Vysoká škola ekonomická v Praze

Fakulta informatiky a statistiky

Teorie samoorganizace a optimalizační algoritmy založené na swarm intelligence

Bakalářská práce

Ve studijním programu Aplikovaná informatika

Autor: David Říha

Vedoucí bakalářské práce: Ing. Ludmila Jiříčková

Praha, duben 2018

Prohlášení

Prohlašuji, že jsem bakalářskou práci Teorie samoorganizace a optimalizační algoritmy založené na swarm intelligence vypracoval samostatně za použití v práci uvedených pramenů a literatury.

V Praze dne ……..

David Říha

Poděkování

Chtěl bych poděkovat své vedoucí bakalářské práce Ing. Ludmile Jiřičkové za odborné vedení, za pomoc a rady při zpracování této práce a její trpělivost a ochotu poradit i v době posledního stadia paniky.

Abstrakt

Bakalářská práce se zabývá problematikou samoorganizace a jejího využití v optimalizačních algoritmech. Pro komplexnější pohled na věc, práce krátce pojednává o teorii chaosu, která je s teorií samoorganizace velmi úzce spojena. V krátkosti také popisuje organizaci, kterou demonstruje na modelech organizace. V druhé části se práce zaměřuje na optimalizační algoritmy založené na inteligenci hejna a jejich využití. Pro tyto algoritmy je důležité projevování samoorganizace. V praktické části se práce zabývá právě optimalizačními algoritmy a zaměřuje se na Ant Colony Optimization. V této části je proveden výzkum vlivu parametrů α a β na optimalizačním algoritmu Ant Colony System. Výsledky jsou podložené výstupem z programu.

Klíčová slova

samoorganizace, teorie chaosu, inteligence hejna, optimalizační algoritmy, optimalizace mravenčí kolonií

Abstract

This bachelor thesis follows up the problematics of self-organization and its utilization in optimization algorithms. For a more complex point of view, this thesis shortly outlines theory of chaos, which is connected very closely to self-organization. After theory of chaos, it takes a short look at organization and demonstrates it on models of organization. In the second part of the theoretical part thesis focuses on optimization algorithms based on swarm intelligence and its utilization. The main part of these algorithms is manifestation of self-organization. In practical part thesis takes interest in optimization of algorithms especially in Ant Colony Optimization. In this part, research takes place and it is focused on the impact of α and β parameters on optimization algorithms of Ant Colony System. The results are underlined by program output.

Keywords

self-organization, chaos theory, swarm intelligence, optimization algorithms, ant colony optimization

Obsah

Úvod ... 8
Cíle práce ... 9
1. Teorie chaosu ... 10
1.1. Historie ... 10
2. Organizace ... 12
2.1. Mechanistický model organizace ... 12
2.2. Organický model organizace ... 15
3. Samoorganizace ... 18
3.1. Historie ... 18
3.2. Termodynamický paradox ... 19
3.3. Dva základní příklady samoorganizačních systémů ... 20
3.3.1. Magnetizace ... 20
3.3.2. Bénardova nestabilita ... 21
3.4. Nelinearita a zpětná vazba ... 21
3.5. Bifurkace, Symmetry breaking ... 22
3.6. Samoorganizace vs. Emergence ... 24
3.6.1. Emergence ... 24
3.6.2. Charakteristiky emergence ... 24
3.6.3. Samoorganizace ... 25
3.6.4. Charakteristiky samoorganizace ... 25
3.6.5. Porovnání emergence a samoorganizace ... 26
3.6.5.1. Podobnosti a rozdíly ... 26
3.6.5.2. Samoorganizace bez emergence ... 26
3.6.5.3. Emergence bez samoorganizace ... 27
3.6.5.4. Kombinace samoorganizace a emergence ... 27
4. Swarm intelligence optimization ... 28
4.1. Genetic Algorithms (GA) ... 28
4.2. Artificial Bee Colony (ABC) ... 29

4.3. Particle Swarm Optimization (PSO) 32
4.4. Cuckoo Search Algorithm (CSA) 33
4.5. Glowworm Swarm Optimization (GSO) 35
4.6. Ant Colony Optimization (ACO) 37
5. Metodická část 40
5.1. Problém obchodního cestujícího 40
5.2. ACO software 40
5.3. Popis programu 41
5.4. Použití programu 41
5.5. Modifikace Ant Colony System 43
6. Praktická část 44
6.1. Vliv parametrů α a β na výkonnost algoritmu 44
6.2. Výchozí nastavení aplikace 44
6.3. Kombinace parametrů α a β 45
6.4. Vyhodnocení vlivu parametrů α a β 54
Závěr 56
Seznam literatury 58
Seznam obrázků 60
Slovníček 62

Úvod

Samoorganizace, také známá jako spontánní řád, je proces probíhající mezi jednotlivými entitami, kdy z neuspořádaného systému vzniká řád. Ať se jedná o miniaturní atomy, živé bytosti, nebo mechanismy ekonomie a informačních technologií, tak se v nich samoorganizace projevuje v jedné nebo druhé formě.

Samoorganizace se dá jednoduše popsat jako uspořádání jednotlivců, v rámci společnosti osob, které nejsou nikým vedeny. Chybí osoba, která by zajišťovala řád ve skupině. Skupina osob se nachází bez vedoucí osoby, což by mělo mít za následek rozpad systému kvůli neorganizovanosti celku. V tu chvíli ale přichází na scénu samoorganizace. Což znamená, že jednotlivé komponenty skupiny spolu začnou spolupracovat a organizovat se sami navzájem. Rozdělí si mezi sebou úkoly a sjednotí se za účelem splnění společného cíle.

V 1. kapitole práce bude představena teorie chaosu, jelikož neuspořádaný systém je hlavním předpokladem pro vznik samoorganizace. Zde si vysvětlíme, co znamená pojem teorie chaosu, a krátce si nastíníme historický vývoj.

Ve 2. kapitole se podíváme, co je to organizace. Nastíníme si, co pojem organizace vlastně znamená a získané poznatky demonstrujeme na dvou pohledech, na mechanistickém a organickém modelu organizace.

Ve 3. kapitole si vysvětlíme, co znamená samoorganizace. Krátce se podíváme na historický vývoj teorie samoorganizace. Ukážeme si dva základní příklady samoorganizačních systémů. Zjistíme, jaký vliv má zpětná vazba a popíšeme si rozdíly mezi samoorganizací a emergencí na základě jejich rozdílných charakteristik.

V 4. kapitole se zaměříme na optimalizační algoritmy, které jsou inspirovány přírodními systémy. Popíšeme si například algoritmy fungující na principu mraveniště, parazitním chování kukaček, včelí kolonie a další.

V 5. kapitole si položíme základy pro praktickou část. Povíme si něco o problému obchodního cestujícího a využitém programu, s kterým dále pracuji, a získáme zde přehled o jednotlivých parametrech, které jdou v aplikaci nastavit.

V 6. kapitole proběhne můj vlastní výzkum, kdy porovnávám změnu hodnot parametrů α a β. Zkoumám vliv těchto parametrů na výkonnost algoritmu.

V poslední části si poznatky shrneme v závěru.

Cíle práce

Cíle práce jsou:

- Vysvětlení pojmu samoorganizace a jejího využití v optimalizačních algoritmech
- Uvedení příkladů optimalizačních algoritmů a jejich využití
- Zkoumání vlivu parametrů α a β v optimalizačním algoritmu Ant Colony System

Přínosem práce je porovnávání výstupu z programu, kde je implementovaný algoritmus Ant Colony Systém, za účelem prozkoumání síly vlivu těchto parametrů.

1. Teorie chaosu

Teorie chaosu, jako taková, je široký pojem, který zaštiťuje mnohé pohledy rozdílných vědních oborů na chování nelineárních systémů charakterizovaných jevem známým jako deterministický chaos. Zajímavé je, že systémy vykazující deterministický chaos jsou v rozporu s klasickým chápáním chaosu. Klasické pojetí slova chaos v matematice a fyzice je bráno jako zcela neuspořádaný. Kdežto systémy vykazující deterministický chaos, jsou složitě uspořádané. [2]

Termín „chaos“ je používán jako prostředek k popsaní nelineárních systémů, které mají tendence chovat se chaoticky, až nahodile. Čímž se liší od tradičních (lineárních) systémů. Samotné slovo *náhoda* má původ ve francouzském idiomu, který popisoval z pohledu jezdce nepředvídatelný pohyb koně. Pro jezdce se pohyb zdál takřka nesmyslný, kdežto kůň pouze reagoval na nerovnosti povrchu a přizpůsobil tomu svůj pohyb. [1, 2]

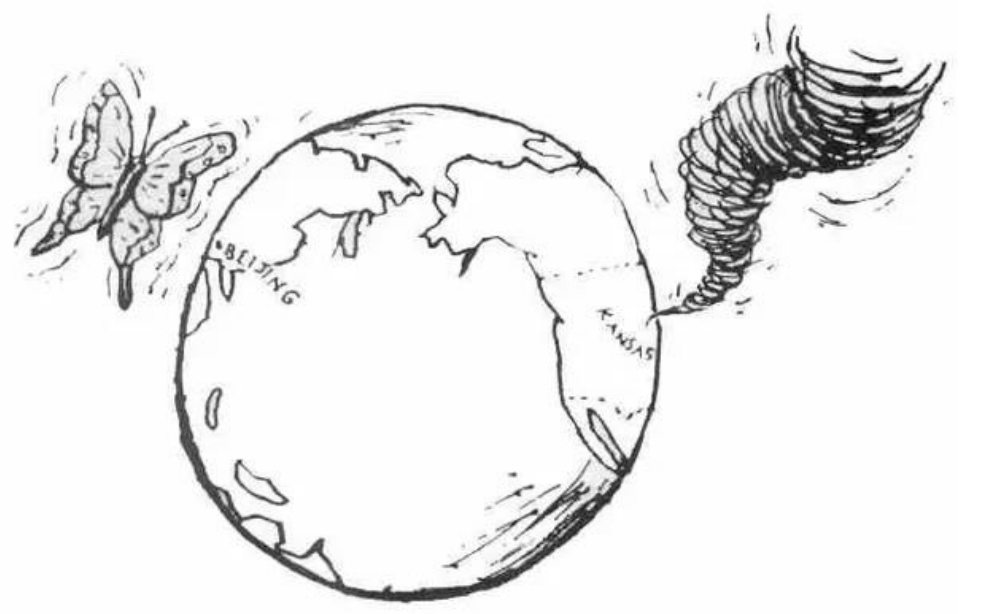
Pokud bychom chtěli získat větší přehled, o tom, jak tyto systémy fungují, a popsat jejich funkčnost, je nesmyslné studovat jednotlivé části nezávisle na celém systému. Všechny části systému spolupracují a jsou navzájem propojené vazbami, které jsou důležité pro fungování celku. Tudíž bez zohlednění celého systému a vzájemných interakcí je nemožné přiblížit se hlubšímu pochopení. Vezměme si jako příklad mraveniště. Pokud bychom začali studovat chování jednotlivých mravenců v mraveništi, nezjistili bychom žádné informace o tom, jak mraveniště funguje jako celek. Mraveniště funguje jako jeden ekosystém, kdy mravenci nejsou pouze jedinci, kteří se snaží přežít, ale jsou integrování do tohoto systému. Díky komunikaci a jejich spolupráci zajišťují přežití a úspěch celého mraveniště. [2,3]

1.1. Historie

Počátky teorie chaosu lze vystopovat až k druhé polovině 19. století, kdy studie provedené Jamesem Clerkem Maxwellem o mikroskopických nahodilostech molekul v plynu se dají považovat za první studie teorie chaosu. [1]

Na přelomu 19. a 20. století se Henri Poincaré zabýval ve svých studií tzv. problému tří těles, kde objevil citlivou závislost vývoje systému na počátečních podmínkách, tz. malá změna se v nelineárním deterministickém systému může mít později za následek velký rozdíl v konečném důsledku. Později vyslovil myšlenku, že podobný jev může mít běžný výskyt například v meteorologii. [1]

Na myšlenku Henri Poincarého navázal Edward Lorenz, který tomuto jevu dal název Motýlí efekt neboli efekt motýlího křídla, který zaobaluje výše zmíněnou závislost vývoje systému. Ve své přednáškách Lorenz tento jev jednoduše vysvětloval tímto způsobem – Motýl třepe křídly na jednom místě na světě, například v Pekingu, a tento jev může mít za následek vyvolání zemětřesení v na opačné straně země koule, například v Kansasu. Můžeme vidět na obrázku 1 [1,2]



Obrázek 1: Motýlí efekt.

Zdroj: https://www.quora.com/Whats-the-best-real-life-example-of-the-butterfly-effect

Mnoho dalších badatelů se zabývalo těmito jevy, např.: David Roelle, Benoit Mandelbrot, Stuart Kauffman, atd., a to vedlo k položení základů studia nelineárních systémů a teorie chaosu v 80. letech 20. století. Mnoho vědců začalo upínat pozornost směrem k teorii chaosu, díky tomu, že univerzalita principů a zákonitostí zajišťovala možnost využití těchto principů v téměř každém odvětví. [2]

Popularita teorie chaosu dospěla do takového bodu, že v dnešní době bychom velmi těžko hledali obor, kterého se aspoň okrajově nedotkla. Mezi klasické příklady patří *matematika*, *chemie*, *fyzika*, *předpověď počasí*, ale nesmíme zapomenout i na takové obory jako *psychologie*, *sociologie*, *ekonomika*, *management* atd. [2]

2. Organizace

K bližšímu pochopení samoorganizace se musíme podívat na pojem, který je s ním velmi úzce spojen. Tímto pojmem je „organizace“. My se na organizaci budeme dívat z pohledu korporátního, kdy za organizace bude označovat skupinu lidí, kteří jednají v rámci společného cíle, a jsou k tomu motivováni a kontrolováni vůdčí osobou. [9,10]

Tato skupina lidí je rozdělena do organizačních struktur, která napomáhá splnění společného cíle. Organizace je vymezena vůči okolí a je jí přidělena role v rámci vlastní organizační struktury, kde každý člen má přidělená práva a povinnosti z ní plynoucí. Pro zajímavost slovo organizace bylo převzaté z řečtiny, kde slovo *organon,* znamenalo nástroj nebo instrument, který napomáhá splnění cíle. [9,11]

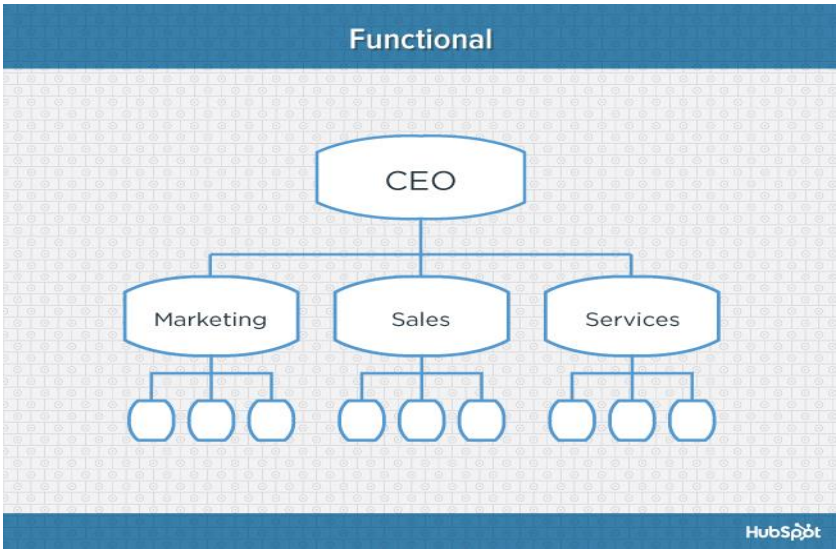
Pohled na organizaci je z různých úhlů pohledu velmi odlišný. Níže zmíníme dva z těchto pohledů. Prvním z nich je úhel pohledu, kdy se na organizaci díváme jako na mechanický stroj a druhý je pohled na organizaci jako na živý organismus. [11]

2.1. Mechanistický model organizace

Podle klasické teorie managmentu můžeme hledět na organizaci jako na stroj, která se skládá z jednotlivých součástek. Tyto součástky mají své funkce a každá součástka musí své funkce zdárně plnit, aby mohl stroj fungovat. Zde můžeme lidi v podstatě považovat za součástky stroje. Není u nich žádoucí vykazování činnosti mimo svoji vlastní určenou roli. Lidé přichází do práce v předem určený čas, pracují na předem připravené práci, poté mají čas na odpočinek a po odpočinku se vrací zpět ke své práci. Práce je velmi rutinní a opakuje se. Jedna směna je nahrazena jinou a takto může společnost fungovat nepřetržitě i 24 hodin denně. Tento model můžeme vidět například v továrnách, bankách apod. [9]

Podobný úhel pohledu můžeme najít i u restauracích a jiných organizacích poskytující služby, které pracují na základě podobných principů. Každá interakce je předem naplánovaná. Ku příkladu zaměstnanci jsou firmou školeni, jak reagovat při interakci s klientem. To je možné vidět u obchodních zástupců. Pozdrav, oční kontakt, rozhovor s klientem to vše je často dáno a nakázáno politikou firmy. Zaměstnanec je zkoušen ze znalostí postupů a kontrolován v jejich dodržování za účelem dosažení nejvyšší efektivity a nejlepších výsledků. Veškeré činnosti jsou poté zkoumány a ohodnoceny. Tím může organizace sledovat, jak je zaměstnanec výkonný a zdali je možné další činnosti zlepšit a standardizovat. [9]

Pokud bychom se chtěli podívat, jak vzniká koncept mechanické modelu organizace, musíme se podívat na to, jak inženýři vytváří stroje. Neboť právě zde se vzala inspirace pro tento model. Stroje jsou vytvářené jako síť vzájemně souvisejících částí, které mají určené místo a jejich funkce je přesně definována. Funkce jednotlivých částí musí být synergické. Schéma mechanistické organizace můžeme vidět na obrázku 2. Propojení jednotlivých částí probíhá skrze nadřazené funkce. Nadřazená funkce dává příkazy a omezuje podřízenou funkci. Ta je povinna vykonávat rozkazy dané nadřazenou funkcí. To zajišťuje, že můžeme předem odhadnout, jak bude příkaz vykonán a určit předpokládanou cestu skrze organizaci od nejvyšších pozic až po pozice nejnižší. [9]



Obrázek 2: Mechanistický model organizace.

