清華大學資訊工程系109學年度上學期專題報告

專題名稱	Routing multiple vehicles in road networks with constrained communication capability	
學號	106062212	106062132
姓名	郭蕙綺（組長）	趙仰生（組員）

摘要

隨著Deep Learning在各個領域的興起，越來越多衍生的技術蓬勃發展，而Reinforcement Learning更是其中不可或缺的一環，我們可以看到從Alpha Go打敗世界棋王，到自走車甚至是工業的應用，此項技術已經變成了資工領域非常熱門的研究主題，近年來，更多研究旨在運用RL來解決圖論問題，了解此項技術的運用，以及深入談討其在圖論問題的效能，甚至是能否擊敗傳統演算法，成為我們此次專題研究的主要方向。

研究動機及目的

從接觸程式到學習資料結構以來，我們一直都是透過傳統演算法來解決圖論的問題，其中複雜的邏輯往往相當棘手，而對於Reinforcement learning有著好奇心的我們，在學習的過程中發現，利用RL來解決類似的問題時，我們並不需要設計以往傳統的演算法，而是需要訓練一個模型，簡單來說，就是訓練一個能夠幫助我們解決問題的模型，由於對於此種解決問題的方法感到非常有興趣，因此選擇此題目當作我們的專題。

現有相關研究概況及比較

現今許多論文在探討如何運用RL來解決圖論問題，例如：Vehicle Routing Problem以及Traveling Salesman Problem，其設計模型的架構，以及Reward Function的給定，給了我們非常多靈感，但是在這麼接近實際生活以及這麼多限制的情況下，還真是少之又少，所以在過程中許多環境的設計與實作細節，都是由我們與教授討論出來的結果。
在效能方面，Reinforcement Learning著實表現不錯，不過其需要的訓練時間也會隨著訓練規模變大而增加，對比傳統演算法，只需要完整執行一遍的運算時間，是未來必須要解決的課題。

設計原理

運用deep reinforcement learning中的Q-Learning來實作，其中DRL的概念如圖1，神經網路代表agent，將看到的state輸入到神經網路後，根據輸出的值決定下一個action，而環境接收到action後會給出reward及下一個state的值；再利用Double Q learning 的技巧，如圖2，其中Q value代表的是 “在某個state s，採取某個action a，根據policy pi，直到整個環境結束會得到的reward總和”的預估值，利用兩個神經網路，一個是用來持續更新的網路，另一個是目標網路，讓整個訓練過程不會一直追逐一個持續變動的目標，由於兩個網路的輸入為前後的state，因此，輸出的值應相差一個reward，讓agent持續與環境做互動，根據獲得的reward來更新神經網路，而目標網路則是在一段時間後，使其與持續更新的網路相等即可。

