Vlci a smečkový algoritmus ve světě membránových agentů

Daniel Valenta, Lucie Ciencialová, Luděk Cienciala

Slezská univerzita v Opavě, Filozoficko-přírodovědecká fakulta v Opavě,
Ústav informatiky
a
Research Institute of the IT4Innovations Centre of Excellence
Bezručovo náměstí 1150/13, 746 01 Opava
Email: daniel.valenta@fpf.slu.cz, lucie.ciencialova@fpf.slu.cz, ludek.cienciala@fpf.slu.cz

Abstrakt

Membránové systémy, zavedené v roce 1998 Gheorghem Păunem, představují jeden z biologicky motivovaných výpočetních modelů teoretické informatiky Păun (2000). Od jejich uvedení vznikla celá řada různých variant membránových systémů, v závislosti na struktuře a používaných pravidlech. Model P kolonie, jeden z modelů patřících mezi membránové systémy, se skládá z jedno-membránových agentů sdílejících společné prostředí. Agenti mají omezený počet objektů a množinu programů určujících způsob vývoje objektů nebo výměnu objektů s prostředím. Přestože se jedná o jednoduchý model tvořený jedno-membránovými agenty, reprezentací prostředí a chováním agentů připomíná tradiční multiagentní systémy, které využívají algoritmy aplikované v praxi – například pro řešení optimalizačních úloh. Jedním z takových multiagentních systémů je tzv. smečkový algoritmus inspirovaný chováním vlků v přírodě, jejich hierarchií a způsobem lovu. V našem příspěvku představíme jeden z nových úhlů pohledu na chování P kolonií a algoritmů inspirovaných chováním vlků nazývaných smečkové algoritmy. Poukážeme na okolnosti související právě se simulací a porovnáním těchto dvou poměrně odlišných biologicky inspirovaných systémů, a to diskrétně založeného systému a systému využívajícího spojitých funkcí.

1 Úvod

V teoretické informatice se setkáváme s celou řadou výpočetních modelů. V našem článku budeme studovat vlastnosti P kolonií, které představují jeden z biologicky motivovaných výpočetních modelů, spojující dvě oblasti teoretické informatiky, kolonie (více v Kelemen a Kelemenová (1992)) a membránové systémy (např. Păun a spol. (2010)). P kolonie byly zavedeny v roce 2004 v práci Kelemen a spol. (2004).

Do prostředí P kolonie umístíme agenty představující organismy „žijící“ v P koloniích. Každý agent je tvořen pouze jednou membránou ohraničující oblast s objekty. Jako objekt si můžeme představit látky, které organismy mění, přijímají, či vylučují do prostředí. V každém okamžiku jsou všechny objekty uvnitř agenta změněny nebo přesunuty. Počet objektů v každém agentu je pevně dán a je neměnný v průběhu celého výpočtu, je také stejný pro každého agenta. Počet objektů je parametrem P kolonie, který nazýváme kapacitou.

V prostředí se na začátku výpočtu nachází pouze kopie speciálního objektu nazývaného environmentální. Tohoto objektu je dostatečné množství pro výpočet systému. Nemůže se stát, že by ho bylo nedostatek. V přírodě si pod environmentálním objektem můžeme představit vzduch nebo vodu v závislosti na tom, o jakých organismech budeme uvažovat.

Každý agent má svou množinu programů, které jsou tvořeny pravidly. Počet pravidel v programu je shodný s počtem objektů uvnitř agenta. Agent je určen svým stavem (obsaženými objekty) a svou množinou programů. Programy určují činnost agenta, v přeneseném slova smyslu jeho životní projevy.

Přestože agenty můžeme chápat jako nezávislé organismy, mohou svou činnost navzájem ovlivňovat, a to prostřednictvím prostředí, které představuje komunikační kanál. Agent do něj může umístit objekty odpovídající jeho stavu v daném okamžiku a tím ovlivnit činnost ostatních agentů.

V tomto příspěvku se budeme věnovat také dalšímu přírodou motivovanému modelu nazývaného smečkové algoritmy. Daný model je inspirovaný chováním vlků, jejich vzájemnou hierarchií a způsobem zajišťování potravy. Uvedené algoritmy byly publikovány v práci Mirjalili a spol. (2014). Vstupem algoritmu je funkce reprezentující prostředí, v němž jsou umístění náhodně agenti. Změna ohodnocení a pohyb agentů probíhá pomocí fitness funkce. Mezi agenty je hierarchické uspořádání podobně jako u vlků a to Alpha, Beta, Delta a Omega. Pohyb agentů je závislý na pozici třech nejvýše hierarchicky postavených jedinců.

P kolonie i smečkové algoritmy představují multiagentní systémy.