Zdroj: https://blog.hubspot.com/marketing/team-structure-diagrams

Tento způsob funkčnosti organizace má své výhody i nevýhody. Za výhody můžeme považovat předvídatelnost událostí a stabilitu celkové organizace. Stejně tak jako existují výhody, nic není dokonalé, proto zde existují i nevýhody, Mezi ně patří pomalá reakční doba na událost a obtížné reagování na změny. Organizace byla založená za účelem splnění předem zadaného úkolu. Není vytvořená za účelem inovace. [9]

Jedna z myšlenek školy byrokratického řízení klasické teorie managmentu nám říká, že organizace by měla být racionální systém, jehož hlavní prioritou je jeho efektivita. Zde vyvstává problém. Ačkoliv daná myšlenka může popisovat ideální stav a funkci organizace, není jednoduché tohoto stavu dosáhnout a následně ho i udržet. Na vině je lidský faktor. U neživých předmětů, jako jsou součástky stroje, můžeme očekávat a předpovídat jejich chování. Lidé jsou ale více komplexní, složitější, a to je činí i více nepředvídatelnými. [9]

Tato myšlenka je v dnešní době považována za zastaralou. Ačkoliv teoretici uznali, že v organizaci je potřebná jistá forma vedení, angažovanosti, volnosti a rovnosti, stále se snažili o to, aby byl člověk začleněn do mechanistického pohledu na organizaci a stal se pouhou součástkou ve stroji s předem určeným místem a rolí. [9]

To ovšem neznamená, že se nenajdou příklady dnešních firem, které jsou založené na tomto pohledu na organizační strukturu. Mezi ně patří například McDonald's. K tomu, aby tento model fungoval musí být splněno několik podmínek:

- Vykonávaný úkol je přímočarý [9]
- Prostředí je stabilní natolik, aby byl zaručen kvalitní výstup z úkolu [9]
- Výstup z úkolu je homogenní produkt [9]
- Vysoká úroveň přesnosti [9]
- Lidské „součástky" nevykazují odchylky od jejich přednastavené role [9]

Pokud zůstaneme u příkladu McDonald's. Tato firma splňuje výše zmíněné podmínky. Poskytují nejen produkty, které musí být homogenní, ale musí zajistit, že přesnost ve výrobě téměř identických produktů je zastoupena v rámci všech poboček. Když se podíváme na zaměstnance, většina z nich jsou studenti, nebo lidé na částečný úvazek. Tito lidé projdou školením a je od nich vyžadováno chování podle firmou nastaveného vzorce. Neočekává se od nich, že budou „myslet". Očekává se od nich, že budou „poslouchat". [9]

Stejně jako máme několik podmínek, které potřebujeme splnit k úspěšnému aplikování výše zmíněného modelu, tak existují i jistá omezení. Mezi tyto omezení patří:

- Organizační struktura se může stát příliš složitou, což zapříčiní problémy v adaptování se na změny
- Nebezpečí vyústění v bezmyšlenkovité a nezpochybnitelné propadnutí administrativním procedurám
- Nemůžeme odhadnout následky v případě, kdy nějaká osoba odstoupí od svého úkolu a dá přednost svým zájmům, před splněním cíle organizace.
- Může vést k dehumanizaci zaměstnanců, hlavně na nižších úrovních organizační struktury

Je tedy potřebné zvolit správnou strukturu organizace a nezapomenout na to, jaké jsou podmínky a jaká omezení. [9]

Mezi podobná zařízení, která splňují výše zmíněné podmínky, nepřekračují uvedená omezení a do jisté míry úspěšně zapojují do fungování model mechanistické struktury patří například *dopravní služby*, *chirurgická oddělení*, *oddělení údržby letadel* a další. [9]

Dalším záporem, který zmíníme je problém zbavení se odpovědnosti. Jedním z kořenů problému je kontrola. Kontrola je prováděna nad zaměstnanci, týmy nebo odděleními, místo aby byla integrovaná jako jejich součást. Nadřízení monitorují výkon zaměstnanců, ale současně i zbavují zaměstnance zodpovědnosti. Zaměstnanec dostane určitou zodpovědnost v rámci své role v organizaci. Není žádoucí, aby z této role vystoupil. Neočekává se od něj „projev vlastního myšlení". Což může mít za následek úmyslné ignorování nastávajícího problému z důvodu, že nespadá pod jeho pracovní roli. [9]

Mechanistický model organizace byl velmi populární. Bylo to zapříčiněno nejen efektivitou modelu při vykonávání zadaných úkolů, které mohli být velmi jednoduše automatizovány, ale i příslibem pevné kontroly zaměstnanců a jejich aktivit. Což v dnešní době, kdy je na vzestupu vliv sociální a ekonomický, tak se v tomto modelu stále více a více projevují jeho omezení. [9]

2.2. Organický model organizace

Oproti mechanistickému pohledu na organizaci, organický pohled nehledá zastání v tvrzení, že organizace by měla fungovat jako stroj. Naopak hledá inspiraci v živých organismech. Můžeme sledovat závislost druhů organizace na prostředí ve kterém se nachází. Některé druhy organizace se adaptovali a prospívají v různých prostředí. Je to podobné jako u živých organismů. Velbloud je uzpůsoben pro život na poušti, a naproti tomu želva by bez vody na poušti dlouho nepřežila. [9]

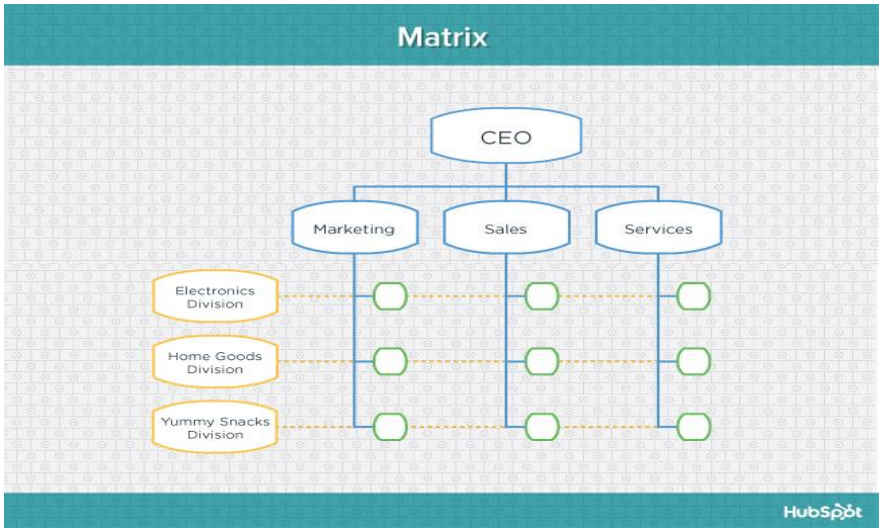
V předchozí kapitole jsme si pověděli o omezení mechanistického ohledu, a o tom, jak tento pohled u velké části moderních organizací není dostatečný. Díky tomu se směr pohledu na organizaci posunul jiným směrem. Zamířil do oboru biologie. To mělo za následek, že teorie organizace připomínala biologii ve smyslu, že termíny jako *molekuly, buňky, komplexní organismy, druhy* a *ekologie*, můžeme v teorii organizace nahradit pojmy *jedinec, skupina, organizace, druh organizace* a *sociální ekologie.* Tento vývoj měl za následek i snížení důležitosti efektivity organizace a splnění úkolu. Ta byla prioritní u mechanistického pohledu. U organického pohledu se zaměřujeme na problém přežití a další „biologické" problémy. [9]

Stejně jako organismy jsou otevřené systémy, tak to samé platí i pro organizace. Neboť organizace musí dosáhnout určité úrovně propojení s prostředím, jestli má přežít. Přístup otevřeného systému se obvykle zaměřuje na několik klíčových částí. Prvním z nich je důraz na prostředí, ve kterém systém existuje. V tomto je velký rozdíl oproti mechanistickému pohledu, kdy nebyla prostředí věnována žádná pozornost a systém byl považován za uzavřený. Druhým je existence vzájemně souvisejících vazeb mezi subsystémy. Subsystém můžeme pospat, jako organizaci obsahující jedince (Ti fungují každý jako samostatný systém), jedinci patří do skupiny nebo oddělení, skupiny spadají pod vedoucího a tak dále. Na celkovou organizaci koukáme jako na

systém, a jeho jednotlivé části jsou subsystémy. Je to stejná jako u buněk, ty jsou subsystémy orgánů i přesto jsou komplexní otevřený systém sami o sobě. Třetí část je snaha o vytvoření souladu mezi rozdílnými systémy a identifikace a eliminace potencionálních dysfunkcí. Podobně jako sociotechnický přístup k pracovnímu prostředí zdůrazňuje důležitost propojení lidského a technického aspektu, tak teorie otevřených systému dělá to samé se subsystémy. [9]

V případě že v prostředí nastane změna a vyvstanou nové problémy a výzvy, je potřeba flexibilní a otevřené organizace. Nejlépe je to vidět v prostředí, kde organizace musí inovovat z toho důvodu, že stagnace a zaostání ve vývoji by způsobilo její konec. Můžeme si to demonstrovat na firmách vyrábějící elektroniku. Pokud bychom zůstali u čistě mechanistického modelu organizace, nedokázali bychom včas reagovat na vývoj technologií a změnu preferencí zákazníka. Produkt musí být stále upravován a vylepšován. To by nebylo možné bez komunikace a spolupráce napříč odděleními a jejich zaměstnanci. Důležitý aspektem jsou schůzky, na kterých probíhá výměna informací, identifikace problémů a rozhodnutí o dalším pokračování. [9]

Organický pohled na organizaci si ještě více přiblížíme na maticové struktuře zobrazené na obrázku 3. Maticová struktura je pouze jednou z mnoha forem, které může organická struktura mít. Vliv nárůstu komplexnosti, složitosti a rozbouřenosti prostředí zapříčinil vznik mnohých dalších struktur, mezi které patří například *divizionální organizační struktura*, *hybridní organizační struktura* a další. [9]



Obrázek 3: Organický model organizace.

Zdroj: https://blog.hubspot.com/marketing/team-structure-diagrams

Maticová struktura patří mezi nejznámější organizační struktury. Také známá pod jiným názvem jako „projektová organizační struktura“. Jako součást této struktury vznikají podjednotky neboli týmy, kteří jsou současně i členi jednotlivých oddělení. To má za následek vznik dvojí pravomoci. Každý tým má vedoucího, ale současně spadá i pod vedoucího oddělení. Což může zapříčinit kolizi a rozpory mezi nadřízenými. Má to ovšem i světlé stránky. Mezi ně se dá počítat adaptabilita při jednání s prostředím, efektivnější využívání lidských zdrojů a lepší využití specializace zaměstnanců. Také dochází k určité formě oslabení vlivu a kontroly, což umožňuje lidem na nižší úrovni struktury angažovat se a přispět v případech, kde by za normálních okolností angažovat nemohli. [9]

3. Samoorganizace

Na začátek je potřeba si vyjasnit, co pojem samoorganizace znamená. Tento jev je také znám pod jiným názvem, a tím je spontánní řád. Označujeme tím proces, kdy z původně deterministického systému vzniká řád díky vzájemné interakci jednotlivých agentů bez potřeby kontroly externího agenta. Agent v tomto kontextu znamená jednoduchou část celku, která na základě svých vjemů a úrovně seznámení s prostředím volí adekvátní reakci, která maximalizuje výkon agenta. [4,8]

Projevy samoorganizace můžeme sledovat napříč všemi obory na které si vzpomeneme. Jelikož je velmi úzce spojena s teorií chaosu, obory ve kterých se samoorganizace projevu jsou různorodé. Škála oborů se táhne od matematiky až po psychologii. V jedné části práce se zaměříme na využití získané studiem světa zvířat zaměřené na jejich kolektivní chování. [5,8]

Vezměme si například mraveniště, včelí roj, stádo ovcí nebo hejno ryb. Všechny se pohybují podobných způsobem. Podívejme se na případy, kdy se tyto skupiny museli vyhnout nebezpečí, změnit kurz při cestování atd. Všechny tyto pohyby a mechanismy dávají zdání, jako by byli ovládáni jednou myslí, která je kontroluje. Toto chování bylo reprodukováno v počítačových simulacích, kdy bylo skupině dáno pár základních pravidel, podle kterých se měli členové skupiny chovat. Na základě toho začali vznikat logické vzorce chování aplikovatelné v různých odvětvích. [8]

Níže bych chtěl také nastínit rozdíl mezi pojmem samoorganizace a emergence. Díky tomu, že se v mnoha systémech velmi často vyskytují ruku v ruce oba pojmy, dochází k tomu, že jsou tyto dva pojmy zaměňovány. [7]

3.1. Historie

Příběh samoorganizace je velmi starý a má kořeny v antickém Řecku, kdy dva atomisté Democritus a Lucretius představili svoji myšlenku, že inteligentní design není potřebný k tvorbě řádu. Když dáme systému dostatek času, začne vznikat řád bez ohledu na externí zásahy inteligentního designu. [5,6]

Samotný termín “samoorganizace” byl poprvé použit významným německým filosofem jménem Immanuel Kant roku 1790 ve jeho knize Kritika Soudnosti. V této knize se zabýval i tím, jestli teleologii lze považovat za smysluplný koncept, či nikoliv. Tvrdil, že se jedná o smysluplný koncept v tom případě, že existují entity, jejichž části se chovají, jako by měli vlastní rozum a sami se i řídí. [5]

Tento pojem byl zahrnut až v roce 1947 do moderní vědy díky W. Ross. Ashbymu. Později díky Heinz von Foerster, Gordon Paska a dalších byl tento pojem rozšířen, a stále více vědců se jím začalo ve svých studiích zabývat. Zájem o samoorganizaci je v dnešní době na vrcholu a poznatky získané studiem nám i dodnes představují zajímavé pohledy na systémy. [5,8]

3.2. Termodynamický paradox

Pohled na systém, který, není řízen externím agentem, není pro člověka přirozený. Pokud systém ukazuje známky organizace, člověk má tendence hledat za tím skrytou sílu, v tomto případě externího agenta. Pokud nenalezneme agenta, který organizaci řídí, máme tendence přiřknout tento jev nějaké neznáme síle. Postupem času byla tato neznámá síla objevena a učinili jsme první krok k vysvětlení samoorganizace. Tajemnou sílou je druhý zákon termodynamiky hlásící, že v systému, který je ponechán bez zásahu, může entropie pouze růst, nikoliv se snižovat. [4,8]

Jako příklad nám poslouží proces krystalizace. Při procesu krystalizace dochází ke snižování entropie, což by podle druhého zákonu termodynamiky nemělo být možné, ale tato ztráta entropie je vyvážena zvýšením entropie v kapalině, z které se tvoří krystal. [4,8]

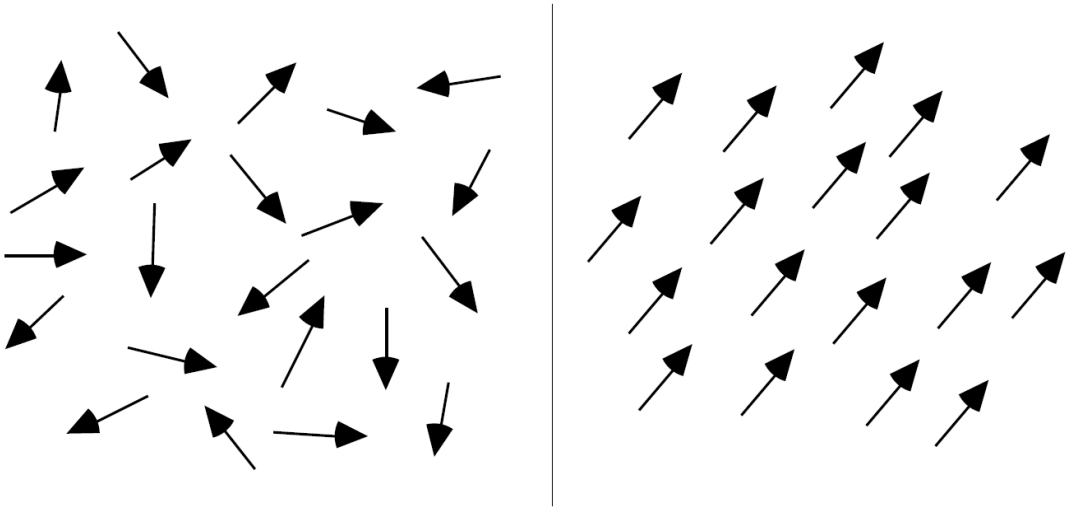
V případě systému, který nedosahuje rovnováhy nastává paradox. Nastává v tzv. otevřených systémech pro které je typické, že skrze ně prochází energie nebo hmota. Díky tomu se entropie generující v systému aktivně rozptyluje, nebo je přenášena mimo systém. V podstatě tímto způsobem obejdeme druhý zákon termodynamiky. [4,8]

Příkladem nám budiž živé organismy. Rostliny nebo zvířata přijímají energii a hmotu s nízkou entropií ve formě světla a jídla. Dojde k procesům ke zpracování a vše co rostlina, nebo zvíře nevyužije se vrátí zpět jako odpad s vysokou úrovní entropie. Pomocí tohoto mechanismu je umožněno organismům snižovat vlastní vnitřní entropii. [4,8]

3.3. Dva základní příklady samoorganizačních systémů

3.3.1. Magnetizace

Magnetizace by se dala považovat za nejjednodušší proces projevující samoorganizaci zkoumaný mnohými lidmi. Každý magnetický materiál se skládá z malinkých magnetů, které nazýváme spiny. Pro spin je typické, že mají svůj směr. Tento směr je udán jejich vlastním magnetickým polem. Díky různému směru spinu se magnetické pole jednotlivých spinů vyruší. Tento stav má na svědomí náhodný pohyb molekul v materiálu. [4,8]

