Algorithmic Methods for Mathematical Models
Lab 5 — GRASP and BRKGA Metaheuristics

Marcel Cases Freixenet

May 14, 2021

Tasks and results

a) GRASP

Prepare a pseudocode for the GRASP constructive algorithm. Specify the greedy function and the RCL.

Algorithm

Initialize candidate set $\mathcal{C}$, index c
$\omega \leftarrow \{\}$
while ω is not a solution
  evaluate $q(c, \omega) \quad \forall c \in \mathcal{C}$
  $q_{min} \leftarrow \min\{q(c) | c \in \mathcal{C}\}$
  $q_{max} \leftarrow \max\{q(c) | c \in \mathcal{C}\}$
  $RCL_{max} \leftarrow \left\{c \in \mathcal{C} | q(c) \geq q^{max} - \alpha(q^{max} - q^{min})\right\}$
  Select $c \in$ RCL at random
  $\omega \leftarrow \omega U \{c_{best}\}$
  update $\mathcal{C}$
return $\langle f(\omega), \omega \rangle$

RCL

$$RCL_{max} \leftarrow \left\{c \in \mathcal{C} | q(c) \geq q^{max} - \alpha(q^{max} - q^{min})\right\}$$

Tune parameter α. Generate at least 2 new random instances of medium size and run the constructive phase of GRASP for different values of α from 0-1 in steps of 0.1. For every value, run the algorithm at least three times and compute the average of the cost of the obtained solutions. Prepare plots with the obtained values. Find the best value of α and use it for the rest of experiments.

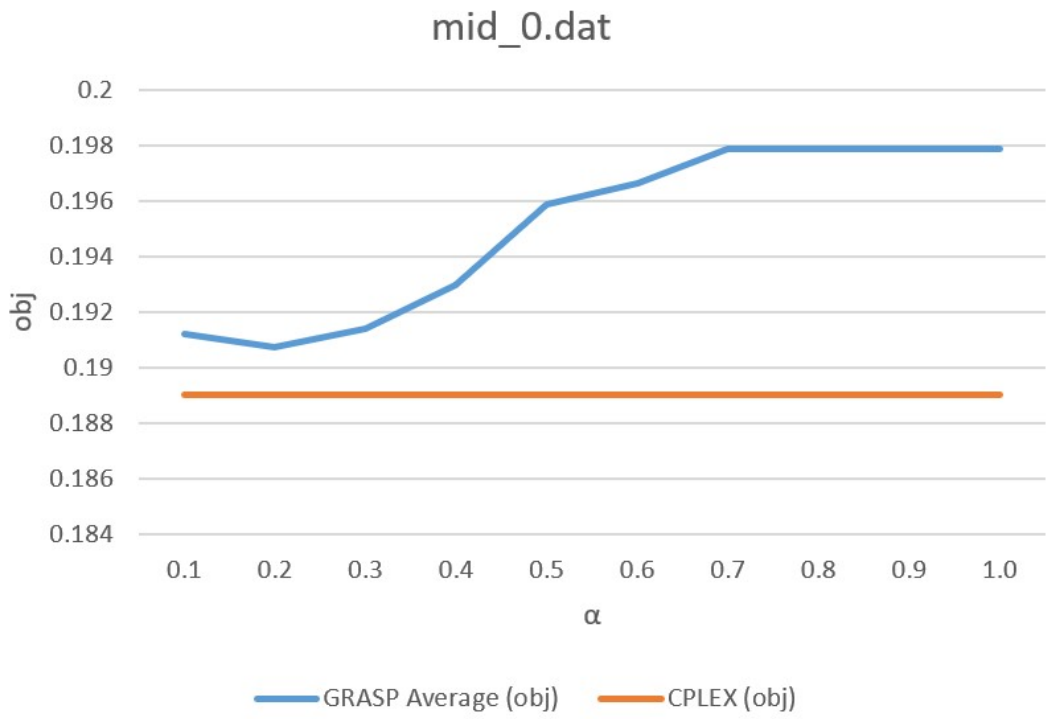
We have generated two new instances (*mid_0.dat* and *mid_1.dat*) with parameters

