

Reinforcement Learning in BCC

Fologea Valentin-Alexandru
Sanda Marian-Tiberiu
Ivascu Ioan-Andrei
Grupa 343

Ce este Deep Reinforcement Learning?

- **Deep RL** combină **Reinforcement Learning (RL)** cu **Deep Learning**, permițând agenților să ia decizii complexe într-un mediu dat.
- Agenții interacționează cu mediul, primesc recompense și învață să îmbunătățească deciziile lor pe termen lung.
- **Aplicații comune:** Jocuri (de exemplu AlphaGo, Atari), roboți autonomi, vehicule autonome.

Beneficii:

- Scalabilitate pentru medii complexe.
- Capacitatea de a învăța direct din date brute (ex: imagini sau acțiuni).

Jocul nostru: BCC

- **BCC** este un platformer 2D creat în **Godot**, în care jucătorii îl urmează pe **Aurora**, o prințesă pierdută, captivă într-o lume întunecată.
- **Scopul jocului:** Aurora trebuie să înfrunte inamicii și să găsească calea spre libertate, explorând nivele complexe și pline de provocări.
- **Repo GitHub:** [BCC GitHub Repo](#)



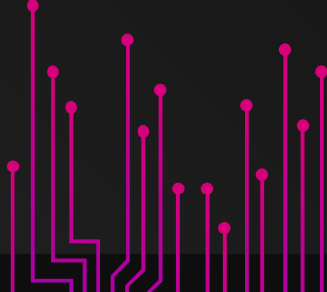


Modele de Deep RL folosite

Stable baselines

- O bibliotecă populară de **Reinforcement Learning** în Python, utilizată pentru implementarea și antrenarea agenților RL.
- Suportă mai mulți algoritmi RL, cum ar fi PPO, DDPG, A2C, SAC.
- Este optimizată pentru stabilitate și performanță.

Clean RL

- Un framework minimalist pentru Reinforcement Learning, destinat cercetării și învățării.
 - Pune accent pe **claritatea codului** și pe reproducibilitate, facilitând analiza experimentelor RL.
 - Ușor de integrat și utilizat pentru prototipuri rapide.
- 

Cum integrăm RL în Godot?

- Folosim **addon-ul rl-agents** din **Godot** pentru integrarea algoritmilor RL.
- Conectăm framework-urile Python (**Stable Baselines** și **Clean RL**) cu Godot printr-o conexiune **TCP**.
- Addon-ul facilitează interacțiunea dintre jocul creat în Godot și agenții antrenați în Python.

Documentație utilă:

- [Stable Baselines pentru Godot RL](#)
- [Clean RL pentru Godot RL](#)

Pași pentru rulare:

1. **Instalarea pachetelor** necesare pentru framework-urile RL (Stable Baselines și Clean RL).
2. Pornirea serverului TCP prin rularea scripturilor Python
3. Rularea scenei curente din **Godot** pentru a începe interacțiunea dintre agent și mediul de joc.

Reward System

0.3 – Găsirea unei noi distanțe minime între caracter și final (minim local).

0.1 – Mișcarea în direcția finalului.

50 – Terminarea cu succes a nivelului, plus un **reward dinamic** bazat pe viteza de finalizare.

5 – Terminarea nivelului cu cel mai scurt timp (minim local).

-1.5 – Penalizare dacă caracterul pierde HP (scalată în funcție de HP curent).

-1 – Penalizare pentru utilizarea abilității **stun** atunci când aceasta este în cooldown.

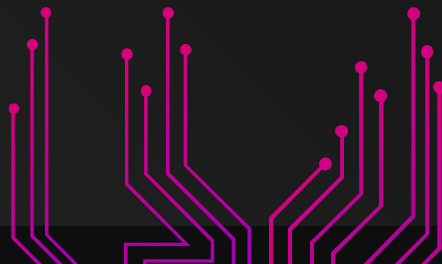
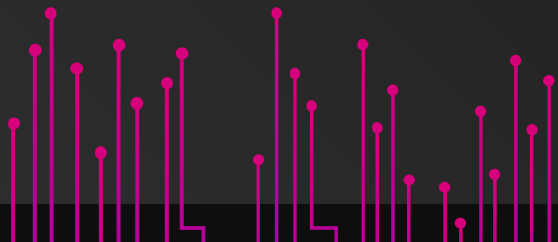
-1 – Dacă nivelul nu este finalizat în timpul propus (10 minute).

0.2 – Lovirea inamicilor (cu atac basic sau stun).

