An Iterative Local Search Based hybrid algorithm for the service area problem

Authors: Kong Yunfeng, Kong Yunfeng

Date: 2021-05-19T00:00:00+00:00

Abstract

This article presents a hybrid algorithm for the service area problem. The design of service areas is one of the essential issues in providing efficient services in both the public and private sectors. For a geographical region with a number of small spatial units, the service area problem is to assign the service-demand units to the service-supply units such that each facility has a service area. The basic criteria for the service areas are the highest service accessibility, the contiguous service areas, and that the service demand does not exceed the service supply in each service area. A hybrid algorithm for the service area problem is proposed by extending iterative local search (ILS) algorithm with three schemes: population-based ILS, variable neighborhood descent (VND) search, and set partitioning. The performance of the algorithm was tested using 60 well-designed instances. Experimentation showed that the instances could be solved effectively and efficiently. The solutions found by the hybrid algorithm approximate optimal solutions or the lower bounds with an average gap of 0.15%.

Full Text

Preamble

An Iterative Local Search Based Hybrid Algorithm for the Service Area Problem

Yunfeng Konga,b
a Key Laboratory of Geospatial Technology for the Middle and Lower Yellow River Regions, Ministry of Education, Henan University, Kaifeng, China
b College of Environment and Planning, Henan University, Kaifeng, China

Abstract: This article presents a hybrid algorithm for the service area problem. The design of service areas represents one of the essential issues in providing efficient services across both public and private sectors. For a geographical region composed of numerous small spatial units, the service area problem involves

assigning service-demand units to service-supply units such that each facility obtains a service area. The fundamental criteria for service areas include maximizing service accessibility, ensuring contiguous service areas, and guaranteeing that service demand does not exceed service supply within each area. We propose a hybrid algorithm that extends the iterative local search (ILS) framework with three enhancement schemes: population-based ILS, variable neighborhood descent (VND) search, and set partitioning. The algorithm' s performance was evaluated using 60 carefully designed test instances. Experimental results demonstrate that the instances can be solved effectively and efficiently, with solutions produced by the hybrid algorithm approximating optimal solutions or lower bounds with an average gap of merely 0.15%.

Keywords: service area problem; hybrid algorithm; local search; set partitioning

Introduction

This article addresses the service area problem (SAP). The design of service areas constitutes a critical challenge in delivering efficient services in both public and private sectors (Daskin 2011). The delineation of service areas for schools (Ferland and Guenette 1990; Schoepfle and Church 1991; Caro et al. 2014), hospitals and healthcare centers (Pezzella et al. 1981; Jacobs et al. 1996; Emiliano et al. 2017; Hu et al. 2018), disaster shelters (Li et al. 2008; Hu et al. 2014), green energy resources (Yanik et al. 2016), and various other facilities can be generalized as a contiguity-constrained capacitated facility SAP. For a geographic area comprising small spatial units, the SAP requires assigning service-demand units to service-supply units such that each facility receives a service area while satisfying specific criteria. The fundamental design criteria include maximizing service accessibility, ensuring service area contiguity, and preventing service demand from exceeding service supply within each area. Service accessibility can be evaluated by the total travel distance from demand units to their assigned supply units, with shorter travel distances generally preferred. Contiguous service areas are also necessary to satisfy policy and management requirements. Additionally, the total service demand within each service area must not exceed its service capacity.

The SAP can be formulated as a contiguity-constrained generalized assignment problem (GAP) that aims to minimize total travel distance while satisfying facility capacity constraints and service area contiguity requirements. The GAP is known to be NP-hard, and spatial contiguity constraints pose significant modeling and computational challenges for geographic problems (Duque et al. 2011). Consequently, researchers have proposed various exact and heuristic methods to solve the SAP.

Mixed integer linear programming (MILP) has been widely employed for districting problems. However, due to their computational complexity, many MILP models developed since the 1960s have simplified the problem by omitting cer-

tain districting constraints such as contiguity and unit integrity, requiring post-hoc adjustments to repair split units and fragmented districts. Such approaches have been applied to school districting (Koenigsberg 1968; Franklin and Koenigsberg 1973; Caro et al. 2004), political districting (Hess et al. 1965; Hojati 1996; George et al. 1997; Li et al. 2007), and territory design problems (Kalcsics et al. 2005). While these modeling and repair approaches could be adapted for the SAP, they may struggle to find satisfactory solutions when spatial mismatches exist between service demands and supplies.

