An Evolutionary Bi-Objective Approach to the Capacitated Facility Location Problem with Cost and CO_2 Emissions

Irina Harris
Cardiff Business School
Cardiff University
Cardiff, CF10 3EU, UK
HarrisI1@cardiff.ac.uk

Christine L Mumford
School of Computer Science & Informatics
Cardiff University
Cardiff, CF24 3AA, UK.
C.L.Mumford@cs.cardiff.ac.uk

Mohamed M Naim
Cardiff Business School
Cardiff University
Cardiff, CF10 3EU, UK
NaimMM@cardiff.ac.uk

ABSTRACT

It is strategically important to design efficient and environmentally friendly distribution networks. In this paper we propose a new methodology for solving the capacitated facility location problem (CFLP) based on combining an evolutionary multi-objective algorithm with Lagrangian Relaxation where financial costs and CO_2 emissions are considered simultaneously. Two levels of decision making are required: 1) which facilities to open from a set of potential sites, and 2) which customers to assign to which open facilities without violating their capacity. We choose SEAMO2 (Simple Evolutionary Multi-objective Optimization 2) as our multi-objective evolutionary algorithm to determine which facilities to open, because of its fast execution speed. For the allocation of customers to open facilities we use a Lagrangian Relaxation technique. We test our approach on large problem instances with realistic qualities, and validate solution quality by comparison with extreme solutions obtained using CPLEX®.

Categories and Subject Descriptors

I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search—*graph and tree search strategies, heuristic methods*

General Terms

Algorithms, Design, Experimentation, Performance

1. INTRODUCTION

One of the most important strategic issues for many businesses is where to site various storage and service facilities, and which facility should serve a particular customer. Usually there are restrictions that determine potential locations for facilities, and geographical, planning or financial constraints will influence a facility's capacity for storing goods and/or serving customers. The capacitated facility location problem (CFLP) provides a popular model for various distribution networks, and involves making decisions at two levels: 1) which facilities to open from a set of potential sites, and 2) which customers to assign to which open facilities. Furthermore, the CFLP requires that all constraints are complied with, for example, all customer demand should be met and no capacity constraint should be violated (i.e., demand on a particular facility must not outstrip supply). Traditionally, the design of a distribution network is driven by a need to reduce costs or maximize profit, and for this reason formulation is generally based on single objective optimization with all other potential objectives, such as customer service levels, modelled as extra constraints, so that a certain minimum level (e.g., of customer service) has to be maintained. In this paper we are concerned with *single source facility location*, in which each individual customer is required to source its total demand from a single facility. Although the CFLP is NP-Hard [14], Mixed Integer Programming (MIP) packages, such as CPLEX® can be used very effectively for small and even moderate sized instances. We shall show later in this paper that MIP does not cope well with very large instances, and thus there is an need for efficient heuristic/metaheuristic approaches. This is especially true for multi-objective optimization techniques, where repeated evaluations for each of objective functions are required to obtain a set of tradeoff solutions.

The motivation for our work stems from our view that, with recent environmental concerns and high levels of commercial competition, companies could benefit from using multi-objective optimization techniques when designing their distribution networks, to give extra flexibility to deal with key objectives simultaneously [12]; for example, minimizing the environmental impact of their business, while improving customer service levels at the same time as they are reducing cost or maximizing profits.

Increasingly, the UK and other nations throughout the world are recognizing the importance of reducing the environmental impact of their operations (e.g., see [2] and [13]). Some researchers have started to incorporate environmental measures with traditional objectives within a supply chain context. For example, Paksoy [16] proposed a multi-period supply chain design model which aims to minimize total transportation costs, and CO_2 emissions from transportation and manufacturing. Modern multi-objective objective optimization techniques [5] offer the flexibility to find good compromise solutions that balance the many important objectives, so that a decision maker has a choice of tradeoff

GECCO'11, July 12-16, 2011, Dublin, Ireland

solutions without the need to make *a priori* decisions regarding the relative importance of the various objectives. Instead of a single solution (usually optimized on cost), multi-objective optimization techniques can offer a decision maker a choice of tradeoff solutions, providing sufficient options to give him/her the power to make an informed choice that balance ALL the important objectives. It may be possible, for example, to greatly reduce energy consumption or waste gas emissions whilst incurring a very small increase in economic costs. Such compromise solutions can be easily missed using traditional single objective methods.Farahani *et al.*[7] in their recent review on multi-criteria location problems and solution approaches also point out that sustainability (environment and social) in location analysis is one of the important areas for future research.

In this paper we build on the multi-objective work carried out on the uncapacitated facility location problem (UFLP) in [11] and extend to the two-objective CFLP, balancing economic cost and CO_2 emissions. The multi-objective algorithm and Lagrangian Relaxation software have been built in Java. We were fortunate to have the opportunity to collaborate with a major supermarket chain during the past two years, which has given us a practical insight into the operation of a real-world distribution network. Applying this knowledge we have generated a range of large test instances with realistic characteristics (which can be obtained from [1]) to test our multi-objective approach. Furthermore we use the Company's model to evaluate the various costs associated with the different warehouse and transport activities within the supply chain. Although locations, capacities and levels of demand etc. are all randomly generated for our test instances, reasonable upper and lower bounds have been observed based on Company data. We have used UK Government sources to obtain correct environmental information regarding energy consumption.