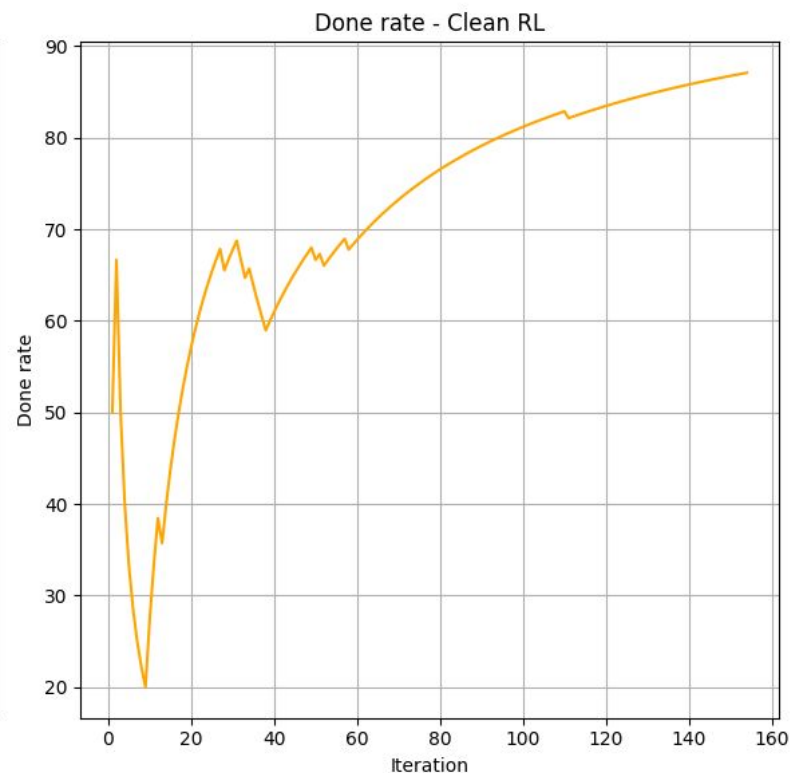
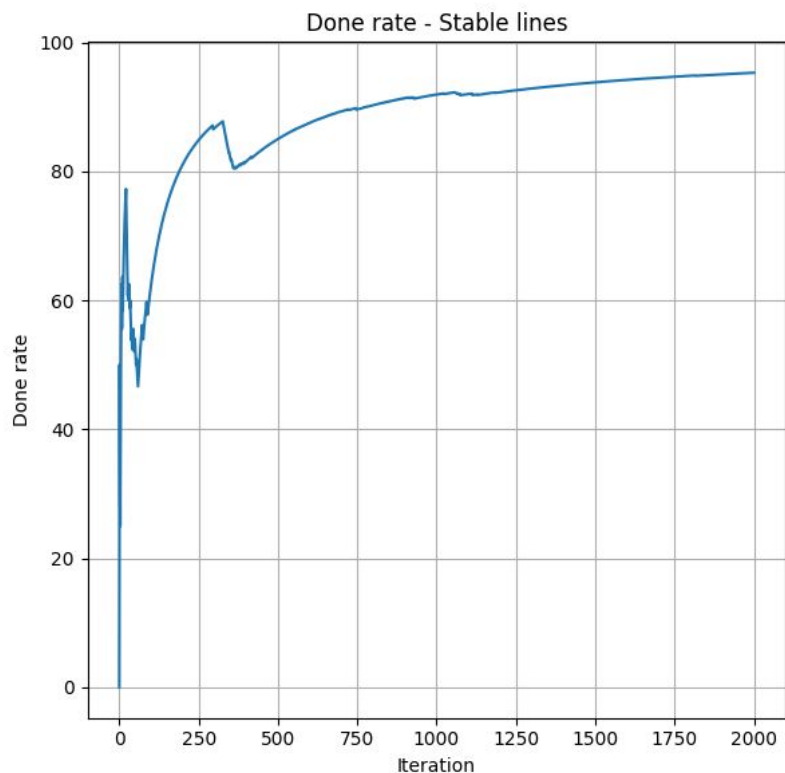
0.6 – Eliminarea unui inamic.

Observații primite de agent

- **Distanța până la finalul nivelului:**
 - Normalizată în intervalul $[0, 1]$.
- **HP-ul curent:**
 - Normalizat în intervalul $[0, 1]$.
- **Datele de la RaySensor:**
 - Acestea sunt normalizate automat de addon-ul Godot RL în intervalul $[-1, 1]$.