Efektivitu určitého algoritmu z pohledu časové nebo paměťové složitosti posuzujeme ve vztahu ke konkrétnímu výpočetnímu modelu. Použití výpočetních

modelů nám umožňuje objektivně sledovat výkon, efektivitu algoritmů, a to nezávisle na specifických vlastnostech konkrétní implementace. Algoritmus může být velmi efektivní pro určitý výpočetní model, zatímco na jiném modelu může fungovat jen omezeně, pomaleji, nebo nemusí být realizovatelný vůbec.

Ačkoli v praxi nás zajímá zejména uplatnění modelu na soudobém hardware - elektronickém počítači, z hlediska vědy a budoucnosti jsou zajímavé také abstraktní výpočetní modely, pro které nemáme k dispozici skutečný stroj, který by jejich činnost realizoval. V oblasti teoretické informatiky je abstraktních výpočetních modelů celá řada. Takový model může být například daleko „paralelnější“, než může být v současné době ten nejlepší skutečný počítač. Takové modely jsou obvykle výpočetně nebo paměťově velmi náročné a v současné době nedokážeme využít jejich plný praktický potenciál. V mnoha případech je ale možná takzvaná počítačová simulace takového modelu, která umožní sledovat a zkoumat chování daného modelu.

Musíme si uvědomit, že se jedná primárně o výpočetní modely inspirované přírodou, nikoli biologické modely simulující chování určitých organismů. Vzhledem k probíhajícímu výzkumu buněk, buněčných struktur, DNA výpočtů, kvantových výpočtů a nanotechnologií je zde jistá pravděpodobnost jejich implementace na vysoce paralelním neelektrickém (biologickém) substrátu. V syntetické biologii se už v současné době můžeme setkat se syntetizovaným genomem a programovým chováním. V případě naplnění představy o vytvoření masivních nedeterministických „in vivo“ (vnitro buněčných) modelů paralelních procesorů inspirovaných například membránovými výpočty, by to znamenalo významné zvýšení rychlosti zpracovávání informací paralelním způsobem a možnost využívat nedeterministických postupů.

V našem příspěvku se zaměříme na porovnání dvou zcela odlišných biologicky inspirovaných modelů, membránových systémů a smečkových algoritmů. Zaměříme se na možnosti a omezení membránových systémů pro simulaci tradičních přístupů v oblasti přírodou inspirovaných optimalizačních algoritmů, které jsou již využívány při řešení složitých úloh, jako jsou právě smečkové algoritmy.

2 P kolonie a 2D P kolonie

Jak jsme již uvedli, P kolonie jsou jednou z variant výpočetních modelů ze skupiny membránových systémů a inspirují se strukturou a činností živých organismů, kteří spolu žijí ve společném prostředí, jako jsou například mravenci nebo včely. U mravenců mezi nejdůležitější komunikační prostředky patří mravenčí pachy, tzv. feromony. I přes malý mozeček, který mají, jsou schopni vytvořit poměrně složité společenské struktury. Jako druhý příklad jsme uvedli včely. Žihadllový aparát včely tvoří několik žláz. V jejich výměšcích je histamin, fosfolipáza a hyaluronidáza vyvolávající nepříjemné pocity svědění a pálení v okolí vpichu. Tyto žlázy vylučují i feromony, které slouží k upozornění na nebezpečí. Do rány vpichu vstřikují včely i látku, pomocí které se navigují ostatní včely a vpichují další žihadla v blízkosti prvního bodnutí. Kromě jedu tedy včely vpichují i dávku izoamylacetátu. Včely z jednoho úlu se poznají dle pachu, který se skládá z mnoha složek. Pyl a nektar dvou úlů nevoní stejně. Povrch těla včely je uzpůsoben tak, že nabírá vůni úlu a udržuje si ji. Včely značí cestu k nektaru, to je i nápomocno k nalezení cesty zpátky k úlu. Chemické látky tedy slouží k dorozumívání včel a umožňují fungování včelstva. Komunikují prostřednictvím prostředí, ve kterém žijí.

Model P kolonie se skládá z jednoduchých agentů, umístěných do společného prostředí a představují multiagentní systém. Prostředí slouží jako komunikační kanál a také jako skladiště objektů.

Každý agent je tvořen jednou membránou, která ohraničuje oblast s objekty. Počet objektů je shodný pro všechny agenty. Činnost agentů je dána programy. Program je tvořen pravidly a obsahuje tolik pravidel, kolik je objektů uvnitř agenta. Existují tři základní typy pravidel a to přepisující, komunikační a řídící pravidla.

Nyní si uvedeme tvary uvedených pravidel a pokusíme se je alespoň z části interpretovat v přírodě.

Přepisující pravidla mají tvar $a \rightarrow b$. Aplikací pravidla je objekt a přepsán na objekt b. Dané pravidlo umožňuje změnu látky a na látku b uvnitř organismu.

