Monte Carlo Vehicle Routing

Tristan Cazenave[1] and **Jean-Yves Lucas**[2] and **Hyoseok Kim**[2] and **Thomas Triboulet**[2]

Abstract. Nested Rollout Policy Adaptation (NRPA) is a Monte Carlo search algorithm that learns a playout policy in order to solve a single player game. In this paper we apply NRPA to the vehicle routing problem. This problem is important for large companies that have to manage a fleet of vehicles on a daily basis. Real problems are often too large to be solved exactly. The algorithm is applied to standard problem of the literature and to the specific problems of EDF (Electricité De France, the main french electric utility company). These specific problems have peculiar constraints. NRPA gives better result than the algorithm previously used by EDF.

1 Introduction

Monte Carlo Tree Search (MCTS) has been successfully applied to many games and problems [8].

Nested Monte Carlo Search (NMCS) [9] is an algorithm that works well for puzzles. It biases its playouts using lower level playouts. At level zero NMCS adopts a uniform random playout policy. Online learning of playout strategies combined with NMCS has given good results on optimization problems [38]. Other applications of NMCS include Single Player General Game Playing [32], Cooperative Pathfinding [6], Software testing [36], heuristic Model-Checking [37], the Pancake problem [7], Games [11], Cryptography [19] and the RNA inverse folding problem [33].

Online learning of a playout policy in the context of nested searches has been further developed for puzzles and optimization with Nested Rollout Policy Adaptation (NRPA) [40]. NRPA has found new world records in Morpion Solitaire and crosswords puzzles. Stefan Edelkamp and co-workers have applied the NRPA algorithm to multiple problems. They have optimized the algorithm for the Traveling Salesman with Time Windows (TSPTW) problem [12, 21]. Other applications deal with 3D Packing with Object Orientation [23], the physical traveling salesman problem [24], the Multiple Sequence Alignment problem [25] or Logistics [22]. The principle of NRPA is to adapt the playout policy so as to learn the best sequence of moves found so far at each level.

Electricité De France (EDF) is the main french electric utility company. Each year, eleven millions of services have to be carried out by EDF technicians at customers places (problem fixing, meter replacement, energetic diagnosis, business prospects). Thus, all together, EDF techicians drive by car more than 220 000 kilometers per year. Each service is accomplished by a technician having the required skills within a time window determined during the appointment booking. As a result, numerous Vehicle Routing Problems with Time Windows (VRPTW) have to be solved on a daily basis (one problem per catchment area). Any of these VRPTW involves quite a big search space, since EDF technicians of a single catchment area have to achieve every day hundreds of visits at customers places. The objective of the work described in this paper was to adapt the NRPA algorithm to the specifics of EDF problem.

Related approaches that apply Artificial Intelligence techniques to transportation include agent based approaches [5], Monte Carlo Search approaches [42, 30, 22, 1] and learning of simulations of urban traffic [16]. Our work is close to the Monte-Carlo Search approach applied to the Capacitated Vehicle Routing Problem with Time Windows (CVRPTW).

We now give the outline of the paper. The next section briefly describes the state of the art of Vehicle Routing. The third section details the NRPA algorithm and its proposed improvements. The fourth section describe the problems of EDF and the current solver. The fifth section gives experimental results for various instances of Vehicle Routing, both from literature (Solomon instances) and from real-life problems. The last section concludes.

2 The Vehicle Routing Problem

The Vehicle Routing Problem (VRP) is one of the most studied optimization problem. It was first stated in 1959 in [18]. Basically, it consists of finding an optimal route for a number of vehicles that are used to deliver goods or services to a set of customers, taking into account a set of constraints. In the simpler variant of the problem, all vehicles start from a single depot and end their tour in this depot. The objective function to be minimized may combine up to three criteria (namely the number of customers that are not serviced, the number of vehicles used, the total distance travelled by the vehicles), each criterion having an associated weight. The VRP can be formalized as a graph problem : let G = (V, A) be a directed graph where V is the set of vertices, and A is the set of arcs. Each arc is labelled with a non-negative number. One of the vertices represent the depot, the others represent the customers locations. The arcs symbolize the roads between two customers, and the label of the arcs give the distance (or the travel time, or the travel cost) between two points. Lets us remind that the G is a directed graph, and thus its associated distance matrix is not necessarily symmetric.Now the problem consists of finding the minimum number of tours so that each vertex except the depot belongs to one and only one tour, and the depot belongs to all of them. The VRP is of the non deterministic polynomial time hard type (NP-Hard), which implies that so far we don't know of a general method which is able to optimally solve any instance of the problem in polynomial time. As a result, the VRP is considered very difficult. Nowadays exact methods can solve to optimality quite challenging instances of the problem (for instance up to one hundred visits or so with Branch and Price methods).

