledgments

This work was supported by the Swiss National Science Foundation through the project 205321-147138/1: Steiner trees for functional analysis in cancer system

biology, and the National Institute of Health (U54-NS-091046 and U01-CA-184898).

References

1. D. Szklarczyk, A. Franceschini, M. Kuhn, M. Simonovic, A. Roth, P. Minguez, T. Doerks, M. Stark, J. Muller, P. Bork, L. J. Jensen and C. v. Mering. STRING 8 - a global view on proteins and their functional interactions in 630 organisms. *Nucleic Acids Res* 37: D412 - D416, 2011.
2. D. Bienstock, M. X. Goemans, D. Simchi-Levi, and D. Williamson. A note on the prize-collecting traveling salesman problem. *Mathematical Programming*, 59:413 - 420, 1993.
3. M. X. Goemans and D. P. Williamson. The primal-dual method for approximation algorithms and its application to network design problems. *Approximation algorithms for NP-hard problems*, 144191, 1996.
4. D. S. Johnson, M. Minkoff, and S. Phillips. The prize- collecting Steiner tree problem: Theory and practice. In Proceedings of 11^{th} *ACM-SIAM Symposium on Discrete Algorithms*, 760-769, 2000.
5. P. Feofiloff, C.G. Fernandes, C.E. Ferreira, and J.C. Pina. Primal-dual approximation algorithms for the prize-collecting Steiner tree problem. *Information Processing Letters*, 103(5), 195-202, 2007.
6. I. Ljubic, R. Weiskircher, U. Pferschy, G. Klau, P. Mutzel, and M. Fischetti. Solving the prize-collecting Steiner tree problem to optimality. *Seventh Workshop on Algorithm Engineering and Experiments*, 6876, 2005
7. I. Ljubic, R. Weiskircher, U. Pferschy, G. Klau, P. Mutzel, and M. Fischetti. An algorithmic framework for the exact solution of the prize-collecting Steiner tree problem. *Math Progam*, 105(2):42749, 2006.
8. M. B. Bechet, S. Bradde, A. Braunstein, A. Flaxman, L. Foini, and R. Zecchina. Clustering with shallow trees. *J. Stat. Mech.*, 12010, 2009.
9. M. Haouari, S. B. Layeb, and H. D. Sherali. Algorithmic expedients for the Prize Collecting Steiner Tree Problem. *Discrete Optimization*, 7:3247, 2010.
10. A. S. Cunha, A. Lucena, N. Maculan, M. G.C. Resende. A relax-and-cut algorithm for the prize-collecting Steiner problem in graphs. *Discrete Applied Mathematics*, 157:1198-1217, 2009.
11. S. A. Canuto, M. G. C. Resende, and C. C. Ribeiro. Local search with perturbation for the Prize-collecting Steiner tree problem in graphs. *Networks*, 38:5058, 2001.
12. O. Chapovska, and A. P. Punnen. Variations of the Prize-Collecting Steiner Tree Problem. *Networks*, 47(4):199205, 2006.
13. J. E. Beasley. An SST-based algorithm for the Steiner problem in graphs. *Networks*, 19:1-16, 1989.
14. C. W. Duin and A. Volgenant. Some generalizations of the Steiner problem in graphs. *Networks*, 17(2):353364, 1987.
15. M. Fischetti. Facets of two Steiner arborescence polyhedra. *Mathematical Programming*, 51:401-419, 1991.
16. A. Lucena, and M. G. C. Resende. Strong lower bounds for the Prize-collecting Steiner problem in graphs. *Discrete Applied Mathematics*, 141:277-294, 2004.
17. Z. H. Fu, and J. K. Hao. Knowledge Guided Tabu Search for the Prize Collecting Steiner Tree Problem in Graphs. 11^{th} *DIMACS challenge workshop*, 2014.

18. G.W. Klau, I. Ljubic, A. Moser, P. Mutzel, P. Neuner, U. Pferschy, and R. Weiskircher. Combining a memetic algorithm with integer programming to solve the prize- collecting Steiner tree problem. *Genetic and Evolutionary Computation Conference*, 3102:13041315, 2004.
19. M. T. Dittrich, G. W. Klau, A. Rosenwald, T. Dandekar, and T. Mueller. Identifying functional modules in proteinprotein interaction networks: an integrated exact approach. *Bioinformatics*, 26:22331, 2008.
20. M. B. Bechet, C. Borgs, A. Braunstein, J. Chayesb, A. Dagkessamanskaia, J.M. Franois, and Zecchina R. Finding undetected protein associations in cell signaling by belief propagation. *PNAS*, 108:882887,2010.
21. N. Tuncbag, A. Braunstein, A. Pagnani, S. C. Huang, J. Chayes, C. Borgs, R. Zecchina and E. Fraenkel. Simultaneous Reconstruction of Multiple Signaling Pathways via the Prize-Collecting Steiner Forest Problem. *Journal of Computational Biology*, 20(2):124-136, 2013.
22. N. Tuncbag, S. McCallum, S. C. Huang, and E. Fraenkel. SteinerNet: a web server for integrating omic data to discover hidden components of response pathways. *Nucleic Acids Research*, 1-5, 2012.
23. P. Mertins, D. R. Mani, K. V. Ruggles, M. A. Gillette, K. R. Clauser, P. Wang, X. Wang, J. W. Qiao, S. Cao, F. Petralia, E. Kawaler, F. Mundt, K. Krug, Z. Tu, J. T. Lei, M. L. Gatza, M. Wilkerson, C. M. Perou, V. Yellapantula, K. Huang, C. Lin, M. D. McLellan, P. Yan, S. R. Davies, R. R. Townsend, S. J. Skates, J. Wang, B. Zhang, C. R. Kinsinger, M. Mesri, H. Rodriguez, L. Ding, A. G. Paulovich, D. Feny, M. J. Ellis, S. A. Carr & NCI CPTAC. Proteogenomics connects somatic mutations to signalling in breast cancer. *Nature*, 534:5562, June 2016.
24. M. Akhmedov, I. Kwee, and R. Montemanni. A divide and conquer matheuristic algorithm for the Prize-collecting Steiner Tree Problem. *Computers and Operations Research*, 70:18-25, 2016.
25. M. Akhmedov, I. Kwee, and R. Montemanni. A fast heuristic for the prize-collecting Steiner tree problem. *Lecture Notes in Management Science*, 6:207-216, 2016.
26. R.C. Prim. Shortest connection networks and some generalizations. *Bell System Technical Journal*, 13891401, 1957.