Vytvoření oddílů na šachový tábor

Adéla Bierská, 493121

Popis problému

Cílem tohoto projektu je aplikace, která bude rovnoměrně rozdělovat děti do oddílů na dětském šachovém táboře. O dětech kromě jejich věku víme i jejich šachovou výkonost (ELO) a máme hodnocení jejich sportovních, vůdčích a herních dovedností od jejich vedoucích z minulých ročníků tábora.

Hodnocení sportovních a herních dovedností u každého dítěte jsou průměrné hodnoty hodnocení vedoucími na stupnici -3 až 3. Vůdčí dovednosti mají rozsah stupnice pouze 0 až 3. Počáteční hodnota ELO je 1000, může dosahovat hodnot klidně i přes 2600, ale na našem táboře se pohybujeme maximálně kolem 2100.

Na oddíly máme následující požadavky:

- Je jich 8, každý má stejné množství (± 1) dětí.
- Věkové rozložení dětí v oddílech je co nejrovnoměrnější, tj. měly by mít co nejpodobnější průměr a zároveň by v každém oddíle měly být zastoupeny všechny věkové skupiny.
- Dívky a nové děti by měly být rozděleny v oddílech rovnoměrně.
- Žádní dva sourozenci nesmí být v jednom oddíle.

Jelikož se jedná o NP-těžký problém, je pro jeho řešení třeba využít heruistik. V rámci projektu jsem se rozhodla jich vyzkoušet více, a to hladový algoritmus, tabu table, genetický algoritmus, simulované žíhání a výběr z N sousedních řešení.

Popis a hodnocení implementace

Program je napsán v jazyce C#, nebyly použity žádné externí knihovny kromě knihovny na statistické funkce. Použitá verze .NETu je 8.0.

Pokyny ke spuštění

Je třeba mít nainstalovaný .NET 8.0. Kód se spouští příkazem:

```
dotnet run --project PA026
```

Očekávaným vstupním souborem je TSV se seznamem dětí a CSV obsahující řádky se skupinami sourozenců. S projektem je dodaný anonymizovaný soubor dětí, cesty k souborům jsou vepsány v `Program.cs`.

Inicializace řešení

Po načtení souboru s dětmi jsou vyplněny chybějící hodnoty (např. hodnocení u nových dětí) prostým průměrem daného sloupce nebo průměrem k nejbližších sousedů.

Problém je reprezentován jako seznam hodnot v rozsahu 0 až 7, kde index v seznamu odpovídá id dítěti v iniciálním seznamu a hodnota 0 až 7 přiřazení tohoto dítěte do oddílu. Pro optimalizaci řešení jsem naimplementovala krom náhodné inicializace 2 pokročilejší algoritmy:

Inicializace dle věku

Seřadí děti dle věku a poté jen periodicky přižazuje hodnoty 0 až 7, čímž dosáhně předtřízení oddílů dle věku. Tím se zvýší kvalita iniciálního řešení a rychlost nalezení dobrého rozdělení, zároveň ale hrozí zapadnutí do lokálního optima.

Inicializace pomocí řídících pravidel

Zkombinuje věk a hodnocení dětí do jedné hodnoty společně s koeficientem pro dívky a nové děti. Pomocí prioritní fronty vždy vybírá dítě s nejvyšším tímto skóre a přiřadí jej do oddílu s nejnižším aktuálním skóre. Již iniciální řešení je velmi obstojné, ale obtížně se následovně zlepšuje heruistikami.

