

# Spring School on Integrated Operational Problems

May 14-16, 2018, Troyes, France

## PLAN

1rst session

Introduction (15-30 minutes)

C++ (1 hour)

The Clarke and Wright algorithm

Modifying the CAW and testing it

Adding your own strategy to the CAW

How to debug ?

2nd session

JSON (30 minutes)

Understanding the role of the JSON files

Adding two new algorithms to the CAW

PHP (30 minutes)

Adding a new problem to the web-site

Conclusion will be yours !



# Spring School on Integrated Operational Problems

May 14-16, 2018, Troyes, France

JSON or  
*"How to customize the web-services ?"*



# JSON customization

A JSON file contains the description of the list of problems the web-site offers.

Welcome to the Routing Problems website !

Problems

**select a problem :**

Travelling Salesman	<input type="button" value="select"/>	<input type="button" value="info"/>
Open Vehicle Routing	<input type="button" value="select"/>	<input type="button" value="info"/>
Clarke and Wright	<input type="button" value="select"/>	<input type="button" value="info"/>

Real instance

Data's instance

Distance matrix

Algorithms

Solutions

```
<dir>\services\problems.json
```

# JSON customization

We cannot (and do not want to) teach you full JSON syntax in 20 minutes.  
We'll give you here a simple procedure to follow « blindly ».

Hopefully, it's not so complex.

But, if you want to acquire more knowledge about JSON  
have a look on Internet, there are thousands of on-line documentations about it.

# JSON customization

Each element of the JSON array in `problems.json` describes one web-service, with the help of several attributes.

Welcome to the Routing Problems website !

### Problems

**select a problem :**

Travelling Salesman	<input type="button" value="select"/>	<input type="button" value="info"/>
Open Vehicle Routing	<input type="button" value="select"/>	<input type="button" value="info"/>
Clarke and Wright	<input type="button" value="select"/>	<input type="button" value="info"/>

Real instance

Data's instance

Distance matrix

Algorithms

Solutions

```
/d/projects/RP/RPWS/services
{
  "name": "TSP",
  "full_name": "Travelling Salesman",
  "problem_location": "services/problems/TSP/",
  "instance_location": "services/problems/TSP/instance.json",
  "algorithms_location": "services/problems/TSP/algorithms.json",
  "description": "wiki:<i>The Travelling Salesman Problem (TSP) asks the following question: Given a list of cities and the distances between each pair of cities, what is the shortest possible route that visits each city exactly once and returns to the origin city? It is an NP-hard problem in combinatorial optimization, important in operations research and theoretical computer science.</i>",
  "instance_txt": "An instance of this problem is <br> - a set of points without other property.<br> - a distance matrix ",
  "solution_txt": "A solution is an unique trip <br><center> <img src=\"services/problems/TSP/Img/Solution_Euc.PNG\" alt=\"solution euclidiane\" height=\"200\"> <img src=\"services/problems/TSP/Img/Solution_Real.PNG\" alt=\"solution euclidiane\" height=\"200\"></center> ",
},
{
  "name": "OVRP",
  "full_name": "Open Vehicle Routing",
  "problem_location": "services/problems/OVRP/",
  "instance_location": "services/problems/OVRP/instance.json",
  "algorithms_location": "services/problems/OVRP/algorithms.json",
  "exemple_location": "services/problems/OVRP/solution.jpg",
  "description": "<i>In the Open Vehicle Routing Problem (OVRP), the objective is to minimize the total route cost to deliver all the client without taking into account the cost of vehicle when it leaved or come back to the depot.</i>"
},
{
  "name": "CAW",
  "full_name": "Clarke and Wright",
  "problem_location": "services/problems/CAW/",
  "instance_location": "services/problems/CAW/instance.json",
  "algorithms_location": "services/problems/CAW/algorithms.json",
  "description": "<i>The classroom traditionnal Clarke and Wright algorithm.</i>",
  "instance_txt": "An instance of this problem is a set of points, a distance matrix, a depot and a capacity."
},
@
25,1-8 25%
```

# JSON customization

Some will be displayed on the screen to help the user.

Welcome to the Routing Problems website !

