An Efficient Metaheuristic Approach for the Multi-Period Technician Routing and Scheduling Problem

Abouliakdane Khattara
Electrical Engineering Department,
Ferhat Abbas Setif 1 University,
Setif, Algeria
yekdane87@univ-setif.dz

Wahiba Ramdane Cherif-Khettaf
LORIA, UMR 7503,
University of Lorraine,
Nancy, France
ramdanec@loria.fr

Mohammed Mostefai
Electrical Engineering Department,
Ferhat Abbes Setif 1 University,
Setif, Algeria
mostefai@univ-setif.dz

***Abstract*—In this paper, we address a new variant of the Multi-Period Technician Routing and Scheduling Problem. This problem is motivated by a real-life industrial application in a telecommunication company. It is defined by a set of technicians having distinct skills that could perform a set of geographically scattered tasks over a multi-period horizon. Each task is subject to time constraints and must be done at most once over the horizon by one compatible technician. The objective is to minimize the total working time (composed by routing time, service time, and waiting time), the total cost engendered by the rejected tasks, and the total delay. Two variants of variable neighborhood descent are proposed, and three variants of variable neighborhood search to solve this problem. Computational experiments are conducted on benchmark instances from the literature. An analysis of the performance of the proposed local search procedures is given. The results show that our methods outperform the results of a mimetic method published in the literature.**

Keywords-technician routing and scheduling problem (TRSP); variable neighborhood search (VNS); variable neighborhood descent (VND)

I. INTRODUCTION

Technician routing and scheduling problem (TRSP) is a new challenge in logistics for the service sector and especially for utility companies in energy ([illegible] electricity), telecommunications, and water distribution [1]. The TRSP consists of planning tasks assigned to commercial or technical personnel, over a set of periods (days) in order to visit industrial facilities or customers for different types of activities: installation, inspection, repair and maintenance. Until recently, the TRSPs, both static and dynamic cases, have received a limited attention. Thus, the number of publications and scientific reports is limited, although several variants of the TRSP have been studied in the literature. These variants can be divided into two classes: (i) one period TRSP, and (ii) multi-period TRSP. The one period TRSP has been studied by authors who considered constraints as skills, time windows, tools, spare parts, stochastic service and stochastic travel times, multiple depots, and customer priority [1-6]. For the multi-period TRSP, we can mention [illegible], which introduces the multi-period technician scheduling problem with experience based on service times and stochastic customers. The aim is to minimize the expected sum of each day's total service times over a finite horizon. Another multi-period TRSP was proposed in 2007 [8]. This problem consists in computing a schedule for technicians to perform a set of tasks on a five day horizon. The routing aspect is not considered, and tasks have different proficiency skill level constraints, that require a team of technicians. Authors in [illegible] studied the one-periodic variant of this problem, named the service technician routing and scheduling problem by taking on consideration the routing aspect. Authors [illegible] studied a multi-period technician routing problem faced by a water distribution and treatment company. In [9], requests were divided into two categories (users requested interventions and company scheduled visits), and the skill constraints were not included.

In this paper, we propose the study of a new multi-period TRSP variant where skill constraints and routing aspects are considered simultaneously, inspired by a realistic application in the telecommunication field. From the above survey, it appears that most papers on TRSP considered several realistic constraints, but to the best of our knowledge, the multi-periodic variant of TRSP with skill constraints and routing aspects has not been considered in the literature. The papers that consider a multi-periodic TRSP with skill constraints, and routing aspects, included other specific constraints as the technician team constraint [10]. Our study is also an extension of the problem studied in [9, 11 ,12] in which the skill constraints are ignored. As the considered problem is NP-hard and since it results from the combination of complex constraints, large instances can hardly be solved by exact methods. So, the best way to tackle this problem is by using the metaheuristic approaches. We choose a variable neighborhood search (VNS) to solve our problem, because its effectiveness has been proven on a number of variants of vehicle routing problems (VRP) as the vehicle routing problem with time windows [13], the vehicle routing problem with multiple depots and time windows [14, 15], the periodic vehicle routing problem [16], and the workforce scheduling and routing problem [17].

