Analýza citlivosti genetických algoritmů na nastavení parametrů: srovnání fixního a adaptivního přístupu

Anna Krajčírová

Ústav informatiky, Slezská univerzita v Opavě
Bezručovo nám. 13, 749 01 Opava
Email: anna.krajcirova@fpf.slu.cz

Abstrakt

Genetické algoritmy (GA) představují optimalizační metodu inspirovanou evolucí. Jsou využívány při řešení složitých nebo rozsáhlých úloh, u nichž nelze efektivně aplikovat klasické postupy. Výkonnost GA je významně ovlivněna nejen formulací fitness funkce, ale také nastavením jeho parametrů. Nevhodná volba těchto parametrů může ovlivnit rychlost konvergence i kvalitu nalezeného řešení.

Cílem tohoto článku je experimentálně porovnat fixní a adaptivní řízení parametrů GA a analyzovat jejich vliv na průběh evoluce a výslednou kvalitu řešení. Srovnávací analýza je provedena na třech typech úloh: (1) maximalizaci jednorozměrné funkce, (2) maximalizaci dvourozměrné funkce a (3) řešení problému obchodního cestujícího. Na základě experimentálních výsledků jsou porovnány rozdíly mezi oběma přístupy a jejich praktické implikace.

1 Úvod

Genetické algoritmy (GA) spadají do třídy evolučních algoritmů (EA), inspirovaných principy přirozeného výběru známými z biologie. S myšlenkou evolučních výpočtů přišel v roce 1960 Ingo Rechenberg ve své práci *Evolution Strategies*. Za průkopníka genetických algoritmů je považován John Holland, který se svými studenty v raných 70. letech položil základy této oblasti (Trinh, 2025).

Genetické algoritmy jsou široce využívány při řešení optimalizačních problémů, zejména v případech, kdy je nalezení přesného řešení výpočetně náročné nebo prakticky neproveditelné. Typickými oblastmi jejich použití jsou kombinatorické úlohy, nelineární optimalizace nebo problémy s mnoha lokálními extrémy, ve kterých mohou tradiční gradientní metody selhávat.

Jednou z klíčových vlastností genetických algoritmů je schopnost prohledávat rozsáhlý stavový prostor bez nutnosti explicitní znalosti jeho struktury. Na druhou stranu je jejich výkonnost silně ovlivněna volbou parametrů, jako jsou velikost populace, pravděpodobnost křížení a pravděpodobnost mutace. Nevhodně zvolené parametry mohou vést k předčasné konvergenci, stagnaci algoritmu nebo naopak k nedostatečnému využití nalezených řešení (Le a spol., 2023).

Tradiční přístup k návrhu genetických algoritmů využívá pevně nastavené hodnoty těchto parametrů, které jsou určeny před zahájením výpočtu. Alternativou je adaptivní řízení parametrů, kdy se jejich hodnoty mění v průběhu evoluce v závislosti na aktuálním stavu hledání. Tento přístup umožňuje lépe vyvažovat fáze explorace, tedy prohledávání dosud neprozkoumaných oblastí stavového prostoru, a exploitace, tedy intenzivního prohledávání okolí již nalezených kvalitních řešení. Adaptivní přístup tak má potenciál zvýšit robustnost algoritmu.

Cílem tohoto článku je analyzovat vliv nastavení parametrů genetického algoritmu na jeho výkonnost a porovnat přístup s fixními parametry a adaptivní variantou algoritmu. Experimenty jsou provedeny na třech typech úloh s různou mírou složitosti: jednorozměrné a dvourozměrné optimalizační úloze a problému obchodního cestujícího (TSP). Výsledky jsou hodnoceny z hlediska kvality nalezeného řešení, rychlosti konvergence a stability mezi jednotlivými běhy.

2 Genetické algoritmy

Biologickou motivací genetických algorimtů je napodobování principů biologické evoluce a přirozeného výběru při řešení optimalizačních a prohledávajících úloh. Dochází k simulaci vývoje populace jedinců, která se postupně přízpůsobuje svému prostředí stejně, jako tomu je v přírodě. Jinými slovy, populace se zlepšuje v řešení daného problému.

V biologii jsou genetické informace uchovávány v chromozomech, které jsou předávány potomkům. Z informatického hlediska lze chromozom chápat jako kód reprezentující konkrétní řešení (tedy jedince). Při tvorbě nové generace dochází ke kombinování genetických informací rodičů. Způsob, jakým vzniká nový jedinec (jeho chromozom), je určen genetickými operátory selekce, křížení a mutace. Ty určují výběr rodičů, způsob kombinace jejich genetické informace a její následnou modifikaci případnou mutací.

Klíčovým pojmem genetických algoritmů je *fitness funkce*, která určuje kvalitu daného řešení

vzhledem k řešené úloze. Každému jedinci je přiřazena číselná hodnota vyjadřující jeho zdatnost, která ovlivňuje pravděpodobnost jeho výběru pro reprodukci.

