دراسة تقنيات التوجيه في الشبكات المعرفة برمجياً باستخدام التعلم الآلي

المهندس أحمد محيو الشيخ الدكتور المهندس علي الحاتم

الملخص

يعمل توجيه التدفق على تحقيق تحسينات دقيقة في أداء الشبكة من خلال توجيه تدفقات حزم البيانات المختلفة عبر مسارات شبكية متنوعة. حيث يوفر التحكم المركزي في الشبكات المعرفة البرمجية (Software-Defined Network) SDN إطاراً لتحسينات مركزية في الشبكة، مثل توجيه التدفق الأمثل، إن توجيه تدفق يتكيف مع تغيرات الأحمال يتطلب نماذج متطورة. نهدف من خلال هذه الدراسة إلى اتباع نهج يعتمد على التعلم المعزز، حيث ينتج عن التوجيه المباشر للتدفق فضاء من الثنائية (حالة–إجراء) يزداد مع زيادة عدد التدفقات الجارية، مما يؤدي إلى ظهور مشكلة محتملة في قابلية التوسع. نشير إلى أن هذه المشكلة تنشأ بشكل أساسي من المساحة المطلوبة لتخزين جدول Q الذي يتضمن الثنائيات الخاصة بالتدفق، بالإضافة إلى الوقت اللازم للتعلم حتى الوصول إلى التقارب والذي يتطلب استكشاف هذا الفضاء الكبير من الحالات والأفعال. من جانب أخر، فإنّ هناك عدد قليل من الحسابات المطلوبة لإيجاد الحالات (ويتطلب فقط تحديث قيم Q في كل خطوة). تم حل مشكلة قابلية التوسع من خلال استخدام عدة متحكمات SDN Controllers بحيث يكون لكل متحكم عدد أقل من العقد والتدفقات لإدارتها، وبالتالي جدول Q أصغر وفضاء حالة–إجراء أصغر. يقوم كل متحكم بتحسين توجيه التدفق ضمن شبكة صغيرة ومعقولة باستخدام التعلم المعزز، حيث يعمل كل نظام مستقل (Autonomous System) AS بتحسين التوجيه داخل المجال الخاص به وقد أظهرت النتائج التجريبية أن هذا النهج يُحقّق تحسينات جوهرية في الأداء: حيث بلغ متوسط زمن التقارب للمتحكمات المتعددة 10 خطوات فقط، مقارنةً بـ 40.5 خطوة في حالة المتحكم

المركزي الوحيد. كما انخفض متوسط زمن الاستجابة من 31.30 مللي ثانية (في النظام الأحادي) إلى 23.14 و23.99 مللي ثانية في أنظمة المتحكمات المتعددة. هذه النتائج تؤكد أن توزيع مسؤوليات التحكم يقلل بشكل فعّال من تعقيد فضاء التعلم، ويسرع من عملية التقارب، ويحسّن كفاءة توجيه التدفق، مما يجعل الحل المقترح قابلاً للتطبيق في الشبكات الكبيرة والديناميكية.

الكلمات المفتاحية: الشبكات المعَرفة برمجياً، توجيه التدفق، التعلم المعزز، التوسع.

Studying Routing Techniques in SDN Software-Defined Networking Using Machine Learning

ENG.Ahmad Alshiekh Dr.Ali **Alhatem**

Abstract

Flow routing enables fine-grained improvements in network performance by directing different data packet flows through diverse network paths. Centralized control in Software-Defined Networking (SDN) provides a framework for centralized network optimizations, such as optimal flow routing. However, adaptive flow routing that responds to dynamic traffic load variations requires sophisticated models. This study adopts a reinforcement learning-based approach, where direct flow routing generates a state-action pair space that grows with the number of active flows, leading to a potential scalability issue. This problem primarily arises from the large memory space required to store the Q-table containing flow-specific state-action pairs, as well as the prolonged learning time

needed to reach convergence, which involves exploring this vast state-action space. On the other hand, the computational overhead per step remains low, as it only requires updating Q-values at each step. To address this scalability challenge, we propose a distributed approach employing multiple SDN controllers, where each controller manages a smaller subset of network nodes and flows. Consequently, each controller operates with a smaller Q-table and a reduced state-action space. Each controller independently optimizes flow routing within its manageable subnetwork using reinforcement learning, with every Autonomous System (AS) improving routing within its own domain. Experimental results demonstrate that this approach achieves significant performance improvements: the average convergence time for multi-controller setups is only 10 steps, compared to 40.5 steps in the single centralized controller case. Furthermore, the average latency decreased from 31.30 ms (in the single-controller system) to 23.14 ms and 23.99 ms in the multi-controller systems. These results confirm that distributing control responsibilities effectively reduces the complexity of the learning space, accelerates convergence, and enhances flow routing efficiency, making the proposed solution suitable for large-scale and dynamic networks.

Keywords: Software-Defined Networking (SDN), Flow Routing, Reinforcement Learning, Scalability.

1.مقدمة:

يُعد توجيه التدفقات (Flow Routing) مشكلة أساسية في شبكات تبديل الحزم، حيث يجب أن تتبع الحزم الخاصة بتدفق معين – كتدفق من مضيف مصدر محدد إلى مضيف وجهة محدد أو تدفق بروتوكول التحكم في النقل – نفس مسار التوجيه (Routing Path) عبر شبكة عقد تبديل الحزم (Packet–Switching Nodes) ومع ذلك، يمكن أن تتبع مسارات مختلفة حتى بين نفس زوج المبدلات (المصدر–الوجهة) حيث تكون مبدلة المصدر هي مبدلة الدخل Ingress) (Switchلنطاق التوجيه (Routing Domain) وتكون مبدلة الوجهة هي مبدلة الخرج (Egress Switch)لنطاق التوجيه للتدفقات المعتبرة بهدف تحسين أداء الشبكة [1]. وبالتالي يختلف توجيه التدفقات (Flow Routing) بشكل جوهري عن توجيه المسارات الكلاسيكي (Classical Path Routing)القائم فقط على مضيف الوجهة كما في بروتوكول الإنترنت (Internet Protocol – IP) أو على زوج المبدلات المصدر–الوجهة. يمكّن توجيه التدفقات مجموعة واسعة من التكيفات لتحسين الأحمال على وصلات الشبكة (Network Links) أي لإجراء هندسة مرور فعالة بهدف تحسين مقاييس أداء الشبكة المختلفة [1] . ومع أن توجيه التدفقات يوفّر مرونة كبيرة في تحسين أداء الشبكة، إلا أن تحقيق هذه المرونة في بيئات ديناميكية وكبيرة الحجم يواجه تحديات جوهرية، خاصةً عند الاعتماد على تقنيات التعلّم الذكي مثل التعلّم المعزز (Reinforcement Learning). ففي الشبكات المعرفة برمجياً (SDN)، حيث يُدار التوجيه من خلال متحكم مركزي، يؤدي تزايد عدد التدفقات النشطة إلى نمو هائل في فضاء الحالات (State Space) وفضاء الأفعال (Action Space)، ما يُعقّد عملية التعلّم ويُبطئ التقارب، بل وقد يجعل النظام غير قابل للتوسع. ويزداد الوضع سوءًا عندما يُطلب من المتحكم اتخاذ قرارات توجيه في الوقت الفعلي لتدفقات متنوعة من حيث الحجم، الأولوية، ومتطلبات زمن الوصول. وبالتالي، يبرز التحدي البحثي المركزي: كيف يمكن تحقيق توجيه تدفقات ذكي وقابل للتوسع، يوازن بين كفاءة التعلّم، سرعة الاستجابة، وتحسين أداء الشبكة الشامل، دون التضحية بقابلية التوسع أو جودة الخدمة؟

2.هدف البحث

يهدف هذا البحث إلى معالجة مشكلة قابلية التوسع من خلال استخدام عدة متحكمات SDN بحيث يكون لكل متحكم مجال خاص به و بالتالي يعمل كل متحكم على اتخاذ قرار التوجيه لعدد أقل من العقد والتدفقات، الأمر الذي ينتج عنه فضاء حالة–إجراء أصغر لاستكشافه مما يسهم بالحصول على جدول Q أصغر. نهدف أيضاً إلى تحسين توجيه التدفق عن طريق استخدام التعلم المعزز الذي يعمل على اختيار المسار الأفضل كما سنرى لاحقاً.

3.مواد وطرق البحث

تم استخدام محاكي Mininet، وهو برنامج محاكاة يعتمد على Linux من أجل نمذجة الشبكات المعرفة برمجياً [2]. يعمل Mininet على تشغيل وإدارة مجموعة من الأجهزة المضيفة (Hosts) والمبدلات (Switches) والموجهات (Routers) والوصلات (Links) ووحدات التحكم (Controllers) باستخدام المحاكاة الافتراضية (Virtualization) لجعل نظام واحد يبدو وكأنه شبكة كاملة. يتضمن Mininet واجهة أوامر خاصة بالشبكة (Network–Aware CLI)، كما يوفر Mininet واجهة لبرمجة التطبيقات (Python API) لإنشاء الشبكة واختبارها [2]. يحتوي Mininet على واجهة مستخدم بسيطة يوفرها الملف النصي (Script) [4].