In this paper, we propose two variants of variable neighborhood descent, as well as three variants of variable neighborhood search to solve the TRSP with skill constraints and routing aspects.

Corresponding author: Abouliakdane Khattara

II. PROBLEM DESCRIPTION

We consider a multi-period horizon H of several days (typically one week). For each day $h \in H$, a set of technicians K with different skills are available (a technician has one skill or more). Each technician $k \in K$ has a known starting and ending location $d \in D$, which corresponds to the technician's home or office (the starting location is the same with the ending location). Each technician has a working time limit per day *Maxtimek,h*. Requests belong to two categories: non-urgent tasks (*NT*) generated by the company, and urgent requests (*UT*) formulated by customers through a call center, for emergency reasons. Note that $UT \cup NT = T$, with T the set of all tasks known in advance. Let *si* be the service time of the task i. The urgent tasks $i \in UT$ are planned on a fixed day *hi* and are subjected to customer appointments within a given time window (*bi*, *ei*), where *bi* is the beginning of the time window, and *ei* the end of the time window. The task i could be affected to a technician k if the arriving time at task i (denoted *aik*) is before the end of the time windows (*aik*<*ei*). If $\forall$ $k \in K$, *aik*≥*ei*, then the task i is considered as a rejected task. If *aik*≤*bi*, a waiting time *Wik* occurs, with *Wik*=*bi*-*aik*. If *aik*≤*ei*, and *aik*+*si*>*ei*, a delay *Lik* occurs, with *Lik*=(*aik*+*si*)-*ei*. Non-urgent tasks are characterized by a validity period composed of one or several days [*hbi*, *hei*]∈H, where *hbi* is the early day and *hei* is the deadline for the execution of i. A request i requires certain skills (qualifications), and must be executed by only one compatible technician. The goal is to build a set of routes per day and per technician (at most |*Kh*| routes per day). Eac[illegible] *Rhk* is a sequence of tasks assigned to only one techn[illegible] k and one-day h. The following constraints must be satisfie[illegible] each task must be executed at most once within the validi[illegible] period or within the time window, 2) the total time of each route *Rhk* should not exceed *Maxtimek,h*, 3) t[illegible]etence requirements must be respected, 4) each rout[illegible]ust [illegible]t and end at the same location $d \in D$. The o[illegible]ctive [illegible] measured in monetary units denoted [illegible] c[illegible] in minimizing three costs: (i) the total working ti[illegible]posed by routing time that depends on the number [illegible] kilomet[illegible]elled by each technician, the service time a[illegible] the waiting ti[illegible] (ii) the total cost engendered by the eject[illegible]ks, and (iii) the total delay.

III. SOLUTION METHODOLOGY

In this section, we describ[illegible]e general framework of the variable neighborhood searc[illegible]VNS) [illegible]d then we present the basic components of the V[illegible] that [illegible] have de[illegible]loped to solve our problem.

A. Variable Neighborhood Sear[illegible]

VNS is a metahe[illegible]r a frame[illegible]k created 1997 [18, 19] for approximate[illegible]olvi[illegible] optimization problems, including combinatorial and [illegible]n-linea[illegible] optimization problems [20, 21]. VNS [illegible] base[illegible] systematic changes of neighborhood stru[illegible]ring the search for a (near) optimal solution of a considere[illegible]lem. These changes occur in both descent phase, to improv[illegible] solution, and shaking and perturbation phase that aims to escape local optima traps. The main structure of the VNS (Algorithm 1) is shown in Figure 1.

```
Function VNS_heuristic (x,k_max,t_max)
1. repeat
2.    repeat
3.       x' ← Shake (x,k);
4.       x" ← Improvement procedure (x');
5.       Neighbor[illegible]hange(x,x",k);
6.    until k = k[illegible]
7.    t ← Cpu[illegible]();
8. Until t > [illegible]
```

Fig. 1. Algorit[illegible]ariable neighbo[illegible]od search VNS

The inputs of V[illegible]S heuris[illegible] $_{max}$ and t_{max}, and they present the initial [illegible]ution, the nu[illegible] of neighborhoods to be explored and th[illegible] maxi[illegible]n allowed CPU time. The main ingredients o[illegible]ariab[illegible] neighb[illegible]hood search include an improvement [illegible]du[illegible]o impr[illegible]e the current solution and a shaking procedu[illegible]erturb [illegible] search and escape from the correspon[illegible]g valley, [illegible] 3 and 4. The improvement procedu[illegible] line 5 could [illegible] single local search or an ordered list of s[illegible]eighborhoods.

