

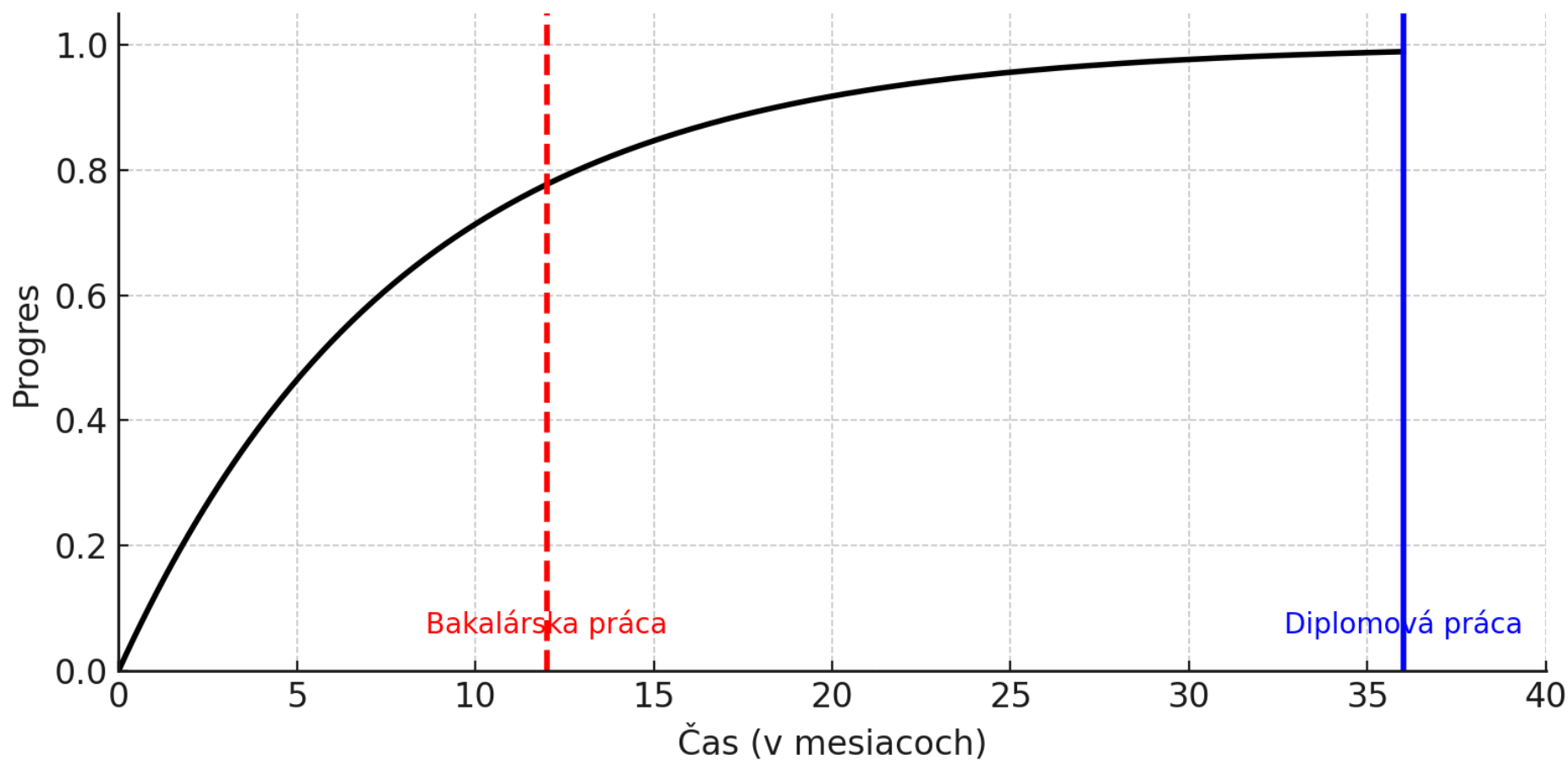
# Autonómne jazdiaci agent pre hru Trackmania

