reinforcement learning ашиглан SNAKe тоглох нь

Гантулга Гантөмөр, Луубаатар Бадарч

1Шинжлэх Ухаан Технологийн Их сургууль

Мэдээлэл, холбооны технологийн сургууль

limited.tulgaa@gmail.com, luubaatar@must.edu.mn

Хураангуй

Энэ ажлын хүрээнд олны танил Могой-Snake тоглоомыг хэд хэдэн аргаар хэрэгжүүлж, тэдгээрийн үр дүнг харьцуулах зорилго тавьсан. Үүнд:

- Энгийн буюу өөрсдийн зохиосон алгоритм;

- Reinforcement-Learning-ийн MDP арга;

- Reinforcement-Learning-ийн Q-Learning арга;

- Reinforcement-Learning-ийн Deep Q-Network арга;

Эхний 2 арга нь урьдчилон тодорхойлж өгсөн нөхцлийн дагуу гүйцэтгэдэг аргууд бол сүүлийн 2 арга нь тоглолтын талбай, төлөвийг өөрөө оролдлого хийж, алдаа гаргах замаар үнэлэн автоматаар тоглож сурдаг аргууд юм. Туршилтын үр дүнд Q-Learning болон Deep Q-Network аргууд нь өрсөлдөхүйц үр дүн үзүүлж байсан. Сургалтын төлөвийн элемэнтүүдийг нарийн тодорхойлж, суралцах хугацааг уртасгах юм бол агент буюу могойны тоглолтын чадвар улам өсөн нэмэгдэх боломжтой нь харагдаж байна.

Түлхүүр үг: *Q-Learning, Reinforcement Learning, Markov Decision Process, Deep Q-Network.*

# Удиртгал

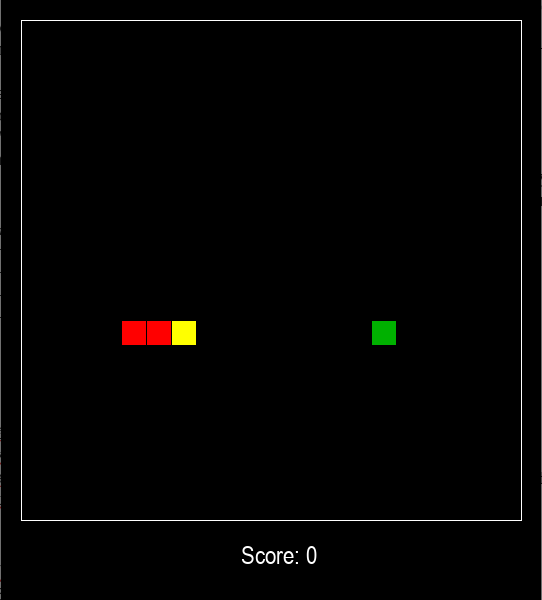
Орчин үед технологи хөгжихийн хэрээр автоматжуулалт, робот систем, хиймэл оюуны салбаруудад томоохон дэвшлүүд гарч байна. Хүчирхэг тооцоолох чадамж өндөр компьютруудын ачаар сүүлийн жилүүдэд өндөр үр дүн үзүүлж байгаа салбар бол Reinforcement Learning [1] юм. Амжилттай хэрэгжсэн жишээнүүд нь DeepMind–AlpaGo, StartCraftII [2], OpenAI–Dota2 OpenAI Five [3] зэрэг юм. Энэхүү ажлын хүрээнд бид RL ыг програмчлалд хэрхэн энгийн байдлаар буулган ашиглаж байгааг харуулахыг зорьсон бөгөөд snake тоглоомыг тоглуулж болох хэд хэдэн аргуудыг туршиж өөр өөр аргуудын давуу болон сул талыг харуулахыг зорисон болно. Тоглоомын код болон бүх алгоритмууд Python хэл дээр бичигдсэн бөгөөд Deep Q-Network [4] сургахдаа Tensorflow [5] (Машин сургалтад зориулагдсан сан) Python хэлний open-source сан ашигласан.

