T.C.
PAMUKKALE ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

AĞ GÜVENLİĞİ İÇİN YIĞILMA İŞLEMİNE DAYALI RUPTURE DERECESİ PARAMETRELERİNİN HESAPLANMASI

YÜKSEK LİSANS TEZİ

MUAMMER AĞTAŞ

DENİZLİ, AĞUSTOS - 2024

T.C.
PAMUKKALE ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

AĞ GÜVENLİĞİ İÇİN YIĞILMA İŞLEMİNE DAYALI RUPTURE DERECESİ PARAMETRELERİNİN HESAPLANMASI

YÜKSEK LİSANS TEZİ

MUAMMER AĞTAŞ

DENİZLİ, AĞUSTOS - 2024

Bu tezin tasarımı, hazırlanması, yürütülmesi, araştırmalarının yapılması ve bulgularının analizlerinde bilimsel etiğe ve akademik kurallara özenle riayet edildiğini; bu çalışmanın doğrudan birincil ürünü olmayan bulguların, verilerin ve materyallerin bilimsel etiğe uygun olarak kaynak gösterildiğini ve alıntı yapılan çalışmalara atfedildiğine beyan ederim.

Muammer AĞTAŞ

ÖZET

AĞ GÜVENLİĞİ İÇİN YIĞILMA İŞLEMİNE DAYALI RUPTURE DERECESİ PARAMETRELERİNİN HESAPLANMASI
YÜKSEK LİSANS TEZİ
MUAMMER AĞTAŞ
PAMUKKALE ÜNİVERSİTESİ FEN BİLİMLERİ ENSTİTÜSÜ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

(TEZ DANIŞMANI: PROF. DR. TUFAN TURACI)

DENİZLİ, AĞUSTOS - 2024

Ağ güvenliği bilgi işlem alanında önemli bir konudur. Zedelenebilirlik, bir ağda bulunan cihazların veya bağlantıların zarar görmesi durumunda iletişimlerinin kopana kadar gösterdikleri dayanma gücüne denir. Bu zedelenebilirlik ölçümünün yapılabilmesi için öncelikle ağın, cihazlar tepelerle, bağlantılar ayrıtlarla ifade edilecek şekilde graflarla modellenmesi gereklidir.

Rupture derecesi, graflarla modellenen ağlarda en önemli zedelenebilirlik parametrelerinden biridir.$G(V(G), E(G))$ basit, yönsüz bir graf olsun. Rupture derecesi $r(G) = max\{w(G-S) - |S| - m(G-S): S \subset V(G)$ ve $w(G-S) > 1\}$ ile tanımlanır, burada $w(G-S)$ bileşen sayısı ve $m(G-S)$tepeler koparıldıktan sonra grafta kalan en büyük bileşenin tepe sayısıdır. Bu tezde, G grafının her bir tepe için *ağ yığılma* (*agglomeration*) işlemine dayanan tepe daralma yöntemin ele alınmıştır. Daha sonra, agglomeration rupture (yığılma kopma) derecesi ve ortalama alt agglomeration rupture (ortalama alt yığılma kopma) derecesi olarak adlandırılan iki adet graf zedelenebilirlik parametresi sunulmuştur. Ayrıca bazı graf aileleri için bu parametrelerin kesin değerleri verilmiştir. Son olarak, agglomeration rupture derecesi ve ortalama alt agglomeration rupture derecesi değerlerini elde etmek için polinom zamanlı bir sezgisel algoritma önerilmiştir.

ANAHTAR KELİMELER: graflar, ağ tasarımı ve iletişimi, karmaşık ağlar, zedelenebilirlik, bağlantılılık, rupture derecesi, yığılma işlemi

ABSTRACT

COMPUTING RUPTURE DEGREE PARAMETERS BASED ON AGGLOMERATION OPERATION FOR NETWORK SECURITY
MSC THESIS
MUAMMER AGTAS
PAMUKKALE UNIVERSITY INSTITUTE OF SCIENCE
COMPUTER ENGINEERING

(SUPERVISOR: PROF. DR. TUFAN TURACI)

DENİZLİ, AUGUST 2024

Network security is an important issue in computing. The vulnerability is a in case the devices or connections in the network are damaged, it refers to the endurance they show until their communication is broken. In order to make this vulnerability measurement, the network must first be modeled as graphs, with devices expressed as vertices and connections as edges.

The rupture degree is one the most important vulnerability parameter in networks which are modelled by graphs. Let $G(V(G), E(G))$be a simple undirected graph. The rupture degree is defined by $r(G) = max\{w(G-S)-|S|-m(G-S): S \subset V(G) \text{ and } w(G-S)>1\}$, where $m(G-S)$ is the order of a largest connected component in $G-S$ and $w(G\text{-}S)$ is the number of components of $G-S$, respectively. In this thesis, we consider the vertex contraction method based on the network agglomeration operation for each vertex of graph G. Then, we have presented two graph vulnerability parameters called by agglomeration rupture degree and average lower agglomeration rupture degree. Furthermore, the exact values of them for some graph families are given. Finally, we proposed a polynomial time heuristic algorithm to obtain the values of agglomeration rupture degree and average lower agglomeration rupture degree.

KEYWORDS: graphs, network design and communication, complex networks, vulnerability, connectivity, rupture degree, agglomeration

İÇİNDEKİLER

Sayfa

ŞEKİL LİSTESİ

<u>Sayfa</u>

TABLO LİSTESİ

Sayfa

SEMBOL VE KISALTMALAR LİSTESİ

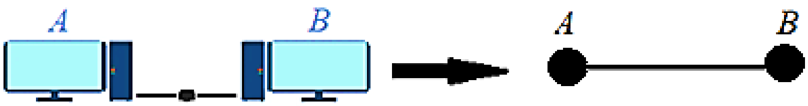
$\boldsymbol{G}$	:	Graf
$\boldsymbol{V(G)}$	:	G grafının tepeleri
$\boldsymbol{E(G)}$	:	G grafının ayrıtları
$\boldsymbol{n}$	:	Bir G grafının tepe sayısına bağlı seviyesi
$\boldsymbol{v}$	:	Bir grafın tepesi
$\boldsymbol{e}$	:	Bir grafın ayrıtı
$\boldsymbol{N(u)}$	:	Açık komşuluk
$\boldsymbol{d_G(u)}$	:	Tepe derecesi
$\Delta(\boldsymbol{G})$	:	Maksimum tepe derecesi
$\boldsymbol{\delta(G)}$	:	Minimum tepe derecesi
$\boldsymbol{d(u,v)}$	:	İki tepe arası mesafe
$\boldsymbol{k(G)}$	:	Bağlantılılık sayısı
$\boldsymbol{r(G)}$	:	Rupture derecesi
$\boldsymbol{S}$	:	Koparılan tepeler kümesi
$\boldsymbol{\vert S\vert}$	:	Koparılan tepelerin sayısı
$\boldsymbol{G-S}$	:	G grafından S tepe kümesinin çıkarılması ile oluşan yeni graf
$\boldsymbol{m(G-S)}$	:	$G-S$ grafınınen büyük bileşeninin tepe sayısı
$\boldsymbol{w(G-S)}$	:	$G-S$ grafının bileşen sayısı
$\boldsymbol{r^{agg}(G)}$	:	Agglomeration rupture derecesi
$\boldsymbol{r_{av}^{agg}(G)}$	:	Ortalama alt agglomeration rupture derecesi
$\boldsymbol{P_n}$	:	Yol graf
$\boldsymbol{C_n}$	:	Çevre graf
$\boldsymbol{K_n}$	:	Tam graf
$\boldsymbol{K_{1,n-1}}$	:	Yıldız graf
$\boldsymbol{W_{1,n}}$	:	Tekerlek graf
$\boldsymbol{K_{m,n}}$	:	İki parçalı tam graf
$\boldsymbol{ARD}$	:	Agglomeration rupture derecesi
$\boldsymbol{ALARD}$	:	Ortalama alt agglomeration rupture derecesi
$\boldsymbol{O(q)}$	:	Bir algoritmanın q değişkenli zaman karmaşıklığı

ÖNSÖZ

Çalışmalarımda uçtan uca takipte olan, bilgi ve tecrübesiyle beni yönlendiren, benden yardımlarını, desteğini ve sabrını esirgemeyen çok değerli hocam Prof. Dr. Tufan TURACI' ya ve süreç boyunca yanımda olup, maddi-manevi desteklerini benden esirgemeyen eşim Ahsen KIPÇAK' a en içten teşekkürlerimi ve saygılarımı sunarım.

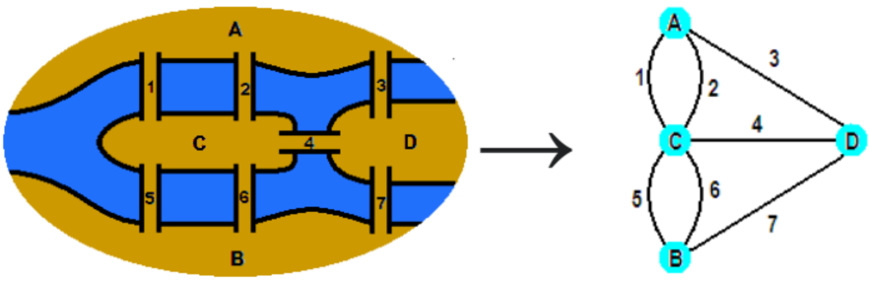
1. GİRİŞ

Günümüzün teknoloji devri olduğu kabul edilmiş bir gerçektir. Neredeyse her birey bu teknolojinin en belirgin örneklerinden olan bilgisayar, tablet pc, akıllı telefonlar vb. cihazlar ile doğrudan temas halindedir. Bu cihazlar kablolu ya da kablosuz bağlantılar vasıtasıyla birbirleri ile kolaylıkla haberleşebildiğinden dolayı iletişim başta olmak üzere birçok işlem kilometrelerce öteden halledilebilir durumdadır. Bu bağlantı kompleksine bilgisayar biliminde *Ağ* denilmektedir. Ağlar teknolojinin her alanında sıklıkla kullanılmaktadır. Gayet tabi teknolojinin bu denli yararı olduğu kadar riskleri de bulunmaktadır. Bu risklerin başında ise güvenlik zafiyeti konusu yer almaktadır. En çok görülen güvenlik sorunları ise ağlarda görülebilmektedir. Bu ve benzeri pek çok sorunun kolayca giderilmesi, güvenilirliğinin ölçülmesi, analizi vb. durumlar matematiksel modeller ile rahatlıkla gerçekleştirilmektedir. Matematiksel modellemenin en belirgin örneklerinden birisi ise graf teorisidir. Graf modelleme; ağları, şehir planlamalarını, şebeke projeleri, altyapı projeleri, yangın sistemleri vb. birçok sistem yaratırken kullanılmaktadır. Graflar tepe ve ayrıt denilen iki bileşenden oluşmaktadır. Örneğin 2 makineli bir bilgisayar ağını modelleyecek olur isek A ve B makineleri birer tepe, bu iki makinenin birbiri ile yaptığı bağlantı ise ayrıt olarak nitelendirilmektedir. Şekil 1.1 de bu örneğin graf modeli gösterilmiştir.