Komunikační pravidla mají tvar $c \leftrightarrow d$. Pomocí tohoto pravidla je objekt c, který je uvnitř agenta, přesunut ven, a objekt d, který je vně, je přesunut dovnitř agenta. Komunikační pravidla umožňují komunikaci organismů – agentů. Jedna látka c je vyloučena do prostředí a další látka d je přijata organismem. Objekt d můžeme klidně i chápat jako zpětnou vazbu při vyloučení objektu c.

Pomocí **řídících pravidel** dáváme agentům možnost výběru mezi dvěma možnostmi. Řídící pravidla mají tvar $\langle a \rightarrow b, c \leftrightarrow d / c' \leftrightarrow d' \rangle$. Pravidlo skládá ze dvou částí, komunikačních pravidel. Pokud je řídící pravidlo aplikováno, má část $c \leftrightarrow d$ vyšší prioritu k provedení než část $c' \leftrightarrow d'$. To znamená, že agent vybere komunikační pravidlo $c \leftrightarrow d$ (pokouší se najít uvnitř objekt c a objekt d v prostředí). Pokud toto pravidlo může být vykonáno, tak je použito. Když první pravidlo nemůže být provedeno, agent použije druhé pravidlo z dané dvojice pravidel – $c' \leftrightarrow d'$. U řídících pravidel organismus zjišťuje, jestli se v prostředí vyskytuje určitý typ látky, a pokud ne, přizpůsobí své chování tomuto nedostatku. Například pokud mravenec nenajde danou látku, ztratí feromonovou stopu, bude ji dále hledat, a pokud ji najde, bude ji sledovat.

Výpočet P kolonie začíná v počáteční konfiguraci, která je definována následujícím způsobem: prostředí a všichni agenti obsahují pouze kopie objektů e, kde e je environmentální objekt. Aplikováním programů mohou agenti měnit svůj obsah a pomocí prostředí mohou ovlivňovat chování ostatních agentů v dalších krocích výpočtu. V přírodě se také často prostředí využívá jako komunikační kanál. Některé organismy vylučují jisté látky do prostředí a tím ovlivňují chování ostatních. V každém kroku výpočtu každý agent nedeterministicky vybere jeden ze svých aplikovatelných programů a vykoná jej. Výpočet končí zastavením, kdy žádný agent nemůže aplikovat žádný ze svých programů. Výsledkem výpočtu je počet určitých objektů v prostředí na konci výpočtu. Z důvodu nedeterminismu v průběhu výpočtu můžeme získat několik výpočtů, které končí zastavením.

P kolonie jsou výpočetně úplné. Zajímavá je otázka, jaký je nejmenší počet agentů a počet programů uvnitř agenta při zachování výpočetní úplnosti. Cílem tohoto článku není výpočetní síla uvedených systémů, proto je nebudeme dále rozvádět. Pro získání informací o výpočetní síle a vlastnostech P kolonií doporučujeme čtenáři článek Ciencialová a spol. (2019). Více informací o membránových systémech čtenář nalezne v knize Păun a spol. (2010).

Pokud rozšíříme původní model P kolonií o dvourozměrné prostředí tak, aby agenti neměli přímý přístup ke všem objektům a museli se po dosažení cíle pohybovat, budeme daný model nazývat 2D P kolonie. Dané rozšíření umožní právě použití pro modelování reálných situací a tím i predikci budoucího stavu reálného prostředí. Například v práci Cienciala a spol. (2014) se autoři věnovali simulaci bleskových povodní pomocí 2D P kolonií. V reálném světě (stejně tak i v kybernetickém světě) se liší koncentrace látek v závislosti na místě a živé organismy nemají možnost vědět, co je „za horizontem“. Tyto úvahy nás inspirovaly k zavedení nového modelu P kolonií, kde jsou agenti umístěni ve 2D kartézské mřížce. Agenty tedy umístíme do této mřížky a „pohled“ agenta omezíme na buňky bezprostředně jej obklopující. Na základě obsahu těchto buněk je ovlivněna poloha agenta v mřížce v dalším kroku výpočtu. Obsah agenta je také limitován dvěma objekty, kolonie má tedy kapacitu 2.

A jak se změní pravidla? Budeme vycházet z původního modelu P kolonií, tak jak jsme je popsali. Program agenta může být tedy sestaven ze dvou typů pravidel přepisujících a komunikačních s tím rozdílem, že komunikační pravidlo bude mít navíc vliv pouze na místo, kde se právě agent nachází.