[1] LAMSADE, Université Paris-Dauphine, PSL, CNRS, France, email: Tristan.Cazenave@dauphine.psl.eu

[2] EDF R&D, France, email: jean-yves.lucas@edf.fr hyoseok.kim@ensiie.fr thomas.triboulet@edf.fr

2.1 VRP variations

The VRP is a real challenge for delivery companies operating a fleet of vehicles. It has given rise to a number of variations. The CVRP (Capacitated VRP) is defined as a VRP with a demand associated to each customers (e.g. the number of parcels they have purchased) and each vehicle has a limited capacity. The VRP with time windows (VRPTW) implies to serve each customer within a given time window (possibly different for each customer). The capacitated VRP with time windows (CVRPTW) combines the characteristics of the two previous variants. Although the previous variations are the most widely studied because of their great practical importance, many other extensions of the basic VRP have also been proposed. Among them, let us just mention two of them. First, the dynamic VRP (DVRP) where vehicles may be dynamically re-routed during their tour in order to fulfill new customers orders. Second, the VRP with Pickup and Delivery (VRPPD) where a fleet of vehicles must satisfy transportation requests (e.g. picking parcels up at some places and delivering them at given locations).

2.2 Principal methods used for solving VRP

A wide range of methods have been used to solve the VRP. These methods may be broken down into three sub-groups, namely exact methods, heuristic methods and metaheuristic methods. These methods are presented hereafter.

Exact methods These methods tend to find an optimal solution for the VRP. As mentioned above, they are thus used to solve instances of the VRP of consequent size. Among the most studied exact methods for the VRP and variations we may mention the Branch-and-Cut and the Branch-and-Price algorithms [28], the column-generation algorithm [4] and the set-partitioning method [2].

These methods can also provide a bound of optimum, especially in relaxing some of the most complicated constraints. For example, in suppressing the classical subtour elimination constraint.

Heuristic methods Not in a comprehensive manner, let us mention two interesting approaches. First the cluster-first, route-second heuristic [26]. Second, the savings heuristic [14] [3]. For the local search approaches we see in [44] a heuristic projection pool with powerful insertion and guided local search strategies. The local search described in [39] gives reference solutions on several instances of the benchmark VRP Solomon instances tested in this work.

Metaheuristic methods The most commonly used metaheuristics for solving VRP and variations are the particle swarm optimization [31], simulated annealing [13], genetic algorithm [34] [43] and tabu search [15] [35]. Among the best algorithms for solving the VRP Solomon instances tested in this work, we can cite genetic algorithm described in [29], evolutionnary algorithm detailed in [17] or tabu Search proposed in [20].

3 Nested Rollout Policy Adaptation for Vehicle Routing

In this section we start with explaining the NRPA algorithm. Then we give our modelling of the CVRPTW problem for NRPA. We finish the section describing how weight are heuristically initialized in order to speed-up convergence of the algorithm.

3.1 Description of NRPA

An effective combination of nested levels of search [9] and of policy learning has been proposed with the NRPA algorithm [40]. NRPA holds world records for Morpion Solitaire and crosswords puzzles.

NRPA is given in algorithm 3. The principle is to learn weights for the possible actions so as to bias the playouts. The playout algorithm is given in algorithm 1. It performs Gibbs sampling, choosing the actions with a probability proportional to the exponential of their weights.

The adaptive rollout policy is a policy parameterized by weights on each action. During the playout phase, action is sampled according to this weights. The playout algorithm is given in algorithm 1. It uses Gibbs sampling, each move is associated to a weight. A move is coded as an integer that gives the index of its weight in the policy array of floats. The algorithm starts with initializing the sequence of moves that it will play (line 2). Then it performs a loop until it reaches a terminal states (lines 3-6). At each step of the playout it calculates the sum of all the exponentials of the weights of the possible moves (lines 7-10) and chooses a move proportional to its probability given by the softmax function (line 11). Then it plays the chosen move and adds it to the sequence of moves (lines 12-13).

Then, the policy is adapted on the best current sequence found, by increasing the weight of the best actions. The Adapt algorithm is given in algorithm 2. The weights of the actions are updated at each step of the algorithm so as to favor moves of the best sequence found so far at each level. The principle of the adaptation is to add α to the action of the best sequence for each state encountered in the best sequence (lines 3-5) and to decrease the weight of the other possible actions by an amount proportional to their probabilities of being played (lines 6-12). The adaptation algorithm is given in algorithm 2.

In NRPA, each nested level takes as input a policy, and returns a sequence. Inside the level, the algorithm makes many recursive calls to lower levels, providing weights, getting sequences and adapting the weights on those sequences. In the end, the algorithm returns the best sequence found in that level. At the lowest level, the algorithm simply makes a rollout.

The NRPA algorithm is given in algorithm 3. At level zero it simply performs a playout (lines 2-3). At greater levels it performs N iterations and for each iteration it calls itself recursively to get a score and a sequence (lines 4-7). If it finds a new best sequence for the level it keeps it as the best sequence (lines 8-11). Then it adapts the policy using the best sequence found so far at the current level (line 12).

NRPA balances exploitation by adapting the probabilities of playing moves toward the best sequence of the level, and exploration by using Gibbs sampling at the lowest level. It is a general algorithm that has proven to work well for many optimization problems.

Playout policy adaptation has also been used for games such as Go [27] or various other games with success [10].

