Modern Approaches to the Rich Vehicle Routing Problem

Henry F. R. Fellows
Division of Science and Mathematics
University of Minnesota, Morris
Morris, Minnesota, USA 56267
fello056@morris.umn.edu

ABSTRACT

The Rich Vehicle Routing Problem is a class of problems that revolve around finding the most optimal route for a certain set of deliveries. Obtaining approximate solutions to these computationally hard problems is both economically valuable and Henry: academically interesting?. Despite a long history of research in the field, new methods and approaches to specific subproblems are frequently introduced. This paper summarizes the problem, its applications, recent algorithms and approaches.

Keywords

Vehicle Routing Problem, Black-box optimization, Genetic Algorithms, Probability Collectives, Distributed Systems, Henry: Reviewers! skip to the end to read notes!

1. INTRODUCTION

Routing is a deeply pervasive feature of the modern world. From the point-to-point navigation of everyday driving to the shipping of bulk freight, the task of creating those routes has been increasingly automated. The long term trend towards self-driving vehicles means that computers will soon control the entire process of transportation; aside from selecting a departure and destination, humans need no longer be involved. Software to create routes for school buses, mail deliveries and garbage trucks have been in use for decades; everything from Amazon to Uber uses these systems on a daily basis. The problem of finding efficient routes quickly is still an ongoing problem in the real world. The exact impact of improvements in routing is difficult to measure, but in 2011, the external costs of traffic jams in the EU were 1-2% of GDP[3]. A small improvement - say five percent - in routing efficiency represents a staggering amount of money. The general case of routing is the traveling salesman problem. The traveling salesman problem asks for the shortest route which passes through each point once. Assuming that each pair of points is connected by a link, the number of
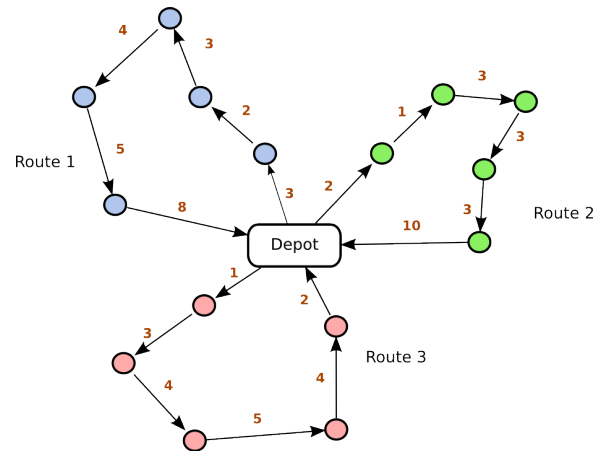
UMM CSci Senior Seminar Conference, December 2015 Morris, MN.

Figure 1: Example of the VRP
Henry: Needs to replaced w/own image.

potential routes is $\frac{1}{2}n!$, which grows extremely quickly, and is significant even for small values. For $n = 10$, the number of routes reaches 1814400. The traveling salesman problem is in a class of problems known as NP-complete. Although it has not yet been proven, it is considered likely that there is no algorithm for NP-complete problems that allows them to be solved quickly. Fortunately, solutions of NP-complete problems are easy to verify. In practice, the traveling salesman problem is too abstract for real-world routing problems, and so a variant was proposed that better represents the challenges of routing vehicles.

1.1 Routing in the real world

Routing in the real world is often a complicated and messy affair. The vehicle has to have enough capacity for all of its deliveries, and often, customers expect that the vehicle arrives near a specified time. New stops may be added during the day, or the exact demand of each customer might not be known ahead of time. In some cases, the customer might not know how much of a product they need until the vehicle arrives. Dealing with these types of problems is the domain of the Rich Vehicle Routing Problem.