In addition, similar to the Company's logistic model, our data sets have two capacity constraints for each potential facility: the maximum number of cases and the maximum number of stores that can be served. The two-objective CFLP model aims to balance the financial cost (£) and the environmental impact (kg CO_2), taking into account activities such as picking and loading the goods as well as transporting them and opening the depots needed to serve the customers' needs. The environmental impact is extracted from running the logistics network in terms of CO_2 emissions from the transportation and also from the emissions generated by energy use for the day-to-day running of the depots.

To solve the multi-objective CFLP and generate a set of tradeoff solutions for our benchmark data, we use a solution technique which is based on the elitist evolutionary multi-objective algorithm SEAMO2 (Simple Evolutionary Multi-objective Optimization) [15, 17]. We utilize a Lagrangian Relaxation technique [8, 9] to perform the allocation of customers to depots, which is described in detail in Section 3.1.

2. PROBLEM FORMULATION

We assume that customers have a certain demand in cases (for a single product type) and associated transportation and warehousing costs for a particular depot. Each depot has a given capacity in cases and the number of stores it is able to serve. Each customers is served directly by a single depot, and transportation costs are based on stem distances and reflect both time and distance based components. The warehouse (variable) costs reflect any associated costs with picking and loading the goods.

The problem is to determine how many facilities to open in order to satisfy all customer demand while solving both objectives simultaneously: minimize the environmental impact from operating depots and transport in terms of CO_2 emissions, and minimize the overall financial cost. As mentioned above, the CFLP is divided into two sub-problems: determine which depots to open, and assign customers to the open facilities without violating the number of cases or the number of customers capacity constraints.

The following notation is used in the formulation of the model:

Glossary

$V_{DC} = \{1...i\}$	set of potential depots;
$V_C = \{1...j\}$	set of customers;
c_{ij}	cost of attending demand d_j from customer j to depot i consisting of overall transportation and depot costs;
f_i	fixed cost for operating a depot i;
d_j	demand of customer j;
q_i	capacity (cases) of facility i, $i \in V_{DC}$;
n_i	capacity (number of customers) of facility i, $i \in V_{DC}$;
e_t_{ij}	CO_2 emissions from transport between depot i and customer j to satisfy customer demand d_j;
e_g_i	CO_2 emissions from gas consumption for each depot i, $i \in V_{DC}$;
e_e_i	CO_2 emissions from electricity consumption for each depot i, $i \in V_{DC}$;

The decision variables are:

x_{ij}	equals 1 if customer j is allocated to facility i, and 0 otherwise;
y_i	equals 1 if depot is chosen to operate and 0 otherwise;

The following economic and environmental objective functions are considered simultaneously as part of the network design:

- *Minimising costs.* This financial objective finds the best combination of open depots that allows minimization of the overall cost for the entire the network. It consists of the variable costs (transport and depots) of servicing the demand of all customers, plus the fixed cost of running the open depots.

$$minimize[\sum_{i \in V_{DC}} \sum_{j \in V_C} c_{ij} x_{ij} + \sum_{i \in V_{DC}} f_i y_i] \tag{1}$$

- *Minimizing the CO_2 emissions from transport and running depots.* The environmental objective is expressed as CO_2 emissions and aims to find the best number of open facilities that minimizes the total CO_2 emissions from transportation and energy consumption for running facilities. The first term of the formulation represents the emissions from transport to attend the demand of customers by the open depots, and the second term represents the total emissions from the electricity and gas usage of operating open depots.

$$minimize[\sum_{i \in V_{DC}} \sum_{j \in V_C} e\_t_{ij} x_{ij} + \sum_{i \in V_{DC}} (e\_g_i + e\_e_i) y_i] \tag{2}$$

- *Subject to following constraints:*

$$\sum_{i \in V_{DC}} x_{ij} = 1, j \in V_C \tag{3}$$

$$x_{ij} \le y_i, j \in V_C, i \in V_{DC} \tag{4}$$

$$\sum_{j \in V_C} d_j x_{ij} \le q_i, \forall i \in V_{DC} \tag{5}$$

$$\sum_{j \in V_C} x_{ij} \le n_i, \forall i \in V_{DC} \tag{6}$$

$$x_{ij} \in \{0, 1\}, i \in V_{DC}, j \in V_C \tag{7}$$

$$y_i \in \{0, 1\}, i \in V_{DC} \tag{8}$$

Constraints (3) and (7) ensure that each customer is attended by only one depot and the demand is satisfied by that facility. (4) assigns the customers to open depots only, and (5) and (6) ensure that the capacity constraints for demand (cases) and number of stores for the depots are not violated. Finally, (7) and (8) define decision variables as binary.

3. MULTI-OBJECTIVE OPTIMIZATION

The evolutionary multi-objective algorithm SEAMO2 [15, 17] (see Algorithm 1) was chosen for our work. Following our exploratory tests (unpublished) on the UFLP, this algorithm proved considerably faster than better known algorithms such as NSGA-II with very small reductions in solution quality (see also [3]).