## RL-ын үндсэн ойлголтууд

RL-ийн агент нь өөрийн хүрээлэн буй орчинд тодорхой нэг үйлдэл(action-a) хийх ба үр дүнд агентын нөхцөл байдлаас хамааран тодорхой хэмжээний шагнал (reward-r) авна. r ямар байхаас хамааран тухайн нөхцөл байдалд (state-s) ямар үйлдэл хийгээд ямар нөхцөл байдалд(state’-s’) орсноос хамааран дүгнэх замаар агентыг сургадаг. Ийм бүтцээр задалж үзсэнийг MDP (Markov Decision Problem) [6] гэж авч үздэг. RL нь одоогийн нөхцөл байдлаас дараа дараачийн нөхцөлүүдийг зөв үнэлж хугацааны төгсгөлд reward хамгийн их байхаар тухайн орчинд үйлдэл хийх чадамжтай агентыг сургах зорилготой ба үүнийг үнэлэх тооцоолох арга нь MDP юм. RL-ыг хэрэгжүүлдэг олон аргууд байдаг ба ерөнхийдөө On-Policy [7] ба Off-Policy [8] гэж хоёр ангилдаг. Бидний хэрэгжүүлсэн арга нь “Off-Policy” төрлийн арга бөгөөд орчны тухай мэдээлэлгүйгээр суралцах боломжийг олгодог арга юм.

## Snake

Snake тоглоомын дүрэм нь энгийн бөгөөд могой нь биеэ мөргөх юм уу тоглоомын талбараас хэтэрсэн тохиолдолд тоглоом дууссан гэж үзэх бөгөөд RL-ын хувьд энэ нь "Terminal state" буюу эцсийн цэг болох юм. Могой нь зөвхөн "up", "down", "right", "left" гэх дөрвөн үйлдэл хийх ба агент нь эдгээр үйлдлүүдэс сонгон явна. Могой нь буцаж хөдлөх боломжтой бөгөөд энэ тохиолдолд биеэ мөргөсөнд тооцно.



1. Snake тоглоомын талбар

Snake тоглоомоос state болгон ашиглаж болох мэдээллүүд нь:

* Тоглоомын талбар
* Хоолны байршил
* Могойны биеийн байршил
* Талбарт харагдаж байгаа зүйлстэй хамааралтай бүх мэдээлэл

Бүх state-ыг

Хийж болох үйлдлүүд

; ; ;

Тоглоомын явцад ямар нэг алгоритм ашиглан хийх үйлдлээ сонгох ба алгоритмыг тоглоомын state-ээс хамаарсан функ гэж үзээд гэж үзье.