2. THE VEHICLE ROUTING PROBLEM

Originally titled the Truck Dispatching Problem, the original formulation of the Vehicle Routing Problem (VRP) was created by G. B. Danzig and J. H. Ramser in 1959[4]. The

premise of the problem is that each vehicle (or truck), has a limited capacity and the fleet must make deliveries to as many customers as possible, starting from a specific location known as a depot[3]. Alternatively, the vehicles may have to make multiple deliveries in order to satisfy all of the customers. A measure that is also considered worth minimizing is the maximum tour length - the longest single route taken by one truck. Danzig and Ramser note that if the total capacity of the vehicles is less than the total demand of the customers, the problem is mathematically identical to the traveling salesman problem. There has been a great deal of research done on the original VRP, and recent research focuses on versions of the VRP with different constraints or multiple constraints simultaneously. These variations of the Vehicle Routing Problem are collectively referred to as the Rich Vehicle Routing Problem[3] (RVRP). In the following sections of, we discuss prominent variants of the RVRP.

2.1 Decentralized Vehicle Routing Problem

A common problem in rich vehicle routing literature is decentralized control, where each vehicle or depot is modeled as having an independent agent who makes decisions out of self-interest. However, pure self-interest can cause decisions that degrade the solutions of others. Imagine a vehicle which finds a route that fulfills a large number of deliveries extremely quickly; other vehicles might find that their routes force them to cross through this area where they cannot make any deliveries along the way. While it is highly efficient for the first vehicle, it is less efficient for all other vehicles. The ideal is to pursue optimal routes - both on the local (individual) scale, and on a global scale. Finding good methods of making decisions that allow for both autonomy and global utility is an ongoing research question.

2.2 Vehicle Routing Problem with Time Windows

Another variant of the VRP is the vehicle routing problem with time window (VRPTW). The VRPTW describes the situation where the customer expects that the deliveries take place within a certain time interval. Unlike the estimates provided by shipping services, these constraints are set by the customers. A simple example is the constraint that deliveries are only accepted during business hours. This adds the dimension of time to the problem - the time it takes the truck to traverse nodes and to complete the delivery must be considered. The VRPTW is one of the more popular subproblems because deliveries without implicit or explicit time constraints are rare. In practice, the time window constraint is rarely found without other constraints; industrial settings tend to consider the VRPTW to be the base problem for real-world settings.

2.3 Approaches

In general the approaches used to solve the VRP are approximate methods; they intend to find 'good', not perfect solutions. The algorithms tend to use simple rules to guess solutions, and the process of narrowing down or perfecting these guesses is known as optimization. Henry: History, maybe?. Most modern methods of solving the VRP belong to the blackbox style of optimization; blackbox optimization is used when a problem does not have a formal algebraic model, or the model is too computationally expensive [1]. The VRP does have a very good model, but it is extremely computationally expensive to use it, and so the natural choice is to use blackbox methods. The common feature of blackbox methods is the use of stochastic (random) elements, especially in selecting starting states. This makes blackbox methods into a double edged sword; they are often effective at solving otherwise intractable problems, but the solutions frequently fail to produce any insight into the problem. Stochastic decision making makes it hard to determine why the algorithm provides a specific solution.

Henry: Maybe: The preference for blackbox methods may be a reflection of the difficulty in finding/improving [non-guessing] algorithms (or, perhaps equivalently, of the inherent computational limitations of calculating a perfect answer to the VRP)

3. GENETIC AND MEMETIC ALGORITHMS

A genetic algorithm (GA) is a type of problem-solving that is inspired by natural selection. It is often used to find solutions for processes that are not well understood, but can be modeled well. In the case of the VRP, the problem is both known and modeled well, but the approximate solution given by a well-made genetic is often nearly as good, and much faster than computing the exact solution. The fundamental notion of a genetic algorithm is a *genetic representation*, which is a means of representing an individual. By analogy, the representation is the genome of the individual. A common representation is a fixed-length array of bits, but other representations are possible. These genetic representations are manipulated by genetic operators that simulate various processes. A mutation operator provides the analogue to biological mutation by altering values in the genetic representation; the traditional example is a function that flips bits based on a random number generator. The crossover operator represents reproduction - it combines traits from two (usually) individuals to produce a new individual. Genetic algorithms aim to improve their solutions, and they measure how 'good' a solution is by using a fitness function. Fitness functions are problem specific; in VRP, they measure if the individual solves the problem, and how good that solution is.

