Cieľ projektu:

- Preskúmať použitie Monte Carlo Tree Search (MCTS) pre hru s neúplnou informáciou a dosiahnúť čo najefektívnejšie výsledky.
- Upresnenie: Preskúmať MCTS pre hru Sedma a porovnať prístupy založené na determinizačných simuláciách PIMC s rôznymi roll-out politikami a konfiguráciami.

Zhrnutie:

- V prvom semestri som sa sústredil na oboznámenie s MCTS ako takom a zároveň aj v kontexte hier s neúplnou informáciou. Prácu som smeroval tak, aby sa dalo rýchlo začať implementovať bez veľkých prekážok.
- V letnom semestri som prešiel od návrhu k fungujúcej implementácii: kompletný simulátor hry, niekoľko baseline hráčov, human debug rozhranie a dve implementácie (PIMC) Perfect Information MCTS (jednoduchý doménovo nezávislý a špecializovaný s roll-out limitom).

Sedma pravidlá:

Keďže je sedma čisto Česko-Slovenská hra a na internete sa vyskytujú rôzne verzie pravidiel hry, rozhodol som sa riadiť pravidlami ktoré poznám ja sám.

- budeme uvažovať len verziu pre 2 hráčov
- hrá sa so sedmovými kartami (32 kariet)
- na začiatku každý hráč dostane 4 karty, z ostatných sa stáva doberací balíček
- začína sa tzv. kolo
  - začínajúci hráč položí do stredu na (zatiaľ prázdny) odhadzovací balíček ľubovoľnú kartu z ruky
  - nasleduje jeho protihráč, s tým že karta s rovnakým číslom "prebíja"
  - špecialným prípadom je karta s číslom VII, táto prebíja čokoľvek
  - keď sa dostane na rad opäť začínajúci hráč, už nemôže zahrať ľubovoľnú kartu, ale len takú, ktorá prebíja najspodnejšiu (prvú) kartu. Takýmto ťahom sa opäť zopakuje súperov ťah...
  - začínajúci hráč má ale aj druhú možnosť, a to "zložiť" - teda rozhodnúť sa že ďalšiu kartu nebude hrať (či už z dôvodu že nemá takú kartu, alebo nechce zahrať)
  - po tom čo začínajúci hráč zloží, vezme si odhadzovací balíček k sebe hráč, ktorý ako posledný prebil (ak nikto, tak začínajúci). Karty si nevezme do ruky a nehrá s nimi, len si postupne všetky takto získané karty zbiera pri sebe.
- začínajúcim hráčom ďalšieho kola sa stáva hráč, ktorý bral odhadzovaciu kôpku
- pred ďalším ťahom si obaja hráči doberú z doberacieho balíčka opäť do 4 kariet, ak ich je už menej, vezmu si rovnako veľa
- keď hráči minú všetky karty z balíčka a rúk, spočítajú si vo svojich nazbieraných kartách počet kariet s číslom X a A spolu a kto ich má viac, vyhral (prípadná remíza)

Konkrétne aktivity:

1. Implementácia pravidiel hry
   - SedmaState: plná implementácia pravidiel Sedmy (rozdanie, kontrola správnosti ťahov, výpočet víťaza ťahu, agregácia bodov, ukončenie hry).
   - Reprezentácia kariet, ťahov a celej hry ako samostatné Triedy

2. Baseline hráči a human player
   - RandomPlayer: náhodný ťah z legálnych ťahov.

- HeuristicPlayer: jednoduché doménové heuristiky (prebíja sa, šetri bodové karty (X, A)).
  → Použiteľný ako silnejší baseline a ako rollout policy.
- HumanPlayer: Command Line Interface - rozhranie pre manuálne hranie a debugging (možnosť zobraziť niektoré interné stavy). HumanPlayer výrazne uľahčil overovanie pravidiel a porovnávanie rozhodnutí MCTS, ako aj overenie očakávaného správania.

