4OR (2006) 4:221–238
DOI 10.1007/s10288-006-0001-9

REGULAR PAPER

Solving the capacitated location-routing problem by a GRASP complemented by a learning process and a path relinking

Christian Prins · Caroline Prodhon · Roberto Wolfler Calvo

Received: 1 December 2004 / Revised: 1 August 2005 /
Published online: 18 July 2006

Abstract As shown in recent researches, the costs in distribution systems may be excessive if routes are ignored when locating depots. The location routing problem (LRP) overcomes this drawback by simultaneously tackling location and routing decisions. This paper presents a new metaheuristic to solve the LRP with capacitated routes and depots. A first phase executes a GRASP, based on an extended and randomized version of Clarke and Wright algorithm. This phase is implemented with a learning process on the choice of depots. In a second phase, new solutions are generated by a post-optimization using a path relinking. The method is evaluated on sets of randomly generated instances, and compared to other heuristics and a lower bound. Solutions are obtained in a reasonable amount of time for such a strategic problem. Furthermore, the algorithm is competitive with a metaheuristic published for the case of uncapacitated depots.

Keywords Heuristic · Location routing problem · GRASP · Path relinking

MSC Classification 90B06

1 Introduction

Logistic systems involve different decision levels, e.g., strategic for depot location or tactical for vehicle routing. Nevertheless, Salhi and Rand (1989) have shown that

C. Prins · C. Prodhon (✉) · R. W. Calvo
Institute Charles Delaunay, University of Technology of Troyes,
BP 2060, 10010 Troyes Cedex, France
e-mail: caroline.prodhon@utt.fr

C. Prins
e-mail: christian.prins@utt.fr

R. W. Calvo
e-mail: roberto.wolfler@utt.fr

tackling each level separately often leads to suboptimization. The location-routing problem (LRP) appeared relatively recently in literature as a combination of both levels. As shown in a survey (Min et al. 1998), most early published papers consider either capacitated routes or capacitated depots, but not both (Laporte et al. 1988; Chien 1993; Srivastava 1993). In general, the LRP is formulated as a deterministic node routing problem (i.e., customers are located on nodes of the network). However, a few authors have studied the stochastic case (Laporte et al. 1989; Chan et al. 2001) and, more recently, arc routing versions (Ghiani and Laporte 2001; Labadi 2003).

Albareda-Sambola et al. (2005) proposed a two-phase Tabu search (TS) heuristic for the LRP with one single route per capacitated open depot. The two phases consist of an intensification that optimizes the routes and a diversification that modifies the set of open depots. The method has been tested on small instances only (at most 30 customers).

Tuzun and Burke (1999) developed another two-phase TS, but for the LRP with capacitated routes and uncapacitated depots, i.e., a depot may have as many routes as desired. The two phases are also dedicated to routing and location. The principle is to iteratively add a depot to the solution until it deteriorates the total cost. These authors report results for up to 200 customers.

Wu et al. (2002) studied the LRP with capacity on both depots and routes, and homogeneous or heterogeneous fleet types with limited number of vehicles. They designed a simulated annealing (SA) algorithm with a tabu list to avoid cycling.

A heuristic for a three-level LRP (factories, depots, customers) with capacitated routes and depots and a maximum duration per route was developed by Bruns and Klose (1995). Their method is iterative and one iteration begins by building clusters of customers fitting vehicle capacity. Then, an estimate of the routing costs is computed for the allocation of the customers to depots. A Lagrangian heuristic obtained by relaxing capacity constraints determines which depots should be opened to minimize routing costs. Then, a routing phase, involving saving criteria, 2-opt moves and exchanges of customers, is executed. Finally, the estimate for the routing costs is refined and the procedure iterates until the estimate is stabilized or a maximum number of iterations is reached. The heuristic is evaluated on small instances with 2 plants, 10 potential depots and 20 customers.

This paper deals with an LRP with capacitated depots, capacitated routes, and fixed costs to open a depot or a route. The objective is to determine the set of depots to open and the routes originating from each open depot, in order to minimize a total cost comprising the setup costs of depots and routes and the total cost of the routes. A metaheuristic combining greedy randomized adaptive search procedure (GRASP) and a learning process, and finishing by a path relinking is proposed for this problem.

The paper is organized as follows. Section 2 defines the problem, introduces some required notations and proposes an integer linear programming model. Section 3 describes the main components of the GRASP. The post-optimization by a path relinking and the overall structure of the method is summarized in Sect. 4. Computational experiments are presented in Sect. 5. Some concluding remarks close the paper.

2 Problem definition and linear model

The data for the LRP studied in this paper are based on a weighted and directed graph $G = (V, A, C)$. V is a set of nodes comprising a subset I of m possible depot locations and a subset $J = V \backslash I$ of n customers. The traversal cost of any arc $a = (i, j)$ in the arc set A is given by C_a. A capacity W_i and an opening cost O_i are associated with each depot site $i \in I$. Each customer $j \in J$ has a demand d_j. A set K of identical vehicles of capacity Q is available. When used, each vehicle incurs a fixed cost F and performs one single route. The total number of vehicles used (or routes performed) is a decision variable.

The following constraints must hold:

- Each demand d_j must be served by one single vehicle.
- Each route must begin and end at the same depot and its total load must not exceed vehicle capacity.
- The total load of the routes assigned to a depot must fit the capacity of that depot.

The total cost of a route includes the fixed cost F and the costs of traversed arcs. The objective is to find which depots should be opened and which routes should be constructed, in order to minimize the total cost (fixed costs of depots, plus total cost of the routes).

It is well known that undirected graphs could be encoded as symmetric directed graphs in many algorithms. Hence, our model can handle an undirected LRP by replacing each edge by two opposite arcs, giving a symmetric directed graph G. Euclidean instances like those considered in section 5 correspond to a complete directed graph G in which C_{ij} is the Euclidean distance between nodes i and j.