Şekil 1.1: Basit bir graf modeli.

Graf Teori, ünlü matematikçi Leonhard Euler tarafından ilk kez 18. yy da bahsedilmiştir. Leonhard Euler, Königsberg' in 7 Köprüsü isimli problemi graf ile modelleyerek çözmüştür. Königsberg şehrini dörde bölen bir nehir bulunmaktadır. Problem ise "Tüm köprülerden bir defa geçerek başlangıç noktasına geri dönülür mü?" sorusudur. Şekil 1.2 de kara parçaları harflerle, köprüler ise sayılarla etiketlenmiştir. Önce çözümü biraz daha kolaylaştırmak ve şekli gereksiz bileşenlerden arındırmak amacıyla kara parçalarının tepeler, köprülerin ise ayrıtlar olarak gösterildiği bir graf modeli oluşturulmuştur. Euler, bu model ile problemi ispatlamış ve sorunun cevabının "*hayır*" olduğunu tespit etmiştir (West 2001).



Şekil 1.2: Konigsberg köprü probleminin graf modeli (Taylor 2000).

Yukarıda da bahsettiğimiz gibi ağlar graflarla modellenebilir. Sunucular veya hub' lar herhangi bir G grafında tepelerle gösterilir ve ayrıtlar aralarında bağlantı ortamı oluşturur. Bir ağın güvenlik açığı, tepeler ve ayrıtlara göre ağ planlayıcıları için temel öneme sahiptir (Chvatal 1973). Son zamanlarda ağlardaki güvenlik açığı bilişim, matematik, bilgisayar bilimi, kimya ve diğer birçok uygulamalı bilim ve mühendislik bilimi gibi yaygın multidisipliner alanlarda araştırılmaktadır. Ağların güvenlik açığı değeri, bazı tepelerin veya ayrıtların bozulmasından sonra iletişim kesintisine kadar ağın dayanıklılığı olarak tanımlanmaktadır (Chvatal 1973 ve Mishkovski 2011). Bu dayanma gücünün ölçümüne ise *zedelenebilirlik* adı verilmektedir. Ağ tasarlanırken zedelenebilirliği yüksek olması planlaması yapılmalıdır. Zira bu ağın direncinin yüksek olması anlamına gelmektedir. Zedelenebilirliği büyük olan ağlara *Kararlı Ağlar* denilmektedir. Örneğin bir bankanın yerel ağının zedelenebilirliği yüksek olmalıdır, yani kararlı ağ olmalıdır.

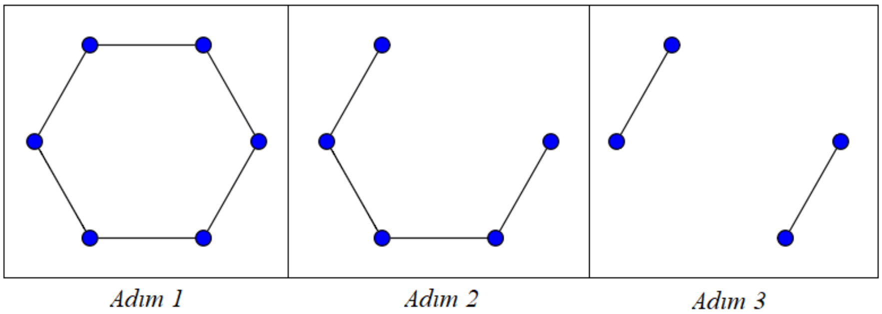
Bu tezde sadece basit graflar ele alınmıştır. Aşağıda bazı notasyonlar verilmiştir. $G\big(V(G), E(G)\big)$ grafı; tepe noktası ve ayrıt kümeleri $V(G)$ ve $E(G)$ile gösterilen, $V(G) = \{v_1, v_2, \dots, v_n\}, E(G) = \{e_1, e_2, \dots, e_m\}, |V(G)| = n$ ve $|E(G)| = m$ olmak üzere basit bağlantılı bir graf olarak ifade edilir. $u \in V(G)$ olsun. $N(u) = \{v \in V(G) | (u, v) \in E(G)\}$ kümesine u'nun *açık komşuluğu* denir. Ayrıca, $|N(u)|$ sayısına u tepesinin derecesi denir ve $d_G(u)$ ile gösterilir. G grafının *maksimum derecesi* $\Delta(G)$ ile gösterilir ve $max\{d_G(v) | \ v \in V(G)\}$şeklinde tanımlanır. Benzer şekilde, G grafının *minimum derecesi* $\delta(G)$ ile gösterilir ve $min\{d_G(v) \mid v \in V(G)\}$ şeklinde tanımlanır (Jung 1978 ve West 2001). $N[u] = \{u\} \cup N(u)$ kümesine

u tepesinin *kapalı komşuluğu* denir. $d(u, v)$, u ve v gibi iki tepe arasındaki mesafeyi temsil eder. Burada mesafe, u ve v tepeleri arasındaki en kısa yolun uzunluğu olarak tanımlanır (Jung 1978 ve West 2001).

Herhangi bir G grafının bağlantılılık sayısı (connectivity), literatürdeki en iyi bilinen zedelenebilirlik parametresidir. Birleştirilmiş bir G grafını birleştirilmemiş bir graf ya da izole tepelerden oluşan bir graf haline getirmek için graftan çıkarılması gereken en az tepe sayısına grafın bağlantılılık sayısı değeri denir ve $k(G)$ ile gösterilir (Frank ve Frisch 1970). Tanımı aşağıdaki (1.1) eşitliğindeki gibidir:

$$k(G) = \min_{S \subseteq V(G)} \{ | S |, w(G - S) \geq 2 \}. \tag{1.1}$$

Örneğin bir C_6 çevre grafının birleştirilmemiş bir graf haline gelmesi için atılması gereken en az tepe sayısı ikidir. Bu nedenle $k(C_6) = 2$ olmaktadır. Şekil 1.3 de adım adım C_6 grafının birleştirilmemiş bir graf olması için en az 2 alt graf bulundurana kadar tepeleri birer birer atılmıştır.



Şekil 1.3: C_6 grafının bağlantılılık sayısı örneği.

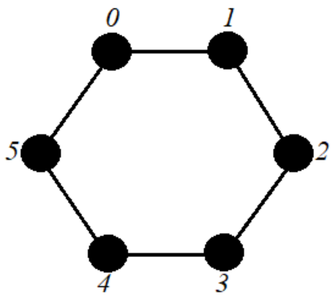
Herhangi bir G grafının bağlantılılık sayısı polinom zamanla hesaplanır. Ağlar için birçok zedelenebilirlik parametresi vardır. Örneğin, Integrity (Bütünlük) (Bagga ve diğ. 1992), Toughness (Dayanıklılık) (Chvatal 1973), Tenacity (Cozzens ve diğ. 1995), Global Distribution Number (Global Dağıtım Sayısı) (Durgut ve diğ. 2019) birçok alanda dikkate alınmakta ve incelenmektedir. Ayrıca ağların güvenlik açığı değerlerini elde etmek için birçok ortalama zedelenebilirlik parametresi önerilmiştir. Örnek olarak, Average lower domination number (Ortalama alt baskınlık sayısı) (Henning 2004), Average lower independence number (Ortalama alt bağımsızlık sayısı) (Aytac ve Turaci 2011), Average lower bondage number (Ortalama alt bağımlılık sayısı) (Turacı 2016), Average lower reinforcement number (Ortalama alt takviye sayısı) (Turacı ve Aslan 2016), Average lower residual domination number (Ortalama alt kalıntı baskınlık sayısı) (Turacı ve Aytac 2019), Average lower link residual domination number (Ortalama alt ayrıt kalıntı baskınlık sayısı) (Turacı 2020) vb. gibi pek çok zedelenebilirlik parametresi verilebilir. Bu parametrelerin değerleri polinom zamanda hesaplanmaz. Çünkü bunlar NP-Hard veya NP-Complete sınıfından problemlerdir.

Rupture (Kopma) derecesi diğer en iyi bilinen zedelenebilirlik parametrelerinden biridir. Li ve diğerleri tarafından tanımlanmıştır (Li ve diğ. 2005). Tanımı aşağıdaki (1.2) eşitliğindeki gibidir:

$$r(G) = max\{w(G-S)-|S|-m(G-S): S \subset V(G) \mid w(G-S) > 1\} \qquad (1.2)$$

Burada $m(G-S)$ ve $w(G-S)$ sırasıyla $G-S$'deki en büyük bağlı bileşenin tepe sayısını ve $G-S$ grafının bileşen sayısını belirtir.

C_6 bir çevre grafı olsun. Bir grafın rupture derecesini hesaplarken n tepe sayısı olmak üzere $2^n - 1$ tane tepe atma kombinasyonu vardır. Rupture derecesinin hesaplanabilmesi için $w(G-S) > 1$ şartından dolayı kalan bileşen sayısı en az 2 olmalıdır yani graf birleştirilmemiş olmalıdır. Şekil 1.4 deki gibi tepeleri etiketlendirilmiş C_6 grafı için Tablo 1.1 de $2^n - 1$ tane kombinasyonu, ilgili değerleri ve rupture derecesi hesaplanabilirliği durumu verilmiştir. C_6 grafı için 63 tane toplam kombinasyon vardır ama bunlardan 32 tanesi rupture derecesi tanımına uygundur.



Şekil 1.4: C_6 grafı.

Tablo 1.1: C_6 grafının rupture derecelerinin hesaplanması.

