Swarm Intelligence

این یک مثال ساده از رفتار جمعی یا Swarm behavior است که افراد برای رسیدن به یک هدف نهایی همکاری می‌کنند. این روش مؤثرتر از زمانی است که افراد جداگانه عمل کنند. Swarm را می‌توان به صورت مجموعه‌ای سازمان یافته از موجوداتی تعریف کرد که با یکدیگر همکاری می‌کنند. در کاربردهای محاسباتی Swarm intelligence از موجوداتی مانند مورچه‌ها، زنبورها، موریانه‌ها، دسته‌های ماهیان و دسته‌ی پرندگان الگو برداری می‌شود. در این نوع اجتماعات هر یک از موجودات ساختار نسبتاً ساده‌ای دارند ولی رفتار جمعی آنها بی‌نهایت پیچیده است. برای مثال در کولونی مورچه‌ها هر یک از مورچه‌ها یک کار ساده‌ی مخصوص را انجام می‌دهد ولی به طور جمعی عمل و رفتار مورچه‌ها، ساختن بهینه‌ی لایهء محافظت از ملکه و نوزادان، تمیز کردن لانه، یافتن بهترین منابع غذایی و بهینه‌سازی استراتژی حمله را تضمین می‌کند. رفتار کلی یک Swarm به صورت غیر خطی از آمیزش رفتارهای تک‌تک اجتماع بدست می‌آید یا به عبارتی یک رابطه‌ی بسیار پیچیده بین رفتار جمعی و رفتار فردی یک اجتماع وجود دارد. رفتار جمعی فقط وابسته به رفتار فردی افراد اجتماع نیست بلکه به چگونگی تعامل میان افراد نیز وابسته است. تعامل بین افراد، تجربه‌ی افراد درباره‌ی محیط را افزایش می‌دهد و موجب پیشرفت اجتماع می‌شود. ساختار اجتماعی Swarm بین افراد مجموعه کانالهای ارتباطی ایجاد می‌کند که طی آن افراد می‌توانند به تبادل تجربه‌های شخصی بپردازند.

Particle Swarm Optimization

الگوریتم PSO یک الگوریتم جستجوی اجتماعی است که از روی رفتار اجتماعی دسته‌های پرندگان مدل شده است. در ابتدا این الگوریتم به منظور کشف الگوهای حاکم بر پرواز همزمان پرندگان و تغییر ناگهانی مسیر آنها و تغییر شکل بهینه‌ی دسته به کار گرفته شد. در PSO، particleها در فضای جستجو جاری می‌شوند. تغییر مکان particleها در فضای جستجو تحت تأثیر تجربه و دانش خودشان و همسایگانشان است. بنابراین موقعیت دیگر particleهای Swarm روی چگونگی جستجوی یک particle اثر می‌گذارد. نتیجه‌ی مدل‌سازی این رفتار اجتماعی فرایند جستجویی است که particleها به سمت نواحی موفق میل می‌کنند. Particleها در Swarm از یکدیگر می‌آموزند و بر مبنای دانش بدست آمده به سمت بهترین همسایگان خود می‌روند.

اساس کار PSO بر این اصل استوار است که در هر لحظه هر particle مکان خود را در فضای جستجو با توجه به بهترین مکانی که تاکنون در آن قرار گرفته است و بهترین مکانی که در کل همسایگی‌اش وجود دارد، تنظیم می‌کند.

```
while(not_termination)
```

```
        UpdateLocalBests()
        UpdateGlobalBest()
        for each particle
                R1=uniform random number
                R2=uniform random number
                V[I][J]=w*V[I][J] + C1*R1*(X_lbest[I][J]-X[I][J]) + C2*R2*(X_gbest[J]-X[I][J])
                X[I][J] = X[I][J]+V[I][J]
        end for
end while
```

Ant Colony Optimization

مورچه ها در مسیرهای عبوری خود ماده شیمیایی بجا می گذارند که بوسیله این ماده مورچه ها هنگام پیدا کردن مواد غذایی دیگر همنوعان خود را با خبر می سازند. مورچه هایی که مسیر منتهی به ماده غذایی را طی می کنند هم از خود ماده شیمیایی را بجا می گذارند. به این ترتیب هر مورچه ای مسیری را دنبال می کند که مورچه های بیشتری از آنجا عبور کرده باشند و این یعنی کوتاهترین مسیر تا غذا.

الگوریتم کلونی مورچه برای اولین بار توسط دوریگو (Dorigo) و همکارانش به عنوان یک راه حل چند عامله (Multi Agent) برای مسائل مشکل بهینه سازی مثل فروشنده دوره گرد (TSP :Traveling Sales Person) ارائه شد.

