**Universitatea Politehnica Bucuresti**

**Facultatea de Electronica, Telecomunicatii si Tehnologia Informatiei**

**Algoritmi de combatere a congestiei**

 **Profesor: Stefan Stancescu**

Student: Toma Oana Madalina

An universitar: 2013-2014

Cuprins

[1. Introducere 4](#_Toc370943939)

[1.1. Congestie 4](#_Toc370943940)

[2. Algoritmi de combatere a congestiei 5](#_Toc370943941)

[2.1 Clasificare 5](#_Toc370943942)

[3. Algoritmi bazati pe AQM (Active Queue Management) 6](#_Toc370943943)

[3.1. Introducere 6](#_Toc370943944)

[3.2. Algoritmul RED 6](#_Toc370943945)

[3.3. Algoritmul GRED 7](#_Toc370943946)

[3.4. ARED (Adaptive RED) 7](#_Toc370943947)

[3.5. NLRED (Nonlinear RED) 8](#_Toc370943948)

[3.6. DRED (Dynamic RED) 8](#_Toc370943949)

[3.7. IRED (Improved RED) 9](#_Toc370943950)

[3.8. RDRED (Real-time Dynamic RED) 9](#_Toc370943951)

[4. Compararea algoritmilor 11](#_Toc370943952)

[4.1. Scenariu de simulare pentru configuratia retelei 11](#_Toc370943953)

[4.2. Performante pentru incarcari de trafic diferite 11](#_Toc370943954)

[4.3. Performantele in conditii de trafic dinamic 13](#_Toc370943955)

[5. Concluzii 14](#_Toc370943956)

[6. Bibliografie 15](#_Toc370943957)

**Lista figurilor**

[Figure 1. Scenariu congestive 4](file:///C%3A%5CUsers%5Cmada%5CDesktop%5CAlgoritmi%20de%20combatere%20a%20congestiei.docx#_Toc370943958)

[Figure 2. Controlul congestiei in bucla inchisa. Algoritmul RED 5](#_Toc370943959)

[Figure 3. Algoritmul Red aplicat buffer-ului unui singur router 6](#_Toc370943960)

[Figure 4. Algoritmii GRED si ARED pentru buffer-ul unui singur router 7](#_Toc370943961)

[Figure 5. Functia de pierdere a pachetelor pentru DRED 7](#_Toc370943962)

[Figure 6. Functia de pierdere a pachetelor pentru NLRED 8](#_Toc370943963)

[Figure 7. Functia de pierdere a pachetelor pentru DRED 8](#_Toc370943964)

[Figure 8. Evolutia lungimii cozii de asteptare in cazul congestiei usoare 12](#_Toc370943965)

[Figure 9. Evolutia lungimii cozii de asteptare pentru congestie masiva 12](#_Toc370943966)

[Figure 10. Evolutia lungimii cozii in conditii de trafic dinamic 13](#_Toc370943967)

# Introducere

## Congestie

“Congestia este procesul de pierdere a pachetelor, transmise prin retea, din cauza lipsei de spatiu in buffere-le de stocare a concentratorilor instalati in reteaua respectiva. Presupun ca avem un segment de retea ca in figura de mai jos:



Figure . Scenariu congestive [1]

 Pot usor observa ca suma pachetelor care vin de la cele 2 calculatoare este mai mare decat capacitatea de transfer a cablului dintre router si calculatorul destinatie. Unele pachete ce sosesc la router, din cauza ca nu vor putea fi transmise spre destinatie, vor fi stocate in bufferele routerului. Aceste pachete sunt din ce in ce mai multe, pana cand apare fenomenul de congestie.

Controlul congestiei este un proces foarte complex care si-a gasit o rezolvare cat de cat multumitoare in ziua de azi. Dar, acest subiect este din ce in ce mai des abordat in cercetarile din informatica din cauza aglomerarii din ce in ce mai mari a retelelor de calculatoare. Putem extinde problema congestiei chiar la nivelul general al unei retele de telecomunicatii. Aspectul care complica si mai mult mecanismele de control al congestiei este acela ca nu se poate localiza pe un nivel particular al stivei de protocoale. Algoritmii de control al congestiei pot fi implementati atat la nivelul routerelor de retea cat si la nivelul protocoalelor de transport implementate in softul de retea al dispozitivelor comunicante. O solutie exclusiva, bazata doar pe controlul unui singur nivel nu duce la o rezolvare multumitoare. De aceea se cauta solutii de compromis intre cele doua nivele.

