Завдання на роботу

ЗАВДАННЯ НА РОБОТУ

1. Написати код, що реалізує алгоритм відповідно до варіанта.
2. Експерементально візначити оптимальні значення основних параметрів алгоритму.
3. З його допомогою знайти екстремуми наступних функцій.
4. Побудувати графіки залежності швидкості збіжності алгоритму від розміру популяції.

Алгоритм

В роботі реалізовано оптимізацію роєм частинок мовою C++. Посилання на гітхаб репозиторій: Bohdan628318ylypchenko/SwarmOptimization.

Рішення складається з частин:

- swarm-core: бібліотека, що реалізує алгоритм рою частинок
- swarm-tests: юніт-тести для бібліотеки
- swarm-demo: демонстраційний додаток – користувач swarm-core, реалізує перебір параметрів, оптимізацію із підібраними параметрами.

Критерій зупинки – виконання заданої кількості ітерацій.

Реалізація містить модифікації:

1. Запобігання вибуху рою: якщо модуль швидкості частки перевищує задане значення, швидкість примусово зменшується до максимальної дозволеної (файл swarm_particle.cpp):

```
128
129         if (speed.mod() >= max_speed_mod)
130         {
131             speed.mut_normalize()
132                  .mut_multiply_on_scalar(max_speed_mod);
133         }
```

   Значення max_speed_mod задається як параметр оптимізації.
2. Множник інерції (файл swarm_particle.cpp):

```
speed
    .mut_multiply_on_scalar(w)
    .mut_add(
        r1.mut_multiply_on_scalar(local_weight)
          .mut_coordinate_multiply((local_min_copy = local_min.pos).mut_subtract(pos))
    )
    .mut_add(
        r2.mut_multiply_on_scalar(global_weight)
          .mut_coordinate_multiply((global_min_copy = global_min_pos).mut_subtract(pos))
    );
```

Множник інерції w

Множник інерції змінюється наступним чином (файл swarm_algorithm.cpp):

```
if (i % comparation_interval == 0)
{
    if (abs(current_min.value - last_min.value) / (real_t)comparation_interval < epsilon)
        w *= 0.5;
    last_min = current_min;
}
```

Кожні comparation_interval (значення comparation_interval є параметром оптимізації) ітерацій порівнюється значення поточного глобального мінімуму із значенням глобального мінімуму comparation_interval ітерацій тому. Якщо модуль різниці є меншим за epsilon (параметр оптимізації) – множник інерції зменшується вдвічі. Така зміна множника інерції адаптує множник до поведінки цільової функції, на відміну від «ручного» задання закону зміни множника.

Параметри алгоритму:

```
Solution optimize(
    natural_t dim, natural_t particle_count,
    real_t min_pos_bound, real_t max_pos_bound,
    real_t local_weight, real_t global_weight,
    real_t max_speed_mod,
    size_t iteration_count,
    real_t initial_w, size_t comparation_interval, real_t epsilon,
    TargetFunction target_function
);
```

- dim: розмірність простору
- particle_count: кількість частинок
- min_pos_bound: нижнє обмеження області пошуку
- max_pos_bound: верхнє обмеження області пошуку
- max_speed_mod: максимальне допустиме значення модуля швидкості
- iteration_count: кількість ітерацій алгоритму
- initial_w: початкове значення множника w
- comparation_interval: кількість ітерацій між порівнянням мінімумів для можливого зменшення w

- epsilon: параметр порівняння мінімумів (застосовується лише для можливого зменшення w)
- target_function: цількова функція: std::function<real_t(const VN&)>

Перебір параметрів

Було виконано перебір значень параметрів:

- particle_count: [100, 1000, 2000, 4000]
- local_weight/global_weight: [(1;5), (2;4), (3;3), (4;2), (5;1)]
- iteration_count: [100, 1000, 5000, 10000, 20000]
- initial_w: [1, 1.2]
- comparation_interval: [10, 100, 1000]
- epsilon: [0.01, 0.001]

Для кожної із 5 цільових функцій було протестовано 4 * 5 * 5 * 2 * 3 * 2 = 1200 комбінацій параметрів. Із отриманого набору значень для кожної функції було обрано топ-20 комбінацій параметрів.

