Inteligencja roju

W pewnym momencie wszyscy musieliście obserwować zachowanie mrówek. Sposób, w jaki poruszają się w skoordynowanej linii jedna za drugą, sposób, w jaki zbierają i przenoszą żywność (większą niż ich rozmiar) do swoich gniazd, sposób, w jaki tworzą mosty, aby zakryć większe szczeliny. Wszystkie te zachowania są niezwykłe, biorąc pod uwagę fakt, że mózgi tych małych stworzeń nie znajdują się w pobliżu ludzkiego mózgu pod względem liczby neuronów, a tym samym połączeń. Tego rodzaju porządkowanie jest nieodłącznym elementem naturalnych procesów i jest regulowane w niezwykły sposób. Należy tutaj zauważyć, że owady te są bardzo małe i ich indywidualna zdolność do osiągania tak dużych celów nie jest indywidualna. Jednak gdy pracują w grupie, są w stanie osiągnąć tak większe cele. W świetle tego owady te nazywane są również owadami społecznymi. Owady społeczne mają pewne znaczące cechy. Żyją w koloniach, mają podział pracy, mają silne interakcje grupowe (bezpośrednie lub pośrednie) i są elastyczne. Wszystkie te zachowania są stosowane w celu osiągnięcia zbiorowej inteligencji grupy. Ten rodzaj zjawisk skłonił naukowców do pracy nad nowym sposobem uzyskania sztucznej inteligencji (AI) o nazwie inteligencja roju (SI). Termin SI został po raz pierwszy wymyślony przez Gerardo Beni i Jing Wanga w 1989 roku w kontekście robotycznych systemów komórkowych. To dziedzina sztucznej inteligencji inspirowana naturalnym zachowaniem i skoordynowanym funkcjonowaniem mniejszych owadów, takich jak mrówki, pszczoły i termity. Dla każdego układu SI istniałaby kolonia prostych czynników (takich samych jak pojedyncza mrówka w kolonii mrówek), które są również nazywane boidami. Każda z tych platform będzie wchodzić w interakcje z sąsiadem i jego otoczeniem (kontekstami), aby osiągnąć indywidualne cele. Razem osiągają jeden większy cel, jakim jest rozwiązanie problemu. Pomysł SI spodobał się badaczom, którzy badają go bardziej w celu zastosowania go w rozwiązywaniu rzeczywistych problemów. W dzisiejszym świecie, w którym napływ informacji jest niekontrolowany, staranne obchodzenie się z takimi informacjami nie jest już możliwe w ramach pojedynczego ludzkiego mózgu lub pojedynczego scentralizowanego systemu z powodu stale rosnącej ilości danych. Zawsze jesteś ograniczony indywidualną zdolnością rasy ludzkiej lub sprzętem maszynowym. SI pojawia się jako alternatywa, w której przetwarzanie informacji jest rozproszone, autonomiczne i naturalnie kontrolowane. W kilku następnych częściach będziesz miał większą jasność na temat tego, jak SI rozwiązuje niektóre złożone problemy w świecie rzeczywistym. W tym rozdziale zajmiemy się następującymi tematami:

* Przegląd inteligencji roju

* Model optymalizacji roju policji

* Model optymalizacji kolonii mrówek

* Biblioteka Masona

* Biblioteka Opt4J

* Aplikacje w analityce dużych zbiorów danych

* Obsługa danych dynamicznych

* Optymalizacja wielu celów

Inteligencja roju

Inteligencja roju jest inspirowana zachowaniem grupowym gatunków takich jak mrówki, termity i pszczoły. U tych gatunków zachowanie grupy w celu osiągnięcia wspólnych większych celów wykracza poza możliwości osób należących do grupy. Jednak każda osoba w ograniczonej zdolności, jak na swoje możliwości, pomaga w osiągnięciu wspólnego zachowania grupy. Jako grupa, gatunki te zachowują się

inteligentnie, bez nadmiernej scentralizowanej władzy lub zarządzania. W dziedzinie informatyki SI jest zbiorem algorytmów i koncepcji, które modelują i formalizują takie zachowania inteligentnych grup. Na bardzo wysokim poziomie SI można postrzegać jako system, który koncentruje się na uzyskiwaniu przydatnych inteligentnych zachowań, które są wynikiem wspólnych wysiłków osób należących do grupy (zwanej także rojem). Te osoby nazywane są agentami. Każdy z tych środków ma jednorodny charakter. Działają asynchronicznie i równolegle bez scentralizowanej kontroli lub nadmiernego zarządzania. Ogólnie rzecz biorąc, ci agenci współpracują ze sobą świadomie lub nieświadomie, aby osiągnąć określony cel, który określa inteligentne zachowanie grupy. Z punktu widzenia sztucznej inteligencji lub informatyki możemy podać następujące informacje jak definicja inteligencji roju:

Inteligencja roju to zbiór inteligentnych systemów inspirowanych zbiorową inteligencją grupy. Tę kolektywną inteligencję osiąga się poprzez bezpośrednie lub pośrednie interakcje czynników o jednorodnej naturze, ale współpracujących ze sobą w ich lokalnym środowisku, bez świadomości globalnego kontekstu lub schematu.

Podczas budowania dowolnego systemu opartego na SI istnieją trzy podstawowe koncepcje lub właściwości, które proponowany system powinien spełniać co najmniej. Te trzy podstawowe właściwości to samoorganizacja (SO), stygmergia i podział pracy. Przyjrzyjmy się teraz kolejno tym właściwościom.

Samoorganizacja

Jest to jedna z najważniejszych cech układów SI. SO jest własnością systemów SI, która określa podstawową współpracę między agentami SI w celu osiągnięcia pożądanego zachowania zbiorowego. SO to jedno globalne zachowanie lub zjawisko, które osiąga się poprzez interakcje między jego agentami lub botami niższego poziomu. Te interakcje zależą od zestawu reguł, które są uwzględniane w oparciu o lokalny kontekst lub środowisko, w którym działają agenci. Agenci ci nie są świadomi żadnych globalnych wzorców ani zachowań. Jednak globalne zachowanie wynika z indywidualnego funkcjonowania agentów. Kluczem jest to, że nie ma zewnętrznego organu kontrolującego lokalne zachowanie agentów. W skrócie, zachowanie globalnej grupy w dowolnym systemie SI osiągane jest przez samoorganizujące się zdolności poszczególnych agentów, których zakres funkcjonalny jest ograniczony do środowiska lokalnego. Istnieją cztery podstawowe aspekty SO. Oni są:

*Pozytywne opinie

* Negatywne opinie

* Losowe fluktuacje behawioralne

* Wiele interakcji między agentami

Pozytywne opinie to pewne zasady, które pomagają w budowaniu najlepszego zachowania roju na świecie. Na przykład rekrutacja pszczół lub wzmocnienie nowych członków zespołu w celu zbierania żywności lepszej jakości z lepszego źródła żywności jest przykładem pozytywnej opinii. Jeśli kolonia pszczół będzie prezentowana z dwoma źródłami pożywienia o podobnym charakterze pod względem jakości pożywienia i znajdującymi się w tej samej odległości, pszczoły będą próbowały zbierać pożywienie z obu źródeł jednocześnie. Jeśli jednak jakość jednego źródła pożywienia jest gorsza, pszczoły najpierw wykorzystają lepsze źródło pożywienia w oparciu o pozytywne opinie otrzymane na temat tego źródła pożywienia. Innym zachowaniem, które może wyniknąć z pozytywnego sprzężenia zwrotnego, jest założenie, że pszczoły są w lepszym źródle pokarmu w trakcie zbierania pokarmu z innego źródła, a następnie kolonia pszczół może całkowicie lub częściowo porzucić to źródło pokarmu. Zrekrutują lub wzmocnią więcej pszczół, aby zbierać żywność z nowo zidentyfikowanych lub lepszych