A second class of MILP models explicitly incorporates district contiguity. Three formulation types for district contiguity—tree-based, order-based, and flow-based—can be embedded within assignment or location-allocation models for districting problems (Shirabe 2005; Duque et al. 2011). Nemoto and Hotta (2003) proposed a MILP model for political districting that considers district contiguity but does not guarantee district compactness. Salazar-Aguilar et al. (2011) developed a bi-objective model for designing commercial territories that incorporates constraints on sales volume balance and district contiguity, though solving instances with 150 units and 6 territories required four hours of computation time. Plane et al. (2019) introduced a minimum inter-person separation model for political districting, which could solve 6$\times$6 grid instances optimally or near-optimally in 6.30–53,632.22 seconds. Kong et al. (2019) formulated a center-based allocation model with flow-based contiguity constraints for districting problems, demonstrating that instances could be solved efficiently when district centers were provided by a K-medoids algorithm. Since service-supply units are predetermined in the SAP, this model could be modified to address the problem.

A third exact approach employs set partitioning. Given a large set of feasible districts generated by constructive or heuristic methods, the set partitioning problem (SPP) model selects an optimal subset of districts to form a districting solution (Garfinkel and Nemhauser 1970; Nygreen 1988; Mehrotra et al. 1998). Results indicate that SPP models can solve small instances exactly and produce satisfactory results for larger problems. To obtain high-quality solutions, researchers have suggested various techniques for generating promising candidate districts. Kong et al. (2017) proposed an iterated local search algorithm with set partitioning for school districting, demonstrating that the SPP model could identify superior solutions from historical districts discovered during local search. However, this algorithm was not comprehensively tested on general SAP instances.

Local-search-based and population-based heuristics represent mainstream approaches for solving the SAP. Starting from an initial solution, local-search algorithms such as greedy search, simulated annealing, tabu search, old bachelor acceptance, and greedy randomized adaptive search procedure (GRASP) iteratively improve the incumbent solution through neighborhood search (Li et al. 2008; Ricca and Simeone 2008; Ríos-Mercado and Fernandez 2009). The neighborhood space is explored using one or more search operators, with the

incumbent solution updated according to specific acceptance criteria.

The one-unit shift operator is widely used, moving a boundary unit to a neighboring district while preserving the connectivity of its original district (Tavares-Pereira et al. 2007; Ricca and Simeone 2008; Xiao 2008; Li et al. 2008; Liberatore and Camachocollados 2016). Butsch et al. (2014) introduced three operators for districting problems: shift, double shift, and swap. Kong et al. (2017) proposed more complex local search operators for school districting, though algorithms employing expensive local search operators have not been thoroughly evaluated in existing literature.

Population-based heuristics have gained popularity for districting problems in recent research. Evolutionary algorithms (Forman and Yue 2003; Bergey et al. 2003; Bacao et al. 2005; Tavarespereira et al. 2007; Chou et al. 2007; Tavares-Pereira et al. 2007; Xiao 2008; Chou 2011; Hu et al. 2014; Liu et al. 2016) maintain and improve multiple candidate solutions using mechanisms such as crossover, mutation, and selection. Crossover operations for districting problems typically overlay two individuals and repair the resulting fragmented areas (Chou et al. 2007; Datta et al. 2008; Xiao 2008; Liu et al. 2016). Mutation operations generally move boundary units to neighboring districts with low probability. To accelerate convergence and improve solution quality, Tavarespereira et al. (2007) proposed an evolutionary algorithm with local search for districting, balancing population diversity with search intensity in a hybrid framework. Additionally, nature-inspired metaheuristics such as scatter search (Salazar-Aguilar et al. 2012) and artificial bee colony (Rincón-García et al. 2015) have been applied to districting problems.

Existing solution methods for districting problems exhibit several limitations. First, while current algorithms can solve artificial or real-world instances, their performance has not been sufficiently evaluated on benchmark instances, leaving the optimality of obtained solutions uncertain. Second, innovative operations research concepts such as hybrid metaheuristics, matheuristics, learning-based adaptive algorithms, and hyper-heuristics have not been adequately exploited for districting problems. Third, although some studies suggest that districting problems are difficult to solve directly using exact methods, advances in MILP —particularly embedded heuristics in mixed-integer programming optimizers— have enabled state-of-the-art solvers to tackle increasingly challenging problems (Lodi 2017). Designing new algorithms based on modern MIP solver frameworks may offer a promising approach for districting problems.

This article presents a hybrid algorithm for the SAP, developed by extending the iterative local search (ILS) framework with three enhancement schemes: population-based ILS, variable neighborhood descent (VND) search, and set partitioning. The algorithm was tested on 60 carefully designed problem instances, demonstrating its effectiveness and efficiency. Compared to exact solutions, the hybrid algorithm' s results approximate optimal solutions or lower bounds with an average gap of 0.15%. To the best of our knowledge, this represents the first application of population-based ILS with VND search and set partitioning to

the SAP.

2. Problem Formulation

Consider a geographic area represented by a set of n small units, denoted as V = {1, 2, ···, n}. Each unit i has a service demand p_i. Let S = {s_1, s_2, ···, s_K} (S V) be a set of K service-supply units, where each unit s_k has a service capacity q_k. Let d_{ik} represent the distance between demand unit i and supply unit k.