      Mecanismele pentru controlul congestiei sunt foarte diverse, motiv pentru care este foarte dificila realizarea unei clasificari. Totusi voi incerca sa prezint, mai jos, trei dintre cele mai importante aspecte ale mecanismelor de control al congestiei, ceea ce imi permite si o oarecare clasificare a acestora.” [1]

# Algoritmi de combatere a congestiei

## Clasificare

Conform [24] algoritmii de combatere a congestiei se clasifica in:





Figure . Controlul congestiei in bucla inchisa. Algoritmul RED

Din cate se poate observa din figura, algoritmii closed loop incearca sa puna toate pachetele in coada inainte de a calcula probabilitatea de pierdere. Spre deosebire de acestia, algoritmii open loop mai intai calculeaza probabilitatea de pierdere a pachetelor si apoi iau decizia de a le pune sau nu in coada.

Conform acestei clasificari algoritmii AQM fac parte din algoritmii closed-loop.

# Algoritmi bazati pe AQM (Active Queue Management)

## 3.1. Introducere

Exista mai multi algoritmi de combatere a congestiei care se bazeaza pe AQM. Cei mai importanti dintre ei sunt: RED (Random Early Detection), GRED (Gentle Random Early Detection) si AGRED (Adaptive Gentle Random Early Detection).

## 3.2. Algoritmul RED

Algoritmul RED (Random Early Detection) detecteaza inceputul congestiei la buffer-ul unui router in stagii preliminare. RED este format din doi algoritmi separate. Primul algoritm este pentru calculul lungimii medii a cozii de asteptare (aql-average queue length) care este calculata pe baza formulei urmatoare:

qavg(t)=(a-wq) x qavg(t-1)+wq x q(t)

unde qavg este lungimea medie a cozii si q este dimensiunea instantanee a cozii si wq$\in $[0, 1] este parametrul de filtrare.

Cel de-al doilea algoritm se foloseste pentru calculul probabilitatii de marcare a unui pachet care defineste cat de repede se pierd pachete la nivelul curent al congestiei (S. Floyd & Jacobson, 1993).

$$p\left(t\right)=\left\{\begin{array}{c}\begin{array}{c}0 cand q\_{avg}\left(t\right)<min\_{th}\\\frac{q\_{avg}\left(t\right)-min\_{th}}{max\_{th}-min\_{th}} × p\_{max} cand min\_{th}\leq q\_{avg}\left(t\right)\leq max\_{th}\end{array}\\q cand max\_{th}<q\_{avg}(t)\end{array}\right.$$

Unde pmax este probabilitatea maxima de pierdere a pachetelor

Algoritmul RED decide sa piarda un pachet prin compararea p(t)-ului calculat cu cele doua praguri, care sunt pragul minim si pragul maxim si sunt reprezentate in figura 2. Daca p(t) este mai mic decat pragul minim niciun pachet nu va fi pierdut si congestia nu se produce. Se continua adaugarea pachetelor in coada, daca p(t) este mai mare decat pragul maxim, se produce o congestive severa si fiecare pachet ce ajunge la router va fi pierdut cu p=1. Daca p(t) este intre pragul maxim si minim algoritmul RED trebuie sa calculeze qavg(t) pentru a adauga in coada pachetul sau pentru a-l pierde. 

Figure . Algoritmul Red aplicat buffer-ului unui singur router [2]

## 3.3. Algoritmul GRED

GRED actioneaza aproape la fel ca algoritmul RED dar principala diferenta intre cei doi este in asezarea optima a parametrilor pentru a avea o mai buna performanta in raport cu pierderile de pachete si incarcarea retelei. In GRED a fost introdus inca un parametru, Double max threshold, care este ilustrat in figura 4.



Figure . Algoritmii GRED si ARED pentru buffer-ul unui singur router [2]

 Algoritmul AGRED incearca sa rezolve problema pierderilor mari de pachete pe care o are GRED. Aceasta problema este rezolvata prin modificarea probabilitatii initiale de pierdere a pachetelor (pinit) pentru situatia in care p(t) se afla intre pragul maxim si Double max threshold.

$$p\_{max}\left(t\right)=p\_{max}\left(t-1\right)+η ×[q\_{avg}\left(t\right)-q^{\*}]$$



Figure . Functia de pierdere a pachetelor pentru DRED [3]

## 3.4. ARED (Adaptive RED)

ARED rezolva problema parametrizarii impuse de RED si creaza o metoda automata de alegere a parametrilor, fata de algoritmul RED original. In particular, principala modificare facuta in algoritmul ARED este asa-numita tune the parameter

$$p\_{max}=\left\{\begin{array}{c}p\_{max}\left(t-1\right)+α cand q\_{avg}>q^{\*}, p\_{max}\leq 0.5\\p\_{max}\left(t-1\right)xβ cand q\_{avg}<q^{\*}, p\_{max}\geq 0.01\end{array}\right.$$

Unde α este incremental si β este factorul de descrestere.

