Paper presentation 10
TaejoonYang 20190379

A comparison of Parallel and Sequential Niching Methods

Introduction

Niching Methods promote the formation and maintenance of stable subpopulations in GA

Examine four niching methods and compare performance on classification/multimodal function optimization

-Parallel niching methods : Sharing, crowding

-Sequential niching methods

-Parallel hill-climber

Parallel Niching-Sharing

Sharing derates each population element's fitness bys amount related to the number of similar individuals in the population

$$f'(i) = \frac{f(i)}{\sum_{j=1}^{n} sh(d(i,j))}$$

$$sh(d) = \begin{cases} 1 - \left(\frac{d}{\sigma_{share}}\right)^{\alpha}, & \text{if } d < \sigma_{share} ; \\ 0 , & \text{otherwise} , \end{cases}$$

Shared fitness, niche count, sharing function, threshold: if distance between two population elements is greater than threshold, they do not affect each others shared fitness

Parallel Niching-Crowding

Insert new elements into the population by replacing similar elements

Deterministic crowding(DC)

Deterministic Crowding

(REPEAT for g generations)

DO $n/2$ times:

1. Select 2 parents, p_1 and p_2, randomly, no replacement
2. Cross them, yielding c_1 and c_2
3. Apply mutation / other operators, yielding c_1' and c_2'
4. IF $[d(p_1, c_1') + d(p_2, c_2')] \leq [d(p_1, c_2') + d(p_2, c_1')]$
   - IF $f(c_1') > f(p_1)$ replace p_1 with c_1'
   - IF $f(c_2') > f(p_2)$ replace p_2 with c_2'

   ELSE

   - IF $f(c_2') > f(p_1)$ replace p_1 with c_2'
   - IF $f(c_1') > f(p_2)$ replace p_2 with c_1'

Parallel hillclimbing

Starts with random generated initial population, forces each element to converge to its nearest attractor

Similar with binary search

Parallel Hillclimbing (Phenotypic)

1. Initialize *Step Size*
2. WHILE *Step Size* $\geq \epsilon$
   (a) FOR each population element
   - Randomly pick a starting variable
   - *Change* = TRUE
   - WHILE *Change*
     – *Change* = FALSE
     – FOR each variable
       * IF adding *Step Size* to current variable yields improved fitness
         · Perform the addition
         · *Change* = TRUE
       * ELSE IF subtracting *Step Size* from current variable yields improved fitness
         · Perform the subtraction
         · *Change* = TRUE

   (b) *Step Size* = *Step Size* / 2

Sequential niching(SN)

Simple GA, Maintaining the best solution of each run offline

Call multiple runs that sequential niching performs to solve a single problem-sequence

Niche radius : to avoid converging to the same area of the search space multiple times

- depress the fitness landscape at all points in radius of solution
- similar with threshold in sharing method

