Connect X

Závěr soutěže pro Seminář z umělé inteligence 2 (NAIL052), MFF UK

Vilém Pech

21. května 2024

Heuristická funkce

Heuristická funkce

S_h Vážená suma (polo)otevřených n-tic hráče h

$$h(stav) = \frac{S_{h^+} - S_{h^-}}{S_{h^+} + S_{h^-} + 1}$$

Heuristická funkce

S_h Vážená suma (polo)otevřených n-tic hráče h

$$h(stav) = \frac{S_{h^+} - S_{h^-}}{S_{h^+} + S_{h^-} + 1}$$

- různé váhy uvnitř S_h

Heuristická funkce

S_h Vážená suma (polo)otevřených n-tic hráče h

$$h(stav) = \frac{S_{h^+} - S_{h^-}}{S_{h^+} + S_{h^-} + 1}$$

- různé váhy uvnitř S_h – téměř žádný vliv

Heuristická funkce

S_h Vážená suma (polo)otevřených n-tic hráče h

$$h(stav) = \frac{S_{h^+} - S_{h^-}}{S_{h^+} + S_{h^-} + 1}$$

- různé váhy uvnitř S_h – téměř žádný vliv
- zefektivnění algoritmu

Heuristická funkce

S_h Vážená suma (polo)otevřených n-tic hráče h

$$h(stav) = \frac{S_{h^+} - S_{h^-}}{S_{h^+} + S_{h^-} + 1}$$

- různé váhy uvnitř S_h – téměř žádný vliv
- zefektivnění algoritmu $\approx +1.2$ hloubky minimaxu

Minimax

```
def minimax(state, depth, maxdepth):
    if state.is_terminal():
        return state.outcome()
    if depth >= maxdepth:
        return h(state)

    best_value = None
    for a in state.possible_actions():
        value = -minimax(state.advance(a), depth+1, maxdepth)
        if best_value is None:
            best_value = value
        else:
            best_value = max(best_value, value)
    return best_val
```

Minimax

```
def minimax(state, depth, maxdepth):
    if state.is_terminal():
        return state.outcome()
    if depth >= maxdepth:
        return h(state)

    best_value = None
    for a in state.possible_actions():
        value = -minimax(state.advance(a), depth+1, maxdepth)
        if best_value is None:
            best_value = value
        else:
            best_value = max(best_value, value)
    return best_val
```

- Problém překročení/nevyužití limitu na tah

Časově omezený minimax

```
def tb_minimax(state, max_time):
    now = time.time()
    if state.is_terminal():
        return state.outcome()
    if now >= max_time:
        return h(state)

    best_value = None
    actions = state.possible_actions()
    subtree_time = (max_time - now) / len(actions)
    for i, a in enumerate(actions):
        value = -tb_minimax(state.advance(a),
                            now + (i + 1) * subtree_time)
        if best_value is None:
            best_value = value
        else:
            best_value = max(best_value, value)
    return best_val
```