Algorithm 1: Genetic Algorithm Pseudocode

Initialize population;
while *not final state* **do**
    Select best individuals via fitness function;
    Generate new individuals using genetic operators;
    foreach *new individual* **do**
        if *better than worst individual* **then**
            Replace that individual;
        end
    end
end
return best solution;

The general form of a GA begins with initializing the population. The canonical example is creating individuals with random representations. Next, the algorithm enters its main loop; the loop is usually terminated when an iteration limit is reached, but other conditions are sometimes used that are specific to the problem. In the loop, the individuals are scored using the fitness function and the best are set aside
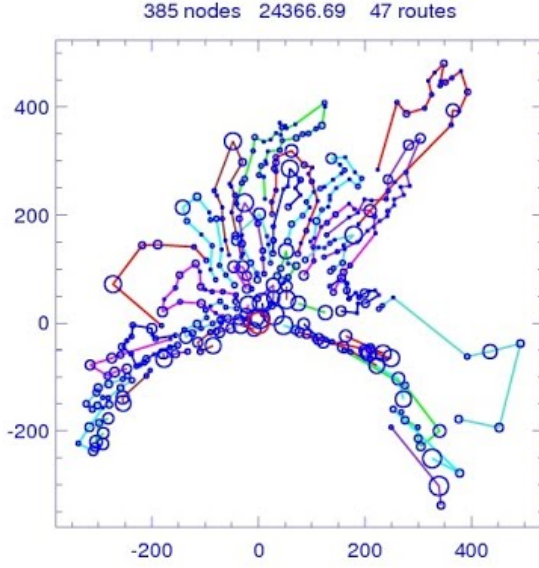
385 nodes   24366.69   47 routes

Figure 2: To be example of selecting good/bad regions

Henry: Needs to replaced w/own image.

Algorithm 2: Human Assisted Genetic Algorithm

Initialize population;
while *number of interactions without improvement* $< lt_{NI}$*, and time* $< T_{max}$ **do**

  Coherent algorithm goes here!;
  Genetic algorithm stuff!;
  Human selection suff!;
  If good tags ;
  - higher selection probability;
  - mutation operator not allowed;
  If bad tags;
  - lower selection probability;
  - mutation operator encouraged;
  Standard heuristic function;
  Crossover;

end
return best feasible solution;

---

for the next stage. The crossover operator is then applied to this group, and the mutation operator on the resulting individuals. Then, it evaluates each individual and replaces the worst members of the population with the new individuals. This process is repeated until the termination condition is reached.

3.1 Human Assisted Routing

Henry: Not sure if it belongs here or in own section. Uses a GA as base algorithm.

Humans tend to be very good at solving visual problems, and this extends to the routing if the problem is displayed appropriately. We have a talent for deciding whether a given route is good or bad by simply looking at a diagram. Amazon's mechanical turk and similar platforms were created to allow developers to make use of human intelligence in solving computationally complex problems. Pursuing the approach of using humans to do things humans are good at anyways, S. Ismail, F. Legras and G. Coppin[2] created a genetic algorithm that uses humans to determine how good a particular solution is.

Henry: describe in words, reference image

3.2 Hybrid Genetic Search with Advanced Diversity Control

Adding time window constraints to customer demand and depot availability poses a significant challenge, especially in relation to the smaller proportion of solutions and the increased computational complexity caused by the increased search space. The opposing demands of temporal and spatial constraints is a particularly frustrating problem in the VRPTW.

The Hybrid Genetic Search with Advanced Diversity Control (HGSADC) address some of these challenges in the VRPTW, especially in route-duration constraints. The primary feature of the HGSADC is the different approach to diversity management in it's population. HGSADC adds diversity to its objectives as a term to be optimized, which allows it to avoid dead-end solutions. The algorithm is considered the current state of the art in the multi-depot vehicle routing problem with time windows[8].

3.2.1 Algorithm and mechanics

HGSADC is a complex algorithm, and cannot be fully described in this paper. What follows is a summary of the algorithm and the most notable features.

