Inserting Active Components of Particle Swarm Optimization in Cellular Genetic Algorithms

Enrique Alba
eat@lcc.uma.es
http://www.lcc.uma.es/~eat
Universidad de Málaga
&
Andrea Villagra
avillagra@uaco.unpa.edu.ar
Universidad Nacional de la Patagonia Austral

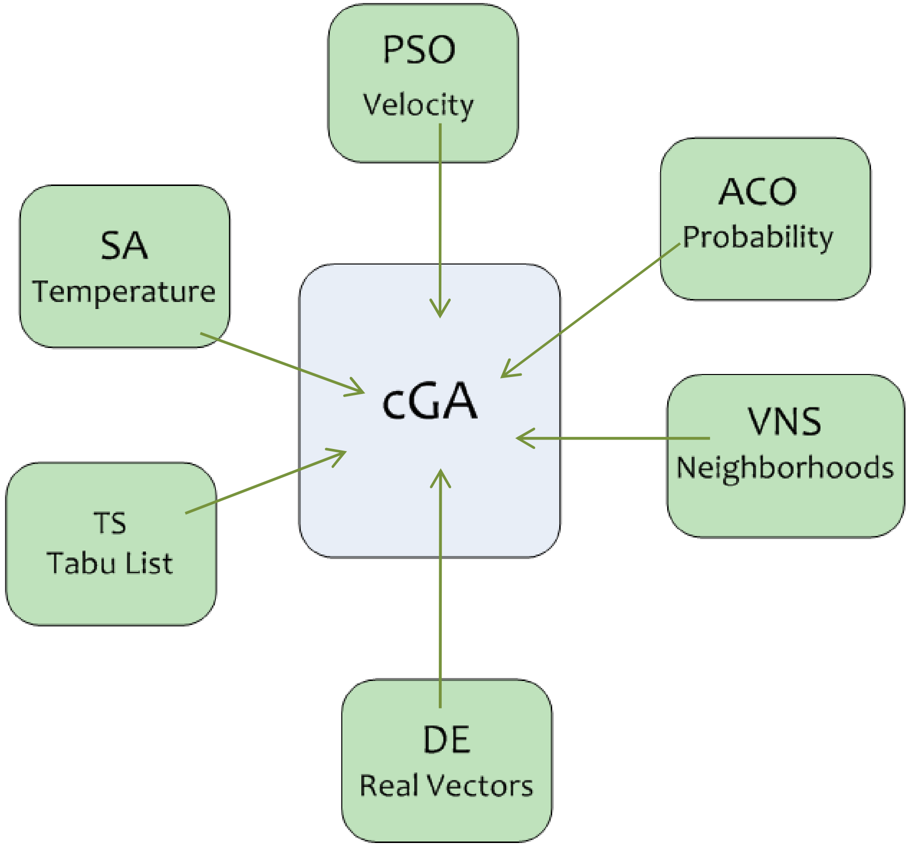
Table of Contents

Motivation

- cGA intrinsic behavior yields a good start efficacy but the basic algorithm needs customization and improvement
- New cGA models should focus in improving both efficiency and efficacy
- There are many methods to do so: selection operator, local search, parallelism, neighborhood definition, population shape, …
- Hybridization between algorithms is always an important research field, but: can we do it in a structured and innovative way ???
- Combinations of algorithms have provided very powerful search algorithms, but: are these algorithms the actual driving forces or there exist some active components in them that make the difference?

Active Components for Hybridization

- Goal: to generate new functional and efficient hybrid algorithms
- Base technique: the cGA Hybrid: active principles of other techniques



Concepts of PSO in cGA

- Personal and social information is maintained
- A mutation operator based in PSO is used inside the cGA
- Two hybrid algorithms:
  - hyCP-local: based on local PSO → information from the individuals' neighborhood
  - hyCP-global: based on global PSO → information from the global best

Concepts of PSO into a cGA

- hybrid algorithms:
  - hyCP-local

Information from the local neighborhood (NEWS)

$$v_{id}(t) = v_{id}(t-1) + \varphi_1(p_{id} - x_{id}(t-1)) + \varphi_2(p_{gd} - x_{id}(t-1))$$

$$x_{id}(t) = x_{id}(t-1) + v_{id}(t)$$

  - hyCP-global

Information from the global best

$$v_{id}(t) = v_{id}(t-1) + \varphi_1(p_{id} - x_{id}(t-1)) + \varphi_2(p_{gd} - x_{id}(t-1))$$

$$x_{id}(t) = x_{id}(t-1) + v_{id}(t)$$

Details on the Algorithms

Algorithm 1 Pseudocode of a cGA

```
Steps-Up(cga) // Algorithm parameters in 'cga'
for s ⟵ 1 to MAX_STEPS do
    for x ⟵ 1 to WIDTH do
        for y ⟵ 1 to HEIGHT do
            nList ⟵ ComputeNeigh (cga,position(x,y));
            parent1 ⟵ IndividualAt(cga,position(x,y));
            parent2 ⟵ LocalSelect(nList);
            DPX1(cga.Pc,nList[parent1],nList[parent2],auxInd.chrom); // Recombination
            BitFlip(cga.Pm,auxInd.chrom); // Mutation
            auxInd.fit ⟵ cga.Fit(Decode(auxInd.chrom));
            InsertNewInd(position(x,y),auxInd,[ifBetter | always],cga, auxPop);
        end for
    end for
    cga.pop ⟵ auxPop;
    UpdateStatistics(cga)
end for
```