### Problems

**select a problem :**

Travelling Salesman	<input type="button" value="select"/>	<input type="button" value="info"/>
Open Vehicle Routing	<input type="button" value="select"/>	<input type="button" value="info"/>
Clarke and Wright	<input type="button" value="select"/>	<input type="button" value="info"/>

Real instance

Data's instance

Distance matrix

Algorithms

Solutions

### Description of the problem :

**Problem name:**  **Abbreviation:**

**Description**

*The classroom traditional Clarke and Wright algorithm.*

**Instance**

An instance of this problem is defined by a depot and a capacity.

**Solution**

A solution is a set of trips starting from the depot. The number of points in a single trip is limited by the capacity. The sum of the delivered goods during the trip must be less than or equal to the capacity.

If you click on  
« info »  
You will see  
this.

# JSON customization

Some will be used « internally » to find the algorithms and data model definitions.

```
"problem_location": "services/problems/TSP/",  
"instance_location": "services/problems/TSP/instance.json",  
"algorithms_location": "services/problems/TSP/algorithms.json",
```

If you remember the structure of the files under `<dir>` you will instantaneously understand that the values of

```
problem_location  
instance_location  
and  
algorithms_location
```

are « pointers » (or path references) to the files stored below in the hierarchy.

# JSON customization - problem

Let's create a new problem.

Go at `<dir>\services\problems`  
and duplicate the `CAW` directory into `WAC`

```
/cygdrive/d/projects/RP/RPWS/services/problems
bome1@UBS-19760 /cygdrive/d/projects/RP/RPWS/services
$ ls
CVS  DMZ  problems  problems.json

bome1@UBS-19760 /cygdrive/d/projects/RP/RPWS/services
$ ls
CVS  DMZ  problems  problems.json

bome1@UBS-19760 /cygdrive/d/projects/RP/RPWS/services
$ cd problems

bome1@UBS-19760 /cygdrive/d/projects/RP/RPWS/services/problems
$ ls
CAW  CVS  DARP  hello  hellox  line  OVRP  rev  TSP  txt  VRP

bome1@UBS-19760 /cygdrive/d/projects/RP/RPWS/services/problems
$ cp -r CAW WAC

bome1@UBS-19760 /cygdrive/d/projects/RP/RPWS/services/problems
$ ls
CAW  CVS  DARP  hello  hellox  line  OVRP  rev  TSP  txt  VRP  WAC

bome1@UBS-19760 /cygdrive/d/projects/RP/RPWS/services/problems
$
```



# JSON customization - problem

One step « upstairs »  
add at the end of the `problems.json` file  
a copy of the description of CAW  
and modify it to become WAC

```
/d/projects/RP/RPWS/services
{
  "description": "<i>In the Open Vehicle Routing Problem (OVRP), the objective is to minimize the total route cost to deliver all the client without taking into account the cost of vehicle when it leaved or come back to the depot.</i>",
  "name": "CAW",
  "full_name": "Clarke and Wright",
  "problem_location": "services/problems/CAW/",
  "instance_location": "services/problems/CAW/instance.json",
  "algorithms_location": "services/problems/CAW/algorithms.json",
  "description": "<i>The classroom traditionnal Clarke and Wright algorithm.</i>",
  "instance_txt": "An instance of this problem is a set of points, a distance matrix, a depot and a capacity.",
  "solution_txt": "A solution is a set of trips from the depot to some points and back to the depot. The number of points in a single trip is limited by the vehicle's capacity because the sum of the delivered goods during the trip cannot exceed the vehicle's capacity."
},
{
  "name": "WAC",
  "full_name": "Wright and Clarke",
  "problem_location": "services/problems/WAC/",
  "instance_location": "services/problems/WAC/instance.json",
  "algorithms_location": "services/problems/WAC/algorithms.json",
  "description": "WAC description",
  "instance_txt": "WAC instance",
  "solution_txt": "WAC solution"
}
}
```

Don't forget to add a column here !



# JSON customization - problem

Restart your web-browser.  
It should look like this now.

Welcome to the Routing Problems website !