Результати тестування (файл automatic optimization.txt):

```
PS C:\Users\Bohdan\swarm> .\swarm-demo.exe

=== griewank: ===

total iteration count = 1200

Top 20 best parameter sets:

i = 0: value: 0; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 5000; initial_w: 1;
comp_interval: 100; e: 0.001

i = 1: value: 0; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 10000; initial_w: 1;
comp_interval: 100; e: 0.01

i = 2: value: 0; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 20000; initial_w: 1;
comp_interval: 100; e: 0.01

i = 3: value: 0; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 20000; initial_w: 1;
comp_interval: 100; e: 0.001

i = 4: value: 0; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 20000; initial_w: 1;
comp_interval: 1000; e: 0.01

i = 5: value: 0; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 10000; initial_w:
1.2; comp_interval: 100; e: 0.001

i = 6: value: 0; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 10000; initial_w: 1;
comp_interval: 10; e: 0.01

i = 7: value: 0; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 10000; initial_w:
1.2; comp_interval: 100; e: 0.01

i = 8: value: 0; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 5000; initial_w: 1;
comp_interval: 10; e: 0.01

i = 9: value: 0; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 10000; initial_w:
1.2; comp_interval: 10; e: 0.001

i = 10: value: 0; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 10000; initial_w: 1;
comp_interval: 1000; e: 0.001

i = 11: value: 0; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 5000; initial_w:
1.2; comp_interval: 10; e: 0.001
```

i = 12: value: 0; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 20000; initial_w: 1; comp_interval: 10; e: 0.01

i = 13: value: 0; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 5000; initial_w: 1; comp_interval: 100; e: 0.01

i = 14: value: 0; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 10000; initial_w: 1; comp_interval: 1000; e: 0.01

i = 15: value: 0; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 5000; initial_w: 1.2; comp_interval: 10; e: 0.01

i = 16: value: 0; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 10000; initial_w: 1; comp_interval: 100; e: 0.001

i = 17: value: 0; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 10000; initial_w: 1.2; comp_interval: 10; e: 0.01

i = 18: value: 0; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 5000; initial_w: 1.2; comp_interval: 100; e: 0.01

i = 19: value: 0; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 5000; initial_w: 1.2; comp_interval: 100; e: 0.001

=== rastringin: ===

total iteration count = 1200

Top 20 best parameter sets:

i = 0: value: 0; particle_count: 2000; local_weight: 4; global_weight: 2; iteration_count: 20000; initial_w: 1; comp_interval: 100; e: 0.001

i = 1: value: 0; particle_count: 2000; local_weight: 4; global_weight: 2; iteration_count: 10000; initial_w: 1; comp_interval: 100; e: 0.001

i = 2: value: 5.68434e-14; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 20000; initial_w: 1.2; comp_interval: 1000; e: 0.01

i = 3: value: 5.68434e-14; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 20000; initial_w: 1.2; comp_interval: 1000; e: 0.001

i = 4: value: 5.68434e-14; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 10000; initial_w: 1.2; comp_interval: 1000; e: 0.01

i = 5: value: 5.68434e-14; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 20000; initial_w: 1.2; comp_interval: 100; e: 0.001

i = 6: value: 1.13687e-13; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 10000; initial_w: 1.2; comp_interval: 100; e: 0.001

i = 7: value: 1.13687e-13; particle_count: 2000; local_weight: 4; global_weight: 2; iteration_count: 20000; initial_w: 1; comp_interval: 1000; e: 0.01

i = 8: value: 1.13687e-13; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 20000; initial_w: 1.2; comp_interval: 10; e: 0.01

i = 9: value: 1.13687e-13; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 20000; initial_w: 1.2; comp_interval: 100; e: 0.01

i = 10: value: 1.13687e-13; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 10000; initial_w: 1.2; comp_interval: 100; e: 0.01

i = 11: value: 1.13687e-13; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 20000; initial_w: 1.2; comp_interval: 10; e: 0.001

i = 12: value: 1.7053e-13; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 20000; initial_w: 1; comp_interval: 1000; e: 0.001