Parallel vs Sequential

Advantage of SN

-simplicity

-ability to work with smaller population

-speed

Disadvantage of SN

-Loss through deration of optimal solution/building blocks

-Repeated search of depressed regions

-Repeated convergence to the same solution

Test problems

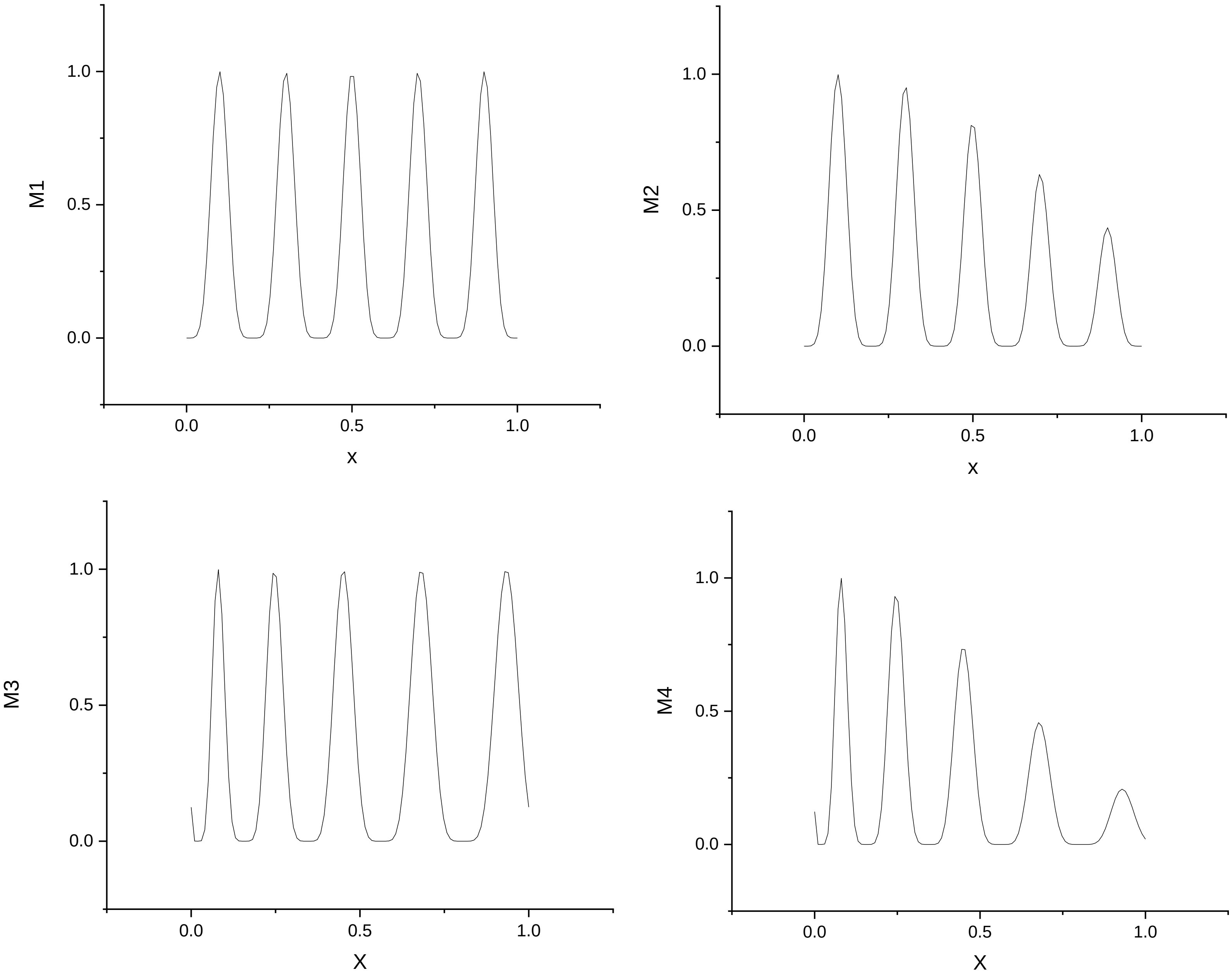
M1-M9 : optimization problems

MUX-6, PAR-8 : classification problems

M1~M4: one dimensional, five-peaked sinusoidal functions

Equally-spaced peaks/not equally spaced peaks

Peaks with uniform height/not uniform height


M1
1.0
0.5
0.0
0.0
0.5
1.0
x
M2
1.0
0.5
0.0
0.0
0.5
1.0
x
M3
1.0
0.5
0.0
0.0
0.5
1.0
X
M4
1.0
0.5
0.0
0.0
0.5
1.0
X

Test problems: M5,M6

M5: two dimensional functions with four peaks of identical height

M6: two dimensional functions with 25 peaks of differing heights

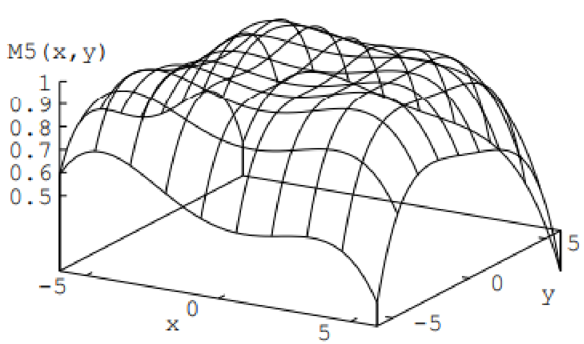


Figure 2: Test Function *M5* is displayed.

