Dynamic VRPs: A Study of Scenarios

Philip Kilby [*] Patrick Prosser [†] Paul Shaw [‡]

Report APES-06-1998 [§], September 1998[¶]

Abstract

Many day-to-day routing problems are *dynamic* in nature: new orders are received as time progresses and must be incorporated into an evolving schedule. Taxis and parcel delivery couriers are examples. However, there is very little work available on these problems in the Operations Research literature.

This paper is a look at some of the issues raised by dynamic problems. We introduce some benchmark data derived from classic VRP benchmarks, and discuss some of the complications in creating this data.

We look how the cost of the system is affected by the degree of dynamicity, defined here to be the proportion of the parcels which are known beforehand. We look also at the cost of maintaining a *commit horizon* where the schedule is fixed ahead of time for communications or other business reasons.

1 Introduction

In the Vehicle Routing Problem, a fleet of vehicles must be routed to visit a set of customers at minimum cost. In the classical VRP all orders for all customers are known *a priori*. However, many routing problems in the real world include at least an element of dynamicity, where the schedule must change in response to new or altered requests. We wish to examine here some of the issues raised by dynamic problems.

In the type of problem we wish to consider, there are a number of customers requiring a visit from a vehicle. Some of the visits are specified *a priori*, and an initial schedule is created covering these visits. As the day progresses, new tasks (either pickups or deliveries) enter the system and must be incorporated into the evolving schedule.

In the model we consider, there is a "tentative schedule" which incorporates all work currently known. This schedule is adjusted as new work arrives, and can be improved providing this does not interfere with decisions that have already been committed to. It is assumed that there is a communication system between the dispatcher and drivers and that the dispatcher tells the driver which visit to serve next only when committing to that decision.

[*]CSIRO Mathematical and Information Sciences, GPO Box 664, Canberra ACT 2601, Australia
[†]University of Strathclyde, 26 Richmond Street, Glasgow, Scotland
[‡]ILOG, BP 85, 9 rue de Verdun, 94253 Gentilly Cedex, Paris
[§]Available from `http://www.cs.strath.ac.uk/□ apes/apereports.html`
[¶]This work was performed while Paul Shaw and Philip Kilby were at the University of Strathclyde.

The usual constraints which appear in the standard vehicle routing problem can also appear, for example the constraint that the load on a vehicle must not exceed the vehicle capacity. Other constraints such as a time window on when the service may begin may also be present. For a description of the standard problem, please see the recent survey of the routing problems by Ball *et al.* [1].

These types of problem have been studied only sporadically in the literature. A recent survey is given by Psaraftis [13], and we refer the reader to that work for a full review. In addition Bertsimas and Simchi-Levi [2] examine the worst and average-case behaviours for some dynamic routing problems.

One of the problems with presentation of this type of problem is that business rules governing operations have the ability to completely change the face of a problem. These changes are much more radical than for equivalent *static* (everything known beforehand) problems. Here are some examples.

Pickup versus delivery. In static problems, pickup problems (picking up goods at the customer) and delivery problems (delivering from a depot to the customer) can be modelled as equivalent. However, in dynamic problems it is physically impossible to add a delivery to a vehicle after it has left the depot. Pickups, on the other hand, can be added to a existing vehicle schedule at almost any time. Solutions for the two problems will look completely different.

When to commit. At some stage in the execution of a schedule, a commitment must be made to serve a customer at a particular time. Exactly when this commitment is made is one of the fundamental questions of dynamic routing. In a "latest commitment" policy, this commitment is left until the last possible moment. In execution, this would require each driver to communicate with a central dispatcher after the completion of each task in order to obtain the next piece of work.

In some scenarios, it may be necessary to commit to a schedule ahead of the latest commit time. We use the term *commit horizon* to describe the extra period the schedule is fixed. For example, a 10-minute commit horizon would allow the commitment to a particular task to be made 10 minutes before the task was due, in order to give the dispatcher and driver time to communicate. The cost of this safety margin is the improvements that an on-line system might have made in the final 10 minutes. Less sophisticated communication systems usually require longer commit horizons.