### Problems

**select a problem :**

Travelling Salesman	<input type="button" value="select"/>	<input type="button" value="info"/>
Open Vehicle Routing	<input type="button" value="select"/>	<input type="button" value="info"/>
Clarke and Wright	<input type="button" value="select"/>	<input type="button" value="info"/>
Wright and Clarke	<input type="button" value="select"/>	<input type="button" value="info"/>

Type of instances

Data's instance

Distance matrix

Algorithms

Solutions

### Description of the problem :

**Problem name:**  **Abbreviation:**

**Description**  
WAC description

**Instance**  
WAC instance

**Solution**  
WAC solution

The WAC  
appears last  
with the  
following help  
text.

# JSON customization - algorithm

Because WAC is a strict copy of CAW, everything will be identical to CAW if you decide to play with the WAC problem. So, no need to test this.

Don't forget to add a column here !

To deeper customize our web-services, instead of adding more problems, we now want to add more algorithms. We want to add a second algorithm to WAC. Modify the WAC\algorithms.json file like this.

```

[
  {
    "name": "WAC1",
    "full_name": "Wright and Clarke 1",
    "description": "Single vehicule, limited capacity, customer delivery planning of goods",
    "time": "<10s",
    "rank": "1",
    "optimality": "no",
    "url": "services/problems/WAC/WAC1.php"
  },
  {
    "name": "WAC2",
    "full_name": "Wright and Clarke 2",
    "description": "Single vehicule, limited capacity, customer delivery planning of goods",
    "time": "<10s",
    "rank": "1",
    "optimality": "no",
    "url": "services/problems/WAC/WAC2.php"
  }
]

```

# JSON customization - algorithm

Rename CAW.php into WAC1.php  
Duplicate WAC1.php into WAC2.php  
Don't modify WAC1.php and WAC2.php,  
for they execute the same CAW algorithm located under program.

```

/cygdrive/d/projects/RP/RPWS/services/problems/WAC
bome1@UBS-19760 /cygdrive/d/projects/RP/RPWS/services/problems/WAC
$ ls
algorithms.json  CAW.php  instance.json  program

bome1@UBS-19760 /cygdrive/d/projects/RP/RPWS/services/problems/WAC
$ ll
total 17
-rwxr-xr-x  1 bome1  None   253  3 avr.  13:41 algorithms.json
-rwxr-xr-x  1 bome1  None  5502  3 avr.  13:41 CAW.php
-rwxr-xr-x  1 bome1  None   675  3 avr.  13:41 instance.json
drwxr-xr-x+ 1 bome1  None    0  3 avr.  13:41 program

bome1@UBS-19760 /cygdrive/d/projects/RP/RPWS/services/problems/WAC
$ mv CAW.php WAC1.php

bome1@UBS-19760 /cygdrive/d/projects/RP/RPWS/services/problems/WAC
$ cp WAC1.php WAC2.php

bome1@UBS-19760 /cygdrive/d/projects/RP/RPWS/services/problems/WAC
$ ls
algorithms.json  instance.json  program  WAC1.php  WAC2.php

bome1@UBS-19760 /cygdrive/d/projects/RP/RPWS/services/problems/WAC
$
```



# JSON customization - algorithm

Restart your web-browser, and select the WAC problem until you reach the algorithms sub-window. You should see this now.

The screenshot shows a web browser window displaying the 'Routing Problems' website. The browser's address bar shows 'localhost/#'. The website has a navigation bar with logos for Lab STICC, UBS, and OMS. Below the navigation bar, there is a search bar and a 'search' button. The main content area features a map of Paris with several numbered markers (1-11) indicating specific locations. The sidebar on the left contains a 'Welcome to the Routing Problems website!' message and a list of navigation options: 'Problem: Wright and Clarke', 'Real instance', 'Data's instance', 'Distance matrix', and 'Algorithms'. The 'Algorithms' section is highlighted and contains a table with the following data:

select an algorithm:						
name	time	rank	exact	exec.	info	
WAC1	<10s	1	no	RUN	?	
WAC2	<10s	1	no	RUN	?	

Below the table, there is a 'Results:' section and a 'Solutions' section. The map on the right shows a detailed view of Paris with various streets and landmarks labeled, and several numbered markers (1-11) indicating specific locations.

# JSON customization - algorithm

If you click on « ? » you will see some helping text displayed.

Algorithm description : ×

Algorithm name:  Abbreviation:

---

**Description**

Single vehicule, limited capacity, customer delivery planning of goods

---

**Parameters**

---

# JSON customization - algorithm

Click on « run » for WAC1 and WAC2 to see the resulting paths.  
Of course, both will display the same paths, because they execute the same code.

