۹مین کنفرانس بین المللی
مهندسی برق، کامپیوتر و مکانیک
The 9th International Conference on
Electrical,computer & mechanical engineering

تحلیل بهینه‌سازی مقاوم مسئله قرار گیری کنترل‌کننده در شبکه‌های مبتنی بر نرم‌افزار

محمد کاظمی، احمدرضا منتظرالقائم

۱- دانشجوی دکتری دانشکده مهندسی کامپیوتر دانشگاه اصفهان

۲- استادیار دانشکده مهندسی کامپیوتر دانشگاه اصفهان

Email: (mohammad.kazemi@eng.ui.ac.ir)

Email: (a.montazerolghaem@comp.ui.ac.ir)

چکیده

شبکه مبتنی بر نرم افزار (SDN) یک الگوی شبکه است که با جدا کردن سطح کنترل شبکه از سطح داده، دید متمرکزی از شبکه را ارائه می‌دهد. سطح کنترل کننده توزیع شده به همراه مزایای آن مشکل تعداد کنترل کننده‌های مورد نیاز و قرارگیری آنها در شبکه را به همراه دارد. این مشکل به عنوان مسئله قرار گیری کنترل کننده (CPP) نامیده می‌شود. در بسیاری از مطالعات انجام شده در این زمینه فرض شده است که اطلاعات شبکه مانند ظرفیت‌ها و نرخ ترافیک‌های تولیدی و غیره بصورت کامل در دسترس می‌باشد. در عمل به دلیل شرایط پویا، اطلاعات بدست آمده دارای عدم‌قطعیت می‌باشند. در مواجهه با این چالش عدم قطعیت، رویکردهای متفاوت وجود دارد ولی یک رویکرد مؤثر، تکنیک‌های بهینه‌سازی مقاوم بوده است. در بهینه سازی مقاوم، هدف، اخذ یک تصمیم امکان‌پذیر و بهینه برای بهینه سازی تابع هدف در بدترین حالت است. روش‌های مقاوم نیز متفاوت می‌باشد و در این مقاله از روش Bertsimas استفاده شده است. در این روش در واقع کنترلی در محافظه کاری عدم‌قطعیت دارد ولی جواب به بهینه نزدیکتر است. در ادامه مقاله روش مقاوم را در سناریوهای مختلف یعنی تأثیر در افزایش و کاهش در پارامترهایی مانند گاما (که نشان‌دهنده درجه محافظه‌کاری است) بررسی کرده و در آخر نیز طی یک سناریو مزیت روش مقاوم را نشان خواهیم داد که اگر ما از قبل پیش‌بینی برای انحرافی که اتفاق می‌افتد داشته باشیم، در زمان اجرای شبکه، مقدار واقعی تابع هدف مساوی یا حتی در مواردی کاهش خواهد داشت.

کلمات کلیدی: شبکه مبتنی بر نرم افزار (SDN)، مسئله قرار گیری کنترل کننده (CPP)، عدم‌قطعیت، بهینه‌سازی مقاوم

۱. مقدمه

پیشرفت‌های جدید در دنیای اطلاعات و ارتباطات مانند اینترنت اشیا و محاسبات ابری و دیگر موارد باعث رشد شدید ترافیک در شبکه می شود، در نتیجه شبکه‌ی سنتی با چالش‌هایی روبه رو شده و توان پاسخگویی به درخواست‌ها را ندارد. با توجه به رشد شبکه‌ها، تعداد عناصر و پیچیدگی شبکه‌ها افزایش پیدا کرده است، بنابراین از چالش‌های مهم، مدیریت شبکه و ارائه کیفیت خدمات[۱](QoS) است[1]. اخیراً با جداسازی سطح کنترل از سطح داده در شبکه‌های مبتنی بر نرم‌افزار[۲] و وجود یک کنترل‌کننده که

[۱] quality of service

[۲] software defined network

یک دید کلی از شبکه دارد، می‌توان مدیریت بهتری در شبکه و استفاده‌ی موثرتر از منابع شبکه در شبکه‌های نسل جدید[۳] (NGN) داشت[2]. در شبکه‌های مبتنی بر SDN، کنترل‌کننده‌ها نقش مهمی در مدیریت ترافیک شبکه و استفاده از منابع شبکه ایفا می کنند. با توجه به اینکه کنترل‌کننده مسئول مدیریت و اعمال تنظیمات در شبکه است بنابراین، خرابی کنترل‌کننده بر عملکرد شبکه تأثیر می‌گذارد. همچنین با بزرگ شدن شبکه، کنترل‌کننده باید مورد تغییراتی قرار بگیرد یا تعدادشان بیشتر شود، که در نتیجه باعث تغییر در معماری SDN می شود[3]. یک کنترل‌کننده به هر حال محدودیت‌هایی مانند: پردازش، حافظه، پهنای باند و غیره دارد و می‌تواند در هر کدام به یک نقطه گلوگاه تبدیل شود، بنابراین افزایش در تعداد کنترل‌کننده باعث افزایش کارایی، مقیاس پذیری، قابلیت اطمینان و کاهش تاخیر انتشار بین فورواردرها و کنترل‌کننده می‌شود. اما تعداد و مکان قرارگیری این کنترل‌کننده‌ها در شبکه‌های مبتنی بر SDN، باعث تاثیر در عملکرد شبکه می شود[4]. مسئله قرارگیری کنترل‌کننده[۴] (CPP) به این امور می‌پردازد که چه تعداد کنترل‌کننده استفاده شود و همچنین مکان آن‌ها در کجای شبکه باشد تا عملکرد، قابلیت اطمینان و کارایی شبکه‌های مبتنی بر SDN بهینه شود. تعداد منا سب کنترل‌کننده‌ها و قرار دادن آنها در مکان منا سب می تواند تاخیر شبکه را کاهش دهد و قرارگیری منا سب کنترل‌کننده‌ها نیز می‌تواند بر سایر مشکلات شبکه مانند سربار سطح کنترل، تحمل خطا و انعطاف‌پذیری تأثیر بگذارد[3]. با این حال، تعداد مناسب کنترل‌کننده‌ها همراه با قراردادن آن‌ها در جای مناسب در شبکه‌های مبتنی بر SDN یک مسئله NP-hard است[5].

