Network Robustness to BDM-Directed Sequential and Simultaneous Attack

June 2018

Author : Antonio Rueda - Toicen
antonio.rueda.toicen@{ algoritmicnaturelab.org, gmail.com}

One can direct attacks to the elements (nodes/edges) in a network according to a “*centrality*” metric that assigns a relative important to each. In a “sequential attack” that erases *i* elements, after an element is removed the centrality metric is re-evaluated to take into account the change in the network’s structure, whereas in a “simultaneous” attack the centrality metric of every element is evaluated once, prior to any deletion [Iyer13]. The Block Decomposition Method (BDM) can be used to assign centrality scores to vertices or edges according to their information contribution [Zenil14, Zenil16, Zenil18a, Zenil18b].

Robustness to directed attacks of vertices in a connected graph can be quantified according to the R-index

$$R = \frac{1}{N}\sum_{i=1}^{N} \sigma(i/N)$$

Being N the number of vertices in the network and $\sigma(i/N)$ the relative size of the network after i vertices have been removed.

For any type of attack to remove vertices from any network, R attains its minimum value of $1/N$ on the star graph and its maximum value of $(1/2)\,(1 - 1/N)$ on the complete graph.Thus, for any network and method of vertex removal, $R \in [0, 1/2]$ [ref] .

The Block Decomposition Method is implemented using the Coding Theorem Method data and code from "Kolmogorov Complexity of 3 × 3 and 4 × 4 Squares" [Soler13]

In[171]:=
```
Clear["Global`*"]
```

In[172]:=
```
D2DRepres = {{{{0,0,0},{0,0,0},{0,0,0}},13.713356989265955`},{{{0,0,0},{0,0,0},{0,0,1}},14.91449
complement1[stringList_]:= 1- stringList;
ApplyTransformation[function_,data_]:=Join[data,function/@data];
get16Symmetric[list_]:=ApplyTransformation[complement1,ApplyTransformation[Reverse/@#&,ApplyTrans
getRepresentative[list_]:=First@Union@get16Symmetric[list];
Table[SquaresHash[D2DRepres[[i,1]]] = D2DRepres[[i,2]],{i,Length@D2DRepres}];
complexitySquare[list_]:= SquaresHash[getRepresentative@list];
square[n_,dim_]:=Partition[IntegerDigits[n,2,dim^2],dim];
getPos[n_,s_,{row_,col_}]:= If[(row > s) ||(col> s), 0, IntegerDigits[n,2,s^2][[(row-1)s+col]]]
```

In[181]:=
```
SquareBDM[matrix_, size_, offset_] := Module[{
 squares = Partition[matrix, {size,size}, offset],
  count
 },
  count = Tally[Flatten[squares, 1]];
   Total[complexitySquare[#]& /@ (First /@ count) + Log2[(Last /@ count)]]
]
```

In[182]:=
```
getAdjMatrixPostVertexDeletion[graph_, vertex_]:= Normal[AdjacencyMatrix[VertexDelete[graph, vert
```

In[183]:=
```
getAdjMatrixPostEdgeDeletion[graph_, edge_]:= Normal[AdjacencyMatrix[EdgeDelete[graph, edge]]]
```

In[184]:=
```
simultaneaousAttackOnEdges[graph_, nEdges_]:=
EdgeDelete[graph, (First /@ getEdgeAttackComplexityRank[graph]) [[1;;nEdges]]]
```

In[185]:=
```
getEdgeAttackComplexityRank[graph_]:= Reverse[SortBy[With[{origBDM = SquareBDM[AdjacencyMatrix[gr
```

In[186]:=
```
getVertexAttackComplexityRank[graph_]:= Reverse[SortBy[With[{origBDM = SquareBDM[AdjacencyMatrix[
```

In[187]:=
```
seqBDMAttackOnVertices[graph_, nVertices_] :=Nest[VertexDelete[#,First@First@getVertexAttackCompl
```

In[188]:=
```
simultaneousBDMAttackOnVertices[graph_, nVertices_]:=
VertexDelete[graph, (First /@ getVertexAttackComplexityRank[graph]) [[1;;nVertices]]]
```

In[189]:=
```
robustnessSimAttackBDM[g_]:=  (Table[Max[Length /@ WeaklyConnectedComponents[VertexDelete[g, (Fir
```

In[190]:=
```
robustnessSimAttack[g_, centrality_]:=  (Table[Max[Length /@ WeaklyConnectedComponents[VertexDele
```