Work known *a priori*. In most cases, some tasks will be known beforehand. How much of the day's work that falls into this category is one measure of the dynamicity of the problem.

Incorporation of improvements. In a model of a dynamic operation which includes an operator who add new tasks as they become available, the continuous improvement algorithm can be seen as an operator who is also making changes. However business rules impact here as well. In some dynamic situations it may be acceptable for telephone operators to be "shut out" for say 30 seconds while an improvement process runs, but other cases may require response to the telephone operator within 2 seconds. On the other hand, if the improvement process must constantly test for and incorporate changes to the schedule made by the operator, its efficiency will be degraded. We will see that the extended improvement process is one of the primary advantages of the dynamic system, so any trade-offs in the performance of the improvement process can impact on overall cost.

The rate of arrival is also important. If new requests arrive in widely separated bursts, there is more time for an improvement phase to operate unhindered, whereas a constant trickle will require many more restarts.

Use of historical information. Historical information can be used to improve the performance of algorithms for solving dynamic problems. Expected quantities and times of orders can be used to help design routes. However, this issue is not addressed in this paper.

Since each of these considerations have a sliding scale of possible values, each of which make sense in different business settings, it is very difficult to study a "typical" scenario. We have chosen to study a particular scenario because it represents what we feel is the smallest interesting step away from the static case. It is close to the traditional VRP, but includes a (scalable) dynamic element. It is also a "real-world" problem in that it models a particular scenario that can be found in the real world.

We look at pickup and delivery problems separately - that is, in a particular instance, either all the visits are pickups or all the visits are deliveries.

Common to both pickup and delivery problems, there is a single depot which has set opening hours. No vehicle can leave the depot before opening time, and all vehicles must return to the depot by the closing time.

The pickup scenario is as follows. There are a certain number of parcels left over from the previous day. These form the initial schedule. Customers ring up to request a pickup. The time of their call is the "available time" of the visit. If the call comes before a particular cut-off time, then the visit is incorporated into the current day's schedule. Otherwise, it is picked up the next day (i.e. it becomes one of the *a priori* visits for the next day).

The delivery scenario is very similar. Parcels arrive at the depot throughout the day. The time of arrival at the depot is the "available time". Parcels that arrive before the cutoff time are included in the current day's schedule. Parcels arriving later are held over for delivery the next day.

Dispatching decisions cannot be pre-empted. That is, once the dispatcher commits to a decision (by telling the driver which visit to serve next) that decision cannot be changed.

The algorithm used for the construction of routes here is the basic "insert and improve" algorithm. As each task becomes available, it is incorporated into the schedule in the cheapest place. Local search is used throughout the working day which constantly seeks to improve the schedule. Improvements are incorporated as they are discovered. This is discussed in more detail in section 2.

In the next section we look at the solution methods adopted for this study. We then describe the data developed for studying dynamic problems. Finally, we present some results that compare the effect of varying some of the parameters discussed above.

2 Solution method

The methods used to solved the dynamic problem here are based on an "insert-and-improve" method. As each visit becomes available it is inserted in the best position. The schedule is continuously improved during the day.

The main advantage in solving a problem dynamically is that there is a lot of time to devote to improvement. In the small problems we are solving here (mostly around

100 visits) many improvements can be tested between the arrival of new orders. So whereas in a static case 10 minutes would seem like a long time to spend finding a solution to a 100 node problem, we have all day so we might as well use it.

A significant disadvantage of dynamic routing is that many of the recent advances in routing techniques which rely on meta-heuristics ([7, 8, 5, 9]) may not be applicable. Meta-heuristics allow an algorithm to find its way out of a local minimum by accepting a number of cost-increasing changes before returning to a (hopefully lower) local minimum. In the dynamic case, however, it would be inappropriate to accept any cost-increasing moves, lest the schedule be locked in to a high-cost solution by a commit at an inopportune time. The problems with incorporation of improvements discussed in section 1 mean that it may be impossible to lock the solution while a meta-heuristic is run to the point where it finds a new local minimum. We use no meta-heuristics in this paper.