Snake game Basic game flow

Initialize Snake game

**while** game running

Get game state from

**while** game is not over

Get action using

Execute action in game

Get agent’s observation

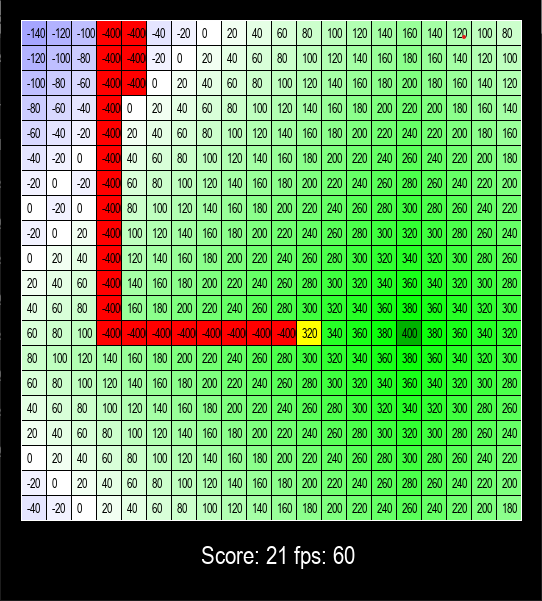
Assign to s

**end while**

**end while**

## MDP (Value Iteration)

Markov Decision Problem (MDP) нь тухайн орчны state хоорондын хамаарлыг орчны динамикаас хамааруулан тооцоолох арга юм. Value Iteration [9] нь MDP-г тооцоолох арга ба Offline Learning алгоритм гэж үздэг. Энэ нь тоглоомын мэдэглэж буй утгууд дээр тооцоолол хийн дүгнэдэг буюу тоглоомыг тоглох шаардлагагүйгээр суралцах бус тооцоолох замаар үнэлдэг арга юм. Snake тоглоомын хувьд тоглоомын талбарын нүд бүрийг state () гэж үзээд Value Iteration ашиглан state бүрийн утгыг хоолны байршлаас хамааран тооцоолох юм.



**Зураг 2. MDP planning with Value Iteration**

Value Iteration ашиглан MDP тооцоолоход

(1)

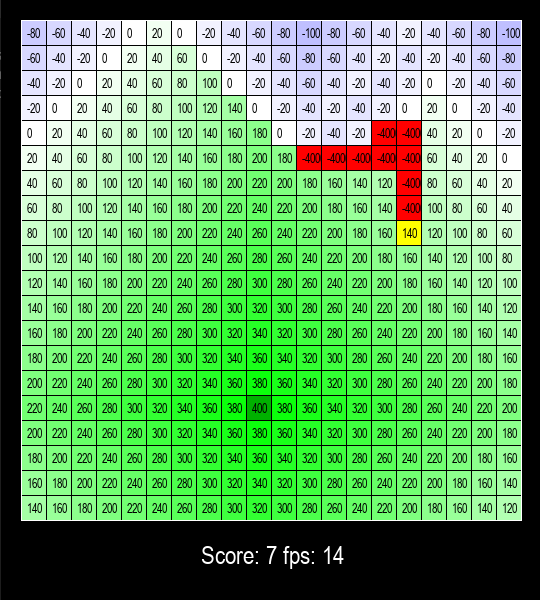
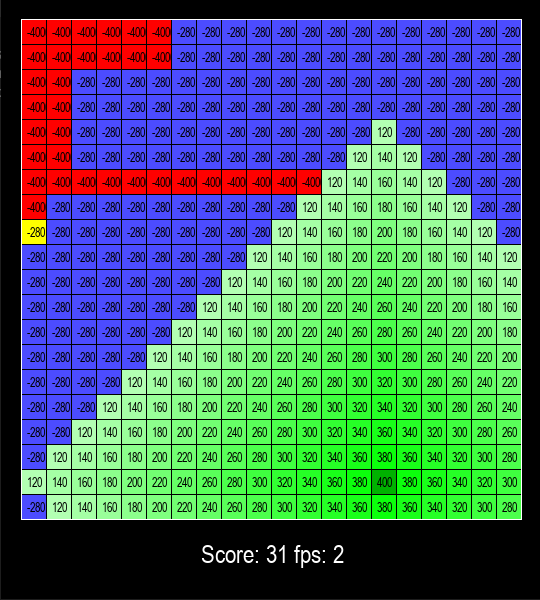
(1) томъёог ашиглах ба γ, , зэрэг утгууд мэдэгдэж байх ёстой. Snake-ын хувьд орчны динамик бүх state-д ижил байх ба агент зорьсон state дээ заавал буудаг буюу тухайн сонгогдсон үйлдэл бүрийн магадлал тогтмол 1 гэж үзэж болно. R буюу reward функцын утгуудыг

(2)

дүрмээр тогтоож өгсөн. Тооцооллыг хялбар байлгах үүднээс γ параметрийг 1 гэж үзсэн болно. (1) томъёог дээрхи нөхцлүүдийг харгалзан хялбарчилвал

(3)

(3) томъёогоор тоглоомыг дүрслэх бүрт state болгон дээр бодогдох ба нэг frame дүрслэхэд нэг ба түүнээс дээш удаа бодож болно.