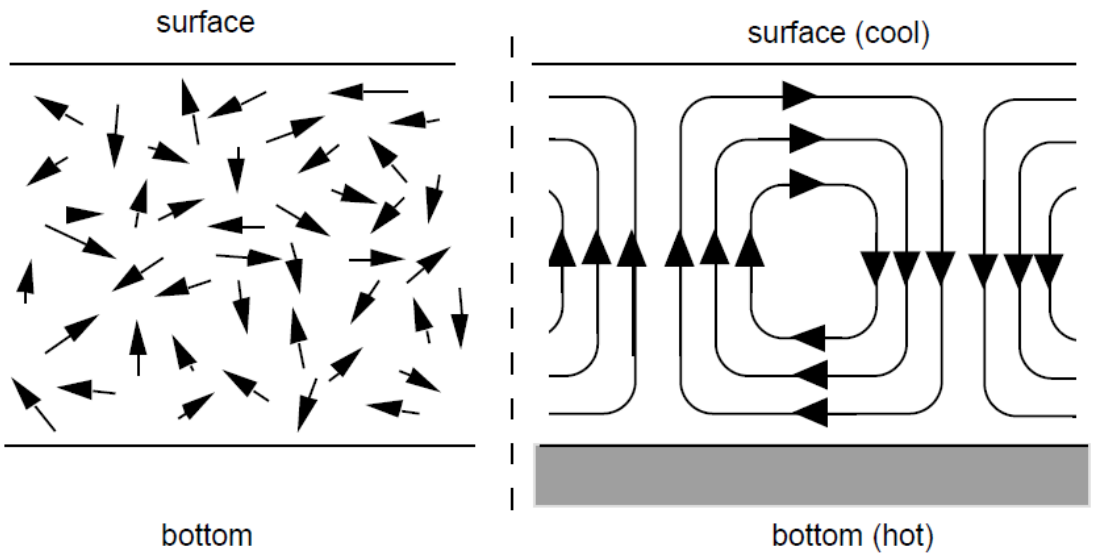


Obrázek 4: neuspořádané spiny (vlevo) a uspořádané spiny (vpravo). Zdroj: [8]

Pokud zvýšíme teplotu daného systému dojde i ke zvýšení pohybu molekul, což ovlivní spiny a sníží se schopnost vzniku a udržení řádu. Naopak pokud teplotu snížíme, spiny se spontánně seřadí v jednom směru. Tento jev můžeme vidět na obrázku 4. Důvodem je přitažlivost magnetického pole. Spiny, které směřují proti sobě, se vzájemně odpuzují a spiny, které směřují stejným směrem, jsou k sobě naopak přitahovány. [4,8]

3.3.2. Bénardova nestabilita

Druhý příklad je velmi podobný předchozímu. Ve smyslu zkoumání pohybu molekul. Tentokrát ale nebudeme používat magnetický materiál, ale kapalinu. [8]



Obrázek 5: Pohyby neuspořádaných molekul kapaliny (vlevo) a pohyb v důsledku projevu Bénardovy nestability. Zdroj: [8]

Na obrázku 5 můžeme vidět, že kapalina je uzavřena v nádobě. Ze spodní části kapalinu zahříváme a současně ji rovnoměrně ochlazujeme z vrchní části. Ohřátá kapalina má menší hmotnost a tendence stoupat vzhůru směrem k povrchu. Zde ochlazovaná kapalina má vyšší hmotnost a má tendence klesat ke dnu. Tato výměna ovšem nemůže proběhnout bez vzájemné koordinace kapaliny. Právě v tomto bodě se objevuje samoorganizace a vzniká proud, který chladnou kapalinu jedním směrem tlačí směrem dolu a druhým směrem tlačí teplou kapalinu vzhůru. [8]

3.4. Nelinearita a zpětná vazba

Na začátek se podíváme a vysvětlíme si, co je tradiční (lineární) systém. Do této kategorie spadá většina systémů. Je pro ně typické, že určitá akce vyvolá odpovídající reakci. Například pokud hodím kámen, doletí určitou vzdálenost a pokud bych ho hodil dvakrát silněji, doletí dvojnásobnou vzdálenost. Proti tomu stojí nelineární systém, který je typický právě pro samoorganizaci. Zde už vztah mezi akcí a reakcí není zcela předvídatelný. Malá akce nemusí vyvolat žádnou reakci nebo naopak vyvolá reakci mnohonásobně silnější. [8]

Jak s tím ale souvisí zpětná vazba? Každý agent ovlivňuje dalšího agenta, který ho ovlivňuje zpětně. Dochází k zacyklení akce a reakce. Pokud je u jednoho agenta vyvolána změna, druhý agent mu poskytne zpětnou vazbu v reakci na vyvolanou změnu a první agent znovu poskytne zpětnou vazbu zpět druhému agentu. [8,9]

Zpětná vazba může být dvojího typu. Rozdělujeme ji na pozitivní a negativní. Za pozitivní zpětnou vazbu považujeme tu, která posiluje původní změnu. Neboli pokud probíhá změna v jednom určitém směru, tak je tato změna zpětnou vazbou podpořena. Díky tomu rapidně narůstá počet odchylek v systému. To vede k rychlému vývoji s nepředvídatelnými výsledky. Za negativní zpětnou vazbu budeme naopak považovat tu, která původní změnu oslabuje, nebo potlačuje. Díky negativní zpětné vazbě se systém udržuje stabilní a snaží se dané odchylky vrátit zpět do původního stavu. [8,9]

V systémech, které jsou více komplexní, dochází k samoorganizaci, která je podpořena pozitivní i negativní zpětnou vazbou. Některé změny jsou posíleny pozitivní vazbou a jiné naopak oslabeny negativní. To má za následek vznik složitého chování, které lze jen těžko předvídat. [8,9]

3.5. Bifurkace, Symmetry breaking

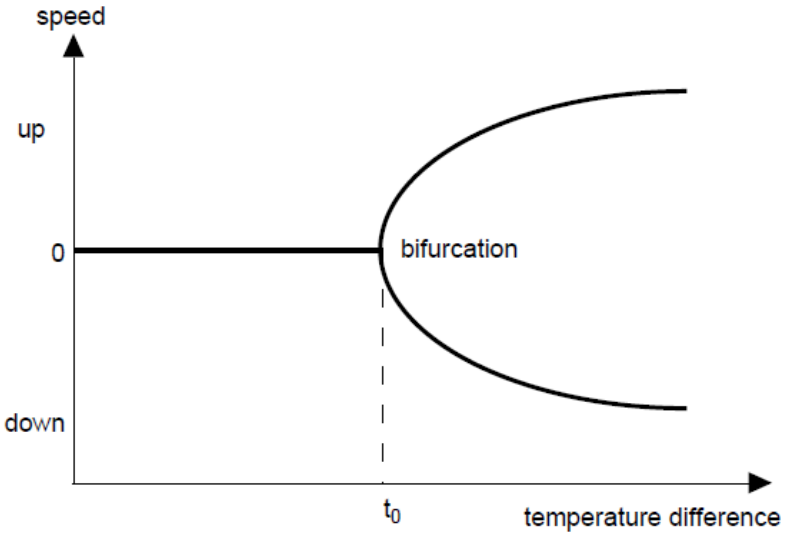
V předchozí kapitole jsme si stanovili, co je lineární, a co nelineární systém. U nelineárního systému neznáme počet možných správných konfigurací. Vezměme si výše zmíněný magnetismus. Nelze určit, kterým směrem se spiny seřadí. Vše závisí na reakci magnetického pole jednotlivých spinů. Podobně to funguje i u Bénardovi nestability. Pokud budeme předpokládat nádobu, která má stálý tvar, tekutina v ní se bude točit buď po směru nebo protisměru hodinových ručiček. [8]

Vítězná konfigurace sytému je výsledkem fluktuace. Malé fluktuace jsou násobené pozitivní zpětnou vazbou. To má bohužel za následek nemožnost určení původní fluktuace, která byla příčinou vítězné konfigurace. Pro pozorovatele ji není možné spatřit. Z toho důvodu se vítězná konfigurace považuje za nepředvídatelnou. [8]

Při zpětném pohledu na systém, před proběhnutím samoorganizace, uvidíme pouze jednu možnou konfiguraci a to neuspořádanou. Pro neuspořádanou konfiguraci platí, že zůstává u všech možných stavů stejná pravděpodobnost uskutečnění. Na příkladu magnetismu máme spiny neuspořádané. Nevíme, jak proběhne jejich samoorganizace a který směr spinu převládne. V této chvíli má ale každý směr spinů šanci stát se dominantním. Při pohledu na systém z globálního hlediska vidíme, že systém je homogenní. Každý směr spinu je rovnoměrně zastoupen. Systém je symetrický. Nezáleží, z jakého směru je pozorován, vypadá stále stejně. [4,8]

Po proběhnutí samoorganizace dojde k dominanci jedné konfigurace. Tím přijdeme i o symetrii. Úhel magnetizace se mění se změnou úhlu pohledu a Bénardova nestabilita vypadá jinak, když se na ní podíváme z obráceného úhlu. Tento jev se nazývá symmetry breaking. Při pohledu na samoorganizaci v souvislosti tohoto jevu se nám systém jeví jako by stál na hraně volby. Z počátku se chová ke všem stejně, později ale začne preferovat jednu z možností. Zajímavé je, že neexistují žádná objektivní kritéria, podle kterých systém rozhoduje pro své preference. Vypadá to, jako by se systém rozhodoval zcela náhodně. Díky velké šíři výsledných možností se nám systém jeví jako nepředvídatelný. [4,8]

K vývoji z neuspořádané konfigurace na uspořádanou většinou dochází za pomocí spouštěče. Za spouštěč můžeme považovat hraniční podmínky systému. Například Bénardova nestabilita se projeví pouze pokud máme dostatečný rozdíl teplot povrchů a dostatečnou velikost povrchů. Před projevením Bénardovi nestability máme rychlost pohybu molekul nulovou. Po projevení nestability se molekuly začnou pohybovat buď po směru nebo proti směru hodinových ručiček. Rychlost pohybu molekul souvisí s rozdílem teplot. Čím vyšší rozdíl, tím vyšší rychlost. [4,8]



Obrázek 6: Graf bifurkace na příkladu Bénardovi nestability. Zdroj: [8]

Na obrázku 6 máme zobrazený graf bifurkace. Ten nám ukazuje závislost rychlosti na teplotním rozdílu. Do rozdílu teplot t_0 je rychlost molekul nulová. Když teplotní rozdíl překročí hodnotu t_0 začne být nulová rychlost nestabilní. Stabilizuje se tím, že vzniknou dvě nové stabilní konfigurace, nahoru a dolu. [4,8]

Mnoho procesu podléhajících samoorganizaci může být znázorněno tímto grafem. Počátek samoorganizace závisí na proměnné, která zobrazuje možné typy řešení. Tato proměnná ukazuje přechod mezi neuspořádanou konfigurací a uspořádanou. Výše zmíněný příklad a graf byl jednoduchá ukázka bifurkace. Existují případy, kdy se konfigurace nemusí větvit pouze ve dvě, ale třeba i ve tři, čtyři a více řešení. Případně každá bifurkace se může i následně větvit kaskádovitě. [4,8]

3.6. Samoorganizace vs. Emergence

Samoorganizace a emergence jsou dva koncepty, které se často kombinují a společně jsou provázané. Z toho důvodu dochází k záměně těchto dvou termínů, případně je jeden z nich označen jako o synonymum druhého. Každý koncept zdůrazňuje rozdílné charakteristiky chování systému. Ukážeme si příklady, kdy může každý z těchto konceptů existovat samostatně nezávisle na druhém, a kdy naopak se tyto koncepty vzájemně proplétají. [7]

3.6.1. Emergence

Tom de Wolf ve svém článku vymezil definici emergence vzhledem k historickému vývoji takto:

„A system exhibits emergence when there are coherent emergents at the macro-level that dynamically arise from the interactions between the parts at the micro-level. Such emergents are novel w.r.t. the individual parts of the system.“ [7]

Ve výše uvedené definici si vysvětlíme několik pojmů. Slovo „emergent“ zde znamená výsledek procesu emergence: vlastnosti, chování, struktura, atd. Slovo „level“ využíváme pro popis pohledu. „Macro-level“ míníme pohled na systém jako na celek a „micro-level“ jako pohled jednotlivých entit, které jsou součástí systému. [7]

3.6.2. Charakteristiky emergence

V této části si shrneme pár charakteristik, kterými se vyznačuje emergence:

Micro-macro efekt - Tímto pojmem se rozumí veškeré vlastnosti systému, které se nachází na macro úrovni, ale vznikají vzájemnou interakcí částí na micro úrovni systému. [7]

Radical Novelty – Kolektivní chování systému není snadno pochopitelné z chování jednotlivých částí. Vždy je nutné dívat se na části systému v kontextu s celkem. [7,8]

Koherence – Také známá jako organizační uzavření. To znamená, že systém začne být zodpovědný za svoji údržbu a nezávislým na okolním systému. [7,8]

Interakce částí – Jednotlivé části spolu musí interagovat, jinak nevzniká emergence na macro úrovni. [7]

Dynamičnost – V systému, kde probíhá emergence, emergent vzniká v průběhu vývoje systému. Takový emergent vykazuje způsob chování, který je možný až po uplynutí určité doby. [7]

Decentralizovaná kontrola - K ovlivnění celkového chování organismu je využívaná kontrola pouze na lokální úrovni. Žádná část systému není kontrolována na macro úrovni, probíhá pouze kontrola jednotlivých částí systému. [7]

Dvousměrné propojení – V emergenci existuje dvousměrné propojení mezi macro a micro úrovní. Z micro úrovně na macro úroveň vzniká emergence. Opačně z macro úrovně na micro se mluví o ovlivňování částí celým systémem. Jako příklad nám poslouží znovu mravenci. Příkladem emergence je cesta vytvořená mravenci. Daná cesta ovlivňuje pohyb mravence na micro úrovni pomocí feromonu. [7]

Odolnost vůči narušení – Systém je odolný vůči narušením. V případě selhání části nehrozí selhání celého systému. Daná část, která selhala může být nahrazena jinou částí. [7]

3.6.3. Samoorganizace

Stejně jako v předchozí části Tom de Wolf vymezil pojem samoorganizace vzhledem k historickému vývoji:

„Self-organisation is a dynamical and adaptive process where systems acquire and maintain structure themselves, without external control." [7]

V definici použitý pojem „structure" může mluvit o prostorové, časové nebo funkční struktuře a „no external control" nám udává, že nesmí být přítomná manipulace, kontrola, tlak nebo ovlivňování vně systému. To neplatí o datovém toku z vnějšího systému, ovšem pouze do té doby, dokud nepřejímá kontrolu. [7]

3.6.4. Charakteristiky samoorganizace

V této části si shrneme pár charakteristik, kterými se vyznačuje samoorganizace:

Navýšení řádu – Vezměme si systém. Ten začíná jako částečně organizovaný nebo se zcela náhodnými vstupními podmínkami. Existuje možnost, že se vlivem změny organizovanost sníží. Obě situace budou řešeny navyšováním pořádku právě díky samoorganizaci. [7]

Autonomie – Neplatí, že každý systém, v kterém se navyšuje řád, je projevem samoorganizace. Dalším kritériem je absence kontroly, jak již bylo zmíněno výše. [7]

Přizpůsobitelnost a odolnost vůči narušení – V případě narušení systému, či změny, daný systém se přizpůsobí a zachová si svoji autonomii. [7]

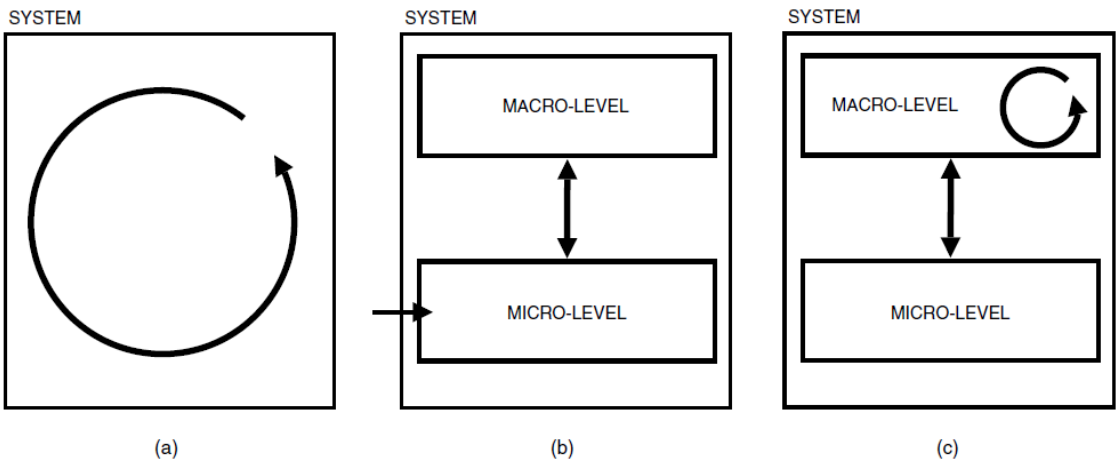
Daleko od rovnováhy (far-from-equilibrium) – Systém, který je daleko od rovnováhy je křehký a více náchylný ke změnám v prostředí, ale zároveň je více dynamický a schopný jednat. [7,8]

3.6.5. Porovnání emergence a samoorganizace

3.6.5.1. Podobnosti a rozdíly

Hlavní podobnost mezi oběma systémy se nachází v tom, že jsou oba dynamické procesy vznikající v průběhu času. Oba jsou odolné vůči narušení. Emergence je odolná ve smyslu odolnosti při poruše jedné části. Porucha součástky systému nekončí jako kompletní selhání. Samoorganizace je naproti tomu odolná ve schopnosti přizpůsobit se změně a schopnosti udržet si zvýšený pořádek. [7]

Samoorganizace a emergence se každá specifikuje na odlišnou část charakteristiky systému. Každá z nich může existovat samostatně, nebo existují společně vedle sebe. Na obrázku 7 si ukážeme jednotlivé projevy samoorganizace a emergence, jejich individuální projevy i jejich kombinaci. [7]



Obrázek 7: (a) samoorganizace bez emergence, (b) emergence bez samoorganizace, (c) kombinace samoorganizace a emergence. Zdroj: [7]

