2014 IEEE Congress on Evolutionary Computation (CEC)
July 6-11, 2014, Beijing, China

Genetic Algorithm with Spatial Receding Horizon Control for the Optimization of Facility Locations

Xiao-Bing Hu
The State Key Laboratory of Earth Surface Processes and Resource Ecology
Beijing Normal University
Beijing, China

Xiao-Bing Hu, Mark S Leeson
School of Engineering
University of Warwick
Coventry, UK

***Abstract*—Inspired by the temporal receding horizon control in control engineering, this paper reports a novel spatial receding horizon control (SRHC) strategy to partition the facility location optimization problem (FLOP), in order to reduce the complexity caused by the problem scale. Traditional problem partitioning methods can be viewed as a special case of the proposed SRHC, i.e., one-step-wide SRHC, whilst the method in this paper is a generalized *N*-step-wide SRHC, which can make a better use of global information of the route network where a given number of facilities need to be set up. With SRHC to partition the FLOP, genetic algorithm (GA) is integrated as optimizer to resolve the partitioned problem within each spatial receding horizon. On one hand, SRHC helps to improve the scalability of GA. On the other, the population feature of GA helps to reduce the shortsighted performance of SRHC. The effectiveness and efficiency of the reported SRHC and GA for the FLOP are demonstrated by comparative simulation results.**

***Keywords*—Genetic Algorithm, Spatial Receding Horizon Control, Facility Location Optimization, Problem Partitioning.**

I. INTRODUCTION

Given a route network, a distribution of users and a set of optional locations, the facility location optimization problem (FLOP) is concerned with where to locate a specific number of facilities, so that the users can access the facilities the most efficiently [1], [2]. The FLOP has a broad real-world application background. For example, a supermarket company setting-up its store chain in a community [3], local government organizing earthquake shelters in a city [4], and a logistic company establishing its distribution stations in a country [5].

To address the FLOP, researchers have already attempted many different methods [3]-[9]. In particular, in common with applications to many other NP-hard problems, evolutionary computation (EC) methods as large-scale parallel stochastic searching and optimization algorithms have demonstrated good potential in resolving the FLOP. However, the poor scalability of these reported methods largely hampers their applications in the large-scale FLOP, which may easily include millions of nodes. As a family member of population-based algorithms, EC methods are generally very expensive in terms of memory demand and computational time in the case of large-scale problems [10], [11]. To address the scalability problem, decentralized and distributed versions of algorithms often need to be developed. Before such decentralized and distributed algorithms can be applied, a problem partitioning method has to be employed in order to divide a large-scale network into some sub-graphs of manageable size. This paper attempts to shed some more light on how to design an effective scalable EC method, to be more precise a genetic algorithm (GA), for the FLOP.

In a conventional problem partitioning method (e.g., see [12], [13]), a large-scale problem is divided into a series of separate sub-problems. Then, each sub-problem is resolved in a rather isolated manner. After all sub-problems have been resolved independently their sub-solutions are integrated together to form a complete solution to the original large-scale problem. However, even though optimal sub-solutions to the sub-problems can be found, the integrated complete solution to the original large-scale problem is often not optimal or even good. In other words, optimal sub-solutions to the sub-problems are often not optimal at all from a global point of view. A major cause of losing the global optimality is the independent/isolated way of resolving each sub-problem. Inspired by the temporal receding horizon control (TRHC) strategy in the area of control engineering [14], [15], we have recently proposed a novel spatial receding horizon control (SRHC) strategy to partition large-scale network coding problems in [16]. In the SRHC problem partitioning strategy, a large-scale problem is divided into many sub-problems, which compose a problem space, a spatial horizon is then defined which covers some sub-problems each time and will recede in the problem space. The spatial horizon is composed of several spatial steps. Each time the spatial horizon recedes by a spatial step. All sub-problems covered by a spatial horizon will be optimized as a whole, and only the sub-solutions to the sub-problems within the first step of the spatial horizon will be saved and fixed, whilst others will be discarded and then recalculated in the next spatial horizon. With the SRHC strategy, a sub-problem will be optimized not in an independent/isolated manner, but by making use of its

This work was supported in part by the Seventh Framework Programme (FP7) of the European Union under Grant PIOF-GA-2011-299725, and the Scientific Research Foundation for the Returned Overseas Chinese Scholars, State Education Ministry, China.

neighboring information in the problem space. Simply speaking, the conventional problem partitioning strategy can be viewed as a one-step-wide SRHC, whilst the new method proposed here is a generalized N-step-wide SRHC. Clearly, by optimizing a sub-problem together with its neighboring sub-problems, it is likely that the quality of the associated sub-solution will be improved in terms of global optimality. The solution quality may be further improved by integrating a population based method into the SRHC strategy by setting up a solution pool for the sub-problems in those decided spatial steps.

This paper particularly attempts to apply the SRHC strategy proposed in [16], and develops an effective GA to resolve the FLOP. As a new application study, the FLOP in this paper will further demonstrate the practicability of the SRHC in [16]. The remainder of this paper is organized as follows. Section 2 gives a mathematical description of FLOP. Section 3 explains the details of the SRHC strategy and the design of GA with SRHC for the FLOP. Some simulation results are discussed in Section 4, and the paper ends with some conclusions and comments for future work in Section 5.

