

UNIVERSITA' LA SAPIENZA

DIPARTIMENTO DI INFORMATICA

AUTONOMOUS NETWORKING

Report Secondo Homework

Author:

Giordano DIONISI

1834919

Mattia LISI 1709782


Michele SPINA 1711821

Supervisor:

Prof. Andrea COLETTA

Prof.ssa Gaia MASELLI

4 dicembre 2021



./Images/logo_firstpage.png



▣./Images/logo_hidden.png

Indice

1	Assunzioni	2
2	Starting Point	2
2.1	Approccio Epsilon-Greedy	2
2.1.1	Ritorno dell'OIV	2
3	Cosa fare? Migliorare l'Esistente	3
3.1	Da AISG ad AISG_Updated	3
3.2	Problemi..?	4
4	Primi Approcci al Q-Learning	4
4.1	Che decisioni prendere? → 2 Actions	4
4.1.1	Torniamo Anche al Depot: 3 Actions	5
4.1.2	Possibile Tentativo: N+1 Actions	5
4.2	Ulteriore Tentativo: Approccio vicino al Depot	5
5	Svolta: 3A_TMGeo	5
5.1	Novità: Grid Algorithm	7
5.1.1	Miglioria: Grid_WGeo	8
6	Sviluppi futuri sulla batteria	8
7	Sezioni Implementate per Componenti	9
8	Conclusioni	9

./Images/logo.png

▣ ./Images/logo_hidden.png

1 Assunzioni

Si assume che sia stato letto e compreso il primo report.

Tutti gli approcci non soddisfacenti, nel primo Homework, non sono stati considerati visto che il secondo Homework è un'estensione del primo: tipo l'algoritmo NONE, UCB e via dicendo non sono stati riconsiderati perchè ovviamente continuerebbero a mantenere pessime performance, naturalmente

2 Starting Point

Consideriamo il miglior algoritmo utilizzato in precedenza: l'AISG.

Il nostro algoritmo competitor è l'MGEO: esso lavora come il Geographical Algorithm, solo che nel caso non ci siano dei nodi vicini per un determinato drone, allora si ha l'azione di tornare al depot

Tale azione è molto energy-consuming, anche se porta grandi risultati.

2.1 Approccio Epsilon-Greedy

Tutti gli algoritmi che vedremo utilizzano l'approccio Epsilon-Greedy, visto che in precedenza ha portato ottimi risultati.

In questo caso si ha che con probabilità:

- $1-\epsilon$ si sfrutta il Q-Learning,
- ϵ si sfrutta l'MGEO modificato per aggiornare i valori del Q-Learning¹.

2.1.1 Ritorno dell'OIV


Precedentemente l'OIV² aveva pessimi risultati, ma lo si è risperimentato perchè in questo caso il problema è leggermente diverso vista la nuova azione introdotta.

Sperimentalmente un OIV = 10 comporta una successiva migliore decisione sub-ottimale. Infatti si ha un'esplorazione iniziale maggiore³ e quindi il learning per il K-Armed Bandit/Q-Learning è sensibilmente migliore. In

¹Non si sfrutta l'approccio randomico visto che in precedenza è stato scadente

²Optimistic Initial Value

³ovviamente la funzione di Reward è la stessa del primo Homework, quindi il valore massimo che si può prendere è pari a 2



./Images/logo.png

▣ ./Images/logo_hidden.png

precedenza non avveniva ciò perchè non vi era anche l'azione del ritorno al depot, quindi non c'erano grandi influenze per ciò.

3 Cosa fare? Migliorare l'Esistente


3.1 Da AISG ad AISG_Updated

L'AISG è un approccio che non tiene in considerazione il rientro al depot, quindi si è realizzato un AISG leggermente migliore: quando non ci sono nodi vicini allora il nodo stesso torna al depot.


Solo con questa modifica si ha:

1. Un Battery-Consuming maggiore → Infatti frequentemente si torna al depot e tale azione consuma molto;
2. Uno Score abbastanza concorrente ad MGEO → Questo ci fa ben sperare visto che si sta sfruttando un semplice approccio K-Armed Bandit e non Q-Learning

Con tutto ciò si ha un ottimo punto di partenza per i successivi algoritmi.