In a multi-objective context, fitness functions are usually based either on a count of how many contemporaries in the population are dominated by a particular individual, or alternatively on by how many contemporaries the individual is itself dominated. This approach, known as *Pareto-based selection*, was first proposed by Goldberg [10], and is favoured, in one form or another, by most researchers (for example see [4, 6, 18]). In contrast, the SEAMO algorithms use a less compute-extensive approach, based on uniform random selection and do not require global fitness functions either to bias the selection of parents or to determine whether or not new offspring are inserted into the population. Instead a few simple rules determine "who shall live and who shall die" within a steady-state environment. Survival decisions are based on the outcome of simple comparisons between offspring solutions and parents (or other population members).

In more detail, after generating the initial population of size N (line 2), evaluating and storing the objectives for each member of the population (line 3), and recording the best-so-far values for each objective (line 4), the algorithm iterates through two nested loops. The inner (For) loop steps through each member of the population in turn, selecting that member as a first parent for a crossover operation (line 7), with the second parent selected at random from the other $N-1$ members of the population (line 8). Crossover is then performed to produce a single offspring (line 9), and then a single mutation is applied to that offspring (line 10). The SEAMO2 algorithm is steady-state, thus each time a new offspring is produced, it is considered for entry into the population, based on a sequence of comparisons. A "strong" offspring will replace a current population member according to the following criteria:

Algorithm 1 *SEAMO2*

```
1:  Begin:
2:  Generate N random individuals
3:  Evaluate the two objectives for each population member
    and store them
4:  Store best-so-far values for each objective
5:  while stopping condition not satisfied do
6:    for each member of the population do
7:      This individual becomes the first parent
8:      Select a second parent at random
9:      Apply crossover to produce single offspring
10:     Apply single mutation to the offspring
11:     Evaluate each objective vector produced by the off-
        spring
12:     if offspring harbors a new best-so-far Pareto com-
        ponent then
13:       a) it replaces a parent, if possible
14:       b) else it replaces another individual that it dom-
          inates at random
15:     else if offspring is a duplicate then
16:       it dies
17:     else if offspring dominates either parent then
18:       it replaces it
19:     else if offspring is neither dominated by nor dom-
        inates either parent then
20:       it replaces another individual that it dominates
          at random
21:     else
22:       otherwise it dies
23: Print all non-dominated solutions in the final population
24: End
```

a. It improves on a "best-so-far" Pareto objective,

b. Or it dominates either of its parents,

c. Or it is neither dominated by nor dominates either parent,

d. Provided that it is not a "duplicate".

The new offspring is first tested for an improvement on a current best-so-far score for at least one objective (line 12). If it succeeds, then the offspring will replace one of its parents and the appropriate best-so-far score will be updated (provided that best-so-far scores for other objectives are not lost in the process). In the present case, we are dealing with only two objectives. Thus, for example, suppose the offspring harbours a global improvement for objective 1, then it will be necessary to avoid deleting a single individual which is harbouring the best-so-far score for objective 2. Therefore, at least one parent will qualify for replacement. Provided the new offspring has not entered the population at this stage, the next test to be carried out (line 15) is to check whether the paired objective values of the offspring are duplicated (or within a small margin of error) in the current population. To help maintain population diversity, if a match is found, the offspring will die. If the offspring survives, the next test is a comparison for dominance between the offspring and its

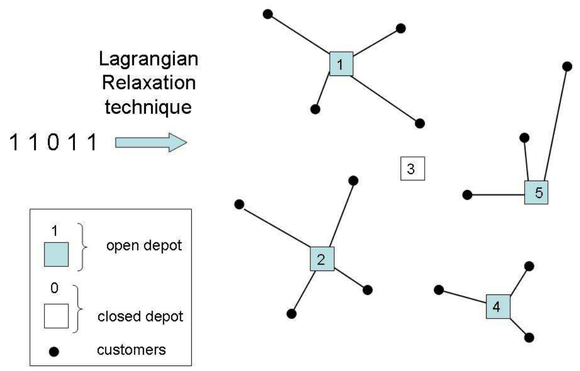

Figure 1: The assignment for Capacitated Facility Location Problem

parents (line 17), starting with the first parent. If the offspring dominates its first parent, then it will replace it in the population, and if it fails, then it will be tested against the second parent, and replace that parent if it dominates it. The replacement of population members by dominating offspring ensures that the solution vectors move closer to the Pareto front as the search progresses. Finally, if a decision has not been made by this stage, the last test establishes whether the new offspring has a mutually non-dominating relationship with both of its parents. If this is the case, the offspring will enter the population if possible, by replacing a current population member that is dominates, found by sampling the population at random without replacement, until a suitably weak member is identified which is dominated by the new offspring. If no such weak member can be identified, the offspring will die. The outer (While) loop repeats the algorithm until a stopping condition is satisfied.

Solution encoding involves the use of simple binary strings, where *1* represents an open depot, and *0* a closed depot, for example 0011011011 for a ten depot problem indicates that depots 3,4,6,7,9 and 10 are open, and the others are closed. The *assignment procedure* for the CFLP is extremely important and ensures that capacities in terms of cases and numbers of stores are not violated. Here, we utilize a *Lagrangian Relaxation (LR)* technique for assigning the customers to open depots. Figure 1 illustrates the assignment procedure. Our technique was adapted from [9] where it was applied to both location and assignment. As a result of applying our procedure, the customers are assigned according to the minimum possible cost.