The problem can be modelled as a zero-one linear program. Let S be a subset of nodes, $\delta^+(S)$ ($\delta^-(S)$) be the set of arcs leaving (entering) S and $L(S)$ the set of arcs with both extremities in S. If S contains only one node x, $\delta^+(x)$ is a simplified form for $\delta^+(\{x\})$. The following boolean variables are used: $y_i = 1$ iff depot i is opened, $f_{ij} = 1$ iff customer j is assigned to depot i, and $x_{ak} = 1$ iff arc a is used in the route performed by the vehicle $k \in K$, and zero otherwise.

Minimize:

$$z = \sum_{i \in I} O_i . y_i + \sum_{a \in A} \sum_{k \in K} C_a . x_{ak} + \sum_{k \in K} \sum_{a \in \delta^+(I)} F . x_{ak}, \tag{1}$$

subject to :

$$\sum_{k \in K} \sum_{a \in \delta^-(j)} x_{ak} = 1 \quad \forall j \in J, \tag{2}$$

$$\sum_{j \in J} \sum_{a \in \delta^-(j)} d_j . x_{ak} \leq Q \quad \forall k \in K, \tag{3}$$

$$\sum_{a\in\delta^+(i)} x_{ak} - \sum_{a\in\delta^-(i)} x_{ak} = 0 \quad \forall k \in K, \quad \forall i \in V, \tag{4}$$

$$\sum_{a\in\delta^+(I)} x_{ak} \leq 1 \quad \forall k \in K, \tag{5}$$

$$\sum_{a\in L(S)} x_{ak} \leq |S| - 1 \quad \forall S \subseteq J, \quad \forall k \in K \tag{6}$$

$$\sum_{a\in\delta^+(i)\cap\delta^-(J)} x_{ak} + \sum_{a\in\delta^-(j)} x_{ak} \leq 1 + f_{ij} \quad \forall i \in I, \quad \forall j \in J, \quad \forall k \in K \tag{7}$$

$$\sum_{j\in J} d_j.f_{ij} \leq W_i.y_i \quad \forall i \in I \tag{8}$$

$$x_{ak} \in \{0,1\} \quad \forall\ a \in A, \quad \forall k \in K \tag{9}$$

$$y_i \in \{0,1\} \quad \forall\ i \in I \tag{10}$$

$$f_{ij} \in \{0,1\} \quad \forall\ i \in I, \quad \forall j \in V \tag{11}$$

The objective function (1) gathers all the costs described before. Constraints (2) guarantee that a customer belongs to one route only and that each customer has only one predecessor. Capacity constraints are satisfied thanks to inequalities (3) and (8). Constraints (4) and (5) ensure the continuity of each route and a return to the depot of origin. Constraints (6) are subtour elimination constraints. Constraints (7) specify that a customer can be assigned to a depot only if a route linking them is opened. Finally, all variables are boolean thanks to (9), (10) and (11).

The LRP is obviously *NP*-hard, since it reduces to the well-known vehicle routing problem (VRP) when $m = 1$. It is much more combinatorial than the VRP, since in addition to the partition of customers into routes and the sequencing of each route, it involves the selection of open depots and the assignment of routes to these depots. Therefore, only very small instances can be solved exactly by commercial LP solvers and basic relaxations like linear ones provide a too weak lower bound. A first non-trivial bound for the LRP with capacitated routes and depots has been proposed only recently by Barreto (2004). It seems to be a good one on small instances but it becomes too time-consuming for large scale instances and anyway, the author reports no gap beyond 50 customers.

The medium and large instances targeted in this paper (up to 20 depot sites and 200 customers) require the development of a metaheuristic, described in the sequel.

3 Components of the GRASP

A GRASP is an iterative procedure and each iteration is composed of two phases (Feo and Resende 1995): construction of a trial solution using a greedy randomized heuristic and improvement of this solution by local search. It may be viewed as a way of

strategically sampling the solution space. Section 3.1 exposes the chosen greedy heuristic and the local search is presented in Sect. 3.2. Then, the added learning process into the GRASP to improve its performances is described in Sect. 3.3.

3.1 Greedy randomized heuristic

In this subsection, the Clarke and Wright saving algorithm (Clarke and Wright 1964) is extended to the LRP and randomized to provide the greedy heuristic required by the GRASP. This algorithm has been selected because it gives excellent solutions for the VRP, when it is followed by local search procedures (Laporte et al. 2000).

3.1.1 Extended Clarke and Wright algorithm

The Clarke and Wright algorithm (CWA in the sequel), as originally designed, starts by building a flower-like trivial solution, in which each customer is visited by one dedicated route. Then, each pair of routes (R, S) whose total load does not exceed Q is inspected to evaluate the saving (equals to the opposite value of cost variation) that would result from their merger (concatenation). The merger giving the largest positive saving is executed. This process is repeated until no capacity-feasible merger with a positive saving can be found.

For instance, if s denotes the unique depot node and i, j, k, l respectively stand for the first and last customers of R and the first and last customers of S, the saving obtained by concatenating S after R is equal to $C_{js} + C_{sk} - C_{jk}$, see upper part of Fig. 1. Note that there exist four possible mergers per pair of routes (R, S), since each route may be inverted or not.

The CWA can be implemented in $O(n^3)$ and, using heapsort (Cormen et al. 1999), in $O(n^2 \log n)$. To achieve the latter complexity, note that a merger can be defined as an arc $[j, k]$ that will link one node j, at one end of a tour, to one node k, at one end of another tour. The list L of the $O(n^2)$ possible mergers can be pre-computed for the initial solution and sorted in decreasing order of saving using heapsort. Since heapsort runs in $O(p \log p)$ to sort p items, this pre-computation costs $O(n^2 \log n)$. Then, CWA basically consists in scanning L and in performing the incumbent merger L_p if it is still valid. The validity check can be done in $O(1)$: j and k must still be located at one end of two distinct trips and the total load of their two trips must not exceed Q. If the merger L_p is accepted, the concatenation of the two trips costs $O(n)$ but at most $(n - 1)$ mergers are performed among the $O(n^2)$ in the list. The overall complexity is then dominated by the sorting phase.

The extended version (ECWA) for the LRP computes an initial solution by assigning each customer to their closest depot in which they can fit. As in the original CWA, a dedicated route is created to link the customer to its selected depot. When all customers are assigned, the depots without customers are closed, giving an initial solution looking like a bunch of flowers.