كما استخدمنا أداة iPerf وهي أداة مفتوحة المصدر ومجانية تُستخدم لقياس وتحليل أداء الشبكات، خاصة في قياس عرض النطاق الترددي (Bandwidth – BW) بين نقطتين على الشبكة، بالإضافة إلى قياس فقدان الحزم (Packet Loss) والتأخير (Delay) والتنوع في التأخير (Jitter). تعتمد iPerf على نموذج العميل والخادم (Client–Server Model)، حيث يُشغل وضع الخادم (Server Mode) على جهة، ويتصل به وضع العميل (Client Mode) ليبدأ اختبار نقل البيانات باستخدام بروتوكولات متعددة مثل بروتوكول التحكم في النقل (Transmission Control Protocol – TCP) وبروتوكول بيانات المستخدم (User

Datagram Protocol – UDP) وبروتوكول التحكم في النقل المتقدم (Stream Control Transmission Protocol – SCTP). تتيح الأداة ضبط عدة وسطاء مثل حجم الحزمة (Packet Size)، مدة الاختبار (Test Duration)، والنطاق الترددي المستخدم (Used Bandwidth)، ما يمكن الباحثين من إجراء اختبارات دقيقة تحت ظروف مختلفة [5].

4. الدراسات السابقة

قام الباحثون في الدراسة [6] بتصميم خوارزمية توجيه في شبكات SDN تعتمد على التعلم المعزز باستخدام Q–learning مع مراعاة معايير جودة الخدمة (Quality of Service) QoS للوصلات الشبكية . تهدف الخوارزمية إلى الاستفادة الكاملة من وصلات الشبكة لتحديد المسار الأفضل. أظهرت النتائج التجريبية أنّ الخوارزمية قادرة على إيجاد المسار الذي يوفر جودة خدمة أفضل بدقة تقارب 100% بعد فترة تدريب معينة. من محدوديات خوارزمية Q–learningأن تصميم الميزات (features) المطلوبة يحتاج لتدخل يدوي، وهو أمر معقد وصعب التطبيق في بيئات الشبكات الحقيقية التي تتميز بتعقيد كبير في الخصائص. لمعالجة هذه المشكلة، اقترح الباحثون دمج الشبكات العصبونية مع التعلم المعزز، بحيث يتم استبدال جدول Q التقليدي بدالة تقريبية يتم تدريبها، مما أدى إلى خوارزمية توجيه قائمة على التعلم المعزز العميق (Deep Q–learning) أظهرت نتائج التجارب أن خوارزمية التوجيه المعتمدة على Deep Q–learning تقدم أداءاً جيداً مع تحسينات ملموسة على جودة الخدمة واستغلال موارد الشبكة. هذا يشمل قدرة أفضل على التكيف مع بيئة الشبكة الديناميكية والمعقدة مقارنة بالخوارزميات التقليدية مثل Dijkstra و RIP.

بينت الدراسة [7] أن التعامل مع أحمال متنوعة ومتفاوتة يستلزم استخدام نموذج معقد، وبالتالي تم التركيز على تحقيق نظام تعلم معزز دون الاعتماد الشبكات العصبونية (model–free RL scheme). تم تصميم نموذج QR–SDN، يقوم بإنشاء مسارات متعددة بين المصدر والوجهة،

مما يحقق تأخيرات تدفق أقل بكثير. أشارت الدراسة إلى الجهود البحثية الإضافية لابتكار نهج قابل للتوسع مع زيادة حجم الشبكة.

قدمت الدراسة [8] نهجاً للتوجيه بعنوان التعلم المعزز والشبكات المعرفة برمجياً للتوجيه الذكي (Reinforcement Learning and Software-Defined Networking for Intelligent Routing - RSIR)، والذي يعتمد على الحاجة إلى إضافة مستوى المعرفة (Knowledge Plane إلى الشبكة، والتي يتم تغذيتها بالبيانات التي تجمعها بواسطة مستوى الإدارة (Management Plane). وبشكل خاص، طوروا خوارزمية توجيه استباقية (proactive) مبنية على التعلم المعزز تعتمد على مقاييس حالة الرابط (link-state metrics) وتم تنفيذها في نموذج أولي مع مصفوفات حركة حقيقية. تمت مقارنة RSIR مع خوارزمية ديكسترا الكلاسيكية (classic Dijkstra's algorithm)، والتي تستند إليها معظم بروتوكولات التوجيه. أظهرت النتائج أن RSIR يحصل على المزيد من المسارات الأقصر ويستطيع توازن الحمل بشكل أفضل، وبالتالي تقليل التأخيرات الإجمالية. من توصيات هذه الدراسة الحاجة إلى تطوير النهج إلى التعلم المعزز العميق.

قدمت الدراسة [9] تقنية Deep Q-Routing (DQR)، والتي تستخدم شبكة Q العميقة (dueling deep Q-network) مع إعادة تشغيل التجربة ذات الأولوية (prioritised experience replay) لحساب مسار أي طلب زوج مصدر-وجهة مع وجود مقاييس QoS متعددة، مثل التأخير (delay)، عرض النطاق الترددي (bandwidth) أو الفقدان (loss). تمت المقارنة مع طرق التعلم الحالية الأخرى للتوجيه المباشر على الإنترنت (greedy online routing)، وأظهرت نتائج أفضل من حيث الفقدان (loss) وتكلفة المسار (path cost)، مع الحفاظ على أفضل عرض نطاق ترددي معظم الأوقات وتأخير معقول.

اقترحت الدراسة [10] نهجاً مبنياً على الشبكة العصبونية العميقة DNN في نشر شبكات SDN/OSPF الهجينة. تقوم وحدة تحكم SDN بأداء توجيه موفر للطاقة وأداء محسّن مع

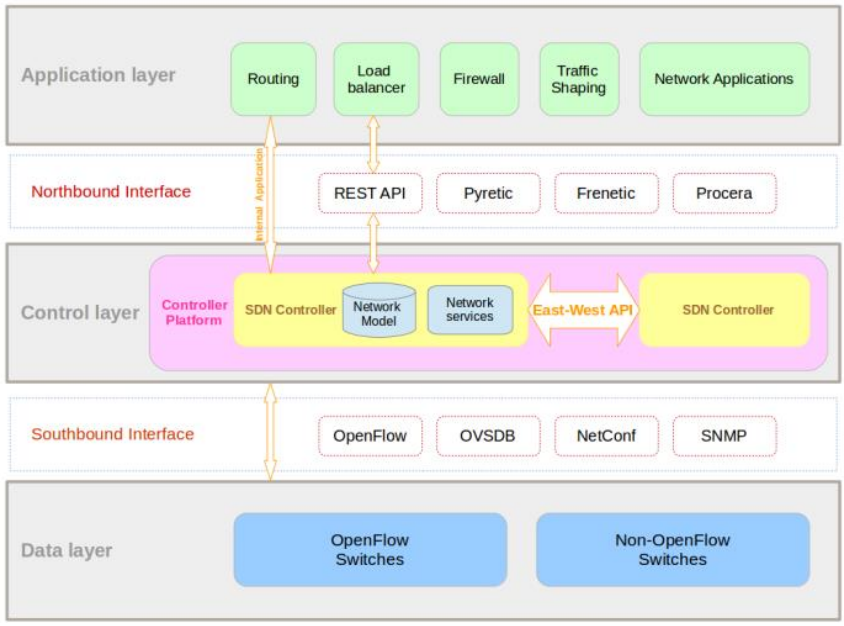
ضمانات جودة الخدمة QoS. تضمّن النموذج المعتمد عليه وحدة تعلم آلي ووحدة التعلم المعزز العميق DRL. يتكوّن التعلم الآلي من شبكة LSTM تقوم بالتنبؤ بتدفق الحركة باستخدام مجموعات بيانات السلاسل الزمنية، والتي تستخرج تقلبات وفترات تكرار بيانات الشبكة قصيرة المدى لضمان تنبؤ تدفق الحركة والتوجيه الموفر للطاقة مع أداء QoS مضمون. تقوم وحدة DRL بالتعلم من البيانات التاريخية الموجودة والتعلم التكراري من التفاعل مع إعدادات الشبكة المحددة.

بناءاً على ما سبق ذكره من دراسات مرجعية وبالإشارة إلى توصيات هذه الأبحاث، تم تحديد مشكلة البحث المتمثلة في تحسين التدفق في شبكات SDN بشكل يراعي قابلية التوسع والكفاءة في اختيار المسارات عن طريق تقسيم الشبكة إلى عدة مجالات واستخدام التعلم المعزز.

