LA-UR-10-00415

*Approved for public release;
distribution is unlimited.*

Title:

Strategic Planning for Disaster Recovery with Stochastic Last Mile Distribution

Author(s):

Russell Bent, Pascal Van Hentenryck, and Carleton Coffrin

Intended for:

Submission to the Seventh International Conference on Integration of Artificial Intelligence (AI) and Operations Research (OR) techniques in Constraint Programming (CPAIOR 2010)

Strategic Planning for Disaster Recovery with Stochastic Last Mile Distribution

Pascal Van Hentenryck[1], Russell Bent[2], and Carleton Coffrin[1]

[1] Brown University, Providence RI 02912, USA
[2] Los Alamos National Laboratory, Los Alamos NM 87545, USA

Abstract. This paper considers the single commodity allocation problem (SCAP) for disaster recovery, a fundamental problem faced by all populated areas. SCAPs are complex stochastic optimization problems that combine resource allocation, warehouse routing, and parallel fleet routing. Moreover, these problems must be solved under tight runtime constraints to be practical in real-world disaster situations. This paper formalizes the specification of SCAPs and introduces a novel multi-stage hybrid-optimization algorithm that utilizes the strengths of mixed integer programming, constraint programming, and large neighborhood search. The algorithm was validated on hurricane disaster scenarios generated by Los Alamos National Laboratory using state-of-the-art disaster simulation tools and is deployed to aid federal organizations in the US.

1 Background & Motivation

Every year seasonal hurricanes threaten coastal areas. The severity of hurricane damage varies from year to year, but considerable human and monetary resources are always spent to prepare for and recover from these disasters. It is policy makers who make the critical decisions relating to how money and resources are allocated for preparation and recovery. At this time, preparation and recovery plans developed by policy makers are often ad hoc and rely on available subject matter expertise. Furthermore, the National Hurricane Center (NHC) of the National Weather Service in the United States (among others) is highly skilled at generating ensembles of possible hurricane tracks but current preparation methods often ignore this information.

This paper aims at solving this problem more rigorously by combining optimization techniques and disaster-specific information given by NHC predictions. The problem is not only hard from a combinatorial optimization standpoint, but it is also inherently stochastic because the exact outcome of the disaster is unknown. Although humans have difficulty reasoning over uncertain data, recent work in the optimization community [13, 5] has shown that stochastic optimization techniques can find robust solutions in problems with uncertainty to overcome this difficulty.

The paper considers the following abstract disaster recovery problem: How to store a single commodity throughout a populated area to minimize its delivery time after a disaster has occurred. It makes the following technical contributions:

1. It formalizes the single commodity allocation problem (SCAP).
2. It proposes a multi-stage hybrid-optimization decomposition for SCAPs, combining a MIP model for stochastic commodity storage, a hybrid CP/MIP model for multi-trip vehicle routing, and a large neighborhood search model for minimizing the latest delivery time in multiple vehicle routing.
3. It validates the approach on the delivery of potable water for hurricane recovery.

Section 2 of this paper reviews similar work on disaster preparation and recovery problems. Section 3 presents a mathematical formulation of the disaster recovery problem and sets up the notations for the rest of paper. Section 4 presents the overall approach using (hopefully) intuitive models. Section 5 presents a number of modeling and algorithmic improvements that refines each of the initial models; it also presents the final version of the optimization algorithm for SCAPs. Section 6 reports experimental results of our complete algorithm on some benchmark instances to validate the approach and Section 7 concludes the paper.

2 Previous Work

The operations research community has been investigating the field of humanitarian logistics since the 1990s but recent disasters have brought increased attention to these kinds of logistical problems [18, 4, 10, 9]. Humanitarian logistics is filled with a wide variety of optimization problems that combine aspects from classic problems in inventory routing, supply chain management, warehouse location, and vehicle routing. The problems posed by humanitarian logistics add significant complexity to their classical variants. The operations research community recognizes that novel research in this area is required to solve these kinds of problems [18, 4]. Some of the key features that characterize these problems are as follows:

1. **Multi-Objective Functions** - High-stake disaster situations often have to balance conflicting objective goals (e.g. operational costs, speed of service, and unserved customers) [3, 8, 2, 12].
2. **Non-Standard Objective Functions** - A makespan time objective in VRPs [3, 6] or equitability objectives [2].
3. **Arbitrary Side Constraints** - Limited resources, a fixed vehicle fleet [2], fixed latest delivery time [3, 2], or a insufficient preparation budget [8, 11].
4. **Stochastic Aspects** - Disasters are inherently unpredictable. Preparations and recovery plans must be robust with respect to many scenarios [8, 12].

Humanitarian logistics also studies these problems at a variety of scales in both space and time. Some problems consider a global scale with time measured in days and weeks [8], while others focus on the minute-by-minute details of delivering supplies from local warehouses directly to the survivors [3, 2]. This paper considers a scale which is often called the “last mile” of distribution. This involves warehouse selection and customer delivery at the city and state scale.