When two routes R and S are merged in ECWA, the resulting route T may be reassigned to the depot r of R, to the depot s of S, or to another depot t, opened or not (lower part of Fig. 1). Hence, $4m$ mergers must now be evaluated for each pair of

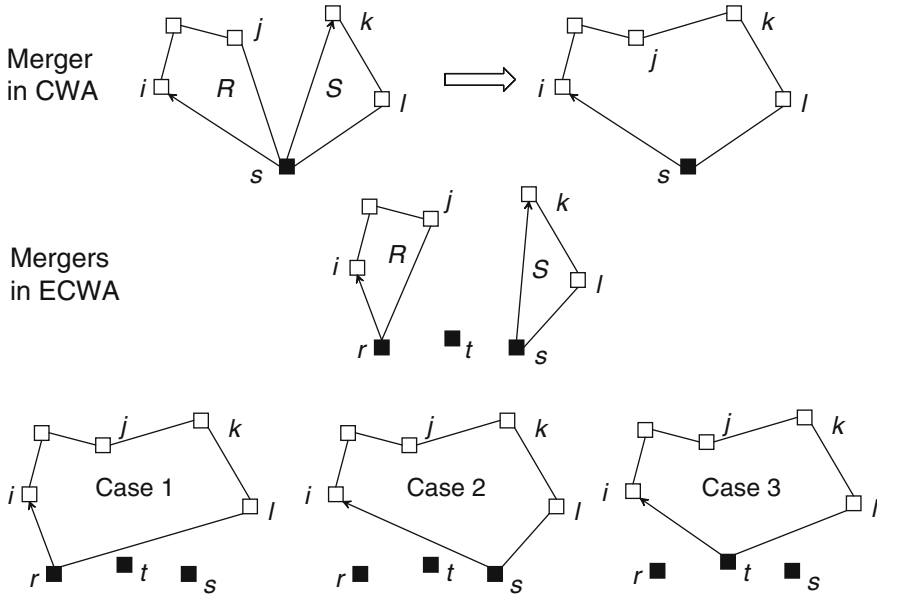

Fig. 1 Mergers in CWA and ECWA

routes. When concatenating S after R for instance, the saving σ can be computed as follows:

$$\sigma = F + C_{ri} + C_{jr} + C_{sk} + C_{ls} - C_{jk} - C_{ti} - C_{lt} + O_r\lambda_r + O_s\lambda_s - O_t(1 - y_t) \tag{12}$$

λ_r (resp. λ_s) is a boolean equal to 1 iff depot r (resp. s) has no more routes after the merger and can be closed. The boolean variable y_t already introduced for the linear model is equal to 1 iff depot t is already opened before the merger.

A straightforward implementation of ECWA runs in $O(mn^3)$ complexity. Using a preliminary ordering of possible mergers like in CWA, a lower complexity in $O(mn^2 \log n)$ is possible.

3.1.2 Randomized ECWA

A randomized version of ECWA is used for the constructive phase of the GRASP. The idea is to create a restricted candidate list (RCL) of size α from the savings calculated during the merger evaluations, and randomly choose one among this list. The elements of the RCL are the pairs of routes giving the α best savings.

Changing the RCL size during the GRASP often gives better results, as shown in (Prais and Ribeiro 2000; Resende and Rebeiro 2003). The firsts use a set of possible values for the size of the RCL, each one associated with a probability of being chosen. These probabilities are periodically re-evaluated in order to favor the sizes giving the best results. They call this approach Reactive GRASP. It seems to improve robustness

and solution quality. But here, because of the complexity of the algorithm, too few iterations are performed, and the re-evaluation would not be pertinent. So, the randomized version of the ECWA uses another technique to vary the size α of the RCL. At each merger, α is randomly selected in $[1, \alpha_{\max}]$, where α_{max} is the maximum RCL size permitted. Then, a uniform draw determines the merger to perform.

3.2 Local search

The improvement phase calls a local search procedure (LS) whose each iteration explores the three following neighborhoods, in the given order.

- *MOVE* One customer is shifted from its current position to another position, in the same route or in a different route which may be assigned to the same depot or not, provided capacities are respected.
- *SWAP* Two customers are exchanged. They may belong to the same route or, if residual capacities allow it, to two distinct routes sharing one common depot or not.
- *OPT* This is a new 2-opt procedure in which two non-consecutive edges are removed, either in the same route or in two distinct routes assigned to a common depot or not. When they belong to different routes, there are various ways of reconnecting the trips. If they are from different depots, edges connecting the last customers of the two considered routes to their depot have to be replaced to satisfy the constraint imposing that a route must begin and finish at the same depot (Fig. 2). This neighborhood is equivalent in the first case to the well-known 2-OPT move (Lin and Kernigham 1973).

The LS performs the first improving move detected. It stops when no additional one can be found.

It is important to notice that no depot can be opened during the local search, enabling to search the three neighborhoods in $O(n^2)$.

The two components ECWA and LS are enough for a basic GRASP performing a fixed number of iterations *maxit*. This basis is summarized in Fig. 3.

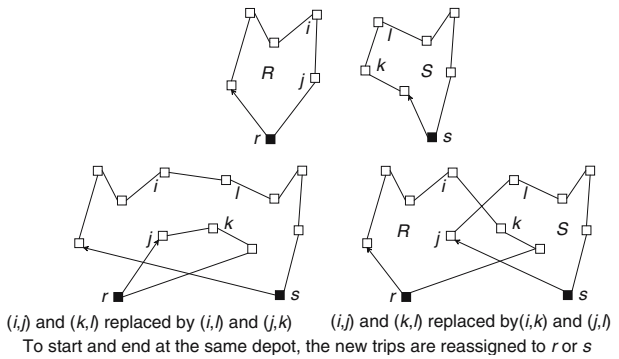

Fig. 2 Example of the new 2-opt procedure

```
bcost := +infinity           //initialize best solution cost
for k := 1 to maxit do       //main loop
   ECWA(S)                   //build one greedy random solution
   LocalSearch(S)            //improve it by local search
   if cost(S) < bcost then   //if best solution is improved
      bsoln := S             //update best solution
      bcost := cost(S)       //and its cost
   endif
endfor return (bsoln)        //the result is the best solution
```

Fig. 3 Basic GRASP algorithm for the LRP

3.3 Learning process

The ECWA might sometimes be too weak to select a good subset from I while a preliminary testing shows that solution quality strongly depends on this selection. Many depots are opened in the bunch of flowers, and the mergers can close some of them but maybe not enough. To help it, a subset SD of available depots is used. SD varies at each iteration for the construction of the bunch of flowers. At the first iteration, all the depots are in SD, then, only two of them composed the set. One is iteratively picked in I to be sure that each depot is open at least once, and the second is randomly chosen among the remaining ones. Then, the assignment of the customers is done on SD. If a customer cannot be assigned by lack of capacity in the depots from SD, it can add its closest depot in the set. During the mergers, SD equals I. All this can be seen as a diversification.