i = 13: value: 1.7053e-13; particle_count: 2000; local_weight: 4; global_weight: 2; iteration_count: 20000; initial_w: 1.2; comp_interval: 1000; e: 0.001

i = 14: value: 2.84217e-13; particle_count: 2000; local_weight: 4; global_weight: 2; iteration_count: 20000; initial_w: 1; comp_interval: 1000; e: 0.001

i = 15: value: 3.41061e-13; particle_count: 2000; local_weight: 5; global_weight: 1; iteration_count: 20000; initial_w: 1; comp_interval: 100; e: 0.001

i = 16: value: 5.68434e-13; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 10000; initial_w: 1.2; comp_interval: 10; e: 0.001

i = 17: value: 6.82121e-13; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 10000; initial_w: 1.2; comp_interval: 10; e: 0.01

i = 18: value: 1.25056e-12; particle_count: 2000; local_weight: 4; global_weight: 2; iteration_count: 20000; initial_w: 1.2; comp_interval: 1000; e: 0.01

i = 19: value: 4.03588e-12; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 5000; initial_w: 1.2; comp_interval: 100; e: 0.01

=== rosenbrock: ===

Top 20 best parameter sets:

i = 0: value: 3.86686; particle_count: 2000; local_weight: 4; global_weight: 2; iteration_count: 20000; initial_w: 1.2; comp_interval: 100; e: 0.01

i = 1: value: 10.2601; particle_count: 2000; local_weight: 5; global_weight: 1; iteration_count: 20000; initial_w: 1.2; comp_interval: 10; e: 0.001

i = 2: value: 10.2606; particle_count: 2000; local_weight: 5; global_weight: 1; iteration_count: 10000; initial_w: 1.2; comp_interval: 10; e: 0.001

i = 3: value: 10.2616; particle_count: 2000; local_weight: 5; global_weight: 1; iteration_count: 5000; initial_w: 1.2; comp_interval: 10; e: 0.001

i = 4: value: 10.2654; particle_count: 2000; local_weight: 5; global_weight: 1; iteration_count: 20000; initial_w: 1.2; comp_interval: 10; e: 0.01

i = 5: value: 10.2663; particle_count: 2000; local_weight: 5; global_weight: 1; iteration_count: 10000; initial_w: 1.2; comp_interval: 10; e: 0.01

i = 6: value: 10.2678; particle_count: 2000; local_weight: 5; global_weight: 1; iteration_count: 5000; initial_w: 1.2; comp_interval: 10; e: 0.01

i = 7: value: 10.2993; particle_count: 2000; local_weight: 5; global_weight: 1; iteration_count: 1000; initial_w: 1.2; comp_interval: 10; e: 0.001

i = 8: value: 10.3364; particle_count: 2000; local_weight: 5; global_weight: 1; iteration_count: 1000; initial_w: 1.2; comp_interval: 10; e: 0.01

i = 9: value: 12.9576; particle_count: 2000; local_weight: 4; global_weight: 2; iteration_count: 20000; initial_w: 1.2; comp_interval: 10; e: 0.01

i = 10: value: 21.2533; particle_count: 100; local_weight: 5; global_weight: 1; iteration_count: 20000; initial_w: 1.2; comp_interval: 10; e: 0.01

i = 11: value: 25.2886; particle_count: 4000; local_weight: 1; global_weight: 5; iteration_count: 20000; initial_w: 1.2; comp_interval: 100; e: 0.01

i = 12: value: 25.4619; particle_count: 2000; local_weight: 5; global_weight: 1; iteration_count: 20000; initial_w: 1.2; comp_interval: 100; e: 0.01

i = 13: value: 25.4629; particle_count: 2000; local_weight: 5; global_weight: 1; iteration_count: 10000; initial_w: 1.2; comp_interval: 100; e: 0.01

i = 14: value: 25.4646; particle_count: 2000; local_weight: 5; global_weight: 1; iteration_count: 5000; initial_w: 1.2; comp_interval: 100; e: 0.01

i = 15: value: 25.4666; particle_count: 2000; local_weight: 5; global_weight: 1; iteration_count: 20000; initial_w: 1.2; comp_interval: 100; e: 0.001