Henry: This algorithm is actually frighteningly complex. It goes on for five pages, and it doesn't go into great detail, either. Also, the pseudocode should be above this HGS-DAC evolves infeasible and feasible solutions as two separate subpopulations. Normally, infeasible ('bad') solutions are removed from the population, but in the HGSADC, they are kept in a seperate subpopulation. The rationale is that this allows the algorithm to create a reserve of genetic diversity that can be used to get the population out of dead ends. Two random individuals are pulled from the combined population and the best of them is chosen to be a parent. The crossover operator is an applied to two parents, and the resulting child is educated. The education process is a limited local search that seeks to improve the child; if the child is still infeasible after this procedure, the repair procedure may be called. P_{rep} is the probability that the repair operator is called; repair is a very intensive local search. Finally, the child is placed into an appropriate subpopulation. If a subpopulation exceeds a maximum size, a survivor selection stage is triggered. A number of individuals is removed until the population returns to the nominal population size μ. The individual that is most similar in terms of fitness score and route to another individual is called the clone. Survivor selection iteratively removes the clone until the population size is returned to μ. The purpose of this style of population management is to maintain diversity. A diversification round may be started if there has been It_{div} iterations without improvement. Henry: More when I understand it.

3.2.2 Results

It's crazy good. Henry: More to come, but to say that it blows everything else out of the water is an understatement.

Algorithm 3: Hybrid Genetic Search with Advanced Diversity Control

---

Initialize populations at 4μ size;
while *number of interactions without improvement $< lt_{NI}$, and time $< T_{max}$* **do**
    Select parent solutions P_1 and P_2;
    Create child C from P_1 and P_2 (crossover);
    Educate C (local search procedure);
    if *C infeasible* **then**
        Insert C into infeasible subpopulation;
        Repair with probability P_{rep};
    end
    if *C feasible* **then**
        Insert C into feasible subpopulation;
    end
    if *maximum subpopulation size, μ, reached* **then**
        Select survivors;
    end
    if *best solution not improved for It_{div} iterations* **then**
        Diversify population;
    end
    Adjust penalty parameters for infeasibility;
    if *number of iterations modulo$It_{dec} = 0$* **then**
        Henry: aka every n iterations;
        Decompose the master problem;
        Use HGSADC on each subproblem;
        Reconstitute three solutions, and insert them in the population;
    end
end
return best feasible solution;

---

4. AGENT-BASED MODELS

Agents are simply small decision making elements that typically have some internal state and the ability to make decisions based on this internal state. In optimization, a typical goal is for an agent to attempt to achieve some sort of optimal state. Agents (should) take actions to improve their state, and we describe this as the agent acting in its own interest. The function that returns the quantification of this local optimization is termed the local utility function, in contrast to the global, or system utility function. The global utility function describes how optimal the system is, which can be quite different from the local utility. Henry: Make up definition? Agents are very poorly defined in literature. We have chosen to discuss two major approaches in modeling individuals for the decentralized VRP. The first is agent-based models, the more traditional approach, and the second is probability collectives, a relatively recent approach that has a legacy in genetic algorithms.

4.1 Distributed reverse Vickrey auctions

Saleh et al.[6] were examining a version of the DCVRP where each depot and customer are treated as agents. The authors were interested in designing a system that encouraged near-optimal solutions while still being completely self-interested. They pursued this by building mechanisms that encouraged depots to offer an accurate estimate of minimum costs while still allowing the cost functions of each depot to remain private. Henry: neat reverse vickrey auction info goes here once I understand game theory.
Henry: Maybe add very short pseudocode.

The core feature of the algorithm is a reverse vickrey auction. Each depot k submits a bid to each unassigned customer j where $q_k^i \geq d_z$. Bids consist of the estimated costs of servicing the customer. The customer chooses the lowest bid satisfying their demands and notifies k^*. The depot k^* may receive responses from other customers. From the given responses, the depot chooses the customer j^* with the lowest insertion cost (the customer whose addition to the vehicle's route will cause the least change in the routing cost). Once the customer is assigned to a depot, the customer j^* submits payment to depot k^*. The payment is equal to the second lowest offer of the game. This, combined with the fact that bids are sealed, encourages depots to provide the lowest possible rate (because if they win, they are still paid more than the estimate). The completion of these steps is a single auction round, and many rounds may take place until every customer is assigned. Once the auction rounds are complete, the depot can apply other optimization strategies to its routes, or in the case of small routes, an exact solution.

