Appendix. Online Supplement

A. Detailed Results of IRMS under the Time Limit $\hat{t} = 7200$ Seconds

Table 1 summarizes the detailed results of IRMS under a long time limit $\hat{t} = 3600$ seconds. In Table 1, columns 1-3 describe for each instance its name (Instance), K value, and best known value (BKV) reported in the literature, respectively. Columns 4-6 report the results of IRMS, including the best result (i.e., $\hat{f}$) found during 30 runs, the average result (i.e., $\bar{f}$), and average computation time (i.e., $\bar{t}$) at each run. From it, we observe that IRMS also performs excellent performance. In particular, it finds new upper bounds for nine instances, and matches previous best-known upper bounds on 23 instances.

Table 1 **Results of IRMS on Synthetic and Real-world Benchmarks under** $\hat{t} = 7200$ **Seconds**

			IRMS		
Instance	K	BKV	$\hat{f}$	$\bar{f}$	$\bar{t}$
BA500	50	195*	**195**	195.0	0.0
BA1000	75	558*	**558**	558.0	1.5
BA2500	100	3704*	**3704**	3704.0	4.5
BA5000	150	10196*	**10196**	10196.0	18.8
ER235	50	295	**295**	295.0	7.9
ER466	80	1524	**1524**	1524.0	79.2
ER941	140	5012	**5012**	5023.3	1817.8
ER2344	200	902498	918952	941170.2	5673.8
FF250	50	194*	**194**	194.0	0.0
FF500	110	257*	**257**	257.0	0.7
FF1000	150	1260*	**1260**	1260.0	18.5
FF2000	200	4545*	**4545**	4545.0	264.2
WS250	70	3083	**3083**	3132.7	4127.2
WS500	125	2072	**2072**	2078.1	813.2
WS1000	200	109677	137766	144237.3	3149.4
WS1500	265	13098	**13098**	13103.4	2461.6
Bovine	3	268	**268**	268.0	0.0
Circuit	25	2099	**2099**	2099.0	1.3
Ecoli	15	806	**806**	806.0	0.0
USAir97	33	4336	**4336**	4596.0	1452.8
HumanDi	52	1115	**1115**	1115.0	1.5
TreniR	26	918	**918**	918.0	1.3
EU_fli	119	348268	**348268**	348268.0	749.4
openfli	186	26783	26875	28363.8	4400.1
yeast1	202	1412	**1412**	1412.0	34.0
H1000	100	306349	**306349**	308345.0	3561.5
H2000	200	1242739	**1236887**⋆	1250761.8	5352.7
H3000a	300	2840690	**2799868**⋆	2840491.0	6055.2
H3000b	300	2837584	**2794262**⋆	2821455.9	6184.0
H3000c	300	2835369	**2783248**⋆	2825113.4	6682.2
H3000d	300	2828492	**2802615**⋆	2838392.3	5612.8
H3000e	300	2843000	**2798688**⋆	2841275.9	5836.7
H4000	400	5038611	**4977344**⋆	5074711.6	5894.7
H5000	500	7964765	**7956481**⋆	8058861.6	6062.2
powergr	494	15862	15863	15870.8	5764.8
Oclinks	190	611253	614467	614467.0	1494.1
faceboo	404	420334	705403	723233.5	5915.9
grqc	524	13591	**13590**⋆	13598.0	5945.4
hepth	988	106276	110352	114505.1	6367.1
hepph	1201	6155877	9309386	9543464.4	5009.6
astroph	1877	53963375	56849750	57688316.6	5149.1
condmat	2313	2298596	9359852	10415923.6	4155.8

* Optimal results obtained by branch-and-cut algorithm within 5 days.
⋆ Improved best upper bounds.

B. Detailed Results of CEMCNP under the Time Limit $\hat{t} = 3600$ Seconds

Since the source code of CEMCNP is not available to us, we have re-implemented it according its pseudo code. Detailed comparison between our implemented CEMCNP algorithm and reported CEMCNP algorithm are summarized in Table 2. From it, we observe that our implemented CEMCNP algorithm performs slightly worse performance than the reported results on most of instances. However, some reported results are obtained under a longer computation time than the time limit. For example, at least 4000 seconds are required to achieve the results of H3000e, hepth, hepph and condmat. It is worthy noting that for some instances (e.g, grqc, hepth and condmat), our implemented CEMCNP algorithm achieves better performance than the original version.