Hledání nejlepšího řešení

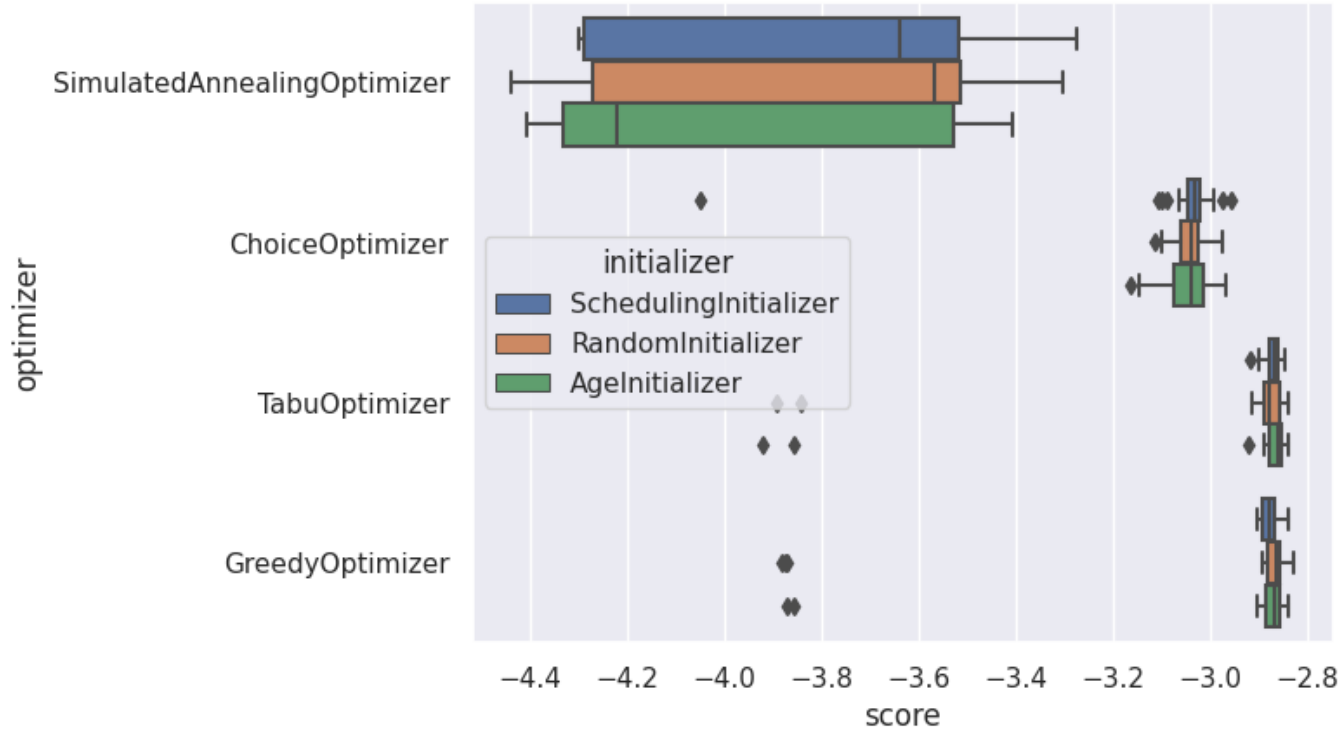
Pro hledání nejlepšího řešení byly použity algoritmy učené v předmětu IV126, a to hladový algoritmus, tabu table, genetický algoritmus, simulované žíhání a výběr z N sousedních řešení. Jako sousední řešení bereme řešení, která má vyměněné hodnoty na 2 pozicích v seznamu.

Program po skončení vypíše statistiky o jednotlivých oddílech, aby bylo možné jednodušeji posoudit výsledky. Zde je příklad takovéhoto výstupu:

Name	Age	ELO	Height	OtherSkills	SportSkills	Leaders	Newbies	Girls
0	12.79 3.19	1448.2 431.1	150.0 00.0	0.26 0.97	0.50 1.44	0.44 0.65	3	3
7	12.85 2.76	1283.1 323.7	150.0 00.0	0.83 1.27	0.45 0.97	0.48 0.78	2	4
4	12.85 3.05	1183.6 312.9	150.0 00.0	1.26 0.85	0.93 0.93	0.17 0.32	2	4
6	12.77 3.47	1305.2 360.3	150.0 00.0	0.78 0.83	0.48 1.15	0.45 1.01	3	3
2	12.71 2.73	1310.9 384.2	150.0 00.0	0.77 1.13	0.11 1.41	0.79 0.75	2	4
3	12.69 2.18	1288.3 332.8	150.0 00.0	0.77 0.94	0.45 1.00	0.50 0.81	2	4
1	12.79 4.04	1309.5 339.2	150.0 00.0	0.77 0.86	0.47 1.07	0.46 0.78	2	4
5	12.77 2.59	1278.2 317.7	150.0 00.0	0.86 0.83	0.51 0.92	0.44 0.52	2	4
StdDev	00.05 0.58	0072.4 040.3	000.0 00.0	0.27 0.16	0.28 0.21	0.17 0.21	0.46	0.46