3.2 Modelling of the problem

There are several design choices when implementing NRPA. The first choice is to decide how to code the possible moves. For the vehicle routing problem we choose to code a move as a starting node, and arrival node and a vehicle.More precisely, a solution to the problem is an ordered sequence of visits, including special visits (SV) that represent the fact that the corresponding technician is at the depot. So a sequence solution always starts and ends with a SV. If there

Algorithm 1 The playout algorithm

```
playout ($state$, $policy$)
    $sequence \leftarrow$ []
    while true do
        if $state$ is terminal then
            return (score ($state$), $sequence$)
        end if
        $z \leftarrow 0.0$
        for $m$ in possible moves for $state$ do
            $z \leftarrow z$ + exp ($policy$ [code($m$)])
        end for
        choose a $move$ with probability $\frac{exp(policy[code(move)])}{z}$
        $state \leftarrow$ play ($state$, $move$)
        $sequence \leftarrow sequence$ + $move$
    end while
```

Algorithm 2 The Adapt algorithm

```
Adapt ($policy$, $sequence$)
    $polp \leftarrow policy$
    $state \leftarrow root$
    for $move$ in $sequence$ do
        $polp$ [code($move$)] $\leftarrow$ $polp$ [code($move$)] + $\alpha$
        $z \leftarrow 0.0$
        for $m$ in possible moves for $state$ do
            $z \leftarrow z$ + exp ($policy$ [code($m$)])
        end for
        for $m$ in possible moves for $state$ do
            $polp$ [code($m$)] $\leftarrow$ $polp$ [code($m$)] - $\alpha * \frac{exp(policy[code(m)])}{z}$
        end for
        $state \leftarrow$ play ($state$, $move$)
    end for
    $policy \leftarrow polp$
```

Algorithm 3 The NRPA algorithm.

```
NRPA ($level$, $policy$)
    if level == 0 then
        return playout (root, $policy$)
    else
        $bestScore \leftarrow -\infty$
        for N iterations do
            (result,new) $\leftarrow$ NRPA($level - 1$, $policy$)
            if result $\geq$ bestScore then
                bestScore $\leftarrow$ result
                seq $\leftarrow$ new
            end if
            policy $\leftarrow$ Adapt (policy, seq)
        end for
        return (bestScore, seq)
    end if
```

is a SV between two visits within the sequence, that means the end of a tour for a technician, and the beginning of a tour for another technician (with no chronological ordering of these two tours that will be carried out simultaneously).

Another design choice is to define the score of a playout. Score includes number of non visited customers multiplied by a great penalization number, number of used vehicles multiplied by an rather great weight, and number of kilometers. We filter movements that do not respect the time windows in the playout algorithm : all solutions respect customers time windows and vehicle time windows.

If there were more objectives, a lexicographic ordering of the objectives would be the best way to represent the scores of a playout. Playouts could then be compared simply and easily.

3.3 Initialization of the Weights

In NRPA weights are uniformly initialized to 0.0. We propose to initialize the weight of NRPA heuristically in order to speedup convergence. The greater the distance from the current city to another city the less likely the other city is a good choice. Standard NRPA starts with an uniform policy with all weights set to 0.0 and does not distinguishes between close and far cities. The heuristic initialization of the weights initializes the weight with a value proportional to the inverse of the distance. This initialization is only used for the first iteration of NRPA.

Algorithm 4 The NRPA algorithm with quantiles.

```
NRPA_with_quantile ($level$, $policy$)
    allScores $\leftarrow$ []
    quantile $\leftarrow -\infty$
    if level == 0 then
        (result,new) $\leftarrow$ playout(root,policy)
        allScores $\leftarrow$ allScores + result
        quantile $\leftarrow$updateQuantile(allScores)
        return (result,new)
    else
        $bestScore \leftarrow -\infty$
        for N iterations do
            (result,new) $\leftarrow$ NRPA($level - 1$, $policy$)
            if result $\geq$ bestScore then
                bestScore $\leftarrow$ result
                seq $\leftarrow$ new
            else
                if result $\leq$ quantile then
                    policy $\leftarrow$ Adapt_Bad(policy, new)
                end if
            end if
            policy $\leftarrow$ Adapt(policy, seq)
        end for
        return (bestScore, seq)
    end if
```

3.4 The Quantile Heuristic

The objective of the Quantile Heuristic is to penalize movements from bad solutions. Solution scores for all playouts need to be stored. When a new playout is computed, if its score is worse than a defined quantile, then its movements weights are decreased in the weight. Efficient implementation is needed to limit the increase of computation time needed to sort the solution scores and compute the quantile. Sequence of algorithm is described in 4. The adapt function is the same as for good solutions, with negative coefficient α, as described in 5. The coefficient α used for bad solutions can be different from the one used to adapt policy to good solutions.

Algorithm 5 The adapt algorithm to bad solutions

```
Adapt_Bad ($policy$, $sequence$)
    $polp \leftarrow policy$
    $state \leftarrow root$
    for $move$ in $sequence$ do
        $polp$ [code($move$)] $\leftarrow$ $polp$ [code($move$)] - $\alpha$
        $z \leftarrow 0.0$
        for $m$ in possible moves for $state$ do
            $z \leftarrow z$ + exp ($policy$ [code($m$)])
        end for
        for $m$ in possible moves for $state$ do
            $polp$ [code($m$)] $\leftarrow$ $polp$ [code($m$)] + $\alpha * \frac{exp(policy[code(m)])}{z}$
        end for
        $state \leftarrow$ play ($state$, $move$)
    end for
    $policy \leftarrow polp$
```

4 The EDF Capacitated Vehicle Routing Problem with Time Windows

In this section we first explain the optimization problem encountered at EDF and then describe the current EDF approach to the problem.