B. L[illegible]al Solu[illegible]

We propose to u[illegible]s an initial solution the best insertion m[illegible] with sorting list. This method is performed by two [illegible]ps. In the first step, a list of unserved tasks (L, L=T), is [illegible]orted in inc[illegible]sing order according to validity day (*VD*), that [illegible]epresents the [illegible]ngth of the period (number of days) in which [illegible] tasks can[illegible] done, $VD_i = he_i - hb_i$. In the second step, the [illegible]t a task i from the head of L, and scan all feas[illegible]rtions in all routes R_{hk}. The insertion cost of i in a route R_{hk} between two tasks x and y, named $\delta(i, R_{hk}, x, y)$ will be [illegible] as in (1). The algorithm performs the best insertion.

$$\begin{aligned} (i, Rhk, x, y) &= C_{xi} + C_{iy} - C_{xy} + \Sigma\left(j \in Rhk \cup i\right) W_{jk} \\ &+ \Sigma\left(j \in Rhk \cup i\right) L_{jk} \end{aligned} \tag{1}$$

[illegible] Local Search Procedures

We propose five local search operators to be used either individually or together to focus the search in the inner loop of VNS. We consider three intra-route and two inter-route local search methods. The best improvement strategy is used for each method. The local search methods are:

- One intra-route relocate: one node (task) from the route is removed and reinserted in other positions in the same route.
- One intra-route exchange: two nodes (tasks) are exchanged in the same route.
- 2 opt: two arcs are removed and reinserted in the same route
- One inter-route relocate: one node (task) from the route is removed and reinserted in one other route in the solution.
- One inter-route exchange: two nodes (tasks) are exchanged between two different routes.

D. Variable Neighborhood Descent Procedures

The variable neighborhood descent (VND) procedures explore several neighborhood structures either in a sequential or nested (or composite) fashion to possibly improve a given solution [21] because the solution which is a local optimum with respect to several neighborhood structures is more likely to be a global optimum than the solution generated as a local optimum for just one neighborhood structure. The order of neighborhoods may play an important role in the quality of the final solution [22]. Two variants of VND are discussed in this paper regarding the decision made in neighborhood change procedure. If an improvement has been detected in some neighborhood: (1) Basic VND (B-VND): we return to the first neighborhood on the list, (2) Union VND (U-VND): at each iteration all the neighborhoods in the list are used to explore the search, and the next incumbent solution is the best one found by the best neighborhood. The outline of basic VND is presented in Algorithm 2 (Figure 2). The steps of the sequential neighborhood change which is presented in line 5 (Algorithm 1) and line 7 (Algorithm 2) are given in Algorithm 3 (Figure 3). If an improvement of the incumbent solution in some neighborhood structure occurs, the search is resumed in the first neighborhood structure (according to the defined order) of the new incumbent solution, otherwise the search is continued in the next neighborhood (according to the defined order).

```
Function B-VND (x,k_max)
1.  repeat
2.  |  stop ← false;
3.  |  k ← 1;
4.  |  x' ← x;
5.  |  repeat
6.  |  |  x" ← argmin_{y∈Nk(x)} f(y);
7.  |  |  Neighborhood_Change(x,x",k);
8.  |  until k = k_max;
9.  |  If f(x') ≤ f(x) then
10. |  |  stop ← true;
11. |  end
12. until stop = true ;
13. return x'
```

Fig. 2. Algorithm 2: Variable neighborhood descent VND

```
Function Neighborhood_Change (x,x',k )
1.  If f(x') < f(x) then
2.  |  x ← x';
3.  |  k ← 1 ;
4.  else
5.  |  k ← k+1;
6.  end
```

Fig. 3. Algorithm 3: Neighborhoo[illegible]nge procedu[illegible]

E. Shaking Procedure

The shaking procedure is used [illegible] VNS a[illegible]oned in line 3 of Algorithm 1 in order to hopefully resolve [illegible] minima traps. Our shaking procedure [illegible]sists in selecting a random solution from the k^{th} neighbor[illegible]d str[illegible]ure, $N_k(x)$.