II. PROBLEM DESCRIPTION OF FLOP

We need a mathematical model of the FLOP. Suppose a route network $G(V,E)$ is composed of node set V and a connection set E. V has N_N different nodes which represent user and/or candidate facility locations (CFLs), and N_L links/connections. In this study, there are two values associated with each node i, (i=1,…,N_N): one is the category of node i, denoted as $C(i)$, and the other is the number of users at node i, denoted as $U(i)$. Basically, there may be 4 categories of nodes: user location only, CFL only, both, and neither of them (e.g., just a route junction), and the associated $C(i)$ can be valued as 1, 2, 3 and 0, respectively. Thus, for $C(i)$=2 or 0, we always have $U(i)$=0, while for $C(i)$=1 or 3, $U(i)\geq 1$. Based on node category, all nodes in the route network can be divided into 4 sets: Ω_1, Ω_2, Ω_3 and Ω_0, and set Ω_m includes all nodes with $C(i)=m$, m=1, 2, 3 or 0. The route network can be recorded as an $N_N \times N_N$ adjacent matrix A. The matrix entry $A(i,j)$=1, i=1,…,N_N and j=1,…,N_N, defines a connection, i.e., a direct route link from node i to node j. Otherwise, $A(i,j)$=0 means no direct route. We assume $A(i,i)$=0, i.e., no self-connecting route is allowed. If $A(i,j)$=1, then the direct connection between node i to node j has a length $L(i,j)>0$.

Suppose it is planned to set up N_F facilities in the route network. Then, the FLOP aims to find the best N_F locations among the sets Ω_2 and Ω_3, so that the overall distance from all nodes in Ω_1 and Ω_3 to access a facility (no matter which facility) is minimized. Let S be a solution to the FLOP. S is clearly a set of N_F nodes, and

$$S \subseteq \Omega_2 \bigcup \Omega_3 . \tag{1}$$

For any node i in $\Omega_1 \cup \Omega_3$, suppose the closest facility is located in node $F(i)$, and

$$F(i) \in S . \tag{2}$$

Then, the FLOP can be mathematically described as the following minimization problem

$$\min_{S \subseteq \Omega_2 \cup \Omega_3} \sum_{i \in \Omega_1 \cup \Omega_3} U(i) \times D_{\min}(i, F(i)), \tag{3}$$

where $D_{\min}(i,F(i))$ is the minimal distance from node i to node $F(i)$, and is calculated as

$$D_{\min}(i, F(i)) = \sum_{k=1}^{N_{R2CF}(i)-1} L(R_{CF,i}(k), R_{CF,i}(k+1)), \tag{4}$$

where $R_{CF,i}$ denotes the shortest route from node i to its closest facility, i.e., node $F(i)$, $R_{CF,i}(k)=n$ means node n is the kth node along the route $R_{CF,i}$, k=1,…,$N_{R2CF}(i)$, and n=1,…,N_N, and $N_{R2CF,i}$ tells how many nodes, including node i and its closest facility, are included in the route $R_{CF,i}$. Fig.1 gives an example of route network for the FLOP.

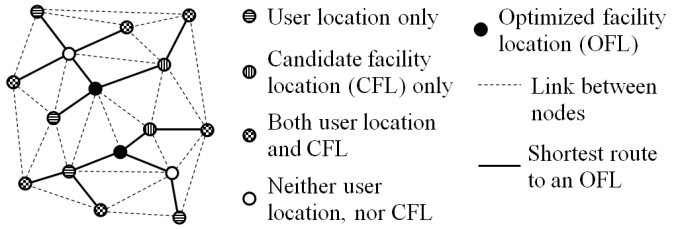


Fig.1 An example of facility location optimization problem (FLOP).

Suppose there are $N_{2\cup 3}$ nodes in $\Omega_2 \cup \Omega_3$ in total. Then the size of solution space for the FLOP is

$$C_{N_{2\cup 3}}^{N_F} = \frac{\prod_{i=0}^{N_F-1}(N_{2\cup 3}-i)}{\prod_{i=2}^{N_F} i} . \tag{5}$$

Basically, a larger $N_{2\cup 3}$ and a larger N_F mean a more complicated FLOP. For a given N_F, if all nodes in the network are available to set up facility, i.e., $N_{2\cup 3}=N_N$, then the FLOP has the maximal solution space, which makes the optimization the most difficult.

To address the above FLOP, firstly, we need to resolve a many-to-many route optimization problem, where, for two given sets of nodes, one as sources and the other as destinations (e.g., set $\Omega_2 \cup \Omega_3$ and set S), we must find the closest destination node for each source node. Fortunately, there are many mature and effective methods to resolve this problem (for example, see [17]), and they will not be discussed further in this paper. The difficulty in the FLOP is to find the best set of destinations for the many-to-many route optimization problem. This is obviously a combinatorial, NP-hard problem, and therefore, evolutionary computation methods such as GA are highly appropriate.