4.2 Probability Collectives

Probability Collectives (PC) takes a different approach to selecting a good solution. Instead of attempting to evolve an exemplary individual in a population in the method of genetic or memetic algorithms, probability collectives selects optimal strategy for each agent[5]. A PC agent is a self-interested, learning individual that selects a strategy with the highest probability of optimizing the local and global objectives. In contrast to the Distributed Reverse Vickrey Auction, the agents willingly and freely share information about their strategies, and the cost function is global.

4.2.1 Mechanics of Probability Collectives

The general form of PC in the context of VRP is as follows, along with the flowchart below. Imagine you have N agents, which can be depots or trucks, each having a set X of strategies, of (potentially variable) length m. The strategies are most often routes in RVRP applications[7], but can sometime be as large as entire sets of routes or as small as individual actions. The set is represented as $X_i = \{X_i^{[1]}, X_i^{[2]}, ..., X_i^{[m]}\}$.

A strategy for agent i is denoted as $X_i^{[r]}$, where r is an identifier for that strategy. For each agent, assign a uniform probability, $1/m_i$ to all actions; the resulting probability of that strategy is signified by $q(X_i^{[r]})$. The agents then selects a random action r and a sampling of random actions from other agents. The resulting set, the 'combined strategy set', $Y_i^{[r]} = \{X_1^{[?]}, ..., X_i^{[r]}, ..., X_N^{[?]}\}$, represents a guess as to a potential future. Accordingly, for each of the strategy sets $Y_i^{[r]}$, compute the expected local utility - the measure of how good the solution is expected to be for the agent - using the following measure:

$$Exp.\ Utility\ of\ Agent\ i^r = q_i^r \prod_{(i)} q(x_{(i)}^{[?]}) \cdot G(Y_i^{[r]}) \qquad (1)$$

Where q_i^r represents the probability of action r for vehicle i, (i) is the set of all agents excluding i, and G is the function that computes global utility for a given set of strategies. G is problem specific, but in the RVRP it could be a measure of unused capacity, unvisited destinations, maximum tour length or various combinations of measurements.

After computing the local utility, the next step is to update the probability of each action for all agents as follows:

$$q(X_i^{[r]}) < -q(X_i^{[r]}) - \alpha_{Step} \cdot q(X_i^{[r]}) \cdot k_r^{update} \qquad (2)$$

where k is the iteration, α_{step} is a constant that controls the amount of change each step. Typical values range are around 0.98.

$$k_r^{update} = \frac{Contrib.\ of\ Agent\ i}{T_k} + S_i(q) + ln(q(X_i^{[r]})) \qquad (3)$$

Here, temperature T_k is a scalar that represents the relative importance of the of the contribution of agent i over time. It is computed as $T_k = T_0/ln(k)$, and T_0 is the initial temperature. A example value might be 30. Henry: So, the PC papers reference an α_T, which might be jammed in the numerator to add a rate control. Might just skip. The contribution of Agent i is then:

$$Contrib.\ Agent\ i = Exp.\ Utility\ Agent\ i^r - Exp.\ Global\ Utility \qquad (4)$$

The $S_i(q)$ term is the entropy of the combined strategy set. In information theory, Entropy is the average value of the information in a message. Probability collectives treats the combined strategy set as a message in order to find the set with the highest Entropy. As it increases, the probability distribution more clearly distinguishes the contribution of each strategy toward optimizing the expected global utility. When it reaches a maximum, it represents the set that has the most information about the probabilities of each strategy. Entropy is computed by:

$$S_i(q) = -\sum_{r=1}^{m_i} q(X_i^{[r]}) \cdot ln(q(X_i^{[r]}) \qquad (5)$$
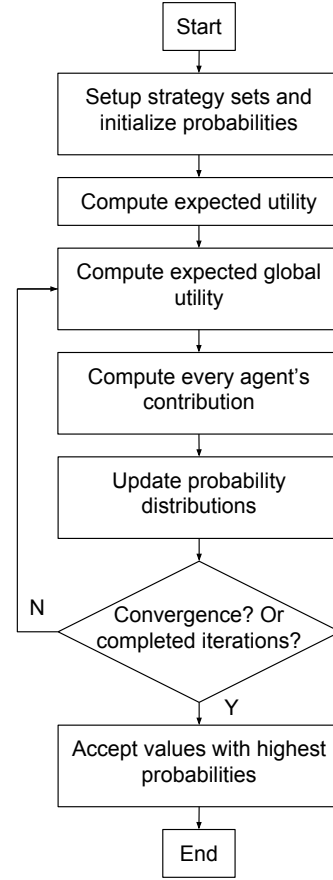

Figure 3: Probability Collectives Algorithm.