5.بنية شبكة المعرفة برمجياً Software-Defined Networking

تقترح مبادرة الشبكات المُعرَّفة برمجيا (SDN)، التي تقودها مؤسسة الشبكات المفتوحة (ONF) (the Open Networking Foundation)، بنية مفتوحة جديدة لمواجهة تحديات الشبكات الحالية، مع إمكانية تسهيل أتمتة عمليات تهيئة الشبكة، والأفضل من ذلك، برمجة الشبكة بالكامل. وعلى عكس البنية التقليدية الموزعة للشبكات، حيث تكون أجهزة الشبكة مغلقة ومتكاملة رأسياً (vertically-integrated)، وتدمج البرمجيات مع الأجهزة، فإن بنية SDN (كما في الشكل (1)) ترفع مستوى التجريد (abstraction) عن طريق فصل مستوى البيانات عن مستوى التحكم في الشبكة. وبهذه الطريقة، تصبح أجهزة الشبكة مجرد مبدلات بسيطة لإعادة التوجيه، بينما تتركز كل منطقية التحكم (control logic) في وحدات تحكم برمجية، مما يوفر إطاراً برمجياً مرناً لتطوير التطبيقات المتخصصة ونشر الخدمات الجديدة [12].

يُعتقد أن هذه الجوانب في بنية SDN تبسط وتحسن إدارة الشبكة من خلال إتاحة إمكانية الابتكار، وتخصيص السلوكيات، والتحكم في الشبكة وفقًا لسياسات عالية المستوى يتم التعبير عنها كبرامج مركزية، مما يسمح بتجاوز تعقيدات تفاصيل الشبكة على المستوى المنخفض والتغلب على المشاكل البنيوية الأساسية .يُضاف إلى هذه الميزات قدرة SDN على التعامل بسهولة مع عدم تجانس البنية التحتية وذلك بفضل التجريد الذي توفره الواجهة البينية الجنوبية (Southbound interface) في SDN [12].



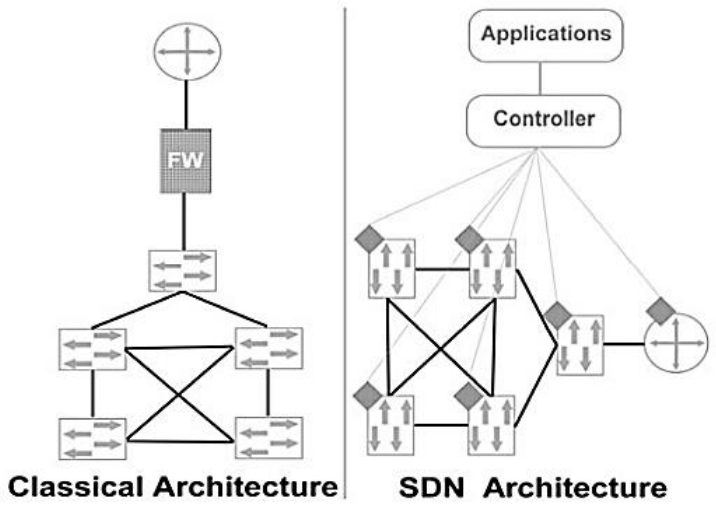
الشكل (1): بنية الشبكات المُعرَّفة برمجيا الموزعة ذات الثلاث طبقات [12].

6.مقارنة بين الشبكات المعرفة برمجياً والشبكات التقليدية

في البنية التحتية التقليدية (كما في الشكل (2))، يتطلب تنفيذ الشبكة وتهيئتها واستكشاف أخطائها وإصلاحها تدخل مهندسي شبكات وأنظمة ذوي مهارات تقنية

عاليـــة، بالإضـــافة إلـــى التكـــاليف التشـــغيلية المرتفعـــة المرتبطـــة بتـــوفير وإدارة الشـــبكات الكبيـرة متعـددة المـوردين. وفـي الواقـع، إن تنـوع عناصـر الشـبكة وتعقيـدها [12] يجعـلان صـيانتها باهظـة التكلفـة، والبنيـة التحتيـة الأساسـية أقـل موثوقيـة فـي حالـة حـدوث أعطـال متكررة للشبكة، خاصة إذا لم يتم تجهيز خطط احتياطية ضمن البنية التحتية.

نظـراً لأن SDN تفصـل قـرارات التوجيـه وإعـادة التوجيـه الخاصـة بعناصـر الشـبكة (مثـل الموجّهـات والمبـدلات ونقـاط الوصـول) عـن مسـتوى البيانـات (data plane)، فـإن إدارة الشـبكة تصـبح أقـل تعقيـداً (كمـا موضـح فـي الجـدول (1)) الـذي يوضـح تفـوق شـبكات SDN علـى الشـبكات التقليديـة؛ وذلـك لأن مسـتوى الـتحكم (control plane) يتعامـل فقـط مـع المعلومـات المتعلقـة بهيكليـة الشـبكة المنطقيـة (logical network topology)، وتوجيـه حركـة المـرور، ومـا إلـى ذلـك. فـي المقابـل، يقـوم مسـتوى البيانـات بتنسـيق حركـة مـرور الشـبكة وفقًـا للتهيئـة التـي تـم إعـدادها فـي مسـتوى الـتحكم فـي بنيـة SDN، تكـون عمليـات الـتحكم مركزيـة فـي وحـدة تحكـم (Controller) هـي التـي تفـرض سياسات الشبكة [13].



الشكل (2): بنية الشبكات المعرفة برمجيا مقابل بنية الشبكات التقليدية [13].

جدول (1): مقارنة بين الشبكات المُعرَّفة بالبرمجيات (SDN) والشبكات التقليدية [13].

الخصائص	البنية التقليدية	بنية SDN
القابلية للبرمجة		✓
التحكم المركزي		✓
التهيئة والعُرضة للخطأ	✓	
التحكم المعقد بالشبكة	✓	
مرونة الشبكة		✓

أداء مُحسَّن		✓
سهولة التنفيذ		✓
تهيئة فعّالة		✓
إدارة مُحسَّنة		✓

7.مشاكل توجيه التدفق في SDN

تتمثل التحديات والقضايا ذات الأهمية القصوى في توجيه حركة المرور ضمن بيئة SDN فيما يلي [14] :

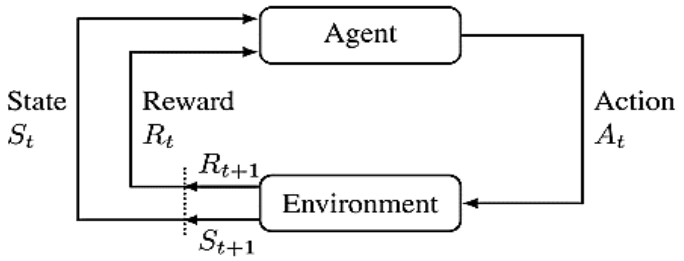
- **قابلية التوسع** (Scalability) :إن قدرة شبكات الـ SDN على التعامل مع أعداد هائلة من الأجهزة وتدفقات حركة المرور له أمر بالغ الأهمية. وهذا يطرح تحدياً، حيث يجب على وحدات تحكم SDN إدارة كميات كبيرة من البيانات وتحديد المسارات بسرعة.
- **التكيف في الوقت الفعلي مع تغيرات الطوبولوجيا** Real-time adaptation to topology) (shifts : يجب على شبكات SDN الاستجابة الفورية للتغيرات الديناميكية في بنية الشبكة. وهذا أمر حتمي، حيث يجب على وحدات تحكم SDN إعادة توجيه حركة المرور بسرعة لتجاوز الاتصالات الفاشلة أو العُقد المزدحمة.
- **تلبية متطلبات جودة الخدمة (QoS) المتنوعة** : يعد الالتزام بمتطلبات جودة الخدمة المحددة لكل تطبيق أمراً ضرورياً لشبكات .SDN. يجب على وحدات تحكم SDN تحديد أولويات حركة المرور وتوجيهها بفعالية، بما يتماشى مع الاحتياجات الخاصة بكل تطبيق.
- **اعتبارات الأمن والخصوصية المرتفعة** :إن ضمان أمن وسرية بيانات المستخدم داخل شبكات SDN يعتبر أمر بالغ الأهمية. إن التحكم المركزي والبيانات المركزية في شبكات SDN يجعلانها عرضة للانتهاكات المحتملة.

- **الاستخدام الأمثل للموارد** :يعد الاستغلال الفعال لموارد الشبكة مطلباً أساسياً لشبكات SDN.يجب على وحدات تحكم SDN صياغة قرارات توجيه تقلل من الازدحام وتعظم من استخدام عرض النطاق الترددي.