The operations research community has mainly formulated these problems using MIP models. Many of the humanitarian logistics problems are complex and MIP formulations do not always scale to real world instances [2, 3]. Additionally, it was shown that MIP solvers can have difficulty with some of the unique features of these kinds of problems even when problem sizes are small (e.g., with minimizing the latest delivery time in VRPs [6]). Local search techniques are often used to scale the problems to real world instances [3, 6]. This paper demonstrates how hybrid optimization methods and recent advances in the optimization community can yield high-quality solutions to such challenges. To the best of our knowledge, SCAPs are the first humanitarian logistic problem to investigate the "last mile" vehicle routing problem and stochastic disaster information simultaneously.

3 The Single Commodity Allocation Problem (SCAP)

In formalizing SCAPs, a populated area is represented as a graph $G = \langle V, E \rangle$ where V represents those sites of interest to the allocation problem, i.e., sites requiring the commodity after the disaster (e.g., hospitals, shelters, and public buildings) and vehicle storage depots. The required commodity can be stored at any node of the graph subject to some side constraints. For simplicity, we assume the graph is complete and the edges have weights representing travel times. The weights on the edges form a metric space but it is not Euclidean due to the transportation infrastructure. Moreover, the travel times can vary in different disaster scenarios due to road damage. The primary outputs of a SCAP are (1) the amount of commodity to be stored at each node; (2) for each scenario and each vehicle, the best plan to deliver the commodities. Figure 1 summarizes the entire problem, which we now describe in detail.

Objectives The objective function aims at minimizing three factors: (1) The amount of unsatisfied demands; (2) the time it takes to meet those demands; (3) the cost of storing the commodity. Since these values are not expressed in the same units, it is not always clear how to combine them into a single objective function. Furthermore, their relative importance is typically decided by policy makers on a case-by-case basis. For these reasons, this paper uses weights W_x, W_y, and W_z to balance the objectives and to give control to policy makers.

Side Constraints The first set of side constraints concerns the nodes of the graph which represent the repositories in the populated area. Each repository $R_{i \in 1..n}$ has a maximum capacity RC_i to store the commodity. It also has a one-time initial cost RI_i (the investment cost) and an incremental cost RM_i for each unit of commodity to be stored. As policy makers often work within budget constraints, the sum of all costs in the system must be less than a budget B.

The second set of side constraints concerns the deliveries. We are given a fleet of m vehicles $V_{i \in 1..m}$ which are homogeneous in terms of their capacity VC. Each vehicle has a unique starting depot D_i^+ and ending depot D_i^-. Unlike

Given:

Repositories: $R_{i \in 1..n}$
  Capacity: RC_i
  Investment Cost: RI_i
  Maintenance Cost: RM_i
Vehicles: $V_{i \in 1..m}$
  Capacity: VC
  Start Depot: D_i^+
  End Depot: D_i^-
Scenario Data: $S_{i \in 1..a}$
  Scenario Probability: P_i
  Available Sites: $AR_i \subset \{1..n\}$
  Site Demand: $C_{i,1..n}$
  Travel Time Matrix: $T_{i,1..l,1..l}$
Weights: W_x, W_y, W_z
Budget: B

Output:

The amount stored at each warehouse
Delivery schedules for each vehicle

Minimize:

$W_x *$ Unserved Demands $+$
$W_y * MAX^i_{1..m}$Tour Time$_i +$
$W_z *$ Investment Cost $+$
$W_z *$ Maintenance Cost

Subject To:

Vehicle and site capacities
Vehicles start and end locations
Costs $\leq B$

Notes:

Every warehouse that stores a unit must be visited at least once

Fig. 1. Single Commodity Allocation Problem Specification

classic vehicle routing problems [17], customer demands in SCAPs often exceed the vehicle capacity and hence multiple deliveries are often required to serve a single customer.

Stochasticity SCAPs are specified by a set of a different disaster scenarios $S_{i \in 1..a}$, each with an associated probability P_i. After a disaster, some sites are damaged and each scenario has a set AR_i of available sites where the stored commodities remain intact. Moreover, each scenario specifies, for each site R_i, the demand C_i. Note that a site may have a demand even if a site is not available. Finally, site-to-site travel times $T_{i,1..l,1..l}$ (where $l = |V|$) are given for each scenario and capture infrastructure damages.

Unique Features Although different aspects of this problem were studied before in the context of vehicle routing, location routing, inventory management, and humanitarian logistics, SCAPs present unique features. Earlier work in location-routing problems (LRP) assumes that (1) customers and warehouses (storage locations) are disjoint sets; (2) the number of warehouses is $\approx 3..10$; (3) customer demands are less than the vehicle capacity; (4) customer demands are atomic

None of these assumptions hold in the SCAP context. In a SCAP, it may not only be necessary to serve a customer with multiple trips but, due to the storage capacity constraints, those trips may need to come from different warehouses. The key features of SCAP are: (1) each site can be a warehouse and/or customer; (2) one warehouse may have to make many trips to a single customer; (3) one customer may be served by many warehouses; (4) the number of available vehicles is fixed; (5) vehicles may start and end in different depots; (6) the objective is to minimize the time of the last delivery. Minimizing the time of the last delivery is one of the most difficult aspects of this problem as in demonstrated in [6].