III. GA WITH SRHC FOR FLOP

A. From TRHC to SRHC

First of all, a brief review on the conventional receding horizon control (RHC) strategy in control engineering will be very useful. To distinguish it from the method proposed in this paper, the conventional RHC in dynamic control problems is

hereafter referred to as temporal receding horizon control (TRHC). TRHC, also known as model predictive control, has proved to be a highly effective online optimization strategy in the area of control engineering, and it exhibits many advantages with respect to other control strategies [14], [15]. It is easy for TRHC to handle complex dynamic systems with various constraints. It also naturally exhibits promising robust performance against uncertainties since the online updated information can be sufficiently used to improve the decision. Simply speaking, TRHC is an *N*-step-ahead online optimization strategy to deal with dynamic problems. In this framework, decision is made by looking ahead for *N* steps in terms of a given cost/criterion, and the decision is only implemented by one step. Then the implementation result is checked, and a new decision is made by taking updated information into account and looking ahead for another *N* steps.

Fig.2 illustrates the basic idea of TRHC by comparing it with some other optimization strategies in an intuitive way. The offline optimization strategy, as shown in Fig.2.(a), is clearly not suitable for dynamic environments. The conventional dynamic optimization process, as shown in Fig.2.(b), is often criticized for its poor real-time properties and poor performance under disturbances and/or uncertainties in dynamic environments. As illustrated in Fig.2.(c), thanks to the idea of a temporal receding horizon, the TRHC strategy provides a possible solution to the problems confronted by the conventional dynamic optimization strategy. A properly chosen temporal receding horizon can effectively filter out most unreliable information and reduce the scale of the problem. The latter is especially important for complex systems and time-consuming algorithms to satisfy the time limit on the online optimization process. TRHC has now been widely accepted in the area of control engineering [14], [15]. Attention has also been paid to applications of TRHC to areas such as management and operations research [18], [19]. Particularly, the TRHC strategy has recently been reported to be successfully integrated into population-based algorithms to tackle various dynamic NP-hard optimization problems [20]-[22].

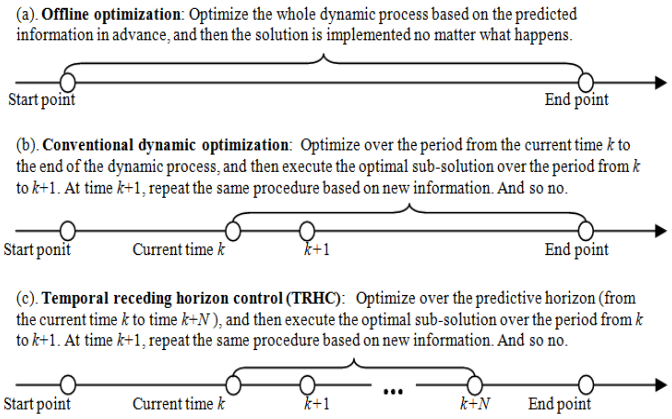


Fig.2 Illustration of temporal receding horizon control (TRHC).

Inspired by the fact that the success of the TRHC strategy largely results from decomposing a complex dynamic process into a serial of temporally associated sub-processes, here we are concerned with how to extend the basic idea of TRHC in order to decompose a large-scale static problem into a series of associated sub-problems (it should be noted that conventional partitioning methods decompose a static problem into a set of *separated* sub-problems). The question is then in what terms could sub-problems be associated in static environments? Basically, we need to create a problem-specific artificial space, project into the space all parts that compose a solution to the original static problem, and then design a spatial horizon which recedes in the space. As the spatial horizon recedes out, the value/status of each part will be optimized along together with all other parts that are within the current horizon scope. Once the values/statuses of all parts are optimized, a final solution to the original static problem is determined. Now, one can see that sub-problems will be spatially associated in the artificial space. Therefore, we call our new strategy for decomposing static problems as spatial receding horizon control (SRHC).

After an artificial space is designed and all parts that compose a solution are projected into the space, it is crucial to design a spatial horizon receding process to decompose the original static problem into a serial of spatially associated sub-problems. A basic spatial horizon receding process can be described as follows. Suppose a solution to a large-scale static problem is composed of *M* local parts. The SRHC strategy makes use of spatial structure (where positions indicate strength of influence between parts of a solution) to move from purely local, part-by-part, optimization to using information from the neighboring, sub-global context. An optimization algorithm is applied many times to determine the *M* parts in a solution. Starting with a specified part, the algorithm calculates at each time step the *N* new parts (usually $N<<M$), which are the most associated with the *decided* parts, (i.e., parts which have already been optimized in the previous iterations). Although the algorithm will optimize *N* parts each time, only the part that is the most associated with the decided parts will be added to the list of decided parts. The other *N*-1 parts will be discarded to be recalculated in later iterations. The algorithm keeps running until all *M* parts have been optimized. This leads to a general *N*-step-wide static problem partitioning method. Existing problem-partitioning methods may be considered as a one-step-wide SRHC strategy, i.e., each part of a solution is determined in an isolated manner; see for example [12]. In the generalized *N*-step-wide SRHC strategy, each part is calculated by referring to its most relevant surrounding parts. In other words, sub-global information is used in the determination of a local part. The extra information considered by the *N*-step-wide SRHC strategy can improve the quality of each part and that of the global solution.