./Images/First_Plot_Score.png



./Images/First_Plot_Energy.png

Figura 1: AISG vs AISG_UPDATED vs MGEO vs GEO

./Images/logo.png

▣ ./Images/logo_hidden.png

Si è inserito anche il Geographical Routing per capire il suo livello di prestazioni → Come l'AISG anche il GEO non considera l'opzione di tornare al depot e questo lo esclude da molte possibili scelte → Perciò ha un andamento chiaramente inferiore rispetto all'MGEO ed all'AISG_Updated

3.2 Problemi..?

Purtroppo anche l'AISG Updated è lontano dall'MGEO, soprattutto a livello di score → Ciò è spiegabile perchè con il K-Armed Bandit⁴ si hanno azioni ed un solo stato per il sistema, quindi non si sa cosa è accaduto in precedenza (nessun concetto di storia) → Ciò crea dei problemi, perchè in tale approccio si ha un learning molto limitato e nel problema in cui i droni quasi mai tornano al depot, allora i problemi possono aumentare esponenzialmente.

Quindi si passa al Q-Learning, che è un'estensione del K-Armed Bandit → Esso sfrutta gli stati e la storia passata ed infatti, nel nostro problema si adatta benissimo portando risultati sensibilmente migliori.

4 Primi Approcci al Q-Learning

4.1 Che decisioni prendere? → 2 Actions

Ora il problema è: Come implementare una base di Q-Learning? Supponiamo di avere μ droni, l'idea più banale è avere μ stati e sole due azioni.

Per ogni stato⁵ si hanno due possibili azioni:

1. Tenere il pacchetto per sè;
2. Passare il pacchetto ad un qualsiasi vicino.

Grande problema: non sfrutta il ritorno al depot → Ciò comporta che il suo consumo energetico è bassissimo, ma tantissimi pacchetti scadono, soprattutto quando si hanno pochi droni.

Ricorda: il nostro obiettivo principale è lo score piuttosto che il consumo energetico → Si vuole che vengano consegnati più pacchetti possibili e per tale scopo tale approccio non è soddisfacente.

⁴Tipologia di Reinforcement Learning utilizzata in AISG Updated

⁵Ovvero drone, come detto



▣./Images/logo_hidden.png

4.1.1 Torniamo Anche al Depot: 3 Actions

Consideriamo un'azione in più: **Tornare al Depot.**

Il consumo energetico aumenta, ma le prestazioni hanno un netto miglioramento \rightarrow Purtroppo si è ancora troppo distanti rispetto alle prestazioni dell'MGEO

4.1.2 Possibile Tentativo: $N+1$ Actions

E se le azioni possibili non fossero solo 3 ma $n+1$, ovvero:

- Tenere il pacchetto;
- Andare al Depot;
- Passarlo ad uno degli $n-1$ vicini⁶.

Problema: non converge mai \rightarrow Non riesce mai ad imparare abbastanza e ciò comporta che a fine simulazione ancora non si è arrivati ad avere delle scelte significative, perchè l'algoritmo sta ancora un pò in fase di esplorazione \rightarrow Ciò spiega le disastrose prestazioni, rispetto agli altri algoritmi.

4.2 Ulteriore Tentativo: Approccio vicino al Depot

Se si è entro un certo range dal depot e si ha almeno un pacchetto, allora si va verso il depot, se non ci sono altre possibilità.

Si rientra al depot se non si consuma eccessivamente \rightarrow Se si è dall'altra parte dello scenario, allora si consuma troppa energia e si evita. In tal caso si aspetta un drone vicino o si mantiene il pacchetto.

Chiaramente questa modifica non può fornire risultati buoni, perchè può succedere che un drone molto lontano dal depot non ha vicini e quindi tutti i pacchetti da lui generati continuano a morire. Lo score è il nostro obiettivo, quindi questo è un tentativo fallito.

5 Svolta: 3A__TMGEO

Le seguenti migliorie generano ottimi risultati. Definiamo ξ il nodo attuale, ζ il suo buffer e η i pacchetti, α un eventuale suo vicino:

⁶Non obbligatoriamente tutti contemporaneamente visibili in un certo istante.



▣ ./Images/logo_hidden.png

- Se in ζ ci sono pacchetti che stanno per morire, allora ξ torna verso il depot, per evitare che η vengano persi e quindi che la Delivery Ratio aumenti sensibilmente (così lo Score);
- Se ξ ha un pacchetto ed α sta andando verso il depot, allora ξ gli passa il pacchetto, perchè arriveranno molto velocemente, rispetto a cercare delle trasmissioni eventuali.
- Se ξ ha un pacchetto ed α è in uno stato di *GoMustBack*, allora ξ gli passa il pacchetto, perchè con molta probabilità arriveranno molto velocemente, rispetto a cercare delle trasmissioni eventuali.
- ξ deve ritornare al depot, esse entra in uno stato di *GoMustBack*⁷. Un drone in uno stato di *GoMustBack* prosegue lungo la propria traiettoria fintanto che si sta avvicinando al depot (esse è possibile osservarlo calcolando l'angolo tra la traiettoria del drone e la traiettoria che dovrebbe percorrere per tornare al depot), in caso contrario torna al depot. Tramite questo approccio si massimizzano gli spostamenti utili del drone, minimizzando la lunghezza dei percorsi da e verso il depot. *GoMustBack* o fare la scelta migliore secondo il Q-learning.