3.6.5.2. Samoorganizace bez emergence

Některé víceagentové systémy jsou autonomní a zvyšují svůj řád skrze interakce. Nicméně systémy nevykazují vlastnosti emergence. Pokud chybí jedna z jejích charakteristik, emergence neproběhne. Příkladem nám buď systém, kde je jeden kontrolní agent, který řídí chování systému (chybí decentralizovaná kontrola). [7]

3.6.5.3. Emergence bez samoorganizace

Zde je typický projev macro-micro efektu bez projevu samoorganizace. Na příklad se podíváme do fyziky. Termodynamika se může objevit ve stacionárních systémech (neprojevují samoorganizaci). Stacionární procesy se vyznačují tím, že v nich nedochází k nárůstu řádu. Vezměme si plyn. Ten zaobírá určitý objem v prostor. Tento objem projevuje znaky emergence, které vznikají interakcemi jednotlivých částic. Nicméně, plyn je ve stacionárním stavu a my nemůžeme v průběhu času změnit strukturu, pouze místo. V tomto případě máme sytém s výchozími podmínkami pro ukázání emergence. [7,8]

3.6.5.4. Kombinace samoorganizace a emergence

Ve většině systémech se objevuje emergence a samoorganizace ruku v ruce. Ve velmi komplexních systémech je dokonce kombinace výše zmíněných dvou konceptů doporučována. [7]

Ve velmi komplexních systémech je kladen důraz na to, aby jednotlivý agenti byli relativně jednodušší. Samoorganizace totiž vyžaduje navýšení řádu při podněcování určité funkce nebo vlastnosti. Jednoduchý agent nemůže ovládat takto komplexní systém, proto díky vzájemným interakcím vzniká globální chování systému (emergence). Funguje to i obráceným směrem. Kvůli komplexnosti systému není možné si vynutit reakci, která vyústí v emergenci. Jediná možnost na macro úrovni je nechat systém autonomně se zorganizovat. [7]

4. Swarm intelligence optimization

Swarm intelligence (SI) nebo také Inteligence hejna vzbudila zájem mezi mnohými vědci po celém světě. Podle Bonabeau definujeme swarm intelligence jako „The emergent collective intelligence of groups of simple agents“, což znamená, že swarm intelligence je kolektivní inteligence chování vykazující známky samoorganizace a decentralizovanosti systému. [12, 13]

Optimization neboli optimalizace je proces úpravy systému za účelem dosažení nejlepšího možného výsledku. Existují i případy, kdy se hledání nejlepšího možného řešení optimalizace systému stává neefektivním, v tom případě nehledáme nejlepší řešení, ale dostatečné pro optimalizaci. Z jiného úhlu pohledu se můžeme podívat na optimalizaci jako na hledání bodu maxima nebo minima v určitém omezeném prostoru systému. [13,14]

Při řešení optimalizace reálných problémů využili vědci znalosti získané studiem SI. Studovali chování živých systémů za účelem aplikace objevených principů při řešení optimalizačních problémů. Mezi zkoumané živé systémy patří například kolonie mravenců, včelí roj, apod. [12,14]

Níže se podíváme na několik algoritmů založených na optimalizaci využívající znalosti SI jako je *Genetic Algorithms*, *Artificial Bee Colony*, *Particle Swarm Optimization*, *Cuckoo Search Algorithms* a *Ant Colony Optimization* [12,14]

4.1. Genetic Algorithms (GA)

Genetic Algorithms (GA) byl představen Johnem Hollandem v roce 1975 jako algoritmus pro vyhledávání optimalizace na základě přirozeného výběru. Základní koncept algoritmu kopíruje koncept přirozeného výběru „přežijí jen nejsilnější“. Algoritmus simuluje proces pozorovaný v přírodních systémech, kde nejsilnější jedinci jsou schopni se adaptovat a přežít. Nicméně slabí jedinci se neadaptují a tudíž nepřežijí. [12,13]

Při aplikaci GA vytvoříme pro členy populace žebříček založený na vhodnosti k řešení aktuálního problému. Populace je reprezentována sadou řetězců, také nazývanou chromozomy. V tomto případě chromozomem myslíme člena populace. Chromozom je vytvořen na základě informace získané z člena, který byl v předchozí generaci považován za nejvhodnějšího. Byl umístěn nahoře na žebříčku vhodnosti. Tento algoritmus na začátku vytvoří počáteční populaci a překombinuje ji takovým způsobem, aby jeho hledání navedl směrem k výhodnějšímu stavu. Každé řešení je zakódováno v chromozomu a pomocí výsledků z žebříčku vhodnosti se rozhodne, jak je daný chromozom odolný a schopný se reprodukovat. Při zkoumání hodnot vhodnosti rozlišíme vhodnosti na nižší a vyšší. Pokud hledáme řešení pro maximalizaci zvolíme si výsledek, kde

hodnota vhodnosti dosahuje vyšší pozice na žebříčku vhodnosti a pokud pro minimalizaci, tak se přikloníme k nižší pozici. [12,13]

Základní GA má 5 hlavních částí:

- Generátor náhodných čísel [13]
- Jednotku pro hodnocení vhodnosti [13]
- Proces reprodukce – Proces výběru nejvhodnějších kandidátů populace [13]
- Proces křížení – Proces mísení nejvhodnějších chromozomů a předání nadřazených genů nastupující generaci [13]
- Proces mutace – Proces upravující některé geny v chromozomu [13]

Všechny tyto části jsou propojené a vzájemně na sebe navazují. Na počáteční populaci aplikujeme hodnocení vhodnosti. Sestavíme žebříček hodnot, a na základě výsledků budou chromozomy s méně vhodnými výsledky vynechány a s vhodnějšími zachovány. Ty boudou později použity k reprodukci. Po tom, co nám zbydou pouze vhodné chromozomy dochází k procesu křížení. Náhodně vezmeme dva chromozomy a zkřížíme je. Posledním procesem je proces mutace. Při něm je jeden z genů náhodně zmutován. Bez mutace by algoritmus nemohl zajistit kompletní prozkoumání problému. [12,13]

Ve výše popsaném procesu se vyskytuje určitá forma elitářství, která zde stojí jako určitá kontrola kvality. Hlídá nám dvě věci. První věc je zajištění, aby další generace byla aspoň stejná nebo lepší než generace předchozí a druhá je zamezení ztráty nejlepšího řešení během procesu křížení a mutace. [12,13]

Využití algoritmu GA je rozmanité. Mezi pár příkladů patří *robotika*, *matematika*, *manufaktura*, *hledání cest*, *plánování* a další. [12,13]

4.2. Artificial Bee Colony (ABC)

Artificial Bee Colony (dále ABC) je algoritmus založený na reálném fungování včelího úlu. Včelí úl vykazuje známky kolektivní inteligence. Algoritmus tyto projevy kolektivní inteligence zkoumá, kopíruje a adaptuje jejich proces hledání potravy včel. Včely hledají svoji potravu tzv. nektar. Při návratu do úlu se podělí o informace o jeho nalezišti. Uměle vytvořené agenty, kteří mají základ ve zjištěných poznatcích, můžeme roztřídit do tří skupin:

- Včely dělnice – zaměřují se na sběr potravy, nesou informaci o lokaci zdroje potravy
- Včely pozorovatelé – Shromažďují informace o zdroji potravy a předávají je dále
- Včely průzkumnice – Jejich úkolem je vyhledávání nových zdrojů potravy

Každá z těchto skupin včel má specifický úkol, který je jim přiřazen za účelem úspěšného ukončení algoritmu. [13,17]

Obecný proces ABC algoritmu si níže probereme krok za krokem.

- **Krok 1 - Fáze inicializace**

Všechny vektory $\underset{x_1}{\rightarrow}$ zdroje potravy známé populaci jsou zahájeny (i=1…SN, kdy SN je velikost populace) včelami průzkumnicemi a nastavením kontrolních parametrů. Každý vektor $\underset{x_1}{\rightarrow}$ má n počet proměnných, které optimalizujeme. Rovnice (1) je využívána v inicializační fázi:

$$x_i = l_i + rand(0,1) * (u_i - l_i) \qquad (1)$$

kde l_i a u_i jsou dolní a horní hraniční parametr x_i. [13,17]

- **Krok 2 – Fáze včel dělnic**

V této fázi dochází k vyhledávání nových zdrojů potravy $\underset{v_i}{\rightarrow}$. Frekvence hledání nových zdrojů potravy se zvyšuje s požadavkem úlu na vyšší množství nektaru v okolí zdroje potravy $\underset{x_1}{\rightarrow}$. Poté, co je nalezen zdroj potravy, dochází k ohodnocení jeho výnosnosti. Nový zdroj potravy a jeho okolí je charakterizováno funkcí (2):

$$v_i = x_i + \emptyset_i(x_i - x_i) \qquad (2)$$

kde x_j je náhodně vybraný zdroj potravy a $\emptyset_i$ je náhodné číslo z [-a, a]. Jakmile je nový zdroj potravy v_i nalezen, jeho výnosnost je ohodnocena a proběhne „greedy selection“ mezi $\underset{x_i}{\rightarrow}$ a $\underset{v_i}{\rightarrow}$. Pokud je hodnota výnosnosti u v_i vyšší než u jeho rodičovské x_i, x_i nahradíme v_i, jinak zachováme původní x_i. Poté co všechny včely dělnice dokončí proces hledání, včely pozorovatelé ohodnotí informace získané od včel dělnic. Poté vyberou zdroj s nejvíce nektarem a sdílí informace o zdroji potravy s ostatními. Hodnota výnosnosti (fitness value) je popsána za použitím rovnice (3):

$$fit_i\left(\underset{x_i}{\rightarrow}\right) = \begin{cases} \dfrac{1}{1 + f_i\left(\underset{x_i}{\rightarrow}\right)} & Když \quad f_i\left(\underset{x_i}{\rightarrow}\right) \geq 0 \\ 1 + abs\left(f_i\left(\underset{x_i}{\rightarrow}\right)\right) & Když \quad f_i\left(\underset{x_i}{\rightarrow}\right) < 0 \end{cases} \qquad (3)$$

kde $f_i\left(\underset{x_i}{\rightarrow}\right)$ je hodnota objektivní funkce řešení $\left(\underset{x_i}{\rightarrow}\right)$. [13,17]

- **Krok 3 – Fáze včel pozorovatelů**

Včely pozorovatelé čekají v úlu a vybírají zdroj potravy na základě hodnoty pravděpodobnosti za využití hodnoty výnosnosti a informace sdílené včelami dělnicemi. Hodnotu pravděpodobnosti p_i spočítáme pomocí rovnice (4):

$$p_i = \frac{fit_i\left(\underset{x_i}{\rightarrow}\right)}{\sum_{i=1}^{SN} fit_i\left(\underset{x_i}{\rightarrow}\right)} \qquad (4)$$

- **Krok 4 – Fáze včel průzkumnic**

Včely průzkumnice jsou nezaměstnané včely, které náhodně vybírají zdroje potravy. Včely dělnice, jejichž hodnota výnosnosti nemůže být vylepšena v předem daném počtu iterací se stávají včelami pozorovateli a všechny jejich zdroje potravy zůstávají opuštěné. [13]

- **Krok 5 - Fáze zapamatování**

Nejvyšší hodnota výnosnosti a pozice zdroje potravy náležící dané hodnotě je uložena do paměti. [13]

- **Krok 6 – Fáze ukončení**

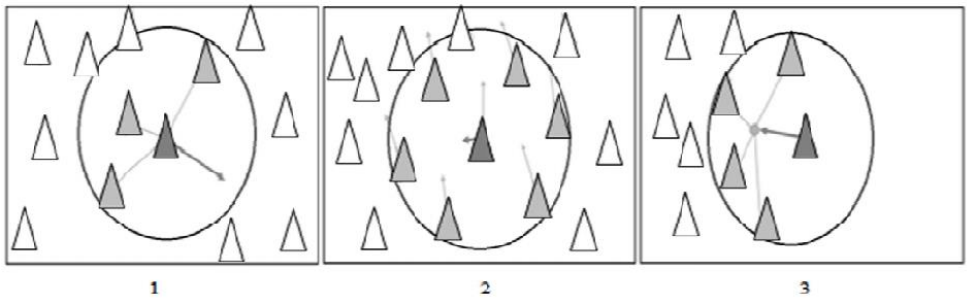
Pokud jsou podmínky pro ukončení algoritmu naplněny, program se ukončí. Pokud podmínky naplněny nejsou, program se vrátí zpět do kroku dva a zopakuje všechny následující fáze, dokud se podmínky nenaplní. [13]

Výhoda ABC algoritmu stojí na jeho schopnosti jednoduché implementace, robustnosti a vysoké flexibilitě. Vysoká flexibilita ABC je charakterizovaná tím, že vyžaduje pouze dva kontrolní parametry. Mezi tyto parametry patří číslo maximálního cyklu a velikost kolonie. Z toho nám plyne, že můžeme snižovat nebo zvyšovat počet včel v kolonii, aniž bychom museli upravovat algoritmus. ABC algoritmus může být využit v mnoha optimalizacích bez sebemenších modifikací. Vyžadovaný počet kontrolních parametrů je menší než u jiných algoritmů. [13,17]

Využití ABC zasahuje do mnohých odborů. Ať mluvíme o elektronice, plánování, zpracování obrazu, problémech technických návrhů nebo networkingu. [13,17]

4.3. Particle Swarm Optimization (PSO)

Particle Swarm Optimization (PSO) je velmi efektivní optimalizační algoritmus založený na jednoduchém mechanismu, který napodobuje kolektivní chování ptačích a rybích hejn, aby navedl jednotlivé částice k hledání souhrnného optimálního řešení. [12,13]



Obrázek 8: Základní vzorce chování u PSO. Zdroj: [13]

PSO se vyznačuje třemi základními vzorci chování zobrazeném na obrázku 8:

1. Oddělení – Částice se vyhýbají ostatním částicím [12,13]
2. Zarovnání – Částice se pohybují směrem k průměrnému směru lokálního hejna a udržuje jejich rychlost [12,13]
3. Soudržnost – Částice se pohybují směrem k průměrné pozici lokálního hejna [12,13]

Níže si popíšeme rovnice (5)(6) pro PSO algoritmus:

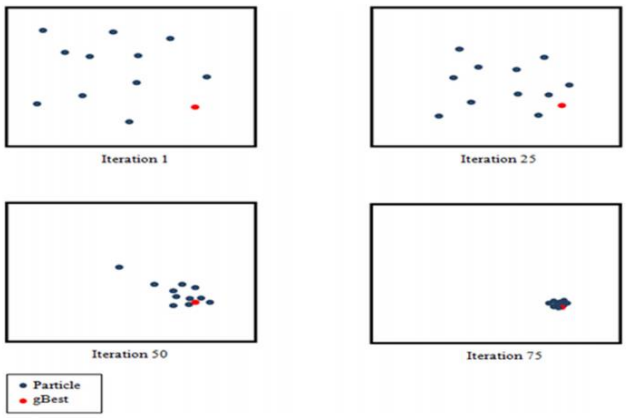
$$v_{id}^{t+1} = v_{id}^{t} + c_1 * rand(0,1) * \left(p_{id}^{t} - x_{id}^{t}\right) + c_2 * rand(0,1) * \left(p_{gd}^{t} - x_{id}^{t}\right) \qquad (5)$$

$$x_{id}^{t+1} = x_{id}^{t} + v_{id}^{t+1} \qquad (6)$$

kde v_{id}^{t} je částice rychlosti a x_{id}^{t} částice pozice. *d* je dimenze v prostoru, kde probíhá vyhledávání, *i* je částicový index a *t* je číslo iterace. *c1* a *c2* reprezentují rychlost regulující průběh. Řídí směr mezi souhrnnou nejoptimálnější částicí a lokální nejoptimálnější částicí. p_i je nejlepší pozice dosažená částicí *i* a p_g je nejlepší pozice nalezená částicí sousedící s částicí i. Pro $rand(0,1)$ platí, že nabývá náhodného čísla mezi 0 a 1. [13]

PSO algoritmus začíná inicializací populace. Poté proběhne výpočet hodnoty vhodnosti u každé částice, následující je aktualizace lokálních a souhrnných nejlepších částic a později dojde také k aktualizaci rychlost a směru částic. Toto probíhá, dokud nejsou podmínky ukončení naplněny. [12,13]

Na obrázku 9 níže můžeme vidět výstup z PSO algoritmu v průběhu narůstání iterací. V první iteraci se částice rozprostírají za účelem nalezení nejlepšího řešení. Každá částice je ohodnocena. Nejlepší lokální řešení jsou nalezena a postupně konvergují směrem k částici s nejlepším souhrnným řešením. [13]



Obrázek 9: PSO pohyb směrem k souhrnnému optimu. Zdroj: [13]

S využitím PSO algoritmu se pojí mnoho výhod. Mezi ně patří například jednoduchá implementace. Má pouze pár parametrů k nastavení, je efektivní pří celkovém prohledávání a může být jednoduše rozdělen do paralelních procesů. [12,13]

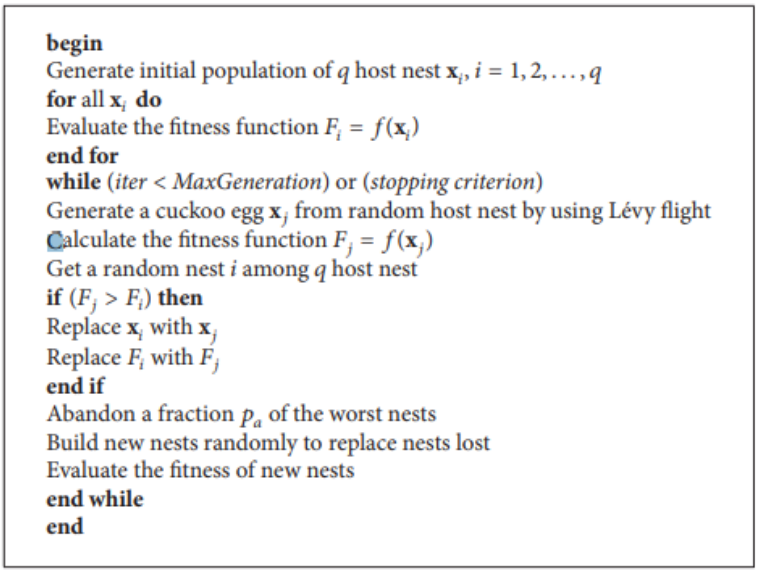
Využití pro algoritmus PSO můžeme najít v různých oborech. Například v *networkingu*, *napájecích systémech*, *kontrolních systémech*, *zpracováním obrazu* a v mnoho dalších. [12,13]