Adding a memorization during the diversification mode would enable the algorithm to learn about the good subset to open, and to find better solutions. An intensification mode using this learning process on the depots has been implemented and oscillations between diversifications and intensifications are used.

The *diversification mode* (initial mode) is applied during *maxitdiv* iterations. The aim is to explore the solution space by varying the set of open depots, as explained above. Then, the LS tries to improve the result. If the obtained solution is the best one found during the diversification period, it is recorded in *bsolndiv*, and in *BestSol* if it is the best one until now in the entire algorithm.

The *intensification mode* lasts *maxitint* iterations and restricts the level of randomness: SD is composed of the subset of depots opened in *bsolndiv*, and not only during the construction of the bunch of flowers, but also during the mergers. If necessary, to assign all the customers, additional depots can be added to SD, as already described, when constructing the initial bunch of flowers. This mode essentially tries to improve the routing from selected depots. Only the ones in SD can be opened but it is possible to close some of them. Then, once again, the LS is performed and if the obtained solution is the best one found until now, it is saved in *BestSol*.

These two modes with learning process have similarities with the memorization proposed by (Fleurent and Glover 1999). They store some elite solutions during the GRASP iterations and they grant the "strongly determined" variables the most used in these solutions to be integrated during the next constructions. Strongly determined variables are the ones than cannot be modified without eroding the objective value or changing significantly other variables, like the opening depot variable y_i.

4 Path relinking and whole algorithm

This Section presents the path relinking (PR) proposed as post-optimization for the GRASP and the whole algorithm of the metaheuristic. The parameters used are also given.

4.1 Path relinking

A PR can be applied to most metaheuristics as a kind of intensification or post-optimization: new solutions are generated by exploring trajectories that connect a small subset of high-quality solutions collected during the search. Each trajectory is obtained by introducing in a solution U some attributes from another solution T called the guiding solution. Laguna and Martí (1995) have first introduced the PR with a GRASP as an intensification. Then, Piñana et al. (2004) have used the PR as a post-optimization in a two-phase implementation.

In intensification, the PR is realized between each locally optimal solution U from each iteration of the GRASP, and another one T. The latter is randomly chosen among a group of elite solutions, often with a bias to the most distant ones from U, that in general permits to obtain better results (Piñana et al. 2004; Resende and Werneck 2002). The group of elite solutions is composed of solutions found during the GRASP under conditions of being among the best ones and the more distant from each other in the group.

In post-optimization, the principle is similar, but the PR is done at the end and only between solutions from the elite group. Usually, it is a kind of generational method, the new solutions found replacing the worst ones from the group if these new ones are better and distant enough from the elite solutions.

Here, the PR is used as post-optimization but no substitution by the new solutions occurs in the elite group. It is a trade-off to obtain some better results in short time.

4.1.1 Distance

To constitute the group of elite solutions, a "distance" $D(T, U)$ measuring the differences between two solutions T and U is useful. It can be defined as follows. For each pair (i, j) of consecutive customers in T, four cases are considered:

- The pair (i, j) or (j, i) is found in U, so the pair is said not broken, and does not contribute to the distance measure.
- i and j are no longer adjacent in U, but they are in a common trip: count 1 in the distance.
- i and j are in different trips in U, but assigned to one common depot: count 3.
- i and j are assigned to two different depots in U : count 10.

Moreover, for each customer i serviced at the beginning of a trip in T, count 10 if i is assigned to another depot in U.

This measure is not exactly a distance in the mathematical sense, because, among others, the triangle inequality does not hold. However, note that $D(T, U) \geq 0$ and

$T = U \Longleftrightarrow D(T, U) = 0$. Moreover, this distance measure correctly detects equivalent solutions, e.g., when the trips are renumbered or when some of them are inverted (performed backward by the vehicle).

4.1.2 Path relinking method

The path relinking proposed in this paper consists of finding two trajectories between each pairs of elite solutions : from U to T, and then from T to U by the following principle. Let *NBestSet* be the set of the *NBest* best solutions found during the diversification mode of the GRASP and *DistSet* be the set of *NMaxDist* solutions used for the PR (elite group). The first solution introduced in *DistSet* is the best one. Then, among *NBestSet*, the farthest solution from *DistSet* is added and so on until reaching *NMaxDist* solutions.

From the couple (T, U), U is transformed step by step into T, the latter called the guiding solution. The attributes introduced in U are the edges that repair broken pairs of customers (i, j), and they are taken in the order encountered in T.

For each pair of consecutive customers (i, j) in T but non adjacent in U, the pair is said broken and a repair occurs. It consists in shifting the whole sequence of consecutive customers located after i in T such that no new broken pair appears in U. The procedure is repeated as long as the capacity constraints hold for both routes and depots or until finding U equal to T. When capacity violations happen, they are corrected. A check looks for the customers from U, in the violated route or depot, that are not in the same sequence as in T, and a correction moves chains of customers as for a repair, except that the pairs (i, j) are considered in the order encountered in U, until respecting all the capacity constraints. Each feasible solution obtained after a repair undergoes a local search LS.

4.2 Algorithm overview

The components previously described are integrated in the general algorithm summarized by Fig. 4. The oscillation between diversification and intensification phases is simply implemented by using a boolean flag *divmode*, true in diversification mode. The *NBest* best solutions obtained are stored in a table *S*.