Define a_{ij} to indicate whether units i and j share a border, and let N_i be the set of units adjacent to unit i (N_i = {j | a_{ij} = 1}).

The SAP can be formulated as a contiguity-constrained GAP using a mixed integer linear programming model. Service area contiguity is ensured through a network flow model (Shirabe 2005; Duque et al. 2011). Let x_{ik} {0,1} denote whether unit i is assigned to supply unit s_k, H_k (H_k ≥ 0) represent the service overload of supply unit s_k, and f_{ijk} (f_{ijk} ≥ 0) denote the flow from unit i to unit j in service area k. The SAP model is formulated as follows:

Minimize

$$\alpha \sum_{k \in S} H_k + \sum_{i \in V} \sum_{k \in S} p_i d_{ik} x_{ik}$$

Subject to:

$$\sum_{k \in S} x_{ik} = 1, \quad \forall i \in V$$

$$\sum_{i \in V} p_i x_{ik} \leq q_k + H_k, \quad \forall k \in S$$

$$f_{ijk} \leq (n - K) x_{ik}, \quad \forall i \in V, j \in N_i, \forall k \in S$$

$$f_{ijk} \leq (n - K) x_{jk}, \quad \forall i \in V, j \in N_i, \forall k \in S$$

$$\sum_{j \in N_i} f_{jik} - \sum_{j \in N_i} f_{ijk} \geq x_{ik}, \quad \forall i \in V \setminus S, \forall k \in S$$

$$x_{ik} \in \{0, 1\}, \quad \forall i \in V, k \in S$$

$$f_{ijk}, H_k \geq 0, \quad \forall i \in V, j \in N_i, k \in S$$

The objective function (1) minimizes total travel distance from service demand units to their assigned supply units while penalizing service overloads through a sufficiently large coefficient α. Constraints (2) ensure each unit i is assigned to exactly one service-supply unit. Constraints (3) represent soft capacity constraints. Constraints (4), (5), and (6) implement flow-based contiguity formulations, adapted from Duque et al. (2011). For adjacent units i and j, if both are assigned to service-supply unit s_k (x_{ik} = 1 and x_{jk} = 1), flow may exist from unit i to j in area k (f_{ijk} ≥ 0); otherwise, no flow occurs between them (f_{ijk} = 0), as enforced by constraints (4) and (5). Constraints (6) ensure that one unit of flow is created at each non-service-supply unit i when assigned to service-supply unit s_k. Together, constraints (4), (5), and (6) enforce that all flows generated within service area k must converge at the sink unit k.

Constraints (7) and (8) impose variable restrictions. Notably, the model employs soft constraints (3) on facility capacities. Service overloads (H_k) are penalized in the objective function, offering two advantages: first, soft constraints facilitate efficient identification of feasible solutions by the MIP optimizer while guiding the search toward preferred solutions; second, hard capacity constraints may render some instances infeasible (e.g., when total demand exceeds total supply or when assignment and capacity constraints cannot be simultaneously satisfied), a situation that soft constraints can handle effectively.

3.1 Iterative Local Search

We propose a hybrid algorithm for the SAP based on the iterative local search (ILS) metaheuristic. ILS is a simple, easy-to-implement, and highly effective approach for discrete optimization problems (Lourenço et al. 2010). The method begins with an initial solution and iteratively improves it through alternating phases of local search and perturbation. Local search heuristics intensively explore the solution space, but iterative search may become trapped in local optima far from the global optimum. ILS escapes these local optima by applying perturbations to the current local minimum. The basic ILS procedure is illustrated in Algorithm 1.

Algorithm 1: Iterated Local Search
1. s_0 = GenerateInitialSolution();
2. s = LocalSearch(s_0);
3. Repeat until termination condition met:
4. s = Perturbation(s);
5. s* = LocalSearch(s);
6. if f(s) < *f(s) then* $s = s$;
7. Output s.

3.2 Initial Solution

Initial solutions are generated using a transportation problem (TP) model, which assigns demand from each unit to one or multiple supply units at minimum travel cost. The TP can be efficiently solved by a MIP solver due to its bipartite structure. Small random coefficients _{ik} (| _{ik}| < 0.02) are incorporated into the objective function to obtain randomized solutions.

Minimize

$$\sum_{i \in V} \sum_{k \in S} (1 + \epsilon_{ik}) p_i d_{ik} x_{ik}$$

Subject to:

$$\sum_{k \in S} x_{ik} = 1, \quad \forall i \in V$$

$$\sum_{i \in V} p_i x_{ik} \leq q_k, \quad \forall k \in S$$

$$x_{ik} = [0, 1], \quad \forall i \in V, k \in S$$

The TP solution requires repair. First, some units may be split across multiple service areas; for each split unit, the algorithm assigns it to the area containing the largest portion. Second, some service areas may be non-contiguous. Non-contiguous areas are repaired by removing fragmented units and reassigning them to neighboring areas. Note that solutions from region growth may violate service capacity constraints. The algorithm permits infeasible solutions, but those with service overloads are penalized through the objective function (1) during subsequent local search, guiding them toward feasibility.