Vybrané algoritmy byly hodnoceny pomocí benchmarku na 30 iteracích v kombinaci se všemi druhy inicializace. Výsledky tohoto benchmarku jsou v následujícíc tabulce a grafu:

		score	time
initializer	optimizer		
SchedulingInitializer	TabuOptimizer	-2.872637	34.403733
	GreedyOptimizer	-2.877040	40.973967
AgeInitializer	TabuOptimizer	-2.932897	34.285900
	GreedyOptimizer	-2.933960	41.055167
RandomInitializer	TabuOptimizer	-2.941113	33.909467
	GreedyOptimizer	-2.966013	40.920333
	ChoiceOptimizer	-3.043123	31.259833
AgeInitializer	ChoiceOptimizer	-3.047127	31.276700
SchedulingInitializer	ChoiceOptimizer	-3.066953	31.295167
RandomInitializer	SimulatedAnnealingOptimizer	-3.771730	30.632033
SchedulingInitializer	SimulatedAnnealingOptimizer	-3.827477	30.574533
AgeInitializer	SimulatedAnnealingOptimizer	-3.965197	30.622100



Genetický algoritmus

Jedinec je reprezentován stejně jako u ostatních algoritmů, tj. jedná se o seznam hodnot 0 – 7 znázorňující přiřazení dítěte do oddílu. Křížení je prováděno pomocí order crossover, tj. z jednoho rodiče zkopírujeme část hodnot na stejnou pozici a zbytek doplníme z druhého rodiče. Aby byl zachován počet hodnot, bylo třeba je učinit unikátními, proto jim bylo přiřazeno písmeno unikátní v rámci oddílu.

Byl vyřazen z hodnocení, křížení způsobovalo potomky tak málo podobné rodičům, že se jednalo prakticky o generátor náhodných řešení. Proto nebyl moc úspěšný.

Simulované žíhání

Pravděpodobně obsahuje bug, protože dopadá hůř než prostý hladový algoritmus. Bohužel se tento bug nepovedlo najít.

Tabu table

Pracuje s tabu table velikosti 100 a aspiračním kritériem přijímajícím lepší řešení. V kombinaci s inicializací řídícím pravidlem podává stabilně nejlepší výsledky.

Hladový algoritmus

Vyzkouší jedno sousední řešení a pokud je lepší, přijme jej. V kombinaci s náhodnou inicializací v benchmarku dosáhl nejlepšího řešení, ale není tak rychlý při zapadnutí do lokálního minima může dosahovat nedostatečných výsledků.

Výběr z n sousedů

Vyzkouší n sousedních řešení a vybere nejlepší z nich (pokud je celkovým zlepšením). Výrazněji naráží na problém s lokálním optimem, proto dosahuje horších výsledků než hladový algoritmus.

Hodnocení vedoucími

Kromě benchmarku byly výsledné oddíly nasdíleny vedoucím z tábora k hodnocení. Objevil se tak problém ve funkci na evaluaci kvality rozdělení. Výsledné řešení, které se zdály dobré, ve skutečnosti slabě zohledňovaly kritérium na rovnoměrné věkové rozdělelní v oddílech. Proto vznikaly oddíly, kde bylo vícero nejmladších dětí vyvažováno větším počtem nejstarších dětí. V praxi se ale nejedná o vyvážení, protože děti v rozmezí 14 – 18 let jsou všechny víceméně podobně samostatné, zatímco rozdíl mezi 6letým a 9letým dítětem je výraznější.

Možnosti rozšíření

- Aplikace je předpřipravena pro možnosti parametrizování vyhodnocovací funkce oddílů uživatelem.
- Do budoucna k ní bude doimplementováno UI.
- Aplikace počítá i s výškou dětí, tyto data ale aktuálně nemáme k dispozici.
- V datech naopak jsou LARPové role dětí, které by u výpočtu také mohly být zohledněny.