The parameters of the metaheuristic have been set empirically after a testing phase : the RCL maximum length $\alpha_{\max} = 7$, the number of iterations per diversification period $maxitdiv = 7$, number of iterations per intensification period $maxitint = 5$, the maximum number of total iterations $maxit = 60$ to have five swaps of periods, $NBest = 10$ in order to take solutions from different diversification periods, and $NMaxDist = 3$ to restrain the trajectories to explore in the Path Relinking to the ones between the most diversified solutions.

5 Computational evaluation

This section is devoted to the computational evaluation. The proposed GRASP is appraised on three sets of randomly generated instances. The algorithm is compared

```
//GRASP PHASE
divmode  := true                        //Diversification mode
bcostdiv := +infinity                   //Best soln cost in this mode
SD    := I                              //All depots are in Delta
itdiv := 0                              //Iteration counter by mode
for k:= 1 to maxit do                   //Main loop
   ECWA(SD,S(k))                        //Get soln S(k) using depots
                                        //in Delta
   LocalSearch(S(k))                    //Improve it by local search
   if divmode = true then               //If diversification mode
      if cost(S(k)) < bcostdiv then     //If best soln of last period
                                        //is improved
         bsolndiv := S(k)               //Update this soln
         bcostdiv := cost(S(k))         //And its cost
      endif
      itdiv := itdiv + 1                //Count one iteration
      ChooseDepots(SD)                  //Enable 2 depots
      if itdiv = maxitdiv then          //If end of diversification mode
         divmode := false               //Switch to intensification mode
         itint := 0                     //Reset iteration counter for
                                        //this mode
         SD := depots of bsolndiv       //Restrict possible depots
      endif
   else                                 //If current mode is
                                        //intensification
      itint := itint + 1                //Count 1 iteration in this mode
      if itint = maxitint then          //If end of intensification mode
        divmode  := true                //Switch to diversification
        bcostdiv := +infinity           //Reset best soln cost in
                                        //the mode
        itdiv    := 0                   //Reset iteration counter for
                                        //this mode
      endif
   endif
endfor
//PATH RELINKING PHASE
Elite(S,B)                              //Subset B of elite solutions
bsoln := best solution of S             //Best solution before the PR
for each pair {u,v} in B                //Do two paths for each pair
                                        //in B
   PathRelinking (u,v,Path)             //Returning a sequence of
                                        //solutions Path
   for each z in Path                   //For each solution z of Path
      if cost(z) < cost(bsoln) then     //If bsoln is improved
        bsoln := z                      //Then update bsoln
      endif
   endfor
endfor return (bsoln)                   //Return the best solution
```

Fig. 4 Proposed GRASP algorithm for the LRP

with a multi-start local search heuristic proposed by Prins et al. (2004) on the first set of instances, with a cluster based heuristic and a lower bound proposed by Barreto (2004) on the second set of instances, both sets having capacitated routes and depots, and also with the two-phase TS proposed by Tuzun and Burke (1999), using the third set of instances with uncapacitated depots.

The computational results are given in Tables 1, 2 and 3. GRASP, HS, TS, CL and LB refers to the results obtain respectively by the proposed metaheuristic, the heuristic proposed in Prins et al. (2004), the tabu search from Tuzun and Burke (1999), and the

Table 1 Results with capacities on routes and depots – first set

n	m	β	Type	HS		GRASP		Gap (%)	CD	CR
				Cost	CPU	Cost	CPU			
20	5	0	a	55,806	0.00	55,021	0.00	−1.41	8,516	5,894
20	5	0	b	39,104	0.00	39,104	0.00	0.00	7,749	7,869
20	5	2	a	49,668	0.00	48,908	0.00	−1.53	8,065	4,942
20	5	2	b	37,542	0.00	37,542	0.00	0.00	6,956	7,877
50	5	0	a	98,079	0.02	90,632	0.03	−7.59	8,481	5,433
50	5	0	b	72,159	0.03	64,761	0.03	−10.25	7,693	8,229
50	5	2	a	90,188	0.03	88,786	0.04	−1.55	9,773	4,956
50	5	2	b	68,675	0.05	68,042	0.04	−0.92	9,773	6,454
50	5	3	a	96,756	0.01	87,380	0.04	−9.69	9,481	5,702
50	5	3	b	61,844	0.05	61,890	0.03	0.07	9,481	7,155
100	5	0	a	284,613	0.48	279,437	0.46	−1.82	44,297	6,106
100	5	0	b	218,052	0.74	216,159	0.39	−0.87	44,297	6,939
100	5	2	a	198,041	0.32	199,520	0.29	0.75	51,123	4,053
100	5	2	b	159,812	0.63	159,550	0.37	−0.16	51,123	5,209
100	5	3	a	204,906	0.30	203,999	0.36	−0.44	44,144	4,628
100	5	3	b	155,865	0.25	154,596	0.34	−0.81	44,144	6,028
100	10	0	a	323,438	0.78	323,171	0.62	−0.08	51,018	4,581
100	10	0	b	277,678	0.38	271,477	0.49	−2.23	49,077	6,264
100	10	2	a	292,181	0.39	254,087	0.65	−13.04	51,467	3,987
100	10	2	b	246,109	1.75	206,555	0.50	−16.07	49,862	5,179
100	10	3	a	257,932	0.67	270,826	0.59	5.00	52,930	4,481
100	10	3	b	209,339	1.12	216,173	0.66	3.26	48,233	6,498
200	10	0	a	576,134	6.12	490,820	8.62	−14.81	88,717	4,681
200	10	0	b	402,097	16.53	416,753	6.32	3.64	95,261	5,953
200	10	2	a	551,914	8.49	512,679	9.24	−7.11	102,405	4,193
200	10	2	b	388,975	4.38	379,980	6.12	−2.31	93,457	4,331
200	10	3	a	555,006	15.19	496,694	7.08	−10.51	81,066	5,511
200	10	3	b	449,288	23.69	389,016	4.84	−13.42	87,198	5,792
Average				218,871	2.75	207,322	1.61	−3.77		
Mean deviation				4.48		0.71				
Standard deviation				6.16		1.51				