3.3 Local Search

The design of local search operators is critical for repeatedly improving the incumbent solution. The one-unit shift operator is widely used, moving a boundary unit to a neighboring district while maintaining the connectivity of its original district (Tavares-Pereira et al. 2007; Ricca and Simeone 2008; Xiao 2008; Li et al. 2008; Liberatore and Camachocollados 2016). Butsch et al. (2014) introduced three operators for districting problems: shift, double shift, and swap.

Our algorithm employs two local search operators that attempt to move one or more boundary units to neighboring service areas. Only feasible moves are permitted, as moving a boundary unit from its original area may render that area non-contiguous. The one-unit shift and two-unit shift operators are illustrated in Figure 1 [Figure 1: see original paper].

The one-unit shift moves boundary unit i to a neighboring area k, as outlined in Algorithm 2. The algorithm selects all boundary units, shuffles them randomly, and attempts to move each unit from its original area to a neighboring area. A move is accepted if the original area remains connected and the new solution improves upon the incumbent.

Algorithm 2: One-Unit Shift

```
1. ulist = BoundaryUnits(s);
2. ulist = Shuffle(ulist);
3. for each i in ulist:
4.     for each k in NeighboringAreas(i):
5.         s  = MoveUnit(i, k);
6.         if IsContiguous(s ) and f(s ) < f(s) then s = s ;
7. Output s.
```

The two-unit shift moves boundary unit i to a neighboring area k while simultaneously moving boundary unit j in area k to one of its neighboring areas (Algorithm 3). In other words, one unit enters area k while another unit leaves it. Unlike the one-unit shift involving two areas, a two-unit shift typically involves three areas; swapping two units between adjacent areas represents a special case involving only two areas.

Algorithm 3: Two-Unit Shift

```
1. ulist = BoundaryUnits(s);
2. ulist = Shuffle(ulist);
3. for each i in ulist:
4.     for each k in NeighboringAreas(i):
5.         for each j in BoundaryUnits(k):
6.             for each l in NeighboringAreas(j)  {OriginalArea(i)}:
7.                 s  = MoveTwoUnits(i, k, j, l);
8.                 if IsContiguous(s ) and f(s ) < f(s) then s = s ;
9. Output s.
```

These operators exhibit different search spaces and computational complexities of O(Kn) and O(Kn^2), respectively. However, the actual number of possible moves is typically much smaller, as only boundary units can be moved to neighboring areas, and each boundary unit has only a few neighboring areas available.

3.4 Perturbation

Perturbation is a key component of the ILS algorithm. Local search operators may quickly converge to local optima. A ruin-and-recreate procedure effectively perturbs SAP solutions to escape these optima. Multiple ruin methods exist: deleting boundary units, deleting all units in certain areas, or deleting units in a connected region, followed by reassigning deleted units to their nearest facilities. An alternative approach moves boundary units to neighboring areas, typically requiring repair since perturbed solutions may contain non-contiguous

areas. Although perturbation may worsen the objective value, subsequent local search can significantly improve solution quality.

Perturbation strength critically affects ILS performance. Excessively strong perturbation tends to generate much worse solutions, reducing the probability of finding better solutions. Conversely, overly weak perturbation often allows the solution to return to its original local optimum after local search. Appropriate perturbation strength maintains search diversification and facilitates discovery of superior solutions.

Our ILS algorithm randomly employs four perturbation schemes: ruin boundary units and recreate, ruin service areas and recreate, ruin a connected region and recreate, and move boundary units to neighboring areas. Let "strength" denote the perturbation intensity as a percentage of solution components (e.g., strength% of boundary units, service areas, or all units). In all schemes, perturbed units/areas are selected randomly.

3.5 Set Partitioning

Let Ω denote the set of historical service areas identified during the ILS algorithm. Each area i has an objective value c_i and contains a set of units U_i. Let subset Ω_j = {i | i Ω, j U_i}. The SPP model (13)-(15) selects a subset of service areas from Ω that minimizes total objective value while covering all units in set V. The decision variable x_i indicates whether candidate area i is selected.

Minimize

$$\sum_{i \in \Omega} c_i x_i$$

Subject to:

$$\sum_{i \in \Omega_j} x_i = 1, \quad \forall j \in V$$

$$x_i \in \{0, 1\}, \quad \forall i \in \Omega$$

Set partitioning can be incorporated into the ILS algorithm as a post-processing procedure to improve SAP solutions. Although the SPP is NP-hard, modern MIP optimizers can efficiently solve instances with considerable sizes of Ω and V.