The screenshot shows a web browser window displaying the 'Routing Problems' website. The page features a map of Paris with a red path marked with numbered points (1-11). The sidebar on the left contains the following information:

**Problem: Wright and Clarke**

**Real instance**

**Data's instance**

**Distance matrix**

**Algorithms**

select an algorithm:

name	time	rank	exact	exec.	info
WAC1	<10s	1	no	<input type="button" value="RUN"/>	<input type="button" value="?"/>
WAC2	<10s	1	no	<input type="button" value="RUN"/>	<input type="button" value="?"/>

**Results:**

WAC1	0:00:03	running...
WAC1	0:00:00	
WAC2	0:00:00	

**Solution from WAC1 (2)**

trips: ALL	250	0	0	0.7	<input type="button" value="display"/>
------------	-----	---	---	-----	--

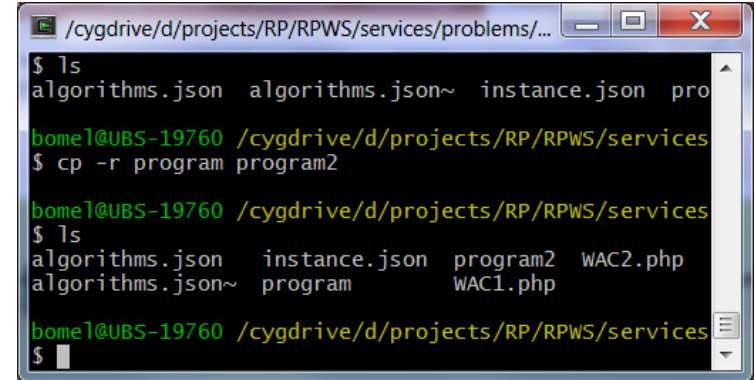
**Properties:**

runPos	1
cout	16309.3
duree	0
success	true
verif	0

# JSON customization - algorithm

We want to create a different executable for WAC2

Duplicate WAC\program into program2

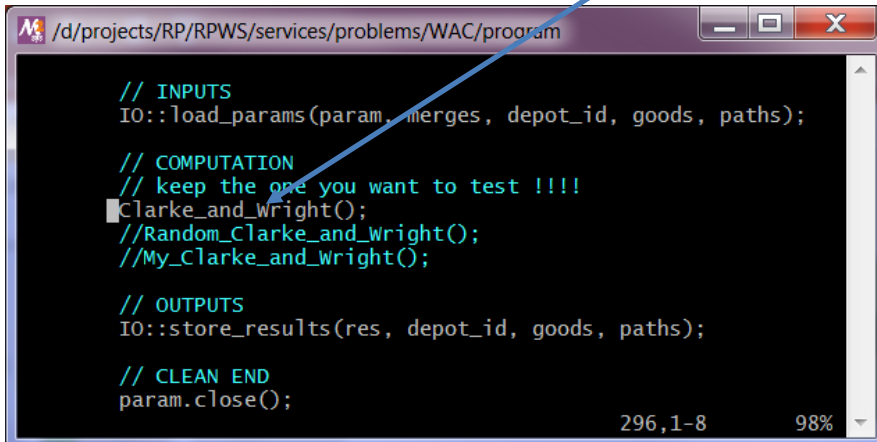


```
/cygdrive/d/projects/RP/RPWS/services/problems/...
$ ls
algorithms.json  algorithms.json~  instance.json  pro
bome1@UBS-19760 /cygdrive/d/projects/RP/RPWS/services
$ cp -r program program2
bome1@UBS-19760 /cygdrive/d/projects/RP/RPWS/services
$ ls
algorithms.json  instance.json  program2  WAC2.php
algorithms.json~  program      WAC1.php
bome1@UBS-19760 /cygdrive/d/projects/RP/RPWS/services
$
```

Modify WAC\program\main.cc to call Clarke\_and\_Wright()

Modify WAC\program2\main.cc to call Random\_Clarke\_and\_Wright()

Compile and test them.

