Optimization Software.
Heuristic Solvers for VRPs.

Introduction to OR-Tools.

Ruslan Sadykov

INRIA Bordeaux—Sud-Ouest

17 October 2022

OR-Tools : open-source solvers for combinatorial optimization

- Linear Programming Solver
- A wrapper for Mixed-Integer Linear Programming Solvers
- (Original) Constraint Programming Solver
- CP-SAT Solver (discrete domains)
- Solver for Scheduling Problems (based on CP solvers)
- Solver for Vehicle Routing Problems (based on the original CP solver)

OR-Tools : open-source solvers for combinatorial optimization

- Linear Programming Solver
- A wrapper for Mixed-Integer Linear Programming Solvers
- (Original) Constraint Programming Solver
- CP-SAT Solver (discrete domains)
- Solver for Scheduling Problems (based on CP solvers)
- Solver for Vehicle Routing Problems (based on the original CP solver)

OR-Tools : Installation and running Jupyter Notebooks

First, create a new folder and go there. Then

```
python3 -m pip install --upgrade --user ortools
git clone https ://github.com/google/or-tools.git or-tools
cd or-tools/examples/notebook
jupyter notebook
```

OR-Tools : Constraint Programming Example

```python
from ortools.constraint_solver import pywrapcp
solver = pywrapcp.Solver(’CPSimple’)
num_vals = 3
x = solver.IntVar(0, num_vals - 1, ’x’)
y = solver.IntVar(0, num_vals - 1, ’y’)
z = solver.IntVar(0, num_vals - 1, ’z’)
solver.Add(x  != y)
decision_builder = solver.Phase([x, y, z],
                                solver.CHOOSE_FIRST_UNBOUND,
                                solver.ASSIGN_MIN_VALUE)
status = solver.Solve(model)
solution = ’Solution :\n’
for var in [x, y, z] :
    solution += ’ {} = {}’.format(var.Name(), var.Value())
print(solution)
```

See Notebook

`constraint_solver/simple_cp_program.ipynb`

OR-Tools : Other Constraint Programming Examples

- Notebook `constraint_solver/cp_is_fun_cp.ipynb`
- Notebook `constraint_solver/nqueens_cp.ipynb`

OR-Tools : Simple VRP example

Guide :
`developers.google.com/optimization/routing/vrp`
Notebook : `constraint_solver/vrp.ipynb`

OR-Tools : CVRP example (+ Heterogeneous Fleet)

Guide :
`developers.google.com/optimization/routing/cvrp`
Notebook : `constraint_solver/cvrp.ipynb`

OR-Tools : VRP layer (I)

Two sets of indices

- Nodes (a special node is depot)
- Vehicles

Path variables

- **next(i)** — immediate successor of node *i*
- **vehicle(i)** — vehicle visiting node *i*
- **active(i)** — node *i* is visited or not

Relations between path variables

- $active(i) = 0 \Leftrightarrow next(i) = i \Leftrightarrow vehicle(i) = -1$
- $next(i) = j \Rightarrow vehicle(i) = vehicle(j)$

OR-Tools : VRP layer (II)

Dimension variables

- **cumul(i,d)** — quantity of dimension *d* when arriving to node *i*
- **transit(i,d)** — quantity of dimension *d* added after visiting node *i*

Relation between path and dimension variables

- $next(i) = j \Rightarrow cumul(j, d) = cumul(i, d) + transit(i, d)$

Costs

- Vehicle arc costs
- Dimension span costs (i.e. maximim time)
- Disjunction costs (node is visited or not)

OR-Tools : VRPTW example

Guide :

`developers.google.com/optimization/routing/vrptw`

Notebook :

`constraint_solver/vrp_time_windows.ipynb`

OR-Tools : Multi-Depot VRP example

Notebook :
`constraint_solver/vrp_starts_ends.ipynb`

OR-Tools : Team Orienteering example

Guide :

`developers.google.com/optimization/routing/penalties`

Notebook : `constraint_solver/vrp_drop_nodes.ipynb`

OR-Tools : Routing and Scheduling example

Guide :

`developers.google.com/optimization/routing/cvrptw_resources`

Notebook : `constraint_solver/vrp_resources.ipynb`

OR-Tools : Multi-Trip VRP example

Notebook : `constraint_solver/cvrp_reload.ipynb`