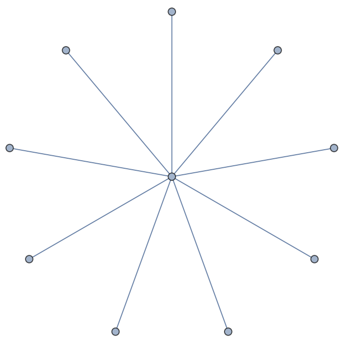
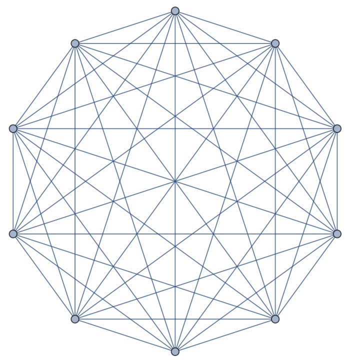
In[191]:=
```
robustnessSeqAttack[graph_, centrality_] :=
(1/VertexCount[graph])Total[(1 / VertexCount[graph])*Map[Max,Map[Length, ConnectedComponents /@D
```

In[192]:=
```
robustnessSeqAttackBDM[graph_] :=
(1/VertexCount[graph])Total[(1 / VertexCount[graph])*Map[Max,Map[Length, WeaklyConnectedComponen
```

Exploration with Star and Complete Graphs

The star graph and the complete graph represent the lowest and highest extremes of robustness in connected graphs, respectively.

In[193]:=
```
{StarGraph[10], CompleteGraph[10]}
```

Out[193]= {  ,  }

Expected vs computed robustness in star and complete graph

In[194]:=
```
{N@(1/100), robustnessSeqAttackBDM[StarGraph[100]]}
```

Out[194]= {0.01, 0.00990}

In[195]:=
```
{.5 * (1 - 1/100), robustnessSeqAttackBDM[CompleteGraph[100]]}
```

Out[195]= {0.495, 0.495}

In[196]:=
```
SquareBDM[AdjacencyMatrix[StarGraph[5]], 4, 1]
```

Out[196]= 105.527

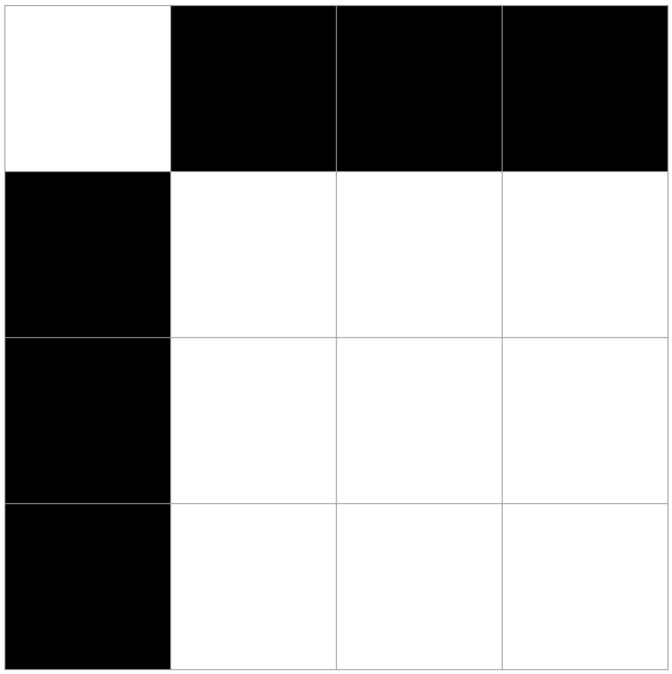
In[197]:=
```
getVertexAttackComplexityRank[StarGraph[5]]
```

Out[197]= {{1, 83.5206}, {5, 75.9639}, {4, 75.9639}, {3, 75.9639}, {2, 75.9639}}

In[198]:=
```
getEdgeAttackComplexityRank[StarGraph[5]]
```

Out[198]= {{1 ⟷ 2, 2.98977}, {1 ⟷ 4, 2.71213}, {1 ⟷ 3, 1.18206}, {1 ⟷ 5, -0.571505}}

In[204]:=
```
AdjacencyMatrix[VertexDelete[StarGraph[5], 2]] // ArrayPlot[#, Mesh → True] &
```

Out[204]=


In[199]:=
```
origBDM = SquareBDM[AdjacencyMatrix[StarGraph[5]], 4, 1]
```

Out[199]= 105.527

In[222]:=
```
Table[SquareBDM[AdjacencyMatrix[VertexDelete[StarGraph[5], i]], 4, 1], {i, 1, 5, 1}]
```

Out[222]= {22.0067, 29.5634, 29.5634, 29.5634, 29.5634}

In[200]:=
```
del1BDM = SquareBDM[AdjacencyMatrix[VertexDelete[StarGraph[5], 1]], 4, 1]
```