Algoritmus pracuje s populací jedinců, která se v průběhu generací vyvíjí. V každé generaci jsou preferováni jedinci s vyšší hodnotou fitness, kteří mají větší pravděpodobnost předat své vlastnosti dalším generacím (Koza, 1994).

2.1 Schéma algoritmů

Zjednodušeně můžeme princip genetického algoritmu popsat následujícím pseudokódem:

```
inicializuj populaci P
vyhodnoť fitness každého jedince v P

while (není splněna podmínka ukončení):
    vytvoř novou populaci P'

    zachovej nejlepší jedince (elitismus)

    while (|P'| < velikost populace):
        vyber rodiče p1, p2 (selekce)
        vytvoř potomka (křížení)
        aplikuj mutaci
        přidej potomka do P'

    nahraď P ← P'
    vyhodnoť fitness

vrať nejlepší nalezené řešení
```

Jako podmínku ukončení můžem zvolit dosažení maximálního počtu generací nebo nalezení dostatečně kvalitního řešení (Lingaraj, 2016; Trinh, 2025).

2.2 Reprezentace řešení

Způsob reprezentace jedince závisí na typu řešeného problému, například:

- binární reprezentace, kde je řešení kódováno jako řetězec bitů, což je klasický přístup využívaný v raných genetických algoritmech,
- pro spojité optimalizační úlohy jsou jedinci reprezentováni jako vektory reálných čísel,
- pro kombinatorické úlohy, jako je problém obchodního cestujícího, se používá permutační reprezentace, kde chromozom představuje pořadí navštívených prvků.

Volba vhodné reprezentace má zásadní vliv na efektivitu genetického algoritmu, protože ovlivňuje návrh genetických operátorů i schopnost algoritmu prohledávat stavový prostor (Le a spol., 2023).

2.3 Genetické operátory

Prohledávání stavového prostoru genetickými algoritmy probíhá pomocí genetických operátorů - selekce, křížení a mutace.

2.3.1 Operátor selekce

Operátor selekce určuje, kteří jedinci budou vybráni jako rodiče pro vytvoření nové generace. Cílem je zvýhodnit kvalitnější jedince, tedy ty s co nejlepší fitness, aniž by došlo ke ztrátě diverzity populace.

V této práci byla použita *turnajová selekce*, při které je náhodně vybrána skupina jedinců a z ní je zvolen nejlepší, přičemž se proces opakuje pro výběr každého rodiče. Výhodou této metody je její jednoduchost a možnost řídit selekční tlak pomocí velikosti turnaje (Miller a Goldberg, 1995).

Další často používanou metodou je *ruletová selekce* (*Roulette Wheel Selection*), kde je pravděpodobnost výběru jedince úměrná jeho fitness, respektive poměru jeho fitness vůči celkovému součtu fitness všech jedinců. Alternativou je také *rank-based selekce*, která přiděluje pravděpodobnosti na základě pořadí jedinců, čímž omezuje vliv extrémních hodnot fitness (Lingaraj, 2016; BAJPAI a Kumar, 2010).

2.3.2 Operátor křížení

Křížení (*crossover*) slouží ke kombinaci genetické informace dvou rodičů za účelem vytvoření potomka. Tento operátor umožňuje prozkoumávat nové oblasti stavového prostoru.

Pro spojité úlohy byl použit *aritmetický crossover*, při kterém je potomek vytvořen jako lineární kombinace rodičů:

$$potomek = \alpha p_1 + (1 - \alpha) p_2 \tag{1}$$

Pro problém obchodního cestujícího byl použit operátor *Order Crossover* (OX), který zachovává relativní pořadí prvků a produkuje validní permutace.

Mezi další běžně používané operátory patří například jednobodové nebo dvoubodové křížení, typické pro binární reprezentaci, nebo *uniformní crossover*, který kombinuje geny rodičů na základě náhodné masky. U permutačních úloh se často využívají operátory jako PMX (*Partially Mapped Crossover*), které zachovávají validitu permutace (Lingaraj, 2016).

2.3.3 Operátor mutace

Mutace slouží k zavedení náhodných změn do jedinců a zajišťuje diverzitu populace. Pomáhá algoritmu uniknout z lokálních optim.

Pro spojité úlohy byla mutace realizována přidáním gaussovského šumu k jednotlivým proměnným. U problému obchodního cestujícího

byla použita *swap mutace*, při které dojde k prohození dvou náhodně vybraných prvků v permutaci.

Mezi další často používané mutační operátory patří například *uniformní mutace*, která nahrazuje hodnotu genu náhodně zvolenou hodnotou z definičního oboru, nebo *neuniformní mutace*, jejíž velikost změny se v průběhu evoluce postupně snižuje. U permutačních úloh se kromě swap mutace využívá také *inverzní mutace*, která obrací pořadí prvků v části chromozomu (Lingaraj, 2016).

2.4 Parametry GA

Výkonnost genetického algoritmu závisí na několika základních parametrech:

- velikost populace,
- pravděpodobnost křížení P_c,
- pravděpodobnost mutace P_m,
- počet generací.

