AI (nejen) v krabičkách od sirek

Soustředění mladých matematiků a fyziků 2022

Autoři: Richard Dobíšek, Vojtěch Procházka

Vedoucí: Lýda Cehákova

Anotace

V tomto projektu jsme se zabývali vytvořením AI v krabičkách od zápalek pro hru hexapawn na poli 3x3 a dále AI v počítači pro stejnou hru na poli 4x4. Dále jsme se zabývali analyzováním dat s trénováním a zkoušením různých způsobů tréninku.

Annotation

In this project we made an AI for the game Hexapawn on a 3 by 3 board in matchboxes and a computer version for a 4 by 4 board. Next, we analysed data from training the AI and tested different training methods.

Table of Contents

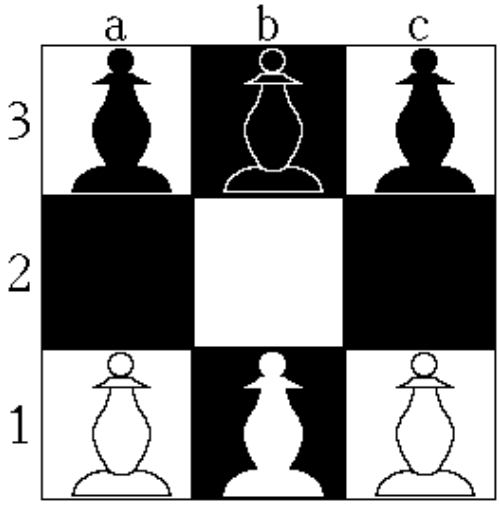
Úvod

V této práci jsme se zabývali výrobou umělé inteligence pro hru Hexapawn nejdříve na poli 3x3 a poté v podobě krabiček od zápalek a poté na poli 4x4 ve formě počítačového programu, který funguje na úplně stejném principu. Pro první nápad na hru a AI z krabiček od sirek jsme se inspirovali podle článku ze Scientific American od Martina Gardnera[1].

Fyzická 3x3 Hexapawn

Hexapawn pravidla.

Hra hexapawn se hraje na šachovnici 3 x 3 s bílými a černými šachovými pěšci. Pěšci začínají rozestaveni na krajních pozicích (viz obrázek). Bílý začíná a hráči se střídají na tahu.
V každém tahu musí hráč pohnout jedním ze svých pěšců. Pěšci se pohybují stejně jako v šachách, to znamená, že buď se pohnou o jeden krok dopředu, nebo vezmou jiného pěšce diagonálně. Hráč vyhraje když, buď dopraví jednoho ze svých pěšců na poslední řádek, sebere všechny soupeřovy pěšce, nebo zabrání soupeřovi v táhnutí.



Obrázek 1: Základní pozice hry hexapawn

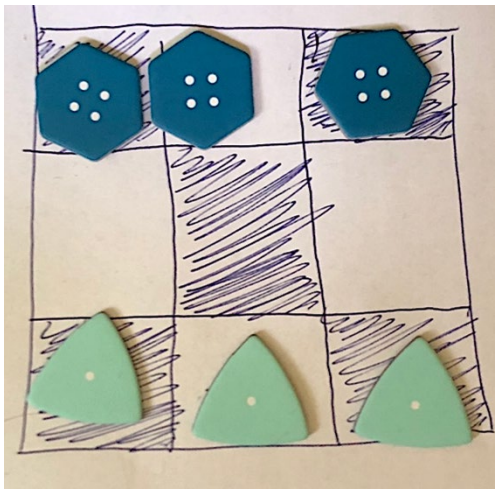
Jak funguje zápalkové AI

Zápalkové AI funguje v několika krabičkách od zápalek, kde na každé krabičce je nakreslená jedna z možných pozic, ve kterých se hra může nacházet. V krabičkách se nachází jeden korálek za každý možný tah, co počítač může udělat. Když se počítač dostane na tah, z krabičky s pozicí, ve které se hra nachází, se náhodně vytáhne jeden korálek a zahraje se příslušný tah. Po skončení hry se počítač učí odměnami a tresty. Trestá se odebráním posledního zahraného korálku a odměňuje se přidáním korálků do každé krabičky, přes kterou se dostal až k vítězství. AI je postavená, aby hrála za černého, který má v této pozici garantovanou výherní strategii a trénuje se k tomu, aby ji ve všech pozicích našel.