Table 2 Detailed Results of CEMCNP on Synthetic and Real-world Benchmarks under $\hat{t} = 3600$ Seconds

			Reported Results		Implemented Results		
Instance	K	BKV	$\hat{f}$	$\bar{t}$	$\hat{f}$	$\overline{f}$	$\bar{t}$
BA500	50	195	**195**	0.0	**195**	195.0	0.0
BA1000	75	558	**558**	0.2	**558**	558.0	54.0
BA2500	100	3704	**3704**	0.2	**3704**	3704.0	347.3
BA5000	150	10196	**10196**	1.8	**10196**	10196.0	9.6
ER235	50	295	**295**	19.8	297	302.8	0.0
ER466	80	1524	**1524**	98.4	1569	1630.6	1.7
ER941	140	5012	**5012**	694.6	5363	5635.3	2.7
ER2344	200	902498	**912346**	3069.0	1012527	1060618.6	847.9
FF250	50	194	**194**	0.3	**194**	194.0	178.2
FF500	110	257	**257**	0.2	**257**	258.6	0.3
FF1000	150	1260	**1260**	120.6	**1260**	1260.0	1080.9
FF2000	200	4545	**4545**	486.4	4546	4552.5	1458.1
WS250	70	3083	**3083**	623.1	4203°	5447.9	0.5
WS500	125	2072	**2072**	105.8	2085	2193.0	153.7
WS1000	200	109677	**109935**	1256.2	154899	169877.2	19.3
WS1500	265	13098	**13098**	658.2	13664	27810.6	108.0
Bovine	3	268	**268**	0.0	**268**	268.0	0.0
Circuit	25	2099	**2099**	0.1	2101	2188.7	0.1
Ecoli	15	806	**806**	0.0	**806**	808.8	0.1
USAir97	33	4336	**4336**	856.4	**4336**	5149.5	0.1
HumanDi	52	1115	**1115**	0.5	**1115**	1115.0	319.9
TreniR	26	918	**918**	0.2	**918**	918.0	10.9
EU_fli	119	348268	**348325**	431.6	350762	357502.7	89.5
openfli	186	26783	**26796**	3165.8	29481	31377.7	86.4
yeast1	202	1412	**1412**	9.8	**1412**	1414.3	1085.8
H1000	100	306349	**307113**	1068.4	330493	337217.0	211.4
H2000	200	1242739	**1245637**	1527.2	1324988	1344914.2	126.4
H3000a	300	2840690	**2842695**	1204.3	2962661	3042695.4	165.2
H3000b	300	2837584	**2840867**	2040.0	2968601	3035272.7	438.0
H3000c	300	2835369	**2831643**	2159.3	2956916	3008793.3	903.6
H3000d	300	2828492	**2830284**	3895.2°	2967747	3030236.3	51.6
H3000e	300	2843000	**2846536**	4035.6°	3012046	3043889.5	25.9
H4000	400	5038611	**5096437**	1563.7	5261850	5383301.3	347.5
H5000	500	7964765	**8007638**	2024.1	8206499	8348589.5	696.5
powergr	494	15862	**15906**	856.7	15965	16084.9	3393.5
Oclinks	190	611253	**615467**	267.8	622237	626701.5	836.1
faceboo	404	420334	**589763**	3526.9	794938	857245.1	685.4
grqc	524	13591	13743	1025.0	**13666**	13701.7	3391.1
hepth	988	106276	115309	4521.4°	**109504**	111495.6	1602.3
hepph	1201	6155877	**7556094**	4025.1°	8464199	10530358.9	3513.9
astroph	1877	53963375	**57895042**	3105.2	59476077	60449655.3	3512.9
condmat	2313	2298596	7658643	4203.5°	**3756761**	4632078.6	3555.9

° indicates a longer computation time than 3600 seconds.

C. TarjanInComponent Procedure

Algorithm 1 realizes the TarjanInComponent procedure used in the articulation point impact strategy. It starts the search from a root node, and recursively builds a depth first search (DFS) tree. During DFS phase, all articulation points are identified. Once all nodes are visited, the evaluation of all nodes is finished.

Algorithm 1 Pseudo Code of TarjanInComponent Procedure

Input: A large connected component $\mathcal{C}$, root node x, time stamp $Count$, γ and η

$dfn[x] \leftarrow ++Count$
$low[x] \leftarrow dfn[x]$
for each neighbour w of x in $\mathcal{C}$ **do**
  if w has not been visited **then**
    TarjanInComponent$(\mathcal{C}, w, Count, \gamma, \eta)$
    $low[x] \leftarrow \min\{low[x], low[w]\}$
    // w is not the parent of v
    if $dfn[x] < dfn[w]$ **then**
      $\gamma[x] += \gamma[w]$
    end if
    if $dfn[x] < low[w]$ **then**
      The number of x's subtrees increases one
      if x is not the root node **then**
        x is marked as an articulation point
        $\eta[x] += \gamma[w]$
        $\psi[x] += \frac{\gamma[w](\gamma[w]-1)}{2}$
      else
        if x is the root $\wedge$ x has more than one subtree **then**
          x is marked as an articulation point
        end if
      end if
    end if
  else
    $low[x] \leftarrow \min\{low[x], dfn[w]\}$
  end if
end for