مورچه ها هنگام راه رفتن از خود ردی از ماده شیمیایی فرومون(Pheromone) بجای می گذارند البته این ماده بزودی تبخیر می شد ولی در کوتاه مدت بعنوان رد مورچه بر سطح زمین باقی می ماند. یک رفتار پایه ای ساده در مورچه های وجود دارد :

آنها هنگام انتخاب بین دو مسیر بصورت احتمالاتی(Statistical) مسیری را انتخاب می کنند که فرومون بیشتری داشته باشد یا بعبارت دیگر مورچه های بیشتری قبلا از آن عبور کرده باشند. حال دقت کنید که همین یک تمهید ساده چگونه منجر به پیدا کردن کوتاهترین مسیر خواهد شد .

نکته بسیار با اهمیت این است که هر چند احتمال انتخاب مسیر پر فرومون ت توسط مورچه ها بیشتر است ولی این کماکان احتمال است و قطعیت نیست. یعنی اگر مسیر CED پرفرومون تر از CFD باشد به هیچ عنوان نمی شود نتیجه گرفت که همه مورچه ها از مسیرCED عبور خواهند کرد بلکه تنها می توان گفت که مثلا ۹۰٪ مورچه ها از مسیر کوتاهتر عبور خواهند کرد.

نکته دیگر مسئله تبخیر شدن فرومون بر جای گذاشته شده است. برفرض اگر مانع در مسیر AB برداشته شود و فرومون تبخیر نشود مورچه ها همان مسیر قبلی را طی خواهند کرد. ولی در حقیقت این طور نیست. تبخیر شدن فرومون و احتمال به مورچه ها امکان پیدا کردن مسیر کوتاهتر جدید را می دهند.

Pseudo-code

```
procedure ACO
        while(not_termination)
                generateSolutions();
                pheromoneUpdate();
                UpdateActionProbabilitys();
        end while
 end procedure
```

Arc Selection:

An ant will move from node i to node j with probability

$$p_{ij}^{k}(t) = \frac{\left[\tau_{ij}(t)\right]^{\alpha}.\left[\eta_{ij}\right]^{\beta}}{\sum_{\ell \in J_i^k}\left[\tau_{il}(t)\right]^{\alpha}.\left[\eta_{il}\right]^{\beta}}$$

where,

τ_{ij} is the amount of pheromone on arc i,j

α is a parameter to control the influence of τ_{ij}

η_{ij} is the desirability of arc i,j (a priori knowledge, typically $1/d_{ij}$)

β is a parameter to control the influence of η_{ij}

$$\tau_{ij}(t) = (1-\rho).\tau_{ij}(t) + \Delta\tau_{ij}(t)$$

where,

τi,j is the amount of pheromone on a given arc i,j

ρ is the rate of pheromone evaporation

and Δτi,j is the amount of pheromone deposited, typically given by

$$\Delta\tau_{ij}^{k} = 1/L^{k}(t) \quad if\ \ (i,j) \in T^{k}(t)\ else\ 0.$$

where L^k is the cost of the kth ant's tour (typically length).

Genetic Algorithm

الگوریتم های ژنتیک یکی از الگوریتم های جستجوی تصادفی است که ایده آن برگرفته از طبیعت می باشد. الگوریتم های ژنتیک در حل مسائل بهینه سازی کاربرد فراوانی دارند . به عنوان مثال می توان به مسئله فروشنده دوره گرد اشاره کرد . در طبیعت از ترکیب کروموزوم های بهتر ، نسل های بهتری پدید می آیند . در این بین گاهی اوقات جهش هایی نیز در کروموزوم ها روی می دهد که ممکن است باعث بهتر شدن نسل بعدی شوند. الگوریتم ژنتیک نیز با استفاده از این ایده اقدام به حل مسائل می کند.

Fitness (برازش)

با استفاده از این عملگر ، میزان بهینگی هر کروموزوم را تعیین می کنیم . به عنوان مثال در مسئله فروشنده دوره گرد ، تورهای با مسافت کمتر بهینه تر هستند . و یا در مسئله n-وزیر تعداد برخوردهای کمتر باعث بهینگی بیشتر کروموزوم می شود . بنابراین می توان نتیجه گرفت که عملگر Fitness برای هر کروموزوم احتمالی را نسبت می دهد که این احتمال ، همان احتمال ترکیب شدن کروموزم برای تولید نسل های آینده را نشان می دهد . بدیهی است که کروموزوم های بهینه تر شانس بیشتری برای ترکیب با دیگر کروموزوم ها خواهند داشت . بنابراین احتمالی که به آنها نیز نسبت می دهیم باید بیشتر باشد .