تاکنون مقالات زیادی به موضوع قرارگیری کنترل‌کننده پرداخته‌اند و راه‌حل‌هایی برای CPP پیشنهاد داده‌اند که بیشتر این راه‌حل‌ها با هدف بهینه‌سازی تعداد و مکان کنترل‌کننده‌ها در شبکه های مبتنی بر SDN هستند. با این حال، بیشتر مقالات فرض می‌کنند که اطلاعات شبکه مانند ظرفیت کنترل‌کننده و یا ترافیک تولیدی هر سوییچ و موارد دیگر در شبکه ثابت است، در حالیکه اطلاعات شبکه در حال تغییر است. برای نمونه ترافیک شبکه به صورت پویا در زمان و مکان تغییر می‌کند. می‌توان در مورد زمان به این مثال اشاره کرد که در یک کسب‌وکار تعداد کاربران در دوره های مختلف می‌تواند متفاوت باشد. برای مکان نیز، سطح اقتصادی که در مناطق مختلف وجود دارد، بسیار متفاوت است که منجر به ترافیک شبکه پویا و غیرقابل پیش بینی شده است[6]. بنابراین مهم است که تاثیر ترافیک پویا بر روی قرارگیری کنترل‌کننده در نظر گرفته شود. به منظور مطالعه تأثیر ترافیک پویا بر قرارگیری کنترل‌کننده، در این مقاله از تئوری بهینه‌سازی مقاوم برای در نظر گرفتن پارامترهای غیرقطعی یا همان ترافیک پویا استفاده می‌کنیم و اثر آن بر مسئله مورد نظر را مورد بررسی قرار می‌دهیم. در قسمت شبیه‌سازی روش مقاوم مسئله را در سناریوهای مختلف بررسی کرده و در آخر نیز نشان خواهیم داد که اگر از قبل پیش‌بینی برای انحرافی که در حین اجرای شبکه اتفاق می‌افتد داشته باشیم، در نهایت ، مقدار واقعی تابع هدف بهبود پیدا خواهد کرد .

در ادامه مقاله به معرفی بهینه‌سازی مقاوم و انواع آن پرداخته و سپس مسئله را به کمک این تکنیک بهینه‌سازی، مدل می‌کنیم.

۲. انواع بهینه‌سازی مقاوم

[۳] next generation networks

[۴] controller placement problem

الگوی کلاسیک برنامه‌نویسی ریاضی به این صورت است که فرض می‌شود داده‌های ورودی کاملا مشخص هستند. با این حال در این روش از تأثیر عدم‌قطعیت داده در کیفیت و امکان‌پذیر بودن مسئله چشم‌پوشی شده است. قابل درک است که در عمل، داده‌ی ورودی می‌تواند مقادیری متفاوت از آن چه در برنامه فرض شده است، باشد. دراین صورت امکان دارد بعضی از قیدها نقض شوند و یا حتی جواب تابع هدف تغییر کند و ممکن است دیگر مقدار بهینه نباشد[7].

بنابراین نیاز است مدلی توسعه داده بشود که در آن عدم‌قطعیت داده مورد توجه قرار گیرد. در این راستا سه روش برای در نظر گرفتن عدم‌قطعیت پیشنهاد شده است :

۲. ۱. روش Soyster

در این روش Soyster یک مدل ریاضی خطی ارائه می‌دهد که تمامی داده‌ها متعلق به یک مجموعه‌ی محدب می‌باشند. در این روش فرض می‌شود تمامی داده‌های ورودی دارای عدم‌قطعیت می‌باشند. بنابراین این محافظه‌کارترین روش در برخورد با عدم‌قطعیت است. اما در واقعیت فقط تعدادی از داده‌ها امکان تغییر پیدا می‌کنند، در نتیجه تابع هدف در این روش کمی از تابع هدف واقعی یا اسمی فاصله می‌گیرد[7].

۲. ۲. روش Nemirovski و Ben-Tal

در این روش محافظه‌کاری کمتری نسبت به روش قبل در نظر گرفته شده است. در این روش یک مسئله خطی غیرقطعی ارائه شده است که داده‌هایی که عدم‌قطعیت دارند از یک مجموعه بیضوی خواهند آمد. در این روش نویسنده نشان می‌دهد که احتمال آنکه قید (i) ام نقض شود حداکثر یک عدد مشخصی خواهد بود[7].