źródeł żywności. Takie zachowanie, które zwiększa szanse przeżycia dla całej społeczności, jest wynikiem SO. Każdy indywidualny typ pszczół zna swoją rolę i obowiązki oraz wykonuje działania, które prowadzą do wykonania danego zestawu zadań. Podobne zachowanie SO obserwuje się także u mrówek. Kolonia mrówek jako całość zawsze stara się zbudować gniazdo, które będzie bezpieczne od trudnych warunków i organizować indywidualne działania mrówek, aby zlokalizować źródło pożywienia najbliższe spośród wszystkich dostępnych źródeł pożywienia. Mrówki stosują unikalny i inteligentny algorytm lokalizowania najbliższego i najobfitszego źródła pokarmu. Po ustanowieniu schronienia (kolonii) najważniejszym aspektem przetrwania kolonii jest znalezienie najbliższego i najbogatszego źródła pożywienia. Mrówki robotnicze (same i w sposób samoorganizujący się) zaczynają się przemieszczać w wielu partiach w niezależnych kierunkach. Podczas eksploracji różnych miejsc wydzielają substancję chemiczną zwaną feromonami. Podczas gdy wciąż badają źródło pożywienia, ilość feromonów jest stała i wskazuje, że poszukiwania wciąż trwają. Po znalezieniu źródła mrówka przemierza ścieżkę z powrotem do kolonii. Jednak tym razem wydziela różnorodną ilość feromonów. Im większa ilość feromonów, tym większe i obfite źródło pożywienia. Ten sygnał jest wystarczający, aby inne mrówki w kolonii natychmiast przeszły tą samą ścieżką (ponownie w sposób samoorganizujący się). Nie ma centralnego mechanizmu dowodzenia i kontroli, który śledzi wszystkie mrówki znajdujące się na określonej ścieżce. Jednak ogólne osiągnięcie celu (w tym przypadku znalezienie pożywienia) nie zależy od centralnego dowództwa, o ile mrówki samoorganizują się. Jeśli źródło pokarmu nagle zanika, mrówki mają plan awaryjny oparty na wtórnych źródłach pożywienia znalezionych przez inny zestaw mrówek i na podstawie poziomu feromonu na alternatywnej ścieżce. Jak widać SO dla spełnienia indywidualnej odpowiedzialności jest kluczem do przetrwania mrówek. Systemy AI czerpią wiele inspiracji z tych przykładów i powinny być budowane przy użyciu środków do samoorganizacji z konkretną odpowiedzialnością za pracę w kontekście środowiska aplikacji. Ważnym aspektem jest jednak to, że poszczególni agenci działają bez lidera lub scentralizowanej kontroli w oparciu o prostą regułę dla swoich działań w środowisku. Te proste zasady, działając w harmonii, powodują inteligentne zachowanie, które znacznie wykracza poza łączną sumę możliwości poszczególnych agentów.

Stygmergia

Reguły muszą reagować na zmiany stanu środowiska, a agent powinien mieć możliwość samodzielnego dostosowania się do zmian i dalszego wykonywania swojej funkcji. To zachowanie nazywa się stygmatyzmem. Bez tej właściwości agent nie może się samoorganizować i będzie wymagał scentralizowanego agenta kontrolującego. W przypadku stygmergii agent jest informowany o kontekście, w którym działa, nawet jeśli środowisko zmieni się w porównaniu z poprzednią interakcją agentów z nim. Weźmy na przykład mrówkę poruszającą się po ścieżce do źródła żywności, a na ścieżkę wylewa się trochę wody. Gdy mrówka napotka po drodze wodę, zaczyna szukać alternatywnej ścieżki opartej na sygnale feromonu. Może również przejść przez swoją drogę powrotną do kolonii, a następnie samodzielnie rozpocząć od nowa na innej ścieżce (bez centralnej kontroli). Jednocześnie mrówka zostawia ślady dla innych mrówek, aby wiedzieć, że na określonej drodze do źródła pożywienia pojawiają się problemy. Inne mrówki natychmiast dostosowują się do zmiany otoczenia w oparciu o doświadczenia poprzednich mrówek i modyfikują swoje trajektorie w oparciu o proste zasady. Mrówki wchodzą ze sobą w interakcje bez wyraźnej komunikacji, ale tylko z modyfikacjami stanu środowiska. W tym momencie mrówki stosują zasady uczenia się przez wzmocnienie, które zbadaliśmy w poprzednim rozdziale. W drodze do źródła jedzenia iz powrotem mrówka nieustannie dostosowuje się do środowiska w oparciu o nagrodę za każde indywidualne działanie i stan środowiska. Celem pojedynczego agenta (w tym przypadku mrówki) jest maksymalne zwiększenie nagrody (zlokalizowanie źródła pożywienia lub przyniesienie pożywienia z powrotem do kolonii) niezależnie.

Podział pracy

Jest to najbardziej fundamentalny aspekt SI. Indywidualny agent w obrębie roju ma bardzo ograniczone możliwości osiągnięcia celu dla całego roju. Naturalny system stosuje podział pracy z poszczególnymi agentami wykonującymi zestaw bardzo szczegółowych obowiązków, które przyczyniają się do ogólnego sukcesu roju. Na przykład wszystkie pszczoły w ulu nie robią tego samego. W ulu pszczół występuje wyraźny podział pracy w zależności od rodzaju pszczoły. Królowa pszczół jest odpowiedzialna za składanie jaj, samce drony są odpowiedzialne za rozmnażanie, a pszczoły robotnice budują ul i pracują, aby zdobyć żywność dla całej populacji. Opiekują się również pszczołami królowymi i dronami, karmiąc je. W systemach AI każdy indywidualny agent musi być zaprogramowany tak, aby miał własne reguły oparte na kontekście środowiskowym, aby wykonywać określony zestaw obowiązków. Dzięki podziałowi pracy systemy przetwarzania równoległego mogą efektywnie działać i rozdzielają obciążenia pracą, nie tracąc z oczu ogólnej nagrody i celu. Na tym tle z SI przyjrzyjmy się niektórym zaletom inteligencji zbiorowej w celu maksymalizacji korzyści.

Zalety kolektywnych inteligentnych systemów

Inteligentne systemy zbiorowe mają następujące zalety:

* Elastyczność: agenci mają swoje indywidualne zasady działania w kontekście środowiska. Agent reaguje na zmiany w środowisku, a następnie cała populacja wykazuje elastyczność w celu dostosowania się do zmian w środowisku.

* Wytrzymałość: Ponieważ agenci są indywidualnie bardzo małą jednostką w ramach całości, nawet jeśli jeden agent zawiedzie, społeczność nie ucierpi, a ogólny cel może zostać osiągnięty.

* Skalowalność: Ponieważ poszczególne agenty są małymi jednostkami niezależnej pracy, możliwe jest skalowanie od setek do tysięcy do milionów takich inteligentnych agentów w oparciu o przypadek użycia i osiągnięcie wykładniczo wyższych zwrotów i kumulatywnej inteligencji.

* Decentralizacja: Ponieważ w kolonii nie ma centralnej kontroli, agentów można rozmieścić na krawędzi obliczeń (realistyczny scenariusz w przypadku przypadków użycia Internetu Rzeczy). W przeciwieństwie do rozproszonego środowiska obliczeniowego, w którym centralny serwer węzłów musi być przyrostowo wydajny, w przypadku SI nie ma potrzeby scentralizowanego sterowania, ponieważ agenty działają w oparciu o reguły w środowisku.

* Samoorganizacja: możliwe rozwiązania, które wdrażają algorytmy oparte na SI, mogą ewoluować i dostosowywać się do zmian w środowisku oraz pojawiać się bez wcześniejszego zdefiniowania.

* Adaptacja: Agenci i system jako całość mogą dostosować się i dostosować do predefiniowanego środowiska wraz z nowymi zmianami w środowisku. Adaptacja jest również unikalną cechą pojedynczego agenta zamiast być centralnie kontrolowanym.

* Zwinność i szybkość: System inteligencji oparty na algorytmach roju wykazuje zwinność i lepszą prędkość przy każdej interakcji z otoczeniem.

Projektując systemy oparte na SI, istnieją pewne zasady przewodnie, których należy przestrzegać w celu opracowania samowystarczalnych systemów.

Zasady projektowania systemów SI

Zasady projektowania, które należy wziąć pod uwagę przy opracowywaniu układów SI, są następujące:

* Zasada bliskości: Poszczególni agenci w roju powinni móc komunikować się z centrum populacji w rozsądnym czasie, badając indywidualnie przestrzeń poszukiwań. Na przykład mrówka poszukująca pożywienia powinna móc zgłosić się do kolonii, gdy tylko zostanie znalezione źródło pożywienia. Zgłaszanie musi odbywać się w sposób uwzględniający czas, aby źródło żywności było istotne. Zasada bliskości określa domyślną granicę demograficzną dla członków.

* Zasada jakości: Podczas gdy niezależni agenci docierają do rozwiązania niezależnie w przestrzeni poszukiwań, rój powinien być w stanie określić jakość rozwiązania i ruszyć w tym kierunku. Po raz kolejny, jeśli wiele mrówek znajdzie źródło pożywienia, każde z nich wraca z różnymi poziomami feromonów po drodze proporcjonalnie do jakości i ilości źródła pożywienia. Pomaga to grupie jako całości zdecydować, do którego źródła żywności się udać. Jednak nie ma centralnego polecenia określającego standard jakości i określające ścieżkę. Z drugiej strony agenci komunikują się i współpracują, aby dotrzeć do właściwego źródła żywności.