The method used to find a solution is to first find an initial schedule which incorporates all visits known *a-priori*. Each dynamic visit is then added to the schedule, as it becomes available, by inserting it in the legal position which increases the cost by the least amount.

2.1 Initial schedule

The initial solution is effectively a standard VRP and can use any standard method. In this paper we use a method which (like Caseau [3]) inserts a number of visits and improves before continuing. This resembles the dynamic procedure quite strongly and has been shown to be an effective technique.

For the problems presented in section 5 we insert 10 visits at each iteration, and improve using a first accept local search incorporating 2-Opt [10], Or-opt [11], relocate [14], exchange [15] and cross [12].

We have found that the order of insertion has a large influence on the effectiveness of this and other insertion-based heuristics. We use insertion in the order given by a 2-optimal Travelling Salesman Tour around all visits. That is, we find a tour which ignores all constraints, but simply covers all visits with a single vehicle using travel distance as the objective function. This tour is then improved using 2-opt. The visits known *a-priori* are inserted into the initial plan in the order they appear in this travelling salesman tour[1].

2.2 Improvement

In section 1 we discussed the problems of incorporating improvements into a schedule. For the purposes of this paper, we have adopted a fairly simple scheme. We use the concept of a time step. The improvement is allowed to run unhindered for one time-step. Any improvements that it has found during that time are then incorporated into the schedule. Any new work which has become available while the improvement process was running is added in at the least expensive position. Any decisions which will trigger during the next time step are then fixed so that they cannot be altered by the improve process. The cycle then repeats with a new improvement phase.

[1] Just for clarity we repeat that the remaining dynamic visits are inserted in the order that they become available.

For example, say the current time is 10:00am, and we are using a time-step of 5 minutes. We first of all incorporate all new work that became known since 9:55. Say the current schedule requires vehicle 1 to be dispatched to visit 50 at 10:02. Then that assignment is fixed, so that while the improvement is running in the next time step, vehicle 1 can be dispatched on time without the possibility of the schedule changing.

The size of time-step has an effect on the solution. A long time-step means commitments are made further into the future, and so there is less scope for improvement. On the other hand it also means that a meta-heuristic might be able to be used more effectively.

In the tests run in section 5 a time-step was chosen so that there were about 50 time-steps during the working day.

3 Data

Benchmark dynamic datasets have been created using the VRP benchmark datasets of Taillard [15] (13 files), Christophides [4] (7 files) and Fisher [6] (2 files). The numbers of visits varies between 50 and 385. Our objective was to design a set of data which were the dynamic equivalent of a set of VRP data. Therefore none of the datasets have time windows.

We wish to test pickup and delivery systems separately, and so data-files were created for each case. There are therefore two versions of each of the original data-files. The data for pickup and delivery cases of each file differ only in the sign on the "DEMAND" field for each customer. A pickup is signified by a positive demand, a delivery by a negative demand.

We need to add three types of data to the basic VRP data in the original benchmarks. First an *available time* marking when the pickup or delivery first entered the system. We assume that nothing is known about the task until its available time. We have used a uniform random distribution to generate available times for each task. The times are spread throughout the working day (see below).

Second, a *duration* for each task is required. The duration was chosen so that approximately the same time would be spent performing tasks as travelling between the tasks. This time is needed to "pad out" the day for small problems, or else the whole day's work can be completed in just a small part of the afternoon.

Finally, a concept of the *working day* is required. We chose the working day to be eight times the maximum distance between pairs of visits. This fairly arbitrary figure was chosen as an estimate of travel times in a moderate sized city. However, the working day and durations must be chosen with complimentary values so that the expected time spent performing tasks plus travel time is roughly a working day.

The objective function (cost) for each problem is the total distance travelled (regardless of the number of vehicles required).

The data are available the APES web page:

`http://www.cs.strath.ac.uk/ apes/apedata.html`

along with a description of the datafile format.

4 Scheduling Rules

All during the day, the system maintains a “tentative schedule” - the schedule as it would be implemented if everything were to be frozen. However the tentative schedule is always changing. New tasks are added in as the day progresses, and the system is also making changes as improvements are found by the continuous improvement process.