Timotej Melkovič



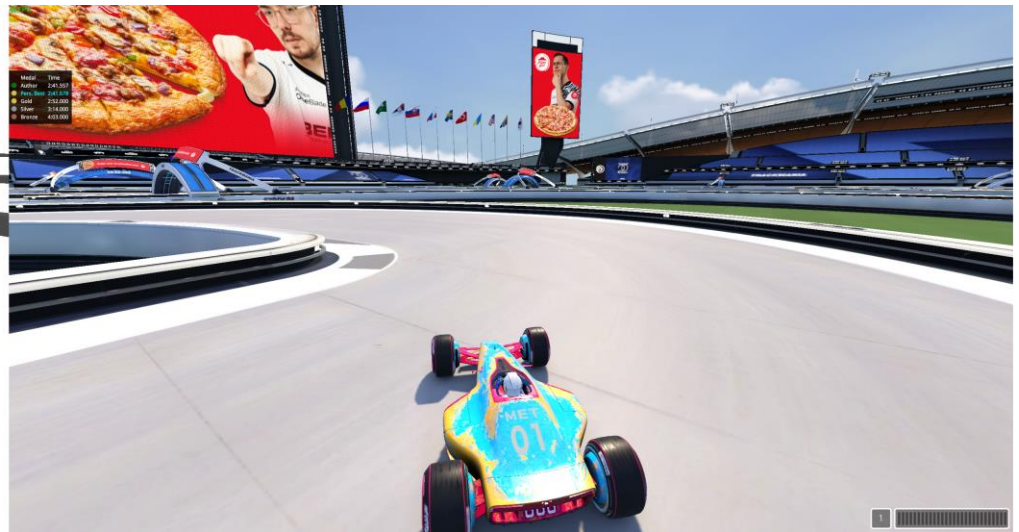
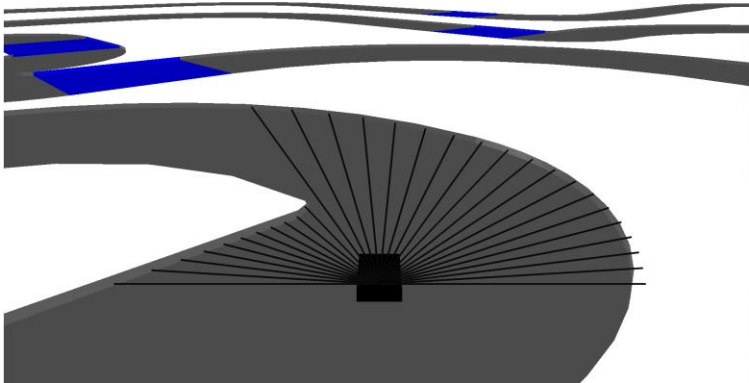
FAKULTA MATEMATIKY,  
FYZIKY A INFORMATIKY  
Univerzita Komenského  
v Bratislave

# Krivka vývoja práce



# Východiská – Bakalárska práca

- Tréning RL agenta v hre Trackmania
- Agent je dopredná neurónová sieť, ktorej výstupom sú akcie agenta
- Agent dokázal dokončiť danú trať
- Vývojové prostredie (framework) funkčný základ pre diplomovú prácu



# Nedostatky z bakalárskej práce

- Tréning neprebiehal plne automaticky
- RL používaný ako black-box bez porozumenia
- Absencia porovnania rôznych RL algoritmov
- Jednoduchá forma vyhodnotenia výsledkov
- Agent jazdil, ale nedostatočne stabilne

# Ciele diplomovej práce

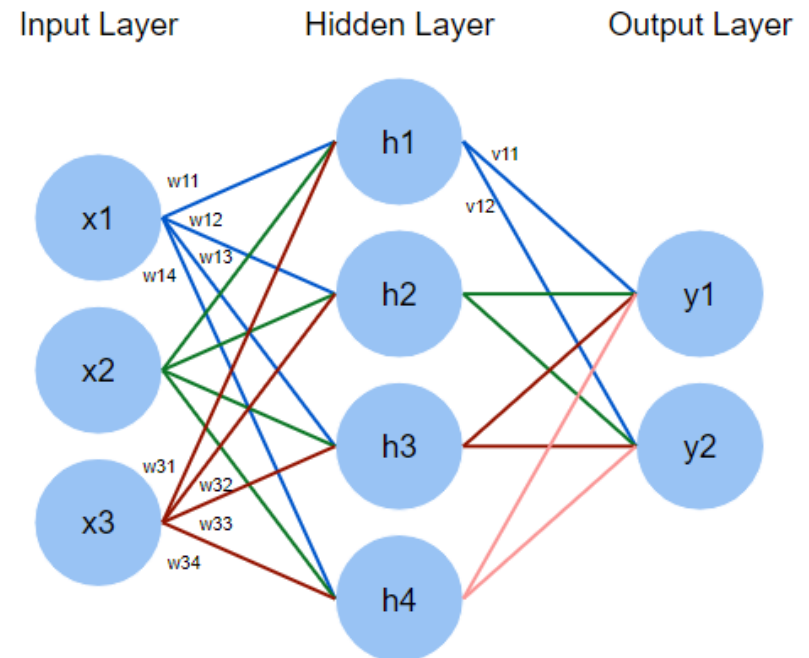
- Preskúmať prístup neuroevolúcie
- Navrhnuť hodnotenie, ktoré nebude založené len na jednej skalárnej reward funkcii
- Hodnotenie by malo rešpektovať viacere kritérií. (Prejdenie trate, rýchlosť, bezpečnosť)
- Automatizovaný tréning s metrikami a vyhodnotením
- Agentovi zadať trať vopred a nespoliehať sa iba na lokálnu informáciu