i = 16: value: 25.467; particle_count: 2000; local_weight: 5; global_weight: 1; iteration_count: 10000; initial_w: 1.2; comp_interval: 100; e: 0.001

i = 17: value: 25.4689; particle_count: 2000; local_weight: 5; global_weight: 1; iteration_count: 5000; initial_w: 1.2; comp_interval: 100; e: 0.001

i = 18: value: 25.7142; particle_count: 4000; local_weight: 1; global_weight: 5; iteration_count: 20000; initial_w: 1.2; comp_interval: 10; e: 0.01

i = 19: value: 26.499; particle_count: 4000; local_weight: 1; global_weight: 5; iteration_count: 20000; initial_w: 1.2; comp_interval: 10; e: 0.001

=== simple: ===

total iteration count = 1200

Top 20 best parameter sets:

i = 0: value: 1.89535e-37; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 20000; initial_w: 1.2; comp_interval: 10; e: 0.001

i = 1: value: 5.59124e-34; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 20000; initial_w: 1.2; comp_interval: 10; e: 0.01

i = 2: value: 7.9184e-33; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 20000; initial_w: 1.2; comp_interval: 100; e: 0.01

i = 3: value: 1.52489e-32; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 20000; initial_w: 1; comp_interval: 100; e: 0.001

i = 4: value: 9.88347e-32; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 10000; initial_w: 1.2; comp_interval: 10; e: 0.001

i = 5: value: 1.04712e-31; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 20000; initial_w: 1; comp_interval: 100; e: 0.01

i = 6: value: 1.68742e-31; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 20000; initial_w: 1.2; comp_interval: 100; e: 0.001

i = 7: value: 6.86586e-31; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 20000; initial_w: 1; comp_interval: 10; e: 0.001

i = 8: value: 8.44428e-31; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 10000; initial_w: 1.2; comp_interval: 10; e: 0.01

i = 9: value: 8.46873e-31; particle_count: 4000; local_weight: 5; global_weight: 1; iteration_count: 20000; initial_w: 1.2; comp_interval: 1000; e: 0.001

i = 10: value: 9.34761e-31; particle_count: 4000; local_weight: 5; global_weight: 1; iteration_count: 20000; initial_w: 1; comp_interval: 1000; e: 0.001

i = 11: value: 9.34942e-31; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 20000; initial_w: 1; comp_interval: 1000; e: 0.01

i = 12: value: 1.76997e-30; particle_count: 4000; local_weight: 5; global_weight: 1; iteration_count: 20000; initial_w: 1; comp_interval: 1000; e: 0.01

i = 13: value: 2.29026e-30; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 10000; initial_w: 1; comp_interval: 100; e: 0.01

i = 14: value: 3.00098e-30; particle_count: 4000; local_weight: 5; global_weight: 1; iteration_count: 20000; initial_w: 1.2; comp_interval: 1000; e: 0.01

i = 15: value: 6.83583e-30; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 10000; initial_w: 1; comp_interval: 100; e: 0.001

i = 16: value: 2.30303e-29; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 10000; initial_w: 1.2; comp_interval: 100; e: 0.01

i = 17: value: 3.28077e-29; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 20000; initial_w: 1.2; comp_interval: 1000; e: 0.001

i = 18: value: 6.88718e-29; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 10000; initial_w: 1.2; comp_interval: 100; e: 0.001

i = 19: value: 1.20534e-28; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 20000; initial_w: 1; comp_interval: 1000; e: 0.001

=== ackley: ===

total iteration count = 1200

Top 20 best parameter sets:

i = 0: value: 7.54952e-15; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 5000; initial_w: 1.2; comp_interval: 10; e: 0.001

i = 1: value: 7.54952e-15; particle_count: 4000; local_weight: 5; global_weight: 1; iteration_count: 20000; initial_w: 1; comp_interval: 100; e: 0.001

i = 2: value: 7.54952e-15; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 20000; initial_w: 1.2; comp_interval: 10; e: 0.001

i = 3: value: 7.54952e-15; particle_count: 4000; local_weight: 5; global_weight: 1; iteration_count: 10000; initial_w: 1; comp_interval: 100; e: 0.001

i = 4: value: 7.54952e-15; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 20000; initial_w: 1.2; comp_interval: 100; e: 0.01