على الرغم من هذه التحديات، تُعد SDN تقنية واعدة لديها القدرة على إحداث ثورة في طريقة إدارة الشبكات. ومع نضوج هذه التقنية، من المرجح أن يتم التعامل مع هذه التحديات، وستصبح SDN حلاً أكثر جدوى لمجموعة واسعة من الشبكات.

8. تحسين التوجيه القائم على التعلم المعزز:

تُطبَّق خوارزميات التعلم المعزز (RL) بشكل عام لحل مسائل اتخاذ القرار. وعند تطبيقها لتحسين التوجيه، تؤدي وحدة التحكم دور **الوكيل (agent)**، بينما تمثل الشبكة **البيئة (environment)** ويتألف **فضاء الحالة (state space)** من حالات الشبكة وحركة المرور. أما **الإجراء (action)** فهو حل التوجيه المختار. وتُعرَّف **المكافأة (reward)** بناءً على مقاييس التحسين، مثل تأخير الشبكة كما هو موضح في الشكل (3) [15].



الشكل (3): آلية عمل التعلم المعزز [15].

عند كل خطوة زمنية t، يراقب الوكيل حالة S_t، ويختار إجراءً A_t من فضاء الإجراءات A، ويستقبل مكافأة فورية R_t (تشير إلى مدى جودة أو سوء هذا الإجراء)، ثم ينتقل إلى الحالة التالية S_t +1. يتمثل هدف الوكيل في تعلم **سياسة السلوك المثلى** π **(optimal behavior policy)** ، وهي عبارة عن ربط مباشر من فضاء الحالة S إلى فضاء الإجراءات A أي (π: S → A)، وذلك بهدف تعظيم المكافأة المتوقعة طويلة الأمد [16] .

انطلاقًا من سياسة السلوك π، يستطيع الوكيل تحديد أفضل إجراء مقابل لحالة معينة. في التعلم المعزز، تُستخدم **دالة القيمة (value function)** لحساب المكافأة طويلة الأمد لإجراء معين في حالة معينة. إن دالة القيمة الأشهر هي **دالة-Q (Q-function)**، والتي تستخدمها خوارزمية **تعلم Q (Q-learning)** لتعلم جدول يخزن جميع أزواج الحالة-الإجراء (state-action pairs) ومكافآتها طويلة الأمد[16].

9. خوارزمية Q-LEARNING

تُعد (Q-learning) خوارزمية تعلم المعزز غير معتمدة على النموذج (model-free)، تهدف إلى تعليم الوكيل سياسة الإجراءات التي يجب اتباعها وفقاً للحالة والملاحظات المستقاة من البيئة. وبصفته أسلوباً غير معتمد على النموذج، فإنه لا يستخدم احتمالية الانتقال (transition probability) [17] .

يعمل هذا الأسلوب ضمن إطار عمليات ماركوف لاتخاذ القرار *Markov Decision Process* (MDP)، حيث يسعى لإيجاد سياسة مثلى (optimal policy) من خلال تعظيم القيمة المتوقعة للمكافأة التراكمية (cumulative reward) المحسوبة على مدار جميع الخطوات المتتالية، بدءاً من الحالة الحالية. وفي الوقت الحاضر، يُشكِّل هذا الأسلوب خط أساس (baseline) تُقارَن به أساليب التعلم المعزز الأخرى كما هو موضح في المعادلة (1) [17].

$$Q(S_t, A_t) \leftarrow (1-\alpha)Q(S_t, A_t) + \alpha\left(R_{t+1} + \gamma \max_{a\in A} Q(S_{t+1}, a)\right) \quad [17] \quad (1)$$

حيث يحدد **معدل التعلم** (**learning rate**) α مدى سرعة استيعاب قيم Q المكتسبة حديثًا. ويمثل **معامل الخصم** (**discount rate**) γ مدى أهمية المكافآت المستقبلية المتوقعة. يتم تمثيل المكافآت المستقبلية المتوقعة بالمعادلة $\max_{a\in A} Q(S_{t+1}, a)$، والتي تمثل بشكل أساسي أقصى مكافأة يمكن الحصول عليها عند اتخاذ الإجراء a ذي القيمة الأعلى.

1.9 تصميم فضاء الحالة (STATE-SPACE DESIGN)

بفرض لدينا الشبكة G(V, E) حيث E هي مجموعة من الحواف التي تربط مجموعة الرؤوس V. نركز على تدفقات الاتصال الأحادية البث (unicast communication flows)، أي التدفقات التي تنقل البيانات من مرسل معين إلى مستلم واحد. يشار إلى نقل البيانات إلى طبقة التطبيق أو النقل من مرسل معين (مضيف المصدر) s_f إلى مستلم معين (مضيف الوجهة) d_f بالتدفق f. نرمز إلى مجموعة التدفقات بالرمز F. نفترض أن التدفق f ينقل معدل حركة معين R_f من مضيف المصدر sf إلى الشبكة. المسار (Ps_f,d_f) هو تسلسل من الرؤوس P = (v1, … , vn) من مجموعة جميع المسارات الممكنة $Ps_f,d_f = \{ Ps_f,d_{f,1}, Ps_f,d_{f,2}, \ldots\}$ P $\in$ التي تربط مضيف المصدر s_f بمضيف الوجهة d_f، حيث يمكن تحديد مجموعة (Ps_f,d_f) بواسطة خوارزمية بحث في الرسم البياني، مثل البحث بالعمق (Depth-First Search – DFS) [18].

يراقب وكيل التعلم المعزز، الذي قد يعمل في وحدة تحكم SDN، البيئة (أي الشبكة) من خلال قياس المؤشرات الرئيسية للأداء المطلوبة، مثل زمن الاستجابة أو عرض النطاق الترددي، في خطوات زمنية منفصلة.... t = 0, 1,2, الملاحظة تتكون من حالة البيئة S_t من مجموعة الحالات

S ={S^1, S^1, …} ومكافأة $R_t \in R \subset R$. نحدد الحالة S_t لتتكون من جدول، يحتوي على المسار المحدد حالياً P لكل تدفق f:

$$S_t = \begin{cases} f_{s_1,d_1} : P_{s_1,d_1,t} \\ \vdots \\ f_{s_i,d_i} : P_{s_i,d_i,t} \end{cases} \quad [7] \quad (2)$$

يتكون فضاء الحالة من قاموس من المبدلات والقيم مع التدفقات كمفاتيح والمسار الحالي كقيمة لكل مفتاح. نلاحظ أن هذا القاموس هو مجرد تطبيق ممكن واحد لفضاء الحالة. يمكن أيضاً تمثيل الحالات كقائمة، يمكن ربطها مباشرة بمدخلات الشبكة العصبية المستخدمة في التعلم Q العميق. لقد فضلنا القاموس مع التدفقات كمفاتيح والمسارات كقيم، حيث أن هذا تمثيل مباشر لفضاء الحالة من وجهة نظر تنفيذ البرمجة [7].

2.9 تصميم فضاء الإجراء (ACTION–SPACE DESIGN)

اعتماداً على الحالة S_t ومكافأتها المقابلة R_t ، يتم اختيار إجراء A $A_t \in$ (حيث قد تعتمد مجموعة الإجراءات الممكنة A بشكل عام على الحالة S_t). مجموعة الإجراءات A = { $A_{t,1}$, $A_{t,2}$, …} تُحدد بواسطة مجموعة المسارات الممكنة، بما في ذلك المسار الحالي، أي A = Ps_f,d_f للتدفق f. يتم اختيار أحد هذه المسارات الممكنة إما لاستبدال المسار الحالي أو الاحتفاظ به. وبالتالي، يغير الإجراء القيمة، أي المسار الحالي، لمبدل، أي تدفق.

ف إجراء A_t المطبق على تدفق واحد بواسطة: [7] (3)

$$A_t = \{f_{s_1,d_1} : P_{s_1,d_1,t} \Rightarrow P_{s_1,d_1,t+1}\}$$

الآن يبقى السؤال عما إذا كان يجب تغيير تدفق واحد أو عدة تدفقات بإجراء واحد. يمكننا تغيير تدفق واحد في خطوة زمنية، أي إجراء تغيير تدفق واحد كما هو محدد في

المعادلة (4). بدلاً من ذلك، يمكننا تغيير جميع التدفقات في خطوة زمنية، أي اتخاذ الإجراء:

$$A_t = \begin{cases} \{f_{s_1,d_1}: P_{s_1,d_1,t} \Rightarrow P_{s_1,d_1,t+1}\}, \\ . \\ . \\ . \\ \{f_{s_i,d_i}: P_{s_i,d_i,t} \Rightarrow P_{s_i,d_i,t+1}\}, \end{cases} \quad [7] \ (4)$$

والتي نشير إليها بتغيير مباشر (Direct Change). من الواضح، كما تشير المعادلتان (3) و(4)، أن تصميم الإجراء A_t له تأثير على حجم فضاء الإجراء $|A|$. عيب نهج التغيير المباشر هو أن فضاء الإجراء يتناسب مع حاصل ضرب أعداد المسارات الممكنة للتدفقات؛ بالمقابل، يتناسب فضاء إجراء تغيير تدفق واحد مع مجموع أعداد المسارات الممكنة للتدفقات. من ناحية أخرى، يسمح التغيير المباشر بالتبديل المباشر في خطوة زمنية واحدة إلى الحالة المرغوبة (تكوين التوجيه) من أجل تحقيق أداء توجيه تدفق أعلى [7].