Přidáme nový typ pravidel pro pohyb agenta ve 2D prostředí. Podmínkou pro pohyb agenta je nalezení konkrétních objektů v konkrétních místech prostředí. To je určeno pomocí matice základních objektů. Agent hledá nejvýše jeden objekt v každé buňce okolí. Pokud takové objekty nalezne, přemístí se o jednu buňku nahoru, dolů, doleva nebo doprava. Řídící pravidlo je nahrazeno pravidlem pohybovým. Pro jednoduchost chování agenta stanovíme ještě jednu podmínku: pokud agent mění polohu, pak nemůže komunikovat s prostředím. To znamená, že pokud program obsahuje pravidlo pro pohyb, tak druhým pravidlem musí být pravidlo přepisovací.

Nyní uvedeme formální definici 2D P kolonie.

Definice: 2D P kolonie je struktura:
$\Pi = (A, e, Env, B_1, \ldots, B_k, f), k \geq 1$, kde

- A je abeceda kolonie, její prvky nazýváme objekty,
- $e \in A$ je základní objekt prostředí 2D P kolonie, který nazýváme environmentální
- Env je dvojice $(m \times n, w_E)$, kde $m \times n, m, n \in N$ je matice o velikost $m \times n$ multimnožin objektů nad množinou $A - \{e\}$, $m, n \in N$ je velikost prostředí a w_E je počáteční obsah prostředí.
- $B_i,\ 1 \leq i \leq k$, jsou agenti, každý agent je struktura $B_i = (O_i, P_i, [r_i, s_i])\ ,\ 0 \leq o \leq m,\ 0 \leq p \leq n$, kde:
  - O_i je multimnožina nad množinou A, určuje počáteční stav (obsah) agenta, $|O_i| = 2$,
  - $P_i = \{p_{i,1}, \ldots, p_{i,l_i}\}, l \geq 1, 1 \leq i \leq k$ je konečná množina programů, kde každý program obsahuje právě 2 pravidla v jedné z následujících forem:
    - $a \rightarrow b$, zvané vývojové pravidlo, $a, b \in A$,
    - $c \leftrightarrow d$, zvané komunikační pravidlo, $c, d \in A$,
    - $[a_{q,r}] \rightarrow s, 0 \leq q, r \leq 2, s \in \{\Leftarrow, \Rightarrow, \Uparrow, \Downarrow\}$, zvané pohybové pravidlo,
  - $[r_i, s_i]$ je počáteční pozice agenta B_i ve 2D prostředí,
    $0 \leq r_i \leq m-1,\ 0 \leq s_i \leq n-1,\ 1 \leq i \leq k$,
- $f \in A$ je konečný objekt kolonie.

Konfigurace 2D P kolonie je dána stavem prostředí, tedy maticí typu $m \times n$ tvořenou multimnožinou objektů z $A - \{e\}$, a stavem všech agentů, tedy páry jejich objektů z abecedy A a jejich souřadnicemi v prostředí. Počáteční konfigurace je dána definicí 2D P kolonie.

Výpočetní krok se skládá ze tří částí. První část spočívá ve stanovení aplikovatelné množiny programů v závislosti na aktuální konfiguraci P kolonie. Existují programy patřící do množin programů všech agentů. Ve druhé části je u každého agenta vybrán jeden program patřící do množiny aplikovatelných programů. Neexistuje kolize mezi komunikačními pravidly různých programů. Ve třetí části jsou vykonány vybrané programy.

Vykonáním programů se změní konfigurace kolonie. Změna konfigurace je založena na změně stavu prostředí, obsahu a umístění agentů.

Výpočet probíhá nedeterministicky a maximálně paralelně a končí zastavením, kdy žádný agent nemá aplikovatelný program.

Výsledek výpočtu je počet kopií finálního objektu umístěného v prostředí na konci výpočtu.
Cílem výzkumu 2D P kolonií je sledovat jejich chování (změny konfigurací, stavů agentů), nikoli výpočetní sílu. Můžeme definovat několik pohledů, jak posoudit dynamiku výpočtu:

- počet pohybů agentů,
- počet „navštívených“ buněk (nebo „nenavštívených“ buněk),
- počet kopií určitého objektu v „domovské“ buňce nebo v celém prostředí.

Navíc můžeme uvažovat, zda daný počet sledujeme po jednotlivých krocích výpočtu nebo až na konci výpočtu.

3 Vlci a smečkový algoritmus

Nyní se na chvíli oprostíme od membránových systémů a zaměříme jednu z metod optimalizačních algoritmů, a to na takzvaný smečkový algoritmus.

Smečkový algoritmus, anglicky zvaný Grey wolf optimization algorithm, je zavedenou metodou pro řešení problémů v oblasti globální optimalizace. Vstupem je multidimenzionální matematická funkce a cílem je nalezení globálního extrému, tedy nejvyšší nebo nejnižší bod této matematické funkce.

U problému nalezení globálního optima se ještě pozastavíme. Ačkoli se tento problém může zdát na první pohled triviální, v případě některých funkcí tomu tak není. Existují funkce velmi rozměrné, které mají mnoho extrémů nebo nejsou diferencovatelné. Taková matematická funkce může představovat velmi složitý problém ze skutečného světa, který potřebujeme vyřešit v určitém časovém úseku, přičemž obecně známé deterministické přístupy a algoritmy selhávají – nemáme dostatek času projít všechna možná řešení.