3.6 Contiguity of Service Areas

Several algorithms are commonly used to maintain service area contiguity during local search and perturbation. The first algorithm checks service area connectivity by attempting to construct a spanning tree using all units in an area; if

successful, the area is connected (Liu et al. 2016). The second algorithm identifies fragmented units using the same spanning tree approach: for each service area, a spanning tree is gradually constructed from the supply unit, and any units that cannot be incorporated into the tree are deemed fragmented. The third algorithm repairs solutions with non-contiguous service areas by deleting fragmented units and gradually reassigning them to neighboring areas.

3.7 A Hybrid Algorithm

We introduce a hybrid algorithm based on ILS for solving the SAP. The standard ILS algorithm is enhanced through three schemes: population-based search, variable neighborhood descent (VND) search, and set partitioning. The proposed algorithm is outlined in Algorithm 4.

Algorithm 4: Hybrid Algorithm for SAP
Parameters: population size (psize), minimum dissimilarity between any two solutions (mindiff), perturbation strength (strength), maximum number of consecutive loops without best solution improvement (mloops), time limit for set partitioning (t).

1. P = GenerateInitialSolutions(psize); (Section 3.2)
2. pool = ;
3. s_{best} = Best(P);
4. notImprove = 0;
5. Repeat until termination condition met:
   6. Select solution s from P randomly;
   7. s = Perturbation(s, strength); (Section 3.4)
   8. s* = VNDsearch(s);
   9. if f(s*) < f(s_{best}) then s_{best} = s*, notImprove = 0;
   10. else notImprove += 1;
   11. pool = UpdateServiceAreaPool(pool, s*);
   12. *P = UpdatePopulation(P, s*, mindiff);
6. s = SetPartitioning(pool, t); (Section 3.5)
7. Output s.

The hybrid algorithm maintains multiple candidate solutions, representing a population-based approach rather than the standard single-solution ILS. While perturbation schemes ensure diversification, selecting appropriate strength parameters remains challenging, as optimal perturbation strength may depend on both the scheme and instance characteristics. The population-based ILS extension enhances diversification by maintaining a set of elite, diverse solutions.

In step 12, the population is updated based on solution quality and similarity, selecting only elitist and dissimilar solutions to form the new population. Dissimilarity between any two solutions is defined as the percentage of demand units assigned to different facilities. The updated population must satisfy a minimum dissimilarity threshold (mindiff). Although the new population size may occasionally fall below popsize, subsequent perturbation can generate additional dissimilar solutions.

The second enhancement replaces simple local search with VND search. Simple local search improves solutions by sequentially applying one-unit and two-unit shifts, whereas VND systematically explores different neighborhood structures through deterministic neighborhood changes until no further improvement is possible. VND enables ILS to thoroughly explore the solution space and reach local optima before perturbation moves the search to new regions.

The third scheme improves ILS solutions through set partitioning. During each ILS iteration, all service areas from solution s* are recorded in the pool, including their constituent units and objective values. After ILS completes, thousands of areas may be accumulated in the pool, and the SPP model (Section 3.5) can select a superior solution from these candidate service areas.

4.1 Experiment Settings

Three study areas—ZY, GY, and GY2 from Kong et al. (2019)—were adapted for service area delineation, containing 324, 297, and 1,276 spatial units, respectively. Figure 2 [Figure 2: see original paper] illustrates the spatial units and service demand in each unit.

Using these study areas, we prepared 60 SAP instances to test the hybrid algorithm. For each area, 20 instances were designed as follows: (1) 36, 44, and 24 units were designated as candidate service-supply units with capacities of 9,195, 42,326, and 1,324,763, respectively; (2) a random number of facilities were selected from candidates—13–17, 37–41, and 18–22 service-supply units for the three areas; (3) facility capacities were adjusted proportionally to achieve supply-demand ratios of approximately 1.15 and 1.03; (4) facilities were selected by solving a capacitated p-median problem, with capacities adjusted to maintain the same supply-demand ratios.

Table 1 summarizes instance attributes. The 20 instances per study area are classified into four sets: random facility locations with higher supply-demand ratio (A), p-median facility locations with higher ratio (B), random locations with lower ratio (C), and p-median locations with lower ratio (D). This design yields diverse instances varying in demand units, facility count, facility locations, supply-demand ratio, and average units per facility.

The proposed algorithm was implemented in Python and executed using PyPy 6.0, a fast and compliant Python implementation (see http://pypy.org). Each instance was solved ten times. Due to randomness in initial solution generation,

local search, and perturbation, repeated executions produce different solutions.

To verify solution optimality, instances were also solved using the MILP model from Section 2. Experiments ran on a desktop computer with an Intel Core i7-6700 3.40 GHz CPU, 8 GB RAM, and Windows 10 operating system.

Algorithm parameters are listed in Table 2 . The coefficient α in objective function (1) was set to 10,000, sufficiently large to penalize service overloads. Parameter sensitivity was analyzed by solving instances with modified parameters. The TP and SPP models were solved using IBM ILOG CPLEX Optimizer 12.6.3. For the SAP model, CPLEX parameters were set to MIPGap = 10^{-10} and Timelimit = 7200 seconds. The SAP model was solved in two stages: (1) solve a transportation problem (Section 3.2) and repair the fragmented solution; (2) solve the SAP model using the repaired solution as a warm start.