۲. ۳. روش Bertsimas and Sim

در این روش نویسنده یک مسئله‌ی خطی ارائه می‌دهد که برتری روش Soyster را نیز حفظ می‌کند. این روش کنترل کاملی از نظر محافظه‌کاری دارد و می‌تواند بر اساس واقعیت یا فرض مسئله کم یا زیاد شود. در این مدل فرض می‌شود که تعداد مشخصی از ورودی‌ها دارای عدم‌قطعیت هستند. این تعداد مشخص که با عنوان پارامتر گاما شناخته می‌شود توسط طراح سیستم تعیین می‌شود. نزدیکتر کردن شرایط مسئله به واقعیت باعث نزدیکتر شدن به نتیجه یا همان تابع هدف واقعی می‌شود[7].

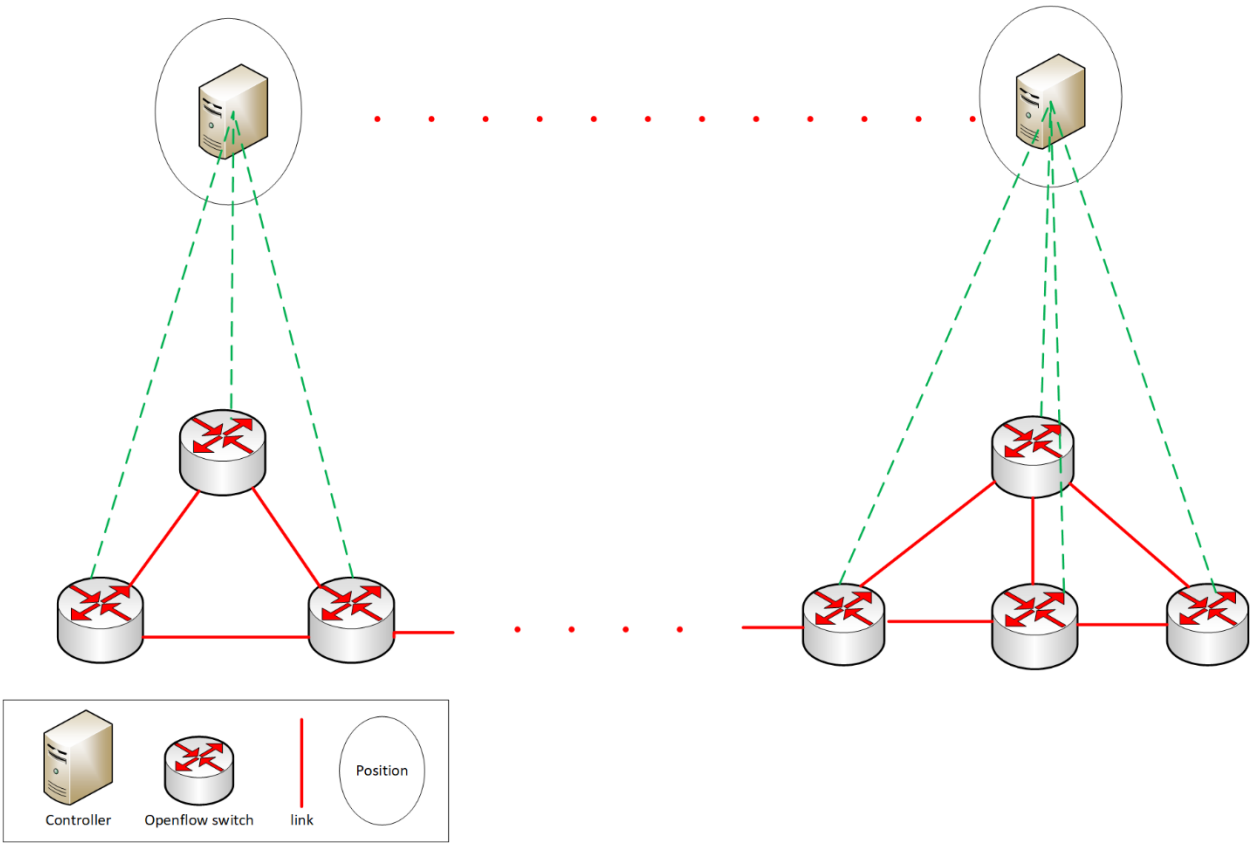
۲. ۴. مزیت روش Bertsimas and Sim

پارامتر گاما یا همان پارامتری که میزان محافظه‌کاری را مشخص می‌کند، توسط طراح سیستم قابل تنظیم است بنابراین مزیت این روش این است که طراح سیستم می‌تواند با مشخص کردن گاما هم محافظه‌کاری در مقابل عدم‌قطعیت را در نظر بگیرد و هم جوابی که بدست می‌آید به مقدار جواب بهینه‌ی اسمی نزدیکتر است. مزیت دیگر این روش این می‌باشد که اگر یک مسئله خطی به آن

بدهیم و عدم‌قطعیت را درنظر بگیریم، ساختار برنامه را حفظ کرده و باعث پیچیدگی بیشتر نمی‌شود. در این روش تضمین می‌دهد که اگر تغییرات عدم‌قطعیت کمتر یا مساوی گاما شود، جواب مسئله ممکن و شدنی خواهد بود. با این حال نشان می‌دهد که اگر تغییرات حتی بیشتر از گاما نیز بشود با احتمال بالایی باز هم جواب مقاوم‌سازی ممکن و شدنی خواهد بود.