Tyto parametry ovlivňují rovnováhu mezi explorací (prohledávání nových oblastí) a exploitací (vylepšování existujících řešení) (Trinh, 2025).

3 Nastavení experimentů

Za cíl našich experimentů jsme si stanovili analýzu vlivu nastavení parametrů genetického algoritmu na kvalitu nalezeného řešení, rychlost konvergence a stabilitu výsledků, přičemž jsme se zabývali hlavně porovnáním vlivu pravděpodobnosti mutace P_m. Mimo zkoumání vlivu parametrů jsme porovnávali také přístup s fixními parametry a adaptivním řízením pravděpodobnosti mutace.

3.1 Testovací úlohy

Pro experimentální analýzu jsme zvolili tři typy optimalizačních úloh, reprezentující různé charakteristiky stavového prostoru a různou obtížnost.

3.1.1 Jednorozměrná optimalizační úloha

Jako jednorozměrnou optimalizační úlohu jsme zvolili funkci:

$$y = sin(x) \cdot x \tag{2}$$

s definičním oborem $\langle 0; 20 \rangle$. Tato úloha slouží jako základní testovací scénář s relativně jednoduchou strukturou fitness funkce.

3.1.2 Dvourozměrná optimalizační úloha

Dvourozměrná úloha byla definována funkcí:

$$y = sin(x) \cdot cos(y) + 0,1x + 0,1y \tag{3}$$

na definičním oboru $(x, y) \in \langle 0; 20 \rangle \times \langle 0; 20 \rangle$.

Tato funkce kombinuje periodickou složku s lineárním trendem, čímž vytváří komplexní fitness krajinu s více lokálními extrémy. Úloha je vhodná pro analýzu schopnosti algoritmu překonávat lokální optima.

3.1.3 Problém obchodního cestujícího (TSP)

Třetí testovací úlohou byl problém obchodního cestujícího. TSP je klasický optimalizační problém, jehož cílem je najít nejkratší možnou cestu, která prochází všemi zadanými městy (body), přičemž každé město navštíví právě jednou a vrátí se do výchozího bodu. Jedná se o tzv. NP-težký problém, což znamená, že s počtem měst roste počet možných kombinací exponenciálně. Pro pouhých $n = 10$ měst je počet možných kombinací přes $18\,000$. Řešení této úlohy hrubou silou pro vetší počet měst tedy nepřipadá v úváhu, díky čemuž je vhodná pro testování genetických algoritmů (Cook a spol., 1997).

Bylo uvažováno $n = 30$ měst, což je přibližně $4,42 \times 10^{30}$, jejichž souřadnice byly generovány náhodně v rovině. Pro zajištění reprodukovatelnosti byly souřadnice měst generovány pouze jednou a následně použity ve všech bězích experimentu (Matai a spol., 2010).

Fitness funkce byla definována jako převrácená hodnota celkové délky uzavřené cesty d:

$$f = \frac{1}{d} \tag{4}$$

3.2 Nastavení genetických algoritmů

Pro všechny experimenty byla použita následující konfigurace genetického algoritmu:

- velikost populace: 80 a 180 jedinců,
- počet generací: 30 pro 1D funkci, 150 pro 2D funkci, 400 pro TSP,
- pravděpodobnost křížení: $P_c = 0,8$ a $P_c = 0,5$,
- pravděpodobnost mutace: $P_m = 0,01$, $P_m = 0,05$ a $P_m = 0,1$
- selekce: turnajová selekce (velikost turnaje 3),
- elitismus: nejlepší jedinec byl vždy zachován do další generace.

Pro spojité úlohy byla mutace realizována přidáním gaussovského šumu, zatímco u TSP byla použita swap mutace. Pro křížení byl u spojitých úloh použit aritmetický operátor, zatímco u TSP byl aplikován operátor Order Crossover (OX).

Data pro každou konfiguraci algoritmu byla vyhodnocována na základě 30 běhů.

3.3 Adaptivní řízení parametrů

V adaptivní variantě genetického algoritmu byla dynamicky upravována pravděpodobnost mutace P_m v závislosti na průběhu optimalizace. Cílem bylo reagovat na stagnaci algoritmu a současně podpořit jemnější doladění řešení v případě úspěšného vývoje.

Pokud v aktuální generaci došlo ke zlepšení nejlepší dosažené hodnoty fitness oproti dosud nalezenému maximu, byla pravděpodobnost mutace snížena vynásobením koeficientem 0,9. Tím docházelo k postupnému omezení náhodnosti a podpoře exploatace okolí kvalitního řešení.

Naopak pokud během 10 po sobě jdoucích generací nedošlo ke zlepšení hodnoty fitness (stav stagnace), byla pravděpodobnost mutace zvýšena. U jednorozměrné a dvourozměrné optimalizační úlohy byl použit koeficient 1,5, který umožňuje rychlé zvýšení diverzity populace a intenzivnější exploraci nových oblastí stavového prostoru.