i = 5: value: 7.54952e-15; particle_count: 2000; local_weight: 4; global_weight: 2; iteration_count: 20000; initial_w: 1.2; comp_interval: 10; e: 0.01

i = 6: value: 7.54952e-15; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 10000; initial_w: 1.2; comp_interval: 10; e: 0.001

i = 7: value: 1.11022e-14; particle_count: 4000; local_weight: 4; global_weight: 2; iteration_count: 10000; initial_w: 1.2; comp_interval: 100; e: 0.01

i = 8: value: 1.11022e-14; particle_count: 2000; local_weight: 4; global_weight: 2; iteration_count: 10000; initial_w: 1.2; comp_interval: 10; e: 0.01

i = 9: value: 1.46549e-14; particle_count: 2000; local_weight: 4; global_weight: 2; iteration_count: 20000; initial_w: 1.2; comp_interval: 1000; e: 0.001

i = 10: value: 1.46549e-14; particle_count: 2000; local_weight: 4; global_weight: 2; iteration_count: 20000; initial_w: 1.2; comp_interval: 1000; e: 0.01

i = 11: value: 1.46549e-14; particle_count: 2000; local_weight: 4; global_weight: 2; iteration_count: 20000; initial_w: 1; comp_interval: 10; e: 0.001

i = 12: value: 1.46549e-14; particle_count: 2000; local_weight: 4; global_weight: 2; iteration_count: 20000; initial_w: 1; comp_interval: 100; e: 0.01

i = 13: value: 1.46549e-14; particle_count: 2000; local_weight: 5; global_weight: 1; iteration_count: 20000; initial_w: 1.2; comp_interval: 1000; e: 0.001

i = 14: value: 1.46549e-14; particle_count: 2000; local_weight: 4; global_weight: 2; iteration_count: 20000; initial_w: 1.2; comp_interval: 100; e: 0.001

```
i = 15: value: 1.46549e-14; particle_count: 2000; local_weight: 4; global_weight: 2; iteration_count: 20000;
initial_w: 1; comp_interval: 100; e: 0.001

i = 16: value: 1.46549e-14; particle_count: 2000; local_weight: 4; global_weight: 2; iteration_count: 5000;
initial_w: 1.2; comp_interval: 100; e: 0.001

i = 17: value: 1.46549e-14; particle_count: 2000; local_weight: 4; global_weight: 2; iteration_count: 10000;
initial_w: 1.2; comp_interval: 100; e: 0.001

i = 18: value: 1.46549e-14; particle_count: 2000; local_weight: 4; global_weight: 2; iteration_count: 5000;
initial_w: 1; comp_interval: 10; e: 0.001

i = 19: value: 1.46549e-14; particle_count: 2000; local_weight: 4; global_weight: 2; iteration_count: 5000;
initial_w: 1; comp_interval: 100; e: 0.001

PS C:\Users\Bohdan\swarm>
```

Для функцій griewank, rastringin, sphere(simple), ackley задовільними є будь-які набори параметрів із топ-20. На жаль для функції Розенброка досягнути абсолютного мінімуму (0) не вдалось: найближче значення – 3.86.

В якості остаточного набору параметрів оберемо той, де функція Розенброка набуває значення 3.86:

particle_count: 2000; local_weight: 4; global_weight: 2; iteration_count: 20000; initial_w: 1.2; comp_interval: 100; e: 0.01

Результат оптимізації з фінальними параметрами для кожної функції:

```
PS D:\> .\swarm-demo.exe

=== griewank final run: ===

value = 0.0270664

coord 0 = -8.38419e-09

coord 1 = -1.12733e-08

coord 2 = 6.28227e-09

coord 3 = -3.20084e-10

coord 4 = -7.0073

coord 5 = -7.67228

coord 6 = 1.54277e-08

coord 7 = -2.20997e-08

coord 8 = 2.33247e-08

coord 9 = -2.22778e-08

coord 10 = -3.23583e-08

coord 11 = 2.62084e-08

coord 12 = -3.40656e-08

coord 13 = -4.56479e-08

coord 14 = 4.21444e-09

coord 15 = 2.06126e-08

coord 16 = -2.33898e-08

coord 17 = 1.79709e-08

coord 18 = 1.67989e-08

coord 19 = -1.30143e-08

coord 20 = 8.47474e-10
```