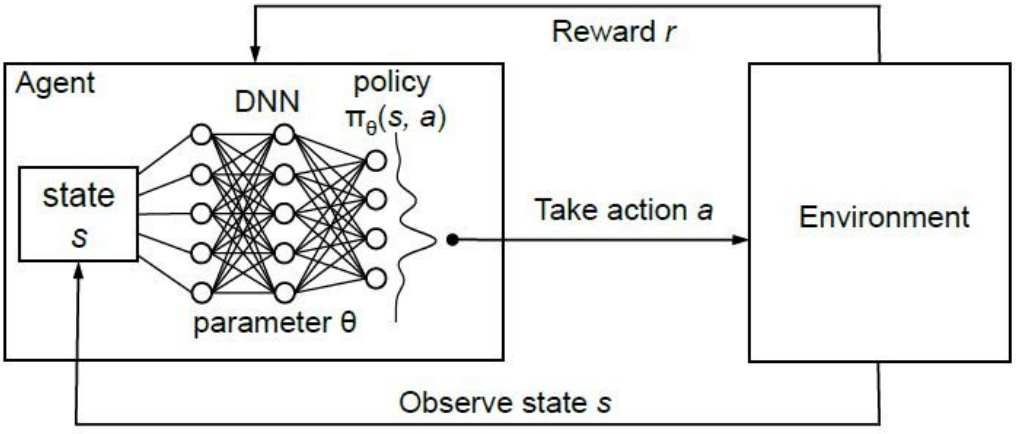

圖1

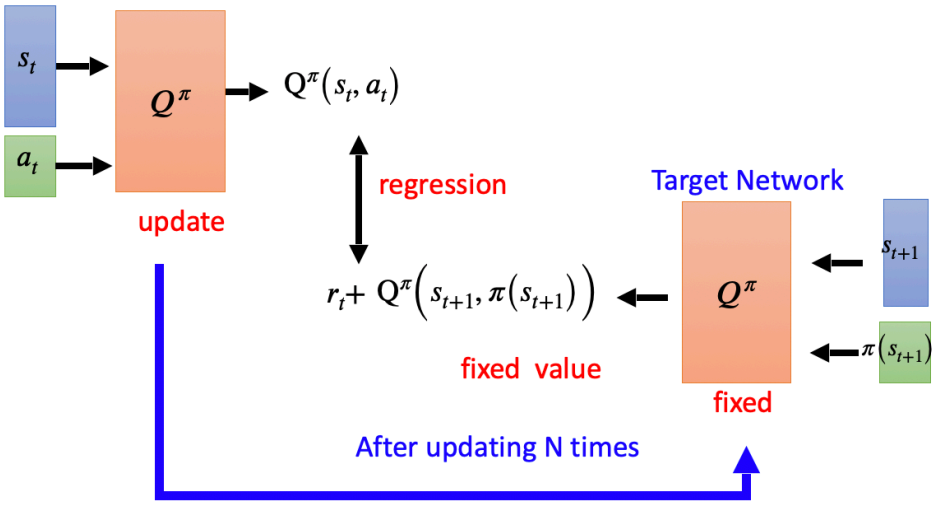

圖2

研究方法與步驟

- 問題描述：給定一張graph，運用deep reinforcement learning讓agent之間溝通並合作，找出能覆蓋全部edge的路徑，並使之花費的時間最少。
- Input：給定edge長度、agent數量及其起始點、agent速度的graph。
- Output：每個agent的路徑、總共花費的iteration數。
- 訓練過程：首先，利用隨機選擇的方式建立環境，確認環境的正確性後將agent加入，將state輸入至agent的網路，根據結果選擇下一條路徑，而環境根據agent的選擇，給出相對應的reward及下一個state，利用Double Q Learning重複訓練，直到reward收斂。
- 模型架構：塔型架構 (number of nodes, 1024, 512, 256, 128, 64, 32, 16, number of nodes)，此架構代表的意義為將特徵從low level到high level一步一步抽取出來，為Deep Learning常用的架構。
- 激活函數：ReLu。
- Optimizer：AdamW。
- 演算法：Double deep Q learning。
- State的設計：state是一個大小為點數量的一維陣列，如果agent所在的點與相連的點是被走過的，值為1，反之則為0。
- Reward的設計：如表1。

時間點	Reward 設計	代表意義
每一次選擇	Reward -= （點與點長度）* 0.001	每一步都扣分，希望Agent 越快完成任務越好。
每一次選擇	如果走到沒走過的點，Reward += 1	希望Agent可以多走一些沒走過的路。
整個環境結束	完成任務，Reward += 10	完成時，給Agent正向的獎勵。
整個環境結束	如果所花時間小於貪婪演算法的時間，Reward += 30	告訴Agent當次epoch是好的。

表1

系統實現與實驗

由於Reinforcement是不穩定的，任何一個環節的參數或是設計出了問題都有可能導致訓練任務失敗無法收斂，因此我們嘗試了許多不同的參數設計，包括：Learning Rate、Algorithm、Reward Function、State，如表2。

而經由長時間agent與環境的互動以及訓練，我們成功打敗了greedy演算法，見圖3。

不過在scale變大之後，效能會逐漸降低，是未來需要解決的問題，見圖4。

參數調整	調整前	調整後	代表意義
Learning rate	0.001	0.00002	learning rate一開始太大，會導致loss發散而無法收斂。
Algorithm	Deep Q learning	Double deep Q learning	Double deep Q learning能降低Q-value被高估的幅度，使訓練更穩定。
State	較複雜的表示法	用1,0來表示	原先希望給agent更多資訊，但卻無法收斂，簡易的表示法卻訓練成功，由此可知，在RL 中，不一定是越複雜的state越能訓練出好的模型。
Reward function	利用較複雜的條件及變數 例：RW -= cost	簡化條件，將變數改為定數 例：RW -= 25	調整前的reward function最後無法收斂，調整後的卻收斂了，如同state的發現，有時候較簡單的表示法較易成功。