3.1 Assigning customers to depots

For a given pattern of open/closed depots, we use Lagrangian Relaxation to assign the stores to open depots. In this work each store is assigned to a single depot, and the total costs are minimized whilst capacity constraints are adhered to. It is worth noting, however, that the assignment process could itself be considered as a multi-objective optimization process, simultaneously minimizing costs and the environmental impact of operations. Fortunately, solution quality is excellent for the present data instances when assignment is based on cost alone (see Section 7).

In our present LR formulation we relax the capacity constraint for the number of cases. Our model for relaxing the two (or more) capacity constraints simultaneously is work-in-progress and will be presented elsewhere. Please note that by relaxing only one constraint initially we were making the assumption that the number of cases is a harder constraint compared to the number of stores constraint, and this seemed appropriate on close examination of the data. Nevertheless, the feasibility of the upper bound (UB) solution was checked for violation of both constraints to ensure only feasible results were produced.

The main step in the Lagrangian relaxation is the determination of a lower bound obtained by relaxing the capacity (cases) constraint using Lagrangian multipliers. Please note that the fixed costs associated with running a depot do not need to be considered here, given that open depots have already been determined. However, the fixed costs must be added to the variable costs for the objectives in the SEAMO2 algorithm (Equations 1 and 2). Let $\lambda_i \in \mathbb{R}, \forall i \in V_{DC}$.

Minimize

$$\sum_{i \in V_{DC}} \sum_{j \in V_C} c_{ij} x_{ij} + \sum_{i \in V_{DC}} \lambda_i \Big(\sum_{j \in V_C} d_j x_{ij} - q_i \Big) \tag{9}$$

subject to

$$\sum_{i \in V_{DC}} x_{ij} = 1, \forall j \in V_C \tag{10}$$

$$\sum_{j \in V_C} x_{ij} \leq n_i, \forall i \in V_{DC} \tag{11}$$

$$x_{ij} \in \{0, 1\}, i \in V_{DC}, j \in V_C \tag{12}$$

In (9) the term in brackets on the right, $(\sum_{j \in V_C} d_j x_{ij} - q_i)$, calculates the difference between the total demand on a facility i imposed by the relaxed formulation, and its ability to meet that demand (i.e., its capacity (cases), q_i). If the capacity is violated, or underutilized, the value of total cost in (9) will change, depending on the value of λ_i.

One issue that needs to be considered regarding the right-hand side of formula (9), is that normal practice dictates that Lagrangian Relaxation only makes adjustments to the cost when a constraint is violated. Thus, in the case of (9) we would expect the term $(\sum_{j \in V_C} d_j x_{ij} - q_i)$ to equal zero, for any facility for which its capacity has not been exceeded. However, this is not the case, as under-utilized capacities will produce non-zero values. Later on in this paper we will make some suggestions as to how the Lagrangian scheme can be adapted to cope with this issue, by constraining the λ_i values: if $\lambda_i = 0$, it follows that $\sum_{i \in V_{DC}} \lambda_i (\sum_{j \in V_C} d_j x_{ij} - q_i)$ also equals zero.

Problem (9) - (12) can be decomposed into $|V_C|$ subproblems. For a given set of multipliers, $\lambda_i \in \mathbb{R}$, the optimal lower bound of the problem (9) - (12), $LB(\lambda)$, can be found by solving the following subproblem for each customer $j \in V_C$.
Minimize

$$\sum_{i \in V_{DC}} (c_{ij} + d_j \lambda_i) x_{ij} \tag{13}$$

subject to

$$\sum_{i \in V_{DC}} x_{ij} = 1, \forall j \in V_C \tag{14}$$

$$\sum_{j \in V_C} x_{ij} \leq n_i, \forall i \in V_{DC} \tag{15}$$

$$x_{ij} \in \{0,1\}, i \in V_{DC}, j \in V_C \tag{16}$$

and then by setting

$$LB(\lambda) = \sum_{j \in V_C} LB^j(\lambda) - \sum_{i \in V_{DC}} \lambda_i q_i \tag{17}$$

(13) is easily solved for the relaxed problem simply by applying a greedy algorithm to allocate each customer along the lowest cost arc, according to the augmented costs, $c_{ij} + d_j \lambda_i$. By suitably modifying the Lagrangian multipliers, it is possible to obtain a feasible solution to the original capacity constrained problem. To provide a good updating formula for the Lagrangian multipliers, we will need an upper bound, in addition to the lower bound in (17).

For an upper bound (UB) we will use a feasible solution obtained on the basis of the allocations of customers to facilities discovered in the evaluation of $LB(\lambda)$. However, it is likely that the allocation made for the lower bound calculation will produce some capacity violations. In order to obtain the best possible upper bound (i.e., with the lowest cost), we need to establish a good method for reallocating customers when facilities are over-subscribed. For an upper bound, we assert that it is better to allocate customers with high demand first, to try to ensure that individual depots have sufficient unused capacity. One possible way of doing this is to sort customers in non-increasing order of demand level (highest demand first), then work through the list, assigning customers in the same way as they were assigned to compute the LB, whenever possible. When capacity constraints are violated for the LB assignment, we iterate through the sorted list of depots, attempting to assign on the basis of the next lowest augmented cost depot, until a legal assignment is found, or the list is exhausted (in which case no feasible solution will be found and the UB will not be updated).