Table 2 Results with capacities on routes and depots – second set

n	m	Q	LB	CL	Gap	GRASP	Gap (%)
21	5	6,000	424.9*	435.9	2.6	429.6	1.1
22	5	4,500	585.1*	591.5	1.1	585.1*	0.0
27	5	2,500	3,062.0*	3,062.0*	0.0	3,062.0*	0.0
29	5	4,500	512.1*	512.1*	0.0	515.1	0.6
32	5	8,000	556.5	571.7	2.7	571.9	2.8
32	5	11,000	504.3*	511.4	1.4	504.3*	0.0
36	5	250	460.4*	470.7	2.2	460.4*	0.0
50	5	160	549.4	582.7	6.1	599.1	9.1
75	10	140	744.7	886.3	19.0	861.6	15.7
88	8	9,000,000	356.4	384.9	8.0	356.9	0.1
100	10	200	788.6	889.4	12.7	861.6	9.3
134	8	850	–	6,238.0	–	5,965.1	–
150	10	8,000,000	4,3406.0	46,642.7	7.5	44,625.2	2.8
Average					5.3		3.5

Table 3 Results for uncapacitated depots and capacitated routes – third set

n	m	β	Type	HS		GRASP		Gap (%)	CD	CR
				Cost	CPU	Cost	CPU			
100	10	0	0.75	1,556.64	5	1,525.25	0.54	−2.02	100	111
100	20	0	0.75	1,531.88	3	1,526.90	0.68	−0.32	100	112
100	10	0	1	1,443.43	3	1,423.54	0.46	−1.38	100	111
100	20	0	1	1,511.39	4	1,482.29	0.60	−1.93	100	117
100	10	3	0.75	1,231.11	4	1,200.24	0.46	−2.51	100	91
100	20	3	0.75	1,132.02	2	1,123.64	0.57	−0.74	100	75
100	10	3	1	825.12	3	814.00	0.38	−1.35	100	43
100	20	3	1	740.64	3	747.84	0.62	0.99	100	41
100	10	5	0.75	1,316.98	3	1,273.10	0.36	−3.33	100	88
100	20	5	0.75	1,274.50	4	1,272.94	0.60	−0.12	100	98
100	10	5	1	920.75	4	912.19	0.34	−0.93	100	51
100	20	5	1	1,042.21	3	1,022.51	0.64	−1.89	100	66
150	10	0	0.75	2,000.97	12	2,006.70	1.88	0.29	100	107
150	20	0	0.75	1,892.84	12	1,888.90	2.69	−0.21	100	93
150	10	0	1	2,022.11	14	2,033.93	1.67	0.58	100	102
150	20	0	1	1,854.97	13	1,856.07	2.21	0.06	100	91
150	10	3	0.75	1,555.82	9	1,508.33	1.96	−3.05	100	76
150	20	3	0.75	1,478.80	12	1,456.82	2.77	−1.49	100	79
150	10	3	1	1,231.34	9	1,240.40	1.78	0.74	100	65
150	20	3	1	948.28	9	940.80	2.37	−0.79	100	38
150	10	5	0.75	1,762.45	9	1,736.90	1.55	−1.45	100	85
150	20	5	0.75	1,488.34	9	1,425.74	2.14	−4.21	100	70
150	10	5	1	1,264.63	10	1,223.70	1.48	−3.24	100	54
150	20	5	1	1,182.28	9	1,231.33	2.25	4.15	100	49
200	10	0	0.75	2,379.47	22	2,384.01	5.13	0.19	100	99
200	20	0	0.75	2,211.74	22	2,288.09	6.83	3.45	100	86
200	10	0	1	2,288.17	23	2,273.19	5.19	−0.65	100	94
200	20	0	1	2,355.81	26	2,345.10	6.98	−0.45	100	93
200	10	3	0.75	2,158.60	20	2,137.08	5.63	−1.00	100	84
200	20	3	0.75	1,787.02	18	1,807.29	6.17	1.13	100	67
200	10	3	1	1,549.79	18	1,496.75	4.04	−3.42	100	62
200	20	3	1	1,112.96	18	1,095.92	5.14	−1.53	100	36
200	10	5	0.75	2,056.11	23	2,044.66	4.71	−0.56	100	72
200	20	5	0.75	2,002.42	20	2,090.95	6.65	4.42	100	77
200	10	5	1	1,877.30	20	1,788.70	3.32	−4.72	100	76
200	20	5	1	1,414.83	17	1,408.63	4.94	−0.44	100	41
Average				1,567	11.5	1,557	2.66	−0.77		
Mean deviation				1.14		0.75				
Standard deviation				1.35		1.54				

clustering method and the lower bound proposed by Barreto (2004). Columns called n, m and β respectively indicate the number of customers, depots and clusters, and "type" represents the capacity of the routes for instances from Prins et al. (2004), or the cluster ratio for instances from Tuzun and Burke (1999). "Q" is the vehicle capacity. "Cost" gives the value of the objective function, and the CPU time is given in minutes except for TS for which it is in seconds. The gap is the relative deviation of the algorithms to the GRASP. CD and CR are respectively the average cost of the open depots and the average total cost of a route (set up plus routing).

5.1 Implementation and instances

The proposed method is coded in C++ and executed on a Dell OPTIPLEX GX260 PC 512MB of RAM, with a Pentium 4 processor clocked at 2.4 GHz and Windows XP. It is tested on three set of instances described below.

The first set composed of 28 instances, with capacitated routes and depots, comes from a paper on a multi-start local search method for the LRP (Prins et al. 2004). The number m of depots considered is set to 5 or 10, the number of clients is $n \in \{20, 50, 100, 200\}$, the vehicle capacity Q is set to 70 or 150 (indicated by letters a and b in the table) and the number of clusters is $\beta \in \{1, 2, 3\}$. $\beta = 1$ means in fact an uniform distribution in the Euclidean plane. The number of depots is arbitrarily limited to ten but adding more depots does not deteriorate the results, as shown in Sect. 5.4.

