# Genetic Algorithms

By: Daniel Otero Gómez

# Main Idea

## Generate Solutions:

Generate matrix of Random Keys and use them to generate factible solutions

## Mutation:

Two steps of a neighborhood search are performed

## Crossover:

1

Random point is used to merge the parent's binary arrays, first parent subset dominate, so second parents subsets are added until the solution is factible or until every subset is added.

If it is not factible then three steps of a neighborhood search are performed over the solution, if it is, two steps are performed

# Crossover:

2

Identify the subsets that the both solutions share and don't, then one child will have every subset that the solutions share and the other will have every subset that they don't.

If it is not factible then three steps of a neighborhood search are performed over the solution, if it is, two steps are performed

3

Orderwise, iterate over the subset arrays and alternate the assignment of the solutions to the childs until a factible solution is reached or every subset is assigned.

If it is not factible then three steps of a neighborhood search are performed over the solution, if it is, two steps are performed

4

Identify the subsets that both solutions have and add them orderwise to the first solution until it is factible, after that the remaining subsets are assigned to the second child until it is factible or there are no more subsets.

```
procedure Genetic_Algorithms()
    for i = 1 to population_size
        P[i] = generate_solution();
    end;
    for i = 1 to generations
        for j = 1 to number_of_children
            (x, y) ← selection(P[]);
            son[j] ← crossover(x, y);
            if random < prob_mutation do
                son[j] ← mutation();
            end;
        end;
        P[] ← update(P[]);
    end;
    return better solution in P[];
end Genetic_Algorithms
```

```
procedure VND()
    s ← initial_solution();
    j = 1;
    while j ≤ number_of_neighborhoods
        Find s' ∈ Nj(s)
        if f(s') < f(s) do
                j = 1;
                s ← s';
        else
                j = j + 1;
        end
    end
    return s
end VNS
```

# CURRENT WORK

| | | npop = 20, nchilds = 20, pmut = 0.4 | | npop = 10, nchilds = 10, pmut = 0.05 | | npop = 10, nchilds = 6, pmut = 0.7 | |
|---|---|---|---|---|---|---|---|
| Files | LB | Scores_GA | Gap_GA | Scores_GA | Gap_GA | Scores_GA | Gap_GA |
| scp41.txt | 429 | 476 | 0.1095571096 | 470 | 0.09557109557 | 479 | 0.1165501166 |
| scp42.txt | 512 | 600 | 0.171875 | 600 | 0.171875 | 622 | 0.21484375 |
| scpnrg1.txt | 160 | 251 | 0.56875 | 254 | 0.5875 | 259 | 0.61875 |
| scpnrg2.txt | 143 | 211 | 0.4755244755 | 821 | 4.741258741 | 217 | 0.5174825175 |
| scpnrg3.txt | 149 | 224 | 0.5033557047 | 208 | 0.3959731544 | 225 | 0.5100671141 |
| scpnrg4.txt | 149 | 227 | 0.5234899329 | 265 | 0.7785234899 | 241 | 0.6174496644 |
| scpnrg5.txt | 149 | 231 | 0.5503355705 | 303 | 1.033557047 | 240 | 0.610738255 |
| scpnrh1.txt | 49 | 87 | 0.7755102041 | 83 | 0.693877551 | 88 | 0.7959183673 |
| scpnrh2.txt | 49 | 88 | 0.7959183673 | 85 | 0.7346938776 | 88 | 0.7959183673 |
| scpnrh3.txt | 46 | 86 | 0.8695652174 | 82 | 0.7826086957 | 85 | 0.847826087 |
| scpnrh4.txt | 45 | 82 | 0.8222222222 | 99 | 1.2 | 79 | 0.7555555556 |
| scpnrh5.txt | 43 | 73 | 0.6976744186 | 100 | 1.325581395 | 74 | 0.7209302326 |
| Mean | 160.25 | 219.6666667 | 0.5719815186 | 280.8333333 | 5.586416721 | 224.75 | 0.5935025023 |