Potřeba řešit složité problémy v uspokojivém čase vedla k výzkumu nových metod založených na optimalizaci. Řada z nich je odvozena heuristicky a motivuje se přírodou. Výstupem takových algoritmů sice nemusí být naprosto přesné řešení, ale zato rychlé a přitom dostatečně přesné. Smečkový algoritmus je jednou z úspěšných a praxí již ověřených metod optimalizace, který se inspiruje společenským soužitím vlků obecných, zejména jejich způsobem lovu a utvářením hierarchie ve smečce. Byl zaveden v roce 2014 a jeho autory jsou Seyedali Mirjalili, Seyed Mohammad Mirjalili a Andrew Lewis (Mirjalili a spol., 2014).

Na proces lovu vlčí smečkou můžeme analogicky nahlížet jako na proces řešení optimalizačního problému. Chování vlků při lovu a vytváření hierarchie dokážeme matematicky popsat a modelovat v multiagentním systému. Vlci (agenti) se v prostředí (matematická funkce) pohybují za účelem nalézt co nejvydatnější kořist (globální extrém funkce).

V hierarchii vlčí smečky rozlišujeme vlky Alpha, Beta, Delta a Omega. Vlci Alpha jsou v hierarchii postavení nejvýše a obvykle tvoří dominantní pár (samec a samice). Jsou vůdci smečky a ostatní vlci ve smečce je plně respektují. Vlci Beta podporují Alpha pár při rozhodovacích činnostech a poskytují jim zpětnou vazbu. Delta vlci se dále dělí na skauty, strážce a ošetřovatelé, plní rozkazy výše postavených vlků a starají se o běžný chod smečky. Vlci omega jsou takzvaní „obětní beránci“, mají například právo jíst až jako poslední a ostatní vlci si na nich mohou „vylít zlost“. Mají však v hierarchii důležitou roli, a to filtrovat agresi a udržovat tím smečku pohromadě.

Také smečkový algoritmus rozlišuje agenty (vlky) typu Alpha, Beta, Delta a Omega. Pozice v prostředí, kde se agent v danou chvíli nachází, představuje jedno z možných řešení daného problému. Tři nejlepší agenti podle fitness funkce, která kvantitativně vyjadřuje kvalitu řešení každého z nich vzhledem k hledanému extrému funkce, označujeme jako agenty Alpha, Beta a Delta. Předpokládáme, že globální extrém (potrava) se nachází někde mezi nimi, což agenti dále zohledňují při svých pohybech. Agenti Alpha, Beta a Delta jsou si v případě smečkového algoritmu rovnocenní a jejich pozice mají stejný význam.

Lov probíhá v několika fázích. Nejprve vlci pátrají po co nejvydatnější potravě s ohledem na úsilí, které musí k jejímu ulovení vynaložit. Následně kořist pronásledují, snaží se ji zahnat do kouta, nebo oddělit jedince od stáda. V další fázi kořist obklučují z různých stran, aby neměla kam utéct. Jakmile kořist obklíčí, útočí na ni z různých směrů a zaměřují se na její slabá místa. Poté, co se kořist vysílí a přestane se bránit, vlci ji zadáví.

Analogicky k tomuto principu algoritmus plynule přepíná mezi fázemi průzkumu a lovu a používá k tomu dva vektory $\vec{A}$ a $\vec{C}$, jejichž prvky mají náhodné hodnoty v rozmezí od -2 do 2. Počet prvků obou vektorů je roven dimenzi prostředí a každý prvek ovlivňuje směr pohybu vlka v konkrétním rozměru prostředí. Prvek vektoru s hodnotou v rozsahu od -1 do 1 přinutí vlky spíše lovit, zatímco jinak se snaží intenzivně procházet okolí a vyhnout se tak nalezení pouze lokálního optima.

Prvky vektoru $\vec{A}$ oproti vektoru $\vec{C}$ navíc závisí na aktuální iteraci algoritmu – s přibývajícími iteracemi algoritmu se zvyšuje pravděpodobnost, že se hodnoty prvků vektoru přibližují k hodnotě 0, což přinutí vlky (agenty) více lovit a méně prozkoumávat okolí.

Prvky vektoru $\vec{C}$ jsou naopak čistě náhodné. Tento vektor simuluje překážky v prostředí a umožňuje v malé míře prozkoumávat okolí i v posledních iteracích algoritmu, kdy většina agentů volí spíše fázi lovu.