Updating the Lagrangian multipliers

For each facility at time step, k

$$s_i^k = \sum_{j \in V_C} x_{ij}^k d_j - q_i \tag{18}$$

where x_{ij}^k is the solution of the Lagrangian relaxation (9) - (12) using $\lambda_i^k \in \mathbb{R}, \forall i \in V_{DC}$ as the Lagrangian multipliers. Now set

$$\lambda_i^{k+1} = \begin{cases} \lambda_i^k + \beta^k s_i^k & if \ s_i^k > 0 \\ 0 & otherwise \end{cases} \tag{19}$$

where β^k is a suitable scalar coefficient. We will start the procedure by seeding all the Lagrangian multipliers to zero. Formula (19) can be explained in the following way. If for a certain facility i, s_i^k is positive, it means that demand outstrips supply for that facility, and thus the corresponding value of λ_i should be increased to increase the cost of assigning customers to that facility in the next round. Similarly, if s_i^k is negative, it means that there is spare capacity, so λ_i should be reduced to make that facility more attractive for assignment in the next iteration. However, as we pointed out earlier, it may not be appropriate to make adjustments to the multipliers when the capacity has not been violated for a facility. Formula (19) ensures that the λ_i^k are always positive.

Tuning the Lagrangian heuristic technique

To ensure that the algorithm is robust and performs efficiently, several experiments were performed in order to tune the β^k coefficient and also to determine how many iterations to perform between updates of the constant α. The coefficient β^k was tested with two different settings (20) and (21). However results were very similar for the two settings, and Equation (20) was incorporated into our final algorithm.

$$\beta^k = \frac{\alpha(UB - LB(\lambda^k))}{\sum_{i \in V_{DC}} (s_i^k)^2} \tag{20}$$

$$\beta^k = \frac{\alpha(UB(\lambda^k) - LB(\lambda^k))}{\sum_{i \in V_{DC}} (s_i^k)^2} \tag{21}$$

Parameter α is a constant in the interval $[0, 2]$ [9]. Here, we start α at 2 and halve it whenever the feasible upper bound fails to improve on the best known feasible upper bound for n iterations. Parameter n was tested in the range [1,100] with step 1 for all benchmark problems, to identify the best value for n. As a result, a value of 70 was chosen for n because our algorithm produced its best solutions (or very close) for most of the instances tested with these values. The total number of iterations was tested at: 500, 1000 and 2000. We discovered no difference in the final results, so a value of 500 was used for the total number of iterations in order to minimize the computational time. Finally, the algorithm for the Lagrangian Relaxation is described in Algorithm 2.

Algorithm 2 *Lagrangian heuristic algorithm for a single source capacitated allocation problem, single product*

```
Begin
(initialization)
Select a tolerance level ε ≥ 0
Set difference = +∞, LB = −∞, UB = +∞, k = 1 and
λ_i^k = 0, i ∈ V_DC
while (difference ≥ ε) OR (k ≤number of iterations) do
   (Computation of a new lower bound)
   Solve the Lagrangian relaxation (9) - (12) using λ_i^k ∈
   ℝ, ∀i ∈ V_DC multipliers (Greedy algorithm with on unca-
   pacitated version based on augmented costs). Let LB(λ^k)
   be its cost.
   if LB(λ^k) solution is feasible then
      STOP algorithm and return cost LB(λ^k)
   else if LB(λ^k) > LB then
      set LB = LB(λ^k)
      (Computation of a new upper bound)
      Determine the corresponding upper bound (modified
      greedy algorithm, as described in the text). Let UB(λ^k)
      be its cost.
      if UB(λ^k) < UB then
         set UB = UB(λ^k)
      Calculate difference = (UB − LB)/LB
      Update parameters s_i^k, β^k and compute Lagrangian
      multipliers λ_i^{k+1} (18)-(20), ∀i ∈ V_DC
      Update k=k+1
Return cost of the UB feasible solution
End
```

4. TEST DATA

Due to the lack of environmental test data for the multi-objective CFLP in the public domain, we generated random data sets [1]. Besides, we preferred to use data sets of realistic sizes with typical properties and constraints experienced in the real world. Thus we used Company data to guide the ranges of values used for demand, productivity, costs and the number of depots. On the other hand, we vastly increased the number of customers in some of our artificially generated instances. We generated environmental data for each depot based on the figures for average consumption of electricity and gas across some real depots. The following procedure was used for calculating this information for each facility, allowing us to derive a formula for energy consumption in kWH for a particular capacity of a depot.

The financial costs consist of both transportation and depot related costs, where transportation costs have distance and time related components. We assume that $c_{ij} = (tc_{ij} + dc_{ij})$, where tc_{ij} is the transportation costs and dc_{ij} is the related depot costs between customer j and depot i. Also, to reflect the fact that costs can vary, depending on geographical locations (e.g., labour costs tend to be higher in London and the South East), each depot has its own rates for transport and warehousing components.