U problému obchodního cestujícího byl zvolen nižší koeficient 1,1. Důvodem je diskrétní charakter problému a delší evoluce, kde příliš agresivní navyšování mutace může vést k narušení již nalezených kvalitních částí řešení a ke ztrátě konvergence. Jemnější adaptace proto umožňuje postupnější obnovu diverzity bez výrazného narušení dosavadního vývoje populace.

Pravděpodobnost mutace byla během celého běhu omezena předem definovanými dolními a horními mezemi, aby nedocházelo k její přílišné redukci nebo naopak k nadměrně náhodnému chování algoritmu.

4 Výsledky experimentů

V této kapitole zhodnotíme data získaná z experimentů. Pro každou úlohu zde uvádíme tabulku se základní statistikou a čtyři grafy, které hodnotí:

- rychlost konvergence pro různá P_m,
- rychlost konvergence pro stejné P_m a jiné P_c a velikost populace (pop_size),
- variabilitu generace, ve které bylo nalezeno nejlepší řešení ($best\_gen$) pro různá P_m,
- variabilitu $best\_gen$ pro stejné P_m a jiné P_c a pop_size.

Pro zkrácení zápisu jednotlivých algoritmů jsme zavedli následující značení: Problém-varianta-pop_size x Pc x Pm. Označení GA, který řeší 1D úlohu, má fixně nastavené parametry, $pop\_size = 80$, $P_c = 0,8$ a $P_m = 0,01$ tak bude vypadat takto: $1Dfix80x0,8x0,01$.

4.1 Jednorozměrná optimalizační úloha

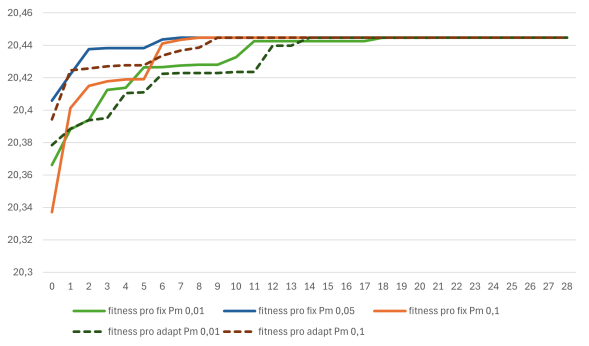
Tato podkapitola se zaměřuje na vyhodnocení genetického algoritmu při řešení jednorozměrné optimalizační úlohy. Porovnávány jsou jak varianty s fixními parametry, tak adaptivní přístup. Výsledky jsou shrnuty v Tabulce 1 a doplněny grafy průběhu konvergence a variability.

Tab. 1: Porovnání výsledků různých nastavení parametrů GA pro 1D úlohu.

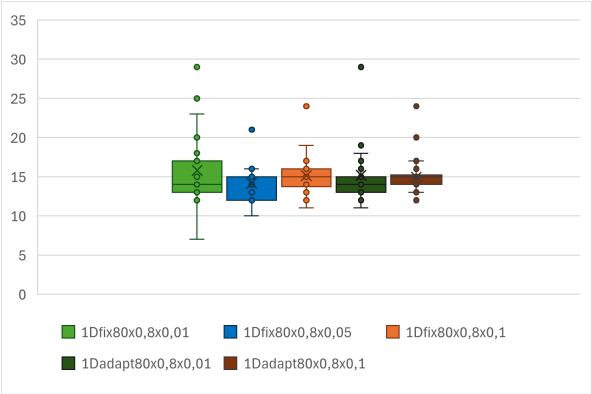
Alg.	Mean fit	Std fit	Med gen	Std gen
fix80x0,8x0,01	20,4447831	5,1E-06	14	4,88
fix80x0,8x0,05	20,4447826	7,9E-06	15	2,02
fix80x0,8x0,1	20,4447841	7,2E-15	15	2,42
adapt80x0,8x0,01	20,4447841	7,2E-15	14	4,12
adapt80x0,8x0,1	20,4447841	7,2E-15	15	2,37
fix180x0,8x0,01	20,4447841	7,2E-15	14	1,19
fix80x0,5x0,01	20,4447840	2,8E-07	15	3,02

Výsledky ukazují, že všechny testované konfigurace genetického algoritmu dosahují prakticky identické hodnoty fitness ($\approx 20,44478$), což odpovídá globálnímu maximu dané funkce. Rozdíly mezi jednotlivými nastaveními se proto projevují především v rychlosti konvergence a stabilitě, i když jen nevýrazně v porovnání se složitějšími úlohami.