[1] Gardner Maertin. How to build a game-learning machine and then teach it to play and to win. *Scientific American*, 1962, 206.3: 138-144..

Jak jsme konstruovali

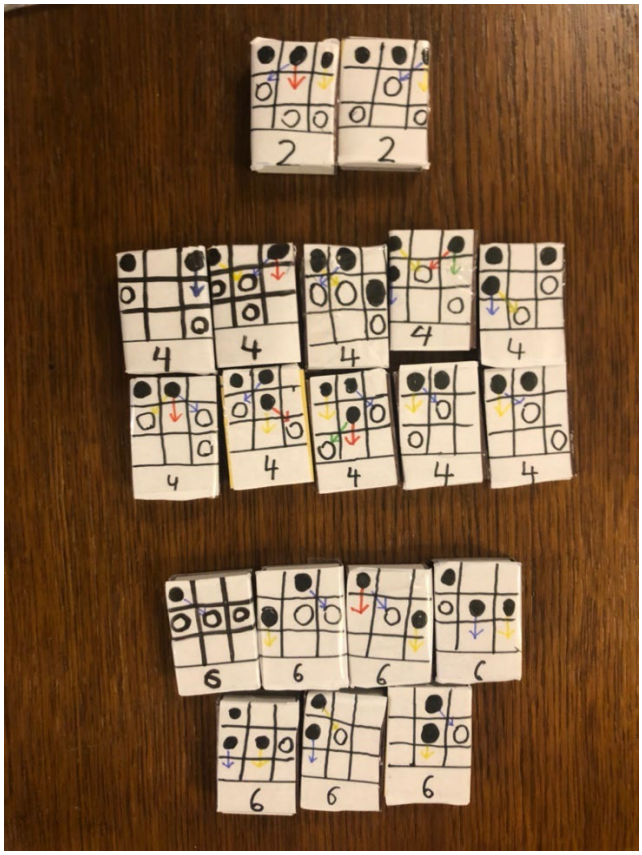
AI z krabiček jsme konstruovali z krabiček od zápalek. Na každou krabičku jsme nalepili jednu z možných pozic herního pole, které jsme nakreslili. Kreslili jsme podle všech konfigurací, do kterých se nám povedlo hru dostat, a poté jsme je překontrolovali podle článku, který jsme našli[2], kde byly obrázky všech krabiček nutných pro hru. Na každé krabičce je nakreslena mřížka 3x3 a v ní pozice všech pěšců znázorněné tečkami příslušné barvy. Dále jsou na mřížce vyznačené možné tahy[3] barvami. Pod mřížkou je napsáno číslo tahu, ve kterém se hra nachází, které slouží pro rychlejší nalezení krabičky, kterou má AI zahrát. Do krabiček jsme místo korálků dali nastříhané kousky barevných brček, které ale slouží stejnému účelu. Hru jsme místo ze šachových pěšců hráli s barevnými žetony a šachovnici jsme si nakreslili na papír.



Obrázek 2: Naše pole pro hraní hexapawn

[2] Gardner Maertin. How to build a game-learning machine and then teach it to play and to win. *Scientific American*, 1962, 206.3: 138-144..