4.4. Cuckoo Search Algorithm (CSA)

Cuckoo Search algorithm (CSA) je algoritmus založený na parazitním chování kukaček. Máme několik předpokladů, podle kterých je toto chování adaptováno. [13,15]

- Každá kukačka snese pouze jedno vejce v náhodně vybraném hostitelském hnízdě, kde vejce reprezentuje možné řešení problému. [13,15]
- CSA se řídí konceptem „přežijí jen nejsilnější“. Pouze ti nejvhodnější s vysokou kvalitou vajec a hostitelských hnízd budou předány další generaci. [13,15]
- Počet hostitelů je předem pevně daný. Hrozí, že hostitelský pták může rozpoznat, že vejce není jeho. V takovém případě parazitické vejce vypudí z hnízda nebo může kompletně opustit hnízdo a postavit si na jiném místě nové. [13,15]

Algoritmus CSA je zobrazen na obrázku 10. Začínáme s populací možných řešení, kdy jsou nová, potencionálně lepší řešení vytvářena (kukaččí vajíčko). Pokud hodnota vhodnosti kukaččího vajíčka bude vyšší než náhodně vybraného vajíčka z volných hostitelských míst, poté bude horší řešení nahrazeno lepším (horší vajíčko, tím lepším). [13,15]



Obrázek 10: Cuckoo search algorithm (CSA). Zdroj: [13]

Při bližším zkoumání, jak kukačka vybírá náhodná místa byl zjištěn vliv tzv. Lévy flight, které je charakterizováno rovnicemi (7)(8):

$$x_i = (iter + 1) = x_i(iter) + \alpha * lévy(\lambda) \qquad (7)$$

$$lévy(\lambda) = \left| \frac{\gamma(1+\lambda) * \sin(\pi\lambda/2)}{\gamma(((1+\lambda))/2) * \lambda * 2^{((\lambda-1)/2)}} \right|^{1/\lambda} \qquad (8)$$

kde x_i je možné řešení, *iter* označuje aktuální číslo generace, λ je konstanta $(1 < \lambda \leq 3)$ a γ je funkce gama definovaná integrálem (9):

$$\gamma(x) = \int_0^\infty e^{-t}\, t^{x-1} dt \qquad (9)$$

Lévy flight je popisován, jako náhodná procházka, která je vytvářena náhodnými kroky podle funkce hustoty pravděpodobnosti s tlustým koncem. Velikost kroku je označena jako α. Začíná jako vysoká hodnota, postupně se snižuje a vrcholí v řešení směrem k poslední generaci. [13,15]

V dnešní době existují i novější verze tohoto algoritmu známé jako modifikované CSA (MCS), které začaly vznikat za účelem zkrácení doby trvání algoritmu. Původní algoritmus nebyl špatný, po určitém čase našel optimální řešení. Ovšem se vzrůstající složitostí a velikostí problému bylo v zájmu potřebnou dobu pro dokončení algoritmu zkrátit. To bylo zajištěno tím, že do algoritmu byl přidán jeden krok navíc. Tím krokem bylo sdílení informace mezi nejlepšími vajíčky. [13,15]

Stejně jako ostatní algoritmy, CSA je mezioborově velmi rozšířen. Mezi obory, kde se tento algoritmus využívá patří třeba *neurální sítě*, *ekonomie*, *vestavěné sítě*, *problém obchodního cestujícího* a další. [13,15]

4.5. Glowworm Swarm Optimization (GSO)

Glowworm Swarm optimization (GSO), také známý jako optimalizace hejnem světlušek, je algoritmus, který směřuje k optimalizaci multimodálních funkcí. Multimodální funkce mají více než jedno lokální minimum. [13]

V GSO máme fyzické entity neboli agenty, které nazýváme světlušky. Podmínky světlušek *m* v čase *t* mají tři základní poziční parametry, které se v průběhu algoritmu mění: [13]

- Místo, kde probíhá vyhledávání $x_m(t)$ [13]
- Úroveň luciferinu $l_m(t)$ [13]
- Rozsah vzdáleností světlušek v okolí $r_m(t)$ [13]

Na počátku jsou světlušky náhodně rozmístěny v pracovním prostoru. Později jsou inicializovány ostatní parametry za použití univerzálních konstant. Celý algoritmus probíhá ve třech fázích. Poté obdobně jako u ostatních metod, aby byl algoritmus ukončen, musí být naplněny podmínky ukončení. Pokud nejsou podmínky naplněny, tři fáze se zopakují. [13]

Mezi výše zmíněné tři fáze patří:

1. Aktualizace úrovně luciferinu. [13]
2. Pohyb světlušek. [13]
3. Aktualizace rozsahu vzdáleností světlušek v okolí. [13]

Pokud chceme, aby došlo k aktualizaci hodnot luciferinu, musí být ověřena vhodnost současné pozice světlušky *m* za použití rovnice (10):

$$l_m(t) = (1 - p) * l_m(t - 1) + \gamma J\big(x_m(t)\big) \qquad (10)$$

kde *p* je faktor vypařování luciferinu, γ je konstanta luciferinu a *J* je objektivní funkce. [13]

Pozice v místě, kde probíhá vyhledávání je aktualizovaná za použití rovnice (11):

$$x_m(t) = x_m(t-1) + s\left(\frac{x_n(t-1) - x_m(t-1)}{\|x_n(t-1) - x_m(t-1)\|}\right) \qquad (11)$$

kde s je velikost kroku, a $\|\ldots\|$ je operátor Euklidovské normy. Pokud je rozdíl mezi x_n *a* x_m velký, nastává explorace. Explorace přichází v případě, že potřebujeme prohledat více místa za účelem nalezení vhodného řešení. Pokud je rozdíl mezi x_n *a* x_m malý, dochází k exploatace, prohledávání lokálních míst za účelem nalezení lokálního optimálního řešení. [13]

Později dochází ke snaze světlušky najít svého souseda. V GSO platí, že světluška *m* považuje světlušku *n* za sousední v případě, že vzdálenost mezi nimi je menší, než je rozsah vzdáleností $r_m(t)$ a současně platí, že světluška *n* svítí jasněji než světluška *m*. Pokud nastane situace, kdy si světluška může vybrat mezi více sousedícími světluškami, vybere si jednu za použití rovnice pravděpodobnosti (12):

$$p_m(t) = \sum_{k \in N_i(t)} \frac{l_m(t) - l_n(t)}{l_k(t) - l_n(t)} \qquad (12)$$

kde pravděpodobnost posunu světlušky *m* směrem k světlušce *n* je rozhodnuta rozdílem úrovně luciferinu mezi nimi a úrovní luciferinu mezi všemi světluškami v dosahu světlušky *m*. Řešení s nejvyšší pravděpodobností je vybráno a světluška se posune směrem k vybranému sousedovi o jeden krok o konstantní velikosti *s*. [13]

V poslední fázi dochází k aktualizaci rozsahu vzdáleností světlušek v okolí $r_m(t)$ v závislosti na limitním omezení komunikace mezi skupinou světlušek. Rozsah vzdáleností světlušek v okolí je vypočítán pomocí následující rovnice (13):

$$r_m(t+1) = min\{r_s, max[0, r_m(t) + \beta(n_d - |n_m(t)|)]\} \qquad (13)$$

kde platí, že r_s je konstanta, která limituje velikost rozsahu vzdáleností, n_d je požadované množství sousedů, $|n_m(t)|$ je počet vhodných světlušek m v okolí v čase t a β modelová konstanta. [13]

Algoritmus GSO je velmi efektivní v případě aplikací, kde je omezený rozsah vzdáleností. Je schopný nalézt různé množství zdrojů a je aplikovatelný na numerické optimalizační úlohy. Nevýhodou je malá přesnost a pomalá konvergence. GSO je využíván v optimalizačních úlohách zabývajících se *vybíráním cest*, *zpracováním obrazu*, *lokalizačních problémech* a dalších. [13]

4.6. Ant Colony Optimization (ACO)

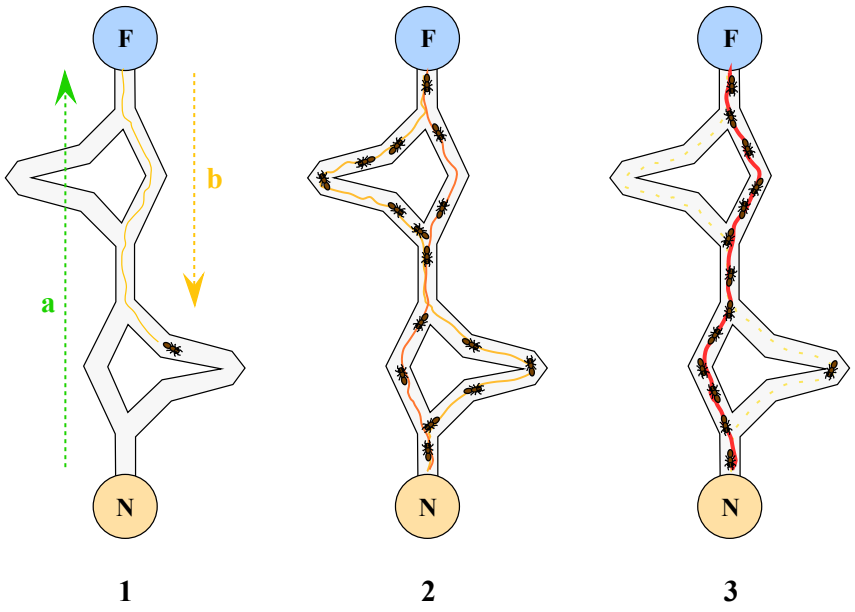
Ant Colony Optimization (ACO) algoritmus, je typ algoritmu založený na kolektivním chování mravenců při sběru potravy. Algoritmus se skládá ze čtyř hlavních komponent:

- Mravenec [13]
- Feromon [13]
- Akce démona [13]
- Decentralizovaná kontrola [13]

Všechny tyto komponenty přispívají k fungování celkového systému. [13,16]

Mravenci jsou imaginární agenti, kteří jsou použiti k imitování explorace a exploatace v místech, kde probíhá vyhledávání. V reálném životě je feromon chemická látka používaná mravenci k označení cest kudy cestovali a v průběhu času se vypařuje. V ACO algoritmu mravenci vypouští feromony při cestování prostorem, kde probíhá vyhledávání a množství feromonu určuje intenzitu stopy. Mravenci si vybírají cestu na základě nejvyšší intenzity stopy. Intenzita stopy může být považována za souhrnnou paměť systému. Akce démona je využívána pro sbírání souhrnných informací, které nemohou být získány úsilím jediného mravence a využijí tyto informaci k rozhodnutí o přidání feromonu za účelem dosažení konvergence. Decentralizovaná kontrola je využita ke zvýšení robustnosti a flexibility algoritmu v dynamickém prostředí. Důležitost decentralizované kontroly v ACO spočívá ve flexibilitě systému, kdy průběh algoritmu neohrozí. Všechny čtyři komponenty přispívají k vzájemné interakci, která vede k objevení nejkratších cest. [13,16]

Na obrázku 11 máme znázorněn proces ACO algoritmu. *N* znázorňuje kolonii, *F* znázorňuje zdroj potravy, *a* cesta za potravou a *b* je cesta zpět do kolonie. Na levém diagramu je znázorněno počáteční prostředí předtím, než algoritmus započne. V této části se mravenci pohybují zcela volně z kolonie k cíli a zpět. Na prostředním diagramu můžeme vidět již několikátou iteraci probíhajícího algoritmu. Zde mravenci objevili různé možnosti cesty mezi kolonií a cílem. Na diagramu číslo tři je vybrána nejkratší cesta. Tato cesta je opakovaně využívána mravenci, a díky tomu je na ní vysoká intenzita feromonu. [13,16]



Obrázek 11: Proces ACO algoritmu.

Zdroj: https://upload.wikimedia.org/wikipedia/commons/a/af/Aco_branches.svg

Kroky zahrnuté ve hledání nejlepšího řešení začínají výběrem dalšího uzlu (ze současné pozice v místě, kde probíhá vyhledávání) za použití rovnice (14):

$$p_{(i,j)}^{k}(t) = \frac{\left(\left[\tau_{ij}(t)\right]^{\alpha} * \left[\eta_{ij}\right]^{\beta}\right)}{\left(\sum_{k \epsilon J_k}\left[\tau_{ij}(t)\right]^{\alpha} * \left[\eta_{ij}\right]^{\beta}\right)} \qquad (14)$$

Kde p_{ij} značí pravděpodobnost cestování z uzlu i do uzlu j. J_k jsou uzle, kterými má mravenec povoleno cestovat z uzle i. η_{ij} nám přispívá k vlivu znalosti a viditelnosti mezi uzli i a j. $\tau_{ij}(t)$ značí množství nevypařeného feromonu mezi uzli i a j v čase t. α a β slouží jako kontrola vlivu $\tau_{ij}(t)$ a η_{ij}. V případě, že je hodnota α vyšší než hodnota β, chování vyhledávání závisí na feromonu. Pokud je ale vyšší hodnota β chování vyhledávání je závislé na viditelnosti nebo znalosti. Každý mravenec má seznam uzlů, který zabraňuje jakémukoliv mravenci v průchodu uzlu víckrát než jednou. [13,16]

Jak jsme si výše zmínili, feromony jsou podstatnou součástí ACO algoritmu. Zanechané stopy feromonu zvyšují pravděpodobnost, že si danou cestu další mravenec zvolí. Pro položení feromonu na cestu je využita rovnice (15):

$$\Delta\tau_{ij}^{k}(t) = \begin{cases} \frac{Q}{L_k}(t) \\ 0 \end{cases} \qquad (15)$$

kde Q je konstanta, L je náklad mravenčí cesty (například délka generované cesty), t je číslo iterace a k označuje specifického mravence. Hodnota $\Delta\tau_{ij}^{k}(t)$ nám ukazuje míru feromonu mezi uzli i a j, které mravenci navštívili v iteraci t. Jedna z hlavních komponent je míra vypařování feromonu určujíce explorační nebo exploatační chování mravence. Příliš vysoké hodnoty explorace vyústí v to, že se mravenec ztratí, zatímco příliš nízké hodnoty vyústí v neschopnost nalézt optimální cestu. Faktor vytrácení feromonu můžeme zjistit za pomoci rovnice (16):

$$\tau_{ij}(t+1) = (1-p) * \tau_{ij}(t) + \sum_{(k=1)}^{m} \left[\Delta\tau_{ij}^{k}(t)\right] \qquad (16)$$

kde m je počet mravenců v systému a p je míra vypařování feromonu nebo faktor vytrácení. [13,16]

ACO algoritmus má několik výhod oproti ostatním. Zahrnuje možnost pozitivní zpětné vazby, která umožní rapidní zrychlení při hledání řešení. Další výhodou je využívání již existujících souhrnných interakcí populace s agenty. Nicméně, existují i nevýhody. Mezi ně patří pomalá konvergence. Proběhnutí konvergence je sice garantované, ale doba potřebná pro dokončení je nejistá. V neposlední řadě je nevýhoda i v klesající výkonosti se zvětšujícím se místem, kde probíhá vyhledávání. [13,16]

Mezi nejčastější využití tohoto algoritmu se řadí obory jako *zpracování obrazu*, *projektový managment*, různé optimalizační problémy jako *problém obchodního cestujícího*, *plánování tras pro dopravní prostředky* a další. [13,16]

5. Metodická část

Potencionál aplikování teorie samoorganizace a optimalizačních algoritmů založených na SI je v dnešní době silný. Přesah těchto znalostí do všech různých oborů dělá z tématu zajímavý předmět diskuze. Když se nad tím zamyslíme, objevíme jistou formu paradoxu, kdy v době technického rozkvětu a vzestupu se vzdalujeme od přírody. Toho se snažíme docílit jejím studiem a aplikováním principů na kterých funguje. Využíváme tedy poznatky získané jejím studiem, a díky nim jsme schopni pokročit v nejen technické vyspělosti, ale i společenské.

V mé práci nastiňuji celkový obraz teorie samoorganizace a později se zaměřuji na optimalizační parametry založených na SI. Jako předmět mojí praktické části jsem zvolil Ant Colony Optimization (ACO) algoritmus z toho důvodu, že je to typický zástupce optimalizačních algoritmů založených na SI. Díky množství parametrů a modifikací je snadno upravitelný a vhodný pro zkoumání vlivu jednotlivých parametrů. Jak jsem již výše zmínil, existuje nepřeberné množství oborů, kde lze využít poznatky získané studiem SI. Například ACO algoritmus můžeme využít při řešení problému obchodního cestujícího, na který jsem se zaměřil i v mé práci.

5.1. Problém obchodního cestujícího

Problém obchodního cestujícího (TSP) nám popisuje problém obchodníka, který začíná ve svém domovském městě a snaží se najít nejkratší cestu skrze předem daný seznam měst a vrátit se zpět pod podmínkou, že každé město navštíví pouze jednou. Ve své práci se budu věnovat studiu řešení TSP, kdy budu porovnávat výstupy z ACO algoritmu v závislosti na měnících se parametrech α a β.

5.2. ACO software

K výzkumu závislostí parametrů v praktické části bakalářské práce jsem využil software Ant Colony Algorithm dostupný na https://www.codeproject.com/Articles/644067/Applying-Ant-Colony-Optimization-Algorithms-to-Sol#_Toc364710429. Tento program byl vytvořený za účelem poskytnutí implementace rozdílných modifikací algoritmu ACO pro řešení problému obchodního cestujícího.

Předtím, než jsem si vybral tento program k mému výzkumu. Chtěl jsem využít program ACOTSP. Bohužel kvůli nedostatečnému zobrazení výsledků a mezikroků probíhajícího algoritmu jsem od něj musel upustit a použít výše zmíněný program.