4.1 Description of the EDF problems

The Vehicle Routing Problem modeled here includes time windows, which is a classical feature of VRP problems. That means that :

- Each technician has an availability time window. Each tour starts at the beginning of this time window and must end before the end of this time window.
- Each appointment has a time window and a duration. A tour visiting the appointment must start after the beginning of the time window and end (including the appointment duration) before the end of the time window
- If a tour arrives at an appointment location before the beginning of its time windows, it is possible to add waiting time before starting the appointment.

Time windows and technicians are input data of the problem. Duration of appointments and duration of all possible trajects between appointments or depot and appointments are also input data of the problem.

The problem also includes capacities, another classical feature of VRP problems : Each vehicle has several stock capacities (in EDF problem 6 per vehicle).

- Each vehicle has several stock capacities (in EDF problem 6 per vehicle). Vehicle stocks are initialized with initial capacity at the begining of each tour.
- For any stock, each operation will consume a stock quantity.
- For each tour, for each capacity, the sum of the quantity used by the operation of the tour must not be greater than the capacity of the stock.

Several peculiar features are taken into account in EDF modelization.

- Taking into account an off-duty time window for lunch break. No place is imposed for it. It can interrupt a trip but not a service to a customer.
- Taking into account specific skills for visits. For each visit, only a subset of technicians is skilled to carry out the technical operations.
- Traject distance and traject duration are not proportionnal, because vehicle mean speed depends on the traject. Consequently, 2 different traject matrixs are provided.
- Traject matrix and are not symetric : traject between two points depends on the sense of the traject.

Another difference with academic data is that on real problems, there is not always enough people to carry out the whole set of visits. Consequently, we must first maximize the number of visits that will be done. Some of the appointments have a high priority, and because of customers satisfaction, it is not possible to cancel it. Other appointments have less priority, because they consist for example in a technical operation that can be postponed, with no corresponding customer appointment. Moreover, network preventive maintenance visits have less priority than troubleshooting activities. In order to evaluate a solution, a lexicographic objective function is taken into account :

- Maximization of the number of achieved visits of high priority.
- Maximization of the number of achieved visits of low priority.
- Minimization of the economic function, taking into account number of technicians used (ponderated with a proportional daily wage) and number of kms (ponderated with a proportional km cost).

Thus, when comparing two solutions, we first compare the number of visits of high priority that are achieved in each solution.if these numbers are not equal then we know which solution is best. If they are equal, then we compare the number of visits of low priority, if again these numbers are equal, then we compare the third criterion of the two solutions, that is the economic function. The problem is solved each day for determining the technicians tours of the next day, with a computing time which must not last more than an few hours.

4.2 Description of current EDF approach

Current approach for the problem is based on stochastic greedy algorithm and variable neighborhood search. The stochastic greedy algorithm is based on [41] algorithm. First phase consist in creating tours and insert in each tour the visits that increases the less the travelled distance, until there is no room for new visits. For each insertion, the place in the tour is chosen to minimize the increase of kms. If no insertion is possible, a new round is created. Then the process is repeated, until no visits nor technicians are left.

The local search consist of 2 kinds of movements :

- Small movements consist in choosing randomly an appointment, and finding the best place to move the appointment, i.e. the best place in all the existing tours to reinsert the appointment.
- Large movements consist in choosing randomly a tour, then entirely destroying it and trying to reinsert all its appointments in the other tours. If no improvement is possible, the movement is canceled.

Advantages of such a method is that it computes rapidly good solutions, very often acceptable by operators because greedy algorithm choices, based on distance, are humanly intuitive.

Table 1: The different algorithms tested on the 56 standard instances.