3.9 تصميم المكافأة (REWARD DESIGN)

تستخدم المكافأة R_t +1 لقياس مدى جودة إجراء A_t في حل مشكلة توجيه التدفقات. مدعومين بالاهتمام المتزايد في شبكات زمن الاستجابة المنخفض [19]، [20]، تأخذ تقييماتنا في هذه الدراسة في الاعتبار زمن الاستجابة للمكافأة. أيضاً، فإن الازدحام يزيد بشكل عام زمن الاستجابة، ولكن ليس معدل البت الناقل المستهلك. يتطلب اعتبار معدل النقل للمكافأة معرفة بمعدل البت الناقل المطلوب لكل تطبيق. بدون معرفة بمتطلبات الأجهزة المرسلة، سيكون من غير الواضح ما إذا كان تغيير معدل النقل يعود إلى قرار توجيه سيئ أو فقط لأن الجهاز المرسل قد خفض معدل البت الناقل. إذا كان يجب اعتبار مقاييس أداء متعددة، فيمكن استخدام صيغة موزونة كما هو مقترح في [21].

تتكون مكافأتنا المقترحة R_t من مجموع أزمن الاستجابة L_f على طول المسارات الحالية Ps_{f,d_f} للتدفقات $f \in F$. من أجل إعطاء وزن ثقيل نسبياً للقيم المتطرفة نستخدم الجذر التربيعي للمتوسط:

$$R_t = -\sqrt{\left(\frac{\sum\_\forall f \epsilon F \; L_f^2}{|F|}\right)} \qquad [7]\ (5)$$

نلاحظ أن الإشارة السالبة مطلوبة لأن وكيل التعلم المعزز يسعى لتعظيم مكافأته؛ ومع ذلك، فإن زمن الاستجابة الأعلى أقل رغوبة.

4.9 استراتيجية الاستكشاف (EXPLORATION STRATEGY)

في التعلّم المعزز، يواجه الوكيل (Agent) تحدي تحقيق توازن بين الاستكشاف (Exploration) — أي تجربة إجراءات جديدة لاكتساب معرفة عن البيئة — والاستغلال (Exploitation) — أي استخدام المعرفة الحالية لتعظيم المكافأة. ولتحقيق هذا التوازن، تُطبَّق استراتيجيات استكشاف معيارية. حيث تم تطبيق استراتيجية Softmax بمعامل درجة الحرارة τ، بحيث أن درجة حرارة τ منخفضة تفضل الاستغلال (أي، يتم اختيار الإجراء ذو أعلى قيمة Q أكثر غالباً)، ودرجة حرارة τ عالية تؤدي إلى طابع استكشافي لاقتناء معرفة جديدة.

بما أن قيم Q لدينا تُهيّأ بـ −∞، فيتم استخدام المعادلة التالية إلى:

$$Pr\{A_t = a\} = \frac{exp\left[-\frac{1}{Q(s,a)\cdot\tau}\right]}{\sum_{(b\in A)} exp\left[-\frac{1}{Q(s,b)\cdot\tau}\right]} \qquad [7]\ (6)$$

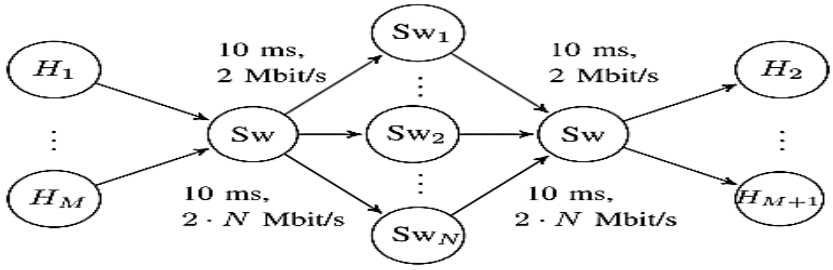
من أجل محاكاة سلوك Softmax لنطاقات القيم السالبة. لا تصل قيم Q أبداً إلى الصفر، في المعادلة (6)، الاحتمالات غير صفرية للتهيئة $Q(s,a) = -\infty$. سيؤدي

هذا إلى تجربة جميع تركيبات الحالة–الإجراء، لأن الاحتمالات لـ Q(s,a) = 0 هي الأعلى. هذا الاستكشاف القسري لجميع تركيبات الحالة–الإجراء، ومع ذلك، سيتناقض مع الفكرة أن الاستكشاف يُتحكم به بواسطة درجة الحرارة τ [7].

10. القابلية للتوسع (SCALABILITY)

القابلية للتوسع تمثل تحدياً رئيسياً في التوجيه المستند إلى التعلم المعزز. ندرس القابلية للتوسع مع الطوبولوجيا في الشكل 4 ذات N مفتاح وسيط وM تدفق، بحيث تدعم روابط المبدلة الوسيطة Swi سرعة نقل تبلغ 2 · i ميغابت/ثانية ولكل تدفق f_i من المضيف H_i إلى H_{i+1} طلب عرض نطاق ترددي يبلغ 2 · i ميغابت/ثانية (ناقص هامش 250 كيلوبت/ثانية). يمكن استنتاج التوزيع الأمثل للتدفقات بسهولة لهذه الطوبولوجيا، أي f_1 عبر Sw1، f_i عبر Swi، وهكذا. بالنسبة لهذه الطوبولوجيا، يمكننا حساب حجم جدول Q لتغيير تدفق واحد كما يلي [7]:

$$\underbrace{|Q(s,a)|}_{\text{(Size of Q-table)}} = \underbrace{N^{\wedge}M}_{\text{(Number of states)}} * \underbrace{([(N-1)\cdot M+1])}_{\text{(Number of actions)}} \quad [7] \ (7)$$



شكل 4: طوبولوجيا تقييم قابلية التوسع مع N من المبدلات الوسيطة و M من التدفقات

11. متحكم Ryu في شبكاتSDN

متحكم Ryu هو إطار عمل مفتوح المصدر مخصص للتحكم البرمجي في الشبكات (SDN)، ويسمح للمطورين بكتابة تطبيقات شبكية متقدمة تسهّل إدارة الشبكات ديناميكيًا ومرونة أكبر من الشبكات التقليدية [22] [23] [24].

- **تعريف** Ryu :هو متحكم شبكي موجه لبيئة SDN ، مُبرمج بالكامل بلغة Python، ويوفر مجموعة من الواجهات البرمجية Application Programming Interface (APIs) تسهّل للمطورين بناء حلول لإدارة وتوجيه البيانات في الشبكة عبر بروتوكولات مثل OpenFlow وغيرها.
- **الوظيفة الأساسية** : إصدار قرارات ديناميكية حول كيفية مرور البيانات بين أجهزة الشبكة (المفاتيح – السويتشات)، عبر إرسال تعليمات مباشرة لها.
- **الدعم البرمجي** : يدعم Ryu كتابة تطبيقات شبكية بسهولة بسبب بساطة Pythonومرونة البرمجة الحدثية(event-driven) ، ويمكن تشغيل التطبيق عبر أداةryu-manager [23][25] .

المميزات:

- **واجهة برمجة تطبيقات واضحة** :يُبسط عمليات تطوير التطبيقات الشبكية، ويحتوي على دوال ووحدات جاهزة للتعامل مع أحداث الشبكة ورسائل OpenFlow [23][25].
- **تعدد البروتوكولات** :يدعم نسخ متعددة من OpenFlow (1.0 حتى 1.5)، ويدعم بروتوكولات أخرى مثل Netconf و OF-config [23].
- **مرونة الاختبار والبحث** :مثالي للبيئات البحثية والأكاديمية بفضل سهولة إنشاء ومحاكاة الشبكات باستخدام أدوات مثل Mininet.

1. كل تطبيق Ryu هو سكربت Python يرث من الصنف RyuApp.
2. يمكن لأي تطبيق مراقبة الأحداث الشبكية (مثل وصول رزمة، تغيير حالة منفذ).
3. يُرسل Ryu الرسائل والتعليمات للمفاتيح مباشرة باستخدام OpenFlow [25][23].