It is important to ensure that feasible solutions exist for each generated instance. With this in mind, capacities were generated for each depot, relative to the computed overall demand across all depots. To achieve this the total demand was multiplied by a capacity ratio value (e.g., a ratio of 2 will ensure that total capacity summed over all potential facilities is twice the demand) and then divided by the total number of potential facilities, to give an equal capacity to each one. For the initial experiments, the following capacity ratio values were tested: 2, 3, 4, 5, 6, 7, 8, 9 and 10. A similar procedure was used to calculate a capacity for the maximum number of stores which each depot could serve. As a result, all depots are equivalent regarding capacity constraints. A simple cost multiplier was used to produce a fixed cost value from a capacity. Different fixed cost ratio values were tried (0.5, 0.75, 1.25 and 1.5) to create a range of instances with different features.

To ensure feasibility and sufficient flexibility for multiple objectives, following some initial tests with the SEAMO2 algorithm, we chose capacity ratio values of 4 and 8 and a fixed cost ratio value of 1.5 for further study, and generated large data instances based on these parameters for our main experimental work. The instances had 10 depots (all with equal capacities for cases and stores) and five different settings for the number of customers: 2000, 4000, 6000, 8000 and 10000. The name given to each instance reflects the different values generated. For example, instance *set1_10_2000_r4.0_fc1.25* has 10 depots, 2000 customers, a capacity ratio of 4 and fixed costs ratio of 1.25. In total, 10 different test instances were generated for analysis of the dual objective CFLP, where financial and environmental objectives are solved simultaneously.

5. TUNING SEAMO2

Prior to full experimentation, the algorithm was tuned using the S metric [18] to compare the quality of solutions produced by different crossover/mutation combinations. Two data instances (*set1_10_2000_r4.0_fc1.25* and *set1_10_8000_r4.0_fc1.25*) were used, and a population of 40 run for 250 generations. Three scenarios were tested: no crossover/no mutation, one-point crossover/mutation, two-point crossover/mutation, and uniform crossover/mutation. In total, 8 different sets of experiments were undertaken for tuning purposes. The S metric and final approximate Pareto frontier were obtained from 20 independent runs for each data instance and settings for mutation/crossover. After performing statistical analyses on the S metric results, it appears *uniform crossover with mutation* performed best amongst all settings. As a final step the size of the population was increased to 100 and the number of generations to 1000 to ensure that the algorithm runs a sufficient amount of time to find really good quality solutions.

6. EXPERIMENTAL SETUP

As mentioned above, a population of 100, with 1000 generations was used with uniform crossover and mutation as the final settings for the SEAMO2 algorithm, and 10 independent replicate runs were used on each of the 10 different data instances.To validate our multi-objective approach, we compared the quality of solutions located at the extremes (i.e., minimum cost and minimum environmental impact), with solutions produced by CPLEX® optimized on the single objectives of cost and environmental impact. Unfortunately, due to the large size of some of our data instances, it was impossible to determine the best CPLEX® solution, in all cases. For the CPLEX® optimization by cost, the instances with 2,000 customers were solved taking between 9 and 30 hours. On the other hand, because of a simpler problem formulation, all instances were solved for CO_2 emissions by CPLEX® within an hour or two.

7. RESULTS

A Pareto plot for each data instance was obtained by aggregating the fronts of 10 independent runs for the SEAMO2 algorithm. Figure 2 illustrates the approximate Pareto frontier obtained for instance *set1_10_2000_r4.0_fc1.25*. The technique found very good solutions for both extreme points of the front. The solution which was found by CPLEX® for the optimization by cost alone, is identical to the solution found by SEAMO2, and the solution found in the CPLEX® optimization by CO_2 is very close to the other extreme solution on the edge of the Pareto front. Figure 3 illustrates trade off solutions for instance *set1_10_6000_r4.0_fc1.25*. In both cases the trade-off solutions appear to be rather unevenly spread across the approximate Pareto front, which could be due to the problem configuration, producing a relatively small number of feasible solutions. A similar pattern was observed for all the data sets, although lack of space prevents us from including the other 8 in this paper.

As can be seen in Figures 2 and 3, the low cost solution produces the highest CO_2 emissions with fewer depots open, whereas the best solution for environmental impact needs more open depots. The compromise solutions highlighted for each of the instances identify reduced CO_2 emissions before the frontier steepens towards high costs.

Tables 1 and 2 present a summary of the results for all instances comparing the SEAMO2 multi-objective algorithm on 10 independent runs with single objective solutions based on costs or emissions using the CPLEX® optimization software. In Table 1, we compare the solution quality produced by optimization for cost or CO_2 emissions using CPLEX® to the best found cost or CO_2 solution found by SEAMO2. As can be seen, SEAMO2 produced high quality solutions which came at around 0.14%-0.88% difference from solutions found by CPLEX® .