Vstupem algoritmu je matematická funkce, která reprezentuje daný problém, který chceme vyřešit. Tato funkce představuje prostředí, do něhož jsou náhodně

umístění agenti. Jedná se tedy o multiagentní systém. Chování agentů v tomto prostředí určitým způsobem simuluje chování skutečných vlků. Algoritmus pracuje v iteracích, přičemž v každé z nich jsou provedeny následující kroky:

1. Ohodnocení agentů pomocí fitness funkce,
2. Určení sociální hierarchie,
3. Pohyb agentů v prostředí na základě pozic tří nejlepších vlků (agentů) v prostředí podle fitness funkce (v závislosti na hodnotách prvků vektorů $\vec{A}$ a $\vec{C}$ pro konkrétní dimenzi prostředí: ve fázi lovu směrem k potravě, ve fázi průzkumu naopak dále od potravy),
4. Kontrola ukončovačího kritéria.

Ve fázi lovu mají agenti tendenci se k potravě přibližovat z různých směrů a dochází tak k jevu obkličování, podobně jako v případě skutečných vlků.

4 Porovnání obou modelů

Oba modely se inspirují v přírodě, a to společenským soužitím vybraných živočichů, avšak v mnohém se značně liší. Zatímco 2D P kolonie pracují s diskrétním prostředím, smečkový algoritmus se spojitou matematickou funkcí. Agenti 2D P kolonie spolu komunikují výhradně pomocí prostředí, zatímco u smečkového algoritmu agenti při svých pohybech využívají informace o pozicích dalších agentů. Takovou informaci agenti membránových systémů nemají k dispozici. Smečkový algoritmus dále silně pracuje s náhodností a pravděpodobností, což 2D P kolonie umožňují jen v omezené míře, a to nedeterministickým výběrem z množiny pravidel pro jednu a tu samou konfiguraci.

Abychom eliminovali rozdíly obou modelů, zavedli jsme nový model 2D P kolonie, jehož prostředí dokáže uchovávat číselné hodnoty uvnitř objektů matice. Model jsme dále rozšířili o takzvanou tabuli, která agentům umožňuje sdílet například informace o nalezených číselných hodnotách v prostředí.

Definice: 2D P kolonie s tabulí, přizpůsobená pro simulaci smečkového algoritmu, je struktura: $\Pi_{gw} = (A, e, Env, B_1, \ldots, B_k, f), k \geq 1$, kde

- $V = \left\{\boxed{\ }, \boxed{\ }', e, b, m, n, f\right\}$,
- $e \in V$ je základní objekt prostředí,
- Env je trojice $(i \times j, w_e, f\,(x, y))$, kde $i, j \in \mathbb{N}$, $w_E = |a_{r,s}|, a_{r,s} = \varepsilon, 1 \geq r \geq i, 1 \geq s \geq j$,
- $A_1, A_2, \ldots, A_k$ jsou agenti, $A_i = (O_i, P_i, [r_x, r_y])$, kde:
  - $|o_i| = 2$,
  - $P_1 = P_2 = \cdots = P_k$, P_i jsou pravidla definovaná níže,
  - $[r, s]$ jsou počáteční souřadnice,
- BB je tabule, kterou popíšeme v dalším textu,
- f je konečný objekt, $f \in V$.

Všimněme si rozdílů oproti definici 2D P kolonie uvedené v druhé kapitole. Abeceda A zahrnuje speciální box-objekt $\boxed{\ }$, který umožňuje uvnitř sebe uchovávat číselné hodnoty. Prostředí env obsahuje navíc funkci $f(x, y)$, která každému bodu v matici prostředí přiřazuje konkrétní číselnou hodnotu podobně, jako je tomu v případě fitness funkce u smečkového algoritmu. V neposlední řadě zde máme navíc tabuli BB (od anglického slova *blackboard*), která je dostupná kdykoli pro čtení a zápis všem agentům a kterou podrobně popíšeme později.

Další vlastnosti takto upravené 2D P kolonie zůstávají neměnné. Konfigurace je dána stavem prostředí a stavem všech agentů. Výpočetní krok spočívá opět v aplikaci pravidel programů určených pro danou konfiguraci. Výpočet je nedeterministický a maximálně paralelní. Výpočet ukončíme ve chvíli, kdy se již agenti nepohybují a zároveň nedochází k dalšímu (průběžnému) zlepšování hodnot v tabuli – agenti Alpha, Beta a Delta mají vždy k dispozici program, který mohou aplikovat, výpočet tedy může probíhat libovolně dlouho. Výsledkem výpočtu jsou pozice agentů v prostředí a stav tabule na konci výpočtu.

Pravidla pro takto upravenou 2D P kolonii jsme navrhli s cílem simulovat činnost smečkového algoritmu. Počáteční konfigurace agenta je ee a jeho pozice v prostředí je $[r, s]$. V následujících odrážkách uvedeme příklad některých pravidel, úplný přehled pravidel je pak dostupný v publikaci Valenta a spol. (2021).