## 3.5. NLRED (Nonlinear RED)

Spre deosebire de ARED, NLRED [4] se concentreaza pe functia de pierdere a pachetelor, pentru ca algoritmul RED are performante scazute si are o functie de pierdere a pachetelor partial liniara. Cand incarcarea traficului este scazuta, RED actioneaza agresiv. Cand incarcarea traficului este mare si parametrii algoritmului RED sunt setati, incarcarea este scazuta. NLRED [4] schimba functia de pierdere a pachetelor, care era liniara, intr-o functie de gradul al doilea, in care probabilitatea de pierdere a pachetelor este resetata la $p\_{max}^{'}=1.5×p\_{max}$.

Noul comportament al functiei de pierdere a pachetelor din NLRED este descris prin:

$$p\left(t\right)=\left\{\begin{array}{c}0, cand q\_{avg}\left(t\right)<min\_{th}\\\left(\frac{q\_{avg}\left(t\right)-min\_{th}}{max\_{th}-min\_{th}}\right)^{2}×p\_{max}^{'}, cand min\_{th}\leq q\_{avg}\left(t\right)\leq max\_{th}\\1, daca max\_{th}<q\_{avg}(t)\end{array}\right.$$



Figure . Functia de pierdere a pachetelor pentru NLRED [3]

## 3.6. DRED (Dynamic RED)

In [5] a fost propus algoritmul DRED. Prima imbunatatire a lui DRED este utilizarea probabilitatii neliniare de pierdere a pachetelor in locul functiei liniare. Noua functie [5] a probabilitatii de pierdere a pachetelor este ilustrata in figura 7.



Figure . Functia de pierdere a pachetelor pentru DRED [3]

 Cea de-a doua imbunatatire a DRED este folosirea parametrului cheie pmax ales automat. Regula dupa care este ales [5] este data de formula:

$$p\_{max}\left(t\right)=p\_{max}\left(t-1\right)+η ×\left[q\_{avg}\left(t\right)-q^{\*}\right]$$

Unde η este dimensiunea pasului potrivit.

## 3.7. IRED (Improved RED)

 Algoritmul IRED [6] rezolva problemele algoritmului RED printr-o metoda foarte simpla, care este si foarte eficenta si nu are complexitate mare. Se observa ca incarcarea traficului si probabilitatea maxima de marcare a pachetelor pmax au un mare impact in echilibrarea cozilor algoritmului RED. IRED stabilizeaza lungimea cozii de asteptare, in jurul unei valori cunoscute, independente de incarcarea traficului, care pana acum era o problema pentru algoritmul RED. IRED [6] utilizeaza sensibilitatea lui RED pentru a crea un rol de adaptare proportional integral (PI) pentru parametrul cheie pmax. Rolul adaptiv PI este dat in formula:

$$p\_{max}\left(t\right)=p\_{max}\left(t-1\right)+η\_{1} ×\left\{\left[e\_{avg}\left(t\right)-e\_{avg}(t-1)\right]+η\_{2} × e\_{avg}(t)\right\}$$

Unde $ e\_{avg}\left(t\right)=q\_{avg}\left(t\right)-q^{\*}$ se noteaza eroarea de calcul a lungimii medie a cozii si $η\_{1}$, $η\_{2}$ > 0 sunt castigurile. Coeficientii proportionali si integrali sunt kp=$η\_{1}$ si ki=$η\_{2}×$ $η\_{1}$. Termenul proportional este $e\_{avg}\left(t\right)$, care poate imbunatati raspunsul sistemului si termenul integral$ \sum\_{}^{}e\_{avg}\left(t\right)$, care este eroarea de calcul a lungimii medie a cozii.

 Aditional, IRED are un rol de auto-tuning pentru dimensiunea pasului $η\_{1}$ pentru a se adapta conditiilor unei retele dinamice.

$$η\_{1}\left(t\right)=η\_{1}\left(t-1\right)+ η\_{3}× \left|e\_{avg}\right|^{ρ}×sgn(e\_{avg})$$

Unde $η\_{3}$, $ρ$>0 sunt parametrii ajustabili. $\left|e\_{avg}\right|^{ρ}$ masoara valoarea deviatiei lungimii medie a cozii. Problema regulii auto-tuning a IRED poate imbunatatii adaptabilitatea si robustetea in retelele dinamice.