**Зураг 3. 1 frame-д 1 удаа (зүүн гар тал), 20 удаа тооцоолох (баруун гар тал)**

**( баруун)**

Агент нь алхам бүртээ могойны толгойг тойрсон дөрвөн нүдний хамгийн их утгатай зүгт хөдлөх ба зөвшөөрөгдсөн нүдний утгууд ижил байвал -ын боломжит хамгийн бага индекс дээрхи үйлдлийг сонгож явах ба сул үйлдэл хийсэнд тооцно. Тоглоомын талбарыг бүрэн бодож дуусах хугацаа бага байх тусам могой нь сул үйлдэл хийх нь багасна. Зураг 3-т харуулж буйгаар агент маань үйл тооцооллоо хийх ба шууд гоалруу чигэлсэн үйлдэл хийх нь урт хугацаанд үр дүнгүй болох тоглуулах явцад харагдсан. Өөрөөр хэлбэл хэт greedy үйлдэл хийх нь Snake тоглоомыг тоглох арга биш гэдгийг харуулж байна.

Snake game with Value Iteration

Initialize Snake game

Initialize state value table by zeros

Set value table update rate

Now our becomes

**while** game running

Reinitialize state value table by zeros

Get game state

**while** game is not over

Choose action using

Execute action in game

Get agent’s observation

Update by update rate

Assign to s

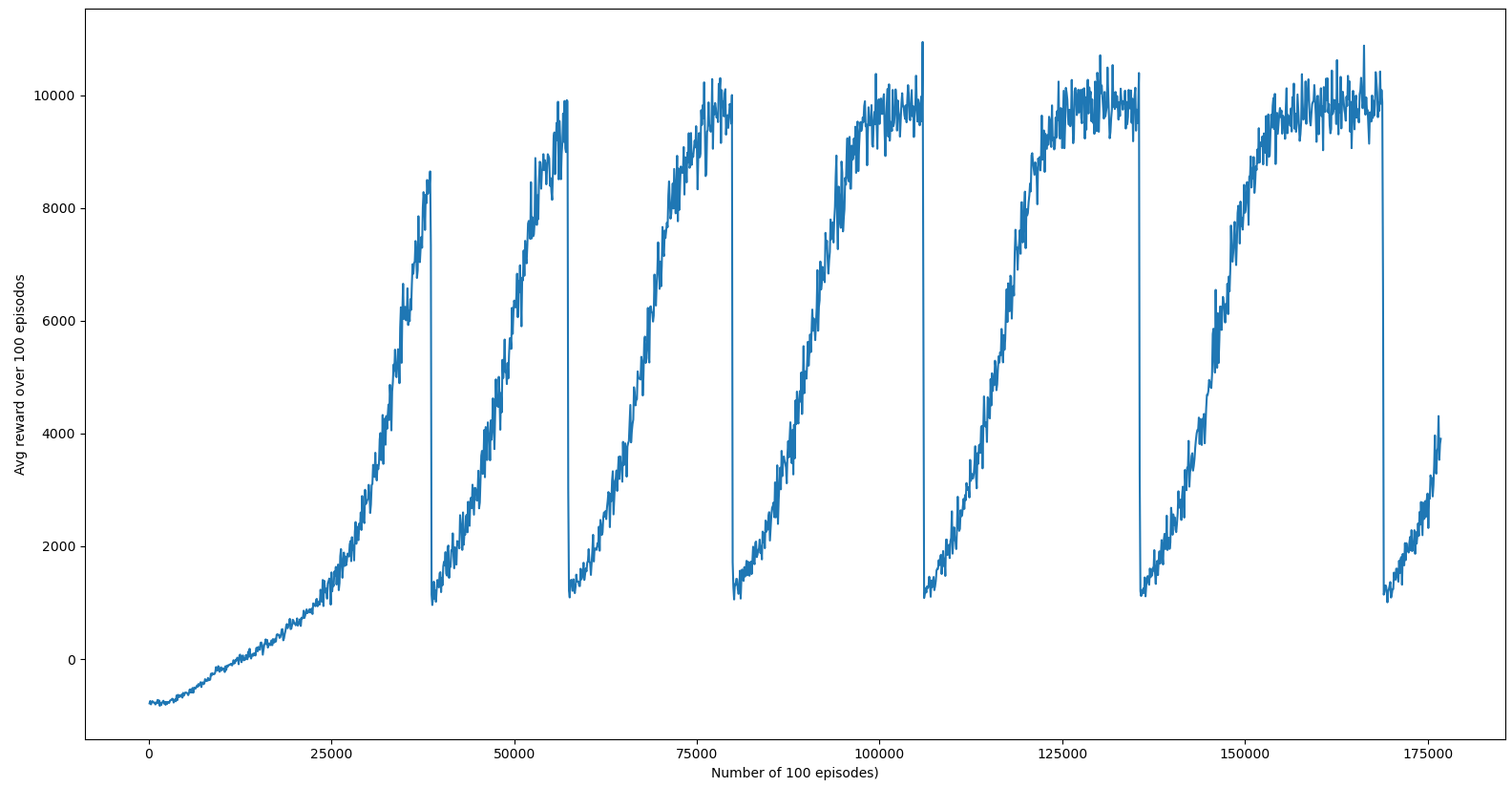
**end while**

**end while**

## Q-Learning

Greedy үйлдэл хийх нь хамгийн сайн арга биш гэдгийг бид MDP ашиглаж харууллаа. Иймээс нэг шат ахин RL ашиглан Online Learning буюу тоглоомыг тоглуулах замаар агентыг сургасан. Q-Learning нь орчны state үүдээс sample хийх замаар буюу тухайн орчныхоо state үүдээс хамааран үйлдэл хийх замаар state бүрийн утгын тооцоолдог арга юм. Value Iteration хэрэгжүүлж байхад тоглоомын талбарын нүд бүр state болж байсан ба reward-ын тархалтаас агентын хийх үйлдлийг тогтоож байсан бол Q-Learning аргаар сургахдаа агентын state нь хоол байрлах зүг могойны толгойг тойрсон 4 нүдны комбинаацууд байхаар тодорхойлсон.