1. $\left\langle e \rightarrowtail \boxed{x}'; e \rightarrow \mathit{Get}(BB[\mathit{alpha}, \boxed{\ }], \right)\rangle$ $x \in \mathbb{R}$ je číslo umístěné v prostředí na pozici $[r, s]$, *alpha* je odkaz na hodnotu v tabuli vyhrazenou pro zápis alfa agenta do tabule; Tento program umožňuje číst hodnotu z prostředí a hodnotu vlka alfa z tabule.
2. $\left\langle x > y : \boxed{x} \rightarrow \boxed{x}, \boxed{y}' \rightarrow A \right\rangle$ – Tento program agent použije pro porovnání své pozice s hodnotami v tabuli a zařadí se do hierarchie. Symbol A v tomto případě reprezentuje vlka Alpha, kterým se agent stane v případě, je-li jeho hodnota vyšší než ta v tabuli, ale obdobné pravidla máme také pro agenty Beta, Delta a Omega.
3. Je-li jeden z vnitřních objektů agenta ekvivalentní s A, tedy agent se zařadil do hierarchie na pozici Alpha, použije se program:
   $\langle \mathit{Update}(\boxed{x}, BB[\mathit{alpha}]), A \rightarrow a' \rangle$,
   Funkce $Update$ je funkcí tabule, která slouží pro

aktualizaci konkrétní hodnoty v tabuli. První argument funkce je objekt agenta k zapsání do tabule, druhý argument pak určuje pozici v tabuli, kam se má daná hodnota zapsat. Obdobné pravidla používají také agenti Beta a Delta, kteří svou hodnotu zapisují do tabule na pozice *BB[beta]* a *BB[delta]*.

4. Agenti Omega (nejníže postavení v hierarchii) do tabule nezapisují. Namísto toho se pohybují v prostředí směrem o jednu pozici nahoru, dolů, vlevo, nebo vpravo. Agent postupně zkouší jednotlivé směry v náhodném pořadí. Agent zvolí daný směr v případě, pokud se daným pohybem nevzdálí od přibližné pozice potravy, která je vypočtena tabulí na základě pozic agentů Alpha, Beta a Delta a nachází se někde mezi nimi. V opačném případě zvolí jiný směr, který dosud nevyzkoušel.

Podrobný popis všech pravidel systému je k dispozici v publikaci Valenta a spol. (2021).

Tabule je struktura

$$BB_{GWO} = ((fnc, rcv), [\overrightarrow{v1}, \overrightarrow{v2}]),$$

kde dimenze obou vektorů $\overrightarrow{v1}$, $\overrightarrow{v2}$ je $j = \max(7, k)$, $k \geq 1$ je počet agentů.

Vektor $\overrightarrow{v1}$ je vektor s prvky pojmenovanými následovně: *AlphaValue, BetaValue, DeltaValue, AlphaPosition, BetaPosition, DeltaPosition, preyPosition*. Je-li $j > 7$, prvky s indexem větším než 7 jsou již bez jména. Počáteční obsah každé z těchto hodnot je 0.

Vektor $\overrightarrow{v2}$ je vektor s prvky pojmenovanými následovně: $A'_0 sDistanceFromPrey$, $A'_1 sDistanceFromPrey$, ..., $A'_k sDistanceFromPrey$, k je počet agentů.
Tyto prvky vektoru slouží k uchovávání hodnot vyjadřujících vzdálenost jednotlivých agentů od tabule, a jsou vypočteny tabulí po každém použití některé z funkcí tabule provádějících čtení nebo zápis. Počáteční obsah každé z těchto hodnot je 0.

Tabule řeší rozdílný způsob komunikace agentů u obou modelů. Umožňuje agentům kdykoli sdílet své fitness hodnoty s dalšími agenty, což je podstatná informace, kterou potřebují k zařazení se do hierarchie.

Pro komunikaci agentů s tabulí má tabule vestavěné funkce Get pro čtení a $Update$ pro zápis. Při každém použití některé z funkcí agentem je vypočtena vzdálenost agenta od prostředí pomocí takzvaných přijímačů, které fungují na podobném principu, jako navigační systém GPS. Dva přijímače neustále rotují kolem prostředí a naslouchají signálům agentů. Výsledná pozice agenta je vypočtena jako průnik dvou kružnic se středem v přijímačích a poloměrem odpovídajícím době putování signálu od agenta. Dobu putování signálu zaokrouhlujeme, čímž vzniká odchylka, která plní funkci určité náhodnosti, podobně jako je tomu v případě náhodných vektorů u smečkového algoritmu.

