The Consistent Vehicle Routing Problem

Chris Groër, Bruce Golden, Edward Wasil

University of Maryland, College Park
University of Maryland, College Park
American University, Washington D.C.

June 12, 2008
VIP'08 Conference
Oslo, Norway

The Classical VRP

- In the classical capacitated VRP, we attempt to find the lowest cost set of routes that meets all customer demand and satisfies vehicle constraints (capacity and total travel time)
- The VRP has been very well-studied over the past 30 years and many variants have been proposed
  - The VRP with Time Windows - each customer must receive service during a certain time period
  - The Period VRP - service occurs over a period of several days and some customers must be serviced multiple times during each period

The Consistent VRP

- In 2006, UPS proposed a new variant that incorporates a customer service component into a Periodic VRP
  - Customer service trumps travel cost
- We are given D days worth of service requirements in advance
- We must generate efficient routes for each day subject to the typical VRP constraints

The Consistent VRP

- We must also satisfy two additional constraints that improve customer service quality
- Each customer must always be visited by the same driver
  - Over time, frequently serviced customers develop a relationship with their driver
- Each customer must receive service at *roughly* the same time each day
  - Customers plan their activities around the driver's expected arrival time

Outline of Talk

- Briefly mention an exact integer programming formulation
- Present a heuristic solution method
- Present some computational results
- Extending the planning horizon

Exact Formulation

- We solved small instances to optimality
- 12-node, 3 day problems required up to several days of computing time using CPLEX 11.0
- To solve problems of practical size involving thousands of nodes, we turn to a heuristic approach

A Simple Guiding Principle

- We attempt to provide consistent service by adhering to a very simple idea
- If customers a and b are serviced by the same driver on some day and a is visited before b, then the same driver must visit a and b in this order whenever they both require service
- By adhering to this idea, the *same driver* constraint is always met
- We hope that this strategy will also lead to consistent service times as well
- We refer to this as the *precedence principle*

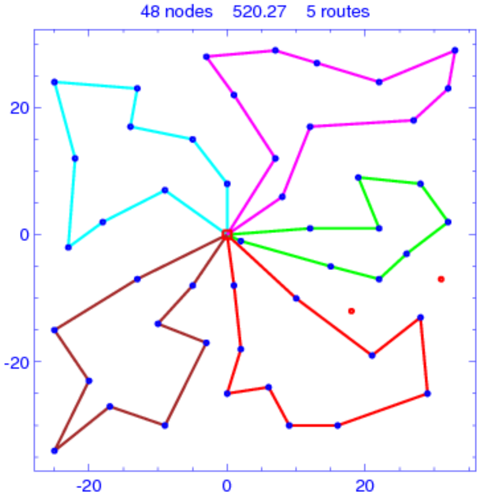
A Heuristic Algorithm

- Our idea is to construct a set of *template routes* that adhere to the *precedence principle*
- The *template routes* consist only of those customers that require service on more than one day

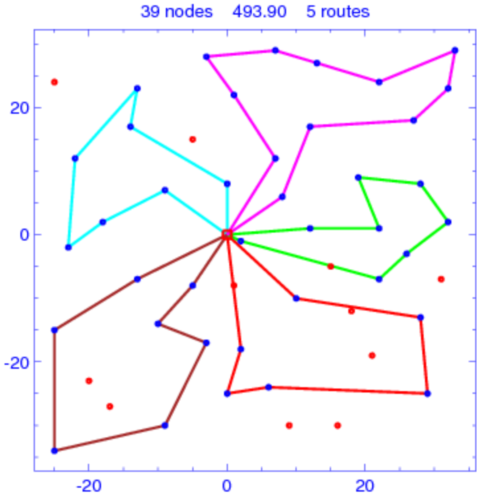
A Heuristic Algorithm

- The routes for day d can be constructed from the template using a simple two-step procedure
  1. Remove from the template all customers not requiring service on day d
  2. Insert all customers that require service only on day d
- The resulting routes for each day are guaranteed to adhere to the consistent driver constraint
- If the number of insertions isn't too large, then we expect consistent service times as well