B. FLOP in the SRHC framework

To apply the SRHC strategy to the FLOP, we need first to define the artificial space, spatial step and receding horizon for the FLOP. Fortunately, it is very straightforward in the case of FLOP. The artificial space is exactly the space where the route network of the FLOP is developed. Then we divide the artificial space into some equally sized rectangular areas, and

each of these is a spatial step. Then a receding horizon is composed of N_{HL} spatial steps. Preferably, the spatial steps of a receding horizon should be N_{HL} successive spatial steps in the artificial space. Actually, from the viewpoint of a separated local optimization (SLO) strategy, each spatial step defines a sub-problem. For the proposed SRHC strategy, a sub-problem is defined by the receding horizon. Suppose the original route network of the FLOP is divided into N_{SP} spatial steps. Starting from a certain spatial step (e.g., the top-left spatial step), we assign a serial number to each spatial step according to successive relationships between spatial steps. Then the first receding horizon is composed of spatial steps 1 to N_{HL}, which together define the first sub-problem to be resolved. After the first sub-problem is resolved, the sub-solution associated with the first spatial step is fixed (i.e., the first spatial step becomes a decided spatial step). Then the first spatial step is removed from the receding horizon, and a new spatial step (based on the serial number) is attached to the receding horizon, in order to compose a new receding horizon, and therefore defines a new sub-problem. In this way, the horizon is receding in the space of route network, until all spatial steps become decided spatial steps. Fig.3 illustrates how artificial space, spatial step and receding horizon are defined in a simplified FLOP, and how the FLOP is resolved as the horizon recedes spatial step by spatial step.

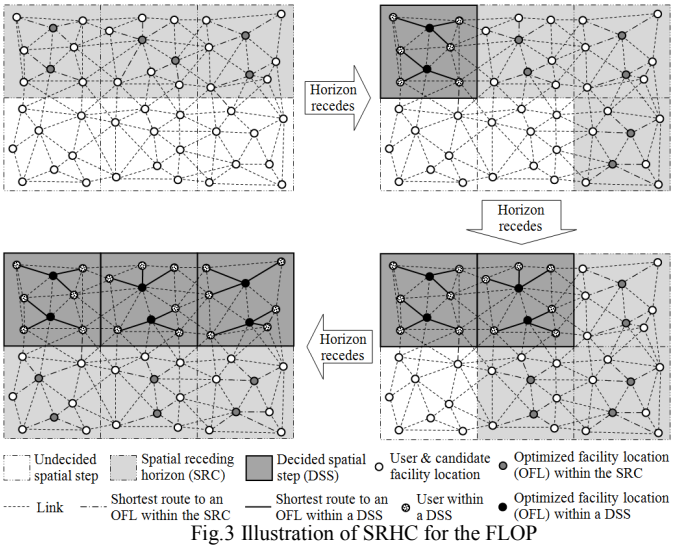


Fig.3 Illustration of SRHC for the FLOP

Now, we can mathematically reformulate the FLOP within the framework of SRHC. Under the SRHC strategy, the GA only needs to optimize the sub-problem defined by the current receding horizon. Let Ω_{CFL_RH} and Ω_{U_RH} be the set of CFL nodes and the set of user nodes covered by the current receding horizon, respectively, S_{RH} be a sub-solution associated with the receding horizon, and S_{RH} identifies N_{RH} facility locations. Then the sub-problem that GA needs to resolve for the current receding horizon is the following minimization problem

$$\min_{S_{RH}\subseteq\Omega_{CFL\_RH}} \sum_{i\in\Omega_{U\_RH}} U(i)\times D_{\min}(i,F(i)), \quad (6)$$

subject to Eq.(4) and

$$F(i)\in S_{RH}. \quad (7)$$

Basically, the above minimization problem aims to choose N_{RH} nodes as facility locations from the set Ω_{CFL_RH}, so that all user nodes in the set Ω_{U_RH} can access facilities most efficiently. Once a sub-problem is resolved, we then move the horizon back, i.e. update the receding horizon by removing its first spatial step and adding a new successive spatial step. The updated receding horizon represents a new sub-problem to be resolved. This process continues in the same vain and a series of minimization problems in the form defined by Eq.(6) is resolved, until a solution to the original FLOP is achieved.

Compared with the SLO strategy, the SRHC strategy is less short-sighted because it considers some successive spatial steps when calculating the sub-solution to a certain single spatial step (i.e., the first spatial step in the receding horizon, which will become a decided spatial step after the calculation). However, it is still very difficult, if not impossible, to completely avoid short-sighted performance under the SRHC strategy. In other words, since the sub-solution to a decided spatial step will not change during subsequent calculations, once the sub-solution is not a part of the global optimal solution to the original FLOP (which is likely to happen), the performance of SRHC is doomed no matter how well all subsequent calculations are conducted.