IV. COM[illegible]ATIO[illegible] RESU[illegible]

All developed algorithm[illegible]mplem[illegible]d in Matlab and all tests were carried out on a [illegible]k[illegible]o Intel Core™ i7-3520M with 2.90GH[illegible] sharing [illegible]emory of 8GB (the algorithms use only [illegible]e C[illegible]. Our problem is an extension of the problem studie[illegible] [9, 1[illegible]2] i[illegible]ch the skill constraints are ignored. For[illegible], we [illegible] instance used in [9, 11, 12], and evaluat[illegible]rmance of our methods with their mimetic algorithm. [illegible]t compare the performance of different local search pr[illegible]s with the initial solution. Different VND procedures a[illegible] then tested and compared. Finally, we compare and evaluate the VNS procedures proposed in this paper.

A. Description of Experimental Data Sets

In order to evaluate a[illegible] the performance of the proposed approaches, w[illegible]ompa[illegible]m with the methods proposed by Tricoire in [illegible]11, 12]. The[illegible]tances of [9, 11, 12] are used for tests. For [illegible] the skill con[illegible]ints in our problem are relaxed, and lunch [illegible]onstrain[illegible]re added. They are inspired by a real li[illegible] case, an[illegible]vailable with detailed experimental res[illegible] in [9] as [illegible]as on the web site: http://www. em[illegible]/z-aut[illegible]outing-pbs/. All instances have a five-day planni[illegible]horiz[illegible]nd thr[illegible] technicians available every day. The den[illegible]re[illegible]domly [illegible]stributed over a $40km^2$ map, and Euclidean d[illegible] are [illegible]. The technicians drive at a constant a[illegible]age spe[illegible]/h. Two sizes of instances are tested C[illegible]ith 100 custom[illegible]nd C2 with 180 customers, each one w[illegible]variants according to the distribution and the perce[illegible]ge [illegible] windows and the percentage of urgent and non-[illegible]gent tasks.

B. E[illegible]luation Of The Performance Of The Local Search [illegible]ocedures

We stud[illegible]e impact of the locale search procedures. The [illegible]esults are sh[illegible]n in Table I. The first column indicates the [illegible]me of each [illegible]stance. Column 2 presents the result found by [illegible]itial [illegible]tion, which is based on the best insertion strateg[illegible]e remained columns provide both the Gap and the computing time for each local search operator. The Gap is [illegible] by (2). Row 13 mentions the average results of all [illegible]. The ranks of the local search operators according to [illegible]eir performances and computing time are provided in the last [illegible]wo rows. The best results are in bold.

$$\text{Gap\%} = \left(f(x)_{heuristic} - f(x)_{heuristic+LS} \right) / f(x)_{heuristic+LS} \quad (2)$$

From Table I, we note that the 2 opt operator is the best one in almost all instances but it is the third one in terms of computing time. It is also worth noting that all the operators perform well and they all improve on average at least 7.14% and at most 9.66% the results found by the heuristic.

C. Variable Neighborhood Descent Procedures

The aim of this section is to evaluate and to compare different variants of VND procedures according to the manner in which the neighborhood is changed after each improvement occurs. Namely, it is obvious that the order of neighborhoods on the list affect the performance of VND procedures [22]. Thus, we take into account two possible orders of LS procedures according to their performance: 1) the value of $f(x)$, and 2) the computing time. The used orders are mentioned in Table II. The results of VND procedures on Tricoire instances are summarized in Table III. The settings of the VND variant are provided in column and row headings as described above. For example, in Table III, the values in the two cells, at the intersection of the row C100_1 and 4^{th} column, correspond to value achieved by B-VND that explores neighborhoods using the 1st order, as well as its execution time in second. The next column reports the percentage deviation of the obtained solution compared to best solution of the mimetic algorithm proposed by Tricoire [9, 11,12]. The deviations are calculated by:

$$Dev\% = \left(f(x)_{VND} - f(x)_{memetic} \right) / f(x)_{memetic} \qquad (3)$$

The next column reports the percentage deviation of the obtained solution compared to the result found by the initial solution, which is calculated by (2). In Table III, we report the results obtained by B-VND and U-VND using the two proposed orders. The average results are mentioned in the two last rows. In Table III, values in bold followed by a star represent new best solution[illegible] by our method.