4.2 Solution Results

Table 3 presents solutions for all 60 instances. Instance names encode the area, number of facilities, and configuration type. For each instance, CPLEX results show lower bounds (LB), upper bounds (UB), and computation time. Upper bounds marked with asterisks indicate optimal solutions. The hybrid algorithm produced 10 solutions per instance; columns Gap_{min}, Gap_{avg}, and Gap_{max} show the best, average, and worst objective gaps relative to the lower bound, calculated as (obj - LB)/LB × 100%. Column Dev shows the standard deviation of objective values, and Time indicates average computation time in seconds. All solutions satisfy capacity and contiguity constraints.

Table 3 demonstrates that CPLEX solved all SAP instances optimally or near-optimally, with 46 optimal solutions and 14 near-optimal solutions having MIP gaps between 0.00% and 0.32%. Contrary to early reports that only very small districting instances could be solved exactly, our experiments solved larger instances, attributable to advances in MILP solvers, CPU performance, and soft-constrained model formulations. Instance complexity depends on both size and supply-demand ratio; instances with higher ratios (sets A and B) were solved more efficiently than those with lower ratios (sets C and D).

The hybrid algorithm also proved effective and efficient. Its solutions approximate optimal solutions or lower bounds with an average gap of 0.15%, ranging from 0.00% to 0.82%. Among the best solutions (Gap_{min}), 28 (46.7%) are optimal. Table 4 summarizes the number of optimal solutions found by CPLEX and the hybrid algorithm. Objective deviations are small, ranging from 0.00% to 0.38%. All instances were solved within reasonable time: averages of 51.55, 41.35, and 68.15 seconds for areas ZY, GY, and GY2, respectively.

Set partitioning significantly improves ILS solutions, with an average improvement of 0.12% across all instances. Table 5 details improvements by instance set, showing particularly notable enhancements for area GY types C and D (0.60% and 0.43%, respectively). Occasionally, when ILS struggles to find feasible solu-

tions, set partitioning can select high-quality feasible solutions. Table 6 shows SPP model solution times, demonstrating that set partitioning improves SAP solutions efficiently.

4.3 Analysis of Algorithm Parameters

Parameter sensitivity was evaluated by solving all instances with nine alternative parameter sets. Table 7 summarizes solution gaps. Column Default shows gaps using parameters from Table 2; other columns show gaps with modified parameters:

- P1: population size = 1 (psize = 1)
- P5: population size = 5 (psize = 5)
- P20: population size = 20 (psize = 20)
- R10: perturbation strength = 10% (strength = 10%)
- R15: perturbation strength = 15% (strength = 15%)
- OP1: one-unit shift operator only
- OP2: two-unit shift operator only
- LS: simple local search in ILS loops
- RG: region growth method for initial solutions

Results reveal several insights. First, comparing Default, OP1, and OP2 shows that the two-unit shift operator outperforms the one-unit shift, while their combination yields slightly better results than two-unit shift alone. The one-unit shift is computationally fast but less effective, whereas the two-unit shift is slower but more powerful. Experiments suggest that using both operators in ILS search provides the best performance.

Second, comparing Default, P1, P5, and P20 indicates that population-based search (psize = 5, 10, or 20) outperforms single-solution search (psize = 1). Maintaining a population of elite, diverse solutions allows local search to explore a larger neighborhood space, increasing the likelihood of finding better solutions. Population-based ILS consistently outperforms its single-solution counterpart, with solution quality showing low sensitivity to population size.

Third, other parameters affect performance but exhibit limited sensitivity to solution quality. Perturbation strength is crucial for balancing intensification and diversification; strengths between 5%-15% are appropriate for most instances. Both simple and VND local search can guide the algorithm to good solutions, though VND' s ability to reach local optima may give it a slight advantage.

Final solutions show minor dependence on initial solution method, with the TP model plus repair procedure generally proving most effective.

5 Conclusion

We proposed a novel hybrid algorithm for the contiguity-constrained capacitated facility SAP. To our knowledge, this is the first application of population-based ILS with VND search and set partitioning for service area delineation.

The method was extensively tested on 60 well-designed instances, demonstrating its capability to find high-quality near-optimal solutions within reasonable computation times. The hybrid algorithm differs from existing approaches in several aspects. Perturbation plays a crucial role in escaping local optima, with appropriate strength guiding ILS toward global optima. Building upon standard ILS, we enhanced it with three schemes: VND search, population-based ILS, and set partitioning. VND search thoroughly explores the incumbent solution' s neighborhood, accelerating ILS convergence. Population-based ILS maintains a solution set, and starting from elite, diverse solutions increases the probability of finding better solutions through perturbation and local search.