Out[200]= 22.0067

In[201]:=
```
del4BDM = SquareBDM[AdjacencyMatrix[VertexDelete[StarGraph[5], 4]], 4, 1]
```

Out[201]= 29.5634

In[202]:=
```
origBDM - del4BDM
```

Out[202]= 75.9639

In[203]:=
```
origBDM = SquareBDM[AdjacencyMatrix[VertexDelete[StarGraph[5], 2]], 4, 1]
```

Out[203]= 29.5634

In[205]:=
```
robustnessSeqAttack[StarGraph[5], DegreeCentrality]
```

Out[205]= 0.16

In[206]:=
```
Reverse[SortBy[Transpose[{VertexList[#], EigenvectorCentrality[#]}], Last]] &@
  StarGraph[5]
```

Out[206]= {{1, 0.333333}, {4, 0.166667}, {3, 0.166667}, {2, 0.166667}, {5, 0.166667}}

In[207]:=
```
getVertexAttackComplexityRank[StarGraph[5]]
```
Out[207]=
```
{{1, 83.5206}, {5, 75.9639}, {4, 75.9639}, {3, 75.9639}, {2, 75.9639}}
```

In[208]:=
```
getVertexAttackComplexityRank[CompleteGraph[5]]
```
Out[208]=
```
{{5, 50.1874}, {4, 50.1874}, {3, 50.1874}, {2, 50.1874}, {1, 50.1874}}
```

In[209]:=
```
robustnessSeqAttack[StarGraph[5], DegreeCentrality]
```
Out[209]=
```
0.16
```

In[210]:=
```
robustnessSeqAttackBDM[StarGraph[5]]
```
Out[210]=
```
0.160
```

In[211]:=
```
robustnessSeqAttackBDM[CompleteGraph[5]]
```
Out[211]=
```
0.400
```

In[212]:=
```
robustnessSeqAttack[CompleteGraph[5], #] & /@
 {BetweennessCentrality, DegreeCentrality,  EigenvectorCentrality}
```
Out[212]=
```
{0.40, 0.40, 0.40}
```

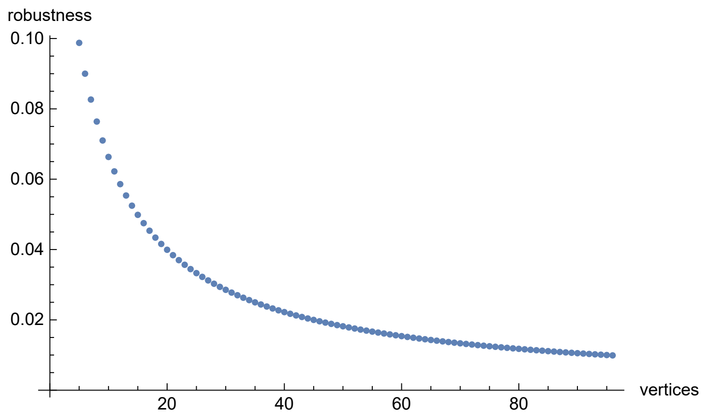
In[213]:=
```
getEdgeAttackComplexityRank[StarGraph[5]]
```
Out[213]=
```
{{1 ⟷ 2, 2.98977}, {1 ⟷ 4, 2.71213}, {1 ⟷ 3, 1.18206}, {1 ⟷ 5, -0.571505}}
```

In[214]:=
```
robustnessSimAttack[StarGraph[5], DegreeCentrality]
```
Out[214]=
```
0.160
```

In[215]:=
```
starGraphBDMRobustness = Table[robustnessSeqAttackBDM[StarGraph[i]], {i, 5, 100, 1}]
```
Out[215]=
```
{0.160, 0.139, 0.122, 0.109, 0.0988, 0.0900, 0.0826, 0.0764, 0.0710, 0.0663, 0.0622, 0.0586,
 0.0554, 0.0525, 0.0499, 0.0475, 0.0454, 0.0434, 0.0416, 0.0399, 0.0384, 0.0370, 0.0357,
 0.0344, 0.0333, 0.0322, 0.0312, 0.0303, 0.0294, 0.0285, 0.0278, 0.0270, 0.0263, 0.0256,
 0.0250, 0.0244, 0.0238, 0.0232, 0.0227, 0.0222, 0.0217, 0.0213, 0.0208, 0.0204, 0.0200,
 0.0196, 0.0192, 0.0189, 0.0185, 0.0182, 0.0179, 0.0175, 0.0172, 0.0169, 0.0167, 0.0164,
 0.0161, 0.0159, 0.0156, 0.0154, 0.0151, 0.0149, 0.0147, 0.0145, 0.0143, 0.0141,
 0.0139, 0.0137, 0.0135, 0.0133, 0.0132, 0.0130, 0.0128, 0.0127, 0.0125, 0.0123,
 0.0122, 0.0120, 0.0119, 0.0118, 0.0116, 0.0115, 0.0114, 0.0112, 0.0111, 0.0110,
 0.0109, 0.0108, 0.0106, 0.0105, 0.0104, 0.0103, 0.0102, 0.0101, 0.0100, 0.00990}
```