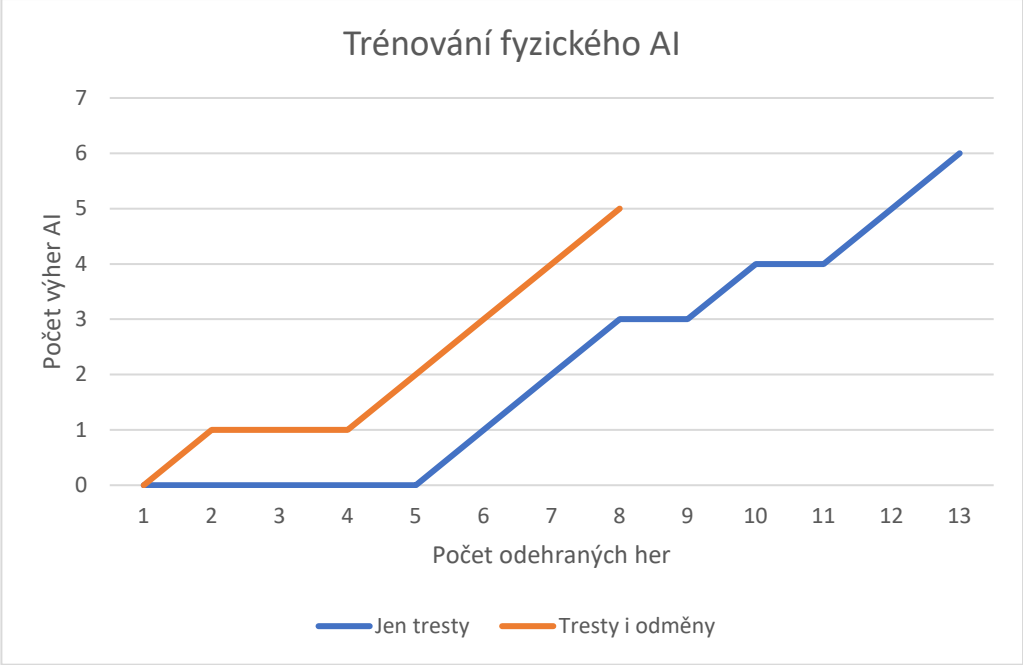
[3] Ne všechny, protože by způsobovali úplně stejnou pozici, ale překlopenou podle osy y



Obrázek 3: Kompletní fyzické krabičkové AI -> 19 krabiček

Pozorování

Při trénování krabičkového AI pomocí trestání jsme po 13 hrách dosáhli stavu, ve kterém bylo AI vlastně neporazitelné. Při trénování pomocí odměn a trestů jsme ho trénovali jen 8 hrami a přišlo nám, že je vytrénovaný možná ještě lépe než při trénování jen tresty. Při trénování tresty zvládl za 13 her vyhrát 6krát. Při trénování tresty a odměnami nás zvládl za zahraných 8 her porazit 5krát. Postup AI znázorňuje následující graf.