Q-Learning нь state бүрт харгалзах үйлдлүүдийг үнэлсэн хүснэгт байх ба тэдгээр үнэлгээнүүдийг агентын авж байгаа reward аас хамааруулан өөрчилж тааруулахан хугацааны төгсгөлд хамгийн их reward авах арга барилд суралцахад оршдог.

 Q-Learning нь тоглоомыг бүрэн үнэлэхийн тулд бүх state үүдийг нэг буюу түүнээс олон удаа тооцоондоо оруулсан байх шаардлагатай байдаг. Хэрвээ агент ямар нэг байдлаар тодорхой хэмжээний эерэг reward өгдөг арга олбол тоглоом нэг л аргаар явж сурсан байх боломжтой. Өөрөөр хэлбэл агентыг үйлдэл бүртээ сайжирдаг байлгахын тулд өөр өөр аргуудыг туршиж үздэг байх буву рандомчлал хэрэгтэй болно. Үүнийг RL-д Exploration and Exploitation [10] гэж авч үздэг ба параметрээр рандомчлал болон сурсан мэдлэг хоёрыг тэнцвэржүүлдэг epsilon greedy гэж нэрлэдэг. Агентыг сургахдаа

**Зураг 4. Нийт сургалтын 100 үе бүрийн дундаж reward**

(4)

томъёог ашиглах ба параметрүүдийг:

(5)

байдлаар тааруулж өгсөн.

Snake game with Q-Learning

Initialize Snake game

Initialize Q table by zeros

Now our becomes

**while** game running

Get game state

**while** game is not over

Choose action using

Execute action in game

Get agent’s observation

Update by equation 4

Assign to s

**end while**

**end while**

Сургалтыг хэд хэдэн янзаар хэрэгжүүлж үзсэн ба хамгийн сайн сурсан агент нь

Зураг 4. суралцсан ба MDP ээс ялгаатай байдлаар суралцаж greedy бус үйлдлүүдийг хийж тодорхой арга барилын дагуу тоглож байгаа нь харагдсан. Гэвч Snake тоглоомыг төгс тоголдог байхын тулд дөрөвхөн нүдний мэдээлэл хангалтгүй болох нь харагдсан.