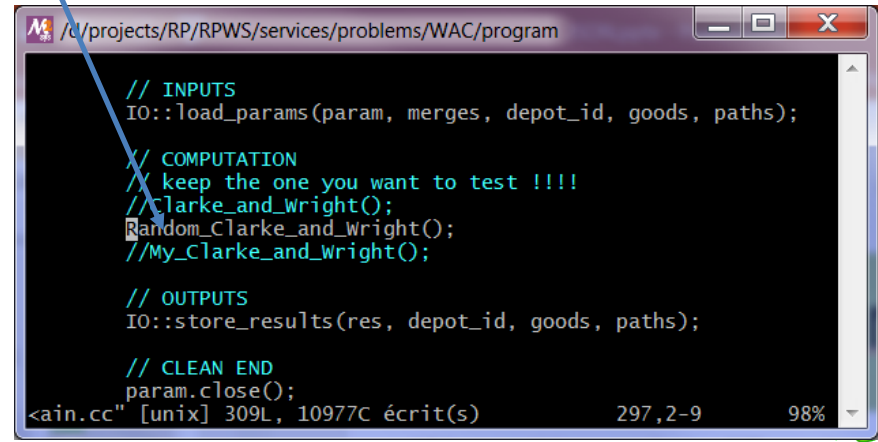


```
/d/projects/RP/RPWS/services/problems/WAC/program
// INPUTS
IO::load_params(param, merges, depot_id, goods, paths);

// COMPUTATION
// keep the one you want to test !!!!
Clarke_and_Wright();
//Random_Clarke_and_Wright();
//My_Clarke_and_Wright();

// OUTPUTS
IO::store_results(res, depot_id, goods, paths);

// CLEAN END
param.close();
296,1-8 98%
```



```
/d/projects/RP/RPWS/services/problems/WAC/program
// INPUTS
IO::load_params(param, merges, depot_id, goods, paths);

// COMPUTATION
// keep the one you want to test !!!!
Random_Clarke_and_Wright();
//My_Clarke_and_Wright();

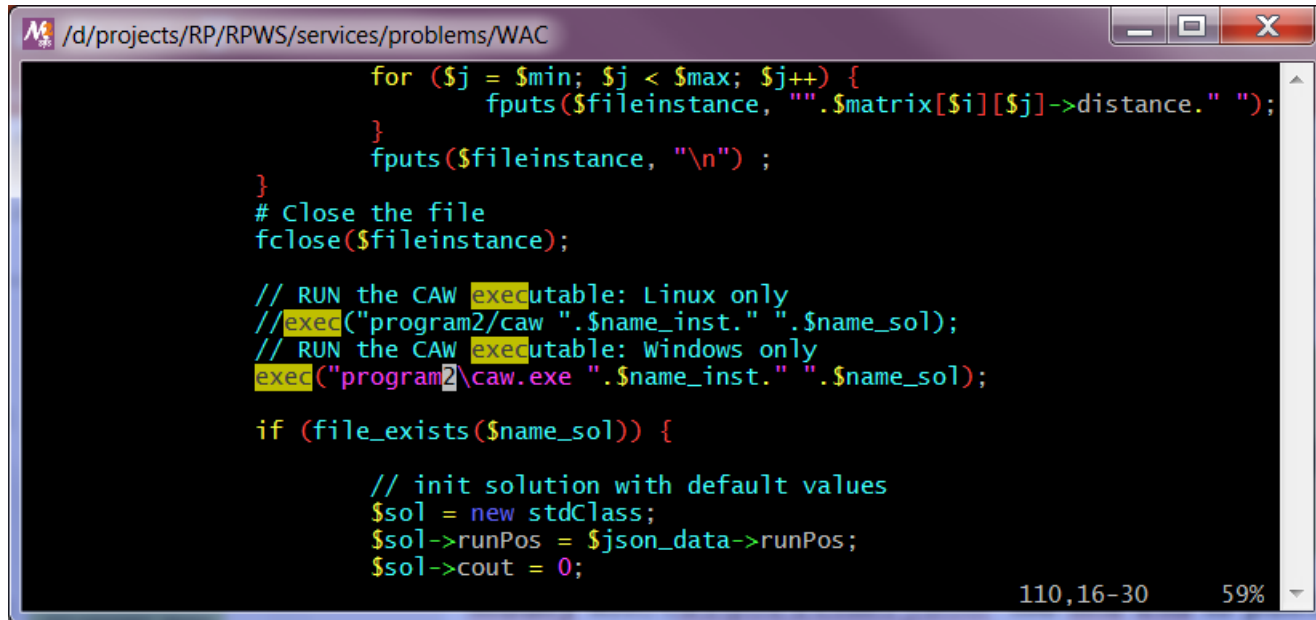
// OUTPUTS
IO::store_results(res, depot_id, goods, paths);

// CLEAN END
param.close();
<ain.cc" [unix] 309L, 10977C écrit(s) 297,2-9 98%
```