Details on the Algorithms

Algorithm 1 Pseudocode of hyCP-local

```
Steps-Up(cga) // Algorithm parameters in 'cga'
for s ⟵ 1 to MAX_STEPS do
   for x ⟵ 1 to WIDTH do
      for y ⟵ 1 to HEIGHT do
         nList ⟵ ComputeNeigh (cga,position(x,y));
         parent1 ⟵ IndividualAt(cga,position(x,y));
         parent2 ⟵ LocalSelect(nList);
         DPX1(cga.Pc,nList[parent1],nList[parent2],auxInd.chrom); // Recombination
         (cga.Pm,auxInd.chrom); // Mutation  MutLPSO (Pm,auxInd.chrom, velocity);
         auxInd.fit ⟵ cga.Fit(Decode(auxInd.chrom));
         InsertNewInd(position(x,y),auxInd,[ifBetter | always],cga, auxPop);
      end for
   end for
   cga.pop ⟵ auxPop;
   UpdateStatistics(cga)
end for
```

Details on the Algorithms

Algorithm 1 Pseudocode of a hyCP-global

```
Steps-Up(cga) // Algorithm parameters in 'cga'
for s ⟵ 1 to MAX_STEPS do
    for x ⟵ 1 to WIDTH do
        for y ⟵ 1 to HEIGHT do
            nList ⟵ ComputeNeigh (cga,position(x,y));
            parent1 ⟵ IndividualAt(cga,position(x,y));
            parent2 ⟵ LocalSelect(nList);
            DPX1(cga.Pc,nList[parent1],nList[parent2],auxInd.chrom); // Recombination
            (cga.Pm,auxInd.chrom); // Mutation MutGPSO (Pm,auxInd.chrom, velocity);
            auxInd.fit ⟵ cga.Fit(Decode(auxInd.chrom));
            InsertNewInd(position(x,y),auxInd,[ifBetter | always],cga, auxPop);
        end for
    end for
    cga.pop ⟵ auxPop;
    UpdateStatistics(cga)
end for
```

Problems and parameters

- Representative set with epistasis, multimodality, and deception
- Parameterization used in our algorithms:

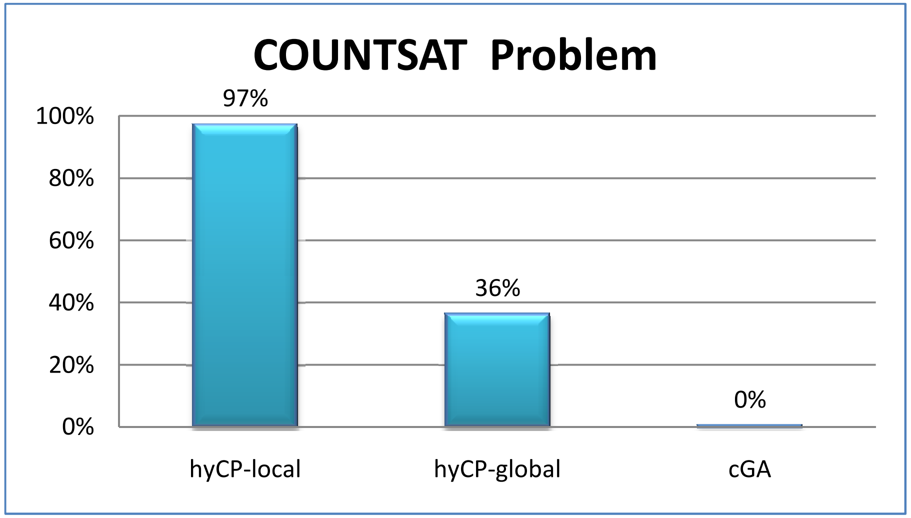
Parameter	Value
Population Size	400 individuals
Selection of Parents	self + CS
Recombination	DPX1, Pc = 1.0
Bit Mutation	(Bit-fip, or mutLPSO or mutGPSO), Pm = 1/L
Replacement	Replace if equal or better