TABLE I. COMPARISON BETWEEN LOCAL SEARCH PROCE[illegible]

		2 Opt		One intra route relocate		One intra route exchange		One inter rou[illegible]ge		[illegible]nter route relocate	
instances	***f(x)*** **Heuristic**	**Gap**	**Time (s)**	**Gap**	**Time (s)**	**Gap**	**Time (s)**	[illegible]	[illegible]	[illegible]	**Time (s)**
C1_1	21024.64	14.04%	0.40	12.74%	0.47	12.79%	0.47	[illegible]	0.29	[illegible]	5.79
C1_2	19347.33	6.15%	0.40	4.63%	0.45	4.65%	0.41	[illegible]	5.49	5.30%	4.31
C1_3	19658.79	8.68%	0.39	6.55%	0.43	5.74%	0.36	[illegible]	[illegible]	4.29%	3.45
C1_4	22232.21	9.86%	0.40	7.29%	0.29	7.32%	0.35	[illegible]	[illegible]	6.10%	2.96
C1_5	18219.65	-2.63%	0.19	-2.75%	0.28	-1.84%	0.24	[illegible]	[illegible]	8.58%	7.05
C2_1	39085.88	10.03%	1.76	7.04%	1.45	7.07%	[illegible]	[illegible]	[illegible]	8.27%	30.95
C2_2	34873.74	12.22%	1.76	6.69%	1.62	5.89%	[illegible]	4.29%	[illegible]	6.47%	20.96
C2_3	36349.52	13.70%	1.70	13.22%	1.75	11.36%	[illegible]	13.09%	[illegible]	15.12%	24.57
C2_4	36679.56	8.59%	1.46	7.32%	1.95	6.00%	[illegible]	7.72%	29.95	7.97%	25.62
C2_5	33700.93	11.00%	1.74	8.42%	1.27	9.58%	[illegible]	[illegible]	28.76	6.23%	19.96
Avg	28117.23	9.66%	1.02	7.47%	0.99	7.14%	0.98	[illegible]	17.95	7.58%	14.56
Rank (*f(x)*)		1		3		5		4		2	
Rank (Time)		3		2		[illegible]		5		4	

TABLE II. ORDERS OF LOCAL SEARCH PROCEDURES

5 Local search operators		**1st**	**2nd**
Order of local search	2 Opt	1	3
	One intra route relocate	3	2
	One Intra route exchange	5	1
	One Inter route exchange	4	5
	One inter route relocate	2	4

From the results presented in Table III, we may concl[illegible] the following: The VND variants that explore neighborhoods i[illegible] 1st order offer the best results in both objective [illegible]ion and CPU time co[illegible]red to the other order type. If we consider the [illegible]verage resul[illegible]ver all test instances, it appears that the best [illegible]ge resu[illegible]are obtained by U-VND even when we change the[illegible]eighborhoods. So we can say that the U-VND is more effective than B-VND in terms of the objective function [illegible]nd CPU Time. From the average results over all test instances [illegible] that all our VND procedures implemented and [illegible]cussed in this paper are competitive and perform better than [illegible]e mimetic algorithm when solving the same problem. For 6 instances among 10, a new best solution is found by our [illegible]ethod.