To further reduce short-sighted performance of SRHC, an effective measure is to dynamically develop a sub-solution pool for the decided spatial steps. In this pool, some quality sub-solutions for decided spatial steps during previous calculations are saved. Then, when resolving the sub-problem associated with the current receding horizon, besides choosing N_{RH} nodes as facility locations from the set Ω_{CFL_RH}, we also need to choose a sub-solution from the pool for decided spatial steps. Therefore, the problem defined by Eq.(6) needs to modified as follows. We set the pool size as N_{PS}, i.e., there are N_{PS} sub-solutions in the pool for decided spatial steps. Let Ω_{U_DSS} be the set of user nodes covered by decided spatial steps, and $S_{DSS}(k)$ the kth sub-solution in the pool for decided spatial steps. Then the minimization problem GA needs to resolve for the current receding horizon is described as following:

$$\min_{k=1,\ldots,N_{PS},S_{RH}\subseteq\Omega_{CFL\_RH}} \sum_{i\in\Omega_{U\_RH}\cup\Omega_{U\_DSS}} U(i)\times D_{\min}(i,F(i)) \quad (8)$$

subject to Eq.(4) and

$$F(i)\in S_{RH}\bigcup S_{DSS}(k). \quad (9)$$

On one hand, the introduction of the sub-solution pool for decided spatial steps can effectively reduce short-sighted performance of SRHC, because the sub-solution for decided spatial steps are now changeable during subsequent calculations. On the other hand, the recalculation of sub-solution for decided spatial steps is computationally cheap, because we do not need to re-choose nodes as facility locations in decided spatial steps, but directly choose from a pool of quality sub-solutions (actually simply choose a serial number in the pool). Regarding how to develop such a pool of quality sub-solutions for decided spatial steps, the population feature of GA gives a perfect solution. Every time after the calculation with GA, instead of choosing only the first best

sub-solution, we actually choose the first N_{PS} best different sub-solutions to develop/update the pool for decided spatial steps.

C. Design of GA with SRHC

A general illustration of GA with SRHC is given in Fig.4, from which one can see that a major difference between the proposed GA with SRHC and traditional GA is that a sub-solution pool for decided spatial steps needs to be developed and updated during the run of GA. The detailed descriptions of GA with SRHC can be found in [16]. In this sub-section, we mainly focus on how to design an effective GA with SRHC for the FLOP.

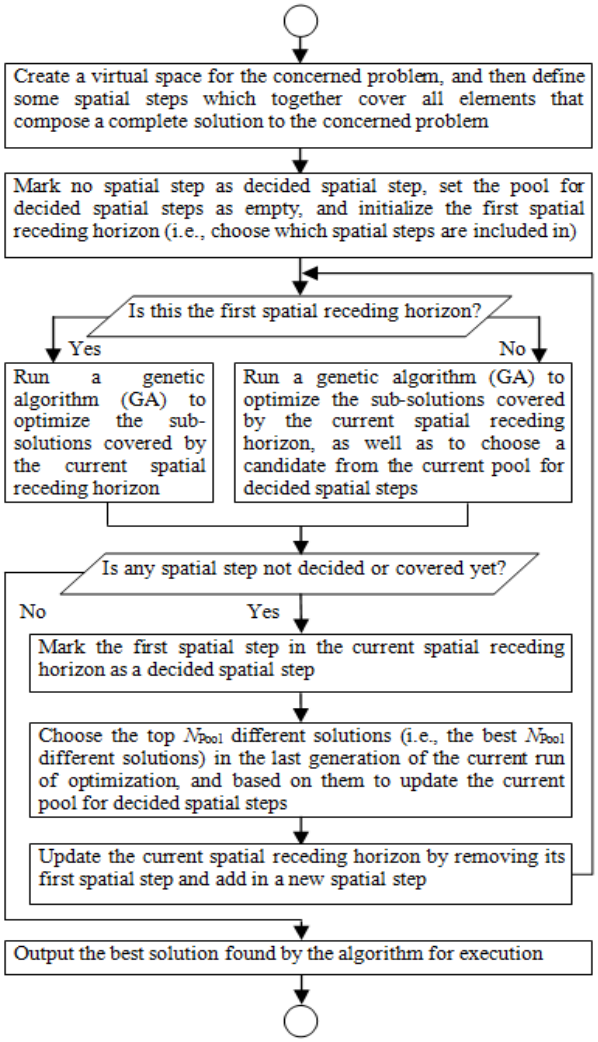


Fig.4 Flowchart of GA with SRHC.

Firstly, we need to decide on chromosome structure. Here we use an integer vector as chromosome as illustrated in Fig.5. The chromosome has N_{RH}+1 genes, which are classified into two categories. Category I has only one gene, i.e., the first gene $g(1)$, which indicates which sub-solution in the pool for decided spatial steps is chosen. The other N_{RH} genes belong to Category II, and they record which N_{RH} nodes in the set of Ω_{CFL_RH} are chosen as facility locations in the current receding horizon. It should be noted that, in traditional GA, with either the global optimization (GO) strategy or the SLO strategy, a chromosome only has genes of Category II.

One may wonder whether two different categories of genes in chromosome would make evolutionary operations more complicated. Actually, the chromosome structure as illustrated in Fig.5 imposes no difficulty in the design of evolutionary operations, i.e., mutation and crossover in GA.

The mutation is simple. Firstly we choose a gene randomly. If the first gene is chosen, then we randomly reset its value between 1 to N_{PS}. Otherwise, we choose for the gene a new node from the set Ω_{CFL_RH}. Here a new node means it is not included in the current chromosome.