⁷Non è uno stato del sistema, quindi per il Q-Learning, ma uno stato personale del drone

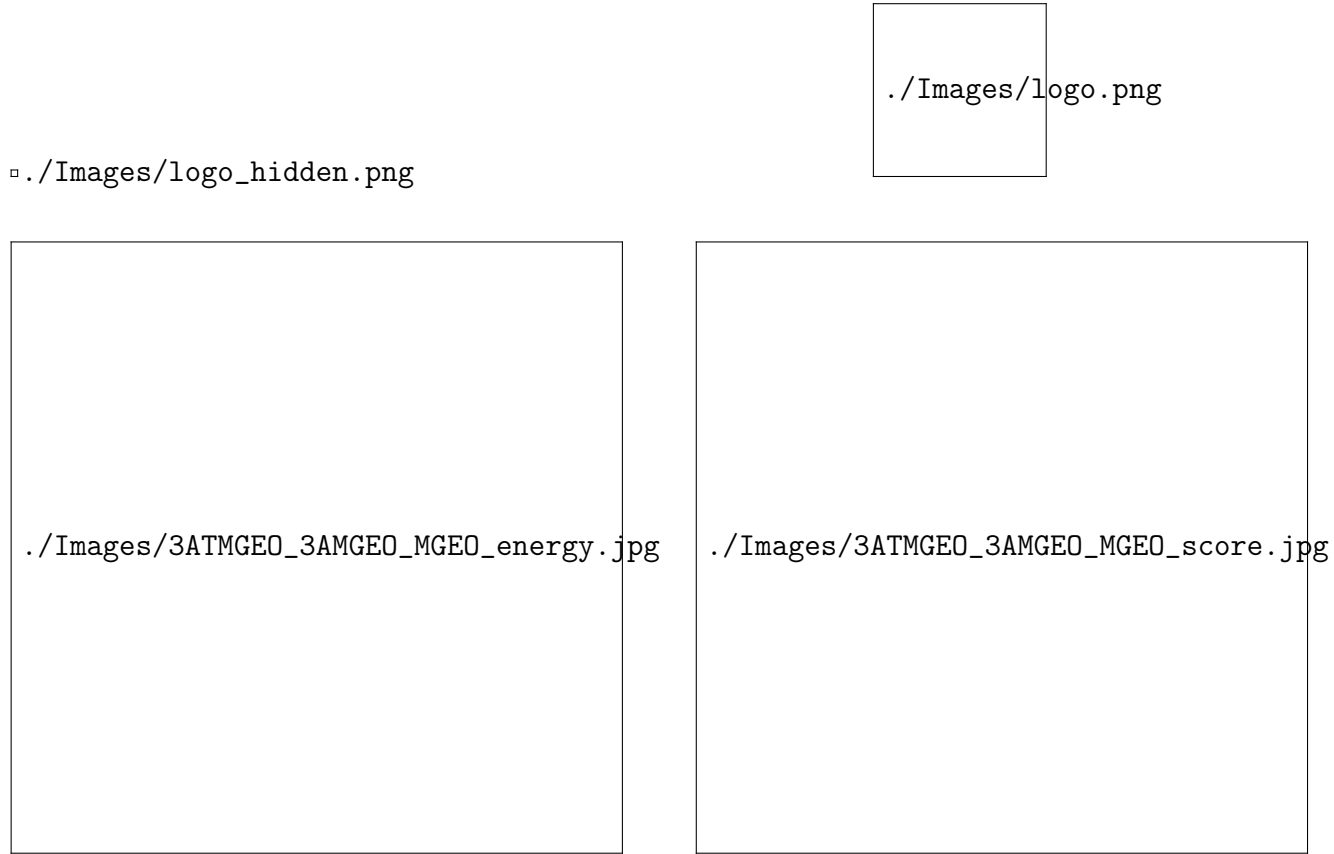


Figura 2: AISG vs AISG_UPDATED vs MGE0 vs GEO

5.1 Novità: Grid Algorithm

Considerare i droni come stati comporta una difficile convergenza, perchè si hanno troppi stati, generalmente. Tale strategia non tiene in considerazione la posizione di un drone ed il passaggio di stato dipende dal passaggio da un drone all'altro per il pacchetto.