Of the many possibilities, we tested only one scheduling rule: a modified latest dispatch rule. The rule assumes the presence of time windows at customers and at the depot (signifying the working day). The system does not commit to a decision until the last possible moment. This means that the system does not commit to the next job until the previous job has been completed. Even then, if the vehicle would arrive before the early time window of the next job, then the system does not commit to that decision. The vehicle waits at the location of its previous task until either a) the tentative schedule changes and it is able to perform its next task immediately, or b) the time window opens on its current next task. As soon as it is able to perform its next task, the system commits that decision, and the vehicle is dispatched.

In the case studied here where visits do not have time windows, the dispatch time is governed by the depot time window (*working day*). The vehicle must be dispatched in time to make its visits and return to depot before the deopt closes.

A slightly different rule is used for the last visit - the return to depot. If the usual rule were used, as soon as a vehicle had performed the last visit on its current tentative schedule it would return home. However, that does not allow for more visits becoming available in its area. Therefore a different “return home” rule is used: don't return to depot until the end of the work day.

5 Computational Experiments

Tests were performed to investigate the effect of three parameters discussed in section 1: the differences between pickup and delivery scenarios, the degree of dynamicity, and the length of the commit horizon.

To test the effect of changing the degree of dynamicity, we use a sliding *cutoff time*. All tasks have an associated “available time”. Any visits with an arrival time after the cutoff time are treated as if they arrived yesterday. As described in section 2, an initial schedule is found that serves all of these visits. The visits with arrival times before the cut-off are then introduced in order of their arrival time.

The cutoff time was expressed as a fraction of the working day in order to compare data with different working days.

We also test the effect of different lengths of commit horizon. For a horizon of n time units, all decision on dispatching vehicles and choosing the next visit were fixed n time units ahead of last possible moment. So a horizon of zero minutes means all decisions are committed to at the last possible moment - the most dynamic case. Again, the horizon is expressed as a fraction of the working day.

We have already mentioned the way time-steps are used in the simulation (section 2.2). The time step is chosen so that about 50 time-steps are needed to cover the working day. The improvement process is run during each time step. In the runs reported here, 30 seconds CPU time was used for improvement in each time step. Because it is not possible to check all possible moves in the time available, a randomisation process was used to select which moves to test during the improvement phase.

5.1 Results

The results are presented in figures 1 and 2. In both figures, the actual data is shown as a background to the "overall" line which shows the averaged data. Both figures also have the same Y axis: the percentage rise in objective compared to the "static" case at the 0 point of the X axis.

Figure 1 presents the results for varying dynamicity. The horizontal axis can be thought of as the fraction of the day that the company continues to accept orders for the same day. Thus the 0 point is a completely static scenario - all order are known beforehand. At the 0.5 point, half the orders are known from the previous day, and fresh orders are taken only for the first half of the day.

First, comparing the pickup and delivery graphs, it is clear that the cost of deliveries grows quicker with dynamicity than the equivalent pickup scenario. For the scenarios examined here it is around twice as expensive to deliver as to pickup. We conjecture that this is primarily because of the reduced opportunity for rerouting. In a pickup scenario, the dispatcher (human or computer) is free to change the tentative schedule of each vehicle right up until the time it is sent to its next pickup point. The schedule can thus be changed in response to new orders. In a delivery scenario, the goods are already loaded onto the vehicle when it leaves the depot. Therefore, only the delivery order can change once the vehicle has left the depot.

The most costly delivery scenarios tended to be those with early dispatch times - those that are most affected by inability to change the tentative schedule after the vehicle is dispatched.

Figure 2 shows the effect of varying the length of the commit horizon. The X axis shows the length of the commit horizon, expressed as a fraction of the working day. The cutoff time is fixed at 0.6 for all these problems, that is 40% of the work is known *a-priori* and 60% arrives dynamically.

The first thing to note for both graphs is that the maximum commit horizon shown is one quarter of a day: longer horizons very quickly become unworkable. You can still see many of the background data lines stopping. this indicates that for higher values of the commit horizon, not all tasks could be performed with the available fleet.