۳. مدل سیستم، مفروضات و فرمول‌بندی مسأله

در این قسمت مطابق شکل (۱) ، فرض می‌شود که تعدادی سوییچ n تا در لایه داده قرار دارند. این لایه داده به چندین دامنه تقسیم شده ا ست، به این صورت که در هر دامنه تعدادی سوییچ قرار دارد و مدیریت و کنترل هر دامنه و سوییچ‌های موجود در آن بر عهده یک کنترل‌کننده می‌باشد. تعداد کنترل‌کننده‌ها در لایه کنترل m تا فرض می‌شود. همچنین تعدادی موقعیت نیز وجود دارد که باید کنترل‌کننده‌ها در آن قرار بگیرند. پارامترهایی که در این م سئله ا ستفاده می‌کنیم نیز در ادامه معرفی می شود: r نرخ ترافیکی که هر سوییچ تولید می‌کند را نشان می‌دهد. ظرفیت یا همان نرخ پردازشی هر کنترل‌کننده نیز μ در نظر گرفته می‌شود. ω و ρ نیز به ترتیب هزینه‌ی نصب هر کنترل‌کننده و هزینه‌ی توان به ازای هر بیت را نشان می‌دهند. علائم نیز در جدول (۱) آمده است.



شکل ۱ : مدل سیستم و توپولوژِی شبکه

جدول ۱: جدول علائم

علائم نشان‌دهنده	پارامتر مسئله
n	تعداد سوییچ
m	تعداد کنترل‌کننده
r	ترافیک تولیدی سوییچ
μ	ظرفیت کنترل‌کننده
ω	هزینه نصب کنترل‌کننده
ρ	هزینه‌ی توان به ازای هر بیت
x	متغیر باینری موقعیت
y	متغیر باینری سوییچ
a,B	متغیرهای دوگان
Γ	تعیین‌کننده میزان محافظه‌کاری

هدف از م سئله cpp در این مقاله (۱) کاهش هزینه جایابی و کاهش و ت ضمین تاخیر بین یک کنترل‌کننده و یک سوییچ می‌با شد. همچنین محدودیت های این م سئله به این صورت می‌با شد که (۳) در هر موقعیت فقط یک کنترل‌کننده باید قرار بگیرد و نیز (۴) هر سـوییچ نیز باید تحت کنترل و مدیریت فقط یک کنترل‌کننده باشـد. همچنین (۵) مجموع درخواسـت‌هایی که به یک کنترل‌کننده تخصیص داده می‌شود نباید از ظرفیت پردازشی آن کنترل‌کننده بیشتر باشد.

در این م سئله از دو متغیر نیز ا ستفاده می شود (۶) و (۷). x زمانیکه یک کنترل‌کننده در یک موقعیت م شخص قرار بگیرد، یک می شود و درغیر این‌صورت مقدار صفر خواهد گرفت. همچنین زمانیکه یک سوییچ تحت کنترل یک کنترل‌کننده قرار گرفته در موقعیت م شخص با شد متغیر y مقدار یک خواهد گرفت. میتوان به جای تابع هدف، یک سقف را درنظر گرفت و با کمینه کردن آن کف، مقدار تابع هدف را کمینه کرد. با توجه به توضیحات داده شده فرمول مسئله بصورت زیر می‌باشد:

$$\min \ t \qquad (۱)$$

$$\text{s.t.} \ \ \omega \sum_p x_{cp} + \rho \sum_s y_{sp} r_s \qquad (۲)$$

$$\sum_c x_{cp} \leq 1 \ , \forall_p \qquad (۳)$$

$$\sum_p y_{sp} = 1 \ , \forall_s \qquad (٤)$$

$$\sum_s y_{sp} \, r_s \leq \mu \, x_{cp} \ , \forall_p \qquad (٥)$$

$$x_{cp} \in \{0,1\} \ , \forall_c \, \forall_p \qquad (٦)$$

$$y_{sp} \in \{0,1\} \ , \forall_s \, \forall_p \qquad (٧)$$

در فرمول بالا ترافیک هر سوییچ r بصورت ثابت و قطعی در نظر گرفته شده است اما همانطور که اشاره شد در عمل ترافیک هر سوییچ بصورت پویا و در زمان متغیر ا ست. برای در نظر گرفتن عدم‌قطعیت ترافیک از راه‌حل بهینه سازی مقاوم و بصورت دقیقتر از روش Bertsimas and Sim استفاده می‌کنیم. بعد از تغییرات مربوطه فرمول مسئله با درنظر گرفتن عدم قطعیت بصورت زیر می‌باشد:

$$\min \ t \qquad (۱)$$

$$\text{s.t} \ \ \omega \sum_p x_{cp} + \rho \sum_s y_{sp} r_s \ + \sum_{s \epsilon J_p} B_{sp} + \Gamma_p a_{p_s} \qquad (۲)$$