Furthermore, service areas explored during local search are reused via set partitioning to select improved solutions. Traditional set-partitioning heuristics struggle to identify promising candidates for large instances, while local-search and population-based heuristics discover many solutions that are typically discarded. Historical solutions in metaheuristics represent high-quality candidates for set-partitioning approaches. Our algorithm records all service areas identified during each iteration and reselects them using an SPP model.

We introduced twelve benchmark instance sets of varying sizes and complexity based on three geographic areas. These instances are diverse in demand units, facility count, facility locations, supply-demand ratio, and average units per facility. Optimal or near-optimal solutions were obtained using CPLEX. The instances and results are available at https://github.com/yfkong and provide valuable benchmarks for evaluating existing and new SAP algorithms.

Future research should investigate parameter settings and adaptive adjustment strategies, as both theoretical and practical analysis could improve algorithm efficiency for specific instances. Given common criteria across districting problems, extending the algorithm to other domains such as political districting and location-districting problems appears promising.

References

Bacao, F., Lobo, V., and Painho, M., 2005. Applying genetic algorithms to zone design. *Soft Computing*, 9(5), 341-348.

Bergey, P. K., Ragsdale, C. T., and Hoskote, M., 2003. A Simulated Annealing Genetic Algorithm for the Electrical Power Districting Problem. *Annals of*

Operations Research, 121(1), 33-55.

Butsch, A., Kalcsics, J., and Laporte, G., 2014. Districting for Arc Routing. *Informs Journal on Computing*, 26(4), 809-824.

Caro, F., et al., 2004. School redistricting: embedding GIS tools with integer programming. *Journal of the Operational Research Society*, 55(8), 836-849.

Chou, C., Chu, Y., and Li, S., 2007. Evolutionary Strategy for Political Districting Problem Using Genetic Algorithm. *Lecture Notes in Computer Science*, 4490(4), 1163-1166.

Chou, C., 2011. A Knowledge-based Evolution Algorithm approach to political districting problem. *Computer Physics Communications*, 182(1), 209-212.

Daskin, M. S., 2011. *Service science*, Hoboken: John Wiley & Sons, 183-283.

Datta, D., et al., 2008. Graph partitioning through a multi-objective evolutionary algorithm: a preliminary study. *Proceedings of the 10th annual conference on Genetic and evolutionary computation*, 12-16 July 2008, Atlanta, 625-632.

Duque, J. C., Church, R. L., Middleton, R. S., 2011. The p-regions problem. *Geographical Analysis*, 43 (1), 104-126.

Emiliano, W. M., Telhada, J., and Carvalho, M. D., 2017. Home health care logistics planning: a review and framework. *Procedia Manufacturing*, 13, 948-955.

Ferland, J. A., and Guenette, G., 1990. Decision Support System for the School Districting Problem. *Operations Research*, 38(1), 15-21.

Forman, S. L. and Yue, Y., 2003. Congressional districting using a TSP-based genetic algorithm. *Lecture Notes on Computer Science*, 2724, 2072–2083.

Franklin, A. D. and Koenigsberg, E., 1973. Computed School Assignments in a Large District. *Operations Research*, 21(2), 413-426.

Garfinkel, R. S. and Nemhauser, G. L., 1970. Optimal Political Districting by Implicit Enumeration Techniques. *Management Science*, 16(8), 495-508.

George, J. A., Lamar, B.W., and Wallace, C. A., 1997. Political district determination using large-scale network optimization. *Socioeconomic Planning Science*, 31(1), 11–28.

Hess, S. W., et al., 1965. Nonpartisan political redistricting by computer. *Operations Research*, 13 (6), 998–1006.

Hojati, M., 1996. Optimal political districting. *Computers & Operations Research*, 23(12), 1145-1151.

Hu, F., Yang, S., and Xu, W., 2014. A non-dominated sorting genetic algorithm for the location and districting planning of earthquake shelters. *International Journal of Geographical Information Science*, 28(7), 1482-1501.

Hu, Y., Wang, F., and Xierali, I., 2018. Automated Delineation of Hospital Service Areas and Hospital Referral Regions by Modularity Optimization. *Health Services Research*, 53(1), 392-410.

Jacobs, D. A., Silan, M. N., and Clemson, B., 1996. An Analysis of Alternative Locations and Service Areas of American Red Cross Blood Facilities. *Interfaces*, 26(3), 40-50.

Kalcsics, J., Nickel, S. and Schroder, M., 2005. Towards a unified territorial design approach: applications, algorithms and GIS integration. *Top*, 13(1), 1-56.

Koenigsberg, E., 1968. Mathematical analysis applied to school attendance areas. *Socio-economic Planning Sciences*, 1(4): 465-475.