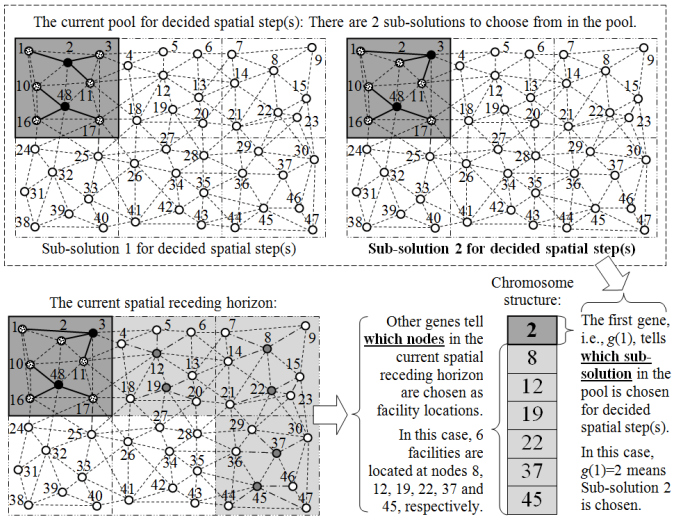


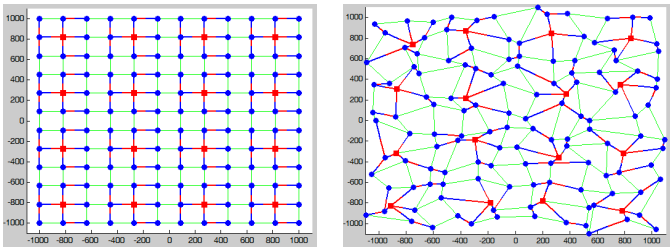
Fig.5 Chromosome structure employed by the GA with SRHC for the FLOP.

The crossover is also simple. For two given parent chromosomes, an offspring chromosome will randomly inherit their first genes. For other genes, we first sort out common nodes shared by parent chromosomes, and then pass them directly onto offspring. Suppose there are N_{CN} common nodes. Then there are still N_{RH}-N_{CN} genes of Category II which need to be filled. Therefore, we randomly choose N_{RH}-N_{CN} nodes from the non-common nodes of parent chromosomes.

IV. Simulation Results

In this section, we mainly compare the proposed SRHC strategy with the GO strategy and the separated local optimization (SLO) strategy (i.e., one-step-wide SRHC with no sub-solution pool for decided spatial steps), in order to assess how different optimization strategies will affect the performance of GA. The population of the GA is always set as N_{POP}=100 in this simulation. For GA with either the SLO strategy or the SRHC strategy, the number of evolved generations is set as N_{EG1}=100. Suppose an original route network for the FLOP is divided into N_{SP} sub-networks as spatial steps. Then the GA with the SLO strategy needs to run N_{SP} times, in order to find a solution to the original FLOP. This means the GA with the SLO strategy will generate $N_{POP}\times N_{EG1}\times N_{SP}$ chromosomes. Similarly, one may deduce that the GA with the SRHC strategy will generate $N_{POP}\times N_{EG1}\times (N_{SP}$-$N_{HL}$+1) chromosomes before a solution to the original FLOP can be found. Since the GA with the GO strategy only needs to run once to get a solution to the original FLOP, to

make a fair comparison, in this simulation, the number of evolved generations in the GA with the GO strategy is set as $N_{EG2}=N_{EG1}\times N_{SP}$. It should be nother that we here fix the population size and allow the GA with the GO strategy to have more evolved generations, rather than fixing the number of evolved generations whilst allowing a larger population. This is because: (i) the FLOP is usually an off-line optimization problem, which means computational time is not the most important issue; (ii) a larger population size demands more computer memory, which might be restricted by available computing hardware resources. Therefore, we fix the population size and allow a larger number of evolved generations.



(a) Grid network (b) Random network

Fig.6 Two examples of test cases.

There are two categories of route networks used in the simulation: grid networks and random networks. In this study, every node in a network is both user location and CFL, and this makes the FLOP most complicated, because, according to Eq.(5), for a given N_F, the solution space reaches the theoretical maximal size. In this study, each node has an average of 4 links to its neighbor nodes (just similar to a real-world route network), and for the sake of simplicity, has only 1 user. The reason for using a grid network is because the theoretical optimal solution for the FLOP can be easily deduced, and therefore the performances of different optimization strategies can be precisely assessed. For example, Fig.6.(a) gives a grid network of 144 nodes, which are evenly distributed in a rectangular area. Suppose we need to optimize the locations of 16 facilities in the network. Then, one can easily work out that the theoretical optimal locations are those red nodes in Fig.6.(a). Then by comparing with the outputs of different optimization strategies, we can clearly see how far such outputs are from the theoretical optimal solution. However, real-world route networks are far more than grid, and their node locations and links are rather random, just like illustrated in Fig.6.(b), which represent the second category of networks used in the simulation. In general, it is very difficult, if not impossible, to deduce the theoretical optimal solution in a network of the second category, i.e., a random network. The reason to use random networks is because they can give us a better clue regarding the applicability of different optimization strategies in a real-world environment. In the simulation, for each category of networks, there are 5 test scenarios with different N_N, N_L and N_F values. Table 1 defines all test scenarios in the simulation. For each random test scenario, 100 networks are randomly generated. For each network, GA with a certain optimization strategy is applied for 100 times. For the SLO strategy and the SRHC strategy to apply, we need to define spatial steps. In the simulation, each spatial step covers a rectangular area of 600×600 (spatial distance unit)^2 as used in Fig.6. Therefore, for instance, for either case in Fig.6, the network is divided into 16 spatial steps. Under the SLO strategy, the GA needs to run 16 times, and each time only considers those nodes located within a given spatial step. Under the SRHC strategy in this study, the horizon is set to have 3 spatial steps, it starts from the top-left spatial step, recedes to the right-hand side by one spatial step each time. When the horizon reaches the right-hand side, it goes down one spatial step, and then recedes to the left-hand side. The horizon keeps receding until it reaches the bottom-right spatial step.