* Zasada różnorodnej reakcji: Podczas gdy agenci rozwiązują wspólny problem, nie powinni koncentrować się na małym regionie w ogólnej przestrzeni wyszukiwania. Muszą być włączone do eksploracji przy wykorzystaniu wcześniej zrozumiałych wzorców. Rój powinien dążyć do dywersyfikacji z pewnym progiem, który określa granicę przeżycia poszczególnych agentów.

* Zasada zdolności adaptacyjnej: rój jako całość powinien być w stanie dostosować się do zmian w otoczeniu. Agenci powinni organizować się w zgodzie ze zmieniającym się środowiskiem.

Dzięki podstawowemu zrozumieniu podstaw SI, zrozumiemy dwa algorytmy, których można użyć do budowy sztucznych czynników, które pracują w grupie o dużych rozmiarach, aby wykonywać wspólnie duże zadania.

Model optymalizacji roju cząstek

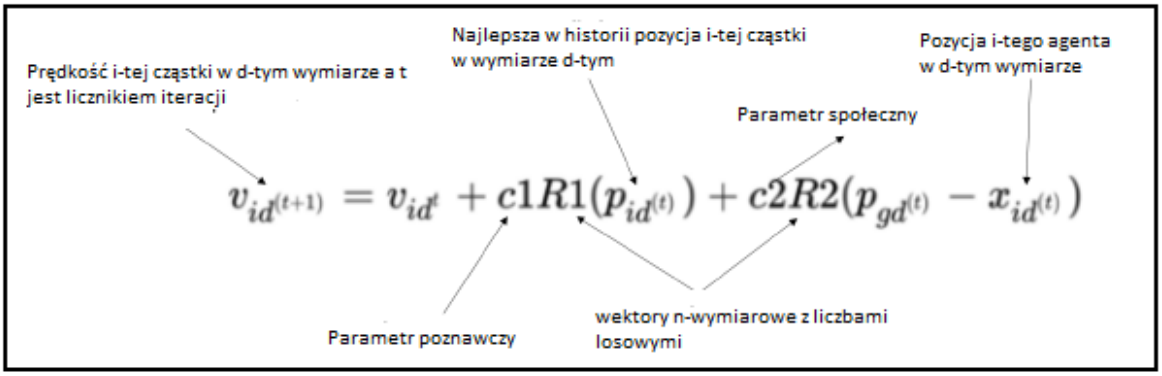
Model optymalizacji roju cząstek (PSO) jest inspirowany stadem ptaków i szkolnym ruchem ryb. Celem modelu PSO jest znalezienie optymalnego rozwiązania (źródła żywności lub miejsca zamieszkania) w dynamicznej przestrzeni. Rój zaczyna się w przypadkowej lokalizacji i losowej prędkości i opiera się na zbiorowym zachowaniu poprzez eksplorację i wykorzystanie przestrzeni poszukiwań. Unikalną cechą PSO jest to, że agenci działają w formacji, która optymalizuje wyszukiwanie, a także minimalizuje zbiorowy wysiłek zmierzający do uzyskania optymalnego rozwiązania. Agenci w roju podążającym za modelem PSO podążają za nimi wedle zasad przewodnich:

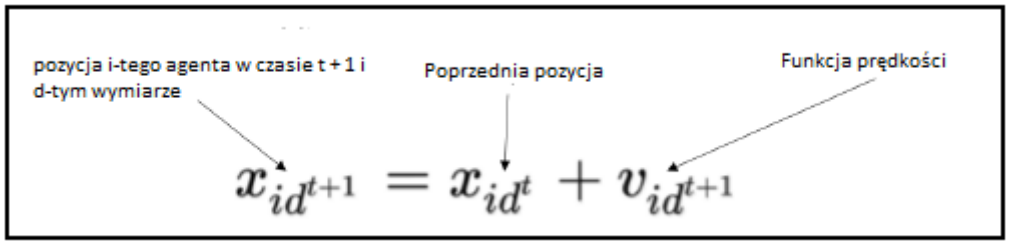
* Separacja: Każdy indywidualny agent jest zaprogramowany w taki sposób, aby był w stanie zachować wystarczającą odległość z towarzyszami stad, aby nie wpadali na siebie i jednocześnie utrzymywali osobną przestrzeń egzystencji, aby być częścią formacji w poszukiwaniu optymalnego rozwiązania. Agent podąża za najbliższym sąsiadem w celu dostosowania swojej pozycji i prędkości w celu zapewnienia odpowiedniego poziomu separacji.

* Wyrównanie: Każdy pojedynczy agent dopasowuje się do ogólnego wzoru roju i średniej prędkości grupy w przestrzeni poszukiwań.

Zgodnie z ogólną zasadą każdy członek roju, który podąża za modelem PCO, nieustannie przekazuje swoje doświadczenia grupie jako całości, aw szczególności najbliższym sąsiadom. Agent ma pogląd na najbliższych członków oraz ich zachowanie i wzorce uczenia się. Środek albo wpływa na ruch (pozycję i prędkość) sąsiedniego agenta na podstawie obserwacji i przydatności swoich doświadczeń w przestrzeni poszukiwań dla lokalnego optymalnego rozwiązania lub dostosowują swój ruch w oparciu o lepsze doświadczenia dla najbliższych członków. Podstawową zasadą jest wyrównanie z najbliższymi sąsiadami, a zatem cały rój jako całość w interesie większego celu. Pierwotnie PSO było proponowane

jako algorytm optymalizacyjny w przestrzeni ciągłego wyszukiwania wartości rzeczywistej, a teraz zostało rozszerzone, aby obsługiwać przypadki użycia wyszukiwania binarnego lub dyskretnego. Algorytm rdzenia jest definiowany przez równania prędkości i pozycji w następujący sposób:





W celu zdefiniowania prędkości (szybkości zmiany lokalizacji dla pojedynczego agenta) ważną rolę odgrywają dwa parametry. Najlepsza pozycja na osi czasu dla pojedynczego agenta jest reprezentowana przez $P_{id}^{(t)}$i $P_{gd}^{(t)}$, które reprezentują pozycję najlepszego agenta w globalnej pozycji roju w kontekście środowiskowym. Gdy te dwa parametry przyczyniają się do ogólnej prędkości roju, jest to optymalna prędkość do poszukiwania rozwiązania w przestrzeni, pod warunkiem, że środowisko jest deterministyczne. Jednak w przypadku środowiska stochastycznego czynniki R1 i R2 odgrywają rolę w dostosowywaniu się do zmian stanu środowiska. Parametry te wprowadzają wymaganą losowość w roju, aby skutecznie zbadać przestrzeń poszukiwań. c1 i c2 są parametrami poznawczymi i społecznymi, które reprezentują względne znaczenie konkretnego czynnika w odniesieniu do najlepszej pozycji roju. Względne wartości tych parametrów stale przesuwają pojedynczego agenta do najlepszej pozycji w roju, nawet jeśli środowisko ulega zmianie. Wpływ względnej różnicy między c1 i c2 można przedstawić w następujący sposób:

c1: c2: Poziom eksploracji

Wysoka: Wysoka: Wysoka eksploracja w odległych regionach

Mały: Mały: Ulepszone wyszukiwanie i niższy poziom eksploracji

Wysoka: Mała: stronniczość w kierunku najlepszej pozycji konkretnego agenta na świecie

Mały: wysoki: nastawienie na najlepszą pozycję globalną roju

Funkcja prędkości ma trzy różne składniki:

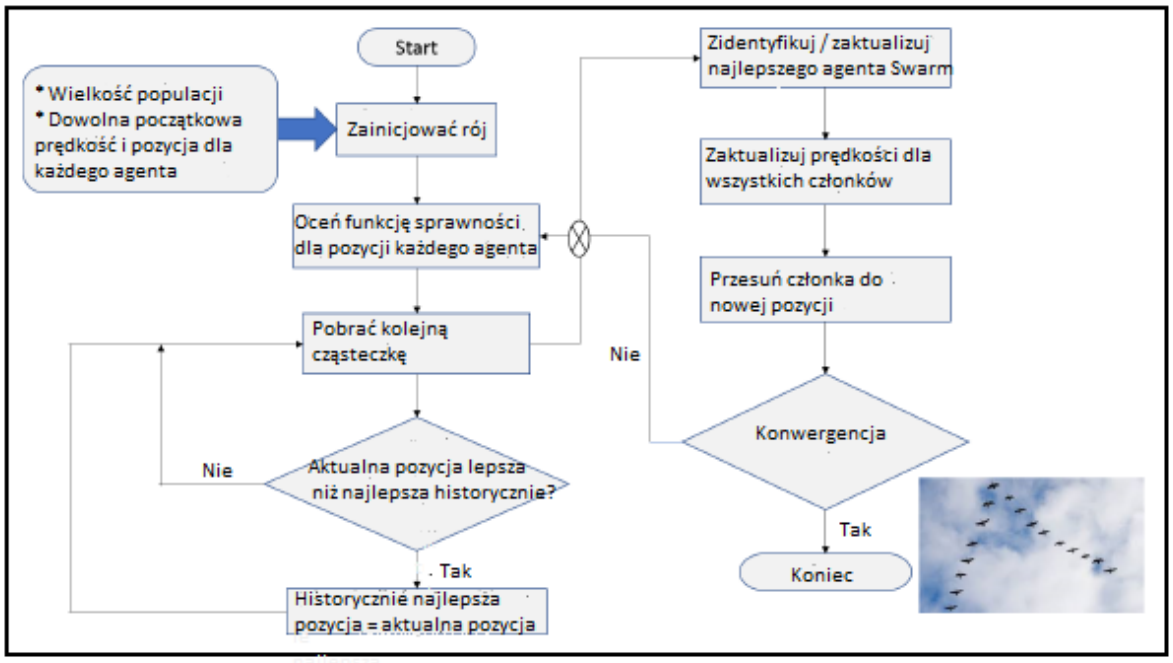
* Bezwładność (v_{id}^t): Bezwładność to opór dowolnego obiektu fizycznego na każdą zmianę jego stanu ruchu. Obejmuje to zmiany prędkości, kierunku lub stanu spoczynku obiektu. Prędkość w czasie t + 1 jest funkcją czasu t. Co oznacza, że rój nie może gwałtownie zmienić prędkości. Zamiast tego następuje stopniowa zmiana prędkości w zależności od zmiany środowiska lub jeśli rój tego potrzebuje zmień

prędkość, aby skutecznie nawigować w przestrzeni wyszukiwania. Z powodu tej bezwładności obserwujemy, że rój ptaków przez większość czasu kontynuuje ten sam kierunek i porusza się w formacji, ponieważ prędkość czynnika w czasie t + 1 zależy od prędkości czynnika w czasie t. Termin ten jest również bardzo ważny dla zmiany najlepszego globalnego agenta w roju. Gdy funkcja sprawności agentów jest bardziej optymalna w porównaniu do globalnej sprawności roju, agent zajmuje pozycję najlepszego agenta na świecie w roju. Podczas przejścia ($p_i = x_i = p_g$) wyrażenie społeczne w równaniu staje się zero ($p_{id}^{(t)} – x_{id}^{(t)} = p_{ig}^{(t)} - x_{id}^{(t)} = 0$). W tym momencie nowy czynnik staje się najlepszą na świecie cząsteczką, poruszając się z nową prędkością, a tym samym zmieniając pozycję w roju.

* Samowiedza ($p_{id}^{(t)} – x_{id}^{(t)}$): ten składnik funkcji prędkości określa indywidualność agentów w roju. Przekłada się to na poziom przyciągania cząstki do jej najlepszej globalnej wartości, która optymalizuje wyszukiwanie w przestrzeni rozwiązania.

* Wiedza społeczna ($p_{ig}^{(t)} - x_{id}^{(t)}$): ten składnik funkcji prędkości definiuje adaptacjędo zachowań społecznych wszystkich agentów. Dzięki temu wyrażeniu określa się stopień uczenia się w grupie i dzielenia się doświadczeniami między poszczególnymi członkami.

Model PSO można przedstawić w następujący sposób:



Uwagi dotyczące wdrożenia PSO

Musimy wziąć pod uwagę następujące kwestie związane z implementacją PSO:

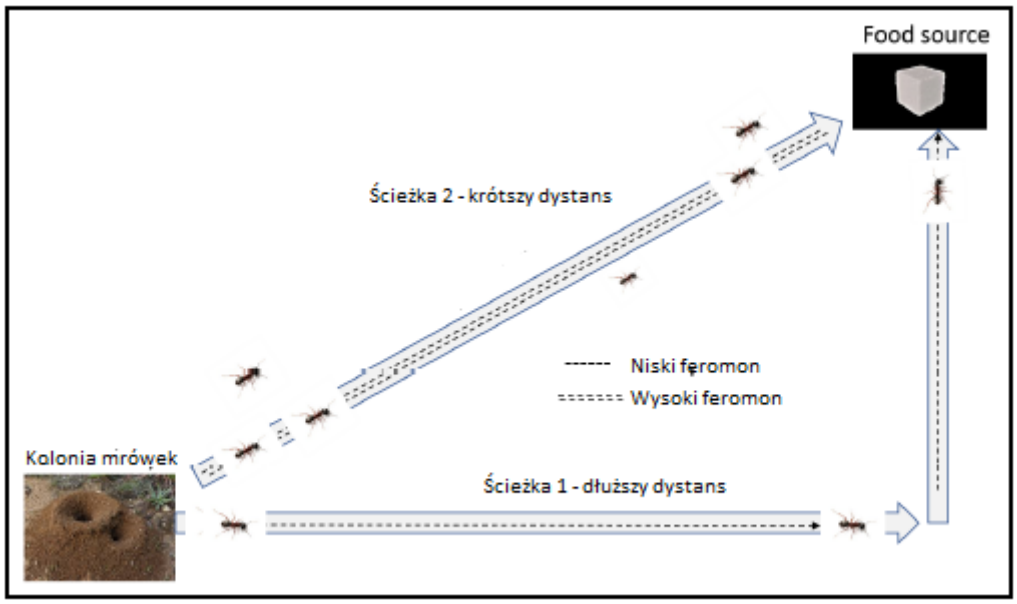
* PSO przechowuje najlepszą pozycję agenta w znacznym i odpowiednim czasie wraz z najlepszą na świecie pozycją dla roju. Dzięki temu agent z maksymalnym wynikiem sprawności ma wpływ na ogólne zachowanie roju, a konwergencja jest szybka.

* PSO jest prostym algorytmem do wdrożenia, ponieważ matematyczne równania prędkości i położenia są łatwe do wdrożenia ze względu na naturalną prostotę.

* PSO może bardzo efektywnie dostosowywać się do zmian w otoczeniu, szybko dostosowując prędkość i pozycje członków w każdej iteracji.

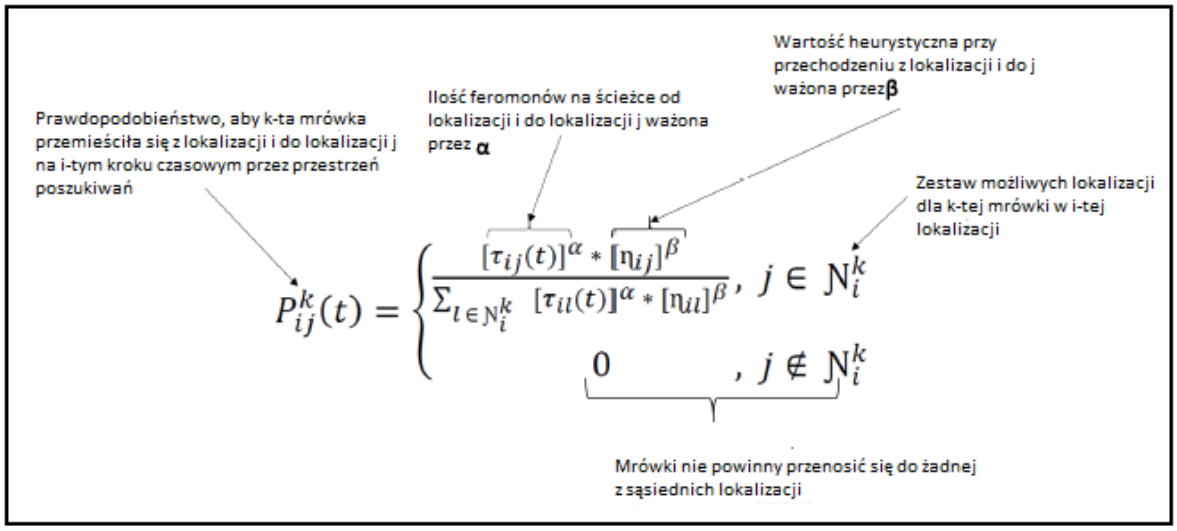
Model optymalizacji kolonii mrówek

Optymalizacja kolonii mrówek (ACO) to kolejna szeroko stosowana i dostosowana odmiana algorytmów SI. Jako minimum, celem kolonii mrówek lub roju sztucznych środków jest optymalne odnalezienie zasobu (pożywienia w przypadku mrówek i paczki w przypadku kolonii robotów w magazynie detalisty), aby: przemierzać minimalną odległość do i od zasobu i lokalizacji bazowej. Ten model jest przydatny w przypadku dronów monitorujących, autonomicznego planowania tras samochodowych i tak dalej. Pozwól nam zrozumieć niektóre zasady działania kolonii mrówek i zapoznaj się z terminologiami, aby można je było zastosować przy projektowaniu sztucznych rojów w oparciu o model ACO. Oto postać kolonii mrówek i źródło żywności:



W tym przykładzie w pobliżu kolonii mrówek znajduje się źródło pożywienia. Istnieją dwie ścieżki do jedzenia. Ścieżka-1 to dłuższa odległość do źródła Żywności, a Ścieżka-2 to najkrótsza odległość. Mrówki rozpoczynają eksplorację przestrzeni poszukiwań niezależnie. Każda mrówka ma zadanie znaleźć źródło pożywienia. Kilka mrówek wybiera długą drogę i znajduje źródło pożywienia, a po drodze wydziela szlak feromonów. Gdy mrówki wracają do kolonii, inne mrówki otrzymują sygnał o znalezieniu źródła pożywienia i zaczynają przemierzać ścieżkę. Tymczasem mrówki krótszą ścieżką wracają szybciej niż na dłuższej ścieżce, a więcej mrówek zaczyna przemierzać krótką ścieżkę, ponieważ efektywny czas jest krótszy. Z biegiem czasu mrówki na krótkiej odległości gromadzą po drodze więcej feromonu, co sygnalizuje kolonii, że źródło pożywienia jest bardziej optymalne pod względem odległości, a także jakości. Z czasem feromon na długodystansowej ścieżce wyparowuje, a ścieżka przestaje istnieć. W końcu mrówki przestają przemierzać długą drogę do źródła pożywienia, a zbiorowe zachowanie kolonii jest w pełni zoptymalizowane. Kiedy naśladujemy koncepcje naturalnych mrówek i ich techniki optymalizacji w projektowaniu sztucznych środków, koncepcje można ulepszyć i dalej zoptymalizować w zależności od środowiska. Na przykład naturalne mrówki nie mają żadnej pamięci. Działają w ramach zestawu zasad określających ich ruch i ogólne zachowanie (wydzielanie feromonów). Sztuczne mrówki (agenci) mogą mieć ograniczoną pamięć, która przechowuje nagrody na podstawie przeszłych działań, a zatem inteligentne zachowanie można poprawić. Naturalne mrówki podlegają ekologicznym modyfikacjom i ograniczeniom. Na przykład woda spada w drodze do źródła żywności. Sztuczne czynniki nie podlegają modyfikacjom ekologicznym i zwykle działają w

kontrolowanym i przewidywalnym środowisku. Sztuczne środki symulują wzorce naturalnych mrówek, osadzając feromon po drodze, aby wzmocnić zachowanie innych środków. Sztuczne środki również przemierzają ścieżkę z większym stężeniem feromonów i uzupełniają ją składnikiem pamięci dla optymalnego zachowania. W przypadku sztucznych mrówek feromon szybko odparowuje, aby kolonia mogła zbadać dalsze optymalizacje. W przeciwieństwie do prawdziwych mrówek, które działają w oparciu o przetrwanie jako podstawowy instynkt. Podstawowa intuicja podczas opracowywania agentów opartych na ACO opiera się na funkcji fitness, która określa działania agentów zwracających po drodze maksymalne poziomy feromonów. Jest to również postrzegane jako problem optymalizacji kosztów, który zmniejsza skumulowane koszty osiągnięcia przez kolonię celu w przestrzeni wyszukiwania Na poziomie algorytmu pojedynczy agent działa w oparciu o regułę prawdopodobieństwa, która pomaga wybrać komponenty (etapy sekwencyjne), które wykorzystują poziomy feromonów na drodze i zmiany środowiskowe. Podczas gdy sztuczne mrówki poruszają się w przestrzeni roztworu w oparciu o funkcję prawdopodobieństwa, musi również określić ilość feromonu, aby mógł się on osadzić. Reguła prawdopodobieństwa nazywa się regułą przejścia stanu:



W poniższym równaniu α i β są parametrami, które kontrolują ogólny wpływ feromonu i podejście heurystyczne w uzyskiwaniu funkcji prawdopodobieństwa. Jest to podobne do wartości c1 i c2 w PSO. Wpływ względnej różnicy między α i β można przedstawić następująco:

α: β: Wpływ na zbieżność

High: Low: Ważne informacje dotyczące feromonów. W przypadku agenta istnieje większa możliwość wyboru działań i pozycji poprzednio zajmowanych przez innych agentów. Może to prowadzić do nasycenia wielu agentów w tym samym regionie, a tym samym do zmniejszenia potencjału roju do zbadania przestrzeni poszukiwań, a tym samym do uzyskania nieoptymalnych wyników.

Niski: wysoki: Algorytm zachowuje się jak stochastyczny algorytm wielozadaniowy, przy czym poszczególni członkowie kolonii wydają się być odpowiedzialni za samodzielne znalezienie optymów przy niskim poziomie współpracy i ograniczonym uczeniu się na drodze wzajemnej.

Zero: High: W tym przypadku algorytm działa jako stochastyczny, chciwy algorytm z poszczególnymi członkami odpowiedzialnymi i przy zerowym uczeniu się na ścieżce innych. Węzeł o minimalnym koszcie otrzyma preferencję, a poziom feromonu na ścieżce nie zostanie przypisany do wieku.

Wysoka: zero: w tym przypadku algorytm działa tak samo jak naturalne mrówki, w których zasadą przewodnią jest wyłącznie feromon i nie ma informacji heurystycznych wykorzystywanych do przeszukiwania przestrzeni problemowej.

Biblioteka MASON

Multi-Agent Simulation Of Neighborhoods or Networks (MASON) to oparta na Javie wieloagentowa biblioteka symulacyjna, która ma ogólną bibliotekę API w celu łatwej symulacji w szczególności algorytmów SI i dowolnego ogólnego algorytmu, który bada przestrzeń wyszukiwania przy użyciu niezależnych agenci w ogólności. Ta biblioteka została stworzona przez George'a Masona z uniwersyteckiego Centrum Złożoności Społecznej i Wydziału Informatyki. Zapewnia szybki i przenośny rdzeń napisany w języku programowania Java i jest obsługiwany przez platformę wizualizacji do testowania i wizualizacji hipotez. Jest to przydatna platforma do modelowania nowych architektur i algorytmów. Celami projektowymi biblioteki MASON są:

* Zapewnienie dużej liczby symulacji i konfigurowalnych eksperymentów. Biblioteka jest bardzo łatwa do rozszerzenia o dodatkowe symulacje i przypadki użycia.

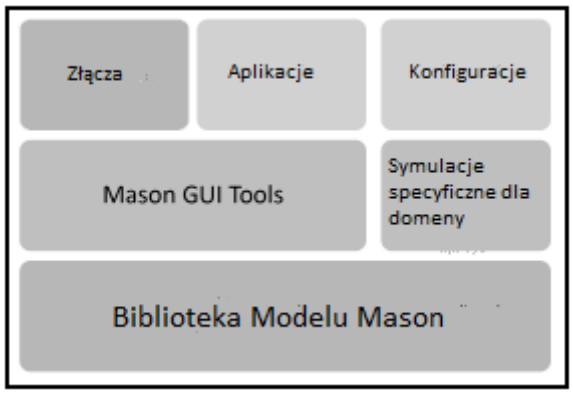
* Wysoki stopień modułowości i elastyczności - platforma jest zbudowana jako architektura warstwowa i wykorzystuje podstawy zorientowane obiektowo, aby utrzymać luźne powiązanie obowiązków poszczególnych elementów składowych.

* Oddzielne narzędzia do wizualizacji - platforma ma warstwę wizualizacji, która jest niezależna od silnika kodu i może być również rozszerzona w zależności od przypadku użycia i kontekstu testowanej hipotezy aplikacji.

MASON to wielofunkcyjny symulator zdarzeń, który działa jako pojedynczy proces, który skutecznie obsługuje dużą liczbę agentów. Aplikacje MASON są tak różnorodne, jak modelowanie złożoności społecznej, modelowanie fizyczne przestrzeni wyszukiwania i interakcje agentów ze środowiskiem, niezależni i abstrakcyjni agenci, których można zaprogramować tak, aby przestrzegali podstawowych zasad i działali jako członkowie roju. Ramy są przydatne do badań i symulacji AI i ML.