coord 21 = 1.62615e-08
coord 22 = 3.94276e-08
coord 23 = -4.45988e-08
coord 24 = 1.85851e-08
coord 25 = -3.18992e-08
coord 26 = -2.79864e-08
coord 27 = -1.53901e-08
coord 28 = -3.88244e-08
coord 29 = -5.15383e-08
coord 30 = 9.29938e-08
coord 31 = -3.75856e-08
coord 32 = -3.79308e-08
coord 33 = -5.47408e-08
coord 34 = 5.42703e-08
coord 35 = -3.74594e-08
coord 36 = -9.7632e-08
coord 37 = 7.63106e-09
coord 38 = -1.41128e-07
coord 39 = -9.56482e-08
coord 40 = 3.98191e-08
coord 41 = -1.61565e-08
coord 42 = -2.11093e-08
coord 43 = -3.94279e-08
coord 44 = -1.11139e-08
coord 45 = -3.64751e-08
coord 46 = 6.84716e-08
coord 47 = -1.69742e-08
coord 48 = 1.46294e-09
coord 49 = 7.26704e-09
=== rastringin final run: ===
value = 1.8036e-07
coord 0 = -3.83476e-06
coord 1 = -2.06532e-06
coord 2 = 4.92875e-06
coord 3 = -5.32896e-06
coord 4 = -3.29141e-06
coord 5 = -2.8781e-06
coord 6 = 5.45997e-07
coord 7 = -1.49354e-06
coord 8 = -2.96511e-08
coord 9 = -1.8785e-06
coord 10 = 3.36806e-06
coord 11 = -1.17994e-06
coord 12 = 5.19628e-06
coord 13 = -8.29278e-07

```
coord 14 = 9.88509e-07
coord 15 = -6.85608e-07
coord 16 = 5.13126e-07
coord 17 = -2.7182e-07
coord 18 = -2.07421e-06
coord 19 = 3.54345e-07
coord 20 = -2.55601e-05
coord 21 = -1.12109e-07
coord 22 = -8.17373e-07
coord 23 = -2.37354e-06
coord 24 = 9.7132e-06
coord 25 = 4.87308e-07
coord 26 = -3.5547e-08
coord 27 = -1.46141e-07
coord 28 = -3.34959e-06
coord 29 = -4.22812e-07
=== rosenbrock final run: ===
value = 3.86686
coord 0 = 0.998561
coord 1 = 0.997359
coord 2 = 0.99494
coord 3 = 0.997108
coord 4 = 0.998252
coord 5 = 0.998374
coord 6 = 0.998447
coord 7 = 0.998843
coord 8 = 0.998804
coord 9 = 0.997635
coord 10 = 0.997115
coord 11 = 0.998105
coord 12 = 0.998673
coord 13 = 0.99901
coord 14 = 0.999561
coord 15 = 0.999581
coord 16 = 0.999362
coord 17 = 0.99995
coord 18 = 1.0005
coord 19 = 1.00106
coord 20 = 1.00259
coord 21 = 1.00626
coord 22 = 1.01353
coord 23 = 1.02835
coord 24 = 1.05772
coord 25 = 1.11979
coord 26 = 1.25605
```

```
coord 27 = 1.57872
coord 28 = 2.49291
coord 29 = 6.21445
=== simple final run: ===
value = 1.96713e-19
coord 0 = 9.81618e-12
coord 1 = -2.18837e-11
coord 2 = 9.39628e-14
coord 3 = 1.62172e-12
coord 4 = -3.46111e-12
coord 5 = 1.04349e-11
coord 6 = -2.10464e-11
coord 7 = -2.97402e-14
coord 8 = 1.19968e-10
coord 9 = 7.32756e-11
coord 10 = 6.88737e-11
coord 11 = 4.31045e-12
coord 12 = -1.11972e-13
coord 13 = -3.61128e-11
coord 14 = -5.39355e-11
coord 15 = 1.15393e-11
coord 16 = 1.2629e-11
coord 17 = -3.33134e-13
coord 18 = 3.41889e-11
coord 19 = 8.88033e-12
coord 20 = 1.82982e-12
coord 21 = 2.5999e-12
coord 22 = -3.92927e-10
coord 23 = 3.29949e-12
coord 24 = 2.61603e-11
coord 25 = 1.34904e-12
coord 26 = -5.14255e-12
coord 27 = 2.06061e-11
coord 28 = 2.21003e-12
coord 29 = -5.0432e-12
coord 30 = -3.26997e-11
coord 31 = -2.22382e-12
coord 32 = 4.18621e-12
coord 33 = 4.23847e-13
coord 34 = -3.07193e-11
coord 35 = 2.0862e-11
coord 36 = 2.49689e-11
coord 37 = 3.20243e-12
coord 38 = -6.80353e-14
coord 39 = -2.82109e-11
```

```
coord 40 = -7.84547e-12
coord 41 = -1.9589e-11
coord 42 = 8.86109e-12
coord 43 = -6.94081e-11
coord 44 = -2.86398e-12
coord 45 = -2.00984e-12
coord 46 = 8.08079e-12
coord 47 = 1.56918e-12
coord 48 = 1.85047e-11
coord 49 = -7.0779e-12
=== ackley final run: ===
value = 2.17604e-14
coord 0 = -4.39653e-15
coord 1 = -5.30514e-15
coord 2 = -6.97095e-15
coord 3 = -4.10238e-15
coord 4 = 1.3052e-15
coord 5 = -1.06006e-14
coord 6 = 9.13598e-15
coord 7 = -1.52001e-16
coord 8 = 8.91772e-15
coord 9 = 2.42424e-15
coord 10 = 7.29722e-15
coord 11 = 9.01609e-16
coord 12 = -6.70299e-15
coord 13 = 1.84372e-15
coord 14 = 5.68265e-15
coord 15 = -2.43554e-15
coord 16 = 7.95683e-15
coord 17 = -5.05383e-15
coord 18 = -2.79508e-15
coord 19 = 6.00163e-15
PS D:\>
```