表2

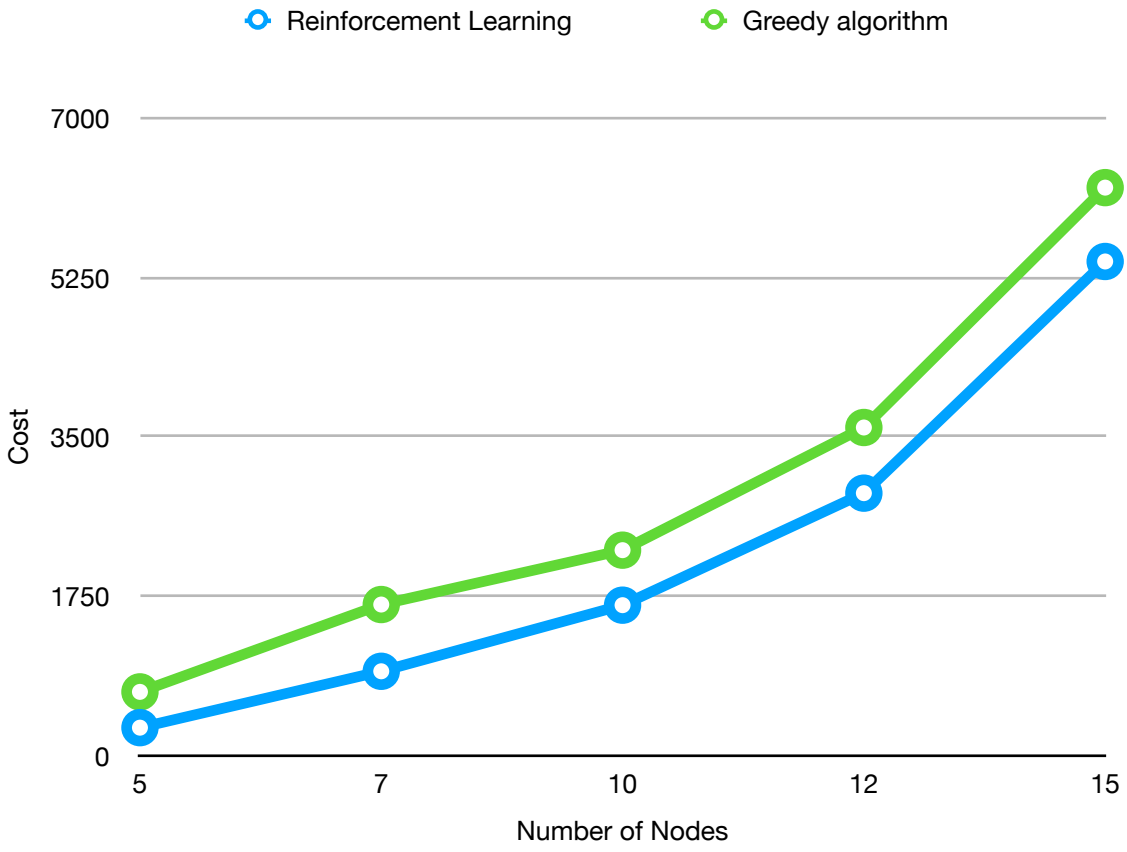
Reinforcement Learning
Greedy algorithm
7000
5250
3500
1750
0
Cost
5
7
10
12
15
Number of Nodes

圖3

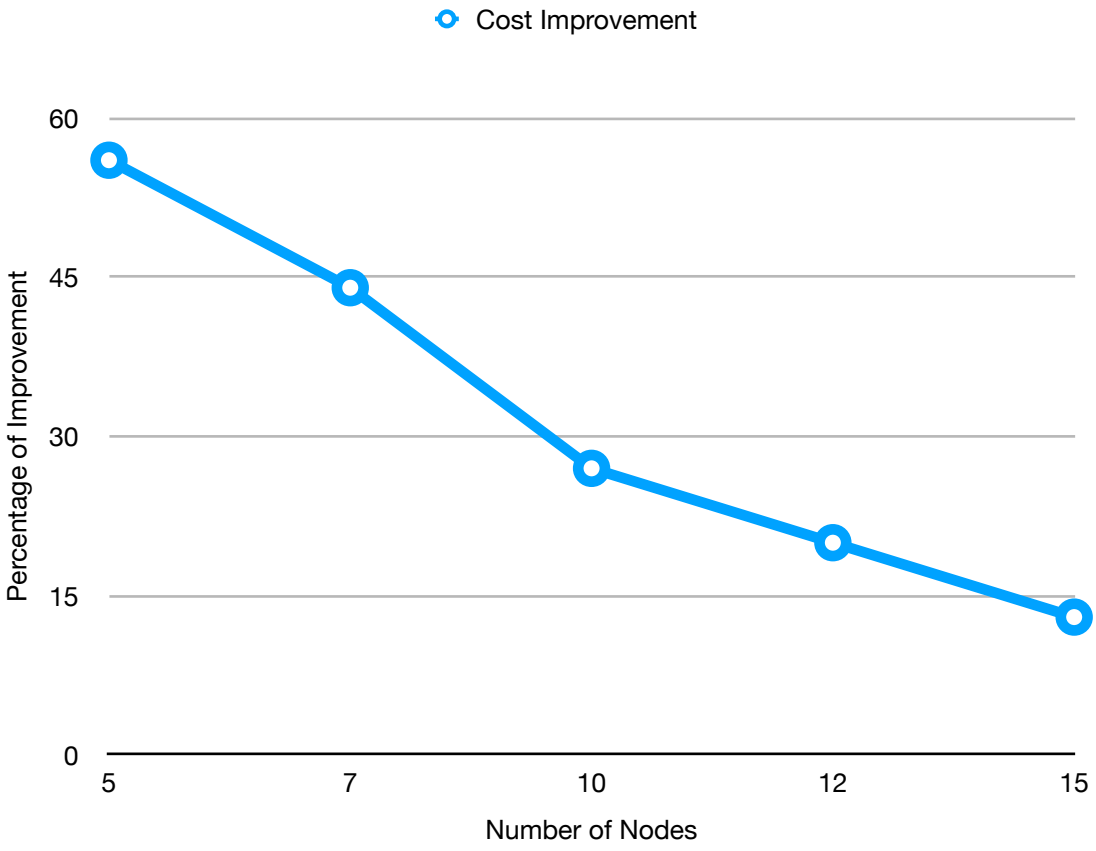
Cost Improvement
60
45
30
15
0
Percentage of Improvement
5
7
10
12
15
Number of Nodes