4 The Basic Approach

This section presents the basic approach to the SCAP problem for simplifying the reading of the paper. Modeling and algorithmic improvements are presented in Section 5. Previous work on location routing [7, 1, 15] has shown that reasoning over both the storage problem and the routing problem simultaneously is extremely hard computationally. To address this difficulty, we present a three-stage algorithm that decomposes the storage, customer allocation, and routing decisions. The three stages and the key decisions of each stage are as follows:

1. **Storage & Customer Allocation**: Which repositories store the commodity and how is the commodity allocated to each customer?
2. **Repository Routing**: For each repository, what is the best customer distribution plan?
3. **Fleet Routing**: How to visit the repositories to minimize the time of the last delivery?

The decisions of each stage are independent and can use the optimization technique most appropriate to their nature. The first stage is formulated as a MIP, the second stage is solved optimally using constraint programming, and the third stage uses large neighborhood search (LNS).

Storage & Customer Allocation The first stage captures the cost and demand objectives precisely but approximates the routing aspects. In particular, the model only considers the time to move the commodity from the repository to a customer, not the maximum delivery times. Let D be a set of delivery triples of the form $\langle source, destination, quantity \rangle$. The delivery-time component of the objective is replaced by

$$W_y * \sum_{\langle s,d,q \rangle \in D} T_{s,d} * q/VC$$

Figure 2 presents the stochastic MIP model, which scales well with the number of disaster scenarios because the number of integer variables only depends on the number of sites n. The meaning of the decision variables is explained in the figure. Once the storage and customer allocation are computed, the uncertainty is revealed and the second stage reduces to a deterministic multi-depot, multiple-vehicle capacitated routing problem whose objective consists in minimizing the latest delivery. To our knowledge, this problem has not been studied before. One of its difficulties in this setting is that the customer demand is typically much larger than the vehicle capacity. As a result, we tackle it in two steps. We first consider each repository independently and determine a number of vehicle trips to serve the repository customers (Repository Routing). A trip is a tour that starts at the depot, visits customers, returns to the depot, and satisfies the vehicle capacity constraints. We then determine how to route the vehicles to perform all the trips and minimize the latest delivery time (Fleet Routing).

Variables:

$Stored_{i \in 1..n} \in [0, RC_i]$ - Units stored
$Open_{i \in 1..n} \in \{0, 1\}$ - More than zero units stored flag

$StoredSaved_{s \in 1..a, i \in 1..n} \in [0, C_{s,i}]$ - Units used at the storage location
$StoredSent_{s \in 1..a, i \in 1..n} \in [0, RC_i]$ - Total units shipped to other locations
$Incoming_{s \in 1..a, i \in 1..n} \in [0, C_{s,i}]$ - Total units coming from other locations
$Unsatisfied_{s \in 1..a, i \in 1..n} \in [0, C_{s,i}]$ - Demand not satisfied
$Sent_{s \in 1..a, i \in 1..n, j \in 1..n} \in [0, RC_i/VC]$ - Trips needed from i to j

Minimize:

$$W_x * \sum_{s \in 1..a} P_s * \sum_{i \in 1..n} Unsatisfied_{s,i} +$$
$$W_y * \sum_{s \in 1..a} P_s * \sum_{i \in 1..n} \sum_{j \in 1..n} T_{s,i,j} * Sent_{s,i,j} +$$
$$W_z * \sum_{i \in 1..n} (RI_i * Open_i + RM_i * Stored_i)$$

Subject To:

$$\sum_{i \in 1..n} (RI_i * Open_i + RM_i * Stored_i) \leq B$$
$$RC_i * Open_i \geq Stored_i \quad \forall i$$
$$StoredSaved_{s,i} + Incoming_{s,i} + Unsatisfied_{s,i} = C_{s,i} \quad \forall s, i$$
$$StoredSaved_{s,i} + StoredSent_{s,i} \leq Stored_i \quad \forall s, i$$
$$\sum_{j \in 1..n} VC * Sent_{s,i,j} = StoredSent_{s,i} \quad \forall s, i$$
$$\sum_{j \in 1..n} VC * Sent_{s,j,i} = Incoming_{s,i} \quad \forall s, i$$
$$StoredSaved_{s,i} + StoredSent_{s,i} = 0 \quad \forall s, i \text{ where } i \text{ not in } AR_s$$

Fig. 2. Storage & Customer Selection: The MIP Model.