TA[illegible] III. [illegible]F DIFFERENT VARIANTS OF VND

			[illegible]						**U-VND**					
		Orders	**1st**			**2nd**			**1st**			**2nd**		
Instances		**Mimetic of Tricoire**	**Value**	[illegible]	**%Dev heuristic**	[illegible]	**% Dev mimetic**	**% Dev heuristic**	**Value**	**% Dev mimetic**	**% Dev heuristic**	**Value**	**% Dev mimetic**	**% Dev heuristic**
C100_1	*f(x)*	17893.91	**17578.37***	[illegible]	19.61%	17594.18	-1.68%	19.50%	**17578.37***	-1.76%	19.61%	17594.03	-1.68%	19.50%
	Time (s)		9.92			11.42			8.97			13.28		
C100_2	*f(x)*	15977.12	17202.92	[illegible]	[illegible]	17136.03	7.25%	12.90%	17153.67	7.36%	12.79%	17164.61	7.43%	12.72%
	Time (s)		9.52			[illegible]			8.06			8.73		
C100_3	*f(x)*	16714.03	[illegible]	4.66%	12.38%	[illegible]	4.88%	12.15%	17491.94	4.65%	12.39%	17538.11	4.93%	12.09%
	Time (s)		7.32			5.93			5.03			6.74		
C100_4	*f(x)*	17489.36	[illegible]	[illegible]	21.58%	18265.73	4.44%	21.72%	18229.85	4.23%	21.95%	18031.33	3.10%	23.30%
	Time (s)		[illegible]			12.68			9.5			14.03		
C100_5	*f(x)*	16025.91	[illegible]	[illegible]	[illegible]	16611.1	3.65%	9.68%	16535.47	3.18%	10.19%	16364.41	2.11%	11.34%
	Time (s)		[illegible]			9.78			9.91			15.47		
C180_1	*f(x)*	28945.36	28607.43	[illegible]	[illegible]	29299.56	1.22%	33.40%	**28405.93***	-1.86%	37.60%	28579.51	-1.26%	36.76%
	Time (s)		[illegible]			107.17			84.19			96.87		
C180_2	*f(x)*	[illegible]	[illegible]	-9.73%	23.86%	27780.88	-10.93%	25.53%	27748.3	-11.04%	25.68%	**27729.19***	-11.10%	25.77%
	Time (s)		[illegible]			72.82			58.22			46.09		
C180_3	*f(x)*	[illegible]	[illegible]	[illegible]	37,35%	27472,29	-0,92%	32,31%	**26034,96***	-6,11%	39,62%	26886,39	-3,04%	35,20%
	Time (s)		[illegible]			89,47			78,46			77,85		
C180_4	*f(x)*	[illegible]	[illegible]	-2,96%	24,98%	29522,29	-2,39%	24,24%	30124,57	-0,40%	21,76%	**29238,94***	-3,33%	25,45%
	Time (s)		[illegible]			76,22			52,22			61,71		
C180_5	*f(x)*	28158,57	[illegible]	-4,54%	25,37%	26566,25	-5,65%	26,86%	**26395,74***	-6,26%	27,68%	26625	-5,45%	26,58%
	Time (s)		78,74			93,93			70,71			60,92		
Average	*f(x)*	23036,94	22655,33	-1,66%	24,11%	22777,77	-1,13%	23,44%	**22569,88**	**-2,03%**	24,58%	22575,15	-2,00%	24,55%
	Time (s)		44,96			48.9			**38.53**			40.17		

It is worth noting that all VND procedures also perform better and they all improve on average at least 23.44% and at most 24.58% the results found by the heuristic. That means that VND procedures improve the solution in average 15% more than the single local search operators (Table I).

D. Variable Neighborhood Search Procedures

In this section we evaluate and compare three variants of VNS procedures regarding to the improvement procedures in the inner loop: (1) B-VNS that uses a simple local search in the inner loop at each iteration and move to the other one as in Algorithms 1 and 3, (2) VNS_B-VND that uses a B-VND procedure, and (3) VNS_U-VND that uses a U-VND in the improvement phase. The neighborhoods in the VND procedures are ordered in [illegible]er. The performances of VNS procedures have bee[illegible]ested [illegible]s C1 of the instances described above. We te[illegible]d our VNS [illegible]cedures by using 4 different time limits, [illegible]ing from 36[illegible]o 1140s. For each instance and each VNS [illegible] the A[illegible]rithm 5 times. The results are summari[illegible]d in Tab[illegible] [illegible]gure 4.