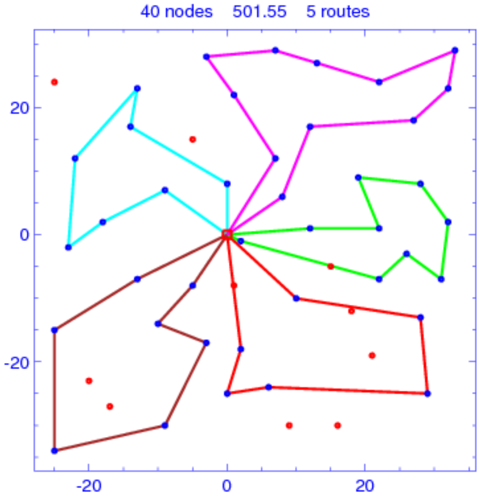
Example Template Routes



Routes for Day d After Removals



Final Routes for Day d After Insertions



Constructing the Template

- The main difficulty in constructing the template is how to interpret the vehicle capacity and travel time constraints
- The template is never actually traversed by a vehicle
- Our strategy is to use a VRP metaheuristic to construct the template and then periodically attempt to construct the routes for each day using the removal and insertion procedure
- We then modify the vehicle capacity and travel time limits for the template if we find violations or excessive slack in the daily routes
- By periodically modifying these template limits, we hope to generate high-quality routes for each day

Outline of Heuristic Algorithm

- We embed this constraint modification procedure into the well-known *Record-to-Record Travel* algorithm

1. Construct an initial template that leads to feasible solutions for each day
2. Repeatedly improve and diversify template using the *Record-to-Record Travel* algorithm, periodically modifying the template limits when we encounter daily routes that are either infeasible or have excessive slack
3. Once a stopping criteria is met, return to the template that led to the lowest cost routes for all D days, and return the set of corresponding daily routes

Constructing the Initial Solution

1. Make an initial estimate of the template capacity and total travel time limits
2. Construct an initial template by assigning a single vehicle to all customers that require service on more than one day
3. Create an initial solution using the Clarke-Wright algorithm
4. Construct the routes for all D days using the removal and insertion procedure
    - If some are not feasible then decrease the violating template limit and return to Step 3

- We now have an initial template that leads to feasible routes for all D days

Improving the Solution

- We use three well-known local search operators to modify existing solutions

1. One Point Move
2. Two Point Move
3. Two Opt

Improving the Solution

- Diversification: Apply local search operators to the template
  - Accept all improving moves and those deteriorating moves that do not worsen total template length by more than a threshold
- Intensification: Apply local search operators to the template, allowing only improving moves
- Construct the routes for the D days
  - If all are feasible, store this template and increase the current template limits
  - If we have violations, revert to most recent feasible template and reduce the current template limits
- If we have reached the same local minimum K times, then revert to the template that led to the lowest cost set of routes across D days, construct these routes for each of the D days, and return
  - Otherwise go back to the Diversification phase

Computational Results: Small Problems

- We constructed a set of 10 small problems and solved them exactly with CPLEX and approximately with our heuristic
- 3 days of service requirements, 10 nodes or 12 nodes
- The heuristic found optimal solution to 6 of the 10 problems, gap averaged less than 3% in other cases
- 9 of the 10 optimal solutions adhered to the *precedence principle*
- Running time of heuristic is less than one second

Computational Results: Simulated Problems

- We simulated a set of 5-day problems where we varied the probability of customers receiving service
- If this probability is high, then we expect the template to lead to very good routes for the individual days
- If the probability is low, then the template will have to undergo substantial modification in order to create the daily routes, and we expect the quality to suffer

Computational Results: Simulated Problems

- Instances generally have 700 total customers and the constraints are designed so that we have 100-150 customers on a route each day, mimicking the routes of a typical package delivery company
- We varied the daily service probability p from 0.6 to 0.9 and generated 5 problems for each value of p
- We generated a set of routes for each day without accounting for consistency
- This gives us some idea of the *cost of consistency*

Computational Results: Simulated Problems

	Consistent Routes				Inconsistent Routes	
p	Avg. Service Time Differential	Max. Service Time Differential	Total Travel Time	Number of Routes	Total Travel Time	Mean Number of Routes
0.6	9.6	30.8	6795.0	5	6425.8	4.88
0.65	8.6	31.0	7337.8	5	6667.6	5.00
0.7	7.8	22.4	7714.8	6	7180.2	5.32
0.75	7.2	24.6	7785.0	6	7356.0	5.92
0.80	5.8	16.2	8222.6	6	7698.4	6.00
0.85	5.8	17.2	8535.0	7	8140.4	6.52
0.9	4.4	11.8	8761.4	7	8321.2	7.00