Časově omezený minimax

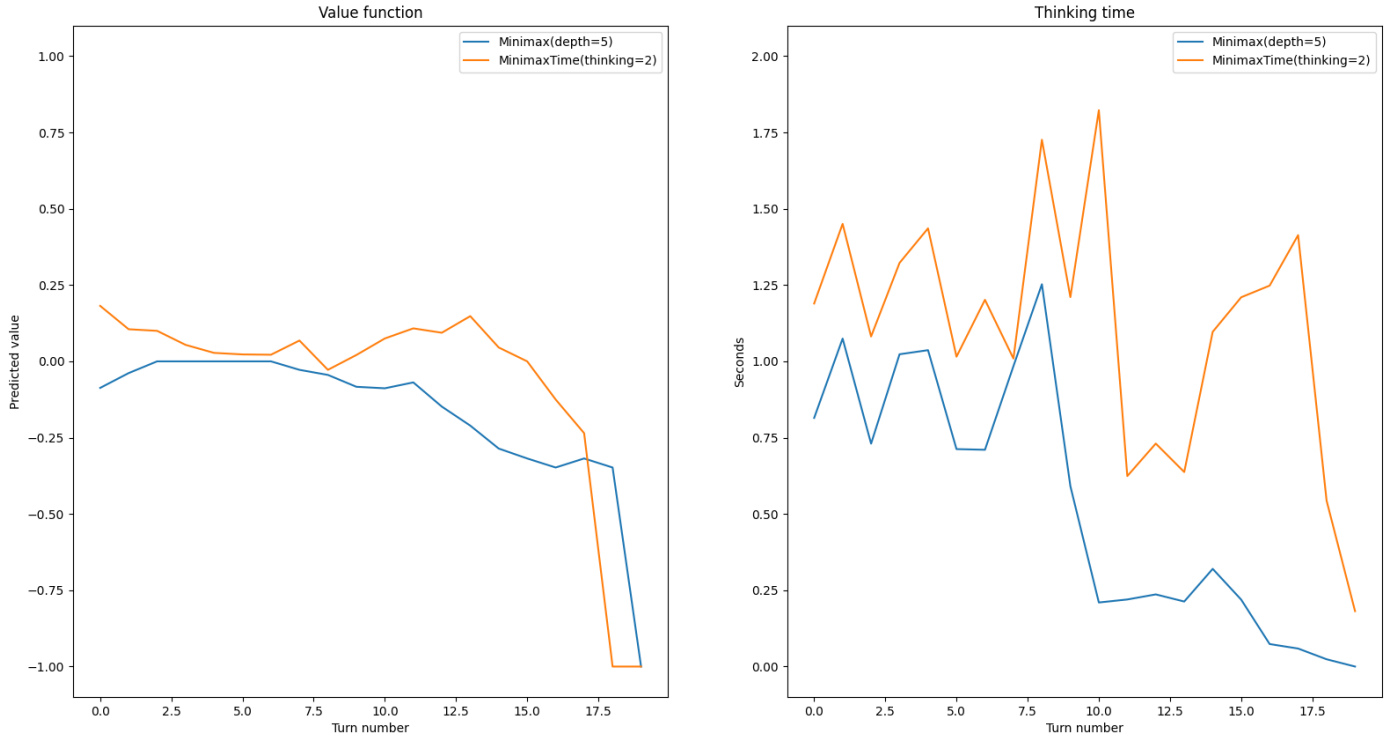
```
def tb_minimax(state, max_time):
    now = time.time()
    if state.is_terminal():
        return state.outcome()
    if now >= max_time:
        return h(state)

    best_value = None
    actions = state.possible_actions()
    subtree_time = (max_time - now) / len(actions)
    for i, a in enumerate(actions):
        value = -tb_minimax(state.advance(a),
                            now + (i + 1) * subtree_time)
        if best_value is None:
            best_value = value
        else:
            best_value = max(best_value, value)
    return best_val
```

- Nevyužívá čas získaný prořezáváním.