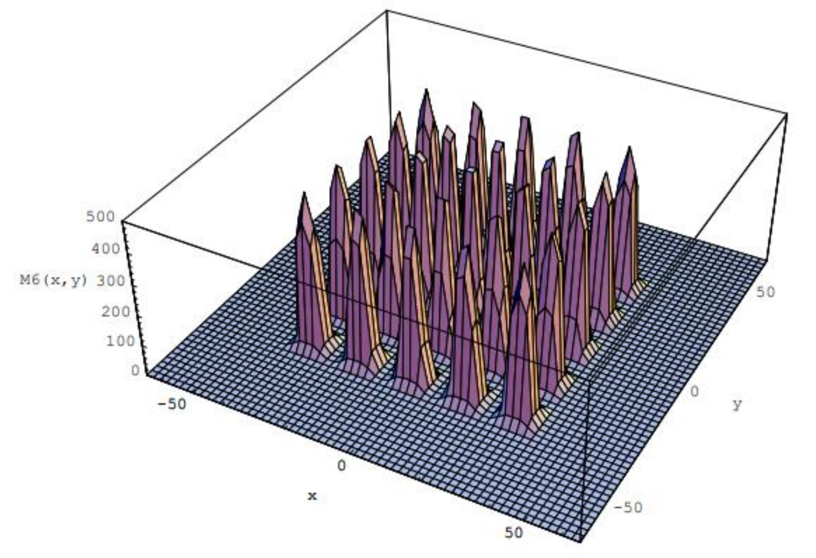


Figure 3: Test Function *M6* is displayed.

Test problems: M7, M8, M9

Massively multimodal, deceptive function

Hardest test problems

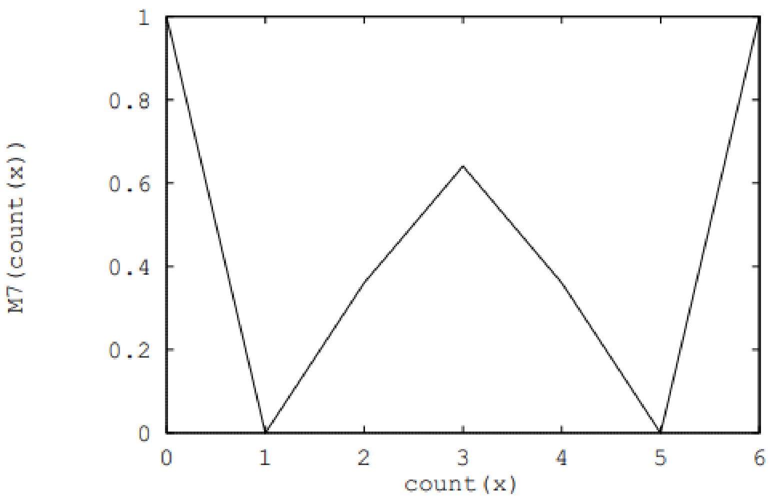


Figure 4: Test Function *M7* is displayed.

Test problems: MUX-6,PAR-8

Classification problem

$$f(POS, NEG) = \begin{cases} 1 + POS & \text{if } NEG = 0 \;; \\ 1 - \frac{NEG}{NTX} & \text{otherwise} \;. \end{cases}$$

MUX-6 : six bit multiplexer problem

PAR-8: 8 bit parity problem

Easiest problem : MUX-6

Intermediate difficulty PAR-8

Result

Easiest problem : M1~M5, MUX-6

Intermediate difficulty : M6, PAR-8

Hardest problem : M7~M9,

Compare the number of GA function(except HC)

Compare total number of function evaluations

Result-Case 1

Most of the case-HC has best performance (m1,2,5,mux-6)
For some case, DC has best performance (m3,4)
On easy problem, HC works well