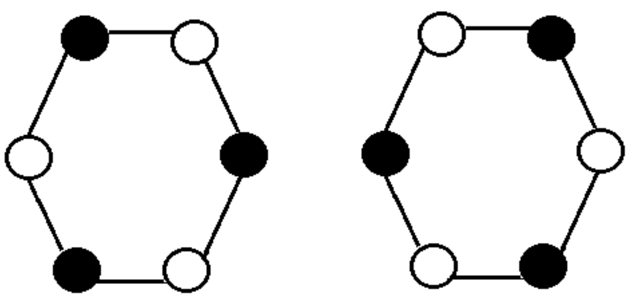
Atılan Tepeler *{S}*	Kalan Bileşen Sayısı *w(G-S)*	Atılan Tepe Sayısı *|S|*	En Büyük Bileşenin Tepe Sayısı *m(G-S)*	Rupture Derecesi *r(G)*	Rupture Derecesi Hesaplanır mı?
{0}	1	1	3	-3	Hayır
{0, 1}	1	2	3	-4	Hayır
{0, 1, 2}	1	3	3	-5	Hayır
{0, 1, 2, 3}	1	4	2	-5	Hayır
{0, 1, 2, 3, 4}	1	5	1	-5	Hayır
{0, 1, 2, 3, 5}	1	5	1	-5	Hayır
{0, 1, 2, 4}	2	4	1	-3	Evet
{0, 1, 2, 4, 5}	1	5	1	-5	Hayır
{0, 1, 2, 5}	1	4	2	-5	Hayır
{0, 1, 3}	2	3	2	-3	Evet
{0, 1, 3, 4}	2	4	1	-3	Evet
{0, 1, 3, 4, 5}	1	5	1	-5	Hayır
{0, 1, 3, 5}	2	4	1	-3	Evet
{0, 1, 4}	2	3	2	-3	Evet
{0, 1, 4, 5}	1	4	2	-5	Hayır
{0, 1, 5}	1	3	3	-5	Hayır
{0, 2}	2	2	3	-3	Evet
{0, 2, 3}	2	3	2	-3	Evet
{0, 2, 3, 4}	2	4	1	-3	Evet
{0, 2, 3, 4, 5}	1	5	1	-5	Hayır
{0, 2, 3, 5}	2	4	1	-3	Evet
{0, 2, 4}	3	3	1	-1	Evet
{0, 2, 4, 5}	2	4	1	-3	Evet
{0, 2, 5}	2	3	2	-3	Evet
{0, 3}	2	2	2	-2	Evet
{0, 3, 4}	2	3	2	-3	Evet
{0, 3, 4, 5}	1	4	2	-5	Hayır
{0, 3, 5}	2	3	2	-3	Evet
{0, 4}	2	2	3	-3	Evet
{0, 4, 5}	1	3	3	-5	Hayır
{0, 5}	1	2	3	-4	Hayır
{1}	1	1	3	-3	Hayır
{1, 2}	1	2	3	-4	Hayır
{1, 2, 3}	1	3	3	-5	Hayır
{1, 2, 3, 4}	1	4	2	-5	Hayır
{1, 2, 3, 4, 5}	1	5	1	-5	Hayır
{1, 2, 3, 5}	2	4	1	-3	Evet
{1, 2, 4}	2	3	2	-3	Evet
{1, 2, 4, 5}	2	4	1	-3	Evet
{1, 2, 5}	2	3	2	-3	Evet

Tablo 1.1: C_6 grafının rupture derecelerinin hesaplanması (Devamı).

{1, 3}	2	2	3	-3	Evet
{1, 3, 4}	2	3	2	-3	Evet
{1, 3, 4, 5}	2	4	1	-3	Evet
{1, 3, 5}	3	3	1	-1	Evet
{1, 4}	2	2	2	-2	Evet
{1, 4, 5}	2	3	2	-3	Evet
{1, 5}	2	2	3	-3	Evet
{2}	1	1	3	-3	Hayır
{2, 3}	1	2	3	-4	Hayır
{2, 3, 4}	1	3	3	-5	Hayır
{2, 3, 4, 5}	1	4	2	-5	Hayır
{2, 3, 5}	2	3	2	-3	Evet
{2, 4}	2	2	3	-3	Evet
{2, 4, 5}	2	3	2	-3	Evet
{2, 5}	2	2	2	-2	Evet
{3}	1	1	3	-3	Hayır
{3, 4}	1	2	3	-4	Hayır
{3, 4, 5}	1	3	3	-5	Hayır
{3, 5}	2	2	3	-3	Evet
{4}	1	1	3	-3	Hayır
{4, 5}	1	2	3	-4	Hayır
{5}	1	1	3	-3	Hayır
{0, 1, 2, 3, 4, 5}	0	6	0	-6	Hayır

Burada her bir kombinasyona karşılık gelen $w(G-S)-|S|-m(G-S)$değerleri hesaplanmıştır. Bu değerlerden maksimumu bize C_6 grafının rupture derecesi değerini vermektedir. Bu da [0, 2, 4] ve [1, 3, 5] kombinasyonlarıdır. Bu nedenle $r(C_6) = -1$ elde edilmiştir.

C_6 grafının alternatif kopma kümeleri ise graf üzerinde karartılmış tepeler kümesiyle Şekil 1.5 de gösterilmiştir.

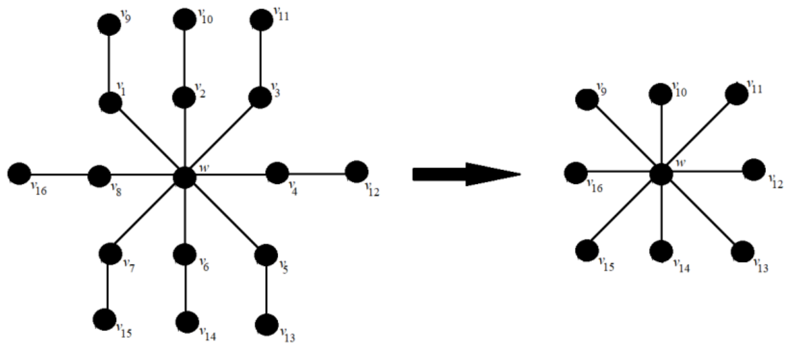


Şekil 1.5: C_6 grafının kopma kümesi.

Li (2004) tarafından rupture derecesi probleminin hesaplanmasının NP-complete bir problem olduğu gösterilmiştir. Fakat büyük graf sınıflarının kopma derecesini belirlemek mümkündür. Çeşitli sezgisel yöntemlerle ve algoritmalarla çok büyük grafların zedelenebilirliği hesaplanabilir. Rupture derecesi hakkında daha fazla sonuca Aslan (2015, 2016)'ın, Aytac ile Aksu (2010, 2013)'nun, Aytac ile Odabaş (2010)'ın, Bacak ile Öz (2017)'ün, Kırlangıç ile Bacak (2012)'ın, Kurkcu ile Aslan (2018)'ın ve Li (2015)'nin makalelerinde yer verilmiştir. Ayrıca Li tarafından n seviyeli ağaçlardaki kopma derecesini izole etmek için karmaşıklığı $O(n^2)$ olan bir algoritma verilmiştir (Li 2008). Rupture derecesi ile ilgili bir diğer ilgi çekici çalışma ise Durgut ve diğerleri tarafından gerçekleştirilmiştir (Durgut ve diğ. 2019). Durgut ve diğerleri (2019) tarafından herhangi bir G grafında rupture derecesini bulmak için bir sezgisel algoritma verilmiştir. Berberler ve diğerlerinin benzer bir çalışması da Integrity (Bütünlük) değerini hesaplamak için bulunmaktadır (Berberler 2017).

Her tepenin önemini belirlemek için bazı farklı yöntemler bulunmaktadır. Bu tezde ağ yığılmaya dayalı tepe daraltma yöntemi kullanılmıştır. Daha sonra ağ yığılma yöntemine dayalı tepe daralma yöntemi ve kopma derecesi birleştirilerek iki yeni güvenlik açığı parametre tanımı yapılmıştır. Yığılmaya dayalı yöntemler kullanılarak kırılganlık açısından daha verimli sonuçlar elde edilmiştir.$v_i \in V(G)$ olsun. Yığılma şu şekilde tanımlanır: v_i tepesi ve v_i ile bağlanan diğer $d_G(v_i)$ tepeleri, birincil $d_G(v_i) + 1$tepelerinin yerini alan yeni bir v'_i tepesine bağlanır ve $d_G(v_i) - 1$ tepelerine bağlanan ayrıtlar orijinal olarak yeni v'_i tepesine bağlanırlar. Örneğin, merkez tepe bir yıldız grafta daraltılmışsa, graf tek bir tepede toplanır. Başka bir örnek Şekil 1.6 da verilmiştir.

Yığılma işlemi farklı ağ güvenlik açığı parametrelerinde de kullanılmıştır, bunlardan bazılarına Berberler ve diğerlerinin (2021), Kunt ve ve Berberler'in (2020) ve Tan ve diğerlerinin (2006) makalelerin de yer verilmiştir. Bu tezde, $r^{agg}(G)$ ile gösterilen, Agglomeration Rupture Derecesi (ARD) olarak adlandırılan yeni graf parametrelerini tanıtmak için rupture derecesi ve yığılma işlemi kavramının yanı sıra ortalama güvenlik açığı parametreleri fikri de dahil edilmiştir. Bununla beraber, bir G grafı için $r_{av}^{agg}(G)$ ile gösterilen Ortalama Alt Agglomeration Rupture Derecesi (ALARD) parametresi de tanımlanmıştır. Ayrıca, agglomeration rupture derecesi ve ortalama alt agglomeration rupture derecesinin ağ güvenlik açığı için iki ölçüm olduğu sonucuna varılmıştır.



Şekil 1.6: *w* tepe noktasındaki yığılma işlemi.

Bu tezde 6 bölüm bulunmaktadır. 2. bölümde agglomeration rupture derecesi ve ortalama alt agglomeration rupture derecesi tanımlanmıştır. 3. bölümde agglomeration rupture derecesi ve ortalama alt agglomeration rupture derecesi hesaplamaları farklı örneklerle gösterilmiştir. 4. bölümde agglomeration rupture derecesi ve ortalama alt agglomeration rupture derecesi değerleri iyi bilinen bazı graf aileleri için elde edilmiştir. 5. bölüm de agglomeration rupture derecesi ve ortalama alt agglomeration rupture derecesi değerlerini hesaplamak için polinom zamanlı sezgisel bir algoritma verilmiştir ve daha sonra hesaplamalı test sonuçları sunulmuştur. Son olarak 6. bölümde elde edilen sonuçlar verilmiştir.

2. ARD VE ALARD TANIMLARI

Bir G grafının bir v_k tepe noktası için, $r_{v_k}^{agg}(G)$ ile gösterilen *alt agglomeration rupture değeri*, v_k tepesi için yığılma işleminden sonra elde edilen G' grafının rupture derecesi değeri olarak tanımlanır.