Selection (انتخاب)

پس از آنکه عملگر Fitness بر روی جمعیت فعلی انجام پذیرفت ، عملگر Selection کار خود را آغاز می کند . وظیفه این عملگر انتخاب کروموزوم هایی از میان جمعیت فعلی برای ترکیب شدن می باشد . عملگر Selection از مقادیر تولید شده برای هر کروموزوم توسط عملگر Fitness در مرحله قبل استفاده کرده و از میان جمعیت ، کروموزوم هایی را برای ترکیب شدن انتخاب می کند . کروموزوم های با مقدار Fitness بیشتر شانس بیشتری برای انتخاب شدن خواهند داشت. در صورتی که تعداد جمعیت K کروموزوم باشد ، جمعیت میانی حاصل از اعمال عملگر Selection نیز باید K کروموزوم داشته باشد . این بدان مفهوم است که کروموزوم های بهتر در جمعیت میانی ممکن است چند بار تکرار شوند .

روش های مختلفی برای انتخاب کروموزوم ها وجود دارد . یکی از معمولترین روش ها روش رقابتی می باشد . در روش رقابتی دو یا چند کروموزوم را به طور تصادفی انتخاب کرده و از میان آنها کروموزومی که Fitness آن بهتر از دیگر کروموزوم های انتخاب شده باشد ، انتخاب می شود . این عمل به تعداد کروموزوم های جمعیت اولیه انجام می شود . محصول نهایی عملگر انتخاب جمعیت میانی می باشد که از این جمعیت در مراحل بعدی استفاده خواهیم کرد .

Crossover (ادغام)

پس از انجام دو مرحله فوق ، شرایط برای ترکیب کروموزوم ها با یکدیگر مهیا شده است . عملگر Crossover از کروموزوم های جمعیت میانی استفاده کرده و آنها را با هم ترکیب می کند . عملگر Crossover با این امید اقدام به ترکیب کروموزوم ها می کند که شاید کروموزوم فرزند حاصل از ترکیب دو یا چند کروموزوم پدر بهینه تر باشد .

ادغام کروموزوم ها نیز روش های مختلفی دارد که از این میان سه روش از محبوبیت بیشتری برخوردارند :

۱. ادغام تک نقطه ای

در این روش از میان کروموزوم های جمعیت میانی دو کروموزوم را انتخاب می کنیم . سپس به طور تصادفی نقطه ای از کروموزوم را انتخاب کرده و تمام ژن های بعد از این نقطه را در دو کروموزوم تعویض می کنیم .

۲. ادغام دو نقطه ای

در این روش نیز از میان کروموزوم های جمعیت میانی دو کروموزوم را انتخاب می کنیم . سپس به طور تصادفی دو نقطه از کروموزوم را انتخاب کرده و تمام ژن های بین این دو نقطه را در دو کروموزوم تعویض می کنیم .

۳. ادغام توسط ماسک

در این روش از یک ماسک برای ادغام استفاده می کنیم . بدین صورت که آرایه را که به تعداد ژن ها دارای عنصر است ایجاد می کنیم . عناصر این آرایه فقط می تواند مقادیر صفر یا یک بپذیرد . عناصر آرایه ماسک را به طور تصادفی مقدار دهی می کنیم . حال با استفاده از این ماسک دو کروموزوم را ادغام می کنیم . مقدار صفر در آرایه ماسک نشان می دهد که ژن باید از کروموزوم اول انتخاب شود . مقدار یک نیز در آرایه ماسک مشخص می کند که ژن باید از کروموزوم دوم انتخاب شود.

روش های یاد شده ، روشهایی کلی می باشند و گاهی اوقات باید تغییراتی در آنها ایجاد کنیم .

Mutation (جهش)

عملگر جهش هنگامی که بر روی کروموزمومی اعمال می شود باعث بروز جهش در آن کروموزوم می شود. روش معمول برای جهش تغییر دادن یک یا چند ژن از کروموزوم بطور تصادفی می باشد .

```
Population = GeneratePopulation(K);
For I = 1 to MaxGenerations
        Fitness(Population);
        If any of chromosomes is optimal Then
                Break;
```

```
        ParentSelection(Population);
        Offspring =  Crossover(Population);
        Mutate(Offspring);
EndFor
```