3. MCTS / PIMC implementácie
   - Determinizer: SedmaState copy-"clone()" a determinize-"shuffleUnknowns()"
     → generátor plných rozdaní konzistentných s pozorovaniami (pre PIMC).
     → Vygenerovanie rýchlych sample-ov s rovnomerným rozdelením kariet protihráča.
   - Základná MCTS infraštruktúra (selection(UCT), expansion, rollout, backpropagation) ako modul separovaný od doménovej logiky (používa len GameState interface).
   - Implementované dve verzie PIMC:
     → Doménovo nezávislý (generický) PIMC: štandardný PIMC štýl — vygeneruje sa niekolko stromov s rôznou (náhodnou) determinizáciou, ktoré potom budujeme ako obyčajné MCTS s random (alebo heuristic) roll-outom; stromy nakoniec nejakým spôsobom spriemerujeme a dostaneme ťah ktorý by mal byť pravdepodobne "dobrý". Tento variant bol navrhnutý ako univerzálny baseline, bez špecifických optimalizácií pre Sedmu.
     → Špecializovaný PIMC s obmedzeným roll-outom: modifikácia, keďže po dohrani jedneho kola vieme posúdiť podľa bodov ktoré ťahy sa oplatili, simulujeme len určitý počet nasledujúcich kôl. Cieľom bolo zredukovať čas rolloutu a zvýšiť kvalitu rozhodnutí tým, že sa venuje väčšia časť výpočtového rozpočtu tree searchu na kritické kroky, nie na dlhé random simulácie.

4. Experimentovanie - testoval som rôzne nastavenia, všetko na 100 hrách, čo by malo dávať aspoň približnu úspešnosť algoritmov

250ms/ťah 50 trees	random rollout			random-heuristic rollout			simple PIMC
	1 round	2 rounds	4 rounds	1 round	2 rounds	4 rounds	
random	26:63	32:46	46:36	35:54	37:45	49:30	44:41
heuristic	52:28	62:23	56:28	56:22	65:16	62:19	78:14

→ v každom jednom porovnaní má PIMC s random rollout-om viac výhier a menej prehier ako PIMC s kombinovaným rollout-om
→ jednoduchý PIMC s takýmito nastaveniami veľmi neobstojí (v priamom porovnaní s náhodnym hráčom bol slabší a s heuristic dopadol podobne ako random, to keď som testoval, tak random vyhral cca 13%)
→ Ďalšie porovnania budeme robiť len s random rollout-om

500ms/ťah	50 trees			200 trees		
rollout	1 round	2 rounds	4 rounds	1 round	2 rounds	4 rounds
random	30:58	42:47	48:40	27:63	43:44	40:44
heuristic	54:34	57:28	59:25	59:29	74:13	58:25

→ nevieme úplne povedať čo je výhodnejšie
→ vidíme že najlepšie výsledky ma doposiaľ vo všetkých testoch MCTS so simulovaním iba 1 kola

150ms/ťah | 1 round random rollout				
trees	50	100	150	200
heuristic	53:26	53:29	47:33	48:38

Záver:

MCTS môže mať veľký význam aj na problémoch s neúplnou informáciou, no vidíme že to o dosť viacej závisí od konkrétneho problému. Schopný MCTS engine bude pravdepodobne vyžadovať nejaké špecifické optimalizácie, no sám som sa presvedčil že sa dajú dosiahnúť aj pekné výsledky. Heuristického hráča sa mi síce s použitím iba zopár optimalizácii nepodarilo úplne prekonať, no keď som sa s tým trochu vyhral, zaznamenal som jednoznačný posun v úspešnosti aj pri pomerne krátkych časových limitoch na ťah.

Dôvodom prečo som neprekonal heuristického hráča môže byť samozrejme aj výber problému (sedma), je to podľa mňa totiž hra, na ktorú poznáme celkom jednoduchú a aj napriek tomu takmer optimálnu stratégiu. Teda aj keď heuristický hráč nehrá optimálne, hrá veľmi dobre.

V tomto projekte som sa pozrel len na jeden z viacerých možných prístupov k neúplnej informácii v MCTS o ktorých som sa dočítal. Možnosti je teda jednoznačne veľa aj na riešenie komplexnejších problémov a určite sa na ne ešte rád pozriem v ďalšom štúdiu.