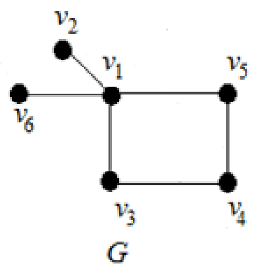
Bir G grafının agglomeration rupture derecesi (ARD) aşağıdaki (2.1) eşitliği ile tanımlanmaktadır:

$$r^{agg}(G) = \max_{v_k \in V(G)} \{r_{v_k}^{agg}(G)\}. \qquad (2.1)$$

Ayrıca, G grafının ortalama alt agglomeration rupture derecesi (ALARD) aşağıdaki (2.2) eşitliği ile tanımlanmaktadır:

$$r_{av}^{agg}(G) = \frac{1}{|V(G)|} \sum_{v_k \in V(G)} r_{v_k}^{agg}(G). \qquad (2.2)$$

Örnek 2.1. Şekil 2.1 de gösterilen G grafı 6 tepeli ve 6 ayrıtlı bir graf olsun. G grafının bağlantılılık sayısı $k(G) = 1$ ve rupture derecesi $r(G) = 1$ dir. G grafının kopma kümesi $\{v_1, v_4\}$ şeklindedir.



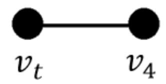
Şekil 2.1: Tepe ve ayrıt sayısı 6 olan G grafı.

Tablo 2.1: Her $v_k \in V(G)$ tepesinin alt agglomeration rupture değerleri.

Tepeler (v_k)	$r_{v_k}^{agg}(G)$
v_1	-1
v_2	0
v_3	1
v_4	1
v_5	1
v_6	0

Her $v_k \in V(G)$ tepesinin alt agglomeration rupture değeri Tablo 2.1 de gösterilmiştir.

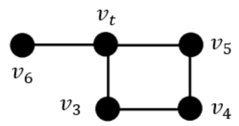
v_1 tepesine yığılma işlemi uygulandığında komşu tepeleri olan v_2, v_3, v_5, ve v_6 tepeleri ile tek bir v_t tepesi haline gelerek Şekil 2.2 de ki P_2 yol grafı elde edilmiştir.



Şekil 2.2: v_1 tepesi yığılınca oluşan P_2 grafı.

Böylece, $r_{v_1}^{agg}(P_2) = -1$ olarak bulunmuştur.

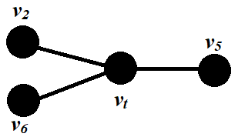
v_2 tepesine yığılma işlemi uygulandığında komşu tepesi olan v_1 tepesi ile tek bir v_t tepesi haline gelerek Şekil 2.3 de ki graf elde edilmiştir.



Şekil 2.3: v_2 tepesi yığılınca oluşan G grafı.

Rupture derecesi formülü bu grafa uygulandığında $r_{v_2}^{agg}(G) = 0$ elde edilir. Ayrıca, v_6 tepesi yığılınca da aynı graf oluşmaktadır ve $r_{v_6}^{agg}(G) = 0$ elde edilir.

v_3 tepesine yığılma işlemi uygulandığında komşu tepeleri olan v_1 ve v_4 tepeleri ile tek bir v_t tepesi haline gelerek $K_{1,3}$ yıldız grafı elde edilmiştir.



Şekil 2.4: v_3 tepesi yığılınca oluşan $K_{1,3}$ grafı.

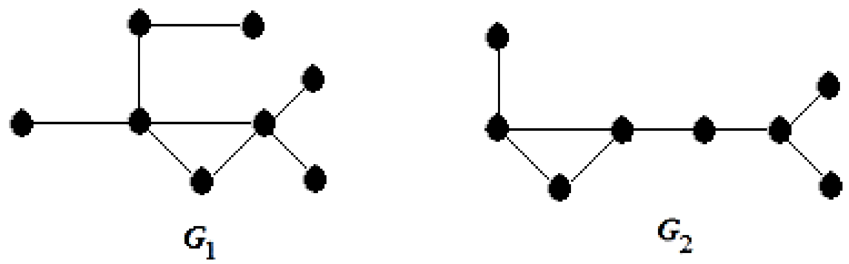
Böylece, $r_{v_3}^{agg}(K_{1,3}) = 1$ elde edilir. Bununla beraber, v_4 ve v_5 tepelerine yığılma işlemi sonucunda $K_{1,3}$ yıldız grafı oluşmaktadır ve böylece, $r_{v_4}^{agg}(G) = 1, r_{v_5}^{agg}(G) = 1$ elde edilir.

Sonuç olarak, $r_{v_1}^{agg}(G) = -1$, $r_{v_2}^{agg}(G) = 0$, $r_{v_3}^{agg}(G) = 1$, $r_{v_4}^{agg}(G) = 1$, $r_{v_5}^{agg}(G) = 1$ ve $r_{v_6}^{agg}(G) = 0$ değerleri elde edilmiştir. Böylece, $r^{agg}(G) = 1$ ve $r_{av}^{agg}(G) = (-1 + 0 + 1 + 1 + 1 + 0)/6 = 0.33$ elde edilmiştir.

3. ARD VE ALARD'IN GÜVENLİK AÇIĞI ÖRNEKLERİ

Agglomeration rupture derecesi ve ortalama alt agglomeration rupture derecesi, bazı grafların güvenlik açığının ölçülmesinde bağlantılılık sayısı ve rupture derecesinden daha verimli olabilir. Bu bölümde bu durum iki farklı örnekle gösterilmiştir.

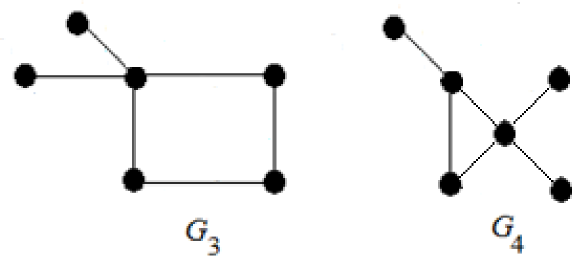
İlk örnekte, Şekil 3.1 de sunulan G_1 ve G_2 grafları ele alınmıştır. Daha sonra, verilen iki grafın hangisinin daha dayanıklı olduğunu ayırt etmek için agglomeration rupture derecesi ve ortalama alt agglomeration rupture derecesi değerleri gösterilmiştir. G_1 ve G_2 graflarının bağlantılılık ve rupture derecesi değerleri ve ayrıca tepe ve ayrıt sayıları eşittir. Yani, $k(G_1) = k(G_2) = 1, r(G_1) = r(G_2) = 1$, $|V\ (G_1)| = |V\ (G_2)| = 8\ ve\ |E(G_1)| = |E(G_2)| = 8$ şeklindedir.



Şekil 3.1: 8 tepeli ve 8 ayrıtlı G_1 ve G_2 grafları.

G_1 ve G_2 graflarının agglomeration rupture dereceleri $r^{agg}(G_1) = 2$ ve $r^{agg}(G_2) = 1$ olup, G_1 ve G_2 graflarının ortalama alt agglomeration rupture dereceleri $r_{av}^{agg}(G_1) = \frac{1}{2}$ ve $r_{av}^{agg}(G_2) = \frac{1}{4}$ olarak elde edilmiştir.

İkinci örnekte, Şekil 3.2 de sunulan G_3 ve G_4 grafları ele alınmıştır. Daha sonra, verilen iki grafı birbirinden ayırmak için kullanılabilen ortalama alt agglomeration rupture derece değerleri gösterilmiştir. Bu örnekte, G_3 ve G_4 graflarının bağlantılılık sayısı, rupture derecesi ve agglomeration rupture derecesi değerleri eşittir. Yani, $k(G_3) = k(G_4) = 1, r(G_3) = r(G_4) = 1\ ve\ r^{agg}(G_3) = r^{agg}(G_4) = 1$ şeklindedir. Ayrıca G_3 ve G_4 graflarının tepe ve ayrıt sayıları $|V(G_3)| = |V(G_4)| = 6\ ve\ |E(G_3)| = |E(G_4)| = 6$ gibi eşittir.



Şekil 3.2: 6 tepeli ve 6 ayrıtlı G_3 ve G_4 grafları.

G_3 ve G_4 graflarının ortalama alt agglomeration rupture dereceleri $r_{av}^{agg}(G_3) = \frac{1}{3}$ ve $r_{av}^{agg}(G_3) = 0$ olarak elde edilmiştir.

Bu örneklerle beraber, iki grafın dayanıklılığının karşılaştırılması yapılırken önerilen iki yeni zedelenebilirlik parametresi olan agglomeration rupture derecesi ve ortalama alt agglomeration rupture derecesi diğer zedelenebilirlik parametrelerine göre daha ayırt edici olabileceği görülmüştür.

4. İYİ BİLİNEN GRAFLARIN ARD VE ALARD DEĞERLERİNİN HESAPLANMASI

Bu bölümde yol graf P_n, çevre grafıC_n, tam graf K_n, yıldız grafı $K_{1,n-1}$, tekerlek grafı $W_{1,n}$ve iki parçalı tam graf $K_{n,m}$ graflarının agglomeration rupture derecesi ve ortalama alt agglomeration rupture derecesi değerleri hesaplanmıştır.

Teorem 1. $G \cong P_n,\ n \geq 4$ olmak üzere n tepeli bir yol graf olsun. Böylece,

$$\text{(a) } r^{agg}(P_n) = 0. \quad \text{(b) } r_{av}^{agg}(P_n) = \begin{cases} -2/n & \text{, n tek ise;} \\ (2-n)/n, & \text{n çift ise.} \end{cases} \tag{4.1}$$

Kanıt. n çift ise $r(P_n) = -1$, n tek ise $r(P_n) = 0$ olduğu bilinmektedir (Li ve diğ. 2005). $\{v_1, v_2, \ldots, v_{n-1}, v_n\}$ kümesinin elemanları,P_n grafının tepeleri olsun. Burada v_1 ve v_n tepelerine minör tepeler, geri kalan tepeler ise majör tepeler olarak adlandırılsın. Minör ve majör tepelerin sayısı sırasıyla 2 ve $n-2$'dir. P_n yol grafının agglomeration rupture derecesi ve ortalama alt agglomeration rupture derecesini hesaplarken n' ye bağlı iki durum vardır.

Durum 1. n çift olsun. P_n'nin tepelerine bağlı olarak iki alt durum vardır.

Alt durum 1.1. Eğer minör bir tepe noktası yığılırsa, P_{n-1} yolu elde edilir. n çift olduğundan $r(P_{n-1}) = 0$ elde edilmektedir. Böylece $r_{v_1}^{agg}(G) = 0$ ve $r_{v_n}^{agg}(G) = 0$ elde edilmiştir.

Alt durum 1.2. Eğer majör bir tepe noktası yığılırsa, P_{n-2} yolu elde edilir. n çift olduğundan $r(P_{n-2}) = -1$ elde edilmektedir. Böylece $r_{v_k}^{agg}(G) = -1$ elde edilmiştir. Burada $k \in \{2, 3, \ldots, n-1\}$ ‘dir.

Son olarak, agglomeration rupture derecesi tanımı ve alt durumlar 1.1 ve 1.2 ile (4.1) eşitliğinin a durumu elde edilmiştir.