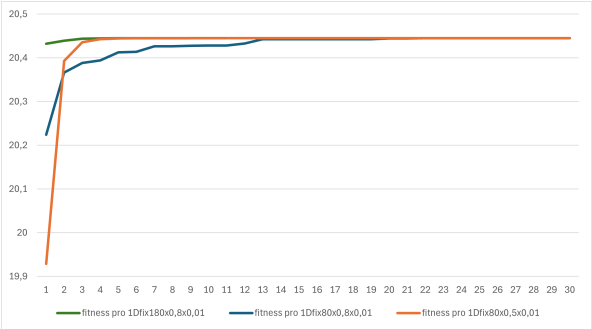
Z Tabulky 1 a Obrázku 1 můžeme vyčíst, ž vliv mutace je v této úloze minimální, což je dáno jednoduchostí optimalizačního problému. Ani adaptivní řízení parametrů zde nepřináší významné zlepšení oproti fixnímu nastavení.



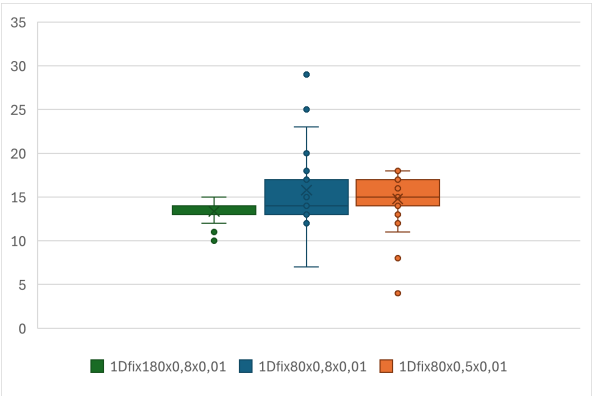
Obr. 1: Porovnání rychlosti konvergence a kvality řešení pro GA 1Dfix80x0,8xPm a 1Dadapt80x0,8xPm.



Obr. 2: Porovnání variance *best_gen* pro GA 1Dfix80x0,8xPm a 1Dadapt80x0,8xPm.



Obr. 3: Porovnání rychlosti konvergence a kvality řešení pro GA $P_m = 0,01$.



Obr. 4: Porovnání variance *best_gen* pro GA $P_m = 0,01$.

Z hlediska konvergence dosahují nejlepší výsledky konfigurace s vyšší populací (1Dfix180x0,8x0,01), která vykazuje nejnižší variabilitu (*Std. best_gen = 1,19*) při zachování nízkého mediánu generace konvergence. Naopak nižší pravděpodobnost křížení (1Dfix80x0,5x0,01) vede k pomalejší a méně stabilní konvergenci, jak je vidět na Obrázcích 3 a 4.

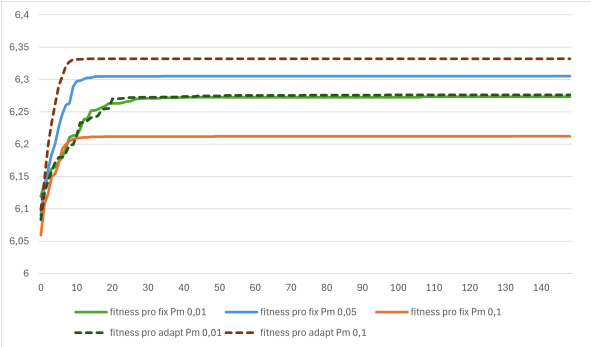
4.2 Dvourozměrná optimalizační úloha

V této podkapitole jsou prezentovány výsledky pro dvourozměrnou optimalizační úlohu, která představuje složitější fitness krajinu s větším počtem lokálních extrémů. Přehled dosažených výsledků je uveden v Tabulce 2 a doplněn grafy.

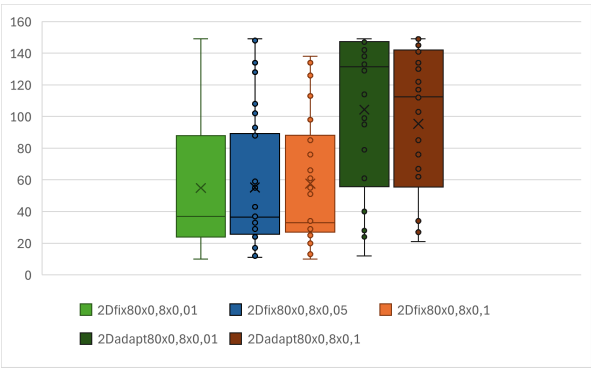
Tab. 2: Porovnání výsledků různých nastavení parametrů GA pro 2D úlohu.

Alg.	Mean fit	Std fit	Med gen	Std gen
fix80x0,8x0,01	6,2733	0,3260	37	42,41
fix80x0,8x0,05	6,3052	0,2894	36,5	42,32
fix80x0,8x0,1	6,2121	0,2331	33	42,08
adapt80x0,8x0,01	6,2765	0,2375	131,5	49,89
adapt80x0,8x0,1	6,3323	0,2876	112,5	45,62
fix180x0,8x0,01	6,4568	0,2523	30	27,78
fix80x0,5x0,01	6,2491	0,2332	36,5	38,02

Na rozdíl od 1D úlohy se zde již výrazněji projevují rozdíly mezi jednotlivými konfiguracemi. Jak je vidět z Tabulky 2 a na Obrázku 7, nejlepší hodnoty fitness dosahuje konfigurace s větší populací (2Dfix180x0,8x0,01), což potvrzuje význam diverzity populace u složitějších problémů.