Table 2 records total execution times for 10 independent

Data instance	Cost solution			CO_2 solutuion		
	CPLEX® (optimisation by cost)	SEAMO2 (best cost solution)	% diff	CPLEX® (optimisation by CO_2)	SEAMO2 (best CO_2 solution)	% diff
set1_10_2000_r4.0_fc1.25	54,476,666.18	54,476,666.18	0	9,110,719.54	9,133,357.28	0.25
set1_10_2000_r8.0_fc1.25	71,012,035.53	71,012,035.53	0	11,350,098.95	11,433,813.45	0.74
set1_10_4000_r4.0_fc1.25	n/a	110,290,153.21	n/a	19,114,557.23	19,193,122.65	0.41
set1_10_4000_r8.0_fc1.25	n/a	142,865,836.50	n/a	23,551,122.74	23,593,999.22	0.18
set1_10_6000_r4.0_fc1.25	n/a	163,086,191.26	n/a	27,502,122.32	27,563,028.70	0.22
set1_10_6000_r8.0_fc1.25	n/a	211,113,103.48	n/a	34,509,852.20	34,559,236.39	0.14
set1_10_8000_r4.0_fc1.25	n/a	229,077,087.04	n/a	39,069,000.26	39,354,372.01	0.73
set1_10_8000_r8.0_fc1.25	n/a	295,814,178.47	n/a	49,987,201.17	50,425,972.79	0.88
set1_10_10000_r4.0_fc1.25	n/a	277,887,312.30	n/a	46,848,513.35	47,210,577.06	0.77
set1_10_10000_r8.0_fc1.25	n/a	359,872,980.28	n/a	58,644,745.65	58,858,491.71	0.36

Table 1: Solution comparison between SEAMO2 and CPLEX® optimization

Data instance	SEAMO2		CPLEX®	
	Num of non-dominated solutions	Time(sec)	By Cost Time(sec)	By CO_2 Time(sec)
set1_10_2000_r4.0_fc1.25	9	1,676	107,544	24
set1_10_2000_r8.0_fc1.25	5	166	33,451	77
set1_10_4000_r4.0_fc1.25	6	5,118	n/a	157
set1_10_4000_r8.0_fc1.25	4	250	n/a	1,066
set1_10_6000_r4.0_fc1.25	10	5,309	n/a	533
set1_10_6000_r8.0_fc1.25	6	409	n/a	1,671
set1_10_8000_r4.0_fc1.25	7	9,840	n/a	564
set1_10_8000_r8.0_fc1.25	4	817	n/a	3,662
set1_10_10000_r4.0_fc1.25	10	9,641	n/a	3,277
set1_10_10000_r8.0_fc1.25	4	440	n/a	8,063

Table 2: Comparison of run times between SEAMO2 and CPLEX®

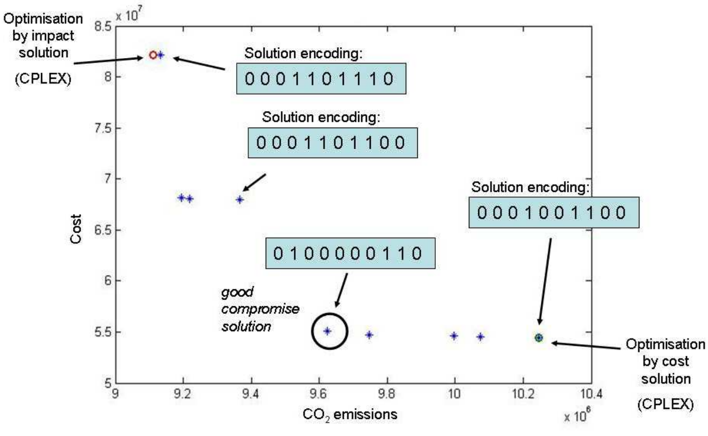

Figure 2: Approximate Pareto frontier for instance: *set1_10_2000_r4.0_fc1.25*

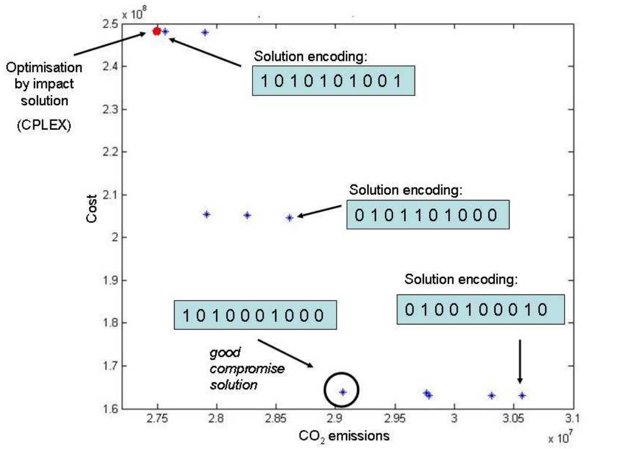

Figure 3: Approximate Pareto frontier for instance: *set1_10_6000_r4.0_fc1.25*

runs of SEAMO2 for each instance, and also the total number of non-dominated solutions produced by SEAMO2. Comparing execution times for 10 runs of SEAMO2 with single runs of CPLEX® for minimizing cost and CO_2 emissions, it can be seen that CPLEX® is highly impractical for large instances, particularly for cost optimization. However SEAMO2 is able to find a set of non-dominated solutions, for which it can be seen that the extreme solutions are close to the optima. These results show us that in terms of execution times, SEAMO2 (combined with LR for allocation of customers to stores) is computationally very fast, and can provide a good set of tradeoff solutions to a decision maker.