Ayrıca ortalama alt agglomeration rupture derecesi aşağıdaki (4.5) eşitliğinde elde edilmiştir:

$$r_{av}^{agg}(G) = \frac{1}{|V(G)|}\left(\sum_{v_k \in V(G)} r_{v_k}^{agg}(G)\right) \tag{4.2}$$

$$= \frac{1}{n}\left(r_{v_1}^{agg}(G) + r_{v_n}^{agg} + \sum_{k=2}^{n-1} r_{v_k}^{agg}(G)\right) \tag{4.3}$$

$$= \frac{1}{n}\left(2(0) + (-1(n-2))\right) \tag{4.4}$$

$$= \frac{2-n}{n}. \tag{4.5}$$

Durum 2. n tek olsun. P_n' nin tepelerine bağlı olarak iki alt durum vardır.

Alt durum 2.1. Eğer minör bir tepe noktası yığılırsa, P_{n-1} yolu elde edilir. n tek olduğundan $r(P_{n-1}) = -1$ elde edilmektedir. Böylece $r_{v_1}^{agg}(G) = -1$ ve $r_{v_n}^{agg}(G) = -1$ elde edilmiştir.

Alt durum 2.2. Eğer büyük bir tepe noktası yığılırsa, P_{n-2} yolu elde edilir. n tek olduğundan $r(P_{n-2}) = 0$ elde edilmektedir. Böylece $r_{v_k}^{agg}(G) = 0$ elde edilmiştir. Burada $k \in \{2, 3, \ldots, n-1\}$ ‘dir.

Son olarak, agglomeration rupture derecesi tanımı ve alt durumlar 2.1 ve 2.2 ile (4.1) eşitliğinin b durumu elde edilmiştir. Ayrıca ortalama alt agglomeration rupture derecesi aşağıdaki (4.9) eşitliğinde elde edilmiştir:

$$r_{av}^{agg}(G) = \frac{1}{|V(G)|}\left(\sum_{v_k \in V(G)} r_{v_k}^{agg}(G)\right) \tag{4.6}$$

$$= \frac{1}{n}\left(r_{v_1}^{agg}(G) + r_{v_n}^{agg} + \sum_{k=2}^{n-1} r_{v_k}^{agg}(G)\right) \tag{4.7}$$

$$= \frac{1}{n}\left(2(-1) + (0(n-2))\right) \tag{4.8}$$

$$= \frac{-2}{n}. \tag{4.9}$$

Durum 1 ve 2 ile ispat tamamlanmıştır.

Teorem 2. $G \cong C_n, n \geq 5$ olmak üzere *n* tepeli bir çevre graf olsun. Böylece,

$$r^{agg}(C_n) = r_{av}^{agg}(C_n) = \begin{cases} -2, \text{ n tek ise;} \\ -1, \text{ n çift ise.} \end{cases} \tag{4.10}$$

Kanıt. *n* çift ise $r(C_n) = -1$, *n* tek ise $r(C_n) = -2$ olduğu bilinmektedir (Li ve diğ. 2005). $\{v_1, v_2, \ldots, v_{n-1}, v_n\}$ kümesinin elemanları, C_n grafının tepeleri olsun. Eğer C_n grafında bir tepe noktası yığılırsa C_{n-1} çevre grafı elde edilir. *n*' ye bağlı olarak iki durum vardır.

Durum 1. *n* çift olsun. *n* çift olduğundan $r(C_{n-2}) = -1$ elde edilmektedir. Böylece, $r_{v_k}^{agg}(G) = -1$ elde edilmiştir, burada $k \in \{1, 2, \ldots, n\}$ ‘dir. (4.10) eşitliğindeki çift durumu elde edilmiştir.

Durum 2. *n* tek olsun. *n* tek olduğundan $r(C_{n-2}) = -2$ elde edilmektedir. Böylece, $r_{v_k}^{agg}(G) = -2$ elde edilmiştir, burada $k \in \{1, 2, \ldots, n\}$ ‘dir. (4.10) eşitliğindeki tek durumu elde edilmiştir.

Böylece, aşağıdaki (4.11) eşitliği elde edilmiştir:

$$r^{agg}(C_n) = r_{av}^{agg}(C_n) = \begin{cases} -2, \text{ n tek ise;} \\ -1, \text{ n çift ise.} \end{cases} \tag{4.11}$$

Durum 1 ve 2 ile ispat tamamlanmıştır.

Teorem 3. $G \cong K_n,\ n \geq 3$ olmak üzere *n* tepeli bir tam graf olsun. Böylece,

$$\text{(a) } r^{agg}(K_n) = 0. \text{ (b) } r_{av}^{agg}(K_n) = 0. \tag{4.12}$$

Kanıt. K_n‘in rupture derecesi $r(K_n) = 1 - n$ ’dir (Li ve diğ. 2005). $\{v_1, v_2, \ldots, v_{n-1}, v_n\}$ kümesinin elemanları, K_n grafının tepeleri olsun. K_n grafının herhangi bir tepe noktası yığılırsaK_1grafı elde edilir. $r(K_n) = 0$ ‘dır. Böylece $r_{v_k}^{agg}(G) = 0$ olur. Burada $k \in \{1, 2, \ldots, n\}$ ‘dir. Sonuç olarak, (4.12) eşitliği elde edilmiştir.

Teorem 4. $G \cong K_{1,n-1}, n \geq 4$ olmak üzere n tepeli bir yıldız graf olsun. Böylece,

$$\text{(a) } r^{agg}(K_{1,n-1}) = n-4. \text{ (b) } r_{av}^{agg}(K_{1,n-1}) = \frac{n^2-5n+4}{n}. \tag{4.13}$$

Kanıt. $K_{1,n-1}$'in rupture derecesi $r(K_{1,n-1}) = n-3$ 'dür (Li ve diğ. 2005).$\{v_c, v_1, v_2, \dots, v_{n-2}, v_{n-1}\}$ kümesinin elemanları, $K_{1,n-1}$ grafının tepeleri olsun. Burada v_c tepesi $K_{1,n-1}$ grafının merkez tepesidir. $K_{1,n-1}$'in tepelerine bağlı olarak iki duruma ayrılmıştır.

Durum 1. Eğer merkez tepe noktası v_c yığılırsa, K_1 grafı elde edilir.$r(K_1) = 0$ olduğu bilinmektedir (Li ve diğ. 2005). Böylece $r_{v_c}^{agg}(G) = 0$ elde edilmiştir.

Durum 2. Eğer $k \in \{2, 3, \dots, n-1\}$için bir tepe $v_k$tepesi yığılırsa, $K_{1,n-1}$ yıldız grafı oluşur. Böylece $k \in \{2, 3, \dots, n-1\}$ için $r_{v_k}^{agg}(G) = n-4$ elde edilmiştir.

Son olarak, agglomeration rupture derecesi tanımı ve durum 1 ve durum 2 den (4.13) eşitliğindeki a durumu elde edilmiştir.

Ayrıca ortalama alt agglomeration rupture derecesi aşağıdaki (4.17) eşitliğinde elde edilmiştir:

$$r_{av}^{agg}(G) = \frac{1}{|V(G)|}\left(\sum_{v_k \in V(G)} r_{v_k}^{agg}(G)\right) \tag{4.14}$$

$$= \frac{1}{n}\left(r_{v_c}^{agg}(G) + \sum_{k=1}^{n-1} r_{v_k}^{agg}(G)\right) \tag{4.15}$$

$$= \frac{1}{n}\left((n-1) + (n-4)\right) \tag{4.16}$$

$$= \frac{n^2-5n+4}{n}. \tag{4.17}$$

Durum 1 ve 2 ile ispat tamamlanmıştır.

Teorem 5. $G \cong W_{1,n}, n \geq 5$ olmak üzere $n+1$ tepeli bir tekerlek graf olsun. Böylece,

$$\text{(a) } r^{agg}(W_{1,n}) = 0. \quad \text{(b) } r_{av}^{agg}(W_{1,n}) = \begin{cases} -2n/(n+1), \text{ n tek ise;} \\ -n/(n+1), \text{ n çift ise.} \end{cases} \tag{4.18}$$

Kanıt. $W_{1,n}$'nin rupture derecesi, n çift ise $r(W_{1,n}) = -2$ ve n tek ise $r(W_{1,n}) = -3$ olarak tanımlanır (Li ve diğ. 2005). $\{v_c, v_1, v_2, \ldots, v_{n-2}, v_{n-1}\}$ kümesinin elemanları, $W_{1,n}$ grafının tepeleri olsun; burada v_c tepesi $W_{1,n}$' nin merkez tepesidir. $W_{1,n}$ nin tepelerine bağlı olarak iki durum vardır.

Durum 1. Eğer merkez tepe noktası v_cyığılırsa, K_1 grafı elde edilir.$r(K_1) = 0$ olduğu bilinmektedir (Li ve diğ. 2005). Böylece $r_{v_c}^{agg}(G) = 0$ elde edilmiştir.

Durum 2. Eğer $k \in \{1, 2, \ldots, n\}$ için bir tepe v_k tepesi yığılırsa, $K_1 + P_{n-3}$ birleşmiş grafı elde edilir. Burada n değerine bağlı olarak iki alt durum vardır.

Alt durum 2.1. Eğer n çift ise $n-3$ tek olacaktır. $n-3$ tek olduğu için $r(K_1 + P_{n-3}) = -1$ elde edilmektedir (Li ve diğ. 2005). Bu nedenle, $r_{v_k}^{agg}(G) = -1$ olmaktadır. Burada $k \in \{1, 2, \ldots, n\}$ dir.

Alt durum 2.2. Eğer n tek ise $n-3$ çift olacaktır. $n-3$ tek olduğu için $r(K_1 + P_{n-3}) = -2$ elde edilmektedir (Li ve diğ. 2005). Bu nedenle $r_{v_k}^{agg}(G) = -2$ olmaktadır. Burada $k \in \{1, 2, \ldots, n\}$ dir.

Son olarak, agglomeration rupture derecesi tanımı ve durum 1 ve 2 ile (4.18) eşitliğindeki a durumu elde edilmiştir.

Ek olarak, n çift ise aşağıdaki (4.22) eşitliği elde edilmiştir:

$$r_{av}^{agg}(G) = \frac{1}{|V(G)|}\left(\sum_{v_k \in V(G)} r_{v_k}^{agg}(G)\right) \tag{4.19}$$

$$= \frac{1}{n}\left(r_{v_c}^{agg}(G) + \sum_{k=1}^{n-1} r_{v_k}^{agg}(G)\right) \tag{4.20}$$

$$= \frac{1}{n+1}(n)(-1) \tag{4.21}$$

$$= \frac{-n}{n+1}. \tag{4.22}$$

Eğer, n tek ise aşağıdaki (4.26) eşitliği elde edilmiştir:

$$r_{av}^{agg}(G) = \frac{1}{|V(G)|}\left(\sum_{v_k \in V(G)} r_{v_k}^{agg}(G)\right) \tag{4.23}$$

$$= \frac{1}{n}\left(r_{v_c}^{agg}(G) + \sum_{k=1}^{n-1} r_{v_k}^{agg}(G)\right) \tag{4.24}$$

$$= \frac{1}{n+1}(n)(-2) \tag{4.25}$$

$$= \frac{-2n}{n+1}. \tag{4.26}$$

Durum 1 ve 2 ile ispat tamamlanmıştır.