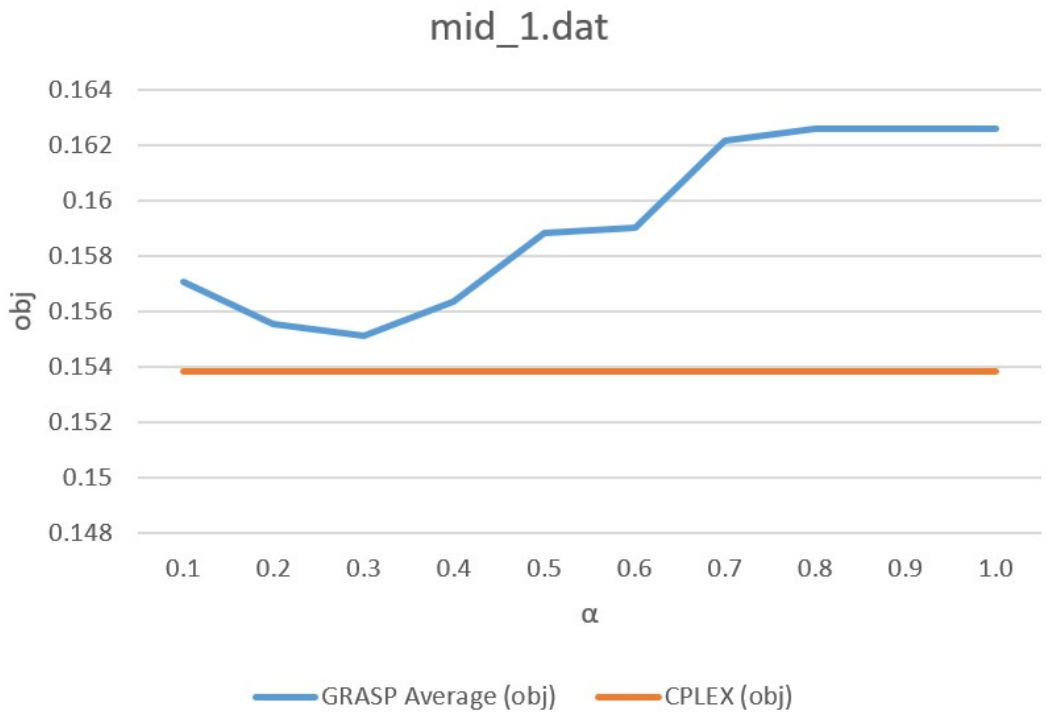
```
numCPUs = 10;
minCapacityPerCPU = 5000;
maxCapacityPerCPU = 10000;

numTasks = 40;
minResourcesPerTask = 100;
maxResourcesPerTask = 500;
```

Then we configure a max run time of 6min. We range α from 0.1 to 1.0, three times for each instance. Then we compute the average of the three values and compare it with the result obtained on CPLEX for the same instance. The overall results are as follows:

dataset	α	Exec1 (obj)	Exec2 (obj)	Exec3 (obj)	GRASP $\overline{Av.}$	CPLEX
mid_0.dat	0.1	0.19073951	0.19073951	0.1922483	0.19124244	0.189041449
	0.2	0.19075063	0.19047316	0.1909681	0.19073063	0.189041449
	0.3	0.1912146	0.19172444	0.19126091	0.191399983	0.189041449
	0.4	0.19299799	0.19217333	0.19380224	0.192991187	0.189041449
	0.5	0.19648592	0.1967134	0.19441534	0.195871553	0.189041449
	0.6	0.19534483	0.1967249	0.19787703	0.19664892	0.189041449
	0.7	0.19787703	0.19787703	0.19787703	0.19787703	0.189041449
	0.8	0.19787703	0.19787703	0.19787703	0.19787703	0.189041449
	0.9	0.19787703	0.19787703	0.19787703	0.19787703	0.189041449
	1.0	0.19787703	0.19787703	0.19787703	0.19787703	0.189041449
mid_1.dat	0.1	0.15716068	0.1563741	0.15773533	0.157090037	0.153853533
	0.2	0.15539569	0.15586493	0.15540536	0.155555327	0.153853533
	0.3	0.15519553	0.15512396	0.1551129	0.15514413	0.153853533
	0.4	0.15691174	0.1555161	0.15668865	0.156372163	0.153853533
	0.5	0.15909265	0.15956	0.1578352	0.158829283	0.153853533
	0.6	0.15949287	0.1582291	0.15936417	0.159028713	0.153853533
	0.7	0.16216716	0.16221654	0.16218204	0.16218858	0.153853533
	0.8	0.16259127	0.16259127	0.16259127	0.16259127	0.153853533
	0.9	0.16259127	0.16259127	0.16259127	0.16259127	0.153853533
	1.0	0.16259127	0.16259127	0.16259127	0.16259127	0.153853533





As we can see on the plots, the closest value for GRASP compared to the solution obtained by the simplex method is when $\alpha \approx 0.25$. This is the value we're gonna use for the following tests.

b) BRKGA

Prepare a pseudocode for the BRKGA decoder algorithm. Specify how the chromosome is used.

Algorithm

$ch \leftarrow$ random keys BRKGA
for each $c \in C$ **do**
  $c.weight = c.weight \times ch\,[c]$
$\omega \leftarrow \emptyset$
forall $t \in T$ **do**
  $c^{min} \leftarrow q(t, \omega)$
  if $c^{min} = \infty$ **then**
    return $infeasible$
  end
  $\omega \leftarrow \omega \bigcup \{\langle t, c^{min} \rangle\}$
end
return ω

.

The chromosome takes random keys form computers, and assigns them by weight to the c set.

Study what is the best combination of BRKGA parameters (size of population, inheritance probability, elite set and mutant percentages).