Časově omezený minimax – vylepšení

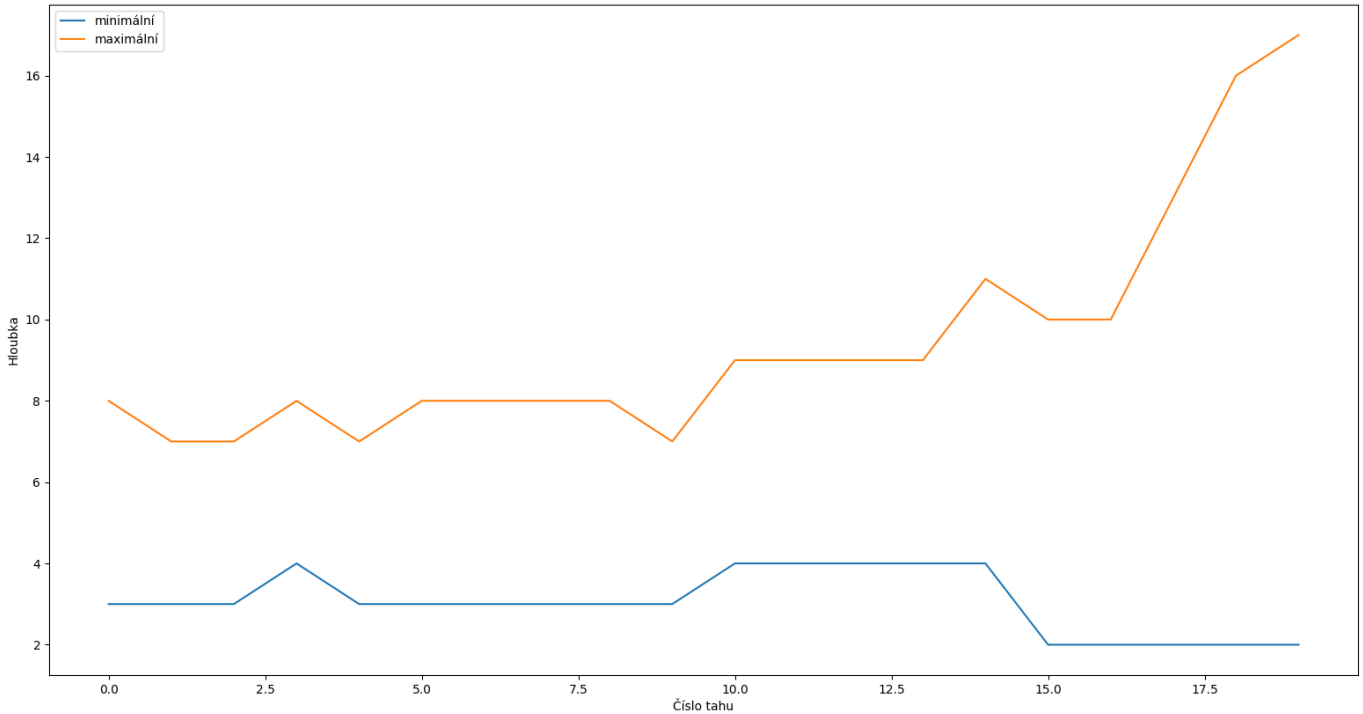
```
def tb_minimax2(state, max_time):
    if state.is_terminal():
        return state.outcome()
    if time.time() >= max_time:
        return h(state)

    best_value = None
    actions = state.possible_actions()
    for i, a in enumerate(actions):
        subtree_time = (max_time - time.time()) / (len(actions) - i)
        value = -tb_minimax2(state.advance(a),
                             time.time() + subtree_time)
        if best_value is None:
            best_value = value
        else:
            best_value = max(best_value, value)
    return best_val
```

Časově omezený minimax – vylepšení



Obrázek: Hodnotová funkce podle minimaxu a č. o. minimaxu (vlevo), využitý čas na tah minimaxu a č. o. minimaxu (vpravo).

Časově omezený minimax – vylepšení



Obrázek: Minimální a maximální hloubka prohledávání v daných tazích.

Problémy časově omezeného minimaxu

Problémy časově omezeného minimaxu

- nerovnoměrné prohledávání stromu

Problémy časově omezeného minimaxu

- nerovnoměrné prohledávání stromu
- nesnadné využití minuty na první tah

Závěr

Závěr

Minimax

Závěr

Minimax

- minimax je velice problémový a neefektivní

Závěr

Minimax

- minimax je velice problémový a neefektivní
- časově omezený minimax je ještě horší než normální

Závěr

Minimax

- minimax je velice problémový a neefektivní
- časově omezený minimax je ještě horší než normální

MCTS

Závěr

Minimax

- minimax je velice problémový a neefektivní
- časově omezený minimax je ještě horší než normální

MCTS

- ve všech praktických ohledech dominuje

Závěr

Minimax

- minimax je velice problémový a neefektivní
- časově omezený minimax je ještě horší než normální

MCTS

- ve všech praktických ohledech dominuje
- s první minutou bylo takřka neporazitelné