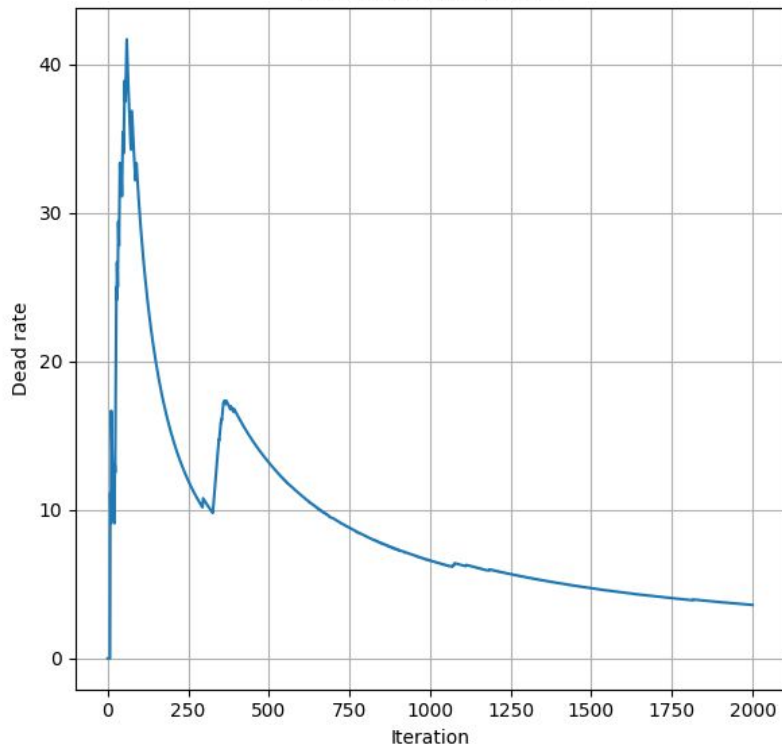


Rata de finalizare a nivelului a crescut semnificativ pentru ambele modele. Stable Baselines atinge aproape 100% după aproximativ 500 de iterații, în timp ce Clean RL ajunge la 90% în aproximativ 150 de iterații, arătând un progres mai rapid la început, dar o convergență ușor mai mică pe termen lung.

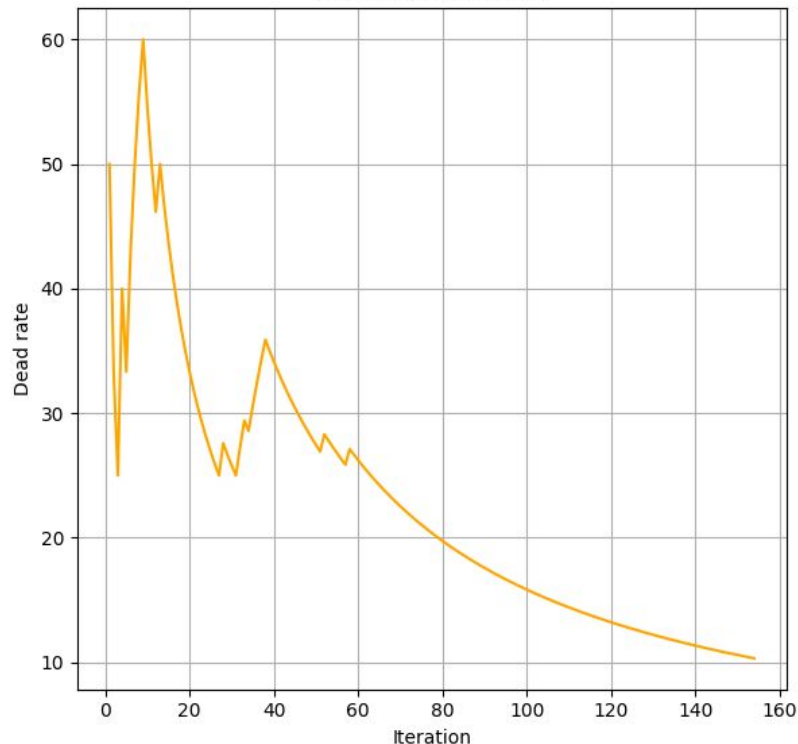


Rata de eşec (moartea personajului) a scăzut substanțial. Stable Baselines ajunge la o rată de sub 5% după 500 de iterări, iar Clean RL o depășește rapid, atingând o performanță similară în aproximativ 100 de iterări, demonstrând un avantaj în evitarea situațiilor critice.