Comparing the graphs for pickup and delivery we see that there is less difference between costs when varying this parameter: at 0.25 of a day the cost of deliveries has risen by 45% compared with 30% for pickups.

The cost of deliveries is rising almost linearly - for every 0.1 part of a day "safety margin" ($\frac{3}{4}$ hour in an 8-hour day) the costs rise by about 17%.

The pickup case is similar, with a rise of approximately 10% for every 0.1 of a day additional commit time.

6 Conclusions

We have examined some of the problems faced when simulating and optimizing dynamic vehicle routing problems, including some of the many business model choices that must be made when operating within a dynamic vehicle routing setting.

We have chosen two particular scenarios for further study - a pickup scenario and a delivery scenario. Using base data from a number of classic vehicle routing benchmarks, we have defined a set of dynamic problems, which, while being dynamic, represent only a small but interesting step away from the static vehicle routing problem.

We have introduced two measures of the dynamicity of a problem: the proportion of the visits known *a-priori*, and the *commit horizon*, and performed a series of tests to examine the effect of these two parameters. We have seen that these two parameters affect the delivery and pickup scenarios in quite different ways.

This work can be seen as a beginning. There are many questions left to investigate. Some of the more interesting ones are:

- what is the effect of other dispatching criteria used in business today
- how can historical information on customer orders be used to inform a dynamic model of expected demand
- what other parameters affect the cost of dynamic routing, and to what degree.

Acknowledgment

The production of this paper was supported by the GreenTrip project, a research and development undertaking partially funded by the ESPRIT Programme of the Commission of the European Union as project number 20603. The partners in this project are Pirelli (I), ILOG (F), SINTEF (N), Tollpost-Globe (N), and University of Strathclyde (UK).

References

[1] M. O. Ball, T. L. Magnanti, C. L. Monma, and G. L. Nemhauser, editors. *Network Routing*, volume 8 of *Handbooks in Operations Research and Management Science*. Elsevier, Amsterdam, 1995.

[2] Dimitris J. Bertsimas and David Simchi-Levi. A new generation of vehicle routing research: Robust algorithms, addressing uncertainty. *Operations Research*, 44(2):286–303, 1996.

[3] Yves Caseau. Heuristics for large constrained vehicle routing problems. Presented at INFORMS conference Montréal, April 1998., April 1998.

[4] N. Christophides and J. Beasley. The period routing problem. *Networks*, 14(237-256), 1984.

[5] Kathryn A Dowsland. Simulated annealing. In Colin R Reeves, editor, *Modern Heuristic Techniques for Combinatorial Problems*, chapter 2. Blackwell Scientific Press, Oxford, UK, 1993.

[6] M. Fisher, R. Jaikumar, and L. van Wassenhove. A generalized assignment heuristic for vehicle routing. *Networks*, 11:109–124, 1981.

[7] F. Glover. Tabu search, part I. *ORSA Journal on Computing*, 1(3):190–206, 1989.

[8] F. Glover. Tabu search, part II. *ORSA Journal on Computing*, 2(1):4–32, 1990.

[9] P. Kilby, P. Prosser, and P. Shaw. Guided local search for the vehicle routing problem. In *Proceedings of the 2nd International Conference on Meta-heuristics*, 1997.

[10] S. Lin. Computer solutions of the traveling salesman problem. *Bell Systems Technical Journal*, 44:2245–2269, 1965.