Mezi další programy, kde jsem zvažoval využití při zpracování dat jsou ACOTSP, AntNet, Myra, AntSolver, AntCar a další.

5.3. Popis programu

Jak jsem již zmínil výše, program Ant Colony Algorithm poskytuje implementaci několika modifikací ACO algoritmu pro řešení TSP pod jedním frameworkem. Mezi tyto modifikace patří Ant System, Ant Colony System, Elitist Colony System, MAX-MIN Ant System, Rank-based Ant System a Best-Worst Ant System. [16]

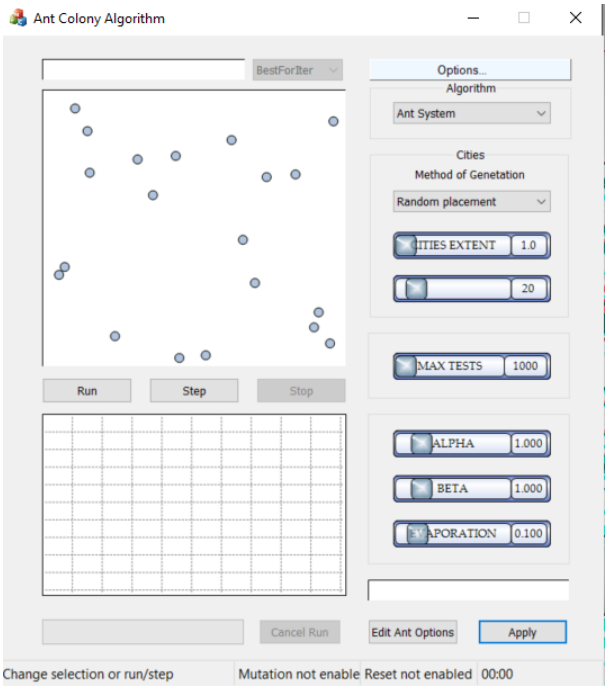
Tento program neřeší námi zadaný typ úlohy, kdy můžeme vložit soubor ve formátu TSPLIB a program nám vyřeší daný TSP, jako to bylo u ACOTSP, ale spíše slouží jako způsob prozkoumávání možností jednotlivých parametrů. K tomu nám vygeneruje mapu měst a na nich spustíme námi vybraný algoritmus podle námi zadaných algoritmů.

Program byl vytvořen v C++ pod operačním systémem Windows za hlavním účelem demonstrování principu řešení problému. Celý program je volně k dispozici včetně zdrojového kódu. Tato otevřenost otevírá velký rámec možností pro vlastní úpravu, a případné vylepšení programu. [16]

5.4. Použití programu

Mnou používaná aplikace je naprogramovaná v programovacím jazyce C++. Proto pro kompilaci kódu a spuštění je potřeba stáhnout MS Visual Studio 2012 na stránkách společnosti Microsoft. To nám umožní spuštění programu Ant Colony Algorithm.

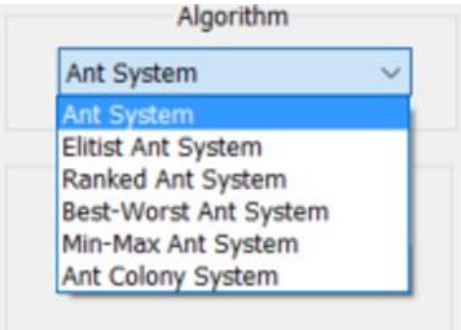
Program má přehledné grafické rozhraní. Rozhraní je možné spatřit na obrázku 12.



Obrázek 12: program Ant Colony Algorithm. Zdroj: autor

Většina parametrů má předem určené výchozí hodnoty, s kterými algoritmus pracuje, pokud neřekneme jinak. Některé parametry nelze nastavit bez zásahu do kódu. Například počet mravenců je zafixován na hodnotě deset. [18]

Na obrázku 13 máme zobrazené parametry, díky kterým můžeme vybrat modifikaci algoritmu, kterou chceme použít. Původní nastavená hodnota je Ant System. Pokud chceme využít jiný algoritmus musíme vybrat z rolovacího seznamu žádoucí modifikaci algoritmu. [18]



Obrázek 13: Parametry pro výběr modifikace algoritmu. Zdroj: autor

Parametry určující zvolenou modifikaci nepotřebují žádné argumenty k doplnění. U ostatních parametrů, pokud chceme některý z nich změnit, musíme vybrat hodnotu. [18]

Mezi další parametry, které musíme nastavit je „Method of generation" měst. Máme zde celkem 5 různých možností, kdy pro můj výzkum stojí za zmínku pouze mnou použitý „random placement". Tento parametr vygeneruje náhodně rozmístěná města v daném pracovním prostoru. Vzdálenost mezi dvěma městy bude počítaná jako Euklidovská vzdálenost dvou bodů. V kódu je také ošetřena možnost generace dvou měst na stejných souřadnicích. [18]

Další parametr, který nastavujeme, je „Cities extent", který nám udává velikost strany nejmenšího čtverce, ve kterém se nachází všechna vygenerovaná města. V neposlední řadě nastavujeme počet měst, které mají být vygenerovány. Aplikace umožňuje generaci měst pouze v rozmezí od 10 do 100. [18]

„Max Tests" je parametr, který nám označuje počet iterací daného algoritmu v rozmezí od 1 000 do 10 000. [18]

Mezi poslední skupinu nastavitelných parametrů patří „alpha", „beta" a „evaporation". Alpha nám zobrazuje sílu závislosti na feromonu, Beta zobrazuje sílu závislosti na znalosti a viditelnosti vzdálenosti měst a evaporation nám udává trvanlivost feromonové stopy. 1- hodnota evaporation lze chápat jako rychlost vypařování feromonové stopy. Hodnoty alpha a beta můžeme zadávat v rozmezí 0,2 až 5 a evaporation můžeme zadávat v rozmezí 0 až 0,8. [18]

Poté, co máme nastavené všechny parametry, dáme „apply“. To nám vygeneruje v pracovním prostoru mapu jednotlivých měst podle zadaných parametrů. „Run“ nám spustí algoritmus, „stop“ algoritmus zastaví a „step“ nám umožní procházet algoritmus po jednotlivých iteracích. [18]

5.5. Modifikace Ant Colony System

Jak jsem zmínil již výše, program umožňuje výběr z několika modifikací ACO algoritmů. Vybral jsem si a změřil se na algoritmus Ant Colony System. Oproti ostatním modifikacím zde existuje několik rozdílů. Hlavním rozdílem je způsob aktualizace feromonu. V ostatních modifikacích dochází k aktualizaci feromonu, až když všichni mravenci dokončí cestu v dané iteraci. Zde se potom, co mravenec cestuje z jednoho města do druhého, feromony na cestě, kudy mravenec prošel, sníží o určitou hodnotu. To nám zvyšuje možnost prohledávání alternativních cest. Stejně tak konstrukce řešení je od ostatních modifikací rozdílná. [16,18]

Algoritmus má parametr $q_0 \in [0,1]$. Tento parametr udává omezení pro dvě pravidla při výběru následujícího města. V každém kroku algoritmus vygeneruje náhodné číslo $0 \leq q \leq 1$. Když $q_0 < q$, potom je následující město vybrané pomocí rovnice (14) pro vyhledávání následujícího uzlu v podkapitole 4.6., pokud ne, je město *j,* označující místo přesunu, získáno pomocí rovnice (17):

$$j = arg\,max_{I \in T_I^k}\{\tau_{iI} * \eta_{iI}^{\beta}\} \qquad (17)$$

kde *i* označuje pozici současného města, *j* označuje pozici města, kam se máme přesunout, T_I^k nám označuje sadu měst, kam se můžeme přesunout ze současného města *i* v kroku k, τ_{iI} množství feromonu na cestě, η_{iI} je konstanta (v programu ve kterém pracuji označené jako city extent), β je parametr algoritmu. [16,18]

Poté, co se mravenec přesune z města *i* do města *j*, feromon na cestě se vypařuje a aktualizace proběhne podle rovnice (18):

$$\tau_{iI} \leftarrow (1 - \xi) * \tau_{iI} + \xi * \tau_0 \qquad (18)$$

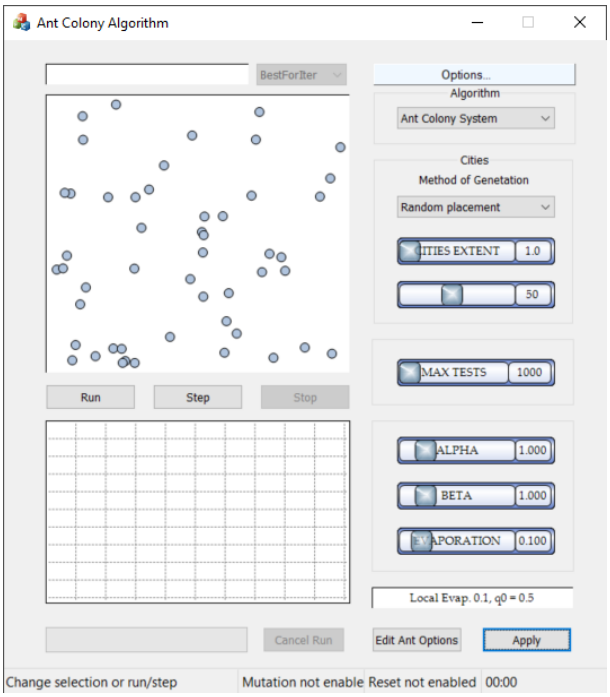
kde ξ je faktor lokálního vypařování, τ_0 dodatečný feromon (parametr algoritmu). Poté, co první mravenec dosáhne k-tého kroku a feromon je upraven, již nepodniká další krok. Čeká, dokud všichni ostatní mravenci nedokončí jejich k-tý krok, a upraví hodnoty feromonu na jejich cestě. V tomto procesu, n+1 mravenec čeká, dokud není provedena aktualizace mravencem n, a tak dále. [16,18]

6. Praktická část

Ve své práci sem se zaměřil na porovnání výstupů z programu, který aplikuje ACO algoritmus na problém obchodního cestujícího. Zkoumám vliv parametru α a β na výkonnost algoritmu. K tomu budu testovat kombinace výše zmíněných parametrů v rozmezí 0,2 – 5. Program neumožňuje nastavení těchto parametrů na nulovou hodnotu. Z logických důvodů je zde aplikováno toto omezení. Pokud by α = 0, potom by algoritmus nebral na vědomí existenci feromonové stopy, objevila by se kompletní absence komunikace mezi mravenci a neprobíhal by v systému vývoj směrem ke konvergenci. Pokud by β = 0, potom ani nejlepší řešení nepatří mezi optimální řešení. Mravenci se řídí čistě pomocí feromonové stopy, pokud je vytvořená feromonová stopa špatná, mravenci se jí budou stejně řídit.

6.1. Vliv parametrů α a β na výkonnost algoritmu

6.2. Výchozí nastavení aplikace



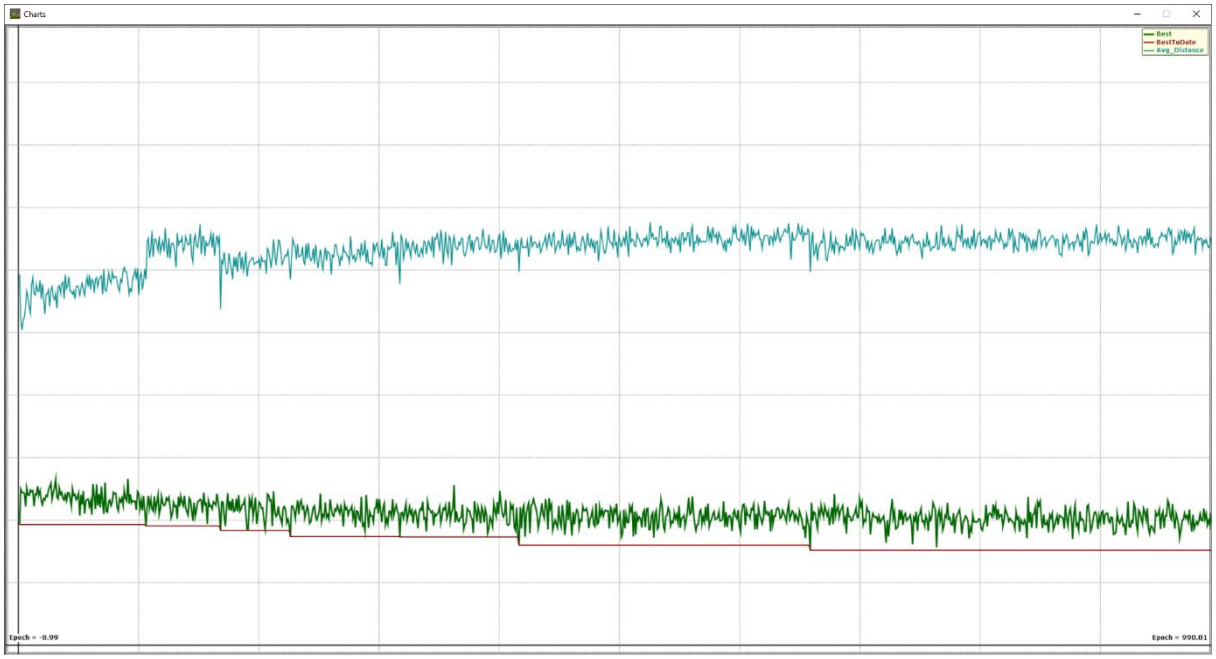
Obrázek 14: Výchozí nastavení aplikace, včetně rozmístění měst na které budeme aplikovat ACO algoritmus. Zdroj: autor

Ve výchozím nastavení, které budu používat rozvržení měst euklidovském prostoru dvou dimenzí dle obrázku 14. Celkem jsem použil 50 měst. Počet mravenců je v aplikaci fixně stanoven na deseti, počet iterací bude stanoven na 1 000 a doporučená hodnota vypařování pro Ant Colony Systém je podle literatury 0,1. [16]

6.3. Kombinace parametrů α a β

V následujících grafických výstupech máme na ose x zobrazený počet iterací. Zelená linie nám ukazuje nejlepší hodnotu délky cesty v dané iteraci. Zobrazuje nám, jak se v dané iteraci prozkoumává místo, kde probíhá vyhledávání. Algoritmus se snaží prozkoumat i možnost alternativní cesty. Červená linie ukazuje nejlepší hodnotu délky cesty k danému času. Postupně zobrazuje, v jaké iteraci bylo nalezeno lepší řešení, než bylo dočasné. Modrá linie nám ukazuje průměrnou vzdálenost. Pro každého mravence v dané iteraci je porovnávána jeho cesta s cestou nejlepšího mravence k danému času. Tato hodnota se používá jako míra pro hodnocení kvality cesty. Malá vzdálenost nám říká, že řešení bylo nalezeno. Algoritmus probíhá v 1000 iterací, v grafu je vidět rozdělení po 100 iteracích.

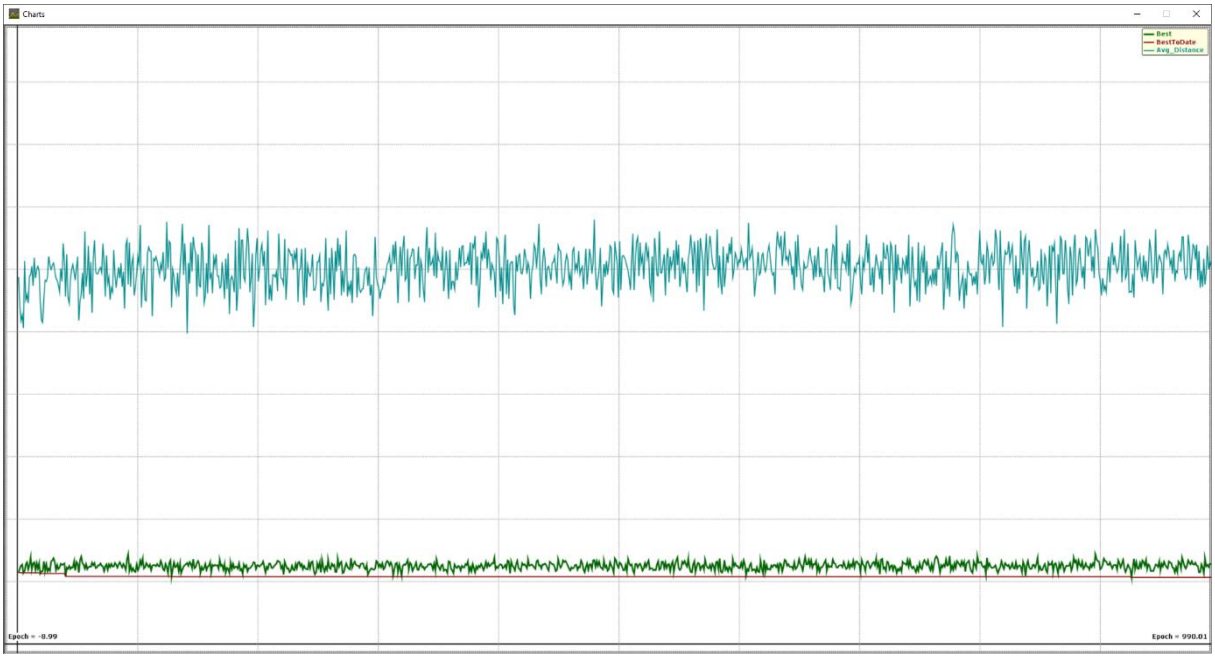
α = 0,2 a β = 0,2



Obrázek 15: Diagram pro α = 0,2 a β = 0,2. Zdroj: autor

Pro hodnoty parametrů **α = 0,2 a β = 0,2**, platí, že existuje velmi malý vliv feromonové stopy a velmi malý vliv znalosti a závislosti na vzdálenost měst. Červená linie nám ukazuje, že optimální řešení se formuje kolem 700sté iterace. Modrá linie nám ukazuje průměrnou vzdálenost mezi městy. Při těchto parametrech průměrná vzdálenost roste a nachází se na vysoké úrovni hladiny, to znamená, že mravenci sice nachází optimální řešení, ale vynakládají vysoké úsilí při prohledávání prostoru.