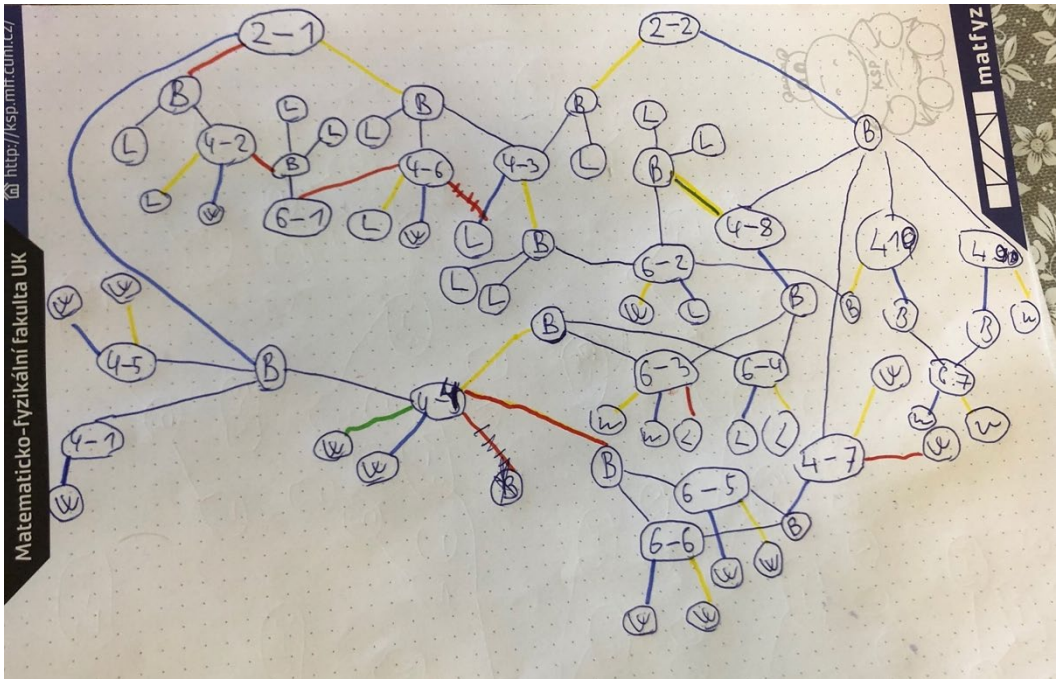


Obrázek 4: Graf trénování fyzické AI

Z tohoto grafu je možno vypozorovat, že s odměňováním má AI více výher, což znamená, že trénink bude trvat déle, ale v průběhu bude AI silnější než při trestání. Toto se projevilo při trénování, když jsme vzdali trénování odměňovacího AI po frustraci z toho, že nás neustále porážel. Naopak jenom trestání může být lepší volba tréninku, když je potřeba rychle vytrénovat AI a je nám jedno, kolik za trénování bude mít výher.

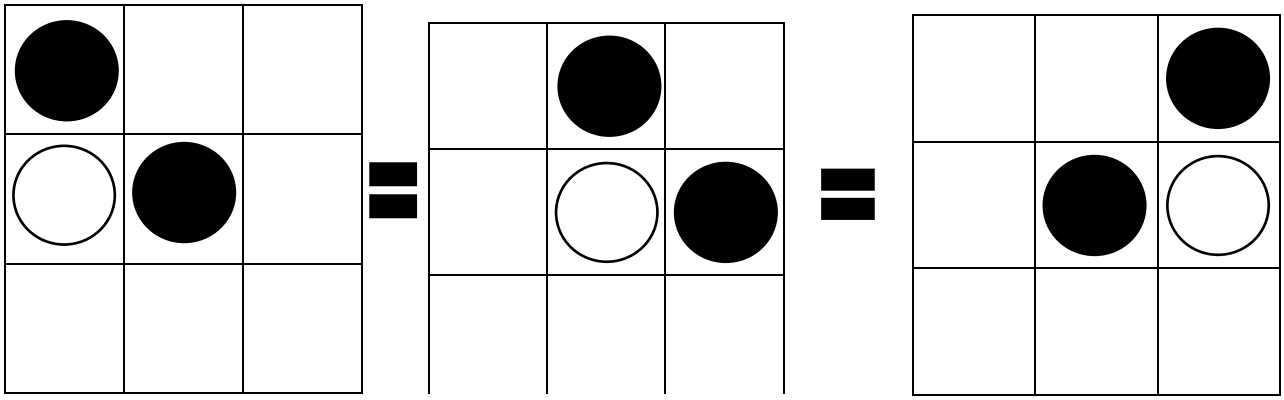
Rozhodli jsme se krabičkovou hru rozšířit na pole 4x4 se 4 pěšáky na každé straně. Bohužel se ukázalo, že stavový prostor hry je příliš veliký na to, abychom ho pokryli krabičkami ze sirek, takže jsme se rozhodli hru naprogramovat.

Programovali jsme v C#, a náš algoritmus funguje prakticky stejně jako krabičkový algoritmus na 3x3. Nejdříve si nakreslíme všechny krabičky a do každé si hodíme korálek za každý tah, který můžeme zahrát. Prakticky projdeme všechny pozice, které můžou nastat a uděláme si orientovaný graf, který je propojí. Uděláme to pomocí klasického BFS. Ve vrcholu bude počáteční pozice-tah 0, z ní povedou cesty do pozic, do kterých se můžeme z počáteční pozice dostat. Z pozic tahu 1 povedou cesty dál a dál, dokud nebudeme mít kompletní orientovaný graf.