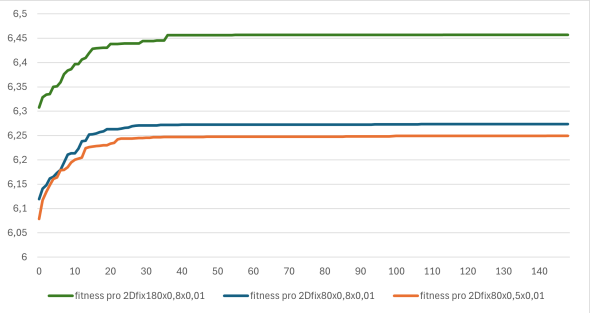
Obr. 5: Porovnání rychlosti konvergence a kvality řešení pro GA 2Dfix80x0,8xPm a 1Dadapt80x0,8xPm.



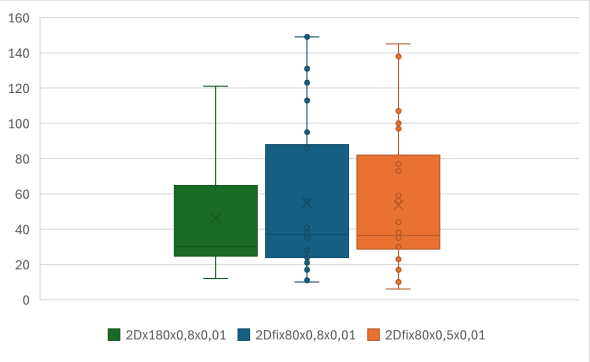
Obr. 6: Porovnání variance *best_gen* pro GA 2Dfix80x0,8xPm a 2Dadapt80x0,8xPm.

Vyšší pravděpodobnost mutace ($P_m = 0,1$) vede ke zhoršení kvality řešení, což naznačuje přílišnou míru náhodnosti. Naopak střední hodnota mutace

($P_m = 0,05$) poskytuje vyvážený kompromis mezi explorací a exploitací. Tato konfigurace dosáhla také největší stability, jak můžeme názorně vidět na Obrázku 8



Obr. 7: Porovnání rychlosti konvergence a kvality řešení pro GA $P_m = 0,01$.



Obr. 8: Porovnání variance *best_gen* pro GA $P_m = 0,01$.

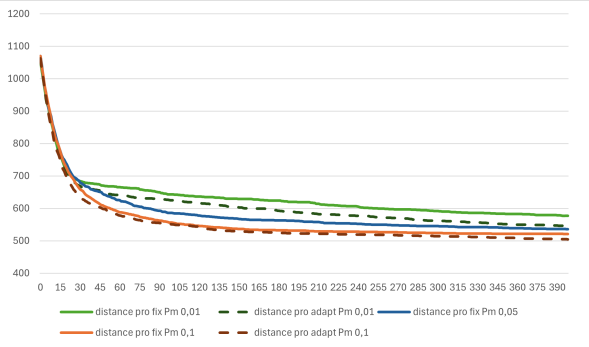
Adaptivní genetický algoritmus dosahuje srovnatelných nebo mírně lepších hodnot fitness (např. 2Dadapt80x0,8x0,1), avšak za cenu výrazně pomalejší konvergence (medián až 131,5 generace). Tento výsledek ukazuje, že adaptivní řízení podporuje důkladnější prohledávání prostoru, ale může zpomalit nalezení řešení.

4.3 Problém obchodního cestujícího

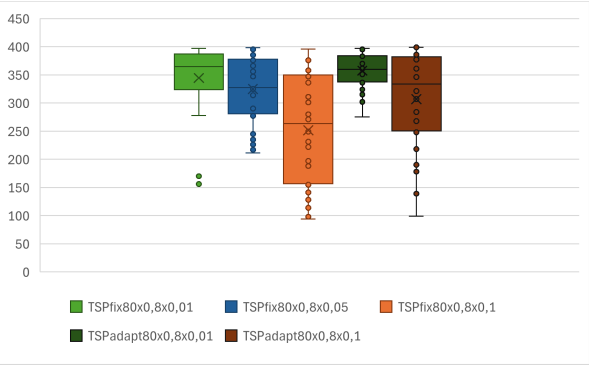
Tato podkapitola se zabývá aplikací genetického algoritmu na problém obchodního cestujícího, který představuje diskrétní kombinatorickou úlohu. Výsledky jsou shrnuty v Tabulce 3 a opět doplněny grafy. Pro lepší interpretaci tentokrát místo hodnoty fitness funkce používáme jako ukazatel kvality nalezeného řešení celkovou délku cesty *distance*.

Tab. 3: Porovnání výsledků různých nastavení parametrů GA pro TSP.