12.مقاييس الأداء (PERFORMANCE METRICS)

نستخدم المقاييس التالية في تقييمنا، وهي [7]:

1.12 الوقت حتى التقارب (Time till convergence)

يصف الوقت حتى التقارب مدى سرعة عثور وكيل التعلم المعزز على حالة زمن استجابة منخفض. يجب ألا يستغرق هذا البحث عن حالة زمن استجابة منخفض وقتاً طويلاً، لأنه في الواقع وفي محاكاتنا، لا يمكن إجراء تسريع (على سبيل المثال، من خلال أجهزة حوسبة أسرع)؛ بل إن مدة البحث مرتبطة بشكل جوهري بتصميم خوارزمية البحث، أي خوارزمية التعلم المعزز ومدة خطوة زمنية (أي الفترة الزمنية لتكرار تعلم واحد). لذلك، يجب على وكيل التعلم المعزز أن يتعلم بسرعة وكفاءة قدر الإمكان. لا يوجد تعريف شائع للتقارب في التعلم المعزز. من أجل تقييم أوقات التقارب كمياً، قمنا أولاً بتنعيم أزمن الاستجابة المتوسطة المقيسة للتدفقات بمتوسط متحرك لـ N =10 خطوة متتالية. بعد ذلك، حسبنا الفروق المحدودة لأزمنة الاستجابة المتوسطة المُنعّمة. إذا كانت التقلبات، أي الفروق المحدودة لأزمن الاستجابة المتوسطة المُنعّمة، أصغر من قيمة عتبة محددة تبلغ 0.4 مللي ثانية/خطوة، فإننا نعتبر النظام متقارباً [7].

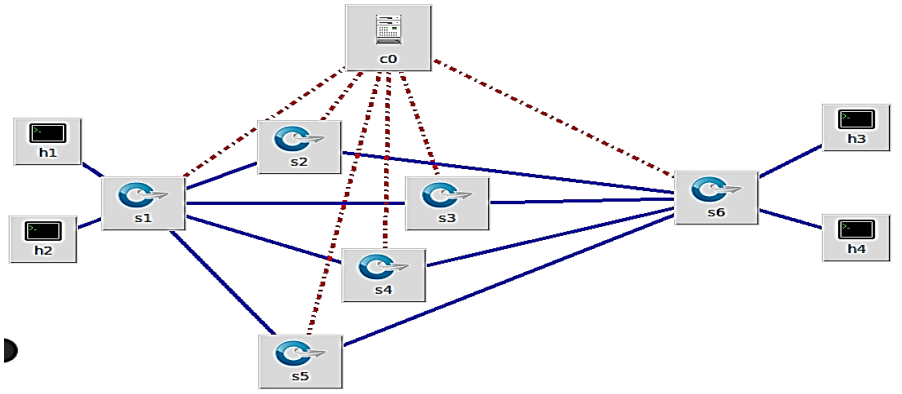
2.12 متوسط زمن استجابة التدفق (AVERAGE FLOW LATENCY)

يُعد زمن الاستجابة مؤشراً مفيداً لأداء الشبكة حيث يكشف عن الازدحام، الذي لا يزيد زمن الاستجابة فحسب، بل يقلل من معدل النقل أيضاً. وبما أنه في الشبكة الحقيقية لن نعرف أزمنة الاستجابة التي تمر بها الحركة الفعلية، فإننا نقيس أزمنة استجابة التدفقات عبر حزم الاستطلاع. نحدد متوسط زمن الاستجابة كمتوسط زمن استجابة التدفق غير الموزون عبر جميع التدفقات على طول مساراتها respective من المصدر إلى الوجهة [7].

13.التنفيذ العملي والنتائج

سوف نقوم بإجراء اختبار وذلك لبنية شبكة مشابهة للشكل 4 كما هي ظاهرة في الشكل (5) وذلك باستخدام متحكم واحد ثم سنقوم بفصل الشبكة الى نطاقين وذلك باستخدام متحكمين بحيث كل متحكم يكون مسؤول عن نطاق.

1.13 السيناريو الأول: شبكة باستخدام متحكم واحد



الشكل 5: شبكة باستخدام متحكم واحد مسؤول عن إدارة التدفقات لكامل الشبكة.

تم استخدام الشيفرة التالية لتوليد تدفق باستخدام أداة iperf بين كل زوج متقابلين من الأجهزة حيث ولد تدفق بين h1 ,h3 وتدفق بين h2,h4:

```python
def startIperf(host1, host2, amount, port, timeTotal, loadLevel):
    info(f"Killing old iperf clients on {host1.name}...\n")
    host1.cmd('pkill -f "iperf -c"') # pkill -f matches full command line
        # Give it a moment to terminate
    time.sleep(0.1)
    bw = float(amount) * (float(loadLevel) / float(10))
    print("Host {} to Host {} Bw: {}".format(host1.name, host2.name, bw))
    command = "iperf -c {} -u -p {} -t {} -b {}M &".format(host2.IP(), port, timeTotal, bw)

    host1.cmd(command)
```

host1: المضيف الذي سيُستخدم كـ client (يرسل البيانات).

host2: المضيف الذي سيُستخدم كـ server (يستقبل البيانات).

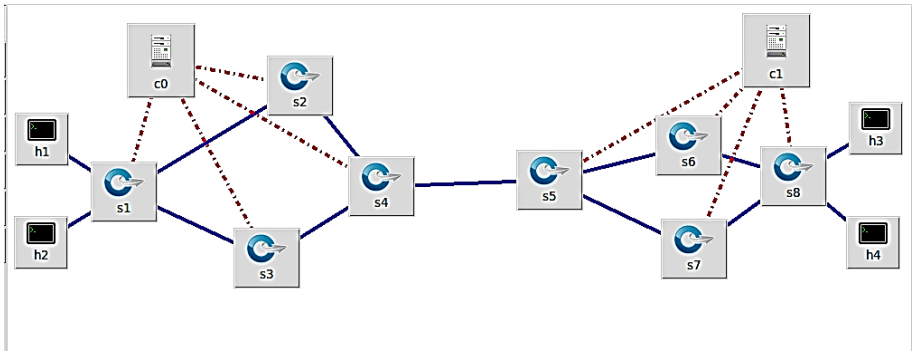
amount: قيمة أساسية لعرض النطاق الترددي (حسب المعادلة 2.0*j+1.75)، تُستخدم لحساب السرعة الفعلية.

port: المنفذ (port) الذي سيُستخدم في الاتصال (5001).

timeTotal: المدة الزمنية (بالثواني) وتم اختيار (20 دقيقة) التي سيُرسل فيها العميل البيانات.

loadLevel: مستوى الحمل (10)، يُستخدم لتعديل السرعة.

2.13 السيناريو الثاني: شبكة باستخدام متحكمين



الشكل 6: شبكة باستخدام متحكمين كل متحكم مسؤول عن توجيه التدفق داخل نطاقه.

وذلك باستخدام نفس اعدادات التدفق باستخدام متحكم واحد حيث يتم ارسال التدفق من النطاق الأول الى الثاني وبالعكس.

3.13 تقييم النتائج

1. الشبكة الأصلية (قبل التقسيم الشكل 5):

- الوصف:

في الشبكة الأصلية، جميع العقد (`s1`, `s2`, `s3`, `s4`, `s5`, `s6`) تُدار بواسطة متحكم وحيد (`c0`).

- المشكلة الرئيسية:

- مساحة الحالة-الفعل (`State-Action Space`):

مع زيادة عدد التدفقات (`flows`) النشطة، يزداد حجم المساحة الحالة-الفعل بشكل كبير.

- وقت التقارب (`Convergence Time`):

يتطلب استكشاف هذه المساحة الكبيرة وقتاً طويلاً للتعلم.

2. الشبكة بعد التقسيم إلى دومينات وذلك لتحقيق هدف البحث (الشكل 6) :

-الوصف:

تم تقسيم الشبكة إلى دومينين:

- الدومين الأول يحتوي على العقد (`s1`, `s2`, `s3`, `s4`) ويُدار بواسطة المتحكم (`c0`).

– الدومين الثاني يحتوي على العقد (`s8` ,`s7` ,`s6` ,s5) ويُدار بواسطة متحكم آخر (`c1`).

– توزيع المسؤوليات:

كل متحكم يركز على إدارة عدد أقل من العقد والتدفقات داخل نطاقه (`domain`)، مما يؤدي إلى:

– مساحة حالة–فعل أصغر:

بسبب عدد العقد والتدفقات الأقل، يتم تخفيض حجم مساحة الحالة–الفعل لكل متحكم.

– وقت تقارب أسرع:

مع مساحة حالة–فعل أصغر، يتم تحقيق الاستقرار (`convergence`) بسرعة أكبر.

3.السبب الأساسي لتقسيم الشبكة إلى دومينات هو تقليل مساحة الحالة–الفعل (`State–Action Space`):

– في الشبكة الأصلية:

– المساحة الحالة–الفعل كبيرة جدًا بسبب عدد العقد الكبير (s1, s2, s3, s4, s5, s6) وعدد التدفقات النشطة في مثالنا هما تدفقين.

– هذا يؤدي إلى مشكلة في التخزين (لتخزين الجدول Q) والتعلم (لاستكشاف المساحة الكبيرة).