Computational Results: Simulated Problems

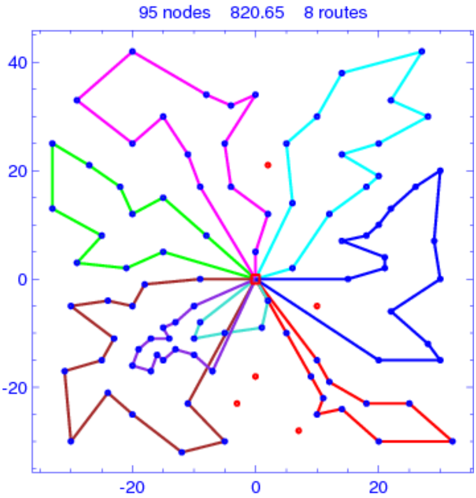
- As expected, the service time differentials decrease as p increases
- Accounting for consistency causes a total travel time increase of between 5 and 10%
- Inconsistent routes occasionally require fewer vehicles
- Running times less than five minutes

Computational Results: Benchmark Problems

- We also generated a set of 12 Consistent VRP benchmark problems using the Christofides problems for the classical VRP
- The service time differentials were again quite small relative to the total allowed vehicle travel time
- Solutions generated without regard for consistency require on average 15% less total travel time
    - This difference is larger than for simulated problems and is due to a more frequent reduction in the number of vehicles required
- Running times less than 2 minutes

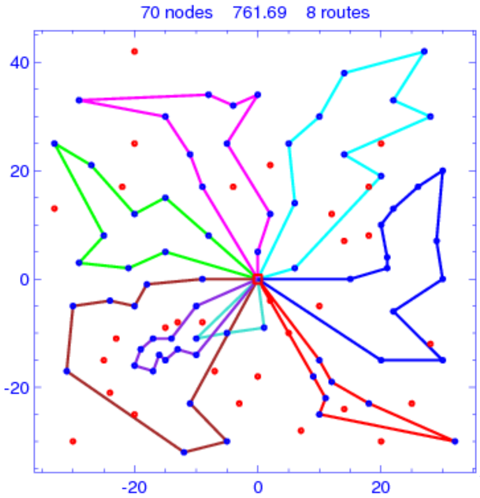
Example Benchmark Solution: Christofides 3

Template



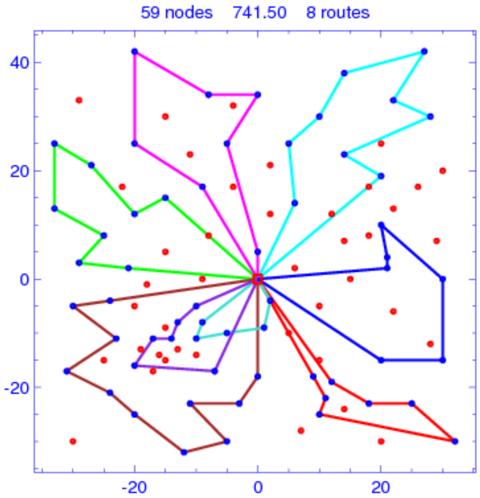
Example Benchmark Solution: Christofides 3

Day 1



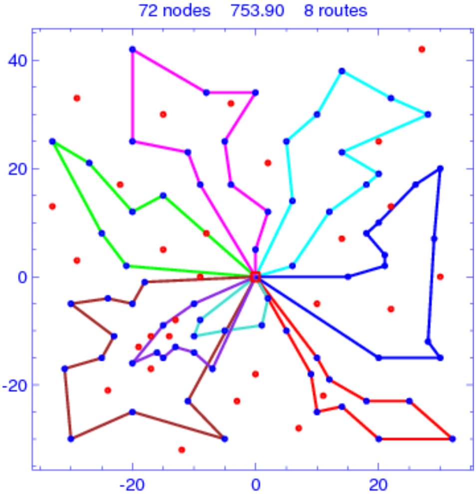
Example Benchmark Solution: Christofides 3