Architektura warstwowa MASON

MASON wdrożył architekturę warstwową z odrębnymi komponentami, które są luźno połączone i zintegrowane z ogólnym interfejsem. Poniższy rysunek pokazuje różne elementy architektury warstwowej:

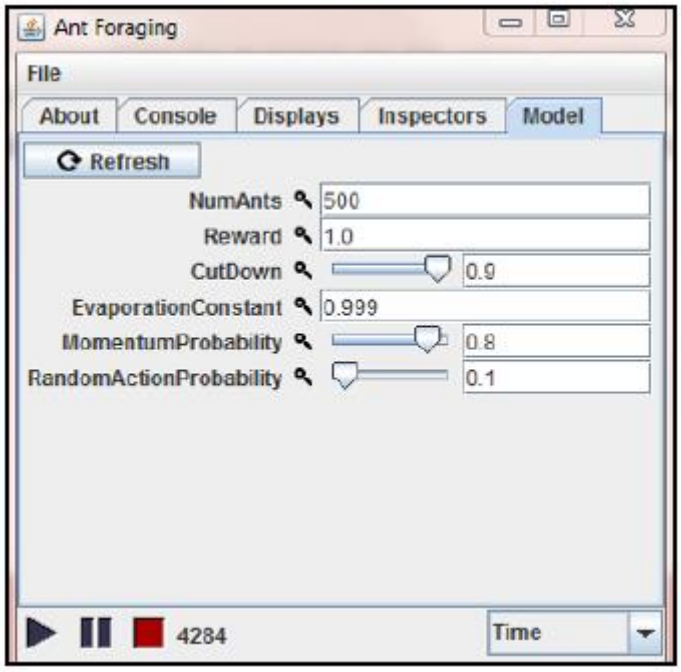


Przede wszystkim biblioteka MASON zawiera dwa główne komponenty: bibliotekę modeli i platformę wizualizacji. Wizualizacja obsługuje renderowanie 2D i 2D. Model i wizualizacje są całkowicie

oddzielone, a modele mogą być niezależnie wykonywane, a wyniki zwracane do konsoli lub plików wyjściowych. Interfejs użytkownika jest luźno sprzężony i działa w oparciu o bieżące stany obiektów w obiektach modelu. Zamiast podejścia odgórnego, które rozpoczyna się od interfejsu użytkownika i inicjuje model, platforma MASON utrzymuje model i komponenty wizualizacji w pełni niezależne. Takie podejście zapewnia elastyczność tworzenia różnych wizualizacji (opartych na Javie lub, jeśli jest to wymagane, opartych na sieci), zgodnie z wymaganiami. Jedną z podstawowych funkcji zaimplementowanych przez Masona jest sprawdzanie. Model może być serializowany na dysk i może być wywoływany na zupełnie innej platformie w innym czasie i jest inicjowany do tego samego stanu. Ułatwia to dużą interoperacyjność i współpracę między zespołami badawczymi. Biblioteka MASON zapewnia prosty interfejs API do tworzenia nowych symulacji. Aby utworzyć nowy obiekt agenta, musi on rozszerzyć klasę sim.engine.SimState. Najprostsza implementacja szkieletu jest następująca:

```
import sim.engine.*;

public class SWARMAgent extends SimState{

public SWARMAgent(long seed){

super(seed);

}

// method used for initialization of the model including the

configurations and the UI

public void start(){

super.start();

}

public static void main(String[] args){

doLoop(SWARMAgent.class, args);

System.exit(0);

}

}
```

MASON tworzy globalny stan instancji symulacji jako podklasę SimState. SimState hermetyzuje harmonogram zdarzeń (sim.engine.Schedule). Agenty są planowane za pomocą instancji klasy sim.engine.Schedule, która ma być zwiększana. Harmonogram jest reprezentacją czasu dla symulatora. Biblioteka Masona zawiera zestaw gotowych symulacji. Przyjrzyjmy się symulacji optymalizacji kolonii mrówek w bibliotece MASON. Jest to implementacja prostego scenariusza, który widzieliśmy na rycinie 9.4, reprezentacja kolonii mrówek i warunki. Pole poszukiwań zawiera dwie przeszkody, źródło pożywienia i lokalizację kolonii. Różne parametry, takie jak liczba mrówek i inne, można konfigurować w następujący sposób:



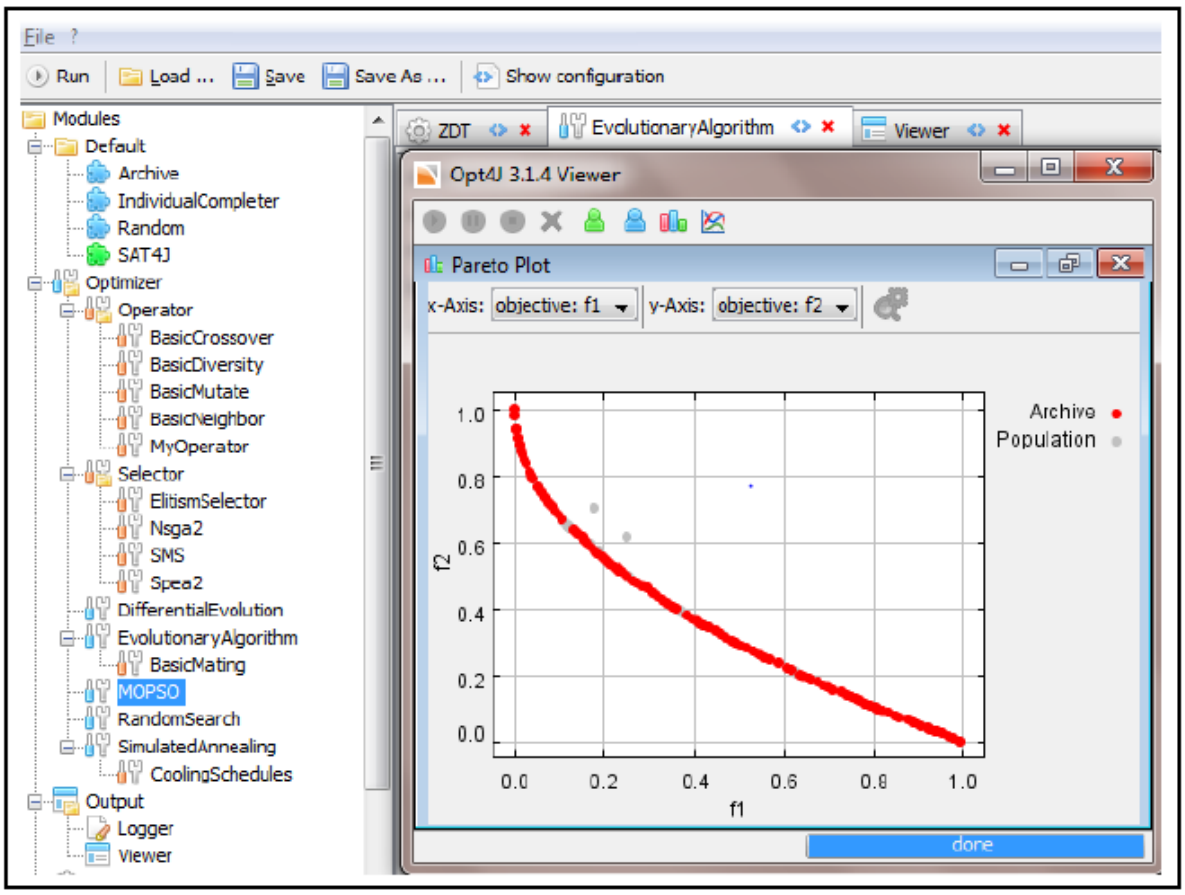
Po rozpoczęciu symulacji mrówki wchodzą indywidualnie na losową ścieżkę i po drodze deponują ślad feromonów. Wysoka wartość stałej parowania zapewnia mrówkom eksplorację przestrzeni poszukiwań zamiast grawitacji na już zbadane ścieżki. Gdy tylko pierwsza mrówka znajdzie źródło pożywienia, zaczyna przemierzać tam i z powrotem między źródłem pożywienia a lokalizacją bazy, pozostawiając ślad feromonu w obie strony. Mrówki są zaprogramowane tak, aby podążały śladem feromonów, a ostatecznie model zbiega się i mrówki wchodzą na zoptymalizowaną najkrótszą ścieżkę.