Repository Routing Figure 3 shows how to create the inputs for repository routing from the outputs of the MIP model. For a given scenario s, the idea is to compute the customers of each repository w, the number of full-capacity trips $FullTrips_{s,w,c}$ and the remaining demand $Demand_{s,w,c}$ needed to serve each such customer c. The full trips are only considered in the fleet routing since they must be performed by a round-trip. The minimum number of trips required to serve the remaining customers is also computed using a bin-packing algorithm. The repository routing then finds a set of trips serving these customers with minimal travel time. The repository routing is solved using a simple CP model depicted in Figure 4. The model uses two depots for each possible trip (a starting and an ending depot localized at the repository) and considers nodes consisting of the depots and the customers. Its decision variables are the successor variables specifying which node to visit next and the trip variables associating a trip with each customer. The circuit constraint expresses that the successor variables constitute a circuit, the vehicle capacity constraint is enforced with a multi-knapsack constraint, and the remaining constraints associate a trip number with every node. This model is then solved to optimality.

```
Given scenario $s$ and for each repository $w \in 1..n$
    $Customers_{s,w} = \{i \in 1..n : Sent_{s,w,i} > 0\}$
    For $c \in Customers_{s,w}$
        $FullTrips_{s,w,c} = \lfloor Sent_{s,w,c} \rfloor$
        $Demand_{s,w,c} = VC * (Sent_{s,w,c} - \lfloor Sent_{s,w,c} \rfloor)$
    $MinTrips_{s,w} = MinBinPacking(\{Demand_{s,w,c} : c \in Customers_{s,w}\}, VC)$
```

Fig. 3. The Inputs for the Repository Routing.

Let:

$Depots^{+}_{s,w} = \{d^{+}_{1}, d^{+}_{2}, ..., d^{+}_{MinTrips_{s,w}}\}$

$Depots^{-}_{s,w} = \{d^{-}_{1}, d^{-}_{2}, ..., d^{-}_{MinTrips_{s,w}}\}$

$Nodes_{s,w} = Depots^{+}_{s,w} \cup Depots^{-}_{s,w} \cup Customers_{s,w}$

$Trips_{s,w} = \{1, 2, ..., MinTrips_{s,w}\}$

Variables:

$Successor[Nodes_{s,w}] \in Nodes_{s,w}$ - Node traversal order

$Trip[Nodes_{s,w}] \in Trips_{s,w}$ - Node trip assignment

Minimize:

$$\sum_{n \in Nodes_{s,w}} T_{s,n,Successor[n]}$$

Subject To:

$circuit(Successor)$

$multiknapsack(Trip, \{Demand_{s,w,c} : c \in Customers_{s,w}\}, VC)$

for $w^{+}_{i} \in Depots^{+}_{s,w}$: $Trip[w^{+}_{i}] = i$

for $w^{-}_{i} \in Depots^{-}_{s,w}$: $Trip[w^{-}_{i}] = i$

for $n \in Customers_{s,w} \cup Depots^{+}_{s,w}$: $Trip[n] = Trip[Successor[n]]$

Fig. 4. The CP Model for Repository Routing.

Fleet Routing It then remains to decide how to schedule the trips for the fleet to perform and to minimize the latest delivery time. The capacity constraints can be ignored now since each trip satisfies them. Each trip is abstracted into a task at the warehouse location and a service time capturing the time to perform the trip. The fleet routing problem then consists of using the vehicles to perform all these tasks while minimizing the latest delivery.

Figure 5 depicts how to compute the inputs for fleet routing given the results of the earlier steps, which consists of computing the proper service times $TripTime_t$ for each trip t. The model for the fleet routing is depicted in Figure 6 and is a standard CP formulation for multiple vehicle routing adapted to minimize the latest delivery time. For each node, the decision variables are its successor, its vehicle, and its delivery time. The objective minimizes the maximum delivery time and the rest of the model expresses the subtour elimination constraints, the vehicle constraints, and the delivery time computation.

The fleet routing problem is solved using LNS [16] to obtain high-quality solutions quickly given the significant number of nodes arising in large instances. At each optimization step, the LNS algorithm selects 15% of the trips to re-

```
Given scenario $s$ and for each repository $w \in 1..n$
    $RoundTrips_{s,w} = \{FullTrips_{s,w,c} : c \in Customers_{s,w}\}$
    $Tasks_{s,w} = \{t_1, t_2, ..., t_{Trips_{s,w}}\} \cup RoundTrips_{s,w}$
    For each $t \in RoundTrips_{s,w}$
        $TripTime_t = 2\ T_{s,w,c}$
    For $t \in Tasks_{s,w} \setminus RoundTrips_{s,w}$
        $TaskNodes_t = \{n \in Nodes_{s,w} \setminus Depots^{-}_{s,w} : Trip[n] = t\}$
        $TripTime_t = \sum_{n \in TaskNodes_t} T_{s,n,Successor[n]}$
```

Fig. 5. The Inputs for the Fleet Routing.

lax, keeping the rest of the routing fixed. The neighborhood is explored using constraint programming allowing up to $(0.15|Nodes_s|)^3$ backtracks.

5 Modeling and Algorithmic Enhancements

We now turn to some modeling and algorithmic improvements to the basic approach which bring significant benefits on real-life applications.