## 3.8. RDRED (Real-time Dynamic RED)

Cele mai multe variante ale algoritmului RED inca mai utilizeaza lungimea medie a cozii ca masura a congestiei, dar acest lucru nu poate reflecta rapid evolutia lungimii cozii. Pe baza cercetarilor anterioare [5], s-a propus un sistem de control al congestiei pentru retelele cu time delay [23], retele in care s-a imbunatatit algoritmul RED cu ajutorul unui controller. Lungimea instantanee a cozii se foloseste ca masura a congestiei, in loc de lungimea medei a cozii, si acest mijloc de control al algoritmului RED modificat a fost denumit Real-Time Dynamic RED (RDRED).

 Pentru inceput, functia neliniara, probabilitate de pierdere a pachetelor, a algoritmului RDRED se calculeaza prin formula:

$$p\left(t\right)=\left\{\begin{array}{c}0, daca q\left(t\right)<min\_{th}\\\left[\frac{q\left(t\right)-min\_{th}}{q^{\*}-min\_{th}}\right]^{2}× \frac{q^{\*}-min\_{th}}{max\_{th}-min\_{th}}×p\_{max}, daca min\_{th}\leq q\left(t\right)\leq q^{\*}\\\left\{\left[\frac{q\left(t\right)-q^{\*}}{max\_{th}-q^{\*}}\right]^{{1}/{2}}×\frac{max\_{th}-q^{\*}}{max\_{th}-min\_{th}}+\frac{q^{\*}-min\_{th}}{max\_{th}-min\_{th}}\right\} ×p\_{max}, daca q^{\*}<q\left(t\right)\leq max\_{th}\\1, daca max\_{th}<q(t)\end{array}\right.$$

Observatii:

1. Se observa ca daca lungimea instantanee a cozii q(t) este mai mica decat valoarea asteptata q\*, deci, RDRED pierde pachete cu o probabilitate mai scazuta. Daca lungimea instantanee a cozii q(t) este mai mare decat valoarea asteptata q\*, RDRED pierde pachetele cu o probabilitate relativ mare. In concluzie, probabilitatea pierderii pachetelor este mica pentru congestia usoara, dar creste la congestie severa.

Probabilitatea maxima de pierdere a pachetelor pmax este realizata automat dupa formula:

$$p\_{max}\left(t\right)=p\_{max}\left(t-1\right)+η×\left[q\left(t\right)-q^{\*}\right]$$

Unde η este constanta castigului, cand q(t)-q\* este eroarea lungimii instantanee a cozii.

1. Se observa din rolul tuning din ecuatia anterioara, cand lungimea instantanee a cozii q(t) este mai mica decat target-ul q\*, probabilitatea de pierdere a pachetelor scade si lungimea instantanee a cozii q(t) creste pana la target-ul q\*. Cand lungimea instantanee a cozii q(t) este mai mare decat target-ul q\*, probabilitatea de pierdere a pachetelor creste si lungimea instantanee a cozii coboara sub target-ul q\*. Deci, acest rol este capabil sa stabilizeze lungimea instantanee a cozii in jurul target-ului.
2. Spre deosebire de algoritmul prezentat in [5], RDRED foloseste lungimea instantanee a cozii in locul lungimii medii a cozii pentru a imbunatati raspunsul sistemului.

# Compararea algoritmilor

## 4.1. Scenariu de simulare pentru configuratia retelei

Simularea s-a realizat cu o retea simpla bottleneck ce contine doua routere si N perechi de sender si noduri sink. Toate link-urile au fost configurate la 10Mbps si 10ms. Au fost implementati urmatorii algoritmi AQM pentru routere: RED, ARED, NLRED, DRED, IRED si RDRED. In tabelul 1 sunt date valorile parametrilor pentru toti algoritmii testati:

Table . Parametrii simularii [3]

|  |  |
| --- | --- |
| Algoritmi | Parametrii |
| RED | wq= 0.002; minth= 100; maxth= 300; pmax = 0.2 |
| ARED | wq= 0.002; minth= 100; maxth= 300; pmax = 0.2; target = 200; α = 0.01; β = 0.9 |
| NLRED | wq= 0.002; minth= 100; maxth= 300; p’max = 0.3 |
| DRED | wq= 0.002; minth= 100; maxth= 300; q\*=200; η= 1.0 x 10−7 |
| IRED | wq= 0.002; minth= 100; maxth= 300; η1= η2=1.0 x 10-3; η3=1.0 x 10-8; ρ=2 |
| RDRED | minth= 100; maxth= 300; q\*=200; η= 1.0 x 10−6 |

## Performante pentru incarcari de trafic diferite

In aceasta simulare traficul TCP variaza intre 100 si 300 pentru a initia conditii diferite ale congestiei. In figurile 8 si 9 este aratata evolutia lungimii cozii de asteptare. Tabelele 2 si 3 sumarizeaza performantele fiecarui algoritm. Din figura 8, doar algoritmul ARED sufera oscilatii foarte mari ale lungimii cozii de asteptare. Ceilalti algoritmi, cum sunt RED, NLRED, DRED, IRED si RDRED stabilizeaza lungimea cozii de asteptare cu mici oscilatii. Din figura 9, cand traficul TCP creste la 300, congestia este masiva si lungimea cozii de asteptare pentru RED si NLRED creste foarte mult. DRED, IRED si RDRED stabilizeaza lungimea cozii de asteptare in jurul valorii target, dar DRED raspunde destul de incet. RDRED atinge cu cea mai mica intarziere si cea mai mare rata de convergenta target-ul lungimii cozii.



Figure . Evolutia lungimii cozii de asteptare in cazul congestiei usoare[3]



Figure . Evolutia lungimii cozii de asteptare pentru congestie masiva[3]

Table . Valoarea medie a lungimii cozii de asteptare [3]

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Algoritmi | RED | ARED | NLRED | DRED | IRED | RDRED |
| TCP 100 | 226.84 | 206.01 | 229.49 | 200.85 | 200.17 | 200 |
| TCP 300 | 291.00 | 254.68 | 289.92 | 204.57 | 199.87 | 200.26 |

Table . Deviatia standard a lungimii cozii de asteptare [3]

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Algoritmi | RED | ARED | NLRED | DRED | IRED | RDRED |
| TCP 100 | 13.74 | 55.78 | 13.48 | 13.93 | 12.71 | 6.89 |
| TCP300 | 62.91 | 14.03 | 48.69 | 13.25 | 13.02 | 6.92 |

## 4.3. Performantele in conditii de trafic dinamic

 In cel de-al doilea scenariu s-a dorit studierea robustetei algoritmilor RED, ARED, NLRED, DRED, IRED si RDRED in conditii de trafic dinamic. Traficul TCP este setat la 100 la inceput. S-au adaugat doua grupuri de trafic, fiecare dintre ele consista in 100 de flow-uri TCP, ele sunt activate la 100s si 200s ramanand active pana la 300s, respectiv 400s. Evolutia lungimii cozii este ilustrata in figura 10. Lungimea cozii de asteptare dupa aplicarea algoritmilor RED, ARED si NLRED variaza cu schimbarea incarcarii traficului. Lungimile cozilor de asteptare oscileaza foarte mult, producand astfel intarzieri ale pachetelor. DRED, IRED si RDRED se comporta bine, chiar si la variatii bruste ale traficului. Lungimea cozii de asteptare a algoritmului RDRED atinge cea mai buna performanta.



Figure . Evolutia lungimii cozii in conditii de trafic dinamic [3]

# Concluzii

Din rezultatele simularii, se poate observa ca RED, ARED, NLRED nu se comporta foarte bine in conditii de incarcare variabila a retelei. Este greu de stabilizat lungimea cozii de asteptare pentru a avea o buna performanta la o gama variata de nivele de incarcare. DRED, IRED si RDRED pot stabiliza foarte bine lungimea cozii, de aici reiese o intarziere acceptabila. RDRED atinge cea mai buna performanta dintre toti algoritmii testati, pe tipurile de retele alese in simulare. IRED si RDRED sunt simplu de implementat si este foarte eficienta pentru a introduce lungimea instantanee a cozii de asteptare, ca masura a congestiei.