The computations have been run with parameters:

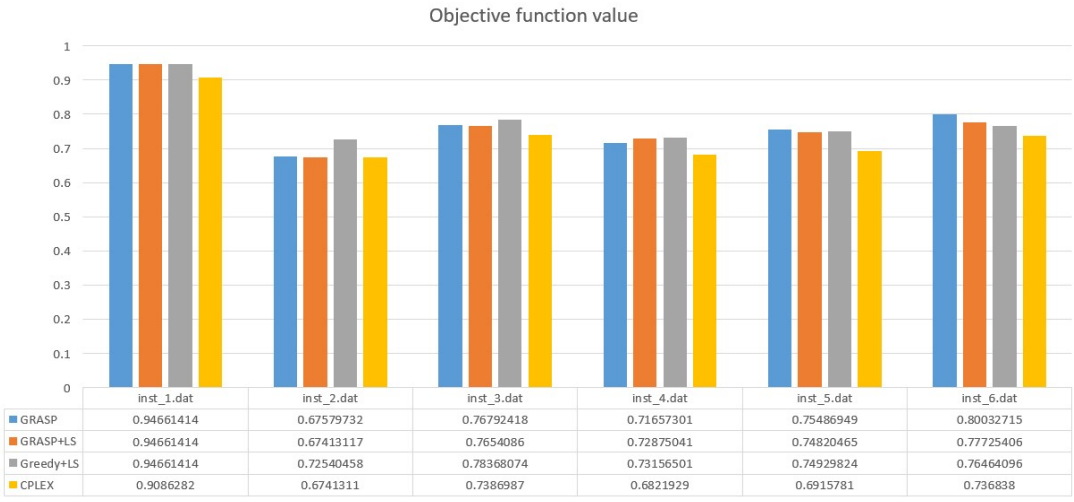
```
eliteProp               = 0.2
mutantProp              = 0.1
inheritanceProb         = 0.7
IndividualsMultiplier   = 1.0
```

c) Solve the same instances that you generated in the last lab session using GRASP, GRASP + LS and BRKGA

dataset	Algorithm	Elap. Time (s)	Obj. Value	Iterations
inst_1.dat	GRASP	600	0.94661414	3261138
	GRASP+LS	600	0.94661414	426675
	Greedy+LS		0.94661414	1
	BRKGA		0.92250252	1825
	CPLEX		0.9086282	1
inst_2.dat	GRASP	600	0.67579732	2109081
	GRASP+LS	600	0.67413117	131426
	Greedy+LS		0.72540458	1
	BRKGA		0.68069166	2512
	CPLEX		0.6741311	1
inst_3.dat	GRASP	600	0.76792418	577404
	GRASP+LS	600	0.7654086	246114
	Greedy+LS		0.78368074	1
	BRKGA		0.74052682	3065
	CPLEX		0.7386987	1
inst_4.dat	GRASP	600	0.71657301	2521897
	GRASP+LS	600	0.72875041	78450
	Greedy+LS		0.73156501	1
	BRKGA		0.68315806	5502
	CPLEX		0.6821929	1
inst_5.dat	GRASP	600	0.75486949	3105566
	GRASP+LS	600	0.74820465	18450
	Greedy+LS		0.74929824	1
	BRKGA		0.70585596	2508
	CPLEX		0.6915781	1
inst_6.dat	GRASP	600	0.80032715	2836117
	GRASP+LS	600	0.77725406	14589
	Greedy+LS		0.76464096	1
	BRKGA		0.73985047	3541
	CPLEX		0.736838	1

d) Compare the results with those obtained in the previous lab session using greedy + LS and CPLEX.

In most of the cases, the more iterations the better the result, compared to the one obtained on CPLEX. In general, GRASP provides better solution than Greedy, and when it is executed together with Local Search, it has an even better approach. BRKGA offers the best result among heuristic methods, but at a higher computational cost.



References

- Class presentations & labs

Annex A: Task Statement

Algorithmic Methods for Mathematical Models (AMMM)

Lab Session 5 – GRASP and BRKGA Metaheuristics

In this fifth session we will focus on using the GRASP and the BRKGA meta-heuristics.

Tasks

In pairs, do the following tasks and prepare a lab report using the Python code that is provided.

a) GRASP:
   - Prepare a pseudocode for the GRASP constructive algorithm. Specify the greedy function and the RCL.
   - Tune parameter α. Generate at least 2 new random instances of medium size and run the constructive phase of GRASP for different values of α from 0-1 in steps of 0.1. For every value, run the algorithm at least three times and compute the average of the cost of the obtained solutions. Prepare plots with the obtained values. Find the best value of α and use it for the rest of experiments.

b) BRKGA
   - Prepare a pseudocode for the BRKGA decoder algorithm. Specify how the chromosome is used.
   - Study what is the best combination of BRKGA parameters (size of population, inheritance probability, elite set and mutant percentages).

c) Solve the same instances that you generated in the last lab session using:
   - GRASP: Constructive phase only.
   - GRASP: Constructive + Local search (do for all combinations)
   - BRKGA

   Configure the heuristics to stop after 10min and plot the quality of the solutions, the number of iterations performed, and time to solve against the size of the instances.

d) Compare the results with those obtained in the previous lab session using greedy + LS and CPLEX.