To conclude, throughout the analysis on our large data sets it was demonstrated that the SEAMO2 algorithm combined with Lagrangian Relaxation (for customer assignment) is able to find efficient trade-off solutions balancing cost and CO_2 emissions for network design very quickly. In many cases it is easy to spot good compromise solutions.

8. CONCLUSION

This paper presents an evolutionary multi-objective approach to the CFLP problem using SEAMO2, balancing two objectives: financial cost and CO_2 emissions. We generated random data sets based on real-world data to use in experiments, some instances with huge numbers of customers. In addition, we integrated a Lagrangian Relaxation (LR) technique to find the best assignment of stores to open depots for any particular individual in the population. Both the multi-objective optimization and the LR have been coded by us in Java. The evolutionary algorithm determines which depots are open, and the LR takes care of the allocation of customers to depots. The quality of results compares very well with single-objective optima obtained using the CPLEX® optimization software, where is possible to make comparisons, and runtimes are considerably faster for SEAMO2. Indeed, CPLEX® is not a realistic option for very large instances. The analysis of our findings confirm that a multi-objective approach can be efficient at identifying good compromise solutions to balance cost and environmental impact, with the goal of making significant savings in CO_2 emissions at an affordable cost. Work-in-progress includes a weighted sum approach to multi-objective optimization, and an improved LR model. In future we plan to extend our work to more complex supply chain problems with more objectives, and also take account of uncertainty in our approach.

9. REFERENCES

[1] Harrisdata. `http://users.cs.cf.ac.uk/C.L.Mumford/Research%20Topics/FLP/papers/data/`.

[2] B. M. Beamon. Designing the green supply chain. *Logistics Information Management*, 12(4):332–342, 1999.

[3] G. Colombo and C. Mumford. Comparing algorithms, representations and operators for the multi-objective knapsack problem. In *Evolutionary Computation, 2005. The 2005 IEEE Congress on*, volume 2, pages 1268–1275, 2005.

[4] D. Corne, J. Knowles, and M. Oates. The Pareto envelope-based selection algorithm for multiobjective optimization. In M. Schoenauer et al., editors, *Parallel Problem Solving from Nature PPSN VI*, volume 1917 of *Lecture Notes in Computer Science*, pages 839–848. Springer Berlin / Heidelberg, 2000.

[5] K. Deb. *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & Sons, 2001.

[6] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimisation: NSGA- *II*. In *Proceedings of the 6th International Conference on Parallel Problem Solving from Nature*, PPSN VI, pages 849–858, London, UK, 2000. Springer-Verlag.

[7] R. Z. Farahani, M. SteadieSeifi, and N. Asgari. Multiple criteria facility location problems: A survey", journal = "applied mathematical modelling. 34(7):1689–1709, 2010.

[8] A. M. Geoffrion. Lagrangean relaxation for integer programming. In R. W. Cottle et al., editors, *Approaches to Integer Programming*, volume 2 of *Mathematical Programming Studies*, pages 82–114. Springer Berlin Heidelberg, 1974.

[9] G. Ghiani, G. Laporte, and R. Musmanno. Designing the logistics network. In *Introduction to Logistics Systems Planning and Control*, Wiley Interscience Series in Systems and Optimization, chapter 3, pages 88–120. John Wiley & Sons, 2004.

[10] D. E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, Reading, MA, 1989.

[11] I. Harris, C. Mumford, and M. Naim. The multi-objective uncapacitated facility location problem for green logistics. In *CEC '09. IEEE Congress on Evolutionary Computation*, pages 2732 –2739, May 2009.

[12] I. Harris, M. Naim, A. Palmer, A. Potter, and C. Mumford. Assessing the impact of cost optimization based on infrastructure modelling on CO2 emissions. *International Journal of Production Economics*, 131(1):313–321, 2011. Innsbruck 2008.

[13] C. Kohn and M. Huge Brodin. Centralised distribution systems and the environment: how increased transport work can decrease the environmental impact of logistics. *International Journal of Logistics: Research and Applications*, 11(3):229–245, 2008.

[14] P. Mirchandani and R. Francis. *Discrete Location Theory*. John Wiley and Sons, 1990.

[15] C. L. Mumford. Simple population replacement strategies for a steady-state multi-objective evolutionary algorithm. In K. Deb et al., editors, *Genetic and Evolutionary Computation, GECCO 2004*, volume 3102 of *Lecture Notes in Computer Science*, pages 1389–1400. Springer-Verlag.

[16] T. Paksoy. Optimizing a supply chain network with emission trading factor. *Scientific Research and Essays*, 5(17):2535–2546, 2010.

[17] C. Valenzuela. A simple evolutionary algorithm for multi-objective optimization (seamo). In *Evolutionary Computation, 2002. CEC '02. Proceedings of the 2002 Congress on*, volume 1, pages 717–722, May 2002.

[18] E. Zitzler, M. Laumanns, and L. Thiele. Spea2: Improving the strength pareto evolutionary algorithm. Technical Report 103, Swiss Federal Institute of Technology (ETH), 2001.