$$\sum_c x_{cp} \leq 1 \qquad (۳)$$

$$\sum_p y_{sp} = 1 \qquad (٤)$$

$$\sum_s y_{sp} \, r_s + \sum_{s \epsilon J_p} B2_{sp} + \Gamma_p a2_p \leq \mu \, x_{cp} \qquad (٥)$$

$$x_{cp} \in \{0,1\} \ , \forall_c \, \forall_p \qquad (٦)$$

$$y_{sp} \in \{0,1\} \ , \forall_s \, \forall_p \qquad (٧)$$

$$a_p + B_{sp} \geq \hat{r}_s y_{sp} \ , \forall_{p, s \epsilon J_p} \qquad (٨)$$

$$B_{sp} \geq 0 \ , \forall_{s \epsilon J_p}, \forall_p \qquad (٩)$$

$$a_p \geq 0 \ , \forall_p \qquad (۱۰)$$

$$a2_p + B2_{sp} \geq \hat{r}_s y2_{sp} \ , \forall_{p, s \epsilon J_p} \qquad (۱۱)$$

$$B2_{sp} \geq 0 \ , \forall_{s \epsilon J_p}, \forall_p \qquad (۱۲)$$

$$a2_p \geq 0 \ , \forall_p \qquad (۱۳)$$

در این فرمول به ازای هر پارمتر غیر قطعی، یعنی r، از دو پارامتر دیگر به نام‌های a و B برای اعمال عدم قطعیت استفاده شده است (قیدهای (۸ تا ۱۳)). و همچنین از پارامتر Γ برای مشخص کردن میزان محافظه‌کاری استفاده شده است.

۴. شبیه‌سازی و تحلیل مسأله مقاوم

۴. ۱. تنظیمات شبیه‌سازی

با پیاده‌ســـازی دو معادله‌ی بالا در حل‌کننده‌ی یالمیپ (yalmip) که در نرم‌افزار متلب اجرا می‌شـــود، روش مقاوم را مورد بررســـی و مقایســـه قرار می‌دهیم. پارامترهایی در این دو معادله وجود دارند (مانند: مقدار نرخ تولیدی هر ســـویچ، مقدار ظرفیت هر کنترل‌کننده، مقدار گاما، مقدار هزینه‌ی نصـــب هر کنترل‌کننده و همچنین مقدار هزینه‌ی توان به ازای هر بیت) که تغییر در این پارامترها باعث تغییر درمقدار تابع هدف خواهد شد. وا ضح ا ست که با افزایش سه پارامتر نرخ تولیدی هر سویچ، هزینه‌ی ن صب هر کنترل‌کننده و هزینه‌ی توان به ازای هر بیت، مقدار تابع هدف افزایش می یابد. اما در ادامه با تغییر در مقدار دو پارامتر دیگر، نتیجه‌ای در مورد تغییری که در تابع هدف ایجاد خواهد شد، می‌گیریم. برای این دو پارامتر تاثیر گذار در معادله افزایش و کاهش آن و تأثیرش در مقدار تابع هدف را تشریح می‌کنیم. در آخر نیز با آزمایشی مزیت روش مقاوم را بیان خواهیم کرد.

در آزمایش‌ها مقادیر هر پارامتر را برای مقایسه تغییر می‌دهیم که ا شاره خواهد شد ولی از مقادیر پیش‌فرض برای پارامترها استفاده می‌کنیم که در جدول (۲) آمده است:

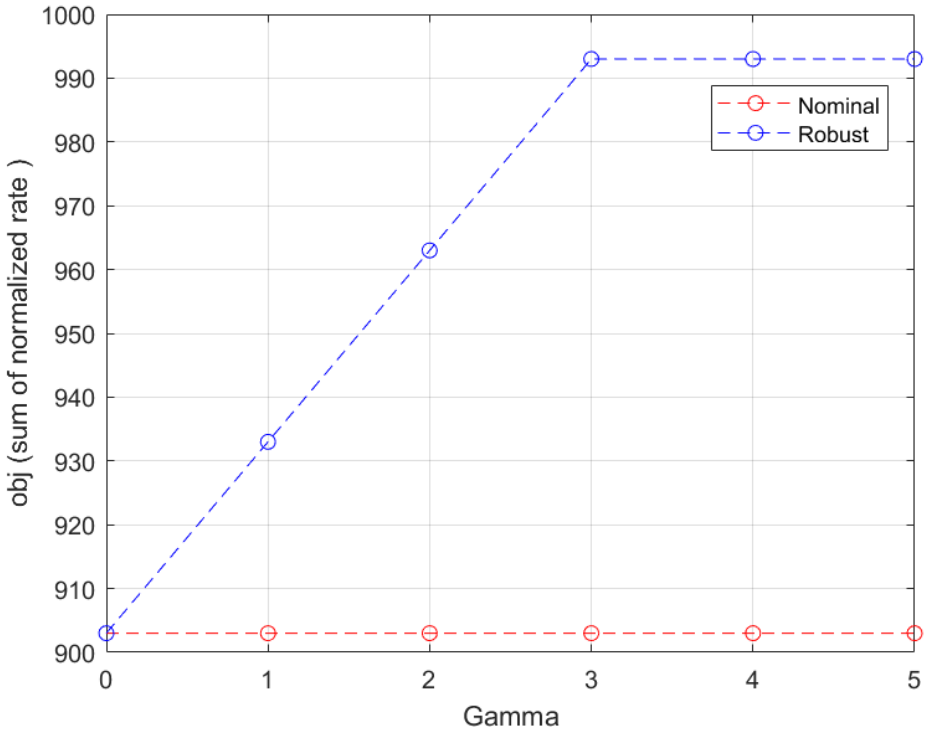
جدول ۲: مقادیر پیش‌فرض

پارامتر	مقادیر داده شده
تعداد کنترل‌کننده (c)	سه عدد
تعداد موقعیت (p)	سه عدد
تعداد سوییچ (s)	۹ عدد
هزینه‌ی نصب هر کنترل‌کننده (ω)	۳ واحد
هزینه‌ی توان به ازای هر بیت (ρ)	۲ واحد

مقدار گاما (Γ)	۱و۲و۳و۴
نرخ تولیدی هر سویچ (r_s)	۱۰۰ واحد
مقدار انحراف ($\hat{r}_s$)	۱۰ واحد
ظرفیت هر کنترل‌کننده (μ)	به ترتیب $[۲۰۰;۳۰۰;۴۰۰]$ واحد

۴. ۲. تغییر گاما

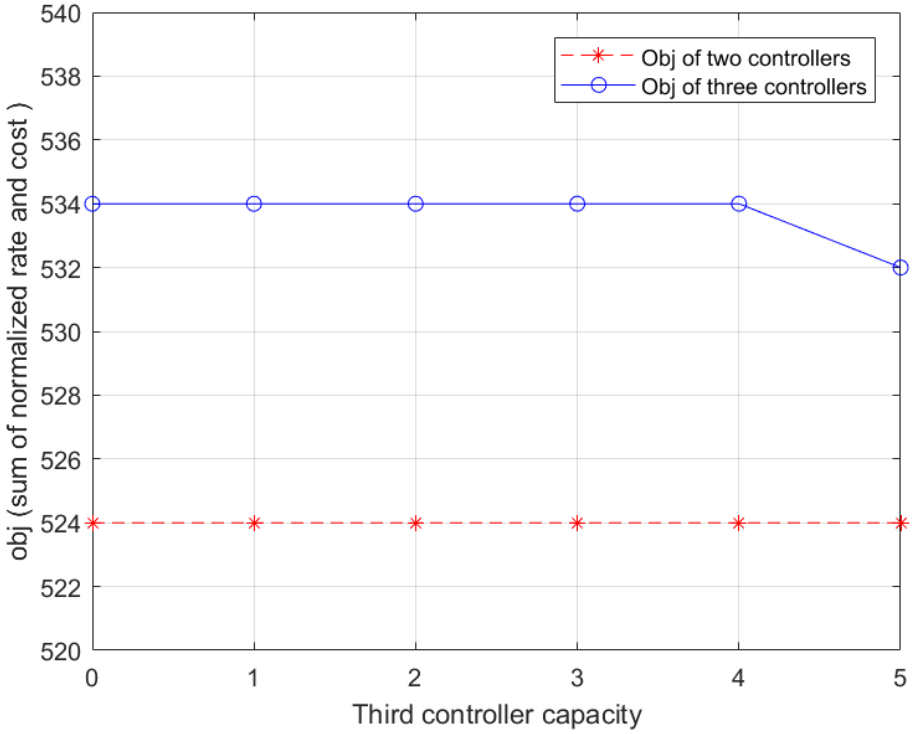
اپراتور شبکه با تنظیم گاما می‌تواند سطح محافظه‌کاری را مشخص کرده و جوابی که بدست می‌آورد را به مقدار جواب بهینه‌ی اسمی نزدیکتر کند. با توجه به معادله‌ی مسئله مقاوم وقتی پارامتر گاما (Γ) افزایش پیدا می‌کند، یعنی عدم‌قطعیت را افزایش داده‌ایم. با افزایش عدم‌قطعیت با توجه به اینکه ما برای هر مقدار غیر قطعی بدترین حالت یعنی کمترین مقدار محبوبیت را درنظر می‌گیریم، در نتیجه در کل مقدار تابع هدف افزایش می‌یابد. در شبیه سازی انجام شده با سه کنترل‌کننده و سه موقعیت و نه سوییچ تابع هدف مسئله‌ی قطعی با مسئله‌ی مقاوم مورد مقایسه قرار می‌گیرد. مقدار تابع هدف برای مسئله‌ی قطعی یک عدد ثابت است و در مسئله‌ی مقاوم با گامای صفر همان مقدار تابع هدف مسئله‌ی قطعی می‌شود ولی همانطور که گفته شد با افزایش گاما مقدار تابع هدف افزایش می‌یابد. این افزایش تا زمانی ادامه می‌یابد که مقدار گاما برابر کل پارمتر مورد بحث (در اینجا نرخ تولیدی هر سوییچ) شود. در نتیجه در این حالت تبدیل به مدل Soyster می‌شود. با توجه به اینکه تعداد موقعیت‌ها سه عدد بود و اگر گاما را برابر سه قرار داده یعنی برای هر موقعیت نرخ سه سوییچ را غیرقطعی در نظر بگیریم و با توجه به اینکه نه عدد سوییچ وجود دارد، بنابراین تمامی نرخ سوییچ ها را غیر قطعی در نظر گرفته است. شکل (۲) توضیحات گفته شده را نشان می‌دهد.