圖4

專題重要貢獻

我們的專題是運用RL來解決類似Vehicle Routing的問題，不過更加接近現實生活實際的情況，例如：車子有自身的速度、道路有不一樣的長度以及在一定的距離以內車子才能互相溝通，此運用未來能夠實現在許多現實生活中的路徑問題，例如：

- 無人鏟雪車：假定有一個地區下大雪，地方政府只有一定數量的鏟雪車，運用我們訓練的模型，將地圖轉成graph後與鏟雪車的所在位置一同輸入，希望能找出最有效率的方法將任務完成。
- 無人巡邏車：概念與鏟雪車相似。

如此一來，能夠將路徑相關的問題交給機器人執行，節省下更多人力資源，也能擁有更佳的路經規劃，使執行任務的效能提升。

團隊合作方式

由於我們的小組只有兩個人，因此所有的任務幾乎都是我們合力完成的，包括環境、模型架構、系統測試、實驗以及報告。大部分的時候，透過討論來分工合作，並且互相分享彼此學習到的經驗與技巧，當意見有分歧的時候，例如：Reward Function的給定或是參數的設定，我們就會各自做實驗並且比較結果與效能，最後討論出比較好的方案，這樣不僅能夠提升工作的效率，也能獲得互相討教學習的機會。

效能評估與成果

我們訓練出的模型，根據實驗結果可以看出來，對於使用Reinforcement Learning解決Routing問題是非常有潛力的，能夠在小sclae的情況下，擊敗greedy演算法，不過當scale變大時，效能逐漸降低或是訓練較難以收斂，如果想要解決此問題，未來必須抽取更多圖的特徵以及進行更多參數調整。

結論

此次專題學到了非常多的東西，從RL最基本的概念，到訓練模型甚至是建構環境，都由我們自己一步一步刻出來，也完成了一些成果，對於此領域的涉獵也幫助許多，未來還需要學習更多相關的知識才能更加優化此模型。也學會如何更好的在團隊裡面分工以及討論，不只是各做各的事情，而是將經驗互相分享，一同解決問題，才能使團隊合作的效益發揮到最大。

參考文獻

1. https://en.wikipedia.org/wiki/Reinforcement_learning
2. https://en.wikipedia.org/wiki/Vehicle_routing_problem
3. Mnih, Volodymyr, et al. “Playing atari with deep reinforcement learning.” *arXiv preprint arXiv:1312.5602.* 2013.
4. Khalil, Elias, et al. “Learning combinatorial optimization algorithms over graphs.” *NeurIPS*. 2017.
5. Mohammadreza Nazari, Afshin Oroojlooy, Lawrence V Snyder, and Martin Tak´aˇc. Deep reinforcement learning for solving the vehicle routing problem. arXiv preprint arXiv:1802.04240, 2018
6. B.-Y. Hsu, C.-Y. Shen, G.-S. Lee, Y.-J. Hsu, C.-H. Yang, C.-W. Lu, M.-Y. Chang, and K.-P. Lin, ``Optimizing k-Collector Routing for Big Data Collection in Road Networks,'' IEEE Global Commuication Conference (Globecom), 2019.
7. Wouter Kool, Herke van Hoof, and Max Welling. Attention, learn to solve routing problems! In International Conference on Learning Representations, 2019.