# PREVIOUS WORK

| Files | LB | Scores_VND | Gap_VND | Scores_SA | Gap_SA | Scores_LS | Gap_LS |
|---|---|---|---|---|---|---|---|
| scp41 | 429 | 464 | 1.081585082 | 457 | 1.065268065 | 464 | 1.081585082 |
| scp42 | 512 | 566 | 1.10546875 | 573 | 1.119140625 | 599 | 1.169921875 |
| scpnrg1 | 160 | 241 | 1.50625 | 236 | 1.475 | 241 | 1.50625 |
| scpnrg2 | 143 | 199 | 1.391608392 | 202 | 1.412587413 | 198 | 1.384615385 |
| scpnrg3 | 149 | 212 | 1.422818792 | 205 | 1.375838926 | 211 | 1.416107383 |
| scpnrg4 | 149 | 219 | 1.469798658 | 221 | 1.483221477 | 224 | 1.503355705 |
| scpnrg5 | 149 | 221 | 1.483221477 | 218 | 1.463087248 | 219 | 1.469798658 |
| scpnrh1 | 49 | 85 | 1.734693878 | 80 | 1.632653061 | 80 | 1.632653061 |
| scpnrh2 | 49 | 83 | 1.693877551 | 77 | 1.571428571 | 82 | 1.673469388 |
| scpnrh3 | 49 | 79 | 1.612244898 | 81 | 1.653061224 | 80 | 1.632653061 |
| scpnrh4 | 49 | 76 | 1.551020408 | 77 | 1.571428571 | 76 | 1.551020408 |
| scpnrh5 | 49 | 71 | 1.448979592 | 69 | 1.408163265 | 75 | 1.530612245 |
| Mean | 161.333333 | 209.6666667 | 1.458463956 | 208 | 1.435906537 | 212.4166667 | 1.462670187 |

# PREVIOUS WORK

| File | LB | Scores_Constructive | Gap_Constructive | Scores_GRASP | Gap_GRASP | Scores_Noise | Gap_Noise |
|---|---|---|---|---|---|---|---|
| scp41 | 429 | 567 | 1.321678322 | 587 | 1.368298368 | 601 | 1.400932401 |
| scp42 | 512 | 800 | 1.5625 | 773 | 1.509765625 | 818 | 1.59765625 |
| scpnrg1 | 160 | 498 | 3.1125 | 494 | 3.0875 | 450 | 2.8125 |
| scpnrg2 | 143 | 480 | 3.356643357 | 449 | 3.13986014 | 390 | 2.727272727 |
| scpnrg3 | 149 | 467 | 3.134228188 | 471 | 3.161073826 | 410 | 2.751677852 |
| scpnrg4 | 149 | 488 | 3.275167785 | 493 | 3.308724832 | 497 | 3.33557047 |
| scpnrg5 | 149 | 476 | 3.194630872 | 471 | 3.161073826 | 412 | 2.765100671 |
| scpnrh1 | 49 | 467 | 9.530612245 | 483 | 9.857142857 | 389 | 7.93877551 |
| scpnrh2 | 49 | 473 | 9.653061224 | 487 | 9.93877551 | 394 | 8.040816327 |
| scpnrh3 | 49 | 436 | 8.897959184 | 454 | 9.265306122 | 365 | 7.448979592 |
| scpnrh4 | 49 | 434 | 8.857142857 | 448 | 9.142857143 | 360 | 7.346938776 |
| scpnrh5 | 49 | 430 | 8.775510204 | 446 | 9.102040816 | 365 | 7.448979592 |
| Means | 161.3 | 501.3333333 | 5.389302853 | 504.6666667 | 5.503534922 | 454.25 | 4.634600014 |