Biblioteka MASON ma wiele innych symulacji wstępnie wbudowanych w pakiet i możemy je eksplorować i eksperymentować z różnymi opcjami. Interfejs API można wykorzystać do rozszerzenia zakresu aplikacji przy minimalnym kodzie i wykorzystania możliwości frameworka oraz warstwy wizualizacji. W następnej sekcji pokrótce omówimy inny framework, Opt4J, który jest zbudowany przede wszystkim do obliczeń ewolucyjnych i może być również używany do eksperymentowania z algorytmami SI.

Biblioteka Opt4J

Opt4J to modułowa struktura optymalizacji metaheurystycznej, którą można zastosować do szeregu algorytmów ewolucyjnych. W kontekście tego rozdziału przyglądamy się implementacji algorytmów SI, takich jak ACO i PSO, za pomocą biblioteki. Biblioteki zajmujące się problemami optymalizacji mają trzy podstawowe elementy na poziomie abstrakcyjnym. Twórca, dekoder i ewaluator. Twórca udostępnia losowe genotypy (szczegółowe informacje na temat genotypu i fenotypów znajdują się w rozdziale 8, Programowanie genetyczne) z przestrzeni wyszukiwania. Reprezentują agentów w przypadku algorytmów SI. Agenty są tworzone przez obiekt twórcy. Biblioteka Opt4J udostępnia klasę org.opt4j.optimizers.mopso.Particle, która działa jako twórca. Agenci w roju to instancje tej klasy faktycznie utworzone przez klasę fabryczną 'org.opt4j.optimizers.mopso.ParticleFactory. Dekoder przekształca genotyp w fenotyp. Dekoder przekształca cechy abstrakcyjne w obiekty materialne i łączy z nimi wzorce zachowań. Na podstawie fenotypu ewaluator określa jakość bieżącego agenta w przypadku algorytmu PSO, funkcja ewaluatora zwraca prędkość i pozycję agenta oraz określa, czy jest to najlepsza pozycja i prędkość w roju. Po zdefiniowaniu podstawowych komponentów środowisko może obsłużyć problem optymalizacji.

Opt4J zapewnia prosty i intuicyjny interfejs użytkownika do ładowania modeli, a także wizualizacji w ograniczony sposób:



Jesteśmy na skraju rewolucji danych, a ilość danych z heterogenicznych źródeł rośnie z dnia na dzień. Chociaż ramy przetwarzania równoległego wraz z przetwarzaniem w chmurze stają się coraz lepsze w przetwarzaniu większej ilości danych, technika brutalnej siły nie będzie w stanie poradzić sobie ze wzrostem ilości danych. Będziemy musieli zastosować inteligentne techniki inspirowane naturą, takie jak programowanie genetyczne, uczenie się ze wzmocnieniem i SI, aby radzić sobie z dużymi zbiorami danych. W następnej sekcji przyjrzymy się niektórym możliwym przypadkom użycia w przypadku dużych zbiorów danych i bazowych zasobów obliczeniowych. Aplikacje w analityce dużych zbiorów danych Z każdą minutą gromadzimy coraz więcej danych na całym świecie i mamy teraz moc obliczeniową do przechowywania i przetwarzania zasobów danych. Pozwól nam krótko zrozumieć podstawową architekturę systemów dużych zbiorów danych. W obecnej formie struktura przetwarzania dużych zbiorów danych jest ogromną kolekcją węzłów obliczeniowych, które są rozproszone na całym świecie. Istnieją dwa podstawowe rozróżnienia we wdrożeniach. Systemy mogą być wdrażane lokalnie dla przedsiębiorstw, a zmiana paradygmatu w kierunku przetwarzania w chmurze polega na zwirtualizowaniu infrastruktury obliczeniowej i rozproszeniu geograficznym w różnych regionach. Niezależne jednostki obliczeniowe są nazywane węzłami. Węzły są połączone i kontrolowane przez scentralizowaną jednostkę obliczeniową, która śledzi wszystkie węzły i różne operacje na tych węzłach. Istnieje podobieństwo między naturalnymi rojami i węzłami dużych zbiorów danych, że węzły są niezależnymi jednostkami pracy. Jednak podobieństwo się kończy tam. Węzłami we wdrożeniu dużych zbiorów danych zarządza jeden lub więcej węzłów głównych, a węzły robocze pracują w synchronizacji na podstawie instrukcji z węzła głównego. Jak widzieliśmy w tym rozdziale, naturalne roje (kolonie mrówek, flota ptaków itd.) Nie mają centralnego dowództwa, a poszczególni

członkowie (agenci) działają w sposób autonomiczny w oparciu o reguły, a agenci są potrafi się dostosować w oparciu o stochastyczny charakter środowiska, w którym działają. Pojęcia SI mogą być stosowane w zabezpieczaniu infrastruktury dużych zbiorów danych, a także zapewnianiu pełnej równowagi węzłów. W głównym nurcie zadanie obliczeniowe jest najpierw przesyłane do węzła głównego, a to z kolei dzieli je na wiele części, które mają zostać wykonane przez dane lub oblicz węzły. W tym momencie zadania są wykonywane niezależnie przez węzły podrzędne. Na podstawie zasobów dostępnych w węzłach podrzędnych zadania kończą się w różnych momentach i wymagają różnych stopni obliczeń i przechowywania. W końcu może się zdarzyć, że obciążenie obliczeniowe rdzenia nie będzie równomiernie rozłożone na węzły. W oparciu o niektóre koncepcje poznane w tym rozdziale, oto ogólny algorytm (ACO) oparty na SI, który można wdrożyć w rozproszonym środowisku komputerowym. Za pomocą tego algorytmu ogólny proces to:

* Rozmnażanie: Jest to proces, w którym generowane są nowe sztuczne mrówki. Kontroler okresowo sprawdza platformę i generuje mrówki na podstawie obciążenia węzłów klastra. Jeśli węzły są przeciążone lub niedociążone, generowane są nowe mrówki do przenoszenia wiadomości.

* Eksploracja: w tym procesie agenci są niezależnie odpowiedzialni za znalezienie przeciążonych węzłów. Mogą śledzić sieć i sprawdzać parametry operacyjne, a po drodze pozostawić próbę symulowanego feromonu (licznik przyrostowy) innym członkom roju, aby otrzymać powiadomienie o przeciążonych lub niedociążonych węzłach.

Mrówki w tym roju poruszają się do przodu i do tyłu (tak samo jak naturalne mrówki, które przemieszczają się z kolonii do źródła pożywienia w obu kierunkach). Dla uproszczenia algorytmu równoważenia obciążenia dwa różne typy mrówek poruszają się w każdym kierunku niezależnie od zadania do wykonania. Ruchoma mrówka jest odpowiedzialna za znalezienie węzła, który jest przeciążony lub niedociążony. Agent ten zaczyna się od tej samej pozycji (węzła), w której się urodził, i zaczyna eksplorować przestrzeń. Agent, który porusza się do tyłu, niesie sygnał (kwantyfikowalny feromon) i tworzy ślad po drodze, który informuje, że dany węzeł jest przeciążony lub niedociążony. Dla uproszczenia w naszym modelu agent, który porusza się do tyłu, jest generowany tylko w przypadku napotkania węzła docelowego (który wymaga równoważenia obciążenia). Agent ruchu do przodu jest generowany w procesie węzła docelowego, gdy próg aktywności węzła zostanie osiągnięty (zarówno wysoki, jak i niski). Na tym tle i podstawowej wiedzy na temat tego podejścia przepływ równoważenia obciążenia dla rozproszonego środowiska obliczeniowego można podzielić na następujące etapy:

* Agenci obliczają i określają ilościowo obciążenie (poniżej i powyżej) w węźle, do którego jest on aktualnie podłączony.

* Zacznij od losowego nowego węzła, aby obliczyć jego przydatność do równoważenia obciążenia.

* Mrówka wsteczna jest generowana po znalezieniu kandydującego węzła. Ten agent aktualizuje informacje o feromonach, aby pozostawić ślad węzłów docelowych.

* Oblicz zbiorowe zapotrzebowanie na równoważenie obciążenia na podstawie kandydujących węzłów znalezionych przez agentów.

* Zrównoważyć obciążenie klastra.