	nrpa		nrpaD		nrpaDQ		opturn	
instance	V	Km	V	Km	V	Km	V	Km
c101	**10**	**828.94**	**10**	**828.94**	**10**	**828.94**	10	852.95
c102	10	986.77	10	858.18	**10**	**843.57**	11	1458.24
c103	10	1117.30	10	872.45	**10**	**857.14**	11	1674.76
c104	10	1120.67	**10**	**894.83**	10	901.79	11	1640.93
c105	11	900.69	**10**	**828.94**	**10**	**828.94**	11	1102.17
c106	11	940.90	**10**	**828.94**	**10**	**828.94**	11	1465.60
c107	10	1071.41	**10**	**828.94**	**10**	**828.94**	10	994.84
c108	10	1104.49	**10**	**830.16**	10	831.04	11	1674.53
c109	10	1061.88	**10**	**842.43**	10	849.09	11	1688.91
c201	4	723.69	**3**	**591.56**	**3**	**591.56**	4	974.46
c202	4	870.43	3	611.08	**3**	**609.23**	4	952.33
c203	4	1108.55	**3**	**617.06**	3	628.40	4	1058.52
c204	4	1043.12	3	773	**3**	**629.67**	4	1265.41
c205	4	720.12	3	591.17	**3**	**588.88**	4	1007.44
c206	4	753.90	**3**	**590.79**	3	600.88	4	1252.91
c207	4	733.45	**3**	**597.53**	**3**	**597.53**	4	1032.26
c208	4	717.03	**3**	**601.05**	3	604.36	4	1189.83
r101	19	1665.61	19	1656.54	**19**	**1655.36**	22	2187.97
r102	18	1520.63	17	1546.12	**17**	**1523.75**	19	1967.61
r103	14	1331.52	14	**1287.66**	**13**	1379.44	17	1887.31
r104	11	1113.42	11	**1074.57**	**10**	1112.86	13	1524.18
r105	15	1467.54	**14**	**1439.11**	14	1456.76	17	1931.07
r106	13	1346.79	**12**	1313.12	13	**1301.35**	15	1687.18
r107	12	1222.77	11	1163.35	**11**	**1158.45**	14	1621.52
r108	11	1142.77	10	1008.88	**10**	**1004.15**	12	1414.54
r109	13	1290.98	**12**	**1218.86**	12	1220.98	16	1866.01
r110	12	1236.98	11	1159.06	**11**	**1158.85**	13	1625.52
r111	12	1181.52	**11**	**1125.77**	11	1189.42	14	1659.14
r112	11	1105.92	**10**	**1018.92**	10	1023.14	12	1513.96
r201	4	1528.10	**4**	**1419.93**	4	1445.68	5	1889.49
r202	4	1369.36	**4**	**1264.17**	4	1305.25	6	1871.68
r203	4	1276.17	3	1198.43	**3**	**1137.19**	4	1632.11
r204	3	1086.88	**3**	**879.22**	3	890.48	4	1127.19
r205	4	1226.37	3	1182.85	**3**	**1174.74**	4	1475.82
r206	3	1230.22	3	1114.39	**3**	**1098.25**	4	1357.13
r207	3	1243.13	3	1017.76	**3**	**975.49**	3	1327.14
r208	3	1048.79	2	860.49	**2**	**859.89**	3	1076.75
r209	4	1191.46	3	1136.35	**3**	**1055.18**	4	1495.34
r210	4	1238.99	**3**	**1117.78**	3	1151.20	4	1588.28
r211	3	1059.63	3	919.90	**3**	**895.23**	3	1239.46
rc101	16	1701.31	**15**	**1625.45**	15	1640.51	19	2483.55
rc102	14	1560.96	**13**	1549.49	14	**1511.67**	16	2261.19
rc103	12	1426.39	**11**	**1321.51**	11	1356.68	14	1952
rc104	11	1264.83	10	1202.19	**10**	**1165.98**	13	1686.93
rc105	15	1610.40	14	1652.48	**14**	**1634.49**	19	2343.03
rc106	12	1451.27	13	**1417.10**	**12**	1428.08	14	1952.11
rc107	12	1404.08	11	1312.12	**11**	**1289.70**	14	1755.26
rc108	11	1380.20	11	1197.96	**11**	**1188.93**	14	1888.36
rc201	5	1695.44	4	1675.35	**4**	**1644.26**	5	2537.02
rc202	5	1439.66	4	1423.27	**4**	**1378.31**	4	2037.64
rc203	4	1384.52	**3**	**1265.06**	3	1295.67	4	1571.54
rc204	3	1185.91	**3**	**960.21**	3	962.46	4	1447.34
rc205	5	1601.08	**4**	**1497.85**	4	1508.59	5	2313.72
rc206	4	1447.54	**3**	**1341.10**	3	1348.50	4	1856.01
rc207	4	1374.81	3	1299.20	**3**	**1294.41**	4	1773.09
rc208	3	1279.30	**3**	**1007.62**	3	1016.23	3	1494.43

Table 2: Summary of the different algorithms tested on the 56 standard instances.

nrpa			nrpaD			nrpaDQ			opturn		
V (mean)	Km (mean)	δ	V (mean)	Km (mean)	δ	V (mean)	Km (mean)	δ	V (mean)	Km (mean)	δ
0	8.21	1216.72	7.59	1097.47	53	**7.57**	**1094.40**	**55**	9.16	1600.10	-50

Table 3: Results for the different algorithms tested on the EDF instances.

	nrpa			nrpaD			nrpaDQ			opturn		
instance	Missed	V	Km	Missed	V	Km	Missed	V	Km	Missed	V	Km
EDF1	8	6	649.17	8	6	464.50	**8**	**6**	**433.97**	9	6	613.41
EDF2	0	6	454.90	**0**	**6**	**448.92**	0	6	456.63	1	6	575.99
EDF4	1	5	172.90	**1**	**5**	**160.99**	1	5	164.98	1	5	222.15
EDF5	**0**	**2**	**224.06**	**0**	**2**	**224.06**	**0**	**2**	**224.06**	0	2	271.24
EDF6	**0**	**1**	**85.27**	**0**	**1**	**85.27**	**0**	**1**	**85.27**	0	1	88.71
EDF7	0	4	302.72	**0**	**4**	**302.7**	0	4	302.8	2	4	370.73
EDF8	2	3	156.12	2	3	158.23	**2**	**3**	**152.40**	2	3	115.63
EDF9	4	11	861.30	**4**	**11**	**767.63**	4	11	777.35	4	12	498.19

Table 4: Summary for the different algorithms tested on the EDF instances.