شکل ۲: تاثیر افزایش گاما در مقدار تابع هدف

۴. ۳. اضافه کردن کنترل‌کننده یا تغییر ظرفیت آن

در این آزمایش فرض می‌کنیم که دو کنترل‌کننده تعداد پنج سوییچ را بر عهده دارند. بررسی می‌کنیم که اضافه کردن یک کنترل‌کننده و همچنین افزایش دادن ظرفیت یک کنترل‌کننده چه تغییری در مقدار تابع هدف ایجاد می‌کند. در شکل (۳) مقدار تابع هدف برای پنج سوییچ و دو کنترل‌کننده و مقدار گامای یک برابر ۵۲۴ می‌باشد. حال اگر یک کنترل کننده اضافه کنیم و در یک موقعیت سومی قرار دهیم، به علت اینکه مقدار گاما را یک فرض کرده‌ایم، در نتیجه یک مقدار انحراف (۱۰ واحد) به هر موقعیت اضافه می‌شود. بنابراین با اضافه شدن کنترل‌کننده در موقعیت سوم ده واحد انحراف به مقدار تابع هدف اضافه می‌شود. در این حالت مقدار تابع هدف برابر ۵۳۴ می‌شود. در ادامه مقدار ظرفیت این کنترل‌کننده را افزایش می دهیم. تا زمانیکه ظرفیت کنترل‌کننده تا حدودی برابر ظرفیت دیگر کنترل‌کننده‌ها باشد، مقدار ترافیک بین این کنترل‌کننده‌ها تقسیم می‌شود. ولی زمانیکه مقدار ظرفیت یک کنترل‌کننده خیلی بیشتر از ظرفیت دیگران باشد، مثلا کنترل‌کننده سوم به تنهایی ظرفیت پردازش کل سوییچ‌های موجود را داشته باشد، بنابراین کنترل‌کننده‌های دیگر از دور خارج شده و این کنترل‌کننده به تنهایی کنترل شبکه را برعهده می‌گیرد. علت کاهش نمودار آبی در عدد چهار در شکل (۳) این می‌باشد که در این جا دو کنترل‌کننده دیگر خاموش شده‌اند و فقط یک کنترل‌کننده وجود دارد پس فقط هزینه نصب و نگهداری یک کنترل‌کننده در مقدار تابع هدف حساب می‌شود، درنتیجه مقدار تابع هدف کاهش پیدا کرده است.



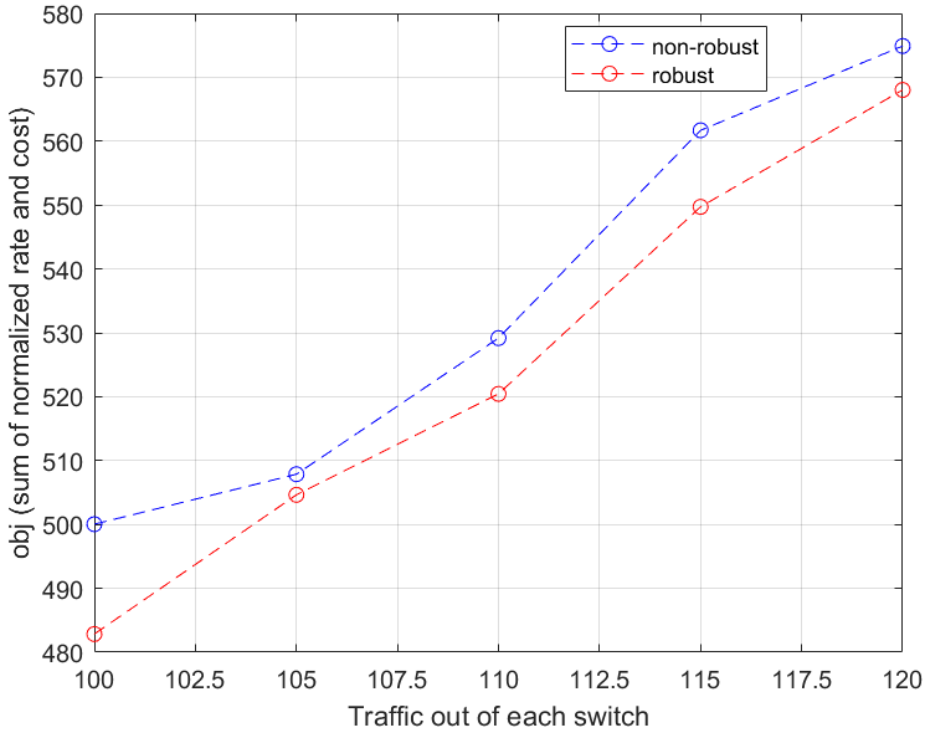
شکل ۳: تاثیر افزایش ظرفیت در مقدار تابع هدف