Dead rate - Stable lines

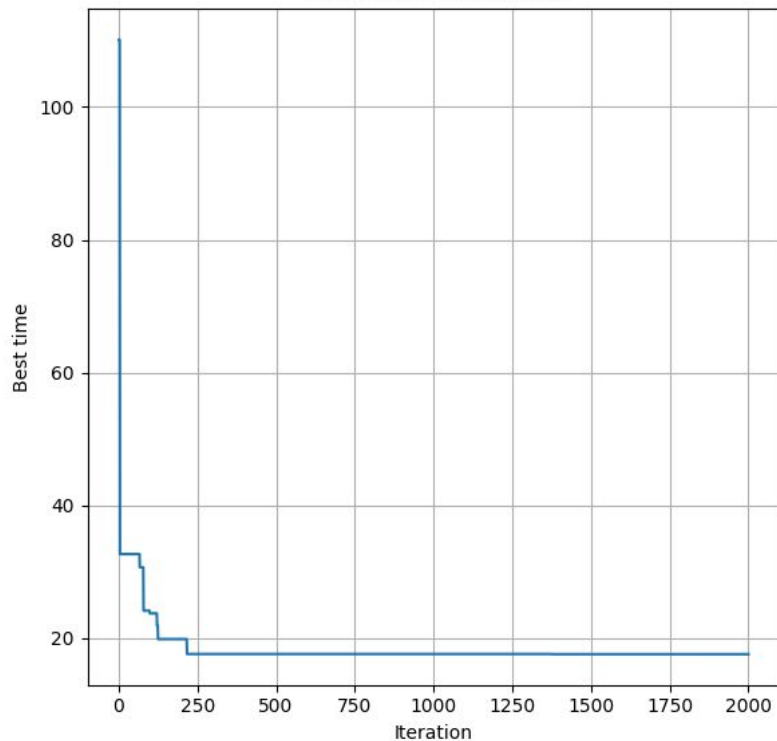


Dead rate - Clean RL

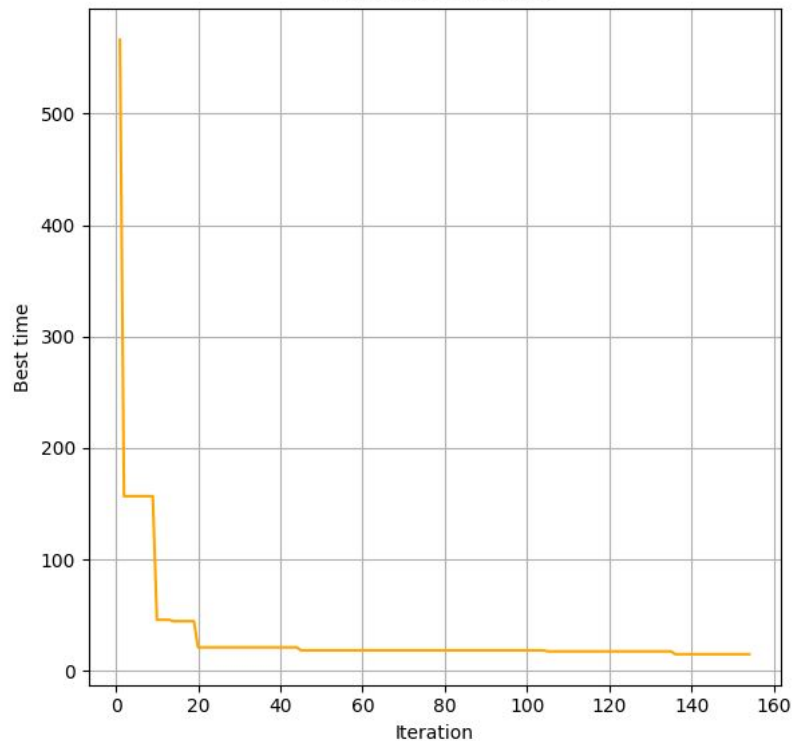


Cel mai bun timp pentru finalizarea nivelului de către un om a fost 23.02 de secunde.
Stable Baselines atinge un timp minim de 17.58 de secunde după 200 de iterații.
Clean RL atinge 15.07 secunde după doar 130 de iterații.