Customer Allocation The assignment of customers to repositories is a very important step in this algorithm because it directly determines the quality of the trips computed by the repository routing and there is no opportunity for correction. Recall that Section 4 uses

$$\sum_{i \in D}^{i=(s,d,q)} T_{s,d} * q/VC$$

as an approximation of travel distance. Our experimental results indicate that this approximation yields poor customer-to-warehouse allocation when there is an abundance of commodities. To resolve this limitation, we try to solve a slightly stronger relaxation, i.e.,

$$\sum_{i \in D}^{i=(s,d,q)} T_{s,d} * \lceil q/VC \rceil$$

but this ceiling function is too difficult for the stochastic MIP model. Instead, we decompose the problem further and separate the storage and allocation decisions. The stochastic MIP now decides which repository to open and how much of the commodity to store at each of them. Once these decisions are taken and once the uncertainty is revealed (i.e., the scenario s becomes known), we solve a customer allocation problem, modeled as a MIP (see Figure 7). This problem must be solved quickly since it is now considered after the uncertainty is revealed. Unfortunately, even this simplified problem can be time consuming to solve optimally. However, a time limit of between 30 and 90 seconds results in solutions within 1% (on average) of the best solution found in one hour. Our results indicate that even suboptimal solutions to this problem yield better customer allocation than those produced by the stochastic MIP.

```
Let:
    $Vehicles_s = \{1, 2, ..., m\}$
    $StartNodes_s = \{D_1^+, \ldots, D_m^+\}$
    $EndNodes_s = \{D_1^-, \ldots, D_m^-\}$
    $Nodes_s = StartNodes_s \cup EndNodes_s \cup \bigcup_{w \in 1..n} Tasks_{s,w}$
Variables:
    $Successor[Nodes_s] \in Nodes_s$ - Node traversal order
    $Vehicle[Nodes_s] \in Vehicles_s$ - Node vehicle assignment
    $DelTime[Nodes_s] \in \{0, \ldots, \infty\}$ - Delivery time

Minimize:
    $MAX_{n \in Nodes_s} DelTime(n)$

Subject To:
    $circuit(Successor)$
    for $n \in StartNodes_s$ such that $n = D_i^+$
        $Vehicle[n] = i$
        $DelTime[n] = Time_{s,n}$
        $DelTime[Successor[n]] = DelTime[n] + TripTime_n + T_{s,n,Successor[n]}$
    for each $n \in EndNodes_s$ such that $n = D_i^-$
        $Vehicle[n] = i$
    for $n \in Nodes_s \setminus StartNodes_s \setminus EndNode_s$
        $Vehicle[n] = Vehicle[Successor[n]]$
        $DelTime[Successor[n]] = DelTime[n] + TripTime_n + T_{s,n,Successor[n]}$
```

Fig. 6. The CP Model for Fleet Routing.

Path-Based Routing The delivery plans produced by the basic approach exhibit an obvious limitation. By definition of a trip, the vehicle returns to the repository at the end of trip. In the case where the vehicle moves to another repository next, it is more efficient to go directly from its last delivery to the next repository (assuming a metric space which is the case in practice). To illustrate this point, consider Figure 8 which depicts a situation where a customer (white node) receives deliveries from multiple repositories (shaded nodes). The figure shows the savings when moving from a tour-based (middle picture) to a path-based solution (right picture). It is not difficult to adapt the algorithm from a tour-based to a path-based routing. In the repository routing, it suffices to ignore the last edge of a trip and to remember where the path ends. In the fleet routing, only the time matrix needs to be modified to account for the location of the last delivery.

Set-Based Repository Routing The SCAP problems generated by hurricane simulators have some unique properties that are not common in traditional VRPs. One of these features appears during repository routing: The first stage solution generates customer demands that are distributed roughly uniformly through the range $0..VC$. This property allows for a repository-routing formulation that scales much better than the pure CP formulation described earlier. Indeed, if the customer demands $d_1, \ldots, d_c$, are uniformly distributed in the range $0..VC$,

Variables:

$Sent_{i\in 1..n, j\in 1..n} \in [0, Stored_i]$ - Units moved from i to j

$VehicleTrips_{i\in 1..n, j\in 1..n} \in [0..\lceil Stored_i/VC \rceil]$ - Trips needed from i to j

Minimize:

$$W_x * \sum_{i\in 1..n} (C_{s,i} - \sum_{j\in 1..n} Sent_{j,i}) +$$
$$W_y * \sum_{i\in 1..n} \sum_{j\in 1..n} T_{s,i,j} * VehicleTrips_{i,j}$$

Subject To:

$$\sum_{j\in 1..n} Sent_{i,j} \le Stored_i \quad \forall i$$
$$\sum_{j\in 1..n} Sent_{j,i} \le C_{s,i} \quad \forall i$$
$$Sent_{i,j} = 0 \quad \forall i, j \text{ where } i \text{ not in } AR_s$$
$$VehicleTrips_{i,j} \ge Sent_{i,j}/VC \quad \forall i, j$$

Fig. 7. The MIP Model for Customer Allocation.