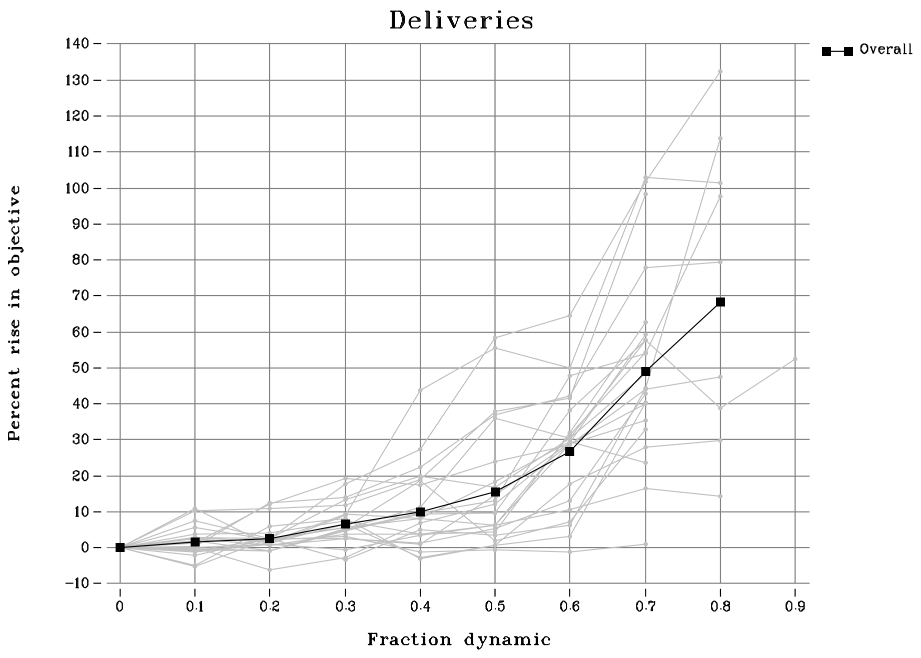
[11] I. Or. *Travelling Salesman-Type Combinatorial Problems and Their Relation to the Logistics of Blood-Banking*. PhD thesis, Department of Industrial Engineering and Management Sciences, Northwest University, Evanston, IL., 1976.

[12] J.-Y. Potvin and J.-M. Rousseau. An exchange heuristic for routing problems with time windows. *Journal of the Operations Research Society*, 46:1433–1446, 1995.

[13] H. Psaraftis. Dynamic vehicle routing: Status and prospects. *Annals of Operations Research*, 61:143–164, 1995.

[14] M. W. P. Savelsbergh. Computer aided routing. Centrum voor Wiskunde en Informatica, Amsterdam, 1988.

[15] E Taillard. Parallel iterative search methods for vehicle-routing problems. *NETWORKS*, 23(8):661–673, 1994.



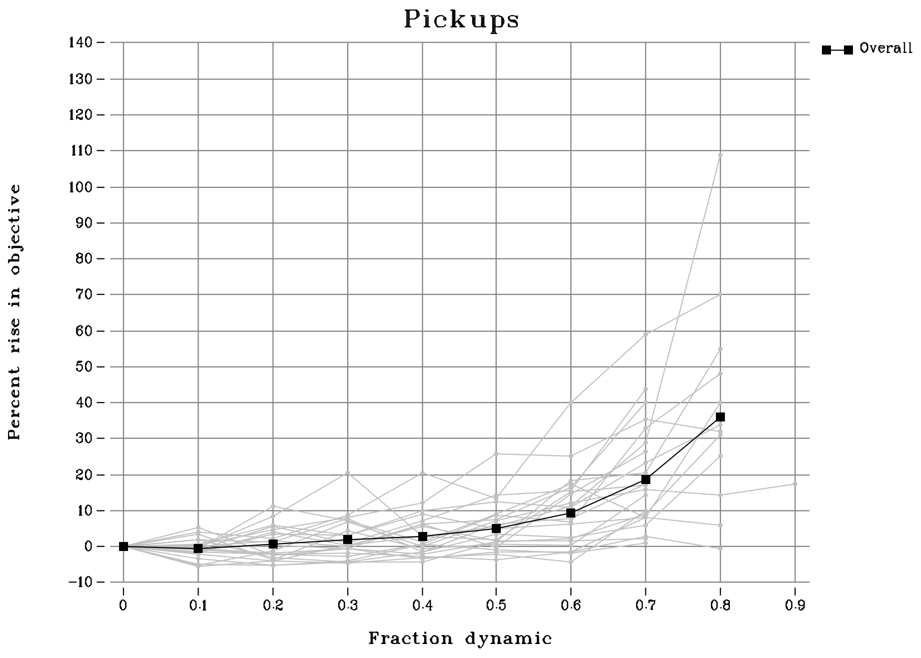
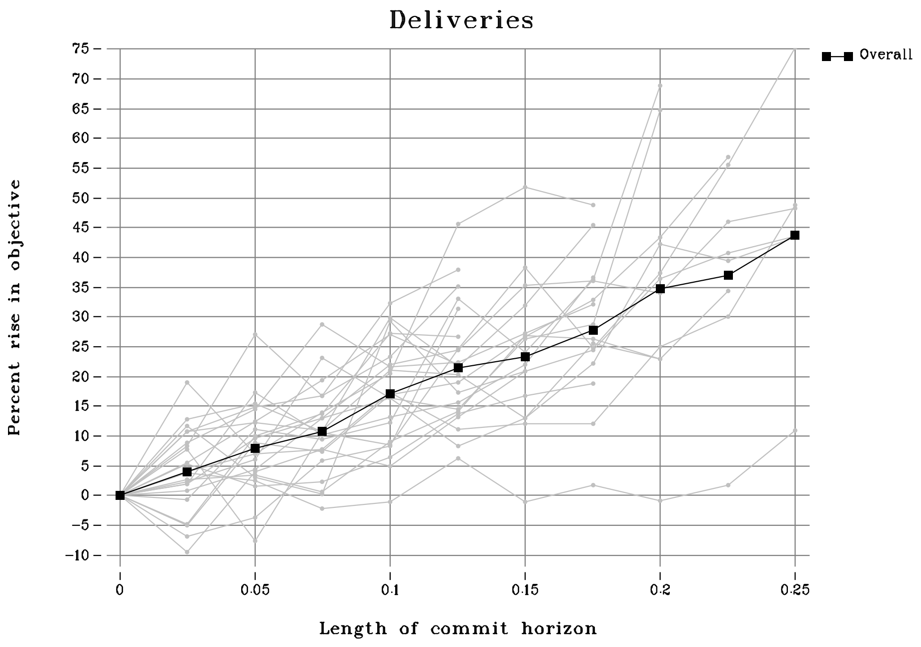