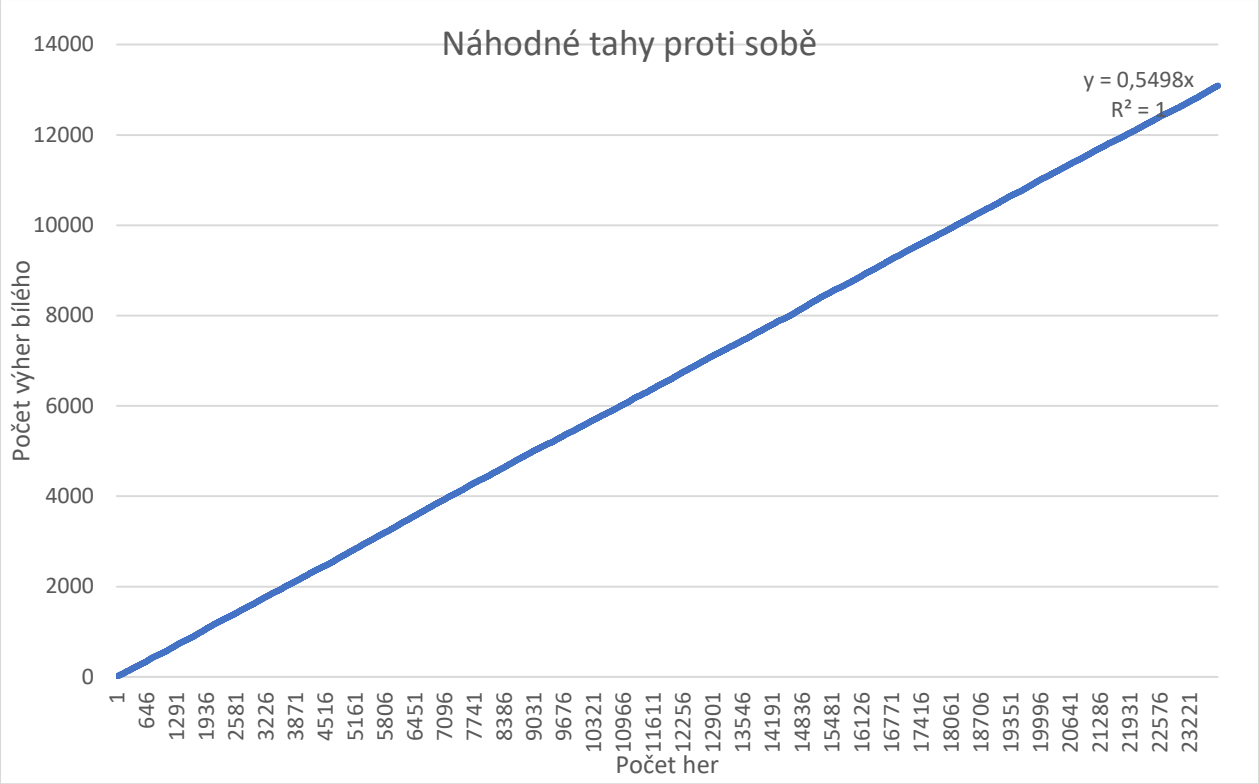
Obrázek 5: Graf všech možností, jak hrát hru hexapawn 3x3. Barevně zakreslené jsou možnosti AI v jednotlivých pozicích

Počet všech možných pozic je u 4x4 několik desítek tisíc, takže v našem kódu všechny pozice optimalizujeme, aby náš algoritmus považoval všechny posunuté a překlopené, ale vlastně stejné pozice, za identické. To nám pak působilo problémy při zpětném zobrazování pozice, jelikož jsme data kompresovali.



Obrázek 6: Schéma různých polí, co se zoptimalizují na stejný

Ještě před tím, než jsme ale měli kód kompletní, tak nás ještě zajímalo, který z hráčů má výhodu. Ve 3x3 to byl černý, ale pro 4x4 nám to nešlo tak lehce vysvětlit. Tak jsme zkusili udělat algoritmus, který dělá náhodné tahy z možností a nechali ho hrát samotného proti sobě, abychom zjistili, kdo má výhodu.