Teorem 6. $G \cong K_{n,m}, 1 < n \leq m$ olmak üzere $n+m$tepeli iki parçalı tam graf olsun. Böylece,

$$\text{(a) } r^{agg}(K_{n,m}) = m-3. \quad \text{(b) } r_{av}^{agg}(K_{n,m}) = \frac{m^2+n^2-3m-3n}{n+m}. \tag{4.27}$$

Kanıt. $K_{n,m}$'in rupture derecesi, $r(K_{n,m}) = 1-m-n$ olarak tanımlanır (Li ve diğ. 2005). $\{v_1, v_2, \ldots, v_n, v_1', v_2', \ldots, v_n'\}$ kümesinin elemanları, $K_{n,m}$ grafının tepeleri olsun; $K_{n,m}$' in tepelerine bağlı olarak iki durum vardır.

Durum 1. $k \in \{1,2,\ldots,n\}$ için eğer v_k tepesi yığılırsa $K_{1,n-1}$yıldız grafı elde edilir. $r(K_{1,n-1}) = n-3$ olmaktadır. Böylece $k \in \{1,2,\ldots,n\}$ için $r_{v_k}^{agg}(G) = n-3$ elde edilmiştir.

Durum 2. $k \in \{1,2,\ldots,m\}$ için eğer v_k' tepesi yığılırsa$K_{1,m-1}$ yıldız grafı elde edilir. $r(K_{1,m-1}) = m-3$olmaktadır. Böylece $k \in \{1,2,\ldots,m\}$ için $r_{v_k}^{agg}(G) = m-3$ elde edilmiştir.

$r(K_{n,m}) = max\{n-3, m-3\}$ve $n \leq m$olduğundan, durum 1 ve 2 ye dayanarak (4.27) eşitliğindeki a durumu elde edilmiştir.

Ayrıca ortalama alt agglomeration rupture derecesi aşağıdaki (4.31) eşitliği elde edilmiştir:

$$r_{av}^{agg}(G) = \frac{1}{|V(G)|}\left(\sum_{v_k \in V(G)} r_{v_k}^{agg}(G)\right) \quad (4.28)$$

$$= \frac{1}{n+m}\left(\sum_{k=1}^{n} r_{v_k}^{agg}(G) + \sum_{k=1}^{m} r_{v_k^{'}}^{agg}(G)\right) \quad (4.29)$$

$$= \frac{1}{n+m}\big((n)(n-3)+(m)(m-3)\big) \quad (4.30)$$

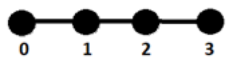
$$= \frac{m^2+n^2-3m-3n}{n+m}. \quad (4.31)$$

Durum 1 ve 2 ile ispat tamamlanmıştır.

5. ARD VE ALARD DEĞERLERİNİ HESAPLAMAK İÇİN SEZGİSEL BİR ALGORİTMA

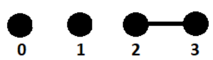
Bu bölümde öncelikle Ek A da agglomeration rupture derecesi ve ortalama alt agglomeration rupture derecesi için sezgisel algoritmanın sözde kodu verilmiştir. Bu algoritma, rastgele bir G grafının agglomeration rupture derecesi ve ortalama alt agglomeration rupture derecesini bulmak için polinom zamanda çalışmaktadır. Önerilen algoritmanın nasıl çalıştığına dair bir örneği aşağıdaki P_4 grafında gösterilmiştir.

P_4 bir yol grafı olsun ve Process fonksiyonundaki düğüm dizisi $[0,1,2,3]$ olsun. Bu graf Şekil 5.1 de gösterilmiştir.



Şekil 5.1: Tepeleri [0, 1, 2, 3] ile etiketlenen P_4 grafı.

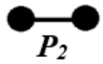
1 nolu tepe ve graf algoritmamızdaki Agglomeration fonksiyonuna girmiştir. Komşular dizimizin içeriği $[[1],[0,2],[1,3],[2]]$ olmuştur. Örneğin sıfırıncı indeksin içeriği 1'dir. Yani 0 nolu düğümün komşusu 1 olmuştur. *aggCluster* dizisinin içeriği de $[0,2]$ olmuştur. Daha sonra karşılık gelen tepeyi ekleyip en büyük sayıdan en küçük sayıya doğru sıralanmıştır. İçerik $[2,1,0]$olmuştur. Sonrasında iki boyutlu *graph* dizisinden satır ve sütunu silinmiştir. Bu süreç aynı zamanda *aggCluster*'a dayanmaktadır. Silme işleminden sonra *components* dizisi oluşturulmuştur ve aşağıdaki Şekil 5.2 deki gibi $components = [[0],[1],[2,3]]$ elde edilmiştir.



Şekil 5.2: İlk silmeden sonraki P_4 grafı.

Artık komşular *components* dizisinden silinmiştir. Özetle bu, *components* dizisindeki agglomeration tepe noktasına bitişik olmayan tepeleri tutma işlevidir.

Artık $components = [[3]]$ dizisi bulunmaktadır. Daha sonra *labeledComponents=components* ataması yapılmıştır. Ayrıca tepedeki etiket numarası 1 olmuştur. labeledComponents tek içerik olduğundan, döngü bir kez döndürülmüştür ve *labeledComponents*$[0][0] =$ *etiket numarası* ataması yapılmıştır. Yani birleştirilen tepelerin komşusuna 1 etiketi verilmiş ve *newGraf* aşağıdaki Şekil 5.3 deki gibi P_2 oluşmuştur.



Şekil 5.3: P_2 grafı.

Oluşturulan *newGraph*, Durgut ve diğerlerinin önerdiği Rupture fonksiyonuna gönderilmiştir ve algoritmadan -1 değeri dönmüştür. Agglomeration işlevi aynı zamanda bu tamsayı değerini de döndürür. Process fonksiyonundan gelen tamsayı değeri kopma sırasına eklenmiştir. Bu olay tüm tepeler için yapılmıştır ve *ruptures*= $[0, -1, -1, 0]$ elde edilmiştir. Sonuçta maksimum değer agglomeration rupture derecesi, aritmetik ortalaması da ortalama alt agglomeration rupture derecesi olmuştur. Sonuç olarak $r^{agg}(P_4) = 0$ ve $r_{av}^{agg}(P_4) = \frac{-1}{2}$ elde edilmiştir.

5.1 Hesaplamalı Testler

Bu bölümde önerilen algoritmamızı gerçekleştirmek için Berberler'in (2017) ve Durgut ve diğerlerinin (2019) referanslarının veri kümeleri kullanılmıştır. Aşağıdaki tablolarda $|V|$ tepe sayısıdır; ARD, agglomeration rupture derecesinin sezgisel sonucudur; ARDopt, agglomeration rupture derecesinin kaba kuvvet sonucunu görüntüler; ALARD, ortalama alt agglomeration rupture derecesinin sezgisel sonucudur; ALARDopt, ortalama alt agglomeration rupture derecesinin kaba kuvvet sonucu; t(s), saniye cinsinden çalışma süresini temsil etmektedir. HP, önerilen algoritma ile elde edilen ARD, ALARD değerleri ile ARD, ALARD sonuçları arasındaki farkın büyüklüğü olan mutlak boşluktur, açılımı ise hata payıdır. Ayrıca %25, %50, %75 ve %100, *G* grafının ayrıt yoğunluğunu belirtmektedir.

Önerilen algoritma JAVA dilinde uygulanmış ve 2,9 GHz işlemci ve 8 GB RAM'e sahip i5-7600U makinesinde test edilmiştir. Gerçek ARD ve ALARD sonuçlarının Tablo 5.1 ve 5.2 de önerilen algoritma ile elde edilen ARD ve ALARD sonuçlarıyla hemen hemen benzer olduğu gözlemlenmiştir. Ayrıca algoritmamız tepe sayıları 100'den fazla olan orta boyutlu graflar için de test edilmiştir. ARD ve ALARD'ın gerçek değerleri bilinmediği için Tablo 5.3 ve 5.4 da sadece ARD ve ALARD'ın CPU zamanı ile sezgisel sonucu verilmiştir. Sonuç olarak algoritma kullanılan bazı graf aileleri üzerinde test edilmiştir. Teoremler 1-6 da elde edilen tüm ARD ve ALARD değerleri algoritmamızda aynı olarak elde edilmiştir.

Tablo 5.1: ARD için küçük boyutlu graflar üzerinde hesaplamalı deneyler.

	25%			50%			75%			95%		
V	ARD	ARD opt	HP	ARD	ARD opt	HP	ARD	ARD opt	HP	ARD	ARD opt	HP
10	0	0	0	0	0	0	1	1	0	0	0	0
11	1	1	0	1	1	0	0	0	0	0	0	0
12	0	0	0	0	0	0	-1	-1	0	0	0	0
13	0	0	0	1	1	0	1	1	0	0	0	0
14	-1	-1	0	-1	-1	0	0	0	0	0	0	0
15	2	2	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	-1	-1	0	0	0	0
17	2	2	0	0	0	0	0	0	0	0	0	0
18	-1	-1	0	-1	-1	0	0	0	0	0	0	0
19	-1	-1	0	-1	-1	0	-1	-1	0	0	0	0
20	-1	-1	0	0	0	0	-1	-1	0	0	0	0
21	-1	-1	0	-2	-2	0	0	0	0	0	0	0
22	-1	-1	0	-1	-1	0	-1	-1	0	0	0	0
23	1	1	0	-2	-2	0	-1	-1	0	0	0	0
24	-3	-3	0	-2	-2	0	0	0	0	0	0	0
25	-3	-3	0	-3	-3	0	0	0	0	0	0	0
26	-3	-3	0	-3	-3	0	1	1	0	0	0	0
27	-4	-4	0	-2	-2	0	-1	-1	0	0	0	0

Agglomeration rupture derecesinin kaba kuvvet algoritma ve sezgisel algoritma sonuçları Tablo 5.1 de görüldüğü gibi %100 oranında aynı çıkmıştır. Buradan sezgisel algoritma ile yapılan agglomeration rupture derecesi hesaplamalarının güvenilirliğinin yüksek olduğu görülmüştür.