TABLE I
DEFINITION OF TEST SCENARIOS

Test scenario		N_N	N_L	N_F
Grid network	Grid1	36	60	4
	Grid2	144	264	16
	Grid3	324	612	36
	Grid4	576	1104	64
	Grid5	900	1740	100
Random network	Rand1	36	60	4
	Rand2	144	264	16
	Rand3	324	612	36
	Rand4	576	1104	64
	Rand5	900	1740	100

Some relevant key average results are listed in Table 2 and Table 3 to assess and compare the performance. In Table 2 and Table, L_{ASR} is the length of average shortest routes, and basically a smaller L_{ASR} means a better solution to the FLOP; R_{FTOS} is the ratio for an optimization strategy to find the theoretical optimal solution based on its 100 runs, and R_{FTOS} is only useful for grid network scenarios; Since for random network scenarios the theoretical optimal solution is not available, $R_{5\%}$ is then used to indicate how many percentage of the first 5% best solutions (of all three strategies) are found by a certain strategy, and a larger $R_{5\%}$ implies that a strategy is more advantageous. From Table 2 and Table 3, one may make the following observations:

- Overall, SLO and SRHC are better than GO, particular in complicated test cases, no matter which category of networks used.
- GO has the worst performance robustness against the change in the problem scale. No matter with which category of networks, the value of L_{ASR} grows very significantly as the problem scale increases. In the grid network scenarios, GO can barely find the theoretical optimal solution when the problem scale is large (e.g., in the cases Grid 3, Grid4 and Grid5). In the random network scenarios, when the problem scale is large, then almost all the first 5% best solutions are found by either SLO or SRHC.
- SLO has a fairly stable performance, which might imply that for the FLOP, by nature, local optimality has a good relationship with the global optimality. In other words, the simulation results of SLO might suggest that, in the FLOP, as long as all sub-problems are optimized locally, then the global optimality may be achievable.

- SRHC achieves the best performance against the change in the problem scale, particularly in complicated cases such as Grid3 to Grid 5, Rand4 and Rand 5. This may suggest that GO sometimes still has short-sighted performance, while SRHC can effectively overcome such limited performance.
- From the grid network scenarios, one may see that SLO and SRHC can both always effectively identify the theoretical optimal solution, and therefore both have very stable performance against the change in the problem scale. However, in the random network scenarios, the performance of either GO or SRHC degrades slightly and gradually. This suggests the random distribution of nodes complicates the problem, and therefore there is often a gap between the theoretical optimal solution and the found best solution.

TABLE II
TEST RESULTS WITH GRID NETWORKS

			Grid1	Grid2	Grid3	Grid4	Grid5
GO	L_{ASR}	Mean	242.4	244.8	251.2	289.5	348.1
		SD	0.0	2.2	25.6	35.1	48.3
	R_{FTOS}		1.00	0.93	0.48	0.24	0.07
SLO	L_{ASR}	Mean	242.4	242.4	242.4	242.4	242.4
		SD	0.0	0.0	0.0	0.0	0.0
	R_{FTOS}		1.00	1.00	1.00	1.00	1.00
SRHC	L_{ASR}	Mean	242.4	242.4	242.4	242.4	242.4
		SD	0.0	0.0	0.0	0.0	0.0
	R_{FTOS}		1.00	1.00	1.00	1.00	1.00

TABLE III
TEST RESULTS WITH RANDOM NETWORKS

			Rand1	Rand2	Rand3	Rand4	Rand5
GO	L_{ASR}	Mean	236.0	241.4	262.9	284.7	366.3
		SD	0.0	4.6	19.9	29.8	52.7
	$R_{5\%}$		0.33	0.08	0.00	0.00	0.00
SLO	L_{ASR}	Mean	236.0	238.1	240.2	243.8	252.1
		SD	0.0	3.8	9.2	13.7	13.5
	$R_{5\%}$		0.33	0.47	0.50	0.49	0.440
SRHC	L_{ASR}	Mean	236.0	238.9	240.9	240.5	242.7
		SD	0.0	4.0	8.6	13.1	15.8
	$R_{5\%}$		0.33	0.45	0.50	0.51	0.56