5 Využití 2D P kolonie s tabulí pro řešení optimalizačních úloh

Implementaci modelu popsaného v předchozí kapitole jsme popsali v publikaci Valenta a spol. (2021). Implementace umožnila model testovat v porovnání se smečkovým algoritmem a sledovat jeho chování. Testování ukázalo, že lze pomocí modelu 2D P kolonie s tabulí úspěšně simulovat smečkový algoritmus. Má to ale určitá omezení.

Velikost kroku agenta je v případě 2D P kolonie omezena vždy na jednu pozici směrem vlevo, vpravo, nahoru, nebo dolů, zatímco smečkové algoritmy umožňují pohyb agentů téměř bez omezení, a to v libovolném směru až do dvojnásobku vzdálenosti od přibližné pozice od potravy. V případě 2D P kolonie s tabulí se navíc pohybují jen agenti Omega, zatímco s případě smečkového algoritmu se v omezené míře pohybují i Alpha, Beta a Delta agenti.

Velikost kroku agenta se negativně projevuje také v případě rozměrných prostředí, kdy agentům 2D P kolonie s tabulí trvá daleko déle, než se dostanou z jednoho místa na druhé.

Model 2D P kolonie je také méně odolný vůči konvergenci k lokálnímu optimu. Pokud agent nalezne lokální extrém a všechny jeho okolní pozice mají horší (nižší) fitness hodnoty, nemá již důvod měnit svou pozici - agenti totiž při svých pohybech zohledňují jen pozice v jejich bezprostředním okolí. Toto omezení dále souvisí s funkcí, která reprezentuje prostředí (daný problém), a kterou je nutné nejprve vhodným způsobem diskretizovat vzorkováním. Popsaný problém se projevuje, jsou-li ve výsledné matici větší plochy se stejnými hodnotami (nedochází k průběžnému zlepšení nalezených hodnot).

Přes tato úskalí se nám podařilo ukázat, že i takto jednoduchý model, jako 2D P kolonie, dokáže po menších úpravách řešit optimalizační problémy na vhodně zvolených a diskretizovaných funkcích.

Předpokládáme, že uvedený model 2D P kolonie můžeme po menších úpravách využít také pro simulaci dalších optimalizačních algoritmů, jako jsou mravenčí kolonie nebo algoritmus hejna částic. Uplatnění si dokážeme představit také ve skutečném multiagentním systému, kde může daný model sloužit například pro řízení činnosti robotů prohledávajících prostředí.

6 Závěr

V tomto článku jsme si popsali dva zcela odlišné modely teoretické informatiky, a to membránové systémy, z nichž jsme si zvolili 2D P kolonie, a optimalizační algoritmy, z nichž jsme si zvolili smečkový algoritmus. Oba modely jsme porovnali a poukázali na odlišnosti, které znemožňují simulaci smečkového algoritmu pomocí 2D P kolonií. Také jsme představili model 2D

P kolonie s tabulí a možností uchovávat číselné hodnoty v takzvaných box-objektech, který umožní simulovat nejen smečkový algoritmus, ale také další optimalizační algoritmy pracující na podobných principech. To může vést k dalšímu výzkumu v oblasti využití membránových systémů pro řešení optimalizačních úloh.

Poděkování

Tento příspěvek vznikl za podpory Slezské univerzity v Opavě v rámci grantu Student Funding Scheme, projekt SGS/8/2022.

Reference

Cienciala, L., Ciencialová, L. a Langer, M. (2014). Modelling of surface runoff using 2D P colonies. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 8340 LNCS:101–116.

Ciencialová, L., Csuhaj-Varjú, E., Cienciala, L. a Sosík, P. (2019). P colonies. *Journal of Membrane Computing*, 1(3):178–197.

Kelemen, J. a Kelemenová, A. (1992). A grammar-theoretic treatment of multiagent systems. *Cybern. Syst.*, 23(6):621–633.

Kelemen, J., Kelemenová, A. a Păun, Gh. (2004). Preview of P colonies: A biochemically inspired computing model. V *Workshop and Tutorial Proceedings. Ninth International Conference on the Simulation and Synthesis of Living Systems (Alife IX)*, str. 82–86, Boston, Massachusetts, USA.

Mirjalili, S., Mirjalili, S. M. a Lewis, A. (2014). Grey wolf optimizer. *Advances in Engineering Software*, 69:46–61.

Păun, Gh., Rozenberg, G. a Salomaa, A. (2010). *The Oxford Handbook of Membrane Computing*. Oxford University Press, Inc., New York, NY, USA.

Păun, Gh. (2000). Computing with membranes. *J. Comput. Syst. Sci.*, 61(1):108–143.

Valenta, D., Langer, M., Ciencialová, L. a Cienciala, L. (2021). On numerical 2d p colonies with the blackboard and the gray wolf algorithm. Freund, R., Ishdorj, T.-O., Rozenberg, G., Salomaa, A. a Zandron, C. (zost.), V *Membrane Computing*, str. 161–177, Cham. Springer International Publishing.