Tablo 5.2: ALARD için küçük boyutlu graflar üzerinde hesaplamalı deneyler.

	25%			50%			75%			95%		
[V]	ALARD	ALARDopt	HP	ALARD	ALARDopt	HP	ALARD	ALARDopt	HP	ALARD	ALARDopt	HP
10	-11/10	-11/10	0	-14/10	14/10	0	-6/10	-6/10	0	-4/10	-4/10	0
11	-7/11	-6/11	-1/11	-14/11	-14/11	0	-12/11	-12/11	0	-6/11	-6/11	0
12	-9/12	-9/12	0	-28/12	-28/12	0	-32/12	-32/12	0	-6/12	-6/12	0
13	-19/13	-19/13	0	-19/13	-19/13	0	-18/13	-18/13	0	-8/13	-8/13	0
14	-32/14	-32/14	0	-34/14	-34/14	0	-26/14	-26/14	0	-10/14	-10/14	0
15	-15/15	-14/15	-1/15	-37/15	-37/15	0	-30/15	-30/15	0	-10/15	-10/15	0
16	-32/16	-32/16	0	-48/16	-48/16	0	-42/16	-42/16	0	-12/16	-12/16	0
17	-23/17	-23/17	0	-43/17	-43/17	0	-40/17	-40/17	0	-14/17	-14/17	0
18	-43/18	-43/18	0	-58/18	-58/18	0	-40/18	-40/18	0	-16/18	-16/18	0
19	-53/19	-53/19	0	-56/19	-56/19	0	-50/19	-50/19	0	-12/19	-12/18	0
20	-63/20	-63/20	0	-69/20	-69/20	0	-53/20	-53/20	0	-20/20	-20/20	0
21	-90/21	-87/21	-3/21	-99/21	-99/21	0	-70/21	-70/21	0	-14/21	-14/21	0
22	-86/22	-86/22	0	-105/22	-104/22	-1/22	-59/22	-59/22	0	-24/22	-24/22	0
23	-95/23	-91/23	-4/23	-109/23	-109/23	0	-62/23	-62/23	0	-26/23	-26/23	0
24	-133/24	-133/24	0	-132/24	-132/24	0	-82/24	-82/24	0	-28/24	-28/24	0
25	-143/25	-142/25	-1/25	-144/25	-144/25	0	-95/25	-95/25	0	-30/25	-30/25	0
26	-148/26	-148/26	0	-168/26	-166/26	-2/26	-96/26	-96/26	0	-32/26	-32/26	0
27	-157/27	-157/27	0	-167/27	-167/27	0	-105/27	-105/27	0	-36/27	-36/27	0

Ortalama alt agglomeration rupture derecesinin kaba kuvvet algoritma ve sezgisel algoritma sonuçları Tablo 5.2 de görüldüğü gibi yüksek oranda aynı çıkmıştır. Özellikle graf yoğunluğu arttıkça hata payı bazı veri setleri için düşmüştür. Buradan sezgisel algoritma ile yapılan ortalama alt agglomeration rupture derecesi hesaplamaların graf yoğunluğu arttıkça güvenilirliğinin arttığı görülmüştür.

Tablo 5.3: Orta boyutlu graflar üzerinde hesaplamalı deneyler.

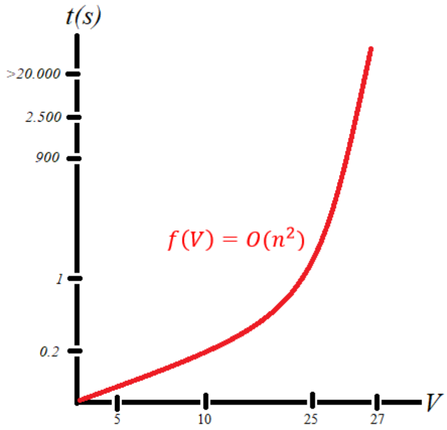
	20%			30%			40%			50%		
|V|	ARD	ALARD	t(s)	ARD	ALARD	t(s)	ARD	ALARD	t(s)	ARD	ALARD	t(s)
100	-33	-4400/100	3.8	-36	-4630/100	3.6	-31	-4203/100	2.9	-26	-3712/100	3.2
110	-42	-5809/110	4.7	-42	-5788/110	4.3	-37	-5293/110	3.1	-33	-4580/110	3.0
120	-48	-7075/120	5.5	-48	-7022/120	5.2	-42	-6346/120	4.0	-35	-5565/120	3.4
130	-51	-8399/130	7.3	-50	-8379/130	6.6	-43	-7652/130	5.0	-33	-6610/130	7.7
140	-61	-9972/140	8.8	-59	-9966/140	8.4	-48	-8990/140	8.8	-39	-7793/140	5.9
150	-72	-12293/150	13.0	-65	-11686/150	10.1	-60	-10548/150	9.7	-46	-9003/150	6.6
160	-73	-13783/160	15.4	-69	-13504/160	12.5	-60	-12125/160	10.9	-47	-10418/160	6.7
170	-82	-16071/170	19.7	-71	-15485/170	18.3	-66	-13863/170	11.2	-53	-11775/170	7.8
180	-79	-18034/180	22.6	-72	-17357/180	18.0	-68	-15719/180	12.9	-57	-13406/180	8.9
190	-95	-20661/190	28.8	-88	-19728/190	23.8	-76	-17706/190	17.6	-57	-15129/190	11.8
200	-101	-23233/200	35.3	-96	-22132/200	31.2	-80	-19638/200	20.2	-66	-16858/200	13.0

Sezgisel algoritma sonuçlarına göre Tablo 5.3 ve 5.4 da verilen orta ölçekli graflarda yapılan hesaplamalarda, grafların tepe sayısı arttıkça ARD ve ALARD değerlerinin düştüğü, grafın yoğunluğu arttıkça da bu değerlerin yükseldiği görülmüştür.

Tablo 5.4: Orta boyutlu graflar üzerinde hesaplamalı deneyler.

	60%			70%			80%			90%		
|V|	ARD	ALARD	t(s)	ARD	ALARD	t(s)	ARD	ALARD	t(s)	ARD	ALARD	t(s)
100	-19	-3033/100	2.1	-11	-4630/100	1.8	-8	-1536/100	2.7	-2	-786/100	1.7
110	-22	-3733/110	3.0	-16	-2812/110	2.2	-10	-1890/110	1.6	-4	-936/110	1.7
120	-26	-4553/120	2.6	-18	-3425/120	2.5	-9	-2279/120	1.9	-3	-1133/120	1.3
130	-27	-5380/130	3.4	-21	-4042/130	2.9	-11	-2692/130	1.9	-3	-1319/130	1.9
140	-29	-6356/140	3.8	-20	-4764/140	2.8	-10	-3148/140	2.6	-3	-1577/140	1.6
150	-36	-7397/150	4.8	-23	-5547/150	3.1	-14	-3657/150	2.3	-4	-1808/150	1.6
160	-36	-8423/160	5.7	-26	-6390/160	3.8	-16	-4207/160	1.7	-5	-2104/160	2.7
170	-40	-9559/170	4.9	-26	-7228/170	3.9	-15	-4789/170	2.2	-7	-2354/170	2.4
180	-42	-10818/180	6.6	-30	-8204/180	4.2	-16	-5407/180	2.7	-4	-2677/180	2.3
190	-44	-12186/190	7.0	-34	-9202/190	4.7	-18	-6083/190	3.7	-7	-2965/190	2.7
200	-54	-13659/200	8.9	-36	-10267/200	5.9	-20	-6819/200	4.5	-7	-3318/200	4.0

Sezgisel algoritmalarımızın karmaşıklığı $O(n^2)$ olduğu için Şekil 5.1 deki gibi bir karmaşıklık grafiği oluşmaktadır. Kaba kuvvet algoritması ile 24-27 tepeye kadar grafların ARD ve ALARD değerlerini uzun sürelerde hesaplarken, tepe sayısı 27 tepeden fazla olan graflarda bu süre çok daha fazla arttığı için RAM bellek aşırı yüklenmiştir. Daha iyi özelliklere sahip bilgisayarlarda ya da daha az kaynak kullanan *C* gibi programlama dillerinde yazılmış algoritma koştuğunda tepe sayısı artabilir fakat bu yöntemler de bir noktaya kadar sonuç verecektir. Bu nedenle sezgisel algoritmalar ile NP-Hard problemleri çözmek kaçınılmazdır.



Şekil 5.4: Sezgisel algoritma zaman karmaşıklığı.

Kaba kuvvet algoritma ile iki haneli tepelerde onlarca dakikada süren hesaplamalar yapılmışken, sezgisel algoritma ile üç haneli tepeler saniyeler içerisinde hesaplanmıştır. 27 tepeye kadar ARD - ALARD değerlerini kaba kuvvet ve sezgisel algoritma ile hesaplayıp yapılan karşılaştırma da hata payının hemen hemen hiç olmadığı, olanlarında ihmal edilebilir eşiğin altında olduğu hesaplamalı testlerde görülmüştür. Bu nedenle orta ve yüksek boyutlu graflar için elde edilen sonuçlara güvenilebileceği hesaplamalı testlerde görülmüştür.

Bu veri setinde genel olarak ARD ve ALARD değerlerinin rupture derecesi değerlerine göre daha yüksek çıktıkları Tablo 5.1, Tablo 5.2, Tablo 5.3 ve Tablo 5.4' de gözlemlenmiştir. Aynı tablolarda, ALARD değerlerinin ise ARD değerlerine göre daha düşük çıktığı gözlemlenmiştir. Bölüm 4' de iyi bilinen graflar için hesaplamış olduğumuz ARD ve ALARD değerlerinin $(-2, +\infty)$ aralığında olduğu görülmüştür.

6. SONUÇ VE ÖNERİLER

Bu tezde graflarda agglomeration (yığılma) bazlı rupture derecesi ele alınmıştır. Agglomeration rupture derecesi $r^{agg}(G)$ ve ortalama alt agglomeration rupture derecesi $r_{av}^{agg}(G)$ tanımlanıp araştırılmıştır, ardından bu değerler iyi bilinen graf aileleri için hesaplanmıştır. Bazı graflarda bu iki parametrenin ayırt edilebilirliği gösterilmiştir. Tezde tepe sayısı, ayrıt sayısı, bağlantılılık sayısı, baskınlık sayısı ve rupture dereceleri eşit olan iki graf ele alınarak agglomeration rupture derecesinin farklı çıkmasıyla bu parametrenin ayırt edici özelliğine değinilmiştir. Aynı şekilde tepe sayısı, ayrıt sayısı, bağlantılılık sayısı, baskınlık sayısı, rupture dereceleri ile birlikte agglomeration rupture dereceleri aynı olan iki farklı grafın ortalama alt agglomeration rupture derecelerinin farklı çıkmasıyla bu parametrenin de ayırt edici özelliğine değinilmiştir. Bazı durumlarda ARD ve ALARD değerlerinin de eşit olduğu farklı ağ modelleri mevcut olabilir. Bu durumda daha farklı zedelenebilirlik parametre değerleri hesaplanarak hangi modelin daha kararlı yapıda olduğu saptanabilir. Bu sonuçlar bize bir ağ tasarlarken, topolojinin zedelenebilirlik parametreleri göz önünde bulundurularak en kararlı olan topolojiyi referans alarak tasarlama gerekliliğini göstermiştir.