Figure 1: Varying dynamicity



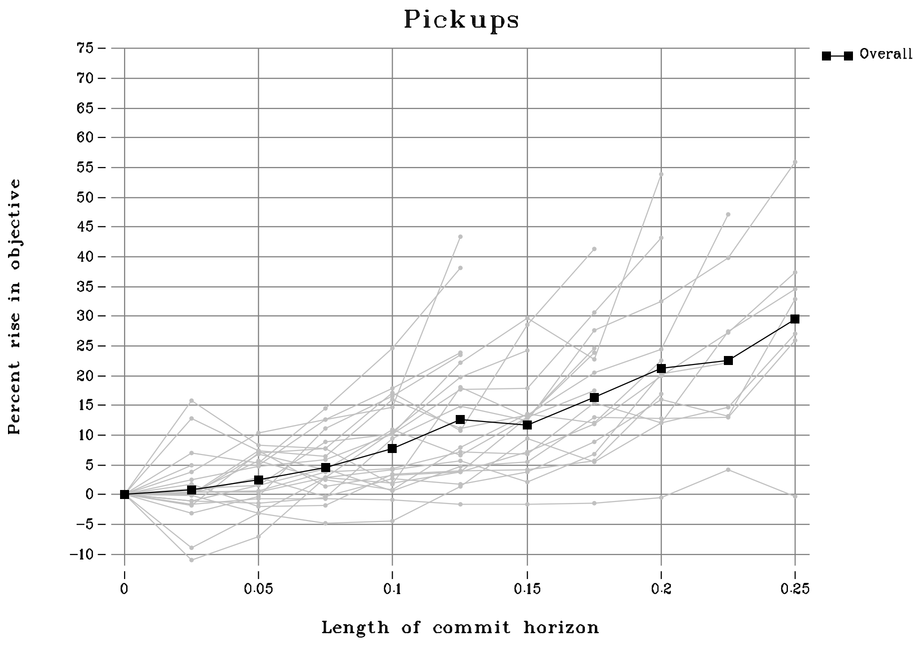


Figure 2: Varying commit horizon (as a fraction of the working day)