# JSON customization - algorithm

Modify `WAC\WAC2.php` file like this to point on the good executable, which is now located under `WAC\program2`.



```
/d/projects/RP/RPWS/services/problems/WAC  
  
        for ($j = $min; $j < $max; $j++) {  
            fputs($fileinstance, "$matrix[$i][$j]->distance." " ");  
        }  
        fputs($fileinstance, "\n" );  
    }  
    # Close the file  
    fclose($fileinstance);  
  
    // RUN the CAW executable: Linux only  
    //exec("program2/caw ".$name_inst." ".$name_sol);  
    // RUN the CAW executable: Windows only  
    exec("program2\caw.exe ".$name_inst." ".$name_sol);  
  
    if (file_exists($name_sol)) {  
        // init solution with default values  
        $sol = new stdClass;  
        $sol->runPos = $json_data->runPos;  
        $sol->cout = 0;  
    }  
  
110,16-30 59%
```

(You will learn more about PHP coding, just after the JSON part)



# JSON customization - algorithm

Restart your browser, and click on « run »  
for WAC1 and WAC2 to see the resulting paths.  
Now the paths are different and you can compare them easily.

Routing Problems

localhost/#

Accueil du site de l'Un... FOREVER Google Google Traduction ESP32 Sémantique Crédit Agricole Esp... France.gouv.fr

Lab-STICC ubs: ONIS

Welcome to the Routing Problems website !

search

Problem: Wright and Clarke

Real instance

Data's instance

Distance matrix

Algorithms

select an algorithm:

name	time	rank	exact	exec.	info
WAC1	<10s	1	no	RUN	?
WAC2	<10s	1	no	RUN	?

Results:

WAC1	0:00:00
WAC2	0:00:00

Solution from WAC1 (1)

trips: ALL 250 0 0 0.7 display

Properties:

runPos	0
cout	16309.3
duree	0
success	true
verif	0

Click here to see WAC1's paths

Paris

14e Arrondissement

15e Arrondissement

16e Arrondissement

17e Arrondissement

18e Arrondissement

19e Arrondissement

20e Arrondissement

1er Arrondissement

2e Arrondissement

3e Arrondissement

4e Arrondissement

5e Arrondissement

6e Arrondissement

7e Arrondissement

8e Arrondissement

9e Arrondissement

10e Arrondissement

11e Arrondissement

12e Arrondissement

13e Arrondissement

# JSON customization - algorithm

Restart your browser, and click on « run »  
for WAC1 and WAC2 to see the resulting paths.  
Now the paths are different and you can compare them easily.

Routing Problems

localhost/#

search

Welcome to the Routing Problems website !

Problem: Wright and Clarke

Real instance

Data's instance

Distance matrix

Algorithms

select an algorithm:

name	time	rank	exact	exec.	info
WAC1	<10s	1	no	RUN	?
WAC2	<10s	1	no	RUN	?

Results:

WAC1	0:00:00
WAC2	0:00:00

Solution from WAC2 (2)

trips: ALL 250 0 0 0.7 display

Properties:

runPos	1
cout	27484.8
duree	0
success	true
verif	0

Click here to see WAC2's paths

# Spring School on Integrated Operational Problems

May 14-16, 2018, Troyes, France

## PLAN

1rst session

Introduction (15-30 minutes)

C++ (1 hour)

The Clarke and Wright algorithm

Modifying the CAW and testing it

Adding your own strategy to the CAW

How to debug ?

2nd session

JSON (30 minutes)

Understanding the role of the JSON files

Adding two new algorithms to the CAW

PHP (30 minutes)

Adding a new problem to the web-site

Conclusion will be yours !