Son olarak, her tepe için daha düşük agglomeration rupture derecesi $r_{v_k}^{agg}(G)$ kümesini ve ayrıca herhangi bir G grafı için $r^{agg}(G)$ ile $r_{av}^{agg}(G)$ değerlerini bulmak için polinom zamanlı sezgisel bir algoritma önerilmiştir. Daha sonra hesaplamalı deneylerin sonuçlarını 200'e kadar tepeli graflar üzerinde sunulmuştur. 27 tepeye kadar kaba kuvvet algoritma ile bulunan değerler ve sezgisel algoritma ile aralarında ki hata payı da verilmiştir. Fakat 27 tepeden sonra kaba kuvvet algoritma karmaşıklığından dolayı çok uzun süreler tüketmeye başlamıştır ve bu nedenle ilerletilememiştir. Sonuçlar, önerilen sezgisel algoritmanın, belirli bir G grafının$r^{agg}(G)$ve $r_{av}^{agg}(G)$değerlerini verimli bir şekilde hesapladığını göstermektedir. Grafların diğer agglomeration (yığılma) tabanlı graf parametrelerini hesaplamak için çeşitli sezgisel yöntemlerin geliştirilmesi, bu alandaki gelecekte çalışılabilecek konulardan olabileceği önerilmektedir.

7. KAYNAKLAR

Aslan, E., “Weak-rupture degree of graphs”, Int. Journal of Foundations of Computer Science, 27 (6), 725–738, (2016).

Aslan, E., “Edge-rupture degree of a graph”, Dynamics of Continuous, Discrete and Impulsive Systems Series B: Applications and Algorithms, 22 (2), 155–161, (2015).

Aytac, A. and Aksu, H., “Some results for the rupture degree”, International Journal of Foundations of Computer Science, 24 (8), 1329–1338, (2013).

Aytac, A. and Aksu, H., “The rupture degree of some graphs”, Mathematica Balkanica, 24 (1-2), 85–101, (2010).

Aytac, A. and Odabas, Z. N., “Computing the rupture degree in composite graphs”, International Journal of Foundations of Computer Science, 21 (3), 311– 319, (2010).

Aytac, A. and Turaci, T., “Vertex vulnerability parameter of gear graphs”, International Journal of Foundations of Computer Science, 22 (5), 1187–1195, (2011).

Aytac, V. and Turaci, T., “Relationships between vertex attack tolerance and other vulnerability parameters”, RAIRO - Theoretical Informatics and Applications, 51 (1), 17–27, (2017).

Bacak-Turan, G. and Oz, E., “Neighbor rupture degree of transformation graphs Gxy”, International Journal of Foundations of Computer Science, 28 (4), 335–355, (2017).

Bagga, K. S., Beineke, L. W., Goddard, W. D., Lipman, M. J. and Pippert, R. E., “A survey of integrity”, Discrete Applied Mathematics, 37-38, 13–28, (1992).

Berberler, Z. N., Yıldırım, H. İ., Iltuzer T. and Tunc I., “Agglomeration-Based Node Importance Analysis in Wheel-Type Networks”, International Journal of Foundations of Computer Science, 32 (3), 26–288, (2021).

Berberler, M. E. andBerberler, Z. N., “Measuring the vulnerability in networks: a heuristic approach”, Ars Combinatoria, 135, 3–15, (2017).

Chvatal, V., “Tough graphs and Hamiltonian circuits”, Discrete Mathematics, 5 (3), 215–228, (1973).

Cozzens, M., Moazzami, D., and Stueckle, S., “The tenacity of a graph”, 17th Int. Conf. Theory and Applications of Graphs, Wiley, New York, 1111-1112 (1995).

Durgut, R., Kutucu, H. and Turacı, T., “Global distribution center number of some graphs and an algorithm”, RAIRO-Operations Research, 53 (4), 1217–1227, (2019).

Durgut, R., Turacı, T. and Kutucu, H., “A heuristic algorithm to find rupture degree in graphs”, Turk. J. Elec. Eng. & Comp Sci, 27, 3433–3441, (2019).

Frank, H. and Frisch, I. T., “Analysis and design of survivable networks”, IEEE Transactions on Communications Technology, 18 (5), 501–519, (1970).

Henning, M. A., “Trees with Equal Average Domination and Independent Domination Numbers”, Ars Combinatoria, 71, 305–318, (2004).

Jung, H. A., “On maximal circuits infinite graphs”, Annals of Discrete Mathematics, 3, 129–144, (1978).

Kunt, A. A. and Berberler, Z. N., “Efficient identification of node importance based on agglomeration in cycle-related networks”, International Journal of Foundations of Computer Science, 31 (7), (2020).

Kirlangic, A. and Bacak-Turan, G., “On the rupture degree of a graph”, Neural Network World, 22 (1), 39–51, (2012).

Kurkcu, O. K. and Aslan, E., “A comparison between edge neighbor rupture degree and edge scattering number in graphs”, International Journal of Foundations of Computer Science 29 (7), 1119–1142, (2018).

Mishkovski, I., Biey, M. and Kocarev, L., “Vulnerability of complex networks”, Communications in Nonlinear Science and Numerical Simulation, 16 (1), 341–349, (2011).

Li, Y., Zhang, S. and Li, X., “Rupture degree of graphs”, International Journal of Computer Mathematics, 82 (7), 793–803, (2005).

Li, Y., “The rupture degree of trees”. International Journal of Computer Mathematics, 85 (11), 1629–1635, (2008).

Li, F., “Isolated rupture degree of trees and gear graphs”, Neural Network World, 25 (3), 287–300, (2015).

Li, F. and Li, X, “Computing the rupture degrees of graphs”. 7th International Symposium on Parallel Architectures, Algorithms and Networks, Hong Kong-China, 368–373, (2004).

Odabas, Z. N. and Aytac, A., “Rupture degree and middle graphs”, Comptes Rendus de Lacademie Bulgare des Sciences, 65 (3), 315–322, (2010).

Tan, Y. J., Jun, W. and Deng, H. Z., “Evaluation method for node importance based on node contraction in complex networks”, Systems Engineering-Theory & Practice, 11 (11), 79–83, (2006).

Taylor, P., “What Ever Happened to Those Bridges?”, (02 July 2024), https://web.archive.org/web/20120319074335/http://www.amt.canberra.edu.au/koenigs.html, (2000).

Turacı, T. and Aytac, A., “Combining the Concepts of Residual and Domination in Graphs”, Fundamenta Informaticae, 166 (4), 379–392, (2019).

Turacı, T., “On combining the methods of link residual and domination in networks”, Fundamenta Informaticae, 174 (1), 43–59, (2020).

Turacı, T. and Aslan, E., “The average lower reinforcement number a graph”, RAIROTheor. Inf. Appl, 50 (2), 135–144, (2016).

Turacı, T., “On the average lower bondage number a graph”, RAIRO-Operations Research, 50 (4-5), 1003–1012, (2016).

West, B.D., “Introduction to Graph Theory”. 2nd ed. Urbana, NJ, USA: Prentice Hall, (2001).

EKLER

8. EKLER

EK A

```
void function Process(graph_parameter mainGraph, node_array_parameter arrayNode){

ruptures[] ← Ø

for i ← 0 to arrayNode's length{

        ruptures[i] ← Agglomeration(mainGraph, arrayNode[i])}

Integer ARD ← largest value of array ruptures

Double ALARD ← sum of all ruptures array elements / arrayNode's length

} # end function


Integer function Agglomeration(graph_parameter mainGraph, node_parameter Node) {

Graph ← mainGraph  # Cloning the master graph to avoid corruptions.

neighbors ← neighboring nodes corresponding to each index.

aggCluster ← neighbors[Node]          # Finding the neighbors of the node.

for i ← 0 to aggCluster's length{      # aggCluster nodes find their neighbors.

        temp[i] ← neighbors[aggCluster[i]]

        for j ← 0 to  temp's length{

                if temp[i] isn't equal to Node{

                        add temp[j] to neighbors }}}

add Node to aggCluster

sort aggCluster by contents from largest to smallest

for i ← 0 to aggCluster's length{      # Reset row and column.

        for j ← 0 to Graph's length{

                Graph[j][aggCluster[i]] ← 0
```

```
                Graph[aggCluster[i]][j] ← 0 }}
components[][] ← newly formed graph sets  # add new graphs.
for i ← 0 to length of components{ # deleting the neighborhood from the components array.
        for j ← 0 to length of components[i]{
        if aggCluster contains components[i][j]{
                components[i][0] ← Ø }}}
for i ← 0 to length of components{
        if the length of components[i] is 0{
                remove the i. variable from components }}

# Creating tagged component ↓
labeledComponents[][] ← Ø
labeledComponents ← components

# new tags ↓
Integer tag_number ← 0
for i ← 0 to length of labeledComponents{
        for j ← 0 to length of labeledComponents[i]{
                tag_number ← tag_number + 1
                labeledComponents[i][j] ← tag_number }}

# remove if empty ↓
for i ← 0 to length of labeledComponents{
        if the length of labeledComponents[i] is 0{
                remove the i. variable from labeledComponents
                remove the i. variable from components }}
```

```
# create new graph ↓
Integer value ← Graph's length – aggCluster's length + 1
newGraph[value][value] ← Ø   # newGraph is the matrix with value*value length.
for i ← 0 to length of labeledComponents{
        for j ← 0 to length of labeledComponents[i]{
                if neighbors contains components[i][j]{
                        newGraph[0][labeledComponents[I][j]] ← 1
                        newGraph[labeledComponents[I][j]][0] ← 1 }}}
for i ← 0 to length of components{
        for j ← 0 to length of components[i]{
                for k ← j to length of components[i]{
                Integer a ← Graph[components[i][j]][components[i][k]]
                newGraph[labeledComponents[i][j]][labeledComponents[i][k]]←a
                Integer b ← Graph[components[i][k]][components[i][j]]
                newGraph[labeledComponents[i][k]][labeledComponents[i][j]]←b
}}}
return function Rupture(newGraph)   # Branched into the heuristic rupture algorithm.
} # end function
```