Závěr

Minimax

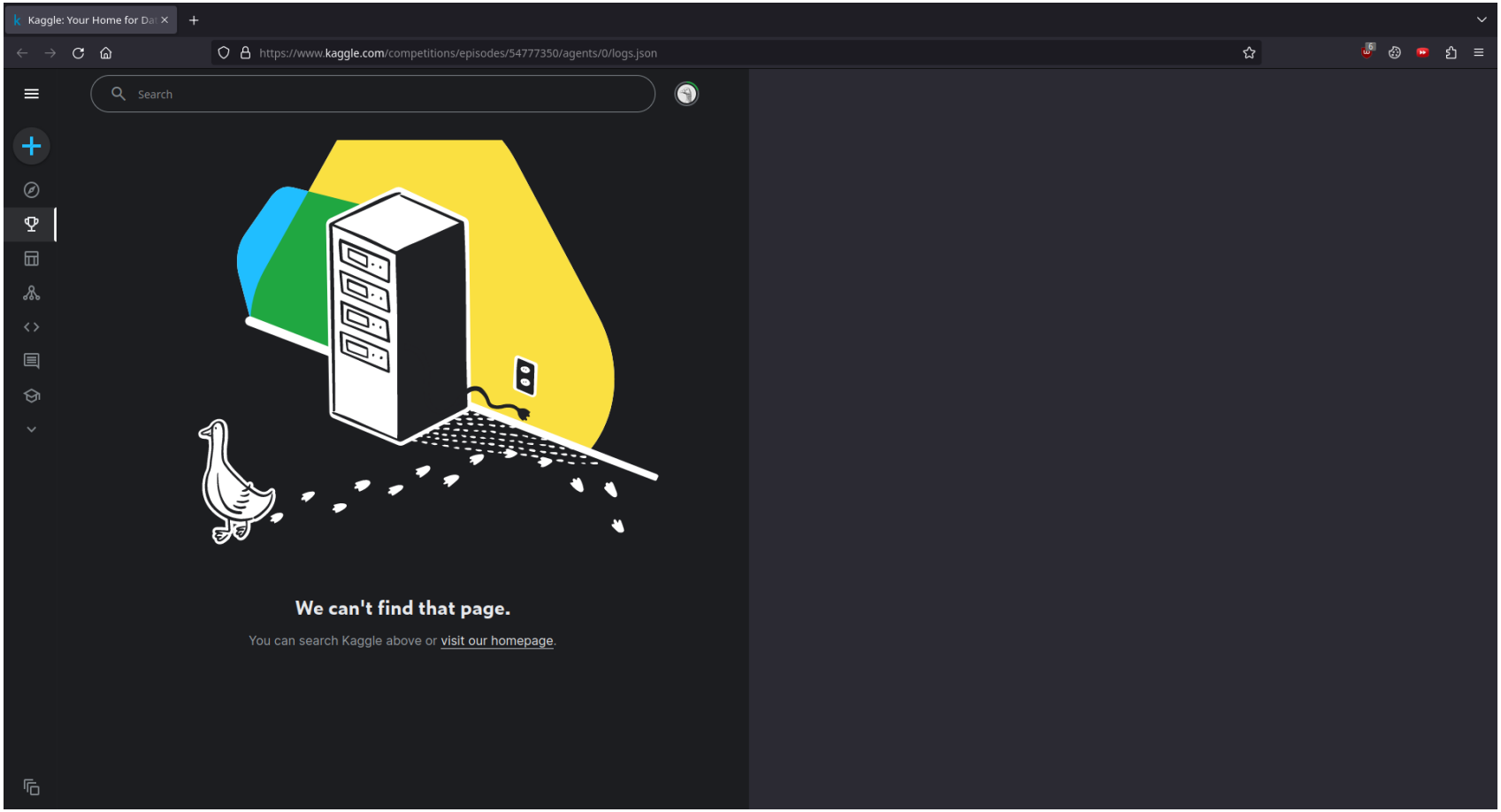
- minimax je velice problémový a neefektivní
- časově omezený minimax je ještě horší než normální

MCTS

- ve všech praktických ohledech dominuje
- s první minutou bylo takřka neporazitelné
- náhodná základní strategie – rychlejší simulace

Výsledek soutěže

Výsledek soutěže



Obrázek: Odkaz na soubor s logem chyby při odevzdání agenta.

Výsledek soutěže

- nelze odevzdat agenta
- naprosto mrvá soutěž
- žádná podpora

Co si z celé soutěže odnést

Co si z celé soutěže odnést

Kéž bych si vybral něco jiného.

(Vilém Pech)