Best time - Stable lines

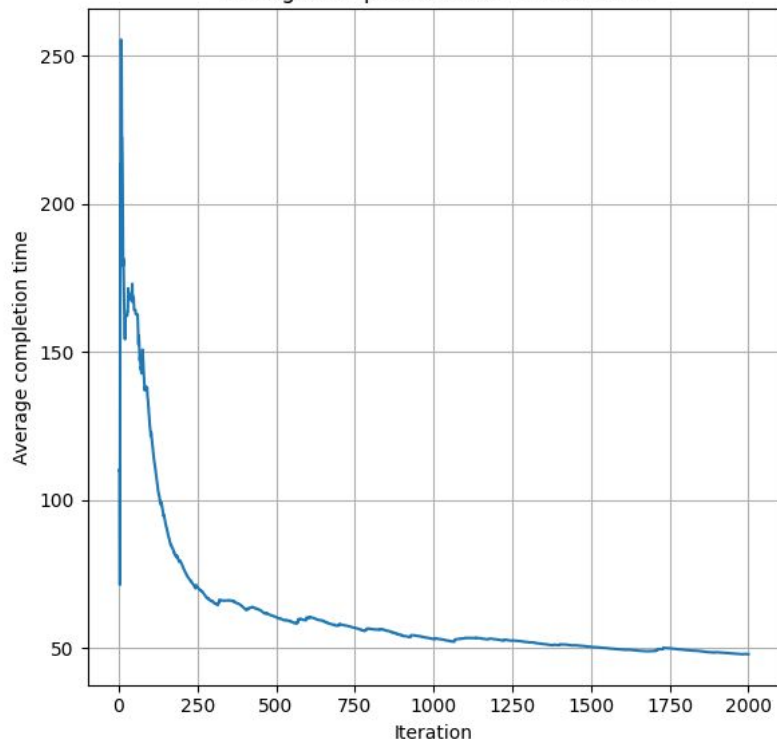


Best time - Clean RL

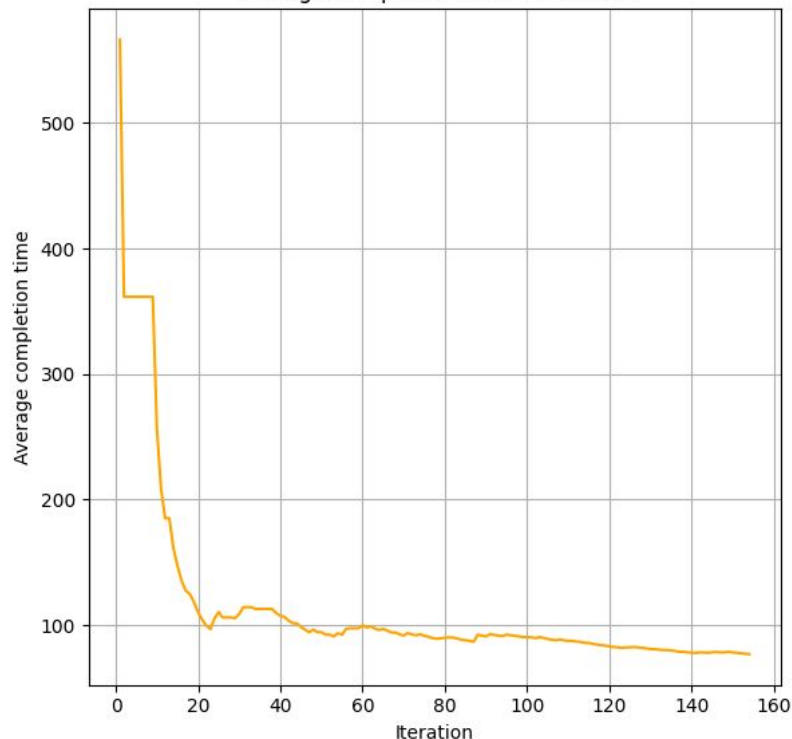


Timpul mediu pentru finalizarea nivelului a fost redus dramatic. Stable Baselines scade la aproximativ 50 de secunde după 500 de iterații, iar Clean RL ajunge la o valoare similară după doar 100 de iterații, indicând o eficiență crescută în antrenarea agenților.