	nrpa			nrpaD			nrpaDQ			opturn		
	Missed	V	Km	Missed	V	Km	Missed	V	Km	Missed	V	Km
Summary (mean)	1.88	4.75	363.30	1.88	4.75	326.54	**1.88**	**4.75**	**324.68**	2.13	4.88	344.51

5 Experimental Results

The operational process of EDF consists in providing a solution to the CVRPTW in less than two hours. Each problem is solved during the night so as to provide solutions for the next day. The parameters used in our experiments for NRPA are to use level 3, $\alpha = 1$ and 100 iterations per level. This algorithm is called nrpa in our tables. The results are given out of 11 independent runs. The distance heuristic is tested and the initialization of the weights is only applied at the first iteration. The corresponding name of the algorithm is nrpaD in the tables. The Quantile heuristic is applied in 80% of the worst cases. It uses all the simulations performed to date and uses a specific $\alpha = 0.5$. The algorithm that uses both the Distance and the Quantile heuristics is named nrpaDQ int the tables.

The current solver of EDF, Opturn is launched only once with 3 000 iterations of greedy search and 3 000 iterations of local search. These values were chosen during the tuning of Opturn after many experiments that showed that no improvements are obtained with bigger values. As a result, the running time of Opturn is smaller than the running time of the NRPA.

The running time for NRPA is approximately 7 000 seconds and up to 8 000 seconds when using the Quantile heuristic. The running time of Opturn is approximately 5 000 seconds. Importantly, these run times are compatible with the operational process as they last approximately two hours.

Table 1 gives the results of runs on the litterature instances. It compares 4 algorithms:

- nrpa : Standard NRPA (3 levels) without heuristics
- nrpaD : NRPA (3 levels) with Distance initialization heuristic
- nrpaDQ : NRPA (3 levels) with Distance initialization heuristic and Quantile heuristic
- opturn : current EDF solver, based on Solomon and LNS heuristic.

To compare the results of the 4 variants, the lexicographical approach takes into account first the number of vehicles used and then the kilometers.

- V : number of needed vehicles
- Km : sum of the Kms of the rounds

Table 2 gives the summary for all the runs of the algorithms on the standard problems. For the Km and the Vehicles (V), the figures are the average of each algorthm on the Solomon instances. The δ value is the number of problems that are solved better than with standard NRPA minus the number of problems that are solved worse. Comparison is lexicographic, based on number of vehicles, and if equals, on the number of Kms. Logically, the δ value is 0 for the reference (nrpa variant).

We observe that nrpaDQ scores 55 while Opturn scores -50. This is a great improvement over the current solver. However the current solver was optimized on the EDF problems. The Distance heuristic improves a lot over standard NRPA. The Distance + Quantile heuristic only improves slightly over the Distance heuristic alone.

Table 3 and Table 4 give the results of the 4 algorithms on the EDF instances. For each algorithm, the 3 components of objective function are displayed:

- Missed : number of missed appointments
- V : number of needed vehicles
- Km : sum of the Kms of the tours

Table 3 compares the results of runs of the algorithms on the EDF problems. For each instance we compare the number of realized technical operations because on real EDF instances there are not necessarily enough vehicles to achieve all the operations. We also measure the number of vehicles and the number of kilometers, as in classical VRP problem instances. To compare the results of the 4 variants, the lexicographical approach takes into account first the achieved operations, then the number of vehicles used and finally the kilometers.

Table 4 gives the average values on the whole set of EDF instances.

The average number of missed visits is smaller with NRPA than with Opturn. It is the same for all NRPA variants. The number of vehicles used for the operations is also slightly smaller and this is the same for all NRPA variants. The average number of kilometers is greater with standard NRPA than with Opturn, however it is smaller with nrpaD and nrpaDQ than with Opturn. With respect to the lexicographic objective function described in subsection 4.1, we can conclude that nrpaD and nrpaDQ improves on the current solver for the operational EDF instances.

The Distance and Quantile heuristics provide solutions with the same values for the 3 components of the objective function. These two heuristics perform better in average than the standard NRPA variant as for the number of travelled kilometers : 324 vs 363, that is a 10 percents improvement. If we compare with Opturn results, the improvement account for around 5 percents. This is still significant from a practical point of view because of the huge amount of kilometers travelled by the technicians on a yearly basis : about 220 millions of kilometers.

6 Conclusion

We have presented the NRPA algorithm and its application to the CVRPTW problems. This problem is important for EDF, a company that plans numerous operations every day on the french electrical network. We have given the modelization used to address the CVRPTW problem and two heuristics to improve on standard NRPA: the Distance heuristic and the Quantile heuristic. The Distance heuristic improves a lot on standard NRPA and the Quantile heuristic is a slight improvement. We also compared NRPA with the heuristics to the current EDF solver Opturn. On standard instances NRPA with the heuristic is much better. On the operational EDF instances it is still better even though Opturn was tuned for these instances while NRPA is a general algorithm that uses a Distance heuristic that works for all kinds of VRP problems.

As we have seen from the examples of the EDF instances, NRPA with heuristics performs better than the current solver. The percentage of kilometers saved by heuristic NRPA is greater than 5% of the total number of kilometers. EDF agents drive hundreds of millions of kilometers each year. The use of heuristic NRPA could save millions of kilometers each year and reduce the carbon footprint of EDF by hundreds of tons of CO_2.