# Nový pohľad na agenta

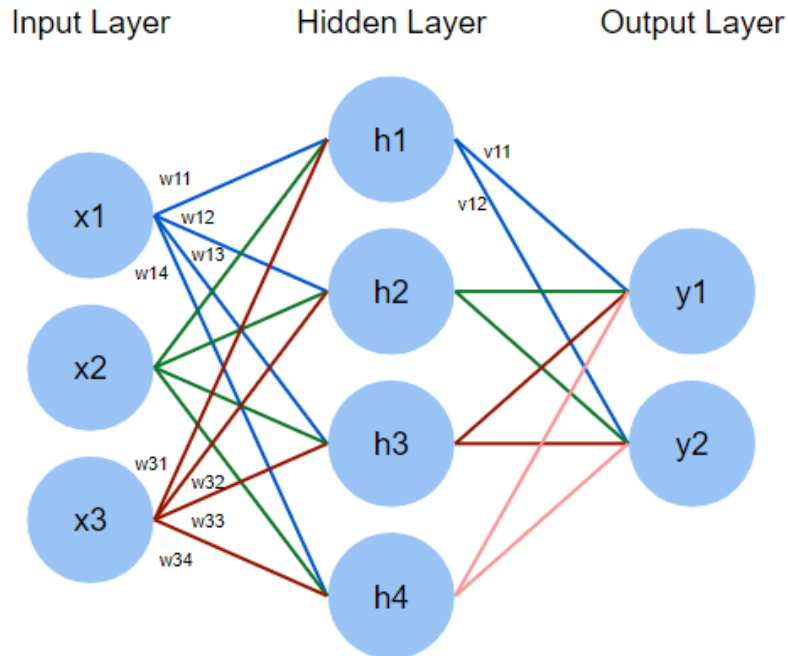
- Agentu stále chápeme ako doprednú neurónovú sieť
  - vstup: senzorické dáta (lidar, rýchlosť, smer...)
  - výstup: riadiace akcie (plyn, brzda, volant)
- Rozdiel je v tom, ako sieť trénujeme:
  - v bakalárskej práci: učenie posilňovaním - po každom kroku sa niečo prepočítava (gradients, aktualizácia politiky)
  - v diplomovej práci: evolučný prístup - počas jazdy pre daného jedinca len dopredné prechody siete
- Epizóda = jazda, po nej len priradíme jedincovi hodnotenie

# Reprezentácia agenta

- Politika agenta = dopredná neurónová sieť s pevnou topológiou
- Jedna skrytá vrstva
- Fixný počet neurónov



# Neurónová sieť ako matice váh



Zdroj: Goodfellow, Bengio, Courville – Deep Learning (2016)

- Potrebujeme matice:
- Váhy vstup  $\rightarrow$  skrytá vrstva:  
 $W_1 \in \mathbb{R}^{3 \times 4}$
- Váhy skrytá vrstva  $\rightarrow$  výstup:  
 $W_2 \in \mathbb{R}^{4 \times 2}$