TABLE IV. EVALUATION OF DIFFERENT VARIA[illegible] OF VN[illegible]

			BVNS		[illegible]B-V[illegible]		VNS_U-VND	
Time limit (s)	**Instances**	**Mimetic of Tricoire**	**% Dev_Best**	**% Dev_Avg**	**% Dev_[illegible]**	**[illegible]ev_Avg**	**[illegible]Dev_Best**	**% Dev_Avg**
360	Average C1	16820.07	-0.27%	1.56%	[illegible]4%	[illegible]%	-1.25%	0.14%
720			-0.56%	1.06%	[illegible]9%	[illegible]	-1.87%	-0.27%
1080			-0.56%	0.99%	[illegible].69%	-0.53[illegible]	-2.11%	-0.47%
1440			-0.57%	0.89%	[illegible]	-0.66%	-2.19%	-0.60%
Average			-0.49%	1.12%	-[illegible]	-0.34%	-1.85%	-0.30%

For each time limit and each VNS variant, we report the deviation from the best solution found over all variants of instance of class C1 in 5 runs compared to the best solution found by the mimetic algorithm of [9, 11, 12] (named % Dev_Best in Table IV). We also compute the deviation from the average value of the solutions found in 5 runs for all instances of C1 compared to the best solution found by the mimetic algorithm (named % Dev_Avg in Table IV). The deviations are calculated by (3).

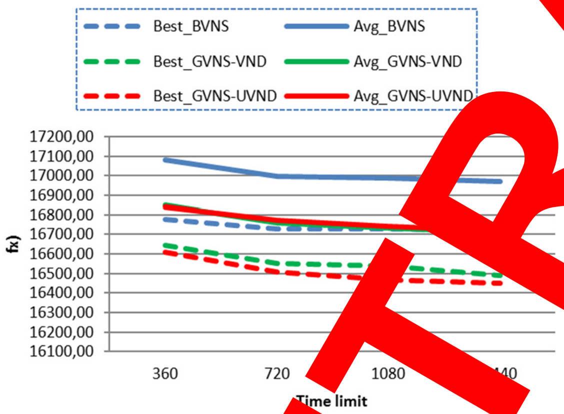

Fig. 4. Comparison[illegible]ween VNS procedures

From the results prese[illegible] in T[illegible]e IV [illegible]d Figure 4 we may draw the following c[illegible]ion[illegible]

Firstly, we remark that the p[illegible]anc[illegible] VNS procedures in this paper depends o[illegible] time li[illegible]e have a long time we achieve the best[illegible]ll VNS v[illegible]ants outperform the results of the mim[illegible] algo[illegible]m even if our VNS methods are stopped at 360[illegible] The [illegible] of all local search procedures is n[illegible]al than only the use of one local search in the inner [illegible] the VNS procedure. If we consider the average results over [illegible]nts of instance C1, it appears that the best average results [illegible] obtunded by VNS_U-VND, and that confirms what we found in the last section.

V. C[illegible]LUSION AND PERSPECTIVES

[illegible]this paper, we considered a new variant of the Multi-[illegible]eriod Techn[illegible]ian Routing and Scheduling Problem motivated by a real-li[illegible]dustrial application in a Telecommunication Company. T[illegible]olve the problem, two variants of variable [illegible]hborhoo[illegible]escent B-VND and U-VND, as well as three va[illegible]iable neighborhood search BVNS, VNS_B-VND and VNS_U-VND are proposed. All heuristic methods were [illegible]ted and compared with the methods proposed by Tricoire [9, [illegible]he results confirm the effectiveness of our methods. [illegible]garding future work, we will generate other instances to [illegible]tensify the experimentations. Also, we will consider the dynamic aspect, where the demands appear dynamically over [illegible]e planning horizon.

REFERENCES

[1] V. Pillac, C. Gueret, A. L. Medaglia, "A parallel matheuristic for the technician routing and scheduling problem", Optimization Letters, Vol. 7, No. 7, pp. 1525-1535, 2013

[2] J. Xu, S. Y. Chiu, "Effective heuristic procedures for a field technician scheduling problem", Journal of Heuristics, Vol. 7, No. 5, pp. 495-509, 2001

[3] E. Hadjiconstantinou, D. Roberts, "Routing under uncertainty: an application in the scheduling of field service engineers", in: The vehicle Routing Problem, pp. 331-352, Society for Industrial and Applied Mathematics, 2001

[4] E. Delage, Re-optimization of Technician Tours in Dynamic Environments with Stochastic Service Time, Ecole des Mines de Nantes, 2010