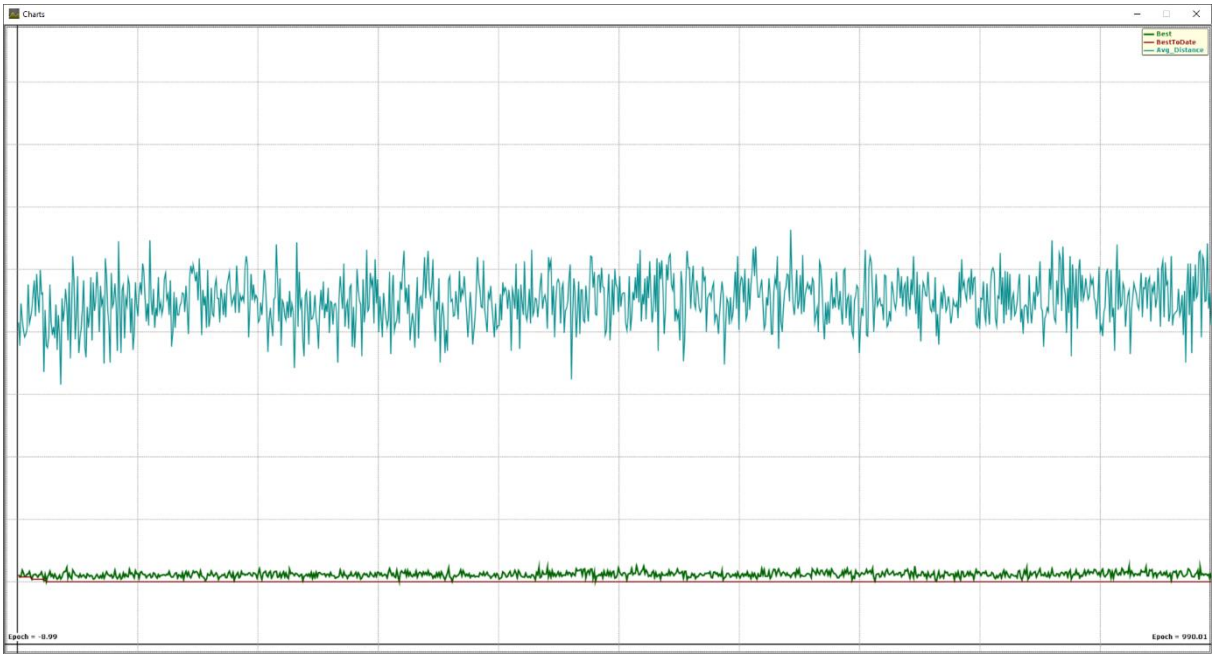
α = 0,2 a β = 3



Obrázek 16: Diagram pro α = 0,2 a β = 3. Zdroj: autor

Pro hodnoty parametrů **α = 0,2 a β = 3**, platí, že existuje velmi malý vliv feromonové stopy a střední vliv znalosti a závislosti na vzdálenost měst. Můžeme si všimnout, že optimální řešení bylo nalezené až v 900sté iteraci. Modrá linie nám ukazuje velké výkyvy v průměrné vzdálenosti měst při vysoké hladině hodnost. Chybí zde ovšem pokles průměrné vzdálenosti, který provází objevování preferované cesty.

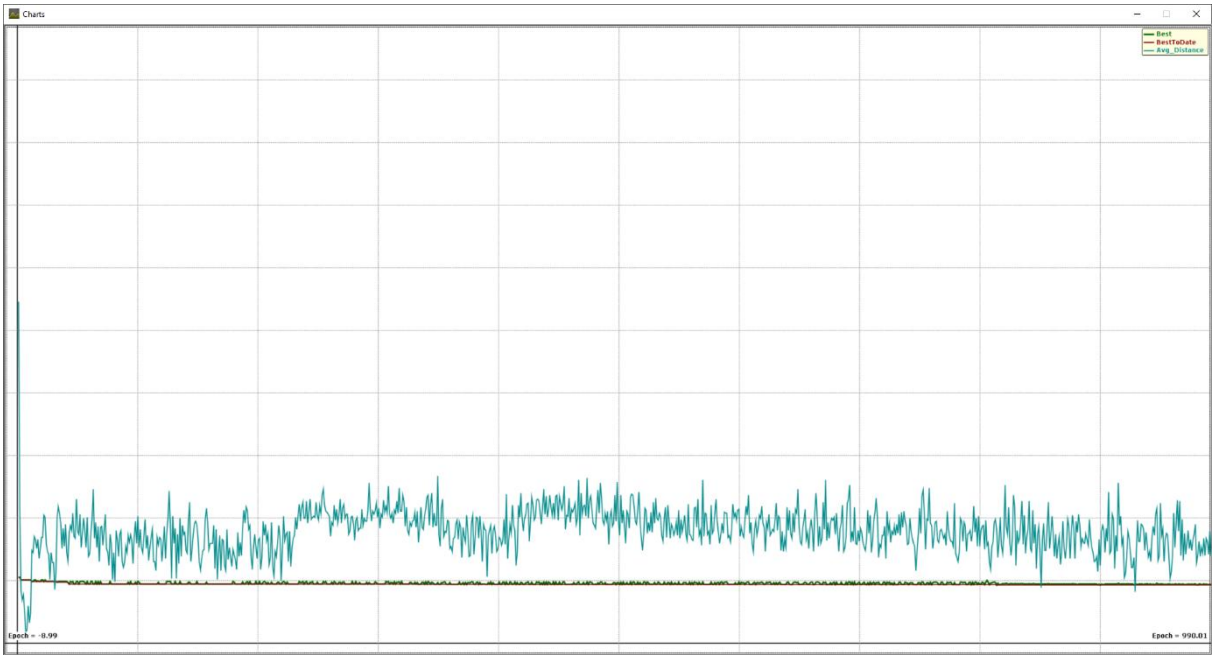
α = 0,2 a β = 5



Obrázek 17: Diagram pro α = 0,2 a β = 5. Zdroj: autor

Pro hodnoty parametrů **α = 0,2 a β = 5**, platí, že existuje velmi malý vliv feromonové stopy a silný vliv znalosti a závislosti na vzdálenost měst. Optimální řešení je nalezeno již v prvních 100 iterací. Ovšem u průměrné vzdálenosti se projevuje podobný problém, jako u předchozích diagramů.

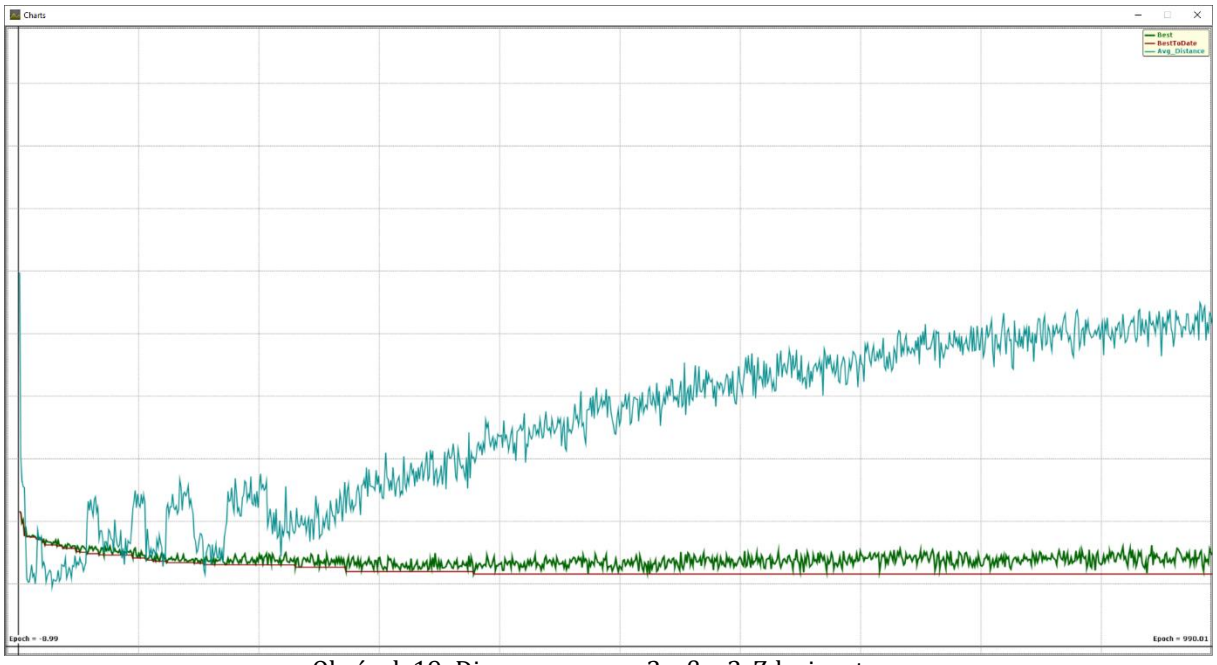
α = 3 a β = 0,2



Obrázek 18: Diagram pro α = 3 a β = 0,2. Zdroj: autor

Pro hodnoty parametrů **α = 3 a β = 0,2**, platí, že existuje střední vliv feromonové stopy a velmi malý vliv znalosti a závislosti na vzdálenost měst. Optimální řešení je nalezeno již v prvních 100 iterací. Průměrná vzdálenost na začátku algoritmu klesá, což znamená značí objev preferované cesty. Poté kolem 10 iterace začne průměrná vzdálenost stoupat. Zde se projevuje jev v literatuře pojmenovaný „deception“. Deception označuje jev, kdy dochází ke zhoršení průměrných výsledku s rostoucím časem. Tento jev je velmi podobný „přeučení“ u neuronových sítí.

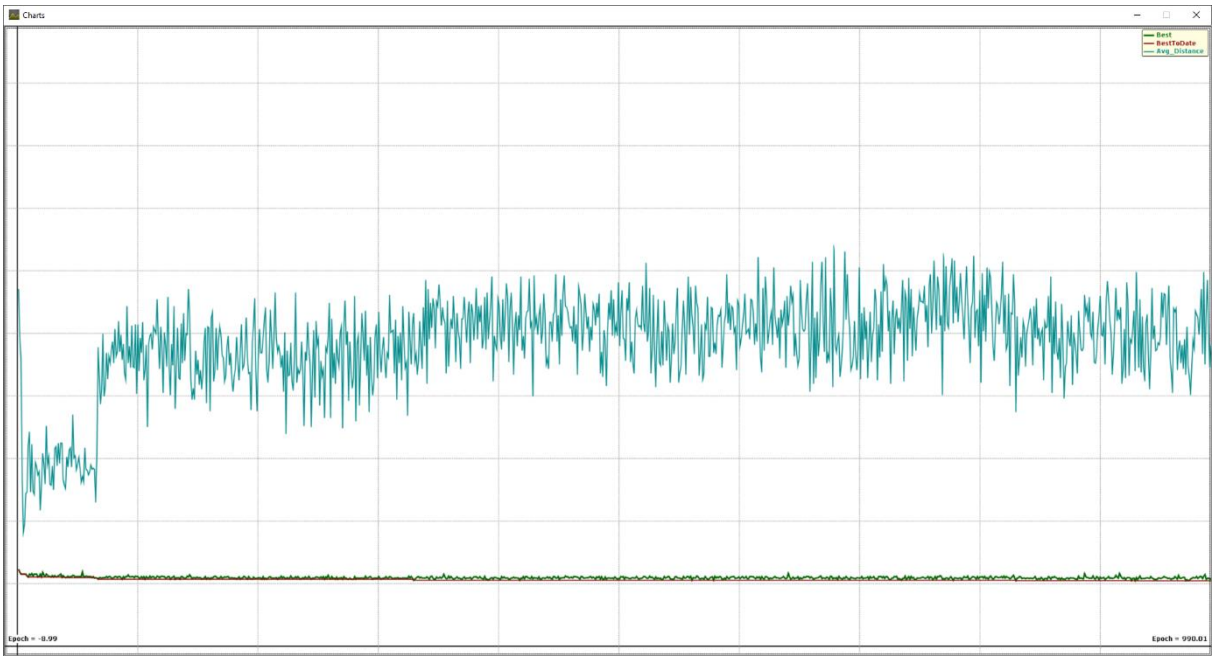
α = 3 a β = 3



Obrázek 19: Diagram pro α = 3 a β = 3. Zdroj: autor

Pro hodnoty parametrů **α = 3 a β = 3**, platí, že existuje střední vliv feromonové stopy a střední vliv znalosti a závislosti na vzdálenost měst. Optimální řešení je nalezeno před 400stou iterací. Průměrná vzdálenost na začátku algoritmu klesá, což znamená značí objev preferované cesty. Zde se opět projeví deception. Deception můžeme sledovat kolem 60. iterace, kdy dojde k výraznému zhoršení průměrných vzdáleností a toto zhoršování prudce stoupá s rostoucími iteracemi.

α = 3 a β = 5



Obrázek 20: Diagram pro α = 3 a β = 5. Zdroj: autor

Pro hodnoty parametrů **α = 3 a β = 5**, platí, že existuje střední vliv feromonové stopy a silná vliv znalosti a závislosti na vzdálenost měst. Optimální řešení je nalezeno v prvních 100 iteracích. Zde se deception projevuje velmi výrazně. V místě, kde je nalezena nejoptimálnější délka, můžeme vidět obrovský skok u průměrných vzdáleností.

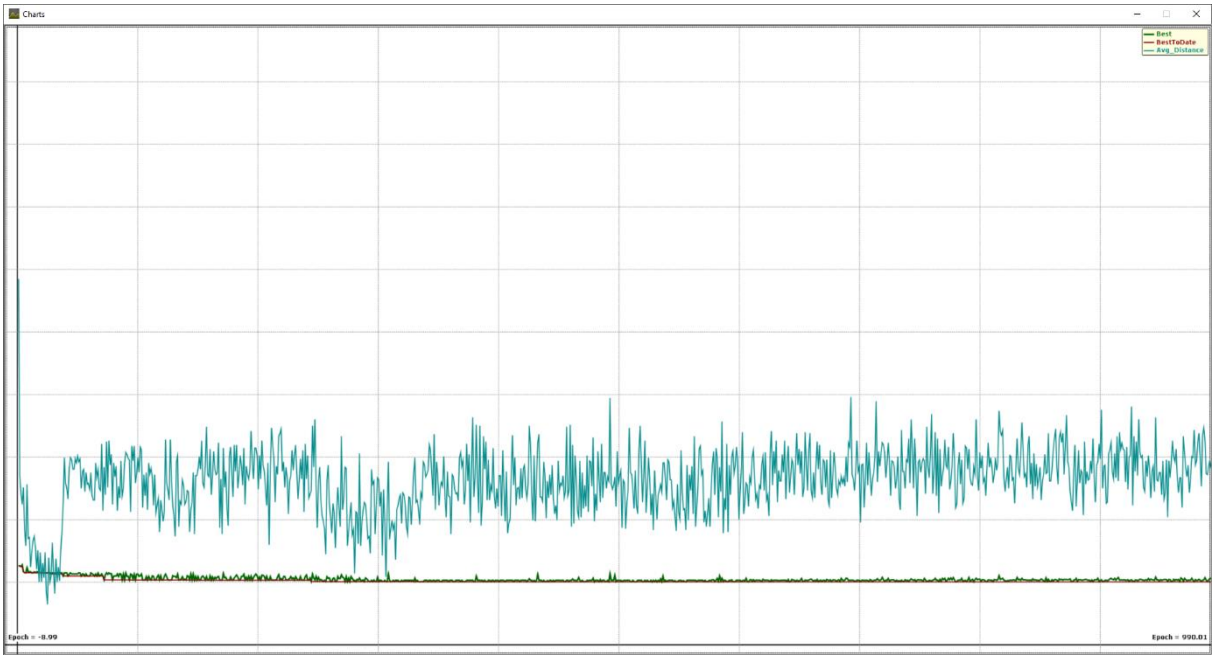
α = 5 a β = 0,2



Obrázek 21: Diagram pro α = 5 a β = 0,2. Zdroj: autor

Pro hodnoty parametrů **α = 5 a β = 0,2**, platí, že existuje silný vliv feromonové stopy a velmi malý vliv závislosti na znalosti a vzdálenost měst. Modrá linie zobrazuje skok průměrné vzdálenosti na nulu. V tomto případě algoritmus vstoupil do stagnace prohledávání. Podle Ant Colony Optimization od Marca Doriga dochází ke stagnaci, pokud je hodnota β = 0, nebo se k nule blíží, a současně α dosahuje vysokých hodnot. Ke stagnaci dochází z toho důvodu, že mravenci budou sledovat pouze feromonovou stopu, aniž by se spoléhali na viditelnost nebo znalost mezi městy.