– في الشبكة المقسمة:

– تم تقسيم الشبكة إلى دومينين، مما خفض عدد العقد التي يديرها كل متحكم.

– كل متحكم يتعامل مع مساحة حالة–فعل أصغر، مما يقلل من متطلبات التخزين ويزيد من سرعة التقارب.

وقت التقارب (`Convergence Time`):

– في الشبكة الأصلية:

– يحتاج المتحكم (`c0`) إلى استكشاف مساحة حالة–فعل كبيرة جدًا، مما يجعل وقت التقارب (`Time till Convergence`) طويلاً.

– في الشبكة المقسمة:

– كل متحكم يتعامل مع مساحة حالة–فعل أصغر، مما يؤدي لاستكشاف المساحة وتحقيق الاستقرار بشكل أسرع.

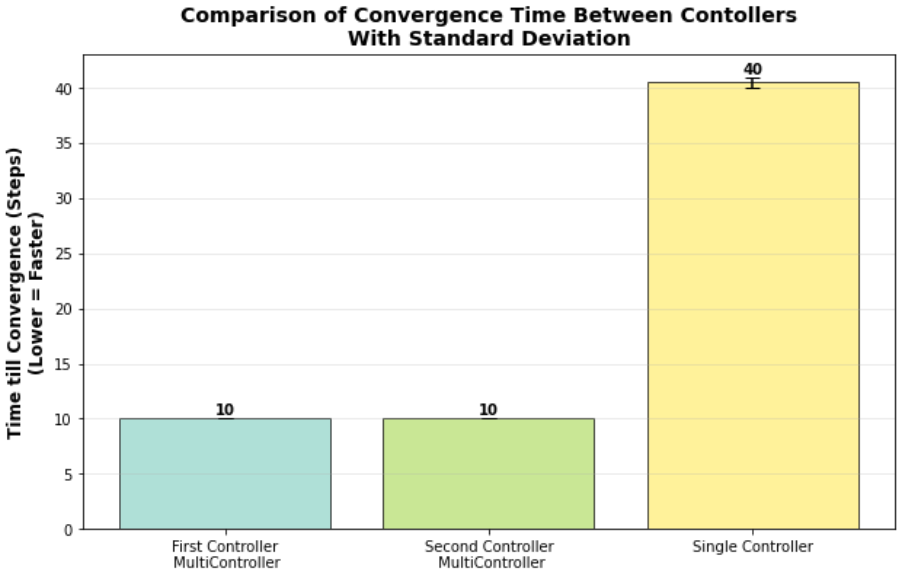
تخفيض العبء الحسابي:

– في الشبكة الأصلية:

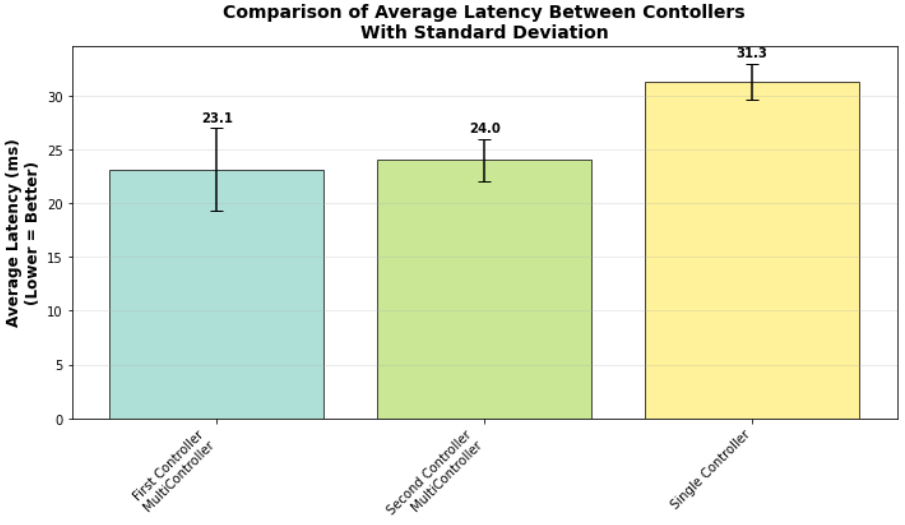
– المتحكم (`c0`) يتحمل عبءً كبيرًا لإدارة جميع العقد والتدفقات.

– في الشبكة المقسمة:

– كل متحكم يركز على نطاق صغير من العقد والتدفقات، مما يخفف العبء الحسابي على كل متحكم.



الشكل 7: نتائج وقت التقارب لمتحكم واحد (الشكل 5) ومتحكمين (الشكل 6).



الشكل 8 : نتائج متوسط التأخير لمتحكم واحد (الشكل 5) ومتحكمين (الشكل 6).

1. Single Controller: يشير إلى نظام يحتوي على متحكم واحد فقط، وهو المسؤول عن إدارة الشبكة بالكامل (شبكة غير مقسمة إلى نطاقات/دومينات).

2. First Controller: يشير إلى المتحكم الأول في نظام يحتوي على نطاقين (دومينين)، وهو المسؤول عن إدارة النطاق الأول.

3. Second Controller: يشير إلى المتحكم الثاني في نفس النظام، وهو المسؤول عن إدارة النطاق الثاني.

4. المعادلة الجديدة لحساب حجم q-table

عدد السويتشات لكل متحكم يصبح $\frac{N}{k}$

حجم الجدول Q لكل متحكم:

$$\underbrace{|Q(s,a)|}_{\text{(Size of 1 controller)}} = \underbrace{(\frac{N}{K})\hat{}M}_{\text{(Number of states)}} * \underbrace{\left(\left[\left(\frac{N}{K}\right)-1\right)\cdot M+1\right]}_{\text{(Number of actions)}}$$

الحجم الإجمالي للجداول Q:

$$\underbrace{|Q(s,a)|}_{\text{(total Size)}} = k. \underbrace{|Q(s,a)|}_{\text{(Size of 1 controller)}} \quad (8)$$

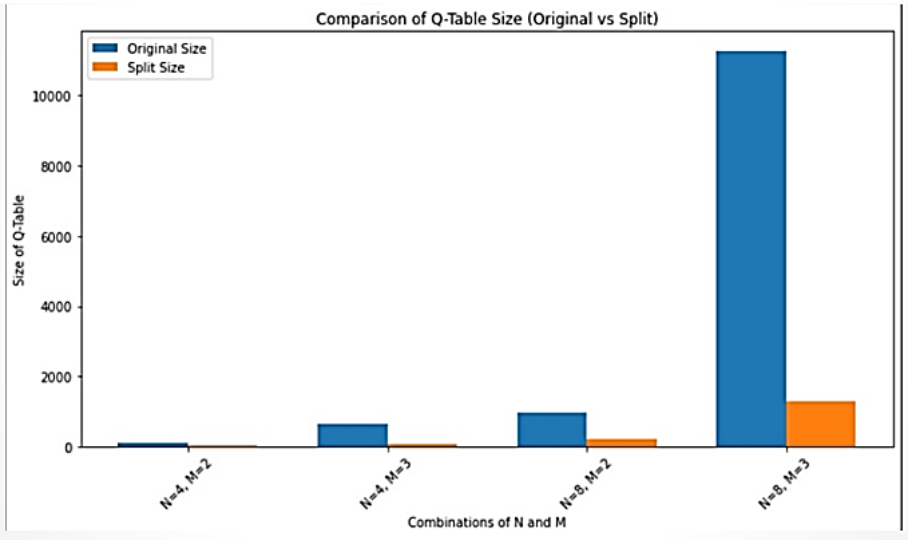
حيث ان :

N: عدد السويتشات الوسيطة .

M : عدد التدفقات.

K: عدد المتحكمات.

ومنه نصل الى النتائج التالية من اجل k=2:



الشكل 8: حجم جدول Q في حال وجود متحكم واحد ومتحكمين كل واحد مسؤول عن مجال خاص به وذلك لقيم مختلفة لكل من N,M.

14.الاستنتاجات والأعمال المستقبلية:

تم تقسيم الشبكة إلى دومينات لمعالجة مشكلة التوسع (`scalability`) المرتبطة بتزايد عدد التدفقات النشطة. في الشبكة الأصلية، كان المتحكم (`c0`) مسؤولاً عن إدارة جميع العقد (`s1`, `s2`, `s3`, `s4`, `s5`, `s6`)، مما أدى إلى مساحة حالة–فعل كبيرة ووقت تقارب طويل. بعد التقسيم إلى دومينين، أصبح كل متحكم يركز على نطاق صغير من العقد، مما خفض حجم المساحة الحالة–الفعل وزمن التقارب. النتائج أظهرت أن الشبكة المقسمة حققت زمن تقارب أسرع ومتوسط زمن وصول أقل مقارنة بالشبكة الأصلية.