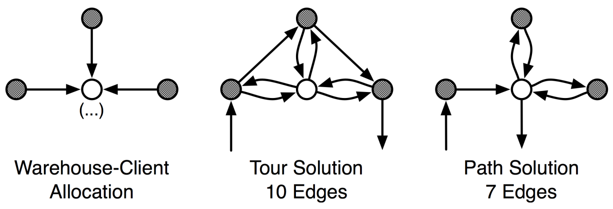

Fig. 8. Illustrating the Improvement of Path-Based Routing.

the number of sets satisfying the vehicle capacity is smaller than c^3 when c is not too large (e.g., $c \le 50$). This observation inspires the following formulation:

1. Use CP to enumerate all customer sets satisfying the capacity constraint.
2. Use CP to compute an optimal trip for those customer sets.
3. Use MIP to find a partition of customers with minimal delivery time.

This hybrid model is more complex but each subproblem is small and it scales much better than the pure CP model.

Aggregate Fleet Routing The most computationally intense phase is the fleet routing and we now investigate how to initialize the LNS search with a high-quality solution. Recall that the fleet routing problem associates a node with every trip. Given a scenario s, a lower bound for the number of trips is,

$$\sum_{i\in 1..n} StoredSent_{s,i}/VC$$

Clearly, the size and complexity of this problem grows with the amount of commodities moved. To find high-quality solutions to the fleet routing subtask, the

```
MULTI-STAGE-SCAP($\mathcal{G}$)
1  $\mathcal{D} \leftarrow StochasticStorageMIP(\mathcal{G})$
2  for $s \in 1..a$
3  do $\mathcal{C} \leftarrow CustomerAllocationProblem(\mathcal{G}_s, \mathcal{D}_s)$
4      for $w \in 1..n$
5      do $\mathcal{T} \leftarrow RepositoryPathRoutingProblem(\mathcal{G}_s, \mathcal{C}_w)$
6      $\mathcal{I} \leftarrow AggregateFleetRouting(\mathcal{G}_s, \mathcal{T})$
7      $\mathcal{S}_s \leftarrow TripBasedFeetRouting(\mathcal{G}_s, \mathcal{T}, \mathcal{I})$
8  return $\mathcal{S}$
```

Fig. 9. The Final Hybrid Stochastic Optimization Algorithms for SCAPs.

idea is to aggregate the trips to remove this dependence on the amount of commodities delivered. More precisely, we define an aggregate fleet routing model in which all trips at a repository are replaced by an aggregate trip whose service time is the sum of all the trip service times. The number of nodes in the aggregate problem is now proportional to the number of repositories. Finding a good initial solution is not important for smaller problems (e.g., $n \approx 25, m \approx 4$), but it becomes critical for larger instances (e.g., $n \approx 100, m \approx 20$). Since the aggregate problem is much simpler, it often reaches high-quality solution quickly.

The Final Algorithm The final algorithm for solving a SCAP instance $\mathcal{G}$ is presented in Figure 9.

6 Benchmarks & Results

Benchmarks The benchmarks were produced by Los Alamos National Laboratory and are based on the infrastructure of the United States. The disaster scenarios were generated by state-of-the-art hurricane simulation tools similar to those used by the National Hurricane Center. Their sizes are presented in Table 1 (The table also depicts the time limit used for fleet routing). Benchmark 3 features one scenario where the hurricane misses the region; this results in the minimum demand being zero. This is important since any algorithm must be robust with respect to empty disaster scenarios which arise in practice when hurricanes turn away from shore or weaken prior to landfall. All of the experimental results have fixed values of W_x, W_y, and W_z satisfying the field constraint $W_x > W_y > W_z$ and we vary the value of the budget B to evaluate the algorithm. The results are consistent across multiple weight configurations, although there are variations in the problem difficulties. It is also important to emphasize that, on these benchmarks, the number of trips is in average between 2 and 5 times the number of repositories and thus produces routing problems of significant sizes.

The Algorithm Implementation and the Baseline Algorithm The final algorithm was implemented in the COMET system [14] and the experiments were run on Intel Xeon CPU 2.80GHz machines running 64-bit Linux Debian. To validate our results, we compare our delivery schedules with those of an agent-based algorithm. The agent-based algorithm uses the storage model but builds a routing

Benchmark	n	m	a	Min Demand	Max Demand	Timeout
BM1	25	4	3	550	2700	30
BM2	25	5	3	6000	8384	60
BM3	25	5	3	0	11000	60
BM4	30	5	3	3500	11000	90
BM5	100	20	3	8200	22000	600

Table 1. SCAP Benchmark Statistics

Benchmark	$\mu(T_1)$	$\sigma(T_1)$	$\mu(T_\infty)$	$\sigma(T_\infty)$	$\mu(STO)$	$\sigma(STO)$	$\mu(CA)$	$\mu(RR)$	$\mu(AFR)$	$\mu(FR)$
BM1	196.3	18.40	78.82	9.829	0.9895	0.5023	11.78	0.2328	23.07	30.00
BM2	316.9	59.00	120.2	20.97	0.5780	0.2725	16.83	0.2343	28.33	60.00
BM3	178.4	15.89	102.1	15.02	0.3419	0.1714	7.192	0.1317	11.98	40.00
BM4	439.8	48.16	169.0	22.60	0.9093	0.4262	22.71	0.2480	33.28	90.00
BM5	3179	234.8	1271	114.5	46.71	25.05	91.06	1.0328	351.7	600.0

Table 2. SCAP Benchmark Runtime Statistics (Seconds)

solution without any optimization. Each vehicle works independently to deliver as much commodity as possible using the following heuristic:

```
Greedy-Truck-Agent()
1  while ∃ commodity to be picked up ∧ demands to be met
2      do if I have some commodity
3            then drop it off at the nearest demand location
4            else  pick up some water from the nearest warehouse
5  goto final destination
```

This agent-based algorithm roughly approximates current relief delivery procedures and is thus a good baseline for comparison.

