ServicePower White Paper

Travel in a Field Service Environment

03/13/02

Draft One

Field service environments are often extremely dynamic. The mix of work is unpredictable, the availability of resources to complete that work is variable, and the business drivers that determine what makes a good schedule are subject to change. In such an environment, where the only constant is change, the complexity of the scheduling problem is immense. However, the reward for solving the problem is significant—particularly if production of high quality, optimized schedules can be automated. A high quality schedule maximizes the productivity of the field force while delivering a defined quality of service. The business return from an intelligent scheduling project is easy to measure, high profile, and delivers a fast payback.

The quality of the schedule depends on many critical factors including:

- How well you know the true field engineering capacity and availability
- Correct recording and matching of skills, competencies, spares holdings, etc.
- How well you can predict job duration
- How well you can predict travel times
- How well you cope with change—how fast you can re-plan
- The ability to consider all of the resources, jobs, and constraints after each change.

Travel is key to the last three points, particularly with regard to re-planning. Building a good plan is one thing, keeping it up-to-date minute-by-minute during the day, as jobs vary from plan and more work comes in, is absolutely vital to any same-day reactive service operation.

In order to maintain the best schedule, you still need to know all these things but be able to re-schedule in as short a time as possible. Expected travel times are fundamental to the quality of the schedule, and also to the ease with which the schedule quality may be maintained.

ServicePower's research and development team has worked on advanced scheduling techniques since the late 1980's. During that time the company has considered many methods of predicting travel times in a field service environment. The broad categories of methods to consider are:

1. Fixed travel times
2. Line-of-sight travel
3. Real-time calculated travel
4. Pre-calculated and stored travel

Method 1: Fixed Travel Times

This works best in an environment where the proportion of travel time to work time is very low, (i.e., a few jobs per day) and scope for optimization is low (i.e., low engineer density). However, when this is not the case or where you have a reasonable level of same-day change, using fixed travel times will produce inaccurate schedules.

Method 2: Line-of-Sight Travel

This method uses trigonometry to calculate the straight-line distance between two points. An average drive speed is then assumed and the distance and speed are used to derive a *drive time estimate*. Such travel times are simple and fast to calculate. However, this method was discarded because line-of-sight (or "as the crow flies") travel, which doesn't go around corners, rivers, lakes, or mountains, does not allow for natural obstacles. Another important factor is that it does not reflect the availability of roads in a true road network. A guess based on straight-line distance has much potential for inaccuracy and therefore has to be discarded.

Method 3: Real-time Calculated Travel

This method, sometimes called *street level routing*, may initially appear attractive since it offers drive time predictions based on true door-to-door travel. The popular Web-based journey time calculators use this method (e.g., MapQuest, Yahoo Maps, etc.).

These systems are based on a digitized representation of the road network. To calculate an approximate drive time, these solutions first plot the fastest route from point A to point B (the start and end street locations fed into them). These solutions then calculate an average speed for each type of road segment (interstate, local highway, community road, etc.), to calculate an approximate average elapsed time to drive from point A to point B.

On further study, however, ServicePower found that using real-time travel calculations as part of an optimizing scheduler presented a number of severe limitations:

a. **Performance vs Accuracy** — Because of the amount of computer processing involved (plotting the route, determining road segment speeds, calculating elapsed drive time), the time required to deliver a single drive time prevents an automated scheduler from trying more than a very limited number of combinations of assigning jobs to technicians. In the real world of field service, only by trying hundreds of thousands of different combinations can you produce an optimal schedule. Consider the following illustration:

A customer phones into an organization's customer call center to request a service appointment In this example, we can assume that there are only six engineers who can be considered reasonable candidates to complete this service appointment.

Lets also assume that the organization is currently experiencing a high workload, and each engineer is completing an average of five jobs per day. So, six engineers completing five jobs per day means that the organization is dealing with thirty jobs per day in total. The next step is to look forward over a five day window to find an appointment offer for this customer, this gives a total of 30*5=150 jobs in the scheduling horizon. In order to find the best appointment offer, if only travel time were important, it would be necessary to look at what the travel time would be if the job were inserted into the schedule before any of the 150 other jobs. However inserting this job involves calculating not just one but two travel times—how long it takes to travel to this job and how long it takes to travel to the next appointment in the schedule. Therefore just looking at six engineers over five days requires a calculation involving 150*2 = 300 travel times.

If, for example, calculating travel times by street level routing takes as little as 0.25 seconds per lookup. In order to get an offer, it would take 300*0.25 = 75 seconds looking up travel times alone. However, in a field service environment, travel is only one of a number (typically ten to thirty) of constraints which must be considered to produce an optimal schedule.

Therefore, the only way to retrieve an appointment offer within a reasonable time frame using street level routing is by considering far fewer options. This results in a less efficient and less accurate schedule with less optimized travel. It also reduces the likelihood of finding the nearest engineer every time.

b. **Accuracy of estimate vs real-life** — It can be more accurate than life—taking no account of the unpredictability of traffic lights, the need to stop for gasoline, and the hundred and one other things that make the journey time vary from its predicted length. The extra time (in processing) and cost (in hardware size and processor speed) needed to look up that travel time could not be justified.

c. **Accuracy vs real-life time-of-day** — If the travel is not adjusted by type of area and time-of-day, then for the majority of the day it is hopelessly inaccurate. If you have ever used a MapQuest type system you will understand that the journey time returned is an average, which does a poor job of handling the variability caused by rush-hour traffic loads.