V. CONCLUSION

This paper describes an effective genetic algorithm (GA) for the facility location optimization problem (FLOP), where a given number of facilities need to be deployed in a route network, so that the users distributed in the network can access the facilities the most efficiently. This work includes two major contributions: (i) introduction of a novel spatial receding horizon control (SRHC) strategy to partition the FLOP; (ii) design of a GA within the SRHC framework to optimize facility locations. A preliminary simulation study shows that the SRHC and the GA are perfect matched to each other, and their combination delivers an effective and efficient method, which has a good potential of resolving the large scale FLOP. Future work may include: (i) Conducting more comprehensive simulation to analyze the influence of SRHC parameters on the performance of GA; (ii) Comparing with more other relevant methods for the FLOP; (iii) Extending the current preliminary results to some real-world FLOPs, such as distributing chain stores, organizing earthquake shelters, and distribution logistic stations; (iv) Investigating the generalization of the SRHC scheme, in order to that it applies to various large-scale problems in addition to the FLOP.

REFERENCES

[1] N. Megiddo and K. Supowit, "On the complexity of some common geometric location problems", *SIAM J. Comput.* 13(1), 182-196, 1984.

[2] M. T. Gastner and M.E.J. Newman, "Optimal design of spatial distribution networks", Physical Review E, 74, 016117, 2006.

[3] D.S. Hochbaum, Approximation Algorithms for NP-hard Problems, PWS, Boston, 1997.

[4] F.Y. Hu, W. Xu and X. Li, "A modified particle swarm optimization algorithm for optimal allocation of earthquake emergency shelters", International J. Geogr. Infor. Sci., 26(9), 1643-1666, 2012.

[5] J. Aerts and G.B. M. Heuvelink, "Using simulated annealing for resource allocation", International J. Geogr. Infor. Sci., 16, 571–587. 2002.

[6] J. W. Billheimer and P. Gray, "Network design with fixed and variable cost elements", Transp. Sci. 7, 49, 1973.

[7] M. Los and C. Lardinois, "Combinatorial programming, statistical optimization and the optimal transportation network problem", Transp. Res., Part B: Methodol. 16, 89, 1982.

[8] N.M. Bradford and R. Sen, "Multi parameter assessment guide for emergency shelters: disaster relief applications". Journal of Performance of Constructed Facilities, 19, 108–116, 2005.

[9] X.W. Chen, J.W. Meaker, and F.B. Zhen, "Agent-based modeling and analysis of hurricane evacuation procedures for the Florida Keys", Natural Hazards, 38, 321–338, 2006.

[10] D. Thierens,"Scalability Problems of Simple Genetic Algorithms", *Evolutionary Computation*, vol. 7, no. 4, pp. 331–352, 1999.

[11] E. Cantu-Paz and D.E. Goldberg, "On the Scalability of Parallel Genetic Algorithms", *Evolutionary Computation*, vol. 7, no. 4, pp. 429–449, 1999.

[12] G. Colombo and S.M. Allen, "Problem decomposition for Minimum Interference Frequency Assignment", *Proc. of the IEEE Congress in and Evolutionary Computation*, Singapore, 2007.

[13] S. Tsutsui, Y. Fujimoto and A. Ghosh, "Forking Genetic Algorithms: GAs with Search Space Division Schemes", *Evolutionary Computation*, vol. 5, no. 1, pp. 61–80, 1997.

[14] D.W. Clarke, *Advances in Model-based Predictive Control*, Oxford University Press, 1994.

[15] J.M. Maciejowski, *Predictive control with constraints*. Marlow: Personal Education Limited, 2002.

[16] X.B Hu and M.S Leeson, "Evolutionary Computation with Spatial Receding Horizon Control to Minimize Network Coding Resources ", *The Scientific World Journal*, in press, 2014.

[17] E.J. Taft and Z. Nashed, *Dynamic Programming: Foundations and Principles*, 2nd Edition, CRC Press, Taylor & Francis Group, 2011.

[18] S. Chand, V.N. Hsu, and S. Sethi, "Forecast, solution, and rolling horizons in operations management problems: a classified bibliography", *Manufacturing & Service Operations Management*, vol.4, no.1, pp.25-43, 2002.

[19] B.De Schutter and T. Van Den Boom, "Model predictive control for max-plus-linear discrete event systems", *Automatica*, vol.37, no.7, pp. 1049-1056, 2001.

[20] X.B. Hu and W.H. Chen, "Genetic Algorithm Based on Receding Horizon Control for Arrival Sequencing and Scheduling", *Engineering Applications of Artificial Intelligence*, 18, 633-642, 2005.

[21] X.B. Hu, W.H. Chen, and E. Di Paolo, "Multi-Airport Capacity Management: Genetic Algorithm with Receding Horizon", *IEEE Transaction on Intelligent Transportation System*, 8, 254-263, 2007.

[22] Z.H. Zhan, J. Zhang, Y. Li, O. Liu, S.K. Kwok, W.H. IP and O. Kaynak, "An efficient Ant Conoly System Based on Receding Horizon Control for Aircraft Arrival Sequencing and Scheduling Problem", *IEEE Transaction on Intelligent Transportation System*, 11, 399-412, 2010.