REFERENCES

[1] Ashraf Abdo, Stefan Edelkamp, and Michael Lawo, 'Nested rollout policy adaptation for optimizing vehicle selection in complex vrps', in *2016 IEEE 41st Conference on Local Computer Networks Workshops (LCN Workshops)*, pp. 213–221. IEEE, (2016).

[2] Y. Agrawal, K. Mathur, and H.M. Salkin, 'A set-partitioning-based algorithm for the vehicle routing problem', *Networks*, **19**(7), 731–749, (1989).

[3] S.P. Anbuudayasankar, K. Ganesh, S.C. Lenny Koh, and Y. Ducq, 'Modified savings heuristics and genetic algorithm for bi-objective vehicle routing problem with forced backhauls', *Expert Systems and Applications*, **30**, 2296–2305, (2012).

[4] N. Azi, M. Gendreau, and J.-Y. Potvin, 'An exact algorithm for a vehicle routing problem with time windows and multiple use of vehicles', *European Journal of Operational Research*, **202**(3), 756–763, (2010).

[5] Ana LC Bazzan and Franziska Klügl, 'A review on agent-based technology for traffic and transportation', *The Knowledge Engineering Review*, **29**(3), 375–403, (2014).

[6] Bruno Bouzy, 'Monte-carlo fork search for cooperative path-finding', in *Computer Games - Workshop on Computer Games, CGW 2013, Held in Conjunction with the 23rd International Conference on Artificial Intelligence, IJCAI 2013, Beijing, China, August 3, 2013, Revised Selected Papers*, pp. 1–15, (2013).

[7] Bruno Bouzy, 'Burnt pancake problem: New lower bounds on the diameter and new experimental optimality ratios', in *Proceedings of the Ninth Annual Symposium on Combinatorial Search, SOCS 2016, Tarrytown, NY, USA, July 6-8, 2016*, pp. 119–120, (2016).

[8] Cameron Browne, Edward Powley, Daniel Whitehouse, Simon Lucas, Peter Cowling, Philipp Rohlfshagen, Stephen Tavener, Diego Perez, Spyridon Samothrakis, and Simon Colton, 'A survey of Monte Carlo tree search methods', *IEEE Transactions on Computational Intelligence and AI in Games*, **4**(1), 1–43, (March 2012).

[9] Tristan Cazenave, 'Nested Monte-Carlo Search', in *IJCAI*, ed., Craig Boutilier, pp. 456–461, (2009).

[10] Tristan Cazenave, 'Playout policy adaptation with move features', *Theor. Comput. Sci.*, **644**, 43–52, (2016).

[11] Tristan Cazenave, Abdallah Saffidine, Michael John Schofield, and Michael Thielscher, 'Nested monte carlo search for two-player games', in *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*, pp. 687–693, (2016).

[12] Tristan Cazenave and Fabien Teytaud, 'Application of the nested rollout policy adaptation algorithm to the traveling salesman problem with time windows', in *Learning and Intelligent Optimization - 6th International Conference, LION 6, Paris, France, January 16-20, 2012, Revised Selected Papers*, pp. 42–54, (2012).

[13] W.C. Chiang and R. Russel, 'Simulated annealing metaheuristics for the vehicle routing problem with time windows', *Annals of Operations Research*, **13**(1), 3–27, (1996).

[14] G. Clarke and J. Wright, 'Scheduling of vehicles from a central depot to a number of delivery points', *Operations Research*, **12**, 171–183, (1964).

[15] J-F. Cordeau, M. Gendreau, and G. Laporte, 'A tabu search heuristic for the periodic and multi-depot vehicle routing problems', *Networks*, **30**(2), 105–119, (1997).

[16] Luca Crociani, Gregor Lämmel, Giuseppe Vizzari, and Stefania Bandini, 'Learning obervables of a multi-scale simulation system of urban traffic.', in *ATT@ IJCAI*, pp. 40–48, (2018).

[17] O. Bräysy D. Mester and W. Dullaer, 'A multi-parametric evolution strategies algorithm for vehicle routing problems', in *Working Paper, Institute of Evolution, University of Haifa, Israel*, (2005).

[18] G.B. Dantzig and J.H. Ramser, 'The truck dispatching problem', *Management Science*, **6**(1), 80–91, (1959).

[19] Ashutosh Dhar Dwivedi, Paweł Morawiecki, and Sebastian Wójtowicz, 'Finding differential paths in arx ciphers through nested monte-carlo search', *International Journal of electronics and telecommunications*, **64**(2), 147–150, (2018).

[20] M. Gendreau F. Geurtin E. Taillard, P. Badeau and J.Y. Potvin, 'A tabu search heuristic for the vehicle routing problem with time windows', in *Transportation Science, 31*, pp. 170–186, (1997).

[21] Stefan Edelkamp, Max Gath, Tristan Cazenave, and Fabien Teytaud, 'Algorithm and knowledge engineering for the tsptw problem', in *Computational Intelligence in Scheduling (SCIS), 2013 IEEE Symposium on*, pp. 44–51. IEEE, (2013).

[22] Stefan Edelkamp, Max Gath, Christoph Greulich, Malte Humann, Otthein Herzog, and Michael Lawo, 'Monte-carlo tree search for logistics', in *Commercial Transport*, 427–440, Springer, (2016).

