Letný semester

sort = Usporiadanie ťahov (v akom poradí sa majú prehľadávať) pre danú pozíciu.
eval = Ohodnotenie pozície číslom z intervalu <-1;1> (odhadnutie, kto vyhráva).

Zamerali sme sa na zlepšenie EngineAB, kde bola hlavná nevýhoda pomalá heuristika (ozn. Heur1), ktorá trvala priemerne 99,8% ťahu. Preto len do hĺbky 4 trval ťah priemerne až 40 sekúnd. *Sort* aj *eval* využívali rovnakú heuristiku Heur1, ale *sort* približne až 35 krát častejšie. Takže najpomalšou časťou ťahu bol *sort* (97% času), a potom *eval* (2,8% času).

Na *sort* sme teda použili lepšiu heuristiku (ozn. Heur2), ktorá vyhodnocuje všetky „dobré tvary" rozmiestnenia kameňov pre oboch hráčov pri jedinom prechode hracej plochy. Čo zrýchlilo *sort* cca 12 násobne. Takže do hĺbky 4 trval ťah priemerne 4 sekundy (*eval* 23%, *sort* 76%). Priemerný počet volaní Heur1 v *evale* na ťah sa nezmenil, takže *sort* s Heur2 je zhruba rovnako dobrý ako s Heur1. *Eval* sme nemenili, takže tento engine (ozn. Eng1) hrá rovnako dobre ako pôvodný (ale rýchlejšie, takže za rovnaký čas dokáže prehľadávať do väčšej hĺbky, a teda nájsť lepší ťah).

Vyskúšali sme Heur2 použiť aj na *eval* (tento engine ozn. Eng2). Eng2 orezáva o niečo viac ťahov ako Eng1 (za ťah prehľadá 7600 pozícií namiesto 9400). Vďaka čomu sa *eval* zrýchlil až 20 násobne (z 900ms na 45ms). Zároveň to zrýchlilo aj *sort* z 3 sekúnd na 1,8 sekundy. Priemerná dĺžka ťahu je teda 1,9 sekundy. Ale Eng1 proti Eng2 vyhral 36 hier zo 60 (pri rovnakej hĺbke 4). Takže Heur1 sa zdá byť trochu lepšia ako Heur2, keď nerátame ich trvanie.

Dostali sme sa ale k rovnakému problému ako na začiatku. *Sort* sa vykonáva 97% času ťahu a *eval* 2,4%. Potrebujeme zrýchliť *sort*.
Pamätáme si množinu (ozn. relSq) voľných políčok v okolí už položených kameňov. Túto množinu po každom zahranom ťahu (svojom aj súperovom) aktualizujeme v konštantnom čase – odstránime políčko položeného kameňa a pridáme okolie tohto kameňa.
V *sorte* nebudeme rátať heuristiku pre každý ťah, ale v jednom prechode cez hraciu plochu ohodnotíme každé políčko číslom a následne políčka z relSq usporiadame podľa tohto čísla. Priemerná dĺžka ťahu tohto engine (ozn. Eng3) je 150ms (*eval* 58%, *sort* 20%, ostatné 22%). Veľkosť relSq je priemerne 70, takže Eng3 z danej pozície simuluje priemerne 2 krát viac ťahov ako Eng2 alebo Eng1. Kvôli tomu prehľadá za ťah až 12200 pozícií (Eng2 len 7600), (toto číslo je ovplyvnené aj iným usporiadaním ťahov). A keďže prehľadáva viac pozícií, tak sa *eval* vykonáva viac krát, a teda dokopy trvá o niečo dlhšie ako v Eng2, aj keď oba používajú v *eval* rovnakú Heur2. Teda Eng3 a Eng2 by mali hrať rovnako dobre za predpokladu, že ich množiny skúšaných ťahov pre každú pozíciu sú vhodne zvolené – obsahujú najlepší ťah vzhľadom na Heur2. Eng3 nad Eng2 vyhral 33 zo 60 hier. Eng3 nad EngineTS vyhral 166 z 200 hier.

Memoizácia

Majme graf, ktorého vrcholy sú možné pozície v hre, a hrana z A do B vedie práve vtedy, keď existuje políčko, ktoré je v A prázdne, v B nie je prázdne, a na všetkých ostatných políčkach sa A, B zhodujú. Táto hrana symbolizuje položenie kameňa na dané políčko. Tento graf je acyklický, keďže kamene v ňom len pribúdajú. Podobá sa na strom, ale niektoré vetvy sa znova stretajú. Môže sa teda stať, že niektoré výpočty budeme opakovať znova. Zároveň dva po sebe idúce ťahy jedného hráča sa dosť podobajú: ten druhý ťah začína na niektorej vetve predchádzajúceho ťahu, opakuje rovnaké výpočty, ale ide do väčšej hĺbky. Tu si vieme pomôcť memoizáciou. Budeme si pamätať vypočítané heuristiky z aktuálneho, ale aj

predošlého ťahu. Čím väčšia bude hĺbka, do ktorej engine prehľadáva, tým sa bude viac vetiev spájať, a memoizácia nám o to viac pomôže. Ale na druhej strane bude narastať aj veľkosť pamätaných dát.

hĺbka	memo z tohto ťahu	memo z predchádzajúceho ťahu
4	25%	1%
5	49%	1,4%
6	53%	1,2%
7	66%	1,4%

(v hĺbke 8 po prvých pár ťahoch dosiahlo memo pamäťový limit)

Memo z predchádzajúceho ťahu sa neukázalo ako veľmi užitočné, keďže „strom“ je veľmi „košatý“, a veľká väčšina vrcholov je v najväčšej hĺbke, ktorú z predchádzajúceho ťahu nemáme. Ale celkovo memo pomoholo – napríklad v hĺbke 5 približne 1,5 násobne zrýchlilo ťah.

Od nájdenia stratégie pre začínajúceho hráča má tento projekt ešte ďaleko, ale vyrobili sme engine, proti ktorému sa dá dosť kvalitne zahrať s možnosťou nastavenia obťažnosti (hĺbkou prehľadávania).
A zaujímavý je aj EngineTS, ktorý má v sebe aj kus náhodnosti, nájde ťah do niekoľko desiatok milisekúnd a aj tak dokáže občas poraziť EngineAB ktorému to trvá výrazne dlhšie. Napríklad proti EngineAB do hĺbky 7 vyhral 1 z 10 hier, keď začínal. Priemerná dĺžka ťahu EngineTS 20ms, EngineAB 100 sekúnd.