Kong, Y., Zhu, Y., and Wang, Y., 2017. A hybrid metaheuristic algorithm for the school districting problem. *Acta Geographica Sinica*, 2017, 72(2): 256-268. (in Chinese)

Kong, Y., Zhu, Y., and Wang, Y., 2019. A center-based modeling approach to solve the districting problem, *International Journal of Geographical Information Science*, 33(2), 403-424.

Li, X., et al., 2008. A Decentralized and Continuity-Based Algorithm for Delineating Capacitated Shelters' Service Areas. *Environment and Planning B: Planning and Design*, 35(4), 593–608.

Li, Z., Wang, R. and Wang, Y., 2007. A quadratic programming model for political districting problem. In: X. Zhang, L. Chen, L. Wu, et al., eds. *The First International Symposium on Optimization and Systems Biology (OSB' 07)*, 8–10 August 2007 Beijing. Beijing: World Publishing Corporation, 427-435.

Liberatore, F., and Camachocollados, M., 2016. A Comparison of Local Search Methods for the Multicriteria Police Districting Problem on Graph. *Mathematical Problems in Engineering*, Article ID 3690474, 13 pages.

Liu, Y., Cho, W. K., and Wang, S., 2016. PEAR: a massively parallel evolutionary computation approach for political redistricting optimization and analysis. *Swarm and evolutionary computation*, 30, 78-92.

Lodi, A., 2017. On Mixed-integer programming and its connection with data science [online]. EPFL. Available from: http://transp-or.epfl.ch/zinal/lectures2017.php [Accessed 5 April 2018].

Lourenço, H.R., Martin, O. and Stützle, T., 2010. Iterated Local Search: Framework and Applications. In: Gendreau, M. and Potvin, J.Y., eds. *Handbook of Metaheuristics*, 2nd. Edition. International Series in Operations Research & Management Science, Vol 146. New York: Springer, 363-397.

Mehrotra, A., Johnson, E. L., and Nemhauser, G. L., 1998. An Optimization Based Heuristic for Political Districting. *Management Science*, 44(8), 1100-1114.

Nemoto, T. and Hotta, K., 2003. Modelling and solution of the problem of optimal electoral districting. *Communications of Operations Research of Japan*, 48(4), 300-306 (in Japanese).

Nygreen, B., 1988. European assembly constituencies for Wales—comparing of methods for solving a political districting problem. *Mathematical Programming*, 42(1), 159-169.

Pezzella, F., Bonanno, R., and Nicoletti, B., 1981. A system approach to the optimal health-care districting. *European Journal of Operational Research*, 8(2), 139-146.

Plane, D. A., Tong, D. and Lei T., 2019. Inter-person Separation: A New Objective Standard for Evaluating the Spatial Fairness of Political Redistricting Plans. *Geographical Analysis*, 51, 251–279.

Ricca, F. and Simeone, B., 2008. Local search algorithms for political districting. *European Journal of Operational Research*, 189(3), 1409-1426.

Rincón-García E. A., et al., 2015. ABC, A Viable Algorithm for the Political Districting Problem. In: Gil-Aluja J., et al. eds. *Scientific Methods for the Treatment of Uncertainty in Social Sciences*. Advances in Intelligent Systems and Computing, 377. Cham: Springer.

Ríos-Mercado, R. Z., and Fernandez, E., 2009. A reactive GRASP for a commercial territory design problem with multiple balancing requirements. *Computers & Operations Research*, 36(3), 755-776.

Salazar-Aguilar, M. A., Ríos-Mercado, R. Z. and Gonzalez-Verlarde, J. L., 2011. A bi-objective programming model for designing compact and balanced territories in commercial districting. *Transportation Research Part C: Emerging Technologies*, 19(5), 885-895.

Salazar-Aguilar, M. A., et al., 2012. Multiobjective Scatter Search for a Commercial Territory Design Problem. *Annals of Operations Research*, 199(1), 343-360.

Schoepfle, O. B. and Church, R. L., 1991. A new network representation of a "classic" school districting problem. *Socio-economic Planning Sciences*, 25(3), 189-197.

Shirabe, T., 2005. A model of contiguity for spatial unit allocation. *Geographical Analysis*, 37(1), 2-16.

Sridharan, R., 1993. A Lagrangian heuristic for the capacitated plant location problem with single source constraints. *European Journal of Operational Research*, 66: 305-312.

Tavarespereira, F., et al., 2007. Multiple criteria districting problems. *Annals of Operations Research*, 154(1), 69-92.

Xiao, N., 2008. A Unified Conceptual Framework for Geographical Optimization Using Evolutionary Algorithms. *Annals of the Association of American Geographers*, 98(4), 798-817.

Yanik, S., Surer, O., and Oztaysi, B., 2016. Designing sustainable energy regions using genetic algorithms and location-allocation approach. *Energy*, 97, 161-172.

Corresponding author: Yunfeng Kong, yfkong@henu.edu.cn
Foundation Support: The National Natural Science Foundation of China, No. 41871307.

Note: Figure translations are in progress. See original paper for figures.