استكشاف استراتيجيات ذكية أكثر لاستكشاف المسارات بهدف تعزيز قابلية التوسع. على سبيل المثال، قد تقتصر عملية استكشاف المسارات على الوصلات ذات الاستخدام العالي (التي تكون أكثر عُرضة للازدحام)، أو تُركّز فقط على التدفقات ذات المعدّل المرتفع، بينما يُوجَّه الجزء الأكبر من التدفقات ذات المعدّل المنخفض باستخدام توجيه تقليدي (مثل أقصر مسار)، أو باستخدام أساليب بديلة تُخفّف الضغط عن الوصلات المزدحمة (كالتوجيه عبر أطول المسارات مثلاً). كما أن العديد من التدفقات في الشبكات التشغيلية لا تتطلب زمن وصول قصير، بل تكتفي باتصال موثوق حتى لو تحمّلت تأخيرات طويلة. ويمكن في هذه الحالة توجيه هذه التدفقات المتسامحة مع التأخير (delay-tolerant flows) باستخدام خوارزميات توجيه تقليدية، مما يتيح اقتصار استخدام استراتيجيات الاستكشاف المعتمدة على التعلّم المعزز فقط على التدفقات الحساسة للزمن (low-latency flows).

في هذه الدراسة، تم تنفيذ آلية أولية لتبادل جداول التوجيه بين المتحكمات المتعددة عبر واجهة برمجة التطبيقات (REST API)، بهدف تمكين التنسيق البيني بين النطاقات المستقلة. ومع ذلك، لم يتم استغلال هذه الاتصالات بشكل كامل لتحسين قرارات التوجيه على مستوى الشبكة ككل. في العمل المستقبلي، يمكن تطوير بروتوكولات تعاون ذكية تتيح للمتحكمات مشاركة معلومات حيوية مثل: حالة الازدحام في الوصلات الحدودية، أو التنبؤات بسلوك التدفقات العابرة (inter-domain flows)، أو حتى سياسات Q-values المُتعلّمة ذات الصلة بالمسارات المشتركة. ومن خلال دمج هذه المعلومات في عملية اتخاذ القرار، يمكن لكل متحكم أن يُحسّن توجيه التدفقات العابرة لحدود نطاقه، ويقلل من التأخير الكلي وتجنب الازدحامات المتكررة عند نقاط التقاطع بين الأنظمة الذاتية (ASes). كما يمكن استغلال قناة الاتصال بين المتحكمات لتنفيذ استراتيجيات توجيه متناسقة، مثل توزيع التدفقات عالية المعدّل عبر مسارات متكاملة تمر بعدة نطاقات دون تعارض، أو تمكين استجابة

جماعيـــة للتغيـــرات المفاجئـــة فـــي الأحمـــال. وبهـــذا، يتحـــول النظـــام مـــن مجموعـــة مـــن المتحكمـــات المســـتقلة إلـــى شـــبكة متعاونـــة مـــن الـــوكلاء الـــذكيين، قـــادرة علـــى تحقيـــق أداء شمولي يفوق مجموع أداء أجزائها الفردية.

15. المراجع

1. Ahuja, Ravindra K., et al. Network Flows: Theory, Algorithms, and Applications. Prentice Hall, 1993.
2. Maugendre, Marion. Development of a Performance Measurement Tool for SDN. 7 Oct. 2015.
3. Lantz, Bob, et al. Introduction to Mininet. Edited by Bob Lantz, 13 Feb. 2017.
4. Linkletter, Brian. "How to Use MiniEdit, Mininet’s Graphical User Interface." Brian Linkletter, 2 Apr. 2015.
5. Sharma, Shubham. Accurate Traffic Generation in the CheesePi Network. Master’s thesis, KTH Royal Institute of Technology, 2017.
6. Fang, Chao, et al. "Research on Routing Algorithm Based on Reinforcement Learning in SDN." Journal of Physics: Conference Series, vol. 1284, Aug. 2019, p. 012053.
7. Rischke, Julius, et al. "QR-SDN: Towards Reinforcement Learning States, Actions, and Rewards for Direct Flow Routing in Software-Defined Networks." IEEE Access, vol. 8, 2020, pp. 174773–174791.
8. Casas-Velasco, David M., et al. "Intelligent Routing Based on Reinforcement Learning for Software-Defined Networking." IEEE Transactions on Network and Service Management, vol. 18, no. 1, Mar. 2021, pp. 870–881.

9. Jalil, S. Q., et al. "DQR: Deep Q-Routing in Software Defined Networks." Proceedings of the International Joint Conference on Neural Networks (IJCNN), Jul. 2020, pp. 1–8.
10. Etengu, R., et al. "AI-Assisted Framework for Green-Routing and Load Balancing in Hybrid Software-Defined Networking: Proposal, Challenges and Future Perspective." IEEE Access, vol. 8, 2020, pp. 166384–166441.
11. Open Networking Foundation. Open Networking Foundation, 2014.
12. Bannour, Fetia, et al. "Distributed SDN Control: Survey, Taxonomy, and Challenges." IEEE Communications Surveys & Tutorials, vol. 20, no. 1, 2017, pp. 333–354.
13. Benzekki, Kamal, et al. "Software-Defined Networking (SDN): A Survey." Security and Communication Networks, vol. 9, no. 18, 2016, pp. 5803–5833.
14. Waseem, Q., et al. "Software-Defined Network (SDN) Forensic." Symmetry, vol. 13, no. 5, 2021, pp. 767–785.
15. Sutton, Richard S., and Andrew G. Barto. Reinforcement Learning: An Introduction. 2nd ed., MIT Press, 2018.
16. Xie, Junfeng, et al. "A Survey of Machine Learning Techniques Applied to Software Defined Networking (SDN): Research Issues and Challenges." IEEE Communications Surveys & Tutorials, vol. 21, no. 1, 2018, pp. 393–430.
17. Jang, Beomseok, et al. "Q-Learning Algorithms: A Comprehensive Classification and Applications." IEEE Access, vol. 7, 2019, pp. 133653–133667.
18. Francois-Lavet, Vincent, et al. "An Introduction to Deep Reinforcement Learning." Foundations and Trends in Machine Learning, vol. 11, nos. 3–4, 2018, pp. 219–354.
19. Sachs, J., et al. "Adaptive 5G Low-Latency Communication for Tactile Internet Services." Proceedings of the IEEE, vol. 107, no. 2, Feb. 2019, pp. 325–349.

20.Xiang, Z., et al. "Reducing Latency in Virtual Machines: Enabling Tactile Internet for Human-Machine Co-Working." IEEE Journal on Selected Areas in Communications, vol. 37, no. 5, May 2019, pp. 1098–1116.
21.Lin, Shih-Chun, et al. "QoS-Aware Adaptive Routing in Multi-Layer Hierarchical Software Defined Networks: A Reinforcement Learning Approach." Proceedings of the IEEE International Conference on Services Computing (SCC), Jun. 2016, pp. 25–33.
22.Montazerolghaem, Ahmadreza, and Somaye Imanpour. "Evaluation and Performance Analysis of the Ryu Controller in Various Network Scenarios." arXiv, 25 May 2025, arxiv.org/abs/2505.19290.
23."Performance Evaluation of Ryu Controller in Software Defined Networks." Journal of Al-Qadisiyah for Computer Science and Mathematics, vol. 14, no. 1, 2022, pp. 1–7, doi.org/10.29304/jqcm.2022.14.1.879.
24.Chauhan, Pinkey, and Mithilesh Atulkar. "Ryu Controller-Based Attack Detection and Mitigation Method in Software Defined Internet of Things." International Journal of Engineering Trends and Technology, vol. 71, no. 9, 2023, pp. 138–156, doi.org/10.14445/22315381/IJETT-V71I9P213.
25."POX and RYU Controller Performance Analysis on Software Defined Networks." EAI Endorsed Transactions on Internet of Things, vol. 9, no. 1, 2023, publications.eai.eu/index.php/IoT/article/view/2821.
26.Ashok, R., et al. "iPerf—The Ultimate Speed Test Tool for TCP, UDP and SCTP." Journal of Computer and Communications, vol. 8, no. 9, 2020, pp. 11–19, www.scirp.org/journal/paperinformation?paperid=1026461 .
27.NetBeez. "iPerf and Network Performance Testing." NetBeez Blog, 2023, netbeez.net/blog/how-to-use-iperf/.

28.Oracle Corporation. "Use iPerf to Test the Throughput Inside an OCI Hub and Spoke VCN Routing Architecture." Oracle Documentation, 2023, docs.oracle.com/en/learn/iperf-testing-in-oci/index.html.
29.Zamfir, Radu, et al. "Mobile Network Operators' Assessment Based on Drive-Test Scenarios Using Open-Source Tools." Applied Sciences, vol. 14, no. 3, 2024, p. 1268, doi.org/10.3390/app14031268.