Możliwość zastosowania SI jest jeszcze większa w przypadku Internetu Rzeczy, w którym obliczenia zbliżają się do krawędzi, a czujniki, które zbierają dane, mogą być traktowane jako członkowie roju i wykonują niezależne operacje dla ogólnej korzyści systemu, działając w oparciu o zasady rozmyte zamiast funkcji zakodowanych na stałe. Urządzenia brzegowe można zaprogramować z funkcjami

eksploracji i wykorzystania w ich środowisku pracy, aby wspólnie osiągnąć niektóre z predefiniowanych celów. Do tej pory obserwowaliśmy optymalizację przetwarzania dużych zbiorów danych w odniesieniu do wolumenu i przetwarzania rozproszonego. Istnieją jednak dwa ważniejsze aspekty dużych zbiorów danych, którymi są różnorodność i szybkość danych. Różnorodność i szybkość danych wymaga rozwiązania wielowymiarowego problemu dużych zbiorów danych. W następnej sekcji krótko omówimy obsługę danych dynamicznych i optymalizację wielu celów, gdy istnieje więcej niż jeden cel (jak w przypadku rzeczywistych scenariuszy) dla systemu przetwarzania danych.

Obsługa danych dynamicznych

Wraz ze wzrostem źródeł danych istnieje potrzeba znalezienia sensu z nich i wykorzystania go do lepszego podejmowania decyzji i uzyskiwania autonomicznych działań. Jednak wraz ze wzrostem liczby wymiarów i zmiennych wejściowych przeszukiwanie przestrzeni rozwiązań staje się intensywne obliczeniowo, a zastosowanie jedynie brutalnej siły i przetwarzania rozproszonego nie jest wystarczające. Możemy wykorzystać algorytmy SI w celu oznaczenia ważnych wymiarów wyższymi wagami wpływającymi na ogólny wynik. W tym konkretnym scenariuszu prędkość generowania danych zwiększa poziom złożoności ze względu na zmienność odbieranych danych. Niektóre z wyzwań, które należy rozwiązać przy projektowaniu roju sztucznych czynników, są związane z dynamiczną przestrzenią docelową, stan środowiska zmienia się bardzo szybko (nawet po przeprowadzeniu optymalizacji i określeniu poziomu feromonu przez inteligentnego agenta) . Gdy rój znajdzie optymalne wartości globalne, rzeczywista wartość może się zmieniać dynamicznie. Wymaga to wbudowania innego zestawu reguł w agentach, które dotyczą dynamiki środowiska. W tym momencie algorytm musi ewoluować, aby wziąć pod uwagę zwiększony koszt optymalizacji w przestrzeni dynamicznego wyszukiwania i pogodzić się z jakością uzyskanego rozwiązania. Sztuczni agenci wymagają poziomu rozmycia wbudowanego w funkcję celu, aby skutecznie radzić sobie z danymi dynamicznymi.

Optymalizacja wielu celów

Do tej pory w tym rozdziale wzięliśmy przykłady problemu z jednym celem (znalezienie źródła pożywienia dla kolonii mrówek). Jednak w rzeczywistych sytuacjach często jest więcej niż jeden cel, który musi zostać osiągnięty zarówno przez poszczególnych agentów, jak i rój. Na przykład w przypadku pszczół miodnych muszą one poszukać źródła pożywienia, zebrać pożywienie i znaleźć bezpieczne i opłacalne miejsce dla ula. Jeden cel jest realizowany kosztem innego celu. Agent powinien zostać zaprogramowany tak, aby rozważał kompromis w większym interesie roju. W miarę możliwości funkcja optymalizacji dla agenta powinna przynieść optymalne rozwiązanie dla więcej niż jednego celu, ale wzajemna wyłączność nie jest możliwa. W takich przypadkach agent powinien być w stanie działać bez centralnej kontroli i decydować o obiektywnej wadze na podstawie kontekstu środowiskowego i powinien faworyzować cel, który spełni ogólny cel roju przez dłuższy okres, zamiast decydować na podstawie krótkoterminowej strategii. Celem optymalizacji jako całości jest osiągnięcie optymalności Pareto dla roju:

Optymalność Pareto to formalnie zdefiniowana koncepcja stosowana do określania, kiedy przydział jest optymalny. Alokacja nie jest optymalna dla Pareto, jeśli istnieje alternatywna alokacja, w której można dokonać poprawy dobrostanu przynajmniej jednego uczestnika bez zmniejszania dobrostanu innego uczestnika. Jeśli istnieje przeniesienie spełniające ten warunek, przeniesienie nazywa się poprawą Pareto. Gdy dalsze ulepszenia Pareto nie są możliwe, alokacja jest optymalna dla Pareto.

Często Zadawane Pytania

P: Jaka jest różnica między paradygmatem przetwarzania rozproszonego a inteligencją roju? W przypadku przetwarzania rozproszonego dzielimy także jednostki pracy na części przetwarzane przez poszczególne węzły.

O: Podstawowa różnica między tymi dwoma typami systemów polega na tym, że rozproszone systemy obliczeniowe są kontrolowane centralnie. Istnieje węzeł główny lub jednostka przetwarzająca, która śledzi wszystkie węzły robocze i przydzielone jednostki robocze na podstawie ich dostępności. Ramy utrzymują również poziom nadmiarowości, dzięki czemu system jest niezawodny w przypadku awarii jednego z węzłów roboczych. W przypadku inteligentnego zachowania roju wykazywanego przez istoty społeczne nie ma scentralizowanej kontroli, a wszyscy agenci działają niezależnie w ramach swoich zasad działania. Agenci są samoorganizujący się i współpracują intuicyjnie i domyślnie zamiast jawnej współpracy zarządzanej przez centralną jednostkę kontrolną.

P: W jaki sposób systemy oparte na algorytmach SI naśladują zjawisko naturalne, takie jak wytwarzanie feromonów?

O: Feromon to substancja chemiczna, która jest wydzielana przez mrówki w drodze do i ze źródła pożywienia, co sygnalizuje innym mrówkom, że w pobliżu znajduje się źródło pożywienia. Ta substancja chemiczna jest podstawowym mechanizmem, w którym mrówki komunikują się ze sobą, a zmienne stężenie feromonu wskazuje mrówkom różne rzeczy. W przypadku środków sztucznych środek utrzymuje kwantyfikację feromonu jako wartości liczbowej, która jest zwiększana w celu wskazania dodatkowych feromonów, a także istnieje proces odparowywania oparty na parametrze czasu. W pewien sposób zachowanie jest symulowane w celu dopasowania do zjawiska naturalnego.

P: Jakie są niektóre przypadki użycia i prawdziwe zastosowania sztucznej inteligencji roju?

O: Zasady SI można zastosować do różnorodnego zestawu problemów i przypadków użycia w różnych branżach. Widzieliśmy już przypadek użycia w obliczeniach rozproszonych do równoważenia obciążenia węzłów. Możemy również wdrażać algorytmy SI w planowaniu logistycznym i optymalizacji łańcucha dostaw, routerach sieciowych i komunikacyjnych, inteligentnym sterowaniu ruchem i flotą, optymalizacją operacji w fabryce i optymalizacją siły roboczej w operacjach obsługi klienta.

Podsumowanie

Widzieliśmy interesujący aspekt budowy AI. Natura ma najlepszy algorytm, jeśli chodzi o harmonijne zarządzanie niezwykle złożonym ekosystemem o ogromnej skali. Czerpiemy inspirację z natury i niektórych najmniejszych stworzeń, które mają małe mózgi, a zatem bardzo małą liczbę neuronów w porównaniu do ludzi. Te małe stworzenia są jednak w stanie wspólnie osiągać wyczyny znacznie większe niż suma ich indywidualnych możliwości. Zasady działania tych społeczności ,stworzenia nie mogą być ignorowane, gdy dążymy do zbudowania systemów AI, które uzupełniają i zwiększają ludzkie możliwości. Widzieliśmy niektóre z podstawowych pojęć naturalnej inteligencji roju i niektóre zasady, które musimy wziąć pod uwagę przy opracowywaniu nowoczesnych systemów opartych na SI. Próbowaliśmy przedstawić zbiorowe zachowania w formie matematycznej i wyprowadzić niektóre wzorce w opracowywaniu zachowania algorytmicznego dla sztucznych czynników z algorytmami PSO i ACO. W tym rozdziale dokonaliśmy przeglądu dwóch struktur obliczeniowych i bibliotek, MASON i Opt4J, które można łatwo wykorzystać do różnych eksperymentów i zaawansowanych analiz. Te biblioteki zapewniają skuteczne warstwy wizualizacji. Omówiliśmy przypadek użycia do równoważenia obciążenia serwerów w rozproszonym środowisku komputerowym. W następnym rozdziale po raz kolejny zaczerpniemy inspirację z natury i przyjrzymy się ważnemu algorytmowi zwanemu uczeniem się przez wzmocnienie. W przeciwieństwie do nauki nadzorowanej,

uczenie się przez wzmocnienie wykorzystuje nagrodę i karę jako wkład w zachowania uczenia się dla sztucznych środków.