In[216]:=
```
Labeled[ListPlot[starGraphBDMRobustness, AxesLabel → {"vertices", "robustness"}],
 "Star Graph - Robustness to Sequential Attack guided by BDM"]
```

Out[216]=



Star Graph – Robustness to Sequential Attack guided by BDM

In[217]:=
```
{robustnessSeqAttackBDM[StarGraph[50]],
 robustnessSeqAttack[StarGraph[50], DegreeCentrality]}
```

Out[217]= {0.0196, 0.020}

In[218]:=
```
{robustnessSeqAttackBDM[CompleteGraph[50]],
 robustnessSeqAttack[CompleteGraph[50], DegreeCentrality]}
```

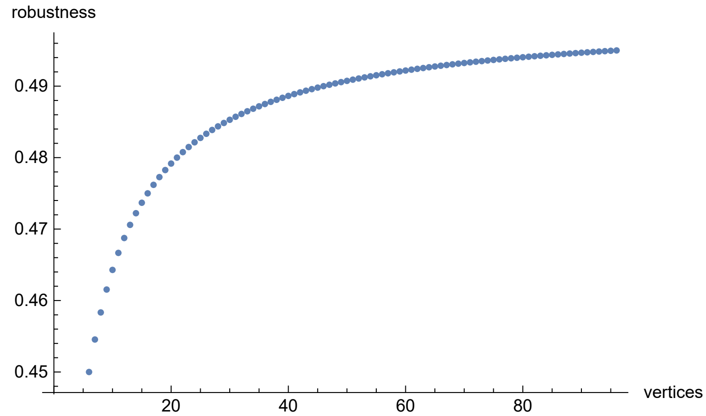
Out[218]= {0.490, 0.49}

In[219]:=
```
completeGraphBDMRobustness =
 Table[robustnessSeqAttackBDM[CompleteGraph[i]], {i, 5, 100, 1}]
```

Out[219]=
```
{0.400, 0.417, 0.429, 0.438, 0.444, 0.450, 0.455, 0.458, 0.462, 0.464, 0.467, 0.469,
 0.471, 0.472, 0.474, 0.475, 0.476, 0.477, 0.478, 0.479, 0.480, 0.481, 0.481, 0.482,
 0.483, 0.483, 0.484, 0.484, 0.485, 0.485, 0.486, 0.486, 0.486, 0.487, 0.487, 0.488,
 0.488, 0.488, 0.488, 0.489, 0.489, 0.489, 0.489, 0.490, 0.490, 0.490, 0.490, 0.490,
 0.491, 0.491, 0.491, 0.491, 0.491, 0.491, 0.492, 0.492, 0.492, 0.492, 0.492, 0.492,
 0.492, 0.492, 0.493, 0.493, 0.493, 0.493, 0.493, 0.493, 0.493, 0.493, 0.493, 0.493,
 0.494, 0.494, 0.494, 0.494, 0.494, 0.494, 0.494, 0.494, 0.494, 0.494, 0.494, 0.494,
 0.494, 0.494, 0.495, 0.495, 0.495, 0.495, 0.495, 0.495, 0.495, 0.495, 0.495, 0.495}
```

In[220]:=
```
Labeled[ListPlot[completeGraphBDMRobustness, AxesLabel → {"vertices", "robustness"}],
 "Complete Graph - Robustness to Sequential Attack guided by BDM"]
```

Out[220]=



Complete Graph - Robustness to Sequential Attack guided by BDM

The robustness to attacks directed using the Block Decomposition Method agrees with those directed using the degree of each vertex.