Obrázek 7: Graf výher náhodného algoritmu

Z grafu jde vidět, že z asi 24 000 her jich bílý vyhrál okolo 13000. To mu dává 55% úspěšnost. Toto ale nutně nemusí znamenat, že má výherní strategii. Možná jen má černý více příležitostí zahrát špatný tah někdy na začátku. Určitě to ale minimálně u nás posílilo ideu, že bílý má výhodu.
Pak jsme se dali do trénování. Základní trénovací algoritmus, který jsme používali byly: pouze trestání, odměňování dvěma korálky a trestání, a odměňování třemi korálky bez trestání.

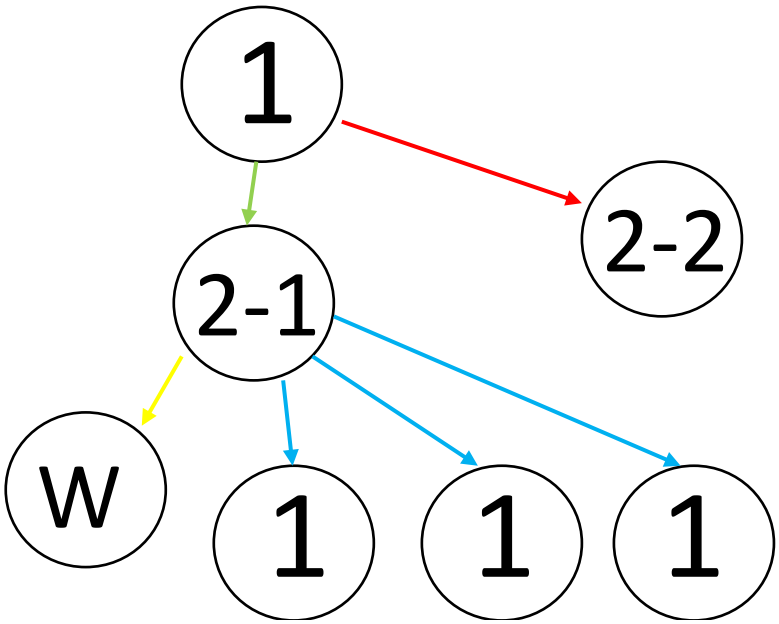
Algoritmy se trénovaly samy se sebou
Zde jsou jejich výsledky

	200_o2t	1000_o2t	5000_o2t	200_o3	1000_o3	5000_o3	200_t	1000_t	5000_t
200_o2t	224-276	125-375	135-365	210-290	181-319	88-412	330-170	344-156	393-107
1000_o2t	358-142	185-315	167-333	304-196	241-259	193-307	372-128	398-102	450-50
5000_o2t	419-81	348-152	429-71	387-113	398-102	391-109	445-55	457-43	493-7
200_o3	294-206	168-332	201-299	280-220	179-321	125-375	356-144	369-131	437-63
1000_o3	314-186	261-239	260-240	262-238	329-171	152-348	375-125	391-109	450-50
5000_o3	404-96	189-311	215-285	396-104	275-225	379-121	425-75	445-55	446-54
200_t	148-352	90-410	101-399	123-377	89-411	78-422	223-277	238-262	328-172
1000_t	122-378	89-411	120-380	100-400	66-434	78-422	211-289	223-277	320-180
5000_t	127-373	91-409	128-372	87-413	63-437	89-411	176-324	189-311	330-170

Obrázek 8: Tabulka her botů proti sobě, bot vlevo hraje za bílé, boti trénováni sami proti sobě

V tabulce jsou v horním a levém řádku různě vytrénované algoritmy. Číslo značí, kolik bylo tréninkových her a symbol za znamená, jakým způsobem se AI trénoval. T znamená trestání

a o znamená odměňování tolika korálky, kolik je napsáno za ním. AI proti sobě následně hráli 500 her a výsledky jsou zaneseny v tabulce. AI, které je směrem doleva hrálo za bílé a nahoru hrálo za černé.
Můžeme si všimnout, že výhradně trestající roboti nedosahují vůbec dobrých výsledků. Následující obrázek vysvětluje proč.