Day 2



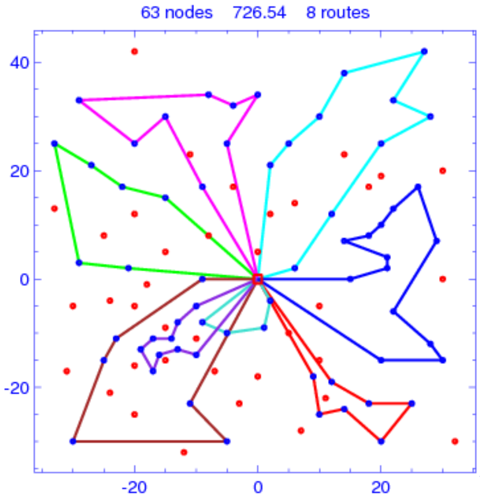
Example Benchmark Solution: Christofides 3

Day 3



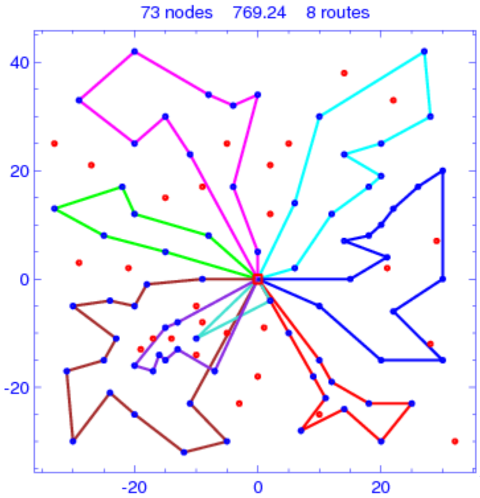
Example Benchmark Solution: Christofides 3

Day 4



Example Benchmark Solution: Christofides 3

Day 5



Computational Results: UPS Data

- We were provided with 5 weeks of data from UPS
  - 3715 total customer locations
  - Travel time matrix
  - Demand amounts
  - Service times

Computational Results: UPS Data

- Interesting properties of the data set

Week	Mean Number of Stops Per Day	Number of Customers With k Stops $k=1$	$k=2$	$k=3$	$k=4$	$k=5$	Number of Template Customers
1	597	838	213	100	60	132	505
2	591	801	215	98	58	133	504
3	566	755	216	84	52	135	487
4	573	807	219	96	44	123	482
5	572	818	201	94	43	130	468

- Most customers that require service during a week are visited only once
- In general, we will make more insertions than removals when creating the daily routes

Computational Results: UPS Data

	Consistent Routes			Inconsistent Routes
Week	Average Maximum Differential	Overall Maximum Differential	Total Travel Time	Total Travel Time
1	19	64	6206	6107
2	29	101	6064	5998
3	24	81	5794	5755
4	35	176	5959	5910
5	25	85	5828	5777

- The mean maximum differentials are low
- The overall maximum differentials suffer due to a larger number of insertions than in the simulated problems
- The total travel times of the consistent routes are only 2% more costly

What happens after D days?

- If we have provided consistent service over a single D-day period, we would like to continue this trend
- We used the first four weeks (20 days) of UPS data to create a set of template routes
    - Customers must require service on 4 of the 20 days to be included in the template
- We then used this template to generate consistent routes for the fifth week

Extending the D-day planning horizon

- We were able to provide consistent service for customers visited during the fifth week using this template
- Template created without knowledge of the fifth week

Total Travel Times			
Day	Routes derived from week 5 template	Routes derived from weeks 1-4 template	Inconsistent Routes
1	1190	1197	1183
2	1132	1136	1119
3	1147	1164	1136
4	1133	1137	1124
5	1226	1265	1214

Extending the D-day planning horizon

- Total travel times of these routes are only 1.2% longer than consistent routes created using a week five template and only 2.1% longer than inconsistent routes
- Looked at maximum service time differentials for customers requiring two or more visits in week five
  - Mean maximum service time differential is 68 minutes
  - Overall maximum service time differential is about 3 hours - due to several non-template customers being visited more than once
- Overall encouraging results - different weeks of UPS data are similar enough to allow for the same template to be used to generate consistent routes

Conclusion

- New VRP variant motivated by real-world customer service considerations
- We have developed exact and heuristic solution methods
- Our heuristic method appears quite effective and is guided by a simple idea
- We are generally able to generate routes that provide consistent service with a relatively small increase in total travel time