Which is the sum of every strategy $q(X_i^{[r]})$ times the natural logarithm of itself. Here, the algorithm checks to see if it will continue updating the global utility or it will terminate. If the probabilities of the combined strategy set have not changed by a constant amount, σ, or if the number of iterations (k) have reached a maximum, the strategies with the highest probabilities will be returned. Otherwise, the algorithm will continue updating the combined strategy set.

To summarize, if a strategy r creates a larger contribution to the optimization of the objective than other strategies, the probability associated with r increases by a larger amount. This entire process is repeated until the probability distribution converges or the maximum number of iterations are completed.

5. CONCLUSION

The vehicle routing problem is an important problem in real world logistics, and as in the field of optimization. The most performant algorithm in RVRP is hybrid genetic search with advanced diversity control, which is dominant within the scope of VRPTW. In decentralized RVRP variants, the specific constraints of each problem limit the comparability of each algorithm. The distributed reverse vickrey algorithm is within several percent of the best known solutions. Probability collectives is not as good in a straightforward sense, but it handles noisy and imperfect agent behaviors better

than other distributed systems. Adding human intuition to optimization systems may be the next major step in improving routing systems. Henry: (...)

6. ACKNOWLEDGEMENTS

7. REFERENCES

[1] S. Amaran, N. V. Sahinidis, B. Sharda, and S. J. Bury. Simulation optimization: a review of algorithms and applications. *4OR*, 12(4):301–333, 2014.

[2] S. Ben Ismail, F. Legras, and G. Coppin. A new interactive evolutionary algorithm for the vehicle routing problem. In *Proceedings of the 14th Annual Conference Companion on Genetic and Evolutionary Computation*, GECCO '12, pages 661–668, New York, NY, USA, 2012. ACM. *Fascinating idea that explores a best-of-both worlds approach.*

[3] J. Caceres-Cruz, P. Arias, D. Guimarans, D. Riera, and A. A. Juan. Rich vehicle routing problem: Survey. *ACM Comput. Surv.*, 47(2):32:1–32:28, Dec. 2014. *This paper is /the/ summary of the field at the moment. It supports the background and describes the major techinques in use in the field.*

[4] G. B. Dantzig and J. H. Ramser. The truck dispatching problem. *Management Science*, 6(1):80–91, 1959. *This paper presents the original problem and context.*

[5] A. J. Kulkarni and K. Tai. Probability collectives for decentralized, distributed optimization: A collective intelligence approach. In *2008 IEEE International Conference on Systems, Man and Cybernetics*, pages 1271–1275, Oct 2008.

[6] M. Saleh, A. Soeanu, S. Ray, M. Debbabi, J. Berger, and A. Boukhtouta. Mechanism design for decentralized vehicle routing problem. In *Proceedings of the 27th Annual ACM Symposium on Applied Computing*, SAC '12, pages 749–754, New York, NY, USA, 2012. ACM. *decentralization.*

[7] M. Vasirani and S. Ossowski. Decentralized coordination strategies for the vehicle routing problem. In *Proceedings of the 2008 ACM Symposium on Applied Computing*, SAC '08, pages 130–131, New York, NY, USA, 2008. ACM. *Unique focus on total decentralization.*

[8] T. Vidal, T. G. Crainic, M. Gendreau, and C. Prins. A hybrid genetic algorithm with adaptive diversity management for a large class of vehicle routing problems with time-windows. *Computers and Operations Research*, 40(1):475 – 489, 2013.

Henry: So, clearly this is clearly very short: Earlier text blocking lead to a mistaken conclusion that the paper would be hitting 6 pages quickly. It also proved easier than expected to summarize some of the methods than expected. Nic McPhee says I should add more on results and I agree.