۴. ۴. مزیت روش مقاوم

هدف از حل برنامه مقاوم این است که ما فرض می‌کنیم مقدار تخمین زده ممکن است دارای خطا باشد. بنابراین، می‌خواهیم اگر این مقدار اشتباه می‌باشد برای آن حالت آماده بوده و از قبل انحراف را در نظر گرفته باشیم و جوابی مقاوم در برابر انحراف به دست آوریم. تابع هدف، معیاری برای ارزیابی جواب‌ها است. حال می‌خواهیم بدانیم با جواب‌هایی که بدست می‌آوریم چقدر توانسته‌ایم مجموع هزینه و نرخ تولیدی را کمینه کنیم. ما مقدار واقعی را نداریم بنابراین نمی‌توانیم جواب دقیق را بدست آوریم. تنها مقداری که داریم مقدار تخمینی از آن است و بنابراین برای مقدار تخمینی مسئله را حل می‌کنیم. در حین اجرای شبکه، مقدار واقعی محبوبیت مشخص می شود. بنابراین ما می‌توانیم متوجه شویم جواب‌هایی که به کار می‌بریم چقدر مناسب هستند. اگر مقدار تخمین زده شده درست باشد (یعنی نرخ‌تولیدی واقعی هر سوییچ برابر با نرخ تخمین زده شده باشد)، مقدار تابع هدف مسئله مقاوم بیشتر از مقدار تابع هدف مسئله اصلی خواهد بود. در مواردی که محبوبیت واقعی انحراف داشته باشد و این انحراف را ما قبلاً در نظر گرفته باشیم، مقدار تابع هدف مقاوم کمتر یا مساوی مقدار تابع هدف برنامه اصلی می شود.

در شکل (۴) سه کنترل‌کننده در هر مرحله کنترل پنج سوییچ را برعهده دارند (هر مرحله نسبت به مرحله‌ی قبل نرخ‌های هر سوییچ را بیشتر فرض می‌کنیم و آزمایش تکرار می‌شود). اگر در این مرحله تابع هدف روش اصلی با تابع هدف روش مقاوم مقایسه کنیم، تابع هدف روش مقاوم کمتر می‌شود. ولی در زمان اجرای آزمایش مقادیر واقعی محبوبیت مشخص می‌شوند و حال اگر دوباره آزمایش را انجام دهیم و این بار مقدار واقعی محبوبیت را در آزمایش قرار دهیم، تابع هدف روش مقاوم مساوی یا حتی کمتر از مقدار

تابع هدف روش اصلی خواهد شد. بنابراین با توجه به توضیحات گفته شده اگر پیش‌بینی از انحراف داشته باشیم تابع هدف روش مقاوم مساوی یا حتی در مواردی کمتر از روش غیر مقاوم می‌شود.



شکل ٤: مزیت روش مقاوم در مقابل روش غیر مقاوم

۵. نتیجه‌گیری

در این مقاله مسئله قرارگیری کنترل‌کننده را با نرخ ترافیک پویا در نظر گرفته و با کمک تکنیک‌های بهینه‌سازی مقاوم این مسئله را فرموله کردیم. در این مقاله انواع بهینه سازی مقاوم مورد بررسی قرار گرفت و روش Bertsimas را با توجه به مزایایش نسبت به دیگر موارد انتخاب کردیم. در قسمت شبیه‌سازی نیز تحلیل حساسیتی نسبت به پارامترهای این مسئله انجام شد. با توجه به اینکه در این مقاله متغیرها بصورت دودویی هستند و در مقیاس‌های بالا این مسئله دچار می‌شود، برای کارهای آتی می‌توان از متغیرهای حقیقی بجای متغیرهای دودویی استفاده کرد که همانند مسئله کوله‌پشتی صفر و یک چند بعدی است و برای حل آن نیز از راه‌حل‌هایی که برای این مسئله می‌باشد، می‌توان استفاده کرد.

۶. مراجع

۱. *A. Shirmarz and A. Ghaffari, "Taxonomy of controller placement problem (CPP) optimization in Software Defined Network (SDN): a survey," J. Ambient Intell. Humaniz. Comput., vol. 12, no. 12, pp. 10473–10498, 2021.*

۲. *R. Masoudi and A. Ghaffari, "Software defined networks: A survey," J. Netw. Comput. Appl., vol. 67, pp. 1–25, 2016.*

۳. *M. Karakus and A. Durresi, "A survey: Control plane scalability issues and approaches in software-defined networking (SDN)," Comput. Networks, vol. 112, pp. 279–293, 2017.*

٤. *B. Heller, R. Sherwood, and N. McKeown, "The controller placement problem," ACM SIGCOMM Comput. Commun. Rev., vol. 42, no. 4, pp. 473–478, 2012.*

٥. *J. Hollinghurst, A. Ganesh, and T. Baugé, "Controller placement methods analysis," in 2016 6th International Conference on Information Communication and Management (ICICM), pp. 239–244,2016.*

۶. *Z. Zhang, J. Lu, and H. Chen, "Controller robust placement with dynamic traffic in software-defined networking," Comput. Commun., vol. 194, pp. 458–467, 2022.*

۷. *D. Bertsimas and M. Sim, "The Price of Robustness," Oper. Res., vol. 52, pp. 35–53, 2004.*