[5] C. E. Cortes, F. Ordonez, S. Souyris, A. Weintraub, "Routing technicians under stochastic service times: a robust optimization approach", TRISTAN VI: The Sixth Triennial Symposium on Transportation Analysis, Phuket Island, Thailand, June 10-15, 2007

[6] V. Pillac, C. Gueret, A. Medaglia, "On the dynamic technician routing and scheduling problem", 5th International Workshop on Freight Transportation and Logistics, Mykonos, Greece, May 21-25, 2012

[7] X. Chen, B. W. Thomas, M. Hewitt, "Multi-Period Technician Scheduling with Experience-based Service Times and Stochastic Customers", Computers and Operations Research, Vol. 82, No. C, pp. 1-14, 2017

[8] W. Jaskowski, ROADEF Challenge 2007: Technicians and Interventions Scheduling for Telecommunications, Poznan University of Technology, 2007

[9] F. Tricoire, Optimisation de Tournees de Vehicules et de Personnels de Maintenance: Application a la Distribution et au Traitement des Eaux, PhD Thesis, Universite de Nantes, 2006 (in French)

[10] A. A. Kovacs, S. N. Parragh, K. F. Doerner, R. F. Hartl, "Adaptive large neighborhood search for service technician routing and scheduling problems", Journal of Scheduling, Vol. 15, No. 5, pp. 579-600, 2012

[11] N. Bostel, P. Dejax, P. Guez, F. Tricoire, "Multiperiod planning and routing on a rolling horizon for field force optimization logistics", in: The Vehicle Routing Problem: Latest Advances and New Challenges, pp. 503-525, Springer, 2008

[12] F. Tricoire, N. Bostel, P. Dejax, P. Guez, "Exact and hybrid methods for the multiperiod field service routing problem", Central European Journal of Operations Research, Vol. 21, No. 2, pp. 359-377, 2013

[13] O. Braysy, "A reactive variable neighborhood search for the vehicle routing problem with time windows", INFORMS Journal of Computing, Vol. 15, No. 4, pp. 347–368, 2003

[14] M. Polacek, R. F. Hartl, K. Doerner, M. Reimann, "A variable neighborhood search for the multi depot vehicle routing problem with time windows", Journal of Heuristics, Vol. 10, No. 6, pp. 613-627, 2004

[15] S. Salhi, A. Imran, N. A. Wassan, "The multi-depot vehicle routing problem with heterogeneous vehicle fleet: Formulation and a variable neighborhood search implementation", Computers & Operations Research, Vol. 52B, pp. 315-325, 2014

[16] M. Elbek, S. Wohlk, "A variable neighborhood search for the multi-period collection of recyclable materials", European Journal of Operational Research, Vol. 249, No. 2, pp. 540-550, 2016

[17] R. L. Pinheiro, D. Landa-Silva, J. Atkin, "A variable neighbourhood search for the workforce scheduling and routing problem", in: Advances in Nature and Biologically Inspired Computing, pp. 247-259, Springer, 2016

[18] N. Mladenovic, P. Hansen, "Variable neighborhood search", Computers & Operations Research, Vol. 24, No. 11, pp. 1097-1100, 1997

[19] P. Hansen, N. Mladenovic, "Variable neighborhood search: P[illegible] and applications", European Journal of Operational Research, V[illegible]30, No. 3, pp. 449-467, 2001

[20] P. Hansen, N. Mladenovic, J. A. M. Perez, "Variable neighbourh[illegible] search: methods and applications", Annals of Operations Research, Vo[illegible] 175, No. 1, pp. 367-407, 2010

[21] P. Hansen, N. Mladenovic, R. Todosijevic, S. [illegible]ariable neighborhood search: basics and variants", [illegible]O Jo[illegible]al on Computational Optimization, Vol. 5, No. 3, pp. 1-[illegible]016

[22] A. Mjirda, R. Todosijevic, S. Hanafi, P. H[illegible] N. [illegible]ic, "Sequential variable neighborhood descent varian[illegible]ical study on the traveling salesman problem", Int[illegible]tional [illegible]tions in Operational Research, Vol. 24, No. 3, pp. 6[illegible]33, 2016