## Deep Q-Network

Энэ арга нь Q-Learning-ын функцыг Neural Network болгон сольсон арга бөгөөд давуу тал нь Q функцын оролт тодорхой хязгаарт хязгаарлагдахгүй, агентын харж байгаагүй state д ойролцоо state үүдээс хамааруулан ойролцоолчлох боломжтой. Мөн Network архитектураас хамааран агент өөр өөр байдлаас суралцах боломжтой. DQN-ыг хэрэгжүүлэхэд гарах нэг ялгаа нь q-table-ыг 0 оор дүүргэж байсан бол Network дээр бүх weight үүдийг 0 оор өгч болдоггүй учир санамсаргүй тоонуудыг олгодог. Мөн Network рандом учир Network-ын алдааг бодох нь асуудалтай болно. Network-ын гаралт мөн тодорхой бус байна. Тиймээс Network-ын таргет утгыг боддог байх тогтвортой Network байх хэрэгтэй ба үүний тулд үндсэн Network-ыг хэсэг хугацаанд өөрчлөлтгүй байлгах хэрэгтэй ба үндсэн Network-ын weight үүдийг ижил архитектуртай Network-д хуулж үлдээдэг [4]. Ингэснээр үндсэн Network-ын алдаа тогтвортой болно. Мөн Network-ыг SGD байдлаар сургах нь үр дүн муутай байдаг ба Mini Batch Gradient Descent аргаар сургах нь илүү үр дүнтэй байдаг. MBGD аргаар сургахын тулд бэлэн өгөдлүүд шаардлагатай бөгөөд Replay Buffer [4] ашиглаж тоглоомын тухайн агшиндах ын утгуудыг хадгалж авна.

DQN аргаар сургахад

(6)

(7)

(8)

(6), (7), (8) томъёоны дагуу явах бөгөөд параметрүүдийг:

(5)

Network архитектур:

TODO

Snake game with DQN

Initialize Snake game

Initialize main and target Q network weights

Now our becomes

**while** game running

Get game state

**while** game is not over

Choose action using

Execute action in game

Get agent’s observation

Append into Replay Buffer

When have enough data to train

Sample data from Replay buffer

update main net by equation 8

After times of training copy main

net to target net

Assign to s

**end while**

**end while**

## Hard coded snake algorithm

TODO

# Дүгнэлт (heading 1)

TODO

# Ном зүй

1. Sutton, Richard S., and Andrew G. Barto. "Reinforcement learning." Journal of Cognitive Neuroscience 11.1 (1999): 126-134.
2. Vinyals, Oriol, et al. "Starcraft ii: A new challenge for reinforcement learning." arXiv preprint arXiv:1708.04782 (2017).
3. Berner, Christopher, et al. "Dota 2 with large scale deep reinforcement learning." arXiv preprint arXiv:1912.06680 (2019).
4. Mnih, Volodymyr, et al. "Playing atari with deep reinforcement learning." arXiv preprint arXiv:1312.5602 (2013).
5. Goldsborough, Peter. "A tour of tensorflow." arXiv preprint arXiv:1610.01178 (2016).
6. Even-Dar, Eyal, Yishay Mansour, and Peter Bartlett. "Learning Rates for Q-learning." Journal of machine learning Research 5.1 (2003).
7. TODO
8. TODO
9. Anahtarcı, Berkay, Can Deha Karıksız, and Naci Saldi. "Value iteration algorithm for mean-field games." Systems & Control Letters 143 (2020): 104744.
10. Coggan, Melanie. "Exploration and exploitation in reinforcement learning." Research supervised by Prof. Doina Precup, CRA-W DMP Project at McGill University (2004).