- Následne výstup neurónovej siete je iba násobenie matíc:

$$H = \sigma(X \cdot W_1 + b_1)$$

$$Y = \sigma(H \cdot W_2 + b_2)$$

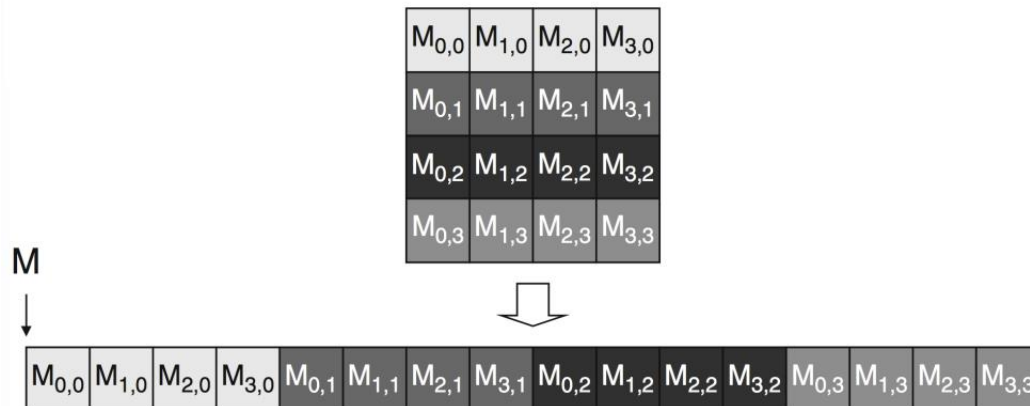
- Kde  $\sigma$  je aktivačná funkcia danej vrstvy
- Kde  $b$  je bias danej vrstvy

Zdroj: <https://datascience.stackexchange.com/questions/75855/what-types-of-matrix-multiplication-are-used-in-machine-learning-when-are-they>

# Matice neurónovej siete ako chromozóm

Zdroj: Evolučné algoritmy – Vladimír Kvasnička

- Všetky váhy a biasy siete spojíme do jedného vektora reálnych čísel
- Tento vektor = chromozóm jedinca
- Každý agent v populácii je jedna konkrétna sada váh a biasov
- Pri evaluácii:
  - Z chromozómu poskladáme sieť
  - Odohráme epizódu, každý krok počítame output pomocou doprednej neurónovej siete
  - Ohodnotíme jedinca podľa toho, ako si počíнал



# Genetický algoritmus

- V každej generácii máme populáciu napr. 32 jedincov
  - Prebieha:
    - Vyhodnotenie - každý jedinec odjazdí trať, zmeráme jeho metriky
    - Selekcia - vyberieme lepších jedincov (turnaj, elitizmus)
    - Kríženie – kombinujeme časti ich chromozómov
    - Mutácia – malé náhodné zmeny váh (gaussovský šum)
    - Vznikne nová populácia - opakujeme
- Priebežne si pamätáme globálne najlepšieho jedinca naprieč generáciami