Obrázek 9: Ilustrační graf potenciální nefunkční situace

Hraje trestající bot proti trestajícímu botovi. 1 zahraje zelený prohrávající tah a dostane se do vrcholu 2-1. 2 má několik možností a náhodně vybere tu žlutou vyhrávající. Ostatní modré jsou prohrávající. 1 se z toho poučí a z vrcholu 1 nahoře už nikdy nevybere tu zelenou prohrávající možnost. Pak se ale tento trestající bot setká se soupeřem, který tuto chybu udělá a on zjistí, že vlastně neví, co má udělat potom. Šance na to, že poté vybere jedinou vyhrávající možnost je malá. To je důvod proč trestající bot naprosto selhává a ani ostatní boti nedosahují takových výsledků.
Toto demonstruje vlastnost AI obecně, že jako nenatrénovaní nedělají správné tahy a trénují se podle dat, co my jim poskytneme. V případě, že jim poskytneme dobrá data, tak s tím není žádný problém. Pokud jim ale dáme špatná data, tak je můžeme vytrénovat k tomu, aby byli ještě horší než nevytrénovaní.

Proto se nabízí varianta trénování robotů jinými způsoby. My jsme se rozhodli trénovat je adaptivním o3 5000 robotem. Ten se natrénuje 5000 hrami sám se sebou pouze odměňováním. Pak začneme tímto robotem trénovat ostatní s tím, že se tento o3 5000 bude také učit.

Výsledky těch samých trénovaných už předem vytrénovaným adaptivním o3 5000,
Zde jsou výsledky

	200_o2t	1000_o2t	5000_o2t	200_o3	1000_o3	5000_o3	200_t	1000_t	5000_t
200_o2t	298-202	282-218	265-235	290-210	302-198	281-219	301-199	269-231	263-237
1000_o2t	317-183	290-210	285-215	308-192	318-182	306-194	327-173	298-202	283-217
5000_o2t	369-131	314-186	387-113	309-191	405-95	426-74	387-113	307-193	387-113
200_o3	303-197	286-214	242-258	278-222	324-176	308-192	302-198	247-253	245-255
1000_o3	231-269	222-278	298-202	231-269	205-295	299-201	318-182	174-326	299-201
5000_o3	242-258	134-366	111-389	256-244	126-374	60-440	326-174	138-362	113-387
200_t	253-247	232-268	234-266	249-251	218-282	217-283	269-231	203-297	239-261
1000_t	241-259	188-312	195-305	241-259	189-311	202-298	285-215	141-359	196-304
100000_t	472-28	494-6	466-34	359-141	487-13	499-1	458-42	382-118	466-34

Tabulka her botů proti sobě, bot vlevo hraje za bílé, boti trénováni sami proti sobě

První číslo ukazuje počet výher bílého, za pomlčkou je počet výher černého
Kdybychom dali AI více her na trénink, (viz 10000_t) tak bychom došli k závěru, že bílí má pozici objektivně vyhranou a prakticky vždy vyhraje.

Závěr

Pomocí jednoduchého principu se nám podařilo vytvořit AI, které se dokáže samo vytrénovat a prakticky vyřešit jednoduché hry. Síla AI závisí přímo na počtu odehraných her, trenérovi i způsobu odměňování. Výhoda našeho AI je, že se natrénuje, aniž by muselo projít všechny možné sekvence tahů a může ho naprogramovat i člověk, který vlastně hexapawn neumí hrát. Nevýhody spočívají v tom, že si musíme vlastně nakreslit úplný orientovaný graf všech pozic, které mohou nastat, takže tuto techniku nemůžeme použít na komplexnější hry.

Bohužel jsme kvůli času nestihli statistiky různých způsobů trénování doladit, takže možná vyjdou