Knihovna pro obecný sekvenční a paralelní genetický algoritmus

Filip Křenek[1]

1 Úvod

Genetické algoritmy představují jednu z významných metod optimalizace inspirovanou přírodou, konkrétně evolučními procesy. Díky své schopnosti vyhledávat řešení složitých problémů, které by konvenční algoritmy řešily obtížně nebo s vysokými nároky na výpočetní výkon, jsou genetické algoritmy využívány v mnoha výzkumných oblastech především k řešení optimalizačních problémů.

Cílem této práce bylo vytvořit univerzální knihovnu pro genetické algoritmy, která by mohla být využívána pro široké spektrum optimalizačních problémů. Tato knihovna umožňuje snadné využití genetických algoritmů s možností výběru jednotlivých genetických operátorů a dalších nastavení tak, aby bylo možné snadno implementovat specifický genetický algoritmus pro řešení konkrétního problému. Knihovna rovněž umožňuje paralelní běh výpočtu fitness funkce, což je hlavní část genetického algoritmu spojujícího s konkrétním řešeným problémem a často může být velmi náročná na výpočet.

2 Gentické algoritmy

Genetické algoritmy (GA) jsou optimalizační a vyhledávací metody inspirované přírodní evolucí. Pracují s populací kandidátních řešení, která se postupně vyvíjí pomocí operací jako selekce, křížení a mutace (genetické operátory). Jedinci s vyšší kvalitou (fitness) mají větší šanci předat své vlastnosti do dalších generací, čímž se populace postupně přibližuje k optimálnímu řešení daného problému.
Jednotlivé kroky algoritmu jsou:

1. Inicializace populace: Náhodné vytvoření počáteční populace jedinců.
2. Hodnocení fitness: Výpočet fitness každého jedince.
3. Selekce rodičů: Výběr rodičů na základě jejich fitness.
4. Křížení: Kombinace genetické informace rodičů pro vytvoření potomků.
5. Mutace: Náhodné úpravy genetické informace pro udržení diverzity.
6. Vytvoření nové populace: Sestavení nové generace z potomků.
7. Kontrola ukončovací podmínky: Pokud je splněna podmínka (např. počet generací), algoritmus končí, jinak se vrací k bodu 2 (Alam et. al (2020)).

[1] student bakalářského studijního programu Informatika a výpočetní technika, specializace Informatika, e-mail: krenekf@students.zcu.cz

3 Implementace knihovny

3.1 Požadavky

Před návrhem knihovny byly analyzovány existující řešení, zejména knihovny Jenetics a JGAP. Přestože obě nabízí široké možnosti konfigurace, obsahují také nedostatky. Knihovna Jenetics je velmi složitá na používání, což může odradit méně zkušené uživatele, zatímco JGAP postrádá podporu vícevláknového zpracování, které je klíčové pro efektivní výpočty náročných fitness funkcí. Cílem vyvíjené knihovny, která se jmenuje GAL4J (Genetic Algorithm Library for Java[1]), bylo vytvořit jednodušší, přehlednější a snadno rozšiřitelné rozhraní, jež by umožnilo rychlou implementaci genetických algoritmů. Důraz byl kladen hlavně na širokou volbu genetických operátorů, které umožňují různé přístupy k řešení optimalizačních problémů. Součástí knihovny je také podpora jak sekvenčního, tak paralelního výpočtu fitness, což výrazně zvyšuje efektivitu a zkracuje dobu běhu algoritmu v případech s složitou fitness funkcí.

3.2 Struktura knihovny

Knihovna GAL4J je rozdělena do několika podadresářů podle funkční odpovědnosti. Balík `algorithm` obsahuje jádro knihovny s řízením běhu algoritmu, správou populace a výpočtem fitness. Reprezentaci řešení zajišťuje balík `chromosome` s třídami pro práci s chromozomy a podporou generických typů. Genetické operátory jsou rozděleny do balíků `selection`, `crossover` a `mutation`, které implementují výběr rodičů, křížení a mutace. Pro paralelní výpočty fitness na vícejádrových procesorech slouží balík `multithread`. Balík `utils` pak nabízí pomocné nástroje pro vizualizaci a kontrolu konfigurace.

4 Testování knihovny

Knihovna GAL4J byla důkladně otestována na dvou klasických problémech genetických algoritmů – problému batohu (knapsack problem) a problému optimalizace obchodního cestujícího (Traveling Salesman Problem). Testování bylo zaměřeno na ověření celkové funkčnosti knihovny včetně implementace všech genetických operátorů a rychlosti sekvenčního a paralelního výpočtu fitness hodnot. Pro problém batohu byla navíc provedeno srovnání výkonu GAL4J s dalšími dvěma knihovnami, Jenetics a JGAP. Výsledky v tab. 1 ukazují, že GAL4J dosahuje podobných fitness hodnot jako ostatní knihovny, ale výrazně lepších časových výsledků v řešení batohu. Díky tomu se jeví jako efektivní a lehká alternativa zejména pro úlohy, kde je klíčová rychlost výpočtu.

Knihovna	**Čas [ms]**	**Fitness**
Jenetics	427	377
JGAP	1649	380
GAL4J	98	378

Tabulka 1: Porovnání výkonu knihoven na problému batohu

Literatura

Alam, T., Qamar, S., Dixit, A. a Benaida, M., 2020. Genetic Algorithm: Reviews, Implementations, and Applications. *CoRR*.

[1]Knihovna je volně dostupná ke stažení na adrese `https://github.com/FildaKrenku/GAL4J`