# Problém jednej skalárnej reward funkcie

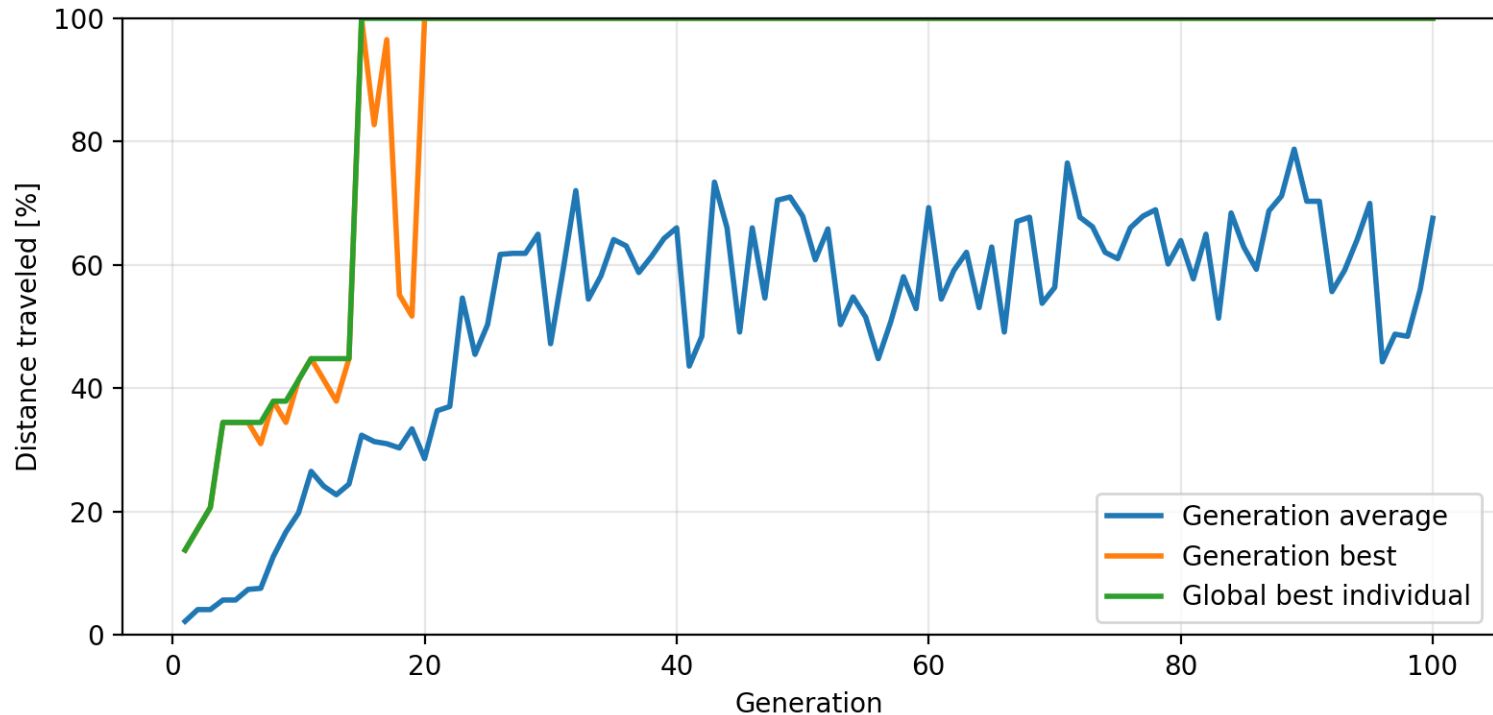
- V bakalárskej práci sme agentovo hodnotenie zlepili do jedného čísla: čas, prejdená vzdialenosť, rýchlosť, kolízie, jazda pri stene...
- Výsledok:
  - ťažko sa ladiace „voodoo“ konštanty
  - agent si našiel rôzne triky (napr. hojdanie od steny k stene), ktoré boli pre reward výhodné, ale pre reálnu jazdu nežiadúce
  - Pri vodičovi máme prirodzene viac cieľov: chce byť rýchly, ale zároveň bezpečný a plynulý