Efficiency Results Table 2 depicts the runtime results. In particular, the table reports, on average, the total time in seconds for all scenarios (T_1), the total time when the scenarios are run in parallel (T_∞), the time for the storage model (STO), the client-allocation model (CA), the repository routing (RR), the aggregate fleet routing (AFR), and fleet routing (FR). The first three fields(T_1, T_∞, STO) are averaged over ten identical runs on each of the budget parameters. The last four fields (CA, RR, AFR, FR) are averaged over ten identical runs on each of the budget parameters and each scenario. Since these are averages, the times of the individual components do not sum to the total time. The results show that the approach scales well with the size of the problems and is a practical approach for solving SCAPs.

Quality of the Results Table 3 depicts the improvement of our SCAP algorithm over the baseline algorithm. Observe the significant and uniform benefits of our approach which systematically delivers about a 50% reduction in delivery time.

Table 4 describes the correlations between the distances in the customer allocation and fleet routing models. The results show strong correlations, indicating

Benchmark	BM1	BM2	BM3	BM4	BM5
Improvement(%)	57.7	40.6	68.8	51.7	50.6

Table 3. Improvements over the Baseline Algorithm.

Benchmark	BM1	BM2	BM3	BM4	BM5
Correlation	0.9410	0.9996	0.9968	0.9977	0.9499

Table 4. Correlations for the Distances in Customer Allocation and Fleet Routing.

that the distances in the customer allocation model are a good approximation of the actual distances in the fleet routing model. Table 5 also reports results on the absolute and relative differences between vehicles in the solutions. They indicate that the load is nicely balanced between the vehicles. More precisely, the maximum delivery times are often within 10% of each other on average, giving strong evidence of the quality of our solutions. Benchmark 5 is an exception because it models emergency response at a state level, not at a city level. In that benchmark, some vehicles have a significantly reduced load because they would have to travel to the other side of the state to acquire more load, which would take too much time to reduce the maximum delivery objective.

Behavioral Analysis Figure 10 presents the experimental results on benchmark 5 (other benchmarks are consistent, but omitted for space reasons). The graph on the left shows how the satisfied demand increases with the budget while the graph on the right shows how the last delivery time changes. Given the weight selection, it is expected that the demand and routing time will increase steadily as the budget increases until the total demand is met. At that point, the demand should stay constant and the routing time should decrease. The results confirm this expectation. The experimental results also indicate the significant benefits provided by our approach compared to the baseline algorithm.

Fleet Routing Figure 11 presents experimental results comparing aggregate (AFR), tour-based (TFR), and path-based (PFR) fleet routing (Only BM1 is presented but other results are consistent). The key insight from these results is to show the benefits of allowing the trips of a repository to be performed by multiple vehicles. Note also the significant improvements obtained by considering paths instead of tours.

Customer Allocation As mentioned earlier, the benefits of separating customer allocation from storage decisions are negligible when the budget is small. However, they become significant when the budget increases and can produce a reduction by up to 16% of the expected maximum delivery time.

7 Conclusion

This paper studied a novel problem in the field of humanitarian logistics, the Single Commodity Allocation Problem (SCAP). The SCAP models the strategic planning process for disaster recovery with stochastic last mile distribution.

Benchmark	BM1	BM2	BM3	BM4	BM5
Absolute Difference	6.7	59.4	39.5	49.1	749
Relative Difference(%)	10.7	12.8	6.7	8.7	46.2

Table 5. The Difference in Delivery Times Between Vehicles.