Для всіх функцій крім функції Розенброка було досягнуто глобального мінімуму. Для функції Розенброка метод зміг наблизитись до глобального мінімуму (хоча значення останніх 5 координат суттєво відрізняються від «справжнього» значення). Той факт, що метод не знайшов глобальний мінімум для функції Розенброка може бути пов’язаний із недостатньо рівномірним початковим покриттям області роєм.

Аналіз залежності наближення до глобального мінімуму від кількості часток

Оберемо всі параметри крім particle_count такими як у попередньому дослідженні:

local_weight: 4; global_weight: 2; iteration_count: 20000; initial_w: 1.2; comp_interval: 100; e: 0.01

Значення particle_count набуватиме значень: [100, 200, 500, 1000, 1500, 2000, 4000, 8000].

Результати (файл particle-count-research.txt):

particle_count	griewank	rastrigin	rosenbrock	simple	ackley
100	0.239569	1.6261	6381.81	0.101886	0.00170992
200	0.0758497	0.0821559	346.543	0.00654462	0.000110415
500	0.0197664	0.00148483	27.9862	4.83732e-07	1.44707e-09
1000	3.33154e-10	0.0632345	6515.88	1.24399e-11	9.88098e-13
1500	9.34800e-14	7.95808e-13	26.8846	2.45843e-14	2.17604e-14
2000	0.0270664	1.8036e-07	3.86686	1.96713e-19	2.17604e-14
4000	0	1.13687e-13	6503.34	7.9184e-33	7.54952e-15
8000	0.0147798	1.13687e-13	6352.69	7.73889e-53	1.46549e-14

Для простих функцій (sphere (simple), ackley, rastrigin) збільшення кількості часток само по собі покращує наближення до глобального мінімуму.

Для функції Розенброка бачимо, що зміна розміру рою не завжди покращує результат. Для оптимальної поведінки рою потрібно корегувати інші параметри відповідно розміру рою.

Griewank реагує подібно до функції Розенброка (значення 0 при particle_count = 0 скоріше є випадковістю) – починаючи з particle_count = 2000 спостерігаємо погіршення результату.