Ma non è importante il drone come soggetto, perchè è importante la sua posizione → Se due droni si trovano nella stessa posizione, allora effettivamente essi sono equivalenti.

Nel Grid Algorithm si divide lo scenario in β parti ed ogni parte corrisponde ad uno stato. Tutte le miglorie che funzionano nel 3A_TGEO funzionano anche qui, perchè esse non dipendono dal design usato.

L'unica azione diversa è il *passaggio ai droni vicini*: non si considerano i droni, quindi si passa il pacchetto ai droni nella cella adiacente (si prende il miglior drone di quella cella)

./Images/logo.png

▣ ./Images/logo_hidden.png

5.1.1 Miglioria: Grid_WGEO

Il Grid Algorithm si utilizza come chiave per il Q-SET l'indice della cella per lo stato, ma tale approccio è debole perchè non considera la direzione dei droni.

Il Grid_WGEO sfrutta sia l'informazione della propria cella e sia l'informazione della cella che verrà attraversata dalla sua traiettoria. In questo modo si hanno delle miglirie, perchè si considera anche la direzione di dove si va, non si considera però il next_target per evitare una mancata convergenza (si è provato anche tale approccio ma è stato fallimentare per questo problema), quindi si ha una convergenza abbastanza veloce poiché ogni cella ha dalle 2 alle 4 celle confinanti.

PLOT DI GRID_WGEO VS MGEO VS 3A_TMGeo

Analizziamo i risultati step-by-step:

- A livello di score è palese che l'andamento è estremamente vicino all'MGEO, quando non si hanno tanti droni → l'MGEO rientra spesso al depot, mentre GRID_WGEO e 3A_TMGeo sfruttano il Q-Learning; Considerare come stati le celle e non i droni comporta un sensibile miglioramento → Ciò perchè due droni che si trovano nella stessa posizione sono equivalenti → Si hanno risultati migliori perchè si riesce a convergere e quindi si riesce ad imparare molto più velocemente e precisamente;
- A livello di energia si hanno risultati migliori del GRID_WGEO rispetto al 3A_TMGeo, visto il diverso concetto di *stato* e perchè si impara più velocemente considerando meno frequentemente l'opzione di ritorno al depot.

Ma è ancora decisamente lineare con la crescita del numero dei droni rispetto all'MGEO, perchè si torna frequentemente al depot anche se ci sono molti vicini, solo perchè il Q-Learning dice che è l'azione ottimale, mentre è sub-ottimale.

6 Sviluppi futuri sulla batteria

Nel corso di questa ricerca si è svolta anche un'indagine su come ridurre al minimo il consumo della batteria senza perdita di score, o addirittura migliorandolo. Si è scelto di ridurre il numero di ritorni facendo convogliare i



▣ ./Images/logo_hidden.png

pacchetti su un unico drone quando due o più droni aventi un pacchetto si incontrano. Per fare ciò sono stati utilizzati due diversi approcci: facendo convogliare i pacchetti sul drone più vicino al depot o scegliendo il drone ai valori del proprio dizionario Q . Questi approcci hanno dato risultati interessanti, abbassando notevolmente il consumo di batteria ottenendo però uno score simile agli algoritmi originali. Sarebbe interessante approfondire in futuro questa ricerca.

7 Sezioni Implementate per Componenti

- **Giordano:** E' il principale responsabile della costruzione e realizzazione dei seguenti algoritmi:

- AISG_Updated
- 3A_MGEO
- 2_Actions
- N+1 Actions
- Grid Algorithm.

Ha inoltre provveduto in maniera principale alla stesura e realizzazione del report.

- **Mattia:** Ha collaborato nelle attività di ricerca sugli sviluppi futuri.
- **Michele:** Ha ideato e realizzato i seguenti algoritmi:

- $GRID_{WN}EXT_U P$
- TMGEO

Ha collaborato nella realizzazione degli altri algoritmi e del report, realizzato i plot e svolto le attività di ricerca sugli sviluppi futuri

8 Conclusioni

Per questo Homework il miglior algoritmo realizzato è il $GRID_WGEO$ per tutto ciò che è stato già largamente spiegato.



▣ ./Images/logo_hidden.png

Di seguito un riassunto di tutti gli algoritmi proposti con le varie performance (a livello di score ed energetico)

PLOT TOTALE