Alg.	Mean dis	Std dis	Med gen	Std gen
fix80x0,8x0,01	576,69	45,47	365	59,33
fix80x0,8x0,05	535,90	38,49	327,5	57,35
fix80x0,8x0,1	521,00	28,23	263,5	99,01
adapt80x0,8x0,01	546,16	47,10	360	31,02
adapt80x0,8x0,1	504,60	27,92	333,5	85,12
fix180x0,8x0,01	510,06	36,55	322	93,08
fix80x0,5x0,01	613,19	40,44	349	44,88



Obr. 9: Porovnání rychlosti konvergence a kvality řešení pro GA různé P_m.



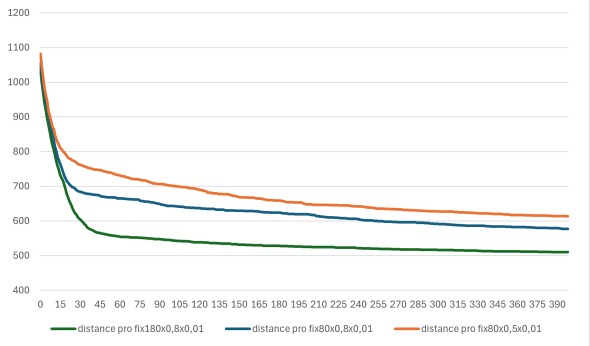
Obr. 10: Porovnání variance *best_gen* pro GA TSPfix80x0,8xPm a TSPadapt80x0,8xPm.

U problému obchodního cestujícího jsou rozdíly mezi konfiguracemi nejvýraznější. Nejlepších výsledků dosahuje adaptivní genetický algoritmus (TSPadapt80x0,8x0,1), který minimalizuje délku trasy na průměrnou hodnotu 504,60, jak je vidět v Tabulce 3 a Obrázku 9. Zajímavé je, že TSPadapt80x0,8x0,1 oproti TSPfix80x0,8x0,1 vykazuje menší zlepšení, než TSPadapt80x0,8x0,01 proti TSPfix80x0,8x0,01. U 2D úlohy jsme přitom mohli pozorovat opačný jev.

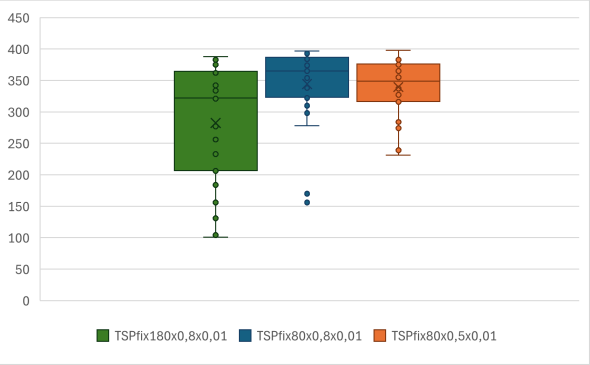
Vyšší mutace ($P_m = 0,1$) se zde ukazuje jako výhodná, protože pomáhá překonávat lokální optima. Naopak nízká mutace vede k horším výsledkům (např. právě TSPfix80x0,8x0,01).

Zvýšení velikosti populace (TSPfix180x0,8x0,01) rovněž vede ke zlepšení kvality řešení, avšak za cenu vyšší variability konvergence. Snížení pravděpodobnosti křížení (TSPfix80x0,5x0,01) má ne-

gativní dopad na kvalitu řešení i rychlost konvergence, jak je možné vyčíst z Tabulky 3 a Obrázků 11 a 12.



Obr. 11: Porovnání rychlosti konvergence a kvality řešení pro GA $P_m = 0,01$.



Obr. 12: Porovnání variance *best_gen* pro GA $P_m = 0,01$.

Adaptivní přístup se v této úloze ukazuje jako nejefektivnější, protože umožňuje dynamicky reagovat na stagnaci a lépe vyvažovat exploraci a exploitaci.

5 Závěr

Výsledky experimentů potvrzují, že vliv nastavení parametrů genetického algoritmu je silně závislý na složitosti řešené úlohy. Zatímco u jednoduchých optimalizačních problémů je možné dosáhnout kvalitních výsledků i s pevně nastavenými parametry, u složitějších úloh se význam vhodného nastavení výrazně zvyšuje.

Pozorovaný kompromis mezi rychlostí konvergence a kvalitou výsledného řešení odpovídá známému konsenzu mezi explorací a exploitací, který je v literatuře považován za jeden z klíčových principů adaptivního řízení parametrů (Eiben a Smith, 2003; Eiben a spol., 2007).

V případě jednorozměrné úlohy dosahovaly všechny testované konfigurace prakticky totožné hodnoty fitness odpovídající globálnímu maximu. Rozdíly mezi jednotlivými nastaveními se projevily především v rychlosti konvergence a její stabilitě, i když ne příliš výrazně. Vliv pravděpodobnosti mutace byl minimální a adaptivní řízení parametrů nepřineslo významné zlepšení. To odpovídá charakteru úlohy s jednoduchou fitness krajinou bez výrazných lokálních extrémů.