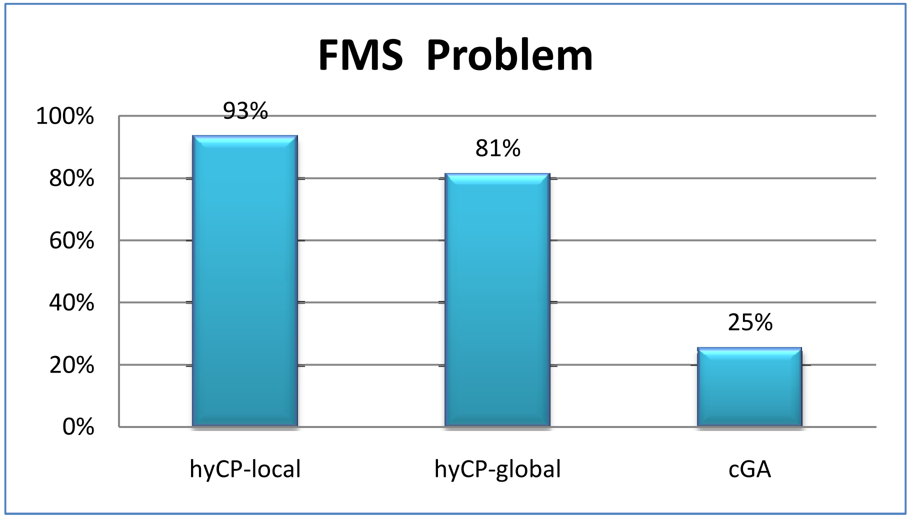
Results: Hit Percentage

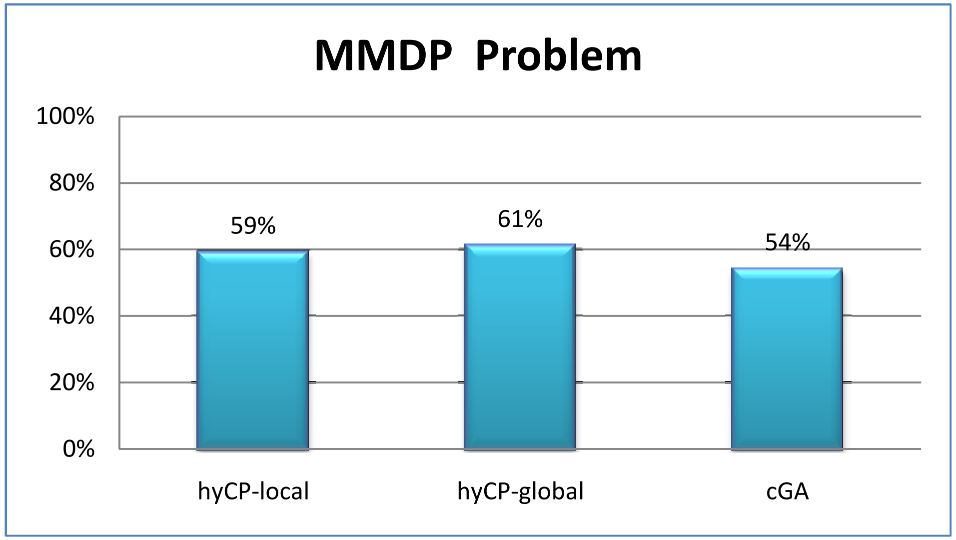
% Success			
Problem	**hyCP-local**	**hyCP-global**	**cGA**
ECC	100	100	100
P-PEAKS	100	100	100
MAXCUT	100	100	100
MMDP	59	61	54
FMS	93	81	25
COUNTSAT	97	36	0

- The success rate for hyCP-local is higher (or at least equal in a few cases) than for the other algorithms

Success Percentage Per Problem







Results: Computational Effort

Evaluations			
Problem	**hyCP-local**	**hyCP-global**	**cGA**
ECC	153 490	157 048	152 662
P-PEAKS	37 655	37 917	39 214
MAXCUT	7 890	6 966	8 303
MMDP	200 800	211 200	144 000
FMS	485 680	424 987	580 080
COUNTSAT	224 800	577 200	1 000 000

- Our hybrid algorithms reduce the number of evaluations required to reach the optimum

Results: Time

Time (ms)			
Problem	**hyCP-local**	**hyCP-global**	**cGA**
ECC	4 116	4 223	2 569
P-PEAKS	3 359	3 345	3 285
MAXCUT	51	50	48
MMDP	6 176	6 457	2 295
FMS	29 497	25 920	26 287
COUNTSAT	1 491	3 468	2 342

- cGA still requires less time to reach the optimum: damn it!

Conclusions and Further Work

- In this work we intend to generate new functional an efficient hybrid algorithms in a structured way
- Indirectly, we try to define what are the actual active components in several metaheuristics
- We incorporate a mutation based on PSO: hyCP-local and hyCP-global
- In all analyzed problems our hybrids obtained equal or better results than the obtained without them (except in real time)
- These results encourage us to expand the set of problems discussed in future work and to incorporate other active components from other metaheuristics: temperature of SA and probability from ACO

Questions and Comments

Málaga

http://www.lcc.uma.es/~eat