Method	$\bar{n}$	$\bar{g}$	*GA*: μ	*Combo*: μ
		M1		
HC	2.72			**1017**
SN	3.68	46.40	738	4112
SH	5.76	8.00	**264**	2431
DC	2.40	28.00	380	1246
		M2		
HC	2.72			**1021**
SN	4.64	75.60	1770	8632
SH	8.96	8.70	442	3827
DC	2.40	27.40	**372**	1264
		M3		
HC	3.04			1150
SN	5.92	26.80	719	4375
SH	6.08	8.70	294	2579
DC	2.08	20.30	**262**	**1013**
		M4		
HC	3.04			1140
SN	5.12	72.40	2445	10231
SH	6.72	9.20	352	2892
DC	2.08	17.00	**210**	**975**
		M5		
HC	2.50			**901**
SN	1.30	32.30	180	1456
SH	2.80	8.00	**103**	1111
DC	5.60	25.60	603	2459
		MUX-6		
HC	10.40			**1257**
SN	6.40	140.70	4423	9439
SH	13.60	8.90	**534**	1931
DC	12.00	44.30	2816	3654

Result-Case 1

SN performs poorly in most case

-SN has squashed several peaks in the fitness landscape

-once population grows large enough to locate one peak, it already locate multiple peaks

Method	$\bar{n}$	$\bar{g}$	*GA:* μ	*Combo:* μ
		M1		
HC	2.72			**1017**
SN	3.68	46.40	738	4112
SH	5.76	8.00	**264**	2431
DC	2.40	28.00	380	1246
		M2		
HC	2.72			**1021**
SN	4.64	75.60	1770	8632
SH	8.96	8.70	442	3827
DC	2.40	27.40	**372**	1264
		M3		
HC	3.04			1150
SN	5.92	26.80	719	4375
SH	6.08	8.70	294	2579
DC	2.08	20.30	**262**	**1013**
		M4		
HC	3.04			1140
SN	5.12	72.40	2445	10231
SH	6.72	9.20	352	2892
DC	2.08	17.00	**210**	**975**
		M5		
HC	2.50			**901**
SN	1.30	32.30	180	1456
SH	2.80	8.00	**103**	1111
DC	5.60	25.60	603	2459
		MUX-6		
HC	10.40			**1257**
SN	6.40	140.70	4423	9439
SH	13.60	8.90	**534**	1931
DC	12.00	44.30	2816	3654

Result-Case 2

Sharing has best performance on both case

DC is hard to find the optimal answer on M6
-DC uses non global optima as stepping-stone, dominated by other local optima

Table 2: Performances are given on the two functions of intermediate difficulty.

Method	$\bar{n}$	$\bar{g}$	*GA:* μ	*Combo:* μ
		M6		
HC	12.29			29,017
SN	3.58	146.30	12,202	46,657
SH	5.12	11.80	**1,638**	**12,910**
DC			$> 1.5 \times 10^6$	
		PAR-8		
HC	48.08			202,387
SN	19.20	36.40	100,557	263,666
SH	9.60	12.60	**17,203**	**54,402**
DC	11.20	87.40	125,850	149,022

Result-Case 3

DC only found answer on most case

M8(scaling of M7), sharing has best performance

-sharing is unable to solve unscaled, massively multimodal, deceptive problem

Table 3: Performances are given on the three func
tions of greatest difficulty. Function evaluations are i
thousands (indicated by the letter K).

Method	$\bar{n}$	$\bar{g}$	*GA:* μ	*Combo:* μ
		M7		
HC				> 2000K
SN			> 1500K	
SH			> 1500K	
DC	20.80	119.80	**81K**	**101K**
		M8		
HC				> 2000K
SN			> 1500K	
SH	19.20	19.20	**13K**	**38K**
DC	22.40	134.40	98K	119K
		M9		
HC				> 2000K
SN			> 1500K	
SH				> 2000K
DC	136.53	337.80	**1253K**	**1342K**

Discussion

-HC is best form easiest problems, but hard to solve high difficulty problems

-SN is weak on easy problems, and unable to solve harder problems. In most case HC works better because it does not destroy fitness landscape

-Sharing works on all levels of complexity, but doesn’t work when has extraneous peaks that are similar in fitness to the desired peaks(use scaling)

-DC is generally good for all levels, but it can lose lower optima