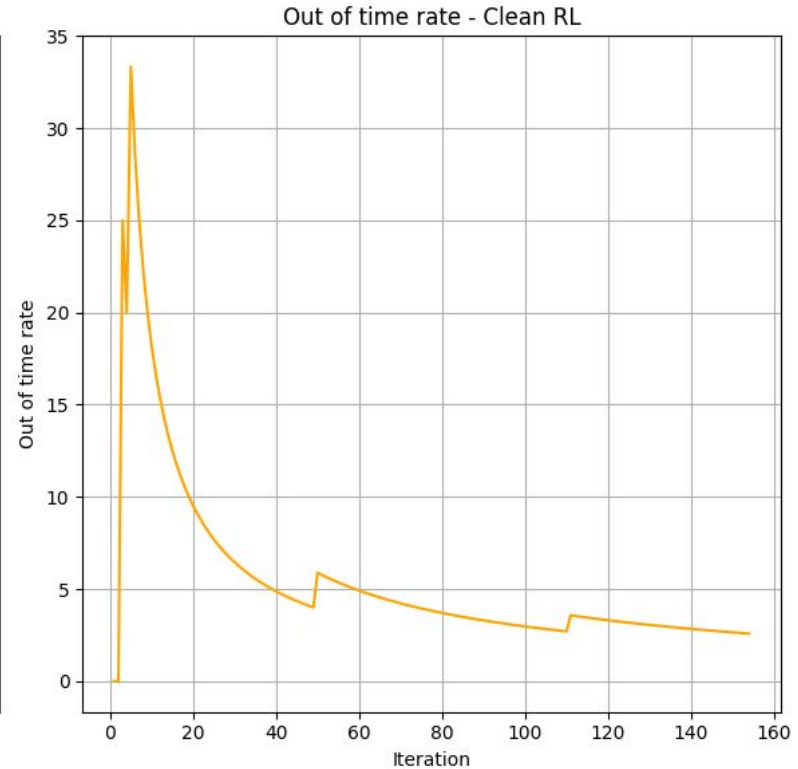
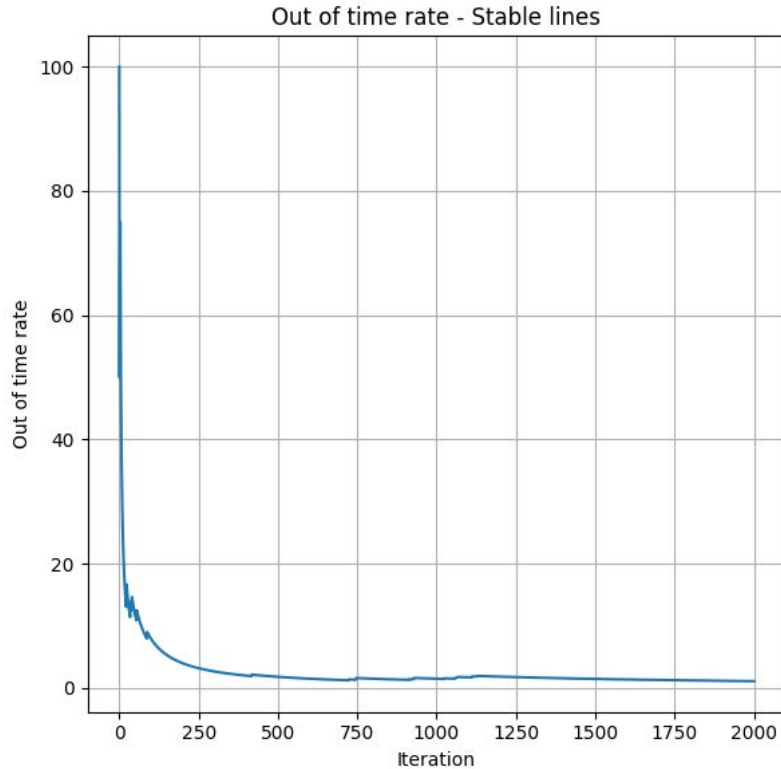
Average completion time - Stable lines



Average completion time - Clean RL



Procentul nivelurilor nefinalizate din cauza expirării timpului a scăzut semnificativ. Stable Baselines ajunge aproape de 0% după 500 de iterări, în timp ce Clean RL atinge aceeași performanță în doar 100 de iterări, demonstrând o adaptare mai rapidă în acest scenariu.



Antrenament



Concluzii

**Timp minim de
completare**

Castigator: CleanRL

**Timp mediu de
completare**

Castigator: Stable Baselines

Rata de convergență

Castigator: CleanRL

Rata de completare

Castigator: Stable Baselines

Robustețe

Castigator: Stable Baselines

The image features a dark gray background with a central white text element. In the four corners, there are abstract, stylized circuit traces in a vibrant magenta color. These traces consist of thin lines that branch out and terminate in small dots, resembling a network or data flow. The top-left and top-right traces are more vertical, while the bottom-left and bottom-right traces are more horizontal, creating a symmetrical, frame-like effect around the central text.

Multumim