The C_{ij} correspond to the Euclidean distances, rounded up to the next integers. The other data (demands, depot capacities and fixed costs) are also integer. They are randomly generated:

- The demand comes from a uniform distribution in the range [11, 20].
- The depot capacities ensure to open at least two or three ones.

The second set is composed of instances for capacitated routes and depots obtained either from literature or adapted from data related with VRP, and is available in the web page http://sweet.ua.pt/~iscf143.

The third set is composed of 36 instances with $n \in \{100, 150, 200\}$, $m \in \{10, 20\}$, $Q = 150$ and uniform demands in $[1, 20]$. Spatial distribution is also controlled, using two factors: the number of clusters (0, 3 or 5, where here 0 refers to a uniform distribution), and the percentage (75% or 100%) of customers that belong to a cluster. Distances are not rounded and solution costs are computed using floating point numbers.

5.2 Results on the first set of instances

The method is tested on the first set of instances and compared with the results found by HS, which is a multi-start heuristic with some randomness in its search and a look-ahead technique. This technique permits to explore a complex neighborhood by trying to modify the set of open depots and transferring customers to other depots. Such sequences of moves may increase the total cost and the algorithm backtracks to the solution as it was before the sequence.

Table 1 shows that on average, GRASP saves 3.77% in comparison with HS. Nevertheless, in some cases, HS could find better results. It is due to the use of a multi-start strategy and randomness. So, by chance, it can sometimes find a better subset of depots to open. As the results are very sensitive to the choice of the depots on theses instances (their cost having a great part on the total cost), the relative deviation can be important (up to 16%).

Another point is that the average running time is more than halved compared to HS. The large running times of HS are due to the search on complex neighborhoods with backtracks.

5.3 Results on the second set of instances

Performances of GRASP is then compared with CL and LB on the second set of instances. CL is a four-phase cluster based heuristic executed 24 times using four clustering methods and six proximity measures. The four phases are: construction of groups of customers with a capacity limit – determination of the distribution route in each customer group – improvement of the routes – location of the depots and assignment of the routes to them. LB is a lower bound obtained by running CPLEX 7.1 on a relaxed two-index integer linear programming formulation, including the violated constraints in the next iterations.

Table 2 provides the results. Asterisks correspond to the optimal solution. It shows that GRASP is not farther than 0.28% of the optimal solution on average when this one is known, while CL is at 1.2%, reaching the optimal solution on two instances only against four for the GRASP. Furthermore, when solving bigger instances, CL becomes clearly worst than GRASP. That is due to the fact that CL runs 24 heuristics that design routes without taking into account the position of the depots. So, such a technique works well and provides a chance to find a good solution, but only on small instances and with a small set of possible depots.

Concerning CPU-time that is not report in the table, CL has a mild advantage. In fact, on all the instances with less than 50 customers, both methods take less than 1 s. On the other ones, GRASP runs less than 25 s, except with 134 customers (50 s) and 150 customers (156 s), while CL still takes 1 s. But it seems to be the running time for only one out of the 24 heuristics (the one providing the best result on the considered instance).

5.4 Results on the third set of instances

Finally, GRASP is tested on a third set of instances, with uncapacitated depots, and its results are compared with TS that is a tabu search metaheuristic specially designed for uncapacitated depots. So it uses a simple and fast strategy consisting of computing the best solution for one open depot, then two, and so on until the total cost increases. Furthermore, the local search in TS is also fast, because the neighborhood may exchange the status open/closed of two depots, but without changing the number of open depots. These strategies are not applicable for the GRASP because it would lead to capacity violations in the general case.

Table 3 compares GRASP and TS, where TS was executed on a 266 MHz Pentium II PC.

The proposed GRASP is competitive with TS, since its average deviation to the TS algorithm is -0.77% and it improves 26 solutions out of 36 (72%), especially all the instances except one with 100 customers. This performance is remarkable and the results can even be better when *maxit* increases (e.g., gap of -1.76% with 500

iterations for an average CPU time of less than 12 min) while our algorithm is designed for a more general LRP with capacitated depots.

However, GRASP consumes more CPU time than TS. Indeed, TS being designed for uncapacitated depots, it can use a simpler and faster search strategy. Nevertheless, CPU time remains reasonable as it does not exceed a few minutes while the problem is a strategic one and instances are of realistic size.

Anyway, most published VRP heuristics require dozens of minutes for instances of this size (Cordeau et al. 2004). Moreover, the running time of the proposed heuristic remains of the same order of magnitude on the more combinatorial case with capacitated depots (see Table 1).

5.5 Impact of the components of the algorithm

As shown in Tables 1, 2 and 3, the GRASP is able to find good quality solutions. But it might be interesting to develop the interest of the different components of the algorithm.

First, a memory is here added to the traditional GRASP method. That permits to reduce the computational time (around 30% less time than a GRASP without memory). Indeed, the complexity of the algorithm depends on the number of depots ($O(mn^2 \log n)$). The diversification mode is necessary to explore the solution space, but once some "good" depots seem to emerge from this phase, restricting the set of available depots allows to decrease the complexity. Furthermore, this technique of memorization provides best results for nearly half of the instances.

Second, a post-optimization operates at the end of the GRASP. The interest of this phase is shown in Table 4, where n and m are the number of customers and of depots respectively, GRASP is the complete algorithm, *WithoutPR* corresponds to

Table 4 Impact of the path relinking

n	m	GRASP	*WithoutPR*	Gap (%)
Capacitated routes and depots				
20	5	45,144	45,151	−0.01
50	5	74,701	74,997	−0.39
100	5	202,210	203,029	−0.37
100	10	257,048	258,748	−0.67
200	10	447,657	448,397	−0.16
Average		207,322	208,054	−0.35
Capacitated routes				
100	10	1191.39	1,210.71	−1.54
100	20	1196.02	1,205.72	−0.88
150	10	1624.99	1,636.66	−0.69
150	20	1466.61	1,493.11	−1.79
200	10	2020.73	2,052.86	−1.52
200	20	1839.33	1,861.52	−1.11
Average		1,557	1,577	−1.25

the GRASP without path relinking and the gap is the improvement supplied by the path relinking. The impact of this last search is confirmed here. A GRASP being a randomized method, it provides a set of fair various solutions at each iteration, but a post-optimization might be necessary to extract really good ones. Nevertheless, the results also show that the instances with capacitated depots take less advantage of the post-optimization. In fact, during the path relinking, each time a broken pair of customers is repaired, it may occur violation of capacity constraints, and even more if the routes and the depots have limited capacities. So to obtain feasible solutions, several repairs have to be done and the exploration of the solution space is more scattered. In these conditions, only 2/3 of the solutions on instances with capacitated depots are improved by the path relinking against almost 90% on instances with only capacitated routes. As this post-optimization is time-consuming especially when violations of capacities occur (around 70% of the time with capacitated depots, 50% of the time otherwise), a suggestion is to use it only with uncapacitated depots. So, the improvement of the results compared with HS on instances with capacitated depots would remain reasonable (-3.44%), in 0.4 minutes only on average.