TIMES

| | npop = 20, nchilds = 20, pmut = 0.4 | npop = 10, nchilds = 10, pmut = 0.05 | npop = 10, nchilds = 6, pmut = 0.7 |
|---|---|---|---|
| Files | Time_GA | Time_GA | Time_GA |
| scp41.txt | 378.0693 | 331.146 | 330.61194 |
| scp42.txt | 409.9087 | 331.88327 | 346.32315 |
| scpnrg1.txt | 2975.2793 | 951.05286 | 809.6917 |
| scpnrg2.txt | 3639.6802 | 666.97766 | 1211.1327 |
| scpnrg3.txt | 4571.991 | 984.34674 | 1306.9711 |
| scpnrg4.txt | 2781.265 | 560.327 | 1039.7277 |
| scpnrg5.txt | 2016.3799 | 452.91058 | 1302.148 |
| scpnrh1.txt | 580.8112 | 402.57492 | 617.5966 |
| scpnrh2.txt | 756.74255 | 830.96027 | 316.99796 |
| scpnrh3.txt | 1452.4935 | 375.24002 | 317.8172 |
| scpnrh4.txt | 1051.9934 | 311.59384 | 502.24942 |
| scpnrh5.txt | 928.94934 | 349.7842 | 443.12744 |
| Mean | 1795.296949 | 545.7330933 | 712.032959 |

# TIMES

| Files | Time_VND | Time_SA | Time_LS | Time_Constructive | Time_GRASP | Time_Noise |
|---|---|---|---|---|---|---|
| scp41 | 66.38048 | 265.9543 | 194.25882 | 0.09774614 | 10.00249883 | 0.130107894 |
| scp42 | 81.9109 | 318.23587 | 185.85767 | 0.117312346 | 9.614420858 | 0.135845263 |
| scpnrg1 | 329.86316 | 329.1465 | 313.46548 | 0.29492503 | 16.70139495 | 0.374160904 |
| scpnrg2 | 325.64615 | 328.35385 | 317.22665 | 0.333223802 | 17.21558182 | 0.303016868 |
| scpnrg3 | 305.04962 | 306.8926 | 352.7319 | 0.319533285 | 17.26591519 | 0.336570761 |
| scpnrg4 | 302.08017 | 308.1545 | 373.29153 | 0.338819411 | 17.27205743 | 0.335159393 |
| scpnrg5 | 302.5499 | 313.1078 | 347.537 | 0.330781252 | 17.76609335 | 0.326855064 |
| scpnrh1 | 287.9095 | 330.36304 | 343.93628 | 0.219647225 | 12.99270469 | 0.222027946 |
| scpnrh2 | 302.747 | 317.33344 | 346.71912 | 0.217735787 | 12.93251374 | 0.219542565 |
| scpnrh3 | 303.5528 | 348.88947 | 348.1915 | 0.231129454 | 13.14968906 | 0.223854835 |
| scpnrh4 | 318.37418 | 322.15634 | 337.86264 | 0.216003276 | 13.08049336 | 0.223210306 |
| scpnrh5 | 320.5488 | 330.9373 | 327.074 | 0.220235046 | 13.29253811 | 0.217360406 |
| Mean | 270.5510559 | 318.2937317 | 315.6793518 | 0.2447576712 | 14.27382512 | 0.2539760171 |

# CONCLUSIONS

- Generating a large amount of childs while having a high mutation probability forces the model to take too long generating solutions and performing the neighborhood search.
- There is a trade-off between generating factible solutions in the crossover and working with non factible solutions within the neighborhood search.
- Variable Neighborhood structure of the genetic algorithm helps the algorithm to escape local minima.
- Introducing random solutions when the algorithm get stuck in local minima has a positive impact in the solution generation.
- The algorithm reaches decent local minimum faster that every other algorithm implemented, however, it is unable of going further, thus, other methods outperform it.
- Working with multiple solutions require that operations are not computationally heavy, being more necessary when there is not an operation parallelization.