U dvourozměrné úlohy se již projevily rozdíly mezi jednotlivými konfiguracemi. Bylo pozorováno, že střední hodnota mutace představuje vhodný kompromis mezi explorací a exploitací, zatímco příliš vysoká mutace vede k degradaci kvality řešení. Adaptivní přístup zde umožnil dosahovat srovnatelných nebo mírně lepších výsledků, avšak za cenu pomalejší konvergence. Tento jev lze interpretovat jako důsledek intenzivnějšího prohledávání stavového prostoru.

Nejvýraznější rozdíly byly pozorovány u problému obchodního cestujícího. V tomto případě se adaptivní genetický algoritmus ukázal jako nejefektivnější přístup, který dosahoval nejlepších hodnot cílové funkce. Vyšší pravděpodobnost mutace se ukázala jako klíčová pro překonávání lokálních optim, zatímco nízká mutace vedla k předčasné konvergenci. Adaptivní řízení umožnilo dynamicky reagovat na stagnaci algoritmu a tím zlepšit kvalitu výsledků.

Napříč všemi úlohami se také potvrdil vliv velikosti populace. Větší populace obecně vedla k lepším výsledkům a stabilnější konvergenci, avšak za cenu vyšší výpočetní náročnosti. Naopak snížení pravděpodobnosti křížení mělo spíše negativní dopad, zejména u složitějších úloh.

Zajímavým zjištěním byl rozdílný vliv adaptivního řízení parametrů v závislosti na počáteční hodnotě mutace. U problému obchodního cestujícího byl přínos adaptivního přístupu výraznější při nízké počáteční hodnotě mutace, zatímco při vyšší mutaci byl rozdíl oproti fixnímu nastavení menší. Tento jev lze vysvětlit tím, že nízká mutace vede k předčasné konvergenci, kterou adaptivní mechanismus dokáže překonat zvýšením diverzity populace.

Naopak u spojité dvourozměrné úlohy byl pozorován opačný trend, kdy adaptivní přístup přinášel větší zlepšení při vyšší počáteční hodnotě mutace. V tomto případě adaptivní mechanismus umožňuje postupné snižování mutace v okolí maxima funkce a tím přejít od ”prohledávání”k ”ladění”. Výsledky tak ukazují, že efektivita adaptivního řízení parametrů je úzce spojena s charakterem řešené úlohy.

Celkově lze konstatovat, že adaptivní řízení parametrů je nejpřínosnější u komplexních optimalizačních problémů, kde pomáhá udržovat rovnováhu mezi explorací a exploitací. U jednodušších úloh však jeho přínos není významný a fixní nastavení parametrů je dostatečné a výpočetně úspornější. Tento závěr je v souladu s dřívějšími přehledovými pracemi, které uvádějí, že adaptivní mechanismy přinášejí největší výhody zejména u rozsáhlých a multimodálních optimalizačních úloh (Eiben a Smith, 2003; Eiben a spol., 2007)

Budoucí práce by se mohla zaměřit na komplexnější analýzu parametrů, analýzu citlivosti samotného adaptačního mechanismu (například vlivu velikosti kroku při změně pravděpodobnosti mutace), případně na vliv zvolených genetických operátorů. Zajímavým směrem je také testování na širší škále problémů a použití statistických testů pro detailnější vyhodnocení výsledků.

Reference

BAJPAI, P. a Kumar, M. (2010). Genetic algorithm - an approach to solve global optimization problems. *Indian Journal of Computer Science and Engineering*, 1:199–206.

Cook, W. J., Cunningham, W. H., Pulleyblank, W. R., Schrijver, A. a Sayer, P. (1997). *Combinatorial optimization*. John Wiley Sons Incorporated.

Eiben, A., Michalewicz, Z., Schoenauer, M. a Smith, J. (2007). Parameter Control in Evolutionary Algorithms. Lobo, F. G., Lima, C. F. a Michalewicz, Z. (eds.), V *Parameter Setting in Evolutionary Algorithms*, vol. 54 z *Studies in Computational Intelligence*, s. 19–46. Springer Verlag.

Eiben, A. E. a Smith, J. E. (2003). *Introduction to Evolutionary Computing*. Springer.

Koza, J. (1994). Genetic programming as a means for programming computers by natural selection. *Statistics and Computing*, 4(2).

Le, K., Wei, T. a Tagalogon, J. (2023). Genetic algorithm and application. *Genetic algorithm and Application*.

Lingaraj, H. (2016). A study on genetic algorithm and its applications. *International Journal of Computer Sciences and Engineering*, 4:139–143.

Matai, R., Singh, S. a Mittal, M. (2010). *Traveling Salesman Problem: an Overview of Applications, Formulations, and Solution Approaches*.

Miller, B. a Goldberg, D. E. (1995). *Genetic algorithms, tournament selection and the effects of noise*.

Trinh, M. V. T. K. (2025). Research on genetic algorithms: Concepts, models, and applications. *International Journal of Education and Social Science Research (IJESSR)*, 8:354–373.