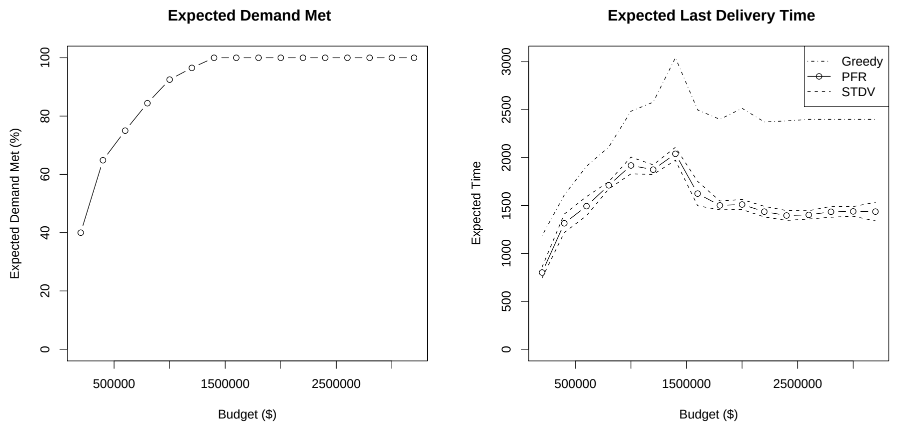

Fig. 10. Varying the Budget on Benchmark 5

The paper proposed a multi-stage stochastic hybrid optimization algorithm that yields high quality solutions to real-world benchmarks provided by Los Alamos National Laboratory. The algorithm uses a variety of technologies, including MIP, constraint programming, and large neighborhood search, to exploit the structure of each individual optimization subproblem. The experimental results on water allocation benchmarks indicate that the algorithm is practical from a computational standpoint and produce significant improvements over existing relief delivery procedures. This work is currently deployed at LANL as part of its mission to aid federal organizations in planning and responding to disasters.

References

1. M. Albareda-Sambola, J. A. Diaz, and E. Fernandez. A compact model and tight bounds for a combined location-routing problem. *Computer & Operations Research, 32:407-428*, 2005.
2. B. Balcik, B. Beamon, and K. Smilowitz. Last mile distribution in humanitarian relief. *Journal of Intelligent Transportation Systems*, 12(2):51–63, 2008.
3. Glay Barbarosoglu, Linet zdamar, and Ahmet evik. An interactive approach for hierarchical analysis of helicopter logistics in disaster relief operations. *European Journal of Operational Research*, 140(1):118 – 133, 2002.
4. B. Beamon. Humanitarian relief chains: Issues and challenges. *34th International Conference on Computers & Industrial Engineering*, pages 77–82, 2008.
5. L. Bianchi, M. Dorigo, L.Gambardella, and W. Gutjahr. A survey on metaheuristics for stochastic combinatorial optimization. *Natural Computing*, 8(2), 2009.
6. Ann Melissa Campbell, Dieter Vandenbussche, and William Hermann. Routing for relief efforts. *Transportation Science*, 42(2):127–145, 2008.

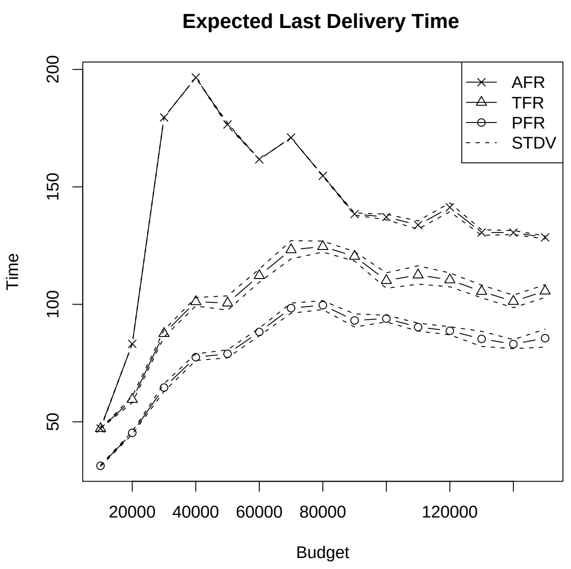

Fig. 11. Comparing Various Algorithms for Fleet Routing.

7. L. I. Burke D. Tuzun. A two-phase tabu search approach to the location routing problem. *European Journal of Operational Research, 116:87-99*, 1999.
8. Serhan Duran, Marco Gutierrez, and Pinar Keskinocak. Pre-positioning of emergency items worldwide for care international. *submitted to Interfaces*, 2008.
9. Fritz institute. http://www.fritzinstitute.org, 2008.
10. United States Government. The federal response to hurricane katrina: Lessons learned, 2006.
11. P. Griffin, C. Scherrer, and J. Swann. Optimization of community health center locations and service offerings with statistical need estimation. *IIE Transactions*, 2008.
12. D. Gunnec and F. Salman. A two-stage multi-criteria stochastic programming model for location of emergency response and distribution centers. In *INOC*, 2007.
13. Peter Kall and Stein W. Wallace. *Stochastic Programming (Wiley Interscience Series in Systems and Optimization)*. John Wiley & Sons, 1995.
14. Comet 2.1 User Manual. Dynadec website. http://dynadec.com/.
15. G. Nagy and S. Salhi. Nested heuristic methods for the location-routing problem. *Journal of Operational Research Society, 47:1166-1174*, 1996.
16. Paul Shaw. Using constraint programming and local search methods to solve vehicle routing problems. In *CP'98*, pages 417–431, 1998.
17. Paolo Toth and Daniele Vigo. *The Vehicle Routing Problem*. SIAM Monographs on Discrete Mathematics and Applications, Philadelphia, Pennsylvania, 2001.
18. L. Van Wassenhove. Humanitarian aid logistics: supply chain management in high gear. *Journal of the Operational Research Society*, 57(1):475–489, 2006.