6 Conclusion

In this paper, a new metaheuristic for the LRP with both capacitated depots and vehicles is presented. The method is a GRASP with learning process based on a randomized and extended version of Clarke and Wright algorithm, followed by a path relinking procedure as post-optimization.

The method is tested on three sets of small and large scale instances with up to 200 customers. The proposed GRASP is able to improves around two thirds of solution values obtained either by heuristics or, when depots are uncapacitated, by a Tabu Search specially tailored for this particular case of the LRP. GRASP is able to reach optimal solutions on a few instances when this one is known, and is closer to a lower bound than a four-phase clustering method. Only the CPU time is a little long in comparison with the TS dedicated to uncapacitated depots, but remains reasonable for such a strategic problem : implementing a new logistic network that will be operated for years is worth spending a few minutes of computer time!

Furthermore, the extended Clarke and Wright algorithm introduced in this paper could be used to provide initial solutions in metaheuristics for other problems where several depots are available, e.g., the multi-depot VRP.

Acknowledgements This research is partially supported by the Conseil Régional de Champagne-Ardenne (Champagne-Ardenne district council) and by the European Social Fund.

References

Albareda-Sambola M, Díaz JA, Fernández E (2005) A compact model and tight bounds for a combined location-routing problem. Comput Oper Res 32(3):407–428

Barreto SS (2004) Análise e Modelização de Problemas de localização-distribuição (Analysis and modelization of location-routing problems)(in Portuguese). PhD thesis, University of Aveiro, campus universitário de Santiago, 3810–193

Bruns A, Klose A (1995) An iterative heuristic for location-routing problems based on clustering. In: Proceedings of the 2nd international workshop on distribution logistics, Oegstgeest
Chan Y, Carter WB, Burnes MD (2001) A multiple-depot, multiple-vehicle, location-routing problem with stochastically processed demands. Comput Oper Res 28:803–826
Chien TW (1993) Heuristic procedures for practical-sized uncapacitated location-capacitated routing problems. Decis Sci 24(5):995–1021
Clarke G, Wright JW (1964) Scheduling of vehicles from a central depot to a number of delivery points. Oper Res 12:568–581
Cordeau JF, Gendreau M, Hertz A, Laporte G, Sormany JS (2004) New heuristics for the vehicle routing problem. In: Technical Report G-2004-33, GERAD
Feo TA, Resende MGC (1995) Greedy randomized adaptive search procedures. J Glob Optim 2:1–27
Fleurent C, Glover F (1999) Improved constructive multistart strategies for the quadratic assignment problem using adaptative memory. INFORMS J Comput 11:198–204
Ghiani G, Laporte G (2001) Location-arc routing problems. OPSEARCH 38:151–159
Labadi N (2003) Problèmes tactiques et stratégiques en tournées sur arcs (Tactical and strategic problems on arc routing). PhD thesis, University of Technology of Troyes
Laguna M, Martí R (1995) Grasp and path relinking for 2-layer straight line crossing minimization. INFORMS J Comput 11(1):44–52
Laporte G, Gendreau M, Potvin JY, Semet F (2000) Classical and modern heuristics for the vehicle routing problem. Int Trans Oper Res 7:285–300
Laporte G, Louveaux F, Mercure H (1989) Models and exact solutions for a class of stochastic location-routing problems. Eur J Oper Res 39:71–78
Laporte G, Norbert Y, Taillefer S (1988) Solving a family of multi-depot vehicle routing and location-routing problems. Transportation Sci 22(3):161–167
Lin S, Kernigham BW (1973) An effective heuristic algorithm for the traveling salesman problem. Oper Res 21:498–516
Min H, Jayaraman V, Srivastava R (1998) Combined location-routing problems: a synthesis and future research directions. European J Oper Res 108:1–15
Piñana E, Plana I, Campos V, Martí R (2004) GRASP and path relinking for the matrix bandwidth minimization. Eur J Oper Res 153:200–210
Prais M, Ribeiro CC (2000) Reactive GRASP: an application to a matrix decomposition problem in TDMA traffic assignment. INFORMS J Comput 12(3):164–176
Prins C, Prodhon C, Wolfler Calvo R (2004) Nouveaux algorithmes pour le problème de localisation et routage sous contraintes de capacité. In: A. Dolgui, Dauzère-Pérès S. (eds) MOSIM'04, vol 2, pp 1115–1122, Ecole des Mines de Nantes, Lavoisier
Resende MGC, Rebeiro CC (2003) Greedy randomized adaptive search procedures. In: Glover F., Kochenberger, G. (eds) Handbook of metaheuristics, Kluwer, Dordrecht, pp 219–249
Resende MGC, Werneck RF (2002) A hybrid heuristic for the p-median problem. In: Technical report TD-SNWRCR, AT & TLab Research, Florham Park, NJ 07932
Salhi S, Rand GK (1989) The effect of ignoring routes when locating depots. Eur J Oper Res 39:150–156
Srivastava R (1993) Alternate solution procedures for the locating-routing problem. OMEGA Int J Manage Sci 21(4):497–506
Tuzun D, Burke LI (1999) A two-phase tabu search approach to the location routing problem. Eur J Oper Res 116:87–99
Wu TH, Low C, Bai JW (2002) Heuristic solutions to multi-depot location-routing problems. Comput Oper Res 29:1393–1415