[23] Stefan Edelkamp, Max Gath, and Moritz Rohde, 'Monte-carlo tree search for 3d packing with object orientation', in *KI 2014: Advances in Artificial Intelligence*, 285–296, Springer International Publishing, (2014).

[24] Stefan Edelkamp and Christoph Greulich, 'Solving physical traveling salesman problems with policy adaptation', in *Computational Intelligence and Games (CIG), 2014 IEEE Conference on*, pp. 1–8. IEEE, (2014).

[25] Stefan Edelkamp and Zhihao Tang, 'Monte-carlo tree search for the multiple sequence alignment problem', in *Eighth Annual Symposium on Combinatorial Search*, (2015).

[26] M. Fisher and R. Jaikumar, 'A generalized assignment heuristic for ve-

hicle routing', *Networks*, **11**(2), 109–124, (1981).

[27] Tobias Graf and Marco Platzner, 'Adaptive playouts in monte-carlo tree search with policy-gradient reinforcement learning', in *Advances in Computer Games - 14th International Conference, ACG 2015, Leiden, The Netherlands, July 1-3, 2015, Revised Selected Papers*, pp. 1–11, (2015).

[28] G. Gutierrez-Jarpa, G. Desaulniers, G. Laporte, and Marianov V., 'A branch-and-price algorithm for the vehicle routing problem with deliveries, selective pickups and time windows', *European Journal of Operational Research*, **206**(12), 341–349, (2010).

[29] M. Barkaoui J. Berger and O. Bräysy, 'A parallel hybrid genetic algorithm for the vehicle routing problem with time windows', in *Working paper, Defense Research Establishment Valcartier, Canada*, (2001).

[30] Jacek Mańdziuk and Cezary Nejman, 'Uct-based approach to capacitated vehicle routing problem', in *International Conference on Artificial Intelligence and Soft Computing*, pp. 679–690. Springer, (2015).

[31] Y. Marinakis, G-R. Iordanidou, and M. Marinaki, 'Particle swarm optimization for the vehicle routing problem with stochastic demands', *Applied Soft Computing*, **13**, 1693–1704, (2013).

[32] Jean Méhat and Tristan Cazenave, 'Combining UCT and Nested Monte Carlo Search for single-player general game playing', *IEEE Transactions on Computational Intelligence and AI in Games*, **2**(4), 271–277, (2010).

[33] Fernando Portela, 'An unexpectedly effective monte carlo technique for the rna inverse folding problem', *bioRxiv*, 345587, (2018).

[34] J.-Y. Potvin and S. Bengio, 'The vehicule routing problem with time windows part ii : Genetic search', *INFORMS Journal on Computing*, **8**(2), 165–172, (1996).

[35] J.-Y. Potvin, T. Kervahut, B.-L. Garcia, and J.-M. Rousseau, 'The vehicule routing problem with time windows part i : Tabu search', *INFORMS Journal on Computing*, **8**(2), 158–164, (1996).

[36] Simon M. Poulding and Robert Feldt, 'Generating structured test data with specific properties using nested monte-carlo search', in *Genetic and Evolutionary Computation Conference, GECCO '14, Vancouver, BC, Canada, July 12-16, 2014*, pp. 1279–1286, (2014).

[37] Simon M. Poulding and Robert Feldt, 'Heuristic model checking using a monte-carlo tree search algorithm', in *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2015, Madrid, Spain, July 11-15, 2015*, pp. 1359–1366, (2015).

[38] Arpad Rimmel, Fabien Teytaud, and Tristan Cazenave, 'Optimization of the Nested Monte-Carlo algorithm on the traveling salesman problem with time windows', in *Applications of Evolutionary Computation - EvoApplications 2011: EvoCOMNET, EvoFIN, EvoHOT, EvoMUSART, EvoSTIM, and EvoTRANSLOG, Torino, Italy, April 27-29, 2011, Proceedings, Part II*, volume 6625 of *Lecture Notes in Computer Science*, pp. 501–510. Springer, (2011).

[39] Y. Rochat and E.D. Taillard, 'Probabilistic diversification and intensification in local search for vehicle routing', in *Journal of Heuristics,1*, pp. 147–167, (1995).

[40] Christopher D. Rosin, 'Nested rollout policy adaptation for Monte Carlo Tree Search', in *IJCAI*, pp. 649–654, (2011).

[41] Solomon, 'Algorithms for the vehicle routing and scheduling problems with time window constraints', in *Operations Research*, (1985).

[42] Kenneth Sörensen and Marc Sevaux, 'A practical approach for robust and flexible vehicle routing using metaheuristics and monte carlo sampling', *Journal of Mathematical Modelling and Algorithms*, **8**(4), 387, (2009).

[43] S.R. Tangiah, K.E. Nygard, and Juell P.L., 'Gideon : A genetic algorithm system for vehicule routing with time window', in *IEEE CAIA 1991 - Proceedings of the 7th IEEE Conference on Artificial Intelligence Applications, 24-28 February 1991, Miami beach, Florida, USA*, pp. 322–328, (1991).

[44] N. Yuichi and B. Olli, 'A powerful route minimization heuristic for the vehicle routing problem with time windows', in *Operations Research Letters, Vol. 37, No. 5*, pp. 333–338, (2009).