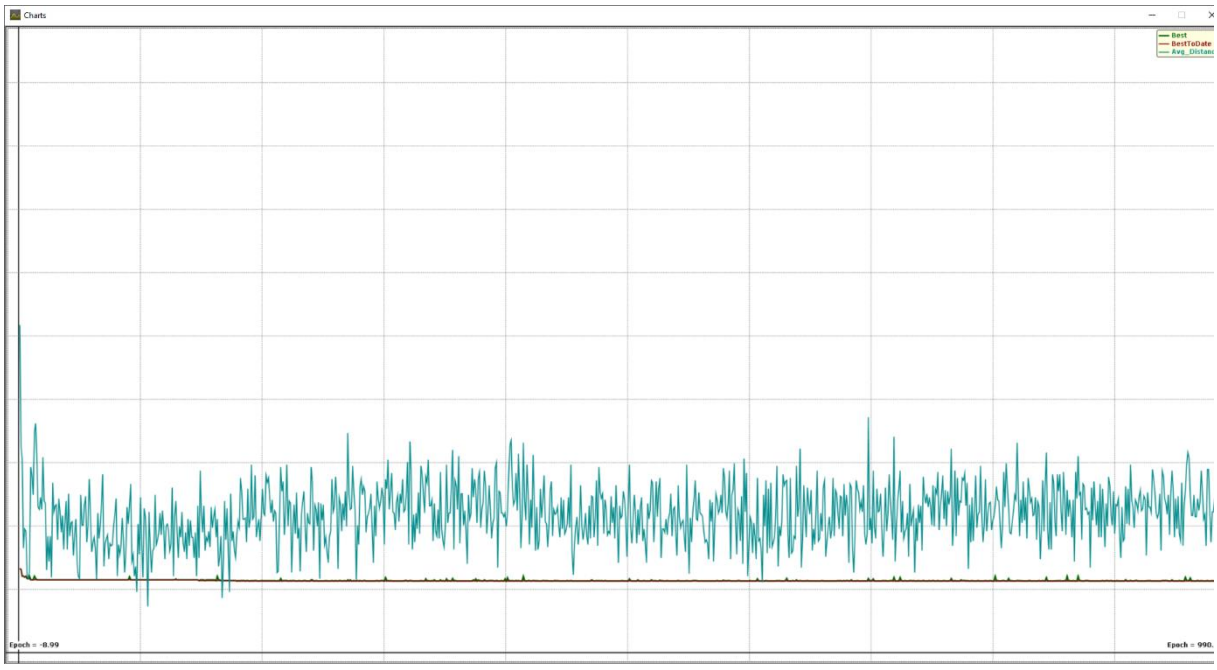
α = 5 a β = 3



Obrázek 22: Diagram pro α = 5 a β = 3. Zdroj: autor

Pro hodnoty parametrů **α = 5 a β = 3**, platí, že existuje silný vliv feromonové stopy a střední vliv závislosti na znalosti a vzdálenost měst. Optimální řešení je nalezeno v prvních 100 iteracích. Stejně jako v předchozích diagramech, tak i zde se projeví deception.

α = 5 a β = 5



Obrázek 23: Diagram pro α = 5 a β = 5. Zdroj: autor

Pro hodnoty parametrů **α = 5 a β = 5**, platí, že existuje silný vliv feromonové stopy a silný vliv závislosti na znalosti a vzdálenost měst. Optimální řešení je nalezeno kolem 10 iterace. Vidíme stále velké výkyvy v průměrných vzdálenostech, ale deception se zde projevuje minimálně.

6.4. Vyhodnocení vlivu parametrů α a β

V praktické části jsem provedl sérii testů, kdy jsem různě upravoval hodnoty parametru α a β. Na začátku jsme si řekli, že tyto parametry by měli být nenulové. Z toho důvodu, že při α = 0 se projevuje absence feromonové stopy, mravenci spolu nekomunikují, a při β = 0 se mravenci řídí čistě pomocí feromonové stopy, bez ohledu na svoji znalost. Na obrázku číslo 24 je zobrazena souhrnná matice zkoumaných parametrů a výsledných zjištění. Dále v textu jsou tato zjištění okomentovaná.

alfa / beta	0,2	3	5
0,2	Vysoké úsilí při prohledávání	Vysoké úsilí při prohledávání	Vysoké úsilí při prohledávání
3	Dobré výsledky	Dobré výsledky	Dobré výsledky
5	Stagnace	Dobré výsledky	Dobré výsledky

Obrázek 24: Matice výsledků zkoumaných kombinací parametrů α a β

Při nízkých hodnotách α i β trvá algoritmu vysoký počet iterací, než nalezne optimální řešení. Síla závislosti na feromonové stopě a síla závislosti na znalosti je malá. Mravencům trvá mnohem déle najít optimální cestu. Se zvyšující se hodnotou parametru β dochází ke zkracování doby nalezení optimální cesty, ale stále zde chybí vliv feromonové stopy. To má za následek vyšší průměrnou vzdálenost, kterou mravenci prochází při hledání optimální cesty. Se zvyšující hodnotou parametru α se algoritmus dostal do prohledávací stagnace. K té dochází, pokud je hodnota parametru β nulová nebo se blíží k nule a zároveň hodnota parametru α je vysoká. Což můžeme vidět na Obrázku 21.

U hodnot parametru α ≥ 1 se začíná objevovat jev, který je v literatuře známý pod pojmem deception. Jeho projev můžeme vidět v mých testech, kdy parametr α má hodnotu 3 nebo 5, a současně parametr β dosahuje hodnot 0,2, 3 nebo 5. Deception je pojem používaný pro jev, kdy se vzrůstajícím časem dochází ke zhoršování průměrných vzdáleností. [16,18]

V Ant Colony Optimization z roku 2004 Marco Dorigo definuje dobrou hodnotu parametrů α i β, kdy ideálně α ≥ 1 a β je v rozmezí 2 až 5. Podle mnou provedených testů tyto hodnoty většinou souhlasí s hodnotami z literatury. V případech, kdy byli v mých provedených testech parametry α i β pod hodnotou stanovených v Ant Colony Optimization dochází k vysokým nárokům při hledání optimálního řešení. Optimální řešení nalezeno je, ale celková optimalizace se nerealizuje. [16,18]

Ant Colony Optimization jako tradiční zástupce optimalizačních algoritmů SI má obrovské využití napříč většinou oborů. Nejen ACO algoritmus, ale i ostatní mají své místo u například vzestupující umělé inteligence. V mé práci jsem dokázal, že správným zvolením parametrů můžeme ušetřit programu práci, nebo naopak, nesprávným zvolením parametrů můžeme zbytečně navýšit nároky na běh algoritmu. Nemůžu nezmínit i další problémy se špatně nastavenými parametry α a β. Při nastavením nízkých hodnot těchto parametrů riskujeme to, že algoritmus nenajde optimální řešení, nebo označí optimální řešení chybně. Což znovu vede k výkonnostním problémům algoritmu. V neposlední řadě se může projevit stagnace, kdy celý algoritmus skončí, aniž by našel optimální řešení.

Závěr

Se samoorganizací se v jisté formě setkáváme každý den. Stačí vyjít ven a rozhlédnout se. Pokud nechceme chodit ven, stačí se podívat z okna. V přírodě je samoorganizace velmi běžná, i když se to na první pohled nezdá. Vezměme si mé oblíbené mravence. Při prvním pohledu na mravenčí kolonii vidíme spoustu práce a pohybu, ale pokud nemáme hlubší znalosti, neuvidíme souvislost se samoorganizací. Nevíme, že se mravenci organizují sami za pomocí feromonu. Bez souvislostí si klidně můžeme myslet, že je tam jeden vedoucí mravenec, který řídí veškerý pohyb a práci.

Prvním cílem, který jsem si stanovil, bylo vysvětlení pojmu samoorganizace. Započal jsem s teorií chaosu, kdy jsem krátce nastínil její historický vývoj a představil základy této teorie. Poté jsem zaměřil na organizaci, vysvětlil jsem, co tento pojem znamená, a blíže jsem ho přiblížil z pohledu modelů organizace. Po organizaci jsem se přesunul k samotné samoorganizaci. Vysvětlil jsem, co pojem samoorganizace znamená a popsal jsem krátce historický vývoj. Ukázal jsem dva základní příklady samoorganizačních systémů, mezi které patří magnetizace a Bénardova nestabilita. Poté jsem porovnával pojmy emergence a samoorganizace, protože tyto dva pojmy se velmi často v literatuře zaměňují. Zakončil jsem to přehledem optimalizačních algoritmů založených na swarm intelligence. Tyto algoritmy využívají principu samoorganizace a jejich využití se rozprostírá přes nesmírnou škálu oborů. S tím souvisí i druhý stanovený cíl, tím bylo uvedení jednotlivých příkladu právě výše zmíněných optimalizačních algoritmů. Popsal jsem šest nejrozšířenějších algoritmů založených na swarm intelligence využívající samoorganizaci, a na konci jsem se zaměřil na nejznámější z těchto algoritmů. Tím algoritmem je Ant Colony Optimization.

Třetí stanovený cíl bylo zkoumání vlivu parametrů α a β v modifikaci optimalizačního algoritmu Ant Colony System. Testování probíhalo způsobem, kdy jsem měnil hodnoty parametrů α a β při stejných výchozích podmínkách a sledoval, jak tyto změny ovlivňují výkonnost algoritmu. Zjistil jsem, že při nízkých hodnotách alfa, je sice nalezena nejkratší cesta, ale algoritmus se ani zdaleka neoptimalizuje. Při vyšších hodnotách těchto parametrů, většinou vyšších než jedna, bylo nalezeno optimální řešení v krátkém čase v nízkém počtu iterací. Při zkoumání se nám objevily i dva zajímavé jevy. Při jedné kombinaci parametrů α a β se algoritmus dostal do prohledávací stagnace. Druhým zajímavým jevem byl deception, kdy dochází ke zhoršování průměrných vzdáleností s rostoucím časem. Ukázalo se, že v algoritmech existuje silná závislost na hodnotách parametrů α a β a mé výsledky testů se ve většině shodují s dobrými hodnotami algoritmu navrhovanými Marcem Dorigem v Ant Colony Optimization.

Práci by bylo možné rozšířit o zkoumání vlivu dalšího parametru, například hodnoty vypařování feromonu. Můžeme zkoumat i závislost jiných parametrů, jako je počet mravenců nebo počet měst. To, ale mluvím o zkoumání parametrů v rámci jedné modifikace algoritmu. Další prostor pro rozšíření je směrem k porovnávání jednotlivých modifikací algoritmů a jejich výkonnosti.

Seznam literatury

[1] WOLFRAM, Stephen. A New Kind of Science: Notes for Chapter 7: Mechanisms in Programs and Nature. WolframScience [online]. 2002 [cit. 2018-03-10]. Dostupné z: https://www.wolframscience.com/reference/notes/971c

[2] DAŇA, Josef. Chaos Theory in Project Management [online]. Brno, 2014 [cit. 2018-03-10]. Dostupné z: https://theses.cz/id/4q9mo4/ Diplomová práce. Masarykova univerzita, Fakulta informatiky. Vedoucí práce doc. Ing. Daniela Kratchounova, PhD..

[3] Rennard, J.P. Social insects and self-organization. rennard.org. [Online] 2003. [cit. 2018-03-10] Dostupné z: http://www.rennard.org/alife/english/antstxt/antstxtgb.html.

[4] SORNETTE, Didier. Critical phenomena in natural sciences: chaos, fractals, selforganization, and disorder : concepts and tools. 2nd ed. New York: Springer, c2006. ISBN 978-3540308829.

[5] Self-organization. In: Wikipedia: the free encyclopedia [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2018-03-15]. Dostupné z: https://en.wikipedia.org/wiki/Self-organization

[6] PALMER, Ada. Reading Lucretius in the Renaissance. London, England: Harvard University Press, 2014. ISBN 9780674725577.

[7] DE WOLF, Tom a Tom HOLVOET. Emergence versus self-organisation: Different concepts but promising when combined [online]. BERLIN: SPRINGER-VERLAG BERLIN, 2005. 1-15 s. ISBN

0302-9743.

[8] HEYLIGHEN, F.: The science of self-organisation and adaptivity. In: The Encyclopedia of Life Support Systems. UNESCO Publishing-Eolss Publishers (2002)

[9] MORGAN, Gareth. Images of organization. Updated ed. Thousand Oaks: Sage Publications, c2006. ISBN 1-4129-3979-8.

[10] ASHBY, W. R. Principles of the self-organizing dynamic system. Journal of General Psychology. 1947, vol. 37, no. 2, s. 125-128. ISSN 0022-1309.

[11] MANAGEMENTMANIA. Manažer. Managementmania.com[online]. © 2011-2013. [vid. 2018-04-02], dostupné z: https:// anagementmania.com/cs/manazer

[12] KENNEDY, James, EBERHART, Russell C. a SHI, Yuhui. Swarm intelligence. San Francisco: Morgan Kaufmann Publishers, ©2001. xxvii, 512 s. The Morgan Kaufmann series in evolutionary computation. ISBN 1-55860-595-9.

[13] Mohd Nadhir, A. W., Nefti-Meziani, S, & Atyabi, A. (2015). A comprehensive review of swarm optimization algorithms. PLoS One, 10(5) doi: http://dx.doi.org.zdroje.vse.cz/10.1371/journal.pone.0122827

[14] Gao, Y., Du, W., & Yan, G. (2015). Selectively-informed particle swarm optimization. Scientific Reports (Nature Publisher Group), 5, 9295. doi:http://dx.doi.org.zdroje.vse.cz/10.1038/srep09295

[15] Ong, P. (2014). Adaptive cuckoo search algorithm for unconstrained optimization. The Scientific World Journal, doi:http://dx.doi.org.zdroje.vse.cz/10.1155/2014/943403

[16] DORIGO, Marco a STÜTZLE, Thomas G. Ant colony optimization. Cambridge: MIT Press, 2004. xiv, 305 s. A Bradford book. ISBN 0-262-04219-3.

[17] KARABOGA, D., GORKEMLI, B., OZTURK, C. and KARABOGA, N. A Comprehensive Survey: Artificial Bee Colony (ABC) Algorithm and Applications. The Artificial Intelligence Review, 06, 2014, vol. 42, no. 1. pp. 21-57 ProQuest Central. ISSN 02692821. DOI http://dx.doi.org/10.1007/s10462-012-9328-0

[18] CodeProject: Applying Ant Colony Optimization Algorithms to Solve the Traveling Salesman Problem [online]. [cit. 2018-04-22]. Dostupné z: https://www.codeproject.com/Articles/644067/Applying-Ant-Colony-Optimization-Algorithms-to-Sol#_Toc364710448

Seznam obrázků

Obrázek 1: Motýlí efekt. Zdroj: https://www.quora.com/Whats-the-best-real-life-example-of-the-butterfly-effect

Obrázek 2: Mechanistický model organizace. Zdroj: https://blog.hubspot.com/marketing/team-structure-diagrams

Obrázek 3: Organicky model organizace. Zdroj: https://blog.hubspot.com/marketing/team-structure-diagrams

Obrázek 4: Neuspořádané spiny (vlevo) a uspořádané spiny (vpravo). Zdroj: [8]

Obrázek 5: Pohyby neuspořádaných molekul kapaliny (vlevo) a pohyb v důsledku projevu Bénardovy nestability, Zdroj: [8]

Obrázek 6: Bifurkace na příkladu Bénardovi nestability. Zdroj: [8]

Obrázek 7: Graf samoorganizace a emergence. Zdroj: [7]

Obrázek 8: Základní vzorce chování u PSO. Zdroj: [13]

Obrázek 9: PSO pohyb směrem k souhrnnému optimu. Zdroj: [13]

Obrázek 10: Cuckoo search algorithm (CSA). Zdroj: [13]

Obrázek 11: Proces ACO algortimu, zdroj: https://upload.wikimedia.org/wikipedia/commons/a/af/Aco_branches.svg

Obrázek 12: program Ant Colony Algorithm. Zdroj: autor

Obrázek 13: Parametry pro výběr modifikace algoritmu. Zdroj: autor

Obrázek 14: Výchozí nastavení aplikace, včetně rozmístění měst, na které budeme aplikovat ACO algoritmus. Zdroj: autor

Obrázek 15: Diagram pro $\alpha = 0,2$ a $\beta = 0,2$. Zdroj: autor

Obrázek 16: Diagram pro $\alpha = 0,2$ a $\beta = 3$. Zdroj: autor

Obrázek 17: Diagram pro $\alpha = 0,2$ a $\beta = 5$. Zdroj: autor

Obrázek 18: Diagram pro $\alpha = 3$ a $\beta = 0,2$. Zdroj: autor

Obrázek 19: Diagram pro α = 3 a β = 3. Zdroj: autor

Obrázek 20: Diagram pro α = 3 a β = 5. Zdroj: autor

Obrázek 21: Diagram pro α = 5 a β = 0,2. Zdroj: autor

Obrázek 22: Diagram pro α = 5 a β = 3. Zdroj: autor

Obrázek 23: Diagram pro α = 5 a β = 5. Zdroj: autor

Obrázek 24: Matice výsledků zkoumaných kombinací parametrů α a β

Slovníček

Ant Colony Optimization (ACO) = Optimalizace mravenčí kolonií. Optimalizační algoritmus založený na inteligenci hejna inspirovaný chováním mravenčí kolonie.

Artificial Bee Colony (ABC) = Optimalizace včelím rojem. Optimalizační algoritmus založený na inteligenci hejna inspirovaný chováním včelího roje.

Cuckoo search algorithm (CSA) = Optimalizační algoritmus založený na inteligenci hejna inspirovaný parazitním chováním kukaček.

Deception = Jev, kdy se s rostoucím časem zhoršují průměrné vzdálenosti u ACO

Don't look bits = Technika, která umožňuje zaměřit se na lokální vyhledávání v místě, kde se může potencionálně něco dít.

Genetic Algorithms (GA) = Optimalizační algoritmus inspirovaný biologickou evolucí

Glowworm swarm optimization (GWO) = Optimalizace hejnem světlušek. Optimalizační algoritmus založený na inteligenci hejna inspirovaný chováním světlušek.

Greedy selection = Výběr lokálního optima, algoritmus dává přednost řešení, které je v dané době pro něj nejlepší

Inteligentní design = Teorie vystupující proti evoluční teorie. Tvrdí, že některé rysy vesmíru a živých stvoření se vysvětlují lépe inteligentní příčinou, a ne přirozených výběrem. Odborný způsob dokázání existence inteligentního stvořitele.

Kolektivní inteligence = Schopnost skupiny najít větší množství řešení, nebo kvalitnější řešení problému, než její členové naleznou sami.

Lévy flights = náhodná procházka, která je vytvářena náhodnými kroky podle funkce hustoty pravděpodobnosti s tlustým koncem.

Náhodná procházka = V matematice a fyzice užívaná formalizace intuitivní myšlenky provádění náhodných kroků. Každý další krok, obvykle stejné délky, je učiněn náhodným směrem.

Particle search optimization (PSO) = Optimalizace hejnem částic. Optimalizační algoritmus založený na inteligenci hejna inspirovaný chováním rybích a ptačích hejn.

Swarm intelligence (SI) = Inteligence hejna. Termín se používá pro poznatky získané sledováním kolektivního chování zvířat

Travelling Salesman Problem (TSP) = Problém obchodního cestujícího popisuje problém obchodníka, který začíná ve svém domovském městě a snaží se najít nejkratší cestu skrze předem daný seznam měst a vrátit se zpět pod podmínkou, že každé město navštíví pouze jednou.