# Bibliografie

[1] <http://rast.orgfree.com/retele/retele_2002_07.html>

[2] Omid Seifaddini , Azizol Abdullah and Hamid Vosough “RED, GRED, AGRED CONGESTION CONTROL ALGORITHMS IN HETEROGENEOUS TRAFFIC TYPES “4th International Conference on Computing and Informatics, ICOCI 2013 28-30 August, 2013 Sarawak, Malaysia. Universiti Utara Malaysia

[3] Minjuan Cheng, Xiaoming Ma, Performance Evaluation of Queue Management Methods for Congestion Control, Journal of Information & Computational Science 9: 6 (2012) 1599–1608

[4] K. Zhou, K. L. Yeung, V. O. K. Li, Nonlinear RED: A simple yet efficient active queue management

scheme, Computer Networks, 50(18), 2006, 3784-3794

[5] M. Cheng, H. Wang, L. Yan, Dynamic RED: A modified random early detection, Journal of

Computational Information Systems, 7(14), 2011, 5243-5250

[6] H. Wang, Z. Ye, B. Wang, Using auto-tuning proportional integral probability to improve random

early detection, Proc. the 13th IEEE International Conference on Communication Technology

(ICCT), 2011, 1107-1111

[7] S. Kunniyur, R. Srikant, An adaptive virtual queue (AVQ) algorithm for active queue management,

IEEE/ACM Transactions on Networking, 12(2), 2004, 286-299

[8] S. Athuraliya, S. H. Low, V. H. Li, Q. Yin, REM: Active queue management, IEEE Network

Magazine, 15(3), 2001, 48-53

[9] H. Wang, C. Liao, Z. Tian, Effective adaptive virtual queue: A stabilising active queue management algorithm for improving responsiveness and robustness, IET Communications, 5(1), 2011,

99-109

[10] H. Wang, O. Li, C. Liao, Z. Tian, Design of adaptive real queue control algorithm supporting

TCP flows, Proc. the 35th Annual IEEE Conference on Local Computer Networks (LCN), 2010,

300-303

[11] F. Ren, C. Lin, B. Wei, A robust active queue management algorithm in large delay networks,

Computer Communications, 28(5), 2005, 485-493

[12] H. Wang, W. Wei, Y. Li, C. Liao, Y. Qiao, Z. Tian, Two-degree-of-freedom congestion control

strategy against time delay and disturbance, Proc. IEEE Global Communications Conference

(GLOBECOM), 2010, 1-5

[13] H. Wang, C. Liao, Z. Tian, Providing quality of service over time delay networks by efficient queue

management, Proc. the 36th Annual IEEE Conference on Local Computer Networks (LCN), 2011,

275-278

[14] R. K. Alasem, M. A. Hossain, I. Awan, Active queue management controller using Smith predictor

for time delay networks, Proc., IEEE International Conference on Networking, Sensing and Control

(ICNSC), 2007, 568-573

[15] C. K. Chen, T. L. Liao, J. J. Yan, Active queue management controller design for TCP communication networks: Variable structure control approach, Chaos, Solitons & Fractals, 40(1), 2009,

277-285

[16] H. Wang, Z. Tian, Q. Zhang, Sliding mode control with fuzzy reaching law for queue management in

the Internet, Proc. the 19th International Conference on Computer Communications and Networks

(ICCCN), 2010, 1-5

[17] H. Wang, Y. Li, B. Wang, Responsive AQM algorithm based on terminal sliding mode technique,

Proc. the 13th IEEE International Conference on Communication Technology (ICCT), 2011, 156-160

[18] F. Zheng, J. Nelson, An H1 approach to the controller design of AQM routers supporting TCP

flows, Automatica, 45(3), 2009, 757-763

 [19] C. K. Chen, Y. C. Hung, T. L. Liao, J. J. Yan, Design of robust active queue management

controllers for a class of TCP communication networks, Information Sciences, 177, 2007, 4059-4071

[20] H. Wang, Z. Tian, Q. Zhang, Self-tuning price-based congestion control supporting TCP networks,

Proc. the 19th International Conference on Computer Communications and Networks (ICCCN),

2010, 1-6

[21] H. Wang, Z. Tian, Intelligent price-based congestion control for communication networks, Proc.

the 18th IEEE International Workshop on Quality of Service (IWQoS), 2010, 1-5

[22] H. Wang, B. Wang, J. Chen, Z. Tian, Design and analysis of effective price for congestion control,

Proc. the 36th Annual IEEE Conference on Local Computer Networks (LCN), 2011, 510-518

[23] M. Cheng, H. Wang, Stabilizing random early detection method, Journal of Jilin University (Information Science Edition), 30, 2012

[24] A. Tanenbaum, Computer networks 5th edition, Practice Hall, USA, 2011

[25] T. Komatsu and A. Namatame, Effectiveness of close-loop congestion controls for DDoS attacks, 2008