In[223]:=
```
starGraphDegreeRobustness =
 Table[robustnessSeqAttack[StarGraph[i], DegreeCentrality], {i, 5, 100, 1}]
```

Out[223]=
```
{0.16, 0.14, 0.12, 0.11, 0.099, 0.090, 0.083, 0.076, 0.071, 0.066, 0.062, 0.059,
 0.055, 0.052, 0.050, 0.048, 0.045, 0.043, 0.042, 0.040, 0.038, 0.037, 0.036, 0.034,
 0.033, 0.032, 0.031, 0.030, 0.029, 0.029, 0.028, 0.027, 0.026, 0.026, 0.025, 0.024,
 0.024, 0.023, 0.023, 0.022, 0.022, 0.021, 0.021, 0.020, 0.020, 0.020, 0.019, 0.019,
 0.019, 0.018, 0.018, 0.018, 0.017, 0.017, 0.017, 0.016, 0.016, 0.016, 0.016, 0.015,
 0.015, 0.015, 0.015, 0.014, 0.014, 0.014, 0.014, 0.014, 0.014, 0.013, 0.013, 0.013,
 0.013, 0.013, 0.012, 0.012, 0.012, 0.012, 0.012, 0.012, 0.012, 0.011, 0.011, 0.011,
 0.011, 0.011, 0.011, 0.011, 0.011, 0.011, 0.010, 0.010, 0.010, 0.010, 0.010, 0.0099}
```

In[225]:=
```
starGraphBDMRobustness === starGraphDegreeRobustness
```

Out[225]= True

In[226]:=
```
completeGraphDegreeRobustness =
 Table[robustnessSeqAttack[CompleteGraph[i], DegreeCentrality], {i, 5, 100, 1}]
```

Out[226]=
```
{0.40, 0.42, 0.43, 0.44, 0.44, 0.45, 0.45, 0.46, 0.46, 0.46, 0.47, 0.47, 0.47,
 0.47, 0.47, 0.48, 0.48, 0.48, 0.48, 0.48, 0.48, 0.48, 0.48, 0.48, 0.48, 0.48, 0.48,
 0.48, 0.48, 0.49, 0.49, 0.49, 0.49, 0.49, 0.49, 0.49, 0.49, 0.49, 0.49, 0.49, 0.49,
 0.49, 0.49, 0.49, 0.49, 0.49, 0.49, 0.49, 0.49, 0.49, 0.49, 0.49, 0.49, 0.49, 0.49,
 0.49, 0.49, 0.49, 0.49, 0.49, 0.49, 0.49, 0.49, 0.49, 0.49, 0.49, 0.49, 0.49, 0.49,
 0.49, 0.49, 0.49, 0.49, 0.49, 0.49, 0.49, 0.49, 0.49, 0.49, 0.49, 0.49, 0.49, 0.49,
 0.49, 0.49, 0.49, 0.49, 0.49, 0.49, 0.49, 0.49, 0.49, 0.49, 0.49, 0.49, 0.50}
```

In[227]:=
```
completeGraphBDMRobustness === completeGraphDegreeRobustness
```

Out[227]= True

References

- [Zenil16] Zenil, H., Hernández-Orozco, S., Kiani, N. A., Soler-Toscano, F., & Rueda-Toicen, A. (2016). A decomposition method for global evaluation of Shannon entropy and local estimations of algorithmic complexity. arXiv preprint arXiv:1609.00110.
- [Zenil14] Zenil H., Soler - Toscano F., Dingle K.and Louis A.(2014) Correlation of Automorphism Group Size and Topological Properties with Program - size Complexity Evaluations of Graphs and Complex Networks, Physica A : Statistical Mechanics and its Applications, vol.404, pp.341–358.
- [Iyer13] Iyer, S., Killingback, T., Sundaram, B., & Wang, Z. (2013). Attack robustness and centrality of complex networks. PloS one, 8(4), e59613.
- [Soler13] Soler-Toscano, F. and Zenil, H. Kolmogorov Complexity of 3 × 3 and 4 × 4 Squares, Wolfram Demonstrations Project, 2013 http://demonstrations.wolfram.com/KolmogorovComplexityOf33And44Squares/
- [Zenil18a] Zenil, H., Kiani, N. A., Zea, A. & Tegnér, J. (2018). Ab initio Algorithmic Causal Deconvolution of Intertwined Programs and Networks by Generative Mechanism. arXiv preprint arXiv:1802.09904.
- [Zenil18b] Zenil, H., Kiani, N. A., Rueda-Toicen, A., Zea, A. & Tegnér, J. (2018) Universal Data Reduction and Network Sparsification Method By Minimal Algorithmic Information Loss arXiv preprint arXiv:1802.05843
- [Zenil17]Zenil, H., Kiani, N. A., Marabita, F., Deng, Y., Elias, S., Schmidt, A. & Tegner, J. (2017). An Algorithmic Information Calculus for Causal Discovery and Reprogramming Systems. arXiv preprint arXiv:1709.05429.