Method 4: Pre-Calculated and Stored Travel

Given our experience with the other methods, ServicePower found that pre-calculated and stored travel delivered the highest quality schedules in real field service environments. It provides the best compromise between accuracy of forecast travel time and speed of calculation. Speed of calculation is essential if you are to solve the big problem of optimizing not just one journey but the entire enterprise-wide schedule. Here's how it works:

d. The geography is first broken down into a number of regions. The size of the region can be selected to ensure the right level of forecast accuracy. For example, in the U.S., regions can be defined by five or nine digit ZIP CODES.

e. The location calculated to give the shortest travel times to all other residences within the region is then selected. This is known as the population weighted centroid. The population centroid is used rather than the geographical centroid as it further improves the accuracy of the method.

f. A large matrix is then entered into the system, pre-loaded with the real average drive times between the centroids of each region. These drive times are calculated using the street level routing methodology described in method 3. They take into account the real road network and the differing average speeds of each type of road segment.

g. The matrix is then stored in memory. An average intra-region travel time is also calculated and stored in memory. In order to retrieve a travel time, the scheduling engine indexes the matrix using the region code of the start and end locations via an in-memory data access. This is much faster than a real-time calculation involving several disk accesses and complex mathematical calculations.

h. Once the average drive time has been retrieved, it is weighted via a travel profile that can be used to factor in the effects of traffic congestion and time-of-day variability. The result is an accurate estimate of the travel time required to get between two locations taking into account time-of-day—all delivered at the speed of an in-memory access. The benefits of this approach include:

   i) Appointment and job booking can consider many more options, including moving existing lower priority jobs aside to make way for new work. This is because travel can be calculated so quickly, and as a consequence, the quality of job booking is greatly improved.

   ii) As the day progresses, things change in the field (new jobs are received, existing jobs are cancelled, etc.). As a result, it is necessary to frequently optimize the schedule to bring it back to its lowest cost and optimal level. Using *simulated annealing* (SA), SERVICE***Power*** can try 100,000 to 200,000 transformations of the schedule in 1 minute, finding 100 possible new schedules in its search for the lowest cost, highest quality, revised schedule.

iii) As shown in method 3 (a), this involves making 200,000 to 400,000 travel calculations. Using in-memory, pre-stored travel this can be achieved. Using just street level routing, calculating the travel time alone would take between 13.8 hours to 27.6 hours! Overall, with pre-stored travel, the quality of the schedule is maintained at a much higher level.

The key remaining imperative is to factor the travel time according to the congestion factor of the area and the time-of-day. As there are no constraints from external lookups (as there are with method 3), SERVICE***Power*** can do this and provide the best estimate for the length of each journey.

Furthermore, the travel time factors can be set and maintained by the user. SERVICE***Power*** supplies the base travel data and the customer modifies it to suit their business model and the way the territories are covered.

The artificial intelligence (AI) techniques used by SERVICE***Power*** allow the system to balance competing factors within the schedule; for instance, what is preferred (and so least cost in scheduling terms), always minimizing travel, or balancing travel against overtime and many other factors.

Ultimately, the main requirement for most organizations, is the need to produce optimal work schedules across the whole of the enterprise. This leads to the realization that optimal service schedules actually require many constraints in order to be properly balanced. These include: travel time, call priority, overtime working policy, service level agreement, customer priority, etc. Often these constraints are in conflict with each other—under some circumstances you do want an engineer to travel farther because you know he has the skill required to fix this particular, high priority customer's problem. The beauty of SERVICE***Power***'s AI approach to scheduling, compared to the less flexible rules based approach, is that AI mirrors the flexibility of the human mind.

Travel time is, of course, important. However, the real goal is to minimize the drive time of your entire field force while still meeting your service goals at reasonable cost. The goal is not to minimize just one engineer's journey time on one particular job. With acknowledgement of this fact comes the realization that it is the relative journey times between different routes around work that is really important, not the accuracy of any one particular journey time.

During implementation we determine the *resolution* of travel matrix required by your business.

In service operations with a small number of engineers in each geography and a good geographical spread of work, we find that resolving travel to a region delineated by a five-digit ZIP CODE delivers an excellent quality schedule. In areas where there is a higher density of work and/or engineers, resolving regions down to a nine-digit ZIP CODE may produce better schedules. The key point is to determine how often there is more than one job to be serviced in the same place within the same time frame. Usually it isn't that often.

The advantage of this technique is speed and scalability. Using clever, sparse-matrix techniques the whole travel matrix can be stored in memory—no need to go across a network, complete a complex calculation, or query a database. The average travel time from the travel matrix is now factored using a travel profile to take into account time-of-day.

This gives us suitable, accurate travel times—correct for the time-of-day and accessible very quickly.

In summary, the real differentiator between SERVICE***Power*** and competing technologies is the fact that SERVICE***Power***'s intelligent algorithm, (intelligent because our SA algorithm knows the best places to look for good solutions) searches hundreds of thousands of different combinations of assigning work to engineers and calculating their routes. SERVICE***Power*** compares each against the other to minimize the drive time of the entire service organization, which is vital for maximizing an organization's profit contribution.