# Multikriteriálne hodnotenie jazdy

Zdroj: Constructing Complex NPC Behavior via Multi-Objective Neuroevolution

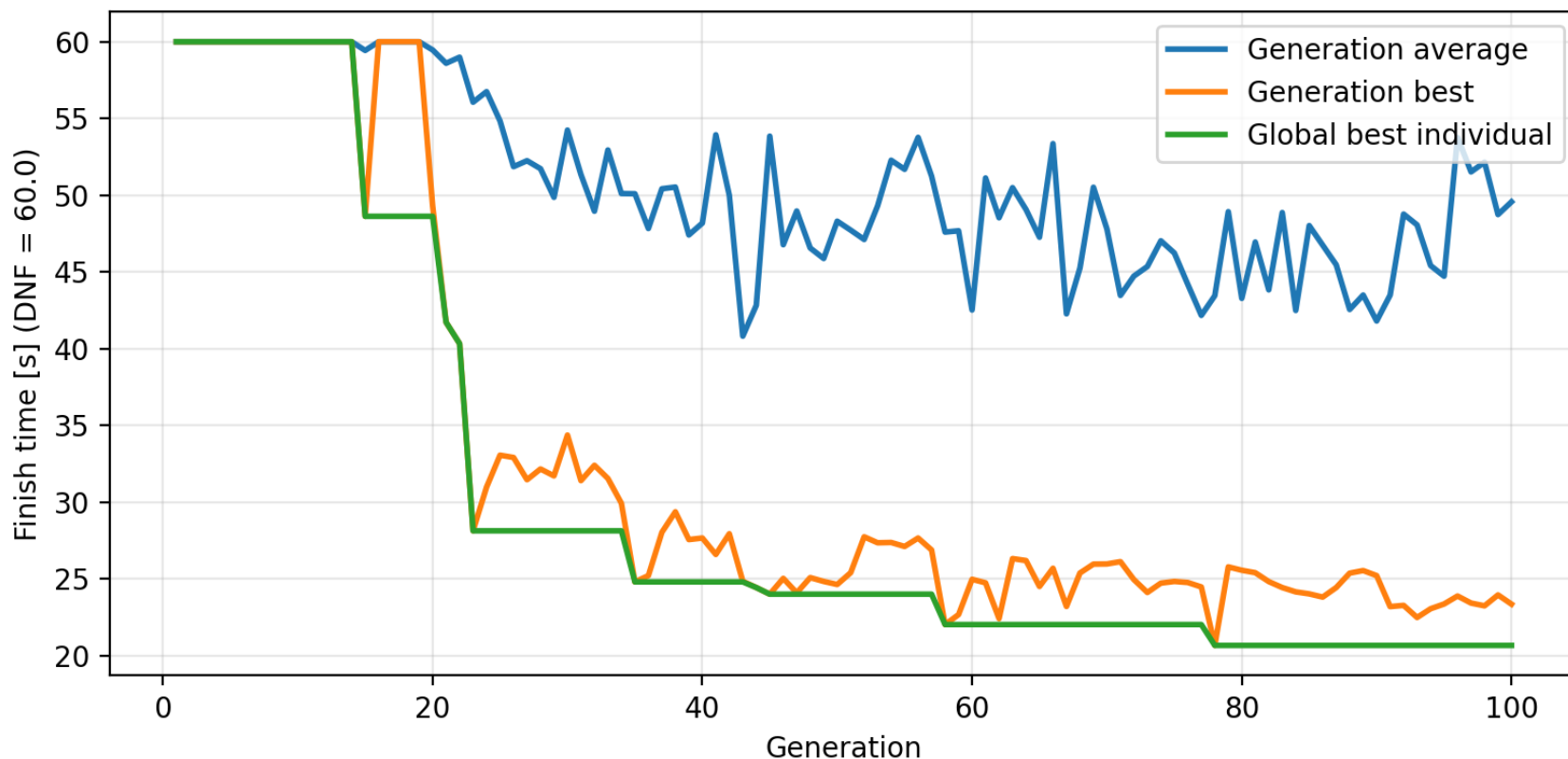
- Pri každom jedincovi meriame viac metrík:
  - Percento prejdenia trate - maximalizujeme
  - Čas jazdy - minimalizujeme
  - Nastala kolízia (bool) - minimalizujeme
  - Celkovo prejdená vzdialenosť (cik-cak = dlhšia dráha pri rovnakom prograse) – minimalizujeme
- Pre jedinca nepočítame jedno „magické“ číslo odmeny, ale každú metriku porovnávame zvlášť

# Výsledky tréningu - prejdená vzdialenosť



- Postupne sa zvyšuje priemerná prejdená vzdialenosť generácie
- Genetický algoritmus skokovo objavuje lepšie riešenia, ktoré sa potom udržia elitizmom.

# Výsledky tréningu - čas jazdy



- Pri rovnakom alebo vyššom prograse sa skracuje čas
- Genetický algoritmus teda nielen dojde ďalej, ale zlepšuje aj rýchlosť jazdy
- Z grafov vidno, že evolúcia dokáže z relatívne náhodných agentov dostať rozumnú pretekársku stratégiu.

# Časová náročnosť a limity

- Máme iba jednu bežiacu inštanciu Trackmanie
  - Jedincov musíme vyhodnocovať sekvenčne
- Jeden beh = jeden agent = jedna jazda
  - Desiatky až stovky generácií × desiatky jedincov
  - tréning trvá hodiny až dni → ťažké experimentovanie
- Limity:
  - obmedzený počet konfigurácií, ktoré reálne otestujeme
  - ťažšie sa hľadá optimálne nastavenie parametrov genetického algoritmu

# Záver

- Z bakalárskeho RL prototypu sme prešli k evolučne trénovanému agentovi
- Agent je stále dopredná NN, ale trénovaná genetickým algoritmom
- Multikriteriálne hodnotenie nám pomáha riešiť konfliktné ciele jazdy
- Prvé výsledky ukazujú, že evolúcia vie agentov postupne výrazne zlepšovať, aj keď je tréning časovo náročný

Ďakujem za  
pozornosť